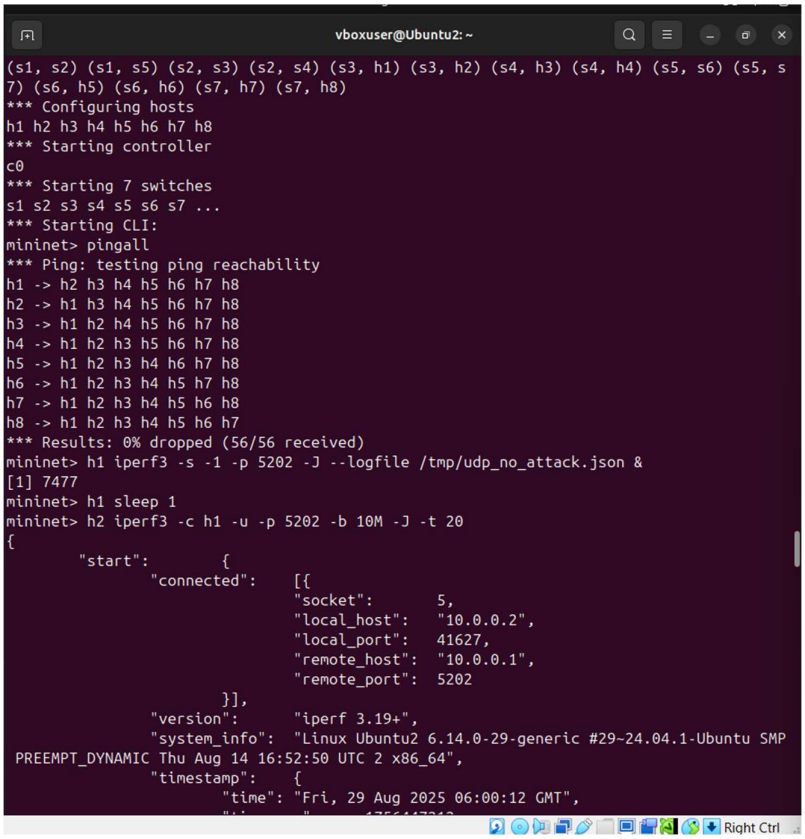
SIT325 Task 6.3D – SDN Performance Analysis Under UDP Flooding Attack

1. Experiment Setup

The experiment was conducted in a virtualized Ubuntu environment using **Mininet** to emulate a software-defined network.

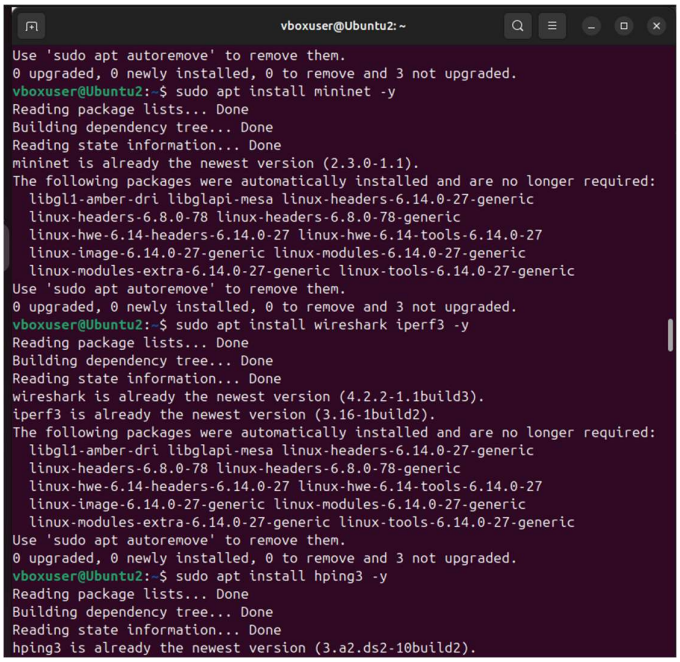
- **Topology:** A tree topology with depth=3 and fanout=2 was created (--topo tree,depth=3,fanout=2).

```
fvboxuser@Ubuntu2:~$ sudo mn --topo tree,depth=3,fanout=2 --controller=remote,ip=
b127.0.0.1,port=6653
*** Creating network
g*** Adding controller
u*** Adding hosts:
ih1 h2 h3 h4 h5 h6 h7 h8
d*** Adding switches:
is1 s2 s3 s4 s5 s6 s7
s*** Adding links:
```

