

## 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

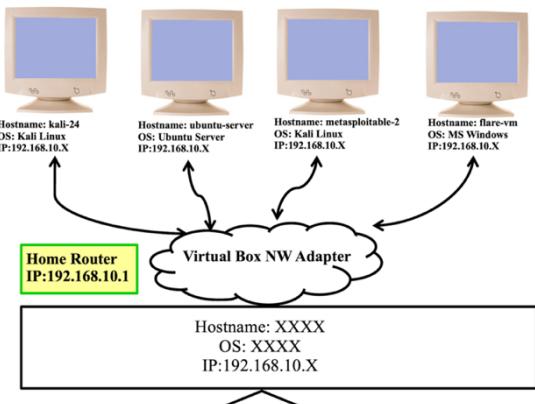
This handout is continuation of our previous Handout#2.3. The objective of scanning is to identify system services and potential entry points in a network by performing NW scanning, port scanning and Services detection, using tools like nmap, zenmap, unicorn, nikto and so on. The objective of vulnerability 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**.

- The steps that we normally perform during vulnerability analysis are:
  - Scanning the target for known vulnerabilities using databases like CVE.
  - Assessing the severity of discovered vulnerabilities using metrics like CVSS.
  - Submit report highlighting the vulnerability, risk levels, and mitigation steps.

### 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:

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



Overview of Metasploit Framework <https://www.metasploit.com/>

The Metasploit Framework 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/>). It provides security professionals and researchers with a comprehensive set of tools for *discovering* and *exploiting* vulnerabilities in various systems and applications. Kali Linux comes pre-installed with Metasploit framework.

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:

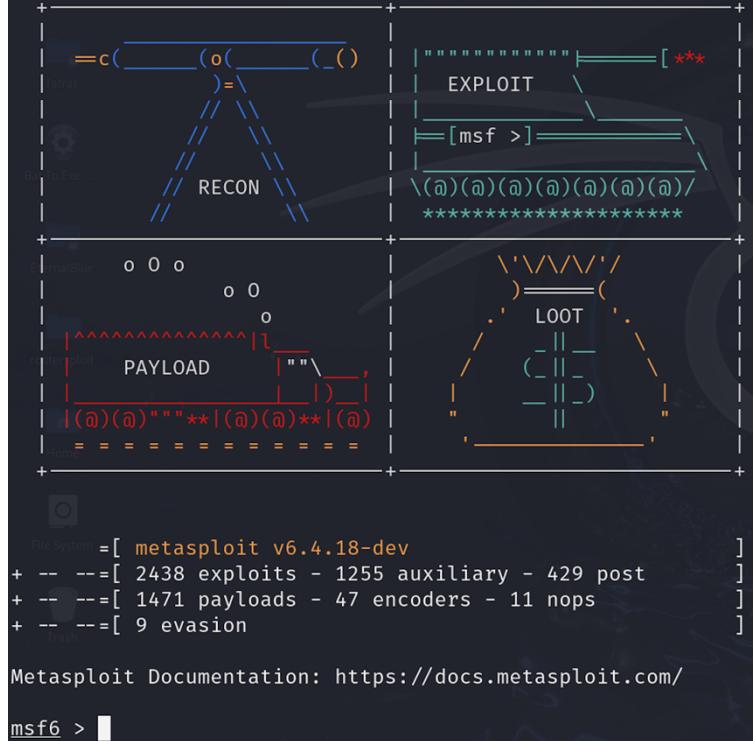
```
$ sudo msfconsole
```

- ASCII logo
  - Metasploit version (6.4.18),
    - 2438 exploits
    - 1255 auxiliary
    - 429 post
    - 1471 payloads
    - 47 encoders
    - 11 nops
    - 9 evasion

```
msf6> help
```

- Once msfconsole is running, you can use its 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 ☺

```
[+] msf6 =[ 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 > ]
```



## 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 task. 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, sniffers, fuzzers, etc. 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 contain self-contained payloads that perform a single task like creating a user `singles/windows/adduser.rb`. The `stagers` sub-directory contains small payloads that establish a connection to download and execute larger payloads. The `stages` sub-directory contains larger payloads that are delivered by stagers. These payloads are used to execute commands, establish remote connections, and provide control over the target.
  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, which are sequences of NOP instructions used in buffer overflow exploits. Adding NOPs can significantly help in modifying the payload signatures and thereby avoiding detection.
  6. **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 extraction, and maintaining persistence.
  7. **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.

