HO#2.4
Scanning & Vulnerability Analysis: Part 2

Phase 1- Reconnaissance and Information Gathering

Dear students we have covered the Information gathering phase (reconnaissance) in HO#2.2, that involves collecting as much public information as possible about the organization, systems, networks, applications, and employees to identify potential vulnerabilities and formulate a strategy for further testing. Passive information gathering (reconnaissance) involves collecting data without directly interacting with the target system, reducing the risk of detection. Gathering information from publicly available sources like news outlets, blogs and social media platforms (Twitter, Facebook, LinkedIn) is named as Open-Source Intelligence (OSINT). The techniques used for OSINT are Web Scraping, Google Dorking, and social media profiling. The tools that we have used for this in HO#2.2 were `host`, `nslookup`, `dig`, `whois`, `knockpy`, `netdiscover`, `traceroute`, `whatweb`, `theHarvester`, `sherlock`, `wfw00f`, `Google Dorking`, `OSINT framework`.

Phase 2- Scanning and Vulnerability Analysis

This handout is continuation of our previous Handout#2.3. *The **objective of scanning** is to identify system services and potential entry points in a network by performing NW scanning, port scanning and Services detection, using tools like* `nmap`, `zenmap`, `unicorn`, `nikto` *and so on.* ***The objective of vulnerability analysis*** *is to dig deeper and perform an in-depth examination to uncover known vulnerabilities and weaknesses in the systems, applications, and their configurations.* To perform vulnerability analysis, we have already covered the use of tools like `nessus`, `searchsploit` and `OpenVAS` in Handout2.3. Today we will perform **Scanning and Vulnerability Analysis using another famous tool called Metasploit Framework.**

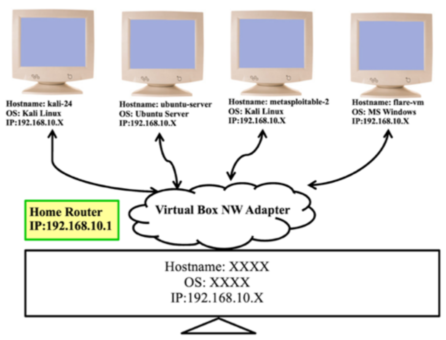
- Scanning and Vulnerability Analysis Tools:
  - OS and NW: `nmap`, `nessus`, `openVAS`, `tripwire`, `wireshark`, `MSF`
  - Web Applications: `Burp Suite`, `nikto`, `OWASP ZAP`,
  - Mobile Applications: `frida`, `drozer`, `MobSF`, `Burp Suite`,

- The steps that we normally perform during vulnerability analysis are:
  - Scanning the target for known vulnerabilities using databases like NVD.
  - Assessing the severity of discovered vulnerabilities using metrics like CVSS/VPR.
  - Submit report highlighting the vulnerability, risk levels, and mitigation steps.
- A must read: https://www.rapid7.com/fundamentals/vulnerability-management-and-scanning/

Environment Setup

You can use the following machines for a hands-on practice of this handout in which I am using kali Linux as attacker machine and scanning Metasploitable 2:

1. Kali Linux (IP: x.x.x.x)
2. Metasploitable 2 (IP: x.x.x.x)



Overview of Metasploit Framework

The Metasploit Framework, MSF (https://www.metasploit.com/) is a widely used open-source penetration testing and exploitation platform developed by H.D. Moore in 2003 in Perl programming language. Later in 2007, the framework was rewritten in Ruby. In 2009, it was acquired by Rapid7 (https://www.rapid7.com/). *MSF provides security professionals and researchers with a comprehensive set of tools that can support all phases of a penetration testing engagement, from information gathering to post-exploitation.* Kali Linux comes pre-installed with a command line version of Metasploit framework that is free. The commercial version that facilitates the automation and management of tasks also has a graphical user interface.

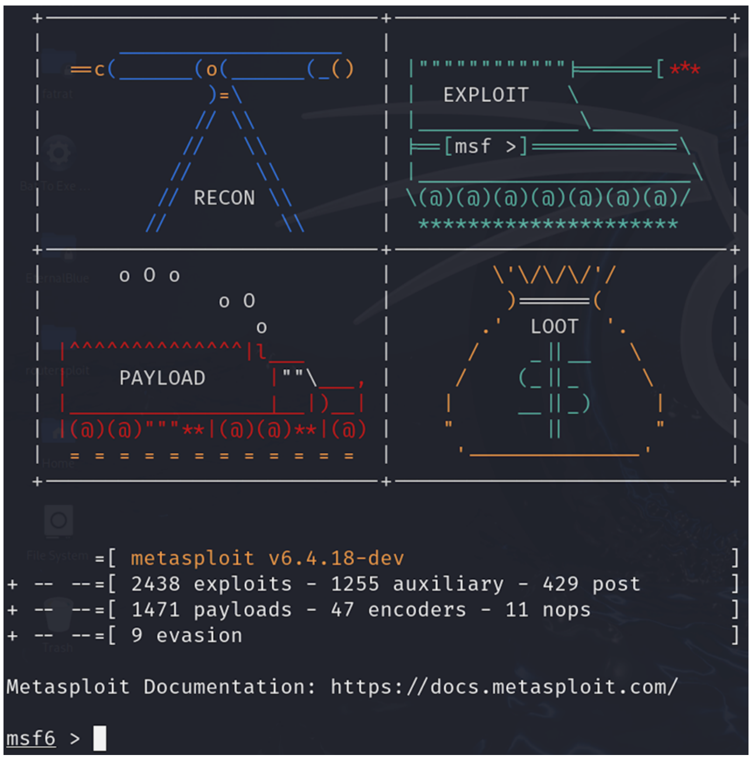
Before we start, let us once again have a clear idea about *vulnerability*, *exploit* and *payload*. Consider a locked refrigerator containing chocolates, fruit trifle, cold drinks etc. Somehow you come to know about its ***vulnerability*** that it can be unlocked using a CD70 key. You ***exploit*** that vulnerability and opens/unlock the refrigerator. Now the ***payload*** is the piece of program that performs the actual task once the vulnerability is exploited, i.e., eating/stealing the chocolates ☺

Accessing Metasploit Framework using msfconsole

There are many interfaces to Metasploit Framework (MSF), e.g., **`msfconsole`**, `msfcli`, `msfgui`, `msfweb`, `armitage`. The one we will be using is `msfconsole` which is probably the most popular interface to MSF. It provides an "all-in-one" centralized console and allows you efficient access to virtually all of the options available in the MSF. The `msfconsole` may appear intimidating at first, but once you learn the syntax of its commands you will learn to appreciate the power of utilizing this interface.
Once you will run `msfconsole` on your Kali Linux machine you will get a screenshot similar to the one given below, that displays some important information about Metasploit Framework:

```
$ sudo msfconsole
```

- ASCII logo
- Metasploit version (6.4.50),
    - 1283 `auxiliary`
    - 2497 `exploits`
    - 1610 `payloads`
    - 49 `encoders`
    - 13 `nops`
    - 9 `evasion`
    - 431 `post`



```
       =[ metasploit v6.4.18-dev                          ]
+ -- --=[ 2438 exploits - 1255 auxiliary - 429 post       ]
+ -- --=[ 1471 payloads - 47 encoders - 11 nops           ]
+ -- --=[ 9 evasion                                       ]

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

msf6 >
```

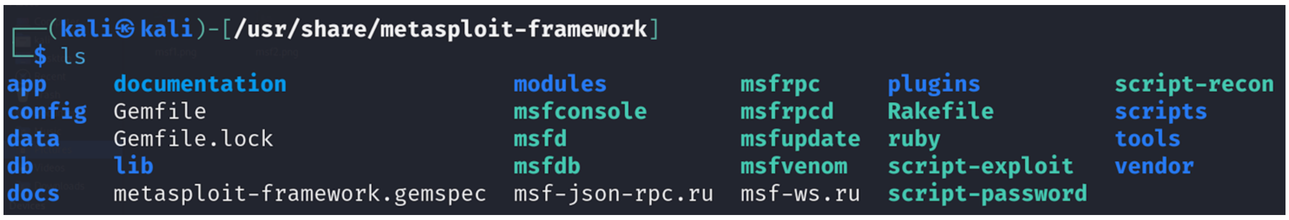
```
msf6> help
```

- Once `msfconsole` is running, you can use it's `help` command to check out the details about different commands.
- Making yourself familiar with these commands will help you throughout this course and will give you a strong foundation for working with Metasploit in general. Better to run `msfconsole` command as `sudo`, as you may need that power later. Good Luck ☺

Anatomy and Structure of Metasploit

In Kali Linux, files related to Metasploit Framework are in `/usr/share/metasploit-framework/` directory. Before moving ahead, one must explore the contents of this directory:

```
$ ls /usr/share/metasploit-framework
```



```
┌──(kali㉿kali)-[/usr/share/metasploit-framework]
└─$ ls
app     documentation                  modules          msfrpc     plugins          script-recon
config  Gemfile                        msfconsole       msfrpcd    Rakefile         scripts
data    Gemfile.lock                   msfd             msfupdate  ruby             tools
db      lib                            msfdb            msfvenom   script-exploit   vendor
docs    metasploit-framework.gemspec   msf-json-rpc.ru  msf-ws.ru  script-password
```

Almost all of your interaction with Metasploit will be through one of its seven *modules*, located under **`/usr/share/metasploit-framework/modules/`** directory. These modules are scripts in Ruby that interface with Metasploit itself to perform some specific tasks. Here is a brief description of each of these directories.

