[version_1.0.27]

Analyzing and Visualizing Streaming Data with Kinesis Data Firehose, OpenSearch Service, and OpenSearch Dashboards

---

Note: It may take up to 25 minutes for the lab to complete set up and launch.

Lab overview and objectives

Big data problems often require solutions in real time. This is the *velocity* part of the five Vs of big data (volume, variety, velocity, veracity, and value). Common data sources for these scenarios include video streams, application logs, and infrastructure devices. Data in these velocity scenarios is called *streaming data*. Amazon Kinesis is a suite of services that you can use to analyze streaming data.

In this lab, you will use Amazon Kinesis Data Streams to collect data from a web server that is hosted in Amazon Elastic Compute Cloud (Amazon EC2). You will then use AWS Lambda to process and enrich the data. You will analyze the data by using OpenSearch Dashboards, where you can build an index and then visualize the data to create dashboards. Business users will be able to access dashboards by using Amazon Cognito.

After completing this lab, you should be able to do the following:

- Describe the lab infrastructure in the AWS Management Console.
- Ingest web server logs into Amazon Kinesis Data Firehose and Amazon OpenSearch Service.
- Observe the data ingestion and transformation process by using Amazon CloudWatch log events.
- Build an index for these logs in OpenSearch Service.
- Visualize data by using OpenSearch Dashboards, including:
    - Create a pie chart that illustrates the operating systems and browsers that visitors use to consume the website.
    - Create a heat map that illustrates how users are referred to product pages (either by the search page or the recommendations page).

Duration

This lab will require approximately **90 minutes** to complete.

AWS service restrictions

In this lab environment, access to AWS services and service actions might be restricted to the ones that are needed to complete the lab instructions. You might encounter errors if you attempt to access other services or perform actions beyond the ones that are described in this lab.

Scenario

The administrator for the university bookstore's website wants to gather insights on how visitors interact with the site. She has been using a web-based JavaScript tracking system that includes charts with information about user activity. The information includes where users are located, which browsers they use, and whether they use mobile devices. The data can also indicate whether visitors reached a bookstore product's page from a search page or a recommendations page.

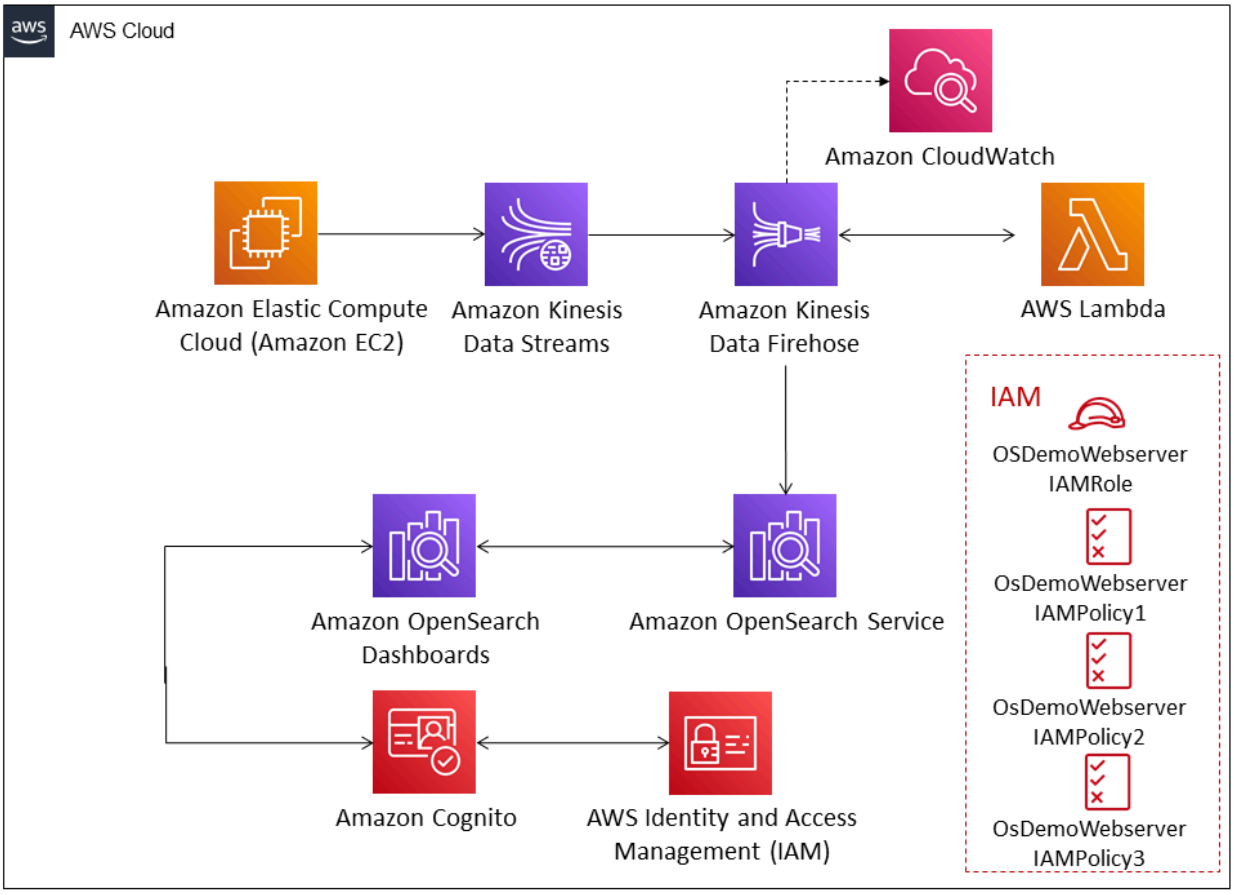
The number of visitors to the site has grown each year. However, because a lot of visitors use browsers that block third-party tracking scripts, she is now concerned that the data about user activity isn't accurate.

While researching how to get data that is more accurate, she came across the Analyze User Behavior Using Amazon Elasticsearch Service, Amazon Kinesis Data Firehose and Kibana post on the *AWS Database Blog*. The post details a solution to use streaming data to analyze the logs for a web server and determine user access patterns. The web administrator heard about your team using AWS services to create data analytics solutions. She contacts you for advice about whether the approach from the blog post could provide the same information as her current tracking system.

The blog post includes files for a basic website with a search page and a recommendation page that refer visitors to a few product pages. You decide to use this basic website to develop a proof of concept (POC) to use streaming data to analyze user activity for the bookstore website.

The solution uses Kinesis Data Firehose to ingest streamed web server access logs and then uses Lambda functions to enrich the data. After using Kinesis and Lambda to process the data, you can use OpenSearch Service to load and index the data. By using OpenSearch Dashboards, you can create visualizations of the data to provide insight about visitors to the university's websites. You can share these visualizations by using Amazon Cognito as an authentication tool and AWS Identity and Access Management (IAM) to authorize access to the dashboard.

When you *start* the lab, the environment will contain the resources that are shown in the following diagram.



