The information contained in this document represents the current view of TETCOS on the issues discussed as of the date of publication. Because TETCOS must respond to changing market conditions, it should not be interpreted to be a commitment on the part of TETCOS, and TETCOS cannot guarantee the accuracy of any information presented after the date of publication.

This manual is for informational purposes only. TETCOS MAKES NO WARRANTIES, EXPRESS, IMPLIED OR STATUTORY, AS TO THE INFORMATION IN THIS DOCUMENT.

Warning! DO NOT COPY

Copyright in the whole and every part of this manual belongs to TETCOS and may not be used, sold, transferred, copied or reproduced in whole or in part in any manner or in any media to any person, without the prior written consent of TETCOS. If you use this manual you do so at your own risk and on the understanding that TETCOS shall not be liable for any loss or damage of any kind.

TETCOS may have patents, patent applications, trademarks, copyrights, or other intellectual property rights covering subject matter in this document. Except as expressly provided in any written license agreement from TETCOS, the furnishing of this document does not give you any license to these patents, trademarks, copyrights, or other intellectual property. Unless otherwise noted, the example companies, organizations, products, domain names, e-mail addresses, logos, people, places, and events depicted herein are fictitious, and no association with any real company, organization, product, domain name, email address, logo, person, place, or event is intended or should be inferred.

Rev 12.1 (V), Feb 2020, TETCOS. All rights reserved.

All trademarks are property of their respective owner.

Contact us at

TETCOS

# 214, 39th A Cross, 7th Main, 5th Block Jayanagar,
Bangalore - 560 041, Karnataka, INDIA. Phone: +91 80 26630624
E-Mail: sales@tetcos.com
Visit: www.tetcos.com
LIST OF EXPERIMENTS

1. Introduction to NetSim.................................................................5

2. Understand working of ARP, and IP Forwarding within a LAN and across a router
........................................................................................................13

3. Simulate and study the spanning tree protocol..................................21

4. Introduction to TCP connection management..................................26

5. Reliable data transfer with TCP......................................................30

6. Mathematical Modelling of TCP Throughput Performance..............35

7. Study how throughput and error of a Wireless LAN network changes as the
distance between the Access Point and the wireless nodes is varied........42

8. WiFi: UDP Download Throughput..................................................49

9. How many downloads can a Wi-Fi access point simultaneously handle?.....60

10. TCP Congestion Control Algorithms..............................................67

11. Multi-AP Wi-Fi Networks: Channel Allocation ................................73

12. Plot the characteristic curve of throughput versus offered traffic for a Pure and
Slotted ALOHA system .........................................................................83

13. Study the working and routing table formation of Interior routing protocols, i.e.
Routing Information Protocol (RIP) and Open Shortest Path First (OSPF)......90

14. The M/D/1 Queue ...........................................................................98

15. Quality of Service (QoS) in 802.11e based WLANs ............................108

16. Analyze the performance of FIFO, Priority and WFQ Queuing Disciplines....112

17. Cyber physical systems (CPS) and IoT – An Introduction....................116

18. One Hop IoT Network over IEEE 802.15.4 ......................................122
19. IoT – Multi-Hop Sensor-Sink Path .............................................................................................................. 128

20. Study how call blocking probability varies as the load on a GSM network is continuously increased .................................................................................................................................................. 144

21. Study the 802.15.4 Superframe Structure and analyze the effect of Superframe order on throughput .................................................................................................................................................. 148

22. Understand the working of OSPF .................................................................................................................. 153

23. Understand the working of basic networking commands (Ping, Route Add/Delete/Print, ACL) .................................................................................................................................................. 163

24. Study how the throughput of LTE network varies as the distance between the ENB and UE (User Equipment) is increased .................................................................................................................................................. 173

25. Study how the throughput of LTE network varies as the Channel bandwidth changes in the ENB (Evolved node) .................................................................................................................................................. 178

26. Simulate and study LTE Handover procedure .................................................................................................. 182

27. Understand the working of LTE Device to Device Communication .................................................................. 190

28. To analyze how the allocation of frequency spectrum to the Incumbent (Primary) and CR CPE (Secondary User) affects throughput .................................................................................................................................................. 195

29. Understanding VLAN operation in L2 and L3 Switches .................................................................................. 201

30. Understanding Access and Trunk Links in VLANs .................................................................................................. 209

31. Understanding Public IP Address & NAT (Network Address Translation) .................................................. 216

32. Understand the events involved in NetSim DES (Discrete Event Simulator) in simulating the flow of one packet from a Wired node to a Wireless node .................................................................................................. 222

33. Understand the working of TCP BIC Congestion control algorithm, simulate and plot the TCP congestion window .................................................................................................................................................. 229

34. Simulating Link Failure ........................................................................................................................................ 234
1. Introduction to NetSim

1.1 Introduction to network simulation with NetSim, NetSim feature list and NetSim Simulation environment

NetSim is a network simulation tool that allows you to create network scenarios, model traffic, design protocols and analyze network performance. Users can study the behavior of a network by test combinations of network parameters. The various network technologies covered in NetSim include:

- Internetworks - Ethernet, WLAN, IP, TCP
- Legacy Networks - Aloha, Slotted Aloha
- Cellular Networks - GSM, CDMA
- Mobile Adhoc Networks - DSR, AODV, OLSR, ZRP
- Wireless Sensor Networks - 802.15.4
- Internet of Things - 6LoWPAN gateway, 802.15.4 MAC / PHY, RPL
- Cognitive Radio Networks - 802.22
- Long-Term Evolution Networks – LTE/LTE-A/LTE Femto Cell/LTE D2D/LTE Vanet
- Software Defined Networking
- Advanced Routing and Switching - VLAN, IGMP, PIM, L3 Switch, ACL and NAT
- 5G NR mmWave – LTE NR

NetSim home screen will appear as shown below:

- **Network Design Window**: NetSim design window or the GUI, enables users to model a network comprising of network devices like switches, routers, nodes, etc., connect them through links, and
model application traffic to flow through the network. The network devices shown are specific to the network technologies chosen by the user.

Description:

1. **File** - In order to save the network scenario before or after running the simulation into the current workspace,
   - Click on File → Save to save the simulation inside the current workspace. Users can specify their own Experiment Name and Description (Optional).
   - Click on File → Save As to save an already saved simulation in a different name after performing required modifications to it.
   - Click on Close, to close the design window or GUI. It will take you to the home screen of NetSim.

2. **Settings** - Go to Settings → Grid/Map Settings and choose the type of environment. Here we have chosen the Grid/Map in the form of a Grid. Map option can be used for specific cases like while designing VANET scenarios.

3. **Help** - Help option allows the users to access all the help features.
   - **About NetSim** – Assists the users with basic information like, Which version of NetSim is used and whether it is a 32-bit build or 64-bit build? What kind of License is being used? Whether Floating or Node Locked?
   - **Video Tutorials** – Assists the users by directing them to our dedicated YouTube Channel “TETCOS”, where we have lots of video presentations ranging from short to long, covering different versions of NetSim up to the latest release.
• **Answers/FAQ** – Assists the user by directing them to our “**NetSim Support Portal**”, where one can find a well-structured “**Knowledge Base**”, consisting of answers or solutions to all the commonest queries which a new user can go through.

• **Raise a Support Ticket** – Assists the user by directing them to our “**NetSim Support Portal**”, where one can “**Submit a ticket**” or in other words raise his/her query, which reaches our dedicated Helpdesk and due support will be provided to the user.

• **User Manual** – Assists the user with the usability of the entire tool and its features. It highly facilitates a new user with lots of key information about NetSim.

• **Source Code Help** – Assists the user with a structured documentation for “**NetSim Source Code Help**”, which helps the users who are doing their R&D using NetSim with a structured code documentation consisting of more than 5000 pages with very much ease of navigation from one part of the document to another.

• **Open Source Code** – Assists the user to open the entire source codes of NetSim protocol libraries in Visual Studio, where one can start initiating the debugging process or performing modifications to existing code or adding new lines of code. Visual Studio Community Edition is a highly recommended IDE to our users who are using the R&D Version of NetSim.

• **Experiments** – Assists the user with separate links provided for 30+ different experiments covering almost all the network technologies present in NetSim.

• **Technology Libraries** – Assists the user by directing them to a folder comprising of individual technology library files comprising all the components present in NetSim.

Below the menu options, the entire region constitutes the Ribbon/Toolbar using which the following actions can be performed:

- Click and drop network devices and right click to edit properties
- Click on Wired/Wireless links to connect the devices to one another. It automatically detects whether to use a Wired/Wireless link based on the devices we are trying to connect
- Click on Application to configure different types of applications and generate traffic
- Click on Plots, Packet Trace, and Event Trace and click on the enable check box option which appears in their respective windows to generate additional metrics to further analyze the network performance.
- Click on Run to perform the simulation and specify the simulation time in seconds.
- Next to Run, we have View Animation and View Results options. Both the options remain hidden before we run the simulation or if the respective windows are already open.
• Display Settings option is mainly used to display various parameters like Device Name, IP, etc., to provide a better understanding especially during the design and animation.

• **Results Window:** Upon completion of simulation, Network statistics or network performance metrics reported in the form of graphs and tables. The report includes metrics like throughput, simulation time, packets generated, packets dropped, collision counts etc.

![](image)

**Description:**

1. Below Simulation Results, clicking on a particular metrics will display the respective metrics window.
2. Clicking on links in a particular metrics will display the plot in a separate window
3. Enabling Detailed View by clicking on it will display the remaining properties
4. Clicking on Restore to Original View will get back to the original view
5. Click on Open Packet Trace / Open Event Trace to open the additional metrics which provide in depth analysis on each Packets / Events.

• **Packet Animation Window:** When we click on run simulation, we have the option to record / play & record animation. If this is enabled, users can view the animation during the run time or upon completion of the simulation users can see the flow of packets through the network. Along with this, more than 25+ fields of packet information is available as a table at the bottom. This table contains all the fields recorded in the packet trace. In addition, animation options are available for viewing different graphs, IP Addresses, Node movement etc.
Description:

1. Click on Play to view the animation. You can Pause the animation at any interval and Play again.
2. Click on Stop to stop the animation. Now click on Play to start the animation from the beginning.
3. Next to that we also have speed controllers to increase/decrease Simulation Time and Animation Speed
4. View More option enables the user to view Plots, Throughputs, and IP Tables during the animation
5. Table Filters are used to filter the packet information's shown in the below table during simulation as per user requirement
6. While setting more than one application, it is differentiated using different color indications
7. Packets are indicated using different color combinations say, blue color indicates control packets, green color indicates data packets and red color indicates error packets.

1.2 How does a user create and save an experiment in workspace?

To create an experiment, select New Simulation-> <Any Network> in the NetSim Home Screen.
Create a network and save the experiment by clicking on File->Save button on the top left.

A save popup window appears which contains Experiment Name, Folder Name, Workspace path and Description.
Specify the Experiment Name and Description (Optional) and then click on Save. The workspace path is non-editable. Hence all the experiments will be saved in the default workspace path. After specifying the Experiment Name click on Save.

In our example we saved with the name MANET and this experiment can be found in the default workspace path as shown below:

Users can also see the saved experiments in Open Simulation menu shown below:

“Save As” option is also available to save the current experiment with a different name.

1.3 Typical sequence of steps to do experiments in this manual

The typical steps involved in doing experiments in NetSim are,

- **Network Set up**: Drag and drop devices, and connect them using wired or wireless links
- **Configure Properties**: Configure device, protocol or link properties by right clicking on the device or link and modifying parameters in the properties window.
- **Model Traffic**: Click on the Application icon present on the ribbon and set traffic flows.
- **Enable Trace/Plots (optional):** Click on packet trace, event trace and Plots to enable. Packet trace logs packet flow, event trace logs each event (NetSim is a discrete event simulator) and the Plots button enables charting of various throughputs over time.

- **Save/Save As/Open/Edit:** Click on File → Save / File → Save As to save the experiments in the current workspace. Saved experiments can then opened from NetSim home screen to run the simulation or to modify the parameters and again run the simulation.

- **View Animation/View Results:** Visualize through the animator to understand working and to analyze results and draw inferences.

**NOTE:** Example Configuration files for all experiments would available where NetSim has been installed. This directory is (`<NetSim_Install_Directory>\Docs\Sample_Configuration\NetSim_Experiment_Manual`)
2. Understand working of ARP, and IP Forwarding within a LAN and across a router

2.1 Theory

In a network architecture different layers have their own addressing scheme. This helps the different layers in being largely independent. Application layer uses host names, network layer uses IP addresses and the link layer uses MAC addresses. Whenever a source node wants to send an IP datagram to a destination node, it needs to know the address of the destination. Since there are both IP addresses and MAC addresses, there needs to be a translation between them. This translation is handled by the Address Resolution Protocol (ARP). In IP network, IP routing involves the determination of suitable path for a network packet from a source to its destination. If the destination address is not on the local network, routers forward the packets to the next adjacent network.

(Reference: A good reference for this topic is Section 5.4.1: Link Layer Addressing and ARP, of the book, Computer Networking, A Top-Down Approach, 6th Edition by Kurose and Ross)

2.2 ARP protocol Description

1. ARP module in the sending host takes any IP address as input, and returns the corresponding MAC address.
2. First the sender constructs a special packet called an ARP packet, which contains several fields including the sending and receiving IP and MAC addresses.
3. Both ARP request and response packets have the same format.
4. The purpose of the ARP request packet is to query all the other hosts and routers on the subnet to determine the MAC address corresponding to the IP address that is being resolved.
5. The sender broadcasts the ARP request packet, which is received by all the hosts in the subnet.
6. Each node checks if its IP address matches the destination IP address in the ARP packet.
7. The one with the match sends back to the querying host a response ARP packet with the desired mapping.
8. Each host and router has an ARP table in its memory, which contains mapping of IP addresses to MAC addresses.
9. The ARP table also contains a Time-to-live (TTL) value, which indicates when each mapping will be deleted from the table.
2.3 ARP Frame Format

<table>
<thead>
<tr>
<th>Hardware Type</th>
<th>Protocol Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hardware Address Length</td>
<td>Protocol address length</td>
</tr>
<tr>
<td>Opcode</td>
<td></td>
</tr>
</tbody>
</table>

sender Hardware Address

Sender Protocol Address(1-2)  Sender Protocol Address(3-4)

Target hardware Address

Target Protocol Address

The ARP message format is designed to accommodate layer two and layer three addresses of various sizes. This diagram shows the most common implementation, which uses 32 bits for the layer three ("Protocol") addresses, and 48 bits for the layer two hardware addresses.

2.4 IP Forwarding Description

1. Every router has a forwarding table that maps the destination addresses (or portions of the destination addresses) to that router’s outbound links.
2. A router forwards a packet by examining the value of a field in the arriving packet’s header, and then using this header value to index into the router’s forwarding table.
3. The value stored in the forwarding table entry for that header indicates the router’s outgoing link interface to which that packet is to be forwarded.
4. Depending on the network-layer protocol, the header value could be the destination address of the packet or an indication of the connection to which the packet belongs.
5. ARP operates when a host wants to send a datagram to another host on the same subnet.
6. When sending a Datagram off the subnet, the datagram must first be sent to the first-hop router on the path to the final destination. The MAC address of the router interface is acquired using ARP.
7. The router determines the interface on which the datagram is to be forwarded by consulting its forwarding table.
8. Router obtains the MAC address of the destination node using ARP.
9. The router sends the packet into the respective subnet from the interface that was identified using the forwarding table.
2.5 Network Set up

Open NetSim and click Examples > Experiments > Working-of-ARP-and-IP-Forwarding-within-a-LAN-and-across-a-router > Sample-1 as shown below:

NetSim UI displays the configuration file corresponding to this experiment as shown below:

2.6 Procedure:

The following set of procedures were done to generate this sample:

**Step 1:** A network scenario is designed in NetSim GUI comprising of 3 Wired Nodes, 2 L2 Switches, and 1 Router in the “Internetworks” Network Library.
Step 2: Right click on the Application Flow App1 CBR and select Properties or click on the Application icon present in the top ribbon/toolbar.

A CBR Application is generated from Wired Node 1 i.e. Source to Wired Node 2 i.e. Destination with Packet Size remaining 1460Bytes and Inter Arrival Time remaining 20000µs.

Transport Protocol is set to UDP instead of TCP. If set to TCP, the ARP table will get updated due to the transmission of TCP control packets thereby eliminating the need for ARP to resolve addresses.

Step 3: Packet Trace is enabled in the NetSim GUI, and hence we can view the ARP Request and ARP Reply packets exchanged initially, before transmission of the data packets.

Step 4: Click on Run simulation. The simulation time is set to 10 seconds. In the “Static ARP Configuration” tab, Static ARP is set to disable.

Click on Accept and then click on OK.

If Static ARP is enabled, then NetSim will automatically create an ARP table for each node. To see the working of the ARP protocol users should disable Static ARP.

By doing so, ARP request would be sent to the destination to find out the destinations MAC Address.

2.7 Output – I:

Once the simulation is complete, to view the packet trace file, click on “Open Packet Trace” option present in the left-hand-side of the Results Dashboard.
NODE 1 will send ARP_REQUEST to SWITCH-4, SWITCH-4 sends this to ROUTER-6, and SWITCH-4 also sends this to NODE-2. ARP-REPLY is sent by the NODE-2 to SWITCH -4, and in-turn SWITCH-4 sends it to NODE-1.

2.8 Inference I:

**Intra-LAN-IP-forwarding:**

**ARP PROTOCOL- WORKING:**

![ARP Protocol Diagram](image)

NODE-1 broadcasts ARP_Request, which is then broadcasted by SWITCH-4. NODE-2 sends the ARP_Reply to NODE-1 via SWITCH-4. After this step, datagrams are transmitted from NODE-1 to NODE-2. Notice the DESTINATION_ID column for ARP_Request type packets, which indicates Broadcast-0.

>Sample-2:

NetSim UI displays the configuration file corresponding to this experiment as shown below:

![NetSim UI Configuration](image)

2.9 Procedure:

The following set of procedures were done to generate this sample.
**Step 1:** A network scenario is designed in the NetSim GUI comprising of 3 Wired Nodes, 2 L2 Switches, and 1 Router.

**Step 2:** Right click on the Application Flow App1 CBR and select Properties or click on the Application icon present in the top ribbon/toolbar.

A CBR Application is generated from Wired Node 1 i.e. Source to Wired Node 3 i.e. Destination with Packet Size remaining 1460Bytes and Inter Arrival Time remaining 20000µs.

Transport Protocol is set to UDP instead of TCP. If set to TCP, the ARP table will get updated due to the transmission of TCP control packets thereby eliminating the need for ARP to resolve addresses.

**Step 3:** Packet Trace is enabled in the NetSim GUI, and hence we can view the ARP Request and ARP Reply packets exchanged initially, before transmission of the data packets.

**Step 4:** Click on Run simulation. The simulation time is set to 10 seconds. In the “Static ARP Configuration” tab, Static ARP is set to disable.

### 2.10 Output – II:

Once the simulation is complete, to view the packet trace file, click on “Open Packet Trace” option present in the left-hand-side of the Results Dashboard.

<table>
<thead>
<tr>
<th>PACKET ID</th>
<th>SEGMENT ID</th>
<th>PACKET_TYPE</th>
<th>CONTROL PACKET_TYPE/APP NAME</th>
<th>SOURCE_ID</th>
<th>DESTINATION_ID</th>
<th>TRANSMITTER_ID</th>
<th>RECEIVER_ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 N/A</td>
<td>Control_Packet</td>
<td>ARP_Request</td>
<td>NODE-1</td>
<td>Broadcast-0</td>
<td>NODE-1</td>
<td>SWITCH-4</td>
<td>ROUTER-6</td>
</tr>
<tr>
<td>0 N/A</td>
<td>Control_Packet</td>
<td>ARP_Request</td>
<td>NODE-1</td>
<td>Broadcast-0</td>
<td>SWITCH-4</td>
<td>NODE-2</td>
<td></td>
</tr>
<tr>
<td>0 N/A</td>
<td>Control_Packet</td>
<td>ARP_Request</td>
<td>ROUTER-6</td>
<td>NODE-1</td>
<td>ROUTER-6</td>
<td>SWITCH-4</td>
<td></td>
</tr>
<tr>
<td>0 N/A</td>
<td>Control_Packet</td>
<td>ARP_Request</td>
<td>ROUTER-6</td>
<td>NODE-1</td>
<td>NODE-1</td>
<td>SWITCH-4</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>CBR</td>
<td>App1_CBR</td>
<td>NODE-1</td>
<td>NODE-3</td>
<td>NODE-1</td>
<td>SWITCH-4</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>CBR</td>
<td>App1_CBR</td>
<td>NODE-1</td>
<td>NODE-3</td>
<td>NODE-3</td>
<td>SWITCH-4</td>
<td></td>
</tr>
<tr>
<td>0 N/A</td>
<td>Control_Packet</td>
<td>ARP_Request</td>
<td>ROUTER-6</td>
<td>Broadcast-0</td>
<td>ROUTER-6</td>
<td>SWITCH-5</td>
<td></td>
</tr>
<tr>
<td>0 N/A</td>
<td>Control_Packet</td>
<td>ARP_Request</td>
<td>ROUTER-6</td>
<td>Broadcast-0</td>
<td>SWITCH-5</td>
<td>NODE-3</td>
<td></td>
</tr>
<tr>
<td>0 N/A</td>
<td>Control_Packet</td>
<td>ARP_Request</td>
<td>NODE-3</td>
<td>ROUTER-6</td>
<td>NODE-3</td>
<td>SWITCH-5</td>
<td></td>
</tr>
<tr>
<td>0 N/A</td>
<td>Control_Packet</td>
<td>ARP_Request</td>
<td>NODE-3</td>
<td>ROUTER-6</td>
<td>SWITCH-5</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

NODE 1 will send ARP_REQUEST to SWITCH-4, SWITCH-4 sends this to ROUTER-6, and SWITCH-4 also sends this to NODE-2. ARP-REPLY is sent by the ROUTER-6 to SWITCH-4, and in-turn SWITCH-4 sends it to NODE-1. Again ROUTER-6 will send ARP_REQUEST to SWITCH-5, SWITCH-5 sends this to NODE-3. ARP_REPLY is sent by NODE-3 to SWITCH-5 and in-turn SWITCH-5 sends it to ROUTER-6.

The IP forwarding table formed in the router can be accessed from the IP_Forwarding_Table list present in the Simulation Results window as shown below:
Click on Detailed View checkbox to view the additional fields as indicated above.

Router forwards packets intended to the subnet 11.2.0.0 to the interface with the IP 11.2.1.1 based on the first entry in its routing table.

2.11 Inference II:

**Across-Router-IP-forwarding:**

**ARP PROTOCOL- WORKING**
NODE-1 transmits ARP_Request which is further broadcasted by SWITCH-4. ROUTER-6 sends ARP_Reply to NODE-1 which goes through SWITCH-4. Then NODE-1 starts sending datagrams to NODE-3. If router has the MAC address of NODE-3 in its ARP table, then ARP ends here and router starts forwarding the datagrams to NODE-3 by consulting its forwarding table. In the other case, Router sends ARP_Request to appropriate subnet and after getting the MAC address of NODE-3, it then forwards the datagrams to NODE-3 using its forwarding table.
3. Simulate and study the spanning tree protocol

3.1 Introduction:

Spanning Tree Protocol (STP) is a link management protocol. Using the spanning tree algorithm, STP provides path redundancy while preventing undesirable loops in a network that are created by multiple active paths between stations. Loops occur when there are alternate routes between hosts. To establish path redundancy, STP creates a tree that spans all of the switches in an extended network, forcing redundant paths into a standby, or blocked state. STP allows only one active path at a time between any two network devices (this prevents the loops) but establishes the redundant links as a backup if the initial link should fail. Without spanning tree in place, it is possible that both connections may simultaneously live, which could result in an endless loop of traffic on the LAN.

(Reference: A good reference for this topic is Section 3.1.4: Bridges and LAN switches, of the book, Computer Networks, 5th Edition by Peterson and Davie)

3.2 Network Setup:

Open NetSim and click Examples > Experiments > Simulate-and-study-the-spanning-tree-protocol > Sample-1 as shown below:

NetSim UI displays the configuration file corresponding to this experiment as shown below:
Note: At least three L2 Switches are required in the network to analyze the spanning tree formation.

3.3 Procedure:

Step 1: A network scenario is designed in the NetSim GUI comprising of 3 Wired Nodes and 3 L2 Switches in the “Internetworks” Network Library.

Step 2: Go to L2 Switch 1 Properties. In the Interface 1 (ETHERNET) > Datalink Layer, “Switch Priority” is set to 2. Similarly, for the other interfaces of L2 Switch 1, Switch Priority is set to 2.

Step 3: Go to L2 Switch 2 Properties. In the Interface 1 (ETHERNET) > Datalink Layer, “Switch Priority” is set to 1. Similarly, for the other interfaces of L2 Switch 2, Switch Priority is set to 1.

Step 4: Go to L2 Switch 3 Properties. In the Interface 1 (ETHERNET) > Datalink Layer, “Switch Priority” is set to 3. Similarly, for the other interfaces of L2 Switch 3, Switch Priority is set to 3.

<table>
<thead>
<tr>
<th>L2_Switch Properties</th>
<th>L2_Switch 1</th>
<th>L2_Switch 2</th>
<th>L2_Switch 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Switch Priority</td>
<td>2</td>
<td>1</td>
<td>3</td>
</tr>
</tbody>
</table>

Note: Switch Priority is set to all the 3 L2 Switches and Switch Priority has to be changed for all the interfaces of L2 Switch.

Switch Priority is interpreted as the weights associated with each interface of a L2 Switch. A higher value indicates a higher priority.
**Step 5:** Right click on the Application Flow “App1 CUSTOM” and select Properties or click on the Application icon present in the top ribbon/toolbar.

A CUSTOM Application is generated from Wired Node 4 i.e. Source to Wired Node 5 i.e. Destination with Packet Size remaining 1460Bytes and Inter Arrival Time remaining 20000µs. Additionally, the “Start Time” parameter is set to 1 second while configuring the application.

![Configure Application](image)

Note: Wired Node 6 is not generating traffic to any other nodes.

Here, Wired Node 4 is sending data to Wired Node 5 and the node properties are set to default.

**Step 6:** Click on Run simulation. The simulation time is set to 10 seconds.

> **Sample-2:**

The following changes in settings are done from the previous sample:

In Sample 2, the “Switch Priority” of all the 3 L2 Switches are changed as follows:

<table>
<thead>
<tr>
<th>L2_Switch Properties</th>
<th>L2_Switch 1</th>
<th>L2_Switch 2</th>
<th>L2_Switch 3</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
3.4 Output:

In the NetSim Design Window, click on Display Settings > Spanning Tree check box.

**Sample 1:**

Go to NetSim Packet Animation Window and click on Play button. We can notice that, after the exchange of control packets, the data packets take the following path. **Wired Node 4 > L2 Switch 1 > L2 Switch 2 > L2 Switch 3 > Wired Node 5.**

**Sample 2:**
Go to NetSim Packet Animation window and click on Play button. We can notice that, after the exchange of control packets, the data packets take the following path. Wired Node 4 > L2 Switch 1 > L2 Switch 3 > Wired Node 5.

Go to Simulation Results window, In the left-hand-side of the Results Dashboard, click on the arrow pointer of Switch MAC address table to obtain the Switch MAC address table list of all the L2 Switches.

For each L2 Switch, a Switch MAC Address Table containing the MAC address entries, the port that is used for reaching it, along with the type of entry can be obtained at the end of Simulation.

![Switch MAC address table](image)

### 3.5 Inference:

Each L2 Switch has an ID which is a combination of its Lowest MAC address and priority. The Spanning tree algorithm selects the L2 Switch with the smallest ID as the root node of the Spanning Tree. The root node forward frames out over all its ports. In the other L2 Switches, the ports that have the least cost of reaching the root switch are set as **Forward Ports** and the remaining are set as **Blocked Ports**. In the Sample 1, L2_Switch 2 was assigned least priority and was selected as a Root Switch. The green line indicates the forward path and the red line indicates the blocked path. The frame from Wired Node 4 should take the path through the L2_Switch 1, 2 and 3 to reach the Wired Node 5. In the Sample 2, L2_Switch 1 was assigned least priority and selected as a Root switch. In this case, the frame from Wired Node 4 takes the path through the L2_Switch 1 and 3 to reach the destination Wired Node 5.
4. Introduction to TCP connection management

4.1 Introduction:

When an application process in a client host seeks a reliable data connection with a process in another host (say, server), the client-side TCP then proceeds to establish a TCP connection with the TCP at the server side. A TCP connection is a point-to-point, full-duplex logical connection with resources allocated only in the end hosts. The TCP connection between the client and the server is established in the following manner, and is illustrated in Fig 1.

1. The TCP at the client side first sends a special TCP segment, called the SYN packet, to the TCP at the server side.
2. Upon receiving the SYN packet, the server allocates TCP buffer and variables to the connection. Also, the server sends a connection-granted segment, called the SYN-ACK packet, to the TCP at the client side.
3. Upon receiving the SYN-ACK segment, the client also allocates buffers and variables to the connection. The client then acknowledges the server’s connection granted segment with an ACK of its own.

This connection establishment procedure is often referred to as the three-way handshake. The special TCP segments can be identified by the values in the fields SYN, ACK and FIN in the TCP header (see Fig 2). We also note that the TCP connection is uniquely identified by the source and destination port numbers (see Fig 2) exchanged during TCP connection establishment and the source and destination IP addresses.

Once a TCP connection is established, the application processes can send data to each other. The TCP connection can be terminated by either of the two processes. Suppose that the client application process seeks to terminate the connection. Then, the following handshake ensures that the TCP connection is torn down.

1. The TCP at the client side sends a special TCP segment, called the FIN packet, to the TCP at the server side.
2. When the server receives the FIN segment, it sends the client an acknowledgement segment in return and its own FIN segment to terminate the full-duplex connection.
3. Finally, the client acknowledges the FIN-ACK segment (from the server) with an ACK of its own. At this point, all the resources (i.e., buffers and variables) in the two hosts are deallocated.

During the life of a TCP connection, the TCP protocol running in each host makes transitions through various TCP states. Fig 1 illustrates the typical TCP states visited by the client and the server during connection establishment and data communication.

**Fig 1 TCP connection establishment between a client and a server**

<table>
<thead>
<tr>
<th>32 bits</th>
<th>Source port #</th>
<th>Destination port #</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Sequence number</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Acknowledgement number</td>
<td></td>
</tr>
<tr>
<td>Header length</td>
<td>Unused</td>
<td>C</td>
</tr>
<tr>
<td></td>
<td>w</td>
<td>C</td>
</tr>
<tr>
<td>Internet checksum</td>
<td>URGENT data pointer</td>
<td></td>
</tr>
<tr>
<td>Options</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Data</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Fig 2 TCP Header**
4.2 Network Setup:

Open NetSim and click **Examples > Experiments > Introduction-to-TCP-connection-management > Sample-1** as shown below:

![NetSim UI](image)

NetSim UI displays the configuration file corresponding to this experiment as shown below:

![Network Configuration](image)

4.3 Procedure:

The following set of procedures were done to generate this sample.

**Step 1:** A network scenario is designed in NetSim GUI comprising of 2 Wired Nodes and 2 Routers in the “Internetworks” Network Library.

**Step 2:** In the General Properties of Wired Node 1 i.e. Source, Wireshark Capture is set to Online.

*Note:* Accept default properties for Routers as well as the Links.
Step 3: Right-click the link ID (of a wired link) and select Properties to access the link's properties. Set Max Uplink Speed and Max Downlink Speed to 10 Mbps. Set Uplink BER and Downlink BER to 0. Set Uplink Propagation Delay and Downlink Propagation Delay as 100 microseconds for the links 1 and 3 (between the Wired Node's and the routers). Set Uplink Propagation Delay and Downlink Propagation Delay as 50000 microseconds for the backbone link connecting the routers, i.e., 2.

Step 4: Right click on the Application Flow App1 FTP and select Properties or click on the Application icon present in the top ribbon/toolbar.

An FTP Application is generated from Wired Node 1 i.e. Source to Wired Node 2 i.e. Destination with File Size set to 14600 Bytes and File Inter Arrival Time set to 10 Seconds.

Step 5: Click on Display Settings > Device IP check box in the NetSim GUI to view the network topology along with the IP address.

Step 6: Click on Run simulation. The simulation time is set to 10 seconds.

4.4 Output:

We have enabled Wireshark capture in Wired Node 1. The PCAP file is generated at the end of the simulation and is shown in Fig 3.

1. The 3-way handshake of TCP connection establishment and TCP connection termination is observed in the packet capture (Figure 3).
2. Data is transferred only after the TCP connection is established.
3. We can access the packet header details of the TCP segments (SYN, SYN-ACK, FIN, FINACK) in Wireshark.

![Fig 3 Packet capture at Wired_Node_1](image_url)
5. Reliable data transfer with TCP

5.1 Introduction:

TCP provides reliable data transfer service to the application processes even when the underlying network service (IP service) is unreliable (loses, corrupts, garbles or duplicates packets). TCP uses checksum, sequence numbers, acknowledgements, timers and retransmission to ensure correct and in order delivery of data to the application processes.

TCP views the data stream from the client application process as an ordered stream of bytes. TCP will grab chunks of this data (stored temporarily in the TCP send buffer), add its own header and pass it on to the network layer. A key field of the TCP header is the sequence number which indicates the position of the first byte of the TCP data segment in the data stream. The sequence number will allow the TCP receiver to identify segment losses, duplicate packets and to ensure correct delivery of the data stream to the server application process.

When a server receives a TCP segment, it acknowledges the same with an ACK segment (the segment carrying the acknowledgement has the ACK bit set to 1) and also conveys the sequence number of the first missing byte in the application data stream, in the acknowledgement number field of the TCP header. All acknowledgements are cumulative, hence, all missing and out-of-order TCP segments will result in duplicate acknowledgements for the corresponding TCP segments.

TCP sender relies on sequence numbering and acknowledgements to ensure reliable transfer of the data stream. In the event of a timeout (no acknowledgement is received before the timer expires) or triple duplicate acknowledgements (multiple ACK segments indicate a lost or missing TCP segment) for a TCP segment, the TCP sender will retransmit the segment until the TCP segment is acknowledged (at least cumulatively). In Figure 1, we illustrate retransmission by the TCP sender after a timeout for acknowledgement.
Fig 4 An illustration of TCP retransmission with timeout. The segment with sequence number 4381 is lost in the network. The TCP client retransmits the segment after a timeout event.

5.2 Network Setup:

We will seek a simple file transfer with TCP over a lossy link to study reliable data transfer with TCP. We will simulate the network setup illustrated in Fig 5 with the configuration parameters listed in detail in Table 1 to study reliable data transfer with TCP connection.

Open NetSim and click Examples > Experiments > Reliable-data-transfer-with-TCP > Sample-1 as shown below:
NetSim UI displays the configuration file corresponding to this experiment as shown below:

![Fig 5 A client and a server network architecture](image)

5.3 Procedure:

The following set of procedures were done to generate this sample.

**Step 1**: A network scenario is designed in NetSim GUI comprising of 2 Wired Nodes and 2 Routers in the “Internetworks” Network Library.

**Step 2**: In the General Properties of Wired Node 1 i.e. Source and Wired Node 2 i.e. Destination, Wireshark Capture is set to Online.

*Note: Accept default properties for Routers as well as the Links.*

**Step 3**: Right-click the link ID (of a wired link) and select Properties to access the link’s properties. Set Max Uplink Speed and Max Downlink Speed to 10 Mbps. Set Uplink BER and Downlink BER to 0. Set Uplink Propagation Delay and Downlink Propagation Delay as 100 microseconds for the links 1 and 3 (between the Wired Node’s and the routers). Set Uplink Propagation Delay and Downlink Propagation Delay as 50000 microseconds and Uplink BER and Downlink BER to 0.00001 for the backbone link connecting the routers, i.e., 2.

**Step 4**: Right click on the Application Flow App1 FTP and select Properties or click on the Application icon present in the top ribbon/toolbar.

An FTP Application is generated from Wired Node 1 i.e. Source to Wired Node 2 i.e. Destination with File Size set to 14600 Bytes and File Inter Arrival Time set to 20 Seconds.

**Step 5**: Click on Display Settings > Device IP check box in the NetSim GUI to view the network topology along with the IP address.

**Step 6**: Click on Run simulation. The simulation time is set to 20 seconds.
5.4 Output:

We aimed to transfer a file of size 14600 bytes (i.e., 10 packets, each of size 1460 bytes) with TCP over a lossy link. In Fig 6, we report the application metrics data for FTP which indicates that the complete file was transferred.

![Fig 6 Application metrics data for FTP](image)

We have enabled WireShark Capture in Wired Node 1 and Wired Node 2. The PCAP files are generated at the end of the simulation and are shown in Fig 7.

![Fig 7 PCAP file at Wired Node 1. TCP ensures reliable data transfer using timeout, duplicate ACKs and retransmissions](image)
5.5 Inference:

1. From Fig 7 and Fig 3, we note that the packets with sequence number 2961, 5921, and 8881 are lost in the network.

2. After receiving three duplicate ACKs (in lines 13, 14 of Fig 7), TCP retransmits the lost packet with sequence number 2961 (in line 15 of Fig 7). After a timeout (see lines 17, 21, 22 and 23, lines 25 and 26), TCP retransmits the lost packet with sequence numbers 5921 and 8881.

