SIEM IMPLEMENTATION AND LOG ANALYSIS WITH SPLUNK

Project Overview

I assumed the role of a Tier 1 SOC Analyst working in a hybrid cloud enterprise environment. I used Splunk as the primary SIEM tool to perform threat detection and incident investigation based on real-world logs (AWS, O365, firewalls, endpoint security, DNS, Sysmon and others.

This project simulates a real SOC environment, providing hands-on experience analyzing multi-source logs and generating reports around observed malicious activities.

Objectives

- Understand and navigate Splunk's interface (Search, Dashboards, Reports).
- Analyze multi-source logs (cloud, network, endpoint).
- Investigate suspicious behavior.
- Correlate events across multiple data sources for complete attack storylines.
- Apply detection logic and write simple correlation searches

Environment Setup

- Install Splunk Enterprise
- Pre-configured Splunk instance loaded with the dataset.
- Unzip the downloaded file into /opt/splunk/etc/apps
- Restart Splunk
- The data will be available by searching: **index=botsv3 earliest=0**.

Deliverables

1.List out the IAM users that accessed an AWS service (successfully or unsuccessfully) in Frothly's AWS environment.

ANSWER

With this command

index=botsv3 sourcetype="aws:cloudtrail" userIdentity.username=*

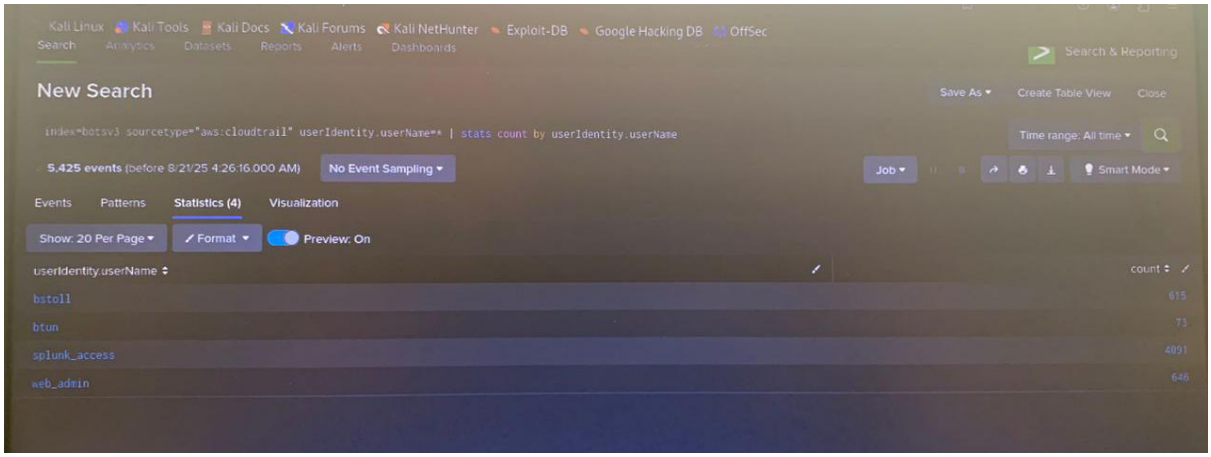
| stats count by userIdentity.userName

Index=botsv3 - Tells Splunk where logs are located.

Sourcetype="aws:cloudtrail" - This allows results only related to AWS CloudTrail logs which contains records of API calls made in AWS.

userIdentity.userName – Helps us focus on only IAM users who performed some action.

stats count by userIdentity.userName – This helps to show in a table each IAM username and the number of times they appear in the logs.



