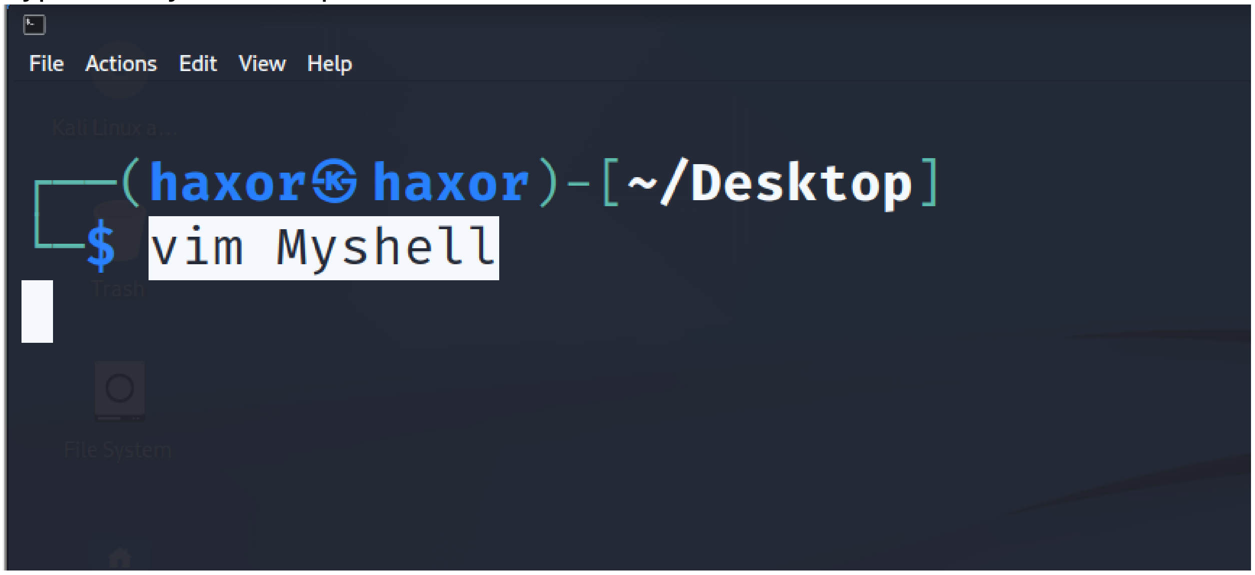
Creating an Executable Script

Time Required: 45 minutes

Objective: Learn to create, save, and run an executable script.

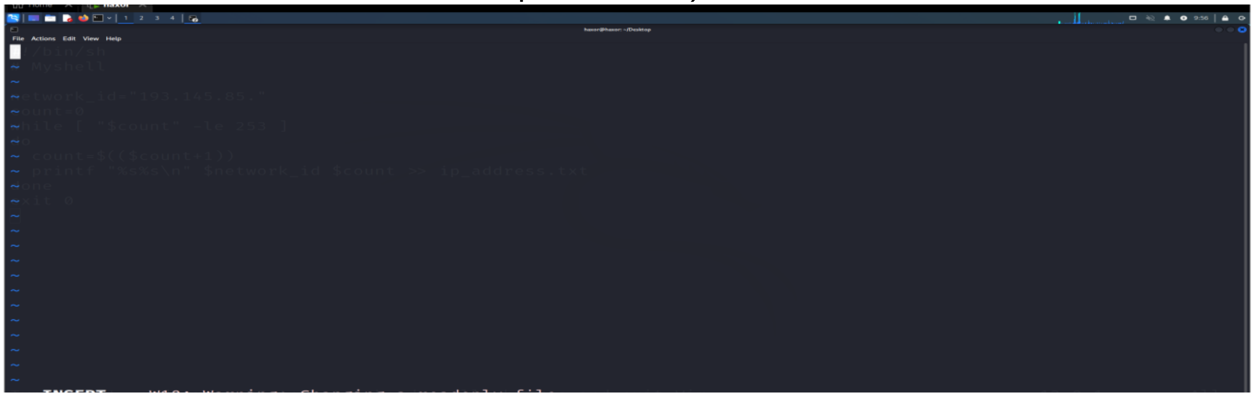
Description: Many hacking tools are written in scripting languages, such as VBScript or JavaScript. In this activity, you create a script that populates a file with a range of IP addresses. This type of file can be used as an input file for Nmap or Fping.

