







CSE 4616 – Wireless Networks Lab

Assignment-1

*Impact of Traffic Load, Network Parameters on IEEE 802.11 Wireless LAN
Performance*

Table of Contents:

 Objective	3
 Simulation Setup Overview	4
 Network Topology Details	5
 Application Layer Setup	5
 Performance Metrics to Analyze	7
 Understanding Cumulative Network Load and Network Saturation	8
Task 1: Understanding Network Saturation due to Mac-Layer Retransmission	9
Task 2: Understanding the impact of Channel Interference	11
Task 3: Understanding the Performance of Wired and Wireless Communication under Network Saturation	13
 Report Submission Guidelines	15

Objective

The aim of this experiment is to evaluate the performance of wireless communication in a simulated network using **OMNeT++ and the INET Framework**.

You should analyze how **host density, traffic intensity, MAC settings, and noise** affect key performance metrics, including:

- **Packet Delivery Ratio (PDR)**
- **Throughput**
- **MAC-layer Retransmission Rate**
- **End-to-End Delay**

The ultimate goal is to **identify the network saturation point**, determine the **primary causes of performance degradation**, and understand the **limitations of wireless communication** under heavy load.



Simulation Setup Overview

- Use the provided .ini, .ned, .cpp, and .h files (available in the updated_files_8_Aug_25.zip package on Google Classroom). [Replace the corresponding default files with these updated ones.](#)
 - `inet/examples/wireless/wiredandwirelesshostswithap/omnetpp.ini` with provided `omnetpp.ini`
 - `inet/examples/wireless/wiredandwirelesshostswithap/WiredAndWirelessHostsWithAP.ned` with provided `WiredAndWirelessHostsWithAP.ned`
 - `/home/src/inet/src/inet/linklayer/ieee80211/mac/contention/contention.h` with provided `contention.h`
 - `/home/src/inet/src/inet/linklayer/ieee80211/mac/contention/contention.cc` with provided `contention.cc`
 - `/home/src/inet/src/inet/linklayer/ieee80211/mac/contention/contention.ned` with provided `contention.ned`
 - `/home/src/inet/src/inet/applications/udpapp/UdpSink.h` with provided `UdpSink.h`
 - `/home/src/inet/src/inet/applications/udpapp/UdpSink.cc` with provided `UdpSink.cc`
 - `/home/src/inet/src/inet/physicallayer/common/packetlevel/Radio.h` with provided `Radio.h`
 - `/home/src/inet/src/inet/physicallayer/common/packetlevel/Radio.cc` with provided `Radio.cc`
 - `/home/src/inet/src/inet/physicallayer/common/packetlevel/RadioMedium.h` with provided `RadioMedium.h`
 - `/home/src/inet/src/inet/physicallayer/common/packetlevel/RadioMedium.cc` with provided `RadioMedium.cc`
 - `/home/src/inet/src/inet/physicallayer/ieee80211/packetlevel/errormodel/Iee80211YansErrorModel.h` with provided `Iee80211YansErrorModel.h`
 - `/home/src/inet/src/inet/physicallayer/ieee80211/packetlevel/errormodel/Iee80211YansErrorModel.cc` with provided `Iee80211YansErrorModel.c`
- Use the **lowest student ID** from your group members as the seed value in the provided `omnetpp.ini` file.
(e.g., `seed-set = 210041000`)
- Set the simulation duration to **40 seconds** by configuring the `sim-time-limit` parameter in the provided `omnetpp.ini` file.

Network Topology Details

Simulate a hybrid network of:

4. Wireless nodes connected via IEEE 802.11 (Wi-Fi)
5. Wired nodes connected via Ethernet
6. A router for wired communication
7. An AP bridging wireless and wired segments

Wireless Segment

- Multiple wireless hosts, placed in a 2D area (0–400 m in X and Y)
- Communicate with an **Access Point** using **IEEE 802.11**
- Radio propagation error modeled using `Ieee80211YansErrorModel`

Wired Segment

- Wired hosts, connected via Ethernet through a central **router**
- One wired host, directly connected to the **Access Point**

Application Layer Setup

- **Wireless hosts** use `UdpBasicApp` to send traffic
- **Wired hosts** use `UdpSink` to respond
- UDP parameters like packet size (e.g., 1000 B), send intervals (e.g., 0.1 s), and stop time are configured in the `.ini` file

Run the simulation from the Terminal

Initial GUI Run (*required only once for any INET project simulation*):

- Before using the terminal, ensure that **at least one simulation run** (for any configuration) is executed in **GUI mode** using the OMNeT++ IDE.
- This step ensures that the **entire INET project is built** successfully.

