UNIVERSITY OF SCIENCE AND TECHNOLOGY OF HANOI

NETWORK SIMULATION

Lecturer: Mrs. Nguyen Minh Huong

CSMA/CA Protocol Without RTS/CTS in Ad-hoc WLAN Network

Members:

Phạm Tuấn Hiệp	22BI13158
Đỗ Quang Anh	22BI13013
Nguyễn Viết Minh Đức	22BI13095
Nguyễn Quý Đức	22BI13094
Ngô Hải Anh	22BI13019
Phạm Quang Dương	22BI13116
Lã Duy Anh	22BI13016

March 13, 2025

Table of Content

1. Introduction

Wireless networks are essential for modern communication, but as the number of connected devices grows, challenges like transmission optimization and collision management become more difficult. One key protocol, CSMA/CA (Carrier Sense Multiple Access with Collision Avoidance), helps reduce collisions by allowing devices to sense the channel before transmitting data. When combined with RTS/CTS (Request to Send / Clear to Send), CSMA/CA addresses issues like the "hidden node" problem. This study will simulate an ad-hoc network using CSMA/CA without RTS/CTS in the NS-3.39 simulator to evaluate how RTS/CTS affects network performance and improves efficiency in wireless communication.

2. Application Building

2.1. Design Scenario

a. Ad-Hoc Topology

We are using an Ad-Hoc network where nodes communicate directly with each other without a central access point. Each node can send and receive packets from other nodes without relying on a router or access point.

```
WifiMacHelper mac;
mac.SetType("ns3::AdhocWifiMac");
```

b. Protocol

Wifi Protocol

- The **Wi-Fi** protocol is used for communication between the nodes. Wi-Fi is a widely used protocol for wireless networks, allowing nodes to communicate over wireless channels.

```
WifiHelper wifi;
devices = wifi.Install(wifiPhy, mac, nodes);
```

UDP Protocol for Application Communication

- The **UDP (User Datagram Protocol)** is used for communication between the client and the server application. We are using UdpEchoClientApplication and UdpEchoServerApplication to simulate client-server applications.

```
UdpEchoServerHelper echoServer(9);

ApplicationContainer serverApps = echoServer.Install(nodes.Get(serverNode));
serverApps.Start(Seconds(2.0));
serverApps.Stop(Seconds(15));

UdpEchoClientHelper echoClient(nodeInterfaces.GetAddress(serverNode), 9);
echoClient.SetAttribute("MaxPackets", UintegerValue(maxPackets));
echoClient.SetAttribute("Interval", TimeValue(Seconds(interval)));
echoClient.SetAttribute("PacketSize", UintegerValue(packetSize));
```

RTS/CTS Protocol

- If the packet size is smaller than or equal to 1000 bytes, RTS/CTS is not necessary, and the packets are sent directly without RTS/CTS exchange. This helps reduce collisions in an Ad-Hoc network.

```
// Disable RTS/CTS by setting the RTS/CTS threshold
UintegerValue threshold(1000);
Config::SetDefault("ns3::WifiRemoteStationManager::RtsCtsThreshold", threshold);

```

c, Mobility

The GridPositionAllocator arranges nodes in a grid with configurable spacing.

ConstantPositionMobilityModel keeps nodes static throughout the simulation.

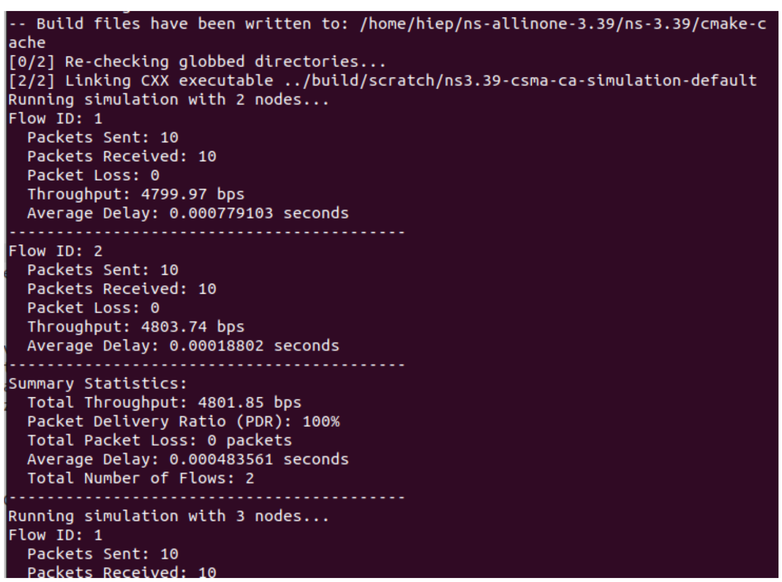
```
MobilityHelper mobility;
mobility.SetPositionAllocator("ns3::GridPositionAllocator",
                              "MinX", DoubleValue(0.0),
                              "MinY", DoubleValue(0.0),
                              "DeltaX", DoubleValue(5.0),
                              "DeltaY", DoubleValue(10.0),
                              "GridWidth", UintegerValue(3),
                              "LayoutType", StringValue("RowFirst"));

// Set the mobility model to be constant position model (fixed positions)
mobility.SetMobilityModel("ns3::ConstantPositionMobilityModel");
mobility.Install(nodes);
```

2.2. Parameters

Parameter	Value
Number of Nodes (N)	2 to 30 (loop in steps of 1)
Packet Size	512 bytes
Max Packets (each client)	10
Transmission Interval	1 second
Simulation Time	15 seconds
Mobility Model	ConstantPositionMobilityModel
Wi-Fi Standard	802.11 ad-hoc (default)
RTS/CTS Threshold	1000 (effectively off)
Flow Monitoring	Enabled

