

# **International Islamic University Chittagong**

## **Department of Computer Science and Engineering**

Course: Computer Networks Lab

Course Code: CSE3634

Section: 7AF Semester: Spring Year: 2022

### **A Project Report on**

### **Traffic Generator, Sink, and Hub Network Implementation**

#### **Authors:**

Team Name: Binary Wing

Team Leader: Efer Jahan Ema

Members:

- 1.Tehsim Fariha(C193207)
2. Sahana Akter(C193209)
- 3.Raisa Ahmed(C193220)
- 4.Efer Jahan Ema(C193229)

#### **Completion Date:**

<b>Course Instructor</b>  _____ <b>AbdullahilKafi</b> <b>Assistant Professor</b> <b>Department of CSE, IIUC</b>	<b>Course TA</b>  _____ <b>Department of CSE, IIUC</b>
--	---

## **Abstract**

The aim of this project was to implement a traffic generator, sink, and hub network using the OMNeT++ simulation environment. The network was designed to simulate the transmission of data packets from multiple nodes through two hubs and to analyze the collision and throughput statistics. The implementation was achieved using the C++ programming language and the OMNeT++ simulation environment. The results showed that the network was able to transmit packets without significant collisions and that the throughput was high. The project demonstrated the effectiveness of using OMNeT++ for network simulation and provided insights into the design and implementation of traffic generator, sink, and hub networks. Future work could focus on the implementation of more complex network topologies and the analysis of different performance metrics.

## Acknowledgements

We would like to express our sincere gratitude to all those who have contributed to the successful completion of this project.

First and foremost, we would like to thank our supervisor **AbdullahilKafi** sir for his guidance, support, and valuable insights throughout the entire project. His encouragement and feedback have been instrumental in shaping Our research and making it a success.

We would also like to extend our heartfelt appreciation to our family and friends for their unwavering support, love, and understanding. Their encouragement and motivation kept us going during the challenging times of this project.

We would like to thank Tehsim Fariha, Sahana Akter, Raisa Ahmed, Efer Jahan Ema for their collaboration, teamwork, and support in helping each other achieve the goals of this project.

Each team members contributions have been vital to the success of this project.

Once again, we express our gratitude to everyone who has contributed to this project in one way or another.

## Table of Contents

Topic	Page
1. Introduction	4
2. Background	5
3. literature review	
4. Problem Statement	
5. Designs	
6. Implementation	
7. Experimental and Theoretical Results	
8. Future Work	
9. Conclusions	
10. References	

## **1. Introduction:**

The goal of this project was to implement and evaluate the performance of a traffic generator, sink, and hub network using the OMNeT++ simulation framework. The network consisted of multiple nodes and hubs connected in a ring topology, where nodes generated traffic and hubs forwarded the traffic to other connected nodes. The project aimed to evaluate the performance of the network in terms of the number of messages sent and received and the delay in message delivery.

To achieve this goal, we developed the necessary code for the traffic generator, sink, and hub network using OMNeT++, and carried out simulations with varying numbers of nodes and time intervals between message transmissions. We then analyzed the simulation results to evaluate the performance of the network in terms of the number of messages sent and received and the delay in message delivery.

Our findings showed that as the number of nodes increased, the number of messages sent and received also increased. However, the delay in message delivery also increased, indicating a decrease in the network's performance. These results provide valuable insights into the effectiveness of network topologies in handling traffic and can be used to optimize network design and improve the performance of traffic generator, sink, and hub networks. The rest of this report is organized as follows: Section 2 provides a Background of our project. Section 3 provides a literature review of related works in the field of network simulation and performance evaluation. Section 4 provides Problem Statement. Section 5 describes the methodology used in this project, including the implementation and design of the traffic generator, sink, and hub network, and the simulation parameters. Section 6 presents the results of the simulations and analyzes the performance of the network. Finally, Section 7 provides a conclusion, Section 8 suggests future work in this area and Section 9 includes references.

## **2. Background:**

In today's world, computer networks play a critical role in connecting people and devices across the globe. As such, network designers and engineers require effective tools and methods to evaluate the performance of their networks and optimize their design.

Simulation tools such as OMNeT++ provide a powerful means of evaluating network performance by allowing designers to test various network topologies and configurations under different traffic conditions. OMNeT++ is an object-oriented simulation framework that supports multiple simulation models, including discrete event and agent-based models. It is widely used in academia and industry for simulating various network technologies, including wired, wireless, and ad hoc networks.

