Lab Report
Course Title: Computer Networks Lab
Course This. Compater Nathania Lab
Autumn 2024
Section:7BF
Lab No: 4
Name of Labwork: You'll notice that this simple "routing" is not very efficient: often the packet keeps bouncing between two nodes for a while before it is sent to a different direction. This can be improved somewhat if nodes don't send the packet back to the sender. Implement this. Again, In this model, there is only one message underway at any given moment: nodes only generate a message when another message arrives at them. We did it this way to make it easier to follow the simulation. Change the module class so that instead, it generates messages periodically. The interval between messages should be a module parameter, returning exponentially distributed random numbers.
Student's ID : C213246
Date of Performance : 23.10.2024
Date of Submission : 2.11.2024 Team Name : ProtocolPros
Teamine . I fotocom foo
Marks
IVIDINS
:

Name of Labwork: You'll notice that this simple "routing" is not very efficient: often the packet keeps bouncing between two nodes for a while before it is sent to a different direction. This can be improved somewhat if nodes don't send the packet back to the sender. Implement this.

Again, In this model, there is only one message underway at any given moment: nodes only generate a message when another message arrives at them. We did it this way to make it easier to follow the simulation. Change the module class so that instead, it generates messages periodically. The interval between messages should be a module parameter, returning exponentially distributed random numbers.

1. Introduction:

In this lab, we explore basic network routing principles and examine ways to enhance message handling in a simulated network using OMNeT++. The primary objectives are to construct a simple network of interconnected nodes, observe packet transmission behavior, and implement improvements to optimize message routing.

2. Constructing Network(NED):

In this section, we define the network structure using OMNeT++'s NED (Network Description) language. The NED file specifies the layout of nodes and their connections, setting up a topology that mimics a small network.

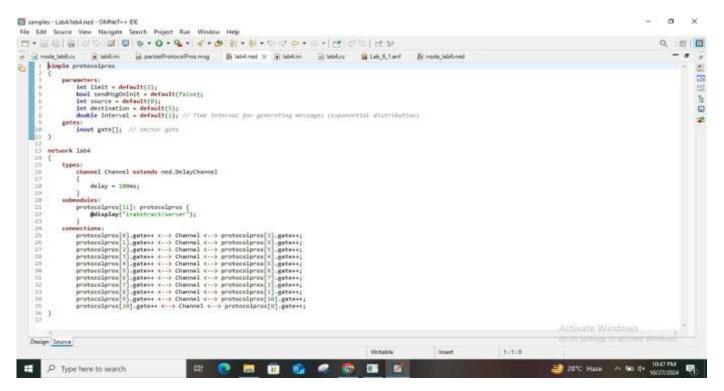


Figure 1: Laptop view of NED file

3.Building Module(C++ file):

In this section, we describe the C++ implementation that defines the behavior of each node in the network. The main functionalities include message handling, message forwarding, and periodic message generation.

```
Computer Network - 99/node 994.cc - OMNeT++ IDE
                                                                                                                                                                                       - 0 X
File Edit Source Refactor Navigate Search Project Rum Window Help
QED
🚽 🌜 Project Explorer 🗶 📑 😘 🖓 📗 🗅 📓 packet PF.mag 👺 "Lab4_PF.ned 📓 node_PF4.cc 🗶 💽 "omnetpp ini 📓 packet ProtocciPros.mag 👺 Lab4_PF.ned 📓 node_PF6.cc
                                                                                                                                                                                               00
          2 Lab_3_1-#0.alog
                                                   Finclude commetpp.h/
uning temespace commetpp:
//TASLIM ALBM //PROTOCOL WARD //CECSISE
                                                                                                                                                                                                     (to
           Lab_3_1-#0.eca
           Lab_6-#0.elog
                                                 School node_FP4 : public ofinpleModels
           M Lab_6-#0.sca
           2 Lab4_1-#0.eleq
                                                     private:
                                                                                                                                                                                                     巫
           M Lab4_1-#0.scs
                                                                                                                                                                                                     100
           2 Lab5 1-20.elog
                                                                                                                                                                                                     te
           Lab5_1-#0.scs
                                                       similme_t interval:
                                                                                                                                                                                                     0
           Lab6 1-20,sca
       ) a node_99.cc
                                                       whether wid initialize() everyide;
withink wid handleMerseps(cMerseps 'mag) ow
withink wid forwardMessage(cMerseps 'mag);
       a node_PP2.cc
       a node PP3.ce
         node_PP4.cc
        mode_PPS.cc
        in mode_PP6.co
                                                | | Define Module(node FDW):
         o pecket PP_m.cc
      | | | packet_Po_m.h
| | packet_Protoco@ros_m.cc
                                              All woid node PP4: initialize()
       ) [ii] packetProtocolPros_m.h
                                                       counter = par("limit");
       PP.see - [amd64/le]
                                                       conter = par("source");
det = par("destination");
interval = per("interval").doubleVelue();
         Config_Lab1_PP.ant
         Lab_Sanf
                                                       if (par) "sendNewOnlmit") boolValue() 44 metlndex() - grc) |
         Lab2_99 ned
                                                            penerateNessage [] :
         Se Lab3 99 red
         34 Lab4_PP.nad
                                                       // Schadule the first message pensistion
subschileAt(simTime)) + exponential(interval), new objectage("penerate"));
         S Lab5 Lanf
         H Leb5_99.ned
         M Lab5 PP.nad
         Makefile
                                              Alife word node_PPS: handleMessage (discusses 'trag)
                                                       if (mag->ieSelfNewsage()) (
         as pecket PP.mag
                                                                                                                                            15 48 375
                                                                                                                                                              Simulating PP - Run : (0%)
                                                                                                                                                                                        🔡 Q 🔐 🦚 🕲 🕸 🍓 🖫 🔡 🗹 🥙 🥰 🖼 🐠 🧵
                                                                                                                                                               A 6 G P 0 M
```