## Basic Commands of msfconsole

I recommend to run `msfconsole` command as `sudo`, as you may need that power later. Since the Meterpreter 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
<code>msf6 &gt; help</code> <code>msf6 &gt; help &lt;command&gt;</code>	The simple <code>help</code> 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
<code>msf6 &gt; banner</code>	Print a stunning ASCII art banner along with version information and module counts
<code>msf6 &gt; exit/quit</code>	The <code>exit</code> or <code>quit</code> command will simply exit <code>msfconsole</code> utility
<code>msf6 &gt; show auxiliary</code>	The <code>show</code> command is passed one argument that can be a module name exploits, payloads, auxiliary...
<code>msf6 &gt; search telnet</code> <code>msf6 &gt; search type:auxiliary telnet</code> <code>msf6 &gt; search type:exploit telnet</code> <code>msf6 &gt; search cve:2017-0144</code>	The <code>search</code> command is used to search for exploits, payloads, auxiliary module. Use <code>search</code> if you are looking for modules that are ready to be used within MSF.
<code>msf6 &gt; searchsploit telnet</code> <code>msf6 &gt; searchsploit eternalblue</code>	The <code>searchsploit</code> is a stand-alone command line tool used to search EDB for publicly available exploits, shellcodes, and vulnerabilities. Use <code>search</code> when you want to find exploits from the EDB that are not yet integrated into MSF.
<code>msf6 &gt;info auxiliary/scanner/portscan/syn</code>	Once you have identified the module you are interested in using, you can use the <code>info</code> command to find out more about it
<code>msf6 &gt;use auxiliary/scanner/portscan/syn</code> <code>msf6 auxiliary(scanner/portscan/syn)&gt;</code>	Once you are done with searching a specific module, then you give <code>use</code> 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 <code>back</code> command to move out of the current context.
<code>msf6 auxiliary(scanner/portscan/syn)&gt; show options</code>	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 <code>show options</code> command to display which settings are available and/or required for that specific module
<code>msf6 auxiliary(scanner/portscan/syn)&gt; show advanced</code>	To view any advanced options that may be available for a given module, you can use the <code>show advanced</code> command
<code>msf6 exploit(multi/handler)&gt;show payloads</code>	When you are in the context of a particular exploit, running <code>show payloads</code> command will only display the payloads that are compatible with that particular

	exploit. Later you can select the payload of your choice using its name or number
<code>msf6 exploit(multi/handler)&gt;show targets</code>	When you are in the context of a particular exploit, and you are not certain whether an OS is vulnerable to this exploit, the <b>show targets</b> command will display the supported OS targets. Later you can select the target of your choice using its name or number
<code>msf6 exploit(multi/handler)&gt; set &lt;param&gt; &lt;value&gt;</code>	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 <b>set</b> command to update the value of a parameter
<code>msf6 exploit(multi/handler)&gt; unset &lt;param&gt;</code>	The <b>unset</b> command is opposite of the <b>set</b> command, which removes a parameter previously configured with <b>set</b> command. You can remove all assigned variables with <b>unset all</b> command
<code>msf6 exploit(multi/handler)&gt; setg RHOSTS &lt;ip&gt;</code>	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 <b>setg</b> command to set the value of that parameter for all modules
<code>msf6 exploit(multi/handler)&gt; run</code>	Once you have configured all parameters marked as required for the module you have loaded, you can execute it using the <b>run</b> or <b>exploit</b> command

## Performing Port Scannings (**portscan**) 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 smptd
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?      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
-  --
0  auxiliary/scanner/portscan/ftpbounce
1  auxiliary/scanner/natpmp/natpmp_portscan
2  auxiliary/scanner/sap/sap_router_portscanner
3  auxiliary/scanner/portscan/xmas
4  auxiliary/scanner/portscan/ack
5  auxiliary/scanner/portscan/tcp
6  auxiliary/scanner/portscan/syn
7  auxiliary/scanner/http/wordpress_pingback_access
```

#	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            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            yes       The target host(s), see https://docs.metasploit.com/docs/using-metasploit/basics/using-metasploit.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`

```
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:** Students are advised to perform other scans inside the `portscan` directory at their own 😊