1. **Auxiliary:** *The **`auxiliary`** sub-directory contains scripts designed to perform information gathering and vulnerability analysis* using port scanners (used to identify known vulnerabilities and gather information), sniffers (used to capture and analyze NW packets), and fuzzers (used to discover unknown vulnerabilities by stress testing systems with malformed data). For example, inside the auxiliary sub-directory, you have different sub-directories, like the `scanner/portscan/` sub-directory having `syn.rb` and `tcp.rb` scripts which are used to perform a syn-scan and tcp-full-scan respectively.
2. **Exploits:** *The **`exploits`** sub-directory contains scripts designed to exploit specific vulnerabilities in operating systems, network services, applications, and so on for different OSs like unix, linux, windows, solaris etc.* These are used to gain unauthorized access to a system by exploiting known vulnerabilities.
3. **Payloads:** *The **`payloads`** sub-directory contains various payloads that runs remotely on the compromised system.* There exist three sub-directories: The `singles` sub-directory contains self-contained payloads that perform a single task, e.g., executes a specified command on the target or download a file. The `stagers` sub-directory contains small payloads that when delivered to the target, establishes a connection back to the attacker's machine. The `stages` sub-directory contains larger payloads that are then sent over this connection.
4. **Encoders:** *The **`encoders`** sub-directory contains scripts used to encode and obfuscate payloads to evade detection* by security solutions like antivirus software for x86, x64, sparc and so on.
5. **Nops:** *The **`nops`** sub-directory contains scripts that generate No Operation (NOP) sleds, to modify payload signature and thereby avoiding detection.*
6. **Evasion:** *The **`evasion`** sub-directory contains scripts that are designed to evade detection by security mechanisms like firewalls and intrusion detection systems (IDS).* These are used to stealthily bypass security defenses during an attack.
7. **Post:** The **`post`** sub-directory contains scripts for post-exploitation activities that can be executed on compromised systems. *The tasks performed can be privilege escalation, data exfiltration, and maintaining persistence.*

Basic Commands of `msfconsole`

I recommend to run `msfconsole` command as `sudo`, as you may need that power later. Since the `msfconsole` utility provides a whole new environment, so students are advised to familiarize themselves with commands of this powerful tool. We will be extensively using these commands in this module, a summary of which are shown in the table below. Remember experimentation is the key to successful learning, so Good Luck ☺

Commands	**Description**

`msf6 > help`
 	
`msf6 > help <command>`
	The simple **help** command will give you a list and small description of all available commands divided into different categories like core commands, module commands, job commands, resource script commands, database backend commands, and so on

`msf6 > banner`	Print a stunning ASCII art banner along with version information and module counts
`msf6 > exit/quit`	The **exit** or **quit** command will simply exit `msfconsole` utility
`msf6 > show nops`	The **show** command is passed one argument that can be a module name and it displays scripts names, disclosure date, rank, check, and description of each

`msf6 > search telnet`
`msf6 > search type:auxiliary telnet`
`msf6 > search type:exploit telnet`
`msf6 > search cve:2017-0144`
	The **search** command is used to search for exploits, payloads, auxiliary module. Use `search` if you are looking for modules that are ready to be used within MSF.

`msf6 > searchsploit telnet`
`msf6 > searchsploit eternalblue`
	The **searchsploit** is a stand-alone command line tool used to search EDB for publicly available exploits, shellcodes, and vulnerabilities. Use `search` when you want to find exploits from the EDB that are not yet integrated into MSF.

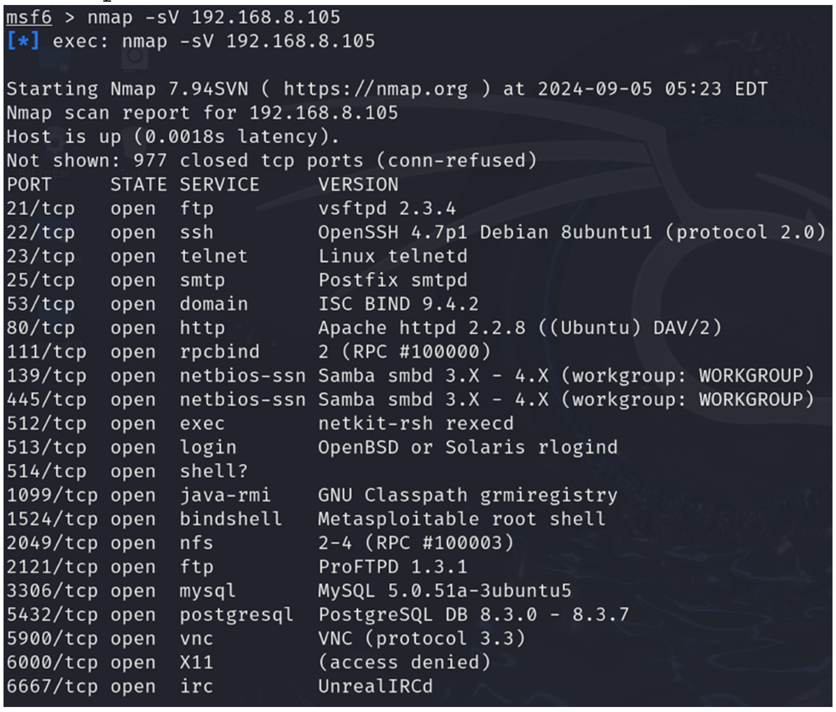
`msf6 >info auxiliary/scanner/portscan/syn`	Once you have identified the module you are interested in using, you can use the **info** command to find out more about it

`msf6>use auxiliary/scanner/portscan/syn`
 	
`msf6 auxiliary(scanner/portscan/syn)>`
	Once you are done with searching a specific module, then you give **use** command followed by the specific scanner/exploit/payload to change your context to that specific module, thus exposing type-specific commands. Once you are finished working with a specific module, you can issue the **back** command to move out of the current context.

`msf6 auxiliary(scanner/portscan/syn)> show options`	Each module has a list of parameters or options, you need to configure. So, once you are in the context of a particular module, you can issue the **show options** command to display which settings are available and/or required for that specific module
`msf6 auxiliary(scanner/portscan/syn)> show advanced`	To view any advanced options that may be available for a given module, you can use the **show advanced** command
`msf6 auxiliary(scanner/portscan/syn)> set <param> <value>`	Before you can use a module to scan or exploit a target it needs to be configured for your specific use case. You can use the **set** command to update the value of a parameter
`msf6 auxiliary(scanner/portscan/syn)> unset <param>`	The **unset** command is opposite of the set command, which removes a parameter previously configured with set command. You can remove all assigned variables with unset all command
`msf6 auxiliary(scanner/portscan/syn)> setg RHOSTS <ip>`	You'll notice that some parameters, such as RHOSTS appear over and over again across multiple modules. Rather than repeatedly entering the RHOSTS value for each new module we load, we can use the **setg** command to set the value of that parameter for all modules
`msf6 auxiliary(scanner/portscan/syn)> run`	Once you have configured all parameters marked as required for the module you have loaded, you can execute it using the **run** or **exploit** command

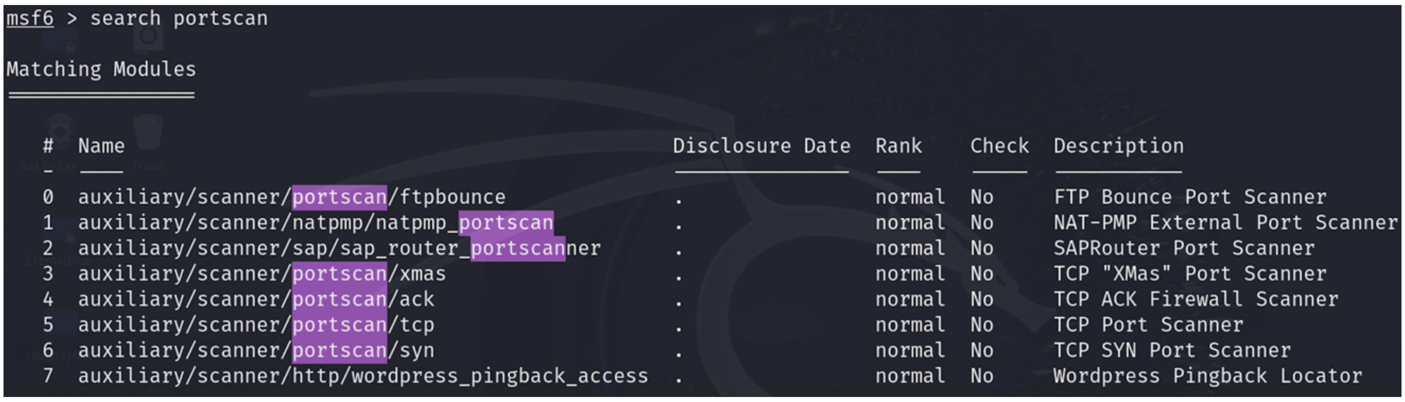
Performing Port Scanning on Metasploitable2

