NetSim™

Simulation Platform for Network R & D

Experiments Manual

TETCOS

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

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. Understand the working of "Connection Establishment" in TCP 26

5. Appreciate the mathematical modelling of TCP and understand the fundamental relationship between packet loss probability and TCP performance. 30

6. Study how throughput and error of a Wireless LAN network changes as the distance between the Access Point and the wireless nodes is varied 36

7. How many downloads can a Wi-Fi access point simultaneously handle?.....43

8. Understand the working of Slow start and Congestion Avoidance (Old Tahoe), Fast Retransmit (Tahoe) and Fast Recovery (Reno) Congestion Control Algorithms in TCP. 50

9. Multi-AP Wi-Fi Networks: Channel Allocation.....56

10. Plot the characteristic curve of throughput versus offered traffic for a Pure and Slotted ALOHA system 64

11. 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 71

12. Study the working and routing table formation of Interior routing protocols, i.e. Routing Information Protocol (RIP) and Open Shortest Path First (OSPF).....78

13. M/D/1 Queuing 87

14. Quality of Service (QoS) in 802.11e based WLANs 93

15. Study the hidden node problem in WLAN.....97

16. Analyze the performance of FIFO, Priority and WFQ Queuing Disciplines....101

17. Study how call blocking probability varies as the load on a GSM network is continuously increased 105

18. Study the 802.15.4 Superframe Structure and analyze the effect of Superframe order on throughput.....109

19. Understand the working of OSPF 114

20. To analyze how the allocation of frequency spectrum to the Incumbent (Primary) and CR CPE (Secondary User) affects throughput.....124

21. Study how the throughput of LTE network varies as the distance between the ENB and UE (User Equipment) is increased 130

22. Study how the throughput of LTE network varies as the Channel bandwidth changes in the ENB (Evolved node) 135

23. Simulate and study LTE Handover procedure 139

24. Understand the working of LTE Device to Device Communication 146

25. Introduction and working of Internet of Things (IoT) 151

26. Understand the working of TCP BIC Congestion control algorithm, simulate and plot the TCP congestion window 157

27. Understanding VLAN operation in L2 and L3 Switches.....162

28. Understanding Access and Trunk Links in VLANs 170

29. Understanding Public IP Address & NAT (Network Address Translation).....176

30. Understand the working of basic networking commands (Ping, Route Add/Delete/Print, ACL) 182

31. Analyze the throughput of an IOT network where a wireless (802.15.4) sensor transmits data to a cloud server 192

32. WiFi: UDP Download Throughput.....197

33. Simulating Link Failure 207

34. IoT – Multi-Hop Sensor-Sink Path.....212

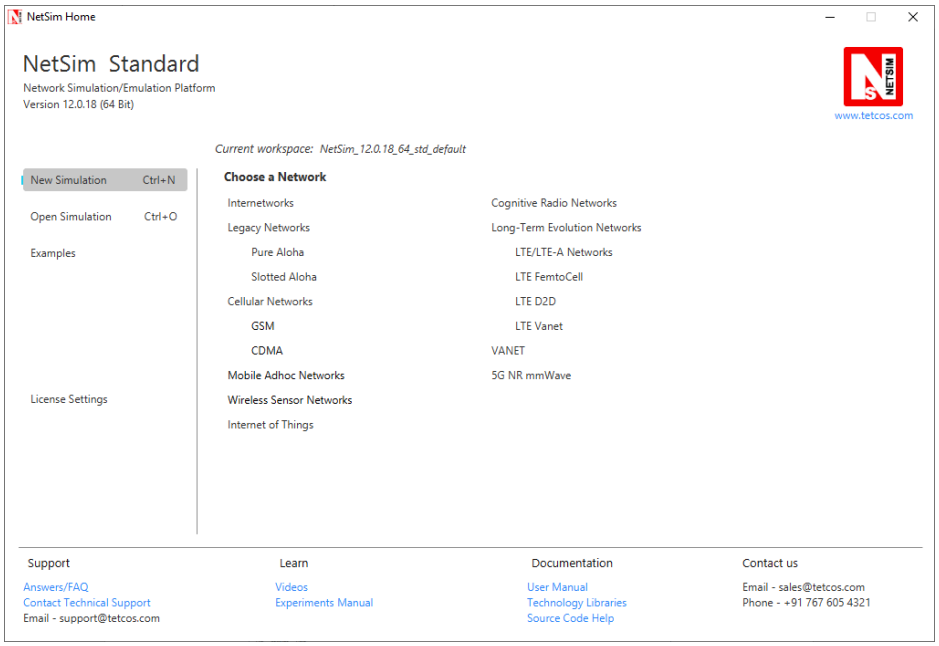
1. Introduction to NetSim

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:

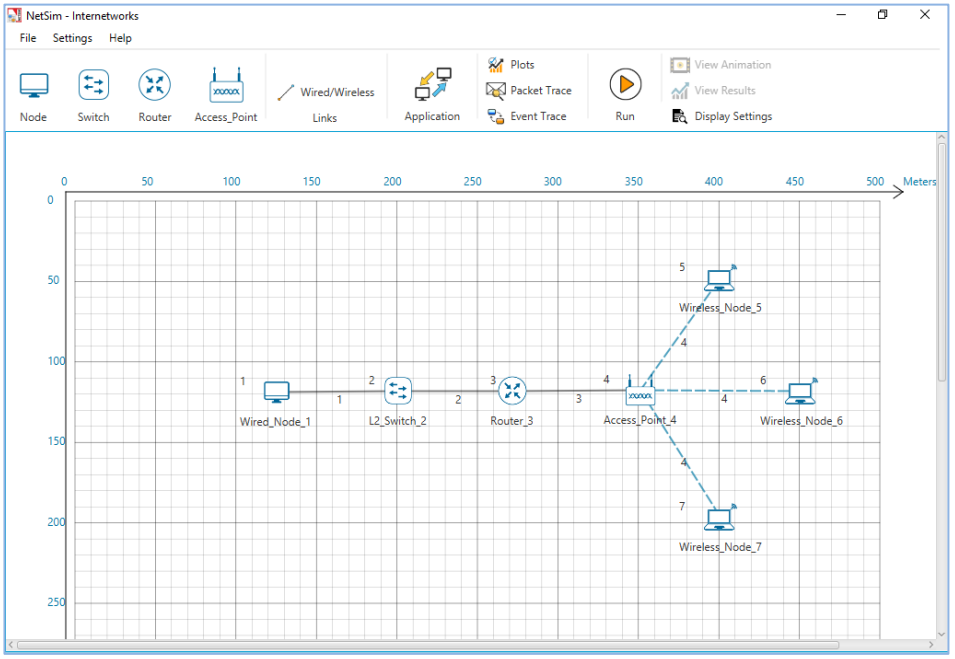
- Internetworks - Ethernet, WLAN, IP, TCP
- Legacy Networks - Aloha, Slotted Aloha
- Cellular Networks - GSM, CDMA
- Mobile Adhoc Networks - DSR, AODV, OLSR, ZRP
- Wireless Sensor Networks - 802.15.4
- Internet of Things - 6LoWPAN gateway, 802.15.4 MAC / PHY, RPL
- Cognitive Radio Networks - 802.22
- 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 → Environmental Settings and choose the type of environment. Here we have chosen the Environment 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.

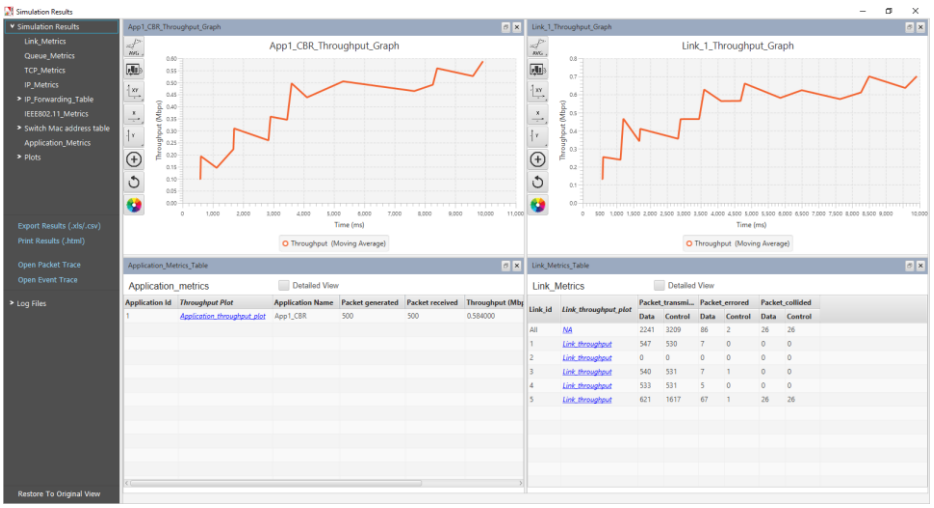
- **Answers/FAQ** – Assists the user by directing them to our **"NetSim Support Portal"**, where one can find a well-structured **"Knowledge Base"**, consisting of answers or solutions to all the commonest queries which a new user can go through.
- **Raise a Support Ticket** – Assists the user by directing 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.

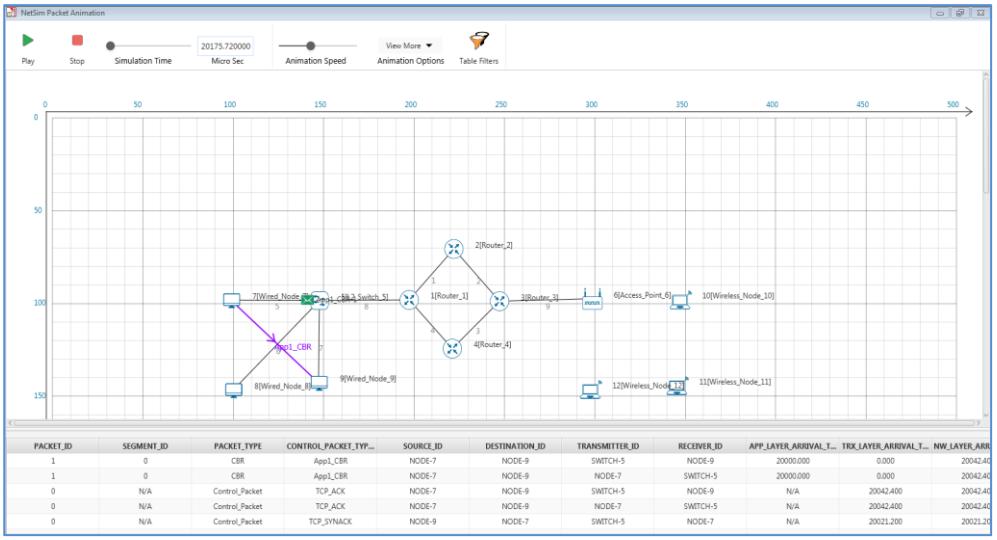
- **Results Window:** Upon completion of simulation, Network statistics or network performance metrics reported in the form of graphs and tables. The report includes metrics like throughput, simulation time, packets generated, packets dropped, collision counts etc.



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.

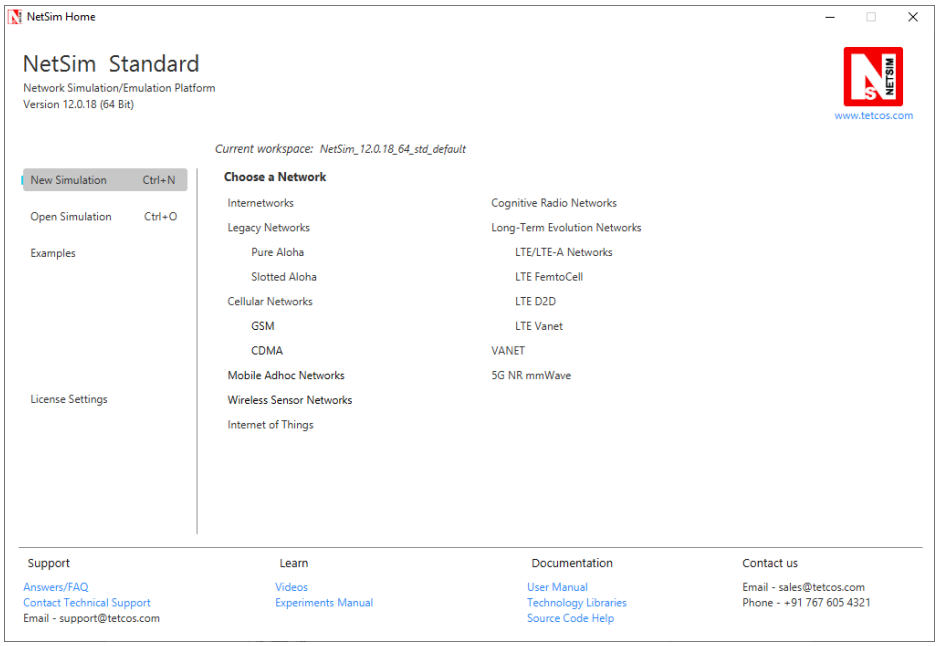


Description:

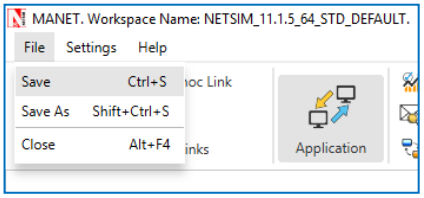
1. Click on Play to view the animation. You can Pause the animation at any interval and Play again.
2. Click on Stop to stop the animation. Now click on Play to start the animation from the beginning.
3. Next to that we also have speed controllers to increase/decrease Simulation Time and Animation Speed
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?

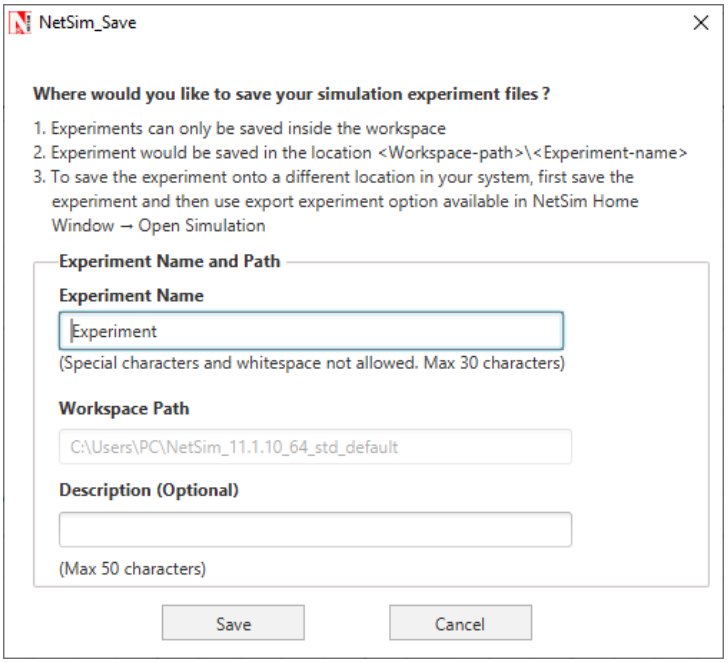
To create an experiment, select New Simulation-> <Any Network> in the NetSim Home Screen.



Create a network and save the experiment by clicking on File->Save button on the top left.

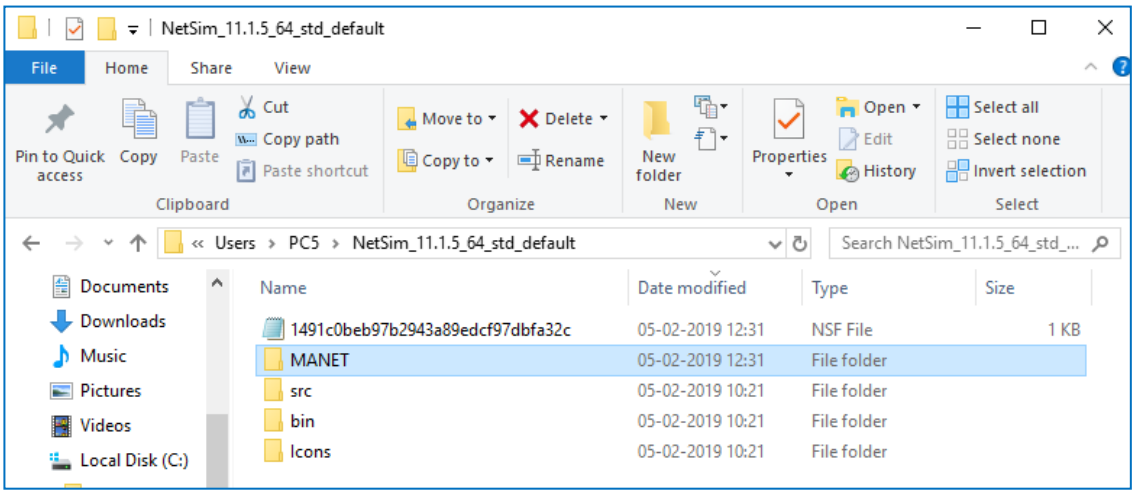


A save popup window appears which contains Experiment Name, Folder Name, Workspace path and Description.

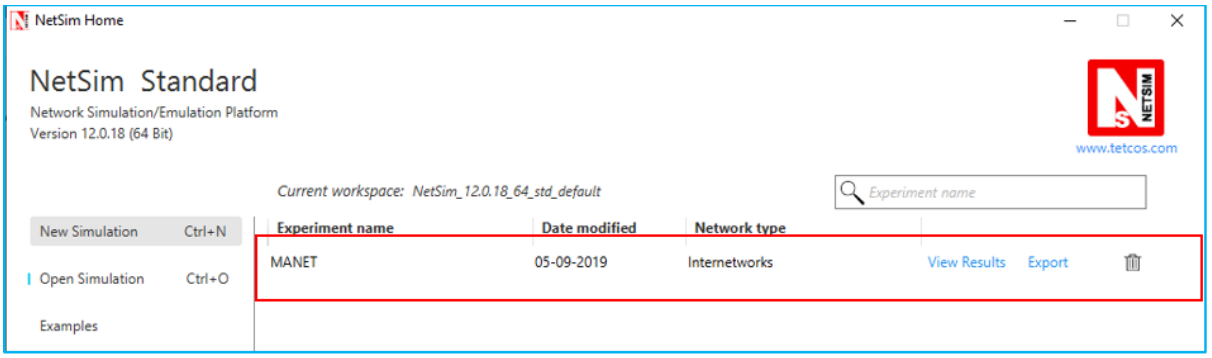


Specify the Experiment Name and Description (Optional) and then click on Save. The workspace path is non-editable. Hence all the experiments will be saved in the default workspace path. After specifying the Experiment Name click on Save.

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>
<tr><td colspan="2">Hardware Type</td><td>Protocol Type</td></tr>
<tr><td>Hardware Address Length</td><td>Protocol address length</td><td>Opcode</td></tr>
<tr><td colspan="3">Sender Hardware Address</td></tr>
<tr><td colspan="2">Sender Protocol Address(1-2)</td><td>Sender Protocol Address(3-4)</td></tr>
<tr><td colspan="3">Target hardware Address</td></tr>
<tr><td colspan="3">Target Protocol Address</td></tr>
</table>

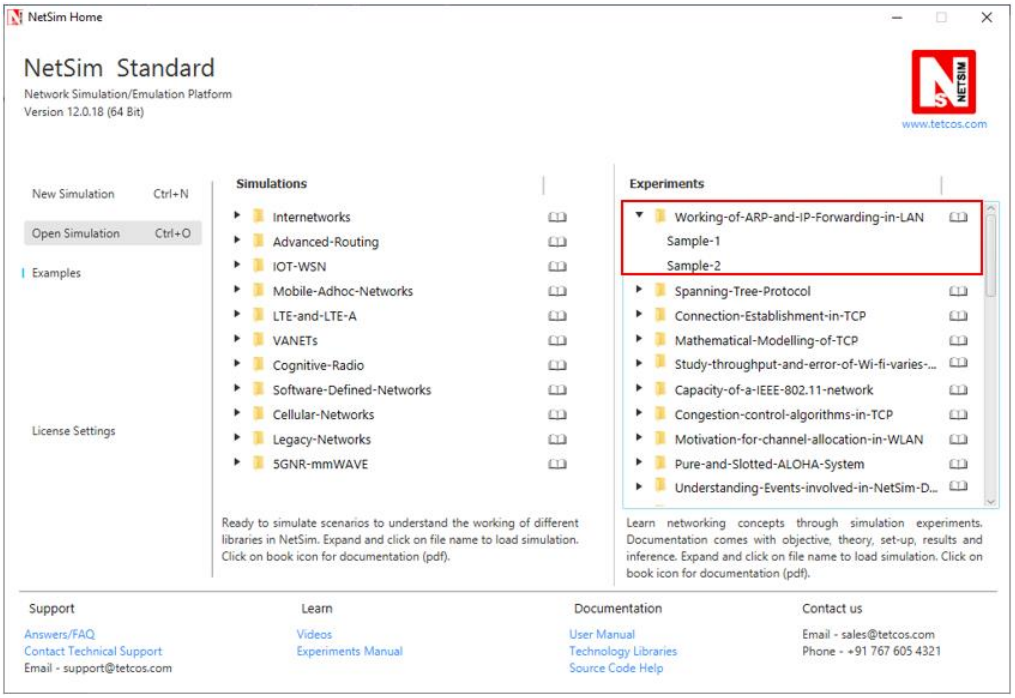
The ARP message format is designed to accommodate layer two and layer three addresses of various sizes. This diagram shows the most common implementation, which uses 32 bits for the layer three ("Protocol") addresses, and 48 bits for the layer two hardware addresses.

2.4 IP Forwarding Description

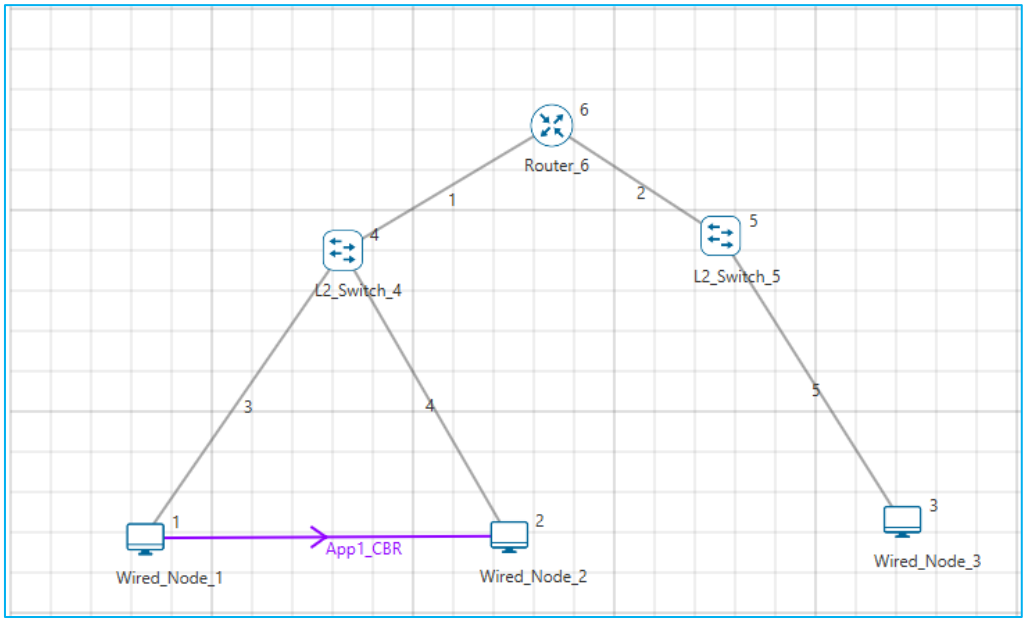
1. Every router has a forwarding table that maps the destination addresses (or portions of the destination addresses) to that router's outbound links.
2. A router forwards a packet by examining the value of a field in the arriving packet's header, and then using this header value to index into the router's 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-in-LAN > 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: TCP Protocol in the TRANSPORT LAYER Properties is disabled in all the Wired Nodes.

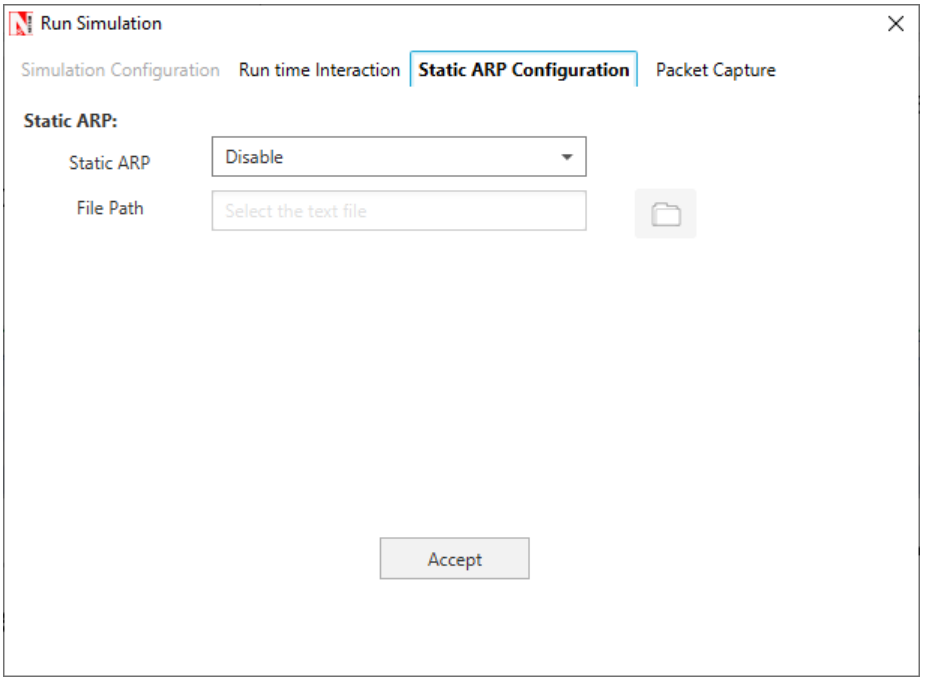
If TCP is enabled, the ARP table will get updated due to the transmission of TCP control packets thereby eliminating the need for ARP to resolve addresses.

Step 3: 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.

Step 4: 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 5: 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 automatically creates 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.

PACKET_ID	SEGMENT_ID	PACKET_TYPE	CONTROL_PACKET_TYPE/APP_NAME	SOURCE_ID	DESTINATION_ID	TRANSMITTER_ID	RECEIVER_ID
0	N/A	Control_Packet	ARP_Request	NODE-1	Broadcast-0	NODE-1	SWITCH-4
0	N/A	Control_Packet	ARP_Request	NODE-1	Broadcast-0	SWITCH-4	ROUTER-6
0	N/A	Control_Packet	ARP_Request	NODE-1	Broadcast-0	SWITCH-4	NODE-2
0	N/A	Control_Packet	ARP_Reply	NODE-2	NODE-1	NODE-2	SWITCH-4
0	N/A	Control_Packet	ARP_Reply	NODE-2	NODE-1	SWITCH-4	NODE-1

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:

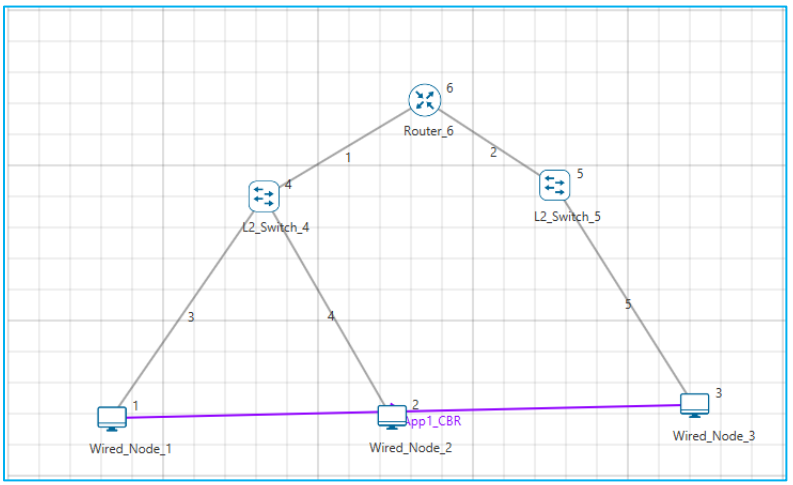
Intra-LAN-IP-forwarding:

ARP PROTOCOL- WORKING:

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:



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: TCP Protocol in the TRANSPORT LAYER Properties is disabled in all the Wired Nodes.

Step 3: 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.

Step 4: 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.

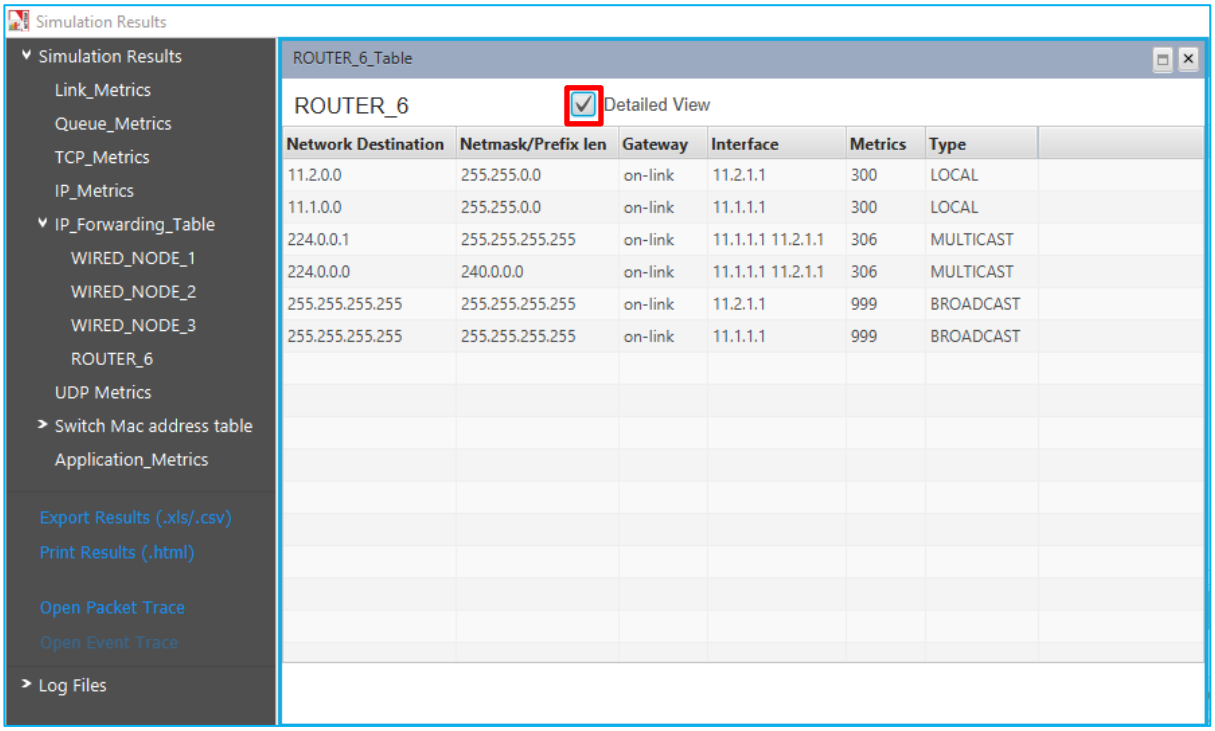
Step 5: Click on Run simulation. The simulation time is set to 10 seconds. In the **"Static ARP Configuration"** tab, Static ARP is set to disable.

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.

PACKET_ID	SEGMENT_ID	PACKET_TYPE	CONTROL_PACKET_TYPE/APP_NAME	SOURCE_ID	DESTINATION_ID	TRANSMITTER_ID	RECEIVER_ID
0	N/A	Control_Packet	ARP_Request	NODE-1	Broadcast-0	NODE-1	SWITCH-4
0	N/A	Control_Packet	ARP_Request	NODE-1	Broadcast-0	SWITCH-4	ROUTER-6
0	N/A	Control_Packet	ARP_Request	NODE-1	Broadcast-0	SWITCH-4	NODE-2
0	N/A	Control_Packet	ARP_Reply	ROUTER-6	NODE-1	ROUTER-6	SWITCH-4
0	N/A	Control_Packet	ARP_Reply	ROUTER-6	NODE-1	SWITCH-4	NODE-1
1	0	CBR	App1_CBR	NODE-1	NODE-3	NODE-1	SWITCH-4
1	0	CBR	App1_CBR	NODE-1	NODE-3	SWITCH-4	ROUTER-6
0	N/A	Control_Packet	ARP_Request	ROUTER-6	Broadcast-0	ROUTER-6	SWITCH-5
0	N/A	Control_Packet	ARP_Request	ROUTER-6	Broadcast-0	SWITCH-5	NODE-3
0	N/A	Control_Packet	ARP_Reply	NODE-3	ROUTER-6	NODE-3	SWITCH-5
0	N/A	Control_Packet	ARP_Reply	NODE-3	ROUTER-6	SWITCH-5	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:



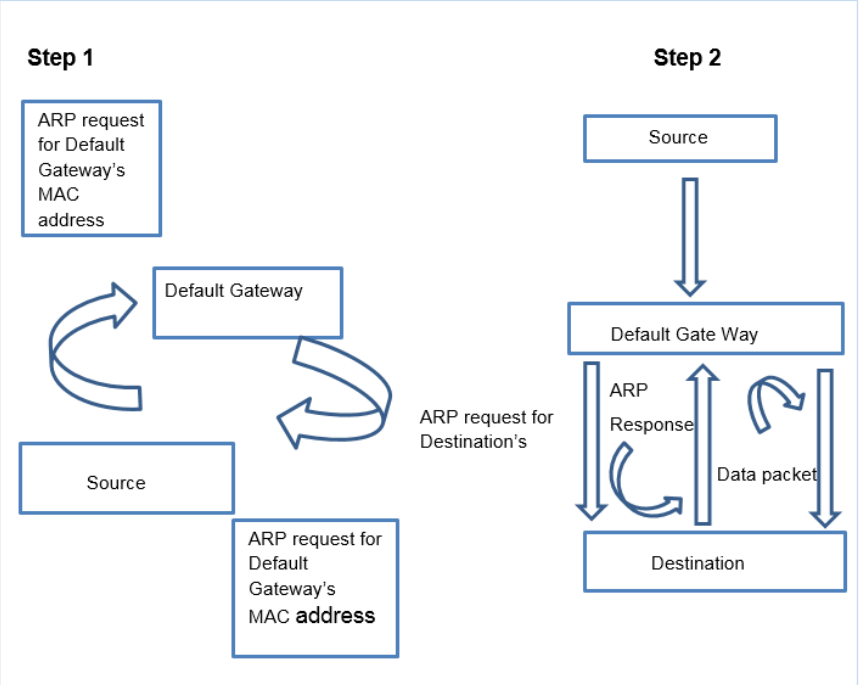
Click on Detailed View checkbox to view the additional fields as indicated above.

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

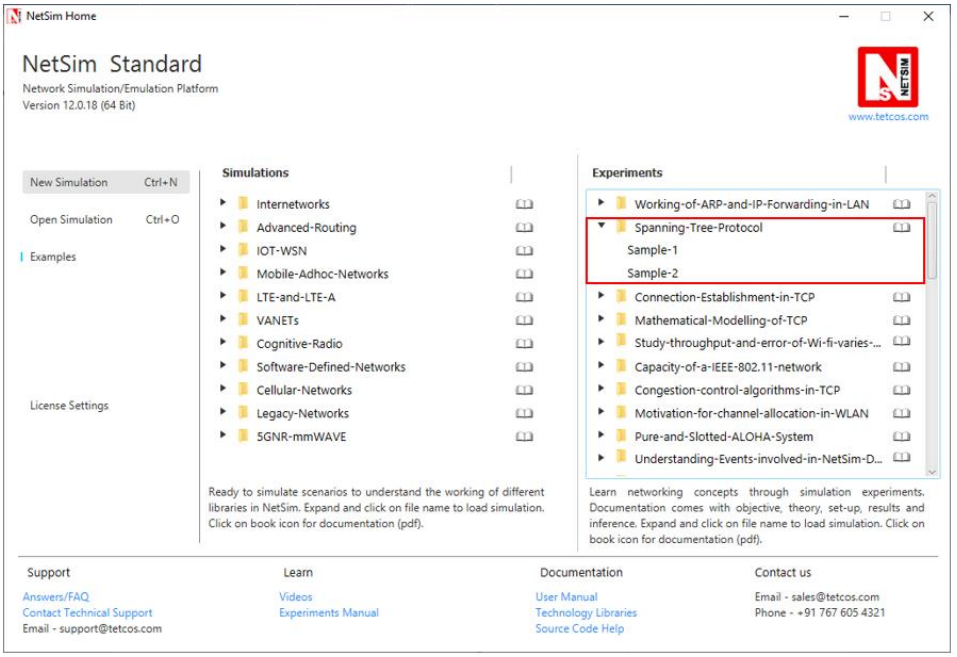
3.1 Introduction:

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.

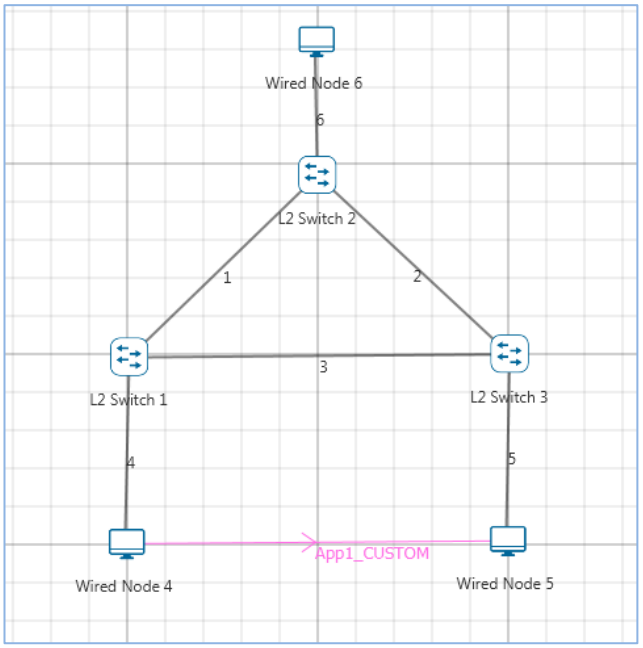
(***Reference:*** *A good reference for this topic is Section 3.1.4: Bridges and LAN switches, of the book, Computer Networks, 5th Edition by Peterson and Davie)*

3.2 Network Setup:

Open NetSim and click **Examples > Experiments > 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.

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.

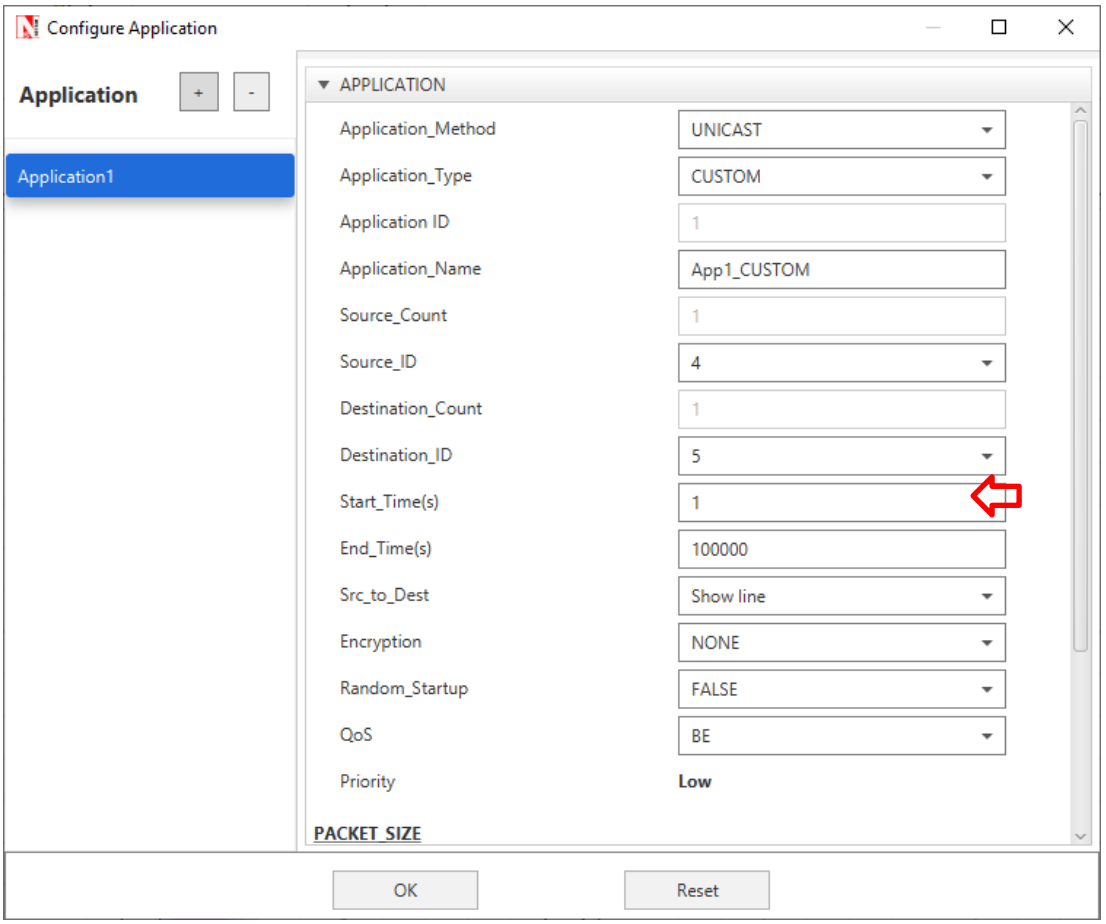
L2_Switch Properties	L2_Switch 1	L2_Switch 2	L2_Switch 3
Switch Priority	**2**	**1**	**3**

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.



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:

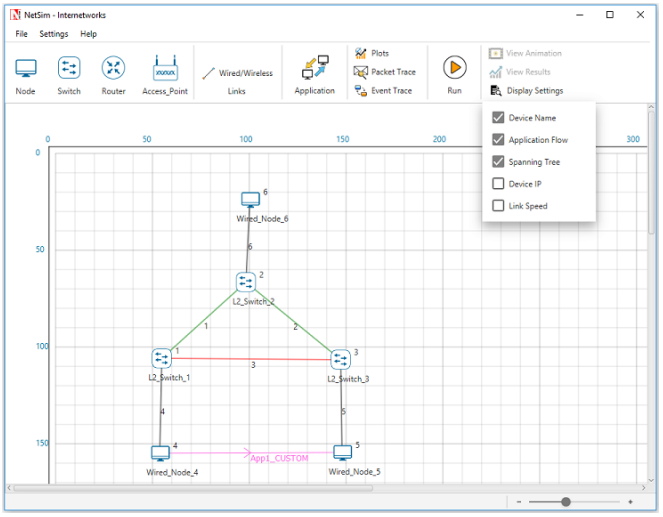
In Sample 2, the "Switch Priority" of all the 3 L2 Switches are changed as follows:

L2_Switch Properties	L2_Switch 1	L2_Switch 2	L2_Switch 3
Switch Priority	**1**	**2**	**3**

3.4 Output:

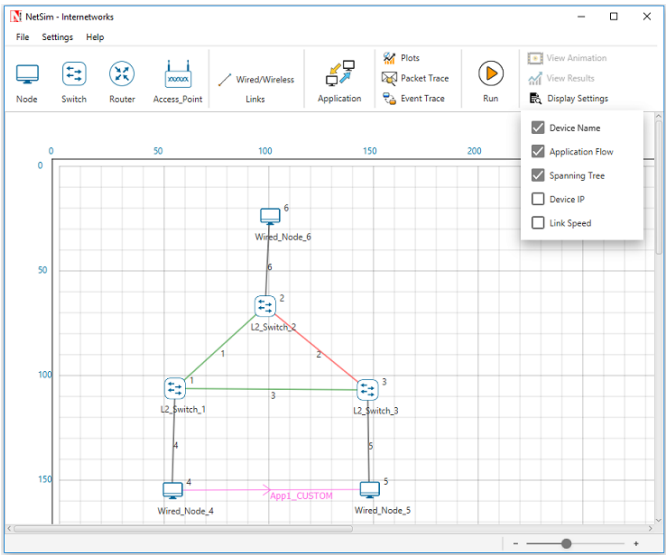
Click on **Display Settings > Spanning Tree** check box in the NetSim GUI.

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.**

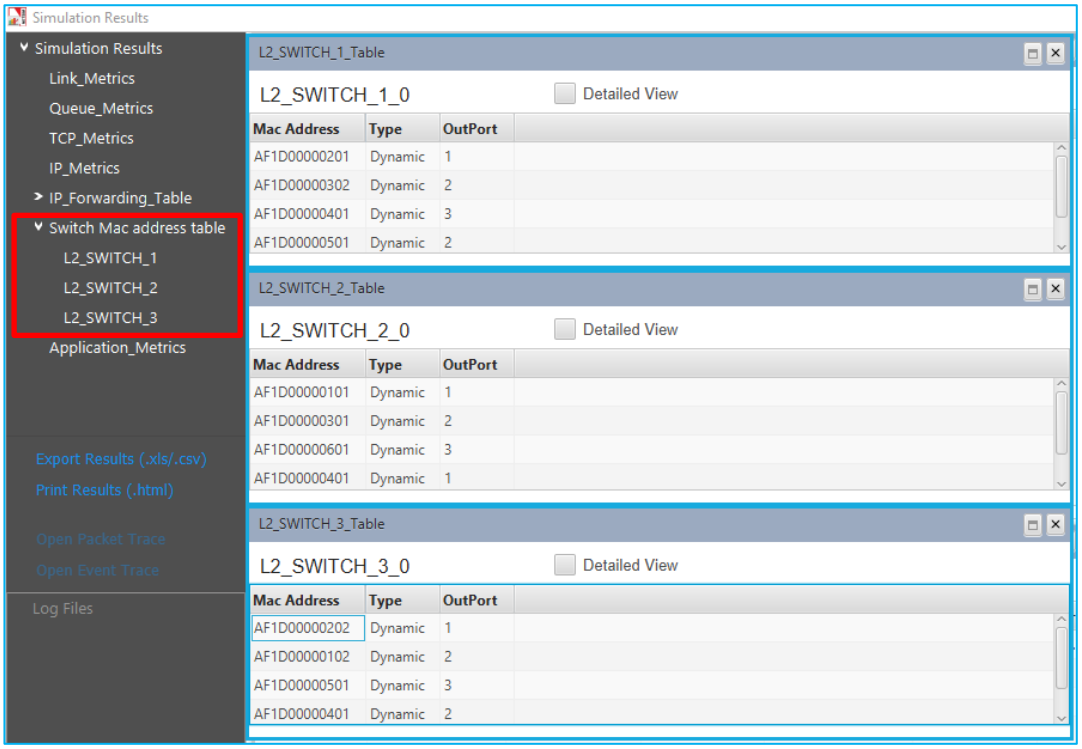
Sample 2:



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.



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. Understand the working of “Connection Establishment” in TCP

4.1 Introduction:

When two processes wish to communicate, their TCP’s must first establish a connection i.e. initialize the status information on each side. Since connections must be established between unreliable hosts and over the unreliable internet communication system, a “three-way handshake” with clock-based sequence numbers is the procedure used to establish a Connection. This procedure normally is initiated by one TCP and responded by another TCP. The procedure also works if two TCPs simultaneously initiate the procedure. When simultaneous attempt occurs, each TCP receives a “SYN” segment which carries no acknowledgement after it has sent a “SYN”.

The simplest three-way handshake is shown in the following figure:

	TCP A			TCP B
1.	CLOSED			LISTEN
2.	SYN-SENT	→	<A: SEQ=100><CTL=SYN>	→SYN-RECEIVED
3.	ESTABLISHED	←	<B: SEQ=300><ACK=101><CTL=SYN, ACK>	←SYN-RECEIVED
4.	ESTABLISHED	→	<A: SEQ=101><ACK=301><CTL=ACK>	→ESTABLISHED
5.	ESTABLISHED	→	<A: SEQ=101><ACK=301><CTL=ACK><DATA>	→ESTABLISHED

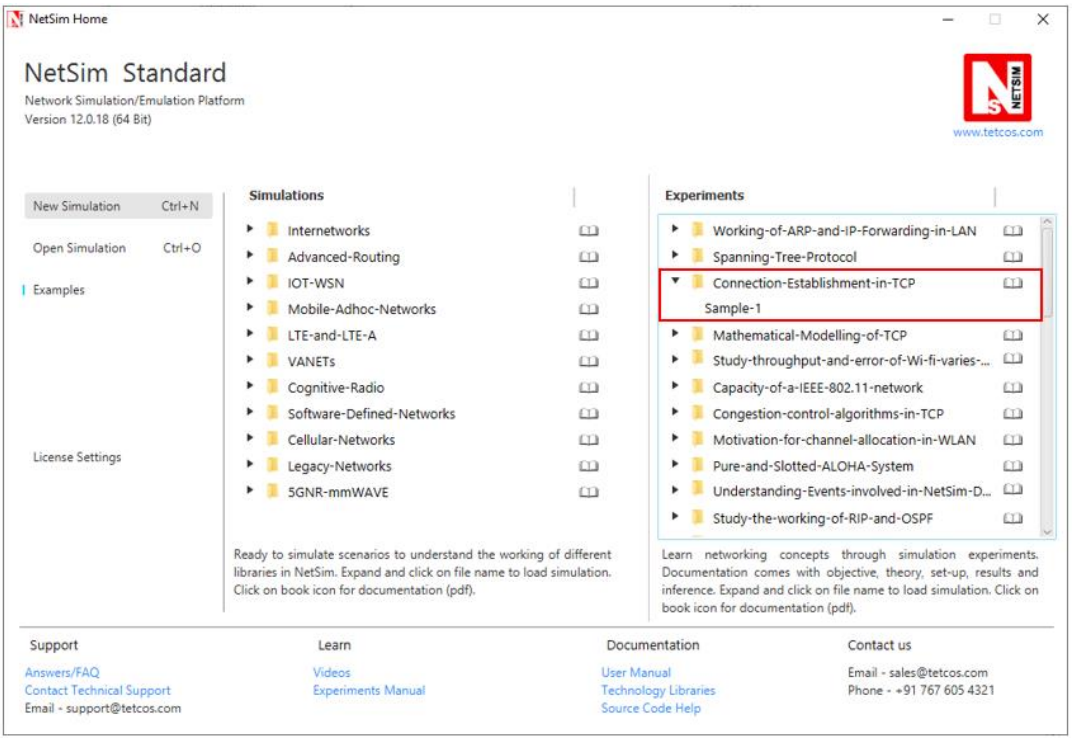
Fig: Basic 3-Way Handshake for Connection Synchronization

Explanation:

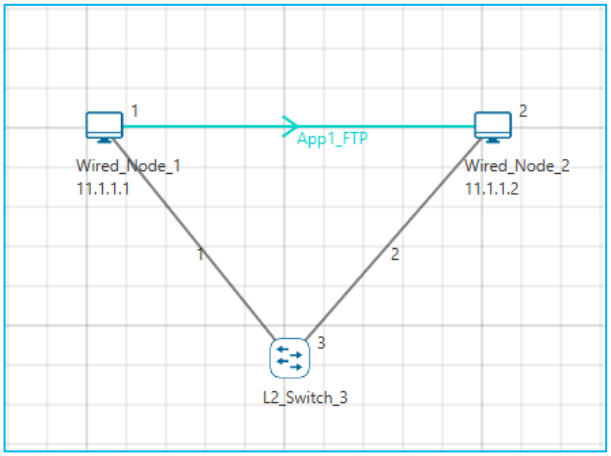
The above figure should be interpreted in the following way. Each line is numbered for reference purposes. Right arrows (→) indicates the departure of a TCP Segment from TCP A to TCP B, or arrival of a segment at B from A. Left arrows (←) indicates the reverse. TCP states represent the state after the departure or arrival of the segment (whose contents are shown in the center of each line). Segment contents are shown in abbreviated form, with sequence number, control flags, and ACK field. In line 2 of the above figure, TCP A begins by sending a SYN segment indicating that it will use sequence numbers starting with sequence number 100. In line 3, TCP B sends a SYN and acknowledges the SYN it received from TCP A. Note that the acknowledgment field indicates, TCP B is now expecting to hear sequence 101, acknowledging the SYN which occupied sequence 100. At line 4, TCP A responds with an empty segment containing an ACK for TCP B's SYN and in line 5, TCP A sends some data.

4.2 Network Setup:

Open NetSim and click **Examples > Experiments > Connection-Establishment-in-TCP > Sample-1** as shown below:



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



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 1 L2 Switch 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 L2 Switch as well as the Links.*

Step 3: 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 remaining 100000 Bytes and File Inter Arrival Time remaining 5 Seconds.

Step 4: Packet Trace is enabled in the NetSim GUI and hence we can observe the TCP Three-Way Handshake that occurs before the data packet transmissions.

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. In the **Static ARP Configuration** tab, Static ARP is set to Disable.

4.4 Output:

Once the simulation begins, Wireshark starts to capture the packets.

1. NODE-1 (11.1.1.1 as per this scenario) sends a control packet of type TCP_SYN requesting a connection with NODE-2 (11.1.1.2 as per this scenario).
2. NODE-2 responds with the control packet of type TCP_SYN_ACK to NODE-1. This TCP_SYN_ACK is the ACK packet sent for the TCP_SYN packet.
3. NODE-1 then sends the TCP_ACK to NODE-2 and internally sets its CONNECTION_STATE as TCP_ESTABLISHED.

Once the connection is established, data transmission starts and we can see the data packets (of size 1500 bytes) sent from NODE-1 to NODE-2.

TCP Three-way Handshake can be observed in Wireshark as shown below:

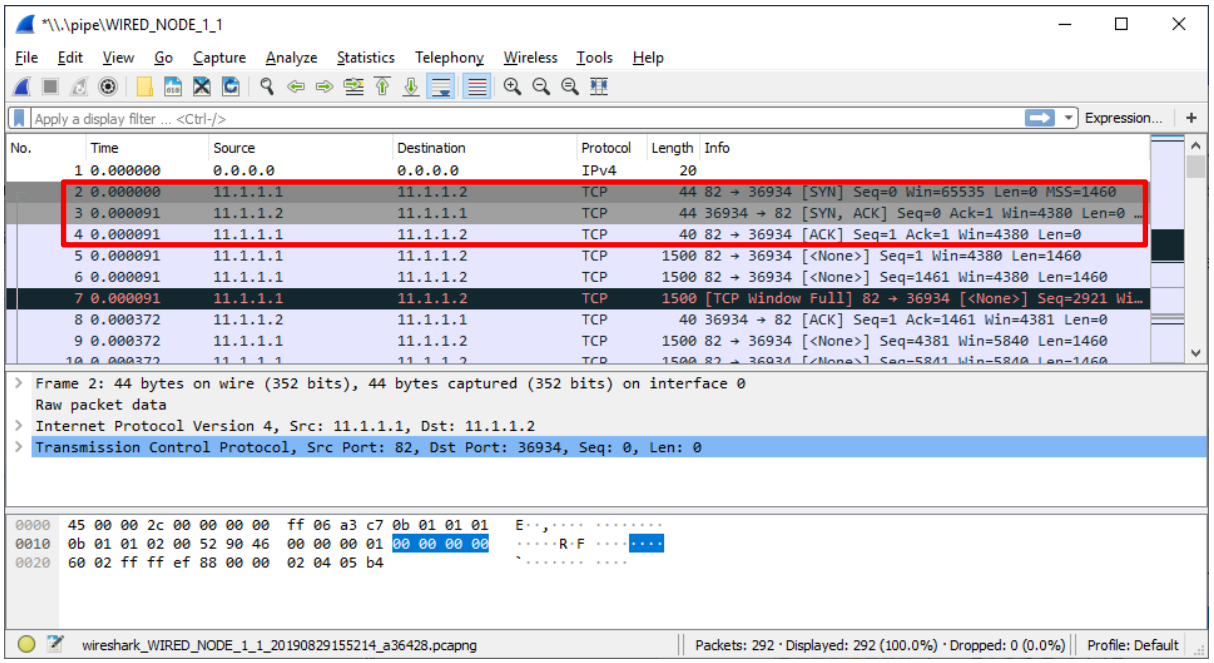


Fig**:** 3-way handshake captured in Wireshark

Similarly, users can also see the TCP 3-Way-Handshake using the Packet Trace. To view the packet trace file, click on **"Open Packet Trace"** option present in the left-hand-side of the Results Dashboard.

PACKET_ID	SEGMENT_ID	PACKET_TYPE	CONTROL_PACKET_TYPE/APP_NAME	SOURCE_ID	DESTINATION_ID	TRANSMITTER_ID	RECEIVER_ID
0	N/A	Control_Packet	TCP_SYN	NODE-1	NODE-2	NODE-1	SWITCH-3
0	N/A	Control_Packet	TCP_SYN	NODE-1	NODE-2	SWITCH-3	NODE-2
0	N/A	Control_Packet	TCP_SYNACK	NODE-2	NODE-1	NODE-2	SWITCH-3
0	N/A	Control_Packet	TCP_SYNACK	NODE-2	NODE-1	SWITCH-3	NODE-1
0	N/A	Control_Packet	TCP_ACK	NODE-1	NODE-2	NODE-1	SWITCH-3
0	N/A	Control_Packet	TCP_ACK	NODE-1	NODE-2	SWITCH-3	NODE-2
1	1	FTP	App1_FTP	NODE-1	NODE-2	NODE-1	SWITCH-3
1	2	FTP	App1_FTP	NODE-1	NODE-2	NODE-1	SWITCH-3
1	1	FTP	App1_FTP	NODE-1	NODE-2	SWITCH-3	NODE-2

Fig**:** 3-way handshake captured in NetSim Packet Trace

From the Packet Trace file, we can observe the following:

1. NODE-1 sends TCP_SYN to NODE-2 via SWITCH-3.
2. NODE-2 responds by sending back TCP_SYNACK to NODE-2 via SWITCH-3.
3. Then NODE-1 sends TCP_ACK to NODE-2 via SWITCH-3 and then starts sending the data packets to NODE-2.

5. Appreciate the mathematical modelling of TCP and understand the fundamental relationship between packet loss probability and TCP performance.