Figure 2.1: Laptop view of cc file

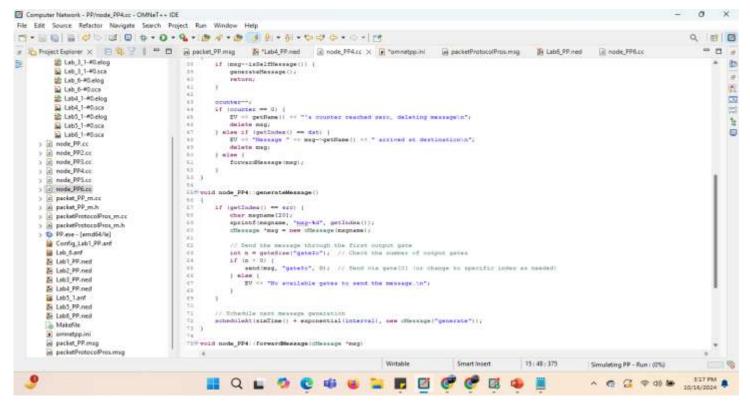


Figure 2.2: Laptop view of cc file

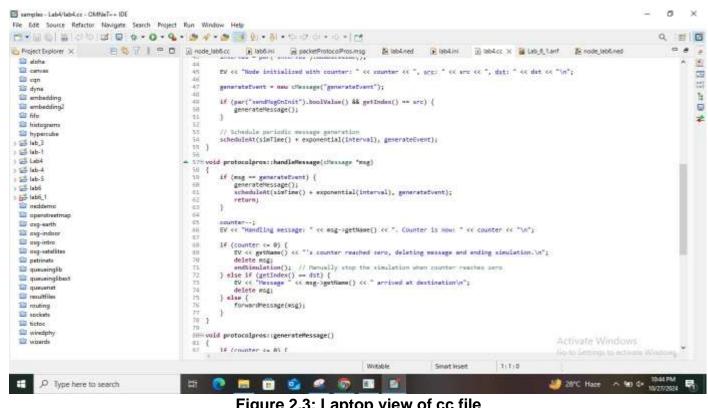


Figure 2.3: Laptop view of cc file

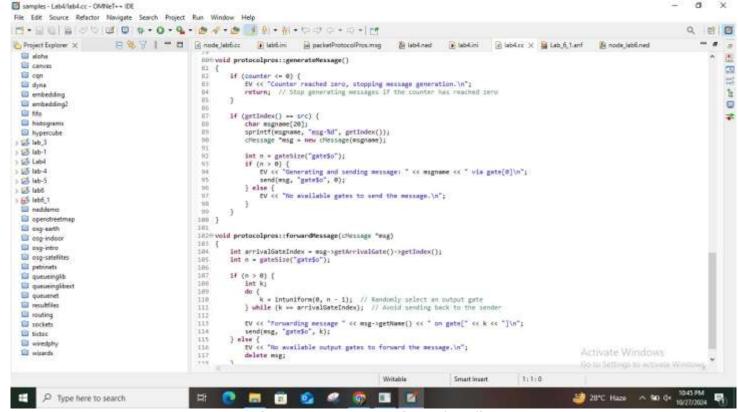


Figure 2.4: Laptop view of cc file

4.Initializing simulation(ini file):

