



# **MINI PROJECT**

## **A REPORT ON**

# **Build a log Analytics Solution on AWS**

Team Member:

Adesh Kumar(171500015)

Ayush Kumar Singh(17150075)

Ishan Rathi(171500144)

Project Guide:

Dr. Manoj Varshney

**Department of Computer Engineering and Applications**



**Declaration**

We, the students of fifth semester of Computer Science Engineering, GLA University, Mathura declare that the work entitled "Build a Log Analytic Solution on AWS" has been half completed under the guidance of Mr. Manoj Varshney, Department of Training and Placement of Computer Science Engineering. This dissertation work is submitted to GLA University in partial fulfilment of the requirement for gain credits of academics in Computer Engineering and Application during the academic year 2019-2020.

**Name of Team Members and Roll No:**

1. Adesh Kumar (171500015)
2. Ayush Kumar Singh (171500075)
3. Ishan Rathi (171500144)

**Course:** B. Tech

**Year:** 3rd

**Semester:** 5th

## **ACKNOWLEDGEMENT**

We have taken efforts in this project. However, it would not have been possible without the kind support and help of many individuals. On the completion of this project, we would like to extend our sincere thanks to all of them. We are highly indebted to this project guide DR. Manoj Varshney for their guidance and constant supervision as well as for providing necessary information regarding the project. We wish to extend our sincere gratitude to Prof. ANAND SINGH JALAL, Head of Department of Computer Engineering and Applications and faculty of CEA Department of GLA University for their guidance, encouragement and give this opportunity and valuable suggestion which prove extremely useful and helpful in the completion of this report. We would also like to thank all those who directly or indirectly supported or helped in completing this project in time. We would like to express our gratitude towards our parents and member of our college for their kind cooperation and encouragement which helped our in completion of this project. All of them have willingly helped out with their abilities.

Last but not least we are thankful to all faculty whose support at various stages, this project would not have materialized.

**Adesh Kumar (171500015)**

**Ayush Kumar Singh (171500075)**

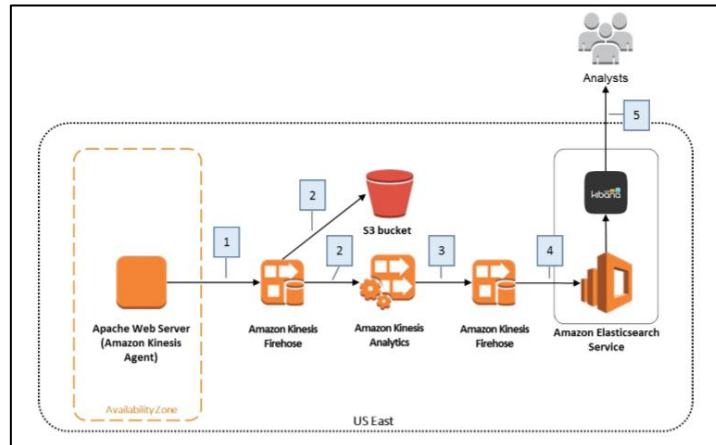
**Ishan Rathi (171500144)**

## Index

Abstract.....	5
Contents.....	5
1. Introduction.....	6
a) Project Description.....	6
b) About Functional Specifications.....	6
c) Hardware Requirement.....	7
d) Software Requirement.....	7
2. Problem Definition.....	7
3. Objective.....	7
4. Implementation Details.....	8
a) Step 1: Set Up Prerequisites.....	9
b) Step 2: Create an Amazon Kinesis Firehose Delivery Stream.....	11
c) Step 3: Install and Configure the Amazon Kinesis Agent.....	15
d) Step 4: Create an Amazon Elasticsearch Service Domain.....	14
e) Step 5: Create a Second Amazon Kinesis Firehose Delivery Stream.....	18
f) Step 6: Create an Amazon Kinesis Analytics Application.....	20
g) Step 7: View the Aggregated Streaming Data.....	23
h) Step 8: Clean Up.....	25
5. Bibliography.....	27

## ABSTRACT

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



The web server in this example will be an Amazon Elastic Compute Cloud (EC2) instance. You will install the Amazon Kinesis Agent on this Linux instance, and the agent will continuously forward log records to an Amazon Kinesis Firehose delivery stream (step 1). Amazon Kinesis Firehose will write each log record to Amazon Simple Storage Service (Amazon S3) for durable storage of the raw log data (step 2), and the Amazon Kinesis Analytics application will continuously run an SQL statement against the streaming input data (step 2). The Amazon Kinesis Analytics application will create an aggregated data set every minute and output that data to a second Firehose delivery stream (step 3). This Firehose delivery stream will write the aggregated data to an Amazon ES domain (step 4). Finally, you will create a view of the streaming data using Kibana to visualize the output of your system (step 5).

## 1. Introduction

### 1.1 Project Description

Amazon Kinesis 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 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 implementation 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 will create a complete end-to-end system that integrates several AWS services. You will 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 will use a user interface (UI) tool that will chart the results.