Subsequent Runs in Terminal (Faster Execution):

- Once the INET project has been built in the IDE, you may run **any configuration in terminal mode** for the rest of your experiments to achieve faster execution.

Running from the Terminal:

- Open a terminal from the project location (use Linux File Explorer, not the Omnet IDE File Explorer !).
- Run the following command
 - `opp_run -u Cmdenv -n ../../../../examples:../../../../../src -l ../../../../src/INET omnetpp.ini -c High_Retransmission`

Here,

- **opp_run**: Launches the OMNeT++ simulation executable.
- **-u Cmdenv**: Runs the simulation using the command-line user interface (Cmdenv), instead of the graphical one (like Tkenv or Qtenv).
- **-n ../../../../examples:../../../../../src**: Sets the **NED (Network Description) folders** — directories where OMNeT++ looks for **.ned** files. These define the simulation network.
- **-l ../../../../src/INET**: Loads a **shared library** (INET in this case), which contains compiled simulation models. The INET framework provides models for internet protocols, wireless, etc.
- **omnetpp.ini**: Specifies the **configuration file** for the simulation — it defines parameters, modules, simulation time, etc.
- **-c High_Retransmission**: Choose the **specific configuration** (from the **omnetpp.ini** file) to run. In this case, it's **High_Congestion**.



Performance Metrics to Analyze

Metric	Indicates	Affected by
PDR	Reliability (success rate)	<ol style="list-style-type: none"> 1. Packet Loss <ol style="list-style-type: none"> a. Packet Corruption due to Collision or Interference b. Queue Overflow
Throughput	Network capacity (data transfer)	<ol style="list-style-type: none"> 1. Maximum Bitrate (Channel Bandwidth) 2. Packet Generation Rate 3. Packet Loss
MAC-layer Retransmissions Rate	Channel contention	<ol style="list-style-type: none"> 1. Packet Loss <ol style="list-style-type: none"> a. Packet Corruption due <ol style="list-style-type: none"> i. Collision ii. Interference b. Queue Overflow
Bit Error Rate (BER) (Errors in received bits due to noise/interference)	Link reliability indicator	<ol style="list-style-type: none"> 1. Channel noise 2. Interference, 3. Fading 4. Modulation and coding scheme
End-to-End Delay (caused by Queueing Delay, Propagation Delay, Retransmissions)	Timeliness of delivery	<ol style="list-style-type: none"> 1. Contention 2. Retransmissions

1. Packet Delivery Ratio (PDR)

$$PDR(\%) = (Number\ of\ Packets\ Successfully\ Received \div Number\ of\ Packets\ Sent) \times 100$$

2. Throughput

$$Throughput = (Total\ Bytes\ Successfully\ Received \div Total\ Simulation\ Time)$$

4. MAC-layer Retransmissions Rate

$$Retransmission\ Rate = Number\ of\ Retransmitted\ MAC\ Frames \div Total\ MAC\ Frames\ Sent$$

Understanding Cumulative Network Load and Network Saturation

$$\text{Cumulative Network Load (bps)} = N \times (\text{Packet Size (bits)} \div \text{Send Interval (s)})$$

Where,

N = Number of wireless hosts generating traffic

Packet Size (bits) = Size of each packet (e. g., $1000\text{ B} = 8000\text{ bits}$)

Send Interval (s) = Time between consecutive packet transmissions from each host

Description:

- **Cumulative network load** represents the **total offered traffic rate (in bits per second)** generated by all wireless hosts combined.
- By **increasing the number of hosts** or **reducing the send interval**, you **increase the cumulative load** on the network.
- When the cumulative load **approaches or exceeds the network capacity**, performance metrics (like PDR and throughput) degrade, indicating **saturation**.

