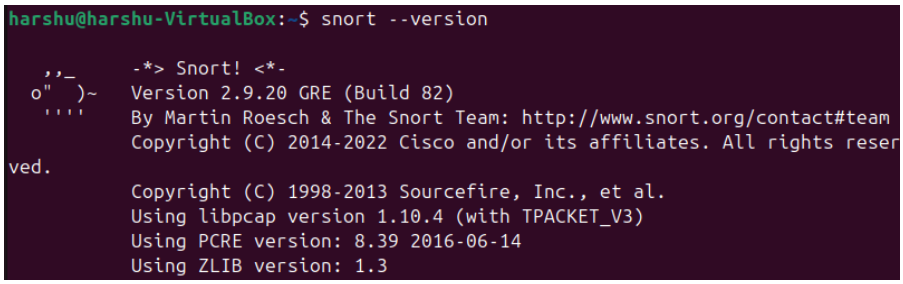
Assignment-2

PART-I

Installed snort successfully on my VM.



1)
a -
Zero-day Attack: It is the security risk or weakness in a software or system that is not known publicly and the vendors or developers are not aware of. In this attack, an attacker exploits the vulnerability before finding any fix to it by a software developer(developer has 0-days to fix this attack). This kind of attack is considered to be dangerous because the developer has not had the chance to fix the vulnerability yet.

Yes, snort can detect zero-day attacks to some extent. Snort can detect some of the zero-day attacks and some of the attacks may not be detected by snort. One study showed that out of 183 zero-day attacks, 17% of them were detected by the snort on average.

Snort checks and analyses the incoming and outgoing packets using Misuse Detection Engine BASE and then compares them with predefined rules and signatures.

Here is how snort helps in detecting zero-day attacks:

- Snort records and checks the packets so that it can identify the malicious patterns.
- It can detect network anomalies and flags them as unusual traffic behaviour which helps the security team to investigate threats.
- Snort is also used for packet sniffing, collecting individual packets in the network can provide us deep insights about how traffic is transmitted through the network.

- Snort notifies the users according to the rules setted in the config file. Snort gives alerts whenever the packet or traffic meets the certain predefined criteria.
- In snort we can also create new rules to make snort detection more efficient.

Sometimes snort may not detect the zero-day attacks here is why:

- Attackers can modify the exploits to pass the Snort detection rules without getting caught to snort.
- There are lack of predefined signatures. So, a snort cannot know the attack pattern without a signature. By this the attack detection can be missed. And also can misclassify the legitimate traffic.

b. Given:

- No. of connections = 1000000
- % of attacks = 1000 attacks
- True positives (Detection rate) = 95%
- Probability that an alarm is an attack(precision) = 95%

True Positives (TP) = True positive rate x % of attacks
= 0.95 X 1000 = 950

We know,

Precision = $\frac{True\ positives}{True\ positives + False\ positives}$

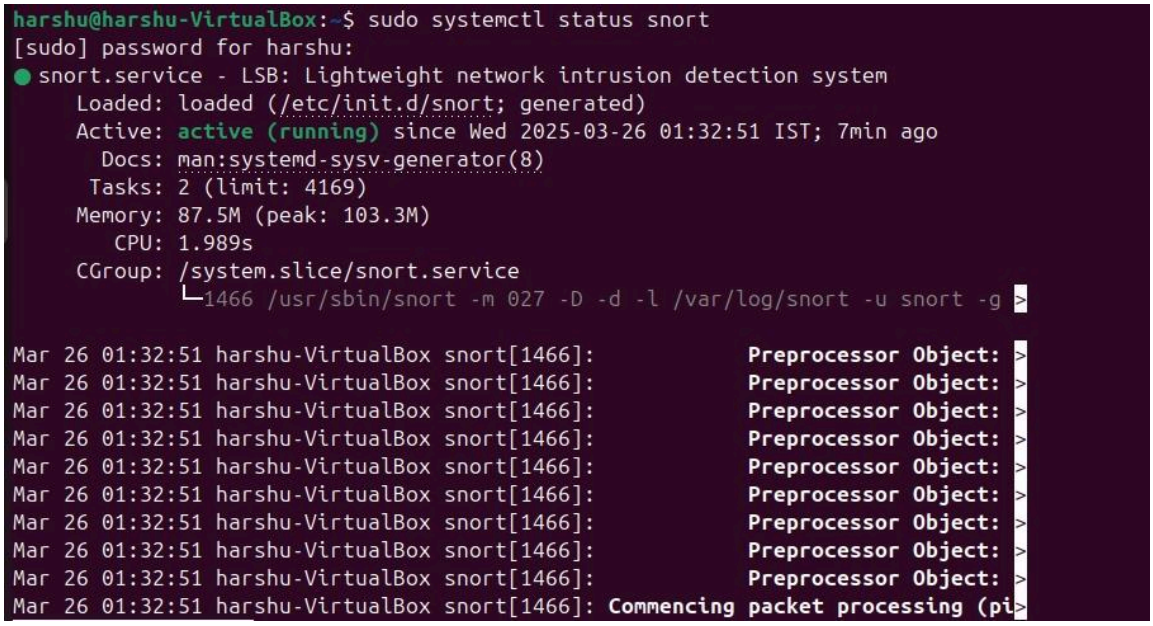
$0.95 = \frac{950}{950 + FP}$

FP = 50

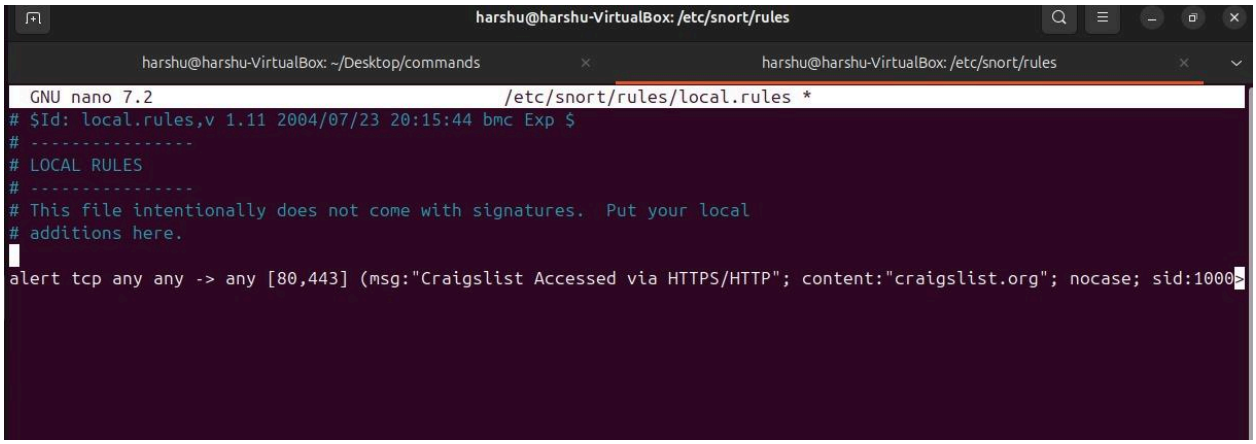
False Positive rate = 50 / 1000000 - 1000
= 50 / 999000
= 0.00005 = 0.005%

Check weather the snort is active or not the below command. By the below picture we can see that it is active.

- Sudo systemctl status snort.



I added the rule in the local.rules file.
Here is the rule which i used:



Alert - Tells snort to generate alert when rule is triggered.
any any (Source IP , Port) - matches traffic from any IP
Any [80, 443] (Destination IP, Port)
msg - This message is displayed in the alert log after traffic matched the rule.
nocase - makes the content check case sensitive.
sid - It is just a unique identifier of this rule.

How does this rule work? - from our machine(source) it intercepts TCP packets going to Craigslist's IP on ports 80 and 443. Checks packet payload for Host: craigslist.org in HTTP header. If a match is found Snort generates alert.

After adding the rule successfully, I checked whether the local.rules file is included in /etc/snort/snort.conf. It was included by default in the .config file.

then i started the snort with the below command

```
harshu@harshu-VirtualBox:/etc/snort/rules$ sudo snort -i enp0s3 -c /etc/snort/snort.conf
[sudo] password for harshu:
Running in IDS mode

        --== Initializing Snort ==--
Initializing Output Plugins!
```

```
harshu@harshu-VirtualBox: ~/Desktop/commands                    har
harshu@harshu-VirtualBox:~/Desktop/commands$ ifconfig
enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.0.2.15  netmask 255.255.255.0  broadcast 10.0.2.255
        inet6 fd00::a00:27ff:fe31:ab55  prefixlen 64  scopeid 0x0<global>
        inet6 fd00::8081:66ca:efca:fa5b  prefixlen 64  scopeid 0x0<global>
        inet6 fe80::a00:27ff:fe31:ab55  prefixlen 64  scopeid 0x20<link>
        ether 08:00:27:31:ab:55  txqueuelen 1000  (Ethernet)
        RX packets 25566  bytes 36415478 (36.4 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 2479  bytes 190922 (190.9 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 173  bytes 16515 (16.5 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 173  bytes 16515 (16.5 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

harshu@harshu-VirtualBox:~/Desktop/commands$
```

Now to alert the site "www.craiglist.org" i used this command:

- curl -v www.craiglist.org

And this is the result i got:

```
harshu@harshu-VirtualBox:/$ curl -v www.craiglist.org
* Host www.craiglist.org:80 was resolved.
* IPv6: (none)
* IPv4: 208.82.238.135
*   Trying 208.82.238.135:80...
* Connected to www.craiglist.org (208.82.238.135) port 80
> GET / HTTP/1.1
> Host: www.craiglist.org
> User-Agent: curl/8.5.0
> Accept: */*
>
< HTTP/1.1 301 Found
< Location: https://www.craigslist.org/
* no chunk, no close, no size. Assume close to signal end
<
* Closing connection
```

These are the below outputs from snort that i got after running the curl command on the website.

```
WARNING: No preprocessors configured for policy 0.
^C*** Caught Int-Signal
===============================================================================
Run time for packet processing was 107.250085 seconds
Snort processed 15 packets.
Snort ran for 0 days 0 hours 1 minutes 47 seconds
   Pkts/min:           15
   Pkts/sec:            0
===============================================================================
Memory usage summary:
  Total non-mmapped bytes (arena):       4882432
  Bytes in mapped regions (hblkhd):      30130176
  Total allocated space (uordblks):      4658416
  Total free space (fordblks):           224016
  Topmost releasable block (keepcost):   66208
===============================================================================
Packet I/O Totals:
   Received:           17
   Analyzed:           15 ( 88.235%)
    Dropped:            0 (  0.000%)
   Filtered:            0 (  0.000%)
Outstanding:            2 ( 11.765%)
   Injected:            0
```

```
===============================================================================
Packet I/O Totals:
   Received:           17
   Analyzed:           15 ( 88.235%)
    Dropped:            0 (  0.000%)
   Filtered:            0 (  0.000%)
Outstanding:            2 ( 11.765%)
   Injected:            0
===============================================================================
Breakdown by protocol (includes rebuilt packets):
        Eth:           15 (100.000%)
       VLAN:            0 (  0.000%)
        IP4:           13 ( 86.667%)
       Frag:            0 (  0.000%)
       ICMP:            0 (  0.000%)
        UDP:            3 ( 20.000%)
        TCP:           10 ( 66.667%)
        IP6:            0 (  0.000%)
    IP6 Ext:            0 (  0.000%)
   IP6 Opts:            0 (  0.000%)
      Frag6:            0 (  0.000%)
      ICMP6:            0 (  0.000%)
       UDP6:            0 (  0.000%)
       TCP6:            0 (  0.000%)
     Teredo:            0 (  0.000%)
    ICMP-IP:            0 (  0.000%)
    IP4/IP4:            0 (  0.000%)
    IP4/IP6:            0 (  0.000%)
    IP6/IP4:            0 (  0.000%)
    IP6/IP6:            0 (  0.000%)
        GRE:            0 (  0.000%)
```

This is the log file generated after doing the curl command. In this basically alert is generated. To open this file i used the below command:

- /var/log/snort cat snort.log

```
(◆l@4m◆}◆b
P◆◆
66RUb◆P◆◆◆◆◆◆◆gze
'1◆(@@◆◆
66RU◆◆P◆◆b◆P`@◆◆◆gg
'1◆(@@◆◆
<'1◆URU◆◆b◆P`?◆◆◆g!◆
E(◆m◆us◆}◆b
P◆◆◆◆b◆P`/◆◆◆g7◆<'1◆URU
,◆r@◆◆◆R◆
P◆◆
   V◆◆ ◆`◆◆◆◆◆◆◆◆g◆<'1◆URU
(◆s@◆◆◆R◆
P◆◆
   V◆◆!&P◆◆◆◆◆◆g3
                s'1◆URU
e◆t@◆◆◆R◆
P◆◆
   V◆◆!&P◆◆/bHTTP/1.1 301 Found
Location: https://www.craigslist.org/

◆◆◆3
    <'1◆URU
(◆u@◆◆◆R◆
P◆◆
   V?◆◆!&P◆◆◆◆◆◆◆g0B
                  <'1◆URU
(◆v@◆◆◆R◆
P◆◆
   V@◆◆!'P◆◆◆◆harshu@harshu-VirtualBox:/var/log/snort$
```