### 1.2 Functional Specifications

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

**Amazon Kinesis Firehose:** Amazon Kinesis Firehose is a fully managed service for delivering real-time streaming data to destinations such as Amazon S3, Amazon Redshift, or Amazon ES. With 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.

**Amazon S3:** 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.

**Amazon Kinesis Analytics:** Amazon Kinesis Analytics is the easiest way to process and analyse streaming data in real-time with ANSI standard SQL. It enables you to read data from Amazon Kinesis Streams and Amazon Kinesis Firehose, and build stream processing queries that filter, transform, and aggregate the data as it arrives.

**Amazon Elasticsearch Service:** Amazon ES is a popular open-source search and analytics engine for big data use cases such as log and click stream analysis. Amazon ES manages the capacity, scaling, patching, and administration of Elasticsearch clusters for you while giving you direct access to the Elasticsearch API.

## 1.3 HARDWARE REQUIREMENT

- 1.3.1 Processor: i3 Processor and above
- 1.3.2 Operating System: Windows
- 1.3.3 RAM: 4 GB and above
- 1.3.4 Hard Disk: 500 GB and above

## 1.4 SOFTWARE SPECIFICATION

- 1.4.1 Technology Implemented: AWS, Cloud Computing, Virtualization
- 1.4.2 Language Used: Sql
- 1.4.3 Web Browser: Any Browser

## 2. Problem Definition

Without log analysis tools, not only would you have to identify which server the log you need is on, but then you'd have to identify which log file it might be in. Once you figured that out, you would then be able to concentrate on getting into the file and searching for clues. However, simply opening the file can be a challenge in itself. A text file that is gigabytes in size can be considered *big data* for most desktops. Though the logs contain what you need, locating the server, identifying the log and searching for the data is simply too much to take on.

## 3. Objective

We want to track user activities on the networks, sensors, websites, etc for fraud detection, false surfing of users on above platforms, interruptions in between networks, etc. We can also take this project for E-commerce companies for customer analysis.

## 4. Implementation Details

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

You will perform the following steps:

- Step 1: Set Up Prerequisites
- Step 2: Create an Amazon Kinesis Firehose Delivery Stream
- Step 3: Install and Configure the Amazon Kinesis Agent
- Step 4: Create an Amazon Elasticsearch Service Domain
- Step 5: Create a Second Amazon Kinesis Firehose Delivery Stream
- Step 6: Create an Amazon Kinesis 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 Analytics application in the Additional Resources section.



## Step 1: Set Up Prerequisites

Before you begin analysing your Apache access logs with Amazon Kinesis Analytics, make sure you complete these prerequisites:

- Create an AWS Account
- Start an EC2 Instance
- Prepare Your Log Files

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

### Create an AWS Account

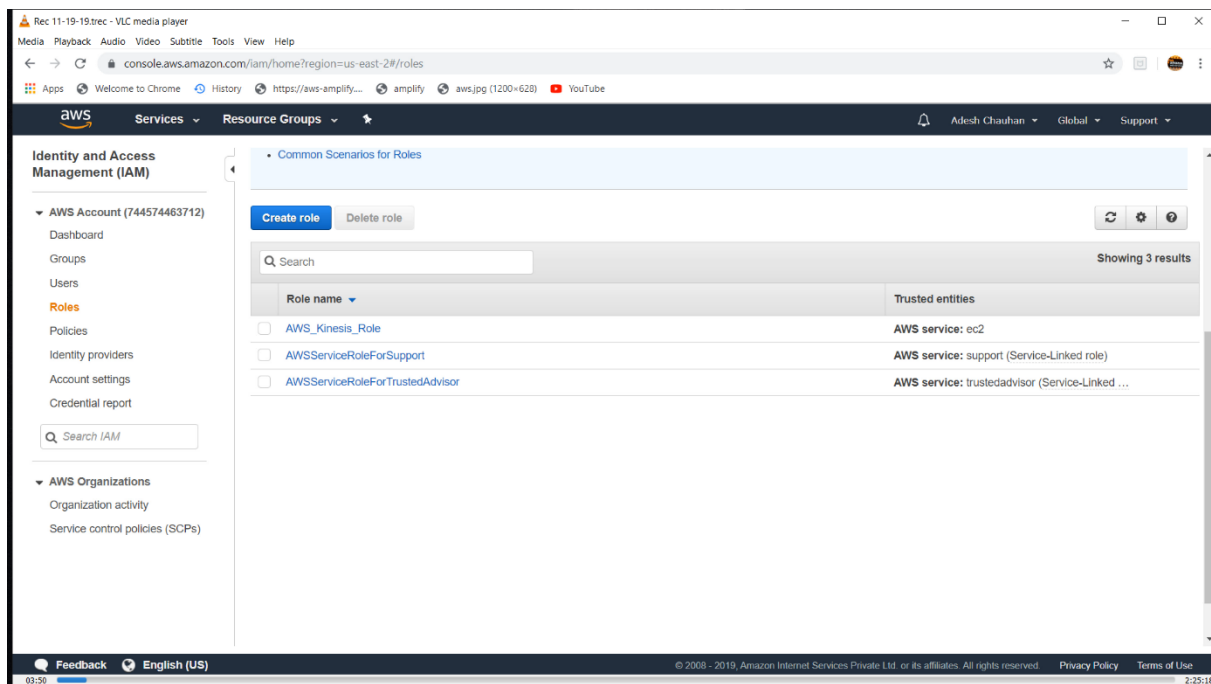
If you already have an AWS account, you can skip this prerequisite and use your existing account. To create AWS account if you do not already one:

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

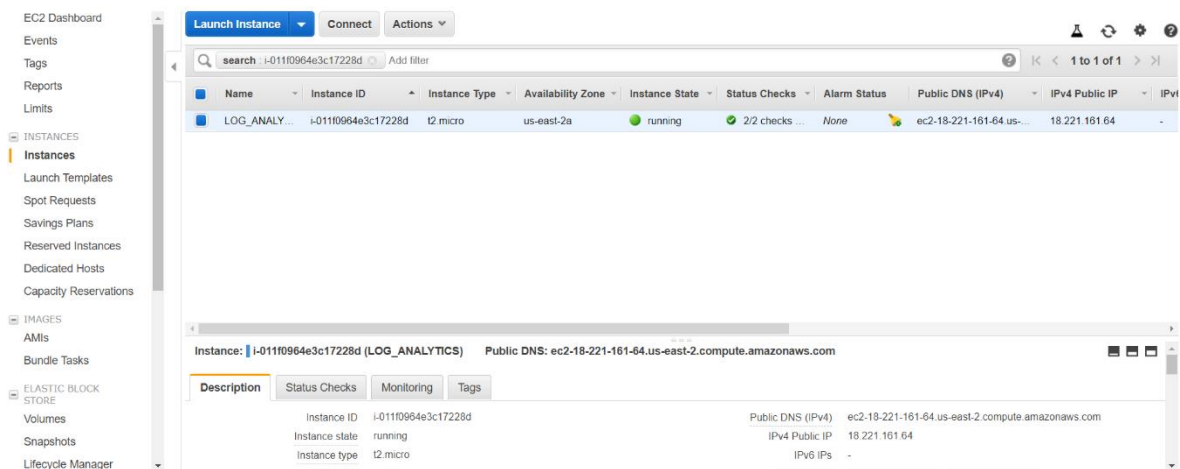
Part of the sign-up procedure involves receiving a phone call and entering a PIN using the phone keypad.

### Start an EC2 Instance

The steps outlined in this implementation assume that you are using an EC2 instance as the web server and log producer. You can use an existing EC2 instance launched from Amazon Linux AMI (version 2015.09 or later) or Red Hat Enterprise Linux (version 7 or later). Otherwise, you can follow the guide [Getting Started with Amazon EC2 Linux Instances](#) to start a new instance.<sup>7</sup> If you create a new instance, choose **Amazon Linux AMI** for your operating system. An instance type of **t2.micro** is sufficient for this tutorial.



You also want to ensure that your EC2 instance has an AWS Identity and Access Management (IAM) role configured with permission to write to Amazon Kinesis Firehose and Amazon CloudWatch. For more information, see [IAM Roles for Amazon EC2](#).<sup>8</sup>



Once you have launched the EC2 instance, you will need to connect to it via SSH. To connect to your instance, follow the guide [Connect to Your Linux Instance](#).<sup>9</sup>

### Prepare Your Log Files

Because Amazon Kinesis Analytics can analyse your streaming data in near Realtime, this tutorial is much more effective when a live stream of Apache access log data is used. If your EC2 instance is not serving HTTP traffic, you will need to generate continuous sample log files.

## Step 2: Create an Amazon Kinesis Firehose Delivery Stream

In Step 1, you created log files on your web server. Before they can be analysed with Amazon Kinesis Analytics (Step 6), the log data must first be loaded into AWS. Amazon Kinesis 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 Elasticsearch Service (Amazon ES).

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

To create the Amazon Kinesis Firehose delivery stream:

1. Open the Amazon Kinesis console at <https://console.aws.amazon.com/kinesis>.
2. Click **Go to Firehose**.
3. Click **Create Delivery Stream**.
4. On the **Destination** screen:
  - a. For Destination, choose Amazon S3.
  - b. For **Delivery stream name**, enter web-log-ingestion-stream.
  - c. For **S3 bucket**, choose **Create New S3 Bucket**.

You will be prompted to provide additional information for the new bucket:

For **Bucket name**, use a unique name. You will not need to use the name elsewhere in this tutorial. However, Amazon S3 bucket names are required to be globally unique.

For **Region**, choose **US Standard**. Choose **Create Bucket**.

- d. For **S3 prefix**, leave it blank (default).
  - e. Click **Next**.
5. On the **Configuration** screen (see below), 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 Amazon S3 bucket on your behalf.
  - a. For **IAM role**, choose **Create/Update Existing IAM Role**.
  - b. Click **Next**.

## Configuration ?

Configure buffer, compression, logging and IAM role options for your delivery stream.

### S3 Buffer

Firehose buffers incoming data before delivering to your S3 bucket. You can configure buffer size and buffer interval. The first satisfied condition will trigger the data delivery to your S3 bucket.