Part 1: Estimate transmission rate in a simple error-less network:

(***Reference:*** *Chapter 5 of the book, High Performance TCP/IP Networking. Concepts, Issues and Solutions by Mahbub Hassan, Raj Jain*)

5.1 Theory:

The two key processes that a model of TCP needs to include are:

1. The dynamics of the window that define the number of packets that a TCP source can transmit into the network
2. The packet loss process that indicates current traffic loads or congestion within the network

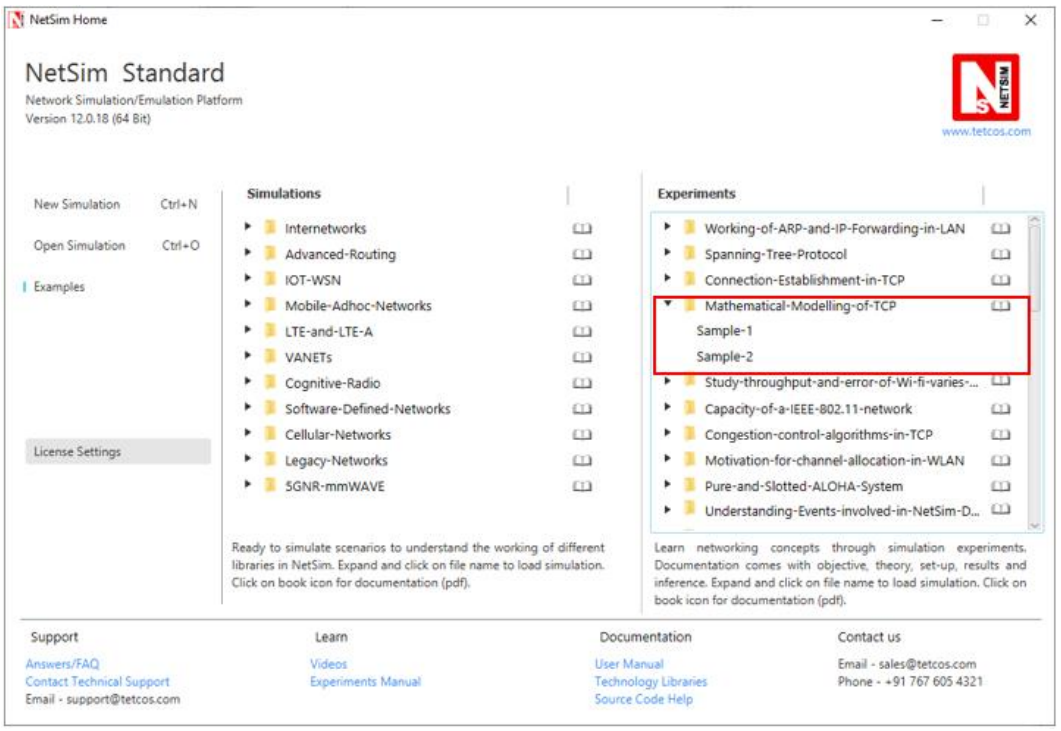
Now, during the interval in which TCP receives information that packets are not being lost in the network, TCP increases its window linearly. When the source deduce that a packet has been lost it reduces its window by a factor of the current window size.

The standard assumption in this case is that the window size is related to the transmission rate by the roundtrip time (RTT)

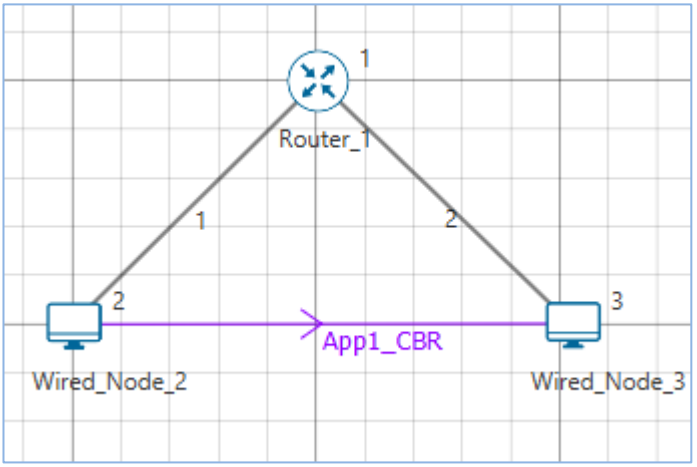
$$TranmissionRate = Window\ Size/RTT$$

5.2 Network setup:

Open NetSim and click **Examples > Experiments > Mathematical-Modeling-of-TCP > Sample-1** as shown below:



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



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 1 Router in the **"Internetworks"** Network Library.

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

Note: *Accept default properties for the Router.*

Step 3: The Link Properties are set according to the table given below:

Link Properties	Wired Link 1	Wired Link 2
Max Uplink Speed (Mbps)	10	10
Max Downlink Speed (Mbps)	10	10
Uplink propagation delay (µs)	100000	100000
Downlink propagation delay (µs)	100000	100000
Uplink BER	0	0
Downlink BER	0	0

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 2 i.e. Source to Wired Node 3 i.e. Destination with Packet Size remaining 1460 Bytes and Inter Arrival Time set to 1168 µs.

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:

$$Generation\ Rate\ (Mbps)\ =\ Packet\ Size\ (Bytes)\ *\ 8/Interarrival\ time\ (\mu s)$$

Step 5: Click on Run simulation. The simulation time is set to 100 seconds.

5.4 Output I:

Theoretical Calculation:

In the output of simulation, the throughput measured equals the transmissions rates since there are packet errors or losses in the network. This implies that the expected theoretical value of throughput should be

$$Throughput(Mbps) = Window\ Size/RTT$$

Window size = 65535Bytes = 65535*8 = 524280bits

RTT = 100ms*4 = 400ms = 0.4s

Expected Throughput (Mbps) = 524280/0.4 = 1.3Mbps

Note that even though we have a 10 Mbps links the Throughput is limited to 1.3 Mbps, because TCP window size is limited to 65KB.

Simulation Output:

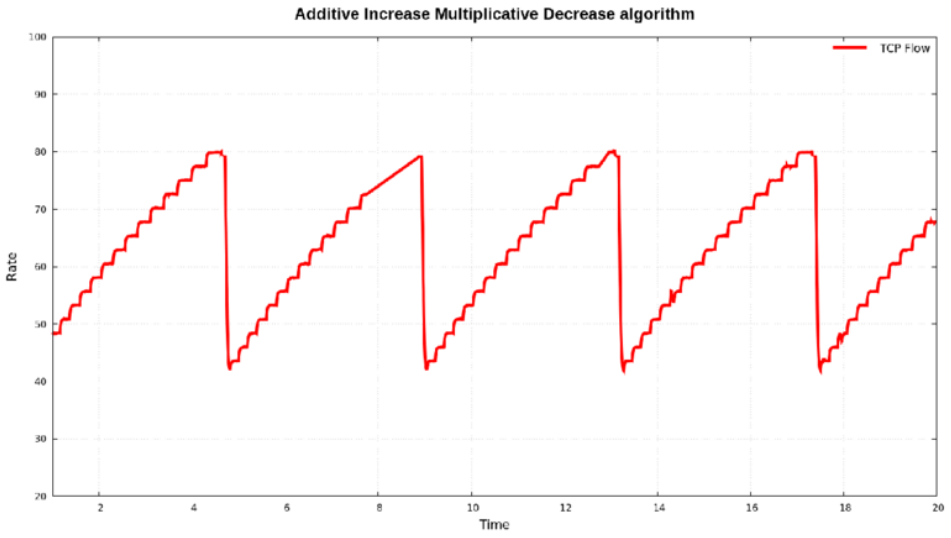
After running simulation, check throughput in Application metrics. The throughput obtained in NetSim is 1.24Mbps. The reason for the slightly lower throughput is due to the time taken initially for window to grow.

Part 2: TCP Model for a packet loss process:

5.5 Theory:

The simplest model in TCP that one can devise considers a periodic pattern in the dynamics of congestion window and the packet losses in steady state. TCP window evolves according to generic dynamics common to all TCP versions with each periodic loss event triggering a single, multiplicative window reduction. Because the steady state situation is being considered, a maximum window size W is always achieved by the time a packet is lost, which results in the window being reduced to W/2. The operation of the system in steady state means that packet losses occur with constant probability p, so that, on average, 1/p packets are transmitted into the network between each packet loss.

A trace of the window size results in a period saw tooth plot



The total number of packets transferred in a period would be the area underneath the trapezoidal shape of the period

$$No\ of\ Packets = \frac{1}{2}\frac{T}{RTT}\left(\frac{W}{2} + W\right)$$

Because of constant loss probability, p, the number of packets is given by 1/p. And the time taken to increase the window from W/2 to W is T = RTT. This gives us

$$\frac{1}{p} = \frac{W}{4}\left(\frac{W}{2} + W\right)$$

Solving this for W and applying that value of W gives us the average sending rate of TCP source, which is the number of packets transmitted during each period is

$$Sending\ Rate = \frac{1}{RTT}\left(\sqrt{\frac{3}{2p}}\right)$$

Where

RTT – Round Trip Time

p – Packet Error Rate = $1 - (1 - BER)^{Packet\ length}$

5.6 Procedure:

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

Step 1: In the Link Properties for both the links, Uplink and Downlink Bit Error Rate is set to 0.000001.

Step 2: Packet Trace is enabled in the NetSim GUI and hence we are able to calculate the Layer-wise Overheads which gets added in different layers of the TCP/IP Stack.

Step 3: Click on Run simulation. The simulation time is set to 100 seconds.

5.7 Observation:

Even though the packet size at the application layer is 1460 bytes, as the packet moves down the layers, some overhead is added which results in a greater packet size. This is the actual payload that is transmitted by the physical layer (1526 Bytes). The overheads added in different layers are shown in the table below and can be obtained from packet trace:

Layer-wise	Overhead (Bytes)
TRANSPORT LAYER	20
NETWORK LAYER	20
DATALINK LAYER	26
PHYSICAL LAYER	0
TOTAL	66

Therefore, the payload size = Packet Size + Overhead

= 1460 Bytes + 66 Bytes

= 1526 Bytes

5.8 Output II:

Theoretical Calculation:

The number of packets transmitted per second is

$X = 1/RTT\sqrt{3/2p}$

RTT = 4*100ms = 0.4 s

Packet length = 1526*8 = 12208 bits

Packet Error Rate = $1 - (1 - 0.000001)^{12208}$ = 0.012

= 1/0.4$\sqrt{\frac{3}{2*0.012}}$ = 27.95 packets per second

Throughput (Mbps) = 27.95*1526*8 = 0.32Mbps

Simulation Output:

After running simulation, check throughput in Application metrics. The throughput obtained in NetSim is 0.26 Mbps.

The reason for the slight difference in the results in that the theoretical model is a simplistic model that does not factor in additional TCP features such as a) limits on the window size enforced by the receiver b) Timeouts c) Duplicate ack's etc.

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

6.1 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.

6.2 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 (11Mcps). An 11 chip Barker code yields 1 million symbols per second (1Msps). These symbols are Differential Phase Shift Keying modulated to get 1 bit per symbol, thereby yielding 1Mbps, and Quarternary Differential Phased Keying modulated to get 2 bits per symbol, thereby yielding 2Mbps. In order to get 5.5Mbps and 11Mbps, each symbol is made from 8 chips, so that the symbol rate is 1.375Msps. A technique called Complementary Code Keying (CCK) then provides 4 bits per symbol, yielding 5.5Mbps, and 8 bits per symbol, which yields 11Mbps.

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 closer 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.

$$SNR = \frac{P_{received}}{N_0 W}$$

Where N_0 is the noise power spectral density (W/Hz) and W is the system bandwidth (nominally 22Mhz). The noise power works out to approximately -100dBm. The received power $(P_{received})$ is obtained by subtracting the *path-loss,* between the transmitter and the receiver, from the transmitted

power (e.g., $P_{transmit} = 0$ dBm, would arise from a transmit power of 1mW). A simple expression for path-loss is given by

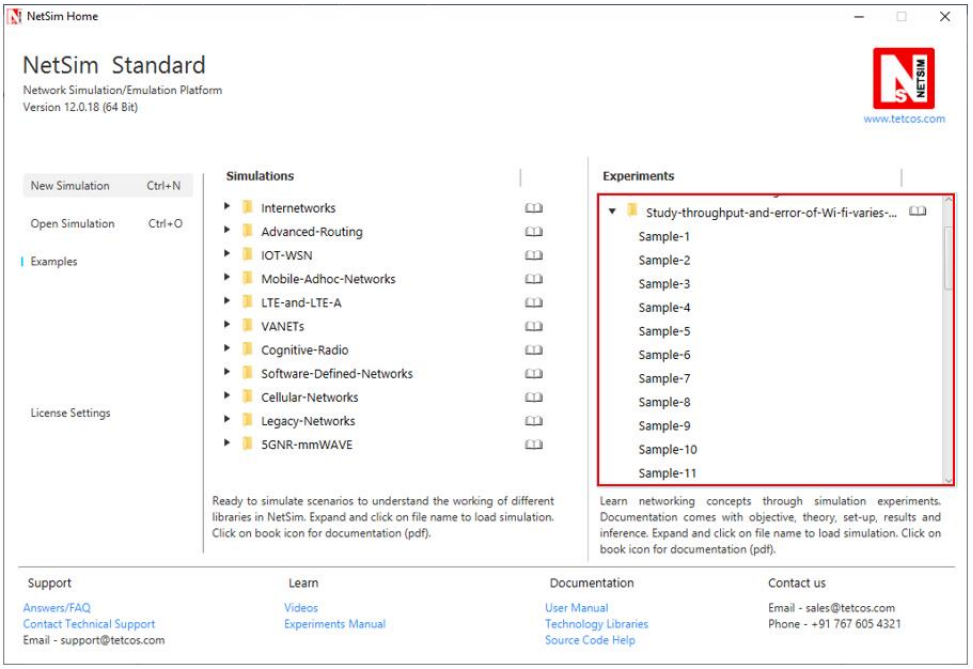
$$P_{received} = P_{transmit} - c_0 - 10\,\eta\,log_{10}\,d$$

where c_0 is the path loss at the "reference" distance of 1m, η 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."

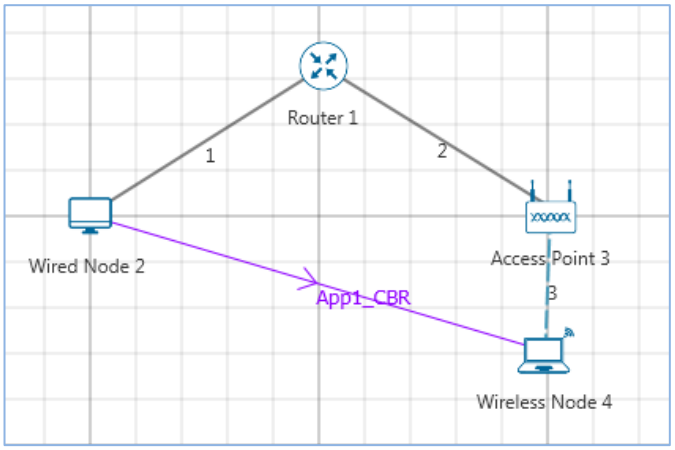
As d increases, the received power decreases; e.g., doubling the distance reduced the received power by approximately 3η, since $\log_{10} 2 \approx 0.3$. Typical values of η, indoors, could be 3 to 5, resulting in 9dB to 15dB additional path loss for doubling the value of d.

6.3 Network Setup

Open NetSim and click **Examples > Experiments > Study-throughput-and-error-of-a-WLAN-varies-with-distance > Sample-1** as shown below:



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



6.4 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.

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 Datalink Layer, Rate Adaptation is set to False.

	Wireless Node 4 Properties	Access Point
X/Lat	200	200
Y/Lon	30	0
Interface_Wireless properties		
IEEE Standard	802.11b	802.11b
Rate _Adaptation	False	False

Step 3: In the Source Node, i.e. Wired Node 2 and in the Destination Node, i.e. Wireless Node 4, TCP Protocol in the Transport Layer is disabled.

Step 4: Wired and Wireless links properties are set as follows:

Wireless Link Properties	
Channel Characteristics	Path loss only
Path Loss Model	Log_Distance
Path Loss Exponent	3

Wired Link Properties	
Uplink Speed (Mbps)	100
Downlink Speed (Mbps)	100
Uplink BER	0.0000001
Downlink BER	0.0000001

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.

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, and 500 m.

6.5 Output:

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

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

$$Phy\ rate\ (802.11b)\ =\ Phy\_layer\_payload * 8/(phy\ end\ time - phy\ arrival\ time - 192)$$

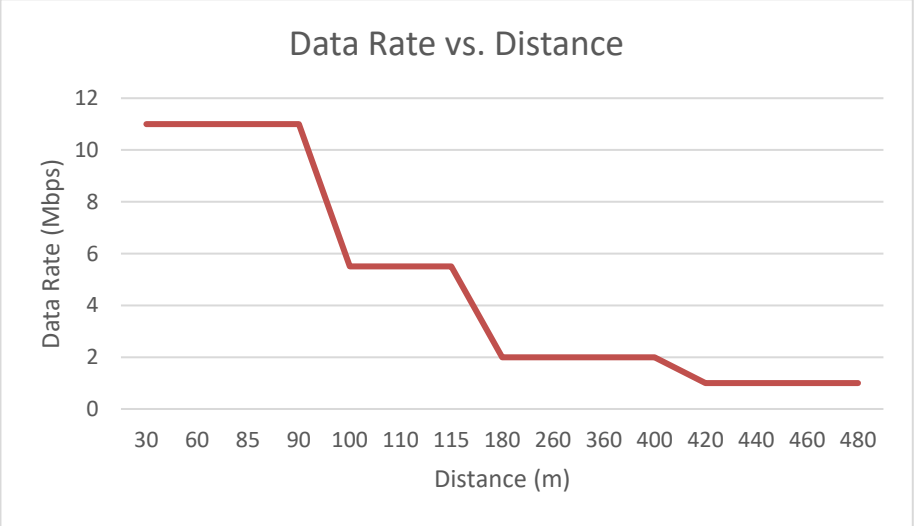
192 micro seconds is the approximate preamble time for 802.11b

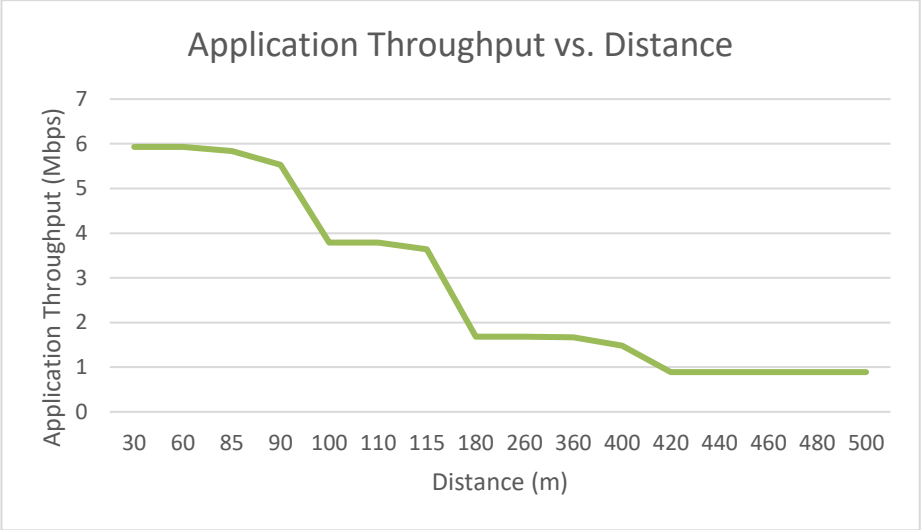
Calculate PHY rate for all the data packets coming from Access Point to Wireless node. For doing this please refer section 7.5.1 How to set filters to NetSim Packet Trace file from NetSim's User Manual. Filter Packet Type to CBR, Transmitter to Access Point and Receiver to Wireless node.

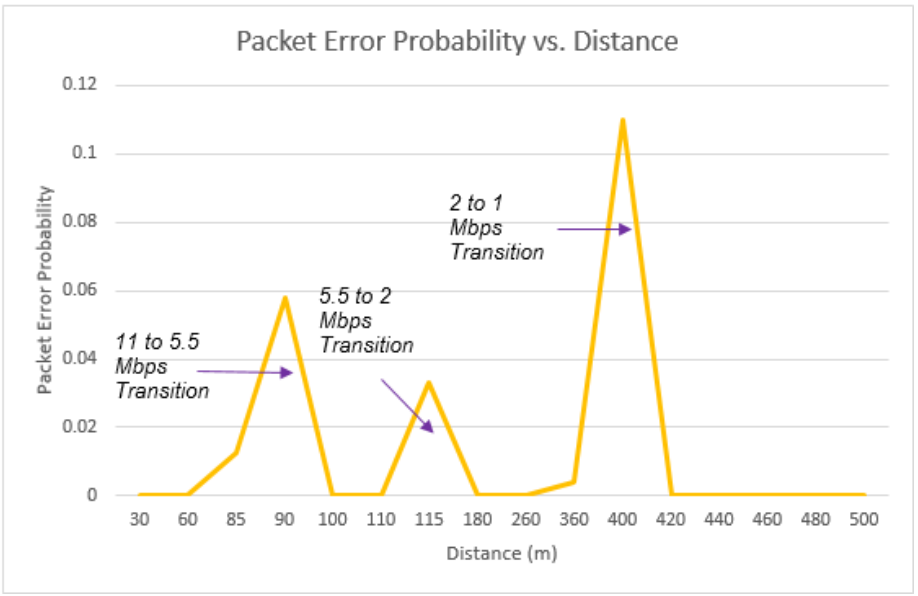
Since $\boldsymbol{PER = 1 - (1 - BER)^{PL}}$ where PER is packet error rate, PL is packet length in bits and BER is bit error rate, we get $\boldsymbol{BER = 1 - e^{\frac{log\,(1-PER)}{PL}}}$

On tabulating the results, you would see

802.11b					
Distance (m)	**PHY rate in Mbps (Channel capacity)**	**Application Throughput (Mbps)**	**Packets Transmitted**	**Packets Errored**	**Packet error probability**
30	11	5.9276	5110	0	0
60	11	5.9276	5110	0	0
85	11	5.842920	5101	64	0.0125
90	11	5.53204	5063	294	0.058
100	5.5	3.78856	3266	0	0
110	5.5	3.78856	3266	0	0
115	5.5	3.64588	3253	110	0.033
180	2	1.6762	1445	0	0
260	2	1.6762	1445	0	0
360	2	1.66808	1445	7	0.004
400	2	1.48016	1436	160	0.110
420	1	0.89204	769	0	0
440	1	0.89204	769	0	0
460	1	0.89204	769	0	0
480	1	0.89204	769	0	0
500	0	0	0	0	0







Note: All the above plots highly depend upon the placement of nodes 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.

6.6 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.

$$Received\ Power = Transmitted\ Power - RF\ losses$$

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 95m, and then to 2 Mbps at 175m and to 1 Mbps at 415m (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.

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

7.1 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 θ bytes per second, what is the maximum number of users (say, n_θ) up to which the throughput obtained by every user is at least θ. We can say that n_θ is the *capacity* of this simple Wi-Fi network for the *Quality of Service (QoS)* objective θ. [1]

7.2 Objective

In this experiment we will learn how to obtain n_θ 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.

7.3 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 (~ 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.

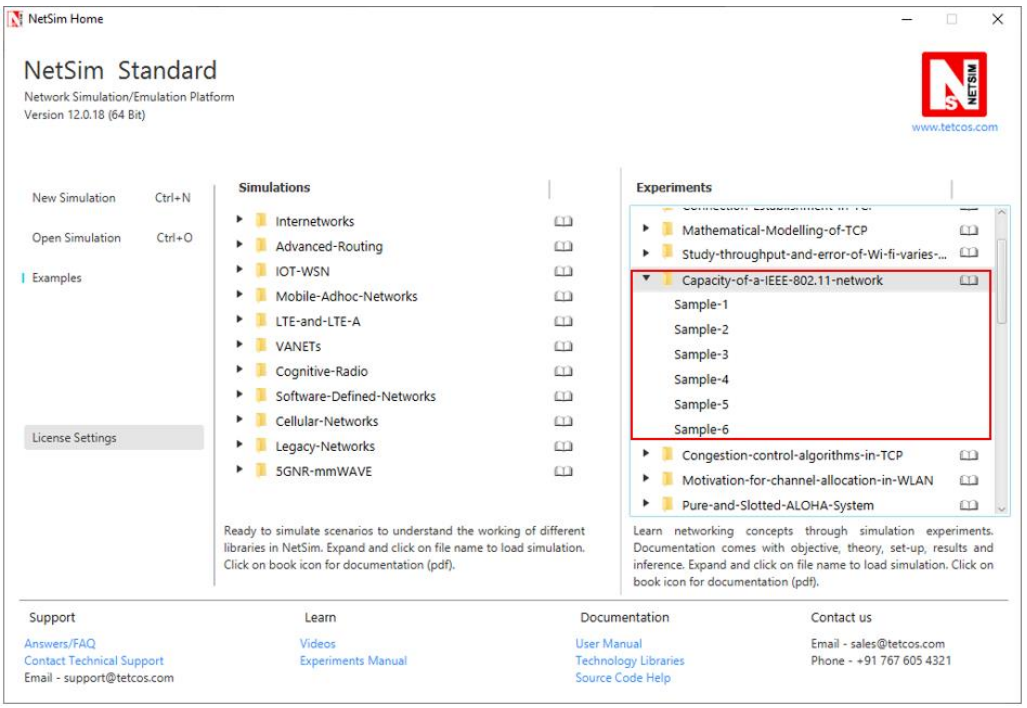
Now suppose that the TCP data packet throughput with the two-node model is Θ. Then

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

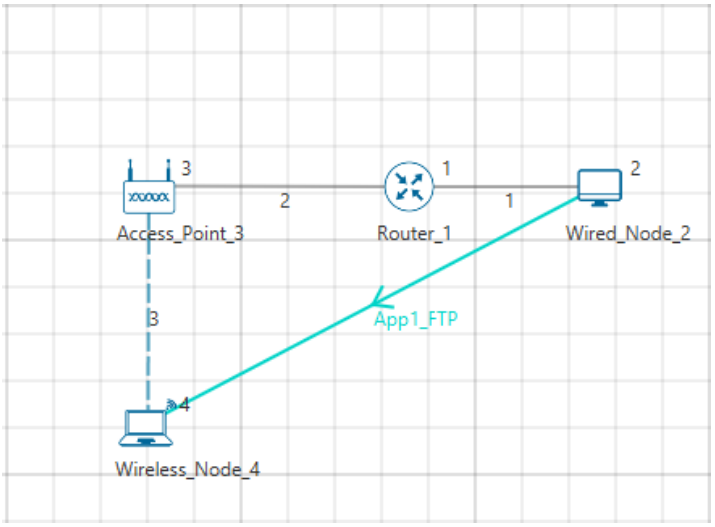
where the $\lfloor x \rfloor$ denotes the largest integer less than or equal to x. Use NetSim to verify that for an 11Mbps Wi-Fi speed, with RTS/CTS enabled the total TCP throughput is 3.4 Mbps. If $\theta = 0.65\ Mbps$, then $n_\theta = \left\lfloor \frac{3.8}{0.65} \right\rfloor = 5$. In this example,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

7.4 Procedure

Open NetSim and click **Examples > Experiments > Capacity-of-a-IEEE-802.11 > Sample-1** as shown below:



NetSim UI displays the configuration file corresponding to this experiment.

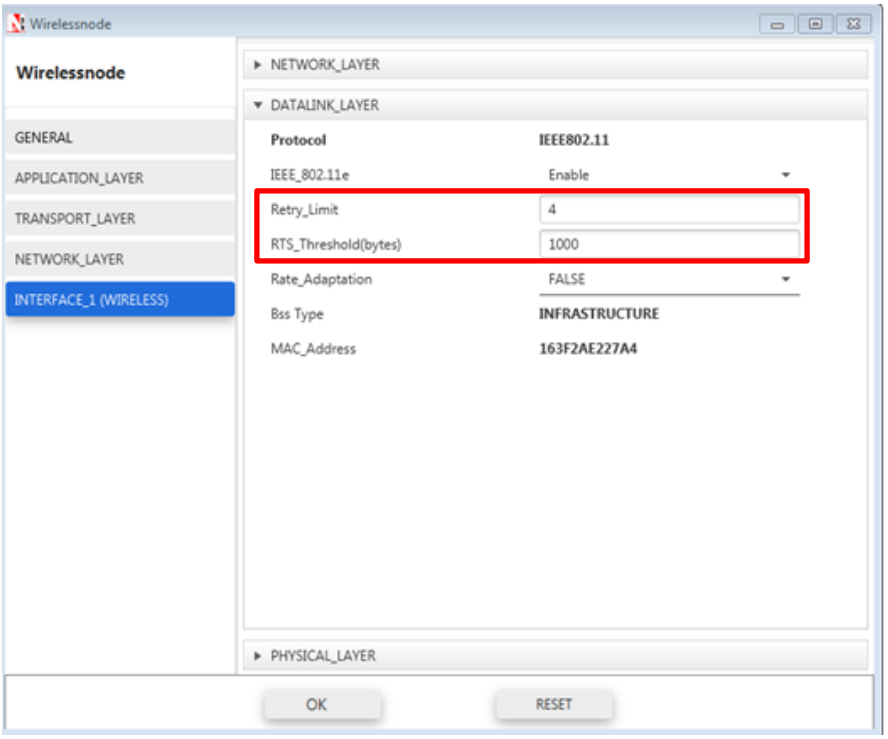


7.5 Procedure:

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.

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.



Step 3: The Link Properties is set as per the table given below:

Wired Link	
Uplink BER rate	0
Downlink BER rate	0
Uplink Delay	0µs
Downlink Delay	0µs

Wireless Link	
Channel Characteristics	No path loss

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 15µs.

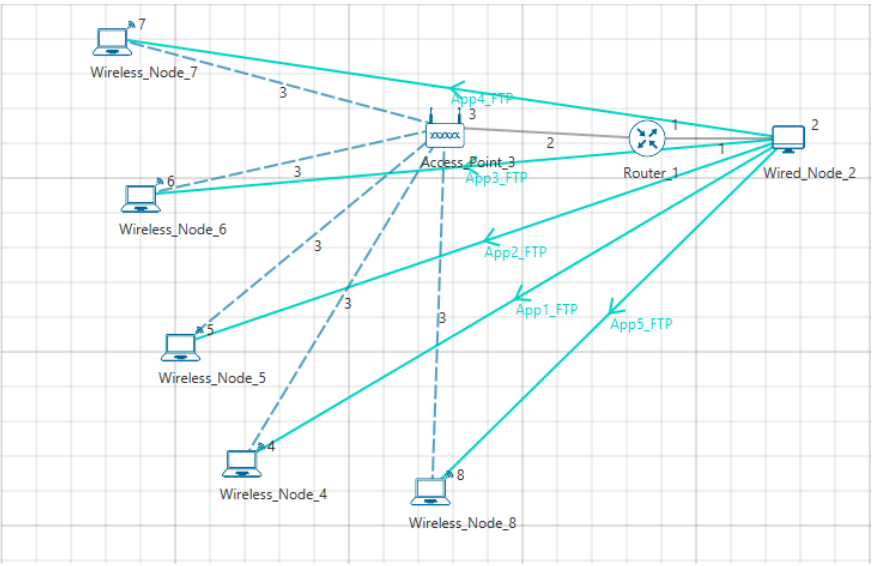
Step 4: Run the Simulation for 15 Seconds and note down the throughput.

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

Properties	App1	App2	App3	App4	App5
Application Type	FTP	FTP	FTP	FTP	FTP
Source Id	2	2	2	2	2
Destination Id	4	5	6	7	8
File size (Bytes)	100000000	100000000	100000000	100000000	100000000
File Inter arrival time	15s	15s	15s	15s	15s

Step 4: 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.**

7.6 Measurements and Output:

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

$$Throughput\ Per\ Device\ (Mbps) = \frac{Sum\ of\ throughpots\ (Mbps)}{Number\ of\ Devices}$$

Sample Number	Number of Devices	Sum of throughputs (Mbps)	Throughput Per Device (Mbps)
1	1	3.38	3.38
2	5	3.40	0.68
3	10	3.19	0.318
4	15	3.25	0.216
5	20	3.22	0.161
6	25	3.25	0.129

***NOTE:* In the referred paper we see that, the throughput value for 11 Mbps WLAN is 3.8 Mbps. Please note that this is the aggregate PHY throughput of the AP. However, in NetSim, we are calculating the total Application throughput.**

To derive the PHY layer throughput from the APP layer throughput, we need to add overheads of all layers

Layer	Overhead (Bytes)
Transport Layer	20
Network Layer	20

MAC Layer	40
PHY layer	48µs = (11*48)/8 = 66
Total Overhead	146

*PHY_Throughput = APP_Throughput * 1606/1460 = 3.41*1606/1460 = 3.79 Mbps)*

7.7 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 θ bytes per second, what is the maximum number of users (say, n_θ)?

If we set θ 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_θ)

7.8 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*

8. Understand the working of Slow start and Congestion Avoidance (Old Tahoe), Fast Retransmit (Tahoe) and Fast Recovery (Reno) Congestion Control Algorithms in TCP.

8.1 Theory:

One of the important functions of a TCP Protocol is congestion control in the network. Given below is a description of the working of Old Tahoe and Tahoe variants (of TCP) control congestion.

Old Tahoe: Old Tahoe is one of the earliest variants of TCP. It implements two algorithms called slow start and congestion avoidance to update the congestion window.

Slow Start: At the start of data transmission the size of congestion window is one. This means TCP can send only one packet until it receives an acknowledgement. When the ACK is received by the sender the congestion window increases to two. Now the sender can send two data packets. Upon the arrival of every new ACK the sender increases its congestion window by one. This phase is known as the slow start phase where the congestion window increases exponentially. So on the arrival of a new ACK, $cwnd$ + = MSS;

Congestion Avoidance: TCP will continue the slow start phase until it reaches a certain threshold, or if packet loss occurs. Now it enters in to a phase called congestion avoidance. Here the congestion window grows linearly. This means that the congestion window increases from ‘n’ to ‘n+1’ only when it has received ‘n’ new ACKs. The rate of growth of congestion window slows down because this is the stage where TCP is susceptible to packet loss. The formula used here is $cwnd$ += ($SMSS * SMSS$)/cwnd

Tahoe (Fast Retransmit): TCP Tahoe implements all the above mentioned algorithms used by Old Tahoe. The Fast Retransmit algorithm was included in Tahoe to improve the response time of TCP.

Fast Retransmit: One of the major drawbacks of Old Tahoe is that it depends on the timer to expire before it can retransmit a packet. TCP Tahoe tries to improve upon Old Tahoe by implementing the Fast Retransmit algorithm. Fast Retransmit takes advantage of the fact that duplicate ACKs can be an indication that a packet loss has occurred. So, whenever it receives 3 duplicate ACKs it assumes that a packet loss has occurred and retransmits the packet.

Reno (Fast Recovery): TCP Reno retains the basic principles of Tahoe such as Slow Start, Congestion Avoidance and Fast Retransmit. However it is not as aggressive as Tahoe in the

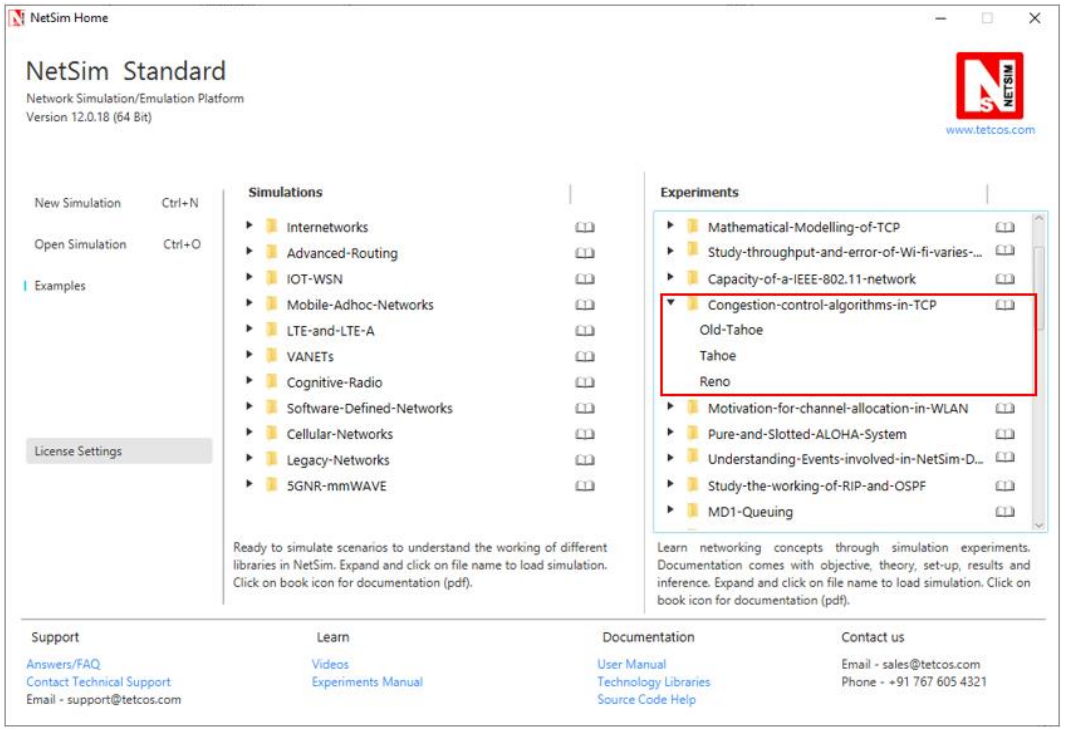
reduction of the congestion window. Reno implements the Fast Recovery algorithm which is described below.

Fast Recovery: The congestion window drop to one on the arrival of a 3 duplicate ACK can be considered as an extreme precaution. Arrival of 3 duplicate ACKs corresponds to light congestion in the network and there is no need for the congestion window to drop down drastically. The Fast Recovery algorithm does the following on the arrival of a third duplicate ACK:

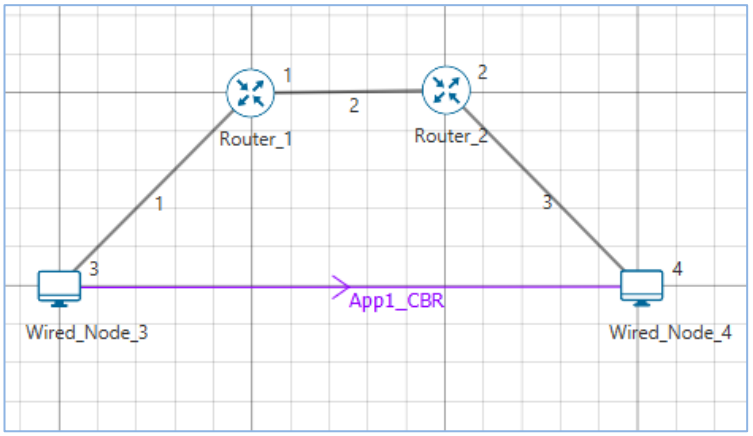
- The threshold value is set to half of the congestion window. $ssthresh = cwnd/2$.
- The congestion window is now set to be threshold plus three times the MSS. $cwnd = ssthresh + 3 * SMSS$.
- On the arrival of another duplicate ACK the congestion window increases by one MSS. This is done because an ACK signifies that a segment if out of the network and the sender can pump in another packet into the network. This is somewhat similar to slow start. $cwnd \mathrel{+}= SMSS$.
- TCP remains in fast recovery phase until it receives a higher ACK from the receiver.
- On receiving a higher ACK the congestion window is set to the threshold value. From now onwards congestion avoidance is followed. $cwnd = ssthresh$.

8.2 Network Set Up:

Open NetSim and click **Examples > Experiments > Congestion-control-algorithms-in- TCP > Old-Tahoe** as shown below:



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



8.3 Procedure:

Old Tahoe:

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 3, in the TRANSPORT LAYER Properties, Congestion Control Algorithm is set to OLD TAHOE.

Note: *Accept default properties for the Router.*

Step 3: The Link Properties are set according to the table given below:

Link Properties	Wired Link 1	Wired Link 2	Wired Link 3
Uplink Speed (Mbps)	20	100	20
Downlink Speed (Mbps)	20	100	20
Uplink propagation delay (µs)	5	100	5
Downlink propagation delay (µs)	5	100	5
Uplink BER	0.0000001	0.0000001	0.0000001
Downlink BER	0.0000001	0.0000001	0.0000001

Wired links between the Node and a Router are configured with a rate lower than that of the packet generation rate to create bottle necks.

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 3 i.e. Source to Wired Node 4 i.e. Destination with Packet Size remaining 1460Bytes and Inter Arrival Time set to 400 µs.

The Packet Size and Inter Arrival Time parameters are set such that the Generation Rate equals 30 Mbps (Approx.). Generation Rate can be calculated using the formula:

$$Generation\ Rate\ (Mbps)\ =\ Packet\ Size\ (Bytes)\ *\ 8/Interarrival\ time\ (\mu s)$$

Step 5: Click on Run simulation. The simulation time is set to 30 seconds.

Tahoe:

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

Step 1: In the Source Node, i.e. Wired Node 3, in the TRANSPORT LAYER Properties, Congestion Control Algorithm is set to TAHOE.

Step 2: Click on Run simulation. The simulation time is set to 30 seconds.

Reno:

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

Step 1: In the Source Node, i.e. Wired Node 3, in the TRANSPORT LAYER Properties, Congestion Control Algorithm is set to RENO.

Step 2: Click on Run simulation. The simulation time is set to 30 seconds.

8.4 Output

Comparison Table:

Congestion Control Algorithm	Throughput (Mbps)
Old Tahoe	7.26
Tahoe	16.10
Reno	17.76

From the above table, throughput for Tahoe is high since Tahoe retransmits the packets faster than Old Tahoe. Throughput for Reno is higher compared to Tahoe since *cwnd* is set to *ssthresh* instead of setting to 1 SMSS.

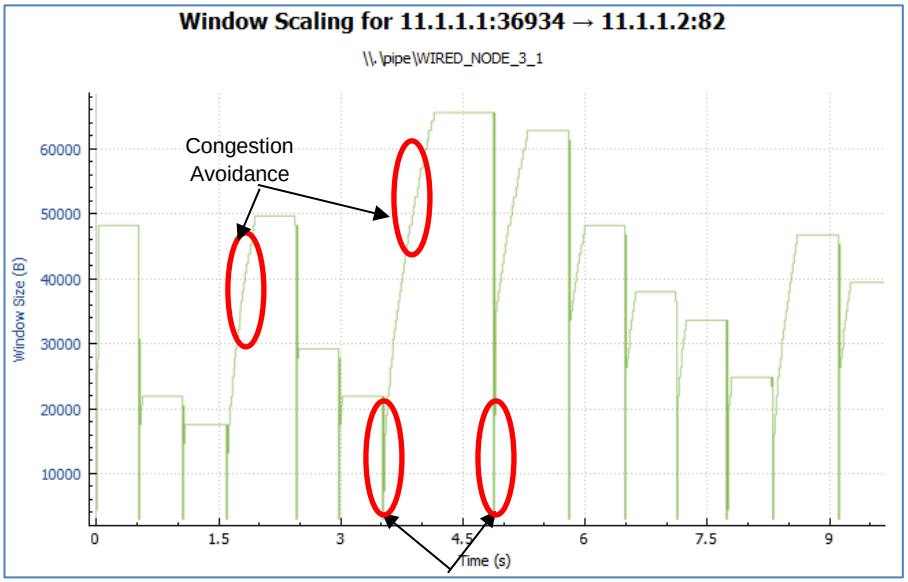
Go to the Wireshark Capture window.

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 - 7.7.5 Window Scaling in user manual)

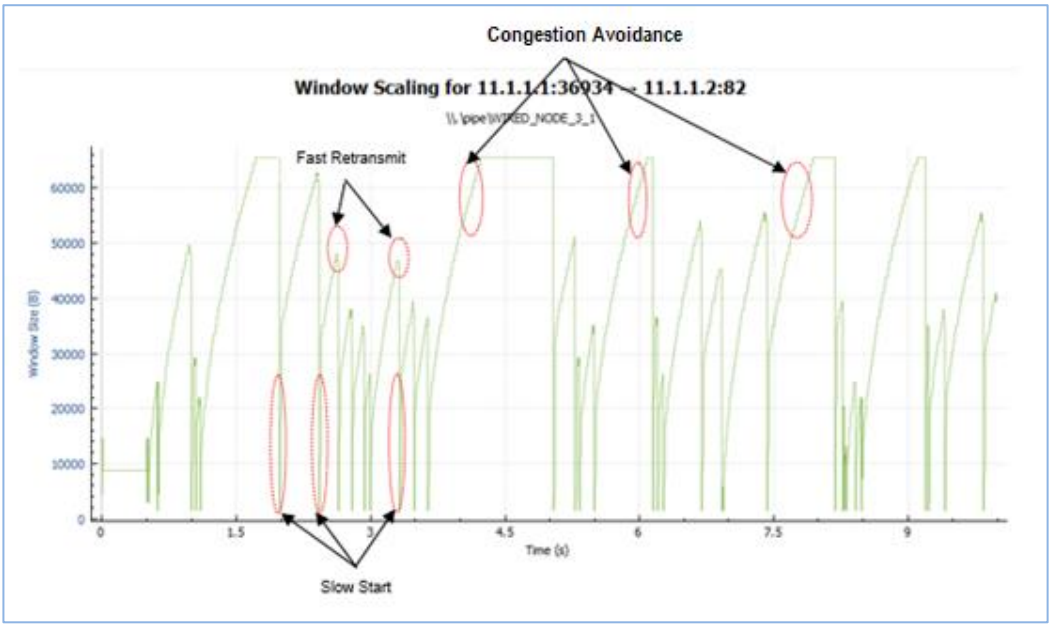
Old Tahoe:



Slow Start

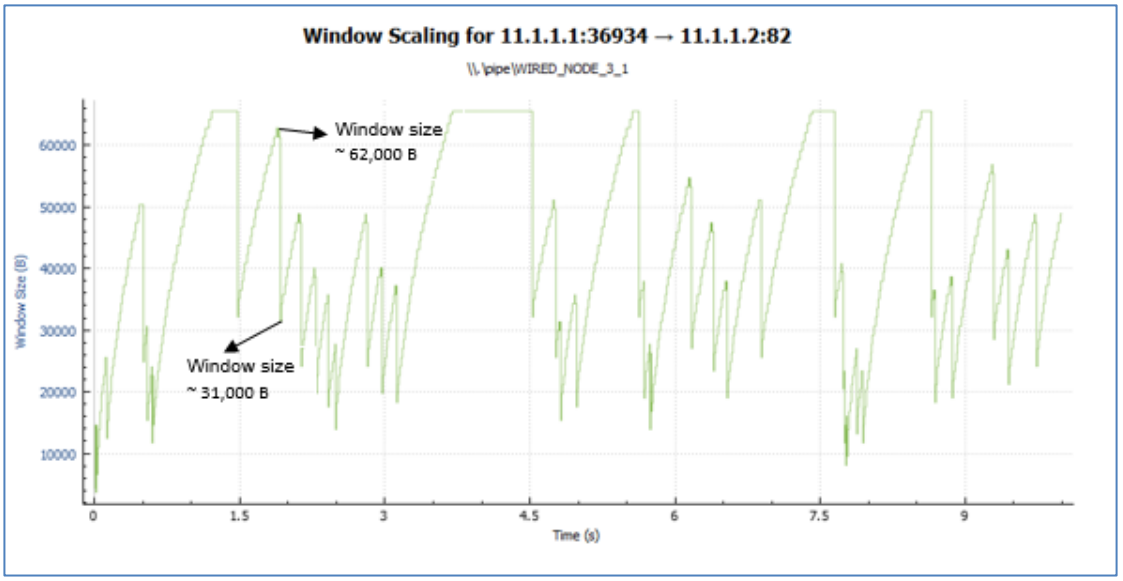
The graph shown above is a plot of Congestion window vs. Time. Each point on the graph represents the congestion window at the time when the packet is sent. You can observe that after every congestion window drop (caused by packet loss or timer expiry) Old Tahoe enters slow start, thereby increasing its window exponentially and then enters congestion avoidance where it increases the congestion window linearly.

Tahoe:



Tahoe uses the fast retransmit algorithm with which it responds to packet errors faster than Old Tahoe. Comparing the graphs of Old Tahoe and Tahoe one can observe that the latter drops the congestion window and retransmits faster than Old Tahoe (when three duplicate ack's are received).

Reno:



TCP Reno upon receiving the third duplicate ACK sets its congestion window according to the formula cwnd = cwnd/2 + 3*MSS. This can be observed in the above graph. On further arrival of duplicate ACKs it increases its congestion window by one MSS. If it receives a higher ACK then it drops the congestion window to the new threshold value. This mechanism is known as fast recovery.

8.5 Inference:

From this experiment we were able to understand how selection of TCP congestion control algorithm can have an impact on the throughput experienced by the applications. The major difference between Old Tahoe, Tahoe and Reno algorithms is the time taken to retransmit packets and the way congestion window size is reduced. There is a considerable improvement in the performance as we go from Old Tahoe to Tahoe and then to Reno. This is evident from the throughput readings obtained from the simulations performed.

9.Multi-AP Wi-Fi Networks: Channel Allocation

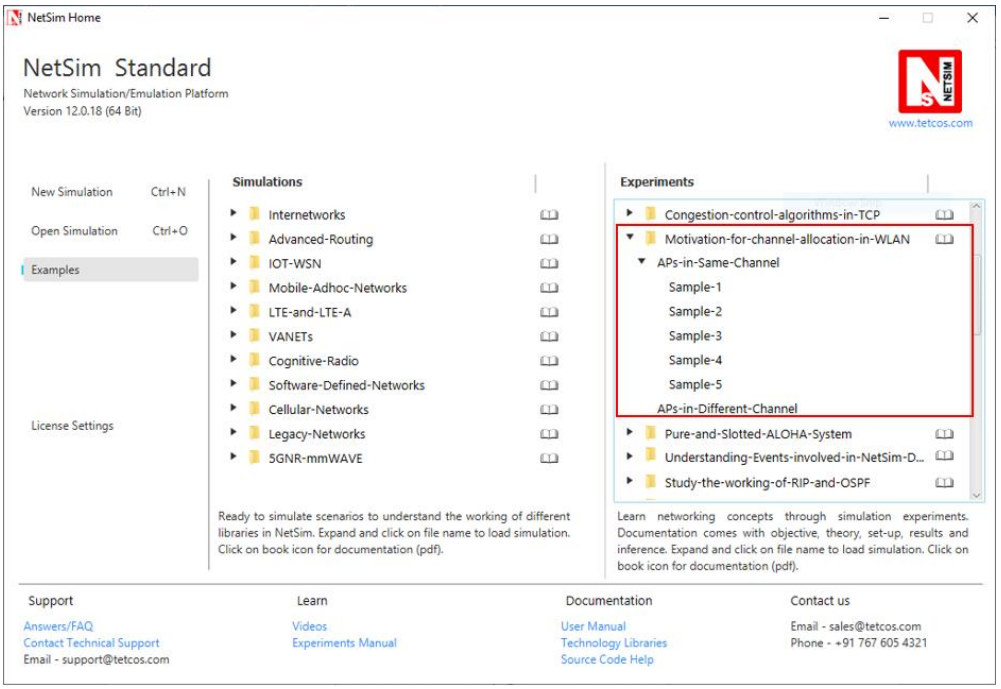
9.1 Introduction

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.

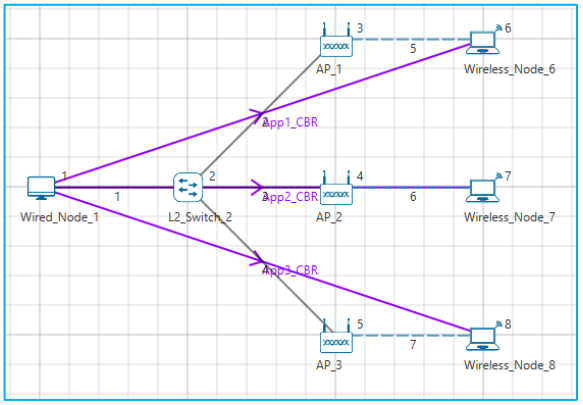
9.2 Network Setup:

Open NetSim and click **Examples > Experiments > Motivation-for-channel-allocation-in-WLAN** as shown below:



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

APs on the same channel:



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:

General Properties		
Device Name	**X**	**Y**
AP_1	15	5
AP_2	15	10
AP_3	15	15
Wireless_Node_6	20	5
Wireless_Node_7	20	10
Wireless_Node_8	20	20

Step 3: TCP is disabled in Wired Node 1.

Step 4: 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 5: In all the Wired Link Properties, Bit Error Rate and Propagation Delay is set to 0.

Step 6: The Wireless Link Properties are set as follows:

Channel Characteristics	PATH LOSS ONLY
Path Loss Model	LOG DISTANCE

Path Loss Exponent	3.5

Step 7: 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.

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:

$$Generation\ Rate\ (Mbps)\ =\ Packet\ Size\ (Bytes)\ *\ 8/Interarrival\ time\ (\mu s)$$

Similarly, two more CBR applications are generated.

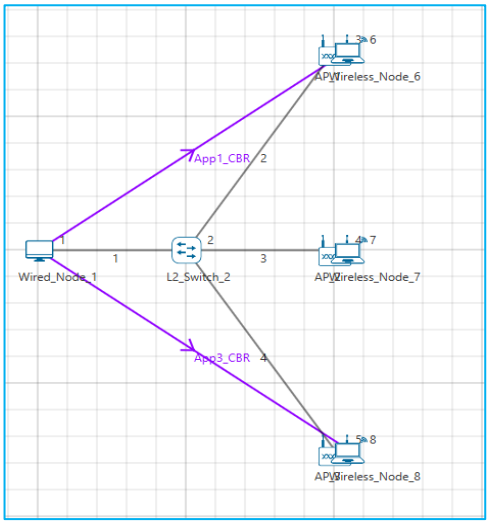
Step 8: Run the Simulation for 10 Seconds and note down the throughput.

Sample2:

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 > Environment Settings > 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 300m 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:

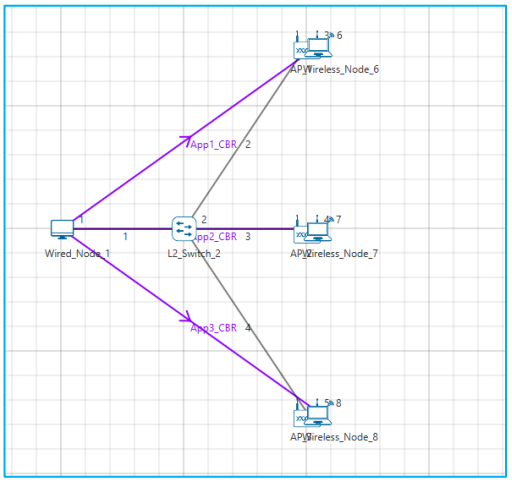
General Properties		
Device Name	**X**	**Y**
AP_1	400	0
AP_2	400	200
AP_3	400	400
Wireless_Node_6	410	0
Wireless_Node_7	410	200
Wireless_Node_8	410	400

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 150m 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:

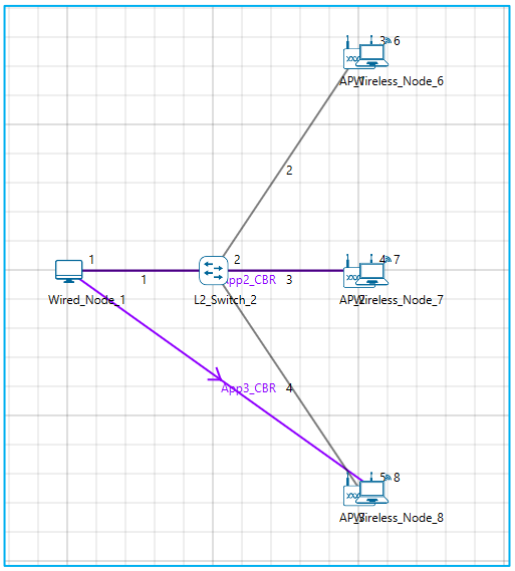
General Properties		
Device Name	**X**	**Y**
AP_1	400	0
AP_2	400	200
AP_3	400	400
Wireless_Node_6	410	0
Wireless_Node_7	410	200
Wireless_Node_8	410	400

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 1500m 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:

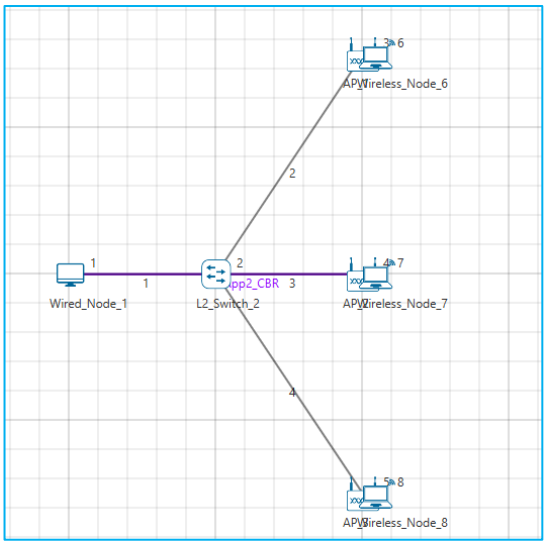
General Properties		
Device Name	**X**	**Y**
AP_1	400	0
AP_2	400	200
AP_3	400	400
Wireless_Node_6	410	0
Wireless_Node_7	410	200
Wireless_Node_8	410	400

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:

General Properties		
Device Name	**X**	**Y**
AP_1	400	0
AP_2	400	200
AP_3	400	400
Wireless_Node_6	410	0
Wireless_Node_7	410	200
Wireless_Node_8	410	400

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

APs in different channel:

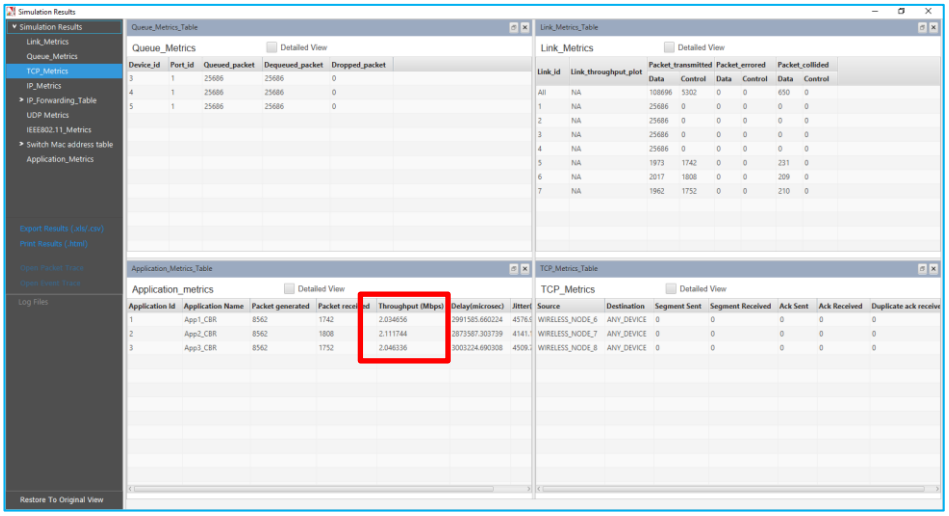
The following changes in settings are done from the previous sample:

Step 1: In 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.

9.3 Output:

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



Sample	Throughput (Mbps)		
	AP_1	AP_2	AP_3
All APs on the same channel			
1	2.03	2.11	2.04
2	5.94	N/A	5.92
3	5.42	0.63	5.41
4	N/A	3.29	3.26
5	N/A	5.92	N/A
Each AP on a different nonoverlapping channel			
1	5.94	5.92	5.92

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

9.4 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 in 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.

9.4.1 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 notcarrying 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.

9.4.2 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.

