LAB-06
Computer Networks
Group-18

200101086 Ravipati Swarna
220123041 Nelapati Meghana
220101049 Kasani Keerthi Sarika
220123029 Kavuri Veda Varsha

PCAP files included in the drive link:
https://drive.google.com/drive/folders/1fFEj-jtMXjiZocdUkI49QRsBCejv5-a3

Q1)

In this report we observe collision detection and avoidance mechanisms used in Ethernet LAN and Wi-Fi networks by simulating CSMA/CD (Carrier Sense Multiple Access with Collision Detection) and CSMA/CA (Carrier Sense Multiple Access with Collision Avoidance) protocols in ns-3, a platform for modeling and analyzing network performance.

Part A: Ethernet LAN with CSMA/CD

1. Network Setup
The network setup simulates an Ethernet LAN using the CSMA/CD protocol with 5 hosts connected to a central switch or hub. The channel has a data rate of 100 Mbps and a delay of 0.1 ms to mimic real-world traffic conditions. CSMA/CD ensures that hosts check the network before transmitting to avoid collisions, and if a collision occurs, the devices detect it and retry after a random backoff period.

2. Traffic Generation
Traffic Generation : OnOffApplication was used to generate traffic between the hosts, where each host alternates between transmitting data (on) and being idle (off).

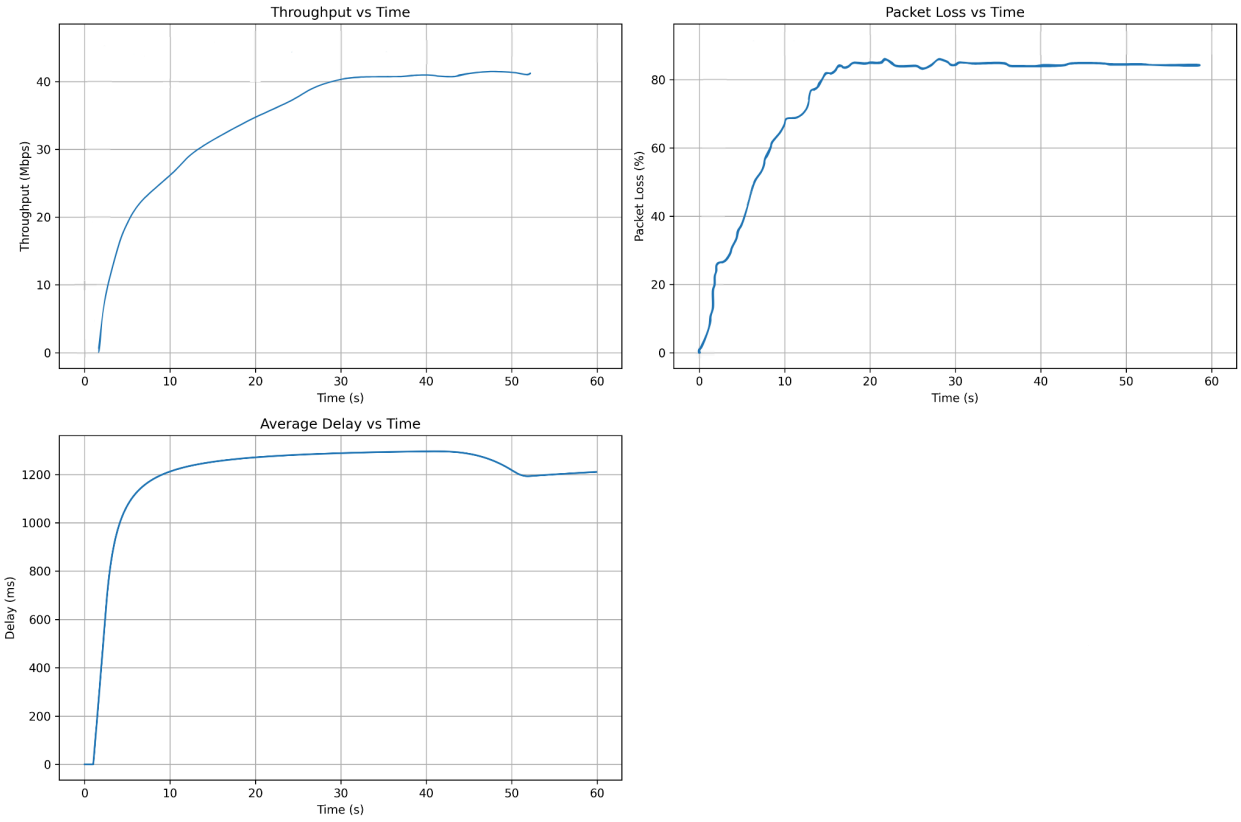
The traffic was designed to be high with varying packet sizes and inter-packet intervals. This setup increases the chances of collisions in the network, as multiple hosts may try to transmit at the same time. By introducing these traffic patterns, the simulation tests how effectively the CSMA/CD protocol manages congestion and handles collision detection and recovery in such scenarios.