## Performing SMB Scannings (**smb**) on Metasploitable2

- Now that we have determined which hosts are available on the network, we can attempt to determine the operating systems they are running. This will help us narrow down our attacks to target a specific system and will stop us from wasting time on those that aren't vulnerable to a particular exploit. SMB stands for Server Message Block, which is a client-server communication protocol that is used for shared access to files, directories, printers, securely over a network between different operating systems, i.e., windows-windows, Unix-Unix, and Unix-Windows. It uses TCP port 139 and 445 for communication.
- Let us search the different matching modules with the keyword **smb**:

```
msf6> search smb
```

<b>msf6 &gt; search smb</b>		<b>Matching Modules</b>	<b>Disclosure Date</b>	<b>Rank</b>	<b>Check</b>	<b>De</b>
#	Name					
description						
-	—	—	—	—	—	—
0	exploit/multi/http/struts_code_exec_classloader	2014-03-06	manual	No	Ap	
	ache Struts ClassLoader Manipulation Remote Code Execution					
1	\_ target: Java	.	.	.	.	
2	\_ target: Linux	.	.	.	.	
3	\_ target: Windows	.	.	.	.	
4	\_ target: Windows / Tomcat 6 & 7 and GlassFish 4 (Remote <b>SMB</b> Resource)	.	.	.	.	
5	exploit/osx/browser/safari_file_policy	2011-10-12	normal	No	Ap	
	ple Safari file:// Arbitrary Code Execution					
6	\_ target: Safari 5.1 on OS X	.	.	.	.	
7	\_ target: Safari 5.1 on OS X with Java	.	.	.	.	
8	auxiliary/server/capture/ <b>smb</b>	.	normal	No	Au	
	thentication Capture: <b>SMB</b>					
9	post/linux/busybox/ <b>smb_share_root</b>	.	normal	No	Bu	
	syBox <b>SMB</b> Sharing					
10	exploit/linux/misc/cisco_rv340_sslvpn	2022-02-02	good	Yes	Ci	
	sco RV340 SSL VPN Unauthenticated Remote Code Execution					
11	auxiliary/scanner/http/citrix_dir_traversal	2019-12-17	normal	No	Ci	
	trix ADC (NetScaler) Directory Traversal Scanner					
12	auxiliary/gather/crushftp_fileread_cve_2024_4040	.	normal	Yes	Cr	
	ushFTP Unauthenticated Arbitrary File Read					
13	auxiliary/scanner/ <b>smb</b> /impacket/dcomexec	2018-03-19	normal	No	DC	

- There are a variety of SMB auxiliaries that are available to us within `/usr/share/metasploit-framework/modules/auxiliary/scanner/smb/` directory. Some of these are `smb_version.rb`, `smb_enumusers.rb`, `smb_login.rb` and so on. First let us use the following command to check out different types of scans that are there in the **smb** module
- Let us use `smb_version`:** Remember when we ran `nmap` on Metasploitable2, we came to know that SMB service was running on port 139 and 445, but it did not tell us its version. Let us use `smb_version.rb` to check out.

```
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         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)> set THREADS 50
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) > █
```

- So we have succeeded in finding the exact version, and that is Samba 3.0.20-Debian ☺

**To Do:** Students are advised to perform other scans inside the **smb** directory at their own ☺

## Performing FTP Scannings (**ftp**) on Metasploitable2

- FTP is mostly used for file sharing between the client and server. It uses TCP port 21 for communication. There exist different types of FTP scans inside MSF, that we can perform on a network or on a specific machine inside the `/usr/share/metasploit-framework/modules/auxiliary/scanner/ftp` directory. Let me give you a brief description of some:
  - Some FTP servers are misconfigured in a way that allows anonymous access to remote users. The `anonymous.rb` probes the target FTP server to check whether it allows anonymous access.
  - The `ftp_version.rb` uses the banner grabbing technique to detect the version of the target FTP server.
  - The `ftp_login.rb` help us perform a brute-force attack against the target FTP server. For this to work, other than `RHOSTS` parameter, you must also configure the `USERPASS_FILE` parameter to the path to the file containing the username/password list. You can either create your own custom list that can be used for a brute-force attack, or there are many wordlists instantly available for use in Kali Linux, located at `/usr/share/wordlists`. ☺

