Project Report: WordPress Hosting, Security Monitoring, and Attack Simulation on Ubuntu Server

Report Issued : 26/06/2025

Submitted by : Rashid Ok

Table of Contents

1. Introduction

This project focuses on deploying a secure WordPress hosting environment using Ubuntu 24.04 on VirtualBox. Key security integrations include Splunk for centralized logging, Snort for intrusion detection, Fail2Ban for automated IP banning, and the ELK stack (Elasticsearch, Logstash, Kibana) for comprehensive log analysis. The project concludes with attack simulations (Brute Force, SQL Injection, XSS) to validate monitoring and defense mechanisms.

1.1 Project Overview

This project demonstrates the deployment and security monitoring of a self-hosted WordPress website on an Ubuntu 24.04 server. The goal is to simulate a production-like environment with layered security and logging mechanisms to detect and prevent cyberattacks such as brute force, SQL injection (SQLi), and cross-site scripting (XSS).

Security solutions like **Snort** (Intrusion Detection System), **Fail2Ban** (Intrusion Prevention System), **Splunk**, and the **ELK Stack** (Elasticsearch, Logstash, Kibana, Filebeat) were configured to collect, analyze, and visualize logs from Apache, WordPress, MySQL, and Snort.

Additionally, various simulated attacks were executed to validate the effectiveness of the detection and prevention systems.

1.2 Objectives

- **Deploy a functional WordPress site** using Apache, MySQL, and PHP.
- **Configure Snort IDS** with custom rules to detect specific attack patterns (SQLi, XSS, Command Injection, brute force).
- **Implement Fail2Ban** to automatically ban IPs based on Snort alerts and Apache logs.
- **Set up Splunk and ELK Stack** to collect and visualize logs from WordPress, Apache, and Snort for real-time monitoring.
- **Perform attack simulations** (e.g., SQLi, brute force, XSS) and verify detection and prevention mechanisms.

- **Build hands-on experience** in Linux server administration, log forwarding, IDS/IPS tuning, and SIEM dashboard usage.

2. Tools and Sub-Tools Overview

2.1 ELK Stack

- **Elasticsearch**: Stores and indexes logs collected from various sources (syslog, Apache logs, Snort alerts) for fast searching and analytics.
- **Logstash**: Acts as a data processing pipeline, ingesting logs from multiple files, parsing, and forwarding them into Elasticsearch.
- **Kibana**: Provides a web-based user interface to visualize and analyze log data stored in Elasticsearch.
- **Filebeat** (optional, if used): Lightweight log shipper installed on Ubuntu to forward system and application logs to Logstash or Elasticsearch.

2.2 Splunk

- **Splunk Enterprise (Windows)**: Centralized platform for collecting, indexing, and visualizing logs forwarded from Ubuntu server.
- **Splunk Universal Forwarder (Ubuntu)**: Installed on Ubuntu to monitor key log files (auth.log, syslog, Apache access/error logs, Snort alerts) and forward them securely to Splunk Enterprise for centralized analysis.

2.3 Snort IDS

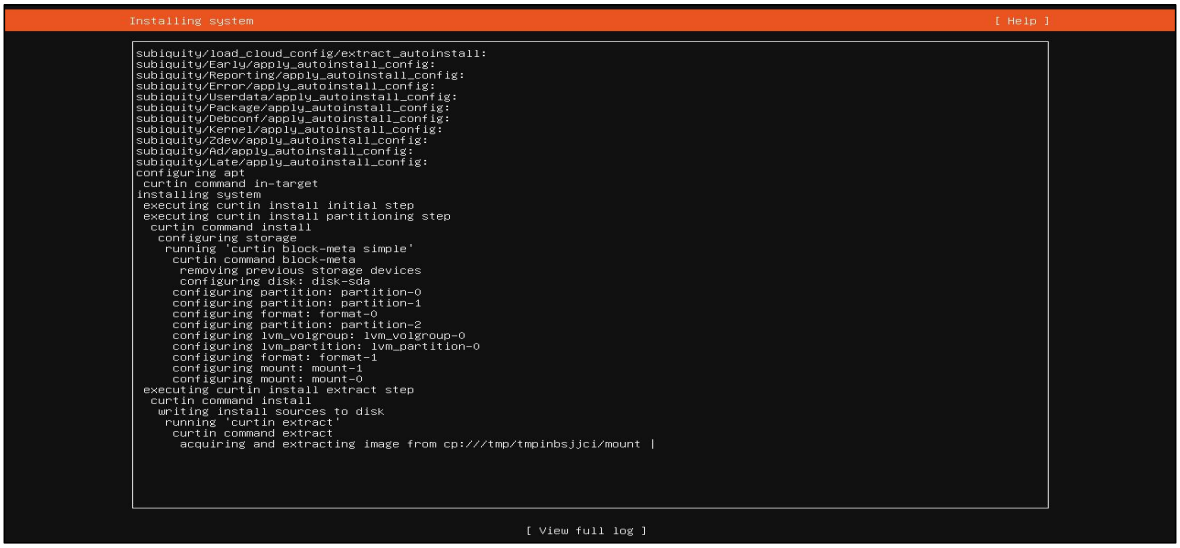
- Network Intrusion Detection System monitoring traffic on the server.
- Detects various attacks (ICMP scanning, SSH attempts, HTTP access, WordPress login attempts, SQL injection, XSS, command injection).
- Custom Snort rules written to identify specific threats relevant to WordPress and SSH.
- Outputs alerts logged locally and forwarded to Splunk/ELK for monitoring.

2.4 Fail2Ban

- Monitors log files for suspicious patterns (using custom filters based on Snort alerts and Apache logs).
- Filters located in /etc/fail2ban/filter.d/ define patterns for brute force and web attacks (WordPress brute force, SQLi, XSS).
- Jails configured in /etc/fail2ban/jail.local specify which logs to monitor and ban policies (ban time, retry limits).
- Automatically bans IP addresses exhibiting malicious behavior to prevent continued attacks.

3 . Environment Setup

Insatalling Ubuntu server in VirtualBox/Vmware



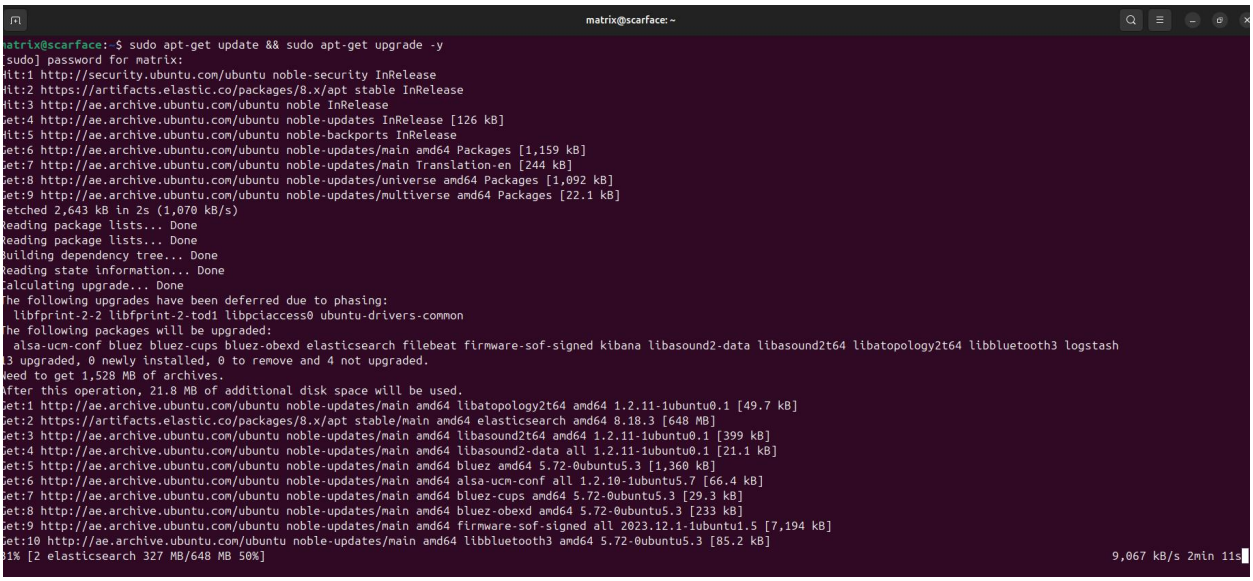
3.1 Updating and Upgrading Ubuntu Packages

- **Update package lists** – Fetches the latest list of available packages.

  sudo apt-get update

- **Upgrade installed packages** – Installs latest versions of installed software.

sudo apt-get upgrade -y



4.Apache and Wordpress Installation and Configuration

4.1 Installing Apache, PHP, MySQL, and Required Modules

Install Apache, PHP, MySQL and required modules – Provides the web and database server environment for WordPress.

sudo apt install apache2 php libapache2-mod-php mysql-server php-mysql -y

4.2 Starting and Enabling Apache Service

- Enable Apache to start at boot – Ensures Apache runs after system reboots.