Lab Tasks

Task 1: Understanding Network Saturation due to Mac-Layer Retransmission

Goal:

Conduct a series of network simulations that **gradually increase the cumulative network load** (traffic generated) by **wireless hosts**.

The objective is to analyze how rising traffic intensity at wireless networks under a certain network capacity leads to an **increase in MAC-layer retransmission** and, ultimately, a **degradation in overall network performance**, thereby identifying the **saturation point**.

- Identify the **saturation point**, characterized by:
 - Sharp PDR drop,
 - Throughput flattening,
 - Rapid increase in retransmissions,
 - Spikes in delay

To simulate **increasing traffic load** and **MAC-layer retransmission** in the **wireless segment of the network**, define separate configurations in the `omnetpp.ini` file for triggering the influencing factors resulting in higher chances of MAC-layer **retransmission**. You may follow the steps listed below:

Steps:

1. Adjust Wireless Host Parameters

- Define the number of wireless hosts and vary this across multiple simulation runs.
- Ensure that changes in the number of hosts contribute to a gradual increase in offered network load.

2. Configure Traffic Generation

- Use **UdpBasicApp** for wireless hosts to generate high-intensity traffic.
- Adjust **packet size** and **send intervals** to control the offered traffic load.
- Study the relationship between offered load and network performance metrics (PDR, throughput, etc.).

4. Configure Distributed Coordination Function (DCF) Parameters

- Apply theoretical knowledge from **Lecture 6.1** and **Lecture 6.2** to modify MAC-layer DCF parameters (e.g., contention window, backoff) to **increase collision probability** and observe the resulting effects.

5. Run Multiple Configurations

- A. Incrementally vary both **the number of wireless hosts** and/or **traffic generation parameters** across simulation runs.
- B. Aim to progressively increase cumulative load until **saturation symptoms** appear.
- C. Repeat A and B for varying **Distributed Coordination Function (DCF) Parameters**.

6. Observe Key Performance Metrics

- **Extract and analyze metrics** from `.vec` and `.sca` files using OMNeT++ IDE and/or Python.
- Monitor the following metrics:
 - **PDR (Packet Delivery Ratio)** – Reduced due to packet corruption from interference. (Extract required data from scalar statistics to calculate PDR for both UDP packets and MAC frames).
 - **Throughput** – Drops as more packets fail to be successfully delivered. (Extract required data from scalar statistics to calculate throughput for both UDP packets and MAC frames).
 - **MAC-layer Retransmission Rate**
 - Hint: You do not need to extract this information from scalar or vector records. Each simulation run will automatically generate a `cwUsed.csv` file, which contains the **minimum contention window values** used by **each transmitting node**. **Please be remembered that each run will overwrite the content of the file.**
 - Apply theoretical knowledge from **Lecture 6.1** to understand the meaning of **minimum contention window value at MAC-layer**.
 - **End-to-End Delay** – May increase due to retransmissions caused by corrupted packets.
 - Hint: You don't need to extract this from scalar or vector records. Each simulation run automatically generates a `.csv` file named `udpPacketTransmissionInfo.csv` that logs the per-packet end-to-end delay. **Please be remembered that each run will overwrite the content of the file.**
- For each performance metric (PDR, Throughput, Retransmission Rate, and End-to-End Delay):
 - Compute and report the **minimum, maximum, and average values across all simulation runs**.

7. Final Observations

- Summarize the findings with clear **interpretation of observed performance trends**.
- Justify the identified **saturation point** in terms of **network capacity and contention limits**.
- Present **key insights** gained from these experiments regarding wireless network behavior under high traffic load.

Task 2: Understanding the impact of Channel Interference

Goal:

Conduct network simulations to evaluate how **channel interference** affects wireless communication performance.