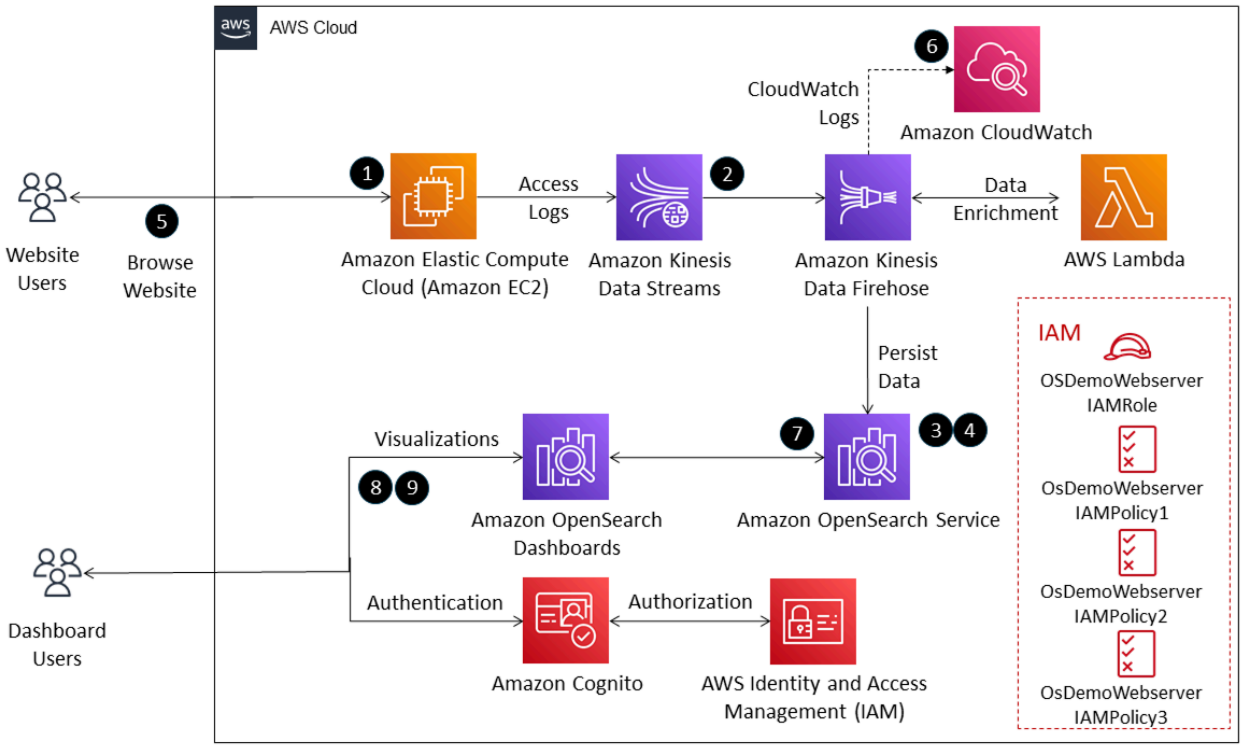
The infrastructure is designed to analyze streaming data and consists of the following components:

- An EC2 instance that runs a web server. The instance has a public subnet.

  The web server that runs on the instance includes a website with the following files. If you would like to review the website package, you can download and open this .zip file:

    - **httpd.conf:** Configuration file for the Apache web server
    - **agent.json:** Configuration file to connect the web server to a Kinesis Data Firehose delivery stream
    - **search.php:** Page where a user can search for a product
    - **recommendation.php:** Page that recommends a particular product based on the user's search history
    - **echo.php**, **kindle.php**, and **firetvstick.php:** Pages for three products on the website
- A Kinesis Data Firehose delivery stream that captures streaming data from the web server logs. You will use the delivery stream to write data to an OpenSearch Service cluster.
- A Lambda function to transform the data. If you would like to review the function, you can download and open this .zip file.
- An OpenSearch Service cluster to index and store the data.
- An OpenSearch Dashboards instance to build data visualizations and gain insights from the data.

By the *end* of the lab, you will have used the architecture to perform several tasks. The table after the diagram provides a detailed explanation of these tasks in relation to the lab architecture.



NUMBERED TASK	DETAIL
1	You will review the EC2 instance configuration for the web server. You will also review the *OsDemoWebserverIAMRole* IAM role and *OsDemoWebserverIAMPolicy* IAM policies to understand the IAM permissions that are applied to the role.
2	You will review the Kinesis data stream that is configured to capture the web access logs for users who access the website.
3	You will also review the configuration for the OpenSearch Service cluster that is used in the lab.
4	You will then set up the OpenSearch Service index.
5	After you configure everything, you will browse the website to generate access logs.
6	After you generate access logs, CloudWatch log events will also be generated in the AWS account. You will review these logs to better understand how Kinesis Data Firehose ingests these and passes them to a Lambda function for further enrichment.
7	You will create an OpenSearch Service index pattern, which is needed to create visualizations in OpenSearch Dashboards.
8	You will build a piece chart visualization in OpenSearch Dashboards.
9	You will wrap up the lab by creating a heat map visualization.

Accessing the AWS Management Console

1. At the top of these instructions, choose ▸ **Start Lab**.
    - The lab session starts.
    - A timer displays at the top of the page and shows the time remaining in the session.

      Tip: To refresh the session length at any time, choose ▸ **Start Lab** again before the timer reaches 0:00.
    - Before you continue, wait until the circle icon to the right of the AWS link in the upper-left corner turns green.
2. To connect to the AWS Management Console, choose the **AWS** link in the upper-left corner.

- A new browser tab opens and connects you to the console.

  Tip: If a new browser tab does not open, a banner or icon is usually at the top of your browser with the message that your browser is preventing the site from opening pop-up windows. Choose the banner or icon, and then choose **Allow pop-ups**.

Task 1: Reviewing the EC2 instance and its security configuration

Start by reviewing the configuration of the EC2 instance for the web server.

3. Find the public IP address for the instance.

   - To access the Amazon EC2 console, in the search box to the right of **Services**, search for and choose **EC2**.
   - In the **Resources** section, choose the **Instances (running)** link.

     A list of EC2 instances displays.
   - Choose the **Instance ID** link for the instance named **OpenSearch Demo**.

     Note: If this instance isn't listed yet, wait until it is available.
   - Copy the **Public IPv4 address** value to a text editor of your choice.

     Tip: To quickly copy the IP address, choose the icon to the left of the IP address.

     You will use this IP address later in the lab to access the web server as a visitor and generate web access logs.

4. Review the security settings that are associated with the web server instance.

   - On the lower part of the summary page for the OpenSearch Demo instance, choose the **Security** tab.
   - Under **IAM Role**, choose the **OsDemoWebserverIAMRole** link.

     The IAM console opens. Multiple IAM policies were created to control access to the resources in this lab. To view the details of each policy, you can choose the plus icon to the left of the policy name.
   - Choose the link for **OsDemoWebserverIAMPolicy1**.
   - Choose the **JSON** tab.

The IAM policy displays in JSON format.

- Review the policy.

  This IAM policy enables the webserver EC2 instance to read and write to the TempS3Bucket S3 bucket.

  Note: There are other policies referenced by the OsDemoWebserverIAMRole, and these perform other functions. If you are interested, review these and identify what they do.

Excellent! In this task, you reviewed the EC2 instance and how it is secured.

Task 2: Reviewing the Kinesis Data Firehose delivery stream

Now you will review the Kinesis Data Firehose delivery stream configuration.

5. To access the Kinesis console, in the search box to the right of **Services**, search for and choose **Kinesis**.

   Note: Multiple services start with "Kinesis," including Amazon Kinesis Data Analytics and Amazon Kinesis Video Streams. Choose the service that is only "Kinesis."

6. Review the Kinesis Data Firehose delivery stream configuration.
   - In the left navigation pane, choose **Amazon Data Firehose**.
   - Choose the **Name** link for the **os-demo-firehose-stream** delivery stream.
   - Notice in the Delivery stream details section of the page the **Source** is specified as **Direct PUT**. Direct PUT was selected because the EC2 instance is running Linux and uses Apache as the webserver. Since the EC2 instance is running Linux, you can use the Kinesis Agent (Linux) to collect the webserver logs. For more information refer to Source, Destination and Name settings in Kinesis Data Firehose Delivery Streams.
   - Notice also, the Destination is specified as **Amazon OpenSearch Service**. The delivery stream will deliver the payload to the Amazon OpenSearch Service so that you can analyze the logs later.
   - Scroll down in your browser. Notice that Monitoring is already enabled and generating Delivery stream metrics.

- On the lower part of the delivery stream details page, choose the **Configuration** tab, specifically the **Transform records** section.
- The Lambda function **os-demo-lambda-function** is associated with the delivery stream, and it is used to transform the access logs, before delivering them to the OpenSearch service for analysis.

  Note: At a high level, the Lambda function enriches the web server logs with more information. For example, the function determines the site visitor's geographical location by converting the visitor's IP address to a location. You can review details about this Lambda function by clicking on the link if you are interested.

Congratulations! In this task, you reviewed the configuration for the Kinesis Data Firehose delivery stream.

Task 3: Reviewing the OpenSearch Service cluster

Now you will review the configuration of the OpenSearch Service cluster that was created for you.

7. Review the configuration of the OpenSearch Service cluster.
   - On the **Configuration** tab for the delivery stream, in the **OpenSearch Service destination** section, under **Domain**, choose the **os-demo** link.

     The OpenSearch Service console opens.

     Review the cluster configuration.
   - In the **General information** section, copy the **OpenSearch Dashboards URL** to a text editor to use later in the lab.

     You might notice that the OpenSearch domain is deleting resources. To see the progress, choose **View details**. Await completion before moving on.

     Note: You might notice that the **Cluster health** is *Yellow*. If you choose the **Info** link to the right of that status, a panel opens and indicates that "Yellow cluster health means all primary shards are allocated, but at least one replica is not. All single-node clusters initialize with yellow cluster health." The cluster that is used in this lab is a single-node cluster; therefore, it will have yellow cluster health throughout the lab. In a production environment, you would configure the cluster with multiple nodes. Increasing the node count to 2 or more would change the cluster health to green.

8. Review the security configuration for the OpenSearch Service cluster.

    - Choose the **Security configuration** tab.

      Review the information in the **Access policy** section.

      This access policy utilizes two roles, **os_demo_firehose_delivery_role** and **osdemocognitoauthuserrole**.

      Now you will review details about these roles.

    - To access the IAM console, in the search box to the right of **Services**, search for and choose **IAM**.
    - In the left navigation pane, choose **Roles**.
    - Choose the **Role name** link for **osdemocognitoauthuserrole**.
    - Review the permissions policy that is associated with the role.

      This role allows the OpenSearch domain to use Amazon Cognito to authenticate users.

    - Return to the **Roles** page, and choose the **Role name** link for **os_demo_firehose_delivery_role**.
    - Review the permissions policy that is associated with the role, **OsdemoFirehoseIAMPolicy**.

      This role is more complex than the other role, but at a high level it allows the OpenSearch domain to:

        - Access Amazon S3 and get and write objects.
        - Get Lambda functions and use them.
        - Create CloudWatch logs.
        - Use the OpenSearch Service domain to do things like configure the domain.

Congratulations! In this task, you reviewed the configuration for the OpenSearch Service cluster.

Task 4: Configuring an OpenSearch index

In this task, you will log in to OpenSearch Dashboards and configure an index for the web server log data that is streamed from Kinesis Data Firehose.

Note: *Indexing* is the method that search engines use to organize data for fast retrieval. The resulting structure is called an index. For more information about OpenSearch indexes, see Index Data.

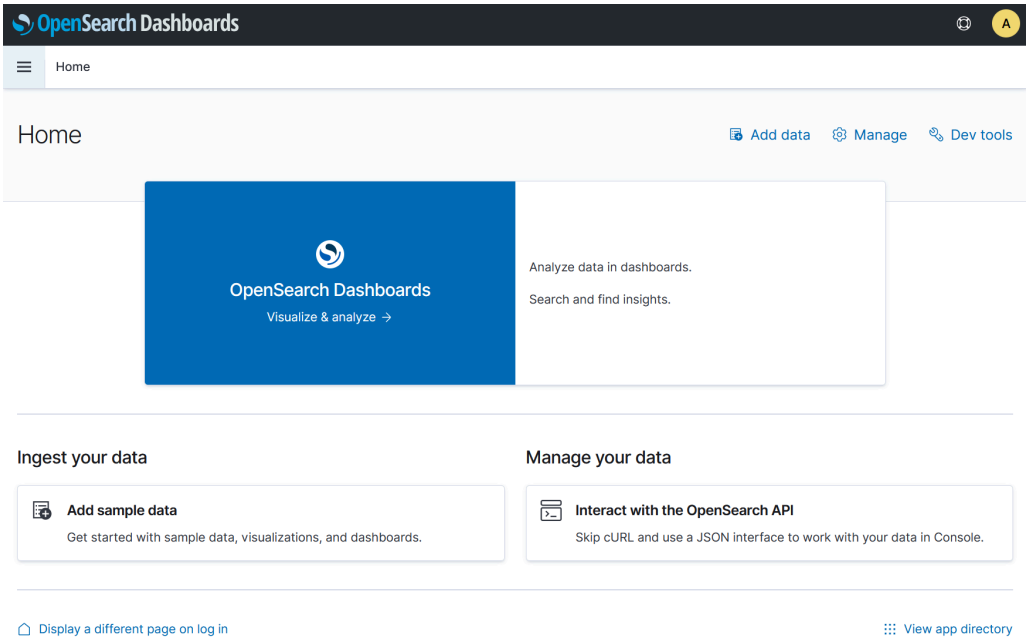
In OpenSearch, the basic unit of data is a JSON *document*. Within an index, OpenSearch identifies each document by using a unique ID.

9. Log in and change the LabUser password.
    - Open a new browser tab or window, and go to the OpenSearch Dashboards URL that you copied previously.
    - Use the following credentials to log in:
        - **Username:** `LabUser`
        - **Password:** `PassW0rd1!`

          Note: Amazon Cognito hosts this login screen. OpenSearch Service depends on Amazon Cognito for authentication and IAM for authorization.

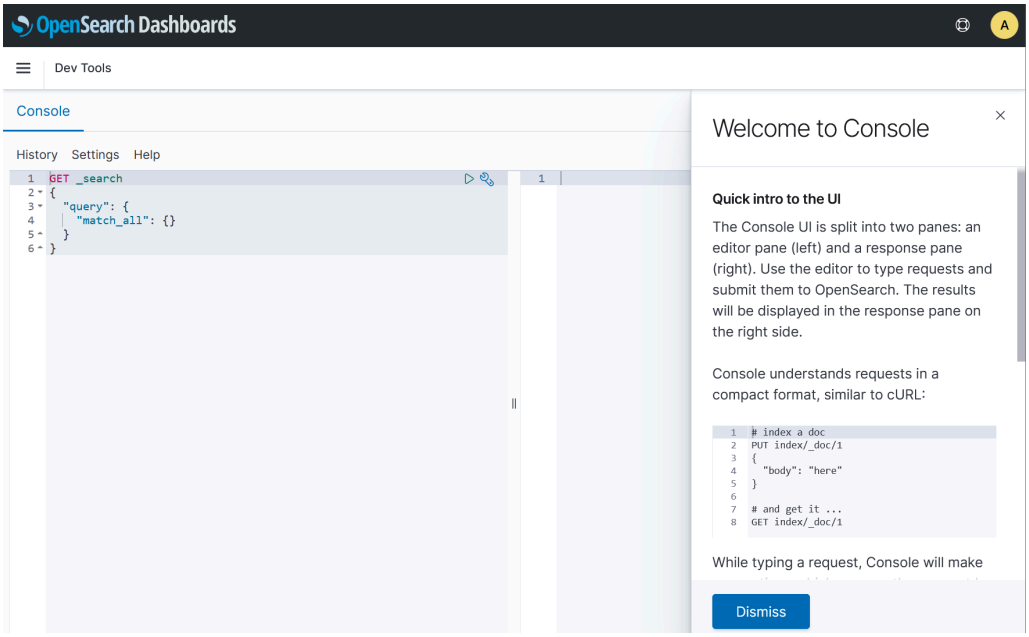
          You are prompted to change the password.
    - Enter the following for the new password: `PassW0rd1!2`
    - Choose **Send**.
    - If you are prompted with a pop-up window that asks for you to Add data or Explore on my own, choose **Explore on my own**.

      The OpenSearch Dashboards landing page displays.



10. Delete the existing web server logs.
    - Choose the menu icon (☰), and then choose **Dev Tools**.

      The Dev Tools console displays.



- In the console area, replace the existing JSON text with the following SQL command:

```
DELETE /apache_logs
```

  This command deletes the *apache_logs* index, if it already exists, that is stored on the OpenSearch Service EC2 instance.

- To run the command, choose the blue arrow icon.

  The following response displays:

```
{
  "acknowledged": true
}
```

11. Now create the OpenSearch index by using the REST API.

    Note: The PUT method is used to create the index. For more information, see Create Index.

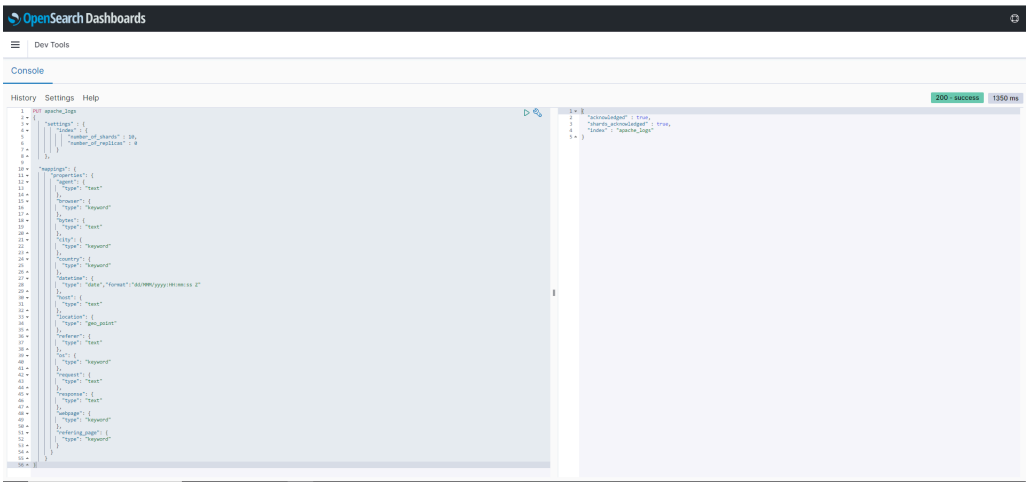
    - Copy and paste the following text into the console:

```
PUT apache_logs
{
    "settings" : {
        "index" : {
            "number_of_shards" : 10,
            "number_of_replicas" : 0
        }
    },
```

```
  "mappings": {
      "properties": {
        "agent": {
          "type": "text"
        },
        "browser": {
          "type": "keyword"
        },
        "bytes": {
          "type": "text"
        },
        "city": {
          "type": "keyword"
        },
        "country": {
          "type": "keyword"
        },
        "datetime": {
          "type":
"date","format":"dd/MMM/yyyy:HH:mm:ss Z"
        },
        "host": {
          "type": "text"
        },
        "location": {
          "type": "geo_point"
        },
        "referer": {
          "type": "text"
        },
        "os": {
          "type": "keyword"
        },
        "request": {
          "type": "text"
        },
        "response": {
          "type": "text"
        },
        "webpage": {
          "type": "keyword"
```

```
        },
        "refering_page": {
          "type": "keyword"
        }
      }
    }
}
```

Note: Copy the command exactly as written, or you might receive errors when you run the command. You should see the exact text that is shown in the following image.



- To run the command, choose the blue arrow icon.

  The following response displays:

```
{
  "acknowledged": true,
  "shards_acknowledged": true,
  "index": "apache_logs"
}
```

Analysis: This command created a new index called *apache_logs*. When the web server logs update because of website traffic, the Kinesis Data Firehose delivery stream will populate the OpenSearch Service cluster with data based on the mappings in the command. The command sets the data types for the fields in the server log files within the OpenSearch Service database. Now that you have an index, you can generate some web access logs and then create visualizations that are based on the data that is in this index.

Congratulations! In this task, you successfully created a new OpenSearch index.

Task 5: Generating web server access logs

Now that you have created the index, it's time to test your infrastructure's ability to stream web server logs through a Kinesis Data Firehose delivery stream into OpenSearch Service.

To begin the testing process, you must first generate some logs on the web server.

12. Open a new browser tab or window, and go to the following URL. Replace <PUBLIC-IP> with the public IP address that you copied previously:
`http://<PUBLIC-IP>/main.php`

13. Generate web server logs by using multiple web browsers.
    - Browse the website and access the various pages.
    - Use a different web browser to access and browse the website.

      Note: For this POC, you need data from at least two web browsers. To diversify the results, you can use multiple devices to access and browse the website.

Excellent! In this task, you populated the web server logs with data so that you can analyze the data later with OpenSearch Dashboards.

Task 6: Observing log events in CloudWatch

Now that you have generated the web server logs, it's important to understand how the data ingestion and processing phases work with your POC and the infrastructure that supports it. This part of the infrastructure is automated in such a way that visitor access logs are pulled from the web server into a Kinesis data stream. The logs are sent to Kinesis Data Firehose, and then a Lambda function processes and enriches the data. After enrichment, the logs are fed back into Kinesis Data Firehose and then to OpenSearch Service for later analysis. This entire process occurs in milliseconds and is transparent to stakeholders who will use OpenSearch Dashboards to analyze the logs.

As a data engineer, how can you observe the data being ingested, processed, and transformed in your POC? CloudWatch Logs can help you see what is happening.

In this task, you will review the CloudWatch Logs information that was generated when the web server access logs were ingested into Kinesis Data Firehose, and then transformed and enriched by Lambda.

Note: In this lab, the workflow events are combined into a log group named */aws/lambda/aes-demo-lambda-function*. The two key events that you will review in this log group are *Incoming Record from Kinesis Firehose* and *Transformed Record going back to Kinesis Firehose*.

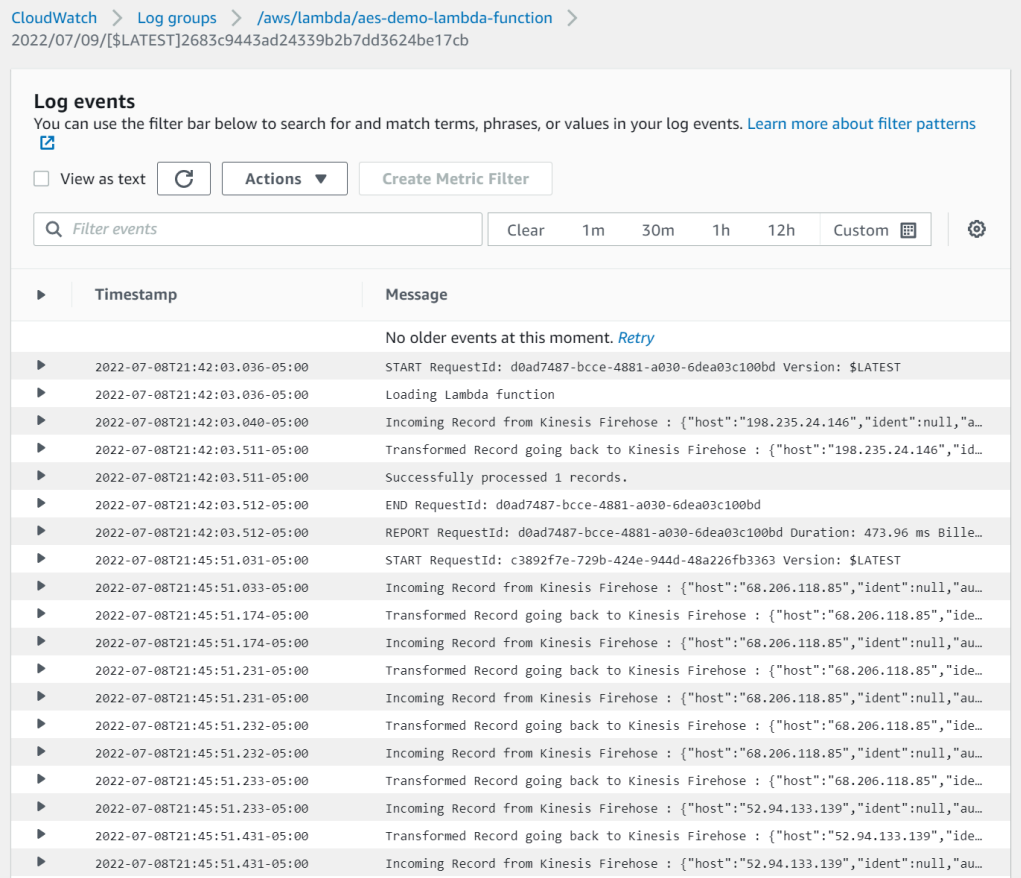
14. Access the log group.
    - Return to the tab or window where the AWS Management Console is open.
    - To access the CloudWatch console, in the search box to the right of **Services**, search for and choose **CloudWatch**.
    - In the left navigation pane, choose **Logs** > **Log groups**.
    - Choose the **Log group** link for the **/aws/lambda/aes-demo-lambda-function** log group.

15. Review a log stream.

    The **Log streams** section lists multiple log streams.

    - Choose the **Log stream** link for the most recent log stream.

      Events in the log stream are listed as shown in the following image. Each event includes a timestamp of when it was created in addition to a message with the details of the event.



16. Expand one of the logs with a message that begins with *Incoming Record from Kinesis Firehose*.

The details of the event are similar to the following:

```
Incoming Record from Kinesis Firehose :
{
    "host": "68.206.xxx.xxx",
    "ident": null,
    "authuser": null,
    "datetime": "09/Jul/2022:02:43:51 +0000",
    "request": "GET /firetvstick.php HTTP/1.1",
    "response": "200",
    "bytes": "305",
    "referer": "http://18.207.217.39/recommendation.php",
    "agent": "Mozilla/5.0 (iPhone; CPU iPhone OS 15_5 like
Mac OS X) AppleWebKit/605.1.15 (KHTML, like Gecko)
Version/15.5 Mobile/15E148 Safari/604.1"
}
```

Analysis: This CloudWatch Logs event was generated when the web server access log was sent from the EC2 instance and ingested into Kinesis Data Firehose. After being ingested, the access log data is routed to Lambda, where a function transforms and enriches the data.

17. Expand one of the logs with a message that begins with *Transformed Record going back to Kinesis Firehose*.

    The details of the event are similar to the following:

```
Transformed Record going back to Kinesis Firehose :
{
    "host": "68.206.xxx.xxx",
    "ident": null,
    "authuser": null,
    "datetime": "09/Jul/2022:02:43:51 +0000",
    "request": "GET /firetvstick.php HTTP/1.1",
    "response": "200",
    "bytes": "305",
    "referer": "http://18.207.217.39/recommendation.php",
    "agent": "Mozilla/5.0 (iPhone; CPU iPhone OS 15_5 like
Mac OS X) AppleWebKit/605.1.15 (KHTML, like Gecko)
Version/15.5 Mobile/15E148 Safari/604.1",
    "webpage": "firetvstick",
    "refering_page": "recommendation",
    "city": "Corpus Christi",
    "country": "United States",
```

```
    "browser": "Mobile Safari",
    "os": "iOS",
    "location": {
        "lat": 27.7693,
        "lon": -97.444
    }
}
```

Analysis: Notice how the data is transformed with additional fields and enriched with the location of the site visitor (by using the visitor's IP address).

18. Expand one of the logs with a message that begins with *REPORT RequestId*.

    The details of the event are similar to the following:

```
REPORT RequestId: d0ad7487-bcce-4881-a030-6dea03c100bd
 Duration: 473.96 ms Billed Duration: 474 ms Memory Size:
128 MB Max Memory Used: 51 MB  Init Duration: 454.88 ms
```

    Analysis: This event appears periodically in the log stream and includes usage information for Kinesis Data Firehose and Lambda. For example, the Billed Duration is the time that it takes Lambda to process a group of web access logs and enrich them. With Kinesis Data Firehose and Lambda, AWS customers are billed for what they use.

Awesome! You have reviewed CloudWatch log events that are associated with the Kinesis Data Firehose delivery stream and Lambda function from your POC. The delivery stream and Lambda function automatically ingested and transformed the web access logs that you created by browsing the site. You can view the log events in CloudWatch to see the process happening and get information about resource use.

Task 7: Creating the OpenSearch Service index pattern

Now that you have collected some access logs and observed how they are processed, you will create an *index pattern* in OpenSearch Service for the data. OpenSearch uses index patterns to identify the OpenSearch indices that you want to use to create visualizations. In this case, you will use the *datetime* field to filter for the data that you are interested in.

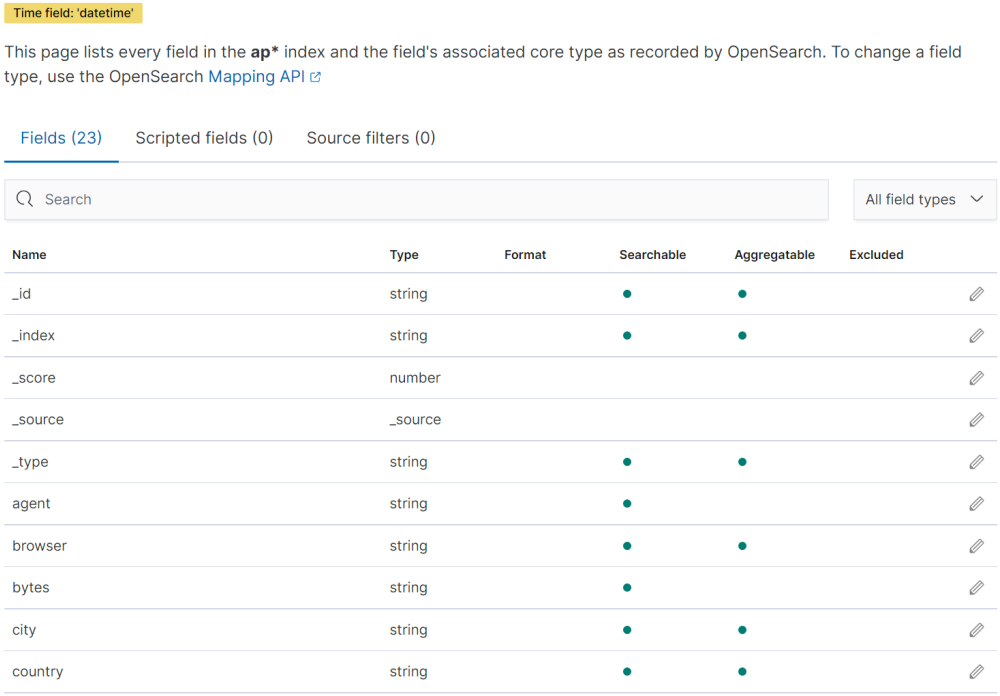
19. Create the index pattern.

    - Return to the browser window where OpenSearch Dashboards is open.

- From the navigation menu, choose **Discover**.
- Choose **Create index pattern**.
- On the step 1 of 2 page, for **Index pattern name**, enter `apache_logs`

  Note: This matches one source.
- Choose **Next step**.
- On the step 2 of 2 page, for **Time field**, choose **datetime**.
- Choose **Create index pattern**.

  OpenSearch Dashboards returns a list of fields and data types that were recorded in OpenSearch Service and shows which fields are searchable, as shown in the following image.



Perfect! In this task, you created an OpenSearch index pattern by using the datetime field that is included in the access log data.

Task 8: Creating a pie chart visualization

Now that the index is set up, you can use OpenSearch Dashboards to analyze and visualize the findings. The first visualization will address the question of whether there is a relationship between the products that customers select and the operating system or web browser that they use. In this task, you will create a stacked pie chart to gain insights to answer this question.

20. Create the visualization and select a data source.

- From the navigation menu, choose **Visualize**.
- Choose **Create new visualization**.
- Choose the **Pie** visualization type.
- Choose **apache_logs***.
- In the upper-right corner, change the time interval to the last 30 minutes.
- Choose **Update** in the lower-right corner.

21. Create the first bucket for the webpage.

    Note: When you setup a bucket for the first time it is important to understand what a bucket aggregation is. Bucket aggregations categorize sets of documents as buckets. The type of bucket aggregation determines whether a given document falls into a bucket or not.

    You can use bucket aggregations to implement faceted navigation (usually placed as a sidebar on a search result landing page) to help your users narrow down the results. To create the bucket:

    - In the **Buckets** section, choose **Add** > **Split slices**.
    - For **Aggregation**, choose **Terms**.

      Important: Don't choose Significant Terms.

    - For **Field**, choose **webpage**.
    - Keep the default values for **Order by**, **Order**, and **Size**.
    - Turn on **Group other values in separate bucket**.
    - Keep the default values for the remaining options.

22. Create a bucket for the operating system of the visitor's device.

    - Choose **Add** > **Split slices**.
    - For **Sub aggregation**, choose **Terms**.
    - For **Field**, choose **os**.
    - Keep the default values for **Order by**, **Order**, and **Size**.
    - Turn on **Group other values in separate bucket**.
    - Keep the default values for the remaining options.

23. Repeat the previous steps to add another bucket for the browser type.
24. Choose **Update** in the lower-right corner.

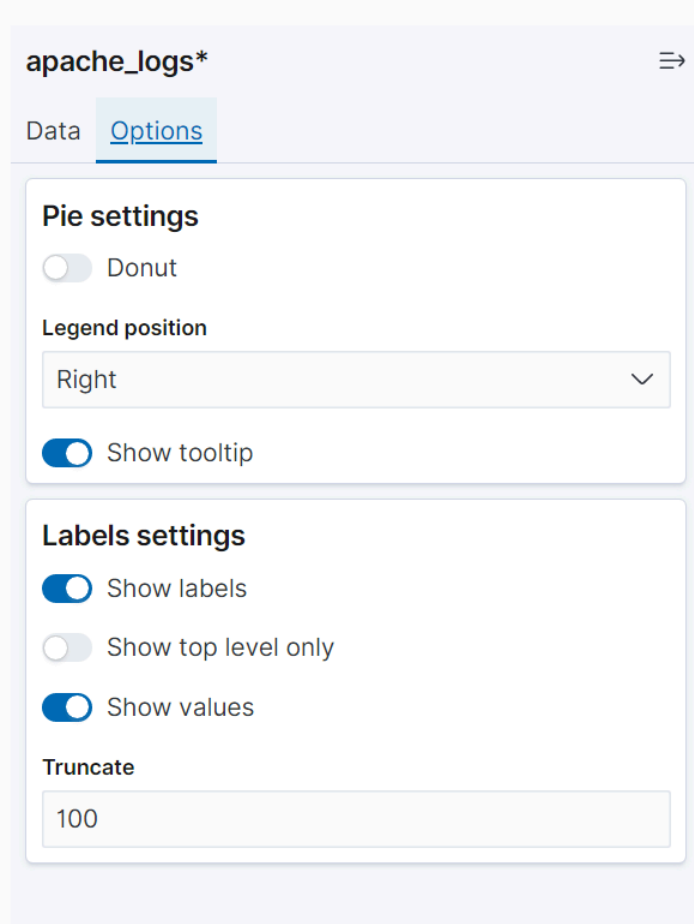
25. Configure filters for the pie chart.

    - In the upper-left corner, choose **Add filter**.
    - For **Field**, choose **webpage**.
    - For **Operator**, choose **is**.
    - For **Value**, choose **kindle**.

- Choose **Save**.

  Your filters are applied to the chart, and additional circles are added to the visualization.

26. Apply the donut style to the pie chart and configure other chart settings.
    - In the pane on the right side of the page, choose **Options**.
    - Turn off the **Donut** and **Show top level only** options as shown in the following image.

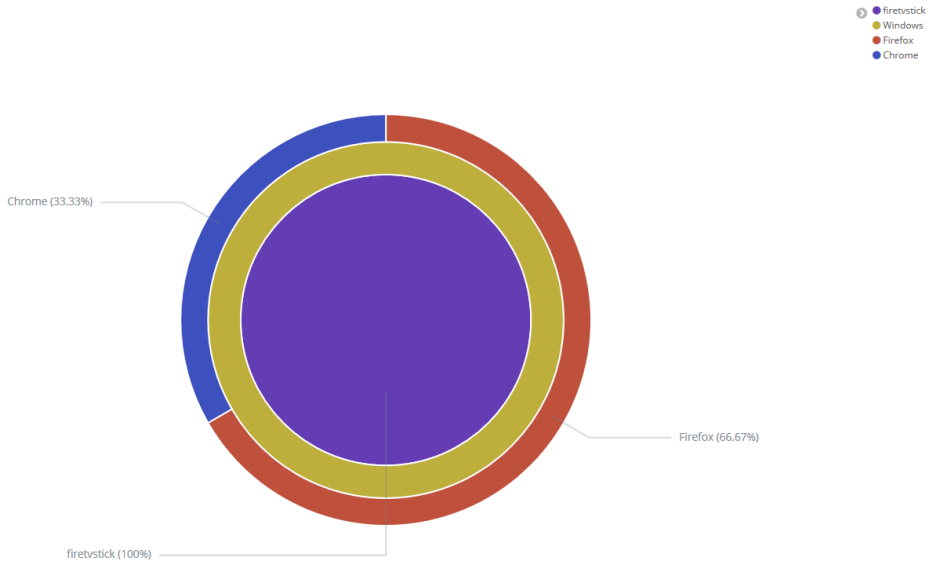


Also, it looks like the UI has been updated. These are toggles now instead of check boxes. -->

- Turn on **Show labels**.
- To apply the changes, choose **Update** in the lower-right corner.

  The data labels are added to the visualization. Your visualization should look similar to the following image.

  Note: The colors on your visualization might not match the colors in the image.

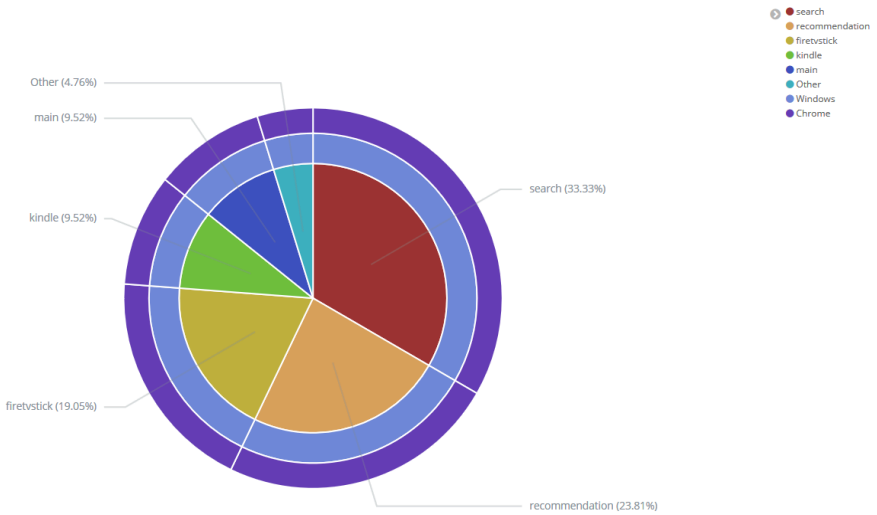


Analysis: This visualization illustrates the percentage of views for a product page from different browsers and operating systems.

27. Interact with the chart to gain insights.

    - Hover over each slice of the pie to see details about the data that is included in the slice.
    - Add a different filter to view only pages where the browser is equal to one of the browsers that you used when you generated the web log data earlier.
    - Also modify your product filter so that the other products are included. To remove the previous filter, hover on it and choose the X icon.

      The resulting pie chart is similar to the following:



Analysis: This pie chart is similar to the previous one but includes the filter for a specific browser.

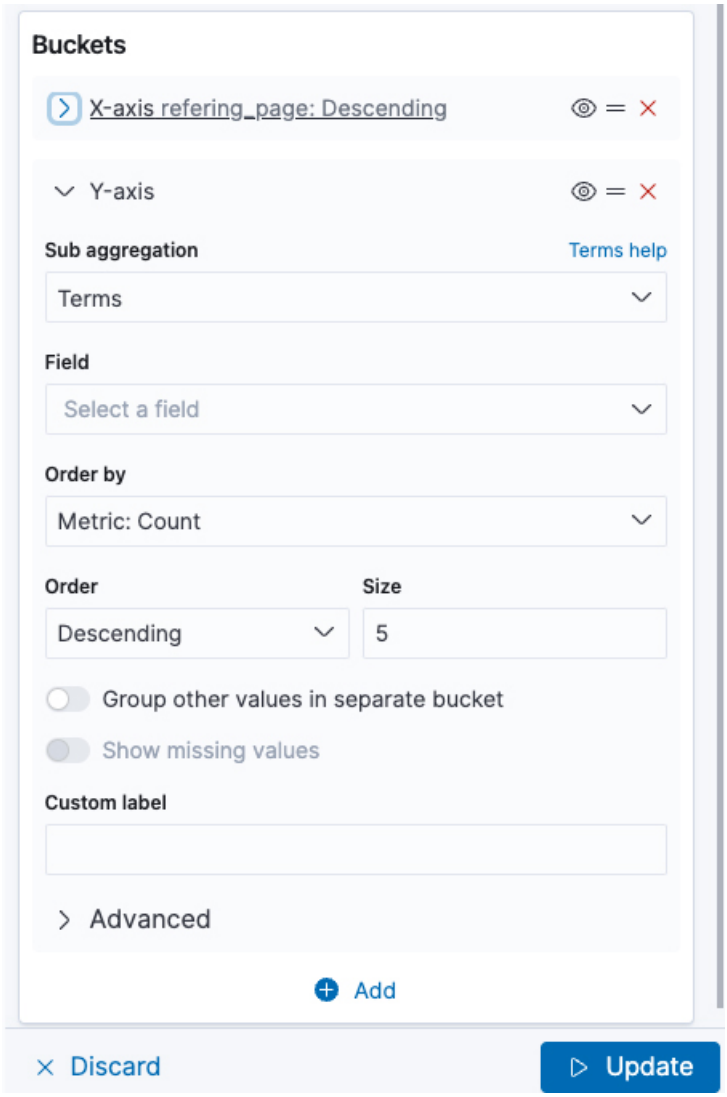
Congratulations! In this task, you successfully created a stacked pie chart to illustrate which operating systems and browsers the website visitors were using.

Task 9: Creating a heat map visualization

Now you will create a heat map visualization to address the question of whether more customers are referred to the product pages from the search page or the recommendations page.

28. Create the visualization and select a data source.
    - In the navigation menu, choose **Visualize**.
    - Choose **Create new visualization**.
    - Choose the **Heat Map** visualization type.
    - Choose **apache_logs***.
29. Create a bucket for the referring page.
    - In the **Buckets** section, choose **Add** > **X-axis**.
    - For **Aggregation**, choose **Terms**.
    - For **Field**, choose **refering_page**.
    - Keep the default values for **Order by**, **Order**, and **Size**.
    - Turn on **Group other values in separate bucket**.
    - Keep the default values for the remaining options.

      Tip: To simplify the following steps, you can collapse the first bucket that you created, as shown in the following screenshot.

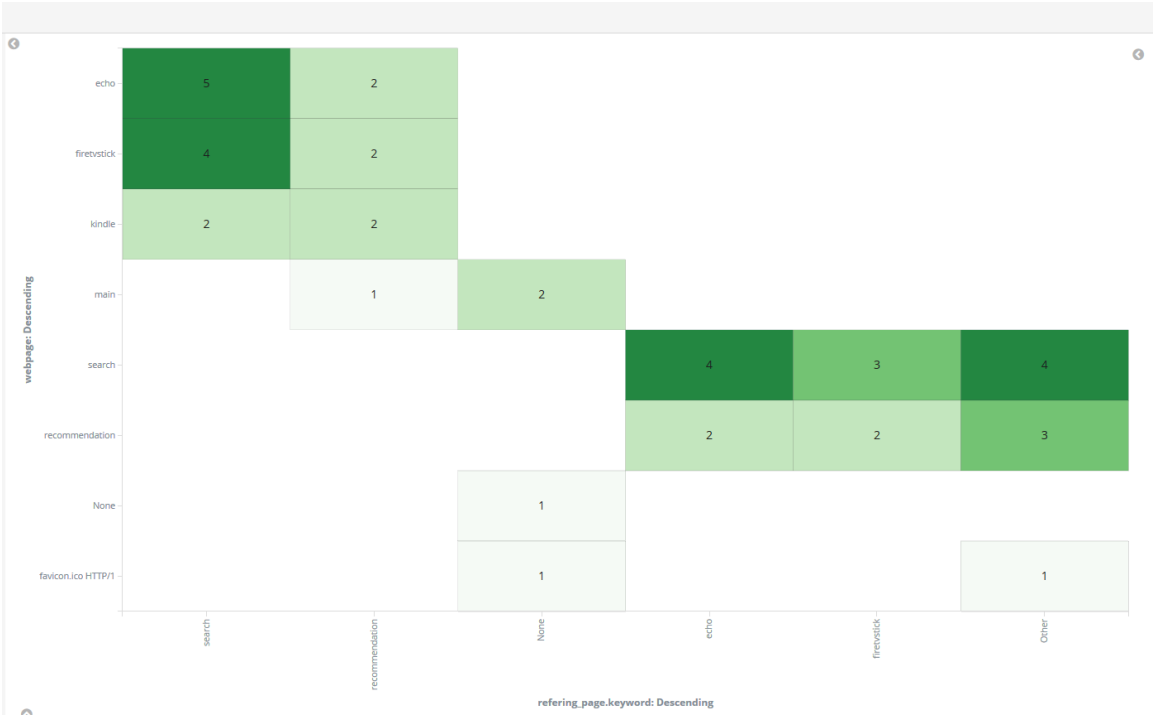


30. Create a bucket for the webpage.
    - Choose **Add** > **Y-axis**.
    - For **Sub aggregation**, choose **Terms**.
    - For **Field**, choose **webpage**.
    - Keep the default values for **Order by**, **Order**, and **Size**.
    - Turn on **Group other values in separate bucket**.
    - Choose the **Options** tab.
    - In the **Labels** section, turn on **Show labels**.
    - To apply the changes, choose **Update** in the lower-right corner.

31. In the upper-right corner, change the duration to the last 1 hour, and then choose **Refresh**.

    The heat map shows the number of webpage views in addition to whether the referrals were from the search page or the recommendations page.

    Your heat map should look similar to the following, but it will be different based on the pages that you visited when you generated the web access logs and the browsers that you used.



Analysis: Based on this image of the visualization, visitors accessed the Echo and FireStick product pages more often from the search page than the recommendations page. The team could infer that the search page is more effective than the recommendations page at directing users to the product pages.

Congratulations! In this task, you created a heat map to gain insights into whether more customers were referred to product pages from the search page or the recommendations page. Your POC to demonstrate how to use Kinesis Data Firehose and OpenSearch Service to analyze streaming data from a website was successful.

Update from the team

You share the POC results with the administrator for the university bookstore's website. She is excited to implement the infrastructure to analyze the streaming data from the web server access logs.

In this lab, you configured OpenSearch Service to use an index for web access log data. You observed how Kinesis Data Firehose ingested that data and then Lambda transformed it quickly. You also created visualizations in OpenSearch Dashboards to analyze your data and generate insights.

Submitting your work

32. To record your progress, choose **Submit** at the top of these instructions.
33. When prompted, choose **Yes**.

    After a couple of minutes, the grades panel appears and shows you how many points you earned for each task. If the results don't display after a couple of minutes, choose **Grades** at the top of these instructions.

    ⚠ **Important:** Some of the checks made by the submission process in this lab will only give you credit if it has been at least 5 minutes since you completed the action. If you do not receive credit the first time you submit, you may need to wait a couple minutes and the submit again to receive credit for these items.

    Tip: You can submit your work multiple times. After you change your work, choose **Submit** again. Your last submission is recorded for this lab.
34. To find detailed feedback about your work, choose **Submission Report**.

Lab complete

Congratulations! You have completed the lab.

35. At the top of this page, choose ■ **End Lab**, and then choose **Yes** to confirm that you want to end the lab.

A message panel indicates that the lab is terminating.

36. To close the panel, choose **Close** in the upper-right corner.

Additional Resources

For more information about the services and concepts covered in this lab, see the following resources:

- Fine-grained access control in Amazon OpenSearch Service
- Bucket aggregations