NetSim®

Accelerate Network R & D

User Manual

A Network Simulation & Emulation Software

By

TETCOS LLP

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

Table of Contents

1 NetSim – Introduction

1.1 Introduction to modeling and simulation of networks

A network simulator[1] enables users to virtually create a network comprising of devices, links, applications etc., and study the behavior and performance of the Network.

Some example applications of network simulators are:

- Protocol performance analysis
- Application modeling and analysis
- Network design and planning
- Research and development of new networking technologies
- Test and Verification

The typical steps followed when simulating any network are:

- **Building the model**: Create a network with devices, links, applications etc.
- **Running the simulation**: Run the discrete event simulation (DES) and log different performance metrics.
- **Visualizing the simulation:** Use the packet animator to view the flow of packets.
- **Analyzing the results**: Examine output performance metrics such as throughput, delay, loss etc. at multiple levels - network, link, queue, application etc.
- **Developing your own protocol / algorithm**: Extend existing algorithms by modifying the simulator's source C code.

1.2 Versions of NetSim – Academic, Standard & Pro

NetSim is used by people from different areas such as industry, defense, and academics to design, simulate, analyze and verify the performance of different networks.

NetSim comes in three versions: **Academic, Standard** and **Pro**. The academic version is used for lab experimentation and teaching. The standard version is used for R & D at educations institutions while NetSim Pro version addresses the needs of defense and industry. The standard and pro versions are available as components in NetSim v13.1 from which users can choose and assemble. A comparison of the features in the three versions are tabulated below Table 1-1.

[1] To be technically precise, NetSim is an end-to-end, full-stack, packet level, continuous time, discrete event network simulator.

Features	Academic	Standard	Pro
Technology Coverage			
Internetworks	Y	Y	Y
Legacy & Cellular Networks	Y	Y	Y
Mobile Adhoc networks	Y	Y	Y
Software Defined Networks	Y	Y	Y
Wireless Sensor Networks	Y	Y	Y
Internet of Things	Y	Y	Y
Cognitive Radio Networks	Y	Y	Y
LTE Networks	Y	Y	Y
5G NR	N	Y	Y
VANET	N	Y	Y
Satellite Communication Networks	N	Y	Y
Performance Reporting Performance metrics available for Network and Sub-networks	Y	Y	Y
Packet Animator Used to animate the packet flow in network	Y	Y	Y
Packet Trace Available in tab ordered .csv format for easy post processing	Y	Y	Y
Event Trace Available in tab ordered .csv format for easy post processing	N	Y	Y
Protocol Library Source Codes with Documentation Protocol C source codes and appropriate header files with extensive documentation	N	Y	Y
External Interfacing Interfacing with SUMO	N	Y	Y
MATLAB	N		
Wireshark	Y		
Integrated debugging Users can write their own code, link their code to NetSim and debug using Visual Studio	N	Y	Y
Plots Allows users to plot the value of a parameter over simulation time	Y	Y	Y
Simulation Scale	100 Nodes	500 Nodes	~ 10,0000 Nodes
Custom Coding and Modeling Support	N	Y	Y
Emulator (Add on) Connect to real hardware running live application	N	Y	Y
TDMA Radio Networks (Add On) TDMA and DTDMA	N	N	Y
Target Users and Segment	Educational (Lab Experimentation)	Educational (Research)	Commercial (Industrial and Defense)

Table 1-1: A comparison of the features of NetSim Academic, Standard and Pro versions

1.3 Components (Technology Libraries) in Pro and Standard versions

Users can choose and assemble components (technology libraries) in NetSim Standard and Pro versions as shown Table 1-2.

Component No	Networks / Protocols Supported	Reference International Standards
Component 1 (Base. Required for all components)	**Internetworks** Ethernet - Fast & Gigabit, ARP, Routing - RIP, OSPF, WLAN - 802.11 a / b / g /p / n / ac & e, Propagation models - HATA Urban / Suburban, COST 231 HATA urban / Suburban, Indoor Home / Office / Factory, Friis Free Space, Log Distance. Shadowing - Constant, Lognormal. Fading - Rayleigh, Nakagami IPv4, Firewalls, Queuing - Round Robin, FIFO, Priority, WFQ, TCP, - Old Tahoe, Tahoe, Reno, New Reno, BIC, CUBIC, Window Scaling, SACK UDP **Common Modules** Traffic Generator: Voice, Video, FTP, Database, HTTP, Email, P2P, Custom, CBR. Virtual Network Stack, Simulation Kernel, Command Line Interface Command Line Interpreter Metrics Engine with packet and event trace Plot Generator Packet Animator, Packet Encryption External Interfaces: MATLAB, Wireshark	IEEE 802.3 IEEE 802.11 a/b/g/n/ac/p/e RFCs 2453, 2328, 826, 793, 2001 and 768
Component 2	**Legacy & Cellular Networks** Aloha – (Pure & Slotted) GSM CDMA	3GPP, ETSI, IMT-MC, IS-95 A/B, IxRTT, 1x-EV-Do, 3xRTT
Component 3	**Advanced Routing** Multicast Routing - IGMP, PIM, Access Control Lists, Detailed Layer 3 switch mode, Virtual LAN (VLAN), Public IP, Network Address Translation (NAT)	IETF RFC's 1771 & 3121
Component 4	**Mobile Adhoc Networks** MANET - DSR, AODV, OLSR, ZRP	IETF RFC 4728, 3561, 3626
Component 5	**Software Defined Network (SDN)**	Based on Open Flow v1.3
Component 6 (Component 4 required)	**Internet of things (IOT)** with RPL protocol Wireless Sensor Networks (WSN)	IEEE 802.15.4 MAC, MANET in L3 RFC 6550
Component 7	**Cognitive Radio Networks** WRAN	IEEE 802.22

Component 8	**Long-Term Evolution Networks:** LTE	3GPP
Component 9 (Component 4 required)	**VANETs:** IEEE 1609 WAVE, Basic Safety Message (BSM) protocol per J2735 DSRC, Interface with SUMO for road traffic simulation	IEEE 1609
Component 10 (Components 8 required)	**5G NR :**3GPP 38 Series. Full Stack covering SDAP, PDCP, RLC – UM, TM, MAC, PHY – FR1 and FR2, mmWave propagation.	3GPP 38.xxx
Component 11 (Component 3 required)	**Satellite Communication Networks:** Geo Stationary Satellite. Forward link TDMA in Ku Band and Return link MF-TDMA in Ka band per DVB S2. Markov Loo Fading model. Device models for Satellite, Satellite Gateway, and Satellite User Terminals	DVB S2
TDMA Radio Networks Add on (Pro version only)	**TDMA Radio Networks** TDMA Link 16, Dynamic TDMA, Frequencies – HF, VHF, UHF Bands, Frequency Hopping	----
Network Emulator Add On	**Network Emulator** Connect real hardware running live applications to NetSim Simulator. IP based, data plane, flow through emulator.	----

Table 1-2: Different Components (Technology Libraries) in Pro and Standard versions of NetSim

2 Installation and License Server Set-up

2.1 System Requirements

2.1.1 NetSim Client (installs locally)

- Hardware: i3 equivalent or above, RAM: 4 GB (Min). 8GB Recommended.
- Monitor resolution: Min - 1024*768, Max - 1920*1080. Optional Scale and layout setting: 100%
- Operating system: 64 bit. Win 8 or Win 10, Language English
- Software: MS Office, Adobe Reader
- Development Tools: Visual Studio
  - NetSim v8 / v8.1 / v8.3 / v9 / v9.1: Microsoft Visual Studio 2010 (or higher)
  - NetSim v10 / v11 / v11.1: Microsoft Visual Studio 2015 (or higher)
  - NetSim v12 / v12.1 / v12.2 / v13.0 / v13.1: Microsoft Visual Studio 2019 (or higher)

Visual Studio Community edition (or higher) is required for writing and debugging custom code.

2.1.2 License Server

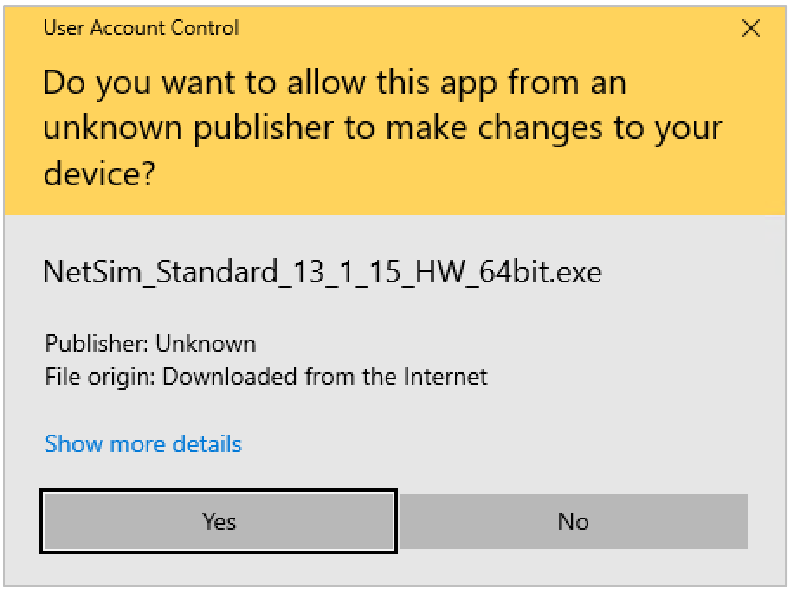
This is applicable when running Host-ID/ Dongle locked floating licenses, and not applicable for node locked licenses.

Any one system will have to be made as the license server, and it is to this PC that the license is locked, either via its MAC ID or via a dongle. The dongle is a USB device which controls the licensing. The system (hardware / OS) requirements are same as that applicable for NetSim clients. USB Port is required for connecting and running the dongle. Client systems should be able to communicate with license server through the network.

2.2 Installing NetSim

Install 64-bit build of NetSim. The starting window will show (i) Version type (Pro, Standard, Academic), (ii) Version Number and build number (Eg: 13.1.15) followed by (iii) Currently supports 64bit in v13.1.

For example, you will see **NetSim_Standard_13_1_15_HW_64bit.exe** for a Standard version install. Double click on the setup file. Click on **Yes** button to install the software.


User Account Control

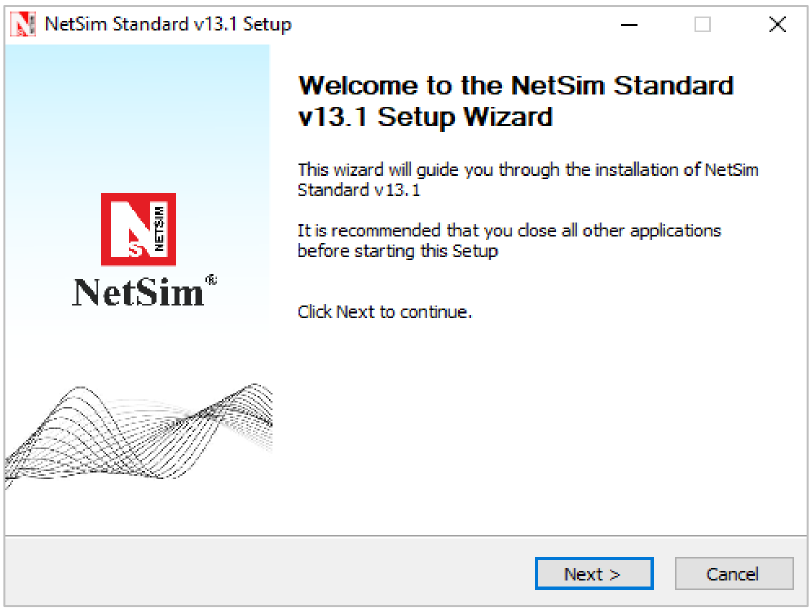
Do you want to allow this app from an unknown publisher to make changes to your device?

NetSim_Standard_13_1_15_HW_64bit.exe

Publisher: Unknown
File origin: Downloaded from the Internet

Figure 2-1: User Account Control message window appears and select Yes button.

Setup prepares the installation wizard and software installation begins with a **Welcome Screen**. Click on **Next** button to continue with the installation.


NetSim Standard v13.1 Setup

Welcome to the NetSim Standard v13.1 Setup Wizard

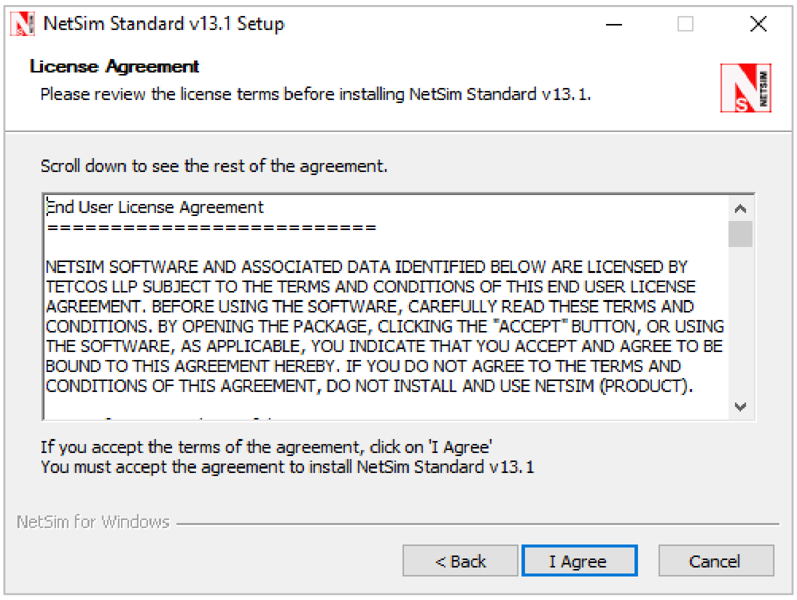
This wizard will guide you through the installation of NetSim Standard v13.1

It is recommended that you close all other applications before starting this Setup

Click Next to continue.

Figure 2-2: Select Next button to continue with the installation

License agreement will be displayed. Read the agreement carefully, scroll down to read the complete license agreement. Click on **I Agree** button else quit the setup by clicking **Cancel** button.


NetSim Standard v13.1 Setup

License Agreement

Please review the license terms before installing NetSim Standard v13.1.

Scroll down to see the rest of the agreement.

```
End User License Agreement
==========================

NETSIM SOFTWARE AND ASSOCIATED DATA IDENTIFIED BELOW ARE LICENSED BY
TETCOS LLP SUBJECT TO THE TERMS AND CONDITIONS OF THIS END USER LICENSE
AGREEMENT. BEFORE USING THE SOFTWARE, CAREFULLY READ THESE TERMS AND
CONDITIONS. BY OPENING THE PACKAGE, CLICKING THE "ACCEPT" BUTTON, OR USING
THE SOFTWARE, AS APPLICABLE, YOU INDICATE THAT YOU ACCEPT AND AGREE TO BE
BOUND TO THIS AGREEMENT HEREBY. IF YOU DO NOT AGREE TO THE TERMS AND
CONDITIONS OF THIS AGREEMENT, DO NOT INSTALL AND USE NETSIM (PRODUCT).
```

If you accept the terms of the agreement, click on 'I Agree'
You must accept the agreement to install NetSim Standard v13.1

Figure 2-3: Select I Agree button

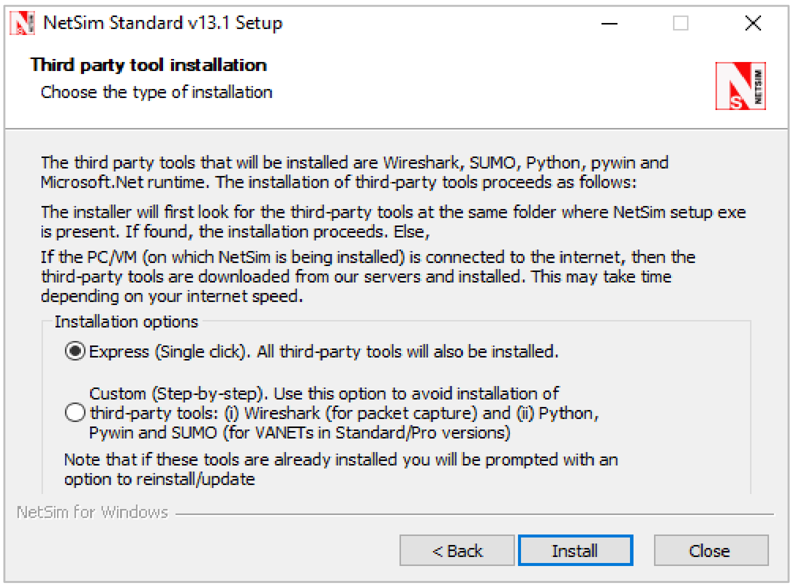
If you agree with the license agreement, you will be prompted to select either one of the installation options, Express (Single-click installation) or Custom (Step-by-Step installation).

Express Installation will install the third-party tools silently along with NetSim without displaying any prompts for the user.

Custom Installation is a step-by-step approach in which a user will be prompted to carry out the installation process and the same applies to the installation of the third-party tools which happens alongside with NetSim.

Both the installation methods are explained below:

2.2.1 Express Installation


NetSim Standard v13.1 Setup

Third party tool installation

Choose the type of installation

The third party tools that will be installed are Wireshark, SUMO, Python, pywin and Microsoft.Net runtime. The installation of third-party tools proceeds as follows:

The installer will first look for the third-party tools at the same folder where NetSim setup exe is present. If found, the installation proceeds. Else,

If the PC/VM (on which NetSim is being installed) is connected to the internet, then the third-party tools are downloaded from our servers and installed. This may take time depending on your internet speed.

Installation options

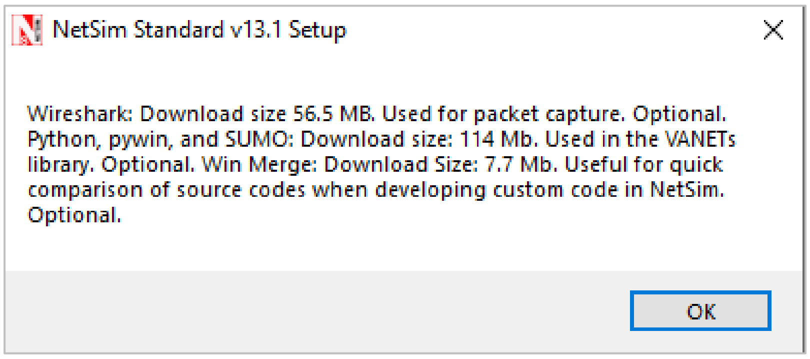
[✓ Express (Single click). All third-party tools will also be installed.]

Custom (Step-by-step). Use this option to avoid installation of third-party tools: (i) Wireshark (for packet capture) and (ii) Python, Pywin and SUMO (for VANETs in Standard/Pro versions)

Note that if these tools are already installed you will be prompted with an option to reinstall/update

Figure 2-4: Select Express (Single click) radio button and click on install

NetSim installation starts, and users can see that the third-party tools download information window click on OK to proceed with the installation.



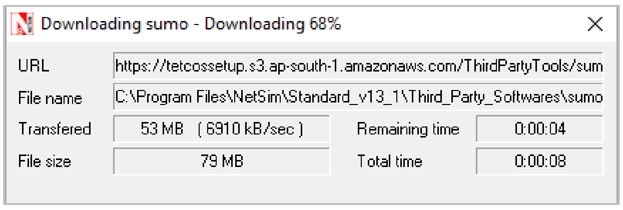
NetSim Standard v13.1 Setup

Wireshark: Download size 56.5 MB. Used for packet capture. Optional. Python, pywin, and SUMO: Download size: 114 Mb. Used in the VANETs library. Optional. Win Merge: Download Size: 7.7 Mb. Useful for quick comparison of source codes when developing custom code in NetSim. Optional.

Figure 2-5: Click on the OK button to proceed installation process of NetSim

The third-party tools like Wireshark, SUMO, python, Winmerge, pywin, and Microsoft.Net will begin to install. Before that, the installer will look for the third-party tools at the same folder where NetSim.exe is present if found, the next step of installation proceeds.

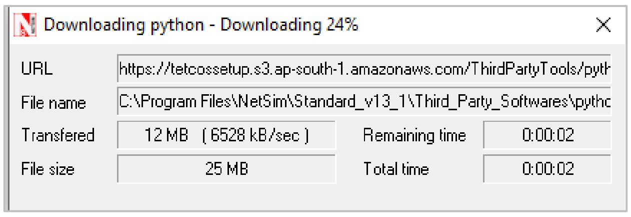
Else, the third-party tools will get downloaded from our NetSim servers and installed if the PC/VM is connected to the Internet.



Downloading sumo - Downloading 68%

URL	https://tetcossetup.s3.ap-south-1.amazonaws.com/ThirdPartyTools/sum		
File name	C:\Program Files\NetSim\Standard_v13_1\Third_Party_Softwares\sumo		
Transfered	53 MB (6910 kB/sec)	Remaining time	0:00:04
File size	79 MB	Total time	0:00:08

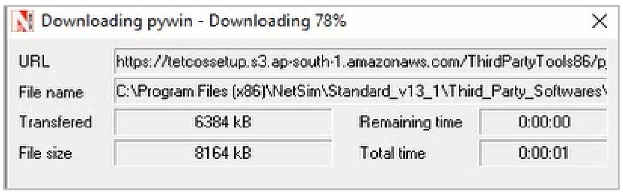
Figure 2-6: Sumo is being downloaded



Downloading python - Downloading 24%

URL	https://tetcossetup.s3.ap-south-1.amazonaws.com/ThirdPartyTools/pyth		
File name	C:\Program Files\NetSim\Standard_v13_1\Third_Party_Softwares\pytho		
Transfered	12 MB (6528 kB/sec)	Remaining time	0:00:02
File size	25 MB	Total time	0:00:02

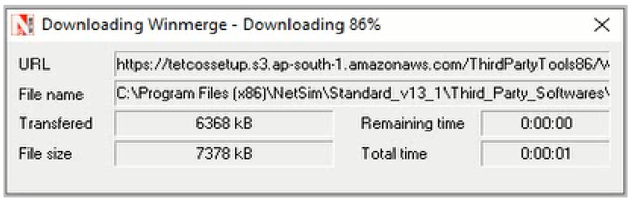
Figure 2-7: python is being downloaded



Downloading pywin - Downloading 78%

URL	https://tetcossetup.s3.ap-south-1.amazonaws.com/ThirdPartyTools86/p		
File name	C:\Program Files (x86)\NetSim\Standard_v13_1\Third_Party_Softwares\		
Transfered	6384 kB	Remaining time	0:00:00
File size	8164 kB	Total time	0:00:01

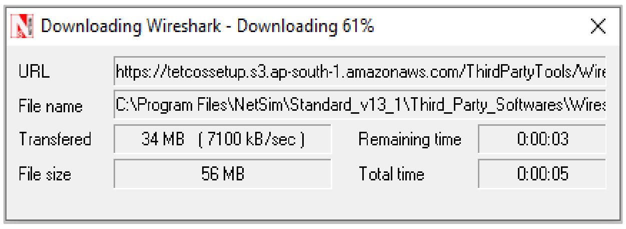
Figure 2-8: pywin is being downloaded



Downloading Winmerge - Downloading 86%

URL	https://tetcossetup.s3.ap-south-1.amazonaws.com/ThirdPartyTools86/V		
File name	C:\Program Files (x86)\NetSim\Standard_v13_1\Third_Party_Softwares\		
Transfered	6368 kB	Remaining time	0:00:00
File size	7378 kB	Total time	0:00:01

Figure 2-9: Winmerge is being downloaded



Downloading Wireshark - Downloading 61%

URL	https://tetcossetup.s3.ap-south-1.amazonaws.com/ThirdPartyTools/Wire		
File name	C:\Program Files\NetSim\Standard_v13_1\Third_Party_Softwares\Wires		
Transfered	34 MB (7100 kB/sec)	Remaining time	0:00:03
File size	56 MB	Total time	0:00:05

Figure 2-10: Wireshark is being downloaded

NetSim installation starts, and users can see that the third-party tools get installed one by one.

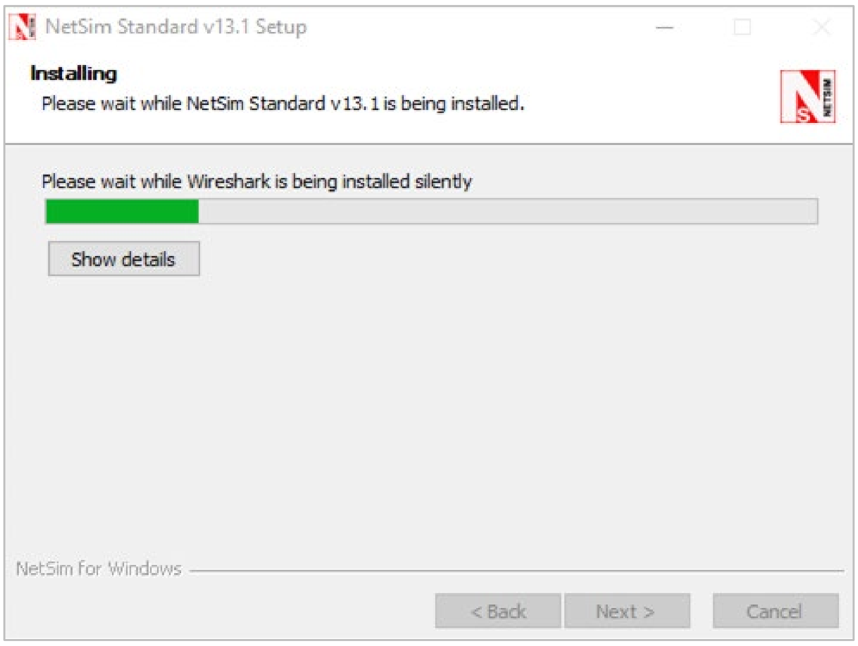

NetSim Standard v13.1 Setup
Installing
Please wait while NetSim Standard v13.1 is being installed.
Please wait while Wireshark is being installed silently
Show details
NetSim for Windows
< Back
Next >
Cancel

Figure 2-11: Wireshark gets installed silently

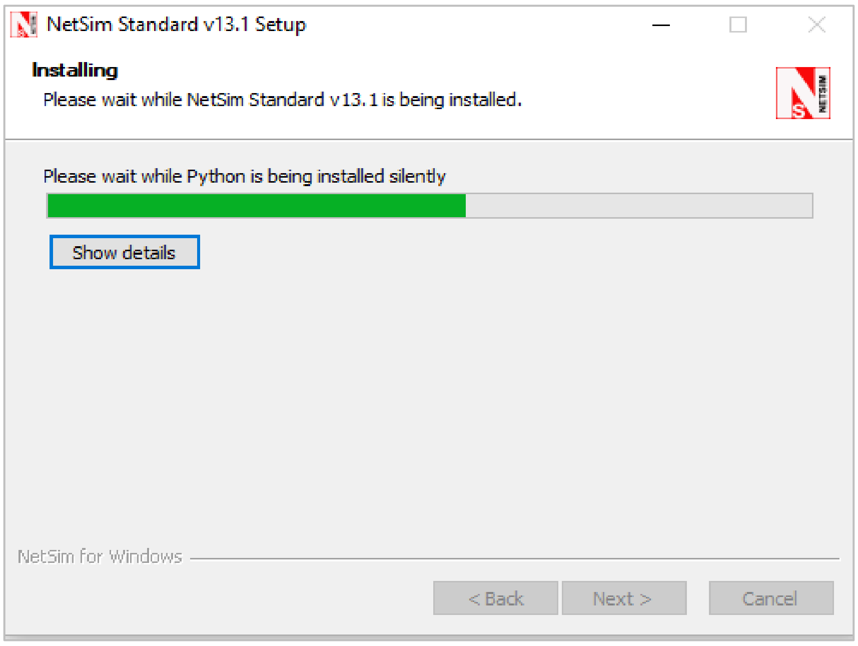
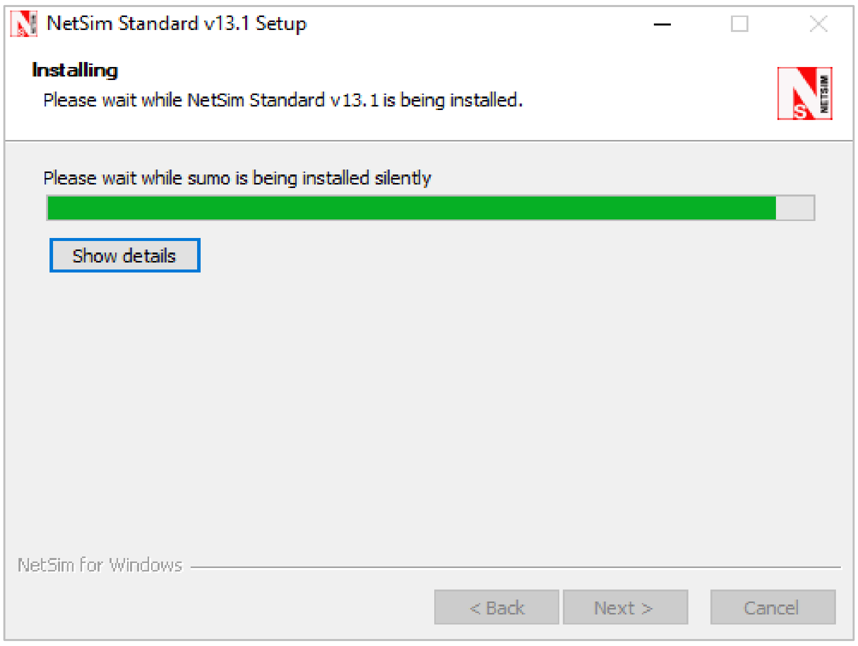

NetSim Standard v13.1 Setup
Installing
Please wait while NetSim Standard v13.1 is being installed.
Please wait while Python is being installed silently
Show details
NetSim for Windows
< Back
Next >
Cancel

Figure 2-12: Python gets installed silently


NetSim Standard v13.1 Setup

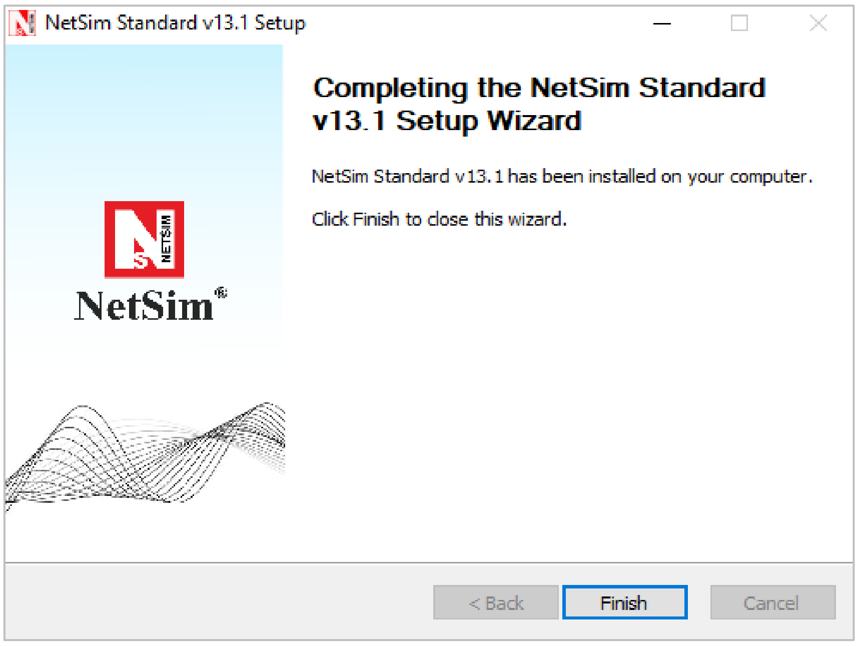
Installing

Please wait while NetSim Standard v13.1 is being installed.

Please wait while sumo is being installed silently

Figure 2-13: Sumo gets installed silently

After the third-party installations, NetSim installation proceeds. Once it is completed, NetSim-complete setup wizard appears as shown below. Click on **Finish** button to complete the installation process of NetSim.


NetSim Standard v13.1 Setup

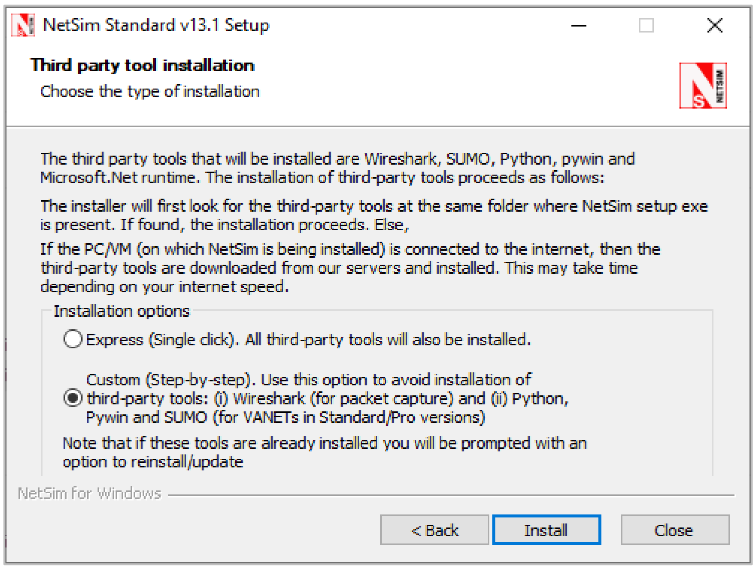
Completing the NetSim Standard v13.1 Setup Wizard

NetSim Standard v13.1 has been installed on your computer.

Click Finish to close this wizard.

Figure 2-14: Select Finish button to complete the installation process of NetSim.

2.2.2 Custom (Step-by-step) installation


NetSim Standard v13.1 Setup

Third party tool installation

Choose the type of installation

The third party tools that will be installed are Wireshark, SUMO, Python, pywin and Microsoft.Net runtime. The installation of third-party tools proceeds as follows:

The installer will first look for the third-party tools at the same folder where NetSim setup exe is present. If found, the installation proceeds. Else,
If the PC/VM (on which NetSim is being installed) is connected to the internet, then the third-party tools are downloaded from our servers and installed. This may take time depending on your internet speed.

Installation options

○ Express (Single click). All third-party tools will also be installed.

● Custom (Step-by-step). Use this option to avoid installation of third-party tools: (i) Wireshark (for packet capture) and (ii) Python, Pywin and SUMO (for VANETs in Standard/Pro versions)

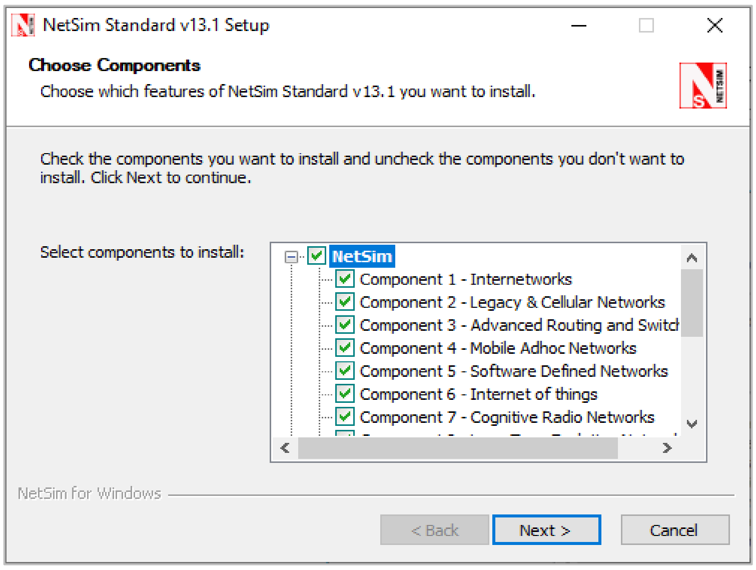
Note that if these tools are already installed you will be prompted with an option to reinstall/update

NetSim for Windows

Figure 2-15: Select Custom Radio button

Now the user will be prompted to select the components to be installed. The list of components is available for selection and assembly only in the Standard and Pro versions of NetSim. NetSim Academic version is available as a single package.

Note: *In Standard and Pro Versions of NetSim, the Choose Components screen will display only those components for which the licenses are obtained by the user. Also, Network Emulator and Real Time Protocol are available as Add-On along with NetSim.*


NetSim Standard v13.1 Setup

Choose Components

Choose which features of NetSim Standard v13.1 you want to install.

Check the components you want to install and uncheck the components you don't want to install. Click Next to continue.

Select components to install:

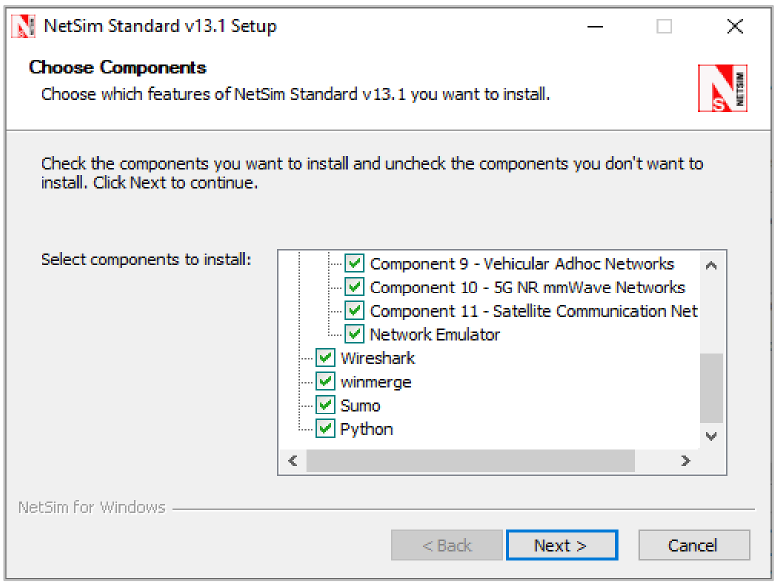
- ✓ NetSim
  - ✓ Component 1 - Internetworks
  - ✓ Component 2 - Legacy & Cellular Networks
  - ✓ Component 3 - Advanced Routing and Switch
  - ✓ Component 4 - Mobile Adhoc Networks
  - ✓ Component 5 - Software Defined Networks
  - ✓ Component 6 - Internet of things
  - ✓ Component 7 - Cognitive Radio Networks

NetSim for Windows

Figure 2-16: list of components is available for selection and assembly only in the Standard and Pro versions

Note: *Select all the supporting applications for complete installation of the software as shown below:*

Click on the **Next** button.



NetSim Standard v13.1 Setup

Choose Components

Choose which features of NetSim Standard v13.1 you want to install.

Check the components you want to install and uncheck the components you don't want to install. Click Next to continue.

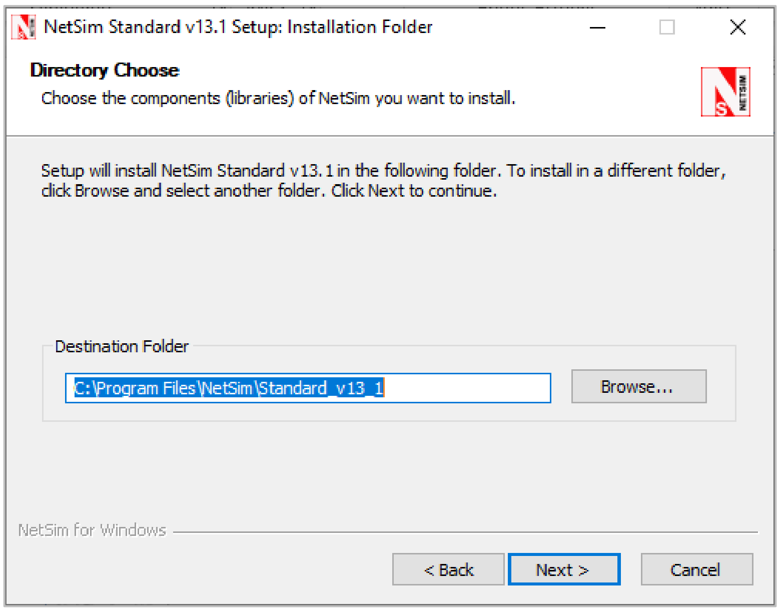
Select components to install:

- ✓ Component 9 - Vehicular Adhoc Networks
- ✓ Component 10 - 5G NR mmWave Networks
- ✓ Component 11 - Satellite Communication Net
- ✓ Network Emulator
- ✓ Wireshark
- ✓ winmerge
- ✓ Sumo
- ✓ Python

Figure 2-17: list of third-party tools

Note: *Sumo, Python and Winmerge comes only as a part of Standard and Pro Version Install.*

In the next screen, you will be requested to enter the installation path. Select the path in which the software needs to be installed and click on **Next** button.



NetSim Standard v13.1 Setup: Installation Folder

Directory Choose

Choose the components (libraries) of NetSim you want to install.

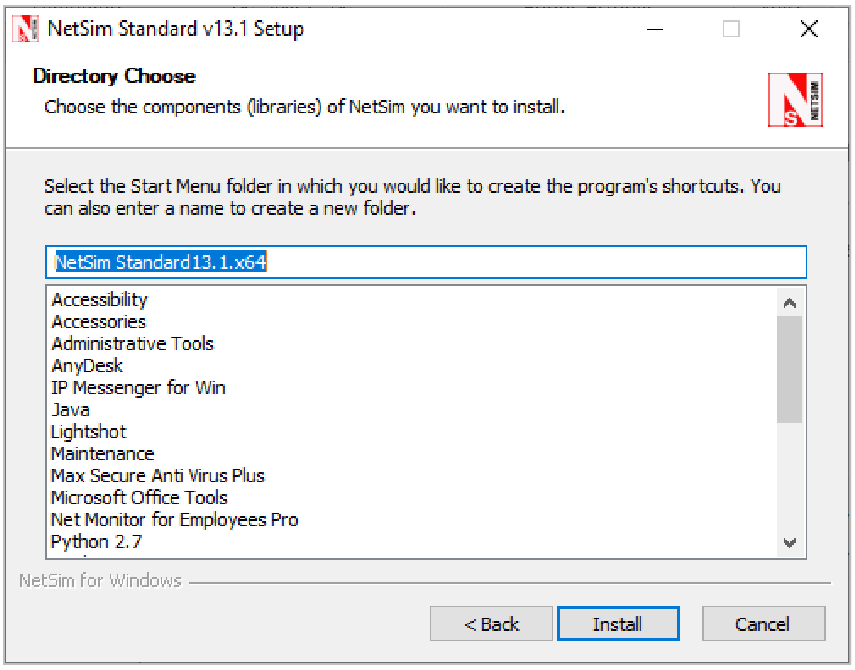
Setup will install NetSim Standard v13.1 in the following folder. To install in a different folder, click Browse and select another folder. Click Next to continue.

Destination Folder

C:\Program Files\NetSim\Standard_v13_1

Figure 2-18: NetSim installation directory path

In the next screen, you will be requested to enter the Start Menu folder name. By default, it shows **NetSim Standard** for Standard version install of NetSim. Click on the **Install** button to start the installation.



NetSim Standard v13.1 Setup

Directory Choose

Choose the components (libraries) of NetSim you want to install.

Select the Start Menu folder in which you would like to create the program's shortcuts. You can also enter a name to create a new folder.

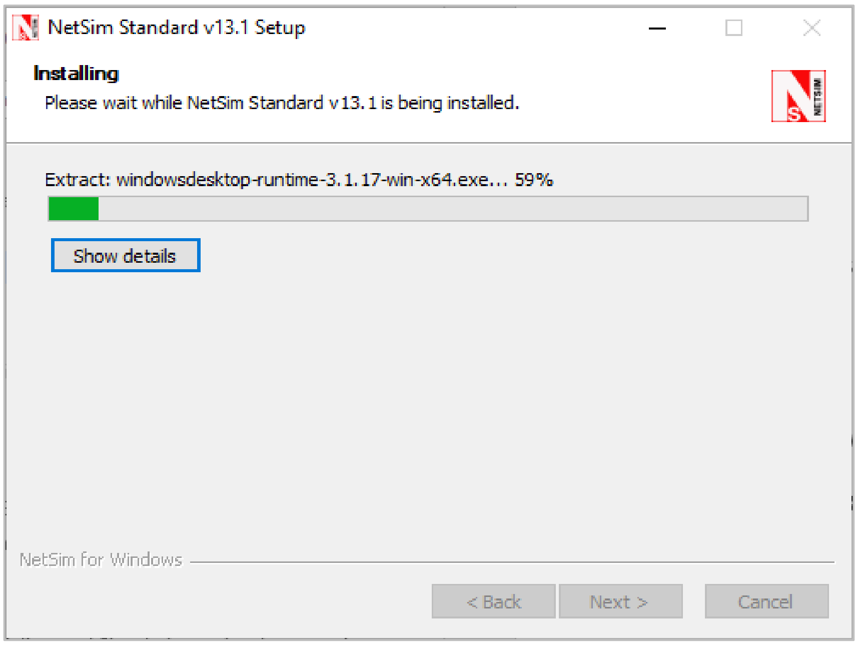
NetSim Standard13.1.x64

Accessibility
Accessories
Administrative Tools
AnyDesk
IP Messenger for Win
Java
Lightshot
Maintenance
Max Secure Anti Virus Plus
Microsoft Office Tools
Net Monitor for Employees Pro
Python 2.7

NetSim for Windows

Figure 2-19: Start Menu folder name

The installation process begins.



NetSim Standard v13.1 Setup

Installing

Please wait while NetSim Standard v13.1 is being installed.

Extract: windowsdesktop-runtime-3.1.17-win-x64.exe... 59%

NetSim for Windows

Figure 2-20: NetSim Standard v13.1 being installed.

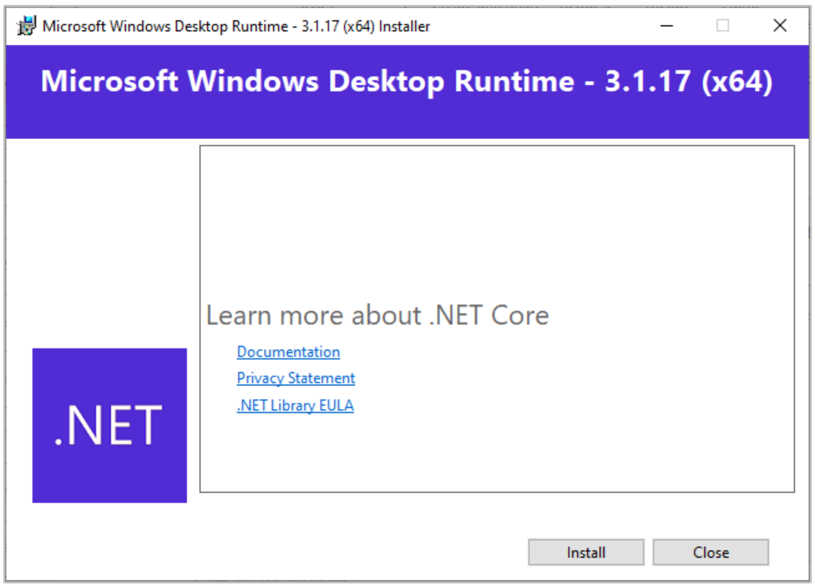
After the installation of required NetSim files, the installation of **third-party tools** begins.

For NetSim Academic Version, Npcap and Wireshark will be installed.

For NetSim Standard and Pro Versions, along with WinPcap and Wireshark installation, Dot net, Sumo, Python installation will start automatically. (*If not deselected during 3rd party software selection*)

If the PC/VM is connected to the Internet third party tools will get downloaded from our NetSim servers (If the third-party tools are not found in folder where NetSim.exe is present) and proceeds with installation.

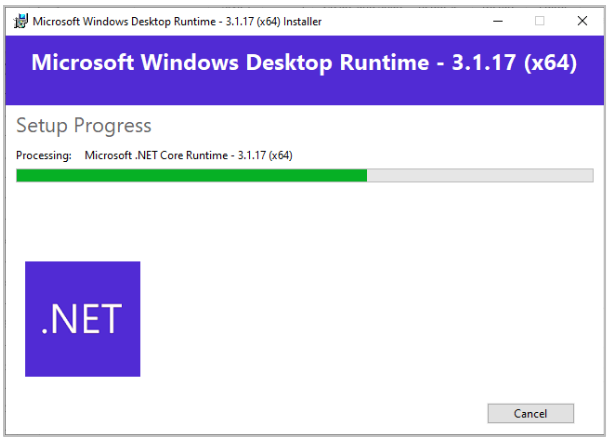
Click on **Install** button to start Dot NET (.NET) installation



Microsoft Windows Desktop Runtime - 3.1.17 (x64) Installer
Microsoft Windows Desktop Runtime - 3.1.17 (x64)
Learn more about .NET Core
Documentation
Privacy Statement
.NET Library EULA
.NET
Install
Close

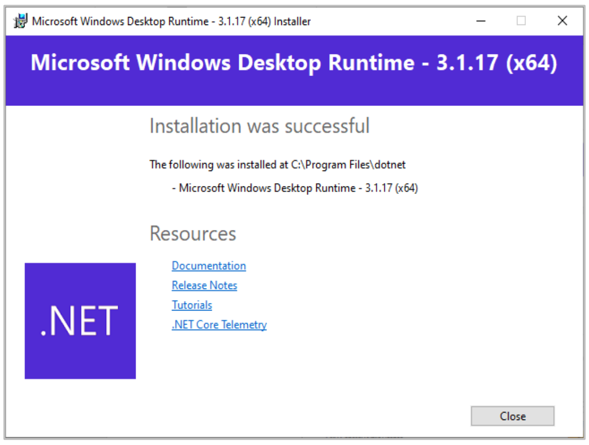
Figure 2-21: Select install button to install Dot NET (.NET)

Installation process begins



Microsoft Windows Desktop Runtime - 3.1.17 (x64) Installer
Microsoft Windows Desktop Runtime - 3.1.17 (x64)
Setup Progress
Processing: Microsoft .NET Core Runtime - 3.1.17 (x64)
.NET
Cancel

Figure 2-22: Dot NET (.NET) installation begins



Microsoft Windows Desktop Runtime - 3.1.17 (x64) Installer
Microsoft Windows Desktop Runtime - 3.1.17 (x64)
Installation was successful

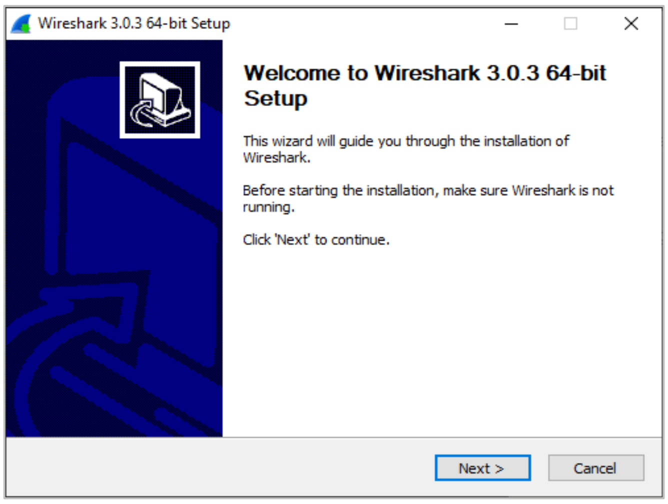
The following was installed at C:\Program Files\dotnet

- Microsoft Windows Desktop Runtime - 3.1.17 (x64)

Resources
Documentation
Release Notes
Tutorials
.NET Core Telemetry
.NET
Close

Figure 2-23: Dot NET (.NET) installation successfully completed

After the successful installation of Dot NET (.NET) and click on close button then Wireshark installation window appears. Click on **Next** button to begin


Wireshark 3.0.3 64-bit Setup

Welcome to Wireshark 3.0.3 64-bit Setup

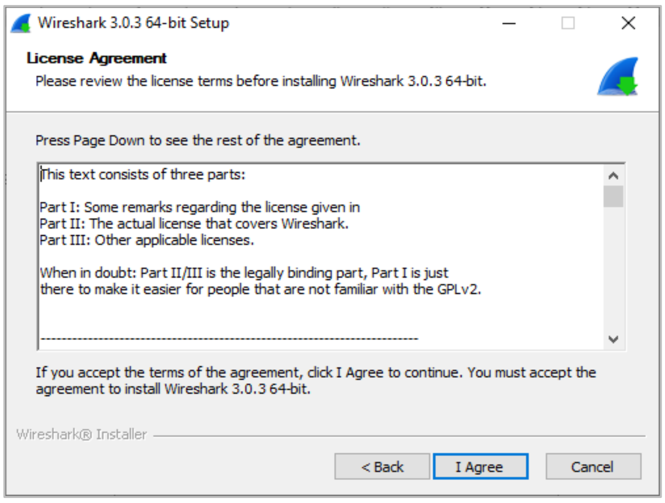
This wizard will guide you through the installation of Wireshark.

Before starting the installation, make sure Wireshark is not running.

Click 'Next' to continue.

Figure 2-24: Select Next button to start Wireshark installation

Wireshark **License Agreement** appears. Click on **I Agree** button.


Wireshark 3.0.3 64-bit Setup

License Agreement

Please review the license terms before installing Wireshark 3.0.3 64-bit.

Press Page Down to see the rest of the agreement.

This text consists of three parts:

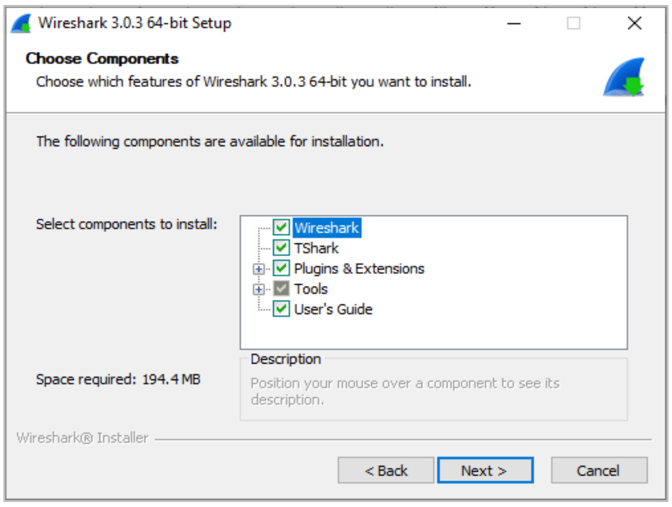
Part I: Some remarks regarding the license given in
Part II: The actual license that covers Wireshark.
Part III: Other applicable licenses.

When in doubt: Part II/III is the legally binding part, Part I is just
there to make it easier for people that are not familiar with the GPLv2.

If you accept the terms of the agreement, click I Agree to continue. You must accept the agreement to install Wireshark 3.0.3 64-bit.

Figure 2-25: Wireshark License Agreement window

Make sure that all the components are selected and click on **Next** button.


Wireshark 3.0.3 64-bit Setup

Choose Components

Choose which features of Wireshark 3.0.3 64-bit you want to install.

The following components are available for installation.

Select components to install:

- ✓ Wireshark
- ✓ TShark
- ✓ Plugins & Extensions
- ✓ Tools
- ✓ User's Guide

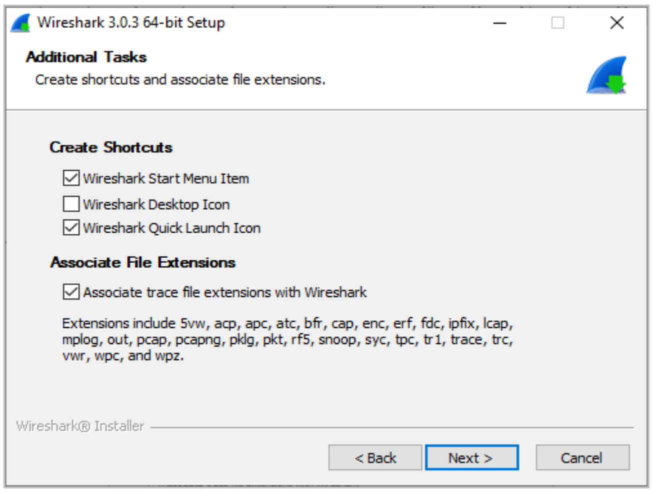
Space required: 194.4 MB

Description

Position your mouse over a component to see its description.

Figure 2-26: Choose Wireshark features

Click on **Next** button.



Wireshark 3.0.3 64-bit Setup

Additional Tasks

Create shortcuts and associate file extensions.

Create Shortcuts

- [✓] Wireshark Start Menu Item
- [ ] Wireshark Desktop Icon
- [✓] Wireshark Quick Launch Icon

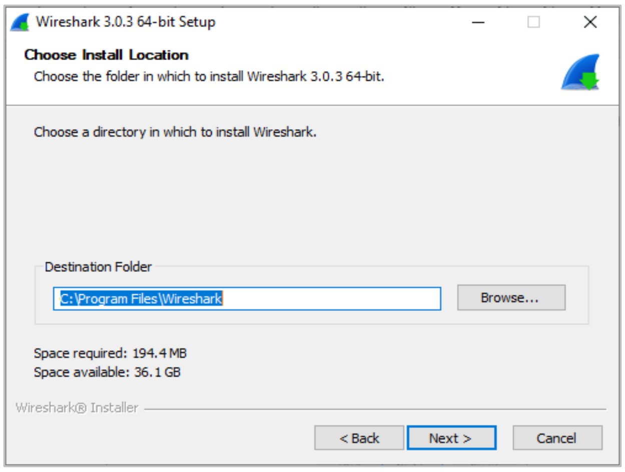
Associate File Extensions

- [✓] Associate trace file extensions with Wireshark

Extensions include 5vw, acp, apc, atc, bfr, cap, enc, erf, fdc, ipfix, lcap, mplog, out, pcap, pcapng, pklg, pkt, rf5, snoop, syc, tpc, tr1, trace, trc, vwr, wpc, and wpz.

Figure 2-27: Select Next button

Select the path in which Wireshark needs to be installed and click on **Next** button.



Wireshark 3.0.3 64-bit Setup

Choose Install Location

Choose the folder in which to install Wireshark 3.0.3 64-bit.

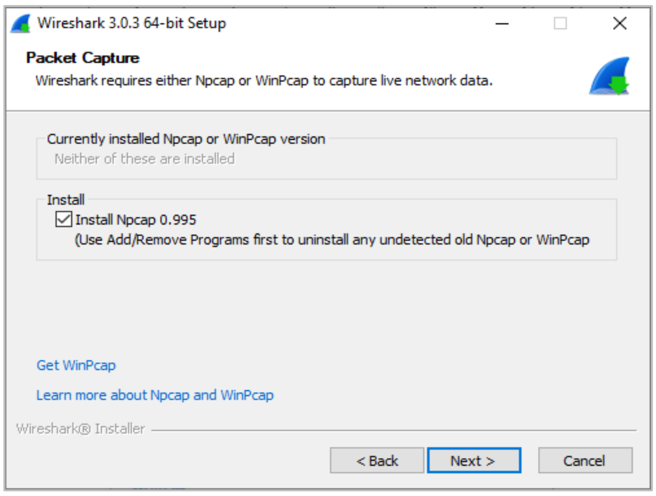
Choose a directory in which to install Wireshark.

Destination Folder: C:\Program Files\Wireshark

Space required: 194.4 MB
Space available: 36.1 GB

Figure 2-28: Wireshark installation directory path

Select **Install Npcap 0.995** and click on **Next** button.



Wireshark 3.0.3 64-bit Setup

Packet Capture

Wireshark requires either Npcap or WinPcap to capture live network data.

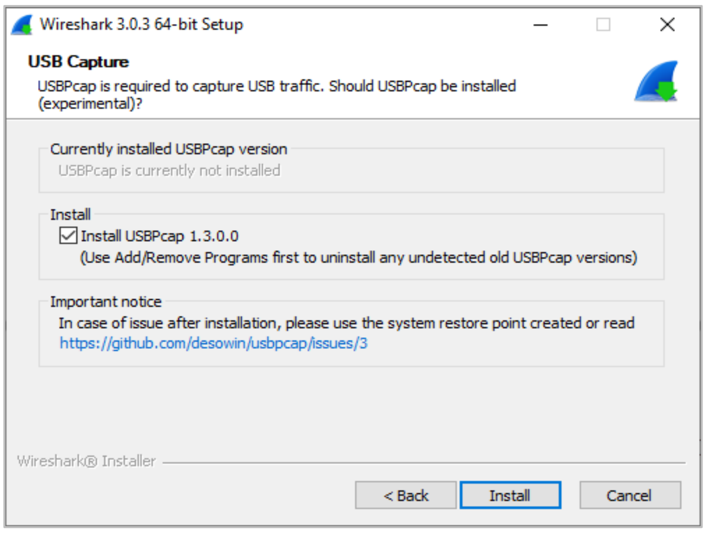
Currently installed Npcap or WinPcap version
Neither of these are installed

Install
- [✓] Install Npcap 0.995

(Use Add/Remove Programs first to uninstall any undetected old Npcap or WinPcap

Figure 2-29: Select Install Npcap 0.995 in Wireshark window

Select **Install USBPcap 1.3.0.0** and click on **Install** button.


Wireshark 3.0.3 64-bit Setup

USB Capture

USBPcap is required to capture USB traffic. Should USBPcap be installed (experimental)?

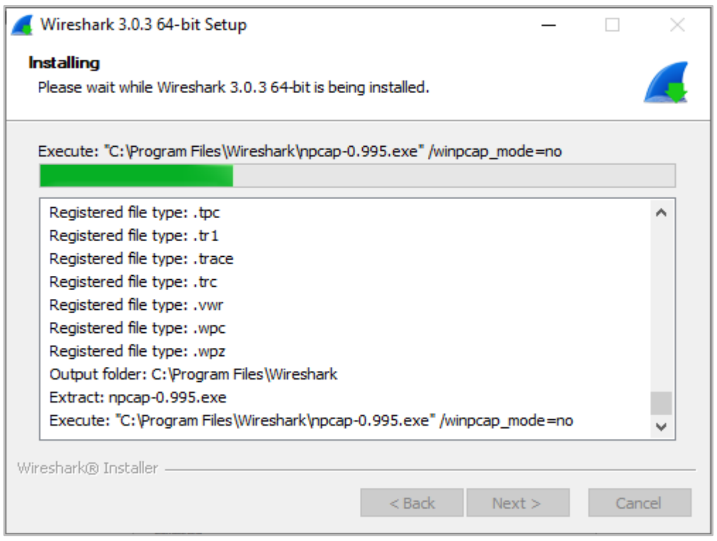
Currently installed USBPcap version
USBPcap is currently not installed

Install
☑ Install USBPcap 1.3.0.0
(Use Add/Remove Programs first to uninstall any undetected old USBPcap versions)

Important notice
In case of issue after installation, please use the system restore point created or read https://github.com/desowin/usbpcap/issues/3

Figure 2-30: Select Install USBPcap 1.3.0.0 in Wireshark window

The installation process begins.


Wireshark 3.0.3 64-bit Setup

Installing

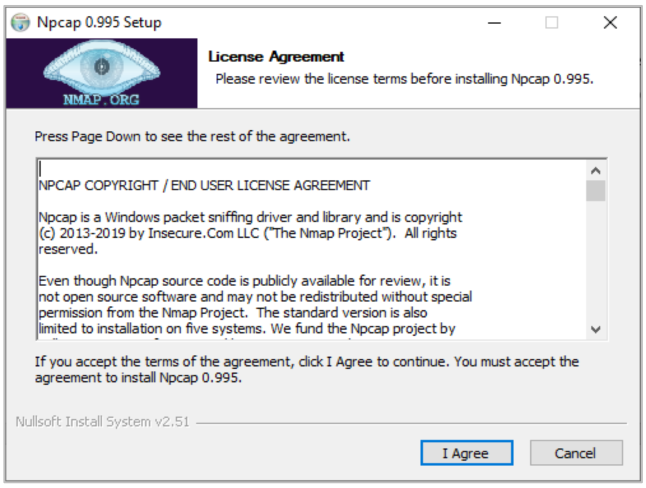
Please wait while Wireshark 3.0.3 64-bit is being installed.

```
Execute: "C:\Program Files\Wireshark\npcap-0.995.exe" /winpcap_mode=no
```

```
Registered file type: .tpc
Registered file type: .tr1
Registered file type: .trace
Registered file type: .trc
Registered file type: .vwr
Registered file type: .wpc
Registered file type: .wpz
Output folder: C:\Program Files\Wireshark
Extract: npcap-0.995.exe
Execute: "C:\Program Files\Wireshark\npcap-0.995.exe" /winpcap_mode=no
```

Figure 2-31: Wireshark installation process begins

Npcap License Agreement window appears. Click on **I Agree** button and proceed with the installation.


Npcap 0.995 Setup

License Agreement

Please review the license terms before installing Npcap 0.995.

Press Page Down to see the rest of the agreement.

NPCAP COPYRIGHT / END USER LICENSE AGREEMENT

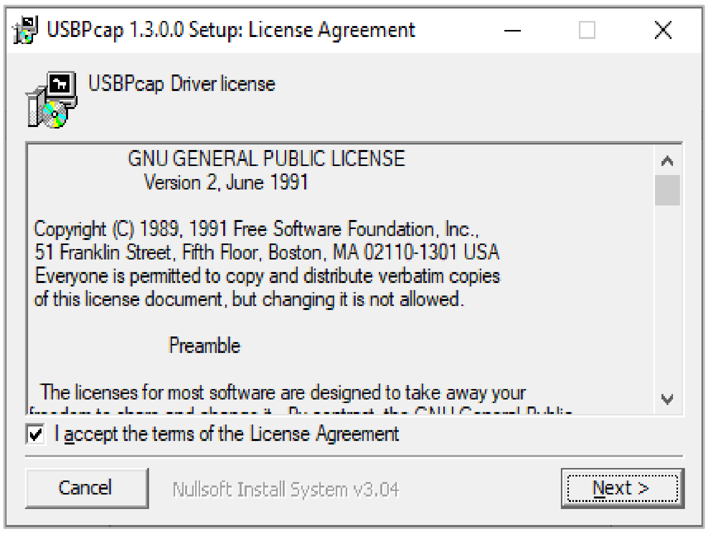
Npcap is a Windows packet sniffing driver and library and is copyright (c) 2013-2019 by Insecure.Com LLC ("The Nmap Project"). All rights reserved.

Even though Npcap source code is publicly available for review, it is not open source software and may not be redistributed without special permission from the Nmap Project. The standard version is also limited to installation on five systems. We fund the Npcap project by

If you accept the terms of the agreement, click I Agree to continue. You must accept the agreement to install Npcap 0.995.

Figure 2-32: Npcap License Agreement window

USBPcap Driver License Agreement window appears. Click on I accept the terms of the License Agreement check box and click on **Next** button.


USBPcap 1.3.0.0 Setup: License Agreement
USBPcap Driver license
I accept the terms of the License Agreement
Cancel
Nullsoft Install System v3.04
Next >

GNU GENERAL PUBLIC LICENSE
Version 2, June 1991

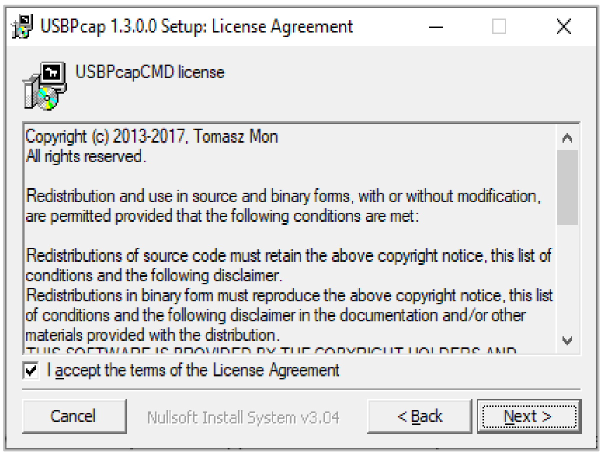
Copyright (C) 1989, 1991 Free Software Foundation, Inc.,
51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

Preamble

The licenses for most software are designed to take away your

Figure 2-33: USBPcap Driver License Agreement window

USBPcap CMD License Agreement window appears. Click on I accept the terms of the License Agreement check box and click on **Next** button.


USBPcap 1.3.0.0 Setup: License Agreement
USBPcapCMD license
I accept the terms of the License Agreement
Cancel
Nullsoft Install System v3.04
< Back
Next >

Copyright (c) 2013-2017, Tomasz Mon
All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

Figure 2-34: USBPcap CMD License Agreement window

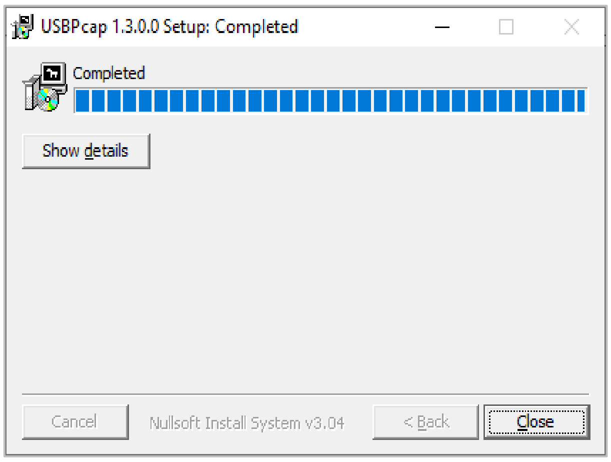
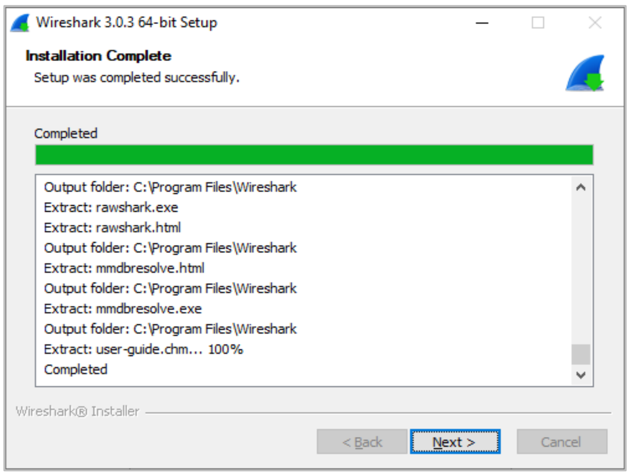

USBPcap 1.3.0.0 Setup: Completed
Completed
Show details
Cancel
Nullsoft Install System v3.04
< Back
Close

Figure 2-35: USBPcap installation is completed

The Installation Complete dialog box appears once the installation process is completed successfully. Click on the **Next** button.


Wireshark 3.0.3 64-bit Setup

Installation Complete

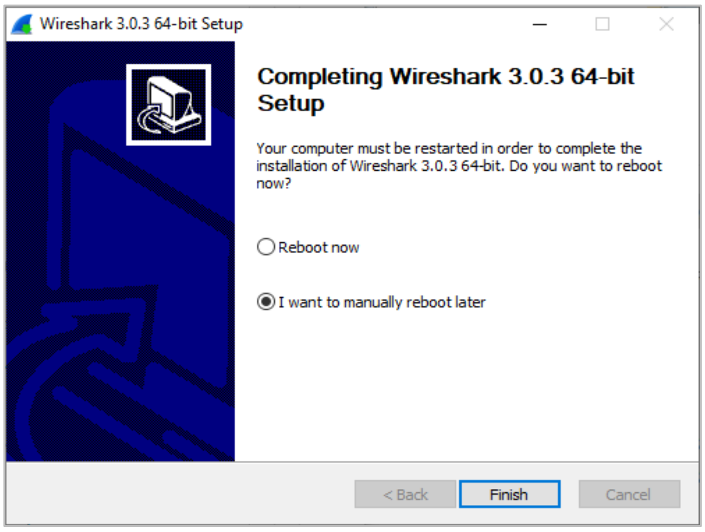
Setup was completed successfully.

Completed

```
Output folder: C:\Program Files\Wireshark
Extract: rawshark.exe
Extract: rawshark.html
Output folder: C:\Program Files\Wireshark
Extract: mmdbresolve.html
Output folder: C:\Program Files\Wireshark
Extract: mmdbresolve.exe
Output folder: C:\Program Files\Wireshark
Extract: user-guide.chm... 100%
Completed
```

Figure 2-36: Installation Complete dialog box and select next button

You will get the Wireshark Completing Setup window. Select the option **I want to manually reboot later**.


Wireshark 3.0.3 64-bit Setup

Completing Wireshark 3.0.3 64-bit Setup

Your computer must be restarted in order to complete the installation of Wireshark 3.0.3 64-bit. Do you want to reboot now?

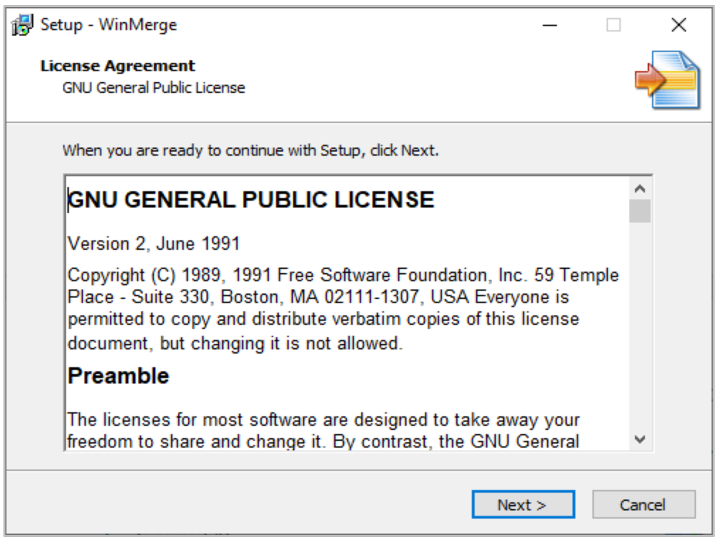
○ Reboot now

● I want to manually reboot later

Figure 2-37: Select the option I want to manually reboot later and Click on Finish button

This completes the Installation of Wireshark software. NetSim complete Setup wizard appears as shown above. After click on **Finish** button to begin with WinMerge installation.

Next the WinMerge **License Agreement** appears. Click on **Next** button


Setup - WinMerge

License Agreement
GNU General Public License

When you are ready to continue with Setup, click Next.

GNU GENERAL PUBLIC LICENSE

Version 2, June 1991

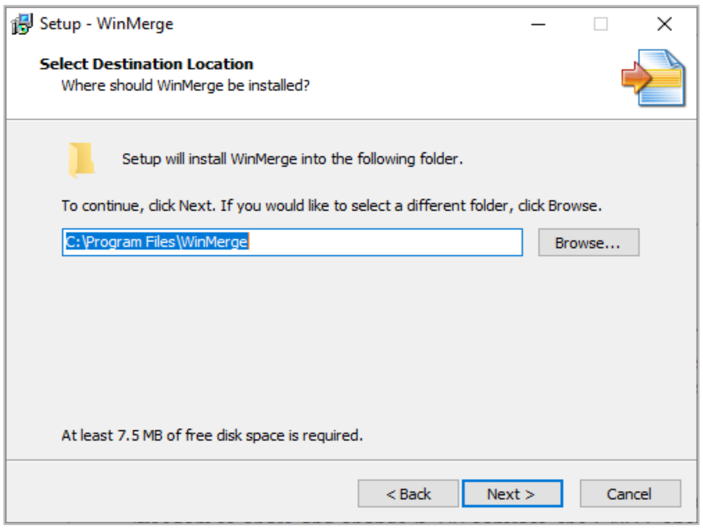
Copyright (C) 1989, 1991 Free Software Foundation, Inc. 59 Temple Place - Suite 330, Boston, MA 02111-1307, USA Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General

Figure 2-38: WinMerge License agreement window

Select the path in which WinMerge needs to be installed and click on **Next** button


Setup - WinMerge

Select Destination Location
Where should WinMerge be installed?

Setup will install WinMerge into the following folder.

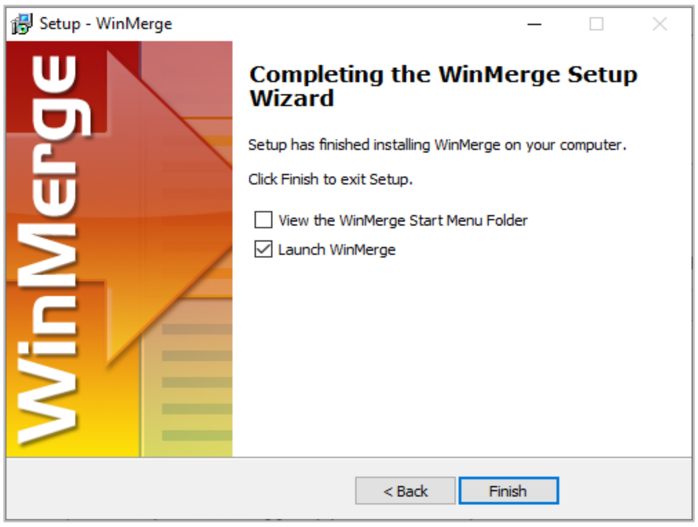
To continue, click Next. If you would like to select a different folder, click Browse.

C:\Program Files\WinMerge

At least 7.5 MB of free disk space is required.

Figure 2-39: Select the location where should WinMerge be installed

Once WinMerge installation completes, click on **Finish** button


Setup - WinMerge

Completing the WinMerge Setup Wizard

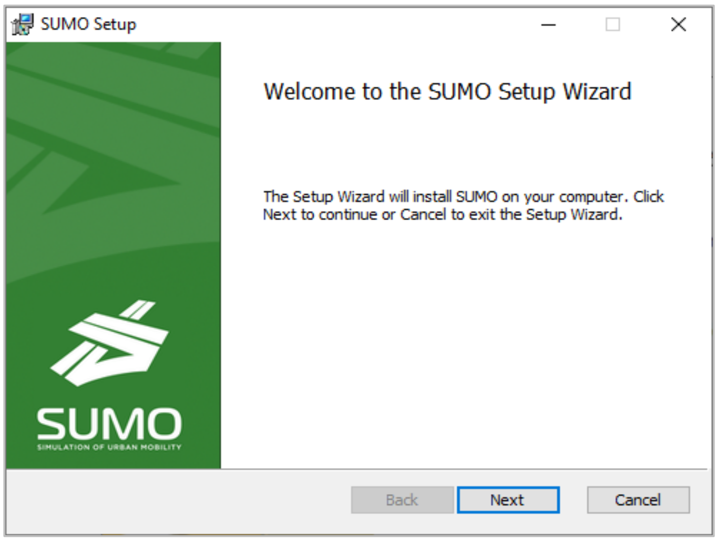
Setup has finished installing WinMerge on your computer.

Click Finish to exit Setup.

☐ View the WinMerge Start Menu Folder
☑ Launch WinMerge

Figure 2-40: Click on Finish button to completes WinMerge installation

Click on **Next** button to start SUMO installation.

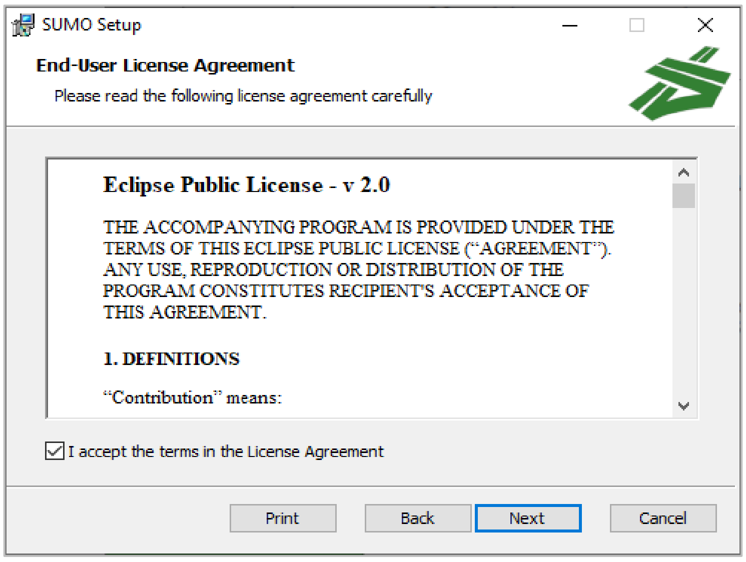


Welcome to the SUMO Setup Wizard

The Setup Wizard will install SUMO on your computer. Click Next to continue or Cancel to exit the Setup Wizard.

Figure 2-41: Sumo Installation starts

SUMO **License Agreement** appears. Accept the terms in license agreement and click on Next to proceed installation



End-User License Agreement

Please read the following license agreement carefully

Eclipse Public License - v 2.0

THE ACCOMPANYING PROGRAM IS PROVIDED UNDER THE TERMS OF THIS ECLIPSE PUBLIC LICENSE (“AGREEMENT”). ANY USE, REPRODUCTION OR DISTRIBUTION OF THE PROGRAM CONSTITUTES RECIPIENT'S ACCEPTANCE OF THIS AGREEMENT.

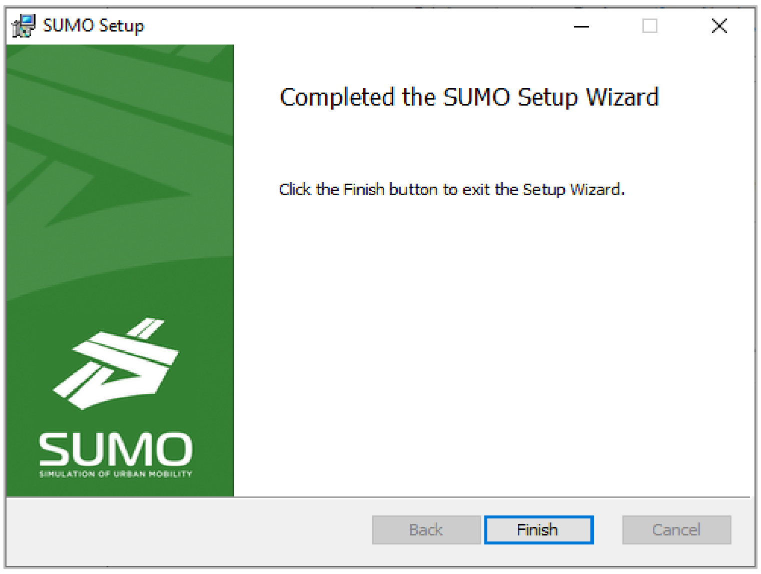
1. DEFINITIONS

“Contribution” means:

[✓] I accept the terms in the License Agreement

Figure 2-42: SUMO License Agreement window

Once SUMO installation completes, click on **Finish** button

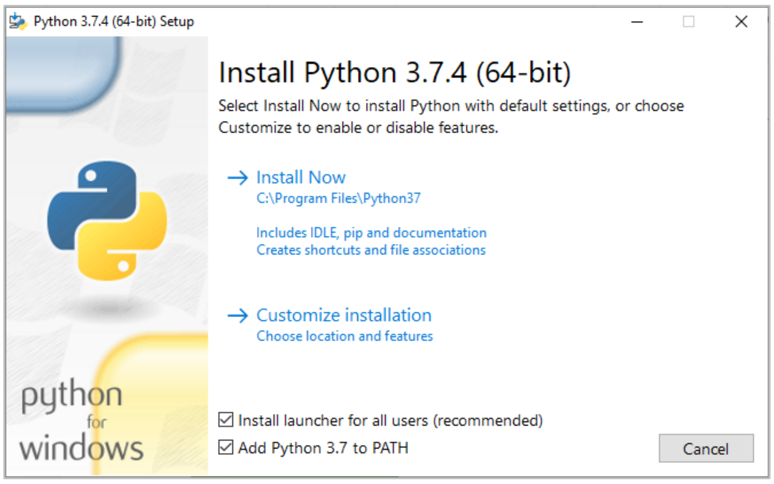

SUMO Setup

Completed the SUMO Setup Wizard

Click the Finish button to exit the Setup Wizard.

Figure 2-43: Complete SUMO Installation

Click on **Next** button to start with Python 3.7.4 installation.


Python 3.7.4 (64-bit) Setup

Install Python 3.7.4 (64-bit)

Select Install Now to install Python with default settings, or choose Customize to enable or disable features.

→ Install Now
C:\Program Files\Python37

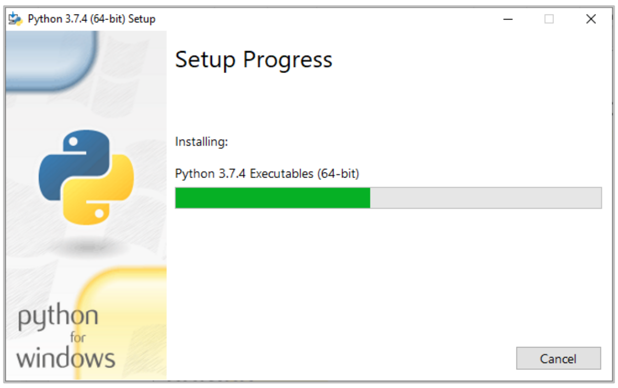
Includes IDLE, pip and documentation
Creates shortcuts and file associations

→ Customize installation
Choose location and features

☑ Install launcher for all users (recommended)
☑ Add Python 3.7 to PATH

Figure 2-44: Select "install Now" option to install Python

The installation begins once you click on **Install** option.

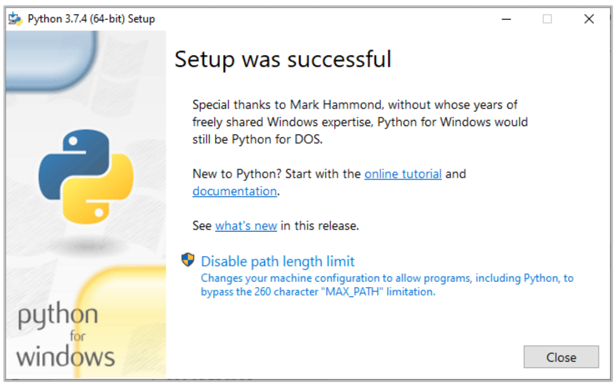

Python 3.7.4 (64-bit) Setup

Setup Progress

Installing:

Python 3.7.4 Executables (64-bit)

Figure 2-45: Python installation begins


Python 3.7.4 (64-bit) Setup

Setup was successful

Special thanks to Mark Hammond, without whose years of freely shared Windows expertise, Python for Windows would still be Python for DOS.

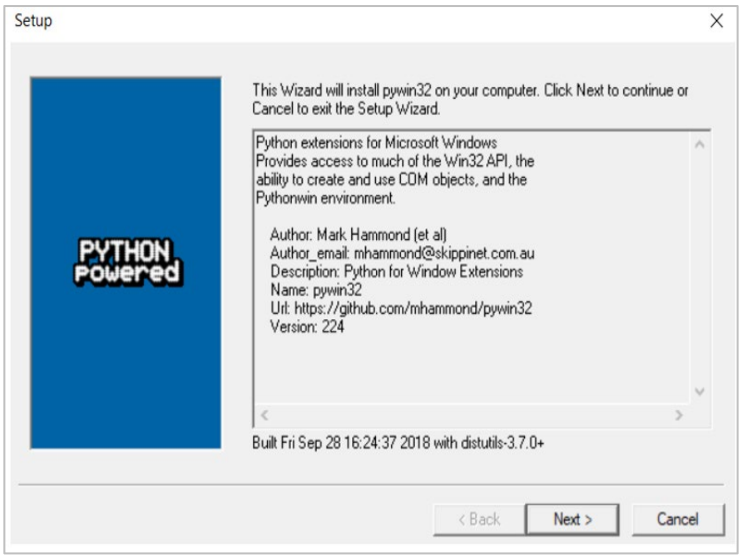
New to Python? Start with the online tutorial and documentation.

See what's new in this release.

Disable path length limit
Changes your machine configuration to allow programs, including Python, to bypass the 260 character "MAX_PATH" limitation.

Figure 2-46: Python installation successfully completed

Once the installation is finished, click on **Close** button to start the installation pywin 32


Setup

This Wizard will install pywin32 on your computer. Click Next to continue or Cancel to exit the Setup Wizard.

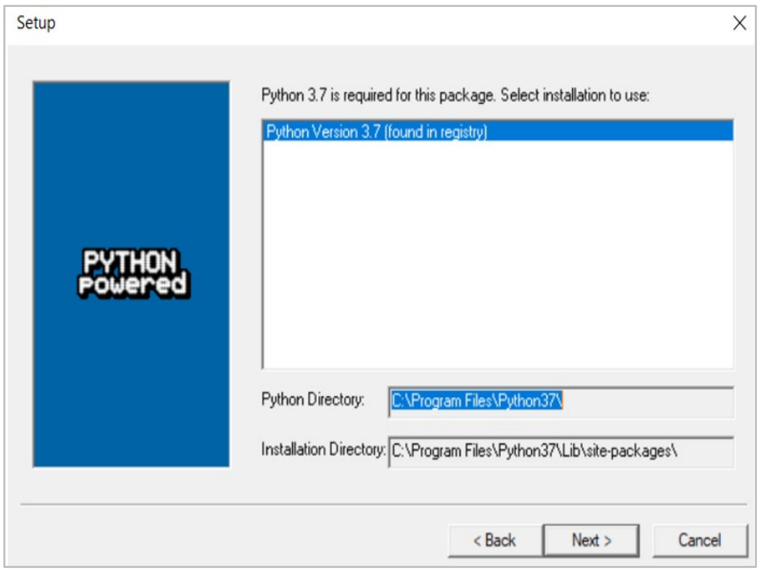
Python extensions for Microsoft Windows
Provides access to much of the Win32 API, the ability to create and use COM objects, and the Pythonwin environment.

Author: Mark Hammond (et al)
Author_email: mhammond@skippinet.com.au
Description: Python for Window Extensions
Name: pywin32
Url: https://github.com/mhammond/pywin32
Version: 224

Built Fri Sep 28 16:24:37 2018 with distutils-3.7.0+

Figure 2-47: pywin 32-224 installation wizard window

Click on **Next** button to select the directory to be used.


Setup

Python 3.7 is required for this package. Select installation to use:

Python Version 3.7 (found in registry)

Python Directory: C:\Program Files\Python37\

Installation Directory: C:\Program Files\Python37\Lib\site-packages\

Figure 2-48: Python directory path

Click on **Next** button to start the installation.

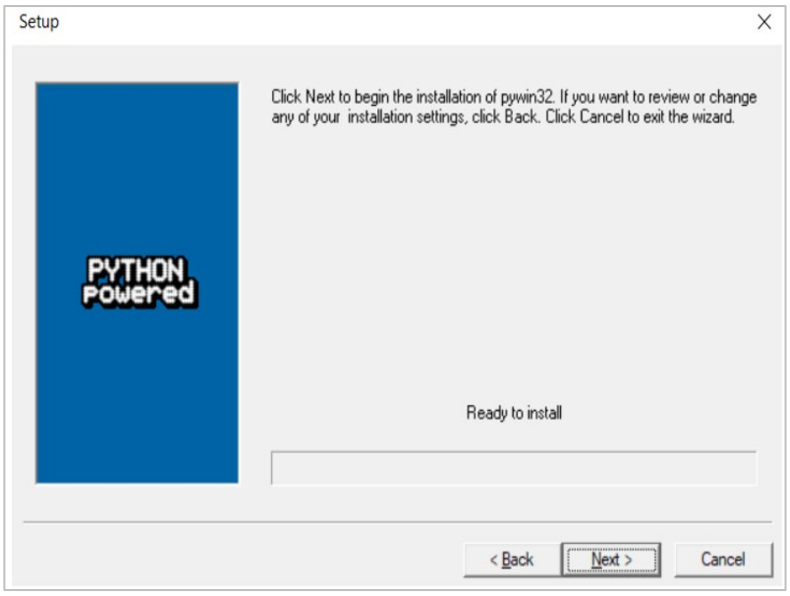

Setup
Click Next to begin the installation of pywin32. If you want to review or change any of your installation settings, click Back. Click Cancel to exit the wizard.
PYTHON Powered
Ready to install
< Back
Next >
Cancel

Figure 2-49: Select Next button to install of pywin32

Once the installation is finished, click on **Finish** button.

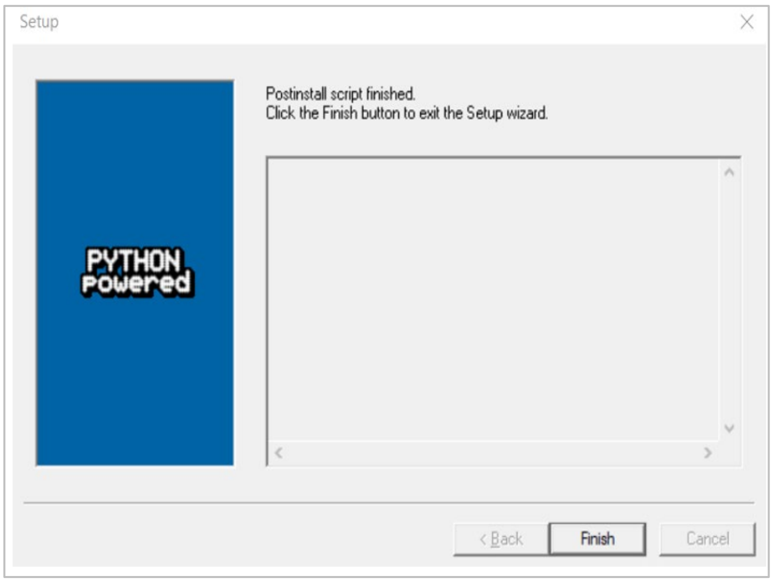
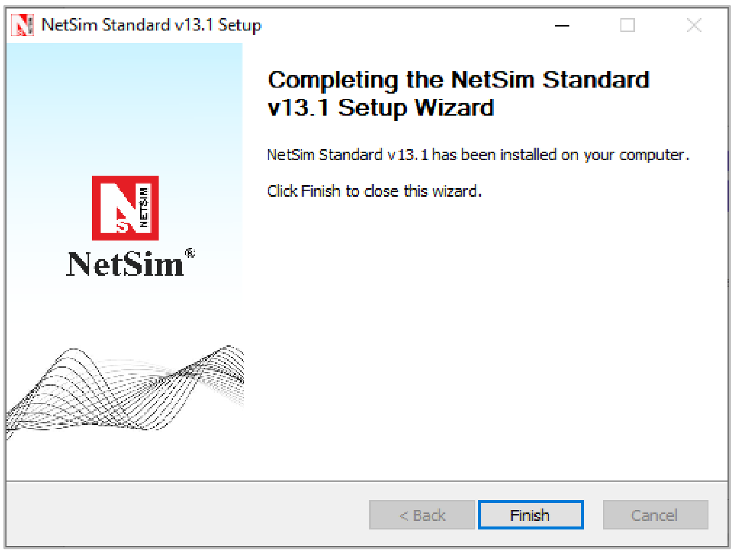

Setup
Postinstall script finished.
Click the Finish button to exit the Setup wizard.
PYTHON Powered
< Back
Finish
Cancel

Figure 2-50: Select Finish button to complete pywin installation

This completes the Installation of pywin software. NetSim complete Setup wizard appears as shown below. Click on **Finish** button to complete the installation process of NetSim.


NetSim Standard v13.1 Setup

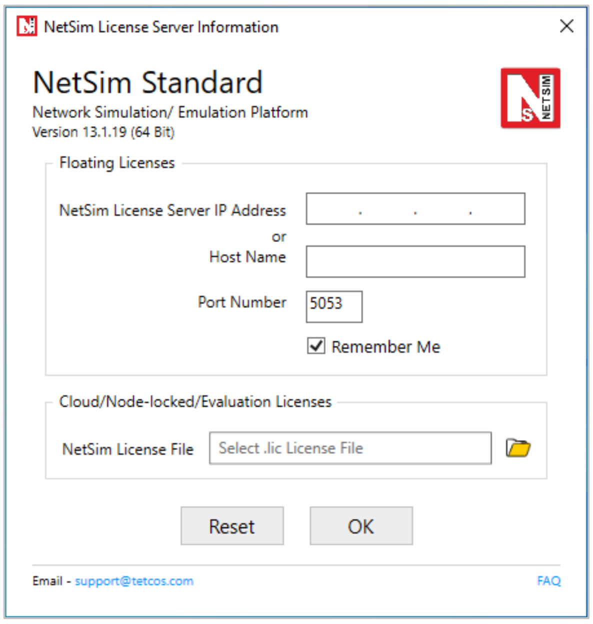
Completing the NetSim Standard v13.1 Setup Wizard

NetSim Standard v13.1 has been installed on your computer.

Click Finish to close this wizard.

Figure 2-51: NetSim complete Setup wizard

After this, to run NetSim, double click on the NetSim icon present in the desktop or right click and choose **Run as administrator** option. A **NetSim License Server Information** screen appears to start with NetSim.


NetSim License Server Information

NetSim Standard
Network Simulation/ Emulation Platform
Version 13.1.19 (64 Bit)

Floating Licenses
NetSim License Server IP Address: . . .
or
Host Name:
Port Number: 5053
✓ Remember Me

Cloud/Node-locked/Evaluation Licenses
NetSim License File: Select .lic License File

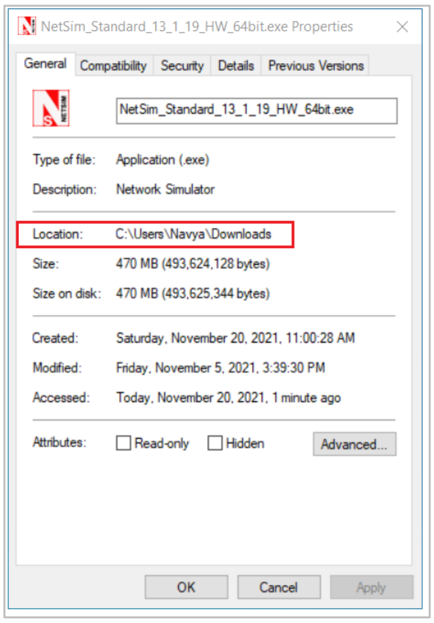
Figure 2-52: Enter NetSim License Server IP Address/Host name/Select NetSim License file

Enter the **NetSim License Server IP Address**, i.e. the system in which the License files are present and the rlm.exe file is running (Refer **Section 2.3.1** to set up NetSim License Server). In case of Cloud/Node-locked/Evaluation license browse the provided LIC file and click on OK button. Once this is done, NetSim Home screen will appear.

2.2.3 Silent installation

Steps for silent installation in NetSim are as follows.

1. For example, let us take the **NetSim_Standard_13_1_19_HW_64bit.exe** setup. Right click on NetSim Standard 64-bit setup → Go to properties and copy the **Location** as shown below.



NetSim_Standard_13_1_19_HW_64bit.exe Properties

General | Compatibility | Security | Details | Previous Versions

NetSim_Standard_13_1_19_HW_64bit.exe

Type of file: Application (.exe)
Description: Network Simulator

Location: C:\Users\Navya\Downloads
Size: 470 MB (493,624,128 bytes)
Size on disk: 470 MB (493,625,344 bytes)

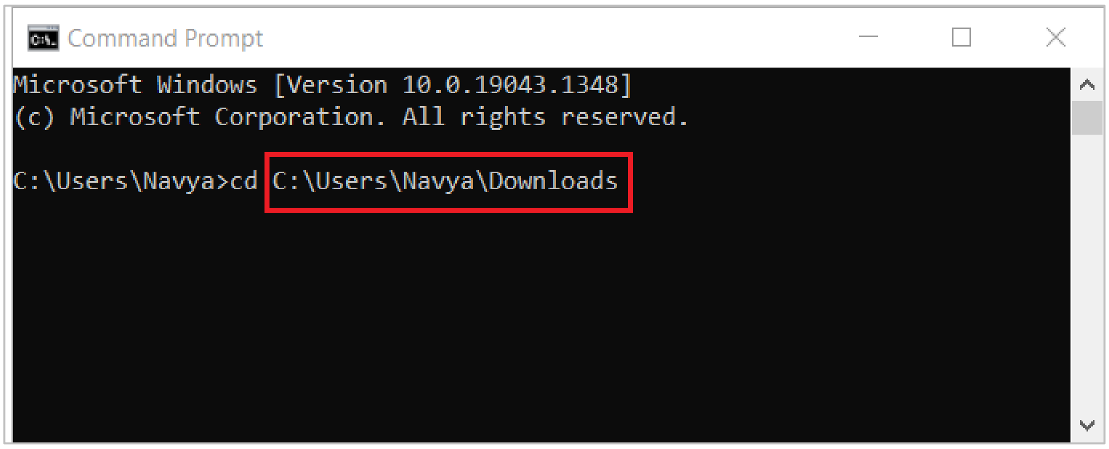
Created: Saturday, November 20, 2021, 11:00:28 AM
Modified: Friday, November 5, 2021, 3:39:30 PM
Accessed: Today, November 20, 2021, 1 minute ago

Attributes: ☐ Read-only ☐ Hidden Advanced...

OK Cancel Apply

Figure 2-53: NetSim Standard 64-bit setup location

2. Open command prompt and paste the copied location as shown below.



Command Prompt

```
Microsoft Windows [Version 10.0.19043.1348]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Navya>cd C:\Users\Navya\Downloads
```

Figure 2-54: Enter setup location in command prompt

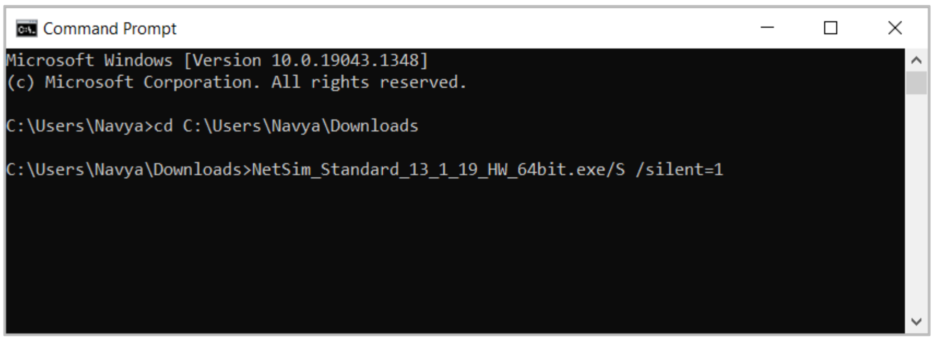
3. Run/Execute Command with the following parameters:

NetSim_Standard_13_1_19_HW_64bit.exe/S /silent=1

><setup location/S<space>/silent=1

   i. silent=1: It will install NetSim and third-party tools silently.

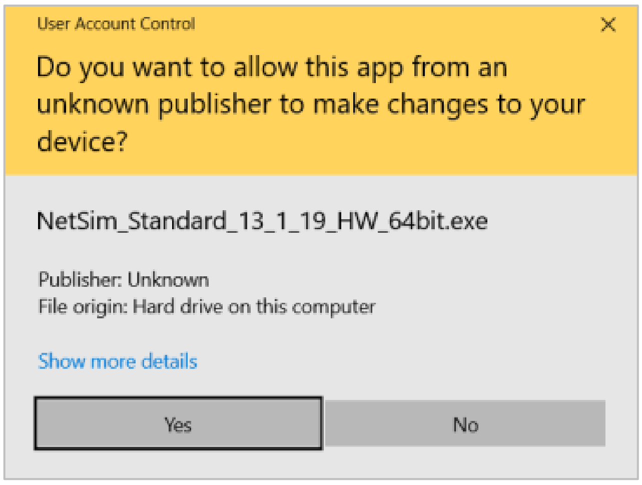
   ii. /S: It will Install NetSim itself silently.



```
Microsoft Windows [Version 10.0.19043.1348]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Navya>cd C:\Users\Navya\Downloads

C:\Users\Navya\Downloads>NetSim_Standard_13_1_19_HW_64bit.exe/S /silent=1
```

Figure 2-55: Silent installation command in command prompt

4. Press the **Enter** key. The following User Account Control message window appears. Click on **Yes** button to begin silent installation of NetSim.



User Account Control

Do you want to allow this app from an unknown publisher to make changes to your device?

NetSim_Standard_13_1_19_HW_64bit.exe

Publisher: Unknown
File origin: Hard drive on this computer

Figure 2-56: User Account Control message window appears and select Yes

Note: *Complete installation of NetSim may take up to 2 or 3 minutes.*

2.3 Setting up License Server

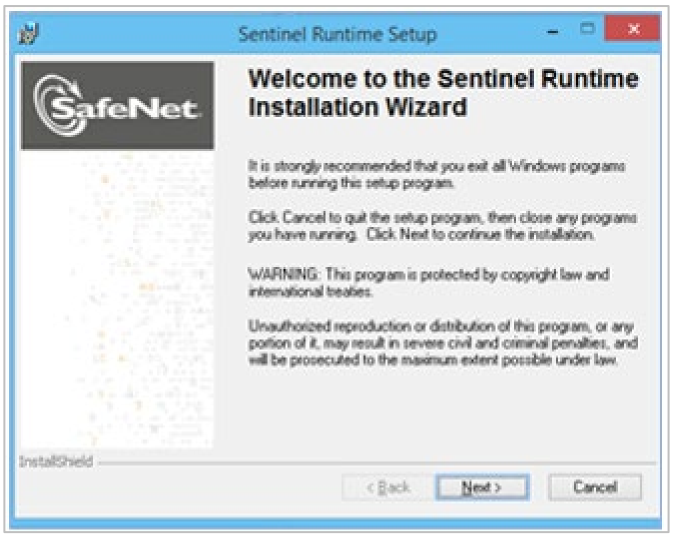
2.3.1 Installing NetSim RLM Dongle Driver Software (Dongle Based Licenses)

This section guides you to install the **RLMDongle Driver** software from the CD-ROM.

1. Insert the CD-ROM disc in the CD drive.
2. Double click on My Computer and access the CD Drive.
3. Double click on Driver_Software folder.
4. Double click on HASPUserSetup.exe

Each prompt displayed during the process tells you what it is about to do and prompts to either **continue** or exit.

Setup prepares the installation wizard and the driver software installation begins with a Welcome Screen. Click on **Next** button.



Welcome to the Sentinel Runtime Installation Wizard

It is strongly recommended that you exit all Windows programs before running this setup program.

Click Cancel to quit the setup program, then close any programs you have running. Click Next to continue the installation.

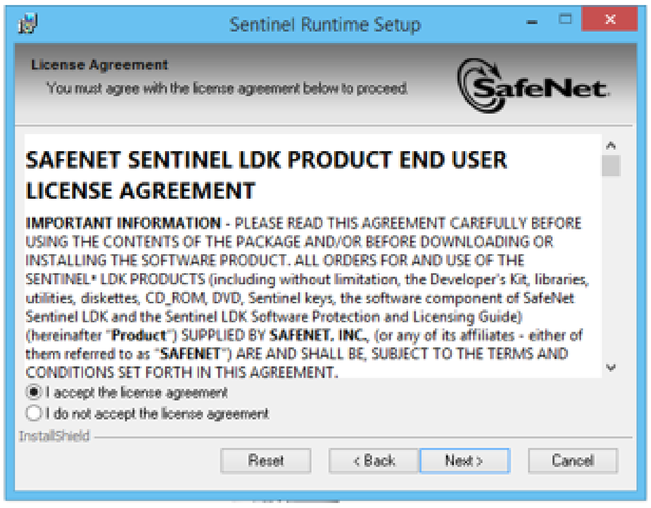
WARNING: This program is protected by copyright law and international treaties.

Unauthorized reproduction or distribution of this program, or any portion of it, may result in severe civil and criminal penalties, and will be prosecuted to the maximum extent possible under law.

Figure 2-57: Sentinel Runtime Setup window and select Next button

Note: *Any other program running during the installation of the Dongle will affect the proper installation of the software.*

Sentinel Runtime Setup License Agreement appears. Read the license agreement carefully, scroll down to read the complete license agreement. If the requirement of the license agreement is accepted, Click on I accept the license agreement and click on Next button else quit the setup by clicking Cancel button.



License Agreement

You must agree with the license agreement below to proceed.

SAFENET SENTINEL LDK PRODUCT END USER LICENSE AGREEMENT

IMPORTANT INFORMATION - PLEASE READ THIS AGREEMENT CAREFULLY BEFORE USING THE CONTENTS OF THE PACKAGE AND/OR BEFORE DOWNLOADING OR INSTALLING THE SOFTWARE PRODUCT. ALL ORDERS FOR AND USE OF THE SENTINEL® LDK PRODUCTS (including without limitation, the Developer's Kit, libraries, utilities, diskettes, CD_ROM, DVD, Sentinel keys, the software component of SafeNet Sentinel LDK and the Sentinel LDK Software Protection and Licensing Guide) (hereinafter "Product") SUPPLIED BY SAFENET, INC., (or any of its affiliates - either of them referred to as "SAFENET") ARE AND SHALL BE, SUBJECT TO THE TERMS AND CONDITIONS SET FORTH IN THIS AGREEMENT.

● I accept the license agreement

○ I do not accept the license agreement

Figure 2-58: Sentinel Runtime Setup License Agreement window appears and select Next button

The installation process begins.

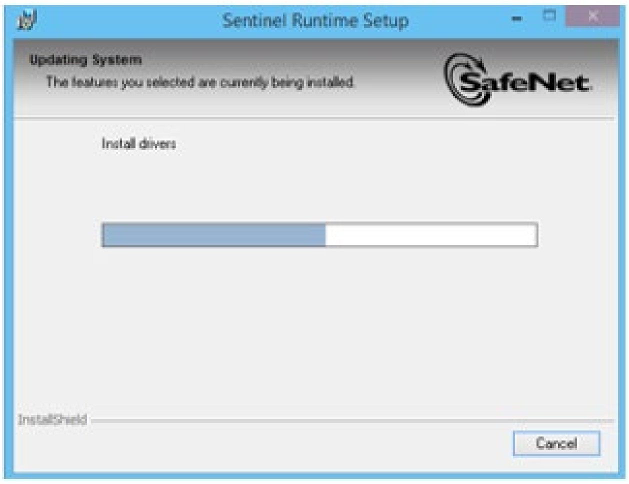
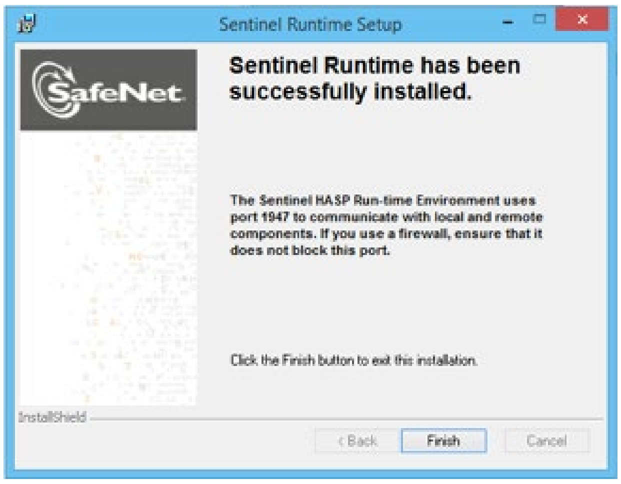


Figure 2-59: Installation process begins

Once the Sentinel Runtime is installed successfully, click on **Finish** button.



Sentinel Runtime has been successfully installed.

The Sentinel HASP Run-time Environment uses port 1947 to communicate with local and remote components. If you use a firewall, ensure that it does not block this port.

Click the Finish button to exit this installation.

Figure 2-60: Sentinel Runtime is installed successfully and select Finish button

Now the RLM driver software is installed successfully. If the driver has been successfully installed, then upon connecting the Dongle in the USB port, a red light will glow (Refer picture below Figure 2-61). If the driver is not properly installed, this light will not glow when the dongle is connected to the USB Port.

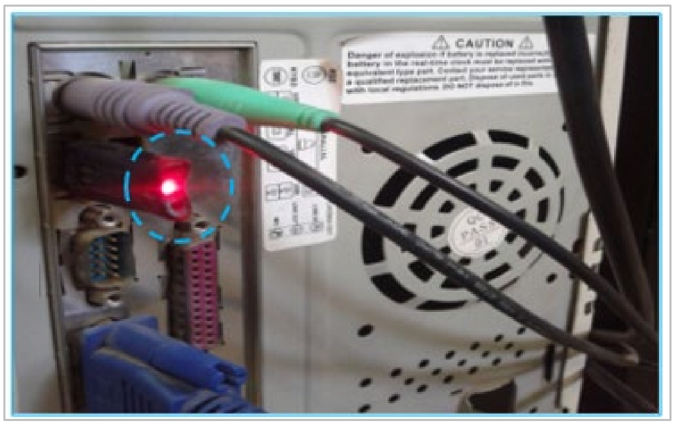


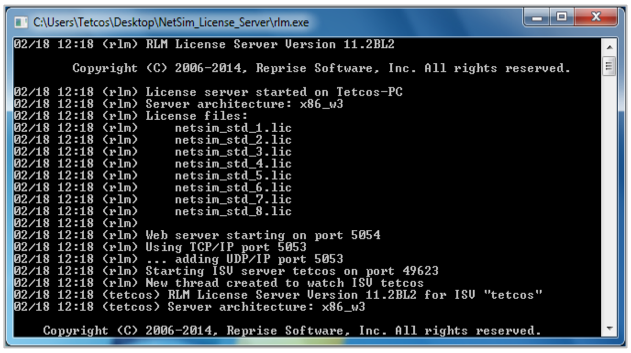
Figure 2-61: Connecting the Dongle in the USB Port

2.3.2 Running NetSim License Server

- Copy the **NetSim License Server** folder and paste it on **Desktop**. Check that it has the license file. If not copy the paste the license file into the **License server folder**
- Double click on **NetSim License Server** folder from Desktop.
- Double click on **rlm.exe**
- For hardware dongle-based users: After the Driver Software installation, connect the **RLM dongle** to the **system USB port**. Double click on My Computer and access the CD Drive. This CD contents will have the NetSim License server folder.

Note*: For running* ***NetSim****,* ***rlm.exe*** *must be running in the server (license server) system and the server system IP address must be entered correctly. Without running* ***rlm.exe****,* ***NetSim*** *won't run.*

While running rlm.exe, the screen will appear as shown below Figure 2-62.