The list of IAM users are **bstoll (615), btun (73), splunk_access (4091) and web_admin (646).**

2. What is the processor number used on the web servers?

ANSWER

With this command,

Index=botsv3 sourcetype=”hardware”

| rex field=_raw “CPU_TYPE\s+(?<processor>[^,\r\n]+)” | table host processor

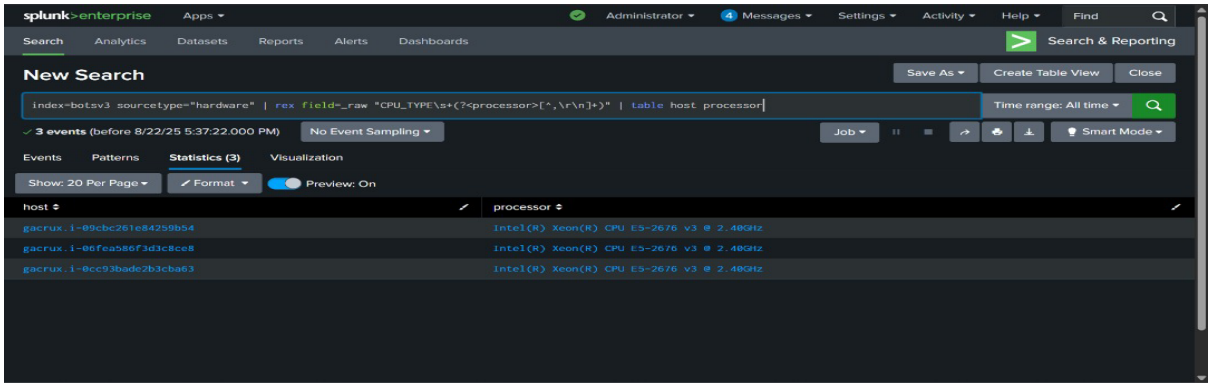
sourcetype=”hardware” – Searches all events where sourcetype is hardware.

rex((Regular Expression extraction – Allows extraction of useful fields from raw log text.

field=_raw – Extraction from the raw event text itself.

“CPU_TYPE\s+(?<processor>[^,\r\n]+)” – Searches for keyword CPU_TYPE in the logs then captures the actual CPU model right after it

| table host processor – Brings all required parameters together in a table.



The processor number on the web servers is **Intel Xeon(R) CPU E5-2676 v3 @ 2.40GHz**

3. Bud accidentally makes an S3 bucket publicly accessible. What is the event ID of the API call that enabled public access?

ANSWER

With this command and its functions

Index=botsv3 sourcetype=”aws:cloudtrail” eventName IN (“PutBucketAcl”, ‘PutBucketPolicy”, “PutBucketPublicAccessBlock”)

| table _time userIdentity.userName eventName requestParameters eventID errorMessage

Index=botsv3 - Tells Splunk where logs are located

Sourcetype=”aws:cloudtrail” - This allows results only related to AWS CloudTrail logs which contains records of API calls made in AWS.

eventName IN (“PutBucketAcl”, ‘PutBucketPolicy”, “PutBucketPublicAccessBlock”) - This filters to only events that could expose an S3 bucket to the public

PutBucketAcl - API call that changes the Access Control List (ACL) of a bucket. N.B: If “AllUsers” or “AuthenticatedUsers” is added the bucket becomes public.

PutBucketPolicy - API call that modifies the bucket policy (can allow public access).

PutBucketPublicAccessBlock – API call that enables/disables the public access block settings