And below is the plain text using tcpdump

```
03/26-15:21:47.966856 208.82.238.1:80 -> 10.0.2.15:60704
TCP TTL:64 TOS:0x8 ID:50957 IpLen:20 DgmLen:44
***A**S* Seq: 0x191EAA01  Ack: 0xBAF4A934  Win: 0xFFFF  TcpLen: 24
TCP Options (1) => MSS: 1460

[**] [1:1000001:0] Craiglist Accessed via HTTPS/HTTP [**]
[Priority: 0]
03/26-15:21:47.967773 208.82.238.1:80 -> 10.0.2.15:60704
TCP TTL:64 TOS:0x8 ID:50958 IpLen:20 DgmLen:40
***A**** Seq: 0x191EAA02  Ack: 0xBAF4A985  Win: 0xFFFF  TcpLen: 20

[**] [1:1000001:0] Craiglist Accessed via HTTPS/HTTP [**]
[Priority: 0]
03/26-15:21:48.231874 208.82.238.1:80 -> 10.0.2.15:60704
TCP TTL:64 TOS:0x8 ID:50959 IpLen:20 DgmLen:101
***AP*** Seq: 0x191EAA02  Ack: 0xBAF4A985  Win: 0xFFFF  TcpLen: 20

[**] [1:1000001:0] Craiglist Accessed via HTTPS/HTTP [**]
[Priority: 0]
03/26-15:21:48.231875 208.82.238.1:80 -> 10.0.2.15:60704
TCP TTL:64 TOS:0x8 ID:50960 IpLen:20 DgmLen:40
***A***F Seq: 0x191EAA3F  Ack: 0xBAF4A985  Win: 0xFFFF  TcpLen: 20

[**] [1:1000001:0] Craiglist Accessed via HTTPS/HTTP [**]
[Priority: 0]
03/26-15:21:48.232585 208.82.238.1:80 -> 10.0.2.15:60704
TCP TTL:64 TOS:0x8 ID:50961 IpLen:20 DgmLen:40
***A**** Seq: 0x191EAA40  Ack: 0xBAF4A986  Win: 0xFFFF  TcpLen: 20

harshu@harshu-VirtualBox:/var/log/snort$
```

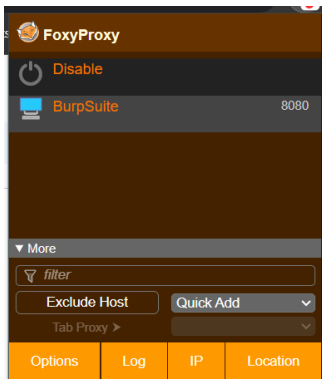
Successfully got the message “Craiglist Accessed via HTTPS/HTTP”

PART-II

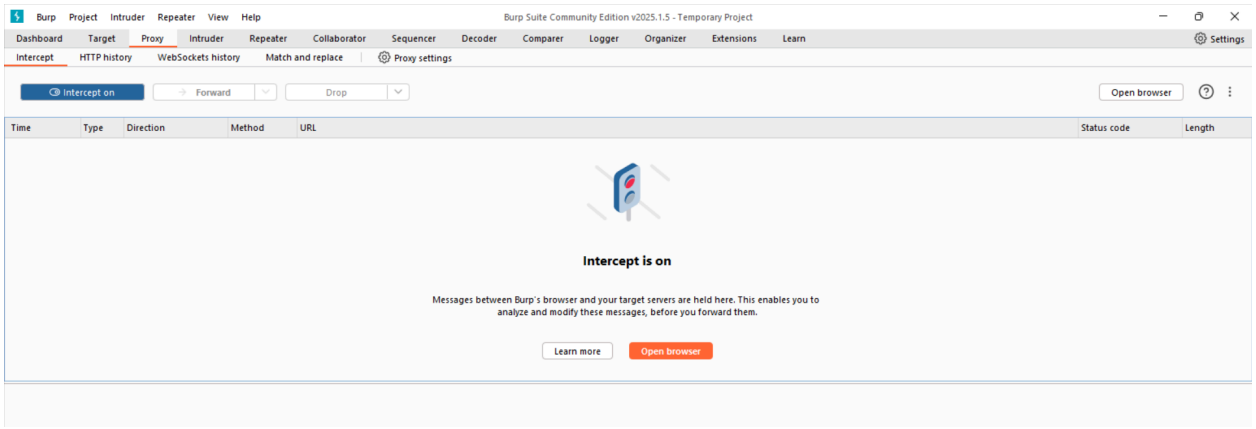
For this question, I have used BurpSuite, I have downloaded the BurpSuite community edition to play with requests and responses.
And the website I choose is:

- https://www.ncrpages.in/

Turn on the proxy server to use the burp suite.



Now turn on the intercept to intercept the request going from the computer to the website.

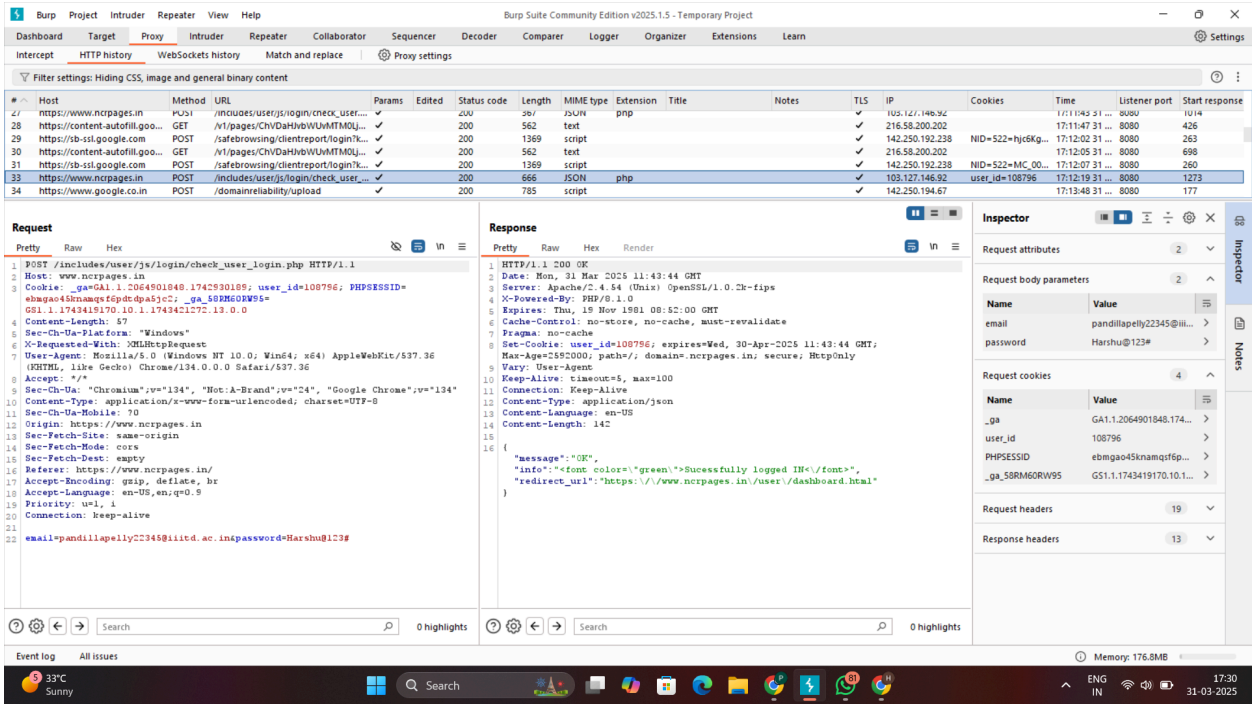


1. kind of information that can be captured while interacting are as follows:
First I have captured the information while logging in the info that i got is as follows:

a) Request Body Parameters
As we can see from the below screenshot the captured request body has the email and password, which are sent as form data for user authentication. As the website is using https request so the login request is encrypted via HTTPS means the credentials are sent in encrypted version using TLS. Then after filling the data is sent using HTTP POST request

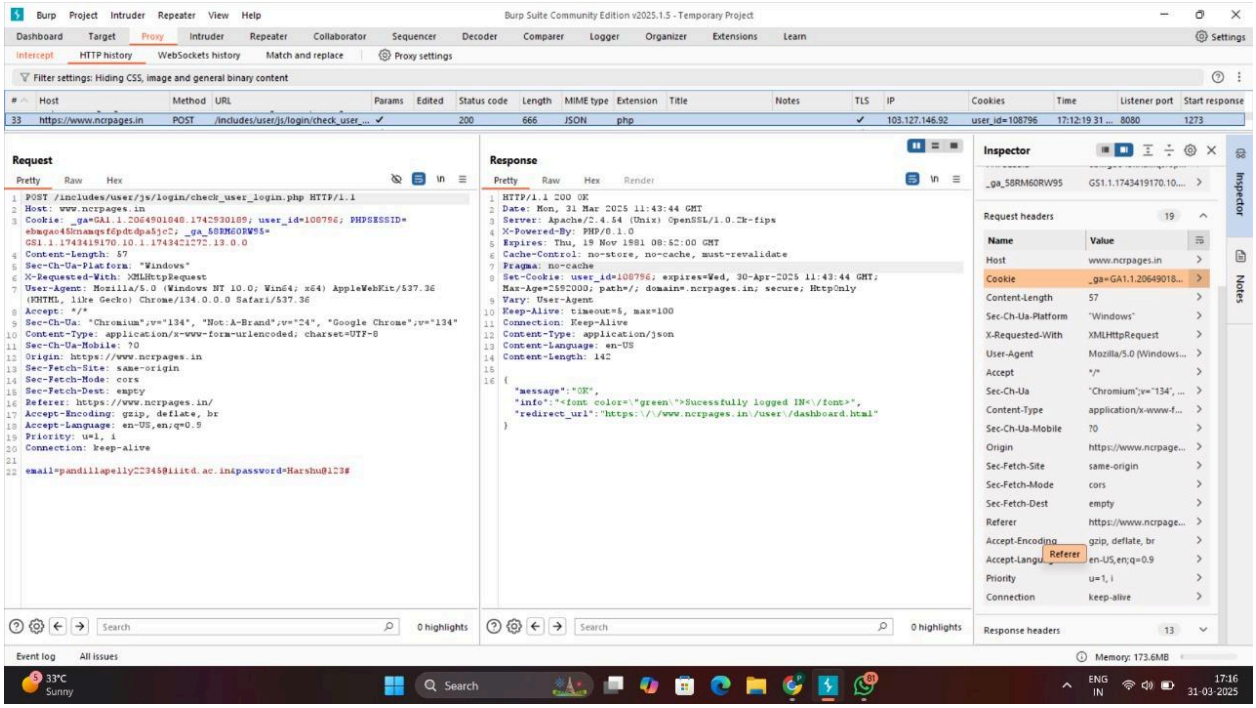
b) Request cookies
Some cookies like session identifiers, user tracking tokens, and unique user IDs were captured. The session token, PHPSESSID, is important because it handles the session between user and the web server. An attacker can steal the cookie if it is not properly secured using Httponly flags. Another tracking cookie from google analytics was there which monitors the user behaviour.

This is the below SS for above two data:



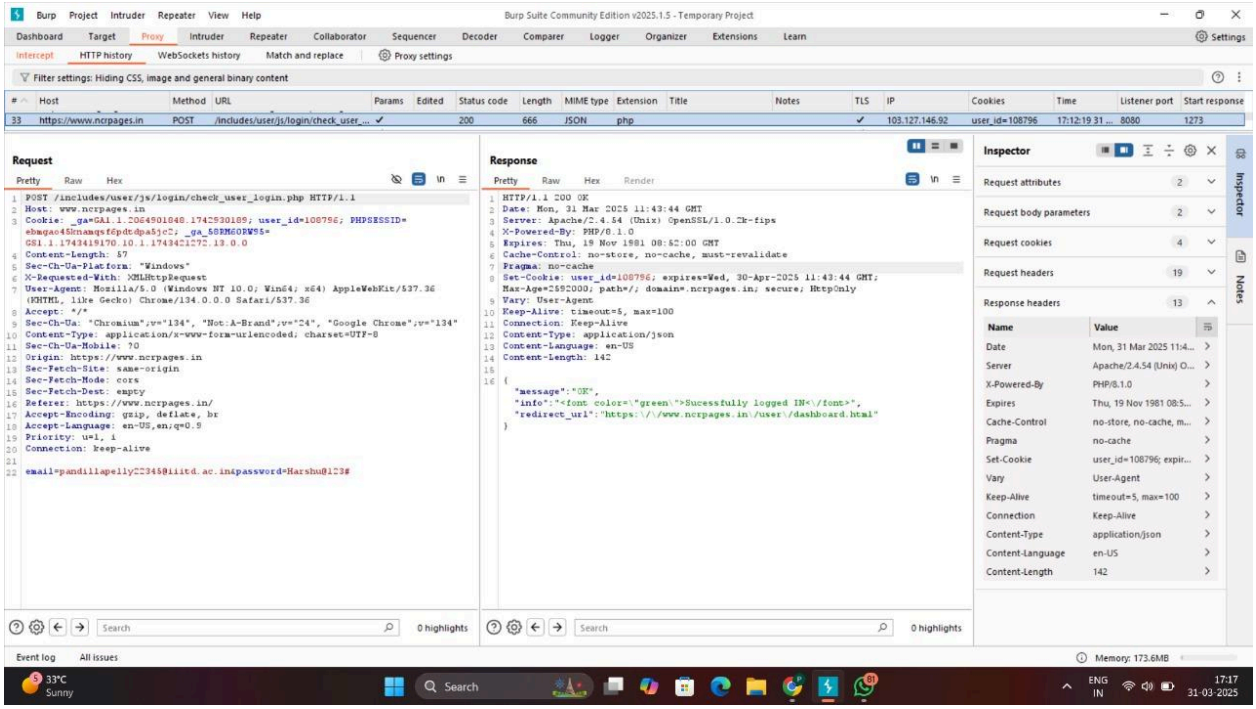
c) Request Headers

These headers captured the details like host, user-agent, referrer, content-type, and security related attributes. Browser and OS were revealed by the User-Agent and Sec-ch-us headers which aids in browser fingerprinting, and the referer header told about the source page(from where the request came from). And content-type header showed form data submission using application/x-www-form-urlencoded. This could be vulnerable to CSRF attacks if not properly secured. Below is the ss for this :