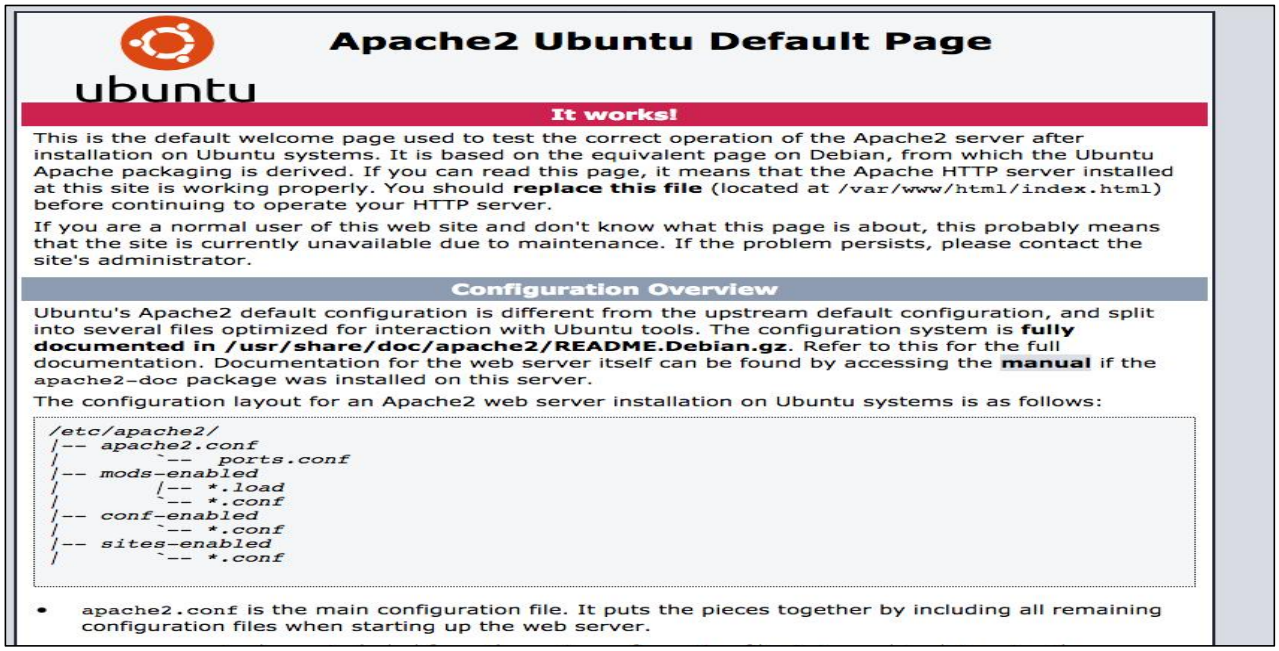
  sudo systemctl enable apache2

- Start Apache service immediately.

  sudo systemctl start apache2

- Check Apache service status.

sudo systemctl status apache2



4.3 WordPress Download and Setup

- Navigate to web root directory.
- Download latest WordPress package.

  sudo wget https://wordpress.org/latest.tar.gz

- Extract WordPress archive.

  sudo tar -xvzf latest.tar.gz

- Move WordPress files to web root.

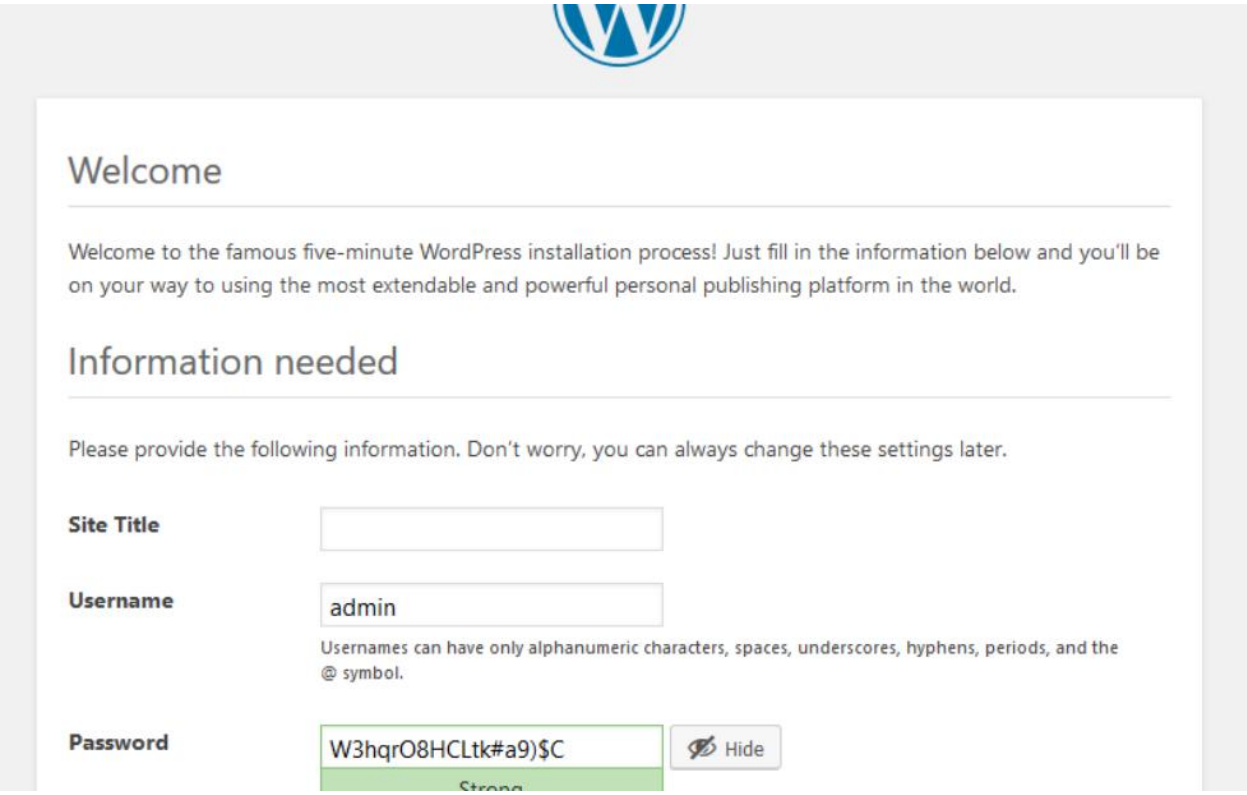
  sudo mv wordpress/* ./

- Set ownership to Apache user (www-data).

  sudo chown -R www-data:www-data /var/www/html

- Set appropriate file permissions.

  sudo chmod -R 755 /var/www/html



4.4 Configuring MySQL for WordPress

- **Access MySQL shell**.

  sudo mysql -u root -p

- **Create WordPress database and user with privileges**.

