

- Overview

What you'll accomplish:

Set up a Kinesis Agent on data sources to collect data and send it continuously to Amazon Kinesis Firehose.

Create an end-to-end data delivery stream using Kinesis Firehose. The delivery stream will transmit your data from the agent to destinations including Amazon Kinesis Analytics, Amazon Redshift, Amazon Elasticsearch Service, and Amazon S3.

Process incoming log data using SQL queries in Amazon Kinesis Analytics.

Load processed data from Kinesis Analytics to Amazon Elasticsearch Service to index the data.

Analyze and visualize the processed data using Kibana.

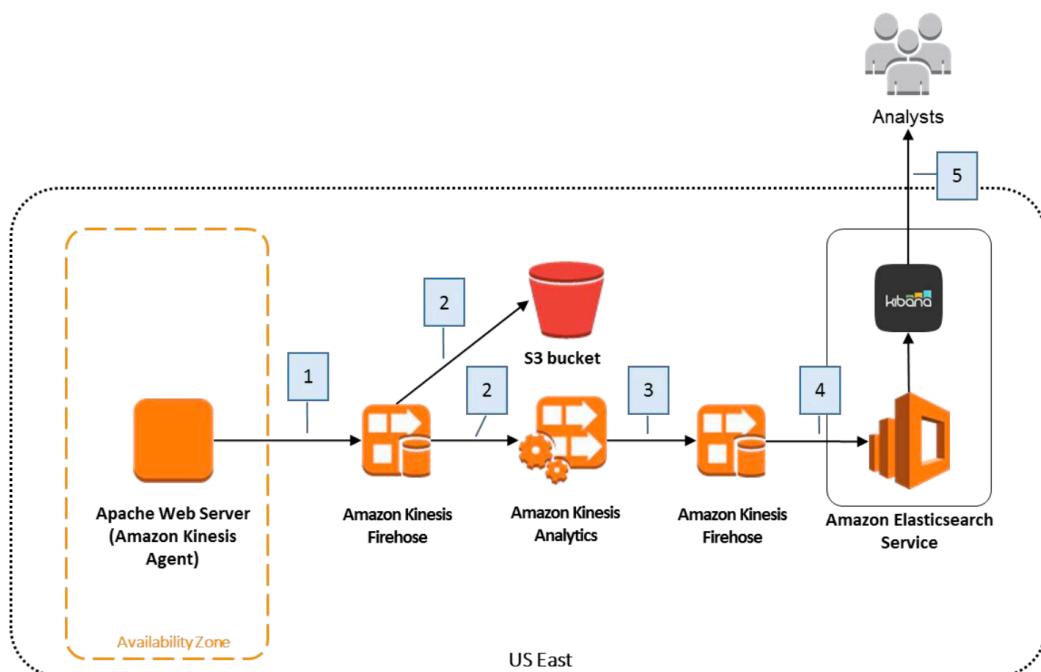
What you'll need before starting:

An AWS Account: You will need an AWS account to begin provisioning resources to host your website. [Sign up for AWS](#).

IT Experience: You will need a basic understanding of web technologies and familiarity with SQL to successfully complete this project.

AWS Experience: No prior experience with AWS is required to successfully complete this project.

An existing server to generate application logs.



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).

• 서비스별 Description

Amazon EC2

Description: 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

Description: 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



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.

Amazon Kinesis Analytics

Description: Amazon Kinesis 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 Streams and Amazon Kinesis Firehose, and build stream processing queries that filter, transform, and aggregate the data as it arrives. Amazon Kinesis 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 Analytics runs your queries continuously, and writes the processed results to output destinations such as Amazon Kinesis Streams and Amazon Kinesis Firehose, which can deliver the data to Amazon S3, Amazon Redshift, and Amazon ES. Amazon Kinesis Analytics automatically provisions, deploys, and scales the resources required to run your queries.

Amazon Elasticsearch Service

Description: 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.

- Realtime Log 생성을 위해
Fake Apache Log Generator
를 설치

<https://github.com/kiritbasu/Fake-Apache-Log-Generator.git>

python 설치 필요

```
[ec2-user@ip-10-0-1-160 ~]$ sudo yum install -y python-pip
Loaded plugins: priorities, update-motd, upgrade-helper
Resolving Dependencies
--> Running transaction check
--> Package python26-pip.noarch 0:9.0.3-1.27.amzn1 will be installed
--> Processing Dependency: python(abi) = 2.6 for package: python26-pip-9.0.3-1.27.amzn1.noarch
--> Processing Dependency: python26-setuptools for package: python26-pip-9.0.3-1.27.amzn1.noarch
```

sudo pip install --upgrade pip
root 권한으로 아래 4개 module install
/usr/local/bin/pip install pytz
/usr/local/bin/pip install numpy
/usr/local/bin/pip install faker
/usr/local/bin/pip install tzlocal

```
[ec2-user@ip-10-0-1-160 Fake-Apache-Log-Generator]$ python apache-fake-log-gen.py --output LOG
[ec2-user@ip-10-0-1-160 Fake-Apache-Log-Generator]$ ls -lrt
total 32
-rw-rw-r-- 1 ec2-user ec2-user    74 Feb 18 00:52 requirements.txt
-rw-rw-r-- 1 ec2-user ec2-user  1859 Feb 18 00:52 README.md
-rw-rw-r-- 1 ec2-user ec2-user 11358 Feb 18 00:52 LICENSE
-rw-rw-r-- 1 ec2-user ec2-user  3614 Feb 18 00:52 apache-fake-log-gen.py
-rw-rw-r-- 1 ec2-user ec2-user   277 Feb 18 02:00 access_log_20200218-020032.log
-rw-rw-r-- 1 ec2-user ec2-user   190 Feb 18 02:01 access_log_20200218-020121.log
```