3. TCP connection is terminated only after the complete file transfer is acknowledged.
6. Mathematical Modelling of TCP Throughput Performance

6.1 Introduction:

The average throughput performance of additive-increase multiplicative-decrease TCP congestion control algorithms have been studied in a variety of network scenarios. In the regime of large RTT, the average throughput performance of the TCP congestion control algorithms can be approximated by the ratio of the average congestion window $cwnd$ and RTT.

6.1.1 Loss-less Network

In a loss-less network, we can expect the TCP congestion window $cwnd$ to quickly increase to the maximum value of 64 KB (without TCP scaling). In such a case, the long-term average throughput of TCP can be approximated as

$$Throughput \approx \frac{64 \times 1024 \text{ (bits)}}{RTT \text{ (in secs)}}$$

6.1.2 Lossy Network

We refer to an exercise in Chapter 3 of Computer Networking: A top-down approach, by Kurose and Ross for the setup. Consider a TCP connection over a lossy link with packet error rate $p$. In a period of time between two packet losses, the congestion window may be approximated to increase from an average value of $W/2$ to $W$ (see Fig 15 for motivation). In such a scenario, the throughput can be approximated to vary from $W/2/RTT$ to $W/RTT$ (in the cycle between two packet losses). Under such assumptions, we can then show that the loss rate (fraction of packets lost) must be equal to

$$p = \frac{1}{\frac{3}{8}W^2 + \frac{3}{4}W}$$

and the average throughput is then approximately,

$$Throughput \approx \sqrt{\frac{3}{2p} \times \frac{MSS \text{ (in bits)}}{RTT \text{ (in secs)}}}$$

6.2 Network Setup:

We will seek a large file transfer with TCP over a loss-less and lossy link to study long-term average throughput performance of the TCP congestion control algorithm. We will simulate the network setup illustrated in Fig 9 with the two (loss-less and lossy) configuration.
parameters listed in detail in Table 1 to study the throughput performance of TCP New Reno.

Open NetSim and click Examples > Experiments > Mathematical-model-of-TCP-throughput-performance > Sample-1 as shown below:

![NetSim Interface](image)

NetSim UI displays the configuration file corresponding to this experiment as shown below:

![Configuration File](image)

**Fig 9 A client and a server network architecture.**

### 6.3 Procedure:

**Sample 1**

The following set of procedures were done to generate this sample.

**Step 1:** A network scenario is designed in NetSim GUI comprising of 2 Wired Nodes and 2 Routers in the “Internetworks” Network Library.
Step 2: In the General Properties of Wired Node 1 i.e. Source, Wireshark Capture is set to Online.

*Note: Accept default properties for Routers as well as the Links.*

Step 3: Right-click the link ID (of a wired link) and select Properties to access the link’s properties. Set Max Uplink Speed and Max Downlink Speed to 10 Mbps. Set Uplink BER and Downlink BER to 0. Set Uplink Propagation Delay and Downlink Propagation Delay as 100 microseconds for the links 1 and 3 (between the Wired Node’s and the routers). Set Uplink Propagation Delay and Downlink Propagation Delay as 50000 microseconds for the backbone link connecting the routers, i.e., 2.

Step 4: Right click on the Application Flow App1 CBR and select Properties or click on the Application icon present in the top ribbon/toolbar.

An CBR Application is generated from Wired Node 1 i.e. Source to Wired Node 2 i.e. Destination with Packet Size set to 1460 Bytes and Inter Arrival Time set to 1168 microseconds.

Step 5: Click on Display Settings > Device IP check box in the NetSim GUI to view the network topology along with the IP address.

Step 6: Click on Plots icon and select the Enable Plots checkbox. This enables us to view the throughput plot of the application App1 CBR.

Step 7: Click on Run simulation. The simulation time is set to 100 seconds. In the “Static ARP Configuration” tab, Static ARP is set to disable.

Sample 2

Step 1: Right-click the link ID (of a wired link) and select Properties to access the link’s properties. Set Max Uplink Speed and Max Downlink Speed to 10 Mbps. Set Uplink BER and Downlink BER to 0. Set Uplink Propagation Delay and Downlink Propagation Delay as 100 microseconds for the links 1 and 3 (between the Wired Node’s and the routers). Set Uplink Propagation Delay and Downlink Propagation Delay as 50000 microseconds and Uplink BER and Downlink BER to 0.0000001 for the backbone link connecting the routers, i.e., 2.

Step 2: Click on Run simulation. The simulation time is set to 100 seconds. In the “Static ARP Configuration” tab, Static ARP is set to disable.

6.4 Output:

In Fig 10, we report the application metrics data for data transfer over a loss-less link (Sample-1).
In **Fig 10**, we report the plot of long-term average throughput of the TCP connection over the loss-less link.

![Application Metrics Table]

**Table:** Application Metrics when BER=0

<table>
<thead>
<tr>
<th>Application Id</th>
<th>Throughput Plot</th>
<th>Application Name</th>
<th>Packet generated</th>
<th>Packet received</th>
<th>Throughput (Mbps)</th>
<th>Delay (microsec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Application_throughput_plot</td>
<td>App1_CBR</td>
<td>85617</td>
<td>42055</td>
<td>4.912024</td>
<td>25657072.446895</td>
</tr>
</tbody>
</table>

**Fig 11** Long-term average throughput of TCP New Reno over a loss-less link

We have enabled WireShark Capture in the Wired Node 1. The PCAP file is generated at the end of the simulation. From the PCAP file, the congestion window evolution graph can be obtained as follows. In Wireshark, select any data packet with a left click, then, go to Statistics > TCP Stream Graphs > Window Scaling. In **Fig 12**, we report the congestion window evolution of TCP New Reno over the loss-less link.
**Fig 12** Congestion window evolution with TCP New Reno over a loss-less link.

In Fig 13, we report the application metrics data for data transfer over a lossy link (Sample-2).

<table>
<thead>
<tr>
<th>Application Id</th>
<th>Throughput Plot</th>
<th>Application Name</th>
<th>Packet generated</th>
<th>Packet received</th>
<th>Throughput (Mbps)</th>
<th>Delay (microsec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Application_throughput_plot</td>
<td>App1_CBR</td>
<td>85617</td>
<td>32589</td>
<td>3.065395</td>
<td>31884960.015729</td>
</tr>
</tbody>
</table>

**Fig 13** Application Metrics when $BER = 1e^{-7}$

In Fig 14, we report the plot of long-term average throughput of the TCP connection over the lossy link.
In Fig 15, we report the congestion window evolution of TCP New Reno over the lossy link.

Fig 15 Congestion window evolution with TCP New Reno over a lossy link

6.5 Observations and Inference

1. In Fig 12, we notice that the congestion window of TCP (over the loss-less link) increases monotonically to 64 KB and remains there forever. So, a block of 64 KBs of data is transferred over a round-trip time (RTT) of approximately 100 milliseconds. Hence, a good approximation of the TCP throughput over the loss-less link is
Throughput $\approx \frac{\text{Window Size (in bits)}}{\text{RTT (in secs)}}$

$= \frac{65535 \times 8}{100 \times 10^{-3}} = 5.24 \text{ Mbps}$

We note that the observed long-term average throughput (see Fig 10) is approximately equal to the above computed value.

2. In Fig 15, for the lossy link with $BER = 1e^{-7}$, we report the congestion window evolution with New Reno congestion control algorithm. The approximate throughput of the TCP New Reno congestion control algorithm for a packet error rate $p$, TCP segment size MSS and round-trip time RTT

$$Throughput \approx \sqrt{\frac{3}{2p} \times \frac{\text{MSS (in bits)}}{\text{RTT (in secs)}}}$$

$\approx \sqrt{\frac{3}{2 \times 1.2 \times 10^{-3}} \times \frac{1460 \times 8}{100 \times 10^{-3}}}$

$= 4.12 \text{ Mbps}$

where the packet error rate $p$ can be computed from the bit error rate ($BER = 1e^{-7}$) and PHY layer packet length (1500 bytes, see packet trace) as

$$p = 1 - (1 - BER)^{1500 \times 8} \approx 1.2e^{-3}$$

We note that the observed long-term average throughput (see Table 11) is approximately equal to the above computed value.
7. Study how throughput and error of a Wireless LAN network changes as the distance between the Access Point and the wireless nodes is varied

7.1 Introduction

In this experiment we will study the physical layer standard for IEEE 802.11b WiFi. A physical layer standard (abbreviated as PHY standard) defines the mechanism by which logical information bits are transmitted over the wireless channel that has been allotted to the WiFi system. WiFi systems are confined to working in an approximately 80MHz bandwidth in the 2.4GHz ISM band. Within this bandwidth, any particular WiFi Access Point (AP) must choose to work in one of 13 channels, each of nominal bandwidth 22MHz. In this experiment, we aim to study how the packet error performance of an IEEE 802.11b AP-STA connection varies as the distance between the AP and the STA varies.

7.2 Background

The IEEE 802.11b standard defines 4 digital modulation schemes for such channels. All are based on Direct Sequence Spread Spectrum (DSSS) with a chipping rate of 11 million chips per second (11 Mcps). An 11 chip Barker code yields 1 million symbols per second (1 Msps). These symbols are Differential Phase Shift Keying modulated to get 1 bit per symbol, thereby yielding 1 Mbps, and Quartenary Differential Phased Keying modulated to get 2 bits per symbol, thereby yielding 2 Mbps. In order to get 5.5 Mbps and 11 Mbps, each symbol is made from 8 chips, so that the symbol rate is 1.375 Msps. A technique called Complementary Code Keying (CCK) then provides 4 bits per symbol, yielding 5.5 Mbps, and 8 bits per symbol, which yields 11 Mbps.

A simple qualitative fact is that, for a given signal to noise ratio at the receiver, as the modulation scheme attempts to send more bits per second, the bit error probability increases. This happens because, as the bit rate increases, the bit sequences that the receiver needs to distinguish between become closely packed, so that bit errors become harder to resolve. The signal to noise ratio (SNR) at the receiver depends on the transmission power, the attenuation of power from the transmitter to the receiver, and noise power.

\[
SNR = \frac{P_{\text{received}}}{N_0W}
\]

Where \(N_0\) is the noise power spectral density (W/Hz) and \(W\) is the system bandwidth (nominally 22 MHz). The noise power works out to approximately \(-100\) dBm. The received power \(P_{\text{received}}\) is obtained by subtracting the path-loss, between the transmitter and the receiver, from the transmitted
power (e.g., $P_{\text{transmit}} = 0 \text{ dBm}$, would arise from a transmit power of 1 mW). A simple expression for path-loss is given by

$$P_{\text{received}} = P_{\text{transmit}} - c_0 - 10 \eta \log_{10} d$$

where $c_0$ is the path loss at the “reference” distance of 1m, $\eta$ is the path-loss exponent and $d$ is the distance between the transmitter and the receiver. It may be noted that this deterministic expression ignores random phenomena such as “shadowing” and “fading.”

As $d$ increases, the received power decreases; e.g., doubling the distance reduced the received power by approximately $3\eta$, since $\log_{10} 2 \approx 0.3$. Typical values of $\eta$, indoors, could be 3 to 5, resulting in 9 dB to 15 dB additional path loss for doubling the value of $d$.

### 7.3 Network Setup

Open NetSim and click **Examples > Experiments > Impact-of-distance-on-Wifi-throughput-and-error > Sample-1** as shown below:

NetSim UI displays the configuration file corresponding to this experiment as shown below:
7.4 Procedure:

The following set of procedures were done to generate this sample.

**Step 1:** A network scenario is created in NetSim GUI comprising of 1 Wired Node, 1 Router, 1 Access Point and 1 Wireless Node in the “Internetworks” Network Library.

**Step 2:** In the Destination Node, i.e. Wireless Node 4, the Interface 1 (WIRELESS) > Physical Layer, Protocol Standard is set to IEEE802.11b and in the Interface 1 (WIRELESS) > Datalink Layer, Rate Adaptation is set to False.

**Step 3:** The position of the Wireless Node and the Access Point in the grid environment is set according to the values given in the below table.

<table>
<thead>
<tr>
<th>Wireless Node 4 Properties</th>
<th>Access Point</th>
</tr>
</thead>
<tbody>
<tr>
<td>X/Lon</td>
<td>200</td>
</tr>
<tr>
<td>Y/Lat</td>
<td>30</td>
</tr>
<tr>
<td></td>
<td>0</td>
</tr>
</tbody>
</table>

**Step 4:** Right-click the link ID (of a wired/wireless link) and select Properties to access the link’s properties. The parameters are set according to the values given in the below table.

<table>
<thead>
<tr>
<th>Wired Link Properties</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Max Uplink Speed (Mbps)</td>
<td>100</td>
</tr>
<tr>
<td>Max Downlink Speed (Mbps)</td>
<td>100</td>
</tr>
<tr>
<td>Uplink BER</td>
<td>0.0000001</td>
</tr>
<tr>
<td>Downlink BER</td>
<td>0.0000001</td>
</tr>
<tr>
<td>Uplink Propagation Delay (µs)</td>
<td>5</td>
</tr>
<tr>
<td>Downlink Propagation Delay (µs)</td>
<td>5</td>
</tr>
</tbody>
</table>
### Wireless Link Properties

<table>
<thead>
<tr>
<th>Channel Characteristics</th>
<th>Path Loss Only</th>
</tr>
</thead>
<tbody>
<tr>
<td>Path Loss Model</td>
<td>Log Distance</td>
</tr>
<tr>
<td>Path Loss Exponent</td>
<td>3</td>
</tr>
</tbody>
</table>

**Step 5:** Right click on **App1 CBR** and select Properties or click on the Application icon present in the top ribbon/toolbar.

A CBR Application is generated from Wired Node 2 i.e. Source to Wireless Node 4 i.e. Destination with Packet Size set to 1450 Bytes and Inter Arrival Time set to 770 µs. It is set such that, the **Generation Rate** equals to 15 Mbps.

Transport Protocol is set to **UDP** instead of TCP.

**Step 6:** Packet Trace is enabled in NetSim GUI. At the end of the simulation, a very large .csv file is containing all the packet information is available for the users to perform packet level analysis.

Similarly do the other samples by varying the distance between Access Point and Wireless Node as 60, 85, 90, 100, 110, 115, 180, 260, 360, 400, 420, 440, 460, 480, up to 500 m.

### 7.5 Output:

Note down the values of Data rate and Throughput for all the samples and compare them with IEEE standards

Phy rate can be calculated from packet trace by using the formula given below:

\[ \text{Phy rate (802.11b)} = \text{Phy\_layer\_payload} * 8 / (\text{phy\ end\ time} - \text{phy\ arrival\ time} - 192) \]

192 micro seconds is the approximate preamble time for 802.11b

Calculate PHY rate for all the data packets coming from Access Point to Wireless node. For doing this please refer section 8.5.1 How to set filters to NetSim Packet Trace file from NetSim’s User Manual. Filter Packet Type to CBR, Transmitter to Access Point and Receiver to Wireless node.

Since \( \text{PER} = 1 - (1 - \text{BER})^{PL} \) where PER is packet error rate, PL is packet length in bits and BER is bit error rate, we get \( \text{BER} = 1 - e^{\log(1 - \text{PER}) / \text{PL}} \)

**Packet error probability = Packets Errored/Packets Transmitted**

On tabulating the results, you would see
<table>
<thead>
<tr>
<th>Distance (m)</th>
<th>PHY rate in Mbps (Channel capacity)</th>
<th>Application Throughput (Mbps)</th>
<th>Packets Transmitted</th>
<th>PacketsErrored</th>
<th>Packet error probability</th>
</tr>
</thead>
<tbody>
<tr>
<td>30</td>
<td>11</td>
<td>5.9276</td>
<td>5110</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>60</td>
<td>11</td>
<td>5.9276</td>
<td>5110</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>85</td>
<td>11</td>
<td>5.842920</td>
<td>5101</td>
<td>64</td>
<td>0.0125</td>
</tr>
<tr>
<td>90</td>
<td>11</td>
<td>5.53204</td>
<td>5063</td>
<td>294</td>
<td>0.058</td>
</tr>
<tr>
<td>100</td>
<td>5.5</td>
<td>3.78856</td>
<td>3266</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>110</td>
<td>5.5</td>
<td>3.78856</td>
<td>3266</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>115</td>
<td>5.5</td>
<td>3.64588</td>
<td>3253</td>
<td>110</td>
<td>0.033</td>
</tr>
<tr>
<td>180</td>
<td>2</td>
<td>1.6762</td>
<td>1445</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>260</td>
<td>2</td>
<td>1.6762</td>
<td>1445</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>360</td>
<td>2</td>
<td>1.66808</td>
<td>1445</td>
<td>7</td>
<td>0.004</td>
</tr>
<tr>
<td>400</td>
<td>2</td>
<td>1.48016</td>
<td>1436</td>
<td>160</td>
<td>0.110</td>
</tr>
<tr>
<td>420</td>
<td>1</td>
<td>0.89204</td>
<td>769</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>440</td>
<td>1</td>
<td>0.89204</td>
<td>769</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>460</td>
<td>1</td>
<td>0.89204</td>
<td>769</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>480</td>
<td>1</td>
<td>0.89204</td>
<td>769</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>500</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Data Rate vs. Distance

![Graph showing the relationship between distance and data rate](image-url)
7.6 Inference:

We notice that as the distance increases, the 802.11b PHY rate (channel capacity decreases) decreases. This is because the underlying data rate depends on the received power at the receiver.

\[
\text{Received Power} = \text{Transmitted Power} - \text{RF losses}
\]

RF losses are directly proportional to distance to the power of path loss exponent. As RF propagation losses increase, the received power decreases.
We can see that the rate drops from 11 Mbps to 5.5 Mbps at around 95 m, and then to 2 Mbps at 175 m and to 1 Mbps at 415 m (in this case the path loss exponent is set to 3.0). We also notice how the packet error rate increases with distance, then when the data rate changes (a lower modulation scheme is chosen), the error rate drops. This happens for all the transitions i.e. 11 to 5.5, 5.5 to 2 and from 2 to 1 Mbps. One must note that WLAN involves ACK packets after data transmission. These additional packet transmission lead to reduced Application throughput of 5.9 Mbps (at lower distances) even though the PHY layer data rate is 11 Mbps and the error rates is almost NIL. The application throughput is dependent on the PHY rate and the channel error rate, and one can notice it drops / rise accordingly.
8. WiFi: UDP Download Throughput

8.1 The Setup and Motivation

The most basic packet transfer service offered by the Internet is called the “datagram” service, in which a series of packets are transmitted to a receiver without any packet loss recovery, flow control, or congestion control. The Internet’s UDP protocol implements the datagram service. In this experiment we will study the performance of UDP transfers from a server on a wireline local area network to WiFi Stations (STA), via WiFi Access Points (AP). The schematic of the network that we will be simulating in NetSim is shown in the figure below:

The server, which contains the data that needs to be transferred to the STAs (say, laptops), is connected by a 100 Mbps switched Ethernet link to an Ethernet switch, which is, in turn, connected to the WiFi APs. Each AP is associated (i.e., connected) at 11 Mbps to a single STA. The objective is to transfer a large number of packets (say, constituting a video) from the server to each of the STAs, the packet stream to each of the STAs being different (e.g., each STA is receiving a different video from the server). In this experiment, we are interested in studying the limitation that the WiFi link places on the data transfers. We assume that the server transmits the packets at a high enough rate so that the queues at the APs fill up, and the rate of the UDP transfers is, therefore, governed by the WiFi link. It may be noted that, in practice, there will be a flow control mechanism between each STA and the server, that will control the rate at which the server releases packets, in order to prevent buffer overflow at the APs.
In this setting, this experiment will ask one precise question. With the buffers at the AP full, at what rate will the WiFi protocol transfer the packets from the APs to the STAs over the wireless link. We will study two cases:

1. A single AP and a single STA: Since there is only one transmitter in this wireless network (namely, the AP), there is no contention, and the rate of packet transfer over the link will be governed by the basic overheads in the protocol, such as the interframe spacings, packet header overheads, transmit-receive turn-around times, and acknowledgement times. We will begin by a simple calculation (essentially timing book-keeping) that will predict the UDP throughput, and then we will verify our calculation using the NetSim simulator.

2. Multiple APs and one STA for each AP: This is the more common situation (for example neighboring apartments in a building, each with one AP and one laptop, all drawing data from the Internet service provider). The performance of such a system depends on the wireless propagation path-loss between the various APs. A predictive analysis is difficult in the general case. For deriving some insight, we will study the case where all the APs are close to each other, and thus exactly one transmission from AP to an STA can be successful at any time. If two or more APs transmit together, then all the transmissions are not successful. Even in this case, the analysis mathematically complex and is available in, Anurag Kumar, D. Manjunath and Joy Kuri. 2008: Wireless Networking. Sec 7.4

### 8.2 Predicting the UDP Throughput

#### 8.2.1 One AP and one STA

As stated above, in the setup described, the AP queue is full. Thus, after a packet is completely transmitted over the wireless link, immediately the process for transmitting the next packet starts. This is illustrated by the upper part of the figure below, where the successive packets from the AP are shown as being sent back-to-back. The time taken to send a packet is, however, not just the time to clock out the physical bits corresponding to the packet over the Wi-Fi medium. After the completion of a packet transfer, the AP’s Wi-Fi transmitter waits for a Distributed Coordination Function Inter-Frame Space (DIFS), followed by a backoff that is chosen randomly between 1 and 32 slots. Upon the completion of the backoff, the packet transmission starts. Each packet carries physical layer overheads, MAC layer overheads, and IP overheads. After the transmission of the packet, there is a Short
Inter-Frame Space (SIFS), which gives time to the receiver (namely, the STA) to transition from the listening mode to the transmit mode. After the SIFS, the STA sends back a MAC acknowledgement (ACK). This completes the transmission of one UDP packet from the AP to the STA. Immediately, the process for sending the next packet can start. The details of the various timings involved in sending a single UDP packet are shown in the lower part of the figure below.

![Diagram showing the timing of packet transmission]

In this experiment, the payload in each packet is the same (1450 Bytes). Since the packets are sent back to back, and the state of the system is identical at the beginning of each packet transmission, the throughput (in Mbps) is computed by the following simple (and intuitive) relation.

### 8.2.1.1 Without RTS / CTS

\[
\text{UDP Throughput (Mbps)} = \frac{\text{Application Payload in Packet (bits)}}{\text{Average Time per Packet(µs)}}
\]

\[
\text{Average time per packet (µs)} = \text{DIFS} + \text{Average Backoff time} + \text{Packet Transmission Time} + \text{SIFS} + \text{Ack Transmission Time}
\]

\[
\text{Packet Transmission Time (µs)} = \text{Preamble time} + (\text{MPDU Size/Data rate})
\]

\[
\text{Average Backoff time (µs)} = (\text{CWmin}/2) \times \text{Slot Time}
\]

\[
\text{Ack Transmission Time (µs)} = \text{Preamble time} + (\text{Ack Packet size/Ack data rate})
\]

\[
\text{DIFS (µs)} = \text{SIFS} + 2 \times \text{Slot Time}
\]

\[
\text{Average Backoff time (µs)} = (\text{CWmin}/2) \times \text{Slot Time}
\]

where
\[ SIFS = 10\ \mu s\ and\ Slot\ Time = 20\ \mu s \]

\[ CW_{\text{min}} = 31\ \text{slots for 802.11b} \]

\[ DIFS = SIFS + 2 \times \text{Slot Time} = 10\ \mu s + 2 \times 20\ \mu s = 50\ \mu s \]

Average Backoff Time = 310\ \mu s

\[ \text{Packet Transmission Time} = 192\ \mu s + \frac{1518 \times 8\ \text{bits}}{11\ Mbps} = 1296\ \mu s \]

Preamble time – 192\ \mu s for 802.11b

\[ \text{MPDU Size} = 1450 + 8 + 20 + 40 = 1518\ \text{Bytes} \]

\[ \text{Ack Transmission Time} = 192\ \mu s + \frac{14\ \text{Bytes} \times 8}{1\ Mbps} = 304\ \mu s \]

Application Payload = 1450\ \text{Bytes}

\[ \text{Average time per packet} = 50 + 310 + 1296 + 10 + 304 = 1970\ \mu s \]

\[ \text{UDP Throughput} = \frac{1450 \times 8}{1970} = 5.92\ Mbps \]

**8.2.1.2 With RTS/CTS**

\[ \text{UDP Throughput (Mbps)} = \frac{\text{Application Payload in Packet (bits)}}{\text{Average Time per Packet(\mu s)}} \]

Average time per packet (\mu s)

\[ = DIFS + \text{RTS Packet Transmission Time} + \text{CTS Packet Transmission Time} + \text{Average Backoff time} + \text{Packet Transmission Time} + \text{SIFS} \]

\[ + \text{Ack Transmission Time} \]

\[ \text{RTS packet transmission time} = \text{Preamble time} + \left( \frac{\text{RTS Packet payload}}{\text{Data Rate}} \right) \]

\[ = 192 + 20 \times \left( \frac{8}{1} \right) = 352\ \mu s \]

\[ \text{CTS packet transmission time} = \text{Preamble time} + \left( \frac{\text{CTS Packet payload}}{\text{Data Rate}} \right) \]

\[ = 192 + 14 \times \left( \frac{8}{1} \right) = 304\ \mu s \]

\[ \text{Average time per packet} = 50 + 352 + 304 + 310 + 1296 + 10 + 304 = 2626\ \mu s \]
**8.2.2 Multiple APs (near each other) and one STA per AP**

Since the AP queues are full, on the WiFi medium the packet transmission can still be viewed as being back-to-back as shown in the upper part of the figure below. However, since there are multiple contending AP-STA links, there are two differences between this figure and the one shown above (for the single AP and single STA case).

With reference to the figure above, note that, all the APs are contending to send their packets to their respective STAs, and the “Backoffs and Collisions” time is due to all the APs. However, finally, only one packet transmission succeeds. We will attribute all the contention overheads to the successful transmission of this packet. Thus, we will call the time duration from the beginning of a DIFS until the end of the ACK for the transmitted packet as the “effective” time taken to transmit that packet on the wireless medium. The average of these effective packet transmission times can be called the “Average time per Packet.”

\[ UDP \text{ throughput} = \frac{1460 \times 8}{2626} = 4.44 \text{ Mbps} \]
With this discussion, and the upper part of the figure above, it follows that the following expression still holds

\[
Total\ UDP\ Throughput\ (Mbps) = \frac{Application\ Payload\ in\ Packet\ (bits)}{Average\ Time\ per\ Packet(\mu s)}
\]

We observe from the figure that the average time per packet will be larger than when there is a single AP-STA pair. Hence, the total UDP throughput will be smaller when there are multiple AP-STA pairs (since the “Application Payload in the Packet” is the same in both cases.

Having obtained the total throughput over all the AP-STA pairs in this manner, by the fact that each packet transmission is with equal probability from any of the AP-STA pairs, the UDP throughput for each AP-STA pair (for \(N\) pairs) is just \(\frac{1}{N}\) of the total throughput.

8.3 Network Setup:

Open NetSim and click Examples > Experiments > WiFi-UDP-Download-Throughput > Part-1 > Without-RTS-CTS as shown below:
8.4 Procedure:

8.4.1 Part-1: Without RTS/CTS

The following set of procedures were done to generate this sample:

**Step 1:** A network scenario is designed in NetSim GUI comprising of 1 Wired Node, 1 Wireless Node, 1 L2 Switch, and 1 Access Point in the “Internetworks” Network Library.

**Step 2:** In the Interface Wireless > Physical Layer Properties of Wireless Node 5, Protocol Standard is set to IEEE 802.11b. In the Interface Wireless > Data Link Layer Properties of Wireless Node 5, RTS Threshold is set to 3000. It will automatically set the same in the Access Point, since the above parameters are Global.

**Step 3:** In the Wired Link Properties, Bit Error Rate and Propagation Delay is set to 0 for both the links.

**Step 4:** In the Wireless Link Properties, Channel Characteristics is set to NO PATH LOSS.

**Step 5:** Right click on the Application Flow App1 CBR and select Properties or click on the Application icon present in the top ribbon/toolbar. A CBR Application is generated from Wired Node 4 i.e. Source to Wireless Node 5 i.e. Destination with Packet Size set to 1450 Bytes and Inter Arrival Time set to 116µs. Transport Protocol is set to UDP.

The Packet Size and Inter Arrival Time parameters are set such that the Generation Rate equals 100 Mbps. Generation Rate can be calculated using the formula:

\[
\text{Generation Rate (Mbps)} = \frac{\text{Packet Size (Bytes)}}{\text{Interarrival time (µs)}} \times 8
\]

**Step 6:** Run the Simulation for 10 Seconds and note down the throughput.
8.4.2 Part-1: With RTS/CTS

The following changes in settings are done from the previous sample:

**Step 1:** In the Interface Wireless > Data Link Layer Properties of Wireless Node 5, RTS Threshold is set to 1000.

**Step 2:** Run the Simulation for 10 Seconds and note down the throughput.

8.4.3 Part-2: Without RTS/CTS: 2APs

The following changes in settings are done from the previous sample:

**Step 1:** A network scenario is designed in NetSim GUI comprising of 1 Wired Node, 2 Wireless Node, 1 L2 Switch, and 2 Access Points in the “Internetworks” Network Library.

**Step 2:** In the Interface Wireless > Data Link Layer Properties of Wireless Node 4, RTS Threshold is set to 3000. It will automatically be set for Wireless Node 6, since the above parameter is Global.

**Step 3:** Two CBR applications are generated from Wired Node 1 i.e. Source to Wireless Node 4 and Wireless Node 6 i.e. Destination with a Generation Rate of 10 Mbps.

**Step 4:** Run the Simulation for 10 Seconds and note down the throughput.

Similarly, the subsequent samples are carried out with 3, 4, and 5 Access Points and Wireless Nodes.

8.4.4 Part-2: With RTS/CTS: 2APs

The following changes in settings are done from the previous sample:

**Step 1:** In the Interface Wireless > Data Link Layer Properties of Wireless Node 4, RTS Threshold is set to 1000. It will automatically be set for Wireless Node 6, since the above parameter is Global.
Step 2: Run the Simulation for 10 Seconds and note down the throughput.

Similarly, the subsequent samples are carried out with 3, 4, and 5 Access Points and Wireless Nodes.

8.5 Output I:

After running simulation, check throughput in Application metrics as shown in the below screenshot:

<table>
<thead>
<tr>
<th>Sample</th>
<th>Predicted Throughput (Mbps)</th>
<th>Simulated Throughput (Mbps)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 (Without RTS/CTS)</td>
<td>5.92</td>
<td>5.92</td>
</tr>
<tr>
<td>2 (With RTS/CTS)</td>
<td>4.44</td>
<td>4.39</td>
</tr>
</tbody>
</table>

Table 8.1 UDP throughput for a single AP-STA, with and without RTS/CTS.

8.6 Output II:

After running simulation, check throughput in Application metrics as shown in the below screenshot:
Table 8.2 UDP throughput for 2, 3, 4, and 5 AP-STA pairs, with and without RTS/CTS.

<table>
<thead>
<tr>
<th>Sample</th>
<th>Throughput (Mbps) with 2 APs</th>
<th>Throughput (Mbps) with 3 APs</th>
<th>Throughput (Mbps) with 4 APs</th>
<th>Throughput (Mbps) with 5 APs</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>App 2: 3.02</td>
<td>App 2: 1.97</td>
<td>App 2: 1.57</td>
<td>App 2: 1.20</td>
</tr>
<tr>
<td></td>
<td><strong>Total: 6.16</strong></td>
<td><strong>Total: 6.18</strong></td>
<td><strong>Total: 6.13</strong></td>
<td><strong>Total: 6.07</strong></td>
</tr>
<tr>
<td>2 (With RTS/CTS)</td>
<td>App 1: 2.34</td>
<td>App 1: 1.58</td>
<td>App 1: 1.21</td>
<td>App 1: 0.95</td>
</tr>
<tr>
<td></td>
<td>App 2: 2.25</td>
<td>App 2: 1.49</td>
<td>App 2: 1.18</td>
<td>App 2: 0.90</td>
</tr>
<tr>
<td></td>
<td><strong>Total: 4.59</strong></td>
<td><strong>Total: 4.65</strong></td>
<td><strong>Total: 4.66</strong></td>
<td><strong>Total: 4.66</strong></td>
</tr>
</tbody>
</table>

8.7 Discussion

Table 8.1 shows the AP-STA UDP throughput (predicted and simulated) for a single AP-STA. Table 8.2 shows the UDP throughputs for 2, 3, 4, and 5 AP-STA pairs; the total throughput is shown along with the individual AP-STA throughputs. We can make the following observations, along with explanations (as bulleted comments) for the observations.

1. The UDP throughput with RTS/CTS turned off is larger than when RTS/CTS is used.
   - The reduction in throughput with RTS/CTS is due to the RTS/CTS overheads. The RTS/CTS mechanism aims at alerting “hidden” nodes that a transmission is about to start and can reduce collisions if there are hidden nodes. Since in
this experiment all node can directly hear each other’s transmissions, the Basic Access mode suffices, whereas RTS/CTS only adds overhead.

2. The UDP throughput increases slightly with two AP-STA pairs than with just one.
   • With just one AP-STA pair, there is wastage of time due to backoffs, even when there is no possibility of contention. When one more AP-STA is added some of this wastage is compensated by two APs attempting, with the possibility that one of them might finish its backoff early and grab the channel, thus reducing the backoff overhead. There is, of course, the additional time wasted due to collisions, but the balance between these two opposing phenomena is such that there is a small gain in throughput.

3. Further increase in the number AP-STA pairs leads to a decrease in throughput, but the decrease is small.
   • The IEEE 802.11 Distributed Coordination Function (DCF) manages the sharing of the WiFi channel in a distributed manner. If there was a centralised scheduler than each AP could be scheduled by turn, without any backoff and collision overheads, and the total throughput would have been just that due to sending UDP packets back-to-back: $\frac{1450 \times 8}{1296} = 8.95$ Mbps. Thus, the total throughput with DCF is smaller than if the UDP packets were being sent back-to-back, about 6 Mbps rather than 8.95 Mbps. However, DCF implements an adaptive attempt rate mechanism, which causes nodes to attempt less aggressively as the number of contending nodes increases. It is this mechanism that prevents the total throughput from dropping steeply as the number of AP-STA pairs increases.

4. The total throughput is distributed roughly equally between the AP-STA pairs.
   • This is another feature of DCF. The contending nodes obtain fair access at the packet level, i.e., each successful packet is from any of the contending nodes with equal probability. The down-side of this feature is that if an AP-STA is using long packets, then that UDP flow will get a larger throughput. In this experiment, all the AP-STA UDP flows are using the same packet lengths.
9. **How many downloads can a Wi-Fi access point simultaneously handle?**

9.1 **Motivation**

Wi-Fi has become the system of choice for access to Internet services inside buildings, offices, malls, airports, etc. In order to obtain access to the Internet over Wi-Fi a user connects his/her mobile device (a laptop or a cellphone, for example) to a nearby Wi-Fi access point (AP). A popular use of such a connection is to download a document, or a music file; in such an application, the user’s desire is to download the file as quickly as possible, i.e., to get a high throughput during the download. It is a common experience that as the number of users connected to an AP increases, the throughput obtained by all the users decreases, thereby increasing the time taken to download their files. The following question can be asked in this context.

If during the download, a user expects to get a throughput of at least $\theta$ bytes per second, what is the maximum number of users (say, $n_\theta$) up to which the throughput obtained by every user is at least $\theta$. We can say that $n_\theta$ is the capacity of this simple Wi-Fi network for the Quality of Service (QoS) objective $\theta$.  

9.2 **Objective**

In this experiment we will learn how to obtain $n_\theta$ in a simple WiFi network where the packet loss due to channel errors is 0. In this process we will understand some interesting facts about how WiFi networks perform when doing file transfers.

9.3 **Theory**

In NetSim, we will set up a network comprising a server that carries a large number of large files that the users would like to download into their mobile devices. The server is connected to a Wi-Fi AP, with the IEEE 802.11b version of the protocol, via an Ethernet switch. Several mobile devices (say, $N$) are associated with the AP, each downloading one of the files in the server. The Ethernet speed is 100Mbps, whereas the mobile devices are connected to the AP at 11Mbps, which is one of the IEEE 802.11b speeds.

We observe, from the above description, that the file transfer throughputs will be limited by the wireless links between the AP and the mobile devices (since the Ethernet speed is much larger than

---

1 It may be noted that the term capacity has several connotations in communications. Our use of the word here must not be confused with the notion of information theoretic capacity of a communication channel.
the Wi-Fi channel speed). There are two interacting mechanisms that will govern the throughputs that the individual users will get:

1. The Wi-Fi medium access control (MAC) determines how the mobile devices obtain access to the wireless medium. There is one instance of the WiFi MAC at each of the mobile devices.
2. The end-to-end protocol, TCP, controls the sharing of the wireless bandwidth between the ongoing file transfers. In our experiment, there will be one instance of TCP between the server and each of the mobile devices.

For simplicity, the default implementation of TCP in NetSim does not implement the delayed ACK mechanism. This implies that a TCP receiver returns an ACK for every received packet. In the system that we are simulating, the server is the transmitter for all the TCP connections, and each user’s mobile device is the corresponding receiver.

Suppose, each of the $N$ TCP connection transmits one packet to its corresponding mobile device; then each mobile device will have to return an ACK. For this to happen, the AP must send $N$ packets, and each of the $N$ mobile devices must send back $N$ ACKs. Thus, for the file transfers to progress, the AP needs to $N$ packets for each packet (i.e., ACK) returned by each mobile device. We conclude that, in steady state, the AP must send as many packets as all the mobile devices send, thus requiring equal channel access to the AP as to all the mobile devices together.

