The Mainframe

IT 360 Final Project Report

12/2/25

IT 360 Final Project Report

Suricata/Splunk

Table of Contents

1. Introduction

In our project we used Suricata, Splunk and a created script. We used these tools to configure rules to create alerts in a log file, take that log file and ingest it into Splunk to create a skeleton report and a table view, and lastly taking that information and inputting the file into the created script to make a more detailed report including severity, analysis and recommendations on how to act.

The purpose of our project was to find tools that could work together seamlessly and make it simpler to create a system where analysts could deal with incidents faster and more effectively. In digital forensics Suricata helps with alerting from specific anomalies, Splunk also helps organize the log file from Suricata very well. The created script is very helpful for digital forensics as it created a very detailed report that is very easy to read and understand.

2. Technical Implementation

2.1 Suricata Installation and Configuration

Suricata was installed on a Linux system:

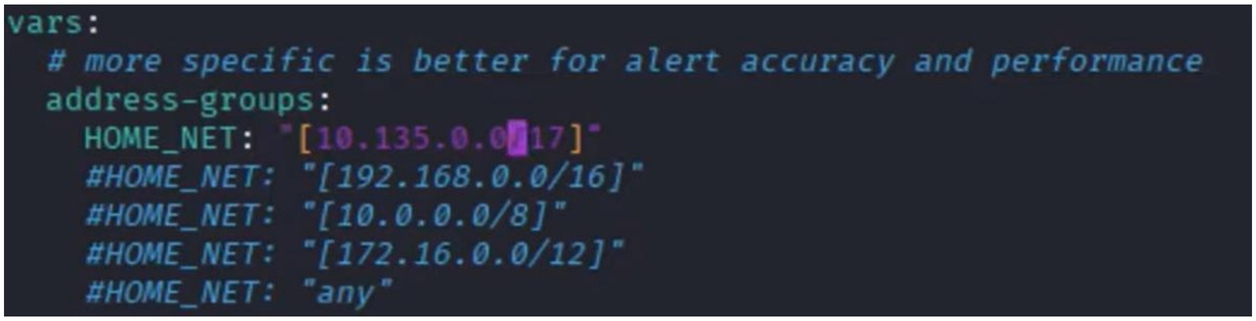
- sudo apt update
- sudo apt install suricata -y

Key configuration steps

- Setting HOME_NET to the correct subnet in /etc/suricata/suricata.yaml
- Updating all interface names under af-packet and pcap to match the actual NIC (e.g., eth0)
- Enabling community-id for flow correlation
- Ensuring eve-log was enabled so logs are written to /var/log/suricata/eve.json
- Running sudo suricata-update to download Emerging Threats rules
- Verifying configuration and rule loading using:
    - sudo suricata -T -c /etc/suricata/suricata.yaml -v

Suricata was then tested by starting the service, monitoring logs with tail -f, and triggering alerts using controlled traffic such as:

- curl http://testmynids.org/uid/index.html



```
vars:
  # more specific is better for alert accuracy and performance
  address-groups:
    HOME_NET: "[10.135.0.0/17]"
    #HOME_NET: "[192.168.0.0/16]"
    #HOME_NET: "[10.0.0.0/8]"
    #HOME_NET: "[172.16.0.0/12]"
    #HOME_NET: "any"
```