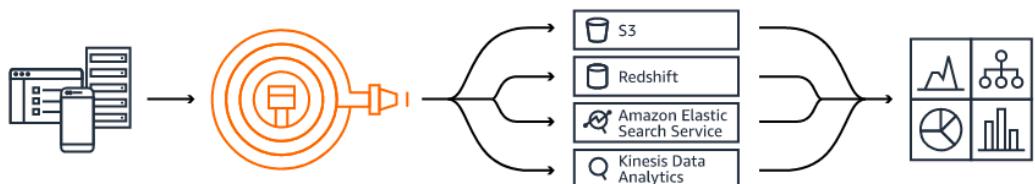
• Amazon Kinesis Firehose

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)

• Kinesis Firehose 생성

Deliver streaming data with Kinesis Firehose delivery streams

Continuously collect, transform, and load streaming data into destinations such as Amazon S3 and Amazon Redshift.



Create delivery stream

Source

To learn about enabling server-side encryption (SSE), see [Data Protection in Amazon Kinesis Data Firehose](#).

Direct PUT or other sources

Choose this option to send records directly to the delivery stream, or to send records from AWS IoT, CloudWatch Logs, or CloudWatch Events.

Kinesis Data Stream

S3 bucket

kh-kinesis **New Bucket Created** ▼



Create new

[View kh-kinesis in S3 console](#)

Permissions

IAM role **New Role Created**
[firehose_delivery_role](#)

Create new or choose

- Amazon Kinesis Agent 설치

<https://docs.aws.amazon.com/firehose/latest/dev/writing-with-agents.html#download-install>
you can configure the EC2 instance to send the data using the Amazon Kinesis Agent software

To set up the agent using the Amazon Linux AMI

Use the following command to download and install the agent:

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

1. Open and edit the configuration file (as superuser if using default file access permissions):

```
/etc/aws-kinesis/agent.json
```

In this configuration file, specify the files ("filePattern") from which the agent collects data, and the name of the delivery stream ("deliveryStream") to which the agent sends data. The file name is a pattern, and the agent recognizes file rotations. You can rotate files or create new files no more than once per second. The agent uses the file creation time stamp to determine which files to track and tail into your delivery stream. Creating new files or rotating files more frequently than once per second does not allow the agent to differentiate properly between them.

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

The default AWS Region is us-east-1. If you are using a different Region, add the `firehose.endpoint` setting to the configuration file, specifying the endpoint for your Region. For more information, see [Agent Configuration Settings](#).

2. Start the agent manually:

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

3. (Optional) Configure the agent to start on system startup:

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

The agent is now running as a system service in the background. It continuously monitors the specified files and sends data to the specified delivery stream. Agent activity is logged in `/var/log/aws-kinesis-agent/aws-kinesis-agent.log`.

- [로그 생성](#)

```
[root@ip-10-0-1-160 Fake-Apache-Log-Generator]#  
python apache-fake-log-gen.py -o LOG -n 0 -p /var/  
log/aws-kinesis-agent/
```

-o LOG : access log를 파일형식으로 저장
-p /var/log/aws-kinesis-agent : 파일 저장 위치

- 수행로그

/var/log/aws-kinesis-agent/aws-kinesis-agent.log

```
2020-02-18 04:55:41.986+0000 ip-10-0-1-160 (Agent.MetricsEmitter RUNNING) com.amazon.kinesis.streaming.agent.Agent [INFO] Agent: Progress: 44729 records parsed (10134908 bytes), and 44729 records sent successfully to destinations. Uptime: 270052ms
2020-02-18 04:56:11.979+0000 ip-10-0-1-160 (FileTailer RUNNING) com.amazon.kinesis.streaming.agent.tailing.FileTailer [INFO] FileTailer[fb:web-log-ingestion-stream:/var/log/aws-kinesis-agent/*access_log*]: Tailer Progress: Tailer has parsed 44729 records (10134908 bytes), transformed 0 records, skipped 0 records, and has successfully sent 44729 records to destination.
2020-02-18 04:56:11.986+0000 ip-10-0-1-160 (Agent.MetricsEmitter RUNNING) com.amazon.kinesis.streaming.agent.Agent [INFO] Agent: Progress: 44729 records parsed (10134908 bytes), and 44729 records sent successfully to destinations. Uptime: 300052ms
```

- 설정파일

/etc/aws-kinesis/agent.json

```
[root@ip-10-0-1-160 Fake-Apache-Log-Generator]# cat /etc/aws-kinesis/agent.json
{
  "cloudwatch.endpoint": "monitoring.ap-northeast-2.amazonaws.com",
  "cloudwatch.emitMetrics": true,
  "firehose.endpoint": "firehose.ap-northeast-2.amazonaws.com",
  "flows": [
    {
      "firehose.endpoint": "firehose.ap-northeast-2.amazonaws.com",
      "filePattern": "/var/log/aws-kinesis-agent/*access_log*",
      "deliveryStream": "web-log-ingestion-stream"
    }
  ]
}
```

firehose.endpoint 지정 필요

filePattern : fake generator 파일 명명 규칙 적용

- aws-kinesis-agent 재 기동

```
[ec2-user@ip-10-0-1-160 Fake-Apache-Log-Generator]$ sudo service aws-kinesis-agent restart
aws-kinesis-agent shutdown                                     [  OK  ]
aws-kinesis-agent startup                                     [  OK  ]
```