**Buffer size\***

Buffer size can range from 1MB to 128MB in 1MB increments.

**Buffer interval\***

Buffer interval can range from 60s to 900s in 1 second increments.

### S3 Compression and Encryption

Firehose can compress and encrypt the data before delivering to your S3 bucket.

**Data compression**



**Data encryption**



### Error Logging

Firehose can log data delivery errors to CloudWatch Logs. If enabled, a CloudWatch Log Group and corresponding Log Stream(s) are created on your behalf. [Learn more.](#)

☒ Enable ☐ Disable

### IAM Role

Firehose needs an IAM role to access your specified resources, such as the S3 bucket and KMS key. [Learn more.](#)

**IAM role\***



\*Required

[Cancel](#)

[Previous](#)

[Next](#)

1. A new screen will open (see below).
  - a. For **IAM Role**, choose **Create a new IAM Role**.
  - b. For **Role Name**, enter `firehose_delivery_role`.
  - c. Click **Next**.

**Amazon Kinesis Firehose is requesting permission to use resources in your account**

---

Click Allow to give Amazon Kinesis Firehose Read and Write access to resources in your account.

▼ Hide Details

---

**Role Summary** ?

---

**Role Description** Provides access to AWS Services and Resources

**IAM Role**

**Role Name**

▶ View Policy Document

7. Review the details of the Amazon Kinesis Firehose delivery stream and choose **Create Delivery Stream**.

### Step 3: Install and Configure the Amazon Kinesis Agent

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 stand-alone Java software application that offers an easy way to collect and send data to 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 pre-process 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 will parse log files in the Apache Common Log format and convert each line in the file to JSON format before sending to your Firehose delivery stream, which you created in Step 2.

1. To install the agent, see [Download and Install the Agent](#).<sup>11</sup>
2. For detailed instructions on how to configure the agent to process and send log data to your Amazon Kinesis Firehose delivery stream, see [Configure and Start the Agent](#).<sup>12</sup>

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

Replace *full-path-to-log-file* with a file pattern 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:

*"/var/log/httpd-access.log\*"*. The value will be different, depending on your use case. Replace *name-of-delivery-stream* with the name of the Firehose delivery stream you created in Step 2.

```
{
  "cloudwatch.endpoint": "monitoring.us-east-1.amazonaws.com",
  "cloudwatch.emitMetrics": true,
  "firehose.endpoint": "firehose.us-east-1.amazonaws.com",
  "flows": [
    {
      "filePattern": "full-path-to-log-file",
      "deliveryStream": "name-of-delivery-stream",
      "dataProcessingOptions": [
        {
          "initialPostion": "START_OF_FILE",
          "maxBufferAgeMillis": "2000",
          "optionName": "LOGTOJSON",
          "logFormat": "COMBINEDAPACHELOG"
        }
      ]
    }
  ]
}
```

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

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

Once started, the agent will look for files in the configured location and send the records to the Firehose delivery stream.

## Step 4: Create an Amazon Elasticsearch Service Domain

The data produced by this tutorial will be stored in Amazon ES for later visualization and analysis. To create the Amazon ES domain:

1. Open the Amazon ES console at <https://console.aws.amazon.com/es>.
2. If you have not previously created an Amazon ES Domain, choose **Get started**. Otherwise, choose **Create a new domain**.
3. On the **Define domain** screen:
  - a. For **Elasticsearch domain name**, enter `web-log-summary`.
  - b. For **Elasticsearch version**, leave it set to the default value.
  - c. Click **Next**.
4. On the **Configure cluster** screen (see below), leave all settings as their default values and click **Next**.

## Configure cluster

Configure the instance and storage settings for your cluster based on the traffic, data, and availability requirements of your application. A cluster is a collection of one or more data nodes, optional dedicated master nodes, and storage required to run Elasticsearch and operate your domain.

### Node configuration

Selecting the correct instance type and instance count depends on the compute, memory, and storage needs of your application. Take into account the size of the Elasticsearch indices, shards, and replicas you intend to create, the types of queries you will run, and the amount of storage you will need as you decide these settings. If you have a large volume of data to upload or anticipate a large amount of query traffic to your domain, you can preconfigure the cluster to handle this requirement.

**Instance count**   


**Instance type**  

☐ Enable dedicated master 

☐ Enable zone awareness 

### Storage configuration

Choose a storage type for your data nodes. If you choose the EBS storage type, you will need to specify the EBS volume type and EBS volume size for the cluster. The EBS volume size setting is configured per instance. Multiply the volume size by the number of data nodes in your cluster for the total storage size available in your cluster. Take into account size of indices, shards, and replicas you intend to create in your cluster when configuring storage settings. Storage settings do not apply to any dedicated master nodes in the cluster.

**Storage type**  

### Snapshot configuration

Once a day, Amazon ES takes an automated snapshot of your cluster. You can set the start hour for the snapshot. We recommend that you choose a time when traffic on your cluster is low.

**Automated snapshot start hour**  

### ► Advanced options

#### 5. On the **Set up access policy** screen (see below):

- a. For **Set the domain access policy to**, choose **Allow open access to the domain**.