10. 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.***

10.1 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 of18%. 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 $S = {}^{1}/_{e}$ or about 0.368. It means that if the system is operating at G=1, the probability of an empty slot is 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;

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

Where, G = Attempts per packet time

ST = Simulation time (in second)

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

$$S = \frac{Number\ of\ packet\ sucess * Slot\ length(s)}{ST}$$

Where, S = Throughput per packet time

ST = Simulation time (in second)

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

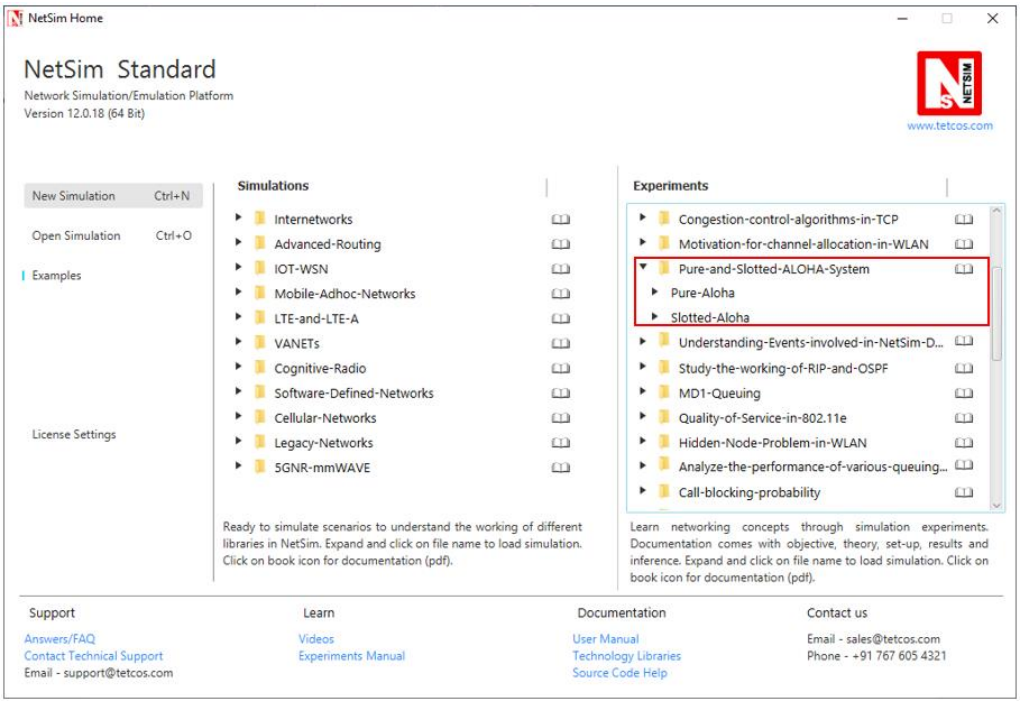
Bandwidth is 10 Mbps and hence, packet time comes as 1.2 milliseconds.

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

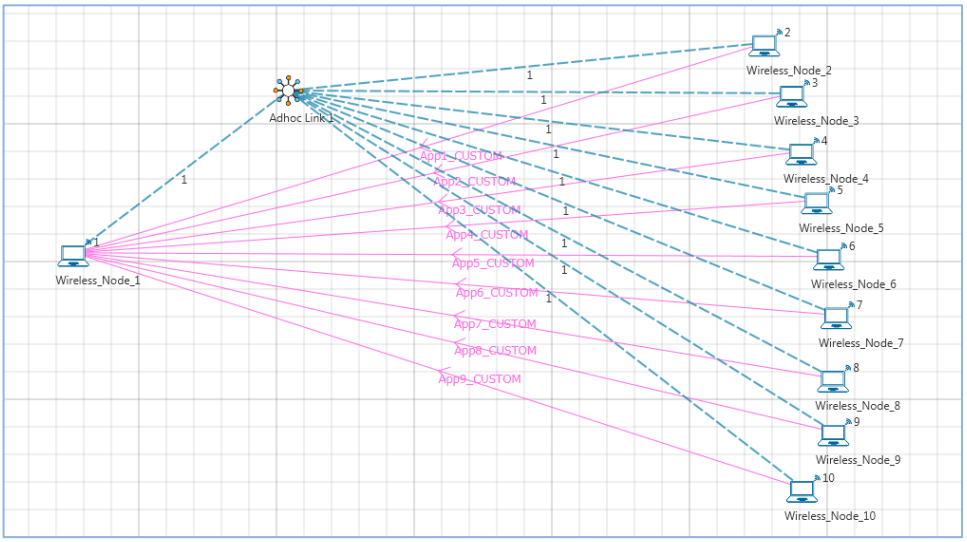
10.2 Network Set Up:

Part-1

Open NetSim and click **Examples > Experiments > Pure-and-Slotted-Aloha-System** 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.)

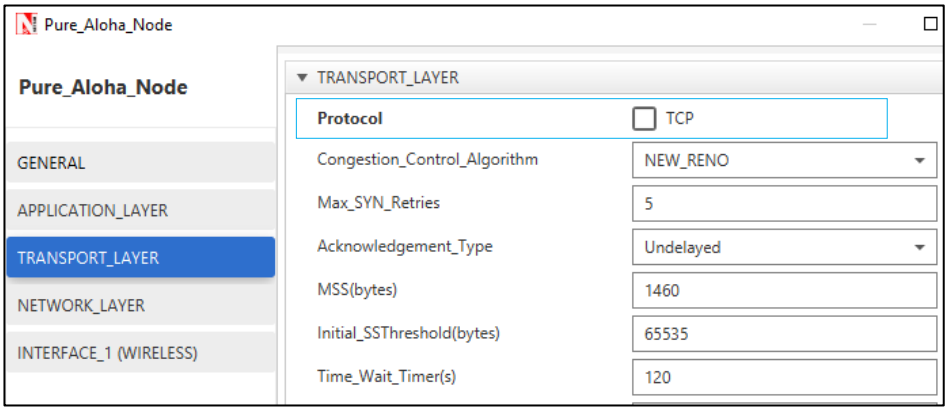
Node 2, 3, 4, 5, 6, 7, 8, 9, and 10 generates traffic. The properties of Node 2, 3, 4, 5, 6, 7, 8, 9, and 10 which transmits data to Node 1 are selected as follows

Wireless Node Properties:

Wireless Node Properties

Transport Layer	
TCP	**disable**
Interface1_Wireless (PHYSICAL_LAYER)	
Data Rate(mbps)	**10**
Interface1_Wireless (DATALINK_LAYER)	
Retry_Limit	**0**
MAC_Buffer	**FALSE**
Slot Length(µs)	**1200**

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



Right click on the Adhoc link and select the channel characteristics as **no path loss**.

Application Properties: Right click on the Application icon and set following properties as shown in below figure:

Application_1 Properties		
Application Method	**Unicast**	
Application Type	**Custom**	
Source_Id	**2**	
Destination_Id	**1**	
Packet Size	**Distribution**	**Constant**
	Value (bytes)	**1460**
Inter Arrival Time	**Distribution**	**Exponential**
	Packet Inter Arrival Time (µs)	**200000**

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: The Simulation Time can be selected only after doing the following two tasks,

- ***Set the properties of Nodes***
- ***Then click on Run Simulation button***
- ***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.

10.3 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:

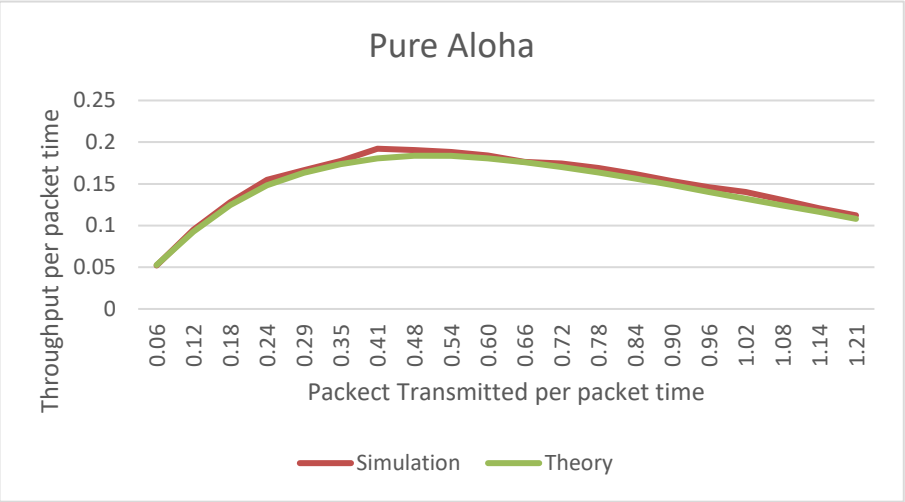
Number of nodes generating traffic	Total number of Packets Transmitted	Total number of Packets Collided	Total number of Packets Success (Packets Transmitted -Packets Collided)	Throughput per packet time(G)	Number of Packets Transmitted per packet time(S)	Packets per packet time theoretical ($S = G * e^{-2G}$)
9	494	60	434	0.05928	0.05208	0.05265
19	978	187	791	0.11736	0.09492	0.09281
29	1482	415	1067	0.17784	0.12804	0.12461
39	1991	700	1291	0.23892	0.15492	0.14816
49	2443	1056	1387	0.29316	0.16644	0.16311
59	2907	1429	1478	0.34884	0.17736	0.17363
69	3435	1834	1601	0.4122	0.19212	0.18075
79	3964	2377	1587	0.47568	0.19044	0.18371

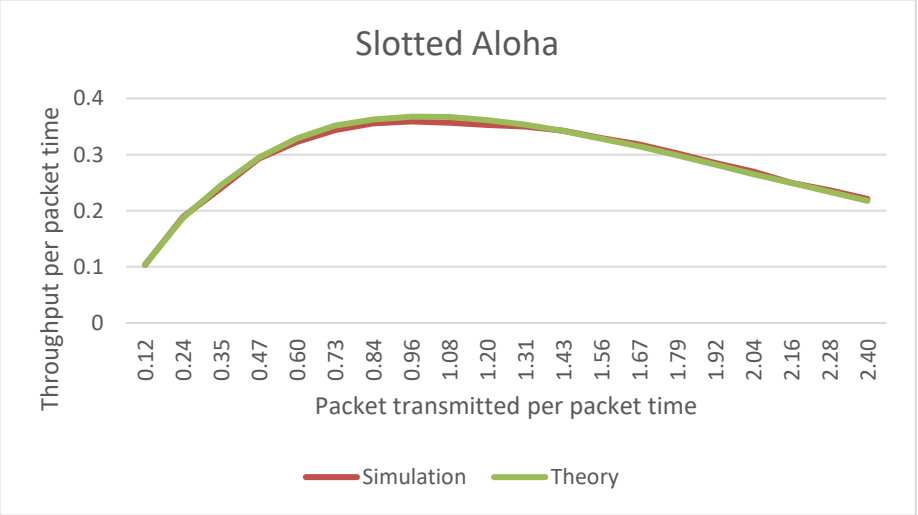
89	4468	2902	1566	0.53616	0.18792	0.18348
99	4998	3468	1530	0.59976	0.1836	0.18073
109	5538	4073	1465	0.66456	0.1758	0.17592
119	6023	4574	1449	0.72276	0.17388	0.1703
129	6503	5102	1401	0.78036	0.16812	0.16386
139	6992	5650	1342	0.83904	0.16104	0.15668
149	7481	6208	1273	0.89772	0.15276	0.14907
159	7998	6787	1211	0.95976	0.14532	0.14078
169	8507	7341	1166	1.02084	0.13992	0.13252
179	9008	7924	1084	1.08096	0.13008	0.12442
189	9486	8483	1003	1.13832	0.12036	0.11682
199	10025	9093	932	1.203	0.11184	0.10848

Slotted Aloha:

Number of nodes generating traffic	Total number of Packets Transmitted	Total number of Packets Collided	Total number of Packets Success (Packets Transmitted -Packets Collided)	Throughput per packet time(G)	Number of Packets Transmitted per packet time(S)	Packets per packet time theoretical ($S = G * e^{-G}$)
19	974	111	863	0.11688	0.10356	0.10399
39	1981	407	1574	0.23772	0.18888	0.18742
59	2893	891	2002	0.34716	0.24024	0.24534
79	3946	1504	2442	0.47352	0.29304	0.29491
99	4976	2286	2690	0.59712	0.3228	0.32865
119	5996	3144	2852	0.71952	0.34224	0.3504
139	6961	3999	2962	0.83532	0.35544	0.36231
159	7971	4979	2992	0.95652	0.35904	0.36752
179	8969	5994	2975	1.07628	0.357	0.36686
199	9983	7042	2941	1.19796	0.35292	0.36156
219	10926	8011	2915	1.31112	0.3498	0.35337
239	11928	9073	2855	1.43136	0.3426	0.34207
259	12969	10224	2745	1.55628	0.3294	0.32825
279	13916	11266	2650	1.66992	0.318	0.31438
299	14945	12430	2515	1.7934	0.3018	0.29841

319	15967	13592	2375	1.91604	0.285	0.28202
339	17011	14765	2246	2.04132	0.26952	0.26508
359	17977	15895	2082	2.15724	0.24984	0.24947
379	18983	17010	1973	2.27796	0.23676	0.23348
399	19987	18146	1841	2.39844	0.22092	0.21792



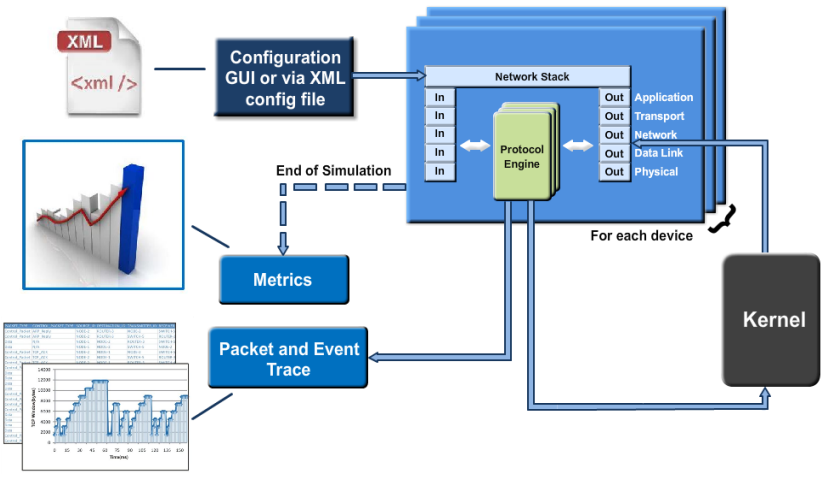


Thus the following characteristic plot for the Pure ALOHA and Slotted ALOHA is obtained, which matches the theoretical result.

11. 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

11.1 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.

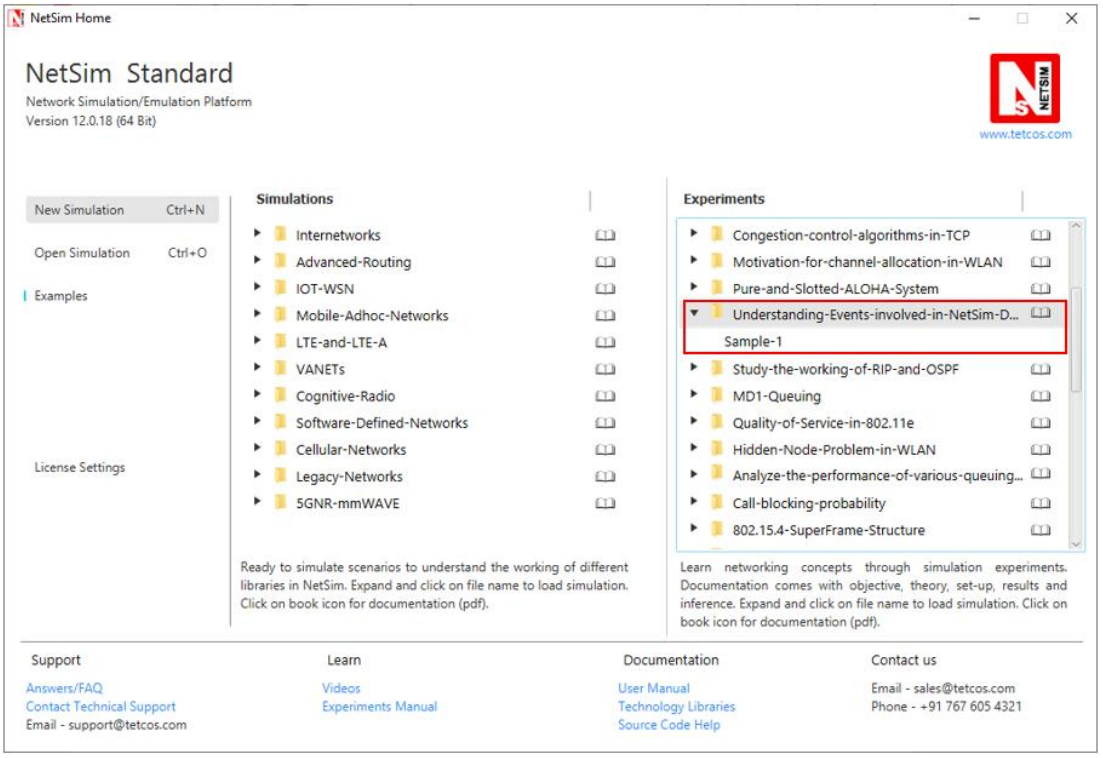
Each event gets added in the Simulation kernel by the protocol operating at the particular layer of the Network Stack. The required sub events are passed into the Simulation kernel. These sub events are then fetched by the Network Stack in order to execute the functionality of each protocol. At the end 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:

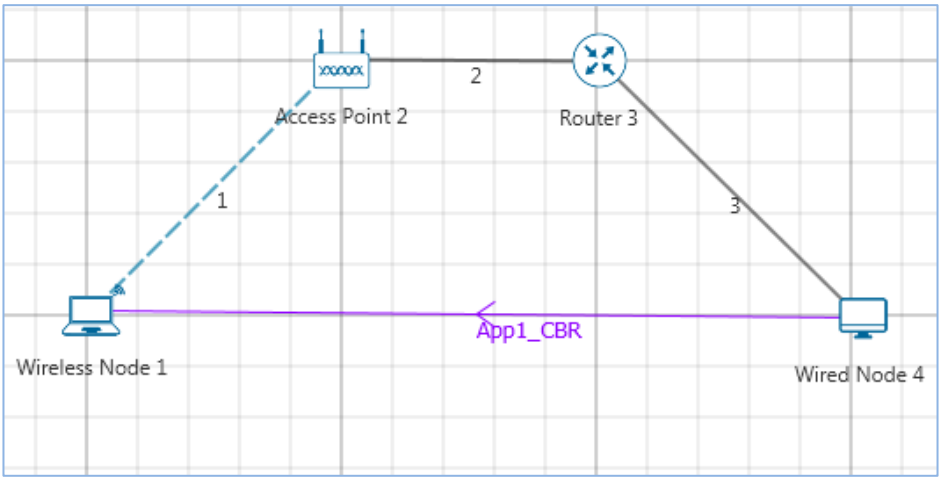
The event trace records every single event along with associated information such as time stamp, event ID, event type etc. in a text file or .csv file which can be stored at a user defined location.

11.2 Network Setup:

Open NetSim and click **Examples > Experiments > Understanding-Events-involved-in-NetSim-DES > Sample-1** as shown below:



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



11.3 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:

Device Positions				
	Access Point 2	Wired Node 4	Wireless Node 1	Router 3
X/Lat	150	250	100	200
Y/Lon	50	100	100	50

Step 3: Go to Wireless Link Properties, the **"Channel Characteristics"** is set to NO PATHLOSS.

Step 4: TCP Protocol is set to Disable in all the devices.

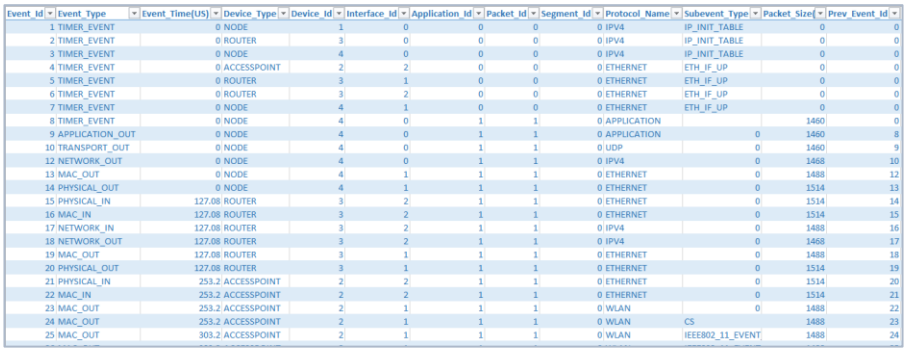
Step 5: Right click on the Application Flow **App1 CBR** and select Properties or click on the Application icon present in the top ribbon/toolbar.

A CBR Application is generated from Wired Node 4 i.e. Source to Wireless Node 1 i.e. Destination with Packet Size remaining 1460Bytes and Inter Arrival Time remaining 20000µs.

Step 6: 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.

11.4 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:



Event_Id	Event_Type	Event_Time(US)	Device_Type	Device_Id	Interface_Id	Application_Id	Packet_Id	Segment_Id	Protocol_Name	Subevent_Type	Packet_Size	Prev_Event_Id
1	TIMER_EVENT	0	NODE	1	0	0	0	0	IPV4	IP_INIT_TABLE	0	0
2	TIMER_EVENT	0	ROUTER	3	0	0	0	0	IPV4	IP_INIT_TABLE	0	0
3	TIMER_EVENT	0	NODE	4	0	0	0	0	IPV4	IP_INIT_TABLE	0	0
4	TIMER_EVENT	0	ACCESSPOINT	2	2	0	0	0	ETHERNET	ETH_IF_UP	0	0
5	TIMER_EVENT	0	ROUTER	3	1	0	0	0	ETHERNET	ETH_IF_UP	0	0
6	TIMER_EVENT	0	ROUTER	3	2	0	0	0	ETHERNET	ETH_IF_UP	0	0
7	TIMER_EVENT	0	NODE	4	1	0	0	0	ETHERNET	ETH_IF_UP	0	0
8	TIMER_EVENT	0	NODE	4	0	1	1	0	APPLICATION		1460	0
9	APPLICATION_OUT	0	NODE	4	0	1	1	0	APPLICATION	0	1460	8
10	TRANSPORT_OUT	0	NODE	4	0	1	1	0	UDP	0	1460	9
12	NETWORK_OUT	0	NODE	4	0	1	1	0	IPV4	0	1468	10
13	MAC_OUT	0	NODE	4	1	1	1	0	ETHERNET	0	1488	12
14	PHYSICAL_OUT	0	NODE	4	1	1	1	0	ETHERNET	0	1514	13
15	PHYSICAL_IN	127.08	ROUTER	3	2	1	1	0	ETHERNET	0	1514	14
16	MAC_IN	127.08	ROUTER	3	2	1	1	0	ETHERNET	0	1514	15
17	NETWORK_IN	127.08	ROUTER	3	2	1	1	0	IPV4	0	1488	16
18	NETWORK_OUT	127.08	ROUTER	3	2	1	1	0	IPV4	0	1468	17
19	MAC_OUT	127.08	ROUTER	3	1	1	1	0	ETHERNET	0	1488	18
20	PHYSICAL_OUT	127.08	ROUTER	3	1	1	1	0	ETHERNET	0	1514	19
21	PHYSICAL_IN	253.2	ACCESSPOINT	2	2	1	1	0	ETHERNET	0	1514	20
22	MAC_IN	253.2	ACCESSPOINT	2	2	1	1	0	ETHERNET	0	1514	21
23	MAC_OUT	253.2	ACCESSPOINT	2	1	1	1	0	WLAN	0	1488	22
24	MAC_OUT	253.2	ACCESSPOINT	2	1	1	1	0	WLAN	CS	1488	23
25	MAC_OUT	303.2	ACCESSPOINT	2	1	1	1	0	WLAN	IEEE802_11_EVENT	1488	24

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.

11.4.1 Events Involved:

Events are listed in the following format:

[EVENT_TYPE, EVENT_TIME, PROTOCOL, EVENT_NO, SUBEVENT_TYPE]

[APP_OUT, 20000, APP, 6, -]

[TRNS_OUT, 20000, UDP, 7, -]

[NW_OUT, 20000, IPV4, 9, -]

[MAC_OUT, 20000, ETH, 10, -]

[MAC_OUT, 20000, ETH, 11, CS]

[MAC_OUT, 20000.96, ETH, 12, IFG]

[PHY_OUT, 20000.96, ETH, 13, -]

[PHY_OUT, 20122.08, ETH, 14, PHY_SENSE]

[PHY_IN, 20127.08, ETH, 15, -]

[MAC_IN, 20127.08, ETH, 16, -]

[NW_IN, 20127.08, IPV4, 17, -]

[NW_OUT, 20127.08, IPV4, 18, -]

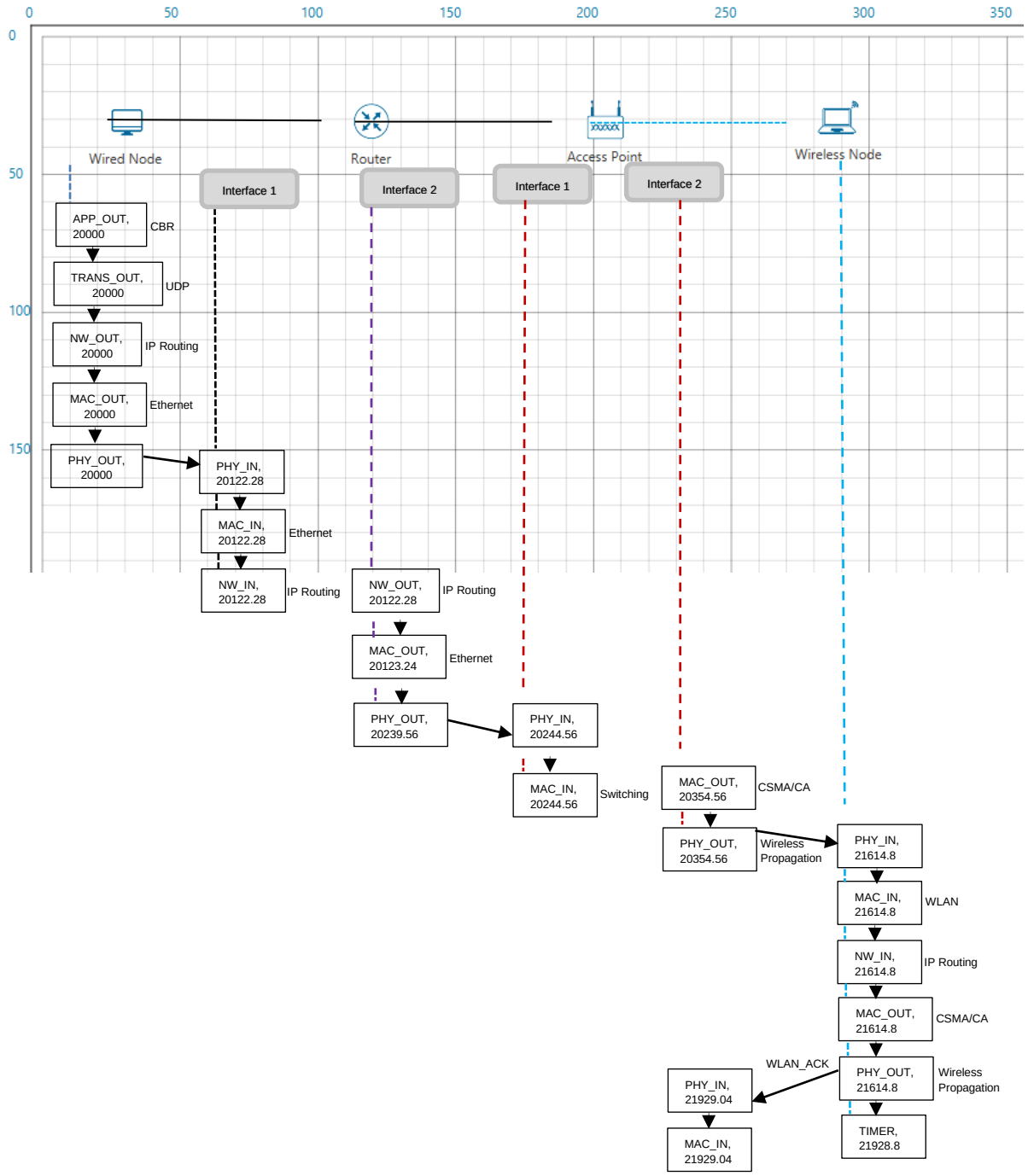
[MAC_OUT, 20127.08, ETH, 19, -]

[MAC_OUT,	20127.08,	ETH,	20,	CS]
[MAC_OUT,	20128.04,	ETH,	21,	IFG]
[PHY_OUT,	20128.04,	ETH,	22,	-]
[PHY_OUT,	20249.16,	ETH,	23,	PHY_SENSE]
[PHY_IN,	20254.16,	ETH,	24,	-]
[MAC_IN,	20254.16,	ETH,	25,	-]
[MAC_OUT,	20254.16,	WLAN,	26,	-]
[MAC_OUT,	20254.16,	WLAN,	27,	DIFS_END]
[MAC_OUT,	20304.16,	WLAN,	28,	BACKOFF]
[MAC_OUT,	20324.16,	WLAN,	29,	BACKOFF]
[MAC_OUT,	20344.16,	WLAN,	30,	BACKOFF]
[MAC_OUT,	20364.16,	WLAN,	31,	BACKOFF]
[PHY_OUT,	20364.16,	WLAN,	32,	-]
[TIMER,	21668.16,	WLAN,	35,	UPDATE_DEVICE_STATUS]
[PHY_IN,	21668.4,	WLAN,	33,	-]
[MAC_IN,	21668.4,	WLAN,	36,	RECEIVE_MPDU]
[NW_IN,	21668.4,	IPV4,	37,	-]
[MAC_OUT,	21668.4,	WLAN,	38,	SEND_ACK]
[TRNS_IN,	21668.4,	UDP,	39,	-]
[APP_IN,	21668.4,	APP,	41,	-]
[PHY_OUT,	21678.4,	WLAN,	40,	-]
[TIMER,	21982.4,	WLAN,	43,	UPDATE_DEVICE]
[PHY_IN,	21982.63,	WLAN,	42,	-]

[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:

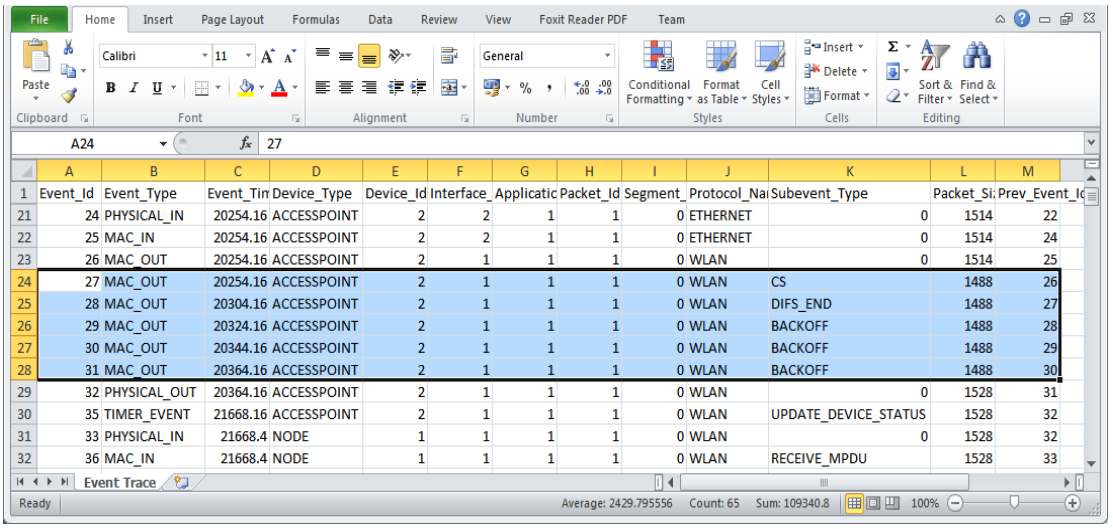


For Example:

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.



Event_Id	Event_Type	Event_Tim	Device_Type	Device_Id	Interface_	Applicatic	Packet_Id	Segment_	Protocol_Nai	Subevent_Type	Packet_Si	Prev_Event_Id
24	PHYSICAL_IN	20254.16	ACCESSPOINT	2	2	1	1	0	ETHERNET	0	1514	22
25	MAC_IN	20254.16	ACCESSPOINT	2	2	1	1	0	ETHERNET	0	1514	24
26	MAC_OUT	20254.16	ACCESSPOINT	2	1	1	1	0	WLAN	0	1514	25
27	MAC_OUT	20254.16	ACCESSPOINT	2	1	1	1	0	WLAN	CS	1488	26
28	MAC_OUT	20304.16	ACCESSPOINT	2	1	1	1	0	WLAN	DIFS_END	1488	27
29	MAC_OUT	20324.16	ACCESSPOINT	2	1	1	1	0	WLAN	BACKOFF	1488	28
30	MAC_OUT	20344.16	ACCESSPOINT	2	1	1	1	0	WLAN	BACKOFF	1488	29
31	MAC_OUT	20364.16	ACCESSPOINT	2	1	1	1	0	WLAN	BACKOFF	1488	30
32	PHYSICAL_OUT	20364.16	ACCESSPOINT	2	1	1	1	0	WLAN	0	1528	31
35	TIMER_EVENT	21668.16	ACCESSPOINT	2	1	1	1	0	WLAN	UPDATE_DEVICE_STATUS	1528	32
33	PHYSICAL_IN	21668.4	NODE	1	1	1	1	0	WLAN	0	1528	32
36	MAC_IN	21668.4	NODE	1	1	1	1	0	WLAN	RECEIVE_MPDU	1528	33

In this manner the event trace can be used to understand the flow of events in NetSim Discrete Event Simulator.

11.5 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.

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

12.1 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.

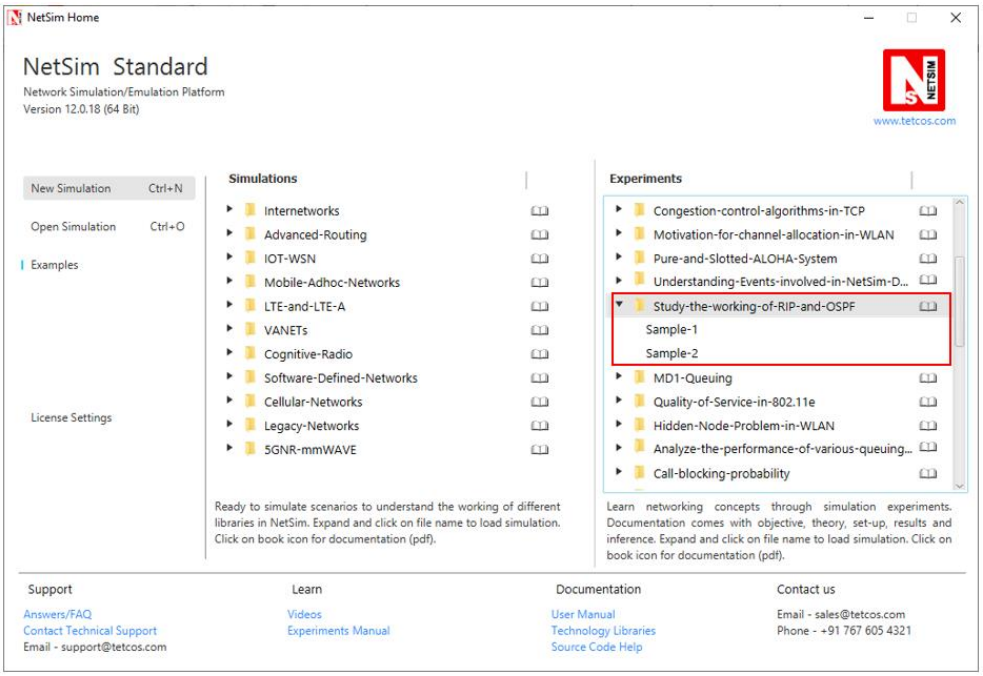
In OSPF, each router has a link state database which is tabular representation of the topology of the network (including cost). Using Dijkstra algorithm each router finds the shortest path between source and destination.

Formation of OSPF Routing Table

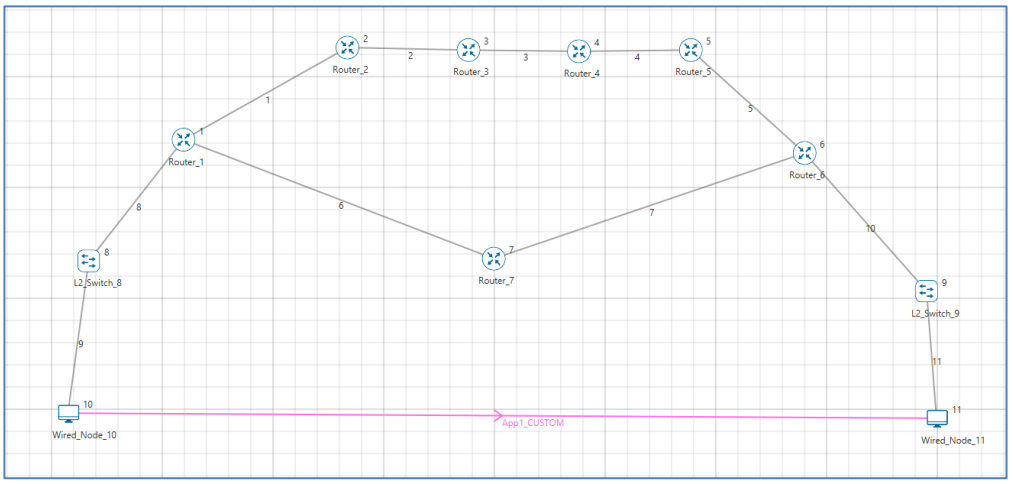
1. OSPF-speaking routers send Hello packets out all OSPF-enabled interfaces. If two routers sharing a common data link agree on certain parameters specified in their respective Hello packets, they will become neighbors.
2. Adjacencies, which can be thought of as virtual point-to-point links, are formed 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

12.2 Network Setup:

Open NetSim and click **Examples > Experiments > Study-the-working-of-RIP-and-OSPF > Sample-1** as shown below:



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



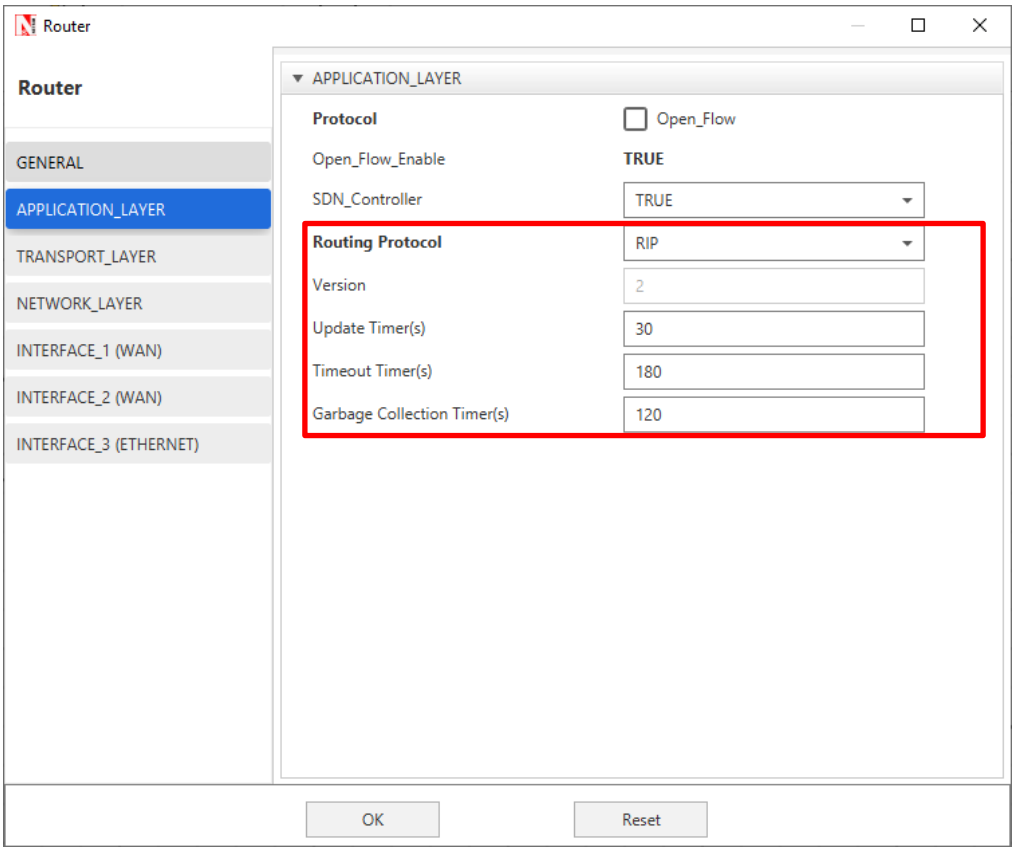
12.3 Procedure:

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: In the Source Node, i.e. Wired Node 10, TCP Protocol in the Transport Layer is disabled.

Step 4: 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.

Step 5: 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.

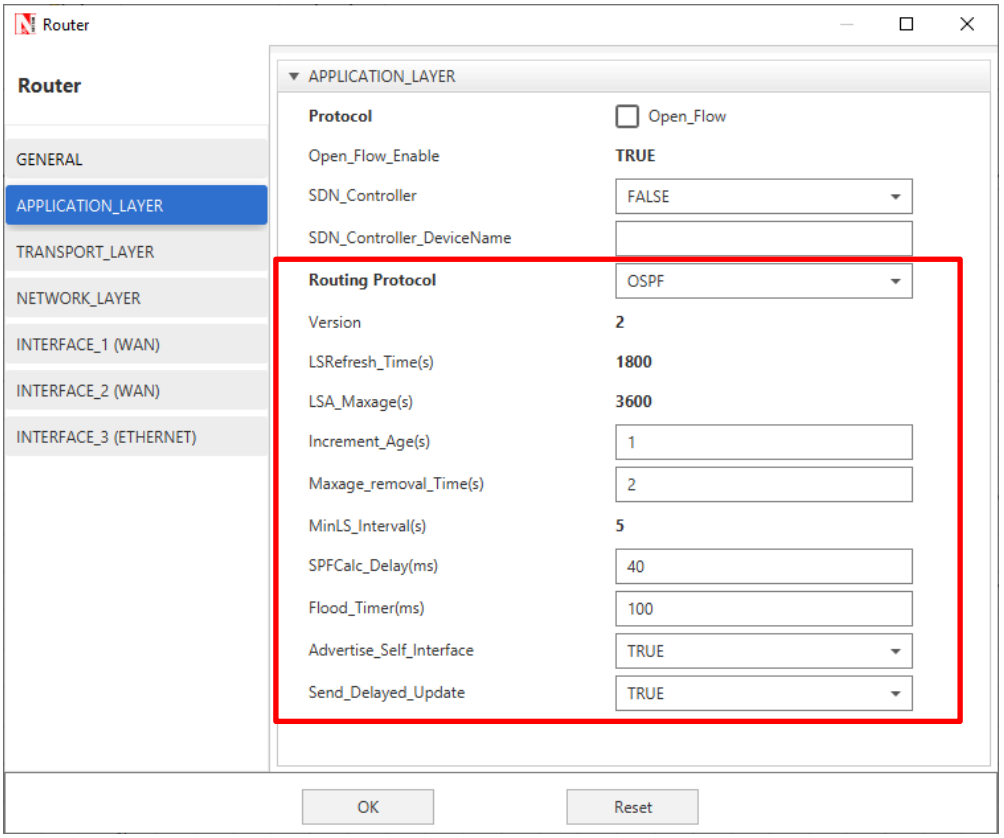
Step 6: Run the Simulation for 100 Seconds.

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.

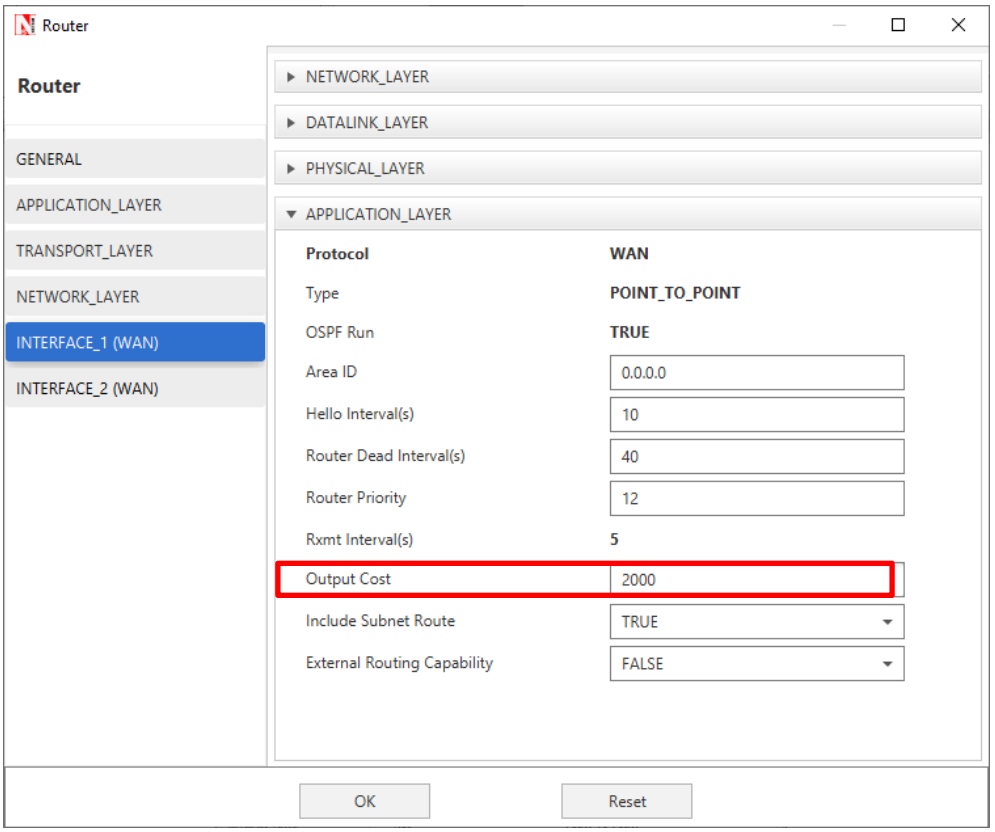
Step 2: Go to Router 1 Properties. In the Application Layer, Routing Protocol is set as OSPF.



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: In the Source Node, i.e. Wired Node 10, TCP Protocol in the Transport Layer is disabled.

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



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.

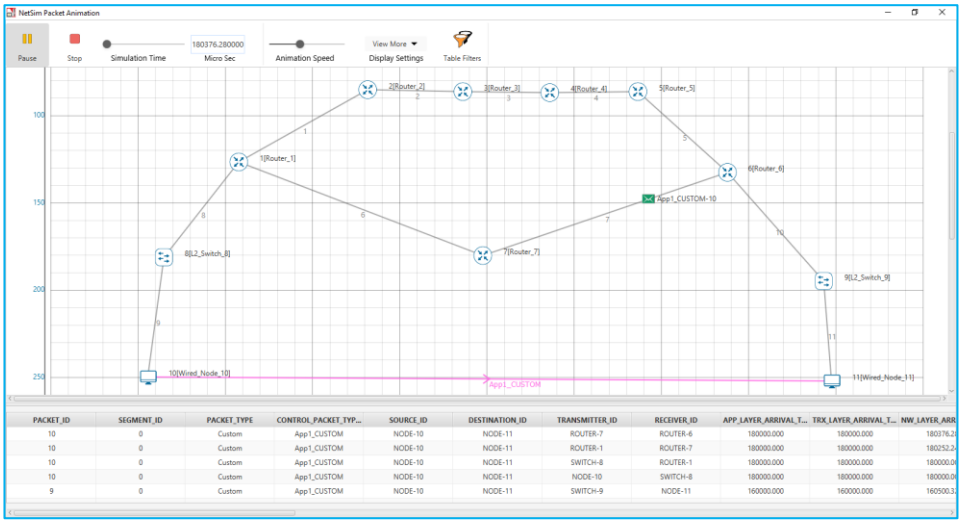
Additionally, the "**Start Time (s)**" parameter is set to 30, while configuring the application. This time is usually set to be greater than the time taken for OSPF Convergence (i.e. Exchange of OSPF information between all the routers), and it increases as the size of the network increases.

Step 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.

12.4 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:

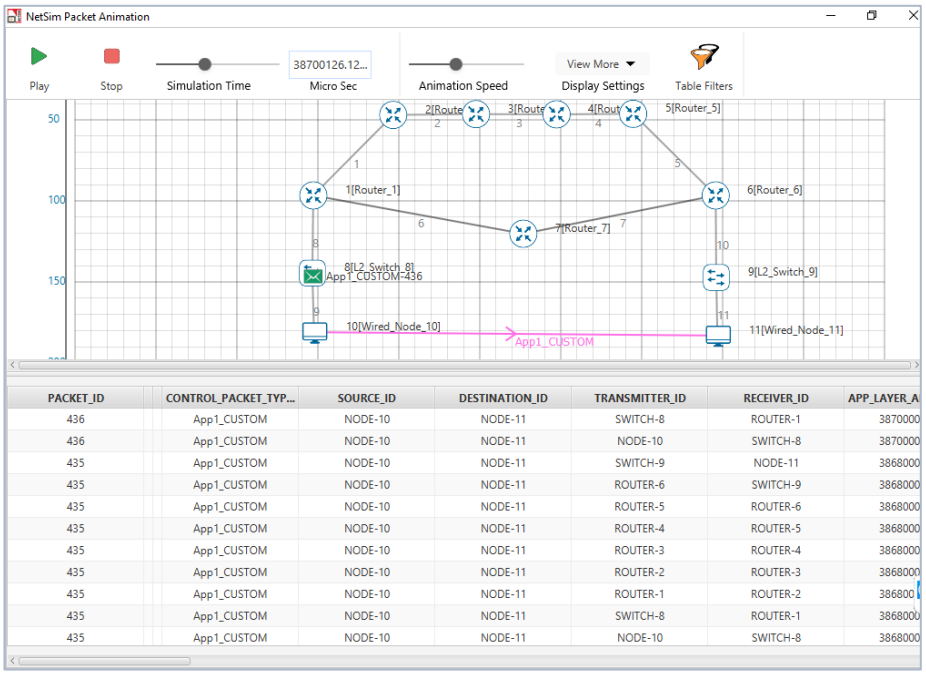


Users can view the same in Packet Trace.

Shortest Path from Wired Node 10 to Wired Node 11 in RIP is **Wired Node 10->L2 Switch 8->Router 1->Router 7->Router 6->L2 Switch 9->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.

12.5 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:



PACKET_ID	CONTROL_PACKET_TYP...	SOURCE_ID	DESTINATION_ID	TRANSMITTER_ID	RECEIVER_ID	APP_LAYER_A
436	App1_CUSTOM	NODE-10	NODE-11	SWITCH-8	ROUTER-1	3870000
436	App1_CUSTOM	NODE-10	NODE-11	NODE-10	SWITCH-8	3870000
435	App1_CUSTOM	NODE-10	NODE-11	SWITCH-9	NODE-11	3868000
435	App1_CUSTOM	NODE-10	NODE-11	ROUTER-6	SWITCH-9	3868000
435	App1_CUSTOM	NODE-10	NODE-11	ROUTER-5	ROUTER-6	3868000
435	App1_CUSTOM	NODE-10	NODE-11	ROUTER-4	ROUTER-5	3868000
435	App1_CUSTOM	NODE-10	NODE-11	ROUTER-3	ROUTER-4	3868000
435	App1_CUSTOM	NODE-10	NODE-11	ROUTER-2	ROUTER-3	3868000
435	App1_CUSTOM	NODE-10	NODE-11	ROUTER-1	ROUTER-2	386800
435	App1_CUSTOM	NODE-10	NODE-11	SWITCH-8	ROUTER-1	3868000
435	App1_CUSTOM	NODE-10	NODE-11	NODE-10	SWITCH-8	3868000

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.

12.6 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 it's 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.***

13. M/D/1 Queuing

13.1 Objective:

To create an M/D/1 queue: a source to generate packets, a queue to act as the buffer and server, a sink to dispose of serviced packets and to study how the queuing delay of such a system varies.

13.2 Theory:

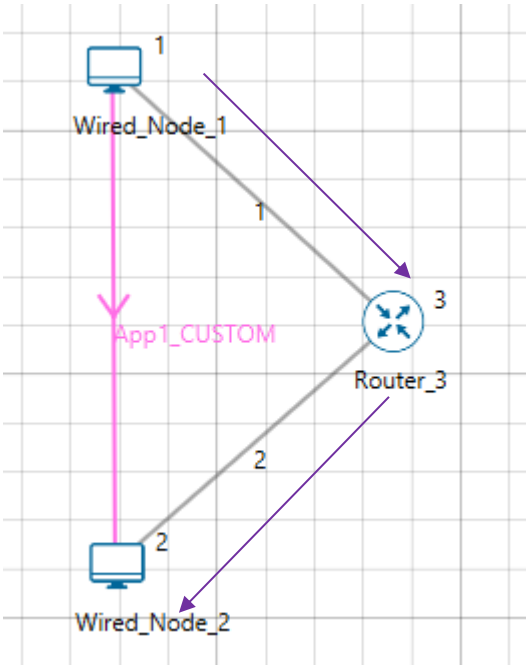
In systems where the service time is a constant, the M/D/1, single-server queue model, can be used. Following Kendall's notation, M/D/1 indicates a system where:

- **Arrivals** are a Poisson process with parameter **λ**
- **Service time(s)** is deterministic or constant
- There is **one server**

For an M/D/1 model, the total expected queuing time is $\boldsymbol{T = \frac{1}{2\mu} \times \frac{\rho}{1-\rho}}$

Where μ **=** Service Rate = 1/Service time and ρ is the utilization given as follows, $\boldsymbol{\rho = \frac{\lambda}{\mu}}$

To model an M/D/1 system in NetSim, we use the following model



Traffic flow from Node 1 to Node 2 (Node 1: Source, Node 2: Sink) **Inter-arrival time**: Exponential Distribution with mean 2000 μs **Packet size**: Constant Distribution with mean of 1250 bytes

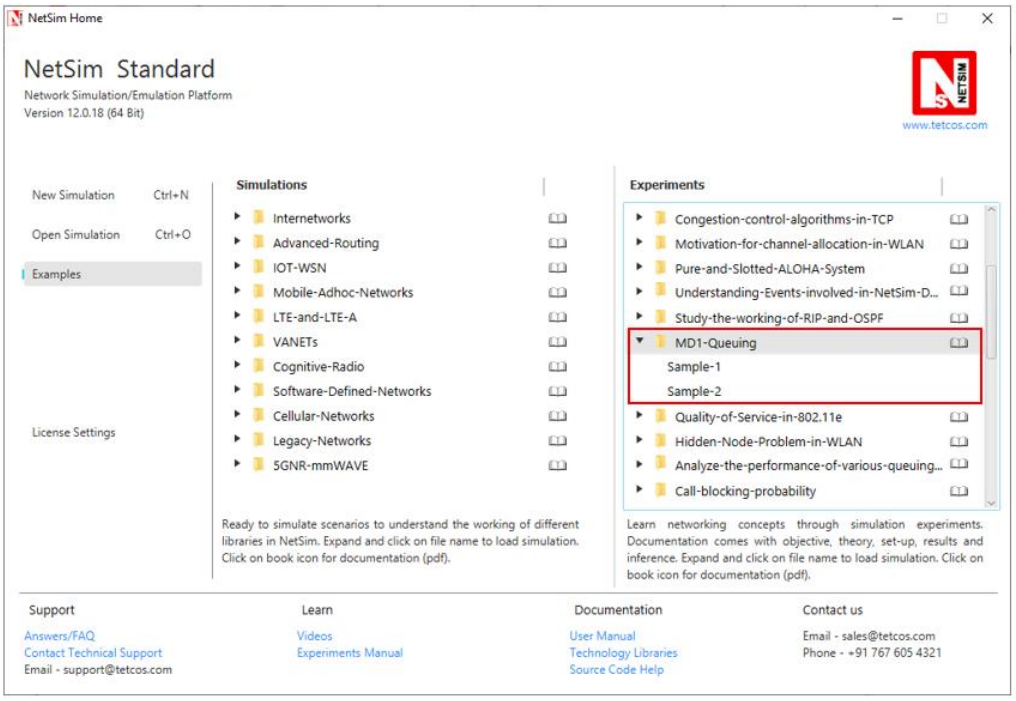
Note:

1. Exponentially distributed inter-arrivals times give us a Poisson arrival process. Different mean values are chosen as explained in the section Sample Inputs. (Dropping the devices in different order may change the result because the random number generator will get initialized differently)

2. To get constant service times, we use constant distribution for packet sizes. Since, the service (which in our case is link transmission) times are directly proportional to packet size (greater the packet size, greater the time for transmission through a link), a constant packet size leads to a constant service time.

13.3 Network Setup:

Open NetSim and Click on **Examples > Experiments > MD1-Queuing > Sample-1** as shown below:



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

13.4 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: TCP Protocol is disabled in both the Wired Nodes.

Step 3: Link Properties are set as per the table given below:

Link Properties	Link 1	Link 2
Uplink Speed (Mbps)	10	**10**
Downlink Speed (Mbps)	10	**10**
Uplink BER	0	**0**
Downlink BER	0	**0**
Uplink Propagation Delay (µs)	0	**0**
Downlink Propagation Delay (µs)	**0**	**0**

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 Wired Node 1 i.e. Source to Wired Node 2 i.e. Destination with Packet Size set to 1250 Bytes and Inter Arrival Time set to 2000 µs.

The Packet Size and Inter Arrival Time parameters are set such that the Generation Rate equals 2.5 Mbps. Generation Rate can be calculated using the formula:

$$Generation\ Rate\ (Mbps) = Packet\ Size\ (Bytes) * 8/Interarrival\ time\ (\mu s)$$

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: Run the Simulation for 100 Seconds.

13.5 Observation:

Even though the packet size at the application layer is 1250 bytes, as the packet moves down the layers, some overhead is added which results in a greater packet size. This is the actual payload that is transmitted by the physical layer. The overheads added in different layers are shown in the below table and can be obtained from the packet trace:

Layer	Overhead (Bytes)
Transport Layer	**8**

Network Layer	**20**
MAC layer	**26**
Physical Layer	**0**
Total	**54**

Therefore, the payload size = Packet Size + Overhead

= 1250 + 54

= **1304 bytes**

Theoretical Calculation:

By formula,

$$\boldsymbol{Queuing\ Time} = \boldsymbol{T} = \frac{\mathbf{1}}{\mathbf{2}\boldsymbol{\mu}} \times \frac{\boldsymbol{\rho}}{\mathbf{1} - \boldsymbol{\rho}}$$

μ = Service Rate, i.e., the time taken to service each packet

= Link capacity (bps) / (Payload Size (Bytes) * 8)

= (10×10^6) / (1304*8)

= **958.59 packets / sec**

λ = Arrival rate, i.e., the rate at which packets arrive (Packets per second)

Inter-arrival time = 2,000 micro sec

Arrival rate **λ** = 1/ Inter Arrival time

= 1/2000 micro sec

= **500 packets / sec**

ρ = Utilization

= λ/μ

= 500/958.59

= 0.522

By formula, **Queuing Time** = $\frac{1}{2\times958.59} \times \frac{0.522}{1-0.522}$ = **569.61 micro sec**

13.6 Output:

After running the simulation, check the **"Delay"** in the Application Metrics.

Delay = 2654.9 micro sec

This Delay (also known as Mean Delay) is the sum of Queuing Delay, Total Transmission time and Routing Delay.

$$\begin{pmatrix}\textbf{Mean}\\\textbf{Delay}\end{pmatrix} = \begin{pmatrix}\textbf{Queuing}\\\textbf{Delay}\end{pmatrix} + \begin{pmatrix}\textbf{Total Transmission}\\\textbf{Time}\end{pmatrix} + \begin{pmatrix}\textbf{Routing}\\\textbf{Delay}\end{pmatrix}$$

Total Transmission Time is the sum of transmission time through Link 1 and Link 2.

Transmission time through each link is the same and is given by:

Transmission time through each link = $\frac{Payload\ Size\ (Bytes)\times 8}{Uplink\ Speed\ (bps)}$

$$= \frac{1304\times 8}{10\times 10^6}$$

= 1000.0 micro sec

Routing Delay is approximately 1 micro sec and can be found from the Event Trace. It is the difference between "Physical In" and "Physical Out" time for the Router.