- Resources Not Found 에러 발생 시

- /etc/aws-kinesis/agent.json 파일 수정 필요
 "firehose.endpoint": "firehose.ap-northeast-2.amazonaws.com",
 추가

```
2020-02-18 04:14:19.950+0000 ip-10-0-1-160 (FileTailer[kinesis:web-log-ingestion-stream:/var/log/fake-log/_access_log*]) com.amazon.kinesis.streaming.agent.tailing.KinesisParser [INFO] KinesisParser[kinesis:web-log-ingestion-stream:/var/log/fake-log/_access_log*]: Continuing to parse /var/log/fake-log/_access_log_20200218-041419.log.  

2020-02-18 04:14:21.497+0000 ip-10-0-1-160 (sender-162) com.amazon.kinesis.streaming.agent.tailing.AsyncPublisher [ERROR] AsyncPublisher[kinesis:web-log-ingestion-stream:/var/log/fake-log/_access_log*]:RecordBuffer(id=2,records=1,kinesis endpoint 설정 추가 필요 amazonaws.services.kinesis.model.ResourceNotFoundException: Stream web-log-ingestion-stream under account 936777008077 not found. (Service: AmazonKinesis; Status Code: 400; Error Code: ResourceNotFoundException) Request ID: cc530c77-bfe8-1d6c-9a96-6aaacd8e0136). Will retry.  

2020-02-18 04:14:29.019+0000 ip-10-0-1-160 (sender-163) com.amazon.kinesis.streaming.agent.tailing.AsyncPublisher [ERROR] AsyncPublisher[kinesis:web-log-ingestion-stream:/var/log/fake-log/_access_log*]:RecordBuffer(id=2,records=12,bytes=2841) Retriable send error (com.amazonaws.services.kinesis.model.ResourceNotFoundException: Stream web-log-ingestion-stream under account 936777008077 not found. (Service: AmazonKinesis; Status Code: 400; Error Code: ResourceNotFoundException; Request ID: c0505871-58a6-6615-9695-3e95edabf24b)). Will retry.
```

- aws-kinesis-agent 수행로그

```
[ec2-user@ip-10-0-1-160 ~]$ tail -f /var/log/aws-kinesis-agent/aws-kinesis-agent.log  

2020-02-18 05:04:11.986+0000 ip-10-0-1-160 (Agent.MetricsEmitter RUNNING) com.amazon.kinesis.streaming.agent.Agent [INFO] Agent: Progress: 44729 records parsed (10134908 bytes), and 44729 records sent successfully to destinations. Uptime: 3570052ms  

2020-02-18 05:11:11.979+0000 ip-10-0-1-160 (FileTailer[fh:web-log-ingestion-stream:/var/log/aws-kinesis-agent/*access_log*].MetricsEmitter RUNNING) com.amazon.kinesis.streaming.agent.tailing.FileTailer [INFO] FileTailer[fh:web-log-ingestion-stream:/var/log/aws-kinesis-agent/*access_log*]: Tailer Progress: Tailer has parsed 44729 records (10134908 bytes), transformed 0 records, skipped 0 records, and has successfully sent 44729 records to destination.  

2020-02-18 05:11:11.986+0000 ip-10-0-1-160 (Agent.MetricsEmitter RUNNING) com.amazon.kinesis.streaming.agent.Agent [INFO] Agent: Progress: 44729 records parsed (10134908 bytes), and 44729 records sent successfully to destinations. Uptime: 3600052ms  

2020-02-18 05:12:00.372+0000 ip-10-0-1-160 (FileTailer[fh:web-log-ingestion-stream:/var/log/aws-kinesis-agent/*access_log*].MetricsEmitter RUNNING) com.amazon.kinesis.streaming.agent.tailing.FirehoseParser [INFO] FirehoseParser[fh:web-log-ingestion-stream:/var/log/aws-kinesis-agent/*access_log*]: Continuing to parse /var/log/aws-kinesis-agent/_access_log_20200218-055120.log.  

2020-02-18 05:12:00.372+0000 ip-10-0-1-160 (FileTailer[fh:web-log-ingestion-stream:/var/log/aws-kinesis-agent/*access_log*].MetricsEmitter RUNNING) com.amazon.kinesis.streaming.agent.tailing.FileTailer [INFO] FileTailer[fh:web-log-ingestion-stream:/var/log/aws-kinesis-agent/*access_log*]: Tailer Progress: Tailer has parsed 53949 records (12223194 bytes), transformed 0 records, skipped 0 records, and has successfully sent 53729 records to destination.  

2020-02-18 05:12:00.372+0000 ip-10-0-1-160 (FileTailer[fh:web-log-ingestion-stream:/var/log/aws-kinesis-agent/*access_log*].MetricsEmitter RUNNING) com.amazon.kinesis.streaming.agent.tailing.FileTailer [INFO] FileTailer[fh:web-log-ingestion-stream:/var/log/aws-kinesis-agent/*access_log*]: Tailer Progress: Tailer has parsed 66806 records (15134812 bytes), transformed 0 records, skipped 0 records, and has successfully sent 66729 records to destination.  

2020-02-18 05:12:00.372+0000 ip-10-0-1-160 (FileTailer[fh:web-log-ingestion-stream:/var/log/aws-kinesis-agent/*access_log*].MetricsEmitter RUNNING) com.amazon.kinesis.streaming.agent.tailing.FileTailer [INFO] FileTailer[fh:web-log-ingestion-stream:/var/log/aws-kinesis-agent/*access_log*]: Tailer Progress: Tailer has parsed 79509 records (18016449 bytes), transformed 0 records, skipped 0 records, and has successfully sent 79229 records to destination.  

2020-02-18 05:12:00.372+0000 ip-10-0-1-160 (FileTailer[fh:web-log-ingestion-stream:/var/log/aws-kinesis-agent/*access_log*].MetricsEmitter RUNNING) com.amazon.kinesis.streaming.agent.tailing.FileTailer [INFO] FileTailer[fh:web-log-ingestion-stream:/var/log/aws-kinesis-agent/*access_log*]: Tailer Progress: Tailer has parsed 92401 records (20934385 bytes), transformed 0 records, skipped 0 records, and has successfully sent 92229 records to destination.  

2020-02-18 05:12:00.372+0000 ip-10-0-1-160 (FileTailer[fh:web-log-ingestion-stream:/var/log/aws-kinesis-agent/*access_log*].MetricsEmitter RUNNING) com.amazon.kinesis.streaming.agent.tailing.FileTailer [INFO] FileTailer[fh:web-log-ingestion-stream:/var/log/aws-kinesis-agent/*access_log*]: Tailer Progress: Tailer has parsed 92401 records (20934385 bytes), and 92229 records sent successfully to destinations. Uptime: 3720052ms
```