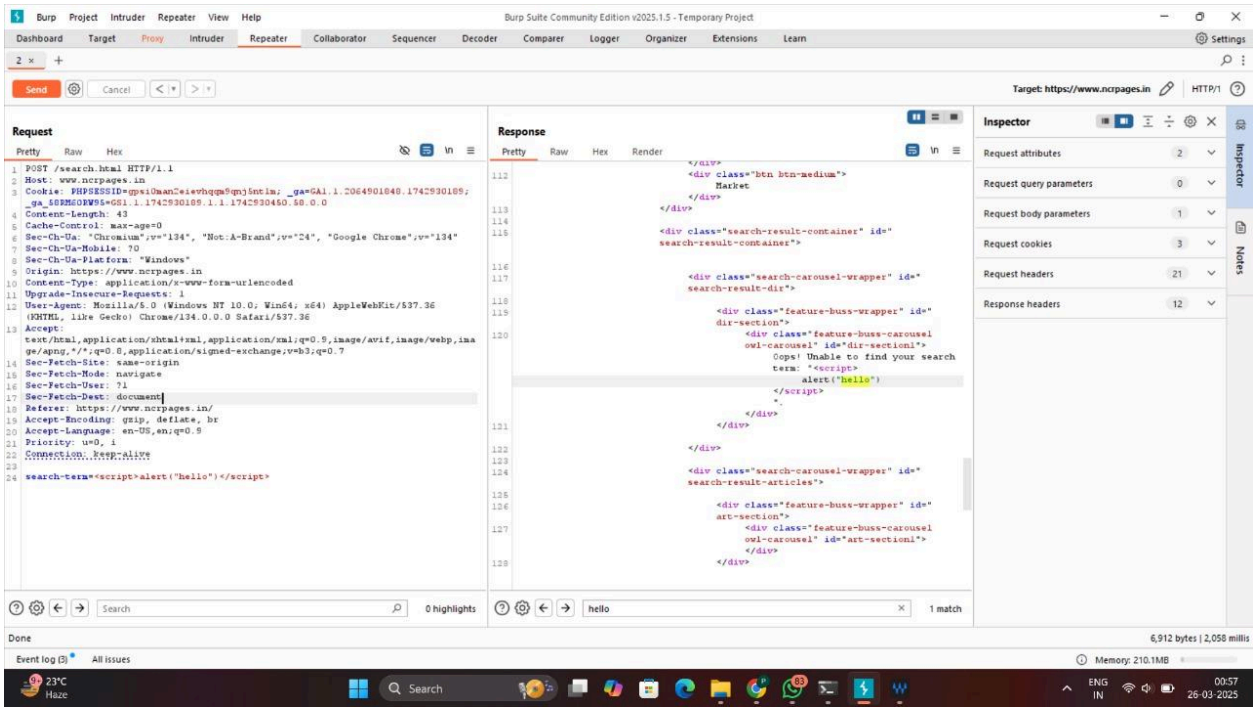


d) Response headers

This provided us with caching policies, session management and content information.



2. The Vulnerabilities i found in the websites are as follows:
i) From the data obtained the content-security-policy was missing which is the measure to protect the XSS attacks. Through Burp Suite I found that while giving the Script <script>alert(“hello”)</script> as input in the search section. In the response the website is not handling to prevent this alert, means in response directly <script>alert(“hello”)</script> is showing as you can see below:

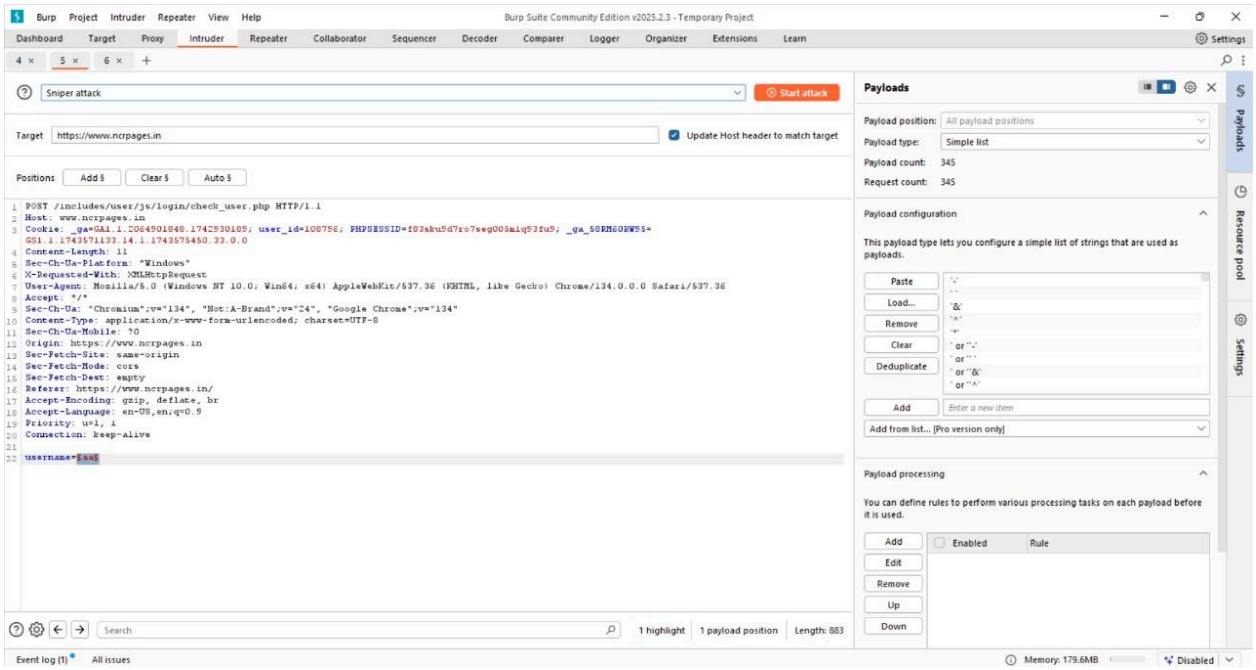


This can cause the website to run this script and we will see an alert in the website. This attack is called XSS (Cross-Site Scripting) attack where the goal is to inject Java-Script code in a web page so that it executes the arbitrary scripts in the user system. Secure websites use input sanitization and output encoding in which the websites encode the special characters like <, >, / to prevent the javascript code execution. For example

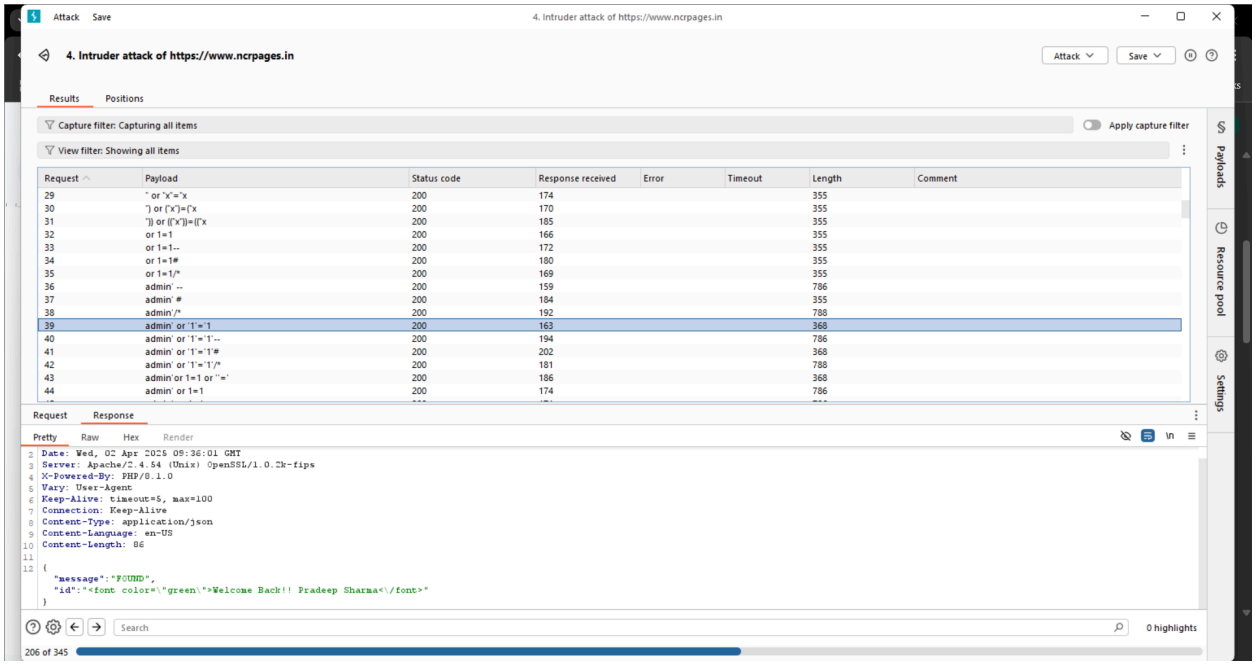
- User Input: <script>alert("hh")</script>
  Encoded Output: <script>alert("hh")</script>

ii) As there is lack of X-Content-Type-Options header in data i have obtained so SQL Injection attack might be possible. So, I have also tried SQL injection attacks on my website. In SQL injection, an attacker can have the access to data, manipulate database contents or it can take the access of admin. To perform this attack first I have opened the burpsuite and sent a login request with any temporary credentials and then I

sent this request to the Intruder to perform SQL injection attack. And then in the username section I have tried several payloads to find vulnerabilities as you can see in the below picture.



Then Started the attack. After the attack i found that for the “admin' or '1'='1” SQL query the website responded like this



From this SQL query we got the Admin name. Means The Sql query successfully executed means the sql injection attack was successful.

3.

a) This site uses the cookies that are used for tracking user behaviour (Google Analytics, Marketo, Facebook Pixel). This also detects anomalies, prevents fraud and mitigates DDoS attacks. These cookies enable secure and HttpOnly flags which prevent unauthorized cookie access and protect the attacks like MITM, phishing attacks etc.

b) This website uses HTTPS instead of HTTP, HTTPS encrypts communication between the browser and server, this prevents from intercepting sensitive information like login details and personal data from attackers. It also protects from MITM attack and eavesdropping by securing session identifiers.

4. (i) To prevent XSS attacks we can add the proper **Content-Security-Policy** (CSP) header to prevent Javascript execution and from loading malicious scripts. We can also add input validation and output encoding which sanitize user inputs by escaping <, >, ‘, “ these characters so that it prevents script execution. And Output encoding also prevents XSS attack by storing the special character in text format instead of interpreting it as code.

ii) We can use parameterized queries which are prepared statements, by using this prevents attacker from running malicious code. We can also use Input validation which rejects the special characters which the SQL has. By this it reduces the risk of execution of SQL queries.

PART-III

Downloading Tor

First installing the required dependencies

- sudo apt install -y build-essential libevent-dev libssl-dev zlib1g-dev \
  liblzma-dev libzstd-dev asciidoc autoconf automake \
  libtool pkg-config git