- Let us check if `vsftpd` running on M2 allows **anonymous** login:

```
msf6> use auxiliary/scanner/ftp/anonymous  
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 <a href="https://docs.metasploit.com/docs/using-metasploit/basics/using-metasploit.html">https://docs.metasploit.com/docs/using-metasploit/basics/using-metasploit.html</a>
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 the `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>
```

**To Do:** Students are advised to perform other scans inside the **ftp** directory at their own ☺

## Performing HTTP Scannings ([http](http://)) on Metasploitable2

- HTTP is a stateless application layer protocol used for the exchange of information on the World Wide Web. HTTP uses TCP port 80 for communication.
- Let us perform a `nmap` scan on port 80 of our Metasploitable2 machine.

```
msf6> nmap -p 80 -sV <IP of M2>
$ nmap -sV -p 80 <IP of M2>
```

```
(kali㉿kali)-[~]
└─$ nmap -sV 192.168.8.110 -p 80
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-09-19 12:28 EDT
Nmap scan report for 192.168.8.110
Host is up (0.0011s latency).

PORT      STATE SERVICE VERSION
80/tcp    open  http      Apache httpd 2.2.8 ((Ubuntu) DAV/2)

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 7.07 seconds
```

- The above output shows that Apache httpd 2.2.8 Web server is running on port 80 at Metasploitable2 machine. The Apache HTTP Server is the most widely used web server software and runs on 67% of all web sites in the world. It is open source and is available for all UNICES, Mac, Linux and Microsoft platforms.
- Now in MSF, there are a variety of HTTP auxiliaries that are available to us within `/usr/share/metasploit-framework/modules/auxiliary/scanner/http/` directory, like `http_version.rb`, `dir_scanner.rb`, `robots_txt.rb` and so on.

### 1. Let us use `http_version.rb`:

```
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          none      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 ☺



<b>PHP Version 5.2.4-2ubuntu5.10</b>	
<b>php</b>	
<b>System</b>	Linux metasploitable 2.6.24-16-server #1 SMP Thu Apr 10 13:58:00 UTC 2008 i686
<b>Build Date</b>	Jan 6 2010 21:50:12
<b>Server API</b>	CGI/FastCGI
<b>Virtual Directory Support</b>	disabled
<b>Configuration File (php.ini) Path</b>	/etc/php5/cgi
<b>Loaded Configuration File</b>	/etc/php5/cgi/php.ini
<b>Scan this dir for additional .ini files</b>	/etc/php5/cgi/conf.d
<b>additional .ini files parsed</b>	/etc/php5/cgi/conf.d/gd.ini, /etc/php5/cgi/conf.d/mysql.ini, /etc/php5/cgi/conf.d/mysql.ini, /etc/php5/cgi/conf.d/pdo.ini, /etc/php5/cgi/conf.d/pdo_mysql.ini
<b>PHP API</b>	20041225
<b>PHP Extension</b>	20060613
<b>Zend Extension</b>	220060519
<b>Debug Build</b>	no
<b>Thread Safety</b>	disabled

**2. Let us use `dir_scanner.rb`:** Let us try some other scans in the http module and gather some more information about the web server. Let us use `dir_scanner.rb` on the 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 > use auxiliary/scanner/http/dir_scanner
msf6 auxiliary(scanner/http/dir_scanner)> show options

Module options (auxiliary/scanner/http/dir_scanner):
=====
Name      Current Setting          Required  Description
--  -----
DICTIONARY /usr/share/metasploit-framework/data/wmap/wmap_dirs.txt    no        Path of word dictionary to use
PATH      /                         yes       The path to identify files
Proxies   no                        no        A proxy chain of format type:host:port[,type:host:port][...]
RHOSTS   192.168.8.110             yes       The target host(s), see https://docs.metasploit.com/docs/using-metasploit/basics/using-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    None                       no        HTTP server virtual host

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

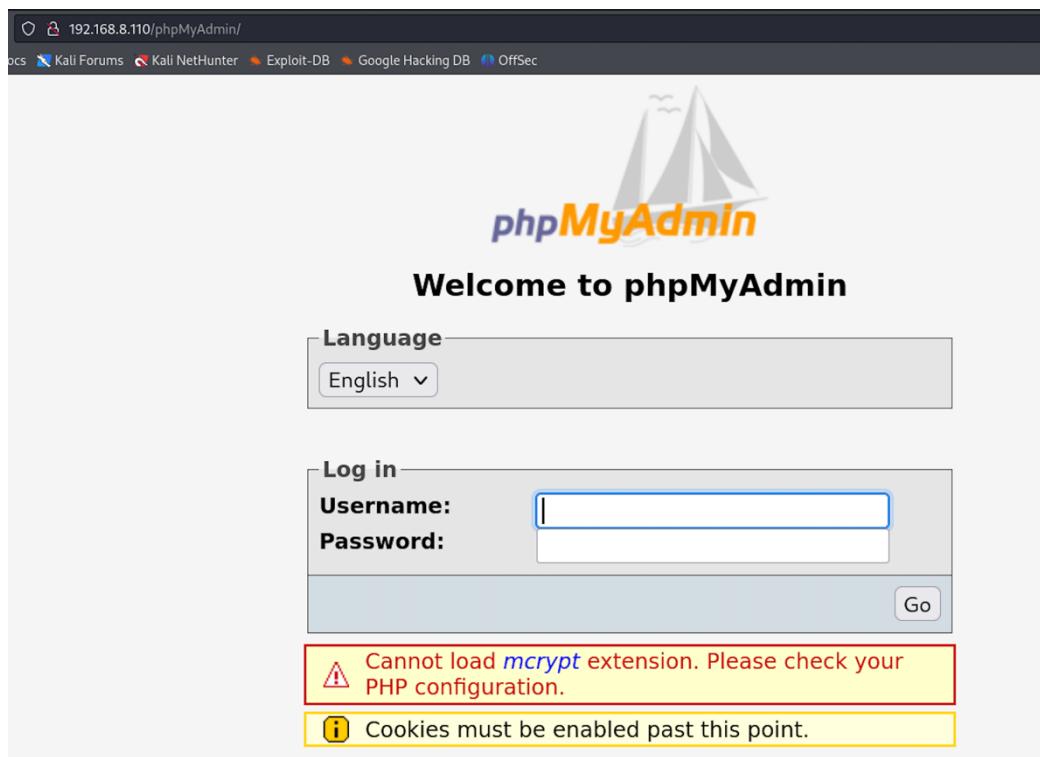
msf6 auxiliary(scanner/http/dir_scanner)> set RHOSTS 192.168.8.110
RHOSTS => 192.168.8.110
msf6 auxiliary(scanner/http/dir_scanner)> 
```