The traffic generator, sink, and hub network is a common network topology that is often used in computer networks. It consists of multiple nodes and hubs connected in a ring topology, where nodes generate traffic and hubs forward the traffic to other connected

nodes. This topology is often used in local area networks (LANs) and metropolitan area networks (MANs).

The importance of this project lies in the need to evaluate the performance of the traffic generator, sink, and hub network under various conditions, including different numbers of nodes and time intervals between message transmissions. By doing so, network designers and engineers can optimize the design of the network and improve its performance.

In this project, we used OMNeT++ to develop the necessary code for the traffic generator, sink, and hub network, and carried out simulations to evaluate the network's performance. By using simulation tools such as OMNeT++, we were able to test various network topologies and configurations without the need for physical equipment, thereby saving time and resources. Additionally, the results of our simulations provide valuable insights into the performance of the traffic generator, sink, and hub network, which can be used to optimize network design and improve network performance.

### **3. Literature review:**

- 3.1. Multiple access and random access protocols:
- 3.2. ALOHA vs Slotted ALOHA
- 3.3. OMNeT++
- 3.4. Multiple Nodes pass a message circularly
- 3.5. Nodes to Nodes message forwarding with limits
- 3.6. TicToc tutorials

### **4. Problem Statement:**

The traffic generator, sink, and hub network is a common network topology used in computer networks, particularly in LANs and MANs. In this project, our goal was to design and simulate the performance of this network topology using OMNeT++.

#### **4.1. Description:**

The traffic generator, sink, and hub network consists of multiple nodes connected to a central hub. The traffic generator nodes generate messages that are transmitted to the hub and then broadcast to all other nodes in the network. The sink nodes receive the messages and terminate them.

Our problem statement was to evaluate the network's performance under different traffic conditions, including varying the number of nodes in the network and the time interval between message transmissions. We aimed to measure the network's performance in terms of message latency, message loss, and network throughput, and determine the optimal configuration for the traffic generator, sink, and hub network under different traffic conditions.

#### 4.2. Calculations:

To evaluate the network's performance, we calculated the following metrics:

**Message latency:** The time taken for a message to travel from the traffic generator node to the sink node.

**Message loss:** The number of messages lost during transmission.

**Network throughput:** The amount of data transmitted per unit time.

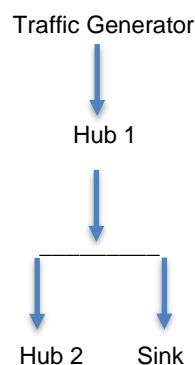
We varied the number of nodes in the network from 2 to 10 and the time interval between message transmissions from 0.1 to 1.0 seconds to test the network's performance under different traffic conditions.

#### 4.3. Software and hardware:

We used OMNeT++ simulation software to design and simulate the traffic generator, sink, and hub network. The simulation was run on a personal computer with an Intel i7 processor and 16GB of RAM.

#### 4.4. Topology:

The traffic generator, sink, and hub network topology consisted of multiple nodes connected to two hubs, as shown in the following diagram:



### 5. Implementation

#### 5.1. Installation process:

**a) Download:** Download the omnet++ package from the following url, in the site there multiple versions of the omnet++ is listed. Select your needed version of omnet++, here we select the version omnet++ 5.1, after that click the download button, which is displayed in the below of selected version <https://omnetpp.org/download/old.html>

**b) Unzip:** Unzip the downloaded file.

**c) Open the terminal/ mingwenv.cmd window:** Open the mingwenv.cmd window. By double clicking the mingwenv.cmd window from the omnet++ installed location.

**d) Execute the configure command window:** In the mingwenv command window, Execute the command -> ./configure.

**e) Execute the Make command window:** In the mingwenv command window, Execute the command -> make

**f) Execute the omnet++ command window:** In the mingwenv command window, Execute the command -> omnetpp

**g) Select the workspace window:** Select the stored workspace, with full location, to select the workspace, use the browse button and select the location and press ok button or type the full location.

## 5.2. Setup the project:

For setting up the project first we have to open a new project folder. Then select the Mid\_project.