At this point, it is important to recall that when several nodes (say, an AP and associated mobile devices) contend for the channel, the WiFi medium access control provides fair access at the packet level, i.e., each contending device has an equal chance of succeeding in transmitting a packet over the channel. Now consider the system that we have set up in this present experiment. There are $N$ mobile devices associated with one AP. Suppose, for example, 10 of them ($N \geq 10$) all have a packet to transmit (and none other has a packet). By the fair access property of the WiFi MAC, each of these 10 nodes, along with the AP, has an equal probability of successfully transmitting. It follows, by the packet level fair access property, that each node will have a probability of $\frac{1}{11}$ of succeeding in transmitting its packet. If this situation continues, the channel access ratio to the AP will be inadequate and the equal channel access argued in the previous paragraph will be violated. It follows from this that, on the average, roughly only one mobile device will have an ACK packet in it; the AP will contend with one other node, thus getting half the packet transmission opportunities.

With the just two nodes contending, the collision probability is small ($\sim 0.06$) and the probability of packet discard is negligibly small. Thus, the TCP window for every transfer will grow to the maximum window size. The entire window worth of TCP data packets for the $N$ sessions will be in the AP buffer, except for a very small number of packets (averaging to about 1) which will appear as ACKs in the mobile devices.
It follows that, in steady state, the system will look like two contending WiFi nodes, one with TCP data packets and the other with TCP ACK packets. This will be the case no matter how many downloading mobile devices there are. The total throughput can be obtained by setting up the model of two saturated nodes, one with TCP data packets, and the other with TCP ACK packets. The data packets of all the TCP connections will be randomly ordered in the AP buffer, so that the head-of-the-line packet will belong to any particular mobile device with probability $\frac{1}{N}$. This throughput is shared equally between the $N$ mobile devices.

Now suppose that the TCP data packet throughput with the two-node model is $\theta$. Then

$$n_\theta = \left\lfloor \frac{\Theta}{\theta} \right\rfloor$$

where the $\lfloor x \rfloor$ denotes the largest integer less than or equal to $x$. Use NetSim to verify that for an 11Mbps Wi-Fi speed, with RTS/CTS enabled the total TCP throughput is 3.4 Mbps. If $\theta = 0.65 \text{ Mbps}$, then $n_\theta = \left\lfloor \frac{3.8}{0.65} \right\rfloor = 5$. In this example, if $N = 5$ the download throughput obtained by each of them will be $0.68 \text{ Mbps}$, but if one more downloading device is added then each will get a throughput less than $\theta = 0.65 \text{ Mbps}$. We say that the capacity of this network for a target throughput of 0.65 Mbps is 5.

9.4 Procedure

Open NetSim and click Examples > Experiments > How-many-downloads-can-a-Wi-Fi-access-point-simultaneously-handle? > Sample-1 as shown below:

NetSim UI displays the configuration file corresponding to this experiment.
9.5 Procedure:

The following set of procedures were done to generate this sample.

**Step 1:** A network scenario is designed in the NetSim GUI comprising of 1 Wired Node, 1 Wireless Node, 1 Access Point, and 1 Router in the “Internetworks” Network Library.

**Step 2:** In the Interface (WIRELESS) > Data Link Layer Properties of Wireless Node 4, Retry Limit is set to 4 and RTS Threshold is set to 1000 bytes.

**Step 3:** Right-click the link ID (of a wired link) and select Properties to access the link’s properties. The Link Properties are set according to the values given in the below table.

<table>
<thead>
<tr>
<th>Wired Link</th>
<th>Protocol</th>
<th>Interface</th>
<th>Enable</th>
<th>Retry Limit</th>
<th>RTS Threshold(Bytes)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>IEEE 802.11</td>
<td></td>
<td></td>
<td>4</td>
<td>1000</td>
</tr>
</tbody>
</table>
Max Uplink Speed (Mbps) | 100
Max Downlink Speed (Mbps) | 100
Uplink BER | 0
Downlink BER | 0
Uplink Propagation Delay (µs) | 0
Downlink Propagation Delay (µs) | 0

<table>
<thead>
<tr>
<th>Wireless Link</th>
</tr>
</thead>
<tbody>
<tr>
<td>Channel Characteristics</td>
</tr>
</tbody>
</table>

**Step 4:** Right click on **App1 FTP** and select Properties or click on the Application icon present in the top ribbon/toolbar.

An FTP Application is generated from Wired Node 2 i.e. Source to Wireless Node 4 i.e. Destination with File Size set to 100000000 Bytes and Inter Arrival Time set to 15 s.

**Step 5:** Run the Simulation for 15 Seconds and note down the throughput.

**Sample 2:**

The following changes in settings are done from the previous sample:

**Step 1:** The number of Wireless Nodes is increased to 5 and FTP applications are generated from Wired Node 2 to each of the Wireless Nodes as shown below:

**No. of wireless nodes = 5**
### Application Properties

<table>
<thead>
<tr>
<th>Properties</th>
<th>App1</th>
<th>App2</th>
<th>App3</th>
<th>App4</th>
<th>App5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Application Type</td>
<td>FTP</td>
<td>FTP</td>
<td>FTP</td>
<td>FTP</td>
<td>FTP</td>
</tr>
<tr>
<td>Source Id</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>Destination Id</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
</tr>
<tr>
<td>File size (Bytes)</td>
<td>100000000</td>
<td>100000000</td>
<td>100000000</td>
<td>100000000</td>
<td>100000000</td>
</tr>
<tr>
<td>File Inter arrival time</td>
<td>15 s</td>
<td>15 s</td>
<td>15 s</td>
<td>15 s</td>
<td>15 s</td>
</tr>
</tbody>
</table>

**Step 2:** Run the Simulation for 15 Seconds and note down the throughput.

**NOTE:** Follow the same procedure for next samples with wireless nodes 10, 15, 20, 25 and note down the sum of throughputs for all applications.

### 9.6 Measurements and Output:

Aggregated download throughput with different values of N (wireless nodes) is shown below:

\[
\text{Throughput Per Device (Mbps)} = \frac{\text{Sum of throughputs (Mbps)}}{\text{Number of Devices}}
\]

<table>
<thead>
<tr>
<th>Sample Number</th>
<th>Number of Devices</th>
<th>Sum of throughputs (Mbps)</th>
<th>Throughput Per Device (Mbps)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>3.42</td>
<td>3.42</td>
</tr>
<tr>
<td>2</td>
<td>5</td>
<td>3.44</td>
<td>0.69</td>
</tr>
<tr>
<td>3</td>
<td>10</td>
<td>3.25</td>
<td>0.325</td>
</tr>
<tr>
<td>4</td>
<td>15</td>
<td>3.27</td>
<td>0.218</td>
</tr>
<tr>
<td>Layer</td>
<td>Overhead (Bytes)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-----------------</td>
<td>------------------</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Transport Layer</td>
<td>20</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Network Layer</td>
<td>20</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MAC Layer</td>
<td>40</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PHY layer</td>
<td>48µs = (11*48)/8 = 66</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Total Overhead</td>
<td>146</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

\[
\text{PHY\_Throughput} = \text{APP\_Throughput} \times \frac{1606}{1460} = 3.42\times1606/1460 = 3.76 \text{ Mbps}
\]

**9.7 Inference:**

We see that as the number of devices increase the aggregate (combined) throughput remains constant whereas the throughput per user decreases.

As discussed earlier, our goal was to identify that if during the download, a user expects to get a throughput of at least \( \theta \) bytes per second, what is the maximum number of users (say, \( n_{\theta} \))?  

If we set \( \theta \) to be 650 Kbps, then we see that from the output table that the maximum number of users who can simultaneously download files is 5 \( (n_{\theta}) \)

**9.8 Reference Documents:**

1. Analytical models for capacity estimation of IEEE 802.11 WLANs using DCF for internet applications. George Kuriakose, Sri Harsha, Anurag Kumar, Vinod Sharma
10. TCP Congestion Control Algorithms

10.1 Introduction:

A key component of TCP is end-to-end congestion control algorithm. The TCP congestion control algorithm limits the rate at which the sender sends traffic into the network based on the perceived network congestion. The TCP congestion control algorithm at the sender maintains a variable called congestion window, commonly referred as $cwnd$, that limits the amount of unacknowledged data in the network. The congestion window is adapted based on the network conditions and this affects the sender’s transmission rate. The TCP sender reacts to congestion and other network conditions based on new acknowledgements, duplicate acknowledgements and timeouts. The TCP congestion control algorithms describe the precise manner in which TCP adapts $cwnd$ with the different events.

The TCP congestion control algorithm has three major phases (a) slow-start, (b) congestion avoidance, and (c) fast recovery. In slow-start, TCP is aggressive and increases $cwnd$ by one MSS with every new acknowledgement. In congestion avoidance, TCP is cautious and increases the $cwnd$ by one MSS per round-trip time. Slow-start and congestion avoidance are mandatory components of all TCP congestion control algorithms. In the event of a packet loss (inferred by timeout or triple duplicate acknowledgements), the TCP congestion control algorithm reduces the congestion window to 1 (e.g., Old Tahoe, Tahoe) or by half (e.g., New Reno). In fast recovery, TCP seeks to recover from intermittent packet losses while maintaining a high congestion window. The new versions of TCP, including TCP New Reno, incorporate fast recovery as well. Fig 16 presents a simplified view of the TCP New Reno congestion control algorithm highlighting slow-start, congestion avoidance and fast recovery phases.

TCP congestion control algorithm is often referred to as additive-increase multiplicative-decrease (AIMD) form of congestion control. The AIMD congestion control algorithm often leads to a “saw tooth” evolution of the congestion window (with linear increase of the congestion window during bandwidth probing and a multiplicative decrease in the event of packet losses), see Fig 20.
10.2 Network Setup:

We will seek a large file transfer with TCP over a lossy link to study the TCP congestion control algorithms. We will simulate the network setup illustrated in Fig 17 with the configuration parameters listed in detail in steps to study the working of TCP congestion control algorithms.

Open NetSim and click Examples > Experiments > TCP-congestion-control-algorithms > Sample-1 as shown below:
NetSim UI displays the configuration file corresponding to this experiment as shown below:

![NetSim UI Configuration](image.png)

**Fig 17** client and a server network architecture.

### 10.3 Procedure:

#### Sample 1

The following set of procedures were done to generate this sample.

**Step 1:** A network scenario is designed in NetSim GUI comprising of 2 Wired Nodes and 2 Routers in the “Internetworks” Network Library.

**Step 2:** In the Source Node, i.e. Wired Node 1, in the TRANSPORT LAYER Properties, Congestion Control Algorithm is set to OLD TAHOE.

**Step 3:** In the General Properties of Wired Node 1 i.e. Source, Wireshark Capture is set to Online.

*Note: Accept default properties for Routers as well as the Links.*

**Step 4:** Right-click the link ID (of a wired link) and select Properties to access the link's properties. Set Max Uplink Speed and Max Downlink Speed to 10 Mbps. Set Uplink BER and Downlink BER to 0. Set Uplink Propagation Delay and Downlink Propagation Delay as 100 microseconds for the links 1 and 3 (between the Wired Node's and the routers). Set Uplink Propagation Delay and Downlink Propagation Delay as 50000 microseconds and Uplink BER and Downlink BER to 0.000001 for the backbone link connecting the routers, i.e., 2.

**Step 5:** Right click on the Application Flow App1 CBR and select Properties or click on the Application icon present in the top ribbon/toolbar.

An CBR Application is generated from Wired Node 1 i.e. Source to Wired Node 2 i.e. Destination with Packet Size set to 1460 Bytes and File Inter Arrival Time set to 1168 microseconds.

**Step 6:** Click on Display Settings > Device IP check box in the NetSim GUI to view the network topology along with the IP address.
**Step 7:** Click on **Plots** icon and select the **Enable Plots** checkbox. This enables us to view the throughput plot of the application **App1 CBR**.

**Step 8:** Click on Run simulation. The **simulation time** is set to 20 seconds. In the “**Static ARP Configuration**” tab, Static ARP is set to **disable**.

**Sample 2**

**Step 1:** In the Source Node, i.e. Wired Node 1, in the TRANSPORT LAYER Properties, Congestion Control Algorithm is set to **TAHOE**.

**Step 2:** Click on Run simulation. The **simulation time** is set to 20 seconds. In the “**Static ARP Configuration**” tab, Static ARP is set to **disable**.

**Sample 3**

**Step 1:** In the Source Node, i.e. Wired Node 1, in the TRANSPORT LAYER Properties, Congestion Control Algorithm is set to **NEW RENO**.

**Step 2:** Click on Run simulation. The **simulation time** is set to 20 seconds. In the “**Static ARP Configuration**” tab, Static ARP is set to **disable**.

**10.4 Output:**

We have enabled WireShark Capture in the Wired Node 1. The PCAP file is generated at the end of the simulation. From the PCAP file, the congestion window evolution graph can be obtained as follows. In Wireshark, select any data packet with a left click, then, go to **Statistics > TCP Stream Graphs > Window Scaling**.

The congestion window evolution for Old Tahoe, Tahoe and New Reno congestion control algorithms are presented in Fig 18, Fig 19, and Fig 20, respectively.

Table 3 shows the throughput values of different congestion control algorithms (obtained from the Application Metrics).
**Fig 18** Congestion window evolution with TCP Old Tahoe. We note that Old Tahoe infers packet loss only with timeouts, and updates the slow-start threshold ssthresh and congestion window cwnd as ssthresh = cwnd/2 and cwnd = 1.

**Fig 19** Congestion window evolution with TCP Tahoe. We note that Tahoe infers packet loss with timeout and triple duplicate acknowledgements, and updates the slow-start threshold ssthresh and congestion window cwnd as ssthresh = cwnd/2 and cwnd = 1.
Fig 20 Congestion window evolution with TCP New Reno. We note that New Reno infers packet loss with timeout and triple duplicate acknowledgements, and updates the slow-start threshold ssthresh and congestion window cwnd as ssthresh = cwnd/2 and cwnd = ssthresh + 3MSS (in the event of triple duplicate acknowledgements).

<table>
<thead>
<tr>
<th>Congestion Control Algorithm</th>
<th>Throughput</th>
</tr>
</thead>
<tbody>
<tr>
<td>Old Tahoe</td>
<td>3.52 Mbps</td>
</tr>
<tr>
<td>Tahoe</td>
<td>3.03 Mbps</td>
</tr>
<tr>
<td>New Reno</td>
<td>4.13 Mbps</td>
</tr>
</tbody>
</table>

Table 3 Long-term average throughput of the different TCP congestion control algorithms

10.5 Observations and Inference

1. We can observe slow start, congestion avoidance, timeout, fast retransmit and recovery phases in the Fig 18, Fig 19, and Fig 20. In Fig 18, we note that Old Tahoe employs timeout, slow-start and congestion avoidance for congestion control. In Fig 19, we note that Tahoe employs fast retransmit, slow-start and congestion avoidance for congestion control. In Fig 20, we note that New Reno employs fast retransmit and recovery, congestion avoidance and slow-start for congestion control.

2. We note that TCP New Reno reports a higher long term average throughput (in comparison with Old Tahoe and Tahoe, see Table 3) as it employs fast retransmit and recovery to recover from packet losses.
11. Multi-AP Wi-Fi Networks: Channel Allocation

11.1 Introduction
A single Wi-Fi Access Point (AP) can connect laptops and other devices that are a few 10s of meters distance from the AP, the actual coverage depending on the propagation characteristics of the building in which the Wi-Fi network is deployed. Thus, for large office buildings, apartment complexes, etc., a single AP does not suffice, and multiple APs need to be installed, each covering a part of the building. We will focus on 2.4GHz and 5GHz systems. In each of these systems the available bandwidth is organized into channels, with each AP being assigned to one of the channels. For example, 2.4GHz Wi-Fi systems operate in the band 2401MHz to 2495MHz, which has 14 overlapping channels each of 22MHz. There are 3 nonoverlapping channels, namely, Channels 1, 6, and 11, which are centered at 2412MHz, 2437MHz, and 2462MHz. Evidently, if neighboring APs are assigned to the same channel or overlapping channels they will interfere, thereby leading to poor performance. On the other hand, since there are only three nonoverlapping channels, some care must be taken in assigning channels to APs so that nearby APs have nonoverlapping channels, whereas APs that are far apart can use the same or overlapping channels.

In this experiment we will understand some basic issues that arise in multi-AP networks, particularly with attention to channel allocation to the APs.

11.2 Network Setup:
Open NetSim and click **Examples > Experiments > Multi-AP-Wi-Fi-Networks-Channel-Allocation** as shown below:
NetSim UI displays the configuration file corresponding to this experiment as shown below:

**APs on the same channel:**

![Diagram of network setup]

**Sample1:**

The following set of procedures were done to generate this sample:

**Step 1:** A network scenario is designed in NetSim GUI comprising of 1 Wired Node, 1 L2 Switch, 3 Wireless Nodes and 3 Access Points in the “Internetworks” Network Library.

**Step 2:** The device positions are set as per the table given below:

<table>
<thead>
<tr>
<th>General Properties</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Device Name</strong></td>
</tr>
<tr>
<td>AP_1</td>
</tr>
<tr>
<td>AP_2</td>
</tr>
<tr>
<td>AP_3</td>
</tr>
<tr>
<td>Wireless_Node_6</td>
</tr>
<tr>
<td>Wireless_Node_7</td>
</tr>
<tr>
<td>Wireless_Node_8</td>
</tr>
</tbody>
</table>

**Step 3:** In the INTERFACE (WIRELESS) > PHYSICAL LAYER Properties of all the Wireless Nodes and Access Points, the Protocol Standard is set to IEEE 802.11 b.

**Step 4:** Right-click the link ID (of a wired link) and select Properties to access the link’s properties. For all the Wired Links, Bit Error Rate and Propagation Delay is set to 0.

**Step 5:** The Wireless Link Properties are set according to the values given in the below table.

<table>
<thead>
<tr>
<th>Channel Characteristics</th>
<th>PATH LOSS ONLY</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Path Loss Model</td>
<td>LOG DISTANCE</td>
</tr>
<tr>
<td>----------------------</td>
<td>--------------</td>
</tr>
<tr>
<td>Path Loss Exponent</td>
<td>3.5</td>
</tr>
</tbody>
</table>

**Step 6:** Right click on the Application Flow App1 CBR and select Properties or click on the Application icon present in the top ribbon/toolbar.

A CBR Application is generated from Wired Node 1 i.e. Source to Wireless Node 6 i.e. Destination with Packet Size set to 1460 Bytes and Inter Arrival Time set to 1168µs.

Transport Protocol is set to **UDP** instead of TCP.

The Packet Size and Inter Arrival Time parameters are set such that the Generation Rate equals 10 Mbps. Generation Rate can be calculated using the formula:

\[
\text{Generation Rate (Mbps)} = \frac{\text{Packet Size (Bytes)}}{\text{Interarrival time (µs)}} \times 8
\]

Similarly, two more CBR applications are generated.

**Step 8:** Run the Simulation for 10 Seconds and note down the throughput.

**Sample 2:**

The following changes in settings are done from the previous sample:

**Step 1:** Before we start designing the network scenario, the Grid Length is set to 1000 meters. This can be set by choosing the Menu Option **Settings > Grid/Map > Grid** from the GUI.

**Step 2:** From the previous sample, we have removed App2 CBR (i.e. from Wired Node1 to Wireless Node7), set distance between the other 2 Access Points (AP 1 and AP 3) as 400m and distance between APs and Wireless nodes as 10m as shown below:
Step 3: The device positions are set according to the table given below:

<table>
<thead>
<tr>
<th>General Properties</th>
<th>Device Name</th>
<th>X / Lon</th>
<th>Y / Lat</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>AP_1</td>
<td>400</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>AP_2</td>
<td>400</td>
<td>200</td>
</tr>
<tr>
<td></td>
<td>AP_3</td>
<td>400</td>
<td>400</td>
</tr>
<tr>
<td></td>
<td>Wireless_Node_6</td>
<td>410</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Wireless_Node_7</td>
<td>410</td>
<td>200</td>
</tr>
<tr>
<td></td>
<td>Wireless_Node_8</td>
<td>410</td>
<td>400</td>
</tr>
</tbody>
</table>

Step 4: Run the Simulation for 10 Seconds and note down the throughput.

Sample3:

The following changes in settings are done from the previous sample:

Step 1: The distance between the Access Points (AP 1 and AP 3) is set to 400m and distance between APs and Wireless nodes as 10m as shown below:
Step 2: The device positions are set according to the table given below:

<table>
<thead>
<tr>
<th>General Properties</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Device Name</strong></td>
</tr>
<tr>
<td>AP_1</td>
</tr>
<tr>
<td>AP_2</td>
</tr>
<tr>
<td>AP_3</td>
</tr>
<tr>
<td>Wireless_Node_6</td>
</tr>
<tr>
<td>Wireless_Node_7</td>
</tr>
<tr>
<td>Wireless_Node_8</td>
</tr>
</tbody>
</table>

Step 3: Run the Simulation for 10 Seconds and note down the throughput.

Sample4:

The following changes in settings are done from the previous sample:

Step 1: From the previous sample, we have removed App1 CBR (i.e. from Wired Node 1 to Wireless Node 6), set distance between the other 2 Access Points (AP 2 and AP 3) as 200m and distance between APs and Wireless nodes as 10m as shown below:
Step 2: The device positions are set according to the table given below:

<table>
<thead>
<tr>
<th>General Properties</th>
</tr>
</thead>
<tbody>
<tr>
<td>Device Name</td>
</tr>
<tr>
<td>AP_1</td>
</tr>
<tr>
<td>AP_2</td>
</tr>
<tr>
<td>AP_3</td>
</tr>
<tr>
<td>Wireless_Node_6</td>
</tr>
<tr>
<td>Wireless_Node_7</td>
</tr>
<tr>
<td>Wireless_Node_8</td>
</tr>
</tbody>
</table>

Step 3: Run the Simulation for 10 Seconds and note down the throughput.

Sample5:

The following changes in settings are done from the previous sample:

Step 1: From Sample 3, we have removed first and third applications as shown below:
Step 2: The device positions are set according to the table given below:

<table>
<thead>
<tr>
<th>Device Name</th>
<th>X / Lon</th>
<th>Y / Lat</th>
</tr>
</thead>
<tbody>
<tr>
<td>AP_1</td>
<td>400</td>
<td>0</td>
</tr>
<tr>
<td>AP_2</td>
<td>400</td>
<td>200</td>
</tr>
<tr>
<td>AP_3</td>
<td>400</td>
<td>400</td>
</tr>
<tr>
<td>Wireless_Node_6</td>
<td>410</td>
<td>0</td>
</tr>
<tr>
<td>Wireless_Node_7</td>
<td>410</td>
<td>200</td>
</tr>
<tr>
<td>Wireless_Node_8</td>
<td>410</td>
<td>400</td>
</tr>
</tbody>
</table>

Step 3: Run the Simulation for 10 Seconds and note down the throughput.

APs in different channel:

The following changes in settings are done from the previous sample:

Step 1: From Sample 3, we have changed standard channel to **11_2462** under INTERFACE (WIRELESS) > DATALINK LAYER Properties of AP 2.

Step 2: Run the Simulation for 10 Seconds and note down the throughput.

**11.3 Output:**

After running simulation, check throughput in Application metrics as shown in the below screenshot:
### Sample Throughput (Mbps)

<table>
<thead>
<tr>
<th>AP</th>
<th>Sample Throughput (Mbps)</th>
</tr>
</thead>
<tbody>
<tr>
<td>AP_1</td>
<td>AP_2</td>
</tr>
<tr>
<td>1</td>
<td>2.03</td>
</tr>
<tr>
<td>2</td>
<td>5.94</td>
</tr>
<tr>
<td>3</td>
<td>5.42</td>
</tr>
<tr>
<td>4</td>
<td>N/A</td>
</tr>
<tr>
<td>5</td>
<td>N/A</td>
</tr>
</tbody>
</table>

**All APs on the same channel**

**Each AP on a different nonoverlapping channel**

---

**NOTE:** Please refer “Wi-Fi UDP Download Throughput” experiment for theoretical WLAN throughput calculations in NetSim Experiment Manual.

### 11.4 Discussion

We recall that each AP is associated with one station (STA; e.g., a laptop). All the APs are connected to the same server which is sending separate UDP packet streams to each of the STAs via the corresponding AP. The packet transmission rate from the server is large enough so that the AP queue is permanently backlogged, i.e., the rate at which the server transmits packets is larger than the rate at which the AP can empty the packet queue.
11.4.1 All APs on the same channel

- **Case 1:** All the APs and their associated STAs are close together, so that all devices (APs and STAs) can sense every other device.
  - The table shows that all the AP-STA links achieve the same UDP throughput. This is because all the AP-STA links are equivalent (since all interfere with each other), and only one can be active at one time. The throughput for this scenario can be predicted from the analysis in Section 7.4 of the book *Wireless Networking* by Anurag Kumar, D. Manjunath and Joy Kuri

- **Case 2:** AP1 and AP3 are close to their associated STAs but are 400m apart. The link from AP2 to its STA is half-way between the other two APs, and is not carrying any traffic.
  - The table shows that both the links from AP1 and AP3 to their respective STAs carry the same throughput, of 5.94Mbps and 5.92Mbps. These are also the throughputs that each link would have if the other was not present, indicating that the two links are far enough apart that they do not interfere.

- **Case 3:** This is the same scenario as Case 2, but the AP2-STA link is now carrying traffic.
  - We find that, in comparison with Case2, the AP1-STA and AP3-STA carry slightly lower throughputs of about 5.4Mbps, whereas the AP2-STA link carries a small throughput of 0.63Mbps. Comparing Cases 1 and 3 we conclude that in these networks there can be severe unfairness depending on the relative placement of the AP-STA links. In Case 1, all the links could sense each other, and each got a fair chance. In Case 3, we have what is called the “link-in-the-middle problem.” The AP2-STA link is close enough to interfere with the AP1-STA link and the AP3-STA link, whereas the AP1-STA link and the AP3-STA link do not “see” each other. The AP2-STA link competes with the links on either side, whereas the other links compete only with the link in the centre, which thereby gets suppressed in favour of the outer links.

- **Case 4:** Here we stop the traffic to AP1 but send the traffic to the AP2-STA link and the AP3-STA link.
  - The two active links interfere with each other, but the situation is symmetric between them (unlike in Case 3), and they obtain equal throughput. Again, the throughput obtained by these two links can be predicted by the analysis mentioned earlier in this section.

- **Case 5:** Now we send traffic only to AP2.
  - The throughput is now 5.92Mbps, since the AP2-STA link can transmit without interference; there are no collisions. The reason that this throughput is less than the sum of the two throughputs in Case 4 is that the single link acting by itself, with all the attendant overheads, is unable to occupy the channel fully.
11.4.2 Each AP on a different nonoverlapping channel

There is only one case here. Having observed the various situations that arose in the previous subsection when all the APs are on the same channel, now we consider the case where all the AP-STA pairs are each on a different nonoverlapping channel. As expected, every AP-STA pair gets the same throughput as when they are alone on the network.
12. Plot the characteristic curve of throughput versus offered traffic for a Pure and Slotted ALOHA system

**NOTE:** NetSim Academic supports a maximum of 100 nodes and hence this experiment can only be done partially with NetSim Academic. NetSim Standard/Pro would be required to simulate all the configurations.

### 12.1 Theory:

ALOHA provides a wireless data network. It is a multiple access protocol (this protocol is for allocating a multiple access channel). There are two main versions of ALOHA: pure and slotted. They differ with respect to whether or not time is divided up into discrete slots into which all frames must fit.

**Pure ALOHA:**

In pure Aloha, time is continuous. In Pure ALOHA, users transmit whenever they have data to be sent. There will be collisions and the colliding frames will be damaged. Senders need some way to find out if this is the case. If the frame was destroyed, the sender just waits a random amount of time and sends it again. The waiting time must be random or the same frames will collide over and over, in lockstep. Systems in which multiple users share a common channel in a way that can lead to conflicts are known as contention systems.

The probability of no other traffic being initiated during the entire vulnerable period is given by $e^{-2G}$ which leads to $S = G * e^{-2G}$ where, $S$ (frames per frame time) is the mean of the Poisson distribution with which frames are being generated. For reasonable throughput $S$ should lie between 0 and 0.5.

$G$ is the mean of the Poisson distribution followed by the transmission attempts per frame time, old and new combined. Old frames mean those frames that have previously suffered collisions.

The maximum throughput occurs at $G = 0.5$, with $S = 1/2e$, which is about 0.184. In other words, the best we can hope for is a channel utilization of 18%. This result is not very encouraging, but with everyone transmitting at will, we could hardly have expected a 100% success rate.

**Slotted ALOHA:**

In slotted Aloha, time is divided up into discrete intervals, each interval corresponding to one frame. In Slotted ALOHA, a computer is required to wait for the beginning of the next slot in order to send the next packet. The probability of no other traffic being initiated during the entire vulnerable period is given by $e^{-G}$ which leads to $S = G * e^{-G}$ where, $S$ (frames per frame time) is the mean of the
Poisson distribution with which frames are being generated. For reasonable throughput $S$ should lie between 0 and 1.

$G$ is the mean of the Poisson distribution followed by the transmission attempts per frame time, old and new combined. Old frames mean those frames that have previously suffered collisions.

It is easy to note that Slotted ALOHA peaks at $G = 1$, with a throughput of $s = \frac{1}{e}$ or about 0.368.

**Calculations used in NetSim to obtain the plot between $S$ and $G$:**

Using NetSim, the attempts per packet time ($G$) can be calculated as follows:

$$G = \frac{\text{Number of packet transmitted} \times \text{Slot length(s)}}{ST}$$

Where, $G = \text{Attempts per packet time}$

$ST = \text{Simulation time (in second)}$

The throughput (in Mbps) per packet time can be obtained as follows:

$$S = \frac{\text{Number of packet success} \times \text{Slot length(s)}}{ST}$$

Where, $S = \text{Throughput per packet time}$

$ST = \text{Simulation time (in second)}$

In the following experiment, we have taken packet size=$1460$ (Data Size) + $28$ (Overheads) = $1488$ bytes.

Bandwidth is $10$ Mbps and hence, packet time comes as $1.2$ milliseconds.

*(Reference: A good reference for this topic is Section 4.2.1: ALOHA, of the book, Computer Networking, 5th Edition by Tanenbaum and Wetherall)*

**12.2 Network Set Up:**

**Part-1**

Open NetSim and click **Examples > Experiments > Throughput-versus-load-for-Pure-and-Slotted-Aloha** as shown below:
NetSim UI displays the configuration file corresponding to this experiment as shown below:

Sample Inputs:

**Input for Sample 1:** Drop 10 nodes (i.e. 9 Nodes are generating traffic.)

Node 2, 3, 4, 5, 6, 7, 8, 9, and 10 generates traffic. The properties of Nodes 2, 3, 4, 5, 6, 7, 8, 9, and 10 which transmits data to Node 1 are given in the below table.

**Wireless Node Properties:**

<table>
<thead>
<tr>
<th>Wireless Node Properties</th>
<th>Interface1_Wireless (PHYSICAL_LAYER)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Rate (Mbps)</td>
<td>10</td>
</tr>
</tbody>
</table>
Interface1_Wireless (DATALINK_LAYER)

<table>
<thead>
<tr>
<th>Property</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Retry_Limit</td>
<td>0</td>
</tr>
<tr>
<td>MAC_Buffer</td>
<td>FALSE</td>
</tr>
<tr>
<td>Slot Length(µs)</td>
<td>1200</td>
</tr>
</tbody>
</table>

(Note: Slot Length(µs) parameter present only in Slotted Aloha → Wireless Node Properties → Interface_1 (Wireless).)

In Adhoc Link Properties, channel characteristic is set as **No Path Loss**.

**Application Properties:** Right click on the Application Flow “App1 CUSTOM” and select Properties or click on the Application icon present in the top ribbon/toolbar. The properties are set according to the values given in the below table.

<table>
<thead>
<tr>
<th>Application_1 Properties</th>
</tr>
</thead>
<tbody>
<tr>
<td>Application Method</td>
</tr>
<tr>
<td>Application Type</td>
</tr>
<tr>
<td>Source_Id</td>
</tr>
<tr>
<td>Destination_Id</td>
</tr>
<tr>
<td>Transport Protocol</td>
</tr>
<tr>
<td>Packet Size</td>
</tr>
<tr>
<td>Value (Bytes)</td>
</tr>
<tr>
<td>Inter Arrival Time</td>
</tr>
<tr>
<td>Packet Inter Arrival Time (µs)</td>
</tr>
</tbody>
</table>

Similarly create 8 more application, i.e. Source_Id as 3, 4, 5, 6, 7, 8, 9 and Destination_Id as 1, set Packet Size and Inter Arrival Time as shown in above table.

**Simulation Time- 10 Seconds**

*Note: Obtain the values of Total Number of Packets Transmitted and Collided from the results window of NetSim.*

**Input for Sample2:** Drop 20 nodes (i.e. 19 Nodes are generating traffic.)

Nodes 2, 3, 4, 5, 6, 7, 8, 9, 11, 12, 13, 14, 15, 16, 17, 18, 19, and 20 transmit data to Node 1.

Continue the experiment by increasing the number of nodes generating traffic as 29, 39, 49, 59, 69, 79, 89, 99, 109, 119, 129, 139, 149, 159, 169, 179, 189 and 199 nodes.

**Part-2 - Slotted ALOHA:**

**Input for Sample1:** Drop 20 nodes (i.e. 19 Nodes are generating traffic.)

Nodes 2, 3, 4, 5, 6, 7, 8, 9, 11, 12, 13, 14, 15, 16, 17, 18, 19, and 20 transmit data to Node 1 and set properties for nodes and application as mentioned above.
Continue the experiment by increasing the number of nodes generating traffic as 39, 59, 79, 99, 119, 139, 159, 179, 199, 219, 239, 259, 279, 299, 319, 339, 359, 379, and 399 nodes.