2.2 Custom Suricata Rules

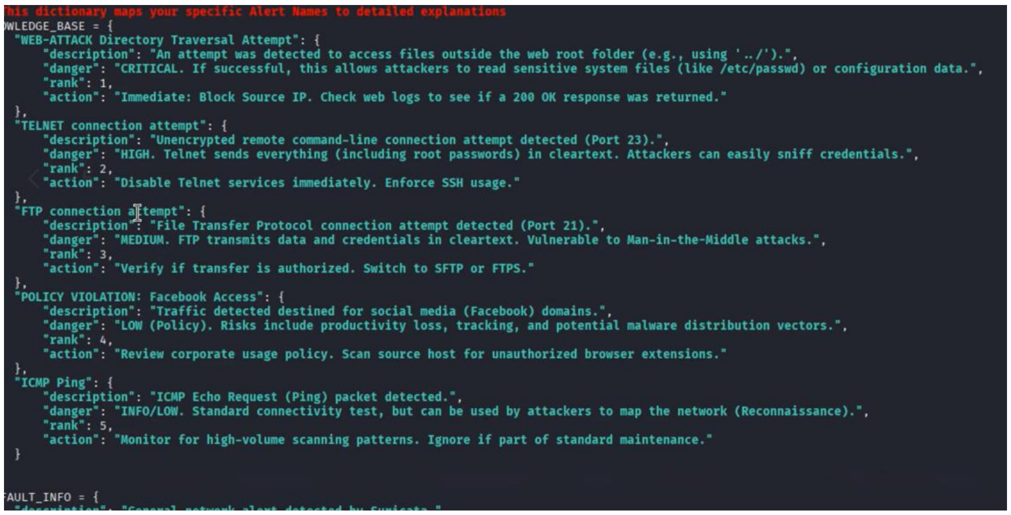
A custom rule file was created at:

- /etc/suricata/rules/local.rules

The following custom rules were developed and tested:

- ICMP Ping: alert icmp any any -> $HOME_NET any (msg:”ICMP Ping”; sid:1; rev:1;)
- Telnet attempt: alert tcp any any -> any 23 (msg:"TELNET connection attempt"; sid:1000001; rev:1;)
- FTP attempt: alert tcp any any -> any 21 (msg:"FTP connection attempt"; sid:1000004; rev:1;)
- DNS Facebook request: alert dns any any -> any any (msg:"POLICY VIOLATION: Facebook Access"; dns.query; content:"facebook.com"; sid:1000006; rev:1;)
- Directory Traversal: alert http any any -> any 80 (msg:"WEB-ATTACK Directory Traversal Attempt"; content:"../"; http_raw_uri; classtype:web-application-attack; zsid:1000005; rev:2;)

Testing involved using commands such as ping, telnet, ftp, nslookup, and curl --path-as-is to verify rule behavior.



```
his dictionary maps your specific Alert Names to detailed explanations
WLEDGE_BASE = {
 "WEB-ATTACK Directory Traversal Attempt": {
     "description": "An attempt was detected to access files outside the web root folder (e.g., using '../').",
     "danger": "CRITICAL. If successful, this allows attackers to read sensitive system files (like /etc/passwd) or configuration data.",
     "rank": 1,
     "action": "Immediate: Block Source IP. Check web logs to see if a 200 OK response was returned."
 },
 "TELNET connection attempt": {
     "description": "Unencrypted remote command-line connection attempt detected (Port 23).",
     "danger": "HIGH. Telnet sends everything (including root passwords) in cleartext. Attackers can easily sniff credentials.",
     "rank": 2,
     "action": "Disable Telnet services immediately. Enforce SSH usage."
 },
 "FTP connection attempt": {
     "description": "File Transfer Protocol connection attempt detected (Port 21).",
     "danger": "MEDIUM. FTP transmits data and credentials in cleartext. Vulnerable to Man-in-the-Middle attacks.",
     "rank": 3,
     "action": "Verify if transfer is authorized. Switch to SFTP or FTPS."
 },
 "POLICY VIOLATION: Facebook Access": {
     "description": "Traffic detected destined for social media (Facebook) domains.",
     "danger": "LOW (Policy). Risks include productivity loss, tracking, and potential malware distribution vectors.",
     "rank": 4,
     "action": "Review corporate usage policy. Scan source host for unauthorized browser extensions."
 },
 "ICMP Ping": {
     "description": "ICMP Echo Request (Ping) packet detected.",
     "danger": "INFO/LOW. Standard connectivity test, but can be used by attackers to map the network (Reconnaissance).",
     "rank": 5,
     "action": "Monitor for high-volume scanning patterns. Ignore if part of standard maintenance."
 }


AULT_INFO = {
```