- Inside MSF console, we can run the `nmap` command to perform a port scan, as we have done many a times in our Handout 2.3
  msf6> `nmap -sV <ip of M2>`



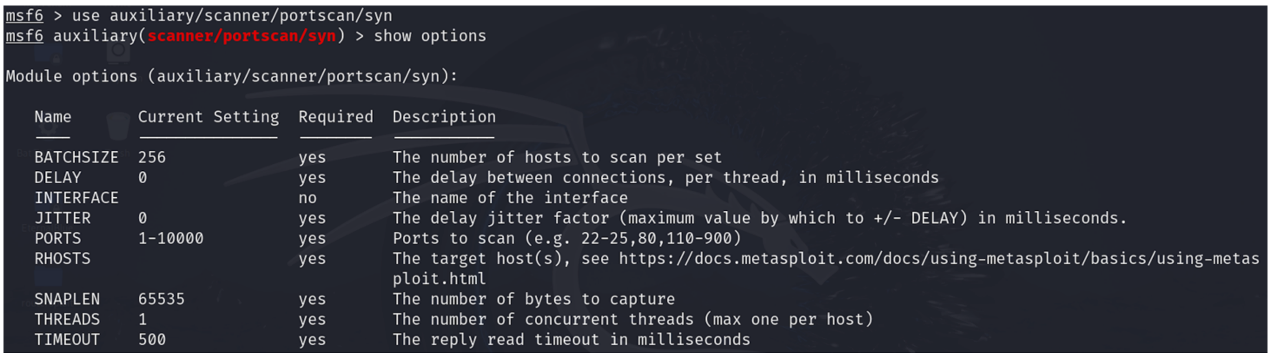
```
msf6 > nmap -sV 192.168.8.105
[*] exec: nmap -sV 192.168.8.105

Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-09-05 05:23 EDT
Nmap scan report for 192.168.8.105
Host is up (0.0018s latency).
Not shown: 977 closed tcp ports (conn-refused)
PORT     STATE SERVICE     VERSION
21/tcp   open  ftp         vsftpd 2.3.4
22/tcp   open  ssh         OpenSSH 4.7p1 Debian 8ubuntu1 (protocol 2.0)
23/tcp   open  telnet      Linux telnetd
25/tcp   open  smtp        Postfix smtpd
53/tcp   open  domain      ISC BIND 9.4.2
80/tcp   open  http        Apache httpd 2.2.8 ((Ubuntu) DAV/2)
111/tcp  open  rpcbind     2 (RPC #100000)
139/tcp  open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
445/tcp  open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
512/tcp  open  exec        netkit-rsh rexecd
513/tcp  open  login       OpenBSD or Solaris rlogind
514/tcp  open  shell?
1099/tcp open  java-rmi    GNU Classpath grmiregistry
1524/tcp open  bindshell   Metasploitable root shell
2049/tcp open  nfs         2-4 (RPC #100003)
2121/tcp open  ftp         ProFTPD 1.3.1
3306/tcp open  mysql       MySQL 5.0.51a-3ubuntu5
5432/tcp open  postgresql  PostgreSQL DB 8.3.0 - 8.3.7
5900/tcp open  vnc         VNC (protocol 3.3)
6000/tcp open  X11         (access denied)
6667/tcp open  irc         UnrealIRCd
```

- There are many scanners inside Metasploit, that help us in information gathering inside the auxiliary module. There are a variety of port scanners that are available to us within `/usr/share/metasploit-framework/modules/auxiliary/scanner/portscan` directory like `syn.rb`, `tcp.rb`, `ack.rb` and so on.
  msf6> `search portscan`



```
msf6 > search portscan

Matching Modules
================

   #  Name                                         Disclosure Date  Rank    Check  Description
   -  ----                                         ---------------  ----    -----  -----------
   0  auxiliary/scanner/portscan/ftpbounce         .                normal  No     FTP Bounce Port Scanner
   1  auxiliary/scanner/natpmp/natpmp_portscan     .                normal  No     NAT-PMP External Port Scanner
   2  auxiliary/scanner/sap/sap_router_portscanner .                normal  No     SAPRouter Port Scanner
   3  auxiliary/scanner/portscan/xmas              .                normal  No     TCP "XMas" Port Scanner
   4  auxiliary/scanner/portscan/ack               .                normal  No     TCP ACK Firewall Scanner
   5  auxiliary/scanner/portscan/tcp               .                normal  No     TCP Port Scanner
   6  auxiliary/scanner/portscan/syn               .                normal  No     TCP SYN Port Scanner
   7  auxiliary/scanner/http/wordpress_pingback_access .            normal  No     Wordpress Pingback Locator
```

- The above screenshot shows different types of scans that we can perform on a network or on a specific machine. Let us perform the **syn** scan
  msf6> `use auxiliary/scanner/portscan/syn`
  msf6 auxiliary(scanner/portscan/syn)> `show options`

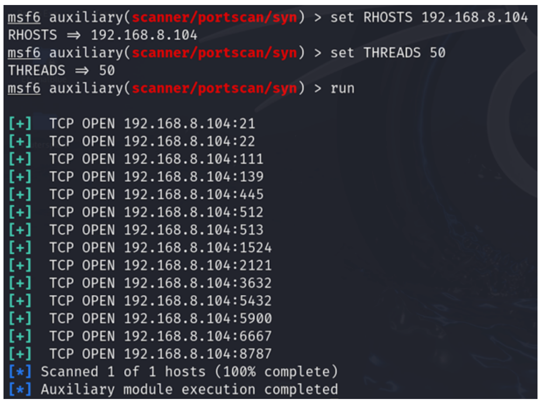


```
msf6 > use auxiliary/scanner/portscan/syn
msf6 auxiliary(scanner/portscan/syn) > show options

Module options (auxiliary/scanner/portscan/syn):

   Name       Current Setting  Required  Description
   ----       ---------------  --------  -----------
   BATCHSIZE  256              yes       The number of hosts to scan per set
   DELAY      0                yes       The delay between connections, per thread, in milliseconds
   INTERFACE                   no        The name of the interface
   JITTER     0                yes       The delay jitter factor (maximum value by which to +/- DELAY) in milliseconds.
   PORTS      1-10000          yes       Ports to scan (e.g. 22-25,80,110-900)
   RHOSTS                      yes       The target host(s), see https://docs.metasploit.com/docs/using-metasploit/basics/using-metas
                                         ploit.html
   SNAPLEN    65535            yes       The number of bytes to capture
   THREADS    1                yes       The number of concurrent threads (max one per host)
   TIMEOUT    500              yes       The reply read timeout in milliseconds
```

- Now you need to set at least the RHOSTS parameter, you may change others as well and run

```
msf6 auxiliary(scanner/portscan/syn)> set RHOSTS <IP of M2>
msf6 auxiliary(scanner/portscan/syn)> set THREADS 50
msf6 auxiliary(scanner/portscan/syn)> run
```

- **Note:** Since we have run it only on the IP of the Metasploitable2 machine, so it has displayed the different open ports on that machine, you may run it on an entire network to get a more detailed view. However, that may take a bit of time.



```
msf6 auxiliary(scanner/portscan/syn) > set RHOSTS 192.168.8.104
RHOSTS => 192.168.8.104
msf6 auxiliary(scanner/portscan/syn) > set THREADS 50
THREADS => 50
msf6 auxiliary(scanner/portscan/syn) > run

[+]  TCP OPEN 192.168.8.104:21
[+]  TCP OPEN 192.168.8.104:22
[+]  TCP OPEN 192.168.8.104:111
[+]  TCP OPEN 192.168.8.104:139
[+]  TCP OPEN 192.168.8.104:445
[+]  TCP OPEN 192.168.8.104:512
[+]  TCP OPEN 192.168.8.104:513
[+]  TCP OPEN 192.168.8.104:1524
[+]  TCP OPEN 192.168.8.104:2121
[+]  TCP OPEN 192.168.8.104:3632
[+]  TCP OPEN 192.168.8.104:5432
[+]  TCP OPEN 192.168.8.104:5900
[+]  TCP OPEN 192.168.8.104:6667
[+]  TCP OPEN 192.168.8.104:8787
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
```