- kinesis analytics 세팅

connect to streaming data

Kinesis Analytics - Create application

Kinesis Analytics applications continuously read and analyze data from a connected streaming source in real-time. To enable interactivity with your data during configuration you will be prompted to run your application. Kinesis Analytics resources are not covered under the [AWS Free Tier](#), and [usage-based charges apply](#). For more information, see [Kinesis Analytics pricing](#).

Application name

Acceptable characters are uppercase and lowercase letters, numbers, underscores, hyphens, and periods.

Description - optional

fake apache log generator ---> aws-kinesis-agent로 accesslog를 firehose로 전송
이 데이터를 kinesis analytics에서 SQL을 이용하여 가공 후 다른 firehose로 전송

Runtime

SQL

Process data in real-time using SQL, which provides an easy way to quickly query large volumes of streaming data without learning new frameworks or languages. [Learn more](#)

Apache Flink

Apache Flink is an open-source framework and distributed processing engine for stateful computations over unbounded and bounded data streams. [Learn more](#)

 After you create the application, you can't change the type or version of the runtime environment.

[Cancel](#)

[Create application](#)

Connect streaming data source

Choose from your Kinesis data streams and Firehose delivery streams, or quickly configure a demo Kinesis stream that can be used to explore Kinesis Analytics.

Choose source

Configure a new stream

Source

Kinesis data stream

Kinesis data stream is an ordered sequence of data records used for rapid and continuous data intake and aggregation.

Kinesis Firehose delivery stream

Kinesis Firehose delivery streams send source records to the destinations that you specify, automatically and continuously.

Kinesis Firehose delivery stream



[Create new](#)

[View web-log-ingestion-stream in Kinesis Firehose](#)

In-application stream name

In your SQL queries, refer to this source as: **SOURCE_SQL_STREAM_001**

Connect streaming data source

Choose from your Kinesis data streams and Firehose delivery streams, or quickly configure a demo Kinesis stream that can be used to explore Kinesis Analytics.

Choose source

Configure a new stream

Source

Kinesis data stream

Kinesis data stream is an ordered sequence of data records used for rapid and continuous data intake and aggregation.

Kinesis Firehose delivery stream

Kinesis Firehose delivery streams send source records to the destinations that you specify, automatically and continuously.

Kinesis Firehose delivery stream

Input firehose 선택

web-log-ingestion-stream



Create new

[View web-log-ingestion-stream in Kinesis Firehose](#)

In-application stream name

In your SQL queries, refer to this source as: **SOURCE_SQL_STREAM_001**

Record pre-processing with AWS Lambda

Kinesis Analytics can invoke your Lambda function to pre-process records before they are used in this application. To pre-process records, your Lambda function must be compliant with the required record transformation output model. [Learn more](#)

Record pre-processing

Disabled

Enabled

Access permissions

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

Access permissions

Create / update IAM role **kinesis-analytics-web-log-aggregation-tutorial-ap-northeast-2**

Choose from IAM roles that Kinesis Analytics can assume

정상적인 Schema 분석을 위해서 fake apache generator에서 최대 sleep time을 10ms 이하로 설정하여 로그를 생성해야함.
[root@ip-10-0-1-160 fake-apache]# python apache-fake-log-gen.py -o LOG -n 0 -p /var/log/aws-kinesis-agent/ --sleep 0.01

Schema

Schema discovery can generate a schema using recent records from the source. Schema column names are the same as in the source, unless they contain special characters, repeated column names, or reserved keywords. [Learn more](#)

[Discover schema](#)

- not enough data in stream web-log-ingestion-stream

Schema

Schema discovery can generate a schema using recent records from the source. Schema column names are the same as in the source, unless they contain special characters, repeated column names, or reserved keywords. [Learn more](#)

 Schema discovery has failed because there is not enough data in stream web-log-ingestion-stream
Increase the throughput into the source and then choose **Retry schema discovery**.