2.3 Splunk Enterprise installation and Configuration

Splunk Enterprise was downloaded from the Splunk website (Linux .deb package) and installed using:

- sudo dpkg -i splunk-10.0.2-*.deb
- sudo /opt/splunk/bin/splunk start --accept-license

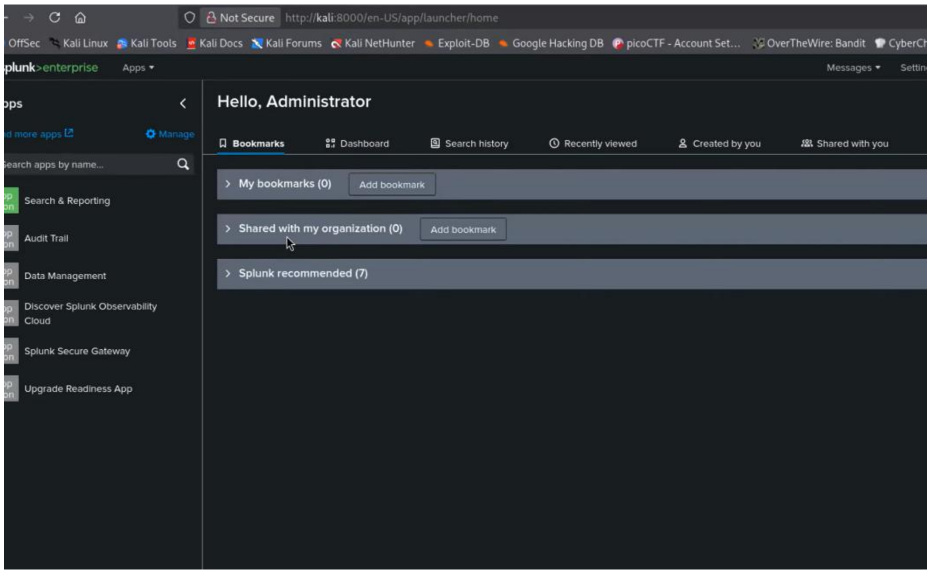
Create admin account

- The terminal will ask you to create a username and password for the admin account
- Follow the prompts to enter username and password

After that, open the link at the end of the output for the splunk web interface

- http://kali:8000
- http://localhost:8000

Enter credentials for the admin account to get into the home page


Not Secure http://kali:8000/en-US/app/launcher/home
OffSec Kali Linux Kali Tools Kali Docs Kali Forums Kali NetHunter Exploit-DB Google Hacking DB picoCTF - Account Set... OverTheWire: Bandit CyberCh
plunk>enterprise Apps
Messages Settin
pps
d more apps
Manage
earch apps by name...
Search & Reporting
Audit Trail
Data Management
Discover Splunk Observability Cloud
Splunk Secure Gateway
Upgrade Readiness App
Hello, Administrator
Bookmarks Dashboard Search history Recently viewed Created by you Shared with you
My bookmarks (0) Add bookmark
Shared with my organization (0) Add bookmark
Splunk recommended (7)

2.4 Ingesting Suricata logs into Splunk

- On the Splunk web browser, go to the top bar and click “Settings”, then “Data inputs”
- Add a new input by clicking on “files and directories” under “local inputs”
- Click on “new local file and directory”

- In the “file or directory” field, hit “browse” to enter the path of the suricata log file
    - /var/log/suricata/eve.json
    - Make sure “continuously monitor” option is selected
    - Cleck next
- In “set source type”, click on the drop down menu “source type” and type “json”
    - Select _json which will work for our eve.json files
    - Click next
- In “Input settings” for the “index” drop down menu, select main
    - In the “host method”, select “constant value”
    - In the “Host Field”, enter a name like kali-vm because this will tag every event and you know exactly which machine sent it
- Click “review”, “submit”, “Start Searching”

2.5 SPL report query

To visualize all Suricata alerts, the following Splunk Search Processing Language (SPL) query was used:

```
index="main" sourcetype="_json" event_type="alert"
| table _time, src_ip, dest_ip, alert.signature, alert.category, alert.severity
| rename _time as "Time", src_ip as "Source IP", dest_ip as "Destination IP",
alert.signature as "Alert Name", alert.category as "Category", alert.severity as
"Severity"
| sort - "Time"
```

This query was saved as a permanent Splunk report for one-click access.

2.6 Report generation script

A Python script (using **pandas** and **FPDF**) was created to turn exported Splunk CSV data into a formatted PDF report. The script:

- Accepts .csv as input
- Sorts and analyzes alert severity
- Generates a final PDF containing:

    - Attack rankings
    - Alert metadata
    - Remediation recommendations

Dependency installed:

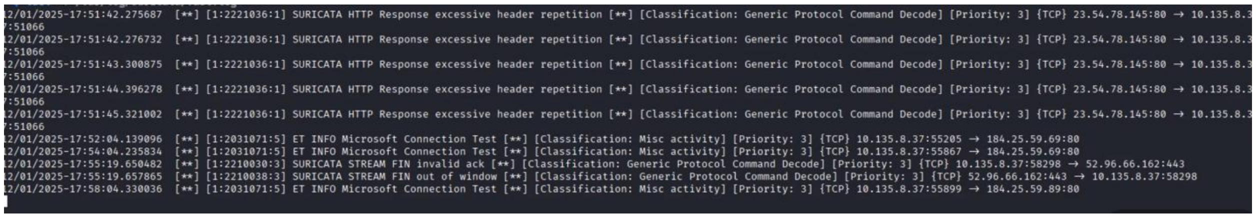
- pip install pandas fpdf --break-system-packages

3. Results

3.1 Suricata Alerts

The system successfully generated alerts for:

- Directory traversal attempts
- ICMP pings
- Telnet/FTP connection attempts
- DNS lookups for Facebook
- Emerging Threats rules triggered by testmynids.org