After this do the sudo apt update using below command

- sudo apt update
- sudo apt upgrade -y

Now Downloading source code of tor by cloning its github

- git clone https://git.torproject.org/tor.git

Now, go into tor directory

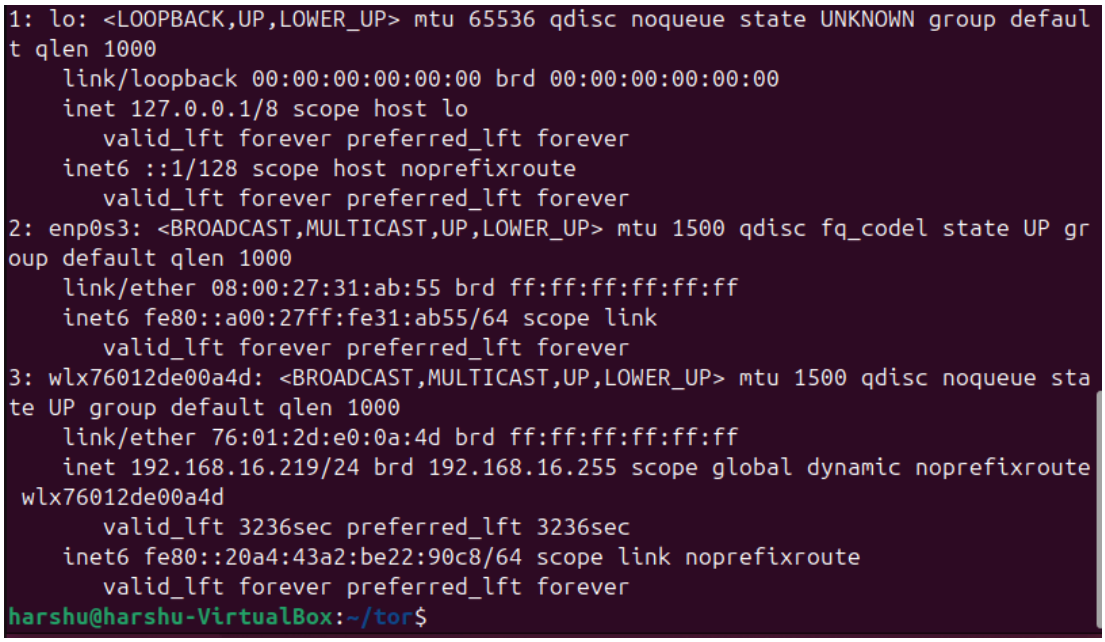
- cd tor

To build and compile the Tor i runned the following commands

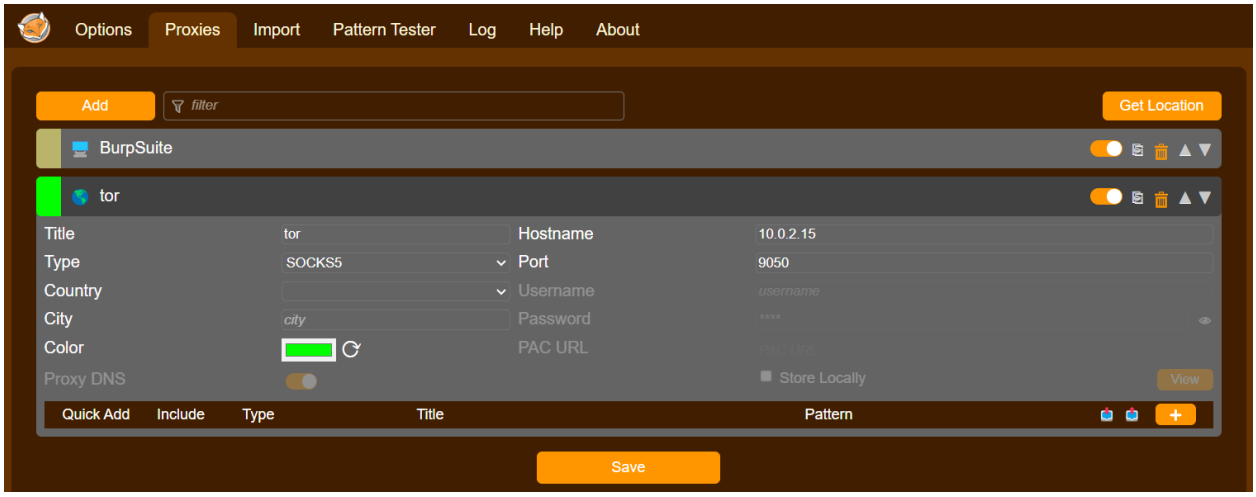
- ./autogen.sh
- ./configure --disable-asciidoc
- make -j$(nproc)
- sudo make install

As there were no wifi options in my linux so I needed to change the network settings. I changed it to Bridged only. And used the external ethernet for the wifi.

Now I am configuring my browser so that I can use Tor.
From the below screenshot we can see the ip address where we have setup the tor.



Now put this ip address in the HOSTname section.
And by default runs a SOCKS5 proxy. So set the type to SOCKS5.
And set the port to 9050



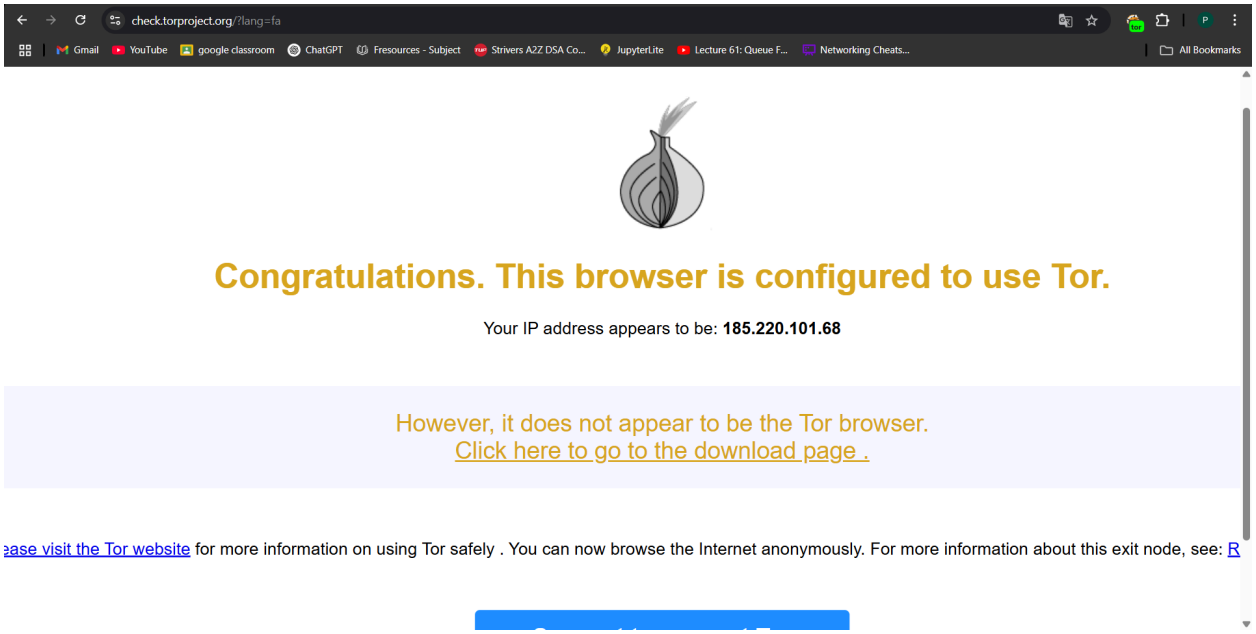
Now turn on the tor using this command

- Tor

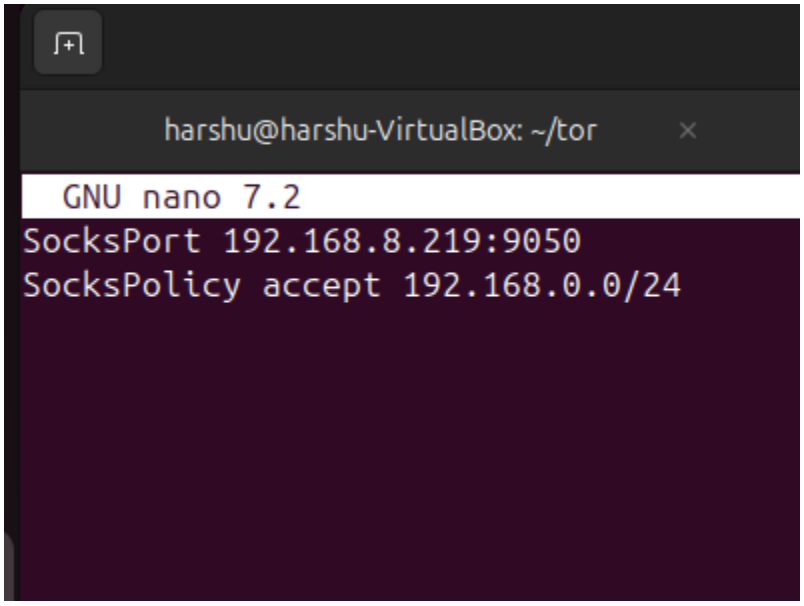
```
harshu@harshu-VirtualBox:~/tor$ tor
Apr 02 16:10:30.751 [notice] Tor 0.4.9.1-alpha-dev (git-9a1e633b27174f0e) runnin
g on Linux with Libevent 2.1.12-stable, OpenSSL 3.0.13, Zlib 1.3, Liblzma 5.4.5,
 Libzstd 1.5.5 and Glibc 2.39 as libc.
Apr 02 16:10:30.751 [notice] Tor can't help you if you use it wrong! Learn how t
o be safe at https://support.torproject.org/faq/staying-anonymous/
Apr 02 16:10:30.751 [notice] This version is not a stable Tor release. Expect mo
re bugs than usual.
Apr 02 16:10:30.753 [notice] Read configuration file "/usr/local/etc/tor/torrc".
Apr 02 16:10:30.760 [warn] You specified a public address '0.0.0.0:9050' for Soc
ksPort. Other people on the Internet might find your computer and use it as an o
pen proxy. Please don't allow this unless you have a good reason.
Apr 02 16:10:30.760 [notice] Opening Socks listener on 0.0.0.0:9050
Apr 02 16:10:30.766 [notice] Opened Socks listener connection (ready) on 0.0.0.0
:9050
Apr 02 16:10:30.000 [notice] Parsing GEOIP IPv4 file /usr/local/share/tor/geoip.
Apr 02 16:10:31.000 [notice] Parsing GEOIP IPv6 file /usr/local/share/tor/geoip6
.
Apr 02 16:10:31.000 [notice] Bootstrapped 0% (starting): Starting
Apr 02 16:10:32.000 [notice] Starting with guard context "default"
Apr 02 16:10:35.000 [notice] Bootstrapped 5% (conn): Connecting to a relay
Apr 02 16:10:36.000 [notice] Bootstrapped 10% (conn_done): Connected to a relay
Apr 02 16:10:36.000 [notice] Bootstrapped 14% (handshake): Handshaking with a re
```

Now checking in my system whether my browser is using Tor or not in this website.

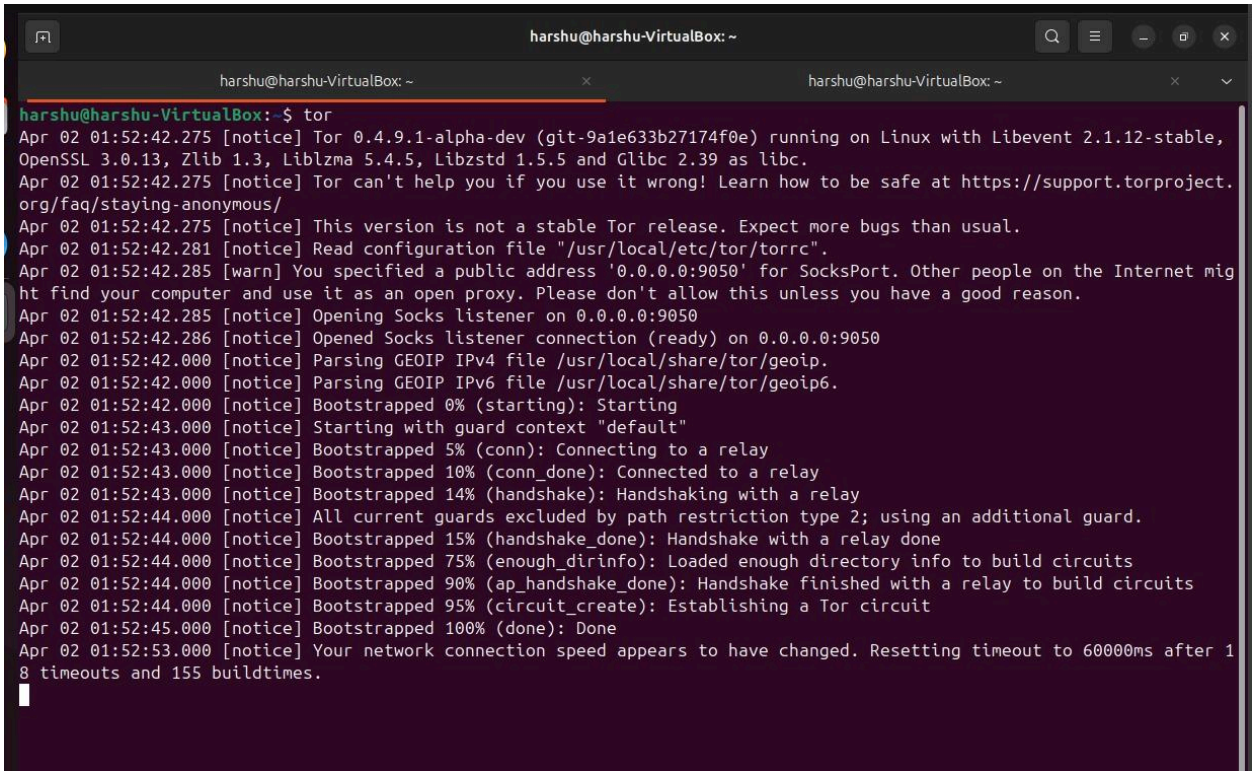
- https://check.torproject.org/?lang=fa



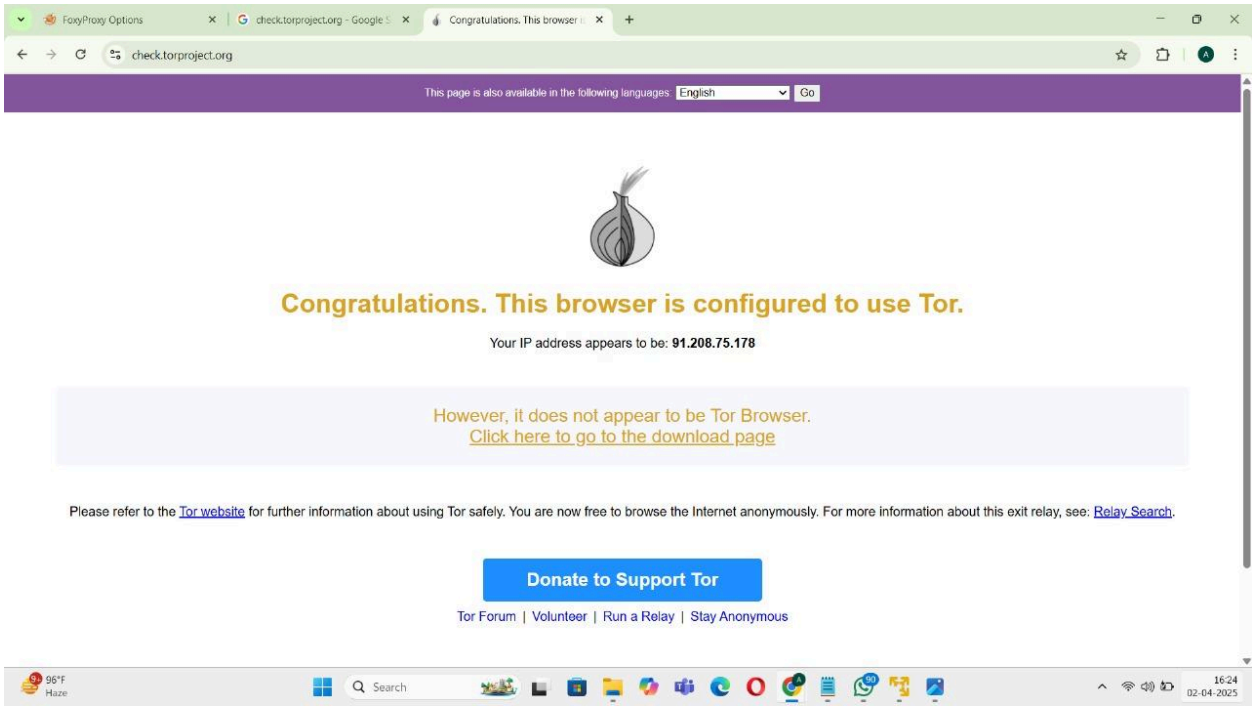
Now to accept connections from any computer on IIITD-LAN
First we need to change the torrc configuration file to allow IIITD LAN connections.