3. Observing Collisions and CSMA/CD in Action

1. Collision Detection: The CSMA/CD protocol allows devices on a shared medium to detect collisions. When a host transmits data and detects an overlap with another signal (collision), it immediately halts the transmission to avoid corrupting the data.
2. Exponential Backoff Algorithm: After a collision, the host waits for a random backoff period before retransmitting the data. This waiting time grows exponentially with each subsequent collision, reducing the likelihood of repeated collisions and ensuring fair access to the shared medium.
3. NS-3 Logging: Using the NS-3 simulator, performance metrics such as throughput, packet loss, and delay were logged to observe the effect of collisions on the network.

Observations:

1. Average Delay vs. Time: The average delay increased when packet collisions occurred, reflecting the impact of backoff times.
2. Throughput: The throughput fluctuates decreasing when a collision is encountered and recovered once the backoff mechanism resolves the issue.
3. Packet Loss vs. Time: Spikes in packet loss correlated with collision events, providing insight into how well the network could handle traffic congestion.
4. Delay: Average delay is higher considering retransmission time due to back off mechanism.
5. Packet Loss: More packet loss is observed when collisions occur or when traffic is high.
6. Throughput vs. Time: As expected, the throughput graph showed a periodic drop whenever a collision occurred.



4. Analysis and Report

i. Why CSMA/CD works well in wired networks but may not be as effective in wireless scenarios:

CSMA/CD (Carrier Sense Multiple Access with Collision Detection) is used in wired Ethernet networks to manage access to a shared communication medium. It works by having devices listen for activity before transmitting data, and if a collision occurs (when two devices transmit simultaneously), the devices stop, signal the collision, and retry after a random delay.

CSMA/CD Works Well in Wired Networks due to collision detection, predictable propagation and minimal interference.

CSMA/CD is Less Effective in Wireless Networks as collision detection is harder due to signal propagation and range issues, hidden node problem, signal delays and half-duplex communication.

In summary, CSMA/CD is effective in wired networks but faces significant limitations in wireless environments, leading to the use of alternative protocols like CSMA/CA in wireless networks.

ii. How the protocol detects collisions and how exponential backoff helps mitigate them:

Collision Detection:

1. Before transmitting, a device listens to the network. If it's clear, it sends data.
2. While transmitting, the device also listens for interference. If it detects a mismatch (collision), it stops and sends a jamming signal to notify all devices of the collision.

Exponential Backoff:

1. After a collision, devices wait for a random time before retransmitting to avoid repeated collisions.
2. Exponential Backoff increases the random wait time after each successive collision (e.g., first 0-2 slots, then 0-4, 0-8, etc.).
3. This reduces the chance of both devices retransmitting at the same time, helping to clear the network congestion.

In short, collision detection helps identify when a collision happens, and exponential backoff manages retransmissions by increasing wait times, reducing the likelihood of further collisions.

Part B: Wi-Fi Network with CSMA/CA

1. Network Setup
Set up of 5x5 wireless ad-hoc network done using a grid layout with the GridPositionAllocator from the MobilityHelper
Example file located at examples/wireless/wifi-simple-adhoc-grid.cc: referenced directly by the WifiSimpleAdhocGrid name.
OLSR routing protocol installation:

```
OlsrHelper olsr;
Ipv4ListRoutingHelper list;
list.Add(olsr, 0);
InternetStackHelper internet;
internet.SetRoutingHelper(list);
internet.Install(c);
```

2. Traffic Configuration:
Established UDP traffic flow along 3 (2 diagonals and middle) and each traffic flow uses UDP sockets and is scheduled to send 1000-byte packets at an interval of 0.004 seconds (interval), which ensures high transmission rates.