Troubleshooting

- To see when data was last written to the stream, go to [CloudWatch metrics](#)

Other ways to populate the stream

- Generate and send sample data with the [Amazon Kinesis Data Generator](#)
- Use the [AWS SDKs / agents](#)

[Retry schema discovery](#)

==> python apache-fake-log-gen.py -o LOG -n 0 -p /var/log/aws-kinesis-agent/ --sleep 0.01

==> 아래와 같이 scheme 분석이 완료되었음

Schema

Schema discovery can generate a schema using recent records from the source. Schema column names are the same as in the source, unless they contain special characters, repeated column names, or reserved keywords. [Learn more](#)

 Schema discovery successful

Detected CSV format and applied schema

- To define a custom schema, choose "Edit schema" in the stream sample below.
- To capture a new stream sample from the selected source for discovery, choose **Retry schema discovery** below.

[Edit schema](#)

[Retry schema discovery](#)

가독성이 높은 SQL 작성을 위해서는

COL0, COL1, COL2 컬럼명을 수정해야함

input stream data를 SQL로 가공 후 다른 firehose로 전송

ex) COL0 -> IP , COL3-> TIME, COL5 -> URI

Raw

Lambda output

Formatted

Filter by column name								
COL0 VARCHAR(16)	COL1 VARCHAR(4)	COL2 VARCHAR(4)	COL3 VARCHAR(32)	COL4 VARCHAR(8)	COL5 VARCHAR(64)	COL6 VARCHAR(4)	COL7 VARCHAR	
77.187.15.133	-	-	[19/Feb/2020:01:45:06	+0000]	GET /app/main/posts HTTP/1.0			
111.30.95.48	-	-	[19/Feb/2020:01:45:09	+0000]	GET /explore HTTP/1.0			
71.57.134.41	-	-	[19/Feb/2020:01:45:08	+0000]	GET /explore HTTP/1.0			
149.46.48.171	-	-	[19/Feb/2020:01:45:05	+0000]	GET /explore HTTP/1.0			
89.6.124.205	-	-	[19/Feb/2020:01:45:07	+0000]	GET /explore HTTP/1.0			
74.186.91.24	-	-	[19/Feb/2020:01:45:09	+0000]	GET /wp-content HTTP/1.0			
177.30.25.11	-	-	[19/Feb/2020:01:45:09	+0000]	GET /app/main/posts HTTP/1.0			
201.47.243.161	-	-	[19/Feb/2020:01:45:06	+0000]	GET /wp-admin HTTP/1.0			
192.247.171.15	-	-	[19/Feb/2020:01:45:06	+0000]	GET /wp-admin HTTP/1.0			
101.141.209.20	-	-	[19/Feb/2020:01:45:07	+0000]	GET /explore HTTP/1.0			

Filter by column name

Column order	Column name	Column type
+ Add column		
1	IP	VARCHAR Length: 16
2	COL1	VARCHAR Length: 4
3	COL2	VARCHAR Length: 4
4	DATETIME	VARCHAR Length: 32
5	COL4	VARCHAR Length: 8
6	URI	VARCHAR Length: 64
7	COL6	VARCHAR Length: 4
8	COL7	VARCHAR Length: 4

Cancel **Save schema and update stream samples**

Application status: READY

위 컬럼 이름을 아래의 데이터를 보고 적당한 이름으로 변경 (가능하면 대문자로 변경)

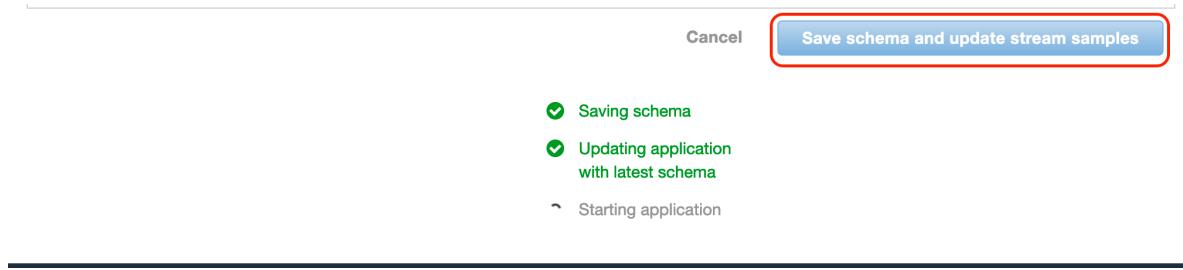
Raw | Lambda output | **Formatted** | Error stream