**Note:** This is not a recommended setting for production Amazon ES domains. You should terminate this Amazon ES domain after completing the tutorial, or apply a more restrictive policy.

- b. Click **Next**.



## Set up access policy



To allow or block access to the domain, select a policy template from the template selector or add one or more Identity and Access Management (IAM) policy statements in the **Edit the access policy** box.

Set the domain access policy to

Allow open access to the domain ▾

### Add or edit the access policy

```

1 {
2   "Version": "2012-10-17",
3   "Statement": [
4     {
5       "Effect": "Allow",
6       "Principal": {
7         "AWS": [
8           "*"
9         ]
10      },
11      "Action": [
12        "es:*"
13      ],
14      "Resource": "arn:aws:es:us-east-1:012345678901:domain/web-log-summary/*"
15    }
16  ]
17 }
```

- Review the details for the Amazon ES domain and click **Confirm and create**. It will take approximately 10 minutes for the Amazon ES domain to be created. While the domain is being created, proceed with the remainder of this guide.

Dashboard
My domains
mynewkinesis
Reserved instances

You have successfully created an Elasticsearch domain.
You can update your domain configuration or access policy now. The domain endpoint will be available once the domain has reached an Active status.

mynewkinesis
Configure cluster
Modify access policy
Manage tags
Delete domain
Upgrade domain

Your domain is being initialized, which takes about 10 minutes. You cannot load data or run queries against your domain until the initialization is complete. The domain status will change to Active as soon as your domain is ready to use.

Overview
Cluster health
Instance health
Indices
Logs
Upgrade history

Domain status Loading
Elasticsearch version 7.1
Endpoint
Domain ARN arn:aws:es:us-east-2:744574463712:domain/mynewkinesis
Kibana
Availability zones 3
Instance type r5.large.elasticsearch
Number of instances 3
Master instance type r5.large.elasticsearch

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

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

1. Open the Amazon Kinesis console at <https://console.aws.amazon.com/kinesis>.
2. Click **Create Delivery Stream**.
3. On the **Create Delivery Stream** screen (see below), do the following:
  - a. For **Destination**, choose **Amazon Elasticsearch Service**.
  - b. For **Delivery stream name**, enter `web-log-aggregated-data`.
  - c. For **Elasticsearch domain**, choose the domain you created in Step 4.
  - d. For **Index**, enter `request_data`.
  - e. For **Index rotation**, leave it set to “NoRotation” (default).
  - f. For **Type**, enter `requests`.
  - g. For **Retry duration (sec)**, leave it set to “300” (default).
  - h. Under **Backup S3 bucket**, for **Backup mode**, select **Failed Documents Only**.
  - i. For **S3 bucket**, choose **New S3 bucket**.

You will be prompted to provide additional information for the new bucket:

For **Bucket name**, use a unique name. You will not need to use the name elsewhere in this tutorial. However, Amazon S3 bucket names are required to be globally unique.

For **Region**, choose **US Standard**. Choose **Create Bucket**.

- j. For **S3 prefix**, leave it blank (default).
- k. Click **Next**.

## Destination ?

Select the destination where your streaming data will be delivered.

**Destination\*** Amazon Elasticsearch Service

**Delivery stream name\*** web-log-aggregated-data

## Elasticsearch domain

**Elasticsearch domain\*** web-log-data

**Index\*** datetime

**Index rotation \*** NoRotation

**Type\*** requests

**Retry duration (sec)\*** 300

Retry duration can range from 0 seconds to 7200 seconds in 1 second increments.

## Backup S3 bucket

Firehose can back up data to your S3 bucket while delivering it to your Elasticsearch cluster.