1. If necessary, boot your computer into Linux, and then open a Terminal shell. Type vim Myshell and press Enter.



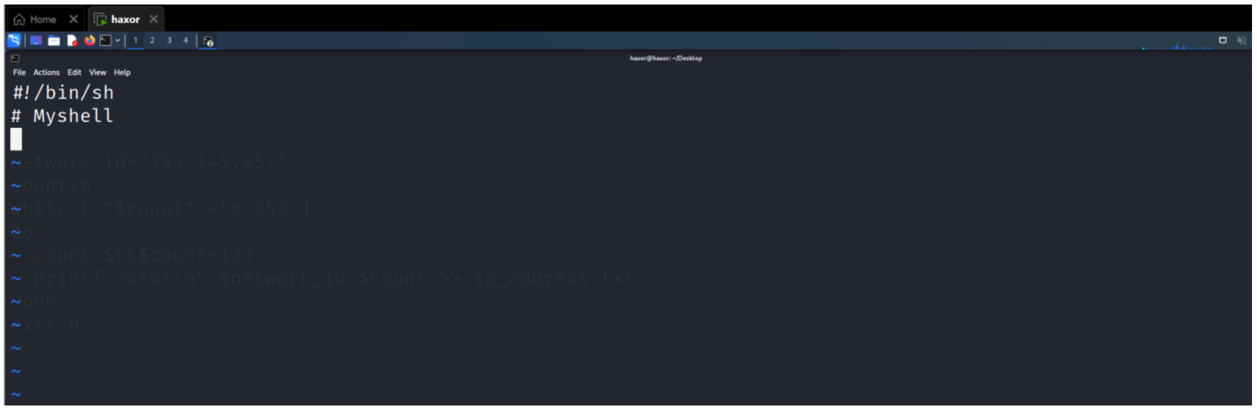
```
┌──(haxor㉿haxor)-[~/Desktop]
└─$ vim Myshell
```

2. Press i to enter insert mode. If this is your first time using the vim editor, use Table 5-1 as a reference. The name “vim” means “vImproved” as it builds on the functionality of the older text editor vi. If you’ve struggled with vi in the past, vim is easier to use. (For a more detailed description of this versatile editor, type man vim in a different Terminal shell and press Enter.)



```
#!/bin/sh
~ Myshell
~
~etwork_id="193.145.85."
~ount=0
~hile [ "$count" -le 253 ]
~o
~ count=$(($count+1))
~ printf "%s%s\n" $network_id $count >> ip_address.txt
~one
~xit 0
~
~
~
~
~
~
~
~
~
~
~
~
~
  INSERT     W10: Warning: Changing a readonly file
```

3. Type #!/bin/sh and press Enter. This line identifies the file you're writing as a script. You should enter a few lines of documentation in any scripts or programs you write because they help later with program modifications and maintenance. When a line is used for documentation purposes, it's preceded with a # character.

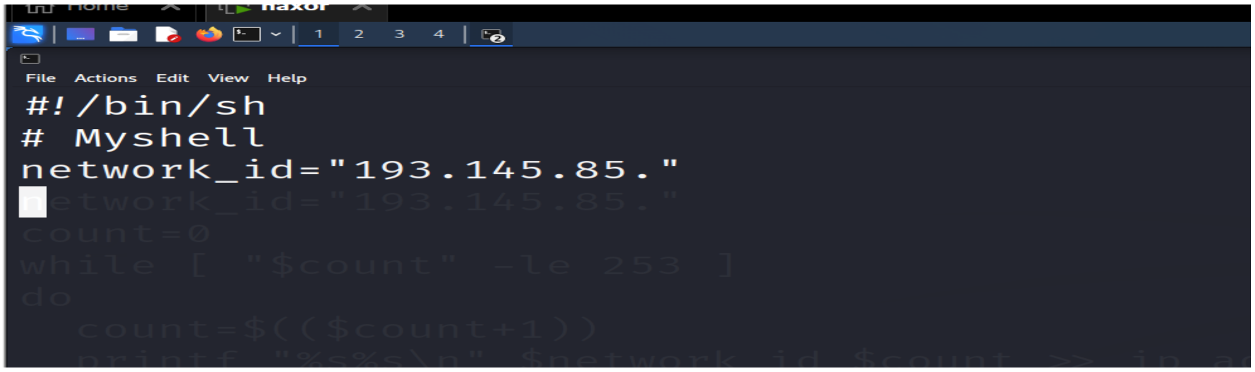


```
#!/bin/sh
# Myshell

~
~
~
~
~
~
~
~
~
~
~
```

4. The second line is the name of the script you're creating. Type # Myshell and press Enter. If this script were used in a production setting, you would also enter the date and your name.