- Let us run now

```
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.



- Let us search exploitDB with Apache 2.2.8  
`$ searchsploit apache 2.2.8`

```
└─$ searchsploit apache 2.2.8
Exploit Title | Path
Apache + PHP < 5.3.12 / < 5.4.2 - cgi-bin Remote Code Execution | php/remote/29290.c
Apache + PHP < 5.3.12 / < 5.4.2 - Remote Code Execution + Scanner | php/remote/29316.py
Apache < 2.0.64 / < 2.2.21 mod_setenvif - Integer Overflow | linux/dos/41769.txt
Apache < 2.2.34 / < 2.4.27 - OPTIONS Memory Leak | linux/webapps/42745.py
Apache CXF < 2.5.10/2.6.7/2.7.4 - Denial of Service | multiple/dos/26710.txt
Apache mod_ssl < 2.8.7 OpenSSL - 'OpenFuck.c' Remote Buffer Overflow | unix/remote/21671.c
Apache mod_ssl < 2.8.7 OpenSSL - 'OpenFuckV2.c' Remote Buffer Overflow (1) | unix/remote/764.c
Apache mod_ssl < 2.8.7 OpenSSL - 'OpenFuckV2.c' Remote Buffer Overflow (2) | unix/remote/47080.c
Apache OpenMeetings 1.9.x < 3.1.0 - '.ZIP' File Directory Traversal | linux/webapps/39642.txt
Apache Struts 2 < 2.3.1 - Multiple Vulnerabilities | multiple/webapps/18329.txt
Apache Struts 2.0.1 < 2.3.33 / 2.5 < 2.5.10 - Arbitrary Code Execution | multiple/remote/44556.py
Apache Struts < 1.3.10 / < 2.3.16.2 - ClassLoader Manipulation Remote Code Execution (Metasploit) | multiple/remote/41690.rb
Apache Struts2 2.0.0 < 2.3.15 - Prefixed Parameters OGNL Injection | multiple/webapps/44583.txt
Apache Tomcat < 5.5.17 - Remote Directory Listing | multiple/remote/2061.txt
Apache Tomcat < 6.0.18 - 'utf8' Directory Traversal | unix/remote/14489.c
Apache Tomcat < 6.0.18 - 'utf8' Directory Traversal (PoC) | multiple/remote/6229.txt
Apache Tomcat < 9.0.1 (Beta) / < 8.5.23 / < 8.0.47 / < 7.0.8 - JSP Upload Bypass / Remote Code Execution | jsp/webapps/42966.py
Apache Tomcat < 9.0.1 (Beta) / < 8.5.23 / < 8.0.47 / < 7.0.8 - JSP Upload Bypass / Remote Code Execution | windows/webapps/42953.txt
Apache Xerces-C XML Parser < 3.1.2 - Denial of Service (PoC) | linux/dos/36906.txt
Webroot Shootbox < 2.32 (Apache) - Local File Inclusion / Remote Code Execution | linux/remote/34.pl