### 12.3 Output:

**Comparison Table:** The values of Total Number of Packets Transmitted and Collided obtained from the network statistics after running NetSim simulation are provided in the table below along with Throughput per packet time & Number of Packets Transmitted per packet time

#### Pure Aloha:

<table>
<thead>
<tr>
<th>Number of nodes generating traffic</th>
<th>Total number of Packets Transmitted</th>
<th>Total number of Packets Collided</th>
<th>Total number of Packets Success (Packets Transmitted - Packets Collided)</th>
<th>Throughput per packet time(G)</th>
<th>Number of Packets Transmitted per packet time(S)</th>
<th>Packets per packet time theoretical ($S = \frac{G}{e^{-2G}}$)</th>
</tr>
</thead>
<tbody>
<tr>
<td>9</td>
<td>494</td>
<td>60</td>
<td>434</td>
<td>0.05928</td>
<td>0.05208</td>
<td>0.05265</td>
</tr>
<tr>
<td>19</td>
<td>978</td>
<td>187</td>
<td>791</td>
<td>0.11736</td>
<td>0.09492</td>
<td>0.09281</td>
</tr>
<tr>
<td>29</td>
<td>1482</td>
<td>415</td>
<td>1067</td>
<td>0.17784</td>
<td>0.12804</td>
<td>0.12461</td>
</tr>
<tr>
<td>39</td>
<td>1991</td>
<td>700</td>
<td>1291</td>
<td>0.23892</td>
<td>0.15492</td>
<td>0.14816</td>
</tr>
<tr>
<td>49</td>
<td>2443</td>
<td>1056</td>
<td>1387</td>
<td>0.29316</td>
<td>0.16644</td>
<td>0.16311</td>
</tr>
<tr>
<td>59</td>
<td>2907</td>
<td>1429</td>
<td>1478</td>
<td>0.34884</td>
<td>0.17736</td>
<td>0.17363</td>
</tr>
<tr>
<td>69</td>
<td>3434</td>
<td>1874</td>
<td>1560</td>
<td>0.4122</td>
<td>0.19212</td>
<td>0.18075</td>
</tr>
<tr>
<td>79</td>
<td>3964</td>
<td>2377</td>
<td>1587</td>
<td>0.47568</td>
<td>0.19044</td>
<td>0.18371</td>
</tr>
<tr>
<td>89</td>
<td>4468</td>
<td>2909</td>
<td>1559</td>
<td>0.53616</td>
<td>0.18792</td>
<td>0.18348</td>
</tr>
<tr>
<td>99</td>
<td>4998</td>
<td>3468</td>
<td>1530</td>
<td>0.59976</td>
<td>0.1836</td>
<td>0.18073</td>
</tr>
<tr>
<td>109</td>
<td>5538</td>
<td>4073</td>
<td>1465</td>
<td>0.66456</td>
<td>0.1758</td>
<td>0.17592</td>
</tr>
<tr>
<td>119</td>
<td>6023</td>
<td>4574</td>
<td>1449</td>
<td>0.72276</td>
<td>0.17388</td>
<td>0.1703</td>
</tr>
<tr>
<td>129</td>
<td>6503</td>
<td>5102</td>
<td>1401</td>
<td>0.78036</td>
<td>0.16812</td>
<td>0.16386</td>
</tr>
<tr>
<td>139</td>
<td>6992</td>
<td>5650</td>
<td>1342</td>
<td>0.83904</td>
<td>0.16104</td>
<td>0.15668</td>
</tr>
<tr>
<td>149</td>
<td>7481</td>
<td>6208</td>
<td>1273</td>
<td>0.89772</td>
<td>0.15276</td>
<td>0.14907</td>
</tr>
<tr>
<td>159</td>
<td>7998</td>
<td>6787</td>
<td>1211</td>
<td>0.95976</td>
<td>0.14532</td>
<td>0.14078</td>
</tr>
<tr>
<td>169</td>
<td>8507</td>
<td>7341</td>
<td>1166</td>
<td>1.02084</td>
<td>0.13992</td>
<td>0.13252</td>
</tr>
<tr>
<td>179</td>
<td>9008</td>
<td>7924</td>
<td>1084</td>
<td>1.08096</td>
<td>0.13008</td>
<td>0.12442</td>
</tr>
<tr>
<td>189</td>
<td>9486</td>
<td>8483</td>
<td>1003</td>
<td>1.13832</td>
<td>0.12036</td>
<td>0.11682</td>
</tr>
<tr>
<td>199</td>
<td>10025</td>
<td>9093</td>
<td>932</td>
<td>1.203</td>
<td>0.11184</td>
<td>0.10848</td>
</tr>
</tbody>
</table>
Slotted Aloha:

<table>
<thead>
<tr>
<th>Number of nodes generating traffic</th>
<th>Total number of Packets Transmitted</th>
<th>Total number of Packets Collided</th>
<th>Total number of Packets Success (Packets Transmitted - Packets Collided)</th>
<th>Throughput per packet time(G)</th>
<th>Number of Packets Transmitted per packet time(S)</th>
<th>Packets per packet time theoretical (S = G * e^(-G))</th>
</tr>
</thead>
<tbody>
<tr>
<td>19</td>
<td>974</td>
<td>111</td>
<td>863</td>
<td>0.11688</td>
<td>0.10356</td>
<td>0.10399</td>
</tr>
<tr>
<td>39</td>
<td>1981</td>
<td>407</td>
<td>1574</td>
<td>0.23772</td>
<td>0.18888</td>
<td>0.18742</td>
</tr>
<tr>
<td>59</td>
<td>2893</td>
<td>891</td>
<td>2002</td>
<td>0.34716</td>
<td>0.24024</td>
<td>0.24534</td>
</tr>
<tr>
<td>79</td>
<td>3946</td>
<td>1504</td>
<td>2442</td>
<td>0.47352</td>
<td>0.29304</td>
<td>0.29491</td>
</tr>
<tr>
<td>99</td>
<td>4976</td>
<td>2286</td>
<td>2690</td>
<td>0.59712</td>
<td>0.3228</td>
<td>0.32865</td>
</tr>
<tr>
<td>119</td>
<td>5996</td>
<td>3144</td>
<td>2852</td>
<td>0.71952</td>
<td>0.34224</td>
<td>0.3504</td>
</tr>
<tr>
<td>139</td>
<td>6961</td>
<td>3999</td>
<td>2962</td>
<td>0.83532</td>
<td>0.35544</td>
<td>0.36231</td>
</tr>
<tr>
<td>159</td>
<td>7967</td>
<td>4974</td>
<td>2993</td>
<td>0.95652</td>
<td>0.35904</td>
<td>0.36752</td>
</tr>
<tr>
<td>179</td>
<td>8969</td>
<td>5994</td>
<td>2975</td>
<td>1.07628</td>
<td>0.357</td>
<td>0.36686</td>
</tr>
<tr>
<td>199</td>
<td>9983</td>
<td>7042</td>
<td>2941</td>
<td>1.19796</td>
<td>0.35292</td>
<td>0.36156</td>
</tr>
<tr>
<td>219</td>
<td>10926</td>
<td>8011</td>
<td>2915</td>
<td>1.31112</td>
<td>0.3498</td>
<td>0.35337</td>
</tr>
<tr>
<td>239</td>
<td>11928</td>
<td>9073</td>
<td>2855</td>
<td>1.43136</td>
<td>0.3426</td>
<td>0.34207</td>
</tr>
<tr>
<td>259</td>
<td>12969</td>
<td>10224</td>
<td>2745</td>
<td>1.55628</td>
<td>0.3294</td>
<td>0.32825</td>
</tr>
<tr>
<td>279</td>
<td>13916</td>
<td>11266</td>
<td>2650</td>
<td>1.66992</td>
<td>0.318</td>
<td>0.31438</td>
</tr>
<tr>
<td>299</td>
<td>14945</td>
<td>12430</td>
<td>2515</td>
<td>1.7934</td>
<td>0.3018</td>
<td>0.29841</td>
</tr>
<tr>
<td>319</td>
<td>15967</td>
<td>13592</td>
<td>2375</td>
<td>1.91604</td>
<td>0.285</td>
<td>0.28202</td>
</tr>
<tr>
<td>339</td>
<td>17011</td>
<td>14765</td>
<td>2246</td>
<td>2.04132</td>
<td>0.26952</td>
<td>0.26508</td>
</tr>
<tr>
<td>359</td>
<td>17977</td>
<td>15895</td>
<td>2082</td>
<td>2.15724</td>
<td>0.24984</td>
<td>0.24947</td>
</tr>
<tr>
<td>379</td>
<td>18983</td>
<td>17010</td>
<td>1973</td>
<td>2.27796</td>
<td>0.23676</td>
<td>0.23348</td>
</tr>
<tr>
<td>399</td>
<td>19987</td>
<td>18146</td>
<td>1841</td>
<td>2.39844</td>
<td>0.22092</td>
<td>0.21792</td>
</tr>
</tbody>
</table>
Thus, the following characteristic plot for the Pure ALOHA and Slotted ALOHA is obtained, which matches the theoretical result.
13. Study the working and routing table formation of Interior routing protocols, i.e. Routing Information Protocol (RIP) and Open Shortest Path First (OSPF)

13.1 Introduction:

**RIP**

RIP is intended to allow hosts and gateways to exchange information for computing routes through an IP-based network. RIP is a distance vector protocol which is based on Bellman-Ford algorithm. This algorithm has been used for routing computation in the network.

Distance vector algorithms are based on the exchange of only a small amount of information using RIP messages.

Each entity (router or host) that participates in the routing protocol is assumed to keep information about all of the destinations within the system. Generally, information about all entities connected to one network is summarized by a single entry, which describes the route to all destinations on that network. This summarization is possible because as far as IP is concerned, routing within a network is invisible. Each entry in this routing database includes the next router to which datagram’s destined for the entity should be sent. In addition, it includes a “metric” measuring the total distance to the entity.

Distance is a somewhat generalized concept, which may cover the time delay in getting messages to the entity, the dollar cost of sending messages to it, etc. Distance vector algorithms get their name from the fact that it is possible to compute optimal routes when the only information exchanged is the list of these distances. Furthermore, information is only exchanged among entities that are adjacent, that is, entities that share a common network.

**OSPF**

In OSPF, the Packets are transmitted through the shortest path between the source and destination.

**Shortest path:** OSPF allows administrator to assign a cost for passing through a link. The total cost of a particular route is equal to the sum of the costs of all links that comprise the route. A router chooses the route with the shortest (smallest) cost.

In OSPF, each router has a link state database which is tabular representation of the topology of the network (including cost). Using Dijkstra algorithm each router finds the shortest path between source and destination.
**Formation of OSPF Routing Table**

1. OSPF-speaking routers send Hello packets out all OSPF-enabled interfaces. If two routers sharing a common data link agree on certain parameters specified in their respective Hello packets, they will become neighbors.

2. Adjacencies, which can be thought of as virtual point-to-point links, are formed between some neighbors. OSPF defines several network types and several router types. The establishment of an adjacency is determined by the types of routers exchanging Hellos and the type of network over which the Hellos are exchanged.

3. Each router sends link-state advertisements (LSAs) over all adjacencies. The LSAs describe all of the router's links, or interfaces, the router's neighbors, and the state of the links. These links might be to stub networks (networks with no other router attached), to other OSPF routers, or to external networks (networks learned from another routing process). Because of the varying types of link-state information, OSPF defines multiple LSA types.

4. Each router receiving an LSA from a neighbor records the LSA in its link-state database and sends a copy of the LSA to all of its other neighbors.

5. By flooding LSAs throughout an area, all routers will build identical link-state databases.

6. When the databases are complete, each router uses the SPF algorithm to calculate a loop-free graph describing the shortest (lowest cost) path to every known destination, with itself as the root. This graph is the SPF tree.

7. Each router builds its route table from its SPF tree

**13.2 Network Setup:**

Open NetSim and click Examples > Experiments > Route-table-formation-in-RIP-and-OSPF > Sample-1 as shown below:
NetSim UI displays the configuration file corresponding to this experiment as shown below:

13.3 Procedure:

**Sample 1:**

The following are the set of procedures were done to generate this sample.

**Step 1:** A network scenario is designed in the NetSim GUI comprising of 2 Wired Nodes, 2 L2 Switches, and 7 Routers.

**Step 2:** Go to Router 1 Properties. In the Application Layer, Routing Protocol is set as RIP.
The Router Configuration Window shown above, indicates the Routing Protocol set as RIP along with its associated parameters. The “Routing Protocol” parameter is Global. i.e. changing in Router 1 will affect all the other Routers. So, in all the Routers, the Routing Protocol is now set as RIP.

**Step 3:** Right click on App1 CUSTOM and select Properties or click on the Application icon present in the top ribbon/toolbar. Transport Protocol is set to UDP.

A CUSTOM Application is generated from Wired Node 10 i.e. Source to Wired Node 11 i.e. Destination with Packet Size remaining 1460Bytes and Inter Arrival Time remaining 20000µs.

**Step 4:** Packet Trace is enabled, and hence we are able to track the route which the packets have chosen to reach the destination based on the Routing Information Protocol that is set.

**Step 5:** Run the Simulation for 100 Seconds.

**Sample 2:**

The following are the set of procedures that are followed to carry out this experiment.

**Step 1:** A network scenario is designed in the NetSim GUI comprising of 2 Wired Nodes, 2 L2 Switches, and 7 Routers.

**Step 2:** Go to Router 1 Properties. In the Application Layer, Routing Protocol is set as OSPF.

![Router Configuration Window](image-url)
The Router Configuration Window shown above, indicates the Routing Protocol set as OSPF along with its associated parameters. The “Routing Protocol” parameter is Global. i.e. changing in Router 1 will affect all the other Routers. So, in all the Routers, the Routing Protocol is now set as OSPF.

**Step 3:** Right click on App1 CUSTOM and select Properties or click on the Application icon present in the top ribbon/toolbar. Transport Protocol is set to UDP.

**Step 4:** Go to Router 7 Properties. In both the WAN Interfaces, the Output Cost is set to 2000.

![Router Configuration Window](image)

The “Output Cost” parameter in the **WAN Interface > Application Layer** of a router indicates the cost of sending a data packet on that interface and is expressed in the link state metric.

**Step 5:** Right click on App1 CUSTOM and select Properties or click on the Application icon present in the top ribbon/toolbar.

A CUSTOM Application is generated from Wired Node 10 i.e. Source to Wired Node 11 i.e. Destination with Packet Size remaining 1460Bytes and Inter Arrival Time remaining 20000µs.

Additionally, the “Start Time (s)” parameter is set to 40, while configuring the application. This time is usually set to be greater than the time taken for OSPF Convergence (i.e. Exchange of OSPF information between all the routers), and it increases as the size of the network increases.

**Step 6:** Packet Trace is enabled, and hence we are able to track the route which the packets have chosen to reach the destination based on the Open Shortest Path First Routing Protocol that is set.

**Step 7:** Run the Simulation for 100 Seconds.
13.4 Output I:

Go to NetSim Packet Animation window and play the animation. The route taken by the packets to reach the destination can be seen in the animation as well as in the below table containing various fields of packet information as shown below:

![NetSim Packet Animation](image)

Users can view the same in Packet Trace.

Shortest Path from Wired Node 10 to Wired Node 11 in RIP is \textbf{Wired Node 10}→\textbf{L2 Switch 8}→\textbf{Router 1}→\textbf{Router 7}→\textbf{Router 6}→\textbf{L2 Switch 9}→\textbf{Wired Node 11}. RIP chooses the lower path (number of hops is less) to forward packets from source to destination, since it is based on hop count.

13.5 Output II:

Go to NetSim Packet Animation window and play the animation. The route taken by the packets to reach the destination can be seen in the animation as well as in the below table containing various fields of packet information as shown below:

![NetSim Packet Animation](image)

![Packet Trace](image)
Users can view the same in Packet Trace.

Shortest Path from Wired Node 10 to Wired Node 11 in OSPF (Use Packet Animation to view) **Wired Node 10->L2 Switch 8->Router 1->Router 2->Router 3->Router 4->Router 5->Router 6->L2 Switch 9->Wired Node 11.** OSPF chooses the above path (cost is less-5) since OSPF is based on cost.

### 13.6 Inference:

**RIP**

In Distance vector routing, each router periodically shares its knowledge about the entire network with its neighbors. The three keys for understanding the algorithm,

1. **Knowledge About The Whole Network** - Router sends all of its collected knowledge about the network to its neighbors.
2. **Routing Only To Neighbors** - Each router periodically sends its knowledge about the network only to those routers to which it has direct links. It sends whatever knowledge it has about the whole network through all of its ports. This information is received and kept by each neighboring router and used to update its own information about the network.
3. **Information Sharing At Regular Intervals** - For example, every 30 seconds, each router sends its information about the whole network to its neighbors. This sharing occurs whether or not the network has changed since the last time, information was exchanged.

In NetSim the Routing Table Formation has 3 stages,

1. **Initial Table**: The Initial Table will show the direct connections made by each Router.
2. **Intermediate Table**: The Intermediate Table will have the updates of the Network in every 30 seconds.
3. **Final Table**: The Final Table is formed when there is no update in the Network.

The data should be forwarded using Routing Table with the shortest distance.

**OSPF**

The main operation of the OSPF protocol occurs in the following consecutive stages, and leads to the convergence of the internetworks:

1. Compiling the LSDB.
2. Calculating the Shortest Path First (SPF) Tree.
3. Creating the routing table entries.
Compiling the LSDB

The LSDB is a database of all OSPF router LSAs. The LSDB is compiled by an ongoing exchange of LSAs between neighboring routers so that each router is synchronized with its neighbor. When the Network converged, all routers have the appropriate entries in their LSDB.

Calculating the SPF Tree Using Dijkstra's Algorithm

Once the LSDB is compiled, each OSPF router performs a least cost path calculation called the Dijkstra algorithm on the information in the LSDB and creates a tree of shortest paths to each other router and network with themselves as the root. This tree is known as the SPF Tree and contains a single, least cost path to each router and in the Network. The least cost path calculation is performed by each router with itself as the root of the tree.

Calculating the Routing Table Entries from the SPF Tree

The OSPF routing table entries are created from the SPF tree and a single entry for each network in the AS is produced. The metric for the routing table entry is the OSPF-calculated cost, not a hop count.

If the application start time isn't changed then,

1. Packets generated before OSPF table convergence may be dropped at the gateway router.
2. The application may also stop if ICMP is enabled in the router
3. If TCP is enabled TCP may stop after the re-try limit is reached (since the SYN packets would not reach the destination)

NOTE: The device / link numbering and IP Address setting in NetSim is based on order in which in the devices are dragged & dropped, and the order in which links are connected. Hence if the order in which a user executes these tasks is different from what is shown in the screen shots, users would notice different tables from what is shown in the screen shots.
14. The M/D/1 Queue

14.1 Motivation:

In this simulation experiment, we will study a model that is important to understand the queuing and delay phenomena in packet communication links. Let us consider the network shown in Figure 14.1. Wired_Node_1 is transmitting UDP packets to Wired_Node_2 through a router. Link 1 and Link 2 are of speed 10 Mbps. The packet lengths are 1250 bytes plus a 54 byte header, so that the time taken to transmit a packet on each 10 Mbps link is $\frac{1304 \times 8}{10} \mu\text{sec} = 1043.2 \mu\text{sec}$. In this setting, we would like answers to the following questions:

1. We notice that the maximum rate at which these packets can be carried on a 10 Mbps link is $\frac{10^6}{1043.2} = 958.59$ packets per second. Can the UDP application send packets at this rate?

2. The time taken for a UDP packet to traverse the two links is $2 \times 1043.2 = 2086.4 \mu\text{sec}$. Is this the time it actually takes for a UDP packet generated at Wired_Node_1 to reach Wired_Node_2.

The answer to these questions depends on the manner in which the UDP packets are being generated at Wired_Node_1. If the UDP packets are generated at intervals of 1043.2 $\mu\text{sec}$ then successive packets will enter the Link 1, just when the previous packet departs. In practice, however, the UDP packets will be generated by a live voice or video source. Depending on the voice activity, the activity in the video scene, and the coding being used for the voice and the video, the rate of generation of UDP packets will vary with time. Suppose two packets were generated during the time that one packet is sent out on Link 1, then one will have to wait, giving rise to queue formation. This also underlines the need for a buffer to be placed before each link; a buffer is just some dynamic random-access memory in the link interface card into which packets can be stored while waiting for the link to free up.

Queuing models permit us to understand the phenomenon of mismatch between the service rate (e.g., the rate at which the link can send out packets) and the rate at which packets arrive. In the network in Figure 14.1, looking at the UDP flow from Wired_Node_1 to Wired_Node_2, via Router 1, there are two places at which queueing can occur. At the interface between Wired_Node_1 and Link 1, and at the interface between Router 1 and Link 2. Since the only flow of packets is from Wired_Node_1 to Wired_Node_2, all the packets entering Link 2 are from Link 1, and these are both of the same bit rate. Link 2, therefore, cannot receive packets faster than it can serve them and, at any time, only the packet currently in transmission will be at Link 2. On the other hand at the Wired_Node_1 to Link 1 interface, the packets are generated directly by the application, which can be at arbitrary rates, or inter-packet times.
Suppose that, at Wired_Node_1, the application generates the successive packets such that the
time intervals between the successive packets being generated are statistically independent, and
the probability distribution of the time intervals has a negative exponential density, i.e., of the form
$\lambda e^{-\lambda x}$, where $\lambda$ (packets per second) is a parameter, called the rate parameter, and $x$ (seconds) is
the argument of the density. The application generates the entire packet instantaneously, i.e., all the
bits of the packet arrive from the application together, and enter the buffer at Link 1, to wait behind
the other packets, in a first-in-first-out manner. The resulting random process of the points at which
packets enter the buffer of Link 1 is called a Poisson Process of rate $\lambda$ packets per second. The
buffer queues the packets while Link 1 serves them with service time $b = 1043.2 \mu$sec. Such a queue
is called an M/D/1 queue, where the notation is to be read as follows:

- The M before the first slash (denoting “Markov”) denotes the Poisson Process of instants at
  which packets enter the buffer
- The D between the two slashes (denoting “Deterministic”) denotes the fixed time taken to
  serve each queued packet
- The 1 after the second slash denotes that there is just a single server (Link 1 in our example)

This way of describing a single server queueing system is called Kendall’s Notation.

In this experiment, we will understand the M/D/1 model by simulating the above described network
on NetSim. The M/D/1 queueing model, however, is simple enough that it can be mathematically
analysed in substantial detail. We will summarise the results of this analysis in the next section. The
simulation results from NetSim will be compared with the analytical results.

### 14.2 Mathematical Analysis of the M/D/1 Queue:

The M/D/1 queueing system has a random number of arrivals during any time interval. Therefore,
the number of packets waiting at the buffer is also random. It is possible to mathematically analyse
the random process of the number of waiting packets. The procedure for carrying out such analysis
is, however, beyond the scope of this document. We provide the final formulas so that the simulation
results from NetSim can be compared with those provided by these formulas.

As described earlier, in this chapter, the M/D/1 queue is characterized by two parameters: $\lambda$ (packets
per second), which is the arrival rate of packets into the buffer, and $\mu$ (packets per second), which is
the rate at which packets are removed from a nonempty queue. Note that $1/\mu$ is the service time of
each packet.

Define $\rho = \lambda \times \frac{1}{\mu} = \lambda/\mu$. We note that $\rho$ is the average number of packets that arrive during the
service time of a packet. Intuitively, it can be expected that if $\rho > 1$ then packets arrive faster than
the rate at which they can be served, and the queue of packets can be expected grow without bound.
When $\rho < 1$ we can expect the queue to be "stable." When $\rho = 1$, the service rate is exactly matched with the arrival rate; due to the randomness, however, the queue can still grow without bound. The details of this case are beyond the scope of this document.

For the $k^{th}$ arriving packet, denote the instant of arrival by $a_k$, the instant at which service for this packet starts as $s_k$, and the instant at which the packet leaves the system as $d_k$. Clearly, for all $k$, $d_k - s_k = \frac{1}{\mu}$, the deterministic service time. Further define, for each $k$,

$$W_k = s_k - a_k$$

$$T_k = d_k - a_k$$

i.e., $W_k$ is called the queuing delay, i.e., time from the arrival of the $k^{th}$ packet until it starts getting transmitted, whereas $T_k$ is called the total delay, i.e., the time from the arrival of the $k^{th}$ packet until its transmission is completed. Considering a large number of packets, we are interested in the average of the values $W_1, W_2, W_3, \ldots$, i.e., the average queueing time of the packets. Denote this average by $W$. By mathematical analysis of the packet queue process, it can be shown that for an M/D/1 queueing system,

$$W = \frac{1}{2\mu} \times \frac{\rho}{1 - \rho}$$

Denoting by $T$, the average total time in the system (i.e., the average of $T_1, T_2, T_3, \ldots$), clearly $T = W + \frac{1}{\mu}$

Observe the following from the above formula:

1. As $\rho$ approaches 0, $W$ becomes 0. This is clear, since, when the arrival rate becomes very small, and arriving packet sees a very small queue. For arrival rate approaching 0, packets get served immediately on arrival.

2. As $\rho$ increases, $W$ increases.

3. As $\rho$ approaches 1 (from values smaller than 1), the mean delay goes to $\infty$.

We will verify these observations in the NetSim simulation.

### 14.3 The Experimental Setup

The model described at the beginning of this chapter is shown in Figure 14.1
Figure 14.1 A single wired node (Wired_Node_1) sending UDP packets to another wired node (Wired_Node_2) through a router (Router 3). The packet interarrival times at Wired_Node_1 are exponentially distributed, and packets are all of the same length, i.e., 1250 bytes plus UDP/IP header.

Open NetSim and Click on Examples > Experiments > The-M/D/1-Queue > Sample-1 as shown below:

NetSim UI displays the configuration file corresponding to this experiment as shown above:

14.4 Procedure:

Sample 1:
The following set of procedures were done to generate this sample:

**Step 1:** A network scenario is designed in NetSim GUI comprising of 2 Wired Nodes and 1 Router in the “Internetworks” Network Library.

**Step 2:** Link Properties are set as per the table given below:

<table>
<thead>
<tr>
<th>Link Properties</th>
<th>Link 1</th>
<th>Link 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Uplink Speed (Mbps)</td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td>Downlink Speed (Mbps)</td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td>Uplink BER</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Downlink BER</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Uplink Propagation Delay (µs)</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Downlink Propagation Delay (µs)</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

**Step 3:** Right click on the Application Flow App1 CUSTOM and select Properties or click on the Application icon present in the top ribbon/toolbar.

A CUSTOM Application is generated from Wired Node 1 i.e. Source to Wired Node 2 i.e. Destination. Transport Protocol is set to UDP with Packet Size set to 1250 Bytes and Inter Arrival Time set to 104319 µs and distribution to Exponential.

The Packet Size and Inter Arrival Time parameters are set such that the Generation Rate equals 0.096 Mbps. Generation Rate can be calculated using the formula:

\[
\text{Generation Rate (Mbps)} = \text{Packet Size (Bytes)} \times \frac{8}{\text{Interarrival time (µs)}}
\]

**Step 4:** Packet Trace is enabled in NetSim GUI. At the end of the simulation, a very large .csv file is containing all the packet information is available for the users to perform packet level analysis.

**Step 5:** Run the Simulation for 1000 Seconds.

Similarly, the other samples are created by changing the Inter Arrival Time per the formula

\[
IAT = \frac{10^6}{958.59 \times \rho}
\]

as per the table given below

<table>
<thead>
<tr>
<th>ρ</th>
<th>IAT (µs)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.01</td>
<td>104319</td>
</tr>
<tr>
<td>0.05</td>
<td>20863</td>
</tr>
</tbody>
</table>
Even though the packet size at the application layer is 1250 bytes, as the packet moves down the layers, overhead is added. The overheads added in different layers are shown in the below table and can be obtained from the packet trace:

<table>
<thead>
<tr>
<th>Layer</th>
<th>Overhead (Bytes)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Transport Layer</td>
<td>8</td>
</tr>
<tr>
<td>Network Layer</td>
<td>20</td>
</tr>
<tr>
<td>MAC layer</td>
<td>26</td>
</tr>
<tr>
<td>Physical Layer</td>
<td>0</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>54</strong></td>
</tr>
</tbody>
</table>

Table 4: Overheads added to a packet as it flows down the network stack

14.5 Obtaining the Mean Queuing delay from the Simulation Output:

After running the simulation, note down the “Mean Delay” in the Application Metrics within the Results Dashboard. This is the average time between the arrival of packets into the buffer at Wired_Node_1, and their reception at Wired_Node_2.
As explained in the beginning of this chapter, for the network shown in Figure 14.1, the end-to-end delay of a packet is the sum of the queueing delay at the buffer between the wired-node and Link_1, the transmission time on Link_1, and the transmission time on Link_2 (there being no queueing delay between the Router and Link_2). It follows that

\[
\text{Mean Delay} = \left( \frac{1}{2\mu} \times \frac{\rho}{1 - \rho} \right) + \frac{1}{\mu} + \frac{1}{\mu}
\]

### 14.6 Output Table

<table>
<thead>
<tr>
<th>Sample</th>
<th>(\rho)</th>
<th>(\lambda)</th>
<th>Mean Delay ((\mu s))</th>
<th>Queuing Delay ((\mu s)) (Simulation)</th>
<th>Queuing Delay ((\mu s)) (Theory)</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>0.05</td>
<td>47.93</td>
<td>2113.53</td>
<td>27.13</td>
<td>27.45</td>
</tr>
<tr>
<td>3</td>
<td>0.10</td>
<td>95.86</td>
<td>2145.10</td>
<td>58.70</td>
<td>57.96</td>
</tr>
<tr>
<td>4</td>
<td>0.15</td>
<td>143.79</td>
<td>2179.27</td>
<td>92.87</td>
<td>92.05</td>
</tr>
<tr>
<td>5</td>
<td>0.20</td>
<td>191.72</td>
<td>2217.93</td>
<td>131.53</td>
<td>130.40</td>
</tr>
<tr>
<td>6</td>
<td>0.25</td>
<td>239.65</td>
<td>2260.98</td>
<td>174.58</td>
<td>173.87</td>
</tr>
<tr>
<td>7</td>
<td>0.30</td>
<td>287.58</td>
<td>2309.81</td>
<td>223.41</td>
<td>223.54</td>
</tr>
<tr>
<td>8</td>
<td>0.35</td>
<td>335.51</td>
<td>2366.81</td>
<td>280.41</td>
<td>280.86</td>
</tr>
<tr>
<td>9</td>
<td>0.40</td>
<td>383.44</td>
<td>2434.61</td>
<td>348.21</td>
<td>347.73</td>
</tr>
<tr>
<td>10</td>
<td>0.45</td>
<td>431.37</td>
<td>2514.11</td>
<td>427.71</td>
<td>426.76</td>
</tr>
<tr>
<td>11</td>
<td>0.50</td>
<td>479.30</td>
<td>2609.91</td>
<td>523.51</td>
<td>521.60</td>
</tr>
<tr>
<td>12</td>
<td>0.55</td>
<td>527.22</td>
<td>2726.41</td>
<td>640.01</td>
<td>637.51</td>
</tr>
<tr>
<td>13</td>
<td>0.60</td>
<td>575.15</td>
<td>2871.75</td>
<td>785.35</td>
<td>782.40</td>
</tr>
<tr>
<td>14</td>
<td>0.65</td>
<td>623.08</td>
<td>3059.09</td>
<td>972.69</td>
<td>968.68</td>
</tr>
<tr>
<td>15</td>
<td>0.70</td>
<td>671.01</td>
<td>3304.49</td>
<td>1218.09</td>
<td>1217.07</td>
</tr>
<tr>
<td>16</td>
<td>0.75</td>
<td>718.94</td>
<td>3653.13</td>
<td>1566.73</td>
<td>1564.80</td>
</tr>
<tr>
<td>17</td>
<td>0.80</td>
<td>766.87</td>
<td>4166.87</td>
<td>2080.47</td>
<td>2086.40</td>
</tr>
<tr>
<td>18</td>
<td>0.85</td>
<td>814.80</td>
<td>5026.42</td>
<td>2940.02</td>
<td>2955.73</td>
</tr>
<tr>
<td>19</td>
<td>0.90</td>
<td>862.73</td>
<td>6750.47</td>
<td>4664.07</td>
<td>4694.39</td>
</tr>
<tr>
<td>20</td>
<td>0.95</td>
<td>910.66</td>
<td>12088.08</td>
<td>10001.68</td>
<td>9910.39</td>
</tr>
</tbody>
</table>

### 14.7 Advanced Topic: The M/G/1 Queue

In Section 14.1, we introduced the M/D/1 queue. Successive packets were generated instantly at exponentially distributed time intervals (i.e., at the points of a Poisson process); this gave the “M” in the notation. The packets were all of fixed length; this gave the “D” in the notation. Such a model was motivated by the transmission of packetized voice over a fixed bit rate wireline link. The voice
samples are packetised into constant length UDP packets. For example, typically, 20ms of voice
samples would make up a packet, which would be emitted at the instant that the 20ms worth of voice
samples are collected. A voice source that is a part of a conversation would have listening periods,
and “silence” periods between words and sentences. Thus, the intervals between emission instants
of successive UDP packets would be random. A simple model for these random intervals is that they
are exponentially distributed, and independent from packet to packet. This, formally, is called the
Poisson point process. With exponentially distributed (and independent) inter-arrival times, and fixed
length packets we obtain the M/D/1 model. On the other hand, some applications, such as video,
generate unequal length packets. Video frames could be encoded into packets. To reduce the
number of bits being transmitted, if there is not much change in a frame, as compared to the previous
one, then the frame is encoded into a small number of bits; on the other hand if there is a large
change then a large number of bits would need to be used to encode the new information in the
frame. This motivates variable packet sizes. Let us suppose that, from such an application, the
packets arrive at the points of a Poisson process of rate \( \lambda \), and that the randomly varying packet
transmission times can be modelled as independent and identically distributed random variables,
\( B_1, B_2, B_3, \ldots \), with mean \( b \) and second moment \( b^{(2)} \), i.e., variance \( b^{(2)} - b^2 \). Such a model is denoted
by M/G/1, where M denotes the Poisson arrival process, and G (“general”) the “generally” distributed
service times. Recall the notation M/D/1 (from earlier in this section), where the D denoted fixed (or
“deterministic”) service times. Evidently, the M/D/1 model is a special case of the M/G/1 model.

Again, as defined earlier in this section, let \( W \) denote the mean queueing delay in the M/G/1 system.
Mathematical analysis of the M/G/1 queue yields the following formula for \( W \)

\[
W = \frac{\rho}{1 - \rho} \frac{b^{(2)}}{2b}
\]

where, as before, \( \rho = \lambda b \). This formula is called the Pollacek-Khinchine formula or P-K formula, after
the researchers who first obtained it. Denoting the variance of the service time by \( Var(B) \), the P-K
formula can also be written as

\[
W = \frac{\rho b}{2(1 - \rho)} \left( \frac{Var(B)}{b^2} + 1 \right)
\]

Applying this formula to the M/D/1 queue, we have \( Var(B) = 0 \). Substituting this in the M/G/1
formula, we obtain

\[
W = \frac{\rho}{1 - \rho} \frac{b}{2}
\]

which, with \( b = 1/\mu \), is exactly the M/D/1 mean queuing delay formula displayed earlier in this
section.
14.8 A NetSim Exercise Utilising the M/G/1 Queue

In this section we demonstrate the use of the M/G/1 queueing model in the context of the network setup shown in Figure 14.1. The application generates exponentially distributed data segment with mean $d$ bits, i.e., successive data segment lengths are sampled independently from an exponential distribution with rate parameter $\frac{1}{d}$. Note that, since packets are integer multiples of bits, the exponential distribution will only serve as an approximation. These data segments are then packetised by adding a constant length header of length $h$ bits. The packet generation instants form a Poisson process of rate $\lambda$. Let us denote the link speed by $c$.

Let us denote the random data segment length by $X$ and the packet transmission time by $B$, so that

$$B = \frac{X + h}{c}$$

Denoting the mean of $B$ by $b$, we have

$$b = \frac{d + h}{c}$$

Further, since $h$ is a constant,

$$Var(B) = Var(X)/c^2$$

These can now be substituted in the P-K formula to obtain the mean delay in the buffer between Node 1 and Link 1.

We set the mean packet size to 100B or 800 bits, the header length $h = 54B$ or 432 bits and $\lambda = 5000$

For a 10Mbps link, the service rate $\mu = \frac{10 \times 10^6}{154 \times 8} = 8116.8$

Using the Pollaczek–Khinchine (PK) formula, the waiting time for a M/G/1 queuing system is

$$w = \frac{\rho + \lambda \times \mu \times Var(s)}{2(\mu - \lambda)}$$

Where $var(s)$ is the variance of the service time distribution $S$. Note that

$$var(s) = \frac{1}{(\mu')^2}$$

where $\mu'$ is the mean service time of the exponential random variable (100B packets and not 154B)

$$\mu' = \frac{10 \times 10^6}{100 \times 8} = 12500$$

Hence substituting into the PK formula, one gets
\[
w = 0.4 + \frac{(3467.7 \times 8116.8)}{1250^2} = 59.5 \mu s
\]

By simulation the queuing delay is 60.5 \( \mu s \).

The queuing delay is not available in the NetSim results dashboard. It can be got from the packet trace. It is the average of (PHY\_layer\_Arrival\_time - APP\_layer\_arrival\_time) for packets being sent from Node_1.
15. Quality of Service (QoS) in 802.11e based WLANs

15.1 Theory:

IEEE 802.11e Medium Access Control (MAC) is a supplement to the IEEE 802.11 Wireless Local Area Network (WLAN) standard to support Quality-of-Service (QoS). When 802.11e is enabled high-priority traffic has a higher chance of being sent than low-priority traffic: an application with high priority traffic waits a little less before its packet is processed and compared to an application with low priority traffic. The various application traffic generated in NetSim have the following priority and QoS values:

<table>
<thead>
<tr>
<th>Application Type</th>
<th>Priority Value</th>
<th>Priority</th>
<th>QoS Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>Voice – One way</td>
<td>8</td>
<td>Medium</td>
<td>RTPS</td>
</tr>
<tr>
<td>Voice – Two way</td>
<td>8</td>
<td>High</td>
<td>UGS</td>
</tr>
<tr>
<td>Video</td>
<td>6</td>
<td>Low</td>
<td>nRTPS</td>
</tr>
<tr>
<td>FTP</td>
<td>2</td>
<td>Low</td>
<td>BE</td>
</tr>
<tr>
<td>Database</td>
<td>2</td>
<td>Low</td>
<td>BE</td>
</tr>
<tr>
<td>Custom</td>
<td>2</td>
<td>Low</td>
<td>BE</td>
</tr>
</tbody>
</table>

eRTPS QoS class is available in NetSim which has a priority value of 4. The QoS class for each application mentioned in the table above is fixed and can be changed by the user.

15.2 Network Setup:

Open NetSim and click on Examples > Experiments > Quality-of-Service-(QOS)-in-802.11e-based-WLANs > Sample-1 as shown below:
NetSim UI displays the configuration file corresponding to this experiment as shown below:

![NetSim UI screenshot]

15.3 Procedure:

Sample 1:
The following set of procedures were done to generate this sample:

Step 1: A network scenario is designed in NetSim GUI comprising of 1 Wired Node, 2 Wireless Nodes, 1 Router, and 1 Access Point in the “Internetworks” Network Library.

Step 2: The device positions are set as per the below table:

<table>
<thead>
<tr>
<th>Access Point 3</th>
<th>Wireless Node 4</th>
<th>Wireless Node 5</th>
</tr>
</thead>
<tbody>
<tr>
<td>X/Lat</td>
<td>250</td>
<td>300</td>
</tr>
<tr>
<td>Y/Lon</td>
<td>100</td>
<td>100</td>
</tr>
</tbody>
</table>
Step 3: Wired Link Properties is set as follows:

![Link Properties Window]

Step 4: Go to Wireless Link Properties, the “Channel Characteristics” is set to NO PATHLOSS.

Step 5: In the Interface Wireless > Data Link Layer Properties of the Access Point, IEEE 802.11e is set to Enable and Buffer Size is set to 5MB.

![Accesspoint]

Step 6: Right click on the Application Flow App1 VOICE or App2 CBR and select Properties or click on the Application icon present in the top ribbon/toolbar. Transport Protocol is set to UDP.

A VOICE Application is generated from Wired Node 1 i.e. Source to Wireless Node 4 i.e. Destination with Packet Size set to 1000 Bytes and Inter Arrival Time set to 800μs. The “Codec” parameter is set to Custom.
A CBR Application is generated from Wired Node 1 i.e. Source to Wireless Node 5 i.e. Destination with Packet Size set to 1000 Bytes and Inter Arrival Time set to 800µs.

The Packet Size and Inter Arrival Time parameters are set such that the Generation Rate equals 10 Mbps. Generation Rate can be calculated using the formula:

\[
\text{Generation Rate (Mbps)} = \frac{\text{Packet Size (Bytes)} \times 8}{\text{Interarrival time (µs)}}
\]

**Step 7:** Run the Simulation for 10 Seconds. Note down the Application Throughput.

**Sample 2:**

The following changes in settings are done from the previous sample:

**Step 1:** In the **Interface Wireless > Datalink Layer** Properties of the Wireless Node 5 and Access Point, IEEE 802.11e is set to Disable.

