Log Analysis using AWS OpenSearch

•••

Team:
Jyotika Hariom Patil
Leo Mei
Mayuri Praveen Shimpi

Background

In one of his articles, Jon Handler mentions Amazon Elasticsearch (Amazon ES) provides opportunities to constant application monitoring. With this abundance of chances comes spread; developers are using Amazon ES for a wide range of responsibilities.

He discusses Amazon Kinesis Firehose which provides reliable, serverless delivery of Apache web logs (or other log data) to Amazon Elasticsearch Service.

M. Shrivastava also discuss Amazon Kinesis agent to stream log records to Amazon Kinesis Firehose delivery stream. The idea is to use the Firehose delivery broadcast to record information on the Amazon ES domain and create a broadcast experience using Kibana to understand trends and patterns about the product.

Objective

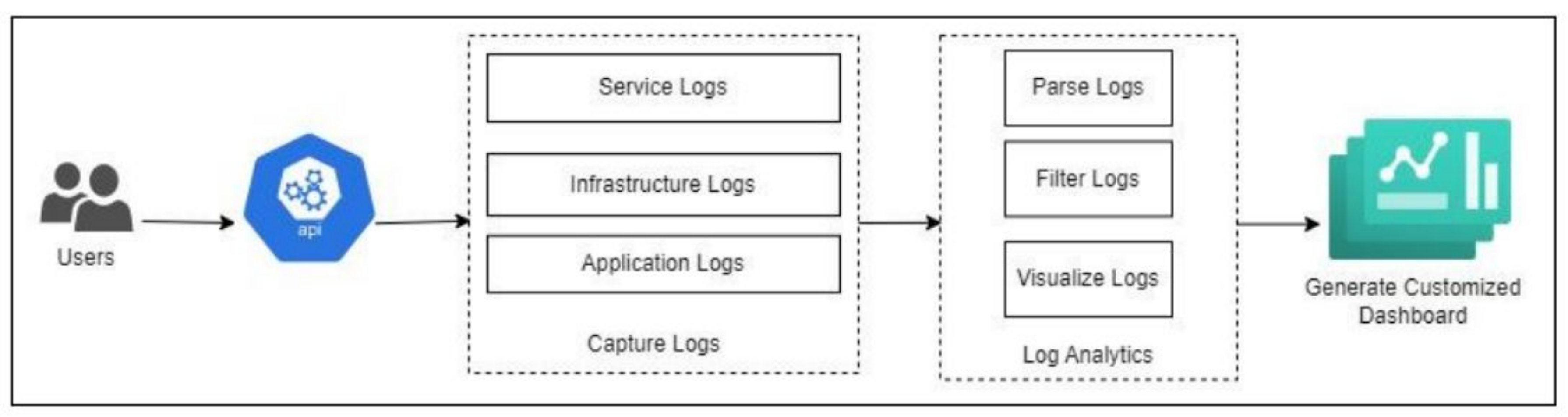
- Deploy a web application on AWS with autoscaling, load balancing, and high availability

- Provide a centralized location for developers to view application logs, infrastructure logs, and AWS service logs.

- Develop graphical dashboards to display the logs, aiding in the monitoring, troubleshooting, alerting, analysis, and resolution of issues in the application, its infrastructure, and AWS services.

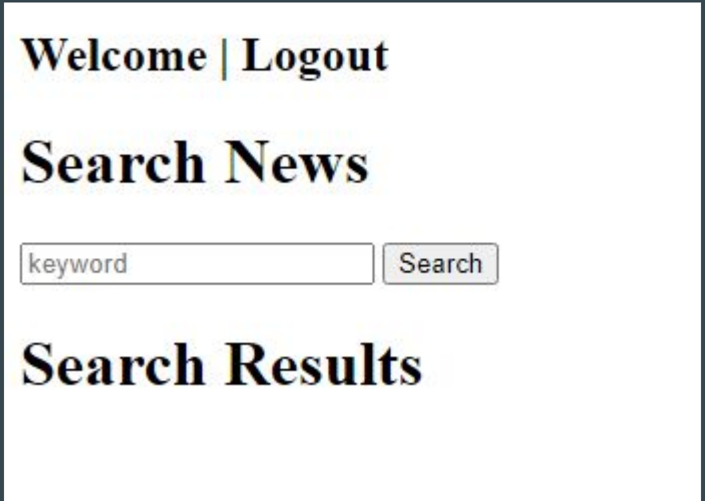
Functionality

Functional Diagram



Application

- A simple web application in which allows users to sign in and search for news with keywords.
- Fetches news article from Google Gnews
- Developed in Java Spring Boot
- Deployed Docker container.



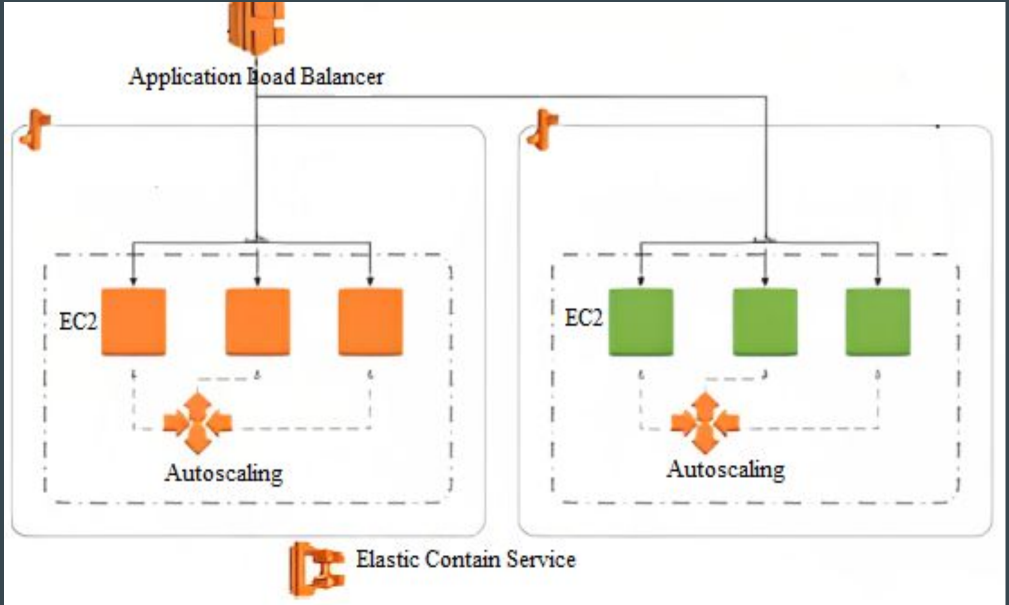
Welcome | Logout

Search News

Search Results

Infrastructure

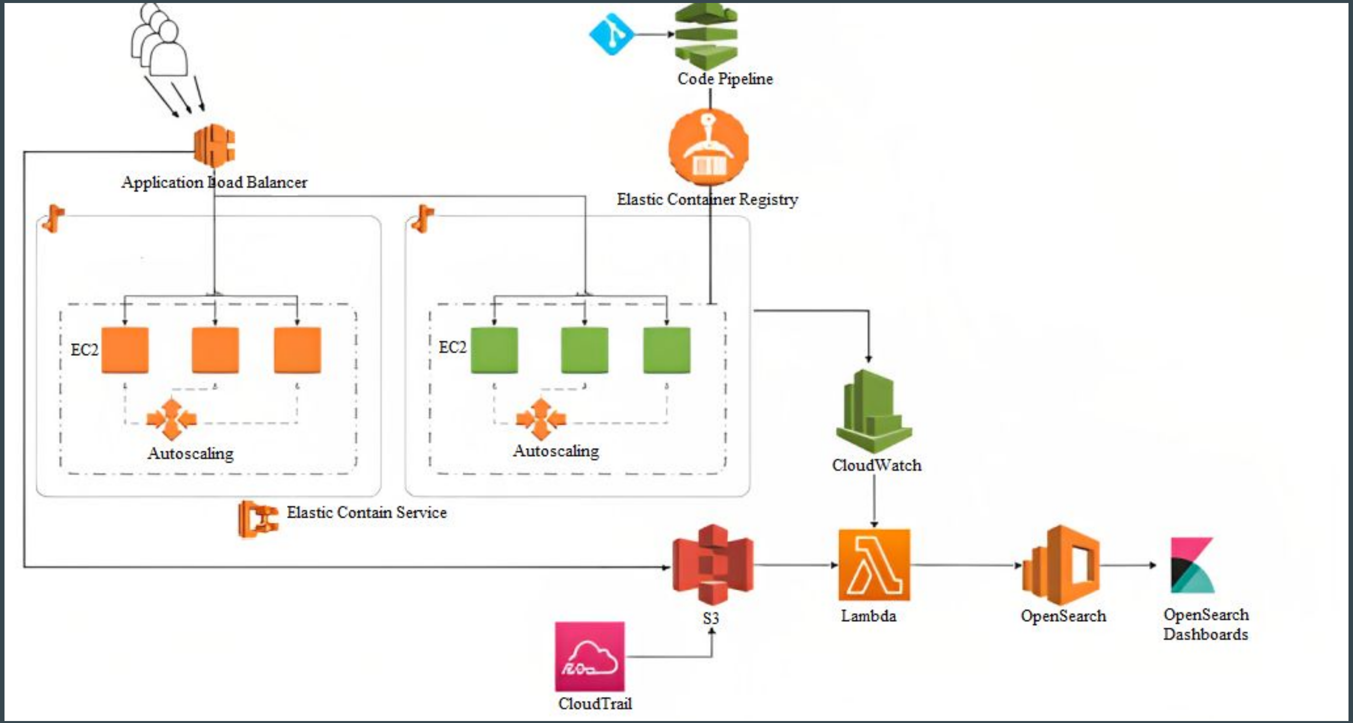
- Elastic Container Service (ECS)
- EC2 Instances
- Virtual Private Cloud (VPC)
- Application Load Balancer (ALB)



AWS Services

- S3
- CloudWatch
- IAM
- Auto Scaling
- Lambda
- CodePipeLine
- CloudTrail
- Amazon OpenSearch service and dashboard (Kibana)
- EC2

Architecture



Implementation

Build a simple web application in Java Spring Boot that fetches news articles from Google Gnews.

Deployed it in Docker Container

Used CodeDeploy for continuous deployment on AWS EC2

Extracted different logs:

- Application Logs - AWS CloudWatch
- Service Logs - AWS CloudTrail, VPC Flow Data and S3
- Infrastructure Logs - AWS S3
- Altering mechanism - AWS OpenSearch

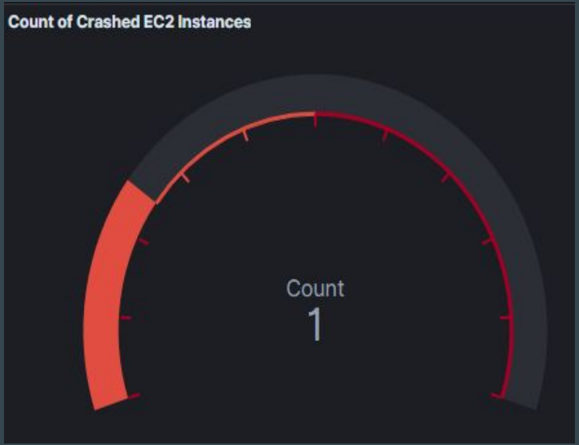
We aggregated, monitored and build Log Analytics Dashboard on AWS OpenSearch Dashboards to gather insights about application performance, users, environment and infrastructure.

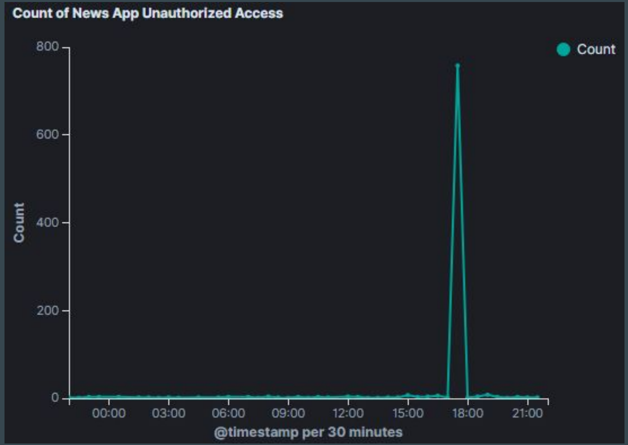
Testing

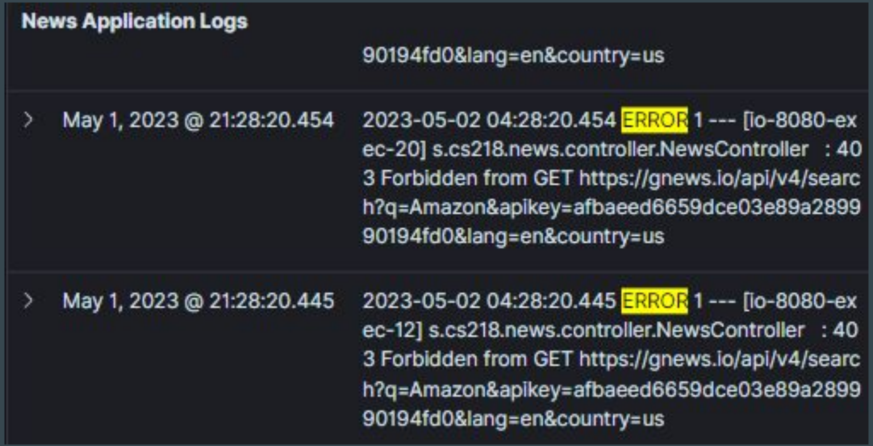
- Load Testing using Locust
- Application testing using Postman
- Lambda testing using event trigger

Dashboard

- Application

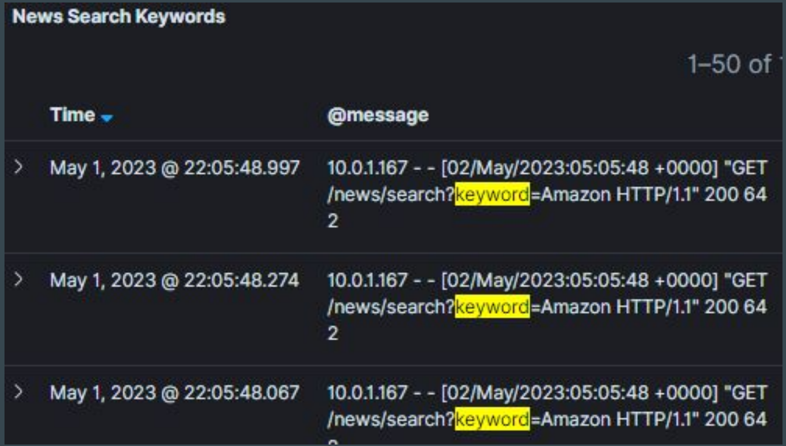






News Application Logs

```
                          90194fd0&lang=en&country=us

>  May 1, 2023 @ 21:28:20.454   2023-05-02 04:28:20.454 ERROR 1 --- [io-8080-ex
                                ec-20] s.cs218.news.controller.NewsController   : 40
                                3 Forbidden from GET https://gnews.io/api/v4/searc
                                h?q=Amazon&apikey=afbaeed6659dce03e89a2899
                                90194fd0&lang=en&country=us

>  May 1, 2023 @ 21:28:20.445   2023-05-02 04:28:20.445 ERROR 1 --- [io-8080-ex
                                ec-12] s.cs218.news.controller.NewsController   : 40
                                3 Forbidden from GET https://gnews.io/api/v4/searc
                                h?q=Amazon&apikey=afbaeed6659dce03e89a2899
                                90194fd0&lang=en&country=us
```

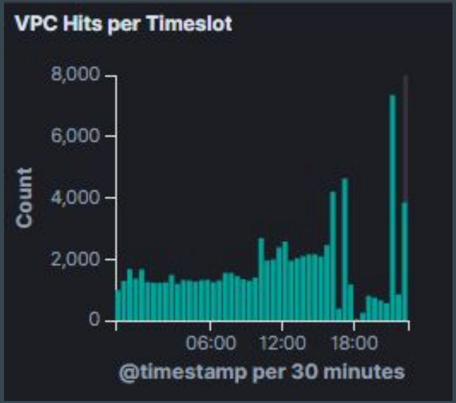


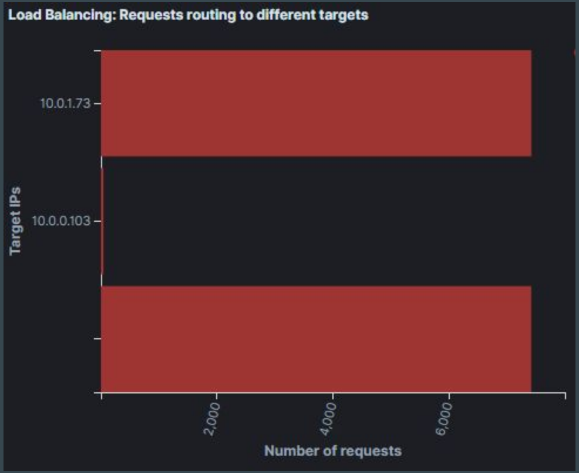
News Search Keywords

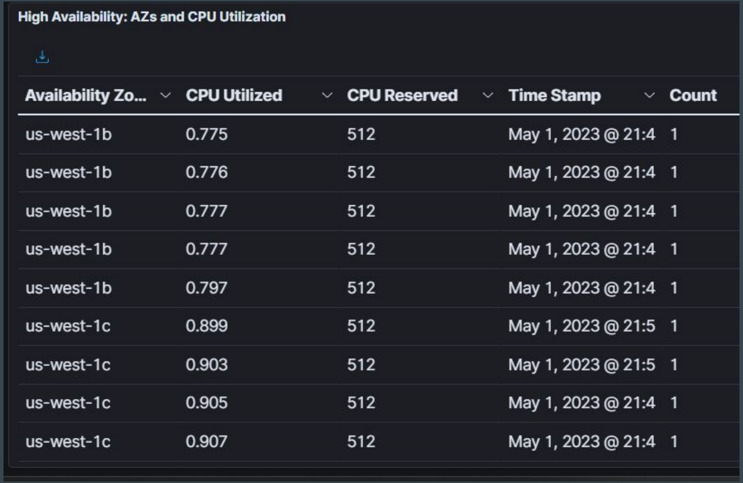
```
                                                                   1–50 of

   Time                         @message

>  May 1, 2023 @ 22:05:48.997   10.0.1.167 - - [02/May/2023:05:05:48 +0000] "GET
                                /news/search?keyword=Amazon HTTP/1.1" 200 64
                                2

>  May 1, 2023 @ 22:05:48.274   10.0.1.167 - - [02/May/2023:05:05:48 +0000] "GET
                                /news/search?keyword=Amazon HTTP/1.1" 200 64
                                2

>  May 1, 2023 @ 22:05:48.067   10.0.1.167 - - [02/May/2023:05:05:48 +0000] "GET
                                /news/search?keyword=Amazon HTTP/1.1" 200 64
```

Dashboard

- Infrastructure

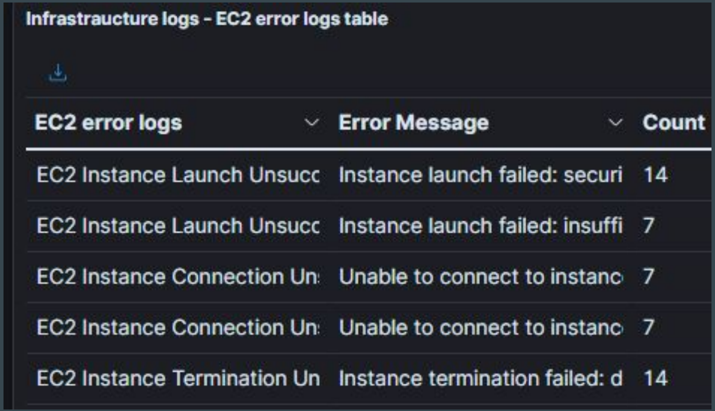






High Availability: AZs and CPU Utilization

Availability Zo...	CPU Utilized	CPU Reserved	Time Stamp	Count
us-west-1b	0.775	512	May 1, 2023 @ 21:4	1
us-west-1b	0.776	512	May 1, 2023 @ 21:4	1
us-west-1b	0.777	512	May 1, 2023 @ 21:4	1
us-west-1b	0.777	512	May 1, 2023 @ 21:4	1
us-west-1b	0.797	512	May 1, 2023 @ 21:4	1
us-west-1c	0.899	512	May 1, 2023 @ 21:5	1
us-west-1c	0.903	512	May 1, 2023 @ 21:5	1
us-west-1c	0.905	512	May 1, 2023 @ 21:4	1
us-west-1c	0.907	512	May 1, 2023 @ 21:4	1

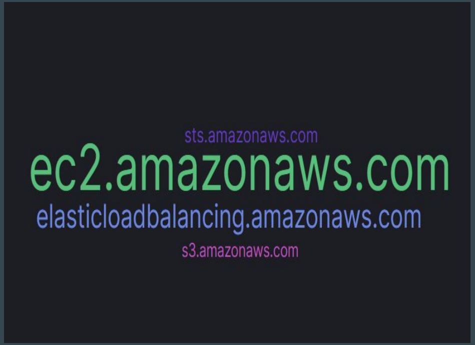


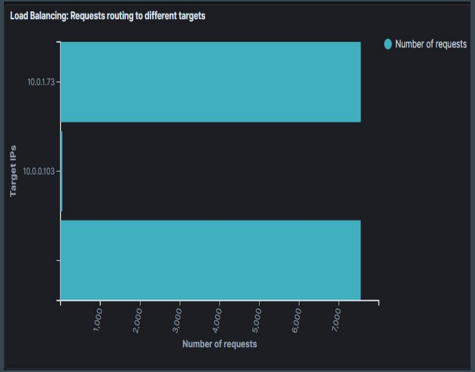
Infrastraucture logs - EC2 error logs table

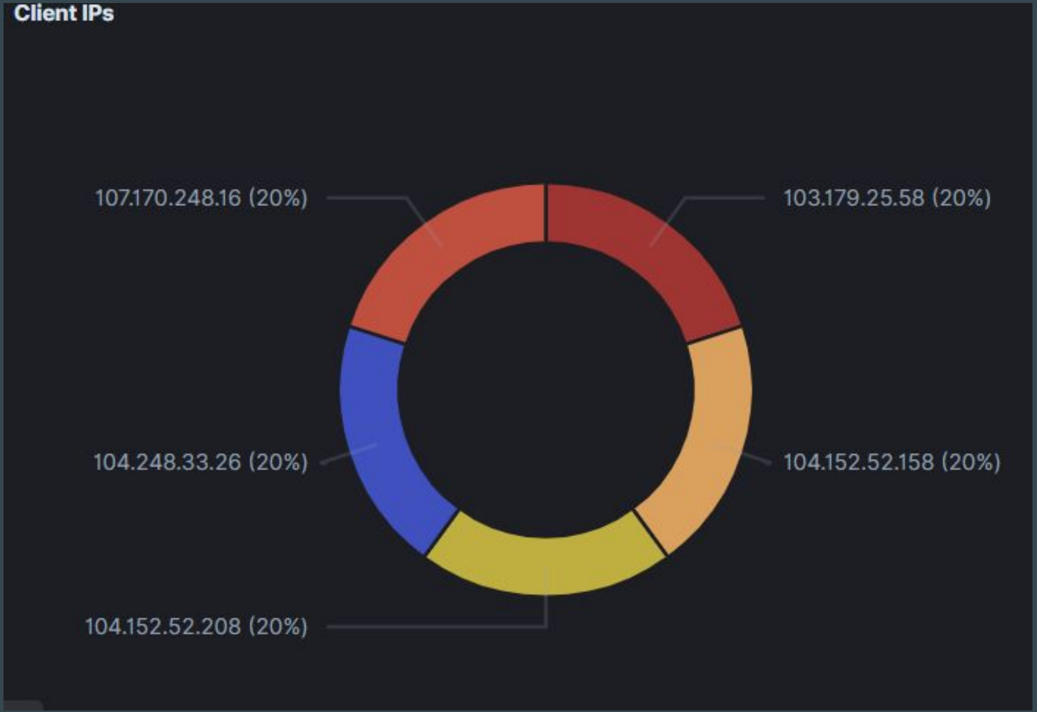
EC2 error logs	Error Message	Count
EC2 Instance Launch Unsucc	Instance launch failed: securi	14
EC2 Instance Launch Unsucc	Instance launch failed: insuffi	7
EC2 Instance Connection Un:	Unable to connect to instanc	7
EC2 Instance Connection Un:	Unable to connect to instanc	7
EC2 Instance Termination Un	Instance termination failed: d	14

Dashboard

- AWS Services




Load Balancing: Requests routing to different targets


Client IPs

DEMO

Postmortem

- Deployment of the news application
  - Experienced difficulty building the Docker image using CodeBuild
  - Misconfigured health check caused EC2 instances to drain and ECS to restart repeatedly
- Log Visualization
  - OpenSearch DSL was insufficient for gathering the data we required
  - Should have preprocessed logs using AWS Lambda before sending them to OpenSearch
- AWS Services
  - Learned valuable lessons about the importance of configuring all individual services before integrating them into the system
  - Lack of documentation from AWS OpenSearch made the team difficult to identify problems

Conclusion

We Auto-deployed a web application to AWS ECS

We aggregated, monitored and build Log Analytics Dashboard on AWS OpenSearch Dashboards to gather insights about application performance, users, environment and infrastructure.

We build Lambdas to parse and filter these logs before ingesting them in AWS OpenSearch as the logs were difficult to understand and we found that querying it with DSL was insufficient.

Finally, we build visualizations and set up alerts in the OpenSearch Dashboard