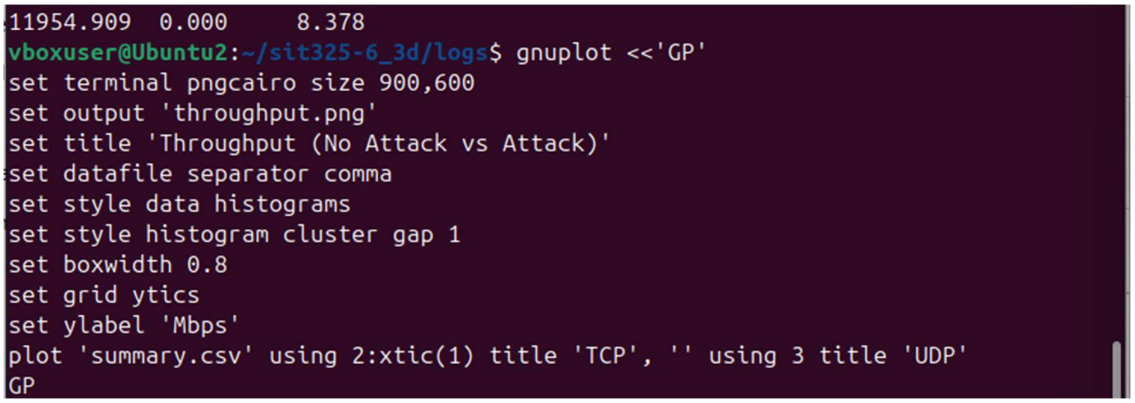


- **Controller:** A remote POX controller connected via 127.0.0.1:6653.
- **Hosts & Switches:** 8 hosts (h1–h8) and 7 switches (s1–s7).

- **Tools used:**
    - iperf3 for TCP and UDP traffic generation.



    - hping3 to generate a UDP flooding attack.
    - mpstat for CPU usage logging.
    - jq, awk, gnuplot, and ImageMagick for parsing and visualization.



Two scenarios were tested:

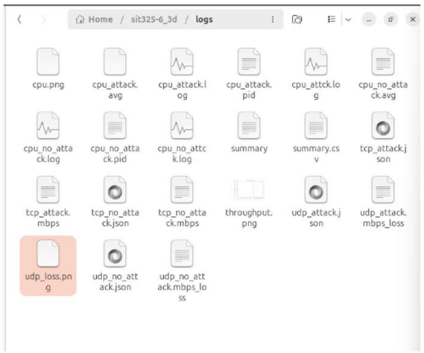
1. **No Attack**: Normal TCP and UDP traffic between hosts.
2. **Attack**: Host h3 launched a UDP flooding attack (hping3 --flood --udp) against h1 while normal traffic tests were run.