**Step 2:** Run the Simulation for 10 Seconds. Note down the Application Throughput.

### 15.4 Output:

<table>
<thead>
<tr>
<th>IEEE 802.11e</th>
<th>Application</th>
<th>Generation rate (Mbps)</th>
<th>Throughput (Mbps)</th>
<th>Delay (Micro. Sec.)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Enable</strong></td>
<td>Voice</td>
<td>10</td>
<td>3.22</td>
<td>945561.8</td>
</tr>
<tr>
<td></td>
<td>CBR</td>
<td>10</td>
<td>2.14</td>
<td>6466262.9</td>
</tr>
<tr>
<td><strong>Disable</strong></td>
<td>Voice</td>
<td>10</td>
<td>2.64</td>
<td>3672706.7</td>
</tr>
<tr>
<td></td>
<td>CBR</td>
<td>10</td>
<td>2.64</td>
<td>3671315.4</td>
</tr>
</tbody>
</table>

### 15.5 Inference:

In sample 1, since QoS is enabled voice sees a higher priority than CBR. Hence voice packets in the queue are first transmitted before CBR packets are transmitted. In sample 2, since QoS has been disabled, priority is not considered for the applications. Hence they both see the same throughput.

As an additional note, when QoS is enabled the throughput for voice is 3.22 Mbps and for CBR it is 2.14, and when QoS is disabled the throughput for both is 2.64 Mbps per application or 5.28 Mbps for both applications put together. This value of around 5.5 Mbps is the maximum throughput an 802.11b access point can support. There is a slight drop in overall throughput when stations are present due to contention between the two stations.
16. Analyze the performance of FIFO, Priority and WFQ Queuing Disciplines

16.1 Introduction

As part of the resource allocation mechanisms, each router must implement some queuing discipline that governs how packets are buffered while waiting to be transmitted. Various queuing disciplines can be used to control which packets get transmitted (based on bandwidth allocation) and which packets get dropped (based on buffer space). The queuing discipline also affects the latency experienced by a packet, by determining how long a packet waits to be transmitted. Examples of the common queuing disciplines are first-in-first-out (FIFO) queuing, priority queuing (PQ), and weighted-fair queuing (WFQ).

16.2 Network Setup:

Open NetSim and click on Examples > Experiments > FIFO-Priority-and-WFQ-Queuing > Sample-1 as shown below:

![NetSim UI with configuration file](image)

NetSim UI displays the configuration file corresponding to this experiment as shown below:
16.3 Procedure:

Sample 1: (FIFO)

The following set of procedures were done to generate this sample:

Step 1: A network scenario is designed in NetSim GUI comprising of 4 Wired Nodes and 2 Routers in the “Internetworks” Network Library.

Step 2: Wired Link Properties is set as follows:

<table>
<thead>
<tr>
<th>Link Properties</th>
<th>Link 1</th>
<th>Link 2</th>
<th>Link 3</th>
<th>Link 4</th>
<th>Link 5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Max Uplink Speed (Mbps)</td>
<td>10</td>
<td>10</td>
<td>10</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>Max Downlink Speed (Mbps)</td>
<td>10</td>
<td>10</td>
<td>10</td>
<td>5</td>
<td>5</td>
</tr>
</tbody>
</table>

Step 3: In the Interface WAN > Network Layer Properties of Router 1, Scheduling Type is set as FIFO. Similarly, Scheduling Type is set as FIFO for Router 2.

Step 4: Three different applications are generated as per the table given below:

NOTE: For Voice application set codec as Custom.

<table>
<thead>
<tr>
<th>Application Properties</th>
<th>Application 1</th>
<th>Application 2</th>
<th>Application 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Application Type</td>
<td>Voice (Codec-Custom)</td>
<td>Video</td>
<td>Custom</td>
</tr>
<tr>
<td>Source_Id</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>Destination_Id</td>
<td>6</td>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td>QoS</td>
<td>RTPS</td>
<td>NRTPS</td>
<td>BE</td>
</tr>
<tr>
<td>Transport Protocol</td>
<td>UDP</td>
<td>UDP</td>
<td>UDP</td>
</tr>
<tr>
<td>Packet Size</td>
<td>1460</td>
<td>50</td>
<td>1000</td>
</tr>
<tr>
<td>Distribution</td>
<td>Constant</td>
<td>Frame_Per_Sec</td>
<td>Constant</td>
</tr>
</tbody>
</table>
### Step 5:
Run the Simulation for 10 Seconds. Note down the Application Throughput.

The following changes in settings are done from the previous sample:

#### Sample 2: (Priority)

##### Step 1:
In the Interface WAN > Network Layer Properties of Router 1, Scheduling Type is set as PRIORITY. Similarly, Scheduling Type is set as PRIORITY for Router 2.

##### Step 2:
Run the Simulation for 10 Seconds. Note down the Application Throughput.

#### Sample 3: (WFQ)

##### Step 1:
In the Interface WAN > Network Layer Properties of Router 1, Scheduling Type is set as WFQ. Similarly, Scheduling Type is set as WFQ for Router 2.

##### Step 2:
Run the Simulation for 10 Seconds. Note down the Application Throughput.

### 16.4 Measurements and Outputs:

<table>
<thead>
<tr>
<th>Application</th>
<th>Traffic Generation Rate (Mbps)*</th>
<th>FIFO-Sample-1 Throughput (Mbps)</th>
<th>Priority-Sample-2 Throughput (Mbps)</th>
<th>WFQ-Sample-3 Throughput (Mbps)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Voice</td>
<td>5</td>
<td>~ (5 / 13.6) *5</td>
<td>3.99</td>
<td>1.92</td>
</tr>
<tr>
<td>Video</td>
<td>2.6</td>
<td>~ (2.6/13.6) *5</td>
<td>0.22</td>
<td>0.84</td>
</tr>
<tr>
<td>Custom</td>
<td>6</td>
<td>~ (6/13.6) *5</td>
<td>0.59</td>
<td>2.01</td>
</tr>
<tr>
<td>Total</td>
<td>13.6</td>
<td>4.77</td>
<td>4.80</td>
<td>4.77</td>
</tr>
</tbody>
</table>

**NOTE:** For Traffic Generation Rate calculation please refer user manual section 6.3.

*The traffic generation rate is based on settings done in step 4.

The 5 mentioned above refers to 5 Mbps which is the data rate of link 4.
16.5 Inference

In FIFO, packets will get served based on their packet arrival time to router. Therefore, since link 4 is a 5 Mbps link, the throughputs of Voice, Video and Custom applications is equal to the ratio of their generation rates.

Priority scheduling technique processes packets based on their priority. Hence voice and video which have higher priority take up the complete bandwidth available.

Weighted fair queuing (WFQ) assigns a weight to each application and hence gives a result between that is in between priority and FIFO.
17. Cyber physical systems (CPS) and IoT – An Introduction

17.1 Introduction:

**Cyber Physical Systems (CPS)** are systems that link the physical world (e.g., through sensors or actuators) with the virtual world of information processing. This does not just mean the convergence. Many systems can be categorized as CPS. Let us consider the example of a smart grids. On the demand side the various domestic appliances (of end users) constitute the physical components, and data of demand load are collected by smart meters. These smart meters connect the physical world to cyber space. The demand load data is transferred via two-way communication channels that are used to measure and control the physical components. On the cyber (cloud) side computations are carried out by the objectives of utility maximization and cost minimization. Based on this a suitable real-time electricity price is calculated.

**Internet of Things (IoT)** is a network of physical devices, vehicles, buildings and other items embedded with electronics, software, sensors, actuators, and network connectivity that enable these objects to collect and exchange data. The IoT network allows objects to be sensed and/or controlled remotely across existing network infrastructure, creating opportunities for more direct integration of the physical world into computer-based systems, and resulting in improved efficiency, accuracy and economic benefit.

IoT is the platform on which cyber physical systems run.

![Diagram of IOT Network Components and TCP/IP stack](image)

Table 5: The IOT Network Components and the TCP/IP stack running in the network devices
17.2 Components of IoT:

1. **Sensors**: Sensors are used to detect physical phenomena such as light, heat, pressure, temperature, humidity etc. Sensors are regarded as a revolutionary information gathering method to build the information and communication system which will greatly improve the reliability and efficiency of infrastructure systems. It follows IPv6 addressing system. IP addresses are the backbone to the entire IoT ecosystem. IPv6’s huge increase in address space is an important factor in the development of the Internet of Things.

2. **LowPAN Gateway**: These are the Gateways to Internet for all the things/devices that we want to interact with. Gateway help to bridge the internal network of sensor nodes with the external Internet i.e., it will collect the data from sensors and transmitting it to the internet infrastructure. A 6LowPAN Gateway will have 2 interfaces, one is Zigbee interface connected to sensors (follows 802.15.4 MAC and PHY) and the other is WAN interface connected to ROUTER.

![Fig 2: The 6LowPAN Gateway’s TCP/IP Stack at the wired and wireless Interfaces](image)

**6LoWPAN** is an acronym of IPv6 over Low Power Wireless Personal Area Network. The 6LoWPAN concept originated from the idea that “the Internet Protocol should be applied even to the smallest devices, and that low-power devices with limited processing capabilities should be able to participate in the Internet of Things.

17.3 Network Setup:

Open NetSim and click Examples > Experiments > Introduction-to-cyber-physical-systems-(CPS)-and-IoT as shown below:
NetSim UI displays the configuration file corresponding to this experiment as shown below:

**Sample 1:**

![NetSim UI configuration](image)

### 17.4 Procedure:

The following set of procedures were done to generate this sample:

**Step 1:** A network scenario is designed in NetSim GUI comprising of 4 Wireless Sensors, 1 Gateway, 1 Router, and 1 Wired Node in the “Internet of Things” Network Library.

**Step 2:** Before we actually designed this network, in the **Fast Config Window** containing inputs for **Grid/Map Settings and Sensor Placement**, the Grid Length and Side Length were set to 500
and 250 meters respectively, instead of the default 100 and 50 meters and we have chosen **Manually Via Click and Drop** option.

**Step 3:** The **Ad hoc Link** is used to link all the Sensors and the Gateway in an ad hoc basis.

The Ad hoc link properties is set to **NO PATHLOSS** for the channel characteristics.

**Step 4:** Right click on the Application Flow **App1 Sensor App** and select Properties or click on the Application icon present in the top ribbon(toolbar).

A Sensor Application is generated from Wireless Sensor 2 i.e. Source to Wired Node 7 i.e. Destination with Packet Size remaining 50 Bytes and Inter Arrival Time remaining 1000000 µs.

**Step 5:** Enable the packet trace and run the Simulation for 100 Seconds.

### 17.5 Output:

![Simulation Output Graph](image)

<table>
<thead>
<tr>
<th>PACKET_ID</th>
<th>SEGMENT_ID</th>
<th>PACKET_TYPE</th>
<th>CONTROL PACKET_PAYLOAD</th>
<th>SOURCE_ID</th>
<th>DESTINATION_ID</th>
<th>TRANSMITTER_ID</th>
<th>RECEIVER_ID</th>
<th>APP_LAYER</th>
<th>ARRIVAL_TIME</th>
<th>DROP_TIME</th>
<th>SIZE</th>
<th>LAYER</th>
<th>METHOD</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>N/A</td>
<td>Control_Packet</td>
<td>AODV_REQ</td>
<td>SENSOR-2</td>
<td>SENSOR-2</td>
<td>SENSOR-0</td>
<td>SENSOR-5</td>
<td>N/A</td>
<td>N/A</td>
<td>2000000</td>
<td>2000000</td>
<td>2000000</td>
<td>2000000</td>
</tr>
<tr>
<td>0</td>
<td>N/A</td>
<td>Control_Packet</td>
<td>AODV_REQ</td>
<td>SENSOR-2</td>
<td>SENSOR-2</td>
<td>SENSOR-0</td>
<td>SENSOR-5</td>
<td>N/A</td>
<td>N/A</td>
<td>2000000</td>
<td>2000000</td>
<td>2000000</td>
<td>2000000</td>
</tr>
<tr>
<td>0</td>
<td>N/A</td>
<td>Control_Packet</td>
<td>AODV_REQ</td>
<td>SENSOR-2</td>
<td>SENSOR-2</td>
<td>SENSOR-0</td>
<td>SENSOR-5</td>
<td>N/A</td>
<td>N/A</td>
<td>2000000</td>
<td>2000000</td>
<td>2000000</td>
<td>2000000</td>
</tr>
<tr>
<td>0</td>
<td>N/A</td>
<td>Control_Packet</td>
<td>AODV_REQ</td>
<td>SENSOR-2</td>
<td>SENSOR-2</td>
<td>SENSOR-0</td>
<td>SENSOR-5</td>
<td>N/A</td>
<td>N/A</td>
<td>2000000</td>
<td>2000000</td>
<td>2000000</td>
<td>2000000</td>
</tr>
</tbody>
</table>

**Sample 2:**
The following changes in settings are done from the previous sample:

**Step 1:** We have only one Sensor and the Sensor Application is generated between that Sensor and the Wired Node.

**Step 2:** Run the Simulation for 10 Seconds.

### 17.6 Output

Users can understand how the IP addresses are changing from IPv6 to IPv4 and vice versa with the help of packet trace file.

After simulation, open packet trace and filter PACKET_TYPE to Sensing and observe the columns SOURCE_IP, DESTINATION_IP, GATEWAY_IP and NEXT_HOP_IP

**SOURCE_IP** – source node IP

**DESTINATION_IP** – gateway IP

**GATEWAY_IP** – IP of the device which is transmitting a packet

**NEXT_HOP_IP** – IP of the next hop

1. Sensor and 6_LOWPAN_Gateways 1st interface follows IPv6 addressing.
2. 6_LOWPAN_Gateways 2nd interface, Router and Wired Node follows IPv4 addressing.
3. From the screenshot below, users can identify the changing of IP addresses from source to destination.
18. One Hop IoT Network over IEEE 802.15.4

18.1 Introduction

The concept of Cyber Physical Systems (CPS) over the Internet of Things (IoT) was explained in Experiment 17: Introduction and working of IoT. In most situations, due to the practical difficulty of laying copper or optical cables to connect sensors and actuators, digital wireless communication has to be used. In such applications, since the energy available in the sensor and actuator devices is small, there is a need for keeping costs low, and the communication performance requirement (in terms of throughput and delay is limited) several wireless technologies have been developed, with the IEEE 802.15.4 standard being one of the early ones.

The IEEE Standard 802.15.4 defines PHY and MAC protocols for low-data-rate, low-power, and low-complexity short-range radio frequency (RF) digital transmissions.

In this experiment, we will study the simplest IEEE 802.15.4 network with one wireless node transmitting packets to an IEEE 802.15.4 receiver, from where the packets are carried over a high speed wireline network to a compute server (where the sensor data would be analyzed).

18.2 The IEEE 802.15.4 PHY and MAC

We will study the IEEE 802.15.4 standard that works in the 2.4 GHz ISM band, in which there is an 80 MHz band on which 16 channels are defined, each of 2 MHz, with a channel separation of 5 MHz. Each IEEE 802.15.4 network works in one of these 2 MHz channels, utilizing spread spectrum communication over a chip-stream of 2 million chips per second. In this chip-stream, 32 successive chips constitute one symbol, thereby yielding 62,500 symbols per second (62.5 Kbps; \( \frac{(2\times10^6)}{32} \)). Here, we observe that a symbol duration is \( 32 \times \frac{1}{2\times10^6} = 16 \) \( \mu \)sec. Binary signaling (OQPSK) is used over the chips, yielding \( 2^{32} \) possible sequences over a 32 chip symbol. Of these sequences, 16 are selected to encode 4 bits \( (2^4 = 16) \). The sequences are selected so as to increase the probability of decoding in spite of symbol error. Thus, with 62.5 Kbps and 4 bits per symbol, the IEEE 802.15.4 PHY provides a raw bit rate of \( 62.5 \times 4 = 250 \) Kbps.

Having described the IEEE 802.15.4 PHY, we now turn to the MAC, i.e., the protocol for sharing the bit rate of an IEEE 802.15.4 shared digital link. A version of the CSMA/CA mechanism is used for multiple access. When a node has a data packet to send, it initiates a random back-off with the first back-off period being sampled uniformly from 0 to \( 2^{\text{macminBE}} - 1 \), where \( \text{macminBE} \) is a standard parameter. The back-off period is in slots, where a slot equals 20 symbol times, or \( 20 \times 16 = 320 \mu \)sec. The node then performs a Clear Channel Assessment (CCA) to determine whether the channel is idle. A CCA essentially involves the node listening over 8 symbols times, and integrating
its received power. If the result exceeds a threshold, it is concluded that the channel is busy and CCA fails. If the CCA succeeds, the node does a Rx-to-Tx turn-around, which takes 12 symbol times and starts transmitting on the channel. The failure of the CCA starts a new back-off process with the back-off exponent raised by one, i.e., to $\text{macminBE}+1$, provided it is less than the maximum back-off value, $\text{macmaxBE}$. The maximum number of successive CCA failures for the same packet is governed by $\text{macMaxCSMABackoffs}$; if this limit is exceeded the packet is discarded at the MAC layer. The standard allows the inclusion of acknowledgements (ACKs) which are sent by the intended receivers on a successful packet reception. Once the packet is received, the receiver performs a Rx-to-Tx turnaround, which is again 12 symbol times, and sends a 22-symbol fixed size ACK packet. A successful transmission is followed by an InterFrameSpace (IFS) before sending another packet.

When a transmitted packet collides or is corrupted by the PHY layer noise, the ACK packet is not generated which the transmitter interprets as packet delivery failure. The node reattempts the same packet for a maximum of $\text{aMaxFrameRetries}$ times before discarding it at the MAC layer. After transmitting a packet, the node turns to Rx-mode and waits for the ACK. The $\text{macAckWaitDuration}$ determines the maximum amount of time a node must wait to receive the ACK before declaring that the packet (or the ACK) has collided. The default values of $\text{macminBE}$, $\text{macmaxBE}$, $\text{macMaxCSMABackoffs}$, and $\text{aMaxFrameRetries}$ are 3, 5, 4, and 3.

18.3 Objectives of the Experiment

In Section 18.2, we saw that the IEEE 802.15.4 PHY provides a bit rate of 250 Kbps, which has to be shared among the nodes sharing the 2 MHz channel on which this network runs. In the simulation experiment, the packets will have an effective length of 109 bytes ($109 \text{ B} = 872 \text{ bits}$). Thus, over a 250 Kbps link, the maximum packet transmission rate is $\frac{250 \times 1000}{872} = 286.70$ packets per second. We notice, however, from the protocol description in Section 18.2, that due to the medium access control, before each packet is transmitted the nodes must contend for the transmission opportunity. This will reduce the actual packet transmission rate well below 286.7.

In this experiment, just one node will send packets to a receiver. Since there is no contention (there being only one transmitter) there is no need for medium access control, and packets could be sent back to back. However, the MAC protocol is always present, even with one node, and we would like to study the maximum possible rate at which a node can send back to back packets, when it is the only transmitter in the network. Evidently, since there is no uncertainty due to contention from other nodes, the overhead between the packets can be calculated from the protocol description in Section 18.2. This has been done in Section 18.6.

This analysis will provide the maximum possible rate at which a node can send packets over the IEEE 802.15.4 channel. Then in Section 18.7, we compare the throughput obtained from the
simulation with that obtained from the analysis. In the simulation, in order to ensure that the node sends at the maximum possible rate, the packet queue at the transmitting node never empties out. This is ensured by inserting packets into the transmitting node queue at a rate higher than the node can possibly transmit.

18.4 NetSim Simulation Setup

Open NetSim and click Examples > Experiments > One-Hop-IoT-Network-over-IEEE-802.15.4 as shown below:

Sample 1:

NetSim UI displays the configuration file corresponding to this experiment as shown below:
18.5 Simulation Procedure

The following set of procedures were done to generate this sample:

**Step 1:** A network scenario is designed in NetSim GUI comprising of 2 Wireless Sensors, a 6 LOWPAN Gateway, 1 Router, and 1 Wired Node.

**Step 2:** In the Interface Zigbee > Data Link Layer of Wireless Sensor 1, **Ack Request** is set to Enable and **Max Frame Retries** is set to 7.

It will automatically be set for Wireless Sensor 2, since the above parameters are Global.

**Step 3:** In the Interface Zigbee > Data Link Layer of 6 LOWPAN Gateway, **Beacon Mode** is set to Disable by default.

**Step 4:** The Ad hoc link properties are set to NO PATHLOSS for the channel characteristics.

**Step 5:** Right click on the Application Flow App1 CUSTOM and select Properties or click on the Application icon present in the top ribbon/toolbar.

A Customer Application is set from Wireless Sensor 1 i.e. Source to Wired Node 5 i.e. Destination. Transport Protocol is set to **UDP** with Packet Size set to 70 Bytes and Inter Arrival Time set to 4000 µs.

The Packet Size and Inter Arrival Time parameters are set such that the Generation Rate equals 140 Kbps. Generation Rate can be calculated using the formula:

\[
\text{Generation Rate (Mbps)} = \frac{\text{Packet Size (Bytes)}}{\text{Interarrival time (µs)}} \times 8
\]

**NOTE:** If the size of the packet at the Physical layer is greater than 127 bytes, the packet gets fragmented. Taking into account the various overheads added at different layers (which are mentioned below), the packet size at the application layer should be less than 80 bytes.

**Step 6:** Run simulation for 10 Seconds and note down the throughput.

Similarly, do the other samples by increasing the simulation time to 50, 100, and 200 Seconds respectively and note down the throughputs.

18.6 Analysis of Maximum Throughput

We have set the Application layer payload as 70 bytes in the Packet Size and when the packet reaches the Physical Layer, various other headers gets added like:

<table>
<thead>
<tr>
<th>Layer</th>
<th>Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>App layer Payload</td>
<td>70 bytes</td>
</tr>
<tr>
<td>Transport Layer Header</td>
<td>8 bytes</td>
</tr>
<tr>
<td>Network Layer Header</td>
<td>20 bytes</td>
</tr>
<tr>
<td>MAC Header</td>
<td>5 bytes</td>
</tr>
</tbody>
</table>
PHY Header (includes Preamble, and Start Packet Delimiter) 6 bytes
Packet Size 109 bytes

By default, NetSim uses Unslotted CSMA/CA and so, the packet transmission happens after a Random Back Off, CCA, and Turn-Around-Time and is followed by Turn-Around-Time and ACK Packet and each of them occupies specific time set by the IEEE 802.15.4 standard as per the timing diagram shown below:

From IEEE standard, each slot has 20 Symbols in it and each symbol takes 16µs for transmission.

<table>
<thead>
<tr>
<th>Symbol Time</th>
<th>16 µs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Slot Time</td>
<td>0.32 ms</td>
</tr>
<tr>
<td>Random Backoff Average</td>
<td>1.12 ms</td>
</tr>
<tr>
<td>CCA</td>
<td>0.128 ms</td>
</tr>
<tr>
<td>Turn-around-Time</td>
<td>0.192 ms</td>
</tr>
<tr>
<td>Packet Transmission Time</td>
<td>3.488 ms</td>
</tr>
<tr>
<td>Turn-around-Time</td>
<td>0.192 ms</td>
</tr>
<tr>
<td>ACK Packet Time</td>
<td>0.192 ms</td>
</tr>
<tr>
<td>Total Time</td>
<td>5.312 ms</td>
</tr>
</tbody>
</table>

\[
\text{Analytical Application Throughput} = \frac{70(\text{bytes in Applayer} \times 8)}{5.312 \text{ ms}} = 105.42 \text{ kbps}
\]

18.7 Comparison of Simulation and Calculation

<table>
<thead>
<tr>
<th>Throughput from simulation</th>
<th>104.74 kbps</th>
</tr>
</thead>
<tbody>
<tr>
<td>Throughput from analysis</td>
<td>105.42 kbps</td>
</tr>
</tbody>
</table>

Throughput from theoretical analysis matches the results of NetSim’s discrete event simulation. The slight difference in throughput is due two facts:

- The average of random numbers generated for backoff need not be exactly 3.5 as the simulation is run for short time.
In the packet trace one can notice that there are OSPF and AODV control packets (required for the route setup process) that sent over the network. The data transmissions occur only after the control packet transmissions are completed.

As we go on increasing the simulation time, the throughput value obtained from simulation approaches the theoretical value as can be seen from the table below:

<table>
<thead>
<tr>
<th>Sample</th>
<th>Simulation Time (sec)</th>
<th>Throughput (kbps)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>10</td>
<td>103.60</td>
</tr>
<tr>
<td>2</td>
<td>50</td>
<td>104.64</td>
</tr>
<tr>
<td>3</td>
<td>100</td>
<td>104.61</td>
</tr>
<tr>
<td>4</td>
<td>200</td>
<td>104.72</td>
</tr>
</tbody>
</table>
19. IoT – Multi-Hop Sensor-Sink Path

**NOTE:** It is recommended to carry out this experiment in Standard Version of NetSim.

19.1 Introduction

The Internet provides the communication infrastructure for connecting computers, computing devices, and people. The Internet is itself an interconnection of a very large number of interconnected packet networks, all using the same packet networking protocol. The Internet of Things will be an extension of the Internet with sub-networks that will serve to connect “things” among themselves and with the larger Internet. For example, a farmer can deploy moisture sensors around the farm so that irrigation can be done only when necessary, thereby resulting in substantial water savings. Measurements from the sensors have to be communicated to a computer in the Internet, where inference and decision-making algorithms can advise the farmer as to required irrigation actions.

Farms could be very large, from a few acres to hundreds of acres. If the communication is entirely wireless, a moisture sensor might have to communicate with a sink that is 100s of meters away. As the distance between a transmitter and a receiver increases, the power of the signal received at the receiver decreases, eventually making it difficult for the signal processing algorithms at the receiver to decode the transmitted bits in the presence of the ever-present thermal noise. Also, for a large farm there would need to be a large number of moisture sensors; many of them might transmit together, leading to collisions and interference.

19.2 Theory:

The problem of increasing distance between the transmitter and the receiver is solved by placing packet routers between the sensors and the sink. There could even be multiple routers on the path from the sensor to the sink, the routers being placed so that any intermediate link is short enough to permit reliable communication (at the available power levels). We say that there is a multi-hop path from a sensor to the sink.

By introducing routers, we observe that we have a system with sensors, routers, and a sink; in general, there could be multiple sinks interconnected on a separate edge network. We note here that a sensor, on the path from another sensor to the sink, can also serve the role of a router. Nodes whose sole purpose is to forward packets might also need to be deployed.

The problem of collision and interference between multiple transmission is solved by overlaying the systems of sensors, routers, and sinks with a scheduler which determines (preferably in a distributed manner) which transmitters should transmit their packets to which of their receivers.
In this experiment, we will use NetSim Simulator to study the motivation for the introduction of packet routers, and to understand the performance issues that arise. We will understand the answers to questions such as:

1. How does packet error rate degrade as the sensor-sink distance increases?
2. How far can a sensor be from a sink before a router needs to be introduced?
3. A router will help to keep the signal-to-noise ratio at the desired levels, but is there any adverse implication of introducing a router?

### 19.3 Network Setup:

Open NetSim and click **Examples > Experiments > IoT–Multi-Hop-Sensor-Sink-Path > Part-1 > Sample-1** as shown below:
19.4 Part 1 – Packet Delivery Rate vs. Distance

In this part, we perform a simulation to understand, “How the distance between the source and sink impacts the received signal strength (at the destination) and in turn the packet error rate?” We will assume a well-established path-loss model under which, as the distance varies, the received signal strength (in dBm) varies linearly. For a given transmit power (say 0dBm), at a certain reference distance (say 1m) the received power is $c_0$ dBm, and decreases beyond this point as $-10\eta \log_{10} d$ for a transmitter-receiver distance of $d$. This is called a power-law path loss model, since in mW the power decreases as the $\eta$ power of the distance $d$. The value of $\eta$ is 2 for free space path loss and varies from 2 to 5 in the case of outdoor or indoor propagation. Values of $\eta$ are obtained by carrying out experimental propagation studies.

Sample 1:

NetSim UI displays the configuration file corresponding to this experiment as shown below:

19.5 Procedure:

The following set of procedures were done to generate this sample:

Step 1: A network scenario is designed in the NetSim GUI comprising of a WSN Sink and 1 Wireless Sensor in Wireless Sensor Networks.

Note: NetSim currently supports a maximum of only one device as WSN Sink.

Step 2: Before we actually designed this network, in the Fast Config Window containing inputs for Grid Settings and Sensor Placement, the Grid Length and Side Length were set to 500 meters.
respectively, instead of the default 50 meters and we have chosen **Manually Via Click and Drop** option.

**Step 3:** The distance between the WSN Sink and Wireless Sensor is 5 meters.

**Step 4:** Go to Network Layer properties of Wireless Sensor 2, the Routing Protocol is set as **AODV**.

**Note:** The Routing Protocol parameter is Global. i.e. It will automatically be set to AODV in WSN Sink.

**Step 5:** In the Interface Zigbee > Data Link Layer of Wireless Sensor 2, **Ack Request** is set to Enable and **Max Frame Retries** is set to 4. Similarly, it is set for WSN Sink 1.

**Step 6:** In the Interface Zigbee > Physical Layer of Wireless Sensor 2, **Transmitter Power** is set to 1mW, **Reference Distance** is set to 1m, **Receiver Sensitivity** is set to -105dBm, and **ED Threshold** is set to -115dBm.

**Step 7:** Right click on the Application Flow App1 CUSTOM and select Properties or click on the Application icon present in the top ribbon/toolbar.

A CUSTOM Application is generated from Wireless Sensor 2 i.e. Source to WSN Sink 1 i.e. Destination with Transport Protocol set to UDP, Packet Size set to 70 Bytes and Inter Arrival Time set to 4000 µs.

The Packet Size and Inter Arrival Time parameters are set such that the Generation Rate equals 140 Kbps. Generation Rate can be calculated using the formula:

\[
\text{Generation Rate (Mbps) = Packet Size (Bytes) \times 8/Interarrival time (µs)}
\]

**Step 8:** The following procedures were followed to set Static IP:

Go to Network Layer properties of Wireless Sensor 2, Click on **Configure Static Route IP**.
Static IP Routing Dialogue box gets open.

Enter the Network Destination, Gateway, Subnet Mask, Metrics, and Interface ID. Click on Add.

You will find the entry added to the below Static IP Routing Table as shown below.

Click on OK.

Step 9: Packet Trace is enabled in NetSim GUI. At the end of the simulation, a very large .csv file is containing all the packet information is available for the users to perform packet level analysis.

Note: Before we click on Run simulation, user need to modify the code as per the “Procedure to log RSSI and BER” given below.

NOTE: The following changes need to be done manually by the user inorder to carry out this experiment.

Procedure to log RSSI and BER (Possible in Standard / Pro Versions only):

RSSI and BER in ZigBee project can be logged into a text file. The following code changes are required to log these parameters into a txt file.

- Go to NetSim Home page and click on Open Simulation.
- Click on Workspace Options and then click on Open Code and open the codes in Visual Studio. Set Win32 or x64 according to the NetSim build which you are using.

Note: We recommend Visual Studio Community Edition 2017 or Higher.
- Go to the Zigbee Project in the Solution Explorer. Open 802_15_4.c file and add the following lines of code highlighted in red, inside the `fn_NetSim_Zigbee_init()` function as shown below:

```c
_declspec (dllexport) int fn_NetSim_Zigbee_Init(struct stru_NetSim_Network
*NETWORK_Formal, \\
NetSim_EVENTDETAILS *pstruEventDetails_Formal, char *pszAppPath_Formal, \\
char *pszWritePath_Formal, int nVersion_Type, void **fnPointer)
{
    FILE* fp;

    pstruEventDetails = pstruEventDetails_Formal;
    NETWORK = NETWORK_Formal;
    pszAppPath = pszAppPath_Formal;
    pszIOPath = pszWritePath_Formal;

    //RSSI BER SNR LOG
    fp = fopen("ZIGBEE_BER_LOG.txt", "w+");
    if (fp)
    {
        fprintf(fp,
            "PACKET_ID, \tTRANSMITTER, \tRECEIVER, \tRX_POWER(dBm), \tTOTAL_RX_POWER(dBm), \tBER\n";
        fclose(fp);
    }

    //RSSI BER SNR LOG

    fn_NetSim_Zigbee_Init_F(NETWORK_Formal, pstruEventDetails_Formal,
    pszAppPath_Formal, \\
    pszWritePath_Formal, nVersion_Type, fnPointer);
    return 0;
}
```

- Add the lines of code highlighted in red inside the `fn_NetSim_Zigbee_Run()` function under `PHYSICAL_IN_EVENT` as shown below:

```c
case PHYSICAL_IN_EVENT:
{
```
NetSim_PACKET *pstruPacket;
PACKET_STATUS nPacketStatus;
double SNR;
double dBER;
FILE* fp;

pstruPacket = pstruEventDetails->pPacket;
if (pstruPacket->nReceiverId && pstruPacket->nReceiverId != pstruEventDetails->nDeviceId)
{
    fnNetSimError("Different device packet received..");
    assert(false);
    return 0;
}

if (!ZIGBEE_CHANGERADIOSTATE(pstruEventDetails->nDeviceId,
    WSN_PHY(pstruEventDetails->nDeviceId)->nRadioState, RX_ON_IDLE))
return 0;

if (WSN_PHY(pstruEventDetails->nDeviceId)->dTotalReceivedPower -
    GET_RX_POWER_mw(pstruPacket->nTransmitterId, pstruPacket->nReceiverId,
    pstruEventDetails->dEventTime) >= WSN_PHY(pstruEventDetails->nDeviceId)->dReceiverSensitivity)
pstruPacket->nPacketStatus = PacketStatus_Collided;
nPacketStatus = pstruPacket->nPacketStatus;
ZIGBEE_SINR(&SNR,
    WSN_PHY(pstruEventDetails->nDeviceId)->dTotalReceivedPower,
    GET_RX_POWER_mw(pstruPacket->nTransmitterId, pstruPacket->nReceiverId,
    pstruEventDetails->dEventTime));

dBER = fn_NetSim_Zigbee_CalculateBER(SNR);

//RSSI BER SNR LOG
double rxpwr = MW_TO_DBM(WSN_PHY(pstruEventDetails->nDeviceId)-
    >dTotalReceivedPower);
double total_rxpwr = GET_RX_POWER_dbm(pstruPacket->nTransmitterId,
    pstruPacket->nReceiverId, pstruEventDetails->dEventTime);
fp = fopen("ZIGBEE_BER_LOG.txt", "a+");  
if (fp)  
{  
fprintf(fp, "\n%lld,%s,%s,%lf,%lf,%lf", pstruPacket->nPacketId,  
DEVICE_NAME(pstruPacket->nTransmitterId),  
DEVICE_NAME(pstruPacket->nReceiverId),  
rxpwr, total_rxpwr, dBER);  
fclose(fp);  
}  
//RSSI BER SNR LOG

if (fn_NetSim_Packet_DecideError(dBER, pstruEventDetails->dPacketSize))

- Right click on the ZigBee project in the solution explorer and click on rebuild.
- After the Zigbee project is rebuild successful, go back to the network scenario.

**Step 10:** Run the simulation for 10 Seconds. Once the simulation is complete, it will generate a text file named ZIGBEE_BER_LOG.txt containing RSSI and BER in the binary folder of NetSim. i.e. <NetSim Install Directory>/bin.

### 19.6 Output:

<table>
<thead>
<tr>
<th>Distance(m)</th>
<th>RSSI (dBm) (pathloss model)</th>
<th>BER</th>
<th>PER</th>
<th>PLR (after MAC retransmissions*)</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>-64.51</td>
<td>0.00</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>10</td>
<td>-75.04</td>
<td>0.00</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>15</td>
<td>-81.20</td>
<td>0.00</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>20</td>
<td>-85.58</td>
<td>0.00</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>25</td>
<td>-88.97</td>
<td>0.00</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>30</td>
<td>-91.74</td>
<td>0.00</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>35</td>
<td>-94.08</td>
<td>0.000005</td>
<td>0.0051</td>
<td>0</td>
</tr>
<tr>
<td>40</td>
<td>-96.11</td>
<td>0.000229</td>
<td>0.2076</td>
<td>0</td>
</tr>
<tr>
<td>45</td>
<td>-97.90</td>
<td>0.002175</td>
<td>0.8905</td>
<td>0.447</td>
</tr>
<tr>
<td>50</td>
<td>-99.51</td>
<td>0.008861</td>
<td>0.9999</td>
<td>1</td>
</tr>
<tr>
<td>55</td>
<td>-100.95</td>
<td>0.022370</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>60</td>
<td>-102.28</td>
<td>0.042390</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>65</td>
<td>-103.49</td>
<td>0.067026</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
The IEEE 802.15.4 MAC implements a retransmission scheme that attempts to recover errored packets by retransmission. If all the retransmission attempts are also errored, the packet is lost.