**Backup mode\*** ☒ Failed Documents Only ☐ All Documents

**S3 bucket\*** Select S3 bucket

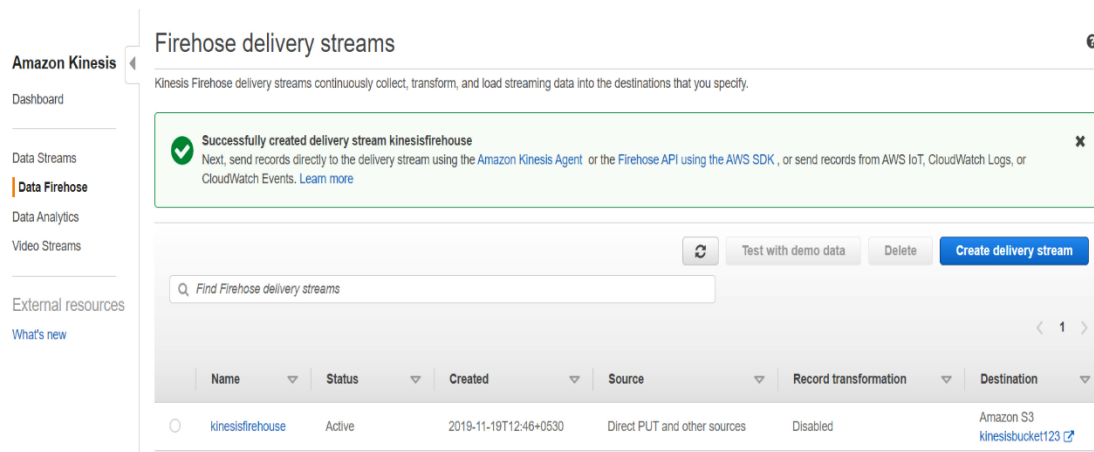
**S3 prefix** S3 prefix

\*Required

Cancel

Next

4. On the **Configuration** 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 Amazon ES domain on your behalf. For **IAM role**, choose **Create/Update Existing IAM Role**.
5. On the next screen, leave the value of **IAM Role** set to “firehose\_delivery\_role”, which you created in Step 2.
6. Set **Policy Name** to **Create a new Role Policy**.
7. Click **Allow**.
8. You will be returned to the Amazon Kinesis Firehose configuration screen. Click **Next**.



- Review the details for your Amazon Kinesis Firehose delivery stream and click **Create Delivery Stream**.

## Step 6: Create an Amazon Kinesis Analytics Application

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

- Open the Amazon Kinesis Analytics console at <https://console.aws.amazon.com/kinesisanalytics>.
- Click **Go to Analytics**.
- Click **Create new application**.
- For **Application name**, enter web-log-aggregation-tutorial.
- Click **Save and continue**.
- To configure the Source data for the Amazon Kinesis Analytics application, click **Connect to a source**.
- Under **Select a stream**, choose the Firehose delivery stream called “weblog-ingestion-stream” that you created in Step 2.
- Amazon Kinesis Analytics will analyze the source data in your Firehose delivery stream and create a formatted sample of the input data for your review:

Formatted stream sample

Raw stream sample

Filter by column name or column type

Edit schema

host VARCHAR(16)	datetime VARCHAR(32)	request VARCHAR(64)	response SMALLINT	bytes SMALLINT	referer VARCHAR(64)
153.233.179.68	29/Aug/2016:13:30:36 -0700	PUT /posts/posts/explore HTTP/1.0	200	5015	http://marquez.biz/main/
28.65.158.143	29/Aug/2016:13:30:36 -0700	GET /apps/cart.jsp?appID=1303 HTTP/1.0	404	5040	http://vazquez.com/wp-content/tags/tag/hom
79.217.236.155	29/Aug/2016:13:30:37 -0700	GET /explore HTTP/1.0	200	5037	http://www.johnson-bradford.com/
120.18.76.166	29/Aug/2016:13:30:37 -0700	POST /wp-content HTTP/1.0	200	4945	http://taylor-clayton.net/posts/index/
98.153.196.4	29/Aug/2016:13:30:37 -0700	GET /posts/posts/explore HTTP/1.0	200	5005	http://www.ward-jones.info/
192.182.180.218	29/Aug/2016:13:30:37 -0700	GET /list HTTP/1.0	200	5057	http://www.bradford.com/posts/wp-content/al
59.139.143.190	29/Aug/2016:13:30:37 -0700	GET /app/main/posts HTTP/1.0	200	5040	http://www.king-newman.com/category/
251.75.71.244	29/Aug/2016:13:30:38 -0700	GET /posts/posts/explore HTTP/1.0	200	4914	http://price.com/main.htm
188.143.103.141	29/Aug/2016:13:30:38 -0700	POST /list HTTP/1.0	200	5094	http://gutierrez-andrews.com/wp-content/exp
103.54.46.71	29/Aug/2016:13:30:38 -0700	GET /search/tag/list HTTP/1.0	200	5127	http://www.walters-espinoza.org/

1. Leave all values set to their defaults, and click **Save and continue**.
2. You will be taken back to the hub screen for your Amazon Kinesis Analytics application. To create the SQL that will analyze the streaming data, click **Go to SQL editor**.
3. When prompted, choose **Yes, start application**.
4. After approximately 60 to 90 seconds, the **Source data** section will present you with a sample of source data that is flowing into your source delivery stream.
5. In the SQL editor, enter the following SQL code:

```
CREATE OR REPLACE STREAM
"DESTINATION_SQL_STREAM" (    datetime
VARCHAR(30),    status INTEGER,    statusCount
INTEGER);
```

```
CREATE OR REPLACE PUMP "STREAM_PUMP" AS
INSERT INTO "DESTINATION_SQL_STREAM"
SELECT
    STREAM                                TIMESTAMP_TO_CHAR('yyyy-MM-
dd"T"HH:mm:ss.SSS',
LOCALTIMESTAMP) 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);
```

14. Click **Save and run SQL**. After about 1 minute, Amazon Kinesis Analytics will display the output of the query:

Add SQL from templates
Export SQL

```

1 CREATE OR REPLACE STREAM "DESTINATION_SQL_STREAM" (datetime TIMESTAMP, status INTEGER, statusCount INTEGER);
2
3 CREATE OR REPLACE PUMP "STREAM_PUMP" AS INSERT INTO "DESTINATION_SQL_STREAM"
4 SELECT STREAM ROWTIME as datetime, "response" as status, COUNT(*) AS statusCount
5 FROM "SOURCE_SQL_STREAM_001"
6 GROUP BY "response", FLOOR(("SOURCE_SQL_STREAM_001".ROWTIME - TIMESTAMP '1970-01-01 00:00:00') minute / 1 TO MINUTE);
7

```

Exit (done editing)
Save and run SQL

Source data
Real-time analytics
Destination

Application status: RUNNING

In-application streams:

DESTINATION\_SQL\_STREAM

error\_stream

Pause results
↻ New results will be added every 2-10 seconds

☐
Scroll to bottom when new results arrive.