Therefore, for simulation

Queuing Delay = 2654.9– (2 × 1000.0) – 1 = 653.9 micro sec

Sample 2

Keeping all the other parameters same as in previous example, if Packet Inter Arrival Time is taken as 1500 micro sec, then

λ = 666.67 packets per sec

Utilization **ρ = λ/μ = 666.67/958.59 = 0.695**

And **Queuing Time T = 1188.56 micro sec**

From NetSim,

Delay =3277.31 micro sec

Therefore, **Queuing Time** = 3277.31 - (2×1000.0) – 1 = **1276.3 micro sec**

NOTE: ***Obtained value is slightly higher than the theoretical value because of initial delays in forming ARP table, Switch table and Routing table etc.***

A Note on M/M/1 queuing in NetSim:

M/M/1 queue can be generated similarly by setting the "**Packet Size Distribution**" as "**Exponential**" instead of "**Constant**". However, the results obtained from simulation deviate from the theoretical value because of the effect of packet fragmentation. Whenever a packet with size greater than Transport Layer MSS and / or MAC Layer MTU (which is 1500 bytes in NetSim) is generated, it gets fragmented in the application layer. Then the packet is sent as multiple frames, and makes it impossible to calculate the exact queuing time.

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

14.1 Theory:

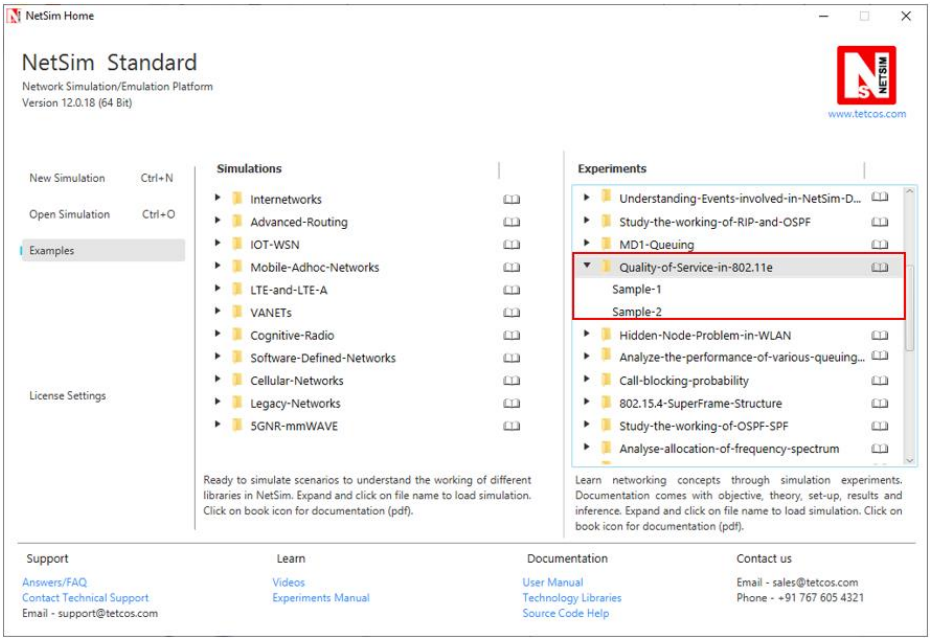
IEEE 802.11e Medium Access Control (MAC) is a supplement to the IEEE 802.11 Wireless Local Area Network (WLAN) standard to support Quality-of-Service (QoS). When 802.11e is enabled high-priority traffic has a higher chance of being sent than low-priority traffic: an application with high priority traffic 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:

Application Type	Priority Value	Priority	QoS Class
Voice – One way **Voice – Two way**	8 8	Medium High	RTPS UGS
Video	6	Low	nRTPS
FTP	2	Low	BE
Database	2	Low	BE
Custom	2	Low	BE

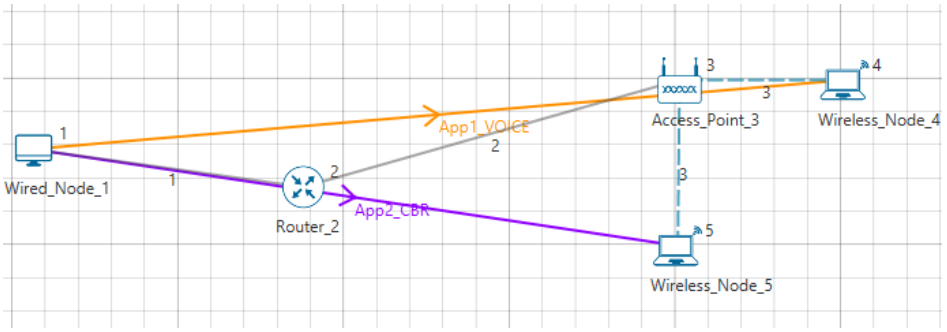
eRTPS QoS class is available in NetSim which has a priority value of 4. The QoS class for each application mentioned in the table above is fixed and can be changed by the user.

14.2 Network Setup:

Open NetSim and click on **Examples > Experiments > Quality-of-Service-in-802.11e > Sample-1** as shown below:



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



14.3 Procedure:

Sample 1:

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

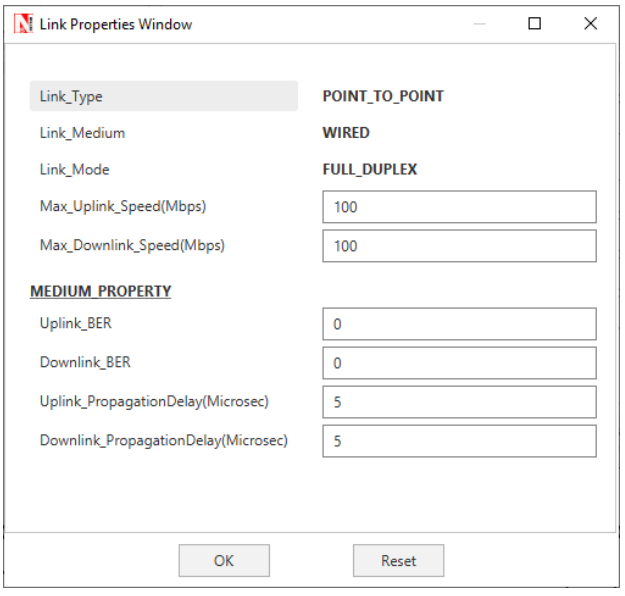
Step 1: A network scenario is designed in NetSim GUI comprising of 1 Wired Node, 2 Wireless Nodes, 1 Router, and 1 Access Point in the **"Internetworks"** Network Library.

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

	Access Point 2	Wireless Node 4	Wireless Node 5
X/Lat	250	300	250
Y/Lon	100	100	150

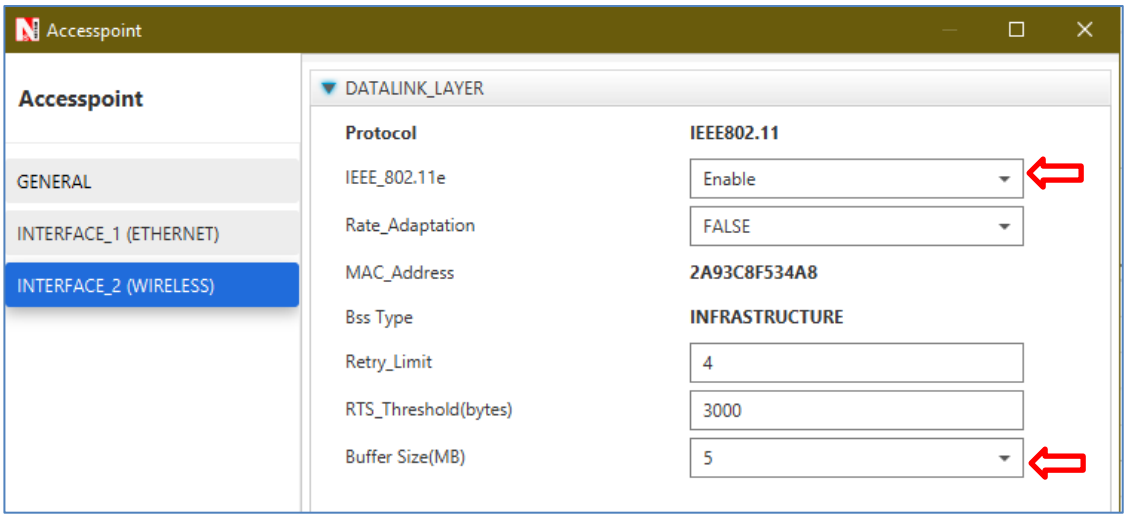
Step 3: TCP Protocol is set to Disable in all the devices.

Step 4: Wired Link Properties is set as follows:



Step 5: Go to Wireless Link Properties, the **"Channel Characteristics"** is set to NO PATHLOSS.

Step 6: 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.



Step 7: 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.

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:

$$Generation\ Rate\ (Mbps)\ =\ Packet\ Size\ (Bytes)\ *\ 8/Interarrival\ time\ (\mu s)$$

Step 8: 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.

14.4 Output:

IEEE 802.11e	Application	Generation rate (Mbps)	Throughput (Mbps)	Delay (Micro. Sec.)
Enable (Sample 1)	Voice	10	3.22	945561.8
	CBR	10	2.14	6466262.9
Disable (Sample 2)	Voice	10	2.64	3672706.7
	CBR	10	2.64	3671315.4

14.5 Inference:

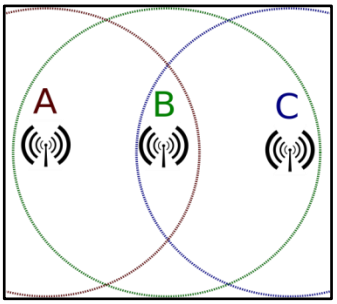
In sample 1, since QoS is enabled voice sees a higher priority than CBR. Hence voice packets in the queue are first transmitted before CBR packets are transmitted. In sample 2, since QoS has been disabled, priority is not considered for the applications. Hence they both see the same throughput.

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.

15. Study the hidden node problem in WLAN

15.1 Theory:

Hidden nodes in a wireless network are nodes that are out of range of other nodes or a collection of nodes. In a wireless network, it is likely that the node at the far edge of the access point's range, which is known as **A**, can see the access point, but it is unlikely that the same node can see a node on the opposite end of the access point's range, **C**. These nodes are known as *hidden*.

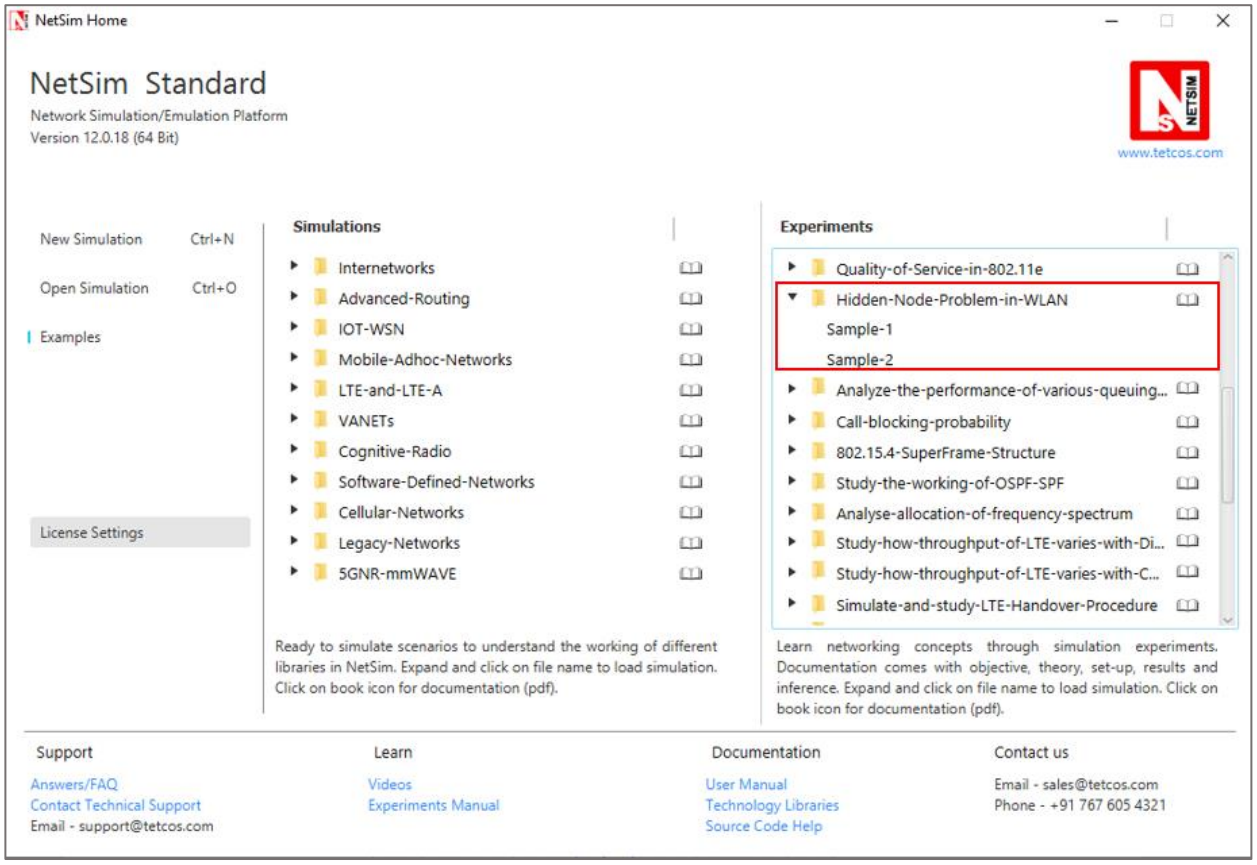


The problem is when nodes A and C start to send packets simultaneously to the access point B. Because the nodes A and C are out of range of each other and so cannot detect a collision while transmitting, Carrier sense multiple access with collision detection (CSMA/CD) does not work, and collisions occur, which then corrupt the data received by the access point.

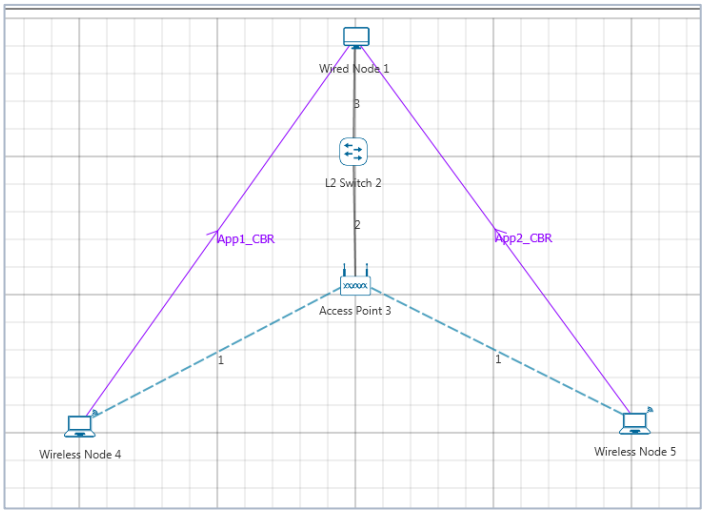
To overcome the hidden node problem, RTS/CTS handshaking (IEEE 802.11 RTS/CTS) is implemented in conjunction with the Carrier sense multiple access with collision avoidance (CSMA/CA) scheme. The same problem exists in a MANET.

15.2 Network Setup:

Open NetSim and click **on Examples > Experiments > Hidden-Node-Problem-in-WLAN > Sample-1** as shown below:



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



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 L2 Switch, and 1 Access Point in the **"Internetworks"** Network Library.

Step 2: The device positions are set as follows:

	Access Point 2	Wireless Node 4	Wireless Node 5
X/Lat	250	150	350
Y/Lon	100	150	150

Step 3: In the Interface Wireless > Data Link Layer Properties of Access Point, **"RTS Threshold"** is set to 3000.

In NetSim RTS CTS mechanism can be enabled or disabled in WLAN using RTS THRESHOLD parameter. RTS CTS mechanism is enabled if the RTS THRESHOLD is less than the packet size. RTS CTS mechanism is disabled if the RTS THRESHOLD is greater than the packet size.

Step 4: TCP Protocol is set to Disable in all the devices.

Step 5: Wireless Link Properties is set as follows:

Wireless Link Properties	
Channel Characteristics	Path Loss Only
Path Loss Model	LOG_DISTANCE
Path Loss Exponent(n)	2

Step 6: Right click on the Application Flow **App1 CBR** or **App2 CBR** and select Properties or click on the Application icon present in the top ribbon/toolbar.

A CBR Application is generated from Wireless Node 4 i.e. Source to Wired Node 1 i.e. Destination with Packet Size remaining 1460 Bytes and Inter Arrival Time remaining 20000µs. Another CBR Application is generated from Wireless Node 5 i.e. Source to Wired Node 1 i.e. Destination with Packet Size remaining 1460 Bytes and Inter Arrival Time remaining 20000µs.

***NOTE:* Packet size here refers to the size of the packet along with the overheads added in the layers above and not the application layer packet size.**

Step 7: Run the Simulation for 10 Seconds.

Sample 2:

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

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

Step 2: Run the Simulation for 10 Seconds.

15.4 Output:

From the **"Link Metrics"** in the Results Dashboard, we can see the Data Packet and Control Packet Collisions. A comparison table with and without RTS/CTS mechanism is given below:

Collided Packets	Without RTS/CTS	With RTS/CTS
Data Packets	22	0
Control Packets	0	20

15.5 Inference:

During simulations performed without RTS CTS mechanism enabled we notice data packet collisions. This is because nodes Wireless Node 4 and Wireless Node 5 transmit in parallel as they are out of range of each other. RTS CTS mechanism helps in avoiding data packet collisions with the help of RTS and CTS control packets that are exchanged before attempting transmissions. However, there can be control packet collisions which may involve, WLAN Ack's and RTS, CTS packets.

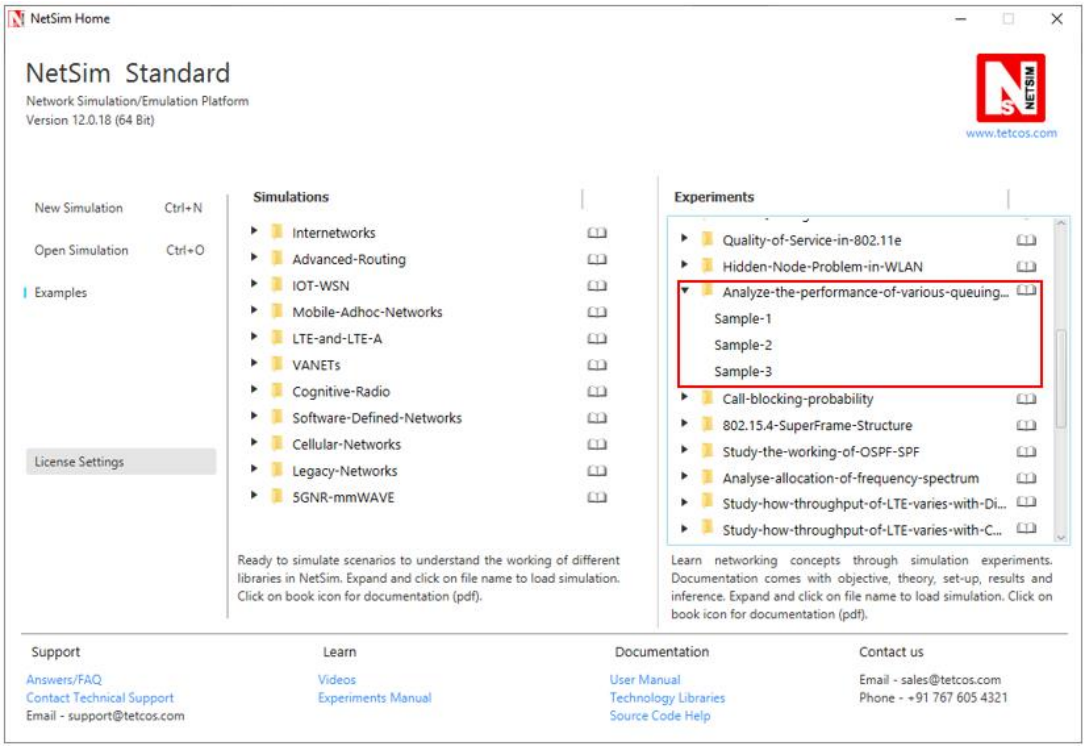
16. Analyze the performance of FIFO, Priority and WFQ Queuing Disciplines

16.1 Introduction

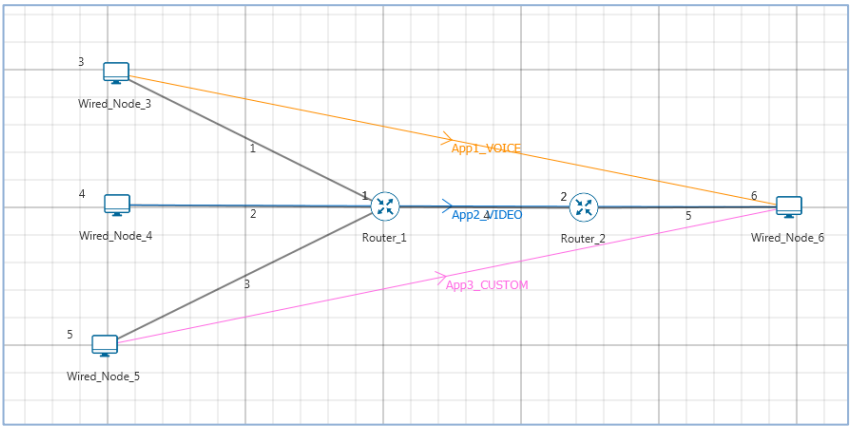
As part of the resource allocation mechanisms, each router must implement some queuing discipline that governs how packets are buffered while waiting to be transmitted. Various queuing disciplines can be used to control which packets get transmitted (based on bandwidth allocation) and which packets get 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 > Analyze-the-performance-of various-queuing-disciplines > Sample-1** as shown below:



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: TCP Protocol is set to Disable in all the devices.

Step 3: Wired Link Properties is set as follows:

Link Properties	Link 1	Link 2	Link 3	Link 4	Link 5
Max Uplink Speed (Mbps)	10	10	10	5	5
Max Downlink Speed (Mbps)	10	10	10	5	5

Step 4: 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 5: Three different applications are generated as per the table given below:

NOTE: ***For Voice application set codec as Custom.***

Application Properties	Application 1	Application 2	Application 3
Application Type	**Voice (Codec-Custom)**	**Video**	**Custom**
Source_Id	**3**	**4**	**5**
Destination_Id	**6**	**6**	**6**
QoS	**RTPS**	**NRTPS**	**BE**
Packet Size			
Distribution	**Constant**	**Frame_Per_Sec**	**Constant**

Value (bytes)	**1460**	**50**	**1000**
Inter Arrival Time			
Distribution	**Constant**	**Pixel_Per_Frame**	**Constant**
Value (micro secs)	**2336**	**100000**	**1333**

Step 6: 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:

Application	Traffic Generation Rate (Mbps)*	FIFO-Sample-1 Throughput (Mbps)	Priority-Sample-2 Throughput (Mbps)	WFQ-Sample-3 Throughput (Mbps)
Voice	5	1.75 ~ (5 / 13.6) *5	3.81	1.89
Video	2.6	0.89 ~ (2.6/13.6) *5	0.28	0.85
Custom	6	2.12 ~ (6/13.6) *5	0.70	2.02
Total	13.6	4.76 ~ 5	4.79 ~ 5	4.76 ~ 5

NOTE: ***For Traffic Generation Rate calculation please refer user manual section 5.3***

*The traffic generation rate is based on settings done in step 5.

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.

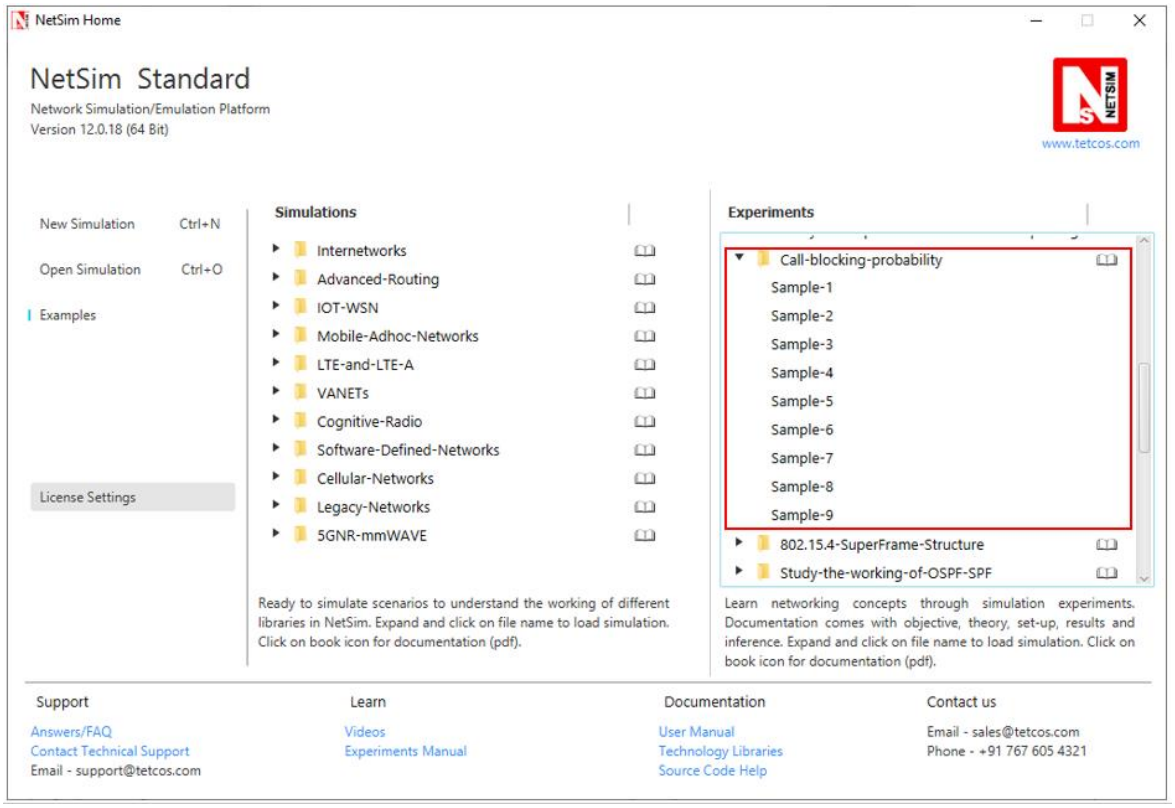
Priority scheduling technique processes packets based on their priority. Hence voice and video which have higher priority take up the complete bandwidth available.

Weighted fair queuing (WFQ) assigns a weight to each application and hence gives a result between that is in between priority and FIFO.

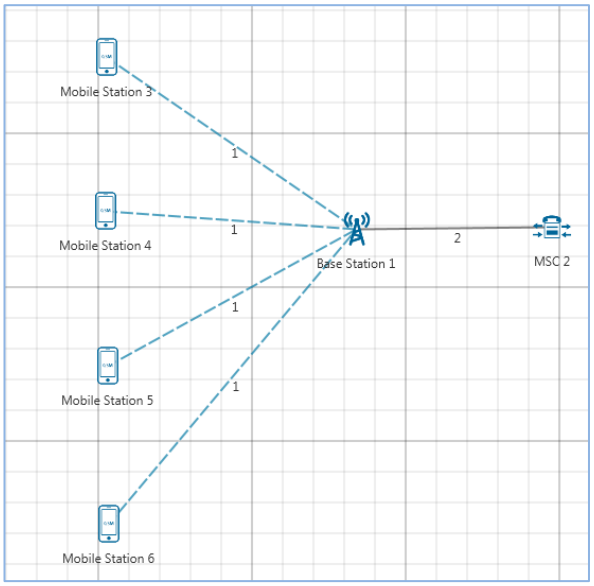
17. Study how call blocking probability varies as the load on a GSM network is continuously increased

17.1 Network Setup:

Open NetSim and click **Examples > Experiments > Call-blocking-probability** as shown below:



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



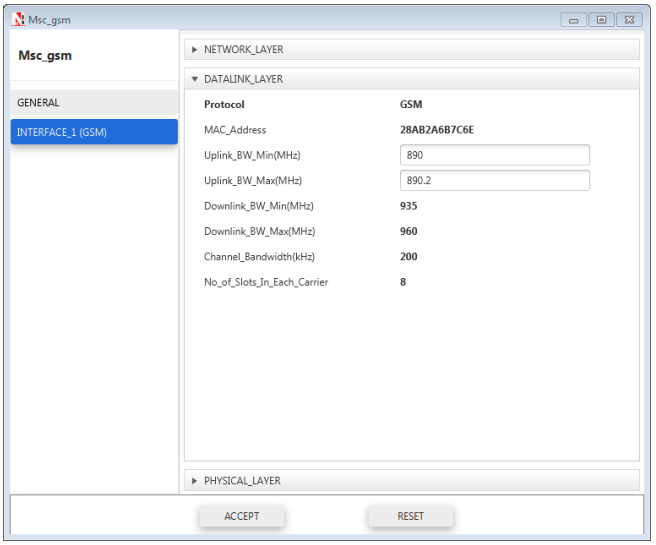
17.2 Procedure:

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:

Application Properties	Application 1	Application2
Application type	Erlang_call	Erlang_call
Source_Id	3	5
Destination_Id	4	6
Call		
Duration_ Distribution	Exponential	Exponential
Duration(s)	60	60
Inter Arrival Time (sec)	10	10
IAT_ Distribution	Exponential	Exponential
Codec	Custom	Custom
Inter Arrival Time distribution	Constant	Constant
Packet Distribution	Constant	Constant
Service Type	CBR	CBR
Packet Size	33	33
Inter Arrival Time (μs)	20000	20000

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:

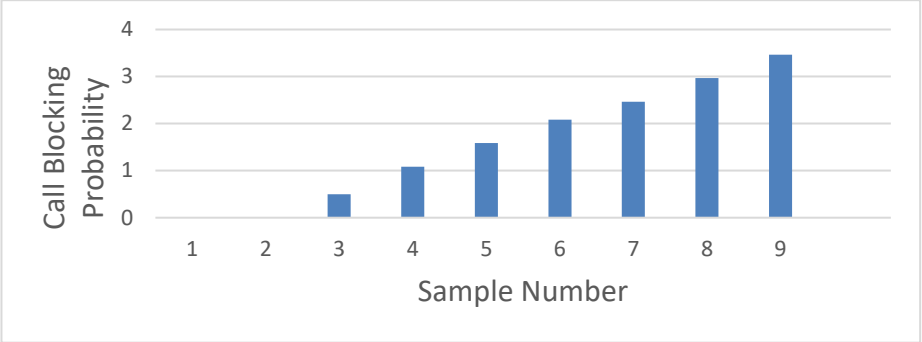
Step 1: Similarly, increase the number of Mobile Stations by 2 up to 20 and set properties for different Samples by adding an application every time and changing Source ID and Destination ID.

Step 2: Run the Simulation for 100 Seconds.

17.3 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:



*** 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.

17.4 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.

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

18.1 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 = aBaseSuperframeDuration * 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.

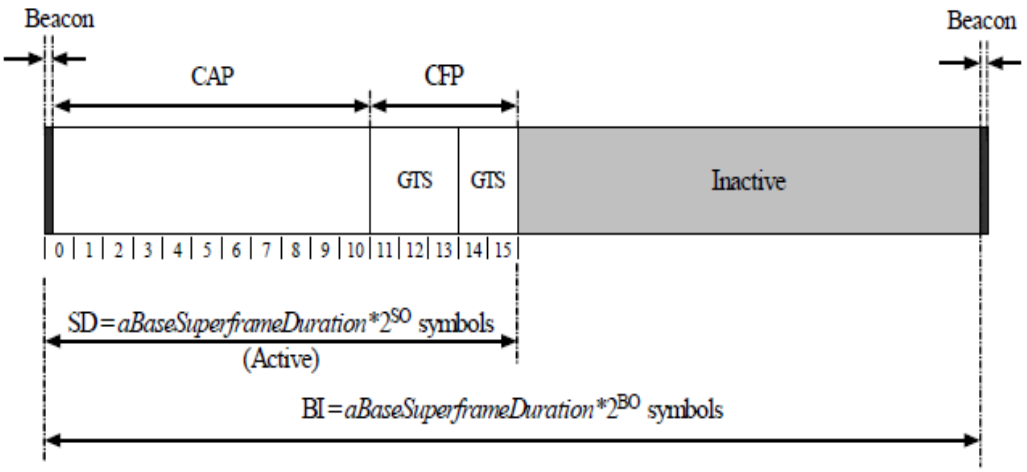


Fig: An example of the Super Frame structure

Theoretical Analysis:

From the above Superframe structure,

$$SuperFrame\ Duration = aBaseSuperframeDuration * 2^{BO}$$

$$Active\ part\ of\ SuperFrame = aBaseSuperframeDuration * 2^{SO}$$

$$Inactive\ part\ of\ SuperFrame = aBaseSuperframeDuration * (2^{BO} - 2^{SO})$$

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)th of the Superframe is inactive and so nodes have only (1/4)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.

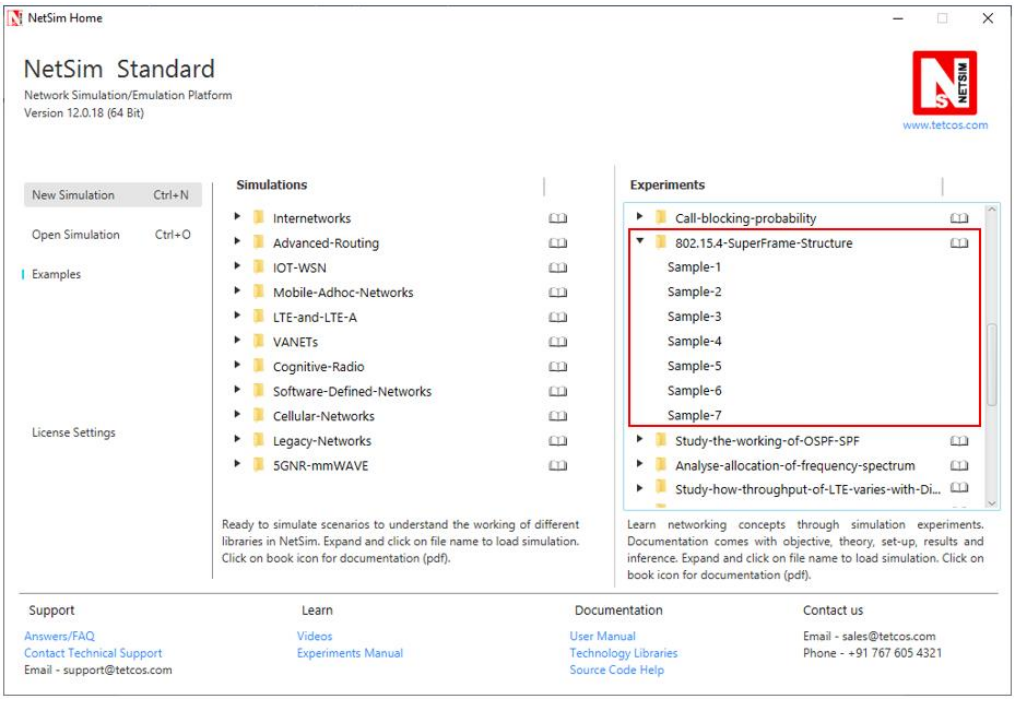
Percentage of inactive and active periods in Superframe for different Superframe Orders is given below:

Beacon Order (BO)	Super Frame Order (SO)	Active part of Superframe(%)	Inactive part of Superframe (%)	Throughput estimated (%)
10	10	100	0	**> 200% of T**
10	9	50	50	**Say T = 21.07 (Got from simulation)**
10	8	25	75	**50 % T**
10	7	12.5	87.5	**25 % T**
10	6	6.25	93.75	**12.5 % of T**
10	5	3.125	96.875	**6.25 % of T**
10	4	1.5625	98.4375	**3.12% of T**
10	**3**	**0.78125**	**99.21875**	**1.56 % of T**

We expect throughput to vary in the active part of the Superframe as sensors can transmit a packet only in the active portion.

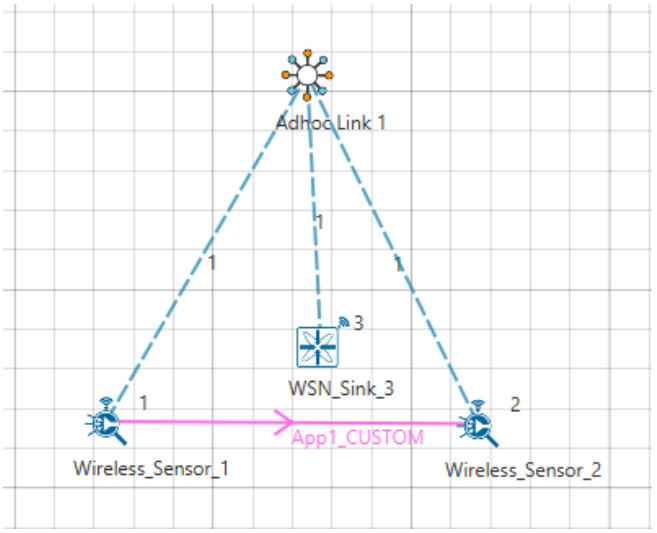
18.2 Network Setup:

Open NetSim and click **Examples > Experiments > 802.15.4-Superframe-Structure** as shown below:



Sample 1:

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



18.3 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:

$$Generation\ Rate\ (Mbps)\ =\ Packet\ Size\ (Bytes)\ *\ 8/Interarrival\ time\ (\mu s)$$

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.

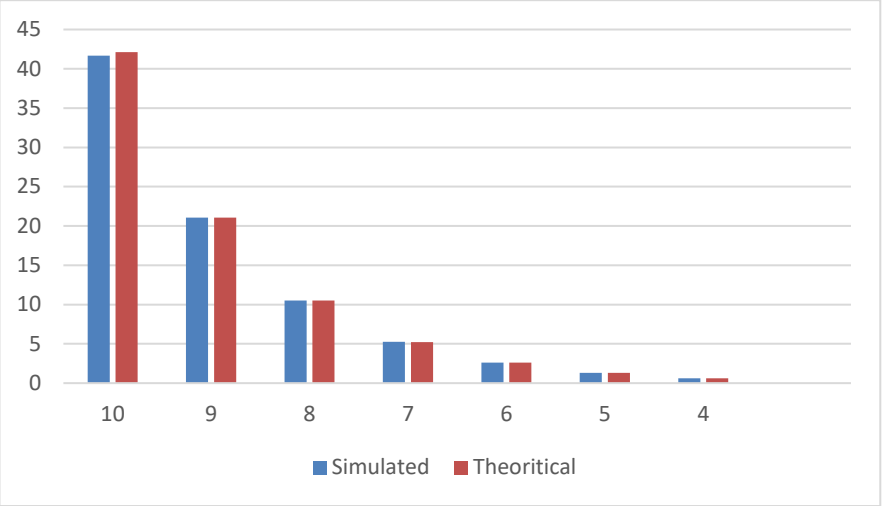
18.4 Output:

The following are the throughputs obtained from the simulation for different Super Frame Orders.

Super Frame Order	Throughput (Kbps)
10	41.63
9	21.07
8	10.5
7	5.25
6	2.63
5	1.30
4	0.62

To obtain throughput from simulation, payload transmitted values will be obtained from Link metrics and calculated using following formula:

$$Application\ Throughput(in\ Mbps) = \frac{Total\ payload\ delivered\ to\ destination\ (bytes)\ *\ 8}{Simulation\ Time(Millisecond)\ -\ App\ Start\ Time(Millisecond)}$$



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.

18.5 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.

19. Understand the working of OSPF

19.1 Objective

To understand the working of OSPF and Shortest Path First (SPF) tree creation.

19.2 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 shorted 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)

The below network shows a sample map of an Autonomous System

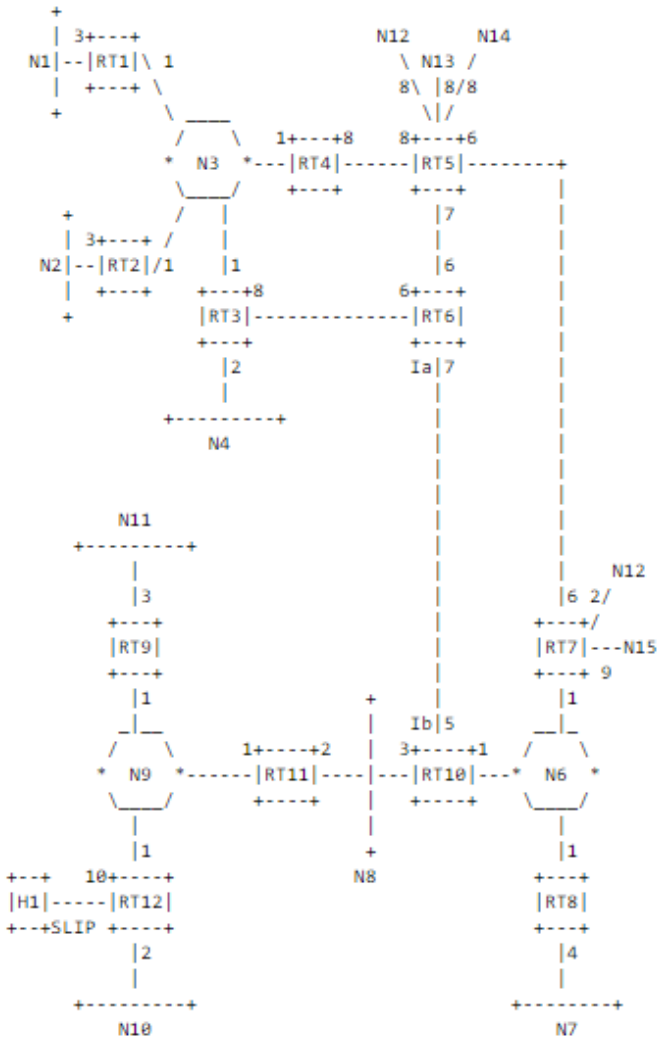


Fig 1. Sample Autonomous system

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).

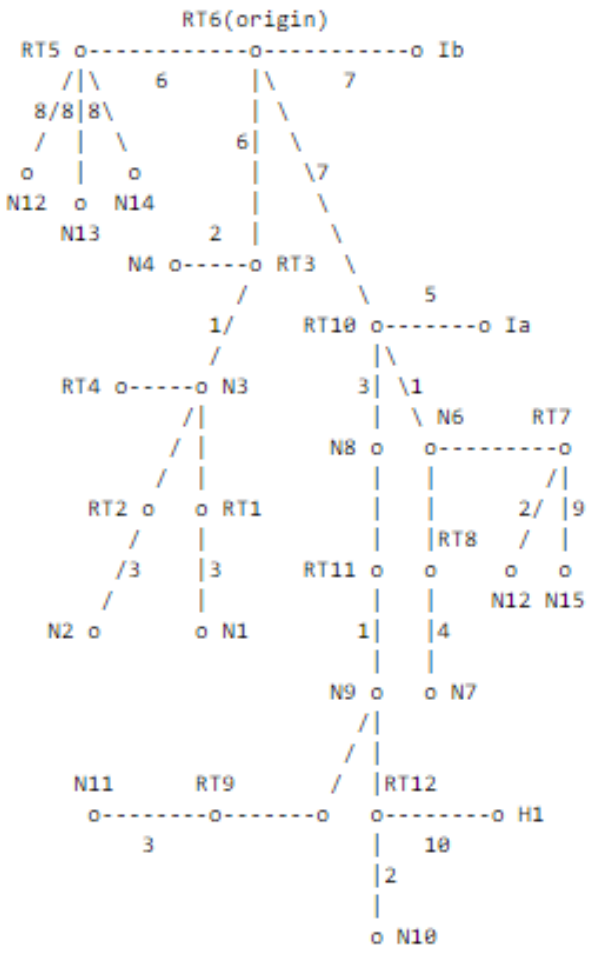
The directed graph resulting from the above network is depicted in the following table. Arcs are labelled with the cost of the corresponding router output interface. Arcs having no labelled cost have a cost of 0. Note that arcs leading from networks to routers always have cost 0.

	FROM																
		RT1	RT2	RT3	RT4	RT5	RT6	RT7	RT8	RT9	RT 10	RT 11	RT 12	N3	N6	N8	N9

TO	RT1													0			
	RT2													0			
	RT3						6							0			
	RT4					8								0			
	RT5				8		6	6									
	RT6			8		7											
	RT7					6									0		
	RT8														0		
	RT9																0
	RT10						7								0	0	
	RT11															0	0
	RT12																0
	N1	3															
	N2		3														
	N3	1	1	1	1												
	N4			2													
	N5																
	N6							1	1		1						
	N7								4								
	N8										3	2					
	N9									1		1	1				
	N10												2				
	N11									3							
	N12					8		2									
	N13					8											
	N14					8											
	N15							9									
	H1												10				

Table 1 Directed graph

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.



SPF tree for Router 6

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

Destination	Next hop	Distance
N1	RT3	10
N2	RT3	10
N3	RT3	7
N4	RT3	8
N6	RT10	8
N7	RT10	12
N8	RT10	10
N9	RT10	11
N10	RT10	13
N11	RT10	14
H1	RT10	21
RT5	RT5	6
RT7	RT10	8

N12	RT10	10
N13	RT5	14
N14	RT5	14
N15	RT10	17

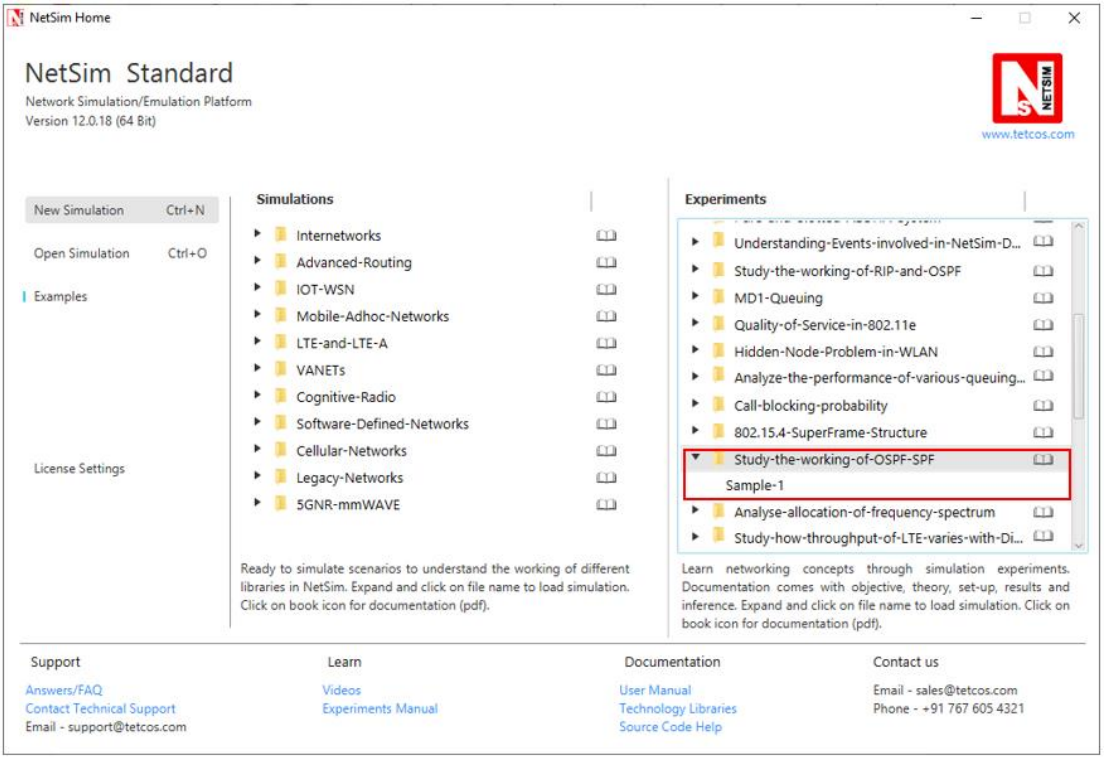
Routing Table for RT6

Distance calculation:

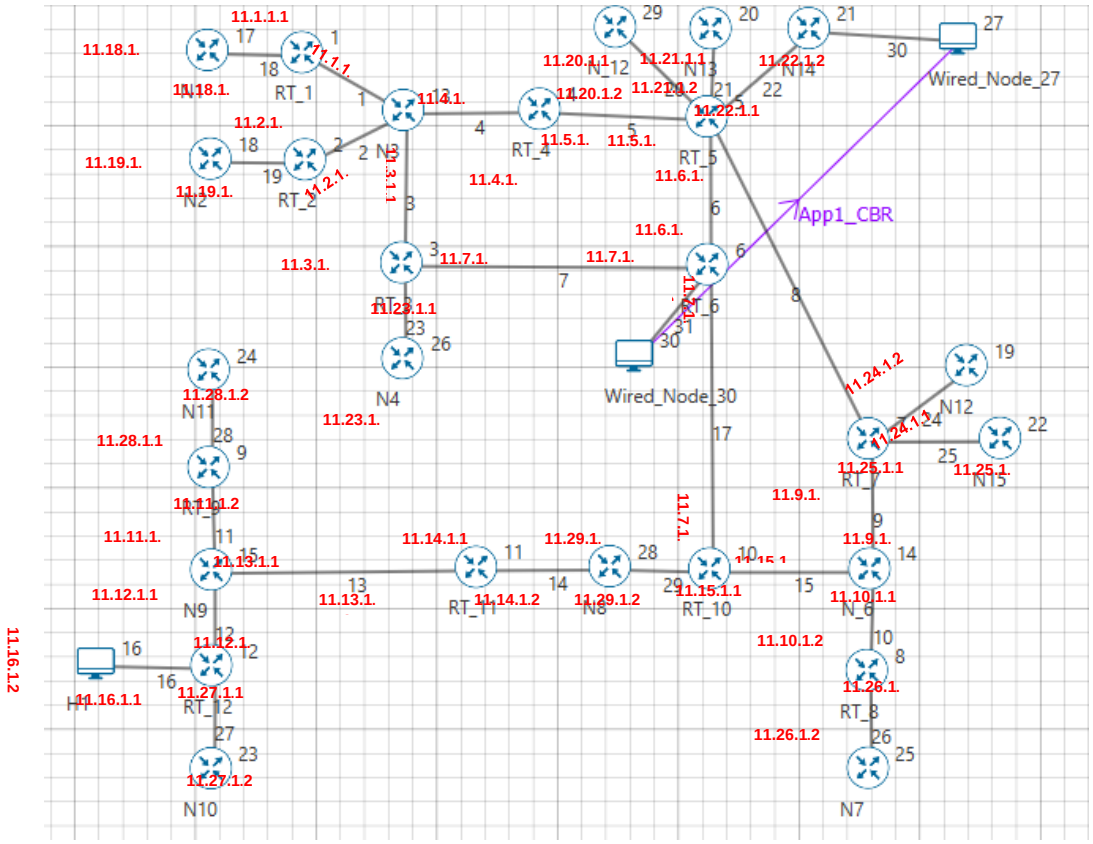
Router6 has 3 interfaces i.e. RT3, RT5 and RT10. The distance obtained is 10 for destination N1 via RT3 interface. The packets from Router6 would reach N1 via RT3, N3 and RT1. The cost assigned to routers in this path is 6+1+3 = 10 (cost can be seen in SPF tree for Router6). This is how distance is calculated.

19.3 Network Setup:

Open NetSim and click on **Examples > Experiments > Study-the-working-of-OSPF-SPF > Sample-1** as shown below:



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



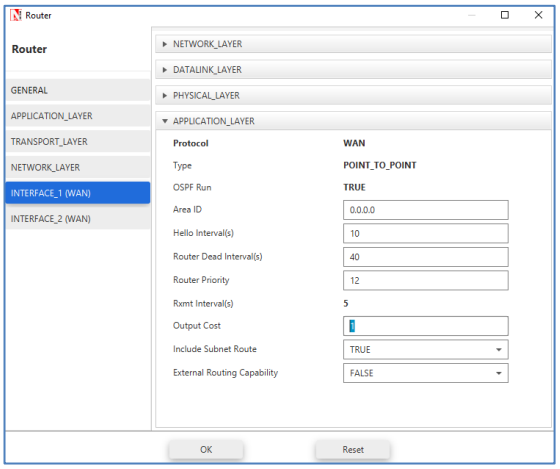
The above network was created in NetSim and it is similar to the network as per the OSPF RFC 2328 (Refer Pg. no. 19 - https://tools.ietf.org/html/rfc2328#page-23)

19.4 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.



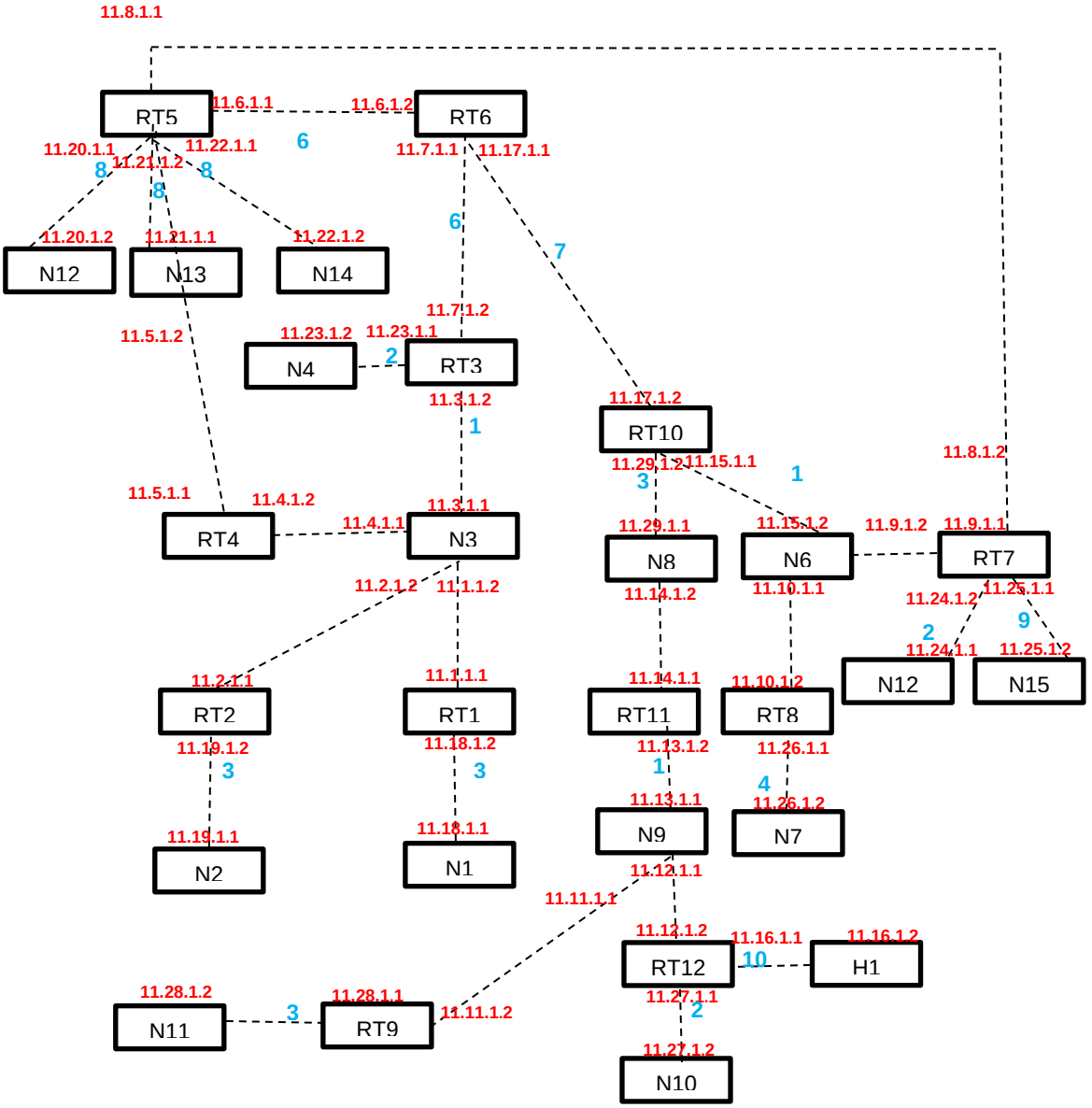
Step 3: Packet Trace is enabled in the NetSim GUI, and hence we are able to track the route which the packets have chosen to reach the destination based on the Output Cost that is set.

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.

19.5 Output:

The following is the shortest path first tree created in NetSim:



SPF tree for Router 6

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:

RT_6_Table

RT_6 Detailed View

Network Destination	Netmask/Prefix len	Gateway	Interface	Metrics	Type
11.3.1.2	255.255.0.0	11.7.1.2	11.7.1.1	7	OSPF
11.1.1.2	255.255.0.0	11.7.1.2	11.7.1.1	7	OSPF
11.2.1.2	255.255.0.0	11.7.1.2	11.7.1.1	7	OSPF
11.3.1.1	255.255.0.0	11.7.1.2	11.7.1.1	7	OSPF
11.4.1.1	255.255.0.0	11.7.1.2	11.7.1.1	7	OSPF
11.15.1.1	255.255.0.0	11.17.1.2	11.17.1.1	8	OSPF
11.23.1.1	255.255.0.0	11.7.1.2	11.7.1.1	8	OSPF
11.4.1.2	255.255.0.0	11.7.1.2	11.7.1.1	8	OSPF
11.9.1.2	255.255.0.0	11.17.1.2	11.17.1.1	8	OSPF
11.10.1.1	255.255.0.0	11.17.1.2	11.17.1.1	8	OSPF
11.15.1.2	255.255.0.0	11.17.1.2	11.17.1.1	8	OSPF
11.1.1.1	255.255.0.0	11.7.1.2	11.7.1.1	8	OSPF
11.2.1.1	255.255.0.0	11.7.1.2	11.7.1.1	8	OSPF
11.23.1.2	255.255.0.0	11.7.1.2	11.7.1.1	8	OSPF
11.9.1.1	255.255.0.0	11.17.1.2	11.17.1.1	9	OSPF

RT_6_Table

RT_6 Detailed View

Network Destination	Netmask/Prefix len	Gateway	Interface	Metrics	Type
11.10.1.2	255.255.0.0	11.17.1.2	11.17.1.1	9	OSPF
11.29.1.2	255.255.0.0	11.17.1.2	11.17.1.1	10	OSPF
11.24.1.2	255.255.0.0	11.17.1.2	11.17.1.1	10	OSPF
11.18.1.2	255.255.0.0	11.7.1.2	11.7.1.1	10	OSPF
11.19.1.2	255.255.0.0	11.7.1.2	11.7.1.1	10	OSPF
11.14.1.2	255.255.0.0	11.17.1.2	11.17.1.1	10	OSPF
11.29.1.1	255.255.0.0	11.17.1.2	11.17.1.1	10	OSPF
11.24.1.1	255.255.0.0	11.17.1.2	11.17.1.1	10	OSPF
11.18.1.1	255.255.0.0	11.7.1.2	11.7.1.1	10	OSPF
11.19.1.1	255.255.0.0	11.7.1.2	11.7.1.1	10	OSPF
11.13.1.2	255.255.0.0	11.17.1.2	11.17.1.1	11	OSPF
11.11.1.1	255.255.0.0	11.17.1.2	11.17.1.1	11	OSPF
11.12.1.1	255.255.0.0	11.17.1.2	11.17.1.1	11	OSPF
11.13.1.1	255.255.0.0	11.17.1.2	11.17.1.1	11	OSPF
11.17.1.2	255.255.0.0	11.17.1.2	11.17.1.1	12	OSPF

RT_6_Table

RT_6 Detailed View

Network Destination	Netmask/Prefix len	Gateway	Interface	Metrics	Type
11.8.1.1	255.255.0.0	11.6.1.1	11.6.1.2	12	OSPF
11.26.1.1	255.255.0.0	11.17.1.2	11.17.1.1	12	OSPF
11.14.1.1	255.255.0.0	11.17.1.2	11.17.1.1	12	OSPF
11.26.1.2	255.255.0.0	11.17.1.2	11.17.1.1	12	OSPF
11.11.1.2	255.255.0.0	11.17.1.2	11.17.1.1	12	OSPF
11.12.1.2	255.255.0.0	11.17.1.2	11.17.1.1	12	OSPF
11.6.1.1	255.255.0.0	11.6.1.1	11.6.1.2	13	OSPF
11.27.1.1	255.255.0.0	11.17.1.2	11.17.1.1	13	OSPF
11.27.1.2	255.255.0.0	11.17.1.2	11.17.1.1	13	OSPF
11.5.1.2	255.255.0.0	11.6.1.1	11.6.1.2	14	OSPF
11.20.1.2	255.255.0.0	11.6.1.1	11.6.1.2	14	OSPF
11.21.1.2	255.255.0.0	11.6.1.1	11.6.1.2	14	OSPF
11.22.1.1	255.255.0.0	11.6.1.1	11.6.1.2	14	OSPF
11.7.1.2	255.255.0.0	11.7.1.2	11.7.1.1	14	OSPF
11.8.1.2	255.255.0.0	11.17.1.2	11.17.1.1	14	OSPF

RT_6_Table

RT_6 Detailed View

Network Destination	Netmask/Prefix len	Gateway	Interface
11.20.1.1	255.255.0.0	11.6.1.1	11.6.1.2
11.21.1.1	255.255.0.0	11.6.1.1	11.6.1.2
11.22.1.2	255.255.0.0	11.6.1.1	11.6.1.2
11.28.1.1	255.255.0.0	11.17.1.2	11.17.1.1
11.28.1.2	255.255.0.0	11.17.1.2	11.17.1.1
11.5.1.1	255.255.0.0	11.7.1.2	11.7.1.1
11.25.1.1	255.255.0.0	11.17.1.2	11.17.1.1
11.25.1.2	255.255.0.0	11.17.1.2	11.17.1.1
11.31.0.0	255.255.0.0	on-link	11.31.1.1
11.17.0.0	255.255.0.0	on-link	11.17.1.1
11.7.0.0	255.255.0.0	on-link	11.7.1.1
11.6.0.0	255.255.0.0	on-link	11.6.1.2
224.0.0.1	255.255.255.255	on-link	11.6.1.2 1...
224.0.0.0	240.0.0.0	on-link	11.6.1.2 1...
255.255.255.255	255.255.255.255	on-link	11.31.1.1

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

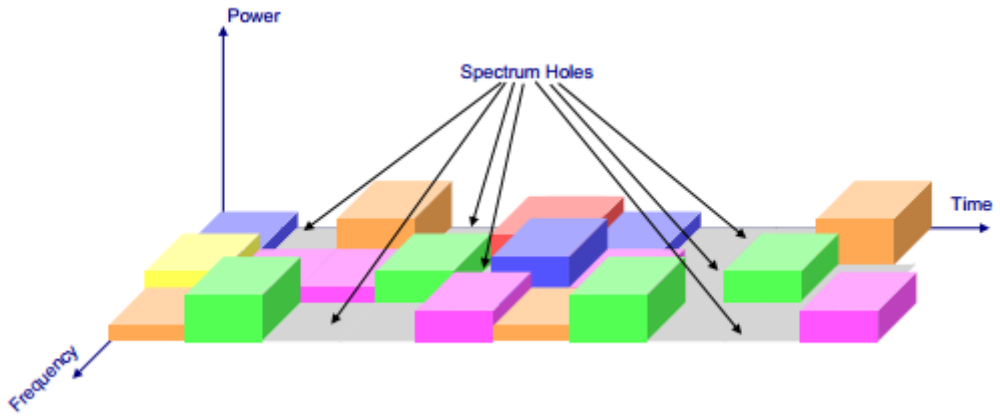
20. To analyze how the allocation of frequency spectrum to the Incumbent (Primary) and CR CPE (Secondary User) affects throughput

20.1 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 SUs (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.

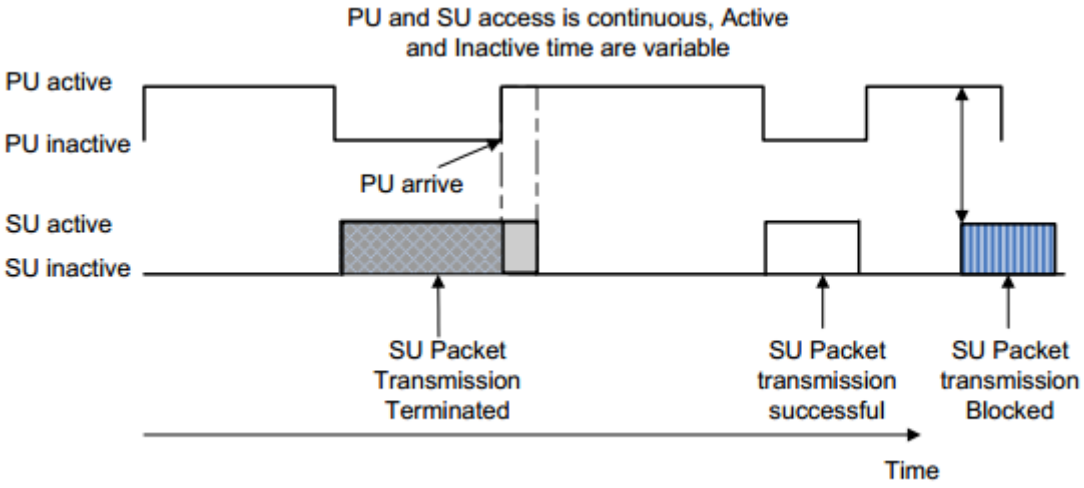


Illustration of forced termination and blocking

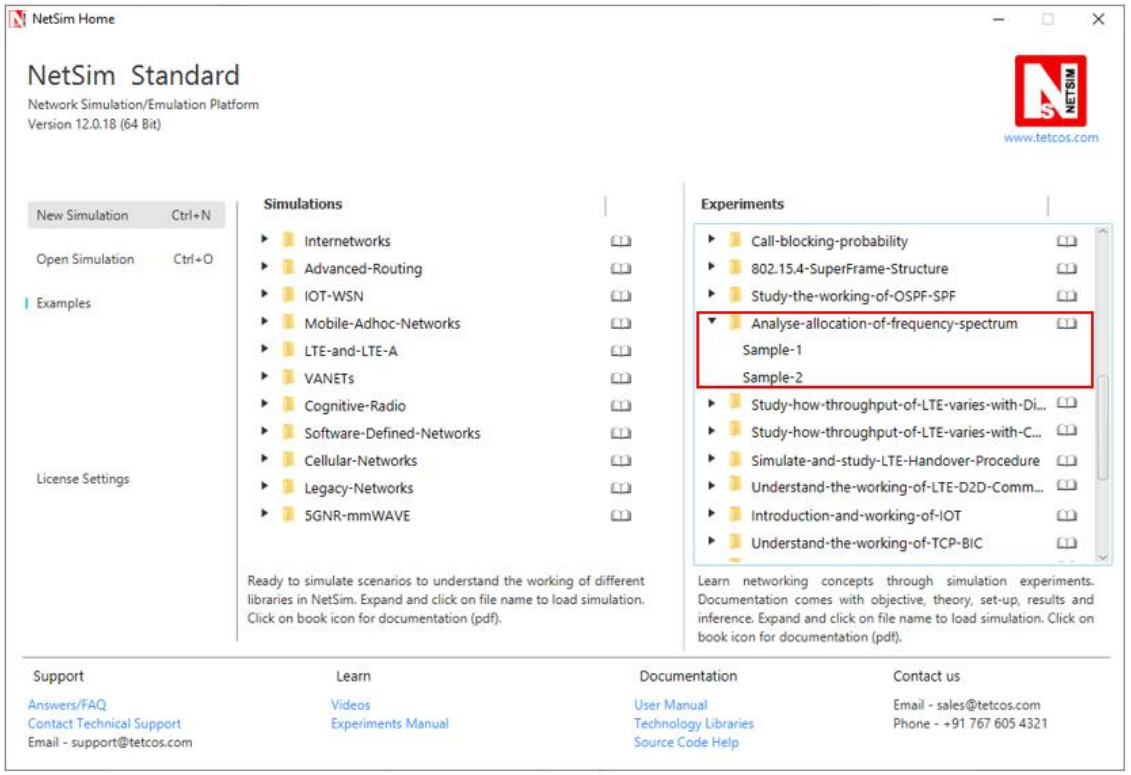
Performance metrics:

The different parameters used to analyze the performance are explained as follows:

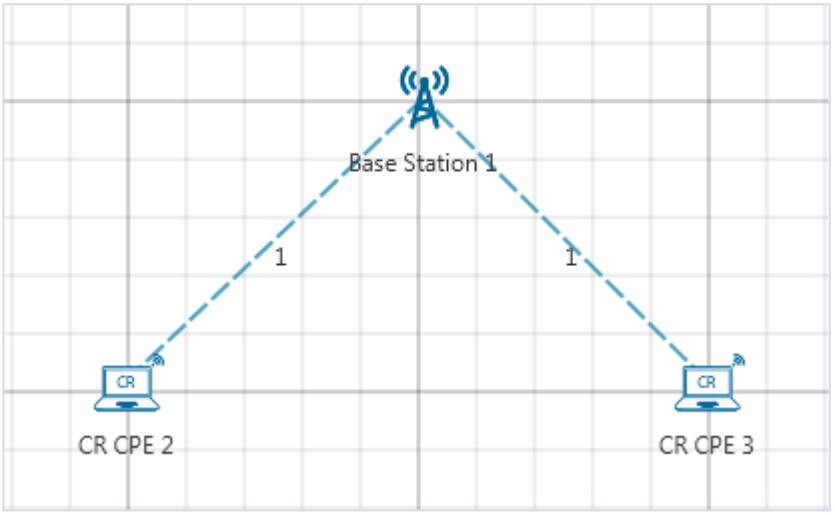
- **Throughput**: It is the rate of successfully transmitted data packets in unit time in the network during the simulation.
- **Spectral Efficiency:** It refers to the information rate that can be transmitted over a given bandwidth in a specific communication system. It is a measure of how efficiently a limited frequency spectrum is utilized by the physical layer protocol, and sometimes by the media access control protocol.

20.2 Network Setup:

Open NetSim and click on **Examples > Experiments > Analyze-allocation-of-frequency-spectrum > Sample-1** as shown below:



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



20.3 Procedure:

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

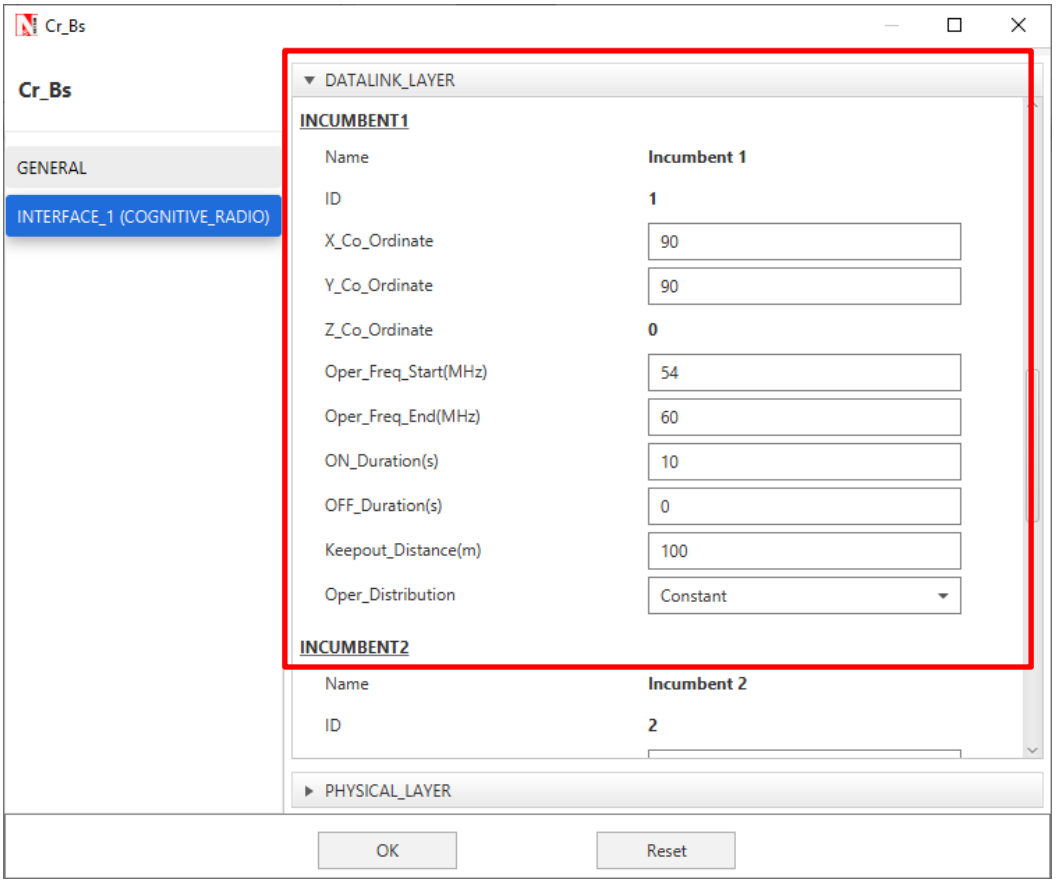
Step 1: A network scenario is designed in NetSim GUI comprising of 1 Base Station and 2 CR CPE's in the **"Cognitive Radio"** Network Library.

Step 2: The device positions are set as follows:

Base Station 1	CR CPE 2	CR CPE 3

X/Lat	100	100	120
Y/Lon	100	120	100

Step 3: In the **Interface Cognitive Radio > Datalink Layer > Incumbment1,** 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.

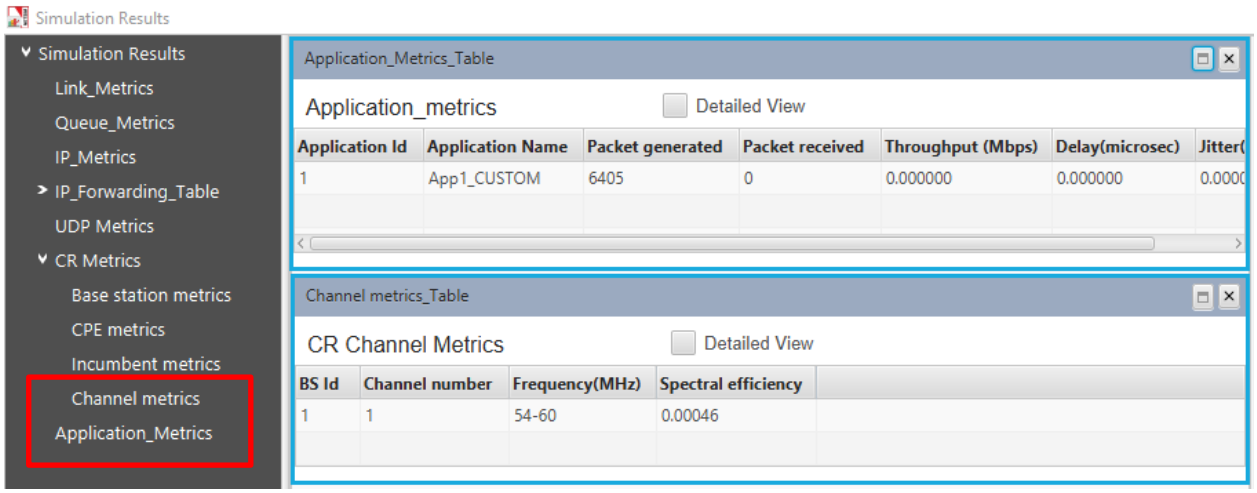
Step 2: Run the Simulation for 100 Seconds.

20.4 Output:

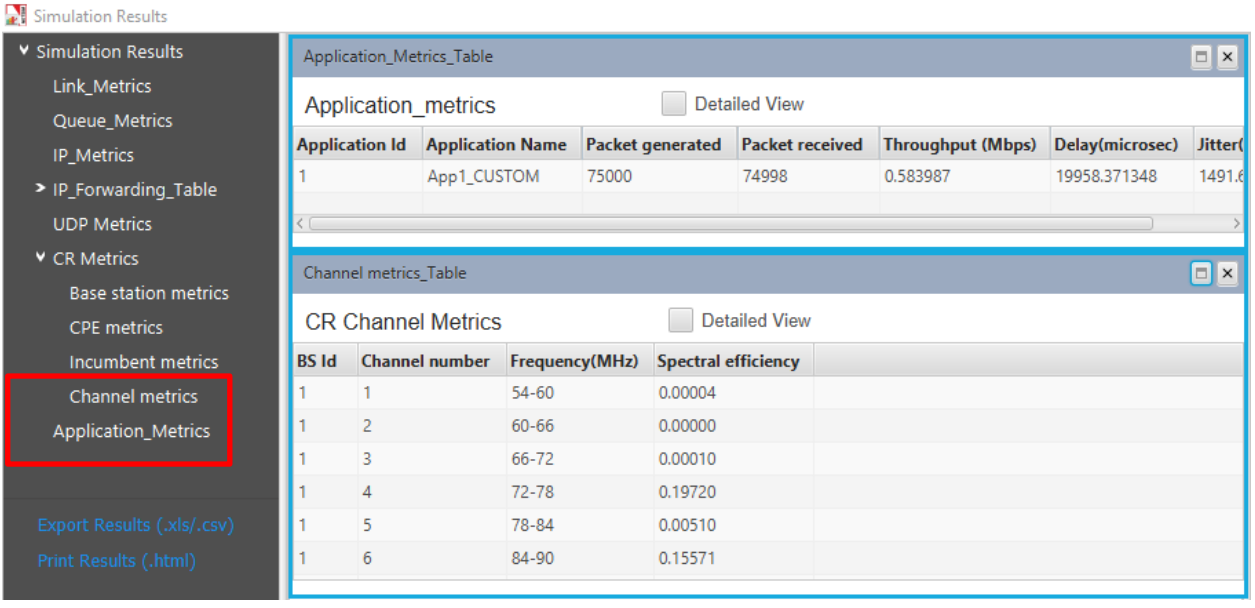
Once after the simulation is complete, go to the Results Dashboard and check the **"Application Metrics"** Table. Throughput of the application will be 0.

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 2:



20.5 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.***

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

21.1 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., Pr = Pt

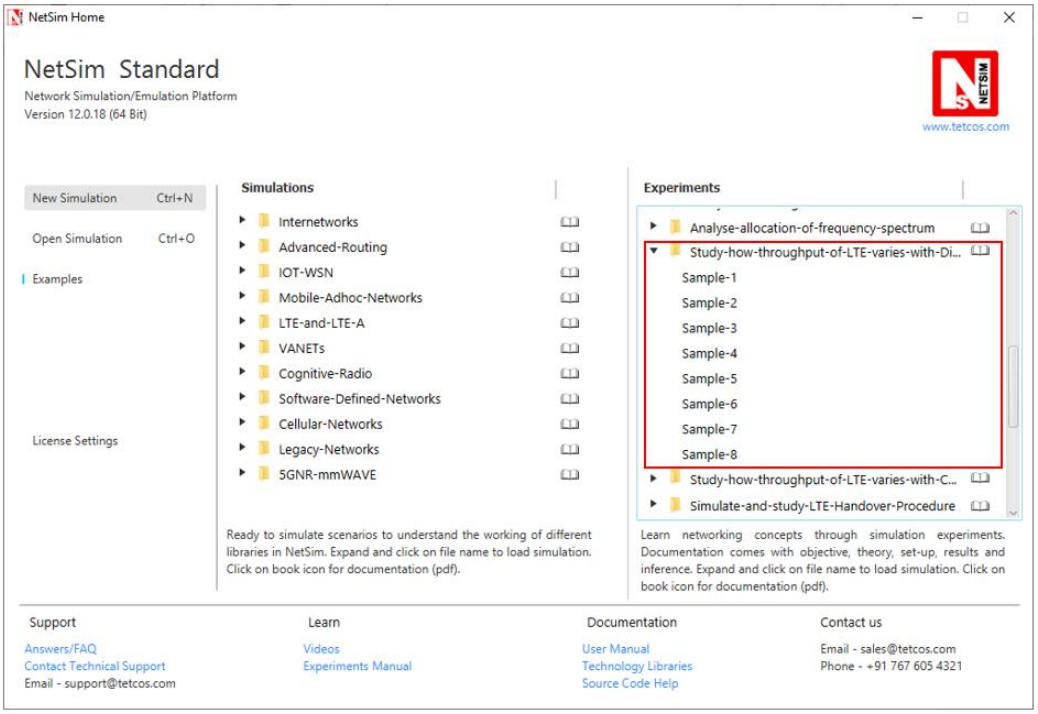
Case 2: When **Line of Sight** is there, Received power P_r is

$$P_r = P_t + G_t + G_r + 20 \log_{10}\left(\frac{\lambda}{4\pi d}\right) + 10\, n \log_{10}\frac{d0}{d}$$

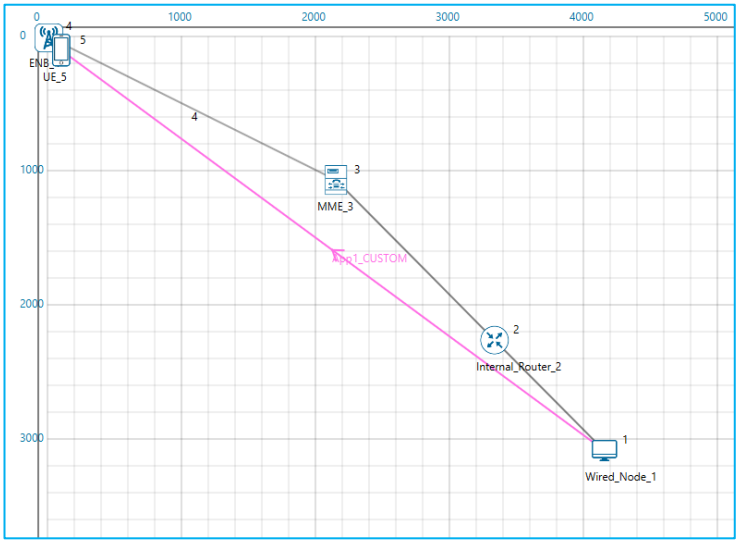
Where **G_t** and **G_r** are gains of transmitting and receiving antenna respectively. Here **d** is the distance between transmitter and receiver, **λ** is the wavelength of the transmitted signal and $d0$ is reference distance at which channel gain becomes 1. **n** is path loss exponent and **P_t** is transmitted power.

21.2 Network Setup:

Open NetSim and click **Examples > Experiments > Study-how-throughput-of-an-LTE-varies-with-Distance > Sample-1** as shown below:



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



21.3 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 Environment 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: TCP Protocol is disabled in Wired Node 1.

Step 3: The device positions are set as per the below table:

	ENB 4	UE 5
X/Lat	0	50
Y/Lon	0	50

Step 4: 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 5: In the General Properties of UE 5 **"Velocity (m/s)"** parameter is set to 0.

Step 6: The Wired Link Properties are set as follows:

Link Properties	Wired Link 2	Wired Link 3	Wired Link 4
Uplink Speed (Mbps)	100	100	100
Downlink Speed (Mbps)	100	100	100
Uplink BER	0	0	0
Downlink BER	0	0	0
Up Time	N/A	0	0
Down Time	N/A	0	0
Uplink Propagation Delay (microsec)	0	0	0
Downlink Propagation Delay (microsec)	0	0	0

Step 7: The Wireless Link Properties are set as follows:

Link Properties	Wireless Link 1
Channel characteristics	**Path Loss Only**
Path Loss Model	**Log Distance**
Path loss Exponent(n)	**4**

Step 8: 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 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:

$$Generation\ Rate\ (Mbps)\ =\ Packet\ Size\ (Bytes)\ *\ 8/Interarrival\ time\ (\mu s)$$

Step 9: 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:

Change in UE Properties: (x, y)	
Sample 2	(100, 100)
Sample 3	(150,150)
Sample 4	(200,200)
Sample 5	(250,250)
Sample 6	(300,300)
Sample 7	(350,350)
Sample 8	(400,400)

21.4 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:

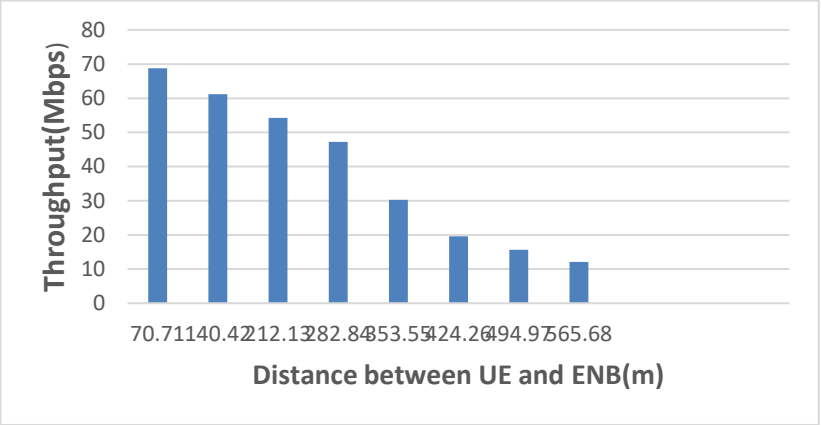
ENB (x_1, y_1) = (0, 0); UE (x_2, y_2) = (50, 50);

Distance = $\sqrt{(50-0)^2+ (50-0)^2}$ **=** $\sqrt{2} \times 50 =$ **50**$\sqrt{\mathbf{2}}$ meters.

Step 2: Open any Excel File and note down the distance between the UE and ENB and the throughput values as shown below:

Sample	Distance between UE and ENB (meters)			Throughput (Mbps)
1	50√2	=	70.71	68.8
2	100√2	=	140.42	61.1
3	150√2	=	212.13	54.3
4	200√2	=	282.84	47.3
5	250√2	=	353.55	30.2
6	300√2	=	424.26	19.6
7	350√2	=	494.97	15.6
8	400√2	=	565.68	12.1

Comparison Chart:



To draw these graphs by using Excel **"Insert →Chart"** option and then select chart type as **"Line chart"**.

21.5 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.

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

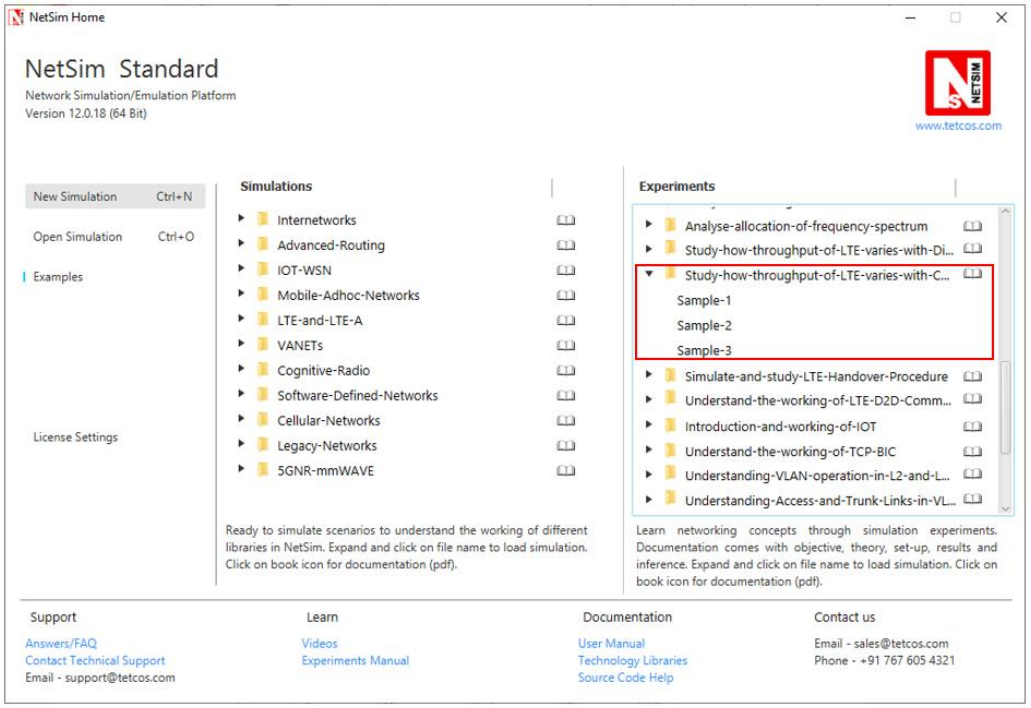
22.1 Theory:

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.

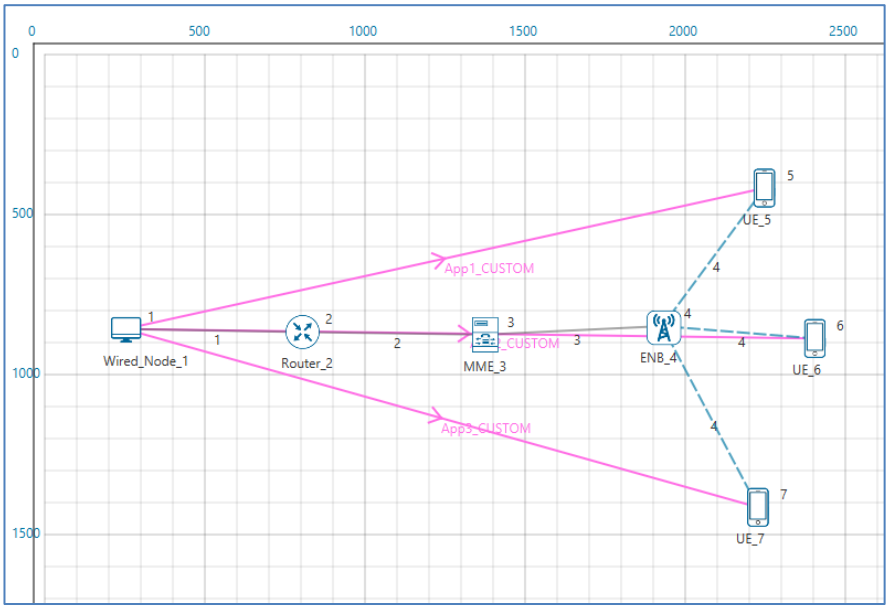
LTE supports flexible carrier bandwidths, from 1.4 MHz up to 20 MHz as well as both FDD and TDD. LTE designed with a scalable carrier bandwidth from 1.4 MHz up to 20 MHz which bandwidth is used depends on the frequency band and the amount of spectrum available with a network operator.

22.2 Network Setup:

Open NetSim and click **Examples > Experiments > Study-how-throughput-of-an-LTE-varies-with-Channel-Bandwidth > Sample-1** as shown below:



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



22.3 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: TCP Protocol is set to Disable in Wired Node 1.

Step 3: In the **Interface LTE > Physical Layer,** 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 4: In the General Properties of all the UE's **"Velocity (m/s)"** parameter is set to 0.

Step 5: The Wired Link Properties are set as follows:

Link Properties	Wired Link 1	Wired Link 2	Wired Link 3
Uplink Speed (Mbps)	1000	1000	1000
Downlink Speed (Mbps)	1000	1000	1000
Uplink BER	0	0	0
Downlink BER	0	0	0

Step 6: In the Wireless Link Properties, Channel Characteristics is set to NO PATHLOSS.

Step 7: Three CUSTOM Applications are configured as per the table given below:

Application Properties	Application 1	Application 2	Application 3
Application Type	Custom	Custom	Custom
Source ID	Wired Node 1	Wired Node 1	Wired Node 1
Destination ID	UE 5	UE 6	UE 7
Packet Size			
Distribution	Constant	Constant	Constant
Value(Bytes)	1460	1460	1460
Inter Arrival Time			
Distribution	Constant	Constant	Constant
Value(µs)	146	146	146

Step 8: 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.

22.4 Output

Add the sum of all throughput values in each sample case:

Example: Sample 1

Application Id	Throughput (Mbps)
1	23.177
2	23.177
3	23.177
Sum	**69.531**

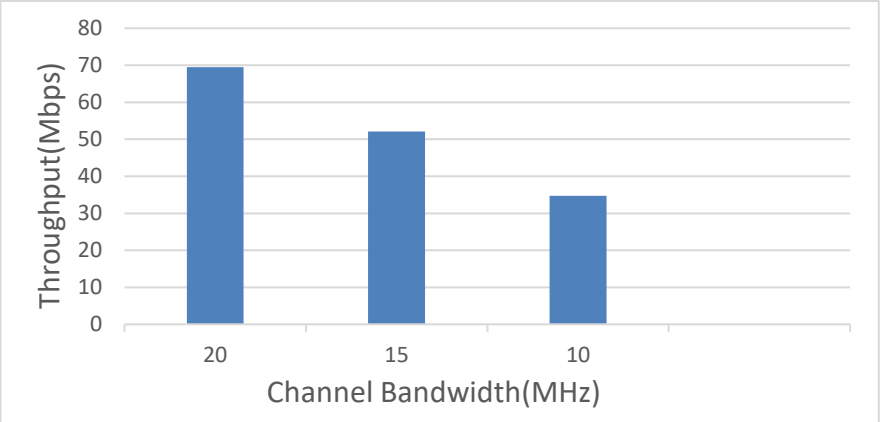
Same procedure can be followed for the other samples.

Open any Excel file and note down the sum of applications throughput values as shown in below table:

Sample	Channel Bandwidth(MHz)	Throughput (Mbps)
1	20	69.53
2	15	52.11
3	10	34.7

Comparison Chart:

To draw these graphs by using Excel **"Insert →Chart"** option and then select chart type as **"Line chart"**.



22.5 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.

Channel Bandwidth (MHz)	1.4	3	5	10	15	20
Transmission Bandwidth Configuration NRB: (1 resource block = 180kHz in 1ms TTI)	6	15	25	50	75	100

23. Simulate and study LTE Handover procedure

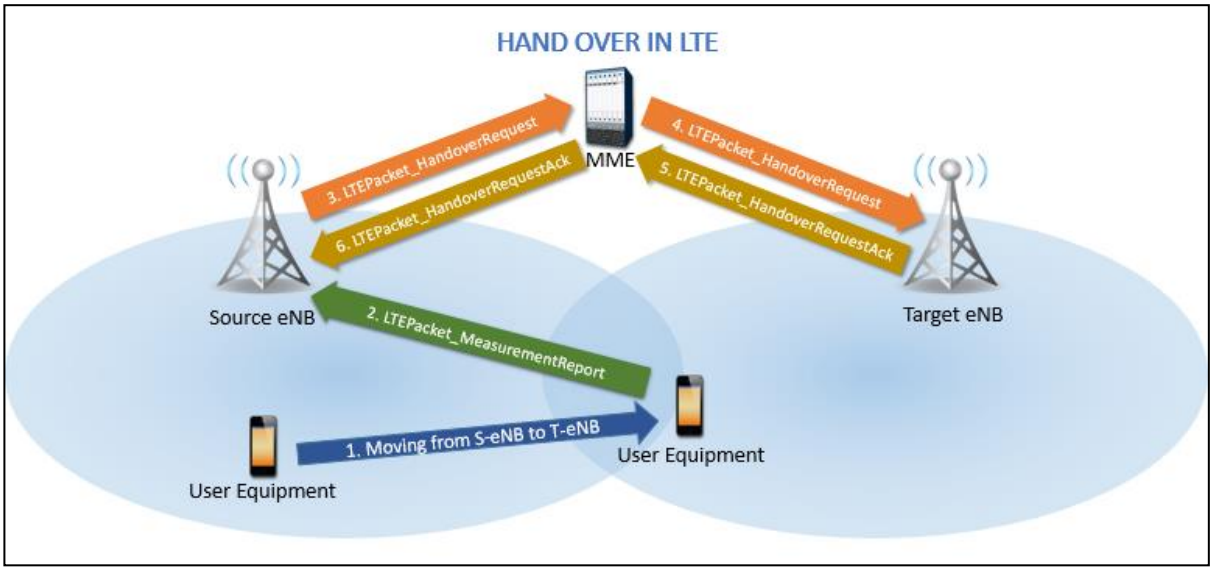
23.1 Introduction:

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.

23.2 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.

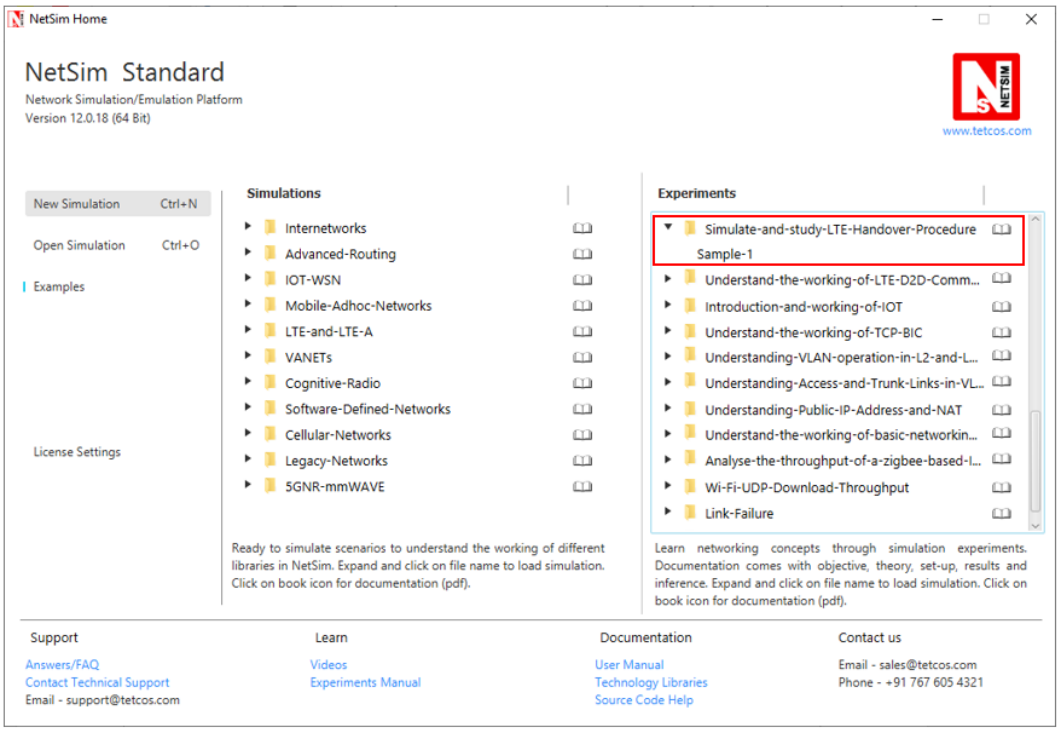


23.3 Analysis/Algorithm:

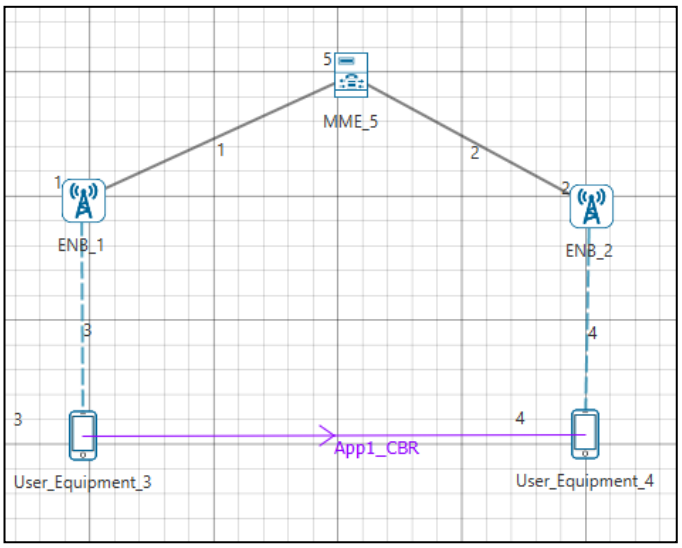
NetSim handover algorithm utilizes the Reference Signal Received Quality (RSRQ) measurements, to trigger the handover. When the target eNB's RSRQ crosses the serving eNB's RSRQ by a factor know as margin of handover (equal to 3dB), hand over is triggered.

23.4 Network Setup:

Open NetSim and click **Examples > Experiments > Simulate-and-Study-LTE-Handover-Procedure > Sample-1** as shown below:



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



23.5 Procedure:

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

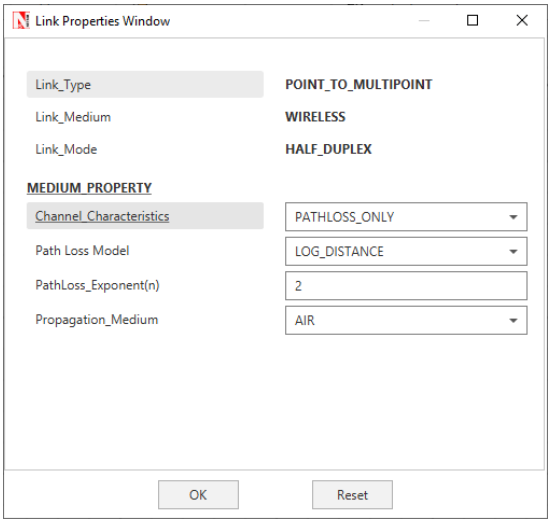
Step 1: A network scenario is designed in NetSim GUI comprising of 2 ENBs, 1 MME, and 2 UEs in the **"LTE/LTE-A"** Network Library.

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

	ENB 1	ENB 2	UE 3	UE 4
X Co-ordinate	1000	4000	1000	4000
Y Co-ordinate	1500	1500	3000	3000

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.

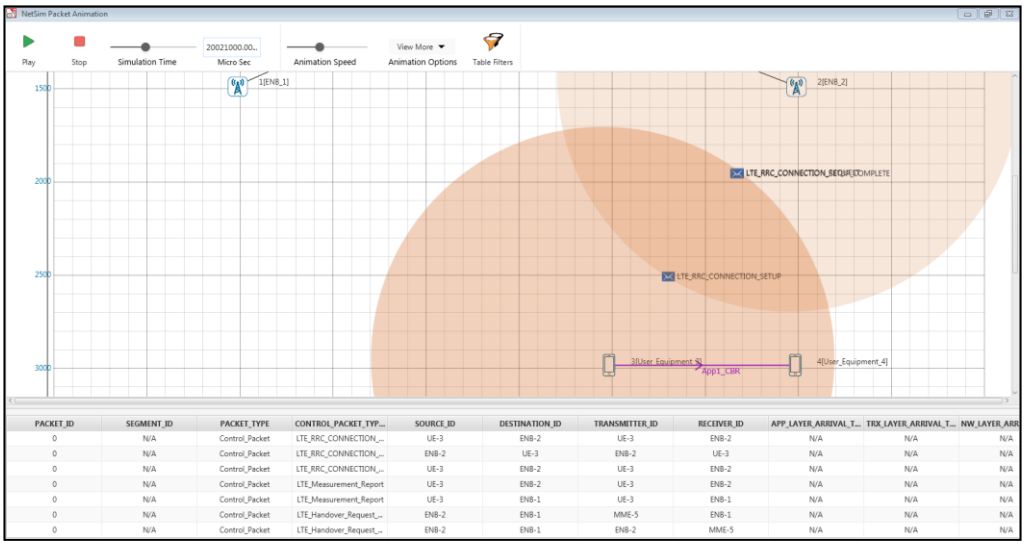
Step 7: Run the Simulation for 50 Seconds.

23.6 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.



23.7 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

23.8 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

24. Understand the working of LTE Device to Device Communication

24.1 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.

24.2 Benefits of D2D communications

Direct communications between devices can provide several benefits to users in various applications where the devices are in close proximity:

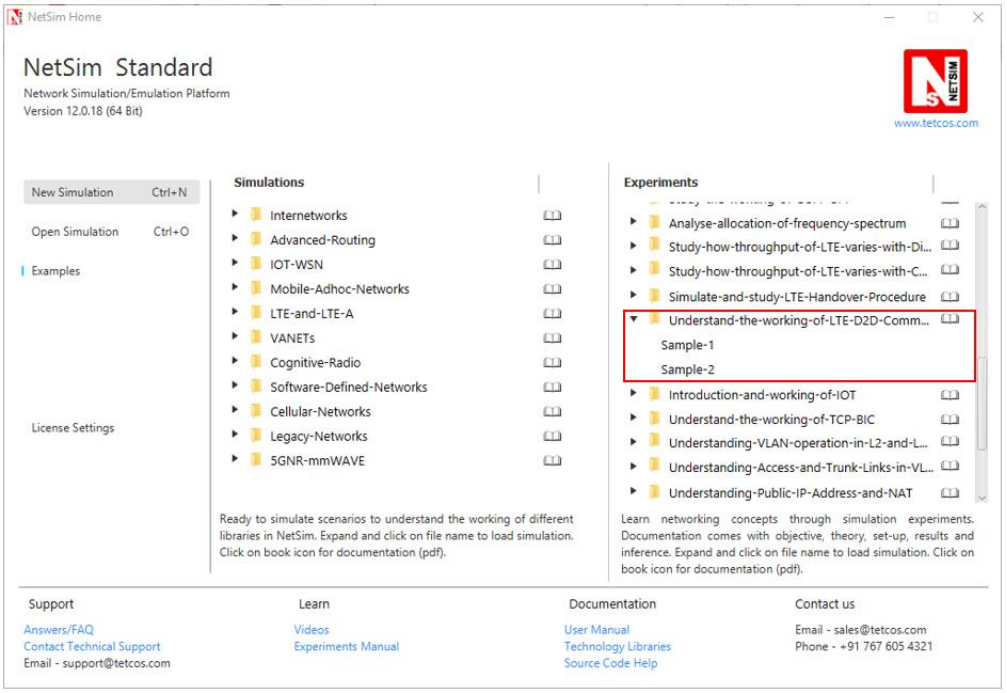
- **Reliable communications:** LTE Device to Device can be used to communicate locally between devices to provide highly reliable communications especially if the LTE network has failed for any reason - even as a result of the disaster.
- **Instant communications:** As the D2D communications does not 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.

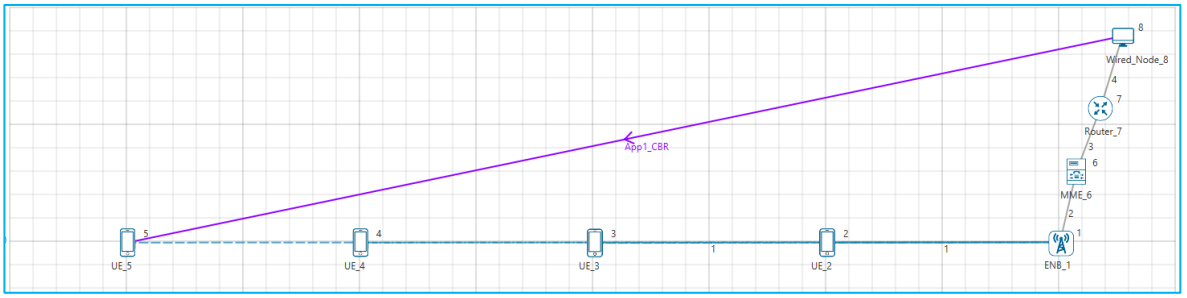
- **Interference reduction:** By not having to communicate directly with a base station, fewer links are required (i.e. essentially only between devices) and this has an impact of the amount of data being transmitted within a given spectrum allocation. This reduces the overall level of interference.
- **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.

24.3 Network Setup:

Open NetSim and click **Examples > Experiments > Understand-the-working-of-LTE-D2D-Communication > Sample-1** as shown below:



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



24.4 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, 1ENB, and 4 UEs in the **"LTE/LTE-A"** Network Library.

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

Device Type	X - Coordinate	Y - Coordinate
eNB 1	4500	1000
UE 2	3500	1000
UE 3	2500	1000
UE 4	1500	1000
UE 5	500	1000

Step 3: TCP Protocol is disabled in all the UEs.

Step 4: 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 5: The Wireless Link properties is set as follows:

Wireless Link Properties	
Channel Characteristics	Path loss only
Path loss Model	Log Distance
Path loss exponent	3.2

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 8 i.e. Source to UE 5 i.e. Destination with Packet Size remaining 1460Bytes and Inter Arrival Time remaining 20000µs.

Step 7: 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 8: Run the Simulation for 10 Seconds.

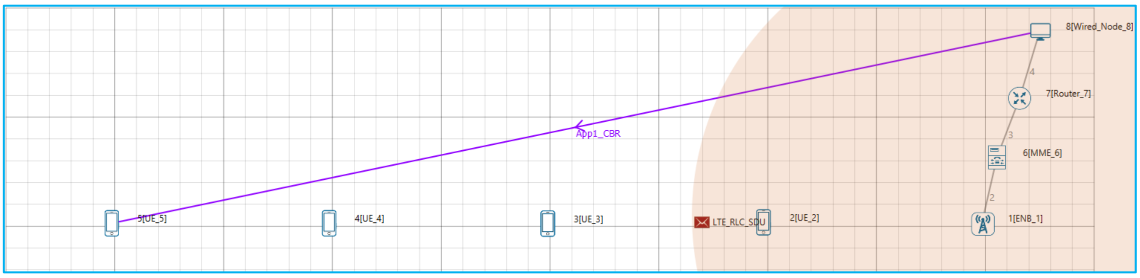
The following changes in settings are done from the previous sample:

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.

24.5 Output:

Sample 1: Without D2D:

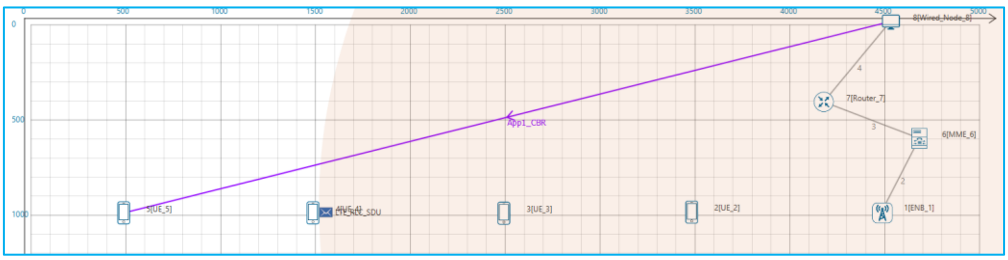


As shown in above figure, application is set from Wired Node 8 to UE 5. As UE 5 is far away from eNB 1, there is too much of attenuation and packets from eNB 1 to UE 5 get errored. We can observe this in Packet Animation. This results in a very low or zero throughput. Users can also observe in the packet 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.

PACKET_ID	SEGMENT_ID	PACKET_TYPE	CONTROL_PACK	SOURCE	DESTI	TRANSMITTER	RECEIVER	PACKET_STATUS
0	N/A	Control_Packet	LTE_RLC_SDU	ENB-1	UE-5	ENB-1	UE-5	Errored
0	N/A	Control_Packet	LTE_RLC_SDU	ENB-1	UE-5	ENB-1	UE-5	Errored
0	N/A	Control_Packet	LTE_RLC_SDU	ENB-1	UE-5	ENB-1	UE-5	Errored
0	N/A	Control_Packet	LTE_RLC_SDU	ENB-1	UE-5	ENB-1	UE-5	Errored
0	N/A	Control_Packet	LTE_RLC_SDU	ENB-1	UE-5	ENB-1	UE-5	Errored
0	N/A	Control_Packet	LTE_RLC_SDU	ENB-1	UE-5	ENB-1	UE-5	Errored
0	N/A	Control_Packet	LTE_RLC_SDU	ENB-1	UE-5	ENB-1	UE-5	Errored
0	N/A	Control_Packet	LTE_RLC_SDU	ENB-1	UE-5	ENB-1	UE-5	Errored
0	N/A	Control_Packet	LTE_RLC_SDU	ENB-1	UE-5	ENB-1	UE-5	Errored
0	N/A	Control_Packet	LTE_RLC_SDU	ENB-1	UE-5	ENB-1	UE-5	Errored
0	N/A	Control_Packet	LTE_RLC_SDU	ENB-1	UE-5	ENB-1	UE-5	Errored
0	N/A	Control_Packet	LTE_RLC_SDU	ENB-1	UE-5	ENB-1	UE-5	Errored

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.

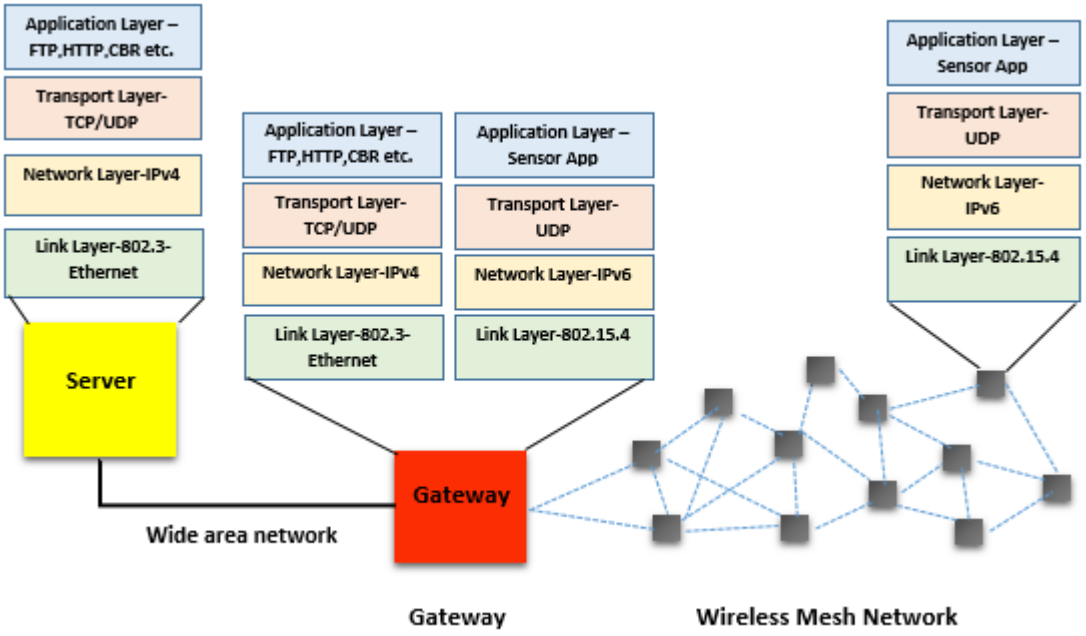
PACKET_ID	SEGMENT_ID	PACKET_TYPE	CONTROL_P	SOURCE_	DEST	TRANSM	RECEIVER_I	APP_LAYE	PACKET_STATUS
0	N/A	Control_Packet	LTE_RLC_SDU	ENB-1	UE-4	ENB-1	UE-4	N/A	Successful
0	N/A	Control_Packet	LTE_RLC_SDU	ENB-1	UE-4	ENB-1	UE-4	N/A	Successful
0	N/A	Control_Packet	LTE_RLC_SDU	UE-4	UE-5	UE-4	UE-5	N/A	Successful
0	N/A	Control_Packet	LTE_RLC_SDU	ENB-1	UE-4	ENB-1	UE-4	N/A	Successful
0	N/A	Control_Packet	LTE_RLC_SDU	ENB-1	UE-4	ENB-1	UE-4	N/A	Successful
0	N/A	Control_Packet	LTE_RLC_SDU	UE-4	UE-5	UE-4	UE-5	N/A	Successful
0	N/A	Control_Packet	LTE_RLC_SDU	ENB-1	UE-4	ENB-1	UE-4	N/A	Successful
0	N/A	Control_Packet	LTE_RLC_SDU	ENB-1	UE-4	ENB-1	UE-4	N/A	Successful
0	N/A	Control_Packet	LTE_RLC_SDU	UE-4	UE-5	UE-4	UE-5	N/A	Successful
0	N/A	Control_Packet	LTE_RLC_SDU	ENB-1	UE-4	ENB-1	UE-4	N/A	Successful
0	N/A	Control_Packet	LTE_RLC_SDU	ENB-1	UE-4	ENB-1	UE-4	N/A	Successful
0	N/A	Control_Packet	LTE_RLC_SDU	UE-4	UE-5	UE-4	UE-5	N/A	Successful
0	N/A	Control_Packet	LTE_RLC_SDU	ENB-1	UE-4	ENB-1	UE-4	N/A	Successful
0	N/A	Control_Packet	LTE_RLC_SDU	ENB-1	UE-4	ENB-1	UE-4	N/A	Successful

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.

25. Introduction and working of Internet of Things (IoT)

25.1 Introduction:

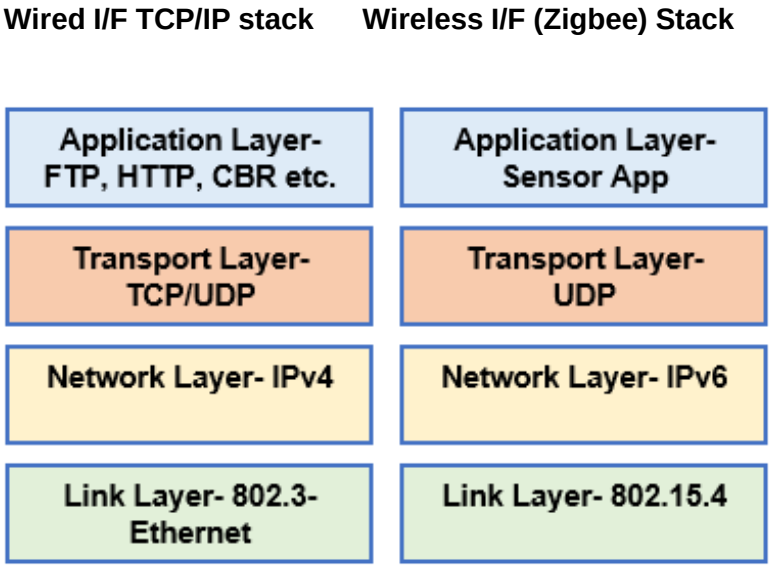
Internet of Things (**IoT**) is a network of physical devices, vehicles, buildings and other items embedded with electronics, software, sensors, actuators, and network connectivity that enable these objects to collect and exchange data. An IoT network allows objects to be sensed and/or controlled remotely 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.



25.2 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.

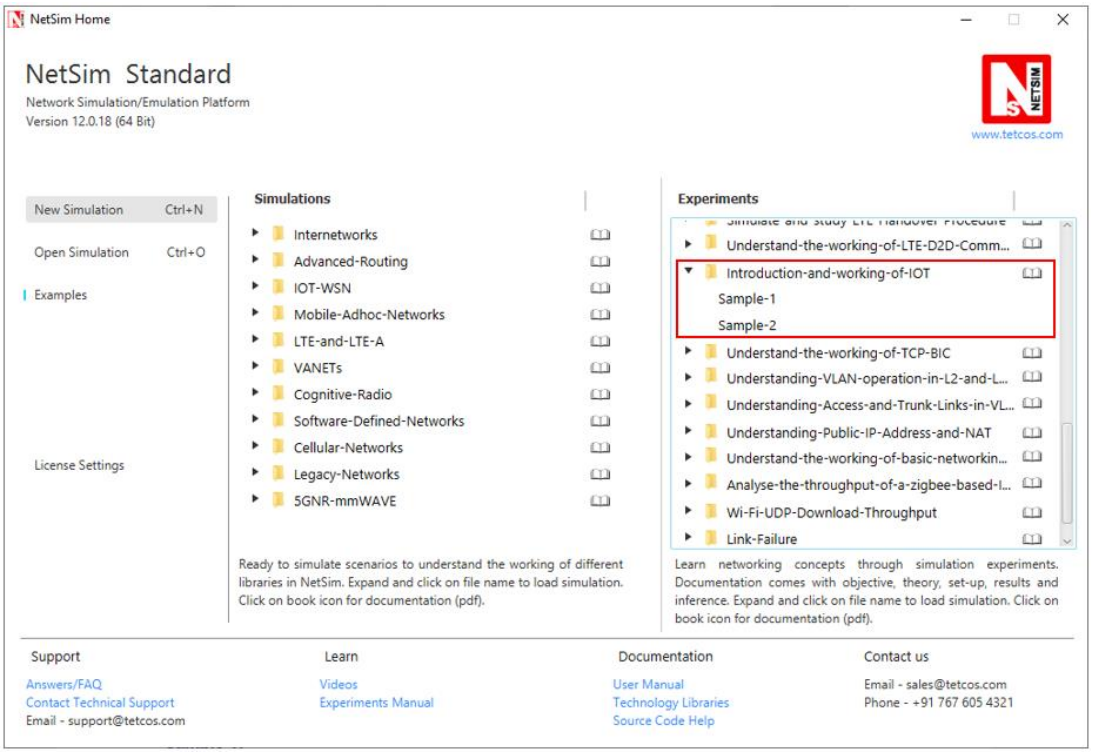


6LowPAN Gateway Stack at wired and wireless Interfaces

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.

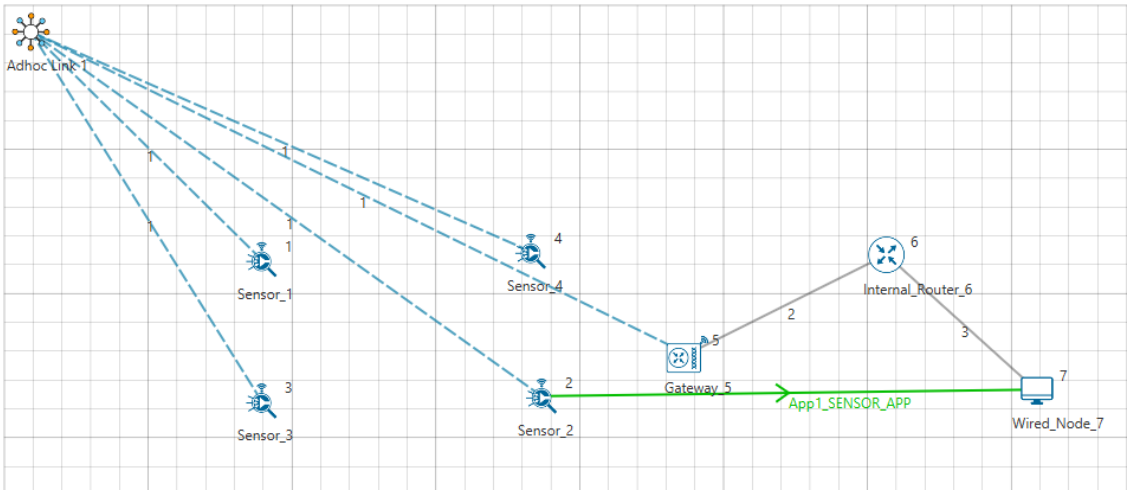
25.3 Network Setup:

Open NetSim and click **Examples > Experiments > Introduction-and-working-of-IOT** as shown below:



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

Sample 1:



25.4 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 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.

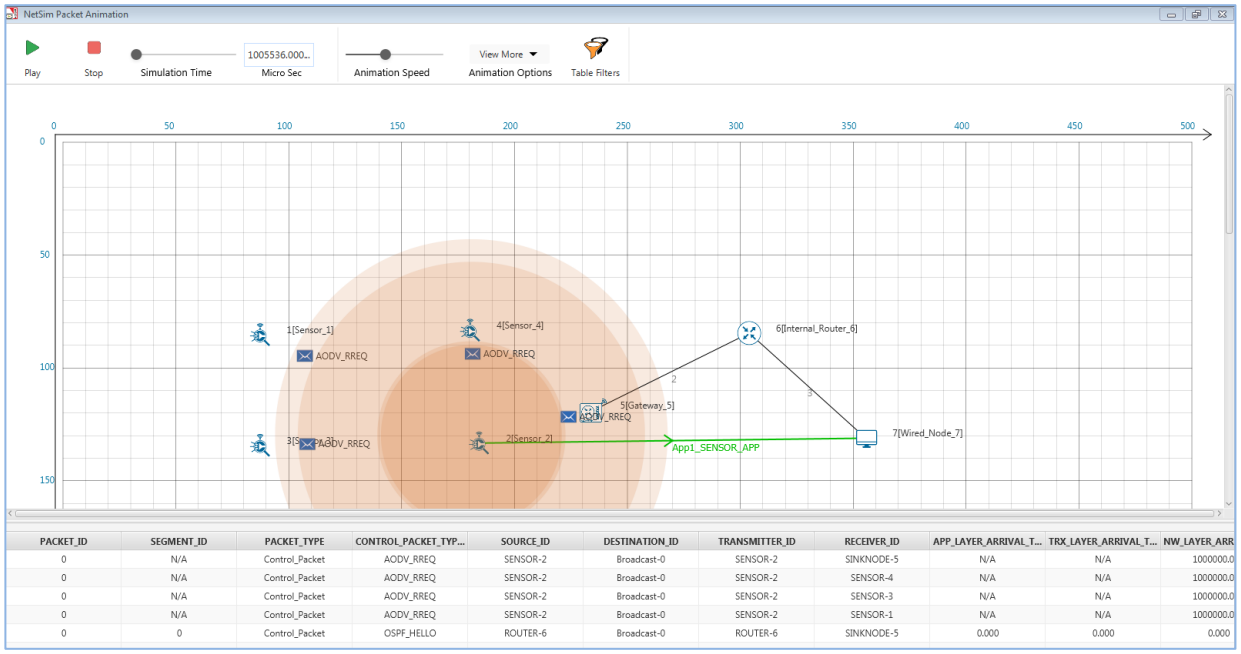
The Ad hoc link properties is set to **NO PATHLOSS** for the channel characteristics.

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.

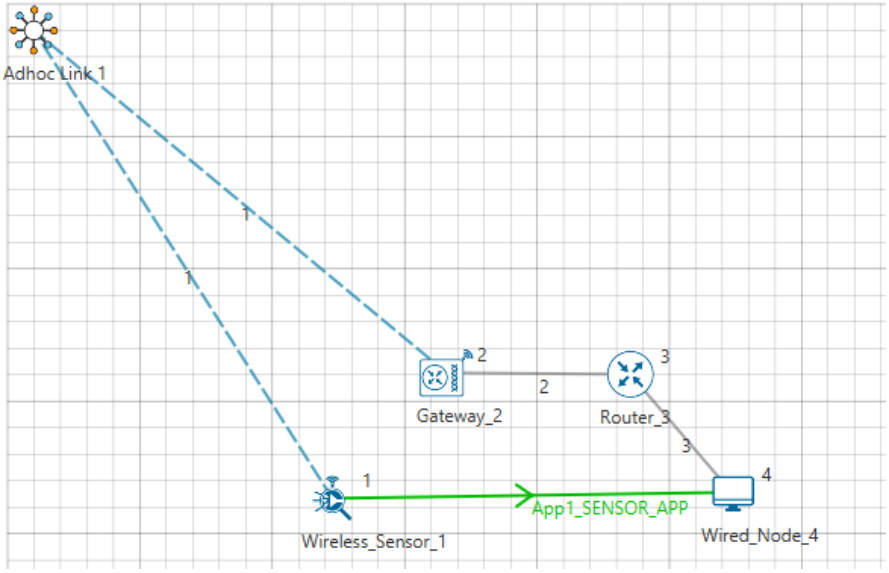
Step 5: Enable the packet trace and run the Simulation for 100 Seconds.

25.5 Output:



PACKET_ID	SEGMENT_ID	PACKET_TYPE	CONTROL_PACKET_TYP...	SOURCE_ID	DESTINATION_ID	TRANSMITTER_ID	RECEIVER_ID	APP_LAYER_ARRIVAL_T...	TRX_LAYER_ARRIVAL_T...	NW_LAYER_ARR
0	N/A	Control_Packet	AODV_RREQ	SENSOR-2	Broadcast-0	SENSOR-2	SINKNODE-5	N/A	N/A	1000000.0
0	N/A	Control_Packet	AODV_RREQ	SENSOR-2	Broadcast-0	SENSOR-2	SENSOR-4	N/A	N/A	1000000.0
0	N/A	Control_Packet	AODV_RREQ	SENSOR-2	Broadcast-0	SENSOR-2	SENSOR-3	N/A	N/A	1000000.0
0	N/A	Control_Packet	AODV_RREQ	SENSOR-2	Broadcast-0	SENSOR-2	SENSOR-1	N/A	N/A	1000000.0
0	0	Control_Packet	OSPF_HELLO	ROUTER-6	Broadcast-0	ROUTER-6	SINKNODE-5	0.000	0.000	0.000

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.

25.6 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.

	PACKET	CONTROL_PACKET	SOURCE_II	DESTINATIC	TRANSMITT	RECEIVE	OURCE_IP	DESTINATION_IP	GATEWAY_IP	NEXT_HOP_IP
7	Sensing	App1_SENSOR_APP	SENSOR-1	NODE-4	SENSOR-1	SINKNODE-	DEC:3017:E256:9BB8:1FE7:	FDEC:3017:E256:9BB8:1FE7:A31F:17B1:87F5	FDEC:3017:E256:9BB8:1FE7:9481:58A1:D94B	FDEC:3017:E256:9BB8:1FE7:A31F:17B1:87F5
8	Sensing	App1_SENSOR_APP	SENSOR-1	NODE-4	SINKNODE-2	ROUTER-3	DEC:3017:E256:9BB8:1FE7:	11.2.1.2	11.2.1.1	11.2.1.2
9	Sensing	App1_SENSOR_APP	SENSOR-1	NODE-4	ROUTER-3	NODE-4	DEC:3017:E256:9BB8:1FE7:	11.3.1.2	11.3.1.1	11.3.1.2
11	Sensing	App1_SENSOR_APP	SENSOR-1	NODE-4	SENSOR-1	SINKNODE-	DEC:3017:E256:9BB8:1FE7:	FDEC:3017:E256:9BB8:1FE7:A31F:17B1:87F5	FDEC:3017:E256:9BB8:1FE7:9481:58A1:D94B	FDEC:3017:E256:9BB8:1FE7:A31F:17B1:87F5
12	Sensing	App1_SENSOR_APP	SENSOR-1	NODE-4	SINKNODE-2	ROUTER-3	DEC:3017:E256:9BB8:1FE7:	11.2.1.2	11.2.1.1	11.2.1.2
13	Sensing	App1_SENSOR_APP	SENSOR-1	NODE-4	ROUTER-3	NODE-4	DEC:3017:E256:9BB8:1FE7:	11.3.1.2	11.3.1.1	11.3.1.2
17	Sensing	App1_SENSOR_APP	SENSOR-1	NODE-4	SENSOR-1	SINKNODE-	DEC:3017:E256:9BB8:1FE7:	FDEC:3017:E256:9BB8:1FE7:A31F:17B1:87F5	FDEC:3017:E256:9BB8:1FE7:9481:58A1:D94B	FDEC:3017:E256:9BB8:1FE7:A31F:17B1:87F5
18	Sensing	App1_SENSOR_APP	SENSOR-1	NODE-4	SINKNODE-2	ROUTER-3	DEC:3017:E256:9BB8:1FE7:	11.2.1.2	11.2.1.1	11.2.1.2
19	Sensing	App1_SENSOR_APP	SENSOR-1	NODE-4	ROUTER-3	NODE-4	DEC:3017:E256:9BB8:1FE7:	11.3.1.2	11.3.1.1	11.3.1.2
23	Sensing	App1_SENSOR_APP	SENSOR-1	NODE-4	SENSOR-1	SINKNODE-	DEC:3017:E256:9BB8:1FE7:	FDEC:3017:E256:9BB8:1FE7:A31F:17B1:87F5	FDEC:3017:E256:9BB8:1FE7:9481:58A1:D94B	FDEC:3017:E256:9BB8:1FE7:A31F:17B1:87F5
24	Sensing	App1_SENSOR_APP	SENSOR-1	NODE-4	SINKNODE-2	ROUTER-3	DEC:3017:E256:9BB8:1FE7:	11.2.1.2	11.2.1.1	11.2.1.2
25	Sensing	App1_SENSOR_APP	SENSOR-1	NODE-4	ROUTER-3	NODE-4	DEC:3017:E256:9BB8:1FE7:	11.3.1.2	11.3.1.1	11.3.1.2
29	Sensing	App1_SENSOR_APP	SENSOR-1	NODE-4	SENSOR-1	SINKNODE-	DEC:3017:E256:9BB8:1FE7:	FDEC:3017:E256:9BB8:1FE7:A31F:17B1:87F5	FDEC:3017:E256:9BB8:1FE7:9481:58A1:D94B	FDEC:3017:E256:9BB8:1FE7:A31F:17B1:87F5
30	Sensing	App1_SENSOR_APP	SENSOR-1	NODE-4	SINKNODE-2	ROUTER-3	DEC:3017:E256:9BB8:1FE7:	11.2.1.2	11.2.1.1	11.2.1.2
31	Sensing	App1_SENSOR_APP	SENSOR-1	NODE-4	ROUTER-3	NODE-4	DEC:3017:E256:9BB8:1FE7:	11.3.1.2	11.3.1.1	11.3.1.2
33	Sensing	App1_SENSOR_APP	SENSOR-1	NODE-4	SENSOR-1	SINKNODE-	DEC:3017:E256:9BB8:1FE7:	FDEC:3017:E256:9BB8:1FE7:A31F:17B1:87F5	FDEC:3017:E256:9BB8:1FE7:9481:58A1:D94B	FDEC:3017:E256:9BB8:1FE7:A31F:17B1:87F5
34	Sensing	App1_SENSOR_APP	SENSOR-1	NODE-4	SINKNODE-2	ROUTER-3	DEC:3017:E256:9BB8:1FE7:	11.2.1.2	11.2.1.1	11.2.1.2
35	Sensing	App1_SENSOR_APP	SENSOR-1	NODE-4	ROUTER-3	NODE-4	DEC:3017:E256:9BB8:1FE7:	11.3.1.2	11.3.1.1	11.3.1.2
39	Sensing	App1_SENSOR_APP	SENSOR-1	NODE-4	SENSOR-1	SINKNODE-	DEC:3017:E256:9BB8:1FE7:	FDEC:3017:E256:9BB8:1FE7:A31F:17B1:87F5	FDEC:3017:E256:9BB8:1FE7:9481:58A1:D94B	FDEC:3017:E256:9BB8:1FE7:A31F:17B1:87F5
40	Sensing	App1_SENSOR_APP	SENSOR-1	NODE-4	SINKNODE-2	ROUTER-3	DEC:3017:E256:9BB8:1FE7:	11.2.1.2	11.2.1.1	11.2.1.2
41	Sensing	App1_SENSOR_APP	SENSOR-1	NODE-4	ROUTER-3	NODE-4	DEC:3017:E256:9BB8:1FE7:	11.3.1.2	11.3.1.1	11.3.1.2
45	Sensing	App1_SENSOR_APP	SENSOR-1	NODE-4	SENSOR-1	SINKNODE-	DEC:3017:E256:9BB8:1FE7:	FDEC:3017:E256:9BB8:1FE7:A31F:17B1:87F5	FDEC:3017:E256:9BB8:1FE7:9481:58A1:D94B	FDEC:3017:E256:9BB8:1FE7:A31F:17B1:87F5
46	Sensing	App1_SENSOR_APP	SENSOR-1	NODE-4	SINKNODE-2	ROUTER-3	DEC:3017:E256:9BB8:1FE7:	11.2.1.2	11.2.1.1	11.2.1.2
47	Sensing	App1_SENSOR_APP	SENSOR-1	NODE-4	ROUTER-3	NODE-4	DEC:3017:E256:9BB8:1FE7:	11.3.1.2	11.3.1.1	11.3.1.2
51	Sensing	App1_SENSOR_APP	SENSOR-1	NODE-4	SENSOR-1	SINKNODE-	DEC:3017:E256:9BB8:1FE7:	FDEC:3017:E256:9BB8:1FE7:A31F:17B1:87F5	FDEC:3017:E256:9BB8:1FE7:9481:58A1:D94B	FDEC:3017:E256:9BB8:1FE7:A31F:17B1:87F5
52	Sensing	App1_SENSOR_APP	SENSOR-1	NODE-4	SINKNODE-2	ROUTER-3	DEC:3017:E256:9BB8:1FE7:	11.2.1.2	11.2.1.1	11.2.1.2
53	Sensing	App1_SENSOR_APP	SENSOR-1	NODE-4	ROUTER-3	NODE-4	DEC:3017:E256:9BB8:1FE7:	11.3.1.2	11.3.1.1	11.3.1.2

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

26.1 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_{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 (Smax) instead of increasing window directly to that midpoint in the next RTT, we increase it by Smax until the distance becomes less than Smax, 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:

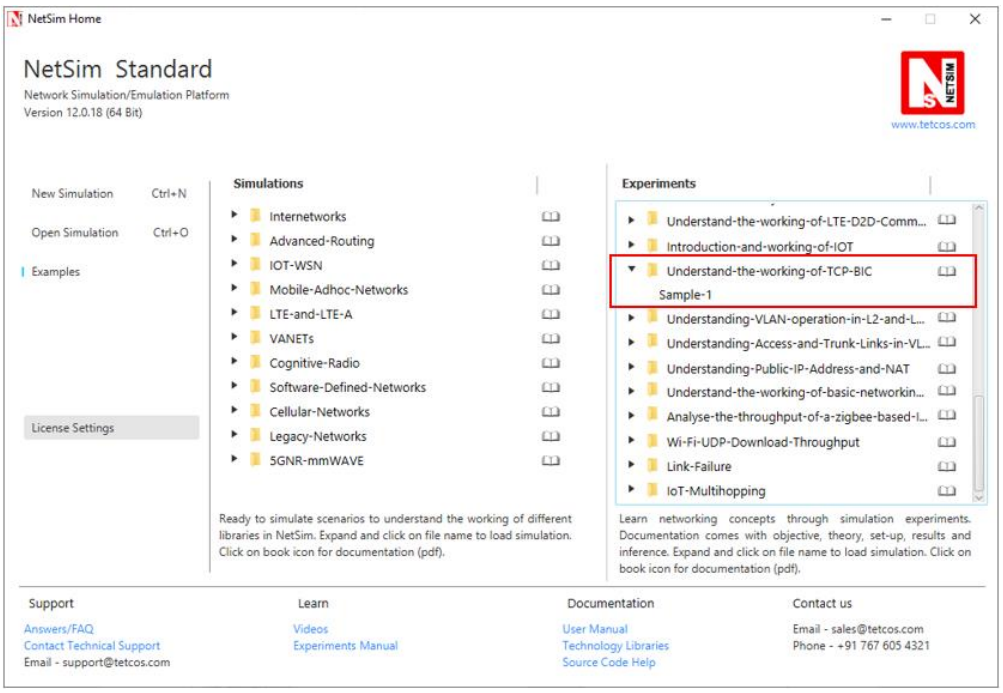
After the window grows past the current maximum, the maximum is unknown. At this time, binary search sets its maximum to be a default maximum (a large constant) and the current window size to be the minimum. So, the target midpoint can be very far. According to binary increase, if the target midpoint 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,…, 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:

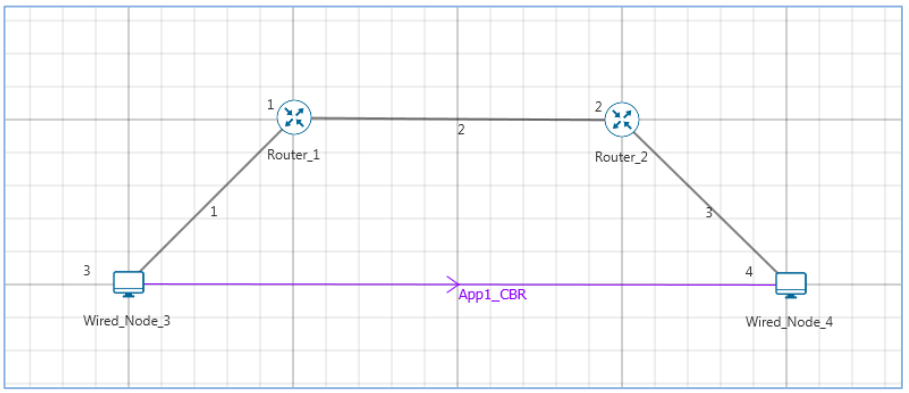
It can be shown that under a completely synchronized loss model, binary search increase combined with multiplicative decrease converges to a fair share. Suppose there are two flows with different window sizes, but with the same RTT. Since the larger window reduces more in multiplicative decrease (with a fixed factor β), 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 max_wini and min_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. max_wini = (max_wini-min_wini)/2), and then readjust the target. After that apply the normal binary increase. This strategy is called fast convergence.

26.2 Network setup:

Open NetSim and click **Examples > Experiments > Understand-the-working-of-TCP-BIC > Sample-1** as shown below:



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



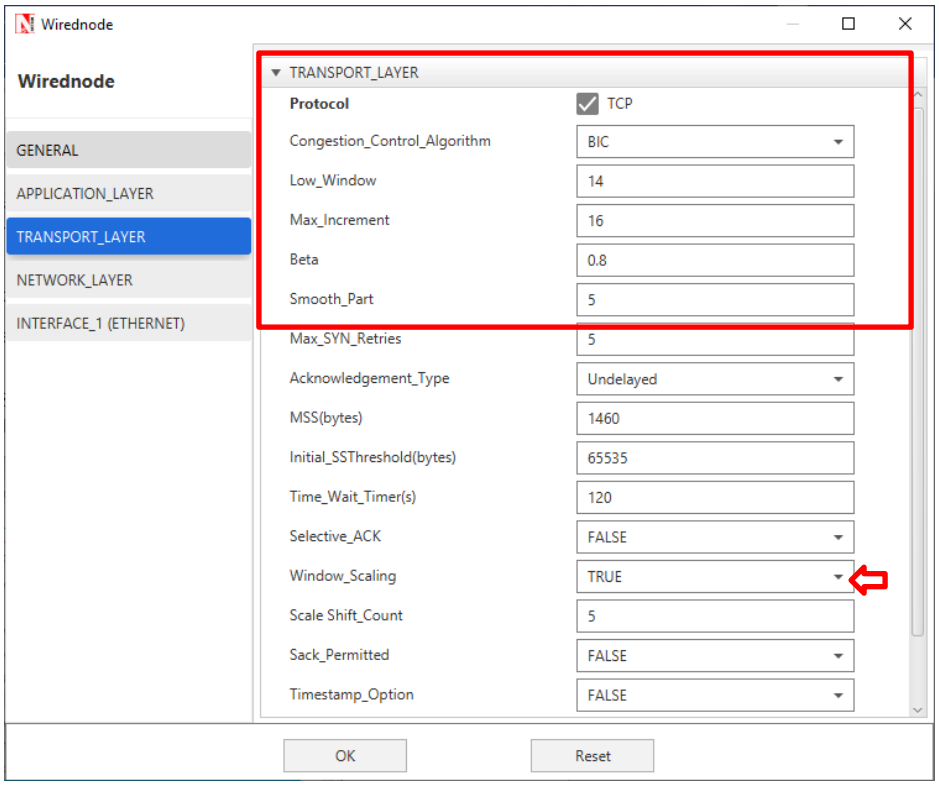
26.3 Procedure:

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:

Link Properties	Wired Link 1	Wired Link 2	Wired Link 3
Uplink Speed (Mbps)	20	100	20
Downlink Speed (Mbps)	20	100	20
Uplink propagation delay (µs)	5	1000	5
Downlink propagation delay (µs)	5	1000	5
Uplink BER	0.00000001	0.00000001	0.00000001
Downlink BER	0.00000001	0.00000001	0.00000001

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.

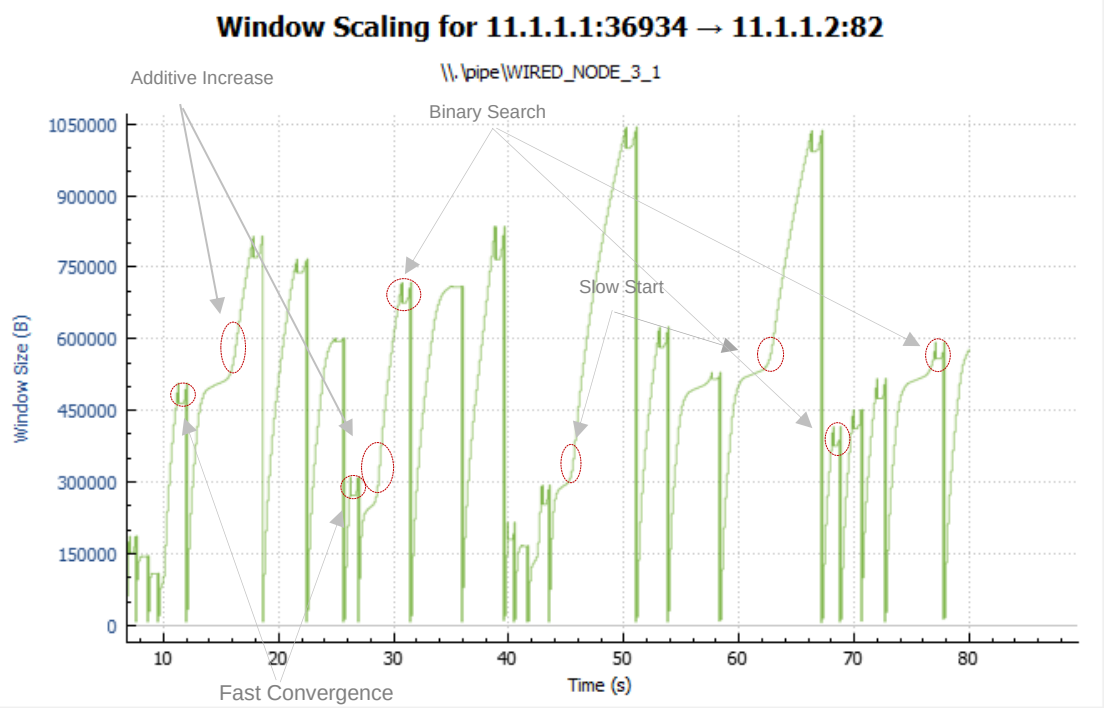
A CBR Application is generated from Wired Node 3 i.e. Source to Wired Node 4 i.e. Destination with Packet Size set to 70 Bytes and Inter Arrival Time set to 400 µs. Additionally, the **"Start Time"** parameter is set to 20 Seconds.

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:

$$Generation\ Rate\ (Mbps)\ =\ Packet\ Size\ (Bytes)\ *\ 8/Interarrival\ time\ (\mu s)$$

Step 6: Click on Run simulation. The simulation time is set to 100 seconds.

26.4 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 - 7.7.5 Window Scaling" in user manual)

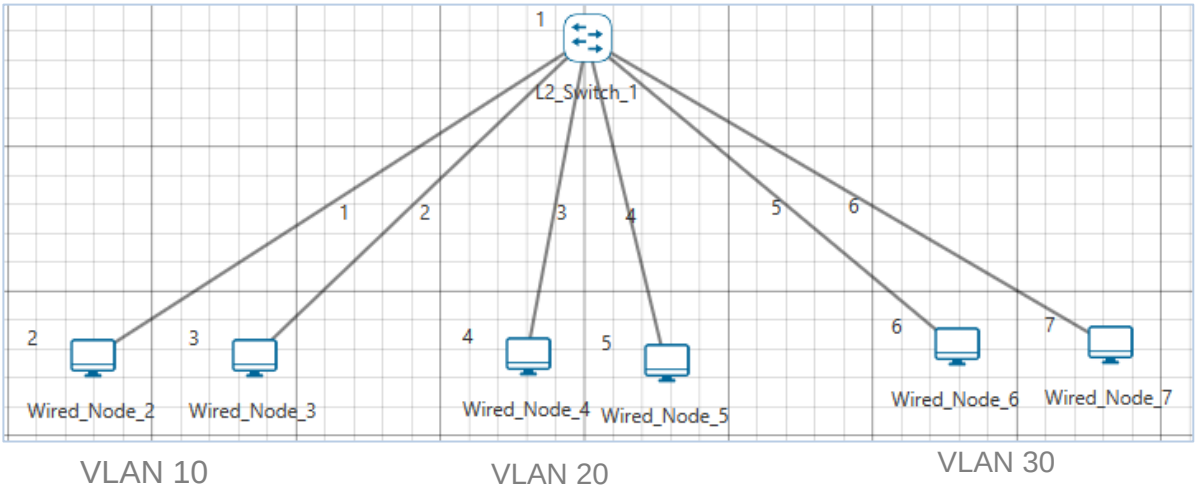
The graph shown above is a plot of Congestion Window vs Time of BIC for the scenario shown above. Each point on the graph represents the congestion window at the time when the packet is sent. You can observe Binary Search, Additive Increase, Fast Convergence, Slow Start phases in the above graph.

27. Understanding VLAN operation in L2 and L3 Switches

27.1 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.

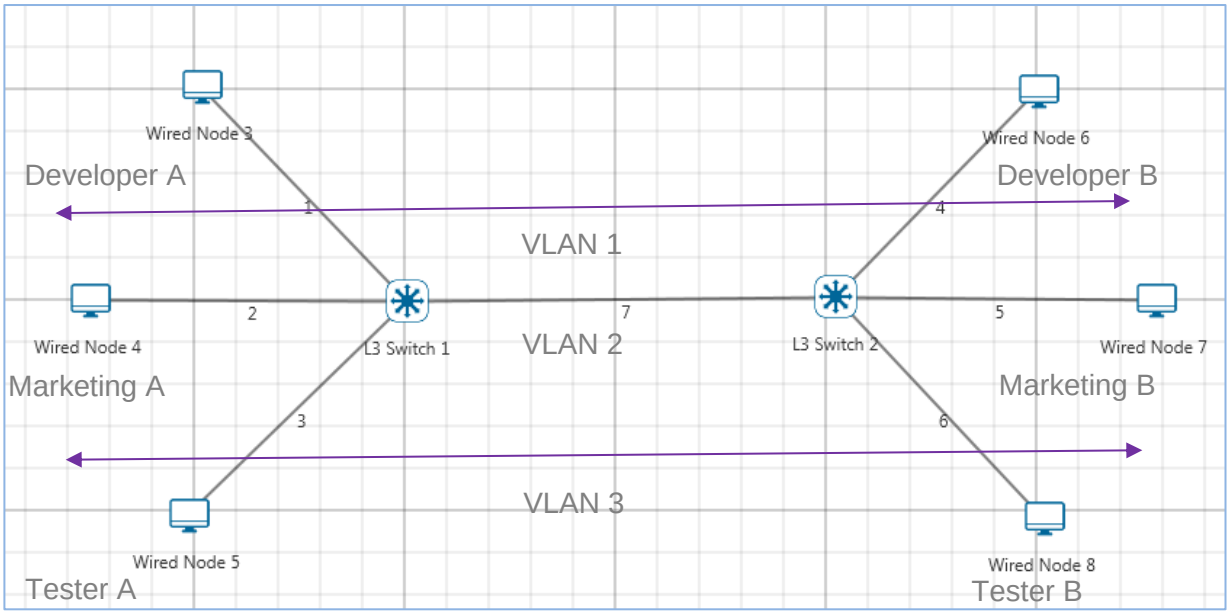


A VLAN behaves just like a LAN in all respects but with additional flexibility. By using VLAN technology, it is possible to subdivide a single physical switch into several logical switches. VLANs are implemented by using the appropriate switch configuration commands to create the VLANs and assign specific 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.



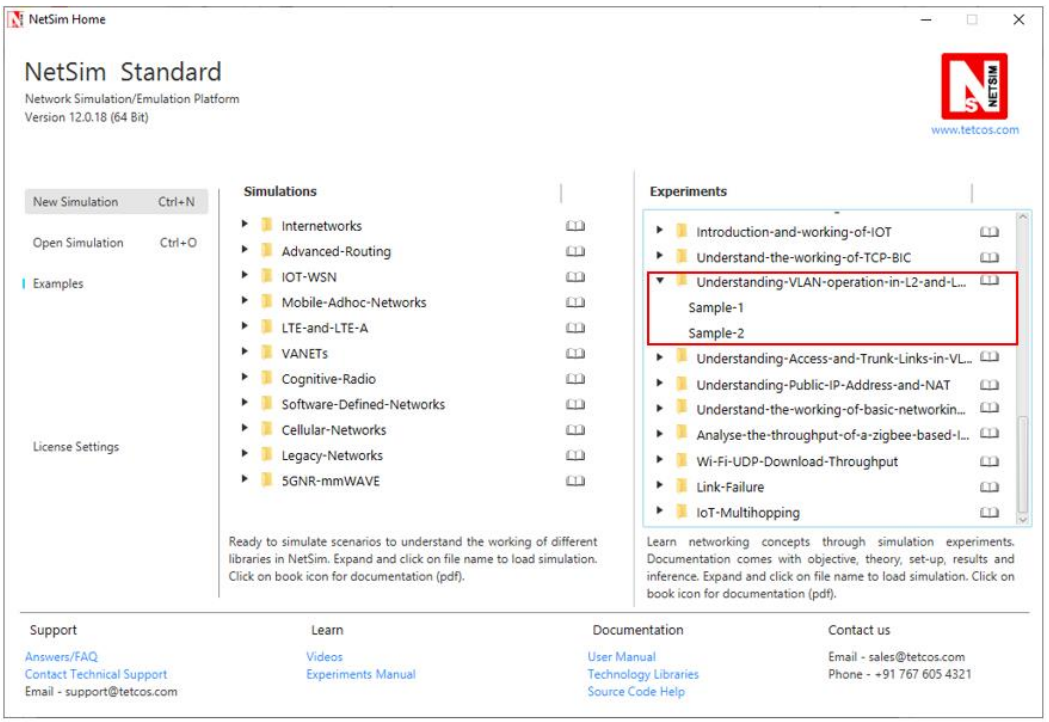
VLANs divide broadcast domains in a LAN environment. Whenever hosts in one VLAN need to communicate with hosts in another VLAN, the traffic must be routed between them. This is known as Inter-VLAN routing. This can be possible by using L3 switch.

What is a layer 3 switch?

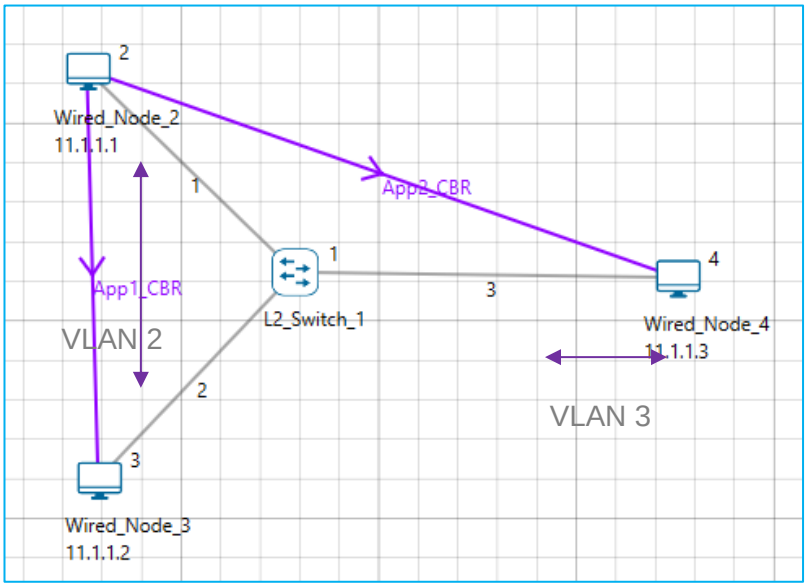
Layer 3 switch (also known as a multi-layer switch) is a multi-functional device that have the same functionality like a layer 2 switch, but behaves like a router when necessary. It's generally faster than a router due to its hardware based routing functions, but it's also more expensive than a normal switch.

27.2 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:



27.3 Procedure:

Sample 1: Intra-VLAN:

Intra-VLAN is a mechanism in which hosts in same VLAN can communicate to each other.

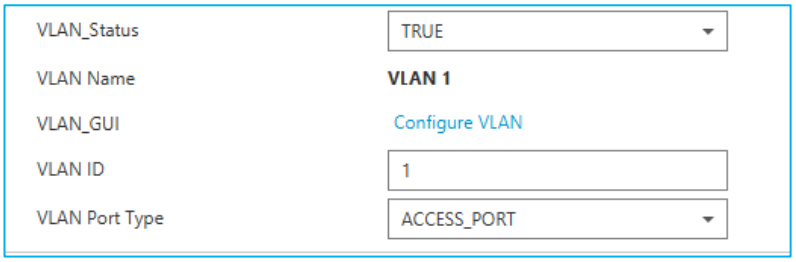
The following set of procedures were done to generate this sample:

Step 1: A network scenario is designed in NetSim GUI comprising of 3 Wired Nodes and 1 L2 Switch in the **"Internetworks"** Network Library.

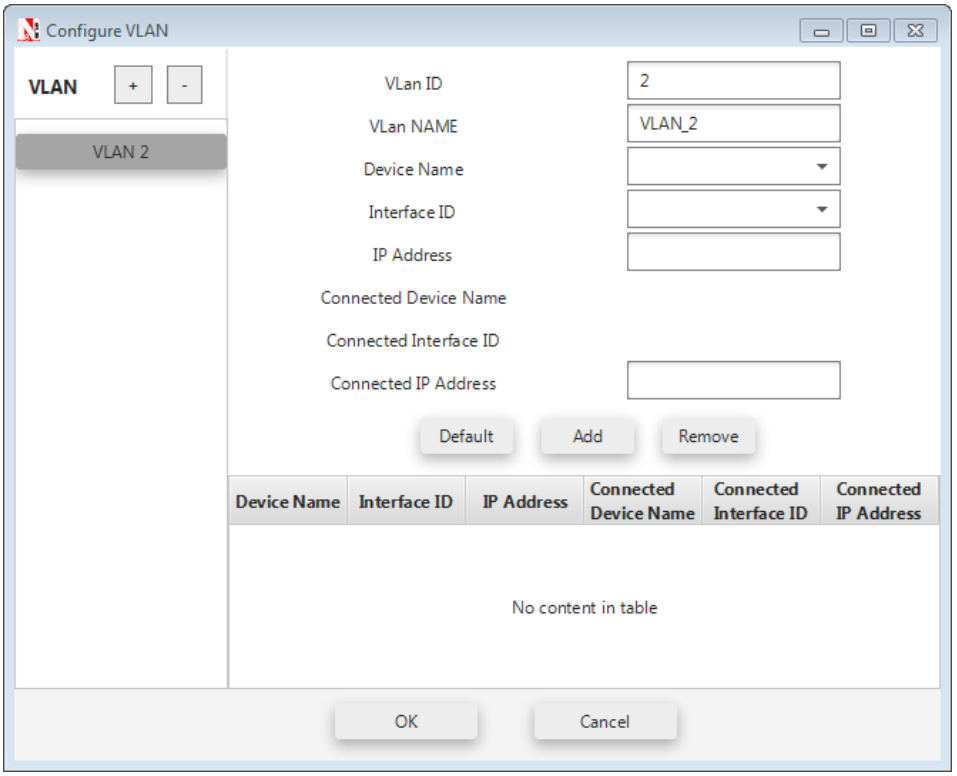
Step 2: L2 Switch 1 Properties are configured as follows:

Switch 1			
Interface ID	VLAN Status	VLAN ID	VLAN Port Type
Interface_1	TRUE	2	Access _Port
Interface_2	TRUE	2	Access _Port
Interface_3	TRUE	3	Access _Port

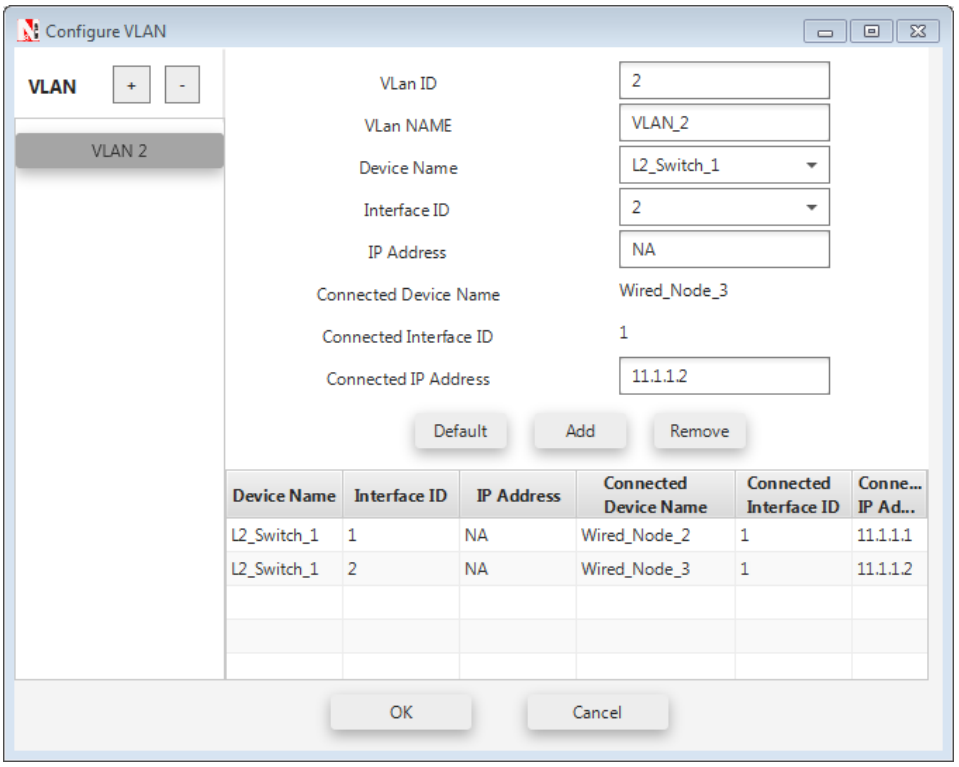
In all the INTERFACE (ETHERNET) > DATALINK LAYER Properties of L2 Switch 1, **"VLAN Status"** is set to TRUE.



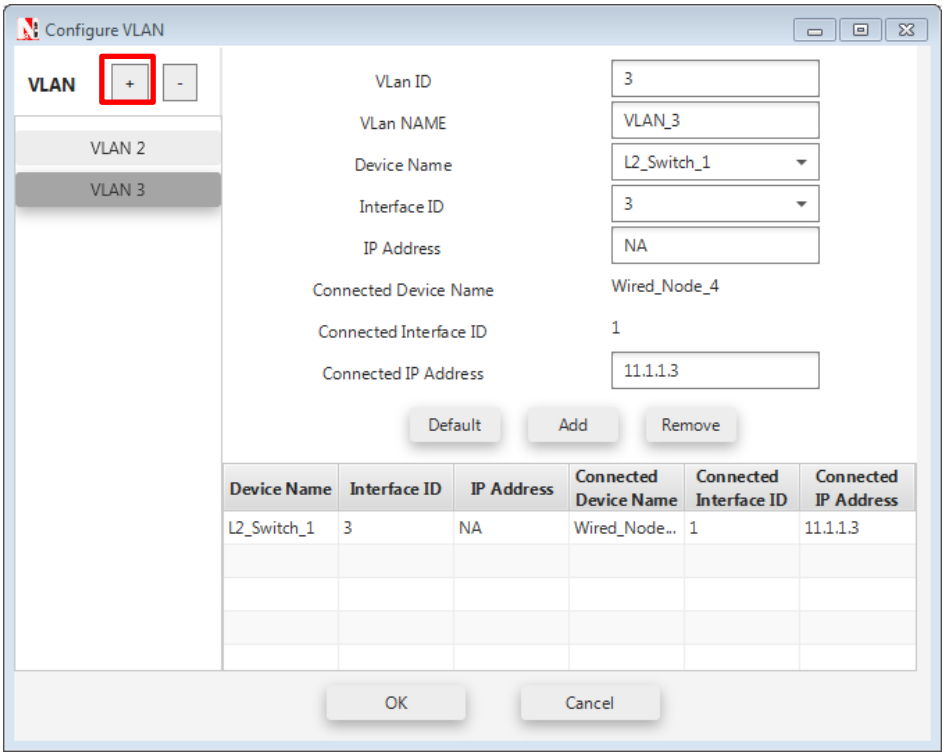
Now click on **"Configure VLAN"** option and the VLAN 2 fields are entered as shown below:



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.

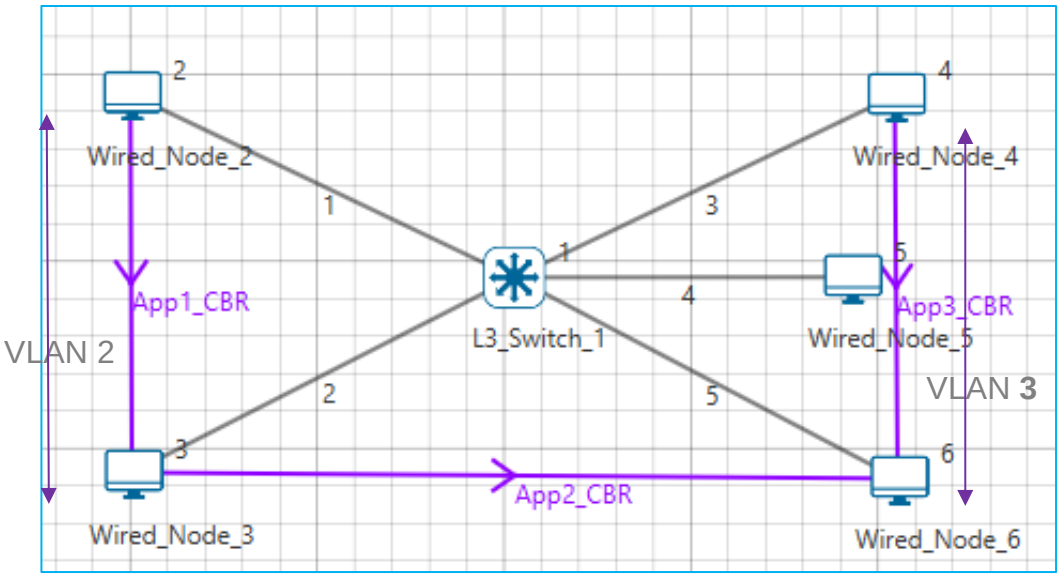


And then we can add the entry to it.

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:

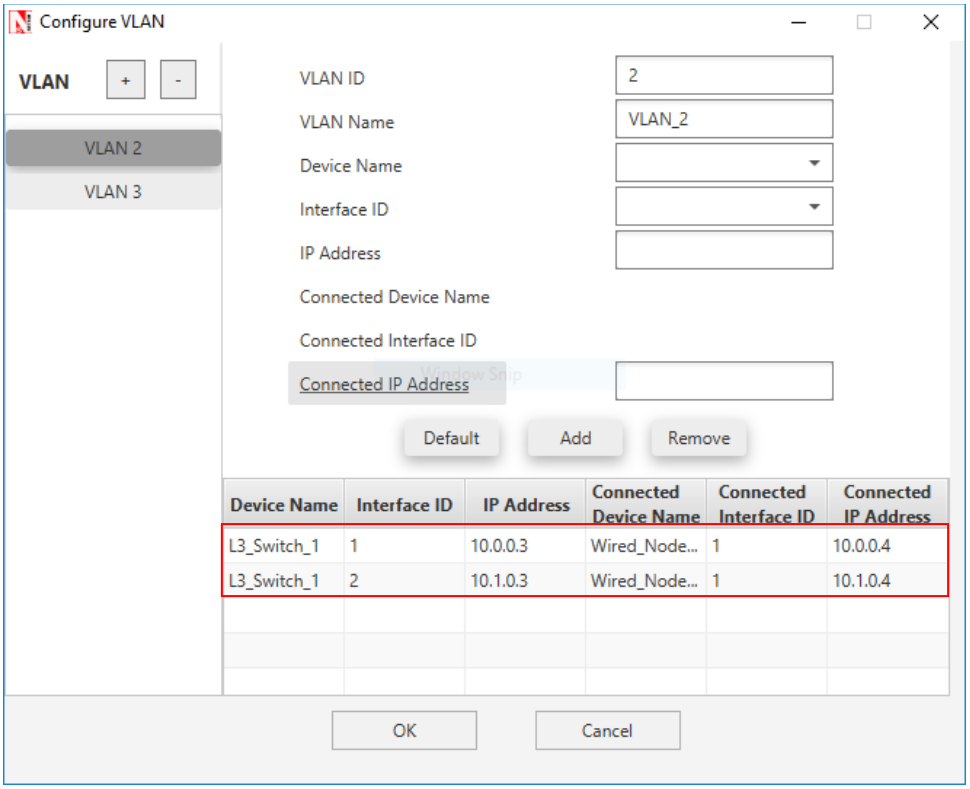
Node	Wired Node2	Wired Node3	Wired Node4	Wired Node5	Wired Node6
	I/f1_Ethernet	I/f1_Ethernet	I/f1_Ethernet	I/f1_Ethernet	I/f1_Ethernet
IP Address	10.0.0.4	10.1.0.4	11.2.0.4	11.3.0.4	11.4.0.4
Default Gateway	10.0.0.3	10.1.0.3	11.2.0.3	11.3.0.3	11.4.0.3

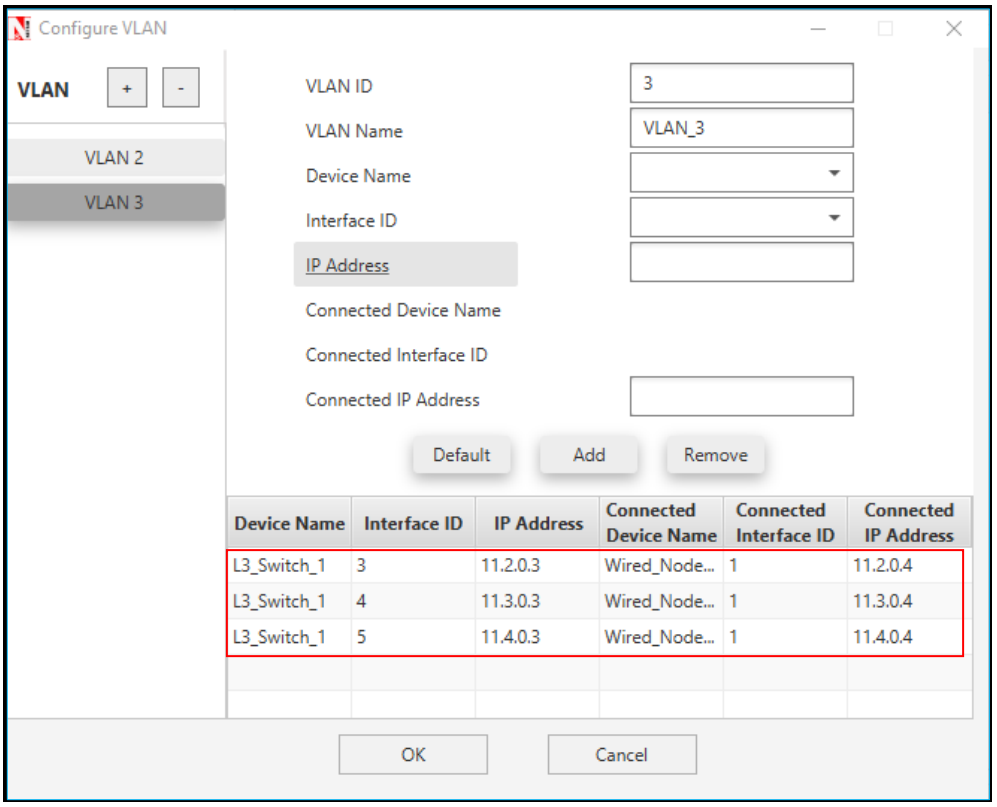
Step 3: The L3 Switch 1 Properties are set as per the below table:

L3 Switch	I/f1_Ethernet	I/f2_Ethernet	I/f3_Ethernet	I/f4_Ethernet	I/f5_Ethernet
	IP Address	IP Address	IP Address	IP Address	IP Address
L3 Switch 1	10.0.0.3	10.1.0.3	11.2.0.3	11.3.0.3	11.4.0.3

L3 Switch 1			
Interface ID	VLAN Status	VLAN ID	VLAN Port Type
Interface_1	TRUE	2	Access _Port
Interface_2	TRUE	2	Access _Port
Interface_3	TRUE	3	Access _Port
Interface_4	TRUE	3	Access _Port
Interface_5	TRUE	3	Access _Port

The VLAN configurations done are shown as follows:





Step 3: Run simulation for 10 seconds and observe the throughputs.

27.4 Output and Inference: I

Throughput (Mbps)	
Application 1	0.58
Application 2	0

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.

27.5 Output and Inference: II

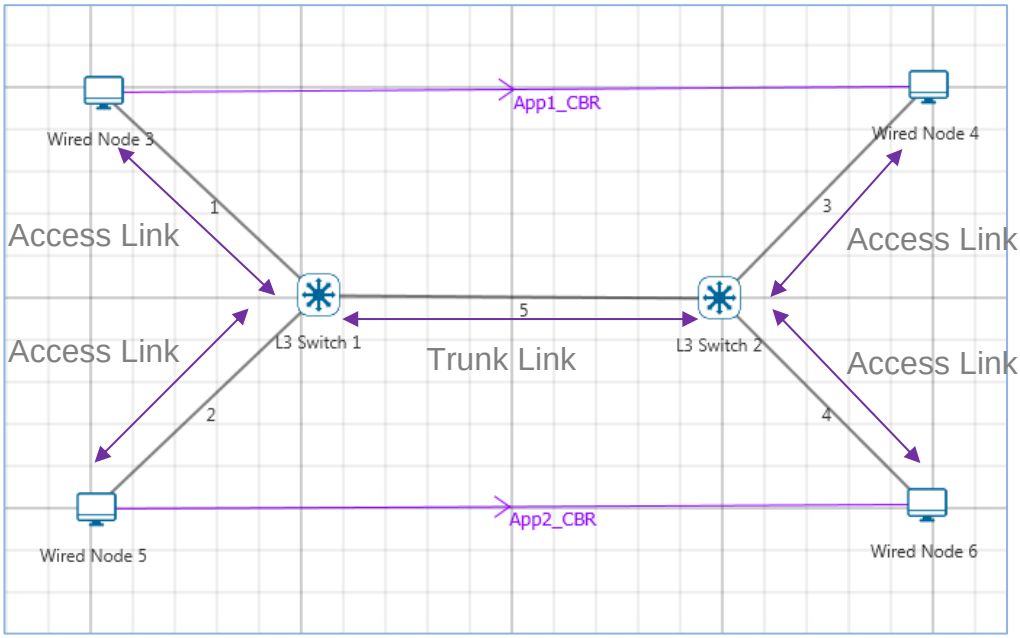
Throughput (Mbps)	
Application 1	**0.58**
Application 2	**0.58**
Application 3	**0.58**

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.

28. Understanding Access and Trunk Links in VLANs

28.1 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.

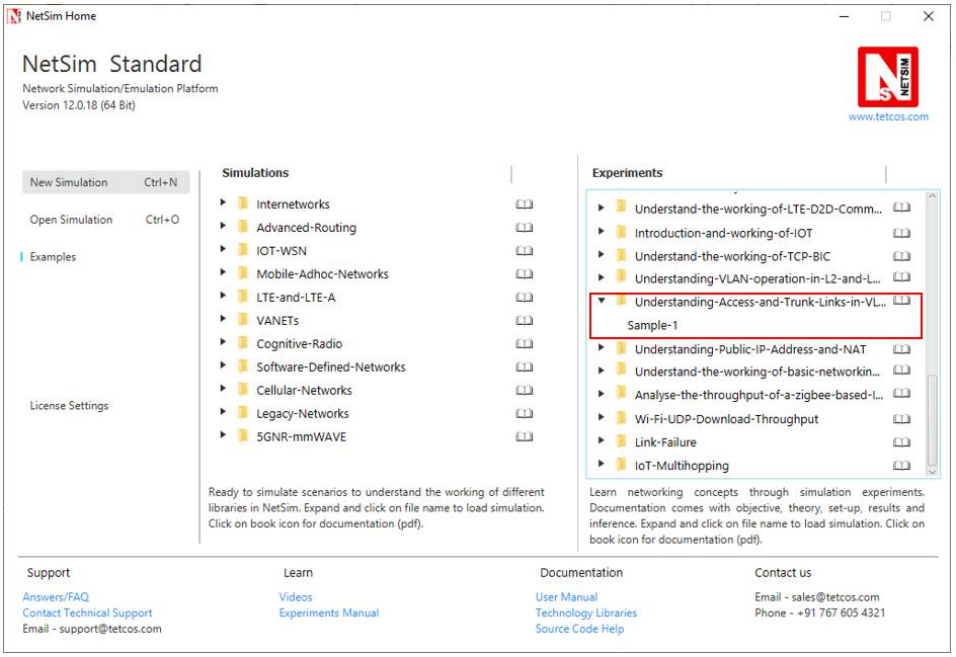
Trunk link:

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

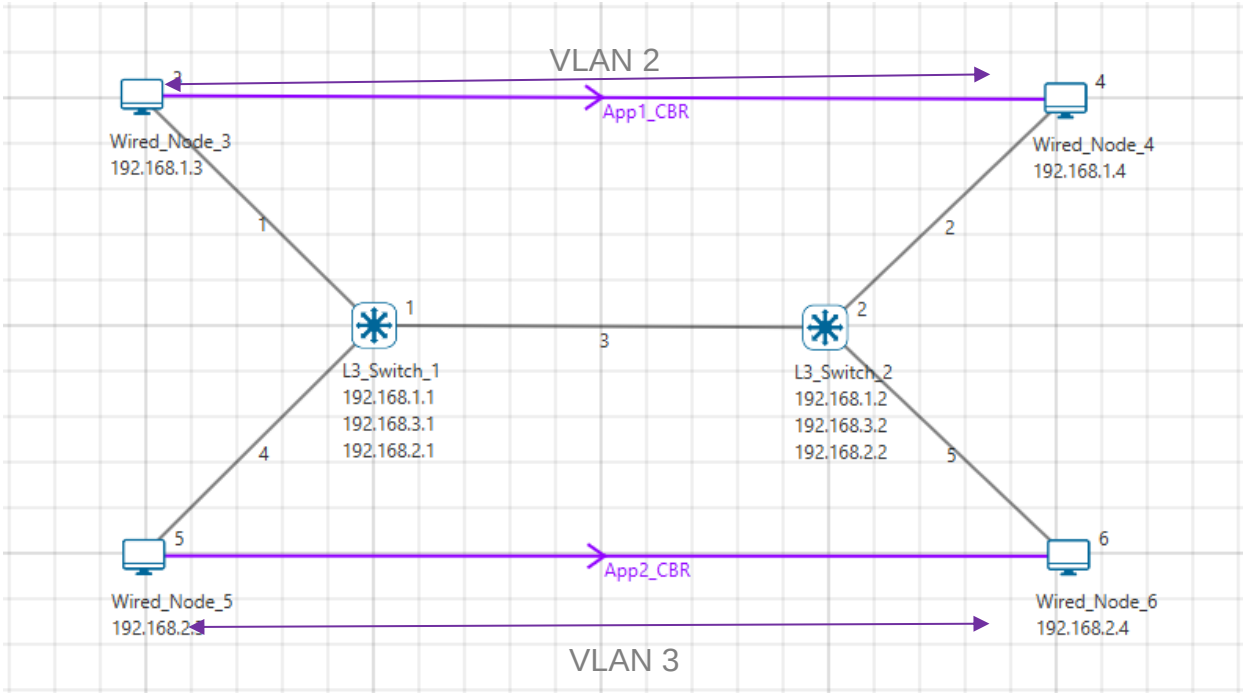
VLAN can span anywhere in network, and that can happen due to trunk link connection. Trunking allows us to send or receive VLAN information across the network. To support trunking, original Ethernet frame is modified to carry VLAN information.

28.2 Network Setup:

Open NetSim and click **Examples > Experiments > Understanding-Access-and-Trunk-Links-in-VLAN** as shown below:



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



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 4 Wired Nodes and 2 L2 Switches in the **"Internetworks"** Network Library.

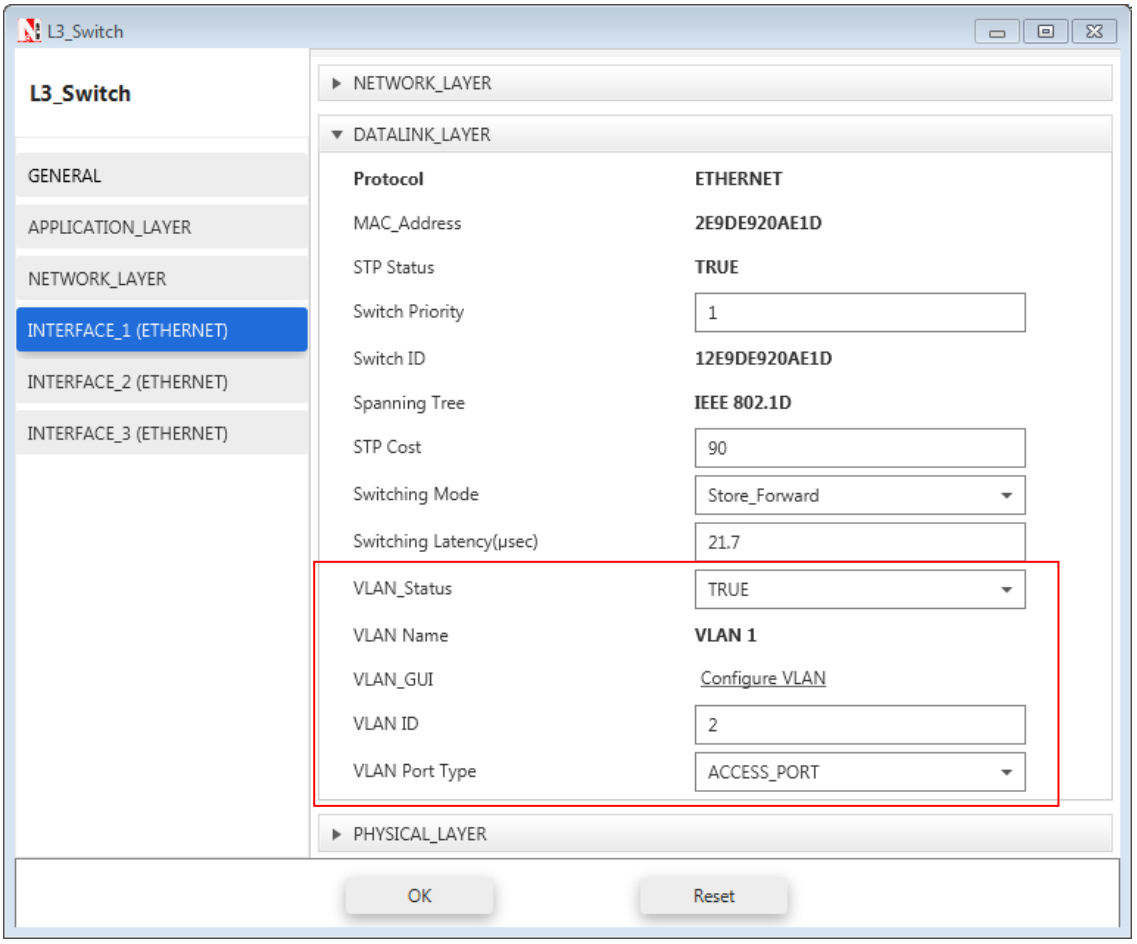
Step 2: In the INTERFACE (ETHERNET) > NETWORK LAYER Properties, set the following:

Node	Wired Node 3	Wired Node 4	Wired Node 5	Wired Node 6
	I/f1_Ethernet	I/f1_Ethernet	I/f1_Ethernet	I/f1_Ethernet
IP Address	192.168.1.3	192.168.1.4	192.168.2.3	192.168.2.4
Default Gateway	192.168.1.1	192.168.1.2	192.168.2.1	192.168.2.2
Subnet Mask	255.255.255.0	255.255.255.0	255.255.255.0	255.255.255.0

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:

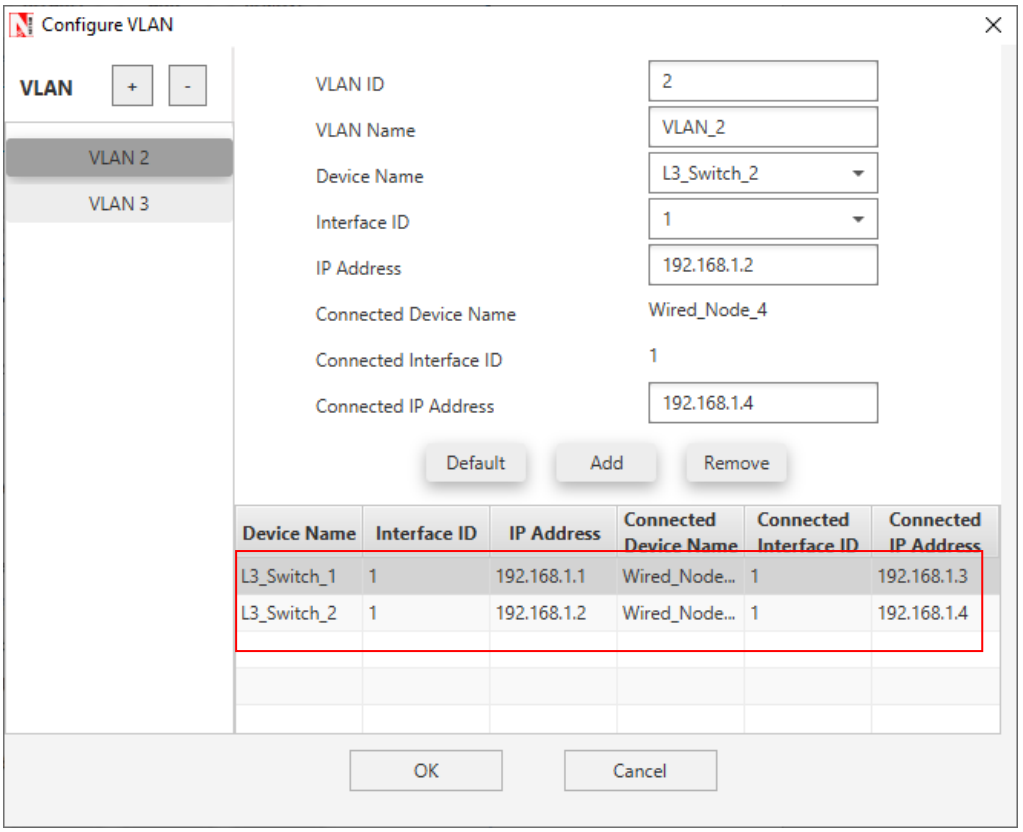
Switch	I/f1_Ethernet	I/f2_Ethernet	I/f3_Ethernet
	IP Address	IP Address	IP Address
L3 Switch 1	192.168.1.1	192.168.3.1	192.168.2.1
L3 Switch 2	192.168.1.2	192.168.3.2	192.168.2.2



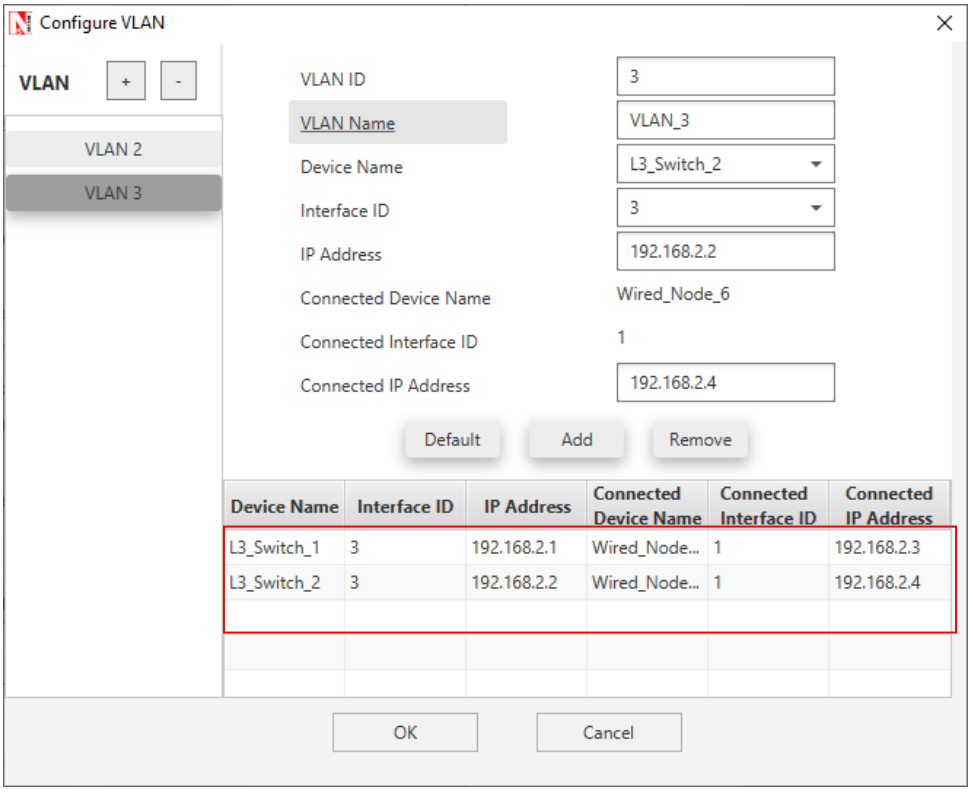
L3 Switch 1			
Interface ID	**VLAN Status**	**VLAN ID**	**VLAN Port Type**
Interface_1	TRUE	2	Access _Port
Interface_2	TRUE	1	Trunk _Port
Interface_3	TRUE	3	Access _Port

L3 Switch 2			
Interface ID	**VLAN Status**	**VLAN ID**	**VLAN Port Type**
Interface_1	TRUE	2	Access _Port
Interface_2	TRUE	1	Trunk _Port
Interface_3	TRUE	3	Access _Port

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:

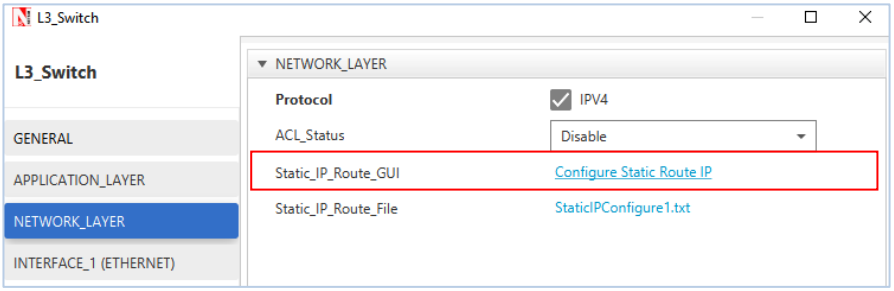


Properties for VLAN 3 is set as per the below screenshot:



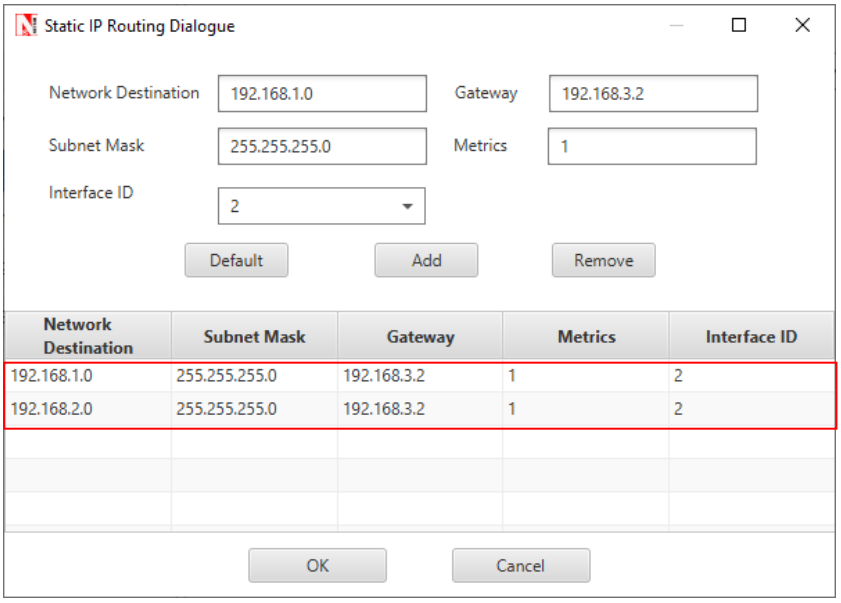
After setting the properties of VLAN2 and VLAN3 click on OK.

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:



Set the properties in Static Route IP window as per the screenshot below and click on **Add**.

Click on **OK**



***NOTE:* Disable TCP in Transport Layer in Wired Node 3 and Wired Node 5.**

Step 6: Run simulation for 10 seconds and observe the throughput.

28.4 Output:

Throughput (Mbps)	
Application 1	0.58
Application 2	0.58

The above results conclude that Trunking allows us to send or receive any VLAN information across the network.

29. Understanding Public IP Address & NAT (Network Address Translation)

29.1 Theory:

29.1.1 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.

29.1.2 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):

Class	Starting IP address	Ending IP address	No. of hosts
A	10.0.0.0	10.255.255.255	16,777,216
B	172.16.0.0	172.31.255.255	1,048,576
C	192.168.0.0	192.168.255.255	65,536

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.**

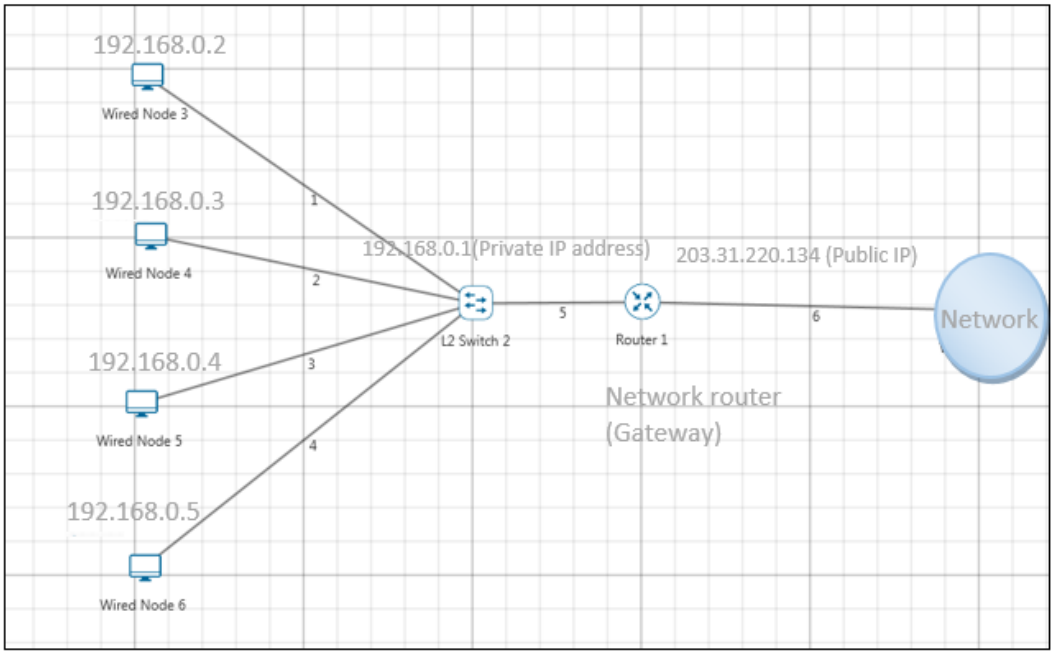
Devices with private IP addresses cannot connect directly to the Internet. Likewise, computers outside the local network cannot connect directly to a device with a private IP. It is possible to interconnect two private networks with the help of a router or a similar device that supports Network Address 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.

29.1.3 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.



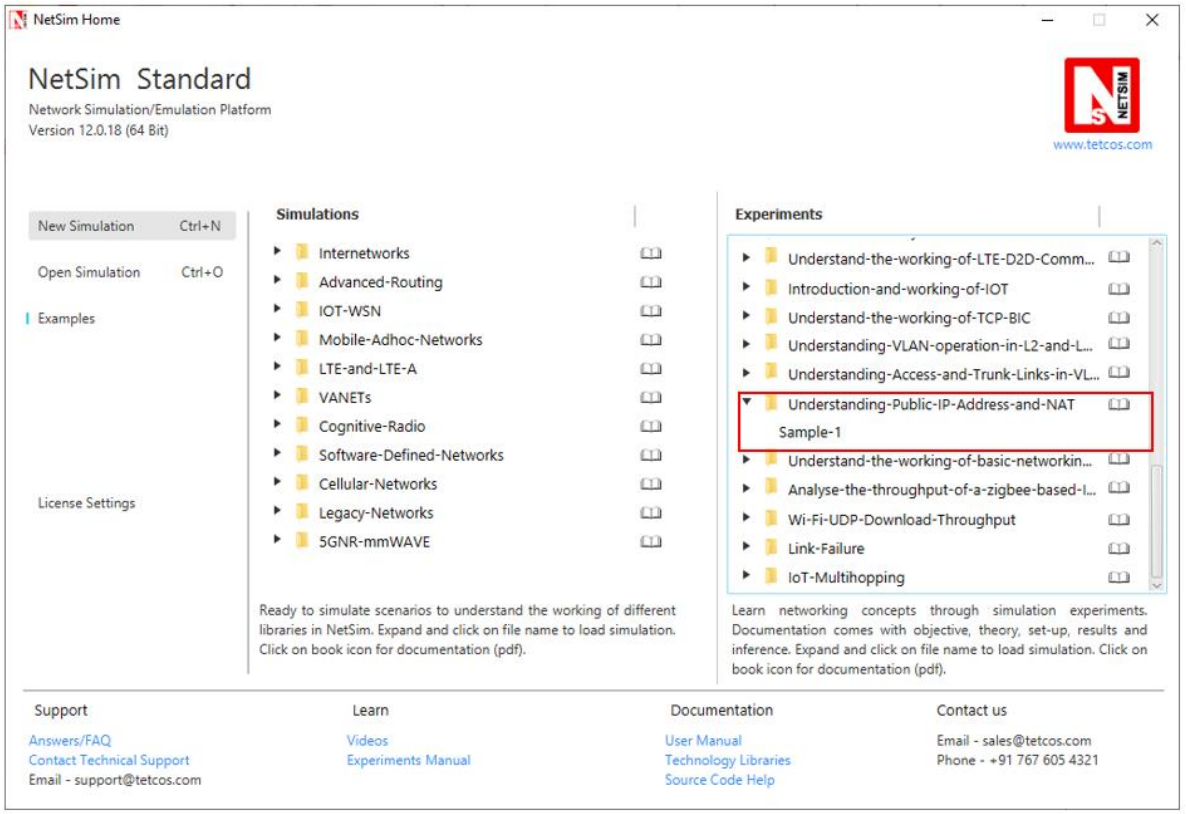
NAT is secure since it hides network from the Internet. All communications from internal private network are handled by the NAT device, which will ensure all the appropriate translations are performed and provide a flawless connection between internal devices and the Internet.

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

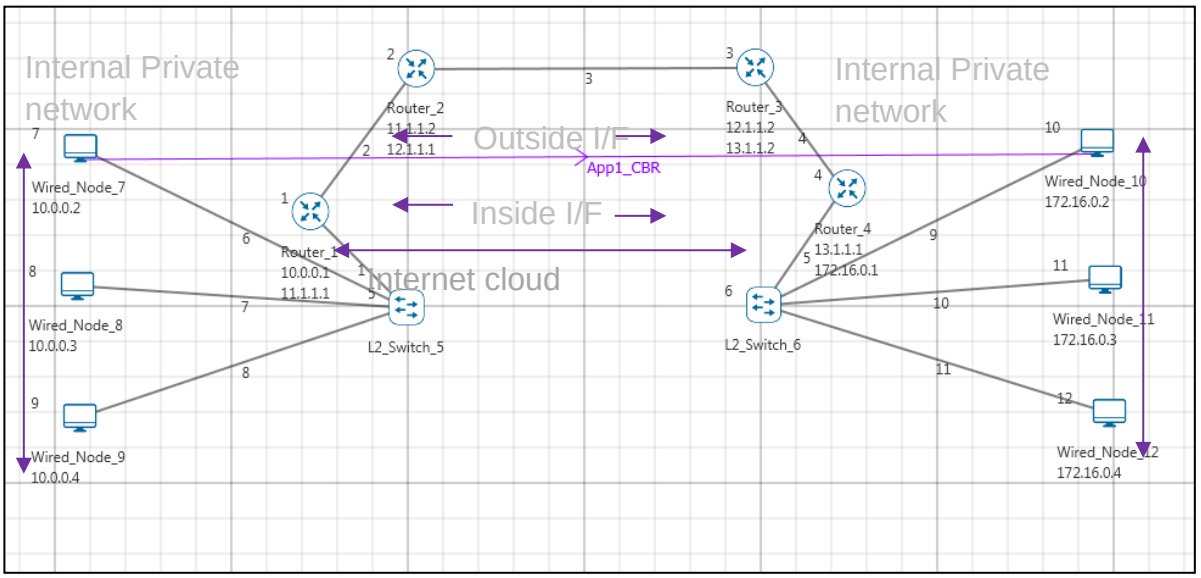
Address (203.31.220.134). This is the IP address the Internet sees as all internal IP addresses are hidden.

29.2 Network Setup:

Open NetSim and click **Examples > Experiments > Understanding-Public-IP-Address-and-NAT > Sample-1** as shown below:



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



29.3 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:

Wired Node	IP address	Subnet mask
7	10.0.0.2	255.0.0.0
8	10.0.0.3	255.0.0.0
9	10.0.0.4	255.0.0.0
10	172.16.0.2	255.255.0.0
11	172.16.0.3	255.255.0.0
12	172.16.0.4	255.255.0.0

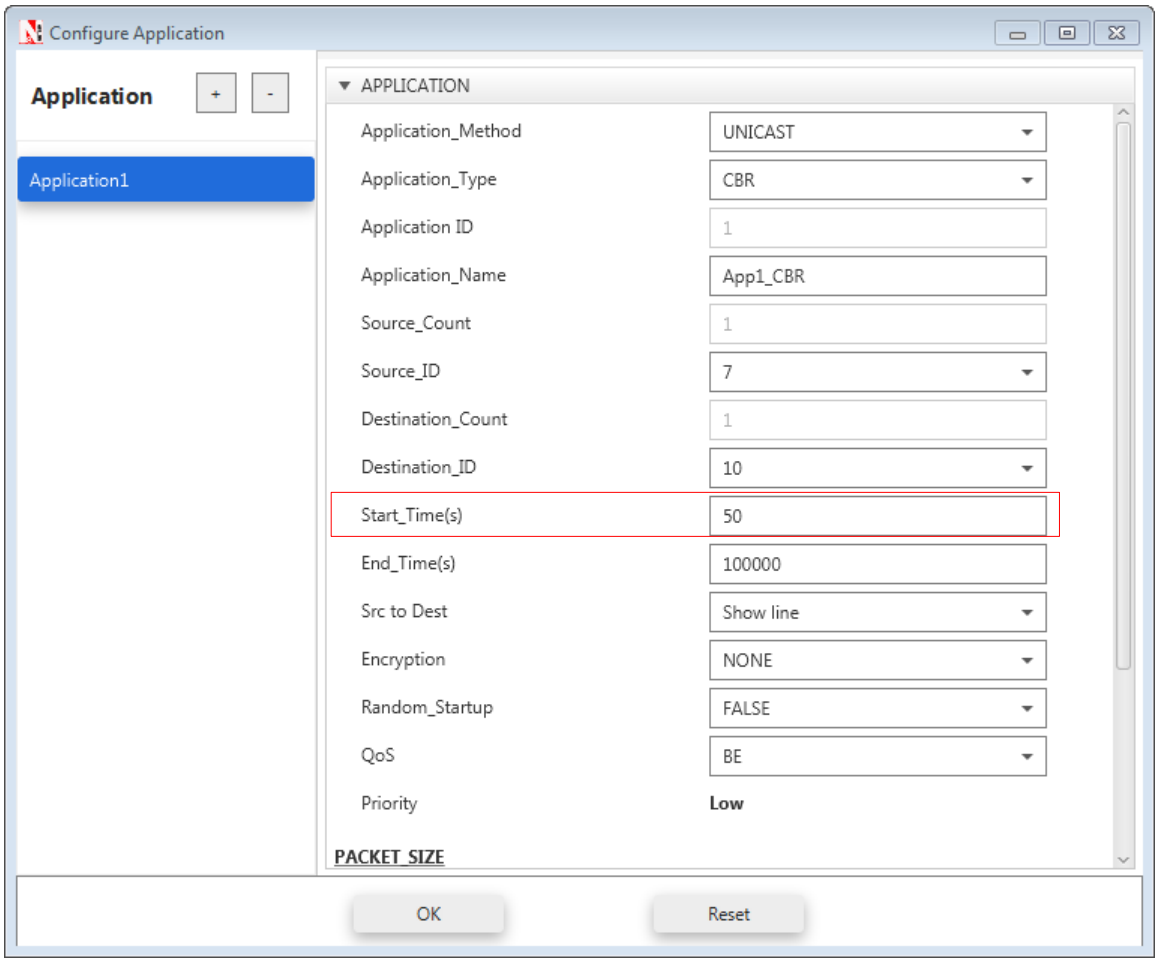
Step 3: The IP Address and the Subnet Mask in Routers are set as per the table given below:

Router	Interface	IP address	Subnet mask
Router 1	**Interface_2(WAN)**	11.1.1.1	255.0.0.0
	Interface_1(Ethernet)	10.0.0.1	255.0.0.0
Router 2	**Interface_1(WAN)**	11.1.1.2	255.0.0.0
	Interface_2(WAN)	12.1.1.1	255.0.0.0
Router 3	**Interface_1(WAN)**	12.1.1.2	255.0.0.0
	Interface_2(WAN)	13.1.1.2	255.0.0.0
Router 4	**Interface_1(WAN)**	13.1.1.1	255.0.0.0
	Interface_2(Ethernet)	172.16.0.1	255.255.0.0

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.

29.4 Output:

After simulation open Packet Trace and filter Packet ID to 1.

PACKET_I	SEGMENT	PACKET_T	CONTROL	SOURCE_(	DESTINATION_I	SOURCE_IP	DESTINATION_IP	GATEWAY_IP	NEXT_HOP_IP
1	0	CBR	App1_CBR	NODE-7	NODE-10	10.0.0.2	10.0.0.1	10.0.0.2	10.0.0.1
1	0	CBR	App1_CBR	NODE-7	NODE-10	10.0.0.2	10.0.0.1	10.0.0.2	10.0.0.1
1	0	CBR	App1_CBR	NODE-7	NODE-10	10.0.0.2	13.1.1.1	11.1.1.1	11.1.1.2
1	0	CBR	App1_CBR	NODE-7	NODE-10	10.0.0.2	13.1.1.1	12.1.1.1	12.1.1.2
1	0	CBR	App1_CBR	NODE-7	NODE-10	10.0.0.2	13.1.1.1	13.1.1.2	13.1.1.1
1	0	CBR	App1_CBR	NODE-7	NODE-10	10.0.0.2	172.16.0.2	172.16.0.1	172.16.0.2
1	0	CBR	App1_CBR	NODE-7	NODE-10	10.0.0.2	172.16.0.2	172.16.0.1	172.16.0.2

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.

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

30.1 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

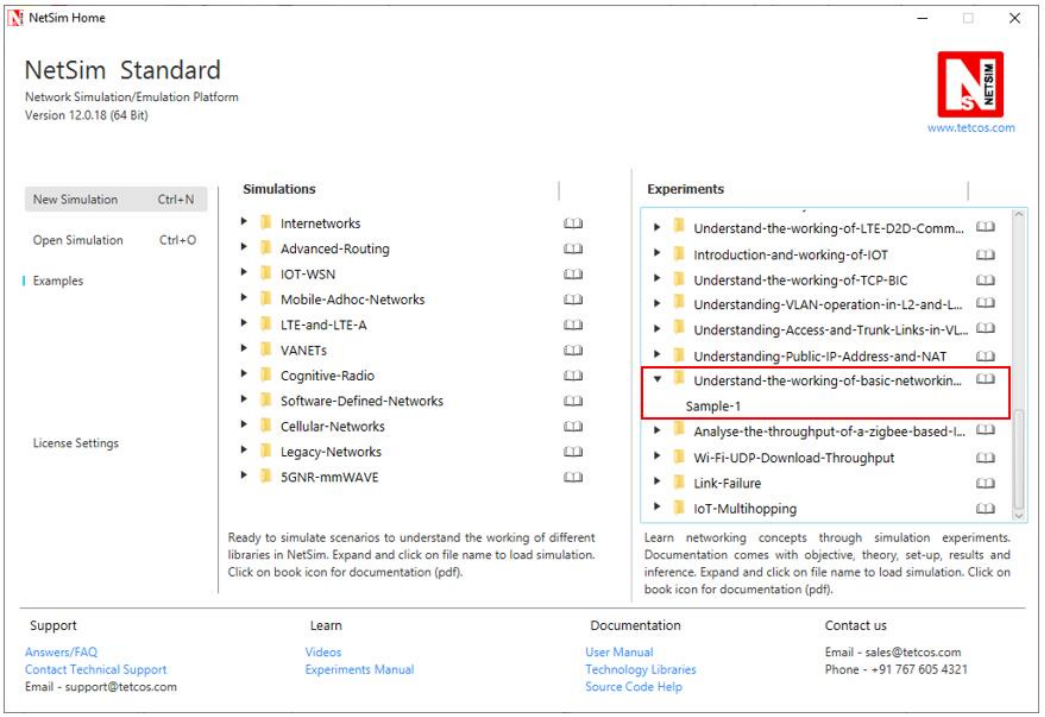
- **route print:** In order to view the entire contents of the IP routing table
- **route delete:** In order to delete all routes in the IP routing table
- **route add:** In order to add a static TCP/IP route to the IP routing table

ACL Configuration

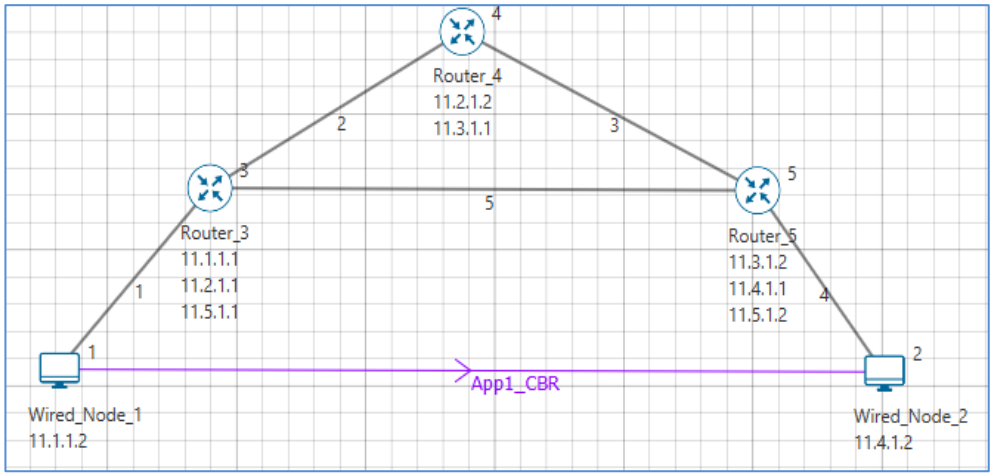
Routers provide basic traffic filtering capabilities, such as blocking the Internet traffic with access control lists (ACLs). An ACL is a sequential list of **Permit** or **Deny** statements that apply to addresses or upper-layer protocols. These lists tell the router what types of packets to: **PERMIT** 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.

30.2 Network setup:

Open NetSim and click **Examples > Experiments > Understand-the-working-of-basic-networking-commands > Sample-1** as shown below:



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



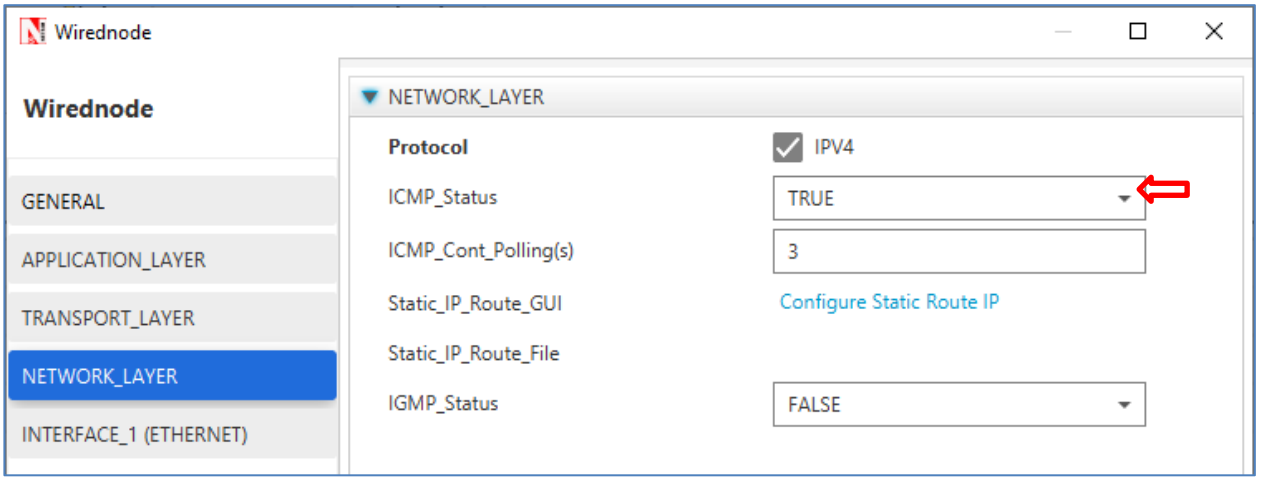
30.3 Procedure:

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

Step 1: A network scenario is designed in NetSim GUI comprising of 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: TCP Protocol in Transport Layer is disabled in all the devices.

Step 4: In the General properties of Wired Node 1, **Wireshark Capture** is set as Online.

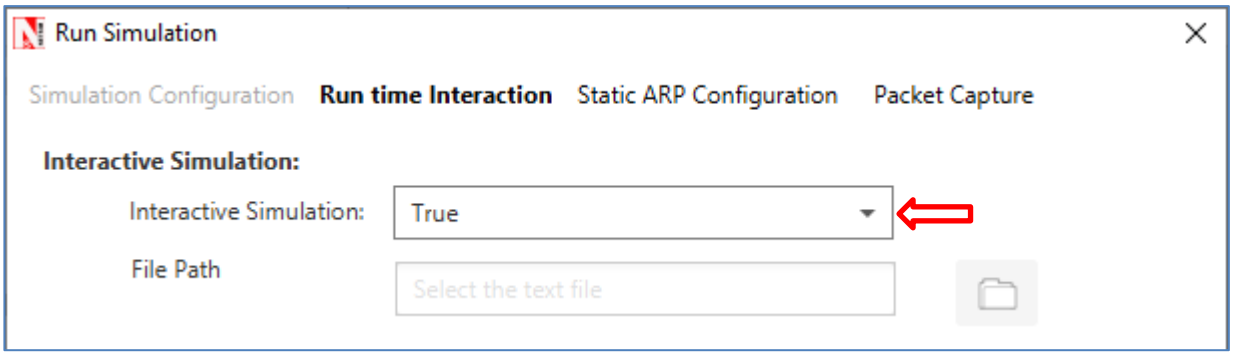
Step 5: Right click on the Application Flow **App1 CBR** and select Properties or click on the Application icon present in the top ribbon/toolbar.

A CBR Application is generated from Wired Node 1 i.e. Source to Wired Node 2 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.

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.

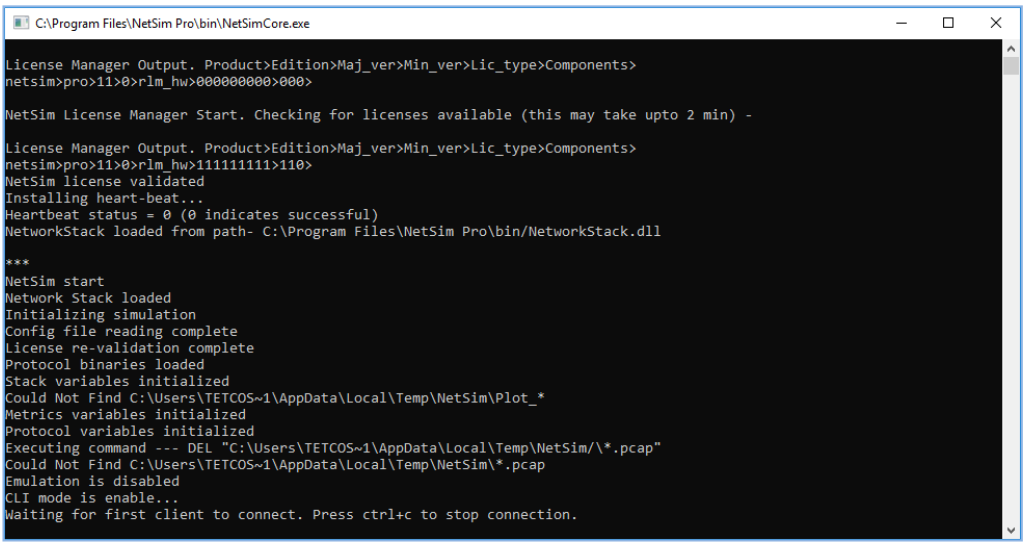
Step 7: Click on Run Simulation. Simulation Time is set to 300 Seconds and in the **Runtime Interaction** tab, Interactive Simulation is set to True.



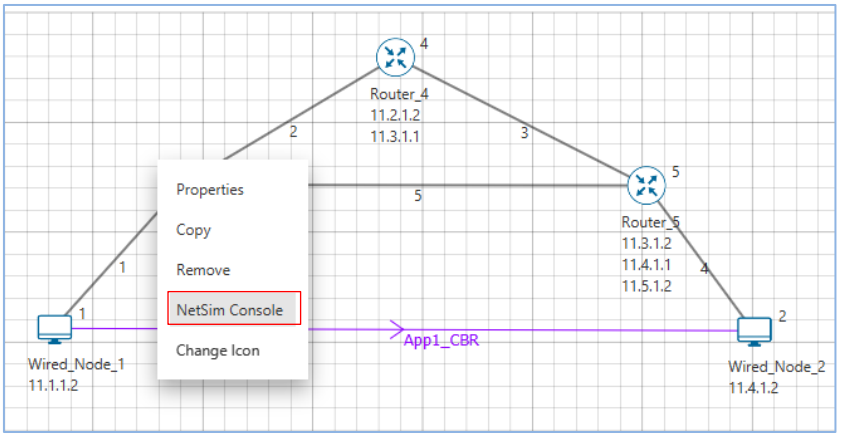
***NOTE:* It is recommended to specify a longer simulation time to ensure that there is sufficient time for the user to execute the various commands and see the effect of that before the Simulation ends.**

Click on **Accept** and then click on **OK**.

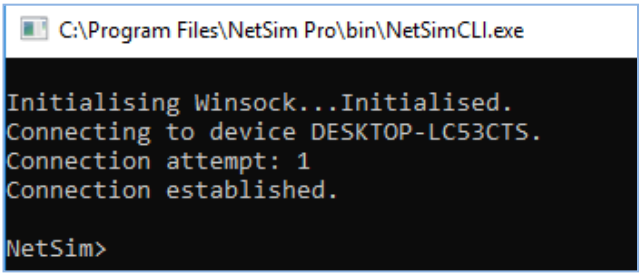
- Simulation (NetSimCore.exe) will start running and will display a message **"waiting for first client to connect"** as shown below:



- 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.



- 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:



- After this the command line interface can be used to execute all the supported commands.

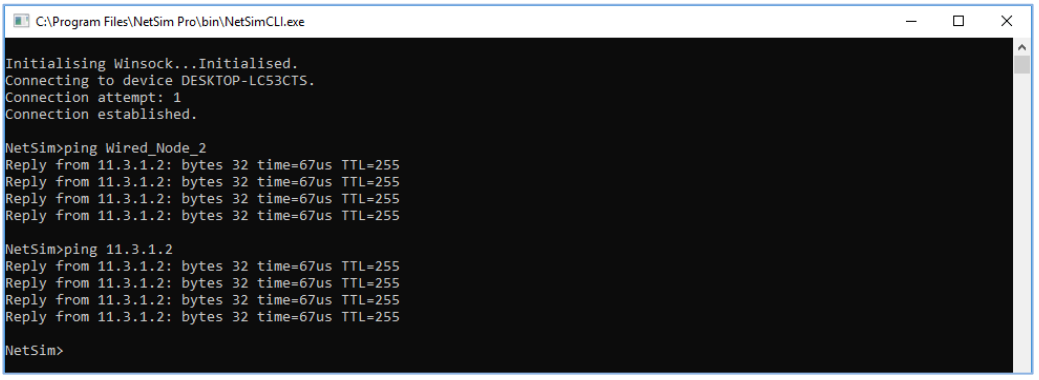
30.4 Network Commands

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

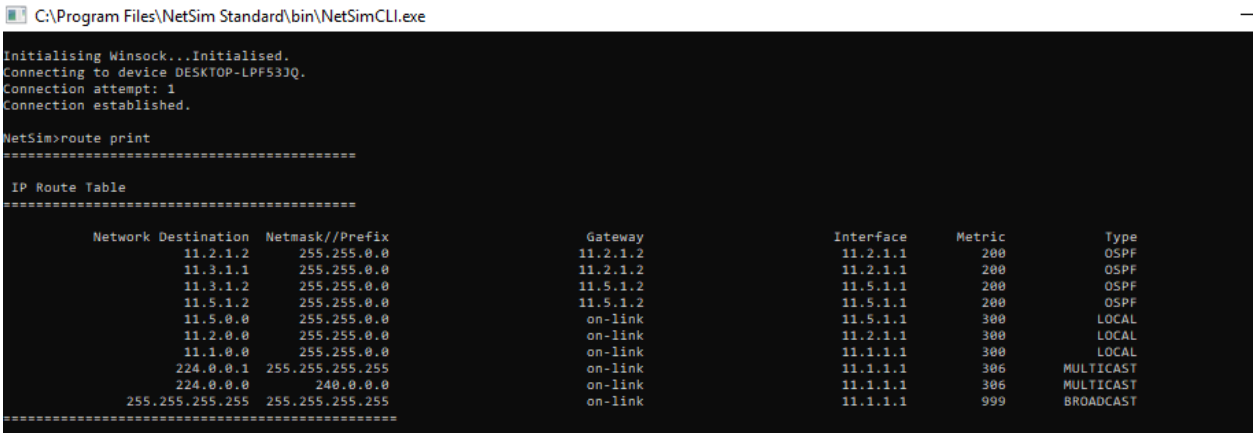
Ping <Node Name> e.g. ping Wired_Node_2



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

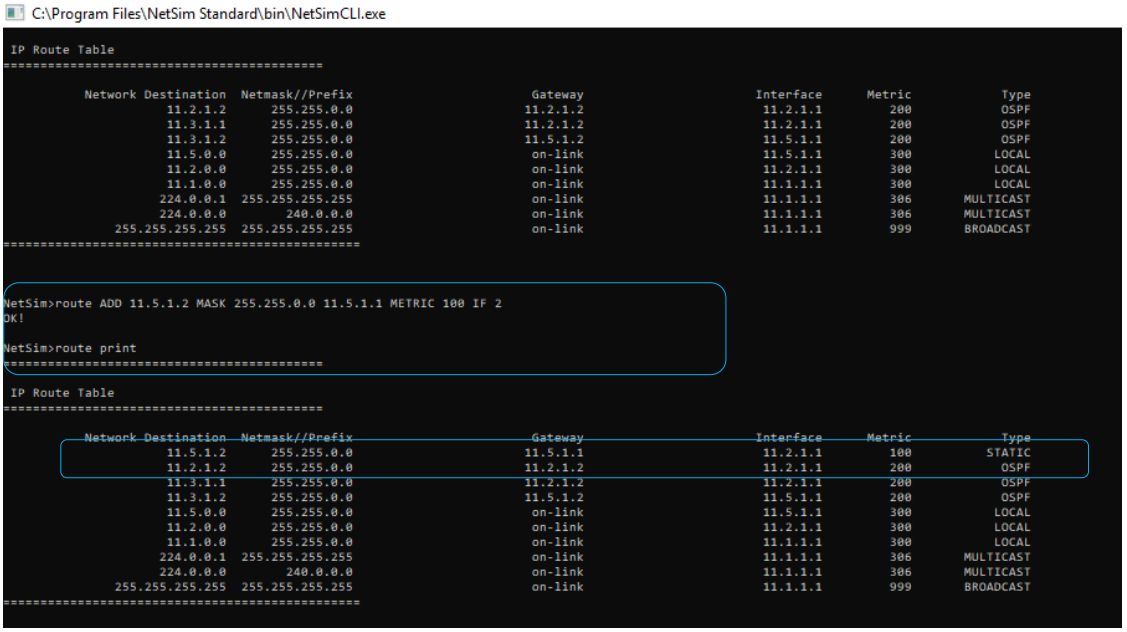
- To check whether route has been deleted or not check again using **route print** command
- To add a static route to the table, you'll type a command using the following syntax

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.
- TCP Protocol in Transport Layer is disabled in all the devices.
- 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