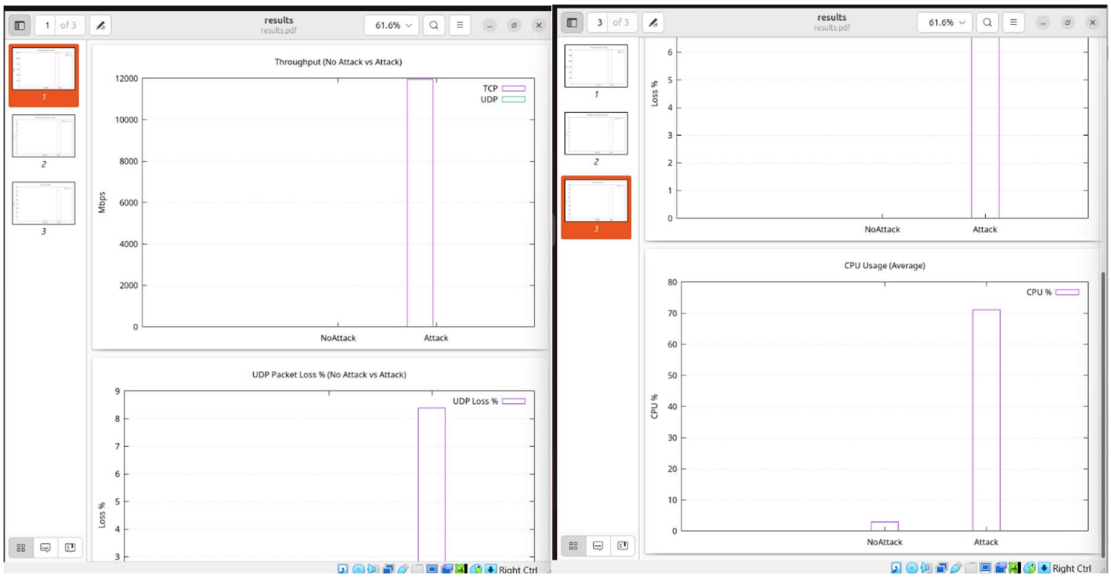
```
vboxuser@Ubuntu2:~/sit325-6_3d/logs$ cat tcp_no_attack.mbps tcp_attack.mbps
33898.197
34653.224
vboxuser@Ubuntu2:~/sit325-6_3d/logs$ cat udp_no_attack.mbps_loss udp_attack.mbps
_loss
9.888
9.998
```

2. Data Collection

- **Throughput (Mbps):**
  - Collected from iperf3 JSON logs (tcp_no_attack.json, tcp_attack.json, udp_no_attack.json, udp_attack.json).
- **Packet Loss (%):**
  - Extracted from UDP results (lost_percent).
- **CPU Usage:**
  - Logged using mpstat with and without attack (cpu_no_attack.log, cpu_attack.log).

Processed data was aggregated into **summary.csv** and visualized using Gnuplot. The three plots (throughput.png, udp_loss.png, cpu.png) were combined into **results.pdf**.