Application status: READY

Filter by column name

COL0 VARCHAR(16)	COL1 VARCHAR(4)	COL2 VARCHAR(4)	COL3 VARCHAR(32)	COL4 VARCHAR(8)	COL5 VARCHAR(64)	COL6 VARCHAR(4)	COL7 VARCHAR
77.187.15.133	-	-	[19/Feb/2020:01:45:06	+0000]	GET /app/main/posts HTTP/1.0		
111.30.95.48	-	-	[19/Feb/2020:01:45:09	+0000]	GET /explore HTTP/1.0		
71.57.134.41	-	-	[19/Feb/2020:01:45:08	+0000]	GET /explore HTTP/1.0		
149.46.48.171	-	-	[19/Feb/2020:01:45:05	+0000]	GET /explore HTTP/1.0		
89.6.124.205	-	-	[19/Feb/2020:01:45:07	+0000]	GET /explore HTTP/1.0		
74.186.91.24	-	-	[19/Feb/2020:01:45:09	+0000]	GET /wp-content HTTP/1.0		
177.30.25.11	-	-	[19/Feb/2020:01:45:09	+0000]	GET /app/main/posts HTTP/1.0		
201.47.243.161	-	-	[19/Feb/2020:01:45:06	+0000]	GET /wp-admin HTTP/1.0		
192.247.171.15	-	-	[19/Feb/2020:01:45:06	+0000]	GET /wp-admin HTTP/1.0		
101.141.209.20	-	-	[19/Feb/2020:01:45:07	+0000]	GET /explore HTTP/1.0		

Save Schema and update stream sample 클릭



Application status가 RUNNING로 변경

A screenshot of the AWS Lambda console showing the application status as "RUNNING". Above the main content area, there is a blue bar with the text "Updating stream sample" and a "Exit (done)" button. Below this, there are tabs for "Raw", "Lambda output", "Formatted" (which is selected), and "Error stream". A message at the top says "Retrieving rows from stream "SOURCE_SQL_STREAM_001"".

Below the tabs, there is a table titled "위에서 수정한 컬럼명으로 변경되어 stream sampling data가 조회됨. 단, fake apache log generator를 on 상태이어야 함". The table has columns: ROWTIME (TIMESTAMP), IP (VARCHAR(16)), COL1 (VARCHAR(4)), COL2 (VARCHAR(4)), DATETIME (VARCHAR(32)), COL4 (VARCHAR(8)), URI (VARCHAR(64)), and COL6 (VARCHAR(4)). The table contains several rows of data.

- Exit로 다음 진행

A screenshot of the AWS Lambda console showing the application status as "RUNNING". Above the main content area, there is a blue bar with the text "Save schema and update stream samples" and an "Exit (done)" button. Below this, there are tabs for "Raw", "Lambda output", "Formatted" (which is selected), and "Error stream".

- Go to SQL Editor



Real time analytics

Author your own SQL queries or add SQL from templates to easily analyze your source data. [Learn more](#)

[Go to SQL editor](#)

SQL 수행

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

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

Save and run SQL

Add SQL from templates Download SQL SQL reference guide Kinesis data generator tool

```

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

```

**SQL 입력 후
Save and run SQL 수행**

Application status: RUNNING

Source Real-time analytics Destination

Streaming data

● SOURCE_SQL_STREAM_001

Reference data (optional)

**fake apache log generator on 상태
유지해야 source 영역 데이터 조회**

Actions ▾

Connect reference data

Filter by column name					
ROWTIME TIMESTAMP	IP VARCHAR(16)	COL1 VARCHAR(4)	COL2 VARCHAR(4)	DATETIME VARCHAR(32)	COL4 VARCH
2020-02-19 02:09:53.592	116.91.150.158	-	-	[19/Feb/2020:02:08:31 +0000]	
2020-02-19 02:09:53.592	142.214.163.194	-	-	[19/Feb/2020:02:08:31 +0000]	
2020-02-19 02:09:53.592	21.200.185.129	-	-	[19/Feb/2020:02:08:31 +0000]	
2020-02-19 02:09:53.592	73.187.68.189	-	-	[19/Feb/2020:02:08:31 +0000]	
2020-02-19 02:09:53.592	174.34.219.121	-	-	[19/Feb/2020:02:08:31 +0000]	
2020-02-19 02:09:53.592	202.171.210.69	-	-	[19/Feb/2020:02:08:31 +0000]	

• 시간이 조금 걸린다.

Real-time analytics

Save and run SQL Add SQL from templates Download SQL SQL reference guide Kinesis data generator tool

```

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

```

Saving SQL

Running SQL

[Close](#)

Real-time analytics

Save and run SQL Add SQL from templates Download SQL SQL reference guide Kinesis data generator tool

```
1 CREATE OR REPLACE STREAM "DESTINATION_SQL_STREAM" (datetime TIMESTAMP, statusCount INTEGER);
2
3 CREATE OR REPLACE PUMP "STREAM_PUMP" AS INSERT INTO "DESTINATION_SQL_STREAM"
4 SELECT STREAM ROWTIME AS datetime,
5 COUNT(*) AS statusCount
6 FROM "SOURCE_SQL_STREAM_001"
7 GROUP BY
8 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

In-application streams:

DESTINATION_SQL_STREAM
 error_stream

Pause results New results are added every 2-10 seconds. The results below are sampled. ⓘ
 Scroll to bottom when new results arrive.

Realtime 집계 데이터가 완성 됨

ROWTIME	DATETIME	STATUSCOUNT
2020-02-19 02:13:00.0	2020-02-19 02:13:00.0	3000

• Destination 설정

destination 진행

Application status: RUNNING

Source Real-time analytics Destination

In-application streams:

DESTINATION_SQL_STREAM
 error_stream

Pause results New results are added every 2-10 seconds. The results below are sampled. ⓘ
 Scroll to bottom when new results arrive.

ROWTIME	DATETIME	STATUSCOUNT
2020-02-19 02:13:00.0	2020-02-19 02:13:00.0	3000
2020-02-19 02:14:00.0	2020-02-19 02:14:00.0	5000

Connect to destination

Destination

Kinesis data stream

Kinesis data stream is an ordered sequence of data records used for rapid and continuous data intake and aggregation.

Kinesis Firehose delivery stream

Kinesis Firehose delivery streams send source records to the destinations that you specify, automatically and continuously.

AWS Lambda function

AWS Lambda is a compute service that lets you run code without provisioning or managing servers.

Kinesis Firehose delivery stream

web-log-aggregated-data



Create new

[View web-log-aggregated-data in Kinesis Firehose](#)

In-application stream

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.

[Learn more](#)

Connect in-application stream

Choose an existing in-application stream

Specify a new in-application stream name

Use this option for in-application streams that you haven't created yet, but plan to create at a later time. Specifying a stream name ensures that you don't lose output data.

In-application stream name

DESTINATION_SQL_STREAM



Output format

JSON

CSV

Access permissions

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

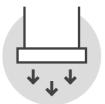
Access permissions

Create / update IAM role kinesis-analytics-web-log-aggregation-tutorial-ap-northeast-2

Choose from IAM roles that Kinesis Analytics can assume

Cancel

[Save and continue](#)



Destination

(Optional) Connect an in-application stream to a Kinesis stream, or to a Firehose delivery stream, to continuously deliver SQL results to AWS destinations. The limit is three destinations for each application.

	Destination	Destination 등록 완료	In-application stream name	ID ⓘ
<input type="radio"/> ⚙️	Firehose delivery stream web-log-aggregated-data ↗		DESTINATION_SQL_STREAM	4.1

● Destination 모니터링

s3에 에러로그 발생

