

Prepared by: Pelin Gülmez

Date: 16 June 2025

SIEM Project – Week 2 Report

Table of Contents

1. Overview
2. Deliverable 1: Log Parsing Rules
3. Deliverable 2: Dashboarding
4. Dashboard Summary Table
5. Tools and Environment
6. Conclusion
7. Attachments

1. Overview

This report summarizes the work done during Week 2 of the SIEM project. The main focus areas were:

- Log parsing techniques for SNORT and UFW logs using Elastic Stack 8.13.4.
- Creating visual dashboards for security monitoring.
- Parsing SNORT alerts and UFW firewall activity logs.

All tasks were performed on an Elastic Stack 8.13.4 instance with Filebeat configurations for IDS (SNORT) and Firewall (UFW) log sources.

2. Deliverable 1: Log Parsing Rules

SNORT Log Parsing

Log Source: /var/log/snort/alert

Ingest Pipeline: parse-snort-pipeline

Extracted Fields:

Field	Description
alert_msg	SNORT alert message
priority	Alert priority level
gid	Generator ID
sid	Signature ID
rev	Rule revision
src_ip	Source IP address
dest_ip	Destination IP address
proto	Protocol (TCP/UDP/ICMP etc.)
time/month/day	Timestamp components

UFW Log Parsing

Log Source: /var/log/ufr.log

Ingest Pipeline: parse-ufw-pipeline

Extracted Fields:

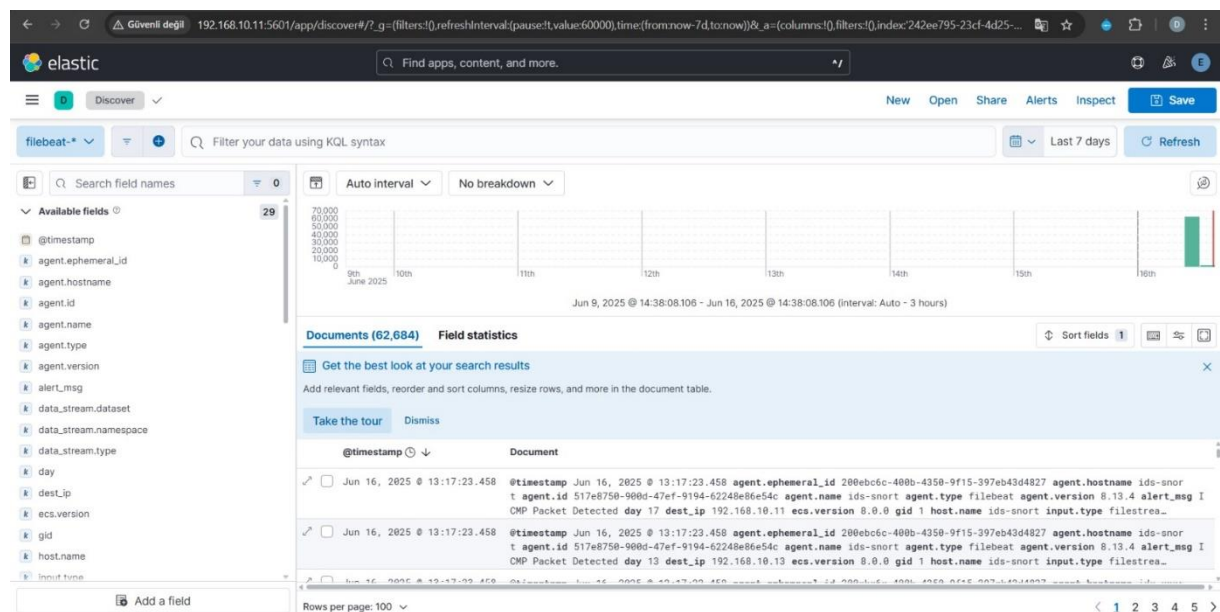
Field	Description
src_ip	Source IP address
dest_ip	Destination IP address
proto	Protocol
src_port	Source port
dest_port	Destination port
action	ALLOW/DENY
length, ttl	Packet metrics

Tools Used:

- Grok Patterns
- Ingest Pipelines (via Kibana > Stack Management)

3. Deliverable 2: Dashboarding

A unified Kibana dashboard was created to visualize SNORT alert messages and firewall activities from UFW.

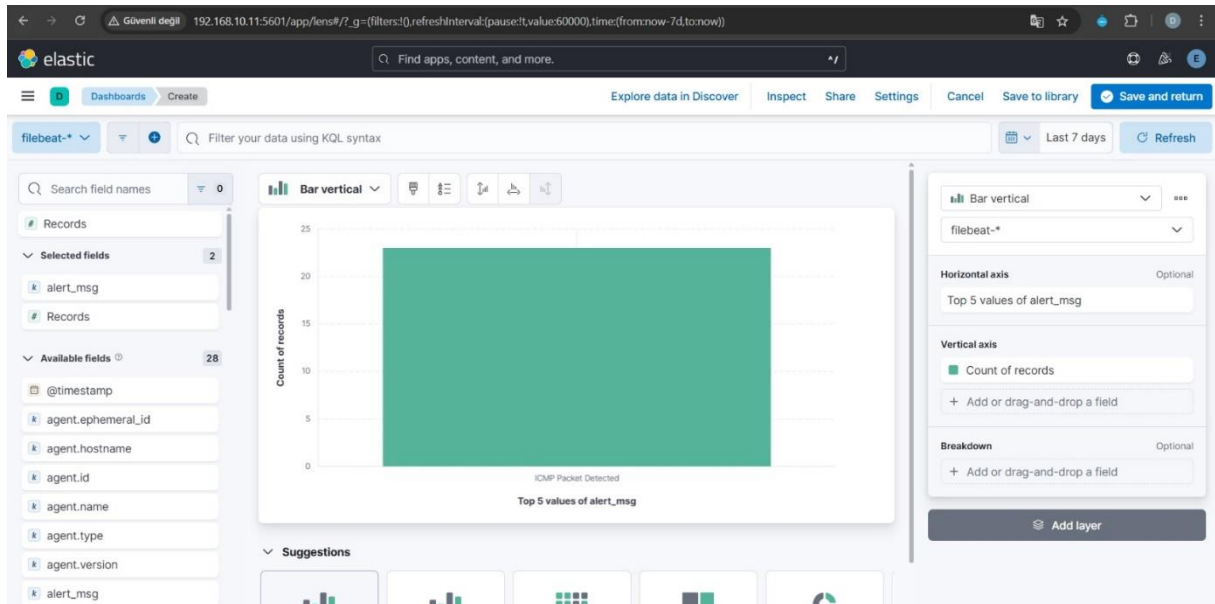


Visualizations:

1. Top SNORT Alert Messages

Type: Pie Chart

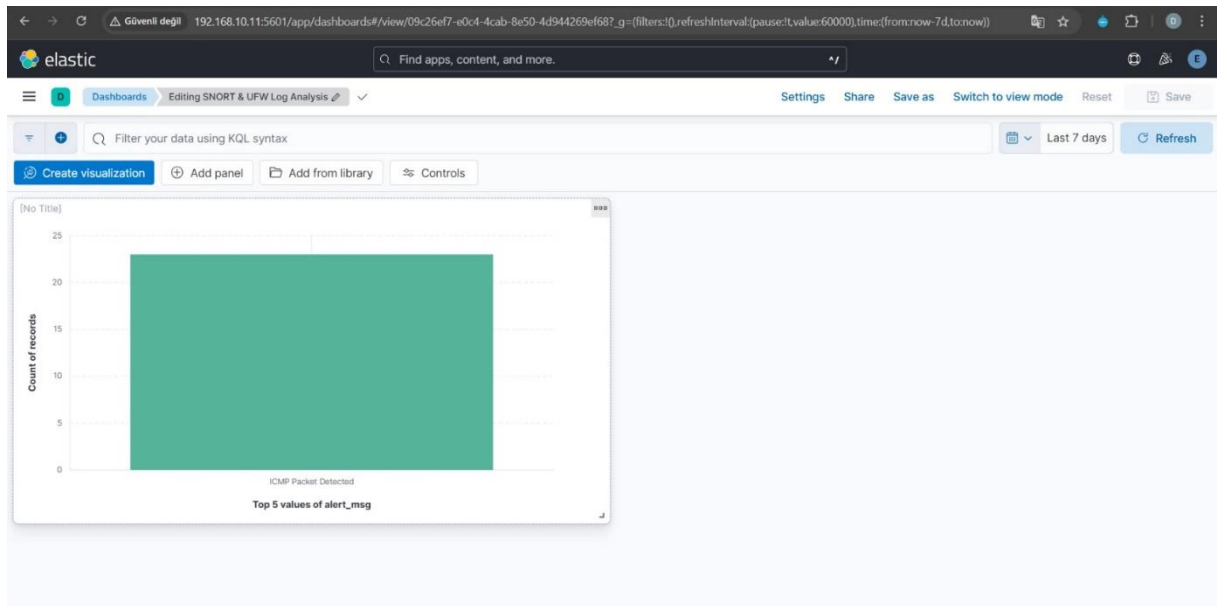
Field: alert_msg



The screenshot shows the Elastic UI interface with a "Save dashboard" dialog box open. The dialog box has the following fields and options:

- Title:** SNORT & UFW Log Analysis
- Description:** (Optional)
- Tags:** (Optional)
- Store time with dashboard:** (Checked)
- Buttons:** Cancel, Save

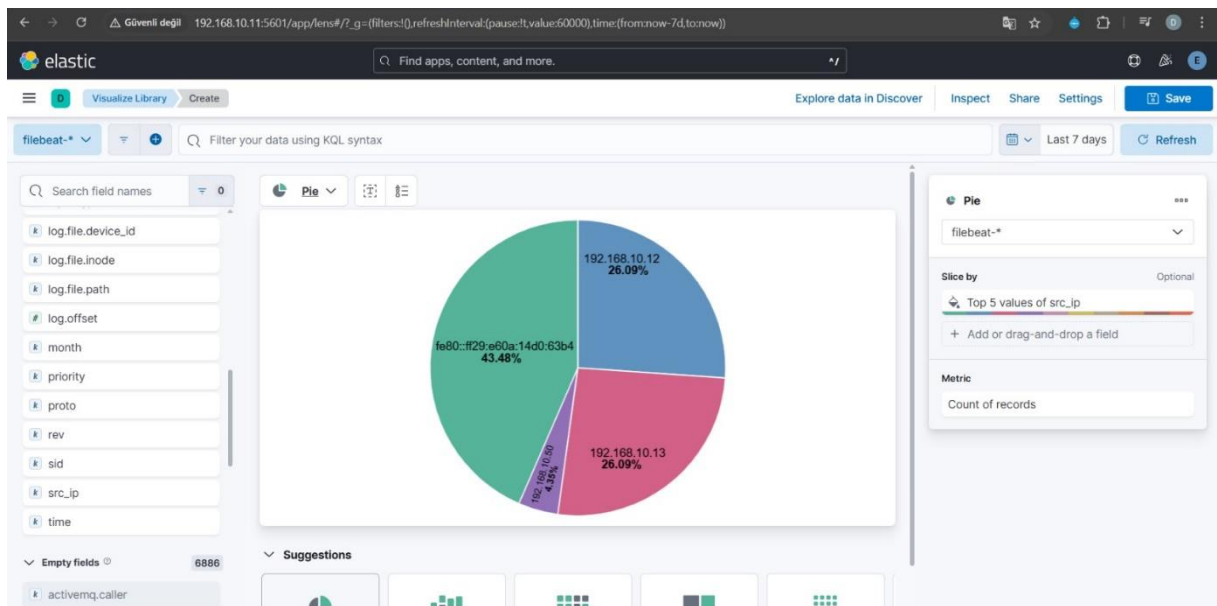
The background shows a visualization titled "Top 5 values of alert_msg" with a single bar for "ICMP Packet Detected".

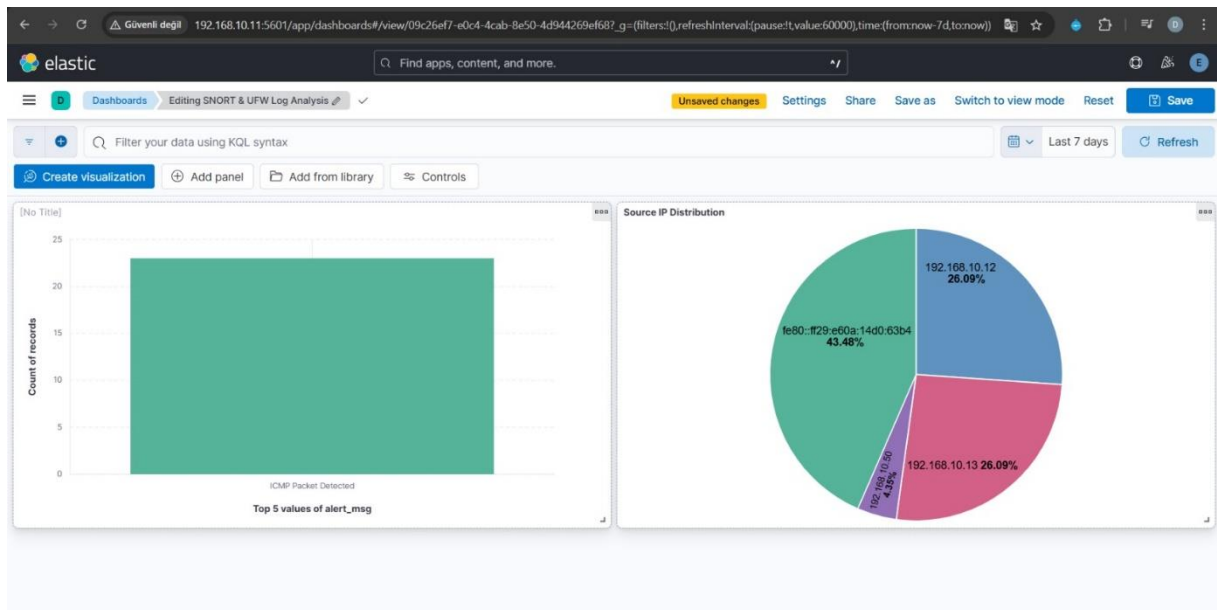
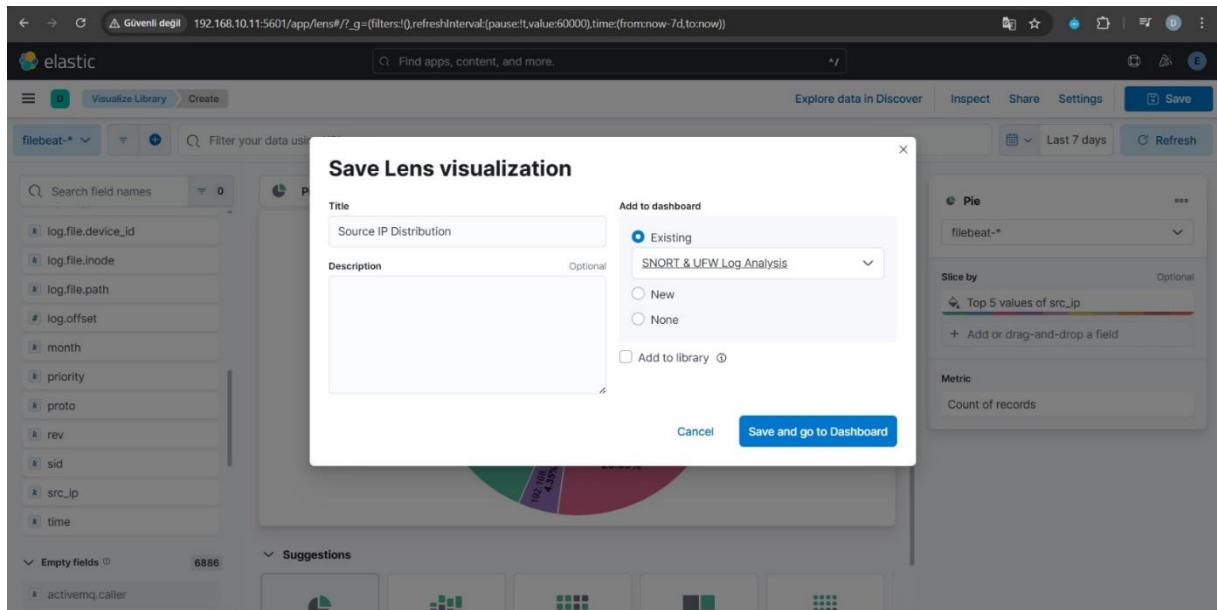


2. Source IP Distribution

Type: Pie Chart

Field: src_ip

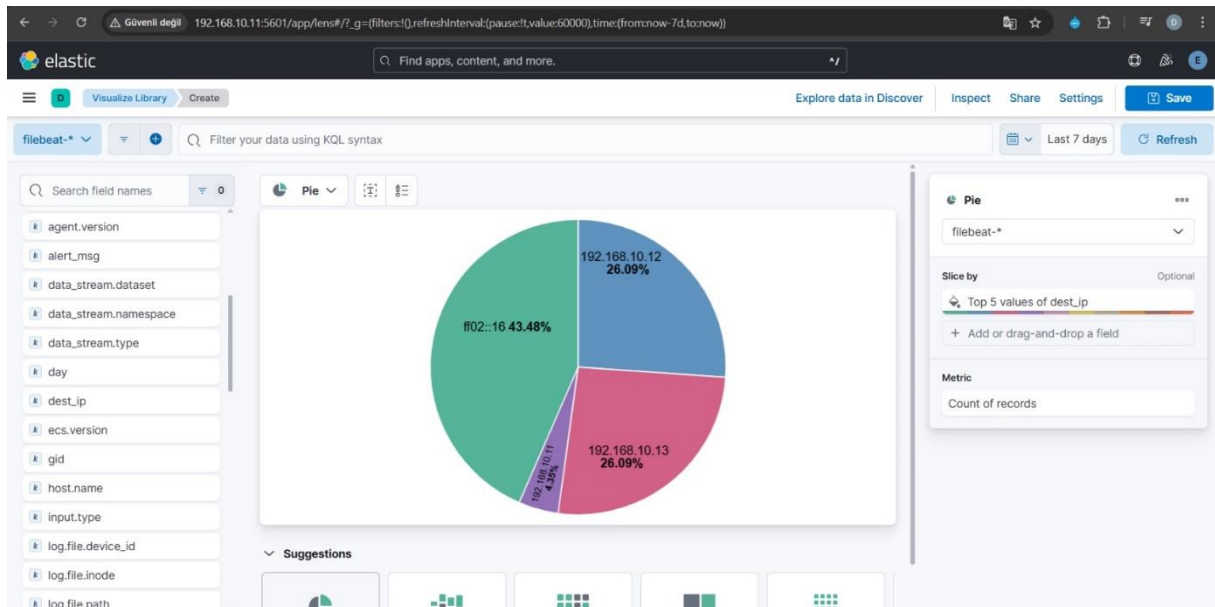




3. Destination IP Distribution

Type: Pie Chart

Field: dest_ip

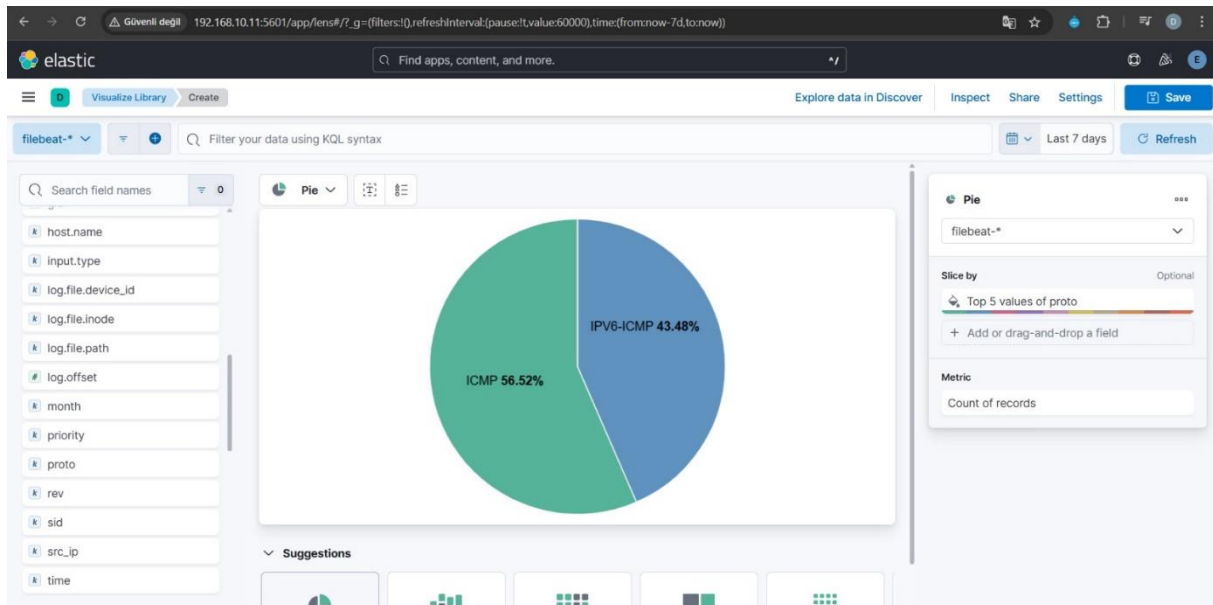


The figure shows the Elasticsearch Kibana interface with a "Save Lens visualization" dialog box open. The dialog prompts for a title, description, and options to add to a dashboard or library. The title is "Destination IP Distribution" and the description is empty. The "Add to dashboard" section has "Existing" selected, and the "Add to library" checkbox is unchecked. The "Save and go to Dashboard" button is highlighted.

4. Protocol Usage Distribution

Type: Pie Chart

Field: proto



The screenshot shows the Elasticsearch Kibana interface with the "Save Lens visualization" dialog box open. The dialog box has a "Title" field with the value "Protocol Usage Distribution" and a "Description" field. The "Add to dashboard" section has three radio buttons: "Existing" (selected), "New", and "None". The "Existing" option has a dropdown menu showing "SNORT & UFW Log Analysis". There is also a checkbox for "Add to library" which is unchecked. The "Save and go to Dashboard" button is highlighted in blue.

Save Lens visualization

Title: Protocol Usage Distribution

Description: [Optional]

Add to dashboard:

- ☒ Existing: SNORT & UFW Log Analysis
- ☐ New
- ☐ None

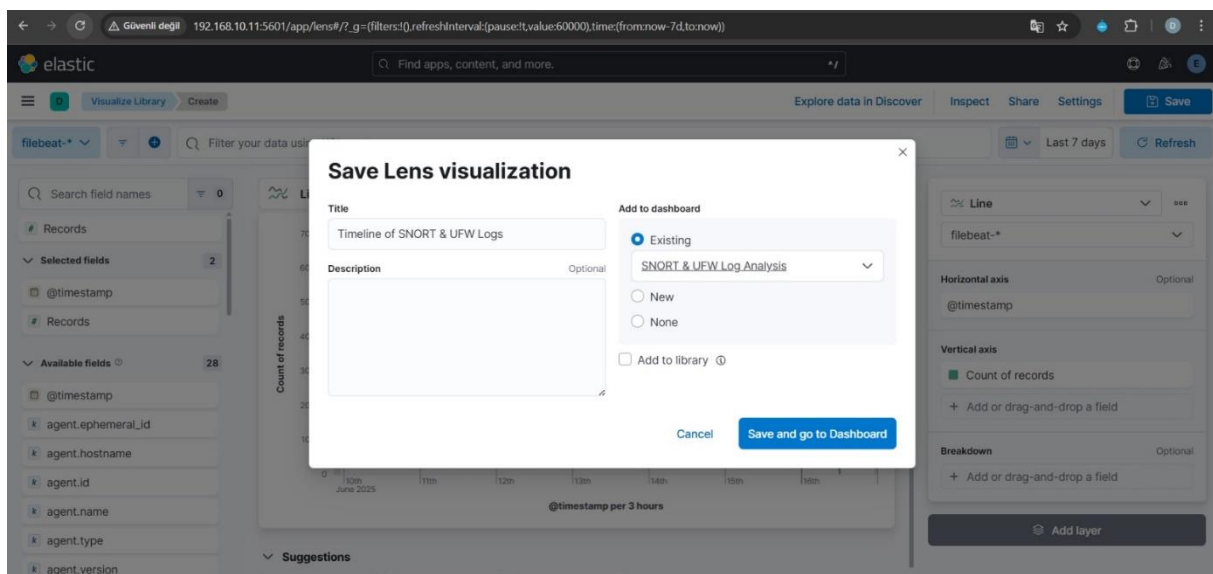
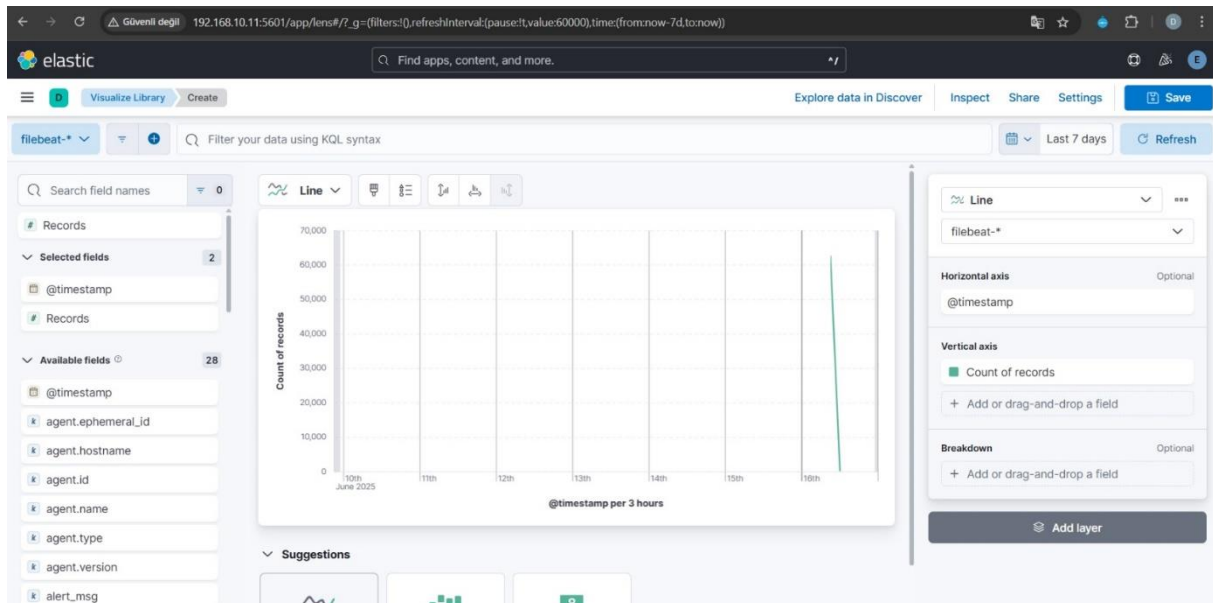
☐ Add to library

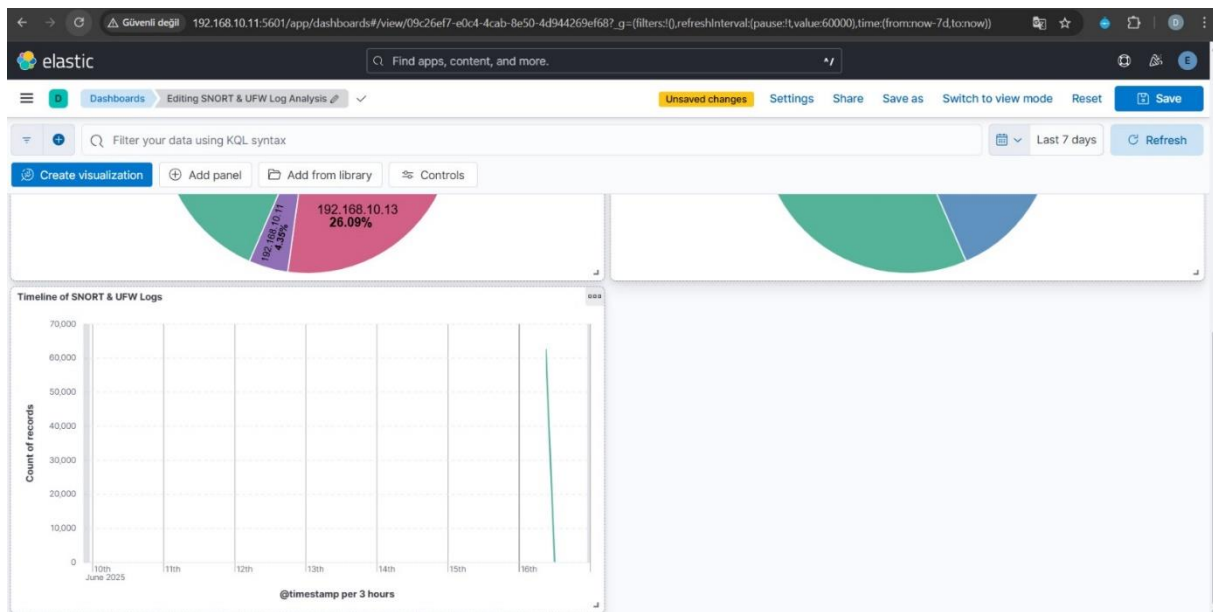
Buttons: Cancel, Save and go to Dashboard

5. Security Events Timeline

Type: Line Chart

Field: @timestamp





4. Dashboard Summary Table

Visualization	Source	Key Field	Description
Top SNORT Alerts	SNORT	alert_msg	Most triggered alert messages
Source IP Distribution	UFW	src_ip	Frequent source IPs
Destination IPs	Both	dest_ip	Targets of activity
Protocol Types	Both	proto	Protocol distribution
UFW Port Access	UFW	dest_port	Targeted ports
UFW Action	UFW	action	ALLOW vs DENY visualization
SNORT Timeline	SNORT	@timestamp	Alerts over time
UFW Timeline	UFW	@timestamp	UFW events over time

5. Tools and Environment

Component	Version / Info
Elastic Stack	8.13.4
Elasticsearch	192.168.10.11:9200
Kibana	192.168.10.11:5601
Filebeat	ids-snort, ufw-logger
Ingest Pipelines	Custom
Snort IDS	2.9.20
UFW Firewall	Default on Ubuntu
Parsing Tools	Grok Patterns
Data Visualization	Kibana Visualizations & Dashboards

6. Conclusion

In this phase of the project, log parsing rules for both SNORT and UFW log sources were successfully developed and implemented through Filebeat ingest pipelines. The parsed logs were effectively visualized via customized Kibana dashboards, providing clear insights into key metrics such as top alerts, accessed ports, protocol distributions, and attack timelines. This foundation establishes a robust and scalable SIEM monitoring framework. In the future, the system can be flexibly enhanced.

7. Attachments

Configuration Files:

Filebeat configuration from SNORT machine

```
GNU nano 8.3 /etc/filebeat/filebeat.yml
filebeat.inputs:
- type: filestream
  id: snort-input
  paths:
    - /var/log/snort/alert
  pipeline: parse-snort-pipeline

output.elasticsearch:
  hosts: ["https://192.168.10.11:9200"]
  username: "elastic"
  password: "31-Qv=pGQa-VlTBPeF7M"
  ssl.verification_mode: none
  setup.ilm.enabled: true

logging.level: debug
logging.to_files: true
logging.files:
  path: /var/log/filebeat
  name: filebeat.log
  keepfiles: 7
  permissions: 0644
```

Filebeat configuration from UFW machine

```
GNU nano 8.3 /etc/filebeat/filebeat.yml
filebeat.inputs:
- type: filestream
  id: ufw-input
  paths:
    - /var/log/ufw.log
  pipeline: parse-ufw-pipeline

output.elasticsearch:
  hosts: ["https://192.168.10.11:9200"]
  username: "elastic"
  password: "31-Qv=pGQa-VlTBPeF7M"
  ssl.verification_mode: none
  setup.ilm.enabled: true

logging.level: debug
logging.to_files: true
logging.files:
  path: /var/log/filebeat
  name: filebeat.log
  keepfiles: 7
  permissions: 0644
```

Ingest Pipelines

parse-snort-pipeline JSON export

The screenshot shows the Elastic Dev Tools interface. The top navigation bar includes the Elastic logo, a search bar, and tabs for Dev Tools and Console. The main navigation bar has links for Console, Search Profiler, Grok Debugger, and Painless Lab (BETA). The Console tab is active, showing a history of requests. The first request is a GET request to `_ingest/pipeline/parse-snort-pipeline`. The response is a JSON object with a status of 200 - OK and a response time of 327 ms. The JSON content is as follows:

```
1 {
2   "parse-snort-pipeline": {
3     "description": "SNORT parsing",
4     "processors": [
5       {
6         "grok": {
7           "field": "message",
8           "patterns": [
9             "%{MONTHNUM:month}/%{MONTHDAY:day}-%{TIME:time} \[\[\*\*\]\] \[\[%{INT:gid}
10            :%{INT:sid}:%{INT:rev}\]\] %{DATA:alert_msg} \[\[\*\*\]\] \[Priority: %{INT
11            :priority}\]\] \[\[%{DATA:proto}\]\] %{IP:src_ip} -> %{IP:dest_ip}""
12          ]
13        }
14      ]
15    }
16  }
```

parse-ufw-pipeline JSON export

The screenshot shows the Elastic Dev Tools interface. The top navigation bar includes the Elastic logo, a search bar, and tabs for Dev Tools and Console. The main navigation bar has links for Console, Search Profiler, Grok Debugger, and Painless Lab (BETA). The Console tab is active, showing a history of requests. The first request is a GET request to `_ingest/pipeline/parse-ufw-pipeline`.

200 - OK168 ms

```
1 {  
2   "parse-ufw-pipeline": {  
3     "description": "UFW parsing",  
4     "processors": [  
5       {  
6         "grok": {  
7           "field": "message",  
8           "patterns": [  
9             ""<{%INT:priority}>{%MONTH} {%MONTHDAY} {%TIME} {%HOSTNAME:hostname}  
              kernel: \[UFW {%WORD:action}\] IN=%{WORD:in_interface} OUT=%{WORD  
              :out_interface} SRC=%{IP:src_ip} DST=%{IP:dest_ip} PROTO=%{WORD:proto}  
              SPT=%{INT:src_port} DPT=%{INT:dest_port}""  
10          ]  
11        }  
12      }  
13    ]  
14  }  
15 }
```

Parsing Patterns

parse-snort-pipeline

Manage processor

Grok documentation

ConfigurationOutput

Processor

Grok

Uses [grok](#) expressions to extract matches from a field.

Field

message

Field to search for matches.

Patterns

Grok expressions used to match and extract named capture groups. Uses the first matching expression.

=

%{MONTHNUM:month}/%{MONTHDAY:day}-%{TIME:time} \[.*\] \[%{INT:gid}:%{INT:s

+

Add pattern

Pattern definitions (optional)

Cancel

Update

Manage processor

[Grok documentation](#) ×

Configuration **Output**

Processor

Grok



Uses [grok](#) expressions to extract matches from a field.

Field

message

Field to search for matches.

Patterns

Grok expressions used to match and extract named capture groups. Uses the first matching expression.

= <{%INT:priority}>{%MONTH} {%MONTHDAY} {%TIME} {%HOSTNAME:hostname} kerne

[+](#) Add pattern

Pattern definitions (optional)

Cancel

Update