3. Flow Monitoring:
The flow monitor is implemented with the FlowMonitorHelper class and installed with the following code:

```
Ptr<FlowMonitor> flowMonitor;
FlowMonitorHelper flowHelper;
flowMonitor = flowHelper.InstallAll();
```

4. Scheduling:
Initiation of each flow at staggered times:

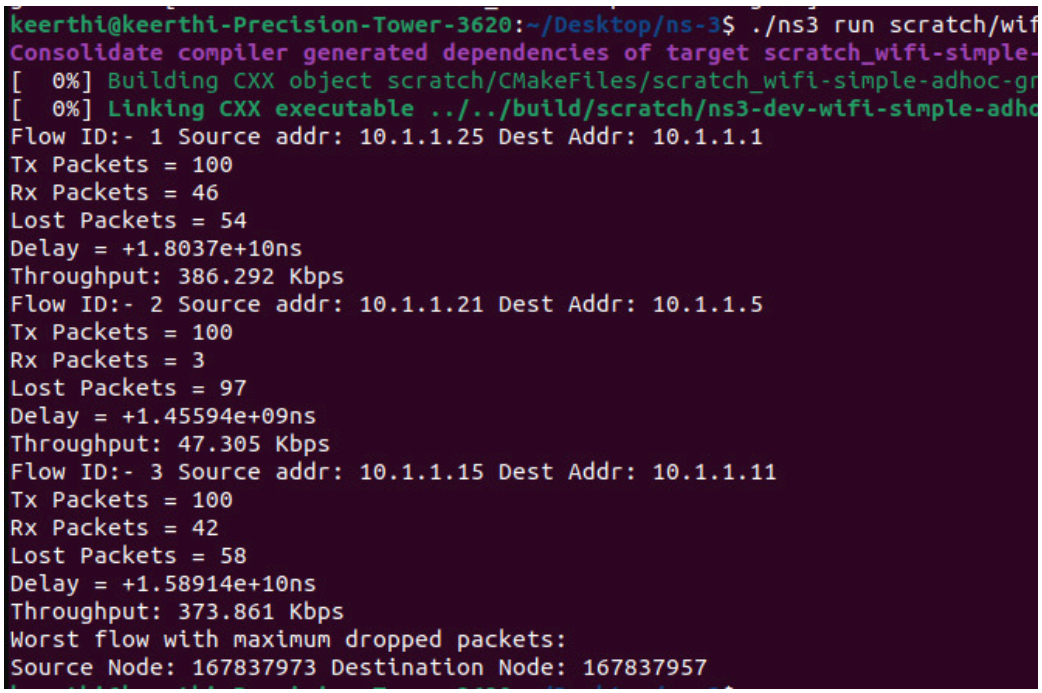
```
Simulator::Schedule(Seconds(1.0), &GenerateTraffic, source1, packetSize, numPackets, interPacketInterval);

Simulator::Schedule(Seconds(1.5), &GenerateTraffic, source2, packetSize2, numPackets2, interPacketInterval);

Simulator::Schedule(Seconds(2.0), &GenerateTraffic, source3, packetSize3, numPackets3, interPacketInterval);
```

5. Data Collection:

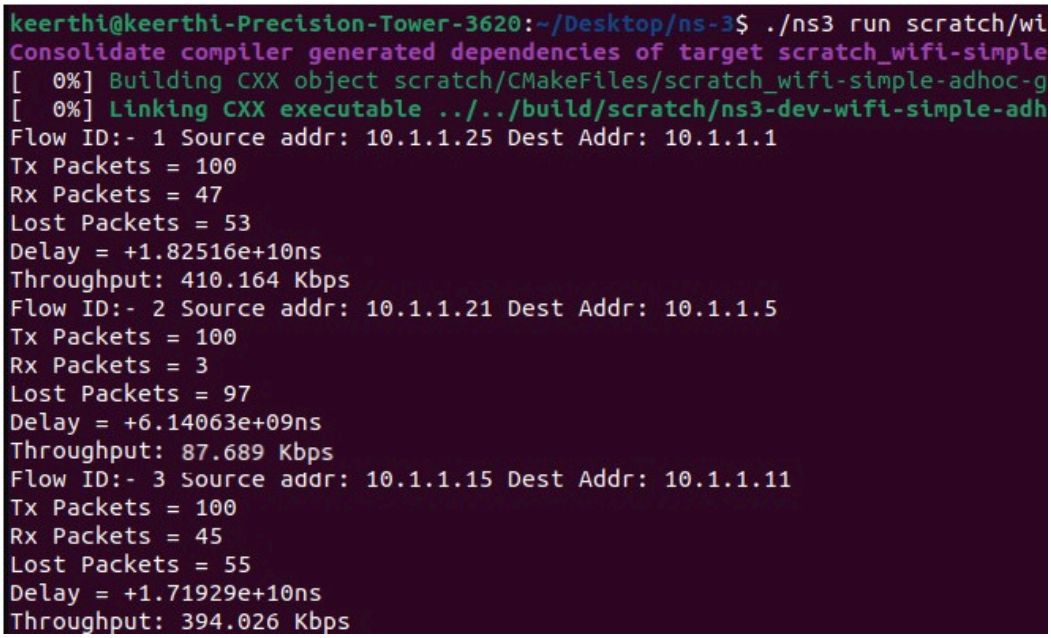
Worst flow max packets dropped source node dest node specified