Shellcodes: No Results
```

- Let us further filter our search with PHP 5.2.4.

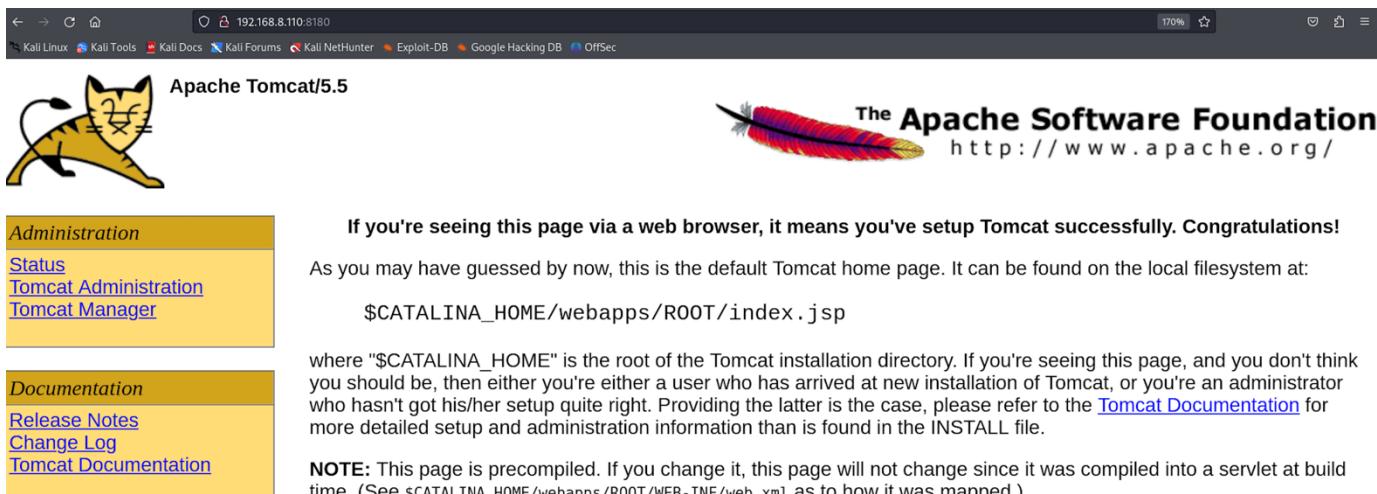
```
$ searchsploit apache 2.2.8 | grep php
```