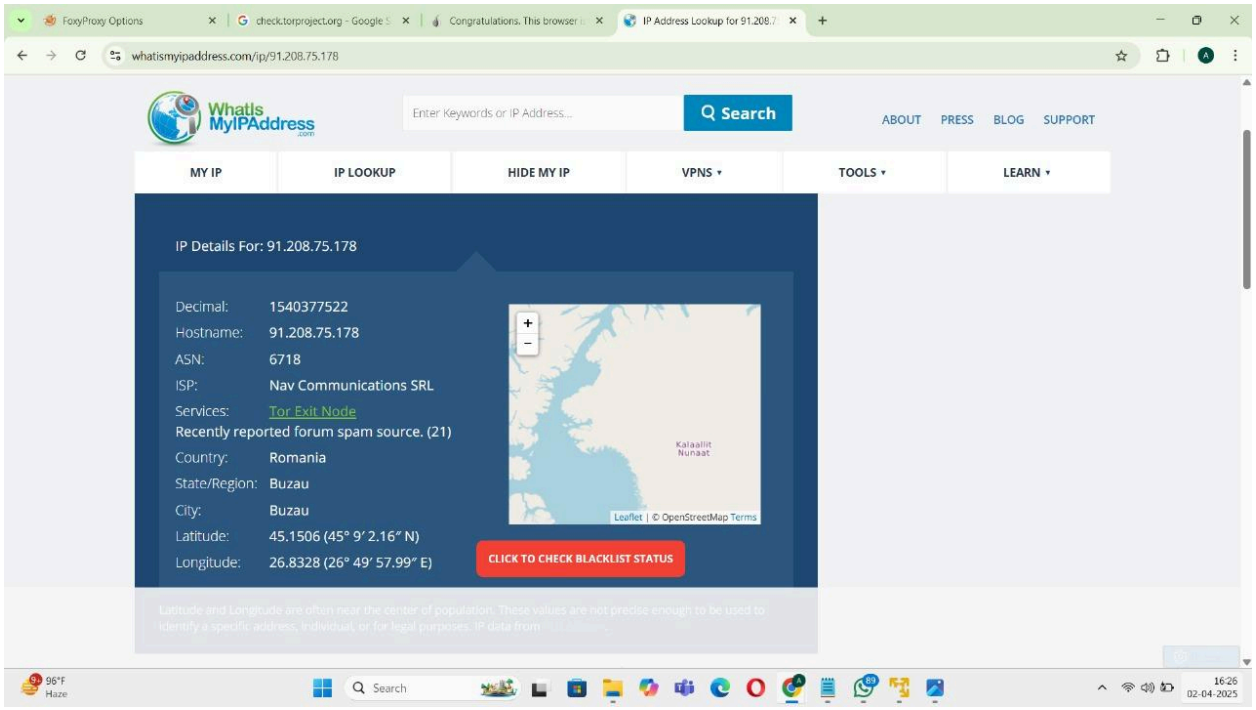


Restart the tor.

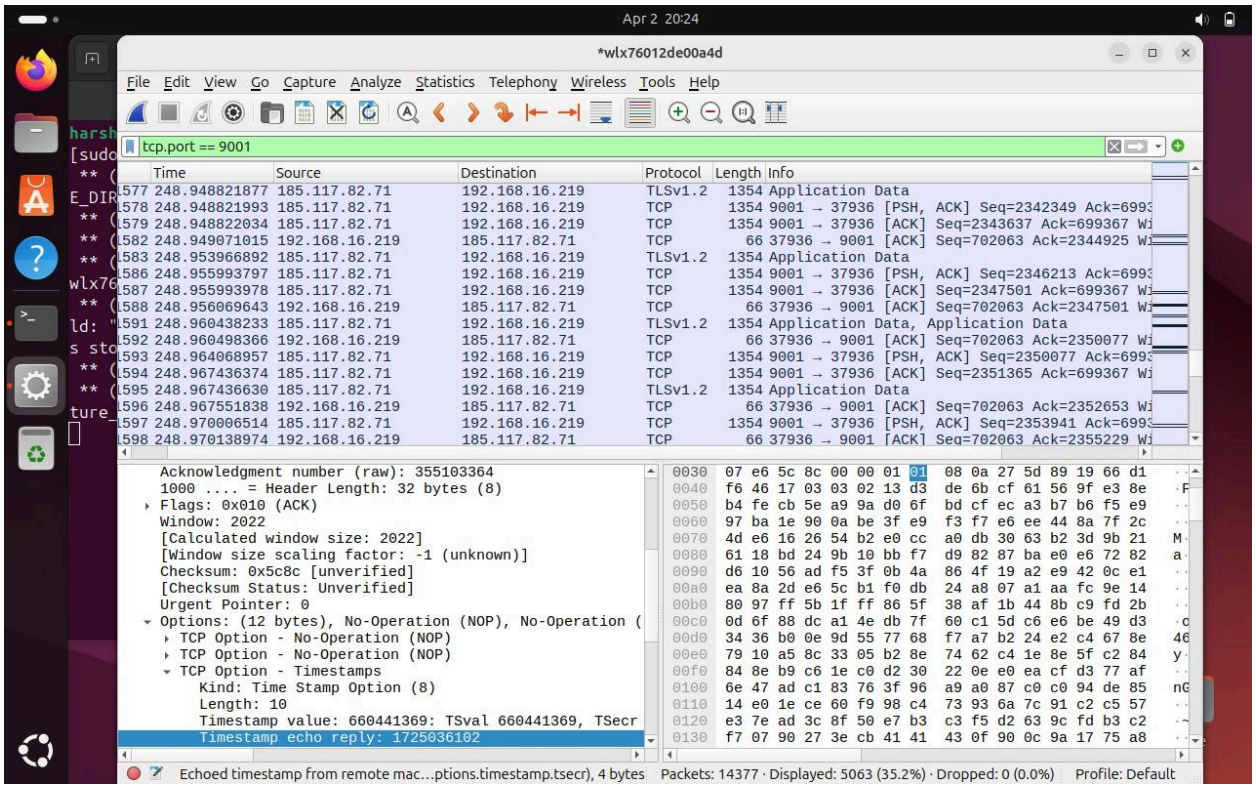


- Now to check whether this tor is allowing all the networks of IIITD LAN. I checked it from my friend's computer as he was connected to IIITD LAN only.
- First change the proxy settings of Friend's computer and put the IP of this where the tor is running in the Host name.
- Then my friend opened https://check.torproject.org/?lang=fa this link and from there it confirmed that my friend's browser is using Tor. As you can see in below screenshot





Then Searched some websites using Tor network.
And these were Intercepting tor packets with wireshark after searching websites on IIITD LAN.

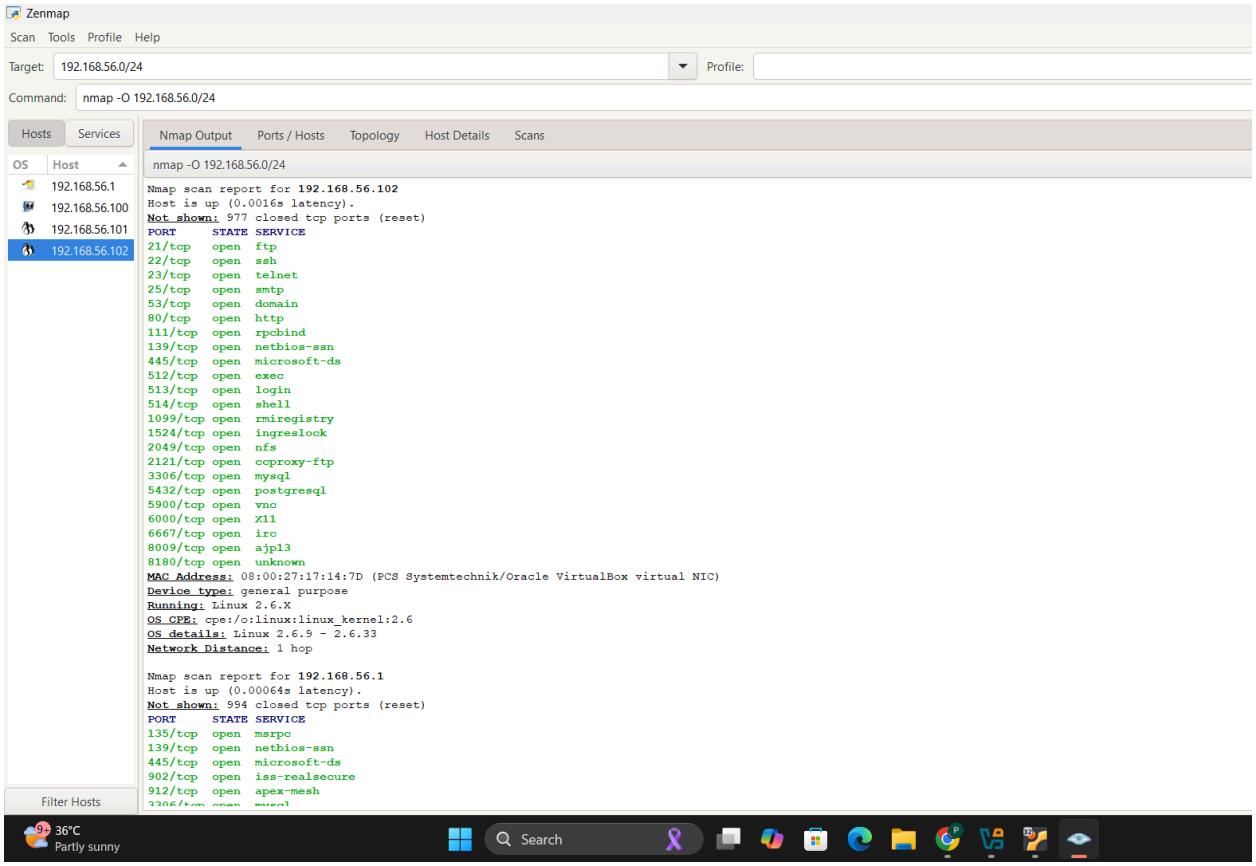


PART-IV

part(a)

After Downloading metasploitable-Linux successfully, I added this in my VitualBox and started. And note that in network settings I changed this into a Host-only adapter.
And also started the Ubuntu and that also set into Host-only adapter.

Now to extract all ip-addresses running behind. I have run this command "**nmap -O 192.168.56.0/24"**, What this will do is that it will check one by one from 192.168.56.1 to 192.168.56.255. And extract the ip addressed running behind and this give the reconnaissance of each ip addresses.



From the above IP addresses the IP address of metasploit system is 192.168.56.102. So we can check the OS version of the metasploit system in the 192.168.56.102 section.

From there we can see that OS version is: - Linux 2.6.9-2.6.33

- OS CPE - Linux_kernel:2.6

Run the same command in terminal.