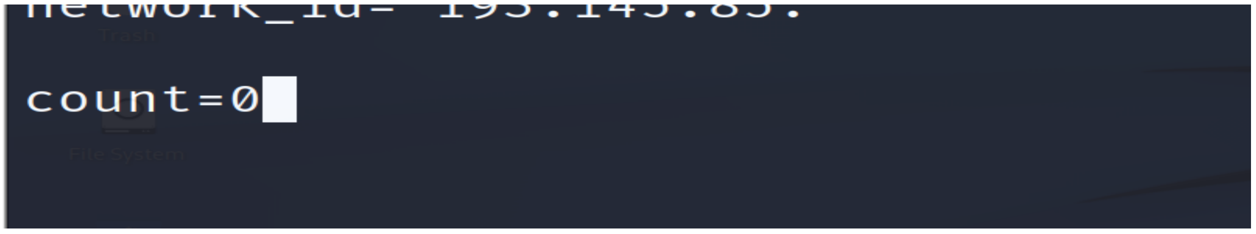
5. Your script should have only #!/bin/sh and # Myshell statements so far.

6. Type network_id="193.145.85." and press Enter. Be sure to include the quotation marks and the period after 85. (Because you aren't using this script on a live network, the address entered in this line doesn't matter.) (SCREENSHOT)



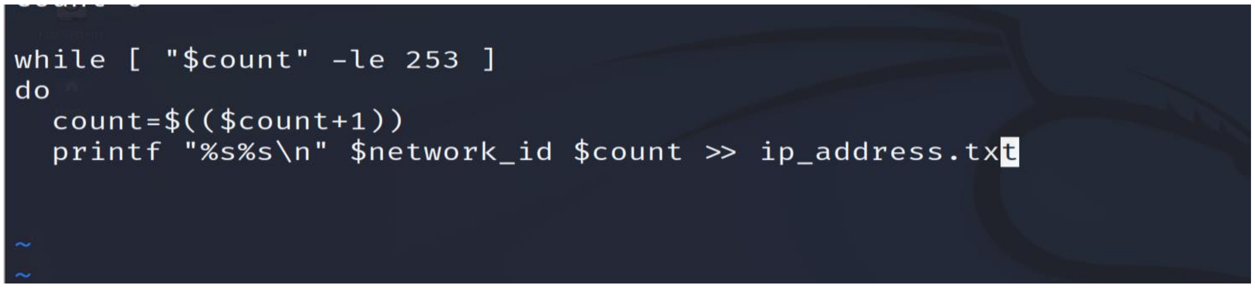
```
#!/bin/sh
# Myshell
network_id="193.145.85."
```

1. Type count=0 and press Enter. This command initializes the count variable to zero, which is always wise because a variable shouldn't be used in a program without having a value set.



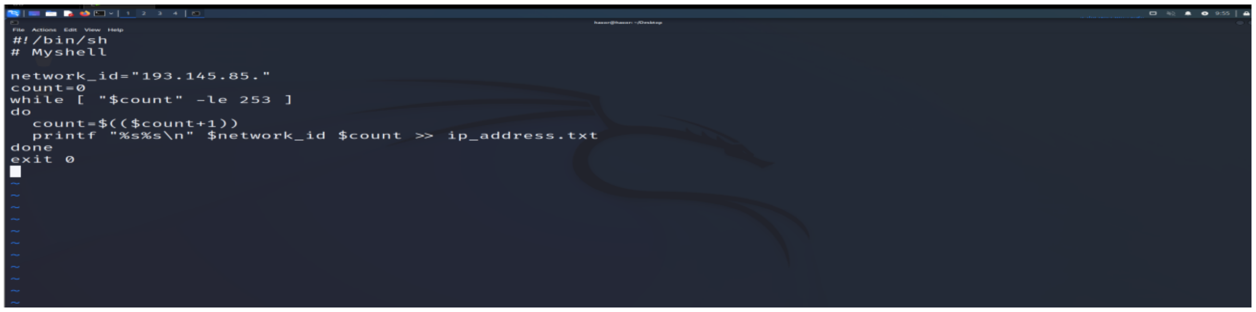
```
network_id= 193.145.85.

count=0
```

You need your script to add the number 1 to the 193.145.85. network ID and continue incrementing and adding numbers to the network ID until the IP address range 193.145.85.1 to 193.145.85.254 is written to a file named ip_address.txt. In programming, this repeated process is called looping. To avoid creating an endless loop, you need to add a condition in a while statement.