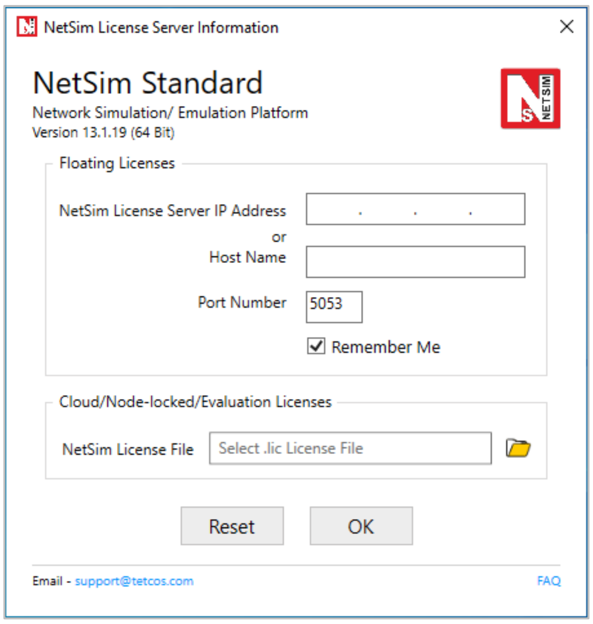


```
C:\Users\Tetcos\Desktop\NetSim_License_Server\rlm.exe
02/18 12:18 (rlm) RLM License Server Version 11.2BL2

        Copyright (C) 2006-2014, Reprise Software, Inc. All rights reserved.

02/18 12:18 (rlm) License server started on Tetcos-PC
02/18 12:18 (rlm) Server architecture: x86_w3
02/18 12:18 (rlm) License files:
02/18 12:18 (rlm)     netsim_std_1.lic
02/18 12:18 (rlm)     netsim_std_2.lic
02/18 12:18 (rlm)     netsim_std_3.lic
02/18 12:18 (rlm)     netsim_std_4.lic
02/18 12:18 (rlm)     netsim_std_5.lic
02/18 12:18 (rlm)     netsim_std_6.lic
02/18 12:18 (rlm)     netsim_std_7.lic
02/18 12:18 (rlm)     netsim_std_8.lic
02/18 12:18 (rlm)
02/18 12:18 (rlm) Web server starting on port 5054
02/18 12:18 (rlm) Using TCP/IP port 5053
02/18 12:18 (rlm) ... adding UDP/IP port 5053
02/18 12:18 (rlm) Starting ISV server tetcos on port 49623
02/18 12:18 (rlm) New thread created to watch ISV tetcos
02/18 12:18 (tetcos) RLM License Server Version 11.2BL2 for ISV "tetcos"
02/18 12:18 (tetcos) Server architecture: x86_w3

    Copyright (C) 2006-2014, Reprise Software, Inc. All rights reserved.
```

Figure 2-62: When NetSim license server system running, window appears

2.3.3 Running NetSim Software

After running rlm.exe, double click the NetSim icon in the Desktop. The screen given below will be obtained. Enter the Server IP address where the rlm.exe is running and click OK.



NetSim License Server Information

NetSim Standard

Network Simulation/ Emulation Platform

Version 13.1.19 (64 Bit)

Floating Licenses

NetSim License Server IP Address: . . .

or

Host Name:

Port Number: 5053

☑ Remember Me

Cloud/Node-locked/Evaluation Licenses

NetSim License File: Select .lic License File

Reset | OK

Email - support@tetcos.com

FAQ

Figure 2-63: Enter NetSim License Server IP Address

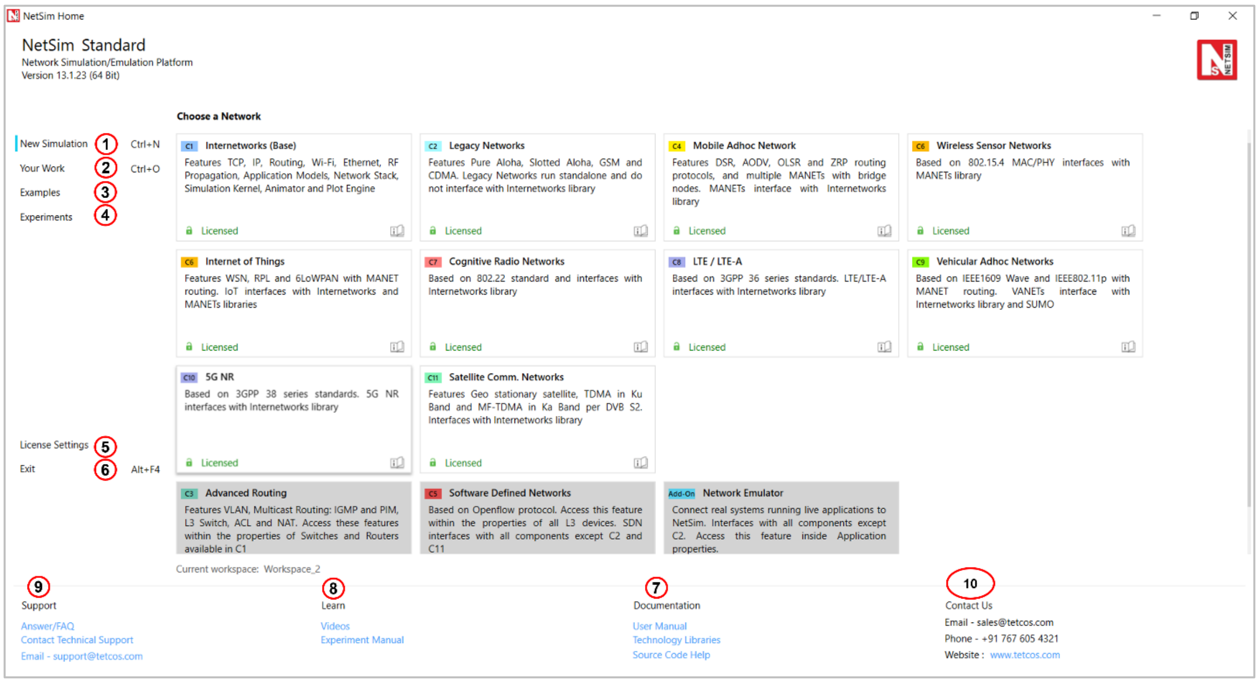
3 NetSim GUI

The graphical user interface (GUI) allows users to interact with the simulator for creating, modifying and saving, simulation experiments and workspaces. This is much easier to use when compared to command line or text-based simulator interfaces. NetSim GUI comprises of the Home Screen, Design Window, Results Window, Animation Window and Plots Window.

3.1 Menus in the NetSim Home Screen

You see the NetSim Home Screen when you run NetSim software for the first time, after checking out a license from the NetSim License Server.

See the following image for an example of the NetSim Home screen as shown below Figure 3-1.


NetSim Home

NetSim Standard
Network Simulation/Emulation Platform
Version 13.1.23 (64 Bit)

New Simulation 1 Ctrl+N
Your Work 2 Ctrl+O
Examples 3
Experiments 4
License Settings 5
Exit 6 Alt+F4

Choose a Network

C1 Internetworks (Base)
Features TCP, IP, Routing, Wi-Fi, Ethernet, RF Propagation, Application Models, Network Stack, Simulation Kernel, Animator and Plot Engine
Licensed

C2 Legacy Networks
Features Pure Aloha, Slotted Aloha, GSM and CDMA. Legacy Networks run standalone and do not interface with Internetworks library
Licensed

C4 Mobile Adhoc Network
Features DSR, AODV, OLSR and ZRP routing protocols, and multiple MANETs with bridge nodes. MANETs interface with Internetworks library
Licensed

C6 Wireless Sensor Networks
Based on 802.15.4 MAC/PHY interfaces with MANETs library
Licensed

C6 Internet of Things
Features WSN, RPL and 6LoWPAN with MANET routing. IoT interfaces with Internetworks and MANETs libraries
Licensed

C7 Cognitive Radio Networks
Based on 802.22 standard and interfaces with Internetworks library
Licensed

C8 LTE / LTE-A
Based on 3GPP 36 series standards. LTE/LTE-A interfaces with Internetworks library
Licensed

C9 Vehicular Adhoc Networks
Based on IEEE1609 Wave and IEEE802.11p with MANET routing. VANETs interface with Internetworks library and SUMO
Licensed

C10 5G NR
Based on 3GPP 38 series standards. 5G NR interfaces with Internetworks library
Licensed

C11 Satellite Comm. Networks
Features Geo stationary satellite, TDMA in Ku Band and MF-TDMA in Ka Band per DVB S2. Interfaces with Internetworks library
Licensed

C3 Advanced Routing
Features VLAN, Multicast Routing: IGMP and PIM, L3 Switch, ACL and NAT. Access these features within the properties of Switches and Routers available in C1

C5 Software Defined Networks
Based on Openflow protocol. Access this feature within the properties of all L3 devices. SDN interfaces with all components except C2 and C11

Add-On Network Emulator
Connect real systems running live applications to NetSim. Interfaces with all components except C2. Access this feature inside Application properties.

Current workspace: Workspace_2

9 Support
Answer/FAQ
Contact Technical Support
Email - support@tetcos.com

8 Learn
Videos
Experiment Manual

7 Documentation
User Manual
Technology Libraries
Source Code Help

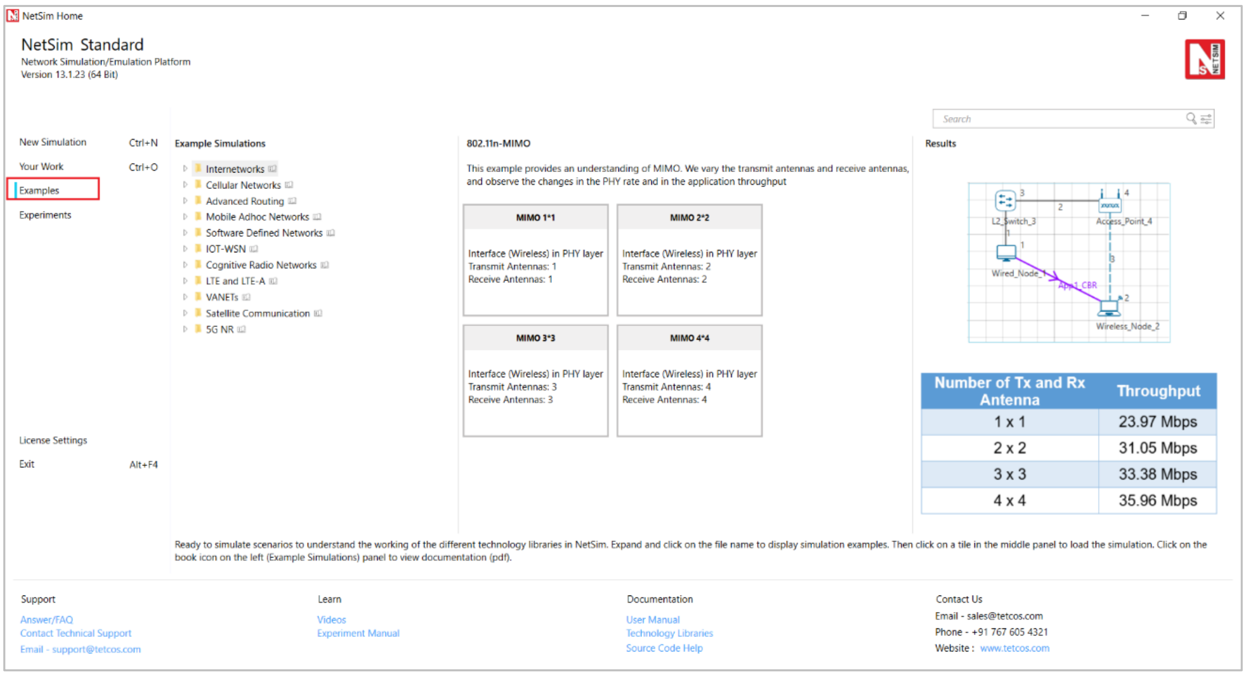
10 Contact Us
Email - sales@tetcos.com
Phone - +91 767 605 4321
Website : www.tetcos.com

Figure 3-1: NetSim Home screen

You see the following items on the NetSim Home screen:

1. **New Simulation:** Use this menu to simulate different types of networks in NetSim. You can simulate the following the types of networks: Internetworks, Legacy Networks, Mobile Adhoc networks, Cellular Networks, Wireless Sensor Networks, Internet of Things, Cognitive Radio Networks, LTE/LTE-A Networks, 5G NR (newly added component in v12), VANETs, and Satellite Communication. Only the networks (components) for which licenses are available will be shown. The networks (components) shown at the bottom with grey background cannot be directly clicked and entered. These features can be accessed through other components given the dependencies.

2. **Your work:** Use this menu to load saved configuration files from the current workspace. You can view, modify or re-run existing simulations. Along with this, users can also export the saved files from the current workspace to their preferred location on their PC's.
3. **Examples:** Use this menu to perform simulations of different kinds categorized technology-wise. Users can choose any network which they want to work. Expand and click on file name to display simulation examples. Then click on a tile in middle panel to load simulation, users can run and analyze the results. Users can click on the book icon present in the right-hand side of each network which opens the corresponding pdf files. This helps the users with all information about the current simulation as well as the entire network technology.


NetSim Home

NetSim Standard
Network Simulation/Emulation Platform
Version 13.1.23 (64 Bit)

New Simulation	Ctrl+N
Your Work	Ctrl+O
Examples	
Experiments	
License Settings	
Exit	Alt+F4

Example Simulations
- Internetworks
- Cellular Networks
- Advanced Routing
- Mobile Adhoc Networks
- Software Defined Networks
- IOT-WSN
- Cognitive Radio Networks
- LTE and LTE-A
- VANETs
- Satellite Communication
- 5G NR

802.11n-MIMO

This example provides an understanding of MIMO. We vary the transmit antennas and receive antennas, and observe the changes in the PHY rate and in the application throughput

MIMO 1*1
Interface (Wireless) in PHY layer
Transmit Antennas: 1
Receive Antennas: 1

MIMO 2*2
Interface (Wireless) in PHY layer
Transmit Antennas: 2
Receive Antennas: 2

MIMO 3*3
Interface (Wireless) in PHY layer
Transmit Antennas: 3
Receive Antennas: 3

MIMO 4*4
Interface (Wireless) in PHY layer
Transmit Antennas: 4
Receive Antennas: 4

Results

Number of Tx and Rx Antenna	Throughput
1 x 1	23.97 Mbps
2 x 2	31.05 Mbps
3 x 3	33.38 Mbps
4 x 4	35.96 Mbps

Ready to simulate scenarios to understand the working of the different technology libraries in NetSim. Expand and click on the file name to display simulation examples. Then click on a tile in the middle panel to load the simulation. Click on the book icon on the left (Example Simulations) panel to view documentation (pdf).

Support
Answer/FAQ
Contact Technical Support
Email - support@tetcos.com

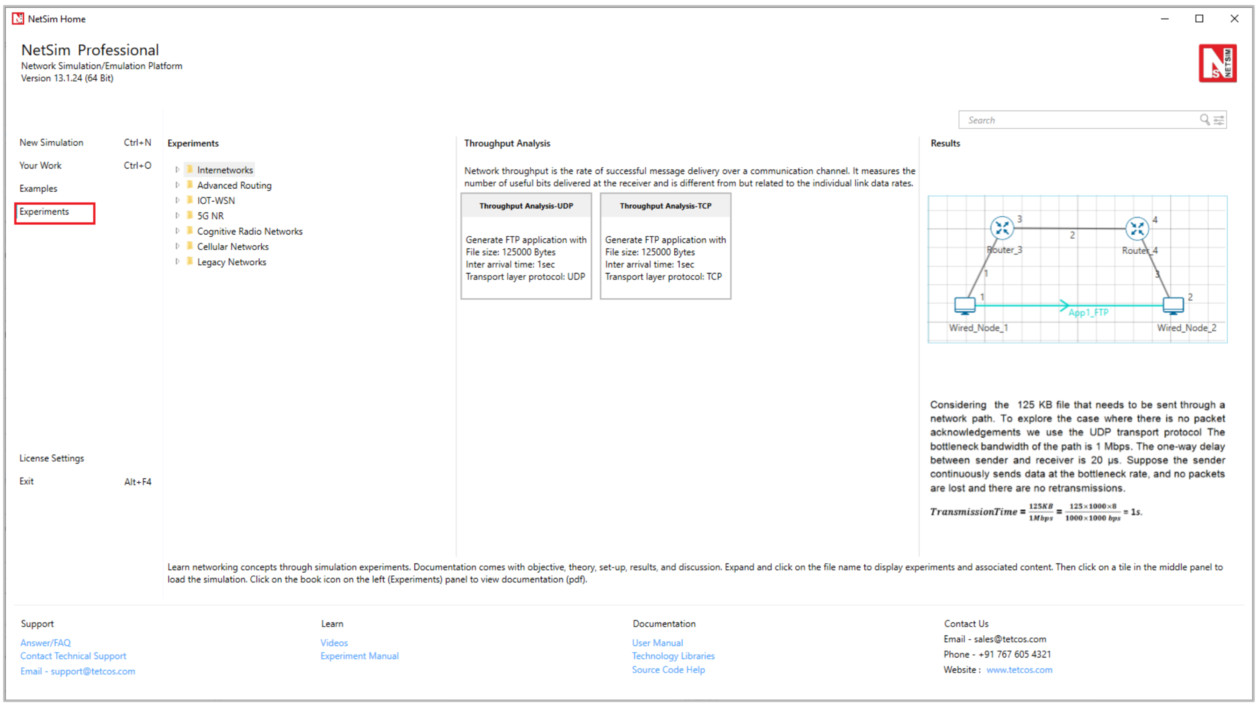
Learn
Videos
Experiment Manual

Documentation
User Manual
Technology Libraries
Source Code Help

Contact Us
Email - sales@tetcos.com
Phone - +91 767 605 4321
Website : www.tetcos.com

Figure 3-2: Featured Examples List Window

4. **Experiment:** Users can use this menu to find experiments section which has various experiments covering all the technologies in NetSim. Users can choose their experiment by Expand and click on file name to display the experiment. Then click on a tile in middle panel to load simulation. All the settings to carry out the experiment are already done. Users can click on the book icon present in the right-hand side of each experiment. This will open the corresponding pdf file for the experiment which consists of detailed description of that experiment.



Network throughput is the rate of successful message delivery over a communication channel. It measures the number of useful bits delivered at the receiver and is different from but related to the individual link data rates.

Throughput Analysis-UDP	Throughput Analysis-TCP
Generate FTP application with File size: 125000 Bytes Inter arrival time: 1sec Transport layer protocol: UDP	Generate FTP application with File size: 125000 Bytes Inter arrival time: 1sec Transport layer protocol: TCP

Considering the 125 KB file that needs to be sent through a network path. To explore the case where there is no packet acknowledgements we use the UDP transport protocol The bottleneck bandwidth of the path is 1 Mbps. The one-way delay between sender and receiver is 20 µs. Suppose the sender continuously sends data at the bottleneck rate, and no packets are lost and there are no retransmissions.

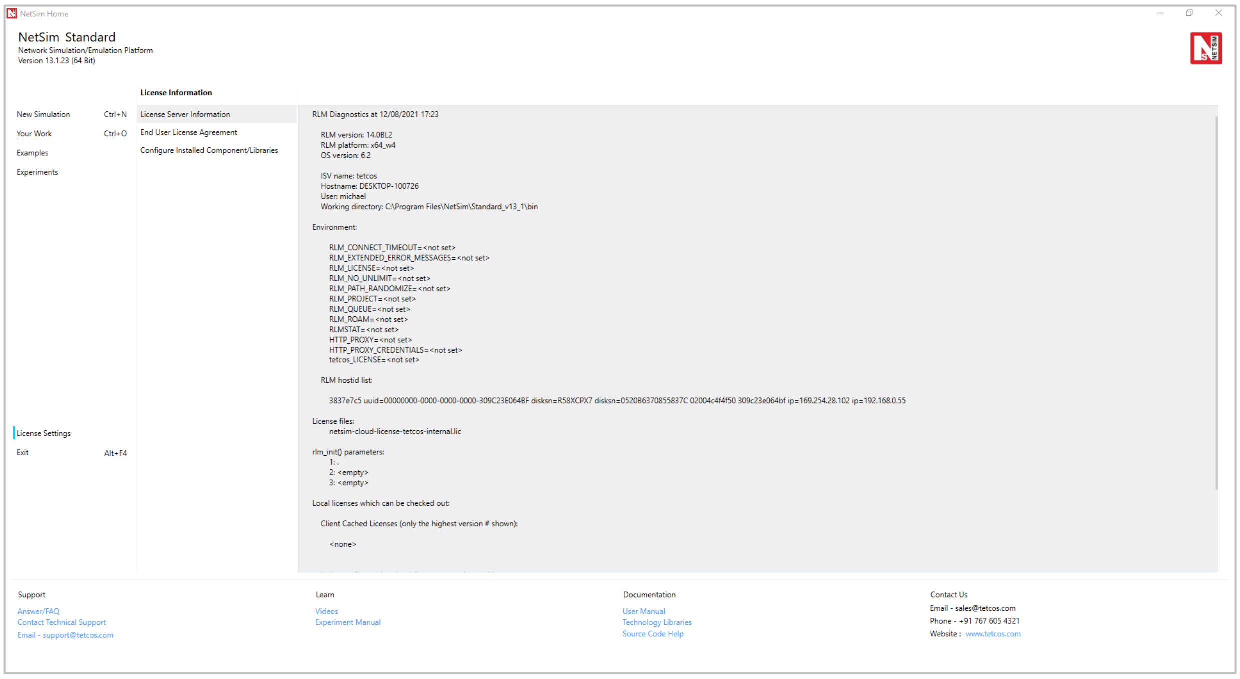
$$TransmissionTime = \frac{125KB}{1Mbps} = \frac{125\times1000\times8}{1000\times1000\ bps} = 1s.$$

Learn networking concepts through simulation experiments. Documentation comes with objective, theory, set-up, results, and discussion. Expand and click on the file name to display experiments and associated content. Then click on a tile in the middle panel to load the simulation. Click on the book icon on the left (Experiments) panel to view documentation (pdf).

Figure 3-3: Experiments List Window

5. **License Settings:** Use this menu to perform the following. Click on License Settings provides users with three sub-menus related to License information.

License Server Information: Use this menu to view details about the NetSim License Server from where the client is checking out licenses.



```
RLM Diagnostics at 12/08/2021 17:23

  RLM version: 14.0BL2
  RLM platform: x64_w4
  OS version: 6.2

  ISV name: tetcos
  Hostname: DESKTOP-100726
  User: michael
  Working directory: C:\Program Files\NetSim\Standard_v13_1\bin

Environment:

    RLM_CONNECT_TIMEOUT=<not set>
    RLM_EXTENDED_ERROR_MESSAGES=<not set>
    RLM_LICENSE=<not set>
    RLM_NO_UNLIMIT=<not set>
    RLM_PATH_RANDOMIZE=<not set>
    RLM_PROJECT=<not set>
    RLM_QUEUE=<not set>
    RLM_ROAM=<not set>
    RLMSTAT=<not set>
    HTTP_PROXY=<not set>
    HTTP_PROXY_CREDENTIALS=<not set>
    tetcos_LICENSE=<not set>

  RLM hostid list:

    3837e7c5 uuid=00000000-0000-0000-0000-309C23E064BF disksn=R58XCPX7 disksn=0520B6370855837C 02004c4f4f50 309c23e064bf ip=169.254.28.102 ip=192.168.0.55

License files:
    netsim-cloud-license-tetcos-internal.lic

rlm_init() parameters:
    1: .
    2: <empty>
    3: <empty>

Local licenses which can be checked out:

  Client Cached Licenses (only the highest version # shown):

    <none>
```

Figure 3-4: NetSim License Server Information window

You will see the following details on the NetSim Home screen, if you click the License Server Info menu item: the type of platform on which NetSim is running, the version of RLM, the Dongle RLM

ID, the IP address of the NetSim License Server, and the path to the license files in the server hosting NetSim License Server.

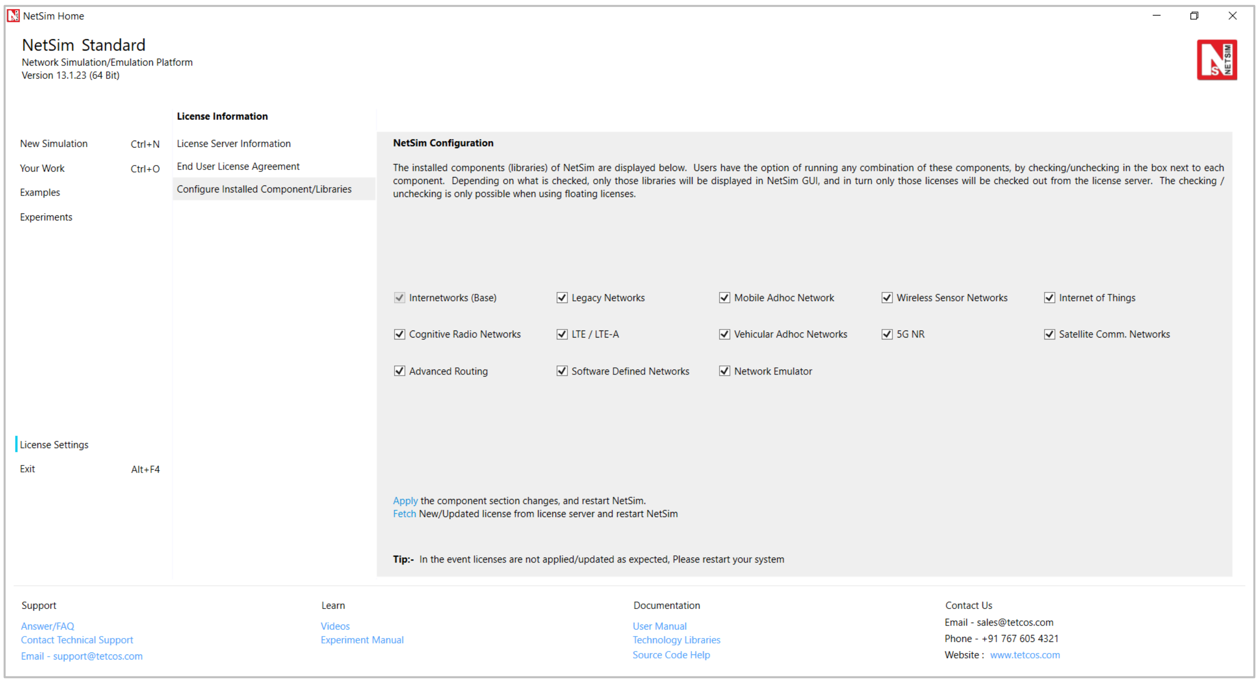
End User License Agreement: Use this menu to view the end user license agreement. You will see the following details on NetSim Home screen, if you click the End User License Agreement menu item: Grant of License and Use of the Services, License Restrictions, License Duration, Upgrade and Support Service etc.

Configure Installed Components/Libraries: Use this menu to allow NetSim users to simulate only specific types of networks (by the licenses and libraries associated with the types of networks). You control access to types of networks by selecting libraries for specific types of networks that NetSim License Server checks out when NetSim users start NetSim.

NetSim Home screen displays libraries for components for which you have purchased licenses.

Note: *You can select or clear libraries and control access to NetSim users, only if you are using floating licenses.*

See the following image for an example of what the NetSim Home screen displays, if you click the Configure Installed Components/Libraries menu item.


NetSim Home

NetSim Standard
Network Simulation/Emulation Platform
Version 13.1.23 (64 Bit)

New Simulation	Ctrl+N
Your Work	Ctrl+O
Examples	
Experiments	
License Settings	
Exit	Alt+F4

License Information

License Server Information
End User License Agreement
Configure Installed Component/Libraries

NetSim Configuration

The installed components (libraries) of NetSim are displayed below. Users have the option of running any combination of these components, by checking/unchecking in the box next to each component. Depending on what is checked, only those libraries will be displayed in NetSim GUI, and in turn only those licenses will be checked out from the license server. The checking / unchecking is only possible when using floating licenses.

✓ Internetworks (Base)	✓ Legacy Networks	✓ Mobile Adhoc Network	✓ Wireless Sensor Networks	✓ Internet of Things
✓ Cognitive Radio Networks	✓ LTE / LTE-A	✓ Vehicular Adhoc Networks	✓ 5G NR	✓ Satellite Comm. Networks
✓ Advanced Routing	✓ Software Defined Networks	✓ Network Emulator		

Apply the component section changes, and restart NetSim.
Fetch New/Updated license from license server and restart NetSim

Tip:- In the event licenses are not applied/updated as expected, Please restart your system

Support	Learn	Documentation	Contact Us
Answer/FAQ	Videos	User Manual	Email - sales@tetcos.com
Contact Technical Support	Experiment Manual	Technology Libraries	Phone - +91 767 605 4321
Email - support@tetcos.com		Source Code Help	Website : www.tetcos.com

Figure 3-5: The Installed Component (Libraries) of NetSim

Use the License Settings menu as follows:

- Select the checkboxes for the component libraries (types of networks) that NetSim users must be allowed to simulate.
- Clear the checkboxes for the component libraries (types of networks) that NetSim users must not be allowed to simulate.

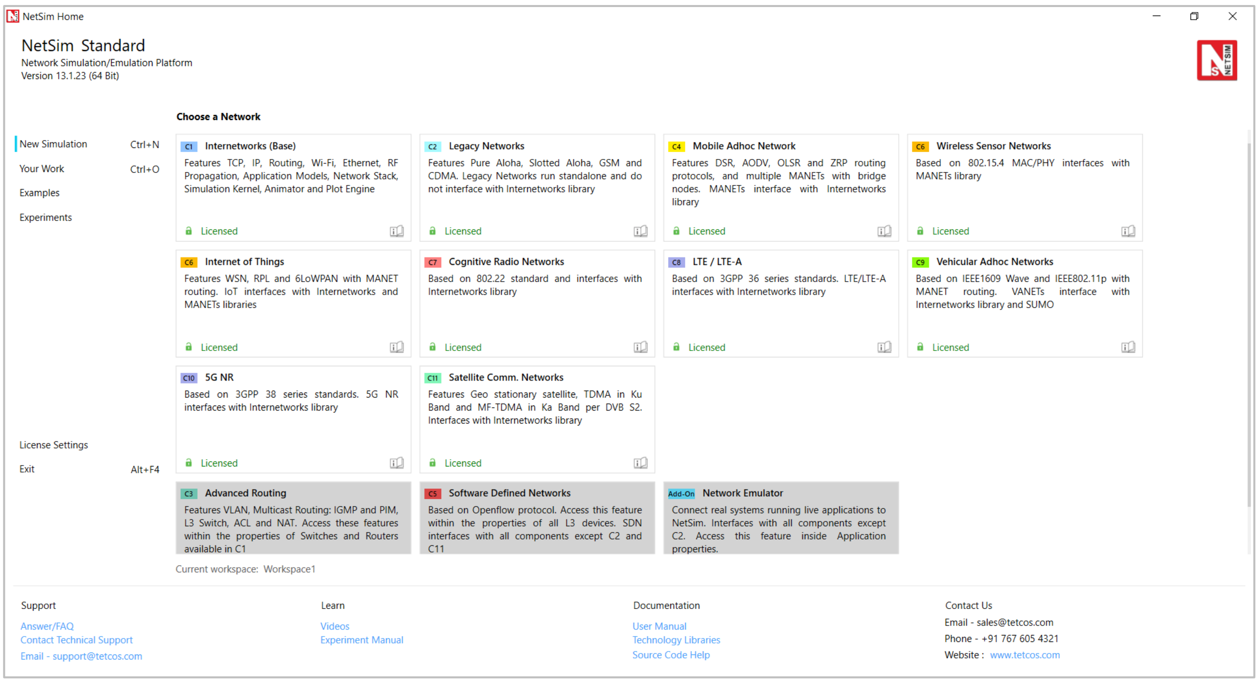
The Internetwork component is greyed out. You cannot clear the Internetworks component because Internetworks is a base component that is required for all the other components to work.

6. **Exit:** Use this option to close opened Netsim tabs.
7. **Documentation:** Use this section to open the following NetSim help documents: These include the ***User Manual*** which consists of complete description about all the features in NetSim and how it can be used by the end users, the ***Technology Libraries*** which provides users with an access to a detailed description of various Network Technologies present in NetSim through individual pdf files, and ***Source code help*** which comes along with Standard and Pro Versions of NetSim, allows users to gain a better understanding of the underlying code structure for in-depth analysis.
8. **Learn:** Use this section to learn more about the software which includes the following: ***Videos*** section can be used to view videos related to NetSim in **TETCOS LLP** YouTube channel. This channel helps users by providing frequent updates on what's new in NetSim, topics related to various network technologies covering different versions of NetSim, and monthly webinars. The ***Experiments Manual*** section grants you access to a well-designed experiments manual covering various networking concepts which helps users to easily understand different networks and also gain ideas to carry out their own experimentations in NetSim.
9. **Support:** Use this section to reach **TETCOS LLP** helpdesk. ***Contact Technical Support*** link can be used to raise a ***trouble ticket***, you can also write to us via ***Email*** to ***support@tetcos.com***, and ***Answers/FAQ*** link grants you access to our **Knowledge Base** which contains answers to all your questions most of the time. Users can utilize the wealth of information present in it, which are further classified into the following: FAQs, Technologies/Protocols, Modelling/UI/Results, and Writing your own code in NetSim.
10. **Contact Us:** Use this section to contact us and know more information about our product. You can write to us via ***Email*** to ***sales@tetcos.com*** or contact us via ***Phone*** to our official number **+91 76760 54321**.

Website: Use this link ***www.tetcos.com*** to visit our website which consists of vast information that will assist you through all walks of NetSim.

3.1.1 Creating "New" Simulations

The Simulation window loads up once user selects the desired network technology from the New Menu. Click on New Simulation and select the desired kind of network to simulate.


NetSim Home

NetSim Standard
Network Simulation/Emulation Platform
Version 13.1.23 (64 Bit)

New Simulation Ctrl+N
Your Work Ctrl+O
Examples
Experiments
License Settings
Exit Alt+F4

Choose a Network

C1 **Internetworks (Base)**
Features TCP, IP, Routing, Wi-Fi, Ethernet, RF Propagation, Application Models, Network Stack, Simulation Kernel, Animator and Plot Engine
Licensed

C2 **Legacy Networks**
Features Pure Aloha, Slotted Aloha, GSM and CDMA. Legacy Networks run standalone and do not interface with Internetworks library
Licensed

C4 **Mobile Adhoc Network**
Features DSR, AODV, OLSR and ZRP routing protocols, and multiple MANETs with bridge nodes. MANETs interface with Internetworks library
Licensed

C6 **Wireless Sensor Networks**
Based on 802.15.4 MAC/PHY interfaces with MANETs library
Licensed

C6 **Internet of Things**
Features WSN, RPL and 6LoWPAN with MANET routing. IoT interfaces with Internetworks and MANETs libraries
Licensed

C7 **Cognitive Radio Networks**
Based on 802.22 standard and interfaces with Internetworks library
Licensed

C8 **LTE / LTE-A**
Based on 3GPP 36 series standards. LTE/LTE-A interfaces with Internetworks library
Licensed

C9 **Vehicular Adhoc Networks**
Based on IEEE1609 Wave and IEEE802.11p with MANET routing. VANETs interface with Internetworks library and SUMO
Licensed

C10 **5G NR**
Based on 3GPP 38 series standards. 5G NR interfaces with Internetworks library
Licensed

C11 **Satellite Comm. Networks**
Features Geo stationary satellite, TDMA in Ku Band and MF-TDMA in Ka Band per DVB S2. Interfaces with Internetworks library
Licensed

C3 **Advanced Routing**
Features VLAN, Multicast Routing: IGMP and PIM, L3 Switch, ACL and NAT. Access these features within the properties of Switches and Routers available in C1

C5 **Software Defined Networks**
Based on Openflow protocol. Access this feature within the properties of all L3 devices. SDN interfaces with all components except C2 and C11

Add-On **Network Emulator**
Connect real systems running live applications to NetSim. Interfaces with all components except C2. Access this feature inside Application properties.

Current workspace: Workspace1

Support
Answer/FAQ
Contact Technical Support
Email - support@tetcos.com

Learn
Videos
Experiment Manual

Documentation
User Manual
Technology Libraries
Source Code Help

Contact Us
Email - sales@tetcos.com
Phone - +91 767 605 4321
Website : www.tetcos.com

Figure 3-6: NetSim Home Screen

Save

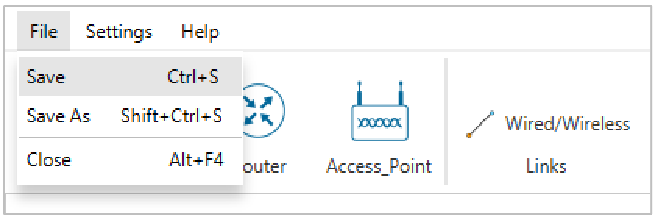


Figure 3-7: To save experiment, Select File >Save. Save As etc

To save experiment, select File → Save, then specify the Experiment Name, Description (Optional) and click Save. The short cut for the same is Ctrl + S.

Save as

To save an already existing/saved experiment by a different name after performing required modifications/changes to it (without overwriting the existing copy), Save As option can be used. Select File → Save As, then specify the Experiment Name, Description (Optional) and click Save. The short cut for the same is Shift + Ctrl + S and F12.

3.1.2 Environment Settings

The settings menu provides user's access to the simulation environment settings.

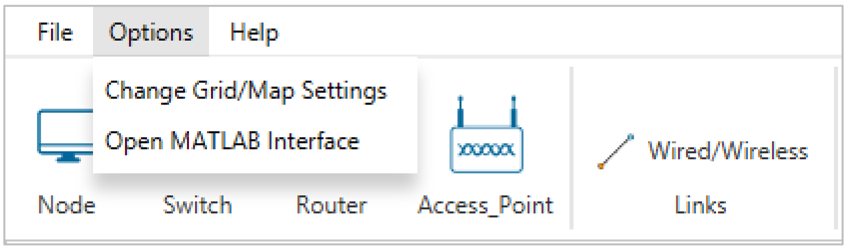
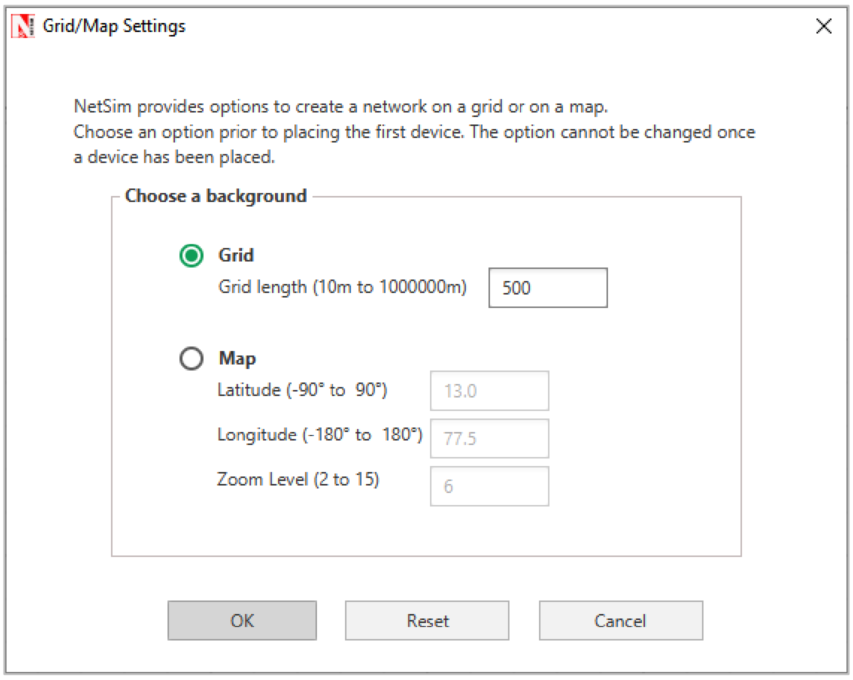


Figure 3-8: Environment settings

The Environment Settings window is used to switch between Grid View and Map View backgrounds in supported network technologies. For Grid view, users can configure the Grid environment length in meters as shown Figure 3-9.


Grid/Map Settings

NetSim provides options to create a network on a grid or on a map.
Choose an option prior to placing the first device. The option cannot be changed once a device has been placed.

Choose a background

[✓ Grid]
Grid length (10m to 1000000m) 500

Map
Latitude (-90° to 90°) 13.0
Longitude (-180° to 180°) 77.5
Zoom Level (2 to 15) 6

Figure 3-9: Grid View Setting

3.1.2.1 The Gird

The Grid coordinate system has its origin at the upper left of the drawing area, and positive Y is down while positive X is to the right.

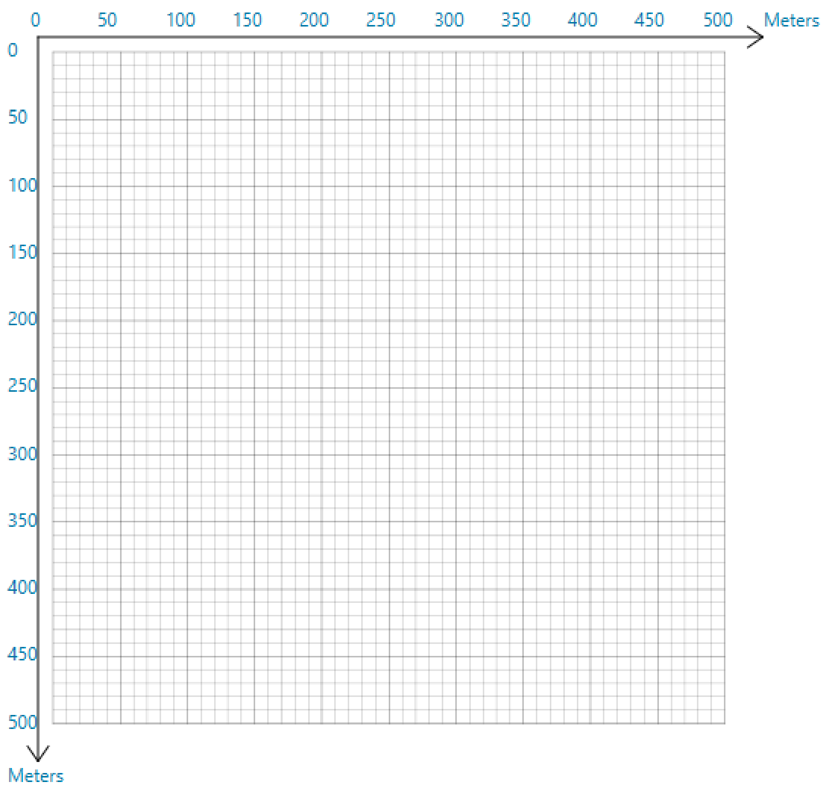
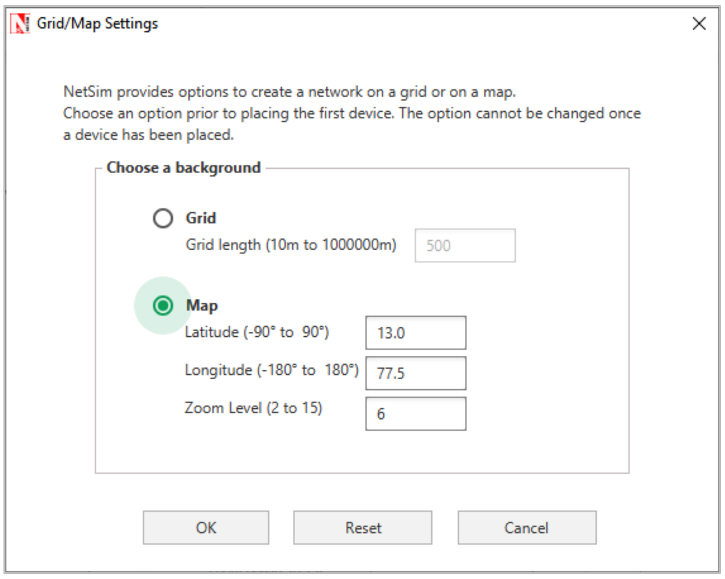


Figure 3-10: NetSim Design window

For Map view users can configure the latitude and longitude respectively.



NetSim provides options to create a network on a grid or on a map.
Choose an option prior to placing the first device. The option cannot be changed once a device has been placed.

Choose a background

Grid
Grid length (10m to 1000000m) 500

[✓ Map]
Latitude (-90° to 90°) 13.0
Longitude (-180° to 180°) 77.5
Zoom Level (2 to 15) 6

Figure 3-11: Map View Setting

Users can zoom in and out of the map to add devices in specific geographical locations.

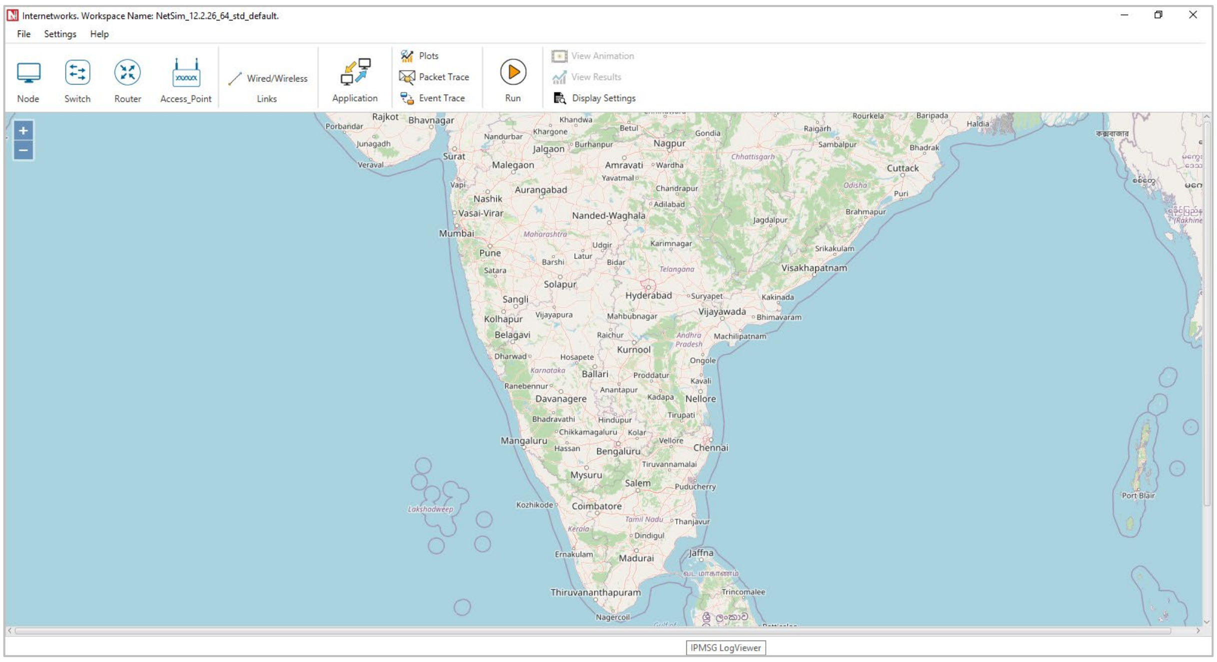


Figure 3-12: Map Design window

Learn

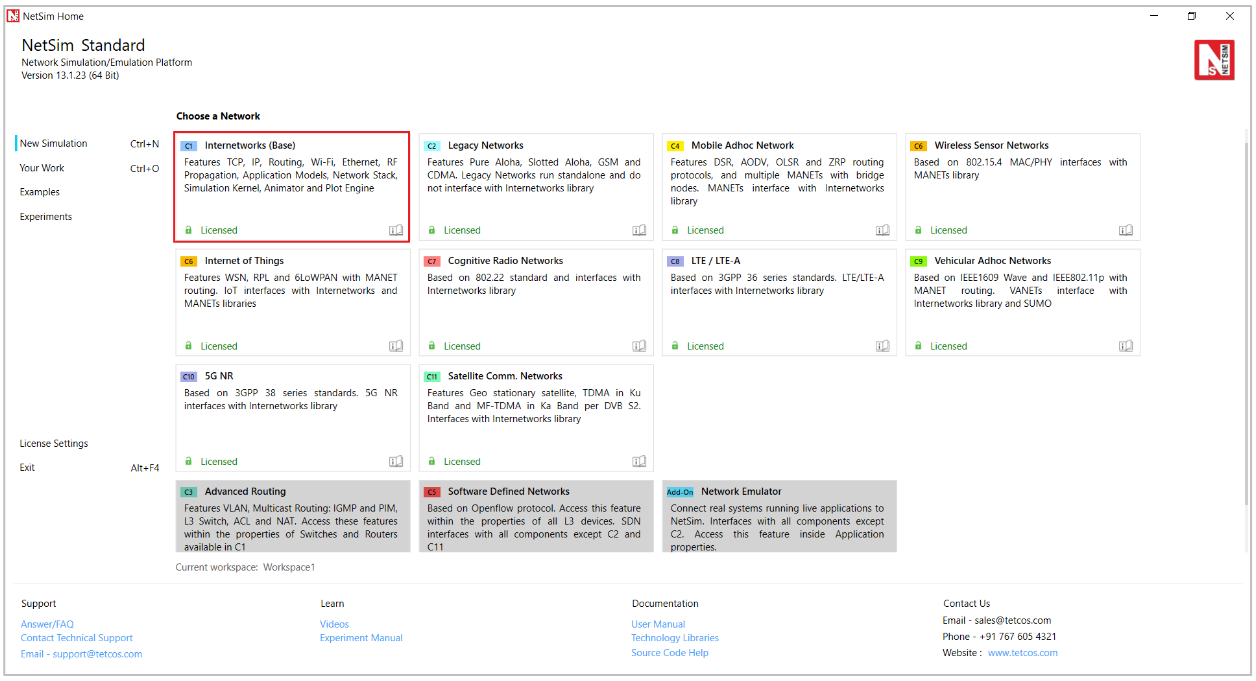
This menu contains link to NetSim Videos on **TETCOS LLP** YouTube Channel and NetSim Experiments manual.

Documentation

This menu contains link to NetSim User Manual, Technology Libraries and NetSim Source Code Help.

3.2 Modeling and Simulating a simple network

This section will demonstrate how to create a basic network scenario and analyze the results. Let us consider Internetworks. To create a new scenario, Go to **New Simulation → Internetworks** as shown below Figure 3-13.



NetSim Home

NetSim Standard
Network Simulation/Emulation Platform
Version 13.1.23 (64 Bit)

Choose a Network

New Simulation	Ctrl+N
Your Work	Ctrl+O
Examples	
Experiments	
License Settings	
Exit	Alt+F4

C1 Internetworks (Base)
Features TCP, IP, Routing, Wi-Fi, Ethernet, RF Propagation, Application Models, Network Stack, Simulation Kernel, Animator and Plot Engine
Licensed

C2 Legacy Networks
Features Pure Aloha, Slotted Aloha, GSM and CDMA. Legacy Networks run standalone and do not interface with Internetworks library
Licensed

C4 Mobile Adhoc Network
Features DSR, AODV, OLSR and ZRP routing protocols, and multiple MANETs with bridge nodes. MANETs interface with Internetworks library
Licensed

C6 Wireless Sensor Networks
Based on 802.15.4 MAC/PHY interfaces with MANETs library
Licensed

C6 Internet of Things
Features WSN, RPL and 6LoWPAN with MANET routing. IoT interfaces with Internetworks and MANETs libraries
Licensed

C7 Cognitive Radio Networks
Based on 802.22 standard and interfaces with Internetworks library
Licensed

C8 LTE / LTE-A
Based on 3GPP 36 series standards. LTE/LTE-A interfaces with Internetworks library
Licensed

C9 Vehicular Adhoc Networks
Based on IEEE1609 Wave and IEEE802.11p with MANET routing. VANETs interface with Internetworks library and SUMO
Licensed

C10 5G NR
Based on 3GPP 38 series standards. 5G NR interfaces with Internetworks library
Licensed

C11 Satellite Comm. Networks
Features Geo stationary satellite, TDMA in Ku Band and MF-TDMA in Ka Band per DVB S2. Interfaces with Internetworks library
Licensed

C3 Advanced Routing
Features VLAN, Multicast Routing: IGMP and PIM, L3 Switch, ACL and NAT. Access these features within the properties of Switches and Routers available in C1

C5 Software Defined Networks
Based on Openflow protocol. Access this feature within the properties of all L3 devices. SDN interfaces with all components except C2 and C11

Add-On Network Emulator
Connect real systems running live applications to NetSim. Interfaces with all components except C2. Access this feature inside Application properties.

Current workspace: Workspace1

Support
Answer/FAQ
Contact Technical Support
Email - support@tetcos.com

Learn
Videos
Experiment Manual

Documentation
User Manual
Technology Libraries
Source Code Help

Contact Us
Email - sales@tetcos.com
Phone - +91 767 605 4321
Website : www.tetcos.com

Figure 3-13: NetSim Home Screen

3.2.1 Creating a Network scenario

In this example, a network with two subnets is designed. Let us say the subnet 1 consists of two wired nodes connected via a Switch and the other subnet consists of one wired node. Both the subnets are connected using a Router. Traffic in the Network flows from a wired node in subnet 1 to the wired node in subnet 2.

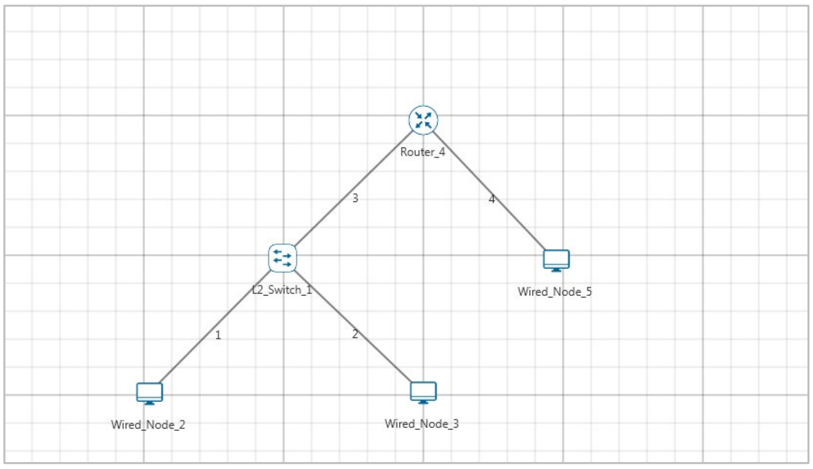


Figure 3-14: Network Topology in this experiment

Perform the following steps to create this network design.

Step 1: Drop the devices. Click on Node icon and select → Wired Node.

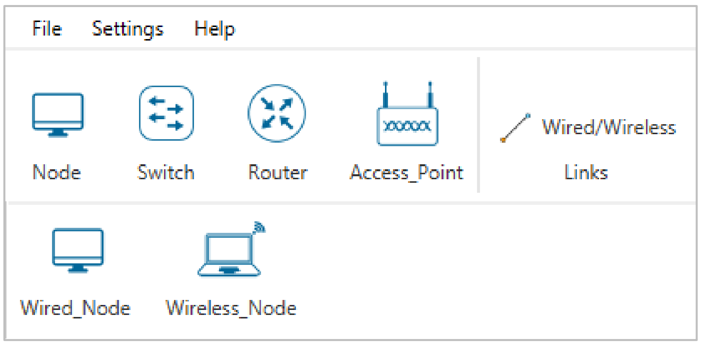


Figure 3-15: Internetworks Device Palette in GUI

Click on the environment (the grid) where you want the Wired Node to be placed. In this way, place two more wired nodes. Similarly, to place a Switch and a Router, click on the respective device and click on the environment at the desired location.

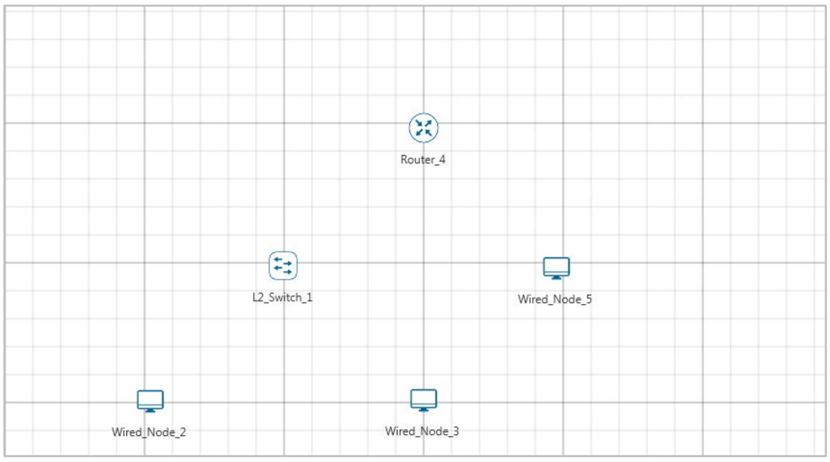


Figure 3-16: Dropped Devices on GUI

Note that NetSim takes the (x, y) position of any device on the grid is the position of top left corner of the icon and not the center of the icon.

Step 2: Connecting devices: Select the link and then left click on one device, free the mouse button, then click on the second device and free the mouse button. The wired links may disappear if you right click anywhere in the environment. Clicking and dragging without freeing the mouse pointer would displace the device in the environment.

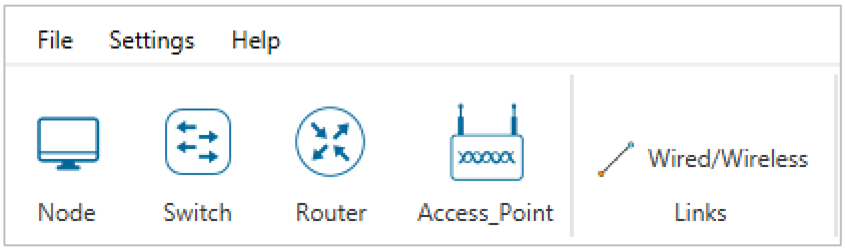


Figure 3-17: To Connect devices select wired/wireless links

For example, select link and the click on Switch followed by router to connect them. In this manner, continue to link all devices.

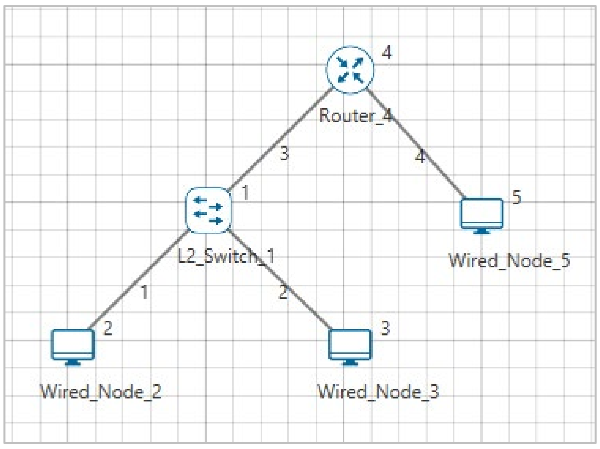


Figure 3-18: Network Topology

3.2.2 Configuring devices and links in the scenario

Step 1: To configure any device, right click on the device and select properties as shown Figure 3-19.

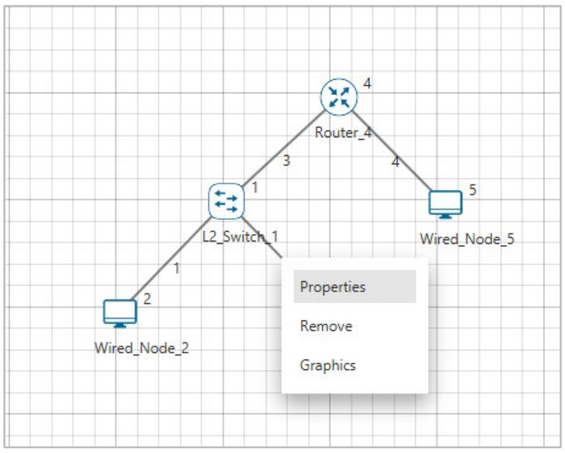
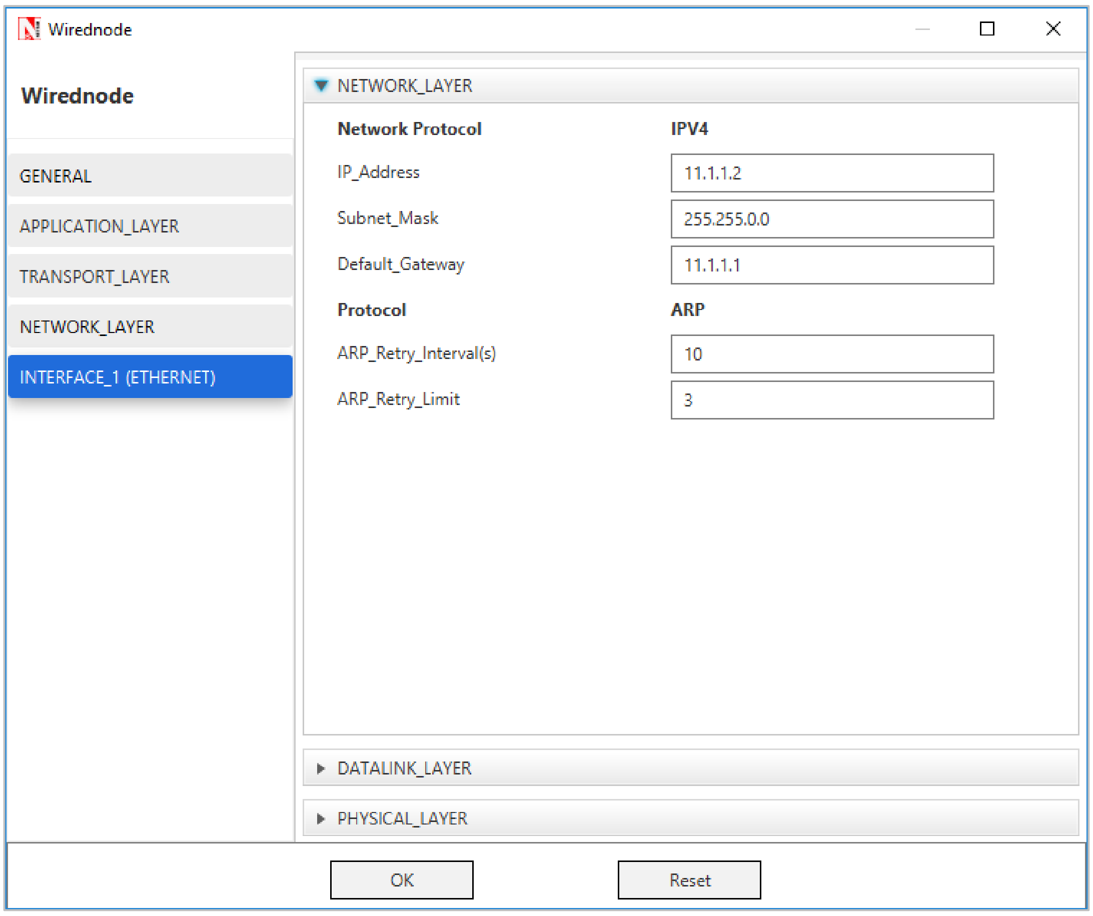


Figure 3-19: Right click on the device and select properties

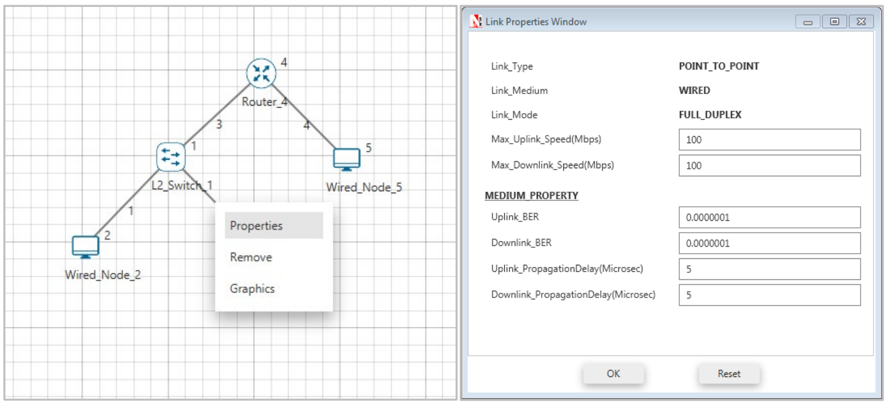
The default properties of any device can be modified per requirement. Then click on OK.



Network Protocol	**IPV4**
IP_Address	11.1.1.2
Subnet_Mask	255.255.0.0
Default_Gateway	11.1.1.1
Protocol	**ARP**
ARP_Retry_Interval(s)	10
ARP_Retry_Limit	3

Figure 3-20: Network layer Properties window for wired node

Step 2: To configure the links, right click on any Link and select Properties as shown Figure 3-21.



Link_Type	POINT_TO_POINT
Link_Medium	WIRED
Link_Mode	FULL_DUPLEX
Max_Uplink_Speed(Mbps)	100
Max_Downlink_Speed(Mbps)	100
MEDIUM_PROPERTY	
Uplink_BER	0.0000001
Downlink_BER	0.0000001
Uplink_PropagationDelay(Microsec)	5
Downlink_PropagationDelay(Microsec)	5

Figure 3-21: Wired Link properties window for links

3.2.3 Display Settings

In NetSim, users can Turn-On or Turn-Off display information such as IP Address of the devices, link speed etc. For doing this click on Display settings as shown below Figure 3-22.

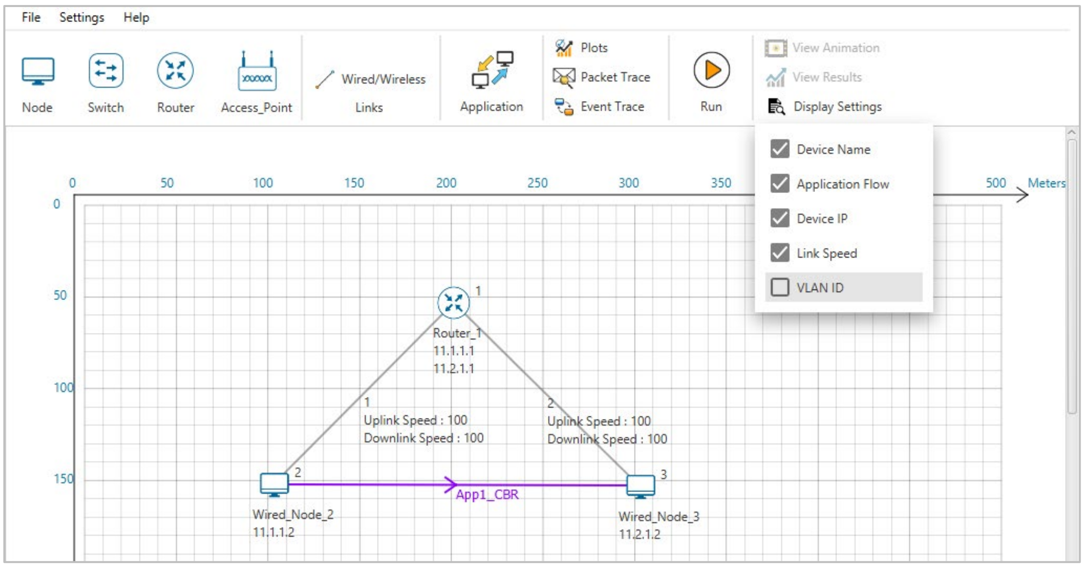


Figure 3-22: Turn-On or Turn-Off display information such as IP Address of the devices, link speed etc

In NetSim the device ID serves as a “device identifier” while the IP Address is an “Interface identifier”

3.2.4 Copy/Paste

In NetSim simple copy paste can be used. Using this feature users can copy all the properties of a device and create a new device with similar properties.

Right click on the device, click on copy and then right click and click paste. The sequence is shown below Figure 3-23/Figure 3-24/Figure 3-25/Figure 3-26/Figure 3-27.

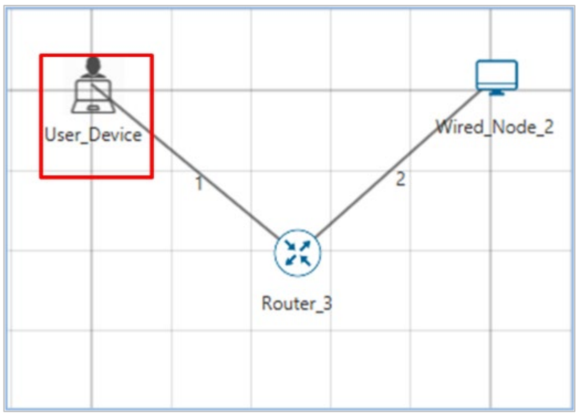


Figure 3-23: Devices present on GUI

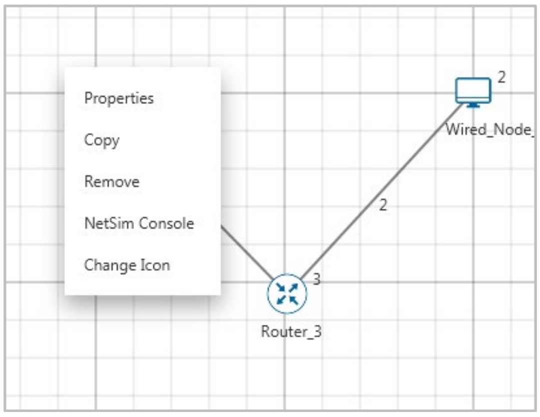


Figure 3-24: Right click on the user device and Select copy

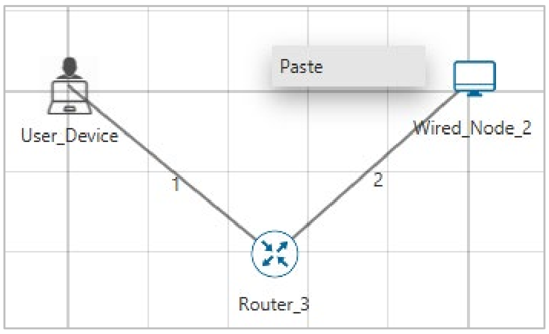


Figure 3-25: Right click on the GUI and Select Paste

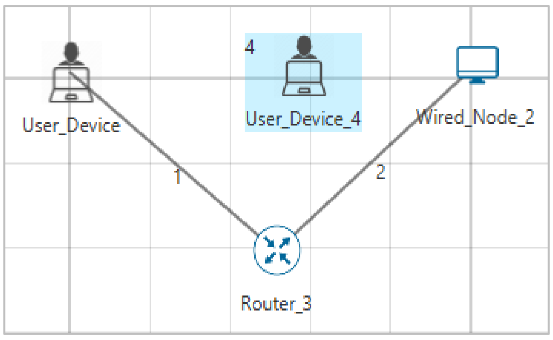


Figure 3-26: Devices Pasted in GUI

Remove in the device options, is used to delete the device from the grid environment. Given below is an example of removing the device User_Device_4 which was previously pasted.

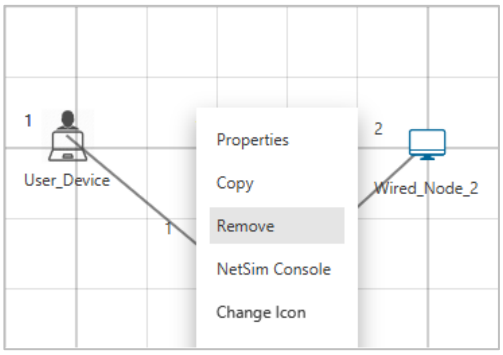


Figure 3-27: Right click on User_Device_4 and Select Remove

3.2.5 Modeling Application Traffic

After the network is configured, user needs to model traffic from Wired Node 2 to Wired Node 3. This is done using the application icon. Click on the Application icon present on the ribbon as shown below Figure 3-28.

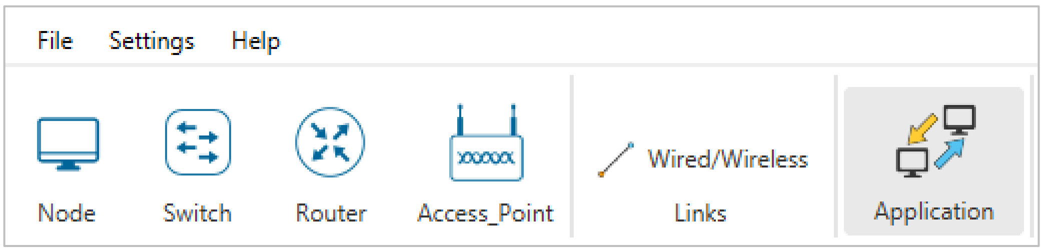
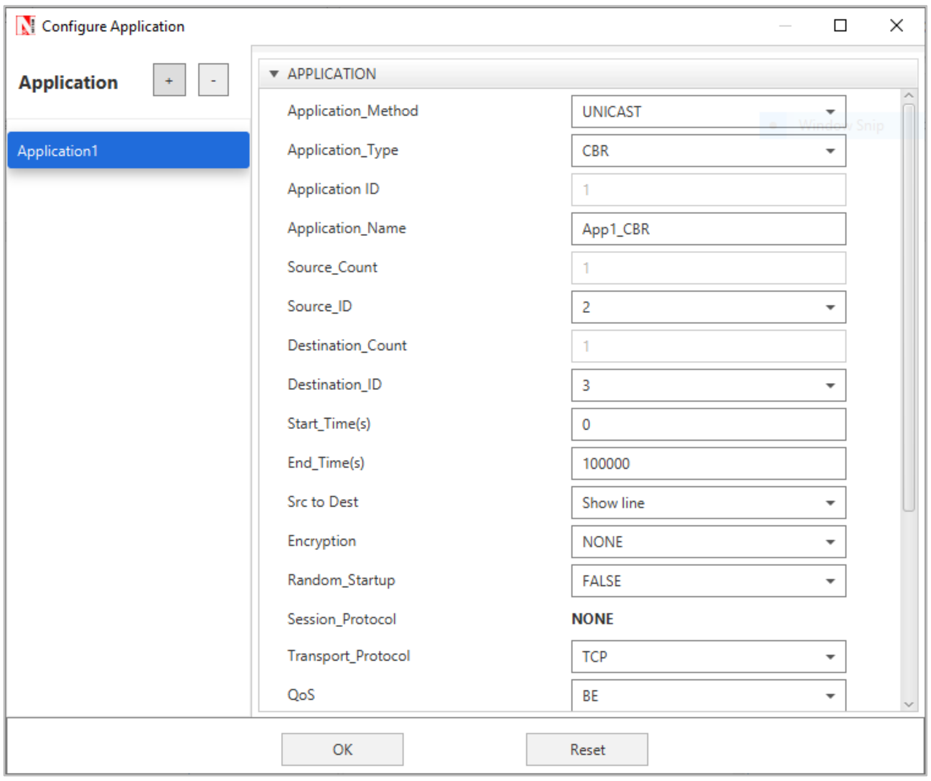


Figure 3-28: Select Application icon present on ribbon

In screen shot shown below the Application type is set to CBR, Source_ID is 2 and Destination_ID is 3. Click on **OK.**


Configure Application

Application + -

Application1

APPLICATION

Application_Method	UNICAST
Application_Type	CBR
Application ID	1
Application_Name	App1_CBR
Source_Count	1
Source_ID	2
Destination_Count	1
Destination_ID	3
Start_Time(s)	0
End_Time(s)	100000
Src to Dest	Show line
Encryption	NONE
Random_Startup	FALSE
Session_Protocol	NONE
Transport_Protocol	TCP
QoS	BE

OK Reset

Figure 3-29: Application Configuration window

3.2.6 Logging Packet/ Event Trace

Packet and Event Trace files are useful for detailed simulation analysis. By default, these are not enabled since it slows down the simulation. To enable logging of Packet Trace / Event Trace click on the icon in the tool bar as shown below. Set the file name and select the required attributes to be logged. For more information, please refer **sections 8.4** and **8.5** respectively.

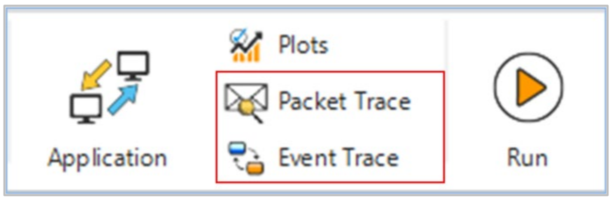


Figure 3-30: Packet Trace and Event Trace options present on ribbon

3.2.7 Run Simulation

For simulating the network scenario created, click on Run Simulation present in the Ribbon.

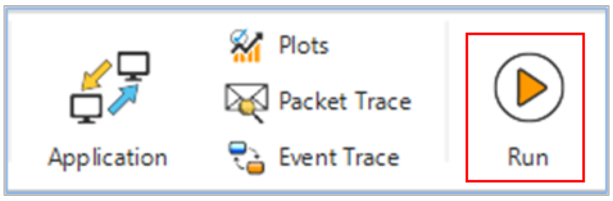
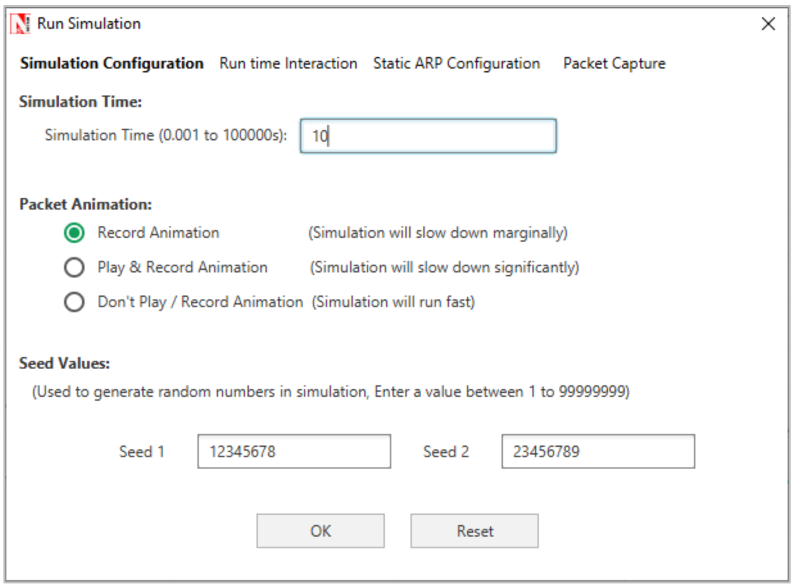


Figure 3-31: Run Simulation icon in the Ribbon

Set the Simulation Time to 10 seconds. Click on OK.


Run Simulation

Simulation Configuration | Run time Interaction | Static ARP Configuration | Packet Capture

Simulation Time:

Simulation Time (0.001 to 100000s): 10

Packet Animation:

- [✓ Record Animation] (Simulation will slow down marginally)
- Play & Record Animation (Simulation will slow down significantly)
- Don't Play / Record Animation (Simulation will run fast)

Seed Values:

(Used to generate random numbers in simulation, Enter a value between 1 to 99999999)

Seed 1: 12345678 Seed 2: 23456789

Figure 3-32: Run Simulation window

3.3 Saving & Opening experiments and Printing results

3.3.1 Opening Saved Experiments:

Click on **Your work** (Ctrl+O).

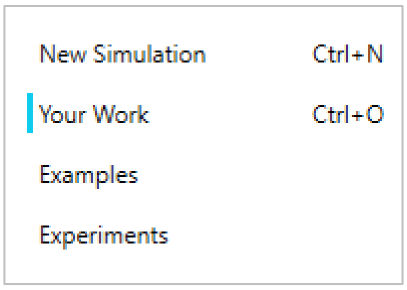


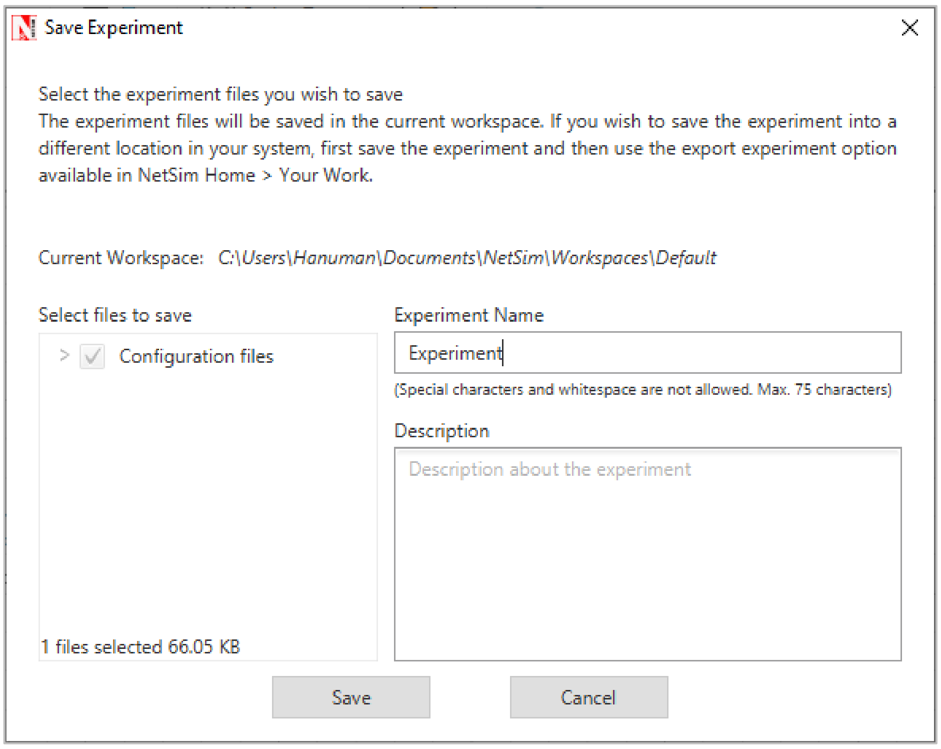
Figure 3-33: Select Your work

Click on the saved experiment file you wish to open.

3.3.2 Saving an Experiment

During Simulation: Users can save by using the short cut CTRL + S.

After Simulation: From Network Window: Click on File > Save button on the top left. Next**,** specify the Experiment Name, Description (Optional) and click on Save.


Save Experiment

Select the experiment files you wish to save
The experiment files will be saved in the current workspace. If you wish to save the experiment into a different location in your system, first save the experiment and then use the export experiment option available in NetSim Home > Your Work.

Current Workspace: *C:\Users\Hanuman\Documents\NetSim\Workspaces\Default*

Select files to save

Configuration files

1 files selected 66.05 KB

Experiment Name

Experiment

(Special characters and whitespace are not allowed. Max. 75 characters)

Description

Description about the experiment

Figure 3-34: Save popup window

Upon saving several files would get saved inside the folder, including:

- Configuration file (xml) & metrics file (xml)
- Trace Files (csv), if enabled, and
- Plot data (txt)

3.4 NetSim Keyboard Shortcuts

NetSim keyboard shortcuts can be used for frequently performed tasks. The keyboard shortcuts that are currently supported are listed in the table below Table 3-1.

Keys	Function
Home Screen	
Ctrl + N	Open a New Network
Ctrl + O	Open a Saved Network
Design Window (Any Network)	
Ctrl + C	Copy
Ctrl + V	Paste
Ctrl + R	Open Run Simulation Window
Ctrl + S	Save the Experiment
Shift + Ctrl + S	Save As (To Save under different name)
Ctrl + P	Open Image/Screenshot of the network scenario that is designed in the GUI
Alt+ F4	Close Window
F1	User Manual Help
Ctrl + '+/-'	Zoom In/Zoom Out
Mouse Click (Left)	Select a device
Ctrl + A	Select All devices in the design environment

Ctrl + Mouse Click (Left) and Drag	Select devices within a selected area
Delete	Deletes the selected devices in the Environment along with any links it may have
Simulation Console	
Ctrl + C	Terminates Simulation in Mid way. Results will be calculated till the time at which the simulation is terminated
Packet Animation Window	
Space bar	To Play/Pause animation

Table 3-1: NetSim keyboard shortcuts

3.5 NetSim Interactive Simulation

NetSim allows users to interact with the simulation at runtime via a socket or through a file. User Interactions make simulation more realistic by allowing command execution to view/modify certain device parameters during runtime.

This section will demonstrate how to perform Interactive simulation for a simple network scenario. Let us consider Internetworks. To create a new scenario, go to New → Internetworks. Click & drop Wired Nodes and Router onto the Simulation Environment and link them as shown below Figure 3-35 or otherwise Open the scenario for Interactive Simulation which is available in "<NetSim Install Dir>\Docs\ Sample_Configuration\Internetworks\Interactive Simulation".

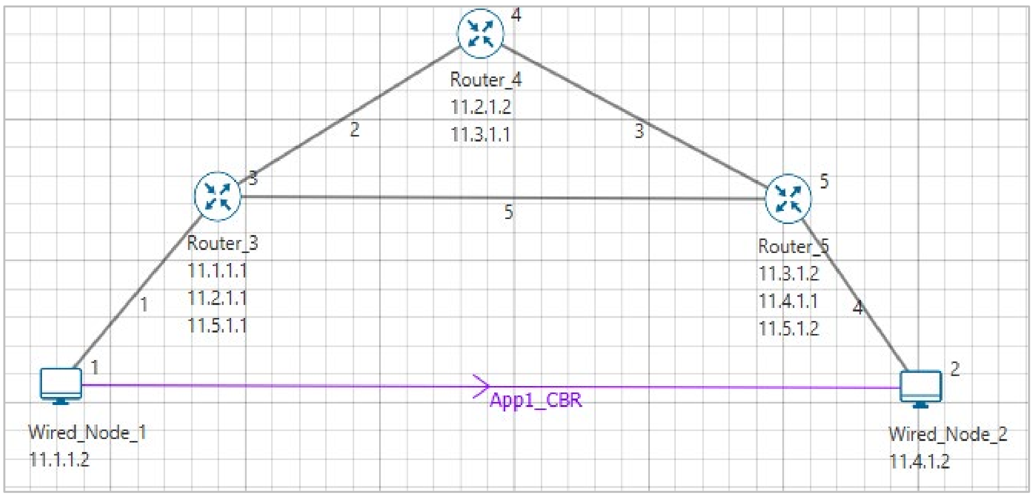


Figure 3-35: Network Topology

- Click on Application icon present in the top ribbon and set the Application type as CBR. The Source_Id is 1 and Destination_Id is 2.
- Set Start Time as 30 Sec
- Enable Plots and Packet trace options
- Click on run simulation option and In the Run time Interaction tab set Interactive Simulation as True and click on Accept as shown below Figure 3-36.

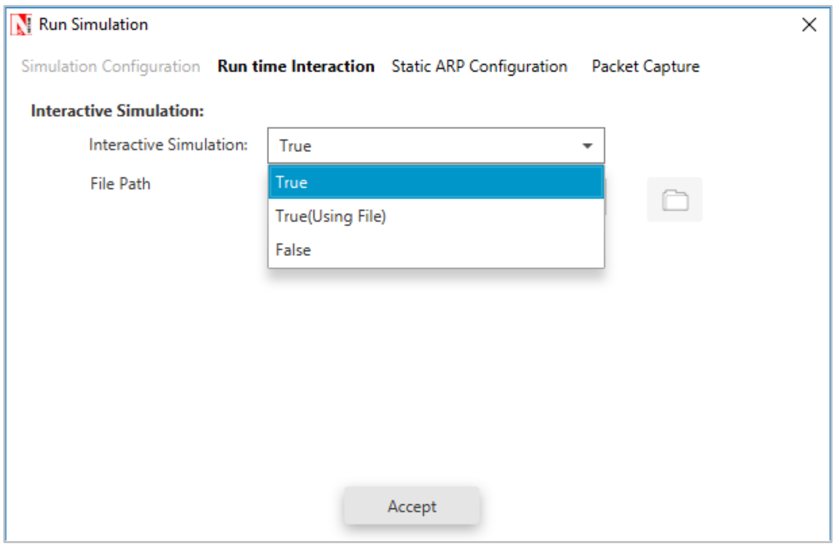
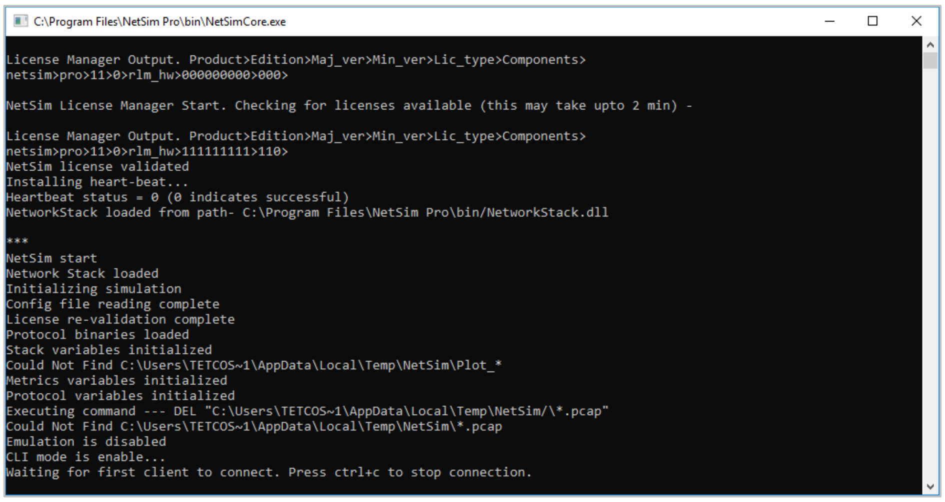


Figure 3-36: Run time Interaction tab set Interactive Simulation as True

- Click on run simulation and set Simulation Time as 500 sec. (It is recommended to specify a longer simulation time to ensure that there is sufficient time for the user to execute the various commands and see the effect of that before Simulation ends) and click OK.
- Simulation (NetSimCore.exe) will start running and will display a message “waiting for first client to connect” as shown below Figure 3-37.



```
License Manager Output. Product>Edition>Maj_ver>Min_ver>Lic_type>Components>
netsim>pro>11>0>rlm_hw>000000000>000>

NetSim License Manager Start. Checking for licenses available (this may take upto 2 min) -

License Manager Output. Product>Edition>Maj_ver>Min_ver>Lic_type>Components>
netsim>pro>11>0>rlm_hw>111111111>110>
NetSim license validated
Installing heart-beat...
Heartbeat status = 0 (0 indicates successful)
NetworkStack loaded from path- C:\Program Files\NetSim Pro\bin/NetworkStack.dll

***
NetSim start
Network Stack loaded
Initializing simulation
Config file reading complete
License re-validation complete
Protocol binaries loaded
Stack variables initialized
Could Not Find C:\Users\TETCOS~1\AppData\Local\Temp\NetSim\Plot_*
Metrics variables initialized
Protocol variables initialized
Executing command --- DEL "C:\Users\TETCOS~1\AppData\Local\Temp\NetSim/\*.pcap"
Could Not Find C:\Users\TETCOS~1\AppData\Local\Temp\NetSim\*.pcap
Emulation is disabled
CLI mode is enable...
Waiting for first client to connect. Press ctrl+c to stop connection.
```

Figure 3-37: Waiting for first client to connect

- After Simulation window opens, goto Network scenario and right click on Router_3 or any other node and select NetSim Console option as shown Figure 3-38.

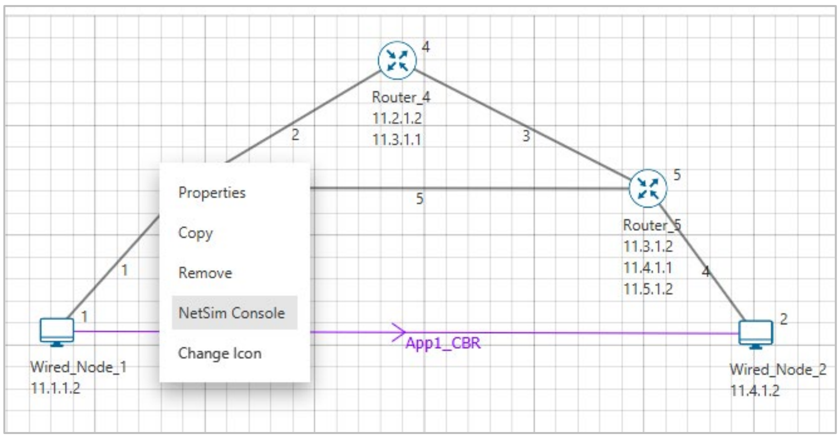
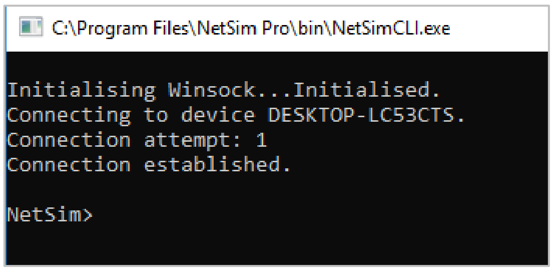


Figure 3-38: Select NetSim Console option

- Now client (NetSimCLI.exe) will start running and it will try to establish connection with NetSimCore.exe. After connection is established, the window will look similar like this shown below Figure 3-39.



C:\Program Files\NetSim Pro\bin\NetSimCLI.exe

```
Initialising Winsock...Initialised.
Connecting to device DESKTOP-LC53CTS.
Connection attempt: 1
Connection established.

NetSim>
```

Figure 3-39: Connection is established

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

<u>Note: Commands are not a case sensitive</u>

3.5.1 Simulation specific (Not applicable for file based interactive simulation)

- Pause
- PauseAt
- Continue
- Stop
- Exit
- Reconnect

Pause: To pause the currently running simulation

PauseAt: To pause the currently running simulation with respect to particular time (Ex: To Pause simulation at 70.2 sec use command as **PauseAt 70.2**)

Continue: To start the currently paused simulation

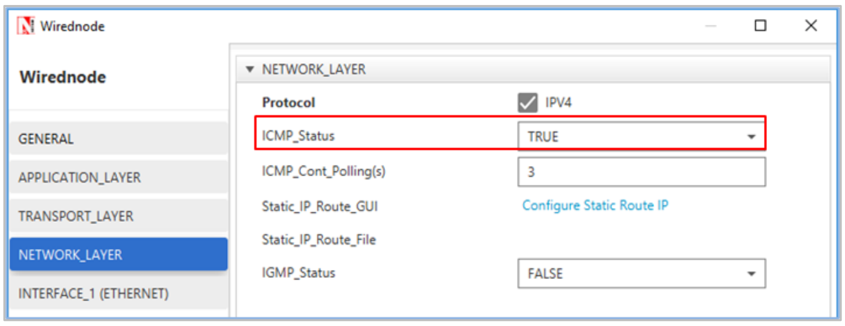
Stop: To stop the currently running simulation (NetSimCore.exe)

Exit: To exit from the client (NetSimCLI.exe)

Reconnect: To reconnect client (NetSimCLI.exe) to simulation (NetSimCore.exe) when we rerun simulation again

3.5.2 Ping Command

- The ping command is one of the most often used networking utilities for troubleshooting network problems.
- You can use the ping command to test the availability of a networking device (usually a computer) on a network.
- When you ping a device, you send that device a short message, which it then sends back (the echo)
- If you receive a reply then the device is in Network, if you don't then the device is faulty, disconnected, switched off, incorrectly configured.
- You can use the ***ping*** cmd with an IP address or Device name.
- **ICMP_Status** should be set as True in all nodes(Wired_Node and Router)



Wirednode

Wirednode

GENERAL
APPLICATION_LAYER
TRANSPORT_LAYER
NETWORK_LAYER
INTERFACE_1 (ETHERNET)

NETWORK_LAYER

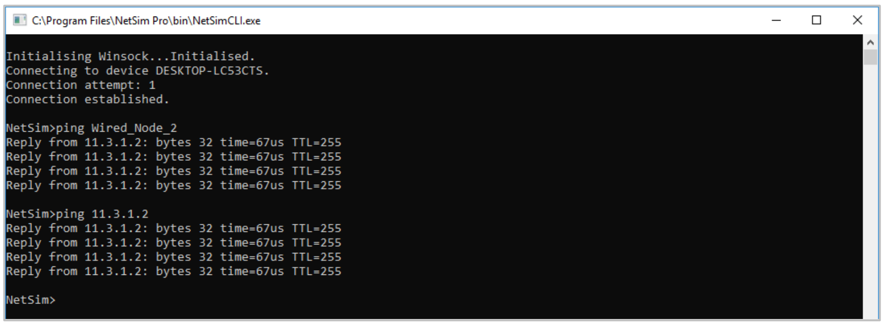
Protocol	✓ IPV4
ICMP_Status	TRUE
ICMP_Cont_Polling(s)	3
Static_IP_Route_GUI	Configure Static Route IP
Static_IP_Route_File	
IGMP_Status	FALSE

Figure 3-40: Set ICMP_Status to True in Network layer window.

- Right click on Wired_Node_1 and go to properties. Under General properties enable Wireshark Capture option as "Online"

```
Ping <IP address> e.g. ping 11.4.1.2
Ping <NodeName> e.g. ping Wired_Node_2
```

3.5.2.1 Ping Command Results

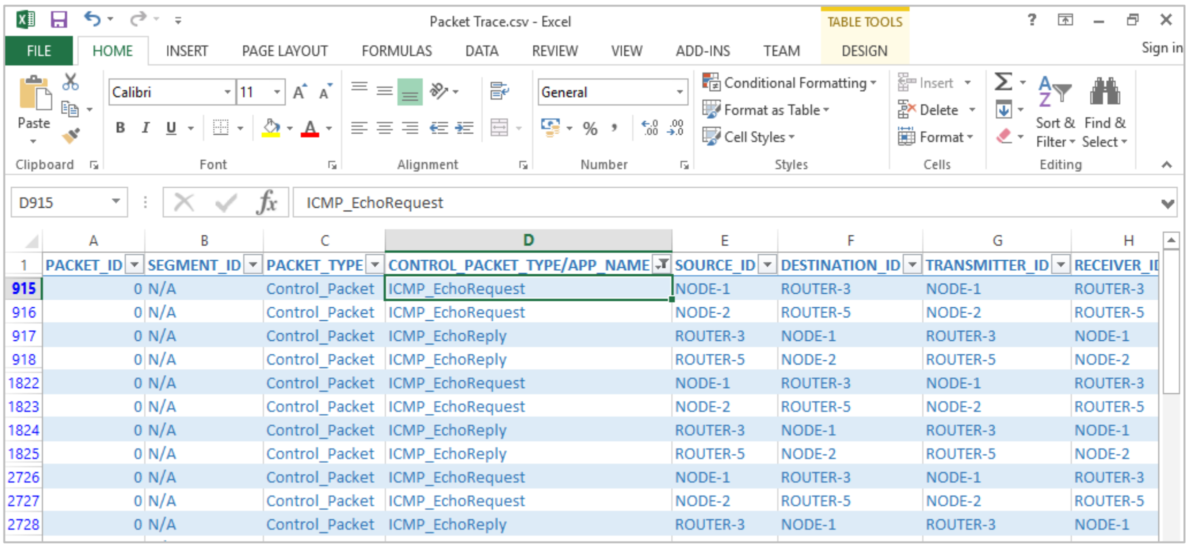


C:\Program Files\NetSim Pro\bin\NetSimCLI.exe

```
Initialising Winsock...Initialised.
Connecting to device DESKTOP-LC53CTS.
Connection attempt: 1
Connection established.

NetSim>ping Wired_Node_2
Reply from 11.3.1.2: bytes 32 time=67us TTL=255
Reply from 11.3.1.2: bytes 32 time=67us TTL=255
Reply from 11.3.1.2: bytes 32 time=67us TTL=255
Reply from 11.3.1.2: bytes 32 time=67us TTL=255

NetSim>ping 11.3.1.2
Reply from 11.3.1.2: bytes 32 time=67us TTL=255
Reply from 11.3.1.2: bytes 32 time=67us TTL=255
Reply from 11.3.1.2: bytes 32 time=67us TTL=255
Reply from 11.3.1.2: bytes 32 time=67us TTL=255

NetSim>
```

Figure 3-41: Pinging to Wired_Node_2

- After simulation open packet trace and filter ICMP_EchoRequest and ICMP_EchoReply from CONTROL_PACKET_TYPE/APP_NAME column

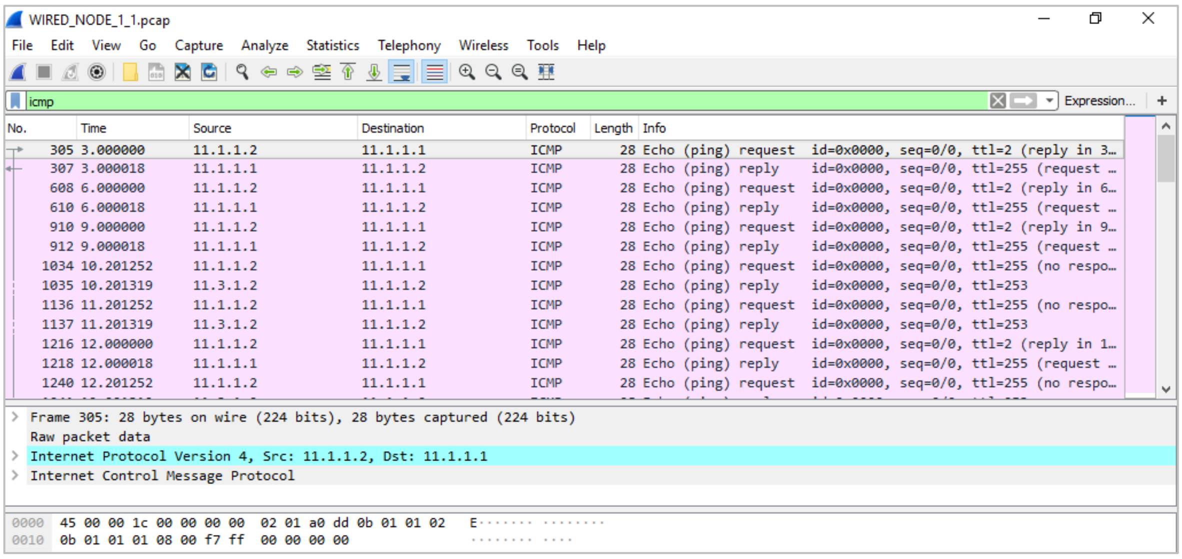


D915 | ICMP_EchoRequest

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

Figure 3-42: ICMP Control Packets in Packet Trace

- Open Wireshark and apply filter ICMP. We can see the ping request and reply packets in Wireshark.



WIRED_NODE_1_1.pcap

icmp

```
No.   Time       Source    Destination  Protocol Length Info
  305 3.000000   11.1.1.2  11.1.1.1     ICMP     28 Echo (ping) request  id=0x0000, seq=0/0, ttl=2 (reply in 3…
  307 3.000018   11.1.1.1  11.1.1.2     ICMP     28 Echo (ping) reply    id=0x0000, seq=0/0, ttl=255 (request …
  608 6.000000   11.1.1.2  11.1.1.1     ICMP     28 Echo (ping) request  id=0x0000, seq=0/0, ttl=2 (reply in 6…
  610 6.000018   11.1.1.1  11.1.1.2     ICMP     28 Echo (ping) reply    id=0x0000, seq=0/0, ttl=255 (request …
  910 9.000000   11.1.1.2  11.1.1.1     ICMP     28 Echo (ping) request  id=0x0000, seq=0/0, ttl=2 (reply in 9…
  912 9.000018   11.1.1.1  11.1.1.2     ICMP     28 Echo (ping) reply    id=0x0000, seq=0/0, ttl=255 (request …
 1034 10.201252  11.1.1.2  11.1.1.1     ICMP     28 Echo (ping) request  id=0x0000, seq=0/0, ttl=255 (no respo…
 1035 10.201319  11.3.1.2  11.1.1.2     ICMP     28 Echo (ping) reply    id=0x0000, seq=0/0, ttl=253
 1136 11.201252  11.1.1.2  11.1.1.1     ICMP     28 Echo (ping) request  id=0x0000, seq=0/0, ttl=255 (no respo…
 1137 11.201319  11.3.1.2  11.1.1.2     ICMP     28 Echo (ping) reply    id=0x0000, seq=0/0, ttl=253
 1216 12.000000  11.1.1.2  11.1.1.1     ICMP     28 Echo (ping) request  id=0x0000, seq=0/0, ttl=2 (reply in 1…
 1218 12.000018  11.1.1.1  11.1.1.2     ICMP     28 Echo (ping) reply    id=0x0000, seq=0/0, ttl=255 (request …
 1240 12.201252  11.1.1.2  11.1.1.1     ICMP     28 Echo (ping) request  id=0x0000, seq=0/0, ttl=255 (no respo…

> Frame 305: 28 bytes on wire (224 bits), 28 bytes captured (224 bits)
  Raw packet data
> Internet Protocol Version 4, Src: 11.1.1.2, Dst: 11.1.1.1
> Internet Control Message Protocol

0000  45 00 00 1c 00 00 00 00  02 01 a0 dd 0b 01 01 02   E······· ········
0010  0b 01 01 01 08 00 f7 ff  00 00 00 00               ········ ····
```

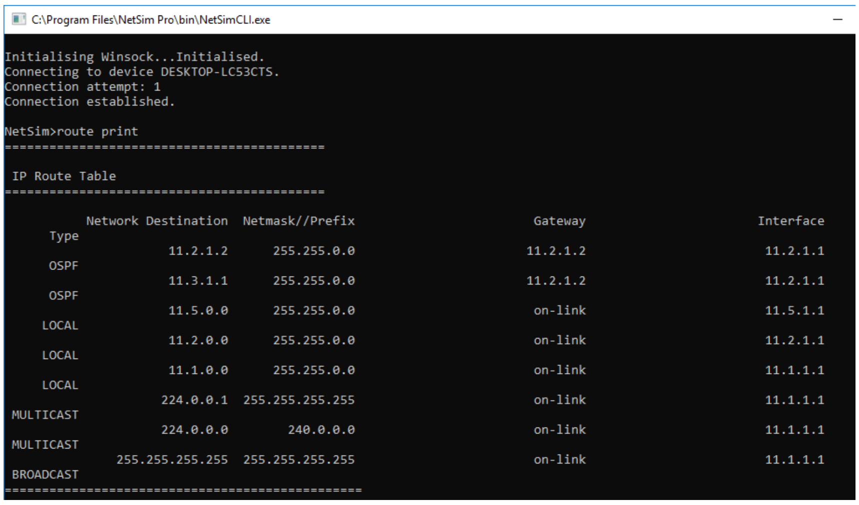
Figure 3-43: ICMP Control Packets in Wireshark

3.5.3 Route Commands

- route print
- route delete
- route add

In order to view the entire contents of the IP routing table, use following commands **route print.**

```
route print
```



```
C:\Program Files\NetSim Pro\bin\NetSimCLI.exe

Initialising Winsock...Initialised.
Connecting to device DESKTOP-LC53CTS.
Connection attempt: 1
Connection established.

NetSim>route print
==========================================

 IP Route Table
==========================================

          Network Destination  Netmask//Prefix                         Gateway                   Interface
     Type
                     11.2.1.2      255.255.0.0                         11.2.1.2                    11.2.1.1
     OSPF
                     11.3.1.1      255.255.0.0                         11.2.1.2                    11.2.1.1
     OSPF
                     11.5.0.0      255.255.0.0                          on-link                    11.5.1.1
    LOCAL
                     11.2.0.0      255.255.0.0                          on-link                    11.2.1.1
    LOCAL
                     11.1.0.0      255.255.0.0                          on-link                    11.1.1.1
    LOCAL
                    224.0.0.1  255.255.255.255                          on-link                    11.1.1.1
 MULTICAST
                    224.0.0.0        240.0.0.0                          on-link                    11.1.1.1
 MULTICAST
              255.255.255.255  255.255.255.255                          on-link                    11.1.1.1
 BROADCAST
================================================
```

Figure 3-44: Network Route Print

- You will see the routing table entries with network destinations and the gateways to which packets are forwarded when they are headed to that destination. Unless you've already added static routes to the table, everything you see here will be dynamically generated.
- In order to delete route in the IP routing table you will type a command using the following syntax

```
Route delete destination_network
```

- So, to delete the route with destination network 11.5.1.2, all we'd have to do is type this command.

```
route delete 11.5.1.2
```

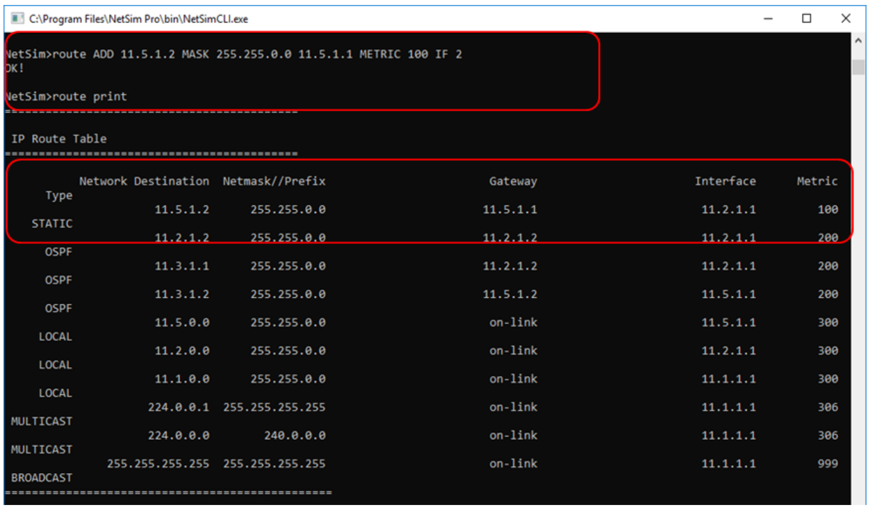
- To check whether route has been deleted or not check again using **route print** command.
- To add a static route to the table, you will type a command using the following syntax.

```
route ADD destination_network MASK subnet_mask gateway_ip METRIC metric_cost IF
interface_id
```

- So, for example, if you wanted to add a route specifying that all traffic bound for the 11.5.1.2 subnet went to a gateway at 11.5.1.1

```
route ADD 11.5.1.2 MASK 255.255.0.0 11.5.1.1 METRIC 100 IF 2
```

- If you were to use the **route print** command to look at the table now, you would see your new static route.



```
C:\Program Files\NetSim Pro\bin\NetSimCLI.exe
NetSim>route ADD 11.5.1.2 MASK 255.255.0.0 11.5.1.1 METRIC 100 IF 2
OK!

NetSim>route print
==========================================

 IP Route Table
==========================================

           Network Destination  Netmask//Prefix                  Gateway                Interface      Metric
     Type
                      11.5.1.2      255.255.0.0                 11.5.1.1                 11.2.1.1         100
   STATIC
                      11.2.1.2      255.255.0.0                 11.2.1.2                 11.2.1.1         200
     OSPF
                      11.3.1.1      255.255.0.0                 11.2.1.2                 11.2.1.1         200
     OSPF
                      11.3.1.2      255.255.0.0                 11.5.1.2                 11.5.1.1         200
     OSPF
                      11.5.0.0      255.255.0.0                  on-link                 11.5.1.1         300
    LOCAL
                      11.2.0.0      255.255.0.0                  on-link                 11.2.1.1         300
    LOCAL
                      11.1.0.0      255.255.0.0                  on-link                 11.1.1.1         300
    LOCAL
                     224.0.0.1  255.255.255.255                  on-link                 11.1.1.1         306
MULTICAST
                     224.0.0.0        240.0.0.0                  on-link                 11.1.1.1         306
MULTICAST
               255.255.255.255  255.255.255.255                  on-link                 11.1.1.1         999
BROADCAST
================================================
```

Figure 3-45: Route added into Network

Note: *Entry added in IP table by routing protocol continuously gets updated. If a user tries to remove a route via route delete command, there is always a chance that routing protocol will re-enter this entry again. Users can use ACL / Static route to override the routing protocol entry if required.*

3.5.4 ACL Configuration

Routers provide basic traffic filtering capabilities, such as blocking Internet traffic, with access control lists (ACLs). An ACL is a sequential list of permit or deny statements that apply to addresses or upper-layer protocols. These lists tell the router what types of packets to: permit or deny. When using an access-list to filter traffic, a permit statement is used to “allow” traffic, while a deny statement is used to “block” traffic.

ACL Commands

- To view ACL syntax use: ***acl print.***
- Before using ACL’s, we must first verify that acl option enabled. A common way to enable ACL use command: ***acl enable.***
- Enters configuration mode of ACL using: ***aclconfig***
- To view ACL Table: ***Print***
- To exit from ACL configuration use command: ***exit***
- To disable ACL use command: ***acl disable*** (use this command after exit from acl configuration)

To view ACL usage syntax use: **acl print**

```
[PERMIT, DENY] [INBOUND, OUTBOUND, BOTH] PROTO SRC DEST SPORT DPORT
IFID
```

3.5.4.1 Step to Configure ACL

- Create Network scenario as shown in below figure.