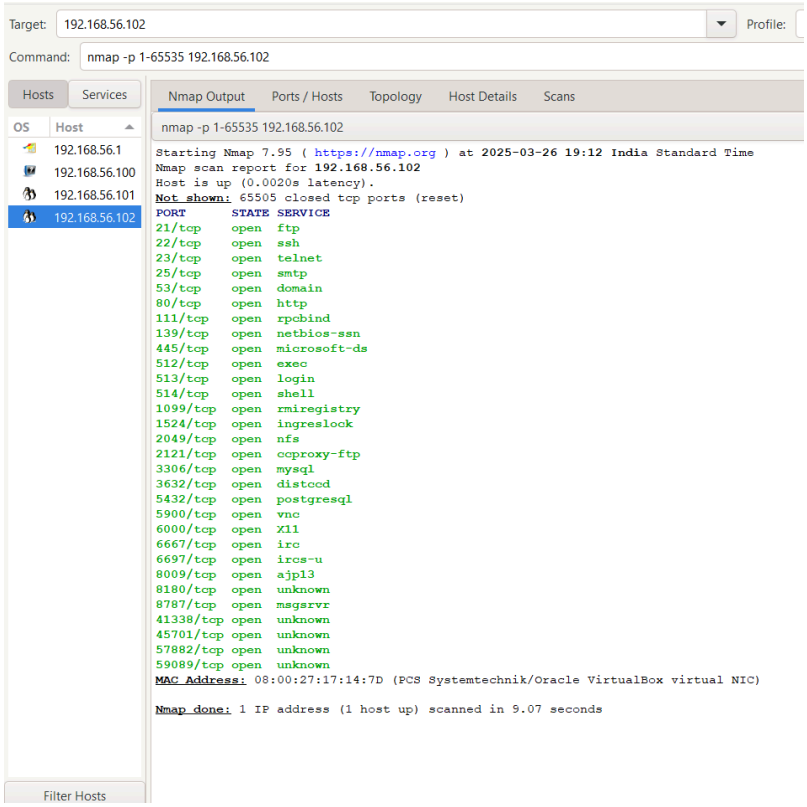
```
Nmap scan report for 192.168.56.102
Host is up (0.0017s latency).
Not shown: 977 closed tcp ports (reset)
PORT     STATE SERVICE
21/tcp   open  ftp
22/tcp   open  ssh
23/tcp   open  telnet
25/tcp   open  smtp
53/tcp   open  domain
80/tcp   open  http
111/tcp  open  rpcbind
139/tcp  open  netbios-ssn
445/tcp  open  microsoft-ds
512/tcp  open  exec
513/tcp  open  login
514/tcp  open  shell
1099/tcp open  rmiregistry
1524/tcp open  ingreslock
2049/tcp open  nfs
2121/tcp open  ccproxy-ftp
3306/tcp open  mysql
5432/tcp open  postgresql
5900/tcp open  vnc
6000/tcp open  X11
6667/tcp open  irc
8009/tcp open  ajp13
8180/tcp open  unknown
MAC Address: 08:00:27:17:14:7D (PCS Systemtechnik/Oracle VirtualBox virtual NIC)
Device type: general purpose
Running: Linux 2.6.X
```

```
MAC Address: 08:00:27:17:14:7D (PCS Systemtechnik/Oracle VirtualBox virtual NIC)
Device type: general purpose
Running: Linux 2.6.X
OS CPE: cpe:/o:linux:linux_kernel:2.6
OS details: Linux 2.6.9 - 2.6.33
Network Distance: 1 hop
```

part(b)

I used the following command to list the open ports on the metasploitable system:

- nmap -p 1-65535 192.168.56.102



From the above picture we can see that these following ports are open:
21, 22, 23, 25, 53, 80, 111, 139, 445, 512, 513, 514, 1099, 1524, 2049, 2121, 3306, 3632, 5432, 5900, 6000, 6667, 6697, 8009, 8180, 8787, 41338, 45701, 57882, 59089

Ports used for by default are mentioned below:

21- File transfer protocol(FTP) runs on this port, which is used for transferring the files between system over the network. This supports both active and passive modes.

22- SSH service runs on this port, this is the secure protocol for remote login and command execution in the another system. Commonly used for remote admin access to the servers.

23 - Telnet service runs on this port, it is the legacy protocol for remote access, this sends the data in plain text which is insecure.

25- Simple mail transfer protocol(SMTP) runs on this port, this is used to send emails between mail servers.

53- Domain name system (DNS) service runs on this port, this basically resolves the domain names into its IP addresses. Runs on both UDP and TCP.

80- HyperText Transfer Protocol (HTTP) service runs on this port, which is used for web browsing. This transfers webpages in plaintext.

111 - (RPCbind) Maps Remote Procedure Call (RPC) services to their corresponding ports.

139- This port is for NetBIOS-SSN, which is used for windows file sharing and communication between two two Windows devices. (This is mainly for older Windows systems)

445- This port is for Microsoft-DS services, same as port 139 port but this is for modern Windows systems. This allows shared drives, folders etc over network.

512- It is for exec services, this executes commands remotely on a Linux system. But this is rarely used because of security reasons.

513- This port is to handle login purposes, this provides the remote login functionality. Insecure.

514- This port is for shell commands, this allows remote command execution in this no authentication is required.

1099- This port is for RMI Registry, Which is used for remote method execution for java applications.

1524- This is for ingreslock, which is originally for Ingres database connections. Nowadays it is a backdoor for attackers.

2049- This port is for NFS (Network File System), which is used for sharing files between Linux systems over network. Also allows the remote directories to be mounted if they were local.

2121- this is used for CCProxy-FTP.

3306 - This port is for MySQL services, it is the default port for MySQL database server. This is used to store and mange structured data.

3632- This port is for Distributed Compiler Daemon services, which allows parallel processing of multiple machines. Used in software development.

5432- PostgreSQL runs on this port, it is similar to mysql but it uses advanced features and scalability.

5900- for Virtual Network Computing(VNC), used for remote access of desktop. This allows screen sharing and remote control.

6000- This is for X11 (X Window System), which displays the graphical applications over the network. Used mainly in Linux based systems.

6667- **IRC (Internet Relay Chat)** – Chat server communication.

8009- Apache JServ Protocol (AJP13) service runs on this port, It is the protocol which is used to connect Apache HTTP Server to a servlet container like Apache Tomcat.

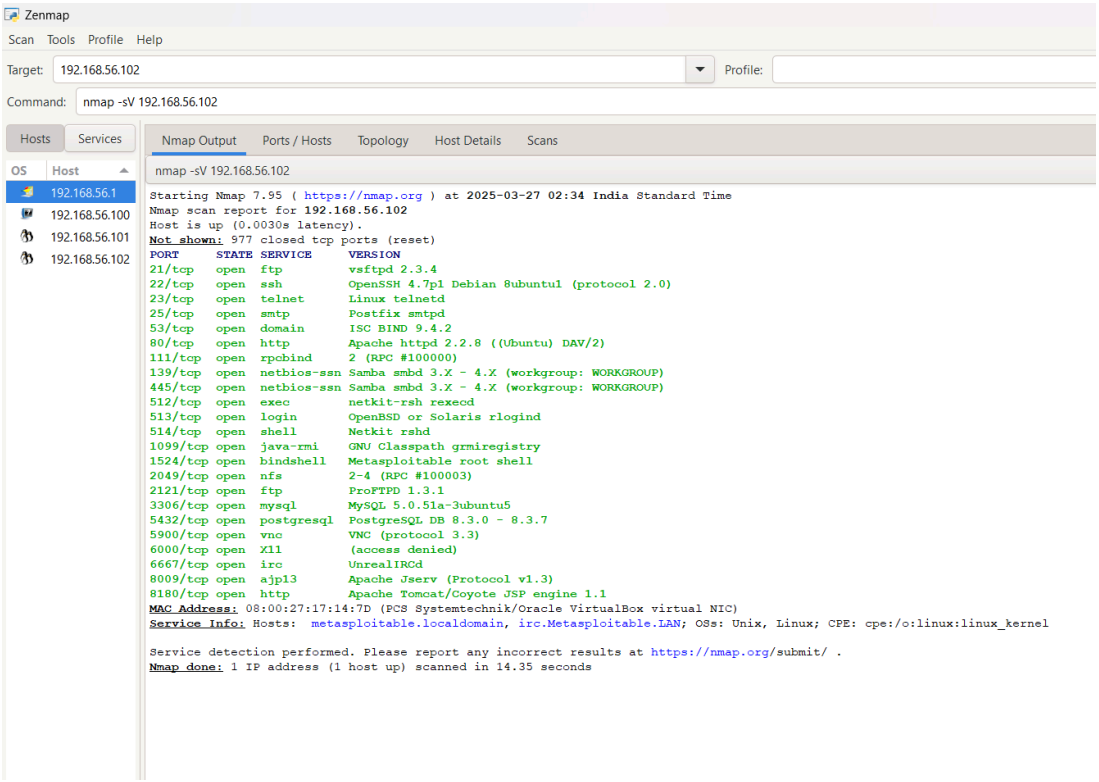
8180- This port is for Tomcat Web Server which is an alternative HTTP port for Apache Tomcat, which is a Java web application server. Sometimes used for testing or development environments.

8787- This is something related to RStudio Server, which runs on this port for remote R programming.

Remaining Ports - Unknown

We can find the applications running on the open ports by the following command:

- Nmap -sV 192.168.56.102



part(c)

Background of the attack-

- FTP is basically a protocol which is used for transferring files between systems on a network. Upon finding about version 2.3.4 it was found that it has a backdoor vulnerability. This Vulnerability allows attackers to gain remote access of shell.
- By using the username ending with :) the exploit is triggered and opens the shell on port 6200.

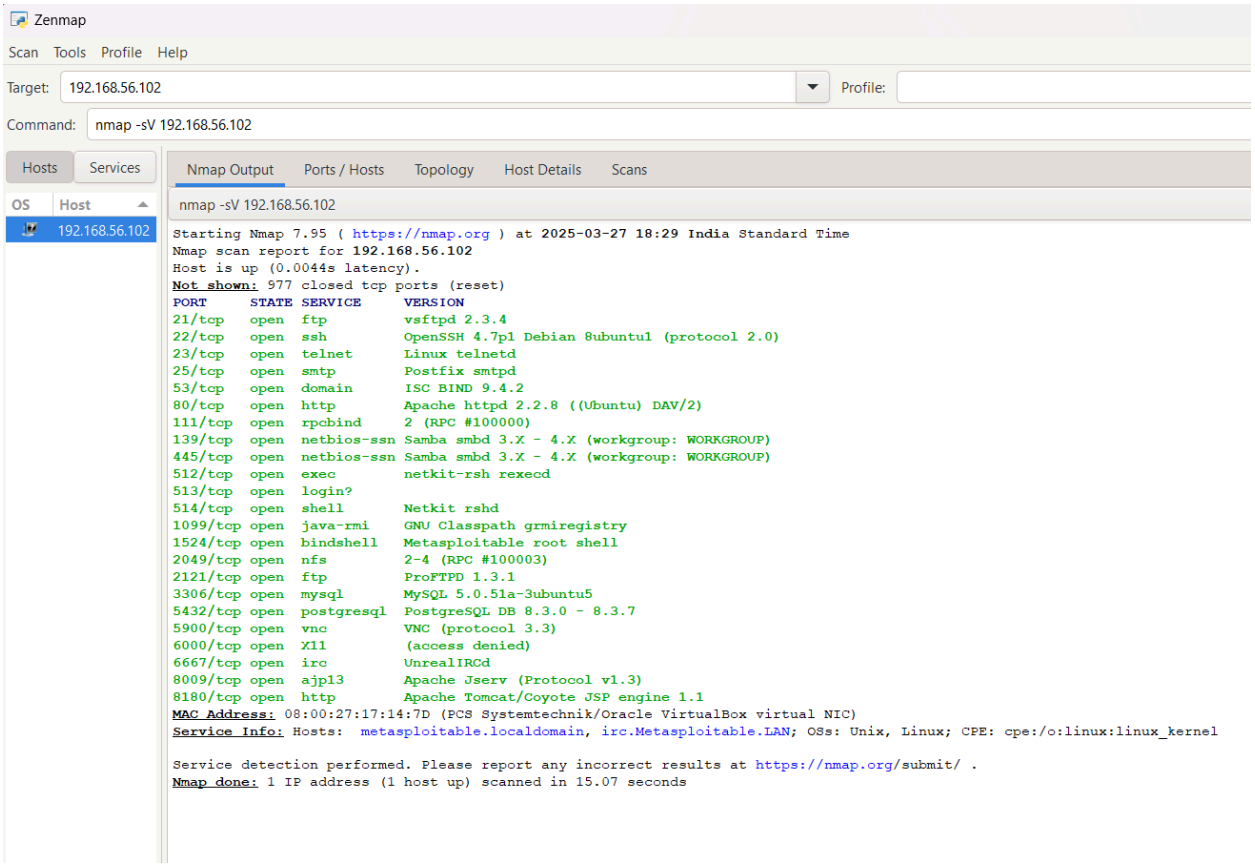
- The outcome of this attack is that the attacker can have access of the root shell on the target machine and be able to run commands on the target machine.

Steps Followed and their Screenshots:

Confirming that FTP is running on target system(metasploitable system) or not

Run the Nmap command:

- nmap -sV 192.168.56.102



The above output in the image tells that port 21 is open at the target machine and also running vsftpd 2.3.4(which is a FTP server). This indicates that the system is actively listening the incoming FTP connections. As vsftpd 2.3.4 is running on port 21 which can allow the unauthorized access to the system.