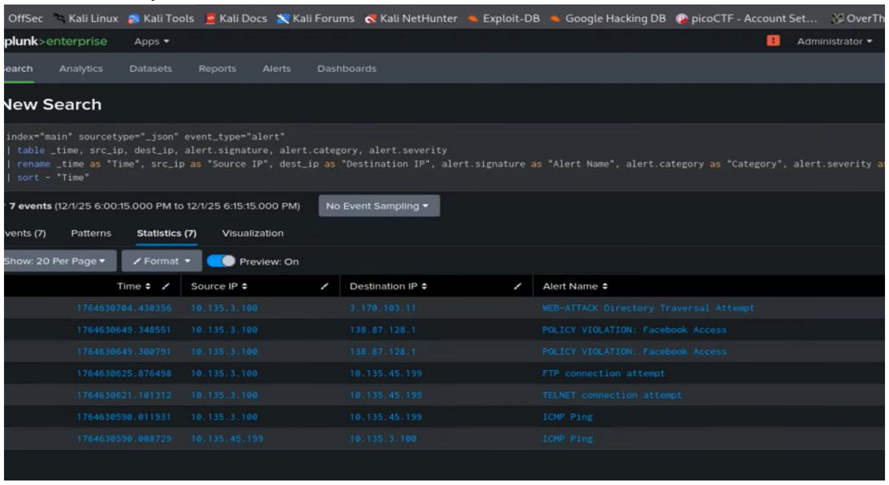
```
2/01/2025-17:51:42.275687  [**] [1:2221036:1] SURICATA HTTP Response excessive header repetition [**] [Classification: Generic Protocol Command Decode] [Priority: 3] {TCP} 23.54.78.145:80 → 10.135.8.3
:51066
2/01/2025-17:51:42.276732  [**] [1:2221036:1] SURICATA HTTP Response excessive header repetition [**] [Classification: Generic Protocol Command Decode] [Priority: 3] {TCP} 23.54.78.145:80 → 10.135.8.3
:51066
2/01/2025-17:51:43.300875  [**] [1:2221036:1] SURICATA HTTP Response excessive header repetition [**] [Classification: Generic Protocol Command Decode] [Priority: 3] {TCP} 23.54.78.145:80 → 10.135.8.3
:51066
2/01/2025-17:51:44.396278  [**] [1:2221036:1] SURICATA HTTP Response excessive header repetition [**] [Classification: Generic Protocol Command Decode] [Priority: 3] {TCP} 23.54.78.145:80 → 10.135.8.3
:51066
2/01/2025-17:51:45.321002  [**] [1:2221036:1] SURICATA HTTP Response excessive header repetition [**] [Classification: Generic Protocol Command Decode] [Priority: 3] {TCP} 23.54.78.145:80 → 10.135.8.3
:51066
2/01/2025-17:52:04.139096  [**] [1:2031071:5] ET INFO Microsoft Connection Test [**] [Classification: Misc activity] [Priority: 3] {TCP} 10.135.8.37:55205 → 184.25.59.69:80
2/01/2025-17:54:04.235834  [**] [1:2031071:5] ET INFO Microsoft Connection Test [**] [Classification: Misc activity] [Priority: 3] {TCP} 10.135.8.37:55867 → 184.25.59.69:80
2/01/2025-17:55:19.650482  [**] [1:2210030:3] SURICATA STREAM FIN invalid ack [**] [Classification: Generic Protocol Command Decode] [Priority: 3] {TCP} 10.135.8.37:58298 → 52.96.66.162:443
2/01/2025-17:55:19.657865  [**] [1:2210038:3] SURICATA STREAM FIN out of window [**] [Classification: Generic Protocol Command Decode] [Priority: 3] {TCP} 52.96.66.162:443 → 10.135.8.37:58298
2/01/2025-17:58:04.330036  [**] [1:2031071:5] ET INFO Microsoft Connection Test [**] [Classification: Misc activity] [Priority: 3] {TCP} 10.135.8.37:55899 → 184.25.59.89:80
```

3.2 Splunk Dashboard Output

Using the real-time viewer, Alerts were displayed instantly as Suricata wrote new events into eve.json. The Splunk report generated a clean table with fields:

- Time
- Source IP
- Destination IP
- Alert Name
- Category
- Severity

This table could be exported as PDF or CSV.



```
index="main" sourcetype="_json" event_type="alert"
| table _time, src_ip, dest_ip, alert.signature, alert.category, alert.severity
| rename _time as "Time", src_ip as "Source IP", dest_ip as "Destination IP", alert.signature as "Alert Name", alert.category as "Category", alert.severity a
| sort - "Time"
```

7 events (12/1/25 6:00:15.000 PM to 12/1/25 6:15:15.000 PM)

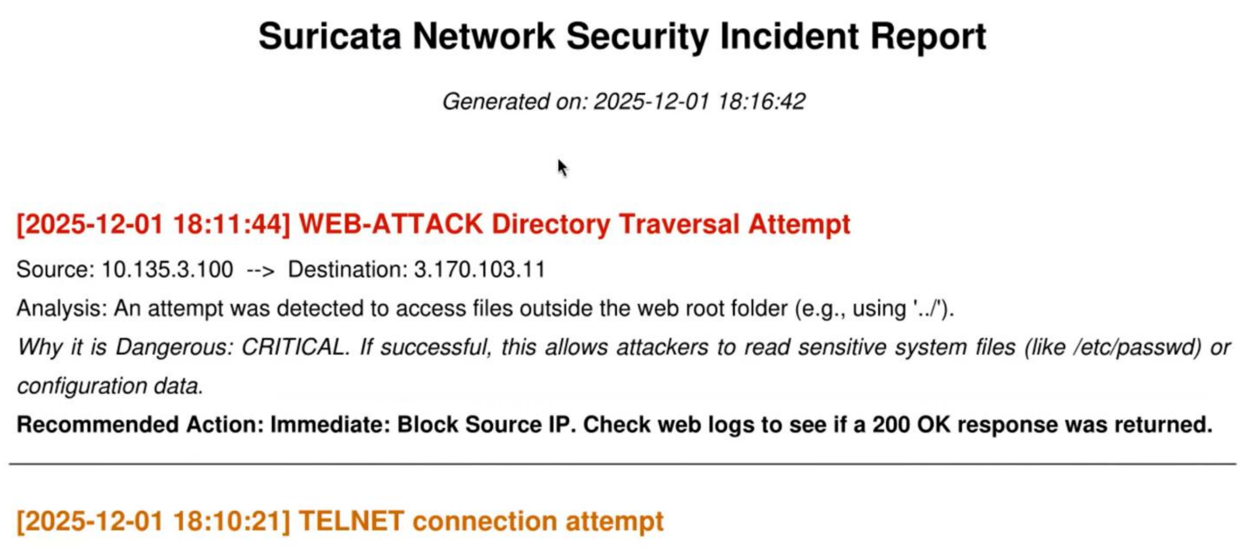
Time	Source IP	Destination IP	Alert Name
1764630704.430356	10.135.3.100	3.170.103.11	WEB-ATTACK Directory Traversal Attempt
1764630649.348551	10.135.3.100	138.87.128.1	POLICY VIOLATION: Facebook Access
1764630649.300791	10.135.3.100	138.87.128.1	POLICY VIOLATION: Facebook Access
1764630625.876498	10.135.3.100	10.135.45.199	FTP connection attempt
1764630621.101312	10.135.3.100	10.135.45.199	TELNET connection attempt
1764630590.011931	10.135.3.100	10.135.45.199	ICMP Ping
1764630590.008729	10.135.45.199	10.135.3.100	ICMP Ping

3.3 Custom Report Script Output

The Python reporting tool produced a professionally formatted PDF showing:

- Most frequent alerts
- Severity levels
- Top source IPs
- Remediation recommendations based on signature categories

The final report functions as a forensic summary of all detected network events.



Suricata Network Security Incident Report

Generated on: 2025-12-01 18:16:42

[2025-12-01 18:11:44] WEB-ATTACK Directory Traversal Attempt

Source: 10.135.3.100 --> Destination: 3.170.103.11

Analysis: An attempt was detected to access files outside the web root folder (e.g., using '../').

Why it is Dangerous: CRITICAL. If successful, this allows attackers to read sensitive system files (like /etc/passwd) or configuration data.

Recommended Action: Immediate: Block Source IP. Check web logs to see if a 200 OK response was returned.

[2025-12-01 18:10:21] TELNET connection attempt

4. Lessons Learned/Conclusion

4.1 Lessons Learned

- The Suricata Custom rules worked
- Both the created script and the SPL script worked
- Splunk was inconsistent with the saving and using it again after it was saved
- Real time monitoring was very buggy, had to change to 15 minutes of generated alerts
- Time management was difficult, finding enough time to do bigger parts of the project
- Starting the project sooner
- Not being able to access or use Spunk and resorting to having to wait longer for the tool so we only had to use the free trail

4.2 Conclusion

This project successfully demonstrates how Suricata and Splunk can be combined to create a powerful detection and forensic reporting tool. Suricata captures network activity through both built-in and custom rules, while Splunk organizes and visualizes alerts in real time. The Python reporting tool adds automated documentation capabilities, providing complete visibility into network threats. This system addresses several challenges in digital forensics including disorganized logs, delayed alerting, and lack of automated reporting. It also delivers a practical solution that can be extended into larger enterprise environments in the future.

5. References

- https://www.youtube.com/watch?v=UXKbh0jPPpg
- https://docs.suricata.io/en/latest/rules/intro.html
- https://www.youtube.com/watch?v=bLW_hBgRqUg