Table 1: Parameter of the application

2.3. Implement



```
-- Build files have been written to: /home/hiep/ns-allinone-3.39/ns-3.39/cmake-c
ache
[0/2] Re-checking globbed directories...
[2/2] Linking CXX executable ../build/scratch/ns3.39-csma-ca-simulation-default
Running simulation with 2 nodes...
Flow ID: 1
  Packets Sent: 10
  Packets Received: 10
  Packet Loss: 0
  Throughput: 4799.97 bps
  Average Delay: 0.000779103 seconds
-----------------------------------------
Flow ID: 2
  Packets Sent: 10
  Packets Received: 10
  Packet Loss: 0
  Throughput: 4803.74 bps
  Average Delay: 0.00018802 seconds
-----------------------------------------
Summary Statistics:
  Total Throughput: 4801.85 bps
  Packet Delivery Ratio (PDR): 100%
  Total Packet Loss: 0 packets
  Average Delay: 0.000483561 seconds
  Total Number of Flows: 2
-----------------------------------------
Running simulation with 3 nodes...
Flow ID: 1
  Packets Sent: 10
  Packets Received: 10
```

Figure 1. Running the simulation with 2-30 nodes

Figure 1 illustrates the simulation whenever it runs following each node and the flow IDs which are contained in the node.

3. Data Collection and Analysis

3.1. Collected Data

We use the flow monitor as the main method to collect data.

```
// Install flow monitor to collect data
Ptr<FlowMonitor> flowMonitor;
FlowMonitorHelper flowHelper;
flowMonitor = flowHelper.InstallAll();

Simulator::Stop(Seconds(15));
Simulator::Run();
Simulator::Destroy();

PrintFlowMonitorStats(flowMonitor);

flowMonitor->SerializeToXmlFile("DataCollectionNode_" + std::to_string(nNodes) + ".xml", true, true);
```

Figure 2: FlowMonitor to store data

3.2. Analyze Data

- Data from Flow Monitor: Data is stored in flows that contain all data about packets sent by a particular host to another.
- Data from the flows contains a lot of information, including:
  - The number of packets sent and received.
  - The average delay.
  - The throughput of each flow.
  - Number of lost packets.

```
Flow ID: 2
  Packets Sent: 10
  Packets Received: 10
  Packet Loss: 0
  Throughput: 4803.74 bps
  Average Delay: 0.00018802 seconds
------------------------------------------
Summary Statistics:
  Total Throughput: 4801.85 bps
  Packet Delivery Ratio (PDR): 100%
  Total Packet Loss: 0 packets
  Average Delay: 0.000483561 seconds
  Total Number of Flows: 2
------------------------------------------
Running simulation with 3 nodes...
Flow ID: 1
  Packets Sent: 10
  Packets Received: 10
  Packet Loss: 0
```

Figure 3: Example of data collection in each flow

- All the data of flows are updated to the summarization of the node that they belong to.

```
Summary Statistics:
  Total Throughput: 900.326 bps
  Packet Delivery Ratio (PDR): 18.75%
  Total Packet Loss: 78 packets
  Average Delay: 0.0011018 seconds
  Total Number of Flows: 32
```

Figure 4: Example of summary node 30

4. Conclusion

Explain the results: Data collision is more likely to occur in ad-hoc wireless networks if all devices are linked to one another if RTS/CTS is disabled. In light of this, as the number of nodes rises, so does the number of clients and the volume of packets be sent. More collisions occur, and more packets are lost as a result.

Future Work:

- Comparison with RTS/CTS: A follow-up experiment could be to re-enable RTS/CTS and compare the results against those presented here.
- Mobility: Introduce mobility models (RandomWaypoint, GaussMarkov, etc.) to see how movement affects ad-hoc performance.
- Different Traffic Patterns: Replace UDP Echo with other traffic types (TCP, real-time video, etc.) to see varied performance behaviors.

5. References:

- NS-3 Official Website
  - Overview & Documentation: https://www.nsnam.org/documentation/
  - Tutorials: https://www.nsnam.org/docs/tutorials/html/
  - Manual: https://www.nsnam.org/docs/manual/html/
  - API Reference (Doxygen): https://www.nsnam.org/doxygen/
- FlowMonitor API in NS-3
  - Class Reference: https://www.nsnam.org/doxygen/classns3_1_1_flow_monitor.html
- IEEE 802.11 Standard
  - Official IEEE: https://ieeexplore.ieee.org/document/7786995
- CSMA/CA Background
  - Understanding the IEEE 802.11 Distributed Coordination Function (DCF): https://www.ietf.org/proceedings/51/slides/manet-3/tsld002.htm
  - CSMA/CA Definition: https://www.techtarget.com/searchnetworking/definition/CSMA-CA#:~:text=Carrier%20sense%20multiple%20access%2Fcollision%20avoidance%20(CSMA%2FCA),over%20a%20data%20link%20layer
- Mobility Models for Wireless Networks
  - NS-3 Mobility Models Reference: https://www.nsnam.org/docs/doxygen/group__mobility.html
- Hidden Node Problem in Wi-Fi (related to RTS/CTS)
  - Sharma, M. & Sonawane, K. (2016). A Comparative Study of Hidden Node Problem in Wireless Ad hoc Networks. IEEE International Conference on Computing, Analytics and Security Trends (CAST): https://ieeexplore.ieee.org/document/7808897