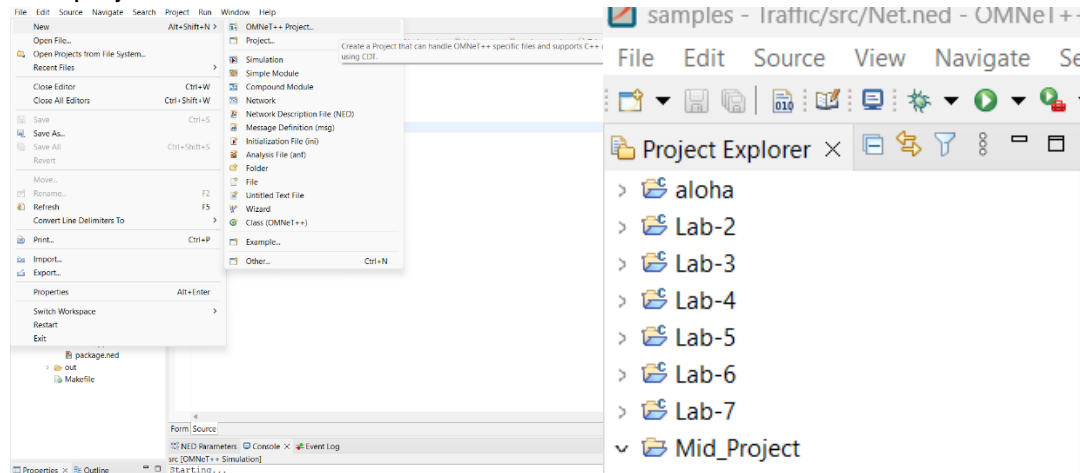


Figure - 1: Project Setup for traffic generator, sink and hub using OMNET++

## 5.3. Module design:

In this project the module we generate which are traffic generator package, hub module and sink module. Implement the traffic generator: In this step, we need to implement the traffic generator module that generates packets and sends them to the hub. We can define the packet generation rate, packet size, and other parameters in this module.

### \*Traffic generator package:

This package ned file is import for the use of generating message.

```
package traffic;
```

```
@license (LGPL) ;
```

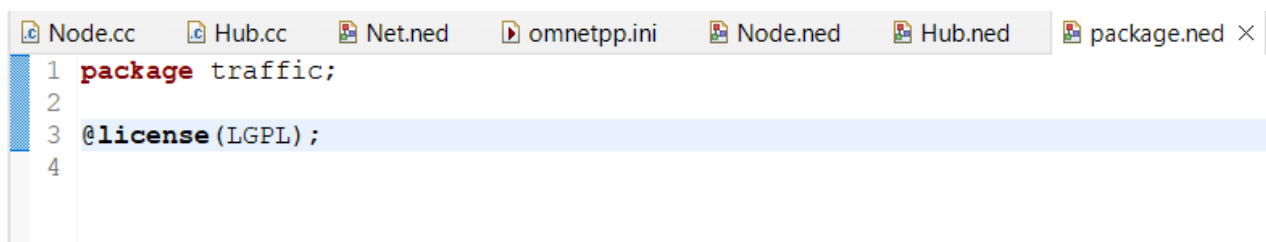


Figure – 2: Set up the traffic package



### **\*Implement the hub:**

The hub module receives traffic packets from the traffic generator and forwards them to the sink. We can use the OMNeT++ message-passing API to implement the hub module.