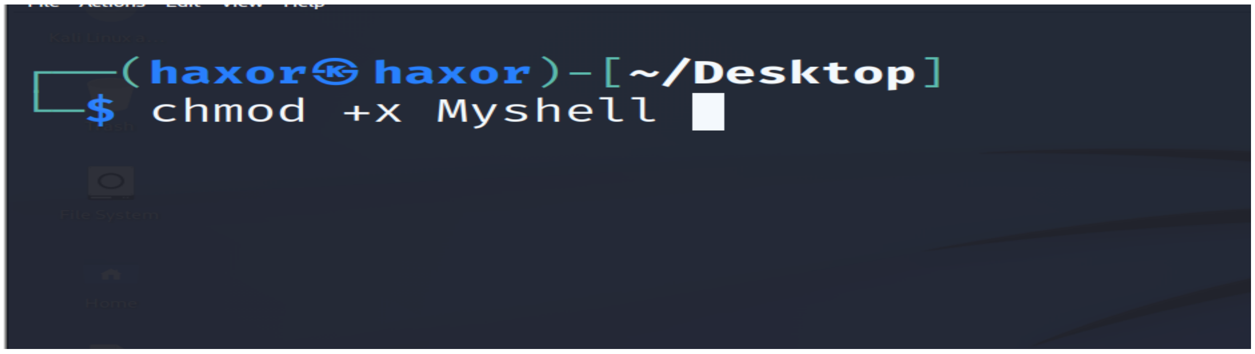
```
while [ "$count" -le 253 ]
do
  count=$(($count+1))
  printf "%s%s\n" $network_id $count >> ip_address.txt
```

2. Type while [“$count” -le 253] and press Enter. Note the spaces inside the square brackets and pay close attention to the use of quotation marks and dollar signs.



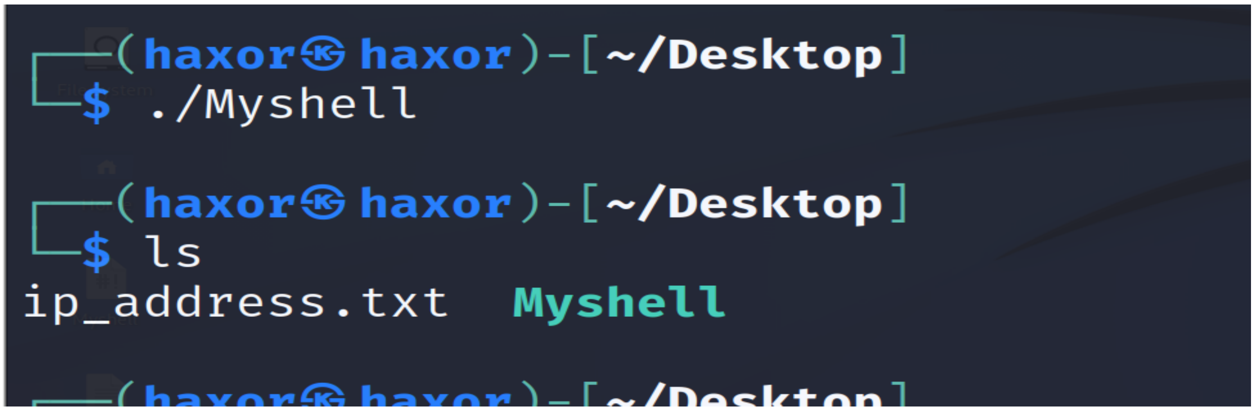
```
#!/bin/sh
# Myshell

network_id="193.145.85."
count=0
while [ "$count" -le 253 ]
do
  count=$(($count+1))
  printf "%s%s\n" $network_id $count >> ip_address.txt
done
exit 0
```

12. Now that you’ve saved your script, you need to make it executable so that you can run it. At the command prompt, type chmod +x Myshell and press Enter.