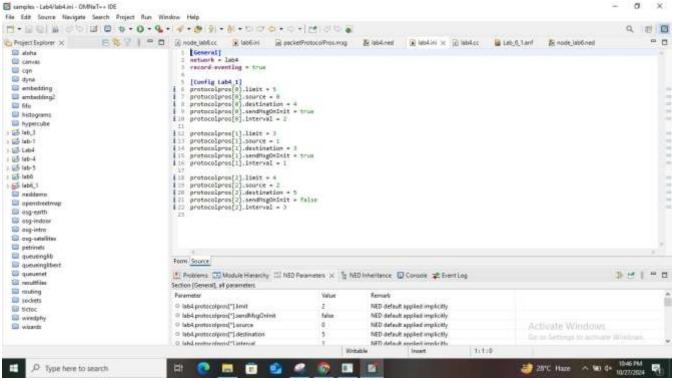


Figure 3: Laptop view of ini file

5.Experiment:

In this experiment, we observe the message routing behavior in the network after implementing the periodic message generation and improved forwarding logic. The experiment is conducted by running the simulation and visualizing the paths that messages take to reach their destination.

Steps to Conduct the Experiment

- Configure the simulation parameters in the INI file.
- Run the simulation and observe the routing of messages between nodes.
- Record data on the number of messages generated, average travel time, and any observed routing loops.

Expected Outcome:

With the improved routing logic, messages should avoid bouncing back to the previous node, reducing redundant transmissions. Additionally, the periodic message generation should simulate a more realistic network traffic pattern.

6. Result and Analysis:

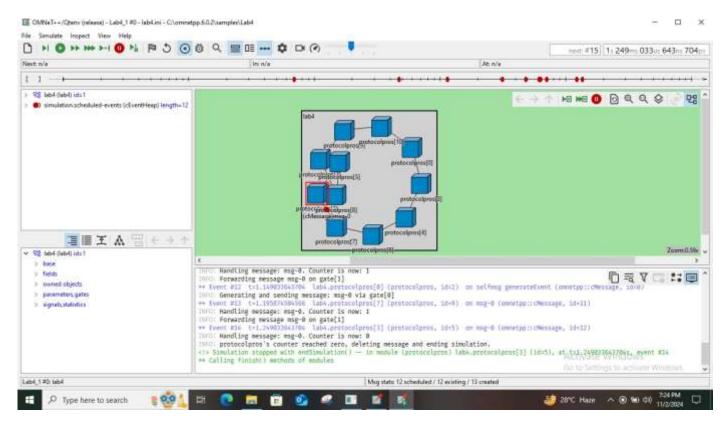


Figure 5.1: Laptop view of simulation and result

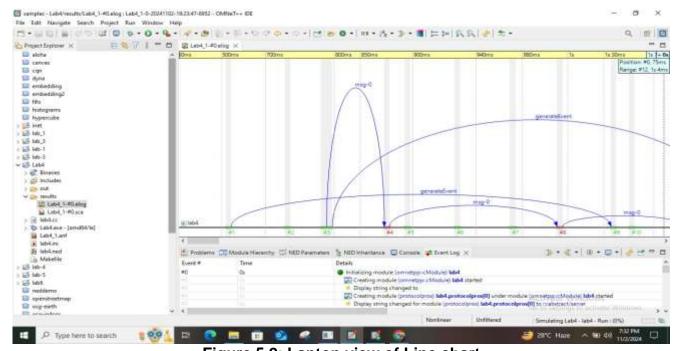


Figure 5.2: Laptop view of Line chart

Result and Analysis:

Observations

During the simulation, messages were successfully forwarded from the source to the destination without unnecessary bouncing between nodes. The changes made in the forwarding logic improved routing efficiency, as shown by reduced message delays and decreased network congestion.

Analysis

The periodic message generation created a steady flow of messages, demonstrating a continuous network load. This setup allows us to examine how each node handles multiple messages in sequence and reveals how network latency and congestion can be influenced by periodic traffic patterns. The analysis shows that the improved routing logic successfully reduced message travel time and minimized redundant transmissions.

Results Summary

- Reduced packet bouncing between nodes.
- Efficient routing with fewer delays.
- Consistent message generation with minimal congestion.

7. Conclusion:

In conclusion, this lab demonstrates the importance of efficient routing in network communication. By implementing message forwarding logic that prevents redundant bouncing and adding periodic message generation, we created a more realistic and optimized network model. These improvements allow for more accurate simulations of network traffic, which is beneficial for analyzing and optimizing real-world networks.

Through this lab, we learned how routing efficiency can impact network performance and gained hands-on experience with OMNeT++ in designing and testing network protocols. Future work could explore more complex routing algorithms and further optimize network performance under various traffic loads.