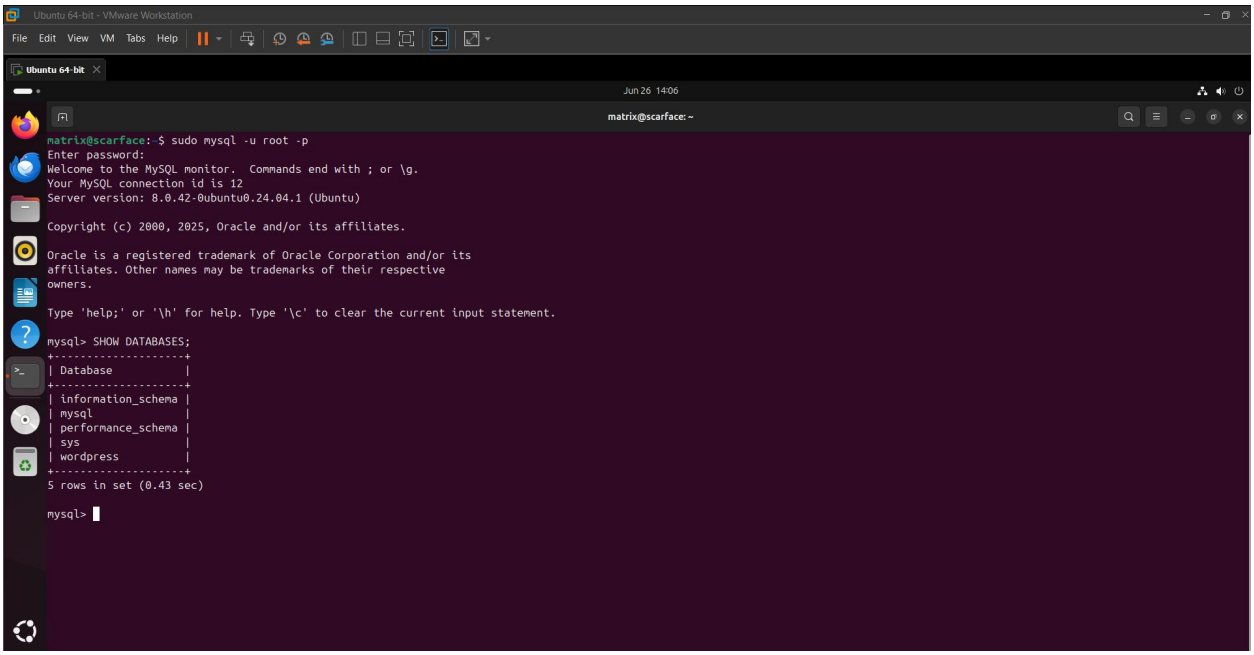
  CREATE DATABASE wordpress;

  CREATE USER 'wpuser'@'localhost' 'wpuser'@'localhost';

  FLUSH PRIVILEGES;

  EXIT;

(i’am done already)



4.5 Apache Virtual Host Configuration

- **Create virtual host configuration file**.

  sudo nano /etc/apache2/sites-available/wordpress.conf

- **Paste the following virtual host configuration**.

  <VirtualHost *:80>
  ServerAdmin webmaster@localhost
  DocumentRoot /var/www/html
  ServerName yourdomain.com
  <Directory /var/www/html>
  Options Indexes FollowSymLinks
  AllowOverride All
  Require all granted

```
    </Directory>
    ErrorLog ${APACHE_LOG_DIR}/error.log
    CustomLog ${APACHE_LOG_DIR}/access.log combined
</VirtualHost>
```

- **Enable the new site and reload Apache**.

```
sudo a2ensite wordpress.conf
sudo systemctl reload apache2
```

4.6 WordPress Configuration File Setup

- **Rename sample config file**.

```
sudo mv wp-config-sample.php wp-config.php
```

- **Edit database connection details**.

```
sudo nano wp-config.php
```

- **Update DB settings**.

```
define('DB_NAME', 'wordpress');define('DB_USER',
'wpuser');define('DB_PASSWORD', 'password');define('DB_HOST', 'localhost');
```

3. Splunk Enterprise on Windows

Purpose

Splunk Enterprise acts as a centralized log collection and analysis platform. Itreceives logs forwarded from the Ubuntu server and provides a user-friendly Interface for searching, alerting, and visualizing data.

Steps and Commands

5.1 Accessing Splunk Web Interface

- **Access Splunk Web Interface**

  Open your browser and navigate to:

  http://127.0.0.1:8000

  Login with your credentials.

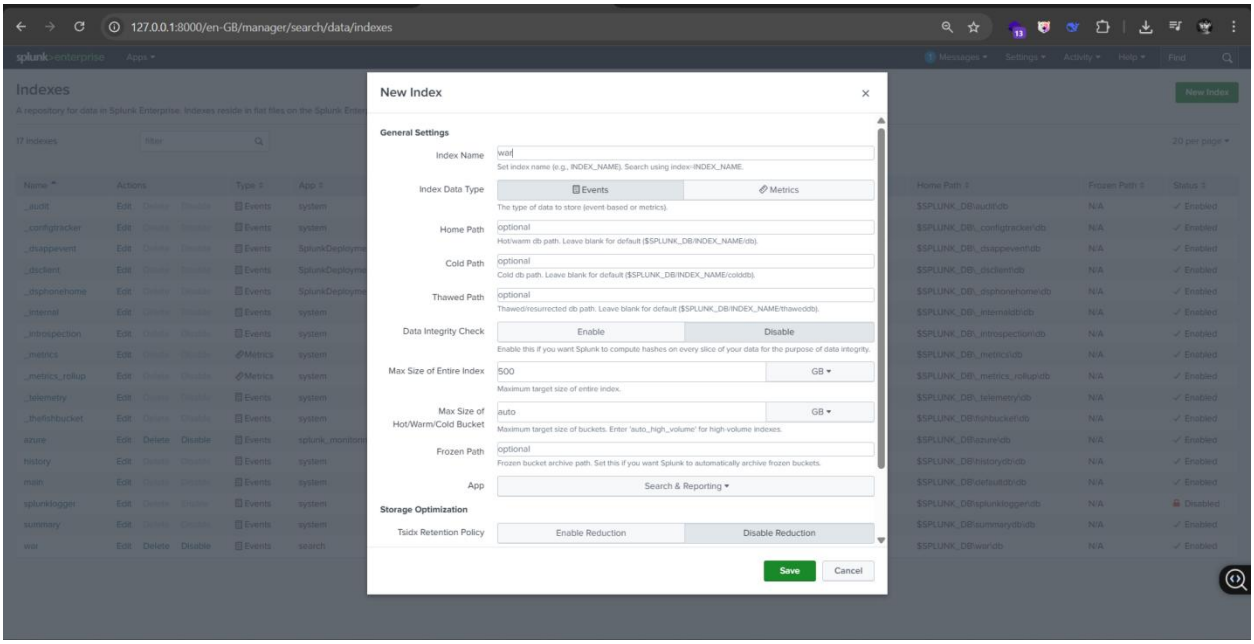
5.2 Creating Index for Log Storage

Create a new index named "war"

This index stores logs forwarded from Ubuntu.

Navigate to:

Settings > Indexes > New Index
Enter war and save.

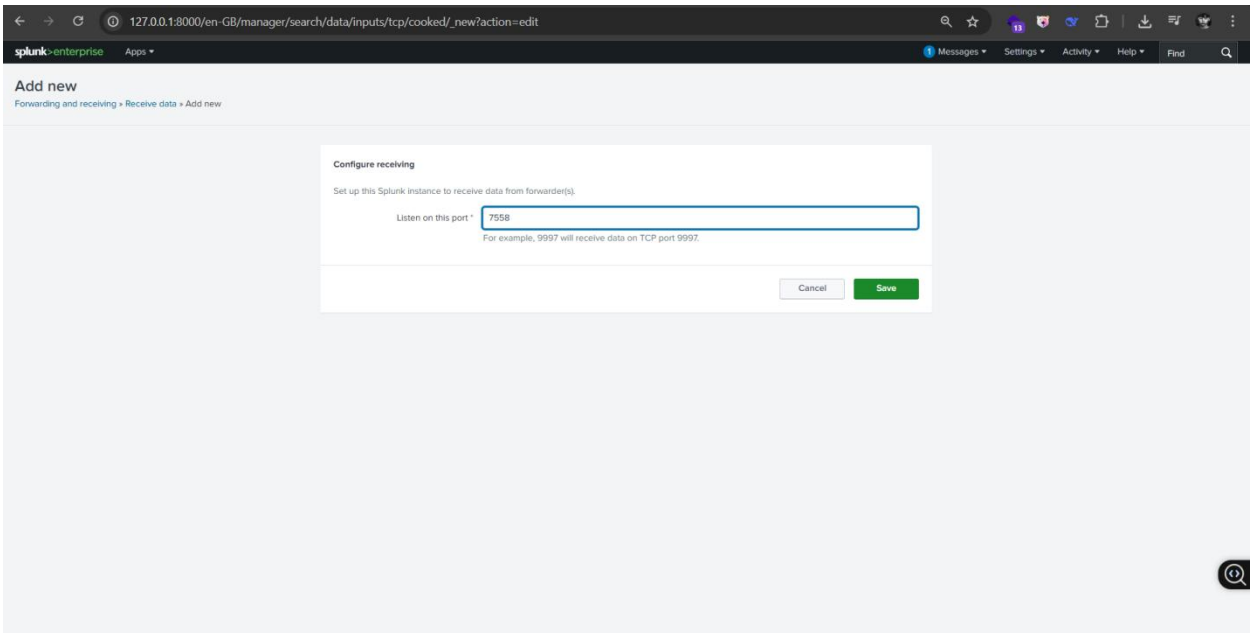


5.3 Configuring Receiving Port for Log Forwarding

Navigate to:

Settings > Forwarding and Receiving > Configure Receiving > Add New

Enter port 7558 and enable.