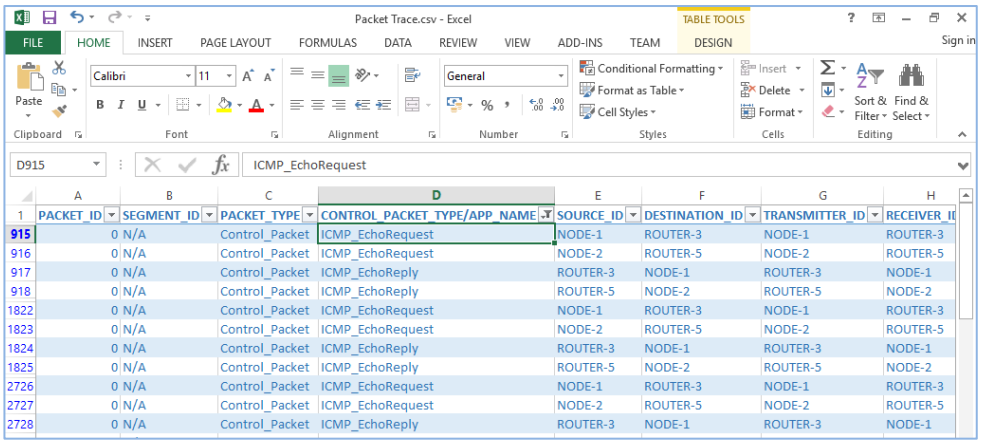
NetSim>

```
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:

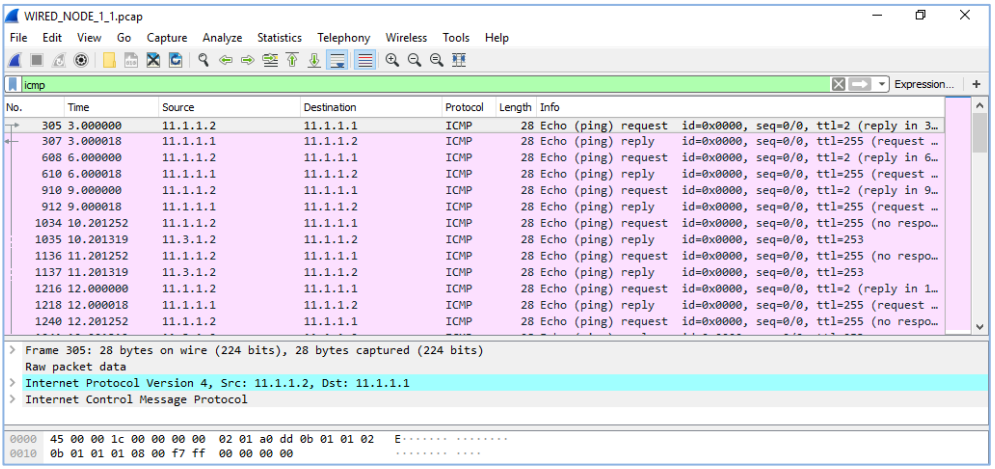
Go to the Results Dashboard and click on **"Open Packet Trace"** option present in the Left-Hand-Side of the window and do the following:

Filter Control Packet Type/App Name to **ICMP EchoRequest** and **ICMP EchoReply**.



	A	B	C	D	E	F	G	H
1	PACKET_ID	SEGMENT_ID	PACKET_TYPE	CONTROL_PACKET_TYPE/APP_NAME	SOURCE_ID	DESTINATION_ID	TRANSMITTER_ID	RECEIVER_ID
915	0	N/A	Control_Packet	ICMP_EchoRequest	NODE-1	ROUTER-3	NODE-1	ROUTER-3
916	0	N/A	Control_Packet	ICMP_EchoRequest	NODE-2	ROUTER-5	NODE-2	ROUTER-5
917	0	N/A	Control_Packet	ICMP_EchoReply	ROUTER-3	NODE-1	ROUTER-3	NODE-1
918	0	N/A	Control_Packet	ICMP_EchoReply	ROUTER-5	NODE-2	ROUTER-5	NODE-2
1822	0	N/A	Control_Packet	ICMP_EchoRequest	NODE-1	ROUTER-3	NODE-1	ROUTER-3
1823	0	N/A	Control_Packet	ICMP_EchoRequest	NODE-2	ROUTER-5	NODE-2	ROUTER-5
1824	0	N/A	Control_Packet	ICMP_EchoReply	ROUTER-3	NODE-1	ROUTER-3	NODE-1
1825	0	N/A	Control_Packet	ICMP_EchoReply	ROUTER-5	NODE-2	ROUTER-5	NODE-2
2726	0	N/A	Control_Packet	ICMP_EchoRequest	NODE-1	ROUTER-3	NODE-1	ROUTER-3
2727	0	N/A	Control_Packet	ICMP_EchoRequest	NODE-2	ROUTER-5	NODE-2	ROUTER-5
2728	0	N/A	Control_Packet	ICMP_EchoReply	ROUTER-3	NODE-1	ROUTER-3	NODE-1

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:

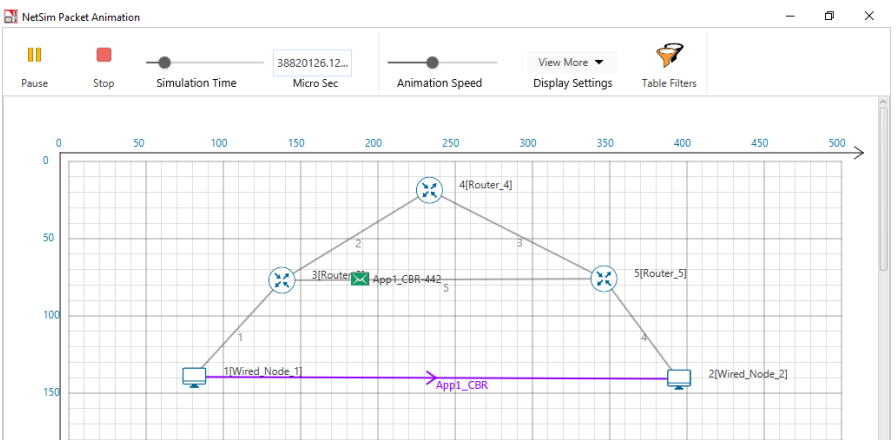
IP_Metrics_Table

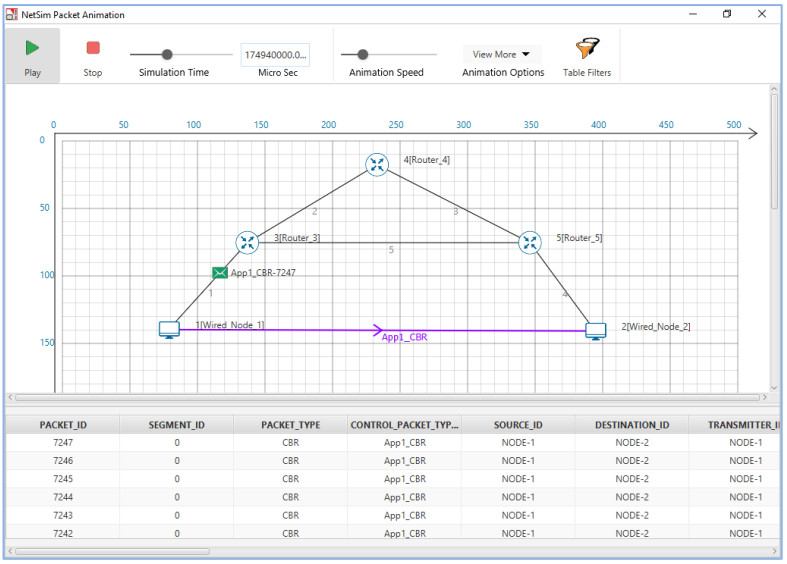
IP_Metrics Detailed View

Device Id	Packet sent	Packet forwarded	Packet received	Packet discarded	TTL expired	Firewall blocked
1	13599	0	0	0	0	0
2	99	0	8419	0	0	0
3	8609	13482	72	0	0	5047
4	74	0	74	0	0	0
5	8605	8435	74	0	0	0

***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

31. Analyze the throughput of an IOT network where a wireless (802.15.4) sensor transmits data to a cloud server

31.1 Introduction:

IEEE Standard 802.15.4 defines the protocol and compatible interconnections for data communication devices using low-data-rate, low-power, and low-complexity short-range radio frequency (RF) transmissions in a wireless personal area network (WPAN). In IOT networks IEEE 802.15.4 standard is commonly used in MAC and PHY layers.

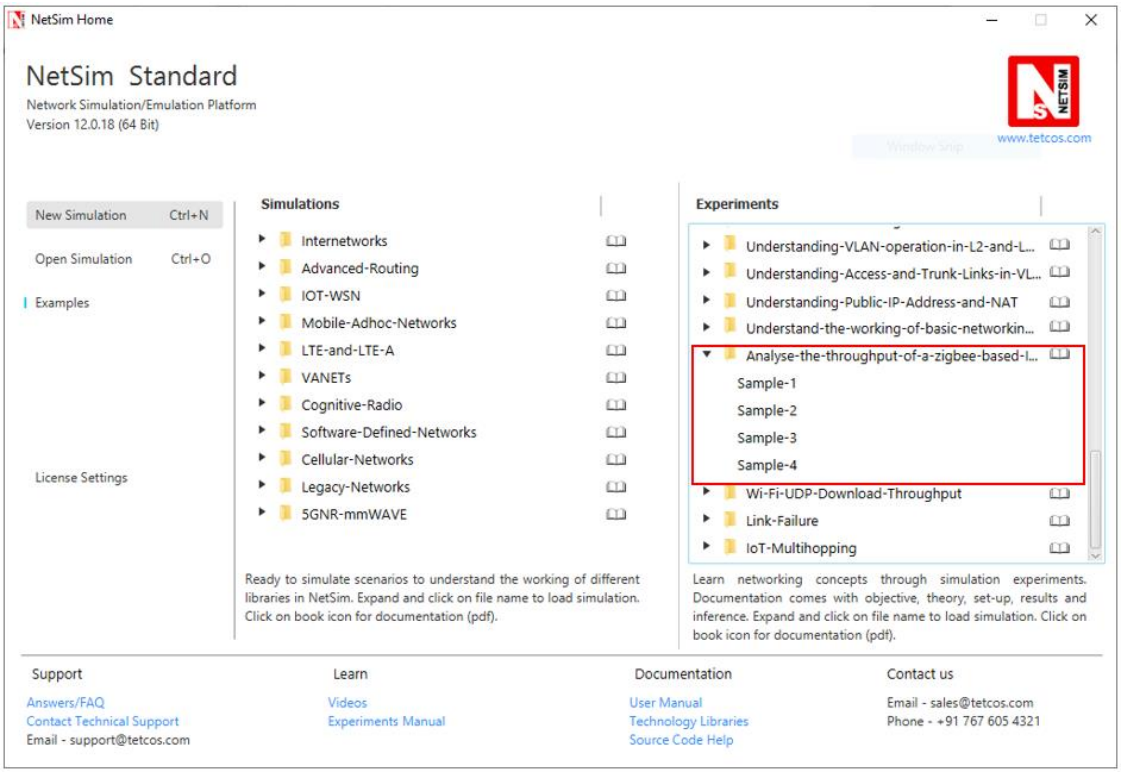
IEEE 802.15.4 PHYs provide the capability to perform CCA in its CSMA-CA mechanism. The PHYs require at least one of the following three CCA methods: Energy Detection over a certain threshold, detection of a signal with IEEE 802.15.4 characteristics or a combination of these methods.

31.2 Theory:

- A packet transmission begins with a random backoff (in number of slots, each slot of 20 T_s duration) which is sampled uniformly from 0 to $2^{macminBE} - 1$ and followed by a CCA.
- CCA failure starts a new backoff process with the backoff exponent raised by one, i.e., to macminBE+1, provided it is lesser than the maximum backoff value given by macmaxBE.
- Maximum number of successive CCA failures for the same packet is governed by macMaxCSMABackoffs, exceeding which the packet is discarded at the MAC layer.
- A successful CCA is followed by the radio turnaround time and packet transmission.
- If the receiver successfully receives the packet i.e., without any collision or corruption due to PHY layer noise, the receiver sends an ACK after the radio turnaround time.
- A failed packet reception causes no ACK generation.
- The transmitter infers that the packet has failed after waiting for macAckWaitDuration and retransmits the packet for a maximum of aMaxFrameRetries times before discarding it at the MAC layer.
- Note that in NetSim the radio turnaround time after a CCA success is not considered.

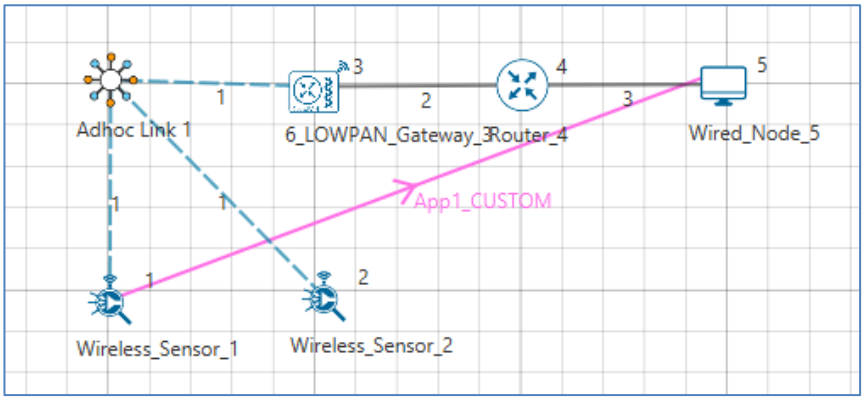
31.3 Network Setup:

Open NetSim and click **Examples > Experiments > Analyse-the-throughput-of-zigbee-based-IoT** as shown below:



Sample 1:

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



31.4 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 Wireless Sensors, a 6 LOWPAN Gateway, 1 Router, and 1 Wired Node.

Step 2: In the Destination Node, i.e. Wired Node 5, TCP Protocol in the Transport Layer is disabled. Similarly, TCP is disabled in Router.

Note: *By default, TCP is disabled in the 6 LOWPAN Gateway and Wireless Sensors.*

Step 3: 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 and 250 meters respectively, instead of the default 100 and 50 meters and we have chosen **Manually Via Click and Drop** option.

Step 4: 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 5: In the Interface Zigbee > Data Link Layer of 6 LOWPAN Gateway, **Beacon Mode** is set to Disable by default.

Step 6: The Ad hoc link properties are set to **NO PATHLOSS** for the channel characteristics.

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 1 i.e. Source to Wired Node 5 i.e. Destination 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:

$$Generation\ Rate\ (Mbps)\ =\ Packet\ Size\ (Bytes)\ *\ 8/Interarrival\ time\ (\mu s)$$

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 8: 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.

NOTE: ***This throughput is obtained when the nodes are placed at the closest possible distance from the PAN coordinator. As the distance is increased, the throughput would reduce.***

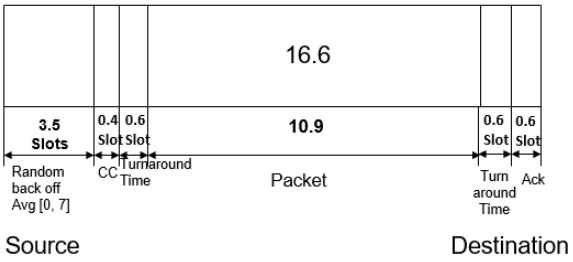
Theoretical Analysis:

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:

App layer Payload	70 bytes
Transport Layer Header	8 bytes

Network Layer Header	**20 bytes**
MAC Header	**5 bytes**
PHY Header	**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.

Symbol Time	T_s	16 µs
Slot Time	20 * T_s	**0.32 ms**
Random Backoff Average	3.5 * $Slots$	**1.12 ms**
CCA	0.4 * $Slots$	**0.128 ms**
Turn-around-Time	0.6 * $Slots$	**0.192 ms**
Packet Transmission Time	10.9 * $Slots$	**3.488 ms**
Turn-around-Time	0.6 * $Slots$	**0.192 ms**
ACK Packet Time	0.6 * $Slots$	**0.192 ms**
Total Time	**16.6 * $Slots$**	**5.312 ms**

$$Application\ Throughput\ = \frac{70(bytes)inApplayer * 8}{5.312\ ms} = 105.42\ kbps$$

31.5 Inference:

Throughput from simulation	**104.74 kbps**
Throughput from analysis	**105.42 kbps**

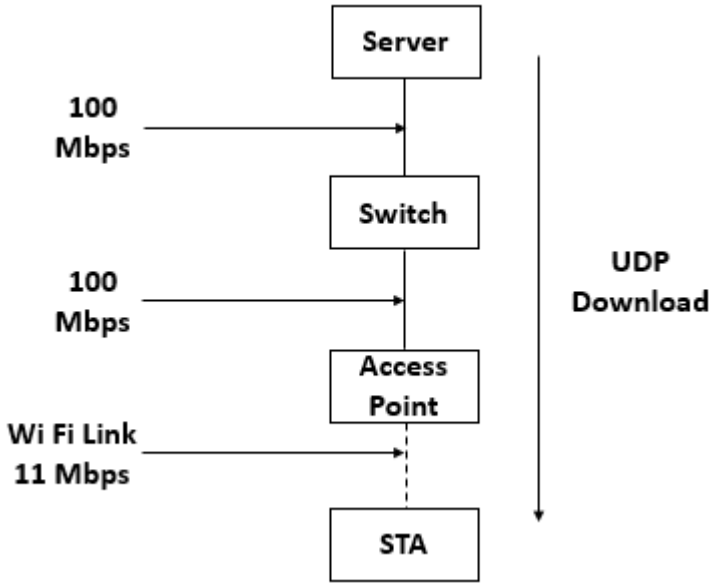
Throughput from theoretical analysis matches the results of NetSim's discrete event simulation. The slight difference in throughput is due to fact that the average of random numbers generated for backoff need not be exactly 3.5 as the simulation is run for short time. Furthermore, in the Network layer AODV protocol is running, so route setup process will take some time. 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:

Sample	Simulation Time (sec)	Throughput (kbps)
1	10	103.65
2	50	104.67
3	100	104.64
4	200	104.74

32. WiFi: UDP Download Throughput

32.1 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, a 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.

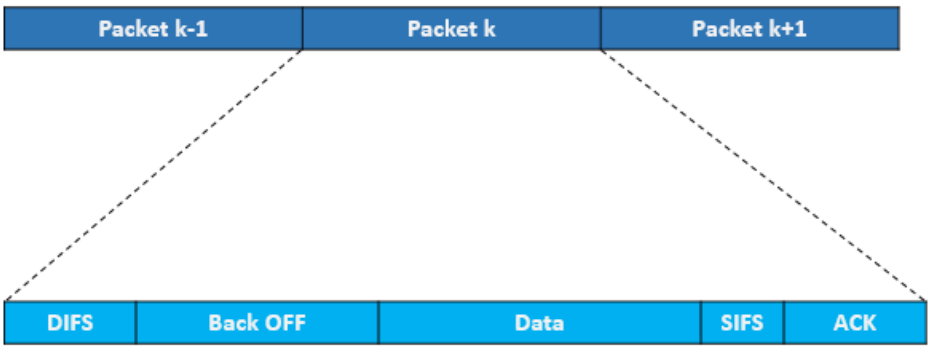
In this setting, this experiment will ask one precise question. With the buffers at the AP full, at what rate will the WiFi protocol transfer the packets from the APs to the STAs over the wireless link. We will study two cases:

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

32.2 Predicting the UDP Throughput

32.2.1 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.



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.

32.2.1.1 Without RTS / CTS

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

$$Average\ time\ per\ packet\ \ (\mu s) = DIFS + Average\ Backoff\ time + Packet\ Transmission\ Time + SIFS + Ack\ Transmission\ Time$$

$$Packet\ Transmission\ Time\ (\mu s) = Preamble\ time + (MPDU\ Size/Data\ rate)$$

$$Average\ Backoff\ time\ (\mu s) = (CWmin/2) * Slot\ Time$$

$$Ack\ Transmission\ Time\ (\mu s) = Preamble\ time + (Ack\ Packet\ size/Ack\ data\ rate)$$

$$DIFS\ (\mu s) = SIFS + 2 * Slot\ Time$$

$$Average\ Backoff\ time\ (\mu s) = (CWmin/2) * Slot\ Time$$

where

Application payload = 1450 Bytes

Average time per packet = 50 + 310 + 1296 + 10 + 304 = 1970 µs

SIFS = 10 µs

Slot time = 20 µs

CWmin = 31 slots for 802.11b

DIFS = SIFS + 2 * Slot Time = 10 µs + 2 * 20 µs = 50 µs

Average Backoff time = 310 µs

Packet Transmission Time = 192 µs + (1518 * 8/11 Mbps) = 1296 µs

Preamble time = 192 µs for 802.11b standard

MPDU Size = 1450 + 8 + 20 + 40 = 1518 Bytes

Ack Transmission Time = 192 µs + (14 Bytes * 8 / 1Mbps) = 304 µs

UDP throughput = 1450*8/ (1970) = 5.92 Mbps

32.2.1.2 With RTS/CTS

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

$$Average\ time\ per\ packet\ \ (\mu s) = DIFS + RTS\ Packet\ Transmission\ Time + CTS\ Packet\ Transmission\ Time + Average\ Backoff\ time + Packet\ Transmission\ Time + SIFS + Ack\ Transmission\ Time$$

RTS packet transmission time = Preamble time + (RTS Packet payload/Data rate) = 192 + 20 * 8 / 1 = 352 µs

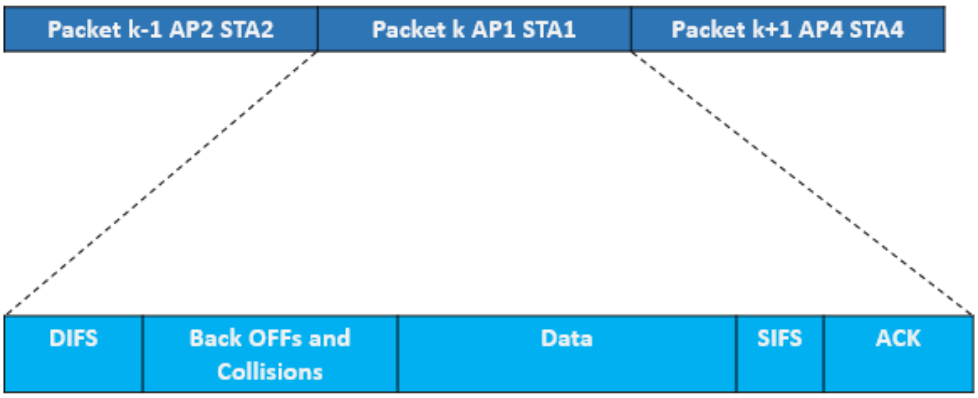
CTS packet transmission time = Preamble time + (CTS Packet payload/Data rate) = 192 + 14 * 8 / 1 = 304 µs

Average time per packet = 50 + 352 + 304 + 310 + 1296 + 10 + 304 = 2626 µs

UDP throughput = 1450*8/ (2626) = 4.44 Mbps

32.2.2 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).



a. Within each transmission period, there is now a “backoffs and collisions” period, where in the figure above we only showed a “backoff” period. Access to the channel is be contention, collision, and backoff, and this “backoffs and collisions” duration is the time taken to select one transmitting AP.
b. The other difference is that, after each “backoffs and collisions” period, any one AP-STA pair “wins” the contention, and the corresponding AP can then send a packet. It turns out that the contention mechanism is such that each of the AP-STA pairs can succeed with equal probability, independent of the pair that has previously been successful. Thus, if there are, say, 5 AP-STA pairs, then each successful packet transmission will be from any of these pairs with a probability of 0.2.

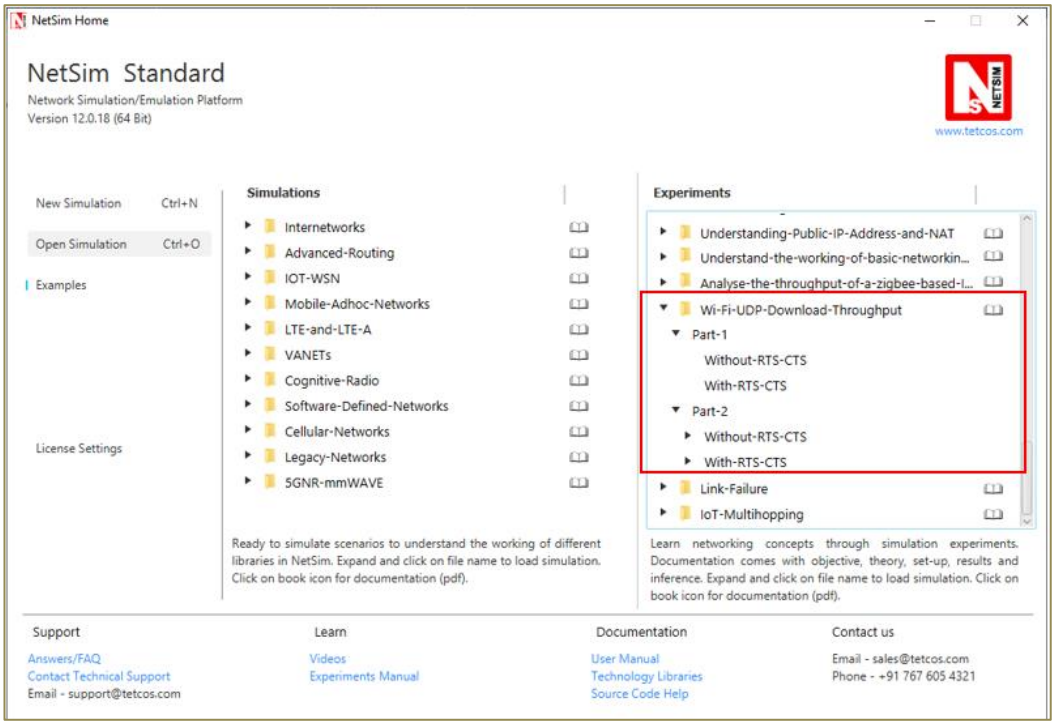
With this discussion, and the upper part of the figure above, it follows that the following expression still holds

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

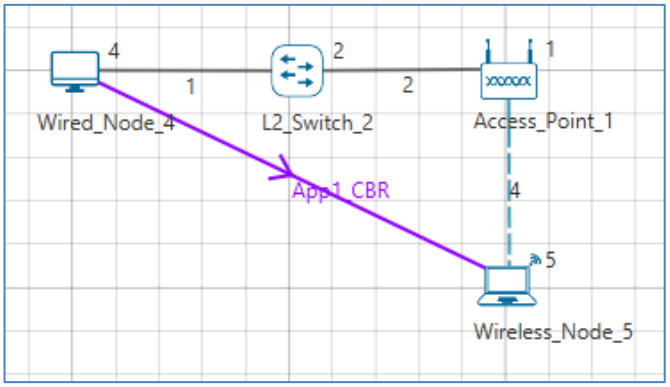
Having obtained the total throughput over all the AP-STA pairs in this manner, by the fact that each packet transmission is with equal probability from any of the AP-STA pairs, the UDP throughput for each AP-STA pair (for N pairs) is just $\frac{1}{N}$ of the total throughput.

32.3 Network Setup:

Open NetSim and click **Examples > Experiments > Wi-Fi-UDP-Download-Throughput > Part-1 > Without-RTS-CTS** as shown below:



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



32.4 Procedure:

32.4.1 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: TCP Protocol is disabled in Wired Node 4.

Step 3: 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 4: In the Wired Link Properties, Bit Error Rate and Propagation Delay is set to 0 for both the links.

Step 5: In the Wireless Link Properties, Channel Characteristics is set to NO PATH LOSS.

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 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.

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:

$$Generation\ Rate\ (Mbps)\ =\ Packet\ Size\ (Bytes)\ *\ 8/Interarrival\ time\ (\mu s)$$

Step 7: Run the Simulation for 10 Seconds and note down the throughput.

32.4.2 Part-1: With RTS/CTS

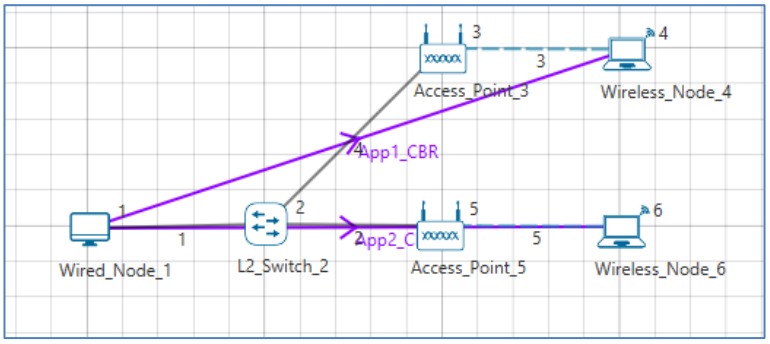
The following changes in settings are done from the previous sample:

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.

32.4.3 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.

32.4.4 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.

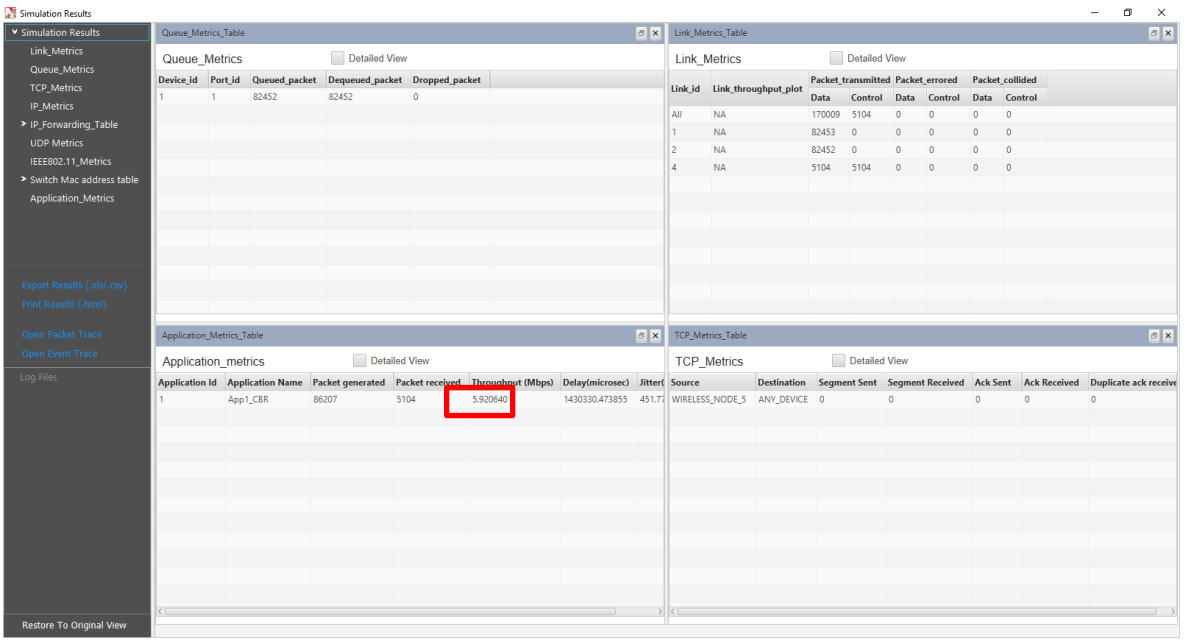
It will automatically be set for Wireless Node 6, since the above parameter is Global.

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.

32.5 Output I:

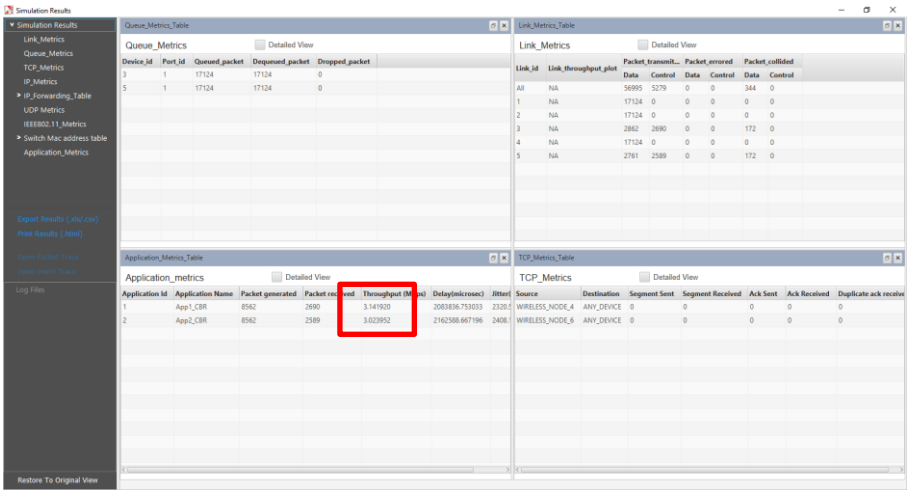
After running simulation, check throughput in Application metrics as shown in the below screenshot:



Sample	Predicted throughput (Mbps)	Simulated Throughput (Mbps)
1 (Without RTS/CTS)	5.92	5.92
2 (With RTS/CTS)	4.44	4.39

32.6 Output II:

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



Sample	Throughput (Mbps) with 2 APs	Throughput (Mbps) with 3 APs	Throughput (Mbps) with 4 APs	Throughput (Mbps) with 5 APs
1 (Without RTS/CTS)	App 1: 3.14 App 2: 3.02 **Total: 6.16**	App 1: 2.12 App 2: 1.97 App 3: 2.09 **Total: 6.18**	App 1:1.58 App 2:1.57 App 3:1.52 App 4:1.46 **Total: 6.13**	App 1: 1.25 App 2: 1.20 App 3: 1.30 App 4: 1.20 App 5: 1.12 **Total: 6.07**
2 (With RTS/CTS)	App 1: 2.34 App 2: 2.25 **Total: 4.59**	App 1: 1.58 App 2: 1.49 App 3: 1.58 **Total: 4.65**	App 1: 1.21 App 2: 1.18 App 3: 1.15 App 4: 1.12 **Total: 4.66**	App 1: 0.95 App 2: 0.90 App 3: 1.00 App 4: 0.92 App 5: 0.89 **Total: 4.66**

33. Simulating Link Failure

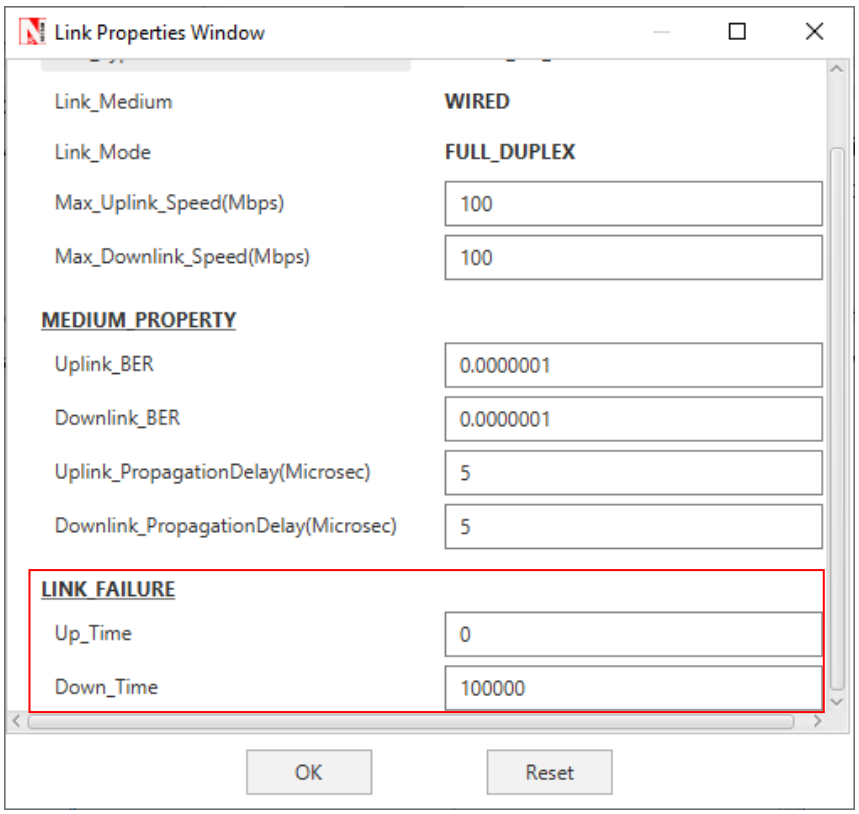
33.1 Objective

To understand the working of Link Failure.

33.2 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:

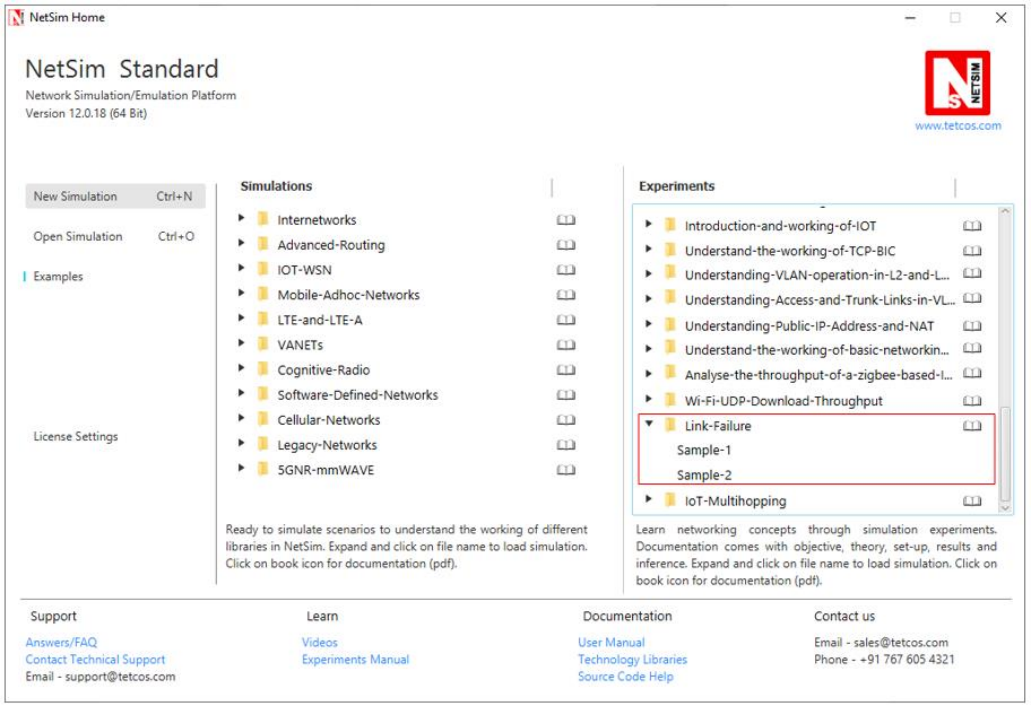


Link Up Time refers to the time at which the link goes up and Link Down Time refers to the time at which a link goes down.

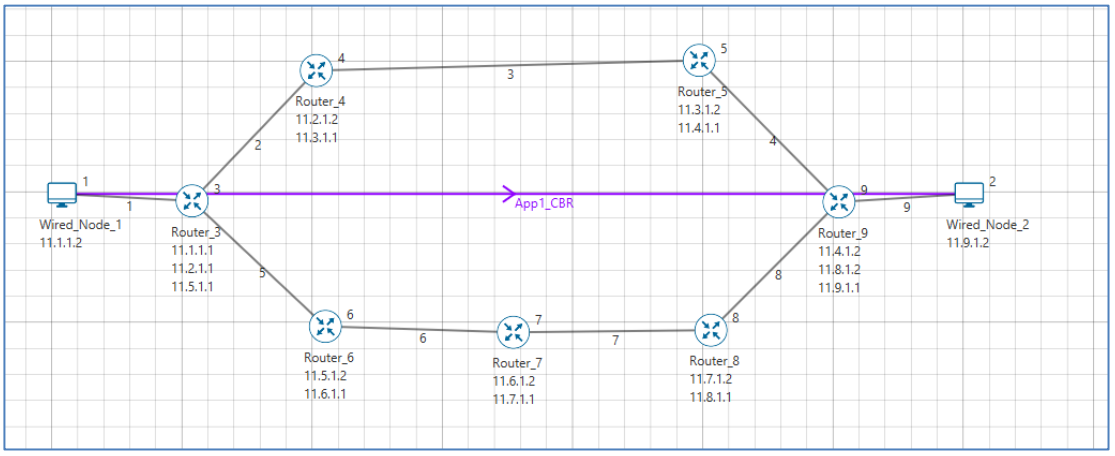
***NOTE:** Link failure can be set only for "WAN Interfaces".*

33.3 Network Setup:

Open NetSim and click **Examples > Experiments > Link-Failure > Sample-1** as shown below:



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



33.4 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.

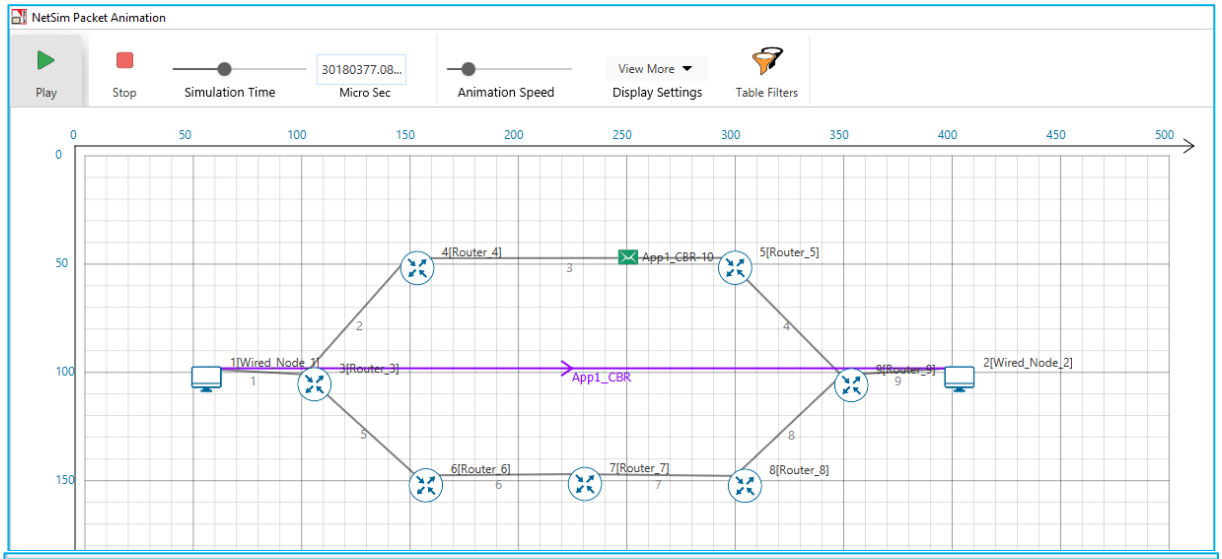
> Sample-2:
The following changes in settings are done from the previous sample:
Step 1: In Link 3 Properties, Link Failure **Up Time** is set to 0 and **Down Time** is set to 50.
This means that the link would fail at 50 Seconds.
Step 2: Run the simulation for 80 Seconds.

33.5 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:



PACKET_ID	SEGMENT_ID	PACKET_TYPE	CONTROL_PACKET_TYPE/APP_NAME	SOURCE_ID	DESTINATION_ID	TRANSMITTER_ID	RECEIVER_ID
8	0	CBR	App1_CBR	NODE-1	NODE-2	ROUTER-9	NODE-2
8	0	CBR	App1_CBR	NODE-1	NODE-2	ROUTER-5	ROUTER-9
8	0	CBR	App1_CBR	NODE-1	NODE-2	ROUTER-4	ROUTER-5
8	0	CBR	App1_CBR	NODE-1	NODE-2	ROUTER-3	ROUTER-4
8	0	CBR	App1_CBR	NODE-1	NODE-2	NODE-1	ROUTER-3

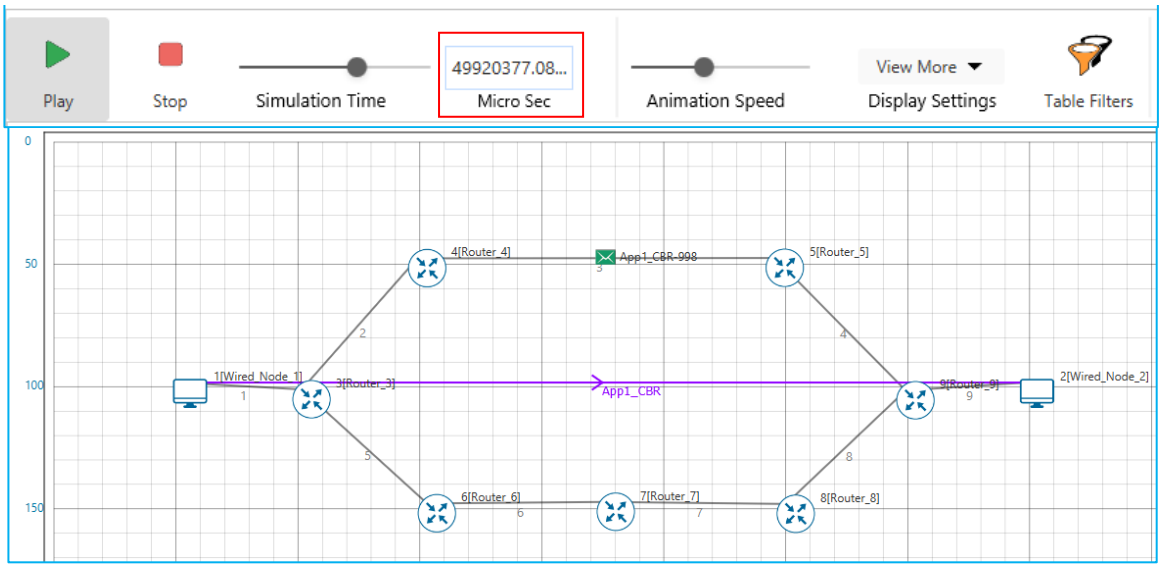
In Sample2,

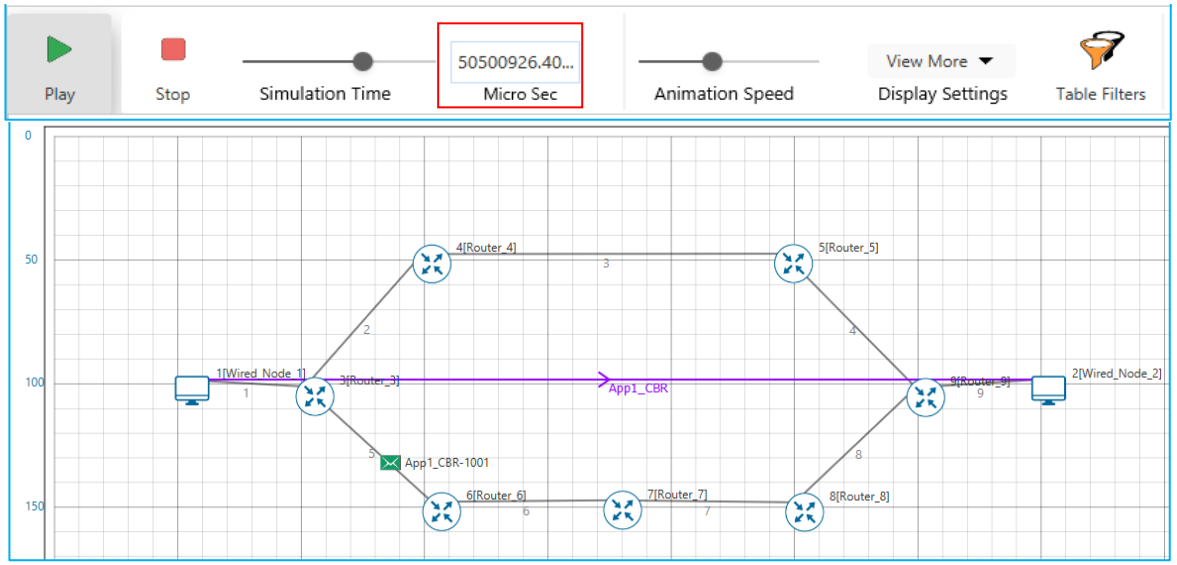
- We create a Link Failure in Link 3, between Router 4 and Router 5 at 50 Seconds.
- Hence the packets are not able to reach the destination. The routing protocol then recomputes an alternate path to the Destination.
- This can be observed in the Packet Trace.
- Go to the Results Dashboard and click 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.

PACKET_ID	SEGMENT_ID	PACKET_TYPE	CONTROL_PACKET_TYPE/APP_NAME	SOURCE_ID	DESTINATION_ID	TRANSMITTER_ID	RECEIVER_ID
994	0	CBR	App1_CBR	NODE-1	NODE-2	ROUTER-3	ROUTER-4
995	0	CBR	App1_CBR	NODE-1	NODE-2	ROUTER-3	ROUTER-4
996	0	CBR	App1_CBR	NODE-1	NODE-2	ROUTER-3	ROUTER-4
997	0	CBR	App1_CBR	NODE-1	NODE-2	ROUTER-3	ROUTER-4
998	0	CBR	App1_CBR	NODE-1	NODE-2	ROUTER-3	ROUTER-4
999	0	CBR	App1_CBR	NODE-1	NODE-2	ROUTER-3	ROUTER-4
1000	0	CBR	App1_CBR	NODE-1	NODE-2	ROUTER-3	ROUTER-4
1001	0	CBR	App1_CBR	NODE-1	NODE-2	ROUTER-3	ROUTER-4
1001	0	CBR	App1_CBR	NODE-1	NODE-2	ROUTER-3	ROUTER-6
1002	0	CBR	App1_CBR	NODE-1	NODE-2	ROUTER-3	ROUTER-6
1003	0	CBR	App1_CBR	NODE-1	NODE-2	ROUTER-3	ROUTER-6
1004	0	CBR	App1_CBR	NODE-1	NODE-2	ROUTER-3	ROUTER-6
1005	0	CBR	App1_CBR	NODE-1	NODE-2	ROUTER-3	ROUTER-6
1006	0	CBR	App1_CBR	NODE-1	NODE-2	ROUTER-3	ROUTER-6
1007	0	CBR	App1_CBR	NODE-1	NODE-2	ROUTER-3	ROUTER-6
1008	0	CBR	App1_CBR	NODE-1	NODE-2	ROUTER-3	ROUTER-6
1009	0	CBR	App1_CBR	NODE-1	NODE-2	ROUTER-3	ROUTER-6

- We can notice that packets are changing its route from,
  N1 > R3 > R4 > R5 > R9 > N2 to **N1 > R3 > R6 > R7 > R8 > R9 > 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:





34. IoT – Multi-Hop Sensor-Sink Path

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

34.1 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.*

34.2 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.

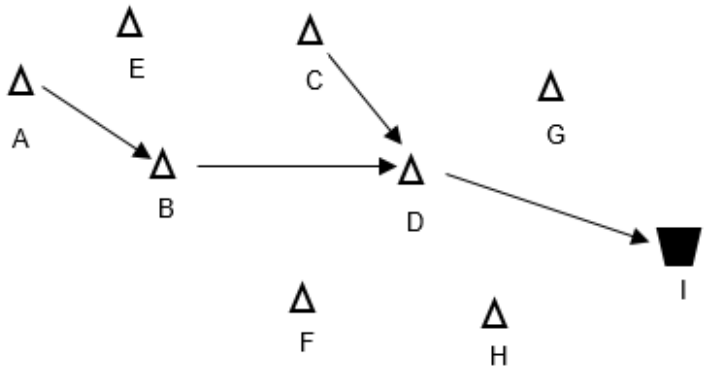


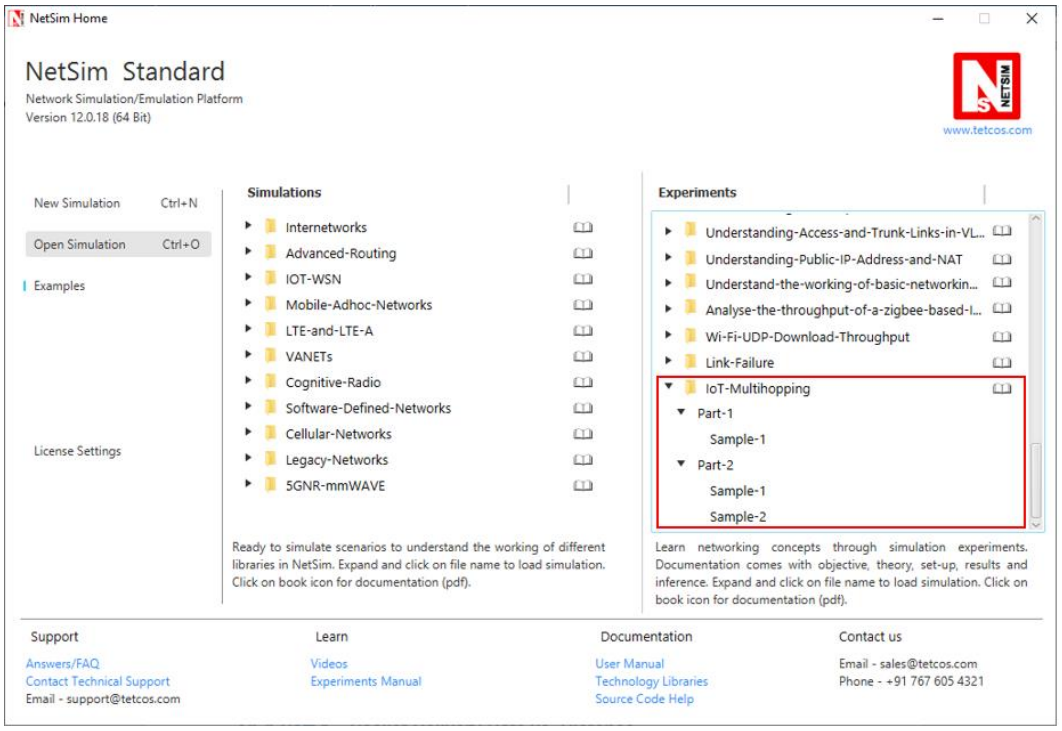
Fig: Data from Sensor A to Sink I takes the path A-B-D-I while data from sensor C to Sink I takes the path C-D-I

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?

34.3 Network Setup:

Open NetSim and click **Examples > Experiments > IoT-Multihopping > Part-1 > Sample-1** as shown below:

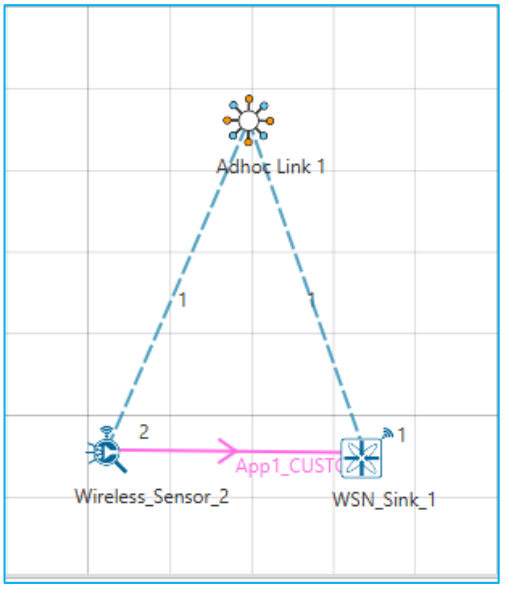


34.4 Part 1 – Packet Delivery Rate vs. Distance

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_0dBm, 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 η power of the distance d. The value of η is 2 for free space path loss and varies from 2 to 5 in the case of outdoor or indoor propagation. Values of η are obtained by carrying out experimental propagation studies.

Sample 1:

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



34.5 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.

Note: *By default, TCP is disabled in all the devices.*

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.

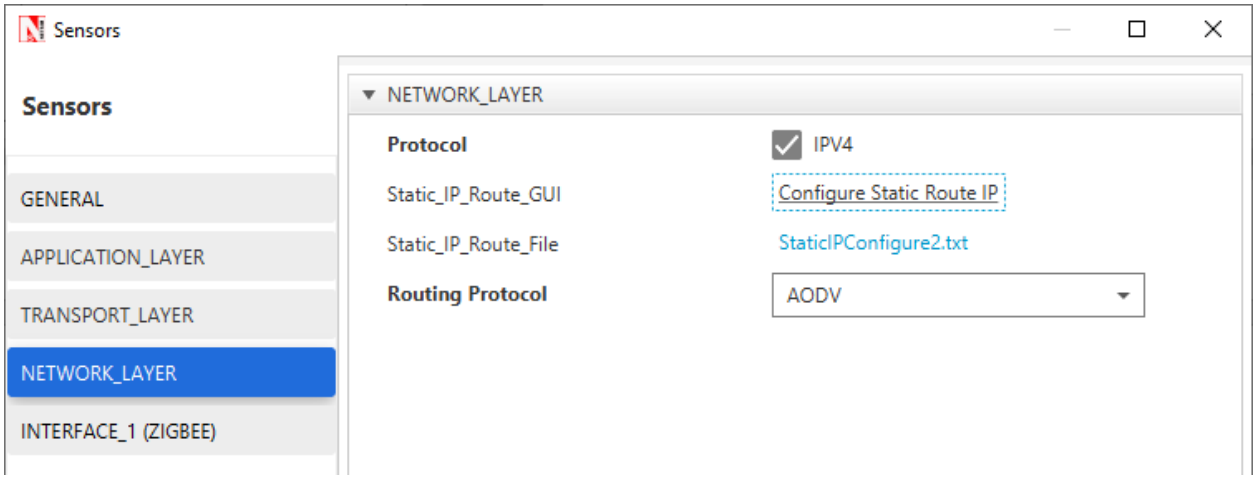
A CUSTOM Application is generated from Wireless Sensor 2 i.e. Source to WSN Sink 1 i.e. Destination with Packet Size set to 70 Bytes and Inter Arrival Time set to 4000 µs.

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:

$$Generation\ Rate\ (Mbps)\ =\ Packet\ Size\ (Bytes)\ *\ 8/Interarrival\ time\ (\mu s)$$

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**.

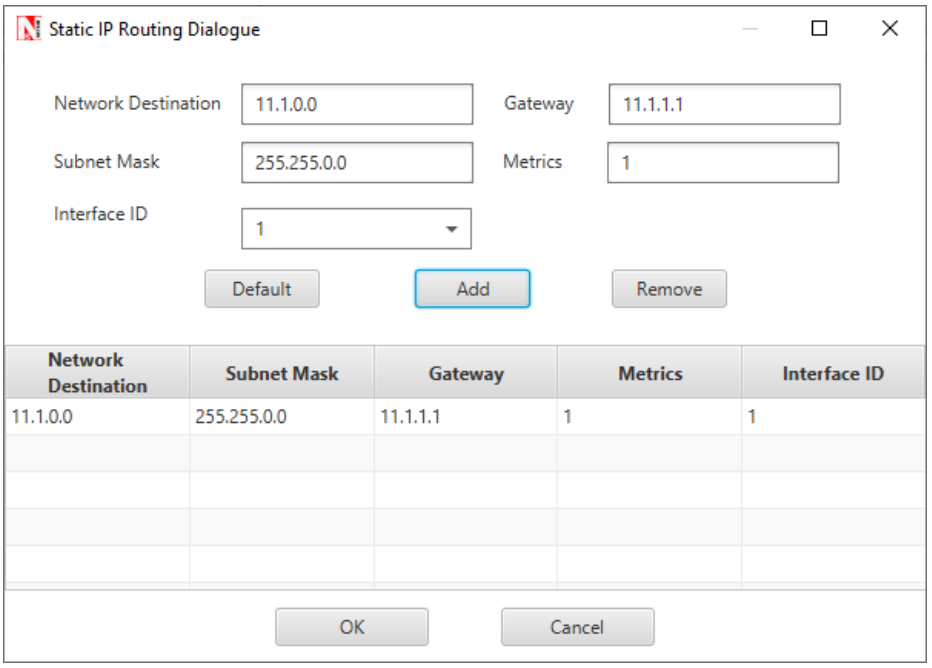


Static IP Routing Dialogue box gets open.

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 follwing lines of code highlighted in red, inside the **fn_NetSim_Zigbee_init()** function as shown below:

**_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,\t\tRECEIVER,\tRX_POWER(dBm),\tTOTAL_RX_POWE
R(dBm),\tBER");
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:

```
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)-
>dReceiverSensivity)
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,\t\t%s,\t%s,\t%lf,\t%lf,\t\t%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**.

34.6 Output:

RSSI, PER, BER vs. Distance (path-loss: linear in log-distance, with $\eta = 3.5$)				
Distance(m)	**RSSI (dBm)** (pathloss model)	**BER**	**PER**	**PLR** (after MAC retransmissions*)
5	-64.51	0.00	0	0
10	-75.04	0.00	0	0
15	-81.20	0.00	0	0
20	-85.58	0.00	0	0
25	-88.97	0.00	0	0
30	-91.74	0.00	0	0
35	-94.08	0.000005	0.0051	0
40	-96.11	0.000229	0.2076	0
45	-97.90	0.002175	0.8905	0.447
50	-99.51	0.008861	0.9999	1
55	-100.95	0.022370	1	1
60	-102.28	0.042390	1	1
65	-103.49	0.067026	1	1
70	-104.62	0.094075	1	1
75	-	-	-	-
80	-	-	-	-

* 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 -92dBm. At a distance of 35m the received power is -94 dBm, and we notice

a small BER of 5×10^{-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,

$$PL - packet\ length\ in\ bits\ at\ the\ PHY\ layer$$

$$PL\ (bits) = \big(70\ (payload) + 57(overhead)\big) * 8$$

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.

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

$$Total\ number\ of\ Lost\ packets = Total\ number\ of\ Packet\ Sent\ by\ SourceMAC - Packets\ Received\ at\ Destination\ MAC$$

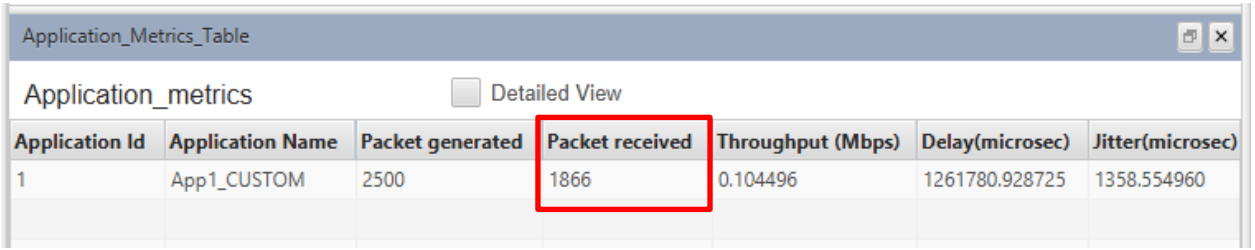
Steps to calculate Packet Loss Rate:

- 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.

PACKET_ID	SEGMENT_ID	PACKET_TYPE	CONTROL_PACKET_TYPE/APP_NAME	SOURCE_ID	DESTINATION_ID	TRANSMITTER_ID	RECEIVER_ID
1850	0	Custom	App1_CUSTOM	SENSOR-2	SINKNODE-1	SENSOR-2	SINKNODE-1
1851	0	Custom	App1_CUSTOM	SENSOR-2	SINKNODE-1	SENSOR-2	SINKNODE-1
1852	0	Custom	App1_CUSTOM	SENSOR-2	SINKNODE-1	SENSOR-2	SINKNODE-1
1853	0	Custom	App1_CUSTOM	SENSOR-2	SINKNODE-1	SENSOR-2	SINKNODE-1
1854	0	Custom	App1_CUSTOM	SENSOR-2	SINKNODE-1	SENSOR-2	SINKNODE-1
1855	0	Custom	App1_CUSTOM	SENSOR-2	SINKNODE-1	SENSOR-2	SINKNODE-1
1856	0	Custom	App1_CUSTOM	SENSOR-2	SINKNODE-1	SENSOR-2	SINKNODE-1
1857	0	Custom	App1_CUSTOM	SENSOR-2	SINKNODE-1	SENSOR-2	SINKNODE-1
1858	0	Custom	App1_CUSTOM	SENSOR-2	SINKNODE-1	SENSOR-2	SINKNODE-1
1859	0	Custom	App1_CUSTOM	SENSOR-2	SINKNODE-1	SENSOR-2	SINKNODE-1
1860	0	Custom	App1_CUSTOM	SENSOR-2	SINKNODE-1	SENSOR-2	SINKNODE-1
1861	0	Custom	App1_CUSTOM	SENSOR-2	SINKNODE-1	SENSOR-2	SINKNODE-1
1862	0	Custom	App1_CUSTOM	SENSOR-2	SINKNODE-1	SENSOR-2	SINKNODE-1
1863	0	Custom	App1_CUSTOM	SENSOR-2	SINKNODE-1	SENSOR-2	SINKNODE-1
1864	0	Custom	App1_CUSTOM	SENSOR-2	SINKNODE-1	SENSOR-2	SINKNODE-1
1865	0	Custom	App1_CUSTOM	SENSOR-2	SINKNODE-1	SENSOR-2	SINKNODE-1
1866	0	Custom	App1_CUSTOM	SENSOR-2	SINKNODE-1	SENSOR-2	SINKNODE-1

  This represents the total number of packets sent by the source.

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



Application Id	Application Name	Packet generated	Packet received	Throughput (Mbps)	Delay(microsec)	Jitter(microsec)
1	App1_CUSTOM	2500	1866	0.104496	1261780.928725	1358.554960

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,

$$Total\ number\ of\ Packet\ Sent\ by\ SourceMAC = 463$$

$$Packets\ Received\ at\ Destination\ MAC = 256$$

$$Total\ number\ of\ Lost\ packets = 463 - 256 = 207$$

$$PLR = \frac{207}{463} = 0.447$$

34.7 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.

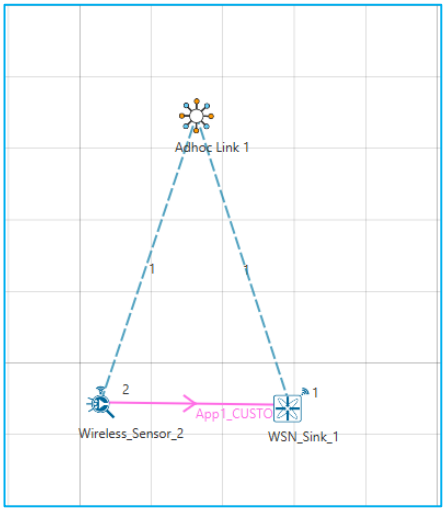
With the above discussion in mind, let us say that the application under consideration requires a measurement delivery rate of at least 80%. Examining the table above, we conclude that the sensor-sink distance must not be more than 40 meters. Thus, even a 1 acre farm $(61m \times 61m)$ would require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

34.8 Part 2 – Reaching a Longer Distance by Multihopping

Sample1:

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



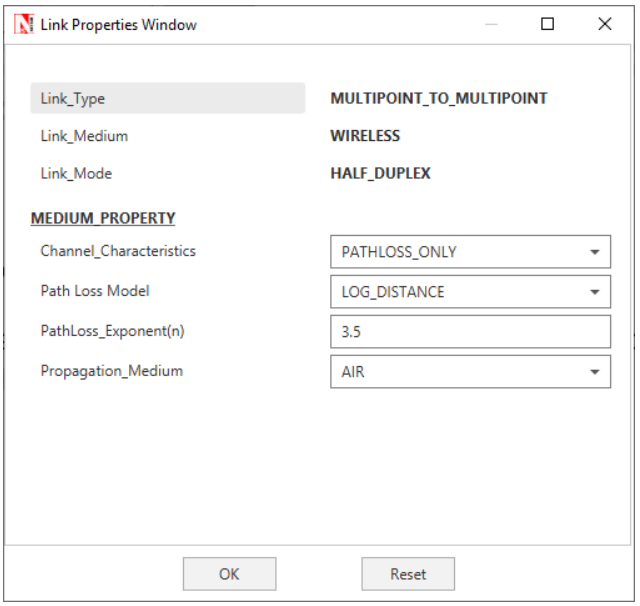
34.9 Procedure:

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.

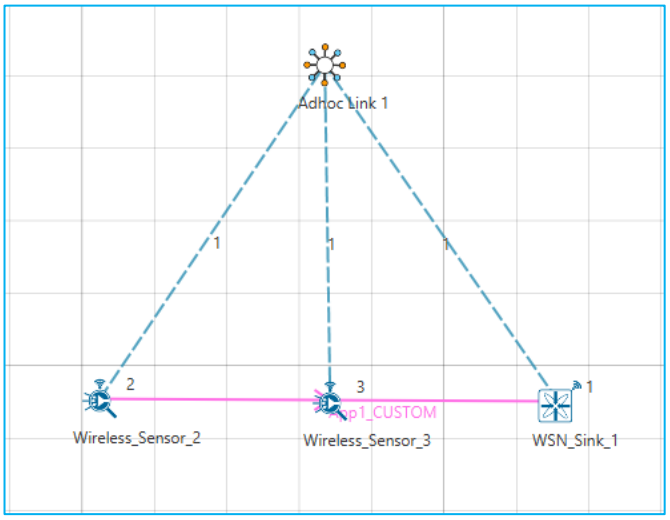
The Packet Size and Inter Arrival Time parameters are set such that the Generation Rate equals 5.6 Kbps. Generation Rate can be calculated using the formula:

$$Generation\ Rate\ (Mbps)\ =\ Packet\ Size\ (Bytes)\ *\ 8/Interarrival\ time\ (\mu s)$$

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:



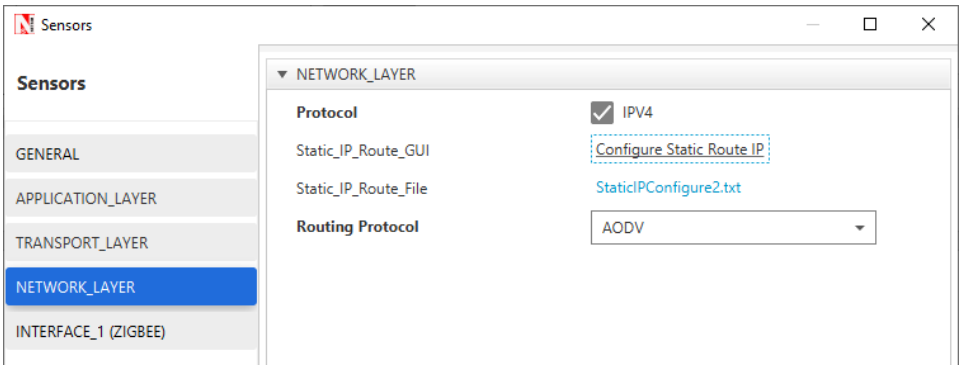
34.10 Procedure:

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**.

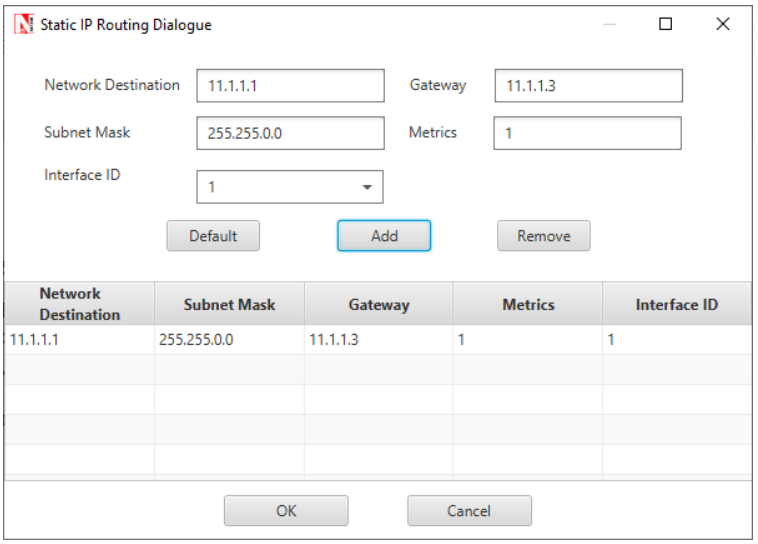


Static IP Routing Dialogue box gets open.

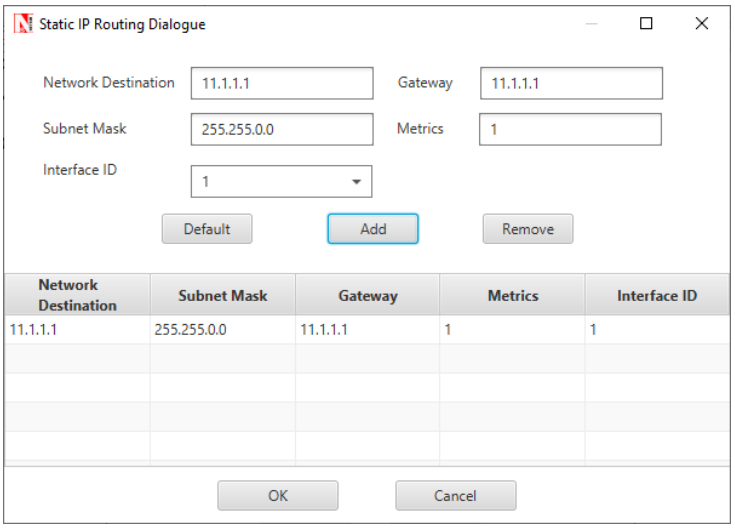
Enter the Network Destination, Gateway, Subnet Mask, Metrics, and Interface ID. Click on **Add**.

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:



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**.

34.11 Output:

	Source-Sink Distance (m)	Packets Generated	Packets Received	Packets Errored (PHY)	Packets Collided	Packet Loss (MAC)	PLR	Mean Delay (μs)
Direct sensor-sink link	40	1000	1012	244	0	0	0	6394.06
Router between sensor and sink	80 (router at midpoint)	1000	1016	540	0	0	0	14075.90

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.***

34.12 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.