```
{"attemptsMade":8,"arrivalTimestamp":1582082760539,"errorCode":"ES.ServiceException","errorMessage":"Error received from Elasticsearch cluster. {"error":{"root_cause":[{"type":"security_exception","reason":"no permissions for [indices:data/write/bulk] and User [name=arn:aws:iam::936777008077:role/firehose_delivery_role2, roles=[arn:aws:iam::936777008077:role/firehose_delivery_role2], requestedTenant=null}","type":"security_exception","reason":"no permissions for [indices:data/write/bulk] and User [name=arn:aws:iam::936777008077:role/firehose_delivery_role2, roles=[arn:aws:iam::936777008077:role/firehose_delivery_role2], requestedTenant=null}"],"status":403}","attemptEndingTimestamp":1582083361354,"rawData":"eyJEQVRFVElNRSI6IjIwMjAtMDItMTkgMDM6MjY6MDAuMDAwIiwiU1RBVFVTQ09VTlQi0jQ1MDB9","subsequenceNumber":0,"esDocumentId":"49604340806861587563985145816073632601584842809442041858.0","esIndexName":"request_data","esTypeName":""}
```

Time	Destination	Message	Error code	Version
2020-02-19T11:29:01.1T+0900D	arn:aws:es:ap-northeast-2:936777008077:domain/web-log-summary	Error received from Elasticsearch cluster. {"error":{"root_cause":[{"type":"security_exception","reason":"no permissions for [indices:data/write/bulk] and User [name=arn:aws:iam::936777008077:role/firehose_delivery_role2, roles=[arn:aws:iam::936777008077:role/firehose_delivery_role2], requestedTenant=null}],"type":"security_exception","reason":"no permissions for [indices:data/write/bulk] and User [name=arn:aws:iam::936777008077:role/firehose_delivery_role2, roles=[arn:aws:iam::936777008077:role/firehose_delivery_role2], requestedTenant=null}],"status":403}}	ES.ServiceException	1
2020-02-19T10:23:01.1T+0900D	arn:aws:es:ap-northeast-2:936777008077:domain/web-log-summary	Error received from Elasticsearch cluster. {"error":{"root_cause":[{"type":"security_exception","reason":"no permissions for [indices:data/write/bulk] and User [name=arn:aws:iam::936777008077:role/firehose_delivery_role2, roles=[arn:aws:iam::936777008077:role/firehose_delivery_role2], requestedTenant=null}],"type":"security_exception","reason":"no permissions for [indices:data/write/bulk] and User [name=arn:aws:iam::936777008077:role/firehose_delivery_role2, roles=[arn:aws:iam::936777008077:role/firehose_delivery_role2], requestedTenant=null}],"status":403}}	ES.ServiceException	1

• Elistic Search

Configure access and security

Amazon Elasticsearch Service offers numerous security features, including fine-grained access control, IAM, Cognito authentication for Kibana, encryption, and VPC access. [Learn more](#)

Network configuration

Choose internet or VPC access. To enable VPC access, we use private IP addresses from your VPC, which provides an inherent layer of security. You control network access within your VPC using security groups. Optionally, you can add an additional layer of security by applying a restrictive access policy. Internet endpoints are publicly accessible. If you select public access, you should secure your domain with an access policy that only allows specific users or IP addresses to access the domain.

- VPC access (Recommended)
 Public access

Fine-grained access control – powered by Open Distro for Elasticsearch

Fine-grained access control provides numerous features to help you keep your data secure. Features include document-level security, field-level security, read-only Kibana users, and 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 Elasticsearch internal database by creating a master username and password. After your domain is set up, you can use 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.

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

Master username
Master usernames 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

Amazon Cognito authentication

Enable to use Amazon Cognito authentication for Kibana. Amazon Cognito supports a variety of identity providers for username-password authentication. [Learn more](#)

- Enable Amazon Cognito authentication

Access policy

Access policies control whether a request is accepted or rejected when it reaches the Amazon Elasticsearch 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

Encryption

These features help protect your data. After creating the domain, you can't change most encryption settings.

Encryption Require HTTPS for all traffic to the domain i

Node-to-node encryption i

Enable encryption of data at rest i

KMS master key (Default) aws/es ▼ i

Description Default master key that protects my Elasticsearch data when no other key is defined

Account 936777008077

Key ARN arn:aws:kms:ap-northeast-2:936777008077:key/0baa82d8-cdf9-4c25-b7dc-37231b96d835

i You enabled fine-grained access control, which requires HTTPS, node-to-node encryption, and encryption at rest.

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

Review

Review the information below, and then choose **Confirm and create**.

Data nodes

[Edit](#)

Availability zones 1-AZ
Instance type r5.large.elasticsearch
Number of nodes 1

Data nodes storage

[Edit](#)

Data nodes storage type EBS
EBS volume type General Purpose (SSD)
EBS volume size 10 GiB

Snapshot configuration

[Edit](#)

Start hour for the daily automated snapshot 00:00 UTC (default)

Elasticsearch parameters

Edit

Allow APIs that can span multiple indices and bypass index-specific access policies	Enabled
Fielddata cache allocation	unbounded (default)
Max clause count	1024 (default)

Network configuration

Edit

Network access

Public access

Fine-grained access control

Edit

Fine-grained access control	Enabled
Master user type	Internal user database

Amazon Cognito authentication

Edit

Amazon Cognito for authentication	Disabled
-----------------------------------	----------

Set up access policy

Edit