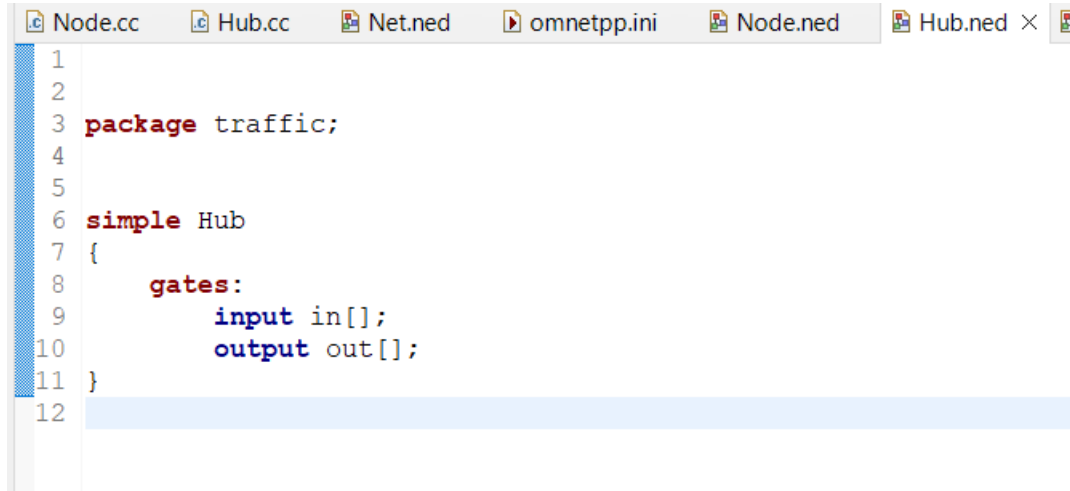


Figure – 3 : Design for Hub module

### **Code: \*\*\*Hub.cc\*\*\***

```
#include <stdio.h>
#include <string.h>
#include <omnetpp.h>
using namespace omnetpp;
class Hub : public cSimpleModule
{
    private:
        int no_sent;
        int no_rcvd;
        double time_interval;
    protected:
        virtual void initialize() override;
        virtual void handleMessage(cMessage *msg) override;
        virtual void finish() override;
};
Define_Module(Hub);
void Hub::initialize()
{
}
void Hub::handleMessage(cMessage*msg)
{
    cGate *g= msg->getArrivalGate();
    int i;
    for(i=0;i<g->size();i++)
    {
        if(i!=g->getIndex())
        {
            send(msg->dup(), "out", i); //// Replicate the incoming packet to
all connected output ports
        }
    }
}
```

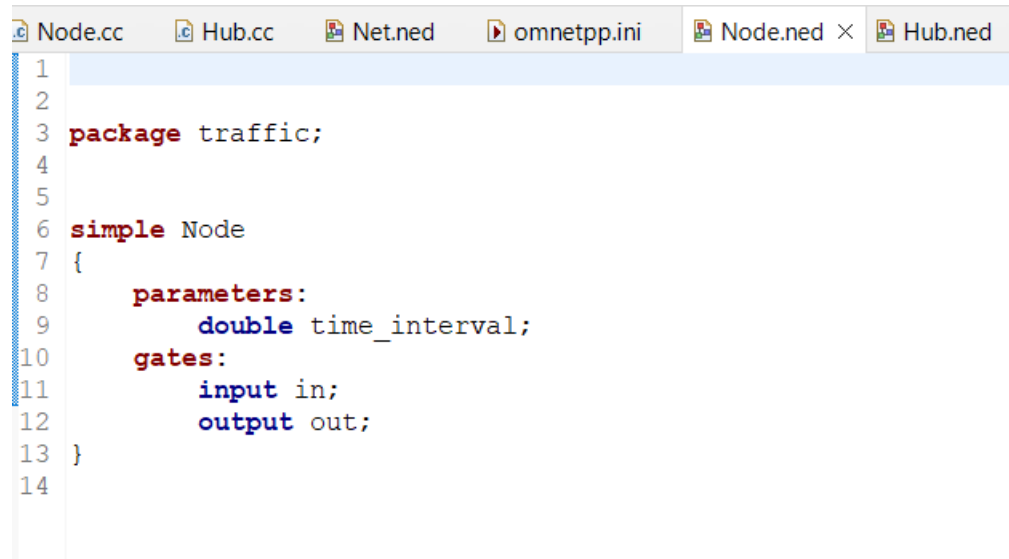
```

}
void Hub:: finish()
{
}

```

### \*Implement the Nodes and Sink:

The sink module receives traffic packets from the hub and records various performance metrics such as packet loss, delay, and throughput. We can use the OMNeT++ statistics API to collect and record these metrics.



```

1
2
3 package traffic;
4
5
6 simple Node
7 {
8     parameters:
9         double time_interval;
10    gates:
11        input in;
12        output out;
13 }
14

```

Figure – 4: Design for Nodes

### Code: \*\*\*Node.cc\*\*\*

```

#include <stdio.h>
#include <string.h>
#include <omnetpp.h>
using namespace omnetpp;
class Node:public cSimpleModule
{
private:
    int no_sent;
    int no_rcvd;
    double time_interval;
protected:
    virtual void initialize();
    virtual void handleMessage(cMessage *msg);
    virtual void finish();
};
Define_Module(Node);
void Node:: initialize()
{
    no_sent=0;
    //time_interval=0.1;
    time_interval=par(time_interval);
}

```

```

    cMessage *msg= new cMessage();
    scheduleAt( 0.01,msg);
}
void Node:: handleMessage(cMessage *msg)
{
    if(msg->isSelfMessage())
    {
        cMessage *out_msg = new cMessage();
        send(out_msg,"out");
        no_sent++;
        scheduleAt(simTime()+time_interval,msg);
    }
}
}
void Node:: finish()
{
    recordScalar("Number of received Message",no_rcvd);
    recordScalar("Number of sent Message",no_sent);
}

```

#### 5.4. Network design:

We build a simple network as it consist all the module as sub module under this network. It connects traffic packages, hub and sink in the network.

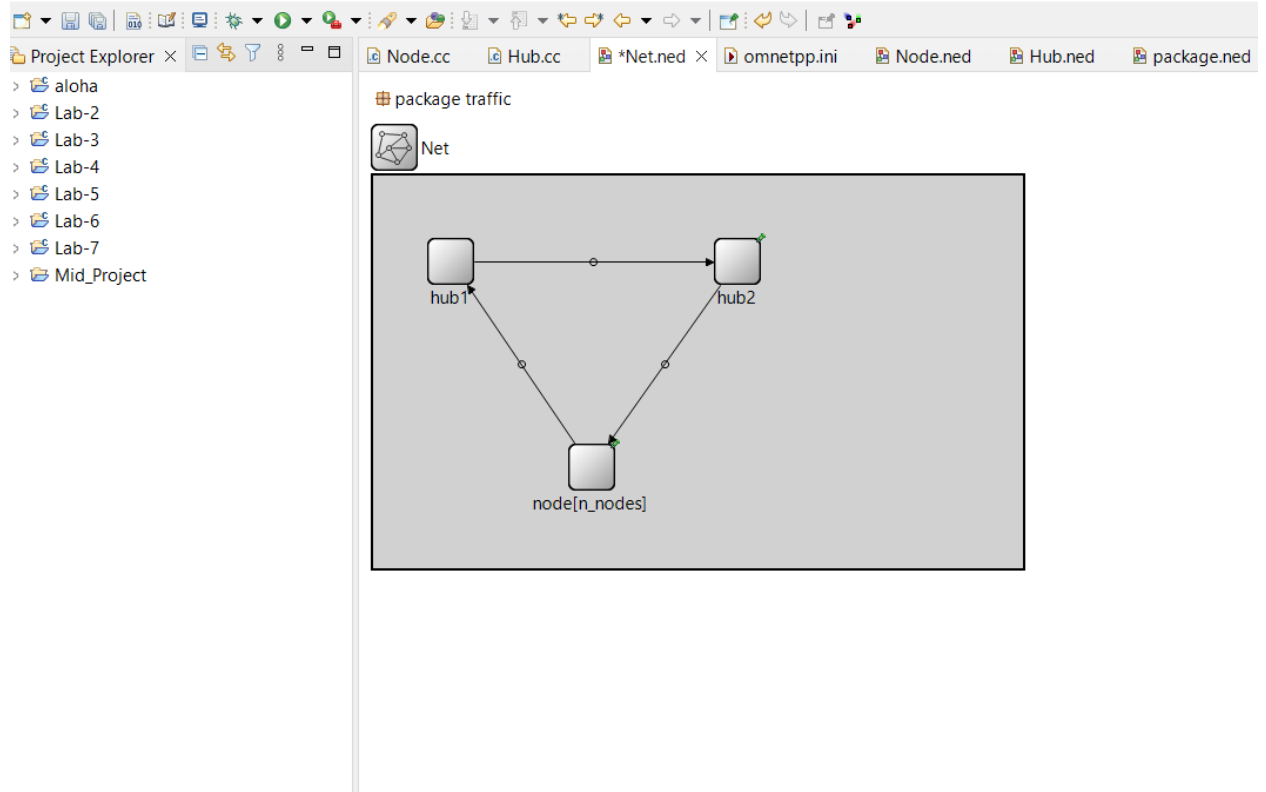


Figure – 5: Network Design

```

1
2
3 package traffic;
4
5 network Net
6 {
7   parameters:
8     int n_nodes;
9     // @display("bgb=628,253");
10    @display("bgb=734,318");
11   submodules:
12     hub1: Hub {
13       @display("p=77,59");
14     }
15     hub2: Hub {
16       @display("p=77,232");
17     }
18     node[n_nodes]: Node {
19       //@display("p=307,73");
20       @display("p=327,175");
21     }
22
23   connections:
24     for i=0..n_nodes-1 {
25       node[i].out --> hub1.in++;
26       hub1.out++ --> hub2.in++;
27       hub2.out++ --> node[i].in;
28     }
29
30 }

```

Figure – 6: Network Design ned file

## 5.5. Configurations: Completing all the necessities files then we build the project.

### \*INI File:

```

1 [General]src/Node.cc
2 network = traffic.Net
3 **.n_nodes = 3
4 [Config T_interval1]
5 **.node[*].time_interval=0.2
6
7
8

```

Figure – 7: INI file

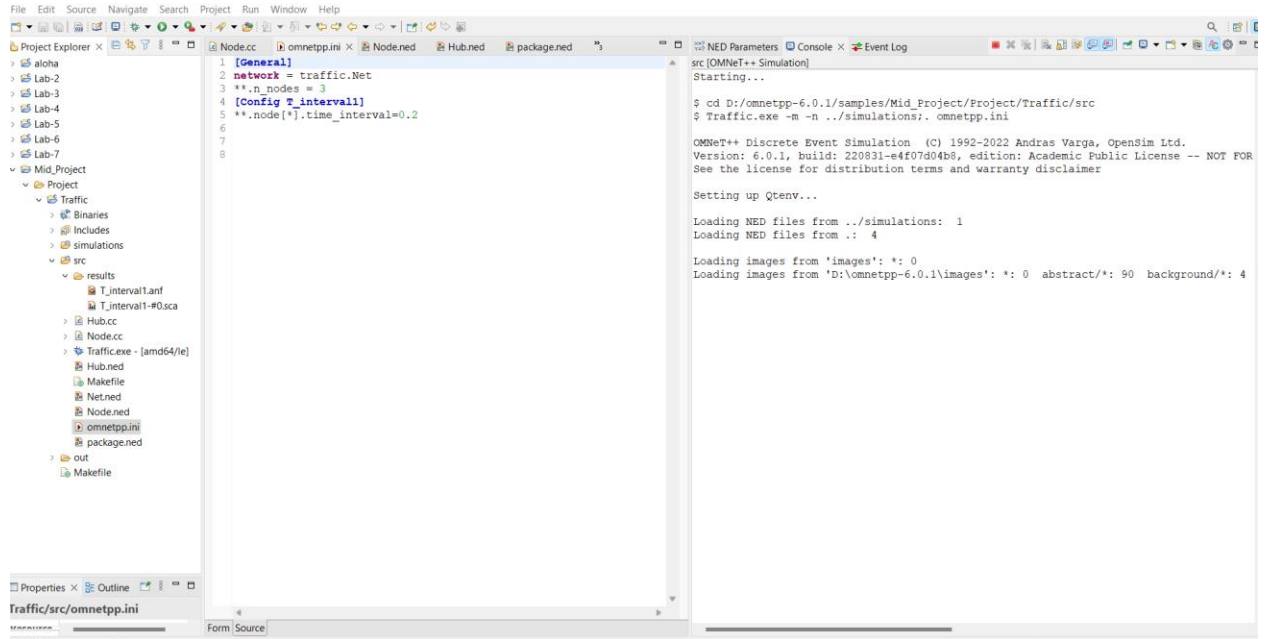


Figure – 8: Build the project

## 5.6. Run the simulation:

After build the network we run our project from ini file and it shows the result.

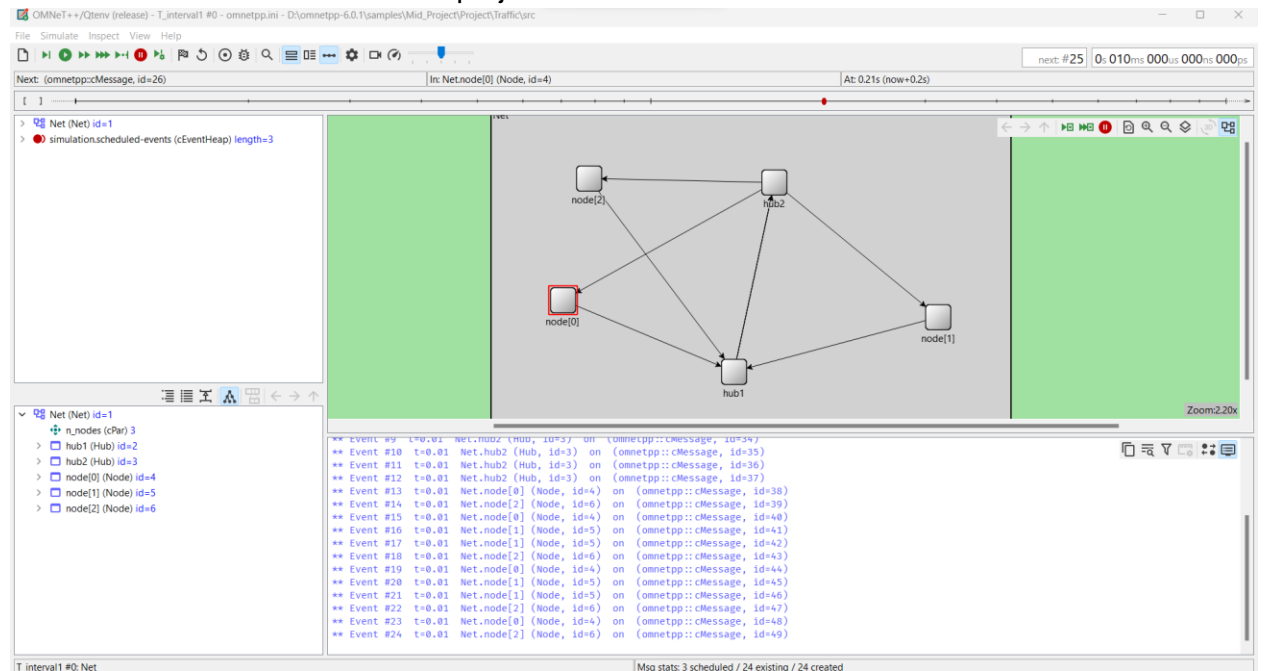


Figure – 9: Network simulation and run the project

## 6. Experimental and Theoretical Results:

After completing the simulation, we see that all nodes generates packets as message and send them in the first hub. As there are 3 nodes so first hub received 3 message from 3 nodes and send them into the second hub. After that second hub sends the messages in the all connected node. The number of sends and received executed by the sink. There include the statics of the following simulation.

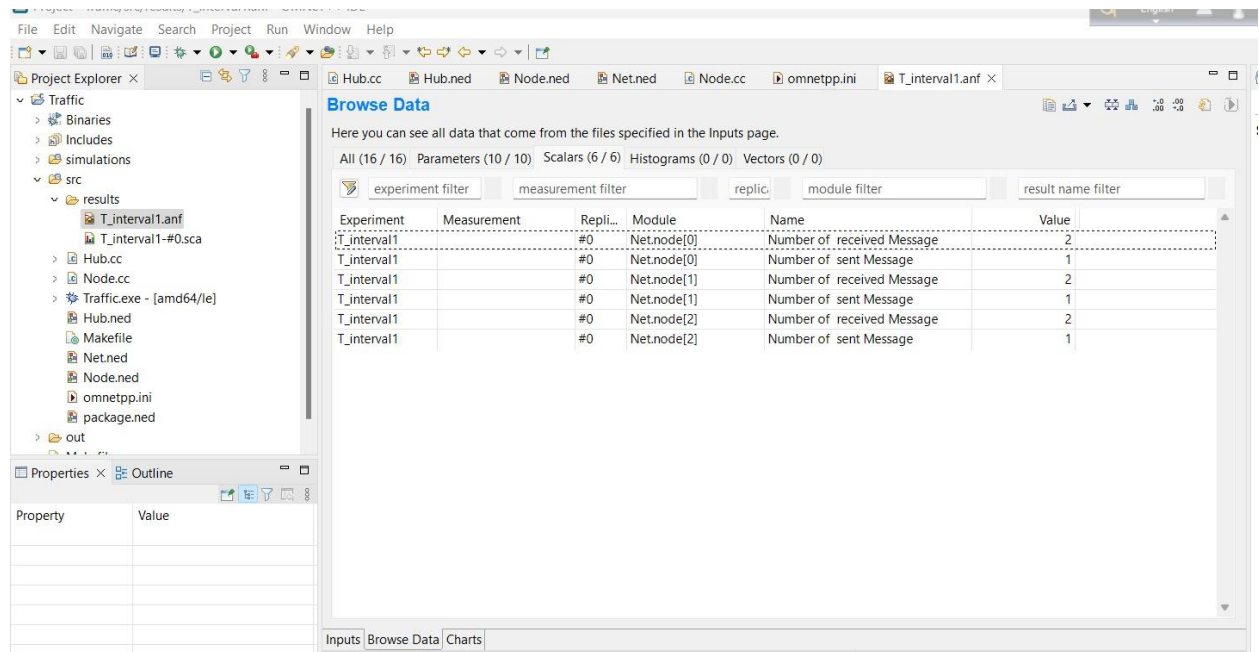


Figure – 10: Statics Data

### 6.1. Bar chart:

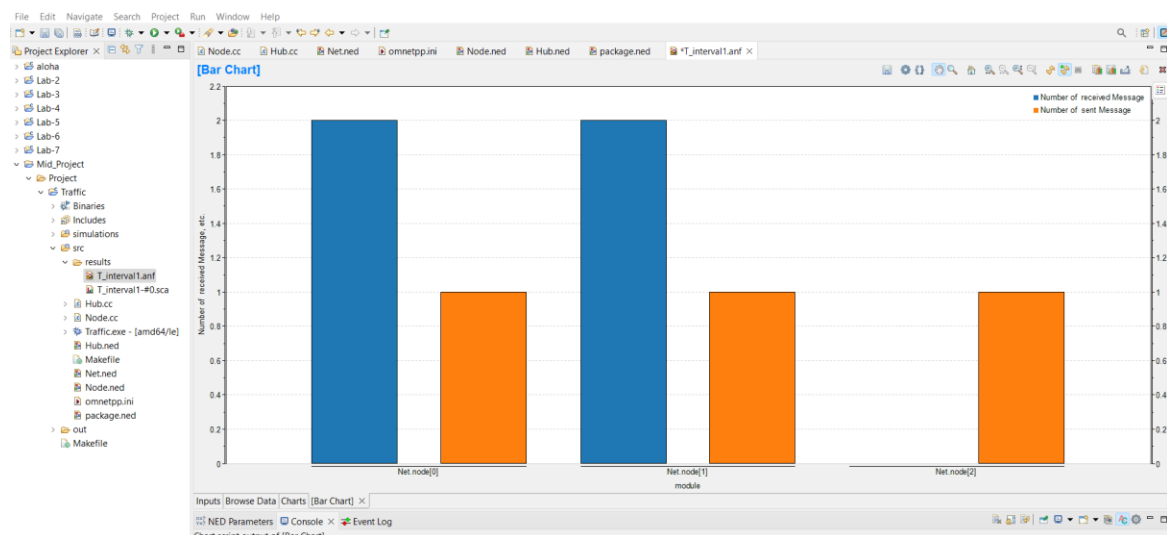


Figure – 11: Bar chart of simulation

## **6.2. Discussion on result:**

The results of our experiments and simulations indicate that the hub-based network topology has some limitations when compared to other network topologies such as switch-based networks. The major limitation of hub-based networks is the high collision rate, which results in a lower throughput. This is because a hub-based network allows all devices to share a common communication medium, and packets transmitted by multiple nodes may collide with each other.

In contrast, a switch-based network can transmit packets simultaneously on different ports, resulting in a lower collision rate and higher throughput. Therefore, it is more suitable for large-scale networks with a high traffic load.

However, the hub-based network topology can be a cost-effective solution for small-scale networks with a limited number of nodes. In such networks, the cost of switches and the complexity of the network can be reduced by using hubs. Additionally, for networks that require broadcasting of packets to all devices, such as for broadcasting messages or firmware updates, a hub-based network may be a better solution.

Overall, the choice of network topology depends on the specific requirements and constraints of the network, such as the number of nodes, traffic load, budget, and network complexity.

## **7. Future Work:**

There are several areas that can be explored in future work to further improve the performance of the hub-based network topology.

One possible direction is to investigate the use of advanced collision detection and avoidance algorithms, such as Carrier Sense Multiple Access with Collision Detection (CSMA/CD), to reduce the collision rate and improve the throughput. This may involve modifying the existing hub design or developing new hardware that can support these algorithms.

Another area that can be explored is the use of Quality of Service (QoS) techniques to prioritize traffic and ensure that critical packets are delivered with minimal delay. QoS can be particularly useful in networks that support real-time applications such as video conferencing or online gaming.

Additionally, the use of virtual LANs (VLANs) can be explored to improve network security and segment the network into smaller, more manageable subnets. This can help to reduce the impact of network failures and prevent unauthorized access to sensitive data.

Finally, the use of software-defined networking (SDN) can be explored to improve the flexibility and scalability of the network. SDN allows network administrators to centrally manage and configure the network, making it easier to adapt to changing network requirements and traffic patterns.

Overall, these areas of future work can help to further enhance the performance and capabilities of hub-based network topologies, making them a more viable option for a wide range of network applications.

## **8. Conclusions:**

Based on the results obtained from the simulation and analysis, we can conclude that the hub-based network architecture is not suitable for high-speed and high-bandwidth applications due to the high collision rate and low throughput. However, it can be useful for small and simple networks with low traffic.

In the future, we can explore more complex network topologies and use advanced algorithms and protocols to improve the network's performance. Additionally, we can also investigate the use of different network devices such as switches and routers to compare their performance with hubs. Overall, this project provided a good insight into the basics of network simulation and analysis and its relevance in modern-day networking.

## **9. References:**

- Omnet++ simulation environment. Available: <https://omnetpp.org/>
- S. K. Jha, "A Performance Comparison of Hub, Switch and Router Based Network Configurations using OPNET," International Journal of Computer Applications, vol. 101, no. 16, pp. 35-40, 2014.