Now we try to exploit the FTP server

Downloaded Metasploit-framework using the below command:

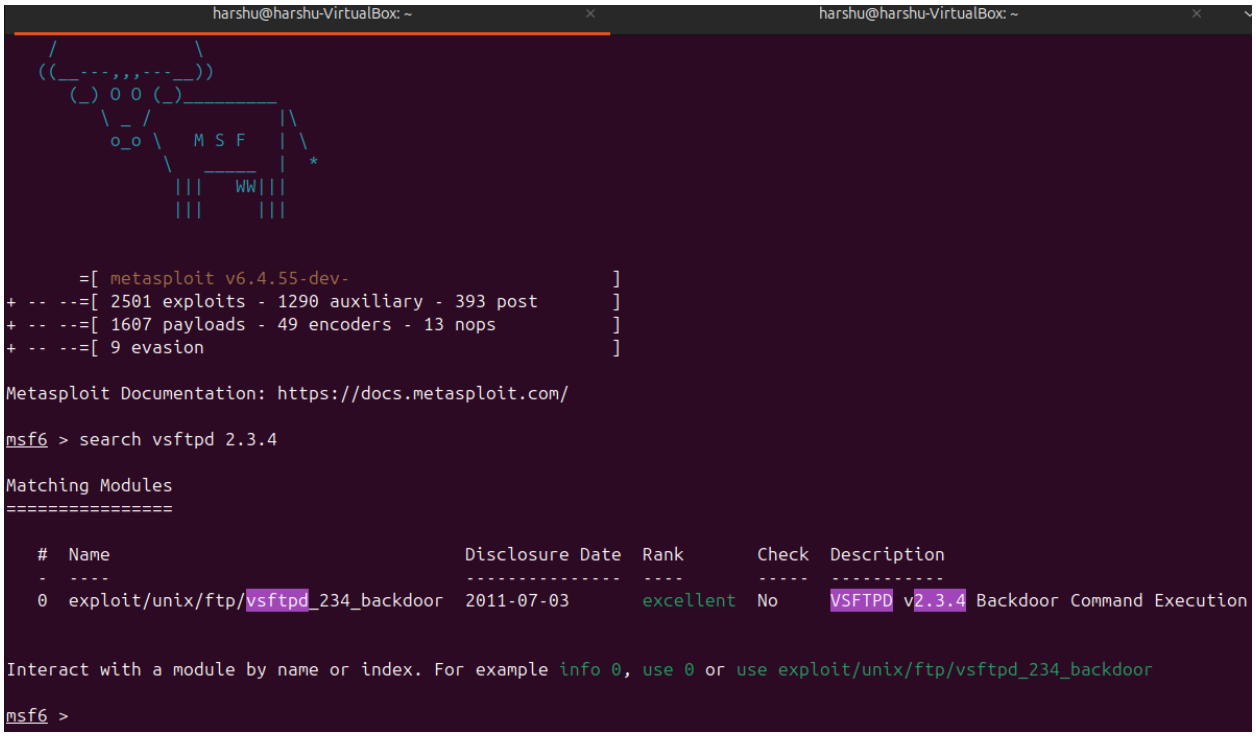
- Curl https://raw.githubusercontent.com/rapid7/metasploit-omnibus/master/config/templates/metasploit-framework-wrappers/msfupdate.erb > msfinstall && chmod 755 msfinstall && ./msfinstall

Next to start metasploit i used the below command

- Msfconsole

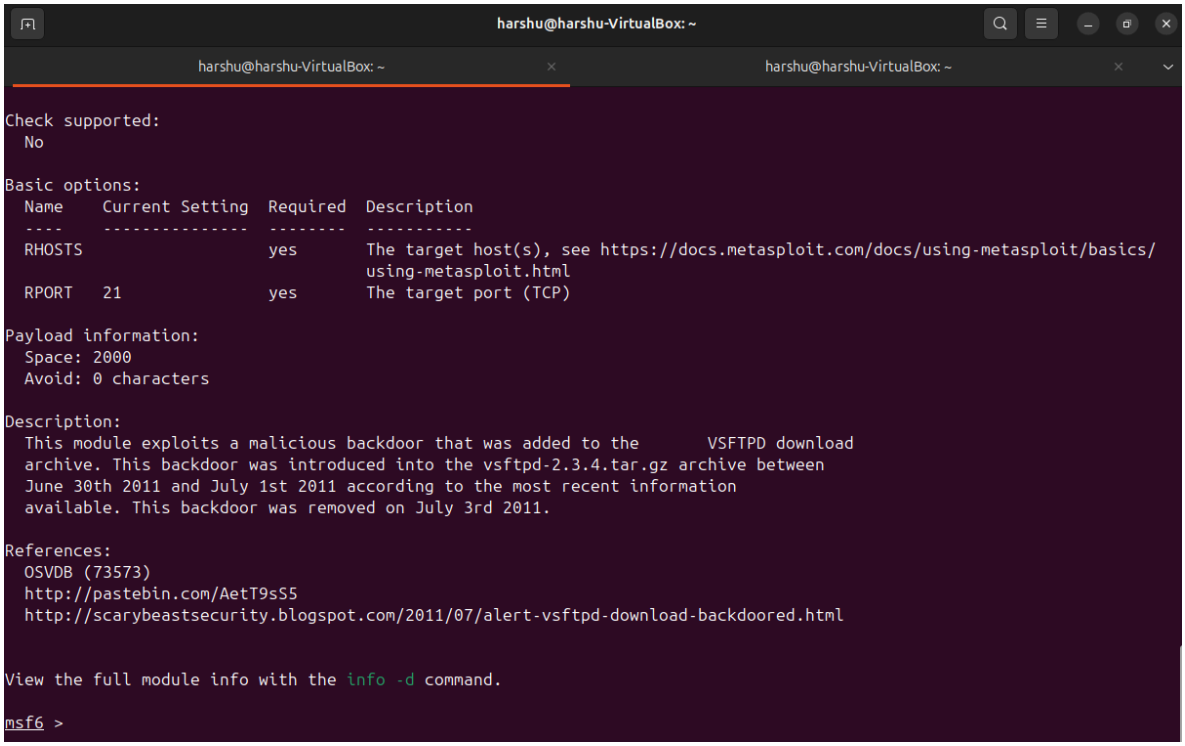
To find for an exploit for vsftpd 2.3.4 I used this command

- Search vsftpd 2.3.4

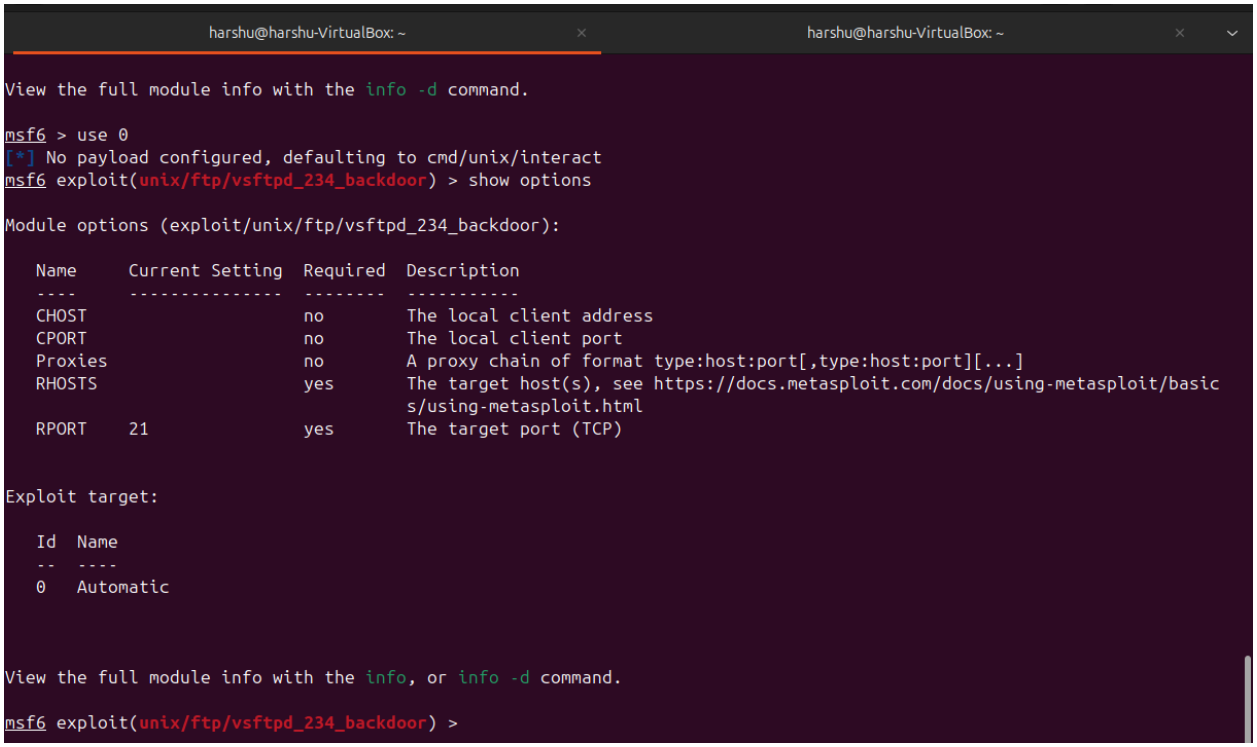


From the above screenshot we can see that it returned the exploit module named exploit/unix/ftp/vsftpd_234_backdoor which is used to exploit the backdoor was accidentally introduced in vsftpd in version 2.3.4.

Ran the command info 0(as given in the above ss) to extract more info about the module as you can see in below ss:



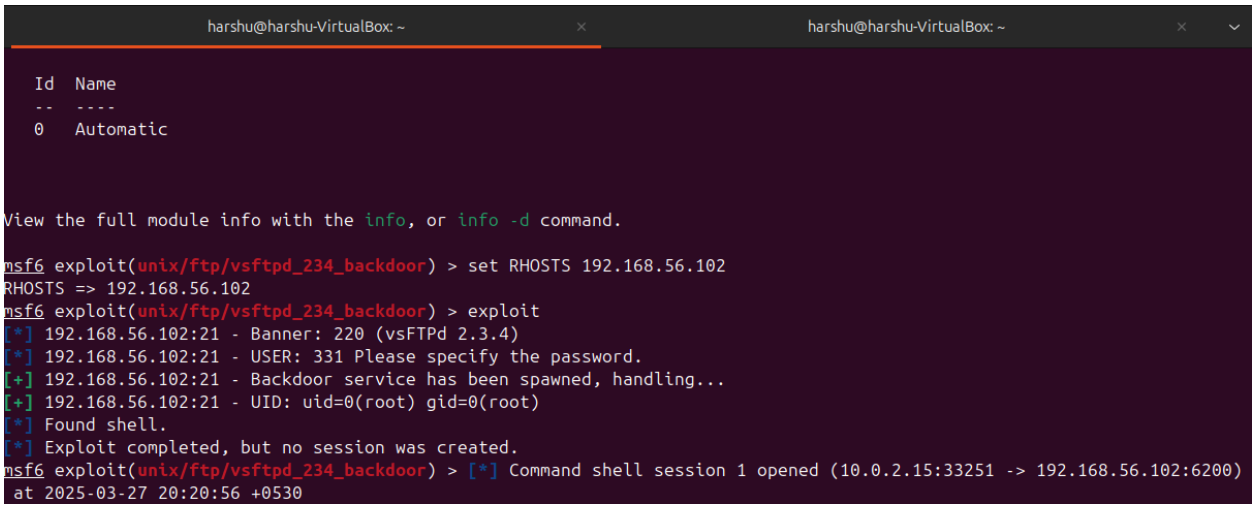
Now using the exploit with the command “use 0”
Then I checked the required options with command “show options” to confirm that everything is configured correctly.



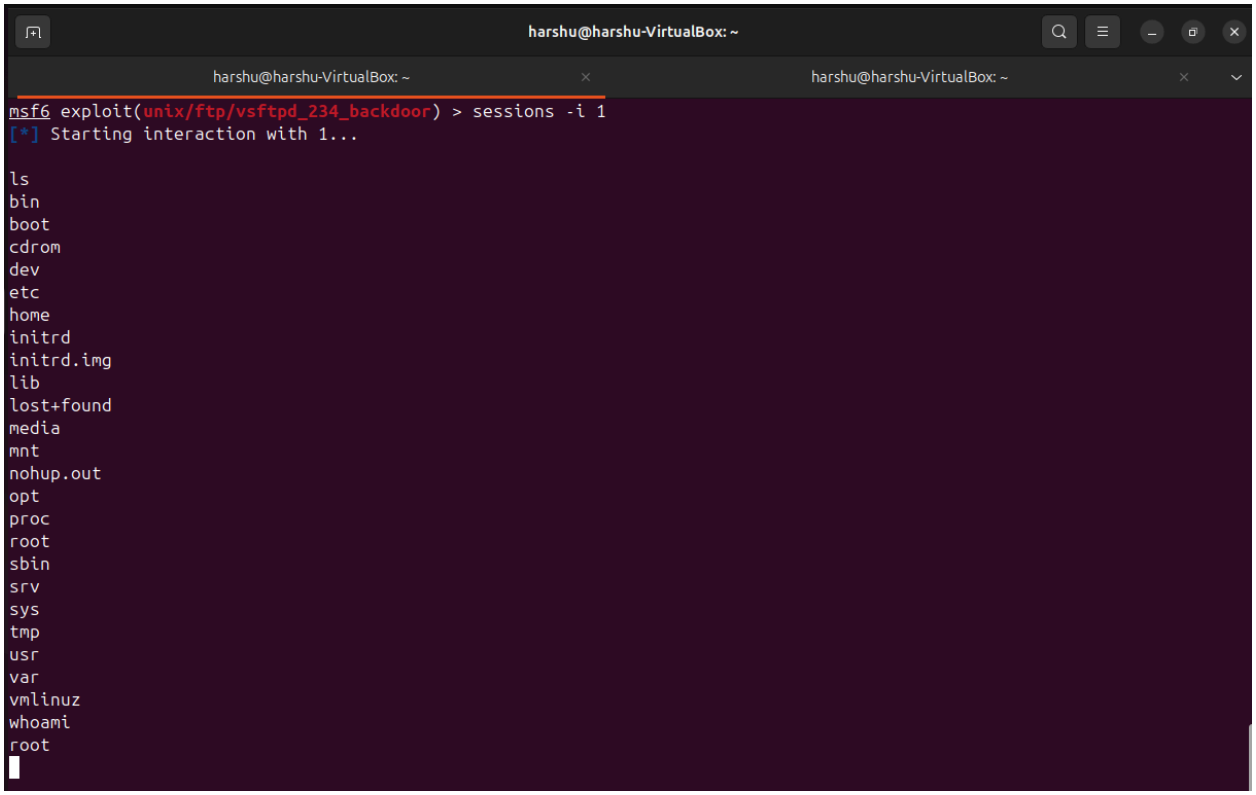
From above ss, the only required option is the target ip address. Now set the IP address of Metasploitable vm.