```
└─(kali㉿kali)-[~]
└─$ searchsploit apache 2.2.8 | grep php
Apache + PHP < 5.3.12 / < 5.4.2 - cgi-bin Remote Code Execution | php/remote/29290.c
Apache + PHP < 5.3.12 / < 5.4.2 - Remote Code Execution + Scanner | php/remote/29316.py
```

- We have found two vulnerabilities and their corresponding exploits. We will exploit this vulnerability in our next handout 😊

### 3. Let us use tomcat\_mgr\_login.rb:

- When we run nmap on Metasploitable2, can see that 8180 port is open and running **tomcat** service. Now on your Kali machine, open a browser and type this url: <http://<ip of M2>:8180>



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 handle HTTP requests. Works mainly with static files; can execute scripts (like PHP) using modules. Ideal for serving websites that don't require dynamic content generation or Java Servlets/JSP. On the contrary, 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 use `tomcat_mgr_login.rb` on the Metasploitable2 and see what further information we can get:

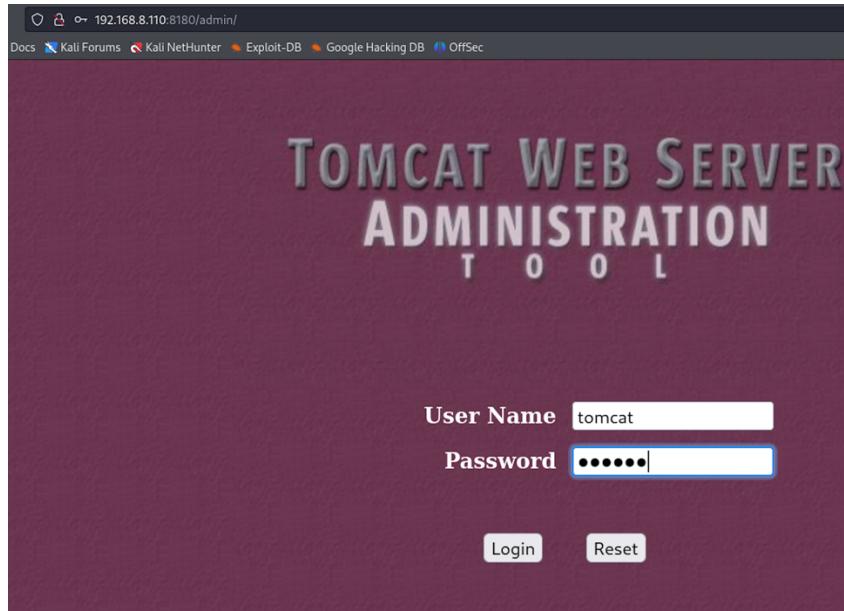
```
msf6> use auxiliary/scanner/http/tomcat_mgr_login
msf6 auxiliary(scanner/http/tomcat_mgr_login)> show options
```

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 database
DB_ALL_PASS	false	no	Add all passwords in the current database to the list
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 database, user, user&realm)
<i>Networks</i>			
PASSWORD		no	The HTTP password to specify for authentication
PASS_FILE	/usr/share/metasploit-framework/data/wordlists/tomcat_mgr_default_pass.txt	no	File containing passwords, one per line
<i>Proxies</i>			
RHOSTS		yes	A proxy chain of format type:host:port[,type:host:port]
RPORT	8080	yes	The target host(s), see https://docs.metasploit.com/it/basics/using-metasploit.html
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/wordlists/tomcat_mgr_default_userpass.txt	no	File containing users and passwords separated by space
USER_AS_PASS	false	no	Try the username as the password for all users
USER_FILE	/usr/share/metasploit-framework/data/wordlists/tomcat_mgr_default_users.txt	no	File containing users, one per line
VERBOSE	true	yes	Whether to print output for all attempts
VHOST		no	HTTP server virtual host

```
msf6 auxiliary(scanner/http/tomcat_mgr_login)> set RHOSTS <IP of M2>
msf6 auxiliary(scanner/http/tomcat_mgr_login)> set RPORT 8180
msf6 auxiliary(scanner/http/tomcat_mgr_login)> set USERNAME tomcat
msf6 auxiliary(scanner/http/tomcat_mgr_login)> set PASSWORD tomcat
msf6 auxiliary(scanner/http/tomcat_mgr_login)> 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 ☺ From our Kali Linux machine, let us now try to login in the admin panel of tomcat server running on Metasploitable machine using the username:tomcat and password:tomcat.

A screenshot of the Tomcat Web Server Administration Tool showing the "Users List" section. The URL in the address bar is 192.168.8.110:8180/admin/frameset.jsp. The left sidebar shows navigation links for Tomcat Server, Service (Catalina), Resources (Data Sources, Mail Sessions, Environment Entries, User Databases), User Definition (Users, Groups, Roles), and a "Logout" link. The main content area displays a table titled "Users List" with three rows. The columns are "User Name" and "Full Name". The data is as follows:

User Name	Full Name
both	
role1	
tomcat	

**To Do:** Students are advised to perform other scans inside the **http** directory at their own ☺

**Assignment:** Students are advised to perform various scans of the SMTP service (an email protocol used for sending email messages from one email account to another via the Internet) running on port 25 of Metasploitable2 and submit a detailed report of it ☺

## 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.*