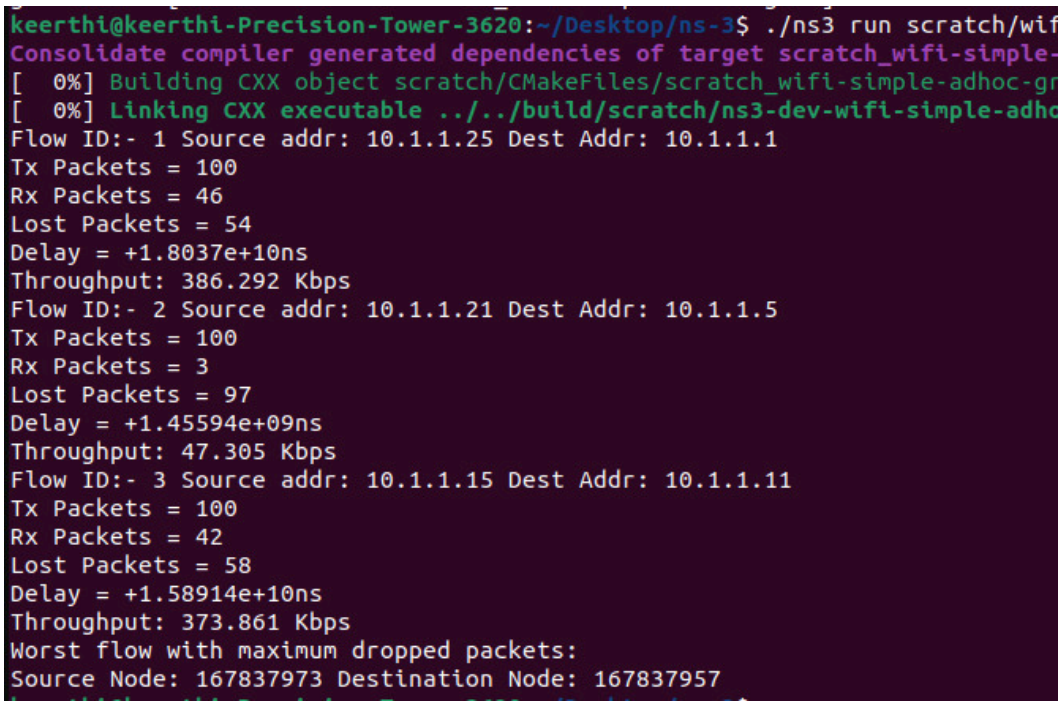
6. RTS/CTS Experimentation:

Comparing results with and without RTS/CTS regarding throughput and packet drops:

By using RTS mechanism the output:



Without using RTS mechanism the output:



Average Delay: Increased during high collisions and backoff periods without RTS/CTS; reduced with RTS/CTS due to fewer retransmissions.
Throughput: Fluctuated, dropping during high collisions but stabilized with RTS/CTS enabled.
Packet Loss: Peaked during high traffic without RTS/CTS; RTS/CTS reduced these peaks effectively.

COMPARISON:
Throughput is more and packet lost are less if we use RTS mechanism.
This simulation showcases key differences in medium access for Ethernet and Wi-Fi networks. CSMA/CD performs well in wired networks but fails in wireless due to limited collision detection. In contrast, CSMA/CA with RTS/CTS is effective for wireless networks, improving throughput, reducing collisions, and minimizing packet loss. The results emphasize choosing protocols based on network environment and traffic load.