Table…- This part displays fields in a table

_time – When the API call happened

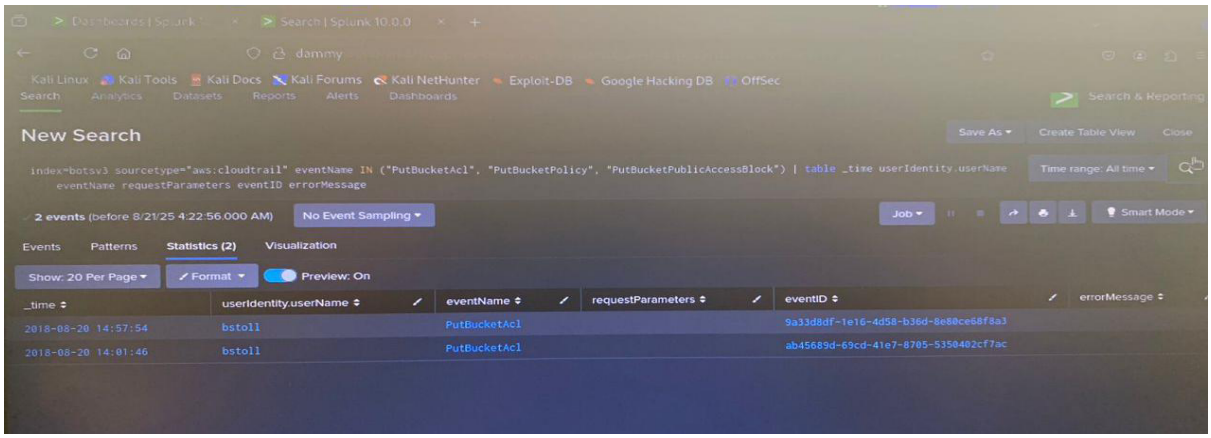
eventName – Which API call whether ACL, Policy or PublicAccessBlock.

requestParameters.bucketName – The bucket name that was modified.

eventID – The unique ID of the event.

requestParameters – The details about the API request.

errorMessage – If the action failed for example if permission was denied.

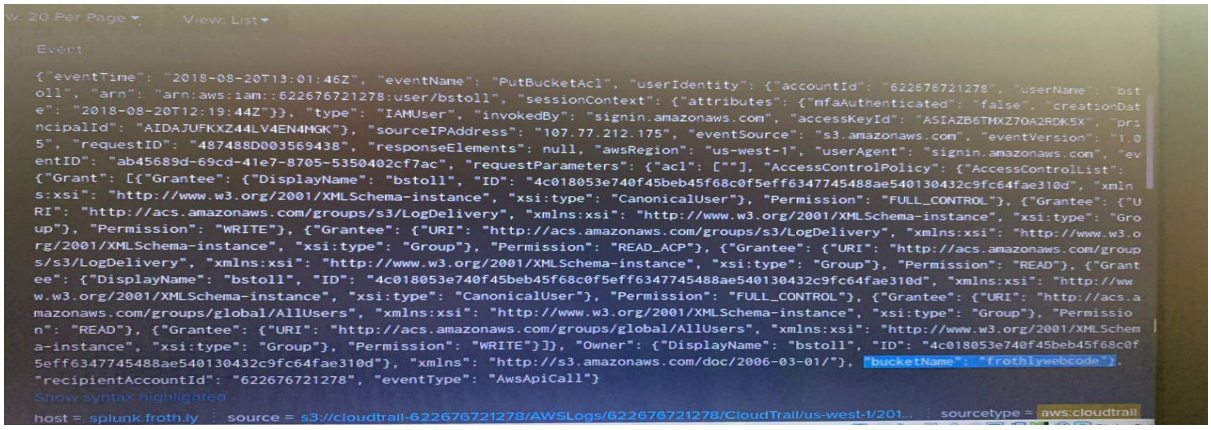


There are two events but only one eventID correlates with our files and the time frame. The **eventID** is **ab45689d-69cd-41e7-8705-5350402cf7ac**

4. What is the name of the S3 bucket that was made publicly accessible?

ANSWER

Since the bucket name was not displayed under fields in the table, select on the requestParameters, open view events then show details of events in raw text.