The table above reports the RSSI (Received Signal Strength), BER (Bit Error Rate), and Packet Error Rate (PER), and the Packet Loss Rate (PLR) as the distance between the sensor to the sink is increased from 5m to 50m with path loss exponent $\eta = 3.5$. We see that the BER is 0 until a received power of about $-92$dBm. At a distance of 35m the received power is $-94$ dBm, and we notice a small BER of $5 \times 10^{-6}$. As the distance is increased further the BER continues to grow and at 45m the BER is about 0.002175, yielding $PER = 0.89$, and $PLR = 0.44$. Here $PER$ is obtained from the following formula (which assumes independent bit errors across a packet)

$$PER = 1 - (1 - BER)^{PL},$$

Where,

$$PL \text{ – packet length in bits at the PHY layer}$$

$$PL \text{ (bits)} = (70 \text{ (payload)} + 57 \text{ (overhead)}) \times 8$$

The $PLR$ in the above table has been obtained from NetSim, which implements the details of the IEEE 802.15.4 MAC acknowledgement and reattempt mechanism. This mechanism is complex, involving a MAC acknowledgement, time-outs, and multiple reattempts. Analysis of the $PLR$, therefore, is not straightforward. Assuming that the probability of MAC acknowledgement error is small (since it is a small packet), the $PLR$ can be approximated as $PER^{K+1}$, where $K$ is the maximum number of times a packet can be retransmitted.

$$PLR = \frac{\text{Total number of Lost Packet}}{\text{Total number of Packet Sent by Source MAC}}$$

Total number of Lost packets

$$= \text{Total number of Packet Sent by Source MAC} - \text{Packets Received at Destination MAC}$$

Steps to calculate Packet Loss Rate:

- Open Packet Trace from the Results Dashboard. Filter the PACKET TYPE column as Custom and note down the packet id of the last packet sent from the PACKET ID column.
This represents the total number of packets sent by the source.

- Note down the Packets Received from the Application Metrics in the Results Dashboard.

This represents the total number of packets received at the destination.

- Calculate the total number of Lost Packets and PLR as follows:

For the above case,

\[
\text{Total number of Packet Sent by Source MAC} = 463 \\
\text{Packets Received at Destination MAC} = 256 \\
\text{Total number of Lost packets} = 463 - 256 = 207 \\
\text{PLR} = \frac{207}{463} = 0.447
\]

19.7 Inference:

It is clear that Internet applications, such as banking and reliable file transfer, require that all the transmitted data is received with 100% accuracy. The Internet achieves this, in spite of unreliable communication media (no medium is 100% reliable) by various protocols above the network layer. Many IoT applications, however, can work with less than 100% packet delivery without affecting the application. Take, for example, the farm moisture sensing application mentioned in the introduction. The moisture levels vary slowly; if one measurement is lost, the next received measurement might
suffice for the decision-making algorithm. This sort of thinking also permits the IoT applications to utilize cheap, low power devices, making the IoT concept practical and affordable.

With the above discussion in mind, let us say that the application under consideration requires a measurement delivery rate of at least 80%. Examining the table above, we conclude that the sensor-sink distance must not be more than 40 meters. Thus, even a 1 acre farm ($61m \times 61m$) would require multi-hopping to connect sensors to a sink at the edge of the farm.

In Part 2 of this experiment we will study the placement of a single router between the sensor and the sink, so as to increase the sensor-sink distance beyond 40 meters.

### 19.8 Part 2 – Reaching a Longer Distance by Multihopping

**Sample 1:**

NetSim UI displays the configuration file corresponding to this experiment as shown below:

![NetSim UI configuration file](image)

### 19.9 Procedure:

The following changes in settings are done from the previous sample:

**Step 1:** The distance between the WSN Sink and Wireless Sensor is 40 meters.

**Step 2:** In the Interface Zigbee > Data Link Layer of Wireless Sensor 2, **Ack Request** is set to Enable and **Max Frame Retries** is set to 3.

**Step 3:** The **Ad hoc Link** properties are set as follows:
Step 4: Right click on the Application Flow App1 CUSTOM and select Properties or click on the Application icon present in the top ribbon/toolbar.

A CUSTOM Application is generated from Wireless Sensor 2 i.e. Source to WSN Sink 1 i.e. Destination with Packet Size set to 70 Bytes and Inter Arrival Time set to 100000 µs.

The Packet Size and Inter Arrival Time parameters are set such that the Generation Rate equals 5.6 Kbps. Generation Rate can be calculated using the formula:

\[
\text{Generation Rate (Mbps)} = \frac{\text{Packet Size (Bytes)} \times 8}{\text{Interarrival time (µs)}}
\]

Step 5: Run the Simulation for 100 Seconds. Once the simulation is complete, note down the Packet Generated value and Throughput value from the Application Metrics.

Note down the Packet Received, Packet Errored, and Packet Collided from the Link Metrics.

Sample-2:

NetSim UI displays the configuration file corresponding to this experiment as shown below:
19.10 Procedure:

The following changes in settings are done from the previous sample:

**Step 1:** One more Wireless Sensor is added to this network. The distance between Wireless Sensor 2 and Wireless Sensor 3 is 40 meters and the distance between Wireless Sensor 3 and the WSN Sink is 40 meters.

**Step 2:** The following procedures were followed to set Static IP:

Go to Network Layer properties of Wireless Sensor 2, Click on **Configure Static Route IP**.

Static IP Routing Dialogue box gets open.

Enter the Network Destination, Gateway, Subnet Mask, Metrics, and Interface ID. Click on **Add**.

You will find the entry added to the below Static IP Routing Table as shown below:

Click on **OK**.
Similarly, Static IP is set for Wireless Sensor 3 as shown below:

![Static IP Routing Dialogue](image1)

**Step 3:** Run the Simulation for 100 Seconds. Once the simulation is complete, note down the Packet Generated value and Throughput value from the **Application Metrics.** Note down the Packet Received, Packet Errored, and Packet Collided from the **Link Metrics.**

**19.11 Output:**

<table>
<thead>
<tr>
<th>Source-Sink Distance (m)</th>
<th>Packets Generated</th>
<th>Packets Received</th>
<th>Packets Errored (PHY)</th>
<th>Packets Collided</th>
<th>Packet Loss (MAC)</th>
<th>PLR</th>
<th>Mean Delay (μs)</th>
</tr>
</thead>
</table>
NOTE: Packet loss (PHY) is the number of packets that were received in error and then recovered by retransmission. Packets received is slightly higher than packets generated on account of retransmissions of successful packets in case of ACK errors.

19.12 Inference:

In Part 1 of this experiment we learnt that if the sensor device uses a transmit power of 0dBm, then for one-hop communication to the sink, the sensor-sink distance cannot exceed 40m. If the sensor-sink distance needs to exceed 40m (see the example discussed earlier), there are two options:

1. The transmit power can be increased. There is, however, a maximum transmit power for a given device. Wireless transceivers based on the CC 2420 have a maximum power of 0dBm (i.e., about 1 mW), whereas the CC 2520 IEEE 802.15.4 transceiver provides maximum transmit power of 5dBm (i.e., about 3 mW). Thus, given that there is always a maximum transmit power, there will always be a limit on the maximum sensor-sink distance.

2. Routers can be introduced between the sensor and the sink, so that packets from the sensor to the sink follow a multihop path. A router is a device that will have the same transceiver as a sensor but its microcontroller will run a program that will allow it to forward packets that it receives. Note that a sensor device can also be programmed to serve as a router. Thus, in IOT networks, sensor devices themselves serve as routers.

In this part of the experiment we study the option of introducing a router between a sensor and the sink to increase the sensor-sink distance. We will compare the performance of two networks, one with the sensor communicating with a sink at the distance of 40m, and another with the sensor-sink distance being 80m, with a sensor at the mid-point between the sensor and the sink.

Part 2, Sample 1 simulates a one hop network with a sensor-sink distance of 40m. We recall from Part 1 that, with the transceiver model implemented in NetSim, 40m is the longest one hop distance possible for 100% packet delivery rate. In sample 2, To study the usefulness of routing we will set up network with a sensor-sink distance of 80m with a packet router at the midpoint between the sensor and the sink.

The measurement process at the sensor is such that one measurement (i.e., one packet) is generated every 100ms. The objective is to deliver these measurements to the sink with 100%
delivery probability. From Part 1 of this experiment we know that a single hop of 80m will not provide the desired packet delivery performance.

The Table at the beginning of this section shows the results. We see that both networks are able to provide a packet delivery probability of 100%. It is clear, however, that since the second network has two hops, each packet needs to be transmitted twice, hence the mean delay between a measurement being generated and it being received at the sink is doubled. Thus, the longer sensor-sink distance is being achieved, for the same delivery rate, at an increased delivery delay.

The following points may be noted from the table:

1. The number of packets lost due to PHY errors. The packet delivery rate is 100% despite these losses since the MAC layer re-transmission mechanism is able to recover all lost packets.

2. There are no collisions. Since both the links (sensor-router and router-sink) use the same channel and there is no co-ordination between them, it is possible, in general for sensor-router and router-sink transmissions to collide. This is probable when the measurement rate is large, leading to simultaneously nonempty queues at the sensor and router. In this experiment we kept the measurement rate small such that the sensor queue is empty when the router is transmitting and vice versa. This avoids any collisions.
20. Study how call blocking probability varies as the load on a GSM network is continuously increased

20.1 Network Setup:

Open NetSim and click **Examples > Experiments > Impact-of-load-on-call-blocking-probability-in-GSM** as shown below:

![NetSim UI displaying configuration file](image)

NetSim UI displays the configuration file corresponding to this experiment as shown below:
**20.2 Procedure:**

The following set of procedures were done to generate this sample:

**Step 1:** A network scenario is designed in NetSim GUI comprising of 4 Mobile Stations, 1 MSC, and 1 Base Station in the “Cellular Networks” Network Library.

**Step 2:** Ensure all the Mobile Stations are placed within the range of Base Station.

**Step 3:** In the Interface GSM > Data Link Layer Properties of MSC 2, Uplink BW Min and Uplink BW Max are set to 890 MHz and 890.2 MHz respectively.

**Step 4:** Right click on the Application Flow App1 ERLANG CALL and select Properties or click on the Application icon present in the top ribbon/toolbar.

The applications are set as per the below table:

<table>
<thead>
<tr>
<th>Application</th>
<th>Properties</th>
<th>Application 1</th>
<th>Application 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Application type</td>
<td>Erlang_call</td>
<td>Erlang_call</td>
<td></td>
</tr>
<tr>
<td>Source_Id</td>
<td>3</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>Destination_Id</td>
<td>4</td>
<td>6</td>
<td></td>
</tr>
<tr>
<td>Call</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Duration_Distribution</td>
<td>Exponential</td>
<td>Exponential</td>
<td></td>
</tr>
<tr>
<td>Duration(s)</td>
<td>60</td>
<td>60</td>
<td></td>
</tr>
<tr>
<td>Inter_Arrival_Time (sec)</td>
<td>10</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>IAT_Distribution</td>
<td>Exponential</td>
<td>Exponential</td>
<td></td>
</tr>
<tr>
<td>Codec</td>
<td>Custom</td>
<td>Custom</td>
<td></td>
</tr>
<tr>
<td>Inter_Arrival_Time_distribution</td>
<td>Constant</td>
<td>Constant</td>
<td></td>
</tr>
<tr>
<td>Packet_Distribution</td>
<td>Constant</td>
<td>Constant</td>
<td></td>
</tr>
</tbody>
</table>
### Step 5: Run the Simulation for 100 Seconds.

The following changes in settings are done from the previous sample:

**Step 1:** In the next sample, increase the number of Mobile Stations by 2 and add one more application between them.

**Step 2:** Run the Simulation for 100 Seconds.

The following changes in settings are done from the previous sample:

**Step 1:** Similarly, increase the number of Mobile Stations by 2 up to 20 and set properties for different Samples by adding an application every time and changing Source ID and Destination ID.

**Step 2:** Run the Simulation for 100 Seconds.

### 20.3 Output

To view the output, go to the Cellular Metrics. In MS metrics, take sum of call blocking probability (It is the as ratio of Total call blocked to Total call generated).

**Comparison Charts:**

![Call Blocking Probability Chart](chart.png)

*** All the above plots highly depend upon the placement of Mobile station in the simulation environment. So, note that even if the placement is slightly different the same set of values will not be got but one would notice a similar trend.
20.4 Inference:

When the number of MS is increased from 4 to 20 the call blocking probability increases from 0 to 3.46. As we increase the number of mobile stations more calls are generated. This increases the traffic load on the system & more calls generated implies more channel requests arrive at the base station but the number of channels is fixed. So when the base station does not find any free channel the call is blocked. An additional observation is that the call blocking is zero until 8 MS. This is because the number of channels is sufficient to handle all call that 6 MS may generate. Only after this the base station does not find free channels and blocks calls.
21. Study the 802.15.4 Superframe Structure and analyze the effect of Superframe order on throughput

21.1 Introduction:

A coordinator in a PAN can optionally bound its channel time using a Superframe structure which is bound by beacon frames and can have an active portion and an inactive portion. The coordinator enters a low-power (sleep) mode during the inactive portion.

The structure of this Superframe is described by the values of macBeaconOrder and macSuperframeOrder. The MAC PIB attribute macBeaconOrder, describes the interval at which the coordinator shall transmit its beacon frames. The value of macBeaconOrder, BO, and the beacon interval, BI, are related as follows:

For $0 \leq BO \leq 14$, $BI = a_{BaseSuperframeDuration} \times 2^{BO}$ symbols.

If $BO = 15$, the coordinator shall not transmit beacon frames except when requested to do so, such as on receipt of a beacon request command. The value of macSuperframeOrder, SO shall be ignored if $BO = 15$.

An example of a Superframe structure is shown in following Figure.

![Fig: An example of the Super Frame structure](image)

**Theoretical Analysis:**

From the above Superframe structure,
\[
\text{SuperFrame Duration} = a\text{BaseSuperframeDuration} \times 2^{BO}
\]
\[
\text{Active part of SuperFrame} = a\text{BaseSuperframeDuration} \times 2^{SO}
\]
\[
\text{Inactive part of SuperFrame} = a\text{BaseSuperframeDuration} \times (2^{BO} - 2^{SO})
\]

If Superframe Order (SO) is same as Beacon Order (BO) then there will be no inactive period and the entire Superframe can be used for packet transmissions.

If BO=10, SO=9 half of the Superframe is inactive and so only half of Superframe duration is available for packet transmission. If BO=10, SO=8 then (3/4)\text{th} of the Superframe is inactive and so nodes have only (1/4)\text{th} of the Superframe time for transmitting packets and so we expect throughput to approximately drop by half of the throughput obtained when SO=9.

Percentage of inactive and active periods in Superframe for different Superframe Orders is given below:

<table>
<thead>
<tr>
<th>Beacon Order (BO)</th>
<th>Super Frame Order (SO)</th>
<th>Active part of Superframe(%)</th>
<th>Inactive part of Superframe (%)</th>
<th>Throughput estimated (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>10</td>
<td>100</td>
<td>0</td>
<td>&gt; 200% of T</td>
</tr>
<tr>
<td>10</td>
<td>9</td>
<td>50</td>
<td>50</td>
<td>Say T = 21.07 (Got from simulation)</td>
</tr>
<tr>
<td>10</td>
<td>8</td>
<td>25</td>
<td>75</td>
<td>50 % T</td>
</tr>
<tr>
<td>10</td>
<td>6</td>
<td>12.5</td>
<td>87.5</td>
<td>25 % T</td>
</tr>
<tr>
<td>10</td>
<td>5</td>
<td>3.125</td>
<td>96.875</td>
<td>12.5 % of T</td>
</tr>
<tr>
<td>10</td>
<td>4</td>
<td>1.5625</td>
<td>98.4375</td>
<td>6.25 % of T</td>
</tr>
<tr>
<td>10</td>
<td>3</td>
<td>0.78125</td>
<td>99.21875</td>
<td>3.12% of T</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1.56 % of T</td>
</tr>
</tbody>
</table>

We expect throughput to vary in the active part of the Superframe as sensors can transmit a packet only in the active portion.

### 21.2 Network Setup:

Open NetSim and click Examples > Experiments > 802.15.4-Superframe-and-effect-of-Superframe-order-on-throughput as shown below:
Sample 1:

NetSim UI displays the configuration file corresponding to this experiment as shown below:

![NetSim UI Configuration](image)

21.3 Procedure:

The following set of procedures were done to generate this sample:

**Step 1:** A network scenario is designed in NetSim GUI comprising of 2 Wireless Sensors and a WSN Sink in the “Wireless Sensor Networks” Network Library.

**Step 2:** Before we actually designed this network, in the Fast Config Window containing inputs for Grid Settings and Sensor Placement, the Grid Length and Side Length were set to 500 and 250 meters respectively, instead of the default 100 and 50 meters and we have chosen Manually Via Click and Drop option.
Step 3: The Ad hoc Link is used to link the Sensors and the Gateway in an ad hoc basis.

The Ad hoc link properties is set to NO PATHLOSS for the channel characteristics.

Step 4: In the Interface Zigbee > Data Link Layer of WSN Sink, Beacon Mode is set to Enable and Beacon Order and Super Frame Order is set to 10 respectively.

Step 5: Right click on the Application Flow App1 CUSTOM and select Properties or click on the Application icon present in the top ribbon/toolbar.

A CUSTOM Application is generated from Wireless Sensor 1 i.e. Source to Wireless Sensor 2 i.e. Destination with Packet Size set to 25 Bytes and Inter Arrival Time set to 3000 µs.

The Packet Size and Inter Arrival Time parameters are set such that the Generation Rate equals 67 Kbps. Generation Rate can be calculated using the formula:

\[
\text{Generation Rate (Mbps)} = \frac{\text{Packet Size (Bytes)} \times 8}{\text{Interarrival time (µs)}}
\]

Step 6: Run the Simulation for 30 Seconds and note down the Throughput value.

Similarly, run the other samples by varying the Super Frame Order to 9, 8, 7, 6, 5, and 4 and note down the throughput values.

21.4 Output:

The following are the throughputs obtained from the simulation for different Super Frame Orders.

<table>
<thead>
<tr>
<th>Super Frame Order</th>
<th>Throughput (Kbps)</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>41.63</td>
</tr>
<tr>
<td>9</td>
<td>21.07</td>
</tr>
<tr>
<td>8</td>
<td>10.5</td>
</tr>
<tr>
<td>7</td>
<td>5.25</td>
</tr>
<tr>
<td>6</td>
<td>2.63</td>
</tr>
<tr>
<td>5</td>
<td>1.30</td>
</tr>
<tr>
<td>4</td>
<td>0.62</td>
</tr>
</tbody>
</table>

To obtain throughput from simulation, payload transmitted values will be obtained from Link metrics and calculated using following formula:

\[
\text{Application Throughput (in Mbps)} = \frac{\text{Total payload delivered to destination (bytes) \times 8}}{\text{Simulation Time (Millisecond)} - \text{App Start Time (Millisecond)}}
\]
**Comparison Chart:** All the above plots highly depend upon the placement of Sensor in the simulation environment. So, note that even if the placement is slightly different the same set of values will not be got but one would notice a similar trend.

**21.5 Inference:**

From the comparison chart both the simulation and theoretical throughputs match except for the case with no inactive period. A sensor will be idle if the last packet in its queue is transmitted. If a packet is generated in inactive period then the packet has to wait in the queue till the next Superframe so sensor has packets waiting in its queue and so it cannot be idle in the next Superframe, but if there is no inactive period then there might be no packets waiting in the queue and so sensor can be idle resulting in lesser throughput.
22. Understand the working of OSPF

22.1 Objective
To understand the working of OSPF and Shortest Path First (SPF) tree creation.

22.2 Theory

**OSPF:**
Open Shortest Path First (OSPF) is an Interior Gateway Protocol (IGP) standardized by the Internet Engineering Task Force (IETF) and commonly used in large Enterprise networks. OSPF is a link-state routing protocol providing fast convergence and excellent scalability. Like all link-state protocols, OSPF is very efficient in its use of network bandwidth.

**Shortest path First Algorithm:**
OSPF uses a shortest path first algorithm in order to build and calculate the shortest path to all known destinations. The shortest path is calculated with the use of the Dijkstra algorithm. The algorithm by itself is quite complicated. This is a very high level, simplified way of looking at the various steps of the algorithm:

- Upon initialization or due to any change in routing information, a router generates a link-state advertisement. This advertisement represents the collection of all link-states on that router.
- All routers exchange link-states by means of flooding. Each router that receives a link-state update should store a copy in its link-state database and then propagate the update to other routers.
- After the database of each router is completed, the router calculates a Shortest Path Tree to all destinations. The router uses the Dijkstra algorithm in order to calculate the shortest path tree. The destinations, the associated cost and the next hop to reach those destinations form the IP routing table.

- In case no changes in the OSPF network occur, such as cost of a link or a network being added or deleted, OSPF should be very quiet. Any changes that occur are communicated through link-state packets, and the Dijkstra algorithm is recalculated in order to find the shortest path.

The algorithm places each router at the root of a tree and calculates the shortest path to each destination based on the cumulative cost required to reach that destination. Each router will have
its own view of the topology even though all the routers will build a shortest path tree using the same link-state database.

**Example:**

Refer Pg. no.18 from OSPF RFC 2328 ([https://tools.ietf.org/html/rfc2328#section-2.3](https://tools.ietf.org/html/rfc2328#section-2.3))

The below network shows a sample map of an Autonomous System

![Sample Autonomous System Diagram](image)

**Fig 1. Sample Autonomous system**

A cost is associated with the output side of each router interface. This cost is configurable by the system administrator. The lower the cost, the more likely the interface is to be used to forward data traffic. Costs are also associated with the externally derived routing data (e.g., the BGP-learned routes).

The directed graph resulting from the above network is depicted in the following table. Arcs are labelled with the cost of the corresponding router output interface. Arcs having no labelled cost have a cost of 0. Note that arcs leading from networks to routers always have cost 0.
A router generates its routing table from the above directed graph by calculating a tree of shortest paths with the router itself as root. Obviously, the shortest-path tree depends on the router doing the calculation. The shortest-path tree for Router RT6 in our example is depicted in the following figure.

**Table 1 Directed graph**
Routing Table

The tree gives the entire path to any destination network or host. However, only the next hop to the destination is used in the forwarding process. Note also that the best route to any router has also been calculated. For the processing of external data, we note the next hop and distance to any router advertising external routes. The resulting routing table for Router RT6 is shown in the following table

<table>
<thead>
<tr>
<th>Destination</th>
<th>Next hop</th>
<th>Distance</th>
</tr>
</thead>
<tbody>
<tr>
<td>N1</td>
<td>RT3</td>
<td>10</td>
</tr>
<tr>
<td>N2</td>
<td>RT3</td>
<td>10</td>
</tr>
<tr>
<td>N3</td>
<td>RT3</td>
<td>7</td>
</tr>
<tr>
<td>N4</td>
<td>RT3</td>
<td>8</td>
</tr>
<tr>
<td>N6</td>
<td>RT10</td>
<td>8</td>
</tr>
<tr>
<td>N7</td>
<td>RT10</td>
<td>12</td>
</tr>
<tr>
<td>N8</td>
<td>RT10</td>
<td>10</td>
</tr>
<tr>
<td>N9</td>
<td>RT10</td>
<td>11</td>
</tr>
<tr>
<td>N10</td>
<td>RT10</td>
<td>13</td>
</tr>
<tr>
<td>N11</td>
<td>RT10</td>
<td>14</td>
</tr>
<tr>
<td>H1</td>
<td>RT10</td>
<td>21</td>
</tr>
<tr>
<td>RT5</td>
<td>RT5</td>
<td>6</td>
</tr>
<tr>
<td>RT7</td>
<td>RT10</td>
<td>8</td>
</tr>
<tr>
<td>N12</td>
<td>RT10</td>
<td>10</td>
</tr>
<tr>
<td>N13</td>
<td>RT5</td>
<td>14</td>
</tr>
<tr>
<td>N14</td>
<td>RT5</td>
<td>14</td>
</tr>
<tr>
<td>N15</td>
<td>RT10</td>
<td>17</td>
</tr>
</tbody>
</table>
Routing Table for RT6

Distance calculation:

Router6 has 3 interfaces i.e. RT3, RT5 and RT10. The distance obtained is 10 for destination N1 via RT3 interface. The packets from Router6 would reach N1 via RT3, N3 and RT1. The cost assigned to routers in this path is 6+1+3 = 10 (cost can be seen in SPF tree for Router6). This is how distance is calculated.

22.3 Network Setup:

Open NetSim and click on Examples > Experiments > Understand-the-working-of-OSPF > Sample-1 as shown below:

NetSim UI displays the configuration file corresponding to this experiment as shown below:
The above network was created in NetSim and it is similar to the network as per the OSPF RFC 2328 (Refer Pg. no. 19 - https://tools.ietf.org/html/rfc2328#page-23)

### 22.4 Procedure:

The following set of procedures were done to generate this sample:

**Step 1:** A network scenario is designed in NetSim GUI comprising of 3 Wired Nodes and 27 Routers in the “Internetworks” Network Library.

**Step 2:** The Output Cost for all the Routers in the network is set as per the network shown in Figure 1.
Step 3: Packet Trace is enabled in the NetSim GUI, and hence we are able to track the route which the packets have chosen to reach the destination based on the Output Cost that is set.

Step 4: Right click on the Application Flow App1 CBR and select Properties or click on the Application icon present in the top ribbon/toolbar. A CBR Application is generated from Wired Node 30 i.e. Source to Wired Node 27 i.e. Destination with Packet Size remaining 1460Bytes and Inter Arrival Time remaining 20000µs. Additionally, the “Start Time(s)” parameter is set to 30, while configuring the application. This time is usually set to be greater than the time taken for OSPF Convergence (i.e. Exchange of OSPF information between all the routers), and it increases as the size of the network increases.

22.5 Output:
The following is the shortest path first tree created in NetSim:
In the above screenshot, red color information represents the interface ip addresses of routers and the blue color represents the cost.

**NOTE:** NetSim, does not implement Link type3 (Link to Stub Network). Hence users would notice a slight difference between the SPF trees of RFC and NetSim.

The IP forwarding table formed in the routers can be accessed from the IP_Forwarding_Table list present in the Simulation Results window as shown below:
In this network, Router6 has 3 interfaces with IP’s 11.7.1.1, 11.6.1.2 and 11.17.1.1 and its network addresses are 11.7.0.0, 11.6.0.0 and 11.17.0.0 since its network mask is 255.255.0.0

From the above screenshot, the router forwards packets intended to the subnet:

- 11.1.1.2, 11.2.1.2, 11.3.1.2, 11.3.1.1, 11.4.1.1 via interface 11.7.1.1 with cost 7 (6+1)
  
- Similarly 11.23.1.1, 11.4.1.2, 11.1.1.1, 11.2.1.1, 11.23.1.2 via interface 11.7.1.1 with cost 8 (6+1+1)
  
- 11.15.1.1, 11.9.1.2, 11.10.1.1, 11.15.1.2 via interface 11.17.1.1 with cost 8 (7+1)
  
- 11.9.1.1, 11.10.1.2 via interface 11.17.1.1 with cost 9 (7+1+1)
  
- 11.29.1.2, 11.29.1.1 and 11.14.1.2 via interface 11.17.1.1 with cost 10 (7+3)
• 11.24.1.2, 11.24.1.1 via interface 11.17.1.1 with cost 10 (7+1+2)
• 11.18.1.2, 11.18.1.1, 11.19.1.2 and 11.19.1.1 via interface 11.7.1.1 with cost 10 (6+1+3)
• 11.13.1.2, 11.11.1.1, 11.12.1.1, and 11.13.1.1 via interface 11.17.1.1 with cost 11 (7+3+1)
• 11.8.1.1 via interface 11.6.1.2 with cost 12 (6+6)
• 11.11.1.2 and 11.12.1.2 via interface 11.17.1.1 with cost 12 (7+3+1+1)
• 11.17.1.2 via interface 11.17.1.1 with cost 12 (7+5)
• 11.26.1.1 and 11.26.1.2 via interface 11.17.1.1 with cost 12 (7+1+4)
• 11.14.1.1 via interface 11.17.1.1 with cost 12 (7+3+2)
• 11.6.1.1 via interface 11.6.1.2 with cost 13 (7+6)
• 11.27.1.1 and 11.27.1.2 via interface 11.17.1.1 with cost 13 (7+3+1+2)
• 11.7.1.2 via interface 11.7.1.1 with cost 14 (8+6)
• 11.5.1.2 via interface 11.6.1.2 with cost 14 (6+8)
• 11.20.1.2, 11.20.1.1, 11.21.1.1, 11.21.1.2, 11.22.1.1, 11.22.1.2 via interface 11.6.1.2 with cost 14 (8+6)
• 11.28.1.2 via interface 11.17.1.1 with cost 14 (7+1+6)
• 11.28.1.1 via interface 11.17.1.1 with cost 14 (7+3+1+3)
• 11.25.1.1, 11.25.1.2 via interface 11.17.1.1 with cost 17 (7+1+9)
• 11.5.1.1 via interface 11.7.1.1 with cost 15 (6+1+8)

We are thus able to simulate the exact example as provided in the RFC and report that SPF Tree obtained and the routing costs match the analysis provided in the RFC
23. Understand the working of basic networking commands (Ping, Route Add/Delete/Print, ACL)

23.1 Theory:

NetSim allows users to interact with the simulation at runtime via a socket or through a file. User Interactions make simulation more realistic by allowing command execution to view/modify certain device parameters during runtime.

**Ping Command**

- The ping command is one of the most often used networking utilities for troubleshooting network problems
- You can use the ping command to test the availability of a networking device (usually a computer) on a network
- When you ping a device, you send that device a short message, which it then sends back (the echo)
- If you receive a reply then the device is in the Network, if you don’t, then the device is faulty, disconnected, switched off, or incorrectly configured.

**Route Commands**

You can use the route commands to view, add and delete routes in IP routing tables

- **route print**: In order to view the entire contents of the IP routing table
- **route delete**: In order to delete all routes in the IP routing table
- **route add**: In order to add a static TCP/IP route to the IP routing table

**ACL Configuration**

Routers provide basic traffic filtering capabilities, such as blocking the Internet traffic with access control lists (ACLs). An ACL is a sequential list of **Permit** or **Deny** statements that apply to addresses or upper-layer protocols. These lists tell the router what types of packets to: **PERMIT** or **DENY**. When using an access-list to filter traffic, a PERMIT statement is used to “**allow**” traffic, while a DENY statement is used to “**block**” traffic.

23.2 Network setup:

Open NetSim and click Examples > Experiments > Basic-networking-commands-Ping-Route-Add/Delete/Print-and-ACL > Sample-1 as shown below:
NetSim UI displays the configuration file corresponding to this experiment as shown below:

23.3 Procedure:

The following set of procedures were done to generate this sample:

Step 1: A network scenario is designed in NetSim GUI comprising of 2 Wired Nodes and 3 Routers in the “Internetworks” Network Library.

Step 2: In the Network Layer properties of Wired Node 1, “ICMP Status” is set as TRUE.

Similarly, ICMP Status is set as TRUE for all the devices.
Step 3: In the General properties of Wired Node 1, **Wireshark Capture** is set as Online.

Step 4: Right click on the Application Flow **App1 CBR** and select Properties or click on the Application icon present in the top ribbon/toolbar.

A CBR Application is generated from Wired Node 1 i.e. Source to Wired Node 2 i.e. Destination with Packet Size remaining 1460Bytes and Inter Arrival Time remaining 20000µs. **Transport Protocol** is set to **UDP**.

Additionally, the “**Start Time(s)**” parameter is set to 30, while configuring the application. This time is usually set to be greater than the time taken for OSPF Convergence (i.e. Exchange of OSPF information between all the routers), and it increases as the size of the network increases.

Step 5: Packet Trace is enabled in NetSim GUI. At the end of the simulation, a very large .csv file is containing all the packet information is available for the users to perform packet level analysis.

Step 6: Click on Run Simulation. Simulation Time is set to 300 Seconds and in the **Runtime Interaction** tab, Interactive Simulation is set to True.

**NOTE:** It is recommended to specify a longer simulation time to ensure that there is sufficient time for the user to execute the various commands and see the effect of that before the Simulation ends.

Click on **Accept** and then click on **OK**.
• Simulation (NetSimCore.exe) will start running and will display a message "waiting for first client to connect" as shown below:

![Screenshot of NetSimCore.exe](image_url)

• Go back to the network scenario. Click on "Display Settings" in the top ribbon/toolbar and select the "Device IP" checkbox inorder to display the IP address of all the devices. Now, Right click on Router 3 or any other Router and select "NetSim Console" option.

![Screenshot of NetSim Console](image_url)

• Now Client (NetSimCLI.exe) will start running and it will try to establish a connection with NetSimCore.exe. After the connection is established, the following will be displayed:

```
Initialising Winsock...Initialised.
Connecting to device DESKTOP-LC53CTS.
Connection attempt: 1
Connection established.
NetSim>
```

• After this the command line interface can be used to execute all the supported commands.
23.4 Network Commands

Ping Command:

- You can use the *ping* command with an IP address or Device name
- ICMP_Status should be set as True in all nodes for ping to work

```
Ping <IP address> e.g. ping 11.4.1.2
Ping <Node Name> e.g. ping Wired_Node_2
```

Route Commands:

- In order to view the entire contents of the IP routing table, use following command *route print*

```
route print
```

- You’ll see the routing table entries with network destinations and the gateways to which packets are forwarded, when they are headed to that destination. Unless you’ve already added static routes to the table, everything you see here is dynamically generated.
• In order to delete a route in the IP routing table you’ll type a command using the following syntax

```
route delete destination_network
```

• So, to delete the route with destination network 11.5.1.2, all we’d have to do is type this command

```
route delete 11.5.1.2
```

• To check whether route has been deleted or not check again using `route print` command

• To add a static route to the table, you’ll type a command using the following syntax

```
route ADD destination_network MASK subnet_mask gateway_ip metric_cost interface
```

• So, for example, if you wanted to add a route specifying that all traffic bound for the 11.5.1.2 subnet went to a gateway at 11.5.1.1

```
route ADD 11.5.1.2 MASK 255.255.0.0 11.5.1.1 METRIC 100 IF 2
```

• If you were to use the route print command to look at the table now, you’d see your new static route.
NOTE: Entry added in IP table by routing protocol continuously gets updated. If a user tries to remove a route via route delete command, there is always a chance that routing protocol will re-enter this entry again. Users can use ACL / Static route to override the routing protocol entry if required.

ACL Configuration:

Commands to configure ACL:

- To view ACL syntax: `acl print`
- Before using ACL, we must first verify whether ACL option enabled. A common way to enable ACL is to use command: `ACL Enable`
- Enter configuration mode of ACL: `aclconfig`
- To view ACL Table: `Print`
- To exit from ACL configuration: `exit`
- To disable ACL: `ACL Disable` (use this command after `exit` from ACL Configuration)

To view ACL usage syntax use: `acl print`

[PERMIT, DENY] [INBOUND, OUTBOUND, BOTH] PROTO SRC DEST SPORT DPORT IFID

Step to Configure ACL:

- To create a new rule in the ACL use command as shown below to block UDP packet in Interface 2 and Interface 3 of Router 3.
- Application properties → Transport Protocol → UDP

Use the command as follows:
NetSim> **acl enable**  
ACL is enable  
NetSim> **aclconfig**  
**ROUTER_3/ACLCONFIG> acl print**  
Usage: [PERMIT, DENY] [INBOUND, OUTBOUND, BOTH] PROTO SRC DEST SPORT DPORT IFID  
**ROUTER_3/ACLCONFIG> DENY BOTH UDP ANY ANY 0 0 2**  
OK!  
**ROUTER_3/ACLCONFIG> DENY BOTH UDP ANY ANY 0 0 3**  
OK!  
**ROUTER_3/ACLCONFIG> print**  
DENY BOTH UDP ANY/0 ANY/0 0 0 2  
DENY BOTH UDP ANY/0 ANY/0 0 0 3  
**ROUTER_3/ACLCONFIG> exit**  
NetSim> **acl disable**  
ACL is disable  
NetSim>  

**Ping Command Results:**
Go to the Results Dashboard and click on “Open Packet Trace” option present in the Left-Hand-Side of the window and do the following:

Filter Control Packet Type/App Name to **ICMP EchoRequest** and **ICMP EchoReply**.

In Wireshark, apply filter as ICMP. we can see the ping request and reply packets in Wireshark.

**ACL Results:**

The impact of ACL rule applied over the simulation traffic can be observed in the IP Metrics Table in the simulation results window. In Router 3, the number of packets blocked by firewall has been shown below:
NOTE: Number of packets blocked may vary based on the time at which ACL is configured.

Users can also observe this in Packet Animation before and after the Packets are blocked as shown below:

- Check Packet animation window whether packets has been blocked in Router_3 or not after entering ACL command to deny UDP traffic
- Before applying ACL rule there is packet flow from Wired_Node_1 to Wired_Node_2
- After applying ACL rule Packet flows up to Router_3 only
24. Study how the throughput of LTE network varies as the distance between the ENB and UE (User Equipment) is increased

24.1 Theory:

LTE or Long Term Evolution, commonly known as 4G LTE, is a standard for wireless communication of high-speed data for mobile phones and data terminals. It is based on the GSM/EDGE and UMTS/HSPA network technologies, increasing the capacity and speed using a different radio interface.

The path loss in LTE is the decay of the signal power dissipated due to radiation on the wireless channels. Path loss may be due to many effects, such as free space loss, refraction, diffraction, reflection, aperture-medium coupling loss, and absorption.

Received power (\(P_r\)) can be calculated as:

**Case 1:** When no path loss Received power is same as Transmitted power, i.e., \(P_r = P_t\)

**Case 2:** When Line of Sight is there, Received power \(P_r\) is

\[
P_r = P_t + G_t + G_r + 20 \log_{10} \left( \frac{\lambda}{4\pi d} \right) + 10n \log_{10} \frac{d_0}{d}
\]

Where \(G_t\) and \(G_r\) are gains of transmitting and receiving antenna respectively. Here \(d\) is the distance between transmitter and receiver, \(\lambda\) is the wavelength of the transmitted signal and \(d_0\) is reference distance at which channel gain becomes 1. \(n\) is path loss exponent and \(P_t\) is transmitted power.