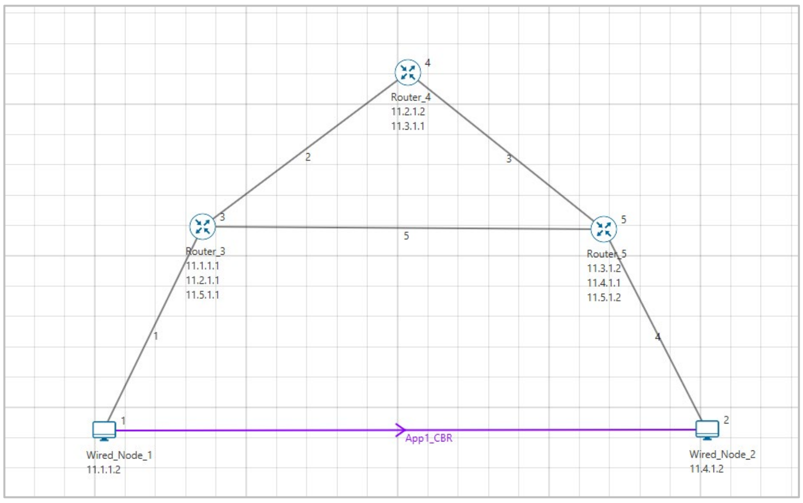


Figure 3-46: Network Scenario

- To create a new rule in the ACL use command as shown below to block UDP packet in Interface_3 of the Router_3.
- Click on the Application icon present in the top ribbon/toolbar.
  - CBR Application from Wired Node 1 to Wired Node 2 with 10 Mbps Generation Rate (Packet Size: 1460, Inter Arrival Time: 1168µs).
  - Set Transport Protocol to UDP.
  - Set Start Time as 30 Sec
- Click on run simulation option and In the Run time Interaction tab set Interactive Simulation as True and click on Accept.
- Set the Simulation Time as 200sec or more. Click Ok.
- Right click on Router_3 and select NetSim Console. Use the command as follows:

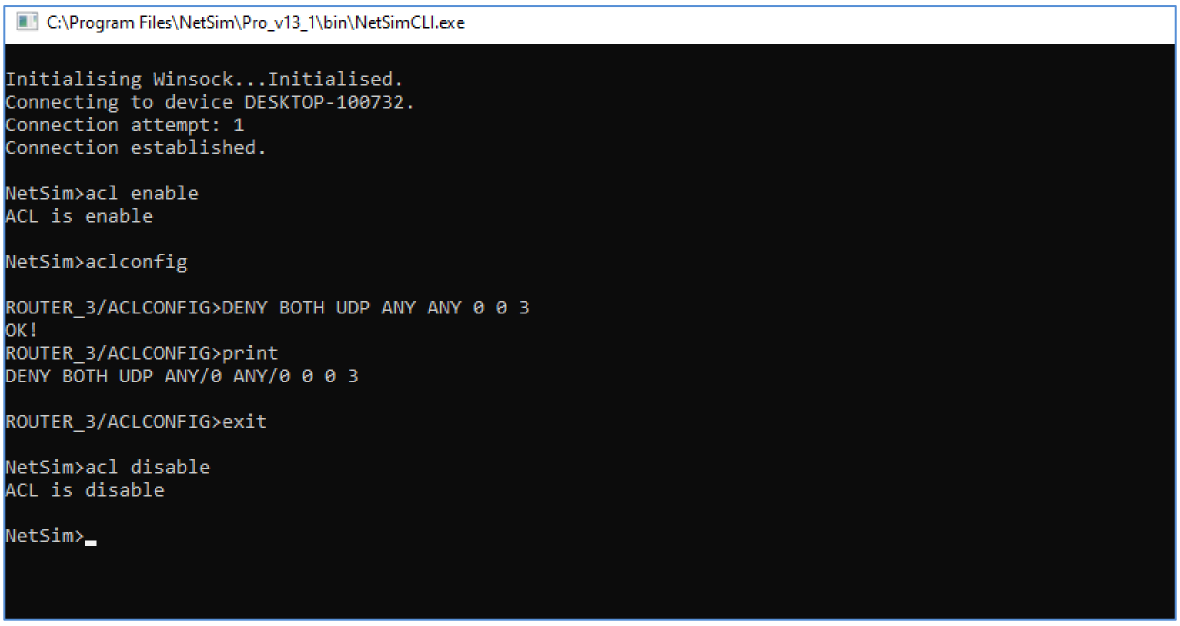
```
NetSim>acl enable
ACL is enable
NetSim>aclconfig
ROUTER_3/ACLCONFIG>acl print
Usage: [PERMIT, DENY] [INBOUND, OUTBOUND, BOTH] PROTO SRC DEST SPORT
DPORT IFID
ROUTER_3/ACLCONFIG>DENY BOTH UDP ANY ANY 0 0 3
OK!
ROUTER_3/ACLCONFIG>print
DENY BOTH UDP ANY/0 ANY/0 0 0 3
```

*ROUTER_3/ACLCONFIG>**exit***

*NetSim>**acl disable***

ACL is disable

NetSim>



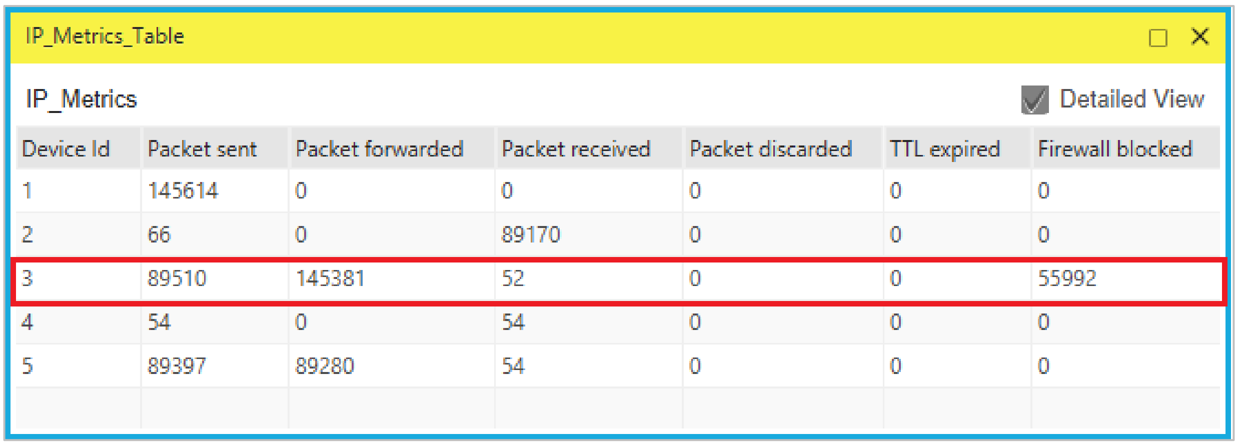
```
C:\Program Files\NetSim\Pro_v13_1\bin\NetSimCLI.exe

Initialising Winsock...Initialised.
Connecting to device DESKTOP-100732.
Connection attempt: 1
Connection established.

NetSim>acl enable
ACL is enable

NetSim>aclconfig

ROUTER_3/ACLCONFIG>DENY BOTH UDP ANY ANY 0 0 3
OK!
ROUTER_3/ACLCONFIG>print
DENY BOTH UDP ANY/0 ANY/0 0 0 3

ROUTER_3/ACLCONFIG>exit

NetSim>acl disable
ACL is disable

NetSim>_
```

Figure 3-47: ACL Configuration command

3.5.4.2 Results

The impact of ACL rule applied over the simulation traffic can be observed in the IP_Metrics_Table in the simulation results window, In Router_3 number of packets blocked by firewall has been shown below.

<u>Note: Results will vary based on time of ACL command are executed</u>



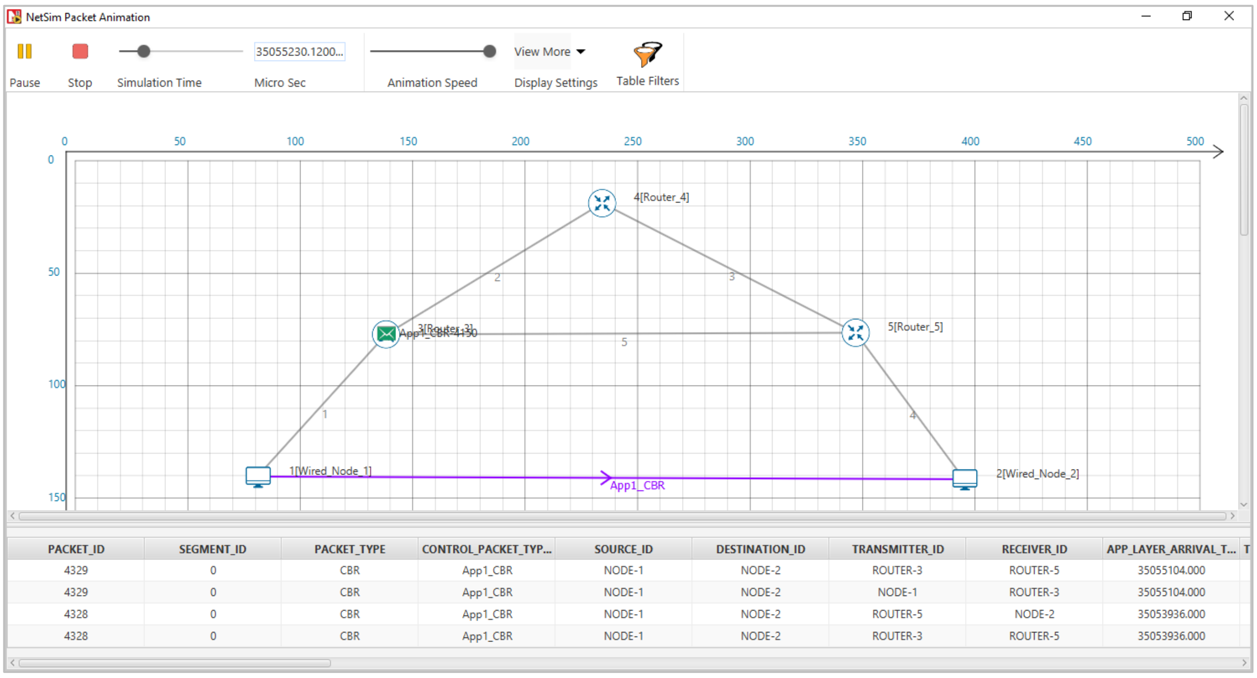
IP_Metrics_Table

IP_Metrics ✓ Detailed View

Device Id	Packet sent	Packet forwarded	Packet received	Packet discarded	TTL expired	Firewall blocked
1	145614	0	0	0	0	0
2	66	0	89170	0	0	0
3	89510	145381	52	0	0	55992
4	54	0	54	0	0	0
5	89397	89280	54	0	0	0

Figure 3-48: IP Metrics Table in result window

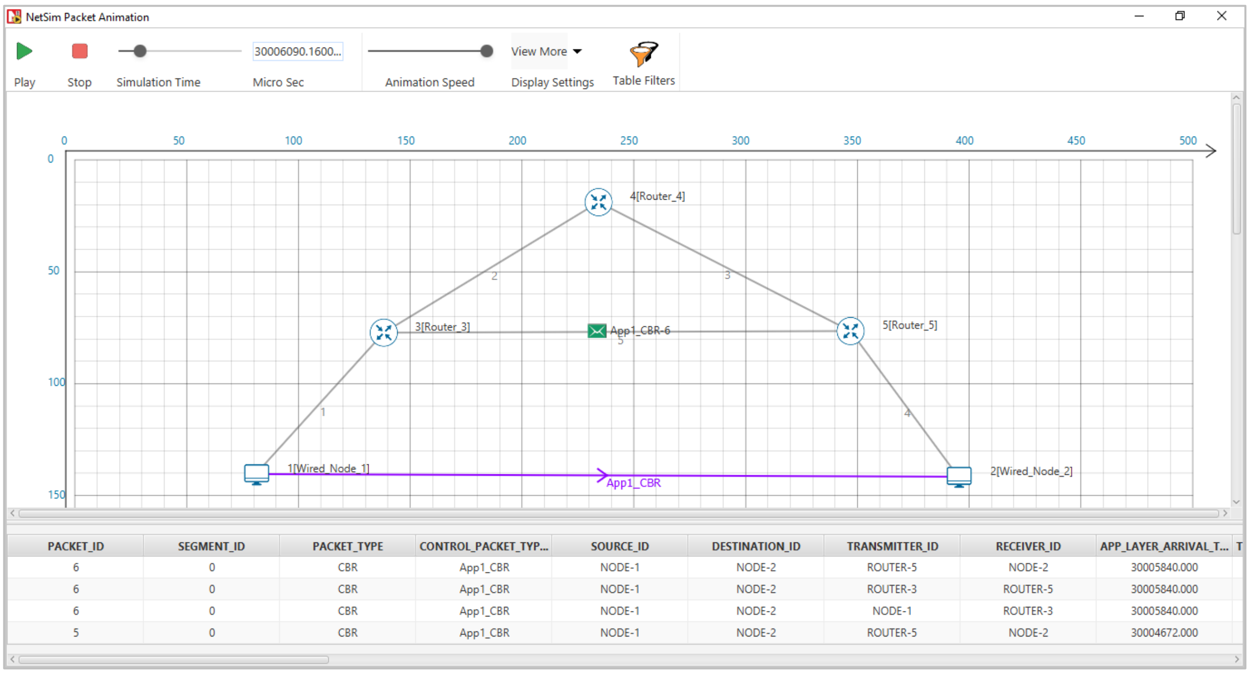
Check Packet animation window whether packets has been blocked in Router_3 for period when the ACL rule is applied. In the Below figure you can observe that packets are getting discarded at Router_3.



PACKET_ID	SEGMENT_ID	PACKET_TYPE	CONTROL_PACKET_TYP...	SOURCE_ID	DESTINATION_ID	TRANSMITTER_ID	RECEIVER_ID	APP_LAYER_ARRIVAL_T...	T
4329	0	CBR	App1_CBR	NODE-1	NODE-2	ROUTER-3	ROUTER-5	35055104.000	
4329	0	CBR	App1_CBR	NODE-1	NODE-2	NODE-1	ROUTER-3	35055104.000	
4328	0	CBR	App1_CBR	NODE-1	NODE-2	ROUTER-5	NODE-2	35053936.000	
4328	0	CBR	App1_CBR	NODE-1	NODE-2	ROUTER-3	ROUTER-5	35053936.000	

Figure 3-49: NetSim Animation Window when ACL rules are applied

Then packet transmission is allowed through same Interface_3 once after ACL is disabled.

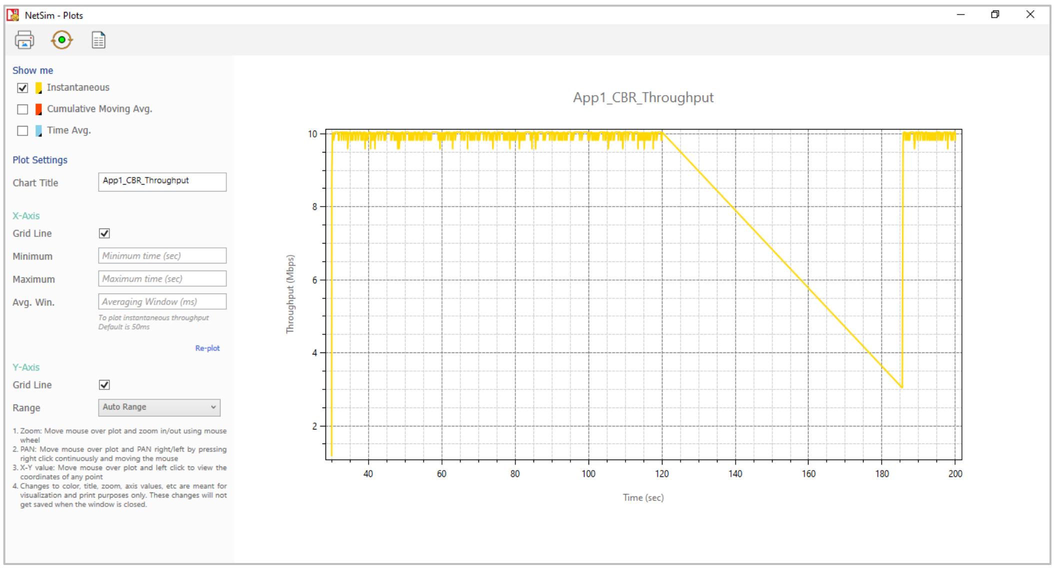


PACKET_ID	SEGMENT_ID	PACKET_TYPE	CONTROL_PACKET_TYP...	SOURCE_ID	DESTINATION_ID	TRANSMITTER_ID	RECEIVER_ID	APP_LAYER_ARRIVAL_T...	T
6	0	CBR	App1_CBR	NODE-1	NODE-2	ROUTER-5	NODE-2	30005840.000	
6	0	CBR	App1_CBR	NODE-1	NODE-2	ROUTER-3	ROUTER-5	30005840.000	
6	0	CBR	App1_CBR	NODE-1	NODE-2	NODE-1	ROUTER-3	30005840.000	
5	0	CBR	App1_CBR	NODE-1	NODE-2	ROUTER-5	NODE-2	30004672.000	

Figure 3-50: NetSim Animation Window After ACL Disabled

The impact of ACL rule applied over the simulation traffic can be observed in the Application throughput plot. Throughput graph will show a drop after ACL is set. If ACL is disabled after a while, application packets will start flowing across the router. The Application throughput plot will show a drop and increase(Moving througput graph) in throughput after setting ACL and disabling ACL respectively.

Example: ACL rule applied at around 120sec user can see the drop in throughput in the graph, since router blocks UDP packets in the plot. Once ACL has been disabled at around 185sec router permits packets and hence increase in throughput can be observed in the plot shown below Figure 3-51.



☑ Instantaneous
☐ Cumulative Moving Avg.
☐ Time Avg.

Chart Title: App1_CBR_Throughput

X-Axis
Grid Line ☑
Minimum: Minimum time (sec)
Maximum: Maximum time (sec)
Avg. Win.: Averaging Window (ms)
To plot instantaneous throughput Default is 50ms

Y-Axis
Grid Line ☑
Range: Auto Range

1. Zoom: Move mouse over plot and zoom in/out using mouse wheel
2. PAN: Move mouse over plot and PAN right/left by pressing right click continuously and moving the mouse
3. X-Y value: Move mouse over plot and left click to view the coordinates of any point
4. Changes to color, title, zoom, axis values, etc are meant for visualization and print purposes only. These changes will not get saved when the window is closed.

Figure 3-51: CBR Application throughput plot

3.5.5 Interactive Simulation using file

Interactive simulation using file allows users to pass commands as input through a text file. In the text file users can provide the commands along with the device in which it has to be executed by specifying the time stamps. This provides the user to have control over the scenario to execute commands at a specified time.

In the Interactive simulation text file, user should specify the exact time in seconds, along with the name of the device.

Format of the input text file for one device.

TIME=<*SIMULATION TIME IN SECONDS*>
DEVICE=<*DEVICE_NAME*>
<*COMMAND TO BE EXECUTED*>

The below Network scenario explains how to perform the Interactive simulation using file as input.

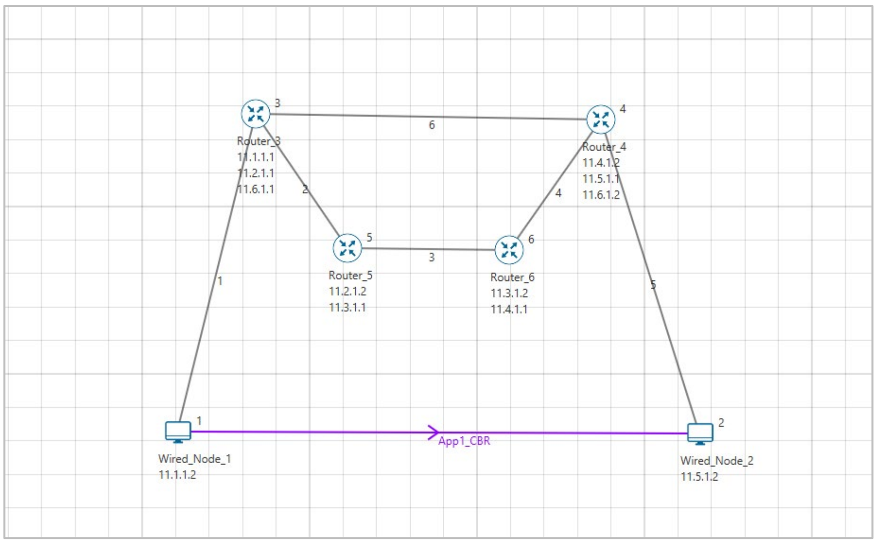
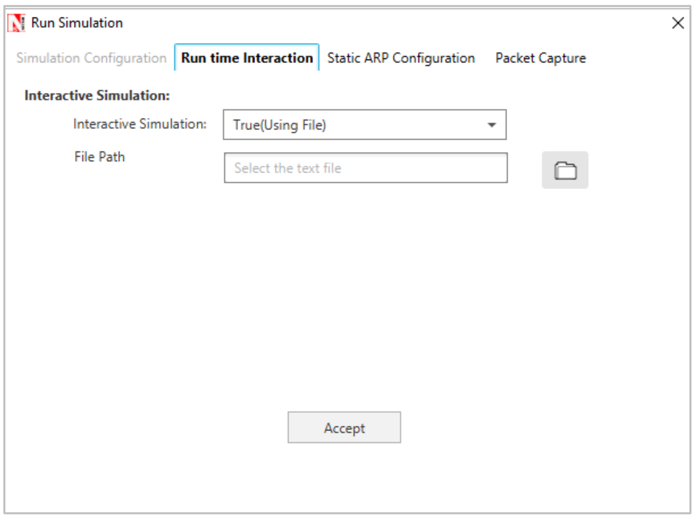


Figure 3-52: Network Topology

- The scenario comprises of 4 Routers, 2 Wired Node
- In Router Application layer, the routing protocol is set as RIP, since it is global property, routing protocol will be set as RIP in all Routers.
- Right click on the Application Flow App1 CBR and select Properties or click on the Application icon present in the top ribbon/toolbar.
- A CBR Application is generated from Wired Node 1 i.e., Source to Wired Node 2 i.e., Destination with Packet Size remaining 1460Bytes and Inter Arrival Time remaining 20000µs. Transport Protocol is set to UDP.
  - Additionally, Set start time as 1 sec.
- In order to set static routes to forward packets from Router 3 to the destination Wired Node via Router 5 instead of Router_4, such that the packets flow from Router 3=> Router 5=>Router 6=>Router4 =>Wired node 2 from a specified time say 25 seconds, the following input can be provided:

  ```
  TIME= 25
  DEVICE=Router_3
  route ADD 11.5.1.2 MASK 255.255.255.0 11.2.1.2 metric 1 if 2
  ```
- Create a text file with the above input and save it as input.txt file.
- Enable packet trace and Open the Run tab and switch to run time simulation tab select the Interactive simulation using file.



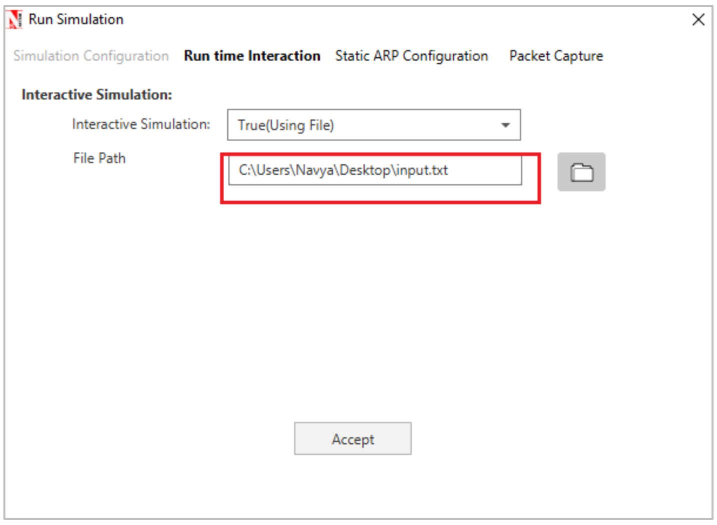
Interactive Simulation:

Interactive Simulation: True(Using File)

File Path: Select the text file

Figure 3-53: Run time Interaction tab with Interactive Simulation option set as True (Using File)

- Browse the Saved input.txt file in file path and select Accept button.



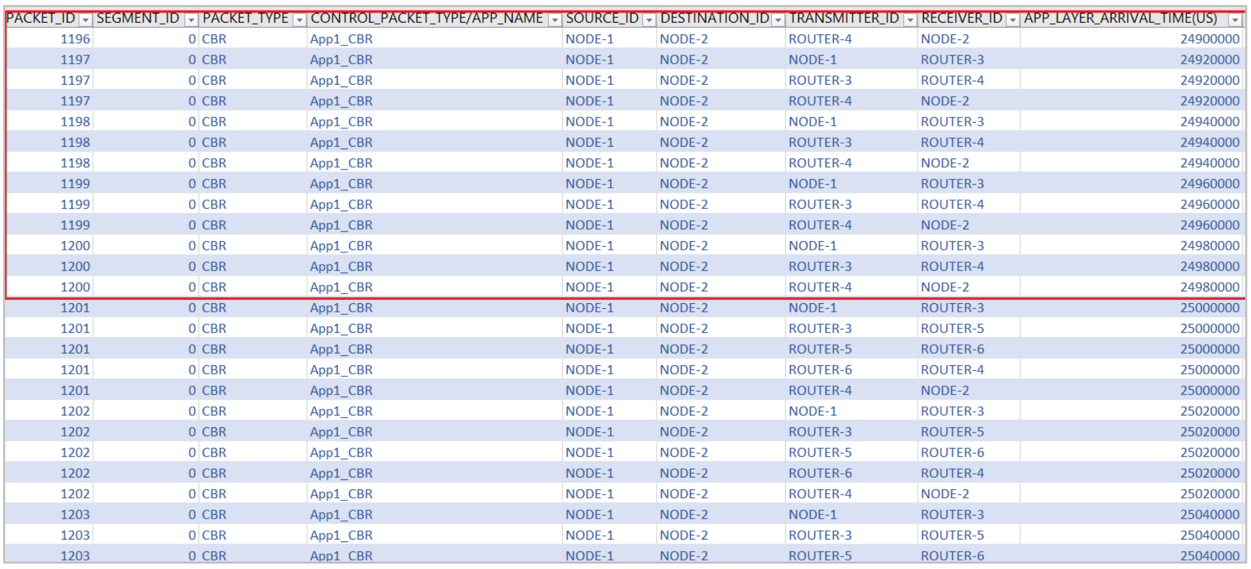
Interactive Simulation:

Interactive Simulation: True(Using File)

File Path: C:\Users\Navya\Desktop\input.txt

Figure 3-54: Run time Interaction tab with Interactive Simulation File Path set

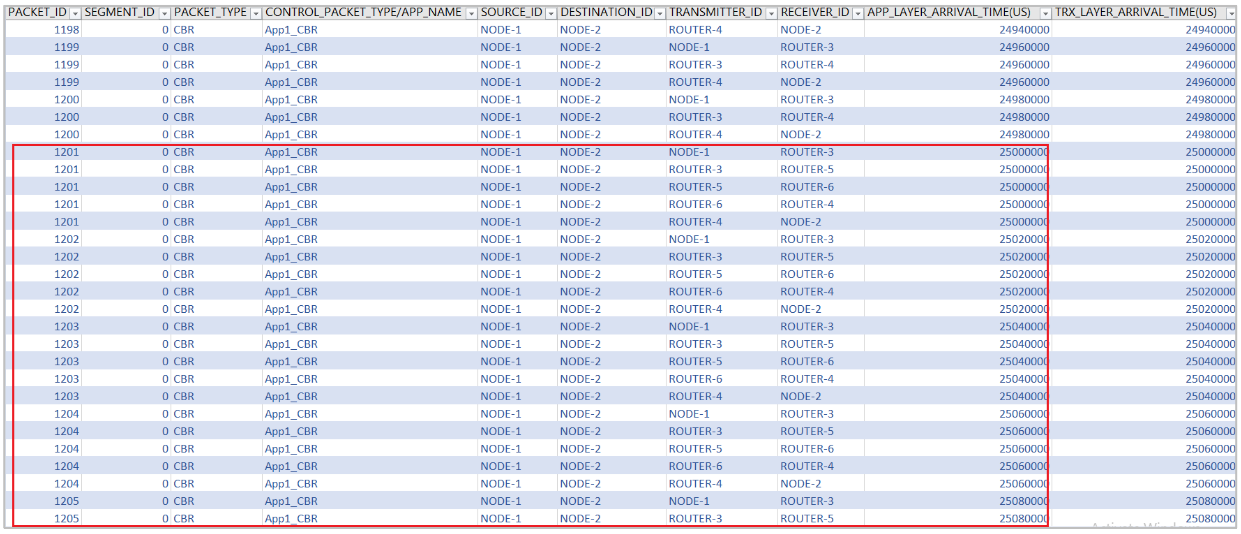
- Open the packet trace, you can observe that till 25 seconds the data packets are transmitted from Wired Node 1=>Router 3=> Router 4=> Wired Node 2.



PACKET_ID	SEGMENT_ID	PACKET_TYPE	CONTROL_PACKET_TYPE/APP_NAME	SOURCE_ID	DESTINATION_ID	TRANSMITTER_ID	RECEIVER_ID	APP_LAYER_ARRIVAL_TIME(US)
1196	0	CBR	App1_CBR	NODE-1	NODE-2	ROUTER-4	NODE-2	24900000
1197	0	CBR	App1_CBR	NODE-1	NODE-2	NODE-1	ROUTER-3	24920000
1197	0	CBR	App1_CBR	NODE-1	NODE-2	ROUTER-3	ROUTER-4	24920000
1197	0	CBR	App1_CBR	NODE-1	NODE-2	ROUTER-4	NODE-2	24920000
1198	0	CBR	App1_CBR	NODE-1	NODE-2	NODE-1	ROUTER-3	24940000
1198	0	CBR	App1_CBR	NODE-1	NODE-2	ROUTER-3	ROUTER-4	24940000
1198	0	CBR	App1_CBR	NODE-1	NODE-2	ROUTER-4	NODE-2	24940000
1199	0	CBR	App1_CBR	NODE-1	NODE-2	NODE-1	ROUTER-3	24960000
1199	0	CBR	App1_CBR	NODE-1	NODE-2	ROUTER-3	ROUTER-4	24960000
1199	0	CBR	App1_CBR	NODE-1	NODE-2	ROUTER-4	NODE-2	24960000
1200	0	CBR	App1_CBR	NODE-1	NODE-2	NODE-1	ROUTER-3	24980000
1200	0	CBR	App1_CBR	NODE-1	NODE-2	ROUTER-3	ROUTER-4	24980000
1200	0	CBR	App1_CBR	NODE-1	NODE-2	ROUTER-4	NODE-2	24980000
1201	0	CBR	App1_CBR	NODE-1	NODE-2	NODE-1	ROUTER-3	25000000
1201	0	CBR	App1_CBR	NODE-1	NODE-2	ROUTER-3	ROUTER-5	25000000
1201	0	CBR	App1_CBR	NODE-1	NODE-2	ROUTER-5	ROUTER-6	25000000
1201	0	CBR	App1_CBR	NODE-1	NODE-2	ROUTER-6	ROUTER-4	25000000
1201	0	CBR	App1_CBR	NODE-1	NODE-2	ROUTER-4	NODE-2	25000000
1202	0	CBR	App1_CBR	NODE-1	NODE-2	NODE-1	ROUTER-3	25020000
1202	0	CBR	App1_CBR	NODE-1	NODE-2	ROUTER-3	ROUTER-5	25020000
1202	0	CBR	App1_CBR	NODE-1	NODE-2	ROUTER-5	ROUTER-6	25020000
1202	0	CBR	App1_CBR	NODE-1	NODE-2	ROUTER-6	ROUTER-4	25020000
1202	0	CBR	App1_CBR	NODE-1	NODE-2	ROUTER-4	NODE-2	25020000
1203	0	CBR	App1_CBR	NODE-1	NODE-2	NODE-1	ROUTER-3	25040000
1203	0	CBR	App1_CBR	NODE-1	NODE-2	ROUTER-3	ROUTER-5	25040000
1203	0	CBR	App1_CBR	NODE-1	NODE-2	ROUTER-5	ROUTER-6	25040000

Figure 3-55: Data packets are transmitting from Wired Node 1=>Router 3=> Router 4=> Wired Node 2 in packet trace before 25th seconds

- After 25th second you can observe that, the routing happens according to the command specified in the input file.
- The data (APP 1 CBR) packets are transmitted from Wired Node 1=>Router 3=>Router 5=>Router 6=>Router 4=>Wired Node 2



PACKET_ID	SEGMENT_ID	PACKET_TYPE	CONTROL_PACKET_TYPE/APP_NAME	SOURCE_ID	DESTINATION_ID	TRANSMITTER_ID	RECEIVER_ID	APP_LAYER_ARRIVAL_TIME(US)	TRX_LAYER_ARRIVAL_TIME(US)
1198	0	CBR	App1_CBR	NODE-1	NODE-2	ROUTER-4	NODE-2	24940000	24940000
1199	0	CBR	App1_CBR	NODE-1	NODE-2	NODE-1	ROUTER-3	24960000	24960000
1199	0	CBR	App1_CBR	NODE-1	NODE-2	ROUTER-3	ROUTER-4	24960000	24960000
1199	0	CBR	App1_CBR	NODE-1	NODE-2	ROUTER-4	NODE-2	24960000	24960000
1200	0	CBR	App1_CBR	NODE-1	NODE-2	NODE-1	ROUTER-3	24980000	24980000
1200	0	CBR	App1_CBR	NODE-1	NODE-2	ROUTER-3	ROUTER-4	24980000	24980000
1200	0	CBR	App1_CBR	NODE-1	NODE-2	ROUTER-4	NODE-2	24980000	24980000
1201	0	CBR	App1_CBR	NODE-1	NODE-2	NODE-1	ROUTER-3	25000000	25000000
1201	0	CBR	App1_CBR	NODE-1	NODE-2	ROUTER-3	ROUTER-5	25000000	25000000
1201	0	CBR	App1_CBR	NODE-1	NODE-2	ROUTER-5	ROUTER-6	25000000	25000000
1201	0	CBR	App1_CBR	NODE-1	NODE-2	ROUTER-6	ROUTER-4	25000000	25000000
1201	0	CBR	App1_CBR	NODE-1	NODE-2	ROUTER-4	NODE-2	25000000	25000000
1202	0	CBR	App1_CBR	NODE-1	NODE-2	NODE-1	ROUTER-3	25020000	25020000
1202	0	CBR	App1_CBR	NODE-1	NODE-2	ROUTER-3	ROUTER-5	25020000	25020000
1202	0	CBR	App1_CBR	NODE-1	NODE-2	ROUTER-5	ROUTER-6	25020000	25020000
1202	0	CBR	App1_CBR	NODE-1	NODE-2	ROUTER-6	ROUTER-4	25020000	25020000
1202	0	CBR	App1_CBR	NODE-1	NODE-2	ROUTER-4	NODE-2	25020000	25020000
1203	0	CBR	App1_CBR	NODE-1	NODE-2	NODE-1	ROUTER-3	25040000	25040000
1203	0	CBR	App1_CBR	NODE-1	NODE-2	ROUTER-3	ROUTER-5	25040000	25040000
1203	0	CBR	App1_CBR	NODE-1	NODE-2	ROUTER-5	ROUTER-6	25040000	25040000
1203	0	CBR	App1_CBR	NODE-1	NODE-2	ROUTER-6	ROUTER-4	25040000	25040000
1203	0	CBR	App1_CBR	NODE-1	NODE-2	ROUTER-4	NODE-2	25040000	25040000
1204	0	CBR	App1_CBR	NODE-1	NODE-2	NODE-1	ROUTER-3	25060000	25060000
1204	0	CBR	App1_CBR	NODE-1	NODE-2	ROUTER-3	ROUTER-5	25060000	25060000
1204	0	CBR	App1_CBR	NODE-1	NODE-2	ROUTER-5	ROUTER-6	25060000	25060000
1204	0	CBR	App1_CBR	NODE-1	NODE-2	ROUTER-6	ROUTER-4	25060000	25060000
1204	0	CBR	App1_CBR	NODE-1	NODE-2	ROUTER-4	NODE-2	25060000	25060000
1205	0	CBR	App1_CBR	NODE-1	NODE-2	NODE-1	ROUTER-3	25080000	25080000
1205	0	CBR	App1_CBR	NODE-1	NODE-2	ROUTER-3	ROUTER-5	25080000	25080000

Figure 3-56: After 25th sec routing is initiating according to the command mentioned in input file

- The same can be observed in Animation window.

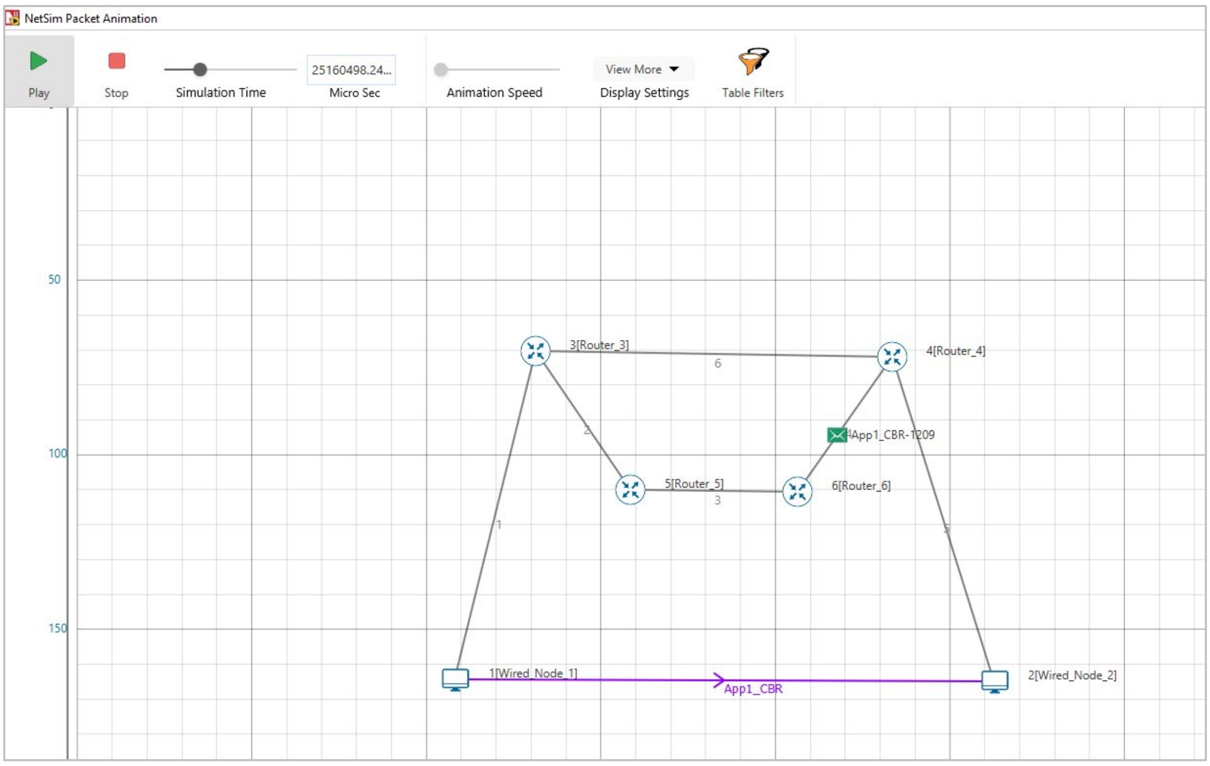


Figure 3-57: NetSim Animation window

- From the above example we can see that until the specified time mentioned in the input text file, the network considers the path formed by the routing protocol i.e RIP. From 25th Second onwards packets are routed according to the modified path specified in interactive simulation input text file.

4 Workspaces and Experiments

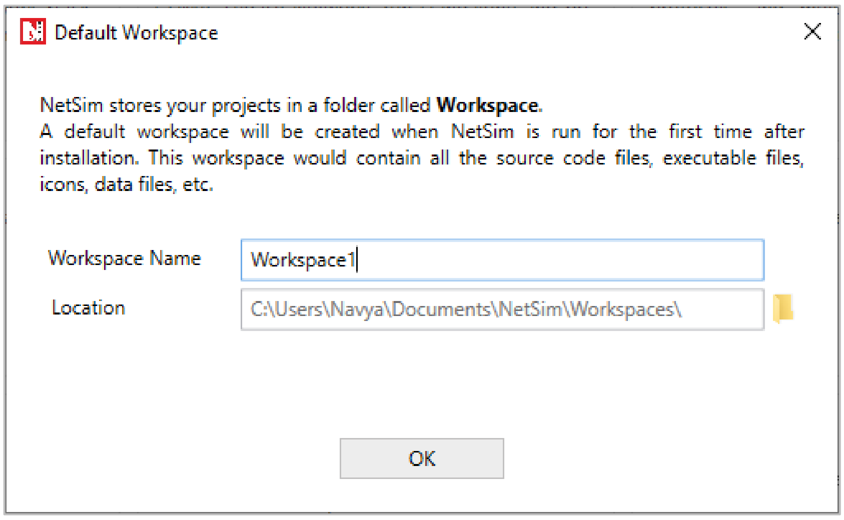
4.1 What is an Experiment and workspace in NetSim?

After users design and simulate a network in NetSim it can be saved as an *experiment*. This experiment is saved in a *Workspace*. A workspace also contains the source codes, executable files, icons, data files, etc. A workspace can contain one or more experiments. While NetSim supports multiple workspaces, users generally work in the *default* workspace. The default workspace of NetSim will have the master source code and the master binaries (compiled library, executable and DLL files).

New workspaces need to be created when:

- The user wants to modify the underlying source code of NetSim.
- A user chooses to organize a large number of saved experiments. The experiments can be saved in a different workspace.
- NetSim running on one PC/VM is time-shared between multiple users. Each user has his/her own workspace.

As mentioned earlier, NetSim stores your experiments (projects) in a folder termed as a Workspace. Default workspace is created in a user selected directory when NetSim is run for the first time after installation. Choose the path and enter the workspace name where you want the default workspace to be created.


Default Workspace

NetSim stores your projects in a folder called **Workspace**.
A default workspace will be created when NetSim is run for the first time after installation. This workspace would contain all the source code files, executable files, icons, data files, etc.

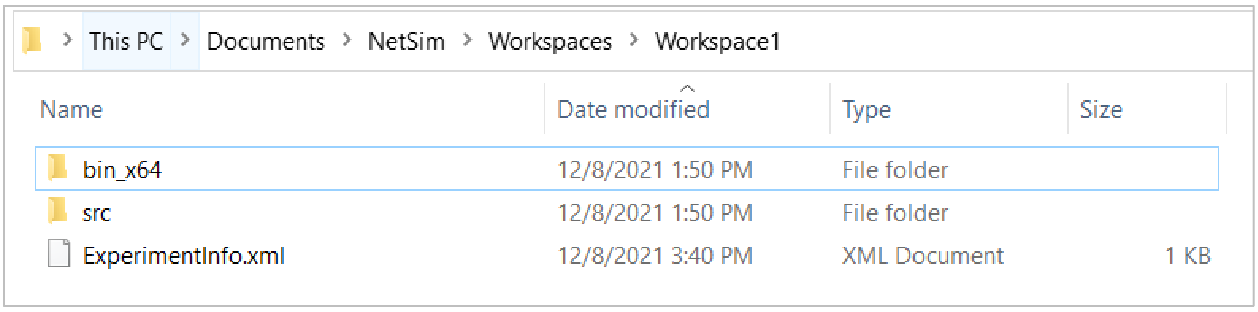
Workspace Name: Workspace1

Location: C:\Users\Navya\Documents\NetSim\Workspaces\

Figure 4-1: Default workspace created in Documents folder

This default workspace contains the following folders:

1. \src - contains protocol source codes.
2. \bin_x64 contains NetSim binaries for 64-bit.

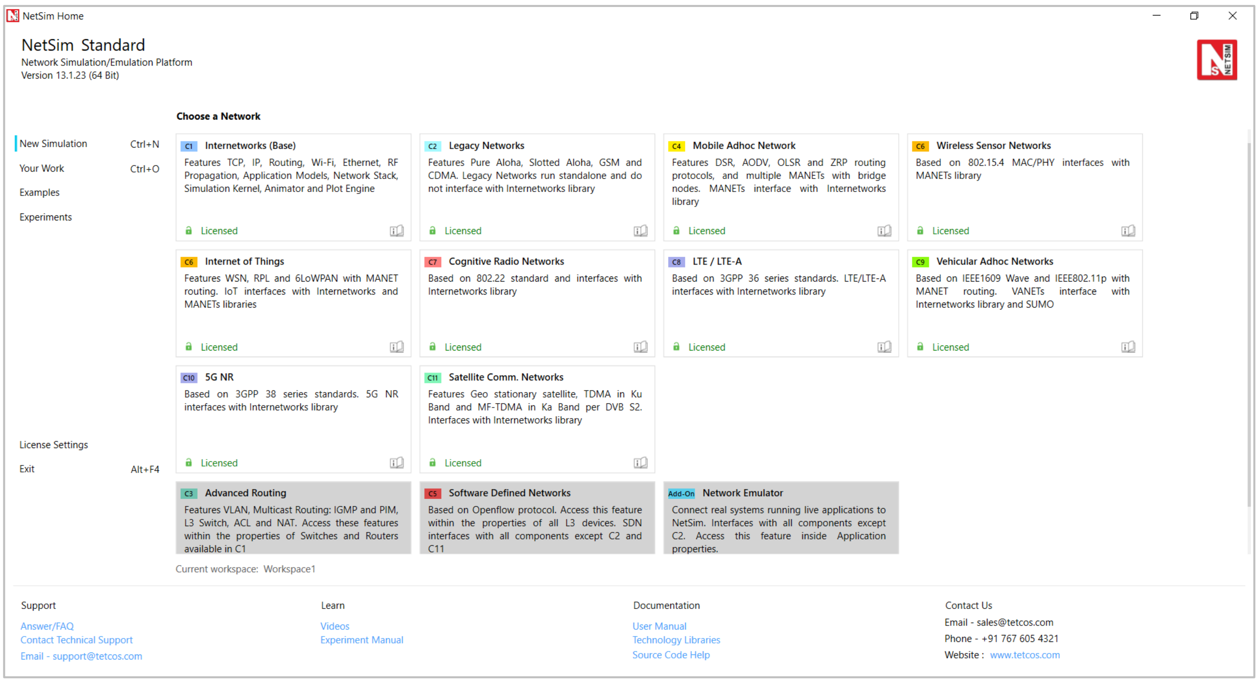


Name	Date modified	Type	Size
bin_x64	12/8/2021 1:50 PM	File folder	
src	12/8/2021 1:50 PM	File folder	
ExperimentInfo.xml	12/8/2021 3:40 PM	XML Document	1 KB

Figure 4-2: Default workspace containing the different folders

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

To create an experiment, select New Simulation - <Any Network> in the NetSim Home Screen as shown Figure 4-3.



NetSim Standard
Network Simulation/Emulation Platform
Version 13.1.23 (64 Bit)

Choose a Network

C1 Internetworks (Base)
Features TCP, IP, Routing, Wi-Fi, Ethernet, RF Propagation, Application Models, Network Stack, Simulation Kernel, Animator and Plot Engine
Licensed

C2 Legacy Networks
Features Pure Aloha, Slotted Aloha, GSM and CDMA. Legacy Networks run standalone and do not interface with Internetworks library
Licensed

C4 Mobile Adhoc Network
Features DSR, AODV, OLSR and ZRP routing protocols, and multiple MANETs with bridge nodes. MANETs interface with Internetworks library
Licensed

C6 Wireless Sensor Networks
Based on 802.15.4 MAC/PHY interfaces with MANETs library
Licensed

C6 Internet of Things
Features WSN, RPL and 6LoWPAN with MANET routing. IoT interfaces with Internetworks and MANETs libraries
Licensed

C7 Cognitive Radio Networks
Based on 802.22 standard and interfaces with Internetworks library
Licensed

C8 LTE / LTE-A
Based on 3GPP 36 series standards. LTE/LTE-A interfaces with Internetworks library
Licensed

C9 Vehicular Adhoc Networks
Based on IEEE1609 Wave and IEEE802.11p with MANET routing. VANETs interface with Internetworks library and SUMO
Licensed

C10 5G NR
Based on 3GPP 38 series standards. 5G NR interfaces with Internetworks library
Licensed

C11 Satellite Comm. Networks
Features Geo stationary satellite, TDMA in Ku Band and MF-TDMA in Ka Band per DVB S2. Interfaces with Internetworks library
Licensed

C3 Advanced Routing
Features VLAN, Multicast Routing: IGMP and PIM, L3 Switch, ACL and NAT. Access these features within the properties of Switches and Routers available in C1

C5 Software Defined Networks
Based on Openflow protocol. Access this feature within the properties of all L3 devices. SDN interfaces with all components except C2 and C11

Add-On Network Emulator
Connect real systems running live applications to NetSim. Interfaces with all components except C2. Access this feature inside Application properties.

Current workspace: Workspace1

Support
Answer/FAQ
Contact Technical Support
Email - support@tetcos.com

Learn
Videos
Experiment Manual

Documentation
User Manual
Technology Libraries
Source Code Help

Contact Us
Email - sales@tetcos.com
Phone - +91 767 605 4321
Website : www.tetcos.com

Figure 4-3: NetSim Home Screen

The created experiment can be saved by clicking on File > Save button on the top left corner of the design window.

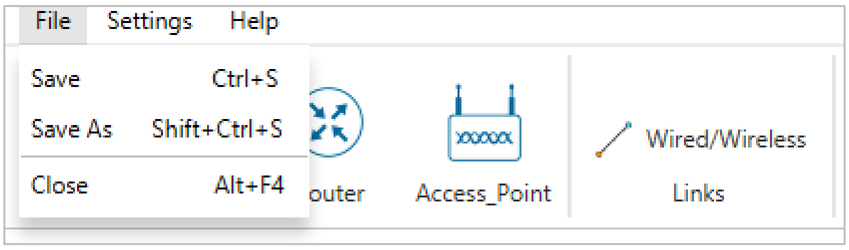
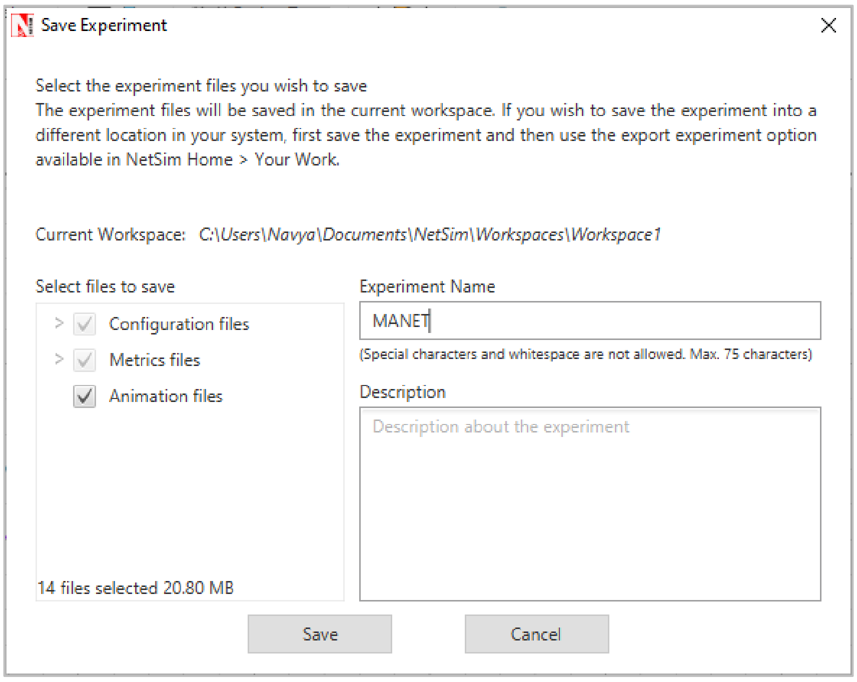


Figure 4-4: Save/Save As an experiment by clicking on File

A save pop-up window appears which asks the user to input an Experiment Name and Description for the experiment. The workspace path is also shown in the window.



Save Experiment

Select the experiment files you wish to save
The experiment files will be saved in the current workspace. If you wish to save the experiment into a different location in your system, first save the experiment and then use the export experiment option available in NetSim Home > Your Work.

Current Workspace: C:\Users\Navya\Documents\NetSim\Workspaces\Workspace1

Select files to save

- [✓] Configuration files
- [✓] Metrics files
- [✓] Animation files

14 files selected 20.80 MB

Experiment Name: MANET

(Special characters and whitespace are not allowed. Max. 75 characters)

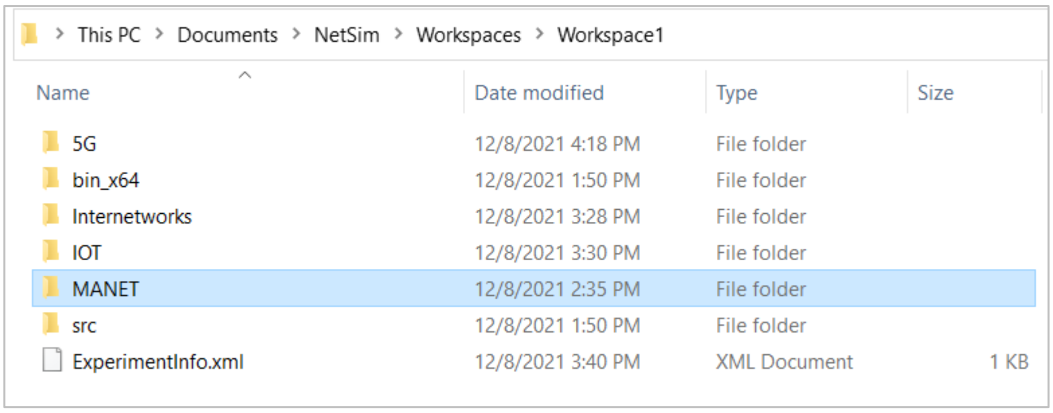
Description: Description about the experiment

Figure 4-5: Save popup window

User needs to input the Experiment Name (Description is optional) and then click on Save. The workspace path is non-editable. The experiment will be saved in the current workspace directory. Users can also select the files which are to be saved into the experiment folder.

- The Configuration file will be mandatorily saved into the experiment folder.
- Optional: Simulation output files such as Metrics.xml, Animation files, Event Trace file, Packet Trace file and Plot data (if enabled).
- Optional: Protocol logs (if written) or Custom Log files (if codes have been modified for logging)

In our example, we saved the experiment with the name MANET and this experiment can be found in the default workspace path as shown below Figure 4-6.

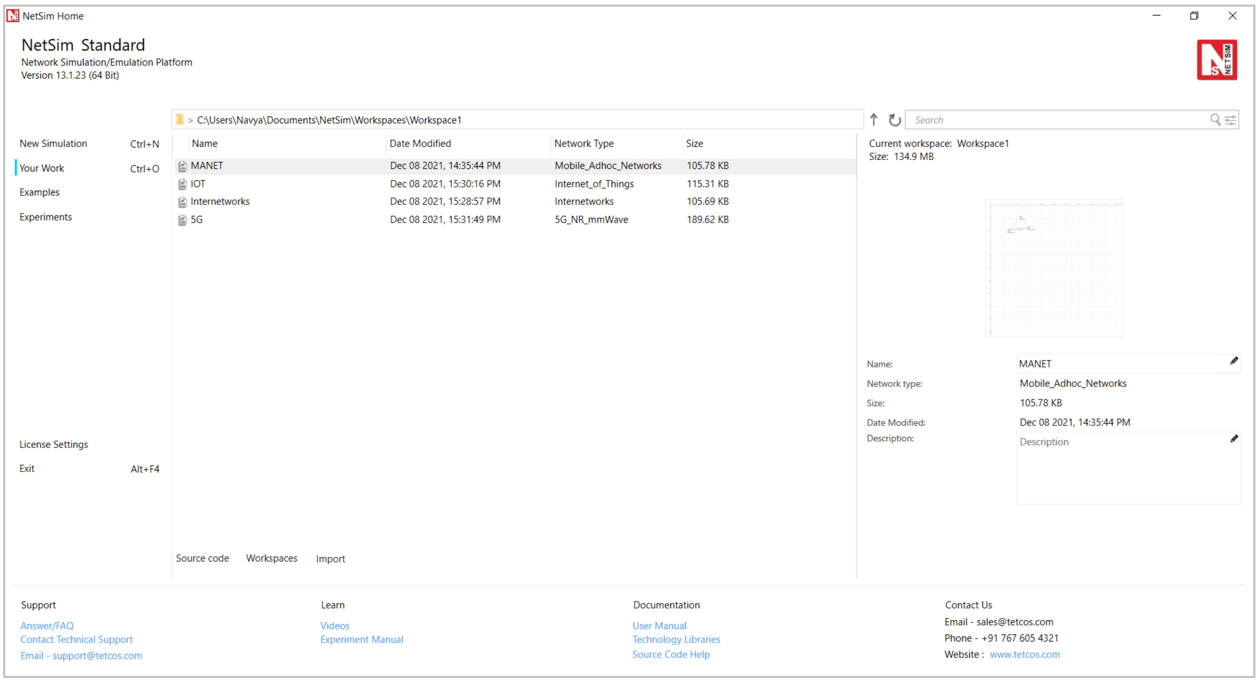


This PC > Documents > NetSim > Workspaces > Workspace1

Name	Date modified	Type	Size
5G	12/8/2021 4:18 PM	File folder	
bin_x64	12/8/2021 1:50 PM	File folder	
Internetworks	12/8/2021 3:28 PM	File folder	
IOT	12/8/2021 3:30 PM	File folder	
MANET	12/8/2021 2:35 PM	File folder	
src	12/8/2021 1:50 PM	File folder	
ExperimentInfo.xml	12/8/2021 3:40 PM	XML Document	1 KB

Figure 4-6: Manet Example Saved in Workspace

Users can also see the saved experiments in Your Work menu as shown below Figure 4-7.



NetSim Home

NetSim Standard
Network Simulation/Emulation Platform
Version 13.1.23 (64 Bit)

> C:\Users\Navya\Documents\NetSim\Workspaces\Workspace1

Search

New Simulation	Ctrl+N
Your Work	Ctrl+O
Examples	
Experiments	
License Settings	
Exit	Alt+F4

Name	Date Modified	Network Type	Size
MANET	Dec 08 2021, 14:35:44 PM	Mobile_Adhoc_Networks	105.78 KB
IOT	Dec 08 2021, 15:30:16 PM	Internet_of_Things	115.31 KB
Internetworks	Dec 08 2021, 15:28:57 PM	Internetworks	105.69 KB
5G	Dec 08 2021, 15:31:49 PM	5G_NR_mmWave	189.62 KB

Current workspace: Workspace1
Size: 134.9 MB

Name:	MANET
Network type:	Mobile_Adhoc_Networks
Size:	105.78 KB
Date Modified:	Dec 08 2021, 14:35:44 PM
Description:	Description

Source code Workspaces Import

Support
Answer/FAQ
Contact Technical Support
Email - support@tetcos.com

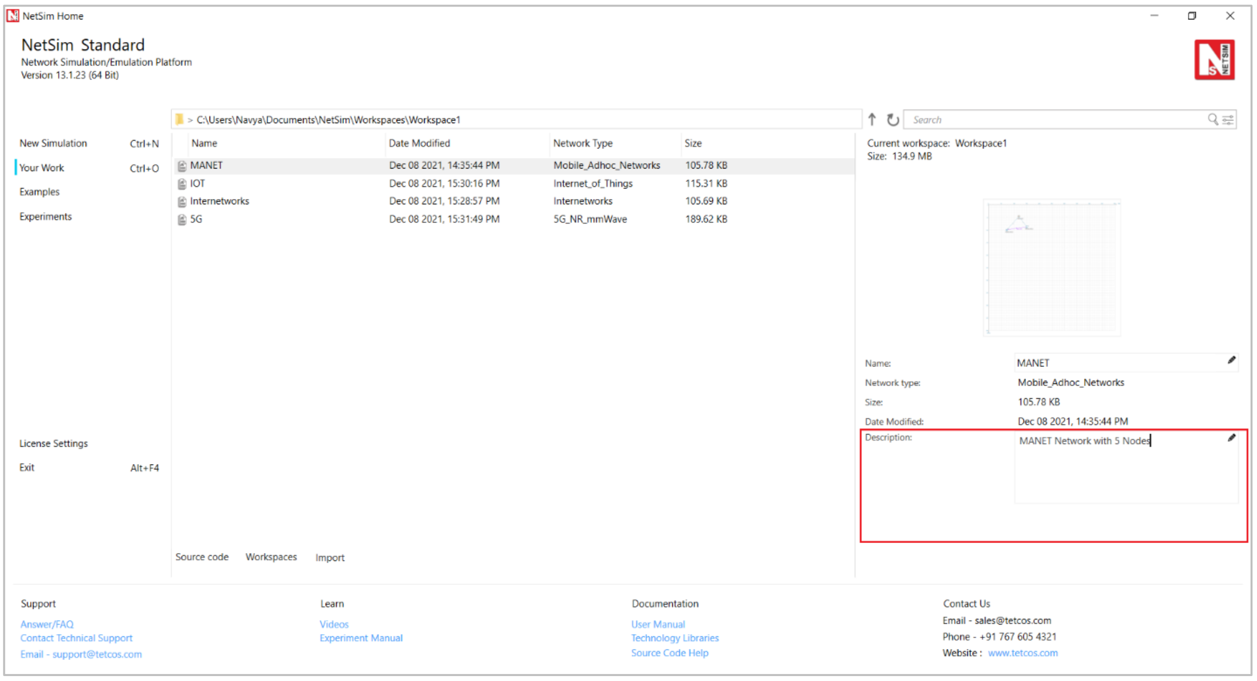
Learn
Videos
Experiment Manual

Documentation
User Manual
Technology Libraries
Source Code Help

Contact Us
Email - sales@tetcos.com
Phone - +91 767 605 4321
Website : www.tetcos.com

Figure 4-7: Saved experiments in Your work menu

If a Description was provided while saving the experiment, it will be displayed on Description panel on the right. Users can also edit the description for an experiment in the description panel.



NetSim Home

NetSim Standard
Network Simulation/Emulation Platform
Version 13.1.23 (64 Bit)

> C:\Users\Navya\Documents\NetSim\Workspaces\Workspace1

Search

New Simulation	Ctrl+N
Your Work	Ctrl+O
Examples	
Experiments	
License Settings	
Exit	Alt+F4

Name	Date Modified	Network Type	Size
MANET	Dec 08 2021, 14:35:44 PM	Mobile_Adhoc_Networks	105.78 KB
IOT	Dec 08 2021, 15:30:16 PM	Internet_of_Things	115.31 KB
Internetworks	Dec 08 2021, 15:28:57 PM	Internetworks	105.69 KB
5G	Dec 08 2021, 15:31:49 PM	5G_NR_mmWave	189.62 KB

Current workspace: Workspace1
Size: 134.9 MB

Name:	MANET
Network type:	Mobile_Adhoc_Networks
Size:	105.78 KB
Date Modified:	Dec 08 2021, 14:35:44 PM
Description:	MANET Network with 5 Nodes

Source code Workspaces Import

Support
Answer/FAQ
Contact Technical Support
Email - support@tetcos.com

Learn
Videos
Experiment Manual

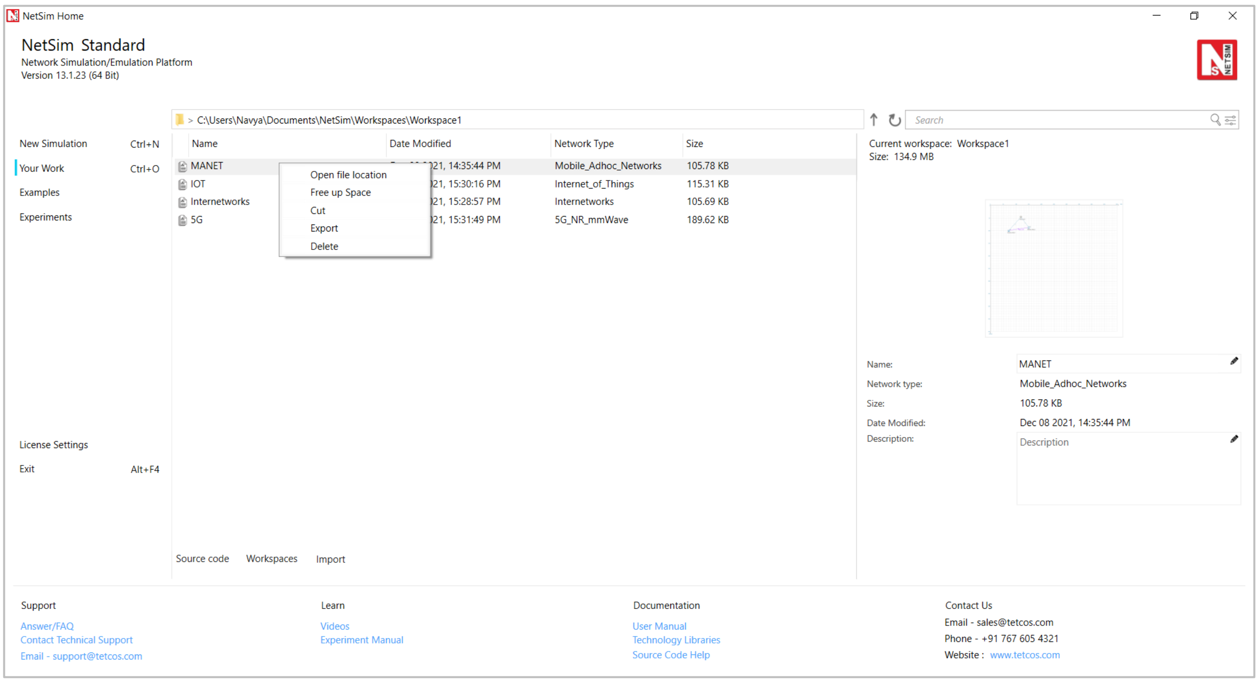
Documentation
User Manual
Technology Libraries
Source Code Help

Contact Us
Email - sales@tetcos.com
Phone - +91 767 605 4321
Website : www.tetcos.com

Figure 4-8: Description will be displayed on the right side of the description panel

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

Users can **Open file location** where the experiment is saved, **delete** the experiment, **Export** the experiment or **Cut** (and paste inside a different folder), **Free up Space** is used to delete the files that may not be important there by reducing the folder size, an experiment by right-clicking on the experiment in the **Your Work** window as shown below in Figure 4-9.



C:\Users\Navya\Documents\NetSim\Workspaces\Workspace1

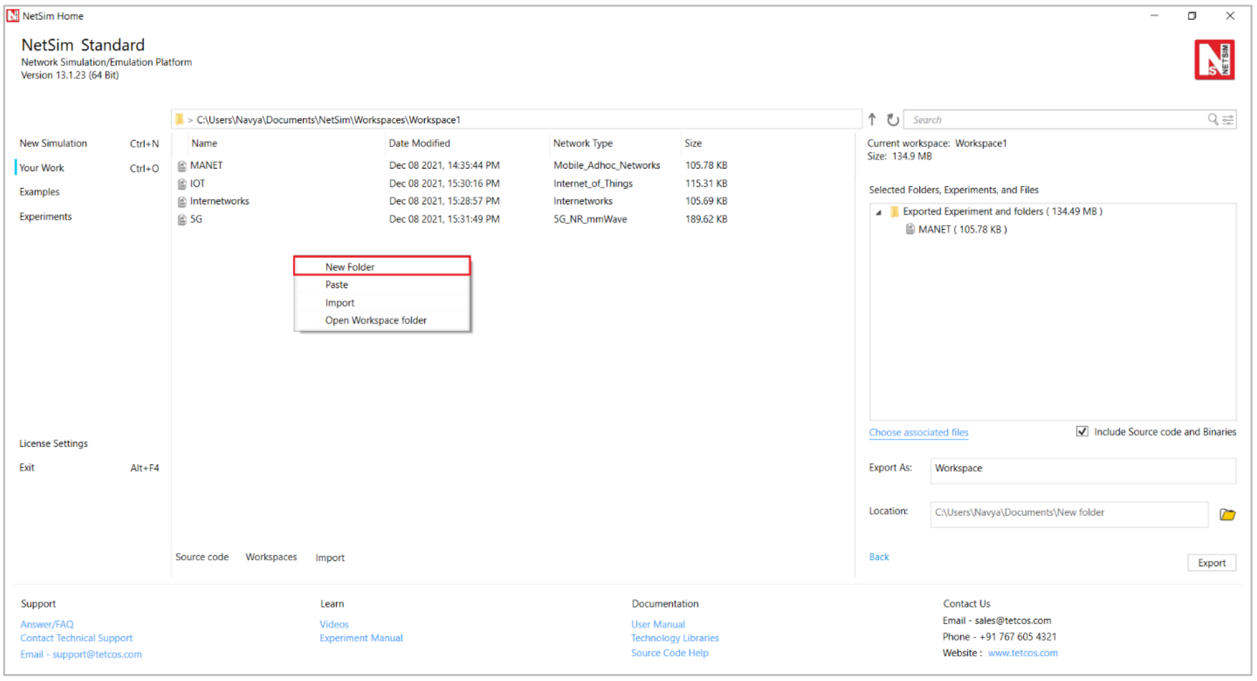
Name	Date Modified	Network Type	Size
MANET	...21, 14:35:44 PM	Mobile_Adhoc_Networks	105.78 KB
IOT	...21, 15:30:16 PM	Internet_of_Things	115.31 KB
Internetworks	...21, 15:28:57 PM	Internetworks	105.69 KB
5G	...21, 15:31:49 PM	5G_NR_mmWave	189.62 KB

Current workspace: Workspace1
Size: 134.9 MB

Name:	MANET
Network type:	Mobile_Adhoc_Networks
Size:	105.78 KB
Date Modified:	Dec 08 2021, 14:35:44 PM
Description:	Description

Figure 4-9: Right click on experiment name to view different options like “Open file location, Cut, Delete, Export” and Free up Space etc.

If the user wants to move the experiment into a New folder, create a **New folder** by right clicking on the experiment panel (either in the white space below the experiment list or on the header) as shown and cut the experiment and paste it inside a new folder.



C:\Users\Navya\Documents\NetSim\Workspaces\Workspace1

Name	Date Modified	Network Type	Size
MANET	Dec 08 2021, 14:35:44 PM	Mobile_Adhoc_Networks	105.78 KB
IOT	Dec 08 2021, 15:30:16 PM	Internet_of_Things	115.31 KB
Internetworks	Dec 08 2021, 15:28:57 PM	Internetworks	105.69 KB
5G	Dec 08 2021, 15:31:49 PM	5G_NR_mmWave	189.62 KB

Current workspace: Workspace1
Size: 134.9 MB

Selected Folders, Experiments, and Files
Exported Experiment and folders (134.49 MB)
MANET (105.78 KB)

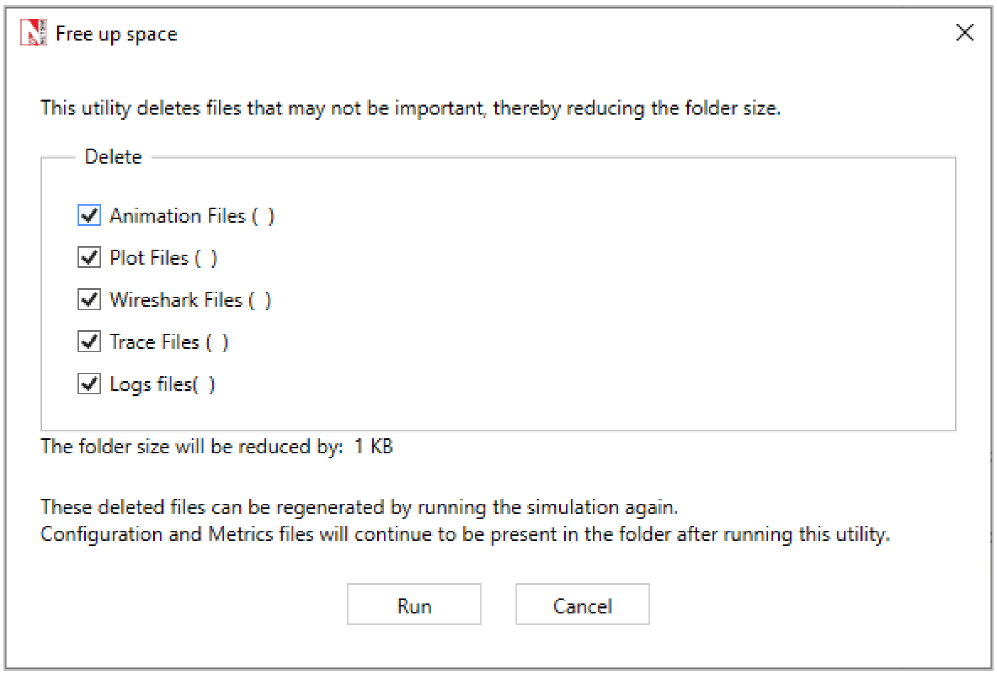
[✓] Include Source code and Binaries

Export As: Workspace

Location: C:\Users\Navya\Documents\New folder

Figure 4-10: Create a “New Folder”

Users can also **free up space** by deleting files which may not be important. Select the files and click on run. The deleted files can be regenerated by running the simulation again.



Free up space

This utility deletes files that may not be important, thereby reducing the folder size.

Delete

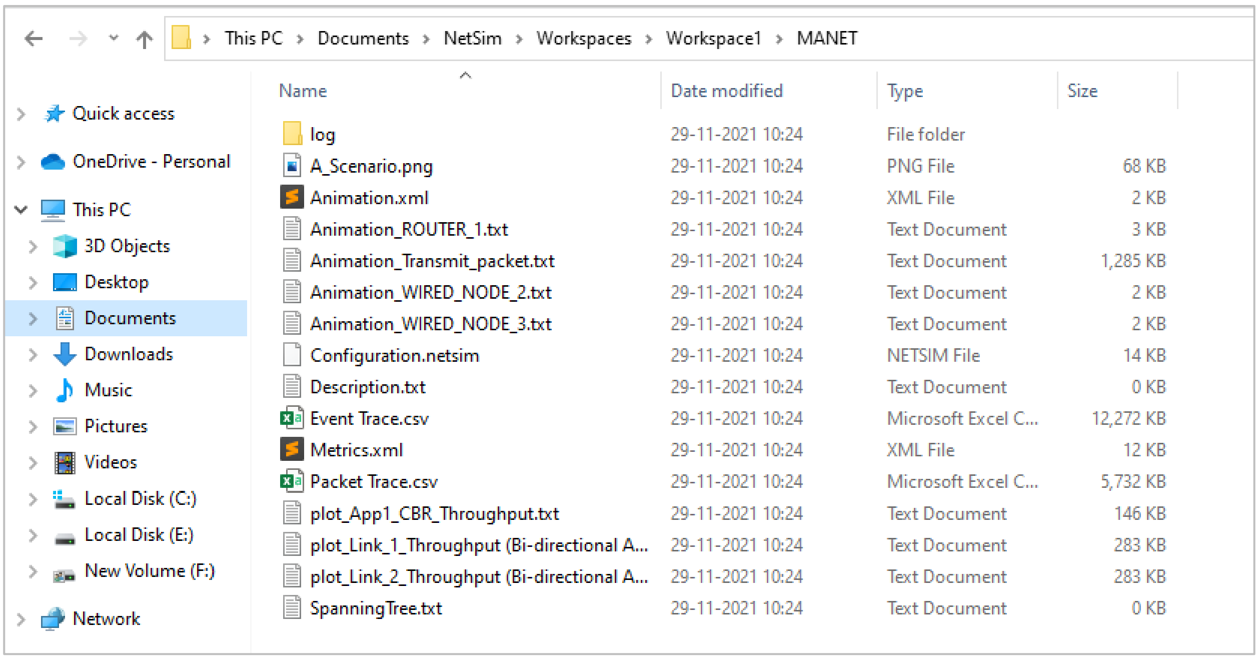
- [x] Animation Files ( )
- [x] Plot Files ( )
- [x] Wireshark Files ( )
- [x] Trace Files ( )
- [x] Logs files( )

The folder size will be reduced by: 1 KB

These deleted files can be regenerated by running the simulation again.
Configuration and Metrics files will continue to be present in the folder after running this utility.

Figure 4-11: Free up space window

In this example, we have saved all the files related to the experiment. You can see the various files stored in the experiment folder in Figure 4-12.



This PC › Documents › NetSim › Workspaces › Workspace1 › MANET

Name	Date modified	Type	Size
log	29-11-2021 10:24	File folder	
A_Scenario.png	29-11-2021 10:24	PNG File	68 KB
Animation.xml	29-11-2021 10:24	XML File	2 KB
Animation_ROUTER_1.txt	29-11-2021 10:24	Text Document	3 KB
Animation_Transmit_packet.txt	29-11-2021 10:24	Text Document	1,285 KB
Animation_WIRED_NODE_2.txt	29-11-2021 10:24	Text Document	2 KB
Animation_WIRED_NODE_3.txt	29-11-2021 10:24	Text Document	2 KB
Configuration.netsim	29-11-2021 10:24	NETSIM File	14 KB
Description.txt	29-11-2021 10:24	Text Document	0 KB
Event Trace.csv	29-11-2021 10:24	Microsoft Excel C...	12,272 KB
Metrics.xml	29-11-2021 10:24	XML File	12 KB
Packet Trace.csv	29-11-2021 10:24	Microsoft Excel C...	5,732 KB
plot_App1_CBR_Throughput.txt	29-11-2021 10:24	Text Document	146 KB
plot_Link_1_Throughput (Bi-directional A...	29-11-2021 10:24	Text Document	283 KB
plot_Link_2_Throughput (Bi-directional A...	29-11-2021 10:24	Text Document	283 KB
SpanningTree.txt	29-11-2021 10:24	Text Document	0 KB

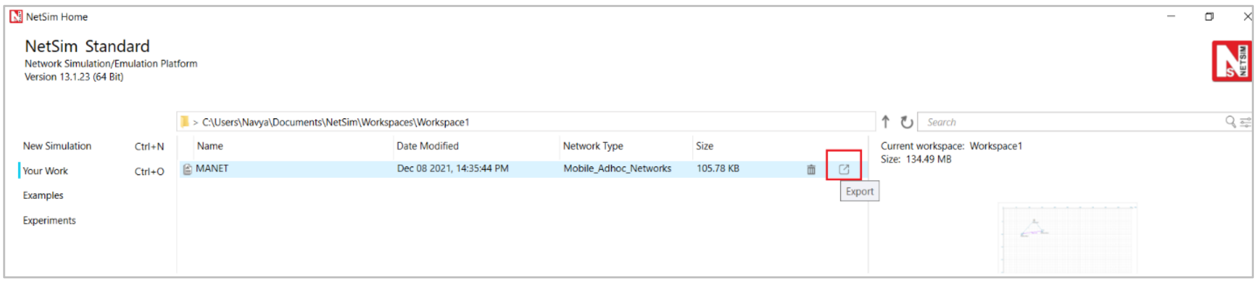
Figure 4-12: Simulation output files in experiment folder

4.3 Should each user have a workspace?

There is no strict association between users and workspaces. A single user can have multiple workspaces (and in turn experiments in each workspace), or multiple users can operate in one workspace.

4.4 How does a user export an experiment?

To save experiments in a different location, you have to first save the experiment in the current workspace and then use the export option present under **Your work** in the NetSim Home Screen as shown in Figure 4-13.



NetSim Home

NetSim Standard
Network Simulation/Emulation Platform
Version 13.1.23 (64 Bit)

C:\Users\Navya\Documents\NetSim\Workspaces\Workspace1

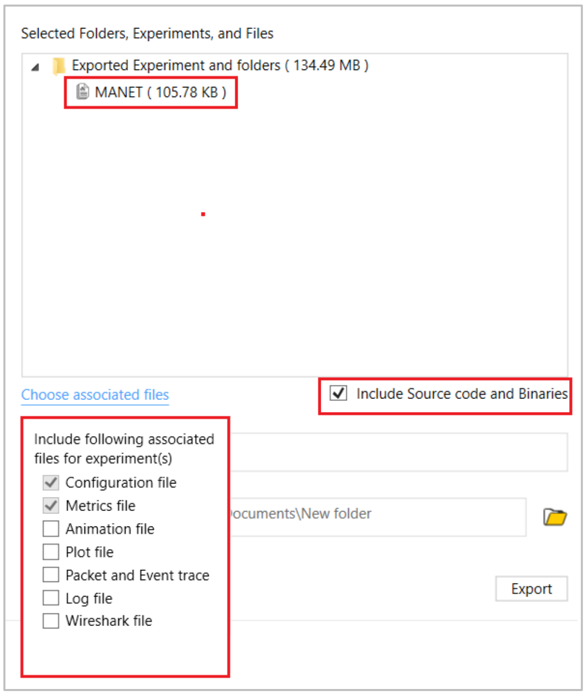
New Simulation	Ctrl+N			
Your Work	Ctrl+O			
Examples				
Experiments				

Name	Date Modified	Network Type	Size
MANET	Dec 08 2021, 14:35:44 PM	Mobile_Adhoc_Networks	105.78 KB

Current workspace: Workspace1
Size: 134.49 MB

Figure 4-13: Export option present in Your work in NetSim Home Screen

If you click on the Export option, an Export Experiment panel appears where you can select the files to be exported. You can also select the source code and binaries if required. While the Configuration file is mandatory, other files are optional.



Selected Folders, Experiments, and Files

Exported Experiment and folders (134.49 MB)
MANET (105.78 KB)

Choose associated files

[✓] Include Source code and Binaries

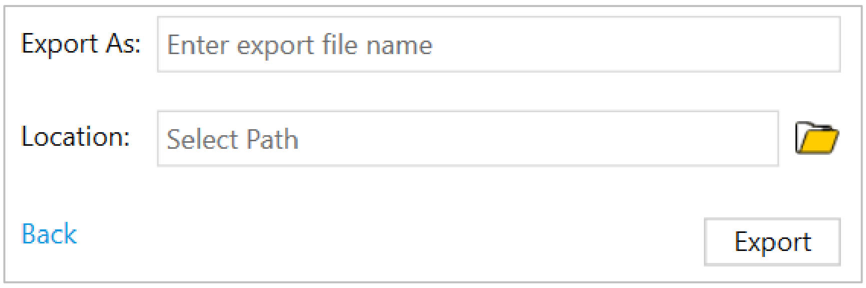
Include following associated files for experiment(s)

- [✓] Configuration file
- [✓] Metrics file
- [ ] Animation file
- [ ] Plot file
- [ ] Packet and Event trace
- [ ] Log file
- [ ] Wireshark file

ocuments\New folder

Figure 4-14: Export Experiment pop-up window

You need to give the destination path and name of the experiment while exporting. The exported experiments will be saved with a .netsim_exp extension.



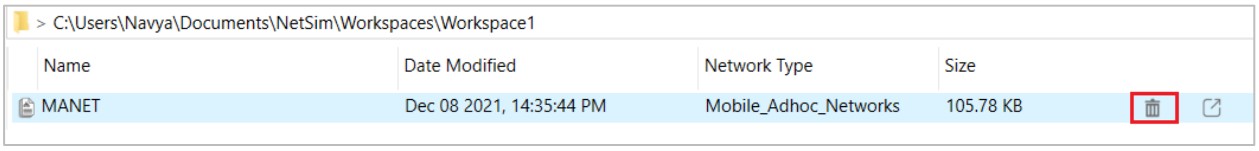
Export As: Enter export file name

Location: Select Path

Figure 4-15: Select Export file and Path in export window

4.5 How does a user delete an Experiment in a workspace?

Users can delete experiments by clicking on the delete icon as shown below in Figure 4-16.

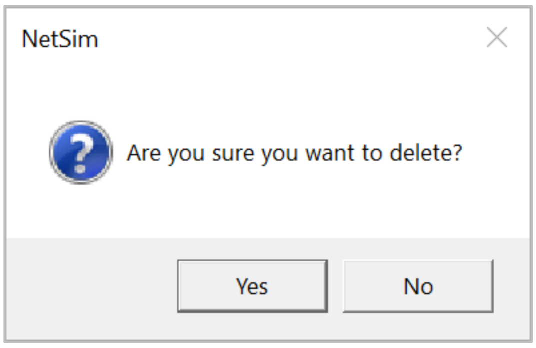


> C:\Users\Navya\Documents\NetSim\Workspaces\Workspace1

Name	Date Modified	Network Type	Size
MANET	Dec 08 2021, 14:35:44 PM	Mobile_Adhoc_Networks	105.78 KB

Figure 4-16: Delete an Experiment in a workspace

It displays a popup window as shown in Figure 4-17. Click on YES.

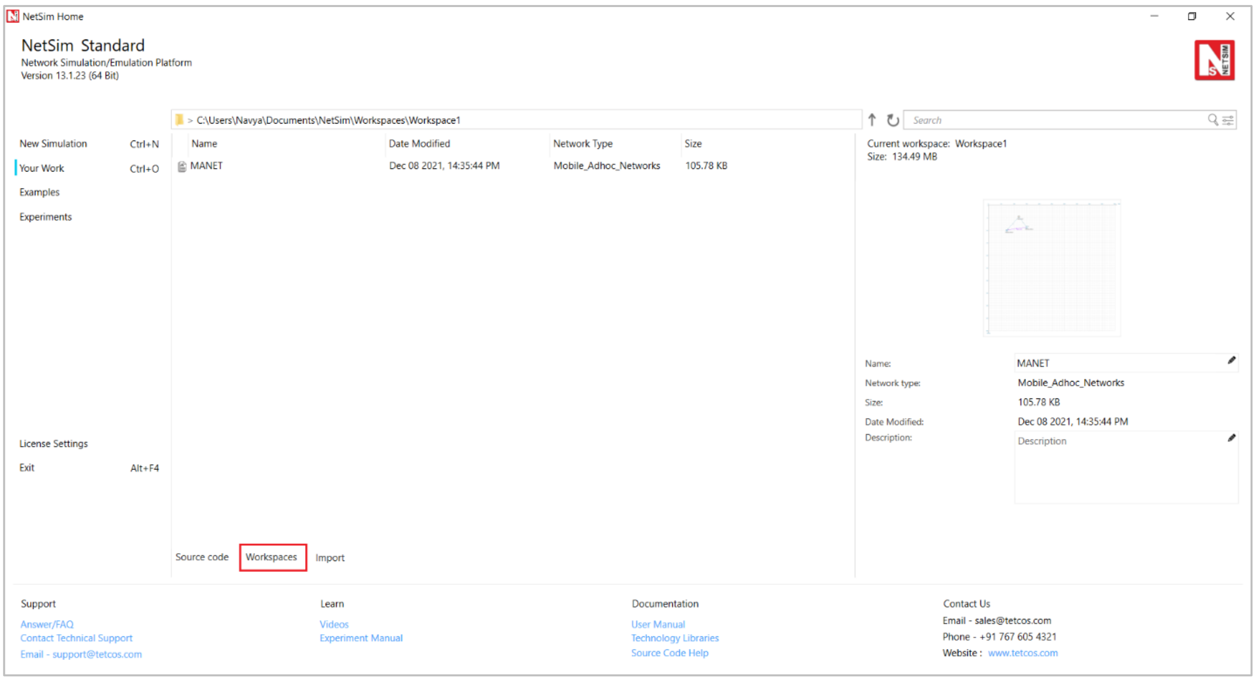


Are you sure you want to delete?

Figure 4-17: Delete experiment confirmation window popup

4.6 How does a user create a new workspace?

To create a new workspace, click on **Workspaces** present in Your work Menu shown as below



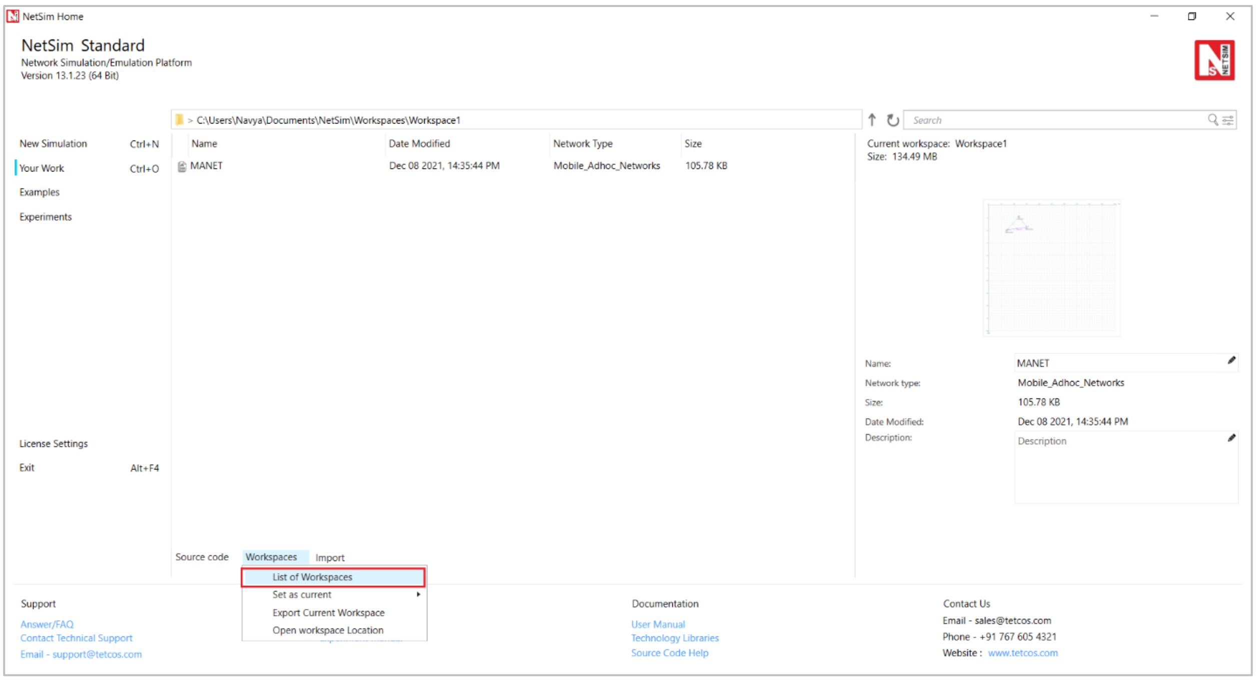
Name	Date Modified	Network Type	Size
MANET	Dec 08 2021, 14:35:44 PM	Mobile_Adhoc_Networks	105.78 KB

Current workspace: Workspace1
Size: 134.49 MB

Name:	MANET
Network type:	Mobile_Adhoc_Networks
Size:	105.78 KB
Date Modified:	Dec 08 2021, 14:35:44 PM
Description:	Description

Figure 4-18: Workspace options present in Your work Menu window

Then select **List of Workspaces** Options



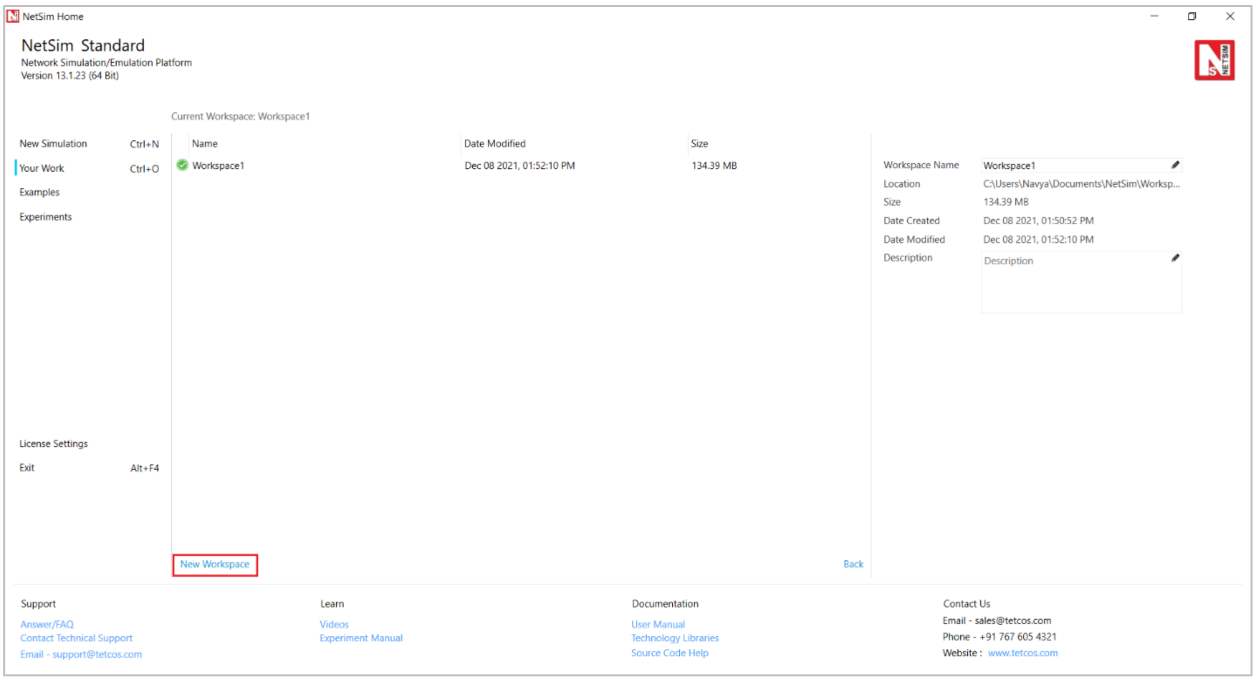
Name	Date Modified	Network Type	Size
MANET	Dec 08 2021, 14:35:44 PM	Mobile_Adhoc_Networks	105.78 KB

Current workspace: Workspace1
Size: 134.49 MB

Name:	MANET
Network type:	Mobile_Adhoc_Networks
Size:	105.78 KB
Date Modified:	Dec 08 2021, 14:35:44 PM
Description:	Description

Figure 4-19: Select List of Workspaces option in Your work window

And select **New Workspace**



NetSim Home

NetSim Standard
Network Simulation/Emulation Platform
Version 13.1.23 (64 Bit)

Current Workspace: Workspace1

		Name	Date Modified	Size
New Simulation	Ctrl+N	Workspace1	Dec 08 2021, 01:52:10 PM	134.39 MB
Your Work	Ctrl+O			
Examples				
Experiments				
License Settings				
Exit	Alt+F4			

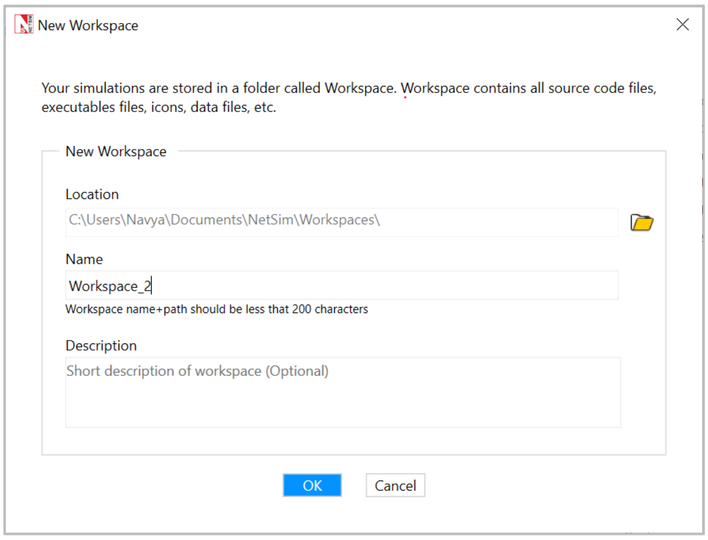
Workspace Name	Workspace1
Location	C:\Users\Navya\Documents\NetSim\Worksp...
Size	134.39 MB
Date Created	Dec 08 2021, 01:50:52 PM
Date Modified	Dec 08 2021, 01:52:10 PM
Description	Description

New Workspace   Back

Support: Answer/FAQ, Contact Technical Support, Email - support@tetcos.com
Learn: Videos, Experiment Manual
Documentation: User Manual, Technology Libraries, Source Code Help
Contact Us: Email - sales@tetcos.com, Phone - +91 767 605 4321, Website : www.tetcos.com

Figure 4-20: Select New Workspace option

A New Workspace pop-up window appears where you can input the Workspace Name, Description and Workspace Path as shown below



New Workspace

Your simulations are stored in a folder called Workspace. Workspace contains all source code files, executables files, icons, data files, etc.

New Workspace

Location
C:\Users\Navya\Documents\NetSim\Workspaces\

Name
Workspace_2
Workspace name+path should be less that 200 characters

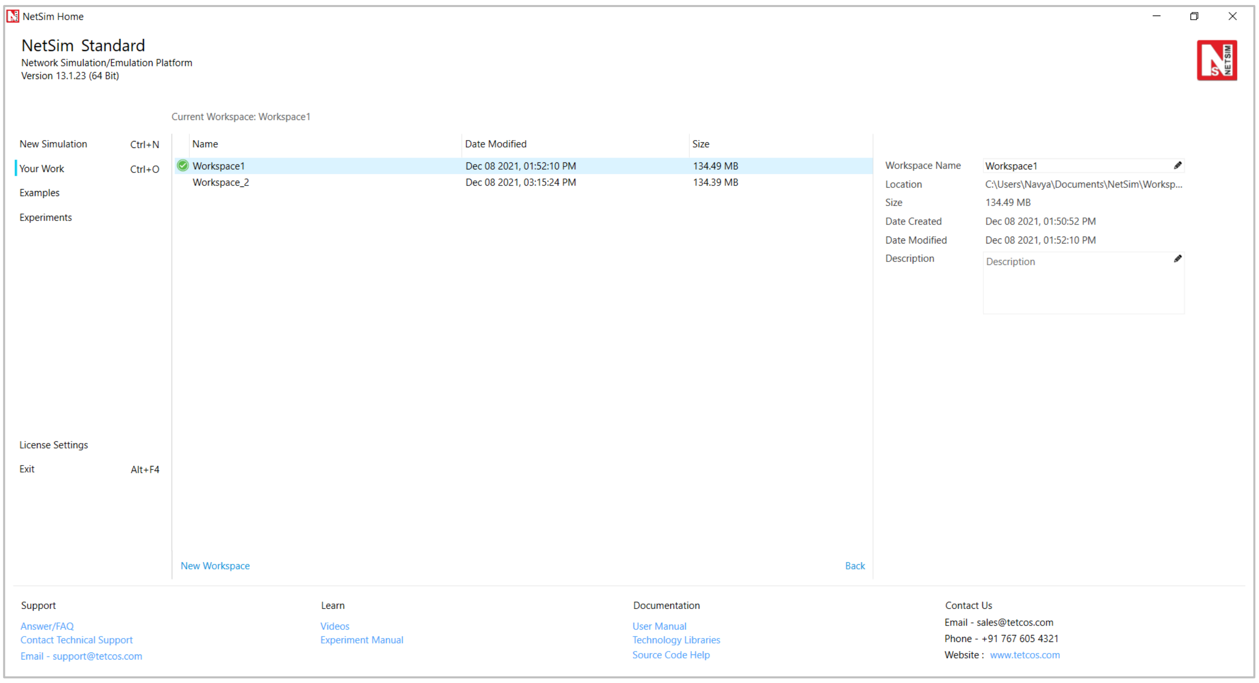
Description
Short description of workspace (Optional)

OK   Cancel

Figure 4-21: New Workspace pop-up window

4.7 How does a user switch between workspaces?

Users can switch from one workspace to another. Select Your work >> Workspaces >> List of Workspaces and click on the workspace you want to switch to



Current Workspace: Workspace1

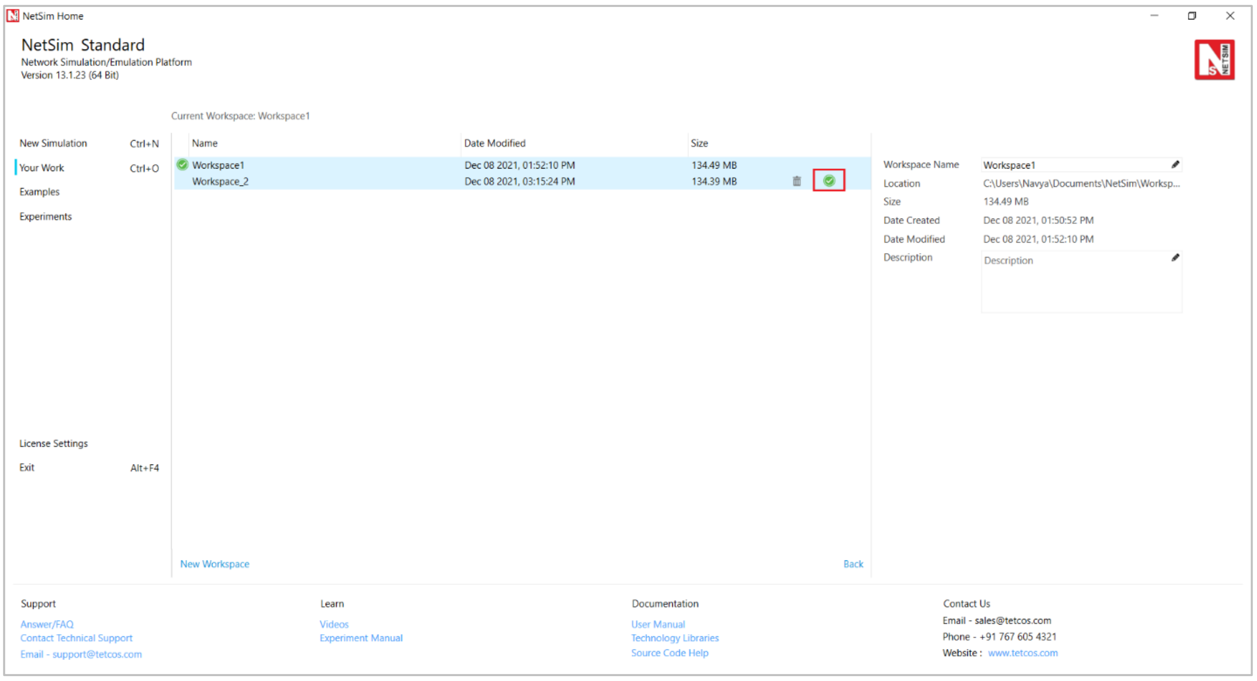
Name	Date Modified	Size
Workspace1	Dec 08 2021, 01:52:10 PM	134.49 MB
Workspace_2	Dec 08 2021, 03:15:24 PM	134.39 MB

Workspace Name	Workspace1
Location	C:\Users\Navya\Documents\NetSim\Worksp...
Size	134.49 MB
Date Created	Dec 08 2021, 01:50:52 PM
Date Modified	Dec 08 2021, 01:52:10 PM
Description	Description

Support: Answer/FAQ, Contact Technical Support, Email - support@tetcos.com
Learn: Videos, Experiment Manual
Documentation: User Manual, Technology Libraries, Source Code Help
Contact Us: Email - sales@tetcos.com, Phone - +91 767 605 4321, Website : www.tetcos.com

Figure 4-22: Switch between workspaces

And then select Set as Current symbol (the green tick mark) as shown below



Current Workspace: Workspace1

Name	Date Modified	Size
Workspace1	Dec 08 2021, 01:52:10 PM	134.49 MB
Workspace_2	Dec 08 2021, 03:15:24 PM	134.39 MB

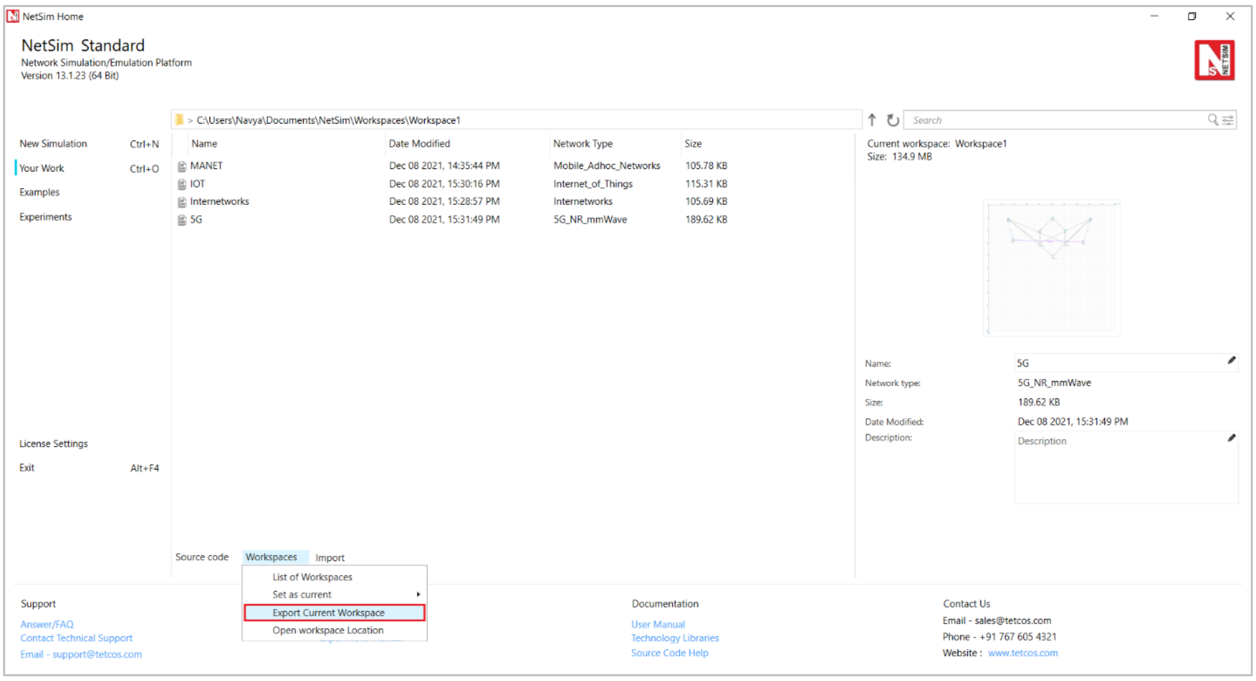
Workspace Name	Workspace1
Location	C:\Users\Navya\Documents\NetSim\Worksp...
Size	134.49 MB
Date Created	Dec 08 2021, 01:50:52 PM
Date Modified	Dec 08 2021, 01:52:10 PM
Description	Description

Support: Answer/FAQ, Contact Technical Support, Email - support@tetcos.com
Learn: Videos, Experiment Manual
Documentation: User Manual, Technology Libraries, Source Code Help
Contact Us: Email - sales@tetcos.com, Phone - +91 767 605 4321, Website : www.tetcos.com

Figure 4-23: Select Green button to “Set as Current” workspace

4.8 How does a user export a workspace?

Users can export only the workspace by selecting Your Work >> Workspaces >> Export Current Workspace as shown below in Figure 4-24.


NetSim Home

NetSim Standard
Network Simulation/Emulation Platform
Version 13.1.23 (64 Bit)

> C:\Users\Navya\Documents\NetSim\Workspaces\Workspace1

New Simulation Ctrl+N
Your Work Ctrl+O
Examples
Experiments

Name	Date Modified	Network Type	Size
MANET	Dec 08 2021, 14:35:44 PM	Mobile_Adhoc_Networks	105.78 KB
IOT	Dec 08 2021, 15:30:16 PM	Internet_of_Things	115.31 KB
Internetworks	Dec 08 2021, 15:28:57 PM	Internetworks	105.69 KB
5G	Dec 08 2021, 15:31:49 PM	5G_NR_mmWave	189.62 KB

Current workspace: Workspace1
Size: 134.9 MB

Name:	5G
Network type:	5G_NR_mmWave
Size:	189.62 KB
Date Modified:	Dec 08 2021, 15:31:49 PM
Description:	Description

License Settings
Exit Alt+F4

Source code Workspaces Import

List of Workspaces
Set as current
Export Current Workspace
Open workspace Location

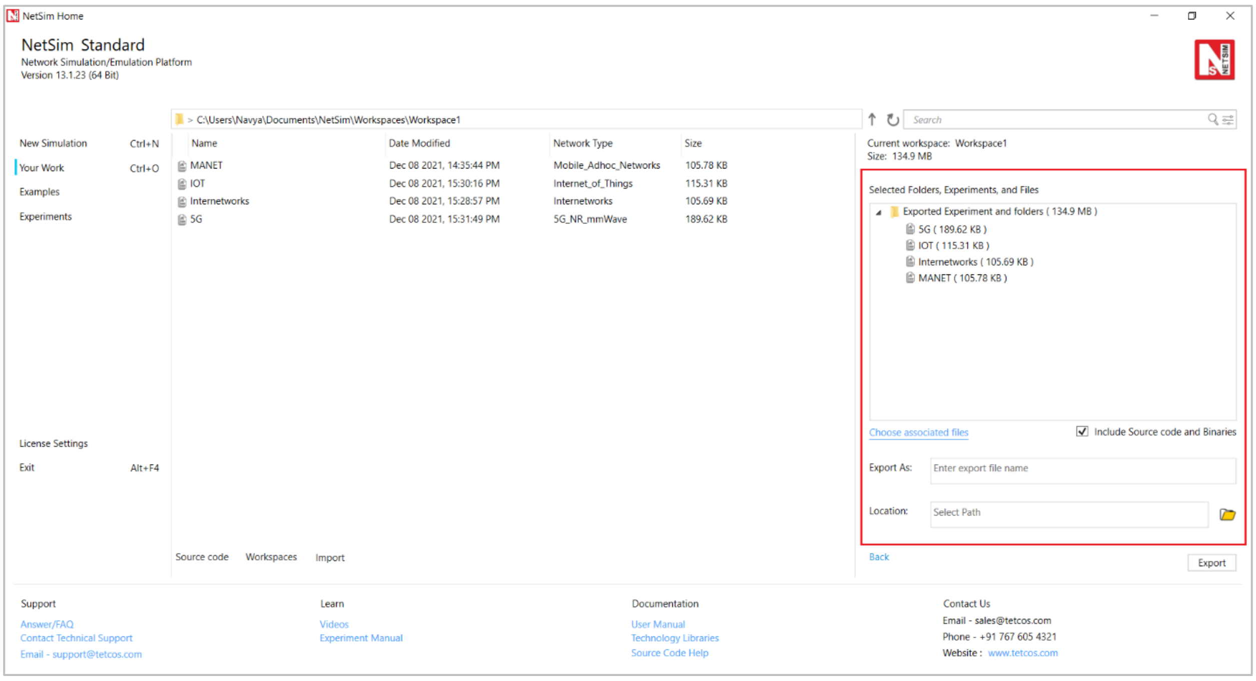
Support
Answer/FAQ
Contact Technical Support
Email - support@tetcos.com

Documentation
User Manual
Technology Libraries
Source Code Help

Contact Us
Email - sales@tetcos.com
Phone - +91 767 605 4321
Website : www.tetcos.com

Figure 4-24: Select Export option in Your work window

This will show the export workspace window panel in the right with all the existing experiments in that particular workspace. This option is similar to exporting an experiment. You can select the files which are to be exported as part of the workspace and then can select the source code and binaries if required. The Configuration file is mandatory and other files are optional.


NetSim Home

NetSim Standard
Network Simulation/Emulation Platform
Version 13.1.23 (64 Bit)

> C:\Users\Navya\Documents\NetSim\Workspaces\Workspace1

New Simulation Ctrl+N
Your Work Ctrl+O
Examples
Experiments

Name	Date Modified	Network Type	Size
MANET	Dec 08 2021, 14:35:44 PM	Mobile_Adhoc_Networks	105.78 KB
IOT	Dec 08 2021, 15:30:16 PM	Internet_of_Things	115.31 KB
Internetworks	Dec 08 2021, 15:28:57 PM	Internetworks	105.69 KB
5G	Dec 08 2021, 15:31:49 PM	5G_NR_mmWave	189.62 KB

Current workspace: Workspace1
Size: 134.9 MB

Selected Folders, Experiments, and Files
Exported Experiment and folders (134.9 MB)
5G (189.62 KB)
IOT (115.31 KB)
Internetworks (105.69 KB)
MANET (105.78 KB)

Choose associated files
✓ Include Source code and Binaries

Export As: Enter export file name
Location: Select Path

Back
Export

License Settings
Exit Alt+F4

Source code Workspaces Import

Support
Answer/FAQ
Contact Technical Support
Email - support@tetcos.com

Learn
Videos
Experiment Manual

Documentation
User Manual
Technology Libraries
Source Code Help

Contact Us
Email - sales@tetcos.com
Phone - +91 767 605 4321
Website : www.tetcos.com

Figure 4-25: Adding default binaries, source code and icons to Selected Experiments list

Users can enter the name and path in which the workspace is to be exported and then click on Export.

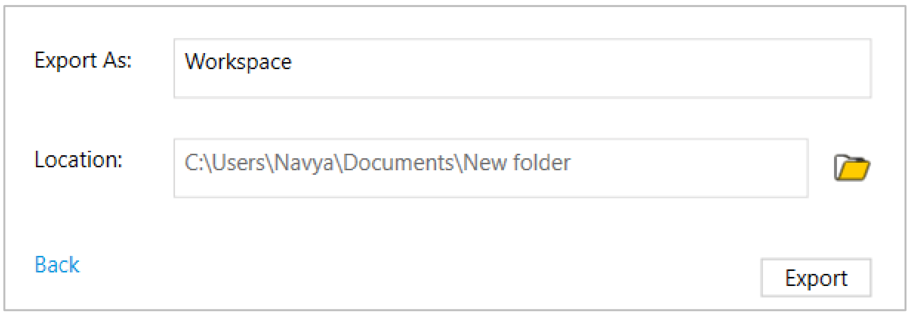
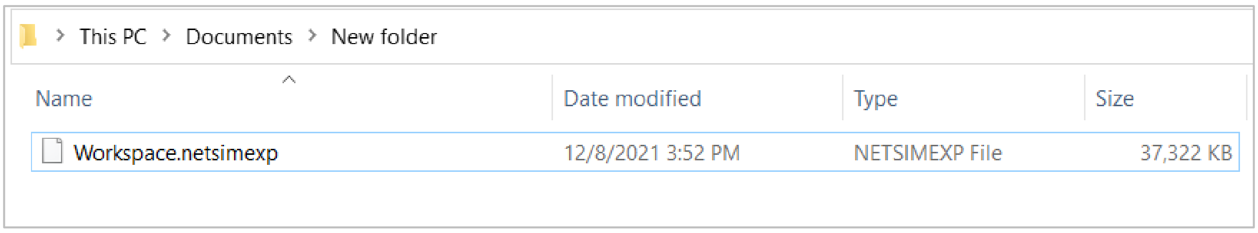


Figure 4-26: Enter the workspace name and select the export location

The workspace will be exported to the path selected. It will have the extension .netsimexp as shown in Figure 4-27.



Name	Date modified	Type	Size
Workspace.netsimexp	12/8/2021 3:52 PM	NETSIMEXP File	37,322 KB

Figure 4-27: Workspace exported to Location.

If you want to remove an experiment from the workspace being exported, right click on that experiment and click on remove as shown in Figure 4-28.

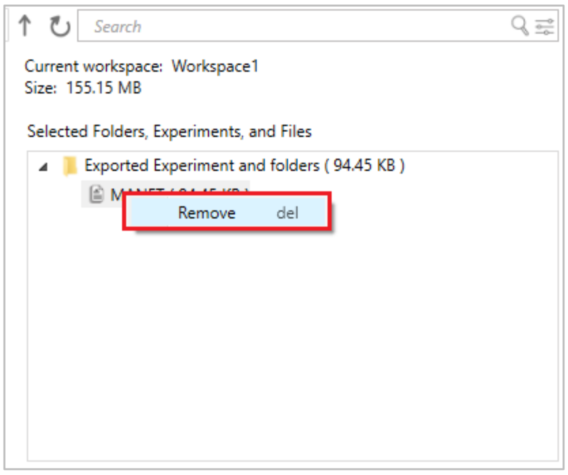
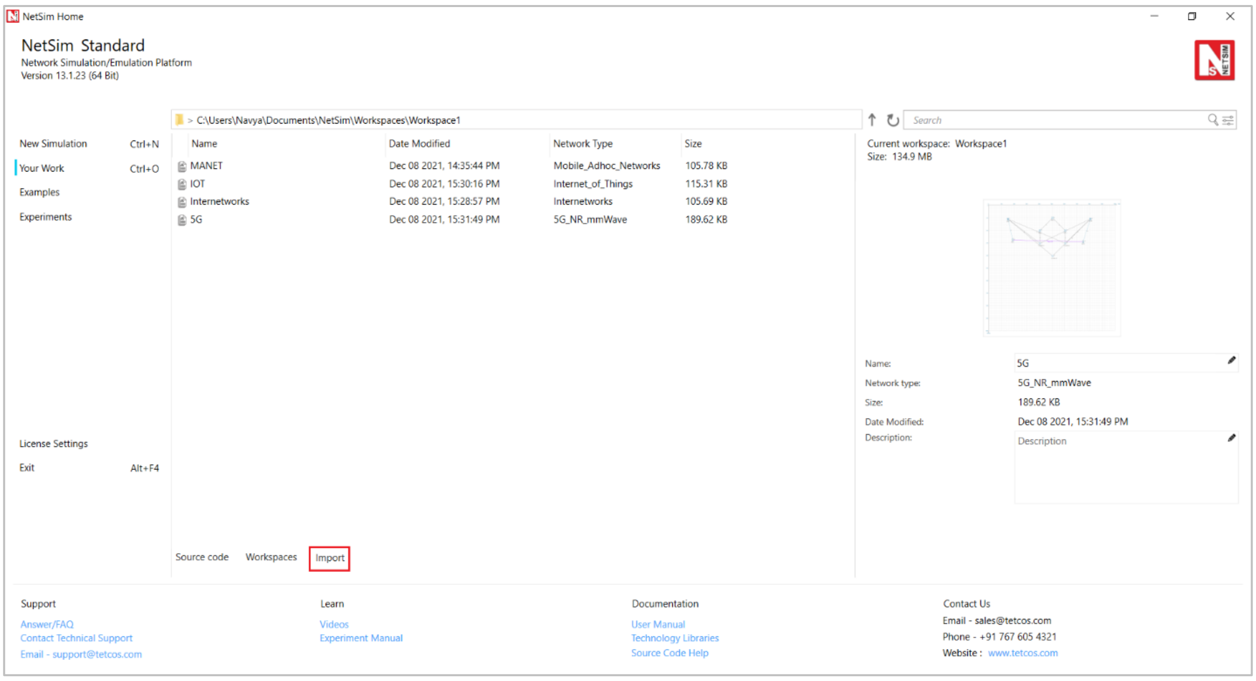

Search
Current workspace: Workspace1
Size: 155.15 MB
Selected Folders, Experiments, and Files
Exported Experiment and folders (94.45 KB)
Remove del

Figure 4-28: In Export list window Right click on experiment and select Remove

4.9 How does a user import experiment and workspace?

You can import only an exported workspace, import experiments and workspaces by first selecting Your work and then selecting the Import option as shown in Figure 4-29.