The objective is to **study the impact of noise** on key performance metrics such as **Packet Delivery Ratio (PDR)**, **Throughput**, and **End-to-End Delay**, **Bit Error Rate (BER)**, and **SNIR**. This task focuses on understanding how **unfavorable channel conditions** (e.g., background noise or weak signal strength) lead to **packet corruption** and performance degradation.

Steps:

1. Configure Channel Interference Parameters

- Introduce **background noise** in the simulation environment.
- Experiment with varying levels of noise power and sensitivity thresholds to simulate different interference levels.

2. Adjust Signal Quality and Range

- Modify **transmission power** or receiver **sensitivity** to observe how weak signals increase packet errors:

3. Configure Traffic Generation

- Use **UdpBasicApp** to send continuous traffic to ensure sufficient packet transmission for analyzing packet loss due to interference.
- Keep traffic generation parameters (packet size, send interval) **constant across runs**, isolating interference effects.

4. Run Multiple Configurations

- Vary **noise levels**, or **transmission power** across different simulations.
- Run scenarios with **gradually worsening channel conditions** to observe how interference impacts performance metrics.

5. Observe Key Performance Metrics

Extract and analyze metrics from `.vec` and `.sca` files using OMNeT++ IDE or Python. Focus on:

- **PDR (Packet Delivery Ratio)** – Reduced due to packet corruption from interference. (Extract required data from scalar statistics to calculate PDR for both UDP packets and MAC frames).
- **Throughput** – Drops as more packets fail to be successfully delivered. (Extract required data from scalar statistics to calculate throughput for both UDP packets and MAC frames).
- **MAC-layer Retransmission Rate**
 - Hint: You do not need to extract this information from scalar or vector records. Each simulation run will automatically generate a `cwUsed.csv` file, which contains the **minimum contention window values** used by **each transmitting node**. **Please be remembered that each run will overwrite the content of the file.**
 - Apply theoretical knowledge from **Lecture 6.1** to understand the meaning of **minimum contention window value at MAC-layer**.
- **End-to-End Delay** – May increase due to retransmissions caused by corrupted packets.
 - Hint: You don't need to extract this from scalar or vector records. Each simulation run automatically generates a `.csv` file named `udpPacketTransmissionInfo.csv` that logs the per-packet end-to-end delay. **Please be remembered that each run will overwrite the content of the file.**
- **Bit Error Rate (BER) and Signal-Noise-Interference Ratio (SNIR)** –
 - Hint: You don't need to extract this from scalar or vector records. Each simulation run automatically generates two `.csv` files named `DataErrorRate.csv` and `HeaderErrorRate` that logs the following items per-packet.
 - **Header length**, and **Bit Error Rate (BER) only for Header** (in `HeaderErrorRate.csv`)
 - **Data length**, and **Bit Error Rate (BER) only for Data** (in `DataErrorRate.csv`)
 - To get the **Bit Error Rate (BER)** for the complete packet you have to calculate it by considering the following above two.
 - The same **SNIR** value appears in both `.csv` files because the calculated **SNIR** is based on both the header and the data combined.
 - **Please be remembered that each run will overwrite the content of the file.**

For each performance metric (PDR, Throughput, and End-to-End Delay, Bit Error Rate (BER), Signal-Noise-Interference Ratio (SNIR) etc.):

- Compute and report the **minimum, maximum, and average values across all simulation runs**.

6. Identify Interference Impact Points

- Determine **critical interference levels** (e.g., noise power threshold) where packet loss or delay sharply increases.
- Analyze **signal quality metrics (SNIR, BER)** to correlate interference with network performance degradation.

7. Final Observations

- Summarize findings, explaining how interference affects packet reliability and throughput.
- Discuss the relationship between **noise, and signal strength**.
- Suggest strategies to mitigate interference.

Task 3: Understanding the Performance of Wired and Wireless Communication under Network Saturation

Goal:

The objective of this task is to **compare the performance of wired and wireless communication segments when the network operates under saturation conditions.**

By running simulations for both segments under **identical network loads** and **channel capacity**, we will analyze how saturation affects the performance of each medium and identify which segment is more resilient to high traffic loads.

