Apache2 Log Analysis

Assigment no -02

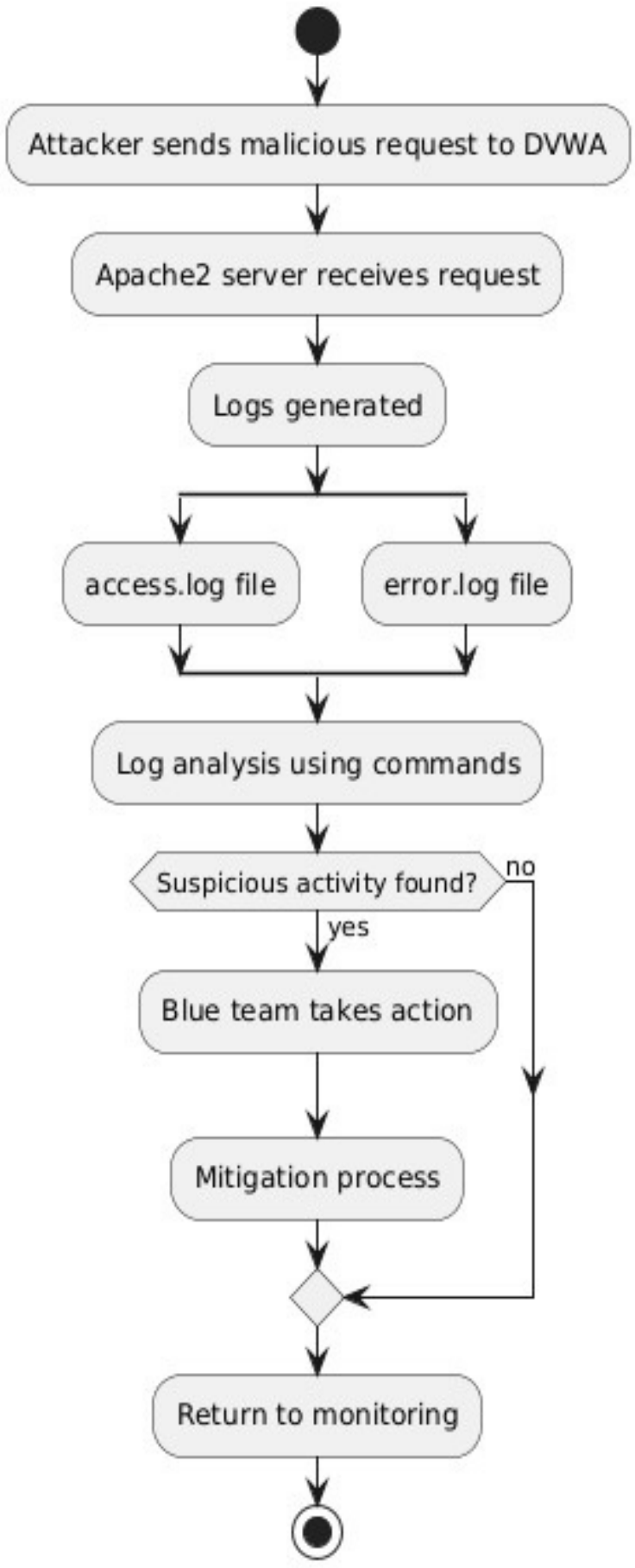
Presented By

Arhant Suhas Gaikwad

Computer Science Department

Research Methodology

1. Process flow diagram



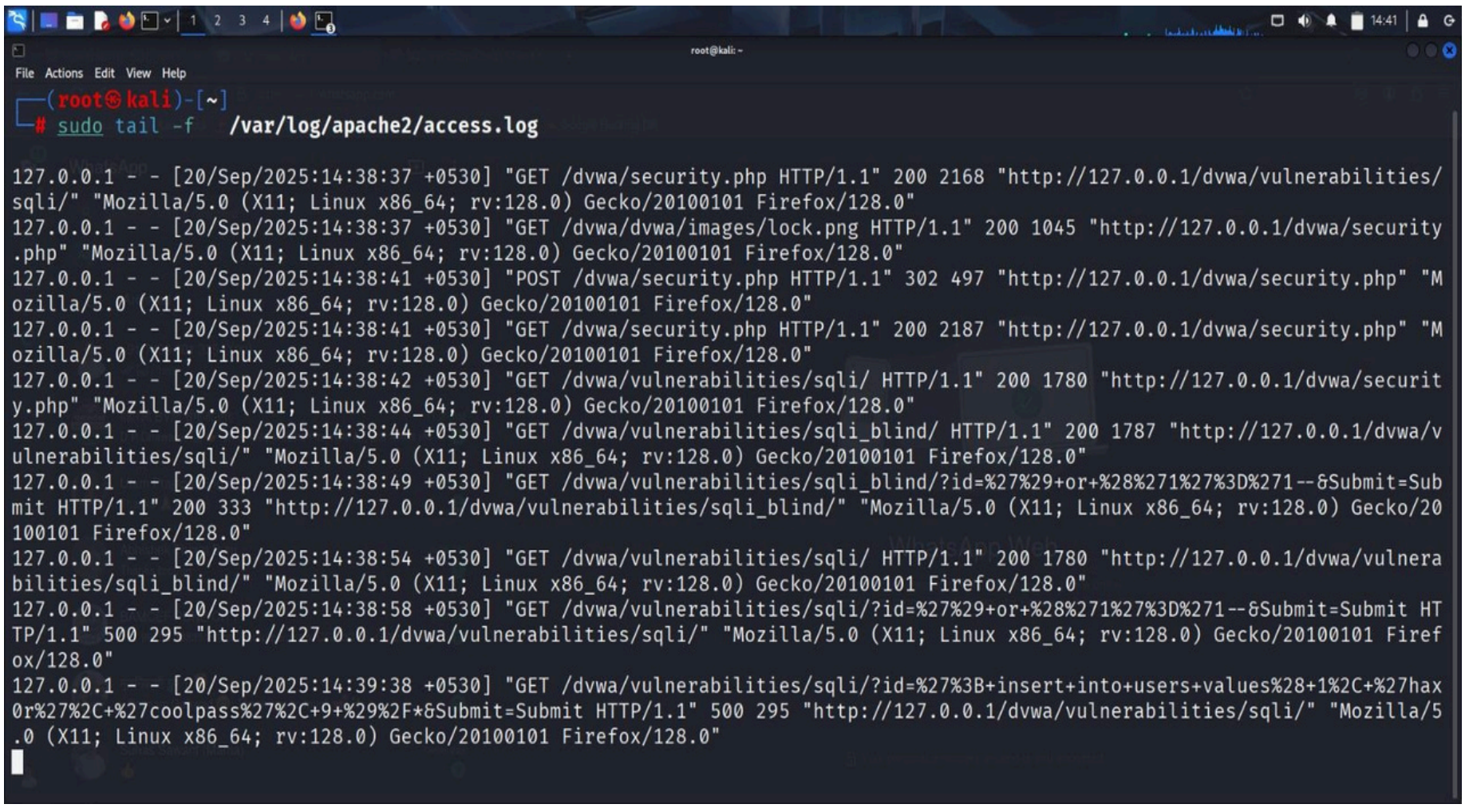
Practical Work

- **command**

1. sudo tail -f /var/log/apache2/access.log

Purpose : To monitor the Apache access log in real-time and see incoming requests as they happen.

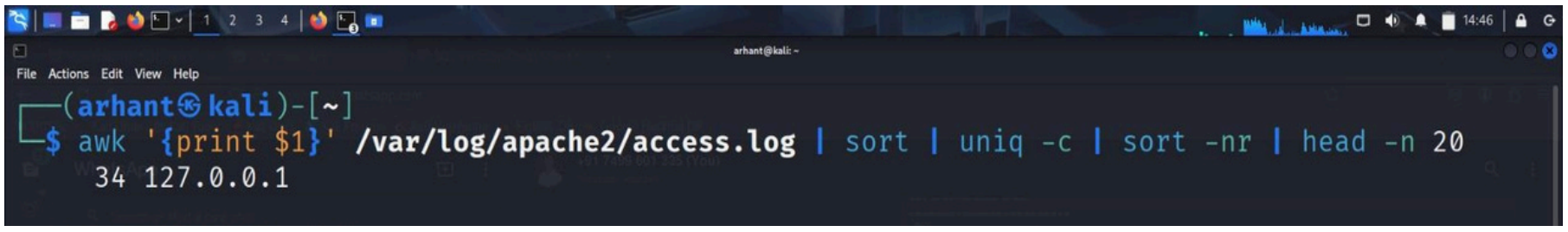
Output :



2. awk '{print $1}' /var/log/apache2/access.log | sort | uniq -c | sort -nr | head -n 20

Purpose :To identify the **most active IP addresses** visiting your Apache server. This is useful for **detecting potential attackers or heavy traffic sources**.

Output :



3.awk '{print $9}' /var/log/apache2/access.log | sort | uniq -c | sort -nr

Purpose :This helps detect errors like **`404 Not Found`** **`Internal Server Error.`** **`500`**

Useful for monitoring server health and spotting potential attacks (e.g., repeated 404s could indicate scanning attempts).

Output :

```
┌──(arhant㉿kali)-[~]
└─$ awk '{print $9}' /var/log/apache2/access.log | sort | uniq -c | sort -nr
     23 200
      5 302
      4 500
      1 404
      1 304
```

4.awk '{print $7}' /var/log/apache2/access.log | sort | uniq -c | sort -nr | head -n 30

Purpose : To identify the **most frequently accessed resources** on your Apache server.

- Helps in detecting **hotspots** (high traffic pages) and potential **attack targets**.
- For example, repeated access to /login.php or /admin.php may indicate **brute-force or scanning attempts**.