Locate the field “bucketName”. The name of the S3 bucket that was made publicly accessible is **“frothlywebcode”.**

5. What is the name of the text file that was successfully uploaded into the S3 bucket while it was publicly accessible?

ANSWER

This command is used

index=botsv3 frothlywebcode *.txt

| rex field=_raw “GET\s+/(?<filename>[^/\s]+\.txt)*

| where isnotnull(filename)

| table _time, filename

| dedup filename

| sort _time

Index=botsv3 frothlywebcode *.txt – Helps filter logs only with frothlywebcode and further narrows results to logs mentioning .txt

| rex field=_raw “GET\s+/(?<filename>[^/\s]+\.txt)* -

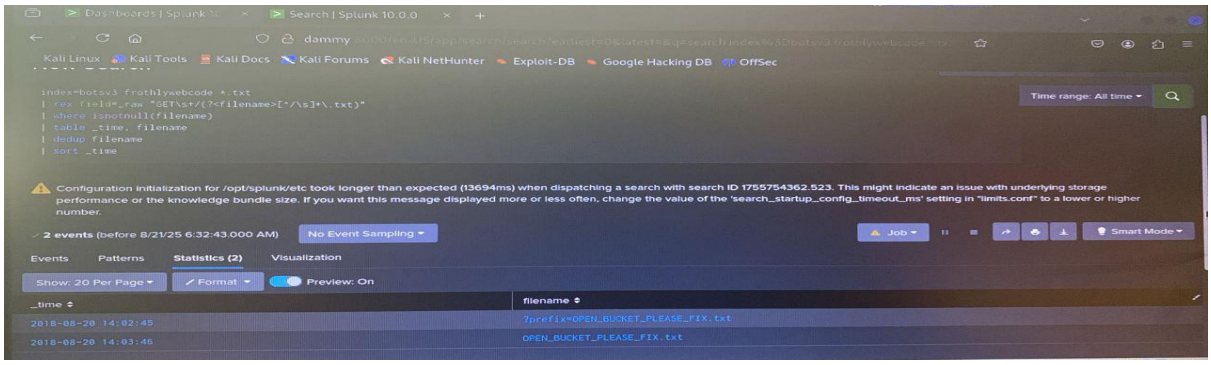
- Uses regex (rex) on the raw log text.
- Looks for HTTP GET requests (GET/filename.txt)
- Captures the .txt file name into a new field called filename.

| where isnotnull(filename) – Ensures only events where “filename” was successfully extracted are kept.

| table _time, filename – Displays only the timestamp of the event (_time) and the extracted filename.

| dedup filename – This keeps only the first occurrence of the filename in case it was requested multiple times

| sort _time – Sorts the table by _time and helps know the order in which .txt files were accessed.



Name of the text file that was successfully uploaded into the S3 bucket while it was publicly accessible is

OPEN_BUCKET_PLEASE _FIX.txt

6. What is the size (in mb) of the .tar.gz file that was successfully uploaded into the S3 bucket while it was publicly accessible? Answer guidance: Round to two decimal places

ANSWER

With this command, we will be able to determine the .tar.gz file.

index=botsv3 frothlywebcode *.tar.gz

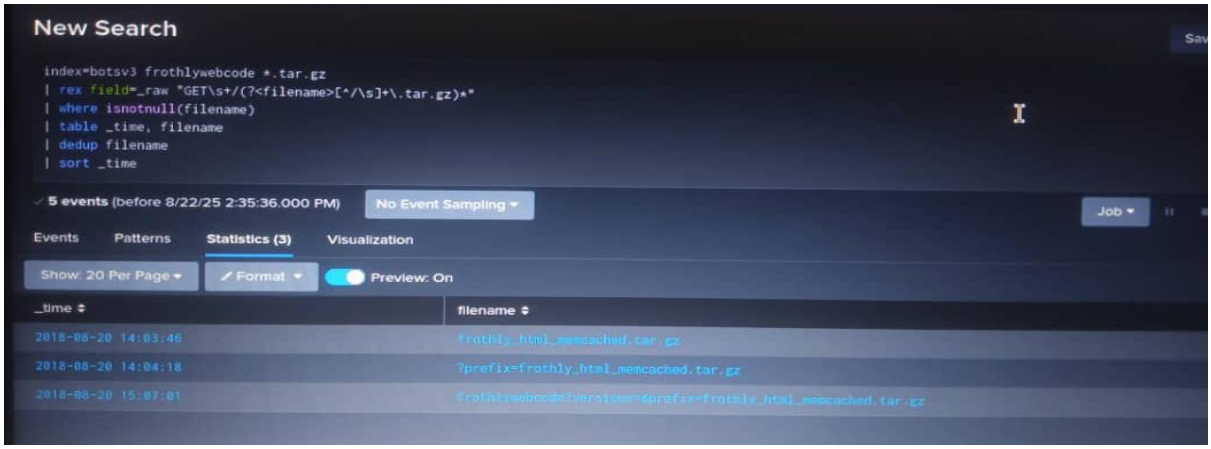
| rex field=_raw “GET\s+/(?<filename>[^/\s]+\.tar.gz)*