4. Splunk Universal Forwarder on Ubuntu

Purpose

Splunk Universal Forwarder runs on Ubuntu to monitor critical log files and securely forward them to Splunk Enterprise for central analysis.

Installation and Configuration Steps

6.1 Installation and Extraction

Extract Splunk Forwarder package (adjust filename accordingly)

```
sudo tar xvzf /tmp/splunkforwarder--Linux--bit.tgz -C /opt
```

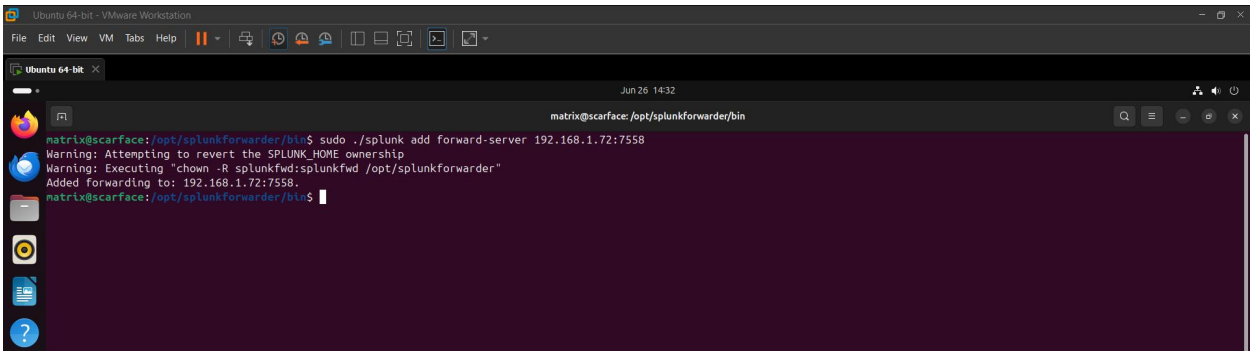
6.2 Starting Forwarder and Accepting License

```
sudo ./splunk start --accept-license
```

6.3 Configuring Forward Server Connection

- **Add Splunk Enterprise server as forward-server** (replace <Windows_IP> with actual IP):

```
sudo ./splunk add forward-server <Windows_IP>:7558
```



6.4 Adding Log File Monitors

- Add monitors for critical logs:

```
sudo ./splunk add monitor /var/log/auth.log -index war
sudo ./splunk add monitor /var/log/syslog -index war
sudo ./splunk add monitor /var/log/apache2/access.log -index war
sudo ./splunk add monitor /var/log/apache2/error.log -index war
sudo ./splunk add monitor /var/log/snort/snort.alert.fast -index war
```

6.5 Manual Configuration via inputs.conf

Alternative: Manual inputs.conf editing

sudo nano /opt/splunkforwarder/etc/system/local/inputs.conf

Example content:

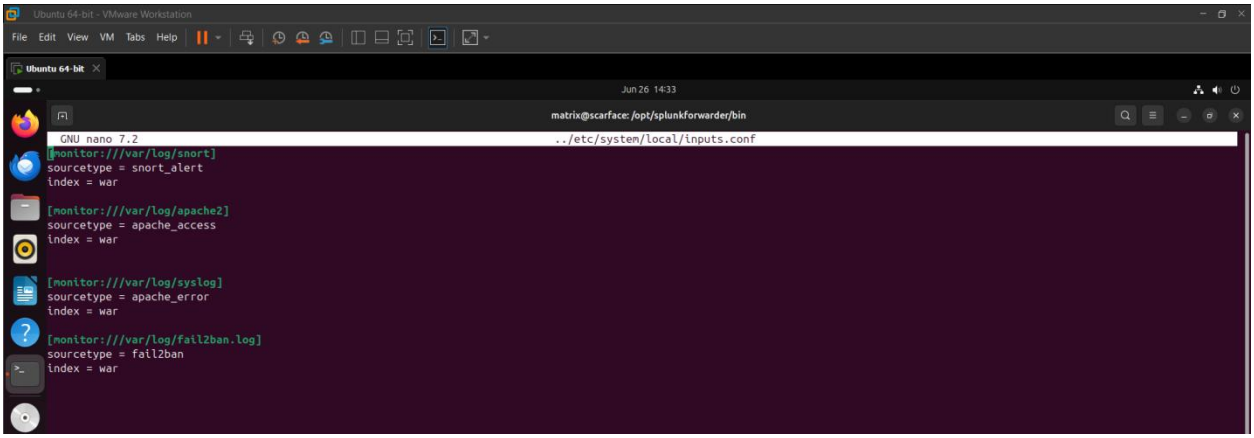
[monitor:///var/log/auth.log]
index = war

[monitor:///var/log/syslog]
index = war

[monitor:///var/log/apache2/access.log]
index = war

[monitor:///var/log/apache2/error.log]
index = war

[monitor:///var/log/snort/snort.alert.fast]
index = war



5. ELK Stack Installation and Configuration on Ubuntu

Purpose

The ELK stack collects, processes, and visualizes logs locally on the Ubuntu server, providing another method for detailed log analysis and dashboarding.

Installation and Commands

7.1 Installing Elasticsearch

Install Elasticsearch

```
sudo apt update
sudo apt install elasticsearch -y
sudo systemctl enable elasticsearch
sudo systemctl start elasticsearch
```

7.2 Installing and Configuring Logstash

Install Logstash

```
sudo apt install logstash -y
```

Create Logstash configuration

```
sudo nano /etc/logstash/conf.d/logstash.conf
```

Add:

```
input {
  file {
    path => [
      "/var/log/syslog",
      "/var/log/apache2/access.log",
      "/var/log/apache2/error.log",
```

```
    "/var/log/snort/snort.alert.fast"
  ]
  start_position => "beginning"
  sincedb_path => "/dev/null"
 }
}

output {
 elasticsearch {
  hosts => ["localhost:9200"]
  index => "logs-%{+YYYY.MM.dd}"
 }
 stdout { codec => rubydebug }
}
```

- Start and enable Logstash

```
sudo systemctl enable logstash
sudo systemctl start logstash
```

7.3 Installing and Starting Kibana

- Install Kibana

```
sudo apt install kibana -y
sudo systemctl enable kibana
sudo systemctl start kibana
```

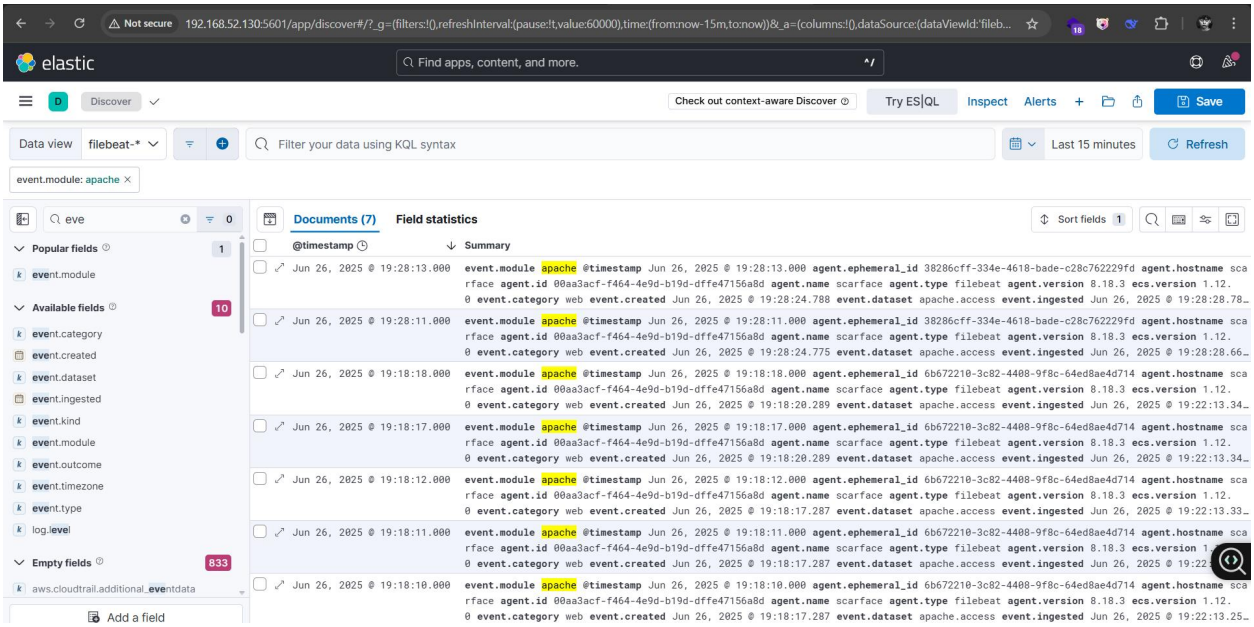
7.4 Accessing and Using Kibana Dashboard

Access Kibana Dashboard

Open browser and navigate to:

http://<your-ubuntu-ip>:5601

Configure index patterns to start visualizing the logs.



