ECE 6610: Wireless Networks

Programming Assignment 1
Due on: September 15, 2025, 11:59 PM ET

Upload date: September 8, 2025

About the assignment:
This programming assignment has three questions and focuses on getting familiar with the ns-3 simulator and accompanying tools. The assignment involves understanding existing tutorial files and making minor manipulations. The deliverables for each question are mentioned at the end of each question and summarized at the end.

Note that t_0 denotes the time of beginning of each simulation and time units are in seconds unless stated otherwise. When starting from an example simulation file from ns-3, copy it to scratch folder, rename it, and edit it from there. Create an MS Word document "PA1.docx" and enter answers for questions. Make sure to include the question numbers. This file ("PA1.docx") is common for all the questions in this assignment. At the top of this docx file, include your names and group name. In addition to this Word document, you will also need to upload files (summarized at the end).

1 Q1 - A wired Point-to-Point network - [30 points]

The goal of this question is to simulate a wired network with two nodes using PPP (Point-to-Point Protocol) and to inspect the transmitted packets using Wireshark. You will need to install Wireshark for this. Question 1 can be expected to be completed in 30 minutes.

The network scenario for this part is the same as the"first.cc" tutorial example (found in the folder "ns3.xx/examples/tutorial"). Copy this source file into the scratch folder with the name "q1.cc". Configure the network (in q1.cc) with the following specifications:

1. The network should consist of two nodes: a server with the IP 192.168.1.2 and a client with the IP 192.168.1.1
2. A PPP link with a data rate 1 Mbps with a channel delay of 10 ms should be established between the server and the client
3. The server node should run the "UDP echo server" application on port 6610. The server application starts and stops at times $(t_0 + 1)$ and $(t_0 + 10)$, respectively
4. The client node runs the "UDP echo client" application and sends packets to port 6610 of the server node. There is an interval of 2 seconds between each packet transmission. Each packet has a payload of 1024 bytes. The client application starts and stops at times $(t_0 + 2)$ and $(t_0 + 10)$ respectively. Ensure that the attribute *MaxPackets* is set to a large number (say 1000).

Note that the above part can be achieved by just manipulating "first.cc". Enable PCAP (Packet Capture) tracing for this simulation. Refer to this tutorial. Capture the PCAP trace for both the nodes. Save this version of the ns-3 source file as "q1.cc".

Q1-1 How many packets did the client send in total?
Q1-2 Open the PCAP files using Wireshark. From the timestamps of packets at the client, what is the round trip time (RTT) as seen by the client? What is the relationship between the RTT and the P2P link delay?
Q1-3 What is the data (content) in the payload of each packet? This can be found in the *Data* field in Wireshark for any UDP packet. How many bytes did each layer (UDP, IP, PPP) add to the payload?
Q1-4 Can you adjust the channel delay in a manner that will lead to a packet collision (between the packet sent by the client and the echo packet from server)? If yes, give an example of such a channel delay. If not, then why not?
Hint: The model documentation for PPP (found here) may help.

Deliverable: File upload: q1.cc

2 Q2 - TCP connection over WiFi - [30 points]

The goal of this question is to set-up a simple WiFi network, transmit packets via TCP, and observe the effect of distance (and hence the received signal strength - RSS). Question 2 can be expected to be completed in 1 hour.

We will start with the "wifi6610.cc" source file provided with the assignment. Configure the simulation to stop at $t_0 + 4$ seconds (variable *simulationTime*). Run this program.

The program displays the average throughput over every 100 ms and the total average throughput at the end as seen from the WiFi AP (which also functions as the TCP server).

Q2-1 What is the configured application data rate at the client? What is the final (total) average throughput (printed as Average throughput after every simulation) achieved? Is there a difference? If yes, why?
Hint: This) table may help. The network is using 802.11n over a 20 MHz bandwidth at some configured modulation and coding scheme (MCS) with a 800-nanosecond guard interval (GI).
Studying the effect of distance: Increase the distance between the client and the WiFi AP. You can do this by changing the location of the AP or the station (STA) in the code section labelled *"Mobility model"*. The coordinates are mentioned in meters. Note that the function *mobility.install()* initializes the positions in the order in which we add Vectors to *positionAlloc*. Check the throughput achieved. Increase the distance in steps of 5 metres (finer precision is not necessary)
Q2-2 When does the throughput drop to zero? We will use this distance in the next question.

Position the STA and AP to be within 5 meters of each other. Enable PCAP tracing for this simulation (variable *pcapTracing*).

Q2-3 Enable packet capture (variable *pcapTracing*). From the PCAP, observe just the first second of simulation. What is the configured beacon interval?
Q2-4 What is the first frame from the STA? What triggers this frame?

3 Q3 - Hidden terminals, MPDU aggregation, and RTS/CTS - [50 points]

This question is expected to take around 1 hour to complete. Starting from wifi6610.cc, configure this simulation to have 2 STAs. Code for the second STA can be mostly copied from the first STA. The specification of second STA is the same as the first. With the AP at [0,0,0], position the STAs at [-10,0,0] and [10,0,0]. Run this simulation and ensure there is application traffic from both STAs. By setting the variable *enableLargeAmpdu* to true or false, you can set the MPDU aggregation limit to a high or a low number of bytes respectively. Similarly, by setting the variable *enableRts* to true or false, you can enable or disable RTS/CTS.

For the questions that require simulation, run the program for 10 seconds of network activity. We will focus on how frame aggregation and RTS/CTS increase or decrease the management overhead.

Q3-1 Set *enableRts* to false. Run the simulation with and without *enableLargeAmpdu.* What is the throughput observed? Elaborate on the reason for the difference observed.
Q3-2 Based on the range that you found out in Q2-2, if the STAs are positioned at [$-x_0$,0,0] and [x_0,0,0], at what value of x_0 approximately does this scenario create hidden terminals?

Position the STAs in a manner that there are hidden nodes (STAs should always be in range of the AP). **Save this ns-3 source file as "q3.cc"**.

Q3-3 With *enableLargeAmpdu* set to false, run the simulation with and without RTS/CTS. How do the throughputs compare? Is there a difference? Give reason.
Q3-4 Briefly explain the frame exchange sequence with RTS/CTS enabled. How does it mitigate the problems caused by hidden nodes?
Q3-5 With *enableLargeAmpdu* set to true, run the simulation with and without RTS/CTS. How is the throughput affected by RTS/CTS? Compare the effect of RTS/CTS from Q3-4 and Q3-3. When is it more reasonable to use RTS/CTS?

Deliverable: File upload - q3.cc

4 Summary of deliverables

Compress all files into a single zip file and upload to the assignment on canvas. It is sufficient if one student from each group submits. Remember to change the header content in wifi6610.cc.
Deliverables:
A zip file containing PA1.docx, q1.cc, q3.cc