| where isnotnull(filename)

| table _time, filename

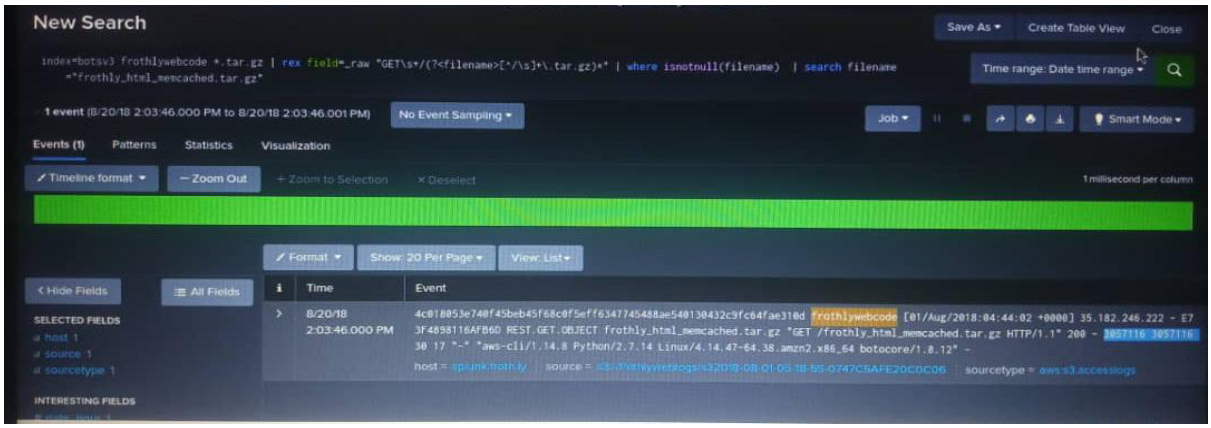
| dedup filename

| sort _time



Using the date and time frame of that of the text file (14:03:46), we can compare it to determine which tar.gz file was uploaded when the S3 bucket was made publicly accessible.

The tar.gz file is *frothly_html_memcached.tar.gz*. Click on the file to view events.



The size of the tar.gz file in bytes is 3057116. **In mb, 3057116 = 3057116/1024/1024= 2.91mb**