6. Snort IDS Installation and Configuration

Purpose

Snort analyzes network traffic to detect malicious activities based on custom and standard rules, alerting when suspicious packets are seen.

Commands and Setup

8.1 Directory Setup for Snort

Create necessary directories

sudo mkdir -p /etc/snort/rules /etc/snort/preproc_rules /var/log/snort /usr/local/lib/snort_dynamicrules

8.2 Editing Snort Configuration

Edit Snort configuration

sudo nano /etc/snort/snort.conf

Set variables:

var HOME_NET 192.168.1.0/24var EXTERNAL_NET anyvar RULE_PATH /etc/snort/rules

Include rules:

include $RULE_PATH/local.rules

include $RULE_PATH/community.rules

8.3 Adding and Customizing Snort Rules

Add custom rules to /etc/snort/rules/local.rules

Examples (including your new rules):

- alert tcp any any -> any 80 (msg:"Web Attack - Command Injection system("; content:"system("; nocase; sid:1002009; rev:1;)

- alert tcp any any -> $HOME_NET 80 (msg:"WordPress Bruteforce Attempt"; content:"POST"; content:"/wp-login.php"; threshold:type threshold, track by_src, count 5, seconds 30; sid:1006001; rev:1;)

- alert tcp any any -> $HOME_NET 80 (msg:"SQL Injection Attempt - ' OR 1=1"; content:"' OR 1=1"; nocase; sid:1002101; rev:1;)

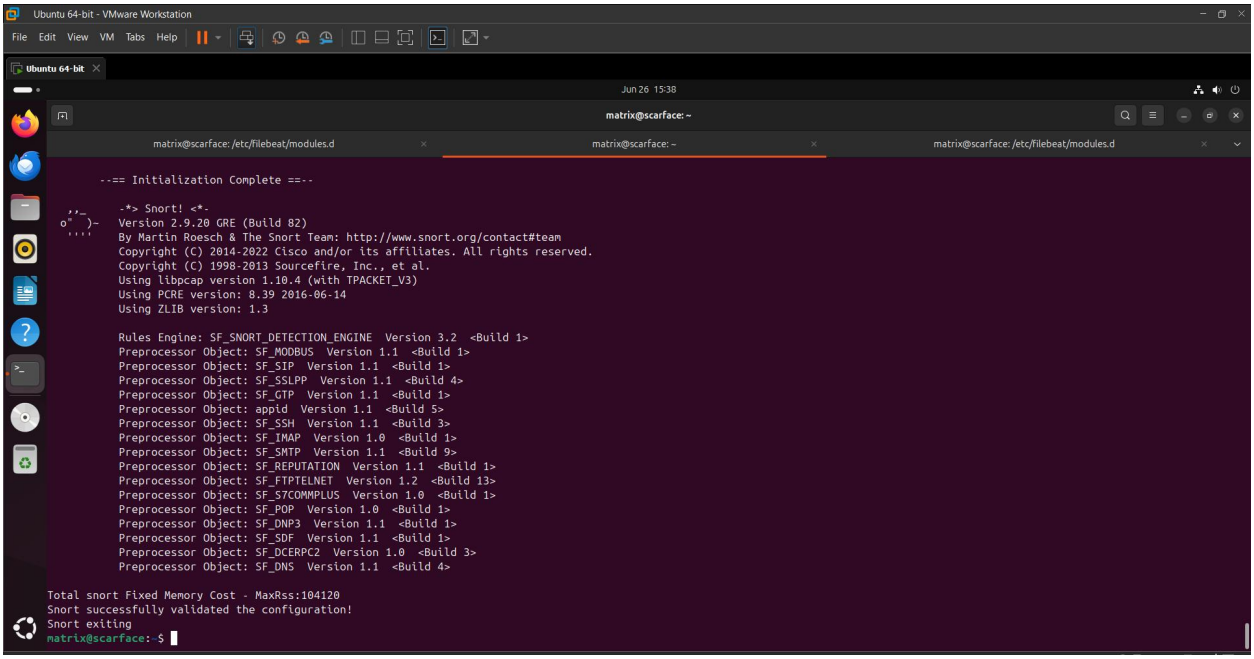
- alert tcp any any -> $HOME_NET 80 (msg:"XSS Attack Attempt - <script>"; content:"<script>"; nocase; sid:1002001; rev:1;)

- alert tcp any any -> $HOME_NET 22 (msg:"SSH Brute Force Attempt"; flags:S; threshold:type threshold, track by_src, count 5, seconds 30; sid:1003008; rev:1;)

8.4 Testing Snort Configuration

Test Snort configuration

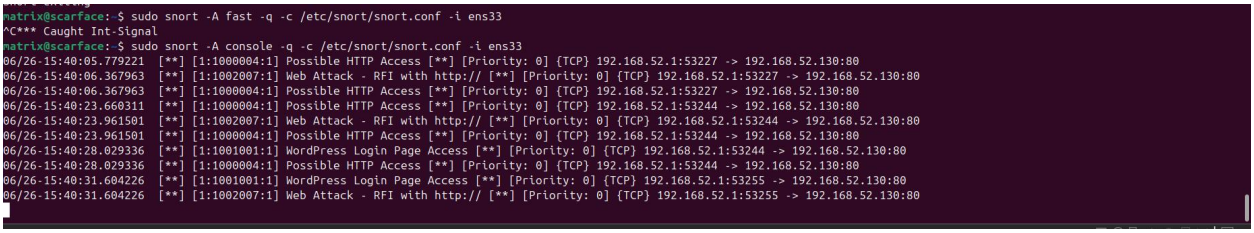
sudo snort -T -c /etc/snort/snort.conf



8.5 Running Snort in Alert Mode

Run Snort in alert mode

sudo snort -A console -q -c /etc/snort/snort.conf -i ens33



7. Fail2Ban Configuration

Purpose

Fail2Ban monitors logs (including Snort alerts and Apache logs) and bans IPs that exhibit suspicious behavior such as brute force attacks or injections.

Commands and Configuration

9.1 Installing Fail2Ban

Install Fail2Ban

sudo apt install fail2ban -y

9.2 Creating Custom Filters for WordPress and Snort Alerts

Create filter files in /etc/fail2ban/filter.d/

Example: wordpress.conf

[Definition]
failregex = <HOST> -.*"(POST|GET).*wp-login.php.*" 200
ignoreregex =

Example: snort-sql-injection.conf

[Definition]

failregex = \[**\] \[.*\] Web Attack - SQL Injection.*\[**\].*{.*-><HOST>}
ignoreregex =

Example: snort-xss.conf

```
[Definition]
failregex = \[\*\*\] \[.*\] Web Attack - XSS.*\[\*\*\].*{.*-    ><HOST>}
ignoreregex =
```

9.3 Configuring Jails for SSH, WordPress, and Snort Filters

Edit /etc/fail2ban/jail.local

```
[sshd]
enabled = true
port = ssh
logpath = /var/log/auth.log
maxretry = 5
findtime = 60
bantime = 1800

[wordpress]
enabled = true
filter = wordpress
port = http,https
logpath = /var/log/apache2/access.log
maxretry = 5
```

```
findtime = 60
bantime = 1800

[snort-sql-injection]
enabled = true
filter = snort-sql-injection
logpath = /var/log/snort/snort.alert.fast
maxretry = 5
findtime = 600
bantime = 1800

[snort-xss]
enabled = true
filter = snort-xss
logpath = /var/log/snort/snort.alert.fast
maxretry = 5
findtime = 600
bantime = 3600
```

9.4 Restarting Fail2Ban and Monitoring Logs

Restart Fail2Ban

```
sudo systemctl restart fail2ban
sudo systemctl status fail2ban
```

View Fail2Ban logs to verify bans

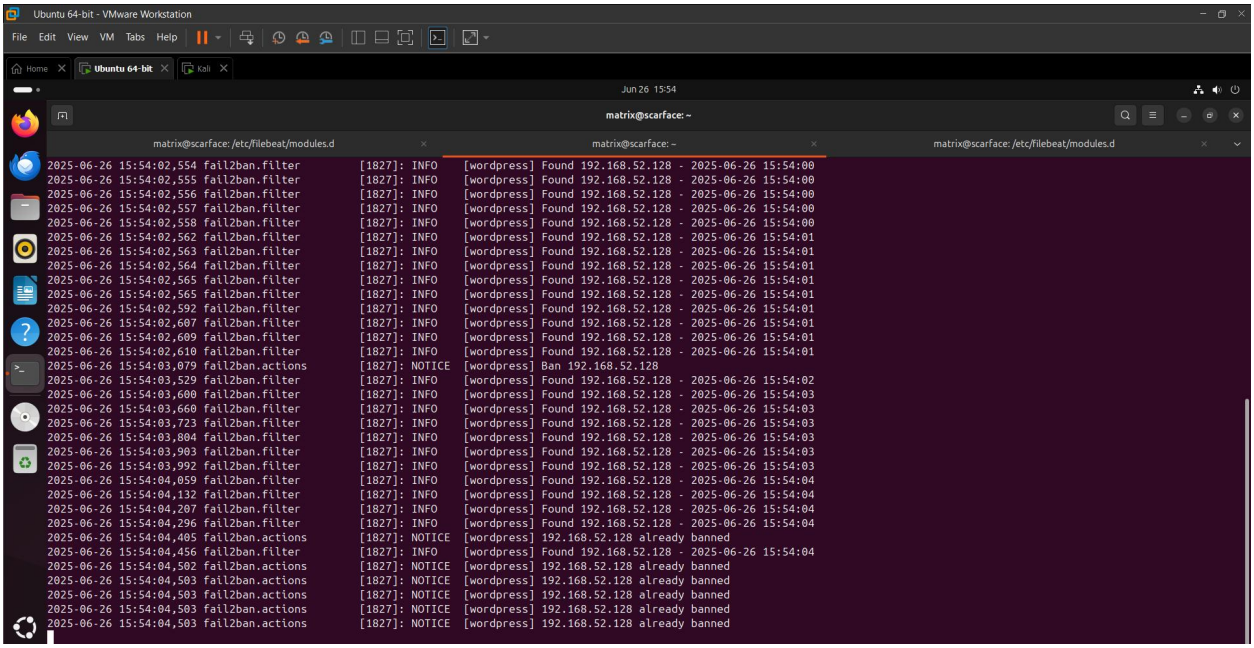
```
sudo tail -f /var/log/fail2ban.log
```

Restart Fail2Ban

```
sudo systemctl restart fail2ban
sudo systemctl status fail2ban
```

View Fail2Ban logs to verify bans

```
sudo tail -f /var/log/fail2ban.log
```



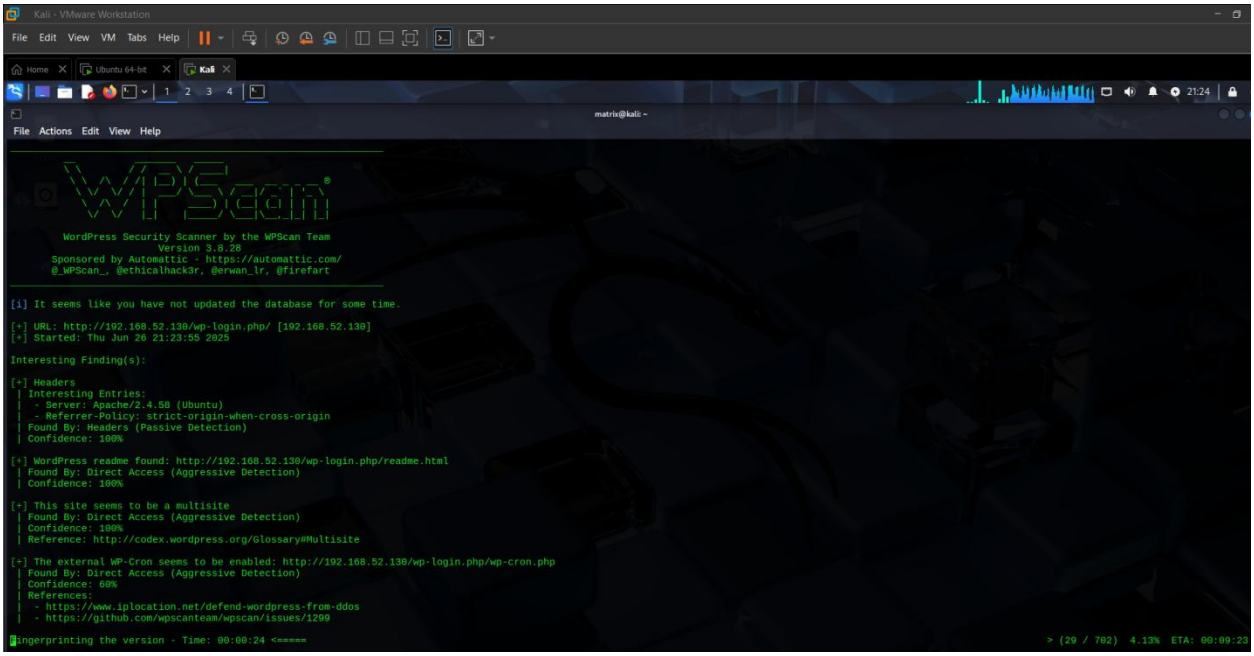
10. Attack Simulation and Verification

Executing Brute Force Attack Simulations

Perform brute force login attempts

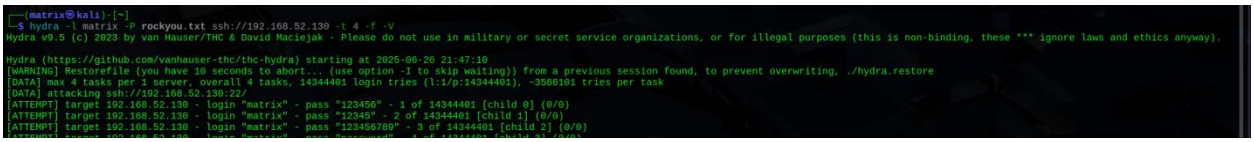
Wordpress

```
wpscan --url http://<target-ip>/wp-login.php --usernames matrix --passwords /path/to/wordlist.txt
```