24.2 Network Setup:

Open NetSim and click **Examples > Experiments > Effect-of-distance-on-LTE-throughput > Sample-1** as shown below:
NetSim UI displays the configuration file corresponding to this experiment as shown below:

![NetSim UI](image)

24.3 Procedure:

Sample 1:

The following set of procedures were done to generate this sample:

**NOTE:** Before placement of any device grid length should be increased and it should be 10000 meters X 10000 meters. Click on Grid/Map Settings present in the ribbon and set grid length as 10000.

**Step 1:** A network scenario is designed in NetSim GUI comprising of 1 User Equipment, 1 ENB, 1 MME, 1 Router, and 1 Wired Node in the “LTE/LTE-A” Network Library.
**Step 2:** The device positions are set as per the below table:

<table>
<thead>
<tr>
<th></th>
<th>ENB 4</th>
<th>UE 5</th>
</tr>
</thead>
<tbody>
<tr>
<td>X/Lat</td>
<td>0</td>
<td>50</td>
</tr>
<tr>
<td>Y/Lon</td>
<td>0</td>
<td>50</td>
</tr>
</tbody>
</table>

**Step 3:** In the Interface LTE > Physical Layer > CA1 and CA2 Properties of ENB 4, Channel Bandwidth is set to 20 MHz for both the carriers.

**Step 4:** In the General Properties of UE 5 “Velocity (m/s)” parameter is set to 0.

**Step 5:** The Wired Link Properties are set as follows:

<table>
<thead>
<tr>
<th>Link Properties</th>
<th>Wired Link 2</th>
<th>Wired Link 3</th>
<th>Wired Link 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Uplink Speed (Mbps)</td>
<td>100</td>
<td>100</td>
<td>100</td>
</tr>
<tr>
<td>Downlink Speed (Mbps)</td>
<td>100</td>
<td>100</td>
<td>100</td>
</tr>
<tr>
<td>Uplink BER</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Downlink BER</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Up Time</td>
<td>N/A</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Down Time</td>
<td>N/A</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Uplink Propagation Delay (microsec)</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Downlink Propagation Delay (microsec)</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

**Step 6:** The Wireless Link Properties are set as follows:

<table>
<thead>
<tr>
<th>Link Properties</th>
<th>Wireless Link 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Channel characteristics</td>
<td>Path Loss Only</td>
</tr>
<tr>
<td>Path Loss Model</td>
<td>Log Distance</td>
</tr>
<tr>
<td>Path loss Exponent(n)</td>
<td>4</td>
</tr>
</tbody>
</table>

**Step 7:** Right click on the Application Flow App1 CUSTOM and select Properties or click on the Application icon present in the top ribbon/toolbar. Transport Protocol is set to UDP.

A CUSTOM Application is generated from Wired Node 1 i.e. Source to UE 5 i.e. Destination with Packet Size set to 1460 Bytes and Inter Arrival Time set to 165 µs.
The Packet Size and Inter Arrival Time parameters are set such that the Generation Rate equals 70 Mbps. Generation Rate can be calculated using the formula:

\[
\text{Generation Rate (Mbps)} = \text{Packet Size (Bytes)} \times 8/\text{Interarrival time (µs)}
\]

**Step 8:** Run the Simulation for 10 Seconds. Under Packet Animation, Don't Play or Record Animation option is selected for the simulation to run faster.

*NOTE: If users wish to view the packet animation, then select Record Animation option.*

**Sample 2:**

The following changes in settings are done from the previous sample for the remaining samples:

**Step 1:** The device positions are changed as follows:

<table>
<thead>
<tr>
<th>Change in UE Properties: (x, y)</th>
<th>Sample 2</th>
<th>(100, 100)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sample 3</td>
<td>(150, 150)</td>
<td></td>
</tr>
<tr>
<td>Sample 4</td>
<td>(200, 200)</td>
<td></td>
</tr>
<tr>
<td>Sample 5</td>
<td>(250, 250)</td>
<td></td>
</tr>
<tr>
<td>Sample 6</td>
<td>(300, 300)</td>
<td></td>
</tr>
<tr>
<td>Sample 7</td>
<td>(350, 350)</td>
<td></td>
</tr>
<tr>
<td>Sample 8</td>
<td>(400, 400)</td>
<td></td>
</tr>
</tbody>
</table>

**24.4 Output:**

**Step 1: Distance calculation:**

Calculate the Distance between ENB \((x_1, y_1)\) and UE \((x_2, y_2)\) as follows: \(\sqrt{(x_2-x_1)^2 + (y_2-y_1)^2}\)

For example, for Sample 1:

ENB \((x_1, y_1)\) = (0, 0); UE \((x_2, y_2)\) = (50, 50);

Distance = \(\sqrt{(50-0)^2 + (50-0)^2} = \sqrt{2} \times 50 = 50\sqrt{2}\) meters.

**Step 2:** Open any Excel File and note down the distance between the UE and ENB and the throughput values as shown below:
### Comparison Chart:

<table>
<thead>
<tr>
<th>Sample</th>
<th>Distance between UE and ENB (meters)</th>
<th>Throughput (Mbps)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$50\sqrt{2}$</td>
<td>70.71</td>
</tr>
<tr>
<td>2</td>
<td>$100\sqrt{2}$</td>
<td>140.42</td>
</tr>
<tr>
<td>3</td>
<td>$150\sqrt{2}$</td>
<td>212.13</td>
</tr>
<tr>
<td>4</td>
<td>$200\sqrt{2}$</td>
<td>282.84</td>
</tr>
<tr>
<td>5</td>
<td>$250\sqrt{2}$</td>
<td>353.55</td>
</tr>
<tr>
<td>6</td>
<td>$300\sqrt{2}$</td>
<td>424.26</td>
</tr>
<tr>
<td>7</td>
<td>$350\sqrt{2}$</td>
<td>494.97</td>
</tr>
<tr>
<td>8</td>
<td>$400\sqrt{2}$</td>
<td>565.68</td>
</tr>
</tbody>
</table>

To draw these graphs by using Excel "Insert ➔ Chart" option and then select chart type as "Line chart".

### 24.5 Inference:

As the distance increases between ENB and UE, throughput decreases. The reason is that as the distance increases between the devices, the received signal power decreases, and the LTE Phy Rate drops as the signal power reduces.
25. Study how the throughput of LTE network varies as the Channel bandwidth changes in the ENB (Evolved node)

25.1 Theory:

LTE or Long Term Evolution, commonly known as 4G LTE, is a standard for wireless communication of high-speed data for mobile phones and data terminals. It is based on the GSM/EDGE and UMTS/HSPA network technologies, increasing the capacity and speed using a different radio interface.

LTE supports flexible carrier bandwidths, from 1.4 MHz up to 20 MHz as well as both FDD and TDD. LTE designed with a scalable carrier bandwidth from 1.4 MHz up to 20 MHz which bandwidth is used depends on the frequency band and the amount of spectrum available with a network operator.

25.2 Network Setup:

Open NetSim and click Examples > Experiments > Impact-of-channel-bandwidth-on-LTE-throughput > Sample-1 as shown below:

NetSim UI displays the configuration file corresponding to this experiment as shown below:
25.3 Procedure:

The following set of procedures were done to generate this sample:

**Step 1:** A network scenario is designed in NetSim GUI comprising of 3 User Equipment’s, 1 ENB, 1 MME, 1 Router, and 1 Wired Node in the “LTE/LTE-A” Network Library.

**Step 2:** In the Interface LTE > Physical Layer of ENB 4, Carrier Aggregation is set to Inter Band Noncontiguous CA.

In the Interface LTE > Physical Layer > CA1 and CA2 Properties of ENB 4, Channel Bandwidth is set to 10 MHz for both the carriers.

**Step 3:** In the General Properties of all the UE’s “Velocity (m/s)” parameter is set to 0.

**Step 4:** The Wired Link Properties are set as follows:

<table>
<thead>
<tr>
<th>Link Properties</th>
<th>Wired Link 1</th>
<th>Wired Link 2</th>
<th>Wired Link 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Uplink Speed (Mbps)</td>
<td>1000</td>
<td>1000</td>
<td>1000</td>
</tr>
<tr>
<td>Downlink Speed (Mbps)</td>
<td>1000</td>
<td>1000</td>
<td>1000</td>
</tr>
<tr>
<td>Uplink BER</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Downlink BER</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

**Step 5:** In the Wireless Link Properties, Channel Characteristics is set to NO PATHLOSS.

**Step 6:** Three CUSTOM Applications are configured as per the table given below:

<table>
<thead>
<tr>
<th>Application Properties</th>
<th>Application 1</th>
<th>Application 2</th>
<th>Application 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Application Type</td>
<td>Custom</td>
<td>Custom</td>
<td>Custom</td>
</tr>
</tbody>
</table>
### Step 7: Run the Simulation for 10 Seconds.

The following changes in settings are done from the previous sample for the remaining samples:

**Sample 2:**

**Step 1:** In the Interface LTE > Physical Layer > CA1 and CA2 Properties of ENB 4, Channel Bandwidth is set to 10 and 5 MHz respectively.

**Step 2:** Run the Simulation for 10 Seconds.

**Sample 3:**

**Step 1:** In the Interface LTE > Physical Layer > CA1 and CA2 Properties of ENB 4, Channel Bandwidth is set to 5 and 5 MHz respectively.

**Step 2:** Run the Simulation for 10 Seconds.

### 25.4 Output

Add the sum of all throughput values in each sample case:

**Example: Sample 1**

<table>
<thead>
<tr>
<th>Application Id</th>
<th>Throughput (Mbps)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>23.177</td>
</tr>
<tr>
<td>2</td>
<td>23.177</td>
</tr>
<tr>
<td>3</td>
<td>23.177</td>
</tr>
<tr>
<td><strong>Sum</strong></td>
<td><strong>69.531</strong></td>
</tr>
</tbody>
</table>

Same procedure can be followed for the other samples.
Open any Excel file and note down the sum of applications throughput values as shown in below table:

<table>
<thead>
<tr>
<th>Sample</th>
<th>Channel Bandwidth(MHz)</th>
<th>Throughput (Mbps)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>20</td>
<td>69.53</td>
</tr>
<tr>
<td>2</td>
<td>15</td>
<td>52.11</td>
</tr>
<tr>
<td>3</td>
<td>10</td>
<td>34.7</td>
</tr>
</tbody>
</table>

**Comparison Chart:**

To draw these graphs by using Excel “Insert ➔ Chart” option and then select chart type as “Line chart”.

![Graph](image)

**25.5 Inference**

LTE provides spectrum flexibility with scalable transmission bandwidth between 1.4 MHz and 20 MHz depending on the available spectrum for flexible radio planning. The 20 MHz bandwidth can provide up to 150 Mbps downlink user data rate and 75 Mbps uplink peak data rate with 2×2 MIMO, and 300 Mbps with 4×4 MIMO.

As the channel bandwidth decreases the number of resource blocks also decreases. If more resource blocks are available then more number of packets can be transmitted.

<table>
<thead>
<tr>
<th>Channel Bandwidth (MHz)</th>
<th>1.4</th>
<th>3</th>
<th>5</th>
<th>10</th>
<th>15</th>
<th>20</th>
</tr>
</thead>
<tbody>
<tr>
<td>Transmission Bandwidth</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Configuration NRB: (1 resource block = 180kHz in 1ms TTI)</td>
<td>6</td>
<td>15</td>
<td>25</td>
<td>50</td>
<td>75</td>
<td>100</td>
</tr>
</tbody>
</table>
26. Simulate and study LTE Handover procedure

26.1 Introduction:

As defined by 3GPP, handover is a procedure for changing the serving cell of a UE. The two eNodeBs involved in the process are typically called the source eNB (S-eNB) and the target eNB (T-eNB). In NetSim, handover procedure is triggered “automatically” by the serving eNodeB of the UE.

26.2 Description and Definitions:

1. A data call is established between the UE, S-eNB (Source-eNB) and the network elements. Data packets are transferred to/from the UE to/from the network in both directions (Downlink as well as Uplink)
2. The network sends the MEASUREMENT CONTROL REQ message to the UE to set the parameters to measure and set thresholds for those parameters. Its purpose is to instruct the UE to send a measurement report to the network as soon as it detects the thresholds.
3. The UE sends the MEASUREMENT REPORT to the Serving eNB, which contains the RQRS from all the nearby eNBs. The Serving eNB makes the decision to hand off the UE to a T-eNB (Target-eNB) using the handover algorithm mentioned in the Introduction
4. The S-eNB then initiates the decision to handover using the X2 interface.
5. The S-eNB issues a HANDOVER REQUEST message to the T-eNB passing necessary information to prepare the handover at the target side
6. The T-eNB sends back the HANDOVER REQUEST ACKNOWLEDGE message including a transparent container to be sent to the UE as an RRC message to perform the handover.
7. The S-eNB generates the RRC (Radio resource control used for signaling transfer) message to perform the handover, i.e., RRC CONNECTION RECONFIGURATION message including the mobility Control Information.
8. The S-eNB starts forwarding the downlink data packets to the T-eNB for all the data bearers which are being established in the T-eNB during the HANDOVER REQ message processing.
9. The T-eNB now requests the S-eNB to release the resources. With this, the handover procedure is complete.
26.3 Analysis/Algorithm:
NetSim handover algorithm utilizes the Reference Signal Received Quality (RSRQ) measurements, to trigger the handover. When the target eNB’s RSRQ crosses the serving eNB’s RSRQ by a factor know as margin of handover (equal to 3dB), hand over is triggered.

26.4 Network Setup:
Open NetSim and click Examples > Experiments > LTE-Handover-procedure > Sample-1 as shown below:
26.5 Procedure:

The following set of procedures were done to generate this sample:

Step 1: A network scenario is designed in NetSim GUI comprising of 2 ENBs, 1 MME, and 2 UEs in the “LTE/LTE-A” Network Library.

Step 2: The device positions are set as per the table given below:

<table>
<thead>
<tr>
<th></th>
<th>ENB 1</th>
<th>ENB 2</th>
<th>UE 3</th>
<th>UE 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>X Co-ordinate</td>
<td>1000</td>
<td>4000</td>
<td>1000</td>
<td>4000</td>
</tr>
<tr>
<td>Y Co-ordinate</td>
<td>1500</td>
<td>1500</td>
<td>3000</td>
<td>3000</td>
</tr>
</tbody>
</table>

Step 3: In the General Properties of UE 3 and UE 4, set Mobility Model as File Based Mobility.

Step 4: Right click on the Wireless Link 3 and select Properties, the following is set:
Similarly, it is set for Wireless Link 4.

**Step 5:** Right click on the Application Flow App1 CBR and select Properties or click on the Application icon present in the top ribbon/toolbar.

A CBR Application is generated from UE 3 i.e. Source to UE 4 i.e. Destination with Packet Size remaining 1460Bytes and Inter Arrival Time remaining 20000µs.

Additionally, the “Start Time(s)” parameter is set to 20, while configuring the application.

**File Based Mobility:**

In File Based Mobility, users can write their own custom mobility models and define the movement of the mobile users. Create a mobility.txt file for UE’s involved in mobility with each step equal to 0.5 sec with distance 50 m. The file present in the Docs folder of NetSim Install directory <\C:\Program Files\NetSim Standard\Docs\Sample_Configuration\NetSim_Experiment_Manual\Experiment-23-LTE-Handover\Sample-1>. For more information, please refer section 3.3.4 “File Based Mobility Format” under MANET Technology Library as shown below:

The NetSim Mobility File format is as follows:

```
mobility.txt

#Initial position of the UE 3

$node_(2) set X_ 1000.0
```
$node_(2) set Y_ 3000.0
$node_(2) set Z_ 0.0

#Initial position of the UE 4
$node_(3) set X_ 4000.0
$node_(3) set Y_ 3000.0
$node_(3) set Z_ 0.0

#Positions of the UE 3 at specific time
$time 0.0 "$node_(2) 1000.0 3000.0 0.0"
$time 0.5 "$node_(2) 1050.0 3000.0 0.0"
$time 1.0 "$node_(2) 1100.0 3000.0 0.0"
$time 1.5 "$node_(2) 1150.0 3000.0 0.0"
$time 2.0 "$node_(2) 1200.0 3000.0 0.0"
$time 2.5 "$node_(2) 1250.0 3000.0 0.0"
$time 3.0 "$node_(2) 1300.0 3000.0 0.0"
$time 3.5 "$node_(2) 1350.0 3000.0 0.0"
$time 4.0 "$node_(2) 1400.0 3000.0 0.0"
$time 4.5 "$node_(2) 1450.0 3000.0 0.0"
$time 5.0 "$node_(2) 1500.0 3000.0 0.0"
$time 5.5 "$node_(2) 1550.0 3000.0 0.0"
$time 6.0 "$node_(2) 1600.0 3000.0 0.0"
$time 6.5 "$node_(2) 1650.0 3000.0 0.0"
$time 7.0 "$node_(2) 1700.0 3000.0 0.0"
$time 7.5 "$node_(2) 1750.0 3000.0 0.0"
$time 8.0 "$node_(2) 1800.0 3000.0 0.0"
$time 8.5 "$node_(2) 1850.0 3000.0 0.0"
$time 9.0 "$node_(2) 1900.0 3000.0 0.0"
$time 9.5 "$node_(2) 1950.0 3000.0 0.0"
$time 10.0 "$node_(2) 2000.0 3000.0 0.0"
$time 10.5 "$node_(2) 2050.0 3000.0 0.0"
$time 11.0 "$node_(2) 2100.0 3000.0 0.0"
$time 11.5 "$node_(2) 2150.0 3000.0 0.0"
$time 12.0 "$node_(2) 2200.0 3000.0 0.0"
$time 12.5 "$node_(2) 2250.0 3000.0 0.0"
$time 13.0 "$node_(2) 2300.0 3000.0 0.0"
$time 13.5 "$node_(2) 2350.0 3000.0 0.0"
$time 14.0 "$node_(2) 2400.0 3000.0 0.0"
$time 14.5 "$node_(2) 2450.0 3000.0 0.0"
$time 15.0 "$node_(2) 2500.0 3000.0 0.0"
$time 15.5 "$node_(2) 2550.0 3000.0 0.0"
$time 16.0 "$node_(2) 2600.0 3000.0 0.0"
$time 16.5 "$node_(2) 2650.0 3000.0 0.0"
$time 17.0 "$node_(2) 2700.0 3000.0 0.0"
$time 17.5 "$node_(2) 2750.0 3000.0 0.0"
$time 18.0 "$node_(2) 2800.0 3000.0 0.0"
$time 18.5 "$node_(2) 2850.0 3000.0 0.0"
$time 19.0 "$node_(2) 2900.0 3000.0 0.0"
$time 19.5 "$node_(2) 2950.0 3000.0 0.0"
$time 20.0 "$node_(2) 3000.0 3000.0 0.0"
$time 20.5 "$node_(2) 3050.0 3000.0 0.0"
$time 21.0 "$node_(2) 3100.0 3000.0 0.0"
Step 6: Packet Trace is enabled in NetSim GUI. At the end of the simulation, a very large .csv file is containing all the packet information is available for the users to perform packet level analysis.

Step 7: Run the Simulation for 50 Seconds.

26.6 Measurements and Outputs

Open Packet Animation:

As UE moves from one position to another it sends measurement report to each ENB in range. As it moves SNR received by each ENB keeps on changing based on distance between ENB and UE. If the difference between SNR received by new ENB to that of old ENB to which it is connected is greater at that point handover occurs.

26.7 Inference

1. As shown in the above packet animation table, UE 3 connected to eNB 1 and UE 4, connected to eNB 2
2. UE 3 is moving from eNB 1 to eNB 2 due to mobility
3. Then UE 3 sends the LTE_Measurement_Report to eNB 1
4. The eNB 1 sends a LTE_Handover_Request message to the eNB 2, if the received SNR by eNB 2 is greater than eNB 1, by 3dB (margin of handover)
5. eNB 2 checks for resource availability and sends a LTE_Handover_Request_Ack message to the eNB 1
6. Now UE 3 starts communicating with eNB 2 shown in the above screenshot
26.8 Additional Notes & References:

1. To calculate and print SNR for each pair of eNB-UE combination please refer NetSim knowledgebase article (https://tetcos.freshdesk.com/solution/articles/14000037296-how-can-i-print-snr-cqi-mcs-index-and-tbs-index-value-to-a-file)

2. If the wireless links have no path loss set, then there will never be any handovers because the received power from all eNB’s will be the same
27. Understand the working of LTE Device to Device Communication

27.1 Theory:

LTE D2D communication is a peer to peer link which does not use the cellular network infrastructure, but enables LTE based devices to communicate directly with one another when they are in close proximity.

D2D would enable the direct link of a device user equipment UE to another device using the cellular spectrum. This could allow large volumes of media or other data to be transferred from one device to another over short distances and using a direct connection. This form of device to device transfer would enable the data to be transferred without the need to run it via the cellular network itself, thereby avoiding problems with overloading the network.

The D2D model can be summarized as follows:

- Each UE produces its D2D identity and transmits it to the eNB during its first access to the network.
- UE’s make D2D spectrum requests including the D2D identity of the target D2D receiver.
- eNB launches a peer discovery procedure for the requested D2D pair.
- eNB allocates cellular resources to valid D2D pairs and informs both D2D peers, tuning them indirectly at the same spectrum portion.
- The UE transmitter sends its data using the spectrum region that has been allocated by the eNB, while the UE receiver tunes to the same spectrum region to receive the transmitted data.
- The UE receiver acknowledges the reception of the data through the eNB.

27.2 Benefits of D2D communications

Direct communications between devices can provide several benefits to users in various applications where the devices are in close proximity:

- **Reliable communications:** LTE Device to Device can be used to communicate locally between devices to provide highly reliable communications especially if the LTE network has failed for any reason - even as a result of the disaster.
- **Instant communications:** As the D2D communications does not rely on the network infrastructure the devices could be used for instant communications between a set
numbers of devices in the same way that walkie-talkies are used. This is particularly applicable to the way communications may be used by the emergency services.

- **Interference reduction**: By not having to communicate directly with a base station, fewer links are required (i.e. essentially only between devices) and this has an impact of the amount of data being transmitted within a given spectrum allocation. This reduces the overall level of interference.

- **Power saving**: Using device to device communication provides energy saving, if the two devices are in close proximity then lower transmission power levels are required.

### 27.3 Network Setup:

Open NetSim and click **Examples > Experiments > LTE-Device-to-Device-Communication > Sample-1** as shown below:

![NetSim UI](image)

NetSim UI displays the configuration file corresponding to this experiment as shown below:

![NetSim Configuration](image)
27.4 Procedure:
The following set of procedures were done to generate this sample:

**Step 1:** A network scenario is designed in NetSim GUI comprising of 1 Wired Node, 1 Router, 1 MME, 1 ENB, and 4 UEs in the “LTE D2D” Network Library.

**Step 2:** The device positions are set as per the below table:

<table>
<thead>
<tr>
<th>Device Type</th>
<th>X - Coordinate</th>
<th>Y - Coordinate</th>
</tr>
</thead>
<tbody>
<tr>
<td>eNB 1</td>
<td>4500</td>
<td>1000</td>
</tr>
<tr>
<td>UE 2</td>
<td>3500</td>
<td>1000</td>
</tr>
<tr>
<td>UE 3</td>
<td>2500</td>
<td>1000</td>
</tr>
<tr>
<td>UE 4</td>
<td>1500</td>
<td>1000</td>
</tr>
<tr>
<td>UE 5</td>
<td>500</td>
<td>1000</td>
</tr>
</tbody>
</table>

**Step 3:** In the Interface (LTE) > Data Link Layer Properties of UE 2, D2D Enable is set to FALSE and similarly for UE 3, UE 4, and UE 5.

**Step 4:** The Wireless Link properties is set as follows:

<table>
<thead>
<tr>
<th>Wireless Link Properties</th>
</tr>
</thead>
<tbody>
<tr>
<td>Channel Characteristics</td>
</tr>
<tr>
<td>Path loss Model</td>
</tr>
<tr>
<td>Path loss exponent</td>
</tr>
</tbody>
</table>

**Step 5:** Right click on the Application Flow App1 CBR and select Properties or click on the Application icon present in the top ribbon/toolbar. Transport Protocol is set to UDP.

A CBR Application is generated from Wired Node 8 i.e. Source to UE 5 i.e. Destination with Packet Size remaining 1460Bytes and Inter Arrival Time remaining 20000µs.

**Step 6:** Packet Trace is enabled in NetSim GUI. At the end of the simulation, a very large .csv file is containing all the packet information is available for the users to perform packet level analysis.

**Step 7:** Run the Simulation for 10 Seconds.

The following changes in settings are done from the previous sample:
Step 1: In the Interface (LTE) > Data Link Layer Properties of UE 2, D2D Enable is set to TRUE and similarly for UE 3, UE 4, and UE 5.

Step 2: Run the Simulation for 10 Seconds.

27.5 Output:

Sample 1: Without D2D:

As shown in above figure, application is set from Wired Node 8 to UE 5. As UE 5 is far away from eNB 1, there is too much of attenuation and packets from eNB 1 to UE 5 get errored. We can observe this in Packet Animation. This results in a very low or zero throughput. Users can also observe in the packet animation that only LTE_RLC_SDUs are errored (in red color). The same can also be seen from the Packet trace by filtering CONTROL_PACKET_TYPE to LTE_RLC_SDU packets. For doing this refer section 7.5 in NetSim’s user manual.

Sample 2: With D2D:

In second case, even though UE 5 is far away from eNB 1, packets will reach to UE 5 via intermediate UEs (in this case UE 4). Users can observe this in Animation and Packet Trace. As shown in the figure below, eNB 1 is first transmitting the LTE_RLC_SDU packets to UE 4 and then UE 4 is transmitting to UE 5 using LTE Device to device communication. In this case, we get considerably higher throughput since the errored packets are less.
Users can also observe this in Packet trace by filtering CONTROL_PACKET_TYPE to LTE_RLC_SDU packets.

From the above figure, users can observe that eNB-1 is transmitting LTE_RLC_SDU packet to UE-4 and then UE-4 is transmitting to UE-5.
28. To analyze how the allocation of frequency spectrum to the Incumbent (Primary) and CR CPE (Secondary User) affects throughput

28.1 Introduction:

An important component of the cognitive radio concept is the ability to measure, sense, learn, and be aware of the parameters related to the radio channel characteristics, availability of spectrum and power, radio’s operating environment, user requirements and applications, available networks (infrastructures) and nodes, local policies and other operating restrictions.

NetSim simulator models IEEE 802.22 Cognitive Radio per the theory explained below.

A spectrum hole has been defined as a band of frequencies assigned to a primary user, but at a particular time and specific geographic location, the band is not being utilized by that user. Cognitive Radio was proposed as the means to promote the efficient use of spectrum by exploiting the existence of spectrum holes.

These spectrum holes are used by the SU for its transmission. This scheme is often referred to as opportunistic spectrum access (OSA). No concurrent transmission of the PU and the SU is allowed. The SU must vacate the channel as soon as the PU reappears, which leads to the forced termination of the SU connection (if there is no other available channel for the SU). Since the SU has no control over the resource availability, the transmission of the SU is blocked when the channel is occupied by the PU. The forced termination and blocking of a SU connection is shown in the below figure. The forced termination probability and blocking probability are the key parameters which determine the throughput of the SU, and thus its viable existence. The forced termination depends on the traffic behavior of the PUs and the SU (e.g. arrival rates, service time etc.). In the case of multiple SU groups with
different traffic statistics, the forced termination and blocking probabilities lead to unfairness among the SU groups.

Illustration of forced termination and blocking

**Performance metrics:**

The different parameters used to analyze the performance are explained as follows:

- **Throughput:** It is the rate of successfully transmitted data packets in unit time in the network during the simulation.
- **Spectral Efficiency:** It refers to the information rate that can be transmitted over a given bandwidth in a specific communication system. It is a measure of how efficiently a limited frequency spectrum is utilized by the physical layer protocol, and sometimes by the media access control protocol.

**28.2 Network Setup:**

Open NetSim and click on *Examples > Experiments > Cognitive-Radio-Impact-of-frequency-allocation-to-PU-and-SU-on-throughput > Sample-1* as shown below:
NetSim UI displays the configuration file corresponding to this experiment as shown below:

![NetSim UI Configuration](image)

**28.3 Procedure:**

The following set of procedures were done to generate this sample:

**Step 1:** A network scenario is designed in NetSim GUI comprising of 1 Base Station and 2 CR CPE's in the "Cognitive Radio" Network Library.

**Step 2:** The device positions are set as follows:

<table>
<thead>
<tr>
<th>Base Station 1</th>
<th>CR CPE 2</th>
<th>CR CPE 3</th>
</tr>
</thead>
</table>
Step 3: In the Interface Cognitive Radio > Datalink Layer > Incumbent1, the following are set as shown below:

Step 4: In the Interface Cognitive Radio > Physical Layer, the Min Frequency and Max Frequency parameters are set to 54 and 60 MHz respectively.

Step 5: Right click on the Application Flow App1 CUSTOM and select Properties or click on the Application icon present in the top ribbon/toolbar.

A CUSTOM Application is generated from CR CPE 2 i.e. Source to CR CPE 3 i.e. Destination with Packet Size remaining 1460Bytes and Inter Arrival Time remaining 20000µs.

Step 6: Run the Simulation for 100 Seconds.

Sample 2:

The following changes in settings are done from the previous sample:
Step 1: In the Interface Cognitive Radio > Physical Layer, the Min Frequency and Max Frequency parameters are set to 54 and 90 MHz respectively.

Step 2: Run the Simulation for 100 Seconds.

28.4 Output:
Once after the simulation is complete, go to the Results Dashboard and check the “Application Metrics” Table. Throughput of the application will be 0.

In the Left-Hand-Side of the Results Dashboard, click on the arrow pointer indicating “CR Metrics”, from the drop down select the “Channel Metrics” which gives you the Spectral Efficiency.

Sample 1:

![Sample 1](image1)

Sample 2:

![Sample 2](image2)
28.5 Inference:

In both the samples, the Secondary User (CR-CPE) lies within the operational region of Primary User (Incumbent), hence the frequency spectrum used by operational Primary User (Incumbent) will not be used by Secondary User (CR-CPE). Also the Operational Interval under Incumbent is set to zero, i.e., the Incumbent will continuously use the channel allocated to it.

In the first sample, both the Primary User (Incumbent) and the Secondary User (CR-CPE) has been allocated the same channel (frequency band of 54 - 60 MHz). As Incumbent will continuously use the channel allocated to it, so there will be no Spectrum Hole, hence the secondary user will not be able to transmit any data in an opportunistic manner. Therefore, the throughput of the application in the CR-CPE and the spectral efficiency is almost equal to zero.

In the second sample, the Primary User (Incumbent) has been allocated frequency band of 54 - 60 MHz and the Secondary User (CR-CPE) has been allocated the frequency band of 54 - 90 MHz. Incumbent will continuously use the channel allocated to it, but the rest channels will remain free i.e. there will be Spectrum Hole, which the CR-CPE will utilize to transmit data.

**NOTE:** The results are highly dependent on position/velocity/traffic etc. Any modifications with the above-mentioned input parameters will change the final output result.
29. Understanding VLAN operation in L2 and L3 Switches

29.1 Introduction to VLAN:

VLAN is called as virtual local area network, used in Switches and it operates at Layer 2 and Layer 3. A VLAN is a group of hosts which communicate as if they were attached to the same broadcast domain, regardless of their physical location.

For example, all workstations and servers used by a particular workgroup team can be connected to the same VLAN, regardless of their physical connections to the network or the fact that they might be intermingled with other teams. VLANs have the same attributes as physical LANs, but you can group end stations even if they are not physically located on the same LAN segment.

A VLAN behaves just like a LAN in all respects but with additional flexibility. By using VLAN technology, it is possible to subdivide a single physical switch into several logical switches. VLANs are implemented by using the appropriate switch configuration commands to create the VLANs and assign specific switch interfaces to the desired VLAN.

Switches implement VLANs by adding a VLAN tag to the Ethernet frames as they enter the switch. The VLAN tag contains the VLAN ID and other information, which is determined by the interface from which the frame enters the switch. The switch uses VLAN tags to ensure that each Ethernet frame is confined to the VLAN to which it belongs based on the VLAN ID contained in the VLAN tag. The VLAN tags are removed as the frames exit the switch on the way to their destination.

Any port can belong to a VLAN, and unicast, broadcast, and multicast packets are forwarded and flooded only to end stations in that VLAN. Each VLAN is considered a logical network. Packets destined for stations that do not belong to the VLAN must be forwarded through a router.
In the below screenshot, the stations in the development department are assigned to one VLAN, the stations in the marketing department are assigned to another VLAN, and the stations in the testing department are assigned to another VLAN.

VLANs divide broadcast domains in a LAN environment. Whenever hosts in one VLAN need to communicate with hosts in another VLAN, the traffic must be routed between them. This is known as Inter-VLAN routing. This can be possible by using L3 switch.

What is a layer 3 switch?

Layer 3 switch (also known as a multi-layer switch) is a multi-functional device that have the same functionality like a layer 2 switch, but behaves like a router when necessary. It’s generally faster than a router due to its hardware based routing functions, but it’s also more expensive than a normal switch.

29.2 Network Setup:

Open NetSim and click Examples > Experiments > Understanding-VLAN-operation-in-L2-and-L3-Switches > Sample-1 as shown below:
NetSim UI displays the configuration file corresponding to this experiment as shown below:

![NetSim UI Configuration File](image)

**29.3 Procedure:**

**Sample 1: Intra-VLAN:**

Intra-VLAN is a mechanism in which hosts in same VLAN can communicate to each other.

The following set of procedures were done to generate this sample:
**Step 1:** A network scenario is designed in NetSim GUI comprising of 3 Wired Nodes and 1 L2 Switch in the “Internetworks” Network Library.

**Step 2:** L2 Switch 1 Properties are configured as follows:

<table>
<thead>
<tr>
<th>Interface ID</th>
<th>VLAN Status</th>
<th>VLAN ID</th>
<th>VLAN Port Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Interface_1</td>
<td>TRUE</td>
<td>2</td>
<td>Access _Port</td>
</tr>
<tr>
<td>Interface_2</td>
<td>TRUE</td>
<td>2</td>
<td>Access _Port</td>
</tr>
<tr>
<td>Interface_3</td>
<td>TRUE</td>
<td>3</td>
<td>Access _Port</td>
</tr>
</tbody>
</table>

In all the INTERFACE (ETHERNET) > DATALINK LAYER Properties of L2 Switch 1, “VLAN Status” is set to TRUE.

Now click on “Configure VLAN” option and the VLAN 2 fields are entered as shown below:
To add a new entry after entering the required fields, click on the ADD button.

To configure another VLAN, click on the “+” symbol located in the top.

And then we can add the entry to it.

**Step 3:** Run simulation for 10 Seconds and observe the throughputs.
Sample 2: Inter-VLAN:

NetSim UI displays the configuration file corresponding to this experiment as shown below:

The following set of procedures were done to generate this sample:

**Step 1:** A network scenario is designed in NetSim GUI comprising of 5 Wired Nodes and 1 L3 Switch in the “Internetworks” Network Library.

**Step 2:** The Wired Node properties are set as per the below table:

<table>
<thead>
<tr>
<th>Node</th>
<th>Wired Node2</th>
<th>Wired Node3</th>
<th>Wired Node4</th>
<th>Wired Node5</th>
<th>Wired Node6</th>
</tr>
</thead>
<tbody>
<tr>
<td>IP Address</td>
<td>10.0.0.4</td>
<td>10.1.0.4</td>
<td>11.2.0.4</td>
<td>11.3.0.4</td>
<td>11.4.0.4</td>
</tr>
<tr>
<td>Default Gateway</td>
<td>10.0.0.3</td>
<td>10.1.0.3</td>
<td>11.2.0.3</td>
<td>11.3.0.3</td>
<td>11.4.0.3</td>
</tr>
</tbody>
</table>

**Step 3:** The L3 Switch 1 Properties are set as per the below table:

<table>
<thead>
<tr>
<th>L3 Switch</th>
<th>I/f1_Ethernet</th>
<th>I/f2_Ethernet</th>
<th>I/f3_Ethernet</th>
<th>I/f4_Ethernet</th>
<th>I/f5_Ethernet</th>
</tr>
</thead>
<tbody>
<tr>
<td>IP Address</td>
<td>10.0.0.3</td>
<td>10.1.0.3</td>
<td>11.2.0.3</td>
<td>11.3.0.3</td>
<td>11.4.0.3</td>
</tr>
</tbody>
</table>
### L3 Switch 1

<table>
<thead>
<tr>
<th>Interface ID</th>
<th>VLAN Status</th>
<th>VLAN ID</th>
<th>VLAN Port Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Interface_1</td>
<td>TRUE</td>
<td>2</td>
<td>Access Port</td>
</tr>
<tr>
<td>Interface_2</td>
<td>TRUE</td>
<td>2</td>
<td>Access Port</td>
</tr>
<tr>
<td>Interface_3</td>
<td>TRUE</td>
<td>3</td>
<td>Access Port</td>
</tr>
<tr>
<td>Interface_4</td>
<td>TRUE</td>
<td>3</td>
<td>Access Port</td>
</tr>
<tr>
<td>Interface_5</td>
<td>TRUE</td>
<td>3</td>
<td>Access Port</td>
</tr>
</tbody>
</table>

The VLAN configurations done are shown as follows:

![Configure VLAN](image1.png)

![Configure VLAN](image2.png)
Step 3: Run simulation for 10 seconds and observe the throughputs.