Output :

```
┌──(arhant㉿kali)-[~]
└─$ awk '{print $7}' /var/log/apache2/access.log | sort | uniq -c | sort -nr | head -n 30
      4 /dvwa/login.php
      3 /dvwa/vulnerabilities/sqli/
      3 /dvwa/security.php
      2 /dvwa/vulnerabilities/sqli/?id=admin%27+--&Submit=Submit
      2 /dvwa/vulnerabilities/sqli/?id=%27%29+or+%28%271%27%3D%271--&Submit=Submit
      2 /dvwa/favicon.ico
      2 /dvwa/dvwa/js/add_event_listeners.js
      2 /dvwa/dvwa/images/theme-light-dark.png
      2 /dvwa/dvwa/images/logo.png
      2 /dvwa/dvwa/css/main.css
```

5. grep " 404 " /var/log/apache2/access.log | awk '{print $1, $7}' | sort | uniq -c | sort -nr | head

Purpose : To identify which IP addresses are requesting non-existent pages and which resources are most targeted.

- Useful for detecting scanning attempts or misconfigured links.

Output:

```
┌──(arhant㉿kali)-[~]
└─$ grep " 404 " /var/log/apache2/access.log | awk '{print $1, $7}' | sort | uniq -c | sort -nr | head
      1 127.0.0.1 /favicon.ico
```

6.awk -F\" '{print $6}' /var/log/apache2/access.log | sort | uniq -c | sort -nr | head

Purpose : To **identify the most common User-Agents** hitting your server — useful for:

- spotting browsers vs bots/scanners,
- finding suspicious or spoofed agents (e.g., lots of requests with the same uncommon UA),
- understanding client mix (mobile vs desktop),
- filtering/reporting for your assignment.

Output :

```
┌──(arhant㉿kali)-[~]
└─$ awk -F\" '{print $6}' /var/log/apache2/access.log | sort | uniq -c | sort -nr | head
     34 Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101 Firefox/128.0
```

7. sudo tail -n 200 /var/log/apache2/error.log

Purpose: quickly view recent errors, warnings, and server messages to troubleshoot problems (startup issues, misconfigurations, PHP/CGI errors, permission problems, module failures).

Output:

```
┌──(arhant㉿kali)-[~]
└─$ sudo tail -n 200 /var/log/apache2/error.log

[sudo] password for arhant:
[Sat Sep 20 13:26:12.141984 2025] [mpm_prefork:notice] [pid 1010:tid 1010] AH00163: Apache/2.4.65 (Debian) configured -- resuming norm
al operations
[Sat Sep 20 13:26:12.142038 2025] [core:notice] [pid 1010:tid 1010] AH00094: Command line: '/usr/sbin/apache2'
[Sat Sep 20 13:55:45.538864 2025] [mpm_prefork:notice] [pid 903:tid 903] AH00163: Apache/2.4.65 (Debian) configured -- resuming normal
 operations
[Sat Sep 20 13:55:45.540616 2025] [core:notice] [pid 903:tid 903] AH00094: Command line: '/usr/sbin/apache2'
[Sat Sep 20 14:38:04.954943 2025] [php:error] [pid 2131:tid 2131] [client 127.0.0.1:39654] PHP Fatal error:  Uncaught mysqli_sql_excep
tion: You have an error in your SQL syntax; check the manual that corresponds to your MariaDB server version for the right syntax to u
se near ''' at line 1 in /var/www/html/dvwa/vulnerabilities/sqli/source/low.php:11\nStack trace:\n#0 /var/www/html/dvwa/vulnerabilitie
s/sqli/source/low.php(11): mysqli_query()\n#1 /var/www/html/dvwa/vulnerabilities/sqli/index.php(34): require_once('...')\n#2 {main}\n
 thrown in /var/www/html/dvwa/vulnerabilities/sqli/source/low.php on line 11, referer: http://127.0.0.1/dvwa/vulnerabilities/sqli/
[Sat Sep 20 14:38:30.032575 2025] [php:error] [pid 4808:tid 4808] [client 127.0.0.1:33626] PHP Fatal error:  Uncaught mysqli_sql_excep
tion: You have an error in your SQL syntax; check the manual that corresponds to your MariaDB server version for the right syntax to u
se near ') or ('1'='1--'' at line 1 in /var/www/html/dvwa/vulnerabilities/sqli/source/low.php:11\nStack trace:\n#0 /var/www/html/dvwa/
vulnerabilities/sqli/source/low.php(11): mysqli_query()\n#1 /var/www/html/dvwa/vulnerabilities/sqli/index.php(34): require_once('...')
\n#2 {main}\n  thrown in /var/www/html/dvwa/vulnerabilities/sqli/source/low.php on line 11, referer: http://127.0.0.1/dvwa/vulnerabili
ties/sqli/
```

Mitigation Work

1.Block obvious malicious IPs

- **sudo ufw deny from 127.0.0.1**

or

- **sudo iptables -A INPUT -s 127.0.0.1 -j DROP**

2. Install Fail2Ban (prevents brute force)

3. Enable ModSecurity (WAF) for Apache (basic install)

- It monitors HTTP requests/responses. Detects and blocks SQL
- Injection, XSS, LFI/RFI, brute-force, and other attacks. Helps
- harden Apache beyond just logs analysis.