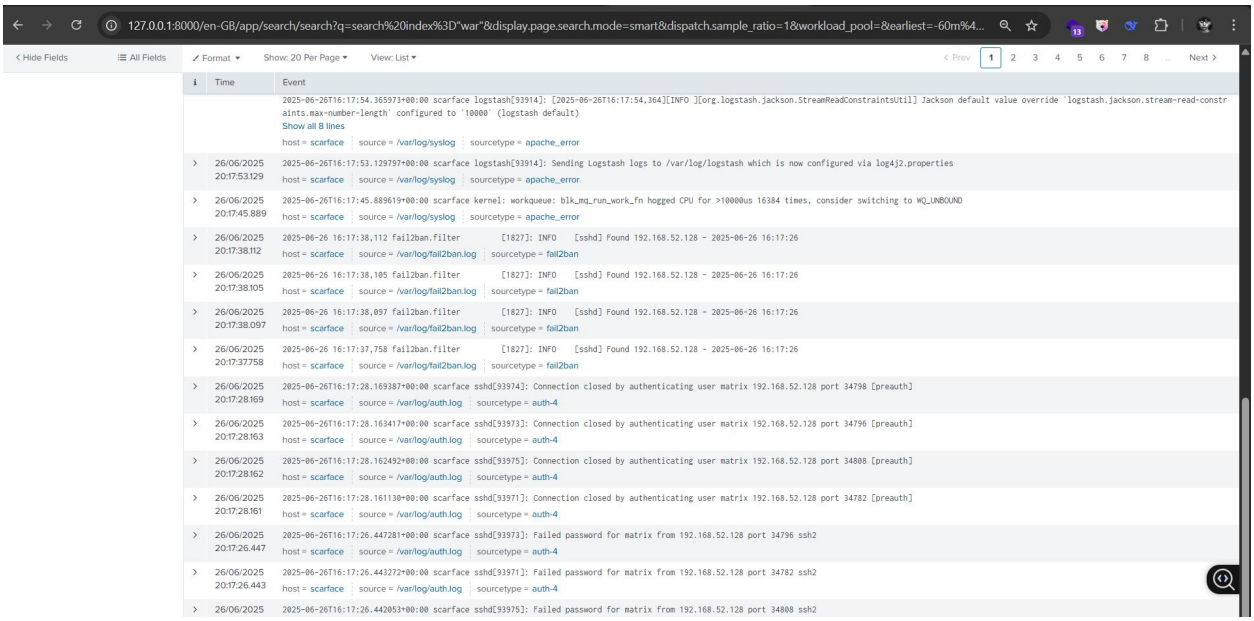


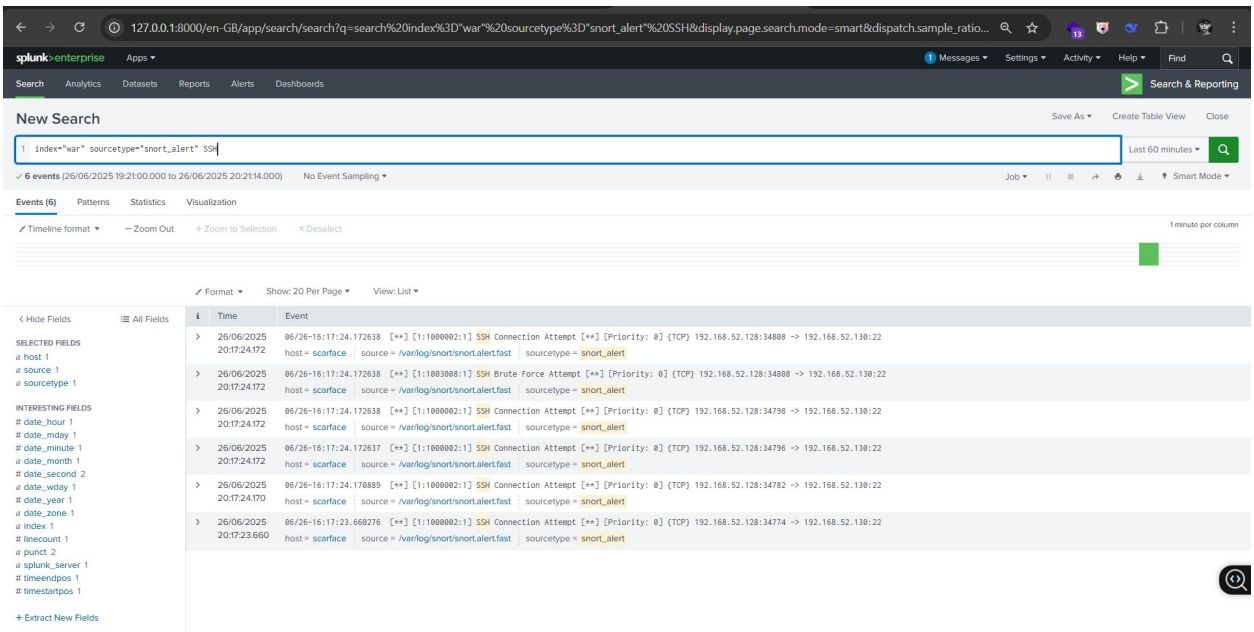
Ssh

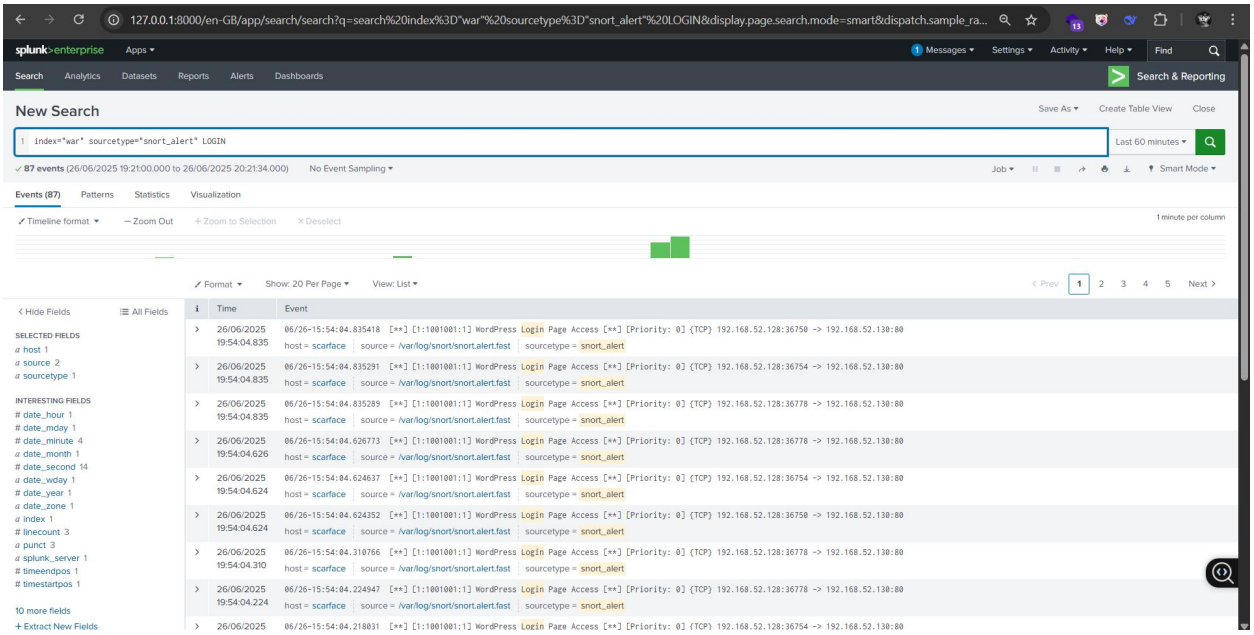
hydra -l matrix -P rockyou.txt ssh://192.168.52.130 -t 4 -f -V



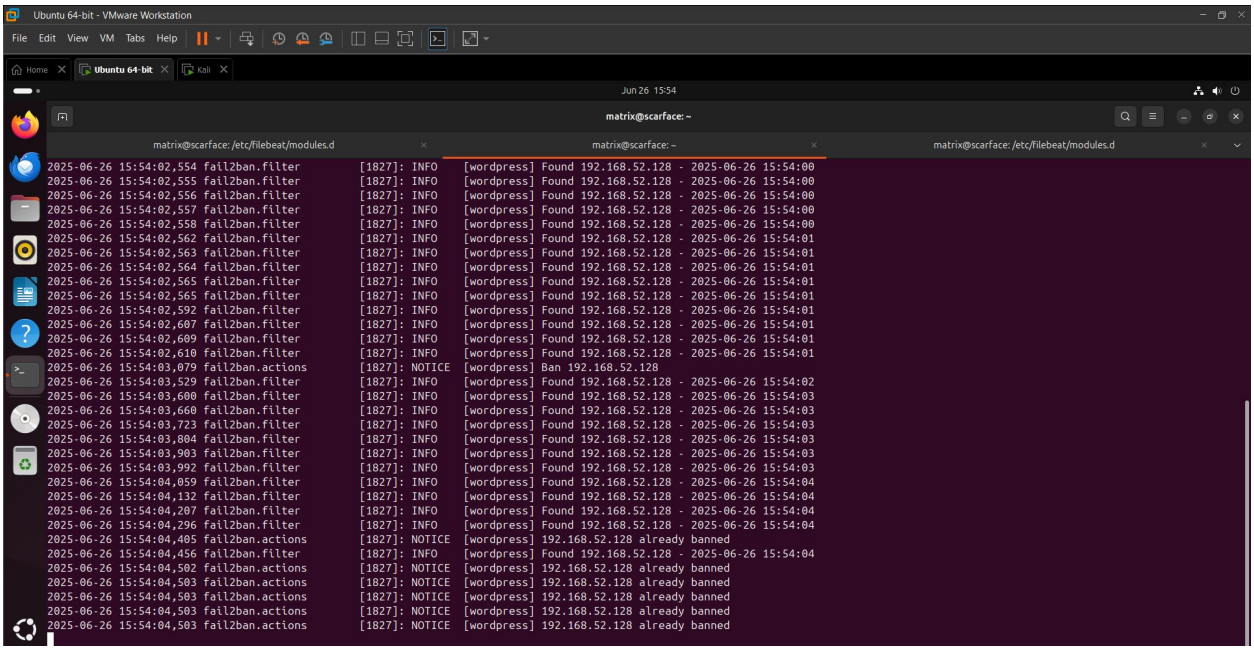
10.1 Monitoring Attack Logs in Splunk and Kibana