C:\Users\Navya\Documents\NetSim\Workspaces\Workspace1

Name	Date Modified	Network Type	Size
MANET	Dec 08 2021, 14:35:44 PM	Mobile_Adhoc_Networks	105.78 KB
IOT	Dec 08 2021, 15:30:16 PM	Internet_of_Things	115.31 KB
Internetworks	Dec 08 2021, 15:28:57 PM	Internetworks	105.69 KB
5G	Dec 08 2021, 15:31:49 PM	5G_NR_mmWave	189.62 KB

Current workspace: Workspace1
Size: 134.9 MB

Name:	5G
Network type:	5G_NR_mmWave
Size:	189.62 KB
Date Modified:	Dec 08 2021, 15:31:49 PM
Description:	Description

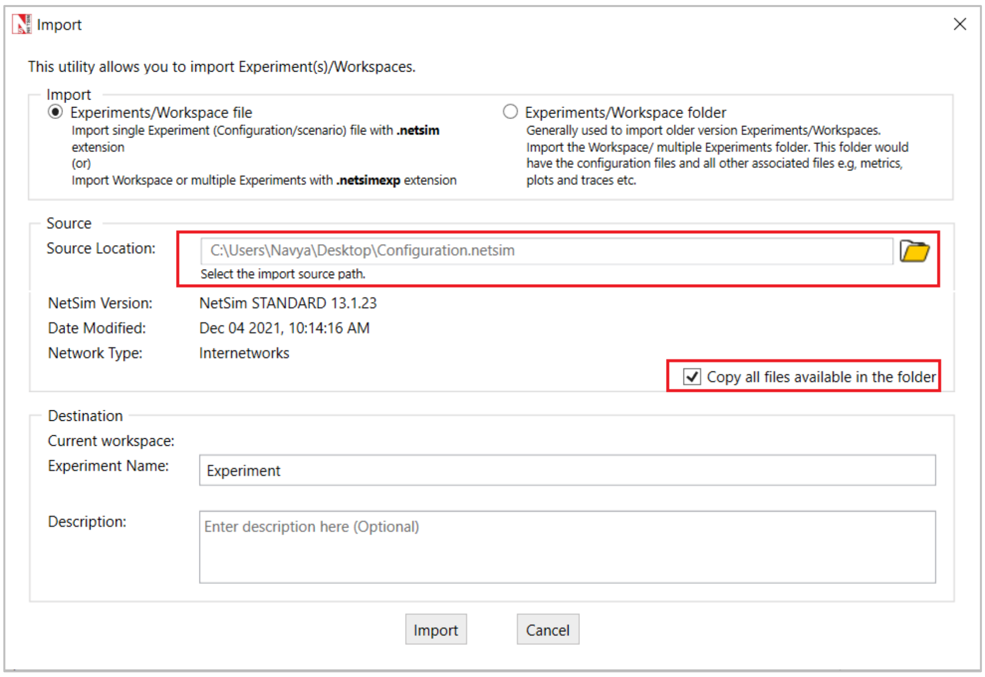
Figure 4-29: Select Import option in Your work window

4.9.1 Importing Configuration.netsim file from experiment folder

Once you click the import option from Your work the following window will open. Click on **Experiment/Workspace file** option and import the Configuration file. Enter the path from where the configuration file has to be imported as shown in Figure 4-30.

Note that

- Only files with "**.netsim"** extension can be imported.
- By Selecting "**Copy all files available in the folder**" option user can import all files present along with Configuration.netsim.



Import

This utility allows you to import Experiment(s)/Workspaces.

Import

[● Experiments/Workspace file] Import single Experiment (Configuration/scenario) file with **.netsim** extension (or) Import Workspace or multiple Experiments with **.netsimexp** extension

○ Experiments/Workspace folder Generally used to import older version Experiments/Workspaces. Import the Workspace/ multiple Experiments folder. This folder would have the configuration files and all other associated files e.g, metrics, plots and traces etc.

Source

Source Location:	C:\Users\Navya\Desktop\Configuration.netsim
	Select the import source path.
NetSim Version:	NetSim STANDARD 13.1.23
Date Modified:	Dec 04 2021, 10:14:16 AM
Network Type:	Internetworks

[✓] Copy all files available in the folder

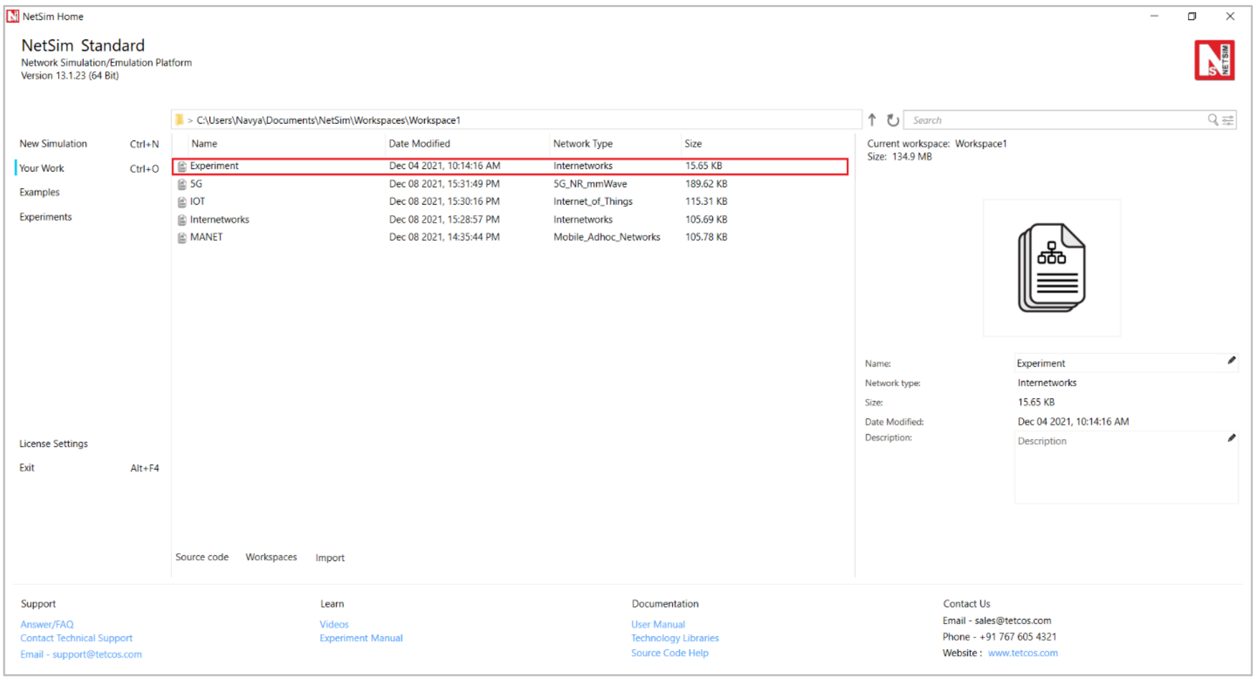
Destination

Current workspace:	
Experiment Name:	Experiment
Description:	Enter description here (Optional)

Import Cancel

Figure 4-30: Importing configuration files

The imported experiment file will be available in the current workspace. It can be seen by clicking on Your Work as shown below



NetSim Home

NetSim Standard
Network Simulation/Emulation Platform
Version 13.1.23 (64 Bit)

New Simulation	Ctrl+N
Your Work	Ctrl+O
Examples	
Experiments	
License Settings	
Exit	Alt+F4

C:\Users\Navya\Documents\NetSim\Workspaces\Workspace1

Name	Date Modified	Network Type	Size
Experiment	Dec 04 2021, 10:14:16 AM	Internetworks	15.65 KB
5G	Dec 08 2021, 15:31:49 PM	5G_NR_mmWave	189.62 KB
IOT	Dec 08 2021, 15:30:16 PM	Internet_of_Things	115.31 KB
Internetworks	Dec 08 2021, 15:28:57 PM	Internetworks	105.69 KB
MANET	Dec 08 2021, 14:35:44 PM	Mobile_Adhoc_Networks	105.78 KB

Source code Workspaces Import

Current workspace: Workspace1
Size: 134.9 MB

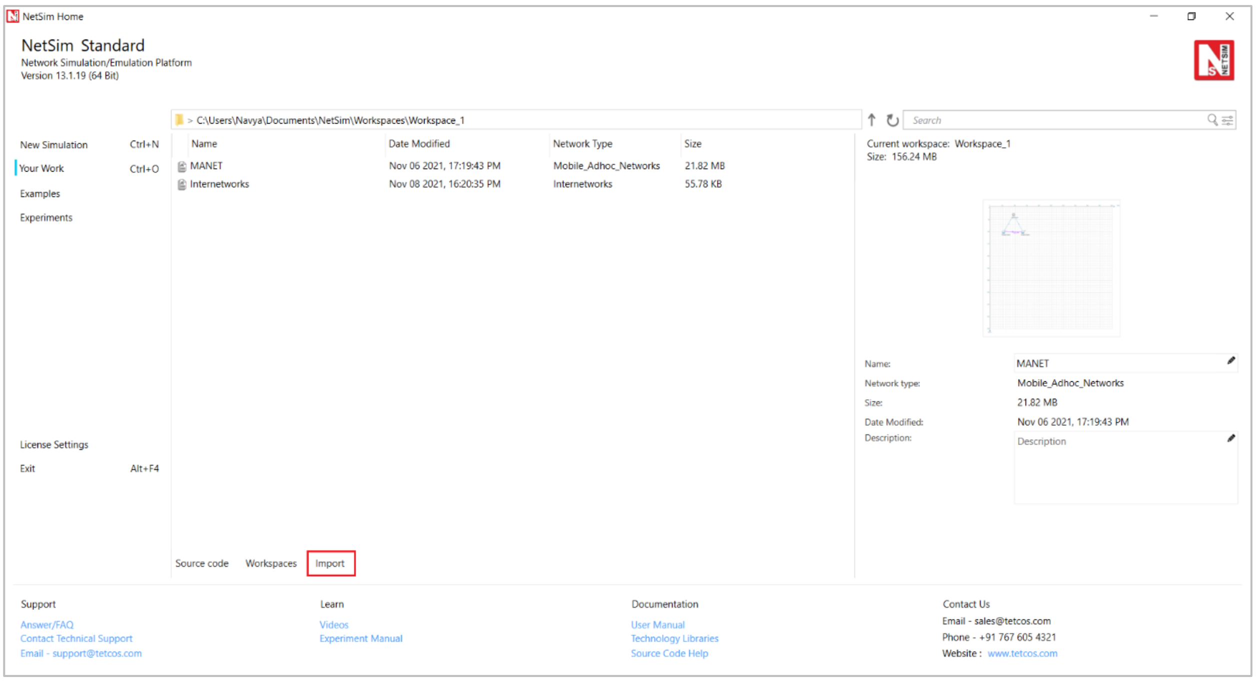
Name:	Experiment
Network type:	Internetworks
Size:	15.65 KB
Date Modified:	Dec 04 2021, 10:14:16 AM
Description:	Description

Support: Answer/FAQ, Contact Technical Support, Email - support@tetcos.com
Learn: Videos, Experiment Manual
Documentation: User Manual, Technology Libraries, Source Code Help
Contact Us: Email - sales@tetcos.com, Phone - +91 767 605 4321, Website : www.tetcos.com

Figure 4-31: Imported experiment file in the current workspace

4.9.2 Import workspace or multiple experiments file

This section explains how (i) You can import multiple experiments into your current workspace, and (ii) Import a complete workspace. Click Your work and then select Import option as shown below


NetSim Home

NetSim Standard
Network Simulation/Emulation Platform
Version 13.1.19 (64 Bit)

> C:\Users\Navya\Documents\NetSim\Workspaces\Workspace_1

New Simulation Ctrl+N	Name	Date Modified	Network Type	Size
Your Work Ctrl+O	MANET	Nov 06 2021, 17:19:43 PM	Mobile_Adhoc_Networks	21.82 MB
Examples	Internetworks	Nov 08 2021, 16:20:35 PM	Internetworks	55.78 KB
Experiments				

Current workspace: Workspace_1
Size: 156.24 MB

Name:	MANET
Network type:	Mobile_Adhoc_Networks
Size:	21.82 MB
Date Modified:	Nov 06 2021, 17:19:43 PM
Description:	Description

License Settings
Exit Alt+F4

Source code Workspaces Import

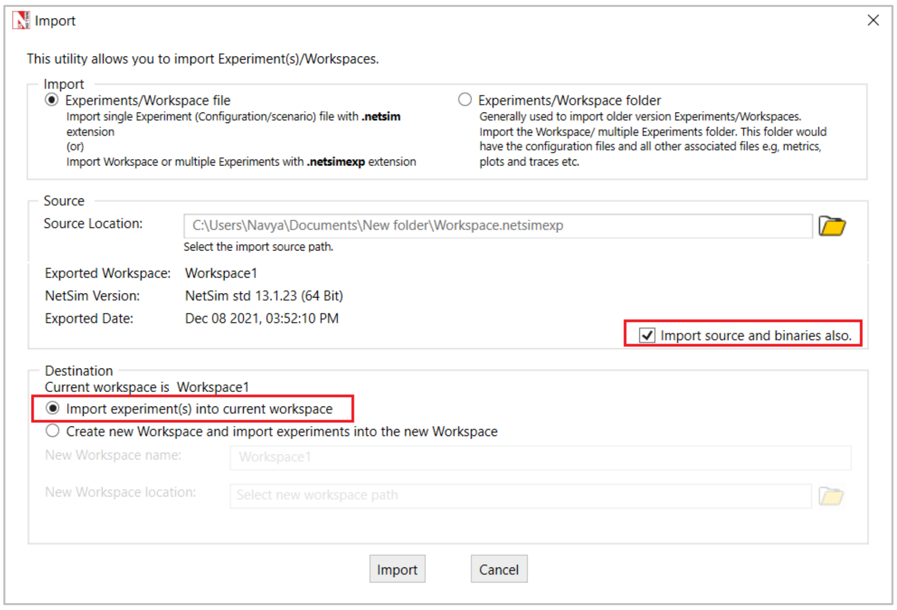
Support: Answer/FAQ, Contact Technical Support, Email - support@tetcos.com
Learn: Videos, Experiment Manual
Documentation: User Manual, Technology Libraries, Source Code Help
Contact Us: Email - sales@tetcos.com, Phone - +91 767 605 4321, Website : www.tetcos.com

Figure 4-32: Import workspace/multiple experiments option in Your work window

You need to input the path from where (i) the experiments (a single folder) or (ii) the workspace will be imported. To import multiple experiments into the current workspace, click on the option as shown in Figure 4-33.

Note that

- Only previously exported experiment/workspaces with ". **netsim_exp**" extension can be imported.
- By Selecting "**Import source and binaries also**" user can import source code and binaries present along with the exported experiments or workspace.


Import

This utility allows you to import Experiment(s)/Workspaces.

Import
[✓ Experiments/Workspace file] Import single Experiment (Configuration/scenario) file with **.netsim** extension (or) Import Workspace or multiple Experiments with **.netsimexp** extension
Experiments/Workspace folder: Generally used to import older version Experiments/Workspaces. Import the Workspace/ multiple Experiments folder. This folder would have the configuration files and all other associated files e.g, metrics, plots and traces etc.

Source
Source Location: C:\Users\Navya\Documents\New folder\Workspace.netsimexp
Select the import source path.

Exported Workspace: Workspace1
NetSim Version: NetSim std 13.1.23 (64 Bit)
Exported Date: Dec 08 2021, 03:52:10 PM

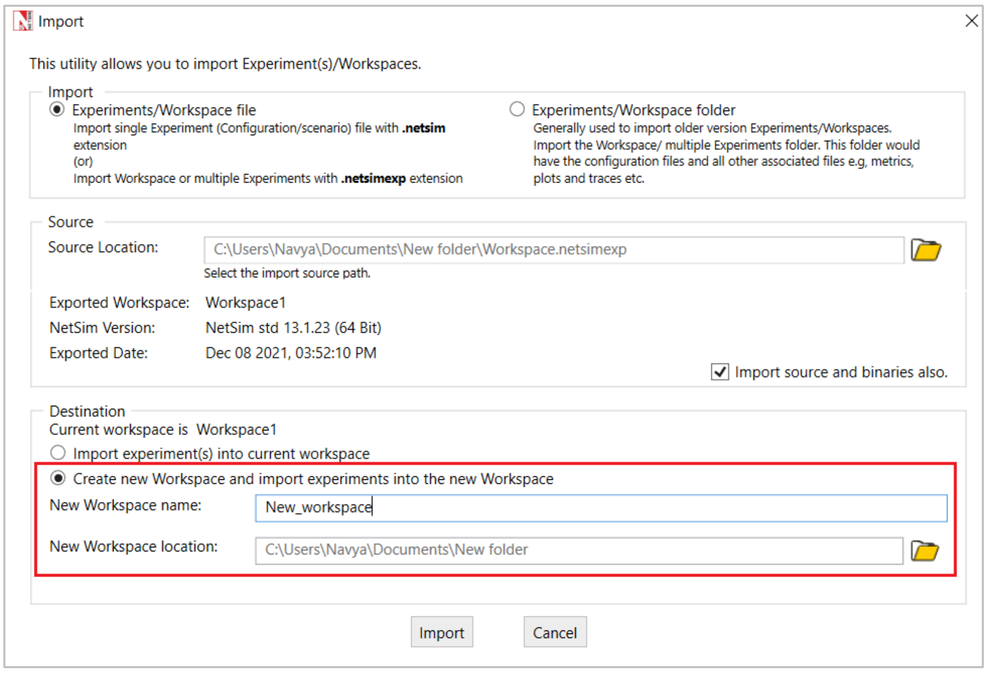
[✓ Import source and binaries also.]

Destination
Current workspace is Workspace1
[✓ Import experiment(s) into current workspace]
Create new Workspace and import experiments into the new Workspace
New Workspace name: Workspace1
New Workspace location: Select new workspace path

Import Cancel

Figure 4-33: Window for importing multiple experiments or workspace into current workspace

If the user wants to import the experiments into the new workspace, select the option as shown in Figure 4-34, and proceed accordingly.



Import

This utility allows you to import Experiment(s)/Workspaces.

Import

[✓ Experiments/Workspace file] Import single Experiment (Configuration/scenario) file with **.netsim** extension (or) Import Workspace or multiple Experiments with **.netsimexp** extension

Experiments/Workspace folder: Generally used to import older version Experiments/Workspaces. Import the Workspace/ multiple Experiments folder. This folder would have the configuration files and all other associated files e.g, metrics, plots and traces etc.

Source

Source Location:	C:\Users\Navya\Documents\New folder\Workspace.netsimexp
	Select the import source path.
Exported Workspace:	Workspace1
NetSim Version:	NetSim std 13.1.23 (64 Bit)
Exported Date:	Dec 08 2021, 03:52:10 PM

[✓ Import source and binaries also.]

Destination

Current workspace is Workspace1

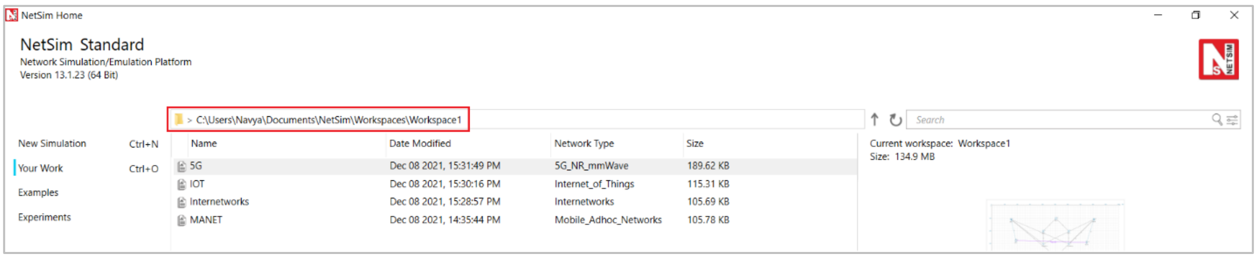
Import experiment(s) into current workspace

[✓ Create new Workspace and import experiments into the new Workspace]

New Workspace name:	New_workspace
New Workspace location:	C:\Users\Navya\Documents\New folder

Figure 4-34: Create a new workspace, and import experiments and source code/binaries, into the new workspace

If you import the experiments into the current workspace then experiments will be displayed in the Your Work menu of the current workspace as shown below.



NetSim Home

NetSim Standard
Network Simulation/Emulation Platform
Version 13.1.23 (64 Bit)

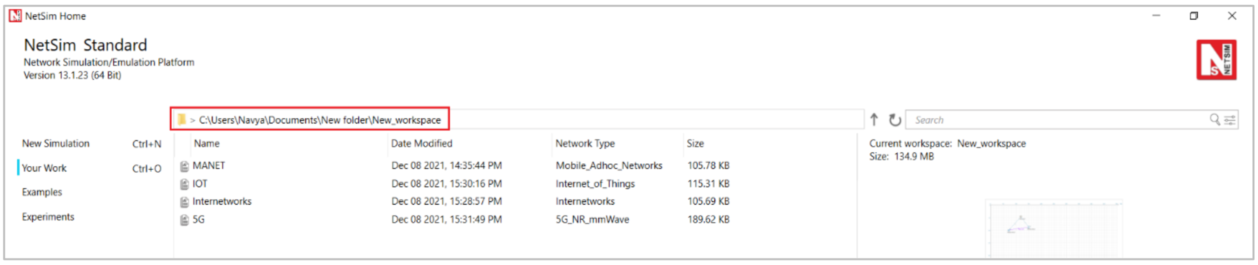
C:\Users\Navya\Documents\NetSim\Workspaces\Workspace1

Name	Date Modified	Network Type	Size
5G	Dec 08 2021, 15:31:49 PM	5G_NR_mmWave	189.62 KB
IOT	Dec 08 2021, 15:30:16 PM	Internet_of_Things	115.31 KB
Internetworks	Dec 08 2021, 15:28:57 PM	Internetworks	105.69 KB
MANET	Dec 08 2021, 14:35:44 PM	Mobile_Adhoc_Networks	105.78 KB

Current workspace: Workspace1
Size: 134.9 MB

Figure 4-35: Two Imported experiments shown in Your Work of current workspace

If you create a new workspace and import experiments and binaries/source code then, these experiments will be shown in the Your Work Menu of the new workspace



NetSim Home

NetSim Standard
Network Simulation/Emulation Platform
Version 13.1.23 (64 Bit)

C:\Users\Navya\Documents\New folder\New_workspace

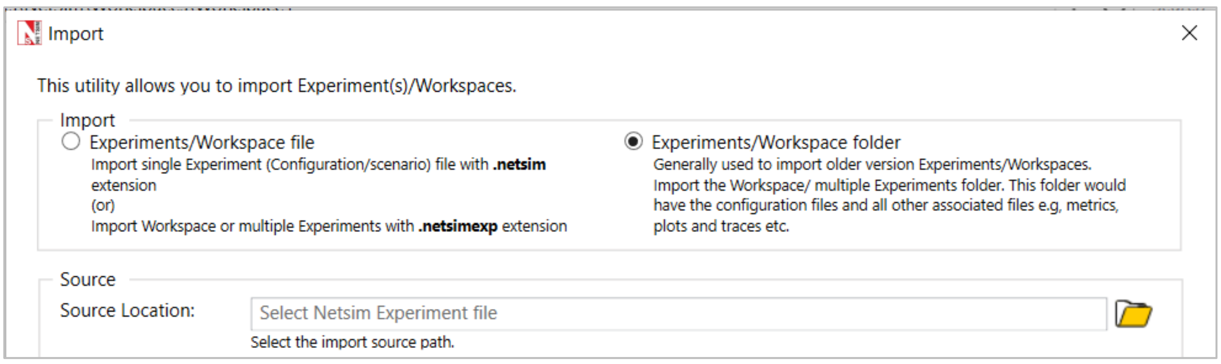
Name	Date Modified	Network Type	Size
MANET	Dec 08 2021, 14:35:44 PM	Mobile_Adhoc_Networks	105.78 KB
IOT	Dec 08 2021, 15:30:16 PM	Internet_of_Things	115.31 KB
Internetworks	Dec 08 2021, 15:28:57 PM	Internetworks	105.69 KB
5G	Dec 08 2021, 15:31:49 PM	5G_NR_mmWave	189.62 KB

Current workspace: New_workspace
Size: 134.9 MB

Figure 4-36: Two Imported experiments in Your Work menu of a newly created workspace

4.10 Import Experiments or Workspace folder

User can import an experiment folder or a workspace folder by choosing the following option in the experiment import window as shown below



Import

This utility allows you to import Experiment(s)/Workspaces.

Import

○ Experiments/Workspace file
Import single Experiment (Configuration/scenario) file with **.netsim** extension
(or)
Import Workspace or multiple Experiments with **.netsimexp** extension

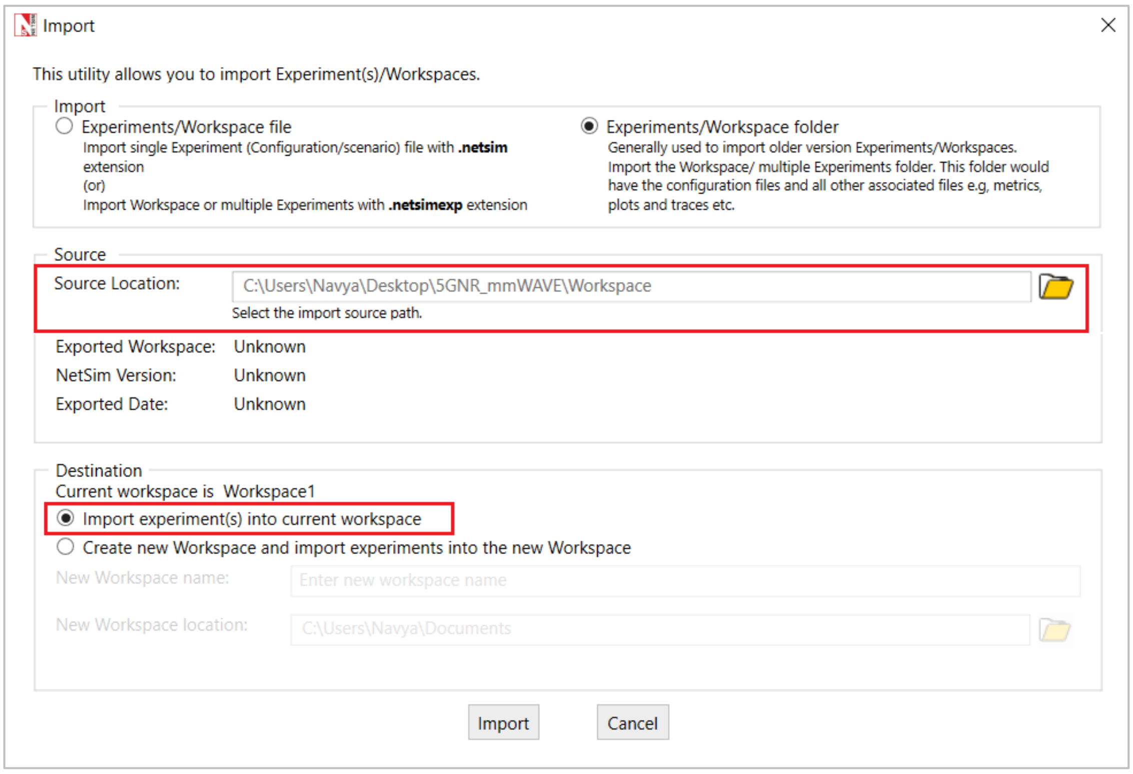
● Experiments/Workspace folder
Generally used to import older version Experiments/Workspaces.
Import the Workspace/ multiple Experiments folder. This folder would have the configuration files and all other associated files e.g, metrics, plots and traces etc.

Source

Source Location: Select Netsim Experiment file
Select the import source path.

Figure 4-37: Option to import Experiment/Workspace folder

You should give the path from where the workspace/experiment folder must be imported. Then click on import as shown below



Import

This utility allows you to import Experiment(s)/Workspaces.

Import

○ Experiments/Workspace file
Import single Experiment (Configuration/scenario) file with **.netsim** extension
(or)
Import Workspace or multiple Experiments with **.netsimexp** extension

● Experiments/Workspace folder
Generally used to import older version Experiments/Workspaces.
Import the Workspace/ multiple Experiments folder. This folder would have the configuration files and all other associated files e.g, metrics, plots and traces etc.

Source

Source Location: C:\Users\Navya\Desktop\5GNR_mmWAVE\Workspace
Select the import source path.

Exported Workspace:	Unknown
NetSim Version:	Unknown
Exported Date:	Unknown

Destination

Current workspace is Workspace1

● Import experiment(s) into current workspace

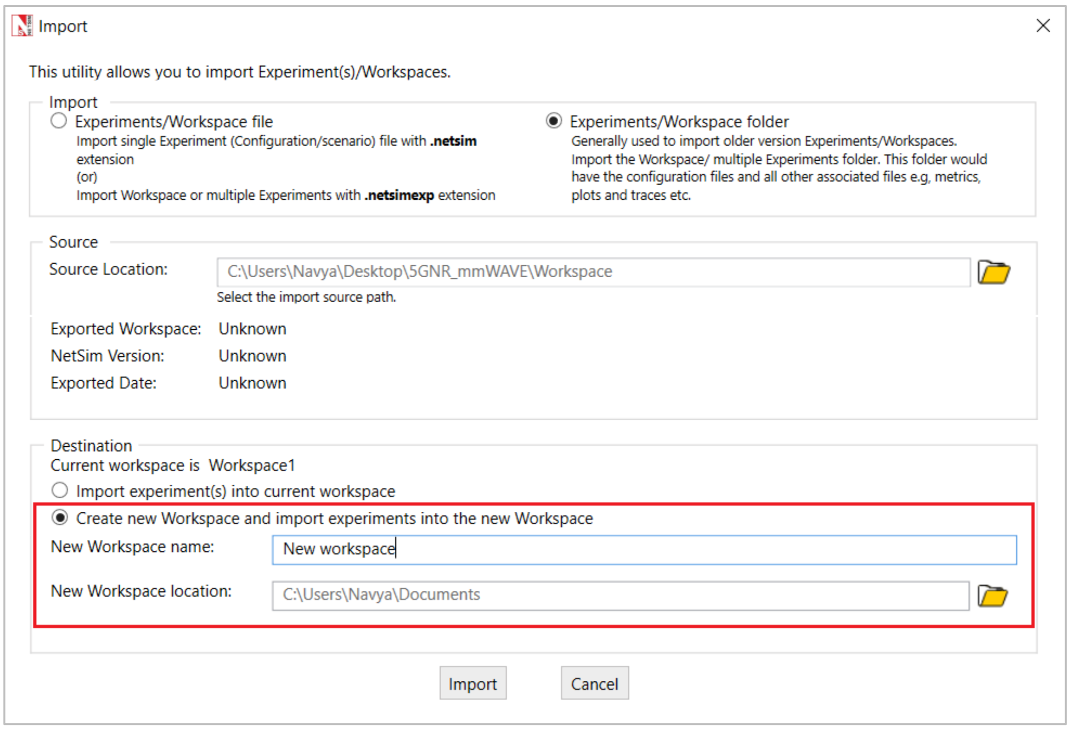
○ Create new Workspace and import experiments into the new Workspace

New Workspace name: Enter new workspace name

New Workspace location: C:\Users\Navya\Documents

Figure 4-38: Window for navigation of experiments/workspace into current workspace

If you want to import the folder into new workspace, choose the option as shown below



Import

This utility allows you to import Experiment(s)/Workspaces.

Import

( ) Experiments/Workspace file
Import single Experiment (Configuration/scenario) file with **.netsim** extension
(or)
Import Workspace or multiple Experiments with **.netsimexp** extension

(●) Experiments/Workspace folder
Generally used to import older version Experiments/Workspaces.
Import the Workspace/ multiple Experiments folder. This folder would have the configuration files and all other associated files e.g, metrics, plots and traces etc.

Source

Source Location:	C:\Users\Navya\Desktop\5GNR_mmWAVE\Workspace
	Select the import source path.
Exported Workspace:	Unknown
NetSim Version:	Unknown
Exported Date:	Unknown

Destination
Current workspace is Workspace1
( ) Import experiment(s) into current workspace
(●) Create new Workspace and import experiments into the new Workspace

New Workspace name:	New workspace
New Workspace location:	C:\Users\Navya\Documents

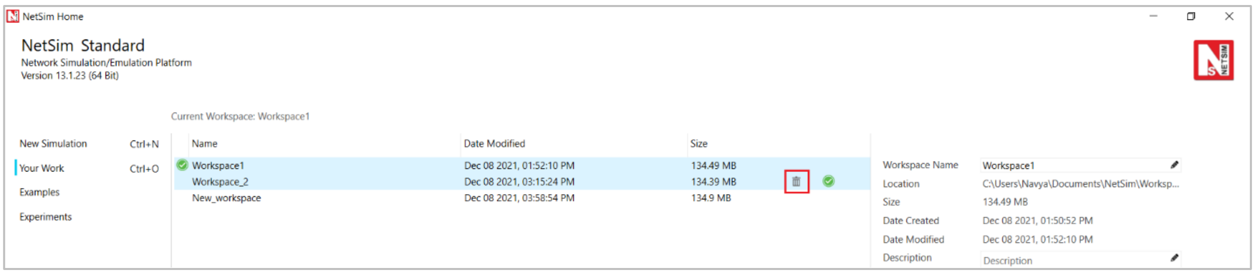
Figure 4-39: Importing the Experiment folder into new workspace

4.11 Import into current workspace vs. creating a new workspace

A new workspace generally needs to be created when the underlying source code is likely to be modified. If you are importing only experiments, then they can be imported into your existing workspace. However, if you wish to import experiments plus the binaries/source codes then we recommend you create a new workspace for the same.

4.12 How does a user delete a workspace?

You can delete a workspace by clicking on the delete icon shown below



NetSim Home

NetSim Standard
Network Simulation/Emulation Platform
Version 13.1.23 (64 Bit)

Current Workspace: Workspace1

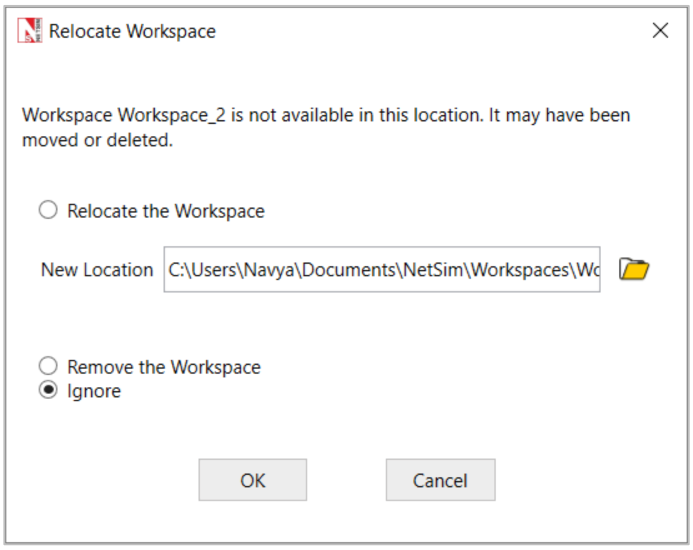
Name	Date Modified	Size
✓ Workspace1	Dec 08 2021, 01:52:10 PM	134.49 MB
Workspace_2	Dec 08 2021, 03:15:24 PM	134.39 MB
New_workspace	Dec 08 2021, 03:58:54 PM	134.9 MB

Workspace Name	Workspace1
Location	C:\Users\Navya\Documents\NetSim\Worksp...
Size	134.49 MB
Date Created	Dec 08 2021, 01:50:52 PM
Date Modified	Dec 08 2021, 01:52:10 PM
Description	Description

Figure 4-40: Delete a workspace by clicking on the delete icon in Your work window

Deleting the current workspace is not allowed. To delete the current workspace, you need to set another workspace as the current workspace (click on that workspace and then on the green tick mark icon) and then delete the workspace. Deleting a workspace will delete all the saved experiments and code modifications done in that workspace.

The following window appears if the current workspace is deleted or removed after re-installation of NetSim.


Relocate Workspace

Workspace Workspace_2 is not available in this location. It may have been moved or deleted.

○ Relocate the Workspace

New Location C:\Users\Navya\Documents\NetSim\Workspaces\Wo

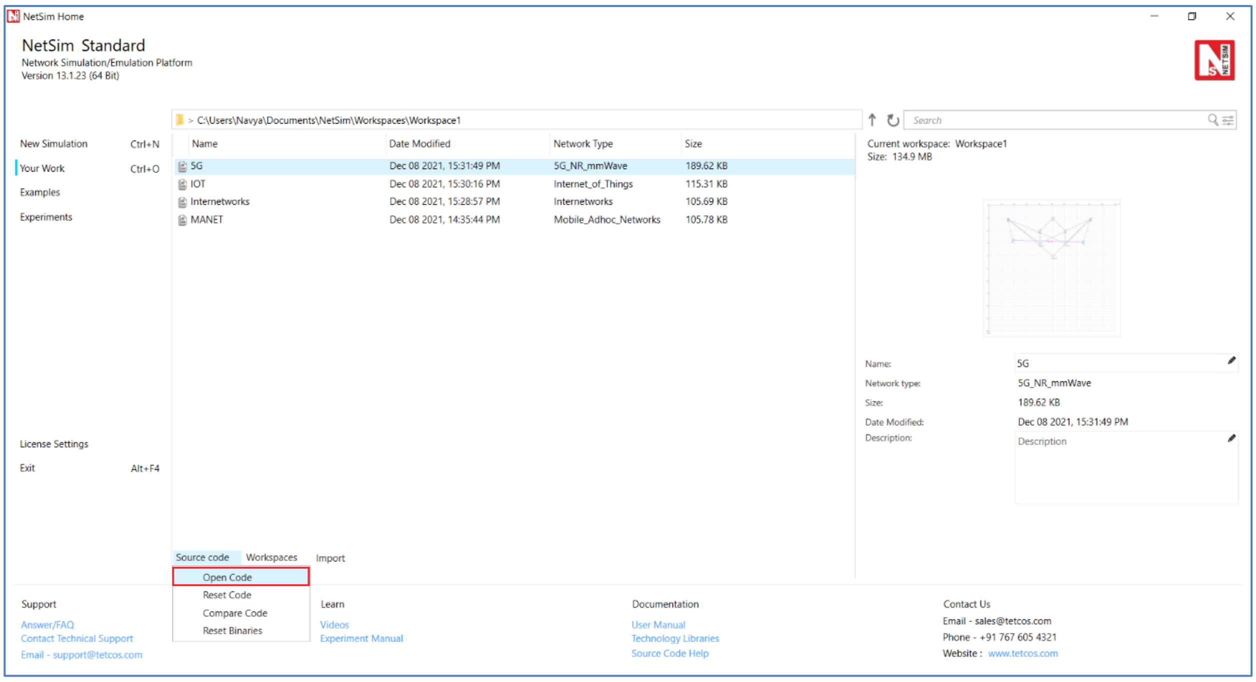
○ Remove the Workspace
● Ignore

Figure 4-41: Relocation workspace window

The "Relocate the workspace" option will allow the user to select a new location for the workspace which was removed/ deleted. User can also ignore the message by selecting "Ignore" option and clicking on OK button.

4.13 How does a user open and modify source codes?

You can modify the source codes within a workspace. For doing this, select Your work ->Source code ->Open Code as shown below


NetSim Home

NetSim Standard
Network Simulation/Emulation Platform
Version 13.1.23 (64 Bit)

C:\Users\Navya\Documents\NetSim\Workspaces\Workspace1

Name	Date Modified	Network Type	Size
5G	Dec 08 2021, 15:31:49 PM	5G_NR_mmWave	189.62 KB
IOT	Dec 08 2021, 15:30:16 PM	Internet_of_Things	115.31 KB
Internetworks	Dec 08 2021, 15:28:57 PM	Internetworks	105.69 KB
MANET	Dec 08 2021, 14:35:44 PM	Mobile_Adhoc_Networks	105.78 KB

Current workspace: Workspace1
Size: 134.9 MB

Name:	5G
Network type:	5G_NR_mmWave
Size:	189.62 KB
Date Modified:	Dec 08 2021, 15:31:49 PM
Description:	Description

Support
Answer/FAQ
Contact Technical Support
Email - support@tetcos.com

Learn
Videos
Experiment Manual

Documentation
User Manual
Technology Libraries
Source Code Help

Contact Us
Email - sales@tetcos.com
Phone - +91 767 605 4321
Website : www.tetcos.com

Figure 4-42: Open code option is available in Your work window

This opens the C source codes in MS Visual Studio. You can then modify the protocol codes and compile the codes. Then create a network in NetSim or open the saved experiment which simulates the protocol that has been modified. Run the simulation. This simulation will run per the modified code. Note that the changes in the source codes applies to the current workspace only.

4.14 How do I reset my code changes?

Each workspace has two Reset options. They are reset:

1. Binaries (compiled files) to default
2. Code (source C codes) to default

5 Simulating different networks in NetSim

The following table lists the networking technologies available in the different versions of NetSim.

Type of Network	NetSim Versions
Internetwork	All versions
Legacy Network	All versions
Cellular Network	All versions
MANET	All versions
Wireless Sensor Network	All versions
Software Defined Network	All versions
Internet of Things	All versions
Cognitive Radio	All versions
LTE	All versions
5G NR	Available only with NetSim Standard and NetSim Pro versions
VANET	Available only with NetSim Standard and NetSim Pro versions
Satellite Communication	Available only with NetSim Standard and NetSim Pro versions
Network Emulator (Add On)	Available only with NetSim Standard and NetSim Pro versions
TDMA Radio Networks (Add on)	Available only with NetSim Pro version

Figure 5-1: Networking technologies available in the different versions of NetSim

NetSim comes with inbuilt examples to help you understand how the different types of networks work.

The devices models in NetSim represent common networking devices in a generic way and do not model any specific vendor's implementation of the device. In real-world networks, each device has specific vendor implementation of networking protocols.

5.1 Internetworks

An Internetwork is a collection of two or more computer networks (typically Local Area Networks or LANs) which are interconnected to form a bigger network.

Internetwork's library in NetSim covers Ethernet, Address Resolution Protocol (ARP), Wireless LAN – 802.11 a / b / g / n / ac and e, Internet Protocol (IP), Transmission Control Protocol (TCP), Virtual LAN (VLAN), User Datagram Protocol (UDP), and routing protocols such as Routing Information Protocol (RIP), Open Shortest Path First (OSPF), Internet Group Management Protocol (IGMP), and Protocol Independent Multicast (PIM).

To simulate Internetworks, click on New Simulation and then click on Internetworks.

5.1.1 Internetworks Examples

To simulate the Examples for different types of Internetworks

1. Go to the NetSim UI and click **Examples**.
   The Example Simulation pane appears at the right.
2. Click the **Internetwork** example you wish to simulate. NetSim UI loads the example.

5.1.2 Internetwork Documentation

To view help documentation users can either click on "Technology Libraries" under documentation in the home screen or click the 'Book' link located next to Internetworks in examples. The help documentation explains the following:

- Introduction
- Simulation GUI
- Model Features
- Featured Examples
- Internetworks Experiments in NetSim
- Reference Documents
- Latest FAQ available online

5.2 Legacy Networks

Legacy networks cover older generation protocols which are rarely used today and not part of the TCP/IP protocol suite. With the advent of TCP/IP as a common networking platform in the mid-1970s, most legacy networks are no longer used.

NetSim Legacy Network library cover Pure Aloha and Slotted Aloha.

ALOHA is a protocol that was developed at the University of Hawaii and used for satellite communication systems in the Pacific. ALOHA protocol was designed to send and receive messages between multiple stations, on a shared medium. Slotted ALOHA is improvised version of pure ALOHA designed to reduce the chances of collisions when sending data between the sender and the receiver.

To simulate Legacy Networks, click on New Simulation and then under Legacy networks click on either Pure Aloha or Slotted Aloha

5.2.1 Legacy Networks Examples

To simulate Pure ALOHA and Slotted ALOHA Examples:

1. Go to the NetSim UI and click **Examples**.
   The Example Simulation pane appears at the right.
2. Click the **Legacy Network** example you want to simulate. NetSim UI loads the example.

5.2.2 Legacy Network Documentation

To view help documentation either click on "Technology Libraries" under documentation in the home screen or click the 'Book' link located next Legacy Networks in examples. The help documentation explains the following:

- Introduction
- Simulation GUI
- Legacy Networks Experiments in NetSim
- Note: Release on Unsupported Basis
- Latest FAQ available online

5.3 Cellular Networks

A cellular network (also known as a mobile network) is a communication network where the last link is wireless. The network is distributed over land areas called cells. Every cell is served by at least one fixed-location transceiver known as a base station. These cells together provide radio coverage over larger geographical areas. User equipment's such as mobile phones can communicate even if the user is moving across different cells.

NetSim cellular networks library covers Global System for Mobile communication (GSM) and Code-Division Multiple Access (CDMA).

To simulate Cellular Networks, click on New Simulation and then under Cellular networks click on either GSM or CDMA.

5.3.1 Cellular Networks Examples

To simulate GSM and CDMA Examples:

1. Go to the NetSim UI and click **Examples**.
   The Example Simulation pane appears at the right.
2. Click the **Cellular Network** example you want to simulate. NetSim UI loads the example.

5.3.2 Cellular Networks Documentation

To view help documentation either click on “Technology Libraries” under documentation in the home screen or click the ‘Book’ link located next Cellular Networks in examples. The help documentation explains the following:

- Introduction
- Simulation GUI
- Featured Examples
- Cellular Networks Experiments in NetSim
- Note: Release on Unsupported Basis
- Latest FAQ available online

5.4 Advanced Routing

NetSim supports the following advanced routing protocols.

- Multicast Routing
  - Internet Group Management Protocol (IGMP)
  - Protocol Independent Multicast (PIM)
- Access Control Lists (ACLs)
- Virtual LAN (VLAN)
- Public IP and Network Address Translation (NAT)

To simulate the above-mentioned routing protocols, click on New Simulation and then Internetworks.

5.4.1 Advanced Routing Examples

To simulate the Examples for different types of Advanced Routing protocols

1. Go to the NetSim UI and click **Examples**.
   The Example Simulation pane appears at the right.
2. Click the **Advanced-Routing** example you wish to simulate. NetSim UI loads the example.

5.4.2 Advanced Routing Documentation

To view help documentation users can either click on “Technology Libraries” under documentation in the home screen or they can click the ‘Book’ link located next to Advanced Routing in examples. The help documentation explains the following:

- Simulation GUI
- Model Features
- Featured Examples
- Advanced Routing Experiments in NetSim
- Reference Documents
- Latest FAQ available online

5.5 MANETs

Mobile Ad-hoc Network (MANET) is an ad hoc network that can change locations and configure itself on the fly. Because MANETS are mobile, they use wireless connections to connect to various networks.

NetSim MANET library covers:

- L3 Routing Protocols – DSR, AODV, OLSR and ZRP

- MAC Layer – IEEE 802.11
- MANET using Bridge_Node (Wired) and Bridge_Node (Wireless)

To simulate MANET, click on New Simulation and then select Mobile Adhoc networks.

5.5.1 MANET Examples

To simulate MANET Examples:

1. Go to the NetSim UI and click **Examples**.
   The Example Simulation pane appears at the right.
2. Click the **Mobile-Adhoc-Networks** example you want to simulate. NetSim UI loads the example.

5.5.2 MANET Documentation

To view help documentation either click on "Technology Libraries" under documentation in the home screen or click the 'Book' link located next MANET Networks in examples. The help documentation explains the following:

- Introduction
- Simulation GUI
- Model Features
- Featured Examples
- Reference Documents
- Latest FAQ available online

5.6 Wireless Sensor Networks (WSN)

Wireless Sensor Network (WSN) is a group of spatially dispersed sensors that monitor and collect the physical conditions of the environment and transmit the data they collect to a central location. WSNs measure environmental conditions such as temperature, sound, pollution levels, humidity, wind, and so on.

WSN in NetSim is part of NetSim's IOT library and covers 802.15.4 MAC, PHY with MANET routing protocols.

To simulate WSN, click on New Simulation and then Wireless Sensor Networks.

5.6.1 Wireless Sensor Networks (WSN) Examples

To simulate Wireless Sensor Networks Examples:

1. Go to the NetSim UI and click **Examples**.
   The Example Simulation pane appears at the right.

2. Click the **IOT-WSN > Wireless-Sensor-Networks** example you want to simulate. NetSim UI loads the example.

5.6.2 WSN Library Documentation

To view help documentation either click on “Technology Libraries” under documentation in the home screen or click the ‘Book’ link located next to IOT-WSN examples. The help documentation explains the following:

- Introduction
- Simulation GUI
- Model Features
- Featured Examples
- IOT-WSN Experiments in NetSim
- Reference Documents
- Latest FAQ available online

5.7 Internet of Things

Internet of things (IoT) is a network of object such as vehicles, people, home appliances that contain electronics, software, actuators that are accessible from the public Internet. The objects are embedded with suitable technology and use IP addresses to interact and exchange data without manual assistance or intervention. The objects can also be remotely monitored and controlled.

In NetSim, IOT is modeled as a WSN that connects to the internet via a 6LowPAN Gateway. WSN for IoT uses the following protocols: AODV and RPL with IPv6 addressing at the L3 layer and 802.15.4 at the MAC & PHY layers. WSN sends data to the LowPAN Gateway which uses a Zigbee (802.15.4) interface and a WAN Interface. The Zigbee interface connects wirelessly to the WSN and the WAN interface connects to the Internet. Additionally, users can also simulate and analyze energy model for IoT.

To simulate IOT, click on New Simulation and then Internet of Things.

5.7.1 Internet of Things (IOT) Examples

To simulate IOT Examples:

1. Go to the NetSim UI and click **Examples**.
   The Example Simulation pane appears at the right.
2. Click the **IOT-WSN > Internet-of-Things** example you want to simulate. NetSim UI loads the example.

5.7.2 IOT Library Documentation

To view help documentation either click on “Technology Libraries” under documentation in the home screen or click the ‘Book’ link located next to IOT-WSN examples. The help documentation explains the following:

- Introduction
- Simulation GUI
- Model Features
- Featured Examples
- IOT-WSN Experiments in NetSim
- Reference Documents
- Latest FAQ available online

5.8 Software Defined Networks (SDN)

Software-defined networking (SDN) is an architecture that makes networks agile and flexible. SDN decouples the network control and forwarding functions. SDN allows you to program your network control and abstracts the physical infrastructure for applications and network services. This approach enables enterprises and service providers to respond quickly to the changing business requirements.

Unlike other technologies, and due to the way SDN works it is not available as a menu item under New Simulation. SDN can be configured when running Internetworks, MANET, IOT, WSN, Cognitive Radio, LTE or VANETs

5.8.1 Software Defined Networks (SDN) Examples

To simulate Software Defined Networks Examples:

1. Go to the NetSim UI and click **Examples**.
   The Example Simulation pane appears at the right.
2. Click the **Software-Defined-Networks** example you want to simulate. NetSim UI loads the example.

5.8.2 SDN Library Documentation

To view help documentation either click on “Technology Libraries” under documentation in the home screen or click the ‘Book’ link located next to Software Defined Network examples. The help documentation explains the following:

- About SDN
- SDN in NetSim

- Featured Examples
- Latest FAQ available online

5.9 Cognitive Radio

Cognitive Radio (CR) is an adaptive, intelligent radio and network technology that automatically detects available channels in a wireless spectrum and changes transmission parameters to enable higher levels of communication. Cognitive Radio can be programmed and configured dynamically to use the best wireless channels in its vicinity to avoid user interference and congestion.

NetSim Cognitive Radio module is based on the IEEE 802.22 standard. Additionally, you can connect a Cognitive Radio with Internetwork devices and run all the protocols supported in Internetworks.

To simulate Cognitive Radio, click on New Simulation and then Cognitive Radio Networks

5.9.1 Cognitive Radio Examples

To simulate Cognitive Radio Examples:

1. Go to the NetSim UI and click **Examples**.
   The Example Simulation pane appears at the right.
2. Click the **Cognitive-Radio** example you want to simulate. NetSim UI loads the example.

5.9.2 Cognitive Radio Library Documentation

To view help documentation either click on “Technology Libraries” under documentation in the home screen or click the ‘Book’ link located next to Cognitive Radio examples. The help documentation explains the following:

- Introduction
- Simulation GUI
- Model Features
- Featured Examples
- Cognitive Radio Networks Experiments in NetSim
- Reference Documents
- Latest FAQ available online

5.10 LTE/LTE-A

Long Term Evolution (LTE) is a standard for 4G wireless broadband technology that offers increased network capacity and speed to mobile device users. LTE offers higher peak data transfer rates -- up to 100 Mbps downstream and 30 Mbps upstream.

NetSim LTE Library support LTE/LTE-Advanced Networks.

Additionally, you can connect an LTE Network with Internetwork devices and run all the protocols supported in Internetworks.

To simulate LTE/LTE-A networks, click on New Simulation and then select LTE/LTE-A Networks.

5.10.1 LTE Examples

To simulate LTE Examples:

1. Go to the NetSim UI and click **Examples**.
   The Example Simulation pane appears at the right.
2. Click the **LTE and LTE-A** example you want to simulate. NetSim UI loads the example.

5.10.2 LTE Library Documentation

To view help documentation either click on “Technology Libraries” under documentation in the home screen or click the ‘Book’ link located next to LTE and LTE-A examples. The help documentation explains the following:

- Introduction
- Simulation GUI
- Model Features
- Featured Examples
- Reference Documents
- Latest FAQ available online

5.11 5G NR

NetSim 5G library features full stack, end-to-end, packet level simulation of 5G NR networks. The 5G library is based on Rel 15 / 3GPP 38.xxx series.

NetSim 5G library models all layers of the protocol stack as well as applications running over the network. This 5G library is architected to connect to the base component of NetSim (and in turn to other components) which provides functionalities such as TCP/IP stack protocols, Wireless protocols, Routing algorithms, Mobility, Output Metrics, Animation, Traces etc.

To simulate 5G NR networks, click on New Simulation and then 5G NR.

5.11.1 5G NR Examples

To simulate 5G NR Examples:

1. Go to the NetSim UI and click **Examples**.
   The Example Simulation pane appears at the right.
2. Click the **5G NR** example you want to simulate. NetSim UI loads the example.

5.11.2 5G NR Library Documentation

To view help documentation either click on “Technology Libraries” under documentation in the home screen or click the ‘Book’ link located next to 5G NR examples. The help documentation explains the following:

- Introduction
- Simulation GUI
- Model Features
- Featured Examples
- Omitted Features
- 5G NR Experiments in NetSim
- Reference Documents

5.12 VANETs

Vehicular Ad-Hoc Network (VANET) is a subset of a Mobile Ad-Hoc Network or MANET that allows vehicle-to-vehicle and vehicle-to-roadside communications to ensure safe transportation.

To simulate VANET click on New Simulation and then click on VANET.

5.12.1 VANET Examples

To simulate VANET Examples:

1. Go to the NetSim UI and click **Examples**.
   The Example Simulation pane appears at the right.
2. Click the **VANETs** example you want to simulate. NetSim UI loads the example.

5.12.2 VANET Library Documentation

To view help documentation either click on “Technology Libraries” under documentation in the home screen or click the ‘Book’ link located next to VANET examples. The help documentation explains the following:

- Introduction
- Simulation GUI

- Model Features
- Featured Examples
- Reference Documents
- Latest FAQ available online

5.13 Satellite Communication

NetSim satellite library models end-to-end, full stack, packet level communication between terrestrial nodes and Geostationary satellites.

The satellite can be thought of as a relay station. It operates on the bent-pipe (transparent star) principle, sending back to Earth what comes in, with only amplification and a shift from uplink to downlink frequency.

The Satellite MAC layer protocol supported in NetSim is TDMA for forward link and MF-TDMA for return link (based on the DVB S2 standards). The forward link is in the Ku band (12 – 18 GHz) while the return link is in the Ka band (26 – 40 GHz)

To simulate Satellite Communication networks, click on New Simulation and then click on Satellite Comm. Networks

5.13.1 Satellite Communication Examples

To simulate the Examples for different types of Internetworks

1. Go to the NetSim UI and click **Examples**.
   The Example Simulation pane appears at the right.
2. Click the **Satellite-Communication** example you wish to simulate. NetSim UI loads the example.

5.13.2 Satellite Communication Documentation

To view help documentation users can either click on “Technology Libraries” under documentation in the home screen or click the ‘Book’ link located next to Satellite-Communication in examples. The help documentation explains the following:

- Introduction
- Simulation GUI
- Model Features
- Featured Examples
- Reference Documents
- Latest FAQ available online

5.14 TDMA Radio Networks

NetSim TMDA Radio Network module uses TDMA/DTDMA in MAC/PHY along with MANET Routing protocols in Layer 3.

To simulate TDMA Radio Networks, click on **New Simulation → TDMA Radio Networks** and select TDMA/DTDMA in MAC/PHY layer of the devices.

Note: *TDMA Radio Network component is available only in NetSim pro version.*

5.14.1 TDMA Radio Network Examples

To simulate TDMA Radio Networks Examples:

1. Go to the NetSim UI and click **Examples**.
   The Example Simulation pane appears at the right.
2. Click the **TDMA Radio Networks** example you want to simulate. NetSim UI loads the example.

5.14.2 TDMA Radio Network Library Documentation

To view help documentation either click on "Technology Libraries" under documentation in the home screen or click the 'Book' link located next to TDMA Radio Networks examples. The help documentation explains the following:

- Introduction
- Simulation GUI
- Model Features
- Featured Examples
- Model Limitations
- Latest FAQ available online.

5.15 Network Emulator Add On

A network simulator mimics the behavior of networks but cannot connect to real networks. NetSim Emulator enables users to connect NetSim simulator to real hardware and interact with live applications.

- NetSim emulator is an IP based, data plane, flow-through emulator. This means:
- It can interact with IP based devices.
- It can emulate data place functionality and not control plane functionality.
- The source and destination for traffic should be external. A virtual device within NetSim cannot be a source or sink for traffic.

5.15.1 Emulation Library Documentation

To view help documentation either click on “Technology Libraries” under documentation in the home screen, The Emulation documentation explains the following:

- Introduction
- Emulation Set-up
- Model Features
- Featured Examples
- Trouble shooting
- Latest FAQ available online.

6 Applications (Network Traffic Generator)

Applications are the sources of traffic in the network. This traffic is modeled as individual packets. These packets flow from the source to the destination over the designed network. As it flows through the network, depending on the devices, link bandwidths and networking protocols, the packets would experience network effects such as delay, error, loss etc.

Applications are generally parameterized in terms of packet size, inter-packet arrival time, priority, transport protocol running below etc. Therefore, each application has its own distinctive traffic pattern and creates its own unique load on the network.

Different applications have differing levels of complexity. Some applications are used to quickly model basic requirements while in other cases parameters can be accurately modeled to carefully reproduce real world characteristics. For example, if the goal is to analyze protocol behavior, then using a simple CBR application (that generates a certain number of packets every second of a fixed size) would suffice.

NetSim allows users to model and simulate different types of applications.

1. CBR
2. Custom
3. COAP
4. Database
5. FTP
6. Email
7. HTTP
8. PEER_TO_PEER
9. Video
10. Voice
11. Sensor App
12. Erlang Call
13. BSM
14. Emulation (available only if Emulator Add-on is licensed)

To set up the application click on the application icon from the tool bar as shown below Figure 6-1.

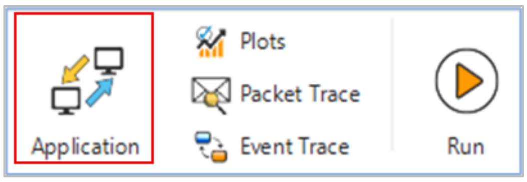
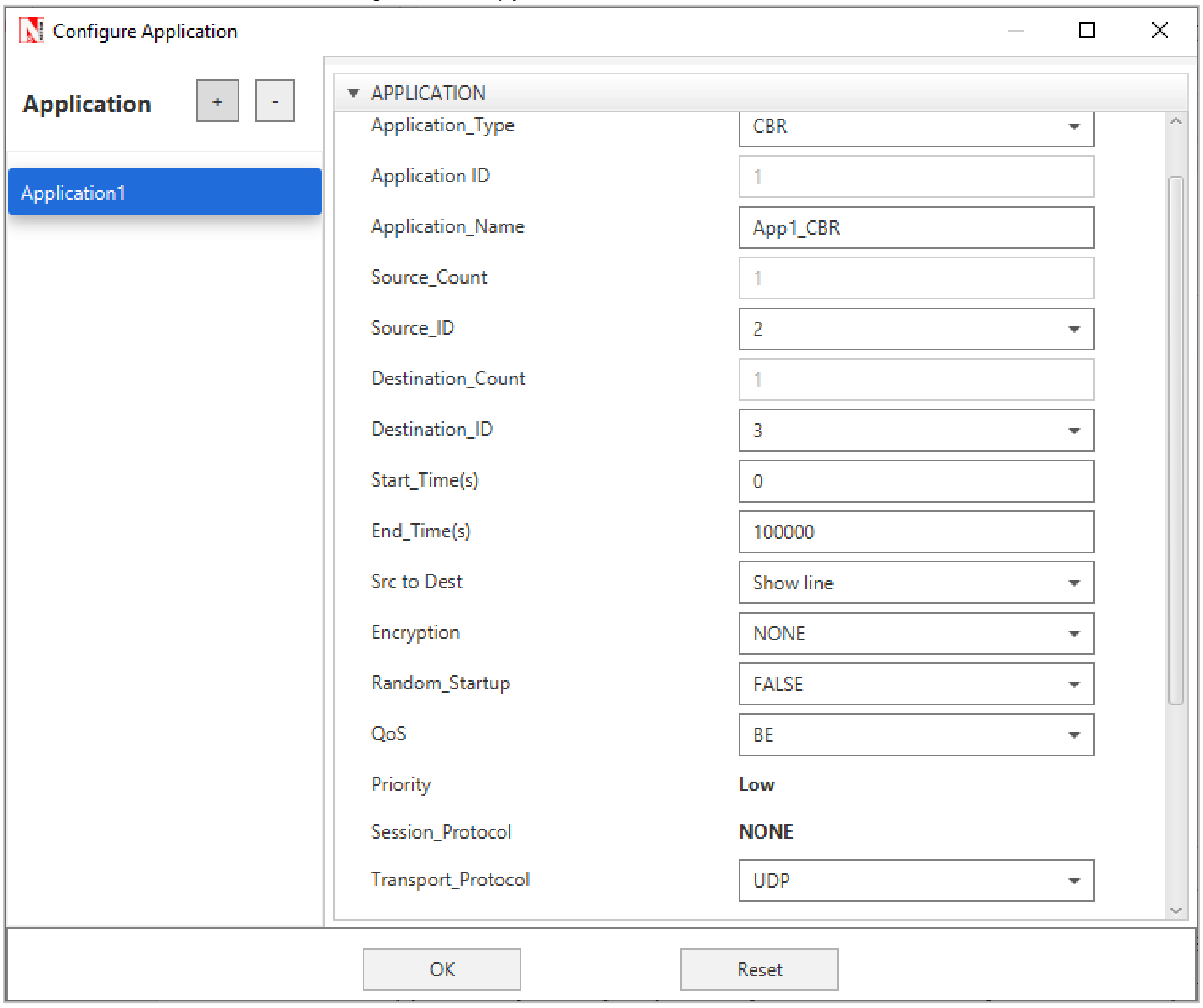


Figure 6-1: Application icon from the tool bar



Application_Type	CBR
Application ID	1
Application_Name	App1_CBR
Source_Count	1
Source_ID	2
Destination_Count	1
Destination_ID	3
Start_Time(s)	0
End_Time(s)	100000
Src to Dest	Show line
Encryption	NONE
Random_Startup	FALSE
QoS	BE
Priority	Low
Session_Protocol	NONE
Transport_Protocol	UDP

Figure 6-2: Application Configuration Window

This properties window allows you to model the application traffic. There may be more than one application you may require for your simulation study. You can add (or) delete one or more applications by clicking on the **"+"** or **"-"** symbols present on top left-hand side next to the Application.

These application models have default values set, for the various application properties, to model standard application behavior. Users can modify the parameters to model their own applications.

6.1 Common properties for all applications

Application Method: It specifies the type of Application method Unicast/Multicast/Broadcast.

Application Type: It specifies the type of application such as CBR, Custom, Peer to Peer, COAP, Email, HTTP, FTP, Voice, Video, Database, Erlang Call, Sensor App, BSM, and Emulation.

Application ID: This property represents the unique identification number of the application.

Application Name: It specifies the name of the application.

Source Count: This property represents number of sources for the application. Voice, Video, FTP, Database and Custom applications have only one source.

Source ID: This property represents the unique identification number of the source.

Destination Count: This property represents number of destinations for the application. Voice, Video, FTP, Database and Custom applications have only one destination.

Destination ID: This property represents the unique identification numbers of the destination.

- For **Unicast** Applications, users can select the ID of a device in the network as the Destination ID.
- For **Broadcast** Applications, the Destination ID, is set to '0'.
- For **Multicast** Applications, users can enter the number of multicast destinations in the Destination Count filed and specify the Device IDs of the destination devices separated by comma (",") in the Destination ID field. E.g., 6, 7, 8

Start time: This property represents the start time of the application in seconds.

End time: This property represents the end time of the application in seconds.

For example, if Start time is 1s and end time is 10s then application starts generating traffic at the 1st second and stops at the 10th second.

Encryption: Encrypts Application packet payload using algorithms such as AES, DES, XOR and TEA. The effect of encryption can be analyzed by enabling Wireshark option in either the source or the destination devices. Refer Section 8.7 on "Packet Capture and Analysis Using Wireshark" for further details.

In NetSim the packet size remains constant when encrypting using these algorithms. Therefore, using different encryption models will not have any impact on the network performance metrics that NetSim outputs. NetSim does not perform decryption of the packet at the receiver end since it does not have any impact on the performance metrics generated.

Random Startup: If random start up is set true, application will start at a random time between 0 and inter-arrival time. Having a random start-up time provides more realism to the model since all applications need not necessarily start at time = 0 in the real world.

QoS: NetSim provides QoS differentiation for the different types of applications through four defined scheduling service types, also called QoS classes as shown below Table 6-1.

QoS Class	Description	Priority
UGS - Unsolicited Grant Service	The UGS scheduling service type is designed to support real-time data streams consisting of fixed-size data packets issued at periodic intervals.	High
rtPS - Real-time Polling Service	The rtPS scheduling service type is designed to support real-time data streams consisting of variable-sized data packets that are issued at periodic intervals. This would be the case, for example, for MPEG (Moving Pictures Experts Group) video transmission.	Medium
ertPS - Extended real-time Polling Service	The ertPS is a scheduling mechanism that builds on the efficiency of both UGS and rtPS. UGS allocations are fixed in size, ertPS allocations are dynamic. The ertPS is suitable for variable rate real-time applications that have data rate and delay requirements.	Normal
nrtPS - Non-real-time Polling Service	The nrtPS is designed to support delay-tolerant data streams consisting of variable-size data packets for which a minimum data rate is required. The standard considers that this would be the case, for example, for an FTP transmission.	Low
BE - Best Effort	The BE service is designed to support data streams for which no minimum service guarantees are required and therefore may be handled on a best basis.	Low

Table 6-1: Different QoS classes with Description and Priority in NetSim

Priority: The priority is automatically set based on the QoS class set by the user. Depending on the scheduling algorithm the router would process packets, with different priorities, differently.

Session Protocol: Session Protocol is applicable only for applications that support RTP (Real-time Transport Protocol)

Transport Protocol: This parameter is newly added to the Applications window where by default it selects the Transport Layer Protocol (either TCP or UDP) depending on the application that is set by the user.

Note: *Users can also change the value of this parameter according to the transport protocol they intend to run a particular application.*

6.2 Application Types

Brief explanation of application types as shown below Table 6-2.

Application Type	Properties	Units	Description
CBR – Constant bit Rate	**Packet size** (Constant distribution) – It is the size of the packet	bytes	Packets of constant size are generated at constant inter arrival times. The generation times would be as follows: Packet 1: Application start time. Packet 2: Packet 1 + Interarrival Time Packet 3: Packet 2 + Interarrival Time .... Packet (n+1): Packet n + Interarrival Time Ends at Application end time.
	Inter Arrival Time (Constant distribution) – It is the gap between two successive packets	µs	
Custom	**Packet size** (Constant, Exponential, Uniform and Normal distribution) – It is the size of the packet	bytes	It is user defined application where the packet size and inter-arrival time can be set per user requirements.
	Inter Arrival Time (Constant, Exponential, Uniform and Normal distribution) – It is the time gap between two successive packets	µs	
Peer to Peer	**File size distribution** (Constant, Exponential distribution)	-	Peer-to-peer network does not have the notion of clients or servers but only equal peer nodes that simultaneously functioning as both "clients" and "servers" to the other nodes on the network. **Ex** – Torrent, LimeWire etc.
	Value – Size of the file	bytes	
	Piece size - Each file is divided into equal sized pieces. This property represents the size of each piece	bytes	
Email	**Email send/receive** Represents the rate at which emails are sent/receive	-	HTTP is a client-server configuration, not a source-destination confirmation. Both sides can send and receive. **Ex** – Outlook, Apple mail, Gmail etc.
	Duration (Constant, Exponential distribution) Time between two successive emails	Seconds	
	Email size (Constant, Exponential distribution) Size of an email	Bytes	
HTTP – Hyper Text Transfer Protocol	**Inter Arrival Time** (Constant, Exponential distribution). It is the time gap between two successive HTTP requests	seconds	HTTP is a request-response application; it uses a client-server configuration, not a source-destination confirmation. The client node sends a page request to the server, and the server responds with the pages (whose size is in bytes). HTTP utilizes TCP to transfer its information between computers (usually Web servers and clients). TCP should mandatorily be set as the transport layer protocol. In the application metrics as part of an HTTP application, users can see two
	Page size (Constant, Exponential distribution) It is the size of each page	bytes	
	Page count – Represents the number of pages	-	

			rows of metrics, one corresponding to requests from the client to the server and the other corresponding to the replies from the server to the client
COAP – Constrained Application Protocol	**Inter Arrival Time** (Constant, Exponential distribution) – It is the time b//w two successive COAP requests.	seconds	It is a specialized web transfer protocol for use with constrained nodes and constrained (e.g., low-power, lossy) networks and designed for M2M applications.
	Page size (Constant, Exponential distribution) – It is the size of each page.	bytes	
	Response time – It is the time taken by a device to generate response	ms	
	Multicast response – Represents the server responds to multicast response or not	-	
	NSTART – Limit the number of simultaneous outstanding interactions that a client maintains to a given server	-	
	DEFAULT_LEISURE – This setting is only relevant in multicast scenarios, outside the scope of the EST-coaps draft	-	
	PROBING_RATE: A parameter which specifies the rate of re-sending Non-confirmable messages.	-	
	Ack required – It represents whether the ack for the request/response to be sent or not	-	
FTP – File Transfer Protocol	**File size** (Constant, Exponential distribution) – It is the size of the file	bytes	It is a standard network protocol used for the transfer of files between a client and server **Note**: Devices must have TCP enabled as the transport layer protocol. **Ex** – FileZilla The generation times would be as follows: File 1: Application start time. File 2: File 1 + Interarrival Time File 3: File 2 + Interarrival Time .... File (n+1): File n + Interarrival Time Ends at Application end time The files are in-turn fragmented into packets during the simulation.
	File Inter Arrival Time size (Constant, Exponential distribution) – It is the gap between two files	seconds	

			Users can generate one file by setting $Application\ End\ Time < (Application\ Start\ Time + File\ Interarrival\ Time)$
Database	**Transaction size** (Constant, Exponential distribution) - It represents the size of each transaction	bytes	A database application is a computer program whose primary purpose is entering and retrieving information from a computerized database. **Ex** – MS Excel, MySQL etc.
	Transaction Inter Arrival Time (Constant, Exponential distribution) – It is the time gap between two successful transactions	µs	
Voice	**Packet size** (Constant, Exponential) – It is the size of the packet	bytes	It allows users to configure voice application between client and server. **Note** – Distribution is constant only for all codec types except custom. **Ex** – Skype
	Packet Inter Arrival Time (Constant, Exponential distribution) - It is the gap between two successful packets	µs	
	Service type – CBR, VBR	-	
	Suppression models available for VBR – Deterministic, Markov chain	-	
	Success ratio - Sets the ratio of the packets that are not silenced during VBR calls	%	
Video	**Model Type** – INDEPENDENT_GAUSSIAN, FIRST_ORDER_DEPENDENT_GAUSSIAN, H_261, H_263, MPEG1_Low_Res, MPEG1_High_Res, MPEG2_Low_Res, MPEG2_High_Res, BUFFERED_VIDEO_STREAMING_1, BUFFERED_VIDEO_STREAMING_2, BUFFERED_VIDEO_STREAMING_3, BUFFERED_VIDEO_STREAMING_4, BUFFERED_VIDEO_STREAMING_5, BUFFERED_VIDEO_STREAMING_6,	-	It allows users to configure video application between client and server. **Ex** – Skype
Erlang Call	**Packet size** (Constant, Exponential distribution) – It is the size of the packet	bytes	The erlang is a unit of traffic density in a telecommunications system. One erlang is the equivalent of one call **Note** – Distribution is constant only for all codec types except custom
	Packet Inter Arrival Time (Constant, Exponential distribution) - It is the gap between two successful packets	µs	

	Call duration (Constant, Exponential distribution) – It is the duration of each call	seconds	
	Call Inter Arrival Time (Constant, Exponential distribution) - It is the gap between two successful calls	seconds	
	Service type – VBR, CBR	-	
	Suppression model available for VBR – Deterministic, Markov chain	-	
	Success ratio - Sets the ratio of the packets that are not silenced during VBR calls	%	
Sensor App	**Packet size** (Constant, Uniform and Normal distribution) – It is the size of the packet	bytes	Used to create application between two sensors. **Ex** – Smart home, Smart water etc.
	Packet Inter Arrival Time (Constant, Uniform and Normal distribution) - It is the gap between two successful packets	µs	
BSM – Basic safety message	**Packet size** (Constant, Uniform and Normal distribution) – It is the size of the packet	bytes	The BSM Application class sends and receives the IEEE 1609 WAVE (Wireless Access in Vehicular Environments) Basic Safety Messages (BSMs). The BSM is a 20-byte packet that is generally broadcast from every vehicle at a nominal rate of 10 Hz. **Note** - Available only with VANET component. **Ex** – Traffic management
	Packet Inter Arrival Time (Constant, Uniform and Normal distribution) - It is the gap between two successful packets	µs	
Emulation	**Source Real IP** - Specifies the real IP Address of source device in Emulation	-	NetSim Emulation application enables users to connect NetSim simulator to real devices and interact with live applications. **Note** - Will be present only when Emulator Add-on is licensed
	Source Port - Specifies the Port no used for transmission by Application running in source device		
	Destination Real IP - Specifies the real IP Address of destination device in Emulation		
	Destination Port - Specifies the Port no used for reception by Application running in destination device		

Table 6-2: Brief explanation of application types

6.2.1 Voice Models

Codec stands for Coder-decoder. Codecs are devices which encode / decode digital data streams. 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.

Various voice codecs are available in NetSim to choose from. Packet size and Inter-arrival time value will vary depending on the codec value chosen.

- **G.711:** G.711 is a Pulse code modulation (PCM) of voice frequencies on a 64-kbps channel. G.711 uses a sampling rate of 8,000 samples per second. Non-uniform quantization with 8 bits is used to represent each sample, resulting in a 64-kbps bit rate.
- **G.729:** The G.729 speech codec uses audio data compression algorithm and compress the data at bit rates that vary between 6.4 and 12.4 kbps. Coding of speech at 8 kbps using conjugate-structure algebraic-code-excited linear prediction (CS-ACELP).
- **G.723:** G.723 is an ITU standard for speech codecs that uses the ADPCM method and provides good quality audio at 24 and 40 Kbps.
- **GSM-FR:** GSM–Full Rate (GSM-FR). The codec operates on each 20ms frame of speech signals sampled at 8 KHz and generates compressed bit-streams with an average bit-rate of 13 kbps. The codec uses Regular Pulse Excited – Long Term Prediction – Linear Predictive Coder (RPE-LTP) technique to compress speech.
- **GSM-EFR:** GSM enhanced full rate speech codec is a speech coding standard that was developed in order to improve the quite poor quality of GSM-Full Rate (FR) codec. Working at 12.2 kbps the EFR provides wire like quality in any noise free and background noise conditions. The EFR 12.2 kbps speech coding standard is compatible with the highest AMR mode (both are ACELP).
- **CELP**: Code excitation linear prediction. This model has a packet size of 18 B and inter-packet arrival time of 30 ms.
- **MELP**: Mixed excitation linear prediction. This model has a packet size of 8 B and an inter-packet arrival time of 22.5 ms.
- **CUSTOM**: It is similar to the CUSTOM application type explained in the table above.

6.2.2 Video Models

Introduction to Video Models

A digital video source (e.g., a movie stored on a computer disk, or a live video from a teleconference) essentially comprises a sequence of images, called frames. Each frame is a digital image comprising an array of pixels. A video source is characterized by the frame rate, expressed in frames per second (fps), and the number of pixels per frame (ppf). Typical values of the frame rate are 30, 50, and 60, whereas typical values of pixels per frame are in the range of 10^5 to 10^6.

Each pixel is encoded into a number of bits to represent the intensity and colour at that point in the image. If the “raw” bits for all the pixels are put together, the number of bits per frame become very large, and it becomes impractical to handle digital video on packet communication networks. Various intraframe and interframe coding techniques are used to reduce the number of bits that need to be sent for each frame.

6.2.2.1 Video Models in NetSim

In NetSim the following video models are available:

1. Basic bits per pixel models:
    - Independent Gaussian (termed as continuous normal VBR prior to v13)
    - First order dependent Gaussian (termed as Continuous State Autoregressive Markov prior to v13)
2. Video Codec Models: H.261, H.263, MPEG1 Low Res, MPEG1 High Res, MPEG2 Low Res, MPEG2 High Res
3. Buffered Video Streaming Model: Options BV1 through BV6

Note: *Quantized State Continuous Time Markov and Simple IPB Composite Model were discontinued from v13 onwards. If older config files with these entries are imported, then NetSim would automatically replace them with the Independent gaussian model with default parameters set.*

Basic bits per pixel models

Since, for a given video, the number of pixels per frame remains constant from frame to frame, it is convenient to consider the number of bits per pixel (obtained by dividing the number of bits generated for a frame by the number of pixels). This measure is also illustrative as it permits comparison between the number “raw” bits generated for each pixel, and the average number of bits per pixel after data compression. This results in a sequence of bits per pixel, say b_k, where k is the sequence number of the frame. For simulating a video stream in a communication network simulation, we need a statistical model of the sequence b_k, k=0,1, 2, ⋯ The bits emitted for each frame (obtained by multiplying the generated bits per pixel by the number of pixels per frame; we denote this sequence by B_k) are then packetized, thereby yielding a sequence of packets which are then emitted into the network for transport.

- **Independent Gaussian**: This simple model uses independent samples from a Gaussian (or Normal) distribution to generate the number of bits per pixel generated for each frame. In this simplest model, the number of bits per pixel is not necessarily an integer (hence, the term “continuous”), and the number of bits in successive frames are assumed to be statistically independent. Hence, this model has just two parameters

- μ (bits): the mean number of bits per pixel. NetSim range [0.01, 1.00]
- σ (bits): the standard deviation of the number of bits per pixel. NetSim range [0.01, 1.00]

The data generation rate (in bits per second) for the video application can be calculated by $\bar{B} = fps \times ppf \times \mu$, and $Var(B) = fps^2 \times ppf^2 \times \sigma^2$.

- **First order dependent Gaussian:** This model incorporates the autocorrelation between the frames. The number of bits generated for each frame is not necessarily an integer, but the number of bits per pixel in successive frames are modeled as a first-order autoregressive process driven by an independent Gaussian sequence (which is itself independent from frame to frame). Thus, starting with bits per pixel b_0, the successive b_k are generated as follows:

$$b_k = a\, b_{k-1} + b\, w_k$$

where

- a and b are parameters of the autoregressive process, and $w_k, k \geq 1,$ is an independent sequence of independent random variables. It is important to note that the b is positive and $|a| < 1$
- w_k is an independent Gaussian sequence with mean η and variance 1.

  With the above conditions on a and $b,$ the sequence b_k (and, therefore, the sequence B_k) has a steady state, with $\bar{b} = \left(\frac{b}{1-a}\right)\eta$ and $Var(b) = \left(\frac{b^2}{1-a^2}\right)$. The steady state variance and standard deviation of the sequence B_k can be obtained by recalling that $B_k = fps \times ppf \times b_k$; i.e., $\bar{B} = fps \times ppf \times \left(\frac{b}{1-a}\right)\eta,$ and $Var(B) = (fps \times ppf)^2 \times \left(\frac{b^2}{1-a^2}\right)$.

Video Codec Models

The list of video codec models and their parameters is provided in table below

Model Type	Packet Size (B) (Exponentially distributed)	Inter packet arrival time (ms)	Frames Per second	Average Bit Rate (Mbps)
H.261	160	20	50	0.064
H.263	160	20	50	0.064
MPEG1_Low_Res	2500	20	50	1
MPEG1_High_Res	7500	20	50	3
MPEG2_Low_Res	7500	20	50	3
MPEG2_High_Res	37500	20	50	15

Table 6-3: The input parameters used in NetSim for the various Video codecs and the resultant average bit rate

Buffered Video Streaming Models

The Buffered Video Streaming Models are based on IEEE 802.11-14/0571r12 standard. The video traffic from Video Source to Video Receiver is generated as follows.

- The packet size (bytes) is generated per the Weibull distribution which has the following formula.

$$f(x;\lambda,k) = \begin{cases} \frac{k}{\lambda}\left(\frac{x}{\lambda}\right)^{k-1} e^{-\left(\frac{x}{\lambda}\right)^{k}} & x \geq 0 \\ 0 & x < 0 \end{cases}$$

- The parameters to use are specified in Table 6-4.

Model_Type	Avg bit rate	λ	k	Frames per second (fps)
Buffered_Video_Streaming_1	2 Mbps	6950	0.8099	31
Buffered_Video_Streaming_2	4 Mbps	13900	0.8099	31
Buffered_Video_Streaming_3	6 Mbps	20850	0.8099	31
Buffered_Video_Streaming_4	8 Mbps	27800	0.8099	31
Buffered_Video_Streaming_5	10 Mbps	34750	0.8099	31
Buffered_Video_Streaming_6	15.6 Mbps	54210	0.8099	31

Table 6-4: Parameters for the Buffered video streaming models BV1 through BV6. $\boldsymbol{\lambda}$ and k parameters are the scale and shape parameters of the Weibull distribution for determining packet size. The average bit rate is the generation rate for the given values of $\boldsymbol{\lambda}, \boldsymbol{k}$ and $\boldsymbol{fps}$.

- The steps pertaining to adding TCP latency in the AP is not required since NetSim models Wi-Fi and TCP protocol operation.
- The standard does not provide the frame per second values. This was derived from the average bit rate, λ and k.

6.3 Network Traffic Generation Rate for Different Applications

This section explains how the traffic generation rate can be calculated for different types of applications:

CBR and Custom application

$$Generation\ Rate\ (Mbps) = \frac{Packet\ size\ (bytes) * 8}{InterArrivalTime(\mu s)}$$

Example: Packet size = 1460 Bytes and Inter arrival time = 20000 μs.

Generation rate (Mbps) = $\frac{1460 \times 8}{20000}$ = 0.584Mbps

Video

The Independent Gaussian model is the simplest of all video models in NetSim. It uses the normal distribution for the generation of bits per pixel. In this model, consecutive packet sizes are independent of each other. The generation rate for video application can be calculated by using the formula shown below:

$$Generation\ Rate\ (bits\ per\ second) = fps \times ppf \times bpp$$

where, fps = frames per second

ppf = pixel per frame

bpp (μ) = bits per pixel (mean)

Users can set the above-mentioned parameters in the Application Properties.

Example: Frames per second = 20, pixels per frame = 10000, bits per pixel = 0.52 then the generation rate would be

$$Generation\ Rate\ (Mps) = 20 \times 10000 \times 0.52 = 104000\ bits\ per\ second = 0.1040\ Mbps$$

Voice

$$Generation\ Rate\ (Mbps) = \frac{Packet\ size\ (bytes) \times 8}{InterArrivalTime(\mu s)}$$

Note that the distribution type is constant (deterministic) for all codec types except custom.

Email

$$Generation\ Rate\ (Mbps) = \frac{Email\ size\ (bytes) \times 8}{Duration(\mu s)}$$

Example: Email size = 20000bytes, Duration = 1s.

$$Generation\ rate\ (Mbps) = \frac{20000 * 8}{1000000} = 0.16\ Mbps$$

HTTP

$$Generation\ Rate\ (Mbps) = \frac{Page\ size\ (bytes) \times 8 \times\ Page\ count}{InterArrivalTime(\mu s)}$$

Example: Page size = 20000 Bytes, Page Count = 2, Inter arrival time = 3s

$$Generation\ rate\ (Mbps) = \frac{20000 \times 8 \times 2}{3000000} = 0.106\ Mbps$$

FTP

$$Generation\ Rate\ (Mbps) = \frac{File\ size\ (bytes) \times 8}{InterArrivalTime(s)}$$

Example: File size = 100000 Bytes, Inter arrival time = 5s

$$Generation\ rate\ (Mbps) = \frac{100000 \times 8}{5} = 0.16\ Mbps$$

Database

$$Generation\ Rate\ (Mbps) = \frac{Packet\ size\ (bytes) \times\ 8}{InterArrivalTime(\mu s)}$$

Example: Packet size = 10000 Bytes, Inter arrival time = 1000000µs

$$Generation\ rate\ (Mbps) = \frac{(10000 \times 8)}{1000000} = 0.08\ Mbps$$

BSM

$$Generation\ Rate\ (Mbps) = \frac{Packet\ size\ (bytes) \times\ 8}{InterArrivalTime(\mu s)}$$

Example: Packet size = 20Bytes and Inter arrival time = 1000000µs.

$$Generation\ rate\ (Mbps) = \frac{20 \times 8}{1000000} = 0.00016\ Mbps$$

Sensor

$$Generation\ Rate\ (Mbps) = \frac{Packet\ size\ (bytes) \times\ 8}{InterArrivalTime(\mu s)}$$

Example: Packet size = 50Bytes and Inter arrival time = 1000000µs.

$$Generation\ rate\ (Mbps) = \frac{50 \times 8}{1000000} = 0.0004\ Mbps$$

6.4 Priority and QoS of Applications

The various application traffic generated in NetSim have the following priority and QoS values as shown below.

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

Custom	2	Low	BE
Video	6	Medium	nRTPS
FTP	2	Low	BE
Database	2	Low	BE

Table 6-5: Priority and QoS of Applications

Note: *Priority of "Normal" has a Priority Value of 4 and "nRTPS" QoS Class. Ex: Video over TCP.*

Priority will have an impact on network performance when multiple applications with different priorities are configured in a network. These packets will be queued and dequeued from the router buffer based on the priority.

6.5 Capture real applications and simulate in NetSim

Users can capture packets from a live network using Wireshark. This can then be used as an input to NetSim as explained in Section 4 of the *Emulator* technology library user guide.

6.6 Modelling Poisson arrivals in NetSim

Any time you have events which occur individually at random moments, but which tend to occur at an average rate when viewed as a group, you have a Poisson process.

For example, we can estimate that a certain node generates 1200 packets per minute. These packets are randomly generated within a minute, but there are on average 1200 packets per minute. If 1200 packets generated per minute that, on average, one packet is generated every $\frac{60}{1200} = 0.05\,s$. So, let's define a variable $\lambda = \frac{1}{0.05} = 20$ and call it the *rate parameter*. The rate parameter λ is a measure of frequency: the average rate of events (packets) per unit of time (in this case, seconds).

Knowing this, we can ask questions like, what is the probability that a packet will be generated within the next second? What's the probability within the next 10 seconds? There's a well-known function to answer such questions. It's called the cumulative distribution function for the exponential distribution, and it looks like this:

$$F(x) = 1 - e^{-\lambda x}$$

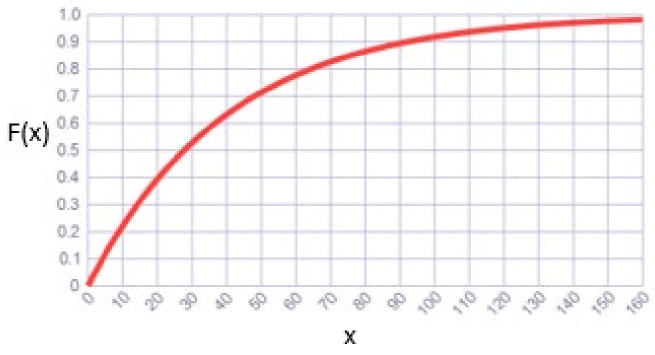


Figure 6-3: Plot for cumulative distribution function for the exponential distribution

Basically, the more time passes, the more likely it is that a packet is generated. The word "exponential", in this context, refers to exponential decay. As time passes, the probability of having *no* packets generated decays towards zero – and correspondingly, the probability of having at least one packet generated increases towards one.

Plugging in a few values, we find that:

- The probability of generating a packet within the next 0.05 seconds is F (0.05) ≈ 0.63
- The probability of generating a packet within 1 second is F (1) ≈ 0.999999998

In particular, note that after 0.05 seconds – the prescribed average time between packets – the probability is F (0.05) ≈ 0.63.

Generating Poisson arrivals in NetSim

We simply write a function to determine the exact amount of time until the next packet. This function should return random numbers, but not the uniform kind of random number produced by most generators. We want to generate random numbers in a way that follows our exponential distribution.

Given that the inverse of the exponential function is ln, it's easy to write this analytically, where R is the random value between 0 and 1:

$$T = -log_e \frac{1-R}{\lambda}$$

Where T is the time at which the next packet is generated.

The simple way of selecting this via the UI is to select exponential distribution for inter-arrival time inside application properties.

6.7 Application Configuration – Special Conditions

1. In a wired network with routers and switches OSPF, spanning tree etc. takes times to converge and hence it is a good practice to set the application start time greater than OSPF convergence

time. In general, the applications can start at 20s for smaller networks and should be increased as the size of the network grows.

2. If applications are started before OSPF convergence, then.
   - Packets generated before OSPF table convergence may be dropped at the gateway router.
   - The application may also stop if ICMP is enabled in the router.'
   - If TCP is enabled TCP may stop after the re-try limit is reached (since the SYN packets would not reach the destination)
3. For MANET networks the application start time should be a min of 5s, since that amount of time is required for convergence of OLSR/ZRP.

7 Running Simulation via Command Line Interface

7.1 Running NetSim via CLI

Advanced users can model their simulation via a configuration file (which can be created without the NetSim GUI) and run the simulation from command line. This is typically done in cases where very large networks are to be simulated (it takes too long to create it in the GUI), or to run a series of simulations automatically. The configuration file contains all required information to run the simulation including the network topology, devices, links, traffic, statistics, traces etc. To run Simulation in NetSim through command line interface (CLI), the following steps have to be followed.

Step 1: Note the Application Path

Application path is the current workspace location of the NetSim that you want to run. The default application path will be something like

"C:\Users\PC\Documents\NetSim\Workspaces\<Your default workspace>\bin_x64" for 64-bit for 64-bit. For more information on NetSim workspace, *Refer Section 4 "Workspaces and Experiments".*

Step 2: Note the IO Path

IO path (Input/output Path) is the path where the input and output files of an application is written. This is similar to the temp path of windows OS. For NetSim, the IO path can be got by **Start → Run → %temp%/NetSim**. Once you reach this folder, the user can notice that the path would be something like

"C:\Users\PC\AppData\Local\Temp\NetSim\std_13.1"

The IO path is the path where the ***Configuration.netsim*** (*NetSim Configuration file*) of the scenario, that will be simulated, should be present.App path and IO path can also be same, i.e., Configuration.netsim can be placed inside the app path (*if the app path has the write permission*). Otherwise, users can create a folder for IO path and Configuration.netsim can be placed inside that folder.

***Note:** Sample configuration.netsim files are available in the <NetSim installation Directory>/Docs/ Sample_Configurations folder of the NetSim install directory inside the respective protocol folder names.*

Step 3: Running NetSim through command line for Simulation.

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>

Note: *File path should be always added in the command prompt within double quotes. For example,*

>cd "C:\Users\PC\Documents\<Your default workspace>\bin_x64"

7.1.1 Running in CLI Mode when using floating licenses

For floating licenses, type the following in the command prompt.

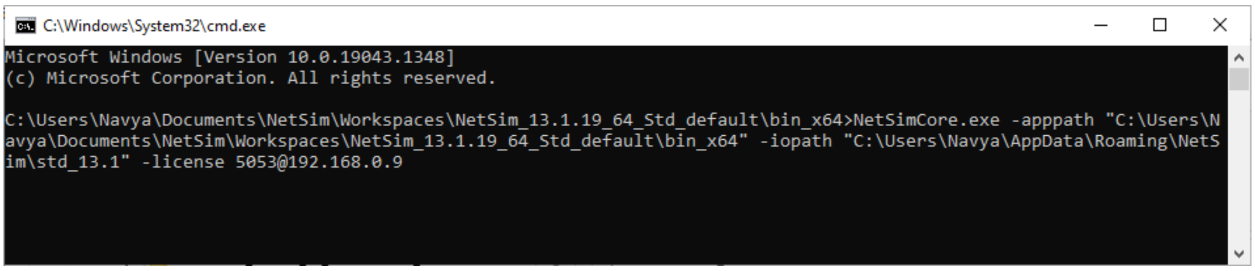
>NetSimCore.exe<space>-apppath<space><app path><space>-iopath<space><io path><space>-license<space>5053@<Server IP Address>

Where,

- **<app path>** contains all files of NetSim including NetSimCore.exe. Specifying the app path is optional. NetSim will take the current path as app path if not specified.
- **<iopath>** contains **Configuration.netsim.** (**Configuration.xsd** is available in the bin folder of NetSim's current workspace path ***C:\Users\PC\Documents\NetSim\Workspaces\<Your default workspace>\bin_x64*** for 64-bit
  Refer section 7.2.4 to know about configuration.xsd file.
- **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.

Note: *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 192.168.0.9.



```
Microsoft Windows [Version 10.0.19043.1348]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Navya\Documents\NetSim\Workspaces\NetSim_13.1.19_64_Std_default\bin_x64>NetSimCore.exe -apppath "C:\Users\N
avya\Documents\NetSim\Workspaces\NetSim_13.1.19_64_Std_default\bin_x64" -iopath "C:\Users\Navya\AppData\Roaming\NetS
im\std_13.1" -license 5053@192.168.0.9
```

Figure 7-1: Running NetSim through CLI mode for floating license

7.1.2 Running in CLI Mode when using node-locked or cloud licenses

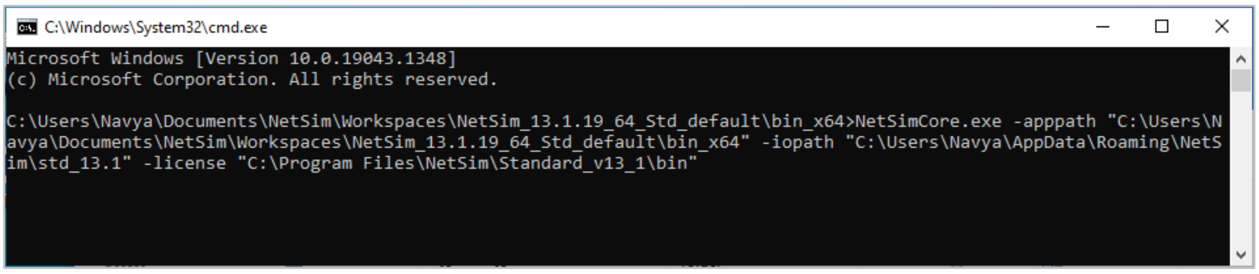
For cloud licenses and node-locked licenses, type the following in the command prompt

>NetSimCore.exe<space>-apppath<space><apppath><space>-iopath<space><io path><space>-license<space><license file path>

Where,

- **<app path>** contains all files of NetSim including NetSimCore.exe
- **<iopath>** contains Configuration.netsim and Configuration.xsd
- **<license file path>** path where the license file is present. This is generally the <NetSim_Installation_Directory>/bin folder.
  For E.g. C:\Program Files\NetSim\Standard_v13_1\bin

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



```
Microsoft Windows [Version 10.0.19043.1348]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Navya\Documents\NetSim\Workspaces\NetSim_13.1.19_64_Std_default\bin_x64>NetSimCore.exe -apppath "C:\Users\N
avya\Documents\NetSim\Workspaces\NetSim_13.1.19_64_Std_default\bin_x64" -iopath "C:\Users\Navya\AppData\Roaming\NetS
im\std_13.1" -license "C:\Program Files\NetSim\Standard_v13_1\bin"
```

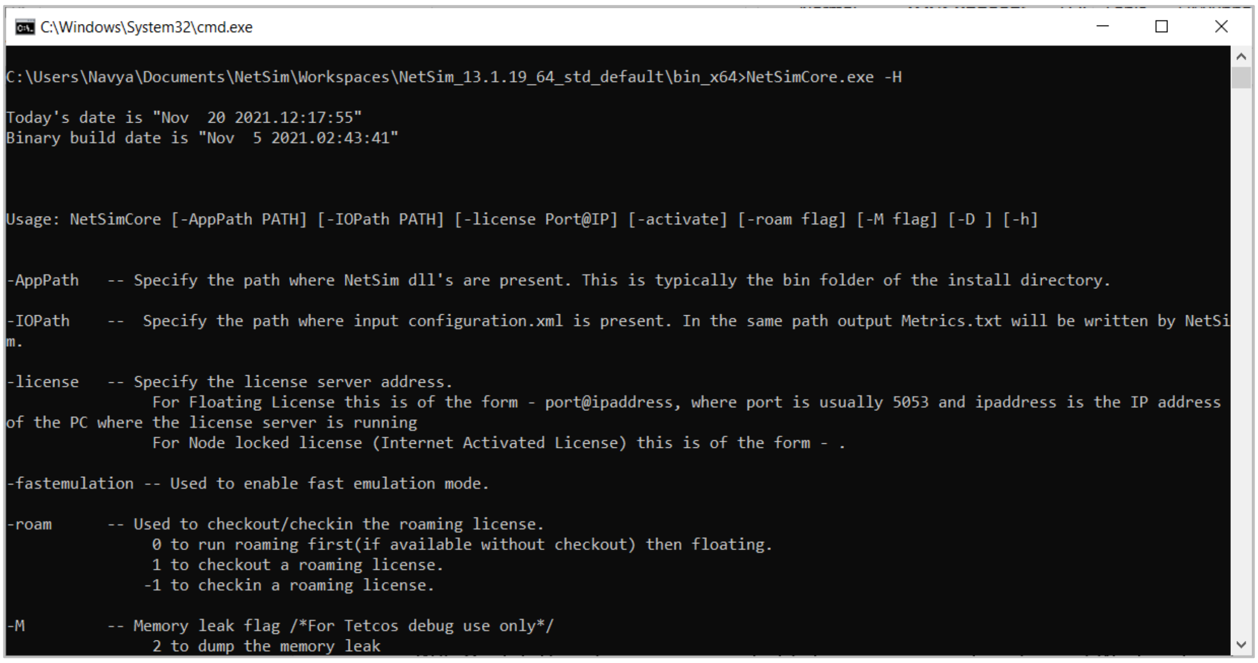
Figure 7-2: Running NetSim through CLI mode for Cloud and Node lock licenses

Once simulation is complete the text files that are requested by the end user in Configuration.netsim will be written in the **<iopath>**.

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

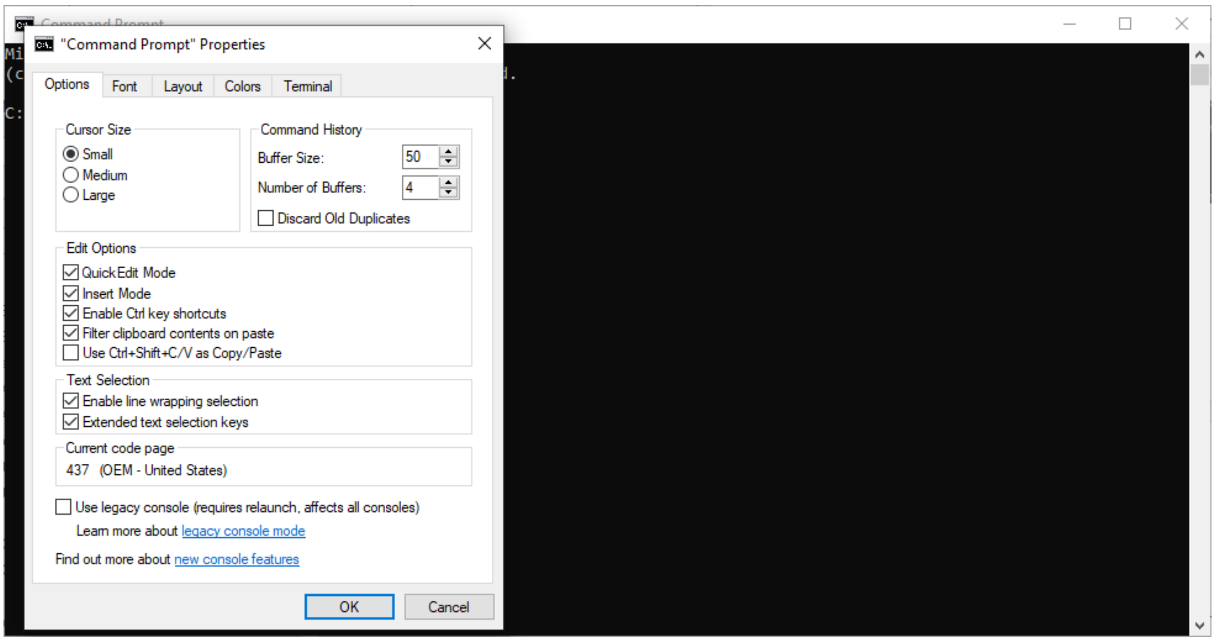


```
C:\Windows\System32\cmd.exe

C:\Users\Navya\Documents\NetSim\Workspaces\NetSim_13.1.19_64_std_default\bin_x64>NetSimCore.exe -H

Today's date is "Nov  20 2021.12:17:55"
Binary build date is "Nov  5 2021.02:43:41"


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 NetSi
m.

-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 license server is running
                For Node locked license (Internet Activated License) this is of the form - .

-fastemulation -- Used to enable fast emulation mode.

-roam      -- Used to checkout/checkin the roaming license.
                0 to run roaming first(if available without checkout) then floating.
                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
```

Figure 7-3: More Options available to run NetSim via CLI

7.1.3 Quick edit for copy pastes in CLI mode

With Quick Edit mode, you can copy text between a command window and Windows-based programs, and you can also paste text into a command window by using a right-click operation. To use Quick edit mode in command prompt users can run the command prompt → **Right Click** the icon in the upper-left corner of the Command Prompt window, and then Click **Properties** →In the options, enable **Quick Edit mode** → and click on **OK**.



"Command Prompt" Properties

Options | Font | Layout | Colors | Terminal

Cursor Size: [✓ Small] Medium Large

Command History
Buffer Size: 50
Number of Buffers: 4
☐ Discard Old Duplicates

Edit Options
☑ QuickEdit Mode
☑ Insert Mode
☑ Enable Ctrl key shortcuts
☑ Filter clipboard contents on paste
☐ Use Ctrl+Shift+C/V as Copy/Paste

Text Selection
☑ Enable line wrapping selection
☑ Extended text selection keys

Current code page
437 (OEM - United States)

☐ Use legacy console (requires relaunch, affects all consoles)
Learn more about legacy console mode

Find out more about new console features

OK | Cancel

Figure 7-4: Quick edit mode via CLI Running

7.2 Understanding the Configuration.netsim file

When a scenario is created in the GUI, NetSim's UI code write all the details about the devices used and its properties, the links used and their properties, the properties of the environment being used, etc. in the file ***Configuration.netsim***

The simulation engine that contains DLLs and NetSimCore.exe reads this Configuration.netsim, executes the simulation and writes output metrics files. The GUI then displays the metrics based on the text files written by the backend.

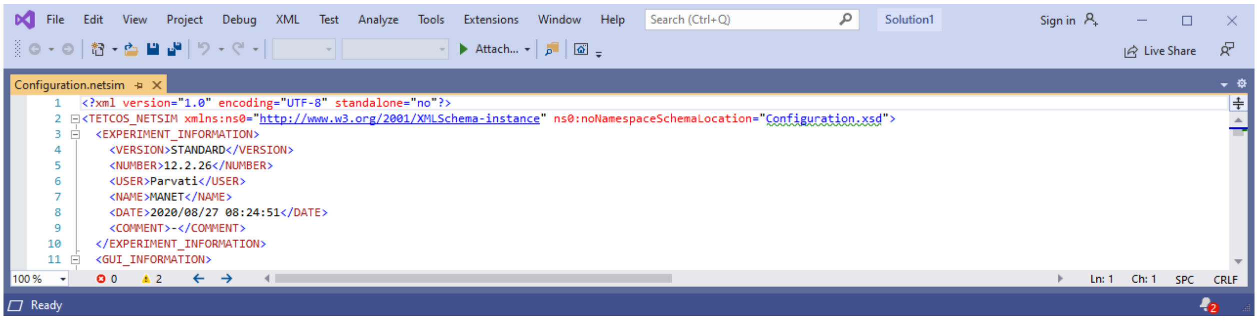
In order to run NetSim through command line (CLI), the user must create the Configuration.netsim file furnishing all the details about the devices, links and the environment of the desired scenario.

7.2.1 How to use Visual Studio to edit the Configuration file?

In Visual Studio, 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.

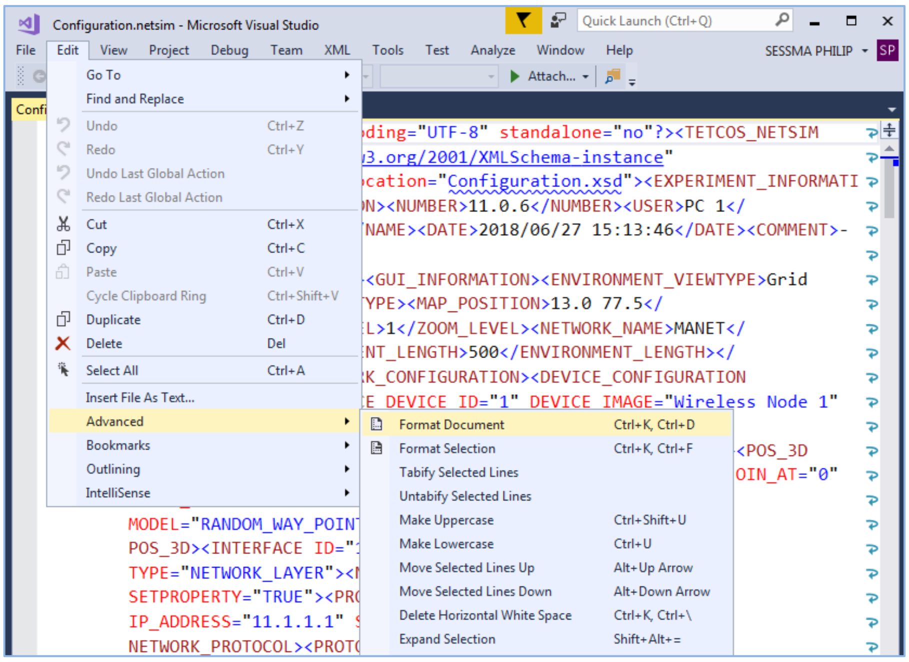
The following screenshot displays the Configuration.netsim which is opened through the Visual Studio as shown below Figure 7-5.



```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<TETCOS_NETSIM xmlns:ns0="http://www.w3.org/2001/XMLSchema-instance" ns0:noNamespaceSchemaLocation="Configuration.xsd">
  <EXPERIMENT_INFORMATION>
    <VERSION>STANDARD</VERSION>
    <NUMBER>12.2.26</NUMBER>
    <USER>Parvati</USER>
    <NAME>MANET</NAME>
    <DATE>2020/08/27 08:24:51</DATE>
    <COMMENT>-</COMMENT>
  </EXPERIMENT_INFORMATION>
  <GUI_INFORMATION>
```

Figure 7-5: Open Configuration.netsim file via Visual Studio

To reformat click on edit→Advanced→Format Document.



```
ding="UTF-8" standalone="no"?><TETCOS_NETSIM
3.org/2001/XMLSchema-instance"
cation="Configuration.xsd"><EXPERIMENT_INFORMATI
N><NUMBER>11.0.6</NUMBER><USER>PC 1</
NAME><DATE>2018/06/27 15:13:46</DATE><COMMENT>-

<GUI_INFORMATION><ENVIRONMENT_VIEWTYPE>Grid
YPE><MAP_POSITION>13.0 77.5</
L>1</ZOOM_LEVEL><NETWORK_NAME>MANET</
NT_LENGTH>500</ENVIRONMENT_LENGTH></
K_CONFIGURATION><DEVICE_CONFIGURATION
E DEVICE ID="1" DEVICE IMAGE="Wireless Node 1"
                                              <POS_3D
                                              OIN_AT="0"
MODEL="RANDOM_WAY_POIN
POS_3D><INTERFACE ID="
TYPE="NETWORK_LAYER"><
SETPROPERTY="TRUE"><PR
IP_ADDRESS="11.1.1.1"
NETWORK_PROTOCOL><PROT
```

Figure 7-6: Reformat the Configuration.netsim file

7.2.2 Sections of Configuration file

These are the different sections in Configuration.netsim:

- EXPERIMENT_INFORMATION
- GUI_INFORMATION
- NETWORK_CONFIGURATION
- SIMULATION_PARAMETER
- PROTOCOL_CONFIGURATION
- STATISTICS_COLLECTION

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.

GUI_INFORMATION:

This section contains the GUI information like the environment length, view type etc. and the network name which is desired to be run.

NETWORK_CONFIGURATION:

This section is used to configure the devices and the links of the desired network at each layer of the TCP/IP stack. It consists of DEVICE_CONFIGURATION, CONNECTION and APPLICATION_CONFIGURATION. DEVICE_CONFIGURATION configures the devices in the desired network while the CONNECTION configures the links in the desired network and APPLICATION configures the Applications.

SIMULATION_PARAMETER:

Simulation time and seed values are described in this section.

PROTOCOL_CONFIGURATION:

IPV4 and static ARP are enabled or disabled in this section. The text files illustrating the static routing and static ARP can be obtained by enabling the corresponding tags in the Configuration.netsim.

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.

7.2.3 Sample Configuration file

Sample “Configuration.netsim” file will be installed in user system along with the software at <NetSim installed Path>\Docs\ Sample_Configuration\ <Network Technology>.User can open and edit these files using Visual Studio 2015/2017/2019 or any XML editor. The purpose of providing the sample “Configuration.netsim” file is to assist the user in writing a network scenario manually by analyzing the format for that specific network technology.

7.2.4 Configuration.xsd file

Configuration.xsd is an XML schema Definition file which is present in the bin folder of NetSim’s current workspace path

<C:\Users\PC\Documents\NetSim\Workspaces\<Your default workspace>\bin_x64> for 64-bit.

Configuration.xsd file can be placed inside the <iopath> along with the configuration.netsim file to verify the contents in the configuration.netsim file. This file checks and validates the structure and vocabulary of a configuration.netsim document against the grammatical rules of the appropriate XML language.

It is not mandatory to place the configuration.xsd file along with the Configuration.netsim file in the iopath. But if it is done, then it will be easier to check & validate changes that are done to the Configuration.netsim file.