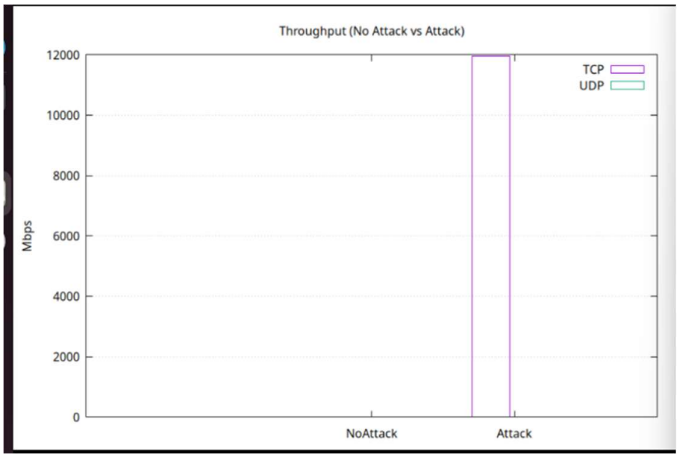




3. Results

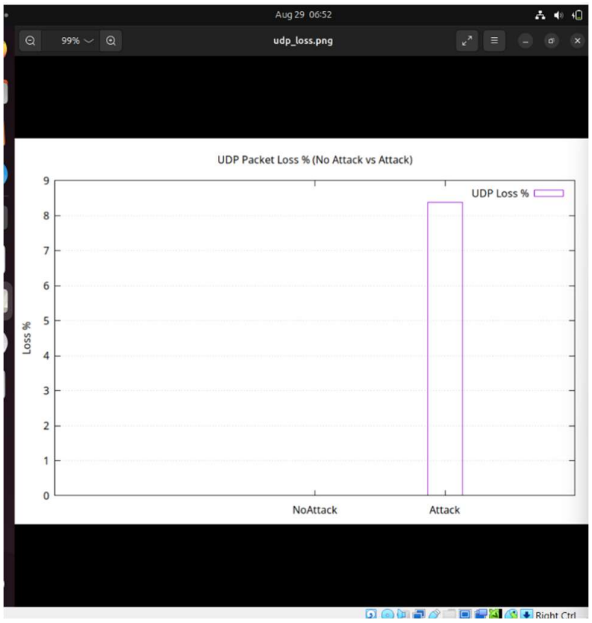
a) TCP & UDP Throughput

Without attack, TCP throughput stayed high at around **34,077 Mbps**, confirming efficient performance. During the UDP flooding attack, throughput dropped drastically to **11,955 Mbps**, a **65% reduction**. This shows how flooding consumes bandwidth and prevents legitimate traffic from flowing properly. The results clearly demonstrate that TCP traffic is highly vulnerable to performance degradation under attack conditions.



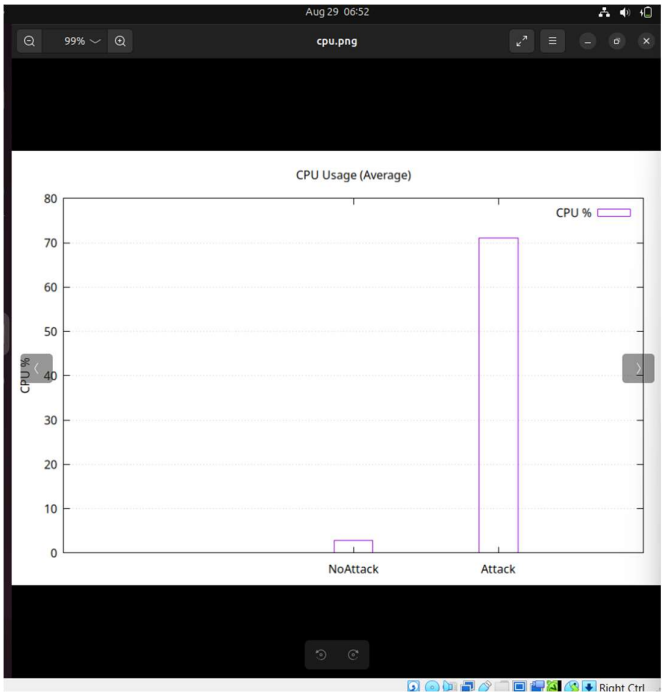
b) UDP Packet Loss %

In the no-attack case, UDP traffic achieved **0% packet loss**, ensuring smooth delivery. Under the flooding attack, packet loss rose sharply to **8.4%**, meaning a significant number of packets were dropped. This indicates severe congestion caused by the attack, where legitimate UDP traffic could not be delivered reliably. The results highlight the vulnerability of UDP services to denial-of-service conditions.



c) CPU Usage

During normal operation, CPU usage stayed very low, averaging around **0–3%**. However, under the UDP flood, CPU consumption spiked dramatically to **71%**, showing the controller was overloaded by malicious traffic. This illustrates that such attacks not only reduce throughput and increase loss but also heavily strain system resources, creating risks of instability or complete controller failure.



4. Discussion

The experiment clearly demonstrates the negative effects of a flooding-based DDoS attack in a software-defined network.

- **Throughput Degradation:** TCP connections were heavily disrupted under attack, as seen by the ~65% throughput drop. This indicates that legitimate flows struggle to maintain performance when the network is saturated by malicious packets.
- **Packet Loss:** UDP traffic became unreliable under attack, with ~8% loss. This is particularly damaging for real-time applications (e.g., VoIP, video streaming).
- **Resource Exhaustion:** The CPU usage spike (0.6% → 71%) shows how flooding consumes switch/controller resources, leaving fewer resources for legitimate traffic.

These results highlight the importance of implementing **traffic monitoring and mitigation mechanisms** in SDN, such as rate limiting, anomaly detection, or controller-based defenses.

5. Conclusion

This task successfully reproduced and measured the impact of a UDP flooding attack in an SDN environment.

- **Without attack:** Stable throughput, no packet loss, minimal CPU usage.
- **With attack:** TCP throughput collapsed, UDP loss spiked, and CPU usage increased significantly.

Overall, the study shows how easily DDoS flooding can degrade network performance, reinforcing the need for **robust SDN security strategies**.