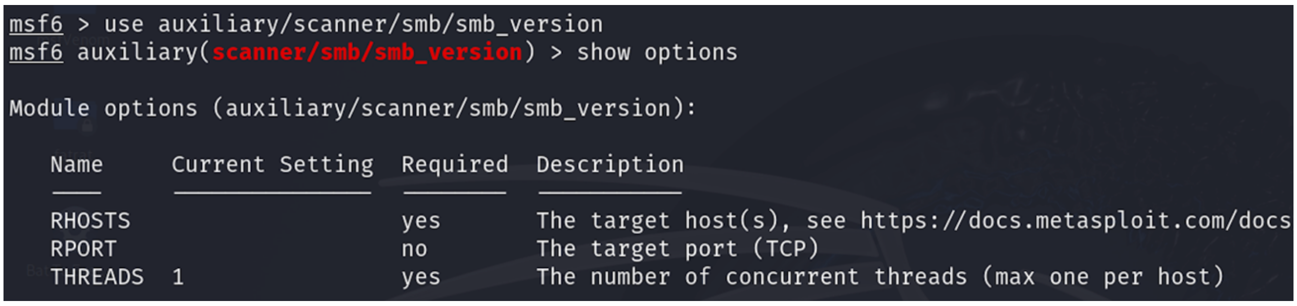
To Do: We have successfully run the **`syn.rb`** script to find out the open ports on Mr. Students are advised to run other scripts like **`ack.rb`** and **`tcp.rb`** at their own and compare the difference between the outputs of these three scans ☺

Performing Version Scanning on Metasploitable2

Now that we have determined about the different ports that are open on Metasploitable2, let us now perform scans to determine the version of services running on different ports:

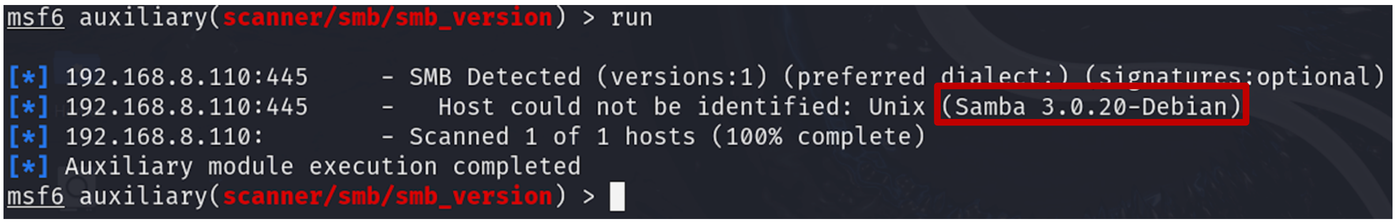
- **`smb/smb_version:`** SMB (Server Message Block) is an application layer protocol that is mainly used for providing shared access to things like files and printers on the network. Remember when we ran different port scanning scripts on Metasploitable2, we came to know that SMB service is running on port 139 and 445, but it did not tell us the version. Let us use `smb_version.rb` script to check out the version of SMB service running on M2.

```
msf6> use auxiliary/scanner/smb/smb_version
msf6 auxiliary(scanner/smb/smb_version)> show options
```



```
msf6 > use auxiliary/scanner/smb/smb_version
msf6 auxiliary(scanner/smb/smb_version) > show options

Module options (auxiliary/scanner/smb/smb_version):

   Name     Current Setting  Required  Description
   ----     ---------------  --------  -----------
   RHOSTS                    yes       The target host(s), see https://docs.metasploit.com/docs
   RPORT                     no        The target port (TCP)
   THREADS  1                yes       The number of concurrent threads (max one per host)
```

```
msf6 auxiliary(scanner/smb/smb_version)> set RHOSTS <IP of M2>
msf6 auxiliary(scanner/smb/smb_version)> run
```



```
msf6 auxiliary(scanner/smb/smb_version) > run

[*] 192.168.8.110:445     - SMB Detected (versions:1) (preferred dialect:) (signatures:optional)
[*] 192.168.8.110:445     -   Host could not be identified: Unix (Samba 3.0.20-Debian)
[*] 192.168.8.110:        - Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf6 auxiliary(scanner/smb/smb_version) >
```

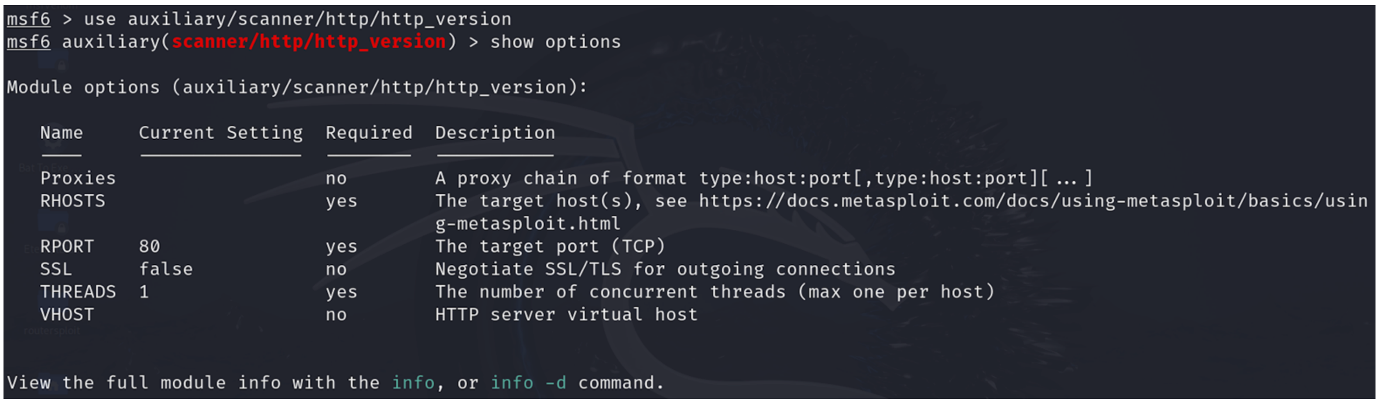
- **`ftp/ftp_version:`** The File Transfer Protocol (FTP) service is a standard network protocol used for transferring files between a client and a server over a TCP/IP network allowing users to upload, download, and manage files on remote systems. FTP typically operates on ports 21 (for command/control) and 20 (for data transfer) and supports authentication via usernames and passwords. When we ran the port scanning scripts on Metasploitable2, we came to know that ftp service is running on port 21, but it did not tell us its version. Let us use `ftp_version.rb` to check out the version of ftp running on M2

```
msf6> use auxiliary/scanner/ftp/ftp_version
msf6 auxiliary(scanner/ftp/ftp_version)> show options
msf6 auxiliary(scanner/smb/smb_version)> set RHOSTS <IP of M2>
msf6 auxiliary(scanner/smb/smb_version)> run
```

The output of this scan tells us that version of ftp running on M2 is `vsftpd 2.3.4`

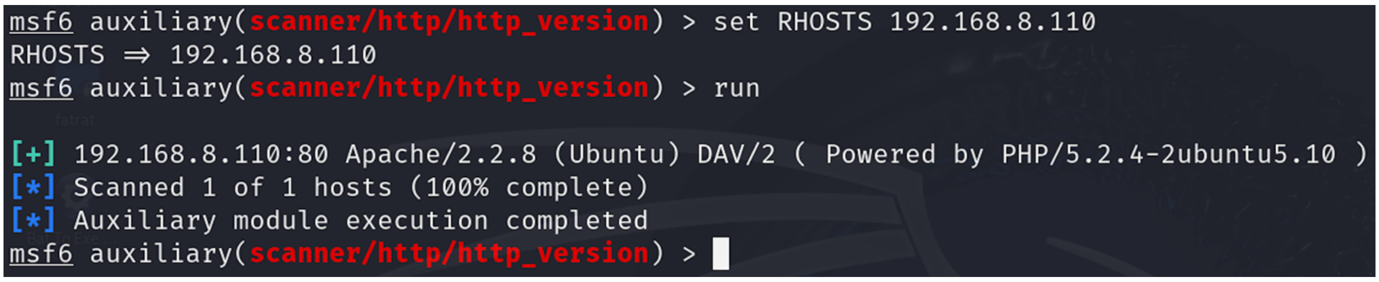
- **`http/http_version:`** Let us now checkout the version of the web server running on M2 at port 80

```
msf6> use auxiliary/scanner/http/http_version
msf6 auxiliary(scanner/http/http_version)> show options
```



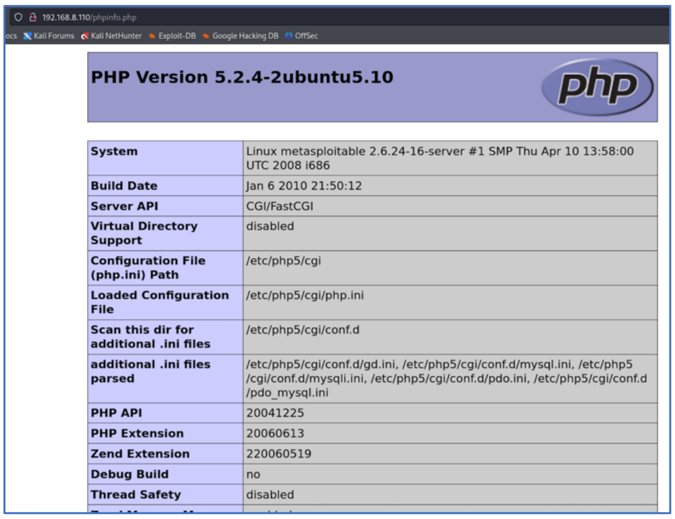
```
msf6 > use auxiliary/scanner/http/http_version
msf6 auxiliary(scanner/http/http_version) > show options

Module options (auxiliary/scanner/http/http_version):

   Name     Current Setting  Required  Description
   ----     ---------------  --------  -----------
   Proxies                   no        A proxy chain of format type:host:port[,type:host:port][ ... ]
   RHOSTS                    yes       The target host(s), see https://docs.metasploit.com/docs/using-metasploit/basics/usin
                                       g-metasploit.html
   RPORT    80               yes       The target port (TCP)
   SSL      false            no        Negotiate SSL/TLS for outgoing connections
   THREADS  1                yes       The number of concurrent threads (max one per host)
   VHOST                     no        HTTP server virtual host


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

- We just need to set the RHOSTS parameter and run the scan.



