

Introduction

NetSim is used by people from different areas such as academic researchers, industrial developers to design, simulate, verify, analyze the performance of different networks protocols and can be used to evaluate the effect of the different parameters on the protocols being studied.

NetSim Versions

NetSim comes in three versions, **Standard**, **Pro** and **Academic**. The standard version is used for project work and research while the academic version is used for lab experimentation and teaching. Pro version addresses the needs of defense and industry. The standard and pro version are available as 7 (seven) components which users can choose and assemble. The academic version is available as a single product and includes all the technologies shown below. The main difference between standard and academic is the availability of protocol source code, and packet tracing facility in the standard version

Component No	Networks / Protocols	International Standards
Component 1	Internetworks Ethernet, WLAN (802.11a, b, g), Routing (RIP/OSPF), TCP, UDP	IEEE 802.11 a/b/g RFC 2453,,2328, RFC's 793, 2001 and 768
Component 2	Legacy Networks Aloha (Pure & Slotted), CSMA/CD, Token ring, Token bus, ATM, X.25, Frame Relay, Real Time (Frame Capture)	IEEE 802.3, 802.4, 802.5, ATM Forum, ITU Forum
Component 3	BGP Networks & MPLS Networks BGP, MPLS	RFC 1771, IETF RFC's 3031 and 3121
Component 4	Advanced Wireless Networks MANET and Wi-Max	IETF RFC 4278, IEEE 802.16D
Component 5	Cellular Networks GSM and CDMA	3GPP, ETSI, IMT-MC, IS-95 A/B, IxRTT, 1x-EV-Do, 3xRTT
Component 6	Wireless Sensor Networks & Personal Area Networks WSN and ZigBee	IEEE 802.15.4 MAC, IETF RFC 4278
Component 7	Cognitive Radio Networks WRAN	IEEE 802.22

Note: NetSim v6 had 16 components which have been rationalized into 7 components in v7. The 16 component table of v6 is given below:

Component No	Protocols	Standards
Component 1	Aloha, Slotted Aloha, Token Ring, Token Bus, CSMA/CD	IEEE 802.3, 802.4, 802.5
Component 2	Net Patrol, LAN Speedometer	N/A
Component 3	Wireless LAN	IEEE 802.11 a/b/g
Component 4	Ethernet Switching, Gigabit Ethernet	IEEE 802.1 d
Component 5	X.25	ITU Forum
Component 6	Frame Relay	ITU Forum
Component 7	ATM	ATM Forum
Component 8	TCP, UDP	RFC's 793, 2001 and 768
Component 9	IP, Routing – RIP, OSPF, BGP	RFC 1058,2328, 1771
Component 10	MANET – DSR	IETF RFC 4278
Component 11	Wi-Max	IEEE 802.16 D
Component 12	Multi Protocol Label Switching (MPLS)	IETF RFCs 3031 and 3121
Component 13	GSM	3GPP, ETSI
Component 14	CDMA	IMT-MC, IS-95 A/B, 1xRTT, 1x-EV-Do, 3xRTT
Component 15	WSN	IEEE 802.15.4 MAC, IETF RFC 4278
Component 16	ZigBee	IEEE 802.15.4 MAC, IETF RFC 4278

What's New in NetSim v7?

NetSim v7 has several major new features including:

- Splendid aesthetic appeal with Map View, Grid View and Zoom capability
- **Internetwork:** With v7 users provides LAN-WAN-LAN modeling capability, while v6 had only stand alone modeling capability. Internetwork runs Ethernet, Wireless LAN, IP Routing and TCP / UDP
- **Cognitive Radio:** Component 7, Cognitive Radio is a new library added in v7.
- **Simulation run independent of UI:** NetSim v7 can run independent of UI by accepting the input scripts or commands in the configuration file from the end-user.
- **FSM protocol models:** Internetwork protocols are now modeled as finite state machines
- **Event Trace:** NetSim v7 allows users to generate event trace file that records every single event along with associated information such as time stamp, event ID, event type etc.
- **IPV4 API:** API's for IPV4 has been designed in NetSim v7 and this is available for the end-user.
- **Network Stack Implementation:** The Network Stack, an exact and powerful illustration of the TCP/IP 5 layer stack model in the real time is a notable feature of NetSim v7.
- **Traffic Generators:** User can generate various types of traffic like Video, FTP, Voice, Database and customers can write their own custom traffic in NetSim v7.
- **Multiple Applications in TCP and UDP:** Multiple applications with desirable traffic types can run at TCP and UDP in NetSim v7.
- **TCP 3 way handshake:** v7 TCP implements connection establishment and clearing
- **API for scheduling algorithms:** Scheduling algorithm API's for buffer management is a new feature in NetSim v7.

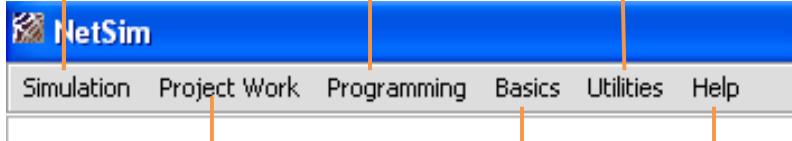
Getting Started

Main Menu

Opens the Simulation menu covering New, Open, Recent Experiments and Delete. User can Analyze, different Internetworks, Legacy, Cellular, BGP, MPLS, Advanced Wireless Networks and Cognitive Radio Networks Scenarios

Opens the Programming menu where different network programming exercises are available. Users can link and run their source code here.

Menu to create users, set passwords, and sample / exam mode. Switching of users can be done through the login as option.

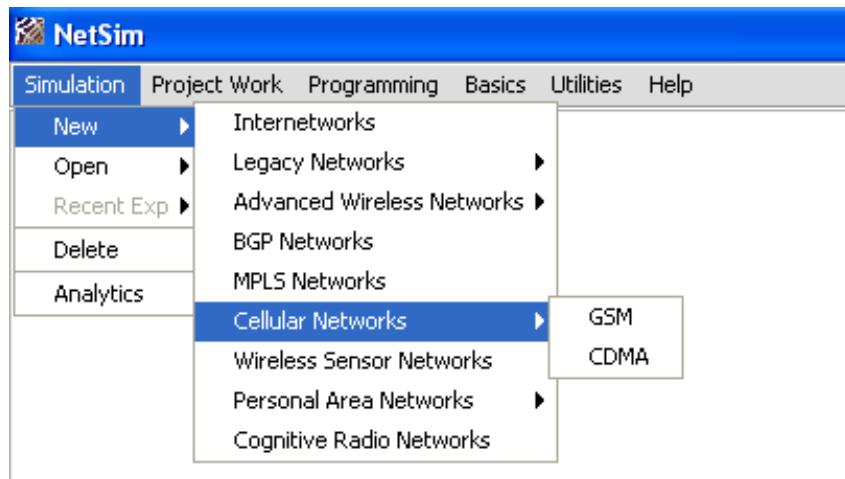


Displays all the Help related to NetSim. Help covers Simulation, Programming and Project Work.

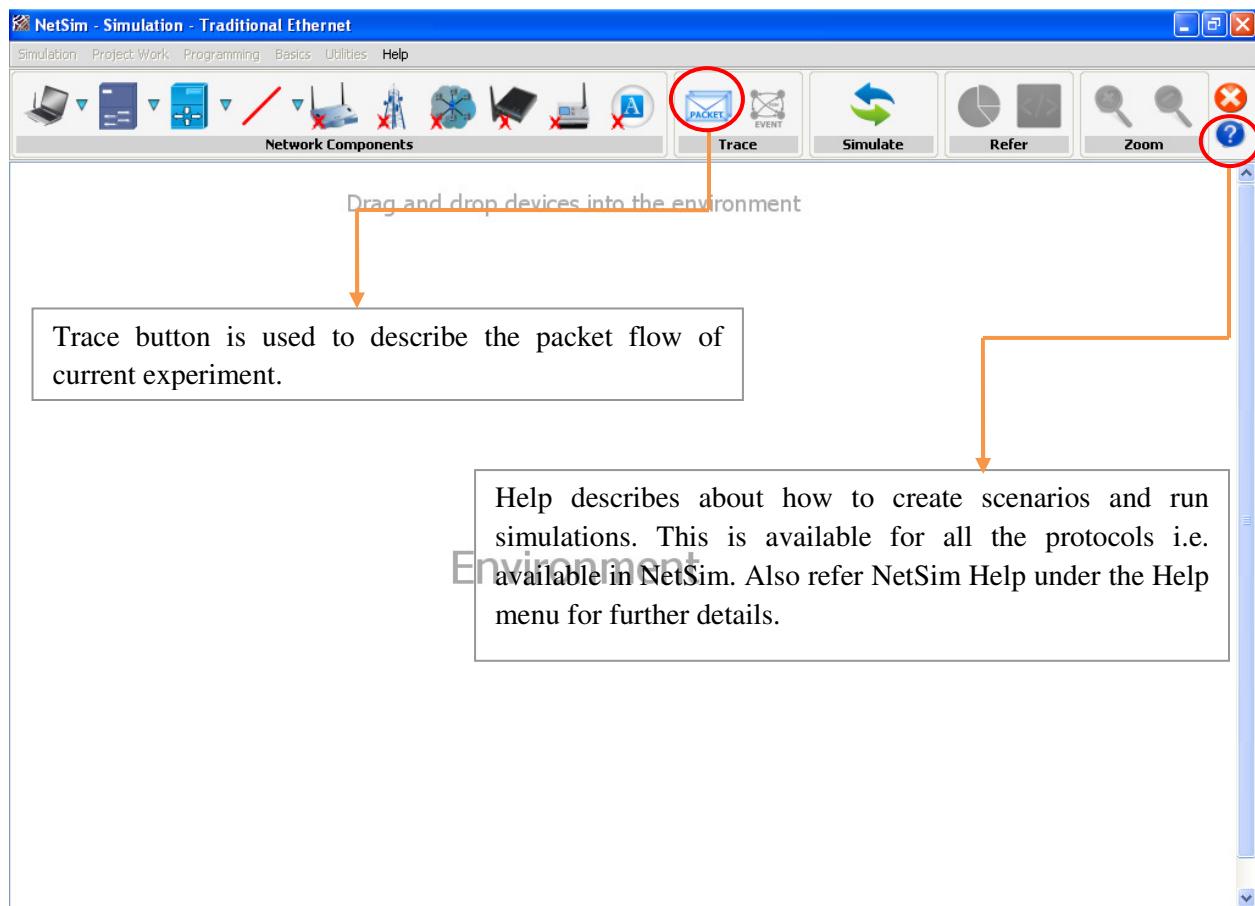
Consists of Animated explanations of networking principles, protocol working and packet formats.

Opens the Project Work menu to New, Open and Delete projects for protocols.
Note: This menu is available only with the Standard and Pro Versions.

Simulation

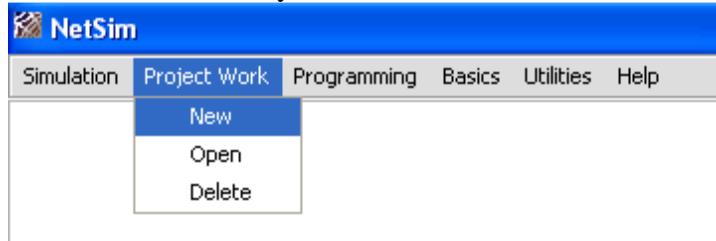


The **Simulation** menu contains options such as **New**, **Open**, **Delete** and **Analytics**. Select the desired protocol in **Legacy/Cellular/ BGP /MPLS/ Personal Area /Wireless Sensor** and **Cognitive Radio** under **New** and the screen similar to the one below will open up.



Project Work

Note: Project Work menu is available only with the Standard and Pro versions.



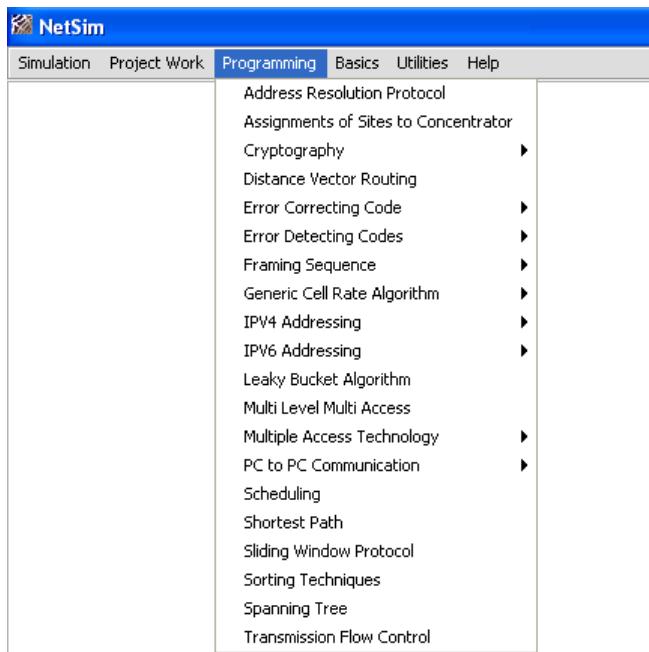
The **Project Work** menu contains options such as **New**, **Open**, **Delete**. Select the desired protocol from Protocol list under **New** and the screen similar to the one below will open up.

Clicking on the layers shows the list of protocols that works in the Respected Layers.

Networks list shows the available protocols for project work.

Selecting the list of primitives for the project work

Programming



The **Programming** menu contains network programming exercises. Run down the menu and select the desired programming exercise. Upon selection a screen similar to the one shown below will open up.

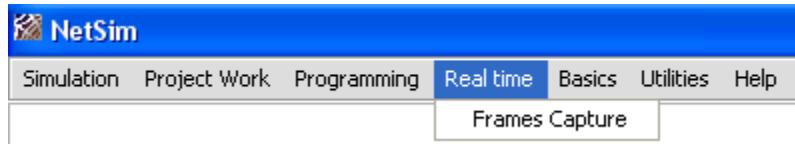
The screenshot shows the "Assignments of Sites to Concentrator" window within the NetSim - Programming interface. The window has a title bar "NetSim - Programming" and includes tabs for "Simulation", "Project Work", "Programming" (selected), "Realtime", "Basics", "Utilities", and "Help". The main area is titled "Assignments of Sites to Concentrator". It features a "Mode" section with "Sample" and "User" radio buttons, where "User" is selected. Below this is an "Input" section with fields for "Number of Sites", "Number of Concentrators", and "Number of Sites / Concentrator", each with dropdown menus. A note "(Click image to select priority)" is followed by a row of seven small icons labeled 1 through 7, with icon 1 highlighted. A "Selected Priority" button and a "Change Priority" link are also present. A "Distance : Enter integer (1 to 99 KM)" field and a "Concentrators" link are located at the bottom of the input section. The "Action" section contains "Run" and "Refresh" buttons. The "Help" section at the bottom includes links for "Concept , Algorithm , Pseudo Code & Flow Chart" and "Interface Source Code". Three callout boxes provide instructions: one pointing to the "User" mode radio button, another pointing to the "Concept , Algorithm , Pseudo Code & Flow Chart" link, and a third pointing to the "Interface Source Code" link.

Using the “User mode” users can link and run their own source code.

Clicking on the Concept, Algorithm, Pseudo Code and Flowchart would open-up for that program.

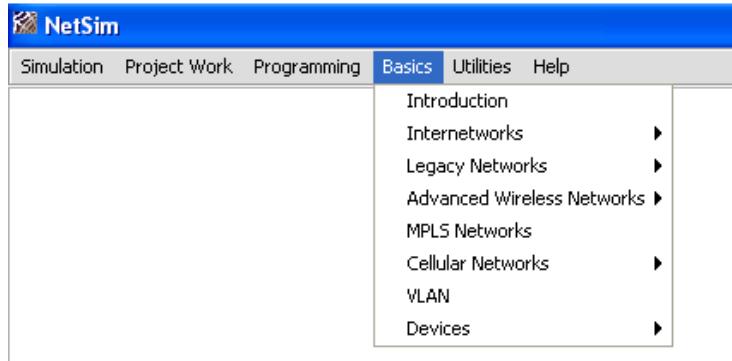
Clicking on Interface Source Code will open the .c source files

Real time



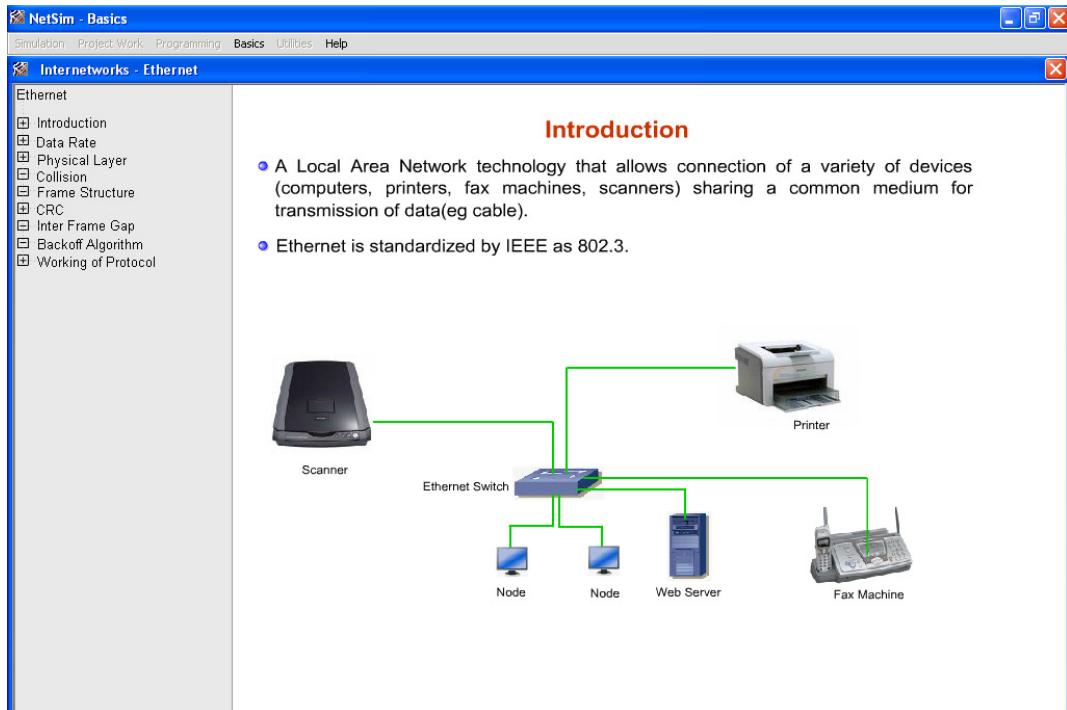
This section consists of Frame Capture. Frame Capture is used to capture packets in real time transfer. **Note: This menu is available only in Academic Version.**

Basics

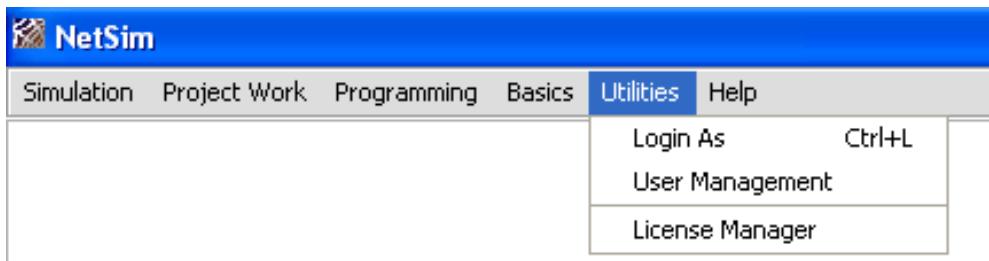


Select the desired item under **Basics** for animated explanations.

Example: Upon selecting “Ethernet” under “Internetworks”, following screen will open out.



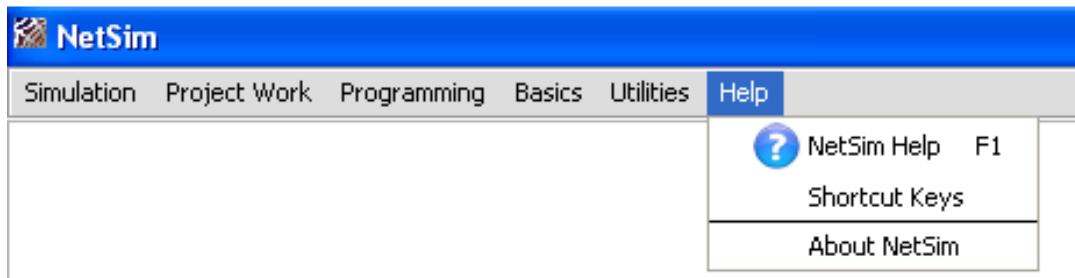
Utilities



This section displays consists of the following options,

- **Login As** - This option is used to login as that user which the user desires. This can be done based on the access provided.
- **User Management** - This option is used for Adding, Deleting, Change Password and Change Mode of the users. Also unwanted experiments can be deleted.
- **License Manager** – For controlling the dongle availability to the NetSim users in RLM Hardware Roaming license.

Help



This section displays all the **Help** related to **NetSim**.

NetSim Help covers **Simulation, Programming and Project Work**.

Shortcut Keys – it contains shortcut keys for all menus and sub menus.

NetSim File Organization

The install directory of NetSim also termed as “application path” is as explained below

Consider *C:\Program Files\NetSim Standard* as <app path>

➤ <*app path*>\bin

- ✓ All the “.dll” files like Simulation.dll, jdic.dll, rlm932.dll,etc.
- ✓ NetSimCore.exe

The following folder contains files essential to developers writing custom code in NetSim.

➤ <*app path*>\src

- <*app path*>\src\Simulation (This folder is not available in the Academic version)
 - <*app path*>\src\Simulation\include
 - ✓ packet.h
 - ✓ Stack.h
 - ✓ main.h
 - ✓ protocol.h
 - <*app path*>\src\Simulation\lib
 - ✓ protocol1.lib
 - ✓ protocol2.lib
 - ✓ protocol3.lib
 - ✓ NetworkStack.lib
 - ✓ Metrics.lib
 - <*app path*>\Docs\Sample_Configurations (This folder contains the sample configuration.xml files)
 - ✓ Configuration OSPF.xml
 - ✓ Configuration RIP.xml
 - ✓ TCP_SampleExperiment1_Configuration.xml

- ✓ Wlan_Node_AP_NodeConfiguration.xml
- ✓ Configuration_CR_FullIncumbentInterference.xml
- <app path>\src\Simulation\RIP
 - ✓ .c file
 - ✓ .h file
- <app path>\src\Simulation\TCP
 - ✓ .c file
 - ✓ .h file
- <app path>\src\Simulation\ARP
 - ✓ .c file
 - ✓ .h file
- <app path>\src\Simulation\FastEthernet
 - ✓ .c file
 - ✓ .h files
- <app path>\src\Simulation\WLAN
 - ✓ .c file
 - ✓ .h files
- <app path>\src\Programming
 - ✓ All .C files of programming exercises

Build (as .dll) your custom code into the bin folder of the app path.

Internetworks

Internetwork simulation is available from v7 of NetSim with LAN-WAN-LAN modeling capability. Internetwork runs Ethernet, Wireless LAN, IP Routing and TCP / UDP and allows users to log packet and event (in NetSim standard version only) traces.

New Experiments

In the **Simulation** menu select → **New** → **Internetworks**

Select either Map View or Grid view, and to perform experiments in **Internetworks**, the following steps should be followed,

- **Create Scenario**
- **Connect the devices**
- **Set Properties for the devices and the links**
- **Enable Packet Trace and Event Trace (Optional)**
- **Simulate**

Create Scenario

Adding Devices – Internetworks comes with the palette of various devices like Switch, Router, Wired Node, Wireless Node, AP, etc.

Select the desired devices in the toolbar and click and drop on the environment. To remove devices, right click on the particular device and then click Remove.

Connect the devices

Select the appropriate link in the toolbar and connect the devices by clicking on the device 1 and device 2.

Set the Properties for the devices and the links and enabling packet & event trace

Right click over the devices and then select Properties to set the properties of the links and the devices

Enable the packet trace to log the packet details by clicking on the packet trace in the toolbar and select the required attributes. Enable the event trace to log the event details by clicking on the event trace in the toolbar and select the required events

Simulate - After creating the **Scenario** the following steps need to be followed,

- Click on Simulate button. Select the Simulation End Time and then Click on “OK” button to start the Simulation

Legacy Networks

New Experiments

In the **Simulation** menu select → **New** → **Legacy Networks**

For example, to arrive **Pure Aloha**,

In the **Simulation** menu select → **New** → **Legacy Networks** → **Pure Aloha**.

To perform experiments in **Legacy Networks**, the following steps should be followed,

- **Create Scenario**
- **Establishing Connections**
- **Set Device Properties**
- **Enable Packet Trace (Optional)**
- **Simulate**

Create Scenario

Adding Node (*Note:* This is applicable for Pure Aloha, Slotted Aloha, Traditional Ethernet, Token Bus and Token Ring) -

- **Click on the Node icon** in the tool bar and drag and drop inside the grid. (*Note:* This is applicable for Pure Aloha and Slotted Aloha)
- **Nodes** cannot be connected directly to each other because an intermediate connecting component (such as **Hub or Concentrator**) is required. (*Note:* This is applicable for Traditional Ethernet, Token Bus and Token Ring)

Adding Hub (*Note:* This is applicable for Traditional Ethernet and Token Bus) -

- **Click on the Hub icon** in the tool bar and drag it onto the environment. By default a Hub has eight ports.

Adding Concentrator (*Note:* This is applicable for Token Ring) -

- **Click on the Concentrator icon** in the tool bar and drag it onto the environment. By default a Concentrator consists of eight ports.

Adding CPE (Customer Premise Equipment) (*Note:* This is applicable for ATM, X.25 and Frame Relay) -

- **Click on the CPE icon** in the tool bar, drag and drop the CPE on the environment.

Adding Switch - Click on the **Switch** icon in the tool bar, drag and drop the Switch on the environment. By default a Switch has eight ports.

Establishing Connections

The steps for connecting CPE and Switch as follows,

- The connections between the two **CPEs** cannot be made in the network.
- On **clicking** the two devices connection can be made.
- The connection possibilities are,
 - **CPE to Switch** and
 - **Switch to Switch**.

(*Note:* Depends upon the simulation, choose either ATM Switch or X.25 Switch or Frame Relay Switch)

Set Node Properties or CPE (Customer Premises Equipment) Properties

Right Click on the appropriate node or CPE to select Properties. Inside the properties' window click on Application 1 to modify its properties.

Transmission Type

This indicates the type of transmission made by this session, either Broadcast or Point to Point.

Destination

This property indicates the Destination Node.

Traffic Type

This property indicates the type of traffic. The traffic can either be Voice or Data.

Voice

Codec

Codec is the component of any voice system that translates between analog speech and the bits used to transmit them. Every codec transmits a burst of data in a packet that can be reconstructed into voice. Five different standards of voice codec's are given which can be selected depending on the variations required.

Service Type

- **CBR** - CBR stands for Constant Bit Rate. Packets of constant size are generated at constant inter arrival times.
- **VBR** - VBR stands for Variable Bit Rate. The two types of Suppression Model that can be selected are,
- **Deterministic**
- **Markov Chain**

Click **OK** to accept the user entered values. Click on the close button at the top right corner to exit the screen.

Data

Packet Size

Distribution: The options available for distribution are,

- **Constant**
- **Exponential**

Mean Packet Size (Bytes): Sets the size of the packets being generated by the chosen distribution. The ranges of values that can be entered are between 65 to 1500 bytes. By default 1500 bytes is entered.

Inter Arrival Time

This indicates the time gap between packets.

Distribution: The options available for distribution are,

- **Constant**
- **Exponential**
- **Uniform**

Mean Inter Arrival Time: Enter the average inter-arrival time between packets. A lower inter-arrival time would lead to a higher generation rate and the vice versa. The range of values that can be entered are between 1000 to 20000 Micro Sec. By default 20000 Micro Sec is entered.

Click **OK** to accept the user entered values. Click on the close button at the top right corner to exit the screen.

Data Link Layer (*Note:* This is applicable for ATM)

Scheduling: This option consists of

- **Round Robin** (by default this schedule is set)
- **Priority**
- **FIFO**

Quality Of Service Parameters (QOS): This options consists of

- **Peak Cell Rate (Sec)** - The values that can be entered should be in the range, **1** to **99999**. By default **10000** is entered in.
- **Cell Delay Variation (Micro Sec)** - The values that can be entered should be in the range, **0** to **99999**. By default **10000** is entered in.
- **Generic Cell Rate (GCRA) Type** - This option consists of,
 - **Virtual Scheduling Algorithm (VSA)**
 - **Continuous State Leaky Bucket Algorithm (CSLBA)**

Data Link Layer (*Note:* This is applicable for X.25)

Virtual Circuit: This option consists of

- **Switched** (by default this option is set)
- **Permanent**

ARQ: This options consists of

- **Stop and Wait**
- **Go Back N**
- **Selective Repeat**

Data Link Layer (*Note:* This is applicable for Frame Relay)

CIR (Committed Information Rate) Value: This option consists of

- 0.056 (by default this value is set)
- 0.112
- 0.224
- 0.448

Modifying/Viewing/Accepting Properties

On opening an already configured properties of an application the input fields will be frozen (i.e. the input cannot be changed). To modify these values click on the **Modify** button in the screen. Now the input value can be changed. Click on the **Accept** button, the modified values will be saved.

This **View** button is enabled once the **Accept** Button is clicked. To view the given values, click on the **View** button.

Set Link Properties (*Note:* This is applicable for ATM, X.25 and Frame Relay)

Error Rate (BER) - This property defines the rate at which the data is affected by error in the network. The values that can be selected are **10^-6, 10^-7, 10^-8, 10^-9** and **No Error**. By **default** the **Error Rate** is **10^-6**. **10^-6** means one error in every 1000000 bits transmitted.

Physical Medium - This property defines the type of **Physical Medium** that is used in the network. The types of **Physical Medium** used in this simulation are **E0, T1, E1, T2** and **E2**. By **default E0** is entered.

Data Rate (Mbps) - This property defines the rate at which the link transmits data. Based on the type of **Physical Medium** the value of **Data Rate** is automatically **calculated** and **substituted**. By **default 0.064 Mbps** is entered. Changes cannot be made to this property.

Distance (Kms) - This property defines the **Distance** between two devices in **Kilometers**. The range of values which can be entered are between **1** to **9999** Kms. By **default 1Km** is entered. A **4** digit value is the maximum **Distance** that can be entered.

Set Switch Properties (*Note:* This is applicable for ATM, X.25 and Frame Relay)

Right click on the appropriate **Switch** and **click Properties**.

Port Properties - The **Port Properties** that are mentioned below are common to all the **Ports** available in a **Switch**.

Sorting technique (*Note:* This is applicable for ATM)

This property consists of the following options,

- Round Robin
- Priority. By default Priority is available.
- FIFO (First In First Out)

ARQ (*Note:* This is applicable for X.25)

Automatic Repeat reQuest consists of the following options,

- Stop and Wait
- Go Back N
- Selective Repeat

Buffer Size (KB) (*Note:* This is applicable for ATM, X.25 and Frame Relay)

- This property specifies size of the **buffer** for that **port** on **Switch**. Options available are → “**8, 16, 32, 64, 128, 256, 512, 1024, 2048 and 4096**”. Default value is **8KB**.

Set Hub or Concentrator Properties

Right click on the appropriate **Hub** and click **Properties**. (For Traditional Ethernet and Token Bus)

Right click on the appropriate **Concentrator** and click **Properties**. (For Token Ring)

Device Id - This property is a value that is generated automatically. Each **Id** generated is **unique**.

Port Properties - The **Port** Properties for each **Port** are as follows,

Data Rate (Mbps) (*Note:* For Token Ring, Options available are **4 and 16 Mbps**. By default the value available is **16 Mbps**.)

- This defines the underlying data rate. Option available is **10 Mbps** indicates 10 Mega Bits per Second (**default value**)

Error Rate (BER) - This defines the rate of the error. Options available are

- **No Error** indicates zero error during transmission (**default value**).
- **10^-6** indicates one error in every 1,000,000 bits transmitted.
- **10^-7** indicates one error in every 10,000,000 bits transmitted.
- **10^-8** indicates one error in every 100,000,000 bits transmitted.
- **10^-9** indicates one error in every 1000,000,000 bits transmitted.

Physical Medium - This defines the type of **Physical Medium**. **Twisted Pair** is the only option available.

Persistence (*Note:* This is applicable for Traditional Ethernet) –

Options available are → “**1, 1/2, 1/3, 1/4... 1/15**”. **Default** value is **1**.

Connected To - This gives the **Node** number of the node, on selecting the check box available in the **Ports** section.

Communication - This property defines the mode of **Communication**. By **default**, a **Star topology network, Token Bus topology network and Token Ring topology network** works in **Half Duplex**. This property cannot be changed.

Enable Packet Trace (Optional)

- Click Packet Trace icon in the tool bar. This is used to log the packet details.
- Select the required attributes and click OK. Once the simulation is completed, the file gets stored in the specified location.

Simulate

After creating the **Scenario** the following steps need to be followed,

- **Click on Validate** button.
- **Click on Simulate** button.
- **Select the Simulation End Time** and then **Click on “OK”** button to start the **Simulation**.

Advanced Wireless Networks

New Experiments

In the **Simulation** menu select → **New** → **Advanced Wireless Networks**

For example, to arrive **MANET**,

In the **Simulation** menu select → **New** → **Advanced Wireless Networks** → **MANET**

To perform experiments in **Advanced Wireless Networks**, the following steps should be followed,

- **Create Scenario**
- **Set Device Properties**
- **Enable Packet Trace (Optional)**
- **Simulate**

Create Scenario

Adding Node -

- **Click** on the **Node** icon in the tool bar and drag and drop it inside the **grid**. (I.e. **Visibility Range** - The systems can move and communicate in this range only).
- **Nodes** are needed to find the Route to data, and then it starts its transmission through path.

A Node cannot be placed on another Node. A Node cannot float outside of the grid.

Adding Base Station (*Note:* This is applicable for Fixed Wi-MAX) -

- **Click** on the **Base Station** icon in the tool bar and **drag** it onto the environment.

Adding Subscriber -

- **Click** on the **Subscriber** icon in the tool bar and **drag and drop** it on the **Base Station** coverage area.
- **Subscriber** cannot be placed on another **Subscriber**. It has to be dragged and placed on the **Base Station** coverage area.

Set Node Properties (*Note:* This is applicable for MANET)

Right Click on the appropriate node to select Properties. Inside the properties' window click on Application 1 to modify its properties.

Transmission Type

This indicates the type of transmission made by this session, Point to Point.

Destination

This property indicates the Destination Node.

Traffic Type

This property indicates the type of traffic. The traffic can either be Voice or Data.

Voice

Codec

Codec is the component of any voice system that translates between analog speech and the bits used to transmit them. Every codec transmits a burst of data in a packet that can be reconstructed into voice. Three different standards of voice codec's are given which can be selected depending on the variations required.

Service Type

- **CBR** - CBR stands for Constant Bit Rate. Packets of constant size are generated at constant inter arrival times.
- **VBR** - VBR stands for Variable Bit Rate. The two types of Suppression Model that can be selected are,
 - **Deterministic**
 - **Markov Chain**
 - **Success Ratio (%)**

Click **OK** to accept the user entered values. Click on the close button at the top right corner to exit the screen.

Data

Packet Size

Distribution: The options available for distribution are,

- **Constant**
- **Exponential**

Mean Packet Size (Bytes): Sets the size of the packets being generated by the Constant distribution. The ranges of values that can be entered are between 10 to 10000 bytes. By default 1472 bytes is entered.

Inter Arrival Time

This indicates the time gap between packets.

Distribution: The options available for distribution are,

- **Exponential**
- **Uniform**
- **Constant**

Mean Inter Arrival Time: Enter the average inter-arrival time between packets. A lower inter-arrival time would lead to a higher generation rate and the vice versa. The ranges of values that can be entered are between 1000 to 20000 Micro Sec. By default 20000 Micro Sec is entered.

Data Link Layer

Protocol- A **default** value has been coded in; hence no change can be made.

MAC Address - A **default** value has been coded in, hence no change can be made.

RTS Threshold (Bytes) - This property is to allow the node to **enable/disable** the RTS/CTS mechanism. By **default** the value that is available is **2347 Bytes**. Limit ranges from **0** to **2347 Bytes**.

Retry Limit - Indicate the number of attempts that can be made by a frame. This **varies** from **1** to **7**. By **default** the **Retry Limit** value is **7**.

MTU- A **default** value has been coded in, hence no change can be made.

Physical Layer

Transmitter Power (milli watts) - This property defines the power level of the **Node**. By **default** the value for **Transmitter Power** is **100 milli watts**.

This **View** button is enabled once the **Accept** Button is clicked. To view the given values, click on the **View** button.

Click **OK** to accept the user entered values. Click on the close button at the top right corner to exit the screen.

Set Base Station (BS) Properties (*Note:* This is applicable for Fixed Wi-MAX)

Right click on the appropriate **Base Station (BS)** and **click Properties**. Options available are,

Wireless Properties - Under **Wireless Properties** tab the options available are,

Connected To - This property defines the **Wireless** medium connected to the **Base Station (BS)**. A **default** value is already entered; hence no changes can be done.

Communication - This property defines the **Communication** mode of the network. By **default**, a **WiMAX** network works in **Half Duplex**. This property cannot be changed.

Wi-MAX Standard – IEEE 802.16-2004

This **standard** is followed for modeling **Data link Layer** and **Physical Layer**.

Data Link Layer - Under **Data Link Layer** the sub-options available are,

MAC Address - A **default** value has been coded in, hence no change can be made.

Scheduling Technique – FIFO is set as scheduling technique.

Physical Layer - Under **Physical Layer** the sub-options available are,

Transmission type - This property specifies that the **OFDM** is used as a **Transmission** technology.

Total Number of Sub Carrier (NFFT) – This property defines the total number of sub carrier in the Channel. By default Total Number of Sub Carrier found is **256**

Number of Used Sub Carries (Nused) – This property defines the Number of used sub carrier in the channel. By default Total Number of used sub carrier found is **192**

Modulation Technique – This property defines the Modulation Technique used in transmission. By default **BPSK** is set as Modulation Technique. QPSK, 16QAM and 64 QAM are other modulation Technique can be selectable by user.

Coding Rate – This property defines the coding rate used in transmission. By default 1/2 is set as coding rate for BPSK Modulation Technique. 1/2 and 3/4 is used as other coding rate for QPSK, 16 QAM and 64 QAM Modulation Technique.

Downlink to Uplink Ratio – This property denotes Downlink and Uplink Transmission Ratio. **1:1** is set as default Downlink to Uplink Ratio.

Frequency (GHz) - This property defines the **frequency** allotted for the channel. By **default Frequency (GHz)** found is **3.5 GHz**.

Nominal Channel Bandwidth (MHz) – This property denotes Channel Bandwidth used for Transmission. **3.5 MHz** is set as default Nominal Channel Bandwidth. 5.5 and 10 MHz are the other channel bandwidth can be selectable by user.

Sampling Factor (n) – This property is used to find the symbol time. **8/7** is set as default Sampling Factor.

Guard Interval (G) – This property is used to ensure the transmission interference. **1/4** is set as default Guard Interval.

OFDM Frame Duration – This property is used to denote the OFDM frame duration value. **10 ms** is set as default OFDM Frame duration value. User can also select 20 ms as OFDM frame duration value.

Channel Error – This property denotes the Channel Error characteristics. By default **None** is set as Channel Error.

Set Subscriber Properties (*Note:* This is applicable for Fixed Wi-MAX)

Right Click on the appropriate Subscriber to select Properties. Inside the properties window click on Application1 to modify its properties.

Transmission Type

This indicates the type of transmission made by this session, Point to Point.

Destination

This property indicates the Destination Subscriber.

Traffic Type

This property indicates the type of traffic. The traffic can either be Voice.

Voice

Call Details

Call Interval Time (Secs)

This property denotes the Call Interval Time. Each Call begins exponentially with mean call inter arrival time. 300 sec is set as default Call Inter Arrival Time. User can select 450 secs or 600 secs as Call Interval Time.

Call Duration (Secs)

This property denotes how long the call continues. By default 60 secs is set as Call duration. User can select 120 and 240 secs as their call duration value.

Codec

Codec is the component of any voice system that translates between analog speech and the bits used to transmit them. Every codec transmits a burst of data in a packet that can be reconstructed into voice.

Service Type

- **CBR** - CBR stands for Constant Bit Rate. Packets of constant size are generated at constant inter arrival times.

Click **OK** to accept the user entered values. Click on the close button at the top right corner to exit the screen.

Packet Size

Packet Size (Bytes): Based on Codec Selection Packet Size can be set.

Inter Arrival Time

This indicates the time gap between packets. Based on Codec selection Inter Arrival Time can be set.

Generation Rate

This denotes the Traffic Generation rate by the Subscriber. Based on Packet size and Inter Arrival Time Generation Rate can be set.

Click **OK** to accept the user entered values. Click on the close button at the top right corner to exit the screen.

Transport Layer

Protocol

This property denotes which protocol is used in Transport layer. By Default **UDP** is used as Transport Layer protocol.

Segment Size (Bytes)

This property denotes maximum amount of data that a single packet contains. By default **1472** is set as default segment size.

Network Layer

Protocol

This property denotes the protocol used in Network layer. By default **IPV4** is used as Network Layer protocol.

IP Address

This property denotes the IP Address set for Subscriber. Based on Subscriber Id, IP Address varies.

Subnet Mask

This property denotes the default subnet mask used in network. By Default **255.255.255.0** is set as Subnet Mask.

Data Link Layer

MTU (Bytes)

This property denotes Maximum Transmission Unit set for Data link Layer protocol. **1500 Bytes** is set as Default MTU for Ethernet Protocol.

QoS Class

This property denotes which QoS Class service is used in Subscriber. By default **UGS** is set as QoS Class which supports Voice Over IP and offers fixed size unsolicited data grants on periodic basis.

MAC Address

This property denotes MAC Address value set for Subscriber. This MAC Address value varies for each Subscriber.

Convergence Sub Layer

This property denotes the transport type for packet. Packet CS is set as Convergence Sub Layer which is used for the transport of all packet-based protocols such as the Internet Protocols (IP).

Physical Layer

Transmission Type

This property denotes the transmission type used in Subscriber. By Default OFDM is set as Transmission Type for Subscriber.

Click **Accept** button.

Modifying/Viewing/Accepting Properties

On opening an already configured properties of an application the input fields will be frozen (i.e. the input cannot be changed). To modify these values click on the **Modify** button in the screen. Now the input value can be changed. Click on the **Accept** button, the modified values will be saved.

This **View** button is enabled once the **Accept** Button is clicked. To view the given values, click on the **View** button.

Set Environment Properties (*Note:* This is applicable for MANET)

Right click in side of the on the Environment and **click Properties**.

Mobility model

Mobility - A **default** value has been coded in; hence no change can be made.

Velocity (m/s) - This property defines movement speed of the nodes. **Default** value is **20 m/s**.

Pause Time (sec) – This property specifies that the “**Pause Time**”. Nodes will remain the same position for this time period.

Physical Layer

Transmission type - This property specifies that the **Direct Sequence Spread Spectrum (DSSS)** is used as a **Transmission** technology.

Channels - The **Channels** that are allowed for the **Nodes** to operate are **1, 6 and 11**. By **default** the value entered is **1**.

Frequency (MHz) - This property defines the **frequency** allotted for the channel selected. By **default Frequency (MHz)** found is **2412 MHz**, since the **Channel** value is **1**. Refer the table for further details,

Channels	Frequency (MHz)
1	2412
6	2437
11	2462

Channel Characteristics - This property defines the **Channel Characteristics** for **Nodes**. It consists of the option **With Shadowing** (by default this option is displayed).

Enable Packet Trace (Optional)

- Click Packet Trace icon in the tool bar. This is used to log the packet details.
- Select the required attributes and click OK. Once the simulation is completed, the file gets stored in the specified location.

Simulate - After creating the **Scenario** the following steps need to be followed,

- **Click on Validate** button.
- **Click on Simulate** button.
- **Select the Simulation End Time.**
- Select the **Mobility Animation**. (*Note:* This is applicable for MANET)
 1. “**On**” – User can view the Mobility animation at the end of the simulation.
 2. “**Off**”- Mobility animation will not be shown at the end of the simulation.

- Click on “OK” button to start the **Simulation**.
- Click on “Stop Simulation” to stop the **Simulation**.

BGP Networks

New Experiments

In the **Simulation** menu select **Simulation → New → BGP Networks**

To perform experiments in **BGP Networks**, the following steps should be followed,

- **Create Scenario**
- **Establishing Connections**
- **Set Device Properties**
- **Set Link Properties**
- **Enable Packet Trace (Optional)**
- **Simulate**

Create Scenario

Adding CPE (Customer Premise Equipment) - Click on the **Router CPE** icon in the tool bar, drag and drop the Router CPE on the **environment**.

Adding Border Router - Click on the **Border Router** icon in the tool bar, drag and drop the Border Router on the environment. An Autonomous system is created by default. Maximum you can have 3 Autonomous systems in a single scenario.

Adding Internal Router - Click on the **Internal Router** icon in the tool bar, drag and drop the Internal Router onto the Autonomous systems created. By default a Router has eight ports.

Establishing Connections

The steps for connecting Router CPE and Router as follows,

- The connections between the two **Router CPEs** cannot be made in the network.
- On **clicking** the two devices connection can be made.
- The connection possibilities are,
 - **CPE to Internal Router**
 - **CPE to Border Router**
 - **Internal Router to Internal Router**
 - **Internal Router to Border Router**
 - **Border Router to Border Router**

Set CPE (Customer Premises Equipment) Properties

Right Click on the appropriate node to select Properties. Inside the properties' window click on Application 1 to modify its properties.

Transmission Type

This indicates the type of transmission made by this session, only Point to Point Transmission can be made.

Destination

This property indicates the Destination CPE.

Traffic Type

This property indicates the type of traffic. The traffic can either be Voice or Data.

Voice

Codec

Codec is the component of any voice system that translates between analog speech and the bits used to transmit them. Every codec transmits a burst of data in a packet that can be reconstructed into voice. Five different standards of voice codec's are given which can be selected depending on the variations required.

Service Type

- **CBR** - CBR stands for Constant Bit Rate. Packets of constant size are generated at constant inter arrival times.
- **VBR** - VBR stands for Variable Bit Rate. The two types of Suppression Model that can be selected are,

- **Deterministic**

- **Markov Chain**

Click **OK** to accept the user entered values. Click on the close button at the top right corner to exit the screen.

Data

Packet Size

Distribution: The options available for distribution are,

- Constant
- Exponential

Mean Packet Size (Bytes): Sets the size of the packets being generated by the chosen distribution. The range of values that can be entered are between 10 to 10000 bytes. By default 1472 bytes is entered.

Inter Arrival Time

This indicates the time gap between packets.

Distribution: The options available for distribution are,

- Constant
- Exponential
- Uniform

Mean Inter Arrival Time: Enter the average inter-arrival time between packets. A lower inter-arrival time would lead to a higher generation rate and the vice versa. The range of values that can be entered are between 1000 to 20000 Micro Sec. By default 20000 Micro Sec is entered.

Click **OK** to accept the user entered values. Click on the close button at the top right corner to exit the screen.

Network Layer

IP Address: Set the IP Address of the Nodes by selecting the appropriate values.

Subnet Mask: Give the valid values in this field.

Priority

Click to select the Priority. Its Values are Low, Normal, Medium and High. By default the Priority is Low.

Modifying/Viewing/Accepting Properties

On opening an already configured properties of an application the input fields will be frozen (i.e. the input cannot be changed). To modify these values click on the **Modify** button in the screen. Now the input value can be changed. Click on the **Accept** button, the modified values will be saved.

This **View** button is enabled once the **Accept** Button is clicked. To view the given values, click on the **View** button.

Set Link Properties

CPE to Internal Router and CPE to Border Router Link - To set the property values **right click** the **Link** between **CPE to Internal Router** or **CPE to Border Router** and select **Properties** option. There are 4 property options that are available. They are,

Error Rate (BER) - This property defines the rate at which the data is affected by error in the network. The values that can be selected are **10^-6, 10^-7, 10^-8, 10^-9** and **No Error**. By **default** the **Error Rate** is **10^-6**. **10^-6** means one error in every 1000000 bits transmitted.

Physical Medium - This property defines the type of **Physical Medium** that is used in the network. The types of **Physical Medium** used in this simulation are **E0, T1, E1, T2** and **E2**. By **default E0** is entered.

Data Rate (Mbps) - This property defines the rate at which the link transmits data. Based on the type of **Physical Medium** the value of **Data Rate** is automatically **calculated** and **substituted**. By **default 0.064 Mbps** is entered. Changes cannot be made to this property.

Distance (Kms) - This property defines the **Distance** between two devices in **Kilometers**. By **default 1Km** is entered. A **4** digit value is the maximum **Distance** that can be entered.

Link Weight - This property defines the **weight** of the **Link**. This property depends on the **Protocol Type** selected inside **Router Properties** (i.e. If the **Protocol Type** is **RIP** then the **default** value is **1** and changes cannot be made. If the **Protocol Type** is **OSPF**, then any value **less than 1000** would be accepted and by **default** the value entered is **1**).

Internal Router to Internal Router and Internal Router to Border Router Link - To change the property values **right click** the **Link** between **Internal Router to Internal Router** or **Internal Router to Border Router** and **click Properties**. There are 4 options for which changes can be made. They are,

Error Rate (BER) - This property defines the rate at which the data is affected by error in the network. The values that can be selected are **10^-6, 10^-7, 10^-8, 10^-9** and **No Error**. By **default** the **Error Rate** is **10^-6**. **10^-6** means one error in every 1000000 bits transmitted.

Physical Medium - This property defines the type of **Physical Medium** used in the network. The **Physical Medium** that is used in connecting 2 **Routers** is a **CAT5**.

Data Rate (Mbps) - This property defines the rate at which the **Link** transmits data. The two types of **Data Rate** that can be selected are, **10** and **100 Mbps**. By **default** **10Mbps** is entered.

Distance (km) - This property defines the **Distance** between two devices in **Kilometers**. By **default 1Km** is entered. A **4** digit value is the maximum Distance that can be entered.

Link Weight - This property defines the **weight** of the **Link**. This property depends on the **Protocol Type** selected inside **Router Properties** (i.e. if the **Protocol Type** is **RIP** then the default value is **1** and changes cannot be made. If the **Protocol Type** is **OSPF**, then any value less than **1000** would be accepted and by **default** the value entered is **1**).

Border Router to Border Router Link - To change the property values **right click** the **Link** between **Border Router to Border Router** and **click Properties**. There are 4 options for which changes can be made. They are,

Error Rate (BER) - This property defines the rate at which the data is affected by error in the network. The values that can be selected are **10^-6, 10^-7, 10^-8, 10^-9** and **No Error**. By **default** the **Error Rate** is **10^-6**. **10^-6** means one error in every 1000000 bits transmitted.

Physical Medium - This property defines the type of **Physical Medium** used in the network. The **Physical Medium** that is used in connecting 2 **Routers** is a **CAT5**.

Data Rate (Mbps) - This property defines the rate at which the **Link** transmits data. The **Data Rate** of **1000Mbps** is fixed here.

Distance (km) - This property defines the **Distance** between two devices in **Kilometers**. By **default 1Km** is entered. A **4** digit value is the maximum Distance that can be entered.

Link Weight - This property defines the **weight** of the **Link**. This property depends on the **Protocol Type** selected inside **Router Properties** (i.e. if the **Protocol Type** is **RIP** then the default value is **1** and changes cannot be made. If the **Protocol Type** is **OSPF**, then any value less than **1000** would be accepted and by **default** the value entered is **1**).

Set Router Properties

Internal Router Properties - To change any property, **right click** on the appropriate **Router** to get the option window and **click Properties**.

Ports Properties - The number of ports available in a **Router** are **8**. For each **Port** in a **Router** the following properties are available,

IP Address - The **IP Address** must be filled up with a valid **IP Address**. By **default** the **IP Address** that would be filled is “**10.1.5.1**”.

Subnet Mask - By **default** the value that is found in the **Subnet Mask** field is “**255.255.255.0**”.

Connected To - This property gives the **Router CPE** that is connected to that **Port** (i.e. If the **CPE1** is connected to **Port1**, then when **Port1** is selected **CPE1** would be displayed).

Routing Properties

Buffer Size (KB) - This property defines the **Buffer size** that should be selected. **Click** on the drop down button to avail the values. The values that are available are, “**8, 16, 32, 64, 128, 256, 512, 1024, 2048 and 4096 (KB)**”. **8KB** is the **default** value.

Scheduling Type - Here the type of scheduler required can be selected. The 2 types of **Scheduling Type** are **Priority** and **FIFO**. By **default Priority** is available.

Routing Protocol - The options available under **Routing Protocol** are,

Protocol Type - The type of **protocol** to be used can be selected. There are two options for this. One is RIP and other one is OSPF.

If you select RIP as your Protocol Type, following options are available.

Periodic Time (1 to 30 sec) - The value that must be entered here should be within the **limit** given. By **default 30 seconds** is entered.

Expiration Time (sec) - Here the value that must be entered should be **greater than** that of **Periodic Time** and the **maximum value** is **180 sec**. By **default** the value entered is **180 seconds**.

Garbage collection Time (1 to 120 sec) - Here the value that can be entered is between **1 to 120 seconds**. By **default** the value entered is **120 seconds**.

If you select OSPF as your Protocol Type, the following options are available for you.

Router Priority (0 to 255) - Here the range of value that can be entered is 0 to 255. By default the value entered is 255.

LSA Refresh Time (sec) - By default the value given here is **150 seconds**.

LSA Max Age (sec) - By default the value given here is **300 seconds**.

Data Link Layer - The options available under this property is,

Protocol Type - By **default** the **Protocol Type** found here is **Ethernet**. Here changes cannot be made

Border Router Properties - To **change** any property, **right click** on the appropriate **Router** to get the option window and **click Properties**.

Ports Properties - The number of ports available in a **Router** are **8**. For each **Port** in a **Router** the following properties are available,

IP Address - The **IP Address** must be filled up with a valid **IP Address**. By **default** the **IP Address** that would be filled is **“10.1.5.1”**.

Subnet Mask - By **default** the value that is found in the **Subnet Mask** field is **“255.255.255.0”**.

Connected To - This property gives the **Router CPE** that is connected to that **Port** (i.e. If the **CPE1** is connected to **Port1**, then when **Port1** is selected **CPE1** would be displayed).

Routing Properties

Buffer Size (KB) - This property defines the **Buffer size** that should be selected. Click on the drop down button to avail the values. The values that are available are, “**8, 16, 32, 64, 128, 256, 512, 1024, 2048 and 4096 (KB)**”. **8KB** is the **default** value.

Scheduling Type - Here the type of scheduler required can be selected. The 2 types of **Scheduling Type** are **Priority** and **FIFO**. By **default Priority** is available.

Routing Protocol – Let say the selected BGP Router has 2 ports. One connected to Router (Internal / Border) of same Autonomous system (AS) and another connected to Router (Internal / Border) of different AS.

The options available under **Routing Protocol** if the port connecting to Routers of same AS are,

Protocol Type - The type of **protocol** to be used can be selected. There are two options for this. One is **RIP** and other one is **OSPF**.

If you select **RIP** as your **Protocol Type**, following options are available.

Periodic Time (1 to 30 sec) - The value that must be entered here should be within the **limit** given. By **default 30 seconds** is entered.

Expiration Time (sec) - Here the value that must be entered should be **greater than** that of **Periodic Time** and the **maximum value** is **180 sec**. By **default** the value entered is **180 seconds**.

Garbage collection Time (1 to 120 sec) - Here the value that can be entered is between **1 to 120 seconds**. By **default** the value entered is **120 seconds**.

If you select **OSPF** as your **Protocol Type**, the following options are available for you.

Router Priority (0 to 255) - Here the range of value that can be entered is 0 to 255. By default the value entered is 255.

LSA Refresh Time (sec) - By default the value given here is **150 seconds**.

LSA Max Age (sec) - By default the value given here is **300 seconds**.

The options available under **Routing Protocol** if the port connecting to Routers of different AS are,

Protocol Type - The type of **protocol** to be used can be selected. By default Protocol Type is BGP 4.

The following options are available for BGP protocol.

Local Preference: The possible values here are 100, 200, 300, 400 and 500. Default value is 100 here.

MED (Multiple Exit Discriminator): Here the range of value that can be entered is from 0 to 100. Default value is 0 here.

Data Link Layer - The options available under this property is,

Protocol Type - By **default** the **Protocol Type** found here is **Ethernet**. Here changes cannot be made

Enable Packet Trace (Optional)

- Click Packet Trace icon in the tool bar. This is used to log the packet details.
- Select the required attributes and click OK. Once the simulation is completed, the file gets stored in the specified location.

Simulate - After creating the **Scenario** the following steps need to be followed,

- **Click on Validate** button.
- **Click on Simulate** button.
- **Select the Simulation End Time** and then **click on “OK”** button to start the **Simulation**.

MPLS Networks

New Experiments

In the **Simulation** menu select **Simulation → New → MPLS Networks**

To perform experiments in **MPLS Networks**, the following steps should be followed,

- **Create Scenario**
- **Establishing Connections**
- **Set Device Properties**
- **Set Link Properties**
- **Enable Packet Trace (Optional)**
- **Simulate**

Create Scenario

Adding CPE (Customer Premise Equipment) - Click on the **Router CPE** icon in the tool bar, drag and drop the Router CPE on the **environment**.

Adding Router - Click on the **Router** icon in the tool bar, drag and drop the Router on the environment. By default a Router has eight ports.

Establishing Connections

The steps for connecting Router CPE and Router as follows,

The connections between the two Router CPEs cannot be made in the network.

- On **clicking** the two devices connection can be made.
- The connection possibilities are,
 - **CPE to Router** and
 - **Router to Router**.

Set CPE (Customer Premises Equipment) Properties

Right Click on the appropriate node to select Properties. Inside the properties' windows click on Application 1 to modify its properties.

Transmission Type

This indicates the type of transmission made by this session, only Point to Point Transmission can be made.

Destination

This property indicates the Destination CPE.

Traffic Type

This property indicates the type of traffic. The traffic can either be Voice or Data.

Voice

Codec

Codec is the component of any voice system that translates between analog speech and the bits used to transmit them. Every codec transmits a burst of data in a packet that can be reconstructed into voice. Six different standards of voice codec's are given which can be selected depending on the variations required.

Service Type

- **CBR** - CBR stands for Constant Bit Rate. Packets of constant size are generated at constant inter arrival times.

Click **OK** to accept the user entered values. Click on the close button at the top right corner to exit the screen.

Data

Packet Size

Distribution: The options available for distribution are,

- **Constant**

Mean Packet Size (Bytes): Sets the size of the packets being generated by the Constant distribution. The value of mean packet size is 1472 bytes

Inter Arrival Time

This indicates the time gap between packets.

Reserved Bandwidth:

This indicates the bandwidth reserved by user. The range of value is between 0.6 Mbps and 11 Mbps

Click **OK** to accept the user entered values. Click on the close button at the top right corner to exit the screen.

Network Layer

IP Address: Set the IP Address of the Nodes by selecting the appropriate values.

Subnet Mask: Give the valid values in this field.

Priority

Click to select the Priority. Its Values ranges from Low, Normal, Medium, and High. By default the Priority is Low.

Modifying/Viewing/Accepting Properties

On opening an already configured properties of an application the input fields will be frozen (i.e. the input cannot be changed). To modify these values click on the **Modify** button in the screen. Now the input value can be changed. Click on the **Accept** button, the modified values will be saved.

This **View** button is enabled once the **Accept** Button is clicked. To view the given values, click on the **View** button.

Set Link Properties

Link Properties

CPE to Router Link - To set the property values **right click** the **Link** between **CPE** to **Router** and select **Properties** option. There are 4 property options that are available. They are,

Error Rate (BER) - This property defines the rate at which the data is affected by error in the network. The values that can be selected are **10^-6**, **10^-7**, **10^-8**, **10^-9** and **No Error**. By **default** the **Error Rate** is **10^-6**. **10^-6** means one error in every 1000000 bits transmitted.

Physical Medium - This property defines the type of **Physical Medium** that is used in the network. The types of **Physical Medium** used in this simulation are **E0, T1, E1, T2** and **E2**. By **default E0** is entered.

Data Rate (Mbps) - This property defines the rate at which the link transmits data. Based on the type of **Physical Medium** the value of **Data Rate** is automatically **calculated** and **substituted**. By **default 0.064 Mbps** is entered. Changes cannot be made to this property.

Distance (Kms) - This property defines the **Distance** between two devices in **Kilometers**. By **default 1Km** is entered. A **4** digit value is the maximum **Distance** that can be entered.

Link Weight - This property defines the **weight** of the **Link**. This property depends on the **Protocol Type** selected inside **Router Properties** (i.e. If the **Protocol Type** is **RIP** then the **default** value is **1** and changes cannot be made. If the **Protocol Type** is **OSPF**, then any value **less than 1000** would be accepted and by **default** the value entered is **1**).

Router to Router Link - To change the property values **right click** the **Link** between **Router to Router** and **click Properties**. There are 4 options for which changes can be made. They are,

Error Rate (BER) - This property defines the rate at which the data is affected by error in the network. The values that can be selected are **10^-6, 10^-7, 10^-8, 10^-9** and **No Error**. By **default** the **Error Rate** is **10^-6**. **10^-6** means one error in every 1000000 bits transmitted.

Physical Medium - This property defines the type of **Physical Medium** used in the network. The **Physical Medium** that is used in connecting 2 **Routers** is a **CAT5**.

Data Rate (Mbps) - This property defines the rate at which the **Link** transmits data. The two types of **Data Rate** that can be selected are, **10** and **100 Mbps**. By **default 10Mbps** is entered.

Distance (km) - This property defines the **Distance** between two devices in **Kilometers**. By **default 1Km** is entered. A **4** digit value is the maximum Distance that can be entered.

Link Weight - This property defines the **weight** of the **Link**. This property depends on the **Protocol Type** selected inside **Router Properties** (i.e. if the **Protocol Type** is **RIP**

then the default value is **1** and changes cannot be made. If the **Protocol Type** is **OSPF**, then any value less than **1000** would be accepted and by **default** the value entered is **1**.

Set Router Properties

Router Properties - To change any property, right click on the appropriate **Router** to get the option window and click **Properties**.

Ports Properties - The number of ports available in a **Router** are **8**. For each **Port** in a **Router** the following properties are available,

Ip Address - The **Ip Address** must be filled up with a valid **Ip Address**. By **default** the **Ip Address** that would be filled is "**10.1.5.1**".

Subnet Mask - By **default** the value that is found in the **Subnet Mask** field is "**255.255.255.0**".

Connected To - This property gives the **Router CPE** that is connected to that **Port** (i.e. If the **CPE1** is connected to **Port1**, then when **Port1** is selected **CPE1** would be displayed).

Router Properties

Buffer Size (KB) - This property defines the **Buffer size** that should be selected. Click on the drop down button to avail the values. The values that are available are, "**8, 16, 32, 64, 128, 256, 512, 1024, 2048 and 4096 (KB)**". **8KB** is the **default** value.

Scheduling Type - Here the type of scheduler required can be selected. The 2 types of **Scheduling Type** are **Priority** and **FIFO**. By **default Priority** is available.

Routing Protocol - The options available under **Routing Protocol** are,

Protocol Type - The type of **protocol** to be used can be selected.

Router Priority – This indicates the priority of router. Range of the value is between **0 and 255**

LSA Refresh time (sec) – The refresh time for LSA for the simulation is **150 s**

LSA Max Age (sec) - The Max Age of LSA for the simulation is **350 s**

MPLS Properties

Traffic Engineering – Here the traffic engineering can be selected or deselected

Protocol type – Based on the selection and de selection of traffic engineering, the protocol type will be displayed. If the traffic engineering is selected, the protocol type is **CR-LDP**. Otherwise, the protocol type is **LDP**

Label Distribution type – Label Distribution type is **Downstream on Demand**

Label Distribution mode – Label Distribution mode is **Ordered**

Label Retention type – Label Retention type is **Conservative**

Data Link Layer - The options available under this property is,

Protocol Type - By **default** the **Protocol Type** found here is **Ethernet**. Here changes cannot be made.

MAC Address – MAC address of router is displayed

Enable Packet Trace (Optional)

- Click Packet Trace icon in the tool bar. This is used to log the packet details.
- Select the required attributes and click OK. Once the simulation is completed, the file gets stored in the specified location.

Simulate - After creating the **Scenario** the following steps need to be followed,

- **Click on Validate** button.
- **Click on Simulate** button.
- **Select the Simulation End Time** and then **click on “OK”** button to start the **Simulation**.

Cellular Networks

New Experiments

In the **Simulation** menu select → **New** → **Cellular Networks**

For Example, to arrive **CDMA**

In the **Simulation** menu select → **New** → **Cellular Networks** → **CDMA**

To perform experiments in **Cellular Networks**, the following steps should be followed,

- **Create Scenario**
- **Set Device Properties**
- **Enable Packet Trace (Optional)**
- **Simulate**

Create Scenario

Adding Base Station - Click on the **Base Station** icon in the tool bar and drag it onto the environment.

Adding Mobile Station -

- Click on the **Mobile Station** icon in the tool bar, drag, and drop it on the **Base Station** coverage area.
- **Mobile Station** cannot be placed on another **Mobile Station**. It has to be dragged and placed on the **Base Station** coverage area.

Set Base Station (BS) Properties

Right click on the appropriate **Base Station (BS)** and click **Properties**. Options available are,

Wireless Properties - Under **Wireless Properties** tab the options available are,

Device Type – This property defines the current device type. **Base Station** is set as default.

Connected To - This property defines the **Wireless** medium connected to the **Base Station (BS)**. A **default** value is already entered; hence no changes can be done.

Code Division Multiple Access (*Note:* This is applicable for CDMA)

Standards – This property defines standards used to perform the CDMA simulation. The default value is set as IS95A/B.

Total Bandwidth – This property defines the range of bandwidth that is going to be used for transmission. By default, **1.25 MHz** is set as default. This value is auto-change based on the standard type selection.

Chip Rate (in mcps) - This property specifies the chip rate of CDMA system. This value is auto-change based on the standard type selection. By default, **1.2288** is set for IS95A/B.

Voice Activity factor – This property defines the Voice activity for current system. This value is always in between 0-1. By default, **0** is set default voice activity factor.

BTS Range – This property defines the range of BTS in km. The default value is set as **1 km**.

Transmission Power (in W) – This property defines the transmission power used by the current base station. **20W** is set as default.

Modulation Technique – This property define the modulation technique used by the BTS. This value is fixed at GMSK.

Multiple Access Technology – This property defines the multiple access technology used by the CDMA system. This is fixed as **Code Division Multiple Access (CDMA)**.

Speech Coding – This property denotes the speech coding technique used in the Base station. This property is fixed at **Linear Predictive Coding (LPC)**.

Target SNR - This property denotes the target SNR used for the current scenario. This value is fixed at **6 db**.

Channel Characteristics-

Path loss exponent – This property defines the path loss exponent of the channel used by the current scenario. By default, this value is set as **4**.

Fading Figure – This property defines the fading figure of channel used by the current scenario. By default, this value is set as **0.5**.

Standard Deviation – This value denotes the standard deviation of fading of channel used by current scenario. By default, this value is set at **6**.

Global System for Mobile Communication (*Note:* This is applicable for GSM)

- **Up Link Bandwidth** – This property defines the range of uplink bandwidth that is going to be used for transmission. By default, **915 and 890** is set as Max and Min.
- **Down Link Bandwidth** – This property defines the range of downlink bandwidth that is going to be used for transmission. By default, **960 and 935** is set as Max and Min. This value is auto set based on the value entered in the uplink bandwidth.
- **Handover type** - This property specifies that the **hard handover** be used as a Handover technique.
- **Channel Bandwidth** – This property defines the channel bandwidth of one channel used by GSM system. The value is fixed at **200 kHz**.
- **BTS Range** – This property defines the range of BTS in km. The default value is set as **1 km**.
- **Transmission Power (in W)** – This property defines the transmission power used by the current base station. **20W** is set as default.
- **Modulation Technique** – This property defines the modulation technique used by the BTS. This value is fixed at GMSK.
- **Channel Data Rate** – This property defines the channel data rate used by GSM system. This value is fixed at **270.83kbps**.
- **Multiple Access Technology** – This property defines the multiple access technology used by the GSM system. This is fixed as **Time Division Multiple Access (TDMA)**.
- **Speech Coding** – This property denotes the speech coding technique used in the Base station. This property is fixed at **Linear Predictive Coding (LPC)**.
- **Duplex Distance** - This property denotes the duplex technique used by the GSM system. This property is fixed as **Frequency Division Duplexing (FDD)**.
- **Duplex Distance** – This property denotes the duplex distance of GSM system. The default value is 45 MHz.
- **Number of slot in each carrier** – This value denotes the number of slot in each frequency carrier. The value is 8

Set Mobile Station Properties

Right Click on the appropriate Mobile Station to select Properties. Inside the properties window click on Application1 to modify its properties.

Transmission Type

This indicates the type of transmission made by this Mobile station, Point to Point.

Destination

This property indicates the Destination Mobile Station.

Traffic Type

This property indicates the type of traffic. The traffic can only be Voice.

Voice

Call Details

Call Interval Time (Secs)

This property denotes the Call Interval Time. Each Call begins exponentially with mean call inter arrival time. 300 sec is set as default Call Inter Arrival Time. User can select 450 secs or 600 secs as Call Interval Time.

Call Duration (Secs)

This property denotes how long the call continues. By default 60 secs is set as Call duration. User can select 120 and 240 secs as their call duration value.

Codec

Codec is the component of any voice system that translates between analog speech and the bits used to transmit them. Every codec transmits a burst of data in a packet that can be reconstructed into voice.

Service Type

- **CBR** - CBR stands for Constant Bit Rate. Packets of constant size are generated at constant inter arrival times.

Click **OK** to accept the user entered values. Click on the close button at the top right corner to exit the screen.

Packet Size

Packet Size (Bytes): Based on Codec Selection Packet Size can be set.

Inter Arrival Time

This indicates the time gap between packets. Based on Codec selection Inter Arrival Time can be set.

Generation Rate

This denotes the Traffic Generation rate by the Mobile Station. Based on Packet size and Inter Arrival Time Generation Rate can be set.

Click **OK** to accept the user entered values. Click on the close button at the top right corner to exit the screen.

Mobility Management Layer

Mobility Model

This property denotes which mobility model is used for performing the mobility. By Default, No Mobility is used. Another available option is Random Walk Model.

Velocity (m/s)

This property denotes velocity of the mobile station if and only if **Random Walk Model is selected**. By default **20** is set as default velocity.

Cellular Networks Properties:

Data Link Layer

Protocol

This property denotes Protocol used in the Data Link Layer

For example, GSM is set as default protocol for GSM

Mobile Number

This property denotes the mobile number of the Mobile station.

IMEI No.

This property denotes the IMEI number of the Mobile station.

Physical Layer

Modulation Technique

This property denotes the type of modulation technique used in the Mobile Station. **GMSK** is set as default.

Transmission Power

This property denotes the transmission power of the mobile station.

Click **Accept** button.

Modifying/Viewing/Accepting Properties

On opening an already configured properties of an application the input fields will be frozen (i.e. the input cannot be changed). To modify these values click on the **Modify** button in the screen. Now the input value can be changed. Click on the **Accept** button, the modified values will be saved.

This **View** button is enabled once the **Accept** Button is clicked. To view the given values, click on the **View** button.

Enable Packet Trace (Optional)

- Click Packet Trace icon in the tool bar. This is used to log the packet details.
- Select the required attributes and click OK. Once the simulation is completed, the file gets stored in the specified location.

Simulate - After creating the **Scenario** the following steps need to be followed,

- **Click on Validate** button.
- **Click on Simulate** button.
- **Select the Simulation End Time and Mobility animation type** then **click on “OK”** button to start the **Simulation**

Wireless Sensor Networks

New Experiments

In the **Simulation** menu select **Simulation → New → Wireless Sensor Networks**

To perform experiments in **Wireless Sensor Networks**, the following steps should be followed,

- **Create Scenario**
- **Set Device Properties**
- **Set Environment Properties**
- **Enable Packet Trace (Optional)**
- **Simulate**

Create Scenario

Adding Sink Node- Click on the **Sink Node** icon in the tool bar and drag and drop inside the grid.

Adding Sensor - Click on the **Sensor Node** icon in the tool bar and drag and drop inside the grid.

Adding Agent- Click on the **Agent** icon in the tool bar and drag and drop inside the grid.

Set Sink Node Properties

Right Click on the Sink Node to select Properties.

Network Layer - The options available under this property is,

Routing Information – Reactive (On-Demand Routing) the process to find a path is only executed when a path is required by a node.

Routing Protocol- Dynamic Source Routing (DSR) which is a routing protocol, used in wireless sensor networks.

Data Link Layer - The options available under this property is,

Protocol – IEEE 802.15.4 is a standard which specifies the physical layer and media access control for low-rate wireless personal area networks (LR-WPANs).

Beacon Mode – Beacon Mode defines synchronization and reliability of the transmission mechanism, whereas beaconless networks adopt a simple lightweight protocol based on CSMA-CA. There are two options available, Enable/Disable. **Default** is set as **enable**

Beacon Order – Beacon Order, describes the interval at which the coordinator shall transmit its beacon frames. **15** denotes that **beacon mode is disabled**, if **Beacon mode is enabled** then the **Beacon Order value** can range from **0 to 14**. **Default Beacon Order value is 15**.

Superframe Order – Superframe Order describes the length of the active portion of the superframe, which includes the beacon frame. This property will be there, only in case of beacon mode is enabled. The value of Superframe Order should be less than **15**. **Default value is 15**.

GTS Mode - GTS (Guaranteed Time Slot) allows a device to operate on the channel within a portion of the superframe that is dedicated (on the PAN) exclusively to that device. This property will be there, only if beacon mode is enabled. By **default** the **GTS Mode is enabled**.

Superframe Duration – The Superframe Duration (SD) is divided into 16 equally sized time slots, during which data transmission is allowed. The SD can be further divided into a Contention Access Period (CAP)and an optional Contention Free Period (CFP) composed of Guaranteed Time Slots. A **default** value has been coded in; hence no change can be made. The fixed value is **15.36 ms**.

Battery Life Extension – The Battery Life Extension (BLE) subfield is 1 bit in length and shall be set to one if frames transmitted to the beaconsing device during its CAP(Contention Access Period) are required to start on or before macBattLifeExtPeriods full backoff periods after the IFS (inter frame space or spacing) period following the beacon. Otherwise, the **BLE** subfield shall be set to **0**. The options available are **True/False**. This property will be there, only in case of beacon mode is **enabled**. By **default** the **Battery Life Extension mode is true**.

Maximum Backoff Exponent – The maximum value of the backoff exponent (BE) in the CSMA-CA algorithm. The Maximum Backoff Exponent value ranges from **3 to 8**. **Default value is 5**.

Minimum Backoff Exponent – The minimum value of the backoff exponent (BE) in the CSMA-CA algorithm. The Minimum Backoff Exponent value ranges from **0 to Max BE**. Default value is **5**.

Max Frame retries – The maximum number of retries allowed after a transmission failure. The Max Frame retries value ranges from **0 to 7**. By default the **Max Frame retries** value is **3**.

MAX CSMA Backoff - The maximum number of backoffs the CSMA-CA algorithm will attempt before declaring a channel access failure. The MAX CSMA Backoff value ranges from **0 to 5**. Default value is **4**.

Unit Backoff Period - The number of symbols forming the basic time period used by the CSMA-CA algorithm. A **default** value has been coded in; hence no change can be made. The value is **20 Symbols**.

Min CAP length - The minimum number of symbols forming the CAP (contention access period). This ensures that MAC commands can still be transferred to devices when GTSs (Guaranteed Time Slot) are being used. A **default** value has been coded in; hence no change can be made. The value is **440 Symbols**

GTS descriptor Persistent Time - A **default** value has been coded in; hence no change can be made. The **GTS descriptor Persistent Time** is 4 s.

Physical Layer - The options available under this property is,

Protocol – IEEE 802.15.4 Phy - IEEE 802.15.4 is a standard which specifies the physical layer and media access control for low-rate wireless personal area networks (LR-WPANs).

Frequency Band - Frequency bands are groupings of radio frequencies that are used by mobile networks to communicate with mobile phones. A **default** value has been coded in; hence no change can be made. The value is **2.4 GHz**.

Data Rate - Data rate or bit rate is the number of bits that are conveyed or processed per unit of time. A **default** value has been coded in; hence no change can be made. The value is **250 kbps**.

Chip Rate - The chip rate of a code is the number of pulses per second (chips per second) at which the code is transmitted (or received). The chip rate is larger than the symbol rate, meaning that one symbol is represented by multiple chips. A **default** value has been coded in; hence no change can be made. The value is **2000 McPs**.

Symbol Rate - In digital communications, symbol rate (also known as baud or modulation rate) is the number of symbol changes (waveform changes or signaling events) made to the transmission medium per second using a digitally modulated signal or a line code. A **default** value has been coded in; hence no change can be made. The value is **62.5 KSymbolsPS**.

Modulation Technique - Offset quadrature phase-shift keying (OQPSK) is a variant of phase-shift keying modulation using 4 different values of the phase to transmit. It is sometimes called staggered quadrature phase-shift keying (SQPSK).

Min LIFS period - The minimum number of symbols forming a LIFS (Long Inter Frame Spacing) period. A **default** value has been coded in; hence no change can be made. The value is **40 Symbols**.

Min SIFS period - The minimum number of symbols forming a SIFS (Short Inter Frame Spacing) period. A **default** value has been coded in; hence no change can be made. The value is **20 Symbols**.

Unit Backoff Time - The number of symbols forming the basic time period used by the CSMA-CA algorithm. A **default** value has been coded in; hence no change can be made. The value is **20 Symbols**.

Turn Around Time - The TX-to-RX turnaround time is defined as the shortest time possible at the air interface from the trailing edge of the last chip (of the last symbol) of a transmitted PPDU to the leading edge of the first chip (of the first symbol) of the next received PPDU. The RX-to-TX turnaround time is defined as the shortest time possible at the air interface from the trailing edge of the last chip (of the last symbol) of a received PPDU to the leading edge of the first chip (of the first symbol) of the next transmitted PPDU. A **default** value has been coded in; hence no change can be made. The value is **12 Symbols**.

Phy SHR Duration – The duration of the synchronization header (SHR) in symbols for the current PHY. The Values available are **3, 7, 10, and 40**. By default Phy SHR Duration is **3**.

Phy Symbol Per Octet - The number of symbols per octet for the current PHY. The Values available are **0.4, 1.6, 2, 8**. By **default** phy symbol per octet is **0.4**.

CCA mode

The following are the available options under CCA mode,

- **Carrier Sense Only (by default this options is selected).**

CCA (Clear Channel Assessment) shall report a busy medium only upon the detection of a signal compliant with this standard with the same modulation and spreading characteristics of the PHY that is currently in use by the device. This signal may be above or below the ED threshold.

- **Energy Detection**

CCA shall report a busy medium upon detecting any energy above the ED threshold.

- **Carrier Sense with Energy Detection**

CCA shall report a busy medium using a logical combination of Detection of a signal with the modulation and spreading characteristics of this standard and Energy above the ED threshold, where the logical operator may be AND/OR.

Receiver Sensitivity –Threshold input signal power that yields a specified PER (packet error rate) conditions PSDU (PHY service data unit) length = 20 octets PER less than 1% Power measured at antenna terminals. Interference not present The Receiver Sensitivity Value should be less than **0**. **Default** value is **-85 dbm**.

ED threshold - The receiver ED threshold is intended for use by a network layer as part of a channel selection algorithm. It is an estimate of the received signal power within the bandwidth of the channel. No attempt is made to identify or decode signals on the channel. If the receive signal power is greater than the ED threshold value then the channel selection algorithm will return false. The **ED threshold** Value should be **less than 0**. **Default** value is **-95 dbm**.

Power Model - The options available under this property is,

Power Source - Power is got from battery.

Initial Power - Power is got from main line, and therefore the initial power is assumed to be infinite.

Set Agent Properties

Right Click on the Agent and **click Properties**. Options available are,

Agent Properties

Mobility Model

Mobility – The options available are,

- **Random Way Point** (default option) - Random Way Point (RWP) model is a commonly used synthetic model for mobility in networks. It is an elementary model which describes the movement pattern of independent nodes by simple terms.
- **Random Walk** - Random walk model is that of a random walk on a regular lattice, where at each step the walk jumps to another site according to a predefined probability distribution.

Velocity (m/s) - This property defines movement speed of the nodes. Velocity value should be less than 100. **Default** value is **20 m/s**.

Pause Time (sec) – This property specifies that the “**Pause Time**”. Nodes will remain the same position for this time period.

Data logging – Data logging is the process of collecting the data, analyzing it, saving and outputting the results of the collection and analysis. It has two options **Enable/Disable**. If it is **enabled** the following two properties will be visible

- **Data File Name** – It denotes that the name of the data file where the sink node stores the data.
- **Path** – It denotes the path, where the file is to be saved.

Sensing Interval (ms) – Sensing interval is the time difference between two sensor events. Sensing Interval should be less than 5000 ms. Default value is 1000 ms.

Set Sensor Properties

Right Click on the appropriate Sensor to select Properties.

Network Layer - The options available under this property is,

Routing Information – Reactive: (On-Demand Routing) the process to find a path is only executed when a path is required by a node.

Routing Protocol- Dynamic Source Routing (DSR) which is a routing protocol, used in wireless sensor networks.

Data Link Layer - The options available under this property is,

Protocol - IEEE 802.15.4 is a standard which specifies the physical layer and media access control for low-rate wireless personal area networks (LR-WPANs).

Mac Address - A **Media Access Control address** (MAC address) is a unique identifier assigned to network interfaces for communications on the physical network segment.

Retry Limit - This indicates the number of attempts that can be made by a frame. This value varies range from **0 to 7**.

Ack request – Frame transmitted with the Acknowledgment Request subfield of its frame control field set to one shall be acknowledged by the recipient. It can be enabled or disabled. By **default** it is **enabled**.

Physical Layer - The options available under this property is,

Protocol – IEEE 802.15.4 Phy is a standard which specifies the physical layer and media access control for low-rate wireless personal area networks (LR-WPANs).

Transmitter Power - This property defines the power level of the **Wireless device**. It ranges from **1 to 100 milli watts**. **Default** value is **20 milli watts**.

Sensor Range - A sensor is a device which receives and responds to a signal within a particular range. This value should be less than **100m**. **Default** value is **100 m**.

Transmitter Radius - The transmission range of the device. This changes based on environmental conditions. This value should be less than **100m**. **Default** value is **100 m**.

Receiver Sensitivity – Threshold input signal power that yields a specified PER (packet error rate) Conditions PSDU (PHY service data unit) length = 20 octets PER less than 1% Power measured at antenna terminals. Interference not present. **Receiver Sensitivity** value should be less than **0**. **Default** value is **-85 dbm**.

ED threshold – The receiver ED threshold is intended for use by a network layer as part of a channel selection algorithm. It is an estimate of the received signal power within the bandwidth of the channel. No attempt is made to identify or decode signals on the channel. If the receive signal power is greater than the ED threshold value then the channel selection algorithm will return false. **ED threshold** value should be less than **0**. **Default** value is **-95 dbm**.

Power Model - The options available under this property is,

Power Source –The options available under the power source is **Battery or Main Line**.

Power Source- Main Line

Power gets from main line, the main line is the general-purpose alternating current (AC) electric power supply.

Power Source - Battery - Power is got from battery. The options available under the **Battery is**,

Energy harvesting – Energy harvesting (also known as power harvesting or energy scavenging) is the process by which energy is derived from external sources (e.g., solar power, thermal energy, wind energy, and kinetic energy), captured, and stored for small, wireless autonomous devices, like those used in wearable electronics and wireless sensor networks. It can be **on or off mode**. **Default** it is **on mode**.

Recharging Current – The current flow during recharge. This property will be there only if the **Energy harvesting** is in **on** mode. **Default** value is **0.4 mA**.

Initial Energy – Initial energy stored in the battery. The Initial Energy value should be less than **1000kW**. **Default** value is **1000 kW**.

Voltage – Voltage is a measure of the energy carried by the charge. The Voltage value should be less than **10.0v**. **Default** value is **3.6 V**.

Transmitting Current –The current required for Transmission. The Transmitting Current value should be less than **20mA**. Default value is **8.8 mA**.

Receiving Current –The current required for receive the data. The Receiving Current value should be less than **20mA**. **Default** value is **9.6 mA**.

Idle Mode Current –The current flow during idle mode. The Idle Mode Current value should be less than **20mA**. Default value is **3.3 mA**.

Sleep Mode Current - The current flow during sleep mode. The Sleep Mode Current value should be less than **20mA**. **Default** value is **0.237 mA**.

Enable Packet Trace (Optional)

- Click Packet Trace icon in the tool bar. This is used to log the packet details.
- Select the required attributes and click OK. Once the simulation is completed, the file gets stored in the specified location.

Simulate - After creating the **Scenario** the following steps need to be followed,

- **Click on Validate** button.
- **Click on Simulate** button.
- **Select the Simulation End Time** and then **click on “OK”** button to start the **Simulation**.

Personal Area Networks

New Experiments

In the **Simulation** menu select **Simulation → New → Personal Area Networks**

For example, to arrive **ZigBee**

In the **Simulation** menu select **Simulation → New → Personal Area Networks → ZigBee**

To perform experiments in **Personal Area Networks**, the following steps should be followed,

- **Create Scenario**
- **Set Device Properties**
- **Set Environment Properties**
- **Enable Packet Trace (Optional)**
- **Simulate**

Create Scenario

Adding Node -

- **Click on the Node icon in the tool bar and drag and drop it inside the grid (i.e. Visibility Range** - The systems can move and communicate in this range only).
- A Node cannot be placed on another Node. A Node cannot float outside the **grid**.

Adding PAN Coordinator - Click on the PAN Coordinator icon in the tool bar and drag and drop inside the **grid.**

Set Node Properties

Right Click on the appropriate node to select Properties. Inside the properties' window clicks on Application 1 to modify its properties.

Transmission Type

This indicates the type of transmission made by this session, Point to Point.

Destination

This property indicates the Destination Node.

Traffic Type

This property indicates the type of traffic. The traffic can either be Voice or Data.

Voice

Codec

Codec is the component of any voice system that translates between analog speech and the bits used to transmit them. Every codec transmits a burst of data in a packet that can be reconstructed into voice. Three different standards of voice codec's are given which can be selected depending on the variations required.

Service Type

- **CBR** - CBR stands for Constant Bit Rate. Packets of constant size are generated at constant inter arrival times.
- **VBR** - VBR stands for Variable Bit Rate. The two types of Suppression Model that can be selected are,
 - **Deterministic**
 - **Markov Chain**

Success Ratio (%)

Click **OK** to accept the user entered values. Click on the close button at the top right corner to exit the screen.

Data

Packet Size

Distribution: The options available for distribution are,

- **Constant**
- **Exponential**

Mean Packet Size (Bytes): Sets the size of the packets being generated by the Constant distribution. The ranges of values that can be entered are between 10 to 10000 bytes. By default 1472 bytes is entered.

Inter Arrival Time

This indicates the time gap between packets.

Distribution: The options available for distribution are,

- **Exponential**
- **Uniform**
- **Constant**

Mean Inter Arrival Time: Enter the average inter-arrival time between packets. A lower inter-arrival time would lead to a higher generation rate and the vice versa. The ranges of values that can be entered are between 1000 to 20000 Micro Sec. By default 20000 Micro Sec is entered.

Data Link Layer

Protocol – IEEE 802.15.4 is a standard which specifies the physical layer and media access control for low-rate wireless personal area networks (LR-WPANs).

Mac Address - A Media Access Control address (MAC address) is a unique identifier assigned to network interfaces for communications on the physical network segment.

Device Type- Full Function Device is one of the devices in IEEE 802.15.4 wireless networks. By **default** the device type is **FFD**.

Retry Limit - Indicate the number of attempts that can be made by a frame. This **varies** from **1** to **7**. By **default** the **Retry Limit** value is **7**.

Ack request – Frame transmitted with the Acknowledgment Request subfield of its frame control field set to one shall be acknowledged by the recipient. It can be enabled or disabled. By **default** it is **enabled**.

Physical Layer

Protocol – IEEE 802.15.4 is a standard which specifies the physical layer and media access control for low-rate wireless personal area networks (LR-WPANs).

Transmitter Power (milli watts) - This property defines the power level of the **Node**. By **default** the value for **Transmitter Power** is **100 milli watts**.

Transmitter Range (meters) - This property defines the Range of Transmitter. By **default** the value for **Transmitter Range** is **100 meters**.

This **View** button is enabled once the **Accept** Button is clicked. To view the given values, click on the **View** button.

Click **OK** to accept the user entered values. Click on the close button at the top right corner to exit the screen.

Modifying/Viewing/Accepting Properties

On opening an already configured properties of an application the input fields will be frozen (i.e. the input cannot be changed). To modify these values click on the **Modify** button in the screen. Now the input value can be changed. Click on the **Accept** button, the modified values will be saved.

This **View** button is enabled once the **Accept** Button is clicked. To view the given values, click on the **View** button.

Set PAN Coordinator Properties

Right Click on the **PAN Coordinator** and **click Properties**. Options available are,

PAN Coordinator Properties

Data Link Layer - The options available under this property is,

Protocol – IEEE 802.15.4 is a standard which specifies the physical layer and media access control for low-rate wireless personal area networks (LR-WPANs).

Beacon Mode – Beacon Mode defines synchronization and reliability of the transmission mechanism, whereas beaconless networks adopt a simple lightweight protocol based on CSMA-CA. There are two options available, Enable/Disable. **Default** is set as **enable**

Beacon Order – Beacon Order, describes the interval at which the coordinator shall transmit its beacon frames. **15** denotes that **beacon mode is disabled**, if **Beacon mode is enabled** then the **Beacon Order value** can range from **0 to 14**. **Default Beacon Order value is 15**.

Superframe Order – Superframe Order describes the length of the active portion of the superframe, which includes the beacon frame. This property will be there, only in case of beacon mode is enabled. The value of Superframe Order should be less than **15**. **Default value is 15**.

GTS Mode - GTS (Guaranteed Time Slot) allows a device to operate on the channel within a portion of the superframe that is dedicated (on the PAN) exclusively to that device. This property will be there, only if beacon mode is enabled. By **default** the **GTS Mode** is **enabled**.

Superframe Duration – The Superframe Duration (SD) is divided into 16 equally sized time slots, during which data transmission is allowed. The SD can be further divided into a Contention Access Period (CAP) and an optional Contention Free Period (CFP) composed of Guaranteed Time Slots. A **default** value has been coded in; hence no change can be made. The fixed value is **15.36 ms**.

Battery Life Extension – The Battery Life Extension (BLE) subfield is 1 bit in length and shall be set to one if frames transmitted to the beacons device during its CAP(Contention Access Period) are required to start on or before macBattLifeExtPeriods full backoff periods after the IFS (inter frame space or spacing) period following the beacon. Otherwise, the **BLE** subfield shall be set to **0**. The options available are **True/False**. This property will be there, only in case of beacon mode is **enabled**. By **default** the **Battery Life Extension mode** is **true**.

Maximum Backoff Exponent – The maximum value of the backoff exponent (BE) in the CSMA-CA algorithm. The Maximum Backoff Exponent value ranges from **3 to 8**. **Default** value is **5**.

Minimum Backoff Exponent – The minimum value of the backoff exponent (BE) in the CSMA-CA algorithm. The Minimum Backoff Exponent value ranges from **0 to Max BE**. **Default** value is **5**.

Max Frame retries – The maximum number of retries allowed after a transmission failure. The Max Frame retries value ranges from **0 to 7**. By **default** the **Max Frame retries** value is **3**.

MAX CSMA Backoff - The maximum number of backoffs the CSMA-CA algorithm will attempt before declaring a channel access failure. The MAX CSMA Backoff value ranges from **0 to 5**. **Default** value is **4**.

Unit Backoff Period - The number of symbols forming the basic time period used by the CSMA-CA algorithm. A **default** value has been coded in; hence no change can be made. The value is **20 Symbols**.

Min CAP length - The minimum number of symbols forming the CAP (contention access period). This ensures that MAC commands can still be transferred to devices when GTSs (Guaranteed Time Slot) are being used. A **default** value has been coded in; hence no change can be made. The value is **440 Symbols**

GTS descriptor Persistent Time - A **default** value has been coded in; hence no change can be made. The **GTS descriptor Persistent Time** is 4 s.

Physical Layer - The options available under this property is,

Protocol – IEEE 802.15.4 Phy - IEEE 802.15.4 is a standard which specifies the physical layer and media access control for low-rate wireless personal area networks (LR-WPANs).

Frequency Band - Frequency bands are groupings of radio frequencies that are used by mobile networks to communicate with mobile phones. A **default** value has been coded in; hence no change can be made. The value is **2.4 GHz**.

Data Rate - Data rate or bit rate is the number of bits that are conveyed or processed per unit of time. A **default** value has been coded in; hence no change can be made. The value is **250 kbps**.

Chip Rate - The chip rate of a code is the number of pulses per second (chips per second) at which the code is transmitted (or received). The chip rate is larger than the symbol rate, meaning that one symbol is represented by multiple chips. A **default** value has been coded in; hence no change can be made. The value is **2000 McPs**.

Symbol Rate - In digital communications, symbol rate (also known as baud or modulation rate) is the number of symbol changes (waveform changes or signaling events) made to the

transmission medium per second using a digitally modulated signal or a line code. A **default** value has been coded in; hence no change can be made. The value is **62.5 KSymbolsPS**.

Modulation Technique - Offset quadrature phase-shift keying (OQPSK) is a variant of phase-shift keying modulation using 4 different values of the phase to transmit. It is sometimes called staggered quadrature phase-shift keying (SQPSK).

Min LIFS period - The minimum number of symbols forming a LIFS (Long Inter Frame Spacing) period. A **default** value has been coded in; hence no change can be made. The value is **40 Symbols**.

Min SIFS period - The minimum number of symbols forming a SIFS (Short Inter Frame Spacing) period. A **default** value has been coded in; hence no change can be made. The value is **20 Symbols**.

Unit Backoff Time - The number of symbols forming the basic time period used by the CSMA-CA algorithm. A **default** value has been coded in; hence no change can be made. The value is **20 Symbols**.

Turn Around Time - The TX-to-RX turnaround time is defined as the shortest time possible at the air interface from the trailing edge of the last chip (of the last symbol) of a transmitted PPDU to the leading edge of the first chip (of the first symbol) of the next received PPDU. The RX-to-TX turnaround time is defined as the shortest time possible at the air interface from the trailing edge of the last chip (of the last symbol) of a received PPDU to the leading edge of the first chip (of the first symbol) of the next transmitted PPDU. A **default** value has been coded in; hence no change can be made. The value is **12 Symbols**.

Phy SHR Duration – The duration of the synchronization header (SHR) in symbols for the current PHY. The Values available are **3, 7, 10, and 40**. By default Phy SHR Duration is **3**.

Phy Symbol Per Octet - The number of symbols per octet for the current PHY. The Values available are **0.4, 1.6, 2, 8**. By **default** phy symbol per octet is **0.4**.

CCA mode

The following are the available options under CCA mode,

- **Carrier Sense Only (by default this options is selected).**

CCA (Clear Channel Assessment) shall report a busy medium only upon the detection of a signal compliant with this standard with the same modulation and spreading characteristics of the PHY that is currently in use by the device. This signal may be above or below the ED threshold.

- **Energy Detection**

CCA shall report a busy medium upon detecting any energy above the ED threshold.

- **Carrier Sense with Energy Detection**

CCA shall report a busy medium using a logical combination of Detection of a signal with the modulation and spreading characteristics of this standard and Energy above the ED threshold, where the logical operator may be AND/OR.

Receiver Sensitivity –Threshold input signal power that yields a specified PER (packet error rate) conditions PSDU (PHY service data unit) length = 20 octets PER less than 1% Power measured at antenna terminals. Interference not present The Receiver Sensitivity Value should be less than **0**. Default value is **-85 dbm**.

ED threshold - The receiver ED threshold is intended for use by a network layer as part of a channel selection algorithm. It is an estimate of the received signal power within the bandwidth of the channel. No attempt is made to identify or decode signals on the channel. If the receive signal power is greater than the ED threshold value then the channel selection algorithm will return false. The **ED threshold** Value should be **less than 0**. Default value is **-95 dbm**.

Set Environment Properties

Right click in side of the on the Environment and **click Properties**.

Channels - There are sixteen non-overlapping channel numbered from **11 to 26 for 2450** MHz band in **802.15.4** standard. Default value is **11**.

Frequency (MHz) - This property defines the **frequency** allotted for the channel selected. By default **Frequency (MHz)** used is **2405 MHz**, since the **Channel** value is **11**.

Channel Characteristics - This property defines the **Channel Characteristics** for the **Agent**. It consists of the following options.

- **Fading and Shadowing** (default option).

- **Fading only**
- **Line of Sight**
- **No Path Loss**

Path loss Exponent – Path loss exponent indicates the rate at which the path loss increases with distance. The value depends on the specific propagation environment. The value varies range from 2 to 5. The **default** value is **3.5**. This property will be enabled only for **Fading and Shadowing, Fading only, and Line of Sight**.

Fading Figure m (0.5 to 5) - The parameter ‘m’, the “fading figure”, is defined as the ratio of moments, and the **default** value is **1.0**. The value of $m = 1$ corresponds to Rayleigh fading, and $m > 1$ indicates better fading conditions than Rayleigh and usually indicates Line of Sight available between agent. This value varies from **0.5 to 5**. This property will be enabled only for **Fading and Shadowing, and Fading only**.

Standard Deviation - Shadowing is caused mainly by terrain features of the radio propagation environment. The mathematical model for shadowing is a log-normal distribution with standard deviation of **5 to 12 dB**. This has been found to be the best available data to be taken. The **default** value is **12 db**. This property will be enabled only for **Fading and Shadowing**.

Enable Packet Trace (Optional)

- Click Packet Trace icon in the tool bar. This is used to log the packet details.
- Select the required attributes and click OK. Once the simulation is completed, the file gets stored in the specified location.

Simulate - After creating the **Scenario** the following steps need to be followed,

- **Click on Validate** button.
- **Click on Simulate** button.
- **Select the Simulation End Time** and then **click on “OK”** button to start the **Simulation**.
-

Cognitive Radio Networks

Cognitive Radio Network simulation is available from v7 of NetSim. Cognitive Radio Networks allows you to connect, if required, with Ethernet, Wireless LAN, IP Routing, TCP / UDP and allows users to log packet and event (in NetSim standard version only) traces.

New Experiments

In the **Simulation** menu select → **New** → **Cognitive Radio Networks**. Then Select either Map View or Grid view, and to perform experiments in **Cognitive Radio Networks**. Then the following steps should be followed,

- **Create Scenario**
- **Connect the devices**
- **Set Properties for the devices and the links**
- **Enable Packet Trace and Event Trace (Optional)**
- **Simulate**

Create Scenario

Adding Devices – Cognitive Radio Networks comes with the palette of various devices like, Cognitive Radio CPE, Cognitive Radio Base Station, Switch, Router, Wired Node, Wireless Node, Access point etc. Select the desired devices in the toolbar and click and drop on the environment. To remove devices, right click on the particular device and then click Remove.

Connect the devices

Select the appropriate link in the toolbar and connect the devices by clicking on the device 1 and device 2.

Set the Properties for the devices and the links and enabling packet & event trace

Right click over the devices and then select Properties to set the properties of the links and the devices.

Enable the packet trace to log the packet details by clicking on the packet trace in the toolbar and select the required attributes. Enable the event trace to log the event details by clicking on the event trace in the toolbar and select the required events

Simulate - After creating the **Scenario** the following steps need to be followed,

- Click on Simulate button.
- Select the Simulation End Time and then Click on “OK” button to start the Simulation.

Technical Notes on Cognitive Radio implementation in NetSim:

- CR BS allocates max one symbol per CPE. If your generation rate is more than the data filled in one symbol then allocation fails and it results in zero throughput.
- The first symbol is reserved for CR control frames or any broadcast PDU
- NetSim gives App layer throughput which is lesser than that of MAC layer throughput because of
 - TCP connection establishment
 - ARP set-up
 - Service flow creation CPE-BS and BS-CPE
 - BW request
- To avoid the above effects
 - Generate custom traffic
 - Set DL/UL Ratio as 1:1 so that the BS transmits what ever it receives
 - Run UDP in transport layer
 - Use static ARP
 - Run Simulation for more than 100 sec

Opening Saved Experiments

Open Network

Select **Network** from **Simulation → Open** menu to open saved experiments. The following steps need to be followed,

- **Select the User** (*Note:* This option is available in Admin login only)
- **Select the Network.** There are various options like Internetworks, Legacy Networks, Advanced Wireless Networks, MPLS Networks, BGP Networks, Cellular Networks, Wireless Sensor Networks, Personal Area Networks and Cognitive Radio Networks will be available. Choose based on the saved experiment that needs to be opened.
- **Select the Protocol** (*Note:* This is not applicable for Internetworks and Cognitive Radio Networks)
- **Select the Config File/ Experiment**
 - For Internetworks and Cognitive Radio Networks
 - **Select the Config File** of the particular **Experiment** that needs to be opened using **Browse** button.
 - For Non-Internetworks (*Note:* This is not applicable for Cognitive Radio Networks)
 - **Select the Experiment** that needs to be opened.
- **Click on Ok** button to **open** the specified **Experiment**. **User** can modify the existing **Scenario**, **Simulate** and **Save** it. Else **Click on Cancel** button to **Exit** the screen.

Open Metrics

Select **Metrics** from **Simulation → Open** menu to open metrics results of the saved experiments. The following steps need to be followed,

- **Select the User** (*Note:* This option is available in Admin login only. This option can be used to open experiments created by different users)
- **Select the Network.** There are nine options like Internetworks, Legacy Networks, Advanced Wireless Networks, MPLS Networks, BGP Networks, Cellular Networks,

Wireless Sensor Networks, Personal Area Networks and Cognitive Radio Networks will be available. Choose based on the saved experiment that needs to be opened.

- **Select the Protocol** (*Note:* This is not Applicable for Internetworks and Cognitive Radio Networks)
- **Select the Metrics File/ Experiment**
 - For Internetworks and Cognitive Radio Networks
 - **Select the Metrics File** of the particular **Experiment** that needs to be opened using **Browse** button.
 - For Non-Internetworks (*Note:* This is not applicable for Cognitive Radio Networks)
 - **Select the Experiment** that needs to be opened.
- **Click on Ok** button to **open** the selected **Experiment**. User can modify the existing **Scenario**, **Simulate** and **Save** it. Else **Click on Cancel** button to **Exit** the screen.

Deleting Saved Experiments

Delete

In the **Simulation** menu select **Delete** option to delete the saved **Simulated Experiments**. To delete **Experiments** the user needs to follow the below given steps,

- Click the **Delete/Delete All** button to **enter** into the next window.
 - **Delete –** Click the **Delete** button to delete single experiment.
 - **Delete All –** Click the **Delete All** button to delete all the saved experiments.
- **In the next window,**
 - **Delete –**
 - Select the **Experiment Name** that needs to be deleted.
 - **Delete All –**
 - Select the **User Name** in order to delete the experiments under that user.(*Note:* This option will be available only in Admin login)
 - Click on **Delete** button to delete the **Experiment(s)**, else click on **Cancel** button to **cancel** deletion.
 - After clicking on the **Delete** button a message appears which says “**Are you sure you want to delete this Experiment(s)?**” Click on **Yes** to continue deleting the **Experiment(s)**, else click on **No** to **cancel** deleting.
 - Click on **OK** button in the “**The experiment is deleted**” or “**The experiments have been deleted**” window.

Note:

- **Admin User:** An option called **User Name** is available only in when the User has logged into Admin. This option can be made use for deleting experiments under any User.
- **Local User:** A User who has a normal privilege other than Admin can delete experiments only which are created by that particular user.
- In the case of Internetwork and Cognitive Radio networks the experiments can be saved by the user at any desired local on the computer. These experiments cannot be deleted via NetSim, and must directly deleted from the computer.

Comparing Saved Experiments

Analytics

Under the **Simulation** menu click **Analytics** to view the **Analytics** screen. This module is designed to enable inter and intra protocol comparison for different performance metrics. A graphing engineer facilitates plotting of different performance metrics across the different experiments. This can be used to infer/trend how changes in input parameters affect network performance. An explanation of the use of the different fields in the **Analytics** screen is detailed below,

- **User:** This Option can be made use of to compare different experiments created by a single user. As Admin is a Super User, he/she can compare all the experiments created by all the users. Things that cannot be done are,
 - User to User comparison cannot be made, i.e. two or more different experiments created by two or more different users cannot be compared.
 - Experiments of a particular Network cannot be compared with experiments of other Networks.
- **In Tool Bar,**

There are various tabs available on the top of the **Tool Bar** like **Internetworks**, **Legacy Networks**, **BGP Networks**, **MPLS Networks**, **Advanced wireless Networks**, **Cellular Networks**, **Wireless Sensor Networks**, **Personal Area Networks** and **Cognitive Area Networks**. Click on the particular **Network tab** for comparing the performance of protocols under that Network.

 - For Internetworks and Cognitive Radio Networks
 - **Select the Metrics File** – **Select the Metrics File** to add it onto the Metrics Table by using **Browse** button.
 - For Non-Internetworks (*Note:* Not applicable for Cognitive Radio Networks)
 - **Select the Experiment** - **Experiments** can be selected based on the tab selected. When one of the tabs is selected, all the experiments saved under the particular **Network** will be listed. Click on the Experiment Name to add it onto the Metrics Table.

- **Export to .csv** - Click on **Export to .csv** to export the **Metrics Table** to a “.csv” format. This action button will export the contents to a “.csv” format that is available.
 - **Print** - Click on the **Print** option to print the **Metrics Table**. This action button will fetch the Print dialogue box.
- **The Graph Field**
 - **Select the Metrics** - Select the coordinates for **Y-axis** by clicking on the dropdown menu.
 - **Graph** - Based on the **X-axis** (i.e. **Metrics File/ Experiment** selected) and **Y-axis** (i.e. **Metrics** selected by using the dropdown menu above the **graph**), a **Bar graph** will be plotted.
- **The Metrics Table** - This is the table that is generated when the Experiments are selected using the **Select the Metrics File/ Experiment** option. The following columns are available in the Metrics Table,
 - The first column in the **Metrics Table** consists of the **Check Boxes** that are used to **select** (i.e. by default all the Experiments are selected) or **deselect** the selected **Experiments**. If an Experiment is selected, then that Experiment would be compared along with other Experiments, else the Experiment would not be compared with other Experiments.
 - The second column in the **Metrics Table** “X” button is available. This “X” button is used for deleting the **Experiment**.
 - For **Internetworks** and **Cognitive Radio Networks**, the **columns** are Experiment Name, Packets Transmitted, Packets Errorred, Packets Collided, Bytes Transmitted, Payload Transmitted and Overhead Transmitted.
 - For **Non-Internetworks**, the **columns** are Experiment Name, Utilization(%), Effective Utilization(%), Overhead(%), Loss(%), Unutilized(%), Delay(ms), Queuing Delay(ms), Medium Access Time(ms) and Transmission Time(ms). (*Note:* This is not applicable for Cellular Networks, Wireless Sensor Networks and Cognitive Radio Networks)
 - For **Cellular Networks**, the **columns** are Experiment Name, Call generated, Call processed, Call blocked, Call blocking probability, Number of channel, Max channel

Use, Average channel use, Handover attempt, Successful handover, Unsuccessful handover, and Call dropping probability.

- For **Wireless Sensor Networks**, the **columns** are Experiment Name, Queuing Delay, Routing Delay, Medium Access Time, Transmission Time, Transmitting Energy Consumption, Sleep Energy Consumption and Idle Energy Consumption.

Running Simulation via CLI

Note: This feature is available from v7 onwards. In v7, NetSim can be run via CLI for Internetworks and Cognitive Radio Networks only. For other networks NetSim v7 can only be run via GUI

To run Simulation in NetSim through command line interface (CLI), the following steps have to be followed.

Step 1: App Path

App path is the file location in the system in which the NetSim7.0 has been installed.

The app path can be found out by right clicking the NetSim Shortcut in the desktop and selecting Open file location (In Vista and Windows 7) or Find Target (In Windows XP). The app path will be something like “C:\Program Files\NetSim Standard\bin”, depending on where NetSim is installed.

Note: The path to the bin folder of the app path must be mentioned

Step 2: IO Path

IO path (Input/output Path) is the path where the input and output files of an application are written. This is similar to the temp path of windows OS. For NetSim, the iopath is %temp%\NetSim.

Note: The IO path must have write permission. For example, Windows 7 does not provide write permission for the app path, explicitly. NetSim may exhibit unpredictable behavior if the path does not have write permission.

The io path is the path where the **Configuration.xml** and **Configuration.xsd** of the scenario, that will be simulated, should be present.

App path and IO path can also be same. i.e., Configuration.xml and Configuration.xsd can be placed inside the app path (if the app path has write permission). Otherwise, users can create a folder for IO path and Configuration.xml and Configuration.xsd can be placed inside that folder.

Note: Sample configuration.xml files are available for Internetworks and Cognitive Radio Networks in the <apppath>/Docs/Sample_Configurations folder of the NetSim install directory. Rename the sample file as “Configuration.xml” before running NetSim through CLI. Depending on your system setting, the extension (.xml) might be implicitly present and not visible. In such cases do not add the extension, as that will cause the file to be named as configuration.xml.xml.

To run NetSim through command line, copy the app path where NetSimCore.exe is present and paste it in the command prompt.

```
>cd <app path>
```

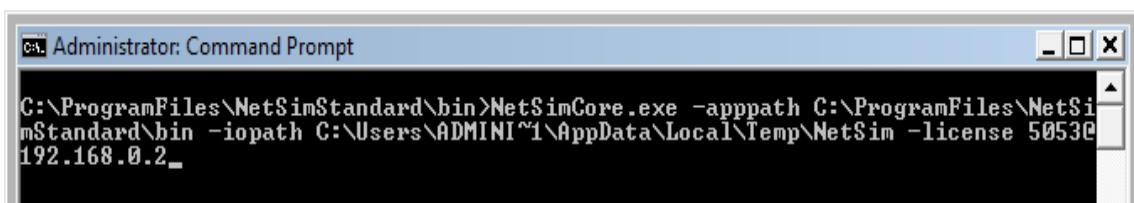
For floating/roaming licenses, type the following in the command prompt

```
>NetSimCore.exe<space>-apppath<space><app path><space>-iopath<space><iopath><space>-license<space>5053@<Server IP Address>
```

where,

- <app path> contains all files of NetSim including NetSimCore.exe
- <iopath> contains Configuration.xml and Configuration.xsd
- 5053 is the port number through which the system communicates with the license server i.e the system in which the dongle is running (for floating license users)
- <Server IP Address> is the ip address of the system where NetSim license server (dongle) is running. Please contact your network administrator / lab in charge to know the IP address of the PC where the NetSim license server is running.

The following screenshot is the example of running NetSim through CLI where the ip address of the NetSim license server is 192.168.0.2



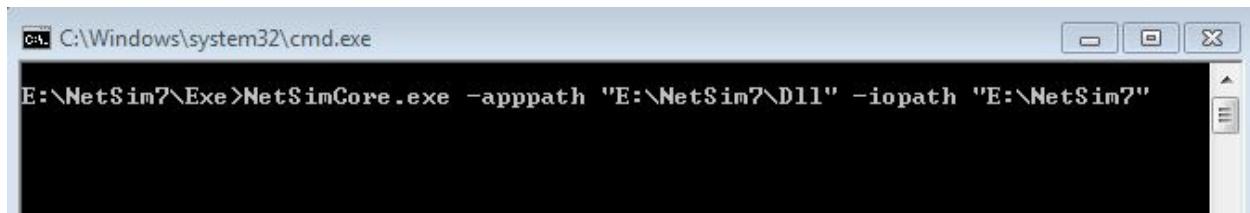
For node-locked licenses, type the following in the command prompt

```
>NetSimCore.exe<space> -apppath<space><app path><space>-iopath<space><iopath><space>
```

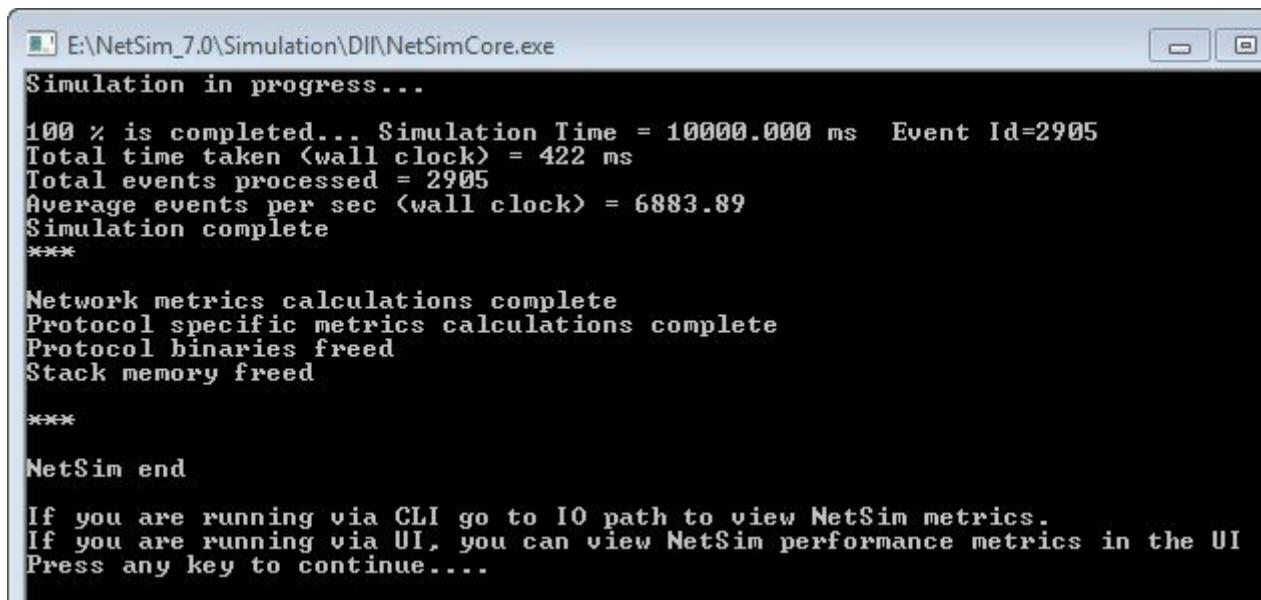
where,

- <app path> contains all files of NetSim including NetSimCore.exe
- <iopath> contains Configuration.xml and Configuration.xsd

The following screenshot is the example of running NetSim through CLI for the node locked license.



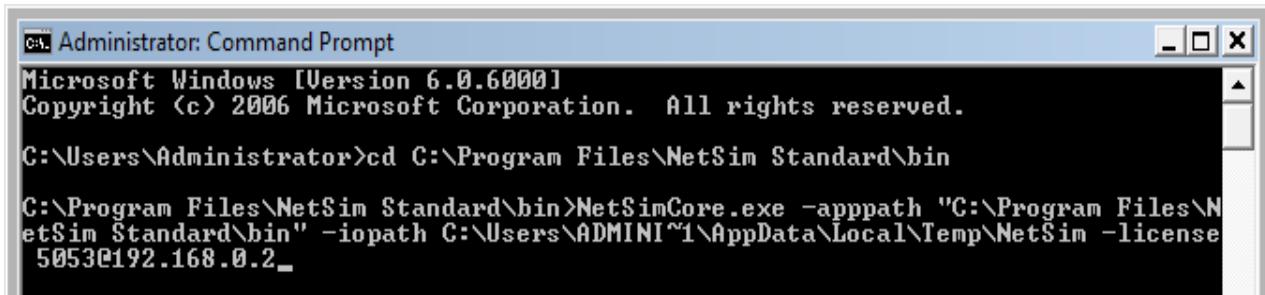
Simulation will be completed successfully and the text files that are requested by the end user in Configuration.xml will be written in the io path.



Note: If the folder name contains white space, then mention the folder path within double quotes while specifying the folder name in the command prompt. For example, if app path contains white

space, then the app path must be mentioned within double quotes in the command prompt as given below.

```
>cd <app path>  
>NetSimCore.exe -apppath <"app path"> -iopath <io path> -license  
5053@<Server IP Address>
```



To know more about the options that are available to run NetSim via CLI, type the following in the command prompt.

```
>cd <app path>  
>NetSimCore.exe -h
```

```
C:\WINDOWS\system32\cmd.exe
D:\Daisy\AppPath of NetSim\NetSim Standard 7.0.3\bin>NetSimCore.exe -h

Usage: NetSimCore [-AppPath PATH] [-IOPath PATH] [-license Port@IP] [-activate]
[-roam flag] [-M flag] [-D ] [-h]

-AppPath -- Specify the path where NetSim dll's are present. This is typically
the bin folder of the install directory.

-IOPath -- Specify the path where input configuration.xml is present. In the
same path output Metrics.txt will be written by NetSim.

-license -- Specify the license server address.
          For Floating License this is of the form - port@ipaddress, where
          port is usually 5053 and ipaddress is the IP address of the PC where the licens
e server is running
          For Node locked license (Internet Activated License) this is of
the form - .

-activate -- Used to activate sw lock. Netsim will ask key to activate.

-roam -- Used to checkout/checkin the roaming license.
        0 to run roaming first(if available without checkout) then float
ing.
        1 to checkout a roaming license.
        -1 to checkin a roaming license.

-M -- Memory leak flag /*For Tetcos debug use only*/
      2 to dump the memory leak
      3 to break at certain allocation number and dump memory leak at
end.
      Example:
      -M 2  dumps the memory leak at the end.
      -M 3  Ask for allocation number and dump the memory leak at end.

-D -- Break the simulation at a certain event id
      Example:
      -D 899 will break the simulation when 899th event is triggered.
      -D -1 will only print the line number and file name in event tra
ce
-h -- Display the help
```

```
D:\Daisy\AppPath of NetSim\NetSim Standard 7.0.3\bin>
```

Simulation Configuration file

Let's see under the hood to know how NetSim is working.

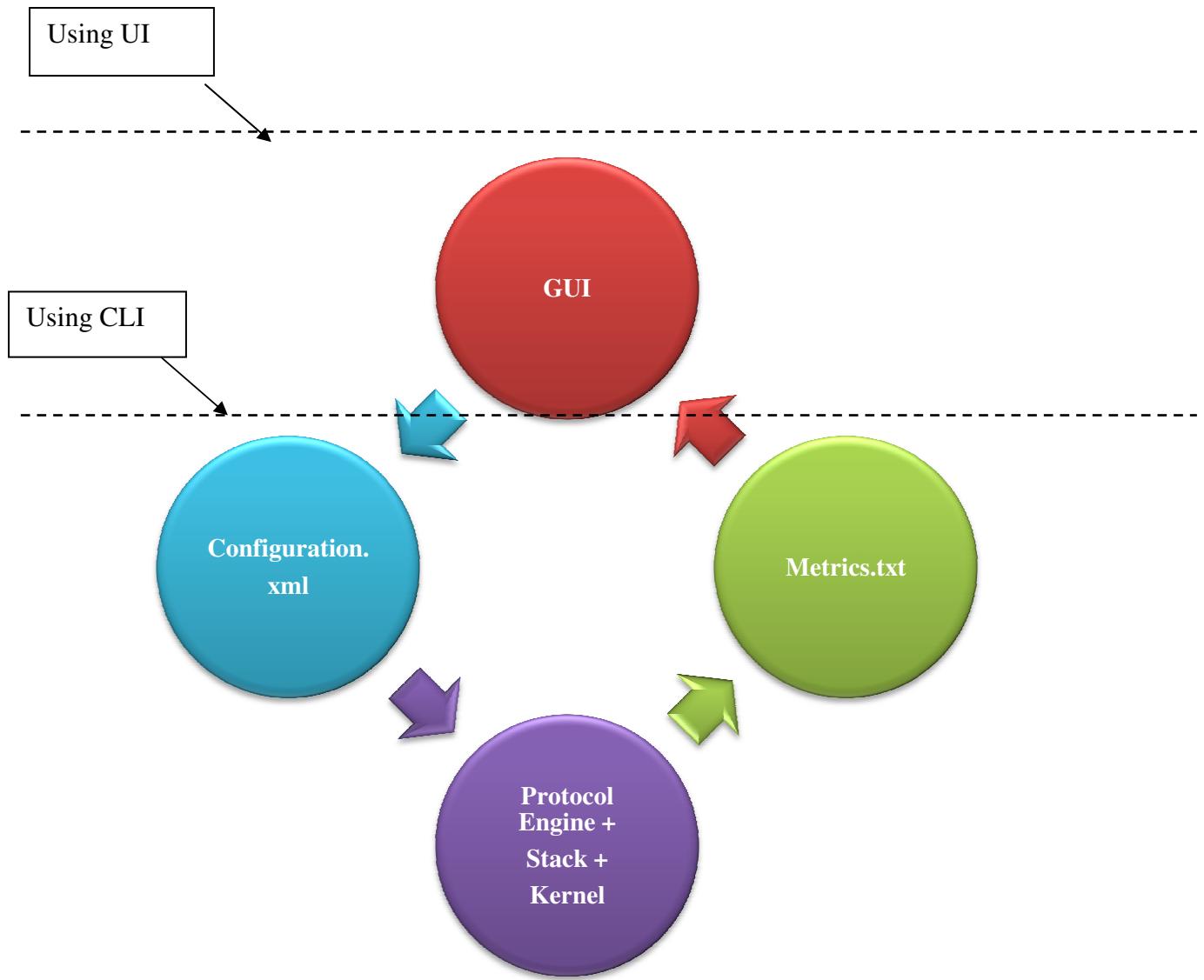


Fig.1. NetSim Architectural Overview

To model a scenario in order to generate metrics in NetSim, GUI will write all the details about the devices used in the scenario and its properties, the links used and their properties, the properties of the environment being used, etc in **Configuration.xml**.

The back-end engine that contains dll's and NetSimCore.exe will read this Configuration.xml, execute the simulation and write output metrics files (in .txt format) to the IO path. Then, the GUI will display the metrics based on the text files written by the backend.

In order to run NetSim through command line (CLI), the user has to create the Configuration.xml furnishing all the details about the devices, links and the environment of the desired scenario.

How to use Visual Studio to edit the Configuration file?

To edit the Configuration.xml, xml editor is required. There are various xml editors available in the market. Visual Studio 2005 is one of the xml editors that can be used to edit the Configuration.xml with efficacy.

This section shows limelight on how to use Visual Studio to edit the Configuration.xml.

XML View

XML view provides an editor for editing raw XML and provides *IntelliSense* and *color coding*.

After you type the element name and press the CTRL+ SPACE, you will be presented with a list of attributes that the element supports. This is known as “IntelliSense”. Using this feature, you can select the options that are required to create the desired scenario.

Color coding is followed to indicate the elements and the attributes in a unique fashion.

The following screenshot displays the Configuration.xml which is opened through the Visual Studio.

Configuration.xml - Microsoft Visual Studio

File Edit View Project Debug XML Tools Test PurifyPlus Window Community Help

Configuration.xml

```
<?xml version="1.0" encoding="UTF-8"?><!--Always mention xml schema file in first
```

To reformat it, click on “Reformat Selection” icon as shown below.

Configuration.xml* - Microsoft Visual Studio

File Edit View Project Debug XML Tools Test PurifyPlus Window Community Help

Reformat Selection

```
<?xml version="1.0" encoding="UTF-8"?>
<!--Always mention xml schema file in first line of configuration.xml-->
<TETCOS_NETSIM xmlns:ns0="http://www.w3.org/2001/XMLSchema-instance" ns0:noNamesp
  <EXPERIMENT_INFORMATION>
    <VERSION>NetSimStandard</VERSION>
    <USER>Admin</USER>
    <NAME>hgf</NAME>
    <DATE>Wed Jul 31 11:01:05 GMT+05:30 2013</DATE>
    <COMMENT>--</COMMENT>
  </EXPERIMENT_INFORMATION>
  <GUI_INFORMATION>
    <!-- The following three sections are used only for the GUI -->
    <ENVIRONMENT_VIEWTYPE>Grid View</ENVIRONMENT_VIEWTYPE>
    <MAP_POSITION>5478 3269</MAP_POSITION>
    <ZOOM_LEVEL>1</ZOOM_LEVEL>
```

```

<?xml version="1.0" encoding="utf-8"?>
<!--Config file-->
<!--Always mention xml schema file in first line of configuration.xml-->
<ETCOS_NETSIM xmlns:ns0="http://www.w3.org/2001/XMLSchema-instance" ns0:noNamespaceSchemaLocation="Configuration.xsd">
<EXPERIMENT_INFORMATION>
    <VERSION>NetSim Standard 7.0</VERSION>
    <USER>Admin</USER>
    <NAME>NetSim</NAME>
    <DATE>Thu Nov 05 14:37:02 IST 2012</DATE>
    <COMMENT>NetSim Configuration File</COMMENT>
</EXPERIMENT_INFORMATION>
<NETWORK_CONFIGURATION>
    <!--The following section of config file is used to set the device properties-->
    <DEVICE_CONFIGURATION DEVICE_COUNT="4">
        <!--Set Device properties-->
        <DEVICE DEVICE_NAME="Node1" DEVICE_ID="1" INTERFACE_COUNT="1" TYPE="NODE">
            <!--Set Pos 3D properties-->
            <POS_3D X-POS="157" Y-POS="110" Z-POS="0" />
            <!--Set Interface properties-->
            <INTERFACE ID="1" INTERFACE_TYPE="Ethernet">
                <!--Configure each layer-->
                <!--Set Physical_Layer properties-->
                <LAYER TYPE="PHYSICAL_LAYER" CONNECTION_MEDIUM="WIRED">
                    <!--Set Protocol properties-->
                    <PROTOCOL NAME="ETHERNET" SETPROPERTY="true">
                        <!--Set protocol properties-->
                        <PROTOCOL_PROPERTY ETHERNET_STANDARD="IEEE802.3u" />
                    </PROTOCOL>
                </LAYER>
                <!--Configure each layer-->
                <!--Set DataLink_Layer properties-->
                <LAYER TYPE="DATALINK_LAYER">

```

Schema View

An XML schema describes the structure of an XML document. The XML Schema language is also referred to as XML Schema Definition (XSD). The purpose of an XML Schema is to define the legal building blocks of an XML document.

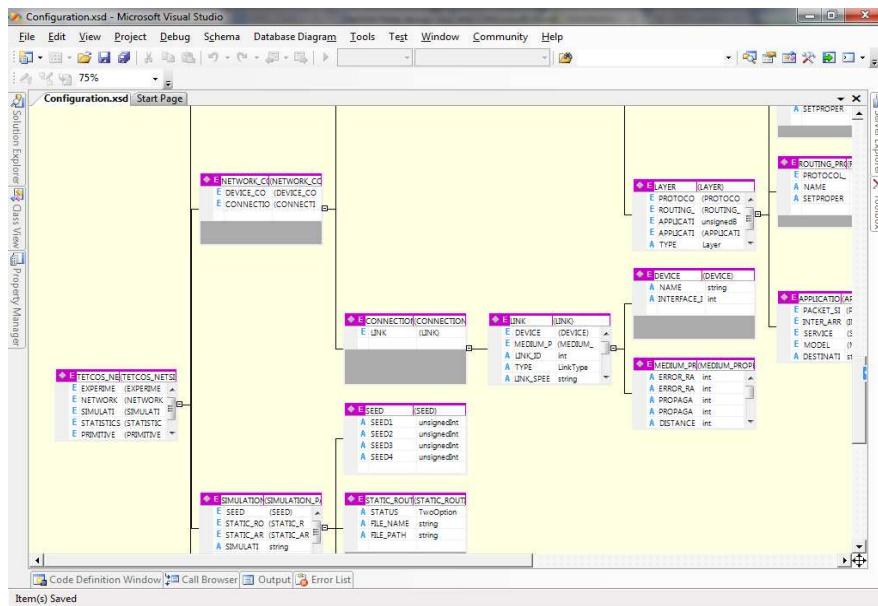
The schema of Configuration.xml is Configuration.xsd and it is read-only.

Configuration.xsd:

- defines elements that can appear in the configuration.xml
- defines attributes that can appear in the configuration.xml
- defines which elements are child elements
- defines the order of child elements
- defines the number of child elements
- defines whether an element is empty or can include text
- defines data types for elements and attributes
- defines default and fixed values for elements and attributes

Thus, configuration.xsd helps you to choose the valid options in the configuration.xsd.

The following screenshot displays the configuration.xsd which is opened through the Visual Studio.



Data View

Data view provides a data grid that can be used to modify .xml files.

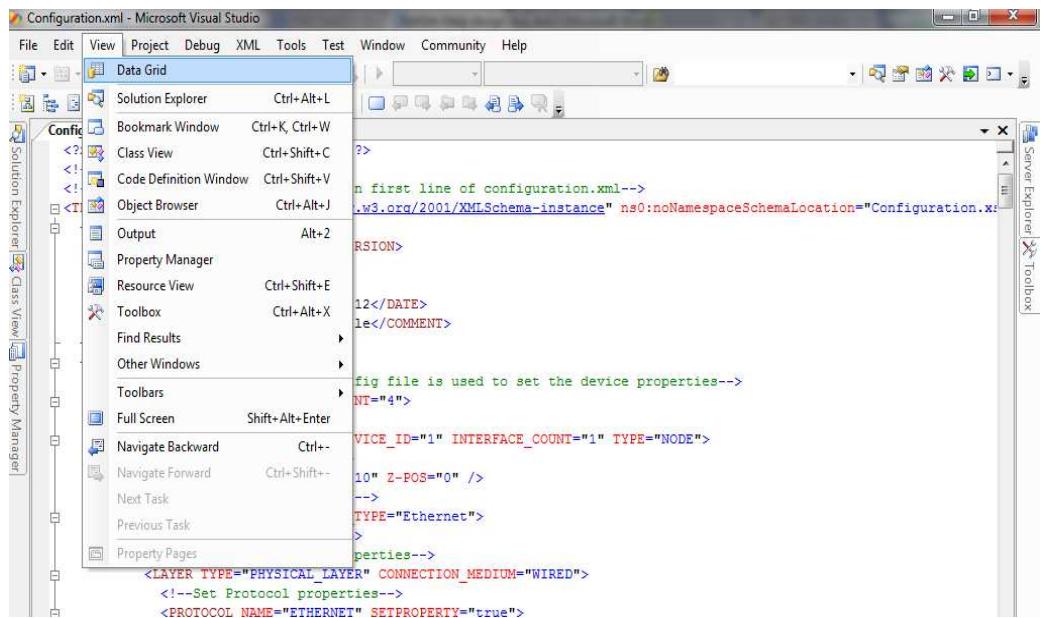
In Data view you can edit existing data tables. Only the content (but not the tags and structure) in an XML file can be edited in Data view.

There are two separate areas in Data view: **Data Tables** and **Data**. The **Data Tables** area is a list of relations defined in the XML file, in the order of its nesting (from the outermost to the innermost).

The **Data** area is a data grid that displays data based on the selection in the Data Tables area.

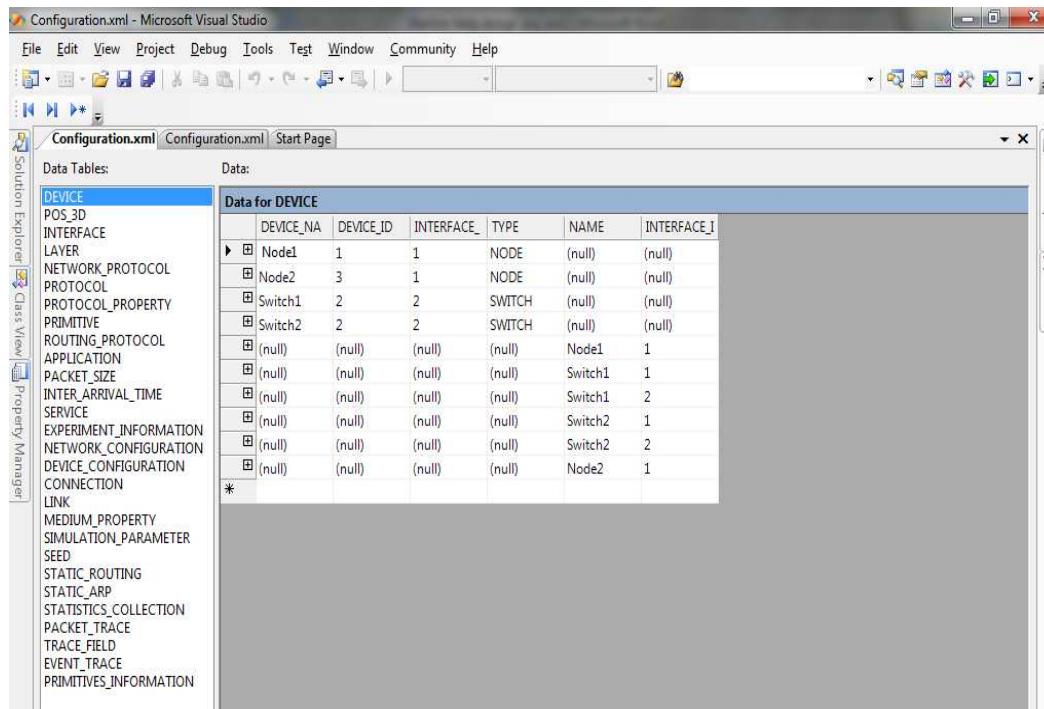
Note: If XML is not structured, then data trying to switch to Data view will generate the following message: "Although this document is well formed, it contains structure that Data View cannot display."

To get the Data View, click View → Data Grid as shown below.



```
?>
<n first line of configuration.xml-->
<!w3.org/2001/XMLSchema-instance" ns0:noNamespaceSchemaLocation="Configuration.xs
RSION>
12</DATE>
le</COMMENT>
fig file is used to set the device properties-->
NT="4">
VICE_ID="1" INTERFACE_COUNT="1" TYPE="NODE">
10" Z-POS="0" />
-->
TYPE="Ethernet">
>
properties-->
<LAYER_TYPE="PHYSICAL_LAYER" CONNECTION_MEDIUM="WIRED">
<!--Set Protocol properties-->
<PROTOCOL NAME="ETHERNET" SETPROPERTY="true">
```

The following screenshot shows the data view of the configuration.xml.



DEVICE	DEVICE_NA	DEVICE_ID	INTERFACE_	TYPE	NAME	INTERFACE_I
POS_3D	Node1	1	1	NODE	(null)	(null)
INTERFACE	Node2	3	1	NODE	(null)	(null)
LAYER	Switch1	2	2	SWITCH	(null)	(null)
NETWORK_PROTOCOL	Switch2	2	2	SWITCH	(null)	(null)
PROTOCOL	(null)	(null)	(null)	(null)	Node1	1
PROTOCOL_PROPERTY	(null)	(null)	(null)	(null)	Switch1	1
PRIMITIVE	(null)	(null)	(null)	(null)	Switch1	2
ROUTING_PROTOCOL	(null)	(null)	(null)	(null)	Switch2	1
APPLICATION	(null)	(null)	(null)	(null)	Switch2	2
PACKET_SIZE	*					
INTER_ARRIVAL_TIME						
SERVICE						
EXPERIMENT_INFORMATION						
NETWORK_CONFIGURATION						
DEVICE_CONFIGURATION						
CONNECTION						
LINK						
MEDIUM_PROPERTY						
SIMULATION_PARAMETER						
SEED						
STATIC_ROUTING						
STATIC_ARP						
STATISTICS_COLLECTION						
PACKET_TRACE						
TRACE_FIELD						
EVENT_TRACE						
PRIMITIVES_INFORMATION						

Sections of Configuration file

There are five sections in configuration.xml, namely,

- EXPERIMENT_INFORMATION
- NETWORK_CONFIGURATION
- SIMULATION_PARAMETER
- STATISTICS_COLLECTION
- PRIMITIVES_INFORMATION

EXPERIMENT_INFORMATION:

This section contains the details about the user credentials, such as the user mode (Admin or Exam or Practice), experiment name, date on which the experiment is created and the comments about the experiment. This section plays a significant role while running NetSim through GUI.

NETWORK_CONFIGURATION:

This section is used to configure the devices and the links of the desired network at the each layer of the OSI stack. NETWORK_CONFIGURATION consists of DEVICE_CONFIGURATION and CONNECTION. DEVICE_CONFIGURATION configures the devices in the desired network while the CONNECTION configures the links in the desired network.

SIMULATION_PARAMETER:

Simulation time, seed values, static routing and static ARP are described in this section. The text files illustrating the static routing and static ARP can be obtained by enabling the corresponding tags in the configuration.xml.

STATISTICS_COLLECTION:

The packet trace and the event trace can be observed in the text files which are created by enabling the tags in this section. The required fields of the packet trace can be enabled in the PACKET_TRACE while the event trace can be enabled in the EVENT_TRACE of this section.

PRIMITIVES_INFORMATION:

The primitives of each protocol involved in the network can be specified in this section. The dll name and its path, the function handling the operation of each primitive are mentioned in the PRIMITIVE tag of each PROTOCOL tag in this section.

Troubleshooting

While running NetSim independent of UI for the scenarios described in the Configuration file, you may bump into few problems.

Note: While running NetSim independent of UI, try to ensure that there are no errors in the Configuration.xml file. The file, ConfigLog.txt, written to the windows temp path would show errors, if any, found by NetSim's config parser.

This section discusses some common issues and solutions:

Problem 1: Specific attributes in the Configuration file are highlighted with zigzag lines

If invalid input is given in the Configuration file, then the corresponding attribute is highlighted in zigzag lines as shown in the figure given below.

```
<LAYER TYPE="NETWORK_LAYER">
    <!--Set Network_Protocol properties-->
    <!--Configure each layer-->
    <!--Set Physical_Layer properties-->
    <NETWORK_PROTOCOL IP_ADDRESS="192.168.0.4" NAME="IPV4" SUBNI
    <PROTOCOL NAME="ARP" SETPROPERTY="true">
        <PROTOCOL_PROPERTY ARP_RETRY_INTERVAL="10" ARP_RETRY_LIMIT="10" />
    </PROTOCOL>
    </LAYER>
</INTERFACE>
<INTERFACE ID="2" INTERFACE_TYPE="ETHERNET">
    <!--Configure each layer-->
    <!--Set Physical_Layer properties-->
    <LAYER TYPE="PHYSICAL_LAYER" CONNECTION_MEDIUM="Wireless">
        <!--Set Protocol properties-->
        <PROTOCOL NAME="ETHERNET" SETPROPERTY="true">
            <!--Set protocol properties-->
            <PROTOCOL_PROPERTY ETHERNET_STANDARD="IEEE802.3u" />
        </PROTOCOL>
    </LAYER>
```

Solution:

To resolve this issue mouse over the corresponding attribute, in order to get the tool tip that furnishes the details about the valid input for that attribute.

Note: If the schema file and the configuration file are not present in the same folder, the zigzag lines won't appear. So place the Configuration file and Schema File in the same location or change the path of schema file in the configuration file.

Problem 2: *Simulation runs successfully but errors are provided in the ConfigLog.txt*

Since, NetSim will crash if all the required parameters are not configured, NetSim addresses this issue by “enabling” default values where ever required.

However, the user is made to be aware of the missing parameters in the Configuration file by mentioning the errors in the ConfigLog.txt.

Example:

Error produced in the ConfigLog.txt when the value for the Update timer for router is not specified in the Configuration file is given below.

Attributes "UPDATE_TIMER" not found in line number 349.

Tag path:

/TETCOS_NETSIM/NETWORK_CONFIGURATION/DEVICE_CONFIGURATION/DEVICE [3] /LAYER [2]/ROUTING_PROTOCOL/PROTOCOL_PROPERTY

Solution: User has to specify the missing parameters in the Configuration file which are mentioned in the ConfigLog.txt, in order to obtain accurate metrics for the desired scenario.

Problem 3: *Simulation is interrupted and the error is displayed in command line*

Example: If the user defined function in the PRIMITIVES_INFORMATION of the Configuration file is incorrect/ missing, then the simulation is interrupted and the corresponding function name will be specified in the command prompt as shown below.

In 1330 line of StackInternal.c file following error occurs

*fn_NetSim_UDP_Init: The specified procedure could not be found.
Error number=127*

Solution: User has to specify the exact function name in the PRIMITIVES_INFORMATION of the Configuration file, in order to simulate the desired scenario.

Problem 4: *Simulation is interrupted and the metrics are not generated*

This issue arises mainly while linking custom code in NetSim. i.e., when the custom code is having any bug, the flow of execution of NetSim gets disturbed and this leads to stand still condition.

Solution: The bugs can be fixed by setting the break points for debugging of custom code and the following steps can be used to fix the issue.

Step 1:

Since the packets flow per events in NetSim, **enable event trace** either in the Configuration file or via the UI. Analyze the Event Trace to identify at which event id the stand still condition occurs.

Step 2:

Run NetSim independent of UI by mentioning the following things in the command prompt.

```
cd <apppath>
-apppath <NetSim install directory> -iopath <IO Path> -license 5053@<License
Server IP Address> -d
```

Example:

```
C:\Program Files\NetSim Standard\bin>NetSimCore.exe -apppath "C:\Program Files\NetSim
Standard\bin" -iopath "D:\NetSim" -license 5053@192.168.0.10 -d
```

Step 3:

Command prompt will ask for break point while debugging.

Set the event id from which the code flow has to be analyzed as the break point in the command prompt as shown below.

```
C:\WINDOWS\system32\cmd.exe - NetSimCore.exe -apppath "C:\Program Files\NetSim Stan... [x]
C:\Program Files\NetSim Standard\bin>NetSimCore.exe -apppath "C:\Program Files\NetSim Standard\bin" -iopath "D:\Daisy\Internal Testing\NetSim 7.0.7 Internal Testing\Config" -license 5053@192.168.0.10 -d
Breakpoint is set...
Enter the event id where u want to break:-1
Gathering NetSim license information
Server = . Roam Enable<1>/Disable<0>=0
Product = netsim_std_1
Version = ?
```

Errors will be thrown in the command prompt, if any as shown below.

```
C:\WINDOWS\system32\cmd.exe - NetSimCore.exe -apppath "C:\Program Files\NetSim Stan... [x]
Version = ?
Product = netsim_std_?
Version = ?
Output of license files --- netsim>std>?>0>rlm_hw>1111111>
NetSim license validated

***  

NetSim start  

Network Stack loaded  

Entering Network Stack  

Initializing simulation  

C:\Program Files\NetSim Standard\bin\libEthernet.dll>>fn_NetSim_Ethernet_Init  

fn_NetSim_Ethernet_Run  

fn_NetSim_Ethernet_Trace  

fn_NetSim_Ethernet_FreePacket  

fn_NetSim_Ethernet_CopyPacket  

fn_NetSim_Ethernet_Metrics  

fn_NetSim_Ethernet_Finish  

fn_NetSim_Ethernet_Configure  

fn_NetSim_Ethernet_ConfigurePrimitives  

fn_NetSim_Ethernet_ConfigPacketTrace  

fn_NetSim_Ethernet_WritePacketTrace  

C:\Program Files\NetSim Standard\bin\libARP.dll>>fn_NetSim_ARP_Init  

fn_NetSim_ARP_Run  

fn_NetSim_ARP_Trace  

fn_NetSim_ARP_FreePacket  

fn_NetSim_ARP_CopyPacket  

fn_NetSim_ARP_Metrics  

fn_NetSim_ARP_Finish  

fn_NetSim_ARP_Configure  

fn_NetSim_ARP_ConfigurePrimitives  

fn_NetSim_ARP_ConfigPacketTrace  

fn_NetSim_ARP_WritePacketTrace  

C:\Program Files\NetSim Standard\bin\libTCP.dll>>fn_NetSim_TCP_Init  

fn_NetSim_TCP_Run  

fn_NetSim_TCP_Trace  

fn_NetSim_TCP_FreePacket  

fn_NetSim_TCP_CopyPacket  

fn_NetSim_TCP_Metrics  

fn_NetSim_TCP_Finish  

fn_NetSim_TCP_Configure  

fn_NetSim_TCP_ConfigurePrimitives  

fn_NetSim_TCP_ConfigPacketTrace  

fn_NetSim_TCP_WritePacketTrace  

C:\Program Files\NetSim Standard\bin\libRouting.dll>>fn_NetSim_RIP_Init  

fn_NetSim_RIP_Run  

fn_NetSim_RIP_Trace  

fn_NetSim_RIP_FreePacket  

fn_NetSim_RIP_CopyPacket  

fn_NetSim_RIP_Metrics  

fn_NetSim_RIP_Finish  

fn_NetSim_RIP_Configure  

fn_NetSim_RIP_ConfigurePrimitives  

fn_NetSim_RIP_ConfigPacketTrace  

fn_Netsim_RIP_WritePacketTrace  

Error in config file---  

More number of Interface is configured compare to setted interface count.  

C:\Program Files\NetSim Standard\bin\libUDP.dll>>fn_NetSim_UDP_Init
```

Step 4:

Fix the error and then run NetSim independent of UI with debug mode as in Step 2. Check whether simulation runs successfully by checking Event Trace, Metrics and ConfigLog files.

Note: Set the breakpoint as -1 when the simulation has to run without any pause in any event.

Problem 5: *Simulation does not commence and error is displayed at the command prompt. Also, Zigzag lines appearing at the tag specifying the Layer in the Configuration file*

This issue arises mainly when the closing tag is not specified correctly for a particular layer in the Configuration file.

Example: If the closing tag is not specified for the Data link Layer, then the zigzag lines appear at the starting tags of Data link Layer and the Network Layer.

```
<!--Configure each layer-->
<!--Set DataLink_Layer properties-->
<LAYER TYPE="DATALINK_LAYER">
    <!--Set Protocol properties-->
    <PROTOCOL NAME="ETHERNET" SETPROPERTY="true" MAC_ADDRESS="AAAAAAAAAAAA">
        <!--Set Protocol properties-->
        <PROTOCOL_PROPERTY />
    </PROTOCOL>
    <!--Configure each layer-->
    <!--Set Network_Layer properties-->
    <LAYER TYPE="NETWORK_LAYER">
        <!--Set Network_Protocol properties-->
        <NETWORK_PROTOCOL IP_ADDRESS="192.168.0.5" NAME="IPV4" SUBNET_MASK="255.255.255.
        <PROTOCOL NAME="ARP" SETPROPERTY="true">
            <PROTOCOL_PROPERTY ARP_RETRY_INTERVAL="10" ARP_RETRY_LIMIT="3" />
        </PROTOCOL>
    </LAYER>
</INTERFACE>
```

When NetSim is made to run through CLI, then the following error gets displayed in the command prompt.

```

C:\Windows\system32\cmd.exe - NetSimCore.exe -apppath "E:\NetSim7\Dll" -iopath "E:\NetSim7" ...
Version = 7
Product = netsim_std_4
Version = 7
Product = netsim_std_5
Version = 7
Product = netsim_std_6
Version = 7
Product = netsim_std_7
Version = 7
Output of license files --- netsim>std>?>0>r1m_hw>1111111>
NetSim license validated

***  

NetSim start  

Network Stack loaded  

Initializing simulation  

file:///E:/NetSim7/Configuration.xml:48: parser error : Opening and ending tag mismatch: LAYER line 32 and INTERFACE  

    </INTERFACE>  

file:///E:/NetSim7/Configuration.xml:173: parser error : Opening and ending tag mismatch: INTERFACE line 20 and DEVICE  

    </DEVICE>  

file:///E:/NetSim7/Configuration.xml:365: parser error : expected '>'
```

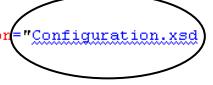
Solution: The bug can be fixed by setting the closing tag correctly in the Configuration file

Problem 6: Zigzag lines appearing at configuration.xsd in the Configuration file

This issue arises when the schema and the configuration file are not in the same folder.

```

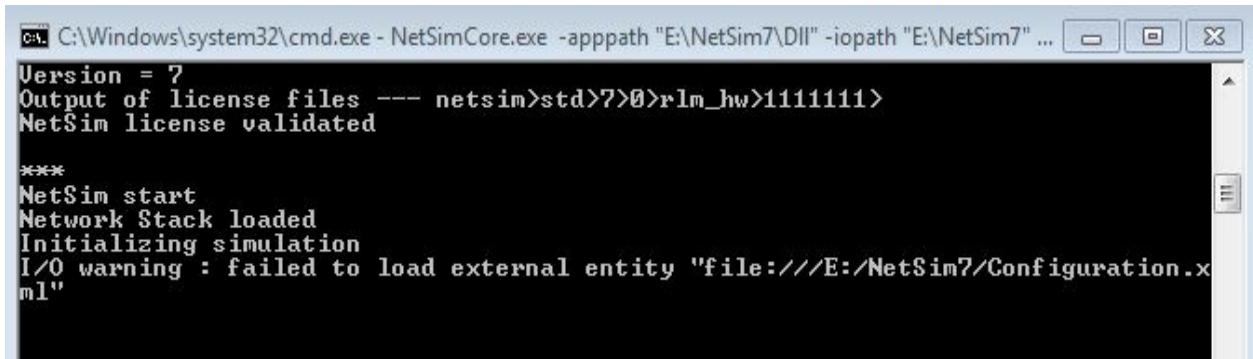
--Config file-->
--Always mention xml schema file in first line of configuration.xml-->
ETCOS_NETSIM xmlns:ns0="http://www.w3.org/2001/XMLSchema-instance" ns0:noNamespaceSchemaLocation="Configuration.xsd"
<EXPERIMENT_INFORMATION>
    <VERSION>NetSim Standard 7.0</VERSION>
    <USER>Admin</USER>
    <NAME>NetSim</NAME>
    <DATE>Thu Nov 05 14:37:02 IST 2012</DATE>
    <COMMENT>NetSim Configuration File</COMMENT>
</EXPERIMENT_INFORMATION>
<NETWORK_CONFIGURATION>
    <!--The following section of config file is used to set the device properties-->
    <DEVICE_CONFIGURATION DEVICE_COUNT="8">
        <!--Set Device properties-->
        <DEVICE DEVICE_NAME="MS1" DEVICE_ID="1" INTERFACE_COUNT="1" TYPE="MOBILE_STATION">
            <!--Set Pos_3D properties-->
            <POS_3D X-POS="157" Y-POS="110" Z-POS="0" />
            <!--Set Interface properties-->
            <INTERFACE ID="1" INTERFACE_TYPE="CDMA">
                <!--Configure each layer-->
                <!--Set Physical_Layer properties-->
                <LAYER TYPE="PHYSICAL_LAYER" CONNECTION_MEDIUM="WIRELESS">
                    <!--Set Protocol properties-->
                    <PROTOCOL NAME="CDMA" SETPROPERTY="true">
                        <!--Set Protocol properties-->
                    <!--
                    Unit-
                    TRANSMITTED POWER = W
```



Solution: The bug can be fixed by placing the Configuration file and schema in the same folder.

Problem 7: Simulation does not commence and the error is displayed at the command prompt

Example:



```
C:\Windows\system32\cmd.exe - NetSimCore.exe -apppath "E:\NetSim7\Dll" -iopath "E:\NetSim7" ...
Version = 7
Output of license files --- netsim>std>?>0>r1m_hw>1111111>
NetSim license validated

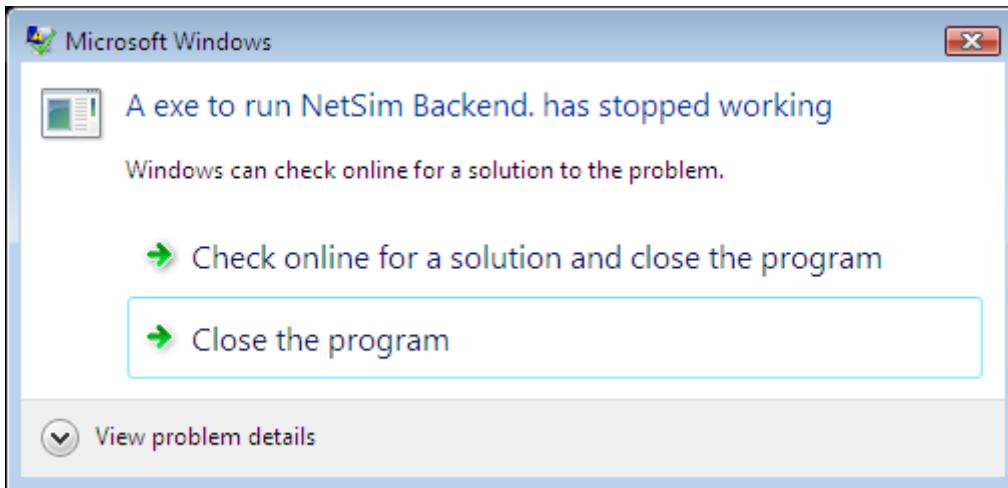
*** 
NetSim start
Network Stack loaded
Initializing simulation
I/O warning : failed to load external entity "file:///E:/NetSim7/Configuration.xml"
```

This problem arises when the Configuration.xml (file name and extension should be correct) is not available in the specified io path.

Solution: The bug can be fixed by ensuring whether file name and extension of Configuration.xml are correct.

Problem 8: Simulation terminates and exhibits unpredictable behavior. An error message stating, “An exe to run NetSim backend has stopped working” is thrown

Example:



This problem arises if there is any flaw in the Configuration.xml or in the dll.

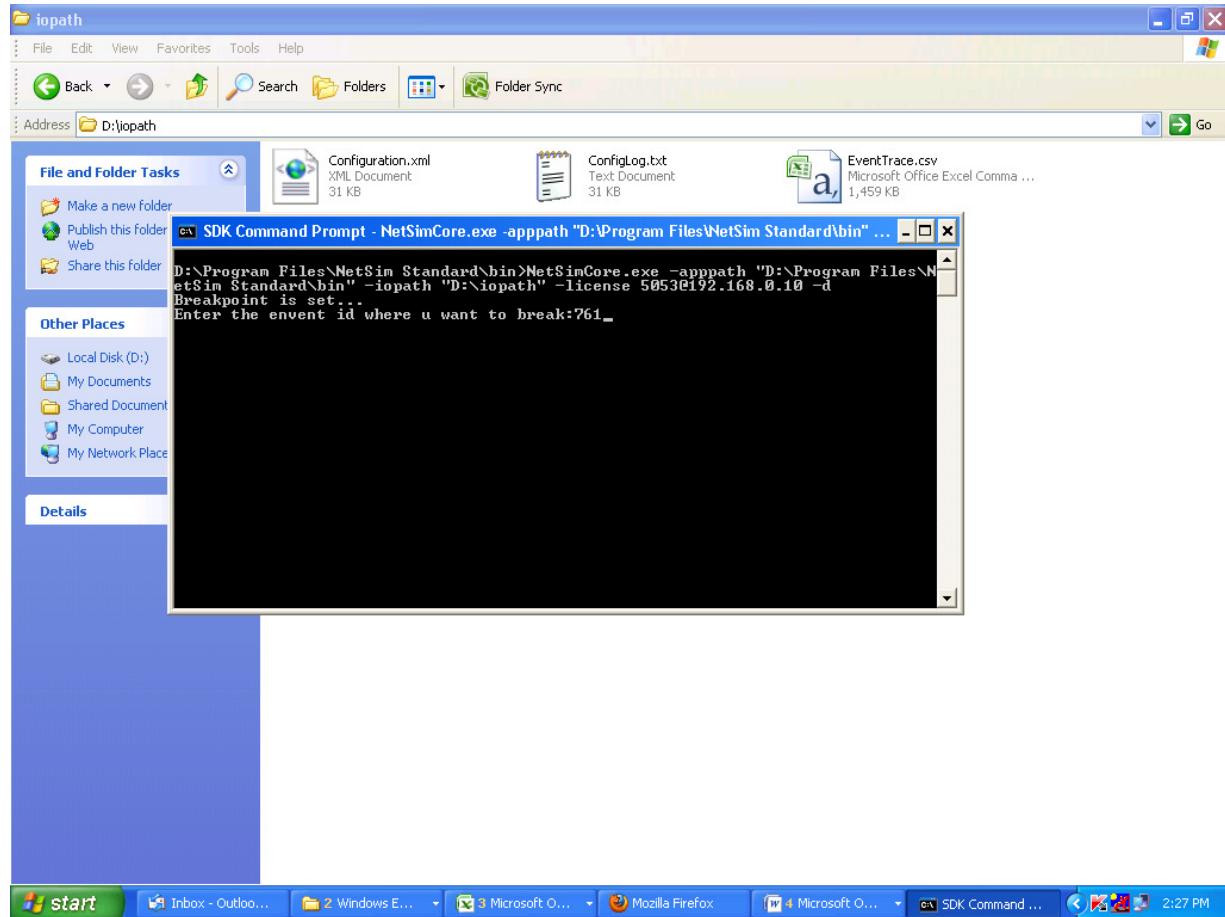
Solution: Check whether the desired scenario has been configured properly in the Configuration.xml.

Enable the event trace in the Configuration.xml so that the event details can be logged.

Run NetSim via CLI again by specifying the event id in order to debug the code, as given below.

```
>NetSimCore.exe<space> -apppath<space><app path><space>-iopath<space><io path><space> -license<space>5053@<Server IP Address><space> -d
```

Event id will be asked in the command prompt. Enter the event id from where the code has to be analyzed, as given below.



Now, user can debug the code, so that the flaw in the Configuration.xml or in the dll can be removed, easily.

Problem 9: Simulation does not commence."No license for product (-1)" is displayed in the command prompt.

Example:

```
C:\WINDOWS\system32\cmd.exe - NetSimCore.exe -apppath "C:\Program Files\NetSim Stan... □ X
C:\Program Files\NetSim Standard\bin>NetSimCore.exe -apppath "C:\Program Files\NetSim Standard\bin" -iopath "C:\DOCUME~1\Tesing1\LOCALS~1\Temp\NetSim" -license 5053@192.168.0.10
Gathering NetSim license information
Server = license          Roam Enable(1)/Disable(0)=0
Product = netsim_std_1
Version = 7
No license for product <-1>
Product = netsim_std_2
Version = 7
No license for product <-1>
Product = netsim_std_3
Version = 7
No license for product <-1>
Product = netsim_std_4
Version = 7
No license for product <-1>
Product = netsim_std_5
Version = 7
No license for product <-1>
Product = netsim_std_6
Version = 7
No license for product <-1>
Product = netsim_std_7
Version = 7
No license for product <-1>
Output of license files --- netsim>std>7>0>rlm_hw_roam>0000000>
Gathering NetSim license information
Server = 5053@192.168.0.10      Roam Enable(1)/Disable(0)=0
Product = netsim_std_1
Version = 7
All licenses in use <-22>
Product = netsim_std_2
Version = 7
All licenses in use <-22>
Product = netsim_std_3
Version = 7
All licenses in use <-22>
Product = netsim_std_4
Version = 7
All licenses in use <-22>
Product = netsim_std_5
Version = 7
All licenses in use <-22>
Product = netsim_std_6
Version = 7
All licenses in use <-22>
Product = netsim_std_7
Version = 7
All licenses in use <-22>
Output of license files --- netsim>std>7>0>rlm_hw_roam>0000000>
Error in getting license
Press any key to continue...
-
```

Solution:

NetSim works based on the client-server architecture. When NetSim runs in the client machine, it will check for the license in the same machine, first. If license is not available in the same machine, then “No license for product (-1)” will be displayed in the command prompt and the server machine will be checked for the availability of license. If no license is available in the server machine also, then again “No license for product (-1)” will be displayed in the command prompt.

So, if ”No license for product(-1)” is displayed in the command prompt two times, then check in the NetSim license server to know about the availability of license and adjust the number of current users of NetSim, in order to get the license.

Performance Metrics

NetSim provides distinct quantitative metrics at various abstraction levels such as Network Metrics, Link Metrics, TCP Metrics, Application Metrics, etc at the end of simulation. With the help of metrics, users can analyze the behavior of the modeled network and can compare the impact of different algorithms on end-to-end behavior.

The following table lists the various files that will be written in the NetSim install directory/ IO path on completion of simulation.

S.No	File	Contents
1	Metrics.txt	Contains the metrics of the network that is simulated recently .
2	MetricsGraph.txt	Contains the data required to plot the graph between the various parameters in metrics against time
3	LicenseErrorLog.txt	Contains the status of the communication between the NetSim dongle and the client.
4	ConfigLog.txt	This file will be written while reading the Configuration file. Provides errors if there are errors in the configuration file.
5	LogFile.txt	Contains the logs as the control flows across various layers in the Network Stack
6	PacketTrace.txt	Contains the detailed packet information. This file will be written only when Packet Trace is enabled.
7	EventTrace.txt	Contains the information about each event. This file will be written only when Event Trace is enabled.

If NetSim runs via the UI, then the metrics will be displayed automatically at the end of simulation with illustrative pie charts and tables.

If NetSim runs via CLI, then the metrics will be written into Metrics.txt and MetricsGraph.txt.

Introduction:

NetSim allows users to generate trace files which provide detailed packet information useful for performance validation, statistical analysis and custom code de-bugging. Packet Trace logs a set of chosen parameters for every packet as it flows through the network such as arrival times, queuing times, departure times, payload, overhead, errors, collisions etc.

By providing a host of information and parameters of every packet that flows through the network, packet trace provides necessary forensics for users to catch logical errors without setting a lot of breakpoints or restarting the program often. Window size variation in TCP, Route Table Formation in OSPF, Medium Access in Wi-fi, etc, are examples of protocol functionalities that can be easily understood from the trace.

Note: Turning on Packet Trace will slow down the simulation significantly

By default, the packet tracing option is turned off.

How to enable Packet Trace via UI?

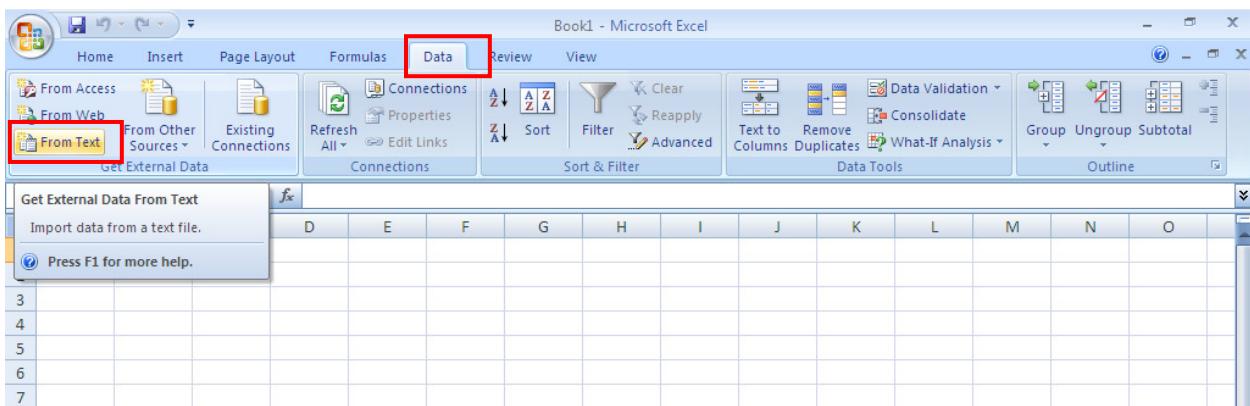
If NetSim runs via the UI, packet trace can be turned on by clicking the Packet Trace icon in the tool bar in the Internetworks menu and selecting the required fields in the packet trace. NetSim will write the packet trace to the specified path during simulation.

How to enable Packet Trace via CLI?

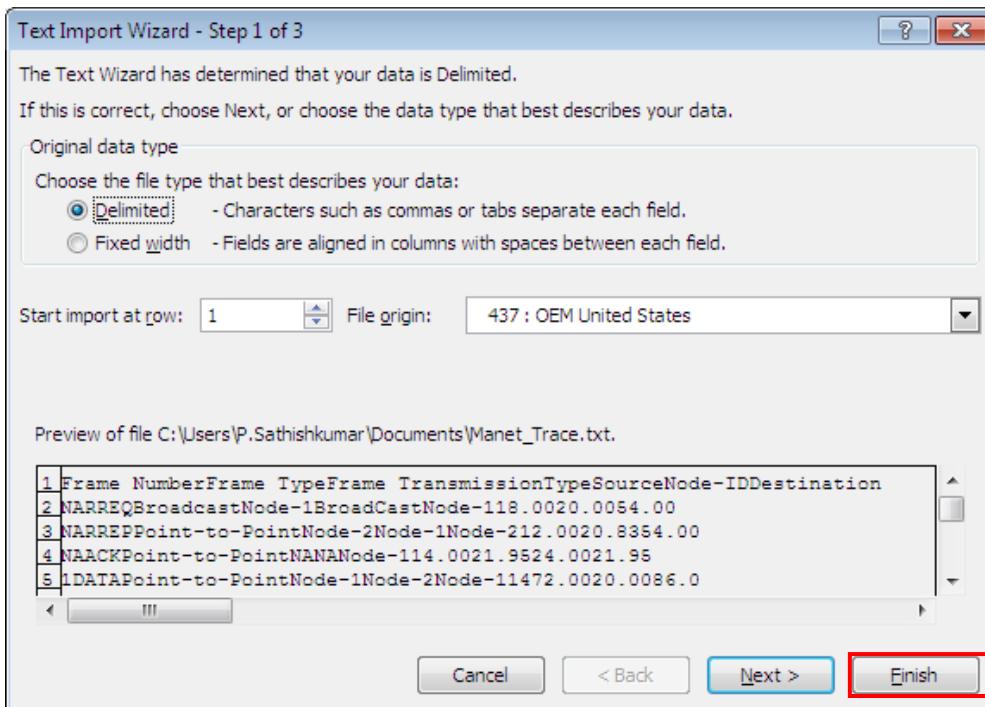
If NetSim runs via CLI, then the packet trace can be turned on by enabling the packet trace in the STATISTICS_COLLECTION tag of the configuration file. NetSim will write the packet trace to the specified path during simulation.

How to import Packet Trace to Excel?

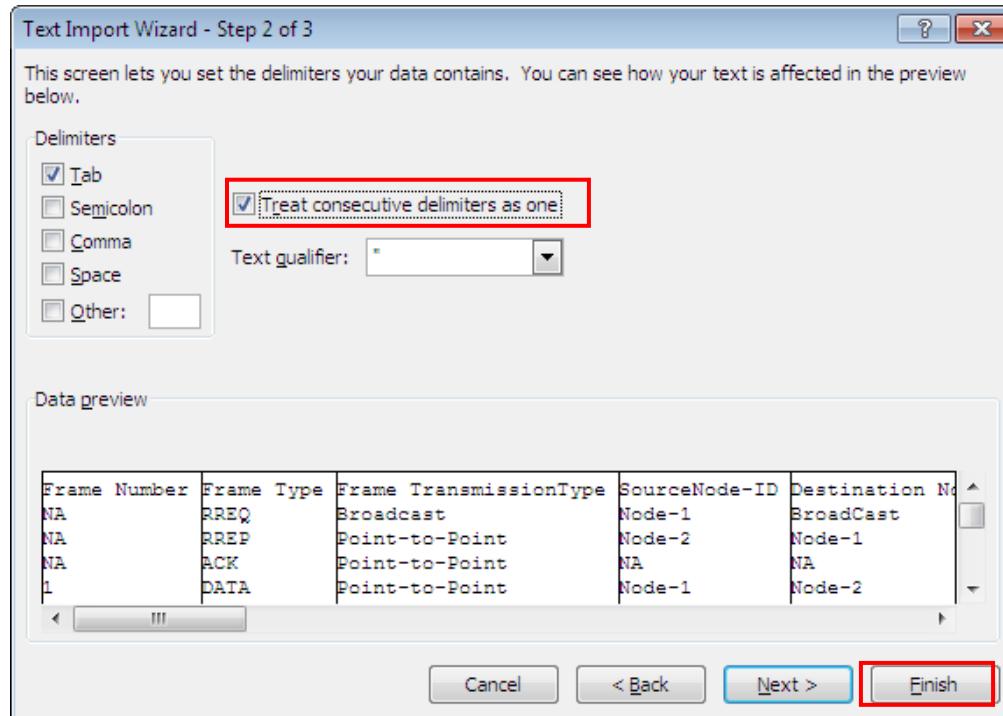
Step 1: Open excel sheet, select Data → From Text, it will ask for file, then select the trace file which you want to export to excel sheet.



Step 2: Select the file click import, the following window will appear,



Step 3: Click next, the following window will appear, in that select the check box as mentioned in the following window, and then click Finish button.



PacketTrace - Microsoft Excel

Home Insert Page Layout Formulas Data Review View

Clipboard Font Alignment Number Conditional Formatting Styles Cells Editing

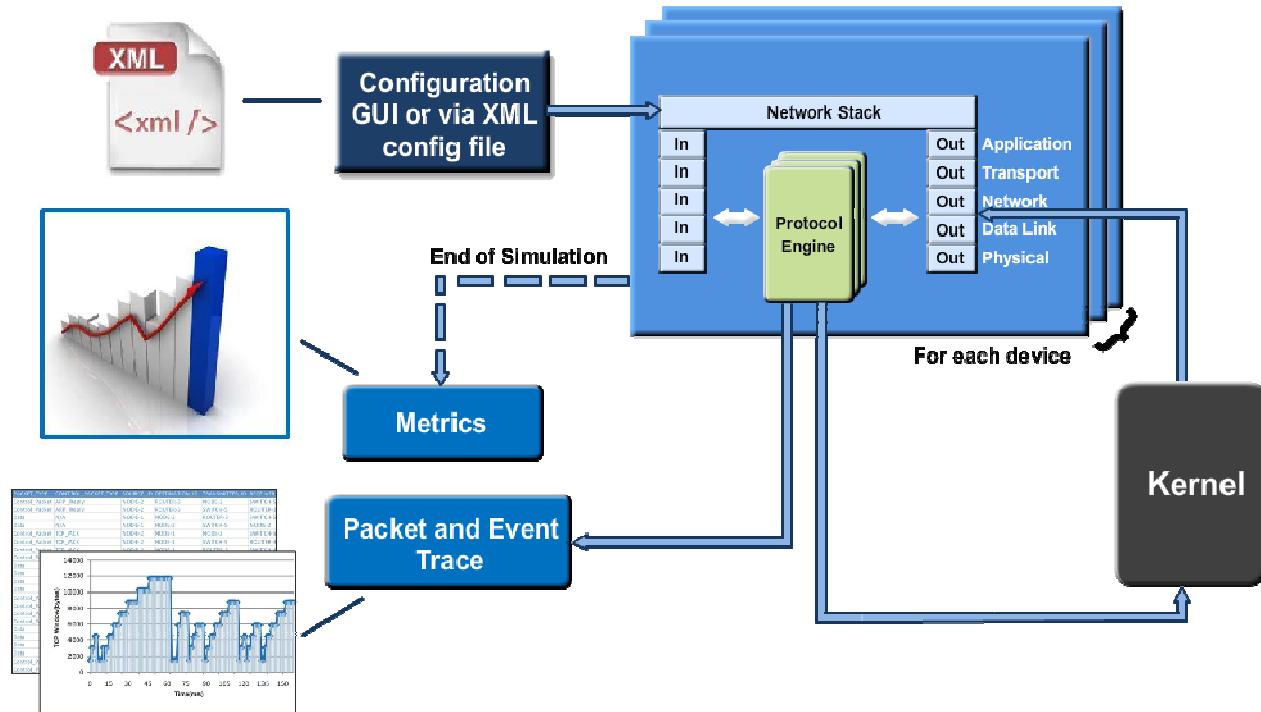
B11

	A	B	C	D	E	F	G	H	
1	PACKET_ID	PACKET_TYPE	CONTROL_PACKET_TYPE	SOURCE_ID	DESTINATION_ID	TRANSMITTER_ID	RECEIVER_ID	APP_LAYER_ARRIVAL_TIME	TRX_L
2	0	Video	N/A	NODE-8	NODE-2	NODE-8	SWITCH-3	33333.333	
3	0	Video	N/A	NODE-2	NODE-8	NODE-2	SWITCH-3	33333.333	
4	0	Video	N/A	NODE-2	NODE-8	SWITCH-3	NODE-8	33333.333	
5	0	Video	N/A	NODE-8	NODE-2	SWITCH-3	NODE-2	33333.333	
6	1	Voice	N/A	NODE-8	NODE-2	NODE-8	SWITCH-3	40000	
7	2	Voice	N/A	NODE-8	NODE-2	NODE-8	SWITCH-3	40000	
8	1	Voice	N/A	NODE-8	NODE-2	SWITCH-3	NODE-2	40000	
9	2	Voice	N/A	NODE-8	NODE-2	SWITCH-3	NODE-2	40000	
10	0	Video	N/A	NODE-8	NODE-2	NODE-8	SWITCH-3	66666.667	
11	0	Video	N/A	NODE-2	NODE-8	NODE-2	SWITCH-3	66666.667	

NetSim Network Stack:

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 802.11 B 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. A list of SUB EVENTS is given below.

PROTOCOL	SUBEVENT
ARP	
	READ_ARP_TABLE
	GENERATE_ARP_REQUEST
	GENERATE_ARP_REPLY
	UPDATE_ARP_TABLE_FWD_PKT
	ARP_REQUEST_TIMEOUT
FAST ETHERNET	
	CARRIER_SENSE
	WAIT_FOR_IFG
	SPANNING_TREE_PROTOCOL
	SWITCHING_TECHNIQUE
	SWITCH_TABLE_FORMATION
	SWITCH_FRAME_FORWARDING
	PHY_SENSE_LINK
RIP	
	SEND_RIP_UPDATE_PKT
	RIP_TIMEOUT
	RIP_GARBAGE_COLLECTION
	ROUTING_TABLE_UPDATION
TCP	
	TCP_ACTIVE_OPEN

	TCP_PASSIVE_OPEN
	TCP_SEND_SYN_ACK
	TCP_RECEIVE_SYN_ACK
	TCP_SEND_SEG
	TCP_RECEIVE_SEG
	TCP_SEND_ACK
	TCP_RECEIVE_ACK
	TCP_SEND_FIN_SEG
	TCP_RECEIVE_FIN_SEG
	TCP_SEND_FIN_ACK
	TCP_RECEIVE_FIN_ACK
	TCP_RETRANSMISSION_TIMEOUT
	TCP_TIME_WAIT_TIMEOUT
WLAN	
	CS
	CHECK_NAV
	NAV_END
	DIFS_END
	BACKOFF
	SEND_ACK
	SEND_RTS
	SEND_CTS
	SEND_MPDU
	CTS_TIMEOUT
	ACK_TIMEOUT
	RECEIVE_RTS
	RECEIVE_CTS
	RECEIVE_ACK
	RECEIVE_MPDU
	UPDATE_DEVICE_STATUS

IEEE 802.11b	
	WAIT_FOR_DIFS
	BACK_OFF
	ACK_TIMEOUT
	CTS_TIMEOUT
	SEND_RTS
	SEND_CTS
	SEND_ACK
	SEND_DATA
	ACK RECEIVED
	RTS RECEIVED
	CTS RECEIVED
	FRAME FORWARD
IEEE 802.11a	Same as IEEE 802.11b
IEEE 802.11g	Same as IEEE 802.11b
OSPF	
	INTERFACEUP
	WAITTIMER
	BACKUPSEEN
	NEIGHBORCHANGE
	LOOPIND
	UNLOOPIND
	INTERFACEDOWN
	HELLORECEIVED
	START_NBR
	TWOWAYRECEIVED
	NEGOTIATIONDONE
	EXCHANGEDONE
	BADLSREQ
	LOADINGDONE

	ADJOK
	SEQNUMBERMISMATCH
	ONEWAY
	KILLNBR
	INACTIVITYTIMER
	LLDOWN
	SEND_LSR
	SEND LSU
	SEND LSA
	RECEIVE_LSR
	RECEIVE LSU
	RECEIVE LSA
	MAXAGE

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 flat text file which can be stored at a user defined location.

Apart from a host of information, the event trace has two special information fields for diagnostics a) A log of the file name and line number from where the event was generated and b) Previous event which triggered the current event.

Note: Turning on Event Trace will slow down the simulation significantly

NetSim provides users with the option of turning on "Event Traces".

How to enable Event Trace via UI?

If NetSim runs via UI, event trace can be turned on by clicking the Event Trace icon in the tool bar in the Internetworks menu and selecting the required fields in the event trace.

How to enable Event Trace via CLI?

If NetSim runs via CLI, then the event trace can be turned on by enabling the event trace in the STATISTICS_COLLECTION tag of the configuration file.

How to import Event Trace to Excel?

Refer Help on “**Generating Packet Trace How to import Packet Trace to Excel?”**

Sample Experiment 1

Objective:

To understand IP forwarding within a LAN and across a router

Note: NetSim Standard Version is required to run this experiment

Introduction:

Nodes in network need MAC Addresses in addition to IP address for communicating with other nodes. In this experiment we will see how IP-forwarding is done when a node wants to send data within a subnet and also when the destination node is outside the subnet.

PART 1: - GUI MODE

Simulation:

How to Create Scenario & Simulate:

Drag and drop devices to create a network as shown in the screen shot below

- **Create Scenario:** “Help → NetSim Help F1→Running Simulation via GUI→ Internetworks → Create Scenario”.

After the network shown below is created, fill values in the properties of every node as shown in the table below.

NODE	TRANSMISSION TYPE	DESTINATION	TRAFFIC_TYPE
Wired Node 1	Point to Point	Wired Node 3	Custom
Wired Node 2	Point to Point	Wired Node 3	Custom
Wired Node 3	Point to Point	Wired Node 1	Custom
Wired Node 4	Point to Point	Wired Node 1	Custom

- In properties option of every node, click on **edit** in the **traffic type** box and fill the values of Packet Inter Arrival Time as given below:

NODE	Intra-LAN-IP-Forwarding	Across-Router -IP-Forwarding
Wired Node 1	1000	100000
Wired Node 2	100000	100000
Wired Node 3	100000	100000
Wired Node 4	100000	1000

Enable the packet trace by clicking on the **Packet Trace** button, give a file name and select file type as .csv. Then, Check ‘**all the attributes**’ button for COMMON ATTRIBTES, TCP and WLAN.

- Click on **simulate**. Give an experiment name and press OK.
- In resources menu on the left click on **Packet Trace Path** to view packet trace.

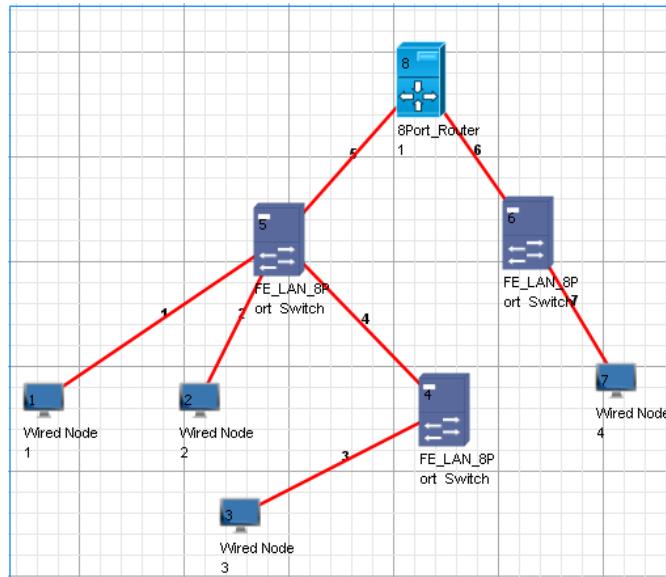
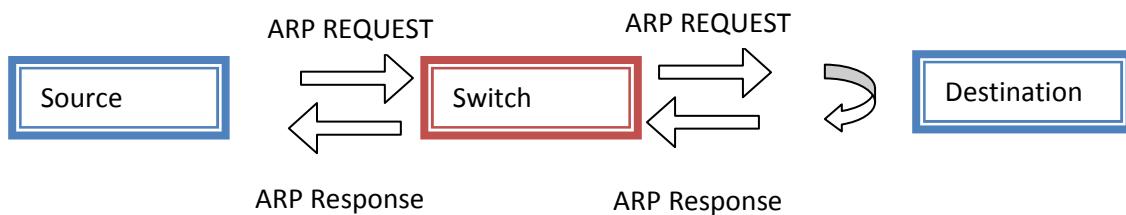


Fig: The Network scenario created in NetSim

Analysis:-

Intra-LAN-IP-forwarding:

- **ARP PROTOCOL- WORKING**

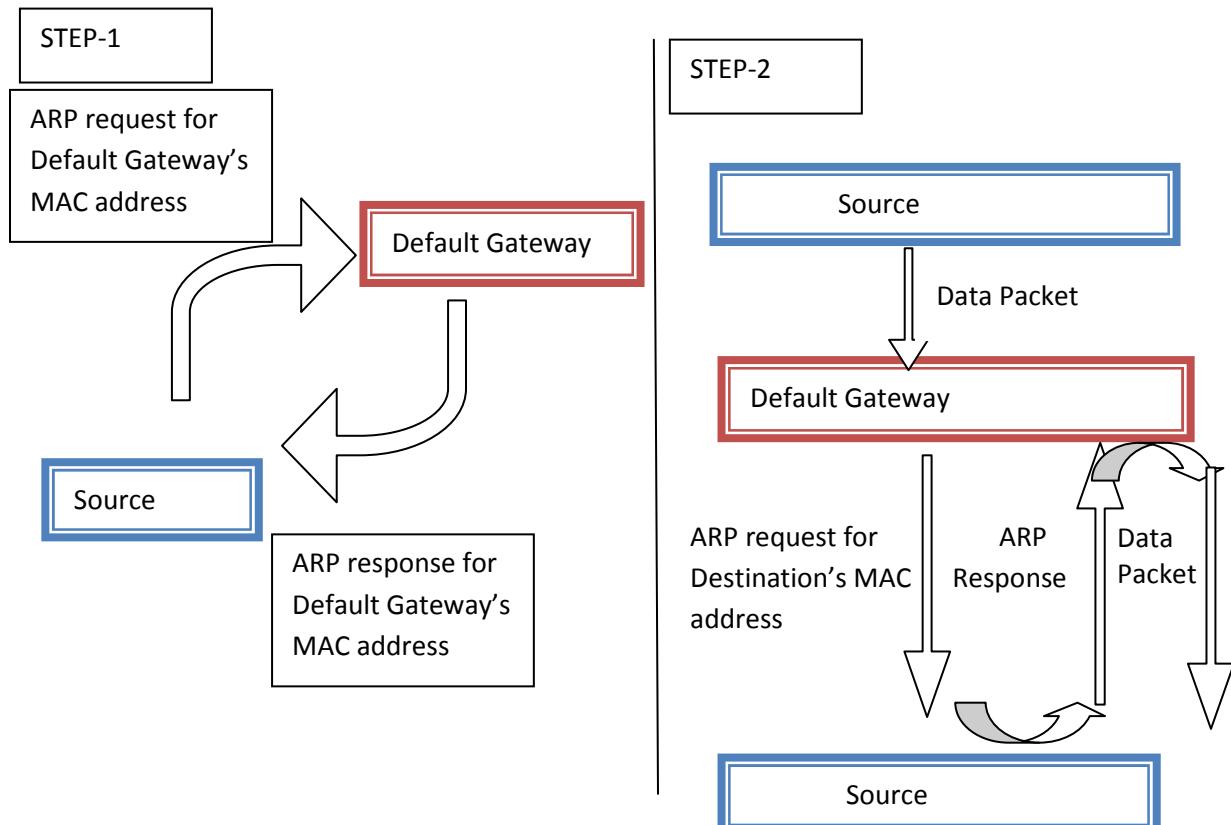


- PACKET TRACE Analysis

CONTROL_PACKET_TYPE	SOURCE_ID	DESTINATION_ID	TRANSMITTER_ID	RECEIVER_ID
ARP_Request	NODE-1	Broadcast-0	NODE-1	SWITCH-5
ARP_Request	NODE-1	Broadcast-0	SWITCH-5	NODE-2
ARP_Request	NODE-1	Broadcast-0	SWITCH-5	ROUTER-8
ARP_Reply	NODE-2	NODE-1	NODE-2	SWITCH-5
ARP_Reply	NODE-2	NODE-1	SWITCH-5	NODE-1
N/A	NODE-1	NODE-2	NODE-1	SWITCH-5
N/A	NODE-1	NODE-2	SWITCH-5	NODE-1

A Brief Explanation: NODE 1 broadcasts ARP request which is then broadcasted by SWITCH-5. NODE -2 sends the ARP reply to Node-1 via SWITCH-5. After this step, data is transmitted from NODE-1 to NODE-2. Notice the DESTINATION_ID column for ARP_Request type packets.

Across-Router-IP-forwarding:



- **PACKET TRACE Analysis**

PACKET_ID	CONTROL_PACKET_TYPE	SOURCE_ID	DESTINATIO_N_ID	TRANSMITTE_R_ID	RECEIVER_ID
0	CONFIG_BPDU	SWITCH-4	Broadcast-0	SWITCH-4	SWITCH-5
0	CONFIG_BPDU	SWITCH-5	Broadcast-0	SWITCH-5	SWITCH-4
0	ARP_Request	NODE-7	Broadcast-0	NODE-7	SWITCH-6
0	ARP_Request	NODE-7	Broadcast-0	SWITCH-6	ROUTER-8
0	ARP_Reply	ROUTER-8	NODE-7	ROUTER-8	SWITCH-6
0	ARP_Reply	ROUTER-8	NODE-7	SWITCH-6	NODE-7
1	N/A	NODE-7	NODE-1	NODE-7	SWITCH-6
1	N/A	NODE-7	NODE-1	SWITCH-6	ROUTER-8
0	ARP_Request	ROUTER-8	Broadcast-0	ROUTER-8	SWITCH-5
0	ARP_Request	ROUTER-8	Broadcast-0	SWITCH-5	NODE-1
0	ARP_Request	ROUTER-8	Broadcast-0	SWITCH-5	NODE-2
0	ARP_Reply	NODE-1	ROUTER-8	NODE-1	SWITCH-5
0	ARP_Reply	NODE-1	ROUTER-8	SWITCH-5	ROUTER-8
1	N/A	NODE-7	NODE-1	ROUTER-8	SWITCH-5
1	N/A	NODE-7	NODE-1	SWITCH-5	NODE-1

A Brief Explanation:

NODE-7 transmits a ARP_request which is further broadcasted by SWITCH-6. ROUTER-6 sends ARP-Reply to node 7 which goes through Switch-6. Then NODE-7 starts to send data to NODE-1.

If the router has the address of NODE-1 in its routing table, ARP protocol ends here and data transfer starts as we see PACKET_ID 1 being sent from NODE-7 to NODE-1.

In other case, which is our case Router sends ARP_request to appropriate subnet and after getting the MAC ADDRESS of the NODE-1, it forwards the packet which it had received from NODE-7.

PART 2: - Changing default Gateway

Do the same as done in part 1 with the difference that in the **properties** of NODE-4, change the **default gateway** to some other value, for ex. “192.168.2.76”. Follow ‘Across-Router-IP-forwarding’ section discussed above and click on simulate.

You will get error. It is so because NODE-4 will check the IP address of NODE 1 and then realize that it isn't in the same subnet. So it will forward it to default gateway. Since the default gateway's address doesn't exist in the network, error occurs.

Inference:

When a node has to send data to a node with known IP address but unknown Mac address, it sends an ARP request. If destination is in same subnet as the source (found through subnet mask) then it sends the ARP (broadcast ARP message) request. Otherwise it forwards it to default gateway. Former case happens in case of intra-LAN communication. The destination node sends an ARP response which is then forwarded by the switch to the initial node. Then data transmission starts.

In latter case, a totally different approach is followed. Source sends the ARP request to the default gateway and gets back the MAC address of default gateway. (If it knows which router to send then it sends ARP request to the corresponding router and not to Default gateway) When source sends data to default gateway (a router in this case), the router broadcasts ARP request for the destined IP address in the appropriate subnet. On getting the ARP response from destination, router then sends the data packet to destination node.

TCP

Sample Experiment 1

Objective:

To understand the working of “Connection Establishment” in TCP using NetSim.

Note: NetSim Standard Version is required to run this Experiment.

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.

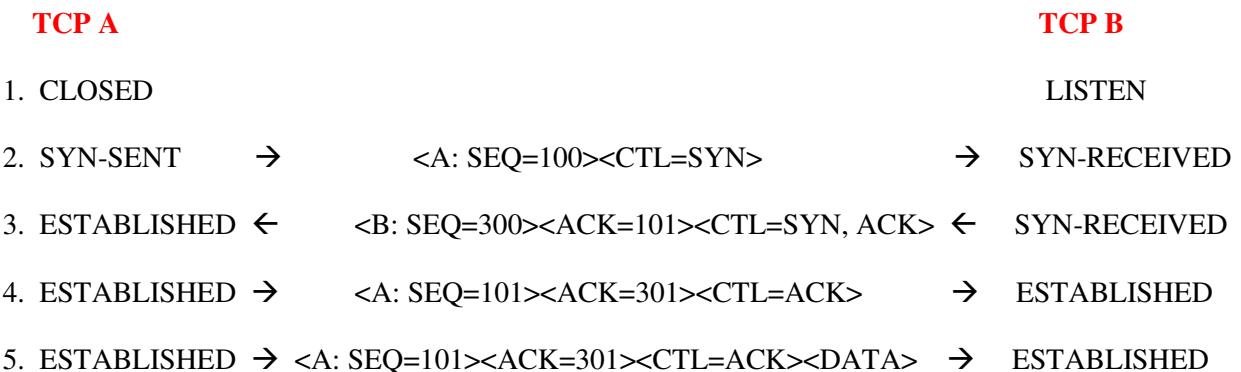


Fig: Basic 3-Way Handshake for Connection Synchronization

Explanation:

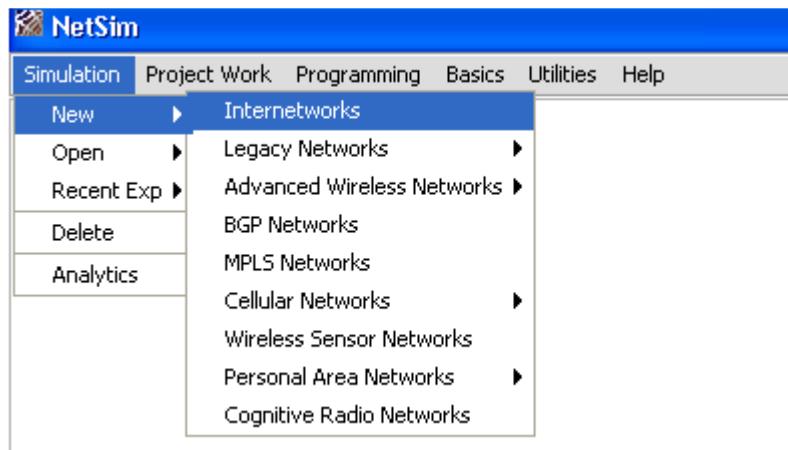
The above figure should be interpreted in the following way. Each line is numbered for reference purposes. Right arrows (\rightarrow) indicate departure of a TCP Segment from TCP A to TCP B, or arrival of a segment at B from A. Left arrows (\leftarrow), indicate the reverse. TCP states represent the 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.

Simulation:

How to Create Scenario & Generate Traffic:

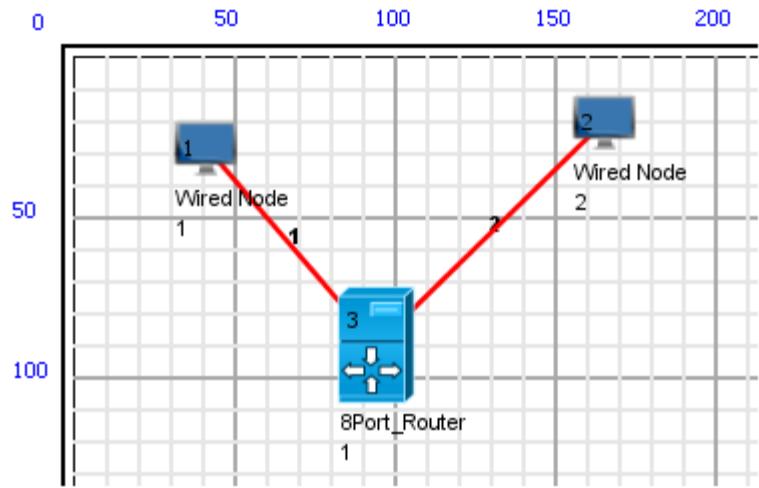
Step1:

Simulation \rightarrow New \rightarrow Internetworks



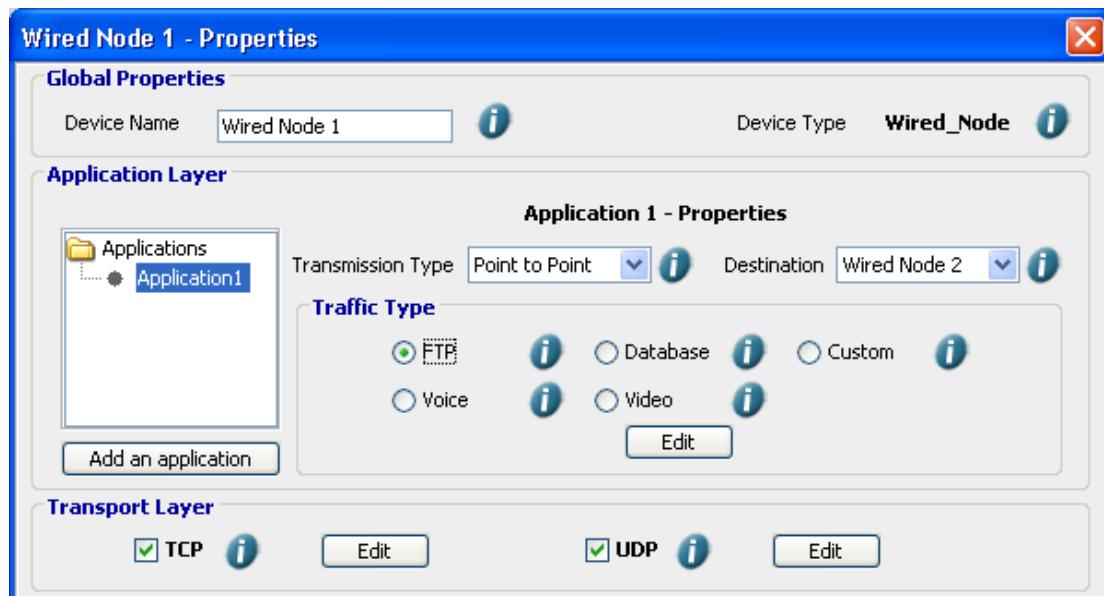
Step2:

Click & drop Wired Nodes and Router onto the Simulation Environment as shown below.



Step3:

Set the properties of the WiredNode1 as follows:



Generally Wired Node properties will be as follows:

Wired Node Properties	Values
Transmission Type	Point to point
Destination	Wired Node X (X will vary)
Traffic Type	FTP

Router Properties: Accept default properties for Router.

Enabling the packet trace:

- Click Packet Trace icon in the tool bar. This is used to log the packet details.
- Select the required attributes and click OK. Once the simulation is completed, the file gets stored in the specified location.

Note: make sure that after enabling the packet trace you select the **TCP option** in the **Internetworks** and then select the required attributes.

Simulation Time - 10 Sec

By clicking on “simulate” we will get the following results:

Metric	Value
PacketsTransmitted	2818
PacketsErrored	21
PacketsCollided	0
BytesTransmitted	2264916.00
PayloadTransmitted	2048180.00
OverheadTransmitted	216736.00

Fig: The Network Metrics.

Explanation of “Connection Establishment” using packet trace

- Go to the specified directory where the packet trace file has been saved.
- Open the file in excel.

Note: please refer

- Help → NetSim Help F1 → Generating Packet Trace → **How to import Packet Trace to Excel?**

PACKET_ID	SEGMENT_ID	PACKET_TYPE	CONTROL_PACKET_TYPE	SOURCE_ID	DESTINATION_ID	TRANSMITTER_ID	RECEIVER_ID	CONNECTION_STATE
0	N/A	Control_Packet	ARP_Request	NODE-1	Broadcast-0	NODE-1	ROUTER-3	N/A
0	N/A	Control_Packet	ARP_Reply	ROUTER-3	NODE-1	ROUTER-3	NODE-1	N/A
0	N/A	Control_Packet	TCP_SYN	NODE-1	NODE-2	NODE-1	ROUTER-3	TCP_SYN_SENT
0	N/A	Control_Packet	ARP_Request	ROUTER-3	Broadcast-0	ROUTER-3	NODE-2	N/A
0	N/A	Control_Packet	ARP_Reply	NODE-2	ROUTER-3	NODE-2	ROUTER-3	N/A
0	N/A	Control_Packet	TCP_SYN	NODE-1	NODE-2	ROUTER-3	NODE-2	TCP_SYN_SENT
0	N/A	Control_Packet	TCP_SYN_ACK	NODE-2	NODE-1	NODE-2	ROUTER-3	TCP_SYN_RECEIVED
0	N/A	Control_Packet	TCP_SYN_ACK	NODE-2	NODE-1	ROUTER-3	NODE-1	TCP_SYN_RECEIVED
0	N/A	Control_Packet	TCP_ACK	NODE-1	NODE-2	NODE-1	ROUTER-3	TCP_ESTABLISHED
1	1 Data	N/A		NODE-1	NODE-2	NODE-1	ROUTER-3	TCP_ESTABLISHED

Fig: 3-way Handshake using packet trace.

In EXCEL go to DATA and select FILTER option to view only the desired rows and columns as shown in the figure

Line 1, 2, 4 and 5 of the above table are ARP related packets and not of interest to us in this experiment.

In line 3 of the above figure we can see that NODE-1 is sending a control packet of type TCP_SYN requesting the connection with the NODE-2, and this control packet is first sent to the ROUTER-3 (receiver ID). In line 6, the ROUTER-3 is sending the TCP_SYN packet that has been received from NODE-1 to the NODE-2. In line 7, NODE-2 is sending the control packet of type TCP_SYN_ACK to NODE-1, and this control packet is first sent to the ROUTER-3. This TCP_SYN_ACK is the ACK packet for the TCP_SYN packet. In line 8, ROUTER-3 is sending the TCP_SYN_ACK, (received from NODE-2) to the NODE-1. In line 9, NODE-1 is sending the TCP_ACK to NODE-2 via ROUTER-3 making the CONNECTION_STATE as TCP_ESTABLISHED.

Once the connection is established, we see that a packet type of type “DATA” is sent from the NODE-1 to the NODE-2 in line 10.

Example Assessment Question:

- Explain the working of “Closing a connection” in TCP using the NetSim’s packet trace.

Sample Experiment 2

Objective: During client-server TCP downloads study the throughputs of Slow start + Congestion avoidance (also known as Old Tahoe) and Fast Retransmit(also known as Tahoe), Congestion Control Algorithms.

Theory:

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

Old Tahoe:

Congestion can occur when data arrives on a big pipe (i.e. a fast LAN) and gets sent out through a smaller pipe (i.e. a slower WAN). Congestion can also occur when multiple input streams arrive at a router whose output capacity is less than the sum of the inputs. Congestion avoidance is a way to deal with lost packets.

The assumption of the algorithm is that the packet loss caused by damaged is very small (much less than 1%), therefore the loss of a packet signals congestion somewhere in the network between the source and destination. There are two indications of packets loss: a timeout occurring and the receipt of duplicate ACKs

Congestion avoidance and slow start are independent algorithms with different objectives. But when congestion occurs TCP must slow down its transmission rate and then invoke slow start to get things going again. In practice they are implemented together.

Congestion avoidance and slow start requires two variables to be maintained for each connection: a Congestion Window (i.e. cwnd) and a Slow Start Threshold Size (i.e. ssthresh). Old Tahoe algorithm is the combination of slow start and congestion avoidance. The combined algorithm operates as follows,

1. Initialization for a given connection sets cwnd to one segment and ssthresh to 65535 bytes.
2. When congestion occurs (indicated by a timeout or the reception of duplicate ACKs), one-half of the current window size (the minimum of cwnd and the receiver's advertised window,

but at least two segments) is saved in ssthresh. Additionally, if the congestion is indicated by a timeout, cwnd is set to one segment (i.e. slow start).

3. When new data is acknowledged by the other end, increase cwnd, but the way it increases depends on whether TCP is performing slow start or congestion avoidance.

If cwnd is less than or equal to ssthresh, TCP is in slow start. Else TCP is performing congestion avoidance. Slow start continues until TCP is halfway to where it was when congestion occurred (since it recorded half of the window size that caused the problem in step 2). Then congestion avoidance takes over.

Slow start has cwnd begins at one segment and be incremented by one segment every time an ACK is received. As mentioned earlier, this opens the window exponentially: send one segment, then two, then four, and so on. Congestion avoidance dictates that cwnd be incremented by $1/cwnd$, compared to slow start's exponential growth. The increase in cwnd should be at most one segment in each round trip time (regardless of how many ACKs are received in that RTT), whereas slow start increments cwnd by the number of ACKs received in a round-trip time.

Tahoe (Fast Retransmit):

The Fast retransmit algorithms operating with Old Tahoe is known as the Tahoe variant.

TCP may generate an immediate acknowledgement (a duplicate ACK) when an out-of-order segment is received out-of-order, and to tell it what sequence number is expected.

Since TCP does not know whether a duplicate ACK is caused by a lost segment or just a re-ordering of segments, it waits for a small number of duplicate ACKs to be received. It is assumed that if there is just a reordering of the segments, there will be only one or two duplicate ACKs before the re-ordered segment is processed, which will then generate a new ACK. If three or more duplicate ACKs are received in a row, it is a strong indication that a segment has been lost. TCP then performs a retransmission of what appears to be the missing segment, without waiting for a re-transmission timer to expire.

Procedure:

How to Create Scenario

- **Create Scenario:** “Help → NetSim Help F1 → Running Simulation via GUI → Internetwork → Create Scenario”.

Sample Inputs:

Follow the steps given in the different samples to arrive at the objective.

Sample 1.a: Old Tahoe (1 client and 1 server)

In this Sample,

- Total no of Node used: 2
- Total no of Routers used: 2

The devices are inter connected as given below,

- Node 1 is connected with Router 1 by Link 1.
- Router 1 and Router 2 are connected by Link 2.
- Node 2 is connected with Router 2 by Link 3.

Set the properties for each device by following the tables,

Node Properties	Wired Node 2
Destination	Node 1
Traffic Type	Custom
Packet Size	
Distribution	Constant
Packet Size (Bytes)	1460
Packet Inter Arrival Time	
Distribution	Constant
Packet Inter Arrival Time (ms)	1300
TCP Properties	

TCP MSS	1460
Congestion Control	Old Tahoe
Window size (Bytes)	8

Router Properties: Accept default properties for Router.

Link Properties	Link 1	Link 2	Link 3
Uplink Error Rate (BER)	10^{-6}	10^{-6}	10^{-6}
Downlink Error Rate (BER)	10^{-6}	10^{-6}	10^{-6}
Medium Type	E2	CAT5	E2
Uplink and Downlink speed (Mbps)	8.448	10	8.448

Simulation Time - 10 Sec

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

- Set the properties of Node , Router, & Link
- Then click on the Simulate button).

Sample 1.b: Tahoe (1 client and 1 server)

Do the experiment as sample 1.a, and change the TCP congestion control algorithm to Tahoe.

Sample 2.a: Old Tahoe (2 clients and 2 servers)

In this Sample,

- Total no of Wired Node's used: 4
- Total no of Routers used: 2

The devices are inter connected as given below,

- Wired Node 1 and Wired Node 2 are connected with Router 1 by Link 1 and Link 2.
- Router 1 and Router 2 are connected by Link 3.
- Wired Node 3 and Wired Node 4 are connected with Router 2 by Link 4 and Link 5.
- Wired Node 1 and Wired Node 2 are not transmitting data in this sample.

Set the properties for each device by following the tables,

Node Properties	Wired Node 3	Wired Node 4
Destination	Node 1	Node 2
Traffic Type	Custom	Custom
Packet Size		
Distribution	Constant	Constant
Packet Size (Bytes)	1460	1460
Packet Inter Arrival Time		
Distribution	Constant	Constant
Packet Inter Arrival Time (ms)	1300	1300
TCP Properties		
TCP MSS	1460	1460
Congestion Control	Old Tahoe	Old Tahoe
Window size (Bytes)	8	8

Router Properties: Accept default properties for Router.

Link Properties	Link 1	Link 2	Link 3	Link 4	Link 5
Uplink Error Rate (BER)	10^{-6}	10^{-6}	10^{-6}	10^{-6}	10^{-6}
Downlink Error Rate (BER)	10^{-6}	10^{-6}	10^{-6}	10^{-6}	10^{-6}

Physical Medium	E2	E2	CAT5	E2	E2
Uplink and Downlink speed (Mbps)	8.448	8.448	10	8.448	8.448

Simulation Time - 10 Sec

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

- Set the properties of Node , Router, & Link
- Then click on the Simulate button).

Sample 2.b: Tahoe (2 clients and 2 servers)

Do the experiment as sample 2.a, and change the congestion control algorithm to Tahoe.

Sample 3.a: Old Tahoe (3 clients and 3 servers)

In this Sample,

- Total no of Node's used: 6
- Total no of Routers used: 2

The devices are inter connected as given below,

- Node 1, Node 2 and Node 3 are connected with Router 1 by Link 1, Link 2 and Link 3.
- Router 1 and Router 2 are connected by Link 4.
- Node 4, Node 5 and Node 6 are connected with Router 2 by Link 5, Link 6 and Link 7.
- Node 1, Node 2 and Node 3 are not transmitting data in this sample.

Set the properties for each device by following the tables,

Node Properties	Wired Node 4	Wired Node 5	Wired Node 6
Destination	Node 1	Node 2	Node 3
Traffic Type	Custom	Custom	Custom
Packet Size			
Distribution	Constant	Constant	Constant

Packet Size (Bytes)	1460	1460	1460
Packet Inter Arrival Time			
Distribution	Constant	Constant	Constant
Packet Inter Arrival Time (ms)	1300	1300	1300
TCP Properties			
TCP MSS	1460	1460	1460
Congestion Control	Old Tahoe	Old Tahoe	Old Tahoe
Window size (Bytes)	8	8	8

Router Properties: Accept default properties for Router.

Link Properties	Link 1	Link 2	Link 3	Link 4	Link 5	Link 6	Link 7
Uplink Error Rate (BER)	10^{-6}	10^{-6}	10^{-6}	10^{-6}	10^{-6}	10^{-6}	10^{-6}
Downlink Error Rate (BER)	10^{-6}	10^{-6}	10^{-6}	10^{-6}	10^{-6}	10^{-6}	10^{-6}
Physical Medium	E2	E2	E2	CAT5	E2	E2	E2
Uplink and Downlink speed (Mbps)	8.448	8.448	8.448	10	8.448	8.448	8.448

Simulation Time - 10 Sec

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

- Set the properties of Node , Router, & Link
- Then click on the Simulate button).

Sample 3.b: Tahoe (3 clients and 3 servers)

Do the experiment as sample 3.a, and change the TCP congestion control algorithm to Tahoe.

Comparison Table:

TCP Downloads	Metrics	Slow start + Congestion avoidance	Fast Retransmit
1 client and 1 server	User lever throughput	5.89	7.04
	Segments retransmitted + Fast retransmitted	124	229
2 clients and 2 servers	User lever throughput	8.19	9.43
	Segments retransmitted + Fast retransmitted	224	253
3 clients and 3 servers	User lever throughput	8.91	9.41
	Segments retransmitted + Fast retransmitted	256	315

Note: To calculate the “User level throughput” for more than one client, add the individual client throughput which is available in Application Metrics (Metrics.txt). In the same way calculate the metrics for “Segments retransmitted + Fast retransmitted”.

Inference:

User lever throughput: User lever throughput of Fast Retransmit is higher when compared then the Old Tahoe (SS + CA). This is because, if a segment is lost due to error, Old Tahoe waits until the RTO Timer expires to retransmit the lost segment, whereas Tahoe (FR) retransmits the lost segment immediately after getting three continuous duplicate ACK’s. This results in the increased segment transmissions, and therefore throughput is higher in the case of Tahoe.

Wireless LAN

Sample Experiment 1

Objective:

To study how the Bit Error Rate (loss) and data of a Wireless LAN (IEEE 802.11b) network varies as the distance between the Access Point and the wireless nodes is varied.

Theory:

Please navigate through the below given path to,

Basics → Internetworks → Wireless LAN → IEEE 802.11 PHY.

In most of the WLAN products on the market based on the IEEE 802.11b technology the transmitter is designed as a Direct Sequence Spread Spectrum Phase Shift Keying (DSSS PSK) modulator, which is capable of handling data rates of up to 11 Mbps. The system implements various modulation modes for every transmission rate, which are Different Binary Phase Shift Keying (DPSK) for 1 Mbps, Different Quaternary Phase Shift Keying (DQPSK) for 2 Mbps and Complementary Code Keying (CCK) for 5.5 Mbps and 11 Mbps.

Large Scale Fading represents Receiver Signal Strength or path loss over a large area as a function of distance. The statistics of large scale fading provides a way of computing estimated signal power or path loss as a function of distance and modulation modes vary depends on the Receiver Signal Strength.

Procedure:

How to Create Scenario & Generate Traffic:

Please navigate through the below given path to,

- **Create Scenario:** “Help → NetSim Help F1 → Running Simulation via GUI → Internetworks → Create Scenario”.

Sample Inputs:

In this Sample experiment 2 Nodes and 2 Switches need to be dragged & dropped onto the Simulation environment. After which the below mentioned experiments are done.

Upon completion of the experiment, “Save” it and plot the graph using Excel.

Inputs for the Sample experiments are given below,

In this Sample experiment, 1 Wireless AP and two Wireless Nodes (WirelessNode1 and WirelessNode2) required needs to be dragged & dropped onto the Simulation environment. Connect the AP and the Nodes. Set the below properties for AP, Wireless Nodes and Wireless Link. Run the simulation. Upon completion of the experiment “Save” them for comparisons that can be carried out in the “Analytics” section by using export to Excel

The follow these steps, Wireless Node 1 transmits data to Wireless Node 2

- Experiment 1: Distance between Node1 and Node2 to Access Point is 5m.
- Experiment 2: Distance between Node1 and Node2 to Access Point is 10m.
- Experiment 3: Distance between Node1 and Node2 to Access Point is 15m.
- And so on till all 80 meter distance.

Inputs for the Sample experiment are given below:

Access Point Properties	AP 1
Protocol	IEEE802.11b
Buffer Size(MB)	5
RTS Threshold(Bytes)	2347
Retry Limit	7
Transmission Type	DSSS
Standard Channel	1_2412
Transmitter power (Milli Watts)	100
SIFS	10
Slot Time	20
CW Minimum	31

Wireless Node Properties	Node 1	
Transmission	Point to Point	
Destination	Node2	
Traffic Type	Custom	
Application Data size	Distribution	Constant
	Data Size (Bytes)	1375
Inter Arrival Time	Distribution	Constant
	Packet Inter Arrival Time (Micro sec)	1000
ARP Retry Interval	10	

ARP Retry Limit	3
Protocol	IEEE802.11b
RTS Threshold(Bytes)	2347
Retry Limit	7
Transmission	DSSS
Standard Channel	1_2412
Transmitter power (Milli Watts)	100
SIFS	10
Slot Time	20
CW Minimum	31

Link Properties	Link 1
Medium Type	WIRELESS
Uplink Speed (Mbps)	11
Downlink Speed (Mbps)	11
Frequency	2412
Path Loss Exponent	3.5
Fading Figure	1.0
Standard Deviation Delay (Micro sec)	0

Simulation Time - 10 Sec

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

- Set the properties for the Wireless Nodes, AP and Wireless Link.
- Click on the Validate & Simulate button).

Output:

To view the output by using NetSim the Sample experiments need to be added onto the Analytics interface. Given below is the navigation for analytics - “Simulation → Analytics”.

Select the experiments by selecting

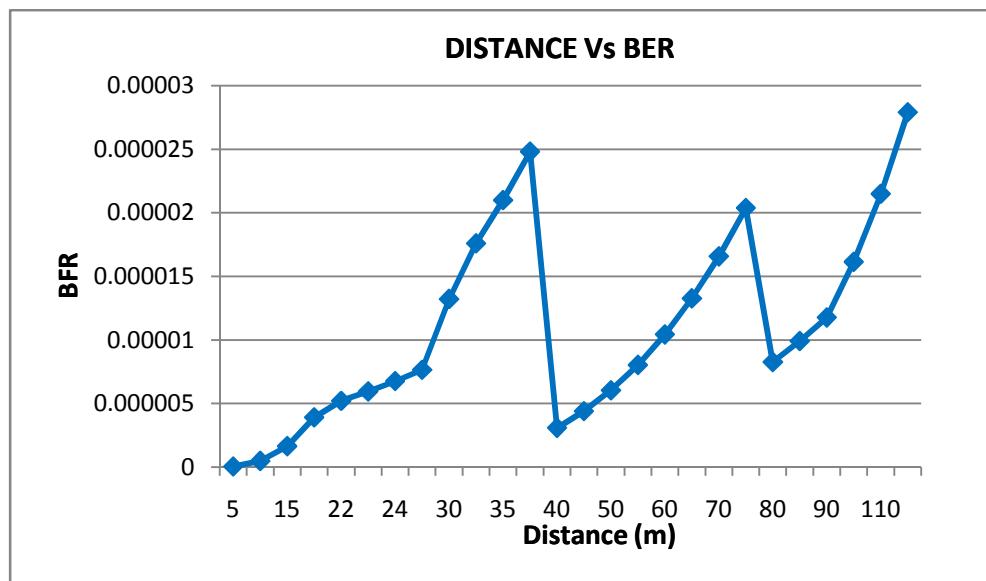
- Wireless LAN Protocols
- Select the Experiments (Note: Click on one experiment after the other to add multiple experiments need to be added onto the Analytics interface).

- Click on “Export to .csv”

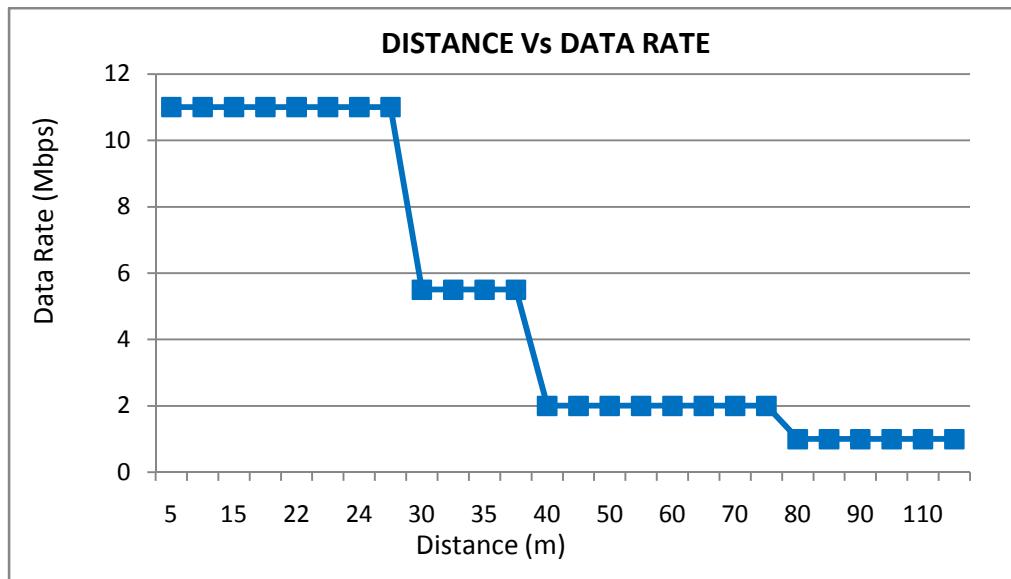
Comparison Chart:

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

Graph I



Graph II



Graph I

Sl. No	Approx Range (m)	Data Rate (Mbps)	Modulation	Loss (%)
1.	1 – 1`7*	11	CCK11	Exponential Increases.
2.	17 – 24*	5.5	CCK5.5	The BER first falls as the data rate drops. Then it starts to exponential increase.
3.	24 - 40 *	2	DQPSK	The BER falls as the data rate drops. Then it starts to exponential increase.
4.	40 – 70 *	1	DBPSK	The BER first falls as the data rate drops. Then it starts to exponential increase.

* denotes the modulation ranges affected by fading effect.

*** All the above plots highly depend upon the placement of Node 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.

Graph II

The data rate between the Access Point and the wireless nodes decreases. This is because data rate depends on the received power at the receiver. Received power is directly proportional to $(1 / \text{distance})$.

Sample Experiment 2

Objective:

Analysis for capacity evaluation of an infrastructure IEEE 802.11 based network carrying a full-duplex packet telephone calls. (Ref IEEE paper “Analytical Models for Capacity Estimation of IEEE 802.11 WLANs using DCF for Internet Applications” by George Kuriakose, Sri Harsha, Anurag Kumar and Vinod Sharma)

Theory:

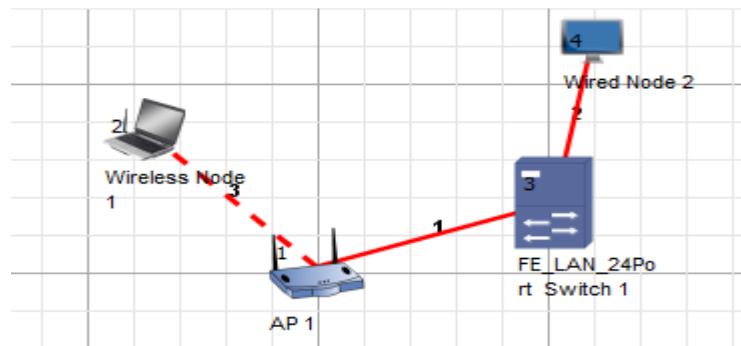
There are N STAs, all associated with a single AP. Each STA has a single full duplex VoIP call to a wired client on the wired LAN via the AP. The calls are not synchronized with each other. Each call results in two UDP streams, one from a remote client to a wireless STA, and another in the reverse direction. We begin by considering the case where each call uses the ITU G711 codec. Packets are generated every 20 ms. Including the IP, UDP and RTP headers, the size of the packet emitted in each call in each direction is 200 bytes every 20 ms.

Sample Inputs:

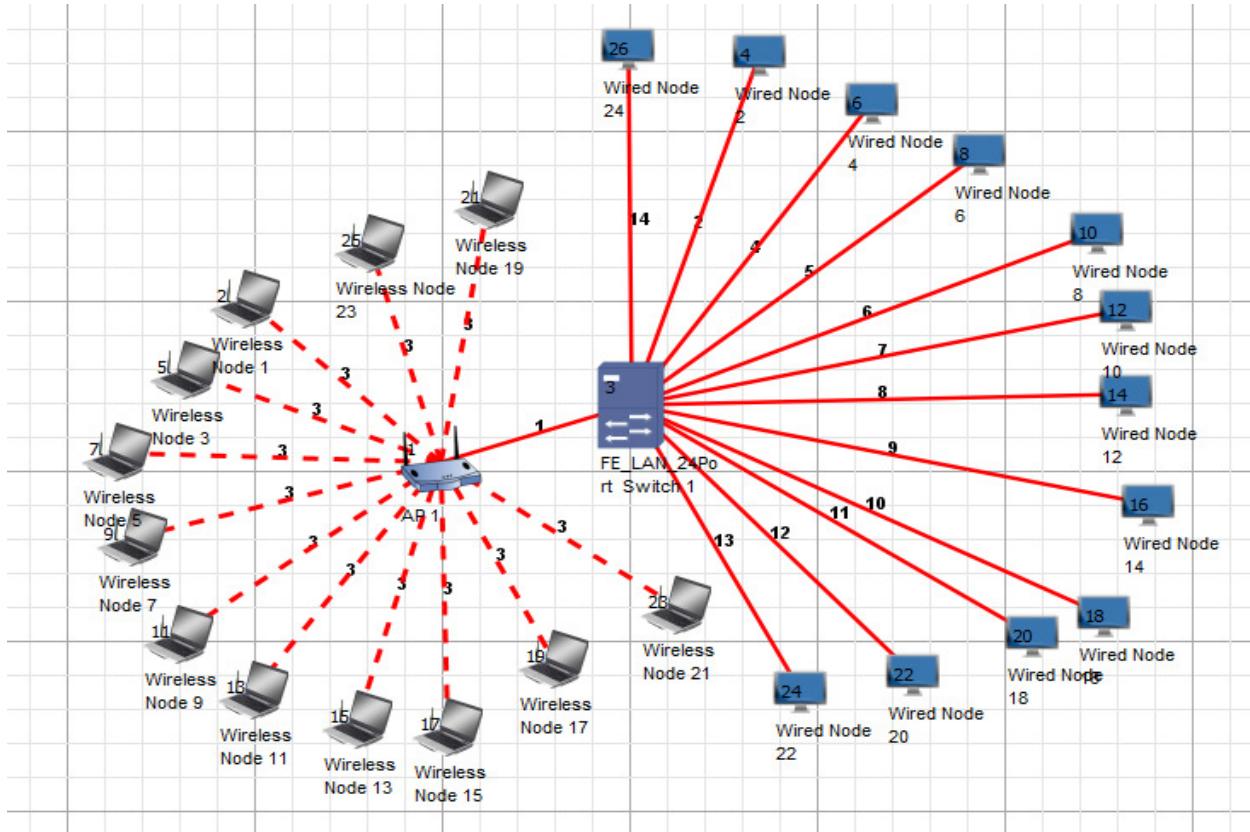
Here we are considering AP, SWITCH, Wired Nodes, and Wireless nodes.

Experiment for 11Mbps:

Experiment1_11Mbps: The Wired Node1 sends Voice traffic (G711 codec) to Destination Wireless Node1 through SWITCH and AP. Similarly the Wireless Node1 sends Voice traffic to Wired Node1 thereby setting up full duplex call. Set the properties as per given below. Run the simulation and save the result.



Experiment2_11Mbps: Add one more Wireless Node 2 to AP as well as Wired Node 2 to SWITCH to setup 2nd call. Then set properties for a call from Wired Node2 to Wireless Node1 and vice versa. Repeat the experiment for 20 wireless node and 20 wired nodes by increasing one wireless and one wired node at a time.



Properties Settings:

Access Point Properties	AP 1
Protocol	IEEE802.11b
Buffer Size(MB)	5
RTS Threshold(Bytes)	1000
Retry Limit	7
Transmission Type	DSSS
Standard Channel	1_2412
Transmitter power	100

SIFS	10
Slot Time	20
CW Minimum	31

Link Properties	All Wireless Links	Link Properties	All Wired Links
Medium Type	WIRELESS	Medium Type	CAT -5
Uplink Speed	11	Uplink Speed	100
Downlink Speed	11	Downlink Speed	100
Frequency	2412	Uplink BER	No Error
Path Loss Exponent	0 or 2	Down Link BER	No Error
Fading Figure	0	Propagation Delay	Default
Standard Deviation	0	Standard Deviation	Default

Note: Path Loss Exponent if set to 2, to achieve 2 mbps transmission rate.

Place all the nodes within 40 meter to get 11 Mbps, and between 70 to 120m to achieve 2 Mbps.

Switch Properties:

Select **FE_LAN_24PortSwitch**. Keep all default properties.

Distribution	Constant	Constant	Constant	Constant	Constant	Constant
Packet Size	160	160	160	160	160	160
Packet Inter Arrival Time						
Distribution	Constant	Constant	Constant	Constant	Constant	Constant
Inter Arrival Time	20000	20000	20000	20000	20000	20000
Service Type	CBR	CBR	CBR	CBR	CBR	CBR

Simulation Time- 10 Sec

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

- Set the properties for the Wireless Nodes, AP, switch and Links.
- Click on the Validate & Simulate button).

Run the simulation save the results by Export to csv.

Experiment for 2Mbps:

Change the Path Loss Exponent of Wireless link to 2. Place Wireless Nodes between 80 to 120 meters. Repeat the same 10 experiment for data rate 2Mbps.

Run the simulation save the results by Export to csv.

Voice Call Capacity ($\phi_{AP-V oIP}$): The Voice Call capacity of the AP is the number of successes of the AP in successive channel slot it utilized. This can be calculated by considering the number of successes of the AP in a time t.

The rate at which a single call sends data to the AP is λ . Since the AP serves N such calls the total input rate to the AP is $N\lambda$. Obviously, this rate should be less than $\phi_{AP-V oIP}(N)$.

Thus, we define

$$N_{max} = \max(\phi_{AP-V oIP}(N) > N\lambda).$$

The arrival Rate λ = Number of bits transmitted per second.

The G.711 the packet size is 160 bytes, UDP header is 8 bytes, IP header is 20 bytes and MAC header is 40 bytes.

The packets inter arrival time for G.711 is 20 ms.

$$\text{Arrival rate} = \text{Packet size} / \text{Time} = ((160+8+20+40)*8) / 20 * 10^{-3} = 0.0912 \text{Mbps.}$$

To Calculation of $\phi_{AP-VoIP}(N)$ from the WLAN Metrics do the following.

1. Calculate sum of the ‘MediumAccessTime(ms)’ and ‘TransmissionTime(ms)’.
2. Divide the result by ‘TransmittedFrameCountUnicast’ of AP. This gives the time taken to transmit the single packet. Time taken per packet.
3. One over Time taken per packet gives the $\phi_{AP-VoIP}(N)$.

$$\phi_{AP-VoIP} = \frac{1}{\frac{\text{MediumAccessTime} + \text{TransmissionTime}}{\text{TransmittedFramecountUnicast}}} \text{Mbps}$$

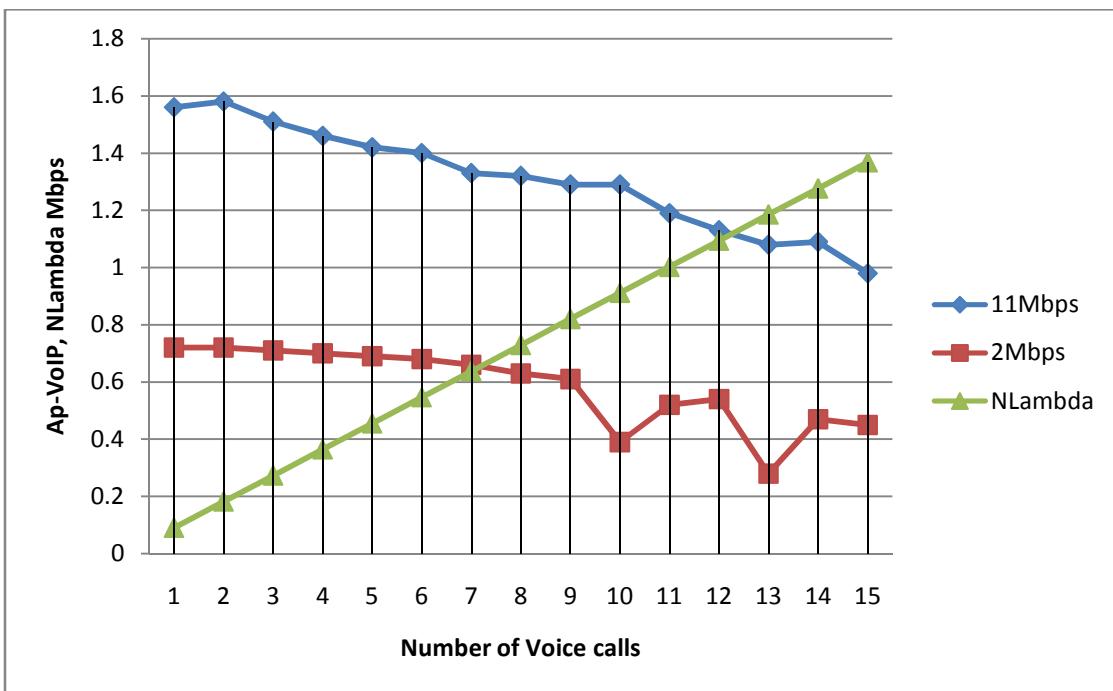
Attempt Rate $\Phi(N)$: If N nodes have non-empty queues at any slot, then calculate t attempt probability of a node. This is the average rate at which the nodes attempt or contend for the channel. The attempt rate is the cumulative number of attempts until time t. The attempt rate is easily obtained from the simulation Metrics file since we just have to count the total number of attempts (TransmittedFrameCountUnicast) in the network and average out on the total simulation time.

$$\Phi(N) = \frac{\text{Total Transmitted Frame CountUnicast}}{\text{SimulationTime}}$$

Total Transmitted Frame Count Unicast = Sum of all ‘TransmittedFrameCountUnicast’ from all nodes.

Plot1:

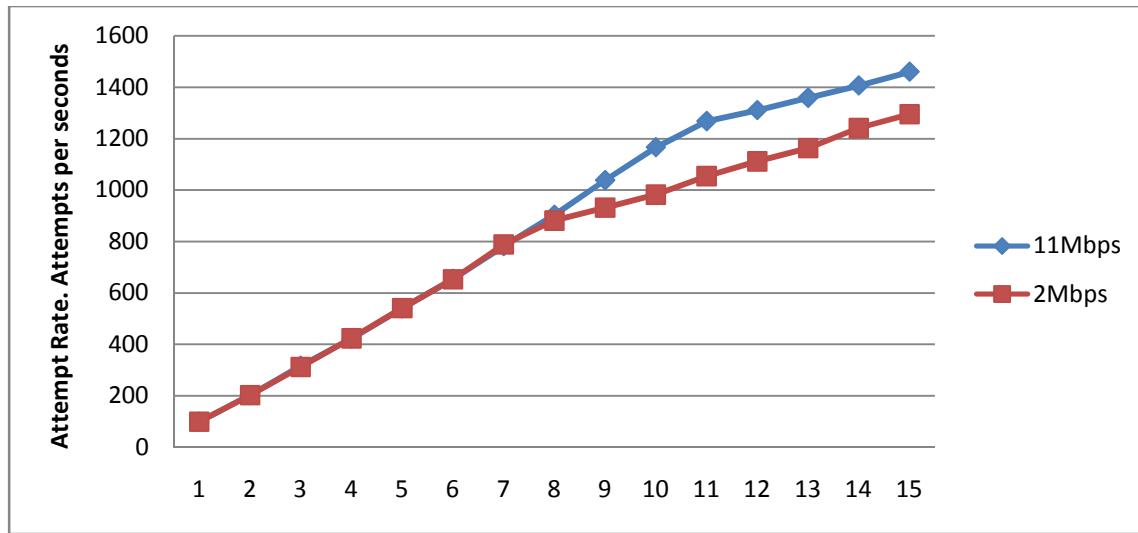
	Packet size = 160 (G.711 packet size) + 8 (UDP header)+ 20(IP header) + 40(Mac haeder) = 228 bytes							
	Lambda L = Packet size / Time = $((160+8+20+40)*8) / 20 * 10^{-3} = 0.0912 \text{ Mbps}$							
		11Mbps			2Mbps			
Experiments	Number of Voice Calls N	Transmitte d FrameUnic ast From AP	Medium Access Time + Transmission Time	Voice Capacity	Transmitted FrameUnicast From AP	Medium Access Time+ Transmission Time	Voice Capacity	Nλ
Experiment1	1	500	320.13	1.56	501	692.71	0.72	0.0912
Experiment2	2	1002	632.8	1.58	1003	1380.79	0.72	0.1824
Experiment3	3	1515	1003.28	1.51	1512	2133.72	0.71	0.2736
Experiment4	4	2027	1385.75	1.46	2022	2882.88	0.70	0.3648
Experiment5	5	2544	1785.44	1.42	2534	3645.16	0.69	0.456
Experiment6	6	3051	2172.08	1.40	3051	4466.32	0.68	0.5472
Experiment7	7	3588	2694.37	1.33	3599	5446.43	0.66	0.6384
Experiment8	8	4017	3046.54	1.32	3777	5985.73	0.63	0.7296
Experiment9	9	4620	3577.5	1.29	3554	5775.76	0.61	0.8208
Experiment10	10	5133	3979.25	1.29	3330	8581.61	0.39	0.912
Experiment11	11	5231	4363.09	1.19	3067	5790.58	0.52	1.0032
Experiment12	12	4817	4230.31	1.13	2813	5196.97	0.54	1.0944
Experiment13	13	4363	4010.46	1.08	2603	9241.48	0.28	1.1856
Experiment14	14	3970	3612.77	1.09	2316	4828.13	0.47	1.2768
Experiment15	15	3524	3587.87	0.98	2107	4643.19	0.45	1.368



Plot2:

Simulation Time = 10 secs					
Attempt Rate = Total Transmitted FrameUnicast / 10					
Experiment	Number of Voice calls	11Mbps		2Mbps	
		Total Transmitted FrameUnicast	Attempt Rate	Total Transmitted FrameUnicast	Attempt Rate
Experiment1	1	1001	100.1	1001	100.1
Experiment2	2	2032	203.2	2030	203
Experiment3	3	3170	317	3122	312.2
Experiment4	4	4229	422.9	4240	424
Experiment5	5	5402	540.2	5415	541.5
Experiment6	6	6549	654.9	6530	653
Experiment7	7	7822	782.2	7888	788.8
Experiment8	8	9035	903.5	8815	881.5
Experiment9	9	10385	1038.5	9317	931.7
Experiment10	10	11662	1166.2	9823	982.3
Experiment11	11	12678	1267.8	10542	1054.2

Experiment12	12	13105	1310.5	11115	1111.5
Experiment13	13	13585	1358.5	11629	1162.9
Experiment14	14	14065	1406.5	12409	1240.9
Experiment15	15	14595	1459.5	12948	1294.8



Inference:

Plot1:

Shows the graph of AP service rate, $\phi_{AP-VoIP}$, versus the number of calls, N, for two PHY rates 11Mbps and 2Mbps, for G.711. Also shown is the line $N\lambda$ the total input rate or load rate to AP. The maximum capacity of voice calls the AP can handle is $N_{max} = \max(\phi_{AP-VoIP}(N) > N\lambda)$.

From the graph we can find the largest N that satisfies this requirement where the input rate crosses the AP service rate. Using the G711 codec for voice calls, at 11 Mbps data rate, we note that the AP service rate crosses the input rate ($N\lambda$), after 12 calls. This implies that a maximum of 12 calls can be handled by AP (while meeting reasonable delays on a 802.11 WLAN). Similarly a maximum of 7 calls can be handled at 2Mbps PHY rate.

Plot2:

Shows attempt rate of a node verses number of VoIPcalls obtained from the simulation in the region of operation. Till the system reaches maximum capacity, the attempt rate is linear.

Sample Experiment 3

Objective:

Capacity evaluation of an infrastructure IEEE 802.11 based network carrying TCP controlled file downloads.(Ref: IEEE paper “Analytical Models for Capacity Estimation of IEEE 802.11 WLANs using DCF for Internet Applications” by George Kuriakose, Sri Harsha, Anurag Kumar and Vinod Sharma)

Theory:

Please navigate through the below given path to, Basics→Internetworks → Wireless LAN.

Wireless local area networks (WLANs) based on the IEEE 802.11 standard are being increasingly deployed in enterprises, academic campuses and homes, and at such places they are expected to become the access networks of choice for accessing the Internet. It therefore becomes important to study their ability to carry common Internet applications such as TCP controlled file downloading, or packet voice telephony.

We consider a single BSS WLAN with N Wireless STAs associated with a single AP. All STAs and AP contend for the channel via the DCF mechanism. Each STA has a single TCP with FTP traffic connection to download a large file from a local file server. Hence, the AP delivers TCP FTP data packets towards the STAs, while the STAs return TCP ACKs.

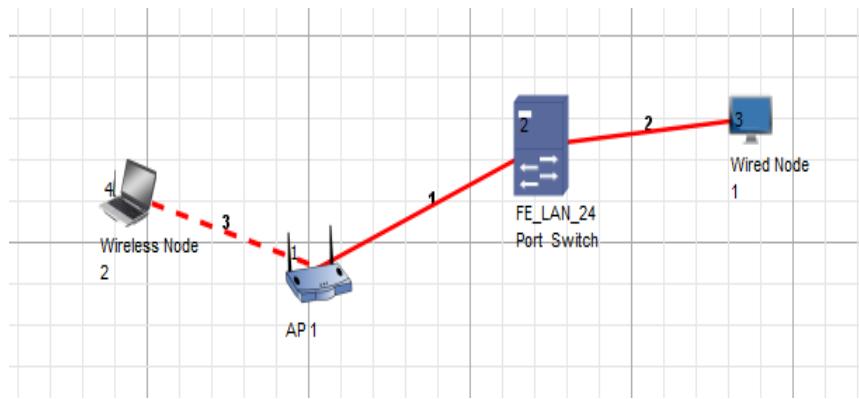
When downloading a file, RTS/CTS is used by the AP to send the data packets, and basic access is used by the STAs to send the ACKs.

Sample Inputs:

Here we are considering a network with AP, SWITCH, Wired Node, and Wireless nodes.

Experiment for 11Mbps:

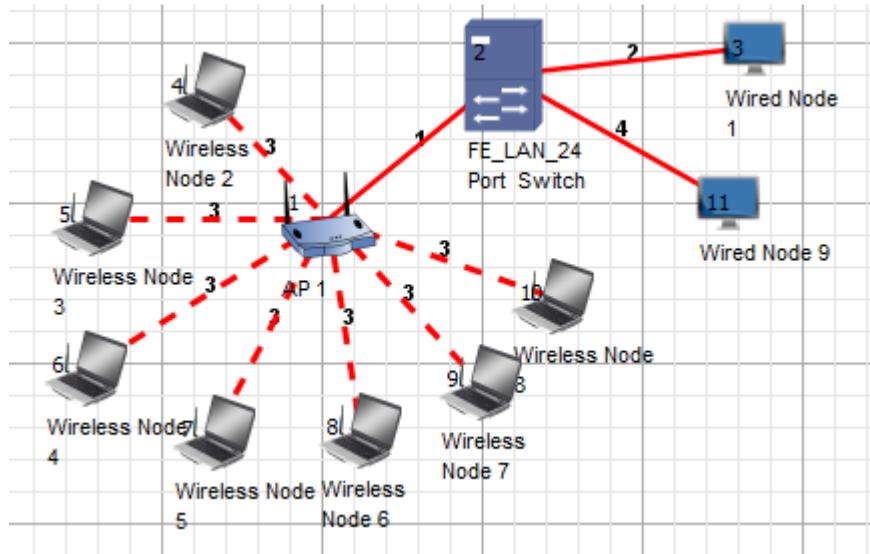
Experiment1_11Mbps: The Wired Node1 sends Application FTP traffic to Destination Wireless Node1 through SWITCH and AP. Set the properties as given below. Run the simulation and save the result.



Experiment2_11Mbps: Add one more Wireless Node 2 to AP. In the Wired Node Properties Add Application2 as FTP with Destination Wireless Node2. Continue the process of adding Wireless Nodes for each new experiment till Experiment 5.

Experiment6_11Mbps: Add Wireless Node6 to AP. Since the maximum number of application per node supported in NetSim is five (5), add another wired node to the switch as shown. Then set an FTP application from the new wired node to wireless node 6.

Continue the process of adding Wireless Nodes for each new experiment till Experiment 10.



Properties Settings:

Access Point Properties	AP 1
Protocol	IEEE802.11b
Buffer Size(MB)	5
RTS Threshold(Bytes)	1000
Retry Limit	7
Transmission Type	DSSS
Standard Channel	1_2412
Transmitter power	100
SIFS	10
Slot Time	20
CW Minimum	31

Link Properties	Wireless Links	Link Properties	Wired Links
Medium Type	WIRELESS	Medium Type	CAT -5
Uplink Speed	11	Uplink Speed	100
Downlink Speed	11	Downlink Speed	100
Frequency	2412	Uplink BER	No Error
Path Loss Exponent	0 or 2	Down Link BER	No Error
Fading Figure	0	Propagation Delay	Default
Standard Deviation	0	Standard Deviation	Default

Note: Path Loss Exponent if set to 2, to achieve 5.5 and 2 mbps transmission rate.

Place all the nodes with in 40 meter to get 11 Mbps, between 50 to 70 meter to achieve 5.5 Mbps and 70 to 120m to achieve 2 Mbps.

Node Properties	Wired Node1	Wired Node2				
Transmission	Point-to-Point	Point-to-Point	Point-to-Point	Point-to-Point	Point-to-Point	Point-to-Point
Destination	Wireless Node 1	Wireless Node 2	Wireless Node3	Wireless Node4	Wireless Node5	Wireless Node-6
Applications	Application1	Application2	Application3	Application4	Application5	Application1
Traffic Type	FTP	FTP	FTP	FTP	FTP	FTP
File Size						
Distribution	Constant	Constant	Constant	Constant	Constant	Constant
File Size	10000000	5000000	5000000	5000000	2500000	2500000
File Inter Arrival Time						
Distribution	Constant	Constant	Constant	Constant	Constant	Constant
File Inter Arrival Time	10	10	10	10	10	10

Keep All Wireless node properties as default.

Simulation Time- 20 Sec

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

- Set the properties for the Wireless Nodes, AP, switch and Links.
- Click on the Validate & Simulate button).

Experiment for 5.5Mbps:

Change the Path Loss Exponent of Wireless link to 2. Place Wireless Nodes between 50 to 70 meters. Repeat the same 10 experiment for data rate 5.5Mbps.

Experiment for 2Mbps:

Change the Path Loss Exponent of Wireless link to 2. Place Wireless Nodes between 80 to 120 meters. Repeat the same 10 experiment for data rate 2Mbps.

Run the simulation save the results in an XL sheet by exporting to csv.

The **aggregate throughput** of the AP is the main performance metric. This denotes the total number of AP successes per unit time. The accuracy of the model can also be checked by finding the conditional collision probability defined as the probability that an attempt of the AP fails due to a collision. The aggregate throughput (in Mbps) of the AP calculated by considering the “PayloadTransmitted” in the wireless link of “Link Metrics”.

$$\text{Aggregate Throughput} = \frac{\text{Payload Transmitted} * 8}{\text{Time}}$$

Time = Simulation Time – FTP File Inter arrival time. In Microseconds

The **collision probability** is the ratio of number of collisions occurred over total number of attempts made by nodes and AP.

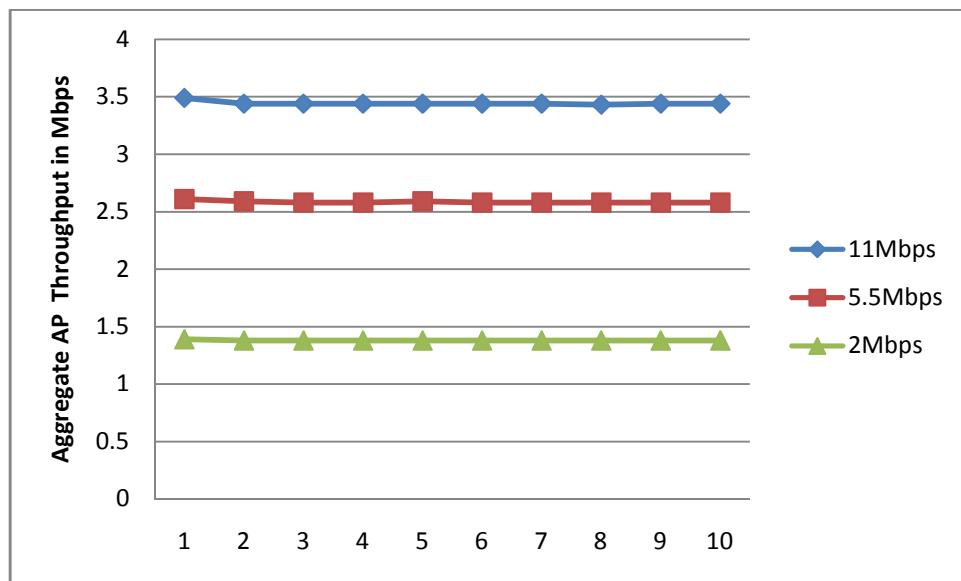
$$\text{Collision Probability} = \frac{\text{Packets_Collided}}{\text{Total Attempts}}$$

Total Attempts = TransmittedUnicastFrames + TransmittedBroadcastFrames +RTSFailureCount.

Plot1:

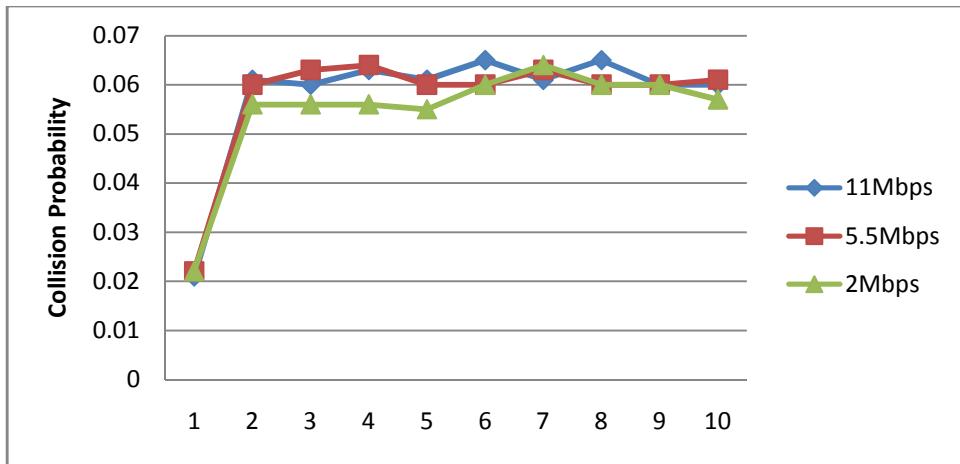
Aggregate Throughput = (Payload Transmitted * 8) / Time in Micro seconds						
Time = Simulation Time - File Interarrival time = 20 - 10 = 10 Secs = 10000000 Microsecs						
	11Mbps		5.5Mbps		2Mbps	
Experiments	Payload Transmitted	Aggregate Throughput Mbps	Payload Transmitted	Aggregate Throughput Mbps	Payload Transmitted	Aggregate Throughput Mbps
Experiment1	4363940	3.49	3266020	2.61	1740320	1.39
Experiment2	4308460	3.44	3242660	2.59	1730100	1.38
Experiment3	4307000	3.44	3235360	2.58	1730100	1.38
Experiment4	4302620	3.44	3230980	2.58	1728640	1.38
Experiment5	4305540	3.44	3238280	2.59	1730100	1.38
Experiment6	4296780	3.44	3232440	2.58	1728640	1.38

Experiment7	4301160	3.44	3228060	2.58	1724260	1.38
Experiment8	4289480	3.43	3232440	2.58	1725720	1.38
Experiment9	4301160	3.44	3229520	2.58	1724260	1.38
Experiment10	4307000	3.44	3229520	2.58	1724260	1.38



Plot2:

Experiments	11Mbps			5.5Mbps			2Mbps		
	Total Attempts	Collided Packets	Collision Probability	Total Attempts	Collided Packets	Collision Probability	Total Attempts	Collided Packets	Collision Probability
Experiment1	6104	126	0.021	4583	104	0.022	2441	54	0.022
Experiment2	6301	389	0.061	4733	280	0.06	2521	141	0.056
Experiment3	6294	379	0.06	4750	302	0.063	2527	142	0.056
Experiment4	6313	400	0.063	4752	305	0.064	2529	142	0.056
Experiment5	6310	388	0.061	4737	279	0.06	2528	133	0.055
Experiment6	6328	414	0.065	4738	279	0.06	2552	157	0.06
Experiment7	6314	389	0.061	4759	302	0.063	2568	171	0.064
Experiment8	6334	419	0.065	4747	278	0.06	2562	161	0.06
Experiment9	6319	383	0.06	4755	285	0.06	2562	155	0.06
Experiment10	6299	348	0.06	4766	291	0.061	2559	146	0.057



Inference:

Plot1:

This plot shows the capacity of the Access Point which is nothing but the saturation throughput of AP. The aggregate TCP throughput through the AP is plotted as a function of the number of FTP connections. It shows that irrespective of the number of FTP connections the aggregate throughput through AP is constant. In other words this is the maximum transfer rate of data through the system. The throughput is shown for all three data rates.

Plot2:

This shows that the collision probability (probability that a packet transmission fails due to collision) is almost equal to 6%. One can note that the collision probability is independent of PHY rates. This collision probability insensitivity with the PHY rate is as expected, since the contention process does not depend on PHY rates. The 6 % collision probability is a general rule of thumb used for verifications of 802.11 based wireless systems.

Routing

Sample Experiment 1:

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

Theory:

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 datagrams 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

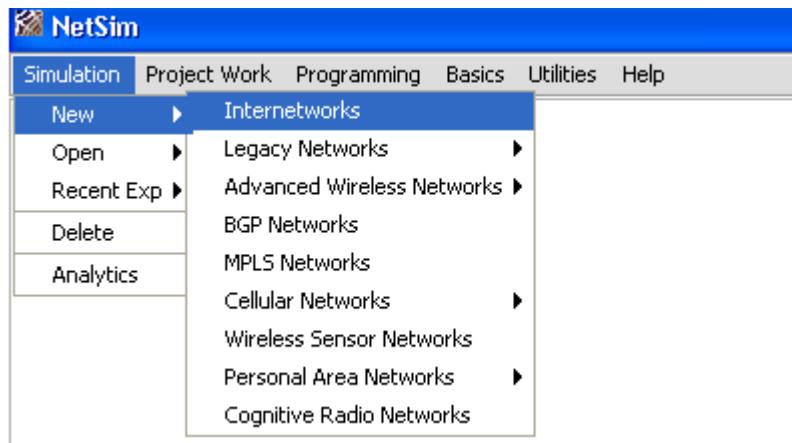
Simulation:

How to Create Scenario & Generate Traffic:

Sample 1:

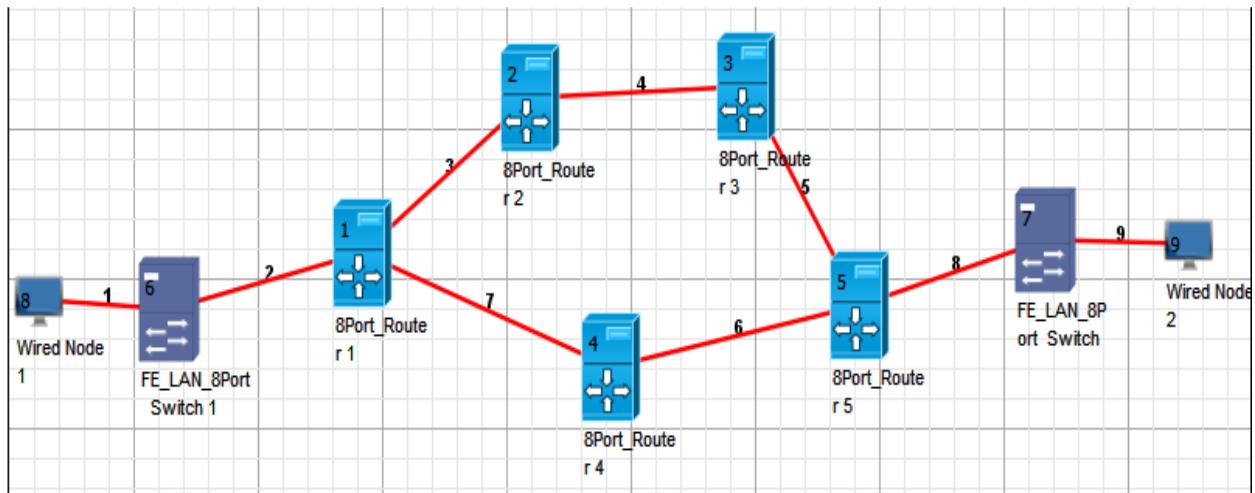
Step1:

Simulation → New → Internetworks



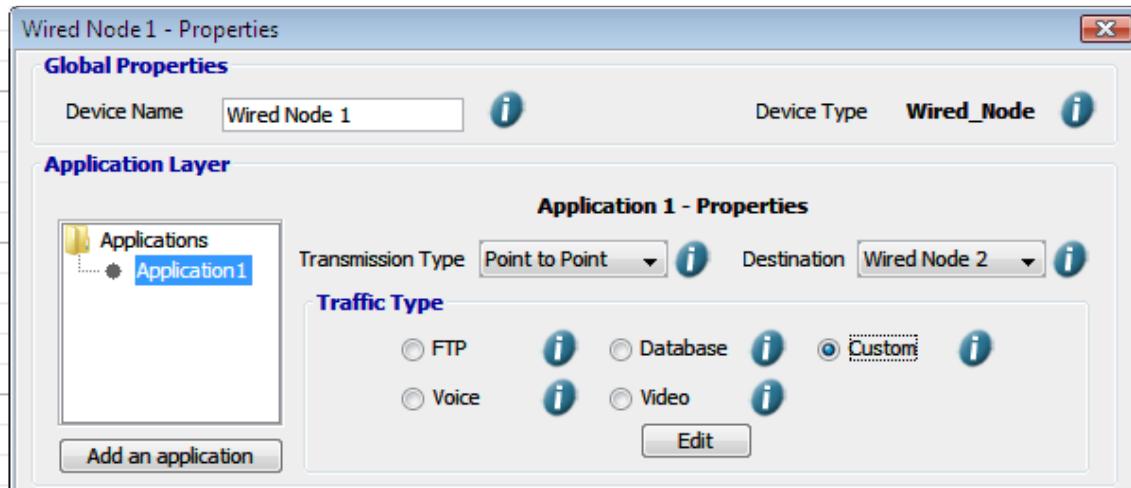
Step2:

Click & drop Routers, Switches and Nodes onto the Simulation Environment as shown

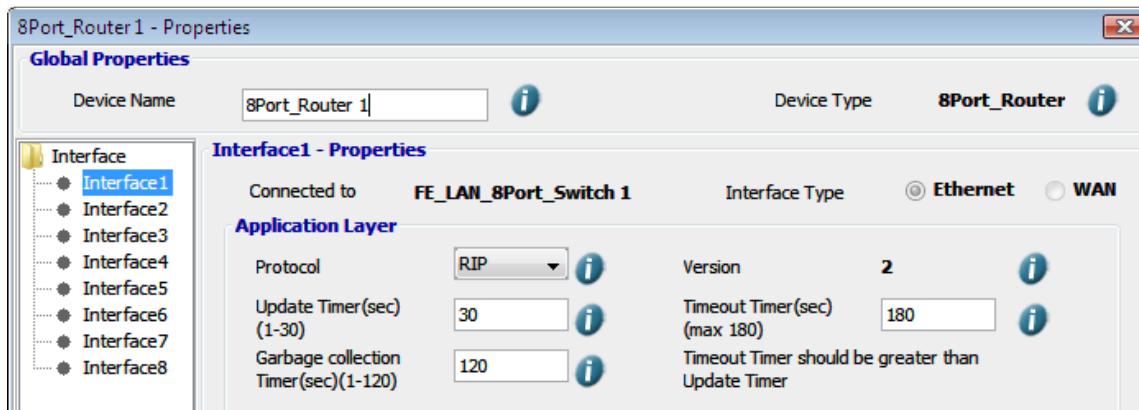


Step3:

Set the properties of the WiredNode1 as follows:



Set the properties of the 8Port_Router1 as follows:



Switch Properties: Accept default properties for Switch.

Link Properties: Accept default properties for Link.

Simulation Time - 100 Sec

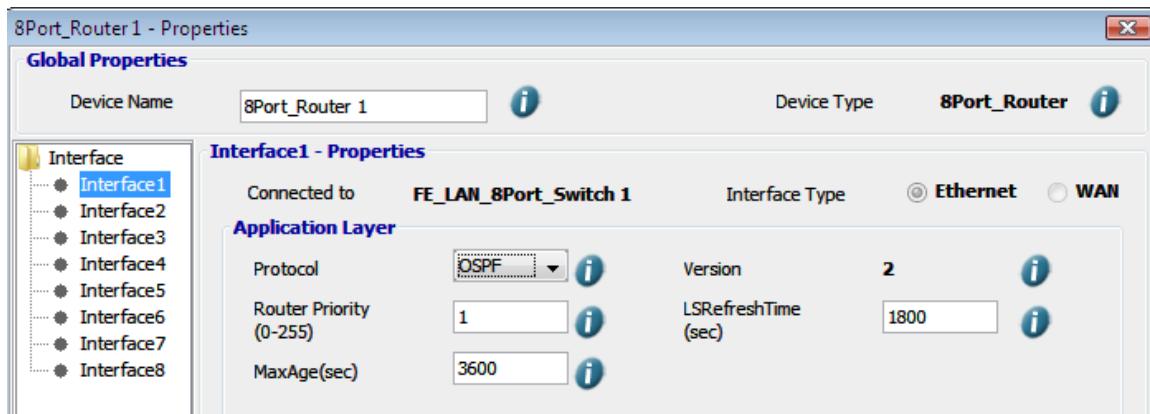
(Note: The Simulation Time can be selected only after doing the following two tasks,

- Set the properties of Node, Switch & Router
- Then click on the Simulate button).

Sample 2:

To model a scenario follows the same steps given in Sample1 and

Set the Router properties as given below



Link Properties:

	Link 3	Link 4	Link 5	Link 6	Link 7
Medium Type	CAT - 5				
Uplink Speed	100	100	100	10	10
Downlink Speed	100	100	100	10	10

Switch Properties: Accept default properties for Switch.

Simulation Time - 100 Sec.

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

- Set the properties of Node, Switch & Router
- Then click on the Simulate button).

Inference:

RIP

In Distance vector routing, each router periodically shares its knowledge about the entire network with its neighbors. The three keys to 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 that router'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

Initial Table: This table will show the direct connections made by each Router.

Intermediate Table: The Intermediate table will have the updates of the Network in every 30 seconds

Final Table: This table is formed when there is no update in the Network.

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

The RIP table in NetSim

- After running sample1, click RIP table in Performance Metrics screen. Then click the router to view the Routing table.
- We have shown the routing table for Router 1,

NetSim - Simulation - Internetworks - t

Simulation Project Work Programming Basics Utilities Help

Save Print

Performance Metrics

Router1

Initial

Dest IP	Next Hop IP	Interface	Metric	Timer
192.2.1.0	192.2.1.1	1	0	0.000000
11.3.0.0	11.3.1.1	2	0	0.000000
14.6.0.0	14.6.1.1	3	0	0.000000

Intermediate

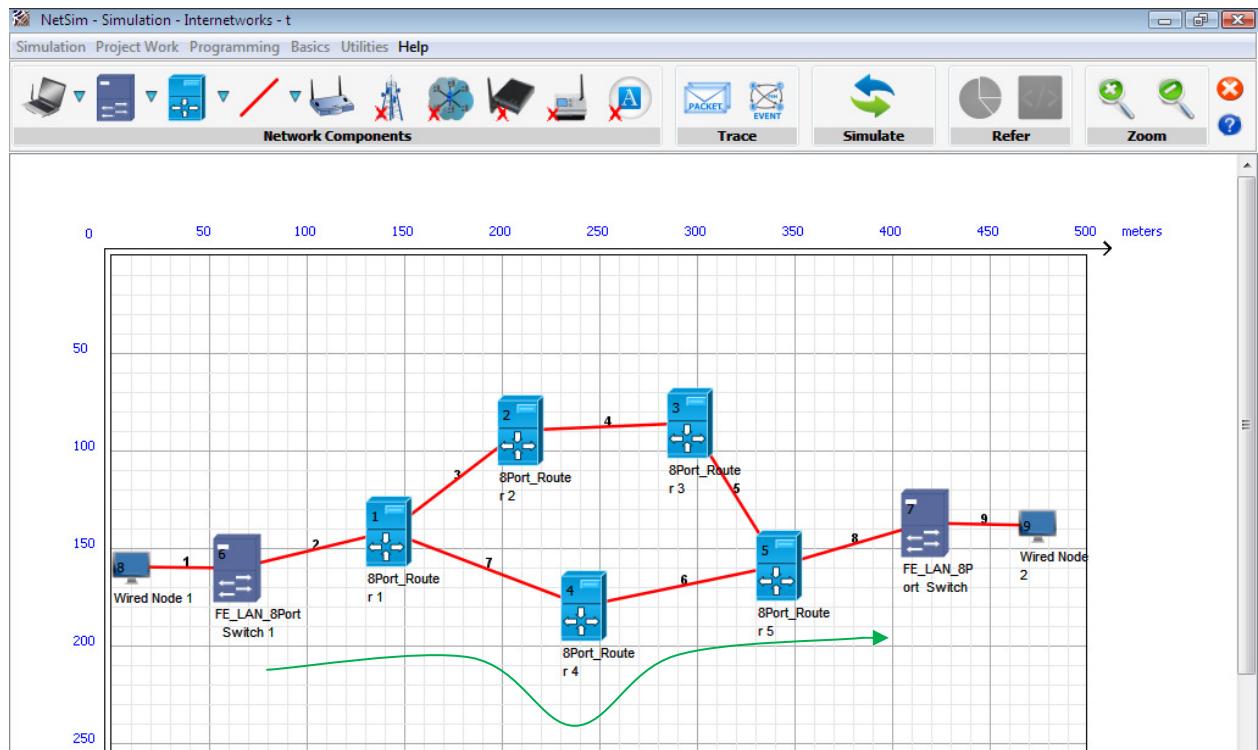
Dest IP	Next Hop IP	Interface	Metric	Timer
192.2.1.0	192.2.1.1	1	0	0.000000
11.3.0.0	11.3.1.1	2	0	0.000000
14.6.0.0	14.6.1.1	3	0	0.000000
12.4.0.0	11.3.1.1	2	1	53.120000

Final

Dest IP	Next Hop IP	Interface	Metric	Timer
12.4.0.0	11.3.1.1	2	1	53.120000
15.7.0.0	14.6.1.1	3	1	53.120000
13.5.0.0	14.6.1.1	3	2	90000043.200000
192.6.3.0	14.6.1.1	3	2	90000043.200000

Shortest Path from Wired_Node1 to Wired_Node2 in RIP

Wired_Node1 → Switch1 → Router1 → Router4 → Router5 → Switch2 → Wired_Node2



OSPF

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

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

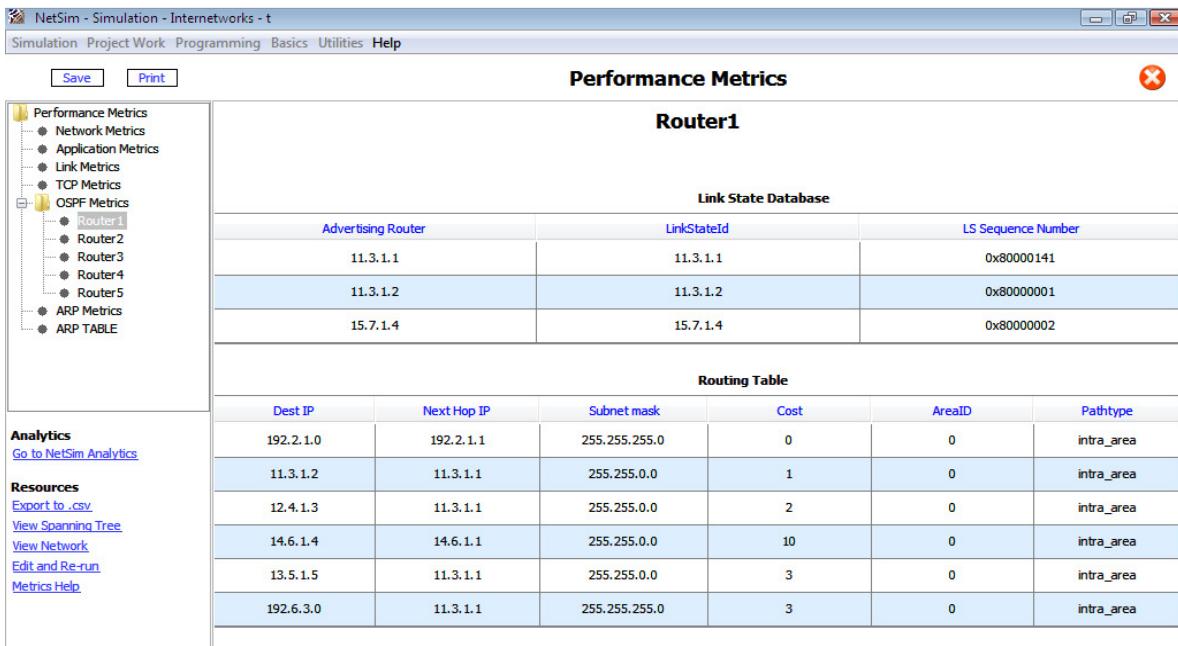
Once the LSDB is compiled, each OSPF router performs a least cost path calculation called the Dijkstra algorithm on the information in the LSDB and creates a tree of shortest paths to each other router and network with themselves as the root. This tree is known as the SPF Tree and contains a single,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

The OSPF table in NetSim

- After running sample2, click OSPF Metrics in Performance Metrics screen. Then click the router to view the Routing table
- We have shown the routing table for Router 1



Shortest Path from Wired_Node1 to Wired_Node2 in OSPF

Wired_Node1 → Switch1 → Router1 → Router2 → Router3 → Router5 → Switch2 → Wired_Node2

Note:

The Cost is calculated by using the following formula

$$Cost = \frac{\text{Reference Bandwidth}}{\text{Link Speed Up}}$$

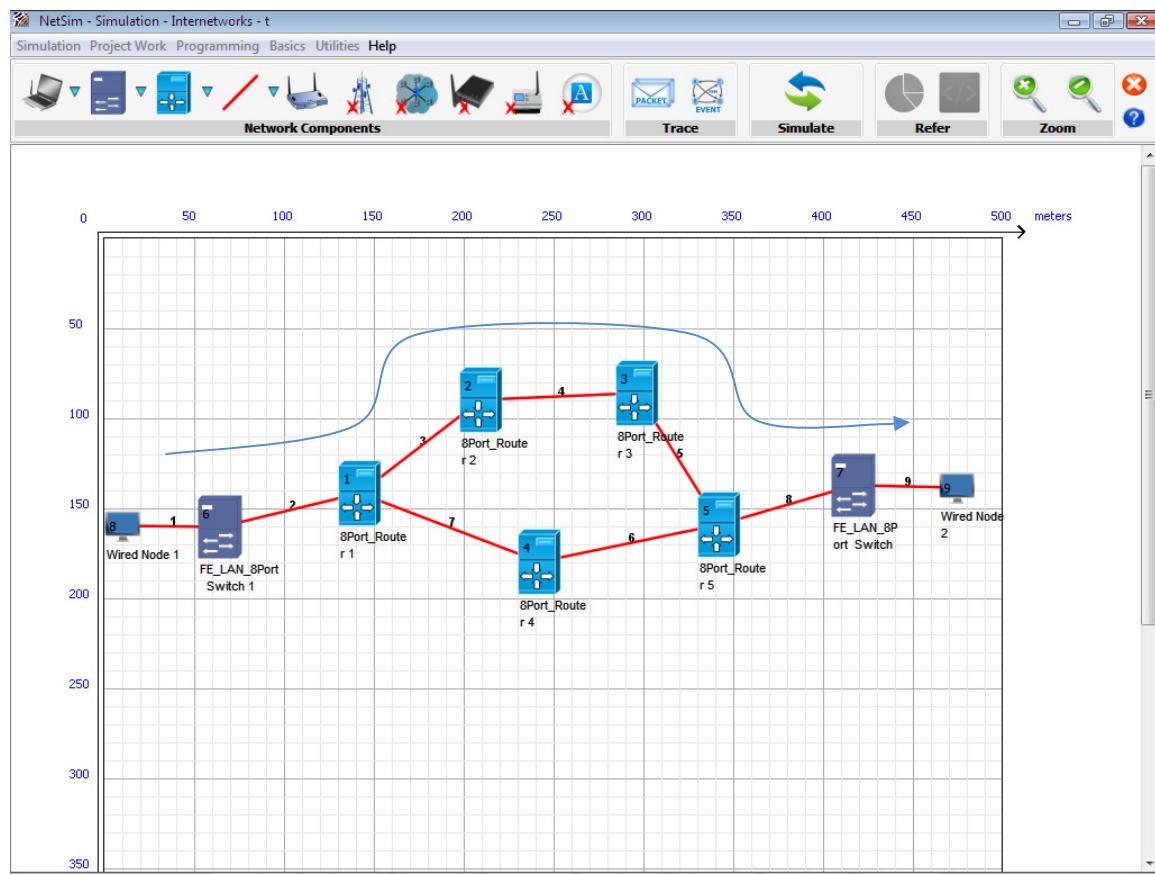
Reference Bandwidth = 100 Mbps

For Example,

Let us take,

Link Speed UP = 100 Mbps

$$Cost = \frac{100 \text{ (Reference Bandwidth)}}{100 \text{ (Link Speed UP)}} = 1$$



Spanning Tree Protocol

Sample Experiment 1

Objective:

Study the working of the spanning tree algorithm by varying the priority among the switches.

Theory:

Refer NetSim Basics on Spanning Tree.

Procedure:

How to Create Scenario & Generate Traffic:

Please navigate through the below given path to,

- **Create Scenario:** “Help → NetSim Help F1 → Running Simulation via GUI → Internetworks → Create Scenario”.

Sample Inputs:

- To do this Sample experiment, 3 Wired Nodes and 3 Switches are needed.
(Note: Minimum three switches are needed in the simulation to study about spanning tree formation)
- Upon completion of the experiment, “Save” it for further comparisons that can be carried out in the “Analytics” section.

Inputs for the Sample experiments are given below,

Sample Input1:

- Click and drop 3 Switches and 3 Wired Nodes onto the Environment
(Note: The Switch may be an 8, 16 or 24 port switch)
- Connect Switch 1 to Switch 2
- Connect Switch 2 to Switch 3
- Connect Switch 3 to Switch 1
- Connect Wired Node 1 to Switch 1
- Connect Wired Node 2 to Switch 3
- Connect Wired Node 3 to Switch 2

Node Properties	Wired Node 1		Wired Node 2	
Transmission Type	Point to Point		Point to Point	
Destination	Wired Node2		Wired Node1	
Traffic Type	Custom		Custom	
Packet Size	Distribution	Constant	Distribution	Constant
	Packet Size (Bytes)	1472	Packet Size (Bytes)	1472
Packet Inter Arrival Time(ms)	Distribution	Constant	Distribution	Constant
	Packet Inter Arrival Time (ms)	1000	Packet Inter Arrival Time (ms)	1000

(Note: Wired Node 3 is not generating Traffic to any other Wired Nodes)

Switch Properties	FE_LAN_8_Port Switch 1	FE_LAN_8_Port Switch 2	FE_LAN_8_Port Switch 3
Switch Priority	2	1	3

(Note: All other properties of Switch is default)

Simulation Time - 10 Sec

Sample Input2:

- Click and drop 2 Wired Nodes and 3 Switches onto the Environment
- Connect Switch 1 to Switch 2
- Connect Switch 2 to Switch 3
- Connect Switch 3 to Switch 1
- Connect Wired Node 1 to Switch 1
- Connect Wired Node 2 to Switch 3
- Connect Wired Node 3 to Switch 2

Node Properties	Wired Node 1		Wired Node 2	
Transmission Type	Point to Point		Point to Point	
Destination	Wired Node2		Wired Node1	
Traffic Type	Custom		Custom	
Packet Size	Distribution	Constant	Distribution	Constant
	Packet Size (Bytes)	1472	Packet Size (Bytes)	1472

Packet Inter	Distribution	Constant	Distribution	Constant
Arrival Time(ms)	Packet Inter	1000	Packet Inter	1000

(Note: Wired Node 3 is not generating Traffic to any other Wired Nodes)

Switch Properties	FE_LAN_8_Port Switch 1	FE_LAN_8_Port Switch 2	FE_LAN_8_Port Switch 3
Switch Priority	1	2	3

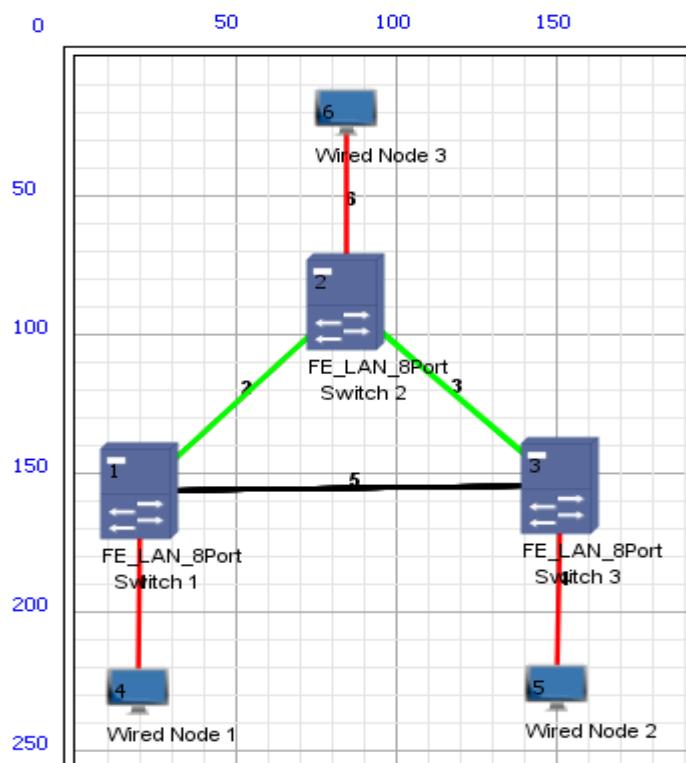
(Note: All other properties of Switch is default)

Simulation Time - 10 Sec

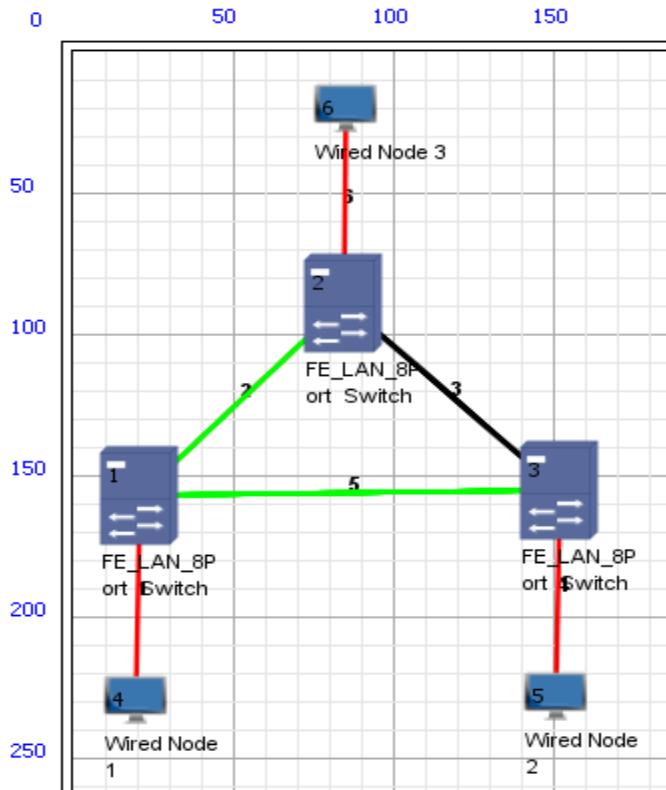
Output:

- To view the output, click the View Spanning Tree Link available on the Performance Metrics screen under Resources.

Sample1:



Sample2:



Inference:

In the Sample 1, FE_LAN_8 Port_Switch 2 was assigned least priority and was selected as a Root switch. The Green line indicates the forward path and the Black line indicates the blocked path. The frame from Wired Node1 should take the path through the FE_LAN_8 Port_Switch 2 to reach the Wired Node2.

In the Sample 2, FE_LAN_8 Port_Switch 1 was assigned least priority and selected as a Root switch. In this case, the frame from Wired Node1 can directly reach the destination Wired Node2.

NetSim – Slotted Aloha

Sample Experiments - User can understand the internal working of Slotted Aloha through these sample experiments. Each sample experiment covers:

- Procedure
- Sample Inputs
- Output
- Comparison chart
- Inference

Index	Objective
Experiment 1	Study the Throughput characteristics of a Slotted ALOHA network

Sample Experiment 1

Objective: Plot the characteristic curve throughput versus offered traffic for a Slotted ALOHA system.

(Reference: Computer Networks, 3rd Edition. Andrew S. Tanenbaum)

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.

Slotted ALOHA:

In slotted Aloha, time is divided up into discrete intervals, each interval corresponding to one frame. In Slotted ALOHA, a computer is required to wait for the beginning of the next slot in order to send the next packet. The probability of no other traffic being initiated during the entire vulnerable period is given by e^{-G} which leads to $S = G e^{-G}$

Where, S (frames per frame time) is the mean of the Poisson distribution with which frames are being generated. For reasonable throughput S should lie between 0 and 1.

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

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

$$G = \frac{TA * PT}{ST * 1000}$$

Where, G = Attempts per packet time

TA = Total Attempt

PT = Packet time (in milli seconds)

ST = Simulation time (in seconds)

The throughput (mbps) per frame time can be obtained as follows:

$$S = \frac{\text{Throughput(in Mbps)} * 1000 * PT}{PS * 8}$$

Where, S = Throughput per frame time

PT = Packet time (in milli seconds)

PS = Packet size (in bytes)

Calculations for the packet time:

$$PT = \frac{\text{Packet Size(bits)}}{\text{Band Width(Mbps)}}$$

In the following experiment we have taken packet size=1498 bytes (1472 + 26 Overheads)

Bandwidth is 10 Mbps and hence, packet time comes as 1.198 ms.

Procedure:

How to Create Scenario:

- **Create Scenario:** “Help → NetSim Help → Running Simulation via GUI → Legacy Networks → Create Scenario”.

Obtain the values of Throughput and Total Attempts from the statistics of NetSim simulation for various numbers of traffic generators.

Sample Inputs:

Input for Sample 1: Node 1 transmits data to Node 2.

Node Properties	NODE 1
Transmission	Point-to-Point
Destination	Node-2
Traffic Type	Data
Application Data Size	
Distribution	Constant
Application Data Size (Bytes)	1472
Inter Arrival Time	
Distribution	Constant
Inter Arrival Time	20000

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 the Validate & Simulate button).

Input for Sample 2: Node 1 transmits data to Node 2, and Node 2 transmits data to Node 1.

Node Properties	NODE 1	NODE 2
Transmission	Point-to-Point	Point-to-Point
Destination	Node-2	Node-1
Traffic Type	Data	Data
Application Data Size		
Distribution	Constant	Constant
Application Data Size (Bytes)	1472	1472
Inter Arrival Time		
Distribution	Constant	Constant
Inter Arrival Time	20000	20000

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 the Validate & Simulate button).

Experiment 1: Node 1 transmits data to Node 2.

Experiment 2: Node 1 transmits data to Node 2, and Node 2 transmits data to Node 1.

Experiment 3: Node 1 transmits data to Node 2, and Node 2 transmits data to Node 3, and Node 3 transmits data to Node 1.

And so on do the experiment by increasing the number of nodes generating traffic as 4, 5, 7, 9, 10, 15, 20 22 and 24 nodes.

Simulation Time - 10 Seconds

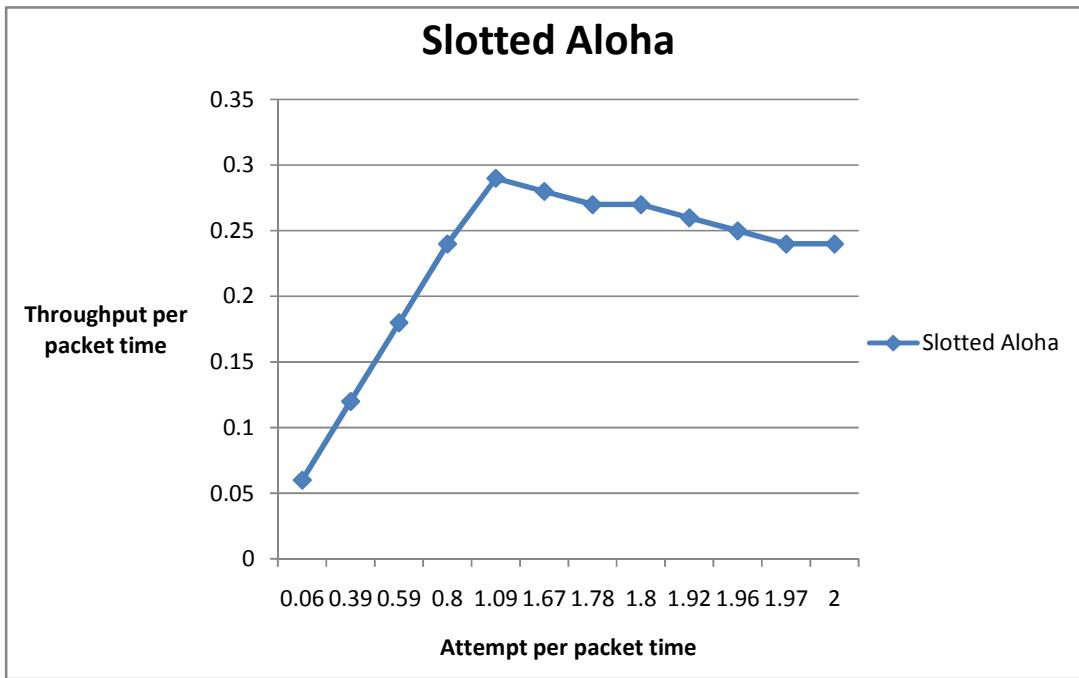
(Note: The Simulation Time can be selected only after doing the following two tasks,

- Set the properties of Nodes
- Then click on the Validate & Simulate button).

Comparison Table:

Number of nodes generating traffic	Throughput (mbps)	Total attempts	Throughput per packet time	Attempts per packet time
1	0.59	499	0.06	0.06
2	1.2	3308	0.12	0.39
3	1.8	4953	0.18	0.59
4	2.4	6691	0.24	0.80
5	2.9	9180	0.29	1.09
7	2.8	14012	0.28	1.67
9	2.7	14868	0.27	1.78
10	2.7	15078	0.27	1.80
15	2.6	16037	0.26	1.92
20	2.5	16437	0.25	1.96
22	2.4	16496	0.24	1.97
24	2.4	16755	0.24	2.00

We have obtained the following characteristic plot for the Slotted ALOHA, which matches the theoretical result.



Note: The optimum value is slightly less than the theoretical maximum of 0.368 because NetSim's simulation is per real-world and includes overheads, inter-frame gaps etc.

NetSim - Traditional Ethernet

Sample Experiments - User can understand the internal working of Traditional Ethernet through these sample experiments. Each sample experiment covers:

- Procedure
- Sample Inputs
- Output
- Comparison chart
- Inference

Index	Objective
Experiment 1	Understand the impact of bit error rate on packet error rate and investigate the impact of error of a simple hub based CSMA / CD network
Experiment 2	To determine the optimum persistence of a p-persistent CSMA / CD network for a heavily loaded bus capacity.

Sample Experiment 1

Objective:

Understand the impact of Bit error rate on packet error rate and investigate the impact of error of a simple hub based CSMA / CD network

Theory:

Bit error rate (BER): The bit error rate or bit error ratio is the number of bit errors divided by the total number of transferred bits during a studied time interval i.e.

$$\text{BER} = \frac{\text{Bit errors}}{\text{Total number of bits}}$$

For example, a transmission might have a BER of 10^{-5} , meaning that on average, 1 out of every of 100,000 bits transmitted exhibits an error. The BER is an indication of how often a packet or other data unit has to be retransmitted because of an error.

Unlike many other forms of assessment, bit error rate, BER assesses the full end to end performance of a system including the transmitter, receiver and the medium between the two. In this way, bit error rate, BER enables the actual performance of a system in operation to be tested.

Bit error probability (p_e): The bit error probability is the expectation value of the BER. The BER can be considered as an approximate estimate of the bit error probability. This estimate is accurate for a long time interval and a high number of bit errors.

Packet Error Rate (PER):

The PER is the number of incorrectly received data packets divided by the total number of received packets. A packet is declared incorrect if at least one bit is erroneous. The expectation of the PER is denoted as packet error probability p_p , which for a data packet length of N bits can be expressed as,

$$p_p = 1 - (1 - p_e)^N$$

It is based on the assumption that the bit errors are independent of each other.

Derivation of the packet error probability:

Suppose packet size is N bits.

p_e is the bit error probability then probability of no bit error=1- p_e

As packet size is N bits and it is the assumption that the bit errors are independent. Hence,

Probability of a packet with no errors = $(1 - p_e)^N$

A packet is erroneous if at least there is one bit error, hence

Probability of packet error=1- $(1 - p_e)^N$

Procedure:

How to create a scenario and generate traffic:

- **Create Scenario:** “Help → NetSim Help → Running Simulation via GUI→ Non-Internetworks → Legacy Networks → Create Scenario”.

Example 1:

Create samples by varying the bit error rate (10^{-6} , 10^{-7} , 10^{-8} , 10^{-9} , No error) and check whether packet error output matches the PER formula.

Sample Inputs:

In this sample experiment, two nodes and one hub are dragged and dropped on the environment builder. And following experiments are performed.

Inputs for the sample experiments are given below,

- Drag and drop Hub on the environment builder.
- Drag and drop Node1 over the hub.
- Drag and drop Node2 over the hub.

Node Properties	Node1
Transmission Type	Point to Point
Destination	2
Application Data Size (Bytes)	1472
Distribution	Constant
Inter Arrival Time (μs)	2500
MTU Size(Bytes)	1500

Hub Properties	Sample1	Sample 2	Sample 3	Sample 4	Sample 5
Data rate (Mbps)	10	10	10	10	10
Error Rate (BER)	No Error	10^{-9}	10^{-8}	10^{-7}	10^{-6}
Physical Medium	Twisted Pair	Twisted Pair	Twisted Pair	Twisted Pair	Twisted Pair

Simulation Time - 1000 Sec

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

- Set the properties for the Node1 & The Hub
- Click on the Validate & Simulate button).

NetSim simulation output:

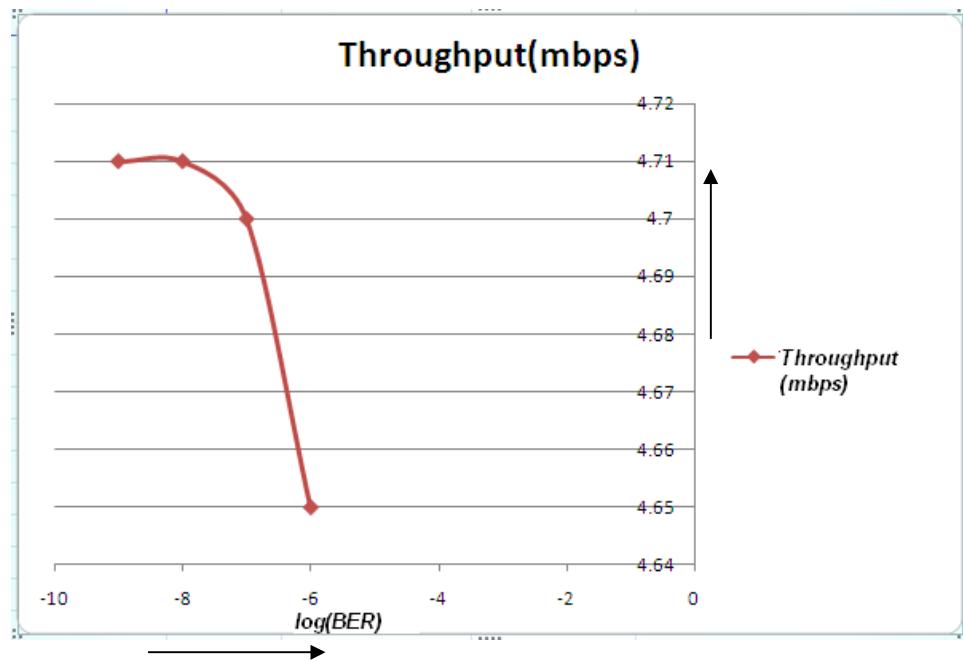
BER	Error packets	Total packets	PER(Error packets/Total packets)	PER-using the formula $p_p = 1 - (1 - p_e)^N$	Throughput (mbps)
0	0	400000	0	0	4.71
1E-09	4	400000	0.00001	0.000011999	4.71
1E-08	57	400000	0.0001425	0.000119993	4.71
1E-07	470	400000	0.001175	0.00119928	4.7
1E-06	4730	400000	0.011825	0.011928293	4.65

Packet size for the calculation of the output table=1500 bytes or 12000 bits

Inference:

It can be observed from the table that PER calculated from the formula and PER obtained from the definition are of the same order and in agreement as the number of error packets increase (for higher BER). This is because the errors are generated randomly in NetSim. As the run time of the simulation is increased the accuracy of NetSim also increases.

Comparing Throughput (mbps) with Bit error rate:



Inference:

We note from the above plot that when there is slight increase in bit error rate from 0, throughput does not change up to two decimal places, but as bit error rate is further increased, throughput starts to decrease, which is expected.

Sample Experiment 2

Objective:

To determine the optimum persistence of a p-persistent CSMA / CD network for a heavily loaded bus capacity.

Theory:

Carrier Sense Multiple Access Collision Detection (CSMA / CD)

This protocol includes the improvements for stations to abort their transmissions as soon as they detect a collision. Quickly terminating damaged frames saves time and bandwidth. This protocol is widely used on LANs in the MAC sub layer. If two or more stations decide to transmit simultaneously, there will be a collision. Collisions can be detected by looking at the power or pulse width of the received signal and comparing it to the transmitted signal. After a station detects a collision, it aborts its transmission, waits a random period of time and then tries again, assuming that no other station has started transmitting in the meantime.

There are mainly three theoretical versions of the CSMA /CD protocol:

1-persistent CSMA / CD: When a station has data to send, it first listens to the channel to see if anyone else is transmitting at that moment. If the channel is busy, the station waits until it becomes idle. When station detects an idle channel, it transmits a frame. If a collision occurs, the station waits a random amount of time and starts all over again. The protocol is called 1-persistent because the station transmits with a probability of 1 whenever it finds the channel idle. Ethernet, which is used in real-life, uses 1-persistence. A consequence of 1-persistence is that, if more than one station is waiting for the channel to get idle, and when the channel gets idle, a collision is certain. Ethernet then handles the resulting collision via the usual exponential back off. If N stations are waiting to transmit, the time required for one station to win the back off is linear in N.

Non-persistent CSMA /CD: In this protocol, before sending, a station senses the channel. If no one else is sending, the station begins doing so itself. However, if the channel is already in use, the channel does not continually sense it for the purpose of seizing it immediately upon detecting the end of the previous transmission. Instead, it waits a random period of time and then repeats the algorithm. Intuitively this algorithm should lead to better channel utilization and longer delays than 1-persistent CSMA

p-persistent CSMA / CD: This protocol applies to slotted channels. When a station becomes ready to send, it senses the channel. If it is idle, it transmits with a probability of p. With a probability $q=1-p$ it defers until the next slot. If that slot is also idle, it either transmits or defers again, with probabilities p and q respectively. This process is repeated until either the frame has been transmitted or another station has begun transmitting. In the latter case, it acts as if there had been a collision (i.e., it waits a random time and starts again). If the station initially senses the channel busy, it waits until the next slot and applies the above algorithm.

How does the performance of LAN (throughput) that uses CSMA/CD protocol gets affected as the numbers of logged in user varies:

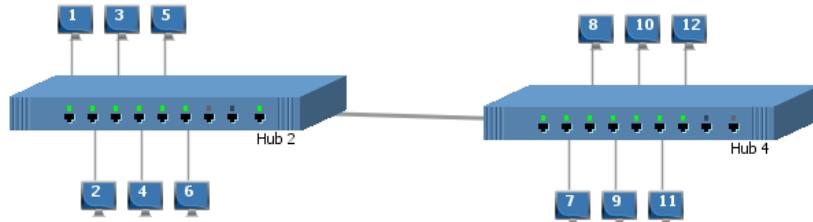
Performance studies indicate that CSMA/CD performs better at light network loads. With the increase in the number of stations sending data, it is expected that heavier traffic have to be carried on CSMA/CD LANs (IEEE 802.3). Different studies have shown that CSMA/CD performance tends to degrade rapidly as the load exceeds about 40% of the bus capacity. Above this load value, the number of packet collision raise rapidly due to the interaction among repeated transmissions and new packet arrivals. Collided packets will back off based on the truncated binary back off algorithm as defined in IEEE 802.3 standards. These retransmitted packets also collided with the newly arriving packets.

Procedure:

How to create a scenario and generate traffic:

- **Create Scenario:** “Help → NetSim Help → Running Simulation via GUI→ Legacy Networks → Create Scenario”.

Scenario:



Sample Input:

In this Sample experiment 12 Nodes and 2 Hubs need to be dragged and dropped onto the Environment Builder.

Input for the Sample experiments (i.e. Totally 11 Samples) are given below,

Sample Input 1:

In the first sample for each Node the following properties have to be set,

Node Properties	Values to be Selected
Transmission Type	Broadcast
Traffic Type	Data
No. of Nodes Transmitting	12
Persistence	1
MTU Size(Bytes)	1500

Vary persistence from 1/2, 1/3, 1/4, 1/5... 1/12, to generate other experiments.

Data Input Configuration: (This window is obtained when Data is selected in Traffic Type):

Packet Size	Distribution	Constant
	Application Data Size (Bytes)	1472
Inter Arrival Time	Distribution	Exponential
	Mean Inter Arrival Time(Micro Sec)	1000

Hub Properties common for Hub1 and Hub2:

Hub Properties	Values to be Selected
Data Rate(Mbps)	10
Error Rate (bit error rate)	No error
Physical Medium	Twisted Pair

Simulation Time - 10 Sec

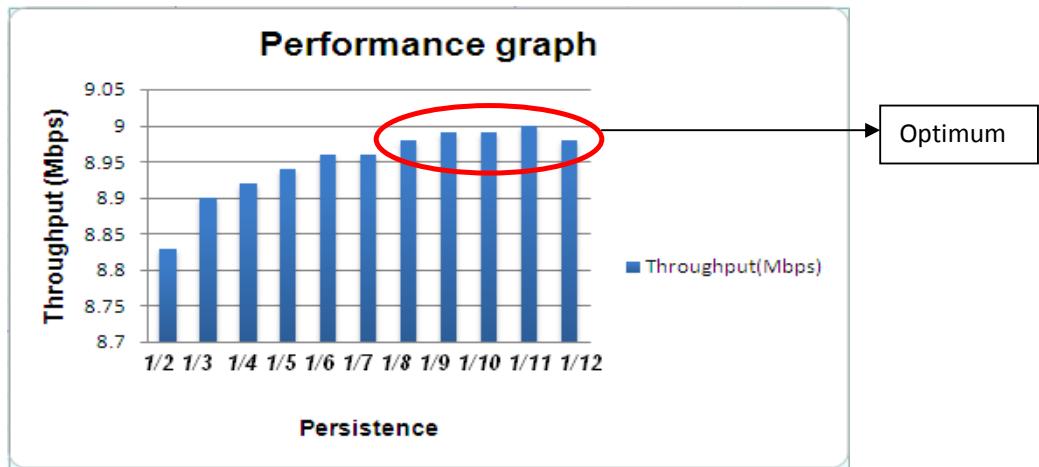
(Note: The Simulation Time can be selected only after the following two tasks,

- Set the properties for the Nodes & The Hub
- Click on the Validate & Simulate button).

Output:

After simulation of each experiment, click on the network statistics and note down the user level throughput values. Open an excel sheet and plot a graph for these noted values against their respective persistence values.

Comparison Chart:



Inference:

As the number of logged in users is quite large in this experiment, the performance of a p-persistent CSMA/CD network with large p, is not optimal because of a large number of

collisions. Therefore, we have minimum throughput when the persistence was 1/2. But as persistence is decreased (lower and lower probabilities), the likelihood of collisions reduce and hence throughput starts to increase. However, beyond a certain limit, in this case 1/11 the probability of transmitting packets becomes very low and hence there aren't many transmissions. Therefore, throughput starts to decline. In this experiment with 12 nodes generating traffic, we notice that the maximum throughput is at a persistence value lying between 1/9 and 1/11.

NetSim - ATM

Sample Experiments - User can understand the internal working of ATM through these sample experiments. Each sample experiment covers:

- Procedure
- Sample Inputs
- Output
- Comparison chart
- Inference

Index	Objective
Experiment 1	Study the effect of Peak Cell Rate (per Sec) and Cell Delay Variation Tolerance on the performance of an ATM Networks
Experiment 2	Study the performance of FIFO, round Robin and Priority queuing techniques in an ATM Networks

Sample Experiment 1

Objective

Study the effect of Peak Cell Rate (per Sec) and Cell Delay Variation Tolerance on the performance of an ATM Networks

Procedure:

How to Create Scenario & Generate Traffic:

Please navigate through the below given path to,

- **Create Scenario:** “Help → NetSim Help → Running Simulation via GUI→ Legacy Networks → Create Scenario”.

Inputs

Follow the steps given in the different samples to arrive at the objective.

In this Sample,

- Total no of CPEs used: 2
- Total no of Switches used: 2

The devices are inter connected as given below,

- CPE 1, CPE 2 is connected with Switch 1 and Switch 2 by Link 1 and Link 3 respectively.
- Switch 1 is connected with Switch 2 by Link 2.

Set the properties of Switch and CPE by following the tables for each sample,

Switch Properties	Switch 1	Switch 2
Scheduling Technique	Priority	Priority
Buffer Size (KB)	8	8

Link Properties	Link 1	Link 2	Link 3
Bit Error Rate	No Error	No Error	No Error
Physical Medium	E0	E0	E0
Distance (kms)	1	1	1

Inputs for Sample 1

CPE Properties	CPE 1
Destination	CPE 2
Transmission Type	Point to Point
Traffic Type	Data
Application Data Size	
Distribution	Exponential
Mean Application Data Size (Bytes)	1500
Inter Arrival Time	
Distribution	Exponential
Mean Inter Arrival Time (micro sec)	20000
Peak Cell Rate (cells/second)	1000
Cell Delay Variation Tolerance (micro secs)	1000

Inputs for Sample 2

CPE Properties	CPE 1
Destination	CPE 2
Transmission Type	Point to Point
Traffic Type	Data
Application Data Size	
Distribution	Exponential
Mean Application Data Size (Bytes)	1500
Inter Arrival Time	
Distribution	Exponential
Mean Inter Arrival Time (micro sec)	20000
Peak Cell Rate (cells/second)	2000
Cell Delay Variation Tolerance (micro secs)	2000

Inputs for Sample 3

CPE Properties	CPE 1
Destination	CPE 2
Transmission Type	Point to Point
Traffic Type	Data
Application Data Size	
Distribution	Exponential
Mean Application Data Size (Bytes)	1500
Inter Arrival Time	
Distribution	Exponential
Mean Inter Arrival Time (micro sec)	20000
Peak Cell Rate (cells/second)	3000
Cell Delay Variation Tolerance (micro secs)	3000

Inputs for Sample 4

CPE Properties	CPE 1
Destination	CPE 2
Transmission Type	Point to Point
Traffic Type	Data
Application Data Size	
Distribution	Exponential
Mean Application Data Size (Bytes)	1500
Inter Arrival Time	
Distribution	Exponential
Mean Inter Arrival Time (micro sec)	20000
Peak Cell Rate (cells/second)	4000
Cell Delay Variation Tolerance (micro secs)	4000

Inputs for Sample 5

CPE Properties	CPE 1
Destination	CPE 2
Transmission Type	Point to Point
Traffic Type	Data
Application Data Size	
Distribution	Exponential
Mean Application Data Size (Bytes)	1500
Inter Arrival Time	
Distribution	Exponential
Mean Inter Arrival Time (micro sec)	20000
Peak Cell Rate (cells/second)	5000
Cell Delay Variation Tolerance (micro secs)	5000

Simulation Time – 10 sec

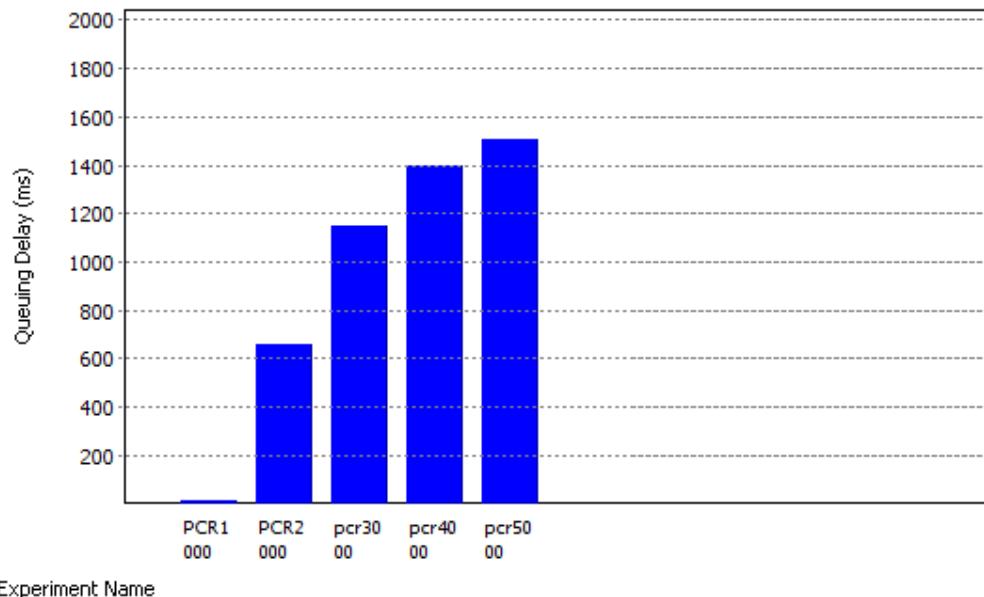
Output

To view the output by using NetSim Sample experiments need to be added onto the Analytics interface. Given below is the navigation for analytics - “Simulation → Analytics”.

Select the experiments by selecting

- Legacy Networks
- Select the Experiments (*Note:* Click one experiment after another to compare the experiments in the Analytics interface).
- Select the Metric: Queuing Delay (ms)

Comparison Charts:



Inference

To see Peak Cell Rate and Cell Delay Variation Tolerance definitions go to NetSim Basics Menu.

When the Peak Cell Rate and Cell Delay Variation Tolerance are changed from 1000 to 5000, the Number of Cells transmitted increases. This is because the conformance of the traffic depends on the Peak Cell Rate and the Cell Delay Variation Tolerance. As the number of cells transmitted increases, queuing delay also increases.

Sample Experiment 2

Objective

Network performance analysis with an ATM switch implementing different scheduling techniques like First in First out (FIFO), Priority, and Round Robin

Theory

In an ATM network, scheduling of cells is the major task of any ATM switch. Scheduling is the process by which the ATM switch determines the sequence of flow of the cells in network. The scheduling is done on various properties like service type, cell category, queue length, arrival time etc.

First In First out (FIFO): FIFO is the simplest way of scheduling. As the name suggests, in this technique, the preference is given to that cell which comes first in the queue irrespective of its priority value. The one which comes next waits until the first finishes. The drawback of this technique is that some cells of very high priority like audio service encounter extra delay that is not ignorable.

Priority: In this technique, each cell is assigned a certain priority value based on its traffic parameters. The scheduler checks the availability of highest priority cells and schedules them before going for the lower priority cells. The drawback of this algorithm is that cells of lowest priority starve for the resources when there are a large number of higher priority cells.

Round Robin: In this technique, the scheduler gives equal preference for all priority types. Therefore, scheduler processes one cell of each priority type (If available) before going for the next cell and cycles through them. Here starvation never occurs because no priority is given. Round robin scheduling may not be desirable if QoS of the different priority type are highly variable.

Procedure:

In this experiment, we are going to analyze the link Utilization (%) of the outgoing link from an ATM switch

First In First Out (FIFO):

In NetSim, Select Simulation → New → Legacy Networks → ATM

In the simulation environment window create the following scenario

How to Create Scenario & Generate Traffic:

Please navigate through the below given path to,

- **Create Scenario:** “Help → NetSim Help → Running Simulation via GUI → Legacy Networks → Create Scenario”.

Inputs

Follow the steps given in the different samples to arrive at the objective.

In all Samples,

- Total no of CPE used: 4
- Total no of ATM switch used: 2

The device are interconnected as given below

- CPE 1 is connected with switch 1 by link 1.
- CPE 2 is connected with switch 1 by link 2.
- CPE 3 is connected with switch 2 by link 4.
- CPE 4 is connected with switch 2 by link 5.
- Switch 1 and switch 2 are connected with link 3.

Set the properties for each device by following the tables,

Switch Properties	Switch 1	Switch 2
Scheduling technique	FIFO	FIFO
Buffer size	4096 KB	4096 KB

CPE Properties	CPE 1
Destination	CPE 3
Transmission Type	Point to Point
Traffic Type	Data
Application Data Size	
Distribution	Constant
Mean Application Data Size (Bytes)	10000
Inter arrival time	
Distribution	Constant
Mean Inter-arrival time (Micro-sec)	10000
Generation rate (Mbps)	8
Scheduling	FIFO
Peak cell rate (cells/sec)	99999
Cell delay Variation tolerance (Micro-sec)	99999
GCRA type	VSA

CPE Properties	CPE 2
Destination	CPE 4
Transmission Type	Point to Point
Traffic Type	Voice
Codec	Constant
Application Data Size (Bytes)	10000
Inter-arrival time (Micro-sec)	20000
Service Type	CBR
Generation rate (Mbps)	4
Scheduling	FIFO
Peak cell rate (cells/sec)	99999
Cell delay Variation tolerance (Micro-sec)	99999
GCRA type	VSA

Link Properties	Link 1	Link 2	Link 3	Link 4	Link 5
Bit Error Rate (BER)	No Error				
Physical medium	E2	E2	E0	T1	T1
Data Rate (MbPS)	8.448	8.448	0.064	1.54	1.54
Distance	1	1	1	1	1

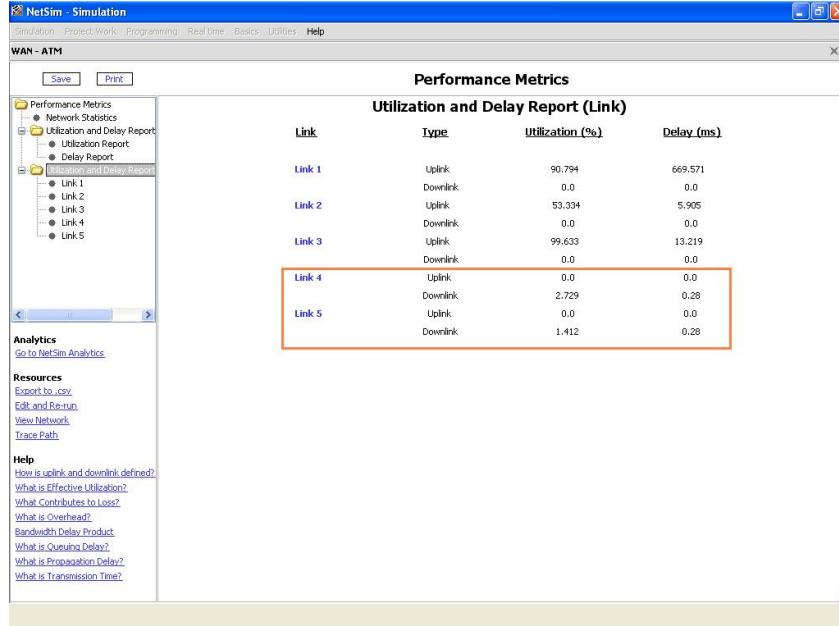
Simulation Time –10 Sec.

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

- o Set the properties of CPE and Switch,
 - o Then click on the Validate & Simulate button).
- “Save” it, upon completion of the experiment.

Output:

Select the metrics Utilization and Delay report (Link). Note down the link utilization of link 4 and link 5.



Priority:

In NetSim, Select Simulation→New→Legacy Networks →ATM

In the simulation environment window create the following scenario

How to Create Scenario & generate traffic:

- **Create Scenario:** “Help → NetSim Help → Running Simulation via GUI→ Legacy Networks → Create Scenario”.

Inputs:

Follow the steps given in the different samples to arrive at the objective.

In all Samples,

- Total no of CPE used: 4
- Total no of ATM switch used: 2

The device are interconnected as given below

- CPE 1 is connected with switch 1 by link 1.
- CPE 2 is connected with switch 1 by link 2.
- CPE 3 is connected with switch 2 by link 4.
- CPE 4 is connected with switch 2 by link 5.
- Switch 1 and switch 2 are connected with link 3.

Set the properties for each device by following the tables,

Switch Properties	Switch 1	Switch 2
Scheduling technique	Priority	FIFO
Buffer size	4096 KB	4096 KB

CPE Properties	CPE 1
Destination	CPE 3
Transmission Type	Point to Point
Traffic Type	Data
Application Data Size	
Distribution	Constant
Mean Application Data Size (Bytes)	10000
Inter arrival time	
Distribution	Constant
Mean Inter-arrival time (Micro-sec)	10000
Generation rate (Mbps)	8
Scheduling	FIFO
Peak cell rate (cells/sec)	99999
Cell delay Variation tolerance (Micro-sec)	99999
GCRA type	VSA

CPE Properties	CPE 2
Destination	CPE 4
Transmission Type	Point to Point
Traffic Type	Voice
Codec	Constant
Application Data Size (Bytes)	10000
Inter-arrival time (Micro-sec)	20000
Service Type	CBR
Generation rate (Mbps)	4
Scheduling	FIFO
Peak cell rate (cells/sec)	99999
Cell delay Variation tolerance (Micro-sec)	99999
GCRA type	VSA

Link Properties	Link 1	Link 2	Link 3	Link 4	Link 5
Bit Error Rate (BER)	No Error				
Physical medium	E2	E2	E0	T1	T1
Data Rate (MbPS)	8.448	8.448	0.064	1.54	1.54
Distance	1	1	1	1	1

Simulation Time –10 Sec.

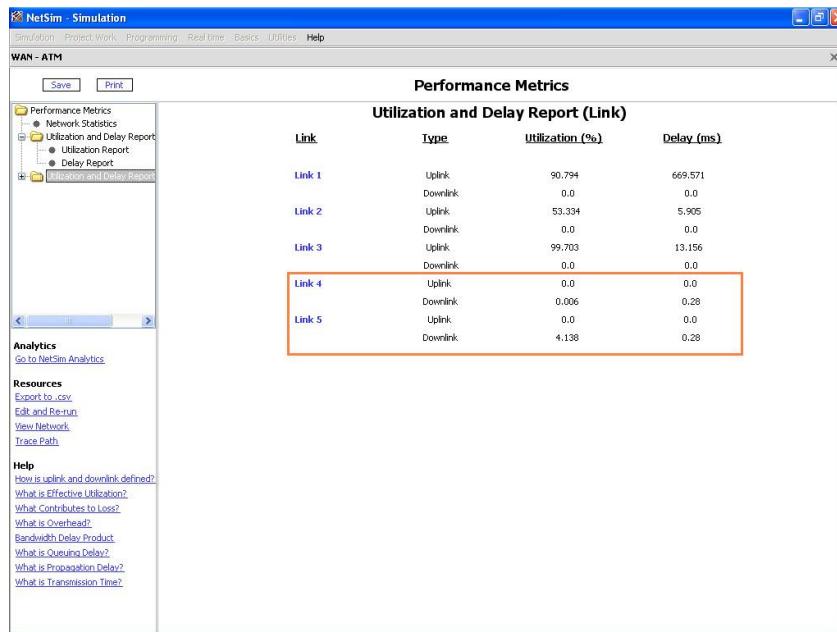
(Note: The Simulation Time can be selected only after doing the following two tasks,

- o Set the properties of CPE and Switch,
- o Then click on the Validate & Simulate button).

“Save” it, upon completion of the experiment.

Output:

Select the metrics Utilization and Delay report (Link). Note down the link utilization of link 4 and link 5.



Round Robin:

In NetSim, Select Simulation → New → Legacy Networks → ATM

In the simulation environment window create the following scenario

How to Create Scenario & generate traffic:

- **Create Scenario:** “Help → NetSim Help → Running Simulation via GUI → Legacy Networks → Create Scenario”.

Inputs:

Follow the steps given in the different samples to arrive at the objective.

In all Samples,

- Total no of CPE used: 4
- Total no of ATM switch used: 2

The device are interconnected as given below

- CPE 1 is connected with switch 1 by link 1.
- CPE 2 is connected with switch 1 by link 2.
- CPE 3 is connected with switch 2 by link 4.
- CPE 4 is connected with switch 2 by link 5.
- Switch 1 and switch 2 are connected with link 3.

Set the properties for each device by following the tables,

Switch Properties	Switch 1	Switch 2
Scheduling technique	Round Robin	FIFO
Buffer size	4096 KB	4096 KB

Link Properties	Link 1	Link 2	Link 3	Link 4	Link 5
Bit Error Rate (BER)	No Error				
Physical medium	E2	E2	E0	T1	T1
Data Rate (MbPS)	8.448	8.448	0.064	1.54	1.54
Distance	1	1	1	1	1

CPE Properties	CPE 1
Destination	CPE 3
Transmission Type	Point to Point
Traffic Type	Data
Application Data Size	
Distribution	Constant
Mean Application Data Size (Bytes)	10000
Inter arrival time	
Distribution	Constant
Mean Inter-arrival time (Micro-sec)	10000
Generation rate (Mbps)	8
Scheduling	FIFO
Peak cell rate (cells/sec)	99999
Cell delay Variation tolerance (Micro-sec)	99999
GCRA type	VSA

CPE Properties	CPE 2
Destination	CPE 4
Transmission Type	Point to Point
Traffic Type	Voice
Codec	Constant
Application Data Size (Bytes)	10000
Inter-arrival time (Micro-sec)	20000
Service Type	CBR
Generation rate (Mbps)	4
Scheduling	FIFO
Peak cell rate (cells/sec)	99999
Cell delay Variation tolerance (Micro-sec)	99999
GCRA type	VSA

Simulation Time –10 Sec.

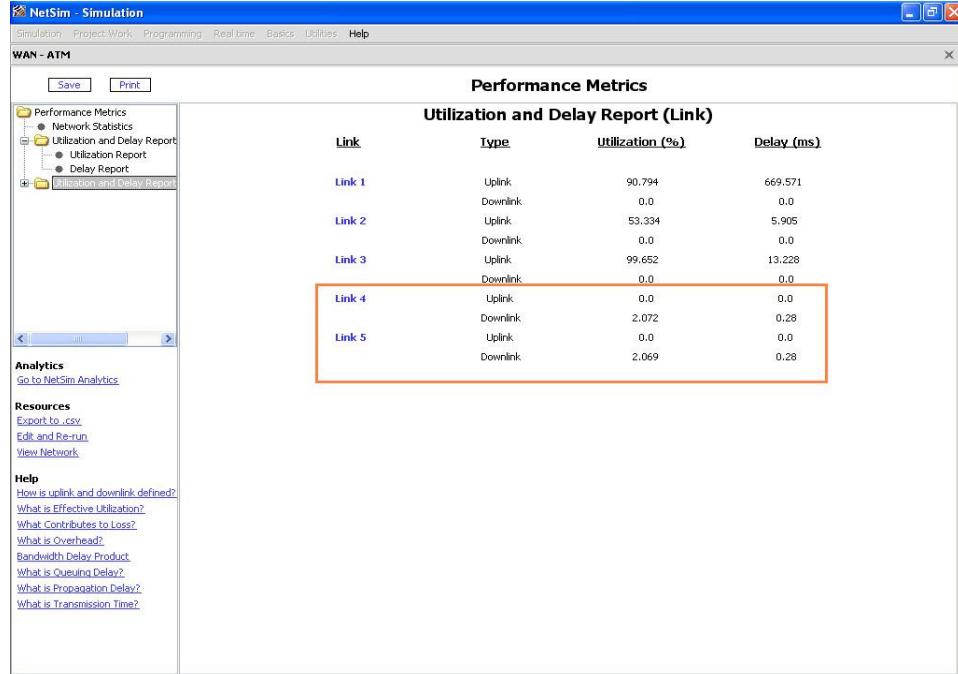
(Note: The Simulation Time can be selected only after doing the following two tasks,

- o Set the properties of CPE and Switch,
- o Then click on the Validate & Simulate button).

“Save” it, upon completion of the experiment.

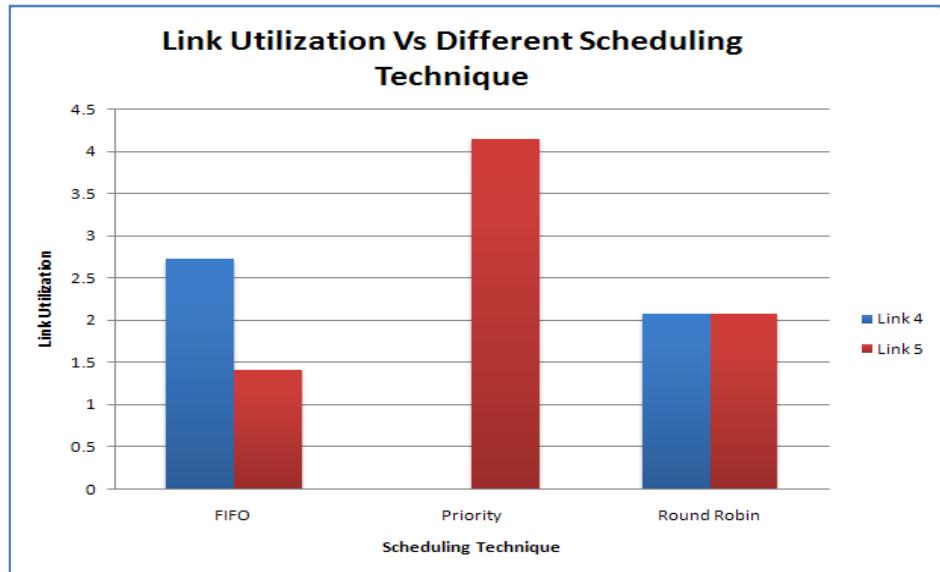
Output:

Select the metrics Utilization and Delay report (Link). Note down the link utilization of link 4 and link 5.

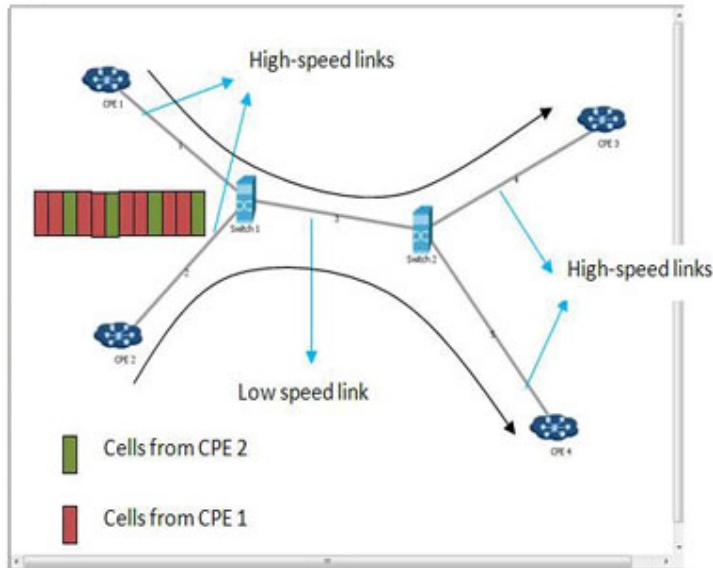


Comparison:

Link Utilization Vs Scheduling Technique			
	FIFO	Priority	Round Robin
Link1 (Utilization)	90.794	90.794	90.794
Link2 (Utilization)	53.334	53.334	53.334
Link3 (Utilization)	99.633	99.703	99.652
Link4 (Utilization)	2.729	0.006	2.072
Link5 (Utilization)	1.412	4.138	2.069



Traffic analysis:



CPE 1 is transmitting Data traffic (being generated at 8 Mbps) to CPE 3 through links 1, 3, and 4. CPE2 transmitting voice traffic (being generated at 4 Mbps) to CPE4 through link 2, 3, and 5. Here, voice traffic has priority over data traffic.

Link speed of link 1 and 2 is high (8.448 mbps) as compared to data rate and also the PCR and CDVT of CPE1 and CPE 2 are high. So there is a low probability that these cells will be dropped. Therefore, all the cells reach switch1 where scheduling will happen.

Link speed of link 3 is very low (0.064 mbps) which means it does not have enough resources to handle the cells. Hence, it will only pass those cells that have high priority (based on Scheduling technique). Link speed of 4 and 5 is high compared to link 3 and so there is no queue buildup on switch 2. Based on what type of cells pass through link 3, determines the utilization of the link 4 and link 5.

Inference:

As we see in chart 1 and table 1, the utilization of link 4 is double of link 5 in case of FIFO. Because, in case of FIFO, scheduler gives preference to which come cells first. Note that the data rate of CPE1 is double than CPE2 data rate and link speed is same hence switch 1 gets two cells from CPE 1 and one cell from CPE2. Hence, switch 1 schedules two packets of CPE1 and one packet of CPE2. Therefore, the number of packet transmitted through link 4 is double than link 5 and hence utilization is also double.

In case of priority, the utilization of the link 4 is 0.006 % and link 5 is 4.138%. This is because the ATM scheduler gives priority to the voice traffic (generated by CPE2 to CPE 4 via link 2, 3, and 5) over data traffic (generated by CPE 1 to CPE 3 via link 1, 3 and 4). The generation rate of voice traffic is 4 mbps which is much greater than the link speed of link 3 (0.064mbps). Hence, scheduler only schedules voice traffic and data traffic keeps waiting in the queue.

In case of round robin the utilization of both the links is the same. Because, in case of Round robin, scheduler schedules the packet in circular fashion means one packet from CPE1 and one packet from CPE2. Note that the data rate of CPE1 and CPE2 is sufficiently high to ensure presence of a packet in the buffer from both the CPEs. Therefore, the number of packets transmitted through link four is same as link five and hence utilization is also same.

NetSim - MANET

Sample Experiments - User can understand the internal working of Dynamic Source Routing through these sample experiments. Each sample experiment covers:

- Procedure
- Sample Inputs
- Output
- Comparison chart
- Inference

Index	Objective
Experiment 1	To create scenario and study the performance of MANET mobility model using NetSim simulation.

Sample Experiment 1

Objective:

To create scenario and study the performance of MANET mobility model using NetSim simulation.

Introduction:

In Random waypoint model, nodes will move freely, and remain in the same position for some time, which is called “Pause Time”, and then it will choose another random position and it will move towards that position with a given velocity. For further details refer Basics → Advanced Wireless Networks → MANET→Mobility.

Procedure:

How to Create Scenario & Generate Traffic:

Please refer,

- **Create Scenario:** “Help → NetSim Help → Running Simulation via GUI→ Advanced Wireless Networks → Create Scenario”.

Sample Inputs:

In this Sample experiment, required number of Nodes needs to be dragged & dropped onto the Simulation Environment. Upon completion of the experiment “Save” them for comparisons that can be carried out in the “Analytics” section.

First create a scenario with 10 nodes. Then follow these steps

- Experiment 1: Node 1 transmits data to Node 6, Node 2 -> Node 7, and Node 3 -> Node 8 And so on. In a circular fashion till all 10 nodes transmit. Set pause time as 1.
- Experiment 2: Repeat the experiment 1 with pause time 2 in environment property.
- Experiment 3: Repeat the experiment 1 with pause time 3 in environment property.
- And so on. Increase the pause time up to 10 in environment property.

Inputs for the Sample experiment, where 5 nodes are transmitting is given below:

Node Properties	Node - 1	Node - 2	Node - 3	Node - 4	Node - 5
Transmission	Point-to-Point	Point-to-Point	Point-to-Point	Point-to-Point	Point-to-Point
Destination	Node6	Node7	Node8	Node9	Node10
Traffic Type	Data	Data	Data	Data	Data
Application Data Size					
Distribution	Constant	Constant	Constant	Constant	Constant
Application Data Size (Bytes)	1472	1472	1472	1472	1472
Inter Arrival Time					
Distribution	Constant	Constant	Constant	Constant	Constant
Mean Inter Arrival Time(Micro sec)	20000	20000	20000	20000	20000
RTS Threshold(Bytes)	0	0	0	0	0
Retry Limit	7	7	7	7	7
Transmitter power (Milli Watts)	100	100	100	100	100

Environment Properties:

Environment Properties	Values
Mobility Model	Random Way Point model
Velocity (m/sec)	50
Pause Time	1
Transmission	DSSS
Channel Number	1
Frequency (MHz)	2412
Channel Characteristics	No Path Loss

Simulation Time - 10 Sec.

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

- Set the properties of Node & Environment.
- Click on the Validate & Simulate button and save the experiment.

Increase the pause time and continue the above procedure up to 10 pause time.

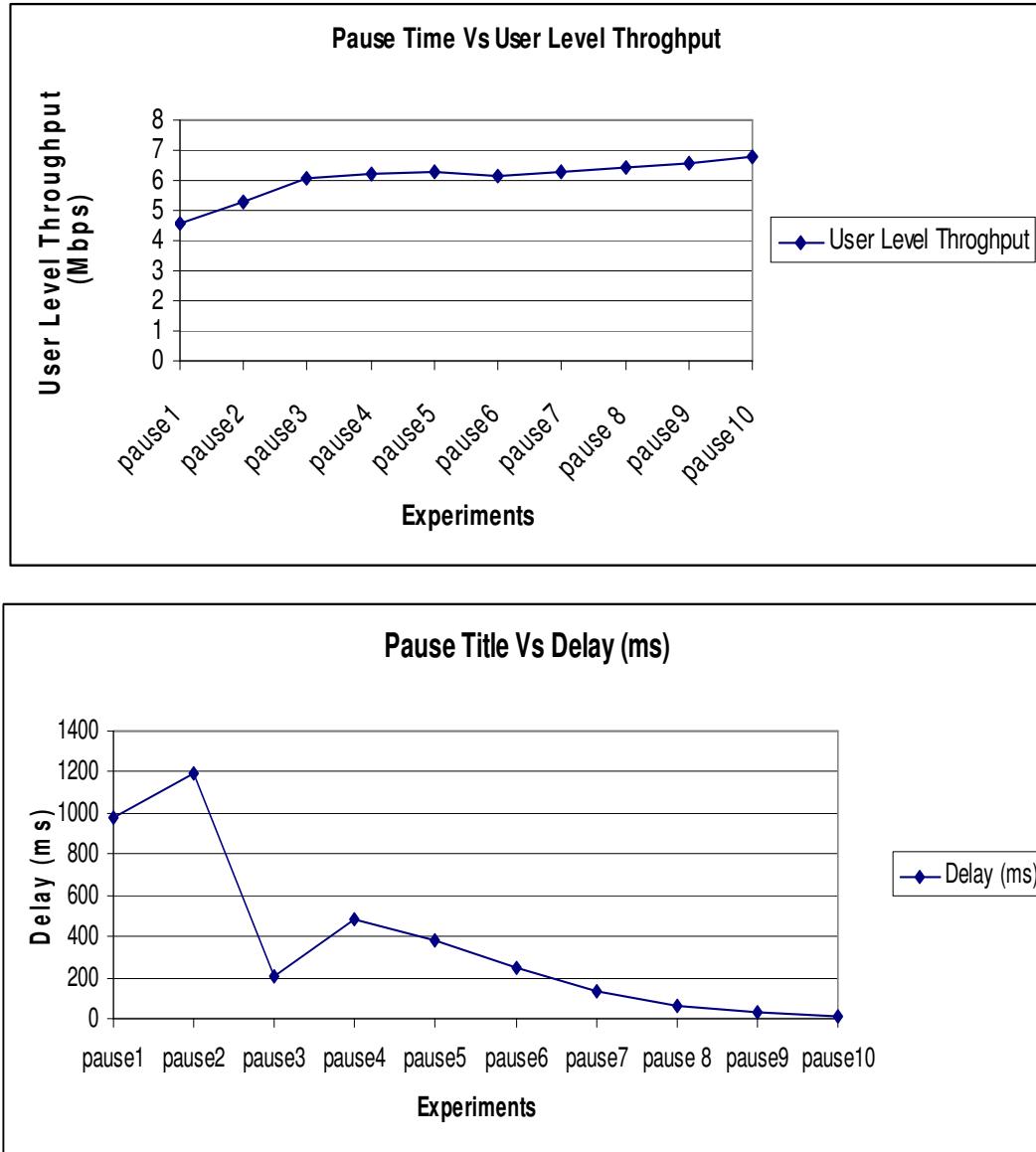
Output:

1. For User level Throughput- Get the User level throughput value from Network Statistics - Performance metrics window for all experiments and draw the graph.
2. For Delay- Get the Delay value from Network - Performance metrics window for all experiments and draw the graph.

Note:

User level Throughput and Delay values can be exported to Excel sheet using “Export to .CSV” Option. Then the required graph can be drawn in the Excel sheet

Comparison Chart:



*** All the above plots highly depend upon the placement of Node 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.

Inference:

As the pause time increases, user level throughput will increase because of lower mobility. There is a lower probability of route errors. This leads to increased number of packets transmitted. It also results in reduced packet waiting time in the queue.

NetSim – Wi-MAX

Sample Experiments - User can understand the internal working of Wi MAX through these sample experiments. Each sample experiment covers:

- Procedure
- Sample Inputs
- Output
- Comparison chart
- Inference

Index	Objective
Experiment 1	To study the Call Blocking probability of a Wi-MAX (IEEE 802.16 – 2004) network varies for UGS (Unsolicited Grant Service) QoS Class as the number of transmitting SSs increase beyond the bandwidth limit.

Sample Experiment 1

Objective:

To study the Call Blocking probability of a Wi-MAX (IEEE 802.16 – 2004) network varies for UGS (Unsolicited Grant Service) QoS Class as the number of transmitting SSs increase beyond the bandwidth limit.

Theory:

Unsolicited Grant Service (UGS) is one of the QoS service types defined in the IEEE 802.16 WiMAX. UGS supports real-time traffic (Voice over IP) and offers fixed size unsolicited data grants (transmission opportunities) on a periodic basis.

Call Blocking:

Call blocking is blocking of user calls due to bandwidth unavailability.

Procedure:

How to Create Scenario:

- **Create Scenario:** “Help → NetSim Help → Running Simulation via GUI→ Advanced Wireless Networks → Create Scenario”.

Sample Inputs:

Follow the steps given in the different samples to arrive at the objective.

Sample 1:

In this Sample,

- Total no of SS's used: 2
- Total no of BS used: 1

Set the properties for each device by following the tables,

SS Properties	SS 1
Destination	2
Traffic Type	Voice
Call Details	
Distribution	Constant
Mean Call Interval Time (Secs)	300
Call Duration (Secs)	60
Codec	G711
Packet size (Bytes)	160
Inter Arrival Time (Micro Secs)	20000
Service Type	CBR

BS Properties	BS1
Modulation Technique	BPSK
Nominal Channel Bandwidth (MHz)	3.5
OFDM Frame Duration (ms)	10

Simulation Time –1000 Sec.

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

- Set the properties of SS & BS,
- Then click on the Validate & Simulate button).

“Save” it, upon completion of the experiment. Also note the Call Drop probability from Network Statistics.

Sample 2:

In this Sample,

- Total no of SS's used: 3
- Total no of BS used: 1

Set the properties for each device by following the tables,

SS Properties	SS 1	SS 2
Destination	2	3
Traffic Type	Voice	Voice
Call Details		
Distribution	Constant	Constant
Call Interval Time (Secs)	300	300
Call Duration (Secs)	60	60
Codec	G711	G711
Packet size (Bytes)	160	160
Inter Arrival Time (Micro Secs)	20000	20000
Service Type	CBR	CBR

BS Properties	BS1
Modulation Technique	BPSK
Nominal Channel Bandwidth (MHz)	3.5
OFDM Frame Duration (ms)	10

Simulation Time –1000 Sec.

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

- Set the properties of SS & BS,
- Then click on the Validate & Simulate button).

“Save” it, upon completion of the experiment. Also note the Call Drop probability from Network Statistics.

Sample 3:

In this Sample,

- Total no of SS's used: 4
- Total no of BS used: 1

Set the properties for each device by following the tables,

SS Properties	SS 1	SS 2	SS 3
Destination	2	3	4
Traffic Type	Voice	Voice	Voice
Call Details			
Distribution	Constant	Constant	Constant
Call Interval Time (Secs)	300	300	300
Call Duration (Secs)	60	60	60
Codec	G711	G711	G711
Packet size (Bytes)	160	160	160
Inter Arrival Time (Micro Secs)	20000	20000	20000
Service Type	CBR	CBR	CBR

BS Properties	BS1
Modulation Technique	BPSK
Nominal Channel Bandwidth (MHz)	3.5
OFDM Frame Duration (ms)	10

Simulation Time –1000 Sec.

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

- Set the properties of SS & BS,
- Then click on the Validate & Simulate button).

“Save” it, upon completion of the experiment. Also note the Call Drop probability from Network Statistics.

Sample 4:

In this Sample,

- Total no of SS's used: 5
- Total no of BS used: 1

Set the properties for each device by following the tables,

SS Properties	SS 1	SS 2	SS 3	SS 4
Destination	2	3	4	5
Traffic Type	Voice	Voice	Voice	Voice
Call Details				
Distribution	Constant	Constant	Constant	Constant
Call Interval Time (Secs)	300	300	300	300
Call Duration (Secs)	60	60	60	60
Codec	G711	G711	G711	G711
Packet size (Bytes)	160	160	160	160
Inter Arrival Time (Micro Secs)	20000	20000	20000	2000
Service Type	CBR	CBR	CBR	CBR

BS Properties	BS1
Modulation Technique	BPSK
Nominal Channel Bandwidth (MHz)	3.5
OFDM Frame Duration (ms)	10

Simulation Time –1000 Sec.

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

- Set the properties of SS & BS,
- Then click on the Validate & Simulate button).

“Save” it, upon completion of the experiment. Also note the Call Drop probability from Network Statistics.

Sample 5:

In this Sample,

- Total no of SS's used: 6
- Total no of BS used: 1

Set the properties for each device by following the tables,

SS Properties	SS 1	SS 2	SS 3	SS 4	SS 5
Destination	2	3	4	5	6
Traffic Type	Voice	Voice	Voice	Voice	Voice
Call Details					
Distribution	Constant	Constant	Constant	Constant	Constant
Call Interval Time (Secs)	300	300	300	300	300
Call Duration (Secs)	60	60	60	60	60
Codec	G711	G711	G711	G711	G711
Packet size (Bytes)	160	160	160	160	160
Inter Arrival Time (Micro Secs)	20000	20000	20000	20000	20000
Service Type	CBR	CBR	CBR	CBR	CBR

BS Properties	BS1
Modulation Technique	BPSK
Nominal Channel Bandwidth (MHz)	3.5
OFDM Frame Duration (ms)	10

Simulation Time –1000 Sec.

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

- Set the properties of SS & BS,
- Then click on the Validate & Simulate button).

“Save” it, upon completion of the experiment. Also note the Call Drop probability from Network Statistics.

Repeat the above sample for Sample 6, 7 ... and 25 by increasing one SS and set the properties for added SS.

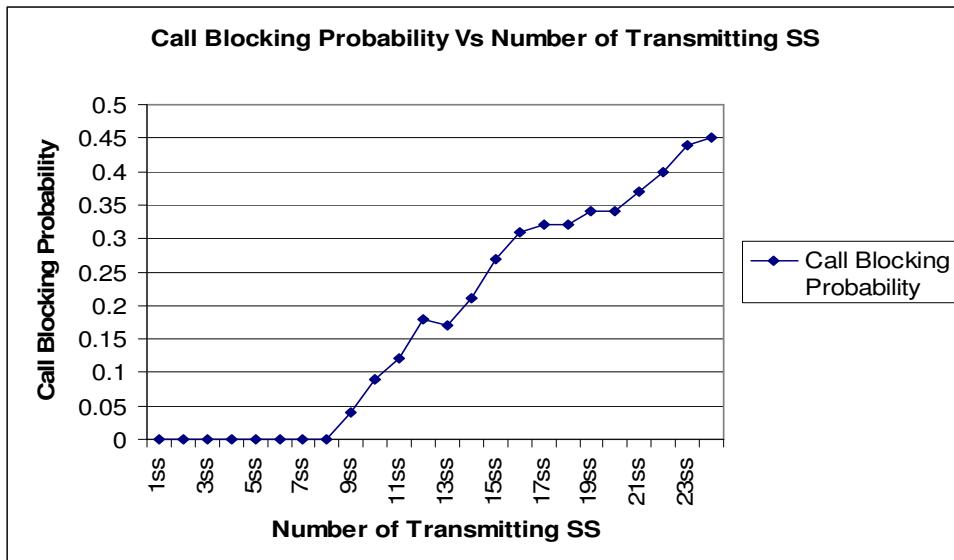
Note:

When you are setting the properties for BS, make sure you have selected the bandwidth as 3.5 MHz and OFDM Frame duration as 10 ms.

Output:

To view the output by using Microsoft Excel the Call Drop Probability values for Sample 1, 2, 3 and 25 need to be added onto the sheet.

Comparison Chart:



Inference:

As the number of transmitting SS increase beyond the Bandwidth (i.e 3.5MHz) limit the call blocking probability increased. As more SS generate traffic, there is greater “load” on the network beyond the Bandwidth limit (3.5 MHz), this leads to increase in Call blocking probability.

Based on Physical layer inputs,

Only 62 OFDM symbols are used to handle all the calls. To handle one call, it requires 17 OFDM symbols. So it can handle simultaneously 3 Calls request. If more than three calls arrive, then the extra arrived calls are blocked.

1. From the graph we have seen the variation in call blocking probability as the numbers of SS are increased from 1 to 25. The Calls generated by all SS are not effectively handled by BS, because of Bandwidth unavailability (i.e. 3.5 MHz)
2. In the graph we are getting a call drop variation in 11SS to 25 SS, because each SS starts the first call exponentially.

NetSim - BGP

Sample Experiments - User can understand the working of BGP and formation of BGP Routing Tables through this sample experiment. The sample experiment covers:

- Procedure
- Sample Inputs
- Output
- Inference

Index	Objective
Experiment 1	Study the working of BGP and formation of BGP Routing table.

Sample Experiment 1

Objective:

Study the working of BGP and formation of BGP Routing table.

Theory:

In BGP, the Packets are transmitted between the Autonomous system using Path vector Routing.

Path Vector Routing:

Path vector routing is used for inter-domain routing. It is similar to distance vector routing. In path vector routing we assume that there is one Router (there can be many) in each autonomous system which acts on behalf of the entire autonomous system. This Router is called the Border Router. The Border Router in one Autonomous System creates a routing table and advertises it to neighboring Border Router which belongs to neighboring autonomous systems. The idea is same as distance vector routing except that only Border Routers in each autonomous system can communicate with each other. The Border Routers advertises the path, not the metric, in its autonomous system or other autonomous systems.

Procedure:**How to Create Scenario:**

- **Create Scenario:** “Help → NetSim Help → Running Simulation via GUI → BGP Networks → Create Scenario”.

Sample Inputs:

Follow the steps given in the different samples to arrive at the objective.

- Total no of CPE s used: 4
- Total no of Internal Routers used: 2
- Total number of Border Routers used: 2

The devices are inter connected as given below,

- CPE 1 and CPE 2 are connected with Internal Router 3 by Link 1 and Link 2.
- Internal Router 3 and Border Router 1 are connected by Link 3.
- Border Router 1 and Border Router 2 are connected by Link 4.
- Border Router 2 and Internal Router 4 are connected by Link 5.
- Internal Router 4 is connected with CPE 3 and CPE 4 by Link 6 and Link 7 respectively.

Set the properties for each device by following the tables,

CPE Properties	CPE 1	CPE 2
Destination	CPE 3	CPE 4
Priority	Low	Low
Traffic Type	Data	Data
Application Data size		
Distribution	Constant	Constant
Application Data size (Bytes)	1472	1472
Inter Arrival Time		
Distribution	Constant	Constant
Mean Inter Arrival Time (μs)	20000	20000

If you want to select your internal gateway protocol as RIP then here is the information you need to fill in for Router properties

Router Properties	BGP Router1		BGP Router2		Internal Router3		
	Port 1	Port 2	Port 1	Port 2	Port 1	Port 2	Port 3
Buffer Size (KB)	8	8	8	8	8	8	8
Scheduling Type	Priority	Priority	Priority	Priority	Priority	Priority	Priority
Protocol Type	RIP	BGP	BGP	RIP	RIP	RIP	RIP
Periodic Time	30	-	-	30	30	30	30
Expiration Time	180	-	-	180	180	180	180
Garbage Collection Time	120	-	-	120	120	120	120
Local Preference	-	100	100	-	-	-	-
MED	-	0	0	-	-	-	-

Router Properties		Internal Router4		
		Port 1	Port 2	Port 3
Buffer Size (KB)	8	8	8	8
Scheduling Type	Priority	Priority	Priority	Priority
Protocol Type	RIP	RIP	RIP	RIP
Periodic Time	30	30	30	30
Expiration Time	180	180	180	180
Garbage Collection Time	120	120	120	120
Local Preference	-	-	-	-
MED	-	-	-	-

If you want to select your internal gateway protocol as OSPF then here is the information you need to fill in for Router properties

Router Properties	BGP Router1		BGP Router2		Internal Router3		
	Port 1	Port 2	Port 1	Port 2	Port 1	Port 2	Port 3
Buffer Size (KB)	8	8	8	8	8	8	8
Scheduling Type	Priority	Priority	Priority	Priority	Priority	Priority	Priority
Protocol Type	OSPF	BGP	BGP	OSPF	OSPF	OSPF	OSPF
Priority	255	-	-	255	255	255	255
Local Preference	-	100	100	-	-	-	-
MED	-	0	0	-	-	-	-

Router Properties	Internal Router4		
	Port 1	Port 2	Port 3
Buffer Size (KB)	8	8	8
Scheduling Type	Priority	Priority	Priority
Protocol Type	OSPF	OSPF	OSPF
Priority	256	256	256
Local Preference	-	-	-
MED		-	-

Link Properties	Link 1	Link 2	Link 3	Link 4	Link 5	Link 6	Link 7
Distance (km)	1	1	1	1	1	1	1
Bit Error Rate (BER)	No Error	No Error	No Error	No Error	No Error	No Error	No Error
Physical Medium	E2	E2	CAT5(10 Mbps)	1000 Mbps	CAT5(10 Mbps)	E2	E2
Link Weight	1	1	1	1	1	1	1

Simulation Time - 10 Sec.

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

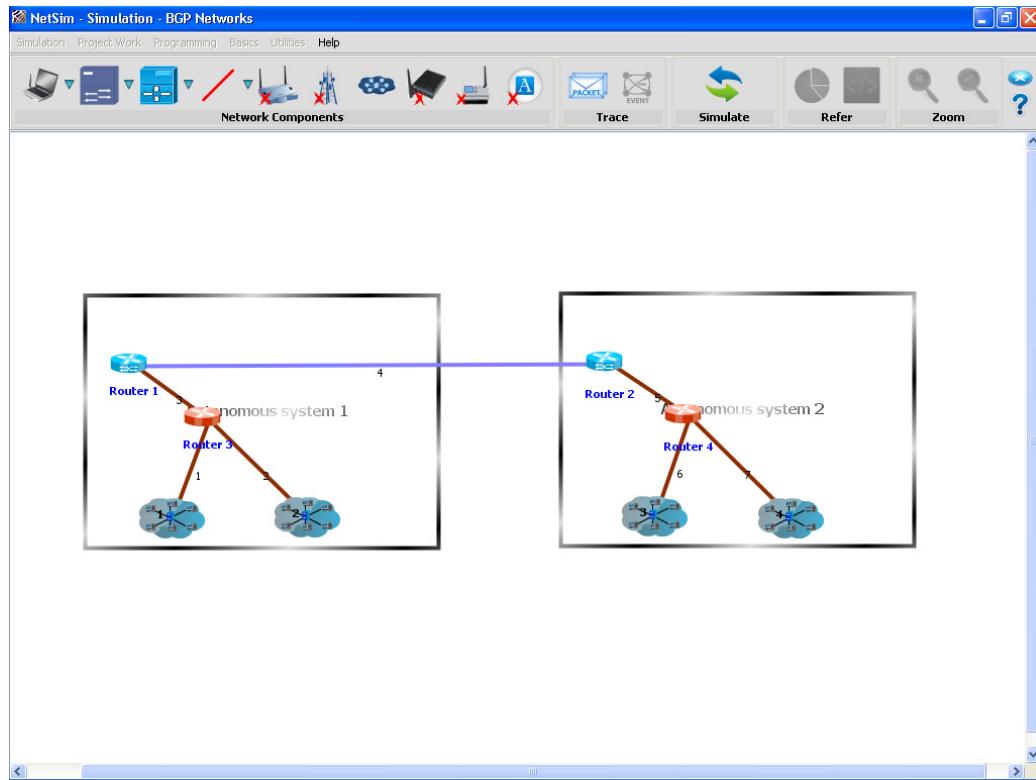
- Set the properties of CPE & Routers

- Then click on the Validate & Simulate button).

Upon completion of the experiment “Save” it for comparison that can be carried out in the “Analytics” section.

Output:

After running this scenario, in Network performance screen, click View Routing table link. Then the following screen would appear



If you click over the internal routers, you will get the RIP/OSPF routing table for internal routers. If you click over the Border router 1 or 2, you will get the routing table for Border routers. We have shown the routing tables for Border Router 1 and 2.

Border Router - 1 (Table Information)			
Border Router Table			
Destination	Next Hop	Interface	AS PATH
10.1.3.0	10.1.3.2	1	AS1
10.1.5.0	10.1.4.2	2	AS2->AS1
10.1.1.0	10.1.3.1	1	AS1
10.1.2.0	10.1.3.1	1	AS1
10.1.6.0	10.1.4.2	2	AS2->AS1
10.1.7.0	10.1.4.2	2	AS2->AS1

Border Router - 2 (Table Information)			
Border Router Table			
Destination	Next Hop	Interface	AS PATH
10.1.5.0	10.1.5.1	2	AS2
10.1.3.0	10.1.4.1	1	AS1->AS2
10.1.6.0	10.1.5.2	2	AS2
10.1.7.0	10.1.5.2	2	AS2
10.1.1.0	10.1.4.1	1	AS1->AS2
10.1.2.0	10.1.4.1	1	AS1->AS2

The Border Routers store the CPE's Network address in its Routing Table as shown in the above Tables under "Destination" column.

Inference:

First the internal Routing tables (RIP/OSPF table) are formed among all the Routers belonging to each of the Autonomous systems. The Border Routers containing the network address of all the CPEs in it's AS represents the routing table of it's AS. Border Routers in each AS, having ability to communicate with another AS, pass their Routing tables resulting in the formation of external Routing tables (BGP table). Then actual packet transmission takes place from Source to Destination between Autonomous systems.

NetSim - MPLS

Sample Experiments - User can understand the internal working of MPLS through these sample experiments. Each sample experiment covers:

- Procedure
- Sample Inputs
- Output
- Comparison chart
- Inference

Index	Objective
Experiment 1	Study how the LSP varies for different traffic in MPLS -TE (Traffic Engineering)

Sample Experiment

Objective:

Study how the LSP varies for different traffic in MPLS -TE (Traffic Engineering)

Theory:

Traffic Engineering is the process of controlling how traffic flows through one's network so as to optimize resource utilization and network performance. Traffic Engineering is needed in the Internet mainly because current IGPs (Interior Gateway Protocols – RIP, OSPF) always use the shortest paths to forward all traffics. But MPLS with traffic engineering use different paths to forward different traffics.

Procedure:

How to Create Scenario:

- **Create Scenario:** “Help → NetSim Help → Running Simulation via GUI → MPLS Networks → Create Scenario”.

Sample Inputs:

Follow the steps given in the different samples to arrive at the objective.

Sample 1:

In this Sample,

- Total no of CPE's used: 4
- Total no of Routers used: 4

The devices are inter connected as given below,

- CPE 1, 2 and Router 2, 3 are connected with Router 1 by Link 1, 2,3,4 respectively.
- Router 2,3 and CPE 3,4 are connected with Router 4 by Link 5,6 7, 8 respectively

Set the properties for each device by following the tables,

CPE Properties	CPE 1	CPE2
Destination	CPE 3	CPE 4
Priority	Low	Low
Traffic Type	Data	Data
Application Data Size (Bytes)	1472	1472
Distribution	Constant	Constant
Inter Arrival Time (μ s)	1963	1963
Reserved Bandwidth (Mbps)	6	6

Link Properties	Link 1	Link 2	Link 3	Link 4
Distance (km)	1	1	1	1
Bit Error Rate (BER)	No Error	No Error	No Error	No Error
Physical Medium	E2	E2	CAT5 (10 Mbps)	CAT5 (10 Mbps)

Link Properties	Link 5	Link 6	Link 7	Link 8
Distance (km)	1	1	1	1
Bit Error Rate (BER)	No Error	No Error	No Error	No Error
Physical Medium	CAT5 (10 Mbps)	CAT5 (10 Mbps)	E2	E2

Router Properties	Router1	Router2	Router3	Router4
Buffer Size (KB)	8	8	8	8
Scheduling Type	Priority	Priority	Priority	Priority
Routing Properties				
Protocol Type	OSPF	OSPF	OSPF	OSPF
Router Priority	255	255	255	255
MPLS Properties				
Traffic Engineering	Enable	Enable	Enable	Enable

Simulation Time –10 Sec.

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

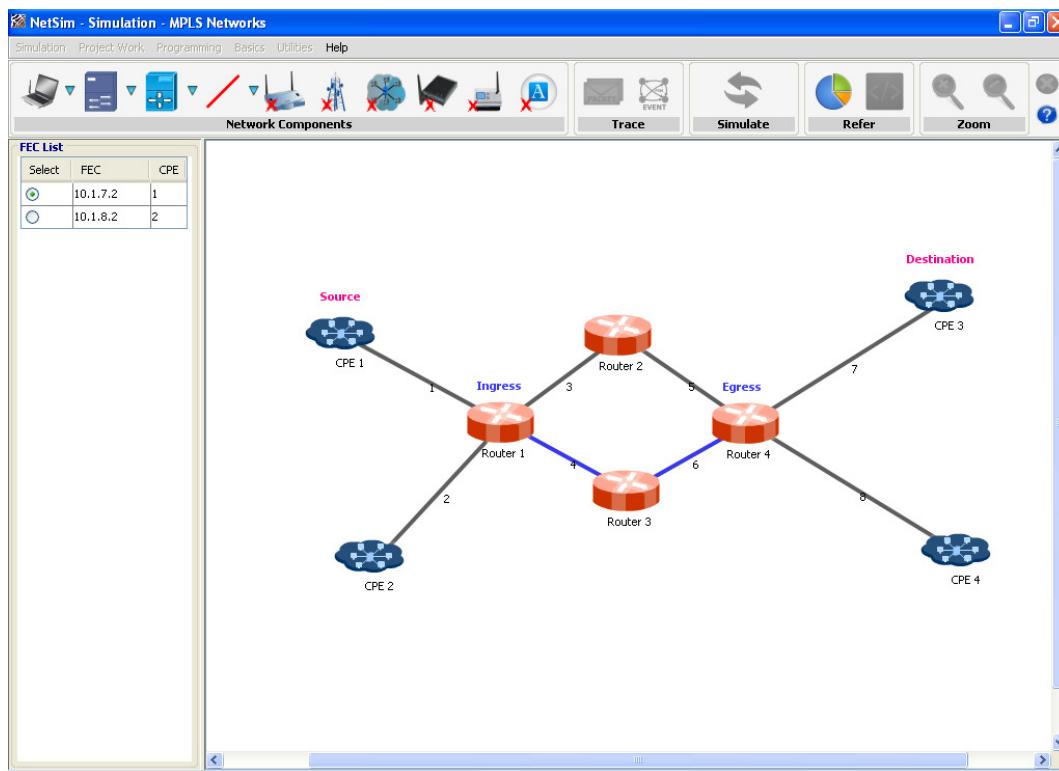
- o Set the properties of CPE & Router, &
- o Then click on the Validate & Simulate button).

Upon completion of the experiment “Save” it for comparison

Output:

After running this scenario, in Network performance screen, click View LSP link. Then the following screen would appear.

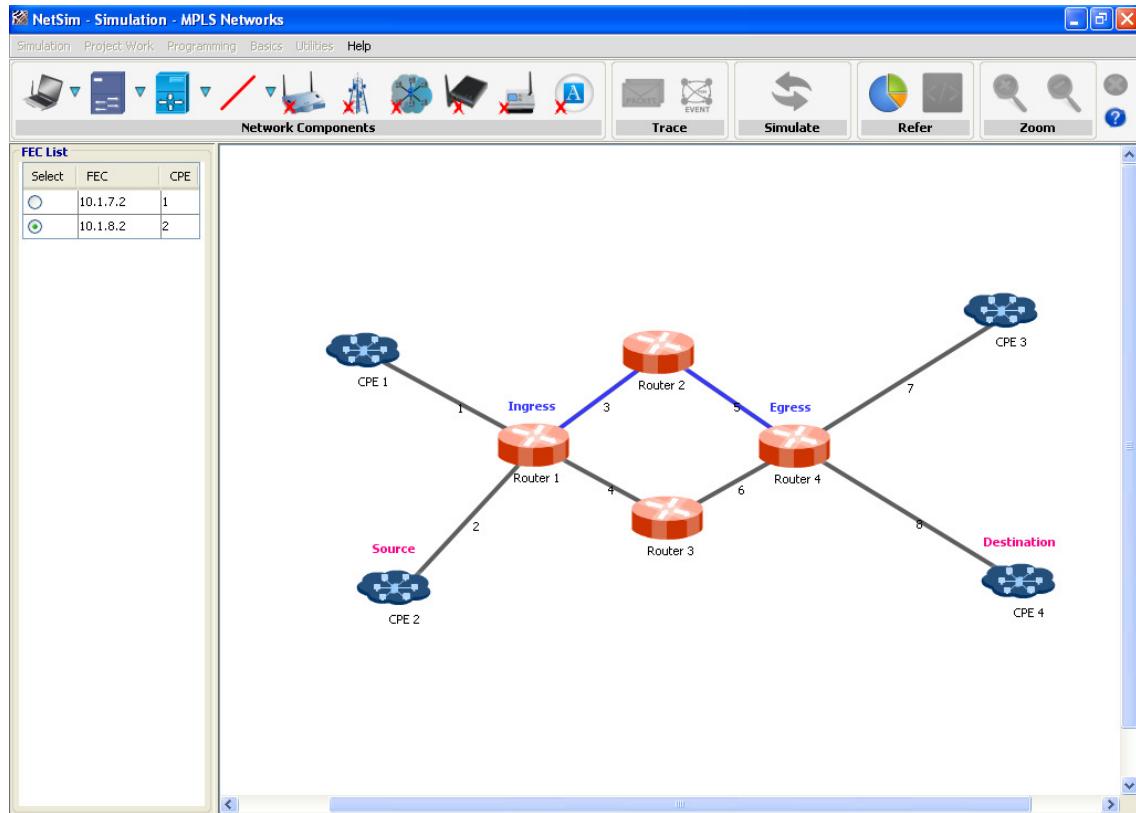
Click 10.1.7.2(CPE-1) to view the LSP for CPE 1's traffic.



Reserved Bandwidth of CPE1's traffic is 6 Mbps. CPE 1's traffic is sent through 4 and 6 links.

So Remaining Bandwidth of 4 and 6 links are 4 Mbps (10-6).

Click 10.1.8.2(CPE-2) to view the LSP for CPE 2's traffic.



Reserved Bandwidth of CPE 2's traffic is also 6 Mbps. Since Link 4 and 6's remaining bandwidth is less than the Reserved Bandwidth of CPE 2's traffic, CPE 2's traffic is sent through the another path through 3 and 5 links.

Sample 2:

In this Sample,

- Total no of CPE's used: 4
- Total no of Routers used: 4

The devices are inter connected as given below,

- CPE 1, 2 and Router 2, 3 are connected with Router 1 by Link 1, 2, 3, 4 respectively.
- Router 2,3 and CPE 3,4 are connected with Router 4 by Link 5,6 7, 8 respectively

Set the properties for each device by following the tables,

CPE Properties	CPE 1	CPE2
Destination	CPE 3	CPE 4
Priority	Low	Low
Traffic Type	Data	Data
Application Data Size (Bytes)	1472	1472
Distribution	Constant	Constant
Inter Arrival Time (μs)	1963	1963
Reserved Bandwidth (Mbps)	6	6

Link Properties	Link 1	Link 2	Link 3	Link 4
Distance (km)	1	1	1	1
Bit Error Rate (BER)	No Error	No Error	No Error	No Error
Physical Medium	E2	E2	CAT5 (10 Mbps)	CAT5 (10 Mbps)

Link Properties	Link 5	Link 6	Link 7	Link 8
Distance (km)	1	1	1	1
Bit Error Rate (BER)	No Error	No Error	No Error	No Error
Physical Medium	CAT5 (10 Mbps)	CAT5 (10 Mbps)	E2	E2

Router Properties	Router1	Router2	Router3	Router4
Buffer Size (KB)	8	8	8	8
Scheduling Type	Priority	Priority	Priority	Priority
Routing Properties				
Protocol Type	OSPF	OSPF	OSPF	OSPF
Router Priority	255	255	255	255
MPLS Properties				
Traffic Engineering	Disable	Disable	Disable	Disable

Simulation Time –10 Sec.

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

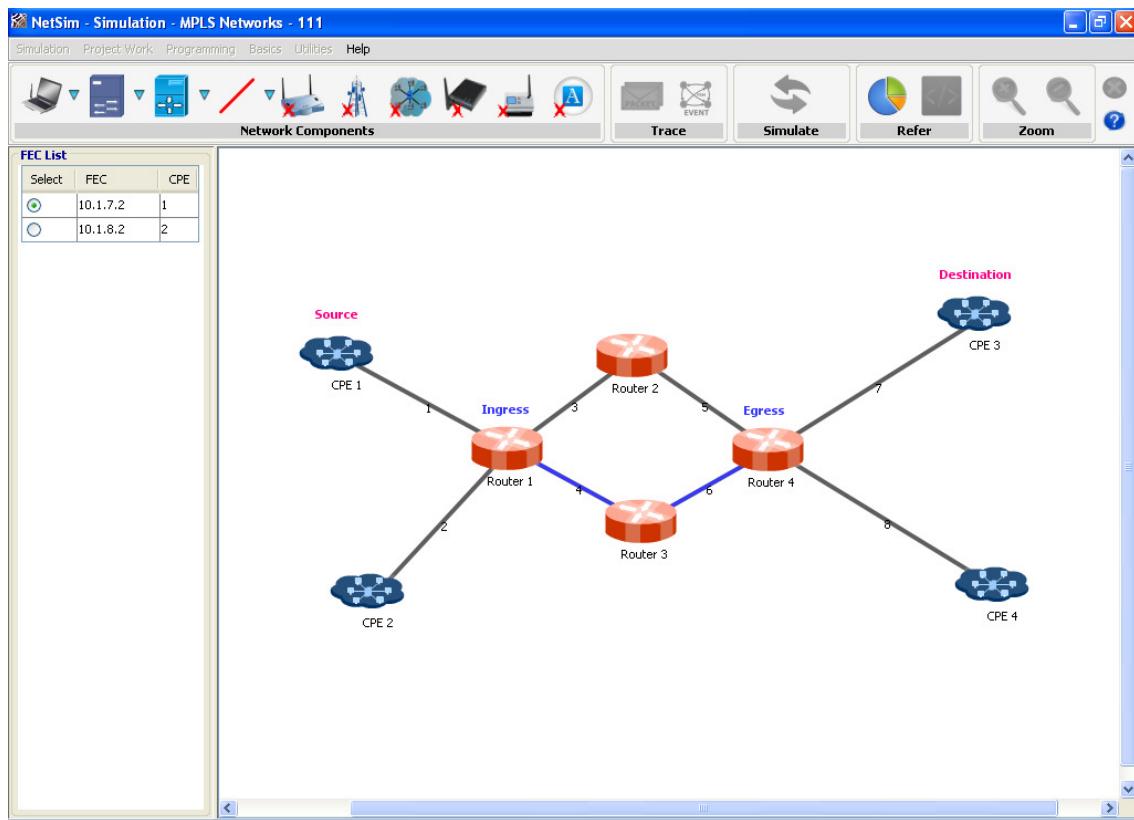
- o Set the properties of CPE & Router, &
- o Then click on the Validate & Simulate button).

Upon completion of the experiment “Save” it for comparison

Output:

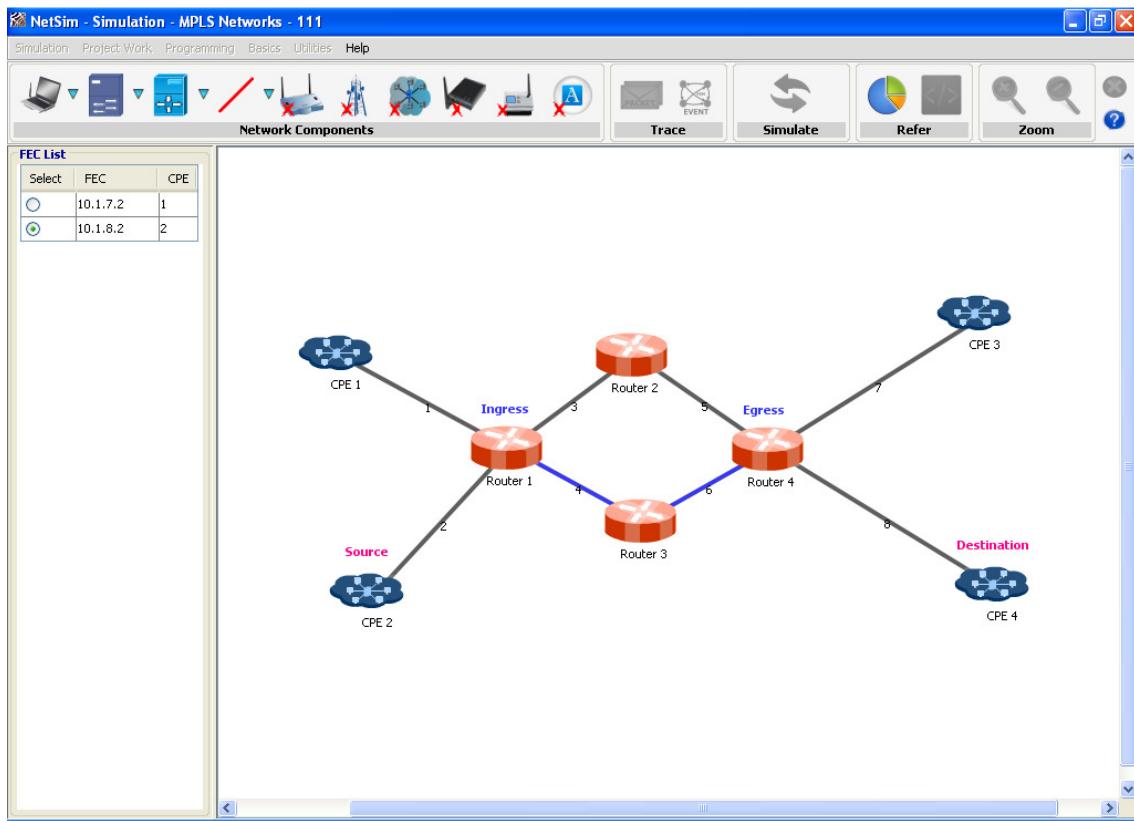
After running this scenario, in Network performance screen, click View LSP link. Then the following screen would appear.

Click 10.1.7.2(CPE-1) to view the LSP for CPE 1's traffic.



CPE 1's traffic is sent through 4 and 6 links

Click 10.1.8.2(CPE-2) to view the LSP for CPE 2's traffic



CPE 2's traffic is also sent through 4 and 6 links

In Without traffic engineering all the traffics will be sent through the same shortest path.

Inference:

In Traffic Engineered MPLS network, according to the traffic constraint different traffics will be sent through different paths. But in without traffic engineering, all traffics will be sent through same shortest path.

NetSim – GSM

Sample Experiments - User can understand the internal working of GSM through these sample experiments. Each sample experiment covers:

- Procedure
- Sample Inputs
- Output
- Comparison chart
- Inference

Index	Objective
Experiment 1	Study how call blocking probability varies as the load on a GSM network is continuously increased.
Experiment 2	Study the effect of mobility on Call blocking probability and Call dropping probability.

Sample Experiment 1

Objective

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

Procedure:

How to Create Scenario & Generate Traffic:

Please navigate through the below given path to,

- **Create Scenario:** “Help → NetSim Help → Running Simulation via GUI → Cellular Networks → Create Scenario”.

Inputs

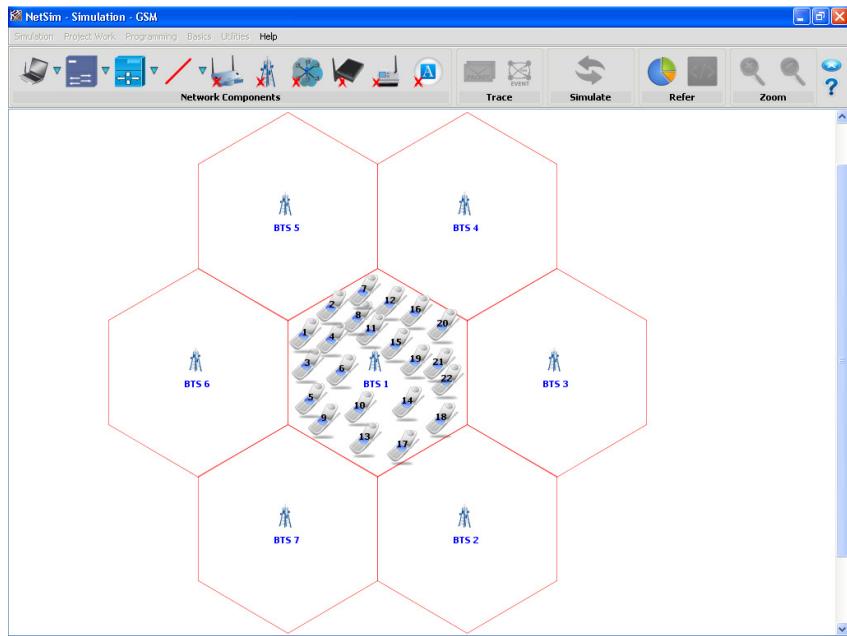
Follow the steps given in the different samples to arrive at the objective.

In this Sample,

- Total no of BTS used: 7
- Total no of MS used: Vary from 4 to 22 in steps of 2.

The devices are inter connected as given below,

- All the MS is placed in the range of BTS1



Set the properties of BTS and MS by following the tables for each sample,

BTS Properties	BTS1
Maximum Uplink bandwidth	891.0 MHz
Minimum Uplink bandwidth	890.0 MHz
Maximum Downlink bandwidth	936.0 MHz
Minimum Downlink bandwidth	935.0 MHz
Channel bandwidth	200 kHz
Number of time slot	8

Inputs for Sample 1

Number of MS = 4

MS Properties	MS 1	MS 3
Destination	MS 2	MS 4
Transmission Type	Point to Point	Point to Point
Traffic Type	Voice	Voice
Call Details		
Distribution	Exponential	Exponential
Mean Call Interval Time (sec)	300	300
Distribution	Exponential	Exponential

Call Duration	60	60
Codec		
Codec	GSM-FR	GSM-FR
Packet Size	33	33
Inter Arrival Time (micro sec)	20000	20000
Service Type	CBR	CBR
Generation rate	0.0132	0.0132
Mobility Model	No Mobility	No Mobility

Inputs for Sample 2

Number of MS = 6

MS Properties	MS 1	MS 3	MS 5
Destination	MS 2	MS 4	MS 6
Transmission Type	Point to Point	Point to Point	Point to Point
Traffic Type	Voice	Voice	Voice
Call Details			
Distribution	Exponential	Exponential	Exponential
Mean Call Interval Time (sec)	300	300	300
Distribution	Exponential	Exponential	Exponential
Call Duration	60	60	60
Codec			
Codec	GSM-FR	GSM-FR	GSM-FR
Packet Size	33	33	33
Inter Arrival Time (micro sec)	20000	20000	20000
Service Type	CBR	CBR	CBR
Generation rate	0.0132	0.0132	0.0132
Mobility Model	No Mobility	No Mobility	No Mobility

Likewise, increase the number of MS by 2 and set properties up to 22 MS.

Simulation Time – 1000 sec

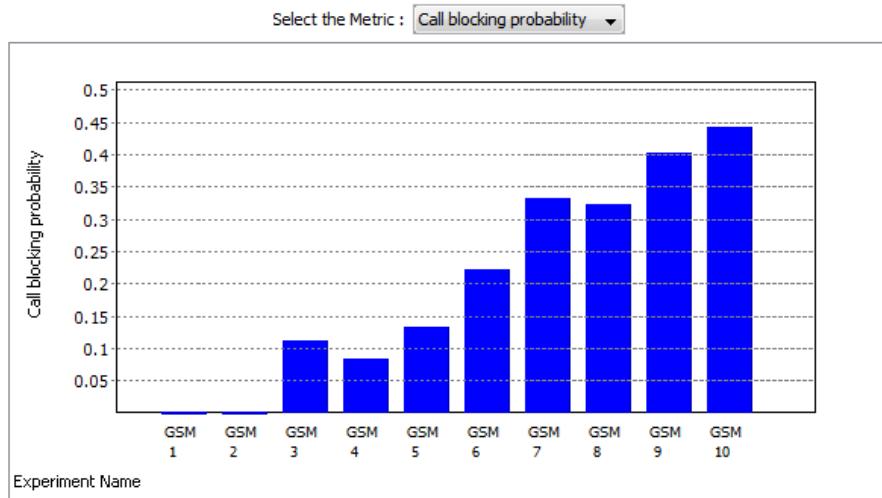
Output

To view the output by using NetSim Sample experiments need to be added onto the Analytics interface. Given below is the navigation for analytics - “Simulation → Analytics”.

Select the experiments by selecting

- Cellular Networks
- Select the Experiments (Note: Click one experiment after another to compare the experiments in the Analytics interface).
- Select the Metric: Call Blocking probability

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.

Inference

When the number of MS is increased from 4 to 22 the call blocking probability increases from 0 to 0.48. This is because; 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 in each base station is fixed (In this experiment it is 40).

Total number of channels in system = 40

Number of channels per BTS = $40/7 = 5$

Number of traffic channels in each BTS = 4 (Here 1 channel is RACH)

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 8 MS may generate. Only after this does the base station not finds free channels and block calls.

Sample Experiment 2

Objective

Study the effect of mobility on Call blocking probability and Call dropping probability.

Procedure:

How to Create Scenario & Generate Traffic:

Please navigate through the below given path to,

- **Create Scenario:** “Help → NetSim Help → Running Simulation via GUI → Cellular Networks → Create Scenario”.

Inputs

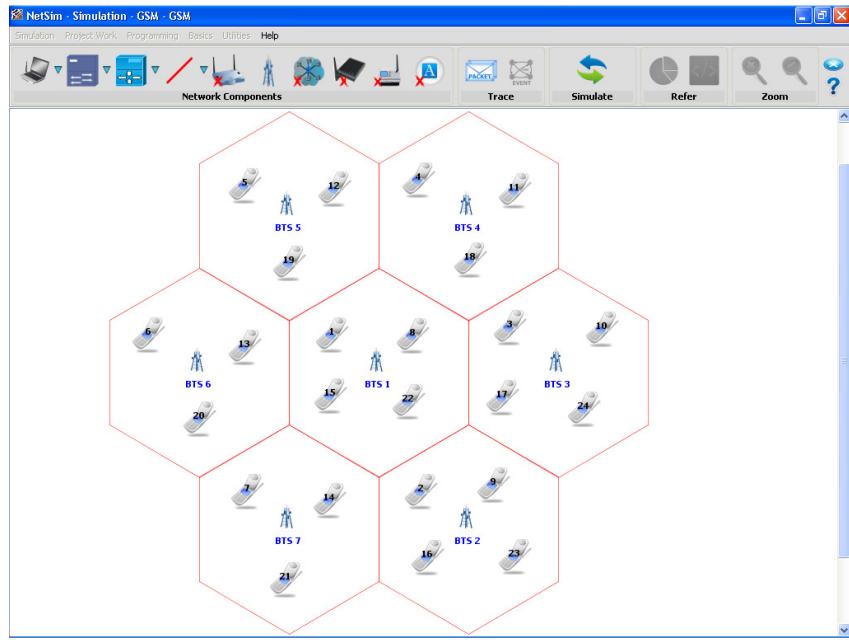
Follow the steps given in the different samples to arrive at the objective.

In all Samples,

- Total no of BTS used: 7
- Total no of MS used: Vary from 2 to 20 in steps of 2.

The devices are inter connected as given below,

- MS1, 8, 15 are placed in BTS1.
- MS2, 9, 16 are placed in BTS2.
- MS3, 10, 17 are placed in BTS3.
- MS4, 11 and 18 are placed in BTS4.
- MS5, 12 and 19 are placed in BTS5.
- MS6, 13 and 20 are placed in BTS6.
- MS7, 14 are placed in BTS7.



Set the properties of BTS and MS by following the tables for each sample,

BTS Properties	BTS1
Maximum Uplink bandwidth	890.4 MHz
Minimum Uplink bandwidth	890.0 MHz
Maximum Downlink bandwidth	935.4 MHz
Minimum Downlink bandwidth	935.0 MHz
Channel bandwidth	200 kHz
Number of time slot	8

Inputs for Sample 1.1

Number of MS = 2

MS Properties	MS 1	MS 2
Destination	MS 2	MS 1
Transmission Type	Point to Point	Point to Point
Traffic Type	Voice	Voice
Call Details		
Distribution	Exponential	Exponential
Mean Call Interval Time (sec)	300	300
Distribution	Exponential	Exponential
Call Duration	60	60

Codec		
Codec	GSM-FR	GSM-FR
Packet Size	33	33
Inter Arrival Time (micro sec)	20000	20000
Service Type	CBR	CBR
Generation rate	0.0132	0.0132
Mobility Model	No Mobility	No Mobility

Inputs for Sample 1.2

Number of MS = 2

MS Properties	MS 1	MS 2
Destination	MS 2	MS 1
Transmission Type	Point to Point	Point to Point
Traffic Type	Voice	Voice
Call Details		
Distribution	Exponential	Exponential
Mean Call Interval Time (sec)	300	300
Distribution	Exponential	Exponential
Call Duration	60	60
Codec		
Codec	GSM-FR	GSM-FR
Packet Size	33	33
Inter Arrival Time (micro sec)	20000	20000
Service Type	CBR	CBR
Generation rate	0.0132	0.0132
Mobility Model	Random Walk	Random Walk
Velocity (m/s)	100	100

Inputs for Sample 2.1

Number of MS = 4

MS Properties	MS 1	MS 2	MS 3	MS 4
Destination	MS 2	MS 1	MS 4	MS 3
Transmission Type	Point to Point	Point to Point	Point to Point	Point to Point
Traffic Type	Voice	Voice	Voice	Voice
Call Details				
Distribution	Exponential	Exponential	Exponential	Exponential
Mean Call Interval Time (sec)	300	300	300	300
Distribution	Exponential	Exponential	Exponential	Exponential
Call Duration	60	60	60	60
Codec				
Codec	GSM-FR	GSM-FR	GSM-FR	GSM-FR
Packet Size	33	33	33	33
Inter Arrival Time (micro sec)	20000	20000	20000	20000
Service Type	CBR	CBR	CBR	CBR
Generation rate	0.0132	0.0132	0.0132	0.0132
Mobility Model	No Mobility	No Mobility	No Mobility	No Mobility

Inputs for Sample 2.2

Number of MS = 4

MS Properties	MS 1	MS 2	MS 3	MS 4
Destination	MS 2	MS 1	MS 4	MS 3
Transmission Type	Point to Point	Point to Point	Point to Point	Point to Point
Traffic Type	Voice	Voice	Voice	Voice
Call Details				
Distribution	Exponential	Exponential	Exponential	Exponential
Mean Call Interval Time (sec)	300	300	300	300
Distribution	Exponential	Exponential	Exponential	Exponential

Codec				
Call Duration	60	60	60	60
Codec	GSM-FR	GSM-FR	GSM-FR	GSM-FR
Packet Size	33	33	33	33
Inter Arrival Time (micro sec)	20000	20000	20000	20000
Service Type	CBR	CBR	CBR	CBR
Generation rate	0.0132	0.0132	0.0132	0.0132
Mobility Model	Random walk	Random walk	Random walk	Random walk
Velocity (m/s)	100	100	100	100

Likewise, set the properties of MS up to 20 MS.

Simulation Time – 1000 sec

Output

To view the output by using NetSim Sample experiments need to be added onto the Analytics interface. Given below is the navigation for analytics - “Simulation → Analytics”.

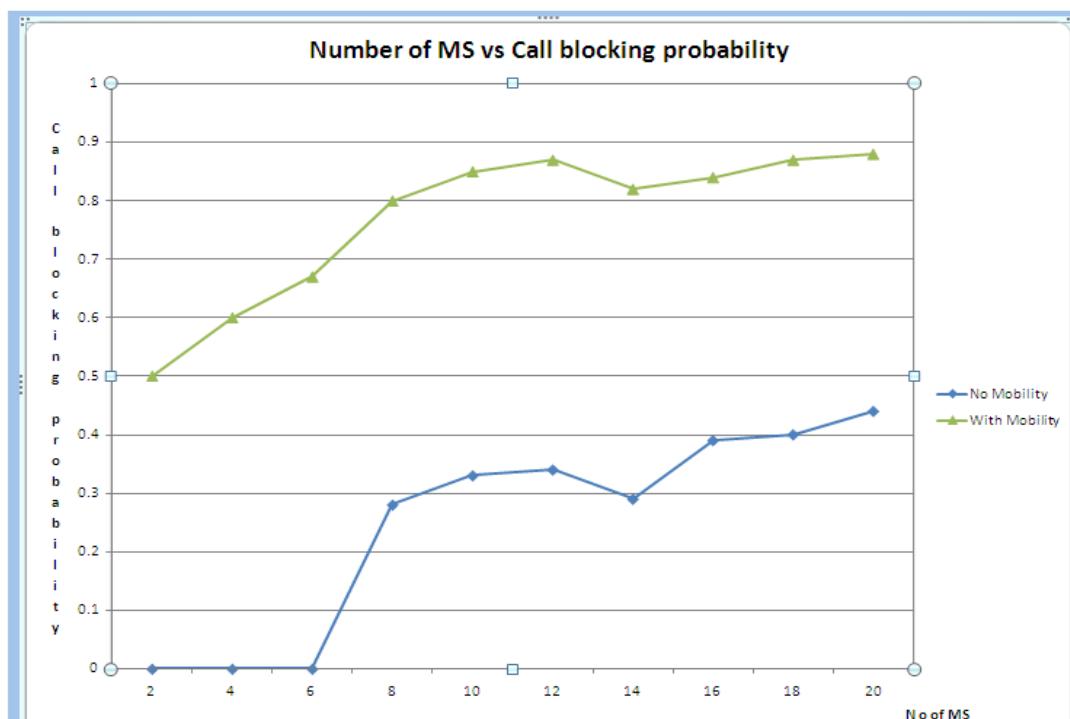
Select the experiments by selecting

- Cellular Networks
- Select the Experiments (Note: Click one experiment after another to compare the experiments in the Analytics interface).
- Add all the *.1 experiment to analytics.
- Click export to .csv
- Select the Metric: Call Blocking probability and save it.
- Now add *.2 experiments to analytics
- Click export to .csv
- Select the Metric: Call Blocking Probability and save it.
- Select the Metric: Call dropping probability.
- Now plot the call blocking probability for both *.1 and *.2 experiment using excel chart

Book1 - Microsoft Excel

A	B	C	D	E
1	Number of MS	Call blocking probability for .1	Call blocking probability for .2	
2	2	0	0.5	
3	4	0	0.6	
4	6	0	0.67	
5	8	0.28	0.8	
6	10	0.33	0.85	
7	12	0.34	0.87	
8	14	0.29	0.82	
9	16	0.39	0.84	
10	18	0.4	0.87	
11	20	0.44	0.88	
12				

Comparison Charts and Inference:



*** All the above plots highly depend upon the placement of MS in the simulation workspace. 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.

The call blocking probability is less in case of no mobility when compare to the same scenario with mobility. This is because,

Total number of channel = 16.

Number of BTS = 7

So, Number of channel per BTS = $16/7 = 2$.

Number of traffic channels = $2-1 = 1$. Here 1 channel is the Random access channel.

From above calculation, it is clear that each base station can handle 1 call at a time.

Now, due to mobility it is likely that mobile station moves from one base station to another and requests for handover. The new base station checks for free channels and if there is no free channel available then that handover request fails and the call is dropped (As we see in Chart 3). So, in case of mobility the network encounters extra blocking due to handover failures. Hence, the call blocking probability is greater in case of mobility as compared to no mobility. The call dropping probability also increases when number of MS increases as explained in another GSM experiment.

If number of MS is, continuously increased then these two graphs will converge. For heavy load, the call blocking probability due to new call request is much higher compared to handover requests. So, the effect of handover on call blocking probability will be very small and two graphs converge.

NetSim – CDMA

Sample Experiments - User can understand the internal working of CDMA through these sample experiments. Each sample experiment covers:

- Procedure
- Sample Inputs
- Output
- Comparison chart
- Inference

Index	Objective
Experiment 1	Study how the number of channels increases and the Call blocking probability decreases as the Voice activity factor of a CDMA network is decreased.

Sample Experiment 1

Objective

Study how the number of channels increases and the Call blocking probability decreases as the Voice activity factor of a CDMA network is decreased

Procedure:

How to Create Scenario & Generate Traffic:

Please navigate through the below given path to,

- **Create Scenario:** “Help → NetSim Help → Running Simulation via GUI → Cellular Networks → Create Scenario”.

Inputs

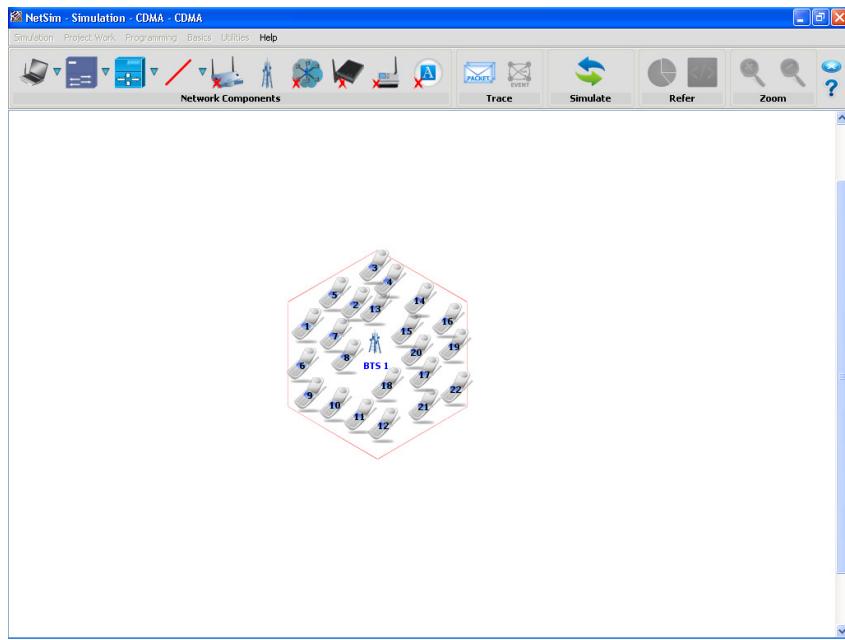
Follow the steps given in the different samples to arrive at the objective.

In all Samples,

- Total no of BTS used: 1
- Total no of MS used: 22

The devices are inter connected as given below,

- All the MS is placed in the range of BTS1



Set the properties of BTS and MS by following the tables for each sample,

MS Properties	MS 1	MS 3	MS 5	MS 7	MS 9
Destination	MS 2	MS 4	MS 6	MS 8	MS 10
Transmission Type	Point to Point				
Traffic Type	Voice	Voice	Voice	Voice	Voice
Call Details					
Distribution	Exponential	Exponential	Exponential	Exponential	Exponential
Mean Call Interval Time (sec)	300	300	300	300	300
Distribution	Exponential	Exponential	Exponential	Exponential	Exponential
Call Duration	60	60	60	60	60
Codec					
Codec	GSM-FR	GSM-FR	GSM-FR	GSM-FR	GSM-FR
Packet Size	33	33	33	33	33
Inter Arrival Time (micro sec)	20000	20000	20000	20000	20000
Service Type	CBR	CBR	CBR	CBR	CBR
Generation rate	0.0132	0.0132	0.0132	0.0132	0.0132
Mobility Model	No Mobility				

Likewise, MS 11 to MS 12, MS 13 to MS 14, MS 15 to MS 17 and MS 19 to MS 20.

Inputs for Sample 1

BTS Properties	BTS
Standards	IS95A/B
Total bandwidth	1.25 MHz
Chip rate	1.2288 McPS
Voice Activity factor	1.0
Transmitter power	20 W
Path loss exponent	3
Fading figure	0.5
Standard deviation	11

Change the voice activity factor from 1.0, 0.9, 0.8, 0.7.... to 0.1.

Simulation Time – 1000 sec

Output

To view the output by using NetSim Sample experiments need to be added onto the Analytics interface. Given below is the navigation for analytics -

“Simulation → Analytics”.

Select the experiments by selecting

- Cellular Networks
- Select the Experiments (Note: Click one experiment after another to compare the experiments in the Analytics interface).
- Select the Metric: Call Blocking probability & Number of channel

Comparison Charts and Inference:

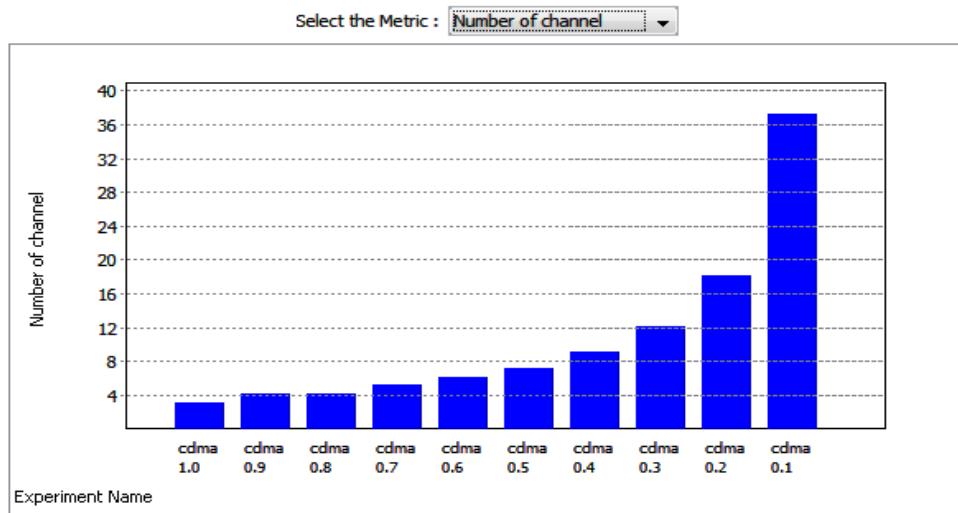


Chart 1

When the system Voice activity factor decreases from 1.0 to 0.1, the number of channels increases from 3 to 37. (Note: All other parameters like Bandwidth 1.25 MHz, chip rate 1.2288McPS, target SNR 6, Path loss exponent 3, Fading figure 0, and standard deviation 11, are constant in all the samples taken.)

In CDMA network, the number of channels is inversely proportional to the voice activity factor.

$$\text{Number of Channels} \propto \frac{1}{\text{Voice activity factor}}$$

Chart 1 is a mirrored form of $y = \frac{1}{x}$ graph. (This is because VAF is decreasing along +ve X)

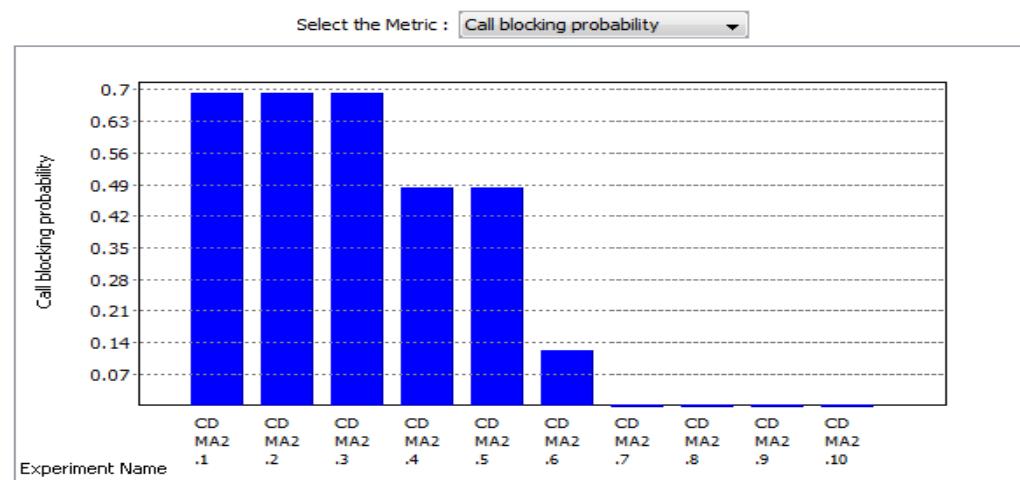


Chart 2

*** 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.

When voice activity factor is decreased the number of channels available increases. Thus the system has more number of channels to handle the same number of calls (Note - Number of MS is constant and their properties are same across all experiments. So, they generate approximately same number of calls throughout).

Let us also understand why the call blocking probability of Sample1, Sample2, and Sample3 is equal and again is equal for Sample4, sample5. In this experiment, all the mobile stations are placed on only BS: base station1. One call requires 2 channel (One is for caller party and another is for called party). So, even if base station 1 has one free channel, the also the call is blocked.

For Sample1,

Total number of channel = 3.

Number of traffic channel = $3-1 = 2$.

Means BTS can handle only 1 call at a time.

For Sample2,

Total number of channel = 4

Number of traffic channel = $4-1 = 3$.

So, here also in this particular scenario where caller and called party are in BTS1, BTS1 can handle only 1 call at a time. The 1 extra channel that is not available in sample1 is wasted throughout. So, number of blocked calls or call blocking probability is same as sample1.

NetSim - WSN

Sample Experiments - User can understand the working of WSN through these sample experiments. Each sample experiment covers:

- Procedure
- Sample Inputs
- Output
- Comparison chart
- Inference

Index	Objective
Experiment 1	Study the SuperFrame Structure and the analyze the effect of SuperFrame order on throughput

Sample Experiment 1

Objective:

Study the SuperFrame Structure and the analyze the effect of SuperFrame order on throughput

Introduction:

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 may enter 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 shall be ignored if BO = 15.

An example of a SuperFrame structure is shown in following Figure.

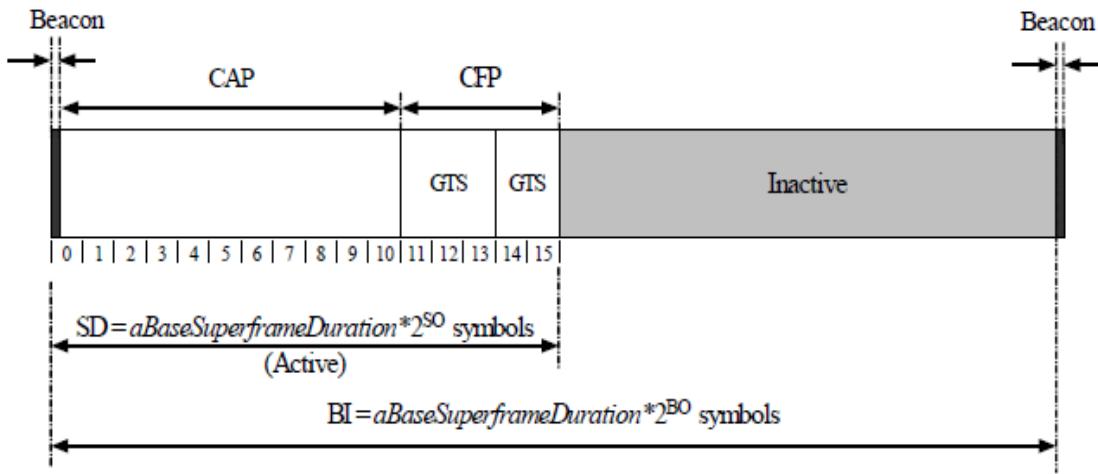


Fig: An example of the Super Frame structure

Theoretical Analysis:

From the above SuperFrame structure,

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

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

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

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)	SuperFrame 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 = 19.166
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 same as the active part of the SuperFrame as sensors can transmit a packet only in the active portion.

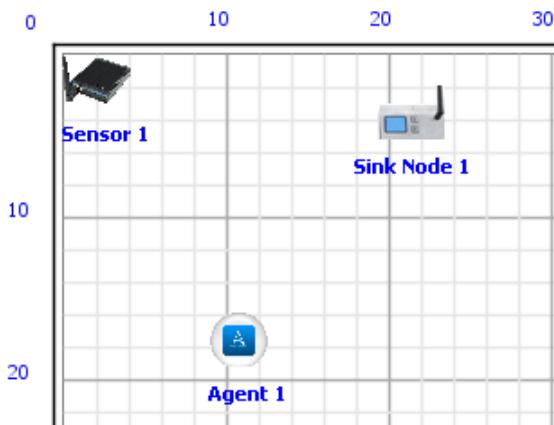
Simulation:

How to Create Scenario & Generate Traffic:

Please refer,

- **Create Scenario:** “Help → NetSim Help → Running Simulation via GUI → Wireless Sensor Networks → Create Scenario”.

Drag & drop one Node onto the Simulation Environment as shown.



Sensor Properties: Accept default properties for Sensor.

Agent Properties: Change the following property for Agent.

Agent Properties	Values
Sensing Interval (ms)	3

PAN Coordinator Properties: Change the following properties for PAN Coordinator.

PAN Coordinator Properties	Values
Beacon Order	10
Super frame Order	10, 9, 8 3 (Vary per experiment)

Run the simulation for different Super Frame Order (change from 10 to 3).

Environment Properties: Accept default properties for Environment.

Simulation Time -30 Sec.

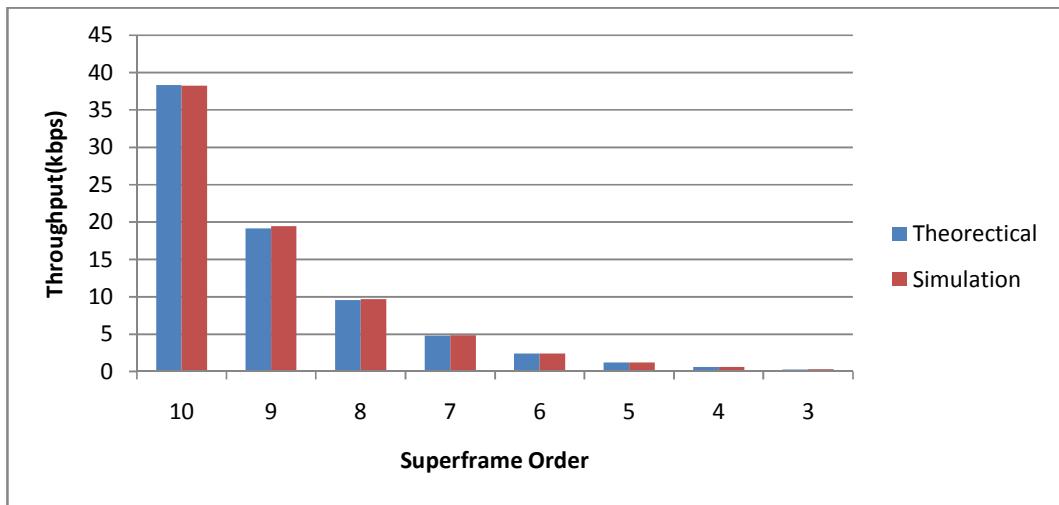
Note: The Simulation Time can be selected only after doing the following two tasks,

- Set the properties of Node, PAN Coordinator & Environment.
- Click on the Validate & Simulate button and save the experiment.

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

SF order	Throughput (Kbps)
10	38.241
9	19.443
8	9.715
7	4.864
6	2.417
5	1.214
4	0.613
3	0.307

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.

Inference:

From the comparison chart both the simulation and theoretical throughputs match except for the case with no inactive period because a sensor will be idle if the last packet in its queue is transmitted and if a packet is generated in inactive period then the packet has to wait in the queue till the next SuperFrame and 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.

NetSim - ZigBee

Sample Experiments - User can understand the working of ZigBee through these sample experiments. Each sample experiment covers:

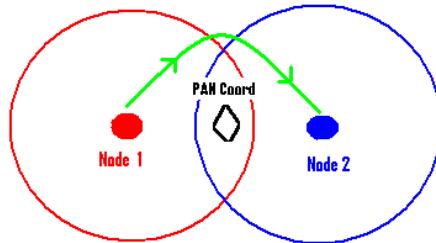
- Procedure
- Sample Inputs
- Output
- Comparison chart
- Inference

Index	Objective
Experiment 1	Analyze the scenario shown, where Node 1 transmits data to Node 2, with no path loss and obtain the theoretical throughput based on IEEE 802.15.4 standard. Compare this with the simulation result.

Sample Experiment 1

Objective:

Analyze the scenario shown, where Node 1 transmits data to Node 2, with no path loss and obtain the theoretical throughput based on IEEE 802.15.4 standard. Compare this with the simulation result.



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 Wireless sensor network IEEE 802.15.4 standard is 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.

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 $2^{macminBE} - 1$ followed by a CCA.
- A 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:

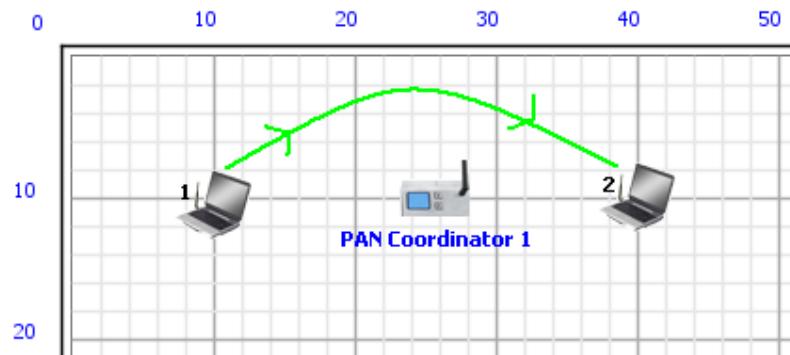
In NetSim the radio turnaround time after a CCA success is not considered.

Simulation:

How to Create Scenario:

Please navigate through the below given path for Create Scenario Help

- **Create Scenario:** “Help → NetSim Help → Running Simulation via GUI → Personal Area Networks → Create Scenario”.



Create the scenario as shown using drag and drop.

Node 1 Properties:

Node Properties	Node - 1
Transmission	Point-to-Point
Destination	Node2
Traffic Type	Data
Distribution	Constant
Application Data Size (Bytes)	501
Distribution	Constant
Mean Inter Arrival Time(μs)	16000
ACK Request	Enable
Retry Limit	7
Transmitter power (mW)	100
Transmitter Range	100

PAN Co-ord Properties: Accept default properties.

Environment Properties: Accept default properties.

Simulation Time: **50 Seconds**

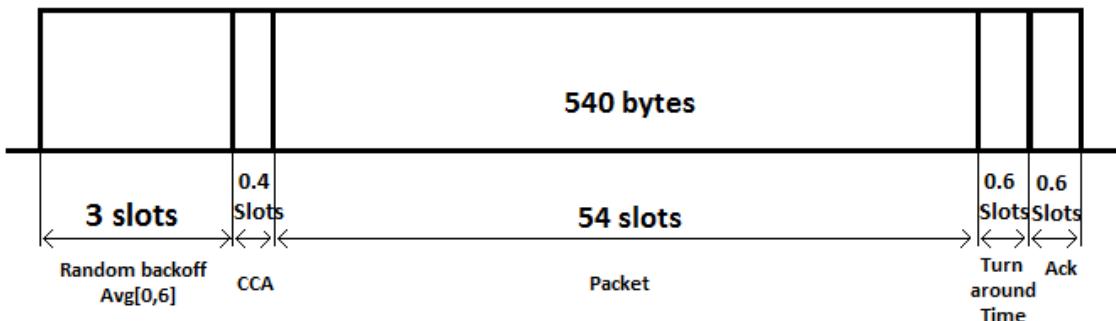
Throughput obtained from the simulation is **206.492 kbps**

Theoretical Analysis:

We have set the Application layer payload as 501 bytes in the Node1 properties and when the packet reaches the physical layer various other headers gets added like

App layer Payload	501 Bytes
IP Header	20 Bytes
MAC Header	13 Bytes
PHY Header	6 Bytes
Packet Size	540 Bytes

In simulation, by default NetSim uses Unslotted CSMA/CA and so a packet transmission happens after a random backoff, CCA 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 standard each slot has 20 Symbols in it and each symbol takes 16μs for transmission

Symbol	T_s	16 μ s
Slots	$20 * T_s$	0.32 ms
Random Backoff Average	$3.5 * Slots$	1.12 ms
CCA	$0.4 * Slots$	0.128 ms
Packet Transmission Time	$54 * Slots$	17.28 ms
Turn-around-Time	$0.6 * Slots$	0.192 ms
ACK Packet Time	$0.6 * Slots$	0.192 ms
Total Time	$59.6 * Slots$	18.912ms

$$Throughput = \frac{501(\text{bytes}) \text{ in App layer}}{18.912 \text{ ms}} = 211.92 \text{ kbps}$$

Inference:

Throughput from simulation	206.492 kbps
Throughput from analysis	211.92 kbps

Throughput from theoretical analysis matches the results of NetSim's discrete event simulation.

Note: The slight difference in throughput is due to fact that the average of random numbers generated for backoff need not be exactly 3 as the simulation is run for short time and also in Network layer DSR protocol is running so, route setup process will take some time.

NetSim – Programming

Programming

This menu contains network programming exercises. Run down the menu and select the desired programming exercise. The programs available are as follows,

- Assignments of Sites to Concentrator
- Address Resolution Protocol
- Cryptography
 - Substitution
 - Transposition
 - XOR
 - Advanced
 - Data Encryption Standard
 - RSA
 - Wired Equivalent Privacy
- Distance Vector Routing
- Error Correcting Code
 - Hamming Code
- Error Detecting Codes
 - Cyclic Redundancy Check
 - Longitudinal Redundancy Check
- Framing Sequence
 - Bit Stuffing
 - Character Stuffing
- Generic Cell Rate Algorithm
 - Virtual Scheduling Algorithm
- IPV4 Addressing
 - Address Mask
 - Binary Conversion
 - Classless InterDomain Routing
 - Network Address
 - Special Addresses
 - Subnetting

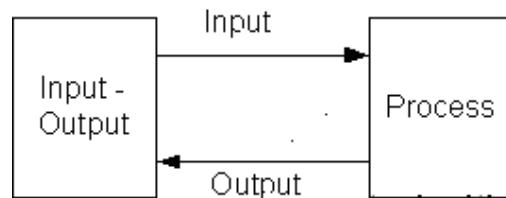
- IPV6 Addressing
 - EUI - 64 Interface Identifier
 - Host Addresses
 - Subnetting
- Leaky Bucket Algorithm
- Multi Level Multi Access
- Multiple Access Technology
 - CDMA
 - TDMA
- PC to PC Communication
 - Socket Programming
 - Chat Application
- Scheduling
- Shortest Path
- Sliding Window Protocol
- Sorting Techniques
- Spanning Tree
- Transmission Flow Control

Upon selection a screen similar to the one shown below will open.

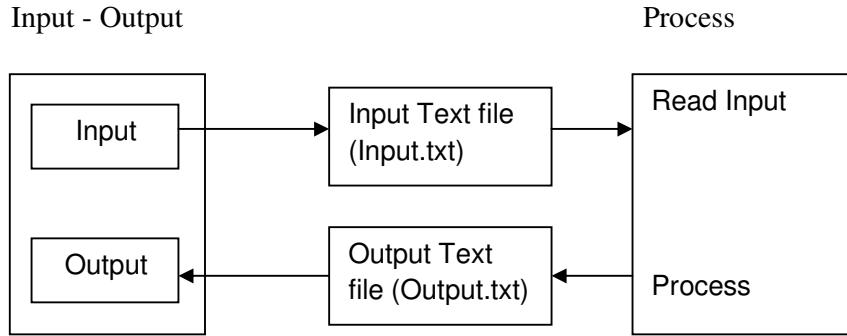
NetSim's Programming Lab has been designed to provide hands - on network programming skills to students. The labs come with a GUI where students can first run the experiment in “**Sample mode**” and subsequently link their own code and visualize its working. Programs can be written in C and the executable can be linked to NetSim.

Architecture

The following **Architecture** is applicable for all the exercises under the **Programming** menu. Each exercise has two modules



Using the **Input - Output** module inputs are given and output is viewed. The working of the concept/algorithm is done in the process module. The link between the **Input - Output** module and process module is as follows



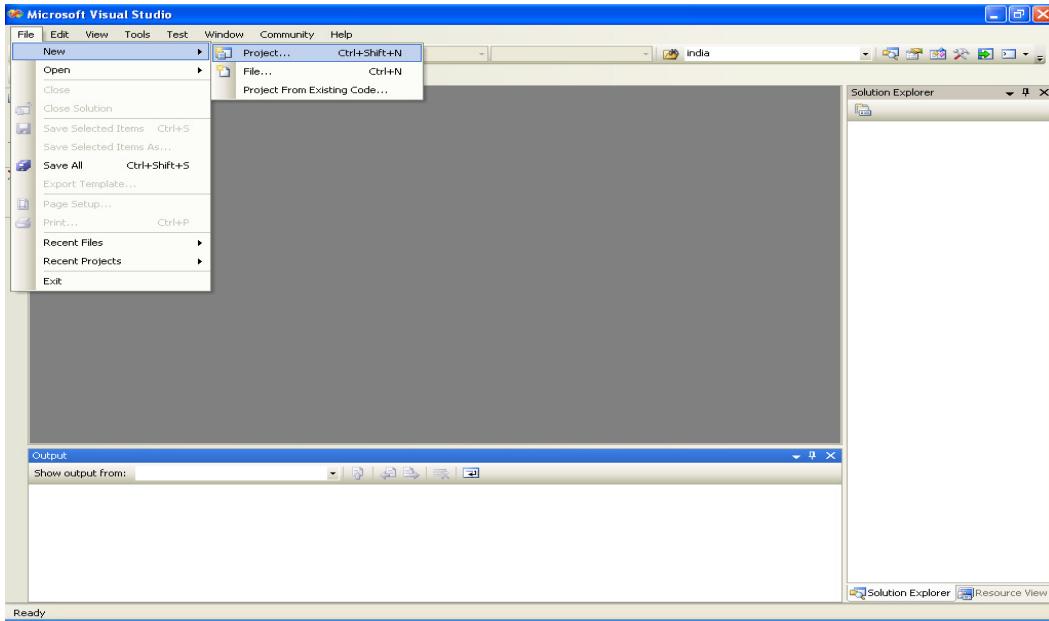
Process is the module for which the user has to write and link the written code when using the user mode.

The code can be written either by C or C++, the executable file created should be linked to the software.

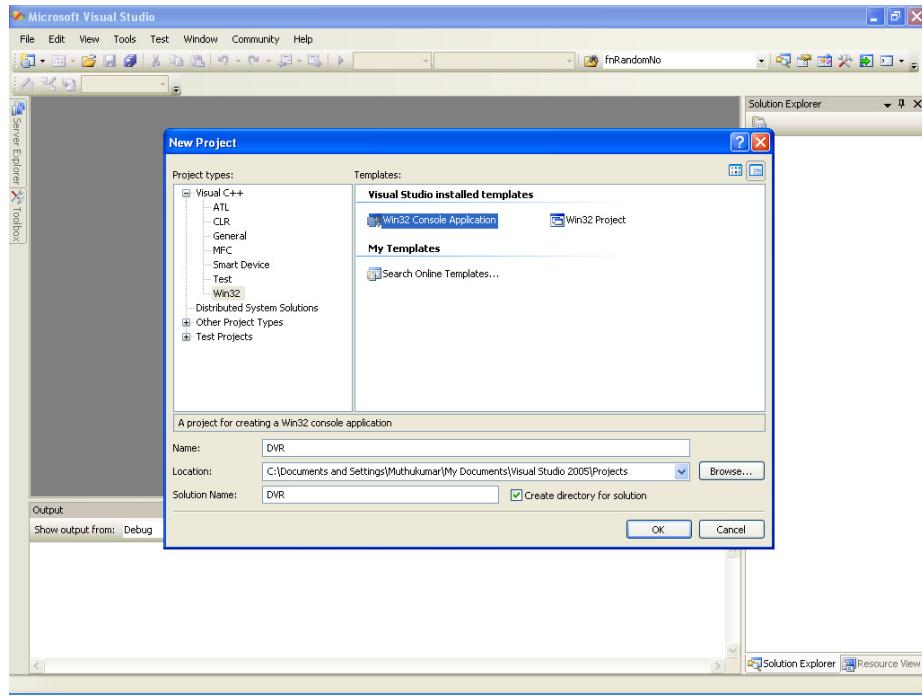
The input and output file format should be as required, as they form the link between the software and the user executable file.

Creating .exe using Visual Studio

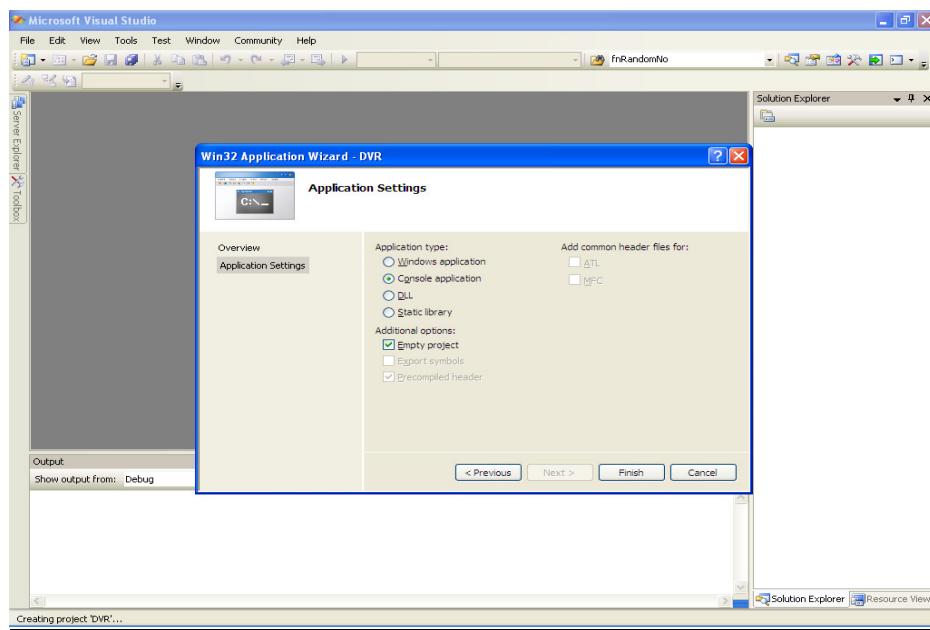
1. Select File→New→Project



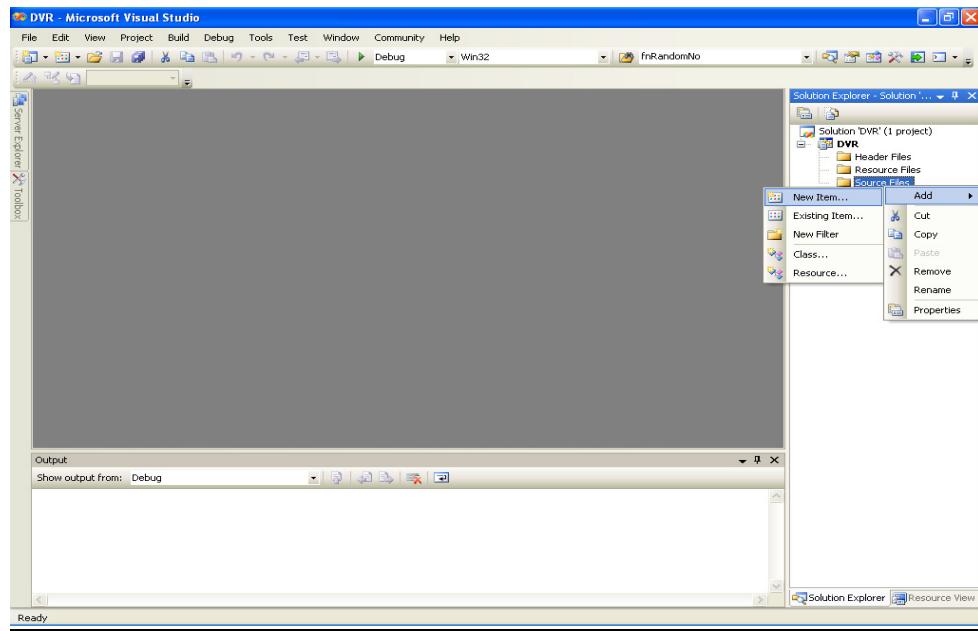
2. Select **Win32Console Application**. Name the project and select location to save the project and then click OK button.



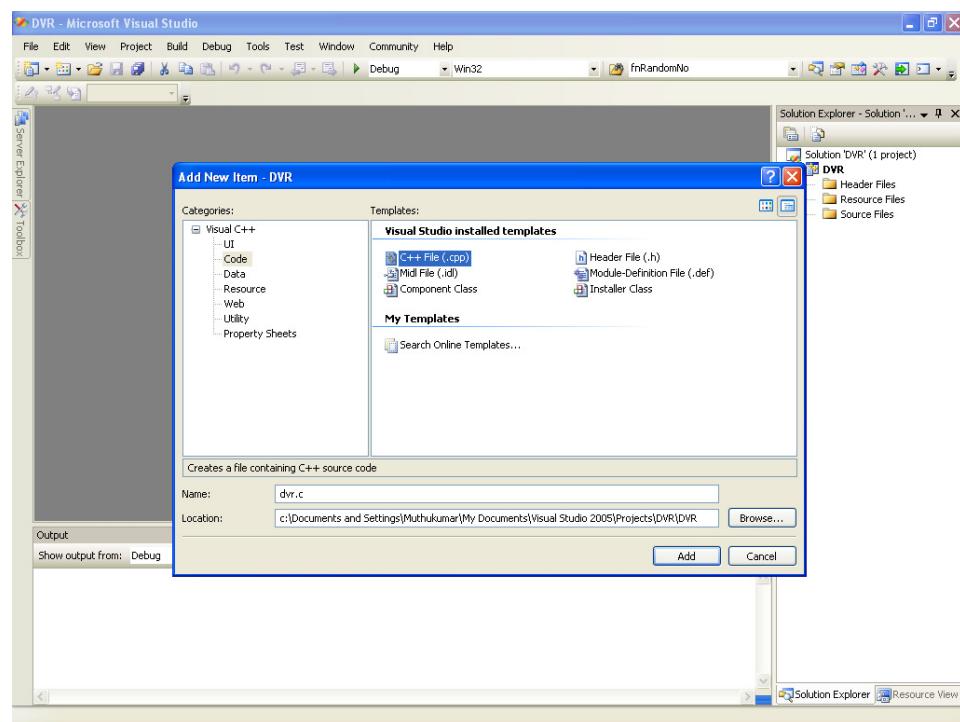
3. Click Next → Check Whether Console application is selected or not. If selected, then select Empty Project, otherwise select Console application and Empty Project and finally click Finish button.



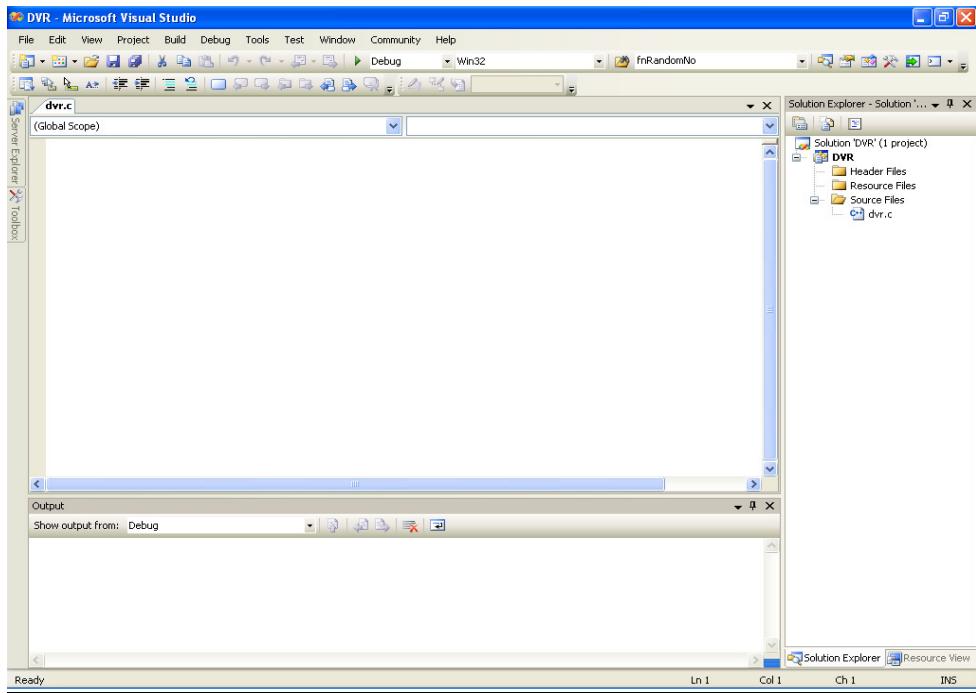
4. Add source codes to the project. Right click on the Source Files → Add → New Item



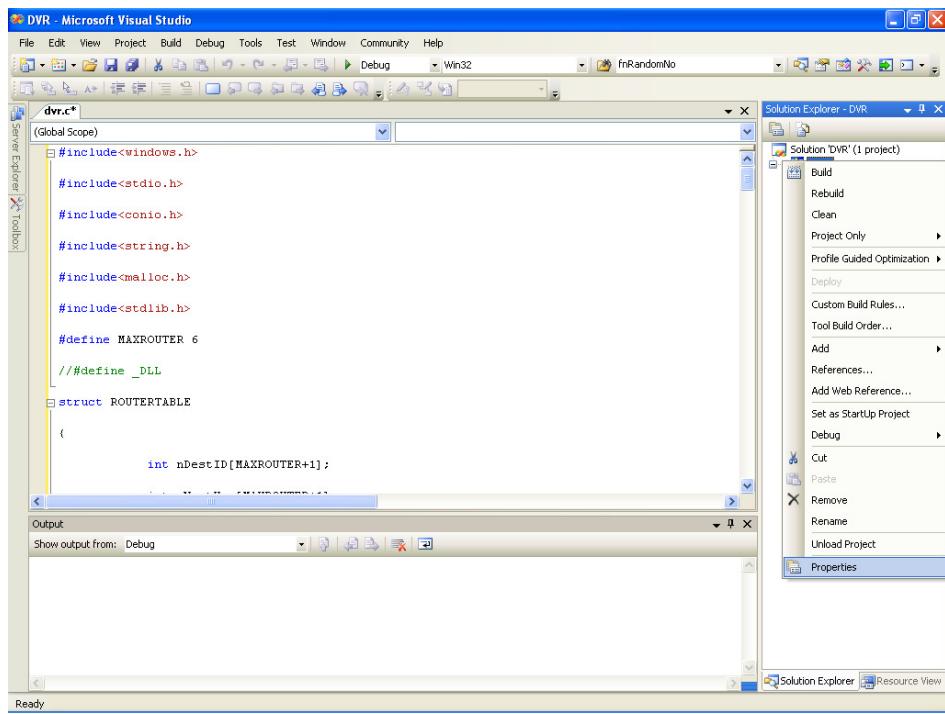
5. Select c++ File (.cpp) and name the file with extension of .c



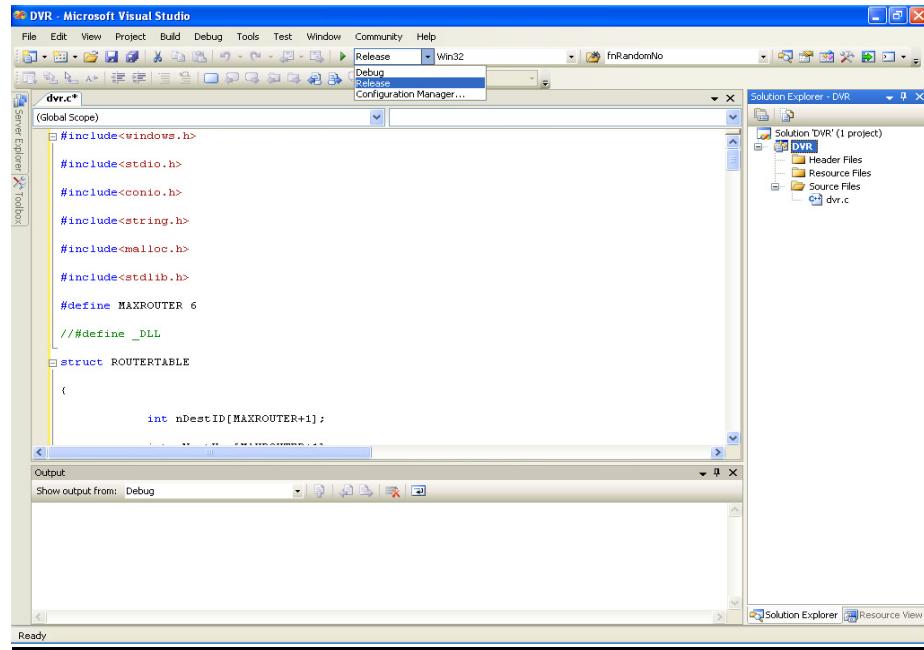
6. Now Source file is created



7. Copy and paste the source code.

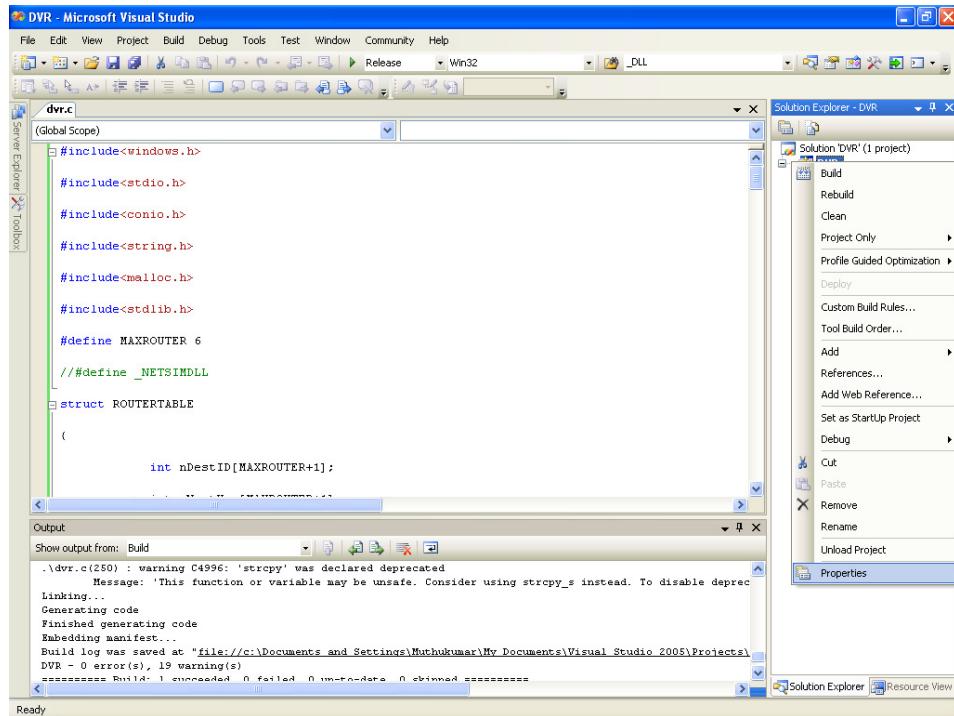


- Select the mode for creating the exe i.e., debug or release mode by choosing the required option as shown in the figure. The preferred setting is debug mode.

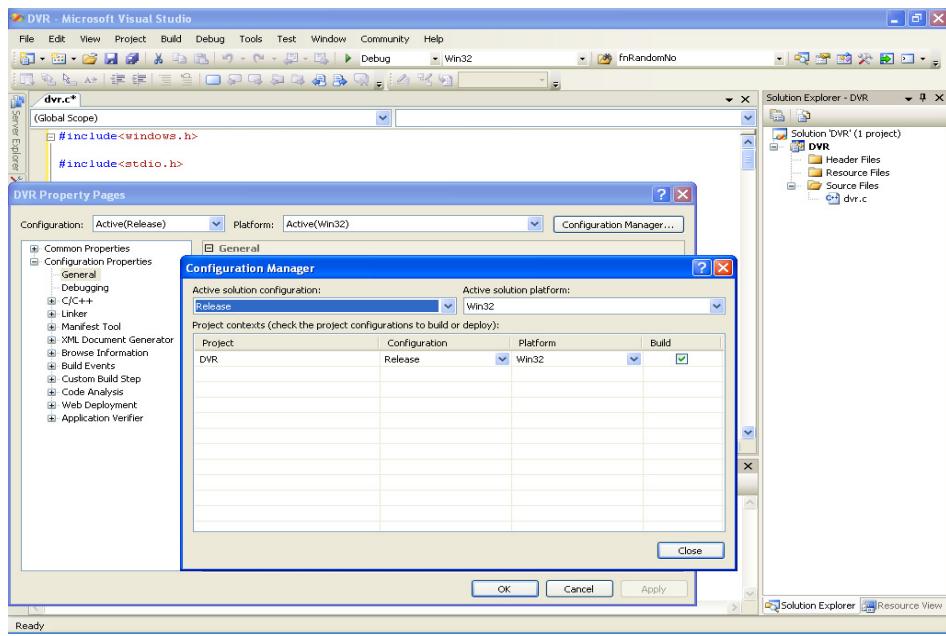


Alternate way to create an exe in Debug/Release Mode

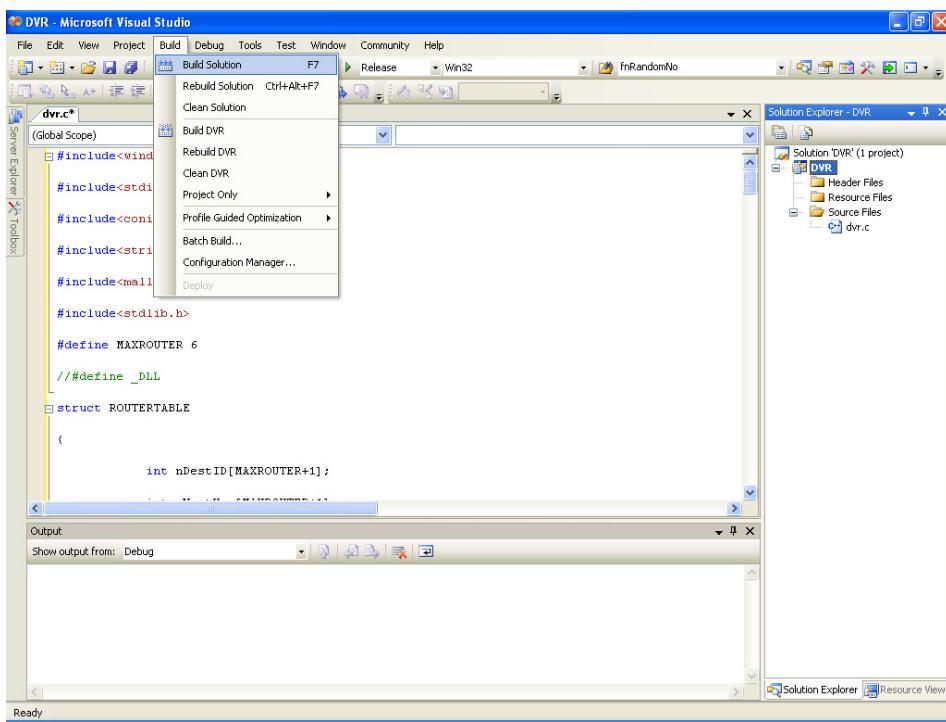
- Right click on the Project File → Properties



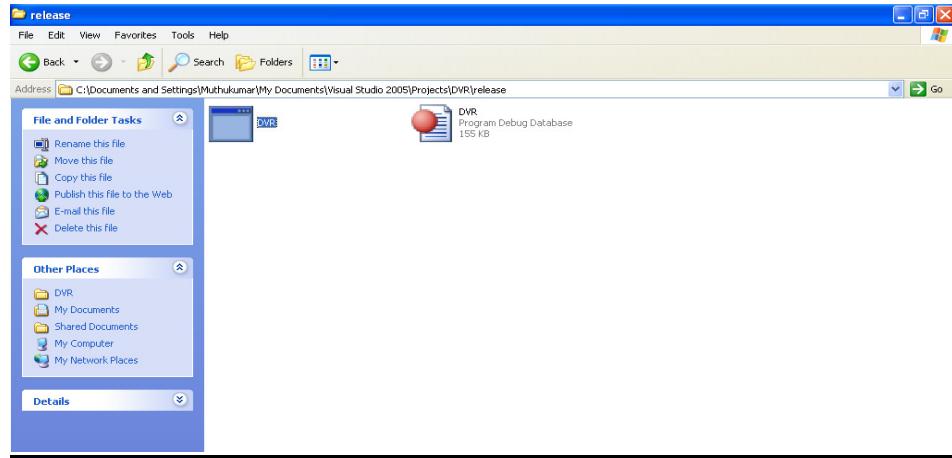
- b. Click Configuration Manager ,a new dialog box will appear. Select Active Solution Configuration as Debug/Release →close and Finally OK.



11. To build the Solution,Select Build Menu →Build Solution



12. Now, Exe is created in the Project Folder as shown below.

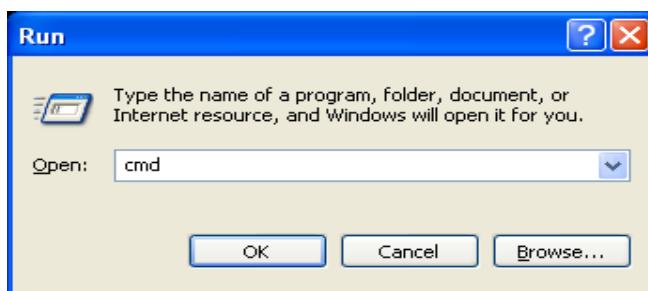


Creating .exe using GCC

- **By using Command Prompt:**

C / C++ files can be created using any editor. Ex: Notepad

Once C / C++ file is ready go to command prompt using Start → Run.



There are three cases for creating Exe (.exe) file using GCC.

Case 1: C programs

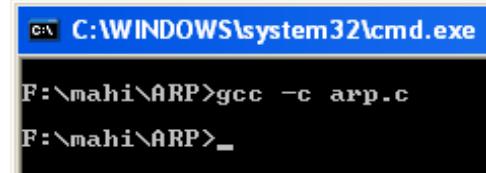
1. Set the path as C program path

```
c:\> C:\WINDOWS\system32\cmd.exe
C:\Documents and Settings\comp.SYS-29>f:
F:\mahi\ARP>_
```

A screenshot of a terminal window. The title bar says 'c:\> C:\WINDOWS\system32\cmd.exe'. The main area shows the command 'cd C:\WINDOWS\system32\cmd.exe' being run, followed by the output 'C:\Documents and Settings\comp.SYS-29>f:' and 'F:\mahi\ARP>_'. The window has a blue header bar and a black body.

2. Create Output file (.o file) using the command

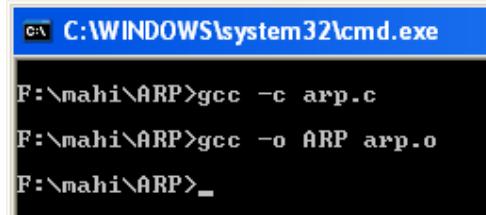
```
gcc - C Filename.c
```



```
C:\WINDOWS\system32\cmd.exe
F:\mahi\ARP>gcc -c arp.c
F:\mahi\ARP>_
```

3. Create exe file (.exe file) using the command

```
gcc -o filename filename.o
```

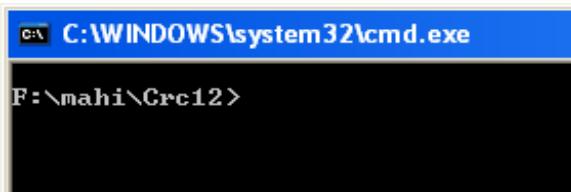


```
C:\WINDOWS\system32\cmd.exe
F:\mahi\ARP>gcc -c arp.c
F:\mahi\ARP>gcc -o ARP arp.o
F:\mahi\ARP>_
```

4. Once exe file is created link that exe file with NetSim → Programming → User mode

Case 2: C++ programs

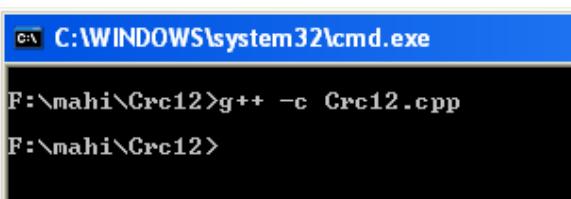
1. Set the path as C++ program path.



```
C:\WINDOWS\system32\cmd.exe
F:\mahi\Crc12>
```

2. Create output file (.o file) using the command.

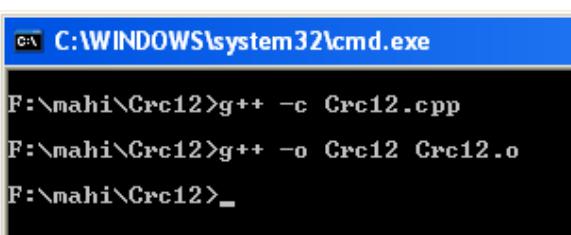
```
g++ -C Filename.c
```



```
C:\WINDOWS\system32\cmd.exe
F:\mahi\Crc12>g++ -c Crc12.cpp
F:\mahi\Crc12>_
```

3. Create exe file (.exe file) using the command

```
g++ -o filename filename.o
```

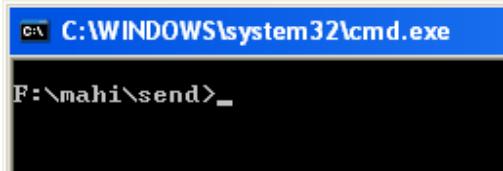


```
C:\WINDOWS\system32\cmd.exe
F:\mahi\Crc12>g++ -c Crc12.cpp
F:\mahi\Crc12>g++ -o Crc12 Crc12.o
F:\mahi\Crc12>_
```

- Once exe file is created link that exe file with NetSim → Programming → User mode

Case 3: Socket programs

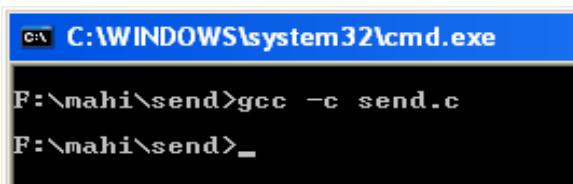
- Set the path to Socket program location.



```
C:\WINDOWS\system32\cmd.exe
F:\mahisend>
```

- Create output file (.o file) using below command.

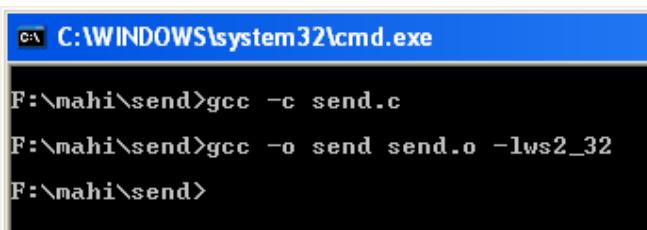
```
gcc -C Filename.c
```



```
C:\WINDOWS\system32\cmd.exe
F:\mahisend>gcc -c send.c
F:\mahisend>
```

- Create exe file (.exe file) using below command

```
gcc -o send send.o -lws2_32
```



```
C:\WINDOWS\system32\cmd.exe
F:\mahisend>gcc -c send.c
F:\mahisend>gcc -o send send.o -lws2_32
F:\mahisend>
```

- Once exe file is created Link that exe file with NetSim → Programming → User mode

MinGW

MinGW, is a collection of freely available and freely distributable Windows specific header files and import libraries combined with GNU GCC compiler. MinGW compiles and links code to be run on Windows platforms providing C, C++ and FORTRAN compilers plus other related tools.

MinGW setup is available under the support files folder in the NetSim CD. The latest version is also available at <http://www.softpedia.com/get/Programming/Components-Libraries/MinGW.shtml>

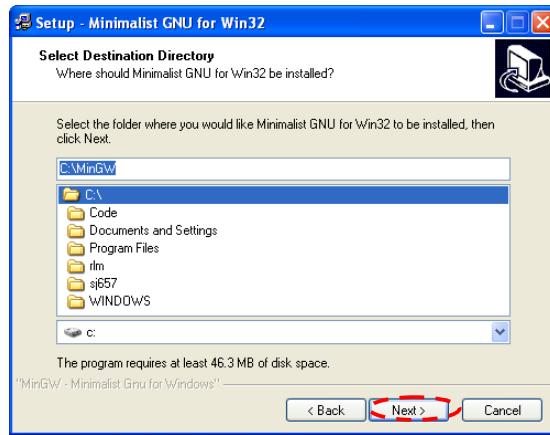
Installation of MinGW:

Setup prepares the installation wizard and software installation begins with a **Welcome Screen**.



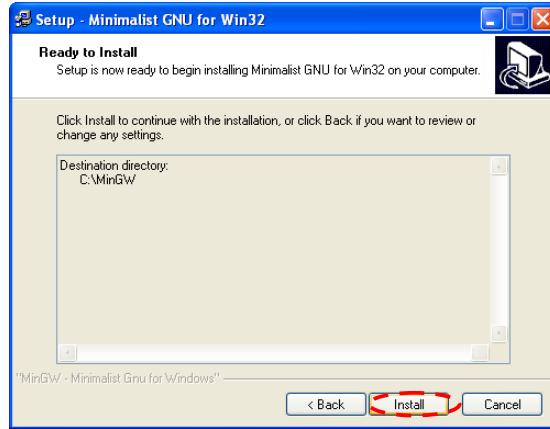
Click on the **Next** button to begin the installation of Minimalist GNU.

In the next screen, the License agreement is displayed. If you agree with the license agreement, you will be requested to read important information about the MinGW software. Click on the **Next** button to continue with the installation. In the next screen, you will be requested for an installation path



Select the path in which Minimalist GNU has to be installed. Click on the **Next** button to continue.

Note: This software has to be installed in the same drive where NetSim is installed.



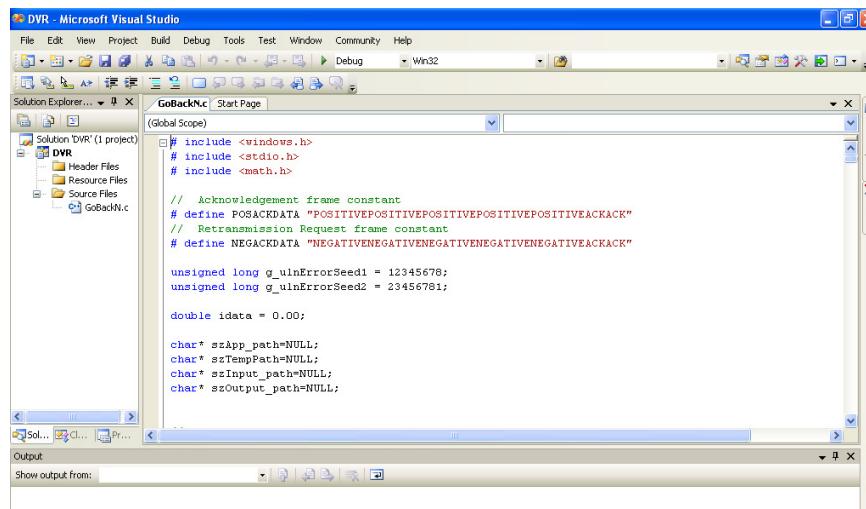
Click on the **Install** button to start installing Minimalist GNU.

After the installation of the software, you will be requested to click Finish to complete the installation process. Click on the **Finish** button to complete the installation.

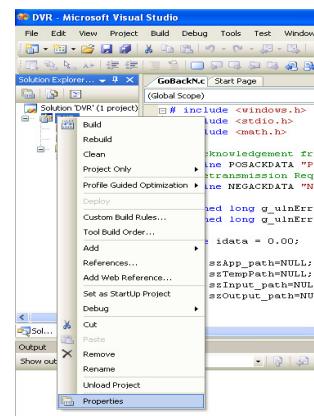
How to De-bug your code linked to NetSim's Programming Exercise

In NetSim, programming exercise menu, users can write and link their code as executables (*.exe). If the user's *.exe file does not provide the correct output, NetSim UI will display the message ‘Error in User Code’. To de-bug your code on getting this message, follow the steps given below:

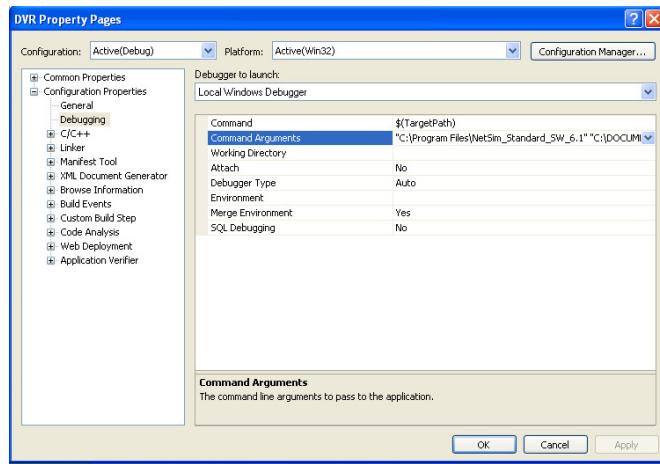
1. Run the scenario again in sample mode. On completion, minimize NetSim and keep it running without closing the programming exercise.
2. Open your code in Visual studio 2005 (or in Eclipse development environment). This is how your code should look when opened in Visual studio as a project (Refer visual studio help on how to create a Win 32 based Exe project)



3. Right click on your project and select properties as shown below



4. Inside the properties window select Debugging and edit the Command Arguments as shown



5. Inside command argument add the following two paths
 - a. Path to where NetSim is installed within double quotes “ “. This is usually C:\Program Files\NetSim Standard. This can be got by right clicking on the NetSim icon and selecting Find target or open file location.
 - b. The windows temporary path which has the NetSim folder for temporary data. This can be got by Start->Run and typing %temp%\NetSim
- c. On clicking this, you will get a path similar to C:\Documents and Settings\George\Local Settings\Temp\NetSim. As mentioned above, this should also be in double quotes “ “, and there should be a single space between the first path and the second path. For example: "C:\Program Files\NetSim_Standard_SW_6.2" "C:\Documents and Settings\George\Local Settings\Temp\NetSim"
6. Now add a breakpoint to your code in your function or in the main (), and proceed with de-bugging
7. At the end check if the output.txt present in the %temp%\NetSim path and the temp.txt present in the %temp%\NetSim path are exactly similar. Exact similarity would indicate that your code will work fine when you run it in use mode the next time.

Address Resolution Protocol

Programming Guidelines

This section guides the user to link his/her own code for Address Resolution Protocol to NetSim.

Pre - Conditions

The user program should read the input from text file named '**Input**' with extension txt.

The user program after executing the concept should write the required output to a file named '**Output**' with extension txt.

Note: The temporary directory is navigated through the following step.

Run → Type "%temp%" → NetSim → "Input.txt" and "Output.txt"

General Program Flow

The program begins with the *Reading of the Inputs* from the input file **Input.txt**.

Executing the required concept and

The results of the program should be written into the output file **Output.txt**.

Input File Format	Output File Format
<p>The first line of the Input.txt contains the number of nodes in the network, following which the IP Addresses are assigned to each nodes (ascending order). The last two lines are the Source node IP Address and the Destination node IP Address respectively.</p> <p>An example format of input.txt is given below,</p> <p>Number_of_Nodes=3</p> <p>Node1_IP_Address=192.168.1.1</p> <p>Node2_IP_Address=192.168.1.2</p> <p>Node3_IP_Address=192.168.1.3</p> <p>Source_Node_IP_Address=192.168.1.3</p> <p>Destination_Node_IP_Address=192.168.1.1</p>	<p>The output file should contain two lines, The first line has the details of the Source (i.e.) in which class it is present. The second line has the result of the ARP i.e., whether destination is present in the class.</p> <p>The second line has the flag value (important) that is used for the animation.</p> <p>The result for the above is:</p> <p>Source is Class C.Destination is present in that class 2</p>

Interface Source Code

Interface Source code written in C is given and using this the user can write only the Address Resolution Protocol inside the function fnARP () using the variables already declared.

To view the interface source code, go to

NetSim Installation path / src / Programming / ARP.c

To find NetSim's Installation path right click NetSim icon and select

- Open file location in Windows 7
- Open file location in Windows Vista
- Properties --> find target in Windows XP.

Sample Scenario:

Objective - To find the Medium Access Control (MAC) Address of the Destination Node using Address Resolution Protocol (ARP).

How to Proceed? - The objective can be executed in NetSim by using the programming exercise available. In the **Programming** menu select **Address Resolution Protocol (ARP)**.

Sample Input - By using the **Input Panel** that is available on the left hand side a **Sample Scenario** is created. The Steps involved in creating are as follows,

- **Sample Mode** should be selected.
- **Click** on the drop down button and select **6 Nodes**.
- List of **Nodes Numbers** along with their **IP Addresses** that would be availed are,

Node Numbers	IP Address
1	192.168.1.1
2	192.168.1.2
3	192.168.1.3
4	192.168.1.4
5	192.168.1.5
6	192.168.1.6

- **ARP Request System** and **ARP Reply System** needs to be selected. That is,
 1. **ARP Request System** → 192.168.1. “1” (Any one **Node Number** to be selected)
 2. **ARP Reply System** → 192.168.1. “6” (Any one **Node Number** to be selected)
- Then **Run** button need to be **clicked**.

Output - Output for the above **Sample** is as follows,

- The **Source Node** (i.e. **Node Number** 1 in the above example) sends the **ARP Request** to the **Connecting Device**.
- The **Connecting Device** then broadcasts the **ARP Request** to all the **Nodes** available in the network.
- The **Destination Node** (i.e. **Node Number** 6 in the above example) sends an acknowledgement in the form of **ARP Reply** (i.e. The **Destination MAC - MAC Address** of the **Destination Node**) to the **Connecting Device**.
- The **Device** then transmits the **ARP Reply** (i.e. The **Destination MAC - MAC Address** of the **Destination Node**) only to the **Source Node**.
- Once the sample experiment is done, then **Refresh** button can be clicked to create new samples.

Assignment of Sites to Concentrator

Programming Guidelines

This section guides the user to link his/her own code for Assignment of Sites to Concentrator to NetSim.

Pre - Conditions

The user program should read the inputted scenario from text file named '**Input**' with extension txt.

The user program after executing the concept should write the required output to a file named '**Output**' with extension txt.

Note: The temporary directory is navigated through the following step.

Run → Type "%temp%" → NetSim → "Input.txt" and "Output.txt"

General Program Flow

The program begins with the *Reading of the Inputs* from the input file **Input.txt**.

Executing the required concept and,

The results of the program should be written into the output file **Output.txt**.

Input file format	Output file format
<p>Input.txt contains the number of sites, number of concentrators, and number of sites that can be assigned to a concentrator, the order in which sites are to be assigned and the distance matrix given by the user. The format of <i>input.txt</i> is:</p> <p>Number_of_Sites=2</p> <p>Number_of_Concentrators=2</p> <p>Sites_per_Concentrator=1</p> <p>Selected_Priority=1>0></p> <p>Distance</p> <p>1>2></p> <p>3>4></p> <p>Note: ‘>’ is the delimiter symbol, which is used to separate each input.</p>	<p>In the output file (i.e. ‘Output.txt’) each line should indicate one complete traverse. For example, for the above input, the output file should be:</p> <p>2>1></p> <p>1>2></p> <p>In each line, the first character indicates the site and the rest indicates the concentrators. The second character (excluding the ‘>’ symbol) is the first concentrator found with minimum distance/cost. The third character is a concentrator, which has the minimum distance/cost compared to the previous one and so on.</p> <p>Note: ‘>’ is the delimiter symbol, which is used to separate each input.</p>

Interface Source Code

Interface Source code code written in C is given using this the user can write only the Assignments of Sites to Concentrators inside the function fnAssignmentsofSites() using the variables already declared.

To view the interface source code, go to

NetSim Installation path / src / Programming/ AssignSites.c

To find NetSim's Installation path right click NetSim icon and select

- Open file location in Windows 7
- Open file location in Windows Vista
- Properties --> find target in Windows XP.

Sample Scenarios:

Objective - To allocate the resource which is concentrator to the require system which is sites based on the distance for each sites to concentrator.

How to proceed? - The objective can be executed in NetSim using the programming exercise available, under programming user has to select Assignment of Sites to Concentrator.

Sample Input - By using the **Input Panel** that is available on the left hand side a **Sample Scenario** is created. The Steps involved in creating are as follows,

- **Sample Mode** should be selected.
- Number of Sites need to be **selected**. The value that has selected ranges from **1** to **7**.
- Number of Concentrators need to be **selected**. The values available for selection are **2, 3 and 4**.
- Number of Sites / Concentrators need to be **selected**. The value that has selected ranges from **1** to **7**.
- **Click** on the image to **select** Priority. **Click** on Change Priority to reset the Priority of the Sites.
- **Enter** the Distance in the given table. Distance should be in the range, **1** to **99** km.
- Then **Run** button need to be **clicked**. **Refresh** button can be used if new Inputs have to be given.
- **Click** on Concept, Algorithm, Pseudo Code & Flow Chart to get help on it.

Output - The following steps are under gone internally,

- The site which has the highest priority searches for a concentrator which is the nearest.

- A red line appears between the site and a concentrator. This indicates the only shortest path available between the site and the concentrators. Hence, it indicates the site has been allocated to that concentrator.
- A site can have only one concentrator, whereas a concentrator can have many sites linked to it.
- Once the sample experiment is done, then **Refresh** button can be **clicked** to create New Samples.

Cryptography - Substitution - Encryption

Programming Guidelines

This section guides the user to link his/her own code for Substitution to NetSim.

Pre-conditions

The user program should read the inputted scenario from text file named '**Input**' with extension txt.

The user program after executing the concept should write the required output to a file named '**Output**' with extension txt.

Note: The temporary directory is navigated through the following step.

Run → Type "%temp%" → NetSim → "Input.txt" and "Output.txt"

General Program Flow

The program begins with the *Reading of the Inputs* from the input file **Input.txt**.

Executing the required concept and, the results of the program should be written into the output file **Output.txt**.

Input File Format	Output File Format
<p>There are six lines in Input.txt file.</p> <p>Cryptographic_Technique=Substitution</p> <p>Cryptographic_Method=Encryption</p> <p>Plain_Text:</p> <p>tetcos</p> <p>Key_Text:</p> <p>2</p>	<p>Plain letter>encrypted letter for the corresponding plain letter></p> <p>t>u>v></p> <p>e>f>g></p> <p>t>u>v></p> <p>c>d>e></p> <p>o>p>q></p> <p>s>t>u></p>

Interface Source Code

Interface Source code written in C is given using this the user can write only the Substitution-Encryption inside the function fnSubstutionEncryption () using the variables already declared.

To view the interface source code, go to

NetSim Installation path / src / Programming/ SubstEncrypt.c

To find NetSim's Installation path right click NetSim icon and select

- Open file location in Windows 7
- Open file location in Windows Vista
- Properties --> find target in Windows XP.

Cryptography - Substitution - Decryption

Programming Guidelines

This section guides the user to link his/her own code for Substitution to NetSim.

Pre - Conditions

The user program should read the inputted scenario from text file named '**Input**' with extension txt.

The user program after executing the concept should write the required output to a file named '**Output**' with extension txt.

Note: The temporary directory is navigated through the following step.

Run → Type "%temp%" → NetSim → "Input.txt" and "Output.txt"

General Program Flow

The program begins with the *Reading of the Inputs* from the input file **Input.txt**.

Executing the required concept and,

The results of the program should be written into the output file **Output.txt**.

Input File Format	Output File Format
There are six lines in Input.txt file. Cryptographic_Technique=Substitution Cryptographic_Method=Decryption Cipher_Text: vgvequ Key_Text: 2	Cipher letter>decrypted letter for the corresponding cipher letter> v>u>t> g>f>e> v>u>t> e>d>c> q>p>o> u>t>s>

Interface Source Code

Interface Source code written in C is given using this the user can write only the Substitution-Decryption inside the function SubstutionDecryption () using the variables already declared.

To view the interface source code, go to

NetSim Installation path / src / Programming/ SubstDecrypt.c

To find NetSim's Installation path right click NetSim icon and select

- Open file location in Windows 7
- Open file location in Windows Vista
- Properties --> find target in Windows XP.

Sample Scenarios:

Objective - Encrypt and decrypt the message with the same key value using Substitution.

How to Proceed? - The objective can be executed in NetSim using the programming exercise available. Under programming user has to select **Cryptography → Substitution**.

❖ Encryption:

Sample Input: In the Input panel the following steps need to be done,

- **Sample Mode** should be **selected**.
- Encryption need to be selected.
- Enter the Plain Text that needs to be encrypted. Maximum of 8 alphabets need to be entered.
- Enter the Key Value. This is an Integer which is within the range **0 to 26**.
- Then **Run** button need to be **clicked**. **Refresh** button can be used if new Inputs have to be given.

Output - The following steps are undergone internally,

- The Plain Text and Key Value entered would be displayed in red color.
- The corresponding Cipher Text would be obtained.
- Once the sample experiment is done, then **Refresh** button can be **clicked** to create New Samples.

❖ Decryption:

Sample Input: In the Input panel the following steps need to be done,

- **Sample Mode** should be **selected**.
- Decryption need to be selected.
- Cipher Text obtained while encryption is filled in the Plain Text and also the Key Text is same as that entered while encrypting.
- Then **Run** button need to be **clicked**. **Refresh** button can be used if new Inputs have to be given.

Output - The following steps are undergone internally,

- The Cipher Text and Key Value entered would be displayed in red color.
- The corresponding Plain Text that had been entered at the time of encrypting is obtained.
- Once the sample experiment is done, then **Refresh** button can be **clicked** to create New Samples.

Cryptography - Transposition - Encryption

Programming Guidelines

This section guides the user to link his/her own code for Transposition Encryption to NetSim.

Pre-conditions

The user program should read the inputted scenario from text file named '**Input**' with extension txt. The user program after executing the concept should write the required output to a file named '**Output**' with extension txt.

Note: The temporary directory is navigated through the following step.

Run → Type "%temp%" → NetSim → "Input.txt" and "Output.txt"

General Program Flow

The program begins with the *Reading of the Inputs* from the input file **Input.txt**.

Executing the required concept and,

The results of the program should be written into the output file **Output.txt**.

Input File Format	Output File Format
Cryptographic_Technique=Transposition	1>3>4>5>2>
Cryptographic_Method=Encryption	T>E>T>C>O>S>T>E>T>C>
Plain_Text: TETCOS	T>S>O>C>E>T>T>E>C>T>
Key_Text: BLORE	

Interface Source Code

Interface Source code written in C is given using this the user can write only the Transposition-Encryption inside the function fnTranspositionEncryption() using the variables already declared.

To view the interface source code, go to

NetSim Installation path / src / Programming/ TranspEncrypt.c

To find NetSim's Installation path right click NetSim icon and select

- Open file location in Windows 7
- Open file location in Windows Vista
- Properties --> find target in Windows XP.

Cryptography - Transposition – Decryption

Programming Guidelines

This section guides the user to link his/her own code for Transposition Decryption to NetSim.

Pre - Conditions

The user program should read the inputted scenario from text file named '**Input**' with extension txt.

The user program after executing the concept should write the required output to a file named '**Output**' with extension txt.

Note: The temporary directory is navigated through the following step.

Run → Type "%temp%" → NetSim → "Input.txt" and "Output.txt"

General Program Flow

The program begins with the *Reading of the Inputs* from the input file **Input.txt**.

Executing the required concept and,

The results of the program should be written into the output file **Output.txt**.

Input File Format	Output File Format
Cryptographic_Technique=Transposition	BELOR>
Cryptographic_Method=Decryption	1>5>2>3>4>
Cipher_Text: TSOCETTECT	T>O>E>T>C>S>C>T>E>T>
Key_Text: BLORE	

Interface Source Code

Interface Source code written in C is given using this the user can write only the Transposition-Decryption inside the function fnTranspositionDecryption() using the variables already declared.

To view the interface source code, go to

NetSim Installation path / src / Programming/ TranspDecrypt.c

To find NetSim's Installation path right click NetSim icon and select

- Open file location in Windows 7
- Open file location in Windows Vista
- Properties --> find target in Windows XP.

Sample Scenarios:

Objective - Encrypt and decrypt the message with the same key text using Transposition.

How to Proceed? - The objective can be executed in NetSim using the programming exercise available. Under programming user has to select **Cryptography → Transposition**.

❖ Encryption:

Sample Input: In the Input panel the following steps need to be done,

- **Sample Mode** should be selected.
- Encryption need to be selected.
- Enter the Cipher Text. Maximum of 14 Characters can be entered.

- Enter the Key Value. Maximum of 8 alphabets can be entered.
- Then **Run** button need to be **clicked**. **Refresh** button can be used if new Inputs have to be given.

Output - The following steps are under gone internally,

- The entered Key Value would be displayed first.
- The order of the Key Value is internally sensed.
- The corresponding Cipher Text is obtained.
- Once the sample experiment is done, then **Refresh** button can be **clicked** to create New Samples.

❖ **Decryption:**

Sample Input: In the Input panel the following steps need to be done,

- Once Encryption is done Decryption has to be selected.
- The Cipher Text obtained for Encryption is the Cipher Text for Decryption. This would be automatically taken. Maximum of 14 Characters can be entered.
- Enter the Key Value will also be taken internally. Maximum of 8 alphabets can be entered.
- Then **Run** button need to be **clicked**. **Refresh** button can be used if new Inputs have to be given.

Output - The following steps are under gone internally,

- The entered Key Value would be displayed first. This is arranged in such a way that the order is changed when compared to Encryption.
- The order of the Key Value is internally sensed.
- The corresponding Plain Text is obtained. This Plain Text would be similar to the Plain Text entered at the time of Encryption.
- Once the sample experiment is done, then **Refresh** button can be **clicked** to create New Samples.

Cryptography - XOR - Encryption

Programming Guidelines

This section guides the user to link his/her own code for XOR Encryption to NetSim.

Pre - Conditions

The user program should read the inputted scenario from text file named '**Input**' with extension txt.

The user program after executing the concept should write the required output to a file named '**Output**' with extension txt.

Note: The temporary directory is navigated through the following step.

Run → Type "%temp%" → NetSim → "Input.txt" and "Output.txt"

General Program Flow

The program begins with the *Reading of the Inputs* from the input file **Input.txt**.

Executing the required concept and,

The results of the program should be written into the output file **Output.txt**.

Input File Format	Output File Format
Cryptographic_Technique=XOR	01010100>01000010>00010110>
Cryptographic_Method=Encryption	01000101>01001110>00001011>
Plain_Text:	01010100>01001100>00011000>
TETCOS	01000011>01001111>00001100>
Key_Text:	01001111>01010010>00011101>
BNLORE	01010011>01000101>00010110>

Interface Source Code

Interface Source code written in C is given using this the user can write only the XOR-Encryption inside the function fnXOREncryption () using the variables already declared.

To view the interface source code, go to

NetSim Installation path / src / Programming/ XorEncrypt.c

To find NetSim's Installation path right click NetSim icon and select

- Open file location in Windows 7
- Open file location in Windows Vista
- Properties --> find target in Windows XP.

Cryptography - XOR - Decryption

Programming Guidelines

This section guides the user to link his/her own code for XOR Decryption to NetSim.

Pre - Conditions

The user program should read the inputted scenario from text file named '**Input**' with extension txt.

The user program after executing the concept should write the required output to a file named '**Output**' with extension txt.

Note: The temporary directory is navigated through the following step.

Run → Type "%temp%" → NetSim → "Input.txt" and "Output.txt"

General Program Flow

The program begins with the *Reading of the Inputs* from the input file **Input.txt**.

Executing the required concept and,

The results of the program should be written into the output file **Output.txt**.

Input File Format	Output File Format
Cryptographic_Technique=XOR	00010110>01000010>01010100>
Cryptographic_Method=Decryption	00001011>01001110>01000101>
Cipher_Text:	00011000>01001100>01010100>
00010110000010110001100000001100000	00001100>01001111>01000011>
1110100010110	00011101>01010010>01001111>
Key_Text:	00010110>01000101>01010011>
BNLORE	

Interface Source Code

Here a skeleton code written in C is given using this the user can write only the XOR-Decryption inside the function fnXORDecryption () using the variables already declared.

To view the interface source code, go to

NetSim Installation path / src / Programming/ XorDecrypt.c

To find NetSim's Installation path right click NetSim icon and select

- Open file location in Windows 7
- Open file location in Windows Vista
- Properties --> find target in Windows XP.

Sample Scenarios:

Objective - Encrypt and Decrypt the message by using the same **Key Text in XOR**.

How to Proceed? - The objective can be executed in **NetSim** using the programming exercise available. In the **Programming** menu select **Cryptography → XOR**.

Note: Encryption should be done first and then **Decryption** should be done.

❖ Encryption:

Sample Inputs - By using the **Input Panel** that is available on the left hand side a **Sample Scenario** is created. The Steps involved in creating are as follows,

- **Sample Mode** should be selected.
- **Encryption Mode** needs to be selected.
- **Plain Text** and **Key Value** need to be entered in the fields available.

That is,

- **Plain Text** → tetcos (Maximum of 8 Characters)
- **Key Text** → 123456 (Maximum of 8 Characters)

Note: If the length of the **Plain Text** and **Key Value** differs then an error would pop out.

- Then **Run** button need to be clicked.

Output - Output for the above Sample is as follows,

- **Plain Text** → t e t c o s
- **Key Text** → 1 2 3 4 5 6
- **Binary of plain text** → 01110011
- **Binary of key text** → 00110110
- **XOR Value** → 01000101

- **ASCII Equivalent** → 69 87 71 87 90 69
- **Cipher Text** → E W G W Z E

❖ **Decryption:**

Sample Inputs - By using the **Input Panel** that is available on the left hand side a **Sample Scenario** is created. The Steps involved in creating are as follows,

- **Sample Mode** should be selected.
- After completing the **Encryption** part, **Decryption Mode** needs to be selected.
- **Cipher Text** (this field is automatically filled) and **Key Value** (this should be same as the one that is entered in Encryption) need to be entered in the fields available. That is,
 - **Cipher Text** →
0100010101011101000111010101110101101001000101 (Cipher Text in Binary Format Maximum 64 bits)
 - **Key Text** → 123456 (Maximum of 8 Characters)
- Then **Run** button need to be clicked.

Output - Output for the above **Sample** is as follows,

- **Cipher Text** → E W G W Z E
- **Key Text** → 1 2 3 4 5 6
- **Binary of cipher text** → 01000101
- **Binary of key text** → 00110110
- **XOR Value** → 01110011
- **ASCII Equivalent** → 116 101 116 99 111 115
- **Plain Text** → t e t c o s

Note -

- **Text in the Input of the Encryption and Output of the Decryption** should be the same if the **Key Value** is same.
- The **Cipher Text** in case of **Encryption** is **Alpha Numeric** (i.e. for the user to understand in a better manner) and in case of **Decryption** the tool converts it to **Binary form** (i.e. since, the tool doesn't recognize **Alpha Numeric**).

Cryptography - Data Encryption Standard (DES) - Encryption

Programming Guidelines

This section guides the user to link his/her own code for DES Encryption to NetSim.

Pre - Conditions

The user program should read the inputted scenario from text file named '**Input**' with extension txt.

The user program after executing the concept should write the required output to a file named '**Output**' with extension txt.

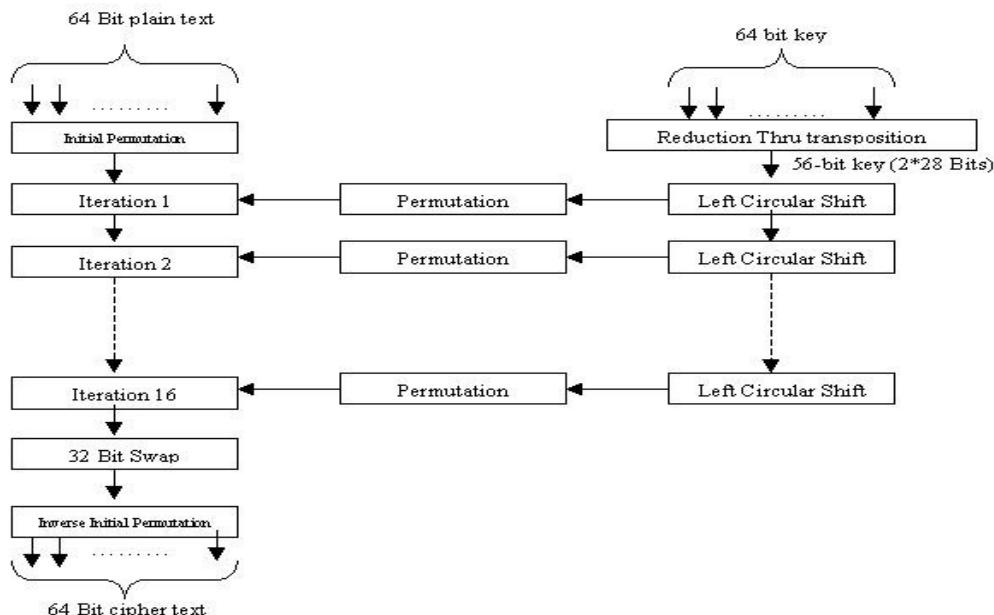
Note: The temporary directory is navigated through the following step.

Run → Type "%temp%" → NetSim → "Input.txt" and "Output.txt"

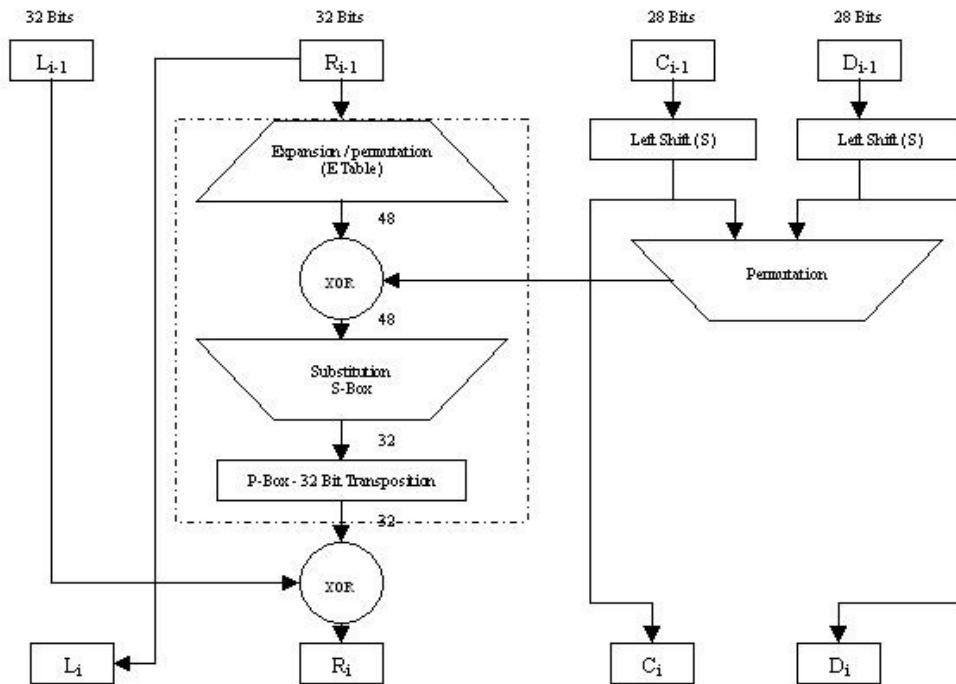
General Program Flow

The program begins with the *Reading of the Inputs* from the input file **Input.txt**.

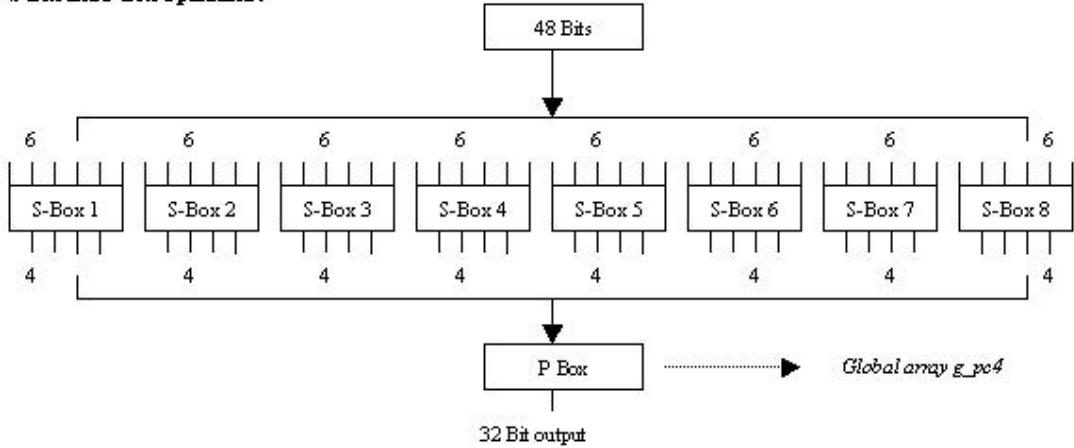
Executing the required concept and,The results of the program should be written into the output file **Output.txt**.



Details of each Iteration: $L_i = R_{i-1}$, $R_i = L_{i-1} \oplus F(R_{i-1}, K_i)$



S-Box and P-Box Operations:



6bit - 4 Bit conversion logic (in each S-Box):

Of the 6 input bits, 4 bits (1,2,3,4) are used for identifying the S-Box column (4 bits representing 16 columns) and the remaining 2 bits (0,5) are used for identifying the S-Box row (2 bits representing 4 rows). The corresponding hexadecimal number is chosen from the S-box. Since each Hexadecimal number represents 4 bits in binary form, the total output is of 32 bits.

Input File Format	Output File Format
Cryptographic_Method=Encryption Key_Text=abcdef1234567890 No_of_Iterations=1 Data=tetcos	The number of lines present depends on the number of iterations chosen. The number of lines equals the number of iterations, plus one. The last line of inputs gives the data (encrypted data if encryption has been chosen, else the decrypted data if decryption has been chosen). The previous lines give the DES Key generated for encryption or decryption. 1101010101111001001101011000010111001100001100 1616075733F724B40

Interface Source Code

Interface Source code written in C is given using this the user can write only the DES-Encryption inside the function fnDESEncryption() using the variables already declared.

To view the interface source code, go to

NetSim Installation path / src / Programming/ DesEncrypt.c

To find NetSim's Installation path right click NetSim icon and select

- Open file location in Windows 7
- Open file location in Windows Vista
- Properties --> find target in Windows XP.

Cryptography - Data Encryption Standard (DES) - Decryption

Programming Guidelines

This section guides the user to link his/her own code for DES Decryption to NetSim.

Pre - Conditions

The user program should read the inputted scenario from text file named '**Input**' with extension txt.

The user program after executing the concept should write the required output to a file named '**Output**' with extension txt.

Note: The temporary directory is navigated through the following step.

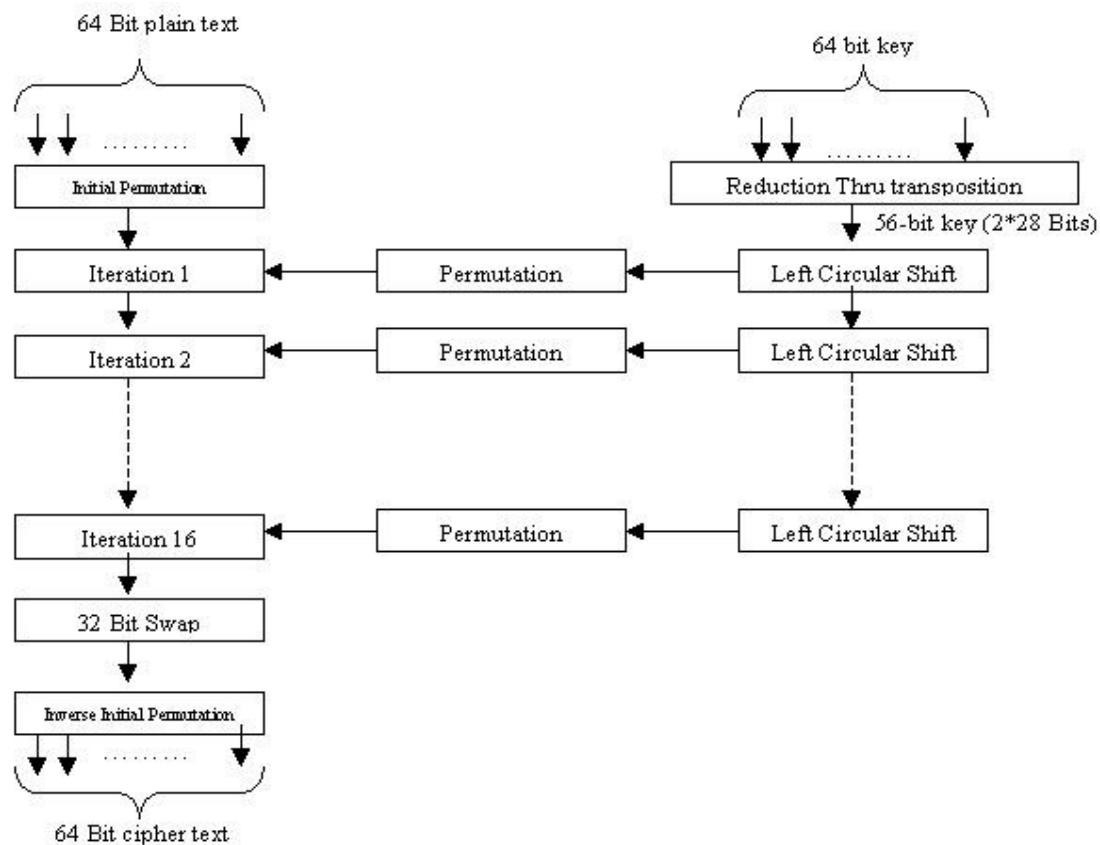
Run → Type "%temp%" → NetSim → "Input.txt" and "Output.txt"

General Program Flow

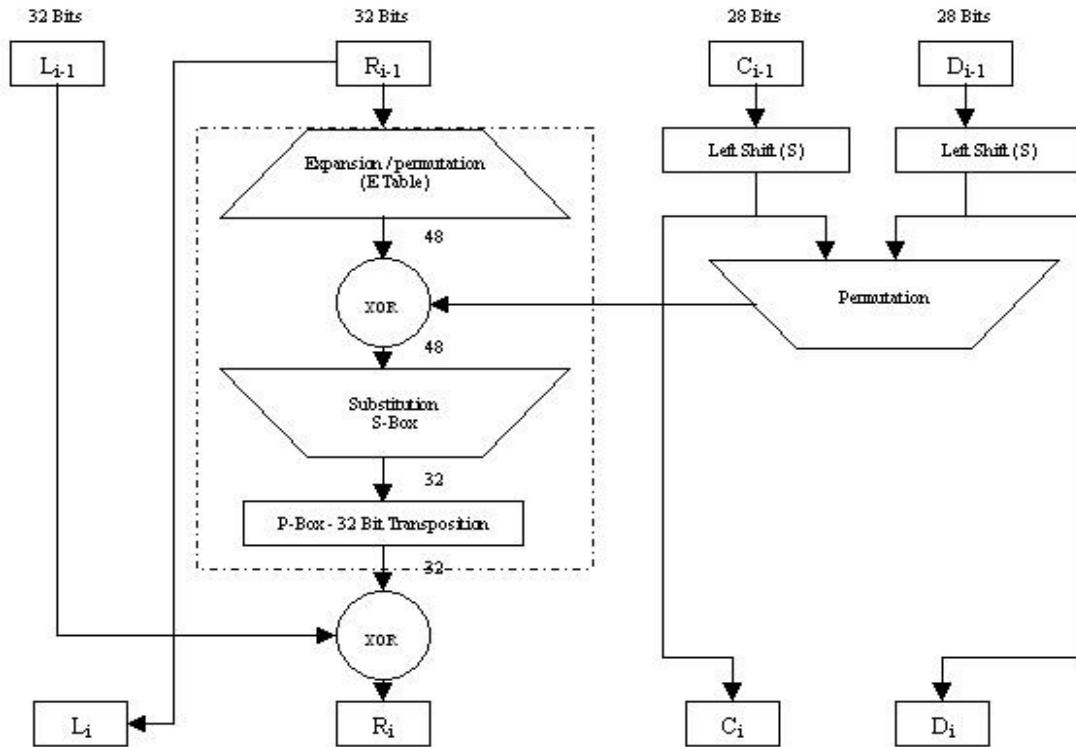
The program begins with the *Reading of the Inputs* from the input file **Input.txt**.

Executing the required concept and,

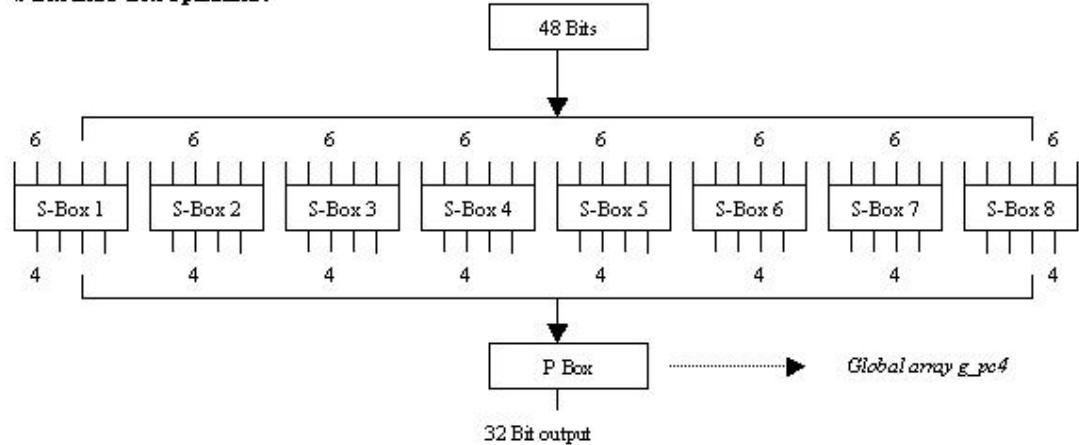
The results of the program should be written into the output file **Output.txt**.



Details of each Iteration: $L_i = R_{i-1}$, $R_i = L_{i-1} \oplus F(R_{i-1}, K_i)$



S-Box and P-Box Operations:



6bit - 4 Bit conversion logic (in each S-Box):

Of the 6 input bits, 4 bits (1,2,3,4) are used for identifying the S-Box column (4 bits representing 16 columns) and the remaining 2 bits (0,5) are used for identifying the S-

Box row (2 bits representing 4 rows). The corresponding hexadecimal number is chosen from the S-box. Since each Hexadecimal number represents 4 bits in binary form, the total output is of 32 bits.

Input File Format	Output File Format
Cryptographic_Method=Decryption Key_Text=abcdef1234567890 No_of_Iterations=1 Data=616075733F724B40	The number of lines present depends on the number of iterations chosen. The number of lines equals the number of iterations, plus one. The last line of inputs gives the data (encrypted data if encryption has been chosen, else the decrypted data if decryption has been chosen). The previous lines give the DES Key generated for encryption or decryption. 11010101011110010011010110000101110011000011001 tetcos

Interface Source Code

Interface Source code written in C is given using this the user can write only the DES-Decryption inside the function fnDESDecryption () using the variables already declared.

To view the interface source code, go to

NetSim Installation path / src / Programming/ DesDecrypt.c

To find NetSim's Installation path right click NetSim icon and select

- Open file location in Windows 7
- Open file location in Windows Vista
- Properties --> find target in Windows XP.

Sample Scenarios:

Objective - Encrypt and decrypt the message with the using DES.

How to Proceed? - The objective can be executed in NetSim using the programming exercise available. Under programming user has to select **Cryptography → Advanced → Data Encryption Standard.**

❖ Encryption:

Sample Inputs - In the Input panel the following steps need to be done,

- **Sample Mode** should be selected.

- Encryption need to be **selected**.
- Enter the Key Text in the provided field. Only hexadecimal characters have to be entered. Maximum of 16 characters can be entered in this field. Characters more than 16 will be filled in the Data field.
- Number of Iterations need to be selected. The value ranges from **1** to **16**.
- Data needs to be entered in this field that has to be encrypted. Data can be entered only when the Enter Key Text field is filed with 16 hexadecimal characters. Maximum of 1500 characters can be stuffed into this field.
- Then **Run** button need to be **clicked**. **Refresh** button can be used if new Inputs have to be given.

Output -

- Key1 (i.e. if Iteration is 1) in binary format is obtained.
- The corresponding Cipher text is obtained.
- Click on Copy button and then on the Paste button to make use of the Cipher text for Decryption.
- Once the sample experiment is done, then **Refresh** button can be **clicked** to create New Samples.

❖ Decryption:

Sample Inputs - In the Input panel the following steps need to be done,

- **Sample Mode** should be **selected**.
- Once the Encryption is done, click on Copy and Paste buttons. The encrypted data is filled in the data field available in the Decryption view.
- Fields such as Enter the Key Text, Number of Iteration and Data is filled in automatically when Copy and Paste button is clicked.
- Then **Run** button need to be **clicked**. **Refresh** button can be used if new Inputs have to be given.

Output - The following steps are undergone internally,

- Key1 (i.e. if Iteration is 1) in binary format is obtained.
- The Data that was entered in the Encryption view would be encrypted and displayed.
- Once the sample experiment is done, then **Refresh** button can be **clicked** to create New Samples.

Rivest-Shamir - Adleman Algorithm (RSA)

Programming Guidelines

This section guides the user to link his/her own code for RSA Algorithm to NetSim.

Pre - Conditions

The user program should read the inputted scenario from text file named '**Input**' with extension txt.

The user program after executing the concept should write the required output to a file named '**Output**' with extension txt.

The path of the input file and the output file can be viewed on clicking the Button "Path" in NetSim.

Note: The temporary directory is navigated through the following step.

Run → Type "%temp%" → NetSim → "Input.txt" and "Output.txt"

General Program Flow

The program begins with the *Reading of the Inputs* from the input file **Input.txt**.

Executing the required concept and,

The results of the program should be written into the output file **Output.txt**.

Input file format	Output file format
<p>Input.txt file has the input data. The format is:</p> <p>Plain_Text=T</p>	<p>dt1>dt2>dt3>dt4>dt5>dt6>dt7>dt8></p> <p>dt 1: specifies the prime P dt 2: specifies the prime Q dt 3: specifies the value of N dt 4: specifies the value of Z dt 5: specifies the value of Kp dt 6: specifies the value of Ks dt 7: specifies the value of the Cipher Text dt 8: specifies the value of the Plain Text</p> <p>Example: For input: 'T' Output: 11>3>33>20>7>3>14>20></p>

Interface Source Code

To view the interface source code, go to

NetSim Installation path / src / Programming/ RSA.c

To find NetSim's Installation path right click NetSim icon and select

- Open file location in Windows 7
- Open file location in Windows Vista
- Properties --> find target in Windows XP.

Sample Scenario:

Objective - Encrypt and decrypt the message with the using DES.

How to Proceed? - The objective can be executed in NetSim using the programming exercise available. Under programming user has to select **Cryptography → Advanced → RSA.**

Sample Inputs - In the Input panel the following steps need to be done,

- **Sample Mode** should be selected.
- Plain Text of only one character has to be entered in the field available.

- Then **Run** button need to be **clicked**. **Refresh** button can be used if new Inputs have to be given.

Output - The following steps are undergone internally,

- Value of the prime numbers P and Q are obtained.
- Value of $N[p^*q]$ is obtained.
- Value of $Z [(p-1)*(q-1)]$ is obtained.
- Value of K_p and K_s is obtained
- Plain Text which is the actual count of the alphabet is obtained.
- The corresponding Cipher Text is obtained.
- Once the sample experiment is done, then **Refresh** button can be **clicked** to create New Samples.

Cryptography - Wired Equivalent Privacy (WEP) – Encryption

Programming Guidelines

This section guides the user to link his/her own code for WEP Encryption to NetSim.

Pre - Conditions

The user program should read the inputted scenario from text file named '**Input**' with extension txt.

The user program after executing the concept should write the required output to a file named '**Output**' with extension txt.

Note: The temporary directory is navigated through the following step.

Run → Type "%temp%" → NetSim → "Input.txt" and "Output.txt"

General Program Flow

The program begins with the *Reading of the Inputs* from the input file **Input.txt**.

Executing the required concept and,

The results of the program should be written into the output file **Output.txt**.

Input File Format	Output File Format
Cryptographic_Method=Encryption Data: TETCOS	There are 3 lines in output.txt file Initialization Vector Key Text Cipher Text Ex: 8A699C 0123456789 364130B9CA018D5B760383CD85528751

Interface Source Code

Interface Source code written in C is given using this the user can write only the WEP-Encryption inside the function fnWEPEncrypt () using the variables already declared.

To view the interface source code, go to

NetSim Installation path / src / Programming/ WEPEncrypt.c

To find NetSim's Installation path right click NetSim icon and select

- Open file location in Windows 7
- Open file location in Windows Vista
- Properties --> find target in Windows XP.

Cryptography - Wired Equivalent Privacy (WEP) - Decryption

Programming Guidelines

This section guides the user to link his/her own code for WEP Decryption to NetSim.

Pre - Conditions

The user program should read the inputted scenario from text file named '**Input**' with extension txt.

The user program after executing the concept should write the required output to a file named '**Output**' with extension txt.

Note:

The temporary directory is navigated through the following step.

Run → Type "%temp%" → NetSim → "Input.txt" and "Output.txt"

General Program Flow

The program begins with the *Reading of the Inputs* from the input file **Input.txt**.

Executing the required concept and,

The results of the program should be written into the output file **Output.txt**.

Input File Format	Output File Format
Cryptographic_Method=Decryption Initialization_Vector=8A699C Key=0123456789 Data: 364130B9CA018D5B760383CD85528751	Plain text Ex: TETCOS

Interface Source Code

Interface Source code written in C is given using this the user can write only the WEP-Decryption inside the function fnWEPDecrypt () using the variables already declared.

To view the interface source code, go to

NetSim Installation path / src / Programming/ WEPDecrypt.c

To find NetSim's Installation path right click NetSim icon and select

- Open file location in Windows 7
- Open file location in Windows Vista
- Properties --> find target in Windows XP.

Sample Scenarios:

Objective - Encrypt and Decrypt the message by using **WEP**.

How to Proceed? - The objective can be executed in **NetSim** using the programming exercise available. In **Programming** menu select **Cryptography → Advanced → Wired Equivalent Privacy**.

Note: **Encryption** should be done first and then **Decryption** should be done.

❖ **Encryption**

Sample Inputs - By using the **Input Panel** that is available on the left hand side a **Sample Scenario** is created. The Steps involved in creating are as follows,

- **Sample Mode** should be selected.
- Under **Cryptographic Method, Encryption Mode** needs to be selected.
- **Input → Data** (Maximum of 1500 Characters) - **NetSim TetCos Bangalore**
- Then **Run** button need to be clicked.

Sample Output - **Output** for the above **Sample** is as follows,

- **Initialization Vector** : 3E41C6
- **Key** : ABCDEF0123
- **The Encrypted Text obtained** -
“786441A955C419F9537576954F897BCC5866549653DB5CA73E1B
4DF63CA939AD”
- **Click on Copy** button and then on **Paste** button. This would **Copy** the **Encrypted Text** onto the **Decryption Input Data field**.

❖ **Decryption**

Sample Inputs - By using the **Input Panel** that is available on the left hand side a **Sample Scenario** is created. The Steps involved in creating are as follows,

- **Sample Mode** should be selected. (This is **Automatically** selected once the **Paste** button is clicked at the time of **Encrypting the Data**).
- Under **Cryptographic Method, Encryption Mode** needs to be selected. (This is **Automatically** selected once the **Paste** button is clicked at the time of **Encrypting the Data**).

- **Enter Key Value** → 3E41C6ABCDEF0123 (16Hexadecimal Characters). (This field is **Automatically** filled once the **Paste** button is clicked at the time of **Encrypting the Data**).
- **Number of Iterations** → 1.
- **Input** → Data - “786441A955C419F9537576954F897BCC5866549653DB5CA73E1B 4DF63CA939AD” (This field is **Automatically** filled once the **Paste** button is clicked at the time of **Encrypting the Data**).
- Then **Run** button need to be clicked.

Sample Output - **Output** for the above **Sample** is as follows,

- **Initialization Vector** : 3E41C6
- **Key** : ABCDEF0123
- The **Encrypted Text** obtained - NetSim TetCos Bangalore

Distance Vector Routing

Programming Guidelines

This section guides the user to link his/her own code for Distance Vector Routing to NetSim.

Pre - Conditions

The user program should read the input from text file named '**Input**' with extension txt.

The user program after executing the concept should write the required output to a file named '**Output**' with extension txt.

Note: The temporary directory is navigated through the following step.

Run → Type "%temp%" → NetSim → "Input.txt" and "Output.txt"

General Program Flow

The program begins with the *Reading of the Inputs* from the input file **Input.txt**.

Executing the required concept and

The results of the program should be written into the output file **Output.txt**.

Input File Format	Output File Format
Router_ID=1>Router_Name=Router_1>No_Of_Neighbour=2>Neighbours_ID=2>5 Router_ID=2>Router_Name=Router_2>No_Of_Neighbour=1>Neighbours_ID=1 Router_ID=3>Router_Name=Router_3>No_Of_Neighbour=0 Router_ID=4>Router_Name=Router_4>No_Of_Neighbour=0 Router_ID=5>Router_Name=Router_5>No_Of_Neighbour=1>Neighbours_ID=1 Router_ID=6>Router_Name=Router_6>No_Of_Neighbour=0	Initial Stage>Source Router ID - 1>Destination RouterID>CostID>Intermediate RouterID Ex: 0>1>2>1>0 0>1>5>1>0 0>2>1>1>0 0>5>1>1>0 1>1>2>1>0 1>1>5>1>0 1>2>1>1>0 1>2>5>2>1 1>5>1>1>0 1>5>2>2>1 2>1>2>1>0 2>1>5>1>0 2>2>1>1>0 2>2>5>2>1 2>5>1>1>0 2>5>2>2>1

Interface Source Code

Interface Source code written in C is given using this the user can write only the Distance Vector Routing inside the function fnDistVectAlgorithm() using the variables already declared.

To view the interface source code, go to

NetSim Installation path / src / Programming/ DVR.c

To find NetSim's Installation path right click NetSim icon and select

- Open file location in Windows 7
- Open file location in Windows Vista
- Properties --> find target in Windows XP.

Sample Scenarios:

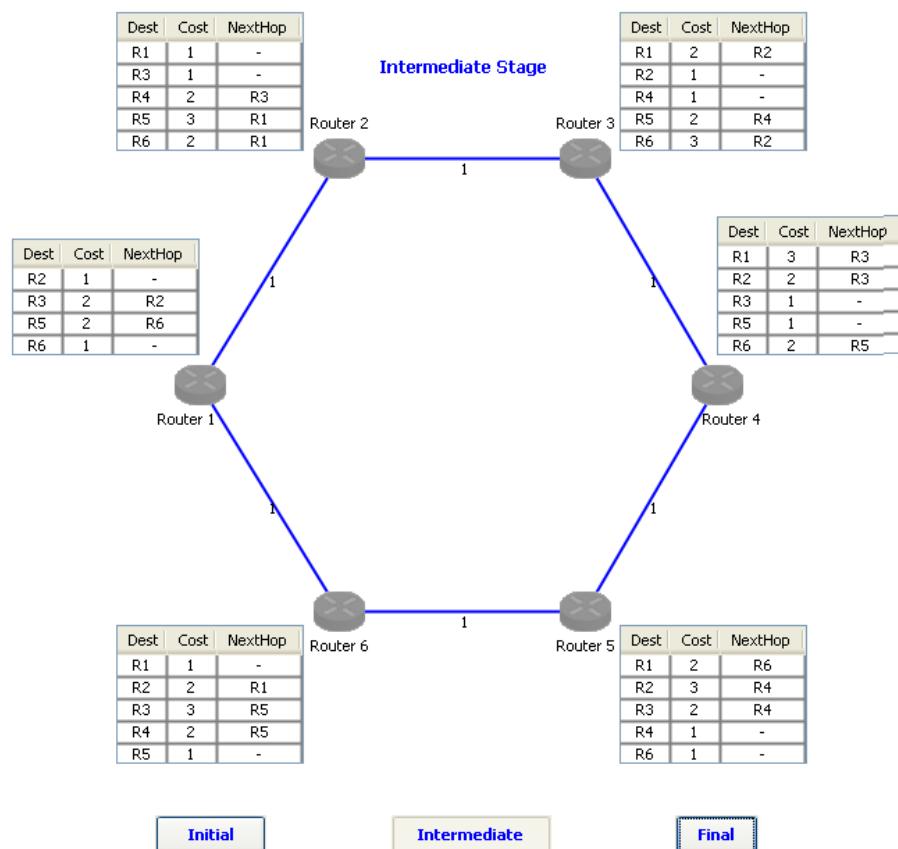
Objective - Find the Shortest Path using Distance Vector Routing.

How to Proceed? - The objective can be executed in NetSim by using the programming exercise available. In the **Programming** menu select **Distance Vector Routing**.

Sample Input - Follow the below given steps,

- Click on two Routers to establish a Path (i.e. 1st Click on Router number 1 and 2, then similarly on Router number 2 and 3, 3 and 4, 4 and 5, 5 and 6, & 6 and 1).
- Router 1, 2, 3, 4, 5, 6 are connected.
- Click on **Initial**, **Intermediate** and **Final** button to execute.

Output - The Output that is obtained is given below,



Initial button can be clicked to view the initial table of the all router.

Intermediate button can be clicked to view the intermediate table of the all router.

Final button can be clicked to view the final table of the all router.

Error Correcting Code - Hamming Code

Programming Guidelines

This section guides the user to link his/her own code for Hamming Code Generator to NetSim.

Pre - Conditions

The user program should read the inputted scenario from text file named '**Input**' with extension txt.

The user program after executing the concept should write the required output to a file named '**Output**' with extension txt.

The path of the input file and the output file can be viewed on clicking the Button "Path" in NetSim.

Note: The temporary directory is navigated through the following step.

Run → Type "%temp%" → NetSim → "Input.txt" and "Output.txt"

General Program Flow

The program begins with the *Reading of the Inputs* from the input file **Input.txt**.

Executing the required concept and,

The results of the program should be written into the output file **Output.txt**.

Input file format	Output file format
Parity=Odd Data=a Error_Position=7 Data_Bits_Original=01100001 Data_Bits_Error=00100001	'Output.txt' file, the first line is the count of check bits to be introduced in the current hamming code application. The next 8 lines is the check bit calculation process, the first four lines are for input data and the next four lines are for the error data. 1 and 2 are the check bits that vary in the Hamming String. The last value is the actual position in the Hamming Code generated where the data bits has been changed. 4 - The number of check bits introduced 3>5>7>9>11>13>0>0>- the check bit of position 1 of input data 3>6>7>10>11>1>1>- the check bit of position 2 of input data 5>6>7>12>13>0>0>- the check bit of position 4 of input data 9>10>11>12>13>0>0>- the check bit of position 8 of input data

	3>5>7>9>11>13>0>0>- the check bit of position 1 of error data 3>6>7>10>11>1>1>- the check bit of position 2 of error data 5>6>7>12>13>1>1>- the check bit of position 4 of error data 9>10>11>12>13>1>1>- the check bit of position 8 of error data 4>8>12>- the check bit position whose value has been changed and the final error position.
--	--

Interface Source Code

To view the interface source code, go to

NetSim Installation path / src / Programming/ HammingCode.c

To find NetSim's Installation path right click NetSim icon and select

- Open file location in Windows 7
- Open file location in Windows Vista
- Properties --> find target in Windows XP.

Sample Scenarios:

Objective - To detect and correct the single bit error occurs in the transmission of data.

How to proceed? - The objective can be executed in NetSim using the programming exercise available, under programming user has to select **Error Correcting Code → Hamming Code**

Sample Inputs - In the Input panel the following steps need to be done,

- **Sample Mode** should be selected.
- Parity need to be selected. Either Odd or Even can be selected.
- Data need to be entered. Maximum of 8 alphabets can be entered.
- Error Position needs to be selected. Based on the input, its values ranges.
- Then **Run** button need to be clicked. **Refresh** button can be used if new Inputs have to be given.

Output - The following steps are under gone internally,

- Data Bits which if in the binary form is obtained. This Data Bits is obtained for Original Data entered as well as for the Error selected.

- Depending on the Parity selected, tables of Hamming String, Original Data and Error Data will be obtained.
- For the Error Data table to be obtained click on the Next button available in the output panel. When the Error Data table is obtained Error Position value is also obtained.
- Once the sample experiment is done, then **Refresh** button can be **clicked** to create New Samples.

Error Detection Code - Cyclic Redundancy Check (CRC) - 12

Programming Guidelines

This section guides the user to Run his/her own code for Cyclic Redundancy Check to NetSim.

Pre - Conditions

The user program should read the input scenario from text file named '**Input**' with extension txt.

The user program after executing the concept should write the required output to a file named '**Output**' with extension txt.

Note: The temporary directory is navigated through the following step.

Run → Type "%temp%" → NetSim → "Input.txt" and "Output.txt"

General Program Flow

The program begins with the *Reading of the Inputs* from the input file **Input.txt**.

Executing the required concept and

The results of the program should be written into the output file **Output.txt**.

Input File Format	Output File format
Algorithm=CRC_12 Condition=No_Error File_Path=C:\Users\P.Sathishkumar\Documents\1 Th.txt>	Output contains two values, which is written in the separate line. The First line has the CRC value of the data (Sender side CRC value). The Second line has the CRC value of the data (Receiver side CRC value). Example: 8CB 000

Interface Source Code

Interface Source code written in C is given using this the user can write only the Cyclic Redundancy Check inside the function fnCRC () using the variables already declared.

To view the interface source code, go to

NetSim Installation path / src / Programming/ Crc12.c

To find NetSim's Installation path right click NetSim icon and select

- Open file location in Windows 7
- Open file location in Windows Vista
- Properties --> find target in Windows XP.

Sample Scenarios:

Objective - To detect the error found in the file transferred between a **Sender** and **Receiver** using **CRC12**.

How to Proceed? - The objective can be executed in **NetSim** using the programming exercise available. In the **Programming Menu** select **Error Detecting Codes → Cyclic Redundancy Check**.

Sample Input

- **For No Error Case -** Follow the below given steps,
 0. **Sample Mode** should be selected.
 1. **Select CRC 12 as Algorithm** from the list available.
 2. Under **Condition, “No Error”** should be selected.

3. Under **Input**, Enter the path of the file name to get its **CRC**. The file should be in “.txt” format which should not exceed 5000bytes.
 4. Click on **Run** button to execute. **Refresh** button can be used if new Inputs have to be given.
- **For Error Case -** Follow the below given steps,
 0. **Sample Mode** should be selected.
 1. **Select CRC 12 as Algorithm** from the list available.
 2. Under **Condition**, “**Error**” should be selected.
 3. Under **Input**, Enter the path of the file name to get its **CRC**. The file should be in “.txt” format which should not exceed 5000bytes.
 4. Click on **Run** button to execute. **Refresh** button can be used if new Inputs have to be given.

Sample Output

- **For No Error Case:** The **Calculated CRC** should be **Zero** when the “.txt file” is received by the **Node2**. The message “**Data Frame is Flowing from Node1 to Node2 with No Error**”.
- **For Error Case:** The **Calculated CRC** should be **Non-Zero** when the “.txt file” is received by the **Node2**. The message “**Data Frame is Flowing from Node1 to Node2 with Error**”.

Error Detection Code - Cyclic Redundancy Check (CRC) – 16

Programming Guidelines

This section guides the user to link his/her own code for Cyclic Redundancy Check to NetSim.

Pre - Conditions

The user program should read the input from text file named ‘**Input**’ with extension txt.

The user program after executing the concept should write the required output to a file named ‘**Output**’ with extension txt.

Note: The temporary directory is navigated through the following step.

Run → Type "%temp%" → NetSim → "Input.txt" and "Output.txt"

General Program Flow

The program begins with the *Reading of the Inputs* from the input file **Input.txt**.

Executing the required concept and

The results of the program should be written into the output file **Output.txt**.

Input File Format	Output File format
Algorithm=CRC_16 Condition=No_Error File_Path=C:\Users\P.Sathishkumar\Documents\1 Th.txt>	Output contains two values, which is the written in the separate line. The First line has the CRC value of the data (Sender side CRC value). The Second line has the CRC value of the data (Receiver side CRC value). Example: 0FCF 0000

Interface Source Code

Interface Source code written in C is given using this the user can write only the Cyclic Redundancy Check inside the function fnCRC () using the variables already declared.

To view the interface source code, go to

NetSim Installation path / src / Programming/ Crc16.c

To find NetSim's Installation path right click NetSim icon and select

- Open file location in Windows 7
- Open file location in Windows Vista
- Properties --> find target in Windows XP.

Sample Scenarios:

Objective - To detect the error found in the file transferred between a **Sender** and **Receiver** using **CRC16**.

How to Proceed? - The objective can be executed in **NetSim** using the programming exercise available. In the **Programming** menu select **Error Detecting Codes → Cyclic Redundancy Check**.

Sample Input

- **For No Error Case** - Follow the below given steps,
 5. **Sample Mode** should be selected.

6. Select **CRC 16** as **Algorithm** from the list available.
 7. Under **Condition**, “**No Error**” should be selected.
 8. Under **Input**, Enter the path of the file name to get its **CRC**. The file should be in “**.txt**” format which should not exceed 5000bytes.
 9. Click on **Run** button to execute. **Refresh** button can be used if new Inputs have to be given.
- **For Error Case** - Follow the below given steps,
10. **Sample Mode** should be selected.
 11. Select **CRC 16** as **Algorithm** from the list available.
 12. Under **Condition**, “**Error**” should be selected.
 13. Under **Input**, Enter the path of the file name to get its **CRC**. The file should be in “**.txt**” format which should not exceed 5000bytes.
 14. Click on **Run** button to execute. **Refresh** button can be used if new Inputs have to be given.

Sample Output

- **For No Error Case:** The **Calculated CRC** should be **Zero** when the “**.txt file**” is received by the **Node2**. The message “**Data Frame is Flowing from Node1 to Node2 with No Error**”.
- **For Error Case:** The **Calculated CRC** should be **Non-Zero** when the “**.txt file**” is received by the **Node2**. The message “**Data Frame is Flowing from Node1 to Node2 with Error**”.

Error Detection Code - Cyclic Redundancy Check (CRC) - 32

Programming Guidelines

This section guides the user to link his/her own code for Cyclic Redundancy Check to NetSim.

Pre - Conditions

The user program should read the input from text file named ‘**Input**’ with extension txt.

The user program after executing the concept should write the required output to a file named ‘**Output**’ with extension txt.

Note: The temporary directory is navigated through the following step.

Run → Type "%temp%" → NetSim → "Input.txt" and "Output.txt"

General Program Flow

The program begins with the *Reading of the Inputs* from the input file **Input.txt**.

Executing the required concept and

The results of the program should be written into the output file **Output.txt**.

Input File Format	Output File format
Algorithm=CRC_32 Condition=No_Error File_Path=C:\Users\P.Sathishkumar\Documents\1 Th.txt>	Output contains two values, which is the written in the separate line. The First line has the CRC value of the data (Sender side CRC value). The Second line has the CRC value of the data (Receiver side CRC value). Example: DD8F598B 00000000

Interface Source Code

Interface Source code written in C is given using this the user can write only the Cyclic Redundancy Check inside the function fnCRC () using the variables already declared.

To view the interface source code, go to

NetSim Installation path / src / Programming/ Crc32.c

To find NetSim's Installation path right click NetSim icon and select

- Open file location in Windows 7
- Open file location in Windows Vista
- Properties --> find target in Windows XP.

Sample Scenarios:

Objective - To detect the error found in the file transferred between a **Sender** and **Receiver** using **CRC32**.

How to Proceed? - The objective can be executed in **NetSim** using the programming exercise available. In the **Programming** menu select **Error Detecting Codes → Cyclic Redundancy Check**.

Sample Input

- **For No Error Case -** Follow the below given steps,

0. **Sample Mode** should be selected.
 1. **Select CRC 32 as Algorithm** from the list available.
 2. Under **Condition**, “**No Error**” should be selected.
 3. Under **Input**, **Enter** the path of the file name to get its **CRC**.
The file should be in “**.txt**” format which should not exceed 5000bytes.
 4. **Click on Run** button to execute. **Refresh** button can be used if new Inputs have to be given.
- **For Error Case -** Follow the below given steps,
 0. **Sample Mode** should be selected.
 1. **Select CRC 32 as Algorithm** from the list available.
 2. Under **Condition**, “**Error**” should be selected.
 3. Under **Input**, **Enter** the path of the file name to get its **CRC**.
The file should be in “**.txt**” format which should not exceed 5000bytes.
 4. **Click on Run** button to execute. **Refresh** button can be used if new Inputs have to be given.

Sample Output

- **For No Error Case:** The **Calculated CRC** should be **Zero** when the “**.txt file**” is received by the **Node2**. The message “**Data Frame is Flowing from Node1 to Node2 with No Error**”.
- **For Error Case:** The **Calculated CRC** should be **Non-Zero** when the “**.txt file**” is received by the **Node2**. The message “**Data Frame is Flowing from Node1 to Node2 with Error**”.

Error Detection Code - Cyclic Redundancy Check (CRC) – CCITT

Programming Guidelines

This section guides the user to link his/her own code for Cyclic Redundancy Check to NetSim.

Pre - Conditions

The user program should read the input scenario from text file named ‘**Input**’ with extension txt.

The user program after executing the concept should write the required output to a file named '**Output**' with extension txt.

Note: The temporary directory is navigated through the following step.

run → Type "%temp%" → NetSim → "Input.txt" and "Output.txt"

General Program Flow

The program begins with the *Reading of the Inputs* from the input file **Input.txt**.

Executing the required concept and

The results of the program should be written into the output file **Output.txt**.

Input File Format	Output File format
Algorithm=CRC_CCITT Condition=No_Error File_Path=C:\Users\P.Sathishkumar\Documents\1 Th.txt>	Output contains two values, which is written in the separate line. The First line has the CRC value of the data (Sender side CRC value). The Second line has the CRC value of the data (Receiver side CRC value). Example: 92BF 0000

Interface Source Code

Interface Source code written in C is given using this the user can write only the Cyclic Redundancy Check inside the function fnCRC () using the variables already declared.

To view the interface source code, go to

NetSim Installation path / src / Programming/ CrcCcitt.c

To find NetSim's Installation path right click NetSim icon and select

- Open file location in Windows 7
- Open file location in Windows Vista
- Properties --> find target in Windows XP.

Sample Scenarios:

Objective - To detect the error found in the file transferred between a **Sender** and **Receiver** using **CRC CCITT**.

How to Proceed? - The objective can be executed in **NetSim** using the programming exercise available. In the **Programming** menu select **Error Detecting Codes → Cyclic Redundancy Check**.

Sample Input

- **For No Error Case** - Follow the below given steps,
 0. **Sample Mode** should be selected.
 1. **Select CRC CCITT** as **Algorithm** from the list available.
 2. Under **Condition**, “**No Error**” should be selected.
 3. Under **Input**, **Enter** the path of the file name to get its **CRC**. The file should be in “**.txt**” format which should not exceed 5000bytes.
 4. **Click on Run** button to execute. **Refresh** button can be used if new Inputs have to be given.

- **For Error Case** - Follow the below given steps,
 0. **Sample Mode** should be selected.
 1. **Select CRC CCITT** as **Algorithm** from the list available.
 2. Under **Condition**, “**Error**” should be selected.
 3. Under **Input**, **Enter** the path of the file name to get its **CRC**. The file should be in “**.txt**” format which should not exceed 5000bytes.
 4. **Click on Run** button to execute. **Refresh** button can be used if new Inputs have to be given.

Sample Output

- **For No Error Case:** The **Calculated CRC** should be **Zero** when the “**.txt file**” is received by the **Node2**. The message “**Data Frame is Flowing** from **Node1** to **Node2 with No Error**”.
- **For Error Case:** The **Calculated CRC** should be **Non-Zero** when the “**.txt file**” is received by the **Node2**. The message “**Data Frame is Flowing** from **Node1 to Node2 with Error**”.

Error Detection Code - Longitudinal Redundancy Check (LRC)

Programming Guidelines

This section guides the user to link his/her own code for Longitudinal Redundancy Check to NetSim.

Pre - Conditions

The user program should read the inputted scenario from text file named '**Input**' with extension txt.

The user program after executing the concept should write the required output to a file named '**Output**' with extension txt.

The path of the input file and the output file can be viewed on clicking the Button "Path" in NetSim.

Note: The temporary directory is navigated through the following step.

Run → Type "%temp%" → NetSim → "Input.txt" and "Output.txt"

General Program Flow

The program begins with the *Reading of the Inputs* from the input file **Input.txt**.

Executing the required concept and,

The results of the program should be written into the output file **Output.txt**.

Input File Format	Output File Format
<p>‘Input.txt’ file has the original data bits, the type of parity (odd or even) and the error data bits. Parity=Odd (or Even) Data=Tetcos Data_Bits_Original=0101010001100101011101000110001 10110111101110011 Data_Bits_Error=001010110100010101010100011000110 110111101110011</p>	<p>‘Output.txt’ file, the First line is the LRC bits value for the original data bits. The Second line is the LRC bits value for the error data bits. 00111010 - LRC bits of the original data 01000101 - LRC bits of the error data</p>

Interface Source Code

Interface source code written in C is given using this the user can write only the Longitudinal Redundancy Check inside the function fnLRC () using the variables already declared. To view the interface source code, go to

NetSim Installation path / src / Programming/ LRC.c

To find NetSim's Installation path right click NetSim icon and select

- Open file location in Windows 7
- Open file location in Windows Vista
- Properties --> find target in Windows XP.

Sample Scenarios:

Objective - To study the working of Longitudinal Redundancy Check (LRC)

How to Proceed? - The objective can be executed in NetSim using the programming exercise available, under programming user has to select Error Detecting Codes in that Longitudinal redundancy Check,

Sample Input - In the Input panel the following steps need to be done,

- **Sample Mode** should be selected.
- Parity need to be selected. Either Odd or Even can be selected.
- Data need to be entered. Maximum of 8 alphabets can be entered.
- Error Position needs to be selected. The value ranges from **1** to **48**.
- Then **Run** button need to be clicked. **Refresh** button can be used if new Inputs have to be given.

Output - The following steps are undergone internally,

- Data Bits which if in the binary form is obtained. This Data Bits is obtained for Original Data entered as well as for the Error selected.
- Depending on the Parity selected, tables of LRC of Original Data Bits and LRC of Error Data Bits are obtained.
- In each of the above mentioned tables last row would contain the Parity.
- Error In Column(s) is obtained.

Framing Sequence – Bit Stuffing

Programming Guidelines

This section guides the user to link his/her own code for Bit Stuffing to NetSim.

Pre - Conditions

The user program should read the input scenario from text file named '**Input**' with extension txt which is in **Temporary Directory**. The user program after executing the concept should write the required output to a file named '**Output**' with extension txt in **Temporary Directory**.

Note: The temporary directory is navigated through the following step.

Run → Type "%temp%" → NetSim → "Input.txt" and "Output.txt"

General Program Flow

The program begins with the *Reading of the Inputs* from the input file **Input.txt**.

Executing the required concept and

The results of the program should be written into the output file **Output.txt**.

Input File	Output File
The “Input.txt” file contains, Destination Address=Value Source Address=Value Data=Message CRC Polynomial=Value Error Status=0 or 1 Seed Value=Value	The “Output.txt” file contains, Message=Value> H=Value>e=Value>l=Value>l=Value>o=Value> Binary Values= Value > CRC Polynomial= Value > CheckSumSender= Value > <Stuffing> Destination Address= Value > Source Address= Value >
Example: Destination Address=00011111 Source Address=00111111 Data=Hello	Data= Value > <DeStuffing> Destination Address= Value > Source Address= Value > Data= Value > Error Status= Value >

CRC Polynomial=10010 Error Status=0 Seed Value=45	CheckSumReceiver= Value > Binary Values= Value > H= Value >e= Value >l= Value >l= Value >o= Value > Message= Value > Example: Message=Hello> H=72>e=101>l=108>l=108>o=111> Binary Values=0100100001100101011011000110110001101111> CRC Polynomial=10011> CheckSumSender=1110> <Stuffing> Destination Address=00011110> Source Address=00111101> Data=01001000011001010110110001101100011011110110> <DeStuffing> Destination Address=00011111> Source Address=00111111> Data=01001000011001010110110001101100011011111110> Error Status=0> CheckSumReceiver=0000> BinaryValues=0100100001100101011011000110110001101111> H=72>e=101>l=108>l=108>o=111> Message=Hello>
---	--

Interface Source Code

Interface source code written in C is given. Using this, the user can write only the functions **fnBitStuffing ()** and **fnDeStuffing ()**, using the variables already declared.

To view the interface source code, go to

NetSim Installation path / src / Programming/ BitStuffing.c

To find NetSim's Installation path right click NetSim icon and select

- Open file location in Windows 7
- Open file location in Windows Vista

- Properties --> find target in Windows XP.

Sample Scenarios:

Objective - To study the working of Bit Stuffing technique.

How to Proceed? - The objective can be executed in NetSim using the programming exercise available, under programming user has to select Framing Sequence → Bit Stuffing.

Sample Inputs - In the Input panel,

- Sample mode should be selected.
- Fill in the HDLC Frame fields available
 - Enter the 8 binary digits in the **Source Address** field.
 - Enter the 8 binary digits in the **Destination Address** field.
 - Enter data, with a maximum of 5 characters
 - CRC polynomial will be chosen by default and select either **Error** or **No Error**.
- Then **Run** button needs to be clicked. **Refresh** button can be used if new Inputs have to be given.

Output - The following steps are done internally by NetSim -

- Data will be converted into ASCII Values. ASCII Values will be converted into Binary Values.
- The Binary Value for CRC polynomial will be shown.
- Checksum will be calculated for the user data in **Sender** side.
- HDLC frame will be formed in Sender side and Bit Stuffing process is animated (Adding ‘0’ for every consecutive **five 1’s**).
- Then Destuffing process will be animated in **Receiver** side.
- Checksum will be calculated in receiver side.
- Again Binary Values will be converted into ASCII values.
- Finally the ASCII values will be converted into Data which the user entered.

Framing Sequence – Character Stuffing

Programming Guidelines

This section guides the user to link his/her own code for Character Stuffing to NetSim.

Pre - Conditions

The user program should read the input scenario from text file named '**Input**' with extension txt which is in **Temporary Directory**

The user program after executing the concept should write the required output to a file named '**Output**' with extension txt in **Temporary Directory**.

Note: The temporary directory is navigated through the following step.

Run → Type "%temp%" → NetSim → "Input.txt" and "Output.txt"

General Program Flow

The program begins with the *Reading of the Inputs* from the input file **Input.txt**.

Executing the required concept and

The results of the program should be written into the output file **Output.txt**.

Input File	Output File
<p>The “Input.txt” file contains,</p> <p>Starting Delimiter=Value</p> <p>Destination Address=Value</p> <p>Source Address=Value</p> <p>Data=Value</p> <p>Checksum=Value</p> <p>Ending Delimiter=Value</p> <p>Example:</p> <p>Starting Delimiter=a</p> <p>Destination Address=eraerwbr</p> <p>Source Address=asdadas</p> <p>Data=sdfgf</p> <p>Checksum=shfsdfsdf</p>	<p>The “Output.txt” file contains,</p> <p>Stuffing></p> <p>Destination Address=Value></p> <p>Source Address= Value></p> <p>Data= Value></p> <p>Checksum= Value></p> <p>DeStuffing></p> <p>Destination Address= Value></p> <p>Source Address= Value></p> <p>Data= Value></p> <p>Checksum= Value></p> <p>Example:</p>

Ending Delimiter=h	Stuffing> Destination Address=eraaerwbr> Source Address=aasdaasdaas> Data=sdfgf> Checksum=shhfsdfsd> DeStuffing> Destination Address=eraerwbr> Source Address=asdadas> Data=sdfgf> Checksum=shfsdfsdsd>
--------------------	--

Interface Source Code

Interface source code written in C is given. Using this, the user can write only the functions **fnCharacterStuffing ()** and **fnDeStuffing ()**, using the variables already declared. To view the interface source code, go to

NetSim Installation path / src / Programming/ CharacterStuffing.c

To find NetSim's Installation path right click NetSim icon and select

- Open file location in Windows 7
- Open file location in Windows Vista
- Properties --> find target in Windows XP.

Sample Scenarios:

Objective - To study the working of Character Stuffing technique.

How to Proceed? - The objective can be executed in NetSim using the programming exercise available, under programming user has to select Framing Sequence → Character Stuffing.

Sample Input - In the Input panel the following steps need to be done,

- **Sample Mode** should be selected.
- Fill in the HDLC Frame fields available.
 - Starting Delimiter has to be filled in.

- Enter the 8 characters in the Destination Address field.
- Enter the 8 characters in the Source Address field.
- Enter in Data field with a maximum of 8 characters.
- Enter the 8 characters in the Check Sum field.
- Ending Delimiter has to be filled in.
- Then **Run** button need to be **clicked**. **Refresh** button can be used if new Inputs have to be given.

Output - The following steps are undergone internally,

- HDLC Frame will be formed in **Sender** side.
- Character stuffing process will be animated in Sender Side.
- Then destuffing process will be animated in Receiver side.
- Once the sample experiment is done **Refresh** button can be **clicked** to create new samples.

Virtual Scheduling Algorithm

Programming Guidelines

This section guides the user to link his/her own code for Virtual Scheduling Algorithm to NetSim.

Pre - Conditions

The user program should read the inputted scenario from text file named '**Input**' with extension txt.

The user program after executing the concept should write the required output to a file named '**Output**' with extension txt.

The path of the input file and the output file can be viewed on clicking the Button "Path" in NetSim.

Note: The temporary directory is navigated through the following step.

Run → Type "%temp%" → NetSim → "Input.txt" and "Output.txt"

General Program Flow

The program begins with the *Reading of the Inputs* from the input file **Input.txt**.

Executing the required concept and,

The results of the program should be written into the output file **Output.txt**.

Input File Format	Output File Format
Input File Content PCR=1 CDVT=0.200 Actual_Arrival_Time=0.100 Previous_Theoretical_Time=0.000	This gives us the format of ' Output.Txt ' to which the user has to write his program Each value should be separated by a delimiter '>'. There should be no extra white spaces or blank lines. Example: Value1>Value2>Value3> Value1 – Cell conformation Flag, Value 2 – Next Cells Expected arrival Time Value 3 – Current cell's Actual arrival time. Note: The above convention to write into the ' Output.Txt ' are mandatory . A sample file format is as follows: 3>2.000>1.000>

Interface Source Code

Interface source code written in C is given using this the user can write only the Longitudinal Redundancy Check inside the **void main ()** using the variables already declared. To view the interface source code, go to

NetSim Installation path / src / Programming/ VSA.c

To find NetSim's Installation path right click NetSim icon and select

- Open file location in Windows 7
- Open file location in Windows Vista
- Properties --> find target in Windows XP.

Sample Scenarios:

Objective - To study the working of Virtual Scheduling Algorithm.

How to Proceed? - The objective can be executed in NetSim using the programming exercise available, under programming user has to select Virtual Scheduling Algorithm,

Sample Input - In the Input panel the following steps need to be done,

- **Sample Mode** should be **selected**.
- Select the Peak Cell Rate (PCR). Its value ranges from **1** to **10**.
- Select the Cell Delay Variation Tolerance (CDVT). Its value ranges from **0.1** to **0.9**.
- Enter in the Actual Arrival Time.
- Then **Run** button need to be **clicked**. **Refresh** button can be used if new Inputs have to be given.

Output - The following steps are undergone internally,

- A graph with the following information is obtained,
 - Cell Confirmation,
 - Actual Arrival Time,
 - Expected Time (TAT)
 - Whether the Cell has been discarded or successfully received. This discarding or receiving of the cell depends on the Actual Arrival Time entered. Go through the 3 rules given in this program.
- Once the sample experiment is done, then **Refresh** button can be **clicked** to create New Samples.

Address Mask

Programming Guidelines

This section guides the user to link his/her own code for Address mask to NetSim.

Pre - Conditions

The user program should read the input from text file named '**Input**' with extension txt.

The user program after executing the concept should write the required output to a file named '**Output**' with extension txt.

Note: The temporary directory is navigated through the following step.

Run → Type "%temp%" → NetSim → "Input.txt" and "Output.txt"

General Program Flow

The program begins with the *Reading of the Inputs* from the input file **Input.txt**.

Executing the required concept and,

The results of the program should be written into the output file **Output.txt**.

File Format

Input File	Output File
<p>This gives us the contents of the 'Input.Txt' from which the user has to get the values for his program</p> <p>The Input File format</p> <p>IP Address></p> <p>Sample Input text Format</p> <p>Let us consider how a given input (Data file and error index) is stored in the text and read.</p> <p>IP_Address=192.169.0.150</p> <p>Prefix_Value=1</p>	<p>The Output File format</p> <p>Binary value of IP Address></p> <p>Prefix value>Suffix value></p> <p>Binary value of Address mask></p> <p>Decimal value of Address mask>.</p> <p>Sample Output text Format</p> <p>11000000 10101001 00000000 10010110 ></p> <p>1>31></p> <p>10000000 00000000 00000000</p> <p>0000000>128>0>0>0></p>

Interface Source Code

Interface source code written in C is given using this the user can write only thefnAddressMAsk () function using the variables already declared. To view the interface source code, go to

NetSim Installation path / src / Programming/ AddressMask.c

To find NetSim's Installation path right click NetSim icon and select

- Open file location in Windows 7
- Open file location in Windows Vista
- Properties --> find target in Windows XP.

Sample Scenarios:

Objective - To understand the concept of finding Address mask through programming.

Sample Inputs - In the Input panel the following steps need to be done,

- **Sample Mode** should be **selected**.
- Select the IP Address
- Select the Prefix value
- Then **Run** button need to be **clicked**. **Refresh** button can be used if new Inputs have to be given.

Output - The following steps are done internally,

- Binary value of IP Address is obtained.
- Prefix, Suffix values are obtained
- Binary value of Address Mask is obtained
- Decimal value of Address Mask is obtained
- Once the sample experiment is done, then **Refresh** button can be **clicked** to create New Samples.

Binary Conversion

Programming Guidelines

This section guides the user to link his/her own code for Binary conversion to NetSim.

Pre - Conditions

The user program should read the input from text file named '**Input**' with extension txt.

The user program after executing the concept should write the required output to a file named '**Output**' with extension txt.

Note: The temporary directory is navigated through the following step.

Run → Type "%temp%" → NetSim → "Input.txt" and "Output.txt"

General Program Flow

The program begins with the *Reading of the Inputs* from the input file **Input.txt**.

Executing the required concept and,

The results of the program should be written into the output file **Output.txt**.

File Format

Input File Format	Output File Format
This gives us the contents of the 'Input. Txt' from which the user has to get the values for his program The Input File format IP Address>	The Output File format First Byte value>128>Quotient of First Byte/128>Remainder of First Byte/128> Previous remainder>64>Quotient of Previous remainder/64>Remainder of Previous remainder /64> Previous remainder>32>Quotient of Previous remainder/32>Remainder of Previous remainder /32> Previous remainder>16>Quotient of Previous remainder/16>Remainder of Previous remainder /16> Previous remainder>8>Quotient of Previous
Sample Input text Format IP_Address=192.168.0.100	

remainder/8>Remainder of Previous
remainder /8>
Previous remainder>4>Quotient of Previous
remainder/4>Remainder of Previous
remainder /4>
Previous remainder>2>Quotient of Previous
remainder/2>Remainder of Previous
remainder /2>
Previous remainder>1>Quotient of Previous
remainder/1>Remainder of Previous
remainder /1>.
.same procedure for all bytes
Binary value>

Sample Output text Format

192>128>1>64>
64>64>1>0>
0>32>0>0>
0>16>0>0>
0>8>0>0>
0>4>0>0>
0>2>0>0>
0>1>0>0>
168>128>1>40>
40>64>0>40>
40>32>1>8>
8>16>0>8>
8>8>1>0>
0>4>0>0>
0>2>0>0>
0>1>0>0>
0>128>0>0>
0>64>0>0>
0>32>0>0>

	0>16>0>0> 0>8>0>0> 0>4>0>0> 0>2>0>0> 0>1>0>0> 100>128>0>100> 100>64>1>36> 36>32>1>4> 4>16>0>4> 4>8>0>4> 4>4>1>0> 0>2>0>0> 0>1>0>0> 11000000 10101000 00000000 01100100 >
--	---

Interface Source Code

Interface source code written in C is given using this the user can write only the fnBinaryConversion () function using the variables already declared. To view the interface source code, go to

NetSim Installation path / src / Programming/ BinaryConversion.c

To find NetSim's Installation path right click NetSim icon and select

- Open file location in Windows 7
- Open file location in Windows Vista
- Properties --> find target in Windows XP.

Sample Scenarios:

Objective - To understand the concept of Binary conversion through programming.

Sample Inputs - In the Input panel the following steps need to be done,

- **Sample Mode** should be selected.
- Select the IP Address

- Then **Run** button need to be **clicked**. **Refresh** button can be used if new Inputs have to be given.

Output - The following steps are done internally,

- Each bit is obtained one by one.
- Binary value of IP Address is obtained.
- Once the sample experiment is done, then **Refresh** button can be **clicked** to create New Samples.

Classless InterDomain Routing

Programming Guidelines

This section guides the user to link his/her own code for Classless Inter Domain Routing to NetSim.

Pre - Conditions

The user program should read the inputted scenario from text file named '**Input**' with extension txt.

The user program after executing the concept should write the required output to a file named '**Output**' with extension txt.

The path of the input file and the output file can be viewed on clicking the Button "Path" in NetSim.

Note:

The temporary directory is navigated through the following step.

Run → Type "%temp%" → NetSim → "Input.txt" and "Output.txt"

General Program Flow

The program begins with the *Reading of the Inputs* from the input file **Input.txt**.

Executing the required concept and,

The results of the program should be written into the output file **Output.txt**.

File Format

Input File Format	Output File Format
<p>This gives the contents of the 'Input. Txt' from which the user has to get the values.</p> <p>The Input File contains:</p> <p>Starting_IP_Address=150.0.0.0 No_of_Networks=3 No_of_Hosts_in_Network_1=512 No_of_Hosts_in_Network_2=1024 No_of_Hosts_in_Network_3=2048 Host_IP_Address=150.0.1.2</p>	<p>The Output file contains the First address, Last address, Mask, Value and Network Address of all the networks.</p> <p>First Address: The Starting Address of the Network</p> <p>Last Address: The last address in the Network.</p> <p>Mask: The Mask address (Mask should be 4 byte format).</p> <p>Number of Node: The number of nodes in the network.</p> <p>Value: The value will be either 0 or 1.</p> <p>0 - The next line has the network address to which the given host belongs.</p> <p>1 - Means that the host does not belong to any network.</p> <p>Network Address: If the Value is 1 then the network to which the host belongs is given.</p> <p>Note: Each data will have a separate line and each line will end with ">"</p> <p>A Sample Output File Format</p> <p>150.0.0.> 150.0.1.255> 255.255.254.0> 512> 150.0.4.0> 150.0.7.255> 255.255.252.0></p>

	<pre> 1024> 150.0.8.0> 150.0.15.255> 255.255.248.0> 2048> 0> 150.0.0.0> </pre> <p>For each network there will be first address, last address, mask and number of hosts. Since the input is 3 networks, there are 3 set of outputs.</p> <p>The last two lines give the details about whether the host is present in any of the network.</p>
--	---

Interface Source Code

Interface source code written in C is given using this the user can write only the CIDR inside the function fnCIDR () using the variables already declared. To view the interface source code, go to

NetSim Installation path / src / Programming/ CIDR.c

To find NetSim's Installation path right click NetSim icon and select

- Open file location in Windows 7
- Open file location in Windows Vista
- Properties --> find target in Windows XP.

Sample Scenarios:

Objective - To Implement Classless InterDomain Routing (CIDR).

How to Proceed? - The objective can be executed in NetSim using the programming exercise available, under programming user has to select Classless InterDomain Routing,

Sample Input - By using the **Input Panel** that is available on the left hand side a **Sample Scenario** is created. The Steps involved in creating are as follows,

- **Sample Mode** should be selected.
- Select the **Starting IP Address** from the given range.

- Select the **No. of Networks** from the given range. Maximum of **6** networks can be selected.
- Select the **No. of Hosts** from the given range.
- Click on **Add** button to add the **No. of Hosts** onto the Hosts Field. The use of **Add** button depends on the **No. of Networks** selected. If a new No. of Hosts has to be added then remove button can be used.
- Select the **Host IP Address**.
- Then **Run** button need to be **clicked**. **Refresh** button can be used if new Inputs have to be given.

Output - The following steps are undergone internally,

- Based on the Starting IP Address a table is obtained.
- There are 4 columns available in the output table, i.e. First Address, Last Address, CIDR Mask and No. of Hosts.
- The First Address of the first network would be nothing but the Starting IP Address which is selected. No. of rows in the table depends on the No. of Networks selected.
- The Last Address depends on the No. of Host selected.
- CIDR Mask is obtained internally by using the following formula,

$$\text{CIDR Mask} = 32 - (\log(\text{No. of Host}) / \log(2))$$

- Once the sample experiment is done, then **Refresh** button can be **clicked** to create New Samples.

Network Address

Programming Guidelines

This section guides the user to link his/her own code for Network Addresses to NetSim.

Pre - Conditions

The user program should read the input from text file named '**Input**' with extension txt.

The user program after executing the concept should write the required output to a file named '**Output**' with extension txt.

Note: The temporary directory is navigated through the following step.

Run → Type "%temp%" → NetSim → "Input.txt" and "Output.txt"

General Program Flow

The program begins with the *Reading of the Inputs* from the input file **Input.txt**.

Executing the required concept and,

The results of the program should be written into the output file **Output.txt**.

File Format

Input File	Output File
IP_Address=192.168.0.140 Prefix_Value=1	<p>The Output File format Binary value of IP Address> Prefix value> Binary value of Address mask> Binary value of Network Address> Decimal value of Network Address>.</p> <p>Sample Output text Format</p> <p>11000000 10101000 00000000 10001100 >1> 10000000 00000000 00000000 00000000> 10000000 00000000 00000000 00000000> 128>0>0>0></p>

Interface Source Code

Interface source code written in C is given using this the user can write only the fnNetworkAddress () function using the variables already declared. To view the interface source code, go to

NetSim Installation path / src / Programming/ NetworkAddress.c

To find NetSim's Installation path right click NetSim icon and select

- Open file location in Windows 7
- Open file location in Windows Vista
- Properties --> find target in Windows XP.

Sample Scenarios:

Objective - To understand the concept of finding Network Address through programming.

Sample Inputs - In the Input panel the following steps need to be done,

- **Sample Mode** should be **selected**.
- Select the IP Address
- Select the Prefix value
- Then **Run** button need to be **clicked**. **Refresh** button can be used if new Inputs have to be given.

Output - The following steps are done internally,

- Binary value of IP Address is obtained.
- Binary value of Address Mask is obtained
- Binary value of Network Address is obtained
- Decimal value of Network Address is obtained
- Once the sample experiment is done, then **Refresh** button can be **clicked** to create New Samples.

Special Addresses

Programming Guidelines

This section guides the user to link his/her own code for Special Addresses to NetSim.

Pre - Conditions

The user program should read the input scenario from text file named '**Input**' with extension txt.

The user program after executing the concept should write the required output to a file named '**Output**' with extension txt.

Note: The temporary directory is navigated through the following step.

Run → Type "%temp%" → NetSim → "Input.txt" and "Output.txt"

General Program Flow

The program begins with the *Reading of the Inputs* from the input file **Input.txt**.

Executing the required concept and,

The results of the program should be written into the output file **Output.txt**.

File Format

Input File	Output File
IP_Address=192.168.0.160 Prefix_Value=5	The Output File format Binary value of IP Address> 32>Prefix value>Suffix value> Binary value of Prefix Part> Binary value of Suffix Part> Condition number in the table>condition> Type of address> Sample Output text Format 11000000 10101000 00000000 10100000 > 32>5>27> 11000>000 10101000 00000000 10100000> It doesn't Satisfy the four conditions> Not a special address>

Interface Source Code

Interface source code written in C is given using this the user can write only the fnSpecialAddress () function using the variables already declared. To view the interface source code, go to

NetSim Installation path / src / Programming/ HammingCode.c

To find NetSim's Installation path right click NetSim icon and select

- Open file location in Windows 7
- Open file location in Windows Vista
- Properties --> find target in Windows XP.

Sample Scenarios:

Objective - To understand the concept of Special Addresses through programming.

Sample Inputs - In the Input panel the following steps need to be done,

- **Sample Mode** should be **selected**.
- Select the IP Address in slash notation
- Then **Run** button need to be **clicked**. **Refresh** button can be used if new Inputs have to be given.

Output - The following steps are done internally,

- Binary value of IP Address is obtained.
- Binary value of Address Mask is obtained
- Prefix, Suffix values are obtained
- Binary value of Prefix part and Suffix part are obtained
- Type of address is obtained
- Once the sample experiment is done, then **Refresh** button can be **clicked** to create New Samples

Subnetting

Programming Guidelines

This section guides the user to link his/her own code for Sub-netting to NetSim.

Pre - Conditions

The user program should read the inputted scenario from text file named '**Input**' with extension txt.

The user program after executing the concept should write the required output to a file named '**Output**' with extension txt.

The path of the input file and the output file can be viewed on clicking the Button "Path" in NetSim.

Note: The temporary directory is navigated through the following step.

Run → Type "%temp%" → NetSim → "Input.txt" and "Output.txt"

General Program Flow

The program begins with the *Reading of the Inputs* from the input file **Input.txt**.

Executing the required concept and,

The results of the program should be written into the output file **Output.txt**.

File Format

Input File Format	Output File Format
Class=B Network_Address=128.0.0.0 No_of_Subnets=2 Host_IP_Address=128.0.0.89	The Output file contains the Default mask (in decimal), Network portion of Default mask (in binary), Host portion of Default mask (in binary), Subnet mask bit value, Network portion of Subnet mask (in binary), Host portion of Subnet mask (in binary), Subnet mask (in decimal), number of zero in host portion, number of host in each subnet, Host address, subnet number of the given host, subnet address of the given host and Subnet address, Starting address, Ending address, Broadcast address of each subnet. The Output File format

	<p>Default mask (in decimal)></p> <p>Network portion of Default mask (in binary)>Host portion of Default mask (in binary)></p> <p>Subnet mask bit value></p> <p>Subnet mask (in decimal)></p> <p>Network portion of Subnet mask (in binary)>Host portion of Subnet mask (in binary)></p> <p>Number of zero in host portion,>Number of host in each subnet></p> <p>Host address>Subnet number of the given host >Subnet address of the given host></p> <p>Subnet address>Starting address>Ending address>Broadcast address ></p> <ul style="list-style-type: none"> . . .until number of subnet is reached <p>Sample Output text Format</p> <p>255.255.0.0</p> <p>11111111 11111111>00000000 00000000></p> <p>1></p> <p>255.255.128.0></p> <p>11111111 11111111 1>00000000 00000000></p> <p>15>32766></p> <p>128.0.0.89>1>128.0.0.0></p> <p>1>128.0.0.0>128.0.0.1>128.0.127.254>128.0.127.255></p> <p>2>128.0.128.0>128.0.128.1>128.0.255.254>128.0.255.255></p>
--	---

Interface Source Code

Interface source code written in C is given using this the user can write only the Leaky bucket algorithm inside the function fnSubnet () using the variables already declared. To view the interface source code, go to

NetSim Installation path / src / Programming/ Subnet.c

To find NetSim's Installation path right click NetSim icon and select

- Open file location in Windows 7
- Open file location in Windows Vista
- Properties --> find target in Windows XP.

Sample Scenarios:

Objective - To understand the concept of Subnetting through programming.

How to Proceed? - The objective can be executed in NetSim using the programming exercise available, under programming user has to select Subnetting,

Sample Inputs - In the Input panel the following steps need to be done,

- **Sample Mode** should be selected.
- Select the Class Name
- Select the Network Address from the given list.
- Select the Number of Subnets from the given list.
- Select the Host IP Address.
- Then **Run** button need to be **clicked**. **Refresh** button can be used if new Inputs have to be given.

Output - The following steps are undergone internally,

- Subnet mask value is obtained.
- Number of host in each subnet is obtained
- Subnet address, Starting address, Ending address and Broadcast address are obtained.
- Subnet address of the given Host IP address is obtained.
- Once the sample experiment is done, then **Refresh** button can be **clicked** to create New Samples.

EUI-64 Interface Identifier

Programming Guidelines

This section guides the user to link his/her own code for EUI 64 Interface Identifier to NetSim.

Pre - Conditions

The user program should read the inputted scenario from text file named '**Input**' with extension txt.

The user program after executing the concept should write the required output to a file named '**Output**' with extension txt.

Note: The temporary directory is navigated through the following step.

Run → Type "%temp%" → NetSim → "Input.txt" and "Output.txt"

General Program Flow

The program begins with the *Reading of the Inputs* from the input file **Input.txt**.

Executing the required concept and,

The results of the program should be written into the output file **Output.txt**.

File Format

Input File	Output File
IPV6_Network_Id=2000:FE21:5931:72 C3 MAC_Address=11-11-FF-FF-12-34	The Output File format MAC Address> First part of MAC Address>Second part of MAC Address FF-EE appended Address> First byte value>Binary value of First byte> 7 th bit value>Complement value of 7 th bit Complemented binary value of First byte>Hexa decimal value of complemented binary value> Interface Id value> Interface Id in colon notation> IPV6 prefix value> IPV6 Address>

	Sample Output text Format 11-11-FF-FF-12-34> 11-11-FF>FF-12-34> 11-11-FF-FF-FE-FF-12-34> 11>00010001> 0>1> 00010011>13> 13-11-FF-FF-FE-FF-12- 34>1311:FFFF:FEFF:1234> 2000:FE21:5931:72C3> 2000:FE21:5931:72C3:1311:FFFF:FEFF:1234>
--	---

Interface Source Code

Interface source code written in C is given using this the user can write only the fnEUI64() function using the variables already declared. To view the interface source code, go to

NetSim Installation path / src / Programming/ IPV6EUI64.c

To find NetSim's Installation path right click NetSim icon and select

- Open file location in Windows 7
- Open file location in Windows Vista
- Properties --> find target in Windows XP.

Sample Scenarios:

Objective - To understand the concept of EUI 64 Interface Identifier through programming.

How to Proceed? - The objective can be executed in NetSim using the programming exercise available, under programming user has to select IPV6 Addressing → EUI 64 Interface Identifier,

Sample Inputs - In the Input panel the following steps need to be done,

- **Sample Mode** should be selected.
- Select the IPV6 Network Id (IPV6 Prefix)
- Enter the MAC Address.
- Then **Run** button need to be clicked. **Refresh** button can be used if new Inputs have to be given.

Output - The following steps are undergone internally,

- MAC Address is divided into two parts
- FF-FE value is appended between the two parts
- Interface Identifier is found by complementing 7th bit of first byte
- IPV6 Network Id (IPV6 prefix) and Interface Identifier are combined to produce IPV6 address
- Once the sample experiment is done, then **Refresh** button can be **clicked** to create New Samples.

IPV6 Host Addresses

Programming Guidelines

This section guides the user to link his/her own code for IPV6 Host Addresses to NetSim.

Pre - Conditions

The user program should read the inputted scenario from text file named '**Input**' with extension txt.

The user program after executing the concept should write the required output to a file named '**Output**' with extension txt.

Note: The temporary directory is navigated through the following step.

Run → Type "%temp%" → NetSim → "Input.txt" and "Output.txt"

General Program Flow

The program begins with the *Reading of the Inputs* from the input file **Input.txt**.

Executing the required concept and,

The results of the program should be written into the output file **Output.txt**.

File Format

Interface Source Code

Interface source code written in C is given using this the user can write only the fnHostAddresses () function using the variables already declared. To view the interface source code, go to

NetSim Installation path / src / Programming/ IPV6HostAddress.c

To find NetSim's Installation path right click NetSim icon and select

- Open file location in Windows 7
- Open file location in Windows Vista
- Properties --> find target in Windows XP.

Sample Scenarios:

Objective - To understand the concept of IPV6 Host Addresses through programming.

How to Proceed? - The objective can be executed in NetSim using the programming exercise available, under programming user has to select IPV6 Addressing →IPV6 Host Addresses,

Sample Inputs - In the Input panel the following steps need to be done,

- **Sample Mode** should be **selected**.
- Enter the IPV6 Address.
- Select the Prefix length.
- Then **Run** button need to be **clicked**. **Refresh** button can be used if new Inputs have to be given.

Output - The following steps are under gone internally,

- IPV6 is separated into two parts: Network Id bits (prefix) and Host Id bits (suffix)
- Starting address of the network is found by replacing each bits of suffix part with zero
- Ending address of the network is found by replacing each bits of suffix part with one

- Once the sample experiment is done, then **Refresh** button can be **clicked** to create New Samples.

IPV6 Subnetting

Programming Guidelines

This section guides the user to link his/her own code for IPV6 Subnetting to NetSim.

Pre - Conditions

The user program should read the inputted scenario from text file named '**Input**' with extension txt.

The user program after executing the concept should write the required output to a file named '**Output**' with extension txt.

Note: The temporary directory is navigated through the following step.

Run → Type "%temp%" → NetSim → "Input.txt" and "Output.txt"

General Program Flow

The program begins with the *Reading of the Inputs* from the input file **Input.txt**.

Executing the required concept and,

The results of the program should be written into the output file **Output.txt**.

File Format

Input File Format	Output File Format
IPV6_Address=1111:2222:3 333:4444:AAAA:BBBB:CC CC:DDDD Prefix_Length=12 Number_of_Subnets=2 Number_of_Level=2 Subnets_in_Level1=2 Subnets_in_Level2=4	The Output File format Binary value of IPV6 Address> Prefix part of IPV6 Address>Suffix part of IPV6 Address> Number of subnets>number of subnet msk bit> Prefix part of IPV6 Address>Subnet Id part of IPV6 address>Suffix part of IPV6 Address> Prefix part of IPV6 Address>Subnet Id part of IPV6 address>Suffix part of IPV6 Address> Prefix part of Level 1 's first subnet Address> Subnet Id part of Level 1 's first subnet Address> Suffix part of Level 1 's first subnet Address> Hexa decimal value of Level 1 's first

	<p>subnet address>prefix length of Level 1 ‘s first subnet></p> <p>.</p> <p>.</p> <p>until number of subnet reached in the first level</p> <p>First level’s subnet number>Prefix part of Level 2 ‘s first subnet Address> Subnet Id part of Level 2 ‘s first subnet Address> Suffix part of Level 2 ‘s first subnet Address> Hexa decimal value of Level 2 ‘s first subnet address>prefix length of Level 2 ‘s first subnet></p> <p>.</p> <p>.</p> <p>until the number of subnets reached in the second level of first level’s subnet</p> <p>.</p> <p>.</p> <p>until the number of subnets reached in the first level</p> <p>Sample Output text Format</p> <pre>1111:2222:3333:4444:AAAA:BBBB:CCCC:DDDD> 000100010001100100100010001000110011001100110 100010001000100101010101010101110111011101111 0011001100110011011101110111011101> 000100010001>000100100010001000100011001100110011 0100010001000100101010101010101010111011101110111 10011001100110011011101110111011101>2>1> 000100010001>0>001001000100010001000110011001100 110100010001000100101010101010101011101110111011101 1110011001100110011011101110111011101> 000100010001>0>001001000100010001000110011001100 11010001000100010010101010101010101011101110111011101 1110011001100110011011101110111011101>1111:2222:3333:4 444:AAAA:BBBB:CCCC:DDDD>13> 000100010001>1>001001000100010001000110011001100 110100010001000100101010101010101011101110111011101 11100110011001100110111011101110111011101>1119:2222:3333:4 444:AAAA:BBBB:CCCC:DDDD>13> 0>1111:2222:3333:4444:AAAA:BBBB:CCCC:DDDD>15> 0>1113:2222:3333:4444:AAAA:BBBB:CCCC:DDDD>15> 0>1115:2222:3333:4444:AAAA:BBBB:CCCC:DDDD>15> 0>1117:2222:3333:4444:AAAA:BBBB:CCCC:DDDD>15> 1>1119:2222:3333:4444:AAAA:BBBB:CCCC:DDDD>15> 1>111B:2222:3333:4444:AAAA:BBBB:CCCC:DDDD>15> 1>111D:2222:3333:4444:AAAA:BBBB:CCCC:DDDD>15> 1>111F:2222:3333:4444:AAAA:BBBB:CCCC:DDDD>15></pre>
--	--

Interface Source Code

Interface source code written in C is given using this the user can write only the fnIPV6Subnetting () function using the variables already declared. To view the interface source code, go to

NetSim Installation path / src / Programming/ IPV6Subnetting.c

To find NetSim's Installation path right click NetSim icon and select

- Open file location in Windows 7
- Open file location in Windows Vista
- Properties --> find target in Windows XP.

Sample Scenarios:

Objective - To understand the concept of IPV6 Subnetting through programming.

How to Proceed? - The objective can be executed in NetSim using the programming exercise available, under programming user has to select IPV6 Addressing →IPV6 Subnetting,

Sample Inputs - In the Input panel the following steps need to be done,

- **Sample Mode** should be **selected**.
- Enter the IPV6 Address.
- Select the Prefix length.
- Select the number of subnets value
- Click the Add value button to add the levels
- Then **Run** button need to be **clicked**. **Refresh** button can be used if new Inputs have to be given.

Output - The following steps are undergone internally,

- IPV6 is separated into two parts: prefix and suffix
- Number of subnet mask id is calculated
- Then Subnet Id portion is derived from suffix part
- Using the subnet id portion, subnets are calculated

- Like that, subnets in the all levels are calculates by using the above steps
- Once the sample experiment is done, then **Refresh** button can be **clicked** to create New Samples.

Leaky Bucket Algorithm

Programming Guidelines

This section guides the user to link his/her own code for Leaky Bucket Algorithm to NetSim.

Pre - Conditions

The user program should read the inputted scenario from text file named '**Input**' with extension txt.

The user program after executing the concept should write the required output to a file named '**Output**' with extension txt.

The path of the input file and the output file can be viewed on clicking the Button "Path" in NetSim.

Note: The temporary directory is navigated through the following step.

Run → Type "%temp%" → NetSim → "Input.txt" and "Output.txt"

General Program Flow

The program begins with the *Reading of the Inputs* from the input file **Input.txt**.

Executing the required concept and,

The results of the program should be written into the output file **Output.txt**.

File Format

Input File	Output File
Output_Capacity=100 Buffer_Size=100 Input_Capacity=100,100,500,1 00,100,100,100,100,100, 	'Output.txt' file contains three lines of data. The data format in the output file should be as follows: Output rate at 1 st second >Output rate at 2 nd second>.....> Discard rate at 1 st second >Discard rate at 2 nd second>.....> Total number of seconds taken > The data should be stored in the file with a delimiter “>” in between. <u>Sample File data</u> 100>100>100>100>100>100>100>100>100> 100> 0>0>300>0>0>0>0>0>0>0> 11>

Interface Source Code

Interface source code written in C is given using this the user can write only the Leaky bucket algorithm inside the function fnLBA() using the variables already declared. To view the interface source code, go to

NetSim Installation path / src / Programming/ LBA.c

To find NetSim's Installation path right click NetSim icon and select

- Open file location in Windows 7
- Open file location in Windows Vista
- Properties --> find target in Windows XP.

Sample Scenarios:

Objective - To understand the concept of Leaky Bucket Algorithm(LBA) through programming.

How to Proceed? - The objective can be executed in NetSim using the programming exercise available, under programming menu user has to select Leaky Bucket Algorithm.

Sample Inputs - In the Input panel the following steps need to be done,

- **Sample Mode** should be selected.
- Select the Output Capacity from the given list of values. The value ranges from **100** to **1000**.
- Select the Size of Buffer from the given list of values. The value ranges from **100** to **1000**.
- Enter in the Input Capacity in the fields provided. The range that can be entered is from **100** to **1000**. These values will be plotted in the graph on the right hand side panel.
- Then **Run** button need to be **clicked**. **Refresh** button can be used if new Inputs have to be given.

Output - The following steps are undergone internally,

- Once the **Run** button is clicked a graph with Output and Discard Rate is obtained.
- The graph explains in detail as in what is the Output and Discard Packet Rate for the given inputs.
- Once the sample experiment is done, then **Refresh** button can be **clicked** to create New Samples.

Multi Level Multi Access

Programming Guidelines

This section guides the user to link his/her own code for Multi-Level Multi Access Protocol to NetSim.

Pre - Condition

User written program should read the value from the '**Input.txt**' in the temporary directory which is having input from the GUI at runtime

The output should be stored in '**Output.txt**' in the temporary directory for display.

User written program should return an integer value.

Note: The temporary directory is navigated through the following step.

Run → Type "%temp%" → NetSim → "Input.txt" and "Output.txt"

General Program Flow

The program begins with the *Reading of the Inputs* from the input file **Input.txt**.

Executing the required concept and, The results of the program should be written into the output file **Output.txt**.

Note: The naming of the input and the output file must be same as the text displayed in the Methodology screen

File Format

Input File	Output File
Number_of_Nodes=2 Node1_Address=250 Node2_Address=500	<p>'Output.txt' file has the decade's number, decade's type, and actual decades, node addresses</p> <p>Decade No>Decade Type>Decade>address1>address2>...</p> <p>Sample Output:</p> <p>0>0>0000100100> 1>1>0000000001> 2>2>0000000001>500> 3>1>0000100000> 4>2>0000000001>250></p>

Interface Source Code

Interface source code written in C is given using this the user can write only the Multi-Level Multi Access Protocol inside the function fnMLMA () using the variables already declared. To view the interface source code, go to

NetSim Installation path / src / Programming/ MLMA.c

To find NetSim's Installation path right click NetSim icon and select

- Open file location in Windows 7
- Open file location in Windows Vista
- Properties --> find target in Windows XP.

Sample Scenarios:

Objective - To study the working of Multi Level Multi Access (MLMA).

How to Proceed? - The objective can be executed in NetSim using the programming exercise available, under programming user has to select Multi-Level Multi -Access Protocol.

Sample Inputs - In the Input panel the following steps need to be done,

- **Sample Mode** should be **selected**.
- Number of Nodes need to be **selected**. The value ranges from **2** to **15**.
- Node Address has to be entered and added onto the Node Address field.
The Number of Nodes selected and the Number of Addresses added into the Address field should match.
- Then **Run** button need to be **clicked**. **Refresh** button can be used if new Inputs have to be given.

Output - The following steps are undergone internally,

- An Output table is formed.
- On the right hand panel an animation would play. This shows that higher the address given to the node higher the priority that node gets. Hence the node with the higher address would transmit the data first.

- Once the sample experiment is done, then **Refresh** button can be **clicked** to create New Samples.

Code Division Multiple Access

Programming Guidelines

This section guides the user to link his/her own code for Code Division Multiple Access to NetSim.

Pre - Conditions

The user program should read the input scenario from text file named '**Input**' with extension txt which is in **Temporary Directory**.

The user program after executing the concept should write the required output to a file named '**Output**' with extension txt in **Temporary Directory**.

Note: The Temporary Directory is navigated through the following step.

Run → Type "%temp%" → NetSim → "Input.txt" and "Output.txt"

General Program Flow

The program begins with the *Reading of the Inputs* from the input file **Input.txt**.

Executing the required concept and

The results of the program should be written into the output file **Output.txt**.

File Format

Input File Format	Output File Format
<p>The “Input.txt” file contains,</p> <p>NumberofTransmittingMobilestations=Value></p> <p>MobileStation=Value>Data=Value>code=Value>Encode=Value>Signal=Value></p> <p>MobileStation=Value>Data=Value>code=Value>Encode=Value>Signal=Value></p> <p>MobileStation=Value>Data=Value>code=Value>Encode=Value>Signal=Value></p> <p>MobileStation=Value></p>	<p>No. of Transmitting MobileStations=Value></p> <p>MobileStation=Value>Data=Value>code=Value>Encode=Value>Signal=Value></p> <p>MobileStation=Value>Data=Value>code=Value>Encode=Value>Signal=Value></p> <p>MobileStation=Value>Data=Value>code=Value>Encode=Value>Signal=Value></p> <p>Interference Pattern=Value></p>

<p>Data=Value>Code=Value></p> <p>Example:</p> <pre>NumberOfTransmittingMobileStations=2> MobileStation=1>Data=10101010> Code=1 1 1 1 1 1 1> MobileStation=2>Data=11110000> Code=1 -1 1 -1 1 -1 1 -1></pre>	<p>MobileStation=Value>code=Value>Decode=Valu e>Data=Value></p> <p>MobileStation=Value>code=Value>Decode=Valu e>Data=Value></p> <p>MobileStation=Value>code=Value>Decode=Valu e>Data=Value></p> <p>Example:</p> <pre>NumberofMobileStations=2> MobileStation=1>Data=1 0 1 0 1 0 1 0 >Code=1 1 1 1 1 1 1 1 >Encode=1 -1 1 -1 1 -1 1 -1 >Signal=1 1 1 1 1 1 1 1 -1 -1 -1 -1 -1 -1 -1 1 1 1 1 1 1 1 1 1 -1 -1 - 1 -1 -1 -1 -1 -1 1 1 1 1 1 1 1 1 1 -1 -1 -1 -1 -1 -1 > MobileStation=2>Data=1 1 1 1 0 0 0 0 >Code=1 -1 1 -1 1 -1 1 -1 >Encode=1 1 1 1 -1 -1 -1 -1 >Signal=1 -1 > InterferencePattern=2 0 2 0 2 0 2 0 0 -2 0 -2 0 -2 0 - 2 2 0 2 0 2 0 2 0 0 -2 0 -2 0 -2 0 -2 0 2 0 2 0 2 0 2 -2 0 -2 0 -2 0 0 2 0 2 0 2 0 2 -2 0 -2 0 -2 0 -2 0 > MobileStation=1>Code=1 1 1 1 1 1 1 >Decode=8 -8 8 -8 8 -8 >Data=1 0 1 0 1 0 1 0 > MobileStation=2>Code=1 1 1 1 1 1 1 >Decode=8 8 8 8 -8 -8 >Data=1 1 1 1 0 0 0 0 ></pre>
---	--

Interface Source Code

Interface source code written in C is given. Using this, the user can write only the function **fnEncode ()** and **fnDecode ()**, using the variables already declared. To view the interface source code, go to

NetSim Installation path / src / Programming/ CDMA.c

To find NetSim's Installation path right click NetSim icon and select

- Open file location in Windows 7
- Open file location in Windows Vista
- Properties --> find target in Windows XP.

Sample Scenarios:

Objective - To study the working of Code Division Multiple Access technique

How to Proceed? - The objective can be executed in NetSim using the programming exercise available. Under programming select Multiple Access Technology → Code Division Multiple Access.

Sample Inputs - In the Input panel the following steps need to be done,

- **Sample Mode** should be selected
- In default, the **Encode** operation will be enabled.
- Choose the **Number of Transmitting Mobile Stations**.
- Enter the **Binary Data** as per the limits.
- Choose the respective **Codes** for each Mobile Station.

Number of Transmitting Mobile Stations → 3 (Maximum of 8 Mobile Stations)

Mobile Station	Binary Data(1Byte)	Code
1	01110111	1 1 1 1 1 1 1 1
2	00110011	1 -1 1 -1 1 -1 1 -1
3	11011101	1 1 -1 -1 1 1 -1 -1

- Then **Run** button needs to be **clicked**. **Refresh** button can be used, if new inputs have to be given.

Output - The Output for the above sample is as follows,

- In **Transmitter side**, the values of each Mobile Station's **Binary Data**, **Code**, **Encode** and **Signal** are shown, if the user clicks the Mobile Station.
- The **Signals** will be broadcast from each Mobile Stations.
- The **Interference Pattern** will be formed in **Base Station** and if the user clicks the Base Station, interference Pattern will be displayed.
- **Decode mode** needs to be **selected**.

- In **Receiver side**, the **Interference Pattern** will be broadcasted to each **Mobile Station** from the **Base Station**.
 - In **Receiver side**, the values of each Mobile Station's **Interference Pattern, Code, Decode and Binary Data** are shown, if the user clicks the Mobile Station.
 - Once the sample experiment is done **Refresh** button can be **clicked** to create new samples.

The Values are showed in Table.

Transmitter Side

Mobile Station	Binary Data(1Byte)	Code	Encode	Signal
1	01110111	1 1 1 1 1 1 1 1	-1, 1, 1, 1, -1, 1, 1, 1	-1 -1 -1 -1 -1 -1 -1 -1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 -1 -1 -1 -1 -1 -1 -1 1
2	00110011	1 -1 1 -1 1 -1 1 - 1	-1, -1, 1, 1, -1, -1, 1, 1	-1 1 -1 1 -1 1 -1 1 -1 1 -1 1 -1 1 1 -1
3	11011101	1 1 -1 -1 1 1 -1 - 1	1, 1, -1, 1, 1, 1, -1, 1	1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 -1 -1 -1 -1 1 1 -1 -1 1 1 1 -1 1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 1 1 1 -1 -1 1 1 -1 -1 -1 -1 1 1 -1 1 -1 1 1 1 1 -1 -1 1 1 -1 -1

In Base Station

Mobile Station	Signal
1	-1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 1
2	-1 1 -1 1 -1 1 -1 1 -1 1 -1 1 -1 1 1 -1 1 1 -1 1 -1 1 -1 1 -1 1 -1 1 -1 1 -1 1 -1 1 -1 1 -1 1 -1 -1 1 -1 1 -1 1 -1 1 -1 1 -1 1 -1 1 -1 1 -1 1 -1 1 -1 1 -1 1 -1 1 -1 1 -1 1 -1 1 -1 1 -1 1 -1 1 -1
3	1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 -1 -1 1 1 -1 -1 1 1 1 1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 1 1 -1 -1 -1 -1 1 1 -1 -1 1 1 1 1 1 1 -1 -1 1 1 -1 -1

Interference Pattern	1 1 -3 -1 -1 1 1 -3 -1 1 3 -1 1 1 3 -1 1 1 -1 3 1 1 -1 3 1 3 1 1 -1 3 1 1 -1 -1 1 -3 -1 -1 1 -3 -1 1 3 -1 1 1 3 -1 1 1 -1 3 1 1 -1 3 1 3 1 1 -1 3 1 1 -1
-----------------------------	--

Receiver Side

Mobile Station	Interference Pattern	Code	Decode	Binary Data(1Byte)
1	1 1 -3 -1 -1 1 1 -3 -1 1 3 -1 1 1 3 -1 1 1 -1 3 1 1 -1 3 1 3 1 1 -1 3 1 1 -1 -1 1 -3 -1 -1 1 -3 -1 1 3 -1 1 1 3 -1 1 1 -1 3 1 1 -1 3 1 3 1 1 -1 3 1 1 -1	1 1 1 1 1 1 1 1	8 8 8 8 -8 8 8 8	01110111
2	1 1 -3 -1 -1 1 1 -3 -1 1 3 -1 1 1 3 -1 1 1 -1 3 1 1 -1 3 1 3 1 1 -1 3 1 1 -1 -1 1 -3 -1 -1 1 -3 -1 1 3 -1 1 1 3 -1 1 1 -1 3 1 1 -1 3 1 3 1 1 -1 3 1 1 -1	1 -1 1 -1 1 -1 1 -1 1	-8 -8 8 8 -8 -8 8 8	00110011
3	1 1 -3 -1 -1 1 1 -3 -1 1 3 -1 1 1 3 -1 1 1 -1 3 1 1 -1 3 1 3 1 1 -1 3 1 1 -1 -1 1 -3 -1 -1 1 -3 -1 1 3 -1 1 1 3 -1 1 1 -1 3 1 1 -1 3 1 3 1 1 -1 3 1 1 -1	1 1 -1 -1 1 1 -1 -1 1	8 8 -8 8 8 8 - 8 8	11011101

Time Division Multiple Access

Programming Guidelines

This section guides the user to link his/her own code for Time Division Multiple Access to NetSim.

Pre - Conditions

The user program should read the input scenario from text file named '**Input**' with extension txt which is in **Temporary Directory**.

The user program after executing the concept should write the required output to a file named '**Output**' with extension txt in **Temporary Directory**.

Note: The Temporary Directory is navigated through the following step.

Run → Type "%temp%" → NetSim → "Input.txt" and "Output.txt"

General Program Flow

The program begins with the *Reading of the Inputs* from the input file **Input.txt**.

Executing the required concept and

The results of the program should be written into the output file **Output.txt**.

File Format

Input File Format	Output File Format
<p>The “Input.txt” file contains,</p> <p>Bandwidth=Value>No. of time slots=Value>Time slot Length=Value>Guard Interval=Value></p> <p>Example:</p> <p>BandWidth=100>TimeSlot=1>TimeSlotLength=100>GuardInterval=0.0></p>	<p>Guard Interval =Value>Bandwidth=Value>start time=Value >end time=Value>nouser=Value></p> <p>channel no=Value>Bandwidth=Value>start time=Value>End time=Value>user=Value></p> <p>Guard Interval =Value>Bandwidth=Value>start time=Value >end time=Value>nouser=Value></p> <p>Guard Interval =Value>Bandwidth=Value>start time=Value >end time=Value>nouser=Value></p> <p>channel no=Value>Bandwidth=Value>start time=Value>End time=Value>user=Value></p> <p>.</p>

	<p>.</p> <p>and so on.</p> <p>The value of the GuardInterval is -1 and the value of the no user is 0</p> <p>Example:</p> <pre>GuardInterval=- 1>Bandwidth=100>StartTime=0.0>EndTime=0.0>nouser= 0> Channelno=1>Bandwidth=100>StartTime=0.0>EndTime= 100.0>user=1> GuardInterval=- 1>Bandwidth=100>StartTime=100.0>EndTime=100.0>no user=0></pre>
--	---

Interface Source Code

Interface source code written in C is given. Using this, the user can write only the function **fnTDMA()**, using the variables already declared. To view the interface source code, go to

NetSim Installation path / src / Programming/ TDMA.c

To find NetSim's Installation path right click NetSim icon and select

- Open file location in Windows 7
- Open file location in Windows Vista
- Properties --> find target in Windows XP.

Sample Scenarios:

Objective - To study the working of Time Division Multiple Access technique.

How to Proceed? - The objective can be executed in NetSim using the programming exercise available, under programming user has to select Multiple Access Technology → Time Division Multiple Access.

Sample Inputs - In the Input panel the following steps need to be done,

- **Sample Mode** should be selected.
- Enter the Value of the **Bandwidth**.

- Enter the Value of the **No. of Time Slots**.
- Enter the Value of the **TimeSlotLength**.
- Enter the Value of the **GuardInterval**

Note:

The Values entered should be within the range.

Bandwidth(kHz)	200
No. of TimeSlots	3
TimeSlotLength(μs)	100
GuardInterval(μs)	10

- Then **Run** button need to be **clicked**. **Refresh** button can be used, if new Inputs have to be given.

Output - The Output for the above sample is as follows,

- Bandwidth is divided into Time slots called channel.
- The first channel is allocated to first mobile station, second channel is allocated to second mobile station and so on.
- The first Mobile Station will access the medium in first channel time slot.
- The Second Mobile Station will access the medium in second channel time slot, and so on.
- During the guard interval, No mobile station will access the medium. This is used to avoid the collision or interference.

Finally the table will be showed.

- Channel no. is obtained.
- Bandwidth value is obtained.
- Start Time and End Time of the Guard Interval is obtained

- Start Time and End Time of the channel is obtained.
- Which Mobile Station is accessing the allocated time slot is obtained.

Channel number	Bandwidth(kHz)	Time(μs)	Mobile Station
0	200	0.0 – 5.0	0
1	200	5.0 – 95.0	1
0	200	95.0 – 100.0	0
0	200	100.0 – 105.0	0
2	200	105.0 – 195.0	2
0	200	195.0 – 200.0	0
0	200	200.0 – 205.0	0
3	200	205.0 – 295.0	3
0	200	295.0 – 300.0	0
0	200	300.0 – 305.0	0
1	200	305.0 – 395.0	1
.			
.			
.			

PC to PC Communication - Socket Programming TCP

Programming Guidelines

This section guides the user to link his/her own code for PC to PC Communication – Socket programming to NetSim.

Pre – Conditions

The user program should read the input scenario from text file named '**Input**' with extension txt.

The user program after executing the concept should write the required output to a file named '**Output**' with extension txt.

General Program Flow

The program begins with the *Reading of the Inputs* from the input file **Input.txt**.

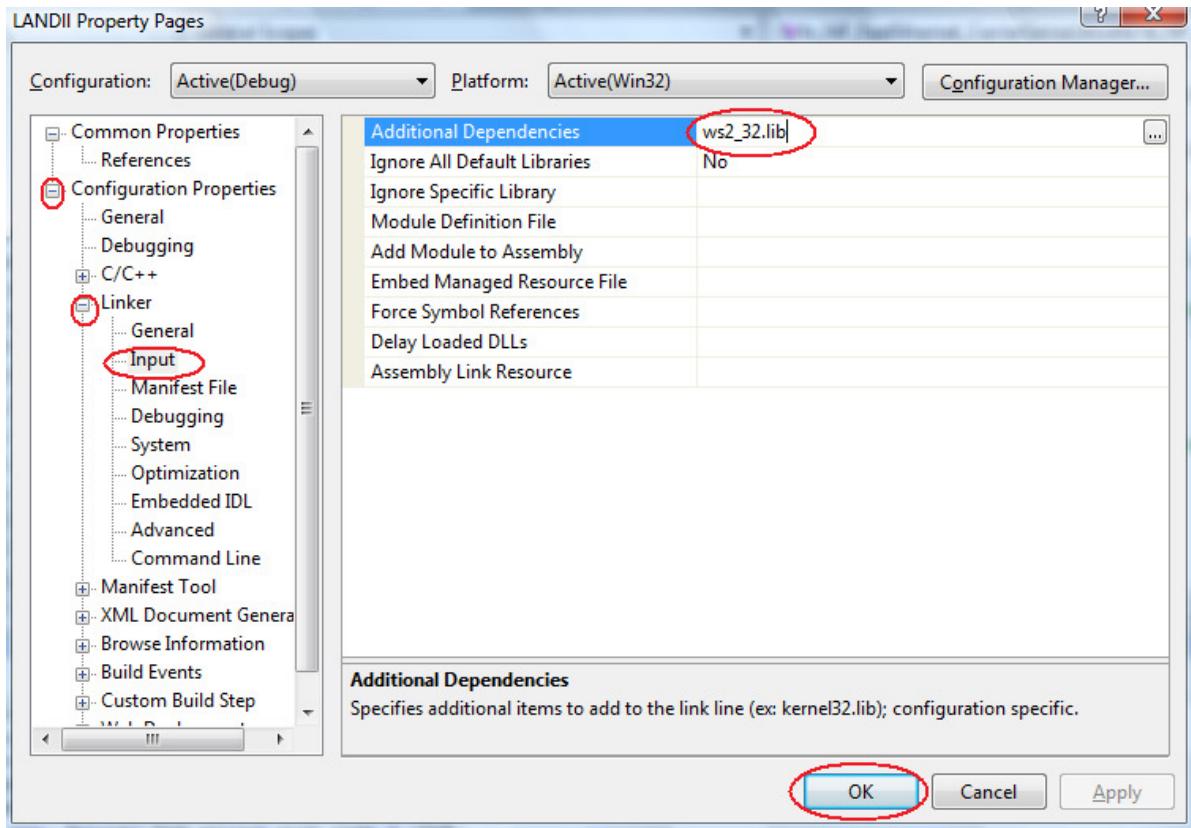
Executing the required concept and

The results of the program should be written into the output file **Output.txt**.

Coding Pre Requisites:

There are certain program prerequisites for the creation of sockets in the Windows-based C complier. They are as follows:

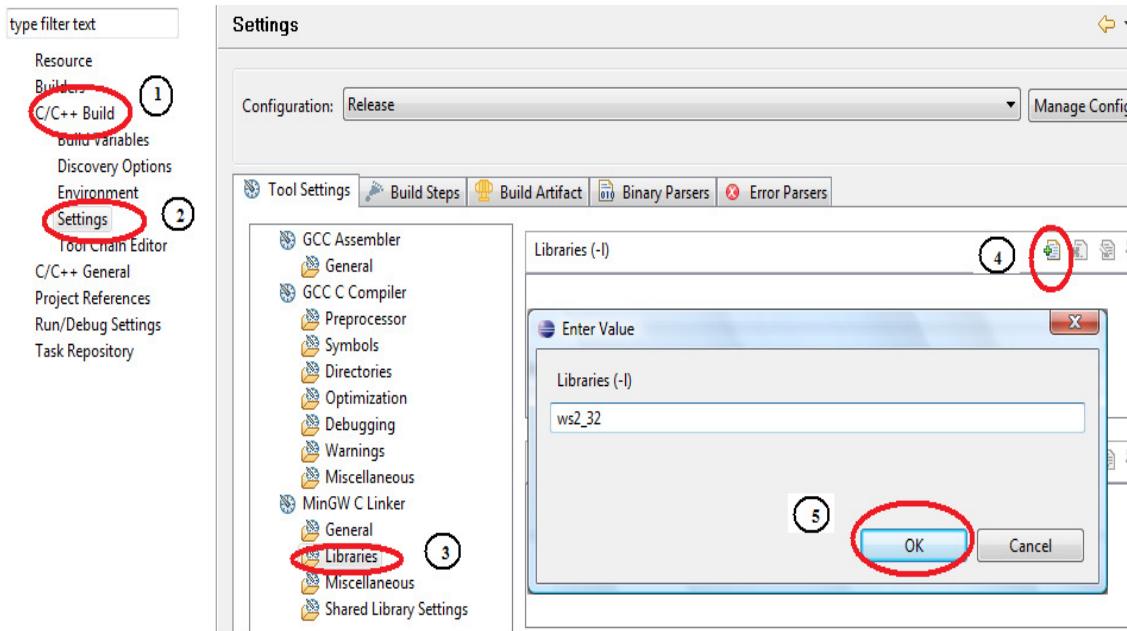
1. The C complier must have **winsock.h** header file in its standard header file folder.
2. In order to use the commands of the winsock.h, there must be **wsock32.lib** library file present in the C complier. It must be linked to your application. For example, if you have a Visual Studio complier 2005 (8.0 versions), the library file linkage must be made in the following way:



- a. Click **Project Menu** in the workspace.
- b. Select **properties** in the pull-down menu or press Alt + F7
- c. Then click **Linker**.
- d. Select **Input**
- e. Type **ws2_32.lib** file in the **Additional Dependencies** text box area.
- e. Click **OK** to confirm the changes in the workspace.

OR

If you have an eclipse complier, the library linkage must be in the following way



- Click **Project Menu** in the workspace.
- Select **properties** in the pull-down menu
- Then click **C/C++ Build**.
- Select **Setting**.
- Select **Libraries** under **MinGW C Linker**
- Press **add** button in the **Libraries (-l)** window
- Type **ws2_32** file in the text box area.
- Click **OK** to confirm the changes in the workspace.

Input File Format	Output File Format
<p>The data format in the input file is as follows,</p> <p>Server Input</p> <p>Protocol=TCP</p> <p>Operation=Server</p> <p>Client</p> <p>Protocol=TCP</p> <p>Operation=Client</p> <p>Destination IP=192.168.0.132</p> <p><Client IP:-Message></p> <p>Ex: 192.168.0.2:- Requesting for the connection</p>	<p>Write the received message</p> <p>Ex:192.168.0.2:- Requesting for the connection</p>

Interface Source Code

Interface source code written in C is given using this the user can write only the TCP Client () function. To view the interface source code, go to

NetSim Installation path / src / Programming/ SocketTCP.c

To find NetSim's Installation path right click NetSim icon and select

- Open file location in Windows 7
- Open file location in Windows Vista
- Properties --> find target in Windows XP.

PC to PC Communication - Socket Programming UDP

Programming Guidelines

This section guides the user to link his/her own code for PC to PC Communication – Socket programming to NetSim.

Pre – Conditions`

The user program should read the input from text file named '**Input**' with extension txt.

The user program after executing the concept should write the required output to a file named '**Output**' with extension txt.

General Program Flow

The program begins with the *Reading of the Inputs* from the input file **Input.txt**.

Executing the required concept and

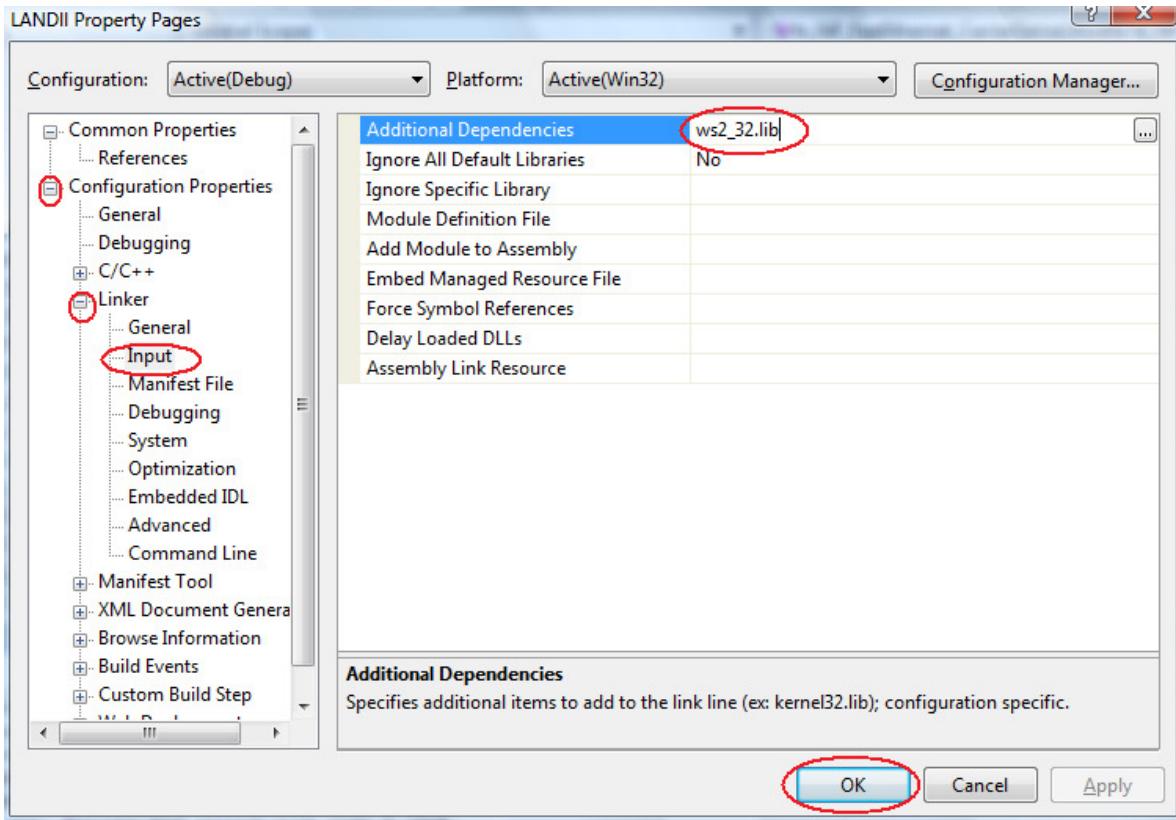
The results of the program should be written into the output file **Output.txt**.

Coding Pre Requisites:

There are certain program prerequisites for the creation of sockets in the Windows-based C complier. They are as follows:

1. The C complier must have **winsock.h** header file in its standard header file folder.
2. In order to use the commands of the winsock.h, there must be **wsock32.lib** library file present in the C complier. It must be linked to your application. For example, if

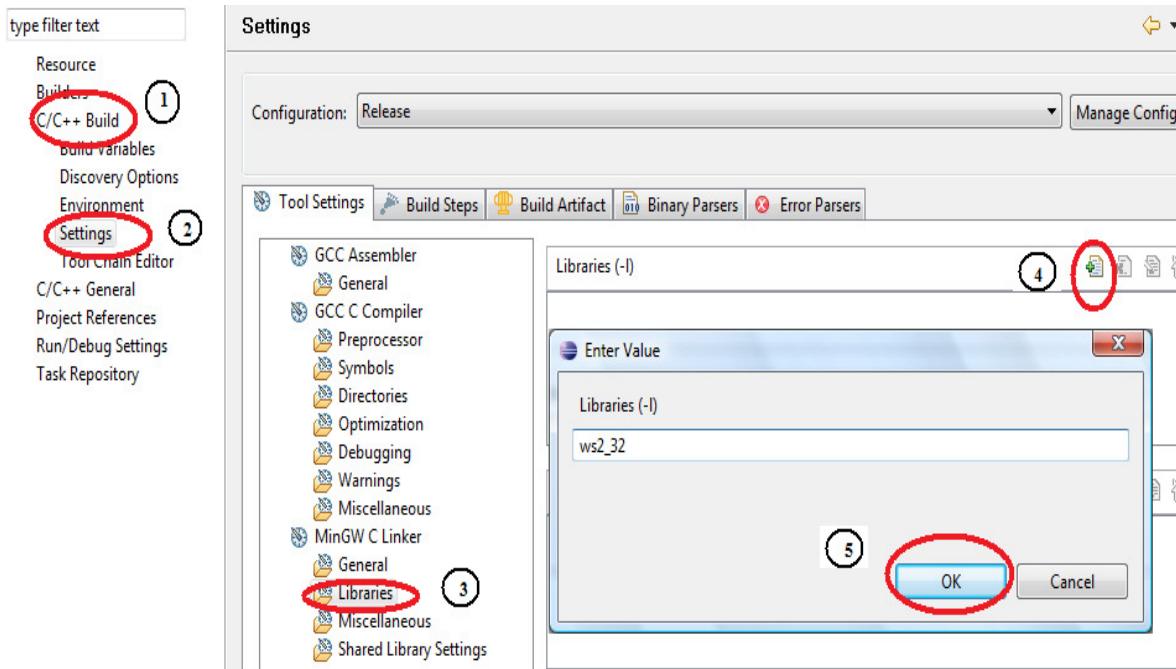
you have a Visual Studio compiler 2005 (8.0 versions), the library file linkage must be made in the following way:



- a. Click **Project Menu** in the workspace.
- b. Select **properties** in the pull-down menu or press Alt + F7
- c. Then click **Linker**.
- d. Select **Input**
- e. Click **OK** to confirm the changes in the workspace.

OR

If you have an eclipse complier, the library linkage must be in the following way



- Click **Project Menu** in the workspace.
- Select **properties** in the pull-down menu
- Then click **C/C++ Build**.
- Select **Setting**.
- Select **Libraries** under **MinGW C Linker**
- Press **add** button in the **Libraries (-l)** window
- Type **ws2_32** file in the text box area.
- Click **OK** to confirm the changes in the workspace.

Input File Format	Output File Format
<p>The data format in the input file is as follows.,</p> <p>Server Input</p> <p>Protocol= UDP</p> <p>Operation=Server</p> <p>Client</p> <p>Protocol=UDP</p> <p>Operation=Client</p> <p>Destination IP=192.168.0.132</p> <p><Client IP:-Message></p> <p>Ex: 192.168.0.2:- Requesting for the connection</p>	<p>Write the received message</p> <p>Ex: 192.168.0.2:- Requesting for the connection</p>

Interface Source Code

Interface source code written in C is given using this the user can write only the UDP () functions. To view the interface source code, go to

NetSim Installation path / src / Programming/ SocketUDP.c

To find NetSim's Installation path right click NetSim icon and select

- Open file location in Windows 7
- Open file location in Windows Vista
- Properties --> find target in Windows XP.

Sample Scenarios:

Objective - Send and Receive the data using PC to PC Communication - Socket Programming

How to Proceed? - Two Systems are required to perform this experiment. When one System is in the **Server Mode** other should be in the **Client Mode**. The objective can be executed in **NetSim** using the programming exercise available. In the **Programming** menu select **PC to PC Communication → Socket Programming**.

Sample Inputs - By using the **Input Panel** that is available on the left hand side a **Sample Scenario** is created. The Steps involved in creating are as follows,

- **Sample Mode** should be selected.
- **Select a Protocol** from the following,
 - **TCP**
 - **UDP**
- Under **Operation, Server or Client** should be selected.
- On the Right hand side panel there is an **Input Field “Enter Data to be Transmitted”**, where **Data** needs to be typed in. This **Data** is later sent to the **Receiver System**. Here Type “Hello”.
- Under **Input** there are two things,
 - When **Operation is Client**, then the **Server’s IP Address** (192.168.1.2) should be given in the **Server IP Address** field.
 - When **Operation is Server**, then the **Server’s IP Address** (192.168.1.1) would be automatically filled in the **Local IP Address** field.

Sample Output:

- On the Right hand side panel there is an **Input Field “Received Data”**, where **Data** would get displayed. “Hello” is the **Data** that is received from the **Client System**.

TCP

- First the **Server** should **click** on the **Run** button after which the **Client** should click on the **Run** button to **Create the socket**
- **Client** should click on the **Connect** button to **establish the connection with server**.
- The **Client** should click on the **Send** button to transmit the data to the **Server**.
- The **Client** should click on the **Close** button to terminate the Connection with **Server**.
- If the **Data** is successfully transmitted then the **Sent Data** would be **Received** in the **Server System**.

UDP

- First the **Server** should **click** on the **Run** button after which the **Client** should click on the **Run** button to **Create the socket**
- The **Client** should click on the **Send** button to transmit the data to the **Server**.
- The **Client** should click on the **Close** button to terminate the Connection with **Server**.
- If the **Data** is successfully transmitted then the **Sent Data** would be **Received** in the **Server System**.

PC to PC Communication – Chat Application TCP

Programming Guidelines

This section guides the user to link his/her own code for PC to PC Communication – Socket programming to NetSim.

Pre – Conditions

The user program should read the input scenario from text file named '**Input**' with extension txt.

The user program after executing the concept should write the required output to a file named '**Output**' with extension txt.

General Program Flow

The program begins with the *Reading of the Inputs* from the input file **Input.txt**.

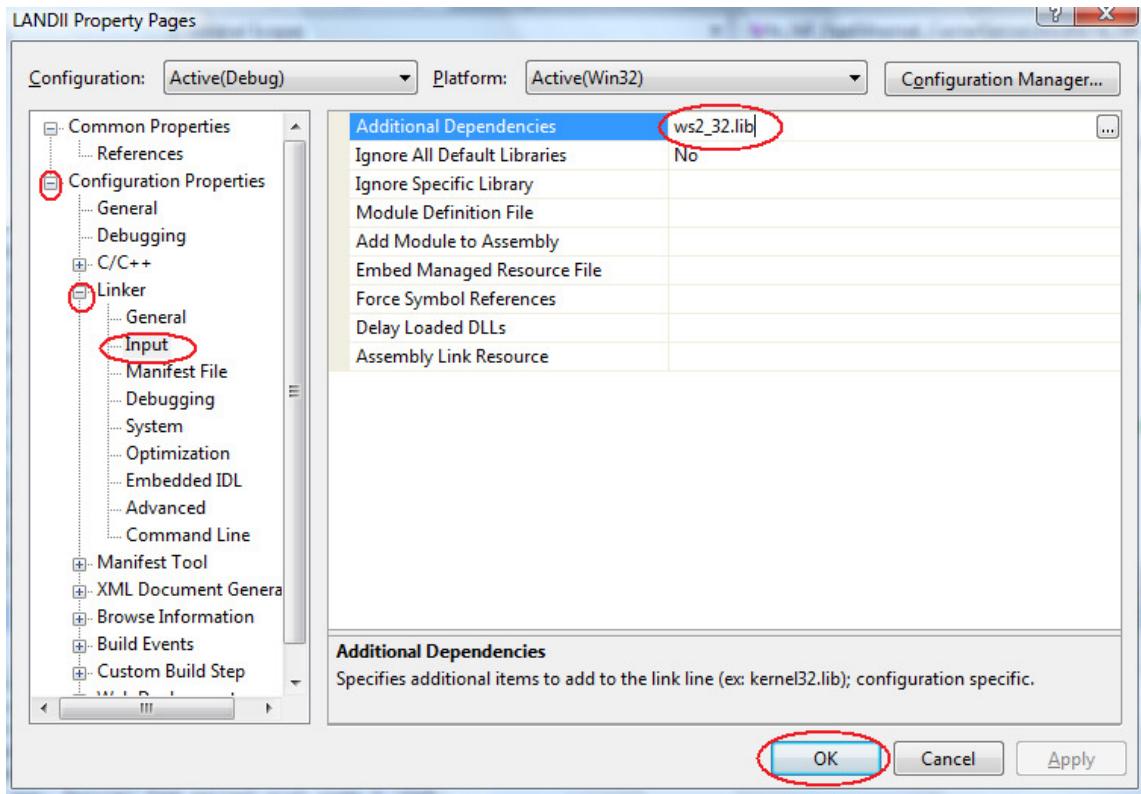
Executing the required concept and

The results of the program should be written into the output file **Output.txt**.

Coding Pre Requisites:

There are certain program prerequisites for the creation of sockets in the Windows-based C complier. They are as follows:

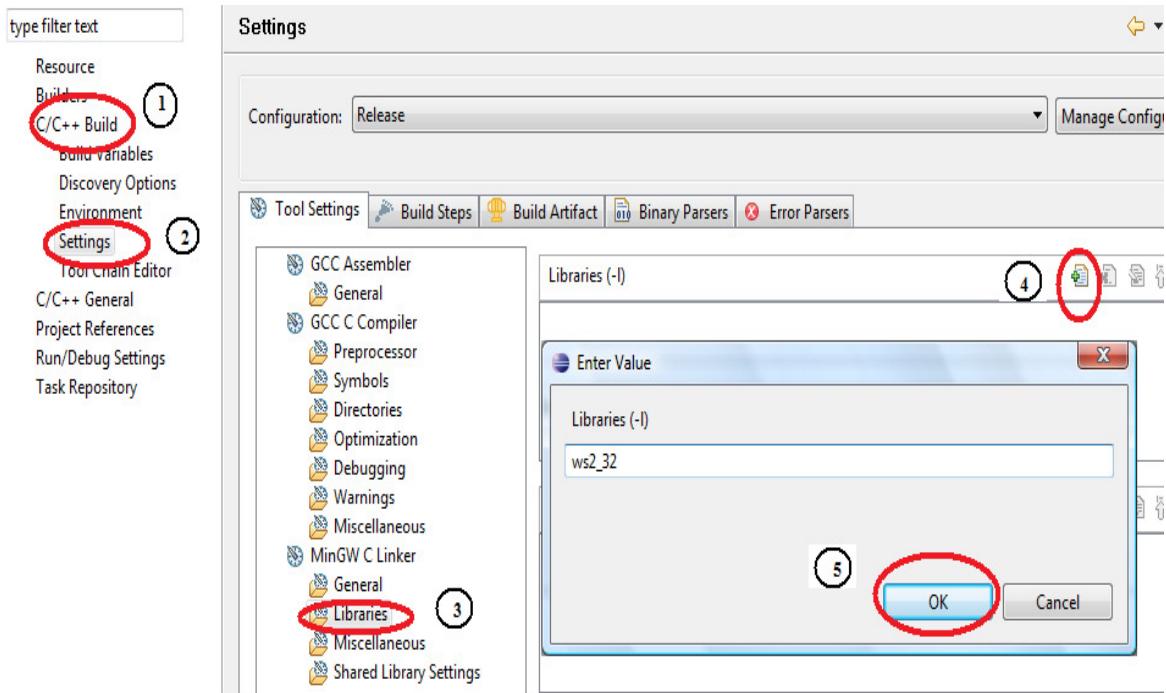
1. The C complier must have **winsock.h** header file in its standard header file folder.
2. In order to use the commands of the winsock.h, there must be **wsock32.lib** library file present in the C complier. It must be linked to your application. For example, if you have a Visual Studio complier 2005 (8.0 versions), the library file linkage must be made in the following way:



- a. Click **Project Menu** in the workspace.
- b. Select **properties** in the pull-down menu or press Alt + F7
- c. Then click **Linker**.
- d. Select **Input**
- e. Type **ws2_32.lib** file in the **Additional Dependencies** text box area.
- e. Click **OK** to confirm the changes in the workspace.

OR

If you have an eclipse complier, the library linkage must be in the following way



- a. Click **Project Menu** in the workspace.
- b. Select **properties** in the pull-down menu
- c. Then click **C/C++ Build**.
- d. Select **Setting**.
- e. Select **Libraries** under **MinGW C Linker**
- f. Press **add** button in the **Libraries (-l)** window
- d. Type **ws2_32** file in the text box area.
- e. Click **OK** to confirm the changes in the workspace.

Input File Format	Output File Format
<p>The data format in the input file is as follows,</p> <p>Receive Input Protocol=TCP Local User Name=Tetcos Send Protocol=TCP Number of Users=2 1=192.168.0.1 2=192.168.0.2 Tetcos:-Hello!!!!!!!!!!!!!!</p>	<p>Write the received message Ex: Tetcos:-Hello!!!!!!!!!!!!!!</p>

Interface Source Code

Interface source code written in C is given using this the user can write the TCP chat functions. To view the interface source code, go to

NetSim Installation path / src / Programming/ ChatTCP.c

To find NetSim's Installation path right click NetSim icon and select

- Open file location in Windows 7
- Open file location in Windows Vista
- Properties --> find target in Windows XP.

PC to PC Communication – Chat Application UDP

Programming Guidelines

This section guides the user to link his/her own code for PC to PC Communication – Socket programming to NetSim.

Pre – Conditions`

The user program should read the input scenario from text file named '**Input**' with extension txt.

The user program after executing the concept should write the required output to a file named '**Output**' with extension txt.

General Program Flow

The program begins with the *Reading of the Inputs* from the input file **Input.txt**.

Executing the required concept and

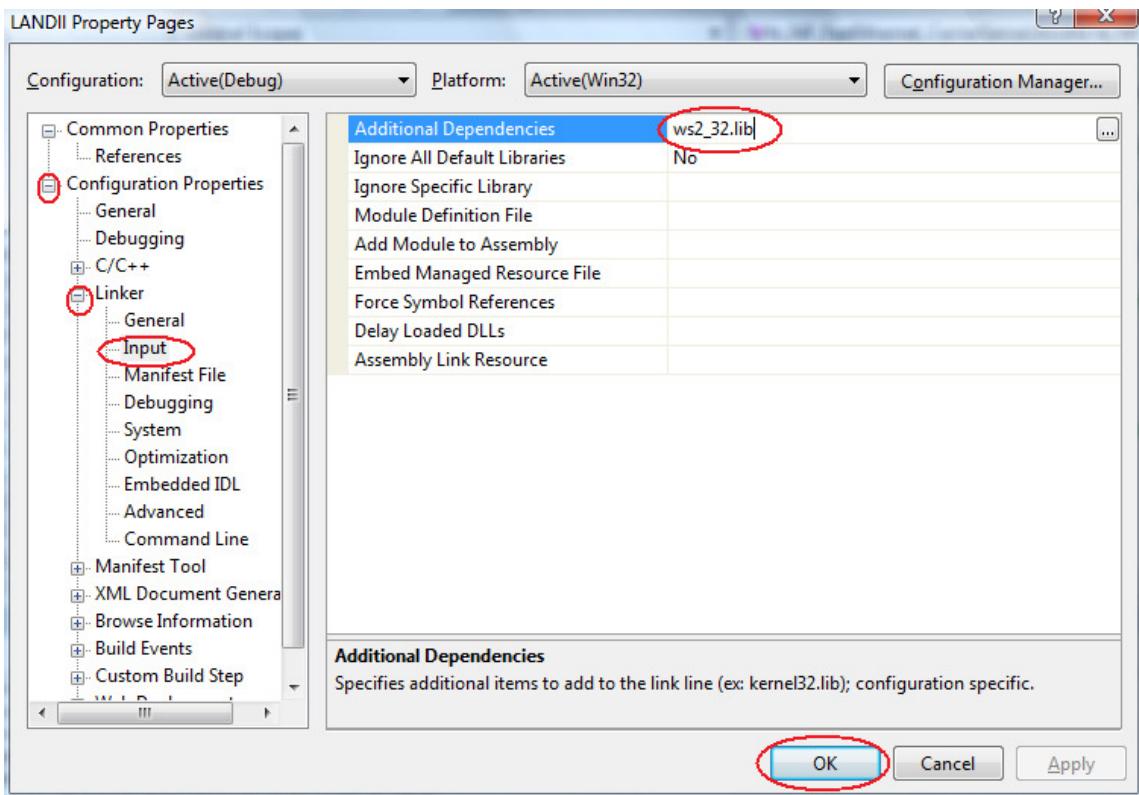
The results of the program should be written into the output file **Output.txt**.

Coding Pre Requisites:

There are certain program prerequisites for the creation of sockets in the Windows-based C complier. They are as follows:

1. The C complier must have **winsock.h** header file in its standard header file folder.

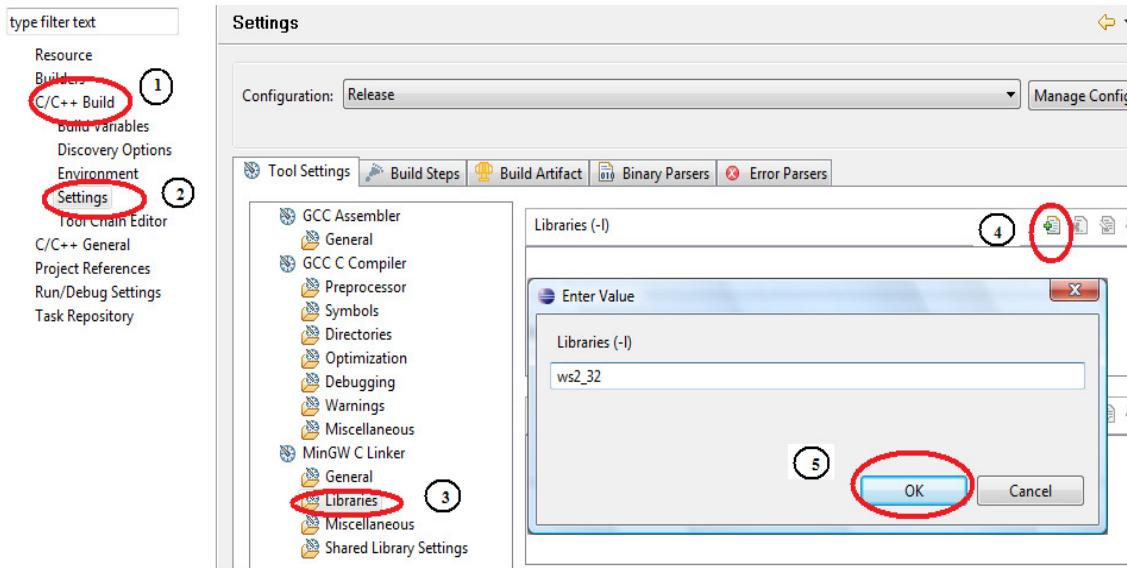
2. In order to use the commands of the winsock.h, there must be **wsock32.lib** library file present in the C complier. It must be linked to your application. For example, if you have a Visual Studio complier 2005 (8.0 versions), the library file linkage must be made in the following way:



- Click **Project Menu** in the workspace.
- Select **properties** in the pull-down menu or press Alt + F7
- Then click **Linker**.
- Select **Input**
- Type **ws2_32.lib** file in the **Additional Dependencies** text box area.
- Click **OK** to confirm the changes in the workspace.

OR

If you have an eclipse complier, the library linkage must be in the following way



- a. Click **Project Menu** in the workspace.
- b. Select **properties** in the pull-down menu
- c. Then click **C/C++ Build**.
- d. Select **Setting**.
- e. Select **Libraries** under **MinGW C Linker**
- f. Press **add** button in the **Libraries (-l)** window
- d. Type **ws2_32** file in the text box area.
- e. Click **OK** to confirm the changes in the workspace..

Input File Format	Output File Format
<p>The data format in the input file is as follows.,</p> <p>Receive Input</p> <p>Protocol=UDP</p> <p>Local User Name=Tetcos</p> <p>Send</p> <p>Protocol=TCP</p> <p>Number of Users=2</p> <p>1=192.168.0.1</p> <p>2=192.168.0.2</p> <p>Tetcos:-Hello!!!!!!!!!!</p>	<p>Write the received message</p> <p>Ex:</p> <p>192.168.0.2:- Requesting for the connection</p>

Interface Source Code

Interface source code written in C is given using this the user can write UDP Chat functions. To view the interface source code, go to

NetSim Installation path / src / Programming/ ChatUDP.c

To find NetSim's Installation path right click NetSim icon and select

- Open file location in Windows 7
- Open file location in Windows Vista
- Properties --> find target in Windows XP.

Sample Scenarios:

Objective - Send and Receive the data using PC to PC Communication – Chat Application

How to Proceed? - Two to Ten Systems are required to perform this experiment.

The objective can be executed in **NetSim** using the programming exercise available. In the **Programming** menu select **PC to PC Communication → Chat Application**.

Sample Inputs - By using the **Input Panel** that is available on the left hand side a **Sample Scenario** is created. The Steps involved in creating are as follows,

- **Sample Mode** should be selected.
- **Select a Protocol** from the following,
 - **TCP**
 - **UDP**
- On the Right hand side panel there is an **Input Field “Send Data”**, where **Data** needs to be typed in. This **Data** is later sent to the **Receiver System**. Here **Type “Hello”**.
- On the Right hand side panel there is “**Receive Data**”, where data is received from user. Ex. “**Hello**”
- Under **Input** there are two things,
 - Number of users must be select between one to ten
 - Enter the IP address of the data where needs to be in all the column in the input pane.

Sample Output:

- On the Right hand side panel there is an **Input Field “Received Data”**, where **Data** would get displayed. “Hello” is the **Data** that is received from the **users**.

TCP

- Click **Run** button to **create the socket** and start Receive Data from the users.
- Click **Send** button to transmit the data to the users.
- Click **Refresh** button to terminate the connection.

UDP

- Click **Run** button to **create the socket** and start Receive Data from the users.
- Click **Send** button to transmit the data to the users.

Scheduling - First In First Out (FIFO)

Programming Guidelines

This section guides the user to link his/her own code for Scheduling - FIFO to NetSim.

Pre - Conditions

The program should read the data from ‘**Input.txt**’ file that is available in the temporary directory.

The program should write the output data to ‘**Output.txt**’ file that is available in the temporary directory.

Note: The temporary directory is navigated through the following step.

Run → Type "%temp%" → NetSim → "Input.txt" and "Output.txt"

General Program Flow

The program begins with the *Reading of the Inputs* from the input file **Input.txt**.

Executing the required concept and, the results of the program should be written into the output file **Output.txt**.

Note: The naming of the input and the output file must be same as the text displayed in the Methodology screen

File Format

Input File Format	Output File Format
Output_Link_Capacity=100 Node's_Priority=8,7,6,5,4,3,2,1, Node's_Capacity=0,0,500,0,0,0,0,0,	Output.txt file format is as follows, Seconds>no of bits transmitted in node 1>Transmission sequence of node 1>... no of bits transmitted in node 8>Transmission sequence of node 8> Sample Output File 1>0>0>0>0>100>1>0>0>0>0>0>0>0>0> 2>0>0>0>0>100>1>0>0>0>0>0>0>0>0> 3>0>0>0>0>100>1>0>0>0>0>0>0>0>0> 4>0>0>0>0>100>1>0>0>0>0>0>0>0>0> 5>0>0>0>0>100>1>0>0>0>0>0>0>0>0>

Interface Source Code

Interface source code written in C is given using this the user can write only the Scheduling inside the function fnFIFO() using the variables already declared. To view the interface source code, go to

NetSim Installation path / src / Programming/ FIFO.c

To find NetSim's Installation path right click NetSim icon and select

- Open file location in Windows 7
- Open file location in Windows Vista
- Properties --> find target in Windows XP.

Sample Scenarios:

Objective - The objective is to allocate the output capacity among the various input capacities using **First In First Out (FIFO) Scheduling** algorithm.

How to Proceed? - The objective can be executed in **NetSim** using the programming exercise available. In the **Programming** menu select **Scheduling**.

Sample Input - By using the **Input Panel** that is available on the left hand side a **Sample Scenario** is created. The Steps involved in creating are as follows,

- **Sample Mode** should be selected.

- **Algorithm** should be selected as **First In First Out (FIFO)**.
- Under **Input** field,
 0. **Enter Output Link Capacity** which is in **MBPS**. The options that are available for selecting are 100, 200, 300, 1000 MBPS.
Select **400 MBPS**.
 1. **Enter Bits / Node** for one second (limit 0 - 9999) in the fields available.
i.e. Node1 - “100”, Node2 - “100”, , Node8 - “100”.
 - The **Priority** for the **Node** needs to be given. Click to Select or Deselect the Priority of a Node. In this **Sample** the **Priorities** are as follows,

Node Number	Priority Given
1	7
2	6
3	8
4	3
5	2
6	1
7	5
8	4

- Click on the **Run** button.

Output

Note: The Transmission of the data takes place in the following sequence. The Bits / Node are represented in different colors. These colors indicate the order in which the transmission takes place.

The Transmission Sequence →  1 2 3 4 5 6 7 8.

Seconds	1	2
Node1	0	100
Node2	0	100
Node3	100	0
Node4	0	100
Node5	0	100
Node6	100	0
Node7	100	0
Node8	100	0

Scheduling - Max - Min Fair (MMF)

Programming Guidelines

This section guides the user to link his/her own code for Scheduling - MMF to NetSim.

Pre - Conditions

The program should read the data from '**Input.txt**' file that is available in the application path.

The program should write the output data to '**Output.txt**' file that is available in the application path.

Note: The temporary directory is navigated through the following step.

Run → Type "%temp%" → NetSim → "Input.txt" and "Output.txt"

General Program Flow

The program begins with the *Reading of the Inputs* from the input file **Input.txt**.

Executing the required concept and, the results of the program should be written into the output file **Output.txt**.

Note: The naming of the input and the output file must be same as the text displayed in the Methodology screen

File Format

Input File Format	Output File Format
Output_Link_Capacity =100 Node's_Capacity=0,0, 300,0,0,0,0,0,	<p>‘Output.txt’ file format is as follows.</p> <p>1st second >Output rate of node 1 >Priority of Node 1>.....>Output rate of node 8.Priority of Node8</p> <p>2nd second >Output rate of node 1 >Priority of Node 1>.....>Output rate of node 8.Priority of Node8</p> <p>Nth second >Output rate of node 1 >Priority of Node 1>.....>Output rate of node 8.Priority of Node8</p> <p>Sample File data</p> <p>1>0.00>0>0.00>0>100.00>1>0.00>0>0.00>0>0.00>0>0.00>0> 2>0.00>0>0.00>0>100.00>1>0.00>0>0.00>0>0.00>0>0.00>0>0.00>0> 3>0.00>0>0.00>0>100.00>1>0.00>0>0.00>0>0.00>0>0.00>0>0.00>0></p>

Interface Source Code

Interface source code written in C is given using this the user can write only the Scheduling inside the function fnMMFA () using the variables already declared. To view the interface source code, go to

NetSim Installation path / src / Programming/ MMF.c

To find NetSim's Installation path right click NetSim icon and select

- Open file location in Windows 7
- Open file location in Windows Vista
- Properties --> find target in Windows XP.

Sample Scenarios:

Objective - The objective is to allocate the output capacity among the various input capacities using **Max-Min Fare(MMF) Scheduling** algorithm.

How to Proceed? - The objective can be executed in **NetSim** using the programming exercise available. In the **Programming** menu select **Scheduling**.

Sample Input - By using the **Input Panel** that is available on the left hand side a **Sample Scenario** is created. The Steps involved in creating are as follows,

- **Sample Mode** should be selected.
- **Algorithm** should be selected as **Max - Min Fair (MMF)**.
- Under **Input** field,
 0. **Enter Output Link Capacity** which is in **MBPS**. The options that are available for selecting are 100, 200, 300, 1000 MBPS. **Select 100 MBPS**.
 1. **Enter Bits / Node** for one second (limit 0 - 9999) in the fields available.
i.e. Node1 - "100" , Node2 - "100", , Node8 - "100".
- **Click on the Run button.**

Output

The Transmission of the data takes place in the following sequence. The **Bits / Node** are represented in different colors. These colors indicate the order in which the transmission takes place.

The Transmission Sequence →  1 2 3 4 5 6 7 8.

Seconds	1	2	3	4	5	6	7	8
Node1	12.50	12.50	12.50	12.50	12.50	12.50	12.50	12.50
Node2	12.50	12.50	12.50	12.50	12.50	12.50	12.50	12.50
Node3	12.50	12.50	12.50	12.50	12.50	12.50	12.50	12.50
Node4	12.50	12.50	12.50	12.50	12.50	12.50	12.50	12.50
Node5	12.50	12.50	12.50	12.50	12.50	12.50	12.50	12.50
Node6	12.50	12.50	12.50	12.50	12.50	12.50	12.50	12.50
Node7	12.50	12.50	12.50	12.50	12.50	12.50	12.50	12.50
Node8	12.50	12.50	12.50	12.50	12.50	12.50	12.50	12.50

Shortest Path - Floyd's

Programming Guidelines

This section guides the user to link his/her own code for shortest path – Floyd's to NetSim.

Pre - Conditions

The program should read the data from ‘Input.txt’ file that is available in the temporary directory.

The program should write the output data to ‘Output.txt’ file that is available in the temporary directory.

Note: The temporary directory is navigated through the following step.

Run → Type "%temp%" → NetSim → "Input.txt" and "Output.txt"

General Program Flow

The program begins with the *Reading of the Inputs* from the input file Input.txt.

Executing the required concept and, the results of the program should be written into the output file Output.txt.

Input file format	Output file format
<p>Algorithm=Floyd's No_of_Router=3 Distance: 999>5>1> 5>999>10> 1>10>999> Source_Router=1 Destination_Router=3</p> <p>Note: '>' is the delimiter symbol, which is used to separate each input.</p>	<p>Consider the source node is 1 Distance>0>1>2>1>2>1> Path>0>1>2>1>4>1> The first line has the all the distance values from the source node to all the other connected nodes. The second line has the path values from the source node to all the other nodes. The first line contains the values stored in the distance array (single dimensional array) used in the algorithm. The second line contains the values stored in the path array (single dimensional array) used in the algorithm.</p> <p>Sample Output: 1>2>3> 1>2> 1>3> 1> >>> path_FS> 0>0>0> 0>0>1> 0>1>0> distance_FS> 0>5>1> 5>0>6> 1>6>0></p> <p>Note: The string Distance and Path in the file is compulsory. The output must be stored in the same format</p>

Interface Source Code

Interface source code written in C is given using this the user can write only the Scheduling inside the function fnFloyd() using the variables already declared. To view the interface source code, go to

NetSim Installation path / src / Programming/ Floyds.c

To find NetSim's Installation path right click NetSim icon and select

- Open file location in Windows 7
- Open file location in Windows Vista
- Properties --> find target in Windows XP.

Sample Scenarios:

Objective - The objective is to find the **Shortest Path** between the two **Routers** using the **Floyds** algorithm.

How to Proceed? - The objective can be executed in **NetSim** using the **Programming** exercise available. Under **Programming Menu** select **Shortest Path**.

Sample Input - By using the **Input** Panel that is available on the left hand side a **Sample Scenario** is created. The **Steps** involved in creating are as follows,

- **Sample Mode** should be selected.
- **Select the Algorithm as Floyd's.**
- **Number of Routers** needs to be selected. **Minimum of 3 Routers** and **Maximum of 8 Routers** can be selected. Let us consider **4 Routers** are selected.
- **Source Router** needs to be given. Let **Router 4** be the **Source Router**.
- **Destination Router** also needs to be given. Let **Router 1** be the **Destination Router**.
- **Click on 2 Routers** to give the **distance** between those **2 Routers**. A **blue arrow** would be pointing from the **Source Node** to the **Destination Node**. The **Distances** between the **Routers** that needs to be given are as follows,

1. Router 4 and Router 1 → 67 Km,
2. Router 4 and Router 3 → 46 Km,
3. Router 3 and Router 1 → 18 Km,
4. Router 3 and Router 2 → 56 Km, and
5. Router 1 and Router 2 → 70 Km.

- Click on **Run** button to view the output in the **Routing Table**.

Output - The **Output** for the above **Scenario** is as follows,

- Click the **Run** button to view program output.
- The **Output** is obtained in a **Routing Table**. All possible **Routes** and the corresponding **Distance** are obtained. Below is a **Routing Table** for the above Inputs,

Routing Table	
Route	Distance[KM]
4>1>	67
4>3>1>	64
4>3>2>	102
4>3>	46
4>	0

- The **Shortest Path** would be highlighted in **green color**.
- Click on **Refresh Button** to **Refresh** the screen and create fresh **Scenarios**.

Shortest Path - Link State

Programming Guidelines

This section guides the user to link his/her own code for shortest path - Link State Routing to NetSim.

Pre - Conditions

The program should read the data from '**Input.txt**' file that is available in the temporary directory.

The program should write the output data to '**Output.txt**' file that is available in the temporary directory.

Note: The temporary directory is navigated through the following step.

Run → Type "%temp%" → NetSim → "Input.txt" and "Output.txt"

General Program Flow

The program begins with the *Reading of the Inputs* from the input file **Input.txt**.

Executing the required concept and, the results of the program should be written into the output file **Output.txt**.

Input file format	Output file format
<p>Input.txt contains the distance matrix given by the user, along with the source node, destination node and the total number of nodes. Here, we have the input as a SYMMETRIC matrix. The format of input.txt is, for example,</p> <p>Algorithm=Link_State No_of_Router=3 Distance: 999>6>5> 6>999>7> 5>7>999> Source_Router=1 Destination_Router=2</p> <p><i>Note:</i> '>' is the delimiter symbol, which is used to separate each input.</p>	<p>Consider the source node is 1 Distance>0>1>2>1>2>1> Path>0>1>2>1>4>1></p> <p>The first line has the all the distance values from the source node to all the other connected nodes.</p> <p>The second line has the path values from the source node to all the other nodes.</p> <p>The first line contains the values stored in the distance array (single dimensional array) used in the algorithm.</p> <p>The second line contains the values stored in the path array (single dimensional array) used in the algorithm.</p> <p>Sample Output:</p> <p>1>2> 1>3>2> 1>3> 1> ">>>> Distance>0>6>5> Path>1000>1>1></p> <p><i>Note:</i> The string Distance and Path in the file is compulsory. The output must be stored in the same format.</p>

Interface Source Code

Interface source code written in C is given using this the user can write only the Scheduling inside the function fnDijkstra() using the variables already declared. To view the interface source code, go to

NetSim Installation path / src / Programming/ LinkState.c

To find NetSim's Installation path right click NetSim icon and select

- Open file location in Windows 7
- Open file location in Windows Vista
- Properties --> find target in Windows XP.

Sample Scenarios:

Objective - The objective is to find the **Shortest Path** between the two **routers** using the **Link State** algorithm.

How to Proceed? - Under **Programming Menu** select **Shortest Path**.

Sample Input - By using the **Input Panel** that is available on the left hand side a **Sample Scenario** is created. The Steps involved in creating are as follows,

- **Sample Mode** should be selected.
- **Select the Algorithm as Runed State.**
- **Number of Routers** needs to be selected. **Minimum of 3 Routers** and **Maximum of 8 Routers** can be selected. Select **5 Routers**.
- **Source Router** needs to be given. Let **Router 5** be the **Source Router**.
- **Destination Router** also needs to be given. Let **Router 3** be the **Destination Router**.
- **Click on 2 Routers** to give the distance between those 2 **Routers**. A blue Line would be pointing from the **Source Node** to the **Destination Node**. The **Distances** between the **Routers** that needs to be given are as follows,
 1. Router 1 and Router 2 → 35,
 2. Router 2 and Router 3 → 30,
 3. Router 3 and Router 4 → 60,
 4. Router 4 and Router 1 → 20,

5. Router 4 and Router 5 → 67, and

6. Router 5 and Router 2 → 58.

- Click on **Run** button to view the output in the **Routing Table**.

Output - The **Output** for the above **Scenario** is as follows,

- Click the **Run** button to view program output.
- The **Output** is obtained in a **Routing Table**. All possible **Routes** and the corresponding **Distance** are obtained. Below is a **Routing Table** for the above Inputs,

Routing Table	
Route	Distance[KM]
5>2>1>4>3>	173
5>2>1>4>	113
5>2>1>	93
5>2>3>	88
5>2>	58
5>4>1>2>3>	152
5>4>1>2>	122
5>4>1>	87
5>4>3>	127
5>4>	67
5>	0

- The **Shortest Path** would be highlighted in **Green** color in the **Routing Table**.
- Click on **Refresh Button** to **Refresh** the screen and create fresh **Scenarios**.

Sliding Window Protocol - Go Back N

Programming Guidelines

This section guides the user to link his/her own code for Go Back N to NetSim.

Pre - Conditions

The user program should read the input scenario from text file named '**Input**' with extension txt.

The user program after executing the concept should write the required output to a file named '**Output**' with extension txt.

Note: The temporary directory is navigated through the following step.

Run → Type "%temp%" → NetSim → "Input.txt" and "Output.txt"

General Program Flow

The program begins with the *Reading of the Inputs* from the input file **Input.txt**.

Executing the required concept and

The results of the program should be written into the output file **Output.txt**.

Window Size:

The **Window Size** for **Go Back N** is 7

Input File Format	Output File Format
Algorithm=Go_Back_N Data_File=C:\Users\P.Sathishkumar\Documents\1 Th.txt> Bit_Error_Rate=6 Sequence_Number=3 Window_Size=7	Value1>Value2>Value3>Value4> Types: There are five types of formats of writing in the output file. Each format is written in specific condition, the types are explained below. The condition of writing the types is explained in the algorithm. Type1: Value1 - "CNT", Value 2 -output of the slidingcount function Value 3 - "FRAMES" Value 4 - "TRANSMIT" Type2: Value1 - "DT",

	<p>Value 2 -Frame number, Value 3 - Frame's Source address, Value 4 - Frame's Destination address.</p> <p>Type3: Value 1 - "EV", Value 2 - Output of the intro_error function, Value 3 - Frame' Source address, Value 4 - Frame's Destination address.</p> <p>Type4: Value 1 - "ACK", Value 2 - "POS", Value 3 - Acknowledgement frame's Source Address, Value 4 - Acknowledgement frame's Destination Address.</p> <p>Type5: Value 1 - "DEL" Value 2 - count of frames being deleted Value 3 - "FRAME" Value 4 - "DELETED"</p> <p>Note: The above convention to write into the 'Output.Txt' is mandatory. Values in Quotes"" to be written into file 'Output.Txt' as it is including Case. DT>Frame No>node1>node2> (DT denotes Data from node 1 to node 2) EV>Error Flag>node1>node2> (EV denotes Error Value - i.e If the above frame has error then set the error flag as 1 else set the flag as 0) ACK>POS>node2>node1> (Acknowledgement for that above frame received is sent to node 2 to node 1)</p> <p>Ex: CNT>1>FRAMES>TRANSMIT> DT>1>node1>node2> EV>0>node1>node2> ACK>POS>node1>node2></p>
--	--

Interface Source Code

Interface source code written in C is given using this the user can write only the Sliding Window Protocols - Go Back N inside the function GoBackN () using the variables already declared. To view the interface source code, go to

NetSim Installation path / src / Programming/ GobackN_SW.c

To find NetSim's Installation path right click NetSim icon and select

- Open file location in Windows 7
- Open file location in Windows Vista
- Properties --> find target in Windows XP.

Sample Scenarios:

Objective - Send and receive the data using **Sliding Window Protocol - Go Back N**.

How to Proceed? - The objective can be executed in **NetSim** using the programming exercise available. Under **Programming Menu** select **Sliding Window Protocol → Go Back N**.

Sample Input - By using the **Input Panel** that is available on the left hand side a **Sample Scenario** is created. The Steps involved in creating are as follows,

- **Sample Mode** should be selected.
 - **Select Go Back N as Algorithm.**
 - **Create a “Data file (.txt)”** and save it on the disk.
 - **Select the path of the above created “Data file (.txt)”** which could be **Maximum of 100000 bytes.**
 - **Select Bit Error Rate (BER)** as “**10^-5**” from the drop down menu.
 - **Select Sequence number (Bits)** as “**3**” from the drop down menu.
 - A default value is entered for **Window Size** as “**7**”.
1. Then **Run** button need to be clicked.

Output - **Output** for the above **Sample** is as follows,

2. An **Output Table** is obtained when **Run** button is clicked. In the **Output Table, Transmitted frame (Data and Acknowledgement)** and their corresponding Counts is obtained. The **Total Count** is also obtained. The table is given below,

Output	
Total data frames to be transmitted - 24	
Transmitted frame	Count
Data	30
Acknowledgement	5
Total = 35	

Note - The “Total data frames to be transmitted” and “Total count” in the **Output** table depends on size of the “.txt” file.

- The details of the **Data Frames** flowing from **Node 1** to **Node 2** are obtained on the right hand side panel. Below are the details that is obtained in the tool,
 - Data Frame 1 is flowing from Node1 to Node2 with no error.
 - Data Frame 2 is flowing from Node1 to Node2 with no error.
 - Data Frame 3 is flowing from Node1 to Node2 with no error.
 - **Data Frame 4 is flowing from Node1 to Node2 with error.**
 - Data Frame 5 is flowing from Node1 to Node2 with no error.
 - Data Frame 6 is flowing from Node1 to Node2 with no error.
 - **Data Frame 7 is flowing from Node1 to Node2 with error.**
 - **Acknowledgement from Node2 to Node1.**
 - Data Frame 4 is flowing from Node1 to Node2 with no error.
 - Data Frame 5 is flowing from Node1 to Node2 with no error.
 - Data Frame 6 is flowing from Node1 to Node2 with no error.
 - Data Frame 7 is flowing from Node1 to Node2 with no error.
 - Data Frame 8 is flowing from Node1 to Node2 with no error.
 - Data Frame 9 is flowing from Node1 to Node2 with no error.
 - Data Frame 10 is flowing from Node1 to Node2 with no error.
 - **Acknowledgement from Node2 to Node1.**
 - Data Frame 11 is flowing from Node1 to Node2 with no error.
 - Data Frame 12 is flowing from Node1 to Node2 with no error.
 - Data Frame 13 is flowing from Node1 to Node2 with no error.
 - Data Frame 14 is flowing from Node1 to Node2 with no error.
 - Data Frame 15 is flowing from Node1 to Node2 with no error.
 - Data Frame 16 is flowing from Node1 to Node2 with no error.
 - Data Frame 17 is flowing from Node1 to Node2 with no error.

- Acknowledgement from Node2 to Node1.
 - Data Frame 18 is flowing from Node1 to Node2 with no error.
 - Data Frame 19 is flowing from Node1 to Node2 with no error.
 - Data Frame 20 is flowing from Node1 to Node2 with no error.
 - Data Frame 21 is flowing from Node1 to Node2 with no error.
 - Data Frame 22 is flowing from Node1 to Node2 with no error.
 - **Data Frame 23 is flowing from Node1 to Node2 with error.**
 - Data Frame 24 is flowing from Node1 to Node2 with no error.
 - Acknowledgement from Node2 to Node1.
 - Data Frame 23 is flowing from Node1 to Node2 with no error.
 - Data Frame 24 is flowing from Node1 to Node2 with no error.
 - Acknowledgement from Node2 to Node1.
- Once the sample experiment is done, then **Refresh** button can be clicked to create new samples.

Sliding Window Protocol - Selective Repeat

Programming Guidelines

This section guides the user to link his/her own code for Selective Repeat to NetSim.

Pre - Conditions

The user program should read the input scenario from text file named '**Input**' with extension txt.

The user program after executing the concept should write the required output to a file named '**Output**' with extension txt.

Note: The temporary directory is navigated through the following step.

Run → Type "%temp%" → NetSim → "Input.txt" and "Output.txt"

General Program Flow

The program begins with the *Reading of the Inputs* from the input file **Input.txt**.

Executing the required concept and

The results of the program should be written into the output file **Output.txt**.

Window Size:

The **Window Size** for Selective Repeat is **7**

Input File Format	Output File Format
Algorithm=Selective_Repeat Data_File=C:\Users\P.Sathishkumar\\Documents\1 Th.txt> Bit_Error_Rate=5 Sequence_Number=3 Window_Size=4	Value1>Value2>Value3>Value4> Types: There are five types of formats of writing in the output file. Each format is written in specific condition, the types are explained below. The condition of writing the types is explained in the algorithm. Type1: Value1 - "CNT", Value 2 - output of the slidingcount function Value 3 - "FRAMES" Value 4 - "TRANSMIT" Type2: Value1 - "DT", Value 2 - Frame number, Value 3 - Frame's Source address, Value 4 - Frame's Destination address. Type3: Value 1 - "EV", Value 2 - Output of the intro_error function, Value 3 - Frame' Source address, Value 4 - Frame's Destination address. Type4: Value 1 - "ACK", Value 2 - "POS", Value 3 - Acknowledgement frame's Source Address, Value 4 - Acknowledgement frame's Destination Address. Type5: Value 1 - "DEL" Value 2 - count of frames being deleted Value 3 - "FRAME" Value 4 - "DELETED" Note: The above convention to write into the ' Output.Txt ' is mandatory .

	<p>Values in Quotes"" to be written into file 'Output.Txt' as it is including Case.</p> <p>DT>Frame No>node1>node2> (DT denotes Data from node 1 to node 2)</p> <p>EV>Error Flag>node1>node2> (EV denotes Error Value - i.e If the above frame has error then set the error flag as 1 else set the flag as 0)</p> <p>ACK>POS>node2>node1></p> <p>(Acknowledgement for that above frame received is sent to node 2 to node 1)</p> <p>Ex:</p> <p>CNT>1>FRAMES>TRANSMIT></p> <p>DT>1>node1>node2></p> <p>EV>0>node1>node2></p> <p>ACK>POS>node1>node2></p>
--	---

Interface Source Code

Interface source code written in C is given using this the user can write only the Sliding Window Protocol - Selective Repeat inside the function SelectiveRepeat () using the variables already declared. To view the interface source code, go to

NetSim Installation path / src / Programming/ SelectiveRepeat_SW.c

To find NetSim's Installation path right click NetSim icon and select

- Open file location in Windows 7
- Open file location in Windows Vista
- Properties --> find target in Windows XP.

Sample Scenarios:

Objective - Send and receive the data using **Sliding Window Protocol - Selective Repeat**.

How to Proceed? - The objective can be executed in **NetSim** using the programming exercise available. Under **Programming Menu** select **Sliding Window Protocol → Selective Repeat**.

Sample Input - By using the **Input Panel** that is available on the left hand side a **Sample Scenario** is created. The Steps involved in creating are as follows,

- **Sample Mode** should be selected.
 - **Select Selective Repeat as Algorithm.**
 - **Create a “Data file (.txt)”** and save it on the disk.
 - **Select the path of the above created “Data file (.txt)”** which could be **Maximum of 100000 bytes.**
 - **Select Bit Error Rate (BER)** as “**10^-5**” from the drop down menu.
 - **Select Sequence number (Bits)** as “**3**” from the drop down menu.
 - A default value is entered for **Window Size** as “**4**”.
1. Then **Run** button need to be clicked.

Output - Output for the above **Sample** is as follows,

2. An **Output Table** is obtained when **Run** button is clicked. In the **Output Table, Transmitted frame (Data and Acknowledgement)** and their corresponding **Counts** is obtained. The **Total Count** is also obtained. The table is given below,

Output	
Total data frames to be transmitted - 24	
Transmitted frame	Count
Data	26
Acknowledgement	7
Total = 33	

Note - The “Total data frames to be transmitted” and “Total count” in the **Output** table depends on size of the “.txt” file.

3. The details of the **Data Frames** flowing from **Node 1 to Node 2** are obtained on the right hand side panel. Below are the details that is obtained in the tool,
 - Data Frame 1 is flowing from Node1 to Node2 with no error
 - Data Frame 2 is flowing from Node1 to Node2 with no error
 - Data Frame 3 is flowing from Node1 to Node2 with no error
 - **Data Frame 4 is flowing from Node1 to Node2 with error**
 - **Acknowledgement from Node2 to Node1**
 - Data Frame 4 is flowing from Node1 to Node2 with no error
 - Data Frame 5 is flowing from Node1 to Node2 with no error

- Data Frame 6 is flowing from Node1 to Node2 with error
 - Data Frame 7 is flowing from Node1 to Node2 with no error
 - Acknowledgement from Node2 to Node1
 - Data Frame 6 is flowing from Node1 to Node2 with no error
 - Data Frame 8 is flowing from Node1 to Node2 with no error
 - Data Frame 9 is flowing from Node1 to Node2 with no error
 - Data Frame 10 is flowing from Node1 to Node2 with no error
 - Acknowledgement from Node2 to Node1
 - Data Frame 11 is flowing from Node1 to Node2 with no error
 - Data Frame 12 is flowing from Node1 to Node2 with no error
 - Data Frame 13 is flowing from Node1 to Node2 with no error
 - Data Frame 14 is flowing from Node1 to Node2 with no error
 - Acknowledgement from Node2 to Node1
 - Data Frame 15 is flowing from Node1 to Node2 with no error
 - Data Frame 16 is flowing from Node1 to Node2 with no error
 - Data Frame 17 is flowing from Node1 to Node2 with no error
 - Data Frame 18 is flowing from Node1 to Node2 with no error
 - Acknowledgement from Node2 to Node1
 - Data Frame 19 is flowing from Node1 to Node2 with no error
 - Data Frame 20 is flowing from Node1 to Node2 with no error
 - Data Frame 21 is flowing from Node1 to Node2 with no error
 - Data Frame 22 is flowing from Node1 to Node2 with no error
 - Acknowledgement from Node2 to Node1
 - Data Frame 23 is flowing from Node1 to Node2 with no error
 - Data Frame 24 is flowing from Node1 to Node2 with no error
 - Acknowledgement from Node2 to Node1
- Once the sample experiment is done, then **Refresh** button can be clicked to create new samples.

Sorting Technique - Bubble Sort

Programming Guidelines

This section guides the user to link his/her own code for Sorting Algorithm to NetSim.

Pre - Condition

User written program should read the value from the '**Input.txt**' in the temporary directory which is having input from the GUI at runtime

The output should be stored in '**Output.txt**' in the temporary directory for display.

User written program should return an integer value.

Note: The temporary directory is navigated through the following step.

Run → Type "%temp%" → NetSim → "Input.txt" and "Output.txt"

General Program Flow

The program begins with the *Reading of the Inputs* from the input file **Input.txt**.

Executing the required concept and, The results of the program should be written into the output file **Output.txt**.

Note: The naming of the input and the output file must be same as the text displayed in the Methodology screen

File Format

Input File Format	Output File Format
<p>Ascending Order</p> <p>Sorting_Type=Bubble Sorting_Order=Ascending Total_Number=3 Number_to_Sort=5,4,3</p> <p>Descending Order</p> <p>Sorting_Type=Bubble Sorting_Order=Descending Total_Number=3 Number_to_Sort=4,5,6</p>	<p>type>index 1>index 2></p> <p>type 0 : specifies Positioning the data value index that ha</p> <p>type 1 : specifies the two index following it are being compared</p> <p>type 2 : specifies the two index following it are being swapped</p> <p>The following types are used in Quick Sort :</p> <p>type 3 : specifies the two index, where index 1 value is copied to index 2 value.</p> <p>type 4 : specifies the two index, where Position data index value of index1 is copied to the index 2 value.</p> <p>index 1 : index 1 is the index of array that</p>

	<p>points to the first data that is being swapped or compared.</p> <p>index 2 : index 2 is the index of array that points to the second data that is being swapped or compared.</p> <p>Sample Output:</p> <p>Ascending Order:</p> <pre>1>0>1> 2>0>1> 1>1>2> 2>1>2> 1>0>1> 2>0>1></pre> <p>Descending Order:</p> <pre>1>0>1> 2>0>1> 1>1>2> 2>1>2> 1>0>1> 2>0>1></pre>
--	---

Interface Source Code

Interface source code written in C is given using this the user can write only the Sorting Algorithm inside the function fnBubblesort () using the variables already declared. To view the interface source code, go to

NetSim Installation path / src / Programming/ BubbleSort.c

To find NetSim's Installation path right click NetSim icon and select

- Open file location in Windows 7
- Open file location in Windows Vista
- Properties --> find target in Windows XP.

Sample Scenarios:

Objective - To study the working of Sorting Techniques.

How to Proceed? -

The objective can be executed in NetSim using the programming exercise available, under programming user has to select Sorting Techniques.

Sample Inputs - In the Input panel the following steps need to be done,

- **Sample Mode** should be selected.
- Select the Bubble Sort as Sorting Type.
- Select the Sorting Order as either Ascending or Descending.
- Select the total number that has to be sorted. The values available are from **3 to 25**.
- Enter the Number Value in the field provided. The value entered should be within the range of **1 to 9999**.
- Then **Run** button need to be clicked. **Refresh** button can be used if new Inputs have to be given.

Output - The following steps are under gone internally,

- According to the Sorting Type selected the Output would vary.
- Number of Comparison would be obtained.
- Number of Swapping would be obtained.
- A table with the values tabulated would be obtained.
- Once the sample experiment is done, then **Refresh** button can be clicked to create New Samples.

Sorting Technique - Insertion Sort

Programming Guidelines

This section guides the user to link his/her own code for Sorting Algorithm to NetSim.

Pre - Condition

User written program should read the value from the '**Input.txt**' in the temporary directory which is having input from the GUI at runtime

The output should be stored in '**Output.txt**' in the temporary directory for display.

User written program should return an integer value.

Note: The temporary directory is navigated through the following step.

Run → Type "%temp%" → NetSim → "Input.txt" and "Output.txt"

General Program Flow

The program begins with the *Reading of the Inputs* from the input file **Input.txt**.

Executing the required concept and, The results of the program should be written into the output file **Output.txt**.

Note: The naming of the input and the output file must be same as the text displayed in the Methodology screen

File Format

Input File Format	Output File Format
Ascending Order Sorting_Type=Insertion Sorting_Order=Ascending Total_Number=3 Number_to_Sort=9,8,7	type>index 1>index 2> type 0 : specifies Positioning the data value index that ha type 1 : specifies the two index following it are being compared type 2 : specifies the two index following it are being swapped The following types are used in Quick Sort : type 3 : specifies the two index, where index 1 value is copied to index 2 value. type 4 : specifies the two index, where Position data index value of index1 is copied to the index 2 value. index 1 : index 1 is the index of array that points to the first data that is being swapped or compared. index 2 : index 2 is the index of array that points to the second data that is being swapped or compared. Sample Output: Ascending Order: 1>0>1> 2>0>1> 1>1>2> 2>1>2> 1>0>1> 2>0>1>
Descending Order Sorting_Type=Insertion Sorting_Order=Descending Total_Number=3 Number_to_Sort=1,9,3	Descending Order: 1>0>1> 2>0>1> 1>1>2> 2>1>2> 1>0>1>

Interface Source Code

Interface source code written in C is given using this the user can write only the Sorting Algorithm inside the function fnInsertsort () using the variables already declared. To view the interface source code, go to

NetSim Installation path / src / Programming/ InsertionSort.c

To find NetSim's Installation path right click NetSim icon and select

- Open file location in Windows 7
- Open file location in Windows Vista
- Properties --> find target in Windows XP.
-

Sample Project:

Objective - To study the working of Sorting Techniques.

How to Proceed? -

The objective can be executed in NetSim using the programming exercise available, under programming user has to select Sorting Techniques.

Sample Inputs - In the Input panel the following steps need to be done,

- **Sample Mode** should be **selected**.
- Select the Insertion Sort as Sorting Type.
- Select the Sorting Order as either Ascending or Descending.
- Select the total number that has to be sorted. The values available are from **3 to 25**.
- Enter the Number Value in the field provided. The value entered should be within the range of **1 to 9999**.
- Then **Run** button need to be **clicked**. **Refresh** button can be used if new Inputs have to be given.

Output - The following steps are undergone internally,

- According to the Sorting Type selected the Output would vary.
- Number of Comparison would be obtained.
- Number of Swapping would be obtained.
- A table with the values tabulated would be obtained.
- Once the sample experiment is done, then **Refresh** button can be **clicked** to create New Samples.

Sorting Technique - Quick Sort

Programming Guidelines

This section guides the user to link his/her own code for Sorting Algorithm to NetSim.

Pre - Condition

User written program should read the value from the '**Input.txt**' in the temporary directory which is having input from the GUI at runtime

The output should be stored in '**Output.txt**' in the temporary directory for display.

User written program should return an integer value.

Note: The temporary directory is navigated through the following step.

Run → Type "%temp%" → NetSim → "Input.txt" and "Output.txt"

General Program Flow

The program begins with the *Reading of the Inputs* from the input file **Input.txt**.

Executing the required concept and, The results of the program should be written into the output file **Output.txt**.

Note: The naming of the input and the output file must be same as the text displayed in the Methodology screen

File Format

Input File Format	Output File Format
<p>Ascending Order</p> <p>Sorting_Type=Quick Sorting_Order=Ascending Total_Number=3 Number_to_Sort=33,22,66</p> <p>Descending Order</p> <p>Sorting_Type=Quick Sorting_Order=Descending Total_Number=3 Number_to_Sort=22,33,44</p>	<p>type>index 1>index 2></p> <p>type 0 : specifies Positioning the data value index that ha</p> <p>type 1 : specifies the two index following it are being compared</p> <p>type 2 : specifies the two index following it are being swapped</p> <p>The following types are used in Quick Sort :</p> <p>type 3 : specifies the two index, where index 1 value is copied to index 2 value.</p> <p>type 4 : specifies the two index, where Position data index value of index1 is copied to the index 2 value.</p> <p>index 1 : index 1 is the index of array that points to the first data that is being swapped or compared.</p> <p>index 2 : index 2 is the index of array that points to the second data that is being swapped or compared.</p>
	<p>Sample Output:</p> <p>Ascending:</p> <p>0>0>0> 1>0>2> 3>1>0> 4>0>1> 0>2>2> 0>0>0></p> <p>Descending:</p> <p>0>0>0> 3>2>0> 1>0>1> 4>0>2> 0>0>0> 1>0>1> 4>0>0> 0>1>1></p>

Interface Source Code

Interface source code written in C is given using this the user can write only the Sorting Algorithm inside the function fnQuicksort () using the variables already declared. To view the interface source code, go to

NetSim Installation path / src / Programming/ QuickSort.c

To find NetSim's Installation path right click NetSim icon and select

- Open file location in Windows 7
- Open file location in Windows Vista
- Properties --> find target in Windows XP.

Sample Scenarios:

Objective - To study the working of Sorting Techniques.

How to Proceed? -

The objective can be executed in NetSim using the programming exercise available, under programming user has to select Sorting Techniques.

Sample Inputs - In the Input panel the following steps need to be done,

- **Sample Mode** should be selected.
- Select the Quick Sort as Sorting Type.
- Select the Sorting Order as either Ascending or Descending.
- Select the total number that has to be sorted. The values available are from **3 to 25**.
- Enter the Number Value in the field provided. The value entered should be within the range of **1 to 9999**.
- Then **Run** button need to be clicked. **Refresh** button can be used if new Inputs have to be given.

Output - The following steps are undergone internally,

- According to the Sorting Type selected the Output would vary.
- Number of Comparison would be obtained.
- Number of Swapping would be obtained.
- A table with the values tabulated would be obtained.
- Once the sample experiment is done, then **Refresh** button can be **clicked** to create New Samples.

Sorting Technique - Selection Sort

Programming Guidelines

This section guides the user to link his/her own code for Sorting Algorithm to NetSim.

Pre - Condition

User written program should read the value from the '**Input.txt**' in the temporary directory which is having input from the GUI at runtime

The output should be stored in '**Output.txt**' in the temporary directory for display.

User written program should return an integer value.

Note: The temporary directory is navigated through the following step.

Run → Type "%temp%" → NetSim → "Input.txt" and "Output.txt"

General Program Flow

The program begins with the *Reading of the Inputs* from the input file **Input.txt**.

Executing the required concept and, The results of the program should be written into the output file **Output.txt**.

Note: The naming of the input and the output file must be same as the text displayed in the Methodology screen

File Format

Input File Format	Output File Format
<p>Ascending Order</p> <p>Sorting_Type=Selection Sorting_Order=Ascending Total_Number=3 Number_to_Sort=88,77,66</p>	<p>type>index 1>index 2> type 0 : specifies Positioning the data value index that ha type 1 : specifies the two index following it are being compared type 2 : specifies the two index following it are being swapped The following types are used in Quick Sort :</p>
<p>Descending Order</p> <p>Sorting_Type=Selection Sorting_Order=Descending Total_Number=3 Number_to_Sort=55,11,22</p>	<p>type 3 : specifies the two index, where index 1 value is copied to index 2 value. type 4 : specifies the two index, where Position data index value of index1 is copied to the index 2 value.</p> <p>index 1 : index 1 is the index of array that points to the first data that is being swapped or compared. index 2 : index 2 is the index of array that points to the second data that is being swapped or compared.</p>

Interface Source Code

Interface source code written in C is given using this the user can write only the Sorting Algorithm inside the function select() using the variables already declared. To view the interface source code, go to

NetSim Installation path / src / Programming/ SelectionSort.c

To find NetSim's Installation path right click NetSim icon and select

- Open file location in Windows 7
- Open file location in Windows Vista
- Properties --> find target in Windows XP.

Sample Scenarios:

Objective - To study the working of Sorting Techniques.

How to Proceed? -

The objective can be executed in NetSim using the programming exercise available, under programming user has to select Sorting Techniques.

Sample Inputs - In the Input panel the following steps need to be done,

- **Sample Mode** should be **selected**.
- Select the Selection Sort as Sorting Type.
- Select the Sorting Order as either Ascending or Descending.
- Select the total number that has to be sorted. The values available are from **3 to 25**.
- Enter the Number Value in the field provided. The value entered should be within the range of **1 to 9999**.
- Then **Run** button need to be **clicked**. **Refresh** button can be used if new Inputs have to be given.

Output - The following steps are undergone internally,

- According to the Sorting Type selected the Output would vary.
- Number of Comparison would be obtained.
- Number of Swapping would be obtained.

- A table with the values tabulated would be obtained.
- Once the sample experiment is done, then **Refresh** button can be **clicked** to create New Samples.

Spanning Tree – Borovska

Programming Guidelines

This section guides the user to link his/her own code for Spanning Tree using Borovska algorithm to NetSim.

Pre - Conditions

The user program should read the input scenario from text file named '**Input**' with extension txt.

The user program after executing the concept should write the required output to a file named '**Output**' with extension txt.

Note: The temporary directory is navigated through the following step.

Run → Type "%temp%" → NetSim → "Input.txt" and "Output.txt"

General Program Flow

The program begins with the Reading of the Inputs from the input file **Input.txt**.

Executing the required concept and

The results of the program should be written into the output file **Output.txt**.

Input File Format	Output File Format
Algorithm=Borovska No_of_Switches=3 No_of_Edges=3 Source_Switch=3,Destination_Switch=1, Distance=11, Source_Switch=1,Destination_Switch=2, Distance=22, Source_Switch=3,Destination_Switch=2, Distance=33,	'Output.txt' file has the edges in the spanning tree, which is selected from the input edges. Node1>Node2>Cost> Example: 1>2>22> 1>3>11>

Interface Source Code

Interface source code written in C is given .Using this the user can write only the Borovska algorithm inside the function fnBorovska() using the variables already declared. To view the interface source code, go to

NetSim Installation path / src / Programming/ Boruvksa.c

To find NetSim's Installation path right click NetSim icon and select

- Open file location in Windows 7
- Open file location in Windows Vista
- Properties --> find target in Windows XP.

Sample Scenarios:

Objective - To find a **Spanning Tree** for a network using **Borovska's** algorithm.

How to Proceed? - The objective can be executed in **NetSim** using the programming exercise available. In the **Programming** Menu select **Spanning Tree**.

Sample Input

- **Sample Mode** should be selected.
- **Select Borovska as Algorithm** from the list available.
- **Select the Number of Switches. Select 4 Switches as an Input.**
- **Click on 2 Switches** to give the **distance** between them. Similarly connect all the **Switches** that are available in the network. The **Distances** between the **Routers** that needs to be given are as follows,
 - Switch 1 and Switch 2 → 67 Km
 - Switch 1 and Switch 4 → 46 Km
 - Switch 1 and Switch 3 → 89 Km
 - Switch 2 and Switch 3 → 78 Km
 - Switch 2 and Switch 4 → 99 Km
 - Switch 3 and Switch 4 → 56 Km
- **Click on Run** button to execute. **Refresh** button can be used if new **Inputs** have to be given.

Sample Output

- The **Spanning Tree Table** with the **Path (Source Switch and Destination Switch)** and **Distance** is obtained. Below is the Table that is obtained for the above inputs,

Spanning Tree Table		
Path		Distance[KM]
1	2	67
3	4	56
1	4	46

- The “**Length of the Spanning Tree (KM)**” would be given below in the output panel. The **Spanning Tree Path** consists of **green** lines, whereas the **Non-Spanning Tree** consists of **red** lines. Here in this Sample, “**Length of the Spanning Tree (KM) → 169**”.
- Once the sample experiment is done, then **Refresh** button can be clicked to create new samples.

Spanning Tree – Kruskal

Programming Guidelines

This section guides the user to link his/her own code for Spanning Tree using Kruskal algorithm to NetSim.

Pre - Conditions

The user program should read the input scenario from text file named ‘**Input**’ with extension txt.

The user program after executing the concept should write the required output to a file named ‘**Output**’ with extension txt.

Note: The temporary directory is navigated through the following step.

Run → Type "%temp%" → NetSim → "Input.txt" and "Output.txt"

General Program Flow

The program begins with the Reading of the Inputs from the input file **Input.txt**.

Executing the required concept and

The results of the program should be written into the output file **Output.txt**.

Input File Format	Output File Format
Algorithm=Kruskal's No_of_Switches=3 No_of_Edges=3 Source_Switch=3,Destination_Switch=1,Distance=23, Source_Switch=1,Destination_Switch=2,Distance=34, Source_Switch=3,Destination_Switch=2,Distance=45,	' Output.txt ' file has the edges in the spanning tree, which is selected from the input edges. Node1>Node2>Cost> Example: 3>1>23> 1>2>34>

Interface Source Code

Interface source code written in C is given .Using this the user can write only the Kruskal algorithm inside the function fnKruskal() using the variables already declared. To view the interface source code, go to

NetSim Installation path / src / Programming/ Kruskal.c

To find NetSim's Installation path right click NetSim icon and select

- Open file location in Windows 7
- Open file location in Windows Vista
- Properties --> find target in Windows XP.

Sample Scenarios:

Objective - To find the **Spanning Tree** for a network by using **Kruskal** algorithm.

How to Proceed? - The objective can be executed in **NetSim** using the programming exercise available. In the **Programming** Menu select **Spanning Tree**.

Sample Input

- **Sample Mode** should be selected.
- **Select Kruskal as Algorithm** from the list available.
- **Select the Number of Switches. Select 4 Switches as an Input.**

- Click on **2 Switches** to give the distance between them. Similarly connect all the **Switches** that are available in the network. The **Distances** between the **Routers** that needs to be given are as follows,
 - Switch 1 and Switch 2 → 67 Km
 - Switch 1 and Switch 4 → 46 Km
 - Switch 1 and Switch 3 → 89 Km
 - Switch 2 and Switch 3 → 78 Km
 - Switch 2 and Switch 4 → 99 Km
 - Switch 3 and Switch 4 → 56 Km
- Click on **Run** button to execute. **Refresh** button can be used if new **Inputs** have to be given.

Sample Output

- The **Spanning Tree Table** with the **Path (Source Switch and Destination Switch)** and **Distance** is obtained. Below is the **Table** that is obtained for the above inputs,

Spanning Tree Table		
Path		Distance[KM]
1	4	46
4	3	56
1	2	67

- The “**Length of the Spanning Tree (KM)**” would be given below in the output panel. The **Spanning Tree Path** consists of **green** lines, whereas the **Non-Spanning Tree** consists of **red** lines. Here in this Sample, “**Length of the Spanning Tree (KM) → 169**”
- Once the sample experiment is done, then **Refresh** button can be clicked to create new samples.

Spanning Tree – Prims

Programming Guidelines

This section guides the user to link his/her own code for Spanning Tree using Prims algorithm to NetSim.

Pre - Conditions

The user program should read the input scenario from text file named '**Input**' with extension txt.

The user program after executing the concept should write the required output to a file named '**Output**' with extension txt.

Note: The temporary directory is navigated through the following step.

Run → Type "%temp%" → NetSim → "Input.txt" and "Output.txt"

General Program Flow

The program begins with the Reading of the Inputs from the input file **Input.txt**.

Executing the required concept and

The results of the program should be written into the output file **Output.txt**.

Input File Format	Output File Format
Algorithm=Prim's No_of_Switches=3 No_of_Edges=3 Source_Switch=1 Source_Switch=3,Destination_Switch=1,Distance=44, Source_Switch=1,Destination_Switch=2,Distance=55, Source_Switch=2,Destination_Switch=3,Distance=66,	' Output.txt ' file has the edges in the spanning tree, which is selected from the input edges. Node1>Node2>Cost> Example: 1>3>44> 1>2>55>

Interface Source Code

Interface source code written in C is given .Using this the user can write only the Prim's algorithm inside the function fnPrims() using the variables already declared. To view the interface source code, go to

NetSim Installation path / src / Programming/ Prims.c

To find NetSim's Installation path right click NetSim icon and select

- Open file location in Windows 7
- Open file location in Windows Vista
- Properties --> find target in Windows XP.

Sample Scenarios:

Objective - To find a **Spanning Tree** for a network using **Prims** algorithm.

How to Proceed? - The objective can be executed in **NetSim** using the programming exercise available. In the **Programming** Menu select **Spanning Tree**.

Sample Input

- **Sample Mode** should be selected.
- **Select Prim's as Algorithm** from the list available.
- **Select the Number of Switches. Select 4 Switches as an Input.**
- **Click on 2 Switches** to give the **distance** between them. Similarly connect all the **Switches** that are available in the network. The **Distances** between the **Routers** that needs to be given are as follows,
 - Switch 1 and Switch 2 → 67 Km
 - Switch 1 and Switch 4 → 46 Km
 - Switch 1 and Switch 3 → 89 Km
 - Switch 2 and Switch 3 → 78 Km
 - Switch 2 and Switch 4 → 99 Km
 - Switch 3 and Switch 4 → 56 Km
- **Source Switch** can be any 1 from list.Here, **Source Switch** is selected as "3".

- Click on **Run** button to execute. **Refresh** button can be used if new **Inputs** have to be given.

Sample Output

- The **Spanning Tree Table** with the **Path (Source Switch and Destination Switch)** and **Distance** is obtained. Below is the **Table** that is obtained for the above inputs,

Spanning Tree Table		
Path		Distance[KM]
3	4	56
4	1	46
1	2	67

- The “**Length of the Spanning Tree (KM)**” would be given below in the output panel. The **Spanning Tree Path** consists of **green** lines, whereas the **Non-Spanning Tree** consists of **red** lines. Here in this Sample, “**Length of the Spanning Tree (KM) → 169**”.
- Once the sample experiment is done, then **Refresh** button can be clicked to create New Samples.

Transmission Flow Control - Go Back N

Programming Guidelines

This section guides the user to link his/her own code for Go Back N to NetSim.

Pre - Conditions

The user program should read the inputted scenario from text file named ‘**Input**’ with extension txt.

The user program after executing the concept should write the required output to a file named ‘**Output**’ with extension txt.

Note: The temporary directory is navigated through the following step.

Run → Type "%temp%" → NetSim → "Input.txt" and "Output.txt"

General Program Flow

The program begins with the *Reading of the Inputs* from the input file **Input.txt**.

Executing the required concept and,

The results of the program should be written into the output file **Output.txt**.

Window Size:

The **Window Size** for Go Back N is **7**

Input File Format	Output File Format
Algorithm=Go_Back_N Data_File=C:\Users\P.Sathishkumar\ Documents\1 Th.txt> BER=0	Value1>Value2>Value3>Value4> Types: There are five types of formats of writing in the output file. Each format is written in specific condition, the types are explained below. The condition of writing the types is explained in the algorithm. Type1: Value1 - "CNT", Value 2 - output of the slidingcount function Value 3 - "FRAMES" Value 4 - "TRANSMIT" Type2: Value1 - "DT", Value 2 - Frame number, Value 3 - Frame's Source address, Value 4 - Frame's Destination address. Type3: Value 1 - "EV", Value 2 - Output of the intro_error function, Value 3 - Frame' Source address, Value 4 - Frame's Destination address.

	<p>Type4:</p> <p>Value 1 - "ACK", Value 2 - "POS", Value 3 - Acknowledgement frame's Source Address, Value 4 - Acknowledgement frame's Destination Address.</p> <p>Type5:</p> <p>Value 1 - "DEL" Value 2 - count of frames being deleted Value 3 - "FRAME" Value 4 - "DELETED"</p> <p>Note: The above convention to write into the 'Output.Txt' is mandatory.</p> <p>Values in Quotes"" to be written into file 'Output.Txt' as it is including Case.</p> <p>DT>Frame No>node1>node2> (DT denotes Data from node 1 to node 2)</p> <p>EV>Error Flag>node1>node2> (EV denotes Error Value - i.e If the above frame has error then set the error flag as 1 else set the flag as 0)</p> <p>ACK>POS>node2>node1></p> <p>(Acknowledgement for that above frame received is sent to node 2 to node 1)</p> <p>Ex: CNT>1>FRAMES>TRANSMIT></p> <p>DT>1>node1>node2></p> <p>EV>0>node1>node2></p> <p>ACK>POS>node2>node1></p> <p>DEL>1>FRAME>DELETED></p>
--	--

Interface Source Code

Interface source code written in C is given using this the user can write only the Transmission Flow Control - Go Back N inside the function GoBackN () using the variables already declared. To view the interface source code, go to

NetSim Installation path / src / Programming/ GoBackN_TFC.c

To find NetSim's Installation path right click NetSim icon and select

- Open file location in Windows 7
- Open file location in Windows Vista
- Properties --> find target in Windows XP.

Sample Scenarios:

Objective - Send and receive the data using **Transmission Flow Control - Go Back N**.

How to Proceed? - The objective can be executed in **NetSim** using the programming exercise available. Under **Programming Menu** select **Transmission Flow Control → Go Back N**.

Sample Input - By using the **Input Panel** that is available on the left hand side a **Sample Scenario** is created. The Steps involved in creating are as follows,

- **Sample Mode** should be selected.
- **Select Go Back N as Algorithm.**
- **Create a “Data file (.txt)”** and save it on the disk.
- **Select the path of the above created “Data file (.txt)”** which could be **Maximum of 100000 bytes.**
- **Select Bit Error Rate (BER)** as “**10^-5**” from the drop down menu.

1. Then **Run** button need to be clicked.

Output - **Output** for the above **Sample** is as follows,

2. An **Output Table** is obtained when **Run** button is clicked. In the **Output Table, Transmitted frame (Data and Acknowledgement)** and their corresponding Counts is obtained. The **Total Count** is also obtained. The table is given below,

Output	
Total data frames to be transmitted - 24	
Transmitted frame	Count
Data	30
Acknowledgement	5
Total = 35	

Note - The “Total data frames to be transmitted” and “Total count” in the **Output** table depends on size of the “.txt” file.

- Once the sample experiment is done, then **Refresh** button can be clicked to create new samples.

Transmission Flow Control - Selective Repeat

Programming Guidelines

This section guides the user to link his/her own code for Selective Repeat to NetSim.

Pre - Conditions

The user program should read the inputted scenario from text file named ‘**Input**’ with extension txt.

The user program after executing the concept should write the required output to a file named ‘**Output**’ with extension txt.

Note: The temporary directory is navigated through the following step.

Run → Type "%temp%" → NetSim → "Input.txt" and "Output.txt"

General Program Flow

The program begins with the *Reading of the Inputs* from the input file **Input.txt**.

Executing the required concept and,

The results of the program should be written into the output file **Output.txt**.

Window Size:

The **Window Size** for Selective Repeat is 7

Input File Format	Output File Format
Algorithm=Selective_Repeat Data_File=C:\Users\P.Sathishkumar\ Documents\1 Th.txt> BER=0	Value1>Value2>Value3>Value4> Types: There are five types of formats of writing in the output file. Each format is written in specific condition, the types are explained below. The condition of writing the types is explained in the algorithm. Type1: Value1 - "CNT", Value 2 - output of the slidingcount function Value 3 - "FRAMES" Value 4 - "TRANSMIT" Type2: Value1 - "DT", Value 2 - Frame number, Value 3 - Frame's Source address, Value 4 - Frame's Destination address. Type3: Value 1 - "EV", Value 2 - Output of the intro_error function, Value 3 - Frame' Source address, Value 4 - Frame's Destination address. Type4: Value 1 - "ACK", Value 2 - "POS", Value 3 - Acknowledgement frame's Source Address, Value 4 - Acknowledgement frame's Destination Address. Type5: Value 1 - "DEL"

	<p>Value 2 - count of frames being deleted</p> <p>Value 3 - "FRAME"</p> <p>Value 4 - "DELETED"</p> <p>Note: The above convention to write into the 'Output.Txt' is mandatory.</p> <p>Values in Quotes"" to be written into file 'Output.Txt' as it is including Case.</p> <p>DT>Frame No>node1>node2> (DT denotes Data from node 1 to node 2)</p> <p>EV>Error Flag>node1>node2> (EV denotes Error Value - i.e If the above frame has error then set the error flag as 1 else set the flag as 0)</p> <p>ACK>POS>node2>node1> (Acknowledgement for that above frame received is sent to node 2 to node 1)</p> <p>Ex:</p> <p>CNT>1>FRAMES>TRANSMIT></p> <p>DT>1>node1>node2></p> <p>EV>0>node1>node2></p> <p>ACK>POS>node2>node1></p> <p>DEL>1>FRAME>DELETED></p>
--	--

Interface Source Code

Interface source code written in C is given using this the user can write only the Transmission Flow Control -Selective Repeat inside the function SelectiveRepeat () using the variables already declared. To view the interface source code, go to

NetSim Installation path / src / Programming/ SelectiveRepeat_TFC.c

To find NetSim's Installation path right click NetSim icon and select

- Open file location in Windows 7
- Open file location in Windows Vista
- Properties --> find target in Windows XP.

Sample Scenarios:

Objective - Send and receive the data using **Transmission Flow Control - Selective Repeat**.

How to Proceed? - The objective can be executed in **NetSim** using the programming exercise available. Under **Programming Menu** select **Transmission Flow Control → Selective Repeat**.

Sample Input - By using the **Input Panel** that is available on the left hand side a **Sample Scenario** is created. The Steps involved in creating are as follows,

- **Sample Mode** should be selected.
- **Select Selective Repeat as Algorithm.**
- **Create a “Data file (.txt)”** and save it on the disk.
- **Select the path of the above created “Data file (.txt)”** which could be **Maximum of 100000 bytes.**
- **Select Bit Error Rate (BER)** as “**10^-5**” from the drop down menu.

1. Then **Run** button need to be clicked.

Output - **Output** for the above **Sample** is as follows,

2. An **Output Table** is obtained when **Run** button is clicked. In the **Output Table, Transmitted frame (Data and Acknowledgement)** and their corresponding **Counts** is obtained. The **Total Count** is also obtained. The table is given below,

Output	
Total data frames to be transmitted -	
Transmitted frame	Count
Data	
Acknowledgement	
Total =	

Note - The “Total data frames to be transmitted” and “Total count” in the **Output** table depends on size of the “.txt” file.

- Once the sample experiment is done, then **Refresh** button can be clicked to create new samples.

Transmission Flow Control - Stop and Wait

Programming Guidelines

This section guides the user to link his/her own code for Stop and Wait to NetSim.

Pre - Conditions

The user program should read the inputted scenario from text file named '**Input**' with extension txt.

The user program after executing the concept should write the required output to a file named '**Output**' with extension txt.

Note: The temporary directory is navigated through the following step.

Run → Type "%temp%" → NetSim → "Input.txt" and "Output.txt"

General Program Flow

The program begins with the *Reading of the Inputs* from the input file **Input.txt**.

Executing the required concept and,

The results of the program should be written into the output file **Output.txt**.

Input File Format	Output File Format
Algorithm=Stop_and_Wait Data_File=C:\Users\P.Sathishkumar\Documents\1 Th.txt> BER=0	Value1>Value2>Value3>Value4> Types: There are five types of formats of writing in the output file. Each format is written in specific condition, the types are explained below. The condition of writing the types is explained in the algorithm. Type1: Value1 - "CNT", Value 2 - output of the slidingcount function Value 3 - "FRAMES" Value 4 - "TRANSMIT" Type2: Value1 - "DT", Value 2 - Frame number, Value 3 - Frame's Source address, Value 4 - Frame's Destination address. Type3:

	<p>Value 1 - "EV", Value 2 - Output of the intro_error function, Value 3 - Frame' Source address, Value 4 - Frame's Destination address.</p> <p>Type4: Value 1 - "ACK", Value 2 - "POS", Value 3 - Acknowledgement frame's Source Address, Value 4 - Acknowledgement frame's Destination Address.</p> <p>Type5: Value 1 - "DEL" Value 2 - count of frames being deleted Value 3 - "FRAME" Value 4 - "DELETED"</p> <p>Note: The above convention to write into the 'Output.Txt' is mandatory. Values in Quotes "" to be written into file 'Output.Txt' as it is including Case. DT>Frame No>node1>node2> (DT denotes Data from node 1 to node 2) EV>Error Flag>node1>node2> (EV denotes Error Value - i.e If the above frame has error then set the error flag as 1 else set the flag as 0) ACK>POS>node2>node1> (Acknowledgement for that above frame received is sent to node 2 to node 1)</p> <p>Ex:</p> <pre>DT>1>node1>node2> EV>0>node1>node2> ACK>POS>node2>node1></pre>
--	--

Interface Source Code

Interface source code written in C is given using this the user can write only the Transmission Flow Control - Stop and Wait inside the function stopandwait() using the variables already declared. To view the interface source code, go to

NetSim Installation path / src / Programming/ StopandWait.c

To find NetSim's Installation path right click NetSim icon and select

- Open file location in Windows 7
- Open file location in Windows Vista
- Properties --> find target in Windows XP.

Sample Scenarios:

Objective - Send and receive the data using **Transmission Flow Control– Stop and wait**.

How to Proceed? - The objective can be executed in **NetSim** using the programming exercise available. Under **Programming Menu** select **Transmission Flow Control → Stop and wait**.

Sample Input - In the Input panel the following steps need to be done,

- **Sample Mode** should be **selected**.
- **Stop and Wait** needs to be **selected** for Algorithm.
- The path of the “**Data file (.txt)**” should be **entered**.
- **Bit Error Rate (BER)** should be **selected**.
- Then **Run** button need to be **clicked**. **Refresh** button can be used if new Inputs have to be given.

Output - The following steps are under gone internally,

- When the **Run** Button is **clicked** the Tool generates the Output in the following format,

Output	
Total data frames to be transmitted	
Transmitted frame	Count
Data	
Acknowledgement	
Total =	

- Once the sample experiment is done, then **Refresh** button can be **clicked** to create New Samples.

NetSim – Real Time

Frames Capture

Description: Monitor the traffic using Frames capture.

To capture & analyze packets in a network we require active systems connected in the network and a network monitoring tool. The network monitoring tool observes the network for data movement, and as the tool recognizes data movement, it will copy the data (even if the data is not destined for the machine running the monitoring software) and analyze it. The analyses of data include the correctness of the data, the data payload, overheads, the time of capture and also the protocols involved. Based on the data analyses the tool calculates the utilization and effective utilization of the network observed.

Procedure:

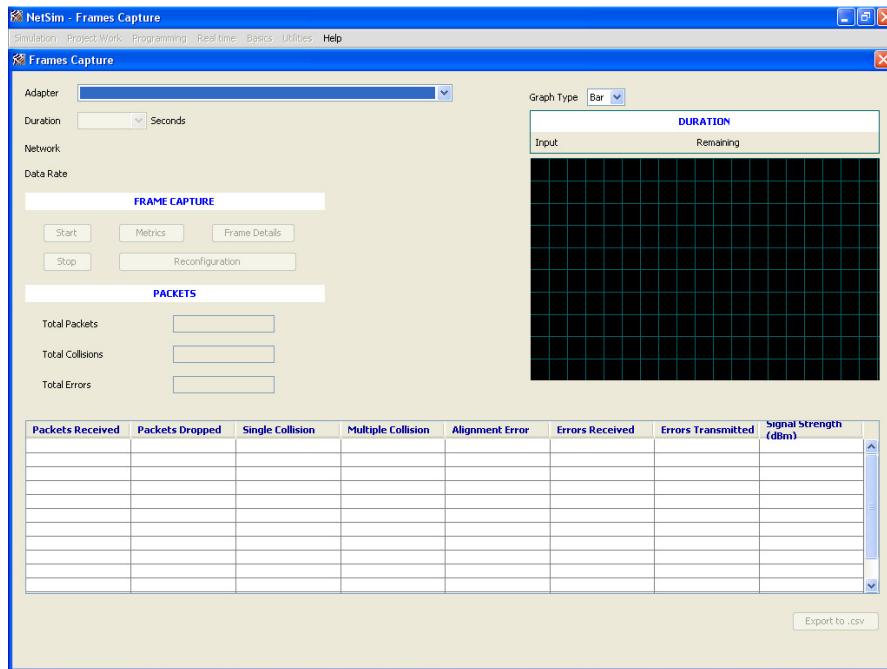
- (1) Connect two systems by using a hub (for testing a single broadcast domain) or via a Switch
- (2) To begin with the experiment, open NetSim



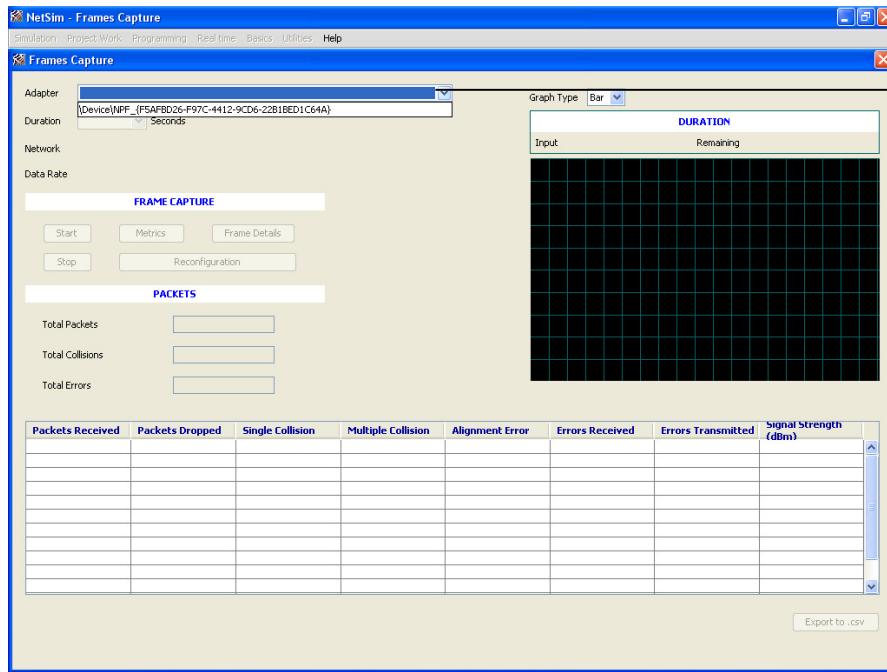
- (3) Click on Real Time, select Frame Capture. The frame capture environment is now open.



The below figure shows the NetSim frame capture environment

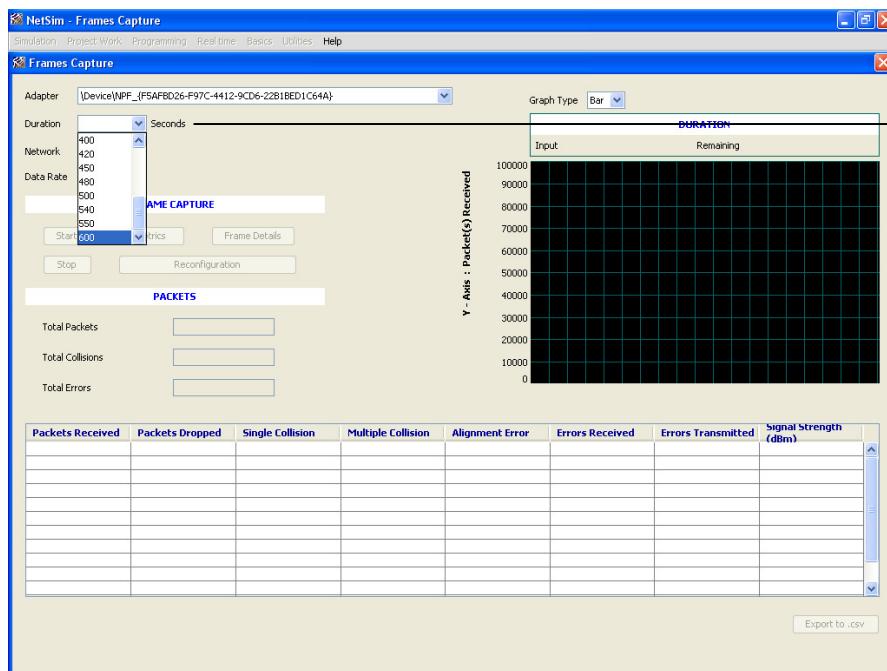


(4) Select the adapter



Select the
network
adaptor

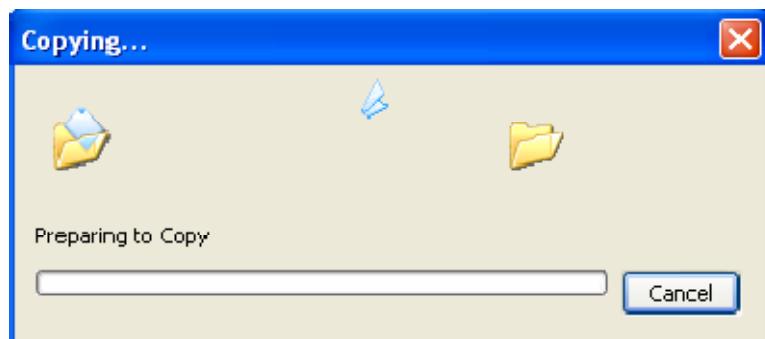
(5) Select how long the traffic is to be monitored



Select the frame capture duration

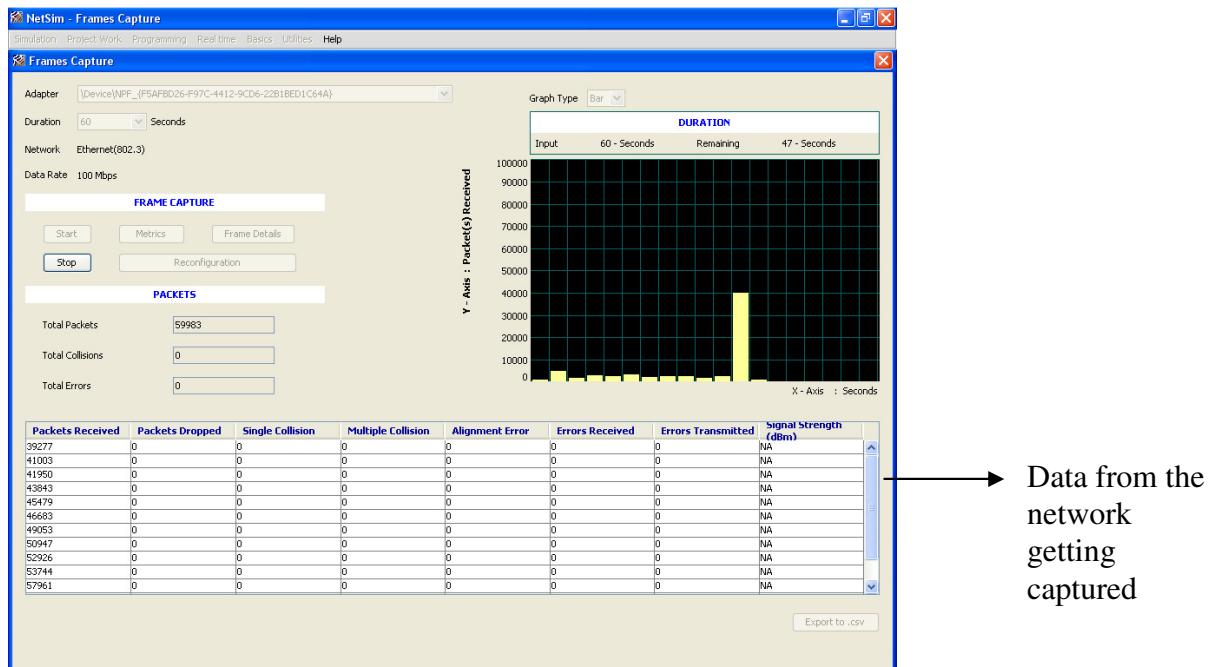
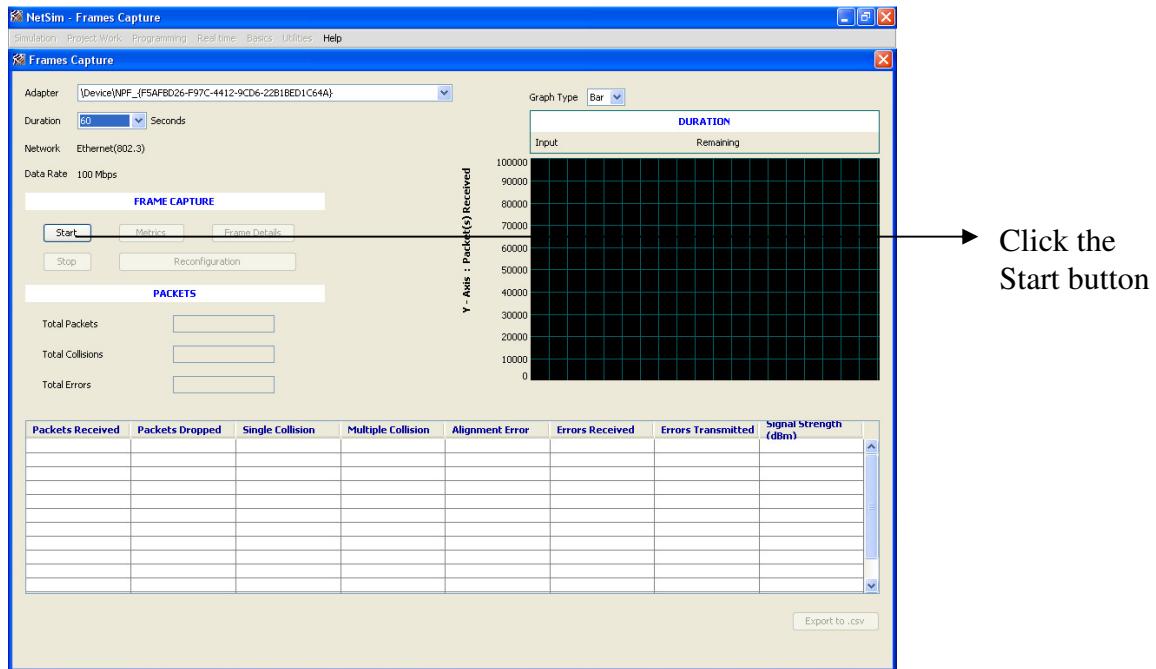
(6) Copy a reasonably large file from one system to the other system.

Note: The file should be large enough for the transfer to take sufficient time (>20 seconds) for the tool to analyze the network accurately.

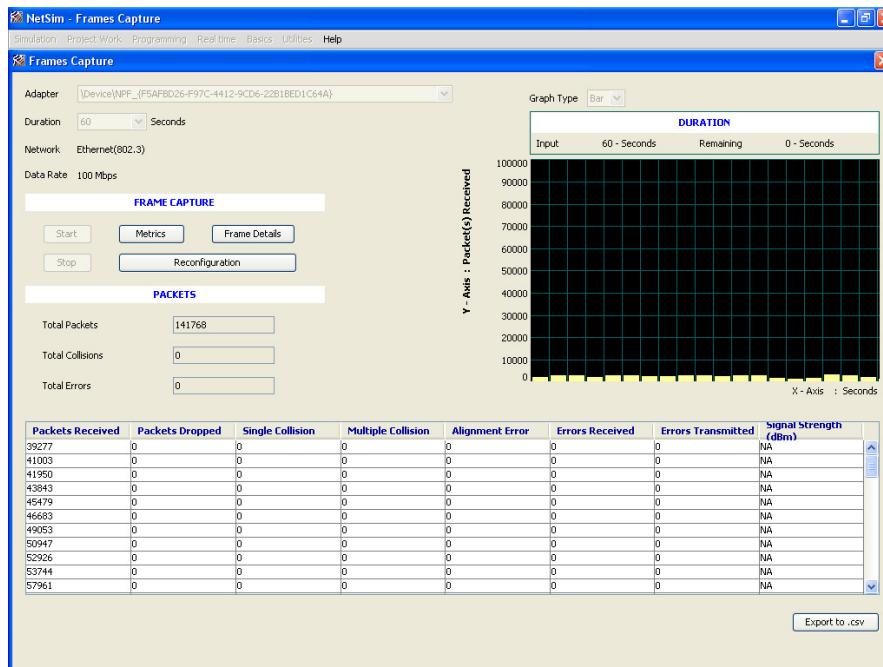


Transfer data from one machine to another

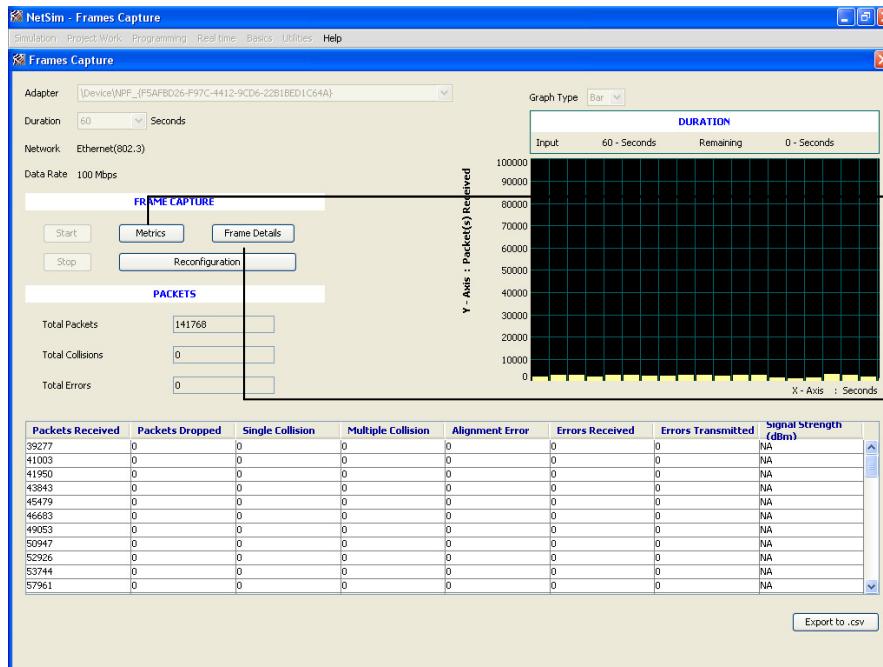
(7) As the file is getting transferred between systems start the traffic monitor by clicking the ‘Start’ button.



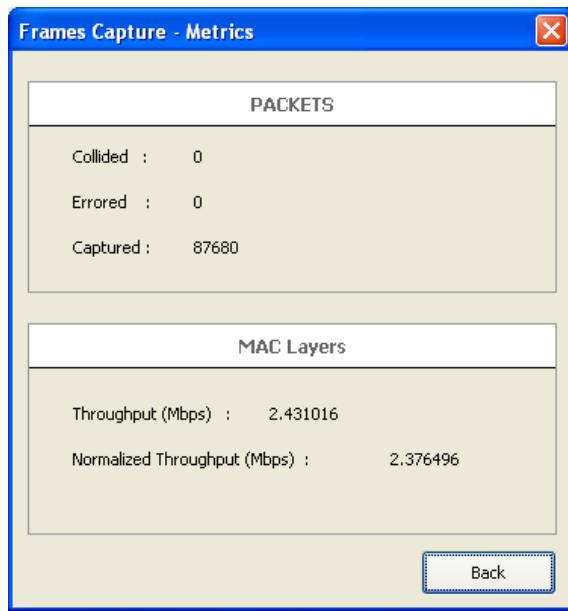
(8) After the duration time end click ‘Export to .csv’ button and note the values.



The details of the frame and the performance of the network can be viewed by clicking ‘Metrics’ and ‘Frame Details’ button.



Metrics Screen:



Frame Details Screen:

CAPTURED FRAME DETAILS									
Captured Seconds	Captured MicroSeconds	Captured Length	Destination MAC Address	Source MAC Address	Network Layer Protocol	Source IP Address	Destination IP Address	Transport Layer Protocol	
139500723	756714	1318	00e04c1100e8	00e1678f7661	IP	192.168.0.152	192.168.0.55	TCP	
139500723	806068	158	00e04c1100e8	00e1678f7661	IP	192.168.0.152	192.168.0.55	TCP	
139500723	806595	142	00e04c1100e8	00e1678f7661	IP	192.168.0.152	192.168.0.55	TCP	
139500723	807284	142	00e04c1100e8	00e1678f7661	IP	192.168.0.152	192.168.0.55	TCP	
139500723	828398	1218	00e04c1100e8	00e1678f7661	IP	192.168.0.152	192.168.0.55	TCP	
139500723	830056	158	00e04c1100e8	00e1678f7661	IP	192.168.0.152	192.168.0.55	TCP	
139500723	830673	142	00e04c1100e8	00e1678f7661	IP	192.168.0.152	192.168.0.55	TCP	
139500723	831254	142	00e04c1100e8	00e1678f7661	IP	192.168.0.152	192.168.0.55	TCP	
139500723	844359	962	00e04c1100e8	00e1678f7661	IP	192.168.0.152	192.168.0.55	TCP	
139500723	846557	158	00e04c1100e8	00e1678f7661	IP	192.168.0.152	192.168.0.55	TCP	
139500723	847166	142	00e04c1100e8	00e1678f7661	IP	192.168.0.152	192.168.0.55	TCP	
139500723	847490	142	00e04c1100e8	00e1678f7661	IP	192.168.0.152	192.168.0.55	TCP	
139500723	849404	654	00e04c1100e8	00e1678f7661	IP	192.168.0.152	192.168.0.55	TCP	
139500723	850567	158	00e04c1100e8	00e1678f7661	IP	192.168.0.152	192.168.0.55	TCP	
139500723	851178	142	00e04c1100e8	00e1678f7661	IP	192.168.0.152	192.168.0.55	TCP	
139500723	851772	142	00e04c1100e8	00e1678f7661	IP	192.168.0.152	192.168.0.55	TCP	
139500723	852531	330	00e04c1100e8	00e1678f7661	IP	192.168.0.152	192.168.0.55	TCP	
139500723	853355	158	00e04c1100e8	00e1678f7661	IP	192.168.0.152	192.168.0.55	TCP	
139500723	853948	142	00e04c1100e8	00e1678f7661	IP	192.168.0.152	192.168.0.55	TCP	
139500723	854554	142	00e04c1100e8	00e1678f7661	IP	192.168.0.152	192.168.0.55	TCP	
139500723	855863	330	00e04c1100e8	00e1678f7661	IP	192.168.0.152	192.168.0.55	TCP	
139500723	856740	158	00e04c1100e8	00e1678f7661	IP	192.168.0.152	192.168.0.55	TCP	
139500723	857341	142	00e04c1100e8	00e1678f7661	IP	192.168.0.152	192.168.0.55	TCP	
139500723	857938	142	00e04c1100e8	00e1678f7661	IP	192.168.0.152	192.168.0.55	TCP	
139500723	858687	330	00e04c1100e8	00e1678f7661	IP	192.168.0.152	192.168.0.55	TCP	
139500723	859521	158	00e04c1100e8	00e1678f7661	IP	192.168.0.152	192.168.0.55	TCP	
139500723	860164	142	00e04c1100e8	00e1678f7661	IP	192.168.0.152	192.168.0.55	TCP	
139500723	860753	142	00e04c1100e8	00e1678f7661	IP	192.168.0.152	192.168.0.55	TCP	
139500723	873240	1226	00e04c1100e8	00e1678f7661	IP	192.168.0.152	192.168.0.55	TCP	
139500723	874731	158	00e04c1100e8	00e1678f7661	IP	192.168.0.152	192.168.0.55	TCP	
139500723	875346	142	00e04c1100e8	00e1678f7661	IP	192.168.0.152	192.168.0.55	TCP	
139500723	877394	142	00e04c1100e8	00e1678f7661	IP	192.168.0.152	192.168.0.55	TCP	
139500723	878198	330	00e04c1100e8	00e1678f7661	IP	192.168.0.152	192.168.0.55	TCP	
Export to .CSV									

(9) Repeat the same procedure by increasing the number of systems in the network, and also increase the number of nodes transferring data in the network.

Inference:

Using Netpatrol one can analyze the throughput of the file transfer in the network. Furthermore you can drill down to each packet and get information relating to packet length, source IP address, Destination IP address, Source MAC Address, Destination MAC Address etc.

NetSim – User Management

User Management

This menu can be used to add new user, delete of existing users, and to set mode of working (Exam/Practice) of the user. It is typically used by professors / administrators to add student users, and then set the user to exam mode during assessments.

Note: As of v7 it is not yet possible to control set / modify user settings from a central location / server. All of the possible user management operations given below will have to be done in each client PC where NetSim is installed. Tetcos is exploring the possibility of providing centralized user management control in NetSim from v8 onwards.

User Management can be reached through **Utilities → User Management**

The Administrator has to enter the correct password and click on OK button to enter into the User Management interface. This is used for verification purpose. The buttons available in User Management window are:

- Add User - This button is used for creating new users. The following fields have to be filled in,
 - Username - Name of the new user needs to be entered in the field provided.
 - Category - By default student is selected.
 - Mode - Either Practice / Exam mode needs to be selected.
 - Click on Accept / Cancel button to accept / cancel the changes.
- Delete User - This button is used for deleting the existing users. The following options have to be filled in,
 - Username - Name of the user needs to be entered in the field provided.
 - Category - By default student is selected.
 - Mode - Either Practice / Exam mode needs to be selected.
 - Click on Accept / Cancel button to accept / cancel the changes.
- Change Mode - This button is used for changing the mode. There are two modes present in the software,

- Practice Mode - Students will be able to get all the help that is associated with the software. Saved experiments can be reused. If this mode is selected, then the user would have access to
 - 1. Under Simulation all the saved experiments in Simulation when the user is in Practice mode can be “opened and reused”, “deleted if they are not required” and “Saved experiment can be used in the Analytics interface”.
 - 2. Under Programming Sample all help documents such “Concepts, Algorithm, Pseudo Code and Flowchart” can be viewed.
 - 3. Basics Menu,
 - 4. NetSim Help.
- Exam Mode - Students will not be able to use the help associated with the software. Also saved experiments in the Practice mode cannot be reused in the Exam mode. If this mode is selected, then the user would have access to,
 - 1. Under Simulation all the saved experiments in Simulation when the user is in Exam mode can be “opened and reused”, “delete if they are not required” and “Saved experiment can be used in the Analytics interface”.
 - 2. Under Programming users cannot view “Concepts, Algorithm, Pseudo Code and Flowchart”.
 - 3. Basics Menu cannot be accessed,
 - 4. NetSim Help can be used.
- Change Password - This button is used to change the password of the user which is selected currently in the left side of the User Management. Administrator can change the password of all the users, whereas other user (other than Administrator) can change his/her password only.
- Experiment Deletion Section: This section is used to delete the saved experiments in NetSim. Administrator can delete the saved experiments of all the users, whereas other user (other than Administrator) can delete his/her saved experiments only.

Note: Experiment Deletion Section can be used to delete the saved experiments in Legacy Networks, Advanced Wireless Networks, BGP Networks, MPLS Networks, Cellular Networks, Wireless Sensor Networks, Personal Area Networks. Saved experiments in Internetworks and Cognitive Radio Networks cannot be deleted through Experiment Deletion Section since these experiments are saved as configuration files in user specified locations.