```
{  
  "Version": "2012-10-17",  
  "Statement": [  
    {  
      "Effect": "Allow",  
      "Principal": {  
        "AWS": [  
          "*"  
        ]  
      },  
      "Action": [  
        "es:*"  
      ]  
    }  
  ]  
}
```

Encryption

Edit

Require HTTPS	Enabled
Node-to-node encryption	Enabled
Encryption of data at rest	Enabled
KMS master key	(Default) aws/es

Cancel

Previous

Confirm

```
/_cluster/health: {"error":{"root_cause":  
[{"type":"security_exception","reason":"no permissions  
for [cluster:monitor/health] and User  
[name=arn:aws:iam::936777008077:user/kilhan.kim,  
roles=[],  
requestedTenant=null]"}],"type":"security_exception","re
```

```
ason": "no permissions for [cluster:monitor/health] and User [name=arn:aws:iam::936777008077:user/kilhan.kim, roles=[], requestedTenant=null]", "status": 403}
```

```
/_cluster/health: {"error":{"root_cause": [{"type":"security_exception","reason":"no permissions for [cluster:monitor/health] and User [name=arn:aws:iam::936777008077:user/kilhan.kim, roles=[], requestedTenant=null]"}],"type":"security_exception","reason":"no permissions for [cluster:monitor/health] and User [name=arn:aws:iam::936777008077:user/kilhan.kim, roles=[], requestedTenant=null]},"status":403}
```

```
{"attemptsMade":8,"arrivalTimestamp":1582171582284,"errorCode":"ES.ServiceException","errorMessage":"Error received from Elasticsearch cluster.\\"error\\":{\\"root_cause\\": [{"\\type\\":\\"security_exception\\",\\\"reason\\":\\"no permissions for [indices:data/write/bulk] and User [name=arn:aws:iam::936777008077:role/firehose_delivery_role3, roles=[arn:aws:iam::936777008077:role/firehose_delivery_role3], requestedTenant=null]\\"}, {"\\type\\":\\"security_exception\\",\\\"reason\\":\\"no permissions for [indices:data/write/bulk] and User [name=arn:aws:iam::936777008077:role/firehose_delivery_role3, roles=[arn:aws:iam::936777008077:role/firehose_delivery_role3], requestedTenant=null]\\"}, {"\\status\\":403}],"attemptEndingTimestamp":1582172183356,"rawData":"eyJEQVRFVEINRSI6ljlwMjAt
```

MDItMjAgMDQ6MDY6MDAuMDAwliwiVVJJljojXcJQt1N
 UliwiTIVNQkVSQ09VTIQiOjk1Nn0=","subsequenceNum
 ber":
 0,"esDocumentId":"496043834098484029137134576
 26912153741308430566429818882.0","esIndexName"
 :"test-database","esTypeName":""}

Access policy changed

Access policy change was saved successfully. It will take some time to process the change. Your domain status will become Active again once your changes are processed.

jojuhiu-es

Edit domain Actions ▾

Modify access policy

Your changes are being processed. You can upload and search your data while we are processing the changes. The domain status will be automatically updated once processing is complete.

Overview Configuration Indices Logs Upgrade history

Elasticsearch Version

Endpoint <https://search-jujuhiu-es-cah2be20if6tylw66jhq2ha.ap-northeast-2.es.amazonaws.com>

Modify the access policy for jojuhiu-es

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

Status **Active**

Domain access policy **JSON defined access policy**

Custom access policy

JSON defined access policy

Allow open access to the domain

Copy access policy [Allow open access to the domain](#)

```

6  "Effect": "Allow",
7  "Principal": "*",
8  "Action": "es:*",
9  "Resource": "arn:aws:es:ap-northeast-2:936777008077:domain/jujuhiu-es/*"
10 }
11 ]
12 }
13 }

```

Access policy will be cleared



Changing to open access will cause your current elements to be lost. Do you want to continue?

Cancel

Clear and continue