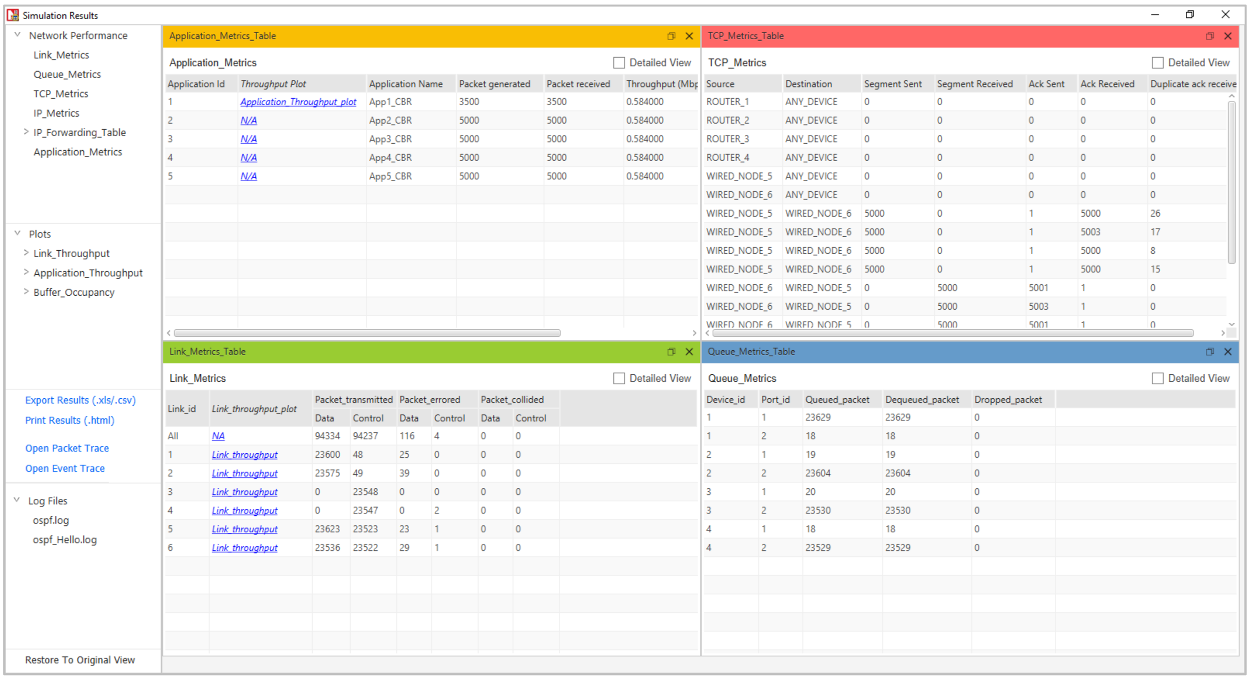
8 Outputs: Results, Plots and Data Files

8.1 Result Window and Plots Windows

The results of a simulation run are presented in a unified dashboard for convenient analysis. Graphics plots comprises of application throughputs, link throughputs, buffer occupancy and TCP congestion windows. The tabular presentation includes end-to-end delays, jitter, errors, packets generated / received / collided, route tables, TCP Acks, retransmissions etc.

Results are organized per interface, per device, per application and per link. In addition, summary metrics are aggregated and presented system-wide (network-level). Information in the trace files contain individual packet flow and individual event execution. Protocol log files records a myriad of information pertaining to protocol operation necessary for in-depth analysis and debugging.

The results can be exported as a .csv file and opened in a spread sheet software like Excel. Results can also be exported in .html format and opened in a browser.



Application_Metrics_Table

Application_Metrics

Application Id	Throughput Plot	Application Name	Packet generated	Packet received	Throughput (Mbp
1	Application_Throughput_plot	App1_CBR	3500	3500	0.584000
2	N/A	App2_CBR	5000	5000	0.584000
3	N/A	App3_CBR	5000	5000	0.584000
4	N/A	App4_CBR	5000	5000	0.584000
5	N/A	App5_CBR	5000	5000	0.584000

TCP_Metrics_Table

TCP_Metrics

Source	Destination	Segment Sent	Segment Received	Ack Sent	Ack Received	Duplicate ack receive
ROUTER_1	ANY_DEVICE	0	0	0	0	0
ROUTER_2	ANY_DEVICE	0	0	0	0	0
ROUTER_3	ANY_DEVICE	0	0	0	0	0
ROUTER_4	ANY_DEVICE	0	0	0	0	0
WIRED_NODE_5	ANY_DEVICE	0	0	0	0	0
WIRED_NODE_6	ANY_DEVICE	0	0	0	0	0
WIRED_NODE_5	WIRED_NODE_6	5000	0	1	5000	26
WIRED_NODE_5	WIRED_NODE_6	5000	0	1	5003	17
WIRED_NODE_5	WIRED_NODE_6	5000	0	1	5000	8
WIRED_NODE_5	WIRED_NODE_6	5000	0	1	5000	15
WIRED_NODE_6	WIRED_NODE_5	0	5000	5001	1	0
WIRED_NODE_6	WIRED_NODE_5	0	5000	5003	1	0
WIRED_NODE_6	WIRED_NODE_5	0	5000	5001	1	0

Link_Metrics_Table

Link_Metrics

Link_id	Link_throughput_plot	Packet_transmitted		Packet_errored		Packet_collided	
		Data	Control	Data	Control	Data	Control
All	NA	94334	94237	116	4	0	0
1	Link_throughput	23600	48	25	0	0	0
2	Link_throughput	23575	49	39	0	0	0
3	Link_throughput	0	23548	0	0	0	0
4	Link_throughput	0	23547	0	2	0	0
5	Link_throughput	23623	23523	23	1	0	0
6	Link_throughput	23536	23522	29	1	0	0

Queue_Metrics_Table

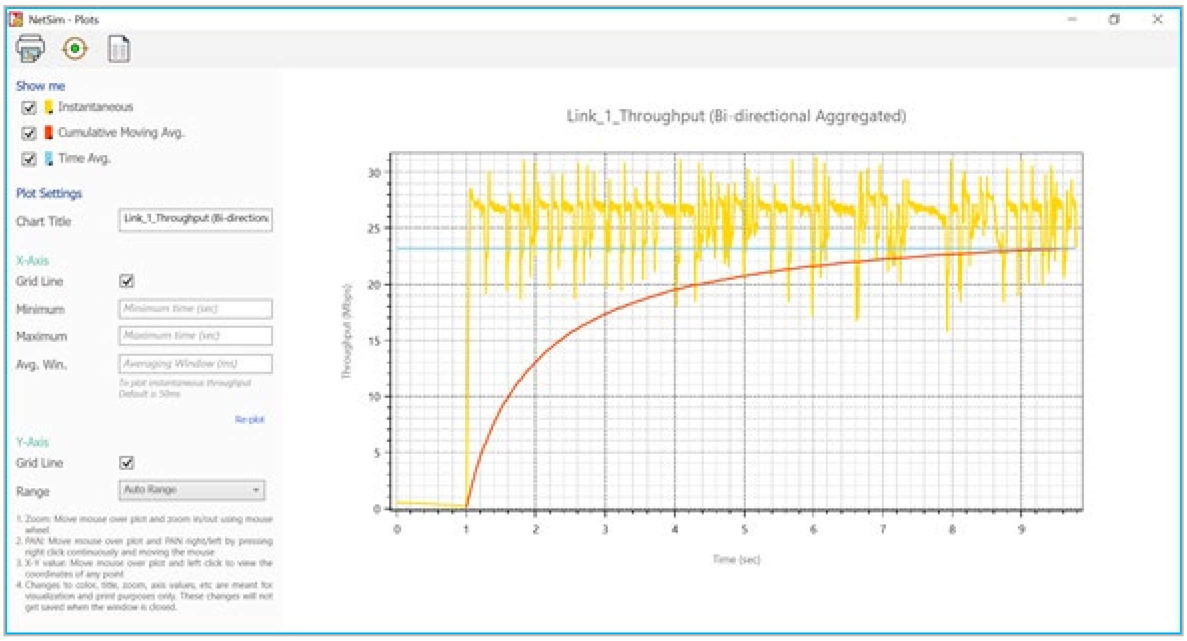
Queue_Metrics

Device_id	Port_id	Queued_packet	Dequeued_packet	Dropped_packet
1	1	23629	23629	0
1	2	18	18	0
2	1	19	19	0
2	2	23604	23604	0
3	1	20	20	0
3	2	23530	23530	0
4	1	18	18	0
4	2	23529	23529	0

Figure 8-1: Result Window

8.1.1 Application and Link Throughput Plots

If plots are enabled, NetSim plots Instantaneous (50 ms averaging window) Throughput, Cumulative moving average Throughput and Time Average throughput for each link and each application.


NetSim - Plots

Show me
- ☑ 🟨 Instantaneous
- ☑ 🟥 Cumulative Moving Avg.
- ☑ 🟦 Time Avg.

Plot Settings

Chart Title	Link_1_Throughput (Bi-direction

X-Axis

Grid Line	☑
Minimum	Minimum time (sec)
Maximum	Maximum time (sec)
Avg. Win.	Averaging Window (ms)
	To plot instantaneous throughput Default is 50ms

Re-plot

Y-Axis

Grid Line	☑
Range	Auto Range

1. Zoom: Move mouse over plot and zoom in/out using mouse wheel
2. PAN: Move mouse over plot and PAN right/left by pressing right click continuously and moving the mouse
3. X-Y value: Move mouse over plot and left click to view the coordinates of any point
4. Changes to color, title, zoom, axis values, etc are meant for visualization and print purposes only. These changes will not get saved when the window is closed.

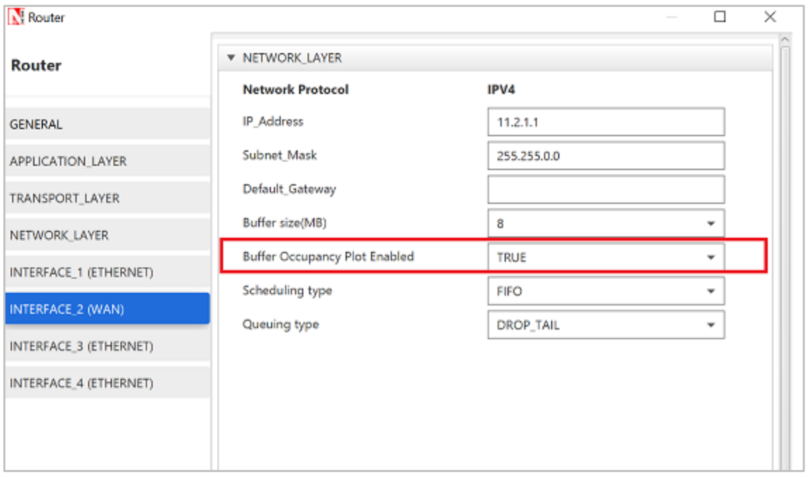
Figure 8-2: Link Throughput plot

Guidance on Zooming, Panning, and obtaining the XY co-ordinate values are provided on the bottom left of the window.

The 're-plot' option can be used to change the X-value, Min and Max, and to change the averaging window for plotting the instantaneous throughput.

8.1.2 Buffer Occupancy Plot

The buffer occupancy over time can be plotted by setting *Buffer Occupancy Plot Enabled* to True. This parameter is available wherever there are buffers in NetSim such as in Router – WAN Port – Network Layer as shown Figure 8-3.


Router

Router

- GENERAL
- APPLICATION_LAYER
- TRANSPORT_LAYER
- NETWORK_LAYER
- INTERFACE_1 (ETHERNET)
- INTERFACE_2 (WAN)
- INTERFACE_3 (ETHERNET)
- INTERFACE_4 (ETHERNET)

▾ NETWORK_LAYER

Network Protocol	**IPV4**
IP_Address	11.2.1.1
Subnet_Mask	255.255.0.0
Default_Gateway	
Buffer size(MB)	8
Buffer Occupancy Plot Enabled	TRUE
Scheduling type	FIFO
Queuing type	DROP_TAIL

Figure 8-3: Buffer Occupancy Plot set to True in WAN Port – Network Layer

Upon simulation the buffer occupancy plot can be opened from the Results Window and would look as shown below in Figure 8-4.

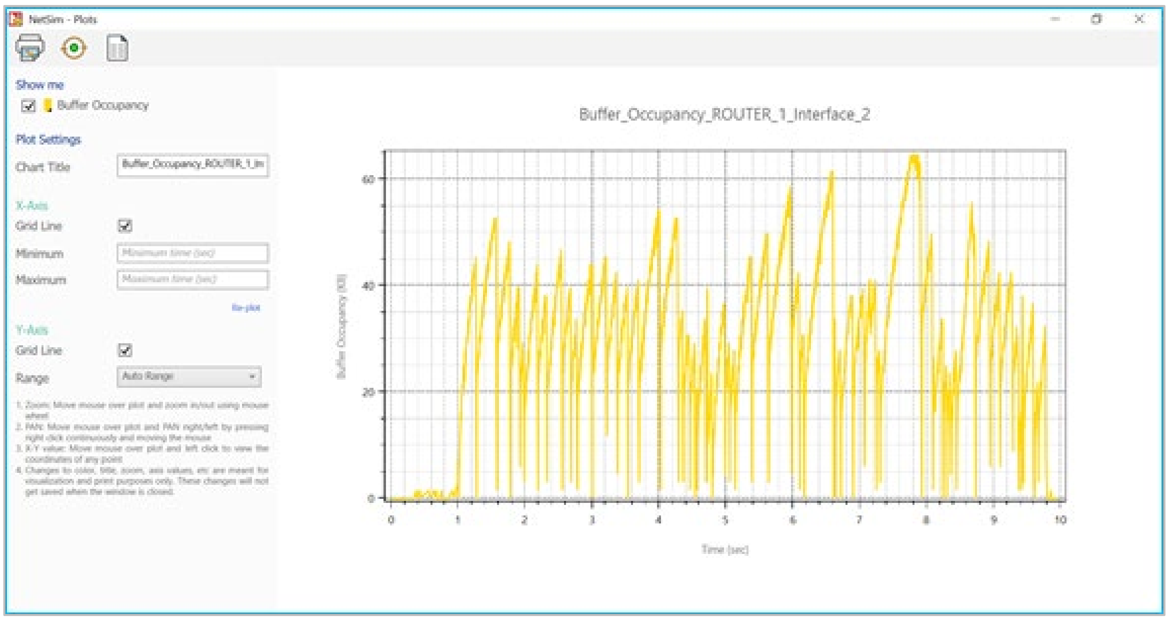
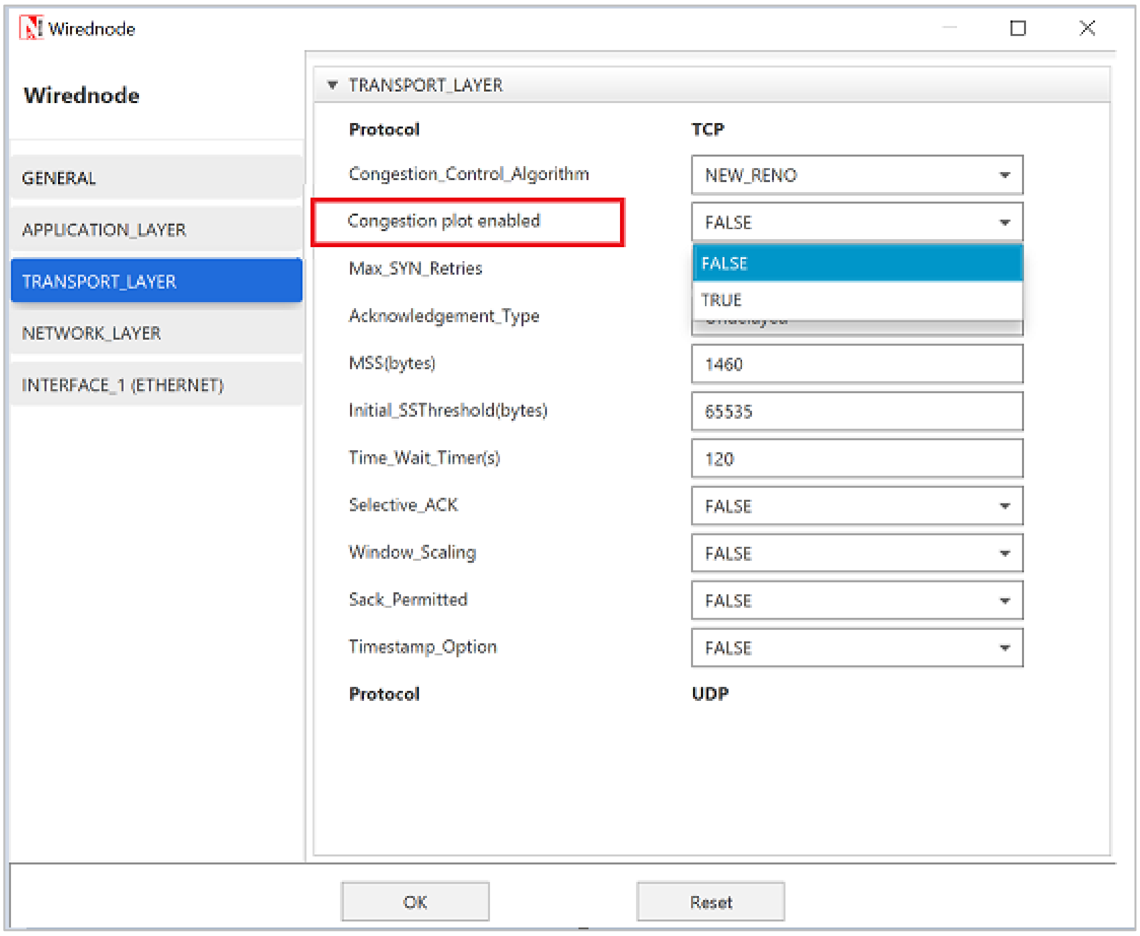


Figure 8-4: Buffer occupancy plot

8.1.3 TCP Congestion Window Plot

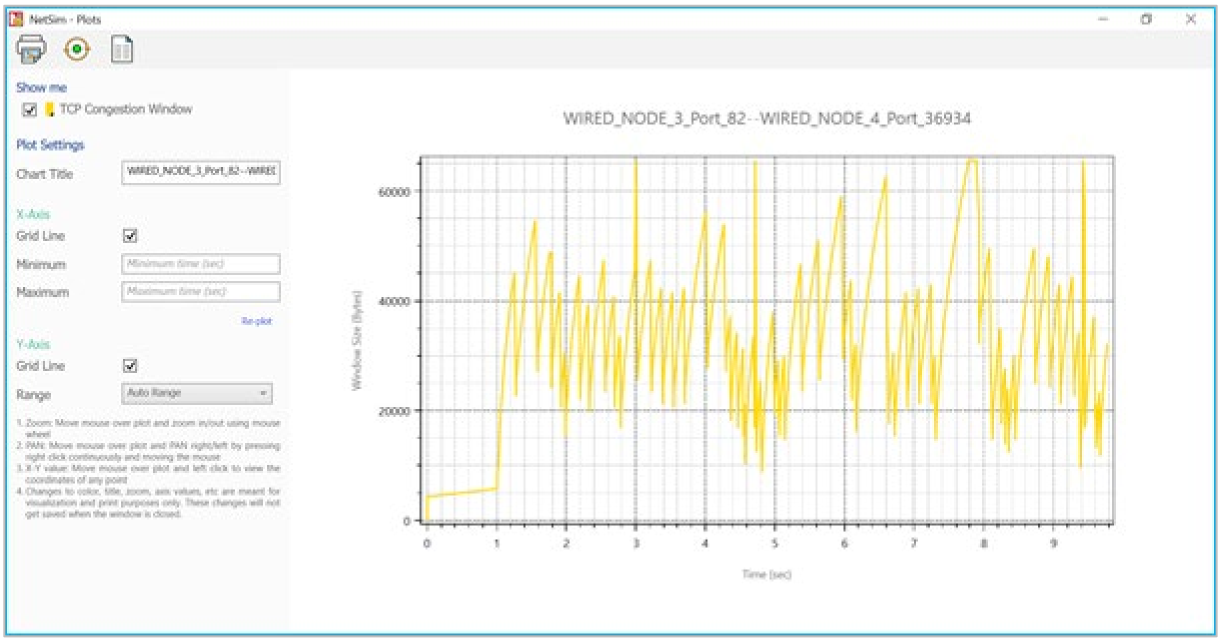
The TCP Congestion window over time can be plotted by setting *Congestion Window Plot Enabled* to True. This parameter is available in the end nodes where TCP has been enabled, for example Wired Node – Transport Layer.



Protocol	**TCP**
Congestion_Control_Algorithm	NEW_RENO
Congestion plot enabled	FALSE
Max_SYN_Retries	
Acknowledgement_Type	
MSS(bytes)	1460
Initial_SSThreshold(bytes)	65535
Time_Wait_Timer(s)	120
Selective_ACK	FALSE
Window_Scaling	FALSE
Sack_Permitted	FALSE
Timestamp_Option	FALSE
Protocol	**UDP**

Figure 8-5: TCP Congestion Plot Enabled set to True in Transport Layer

Upon simulation the TCP congestion window plot can be opened from the Results Window and would look like what is shown below in Figure 8-6.


NetSim - Plots

Show me

☑ TCP Congestion Window

Plot Settings

Chart Title: WIRED_NODE_3_Port_82--WIRED

X-Axis

Grid Line ☑

Minimum: Minimum time (sec)

Maximum: Maximum time (sec)

Re-plot

Y-Axis

Grid Line ☑

Range: Auto Range

1. Zoom: Move mouse over plot and zoom in/out using mouse wheel
2. PAN: Move mouse over plot and PAN right/left by pressing right click continuously and moving the mouse
3. X-Y value: Move mouse over plot and left click to view the coordinates of any point
4. Changes to color, title, zoom, axis values, etc are meant for visualization and print purposes only. These changes will not get saved when the window is closed.

Figure 8-6: TCP Congestion window plot

The down sampling algorithm in NetSim's plot engine leads to approximations while plotting, especially in TCP. To obtain a very precise TCP congestion plot window please enable Wireshark interfacing and view the TCP congestion window in Wireshark.

8.1.4 Notes on plots

1. To accelerate plotting, NetSim uses down-sampling/decimation to choose n points from N for plotting. NetSim generates n random numbers from a discrete uniform $U\,(\,0,\; N-1)$ distribution and plots for these n points.
2. To get a more precise plot users can select the min and max values (time) and replot.
3. The link throughput is calculated as the sum of throughputs in both directions for a full duplex link.
4. Application throughput is plotted till the last packet reaches or till end of simulation time, whichever is earlier.
5. Cumulative Moving Average: This is the average of the metric up until the current time and is defined as

$$\bar{\theta}(t) = \frac{1}{t}\int_{0}^{t} r(u)du$$

6. Instantaneous Throughput (Averaging Window) is defined as

$$\left(\frac{Bits\ transmitted\ in\ Window\ t}{Window\ size\ \mathrm{t}}\right)$$

Each value represents one point in the plot and the calculations are done every t seconds.

The source data file for the plot has the 1st column as time stamp and the 2nd column as Bytes transmitted. NetSim computes and plots the instantaneous throughput from this. For example, if the averaging window were 25ms, and if the source data entries were per the table below

Time (ms)	Bytes
1	5
20	15
40	5
60	10
74	5

Then the throughput in Kbps (since its *milli* seconds is in the denominator) would be

First 25ms (0 to 24) $= \frac{(5+15)\times 8}{25} \quad = \frac{20\times 8}{25} \quad = \frac{160}{25}$

Next 25ms (25 to 49) $= \frac{5\times 8}{25} \quad = \frac{40}{25}$

Next 25ms (50 to 74) $= \frac{(10+5)\times 8}{25} \quad = \frac{15\times 8}{25} \quad = \frac{120}{25}$

8.1.5 Link metrics

Here users can view the values of the metrics obtained based on the overall network and also displays the values of the metrics pertaining to each link.

- **Link ID:** It is the unique Id for the link.
- **Link Throughput Graph:** Plots throughput vs. Simulation time

Formula:

$$Link\ Throughput\ (Mbps) = \frac{Total\ bytes\ transmitted\ over\ the\ link\ \times\ 8}{Simulation\ Time\ (\mu s)}$$

Total Bytes transmitted is counted for successful data packets and control packets.

The calculation is based on the packet size (bytes) at the PHY layer, which would include app layer payload plus the overheads of all layers. Error and collision packets are not included in this calculation and only successful packets are counted for calculation of this metric.

- **Packets Transmitted:** It is the total number of packets transmitted in the link. Along with data packets, it includes protocol control packets like ARP Request, ARP Reply, TCP_ACK, TCP_SYN, RTS, CTS, WLAN_ACK, OSPF_ HELLO, RIP packets etc. Note that this is a link (PHY layer) level measure, and it is not a MAC layer measure. Therefore, the packets transmitted can be greater than the packets generated when running wireless protocols due to re-tries.

- **Packets Errored:** Total number of packets error in the link inclusive of data and control packets.
- **Packets Collided:** Total number of packets collided in the link including data and control packets.
    - If two packets collide then this counter is incremented by two (once for each packet).
    - If a single packet collides N times, then this counter is incremented N times
- **Bytes Transmitted:** It is the total number of bytes transmitted in the link. It is equal to the sum of the 'Payload Transmitted' and 'Overhead Transmitted' transmitted in the link.
- **Payload Transmitted:** It is the total payload transmitted in the link.
- **Overhead Transmitted:** It is the total overhead transmitted in the link. It includes the layer wise overheads and all control packets in the link.

8.1.6 Queue Metrics

Displays the values of the queue metrics for the devices containing buffer queue like routers, access points etc.

- **Device Id** - Unique id number of the device.
- **Port Id** - Unique id number of the port of the device. This is also called as interface id.
- **Queued Packet** - Number of packets queued at a particular port of a device.
- **Dequeued Packet** - Number of packets removed from the queue at a particular port of device.
- **Dropped Packet** - Number of packets dropped at a particular port of a device.

8.1.7 Protocol Metrics

The Performance metrics tables of protocols such as TCP, UDP, IP, IEEE802.11, LTE, AODV and DSR are provided in the respective technology library documentation.

8.1.8 Device Metrics

Displays device related metrics like ARP table, IP forwarding tables. This is also dependent upon the type of network/technology simulated.

IP_Forwarding Table

- **Network Destination** - It represents the Network address of the destination.
- **Netmask/Prefix length** - A 32-bit combination used to describe which portion of an address refers to the subnet and which part refers to the host.
- **Gateway** - It is the IP address of the next-hop router.
- **Interface** - It represents a network connection.
- **Metrics** - It is the value used to choose between two routes.

- **Type** - It represents the type of the network i.e. local/Multicast/Broadcast

Switch MAC Address Table: These metrics will be displayed when we run networks having Switches.

- **MAC Address** - It represents the MAC address of the switch interfaces.
- **Type** - It is the type of the switch.
- **Outport** - It is the output port of the switch.

8.1.9 Cellular Metrics

Displayed if GSM or CDMA is running in the network.

GSM/CDMA Metrics. MS Metrics

- **MS Id** - It is the id of the Mobile station.
- **Call Generated** - It is the number of calls generated by a Mobile Station.
- **Call Blocked** - It is the number of calls blocked by a Base station when no channel available.
- **Call Blocking probability** - It is the probability of calls blocked by a base station.
- **Channel request sent** - It is the number of channel requests sent by a mobile station.
- **Call request sent** - It is the number of call requests sent by a mobile station (at source)
- **Call request received** - It is the number of call requests received by a mobile station (at destination)
- **Call accepted** - It represents the number of calls accepted by a mobile station.
- **Call rejected** - It represents the number of calls rejected by a mobile station.
- **Handover request** - It is the number of handover requests sent by a mobile station. Handover refers to the process of transferring an ongoing call or data session from one channel connected to the core network to another channel.
- **Call dropped** - It represents the number of calls dropped by a BS.
- **Call dropping probability** - It represents the probability of number of calls dropped by a BS.

8.1.10 Channel metrics

- **BS Id** - It is the Id of a Base Station.
- **Channel Id** - It represents the channel number.
- **Uplink frequency** - It is the uplink frequency of the GSM network to send data from mobile station to base station.
- **Downlink frequency** - It is the downlink frequency of the GSM network to send data from base station to mobile station.

- **Time slot** - It represents the time slot. In GSM network, Frequency band is divided into 200kHz carriers and then each carrier is divided into 8 time slots (0-7).

8.1.11 Sensor Metrics (IEEE802.15.4_Metrics)

Displayed if WSN/IOT is running in the network.

- **Device Id** - It represents the Id's of the sensor and LoWPAN Gateway.
- **Packet Transmitted** - It is the number of packets (either data/routing/ZigBee) transmitted by Sensor and LoWPAN gateway
- **Packet Received** - It is the number of packets (either data/routing/ZigBee) received by Sensor and LoWPAN gateway
- **Ack Transmitted** - It is the number of acknowledgements transmitted by a particular device.
- **Ack Received** - It is the number of acknowledgements received by a particular device.
- **CCA Attempt** - It represents the number of Clear channel Assessment attempts at sensors and LoWPAN Gateway used to determine whether the medium is idle or not.
- **Successful CCA Attempt** - It represents the number of successful CCA attempts at sensors and LoWPAN Gateway.
- **Failed CCA** - It represents the number of failed CCA attempts at sensors.
- **Total Backoff Time** - It is the total backoff time obtained. It is the time that sensors have to wait before attempting to access the channel.
- **Average Backoff time** - It is the average backoff time.
- **Beacon Transmitted** - It the total number of beacons transmitted by a LoWPAN Gateway. It transmits network beacons in a beacon enabled mode. If beacon mode is enabled, it follows slotted CSMA/CA algorithm
- **Beacon Received** - It is the total number of beacons received by the sensors.
- **Beacon Forwarded** - It is the total number of beacons forwarded by the sensors.
- **Beacon Time** - It is the total time calculated for beacon transmission at LoWPAN Gateway.
- **CAP Time** - It is the total Contention Access Period obtained during simulation. During this time, sensors compete for channel.
- **CFP Time** - It is the total Contention free period obtained. In CFP, nodes request for guaranteed time slots. If GTS is allocated, nodes can transmit without contention.

8.1.12 Battery Model

- **Device Name** - It represents the Name and Id of the Sensor
- **Initial Energy** - It represents the initial energy of the sensors.
- **Consumed Energy** - This is the total energy consumed by the respective sensor.
- **Remaining Energy** - This is the remaining energy of the sensor at the end of the simulation.

- **Transmission Energy** - It is the energy consumed by the respective sensor for transmitting data.

$$TransmissionEnergy\ (mJ) = TransmitCurrent\ (mA) \times Voltage\ (V) \times AmountOfTimeRadioIsInTransmitState(s).$$

- **Receiving Energy** - It is the energy consumed by the respective sensor while receiving data.

$$ReceivingEnergy\ (mJ) = ReceiveCurrent\ (mA) \times Voltage\ (V) \times AmountOfTimeRadioIsInReceiveState(s).$$

- **Idle Energy** - When the sensor is active and ready but not currently receiving or transmitting data packets, it is said to be in an idle state. This metrics calculates the energy consumed by the sensor in idle state.

$$IdleEnergy\ (mJ) = IdleCurrent\ (mA) \times Voltage\ (V) \times AmountOfTimeRadioIsInIdleState(s).$$

- **Sleep Energy** - This is the energy consumed when the respective sensor is in an inactive mode.

$$SleepEnergy\ (mJ) = SleepCurrent\ (mA) \times Voltage\ (V) \times AmountOfTimeRadioIsInSleepState(s).$$

8.1.13 CR metrics

Displayed if 802.22 cognitive radio is running in the network.

8.1.13.1 Base station Metrics

- **BS Id** - It is the id of a Base Station.
- **Interface Id** - It is the Interface Id of a BS
- **SCH sent** - SCH. It is the number of Superframe Control Headers sent by a BS. SCH carries Base Station's MAC address along with the schedule of quiet periods for sensing, as well as other information about the cell.
- **FCH sent** - It represents the number of Frame Control Headers sent by a BS. It is transmitted as a part of Down Stream (DS) Protocol Data Unit in DS subframe specifies length of either DS-Map if transmitted or US-Map. It is sent in the first two subchannels of the symbol immediately following the preamble symbol.
- **DSA req received** - It is the number of Dynamic Service Addition requests received by a BS used to create a new service flow.
- **DSA rep sent** - It is the number of DSA replies sent by a BS.
- **DSC req received** - It is the number of Dynamic Service Change requests received by a BS to dynamically change the parameters of an existing service flow.
- **DSC rep sent** - It is the number of DSC replies sent by a BS.

- **DSD req received** - It is the number of Dynamic Service Deletion requests received by a BS to delete an existing service flow.
- **DSD rep sent** - It is the number of DSD replies sent by a BS.
- **CHS req sent** - It is the number of Channel Switch Requests sent by a BS.

8.1.13.2 CPE metrics

- **CPE Id** - It represents the Id of Customer Premise Equipment
- **Interface Id** - It represents the Interface Id of the CPE
- **SCH received** - It is the number of Superframe Control Headers received by a CPE.
- **FCH received** - It represents the number of Frame Control Headers received by a CPE.
- **DSA req sent** - It is the number of Dynamic Service Addition requests sent by a CPE.
- **DSA rep received** - It is the number of DSA replies received by a CPE.
- **DSC req sent** - It is the number of Dynamic Service Change requests sent by a CPE.
- **DSC rep received** - It is the number of DSC replies received by a CPE.
- **DSD req sent** - It is the number of Dynamic Service Deletion requests sent by a CPE.
- **DSD rep received** - It is the number of DSD replies received by a CPE.
- **CHS req received** - It is the number of Channel Switch Requests received by a CPE.
- **UCS Sent** - It is the number of Urgent Coexistence Situations sent by a CPE.

8.1.13.3 Incumbent Metrics

- **BS Id** - It represents the Id of the Base Station
- **Incumbent Id** - It represents the Id of the Incumbent.
- **Frequency** - It is the frequency at which the incumbent operates.
- **Operational Time** - It is the active period of the incumbent.
- **Idle Time** - It is the inactive period of the incumbent.
- **Interference Time** - It is the time when interference occurs due to CPE.

8.1.13.4 Channel Metrics

- **BS Id** - It is the Id of the BS
- **Channel Number** - It represents the channel number at which the BS is operating.
- **Frequency** - It is the frequency of the channel at which the BS is operating.
- **Spectral efficiency** - It refers to the information rate that can be transmitted over a given bandwidth in a specific communication system. It is a measure of how efficiently a limited frequency spectrum is utilized by the physical layer protocol, and sometimes by the media access control protocol.

8.1.14 Application Metrics

Displays Application performance metrics.

- **Application Id** - It is the unique Id of the application running at the source.
- **Application Name** - It is unique name of the application running.
- **Source Id** - It is the unique Id of the device running that particular application.
- **Destination Id** - It is the unique Id of the destination device.
- **Packet generated** - It is the total number of packets generated from the source.
- **Packets Transmitted** - It is the total number of packets generated and transmitted from the source.
- **Packet received** - It is the total number of packets received at the destination.
- **Payload Transmitted** - It is the total payload transmitted in bytes. It is equal to the product of 'Packets Transmitted' and 'Packet Size'. This calculation will apply only in case of a constant packet size (CBR, CUSTOM (constant) etc. In other cases, this should be considered as the sum of the payload of the packets transmitted.
- **Payload Received** - It is the total payload received at the destination in bytes.
- **Throughput** - Total user data (or) payload delivered to their respective destination every second.

  If Simulation Time > Application End Time, then

$$Application\ Throughput(Mbps) = \frac{Total\ payload\ delivered\ to\ destination\ (bytes)\ *\ 8}{Time\ last\ received\ packet\ at\ App\ layer(\mu s) - App\ Start\ Time\ (\mu s)}$$

  If Simulation Time < Application End Time, then

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

- **Jitter**

$$Jitter(\mu s) = \frac{TotalPacket\ Jitter\ of\ all\ successfull\ packets}{Total\ Number\ of\ successfully\ recieved\ packets - 1}$$

$$Packet\ Jitter\ (\mu s) = |EndtoEnd\ Delay\ of\ Current\ packet - EndtoEnd\ Delay\ of\ Previous\ Packet|$$

- **Delay** - It is the average amount of time taken (calculated for all successful packets) to reach the destination application layer from when the packet is sent from source's application later. It would APP_IN time at destination – APP_OUT time at source.

8.1.15 IP Metrics

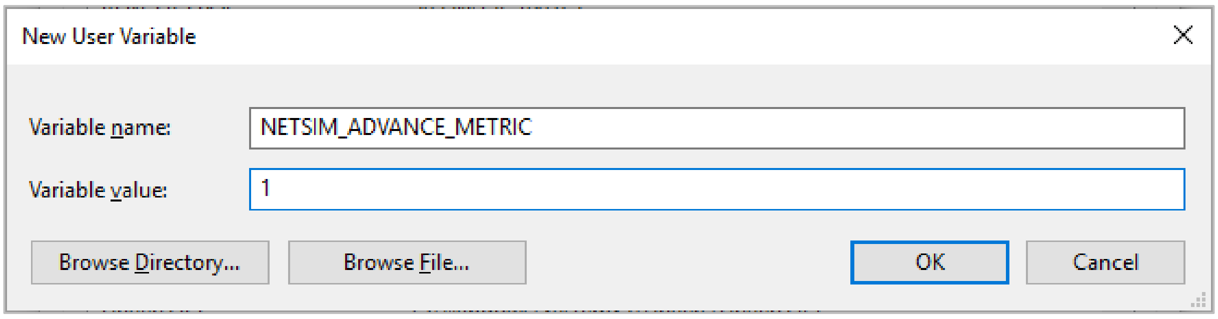
IP layer metrics calculated for the overall network and displayed for each device.

- **Device Id** - It is the unique ID of the Device.
- **Packet sent** - Specifies the number of packets (L3 and above) sent by the node.
- **Packet forwarded** - Specifies the number of packets (L3 and above) forwarding by an intermediate node(s) to next hop/target node.
- **Packets Received** - Specifies the number of packets (L3 and above) successfully received at the destination, from intermediate node(s) and source node(s).
- **Packets discarded** - Specifies the number of packets (L3 and above) discarded when there is no route available.
- **TTL Expired** - Specifies the number of Data and Control packets (L3 and above) dropped when TTL expires.
- **Firewall block** - Specifies the number of packets (L3 and above) blocked by Firewall for example TCP, UDP and ICMP Packets etc.

8.1.16 Advanced Metrics

In the Application metrics table, in addition to packets generated and packets received, additional information on duplicate packets that were received can be obtained. This is achieved by adding the following environment variable:

PC Settings → Properties → Advance system settings → Environment Variables → User Variables → New



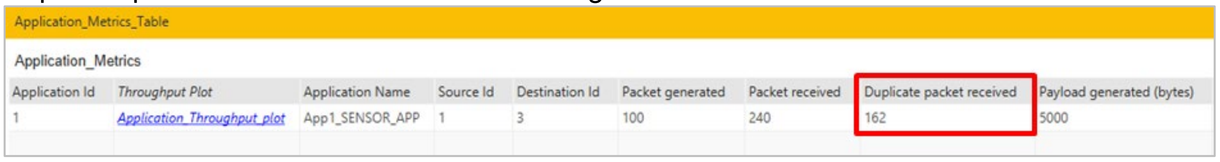
Variable name: NETSIM_ADVANCE_METRIC

Variable value: 1

Figure 8-7: Environment Variables window

***Note:** For the effect of changes to User needs to restart the System.*

The Application metrics table in the results dashboard will display an additional column – Duplicate packet received as shown below Figure 8-8.



Application_Metrics_Table

Application_Metrics

Application Id	Throughput Plot	Application Name	Source Id	Destination Id	Packet generated	Packet received	Duplicate packet received	Payload generated (bytes)
1	Application_Throughput_plot	App1_SENSOR_APP	1	3	100	240	162	5000

Figure 8-8: Application metrics table in results window

***Note**: Note that keeping track of duplicate packets will slow down the simulation*

8.1.17 Notes on metrics

1. The metrics are calculated at each layer and might not be equivalent to the same metric calculated at a different layer. For exactness and precision, we recommend users also verify the results with the event trace & packet trace generated by NetSim.
2. Broadcast / Multicast application will have no entries under Application Metrics in Results window if there are zero packets received. In other words, it will not show ‘0’ throughput. Users may notice that ‘0’ throughput is shown for unicast applications, and this is because of the way Broadcast/Multicast application metrics is architected in NetSim.

8.1.18 The different results files written at the end of simulation

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.xml	Contains the metrics of the network that is simulated recently.
2	Node.pcap	Contains the information of captured packets that is recently simulated.
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.csv	Contains the detailed packet information. This file will be written only when Packet Trace is enabled.
7	EventTrace.csv	Contains the information about each event. This file will be written only when Event Trace is enabled.
8	Animation.txt	Contains the information about the flow of the packet.
9	Static ARP.txt	Contains the information about the dropped devices like Ip address and mac address.

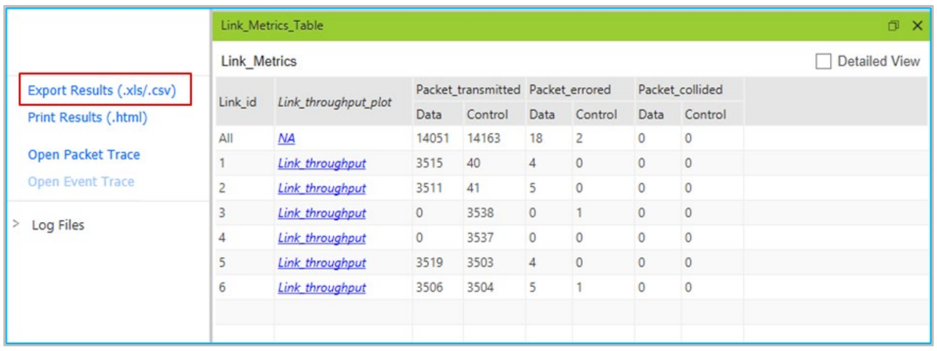
Table 8-1: Different results files written at the end of simulation in I/O Path

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

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

8.2 Export to .csv

In NetSim Result Dashboard, users can use the option **Export Results (.xls/.csv)** to export all the metrics file to XL/CSV file for the further computation or analysis using it.



Export Results (.xls/.csv)
Print Results (.html)
Open Packet Trace
Open Event Trace
Log Files

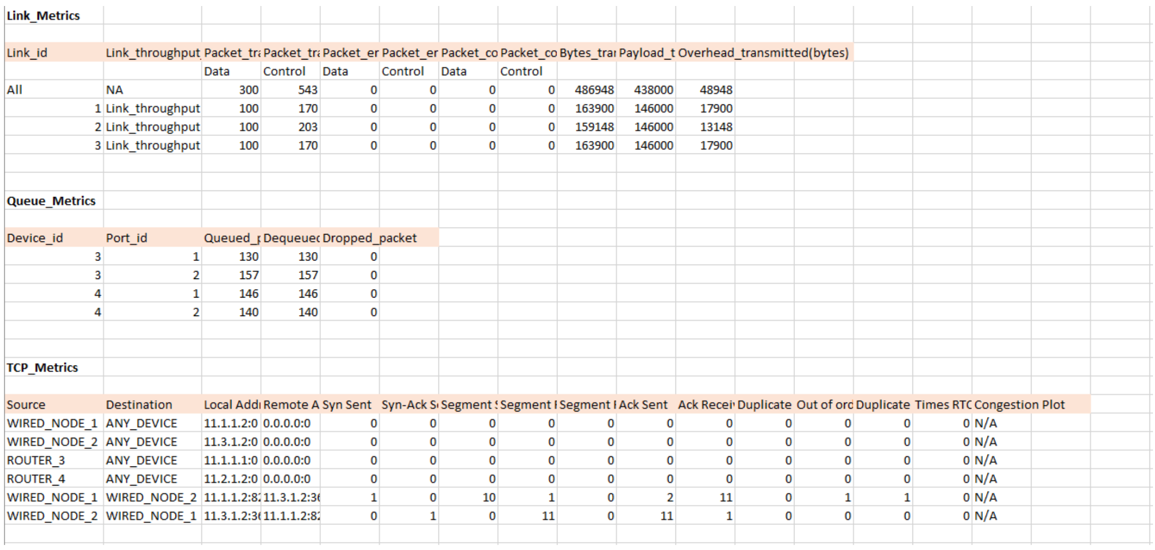
Link_Metrics_Table

Link_Metrics ☐ Detailed View

Link_id	Link_throughput_plot	Packet_transmitted		Packet_errored		Packet_collided	
		Data	Control	Data	Control	Data	Control
All	NA	14051	14163	18	2	0	0
1	Link_throughput	3515	40	4	0	0	0
2	Link_throughput	3511	41	5	0	0	0
3	Link_throughput	0	3538	0	1	0	0
4	Link_throughput	0	3537	0	0	0	0
5	Link_throughput	3519	3503	4	0	0	0
6	Link_throughput	3506	3504	5	1	0	0

Figure 8-9: Select Option Export Results (.xls/.csv) in Result window

XL/CSV file:



Link_Metrics

Link_id	Link_throughput	Packet_tra	Packet_tra	Packet_er	Packet_er	Packet_co	Packet_co	Bytes_trai	Payload_t	Overhead_transmitted(bytes)
		Data	Control	Data	Control	Data	Control			
All	NA	300	543	0	0	0	0	486948	438000	48948
1	Link_throughput	100	170	0	0	0	0	163900	146000	17900
2	Link_throughput	100	203	0	0	0	0	159148	146000	13148
3	Link_throughput	100	170	0	0	0	0	163900	146000	17900

Queue_Metrics

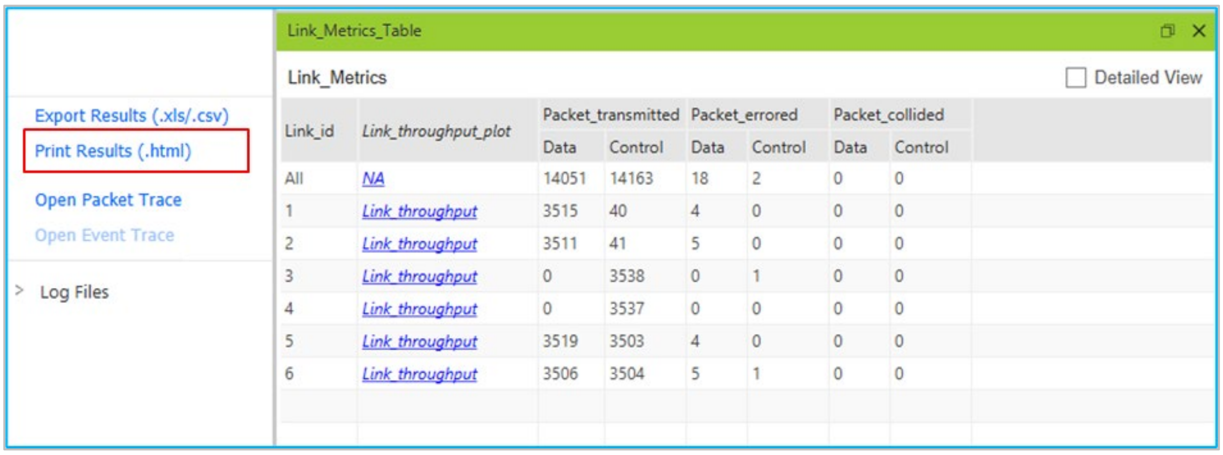
Device_id	Port_id	Queued_p	Dequeuec	Dropped_packet
3	1	130	130	0
3	2	157	157	0
4	1	146	146	0
4	2	140	140	0

TCP_Metrics

Source	Destination	Local Addı	Remote A	Syn Sent	Syn-Ack Sı	Segment !	Segment I	Segment I	Ack Sent	Ack Recei	Duplicate	Out of ord	Duplicate	Times RTC	Congestion Plot
WIRED_NODE_1	ANY_DEVICE	11.1.1.2:0	0.0.0.0:0	0	0	0	0	0	0	0	0	0	0	0	N/A
WIRED_NODE_2	ANY_DEVICE	11.3.1.2:0	0.0.0.0:0	0	0	0	0	0	0	0	0	0	0	0	N/A
ROUTER_3	ANY_DEVICE	11.1.1.1:0	0.0.0.0:0	0	0	0	0	0	0	0	0	0	0	0	N/A
ROUTER_4	ANY_DEVICE	11.2.1.2:0	0.0.0.0:0	0	0	0	0	0	0	0	0	0	0	0	N/A
WIRED_NODE_1	WIRED_NODE_2	11.1.1.2:8:	11.3.1.2:3(	1	0	10	1	0	2	11	0	1	1	0	N/A
WIRED_NODE_2	WIRED_NODE_1	11.3.1.2:3(	11.1.1.2:8:	0	1	0	11	0	11	1	0	0	0	0	N/A

Figure 8-10: Option Export Results (.xls/.csv) to export all the metrics

A web formatted (html file) report can be generated for simulations performed in NetSim, using the Print button present in the results window as shown below Figure 8-11.



Export Results (.xls/.csv)
Print Results (.html)
Open Packet Trace
Open Event Trace
Log Files

Link_Metrics_Table

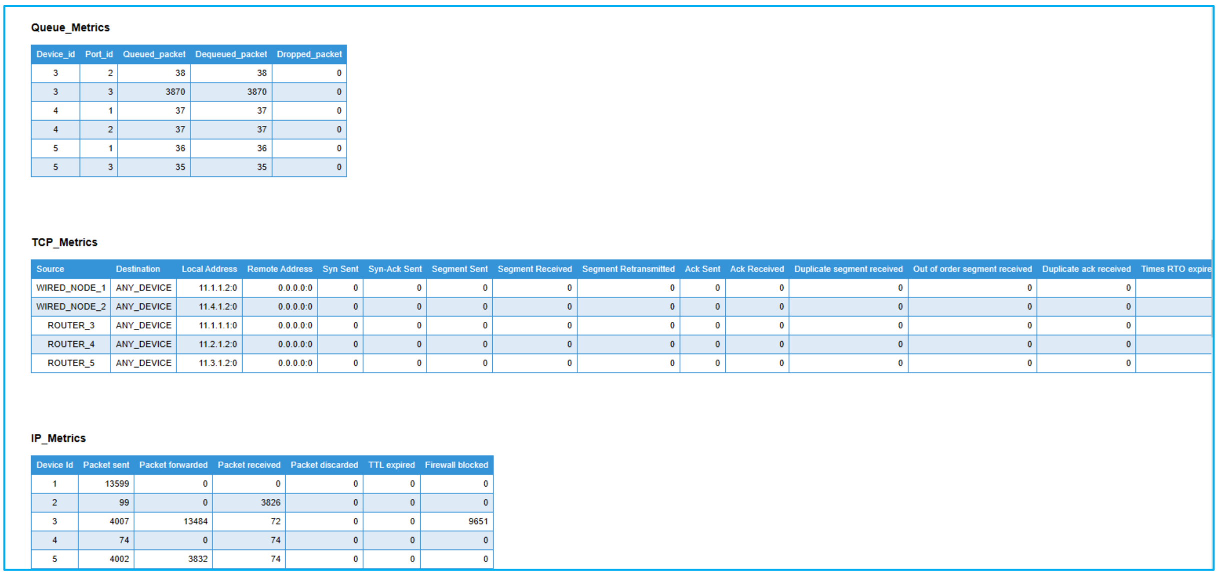
Link_Metrics ☐ Detailed View

Link_id	Link_throughput_plot	Packet_transmitted		Packet_errored		Packet_collided	
		Data	Control	Data	Control	Data	Control
All	NA	14051	14163	18	2	0	0
1	Link_throughput	3515	40	4	0	0	0
2	Link_throughput	3511	41	5	0	0	0
3	Link_throughput	0	3538	0	1	0	0
4	Link_throughput	0	3537	0	0	0	0
5	Link_throughput	3519	3503	4	0	0	0
6	Link_throughput	3506	3504	5	1	0	0

Figure 8-11: Print Results(.html) in Results window

The report that is generated contains:

- A screenshot of the network scenario created in NetSim GUI.
- All the metrics tables that were part of the Simulation Results Window
- Dynamic Metrics Plots (if Dynamic Metrics is enabled prior to Simulation).



Queue_Metrics

Device_id	Port_id	Queued_packet	Dequeued_packet	Dropped_packet
3	2	38	38	0
3	3	3870	3870	0
4	1	37	37	0
4	2	37	37	0
5	1	36	36	0
5	3	35	35	0

TCP_Metrics

Source	Destination	Local Address	Remote Address	Syn Sent	Syn-Ack Sent	Segment Sent	Segment Received	Segment Retransmitted	Ack Sent	Ack Received	Duplicate segment received	Out of order segment received	Duplicate ack received	Times RTO expire
WIRED_NODE_1	ANY_DEVICE	11.1.1.2:0	0.0.0.0:0	0	0	0	0	0	0	0	0	0	0	
WIRED_NODE_2	ANY_DEVICE	11.4.1.2:0	0.0.0.0:0	0	0	0	0	0	0	0	0	0	0	
ROUTER_3	ANY_DEVICE	11.1.1.1:0	0.0.0.0:0	0	0	0	0	0	0	0	0	0	0	
ROUTER_4	ANY_DEVICE	11.2.1.2:0	0.0.0.0:0	0	0	0	0	0	0	0	0	0	0	
ROUTER_5	ANY_DEVICE	11.3.1.2:0	0.0.0.0:0	0	0	0	0	0	0	0	0	0	0	

IP_Metrics

Device Id	Packet sent	Packet forwarded	Packet received	Packet discarded	TTL expired	Firewall blocked
1	13599	0	0	0	0	0
2	99	0	3826	0	0	0
3	4007	13484	72	0	0	9651
4	74	0	74	0	0	0
5	4002	3832	74	0	0	0

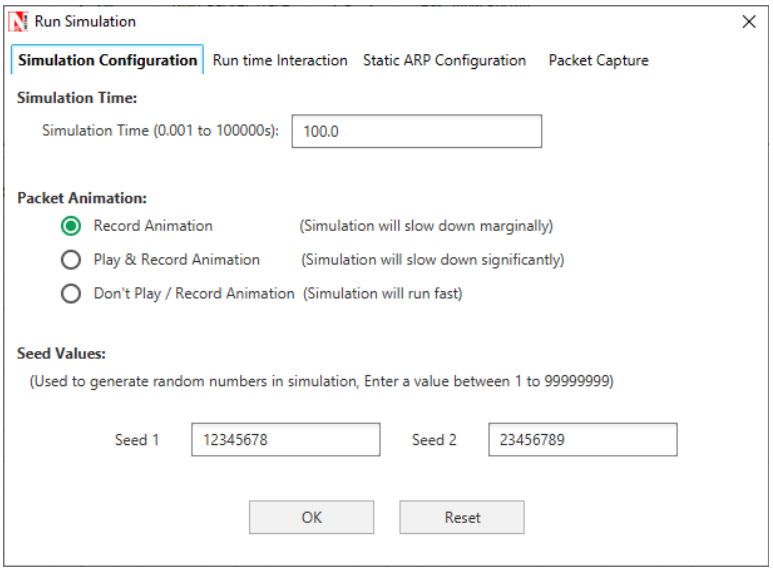
Figure 8-12: html report generated in PDF format

- The report that is generated makes it convenient for documentation, reference, study and further analysis.
- This html report can be printed as PDF or printed out by selecting printer options.

8.3 Packet Animation

NetSim provides the feature to play and record animations to the user. Packet animation enables users to watch traffic flow through the network for in-depth visualization and analysis. Users have the following options before running simulation:

- Record the animation.
- Don't play/ record animation and
- Play and record animation while running simulation.



Simulation Time:

Simulation Time (0.001 to 100000s): 100.0

Packet Animation:

[✓ Record Animation] (Simulation will slow down marginally)
Play & Record Animation (Simulation will slow down significantly)
Don't Play / Record Animation (Simulation will run fast)

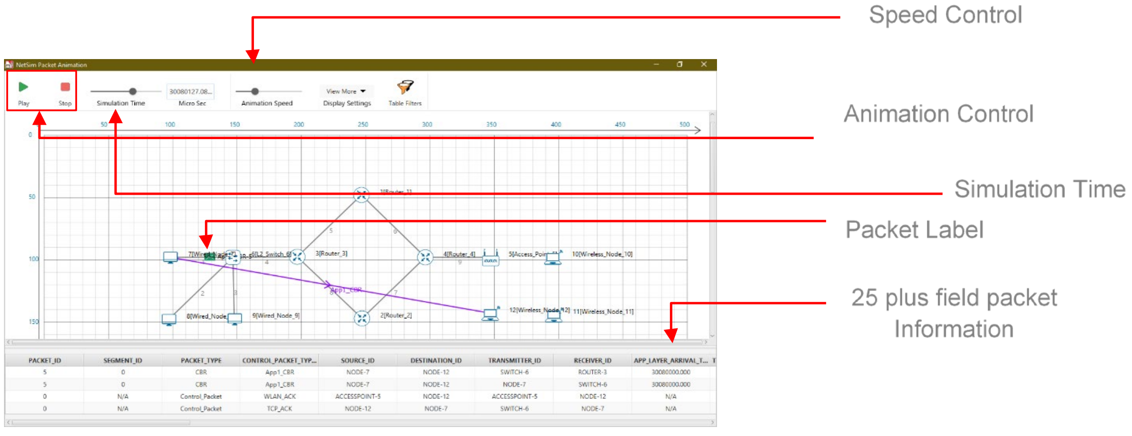
Seed Values:

(Used to generate random numbers in simulation, Enter a value between 1 to 99999999)

Seed 1: 12345678 Seed 2: 23456789

Figure 8-13: Run Simulation window

The packet animation would then be recorded, and the user can view the animation from the NetSim Packet Animation window as shown below Figure 8-14.



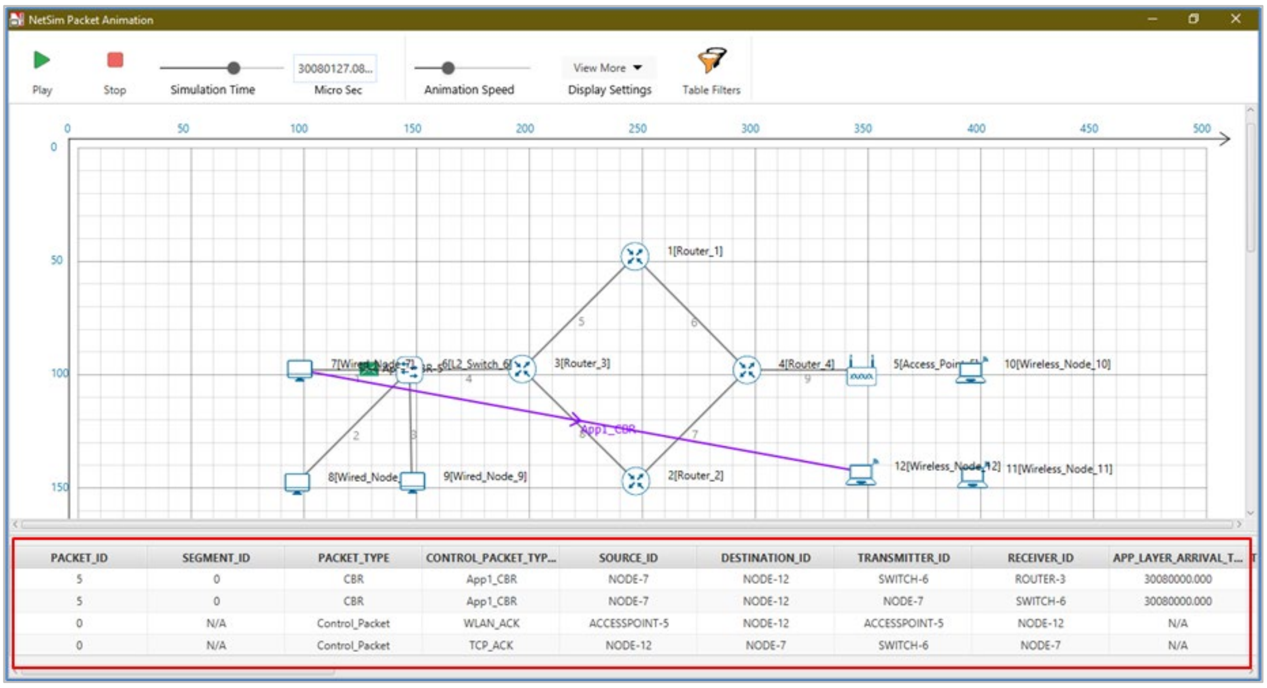
PACKET_ID	SEGMENT_ID	PACKET_TYPE	CONTROL_PACKET_TYP...	SOURCE_ID	DESTINATION_ID	TRANSMITTER_ID	RECEIVER_ID	APP_LAYER_ARRIVAL_T...
5	0	CBR	App1_CBR	NODE-7	NODE-12	SWITCH-6	ROUTER-3	30080000.000
5	0	CBR	App1_CBR	NODE-7	NODE-12	NODE-7	SWITCH-6	30080000.000
0	N/A	Control_Packet	WLAN_ACK	ACCESSPOINT-5	NODE-12	ACCESSPOINT-5	NODE-12	N/A
0	N/A	Control_Packet	TCP_ACK	NODE-12	NODE-7	SWITCH-6	NODE-7	N/A

Figure 8-14: Packet Animation window

While viewing packet animation, user can see the flow of packets as well as the type of packet. Blue color packet denotes control packet, green color is used for data packet and red color is error/collided packet.

8.3.1 Packet animation Table

Packet Animation table is also provided for users to see the flow of packets along with packet animation.



PACKET_ID	SEGMENT_ID	PACKET_TYPE	CONTROL_PACKET_TYP...	SOURCE_ID	DESTINATION_ID	TRANSMITTER_ID	RECEIVER_ID	APP_LAYER_ARRIVAL_T...
5	0	CBR	App1_CBR	NODE-7	NODE-12	SWITCH-6	ROUTER-3	30080000.000
5	0	CBR	App1_CBR	NODE-7	NODE-12	NODE-7	SWITCH-6	30080000.000
0	N/A	Control_Packet	WLAN_ACK	ACCESSPOINT-5	NODE-12	ACCESSPOINT-5	NODE-12	N/A
0	N/A	Control_Packet	TCP_ACK	NODE-12	NODE-7	SWITCH-6	NODE-7	N/A

Figure 8-15: Packet Animation table in animation window

The “**Table Filters**” option available in the Packet Animator Window allows users to filter the parameters that will be displayed in the Packet Trace Window displayed alongside animation.

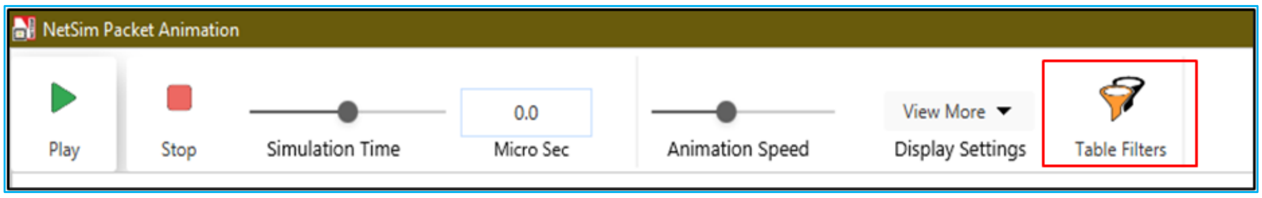


Figure 8-16: Table Filters option available in the Packet Animator Window

***Note**: Packet Animation table would be displayed only if Packet Trace is enabled in the network before running the simulation.*

8.3.2 Packet animation – Display Settings

NetSim Packet Animation can be customized using the View More drop-down list provided with the display settings as shown below Figure 8-17.

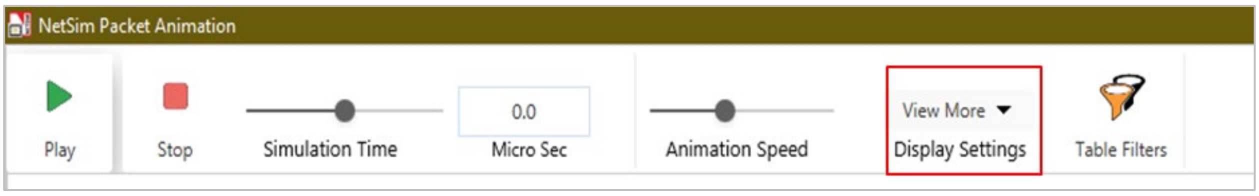


Figure 8-17: Display Settings in Packet animation window

The View More Animation options can be used to view (enable/disable)

- Device Name
- IP address of devices
- VLAN ID
- Application Flow
- Node Movement
- Packet Flow
- Battery Level
- Route tables etc alongside animation

***Note**: The options displayed under **View more** drop down are dependent on the network that is simulated and features that are enabled.*

8.3.3 Example on how to use NetSim packet animation feature:

Case 1: ARP PROTOCOL - WORKING

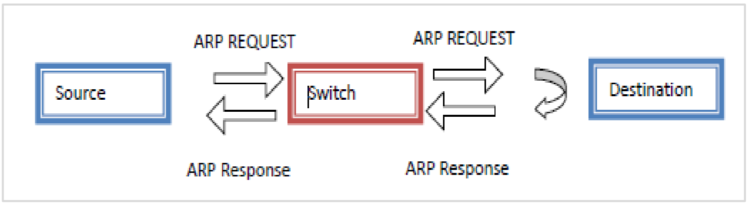


Figure 8-18: Intra LAN IP Forwarding

- Create a scenario with 3 wired nodes, 2 switches and 1 router and connect it based on the following scenario.

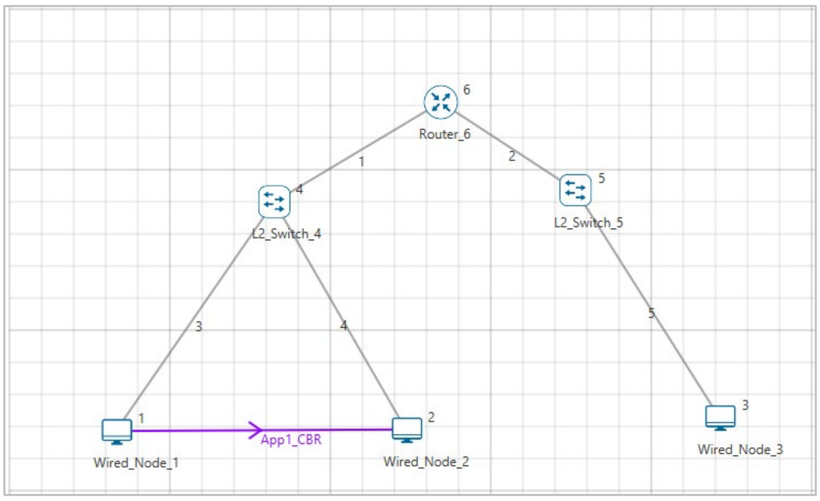
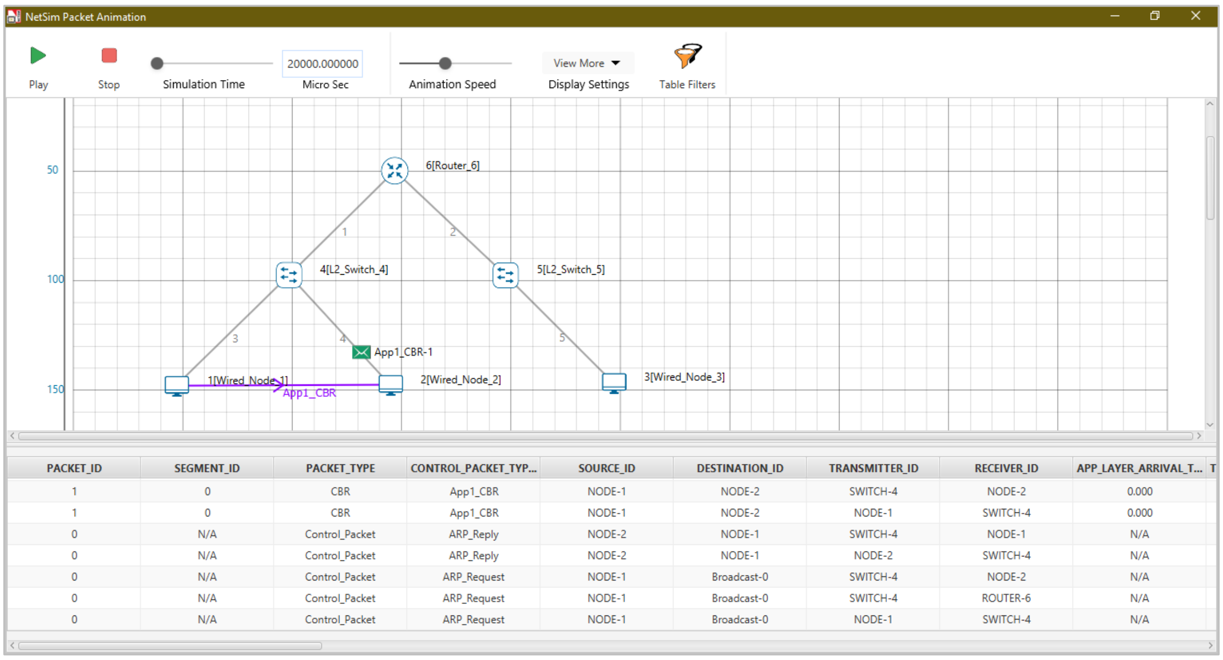


Figure 8-19: Application flow within a LAN

- Transport Protocol is set to **UDP** instead of TCP for all the wired nodes.
- Click on application and set Source_Id and Destination_Id as 1 and 2 respectively.
- Set Simulation time = 100s. After clicking on Run Simulation, edit **Static ARP Configuration** tab by setting Static ARP as Disable. Click on OK button to simulate.

Now click on packet animation and analyze the following:



PACKET_ID	SEGMENT_ID	PACKET_TYPE	CONTROL_PACKET_TYP...	SOURCE_ID	DESTINATION_ID	TRANSMITTER_ID	RECEIVER_ID	APP_LAYER_ARRIVAL_T...	T
1	0	CBR	App1_CBR	NODE-1	NODE-2	SWITCH-4	NODE-2	0.000	
1	0	CBR	App1_CBR	NODE-1	NODE-2	NODE-1	SWITCH-4	0.000	
0	N/A	Control_Packet	ARP_Reply	NODE-2	NODE-1	SWITCH-4	NODE-1	N/A	
0	N/A	Control_Packet	ARP_Reply	NODE-2	NODE-1	NODE-2	SWITCH-4	N/A	
0	N/A	Control_Packet	ARP_Request	NODE-1	Broadcast-0	SWITCH-4	NODE-2	N/A	
0	N/A	Control_Packet	ARP_Request	NODE-1	Broadcast-0	SWITCH-4	ROUTER-6	N/A	
0	N/A	Control_Packet	ARP_Request	NODE-1	Broadcast-0	NODE-1	SWITCH-4	N/A	

Figure 8-20: Packet animation window

- NODE-1 sends ARP_Request which is then broadcasted by SWITCH-4.
- During the process the devices that receive the ARP_Request packet (Switch, Router, and Node-2) will update their ARP table or the switch table.
- NODE -2 sends the ARP_Reply to NODE-1 via SWITCH-4.
- Now NODE-1 updates its ARP table with the MAC address of NODE-2 on receiving the ARP_Reply.

- After this step, NODE-1 starts sending data packets to NODE-2 since the source now has both IP and MAC addresses of destination.

Case 2: Across-Router-IP-forwarding

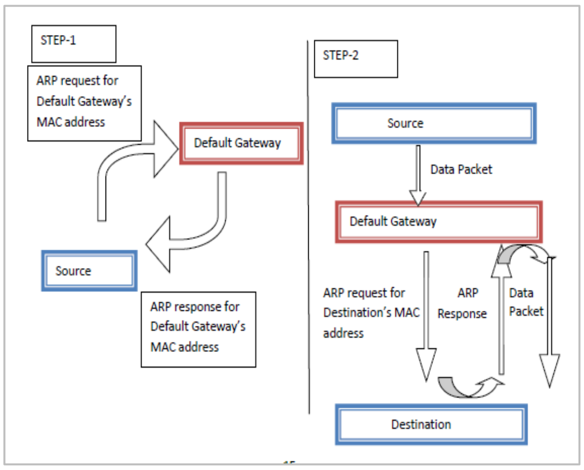
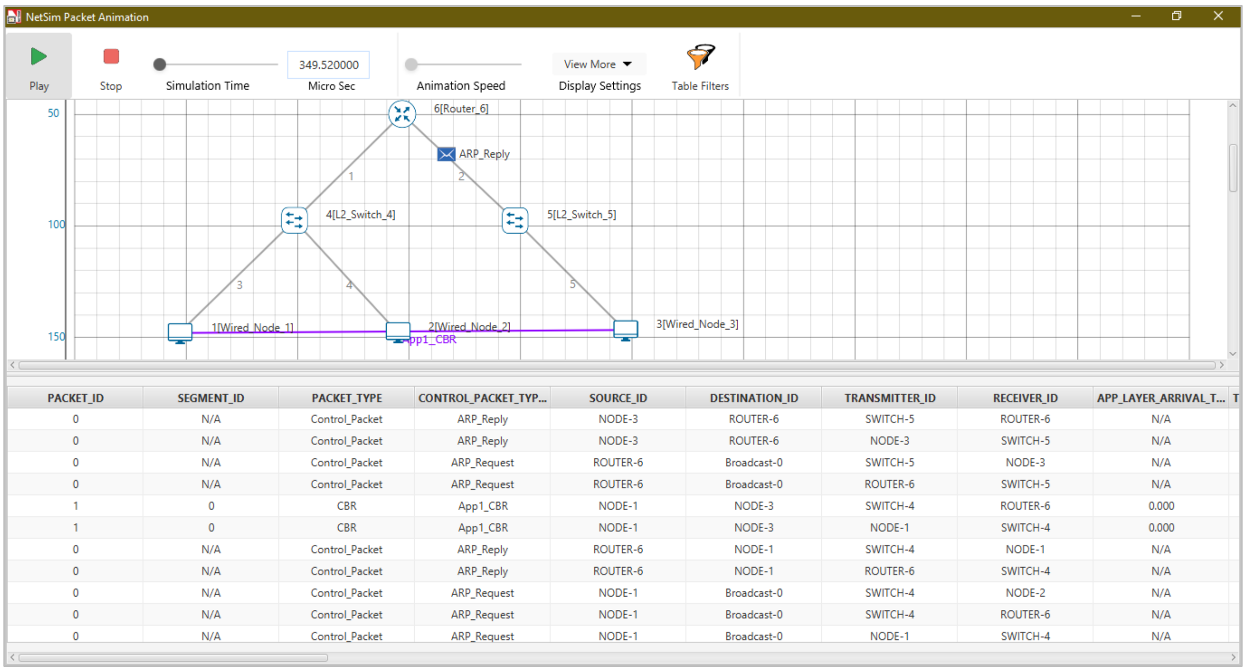

STEP-1
ARP request for Default Gateway's MAC address
Default Gateway
Source
ARP response for Default Gateway's MAC address
STEP-2
Source
Data Packet
Default Gateway
ARP request for Destination's MAC address
ARP Response
Data Packet
Destination

Figure 8-21: Across Router IP Forwarding

- Follow all the steps till Step 2 and perform the following sample.
- To run the simulation, click on the Application icon and set the Source_Id and Destination_Id as 1 and 3 respectively.
- Click on Run Simulation and set Simulation time as 100 sec.
- Then go to **Static ARP Configuration** tab and set Static ARP as Disable. Click on OK button to simulate.

Click on packet animation to analyse the following:



PACKET_ID	SEGMENT_ID	PACKET_TYPE	CONTROL_PACKET_TYP...	SOURCE_ID	DESTINATION_ID	TRANSMITTER_ID	RECEIVER_ID	APP_LAYER_ARRIVAL_T...	T
0	N/A	Control_Packet	ARP_Reply	NODE-3	ROUTER-6	SWITCH-5	ROUTER-6	N/A	
0	N/A	Control_Packet	ARP_Reply	NODE-3	ROUTER-6	NODE-3	SWITCH-5	N/A	
0	N/A	Control_Packet	ARP_Request	ROUTER-6	Broadcast-0	SWITCH-5	NODE-3	N/A	
0	N/A	Control_Packet	ARP_Request	ROUTER-6	Broadcast-0	ROUTER-6	SWITCH-5	N/A	
1	0	CBR	App1_CBR	NODE-1	NODE-3	SWITCH-4	ROUTER-6	0.000	
1	0	CBR	App1_CBR	NODE-1	NODE-3	NODE-1	SWITCH-4	0.000	
0	N/A	Control_Packet	ARP_Reply	ROUTER-6	NODE-1	SWITCH-4	NODE-1	N/A	
0	N/A	Control_Packet	ARP_Reply	ROUTER-6	NODE-1	ROUTER-6	SWITCH-4	N/A	
0	N/A	Control_Packet	ARP_Request	NODE-1	Broadcast-0	SWITCH-4	NODE-2	N/A	
0	N/A	Control_Packet	ARP_Request	NODE-1	Broadcast-0	SWITCH-4	ROUTER-6	N/A	
0	N/A	Control_Packet	ARP_Request	NODE-1	Broadcast-0	NODE-1	SWITCH-4	N/A	

Figure 8-22: Packet animation window

- NODE-1 transmits ARP_Request which is further broadcasted by SWITCH-4. ROUTER-6 sends ARP_Reply to NODE-1 which goes through SWITCH-4. Then NODE-1 starts to send data to NODE-3.
- If the router has the address of NODE-3 in its routing table, ARP protocol ends here, and data transfer starts that is PACKET_ID 1 is being sent from NODE-1 to NODE-3.
- In other case, Router sends ARP_Request to appropriate subnet and after getting the MAC ADDRESS of the NODE-3, it forwards the packet which it has received from NODE-1.
- 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 the 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.

8.3.4 How to record and save Packet animation as a Video file

Note: *The following procedure applies to Windows 10 Operating system only. Users with other versions of Windows can use third-party video capture tools (Link to a list of common tools) to save NetSim packet animation as a video.*

To quickly capture NetSim packet animation, launch the packet animation window. Before playing the animation, press **Windows key + G** on the keyboard to open Game bar. (or Select windows settings and then select Gaming option for Game bar related settings). Now start recording by pressing record option as shown below. (Shortcut to start recording **Windows key + Alt + R**)

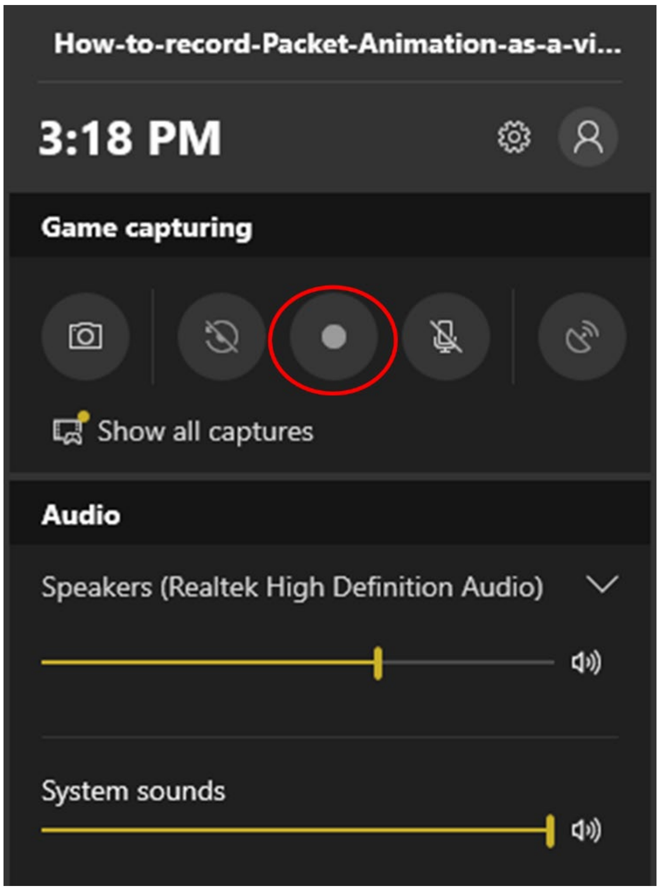
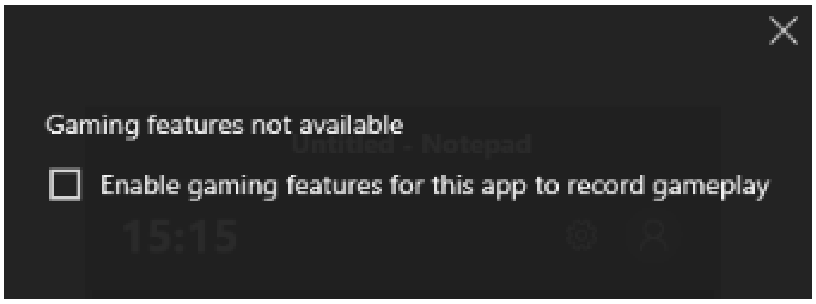

How-to-record-Packet-Animation-as-a-vi...
3:18 PM
Game capturing
Show all captures
Audio
Speakers (Realtek High Definition Audio)
System sounds

Figure 8-23: In packet animation window press Windows key + G on the keyboard and Select Start recording

Then select the checkbox “Enable gaming features for this app to record gameplay” option.



Gaming features not available

☐ Enable gaming features for this app to record gameplay

Figure 8-24: Select the checkbox “Enable gaming features for this app to record gameplay” option

Once you select the checkbox, recording window will open as shown below.

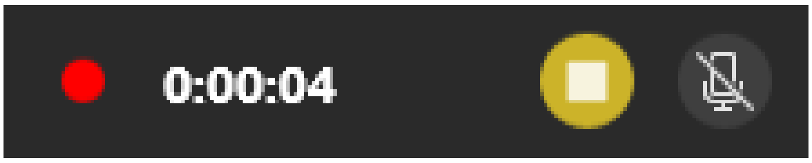
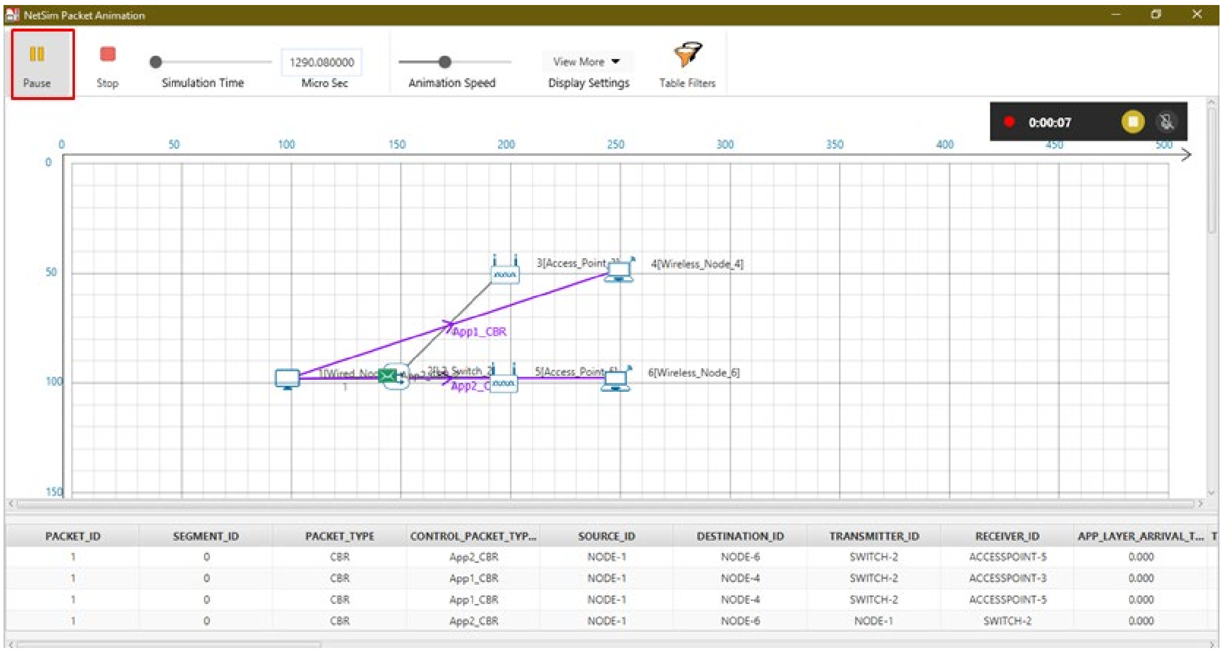


Figure 8-25: Recording window

Now start playing the animation in NetSim using play button in packet animation window.



PACKET_ID	SEGMENT_ID	PACKET_TYPE	CONTROL_PACKET_TYP...	SOURCE_ID	DESTINATION_ID	TRANSMITTER_ID	RECEIVER_ID	APP_LAYER_ARRIVAL_T...	T
1	0	CBR	App2_CBR	NODE-1	NODE-6	SWITCH-2	ACCESSPOINT-5	0.000	
1	0	CBR	App1_CBR	NODE-1	NODE-4	SWITCH-2	ACCESSPOINT-3	0.000	
1	0	CBR	App1_CBR	NODE-1	NODE-4	SWITCH-2	ACCESSPOINT-5	0.000	
1	0	CBR	App2_CBR	NODE-1	NODE-6	NODE-1	SWITCH-2	0.000	

Figure 8-26: Start playing the animation in animation window

Once the animation has been recorded stop recording (Shortcut to stop recording **Windows logo key + Alt + R**). Recorded clips will be saved in windows default videos folder (E.g.: C:\Users\PC\Videos\Captures).

8.4 Packet Trace

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.

The packet trace is written whenever a packet is received at a device. For example, if we have transmission N1 -> N2 -> N3, then the packet trace is written for every packet being received at N2 and at N3. Note that it not written for every packet being transmitted by N1 and the subsequently by N2. This means that packet which are transmitted from N1 but which may have been errored or collided before being received by N2 are not written in the packet trace.

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: *By default, packet tracing option is turned off. Turning on Packet Trace will slow down the simulation significantly. After simulation, users would get the "open packet trace" link in the metrics window (will also get Packet_Trace.csv file in the saved folder).*

8.4.1 How to Enable Packet trace

Step 1: Create network scenario comprising of two Wired nodes and Router, create a traffic flow between the two wired nodes.

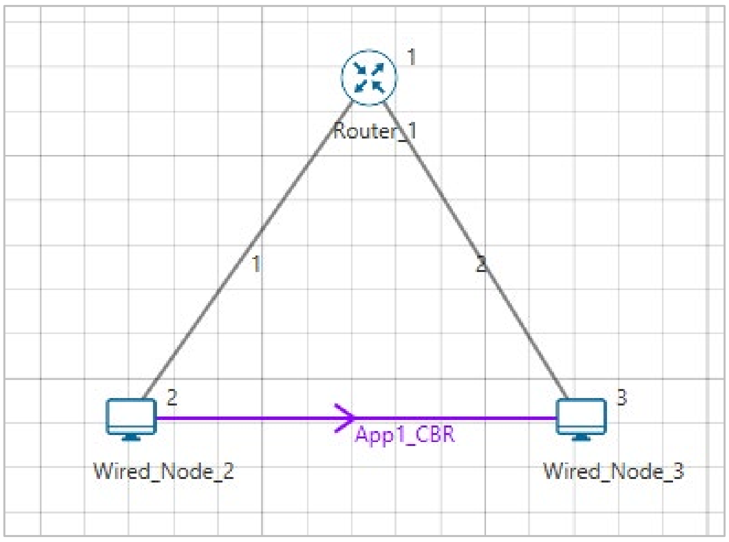


Figure 8-27: Network Scenario

Step 2: By default, Packet trace is disabled, to enable packet trace click on icon in the tool bar as shown in the below figure.

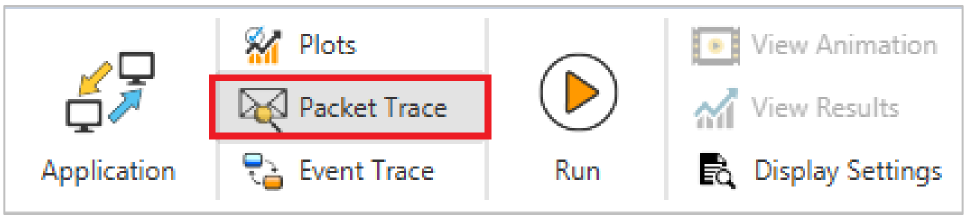


Figure 8-28: Packet trace option in ribbon

Step 3: After Clicking on the packet trace option, users will be allowed select the required properties to log to the packet trace. After selecting the required properties click on the “OK” button.

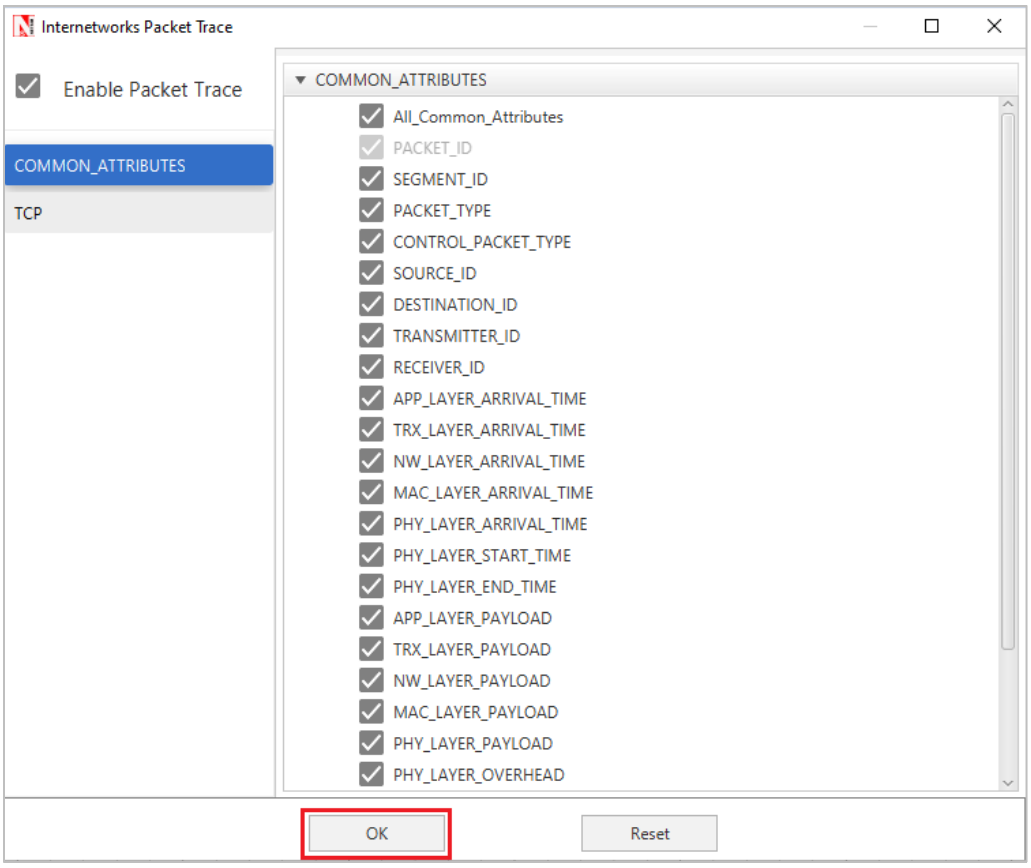
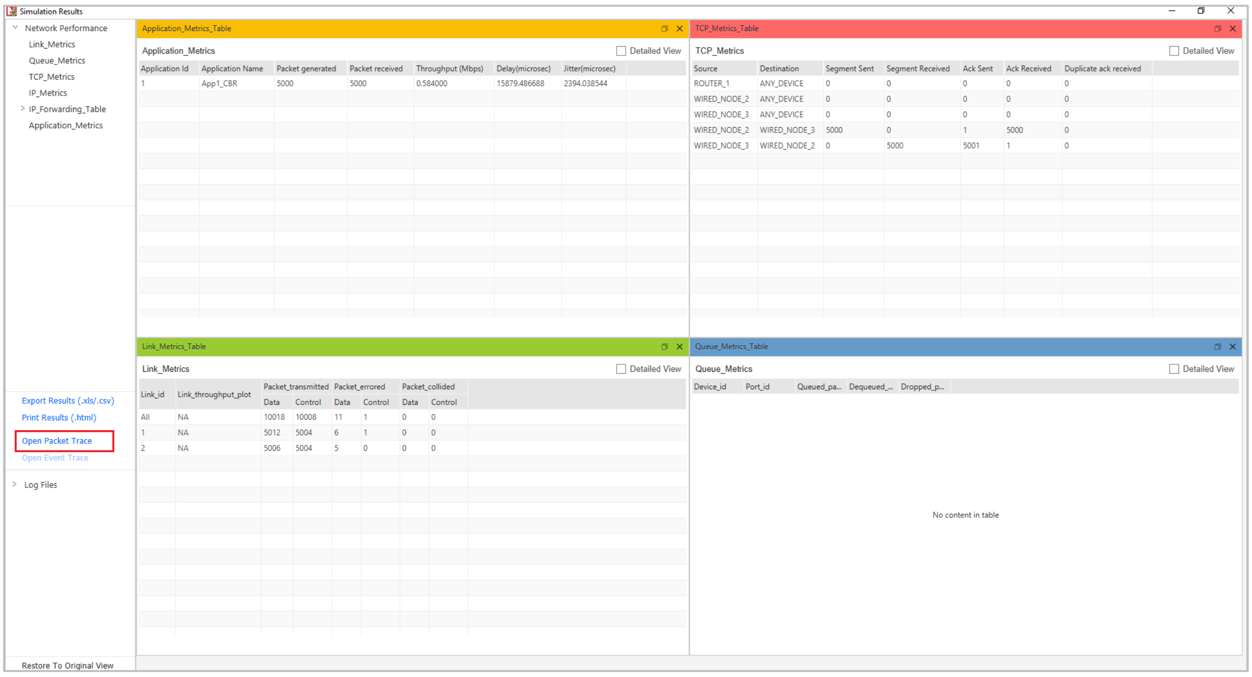


Figure 8-29: Attributes in packet trace

Step 4: Click on the Run button and simulate the scenario. After completion of simulation NetSim Results dashboard window appears.

Click on the Open packet trace option from result dashboard Window as shown below:



Application_Metrics

Application Id	Application Name	Packet generated	Packet received	Throughput (Mbps)	Delay(microsec)	Jitter(microsec)
1	App1_CBR	5000	5000	0.584000	15879.486688	2394.038544

TCP_Metrics

Source	Destination	Segment Sent	Segment Received	Ack Sent	Ack Received	Duplicate ack received
ROUTER_1	ANY_DEVICE	0	0	0	0	0
WIRED_NODE_2	ANY_DEVICE	0	0	0	0	0
WIRED_NODE_3	ANY_DEVICE	0	0	0	0	0
WIRED_NODE_2	WIRED_NODE_3	5000	0	1	5000	0
WIRED_NODE_3	WIRED_NODE_2	0	5000	5001	1	0

Link_Metrics

Link_id	Link_throughput_plot	Packet_transmitted		Packet_errored		Packet_collided	
		Data	Control	Data	Control	Data	Control
All	NA	10018	10008	11	1	0	0
1	NA	5012	5004	6	1	0	0
2	NA	5006	5004	5	0	0	0

Queue_Metrics

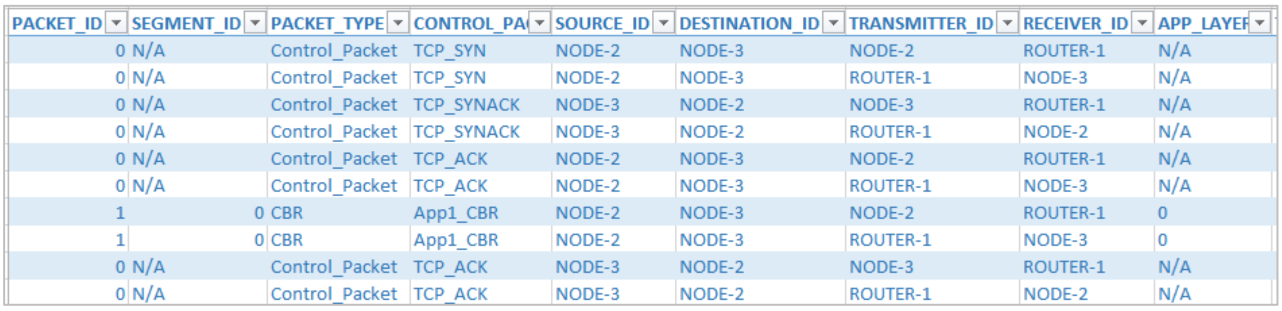
Device_id	Port_id	Queued_pa...	Dequeued_...	Dropped_p...

No content in table

Figure 8-30: Result Dashboard Window

8.4.2 How to set filters to NetSim trace file

Step 1: Open the trace file. (In this example packet trace is opened)



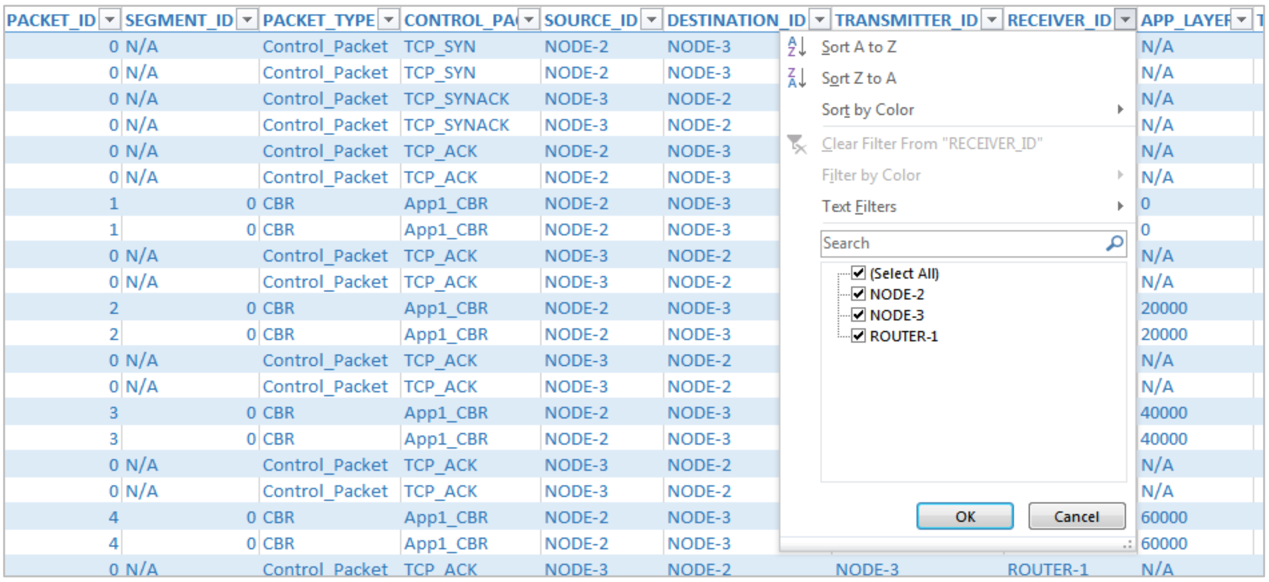
PACKET_ID	SEGMENT_ID	PACKET_TYPE	CONTROL_PA	SOURCE_ID	DESTINATION_ID	TRANSMITTER_ID	RECEIVER_ID	APP_LAYER
0	N/A	Control_Packet	TCP_SYN	NODE-2	NODE-3	NODE-2	ROUTER-1	N/A
0	N/A	Control_Packet	TCP_SYN	NODE-2	NODE-3	ROUTER-1	NODE-3	N/A
0	N/A	Control_Packet	TCP_SYNACK	NODE-3	NODE-2	NODE-3	ROUTER-1	N/A
0	N/A	Control_Packet	TCP_SYNACK	NODE-3	NODE-2	ROUTER-1	NODE-2	N/A
0	N/A	Control_Packet	TCP_ACK	NODE-2	NODE-3	NODE-2	ROUTER-1	N/A
0	N/A	Control_Packet	TCP_ACK	NODE-2	NODE-3	ROUTER-1	NODE-3	N/A
1	0	CBR	App1_CBR	NODE-2	NODE-3	NODE-2	ROUTER-1	0
1	0	CBR	App1_CBR	NODE-2	NODE-3	ROUTER-1	NODE-3	0
0	N/A	Control_Packet	TCP_ACK	NODE-3	NODE-2	NODE-3	ROUTER-1	N/A
0	N/A	Control_Packet	TCP_ACK	NODE-3	NODE-2	ROUTER-1	NODE-2	N/A

Figure 8-31: Packet Trace

Step 2: Click the arrow in the header of the column you want to filter. In the list of text or numbers, uncheck the (Select All) box at the top of the list, and then check the boxes of the items you want to show.

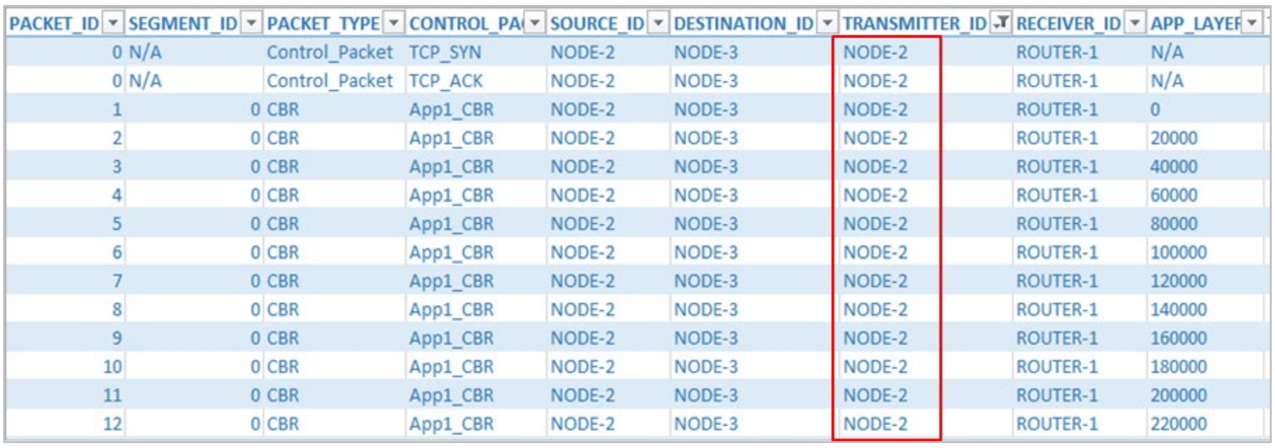
For example, click on arrow of SOURCE_ID and uncheck the "Select all" check box and select NODE 2 then click on OK.

All the rows which are having NODE 2 as source id will be shown below Figure 8-32.



PACKET_ID	SEGMENT_ID	PACKET_TYPE	CONTROL_PA	SOURCE_ID	DESTINATION_ID	TRANSMITTER_ID	RECEIVER_ID	APP_LAYER
0	N/A	Control_Packet	TCP_SYN	NODE-2	NODE-3			N/A
0	N/A	Control_Packet	TCP_SYN	NODE-2	NODE-3			N/A
0	N/A	Control_Packet	TCP_SYNACK	NODE-3	NODE-2			N/A
0	N/A	Control_Packet	TCP_SYNACK	NODE-3	NODE-2			N/A
0	N/A	Control_Packet	TCP_ACK	NODE-2	NODE-3			N/A
0	N/A	Control_Packet	TCP_ACK	NODE-2	NODE-3			N/A
1	0	CBR	App1_CBR	NODE-2	NODE-3			0
1	0	CBR	App1_CBR	NODE-2	NODE-3			0
0	N/A	Control_Packet	TCP_ACK	NODE-3	NODE-2			N/A
0	N/A	Control_Packet	TCP_ACK	NODE-3	NODE-2			N/A
2	0	CBR	App1_CBR	NODE-2	NODE-3			20000
2	0	CBR	App1_CBR	NODE-2	NODE-3			20000
0	N/A	Control_Packet	TCP_ACK	NODE-3	NODE-2			N/A
0	N/A	Control_Packet	TCP_ACK	NODE-3	NODE-2			N/A
3	0	CBR	App1_CBR	NODE-2	NODE-3			40000
3	0	CBR	App1_CBR	NODE-2	NODE-3			40000
0	N/A	Control_Packet	TCP_ACK	NODE-3	NODE-2			N/A
0	N/A	Control_Packet	TCP_ACK	NODE-3	NODE-2			N/A
4	0	CBR	App1_CBR	NODE-2	NODE-3			60000
4	0	CBR	App1_CBR	NODE-2	NODE-3			60000
0	N/A	Control_Packet	TCP_ACK	NODE-3	NODE-2	NODE-3	ROUTER-1	N/A

Figure 8-32: Select Transmitter ID arrow mark in the header in packet trace



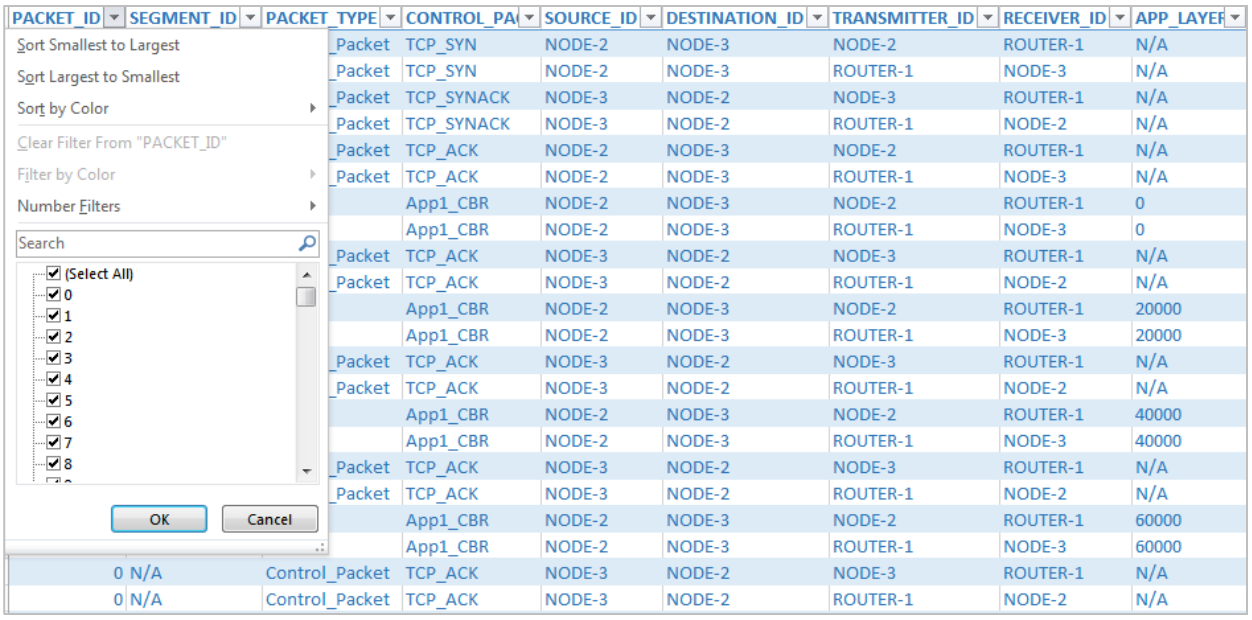
PACKET_ID	SEGMENT_ID	PACKET_TYPE	CONTROL_PA	SOURCE_ID	DESTINATION_ID	TRANSMITTER_ID	RECEIVER_ID	APP_LAYER
0	N/A	Control_Packet	TCP_SYN	NODE-2	NODE-3	NODE-2	ROUTER-1	N/A
0	N/A	Control_Packet	TCP_ACK	NODE-2	NODE-3	NODE-2	ROUTER-1	N/A
1	0	CBR	App1_CBR	NODE-2	NODE-3	NODE-2	ROUTER-1	0
2	0	CBR	App1_CBR	NODE-2	NODE-3	NODE-2	ROUTER-1	20000
3	0	CBR	App1_CBR	NODE-2	NODE-3	NODE-2	ROUTER-1	40000
4	0	CBR	App1_CBR	NODE-2	NODE-3	NODE-2	ROUTER-1	60000
5	0	CBR	App1_CBR	NODE-2	NODE-3	NODE-2	ROUTER-1	80000
6	0	CBR	App1_CBR	NODE-2	NODE-3	NODE-2	ROUTER-1	100000
7	0	CBR	App1_CBR	NODE-2	NODE-3	NODE-2	ROUTER-1	120000
8	0	CBR	App1_CBR	NODE-2	NODE-3	NODE-2	ROUTER-1	140000
9	0	CBR	App1_CBR	NODE-2	NODE-3	NODE-2	ROUTER-1	160000
10	0	CBR	App1_CBR	NODE-2	NODE-3	NODE-2	ROUTER-1	180000
11	0	CBR	App1_CBR	NODE-2	NODE-3	NODE-2	ROUTER-1	200000
12	0	CBR	App1_CBR	NODE-2	NODE-3	NODE-2	ROUTER-1	220000

Figure 8-33: Filter Transmitter ID to NODE 2 in packet trace

Typically, filters can be set to observe **"Errored/Collided/Successful"**packets, packets of destination and packets of source.

8.4.3 Observing packet flow in the Network through packet trace file

Open the packet trace file, Click the arrow ▾ in the header of the column PACKET_ID and uncheck the "Select all" check box and select the packet id which you want to observe, for example 1, and then click on OK.



PACKET_ID	SEGMENT_ID	PACKET_TYPE	CONTROL_PA	SOURCE_ID	DESTINATION_ID	TRANSMITTER_ID	RECEIVER_ID	APP_LAYER
		Packet	TCP_SYN	NODE-2	NODE-3	NODE-2	ROUTER-1	N/A
		Packet	TCP_SYN	NODE-2	NODE-3	ROUTER-1	NODE-3	N/A
		Packet	TCP_SYNACK	NODE-3	NODE-2	NODE-3	ROUTER-1	N/A
		Packet	TCP_SYNACK	NODE-3	NODE-2	ROUTER-1	NODE-2	N/A
		Packet	TCP_ACK	NODE-2	NODE-3	NODE-2	ROUTER-1	N/A
		Packet	TCP_ACK	NODE-2	NODE-3	ROUTER-1	NODE-3	N/A
			App1_CBR	NODE-2	NODE-3	NODE-2	ROUTER-1	0
			App1_CBR	NODE-2	NODE-3	ROUTER-1	NODE-3	0
		Packet	TCP_ACK	NODE-3	NODE-2	NODE-3	ROUTER-1	N/A
		Packet	TCP_ACK	NODE-3	NODE-2	ROUTER-1	NODE-2	N/A
			App1_CBR	NODE-2	NODE-3	NODE-2	ROUTER-1	20000
			App1_CBR	NODE-2	NODE-3	ROUTER-1	NODE-3	20000
		Packet	TCP_ACK	NODE-3	NODE-2	NODE-3	ROUTER-1	N/A
		Packet	TCP_ACK	NODE-3	NODE-2	ROUTER-1	NODE-2	N/A
			App1_CBR	NODE-2	NODE-3	NODE-2	ROUTER-1	40000
			App1_CBR	NODE-2	NODE-3	ROUTER-1	NODE-3	40000
		Packet	TCP_ACK	NODE-3	NODE-2	NODE-3	ROUTER-1	N/A
		Packet	TCP_ACK	NODE-3	NODE-2	ROUTER-1	NODE-2	N/A
			App1_CBR	NODE-2	NODE-3	NODE-2	ROUTER-1	60000
			App1_CBR	NODE-2	NODE-3	ROUTER-1	NODE-3	60000
0	N/A	Control_Packet	TCP_ACK	NODE-3	NODE-2	NODE-3	ROUTER-1	N/A
0	N/A	Control_Packet	TCP_ACK	NODE-3	NODE-2	ROUTER-1	NODE-2	N/A

Figure 8-34: Select Packet ID arrow mark in the header in packet trace

Scenario is as shown below Figure 8-35 and traffic flow is from Wired Node 2 to Wired Node 3.

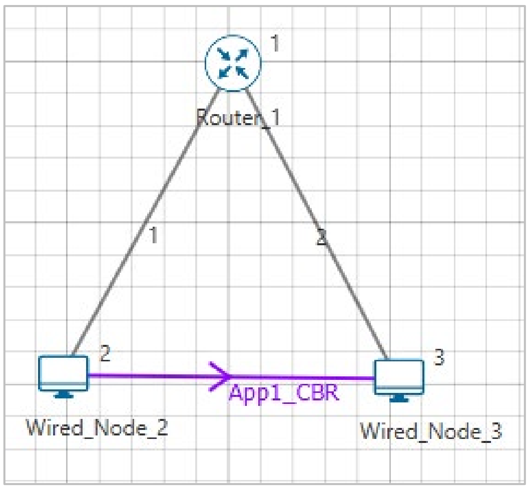
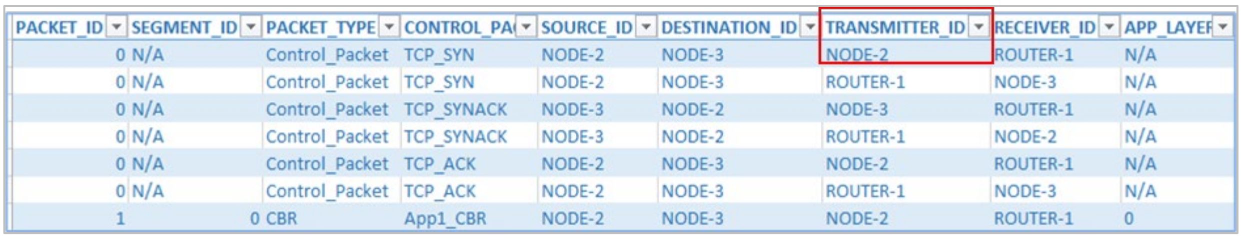


Figure 8-35: Traffic flow is from Wired Node 2 to Wired Node 3

Flow of packet 1 can be observed from the packet trace as shown below Figure 8-36.