```
msf6 auxiliary(scanner/http/http_version) > set RHOSTS 192.168.8.110
RHOSTS ⇒ 192.168.8.110
msf6 auxiliary(scanner/http/http_version) > run

[+] 192.168.8.110:80 Apache/2.2.8 (Ubuntu) DAV/2 ( Powered by PHP/5.2.4-2ubuntu5.10 )
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf6 auxiliary(scanner/http/http_version) >
```

- It's Apache 2.2.8 with PHP 5.2.4. In a browser on our Kali machine, we can navigate to http://192.168.231.109/phpinfo.php and confirm the information, which is shown in the screenshot below ☺



192.168.8.110/phpinfo.php

PHP Version 5.2.4-2ubuntu5.10

System	Linux metasploitable 2.6.24-16-server #1 SMP Thu Apr 10 13:58:00 UTC 2008 i686
Build Date	Jan 6 2010 21:50:12
Server API	CGI/FastCGI
Virtual Directory Support	disabled
Configuration File (php.ini) Path	/etc/php5/cgi
Loaded Configuration File	/etc/php5/cgi/php.ini
Scan this dir for additional .ini files	/etc/php5/cgi/conf.d
additional .ini files parsed	/etc/php5/cgi/conf.d/gd.ini, /etc/php5/cgi/conf.d/mysql.ini, /etc/php5/cgi/conf.d/mysqli.ini, /etc/php5/cgi/conf.d/pdo.ini, /etc/php5/cgi/conf.d/pdo_mysql.ini
PHP API	20041225
PHP Extension	20060613
Zend Extension	220060519
Debug Build	no
Thread Safety	disabled

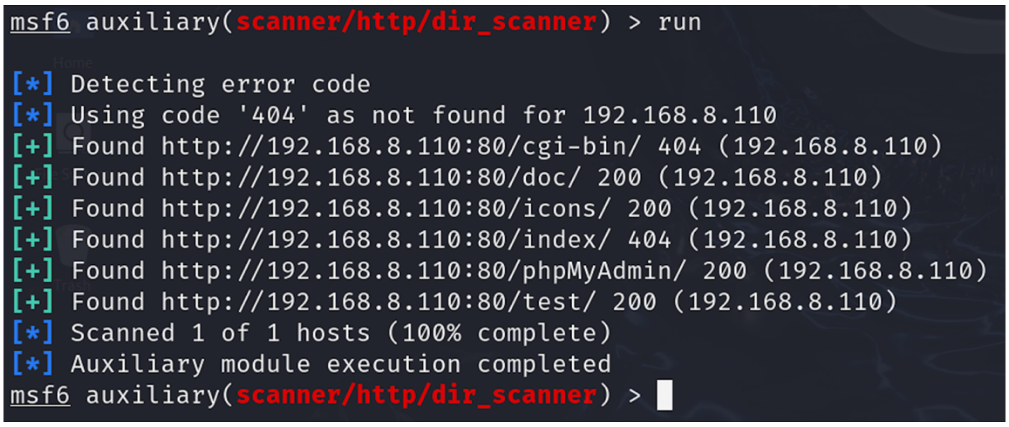
To Do: Students are advised to look for appropriate scripts to find out the versions of **`ssh`**, **`smb`**, **`mysql`**, and **`postgres`** at their own ☺

Performing Directory Scanning on Metasploitable2

A directory scanner in MSF refers to auxiliary scanner modules that can scan for directories, files or shares on NW services like `http, ftp, smb, nfs` and so on.

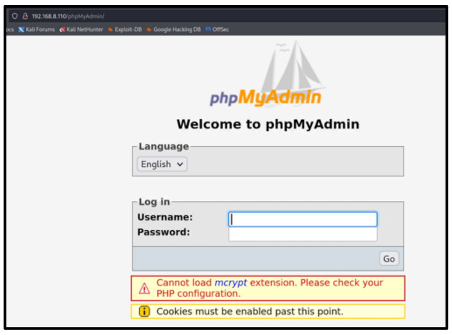
- **`http/dir_scanner:`** Let us run this scanner on **port#80**, where **Apache HTTP server** is running on Metasploitable2 and see what further information we can get:

```
msf6> use auxiliary/scanner/http/dir_scanner
msf6 auxiliary(scanner/http/dir_scanner)> show options
msf6 auxiliary(scanner/http/dir_scanner)> set RHOSTS <IP of M2>
msf6 auxiliary(scanner/http/dir_scanner)> run
```



```
msf6 auxiliary(scanner/http/dir_scanner) > run

[*] Detecting error code
[*] Using code '404' as not found for 192.168.8.110
[+] Found http://192.168.8.110:80/cgi-bin/ 404 (192.168.8.110)
[+] Found http://192.168.8.110:80/doc/ 200 (192.168.8.110)
[+] Found http://192.168.8.110:80/icons/ 200 (192.168.8.110)
[+] Found http://192.168.8.110:80/index/ 404 (192.168.8.110)
[+] Found http://192.168.8.110:80/phpMyAdmin/ 200 (192.168.8.110)
[+] Found http://192.168.8.110:80/test/ 200 (192.168.8.110)
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf6 auxiliary(scanner/http/dir_scanner) >
```

We got 6 directories. Going through their content might give us an edge to hack our target. Let us open the http://<IP of M2>:80/phpMyAdmin inside a browser on our Kali machine.



192.168.8.110/phpMyAdmin/

Welcome to phpMyAdmin

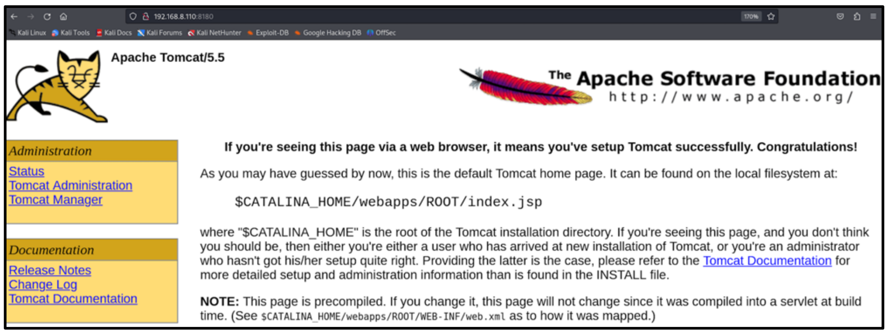
Language: English

Log in
Username:
Password:
Go

Cannot load *mcrypt* extension. Please check your PHP configuration.

Cookies must be enabled past this point.

- **`http/dir_scanner:`** When we performed the scan of M2, we came to know that at port **8180** a service **tomcat** is running. To verify this, on your Kali machine, open a browser and type this url: http://<ip of M2>:8180 and it will display the Apache Tomcat index page.



192.168.8.110:8180

Apache Tomcat/5.5

The Apache Software Foundation
http://www.apache.org/

Administration
Status
Tomcat Administration
Tomcat Manager

Documentation
Release Notes
Change Log
Tomcat Documentation

If you're seeing this page via a web browser, it means you've setup Tomcat successfully. Congratulations!

As you may have guessed by now, this is the default Tomcat home page. It can be found on the local filesystem at:

`$CATALINA_HOME/webapps/ROOT/index.jsp`

where "$CATALINA_HOME" is the root of the Tomcat installation directory. If you're seeing this page, and you don't think you should be, then either you're either a user who has arrived at new installation of Tomcat, or you're an administrator who hasn't got his/her setup quite right. Providing the latter is the case, please refer to the Tomcat Documentation for more detailed setup and administration information than is found in the INSTALL file.

NOTE: This page is precompiled. If you change it, this page will not change since it was compiled into a servlet at build time. (See `$CATALINA_HOME/webapps/ROOT/WEB-INF/web.xml` as to how it was mapped.)

- The Apache HTTP server is primarily designed to serve static content (like HTML, CSS, and images) and can execute scripts (like PHP) using modules.
- The Apache Tomcat server is specifically designed to run Java Servlets and JavaServer Pages (JSP). It supports Java-based web applications, providing an environment for dynamic content generation.

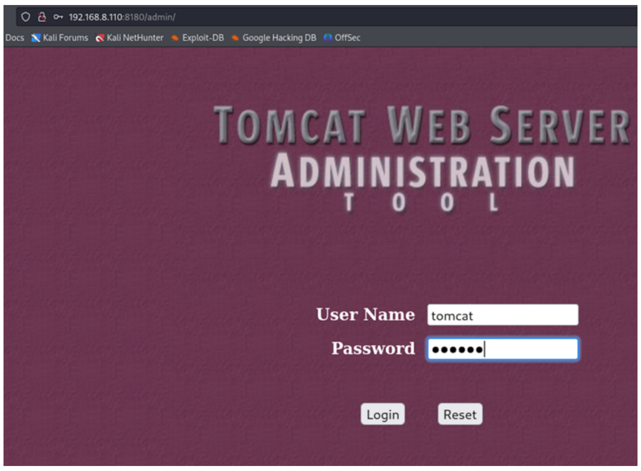
Let us run the directory scanner on **port#8180**, where **Apache Tomcat server** is running on Metasploitable2 and see what further information we can get:

```
msf6> use auxiliary/scanner/http/dir_scanner
msf6 auxiliary(scanner/http/dir_scanner)> show options
msf6 auxiliary(scanner/http/dir_scanner)> set RHOSTS <IP of M2>
msf6 auxiliary(scanner/http/dir_scanner)> run
```

```
[+] Found http://m2:8180/admin/    200 (10.0.2.7)
[+] Found http://m2:8180/webdav/    200 (10.0.2.7)
[+] Found http://m2:8180/tomcat-docs/    200 (10.0.2.7)
```

…………

Let us open the http://<IP of M2>:8180/admin inside a browser on our Kali machine. Try giving some random credentials and see if you can brute-force manually and login.

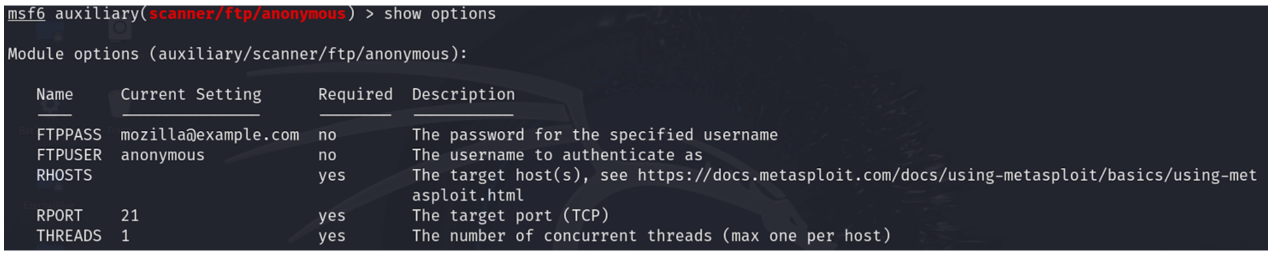


User Name tomcat
Password ••••••

Anonymous User Access w/o Password in NW Services

Some FTP servers are misconfigured in a way that allows anonymous access to remote users. Metasploitable 2 is designed to be vulnerable and may be misconfigured to allow anonymous access like `ftp`, `telnet`, `nfs`, `smtp`, `smb` and so on. Let us search for an appropriate script from auxiliary module of MSF, to check if ftp service running on M2 has anonymous user enabled. The `anonymous.rb` probes the target FTP server to check whether it allows anonymous access.

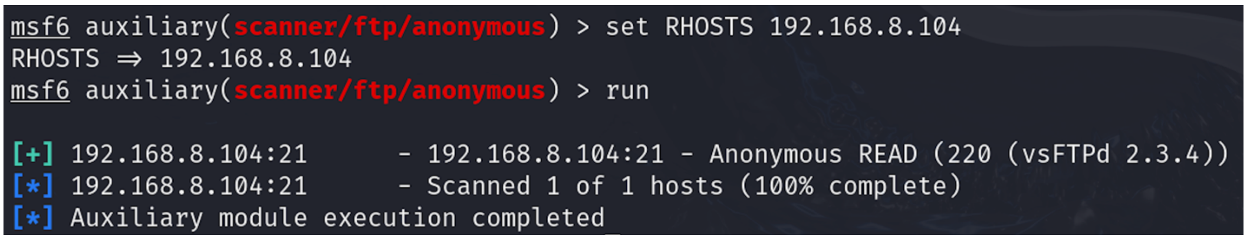
```
msf6> use auxiliary/scanner/ftp/anonymous
msf6 auxiliary(scanner/ftp/anonymous)> show options
```



```
msf6 auxiliary(scanner/ftp/anonymous) > show options

Module options (auxiliary/scanner/ftp/anonymous):

   Name     Current Setting      Required  Description
   ----     ---------------      --------  -----------
   FTPPASS  mozilla@example.com  no        The password for the specified username
   FTPUSER  anonymous            no        The username to authenticate as
   RHOSTS                        yes       The target host(s), see https://docs.metasploit.com/docs/using-metasploit/basics/using-met
                                           asploit.html
   RPORT    21                   yes       The target port (TCP)
   THREADS  1                    yes       The number of concurrent threads (max one per host)
```

- Now you need to set at least the RHOSTS parameter, you may change others as well and run

```
msf6 auxiliary(scanner/ftp/anonymous)> set RHOSTS <IP of M2>
msf6 auxiliary(scanner/ftp/anonymous)> run
```



```
msf6 auxiliary(scanner/ftp/anonymous) > set RHOSTS 192.168.8.104
RHOSTS => 192.168.8.104
msf6 auxiliary(scanner/ftp/anonymous) > run

[+] 192.168.8.104:21      - 192.168.8.104:21 - Anonymous READ (220 (vsFTPd 2.3.4))
[*] 192.168.8.104:21      - Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
```

- The output shows that one can use an `ftp` client to access Metasploitable2 with username of `anonymous` and a blank password. So, in another terminal of Kali, give the following command

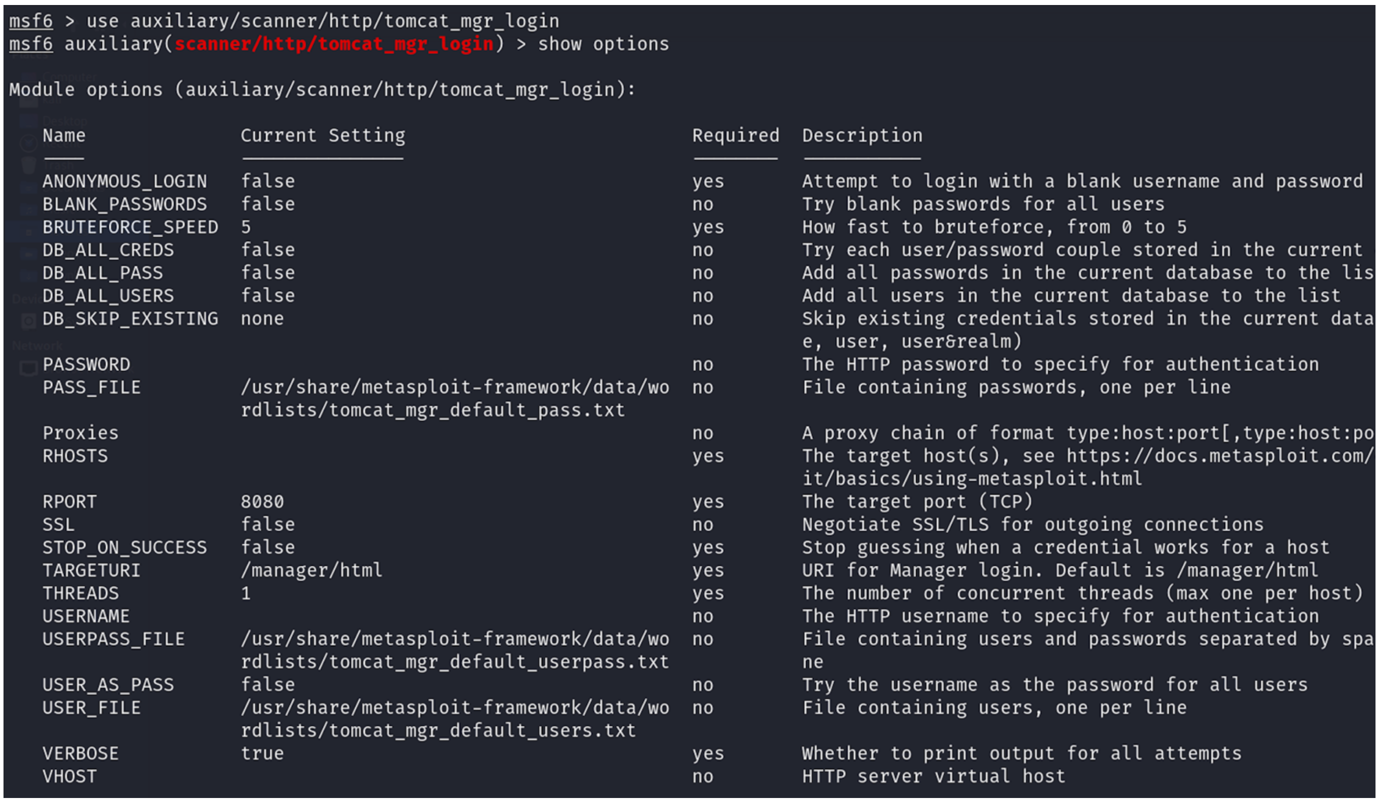
```
$ ftp <ip of M2>
```

Brute-Force Login on Metasploitable2

We have seen that **Apache Tomcat server** is running on M2 at port **8180**. We can access the admin panel from the link http://<IP of M2>:8180/admin, provided if we know the login credentials. The `tomcat_mgr_login.rb` is a script that can perform a brute-force attack against the target Tomcat server with a provided list of usernames and passwords. For this to work, other than `RHOSTS` and `RPORT` parameters, you may have to set other parameters like:

- `USERNAME` & `PASSWORD` (use these if you want to give single username & password)
- `USER_FILE` & `PASS_FILE` (two files containing users and passwords, one per line))
- `USERPASS_FILE` (one file containing users and passwords separated by space)

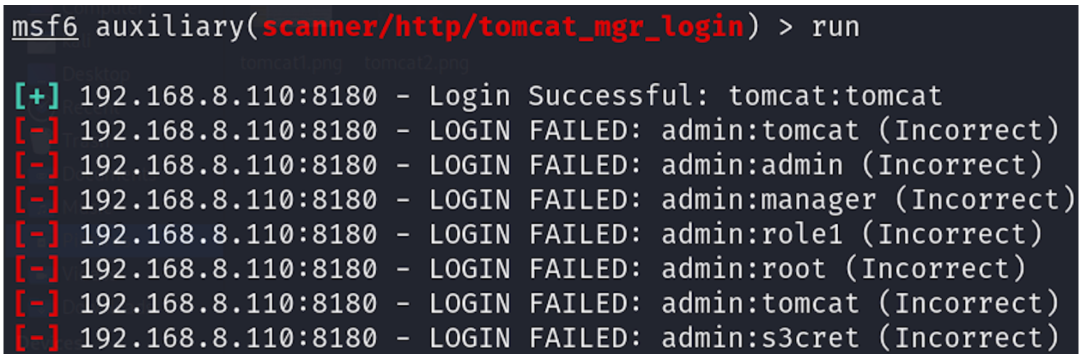
```
msf6> use auxiliary/scanner/http/tomcat_mgr_login
msf6 auxiliary(scanner/http/tomcat_mgr_login)> show options
```