Steps:

1. Wireless Segment Saturation (Run-1)

- Activate only the **wireless segment**, where wireless hosts communicate with the **Sink Node** through the Access Point (AP).
- Configure the wireless traffic load (e.g., send interval, packet size, and number of hosts) to a level where the **wireless network is saturated** (**identified from Task-1 results**).
- Ensure that all other segments (wired hosts) are disabled.

2. Wired Segment Saturation (Run-2)

- Activate only the **wired segment**, where wired hosts communicate with the **Sink Node** through AP using Ethernet.
- Configure the **same offered traffic load and channel capacity** as used in **Run-1** for the wireless segment, ensuring an equivalent comparison.
- Disable wireless hosts during this run.

3. Configure Traffic Generation

- Use **UdpBasicApp** for wireless hosts and **UdpSink** for wired hosts.
- Set consistent UDP parameters (packet size, send interval, simulation time) to ensure a fair evaluation.
- Keep MAC and PHY layer parameters for both runs fixed to isolate the effects of medium differences.

4. Run Multiple Configurations

- Perform simulations by **varying network loads** slightly around the saturation point (e.g., $\pm 10\%$ of the identified saturation load).
- Conduct multiple runs to gather sufficient statistical data.

5. Observe Key Performance Metrics

- Extract and analyze metrics from `.vec` and `.sca` files using OMNeT++ IDE or Python scripts.
- Focus on the following performance metrics:
 - **PDR (Packet Delivery Ratio)** – Reduced due to packet corruption from interference. (Extract required data from scalar statistics to calculate PDR for both UDP packets and MAC frames).
 - **Throughput** – Drops as more packets fail to be successfully delivered. (Extract required data from scalar statistics to calculate throughput for both UDP packets and MAC frames).
 - **MAC-layer Retransmission Rate**
 - Hint: You do not need to extract this information from scalar or vector records. Each simulation run will automatically generate a `cwUsed.csv` file, which contains the **minimum contention window values** used by **each transmitting node**. **Please be remembered that each run will overwrite the content of the file.**
 - Apply theoretical knowledge from **Lecture 6.1** to understand the meaning of **minimum contention window value** at MAC-layer.
 - **End-to-End Delay** – May increase due to retransmissions caused by corrupted packets.
 - Hint: You don't need to extract this from scalar or vector records. Each simulation run automatically generates a `.csv` file named `udpPacketTransmissionInfo.csv` that logs the per-packet end-to-end delay. **Please be remembered that each run will overwrite the content of the file.**

6. Statistical Analysis

- For each metric (PDR, Throughput, End-to-End Delay, etc.):
 - Compute and report **minimum, maximum, and average values** across all runs.

7. Comparison and Bottleneck Analysis

- Compare wired vs. wireless performance using the collected metrics.
- Determine which segment hits its performance limits first under heavy load.
- Analyze **why wireless performance is more affected** compared to wired communication.

8. Final Observations (Expected Findings):

- Summarize the differences between wired and wireless segments under identical loads.
- Justify the observed **performance gap**.
- Identify the **network bottleneck** and explain the role of medium characteristics.
- Provide **recommendations** for improving wireless performance under saturation.

Report Submission Guidelines

1. Submission Requirement:

Your submission must include the following components:

Simulation Files:

- Include all configuration and network design files, such as **.ini**, **.ned**, and **.anf** files.

Statistical Data Files:

- Provide the source files used for generating graphs (e.g., **Excel spreadsheets or Python scripts**).

Report(.docx):

- Present a well-structured document (preferred in .docx) that offers a **logical and step-by-step analysis** of observed trends for all tasks.
 - Include properly labeled plots with clear titles, axes labels, and legends for **Visualizations**.
 - Provide a detailed explanation of the **root causes of performance degradations** and other contributing factors.

2. Group Formation:

- Each group may have up to 3 students, but **smaller groups are encouraged and will be given additional credit**.
- Only one member of the group should submit the final Assignment Content through Google Classroom.

3. Submission Deadline:

- The assignment must be submitted on Google Classroom by **[26 August 2025 2.30 pm]**.