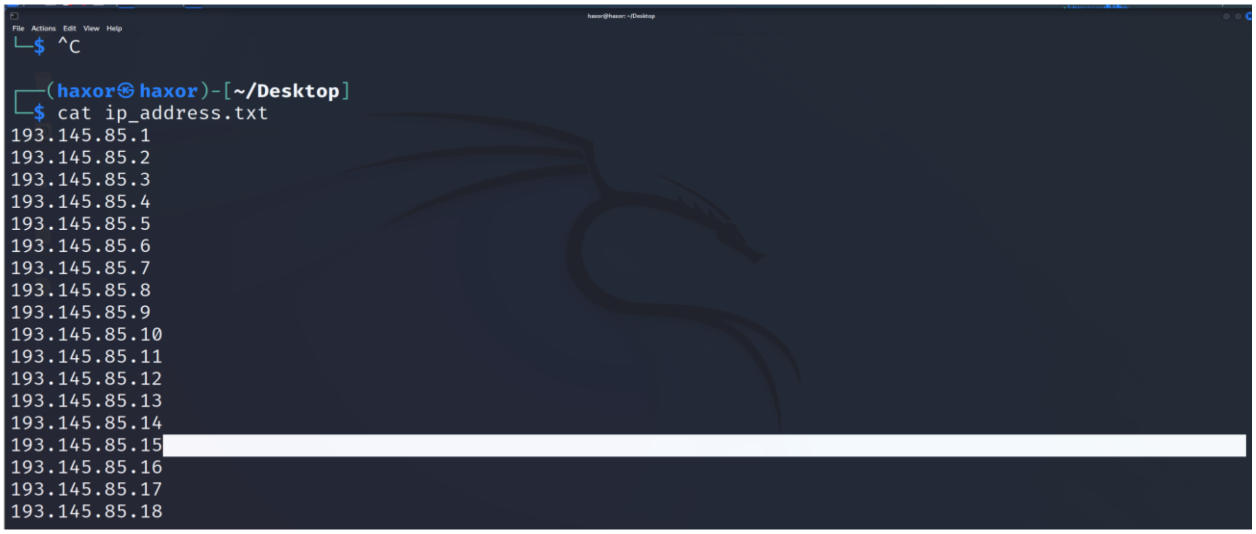
```
┌──(haxor㉿haxor)-[~/Desktop]
└─$ chmod +x Myshell
```

13. To run your script, type ./Myshell and press Enter. Because your script doesn't create any screen output, you need to examine the contents of the ip_address.txt file to see whether the script worked.



```
┌──(haxor㉿haxor)-[~/Desktop]
└─$ ./Myshell

┌──(haxor㉿haxor)-[~/Desktop]
└─$ ls
ip_address.txt   Myshell
```

14. Type cat ip_address.txt and press Enter. How many IP addresses were created in the ip_address.txt file?



```
└─$ ^C

┌──(haxor㉿haxor)-[~/Desktop]
└─$ cat ip_address.txt
193.145.85.1
193.145.85.2
193.145.85.3
193.145.85.4
193.145.85.5
193.145.85.6
193.145.85.7
193.145.85.8
193.145.85.9
193.145.85.10
193.145.85.11
193.145.85.12
193.145.85.13
193.145.85.14
193.145.85.15
193.145.85.16
193.145.85.17
193.145.85.18
```

15. Close the shell.

<u>Conclusion:</u>

In this lab, we delved into network scanning and packet crafting using tools like Nmap, Fping, and Hping3 within a Linux environment. Through hands-on exploration, we learned Nmap commands for SYN scans, script scanning, and port targeting. Additionally, we experimented with crafting different IP packets and creating executable scripts for automating tasks.

<u>References</u>:

1. Mazurczyk, W., & Caviglione, L. (2021). Cyber reconnaissance techniques. Communications of the ACM, 64(3), 86-95.

2. Calderon, P. (2021). Nmap Network Exploration and Security Auditing Cookbook: Network discovery and security scanning at your fingertips. Packt Publishing Ltd.
3. Ndatinya, V., Xiao, Z., Manepalli, V. R., Meng, K., & Xiao, Y. (2015). Network forensics analysis using Wireshark. International Journal of Security and Networks, 10(2), 91-106.