```
msf6 > use auxiliary/scanner/http/tomcat_mgr_login
msf6 auxiliary(scanner/http/tomcat_mgr_login) > show options

Module options (auxiliary/scanner/http/tomcat_mgr_login):

   Name              Current Setting                           Required  Description
   ----              ---------------                           --------  -----------
   ANONYMOUS_LOGIN   false                                     yes       Attempt to login with a blank username and password
   BLANK_PASSWORDS   false                                     no        Try blank passwords for all users
   BRUTEFORCE_SPEED  5                                         yes       How fast to bruteforce, from 0 to 5
   DB_ALL_CREDS      false                                     no        Try each user/password couple stored in the current
   DB_ALL_PASS       false                                     no        Add all passwords in the current database to the lis
   DB_ALL_USERS      false                                     no        Add all users in the current database to the list
   DB_SKIP_EXISTING  none                                      no        Skip existing credentials stored in the current data
                                                                         e, user, user&realm)
   PASSWORD                                                    no        The HTTP password to specify for authentication
   PASS_FILE         /usr/share/metasploit-framework/data/wo  no        File containing passwords, one per line
                     rdlists/tomcat_mgr_default_pass.txt

   Proxies                                                     no        A proxy chain of format type:host:port[,type:host:po
   RHOSTS                                                      yes       The target host(s), see https://docs.metasploit.com/
                                                                         it/basics/using-metasploit.html

   RPORT             8080                                      yes       The target port (TCP)
   SSL               false                                     no        Negotiate SSL/TLS for outgoing connections
   STOP_ON_SUCCESS   false                                     yes       Stop guessing when a credential works for a host
   TARGETURI         /manager/html                             yes       URI for Manager login. Default is /manager/html
   THREADS           1                                         yes       The number of concurrent threads (max one per host)
   USERNAME                                                    no        The HTTP username to specify for authentication
   USERPASS_FILE     /usr/share/metasploit-framework/data/wo  no        File containing users and passwords separated by spa
                     rdlists/tomcat_mgr_default_userpass.txt                   ne
   USER_AS_PASS      false                                     no        Try the username as the password for all users
   USER_FILE         /usr/share/metasploit-framework/data/wo  no        File containing users, one per line
                     rdlists/tomcat_mgr_default_users.txt

   VERBOSE           true                                      yes       Whether to print output for all attempts
   VHOST                                                       no        HTTP server virtual host
```

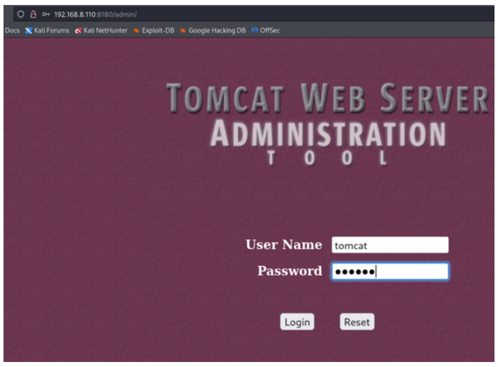
```
$ cat /usr/share/Metasploit-framework/data/wordlists/tomcat_mgr_default_users.txt
$ cat /usr/share/Metasploit-framework/data/wordlists/tomcat_mgr_default_pass.txt
```

```
msf6> set --clear username
msf6> set --clear password
msf6> set --clear user_file
msf6> set --clear pass_file
msf6> set --clear userpass_file
msf6> set user_file /usr/share/Metasploit-framework/data/wordlists/tomcat_mgr_default_users.txt
msf6> set user_file /usr/share/Metasploit-framework/data/wordlists/tomcat_mgr_default_pass.txt
msf6> set RHOSTS <IP of M2>
msf6> set RPORT 8180
msf6> run
```

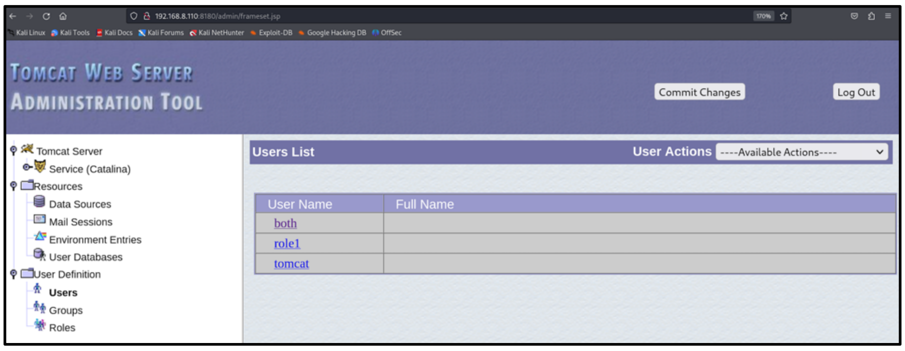