ROWTIME	DATETIME	STATUS	STATUSCOUNT
2016-08-29 13:49:00.0	2016-08-29 13:49:00.0	200	930
2016-08-29 13:49:00.0	2016-08-29 13:49:00.0	301	35
2016-08-29 13:49:00.0	2016-08-29 13:49:00.0	404	40
2016-08-29 13:49:00.0	2016-08-29 13:49:00.0	500	14
2016-08-29 13:50:00.0	2016-08-29 13:50:00.0	200	228
2016-08-29 13:50:00.0	2016-08-29 13:50:00.0	301	9
2016-08-29 13:50:00.0	2016-08-29 13:50:00.0	404	11
2016-08-29 13:50:00.0	2016-08-29 13:50:00.0	500	9
2016-08-29 13:51:00.0	2016-08-29 13:51:00.0	200	219
2016-08-29 13:51:00.0	2016-08-29 13:51:00.0	404	8
2016-08-29 13:51:00.0	2016-08-29 13:51:00.0	500	2
2016-08-29 13:51:00.0	2016-08-29 13:51:00.0	301	10

14. To save the running output of the query, choose the **Destination** tab.
15. Under **Select a stream**, choose the Firehose delivery stream called “weblog-aggregated-data” that you created in [Step 5](#).
16. Leave all other options set to their default values and click **Save and continue**.

## Step 7: View the Aggregated Streaming Data

After approximately 5 minutes, the output of the SQL statement in your Amazon Kinesis Analytics application will be written to your Amazon ES domain. Amazon ES has built-in support for Kibana, a tool that allows users to explore and visualize the data stored in an Elasticsearch cluster.<sup>13</sup> To view the output of your Amazon Kinesis Analytics application in Kibana:

1. Open the Amazon ES console at <https://console.aws.amazon.com/es>.
2. In the **Domain** column, choose the Amazon ES domain called “web-logsummary” that you created in Step 4.
3. The details for the Amazon ES domain will be presented with a link to Kibana. Select the link next to the Kibana item.
4. Because this is the first time you are opening the Kibana application in your Amazon ES domain, you will need to configure it to use the Elasticsearch index name that you created in Step 5. In the **Index name or pattern** field, enter `request_data`.
5. Kibana will automatically identify the DATETIME field in your input data, which contains time data. Choose **Create**.
6. To visualize the data in our Elasticsearch 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. Click on **Visualize** in the navigation bar, and choose **Line chart**.
- b. Choose **From a new search**.

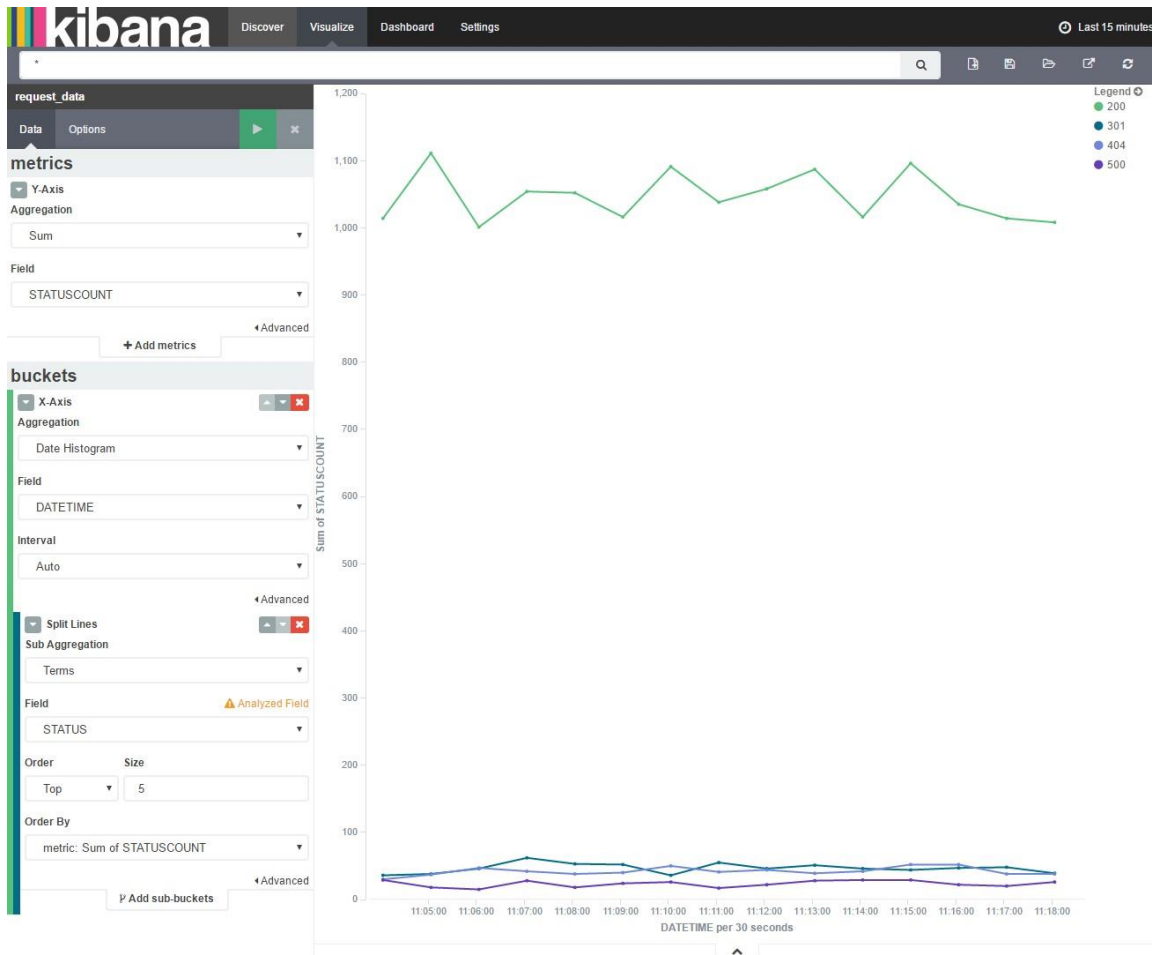
To configure your chart, you first need to tell Kibana what data to use for the y-axis:

- c. In the **metrics** section, click the arrow next to **Y-Axis** to configure this.
- d. Under **Aggregation**, choose **Sum**.
- e. Under **Field**, choose **STATUSCOUNT**.

Now you need to configure the x-axis:

- f. In the **buckets** section, select **X-Axis** under **Select buckets type**.
  - g. Under **Aggregation**, choose **Date Histogram**.
7. In the **buckets** section, choose **Add sub-buckets**. From **Select buckets type**, choose **Split Lines**.

8. Under **Sub Aggregation**, choose **Terms**.
9. Under **Field**, choose **STATUS**.
10. To run the query and view the line chart, click on the green and white “play” button.





## 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.

To terminate the instance, navigate to the EC2 console at <https://console.aws.amazon.com/ec2> and follow these steps:

1. Select the **Running Instances** link, and find the instance you used to generate your Apache access logs.
2. From the **Actions** menu, choose the instance, and then choose **Instance State** ☐ **Terminate**.
3. 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 Firehose delivery stream. Stop the agent with the following command:

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

### Delete the Amazon Elasticsearch Service Domain

To delete the Amazon ES domain, navigate to the Amazon ES console at <https://console.aws.amazon.com/es> and follow these steps:

1. Locate and select the domain “web-log-summary” that you created in Step 4.
2. Scroll down and expand the section **Delete Elasticsearch domain**.
3. Choose **Delete domain**.
4. On the Delete domain confirmation, select the checkbox and choose **Delete**.

### Delete the Amazon S3 Bucket

To delete the Amazon S3 bucket and objects within that bucket, navigate to the Amazon S3 console at <https://console.aws.amazon.com/s3> and follow these steps:

1. Locate the S3 bucket that you created in Step 2.
2. Right-click on the bucket name and choose **Delete Bucket**.

3. To confirm the deletion, type the bucket name and choose **Delete**.

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 Firehose and Amazon Kinesis 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 steps below.

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

To delete the delivery streams and the Amazon Kinesis Analytics application, navigate to the Amazon Kinesis console at <https://console.aws.amazon.com/kinesis> and follow these steps:

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

## Bibliography

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.<sup>14</sup>
- 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.<sup>15</sup>
- For additional examples, see Example Amazon Kinesis Analytics Applications.<sup>16</sup>

## Notes

1. <https://aws.amazon.com/ec2/pricing/>
2. <https://aws.amazon.com/kinesis/firehose/pricing/>
3. <https://aws.amazon.com/s3/pricing/>
4. <https://aws.amazon.com/free/>
5. <https://aws.amazon.com/kinesis/analytics/pricing/>
6. <https://aws.amazon.com/elasticsearch-service/pricing/>
7. [http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/EC2\\_GetStarted.html](http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/EC2_GetStarted.html)
8. <http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/iam-roles-foramazon-ec2.html>
9. <http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/AccessingInstances.html>
10. <https://github.com/kiritbasu/Fake-Apache-Log-Generator>
11. <http://docs.aws.amazon.com/firehose/latest/dev/writing-withagents.html#download-install>
12. <http://docs.aws.amazon.com/firehose/latest/dev/writing-withagents.html#config-start>
13. <https://www.elastic.co/products/kibana>
14. <http://docs.aws.amazon.com/kinesisanalytics/latest/dev/how-it-works.html>
15. <http://docs.aws.amazon.com/kinesisanalytics/latest/dev/streaming-sqlconcepts.html>
16. <http://docs.aws.amazon.com/kinesisanalytics/latest/dev/example-apps.html>