### 29.4 Output and Inference: I

<table>
<thead>
<tr>
<th>Throughput (Mbps)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Application 1</td>
</tr>
<tr>
<td>Application 2</td>
</tr>
</tbody>
</table>

The throughput for 2nd application is zero because the source and destination is in different VLANs, thereby traffic flow or communication between 2 VLANs using Layer2 switch is not possible. To overcome this problem, an L3 switch is used.

### 29.5 Output and Inference: II

<table>
<thead>
<tr>
<th>Throughput (Mbps)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Application 1</td>
</tr>
<tr>
<td>Application 2</td>
</tr>
<tr>
<td>Application 3</td>
</tr>
</tbody>
</table>

In this case, application1 is in VLAN2, application2 is in VLAN3 and application 3 is in between VLAN2 and VLAN3. From the above results, the throughput for application 3 (different VLANs) is non zero, because of using L3 switch. So, communication between 2 VLANs is possible using L3 Switch.
30. Understanding Access and Trunk Links in VLANs

30.1 Theory

The links connecting the end devices are called access links. These are the links usually carrying the Data VLAN information. The link between the switches is called trunk link. It carries packets from all the VLANs.

Access link:

Access link connection is the connection where switch port is connected with a device that has a standardized Ethernet NIC. Standard NIC only understand IEEE 802.3 or Ethernet II frames. Access link connection can only be assigned with single VLAN. That means all devices connected to this port will be in same broadcast domain.

For example twenty users are connected to a hub, and we connect that hub with an access link port on switch, then all of these users belong to same VLAN. If we want to keep ten users in another VLAN, then we need to plug in those ten users to another hub and then connect it with another access link port on switch.

Trunk link:

Trunk link connection is the connection where switch port is connected with a device that is capable to understand multiple VLANs. Usually trunk link connection is used to connect two switches. A
VLAN can span anywhere in network, and that can happen due to trunk link connection. Trunking allows us to send or receive VLAN information across the network. To support trunking, original Ethernet frame is modified to carry VLAN information.

### 30.2 Network Setup:

Open NetSim and click **Examples > Experiments > Understanding-Access-and-Trunk-Links-in-VLANs** as shown below:

NetSim UI displays the configuration file corresponding to this experiment as shown below:
30.3 Procedure:

The following set of procedures were done to generate this sample:

**Step 1:** A network scenario is designed in NetSim GUI comprising of 4 Wired Nodes and 2 L2 Switches in the “Internetworks” Network Library.

**Step 2:** In the INTERFACE (ETHERNET) > NETWORK LAYER Properties, set the following:

<table>
<thead>
<tr>
<th>Node</th>
<th>Wired Node 3</th>
<th>Wired Node 4</th>
<th>Wired Node 5</th>
<th>Wired Node 6</th>
</tr>
</thead>
<tbody>
<tr>
<td>IP Address</td>
<td>192.168.1.3</td>
<td>192.168.1.4</td>
<td>192.168.2.3</td>
<td>192.168.2.4</td>
</tr>
<tr>
<td>Default Gateway</td>
<td>192.168.1.1</td>
<td>192.168.1.2</td>
<td>192.168.2.1</td>
<td>192.168.2.2</td>
</tr>
<tr>
<td>Subnet Mask</td>
<td>255.255.255.0</td>
<td>255.255.255.0</td>
<td>255.255.255.0</td>
<td>255.255.255.0</td>
</tr>
</tbody>
</table>

**NOTE:** The subnet mask of all L3 Switch interfaces is set to 255.255.255.0

**Step 3:** L3 Switch 1 and L3 Switch 2 properties are set as follows:

<table>
<thead>
<tr>
<th>Switch</th>
<th>I/f1_Ethernet</th>
<th>I/f2_Ethernet</th>
<th>I/f3_Ethernet</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>IP Address</td>
<td>IP Address</td>
<td>IP Address</td>
</tr>
<tr>
<td>L3 Switch 1</td>
<td>192.168.1.1</td>
<td>192.168.3.1</td>
<td>192.168.2.1</td>
</tr>
</tbody>
</table>
### L3 Switch 1

<table>
<thead>
<tr>
<th>Interface ID</th>
<th>VLAN Status</th>
<th>VLAN ID</th>
<th>VLAN Port Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Interface_1</td>
<td>TRUE</td>
<td>2</td>
<td>Access _Port</td>
</tr>
<tr>
<td>Interface_2</td>
<td>TRUE</td>
<td>1</td>
<td>Trunk _Port</td>
</tr>
<tr>
<td>Interface_3</td>
<td>TRUE</td>
<td>3</td>
<td>Access _Port</td>
</tr>
</tbody>
</table>

### L3 Switch 2

<table>
<thead>
<tr>
<th>Interface ID</th>
<th>VLAN Status</th>
<th>VLAN ID</th>
<th>VLAN Port Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Interface_1</td>
<td>TRUE</td>
<td>2</td>
<td>Access _Port</td>
</tr>
<tr>
<td>Interface_2</td>
<td>TRUE</td>
<td>1</td>
<td>Trunk _Port</td>
</tr>
<tr>
<td>Interface_3</td>
<td>TRUE</td>
<td>3</td>
<td>Access _Port</td>
</tr>
</tbody>
</table>
Step 4: In the INTERFACE (ETHERNET) > DATALINK LAYER Properties of L3 Switch 1, Click on "Configure VLAN" to view the properties for VLAN 2 set as per the screenshot shown below:

Properties for VLAN 3 is set as per the below screenshot:

After setting the properties of VLAN2 and VLAN3 click on OK.
Step 5: In the NETWORK LAYER Properties of L3 Switch 1, Click on “Configure Static Route IP” to set static route as per the screenshot shown below:

![Static Route IP Configuration](image)

Set the properties in Static Route IP window as per the screenshot below and click on Add.

Click on OK

![Static Route IP Table](image)

**NOTE**: Transport Protocol is set to UDP in Application properties.

Step 6: Run simulation for 10 seconds and observe the throughput.

30.4 Output:

<table>
<thead>
<tr>
<th>Throughput (Mbps)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Application 1</td>
</tr>
</tbody>
</table>
The above results conclude that Trunking allows us to send or receive any VLAN information across the network.
31. Understanding Public IP Address & NAT (Network Address Translation)

31.1 Theory:

31.1.1 Public Address:

A public IP address is assigned to every computer that connects to the Internet where each IP is unique. Hence there cannot exist two computers with the same public IP address all over the Internet. This addressing scheme makes it possible for the computers to “find each other” online and exchange information. User has no control over the IP address (public) that is assigned to the computer. The public IP address is assigned to the computer by the Internet Service Provider as soon as the computer is connected to the Internet gateway.

31.1.2 Private Address:

An IP address is considered private if the IP number falls within one of the IP address ranges reserved for private networks such as a Local Area Network (LAN). The Internet Assigned Numbers Authority (IANA) has reserved the following three blocks of the IP address space for private networks (local networks):

<table>
<thead>
<tr>
<th>Class</th>
<th>Starting IP address</th>
<th>Ending IP address</th>
<th>No. of hosts</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>10.0.0.0</td>
<td>10.255.255.255</td>
<td>16,777,216</td>
</tr>
<tr>
<td>B</td>
<td>172.16.0.0</td>
<td>172.31.255.255</td>
<td>1,048,576</td>
</tr>
<tr>
<td>C</td>
<td>192.168.0.0</td>
<td>192.168.255.255</td>
<td>65,536</td>
</tr>
</tbody>
</table>

Private IP addresses are used for numbering the computers in a private network including home, school and business LANs in airports and hotels which makes it possible for the computers in the network to communicate with each other. For example, if a network A consists of 30 computers each of them can be given an IP starting from 192.168.0.1 to 192.168.0.30.

Devices with private IP addresses cannot connect directly to the Internet. Likewise, computers outside the local network cannot connect directly to a device with a private IP. It is possible to interconnect two private networks with the help of a router or a similar device that supports Network Address Translation.

If the private network is connected to the Internet (through an Internet connection via ISP) then each computer will have a private IP as well as a public IP. Private IP is used for communication within the network whereas the public IP is used for communication over the Internet.
31.1.3 Network address translation (NAT):

A NAT (Network Address Translation or Network Address Translator) is the virtualization of Internet Protocol (IP) addresses. NAT helps to improve security and decrease the number of IP addresses an organization needs.

A device that is configured with NAT will have at least one interface to the inside network and one to the outside network. In a typical environment, NAT is configured at the exit device between a stub domain (inside network) and the backbone. When a packet leaves the domain, NAT translates the locally significant source address into a globally unique address. When a packet enters the domain, NAT translates the globally unique destination address into a local address. If more than one exit point exists, each NAT must have the same translation table. NAT can be configured to advertise to the outside world only one address for the entire network. This ability provides additional security by effectively hiding the entire internal network behind that one address. If NAT cannot allocate an address because it has run out of addresses, it drops the packet and sends an Internet Control Message Protocol (ICMP) host unreachable packet to the destination.

NAT is secure since it hides network from the Internet. All communications from internal private network are handled by the NAT device, which will ensure all the appropriate translations are performed and provide a flawless connection between internal devices and the Internet.

In the above figure, a simple network of 4 hosts and one router that connects this network to the Internet. All hosts in the network have a private Class C IP Address, including the router's private interface (192.168.0.1), while the public interface that's connected to the Internet has a real IP
Address (203.31.220.134). This is the IP address the Internet sees as all internal IP addresses are hidden.

31.2 Network Setup:

Open NetSim and click Examples > Experiments > Understanding-Public-IP-Address-and-NAT-(Network-Address-Translation) > Sample-1 as shown below:

NetSim UI displays the configuration file corresponding to this experiment as shown below:

31.3 Procedure:

The following set of procedures were done to generate this sample:
Step 1: A network scenario is designed in NetSim GUI comprising of 6 Wired Nodes, 2 L2 Switches, and 4 Routers in the “Internetworks” Network Library.

Step 2: In the INTERFACE (ETHERNET) > NETWORK LAYER of the Wired Nodes, the IP Address and the Subnet Mask are set as per the table given below:

<table>
<thead>
<tr>
<th>Wired Node</th>
<th>IP address</th>
<th>Subnet mask</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>10.0.0.2</td>
<td>255.0.0.0</td>
</tr>
<tr>
<td>8</td>
<td>10.0.0.3</td>
<td>255.0.0.0</td>
</tr>
<tr>
<td>9</td>
<td>10.0.0.4</td>
<td>255.0.0.0</td>
</tr>
<tr>
<td>10</td>
<td>172.16.0.2</td>
<td>255.255.0.0</td>
</tr>
<tr>
<td>11</td>
<td>172.16.0.3</td>
<td>255.255.0.0</td>
</tr>
<tr>
<td>12</td>
<td>172.16.0.4</td>
<td>255.255.0.0</td>
</tr>
</tbody>
</table>

Step 3: The IP Address and the Subnet Mask in Routers are set as per the table given below:

<table>
<thead>
<tr>
<th>Router</th>
<th>Interface</th>
<th>IP address</th>
<th>Subnet mask</th>
</tr>
</thead>
<tbody>
<tr>
<td>Router 1</td>
<td>Interface_2(WAN)</td>
<td>11.1.1.1</td>
<td>255.0.0.0</td>
</tr>
<tr>
<td></td>
<td>Interface_1(Ethernet)</td>
<td>10.0.0.1</td>
<td>255.0.0.0</td>
</tr>
<tr>
<td>Router 2</td>
<td>Interface_1(WAN)</td>
<td>11.1.1.2</td>
<td>255.0.0.0</td>
</tr>
<tr>
<td></td>
<td>Interface_2(WAN)</td>
<td>12.1.1.1</td>
<td>255.0.0.0</td>
</tr>
<tr>
<td>Router 3</td>
<td>Interface_1(WAN)</td>
<td>12.1.1.2</td>
<td>255.0.0.0</td>
</tr>
<tr>
<td></td>
<td>Interface_2(WAN)</td>
<td>13.1.1.2</td>
<td>255.0.0.0</td>
</tr>
<tr>
<td>Router 4</td>
<td>Interface_1(WAN)</td>
<td>13.1.1.1</td>
<td>255.0.0.0</td>
</tr>
<tr>
<td></td>
<td>Interface_2(Ethernet)</td>
<td>172.16.0.1</td>
<td>255.255.0.0</td>
</tr>
</tbody>
</table>

Step 4: Right click on the Application Flow App1 CBR and select Properties or click on the Application icon present in the top ribbon/toolbar.

A CBR Application is generated from Wired Node 7 i.e. Source to Wired Node 10 i.e. Destination with Packet Size remaining 1460Bytes and Inter Arrival Time remaining 20000µs.

Additionally, the “Start Time(s)” parameter is set to 50, while configuring the application. This time is usually set to be greater than the time taken for OSPF Convergence (i.e. Exchange of OSPF information between all the routers), and it increases as the size of the network increases.
Step 5: Packet Trace is enabled, and hence we are able to track the route which the packets have chosen to reach the destination.

Step 6: Run the Simulation for 100 Seconds.

31.4 Output:

After simulation open Packet Trace and filter Packet ID to 1.

<table>
<thead>
<tr>
<th></th>
<th>PACKET_ID</th>
<th>SEGMENT</th>
<th>SOURCE_IP</th>
<th>DESTINATION_IP</th>
<th>SOURCE_ID</th>
<th>DESTINATION_ID</th>
<th>SOURCE_IP</th>
<th>DESTINATION_IP</th>
<th>SOURCE_IP</th>
<th>DESTINATION_IP</th>
<th>SOURCE_IP</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>NODE-7</td>
<td>NODE-10</td>
<td>10.0.0.2</td>
<td>10.0.0.1</td>
<td>10.0.0.1</td>
<td>10.0.0.1</td>
<td>10.0.0.1</td>
<td>10.0.0.1</td>
<td>10.0.0.1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>NODE-7</td>
<td>NODE-10</td>
<td>10.0.0.2</td>
<td>10.0.0.1</td>
<td>10.0.0.1</td>
<td>10.0.0.1</td>
<td>10.0.0.1</td>
<td>10.0.0.1</td>
<td>10.0.0.1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>NODE-7</td>
<td>NODE-10</td>
<td>10.0.0.2</td>
<td>10.0.0.1</td>
<td>10.0.0.1</td>
<td>10.0.0.1</td>
<td>10.0.0.1</td>
<td>10.0.0.1</td>
<td>10.0.0.1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>NODE-7</td>
<td>NODE-10</td>
<td>10.0.0.2</td>
<td>10.0.0.1</td>
<td>10.0.0.1</td>
<td>10.0.0.1</td>
<td>10.0.0.1</td>
<td>10.0.0.1</td>
<td>10.0.0.1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>NODE-7</td>
<td>NODE-10</td>
<td>10.0.0.2</td>
<td>10.0.0.1</td>
<td>10.0.0.1</td>
<td>10.0.0.1</td>
<td>10.0.0.1</td>
<td>10.0.0.1</td>
<td>10.0.0.1</td>
</tr>
</tbody>
</table>

**SOURCE_IP** – source node IP (Node)

**DESTINATION_IP** – gateway IP (Router/ Node)

**GATEWAY_IP** – IP of the device which is transmitting a packet (Router/ Node)

**NEXT_HOP_IP** – IP of the next hop (Router/ Node)
Source node 7 (10.0.0.2) wouldn’t know how to route to the destination and hence its default gateway is Router 1 with interface IP (10.0.0.1). So, the first line in the above screenshot specifies packet flow from Source Node 7 to L2 Switch 6 with SOURCE_IP (10.0.0.2), DESTINATION_IP (10.0.0.1), GATEWAY_IP (10.0.0.2) and NEXT_HOP_IP (10.0.0.1). Since Switch is Layer2 device there is no change in the IPs in second line. Third line specifies the packet flow from Router 1 to Router 2 with SOURCE_IP (10.0.0.2), DESTINATION_IP (13.1.1.1- IP of the router connected to destination. Since OSPF is running, the router is looks up the route to its destination from routing table), GATEWAY_IP (11.1.1.1) and NEXT_HOP_IP (11.1.1.2) and so on.
32. Understand the events involved in NetSim DES (Discrete Event Simulator) in simulating the flow of one packet from a Wired node to a Wireless node

32.1 Theory

NetSim’s Network Stack forms the core of NetSim and its architectural aspects are diagrammatically explained below. Network Stack accepts inputs from the end-user in the form of Configuration file and the data flows as packets from one layer to another layer in the Network Stack. All packets, when transferred between devices move up and down the stack, and all events in NetSim fall under one of these ten categories of events, namely, Physical IN, Data Link IN, Network IN, Transport IN, Application IN, Application Out, Transport OUT, Network OUT, Data Link OUT and Physical OUT. The IN events occur when the packets are entering a device while the OUT events occur while the packet is leaving a device.

Every device in NetSim has an instance of the Network Stack shown above. Switches & Access points have a 2 layer stack, while routers have a 3 layer stack. End-nodes have a 5 layer stack.

The protocol engines are called based on the layer at which the protocols operate. For example, TCP is called during execution of Transport IN or Transport OUT events, while 802.11b WLAN is called during execution of MAC IN, MAC OUT, PHY IN and PHY OUT events.

When these protocols are in operation they in turn generate events for NetSim's discrete event engine to process. These are known as SUB EVENTS. All SUB EVENTS, fall into one of the above 10 types of EVENTS.
Each event gets added in the Simulation kernel by the protocol operating at the particular layer of
the Network Stack. The required sub events are passed into the Simulation kernel. These sub events
are then fetched by the Network Stack in order to execute the functionality of each protocol. At the
end of Simulation, Network Stack writes trace files and the Metrics files that assist the user in
analyzing the performance metrics and statistical analysis.

**Event Trace:**

The event trace records every single event along with associated information such as time stamp,
event ID, event type etc. in a text file or .csv file which can be stored at a user defined location.

### 32.2 Network Setup:

Open NetSim and click **Examples > Experiments > Advanced:Simulation-events-in-NetSim-for-
transmitting-one-packet > Sample-1** as shown below:

![NetSim UI](image)

NetSim UI displays the configuration file corresponding to this experiment as shown below:
32.3 Procedure:

The following set of procedures were done to generate this sample:

Step 1: A network scenario is designed in NetSim GUI comprising of 1 Wired Node, 1 Wireless Node, 1 Router, and 1 Access Point in the “Internetworks” Network Library.

Step 2: The device positions are set as per the below table:

<table>
<thead>
<tr>
<th>Device Positions</th>
<th>Access Point 2</th>
<th>Wired Node 4</th>
<th>Wireless Node 1</th>
<th>Router 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>X / Lon</td>
<td>150</td>
<td>250</td>
<td>100</td>
<td>200</td>
</tr>
<tr>
<td>Y / Lat</td>
<td>50</td>
<td>100</td>
<td>100</td>
<td>50</td>
</tr>
</tbody>
</table>

Step 3: Right-click the link ID (of the wireless link) and select Properties to access the link’s properties. The “Channel Characteristics” is set to NO PATHLOSS.

Step 4: Right click on the Application Flow App1 CBR and select Properties or click on the Application icon present in the top ribbon/toolbar.

A CBR Application is generated from Wired Node 4 i.e. Source to Wireless Node 1 i.e. Destination with Packet Size remaining 1460 Bytes and Inter Arrival Time remaining 20000µs.

Transport Protocol is set to UDP instead of TCP.

Step 5: Event Trace is enabled in NetSim GUI. At the end of the simulation, a very large .csv file is containing all the TCP IN and OUT EVENTS is available for the users.

Note: Event trace is only available only in NetSim Standard and Pro versions.

32.4 Output

Once the simulation is complete, go to the Results Dashboard and in the left-hand-side of the window, click on the ”Open Event Trace” Option. An Event trace file similar to the following opens in Excel as shown below:
We start from the **APPLICATION_OUT** event of the first packet, which happens in the Wired Node and end with the **MAC_IN** event of the **WLAN_ACK** packet which reaches the Wired Node. Events in the event trace are logged with respect to the time of occurrence due to which, event id may not be in order.

### 32.4.1 Events Involved:

Events are listed in the following format:

```
[EVENT_TYPE, EVENT_TIME, PROTOCOL, EVENT_NO, SUBEVENT_TYPE]
```

<table>
<thead>
<tr>
<th>EVENT_TYPE</th>
<th>EVENT_TIME</th>
<th>PROTOCOL</th>
<th>EVENT_NO</th>
<th>SUBEVENT_TYPE</th>
</tr>
</thead>
<tbody>
<tr>
<td>APP_OUT</td>
<td>20000</td>
<td>APP</td>
<td>6</td>
<td>-</td>
</tr>
<tr>
<td>TRNS_OUT</td>
<td>20000</td>
<td>UDP</td>
<td>7</td>
<td>-</td>
</tr>
<tr>
<td>NW_OUT</td>
<td>20000</td>
<td>IPV4</td>
<td>9</td>
<td>-</td>
</tr>
<tr>
<td>MAC_OUT</td>
<td>20000</td>
<td>ETH</td>
<td>10</td>
<td>-</td>
</tr>
<tr>
<td>MAC_OUT</td>
<td>20000</td>
<td>ETH</td>
<td>11</td>
<td>CS</td>
</tr>
<tr>
<td>MAC_OUT</td>
<td>20000.96</td>
<td>ETH</td>
<td>12</td>
<td>IFG</td>
</tr>
<tr>
<td>PHY_OUT</td>
<td>20000.96</td>
<td>ETH</td>
<td>13</td>
<td>-</td>
</tr>
<tr>
<td>PHY_OUT</td>
<td>20122.08</td>
<td>ETH</td>
<td>14</td>
<td>PHY_SENSE</td>
</tr>
<tr>
<td>PHY_IN</td>
<td>20127.08</td>
<td>ETH</td>
<td>15</td>
<td>-</td>
</tr>
<tr>
<td>MAC_IN</td>
<td>20127.08</td>
<td>ETH</td>
<td>16</td>
<td>-</td>
</tr>
<tr>
<td>NW_IN</td>
<td>20127.08</td>
<td>IPV4</td>
<td>17</td>
<td>-</td>
</tr>
<tr>
<td>NW_OUT</td>
<td>20127.08</td>
<td>IPV4</td>
<td>18</td>
<td>-</td>
</tr>
<tr>
<td>MAC_OUT</td>
<td>20127.08</td>
<td>ETH</td>
<td>19</td>
<td>-</td>
</tr>
<tr>
<td>Event Type</td>
<td>Timestamp</td>
<td>Interface</td>
<td>Sequence</td>
<td>Message</td>
</tr>
<tr>
<td>-------------</td>
<td>-----------</td>
<td>-----------</td>
<td>----------</td>
<td>----------------</td>
</tr>
<tr>
<td>MAC_OUT</td>
<td>20127.08</td>
<td>ETH</td>
<td>20</td>
<td>CS</td>
</tr>
<tr>
<td>MAC_OUT</td>
<td>20128.04</td>
<td>ETH</td>
<td>21</td>
<td>IFG</td>
</tr>
<tr>
<td>PHY_OUT</td>
<td>20128.04</td>
<td>ETH</td>
<td>22</td>
<td></td>
</tr>
<tr>
<td>PHY_OUT</td>
<td>20249.16</td>
<td>ETH</td>
<td>23</td>
<td>PHY_SENSE</td>
</tr>
<tr>
<td>PHY_IN</td>
<td>20254.16</td>
<td>ETH</td>
<td>24</td>
<td></td>
</tr>
<tr>
<td>MAC_IN</td>
<td>20254.16</td>
<td>ETH</td>
<td>25</td>
<td></td>
</tr>
<tr>
<td>MAC_OUT</td>
<td>20254.16</td>
<td>WLAN</td>
<td>26</td>
<td></td>
</tr>
<tr>
<td>MAC_OUT</td>
<td>20254.16</td>
<td>WLAN</td>
<td>27</td>
<td>DIFS_END</td>
</tr>
<tr>
<td>MAC_OUT</td>
<td>20304.16</td>
<td>WLAN</td>
<td>28</td>
<td>BACKOFF</td>
</tr>
<tr>
<td>MAC_OUT</td>
<td>20324.16</td>
<td>WLAN</td>
<td>29</td>
<td>BACKOFF</td>
</tr>
<tr>
<td>MAC_OUT</td>
<td>20344.16</td>
<td>WLAN</td>
<td>30</td>
<td>BACKOFF</td>
</tr>
<tr>
<td>MAC_OUT</td>
<td>20364.16</td>
<td>WLAN</td>
<td>31</td>
<td>BACKOFF</td>
</tr>
<tr>
<td>PHY_OUT</td>
<td>20364.16</td>
<td>WLAN</td>
<td>32</td>
<td></td>
</tr>
<tr>
<td>TIMER</td>
<td>21668.16</td>
<td>WLAN</td>
<td>35</td>
<td>UPDATE_DEVICE_STATUS</td>
</tr>
<tr>
<td>PHY_IN</td>
<td>21668.4</td>
<td>WLAN</td>
<td>33</td>
<td></td>
</tr>
<tr>
<td>MAC_IN</td>
<td>21668.4</td>
<td>WLAN</td>
<td>36</td>
<td>RECEIVE_MPDU</td>
</tr>
<tr>
<td>NW_IN</td>
<td>21668.4</td>
<td>IPV4</td>
<td>37</td>
<td></td>
</tr>
<tr>
<td>MAC_OUT</td>
<td>21668.4</td>
<td>WLAN</td>
<td>38</td>
<td>SEND_ACK</td>
</tr>
<tr>
<td>TRNS_IN</td>
<td>21668.4</td>
<td>UDP</td>
<td>39</td>
<td></td>
</tr>
<tr>
<td>APP_IN</td>
<td>21668.4</td>
<td>APP</td>
<td>41</td>
<td></td>
</tr>
<tr>
<td>PHY_OUT</td>
<td>21678.4</td>
<td>WLAN</td>
<td>40</td>
<td></td>
</tr>
<tr>
<td>TIMER</td>
<td>21982.4</td>
<td>WLAN</td>
<td>43</td>
<td>UPDATE_DEVICE</td>
</tr>
<tr>
<td>PHY_IN</td>
<td>21982.63</td>
<td>WLAN</td>
<td>42</td>
<td></td>
</tr>
</tbody>
</table>
[MAC_IN, 21982.63, WLAN, 44, RECEIVE_ACK]

[_TIMER, 21985, WLAN, 34, ACK_TIMEOUT]

Event Flow Diagram for one packet from Wired Node to Wireless Node:

For Example:

MAC_OUT in the Access Point involves sub events like CS, DIFS_END and BACKOFF.
As you can see in the trace file shown below, CS happens at event time 20254.16,

Adding DIFS time of 50µs to this will give DIFS_END sub event at 20304.16. Further it is followed by three Backoff's each of 20 µs, at event time 20314.16, 20324.16, 20344.16 respectively.

In this manner the event trace can be used to understand the flow of events in NetSim Discrete Event Simulator.

32.5 Inference

In NetSim each event occurs at a particular instant in time and marks a change of state in the system. Between consecutive events, no change in the system is assumed to occur. Thus the simulation can directly jump in time from one event to the next.

This contrasts with continuous simulation in which the simulation continuously tracks the system dynamics over time. Because discrete-event simulations do not have to simulate every time slice, they can typically run much faster than the corresponding continuous simulation.

Understanding NetSim's Event trace and its flow is very much helpful especially when customizing existing code and debugging to verify the correctness the modified code. The event IDs provided in the event trace can be used to go to a specific event while debugging.
33. Understand the working of TCP BIC Congestion control algorithm, simulate and plot the TCP congestion window

33.1 Theory

In BIC congestion control is viewed as a searching problem in which the system can give yes/no feedback through packet loss as to whether the current sending rate (or window) is larger than the network capacity. The current minimum window can be estimated as the window size at which the flow does not see any packet loss. If the maximum window size is known, we can apply a binary search technique to set the target window size to the midpoint of the maximum and minimum. As increasing to the target, if it gives any packet loss, the current window can be treated as a new maximum and the reduced window size after the packet loss can be the new minimum. The midpoint between these new values becomes a new target. Since the network incurs loss around the new maximum but did not do so around the new minimum, the target window size must be in the middle of the two values. After reaching the target and if it gives no packet loss, then the current window size becomes a new minimum, and a new target is calculated. This process is repeated with the updated minimum and maximum until the difference between the maximum and the minimum falls below a preset threshold, called the minimum increment ($S_{\text{min}}$). This technique is called binary search increase.

Additive Increase:

In order to ensure faster convergence and RTT-fairness, binary search increase is combined with an additive increase strategy. When the distance to the midpoint from the current minimum is too large, increasing the window size directly to that midpoint might add too much stress to the network. When the distance from the current window size to the target in binary search increase is larger than a prescribed maximum step, called the maximum increment ($S_{\text{max}}$) instead of increasing window directly to that midpoint in the next RTT, we increase it by $S_{\text{max}}$ until the distance becomes less than $S_{\text{max}}$, at which time window increases directly to the target. Thus, after a large window reduction, the strategy initially increases the window linearly, and then increases logarithmically. This combination of binary search increase and additive increase is called as binary increase. Combined with a multiplicative decrease strategy, binary increase becomes close to pure additive increase under large windows. This is because a larger window results in a larger reduction by multiplicative decrease and therefore, a longer additive increase period. When the window size is small, it becomes close to pure binary search increase – a shorter additive increase period.

Slow Start:
After the window grows past the current maximum, the maximum is unknown. At this time, binary search sets its maximum to be a default maximum (a large constant) and the current window size to be the minimum. So, the target midpoint can be very far. According to binary increase, if the target midpoint is very large, it increases linearly by the maximum increment. Instead, run a “slow start” strategy to probe for a new maximum up to Smax. So if cwnd is the current window and the maximum increment is Smax, then it increases in each RTT round in steps \(cwnd+1, cwnd+2, cwnd+4, \ldots\), \(cwnd+Smax\). The rationale is that since it is likely to be at the saturation point and also the maximum is unknown, it probes for available bandwidth in a “slow start” until it is safe to increase the window by Smax. After slow start, it switches to binary increase.

**Fast Convergence:**

It can be shown that under a completely synchronized loss model, binary search increase combined with multiplicative decrease converges to a fair share. Suppose there are two flows with different window sizes, but with the same RTT. Since the larger window reduces more in multiplicative decrease (with a fixed factor \(\beta\)), the time to reach the target is longer for a larger window. However, its convergence time can be very long. In binary search increase, it takes \(\log(d) - \log(Smin)\) RTT rounds to reach the maximum window after a window reduction of \(d\). Since the window increases in a log step, the larger window and smaller window can reach back to their respective maxima very fast almost at the same time (although the smaller window flow gets to its maximum slightly faster). Thus, the smaller window flow ends up taking away only a small amount of bandwidth from the larger flow before the next window reduction. To remedy this behaviour, binary search increase is modified as follows. After a window reduction, new maximum and minimum are set. Suppose these values are \(\text{max}_\text{wini}\) and \(\text{min}_\text{wini}\) for flow \(i\) (\(i = 1, 2\)). If the new maximum is less than the previous, this window is in a downward trend. Then, readjust the new maximum to be the same as the new target window (i.e. \(\text{max}_\text{wini} = \frac{(\text{max}_\text{wini} - \text{min}_\text{wini})}{2}\)), and then readjust the target. After that apply the normal binary increase. This strategy is called fast convergence.

**33.2 Network setup:**

Open NetSim and click **Examples > Experiments > Advanced:TCP-BIC-Congestion-control-algorithm > Sample-1** as shown below:
NetSim UI displays the configuration file corresponding to this experiment as shown below:

![NetSim UI Configuration](image)

### 33.3 Procedure:

The following set of procedures were done to generate this sample:

**Step 1:** A network scenario is designed in NetSim GUI comprising of 2 Wired Nodes and 2 Routers in the “Internetworks” Network Library.

**Step 2:** In the General Properties of Wired Node 3 i.e. Source, Wireshark Capture is set to Online and in the TRANSPORT LAYER Properties, Window Scaling is set as TRUE.

**Step 3:** For all the devices, in the TRANSPORT LAYER Properties, Congestion Control Algorithm is set to BIC.
Step 4: The Link Properties are set according to the table given below:

<table>
<thead>
<tr>
<th>Link Properties</th>
<th>Wired Link 1</th>
<th>Wired Link 2</th>
<th>Wired Link 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Uplink Speed (Mbps)</td>
<td>100</td>
<td>20</td>
<td>100</td>
</tr>
<tr>
<td>Downlink Speed (Mbps)</td>
<td>100</td>
<td>20</td>
<td>100</td>
</tr>
<tr>
<td>Uplink propagation delay (µs)</td>
<td>5</td>
<td>1000</td>
<td>5</td>
</tr>
<tr>
<td>Downlink propagation delay (µs)</td>
<td>5</td>
<td>1000</td>
<td>5</td>
</tr>
<tr>
<td>Uplink BER</td>
<td>0.00000001</td>
<td>0.00000001</td>
<td>0.00000001</td>
</tr>
<tr>
<td>Downlink BER</td>
<td>0.00000001</td>
<td>0.00000001</td>
<td>0.00000001</td>
</tr>
</tbody>
</table>

Step 5: Right click on the Application Flow App1 CBR and select Properties or click on the Application icon present in the top ribbon/toolbar.

A CBR Application is generated from Wired Node 3 i.e. Source to Wired Node 4 i.e. Destination with Packet Size set to 1460 Bytes and Inter Arrival Time set to 400 µs. Additionally, the “Start Time” parameter is set to 20 Seconds.

The Packet Size and Inter Arrival Time parameters are set such that the Generation Rate equals 140 Kbps. Generation Rate can be calculated using the formula:

$$\text{Generation Rate (Mbps)} = \text{Packet Size (Bytes)} * \frac{8}{\text{Interarrival time (µs)}}$$

Step 6: Click on Run simulation. The simulation time is set to 100 seconds.
33.4 Output:

Go to the Wireshark Capture window.

Click on data packet i.e. <None>. Go to Statistics → TCP Stream Graphs → Window Scaling.

Click on Switch Direction in the window scaling graph window to view the graph.

(For more guidance, refer to section - 8.7.5 Window Scaling” in user manual)

The graph shown above is a plot of Congestion Window vs Time of BIC for the scenario shown above. Each point on the graph represents the congestion window at the time when the packet is sent. You can observe Binary Search, Additive Increase, Fast Convergence, Slow Start phases in the above graph.
34. Simulating Link Failure

34.1 Objective

To understand the working of Link Failure.

34.2 Theory:

Link failures are a major threat that occur within the network topology. Probably, link failures occur due to low converging time, previously allocated delay and bandwidth, and iterative loops which degrade the performance of the network. So, the route to a destination may indeed become unavailable, when a failure occurs and the routing protocol has to recompute an alternate path around the failure. It affects the packet delivery and creates packet loss.

Users can find the settings for link failure in NetSim by a right click on the link between 2 routers and select properties. It displays the Link Properties Window as shown below:

![Link Properties Window](image)

Link Up Time refers to the time at which the link goes up and Link Down Time refers to the time at which a link goes down.

*NOTE: Link failure can be set only for “WAN Interfaces”.*
34.3 Network Setup:

Open NetSim and click **Examples > Experiments > Advanced:Simulating-Link-Failure > Sample-1** as shown below:

NetSim UI displays the configuration file corresponding to this experiment as shown below:

34.4 Procedure:

The following set of procedures were done to generate this sample:
Step 1: A network scenario is designed in NetSim GUI comprising of 2 Wired Nodes and 7 Routers in the “Internetworks” Network Library.

Step 2: By default, Link Failure Up Time is set to 0 and Down Time is set to 100000.

Step 3: Packet Trace is enabled in NetSim GUI. At the end of the simulation, a very large .csv file is containing all the packet information is available for the users to perform packet level analysis.

Step 4: Right click on the Application Flow App1 CBR and select Properties or click on the Application icon present in the top ribbon/toolbar.

A CBR Application is generated from Wired Node 1 i.e. Source to Wired Node 2 i.e. Destination with Packet Size remaining 1460 Bytes and Inter Arrival Time remaining 20000µs.

Additionally, the “Start Time(s)” parameter is set to 30, while configuring the application. This time is usually set to be greater than the time taken for OSPF Convergence (i.e. Exchange of OSPF information between all the routers), and it increases as the size of the network increases.

Step 5: Run the simulation for 80 Seconds.

> Sample-2:

The following changes in settings are done from the previous sample:

Step 1: In Link 3 Properties, Link Failure Up Time is set to 0 and Down Time is set to 50.

This means that the link would fail at 50 Seconds.

Step 2: Run the simulation for 80 Seconds.

34.5 Output:

In Sample 1,

Go to NetSim Packet Animation Window, click on Play button. We can notice the following:

- Initially OSPF Control Packets are exchanged between all the routers.
- Once after the exchange of control packets, the data packets are sent from the source to the destination.
- The packets are routed to the Destination via, N1 > R3 > R4 > R5 > R9 > N2 as shown below:
In Sample2,

- We create a Link Failure in Link 3, between Router 4 and Router 5 at 50 Seconds.
- Hence the packets are not able to reach the destination. The routing protocol then recomputes an alternate path to the Destination.
- This can be observed in the Packet Trace.
- Go to the Results Dashboard and click on Open Packet Trace option present in the Left-Hand-Side of the window and do the following:
  - Filter Control Packet Type/App Name to APP1 CBR and Transmitter ID to Router 3.
  - We can notice that packets are changing its route from, 
    \[ N1 \rightarrow R3 \rightarrow R4 \rightarrow R5 \rightarrow R9 \rightarrow N2 \]
    to 
    \[ N1 \rightarrow R3 \rightarrow R6 \rightarrow R7 \rightarrow R8 \rightarrow R9 \rightarrow N2 \]
    at 50 Seconds of simulation time, since the link between R4 and R5 is fails at 50 Seconds.
Users can also observe this in Packet animation before and after the Link Failure as shown below: