AWS Use Case – 2

AWS Network Analysis with VPC FlowLog,

CloudWatch Insight and Athena

In today's digital landscape, understanding network traffic is crucial for maintaining security, performance, and compliance within your cloud infrastructure. AWS provides a robust suite of tools to facilitate comprehensive network analysis, enabling organizations to gain deep insights into their Virtual Private Cloud (VPC) activities.

This guide will introduce you to a practical use case of using **AWS VPC Flow Logs, CloudWatch Insights**, and **Athena** for comprehensive network analysis. We will cover the following:

**1. Overview of AWS Network Analysis Tools**

* Introduction to VPC Flow Logs
* Role of CloudWatch Insights
* Utilization of Athena for querying and analysis

**2. Setting Up VPC Flow Logs**

* Enabling VPC Flow Logs
* Configuring log delivery destinations

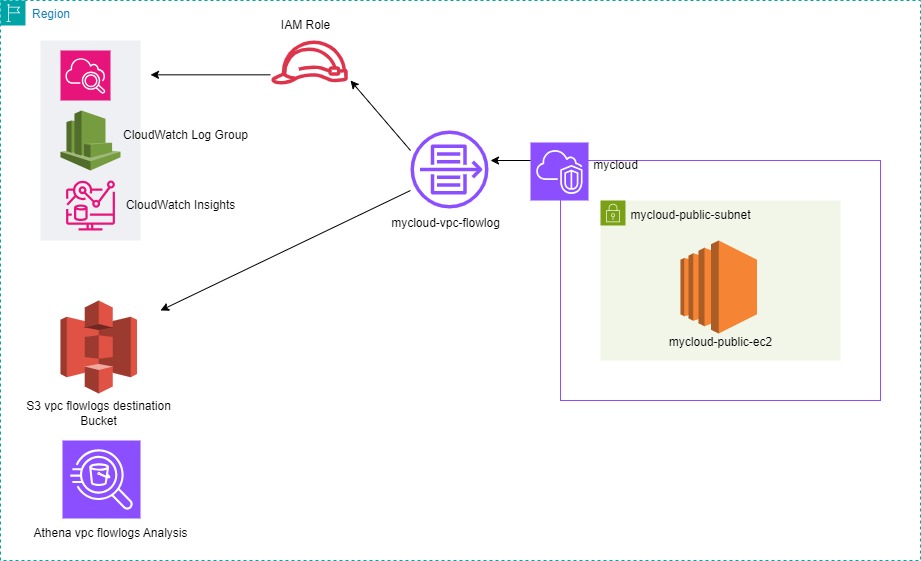
**3. Analysing Logs with CloudWatch Insights**

* Querying logs in CloudWatch
* Creating and interpreting visualizations

**4. Advanced Log Analysis with Athena**

* Setting up Athena for log analysis
* Writing and executing SQL queries
* Example use cases and queries

**Reference Diagram**



Let’s Begin!

**Overview of AWS Network Analysis Tools**

* **Introduction to VPC Flow Logs**

AWS VPC Flow Logs is a feature that enables you to capture information about the IP traffic going to and from network interfaces in your Virtual Private Cloud (VPC).This includes traffic data for each network interface, allowing for detailed analysis and troubleshooting of network connectivity and security issues.

**You can create a flow log for a VPC, a subnet, or a network interface**.

* **Role of CloudWatch Insights**

CloudWatch Insights is a powerful tool for log analytics and visualization within the AWS ecosystem. It allows users to run queries on log data stored in CloudWatch Logs, create dashboards, and generate alerts based on specific log patterns or thresholds

* **Utilization of Athena for Querying and Analysis**

Athena is an interactive query service that makes it easy to analyse data directly in Amazon S3 using standard SQL. Athena is serverless, so there is no infrastructure to manage, and you pay only for the queries you run. It is particularly useful for advanced querying and analysis of large sets of log data.

**Setting Up VPC Flow Logs**

* Create a VPC (mycloud)
* Create an EC2 Instance(mycloud-public-ec2) and configured with outbound internet access.
* Launch a “Hello world” simple html webpage on it.

**Configuring Log Delivery Destinations**

We have to create a CloudWatch log group and an S3 bucket as destinations.

**CloudWatch Logs:**

Facilitates real-time monitoring and alerting.

* Navigate to the CloudWatch dashboard
* under **Logs**, click on **Log groups**.
* Enter a **Log group name**. Make sure the name is unique within your AWS account and region – mycloud-vpc-flowlogs.
* Retention Setting – choose 1 day
* lick on the **Create log group** button.

**Amazon S3:**

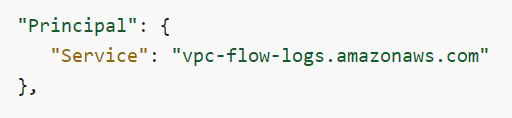
Provides a scalable storage solution for long-term log retention and complex querying with Athena.

* Create a S3 bucket with name “mycloud-vpc-flowlogs”.

**Publish FlowLogs to CloudWatch Logs**

**IAM role for publishing flow logs to CloudWatch Logs**

* The IAM role that's associated with your flow log must have sufficient permissions to publish flow logs to the specified log group in CloudWatch Logs. The IAM role must belong to your AWS account.
* Create an IAM Role – “**Mycloud-vpc-flowlogs-role**” with permission “**CloudWatchFullAccess”**
* Ensure that your role has the following trust policy, which allows the flow logs service to assume the role.
* For **Trusted entity type**, choose **Custom trust policy**. For **Custom trust policy**, replace "Principal": {}, with the following, then and choose **Next**.





**To create a flow log using the console**

* Open the Amazon VPC console – And choose your VPC which needs to create FlowLogs.
* Choose **Actions**, **Create flow log**.
* For **Filter**, specify the type of traffic to log. Choose **All** to log accepted and rejected traffic, **reject** to log only rejected traffic, or **accept** to log only accepted traffic – Here we choose All.
* For **Maximum aggregation interval –** 1min.
* For Destination, choose Send to CloudWatch Logs.
* For **Destination log group**, choose the name of an existing log group – mycloud-vpc-flowlogs.
* For **IAM role –** Choose **Mycloud-vpc-flowlogs-role.**
* For **Log record format –** Choose Default.
* Choose **Create flow log**.

**Analysing Logs with CloudWatch Insights**

**Querying logs in CloudWatch**

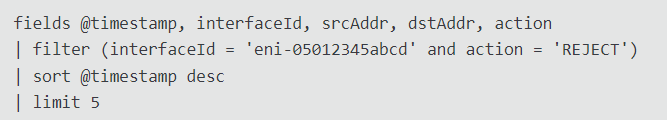
* Open the CloudWatch console
* In the navigation pane, choose **Logs**, **Log groups**.
* Select the name of the log group - mycloud-vpc-flowlogs.
* Select the name of the log stream that contains the flow log records.
* There we can see a log stream for our EC2 instances elastic network interface.

**Run a Query**

* Select Logs, Logs Insights.
* On the **Logs Insights** dashboard, select the log group.

Here we consider a scenario;

If you experiencing intermittent timeouts on a given elastic network interface. The following query checks for any rejects on the elastic network interface over a period of time.



Query Breakdown:

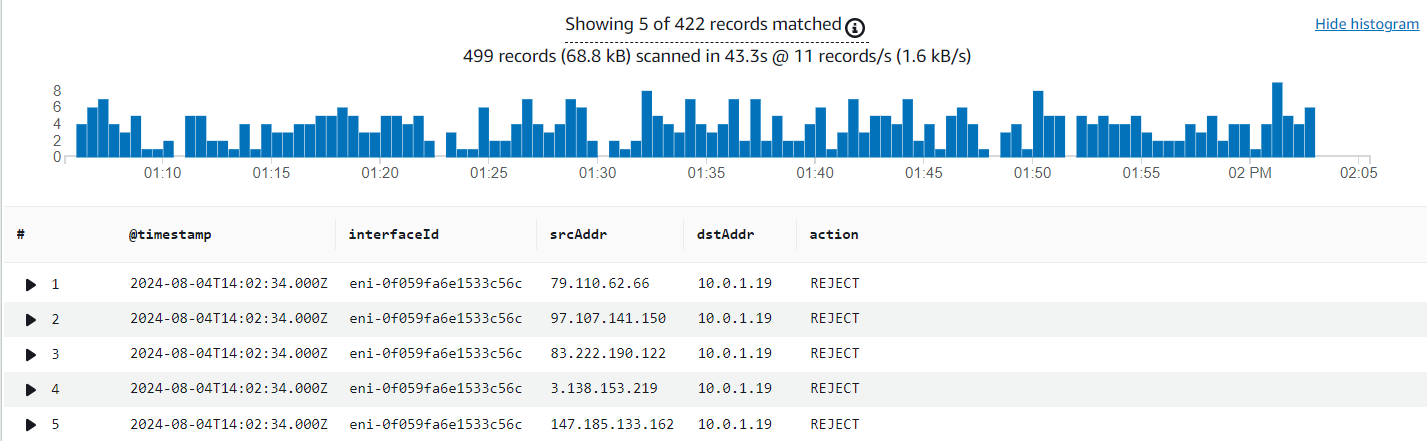
**fields @timestamp, interfaceId, srcAddr, dstAddr, action**: This specifies which fields to include in the query results. Here, you’re selecting the timestamp, network interface ID, source and destination addresses, and action.

**filter (interfaceId = 'eni-05012345abcd' and action = 'REJECT')**: This filters the log entries to include only those where the network interface ID matches 'eni-05012345abcd' and the action is 'REJECT'. It helps in narrowing down to specific traffic that was rejected on a particular network interface.

**sort @timestamp desc**: This sorts the results by the timestamp in descending order. The most recent log entries will appear first.

**limit 5**: This limits the number of results returned to 5. It helps in managing the volume of data you see, focusing only on the most recent 5 rejected actions.

Got a result as below;



Here we can see the 5 source IP addresses are blocked by the eni with a visualized histogram.

**Advanced Log Analysis with Athena**

**Create a flow log that publishes to Amazon S3**

* Open the Amazon VPC console
* Choose **Actions**, **Create flow log**.
* For **Filter**, Choose All
* or **Maximum aggregation interval**, choose 1 min.
* For **Destination**, choose **Send to an Amazon S3 bucket** – mycloud-vpc-flowlogs.
* For **S3 bucket ARN**, specify the Amazon Resource Name (ARN) of an existing Amazon S3 bucket.

E.g.: arn:aws:s3:::mycloud-vpc-flowlogs

* For **Log record format –** use the default**.**
* For **Log file format**, specify the format for the log file.

Choose **Text** – Plain text. This is the default format.

* Leave other options as default
* Choose **Create flow log**.

**Setting Up Athena for Log Analysis**

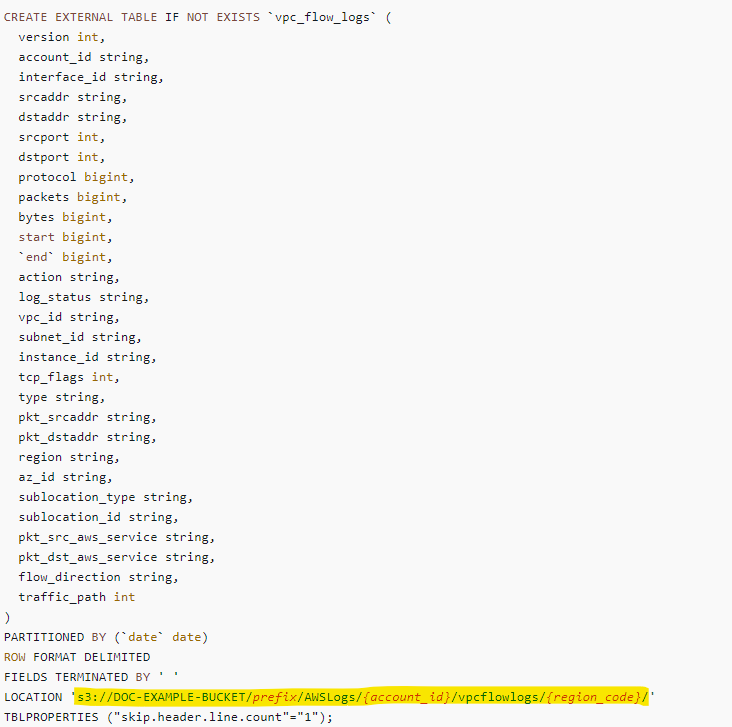
**Creating and querying tables for custom VPC flow logs**

When you create a VPC flow log, you can use a custom format when you want to specify the fields to return in the flow log and the order in which the fields appear.

Modify the LOCATION 's3://DOC-EXAMPLE-BUCKET/*prefix*/AWSLogs/*{account\_id}*/vpcflowlogs/*{region\_code}*/' to point to the Amazon S3 bucket that contains your log data.

Run the query in Athena console. After the query completes, Athena registers the vpc\_flow\_logs table, making the data in it ready for you to issue queries.

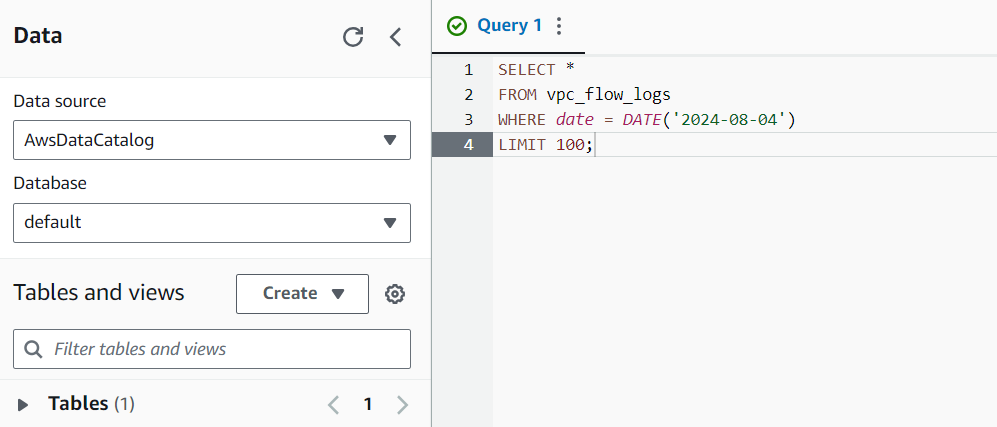
***E.g.: s3://mycloud-vpc-flowlogs/AWSLogs/891377330056/vpcflowlogs/eu-north-1/***



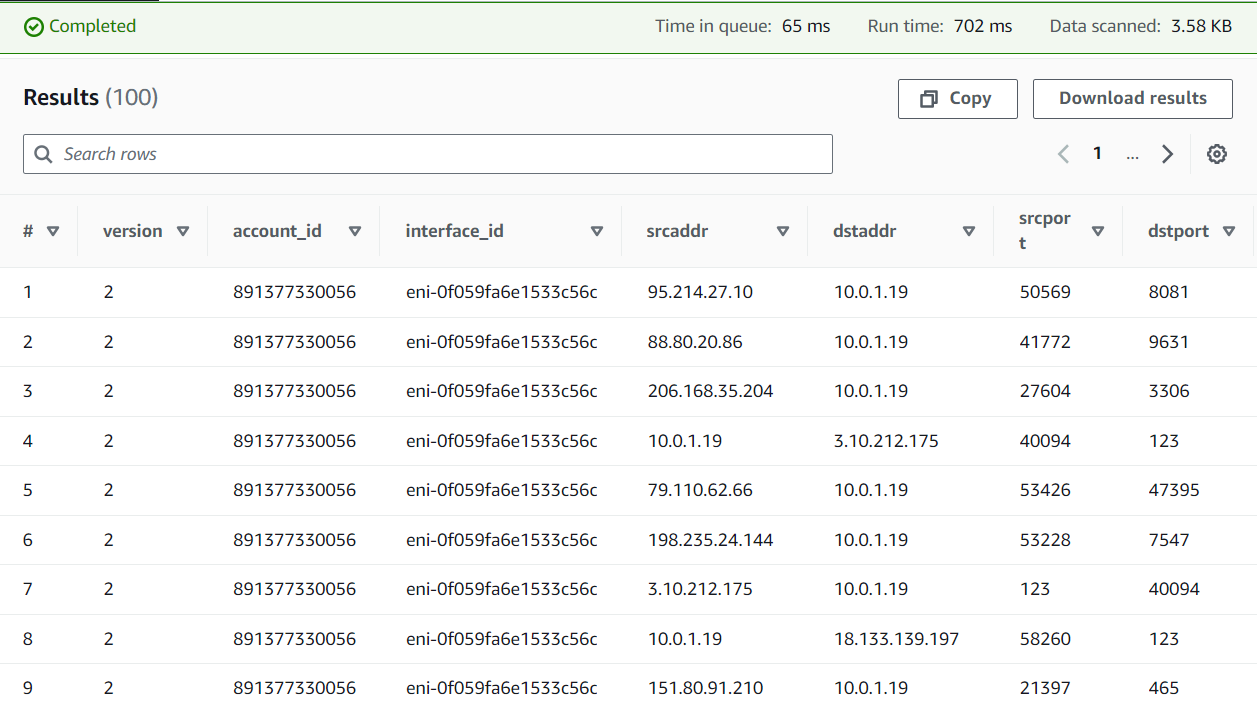
Create partitions to be able to read the data, as in the following sample query. This query creates a single partition for a specified date. **Replace the placeholders for date and location as needed.**



The following example query lists a maximum of 100 flow logs for the date specified.

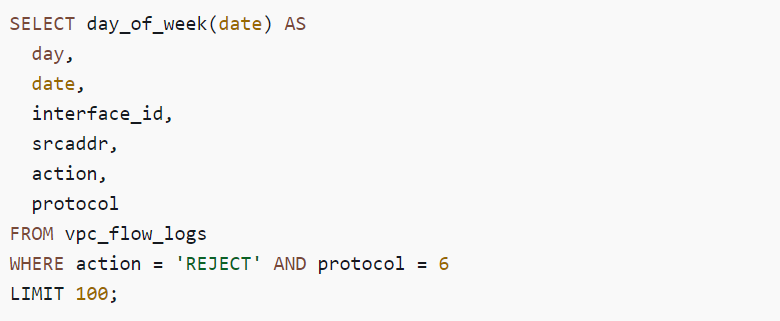


It shows as results below;



Also, we can try another query:

The following query lists all of the rejected TCP connections and uses the newly created date partition column, date, to extract from it the day of the week for which these events occurred.

****