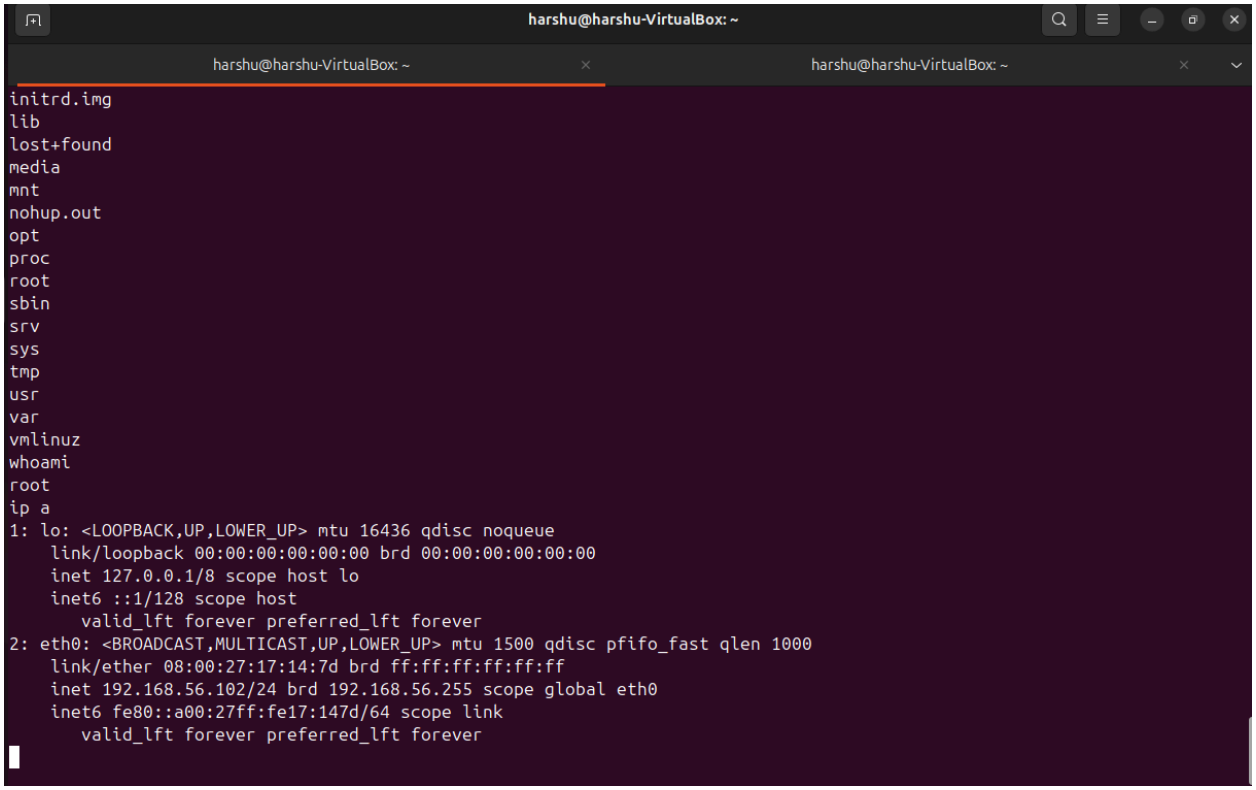
- Set RHOSTS 192.168.56.102

After this, as everything is configured correctly, now its time to run the “exploit” command

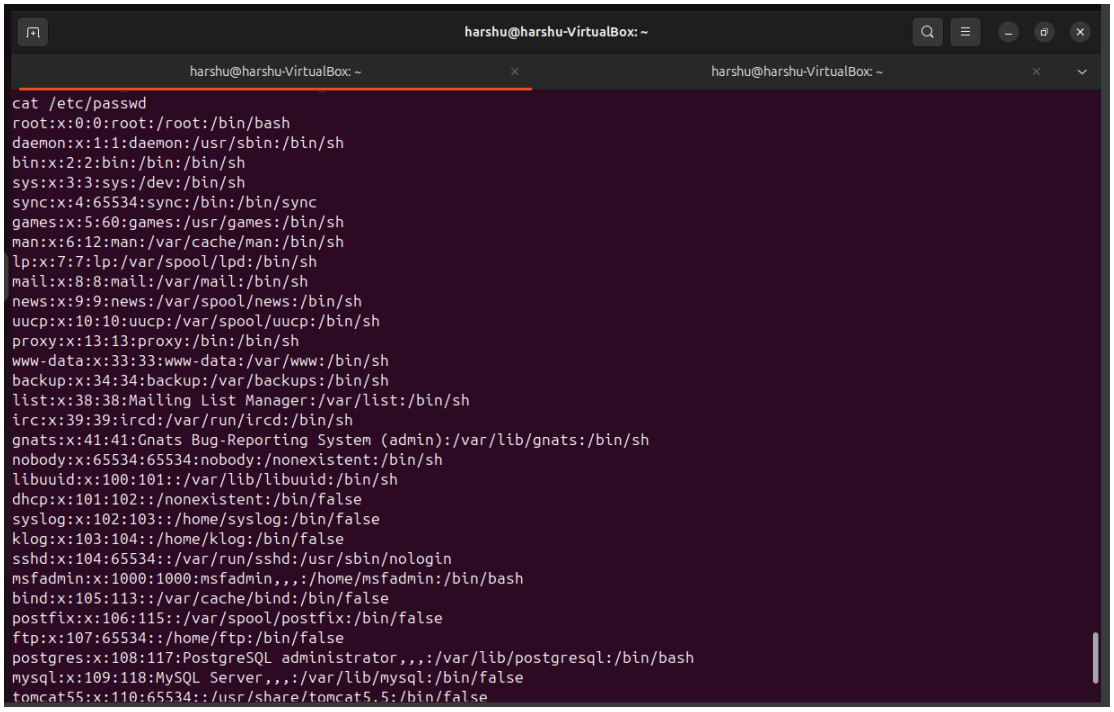


Gained the Shell Access successfully, Now Exploring the system





Now accessing /etc/passwd file, this is a critical file system which contains user account information with all the IDS. By accessing this file an attacker can use all the user data to start targeted attacks.



part(d)

Background of the attack:

- CSRF breaks the trust on the user's browser that a web application has on it which allows attackers doing unauthorized actions as the authenticated user.
- Victim got tricked by an attacker, in this attacker submits the malicious request secretly(from a hidden form or link) and gets authenticated.
- By this, an attacker can change the content of a website, do unauthorized actions(like adding a blog post) which can lead to security risks.

Steps to perform the Attack

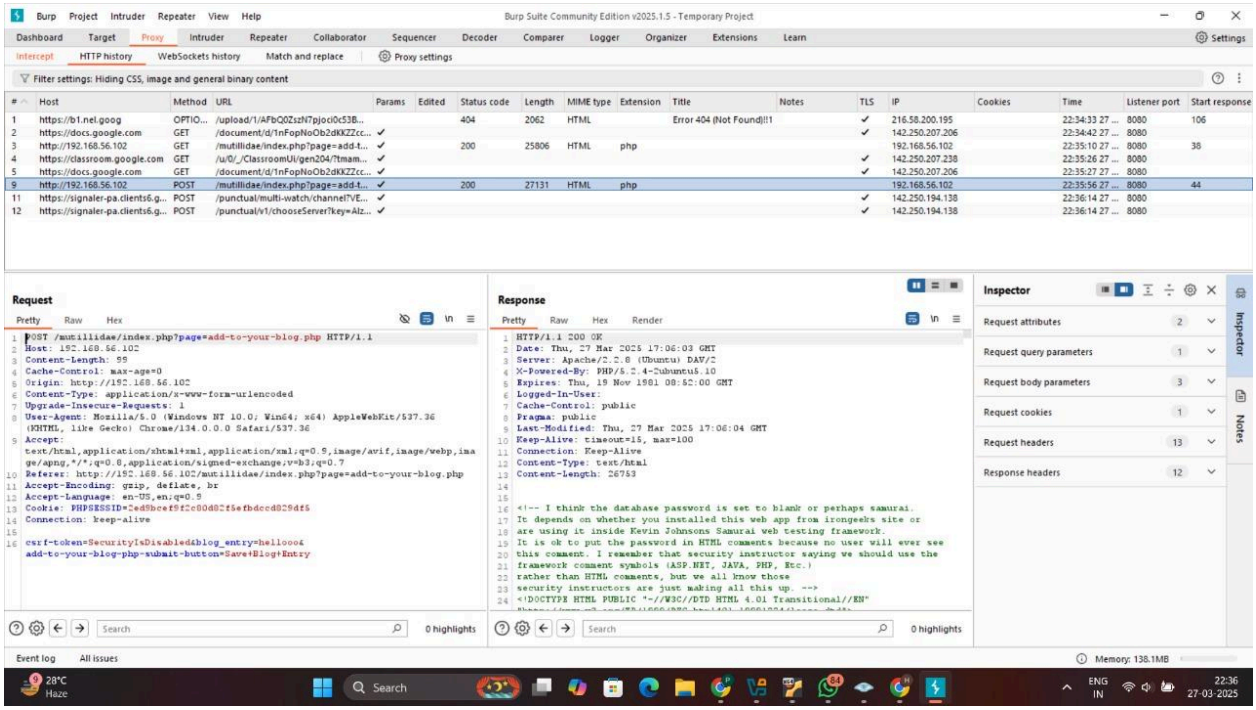
Through nmap it was found that it was open on port 80 means using http

First open browser and go to the following url:

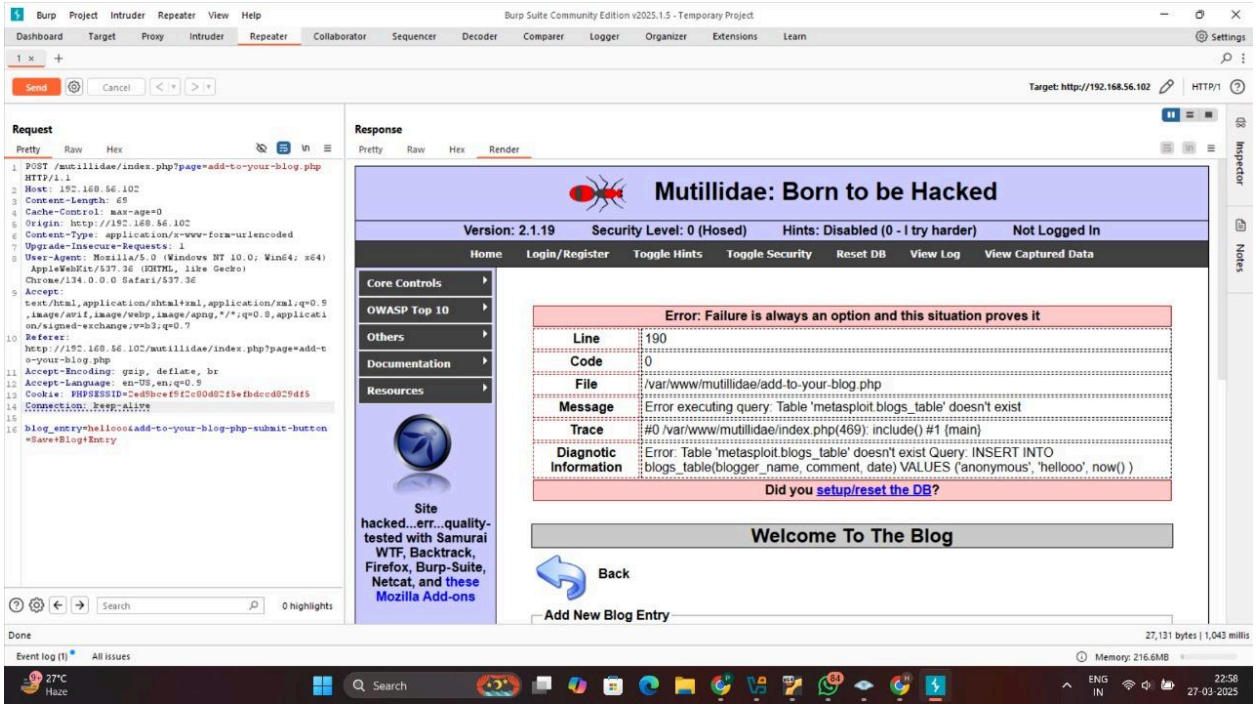
- http://192.168.56.102/mutillidae

Then go to "Add Blog for Anonymous" page. And i typed the text "helloooo"

Now through BurpSuite the Request and the response was monitored.



After removing the CSRF token the response was captured(As shown in below ss). It was coming same as before(with CSRF token).



The Blog post is accepted even without csrf-token, this means that the application is vulnerable to CSRF. In this attacker can submit forged requests.

We can exploit this Vulnerability by creating a user1(Harshu) and user2(attacker). What user 2 will do is that.Or we can basically open mutillidae in two different browser. Through burp suite send a request to burp suite by adding a temp blog. Then from the burp suite we can change the request. In place of user1 userID “Harshu” we can change it to user2 ID i.e. “attacker” and change the message. Now we can see in the Harshu blog section that the attacker had added a message without the user's knowledge.