OR

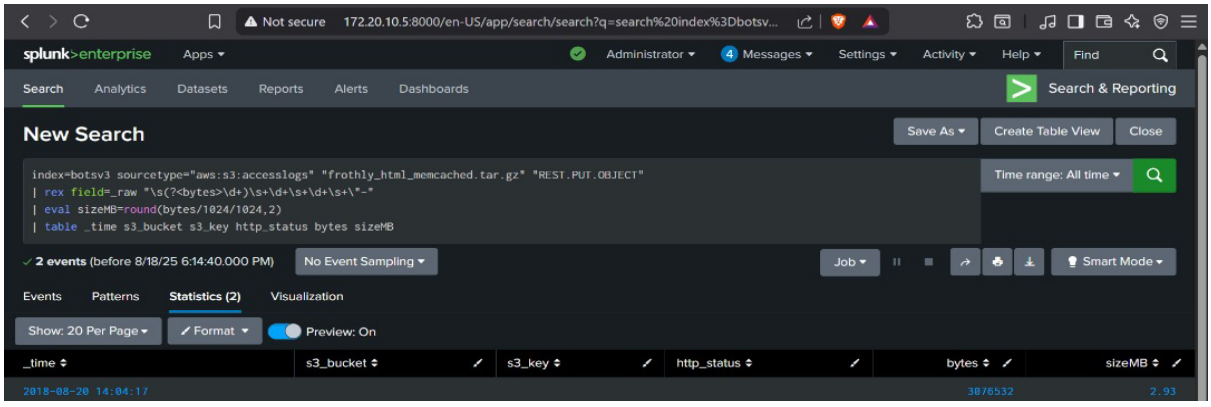
With this direct command

Index=botsv3 sourcetype=”aws:s3:accesslogs” “frothly_html_memcached.tar.gz” “REST.PUT.OBJECT”

| rex field=_raw “\s(?<bytes>\d+)\s+\d+\s+\d+\s+\”_”

| eval sizeMB=round(bytes/1024/1024,2)

| table _time s3_bucket s3_key http status bytes sizeMB



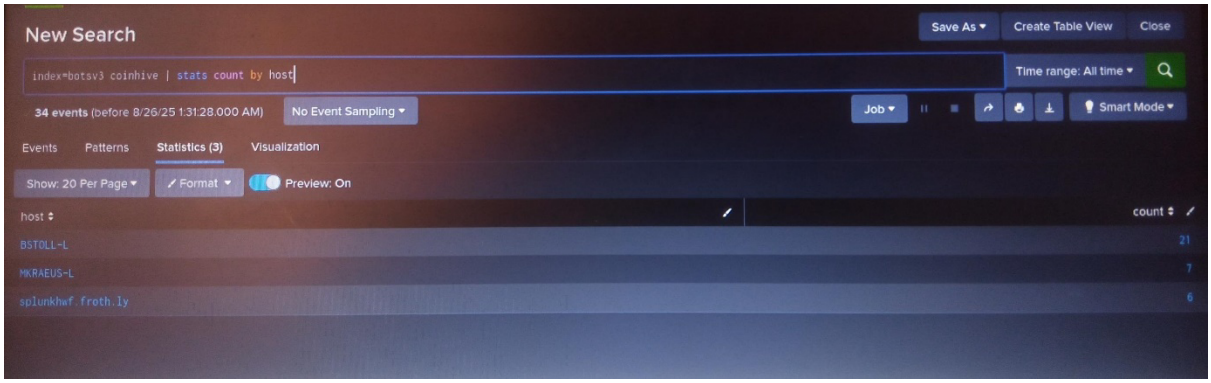
The size of the .tar.gz file=

2.93mb

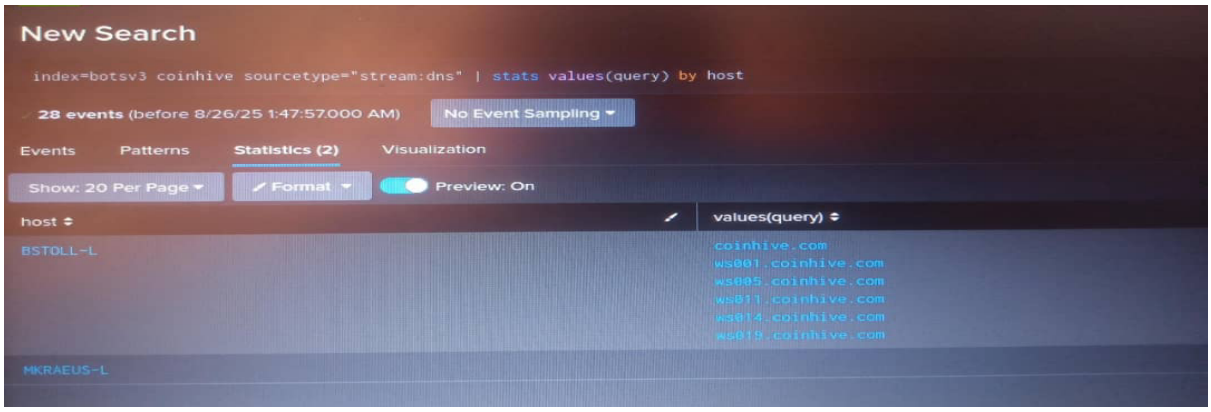
7. What is the short hostname of the only Frothly endpoint to actually mine Monero cryptocurrency?

ANSWER

First, we need to check for known mining indicators, most common is COINHIVE (a monero web miner) and which endpoints communicated with Coinhive servers.



Not all endpoints that see Coinhive traffic are actually mining. So we check which hosts made DNS queries of HTTP requests.



Only one host has DNS queries to coinhive domains.

The short hostname of the only Frothly endpoint to actually mine Monero cryptocurrency =

BSTOLL-L

8. What is the FQDN of the endpoint that is running a different Windows operating system edition than the others?

ANSWER

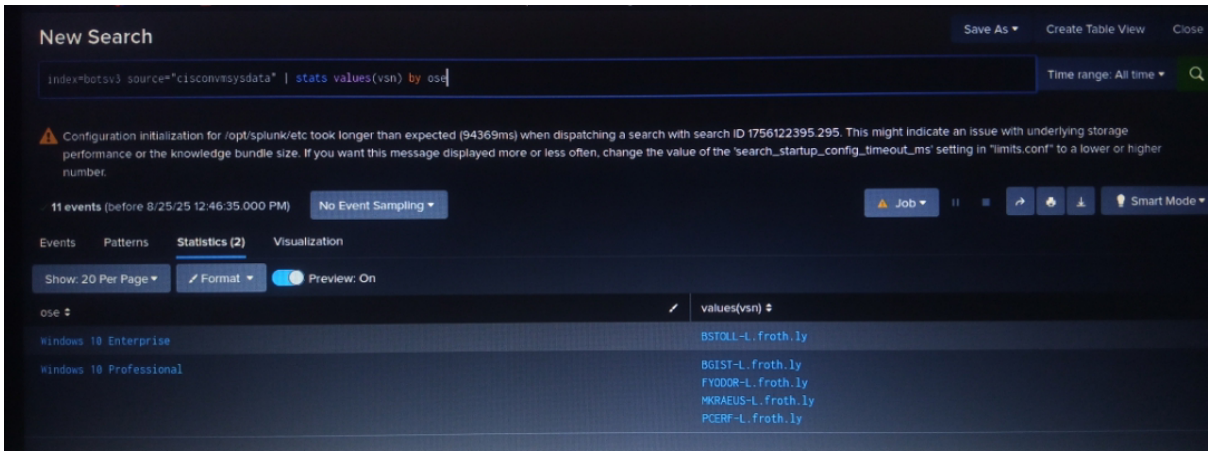
Command used

index=bots3v source=”ciscovmsysmata”

| stats values(vsn) by ose

Source=”ciscosysmata” – Maps OS edition to FQDN

| stats values(vsn) by ose – Collects all values of the field FQDN and sorts group by operating system.



The FDQN of the endpoint =

BSTOLL-L.froth.ly (has a different windows operating system).

CONCLUSION

From this project, I have been able to effectively monitor, analyze and detect suspicious activities across Frothly's environment with the use of Splunk. Through real time log collection and correlation, Splunk enabled the identification of misconfigured S3 buckets, suspicious uploads, unauthorized access attempts and cryptocurrency mining activity on specific endpoints.

This exercise demonstrates the importance of centralized log management and proactive monitoring. Splunk, a Security Information and Event Management tool serves as a vital tool for strengthening an organization security and network by improving incident response times and ensuring compliance with security best practices.