PACKET_ID	SEGMENT_ID	PACKET_TYPE	CONTROL_PA	SOURCE_ID	DESTINATION_ID	TRANSMITTER_ID	RECEIVER_ID	APP_LAYEF
0	N/A	Control_Packet	TCP_SYN	NODE-2	NODE-3	NODE-2	ROUTER-1	N/A
0	N/A	Control_Packet	TCP_SYN	NODE-2	NODE-3	ROUTER-1	NODE-3	N/A
0	N/A	Control_Packet	TCP_SYNACK	NODE-3	NODE-2	NODE-3	ROUTER-1	N/A
0	N/A	Control_Packet	TCP_SYNACK	NODE-3	NODE-2	ROUTER-1	NODE-2	N/A
0	N/A	Control_Packet	TCP_ACK	NODE-2	NODE-3	NODE-2	ROUTER-1	N/A
0	N/A	Control_Packet	TCP_ACK	NODE-2	NODE-3	ROUTER-1	NODE-3	N/A
1	0	CBR	App1_CBR	NODE-2	NODE-3	NODE-2	ROUTER-1	0

Figure 8-36: Flow of packet observed in the packet trace

Note: *In the trace file device IDs are shown not device names. Wired Node 1's ID is 2 so it is Shown as NODE-2, Wired Node 2's ID is 3 so it is shown as NODE -3, Router-1' ID is 1 so it is shown as ROUTER-1. Device IDs are shown on the top of the device icon in the above scenario.*

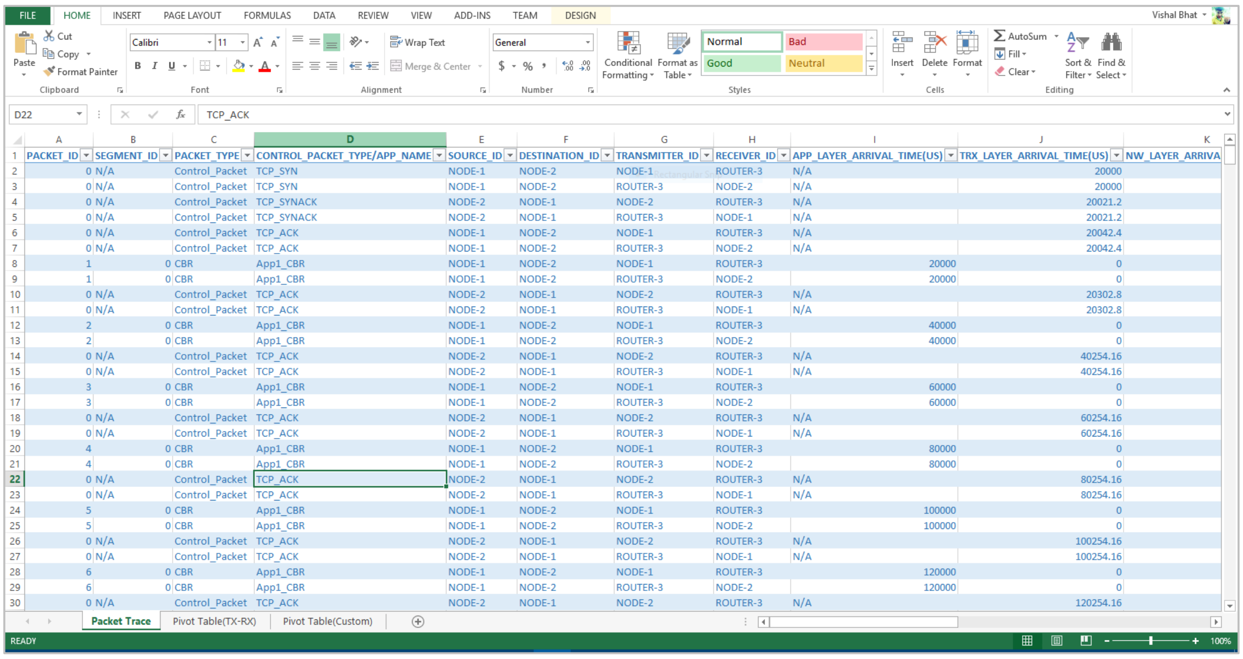
In a scenario source and destinations are fixed but transmitter and receiver are changed. For example, in the above scenario NODE-2 is the source and NODE-3 is the destination, but when NODE- 2 sending the packet to the ROUTER-1 then NODE-2 is the transmitter and ROUTER-1 is the receiver. When ROUTER-1 sending the packet to the NODE-3, ROUTER-1 is the transmitter and NODE-3 is the receiver.

8.4.4 Analysing Packet Trace using Pivot Tables

NetSim Packet trace is saved as a spread sheet. Packet Trace can be converted to an Excel table to make the management and analysis of data easier. A table typically contains related data in a series of worksheet rows and columns that have been formatted as a table. By using the table features, you can then manage the data in the table rows and columns independently from the data in other rows and columns on the worksheet.

PivotTables are a great way to summarize, analyze, explore, and present your data, and you can create them with just a few clicks. PivotTables are highly flexible and can be quickly adjusted depending on how you need to display your results. You can also create Pivot Charts based on PivotTables that will automatically update when your PivotTables do.

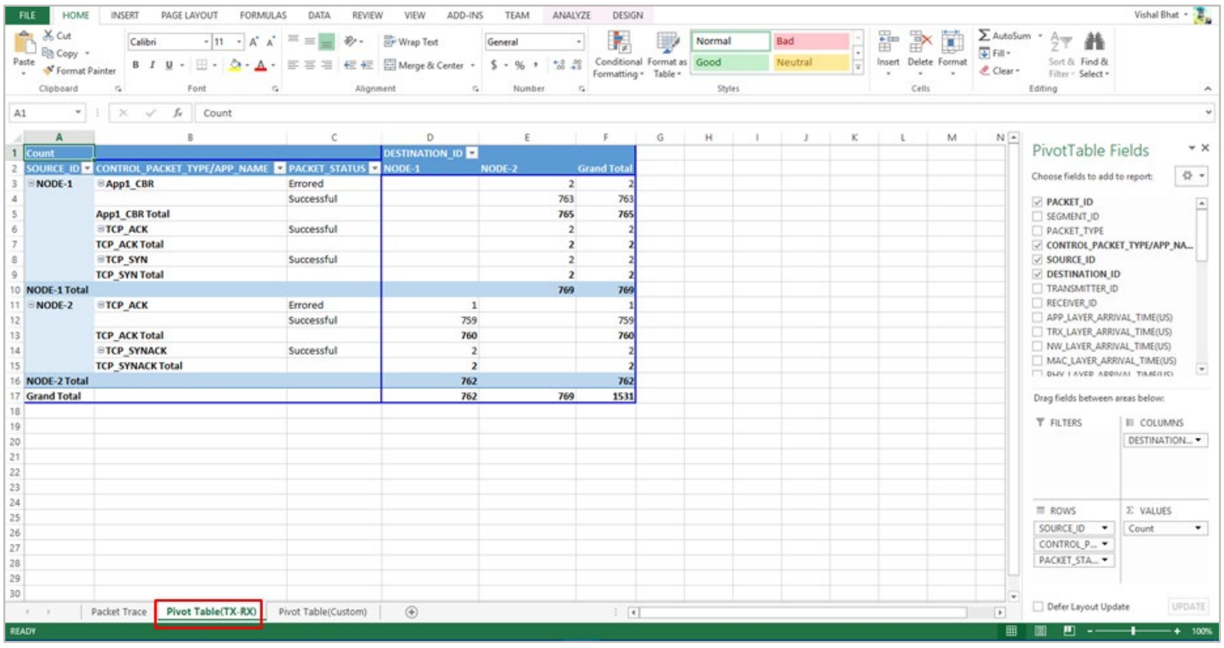
If you enable packet trace, Open Packet Trace link present in the Simulation Results Window can be used to load the packet Trace file in MS-Excel. Formats the spread sheet as a table for convenient analysis.



PACKET_ID	SEGMENT_ID	PACKET_TYPE	CONTROL_PACKET_TYPE/APP_NAME	SOURCE_ID	DESTINATION_ID	TRANSMITTER_ID	RECEIVER_ID	APP_LAYER_ARRIVAL_TIME(US)	TRX_LAYER_ARRIVAL_TIME(US)	NW_LAYER_ARRIVA
0	N/A	Control_Packet	TCP_SYN	NODE-1	NODE-2	NODE-1	ROUTER-3	N/A	20000	
0	N/A	Control_Packet	TCP_SYN	NODE-1	NODE-2	ROUTER-3	NODE-2	N/A	20000	
0	N/A	Control_Packet	TCP_SYNACK	NODE-2	NODE-1	NODE-2	ROUTER-3	N/A	20021.2	
0	N/A	Control_Packet	TCP_SYNACK	NODE-2	NODE-1	ROUTER-3	NODE-1	N/A	20021.2	
0	N/A	Control_Packet	TCP_ACK	NODE-1	NODE-2	NODE-1	ROUTER-3	N/A	20042.4	
0	N/A	Control_Packet	TCP_ACK	NODE-1	NODE-2	ROUTER-3	NODE-2	N/A	20042.4	
1	0	CBR	App1_CBR	NODE-1	NODE-2	NODE-1	ROUTER-3	20000	0	
1	0	CBR	App1_CBR	NODE-1	NODE-2	ROUTER-3	NODE-2	20000	0	
0	N/A	Control_Packet	TCP_ACK	NODE-2	NODE-1	NODE-2	ROUTER-3	N/A	20302.8	
0	N/A	Control_Packet	TCP_ACK	NODE-2	NODE-1	ROUTER-3	NODE-1	N/A	20302.8	
2	0	CBR	App1_CBR	NODE-1	NODE-2	NODE-1	ROUTER-3	40000	0	
2	0	CBR	App1_CBR	NODE-1	NODE-2	ROUTER-3	NODE-2	40000	0	
0	N/A	Control_Packet	TCP_ACK	NODE-2	NODE-1	NODE-2	ROUTER-3	N/A	40254.16	
0	N/A	Control_Packet	TCP_ACK	NODE-2	NODE-1	ROUTER-3	NODE-1	N/A	40254.16	
3	0	CBR	App1_CBR	NODE-1	NODE-2	NODE-1	ROUTER-3	60000	0	
3	0	CBR	App1_CBR	NODE-1	NODE-2	ROUTER-3	NODE-2	60000	0	
0	N/A	Control_Packet	TCP_ACK	NODE-2	NODE-1	NODE-2	ROUTER-3	N/A	60254.16	
0	N/A	Control_Packet	TCP_ACK	NODE-2	NODE-1	ROUTER-3	NODE-1	N/A	60254.16	
4	0	CBR	App1_CBR	NODE-1	NODE-2	NODE-1	ROUTER-3	80000	0	
4	0	CBR	App1_CBR	NODE-1	NODE-2	ROUTER-3	NODE-2	80000	0	
0	N/A	Control_Packet	TCP_ACK	NODE-2	NODE-1	NODE-2	ROUTER-3	N/A	80254.16	
0	N/A	Control_Packet	TCP_ACK	NODE-2	NODE-1	ROUTER-3	NODE-1	N/A	80254.16	
5	0	CBR	App1_CBR	NODE-1	NODE-2	NODE-1	ROUTER-3	100000	0	
5	0	CBR	App1_CBR	NODE-1	NODE-2	ROUTER-3	NODE-2	100000	0	
0	N/A	Control_Packet	TCP_ACK	NODE-2	NODE-1	NODE-2	ROUTER-3	N/A	100254.16	
0	N/A	Control_Packet	TCP_ACK	NODE-2	NODE-1	ROUTER-3	NODE-1	N/A	100254.16	
6	0	CBR	App1_CBR	NODE-1	NODE-2	NODE-1	ROUTER-3	120000	0	
6	0	CBR	App1_CBR	NODE-1	NODE-2	ROUTER-3	NODE-2	120000	0	
0	N/A	Control_Packet	TCP_ACK	NODE-2	NODE-1	NODE-2	ROUTER-3	N/A	120254.16	

Figure 8-37: Sheet 1 is the packet trace

Sheet 2 of the packet trace file has a pivot table – Pivot Table (TX-RX) automatically populated to analyze the packets that were transmitted and received in the network that was simulated. Further users can modify the table by adding or deleting the column headers.



Count			DESTINATION_ID		
SOURCE_ID	CONTROL_PACKET_TYPE/APP_NAME	PACKET_STATUS	NODE-1	NODE-2	Grand Total
NODE-1	App1_CBR	Errored		2	2
		Successful		763	763
	App1_CBR Total			**765**	**765**
	TCP_ACK	Successful		2	2
	TCP_ACK Total			**2**	**2**
	TCP_SYN	Successful		2	2
	TCP_SYN Total			**2**	**2**
NODE-1 Total				**769**	**769**
NODE-2	TCP_ACK	Errored	1		1
		Successful	759		759
	TCP_ACK Total		**760**		**760**
	TCP_SYNACK	Successful	2		2
	TCP_SYNACK Total		**2**		**2**
NODE-2 Total			**762**		**762**
Grand Total			**762**	**769**	**1531**

PivotTable Fields

Choose fields to add to report:

- [✓] PACKET_ID
- [ ] SEGMENT_ID
- [ ] PACKET_TYPE
- [✓] CONTROL_PACKET_TYPE/APP_NA...
- [✓] SOURCE_ID
- [✓] DESTINATION_ID
- [ ] TRANSMITTER_ID
- [ ] RECEIVER_ID
- [ ] APP_LAYER_ARRIVAL_TIME(US)
- [ ] TRX_LAYER_ARRIVAL_TIME(US)
- [ ] NW_LAYER_ARRIVAL_TIME(US)
- [ ] MAC_LAYER_ARRIVAL_TIME(US)

Drag fields between areas below:

FILTERS

COLUMNS: DESTINATION...

ROWS: SOURCE_ID, CONTROL_P..., PACKET_STA...

VALUES: Count

Defer Layout Update

Figure 8-38: Sheet 2 of the packet trace file has a pivot table

Sheet 3 of the packet trace has a blank pivot table – **Pivot Table (Custom)** which can be used to create additional pivot tables from scratch.

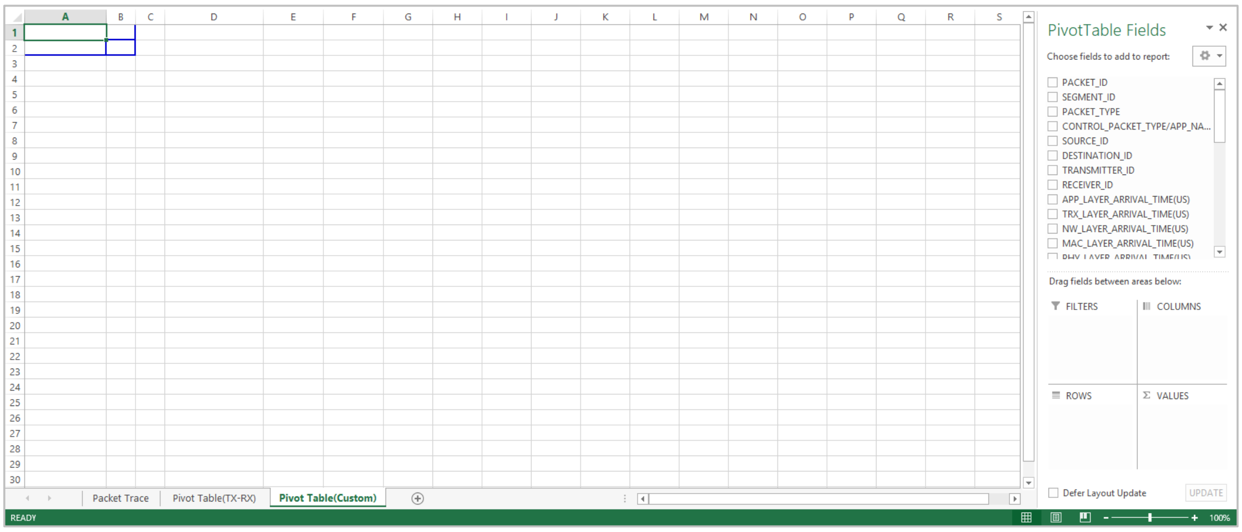
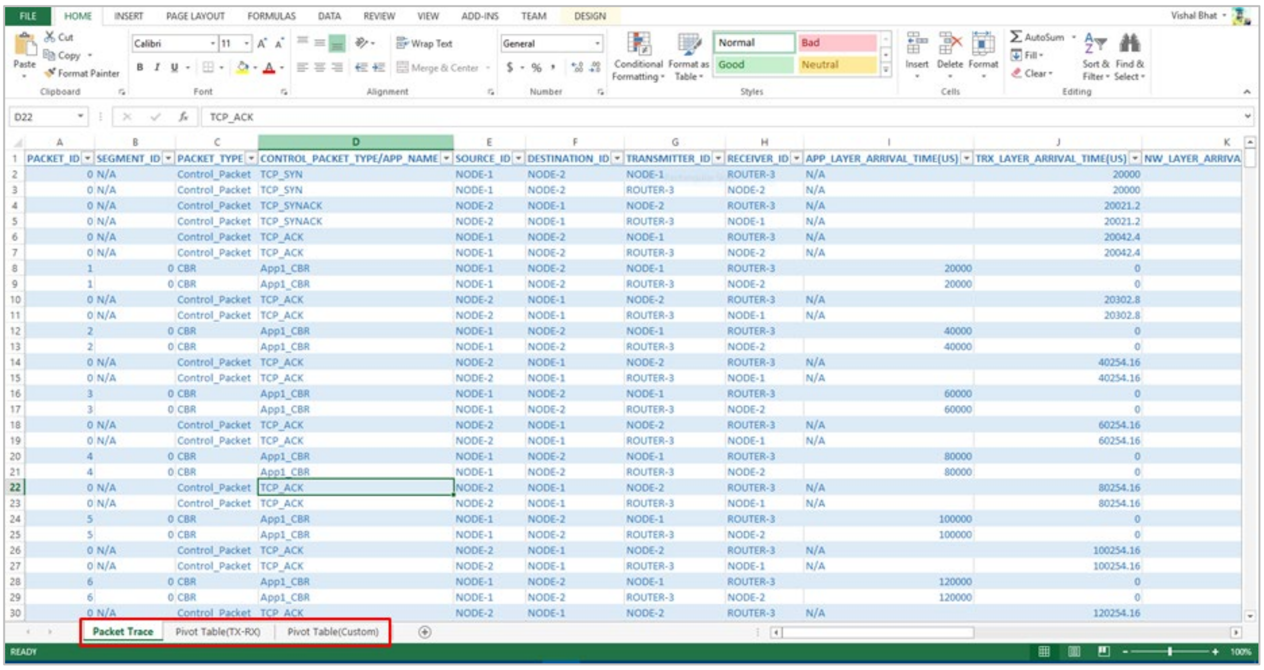


Figure 8-39: Sheet 3 of the packet trace file has a blank pivot table

Steps to analyse the packet trace using pivot tables

Step 1: Click on Packet Trace in the result dashboard, you can find 3 sheets will be created i.e. Packet Trace, Pivot Table (TX-RX), Pivot Table (Custom)



D22 TCP_ACK

	PACKET_ID	SEGMENT_ID	PACKET_TYPE	CONTROL_PACKET_TYPE/APP_NAME	SOURCE_ID	DESTINATION_ID	TRANSMITTER_ID	RECEIVER_ID	APP_LAYER_ARRIVAL_TIME(US)	TRX_LAYER_ARRIVAL_TIME(US)	NW_LAYER_ARRIVA
2	0	N/A	Control_Packet	TCP_SYN	NODE-1	NODE-2	NODE-1	ROUTER-3	N/A	20000	
3	0	N/A	Control_Packet	TCP_SYN	NODE-1	NODE-2	ROUTER-3	NODE-2	N/A	20000	
4	0	N/A	Control_Packet	TCP_SYNACK	NODE-2	NODE-1	NODE-2	ROUTER-3	N/A	20021.2	
5	0	N/A	Control_Packet	TCP_SYNACK	NODE-2	NODE-1	ROUTER-3	NODE-1	N/A	20021.2	
6	0	N/A	Control_Packet	TCP_ACK	NODE-1	NODE-2	NODE-1	ROUTER-3	N/A	20042.4	
7	0	N/A	Control_Packet	TCP_ACK	NODE-1	NODE-2	ROUTER-3	NODE-2	N/A	20042.4	
8	1	0	CBR	App1_CBR	NODE-1	NODE-2	NODE-1	ROUTER-3	20000	0	
9	1	0	CBR	App1_CBR	NODE-1	NODE-2	ROUTER-3	NODE-2	20000	0	
10	0	N/A	Control_Packet	TCP_ACK	NODE-2	NODE-1	NODE-2	ROUTER-3	N/A	20302.8	
11	0	N/A	Control_Packet	TCP_ACK	NODE-2	NODE-1	ROUTER-3	NODE-1	N/A	20302.8	
12	2	0	CBR	App1_CBR	NODE-1	NODE-2	NODE-1	ROUTER-3	40000	0	
13	2	0	CBR	App1_CBR	NODE-1	NODE-2	ROUTER-3	NODE-2	40000	0	
14	0	N/A	Control_Packet	TCP_ACK	NODE-2	NODE-1	NODE-2	ROUTER-3	N/A	40254.16	
15	0	N/A	Control_Packet	TCP_ACK	NODE-2	NODE-1	ROUTER-3	NODE-1	N/A	40254.16	
16	3	0	CBR	App1_CBR	NODE-1	NODE-2	NODE-1	ROUTER-3	60000	0	
17	3	0	CBR	App1_CBR	NODE-1	NODE-2	ROUTER-3	NODE-2	60000	0	
18	0	N/A	Control_Packet	TCP_ACK	NODE-2	NODE-1	NODE-2	ROUTER-3	N/A	60254.16	
19	0	N/A	Control_Packet	TCP_ACK	NODE-2	NODE-1	ROUTER-3	NODE-1	N/A	60254.16	
20	4	0	CBR	App1_CBR	NODE-1	NODE-2	NODE-1	ROUTER-3	80000	0	
21	4	0	CBR	App1_CBR	NODE-1	NODE-2	ROUTER-3	NODE-2	80000	0	
22	0	N/A	Control_Packet	TCP_ACK	NODE-2	NODE-1	NODE-2	ROUTER-3	N/A	80254.16	
23	0	N/A	Control_Packet	TCP_ACK	NODE-2	NODE-1	ROUTER-3	NODE-1	N/A	80254.16	
24	5	0	CBR	App1_CBR	NODE-1	NODE-2	NODE-1	ROUTER-3	100000	0	
25	5	0	CBR	App1_CBR	NODE-1	NODE-2	ROUTER-3	NODE-2	100000	0	
26	0	N/A	Control_Packet	TCP_ACK	NODE-2	NODE-1	NODE-2	ROUTER-3	N/A	100254.16	
27	0	N/A	Control_Packet	TCP_ACK	NODE-2	NODE-1	ROUTER-3	NODE-1	N/A	100254.16	
28	6	0	CBR	App1_CBR	NODE-1	NODE-2	NODE-1	ROUTER-3	120000	0	
29	6	0	CBR	App1_CBR	NODE-1	NODE-2	ROUTER-3	NODE-2	120000	0	
30	0	N/A	Control_Packet	TCP_ACK	NODE-2	NODE-1	NODE-2	ROUTER-3	N/A	120254.16	

Figure 8-40: Packet Trace, Pivot Table (TX-RX), Pivot Table (Custom) in packet trace

Step 2: Click on Pivot Table (Custom) to create your own pivot table.

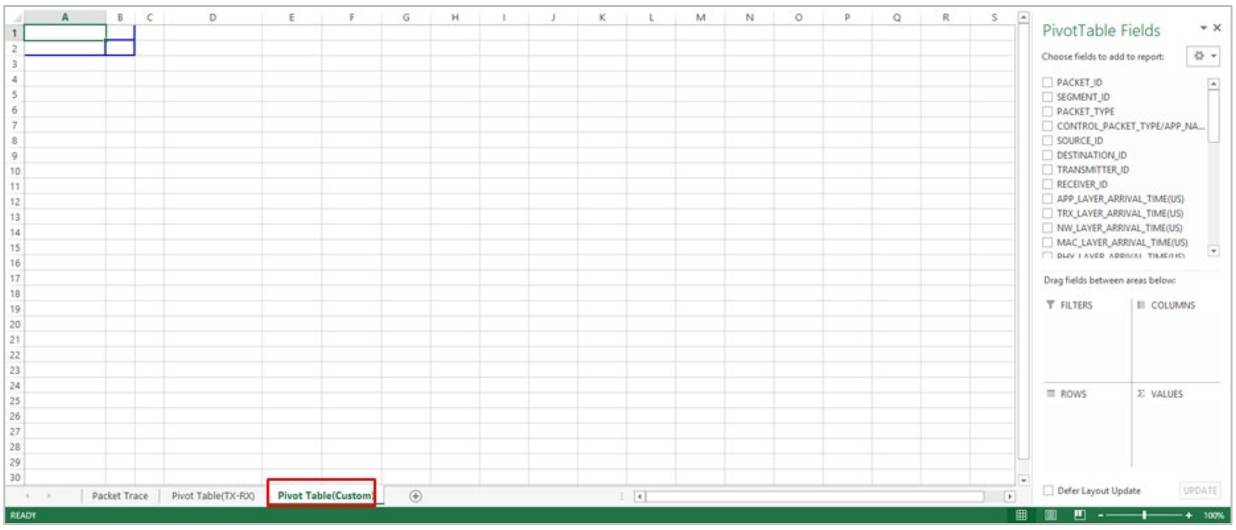


Figure 8-41: Select Blank Pivot Table (Custom) to create your own pivot table

Once you open the sheet PivotTable (Custom), you will need to decide which **fields** to add. Each field is simply a **column header** from the source data. In the **PivotTable Field List**, check the box for each field you want to add.

8.4.5 Packet Transmitted / Received Analysis

- If you want to analyze packets sent from all sources to all destinations, then check SOURCE_ID, DESTINATION_ID and CONTROL_PACKET_TYPE/APP_NAME as shown below Figure 8-42.

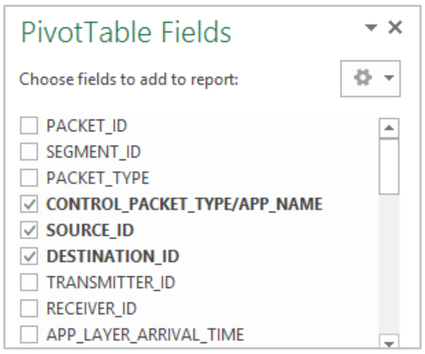
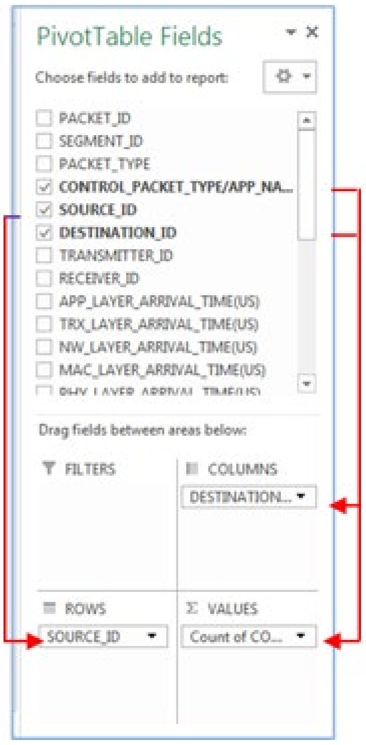


Figure 8-42: Select the check box of SOURCE_ID, DESTINATION_ID and CONTROL_PACKET_TYPE/APP_NAME in PivotTable Fields

- The selected fields will be added to one of the four areas below the Field List. Click SOURCE_ID, hold it and drag to the ROW field. Similarly, DESTINATION_ID to COLUMNS and CONTROL_PACKET_TYPE/APP_NAME to VALUES.



PivotTable Fields

Choose fields to add to report:

- ☐ PACKET_ID
- ☐ SEGMENT_ID
- ☐ PACKET_TYPE
- ✓ CONTROL_PACKET_TYPE/APP_NA...
- ✓ SOURCE_ID
- ✓ DESTINATION_ID
- ☐ TRANSMITTER_ID
- ☐ RECEIVER_ID
- ☐ APP_LAYER_ARRIVAL_TIME(US)
- ☐ TRX_LAYER_ARRIVAL_TIME(US)
- ☐ NW_LAYER_ARRIVAL_TIME(US)
- ☐ MAC_LAYER_ARRIVAL_TIME(US)
- ☐ PHY_LAYER_ARRIVAL_TIME(US)

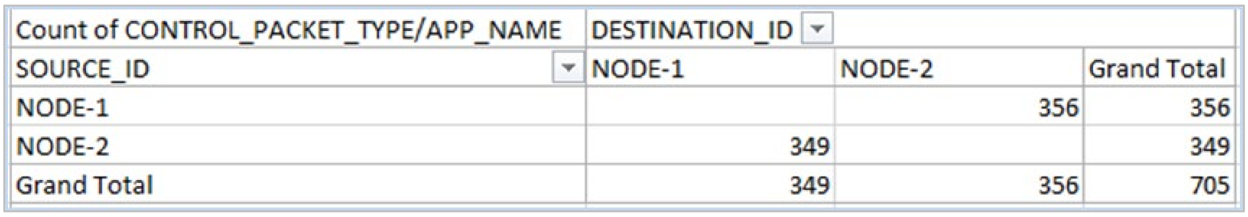
Drag fields between areas below:

FILTERS	COLUMNS
	DESTINATION...

ROWS	VALUES
SOURCE_ID	Count of CO...

Figure 8-43: Selected fields add to one of the four areas below the Field List

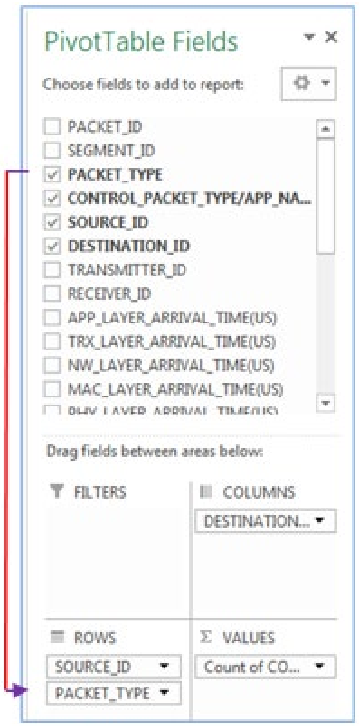
- The PivotTable will calculate and summarize the selected fields. In this example, the PivotTable shows the packets sent from all sources to all destinations.



Count of CONTROL_PACKET_TYPE/APP_NAME	DESTINATION_ID		
SOURCE_ID	NODE-1	NODE-2	Grand Total
NODE-1		356	356
NODE-2	349		349
Grand Total	349	356	705

Figure 8-44: PivotTable Created with selected fields

- The above example shows all the packets which including data packets and control packets.
- If you wish to know how many Data and how many were control packets then, check the PACKET_TYPE and drag it to the ROWS field as shown below Figure 8-45.



PivotTable Fields

Choose fields to add to report:

- [ ] PACKET_ID
- [ ] SEGMENT_ID
- [x] PACKET_TYPE
- [x] CONTROL_PACKET_TYPE/APP_NA...
- [x] SOURCE_ID
- [x] DESTINATION_ID
- [ ] TRANSMITTER_ID
- [ ] RECEIVER_ID
- [ ] APP_LAYER_ARRIVAL_TIME(US)
- [ ] TRX_LAYER_ARRIVAL_TIME(US)
- [ ] NW_LAYER_ARRIVAL_TIME(US)
- [ ] MAC_LAYER_ARRIVAL_TIME(US)

Drag fields between areas below:

FILTERS

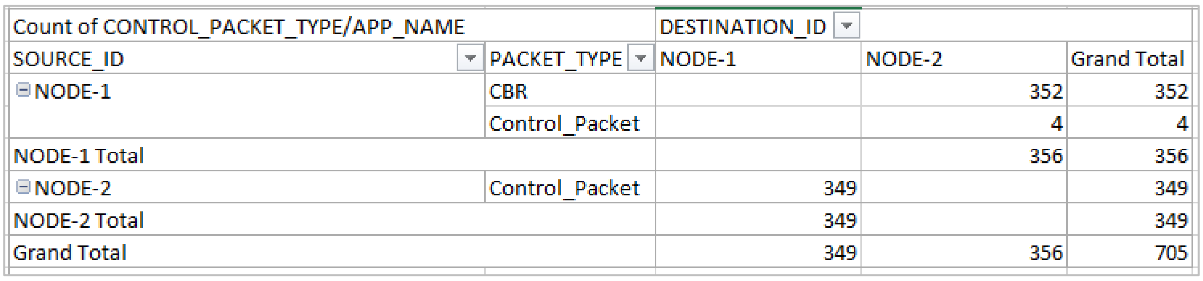
COLUMNS: DESTINATION...

ROWS: SOURCE_ID, PACKET_TYPE

VALUES: Count of CO...

Figure 8-45: Select the PACKET_TYPE Check Box and drag it to the ROWS fields

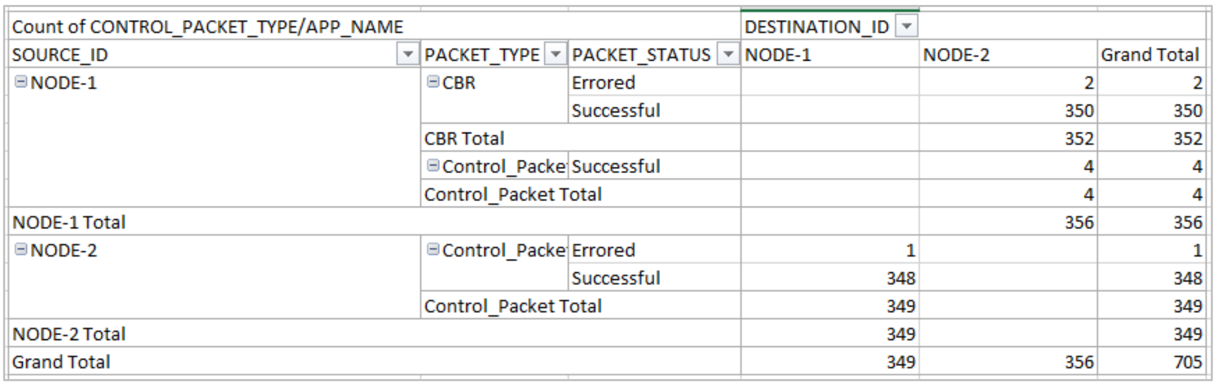
- This will look like



Count of CONTROL_PACKET_TYPE/APP_NAME		DESTINATION_ID		
SOURCE_ID	PACKET_TYPE	NODE-1	NODE-2	Grand Total
⊟NODE-1	CBR		352	352
	Control_Packet		4	4
NODE-1 Total			356	356
⊟NODE-2	Control_Packet	349		349
NODE-2 Total		349		349
Grand Total		349	356	705

Figure 8-46: PivotTable Created with Packet Type

- Further, if you wish to know how many packets got errored and how many were successful, check the PACKET_STATUS field and drag it to the ROWS field.



Count of CONTROL_PACKET_TYPE/APP_NAME			DESTINATION_ID		
SOURCE_ID	PACKET_TYPE	PACKET_STATUS	NODE-1	NODE-2	Grand Total
⊟NODE-1	⊟CBR	Errored		2	2
		Successful		350	350
	CBR Total			352	352
	⊟Control_Packe	Successful		4	4
	Control_Packet Total			4	4
NODE-1 Total				356	356
⊟NODE-2	⊟Control_Packe	Errored	1		1
		Successful	348		348
	Control_Packet Total		349		349
NODE-2 Total			349		349
Grand Total			349	356	705

Figure 8-47: PivotTable Created with Packet Status

8.4.6 Delay analysis

We explain this using a packet trace generated per the following network scenario.

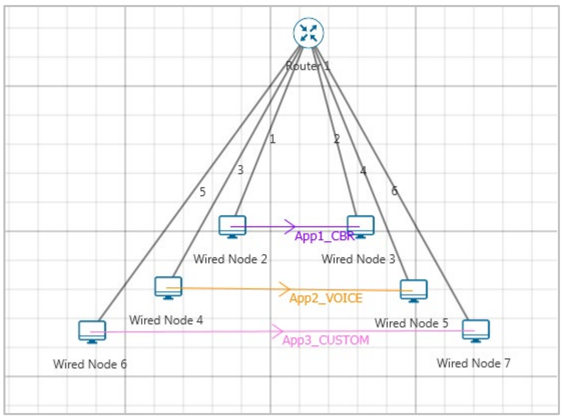


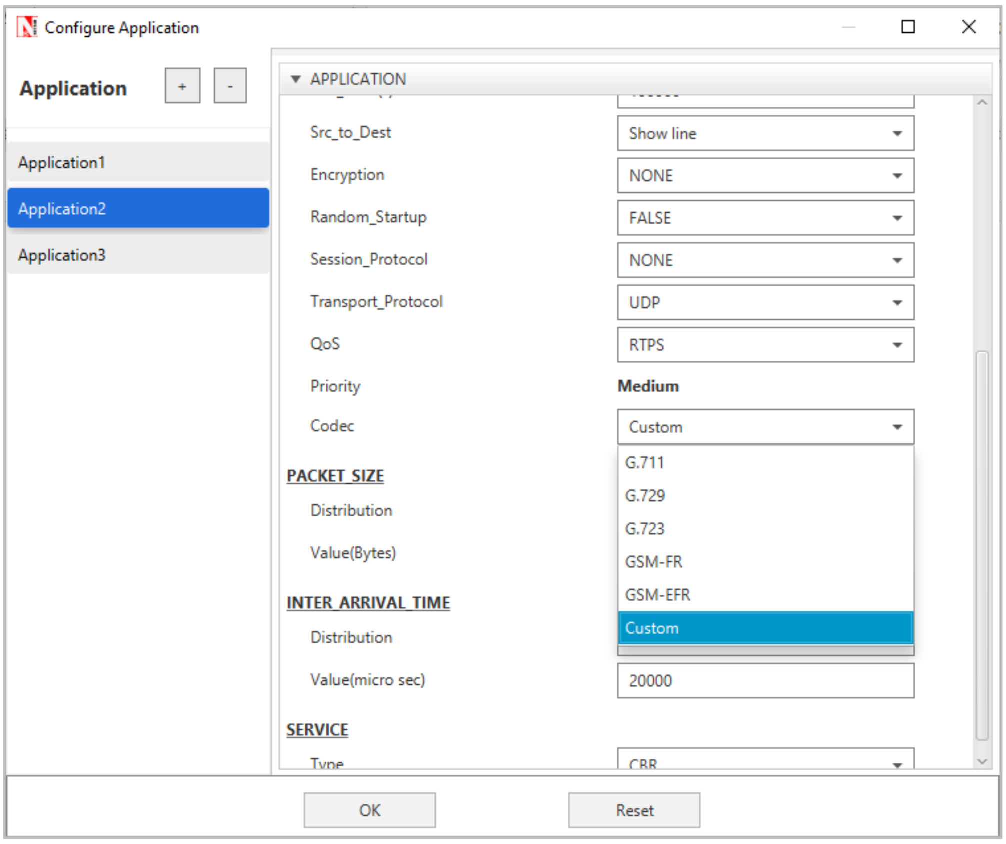
Figure 8-48: Network Topology with different application

Create a network scenario with 1 router and 6 wired nodes. Create 3 applications as per the following Table 8-2.

Application Type	Source Id	Destination Id	Transport Protocol	Packet Size (Bytes)	Inter arrival time (µs)
CBR	2	3	TCP	1460	20000
VOICE	4	5	UDP	1500	20000
CUSTOM	6	7	TCP	1200	20000

Table 8-2: Application Properties

Note: Users need to select Codec as **CUSTOM** for voice application as shown in the below screenshot Figure 8-49.



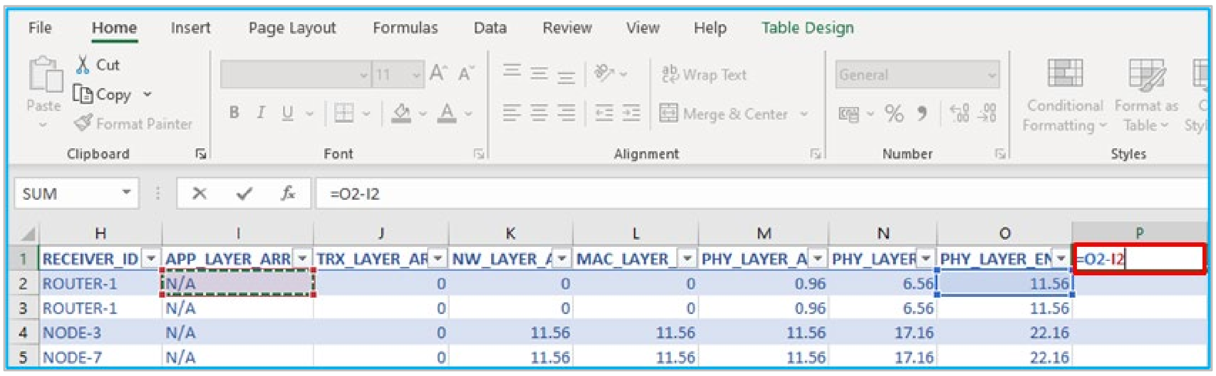
Src_to_Dest	Show line
Encryption	NONE
Random_Startup	FALSE
Session_Protocol	NONE
Transport_Protocol	UDP
QoS	RTPS
Priority	Medium
Codec	Custom
PACKET_SIZE	
Distribution	
Value(Bytes)	
INTER_ARRIVAL_TIME	
Distribution	
Value(micro sec)	20000
SERVICE	
Type	CBR

Figure 8-49: Application properties Window

Enable Packet Trace and simulate the scenario for 10 seconds. Open packet trace and perform the following steps:

- Insert a column after PHY_LAYER_END_TIME, then select the whole column and calculate delay for each and every packet by using the formula.

  =PHY_LAYER_END_TIME – APPLICATION_LAYER_ARRIVAL_TIME

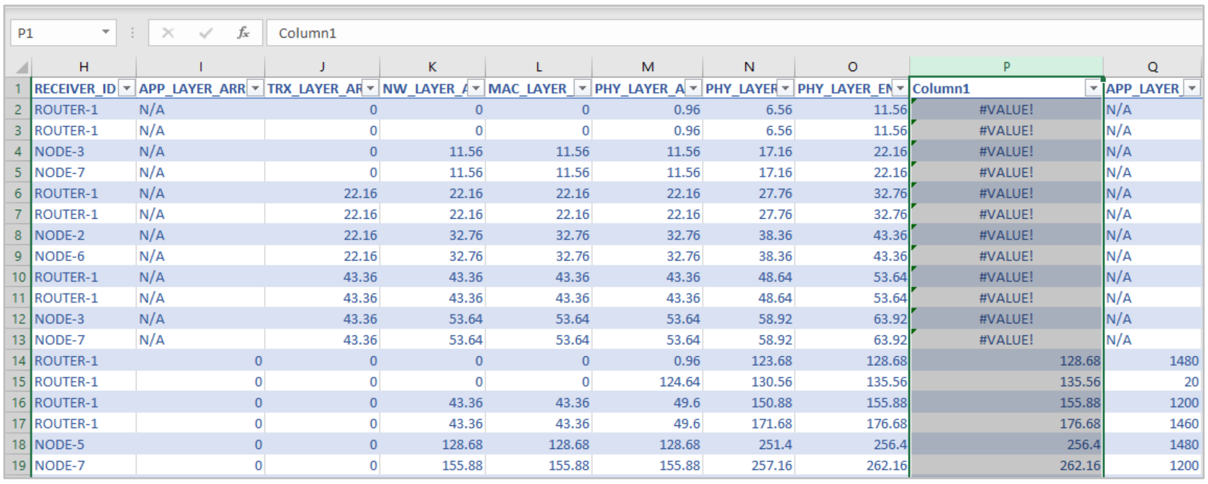


SUM | =O2-I2

	H	I	J	K	L	M	N	O	P
1	RECEIVER_ID	APP_LAYER_ARR	TRX_LAYER_AR	NW_LAYER_A	MAC_LAYER_	PHY_LAYER_A	PHY_LAYER	PHY_LAYER_EN	=O2-I2
2	ROUTER-1	N/A	0	0	0	0.96	6.56	11.56	
3	ROUTER-1	N/A	0	0	0	0.96	6.56	11.56	
4	NODE-3	N/A	0	11.56	11.56	11.56	17.16	22.16	
5	NODE-7	N/A	0	11.56	11.56	11.56	17.16	22.16	

Figure 8-50: Calculate delay in packet trace using PHY_LAYER_END_TIME – APPLICATION_LAYER_ARRIVAL_TIME then Press CTRL + ENTER

- Then Press CTRL + ENTER. This will calculate delay for the whole column shown below.

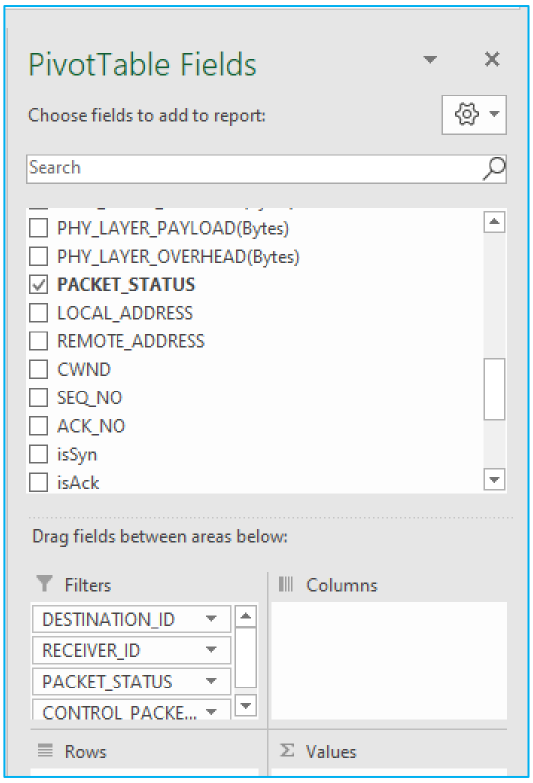


P1 | Column1

	H	I	J	K	L	M	N	O	P	Q
1	RECEIVER_ID	APP_LAYER_ARR	TRX_LAYER_AR	NW_LAYER_A	MAC_LAYER_	PHY_LAYER_A	PHY_LAYER	PHY_LAYER_EN	Column1	APP_LAYER
2	ROUTER-1	N/A	0	0	0	0.96	6.56	11.56	#VALUE!	N/A
3	ROUTER-1	N/A	0	0	0	0.96	6.56	11.56	#VALUE!	N/A
4	NODE-3	N/A	0	11.56	11.56	11.56	17.16	22.16	#VALUE!	N/A
5	NODE-7	N/A	0	11.56	11.56	11.56	17.16	22.16	#VALUE!	N/A
6	ROUTER-1	N/A	22.16	22.16	22.16	22.16	27.76	32.76	#VALUE!	N/A
7	ROUTER-1	N/A	22.16	22.16	22.16	22.16	27.76	32.76	#VALUE!	N/A
8	NODE-2	N/A	22.16	32.76	32.76	32.76	38.36	43.36	#VALUE!	N/A
9	NODE-6	N/A	22.16	32.76	32.76	32.76	38.36	43.36	#VALUE!	N/A
10	ROUTER-1	N/A	43.36	43.36	43.36	43.36	48.64	53.64	#VALUE!	N/A
11	ROUTER-1	N/A	43.36	43.36	43.36	43.36	48.64	53.64	#VALUE!	N/A
12	NODE-3	N/A	43.36	53.64	53.64	53.64	58.92	63.92	#VALUE!	N/A
13	NODE-7	N/A	43.36	53.64	53.64	53.64	58.92	63.92	#VALUE!	N/A
14	ROUTER-1	0	0	0	0	0.96	123.68	128.68	128.68	1480
15	ROUTER-1	0	0	0	0	124.64	130.56	135.56	135.56	20
16	ROUTER-1	0	0	43.36	43.36	49.6	150.88	155.88	155.88	1200
17	ROUTER-1	0	0	43.36	43.36	49.6	171.68	176.68	176.68	1460
18	NODE-5	0	0	128.68	128.68	128.68	251.4	256.4	256.4	1480
19	NODE-7	0	0	155.88	155.88	155.88	257.16	262.16	262.16	1200

Figure 8-51: Calculate delay for the whole column

- Name the column as DELAY.
- Go to Insert->PivotTable and click on OK to create a blank Pivot Table with the newly added column listed under the PivotTable Fields.
- Drag and drop DESTINATION_ID, RECEIVER_ID, PACKET_STATUS and CONTROL_PACKET_TYPE/APP_NAME to FILTERS field shown below Figure 8-51.


PivotTable Fields

Choose fields to add to report:

Search

- [ ] PHY_LAYER_PAYLOAD(Bytes)
- [ ] PHY_LAYER_OVERHEAD(Bytes)
- [x] **PACKET_STATUS**
- [ ] LOCAL_ADDRESS
- [ ] REMOTE_ADDRESS
- [ ] CWND
- [ ] SEQ_NO
- [ ] ACK_NO
- [ ] isSyn
- [ ] isAck

Drag fields between areas below:

Filters: DESTINATION_ID, RECEIVER_ID, PACKET_STATUS, CONTROL_PACKE...

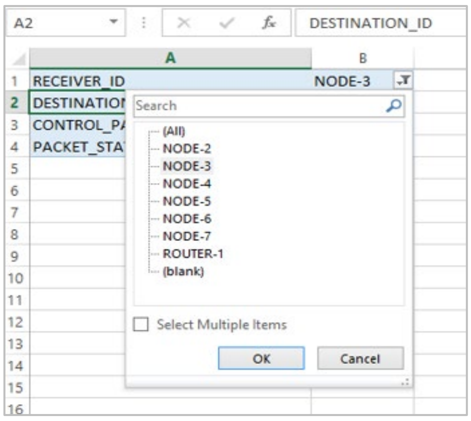
Columns

Rows

Values

Figure 8-52: Added Selected fields to Filter

- Filter RECEIVER_ID to Node-3 by clicking on the drop down and select OK.


A2 DESTINATION_ID

	A	B
1	RECEIVER_ID	NODE-3
2	DESTINATIOI	
3	CONTROL_P/	
4	PACKET_STA	

Search

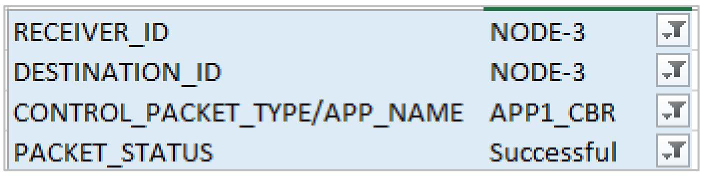
(All)
NODE-2
NODE-3
NODE-4
NODE-5
NODE-6
NODE-7
ROUTER-1
(blank)

☐ Select Multiple Items

OK Cancel

Figure 8-53: Filter RECEIVER_ID to Node-3 by clicking on the drop down

- Similarly filter CONTROL_PACKET_TYPE/APP_NAME to APP1_CBR, DESTINATION_ID to NODE-3 and PACKET_STATUS to Successful



RECEIVER_ID	NODE-3
DESTINATION_ID	NODE-3
CONTROL_PACKET_TYPE/APP_NAME	APP1_CBR
PACKET_STATUS	Successful

Figure 8-54: Similarly filter other fields as per screenshot

- Drag and drop PACKET_ID to ROWS and the Delay value that we calculated earlier to VALUES area.

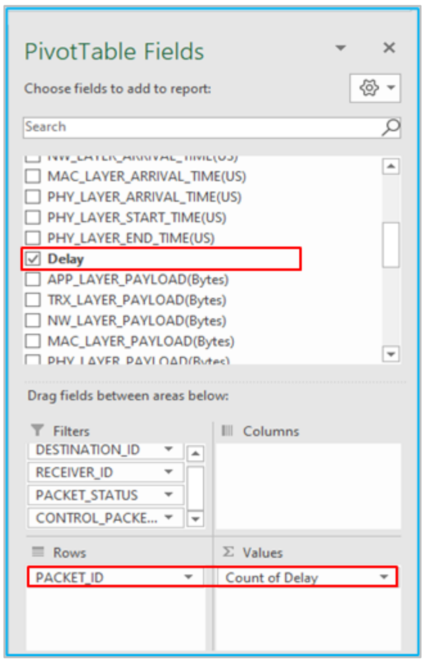


Figure 8-55: Drag and drop DELAY value that we have calculated earlier to ROWS and VALUES field

- Click on Count of DELAY drop down and select Value Field settings, then Select SUM and click on OK.

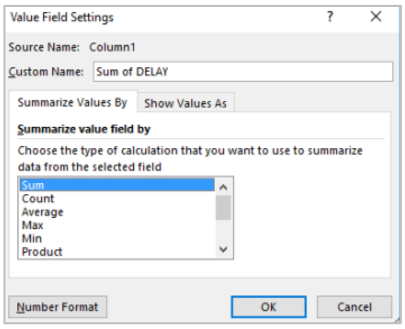


Figure 8-56: Select Count of DELAY drop down and select Value Field settings as SUM

- Again, Drag and drop DELAY to VALUES field.

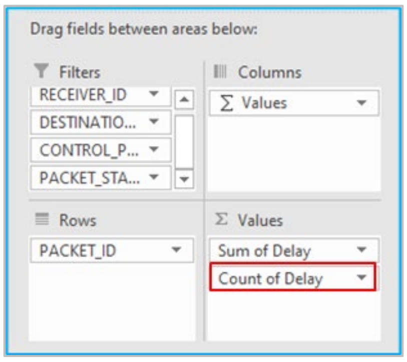
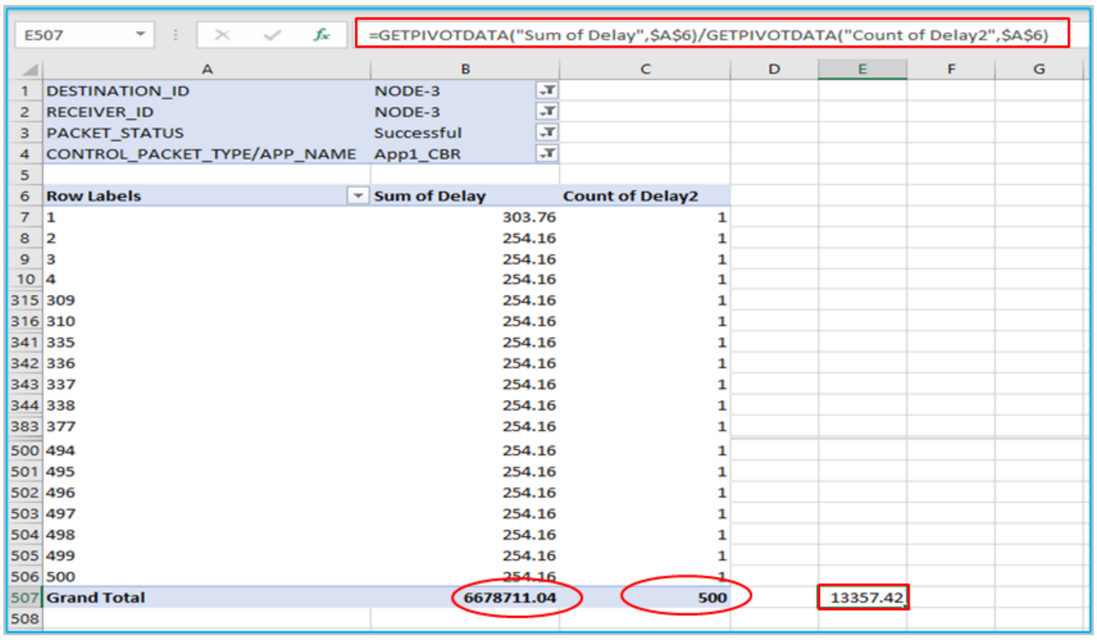


Figure 8-57: Drag and drop DELAY to VALUES field

- Select one cell and calculate the Application Delay, which is the average delay faced by a packet by using the formula.

$$Application\ \ DELAY\ = \frac{Sum\ of\ DELAY\ of\ Successful\ Packets}{Number\ of\ Packets}$$

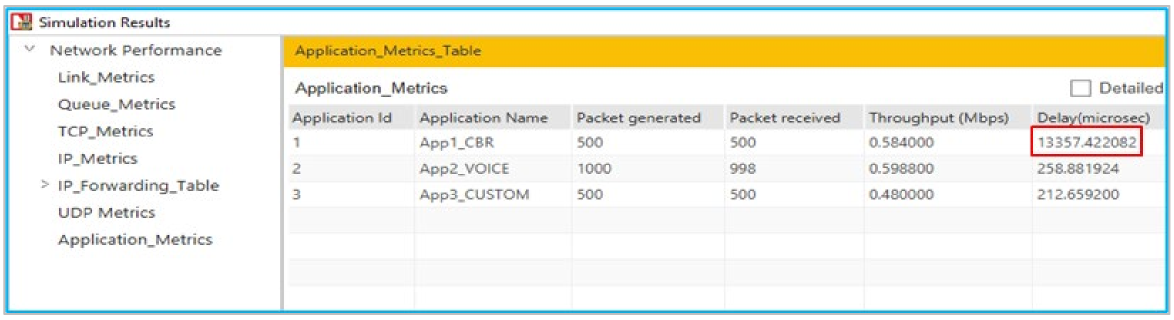


E507 =GETPIVOTDATA("Sum of Delay",A6)/GETPIVOTDATA("Count of Delay2",A6)

	A	B	C	D	E	F	G
1	DESTINATION_ID	NODE-3					
2	RECEIVER_ID	NODE-3					
3	PACKET_STATUS	Successful					
4	CONTROL_PACKET_TYPE/APP_NAME	App1_CBR					
5							
6	Row Labels	Sum of Delay	Count of Delay2				
7	1	303.76	1				
8	2	254.16	1				
9	3	254.16	1				
10	4	254.16	1				
315	309	254.16	1				
316	310	254.16	1				
341	335	254.16	1				
342	336	254.16	1				
343	337	254.16	1				
344	338	254.16	1				
383	377	254.16	1				
500	494	254.16	1				
501	495	254.16	1				
502	496	254.16	1				
503	497	254.16	1				
504	498	254.16	1				
505	499	254.16	1				
506	500	254.16	1				
507	Grand Total	6678711.04	500		13357.42		
508							

Figure 8-58: Calculated the Application Delay in Pivot table

- Compare the obtained value with the DELAY in Application Metrics



Simulation Results

- Network Performance
  - Link_Metrics
  - Queue_Metrics
  - TCP_Metrics
  - IP_Metrics
  - IP_Forwarding_Table
  - UDP Metrics
  - Application_Metrics

Application_Metrics_Table

Application_Metrics ☐ Detailed

Application Id	Application Name	Packet generated	Packet received	Throughput (Mbps)	Delay(microsec)
1	App1_CBR	500	500	0.584000	13357.422082
2	App2_VOICE	1000	998	0.598800	258.881924
3	App3_CUSTOM	500	500	0.480000	212.659200

Figure 8-59: Compare the obtained DELAY with Application Metrics DELAY

- To calculate DELAY for VOICE application, filter DESTINATION_ID to Node-5, RECEIVER_ID to Node-5, CONTROL_PACKET_TYPE/APP_NAME to APP2_VOICE and PACKET_STATUS to Successful
- Similarly calculate and compare DELAY for other applications by following the above procedure.

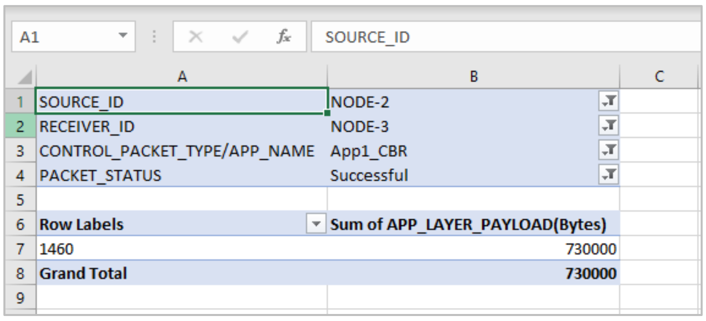
8.4.7 Throughput analysis

To explain how users can perform Throughput Analysis, we have used same network design example as was used for Delay analysis above.

After loading the packet trace switch to sheet Pivot Table (Custom), drag and drop SOURCE_ID, RECEIVER_ID, CONTROL_PACKET_TYPE / APP_NAME and PACKET_STATUS to FILTERS field.

- Similarly drag and drop APP_LAYER_PAYLOAD to ROWS field and VALUES field.

- Filter SOURCE_ID to NODE-2, CONTROL_PACKET_TYPE APP_NAME to APP1_CBR, PACKET_STATUS to Successful and RECEIVER_ID to NODE-3
- Click on Count of APP_LAYER_PAYLOAD drop down and select Value Field settings, then Select Sum and click on OK.
- The pivot table would look like.



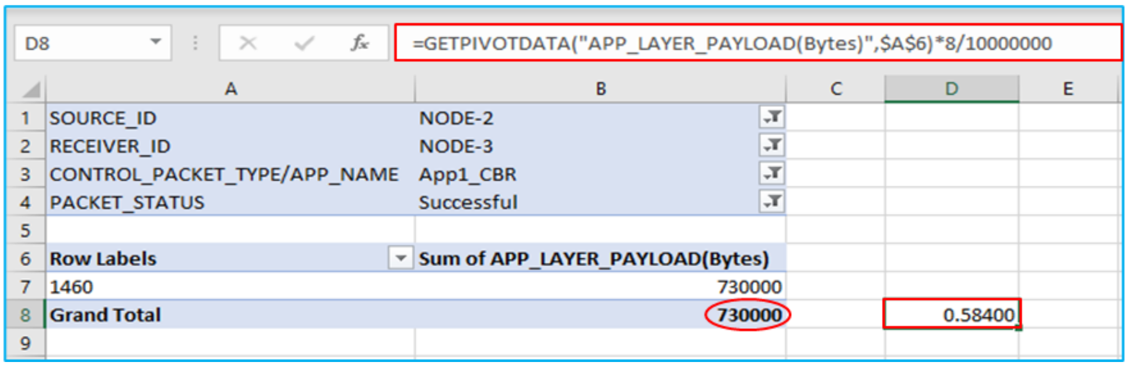
A1 | SOURCE_ID

	A	B
1	SOURCE_ID	NODE-2
2	RECEIVER_ID	NODE-3
3	CONTROL_PACKET_TYPE/APP_NAME	App1_CBR
4	PACKET_STATUS	Successful
5		
6	Row Labels	Sum of APP_LAYER_PAYLOAD(Bytes)
7	1460	730000
8	Grand Total	730000
9		

Figure 8-60: Pivot Table

- Select 1 cell and calculate the throughput by using the formula.

$$Application\ throughput\ (Mbps) = \frac{Sum\ of\ \ Successfully\ Received\ App\ Layer\ Payload\ (Bytes) * 8}{Simulation\ Time\ (\mu s\ )}$$



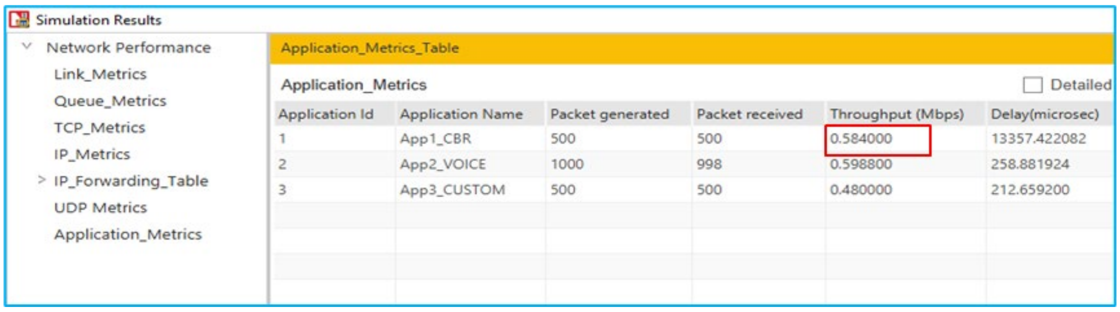
D8 | =GETPIVOTDATA("APP_LAYER_PAYLOAD(Bytes)",A6)*8/10000000

	A	B	C	D	E
1	SOURCE_ID	NODE-2			
2	RECEIVER_ID	NODE-3			
3	CONTROL_PACKET_TYPE/APP_NAME	App1_CBR			
4	PACKET_STATUS	Successful			
5					
6	Row Labels	Sum of APP_LAYER_PAYLOAD(Bytes)			
7	1460	730000			
8	Grand Total	730000		0.58400	
9					

Figure 8-61: Calculate the throughput by using the formula in Pivot Table

EmptyCell=GETPIVOTDATA("APP_LAYER_PAYLOAD(Bytes)",A6,"APP_LAYER_PAYLOAD(Bytes)",1460)*8/10000000

- Now compare the throughput calculated using pivot table with the Application Metrics throughput.



Application Id	Application Name	Packet generated	Packet received	Throughput (Mbps)	Delay(microsec)
1	App1_CBR	500	500	0.584000	13357.422082
2	App2_VOICE	1000	998	0.598800	258.881924
3	App3_CUSTOM	500	500	0.480000	212.659200

Figure 8-62: Compared the calculated throughput using pivot table with the Application Metrics throughput

- To calculate THROUGHPUT for VOICE application, filter SOURCE_ID to Node-4, RECEIVER_ID to Node-5, CONTROL_PACKET_TYPE/APP_NAME to APP2_VOICE and PACKET_STATUS to Successful
- Similarly calculate and compare THROUGHPUT for other applications by following the above procedure.

8.4.8 Plotting with Pivot Charts

In a pivot table, you can create a new field that performs a calculation on the sum of other pivot fields.

- Open Packet Trace, switch to sheet Pivot Table (Custom)
- Drag and drop SOURCE_ID, RECEIVER_ID and PACKET_STATUS to FILTERS field, then CONTROL_PACKET_TYPE/APP_NAME, APP_LAYER_PAYLOAD to ROWS field Figure 8-63.

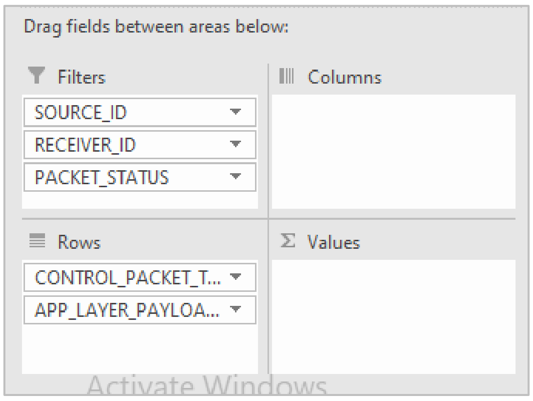


Figure 8-63: Drag and drop Sleeted Fields to one of the four areas below the Field List

- Filter SOURCE_ID to Node 2, Node 4 and Node 6, then RECEIVER_ID to Node 3, Node 5 and Node 7 and PACKET_STATUS to successful

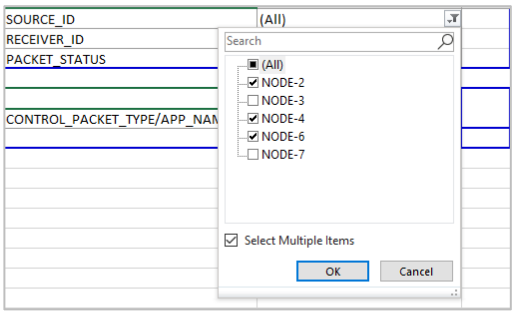
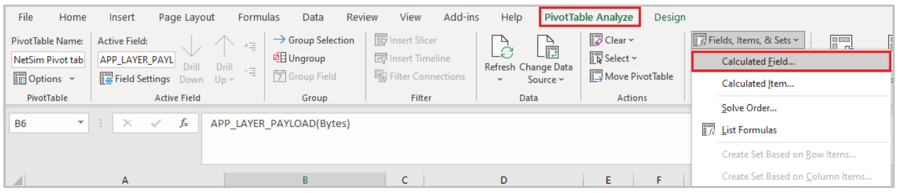


Figure 8-64: Filter Source ID and Destination ID

- Filter CONTROL_PACKET_TYPE/APP_NAME to APP1_CBR, APP2_VOICE and APP3_CUSTOM
- Select a cell in the pivot table, and on the Excel Ribbon, under the PivotTable Tools tab, click the Options tab (PivotTable Analyze tab in Excel 2013).

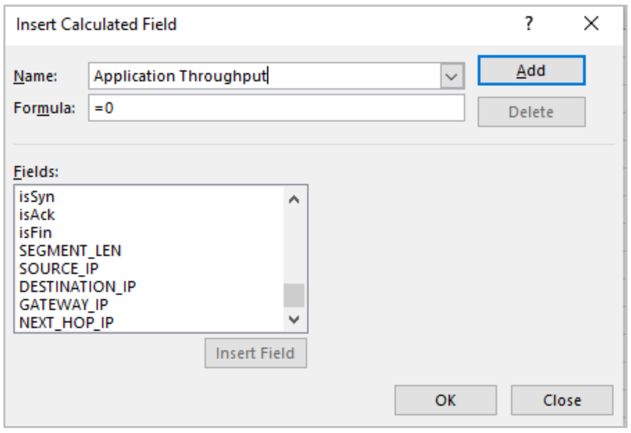
- In the Calculations group, click Fields, Items, & Sets, and then click Calculated Field.



B6 APP_LAYER_PAYLOAD(Bytes)

Figure 8-65: In Calculations group Select Calculated Field

- Type a name for the calculated field, Application Throughput.
- Then click on ADD to save the calculated field.



Name: Application Throughput

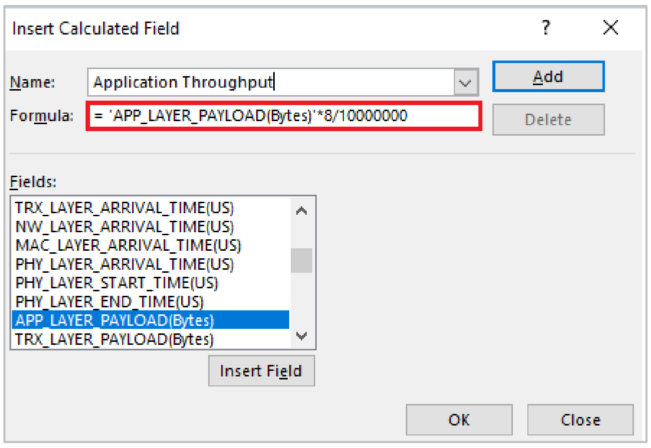
Formula: =0

Figure 8-66: Insert calculated field name and select Add

- Click on Formula text box and then select APP_LAYER_PAYLOAD in the Fields list and click on Insert Field.
- Calculate the throughput by using the following formula shown below and click on OK.

$$Application\ throughput\ (Mbps) = \frac{Sum\ of\ \ Successfully\ Received\ App\ Layer\ Payload\ (Bytes) * 8}{Simulation\ Time\ (\mu s\ )}$$

Formula='APP_LAYER_PAYLOAD(Bytes)'*8/10000000



Name: Application Throughput

Formula: = 'APP_LAYER_PAYLOAD(Bytes)'*8/10000000

Figure 8-67: Calculate the throughput by using the following formula

- Then Drag and drop the newly added Application throughput to values field

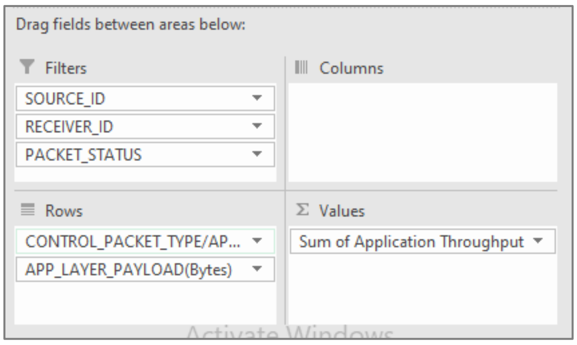

Drag fields between areas below:
Filters
Columns
SOURCE_ID
RECEIVER_ID
PACKET_STATUS
Rows
Values
CONTROL_PACKET_TYPE/AP...
APP_LAYER_PAYLOAD(Bytes)
Sum of Application Throughput

Figure 8-68: Add Application throughput to values field

- Select a cell in the pivot table, and on the Excel Ribbon, under the PivotTable Tools tab, click the Options tab (PivotTable Analyze tab in Excel 2013).
- In the Tools group, click Pivot chart and select OK.

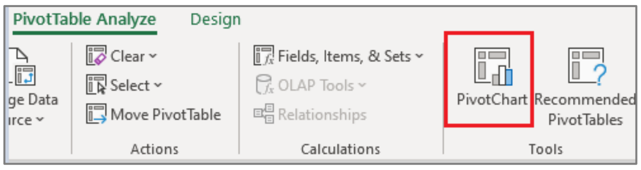


Figure 8-69: In the Tools group Select Pivot chart

- This will display a pivot chart shown below.

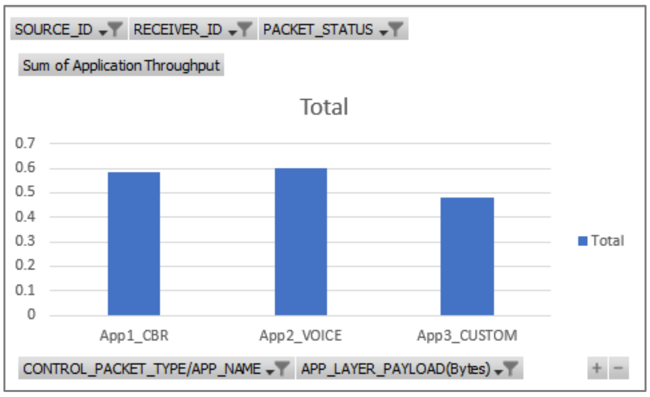


Figure 8-70: Pivot Chart

***(Note:** The procedure may vary with different versions of excel, the given procedure is according to the Excel 2017.)*

8.4.9 Packet Trace Fields

GENERAL FIELDS	DESCRIPTION
PACKET_ID	Specifies the ID of the Data Packets. For control packets this value is set to 0 For every application packet IDs are assigned in serial order. The Packet ID is not a unique number. It is the tuple {Application ID, Packet_ID} that is unique.
SEGMENT_ID	Specifies the ID of the segment of the Data Packet. Segmentation is done in transport layer. If the packet size (generated in the APP layer) is greater than the maximum segment size in TRANSPORT layer, packet will get segmented. For control packets it is N/A

PACKET_TYPE	Specifies the type of application that generates the packet. It can be Control Packet, Custom, CBR, Peer_to_peer, E-Mail, DataBase, FTP, Video, Voice, HTTP.
CONTROL_PACKET_TYPE	Specifies the type of Control Packet transmitted. Following are the Protocol specific control packets WLAN: WLAN_ACK, WLAN_BlockACK OSPF: OSPF_HELLO, OSPF_D-D, OSPF_LSR, OSPF_LSU, OSPF_LSA RIP: RIP_Message GSM: GSM_Channel_Request, GSM_Channel_Granted, GSM_Call_Request, GSM_Channel_Request_For_Incoming, GSM_Call_Accepted CDMA: CDMA_Channel_Request, CDMA_Channel_Granted, CDMA_Call_Request, CDMA_Channel_Request_For_Incoming, CDMA_Call_Accepted DSR, AODV, ZRP, OLSR: RREQ, RREP, NDP_HELLO_MESSAGE, OLSR_TC_MESSAGE Zigbee: Zigbee_BEACON_FRAME, Zigbee_ACK Cognitive Radio: SCH, FCH, DS-MAP, US-MAP, UCD, DCD, BW_REQUEST, UCS_NOTIFICATION LTE: LTE_Measurement_Report, LTE_RRC_CONNECTION_SETUP, LTE_RLC_SDU, LTE_RRC_CONNECTION_REQUEST, LTE_RRC_CONNECTION_SETUP_COMPLETE, LTE page, LTE Ack etc.
SOURCE_ID	Specifies the <Device-type>-<ID> of the source set in the application. Note that if the device name is changed the new name will not reflect in the trace.
DESTINATION_ID	Specifies the <Device-type>-<ID> of the destination set in the application. Note that if the device name is changed the new name will not reflect in the trace. If the application is a broadcast application the destination field will show 0
TRANSMITTER_ID	Specifies the <Device-type>-<ID> of the current node which is transmitting the packet. Note that if the device name is changed the new name will not reflect in the trace. The difference between a Source node and a Transmitter, is that when the Source remains constant across the entire packet transmission whereas the transmitter ID changes with each hop of the packet.
RECEIVER_ID	Specifies the <Device-type>-<ID> of the current node which is receiving the packet. Note that if the device name is changed the new name will not reflect in the trace. The difference between a Destination node and a Receiver, is that when the Destination remains constant across the entire packet transmission whereas the receiver ID changes with each hop of the packet.
APP_LAYER_ARRIVAL_TIME (µs)	Specifies the time at which packet is at the Application_Layer of Source_ID (or Transmitter_ID). This is usually the time at which the packet is generated at Source_ID
TRX_LAYER_ARRIVAL_TIME (µs)	Specifies the time at which packet reaches the Transport_layer from the application layer. This will usually be the same as Application_layer_Arrival_Time unless there are TCP re-transmissions
NW_LAYER_ARRIVAL_TIME (µs)	Specifies the time at which packet reaches the Network_Layer of Transmitter_ID if this is a Router (or) Time at which packet reaches the Network_layer of previous Router / Source_ID (immediate previous Layer 3 or higher device) if current device is Switch / Access Point.
MAC_LAYER_ARRIVAL_TIME (µs)	Specifies the time at which packet reaches MAC_Layer of Transmitter_ID
PHY_LAYER_ARRIVAL_TIME (µs)	Specifies the time at which packet reaches PHY_layer of Transmitter_ID

PHY_LAYER_START_TIME (µs)	Specifies the time at which packet starts betting transmitted in the link between Transmitter_ID and Receiver_ID
PHY_LAYER_END_TIME (µs)	Specifies the time at which packet reaches Phy_Layer of Receiver_ID
APP_LAYER_PAYLOAD (Bytes)	Specifies the size of the Payload at Application Layer
TRX_LAYER_PAYLOAD (Bytes)	Specifies the size of the Payload at Transport Layer
NW_LAYER_PAYLOAD (Bytes)	Specifies the size of the Payload at Network Layer
MAC_LAYER_PAYLOAD (Bytes)	Specifies the size of the Payload at Data Link Layer
PHY_LAYER_PAYLOAD (Bytes)	Specifies the size of the Payload at Physical Layer
PHY_LAYER_OVERHEAD (Bytes)	Specifies the size of the overhead in Physical layer
PACKET_STATUS	Specifies whether the Packet is Successful, Collided or Errored
LOCAL_ADDRESS	Specifies the Port Number at Source Node. Port Numbers are chosen randomly by NetSim.
FOREIGN_ADDRESS	Specifies the Port Number at Destination Node. Port Numbers are chosen randomly by NetSim.
CWND (bytes)	Specifies the current size of the TCP congestion window
SEQ_NO	If TCP is enabled, it specifies the TCP Sequence number of the packet
ACK_NO	If TCP is enabled, it specifies the TCP Acknowledgement number of the packet
RTT (seconds)	Specifies the Round-Trip Time for the packet
RTO (seconds)	Specifies the Retransmission Timeouts
CONNECTION_STATE	Specifies the state of TCP connection
isSyn	If TCP is enabled, it specifies whether the packet is TCP_SYN or not
isAck	If TCP is enabled, it specifies whether the packet is TCP_ACK/TCP_SYN_ACK or not
isFin	If TCP is enabled, it specifies whether the packet is TCP_FIN or not
SEGMENT_LENGTH	Specifies the segment length of the packet
SOURCE_IP	Specifies the IP address of the source
DESTINATION_IP	Specifies the IP address of the destination
GATEWAY_IP	Specifies the IP address of the device which is transmitting a packet
NEXT_HOP_IP	Specifies the IP address of the next hop

Table 8-3: Packet Trace Fields and Description

NOTE:

- *Each line in the packet trace represents one hop of one packet.*
- *The packet trace is logged in ascending order of time as measured in Phy_Layer_End_Time.*

8.5 Event Trace (only in Standard/Pro Version)

8.5.1 NetSim Network Stack and Discrete Event Simulation working

NetSim's Network Stack forms the core of NetSim and its architectural aspects are diagrammatically explained below. It exactly mirrors the TCP/IP stack and has the following five layers.

- Application Layer – CBR, Voice, Video, HTTP, COAP etc.
- Transport Layer – TCP, UDP
- Network Layer – IP, OSPF, AODV, OLSR etc.
- MAC Layer – 802.11, 802.15.4, LTE etc.
- Physical Layer – Wired (P2P, P2MP, MP2MP), Wireless (RF Propagation)

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. In addition to these events there can be ***TIMER*** events associated with each protocol.

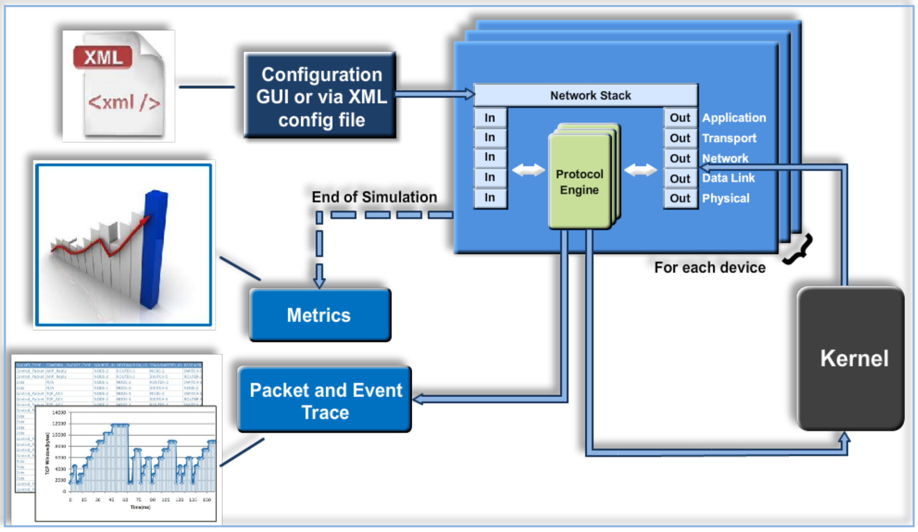


Figure 8-71: Flow of one packet from a Wired node to a Wireless node

Every device in NetSim has an instance of the Network Stack shown above. Switches & Access points have a 2-layer stack, while routers have a 3-layer stack. End-nodes have a 5-layer stack.

The protocol engines are called based on the layer at which the protocols operate. For example, TCP is called during execution of Transport IN or Transport OUT events, while 802.11b WLAN is called during execution of MAC IN, MAC OUT, PHY IN and PHY OUT events.

When these protocols are in operation, they in turn generate events for NetSim's discrete event engine to process. These are known as SUB EVENTS. All SUB EVENTS, fall into one of the above 10 types of EVENTS and TIMER events if applicable.

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.

8.5.2 Event Trace

The event trace records every single event along with associated information such as time stamp, event ID, event type etc. in a text file or .csv file which can be stored at a user defined location. Apart from a host of information, the event trace has two special information fields for diagnostics.

- A log of the file name and line number from where the event was generated (Please refer "**Writing Custom Code in NetSim → Debugging your code → Via CLI**") and
- Previous event which triggered the current event.

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

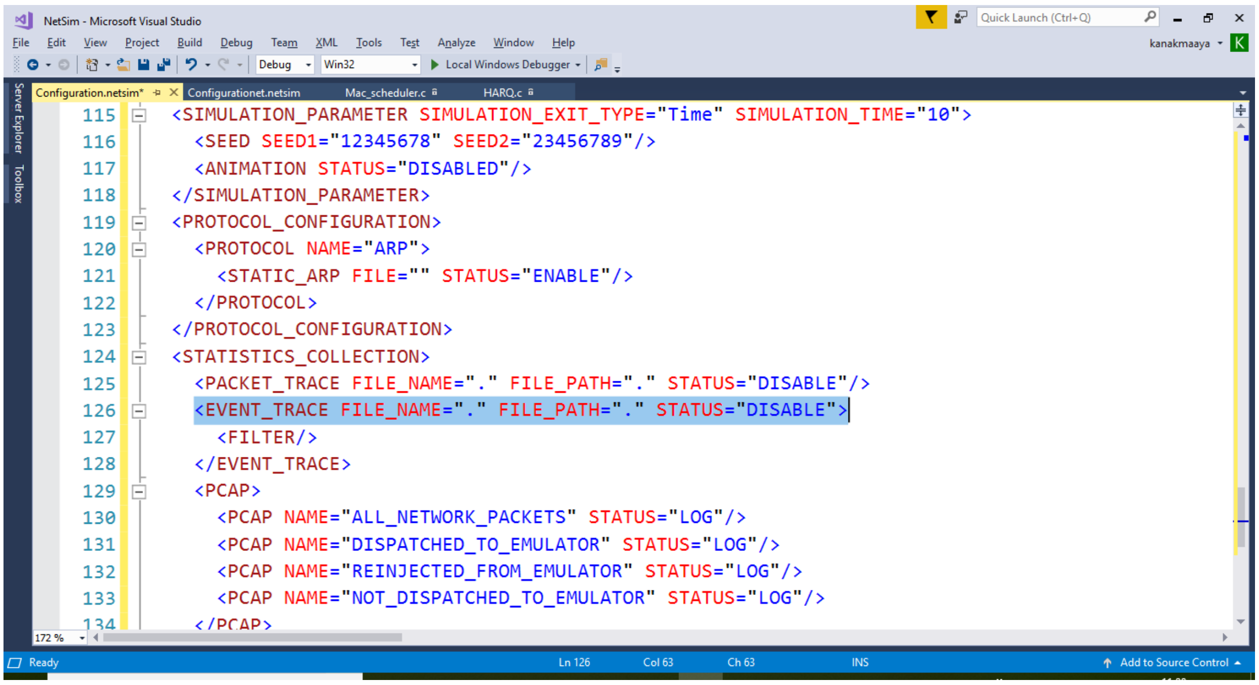
NetSim provides users with the option of turning on "Event Traces".

How to enable Event Trace via GUI?

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

How to enable Event Trace via CLI?

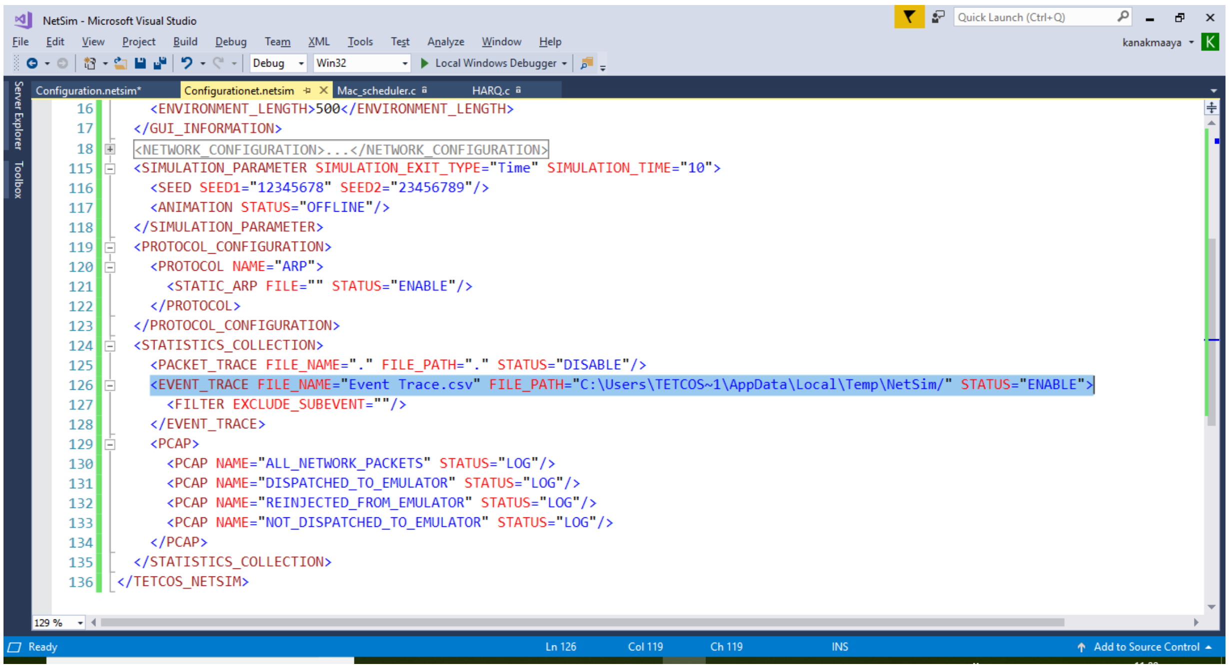
If NetSim runs via CLI, then the event trace can be turned on by enabling the event trace in the STATISTICS_COLLECTION tag of the configuration file. Following is a screenshot of a Configuration.netsim file with Event Trace disabled:



```
<SIMULATION_PARAMETER SIMULATION_EXIT_TYPE="Time" SIMULATION_TIME="10">
  <SEED SEED1="12345678" SEED2="23456789"/>
  <ANIMATION STATUS="DISABLED"/>
</SIMULATION_PARAMETER>
<PROTOCOL_CONFIGURATION>
  <PROTOCOL NAME="ARP">
    <STATIC_ARP FILE="" STATUS="ENABLE"/>
  </PROTOCOL>
</PROTOCOL_CONFIGURATION>
<STATISTICS_COLLECTION>
  <PACKET_TRACE FILE_NAME="." FILE_PATH="." STATUS="DISABLE"/>
  <EVENT_TRACE FILE_NAME="." FILE_PATH="." STATUS="DISABLE">
    <FILTER/>
  </EVENT_TRACE>
  <PCAP>
    <PCAP NAME="ALL_NETWORK_PACKETS" STATUS="LOG"/>
    <PCAP NAME="DISPATCHED_TO_EMULATOR" STATUS="LOG"/>
    <PCAP NAME="REINJECTED_FROM_EMULATOR" STATUS="LOG"/>
    <PCAP NAME="NOT_DISPATCHED_TO_EMULATOR" STATUS="LOG"/>
  </PCAP>
```

Figure 8-72: Open Configuration.netsim in Visual Studio and Event Trace disabled

You can see that the STATUS is set to DISABLE, file name and file path are not set. To enable Event trace these parameters can be modified by editing the Configuration file. Open Configuration.netsim file and provide the file name, path and set status as Enable. Following is a screenshot of a Configuration.netsim file with Event Trace enabled:



```
    <ENVIRONMENT_LENGTH>500</ENVIRONMENT_LENGTH>
  </GUI_INFORMATION>
  <NETWORK_CONFIGURATION>...</NETWORK_CONFIGURATION>
  <SIMULATION_PARAMETER SIMULATION_EXIT_TYPE="Time" SIMULATION_TIME="10">
    <SEED SEED1="12345678" SEED2="23456789"/>
    <ANIMATION STATUS="OFFLINE"/>
  </SIMULATION_PARAMETER>
  <PROTOCOL_CONFIGURATION>
    <PROTOCOL NAME="ARP">
      <STATIC_ARP FILE="" STATUS="ENABLE"/>
    </PROTOCOL>
  </PROTOCOL_CONFIGURATION>
  <STATISTICS_COLLECTION>
    <PACKET_TRACE FILE_NAME="." FILE_PATH="." STATUS="DISABLE"/>
    <EVENT_TRACE FILE_NAME="Event Trace.csv" FILE_PATH="C:\Users\TETCOS~1\AppData\Local\Temp\NetSim/" STATUS="ENABLE">
      <FILTER EXCLUDE_SUBEVENT=""/>
    </EVENT_TRACE>
    <PCAP>
      <PCAP NAME="ALL_NETWORK_PACKETS" STATUS="LOG"/>
      <PCAP NAME="DISPATCHED_TO_EMULATOR" STATUS="LOG"/>
      <PCAP NAME="REINJECTED_FROM_EMULATOR" STATUS="LOG"/>
      <PCAP NAME="NOT_DISPATCHED_TO_EMULATOR" STATUS="LOG"/>
    </PCAP>
  </STATISTICS_COLLECTION>
</TETCOS_NETSIM>
```

Figure 8-73: Event Trace enabled in Configuration.netsim file

Event Trace Metrics:

Event_Id	Specifies the ID of the Event
Event_Type	Specifies the type of event being performed, for e.g. - APPLICATION_IN, APPLICATION_OUT, MAC_OUT, MAC_IN, PHYSICAL_OUT, PHYSICAL_IN, etc.

Event_Time	Specifies the time (in microseconds) at which the event is being executed
Device_Type	Specifies the type of device in which the current event is being executed
Device_Id	Specifies the ID of device in which the current event is being executed
Interface_Id	Specifies the Interface_Id of device in which the present event is being executed.
Application_Id	Specifies the ID of the Application on which the specific event is executed
Packet_Id	Specifies the ID of the packet on which the current event is being executed
Segment_Id	Specifies the ID of the segment of packet on which the current event is being executed
Protocol_Name	Specifies the Protocol which is presently executed
Subevent_Type	Specifies the protocol sub event which is being executed. If the sub event value is 0, it indicates interlayer communication (Ex: MAC_OUT called by NETWORK_OUT) or a TIMER_EVENT which has no sub event.
Packet_Size	Specifies the size of packet during the current event
Prev_Event_Id	Specifies the ID of the event which generated the current event.

Table 8-4: Event Trace fields and Descriptions

8.5.3 Calculation of Delay and Application throughput from event trace

1. Consider the scenario as explained in the section **8.4.5** Delay analysis.

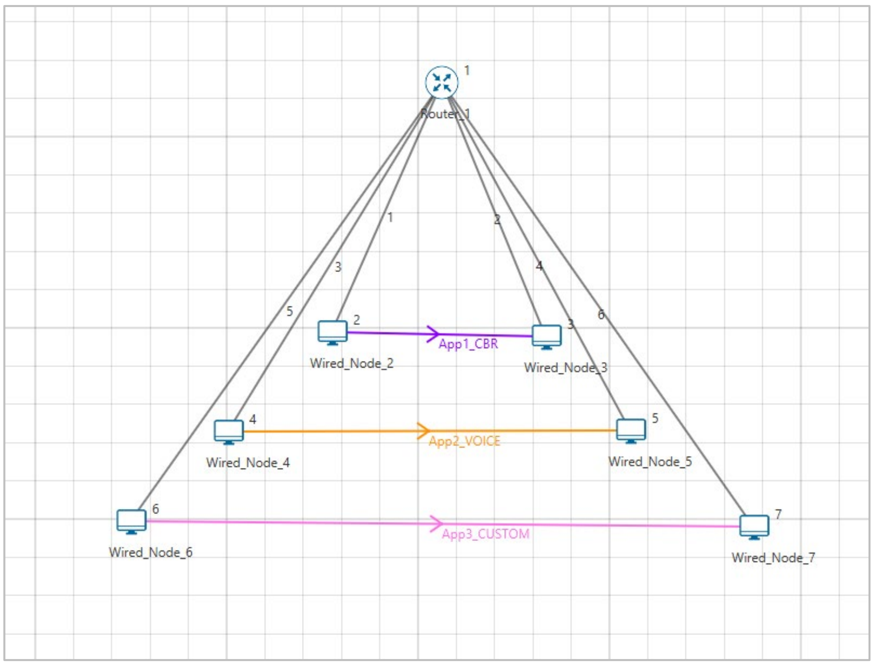
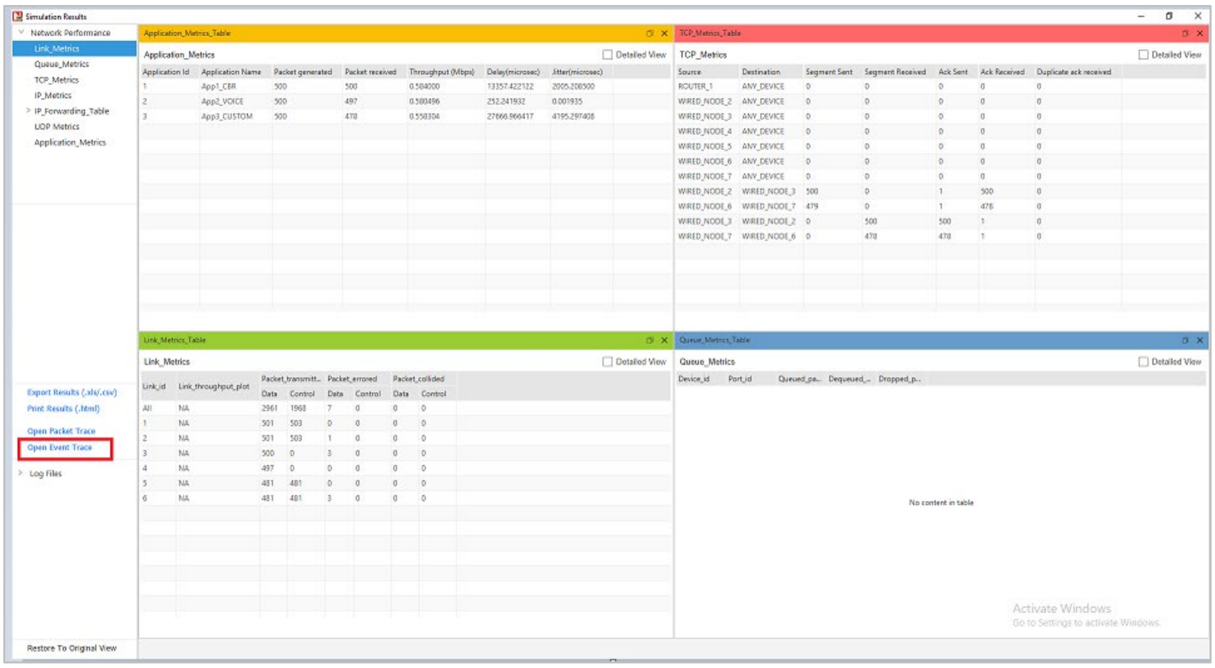


Figure 8-74: Network Scenario to calculate delay and throughput

2. Enable Event trace and simulate the scenario for 10 seconds,
3. Open event trace from the simulation results windows as shown in the below Figure 8-75.

Note: *Event tracing is available only in NetSim standard and pro versions.*



Application_Metrics_Table

Application_Metrics

Application Id	Application Name	Packet generated	Packet received	Throughput (Mbps)	Delay(microsec)	Jitter(microsec)
1	App1_CBR	500	500	0.584000	13357.422122	2005.208500
2	App2_VOICE	500	497	0.580496	252.241932	0.001935
3	App3_CUSTOM	500	478	0.558304	27666.966417	4195.297408

TCP_Metrics_Table

TCP_Metrics

Source	Destination	Segment Sent	Segment Received	Ack Sent	Ack Received	Duplicate ack received
ROUTER_1	ANY_DEVICE	0	0	0	0	0
WIRED_NODE_2	ANY_DEVICE	0	0	0	0	0
WIRED_NODE_3	ANY_DEVICE	0	0	0	0	0
WIRED_NODE_4	ANY_DEVICE	0	0	0	0	0
WIRED_NODE_5	ANY_DEVICE	0	0	0	0	0
WIRED_NODE_6	ANY_DEVICE	0	0	0	0	0
WIRED_NODE_7	ANY_DEVICE	0	0	0	0	0
WIRED_NODE_2	WIRED_NODE_3	500	0	1	500	0
WIRED_NODE_6	WIRED_NODE_7	479	0	1	478	0
WIRED_NODE_3	WIRED_NODE_2	0	500	500	1	0
WIRED_NODE_7	WIRED_NODE_6	0	478	478	1	0

Link_Metrics_Table

Link_Metrics

Link_id	Link_throughput_plot	Packet_transmitt...		Packet_errored		Packet_collided	
		Data	Control	Data	Control	Data	Control
All	NA	2961	1968	7	0	0	0
1	NA	501	503	0	0	0	0
2	NA	501	503	1	0	0	0
3	NA	500	0	3	0	0	0
4	NA	497	0	0	0	0	0
5	NA	481	481	0	0	0	0
6	NA	481	481	3	0	0	0

Queue_Metrics_Table

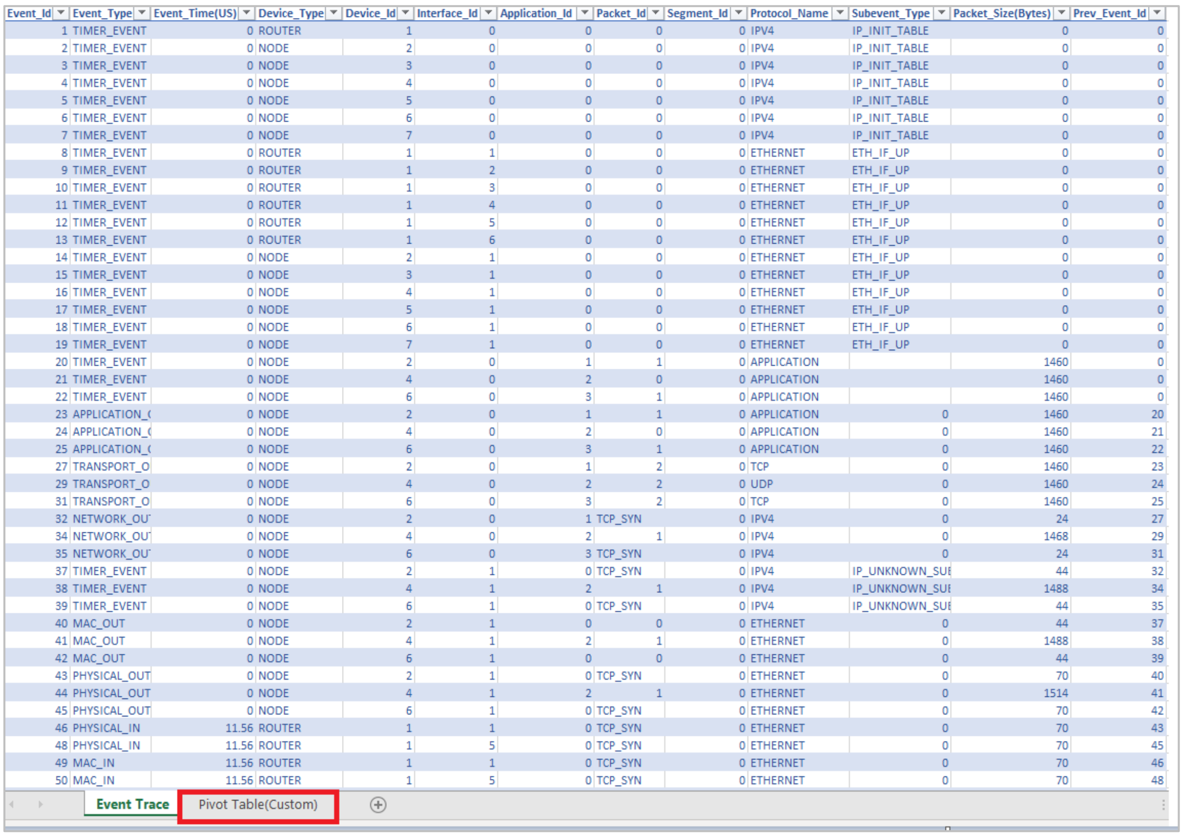
Queue_Metrics

Device_id	Port_id	Queued_pa...	Dequeued_...	Dropped_p...

No content in table

Figure 8-75: Select Event Trace option in results window

4. Click on **Pivot Table (Custom)** in excel sheet as shown below.



Event_Id	Event_Type	Event_Time(US)	Device_Type	Device_Id	Interface_Id	Application_Id	Packet_Id	Segment_Id	Protocol_Name	Subevent_Type	Packet_Size(Bytes)	Prev_Event_Id
1	TIMER_EVENT	0	ROUTER	1	0	0	0	0	IPV4	IP_INIT_TABLE	0	0
2	TIMER_EVENT	0	NODE	2	0	0	0	0	IPV4	IP_INIT_TABLE	0	0
3	TIMER_EVENT	0	NODE	3	0	0	0	0	IPV4	IP_INIT_TABLE	0	0
4	TIMER_EVENT	0	NODE	4	0	0	0	0	IPV4	IP_INIT_TABLE	0	0
5	TIMER_EVENT	0	NODE	5	0	0	0	0	IPV4	IP_INIT_TABLE	0	0
6	TIMER_EVENT	0	NODE	6	0	0	0	0	IPV4	IP_INIT_TABLE	0	0
7	TIMER_EVENT	0	NODE	7	0	0	0	0	IPV4	IP_INIT_TABLE	0	0
8	TIMER_EVENT	0	ROUTER	1	1	0	0	0	ETHERNET	ETH_IF_UP	0	0
9	TIMER_EVENT	0	ROUTER	1	2	0	0	0	ETHERNET	ETH_IF_UP	0	0
10	TIMER_EVENT	0	ROUTER	1	3	0	0	0	ETHERNET	ETH_IF_UP	0	0
11	TIMER_EVENT	0	ROUTER	1	4	0	0	0	ETHERNET	ETH_IF_UP	0	0
12	TIMER_EVENT	0	ROUTER	1	5	0	0	0	ETHERNET	ETH_IF_UP	0	0
13	TIMER_EVENT	0	ROUTER	1	6	0	0	0	ETHERNET	ETH_IF_UP	0	0
14	TIMER_EVENT	0	NODE	2	1	0	0	0	ETHERNET	ETH_IF_UP	0	0
15	TIMER_EVENT	0	NODE	3	1	0	0	0	ETHERNET	ETH_IF_UP	0	0
16	TIMER_EVENT	0	NODE	4	1	0	0	0	ETHERNET	ETH_IF_UP	0	0
17	TIMER_EVENT	0	NODE	5	1	0	0	0	ETHERNET	ETH_IF_UP	0	0
18	TIMER_EVENT	0	NODE	6	1	0	0	0	ETHERNET	ETH_IF_UP	0	0
19	TIMER_EVENT	0	NODE	7	1	0	0	0	ETHERNET	ETH_IF_UP	0	0
20	TIMER_EVENT	0	NODE	2	0	1	1	0	APPLICATION		1460	0
21	TIMER_EVENT	0	NODE	4	0	2	0	0	APPLICATION		1460	0
22	TIMER_EVENT	0	NODE	6	0	3	1	0	APPLICATION		1460	0
23	APPLICATION_(	0	NODE	2	0	1	1	0	APPLICATION	0	1460	20
24	APPLICATION_(	0	NODE	4	0	2	0	0	APPLICATION	0	1460	21
25	APPLICATION_(	0	NODE	6	0	3	1	0	APPLICATION	0	1460	22
27	TRANSPORT_O	0	NODE	2	0	1	2	0	TCP	0	1460	23
29	TRANSPORT_O	0	NODE	4	0	2	2	0	UDP	0	1460	24
31	TRANSPORT_O	0	NODE	6	0	3	2	0	TCP	0	1460	25
32	NETWORK_OU	0	NODE	2	0	1	TCP_SYN	0	IPV4	0	24	27
34	NETWORK_OU	0	NODE	4	0	2	1	0	IPV4	0	1468	29
35	NETWORK_OU	0	NODE	6	0	3	TCP_SYN	0	IPV4	0	24	31
37	TIMER_EVENT	0	NODE	2	1	0	TCP_SYN	0	IPV4	IP_UNKNOWN_SUE	44	32
38	TIMER_EVENT	0	NODE	4	1	2	1	0	IPV4	IP_UNKNOWN_SUE	1488	34
39	TIMER_EVENT	0	NODE	6	1	0	TCP_SYN	0	IPV4	IP_UNKNOWN_SUE	44	35
40	MAC_OUT	0	NODE	2	1	0	0	0	ETHERNET	0	44	37
41	MAC_OUT	0	NODE	4	1	2	1	0	ETHERNET	0	1488	38
42	MAC_OUT	0	NODE	6	1	0	0	0	ETHERNET	0	44	39
43	PHYSICAL_OUT	0	NODE	2	1	0	TCP_SYN	0	ETHERNET	0	70	40
44	PHYSICAL_OUT	0	NODE	4	1	2	1	0	ETHERNET	0	1514	41
45	PHYSICAL_OUT	0	NODE	6	1	0	TCP_SYN	0	ETHERNET	0	70	42
46	PHYSICAL_IN	11.56	ROUTER	1	1	0	TCP_SYN	0	ETHERNET	0	70	43
48	PHYSICAL_IN	11.56	ROUTER	1	5	0	TCP_SYN	0	ETHERNET	0	70	45
49	MAC_IN	11.56	ROUTER	1	1	0	TCP_SYN	0	ETHERNET	0	70	46
50	MAC_IN	11.56	ROUTER	1	5	0	TCP_SYN	0	ETHERNET	0	70	48

Figure 8-76: Select Pivot Table (Custom) in excel sheet

5. A blank **PivotTable** and **Field List** will appear on a new worksheet.

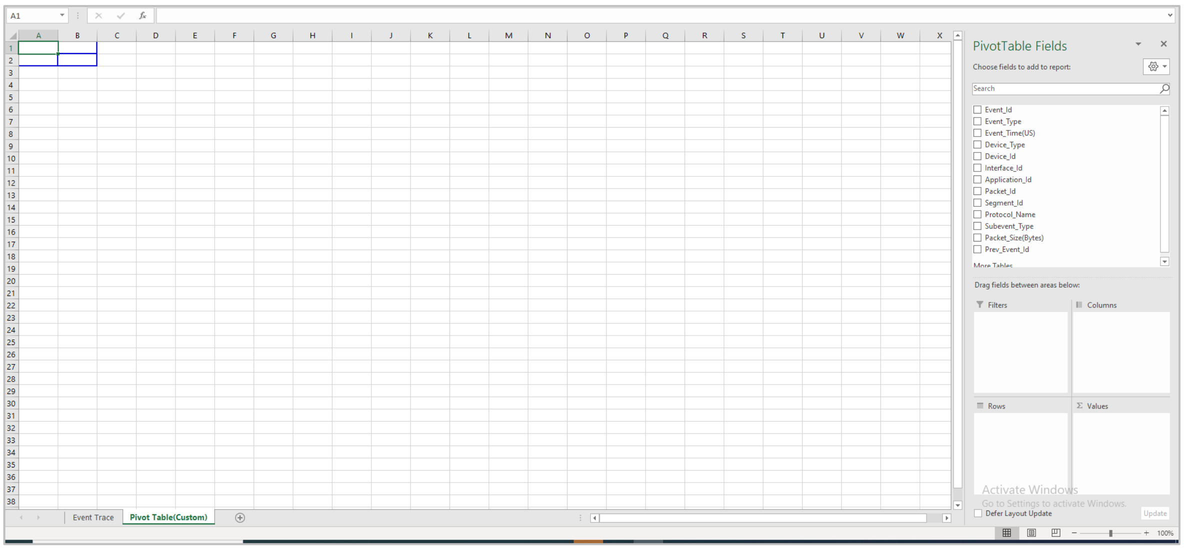


Figure 8-77: A blank PivotTable

6. Once PivotTable worksheet open, you will need to decide which fields to add. Each field is simply a **column header** from the source data. In the **PivotTable Field List**, check the box for each field you want to add.

8.5.3.1 Application Delay Analysis:

1. Drag and drop the Event_Type, Protocol_Name Fields into FILTERS, Packet_Id into ROWS and Device_Id into COLUMNS.
2. Drag and Drop Event_Time Field into VALUES twice, then both will show Sum of Event_Time. Recheck that you have dropped the Event_Time field twice.
3. Click on the second Event_Time field in the VALUES and select the Value Field Settings.

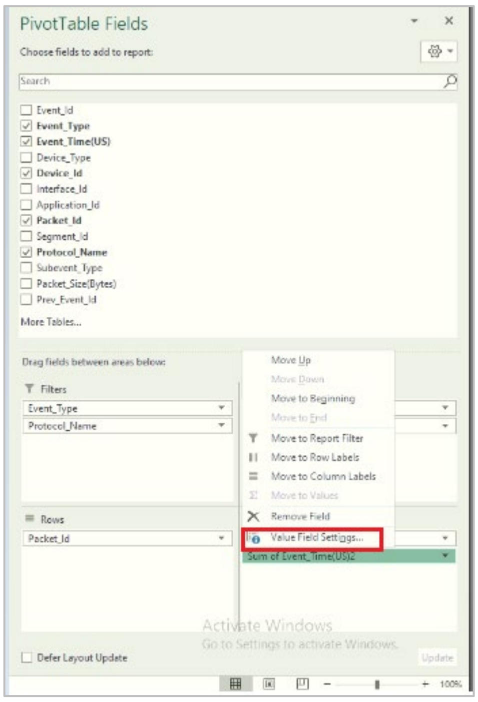
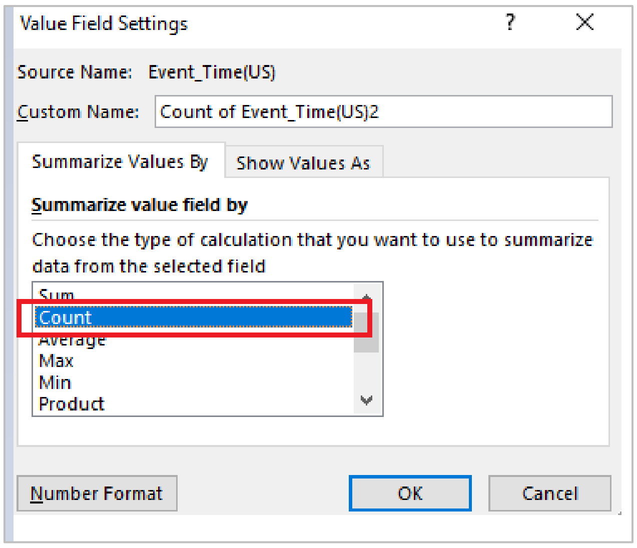


Figure 8-78: Select Second Event_Time field in the VALUES and select the Value Field Settings

4. A window named **Value Field Settings** opens then select **Count** option and click **OK** button.



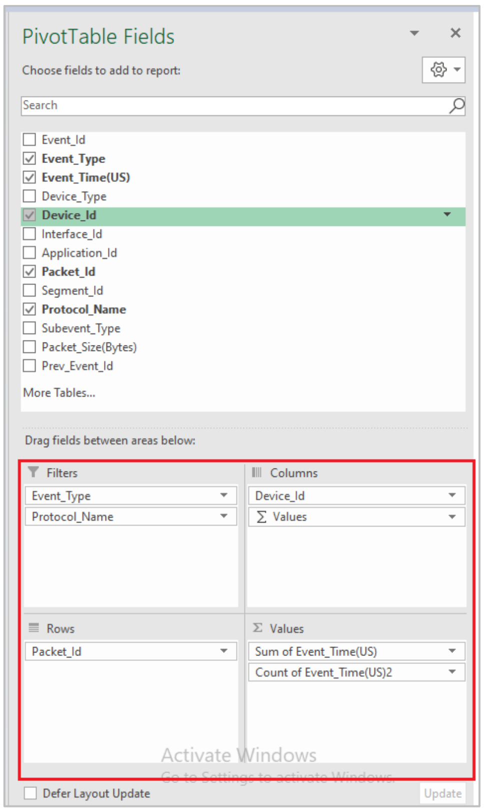
Value Field Settings

Source Name: Event_Time(US)

Custom Name: Count of Event_Time(US)2

Figure 8-79: Select Summarize value field by Count

5. Then finally the **Pivot Table Fields** will be as shown below Figure 8-80.



PivotTable Fields

Choose fields to add to report:

- [ ] Event_Id
- [x] Event_Type
- [x] Event_Time(US)
- [ ] Device_Type
- [x] Device_Id
- [ ] Interface_Id
- [ ] Application_Id
- [x] Packet_Id
- [ ] Segment_Id
- [x] Protocol_Name
- [ ] Subevent_Type
- [ ] Packet_Size(Bytes)
- [ ] Prev_Event_Id

More Tables...

Drag fields between areas below:

Filters	Columns
Event_Type	Device_Id
Protocol_Name	Σ Values

Rows	Values
Packet_Id	Sum of Event_Time(US)
	Count of Event_Time(US)2

Figure 8-80: Selected Fields to one of the four areas Field list

6. In the Event_Type select APPLICATION_IN and APPLICATION_OUT, **Protocol_Name** select **APPLICATION** and in **Column Labels** select the **Source_Id** and **Destination_Id**. In our example source node ID is 2 and destination node ID is 3.

Note: *After selecting the dropdown, to check and uncheck the check box proceed by selecting the selecting the multiple items check box*

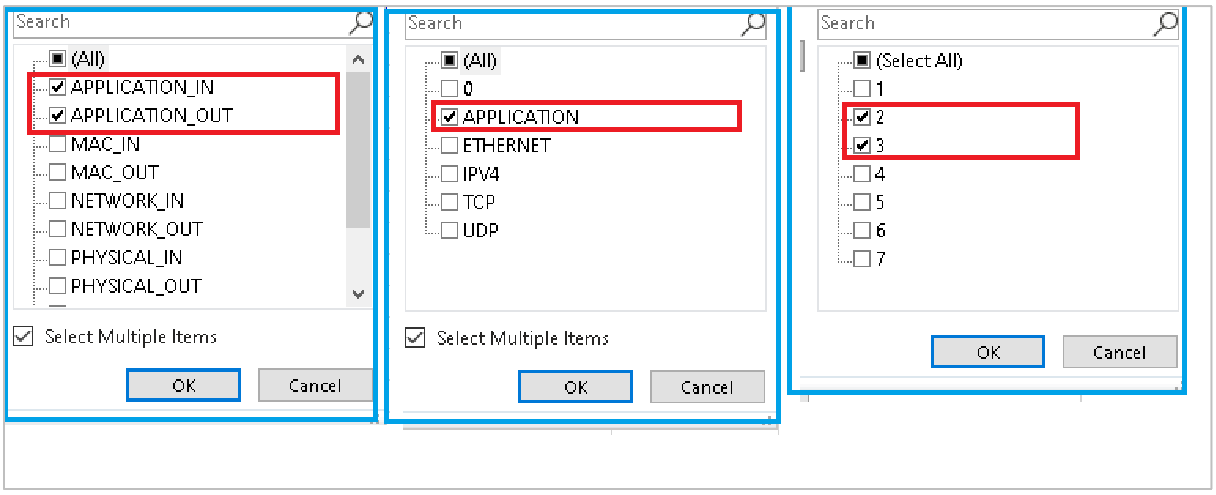
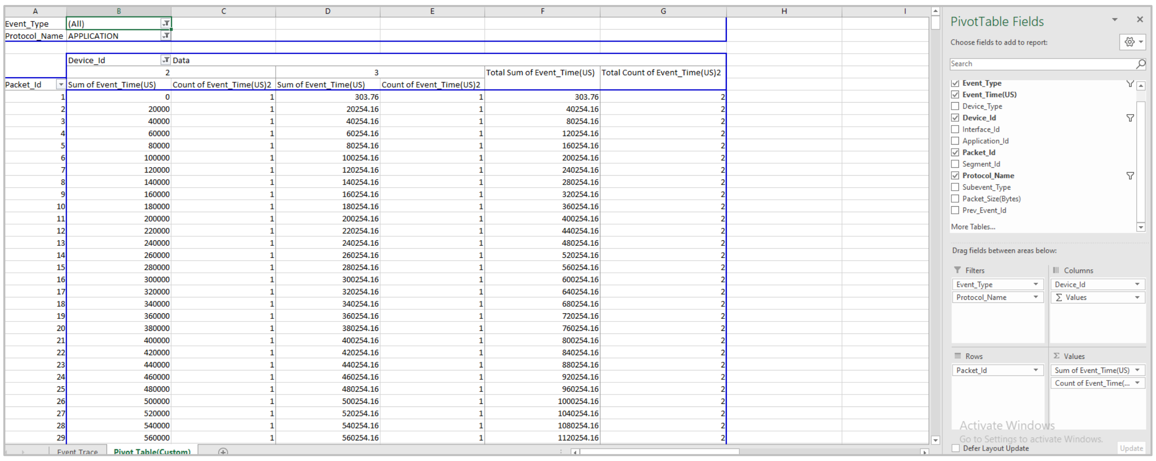


Figure 8-81: Select the Event type, Protocol Name, Source and Destination ID etc

7. The Pivot Table created will be as shown below.

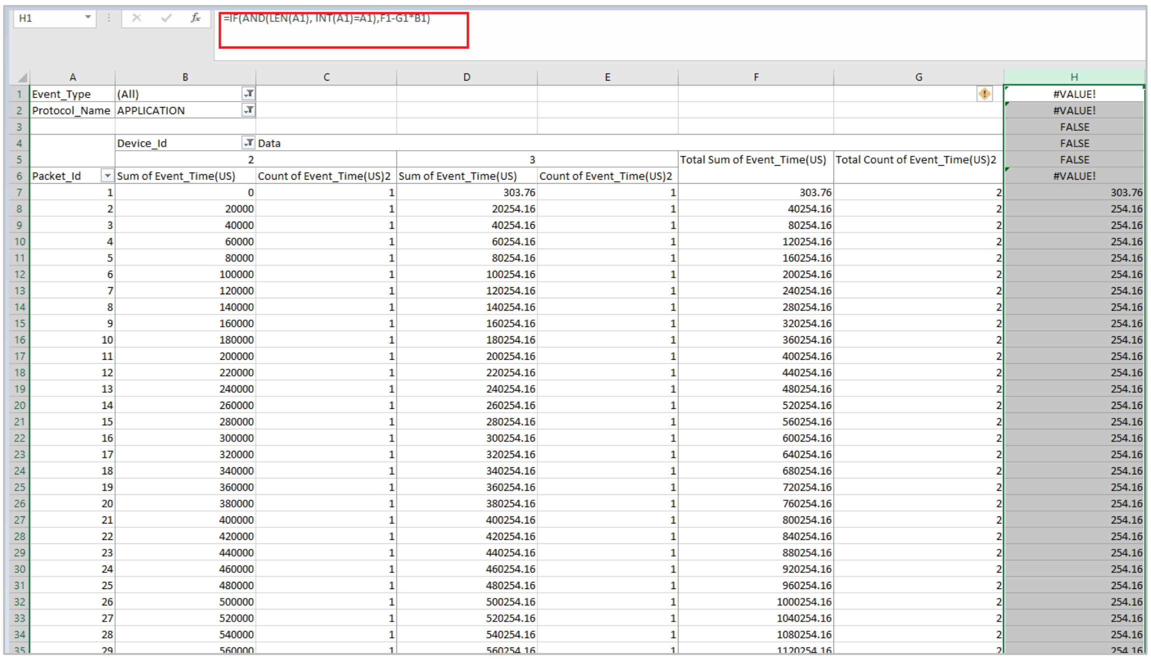


Event_Type	(All)					
Protocol_Name	APPLICATION					
	Device_Id	Data				
	2		3		Total Sum of Event_Time(US)	Total Count of Event_Time(US)2
Packet_Id	Sum of Event_Time(US)	Count of Event_Time(US)2	Sum of Event_Time(US)	Count of Event_Time(US)2		
1	0	1	303.76	1	303.76	2
2	20000	1	20254.16	1	40254.16	2
3	40000	1	40254.16	1	80254.16	2
4	60000	1	60254.16	1	120254.16	2
5	80000	1	80254.16	1	160254.16	2
6	100000	1	100254.16	1	200254.16	2
7	120000	1	120254.16	1	240254.16	2
8	140000	1	140254.16	1	280254.16	2
9	160000	1	160254.16	1	320254.16	2
10	180000	1	180254.16	1	360254.16	2
11	200000	1	200254.16	1	400254.16	2
12	220000	1	220254.16	1	440254.16	2
13	240000	1	240254.16	1	480254.16	2
14	260000	1	260254.16	1	520254.16	2
15	280000	1	280254.16	1	560254.16	2
16	300000	1	300254.16	1	600254.16	2
17	320000	1	320254.16	1	640254.16	2
18	340000	1	340254.16	1	680254.16	2
19	360000	1	360254.16	1	720254.16	2
20	380000	1	380254.16	1	760254.16	2
21	400000	1	400254.16	1	800254.16	2
22	420000	1	420254.16	1	840254.16	2
23	440000	1	440254.16	1	880254.16	2
24	460000	1	460254.16	1	920254.16	2
25	480000	1	480254.16	1	960254.16	2
26	500000	1	500254.16	1	1000254.16	2
27	520000	1	520254.16	1	1040254.16	2
28	540000	1	540254.16	1	1080254.16	2
29	560000	1	560254.16	1	1120254.16	2

Figure 8-82: Created Pivot Table

8. Select the entire empty column H then and enter the formula **=IF(AND(LEN(A1), INT(A1)=A1),F1-G1*B1)** in function and press **CTRL+ENTER**

F column is Total Sum of Event_Time, G Column is Total Count of Event_Time, B Column is Sum of Event_time(µs) of the Source.



H1 =IF(AND(LEN(A1), INT(A1)=A1),F1-G1*B1)

	A	B	C	D	E	F	G	H
1	Event_Type	(All)						#VALUE!
2	Protocol_Name	APPLICATION						#VALUE!
3								FALSE
4		Device_Id	Data					FALSE
5		2		3		Total Sum of Event_Time(US)	Total Count of Event_Time(US)2	FALSE
6	Packet_Id	Sum of Event_Time(US)	Count of Event_Time(US)2	Sum of Event_Time(US)	Count of Event_Time(US)2			#VALUE!
7	1	0	1	303.76	1	303.76	2	303.76
8	2	20000	1	20254.16	1	40254.16	2	254.16
9	3	40000	1	40254.16	1	80254.16	2	254.16
10	4	60000	1	60254.16	1	120254.16	2	254.16
11	5	80000	1	80254.16	1	160254.16	2	254.16
12	6	100000	1	100254.16	1	200254.16	2	254.16
13	7	120000	1	120254.16	1	240254.16	2	254.16
14	8	140000	1	140254.16	1	280254.16	2	254.16
15	9	160000	1	160254.16	1	320254.16	2	254.16
16	10	180000	1	180254.16	1	360254.16	2	254.16
17	11	200000	1	200254.16	1	400254.16	2	254.16
18	12	220000	1	220254.16	1	440254.16	2	254.16
19	13	240000	1	240254.16	1	480254.16	2	254.16
20	14	260000	1	260254.16	1	520254.16	2	254.16
21	15	280000	1	280254.16	1	560254.16	2	254.16
22	16	300000	1	300254.16	1	600254.16	2	254.16
23	17	320000	1	320254.16	1	640254.16	2	254.16
24	18	340000	1	340254.16	1	680254.16	2	254.16
25	19	360000	1	360254.16	1	720254.16	2	254.16
26	20	380000	1	380254.16	1	760254.16	2	254.16
27	21	400000	1	400254.16	1	800254.16	2	254.16
28	22	420000	1	420254.16	1	840254.16	2	254.16
29	23	440000	1	440254.16	1	880254.16	2	254.16
30	24	460000	1	460254.16	1	920254.16	2	254.16
31	25	480000	1	480254.16	1	960254.16	2	254.16
32	26	500000	1	500254.16	1	1000254.16	2	254.16
33	27	520000	1	520254.16	1	1040254.16	2	254.16
34	28	540000	1	540254.16	1	1080254.16	2	254.16
35	29	560000	1	560254.16	1	1120254.16	2	254.16

Figure 8-83: Select Entire empty column H then and enter the formula =IF(AND(LEN(A1), INT(A1)=A1),F1-G1*B1) in function and press CTRL+ENTER

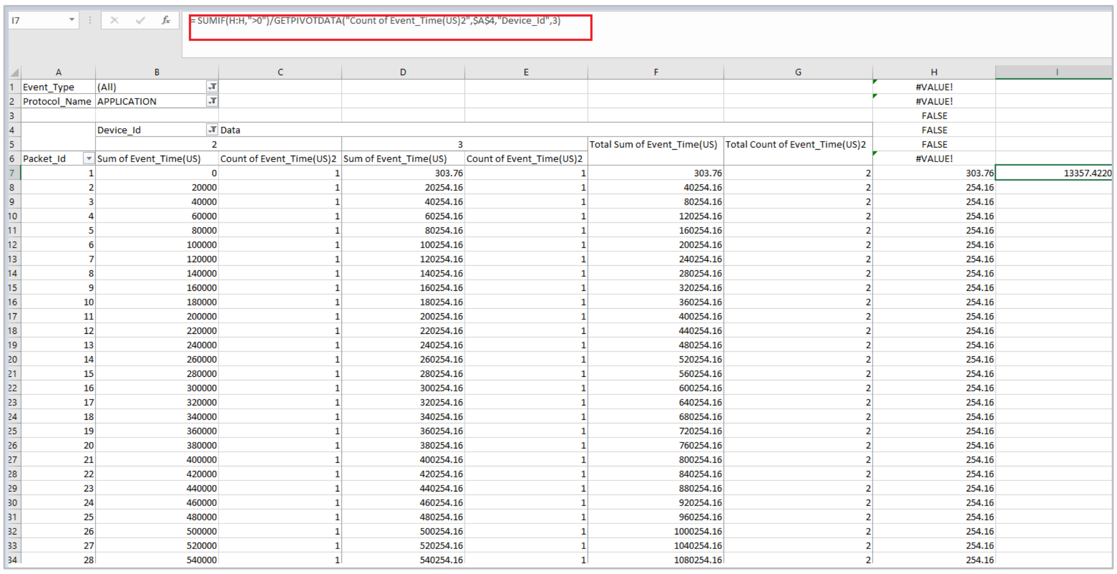
$$\text{App Delay} = \frac{\mathit{Sum\ of\ the\ Delays\ of\ the\ sucessfully\ received\ application\ data\ packets\ by\ the\ destination}}{\mathit{Total\ number\ of\ sucessful\ appplication\ data\ packets\ received\ by\ the\ destination}}$$

***Note:** If the packet size is > 1500 then fragmentation occurs and the packet is received as multiple segments. In NetSim the destination counts each segment as different packet.*

Then in an empty cell enter

=SUMIF(H:H,">0")/GETPIVOTDATA("Count of Event_Time(US)2",A4,"Device_Id",3) where

GETPIVOTDATA ("Count of Event_Time(US)2",A4,"Device_Id",3) gives the total number of packets received by the destination (in this case 3). This will give the exact Application Delay.

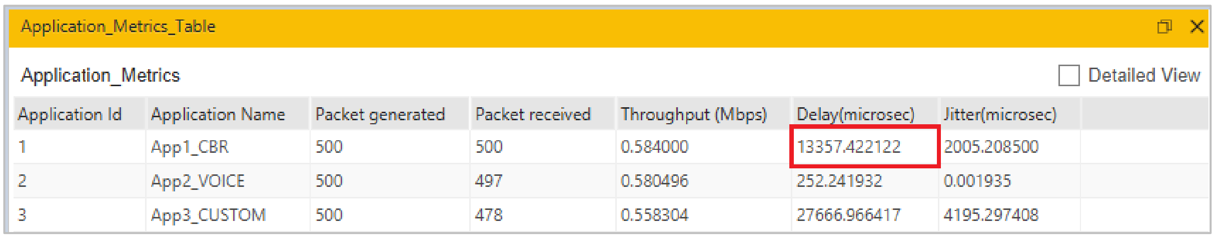


I7 = SUMIF(H:H,">0")/GETPIVOTDATA("Count of Event_Time(US)2",A4,"Device_Id",3)

	A	B	C	D	E	F	G	H	I
1	Event_Type	(All)						#VALUE!	
2	Protocol_Name	APPLICATION						#VALUE!	
3								FALSE	
4		Device_Id	Data					FALSE	
5		2		3		Total Sum of Event_Time(US)	Total Count of Event_Time(US)2	FALSE	
6	Packet_Id	Sum of Event_Time(US)	Count of Event_Time(US)2	Sum of Event_Time(US)	Count of Event_Time(US)2			#VALUE!	
7	1	0	1	303.76	1	303.76	2	303.76	13357.4220
8	2	20000	1	20254.16	1	40254.16	2	254.16	
9	3	40000	1	40254.16	1	80254.16	2	254.16	
10	4	60000	1	60254.16	1	120254.16	2	254.16	
11	5	80000	1	80254.16	1	160254.16	2	254.16	
12	6	100000	1	100254.16	1	200254.16	2	254.16	
13	7	120000	1	120254.16	1	240254.16	2	254.16	
14	8	140000	1	140254.16	1	280254.16	2	254.16	
15	9	160000	1	160254.16	1	320254.16	2	254.16	
16	10	180000	1	180254.16	1	360254.16	2	254.16	
17	11	200000	1	200254.16	1	400254.16	2	254.16	
18	12	220000	1	220254.16	1	440254.16	2	254.16	
19	13	240000	1	240254.16	1	480254.16	2	254.16	
20	14	260000	1	260254.16	1	520254.16	2	254.16	
21	15	280000	1	280254.16	1	560254.16	2	254.16	
22	16	300000	1	300254.16	1	600254.16	2	254.16	
23	17	320000	1	320254.16	1	640254.16	2	254.16	
24	18	340000	1	340254.16	1	680254.16	2	254.16	
25	19	360000	1	360254.16	1	720254.16	2	254.16	
26	20	380000	1	380254.16	1	760254.16	2	254.16	
27	21	400000	1	400254.16	1	800254.16	2	254.16	
28	22	420000	1	420254.16	1	840254.16	2	254.16	
29	23	440000	1	440254.16	1	880254.16	2	254.16	
30	24	460000	1	460254.16	1	920254.16	2	254.16	
31	25	480000	1	480254.16	1	960254.16	2	254.16	
32	26	500000	1	500254.16	1	1000254.16	2	254.16	
33	27	520000	1	520254.16	1	1040254.16	2	254.16	
34	28	540000	1	540254.16	1	1080254.16	2	254.16	

Figure 8-84: Calculated Application Delay using Formula in Pivot table

Compare with the Delay in Application_Metrics_Tables and it would exactly match. There might be slight difference in the decimals due to Excel's round offs.


Application_Metrics_Table

Application_Metrics ☐ Detailed View

Application Id	Application Name	Packet generated	Packet received	Throughput (Mbps)	Delay(microsec)	Jitter(microsec)
1	App1_CBR	500	500	0.584000	13357.422122	2005.208500
2	App2_VOICE	500	497	0.580496	252.241932	0.001935
3	App3_CUSTOM	500	478	0.558304	27666.966417	4195.297408

Figure 8-85: Compare the Application_Metrics_Tables Delay and Pivot table Delay

8.5.3.2 Application Throughput Analysis

1. For Application Throughput drag and drop **Event_type**, **Protocol_Name** Fields in **FILTERS**, **Device_Id** in **ROWS**, **Packet_Size(Bytes)** into **VALUES.** Change the **Value Field Settings** of **Packets_Size(Bytes)** to **SUM** as mentioned in Delay Analysis.

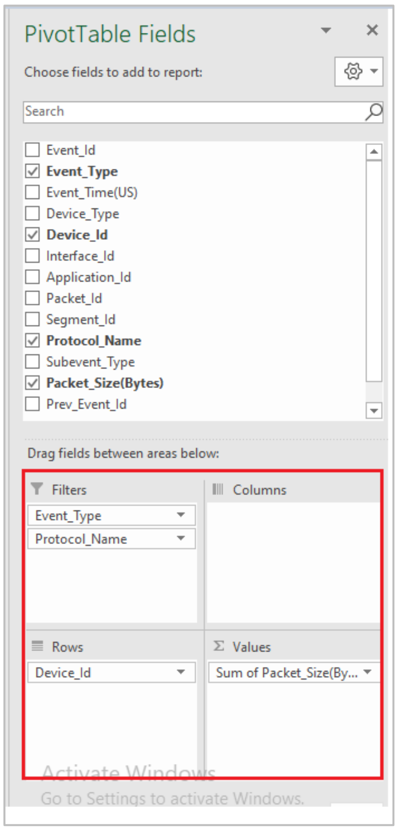


Figure 8-86: Selected Fields to one of the four areas Field list

Then Select the Event_Type as APPLICATION_IN, Protocol_Name as APPLICATION and Device_Id as the Destination (in this case destination id will be 3 since we are calculating for APP1 CBR)

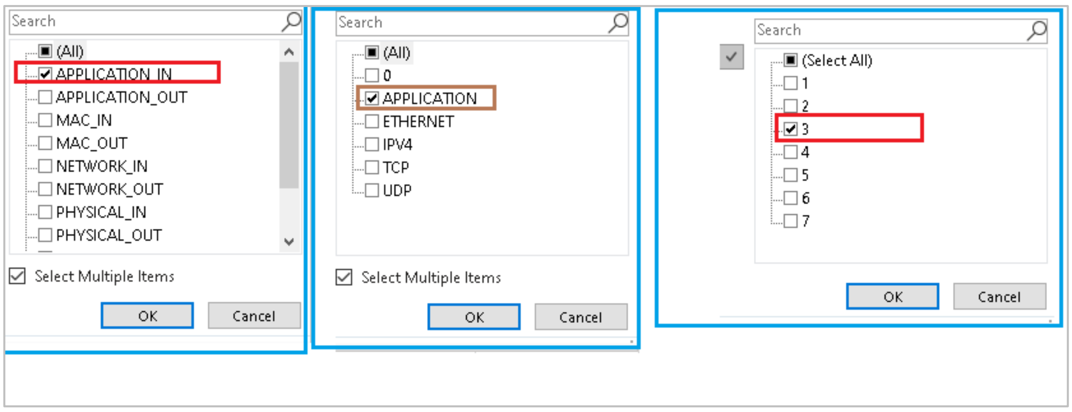
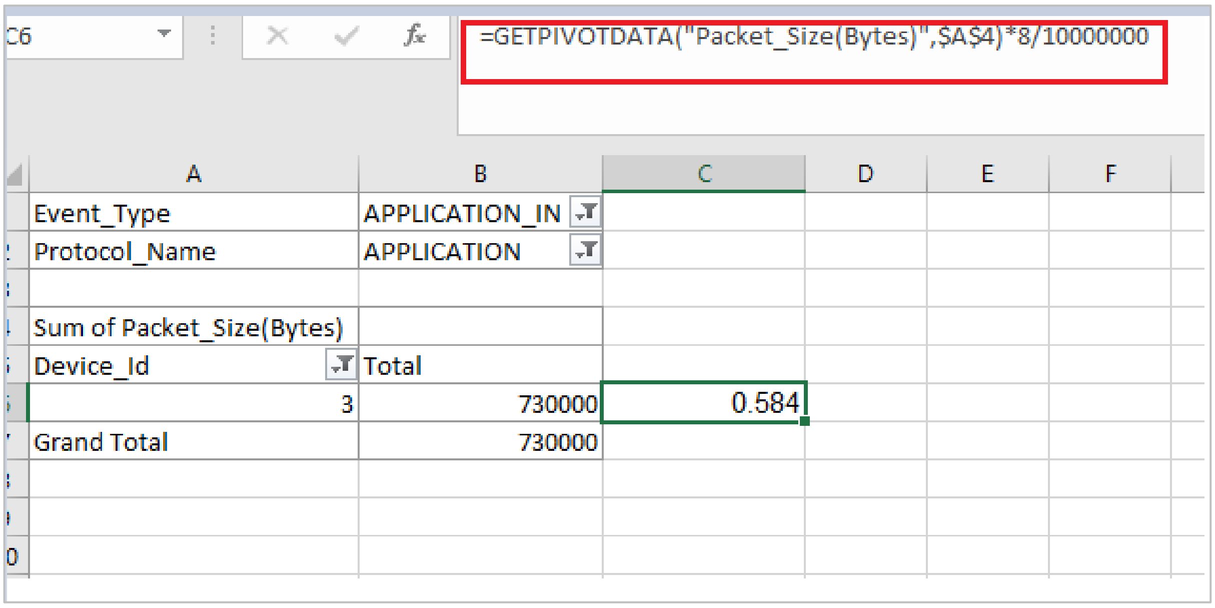


Figure 8-87: Select the Event type, Protocol Name, Source and Destination ID etc

2. App Throughput = $\frac{TotalPayloadapplicationsuccessfullyreceivedbythedestination}{Simulationtime}$

Then in an empty cell type =GETPIVOTDATA ("Packet_Size(Bytes)",A4)*8/10000000

This gives the Application Throughput in Mbps (Multiplied by 8 to convert Bytes to bits, and divided by 100000 to convert into Mega)

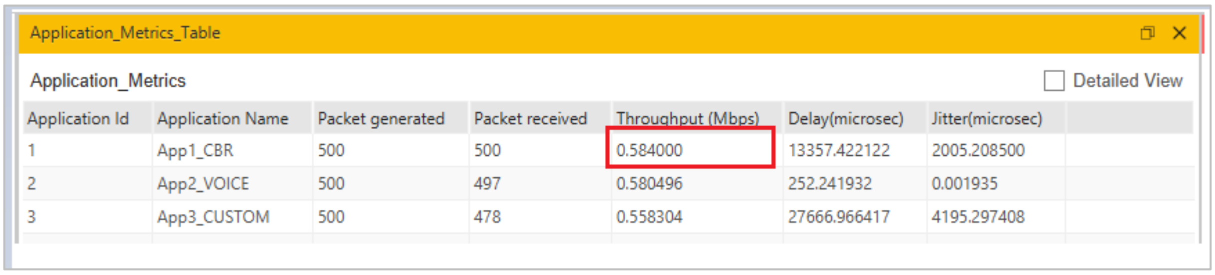

C6

=GETPIVOTDATA("Packet_Size(Bytes)",A4)*8/10000000

A	B	C
Event_Type	APPLICATION_IN	
Protocol_Name	APPLICATION	
Sum of Packet_Size(Bytes)		
Device_Id	Total	
3	730000	0.584
Grand Total	730000	

Figure 8-88: Calculate Application Throughput using formula

Compare with the Application throughput in the **Application_Metrics_Table**


Application_Metrics_Table

Application_Metrics

Detailed View

Application Id	Application Name	Packet generated	Packet received	Throughput (Mbps)	Delay(microsec)	Jitter(microsec)
1	App1_CBR	500	500	0.584000	13357.422122	2005.208500
2	App2_VOICE	500	497	0.580496	252.241932	0.001935
3	App3_CUSTOM	500	478	0.558304	27666.966417	4195.297408

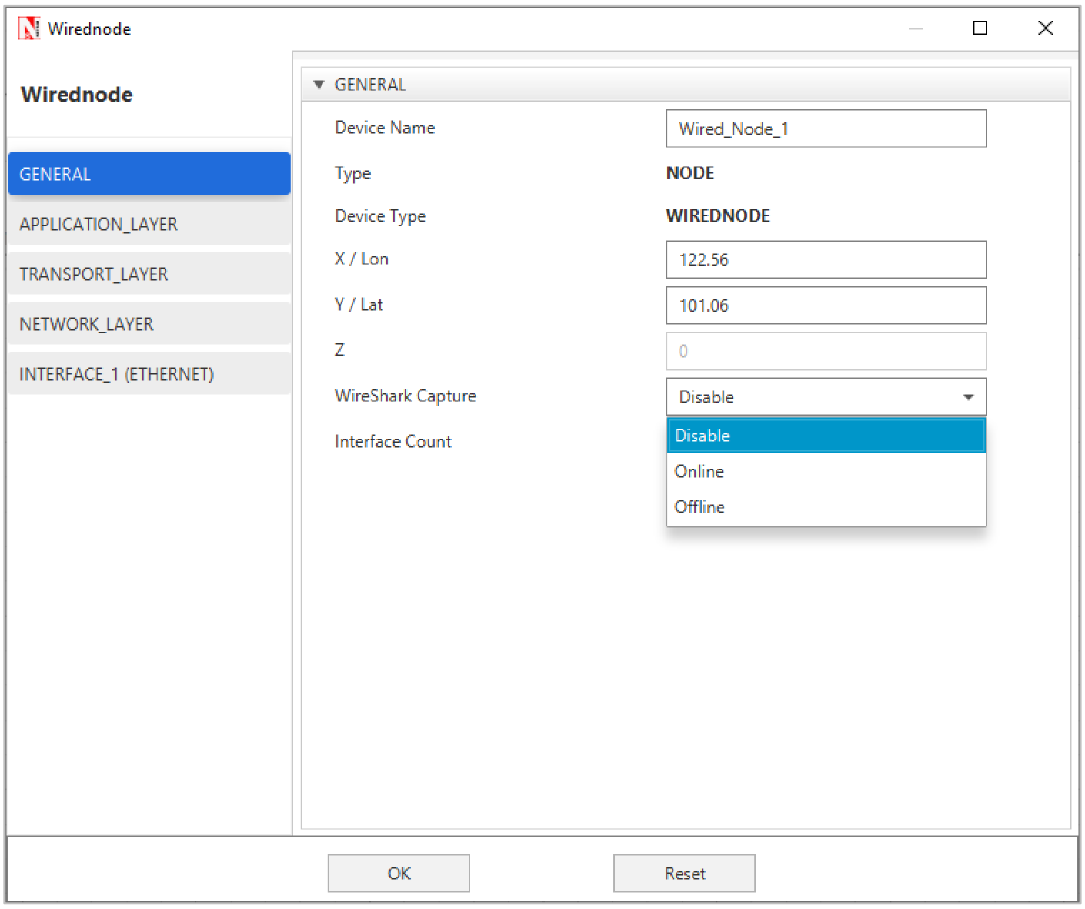
Figure 8-89: Compare the Application_Metrics_Tables throughput and Pivot table throughput

8.6 Packet Capture & analysis using Wireshark

8.6.1 Enabling Wireshark Capture in a node for packet capture

NetSim provides functionality to capture packets in the virtual nodes. The pcap file written by NetSim contains fields of packet layer 3 and above. This pcap file can be opened using the popular software, Wireshark (formerly Ethereal).

To enable packet capture in Wireshark, Right Click on the device where wireshark should be run. In the properties, go to General Properties and set the Wireshark Capture parameter as Online.



Device Name	Wired_Node_1
Type	NODE
Device Type	WIREDNODE
X / Lon	122.56
Y / Lat	101.06
Z	0
WireShark Capture	Disable (Disable / Online / Offline)
Interface Count	

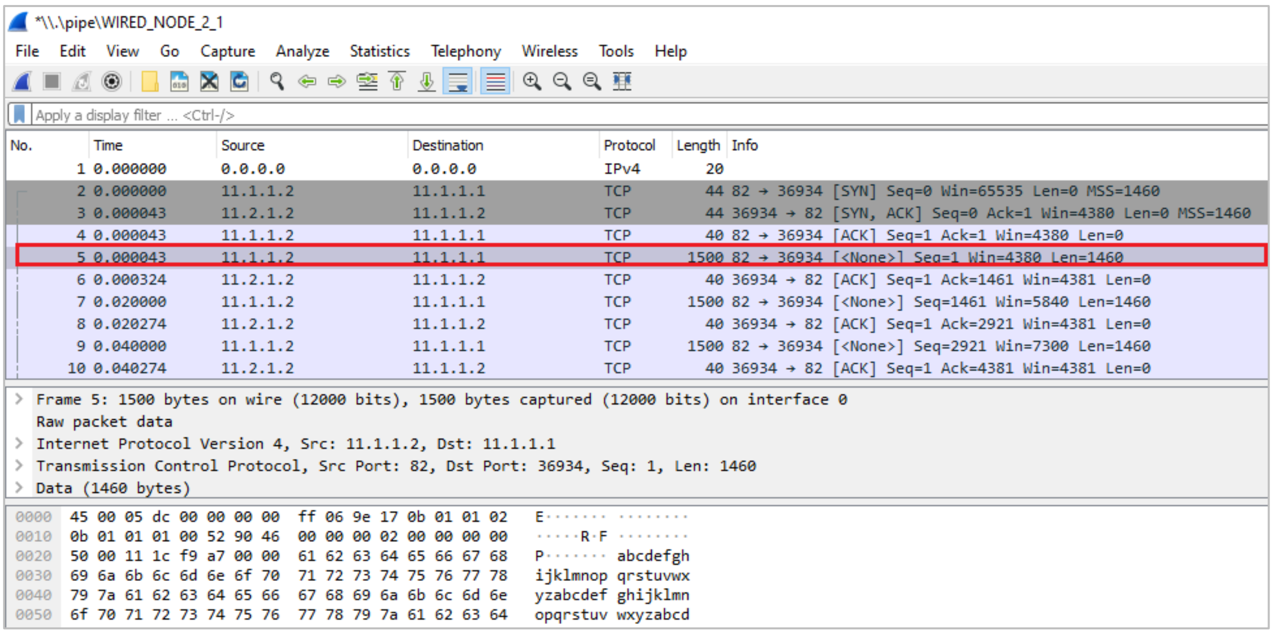
Figure 8-90: Enable Wireshark in General Properties for wired node

Wireshark Capture Options	
Online	Online option will initiate a live interactive packet capture, displaying packets while running simulation
Offline	Offline option will initiate silent packet capture and generate a pcap file which can be opened using Wireshark post-simulation
Disable	Packets are not captured by Wireshark during simulation.

Table 8-5: Wireshark Capture Options and Description

8.6.2 Viewing captured packets

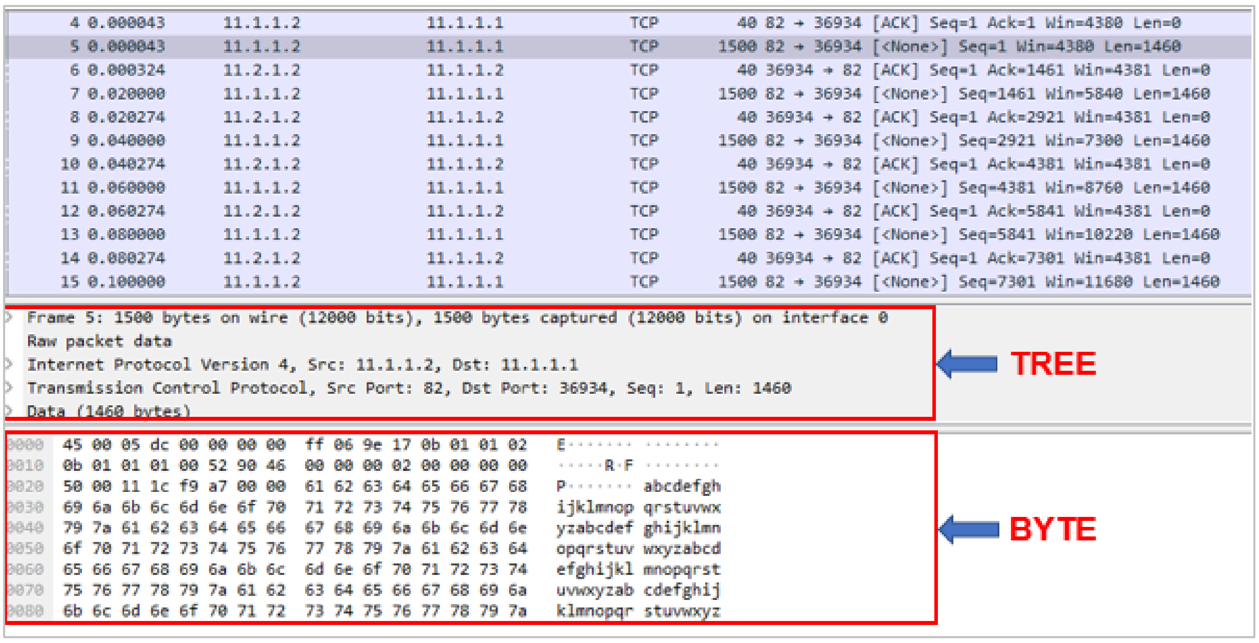
If enabled, Wireshark Capture automatically starts during simulation and displays all the captured packets. To view the details of the packet displayed, click-on the packet as shown below Figure 8-91.


*\\.\pipe\WIRED_NODE_2_1
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>

```
No.  Time      Source     Destination  Protocol Length Info
  1 0.000000  0.0.0.0    0.0.0.0      IPv4       20
  2 0.000000  11.1.1.2   11.1.1.1     TCP        44 82 → 36934 [SYN] Seq=0 Win=65535 Len=0 MSS=1460
  3 0.000043  11.2.1.2   11.1.1.2     TCP        44 36934 → 82 [SYN, ACK] Seq=0 Ack=1 Win=4380 Len=0 MSS=1460
  4 0.000043  11.1.1.2   11.1.1.1     TCP        40 82 → 36934 [ACK] Seq=1 Ack=1 Win=4380 Len=0
  5 0.000043  11.1.1.2   11.1.1.1     TCP      1500 82 → 36934 [<None>] Seq=1 Win=4380 Len=1460
  6 0.000324  11.2.1.2   11.1.1.2     TCP        40 36934 → 82 [ACK] Seq=1 Ack=1461 Win=4381 Len=0
  7 0.020000  11.1.1.2   11.1.1.1     TCP      1500 82 → 36934 [<None>] Seq=1461 Win=5840 Len=1460
  8 0.020274  11.2.1.2   11.1.1.2     TCP        40 36934 → 82 [ACK] Seq=1 Ack=2921 Win=4381 Len=0
  9 0.040000  11.1.1.2   11.1.1.1     TCP      1500 82 → 36934 [<None>] Seq=2921 Win=7300 Len=1460
 10 0.040274  11.2.1.2   11.1.1.2     TCP        40 36934 → 82 [ACK] Seq=1 Ack=4381 Win=4381 Len=0

> Frame 5: 1500 bytes on wire (12000 bits), 1500 bytes captured (12000 bits) on interface 0
  Raw packet data
> Internet Protocol Version 4, Src: 11.1.1.2, Dst: 11.1.1.1
> Transmission Control Protocol, Src Port: 82, Dst Port: 36934, Seq: 1, Len: 1460
> Data (1460 bytes)

0000  45 00 05 dc 00 00 00 00  ff 06 9e 17 0b 01 01 02   E······· ········
0010  0b 01 01 01 00 52 90 46  00 00 00 02 00 00 00 00   ·····R·F ········
0020  50 00 11 1c f9 a7 00 00  61 62 63 64 65 66 67 68   P······· abcdefgh
0030  69 6a 6b 6c 6d 6e 6f 70  71 72 73 74 75 76 77 78   ijklmnop qrstuvwx
0040  79 7a 61 62 63 64 65 66  67 68 69 6a 6b 6c 6d 6e   yzabcdef ghijklmn
0050  6f 70 71 72 73 74 75 76  77 78 79 7a 61 62 63 64   opqrstuv wxyzabcd
```

Figure 8-91: Packets Captured in Wireshark

The detail of the contents of the selected packet can be seen in the below panes as shown below Figure 8-92.



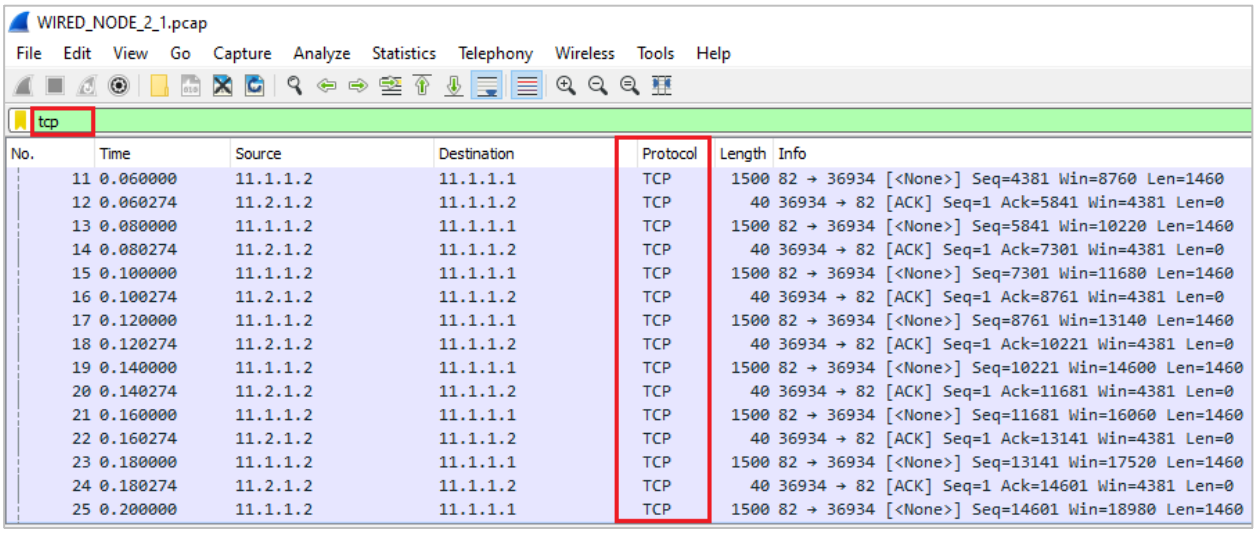
```
  4 0.000043  11.1.1.2   11.1.1.1     TCP        40 82 → 36934 [ACK] Seq=1 Ack=1 Win=4380 Len=0
  5 0.000043  11.1.1.2   11.1.1.1     TCP      1500 82 → 36934 [<None>] Seq=1 Win=4380 Len=1460
  6 0.000324  11.2.1.2   11.1.1.2     TCP        40 36934 → 82 [ACK] Seq=1 Ack=1461 Win=4381 Len=0
  7 0.020000  11.1.1.2   11.1.1.1     TCP      1500 82 → 36934 [<None>] Seq=1461 Win=5840 Len=1460
  8 0.020274  11.2.1.2   11.1.1.2     TCP        40 36934 → 82 [ACK] Seq=1 Ack=2921 Win=4381 Len=0
  9 0.040000  11.1.1.2   11.1.1.1     TCP      1500 82 → 36934 [<None>] Seq=2921 Win=7300 Len=1460
 10 0.040274  11.2.1.2   11.1.1.2     TCP        40 36934 → 82 [ACK] Seq=1 Ack=4381 Win=4381 Len=0
 11 0.060000  11.1.1.2   11.1.1.1     TCP      1500 82 → 36934 [<None>] Seq=4381 Win=8760 Len=1460
 12 0.060274  11.2.1.2   11.1.1.2     TCP        40 36934 → 82 [ACK] Seq=1 Ack=5841 Win=4381 Len=0
 13 0.080000  11.1.1.2   11.1.1.1     TCP      1500 82 → 36934 [<None>] Seq=5841 Win=10220 Len=1460
 14 0.080274  11.2.1.2   11.1.1.2     TCP        40 36934 → 82 [ACK] Seq=1 Ack=7301 Win=4381 Len=0
 15 0.100000  11.1.1.2   11.1.1.1     TCP      1500 82 → 36934 [<None>] Seq=7301 Win=11680 Len=1460

> Frame 5: 1500 bytes on wire (12000 bits), 1500 bytes captured (12000 bits) on interface 0
  Raw packet data
> Internet Protocol Version 4, Src: 11.1.1.2, Dst: 11.1.1.1
> Transmission Control Protocol, Src Port: 82, Dst Port: 36934, Seq: 1, Len: 1460
> Data (1460 bytes)

0000  45 00 05 dc 00 00 00 00  ff 06 9e 17 0b 01 01 02   E······· ········
0010  0b 01 01 01 00 52 90 46  00 00 00 02 00 00 00 00   ·····R·F ········
0020  50 00 11 1c f9 a7 00 00  61 62 63 64 65 66 67 68   P······· abcdefgh
0030  69 6a 6b 6c 6d 6e 6f 70  71 72 73 74 75 76 77 78   ijklmnop qrstuvwx
0040  79 7a 61 62 63 64 65 66  67 68 69 6a 6b 6c 6d 6e   yzabcdef ghijklmn
0050  6f 70 71 72 73 74 75 76  77 78 79 7a 61 62 63 64   opqrstuv wxyzabcd
0060  65 66 67 68 69 6a 6b 6c  6d 6e 6f 70 71 72 73 74   efghijkl mnopqrst
0070  75 76 77 78 79 7a 61 62  63 64 65 66 67 68 69 6a   uvwxyzab cdefghij
0080  6b 6c 6d 6e 6f 70 71 72  73 74 75 76 77 78 79 7a   klmnopqr stuvwxyz
```

Figure 8-92: Packet Information panes

In the above figure, the details of the packet are displayed in both tree form and bytes form. In the tree form, user can expand the data by clicking on the part of the tree and view detailed information about each protocol in each packet.

8.6.3 Filtering captured packets

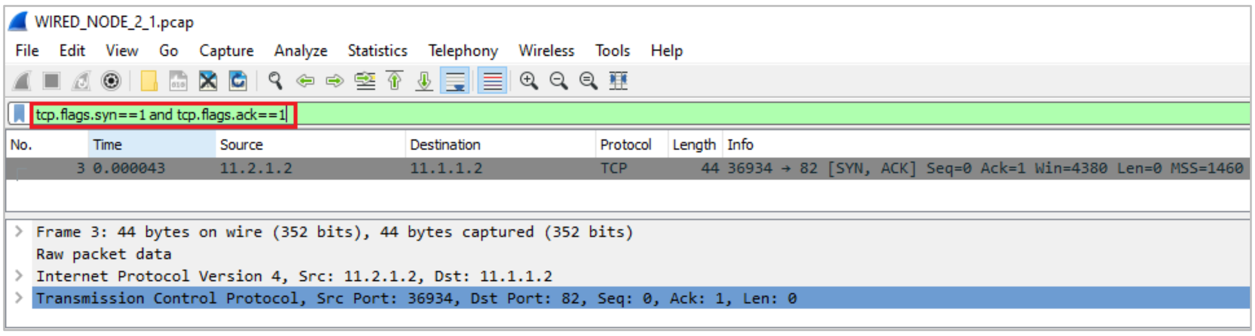
Display filters allow you to concentrate on the packets you are interested in while hiding the currently uninteresting ones. Packets can be filtered by protocol, presence of a field, values of field's etc. To select packets based on protocol, type the protocol in which you are interested in the Filter: field of the Wireshark window and presenter to initiate the filter. In the figure below Figure 8-93, tcp protocol is filtered.


WIRED_NODE_2_1.pcap
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

```
tcp
No.  Time      Source    Destination  Protocol  Length Info
  11 0.060000  11.1.1.2  11.1.1.1     TCP       1500 82 → 36934 [<None>] Seq=4381 Win=8760 Len=1460
  12 0.060274  11.2.1.2  11.1.1.2     TCP         40 36934 → 82 [ACK] Seq=1 Ack=5841 Win=4381 Len=0
  13 0.080000  11.1.1.2  11.1.1.1     TCP       1500 82 → 36934 [<None>] Seq=5841 Win=10220 Len=1460
  14 0.080274  11.2.1.2  11.1.1.2     TCP         40 36934 → 82 [ACK] Seq=1 Ack=7301 Win=4381 Len=0
  15 0.100000  11.1.1.2  11.1.1.1     TCP       1500 82 → 36934 [<None>] Seq=7301 Win=11680 Len=1460
  16 0.100274  11.2.1.2  11.1.1.2     TCP         40 36934 → 82 [ACK] Seq=1 Ack=8761 Win=4381 Len=0
  17 0.120000  11.1.1.2  11.1.1.1     TCP       1500 82 → 36934 [<None>] Seq=8761 Win=13140 Len=1460
  18 0.120274  11.2.1.2  11.1.1.2     TCP         40 36934 → 82 [ACK] Seq=1 Ack=10221 Win=4381 Len=0
  19 0.140000  11.1.1.2  11.1.1.1     TCP       1500 82 → 36934 [<None>] Seq=10221 Win=14600 Len=1460
  20 0.140274  11.2.1.2  11.1.1.2     TCP         40 36934 → 82 [ACK] Seq=1 Ack=11681 Win=4381 Len=0
  21 0.160000  11.1.1.2  11.1.1.1     TCP       1500 82 → 36934 [<None>] Seq=11681 Win=16060 Len=1460
  22 0.160274  11.2.1.2  11.1.1.2     TCP         40 36934 → 82 [ACK] Seq=1 Ack=13141 Win=4381 Len=0
  23 0.180000  11.1.1.2  11.1.1.1     TCP       1500 82 → 36934 [<None>] Seq=13141 Win=17520 Len=1460
  24 0.180274  11.2.1.2  11.1.1.2     TCP         40 36934 → 82 [ACK] Seq=1 Ack=14601 Win=4381 Len=0
  25 0.200000  11.1.1.2  11.1.1.1     TCP       1500 82 → 36934 [<None>] Seq=14601 Win=18980 Len=1460
```

Figure 8-93: TCP Protocol is filtered in Wireshark

You can also build display filters that compare values using a number of different comparison operators like ==, != , >, <, <=, etc. Following is an example displaying filtered packets whose SYN Flag and ACK Flag are set to 1 in a TCP Stream.


WIRED_NODE_2_1.pcap
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

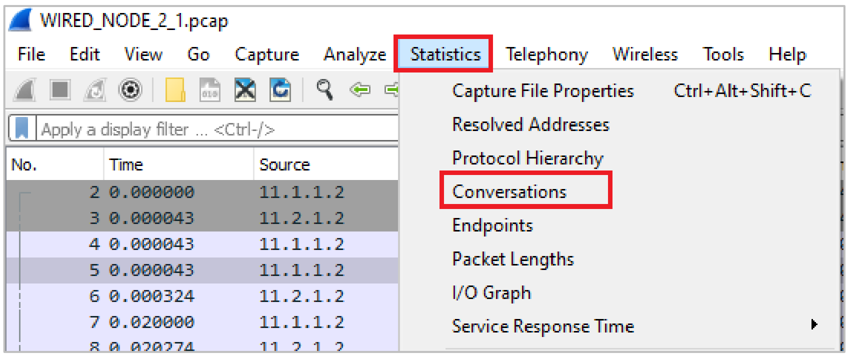
```
tcp.flags.syn==1 and tcp.flags.ack==1
No.  Time      Source    Destination  Protocol  Length Info
   3 0.000043  11.2.1.2  11.1.1.2     TCP         44 36934 → 82 [SYN, ACK] Seq=0 Ack=1 Win=4380 Len=0 MSS=1460

> Frame 3: 44 bytes on wire (352 bits), 44 bytes captured (352 bits)
  Raw packet data
> Internet Protocol Version 4, Src: 11.2.1.2, Dst: 11.1.1.2
> Transmission Control Protocol, Src Port: 36934, Dst Port: 82, Seq: 0, Ack: 1, Len: 0
```

Figure 8-94: Filtered SYN Flag and ACK Flag are set to 1 in a TCP Stream

8.6.4 Analyzing packets in Wireshark

8.6.4.1 Analyzing Conversation using graphs

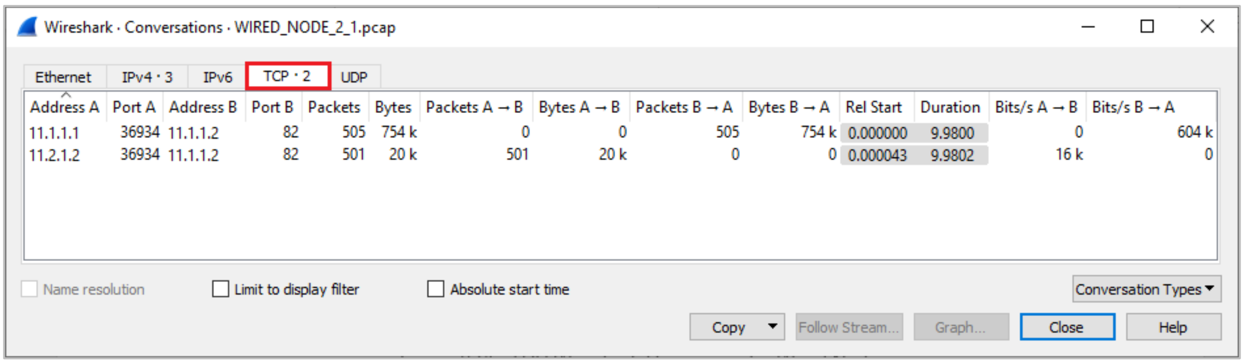
A network conversation is the traffic between two specific end points. For example, an IP conversation is all the traffic between two IP addresses. In Wireshark, Go to Statistics Menu→ Conversations as shown below Figure 8-95.



No.	Time	Source
2	0.000000	11.1.1.2
3	0.000043	11.2.1.2
4	0.000043	11.1.1.2
5	0.000043	11.1.1.2
6	0.000324	11.2.1.2
7	0.020000	11.1.1.2
8	0.020274	11.2.1.2

Figure 8-95: In Wireshark, Go to Statistics Menu >Conversations

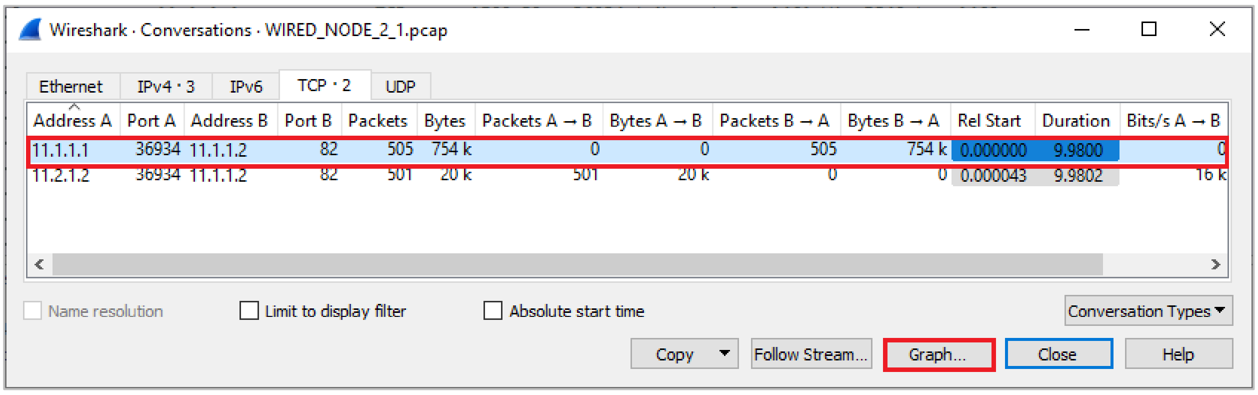
Different types of protocols will be available. User can select the specific conversation by going to the desired protocol. For example, in the following diagram, we have selected TCP.



Address A	Port A	Address B	Port B	Packets	Bytes	Packets A → B	Bytes A → B	Packets B → A	Bytes B → A	Rel Start	Duration	Bits/s A → B	Bits/s B → A
11.1.1.1	36934	11.1.1.2	82	505	754 k	0	0	505	754 k	0.000000	9.9800	0	604 k
11.2.1.2	36934	11.1.1.2	82	501	20 k	501	20 k	0	0	0.000043	9.9802	16 k	0

Figure 8-96: TCP Wireshark Conversion for Wired Nodes

User can also analyze each of the conversation and can create graphs by selecting them and clicking on **“Graph”.**



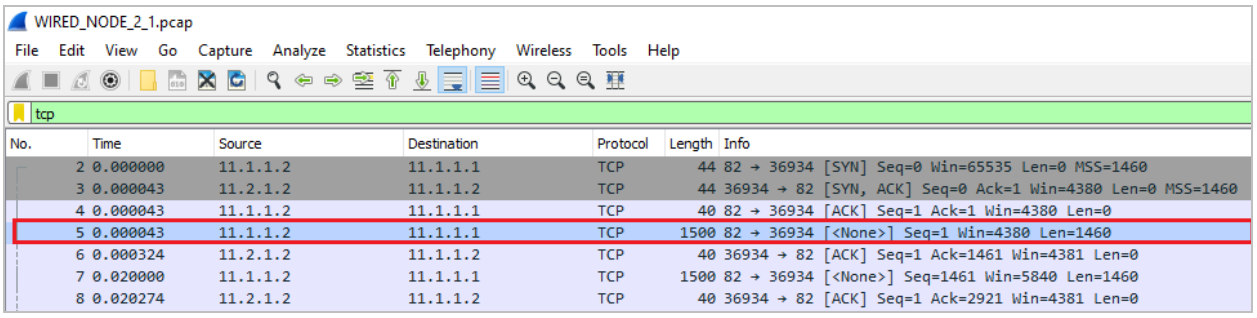
Address A	Port A	Address B	Port B	Packets	Bytes	Packets A → B	Bytes A → B	Packets B → A	Bytes B → A	Rel Start	Duration	Bits/s A → B
11.1.1.1	36934	11.1.1.2	82	505	754 k	0	0	505	754 k	0.000000	9.9800	0
11.2.1.2	36934	11.1.1.2	82	501	20 k	501	20 k	0	0	0.000043	9.9802	16 k

Figure 8-97: Select Graph in Wireshark Conversation

Different types of graphs are possible for Round Trip time, Throughput, Time/Sequence (Stevens), Time/Sequence (tcptrace) and Window Scaling

8.6.5 Window Scaling

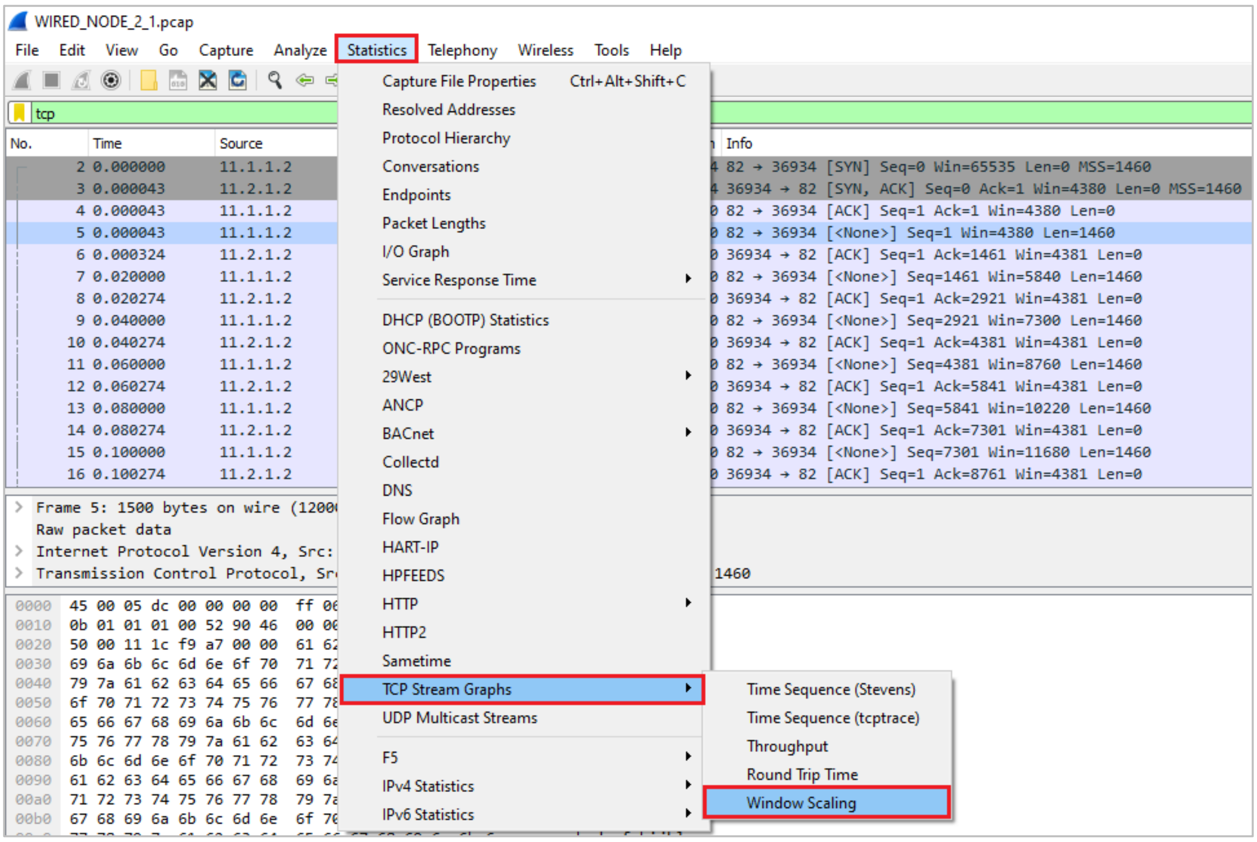
Click on data packet i.e. <None>.


WIRED_NODE_2_1.pcap
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
tcp

```
No.  Time      Source     Destination  Protocol  Length  Info
   2 0.000000  11.1.1.2   11.1.1.1     TCP          44 82 → 36934 [SYN] Seq=0 Win=65535 Len=0 MSS=1460
   3 0.000043  11.2.1.2   11.1.1.2     TCP          44 36934 → 82 [SYN, ACK] Seq=0 Ack=1 Win=4380 Len=0 MSS=1460
   4 0.000043  11.1.1.2   11.1.1.1     TCP          40 82 → 36934 [ACK] Seq=1 Ack=1 Win=4380 Len=0
   5 0.000043  11.1.1.2   11.1.1.1     TCP        1500 82 → 36934 [<None>] Seq=1 Win=4380 Len=1460
   6 0.000324  11.2.1.2   11.1.1.2     TCP          40 36934 → 82 [ACK] Seq=1 Ack=1461 Win=4381 Len=0
   7 0.020000  11.1.1.2   11.1.1.1     TCP        1500 82 → 36934 [<None>] Seq=1461 Win=5840 Len=1460
   8 0.020274  11.2.1.2   11.1.1.2     TCP          40 36934 → 82 [ACK] Seq=1 Ack=2921 Win=4381 Len=0
```

Figure 8-98: Select one data Packet <None>in Wireshark

Choose statistics→TCP Stream Graph→Window Scaling.



```
No.  Time      Source     Info
   2 0.000000  11.1.1.2   4 82 → 36934 [SYN] Seq=0 Win=65535 Len=0 MSS=1460
   3 0.000043  11.2.1.2   4 36934 → 82 [SYN, ACK] Seq=0 Ack=1 Win=4380 Len=0 MSS=1460
   4 0.000043  11.1.1.2   0 82 → 36934 [ACK] Seq=1 Ack=1 Win=4380 Len=0
   5 0.000043  11.1.1.2   0 82 → 36934 [<None>] Seq=1 Win=4380 Len=1460
   6 0.000324  11.2.1.2   0 36934 → 82 [ACK] Seq=1 Ack=1461 Win=4381 Len=0
   7 0.020000  11.1.1.2   0 82 → 36934 [<None>] Seq=1461 Win=5840 Len=1460
   8 0.020274  11.2.1.2   0 36934 → 82 [ACK] Seq=1 Ack=2921 Win=4381 Len=0
   9 0.040000  11.1.1.2   0 82 → 36934 [<None>] Seq=2921 Win=7300 Len=1460
  10 0.040274  11.2.1.2   0 36934 → 82 [ACK] Seq=1 Ack=4381 Win=4381 Len=0
  11 0.060000  11.1.1.2   0 82 → 36934 [<None>] Seq=4381 Win=8760 Len=1460
  12 0.060274  11.2.1.2   0 36934 → 82 [ACK] Seq=1 Ack=5841 Win=4381 Len=0
  13 0.080000  11.1.1.2   0 82 → 36934 [<None>] Seq=5841 Win=10220 Len=1460
  14 0.080274  11.2.1.2   0 36934 → 82 [ACK] Seq=1 Ack=7301 Win=4381 Len=0
  15 0.100000  11.1.1.2   0 82 → 36934 [<None>] Seq=7301 Win=11680 Len=1460
  16 0.100274  11.2.1.2   0 36934 → 82 [ACK] Seq=1 Ack=8761 Win=4381 Len=0

> Frame 5: 1500 bytes on wire (1200
  Raw packet data
> Internet Protocol Version 4, Src:
> Transmission Control Protocol, Sr                                   1460

0000  45 00 05 dc 00 00 00 00  ff 06
0010  0b 01 01 01 00 52 90 46  00 00
0020  50 00 11 1c f9 a7 00 00  61 62
0030  69 6a 6b 6c 6d 6e 6f 70  71 72
0040  79 7a 61 62 63 64 65 66  67 68
0050  6f 70 71 72 73 74 75 76  77 78
0060  65 66 67 68 69 6a 6b 6c  6d 6e
0070  75 76 77 78 79 7a 61 62  63 64
0080  6b 6c 6d 6e 6f 70 71 72  73 74
0090  61 62 63 64 65 66 67 68  69 6a
00a0  71 72 73 74 75 76 77 78  79 7a
00b0  67 68 69 6a 6b 6c 6d 6e  6f 70
```

Figure 8-99: Statistics>TCP Stream Graph>Window Scaling

Click on Switch Direction in the window scaling graph window.

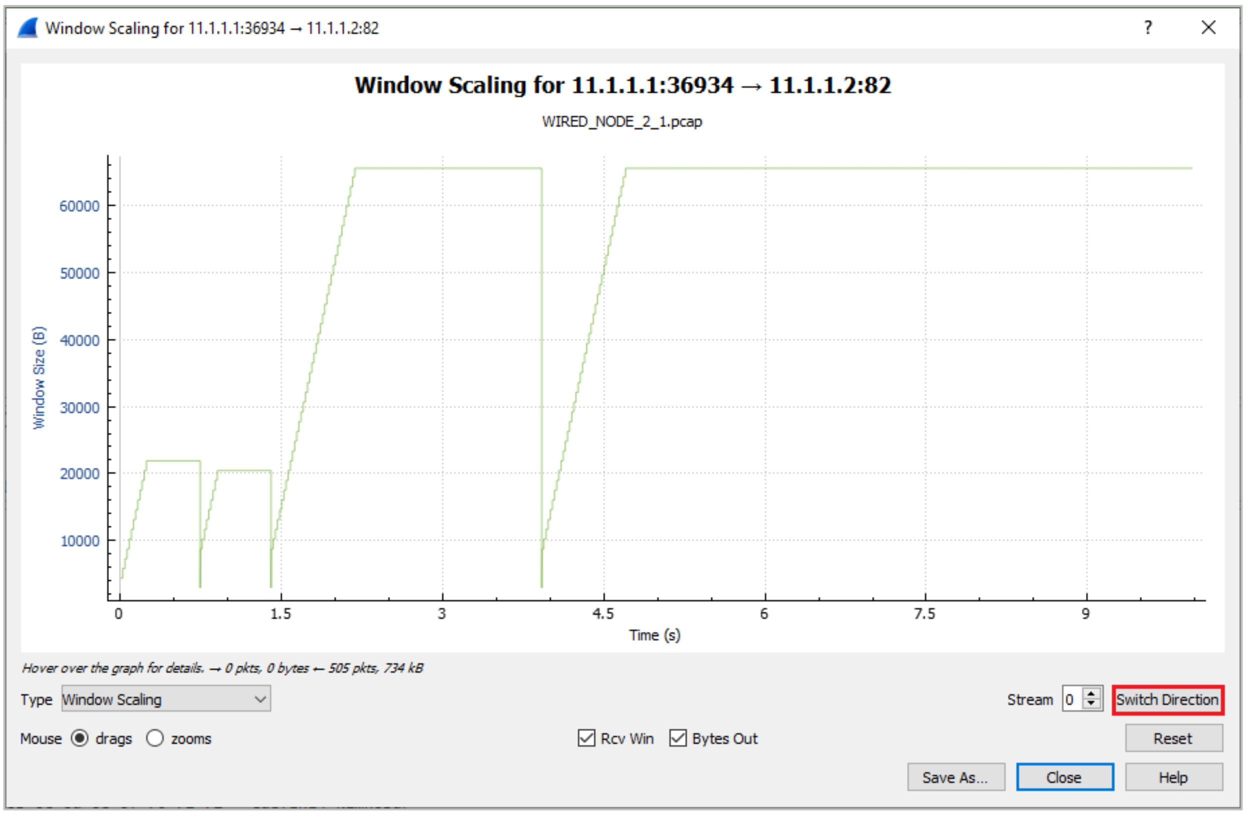
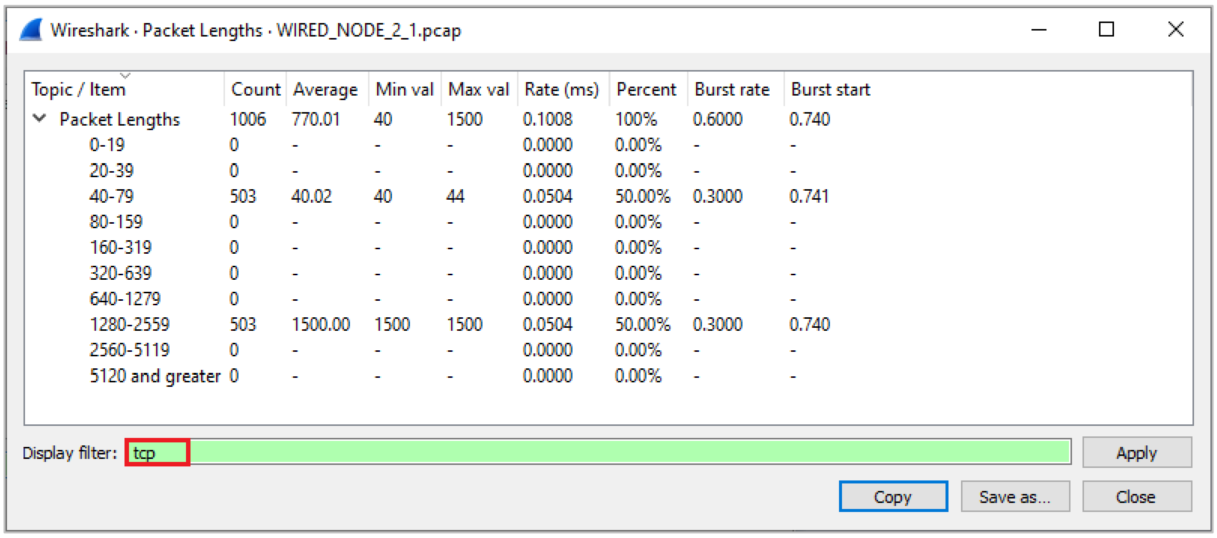


Figure 8-100: TCP Window Scaling Plot

8.6.5.1 Comparing the packet lengths

To analyze the packet sizes of all packets transmitted, go to ***Statistics Menu→Packet lengths***. Users can also set filter to analyze a collection of specific packets only. For example, *tcp* filter is set to obtain the packet length below Figure 8-101.



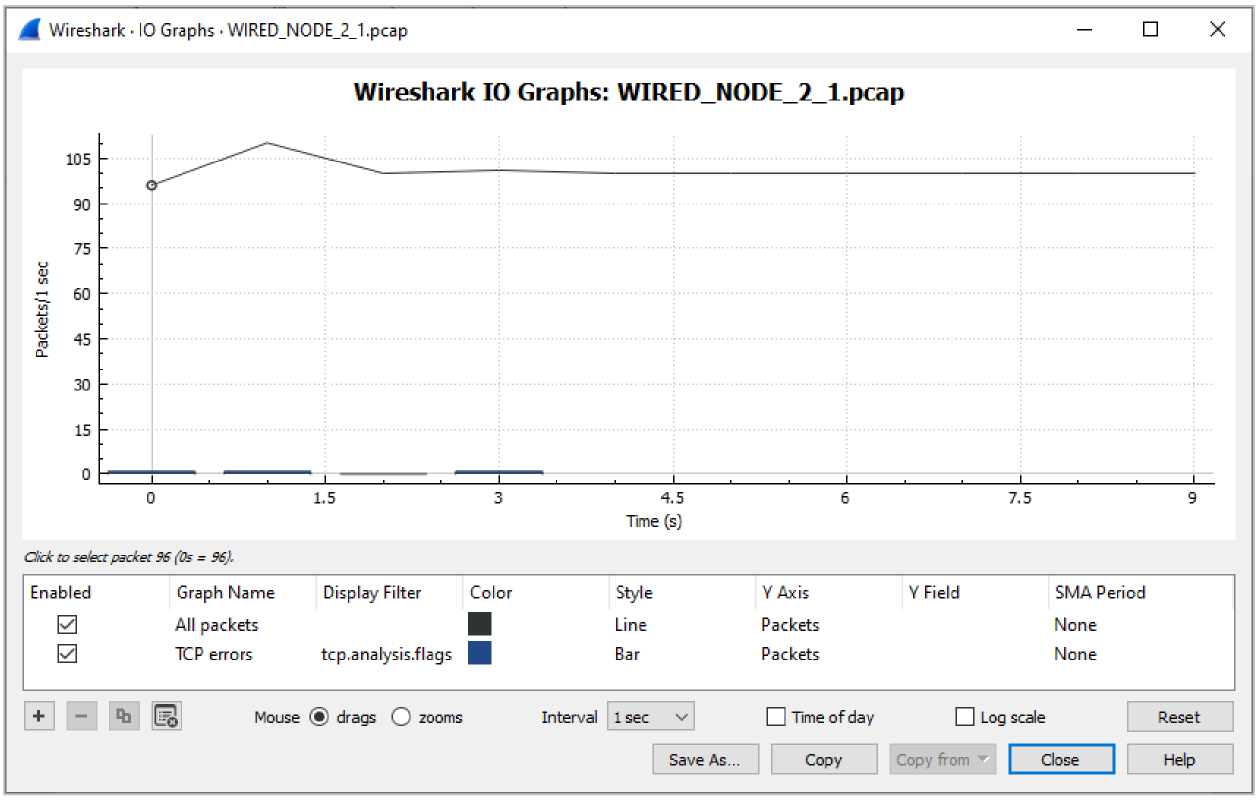
Topic / Item	Count	Average	Min val	Max val	Rate (ms)	Percent	Burst rate	Burst start
Packet Lengths	1006	770.01	40	1500	0.1008	100%	0.6000	0.740
0-19	0	-	-	-	0.0000	0.00%	-	-
20-39	0	-	-	-	0.0000	0.00%	-	-
40-79	503	40.02	40	44	0.0504	50.00%	0.3000	0.741
80-159	0	-	-	-	0.0000	0.00%	-	-
160-319	0	-	-	-	0.0000	0.00%	-	-
320-639	0	-	-	-	0.0000	0.00%	-	-
640-1279	0	-	-	-	0.0000	0.00%	-	-
1280-2559	503	1500.00	1500	1500	0.0504	50.00%	0.3000	0.740
2560-5119	0	-	-	-	0.0000	0.00%	-	-
5120 and greater	0	-	-	-	0.0000	0.00%	-	-

Display filter: tcp

Figure 8-101: Comparing the packet lengths in Wireshark

8.6.5.2 Creating IO graphs

To get the graph, go to ***Statistics Menu → IO Graph***.



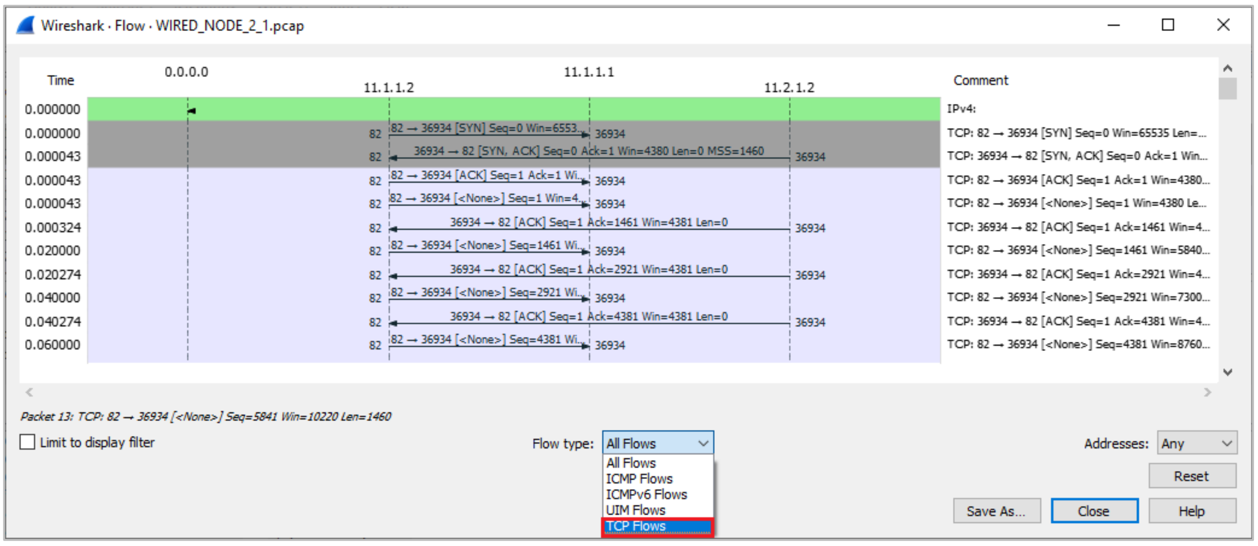
Click to select packet 96 (0s = 96).

Enabled	Graph Name	Display Filter	Color	Style	Y Axis	Y Field	SMA Period
✓	All packets		■	Line	Packets		None
✓	TCP errors	tcp.analysis.flags	■	Bar	Packets		None

Figure 8-102: Statistics Menu > IO Graph in Wireshark

8.6.5.3 Creating Flow graphs

The flow graph feature provides a quick and easy to use way of checking connections between a client and a server. It can show where there might be issues with a TCP connection, such as timeouts, re-transmitted frames, or dropped connections. To access flow graph, go to ***Statistics Menu → Flow Graph*** and select the flow type. By default, you can see the flow graph of all the packets. To get the TCP flow, select TCP flow in "Flow Type" dropdown box and you will obtain the flow as shown Figure 8-103.


Wireshark · Flow · WIRED_NODE_2_1.pcap
0.0.0.0
11.1.1.2
11.1.1.1
11.2.1.2
82 → 36934 [SYN] Seq=0 Win=6553...
36934 → 82 [SYN, ACK] Seq=0 Ack=1 Win=4380 Len=0 MSS=1460
82 → 36934 [ACK] Seq=1 Ack=1 Wi...
82 → 36934 [<None>] Seq=1 Win=4...
36934 → 82 [ACK] Seq=1 Ack=1461 Win=4381 Len=0
82 → 36934 [<None>] Seq=1461 Wi...
36934 → 82 [ACK] Seq=1 Ack=2921 Win=4381 Len=0
82 → 36934 [<None>] Seq=2921 Wi...
36934 → 82 [ACK] Seq=1 Ack=4381 Win=4381 Len=0
82 → 36934 [<None>] Seq=4381 Wi...
Limit to display filter
Flow type:
All Flows
ICMP Flows
ICMPv6 Flows
UIM Flows
TCP Flows
Addresses:
Any
Reset
Save As...
Close
Help

Time	Comment
0.000000	IPv4:
0.000000	TCP: 82 → 36934 [SYN] Seq=0 Win=65535 Len=...
0.000043	TCP: 36934 → 82 [SYN, ACK] Seq=0 Ack=1 Win...
0.000043	TCP: 82 → 36934 [ACK] Seq=1 Ack=1 Win=4380...
0.000043	TCP: 82 → 36934 [<None>] Seq=1 Win=4380 Le...
0.000324	TCP: 36934 → 82 [ACK] Seq=1 Ack=1461 Win=4...
0.020000	TCP: 82 → 36934 [<None>] Seq=1461 Win=5840...
0.020274	TCP: 36934 → 82 [ACK] Seq=1 Ack=2921 Win=4...
0.040000	TCP: 82 → 36934 [<None>] Seq=2921 Win=7300...
0.040274	TCP: 36934 → 82 [ACK] Seq=1 Ack=4381 Win=4...
0.060000	TCP: 82 → 36934 [<None>] Seq=4381 Win=8760...

Packet 13: TCP: 82 → 36934 [<None>] Seq=5841 Win=10220 Len=1460

Figure 8-103: Statistics Menu > Flow Graph in Wireshark

9 Writing Custom Code in NetSim

9.1 Writing your own code

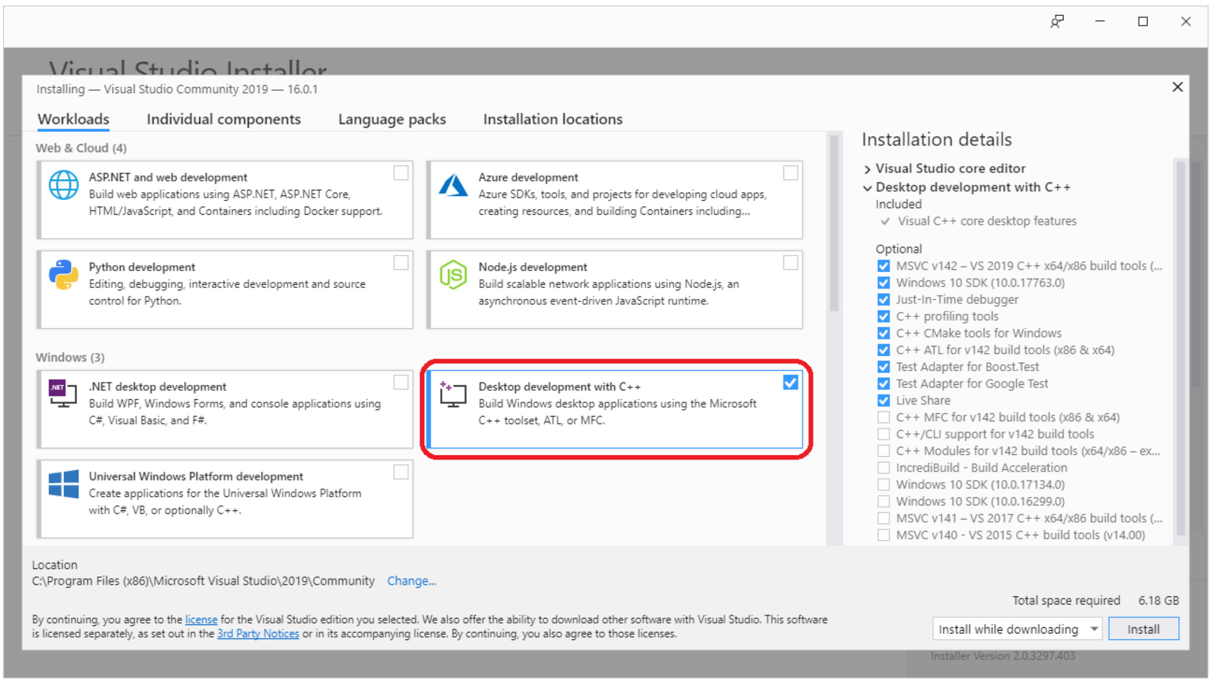
NetSim allows the user to write the custom code for all the protocols by creating a DLL (Dynamic Link Library) for their custom code.

There are various important steps in this process, and each of these steps has various options as explained in the subsequent pages.

9.1.1 Microsoft Visual Studio 2019 Installation Settings

NetSim requires only a few components of Visual Studio Community 2019 edition to be installed. Upon starting the installer:

1. Under the **Workloads** tab users can select **Desktop Development with C++** as shown below Figure 9-1.



Visual Studio Installer

Installing — Visual Studio Community 2019 — 16.0.1

Workloads | Individual components | Language packs | Installation locations

Web & Cloud (4)

- ☐ ASP.NET and web development — Build web applications using ASP.NET, ASP.NET Core, HTML/JavaScript, and Containers including Docker support.
- ☐ Azure development — Azure SDKs, tools, and projects for developing cloud apps, creating resources, and building Containers including...
- ☐ Python development — Editing, debugging, interactive development and source control for Python.
- ☐ Node.js development — Build scalable network applications using Node.js, an asynchronous event-driven JavaScript runtime.

Windows (3)

- ☐ .NET desktop development — Build WPF, Windows Forms, and console applications using C#, Visual Basic, and F#.
- ☑ Desktop development with C++ — Build Windows desktop applications using the Microsoft C++ toolset, ATL, or MFC.
- ☐ Universal Windows Platform development — Create applications for the Universal Windows Platform with C#, VB, or optionally C++.

Installation details

> Visual Studio core editor

∨ Desktop development with C++

Included

✓ Visual C++ core desktop features

Optional

- ☑ MSVC v142 – VS 2019 C++ x64/x86 build tools (...
- ☑ Windows 10 SDK (10.0.17763.0)
- ☑ Just-In-Time debugger
- ☑ C++ profiling tools
- ☑ C++ CMake tools for Windows
- ☑ C++ ATL for v142 build tools (x86 & x64)
- ☑ Test Adapter for Boost.Test
- ☑ Test Adapter for Google Test
- ☑ Live Share
- ☐ C++ MFC for v142 build tools (x86 & x64)
- ☐ C++/CLI support for v142 build tools
- ☐ C++ Modules for v142 build tools (x64/x86 – ex...
- ☐ IncrediBuild - Build Acceleration
- ☐ Windows 10 SDK (10.0.17134.0)
- ☐ Windows 10 SDK (10.0.16299.0)
- ☐ MSVC v141 – VS 2017 C++ x64/x86 build tools (...
- ☐ MSVC v140 - VS 2015 C++ build tools (v14.00)

Location
C:\Program Files (x86)\Microsoft Visual Studio\2019\Community Change...

By continuing, you agree to the license for the Visual Studio edition you selected. We also offer the ability to download other software with Visual Studio. This software is licensed separately, as set out in the 3rd Party Notices or in its accompanying license. By continuing, you also agree to those licenses.

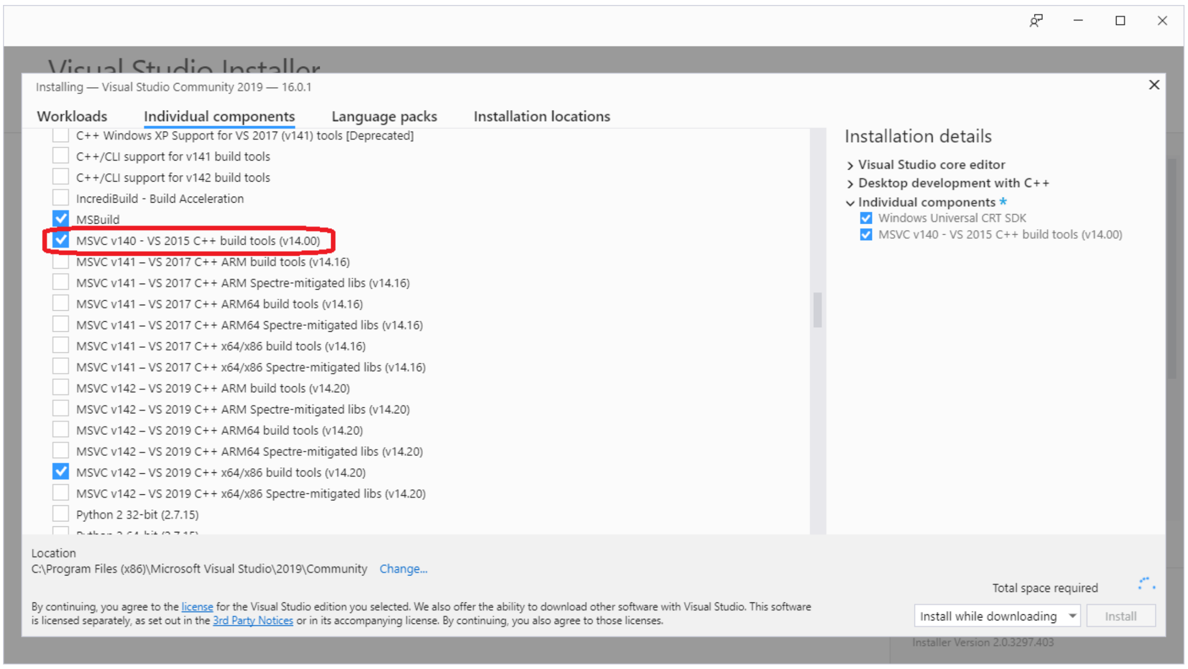
Total space required 6.18 GB

Install while downloading | Install

Installer Version 2.0.3297.403

Figure 9-1: In Workloads tab select Desktop Development with C++

2. Under the **Individual components** tab select **VC++2015.3 V140 toolset for desktop (x86, x64).**



Visual Studio Installer

Installing — Visual Studio Community 2019 — 16.0.1

Workloads | Individual components | Language packs | Installation locations

- [ ] C++ Windows XP Support for VS 2017 (v141) tools [Deprecated]
- [ ] C++/CLI support for v141 build tools
- [ ] C++/CLI support for v142 build tools
- [ ] IncrediBuild - Build Acceleration
- [✓] MSBuild
- [✓] MSVC v140 - VS 2015 C++ build tools (v14.00)
- [ ] MSVC v141 – VS 2017 C++ ARM build tools (v14.16)
- [ ] MSVC v141 – VS 2017 C++ ARM Spectre-mitigated libs (v14.16)
- [ ] MSVC v141 – VS 2017 C++ ARM64 build tools (v14.16)
- [ ] MSVC v141 – VS 2017 C++ ARM64 Spectre-mitigated libs (v14.16)
- [ ] MSVC v141 – VS 2017 C++ x64/x86 build tools (v14.16)
- [ ] MSVC v141 – VS 2017 C++ x64/x86 Spectre-mitigated libs (v14.16)
- [ ] MSVC v142 – VS 2019 C++ ARM build tools (v14.20)
- [ ] MSVC v142 – VS 2019 C++ ARM Spectre-mitigated libs (v14.20)
- [ ] MSVC v142 – VS 2019 C++ ARM64 build tools (v14.20)
- [ ] MSVC v142 – VS 2019 C++ ARM64 Spectre-mitigated libs (v14.20)
- [✓] MSVC v142 – VS 2019 C++ x64/x86 build tools (v14.20)
- [ ] MSVC v142 – VS 2019 C++ x64/x86 Spectre-mitigated libs (v14.20)
- [ ] Python 2 32-bit (2.7.15)

Installation details

› Visual Studio core editor
› Desktop development with C++
⌄ Individual components *
  - [✓] Windows Universal CRT SDK
  - [✓] MSVC v140 - VS 2015 C++ build tools (v14.00)

Location
C:\Program Files (x86)\Microsoft Visual Studio\2019\Community Change...

By continuing, you agree to the license for the Visual Studio edition you selected. We also offer the ability to download other software with Visual Studio. This software is licensed separately, as set out in the 3rd Party Notices or in its accompanying license. By continuing, you also agree to those licenses.

Total space required

Install while downloading | Install

Installer Version 2.0.3297.403

Figure 9-2: In Individual components tab select VC++2015.3 V140 toolset for desktop (x86, x64).

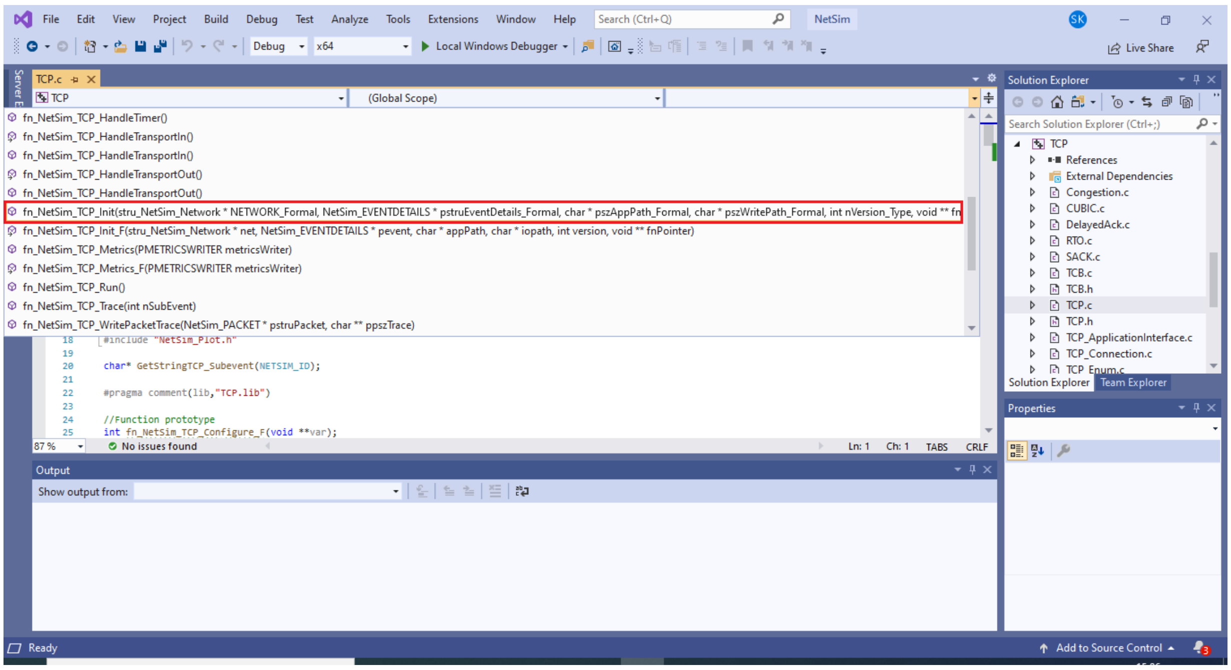
Modifying code

DLL is the shared library concept, implemented by Microsoft. All DLL files have a .dll file extension. DLLs provide a mechanism for sharing code and data to upgrade functionality without requiring applications to be re-linked or re-compiled. It is not possible to directly execute a DLL, since it requires an EXE for the operating system to load it through an entry point. NetSim requires Visual Studio Compiler for building DLL's.

Note: *Make sure that Visual Studio 2015 or above is installed in your system.*

Refer **section 4.13** section "How does a user open and modify source codes" to open NetSim Source Codes

1. After this you may modify the source codes of any project. You can also add new files to the project if required. As an example, let us make a simple source code modification to TCP. Inside Solution Explorer pane in Visual Studio, double click on TCP project. Then open TCP.c file by double clicking on it. Using the drop down list of functions that are part of the current file, choose **fn_Netsim_TCP_Init()**.

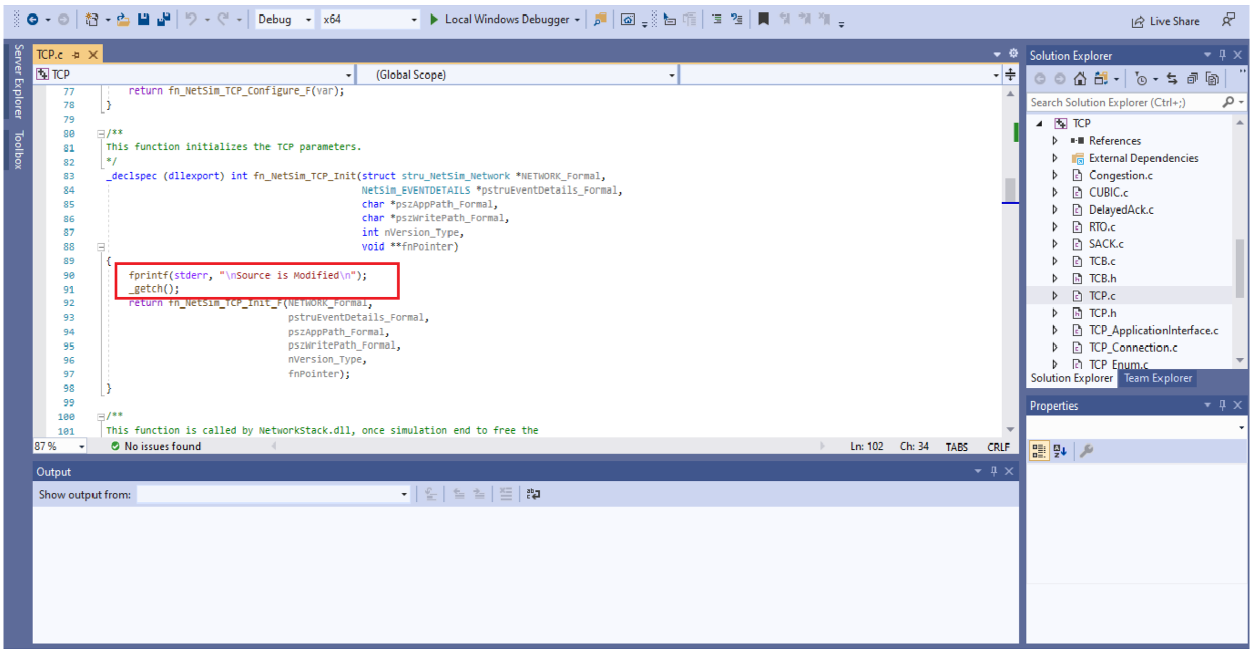


```
fn_NetSim_TCP_HandleTimer()
fn_NetSim_TCP_HandleTransportIn()
fn_NetSim_TCP_HandleTransportIn()
fn_NetSim_TCP_HandleTransportOut()
fn_NetSim_TCP_HandleTransportOut()
fn_NetSim_TCP_Init(stru_NetSim_Network * NETWORK_Formal, NetSim_EVENTDETAILS * pstruEventDetails_Formal, char * pszAppPath_Formal, char * pszWritePath_Formal, int nVersion_Type, void ** fn
fn_NetSim_TCP_Init_F(stru_NetSim_Network * net, NetSim_EVENTDETAILS * pevent, char * appPath, char * iopath, int version, void ** fnPointer)
fn_NetSim_TCP_Metrics(PMETRICSWRITER metricsWriter)
fn_NetSim_TCP_Metrics_F(PMETRICSWRITER metricsWriter)
fn_NetSim_TCP_Run()
fn_NetSim_TCP_Trace(int nSubEvent)
fn_NetSim_TCP_WritePacketTrace(NetSim_PACKET * pstruPacket, char ** ppszTrace)
```

```
#include "NetSim_Plot.h"

char* GetStringTCP_Subevent(NETSIM_ID);

#pragma comment(lib,"TCP.lib")

//Function prototype
int fn_NetSim_TCP_Configure_F(void **var);
```

Figure 9-3: Select fn_Netsim_TCP_Init() in TCP.C in Visual Studio

2. Add the line **fprintf(stderr, "\nSource is Modified\n");** statement inside the **fn_Netsim_TCP_Init()** function as shown below to print “Source is modified”. **_getch();** is added in the next line for the simulation to wait until it gets a user input.

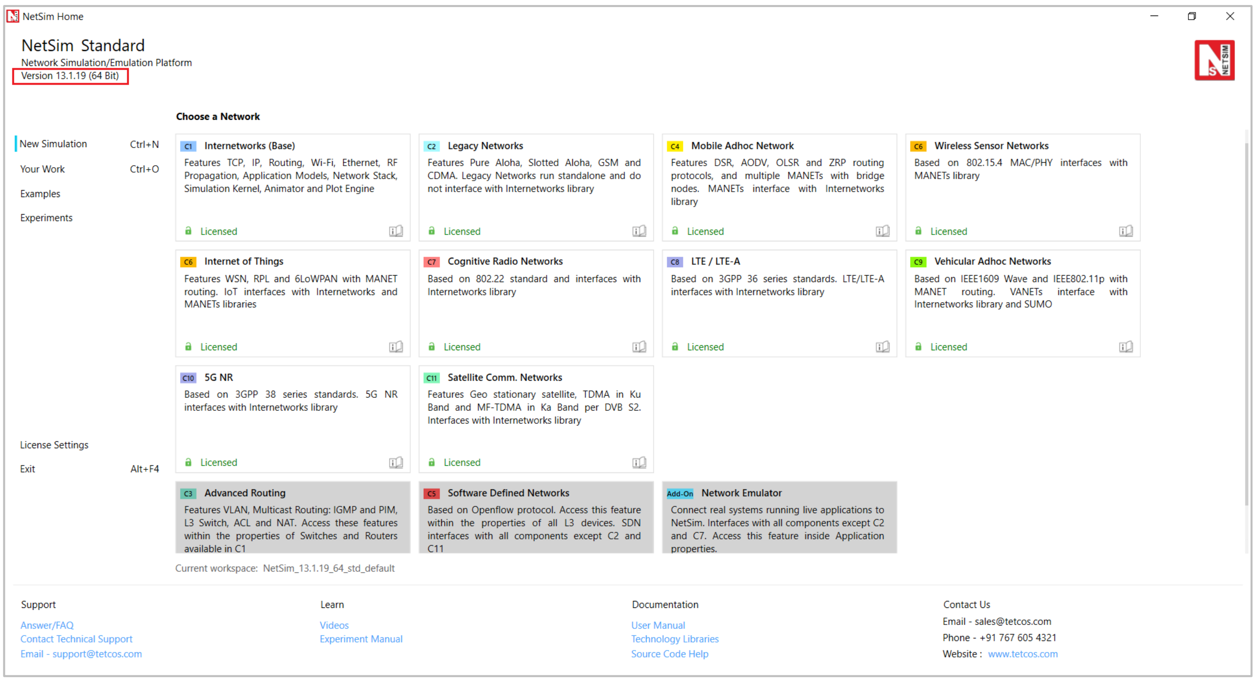


```
    return fn_NetSim_TCP_Configure_F(var);
}

/**
This function initializes the TCP parameters.
*/
_declspec (dllexport) int fn_NetSim_TCP_Init(struct stru_NetSim_Network *NETWORK_Formal,
                                             NetSim_EVENTDETAILS *pstruEventDetails_Formal,
                                             char *pszAppPath_Formal,
                                             char *pszWritePath_Formal,
                                             int nVersion_Type,
                                             void **fnPointer)
{
    fprintf(stderr, "\nSource is Modified\n");
    _getch();
    return fn_NetSim_TCP_Init_F(NETWORK_Formal,
                                pstruEventDetails_Formal,
                                pszAppPath_Formal,
                                pszWritePath_Formal,
                                nVersion_Type,
                                fnPointer);
}

/**
This function is called by NetworkStack.dll, once simulation end to free the
```

Figure 9-4: Source Code modified in fn_Netsim_TCP_Init() function in TCP project

9.1.2 Building Dlls

1. Identify the build of NetSim that is installed in your system from NetSim Home Screen as shown below.



NetSim Home

NetSim Standard
Network Simulation/Emulation Platform
Version 13.1.19 (64 Bit)

Choose a Network

New Simulation Ctrl+N
Your Work Ctrl+O
Examples
Experiments
License Settings
Exit Alt+F4

C1 Internetworks (Base)
Features TCP, IP, Routing, Wi-Fi, Ethernet, RF Propagation, Application Models, Network Stack, Simulation Kernel, Animator and Plot Engine
Licensed

C2 Legacy Networks
Features Pure Aloha, Slotted Aloha, GSM and CDMA. Legacy Networks run standalone and do not interface with Internetworks library
Licensed

C4 Mobile Adhoc Network
Features DSR, AODV, OLSR and ZRP routing protocols, and multiple MANETs with bridge nodes. MANETs interface with Internetworks library
Licensed

C5 Wireless Sensor Networks
Based on 802.15.4 MAC/PHY interfaces with MANETs library
Licensed

C6 Internet of Things
Features WSN, RPL and 6LoWPAN with MANET routing. IoT interfaces with Internetworks and MANETs libraries
Licensed

C7 Cognitive Radio Networks
Based on 802.22 standard and interfaces with Internetworks library
Licensed

C8 LTE / LTE-A
Based on 3GPP 36 series standards. LTE/LTE-A interfaces with Internetworks library
Licensed

C9 Vehicular Adhoc Networks
Based on IEEE1609 Wave and IEEE802.11p with MANET routing. VANETs interface with Internetworks library and SUMO
Licensed

C10 5G NR
Based on 3GPP 38 series standards. 5G NR interfaces with Internetworks library
Licensed

C11 Satellite Comm. Networks
Features Geo stationary satellite, TDMA in Ku Band and MF-TDMA in Ka Band per DVB S2. Interfaces with Internetworks library
Licensed

C3 Advanced Routing
Features VLAN, Multicast Routing: IGMP and PIM, L3 Switch, ACL and NAT. Access these features within the properties of Switches and Routers available in C1

C5 Software Defined Networks
Based on Openflow protocol. Access this feature within the properties of all L3 devices. SDN interfaces with all components except C2 and C11

Add-On Network Emulator
Connect real systems running live applications to NetSim. Interfaces with all components except C2 and C7. Access this feature inside Application properties.

Current workspace: NetSim_13.1.19_64_std_default

Support
Answer/FAQ
Contact Technical Support
Email - support@tetcos.com

Learn
Videos
Experiment Manual

Documentation
User Manual
Technology Libraries
Source Code Help

Contact Us
Email - sales@tetcos.com
Phone - +91 767 605 4321
Website : www.tetcos.com

Figure 9-5: In NetSim Home Screen Identify the build of NetSim

2. By default, x64 is choose for 64-bit version of NetSim. These changes will be automatically applied to all projects that are displayed in the Solution Figure 9-6.

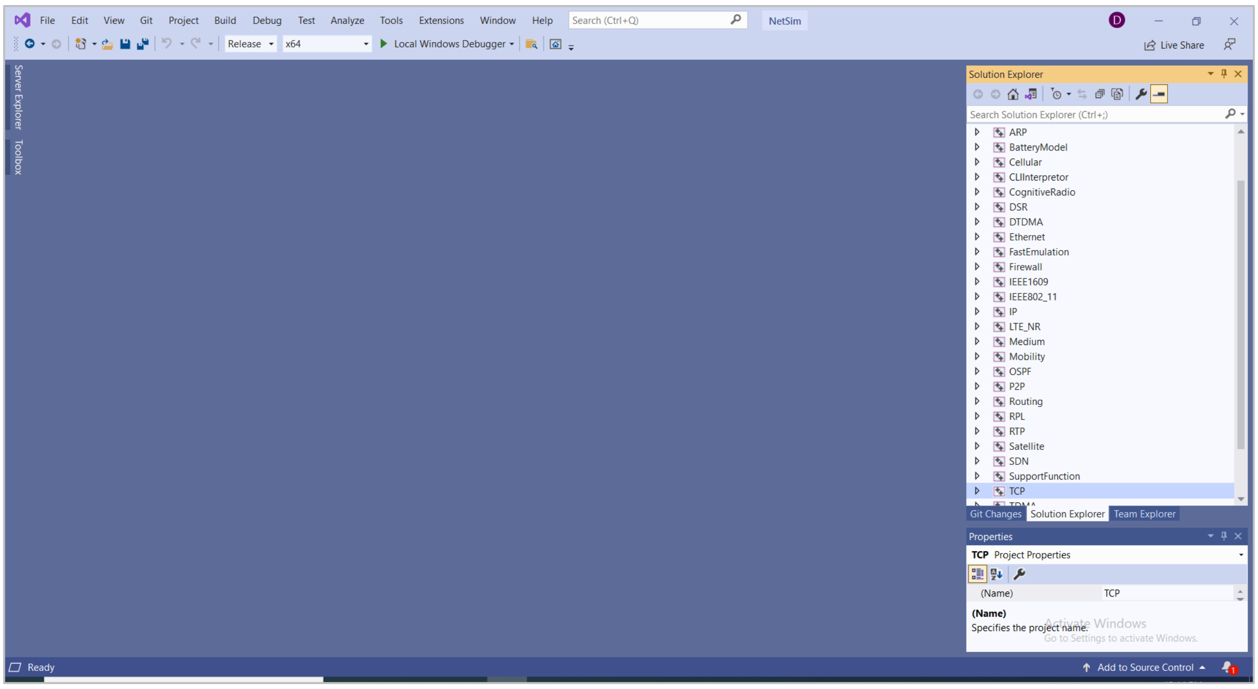
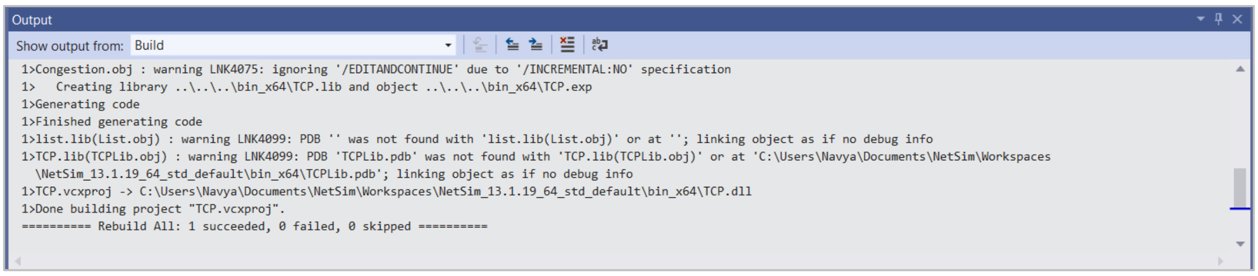


Figure 9-6: Based on the build of NetSim select x64 in Visual studio

3. Now rebuild the network by right clicking on the project header and selecting Rebuild creates a Dll file in the bin folder of NetSim's current workspace path.

< C:\Users\PC\Documents\NetSim\Workspaces\<Your default workspace>\bin_x64> for 64-bit which contains your modifications. If build is successful a message similar to the following will be displayed in the output window as shown below.

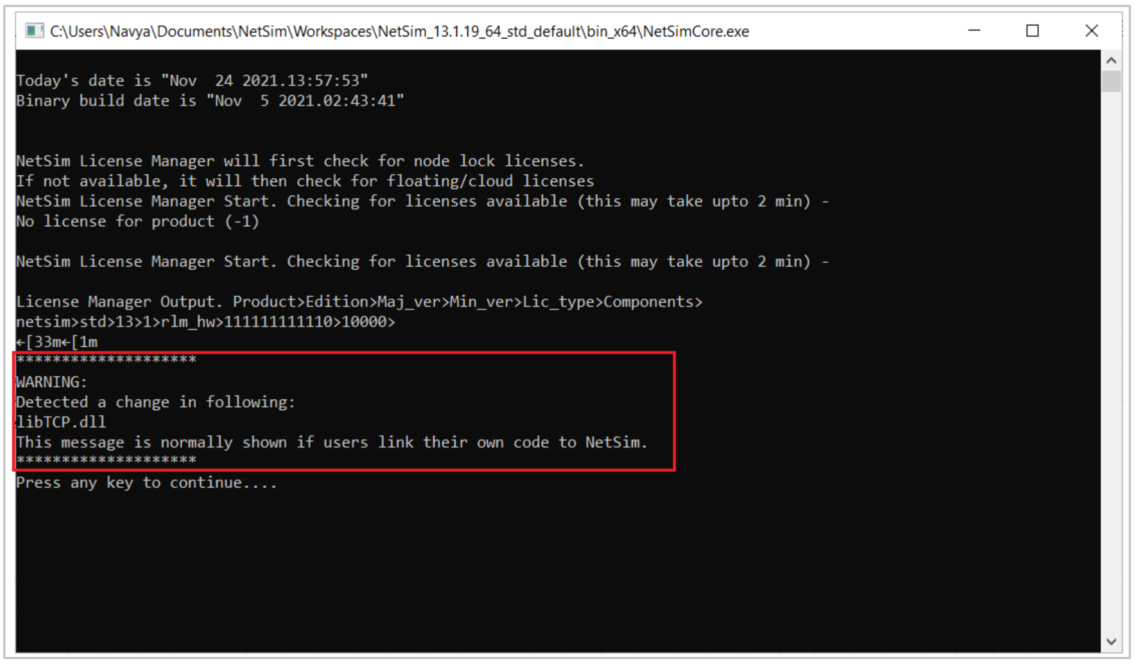


```
1>Congestion.obj : warning LNK4075: ignoring '/EDITANDCONTINUE' due to '/INCREMENTAL:NO' specification
1>   Creating library ..\..\..\bin_x64\TCP.lib and object ..\..\..\bin_x64\TCP.exp
1>Generating code
1>Finished generating code
1>list.lib(List.obj) : warning LNK4099: PDB '' was not found with 'list.lib(List.obj)' or at ''; linking object as if no debug info
1>TCP.lib(TCPLib.obj) : warning LNK4099: PDB 'TCPLib.pdb' was not found with 'TCP.lib(TCPLib.obj)' or at 'C:\Users\Navya\Documents\NetSim\Workspaces
  \NetSim_13.1.19_64_std_default\bin_x64\TCPLib.pdb'; linking object as if no debug info
1>TCP.vcxproj -> C:\Users\Navya\Documents\NetSim\Workspaces\NetSim_13.1.19_64_std_default\bin_x64\TCP.dll
1>Done building project "TCP.vcxproj".
========== Rebuild All: 1 succeeded, 0 failed, 0 skipped ==========
```

Figure 9-7: Build is successful a message similar to the following will be displayed in the output window

9.1.3 Running Simulation

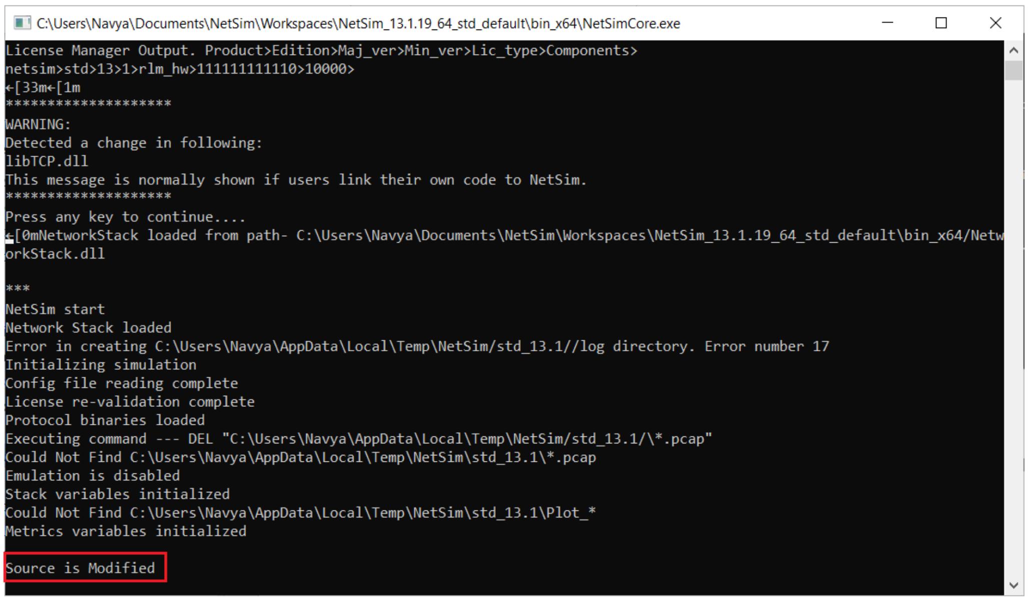
1. After rebuilding the code, user can run the simulation via GUI (Please refer **section 3**). In this case, user can create a scenario in any network which involves TCP protocol. Running the simulation with the custom DLL will initially display a warning message as shown below.


C:\Users\Navya\Documents\NetSim\Workspaces\NetSim_13.1.19_64_std_default\bin_x64\NetSimCore.exe

```
Today's date is "Nov  24 2021.13:57:53"
Binary build date is "Nov  5 2021.02:43:41"


NetSim License Manager will first check for node lock licenses.
If not available, it will then check for floating/cloud licenses
NetSim License Manager Start. Checking for licenses available (this may take upto 2 min) -
No license for product (-1)

NetSim License Manager Start. Checking for licenses available (this may take upto 2 min) -

License Manager Output. Product>Edition>Maj_ver>Min_ver>Lic_type>Components>
netsim>std>13>1>rlm_hw>111111111110>10000>
←[33m←[1m
********************
WARNING:
Detected a change in following:
libTCP.dll
This message is normally shown if users link their own code to NetSim.
********************
Press any key to continue....
```

Figure 9-8: Modified Project display a DLL warning message to NetSim Console

2. The warning message lists the Dll files which have been modified in the bin folder (bin\bin_x64 for 64-bit) of NetSim's current workspace path. After pressing any key, the statement "Source is modified" will be printed to console as shown below.