```
msf6 auxiliary(scanner/http/tomcat_mgr_login) > run

[+] 192.168.8.110:8180 - Login Successful: tomcat:tomcat
[-] 192.168.8.110:8180 - LOGIN FAILED: admin:tomcat (Incorrect)
[-] 192.168.8.110:8180 - LOGIN FAILED: admin:admin (Incorrect)
[-] 192.168.8.110:8180 - LOGIN FAILED: admin:manager (Incorrect)
[-] 192.168.8.110:8180 - LOGIN FAILED: admin:role1 (Incorrect)
[-] 192.168.8.110:8180 - LOGIN FAILED: admin:root (Incorrect)
[-] 192.168.8.110:8180 - LOGIN FAILED: admin:tomcat (Incorrect)
[-] 192.168.8.110:8180 - LOGIN FAILED: admin:s3cret (Incorrect)
```

- We have found the credentials ☺. Now from our Kali Linux machine, let us try to login in the admin panel of tomcat server running on Metasploitable machine using the username:`tomcat` and password:`tomcat`.



User Name tomcat
Password ••••••

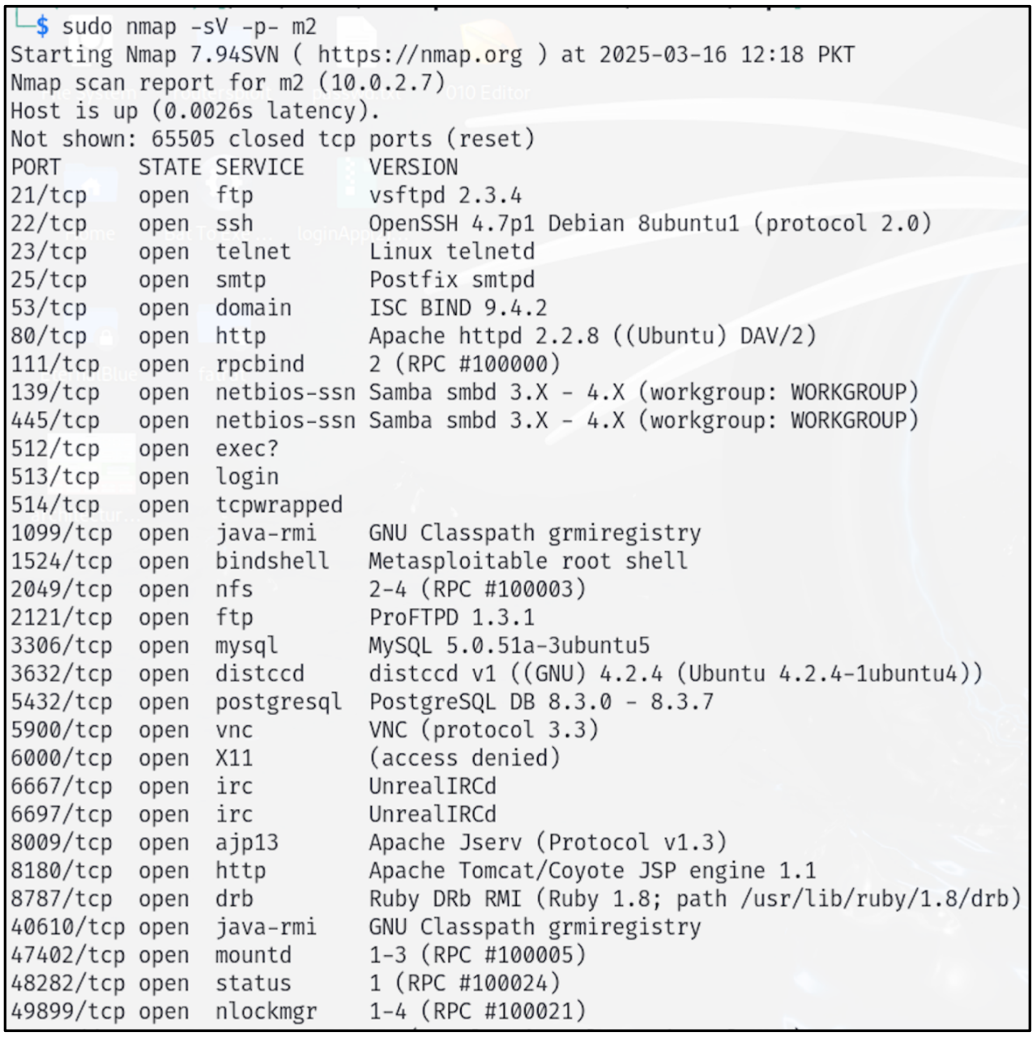


Users List

User Name	Full Name
both	
role1	
tomcat	

To Do: In a similar fashion, students are advised to perform other Brute-Force login scans using the `ssh_login.rb`, `mysql_login.rb`, and `postgres_login.rb` scripts.

Summary (Vulnerable Services running on Metasploitable2)



```
└─$ sudo nmap -sV -p- m2
Starting Nmap 7.94SVN ( https://nmap.org ) at 2025-03-16 12:18 PKT
Nmap scan report for m2 (10.0.2.7)
Host is up (0.0026s latency).
Not shown: 65505 closed tcp ports (reset)
PORT      STATE SERVICE     VERSION
21/tcp    open  ftp         vsftpd 2.3.4
22/tcp    open  ssh         OpenSSH 4.7p1 Debian 8ubuntu1 (protocol 2.0)
23/tcp    open  telnet      Linux telnetd
25/tcp    open  smtp        Postfix smtpd
53/tcp    open  domain      ISC BIND 9.4.2
80/tcp    open  http        Apache httpd 2.2.8 ((Ubuntu) DAV/2)
111/tcp   open  rpcbind     2 (RPC #100000)
139/tcp   open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
445/tcp   open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
512/tcp   open  exec?
513/tcp   open  login
514/tcp   open  tcpwrapped
1099/tcp  open  java-rmi    GNU Classpath grmiregistry
1524/tcp  open  bindshell   Metasploitable root shell
2049/tcp  open  nfs         2-4 (RPC #100003)
2121/tcp  open  ftp         ProFTPD 1.3.1
3306/tcp  open  mysql       MySQL 5.0.51a-3ubuntu5
3632/tcp  open  distccd     distccd v1 ((GNU) 4.2.4 (Ubuntu 4.2.4-1ubuntu4))
5432/tcp  open  postgresql  PostgreSQL DB 8.3.0 - 8.3.7
5900/tcp  open  vnc         VNC (protocol 3.3)
6000/tcp  open  X11         (access denied)
6667/tcp  open  irc         UnrealIRCd
6697/tcp  open  irc         UnrealIRCd
8009/tcp  open  ajp13       Apache Jserv (Protocol v1.3)
8180/tcp  open  http        Apache Tomcat/Coyote JSP engine 1.1
8787/tcp  open  drb         Ruby DRb RMI (Ruby 1.8; path /usr/lib/ruby/1.8/drb)
40610/tcp open  java-rmi    GNU Classpath grmiregistry
47402/tcp open  mountd      1-3 (RPC #100005)
48282/tcp open  status      1 (RPC #100024)
49899/tcp open  nlockmgr    1-4 (RPC #100021)
```

1. **TCP Port 21 - vsftpd 2.3.4 (FTP Server)**
   - **CVE:** CVE-2011-2523
   - **Attack Vector:** This version of vsftpd contains a backdoor that, when a user logs in with a username ending with ":)", opens a reverse shell on port 6200/tcp, granting unauthorized remote access.
2. **TCP Port 22 - OpenSSH 4.7p1 (SSH Server)**
   - **CVE:** No specific CVE is associated with this version, it is susceptible to brute-force attacks due to weak configurations.
   - **Attack Vector:** Attackers can perform brute-force attacks to guess valid SSH credentials, potentially gaining unauthorized access.
3. **TCP Port 23 - Telnet (Remote Login Service)**
   - **CVE:** No specific CVE; however, Telnet transmits data in plaintext.
   - **Attack Vector:** Credentials and data can be intercepted by attackers through network sniffing due to the lack of encryption.
4. **TCP Port 25 - Postfix (SMTP Server)**
   - **CVE:** No specific CVE; misconfiguration can lead to open relay issues.
   - **Attack Vector:** An open mail relay can be exploited by spammers to send unsolicited emails, potentially blacklisting the server.
5. **TCP Port 53 - BIND 9.4.2 (DNS Server)**
   - **CVE:** CVE-2009-0025
   - **Attack Vector:** Allows remote attackers to cause a denial of service via unspecified vectors related to DNSSEC validation.

6. **TCP Port 80 - Apache 2.2.8 (HTTP Server)**
   - **CVE:** CVE-2007-6750
   - **Attack Vector:** Vulnerable to denial of service via partial HTTP requests.
7. **TCP Ports 139 & 445 - Samba 3.0.20 (SMB/CIFS)**
   - **CVE:** CVE-2007-2447
   - **Attack Vector:** A flaw in Samba's "username map script" parameter allows remote code execution when this option is enabled.
8. **TCP Port 512, 513, 514 - Rexec, Rlogin, Rsh (Remote Execution Services)**
   - **CVE:** No specific CVEs; these services are inherently insecure.
   - **Attack Vector:** Transmit data, including passwords, in plaintext, making them susceptible to interception and unauthorized remote command execution.
9. **TCP Port 2049 - NFS (Network File System)**
   - **CVE:** No specific CVE; misconfigurations can lead to unauthorized access.
   - **Attack Vector:** If improperly configured, remote attackers can mount NFS shares and access sensitive files.
10. **TCP Port 2121 - ProFTPD 1.3.1 (FTP Server)**
    - **CVE:** CVE-2006-5815
    - **Attack Vector:** ProFTPD 1.3.1 is vulnerable to a command injection flaw that could allow remote attackers to execute arbitrary commands via crafted input.
11. **TCP Port 3306 - MySQL 5.0.51a (Database Server)**
    - **CVE:** No specific CVE; default configurations may have weak security.
    - **Attack Vector:** Default installations may allow root access without a password or with a weak password, enabling unauthorized database access.
12. **TCP Port 5432 - PostgreSQL 8.3.0 (Database Server)**
    - **CVE:** No specific CVE; potential for weak configurations.
    - **Attack Vector:** Default or weak passwords can allow attackers to gain unauthorized access to the database.
13. **TCP Port 5900 - VNC (Virtual Network Computing)**
    - **CVE:** No specific CVE; depends on configuration.
    - **Attack Vector:** If not properly secured, VNC can allow unauthorized remote desktop access, especially if no password is set.
14. **TCP Port 6667 - UnrealIRCd 3.2.8.1 (IRC Server)**
    - **CVE:** CVE-2010-2075
    - **Attack Vector:** A version of UnrealIRCd was distributed with a backdoor that allows remote command execution by sending crafted commands to the server.

Disclaimer

The series of handouts distributed with this course are only for educational purposes. Any actions and or activities related to the material contained within this handout is solely your responsibility. The misuse of the information in this handout can result in criminal charges brought against the persons in question. The authors will not be held responsible in the event any criminal charges be brought against any individuals misusing the information in this handout to break the law.