127.0.0.1:8000/en-GB/app/search/search?q=search%20index%3D"war"&display.page.search.mode=smart&dispatch.sample_ratio=1&workload_pool=&earliest=-60m%4...
Hide Fields
All Fields
Format
Show: 20 Per Page
View: List
Prev
1 2 3 4 5 6 7 8 ... Next
Time
Event
2025-06-26T16:17:54.365973+00:00 scarface logstash[93914]: [2025-06-26T16:17:54,364][INFO][org.logstash.jackson.StreamReadConstraintsUtil] Jackson default value override `logstash.jackson.stream-read-constraints.max-number-length` configured to `10000` (logstash default)
Show all 8 lines
host = scarface source = /var/log/syslog sourcetype = apache_error
26/06/2025 20:17:53.129
2025-06-26T16:17:53.129797+00:00 scarface logstash[93914]: Sending Logstash logs to /var/log/logstash which is now configured via log4j2.properties
host = scarface source = /var/log/syslog sourcetype = apache_error
26/06/2025 20:17:45.889
2025-06-26T16:17:45.889619+00:00 scarface kernel: workqueue: blk_mq_run_work_fn hogged CPU for >10000us 16384 times, consider switching to WQ_UNBOUND
host = scarface source = /var/log/syslog sourcetype = apache_error
26/06/2025 20:17:38.112
2025-06-26 16:17:38,112 fail2ban.filter [1827]: INFO [sshd] Found 192.168.52.128 - 2025-06-26 16:17:26
host = scarface source = /var/log/fail2ban.log sourcetype = fail2ban
26/06/2025 20:17:38.105
2025-06-26 16:17:38,105 fail2ban.filter [1827]: INFO [sshd] Found 192.168.52.128 - 2025-06-26 16:17:26
host = scarface source = /var/log/fail2ban.log sourcetype = fail2ban
26/06/2025 20:17:38.097
2025-06-26 16:17:38,097 fail2ban.filter [1827]: INFO [sshd] Found 192.168.52.128 - 2025-06-26 16:17:26
host = scarface source = /var/log/fail2ban.log sourcetype = fail2ban
26/06/2025 20:17:37.758
2025-06-26 16:17:37,758 fail2ban.filter [1827]: INFO [sshd] Found 192.168.52.128 - 2025-06-26 16:17:26
host = scarface source = /var/log/fail2ban.log sourcetype = fail2ban
26/06/2025 20:17:28.169
2025-06-26T16:17:28.169387+00:00 scarface sshd[93974]: Connection closed by authenticating user matrix 192.168.52.128 port 34798 [preauth]
host = scarface source = /var/log/auth.log sourcetype = auth-4
26/06/2025 20:17:28.163
2025-06-26T16:17:28.163417+00:00 scarface sshd[93973]: Connection closed by authenticating user matrix 192.168.52.128 port 34796 [preauth]
host = scarface source = /var/log/auth.log sourcetype = auth-4
26/06/2025 20:17:28.162
2025-06-26T16:17:28.162492+00:00 scarface sshd[93975]: Connection closed by authenticating user matrix 192.168.52.128 port 34808 [preauth]
host = scarface source = /var/log/auth.log sourcetype = auth-4
26/06/2025 20:17:28.161
2025-06-26T16:17:28.161130+00:00 scarface sshd[93971]: Connection closed by authenticating user matrix 192.168.52.128 port 34782 [preauth]
host = scarface source = /var/log/auth.log sourcetype = auth-4
26/06/2025 20:17:26.447
2025-06-26T16:17:26.447281+00:00 scarface sshd[93973]: Failed password for matrix from 192.168.52.128 port 34796 ssh2
host = scarface source = /var/log/auth.log sourcetype = auth-4
26/06/2025 20:17:26.443
2025-06-26T16:17:26.443272+00:00 scarface sshd[93971]: Failed password for matrix from 192.168.52.128 port 34782 ssh2
host = scarface source = /var/log/auth.log sourcetype = auth-4
26/06/2025
2025-06-26T16:17:26.442053+00:00 scarface sshd[93975]: Failed password for matrix from 192.168.52.128 port 34808 ssh2


127.0.0.1:8000/en-GB/app/search/search?q=search%20index%3D"war"%20sourcetype%3D"snort_alert"%20SSH&display.page.search.mode=smart&dispatch.sample_ratio...
splunk>enterprise
Apps
Messages
Settings
Activity
Help
Find
Search
Analytics
Datasets
Reports
Alerts
Dashboards
Search & Reporting
New Search
Save As
Create Table View
Close
index="war" sourcetype="snort_alert" SSH
Last 60 minutes
6 events (26/06/2025 19:21:00.000 to 26/06/2025 20:21:14.000)
No Event Sampling
Job
Smart Mode
Events (6)
Patterns
Statistics
Visualization
Timeline format
Zoom Out
Zoom to Selection
Deselect
1 minute per column
Format
Show: 20 Per Page
View: List
Hide Fields
All Fields
SELECTED FIELDS
a host 1
a source 1
a sourcetype 1
INTERESTING FIELDS
date_hour 1
date_mday 1
date_minute 1
a date_month 1
date_second 2
a date_wday 1
date_year 1
a date_zone 1
a index 1
linecount 1
a punct 2
a splunk_server 1
timeendpos 1
timestartpos 1
+ Extract New Fields
Time
Event
26/06/2025 20:17:24.172
06/26-16:17:24.172638 [**] [1:1000002:1] SSH Connection Attempt [**] [Priority: 0] {TCP} 192.168.52.128:34808 -> 192.168.52.130:22
host = scarface source = /var/log/snort/snort.alert.fast sourcetype = snort_alert
26/06/2025 20:17:24.172
06/26-16:17:24.172638 [**] [1:1003008:1] SSH Brute Force Attempt [**] [Priority: 0] {TCP} 192.168.52.128:34808 -> 192.168.52.130:22
host = scarface source = /var/log/snort/snort.alert.fast sourcetype = snort_alert
26/06/2025 20:17:24.172
06/26-16:17:24.172638 [**] [1:1000002:1] SSH Connection Attempt [**] [Priority: 0] {TCP} 192.168.52.128:34798 -> 192.168.52.130:22
host = scarface source = /var/log/snort/snort.alert.fast sourcetype = snort_alert
26/06/2025 20:17:24.172
06/26-16:17:24.172637 [**] [1:1000002:1] SSH Connection Attempt [**] [Priority: 0] {TCP} 192.168.52.128:34796 -> 192.168.52.130:22
host = scarface source = /var/log/snort/snort.alert.fast sourcetype = snort_alert
26/06/2025 20:17:24.170
06/26-16:17:24.170889 [**] [1:1000002:1] SSH Connection Attempt [**] [Priority: 0] {TCP} 192.168.52.128:34782 -> 192.168.52.130:22
host = scarface source = /var/log/snort/snort.alert.fast sourcetype = snort_alert
26/06/2025 20:17:23.660
06/26-16:17:23.660276 [**] [1:1000002:1] SSH Connection Attempt [**] [Priority: 0] {TCP} 192.168.52.128:34774 -> 192.168.52.130:22
host = scarface source = /var/log/snort/snort.alert.fast sourcetype = snort_alert



10.2 Verifying IP Bans with Fail2Ban



11. Conclusion

This project demonstrates a fully integrated cybersecurity setup for a WordPress hosting environment on Ubuntu, employing Apache, MySQL, Snort IDS, Splunk, ELK stack, and Fail2Ban for multi-layered protection. Attack simulations validate the effectiveness of detection, alerting, and automatic mitigation mechanisms, offering a solid foundation for real-world web server defense.