C:\Users\Navya\Documents\NetSim\Workspaces\NetSim_13.1.19_64_std_default\bin_x64\NetSimCore.exe

```
License Manager Output. Product>Edition>Maj_ver>Min_ver>Lic_type>Components>
netsim>std>13>1>rlm_hw>111111111110>10000>
←[33m←[1m
********************
WARNING:
Detected a change in following:
libTCP.dll
This message is normally shown if users link their own code to NetSim.
********************
Press any key to continue....
←[0mNetworkStack loaded from path- C:\Users\Navya\Documents\NetSim\Workspaces\NetSim_13.1.19_64_std_default\bin_x64/Netw
orkStack.dll

***
NetSim start
Network Stack loaded
Error in creating C:\Users\Navya\AppData\Local\Temp\NetSim/std_13.1//log directory. Error number 17
Initializing simulation
Config file reading complete
License re-validation complete
Protocol binaries loaded
Executing command --- DEL "C:\Users\Navya\AppData\Local\Temp\NetSim/std_13.1/\*.pcap"
Could Not Find C:\Users\Navya\AppData\Local\Temp\NetSim\std_13.1\*.pcap
Emulation is disabled
Stack variables initialized
Could Not Find C:\Users\Navya\AppData\Local\Temp\NetSim\std_13.1\Plot_*
Metrics variables initialized

Source is Modified
```

Figure 9-9: Printf Statement written to console

3. Press any key to proceed with the simulation.
4. The warning message will not be displayed if no Dll's are modified in the bin folder of current workspace path (bin\bin_x64 for 64-bit).

9.1.4 Source Code Dependencies

The following are the list of projects that are part of NetSim source codes present in **<NetSim_Install_Directory>/src/Simulation** directory and their dependencies:

PROJECT	DEPENDENCY
Application	IP
Cellular	Application
CLIInterpertor	Firewall, IP
Cognitive Radio	Application
Ethernet	Firewall
IEEE802_11	Battery Model
OSPF	IP
Routing	IP
RPL	IP
ZigBee	Battery Model
ZRP	IP
Aloha	-
AODV	-
ARP`	-
Battery Model	-
CSMACD	-
DSR	-
Firewall	-
IEEE1609	-
IP	-
LTE NR	
Mobility	-
P2P	-
SDN	-
Support Function	-

TCP	-
Token_BR	-
UDP	-
UWAN	
DTDMA	-
TDMA	-
Satellite Comm. Networks	-

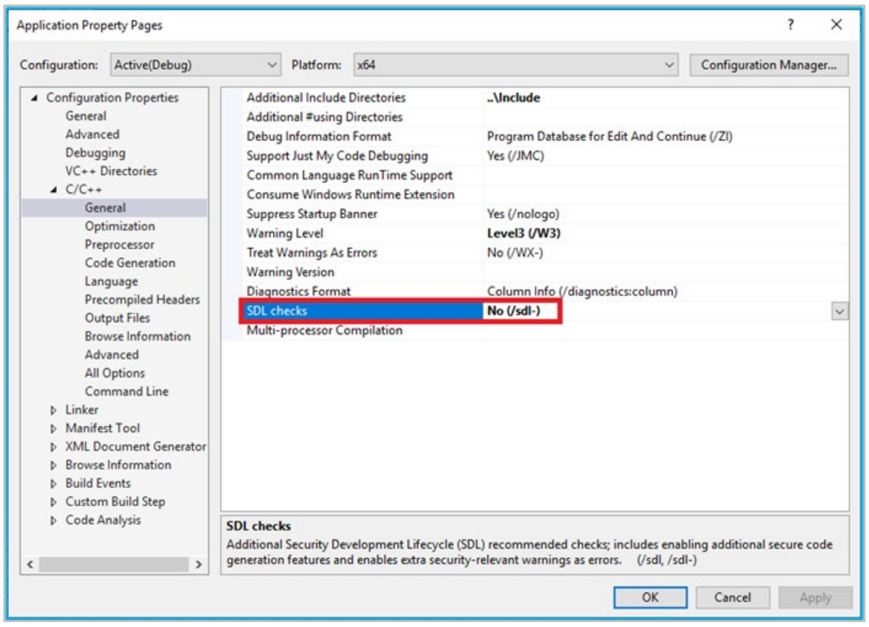
Table 9-1: Source Code Dependencies

For E.g.: To perform modifications to Application Project, IP folder will also be required in addition to lib folder, Include folder and NetSim.sln file.

9.1.5 Enabling Additional Security Checks

SDL – Security Development Lifecycle checks adds recommended Security Development Lifecycle. These checks include extra security-relevant warnings as errors, and additional secure code-generation features.

/sdl enables a superset of the baseline security checks provided by /GS and overrides /GS-. By default, **/sdl** is off. **/sdl-** disables the additional security checks.


Application Property Pages

Configuration: Active(Debug) Platform: x64 Configuration Manager...

- Configuration Properties
  - General
  - Advanced
  - Debugging
  - VC++ Directories
  - C/C++
    - General
    - Optimization
    - Preprocessor
    - Code Generation
    - Language
    - Precompiled Headers
    - Output Files
    - Browse Information
    - Advanced
    - All Options
    - Command Line
  - Linker
  - Manifest Tool
  - XML Document Generator
  - Browse Information
  - Build Events
  - Custom Build Step
  - Code Analysis

Additional Include Directories	..\Include
Additional #using Directories	
Debug Information Format	Program Database for Edit And Continue (/ZI)
Support Just My Code Debugging	Yes (/JMC)
Common Language RunTime Support	
Consume Windows Runtime Extension	
Suppress Startup Banner	Yes (/nologo)
Warning Level	Level3 (/W3)
Treat Warnings As Errors	No (/WX-)
Warning Version	
Diagnostics Format	Column Info (/diagnostics:column)
SDL checks	No (/sdl-)
Multi-processor Compilation	

SDL checks
Additional Security Development Lifecycle (SDL) recommended checks; includes enabling additional secure code generation features and enables extra security-relevant warnings as errors. (/sdl, /sdl-)

Figure 9-10: Enabling Additional Security Checks application properties window

If SDL checks is enabled (/sdl), following warnings will be treated as errors:

Warning enabled by /sdl	Equivalent command-line switch	Description
C4146	/we4146	A unary minus operator was applied to an unsigned type, resulting in an unsigned result.
C4308	/we4308	A negative integral constant converted to unsigned type, resulting in a possibly meaningless result.
C4532	/we4532	Use of continue, break or goto keywords in a __finally/finally block has undefined behavior during abnormal termination.
C4533	/we4533	Code initializing a variable will not be executed.
C4700	/we4700	Use of an uninitialized local variable.
C4703	/we4703	Use of a potentially uninitialized local pointer variable.

C4789	/we4789	Buffer overrun when specific C run-time (CRT) functions are used.
C4995	/we4995	Use of a function marked with pragma deprecated.
C4996	/we4996	Use of a function marked as deprecated.

Table 9-2: SDL Warnings Messages

Reference: https://docs.microsoft.com/en-us/cpp/build/reference/sdl-enable-additional-security-checks?view=vs-2019

9.2 Implementing your code - Examples

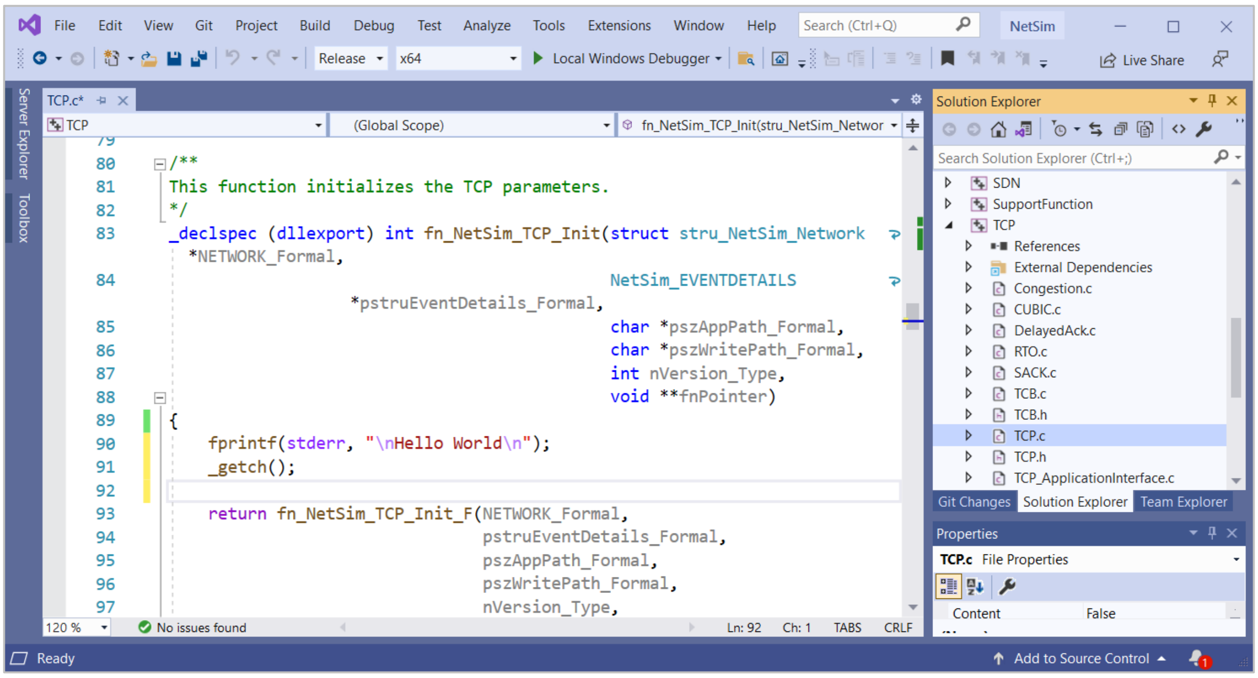
9.2.1 Hello World Program

Objective: Print Hello World from TCP protocol.

Implementation: Add fprintf (stderr, "<MESSAGE>") statement inside the source code of TCP as shown below to print "Hello World" when custom built dll is executing.

fprintf(stderr, "\nHello World\n");

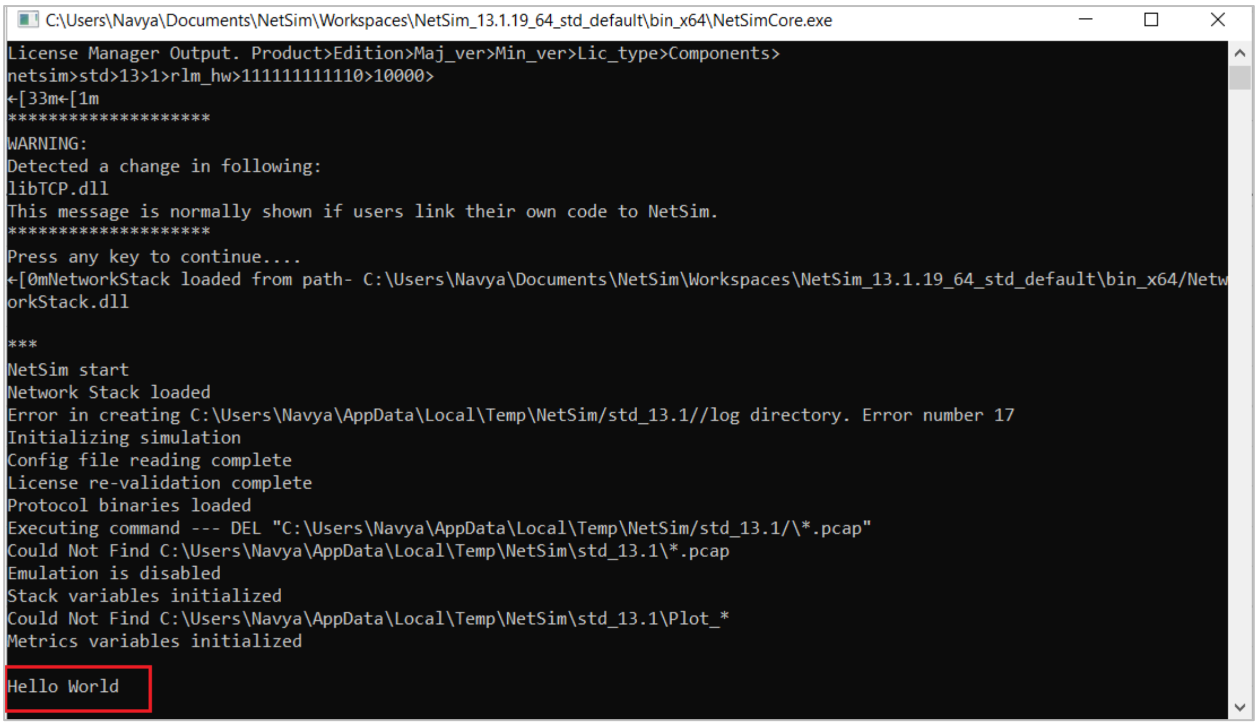
_getch();



```
/**
This function initializes the TCP parameters.
*/
_declspec (dllexport) int fn_NetSim_TCP_Init(struct stru_NetSim_Network
    *NETWORK_Formal,
                                            NetSim_EVENTDETAILS
        *pstruEventDetails_Formal,
                                            char *pszAppPath_Formal,
                                            char *pszWritePath_Formal,
                                            int nVersion_Type,
                                            void **fnPointer)
{
    fprintf(stderr, "\nHello World\n");
    _getch();

    return fn_NetSim_TCP_Init_F(NETWORK_Formal,
                                pstruEventDetails_Formal,
                                pszAppPath_Formal,
                                pszWritePath_Formal,
                                nVersion_Type,
```

Figure 9-11: Hello World Printf Statement added in TCP Project

Build DLL as explained in **Section 9.1.3** and run the simulation, you can see the following output on the console.


C:\Users\Navya\Documents\NetSim\Workspaces\NetSim_13.1.19_64_std_default\bin_x64\NetSimCore.exe

```
License Manager Output. Product>Edition>Maj_ver>Min_ver>Lic_type>Components>
netsim>std>13>1>rlm_hw>111111111110>10000>
←[33m←[1m
********************
WARNING:
Detected a change in following:
libTCP.dll
This message is normally shown if users link their own code to NetSim.
********************
Press any key to continue....
←[0mNetworkStack loaded from path- C:\Users\Navya\Documents\NetSim\Workspaces\NetSim_13.1.19_64_std_default\bin_x64/Netw
orkStack.dll

***
NetSim start
Network Stack loaded
Error in creating C:\Users\Navya\AppData\Local\Temp\NetSim/std_13.1//log directory. Error number 17
Initializing simulation
Config file reading complete
License re-validation complete
Protocol binaries loaded
Executing command --- DEL "C:\Users\Navya\AppData\Local\Temp\NetSim/std_13.1/\*.pcap"
Could Not Find C:\Users\Navya\AppData\Local\Temp\NetSim\std_13.1\*.pcap
Emulation is disabled
Stack variables initialized
Could Not Find C:\Users\Navya\AppData\Local\Temp\NetSim\std_13.1\Plot_*
Metrics variables initialized

Hello World
```

Figure 9-12: Hello World Statement written to console

Press enter then simulation will continue.

9.2.2 Introducing Node Failure in MANET

Objective: Node failure using MANET-DSR using Device Id.

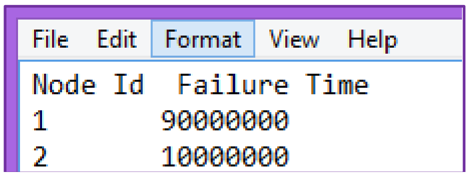
Implementation: Identify the Device ID of the particular node to be failed.

Step 1: Create a file with the name NodeFailure.txt inside the bin folder of NetSim's current workspace path

<C:\Users\PC\Documents\NetSim\Workspaces\<Your default workspace>\bin_x64> for 64-bit.

The file will contain two columns: one being the Node ID of the device to be failed and other being the failure time (in microseconds).

For example, to fail Node Id 2 from10th sec onwards and fail Node Id 1 from 90th sec onwards, the NodeFailure.txt file will be as follows Figure 9-14.



```
Node Id   Failure Time
1         90000000
2         10000000
```

Figure 9-13: NodeFailure.txt file

Step 2: Go to DSR.c in DSR protocol.

Step 3: The function fn_NetSim_DSR_Init() will execute before the protocol execution starts. So, in this function, we will read the NodeFailure.txt and save information regarding which nodes will fail at which time. Add the following code inside the specified function.

```
int i;
FILE *fp1;
char *pszFilepath;
char pszConfigInput[1000];
pszFilepath = fnpAllocateMemory(36,sizeof(char)*50);
strcpy(pszFilepath,pszAppPath);
strcat(pszFilepath,"/NodeFailure.txt");
fp1 = fopen(pszFilepath,"r");
i=0;
if(fp1)
{
        while(fgets(pszConfigInput,500,fp1)!= NULL)
        {
                sscanf(pszConfigInput,"%d    %d",&NodeArray[i],&TimeArray[i]);
                i+=1;
        }
fclose(fp1);
}
```

Step 4: The fn_NetSim_DSR_Run() is the main function to handle all the protocol functionalities. So, add the following code to the function at the start.

```
int i,nFlag=1;
if(nFlag)
{
for(i=0;i<100;i++)
        if((pstruEventDetails->nDeviceId== NodeArray[i]) &&
(pstruEventDetails->dEventTime >= TimeArray[i]))
```

```
                {
                            pstruEventDetails->nInterfaceId = 0;
                            pstruEventDetails->pPacket=NULL;
                            return 0;
                }
        }
```

Step 5: Add the following code inside DSR.h header file.

```
        //Node failure model
        int NodeArray[200];
        int TimeArray[200];
```

Step 6: Build DLL as explained in **Section 9.1.3**.

Step 7: Create a scenario in MANET with DSR Protocol. where data packets should be travelling from source to destination through the mentioned node in NodeFailure.txt file. For that user can increase the pathloss exponent value and the distance among the nodes. User can utilize Packet Animation to check the node failure (i.e. no packets are forwarded by failed nodes) after the mentioned time.

9.3 Debugging your code

This section is helpful to debug the code which user has written. To write your own code please refer **Section 9.1.2**.

9.3.1 Via GUI

Debugging your code via GUI there are two methods available.

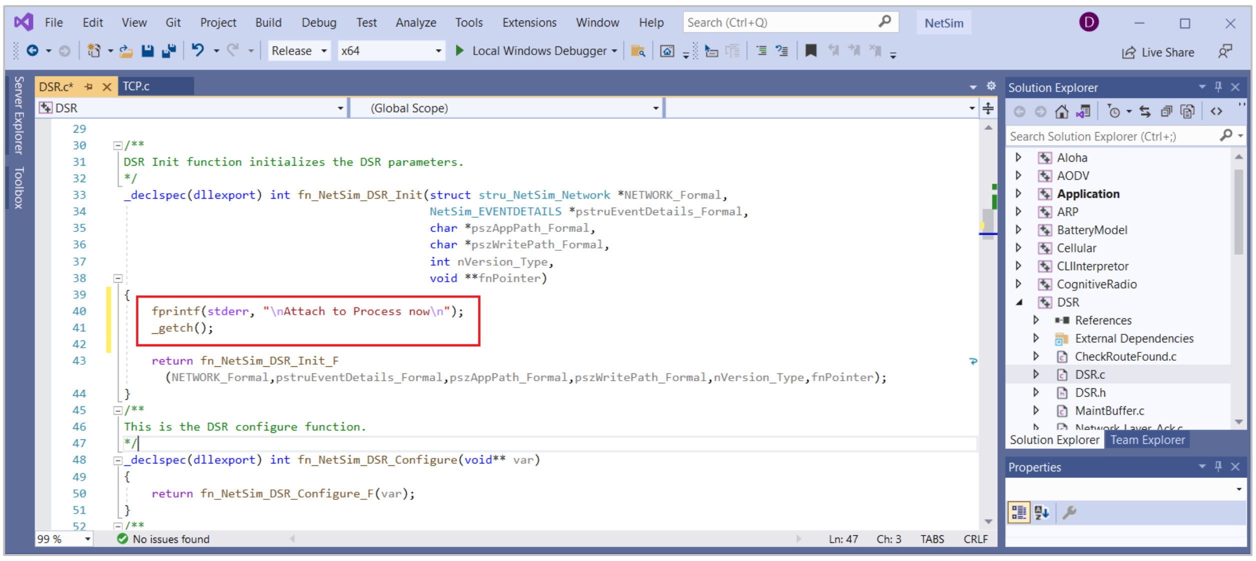
- **Using _getch()**
- **Using Environment Variables (NETSIM_BREAK)**

9.3.1.1 Using _getch()

Step 1: Perform the required modification of the protocol source code and add _getch() (used to hold the program execution until the user enters a character) statement inside init function of the modified protocol. For example, take DSR protocol and add the following lines of code in the init function as shown in the below screenshot Figure 9-15.

fprintf(stderr, "\nAttach to Process now\n");

_getch();

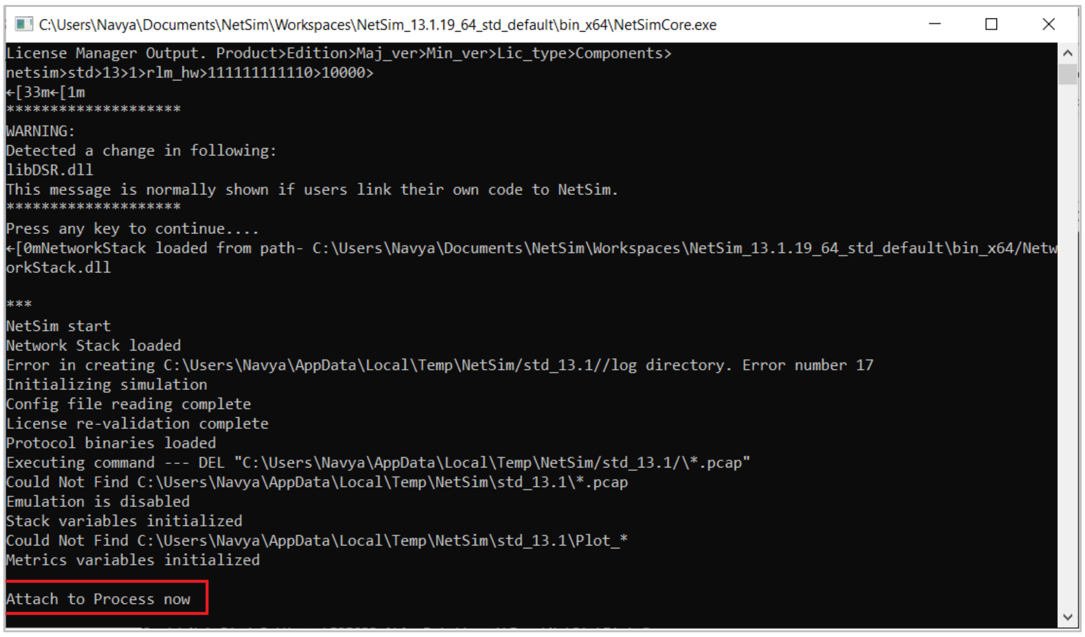


```
/**
DSR Init function initializes the DSR parameters.
*/
_declspec(dllexport) int fn_NetSim_DSR_Init(struct stru_NetSim_Network *NETWORK_Formal,
                                            NetSim_EVENTDETAILS *pstruEventDetails_Formal,
                                            char *pszAppPath_Formal,
                                            char *pszWritePath_Formal,
                                            int nVersion_Type,
                                            void **fnPointer)
{
    fprintf(stderr, "\nAttach to Process now\n");
    _getch();

    return fn_NetSim_DSR_Init_F
      (NETWORK_Formal,pstruEventDetails_Formal,pszAppPath_Formal,pszWritePath_Formal,nVersion_Type,fnPointer);
}
/**
This is the DSR configure function.
*/
_declspec(dllexport) int fn_NetSim_DSR_Configure(void** var)
{
    return fn_NetSim_DSR_Configure_F(var);
}
/**
```

Figure 9-14: Added Two line of Code in DSR protocol

Step 2: Build the DSR protocol as explained in **Section 9.1.3**. Do not close Visual Studio.

Step 3: In NetSim, create a network scenario where the protocol is being used and start the simulation. In the console window user would get a warning message shown in the below screenshot Figure 9-16 and the simulation will pause for user input (because of _getch() added in the init function)



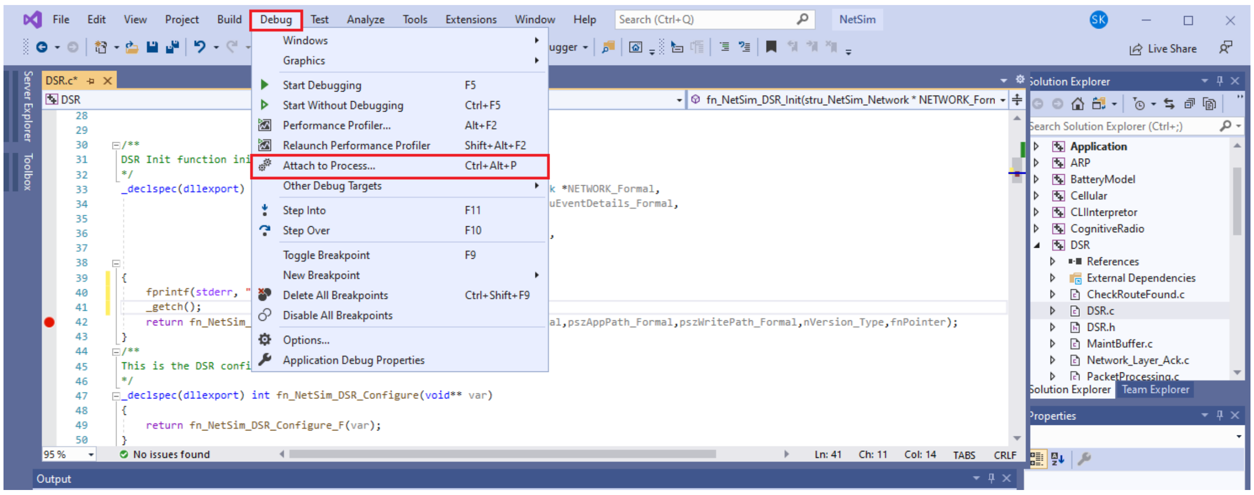
C:\Users\Navya\Documents\NetSim\Workspaces\NetSim_13.1.19_64_std_default\bin_x64\NetSimCore.exe

```
License Manager Output. Product>Edition>Maj_ver>Min_ver>Lic_type>Components>
netsim>std>13>1>rlm_hw>111111111110>10000>
←[33m←[1m
********************
WARNING:
Detected a change in following:
libDSR.dll
This message is normally shown if users link their own code to NetSim.
********************
Press any key to continue....
←[0mNetworkStack loaded from path- C:\Users\Navya\Documents\NetSim\Workspaces\NetSim_13.1.19_64_std_default\bin_x64/Netw
orkStack.dll

***
NetSim start
Network Stack loaded
Error in creating C:\Users\Navya\AppData\Local\Temp\NetSim/std_13.1//log directory. Error number 17
Initializing simulation
Config file reading complete
License re-validation complete
Protocol binaries loaded
Executing command --- DEL "C:\Users\Navya\AppData\Local\Temp\NetSim/std_13.1/\*.pcap"
Could Not Find C:\Users\Navya\AppData\Local\Temp\NetSim\std_13.1\*.pcap
Emulation is disabled
Stack variables initialized
Could Not Find C:\Users\Navya\AppData\Local\Temp\NetSim\std_13.1\Plot_*
Metrics variables initialized

Attach to Process now
```

Figure 9-15: Attach to Process Statement written to console

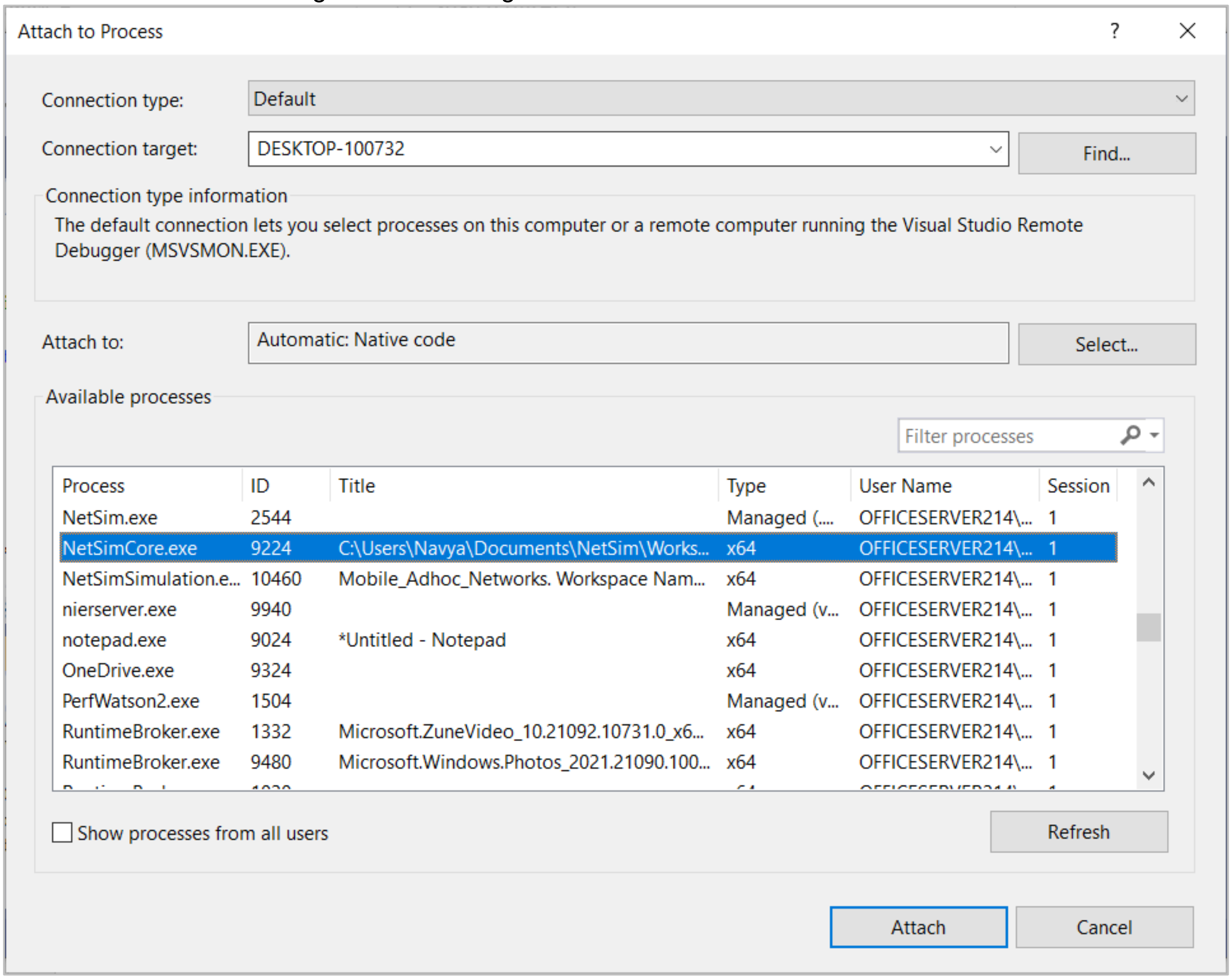
Step 4: In Visual Studio, put break point inside the source code where you want to debug.

Step 5: Go to "Debug→Attach to Process" in Visual studio as shown and attach to NetSimCore.exe.



```
/**
DSR Init function ini                                        
*/
_declspec(dllexport)                                        k *NETWORK_Formal,
                                                            uEventDetails_Formal,

                                                            ,

{
    fprintf(stderr, "
    _getch();
    return fn_NetSim_                                       al,pszAppPath_Formal,pszWritePath_Formal,nVersion_Type,fnPointer);
}
/**
This is the DSR confi
*/
_declspec(dllexport) int fn_NetSim_DSR_Configure(void** var)
{
    return fn_NetSim_DSR_Configure_F(var);
}
```

Figure 9-16: Debug > Attach to Process in Visual studio



Connection type: Default

Connection target: DESKTOP-100732

Connection type information

The default connection lets you select processes on this computer or a remote computer running the Visual Studio Remote Debugger (MSVSMON.EXE).

Attach to: Automatic: Native code

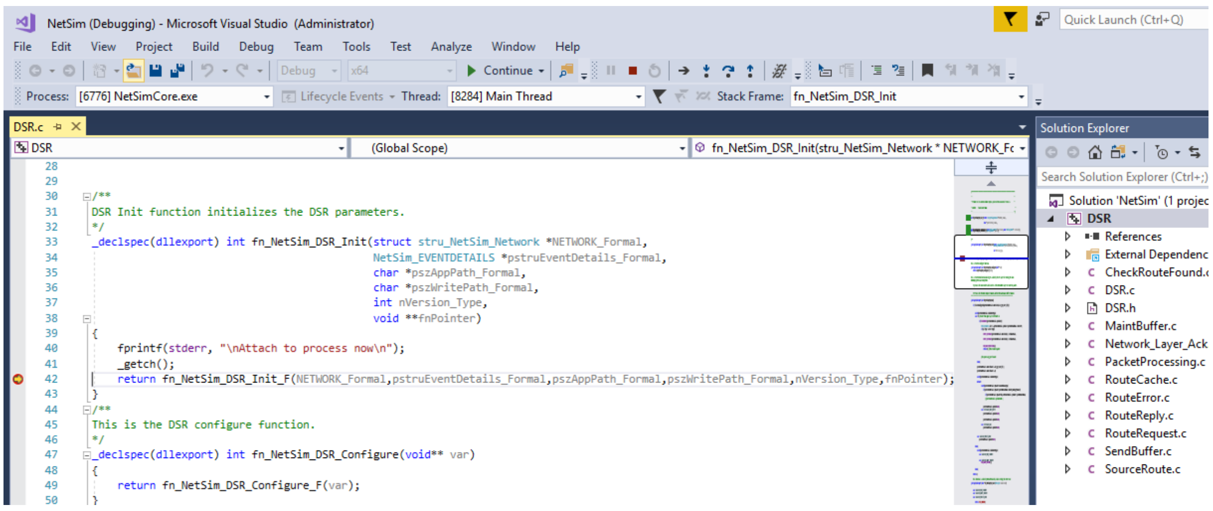
Available processes

Process	ID	Title	Type	User Name	Session
NetSim.exe	2544		Managed (....	OFFICESERVER214\...	1
NetSimCore.exe	9224	C:\Users\Navya\Documents\NetSim\Works...	x64	OFFICESERVER214\...	1
NetSimSimulation.e...	10460	Mobile_Adhoc_Networks. Workspace Nam...	x64	OFFICESERVER214\...	1
nierserver.exe	9940		Managed (v...	OFFICESERVER214\...	1
notepad.exe	9024	*Untitled - Notepad	x64	OFFICESERVER214\...	1
OneDrive.exe	9324		x64	OFFICESERVER214\...	1
PerfWatson2.exe	1504		Managed (v...	OFFICESERVER214\...	1
RuntimeBroker.exe	1332	Microsoft.ZuneVideo_10.21092.10731.0_x6...	x64	OFFICESERVER214\...	1
RuntimeBroker.exe	9480	Microsoft.Windows.Photos_2021.21090.100...	x64	OFFICESERVER214\...	1

☐ Show processes from all users

Figure 9-17: Select NetSimCore.exe in Attach to Process Window

Click on Attach. Press enter in the command window. Then control goes to the project and stops at the break point in the source code as shown below Figure 9-19. All debugging options like step over (F10), step into (F11), step out (Shift + F11), continue (F5) are available.



```
/**
DSR Init function initializes the DSR parameters.
*/
_declspec(dllexport) int fn_NetSim_DSR_Init(struct stru_NetSim_Network *NETWORK_Formal,
                                            NetSim_EVENTDETAILS *pstruEventDetails_Formal,
                                            char *pszAppPath_Formal,
                                            char *pszWritePath_Formal,
                                            int nVersion_Type,
                                            void **fnPointer)
{
    fprintf(stderr, "\nAttach to process now\n");
    _getch();
    return fn_NetSim_DSR_Init_F(NETWORK_Formal,pstruEventDetails_Formal,pszAppPath_Formal,pszWritePath_Formal,nVersion_Type,fnPointer);
}
/**
This is the DSR configure function.
*/
_declspec(dllexport) int fn_NetSim_DSR_Configure(void** var)
{
    return fn_NetSim_DSR_Configure_F(var);
}
```

Figure 9-18: Control goes to the project and stops at the break point in the source code

After execution of the function, the control goes back to NetSim and then comes back to the custom code the next time the function is called in the simulation.

To stop debugging and continue execution, press Shift+F5 (key). This then gives the control back to NetSim, for normal execution to continue.

9.3.1.2 Using Environment Variable

This section is helpful to Debug Using Environment Variable (NETSIM_BREAK). To set Environment variable follow the steps as shown.

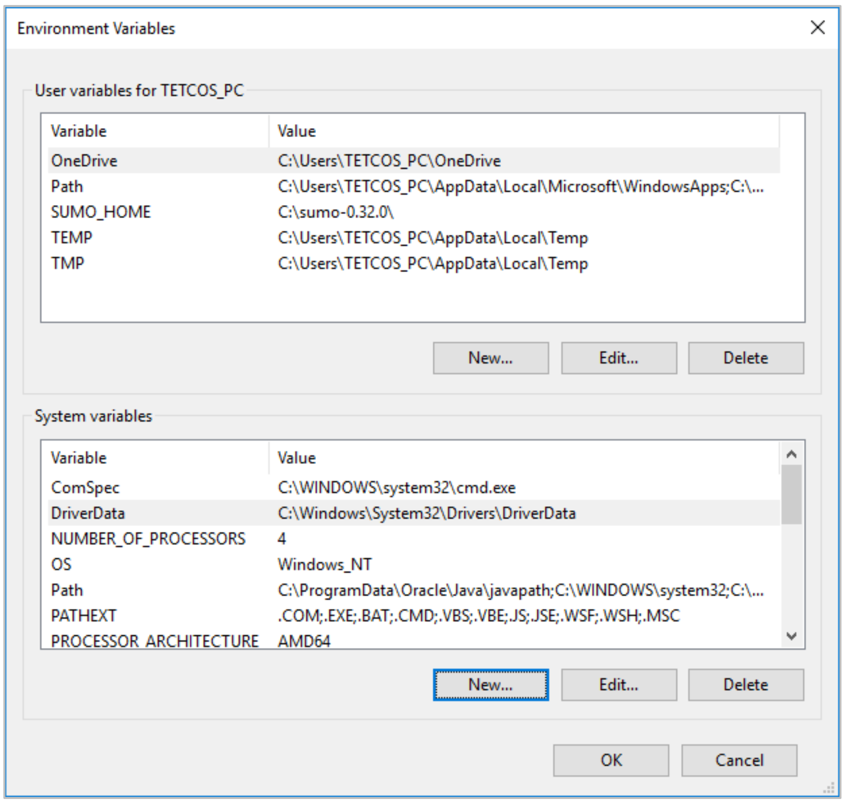
Note: *Setting NETSIM_BREAK Environment Variable will cause the simulation to slow down and it is recommended to remove this Environment Variables after debugging the simulation*

Step 1: Right click on My Computer\ This PC and select Properties.

Step 2: Go to Advanced System setting → Advanced Tab → Environment Variables option

Step 3: Click New in System variables. Type “NETSIM_BREAK” as Variable name and any positive integer as variable value (e.g., 2). Click OK. The value of the variable is the event ID at which you want NetSim Simulation to break. In this example we have set the value to 2, which means that the simulation will break at the previous event.

Note: After adding NETSIM_BREAK in environment variable, user should restart the computer otherwise it won’t affect the simulation.



Environment Variables

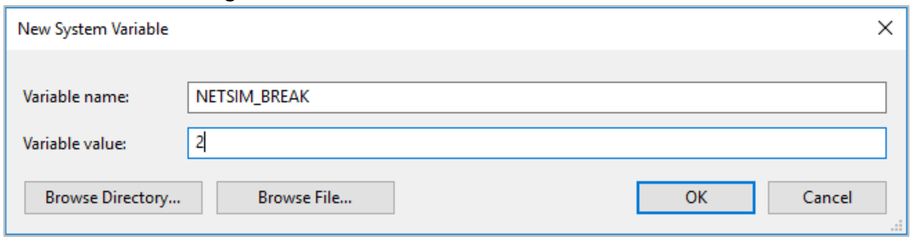
User variables for TETCOS_PC

Variable	Value
OneDrive	C:\Users\TETCOS_PC\OneDrive
Path	C:\Users\TETCOS_PC\AppData\Local\Microsoft\WindowsApps;C:\...
SUMO_HOME	C:\sumo-0.32.0\
TEMP	C:\Users\TETCOS_PC\AppData\Local\Temp
TMP	C:\Users\TETCOS_PC\AppData\Local\Temp

System variables

Variable	Value
ComSpec	C:\WINDOWS\system32\cmd.exe
DriverData	C:\Windows\System32\Drivers\DriverData
NUMBER_OF_PROCESSORS	4
OS	Windows_NT
Path	C:\ProgramData\Oracle\Java\javapath;C:\WINDOWS\system32;C:\...
PATHEXT	.COM;.EXE;.BAT;.CMD;.VBS;.VBE;.JS;.JSE;.WSF;.WSH;.MSC
PROCESSOR_ARCHITECTURE	AMD64

Figure 9-19: Environment Variable Window



New System Variable

Variable name: NETSIM_BREAK

Variable value: 2

Figure 9-20: Add Variable name and Variable Value in New in System Variable

Step 4: Open NetSim and then open the source codes. Please refer **Section 4.13** “How does a user open and modify source codes” for more information.

Step 5: Create a network scenario in NetSim (Internetworks or any other networks)

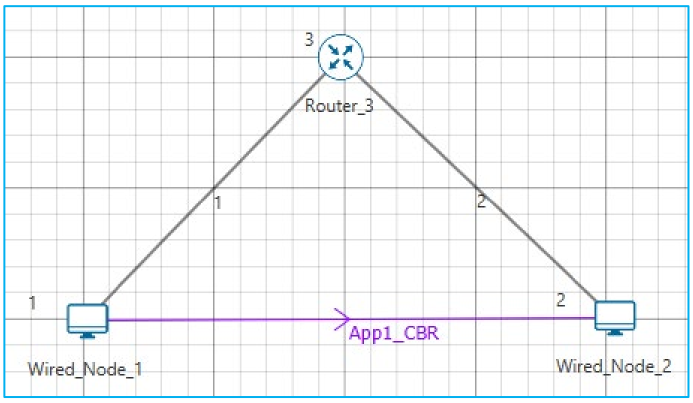
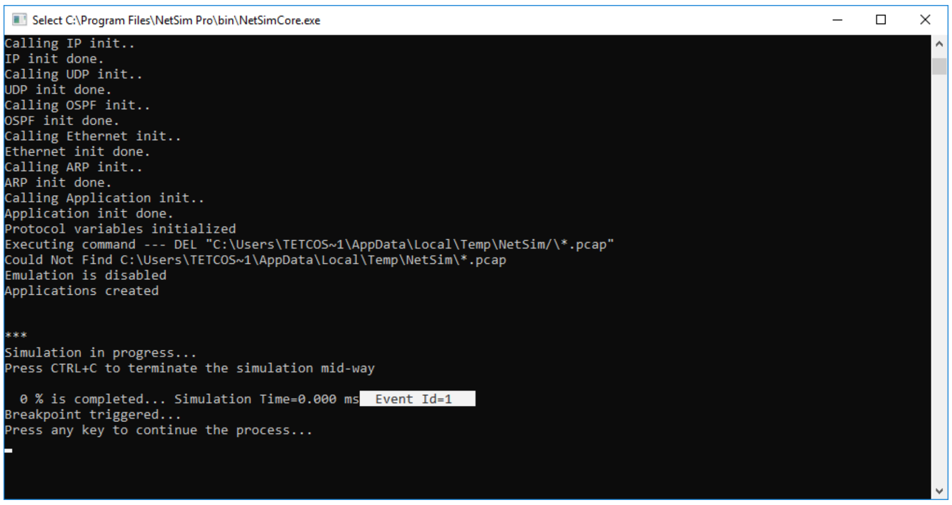


Figure 9-21: Network Topology

Step 6: In this example we are placing a break point in TCP source code and thus TCP should be select in Application properties window.

Step 7: Enable Event trace option and run the simulation.

Simulation will break at event ID 1 as we have set the environment variable to 2 as shown below.



Select C:\Program Files\NetSim Pro\bin\NetSimCore.exe

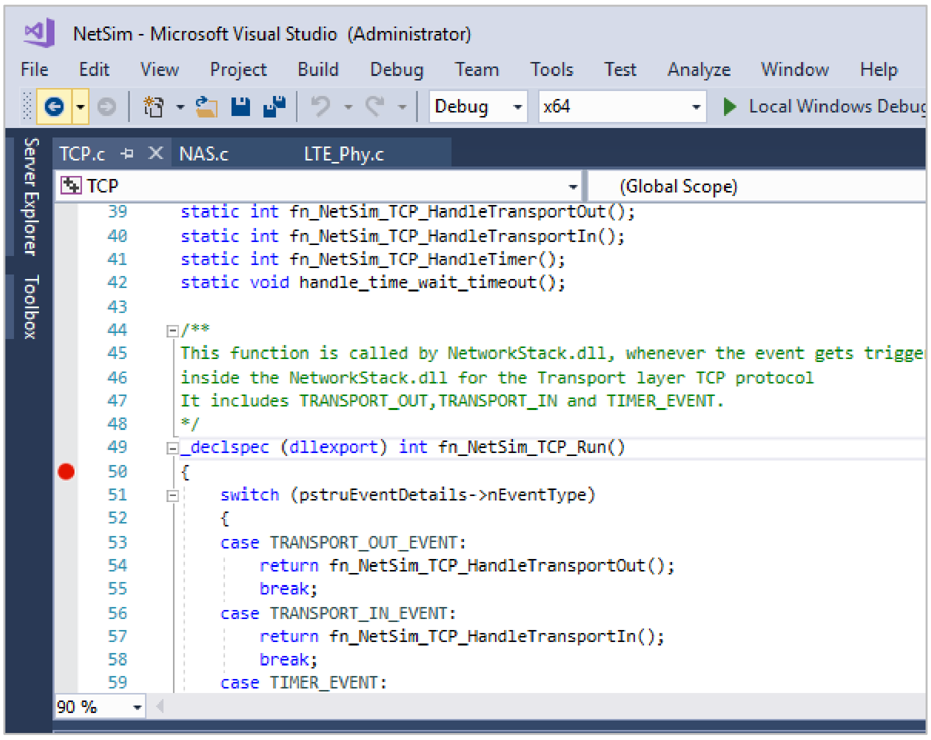
```
Calling IP init..
IP init done.
Calling UDP init..
UDP init done.
Calling OSPF init..
OSPF init done.
Calling Ethernet init..
Ethernet init done.
Calling ARP init..
ARP init done.
Calling Application init..
Application init done.
Protocol variables initialized
Executing command --- DEL "C:\Users\TETCOS~1\AppData\Local\Temp\NetSim/\*.pcap"
Could Not Find C:\Users\TETCOS~1\AppData\Local\Temp\NetSim\*.pcap
Emulation is disabled
Applications created


***
Simulation in progress...
Press CTRL+C to terminate the simulation mid-way

  0 % is completed... Simulation Time=0.000 ms  Event Id=1
Breakpoint triggered...
Press any key to continue the process...
```

Figure 9-22: Simulation will break at event ID 1

Here NetSim breakpoint has been triggered.

Step 8: Inside Solution Explorer pane in Visual Studio, double click on TCP project. Then open TCP.c file by double clicking on it. Using the drop down list of functions that are part of the current file, choose **fn_NetSim_TCP_Run().**

Step 9: In Visual Studio, Set the breakpoint in the code by clicking on the grey area on the left of the line or by right clicking on the line and selecting Breakpoint->Insert Breakpoint.


NetSim - Microsoft Visual Studio (Administrator)
File Edit View Project Build Debug Team Tools Test Analyze Window Help
Debug x64 Local Windows Debug
Server Explorer Toolbox
TCP.c NAS.c LTE_Phy.c
TCP (Global Scope)
90 %

```
39    static int fn_NetSim_TCP_HandleTransportOut();
40    static int fn_NetSim_TCP_HandleTransportIn();
41    static int fn_NetSim_TCP_HandleTimer();
42    static void handle_time_wait_timeout();
43
44  /**
45    This function is called by NetworkStack.dll, whenever the event gets trigge
46    inside the NetworkStack.dll for the Transport layer TCP protocol
47    It includes TRANSPORT_OUT,TRANSPORT_IN and TIMER_EVENT.
48    */
49  _declspec (dllexport) int fn_NetSim_TCP_Run()
50    {
51        switch (pstruEventDetails->nEventType)
52        {
53        case TRANSPORT_OUT_EVENT:
54            return fn_NetSim_TCP_HandleTransportOut();
55            break;
56        case TRANSPORT_IN_EVENT:
57            return fn_NetSim_TCP_HandleTransportIn();
58            break;
59        case TIMER_EVENT:
```

Figure 9-23: Added Break point in line number 50

Step 10: Go to "Debug→Attach to Process" in Visual studio as shown Figure 9-24 and select NetSimCore.exe from the list of processes displayed.

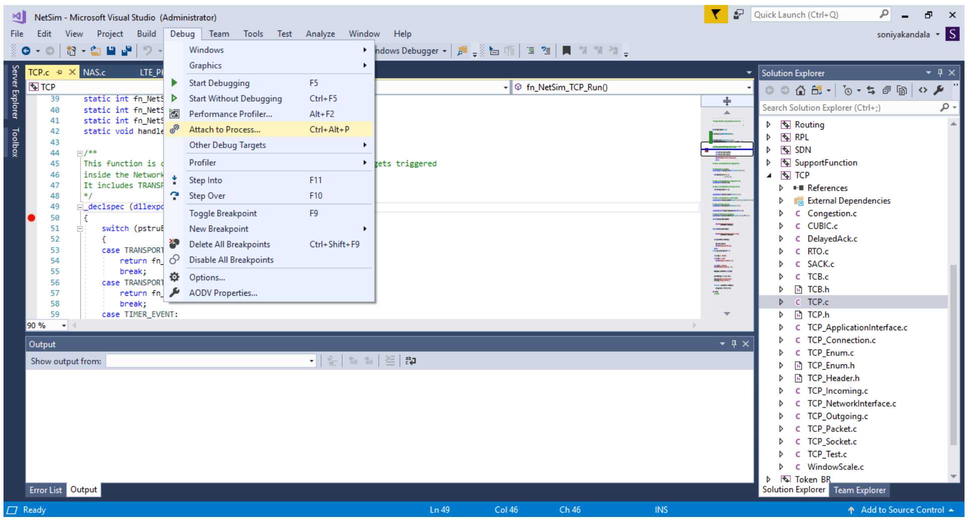
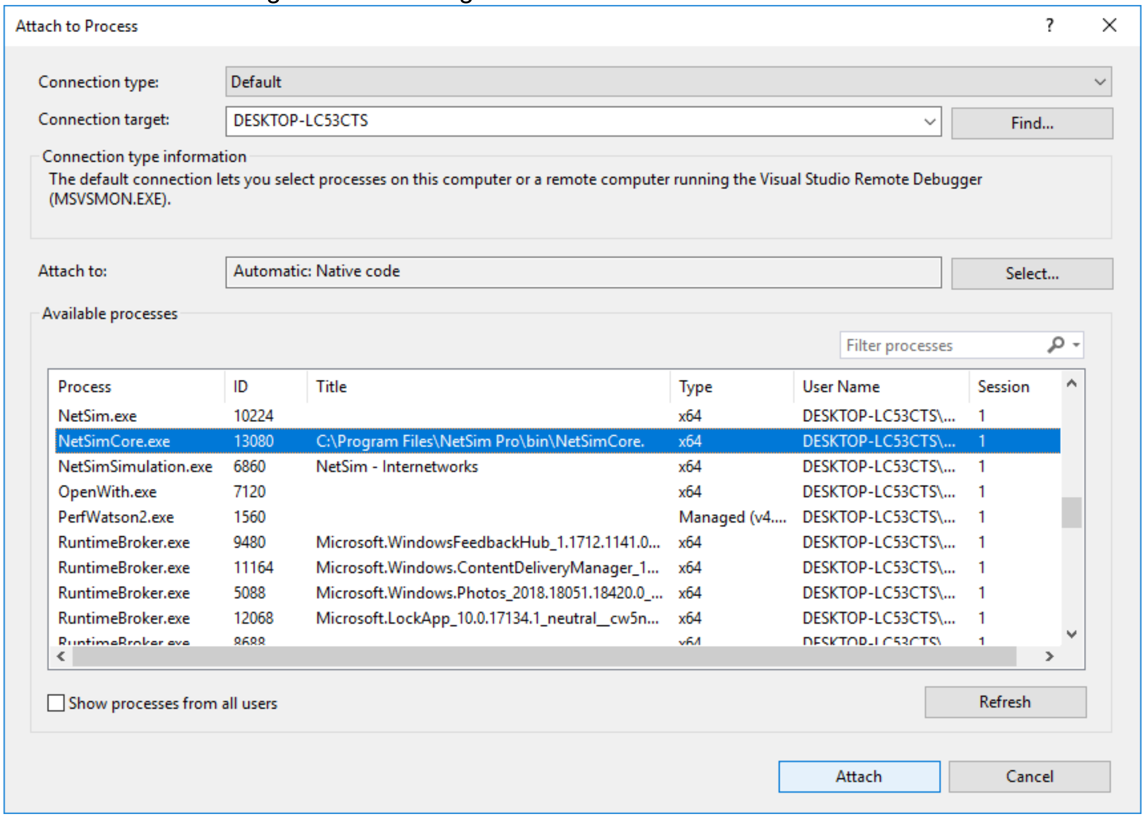

NetSim - Microsoft Visual Studio (Administrator)
File Edit View Project Build Debug Team Tools Test Analyze Window Help
Windows
Graphics
Start Debugging F5
Start Without Debugging Ctrl+F5
Performance Profiler... Alt+F2
Attach to Process... Ctrl+Alt+P
Other Debug Targets
Profiler
Step Into F11
Step Over F10
Toggle Breakpoint F9
New Breakpoint
Delete All Breakpoints Ctrl+Shift+F9
Disable All Breakpoints
Options...
AODV Properties...
Solution Explorer
Output
Ready

Figure 9-24: Debug > Attach to Process in Visual studio


Attach to Process

Connection type: Default

Connection target: DESKTOP-LC53CTS Find...

Connection type information

The default connection lets you select processes on this computer or a remote computer running the Visual Studio Remote Debugger (MSVSMON.EXE).

Attach to: Automatic: Native code Select...

Available processes

Filter processes

Process	ID	Title	Type	User Name	Session
NetSim.exe	10224		x64	DESKTOP-LC53CTS\...	1
NetSimCore.exe	13080	C:\Program Files\NetSim Pro\bin\NetSimCore.	x64	DESKTOP-LC53CTS\...	1
NetSimSimulation.exe	6860	NetSim - Internetworks	x64	DESKTOP-LC53CTS\...	1
OpenWith.exe	7120		x64	DESKTOP-LC53CTS\...	1
PerfWatson2.exe	1560		Managed (v4....	DESKTOP-LC53CTS\...	1
RuntimeBroker.exe	9480	Microsoft.WindowsFeedbackHub_1.1712.1141.0...	x64	DESKTOP-LC53CTS\...	1
RuntimeBroker.exe	11164	Microsoft.Windows.ContentDeliveryManager_1...	x64	DESKTOP-LC53CTS\...	1
RuntimeBroker.exe	5088	Microsoft.Windows.Photos_2018.18051.18420.0_...	x64	DESKTOP-LC53CTS\...	1
RuntimeBroker.exe	12068	Microsoft.LockApp_10.0.17134.1_neutral__cw5n...	x64	DESKTOP-LC53CTS\...	1
RuntimeBroker.exe	8688		x64	DESKTOP-LC53CTS\	1

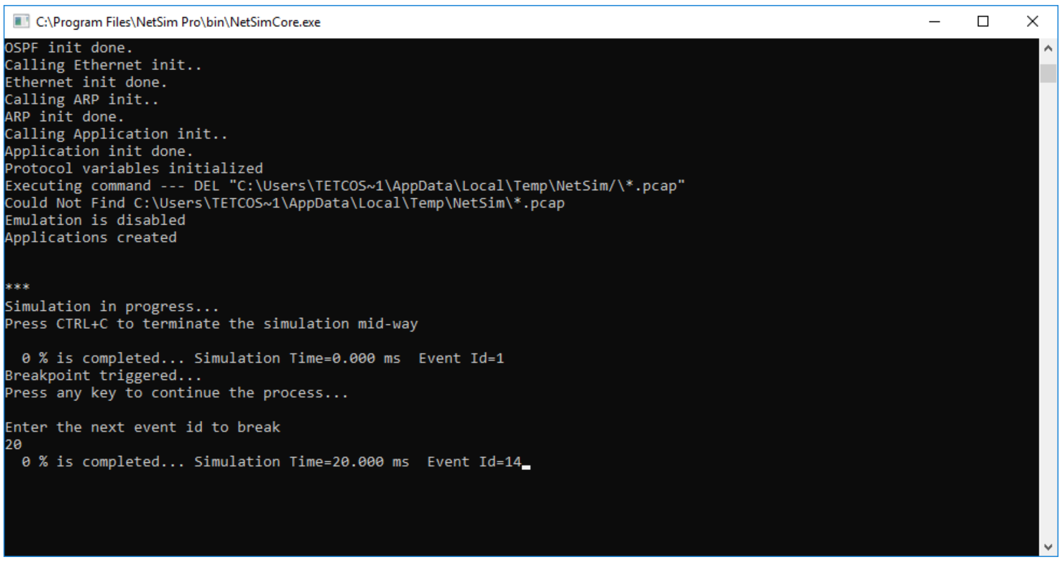
☐ Show processes from all users Refresh

Attach Cancel

Figure 9-25: Select NetSimCore.exe in Attach to Process Window

Click on Attach. Press any key in the command window to continue the process.

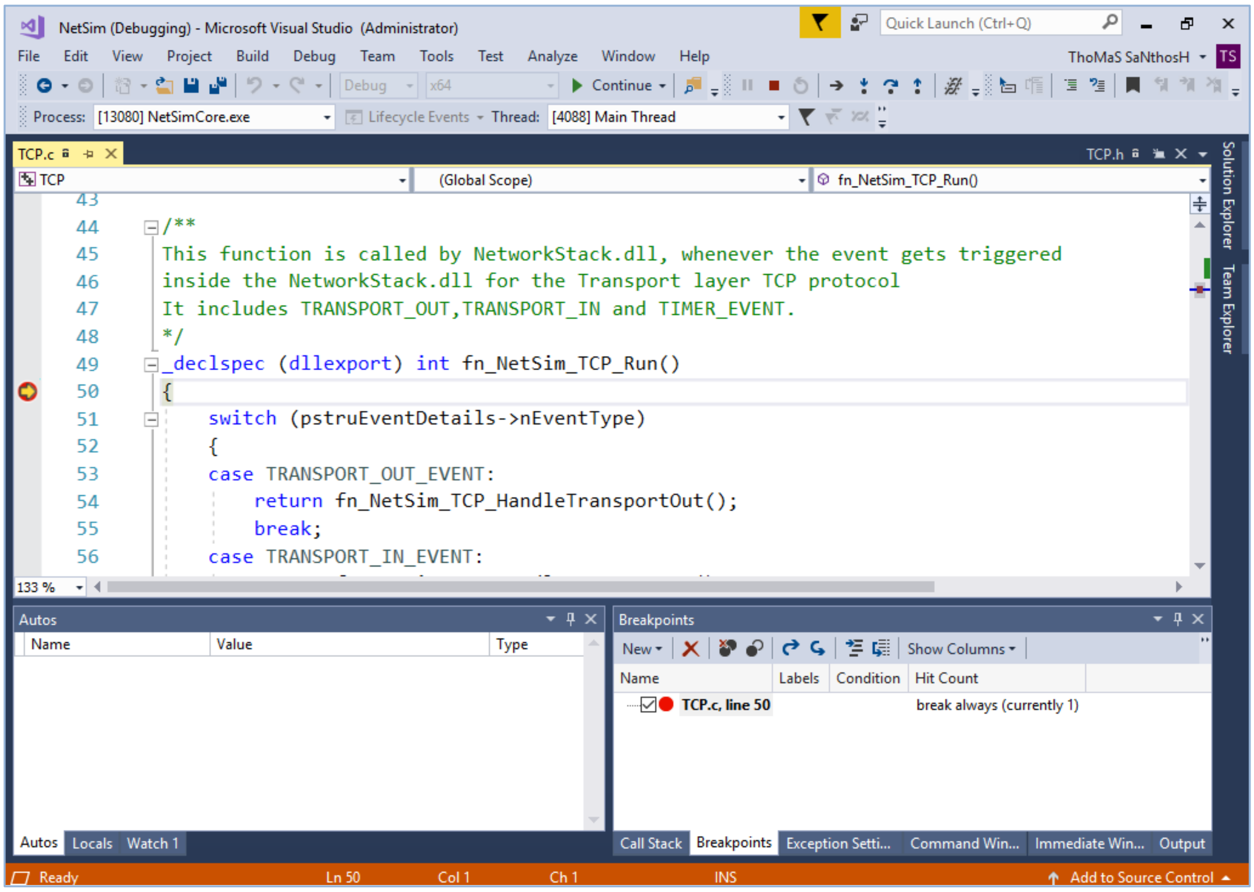
Step 11: Now we need to enter next event ID to break.



```
C:\Program Files\NetSim Pro\bin\NetSimCore.exe
OSPF init done.
Calling Ethernet init..
Ethernet init done.
Calling ARP init..
ARP init done.
Calling Application init..
Application init done.
Protocol variables initialized
Executing command --- DEL "C:\Users\TETCOS~1\AppData\Local\Temp\NetSim/\*.pcap"
Could Not Find C:\Users\TETCOS~1\AppData\Local\Temp\NetSim\*.pcap
Emulation is disabled
Applications created


***
Simulation in progress...
Press CTRL+C to terminate the simulation mid-way

  0 % is completed... Simulation Time=0.000 ms  Event Id=1
Breakpoint triggered...
Press any key to continue the process...

Enter the next event id to break
20
  0 % is completed... Simulation Time=20.000 ms  Event Id=14
```

Figure 9-26: Enter next event ID to break

Then control goes to the project and stops at the break point in the source code (NetSim will break wherever user has set the breakpoint) as shown below Figure 9-27. All debugging options like **step over (F10), step into (F11), step out (Shift + F11), continue (F5)** are available.



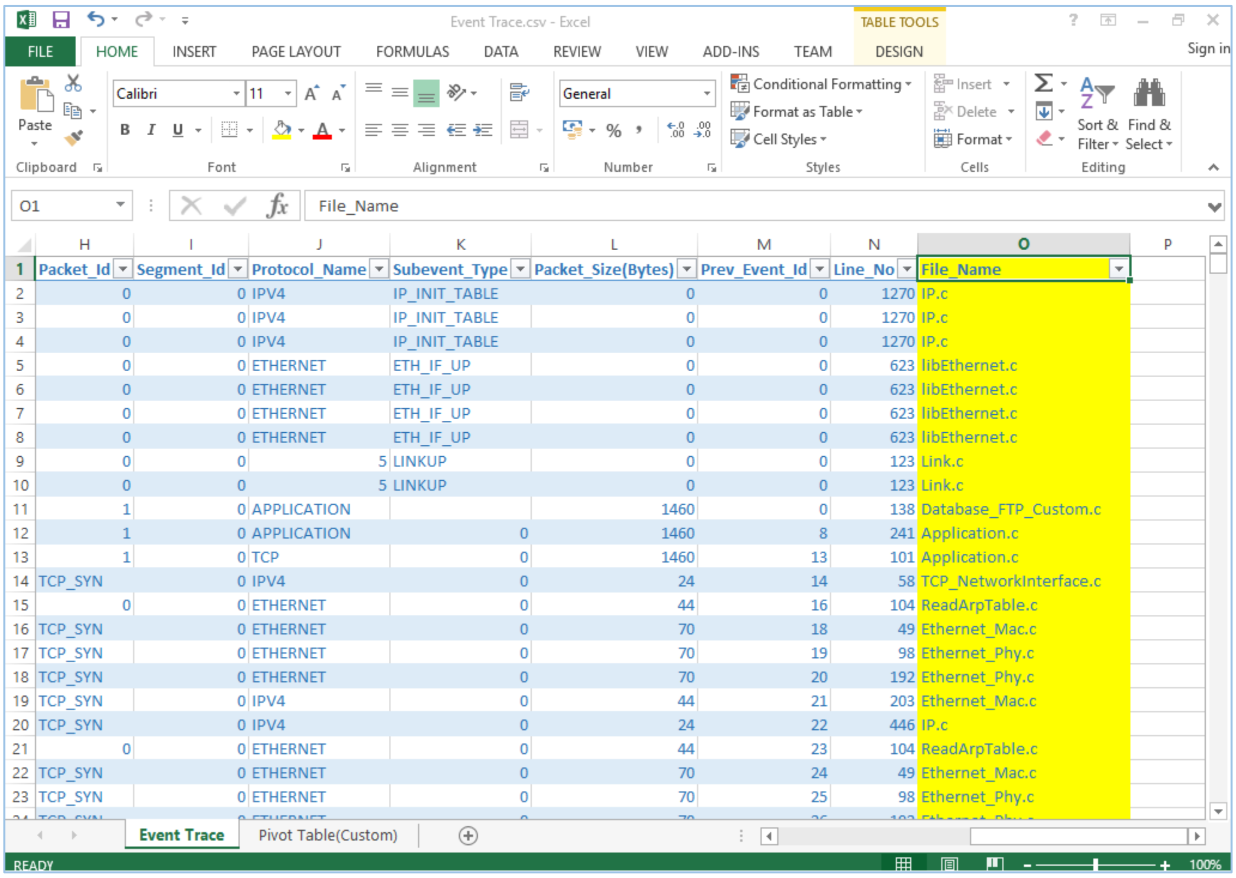
NetSim (Debugging) - Microsoft Visual Studio (Administrator)
Quick Launch (Ctrl+Q)
File Edit View Project Build Debug Team Tools Test Analyze Window Help
ThoMaS SaNthosH
Debug x64 Continue
Process: [13080] NetSimCore.exe Lifecycle Events Thread: [4088] Main Thread
TCP.c TCP.h
TCP (Global Scope) fn_NetSim_TCP_Run()
133 %
Autos Name Value Type
Autos Locals Watch 1
Breakpoints New Show Columns
Name Labels Condition Hit Count
TCP.c, line 50 break always (currently 1)
Call Stack Breakpoints Exception Setti... Command Win... Immediate Win... Output
Ready Ln 50 Col 1 Ch 1 INS Add to Source Control
Solution Explorer Team Explorer

```
/**
This function is called by NetworkStack.dll, whenever the event gets triggered
inside the NetworkStack.dll for the Transport layer TCP protocol
It includes TRANSPORT_OUT,TRANSPORT_IN and TIMER_EVENT.
*/
_declspec (dllexport) int fn_NetSim_TCP_Run()
{
    switch (pstruEventDetails->nEventType)
    {
    case TRANSPORT_OUT_EVENT:
        return fn_NetSim_TCP_HandleTransportOut();
        break;
    case TRANSPORT_IN_EVENT:
```

Figure 9-27: Control goes to the project and stops at the break point in the source code.

After execution of the function, the control goes back to NetSim and then comes back to the custom code the next time the function is called in the simulation. To stop debugging and continue execution, press **Shift+F5** (key). This then gives the control back to NetSim, for normal execution to continue.

If NETSIM_BREAK environment variable is set, NetSim **event trace file** additionally logs the file name and line number of the source code where the event was added as shown below:



	H	I	J	K	L	M	N	O
1	Packet_Id	Segment_Id	Protocol_Name	Subevent_Type	Packet_Size(Bytes)	Prev_Event_Id	Line_No	File_Name
2	0	0	IPV4	IP_INIT_TABLE	0	0	1270	IP.c
3	0	0	IPV4	IP_INIT_TABLE	0	0	1270	IP.c
4	0	0	IPV4	IP_INIT_TABLE	0	0	1270	IP.c
5	0	0	ETHERNET	ETH_IF_UP	0	0	623	libEthernet.c
6	0	0	ETHERNET	ETH_IF_UP	0	0	623	libEthernet.c
7	0	0	ETHERNET	ETH_IF_UP	0	0	623	libEthernet.c
8	0	0	ETHERNET	ETH_IF_UP	0	0	623	libEthernet.c
9	0	0	5	LINKUP	0	0	123	Link.c
10	0	0	5	LINKUP	0	0	123	Link.c
11	1	0	APPLICATION		1460	0	138	Database_FTP_Custom.c
12	1	0	APPLICATION	0	1460	8	241	Application.c
13	1	0	TCP	0	1460	13	101	Application.c
14	TCP_SYN	0	IPV4	0	24	14	58	TCP_NetworkInterface.c
15	0	0	ETHERNET	0	44	16	104	ReadArpTable.c
16	TCP_SYN	0	ETHERNET	0	70	18	49	Ethernet_Mac.c
17	TCP_SYN	0	ETHERNET	0	70	19	98	Ethernet_Phy.c
18	TCP_SYN	0	ETHERNET	0	70	20	192	Ethernet_Phy.c
19	TCP_SYN	0	IPV4	0	44	21	203	Ethernet_Mac.c
20	TCP_SYN	0	IPV4	0	24	22	446	IP.c
21	0	0	ETHERNET	0	44	23	104	ReadArpTable.c
22	TCP_SYN	0	ETHERNET	0	70	24	49	Ethernet_Mac.c
23	TCP_SYN	0	ETHERNET	0	70	25	98	Ethernet_Phy.c

Figure 9-28: NetSim event trace file additionally added two columns the file name and line number of the source code.

9.3.2 Via CLI

Modify the DSR protocol and build the code. Create a scenario on MANET then follow the below steps.

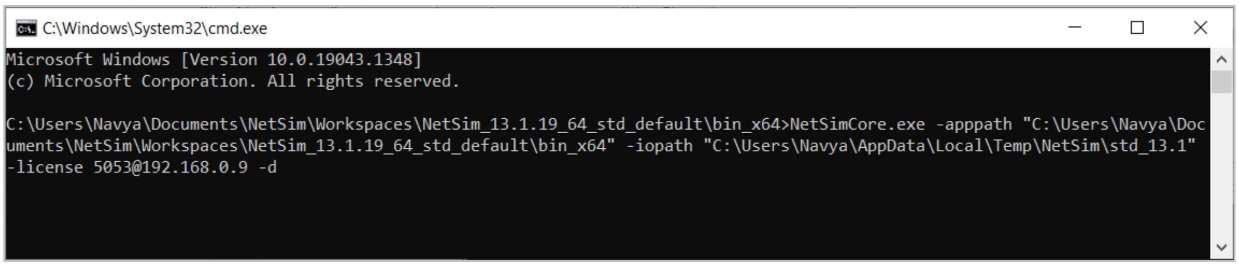
Step 1: Open the Command prompt. Press “windows+R” and type “cmd”.

Step 2: To run the NetSim via CLI copy the path where “NetSimCore.exe” is present.

>cd <apppath>

>NetSimCore.exe<space>-apppath<space><apppath><space>-iopath<space><iopath><space>-license<space>5053@<ServerIP Address><space> -d

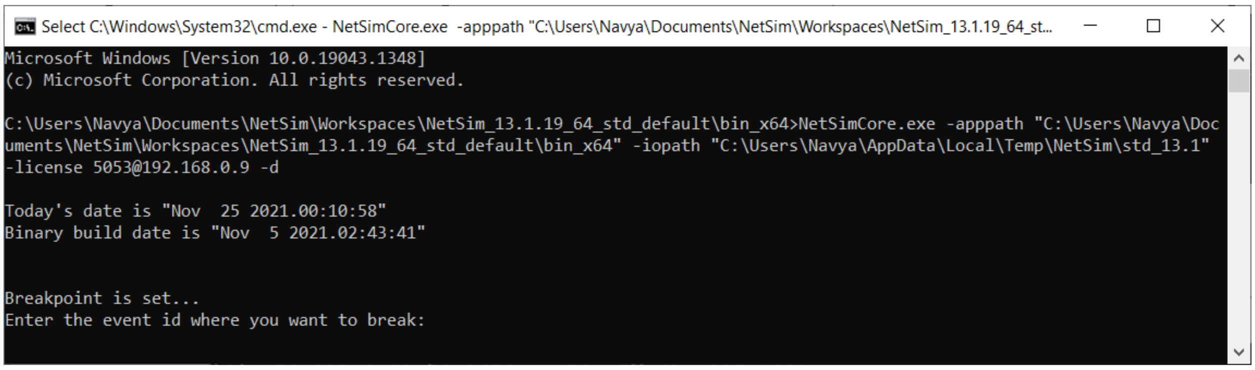
Step 3: Type the following command.



```
Microsoft Windows [Version 10.0.19043.1348]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Navya\Documents\NetSim\Workspaces\NetSim_13.1.19_64_std_default\bin_x64>NetSimCore.exe -apppath "C:\Users\Navya\Doc
uments\NetSim\Workspaces\NetSim_13.1.19_64_std_default\bin_x64" -iopath "C:\Users\Navya\AppData\Local\Temp\NetSim\std_13.1"
-license 5053@192.168.0.9 -d
```

Figure 9-29: Run the NetSim via CLI Mode use the following Command.

Press enter, now you can see the following screen.



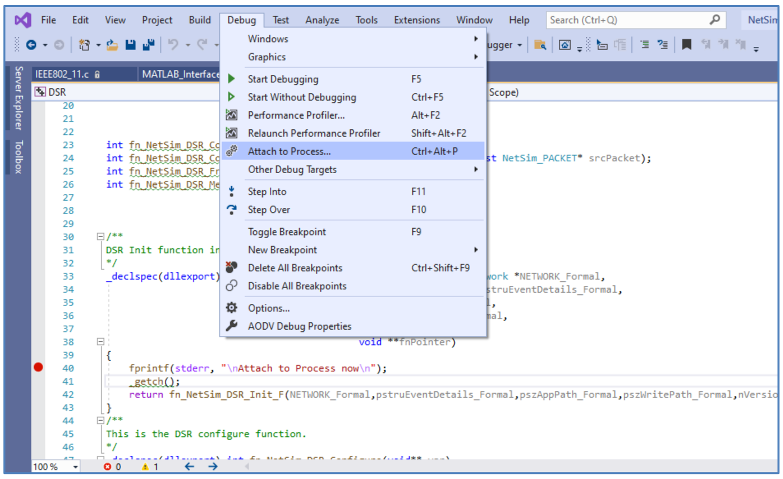
Select C:\Windows\System32\cmd.exe - NetSimCore.exe -apppath "C:\Users\Navya\Documents\NetSim\Workspaces\NetSim_13.1.19_64_st...

```
Microsoft Windows [Version 10.0.19043.1348]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Navya\Documents\NetSim\Workspaces\NetSim_13.1.19_64_std_default\bin_x64>NetSimCore.exe -apppath "C:\Users\Navya\Doc
uments\NetSim\Workspaces\NetSim_13.1.19_64_std_default\bin_x64" -iopath "C:\Users\Navya\AppData\Local\Temp\NetSim\std_13.1"
-license 5053@192.168.0.9 -d

Today's date is "Nov  25 2021.00:10:58"
Binary build date is "Nov  5 2021.02:43:41"


Breakpoint is set...
Enter the event id where you want to break:
```

Figure 9-30: Enter the Event ID

Step 4: Open the Project in Visual Studio and put break point inside the source code.

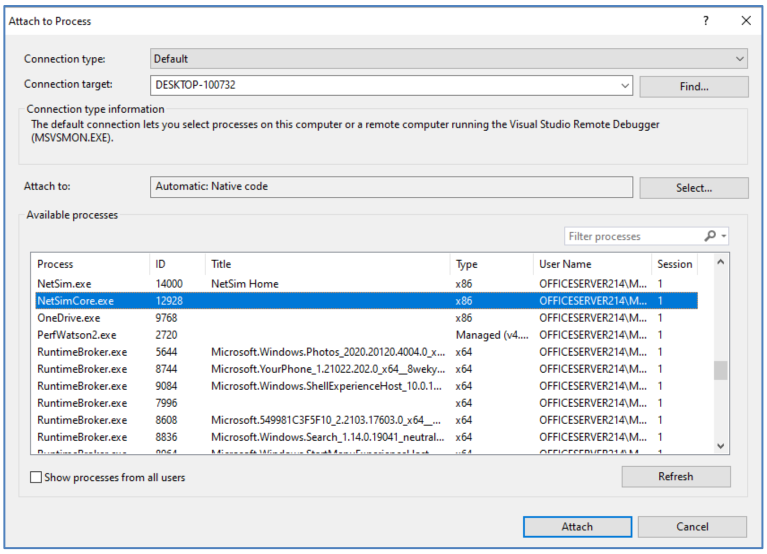
Step 5: Go to "Debug→Attach to Process".



```
    int fn_NetSim_DSR_Co                                    st NetSim_PACKET* srcPacket);
    int fn_NetSim_DSR_Co
    int fn_NetSim_DSR_Fr
    int fn_NetSim_DSR_Me


/**
DSR Init function in
*/
_declspec(dllexport)                                        work *NETWORK_Formal,
                                                            struEventDetails_Formal,
                                                            l,
                                                            nal,
                                            void **fnPointer)
{
    fprintf(stderr, "\nAttach to Process now\n");
    _getch();
    return fn_NetSim_DSR_Init_F(NETWORK_Formal,pstruEventDetails_Formal,pszAppPath_Formal,pszWritePath_Formal,nVersio
}
/**
This is the DSR configure function.
*/
```

Figure 9-31: Debug > Attach to Process

Attach to NetSimCore.exe.



Connection type: Default
Connection target: DESKTOP-100732

Connection type information
The default connection lets you select processes on this computer or a remote computer running the Visual Studio Remote Debugger (MSVSMON.EXE).

Attach to: Automatic: Native code

Available processes

Process	ID	Title	Type	User Name	Session
NetSim.exe	14000	NetSim Home	x86	OFFICESERVER214\M...	1
NetSimCore.exe	12928		x86	OFFICESERVER214\M...	1
OneDrive.exe	9768		x86	OFFICESERVER214\M...	1
PerfWatson2.exe	2720		Managed (v4....	OFFICESERVER214\M...	1
RuntimeBroker.exe	5644	Microsoft.Windows.Photos_2020.20120.4004.0_x...	x64	OFFICESERVER214\M...	1
RuntimeBroker.exe	8744	Microsoft.YourPhone_1.21022.202.0_x64__8weky...	x64	OFFICESERVER214\M...	1
RuntimeBroker.exe	9084	Microsoft.Windows.ShellExperienceHost_10.0.1...	x64	OFFICESERVER214\M...	1
RuntimeBroker.exe	7996		x64	OFFICESERVER214\M...	1
RuntimeBroker.exe	8608	Microsoft.549981C3F5F10_2.2103.17603.0_x64__...	x64	OFFICESERVER214\M...	1
RuntimeBroker.exe	8836	Microsoft.Windows.Search_1.14.0.19041_neutral...	x64	OFFICESERVER214\M...	1

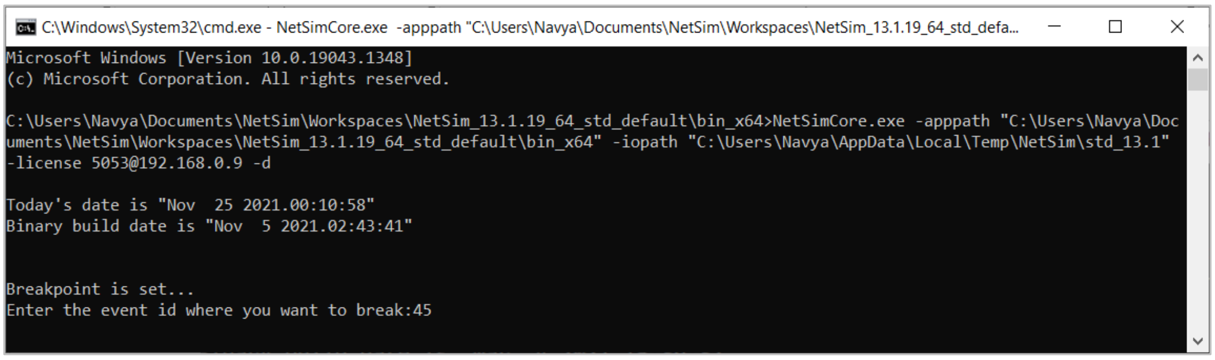
☐ Show processes from all users

Figure 9-32: Select NetSimCore.exe in Attach to Process Window

Click on Attach.

Step 6: Go to command prompt which is already opened in Step 3. Enter the Event Id.

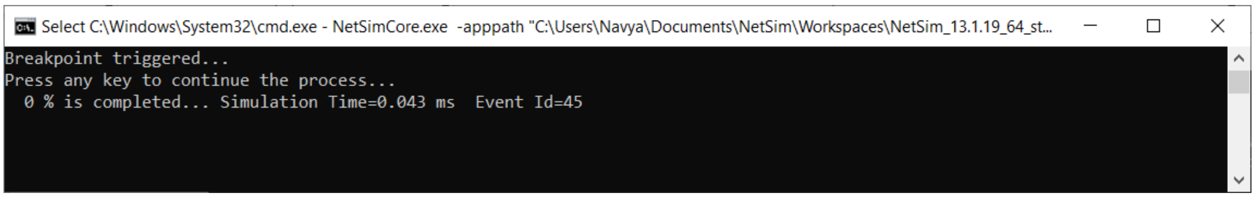
Note: *If you do not want to stop at any event you can specify 0 as event id.*



```
C:\Windows\System32\cmd.exe - NetSimCore.exe  -apppath "C:\Users\Navya\Documents\NetSim\Workspaces\NetSim_13.1.19_64_std_defa...
Microsoft Windows [Version 10.0.19043.1348]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Navya\Documents\NetSim\Workspaces\NetSim_13.1.19_64_std_default\bin_x64>NetSimCore.exe -apppath "C:\Users\Navya\Doc
uments\NetSim\Workspaces\NetSim_13.1.19_64_std_default\bin_x64" -iopath "C:\Users\Navya\AppData\Local\Temp\NetSim\std_13.1"
-license 5053@192.168.0.9 -d

Today's date is "Nov  25 2021.00:10:58"
Binary build date is "Nov  5 2021.02:43:41"


Breakpoint is set...
Enter the event id where you want to break:45
```

Figure 9-33: Enter the Event Id

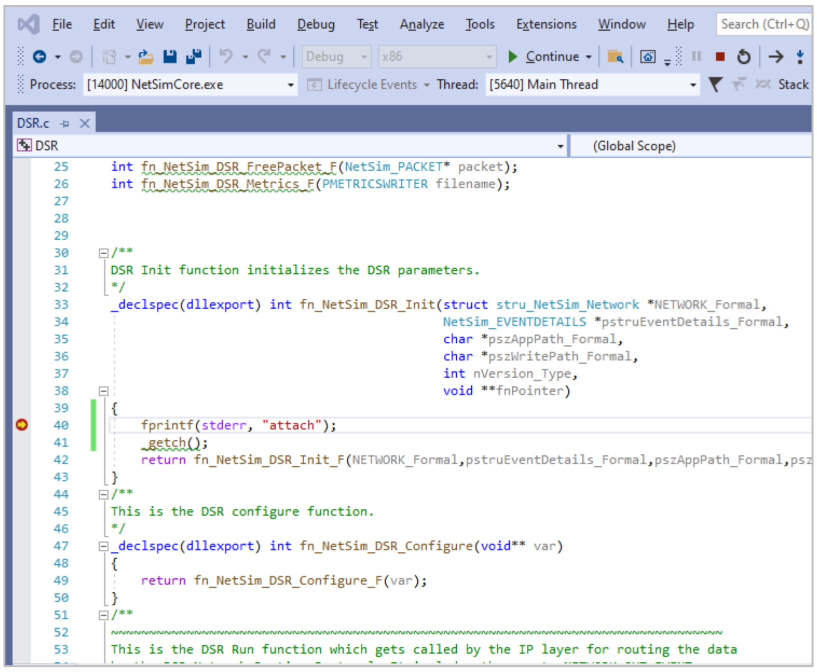
Execution will stop at the specified event.



```
Select C:\Windows\System32\cmd.exe - NetSimCore.exe  -apppath "C:\Users\Navya\Documents\NetSim\Workspaces\NetSim_13.1.19_64_st...
Breakpoint triggered...
Press any key to continue the process...
  0 % is completed... Simulation Time=0.043 ms  Event Id=45
```

Figure 9-34: Execution stops at the specified event

Press enter then control goes to the project and stops at the break point in the source code as shown below.



```
int fn_NetSim_DSR_FreePacket_F(NetSim_PACKET* packet);
int fn_NetSim_DSR_Metrics_F(PMETRICSWRITER filename);



/**
DSR Init function initializes the DSR parameters.
*/
_declspec(dllexport) int fn_NetSim_DSR_Init(struct stru_NetSim_Network *NETWORK_Formal,
                                            NetSim_EVENTDETAILS *pstruEventDetails_Formal,
                                            char *pszAppPath_Formal,
                                            char *pszWritePath_Formal,
                                            int nVersion_Type,
                                            void **fnPointer)
{
    fprintf(stderr, "attach");
    _getch();
    return fn_NetSim_DSR_Init_F(NETWORK_Formal,pstruEventDetails_Formal,pszAppPath_Formal,psz
}
/**
This is the DSR configure function.
*/
_declspec(dllexport) int fn_NetSim_DSR_Configure(void** var)
{
    return fn_NetSim_DSR_Configure_F(var);
}
/**
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
This is the DSR Run function which gets called by the IP layer for routing the data
```

Figure 9-35: Control goes to the project and stops at the break point in the source code

All debugging options like step over (F10), step into (F11), step out (Shift + F11), continue (F5) are available.

After execution of the function, the control goes back to NetSim and then comes back to the custom code the next time the function is called in the simulation.

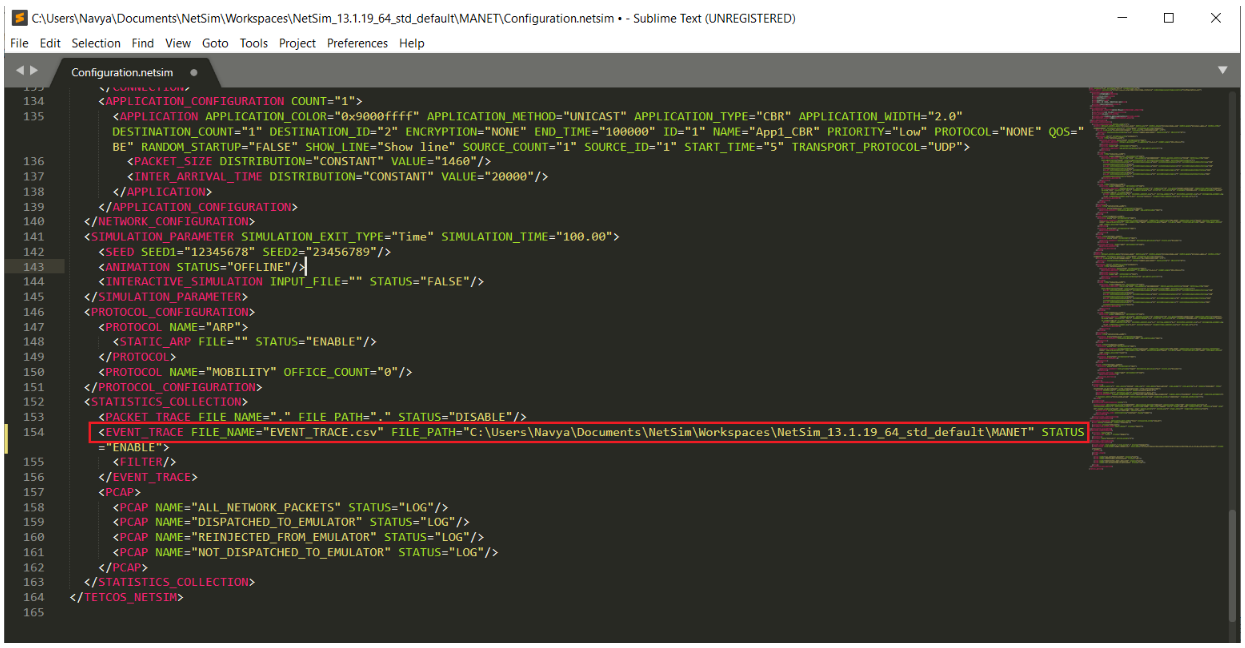
To stop debugging press Shift+F5. This then gives the control back to NetSim, for normal execution to continue.

9.3.3 Co-relating with Event Trace

To debug your own (custom) code, it is often helpful to know which section of the code (file name & line number) generated the event under study. There are 2 ways to enable this feature.

Procedure 1

Step 1: Open configuration.netsim file and provide the file name, path and set status as Enable.

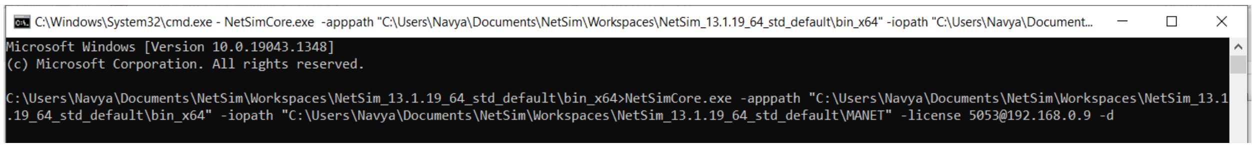

C:\Users\Navya\Documents\NetSim\Workspaces\NetSim_13.1.19_64_std_default\MANET\Configuration.netsim • - Sublime Text (UNREGISTERED)
File Edit Selection Find View Goto Tools Project Preferences Help
Configuration.netsim

```
    <APPLICATION_CONFIGURATION COUNT="1">
      <APPLICATION APPLICATION_COLOR="0x9000ffff" APPLICATION_METHOD="UNICAST" APPLICATION_TYPE="CBR" APPLICATION_WIDTH="2.0"
      DESTINATION_COUNT="1" DESTINATION_ID="2" ENCRYPTION="NONE" END_TIME="100000" ID="1" NAME="App1_CBR" PRIORITY="Low" PROTOCOL="NONE" QOS="
      BE" RANDOM_STARTUP="FALSE" SHOW_LINE="Show line" SOURCE_COUNT="1" SOURCE_ID="1" START_TIME="5" TRANSPORT_PROTOCOL="UDP">
        <PACKET_SIZE DISTRIBUTION="CONSTANT" VALUE="1460"/>
        <INTER_ARRIVAL_TIME DISTRIBUTION="CONSTANT" VALUE="20000"/>
      </APPLICATION>
    </APPLICATION_CONFIGURATION>
  </NETWORK_CONFIGURATION>
  <SIMULATION_PARAMETER SIMULATION_EXIT_TYPE="Time" SIMULATION_TIME="100.00">
    <SEED SEED1="12345678" SEED2="23456789"/>
    <ANIMATION STATUS="OFFLINE"/>
    <INTERACTIVE_SIMULATION INPUT_FILE="" STATUS="FALSE"/>
  </SIMULATION_PARAMETER>
  <PROTOCOL_CONFIGURATION>
    <PROTOCOL NAME="ARP">
      <STATIC_ARP FILE="" STATUS="ENABLE"/>
    </PROTOCOL>
    <PROTOCOL NAME="MOBILITY" OFFICE_COUNT="0"/>
  </PROTOCOL_CONFIGURATION>
  <STATISTICS_COLLECTION>
    <PACKET_TRACE FILE_NAME="." FILE_PATH="." STATUS="DISABLE"/>
    <EVENT_TRACE FILE_NAME="EVENT_TRACE.csv" FILE_PATH="C:\Users\Navya\Documents\NetSim\Workspaces\NetSim_13.1.19_64_std_default\MANET" STATUS
    ="ENABLE">
      <FILTER/>
    </EVENT_TRACE>
    <PCAP>
      <PCAP NAME="ALL_NETWORK_PACKETS" STATUS="LOG"/>
      <PCAP NAME="DISPATCHED_TO_EMULATOR" STATUS="LOG"/>
      <PCAP NAME="REINJECTED_FROM_EMULATOR" STATUS="LOG"/>
      <PCAP NAME="NOT_DISPATCHED_TO_EMULATOR" STATUS="LOG"/>
    </PCAP>
  </STATISTICS_COLLECTION>
</TETCOS_NETSIM>
```

Figure 9-36: Enable Event Trace by editing Configuration.netsim and provide the file name, path and set status as Enable

Step 2: Run the NetSim via CLI in debug mode (Refer NetSim Help in **Section 7**→Running Simulation via CLI) with –d as the fourth parameters.

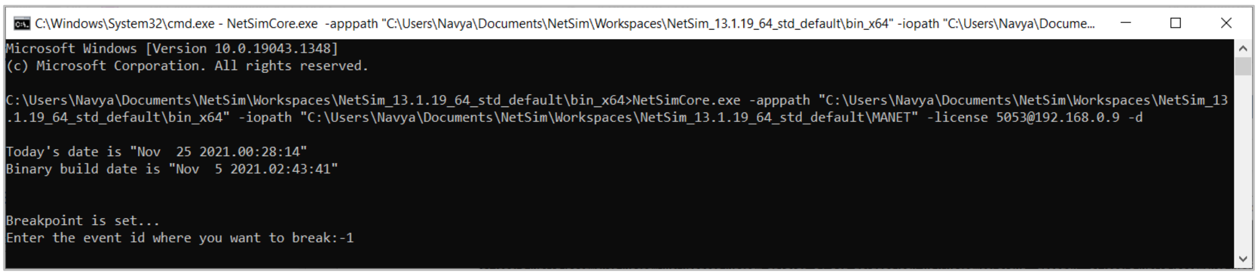
Press enter.


C:\Windows\System32\cmd.exe - NetSimCore.exe -apppath "C:\Users\Navya\Documents\NetSim\Workspaces\NetSim_13.1.19_64_std_default\bin_x64" -iopath "C:\Users\Navya\Document...

```
Microsoft Windows [Version 10.0.19043.1348]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Navya\Documents\NetSim\Workspaces\NetSim_13.1.19_64_std_default\bin_x64>NetSimCore.exe -apppath "C:\Users\Navya\Documents\NetSim\Workspaces\NetSim_13.1
.19_64_std_default\bin_x64" -iopath "C:\Users\Navya\Documents\NetSim\Workspaces\NetSim_13.1.19_64_std_default\MANET" -license 5053@192.168.0.9 -d
```

Figure 9-37: Run the NetSim via CLI

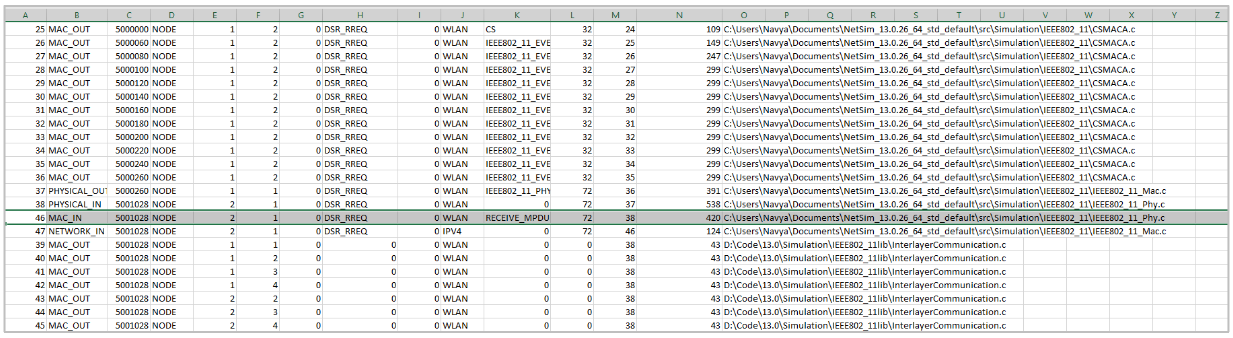
Step 3: Enter -1 as the event ID.


C:\Windows\System32\cmd.exe - NetSimCore.exe -apppath "C:\Users\Navya\Documents\NetSim\Workspaces\NetSim_13.1.19_64_std_default\bin_x64" -iopath "C:\Users\Navya\Docume...

```
Microsoft Windows [Version 10.0.19043.1348]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Navya\Documents\NetSim\Workspaces\NetSim_13.1.19_64_std_default\bin_x64>NetSimCore.exe -apppath "C:\Users\Navya\Documents\NetSim\Workspaces\NetSim_13
.1.19_64_std_default\bin_x64" -iopath "C:\Users\Navya\Documents\NetSim\Workspaces\NetSim_13.1.19_64_std_default\MANET" -license 5053@192.168.0.9 -d

Today's date is "Nov  25 2021.00:28:14"
Binary build date is "Nov  5 2021.02:43:41"


Breakpoint is set...
Enter the event id where you want to break:-1
```

Figure 9-38: Enter -1 as the event ID

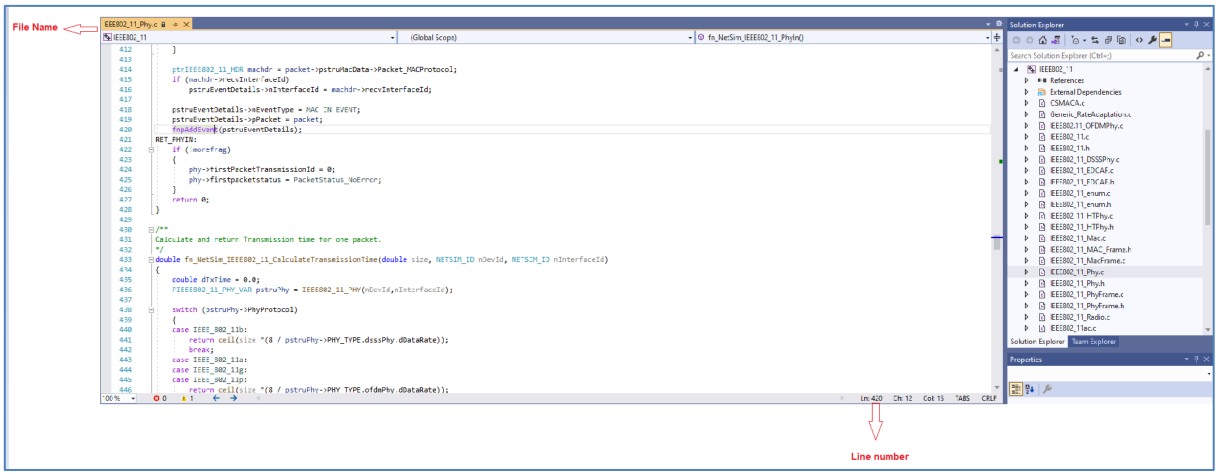
Upon running, NetSim will write the file name and line number of the source code that generated each event.



A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
25	MAC_OUT	5000000	NODE	1	2	0	DSR_RREQ	0	WLAN	CS	32	24	109	C:\Users\Navya\Documents\NetSim_13.0.26_64_std_default\src\Simulation\IEEE802_11\CSMACA.c
26	MAC_OUT	5000060	NODE	1	2	0	DSR_RREQ	0	WLAN	IEEE802_11_EVE	32	25	149	C:\Users\Navya\Documents\NetSim_13.0.26_64_std_default\src\Simulation\IEEE802_11\CSMACA.c
27	MAC_OUT	5000080	NODE	1	2	0	DSR_RREQ	0	WLAN	IEEE802_11_EVE	32	26	247	C:\Users\Navya\Documents\NetSim_13.0.26_64_std_default\src\Simulation\IEEE802_11\CSMACA.c
28	MAC_OUT	5000100	NODE	1	2	0	DSR_RREQ	0	WLAN	IEEE802_11_EVE	32	27	299	C:\Users\Navya\Documents\NetSim_13.0.26_64_std_default\src\Simulation\IEEE802_11\CSMACA.c
29	MAC_OUT	5000120	NODE	1	2	0	DSR_RREQ	0	WLAN	IEEE802_11_EVE	32	28	299	C:\Users\Navya\Documents\NetSim_13.0.26_64_std_default\src\Simulation\IEEE802_11\CSMACA.c
30	MAC_OUT	5000140	NODE	1	2	0	DSR_RREQ	0	WLAN	IEEE802_11_EVE	32	29	299	C:\Users\Navya\Documents\NetSim_13.0.26_64_std_default\src\Simulation\IEEE802_11\CSMACA.c
31	MAC_OUT	5000160	NODE	1	2	0	DSR_RREQ	0	WLAN	IEEE802_11_EVE	32	30	299	C:\Users\Navya\Documents\NetSim_13.0.26_64_std_default\src\Simulation\IEEE802_11\CSMACA.c
32	MAC_OUT	5000180	NODE	1	2	0	DSR_RREQ	0	WLAN	IEEE802_11_EVE	32	31	299	C:\Users\Navya\Documents\NetSim_13.0.26_64_std_default\src\Simulation\IEEE802_11\CSMACA.c
33	MAC_OUT	5000200	NODE	1	2	0	DSR_RREQ	0	WLAN	IEEE802_11_EVE	32	32	299	C:\Users\Navya\Documents\NetSim_13.0.26_64_std_default\src\Simulation\IEEE802_11\CSMACA.c
34	MAC_OUT	5000220	NODE	1	2	0	DSR_RREQ	0	WLAN	IEEE802_11_EVE	32	33	299	C:\Users\Navya\Documents\NetSim_13.0.26_64_std_default\src\Simulation\IEEE802_11\CSMACA.c
35	MAC_OUT	5000240	NODE	1	2	0	DSR_RREQ	0	WLAN	IEEE802_11_EVE	32	34	299	C:\Users\Navya\Documents\NetSim_13.0.26_64_std_default\src\Simulation\IEEE802_11\CSMACA.c
36	MAC_OUT	5000260	NODE	1	2	0	DSR_RREQ	0	WLAN	IEEE802_11_EVE	32	35	299	C:\Users\Navya\Documents\NetSim_13.0.26_64_std_default\src\Simulation\IEEE802_11\CSMACA.c
37	PHYSICAL_OUT	5000260	NODE	1	1	0	DSR_RREQ	0	WLAN	IEEE802_11_PHY	72	36	391	C:\Users\Navya\Documents\NetSim_13.0.26_64_std_default\src\Simulation\IEEE802_11\IEEE802_11_Mac.c
38	PHYSICAL_IN	5001028	NODE	2	1	0	DSR_RREQ	0	WLAN	0	72	37	538	C:\Users\Navya\Documents\NetSim_13.0.26_64_std_default\src\Simulation\IEEE802_11\IEEE802_11_Phy.c
46	MAC_IN	5001028	NODE	2	1	0	DSR_RREQ	0	WLAN	RECEIVE_MPDU	72	38	420	C:\Users\Navya\Documents\NetSim_13.0.26_64_std_default\src\Simulation\IEEE802_11\IEEE802_11_Phy.c
47	NETWORK_IN	5001028	NODE	2	1	0	DSR_RREQ	0	IPV4	0	72	46	124	C:\Users\Navya\Documents\NetSim_13.0.26_64_std_default\src\Simulation\IEEE802_11\IEEE802_11_Mac.c
39	MAC_OUT	5001028	NODE	1	1	0	0	0	WLAN	0	0	38	43	D:\Code\13.0\Simulation\IEEE802_11lib\InterlayerCommunication.c
40	MAC_OUT	5001028	NODE	1	2	0	0	0	WLAN	0	0	38	43	D:\Code\13.0\Simulation\IEEE802_11lib\InterlayerCommunication.c
41	MAC_OUT	5001028	NODE	1	3	0	0	0	WLAN	0	0	38	43	D:\Code\13.0\Simulation\IEEE802_11lib\InterlayerCommunication.c
42	MAC_OUT	5001028	NODE	1	4	0	0	0	WLAN	0	0	38	43	D:\Code\13.0\Simulation\IEEE802_11lib\InterlayerCommunication.c
43	MAC_OUT	5001028	NODE	2	2	0	0	0	WLAN	0	0	38	43	D:\Code\13.0\Simulation\IEEE802_11lib\InterlayerCommunication.c
44	MAC_OUT	5001028	NODE	2	3	0	0	0	WLAN	0	0	38	43	D:\Code\13.0\Simulation\IEEE802_11lib\InterlayerCommunication.c
45	MAC_OUT	5001028	NODE	2	4	0	0	0	WLAN	0	0	38	43	D:\Code\13.0\Simulation\IEEE802_11lib\InterlayerCommunication.c

Figure 9-39: NetSim writes the file name and line number of the source code in Event Trace

In the above trace file Event Id 46 is triggered inside the IEEE802_11_Phy.c file which is present in IEEE802_11 project. Since all the lib files are opaque to the end user, you cannot see the source code of the lib file. However, Event Id 56 is triggered at line number 420 of IEEE802_11_Phy.c file and you can find the location of the event by opening the IEEE802_11_Phy.c file as shown below.



```
412         }
413 
414     ptrIEEE802_11_HDR machdr = packet->pstruMacData->Packet_MACProtocol;
415     if (machdr->recvInterfaceId)
416         pstruEventDetails->nInterfaceId = machdr->recvInterfaceId;
417 
418     pstruEventDetails->nEventType = MAC_IN_EVENT;
419     pstruEventDetails->pPacket = packet;
420     fnpAddEvent(pstruEventDetails);
421 RET_PHYIN:
422     if (!morefrag)
423     {
424         phy->firstPacketTransmissionId = 0;
425         phy->firstpacketstatus = PacketStatus_NoError;
426     }
427     return 0;
428 }
429 
430 /**
431 Calculate and return Transmission time for one packet.
432 */
433 double fn_NetSim_IEEE802_11_CalculateTransmissionTime(double size, NETSIM_ID nDevId, NETSIM_ID nInterfaceId)
434 {
435     double dTxTime = 0.0;
436     PIEEE802_11_PHY_VAR pstruPhy = IEEE802_11_PHY(nDevId,nInterfaceId);
437 
438     switch (pstruPhy->PhyProtocol)
439     {
440     case IEEE_802_11b:
441         return ceil(size *(8 / pstruPhy->PHY_TYPE.dsssPhy.dDataRate));
442         break;
443     case IEEE_802_11a:
444     case IEEE_802_11g:
445     case IEEE_802_11p:
446         return ceil(size *(8 / pstruPhy->PHY_TYPE.ofdmPhy.dDataRate));
```

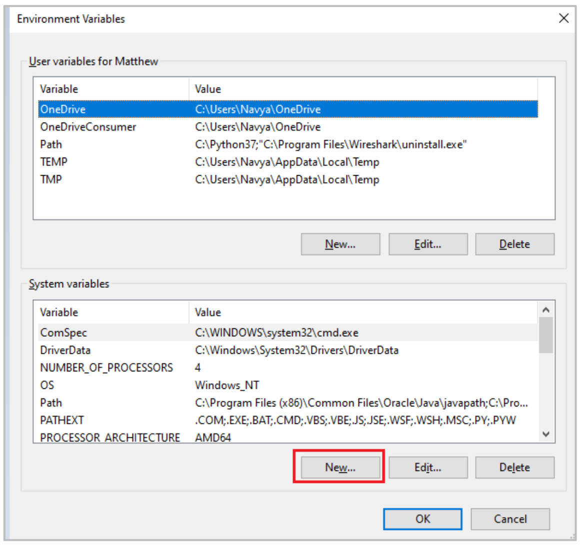
Figure 9-40: IEEE802_11_Phy.c file Code in Visual Studio

Procedure 2:

Step 1: Right click on my computer and select Properties.

Step 2: Go to Advanced System setting → Advanced Tab → Environment Variables.

Step 3: Click New. Type “NETSIM_BREAK” as Variable name and any negative integer as Variable value. Click OK.


Environment Variables

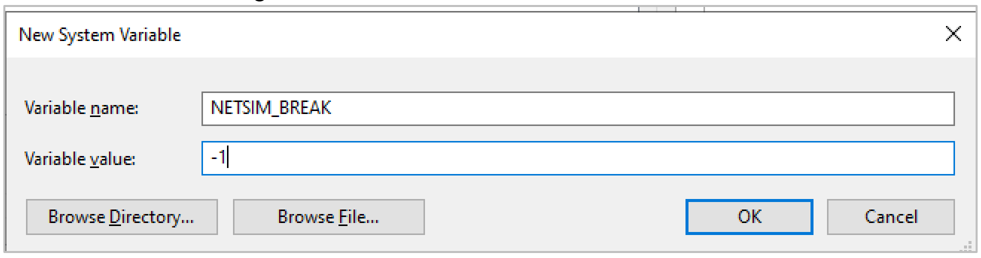
User variables for Matthew

Variable	Value
OneDrive	C:\Users\Navya\OneDrive
OneDriveConsumer	C:\Users\Navya\OneDrive
Path	C:\Python37;"C:\Program Files\Wireshark\uninstall.exe"
TEMP	C:\Users\Navya\AppData\Local\Temp
TMP	C:\Users\Navya\AppData\Local\Temp

System variables

Variable	Value
ComSpec	C:\WINDOWS\system32\cmd.exe
DriverData	C:\Windows\System32\Drivers\DriverData
NUMBER_OF_PROCESSORS	4
OS	Windows_NT
Path	C:\Program Files (x86)\Common Files\Oracle\Java\javapath;C:\Pro...
PATHEXT	.COM;.EXE;.BAT;.CMD;.VBS;.VBE;.JS;.JSE;.WSF;.WSH;.MSC;.PY;.PYW
PROCESSOR_ARCHITECTURE	AMD64

Figure 9-41: Environment Variables Window


New System Variable

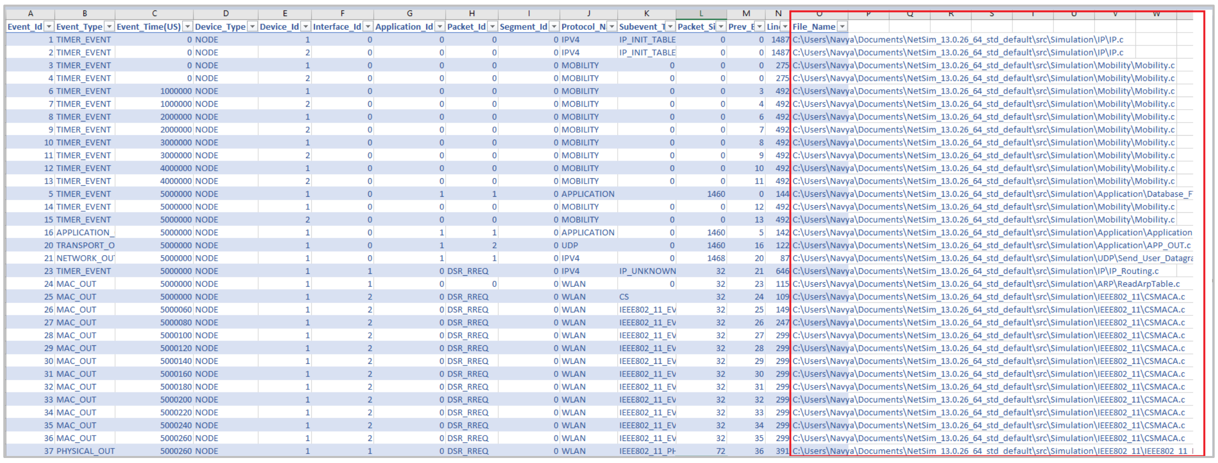
Variable name: NETSIM_BREAK

Variable value: -1

Figure 9-42: Enter Variable name and Value in New System Variable

Step 4: Restart the system.

Step 5: Now perform simulation in NetSim (Enable event trace in GUI). Upon running, NetSim will write the file name and line number of the source code that generated each event.



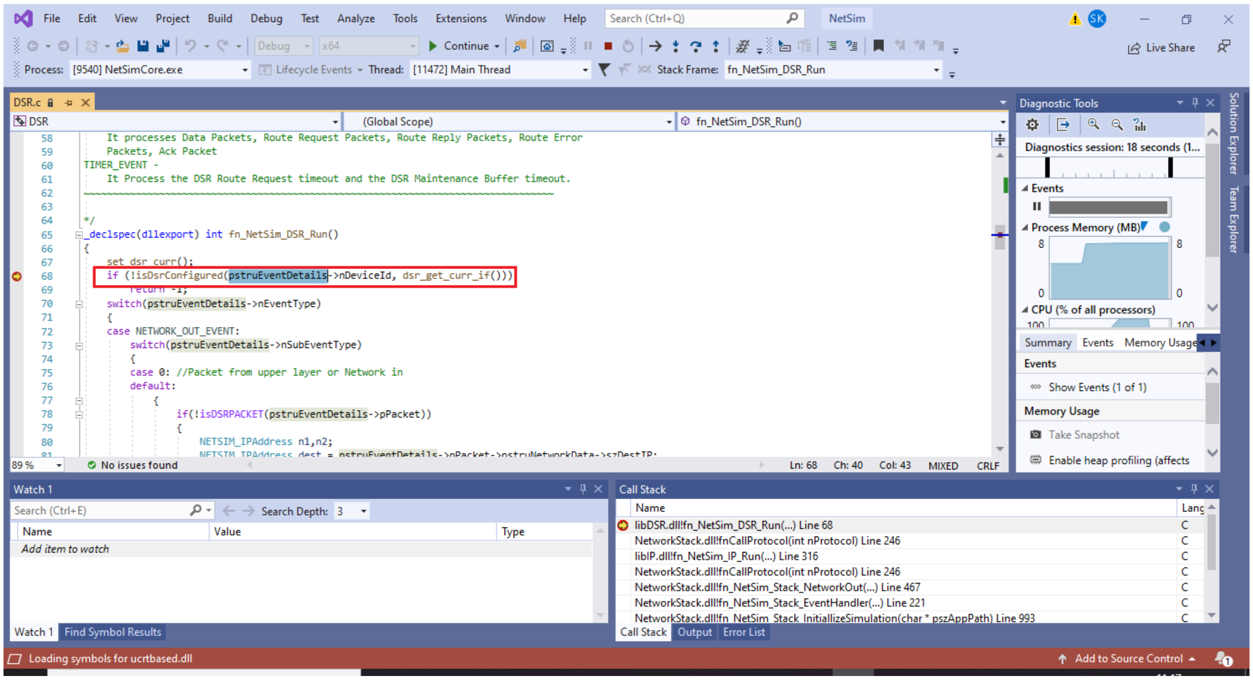
Event_Id	Event_Type	Event_Time(US)	Device_Type	Device_Id	Interface_Id	Application_Id	Packet_Id	Segment_Id	Protocol_N	Subevent_T	Packet_Si	Prev_E	Lin	File_Name
1	TIMER_EVENT	0	NODE	1	0	0	0	0	IPV4	IP_INIT_TABLE	0	0	1487	C:\Users\Navya\Documents\NetSim_13.0.26_64_std_default\src\Simulation\IP\IP.c
2	TIMER_EVENT	0	NODE	2	0	0	0	0	IPV4	IP_INIT_TABLE	0	0	1487	C:\Users\Navya\Documents\NetSim_13.0.26_64_std_default\src\Simulation\IP\IP.c
3	TIMER_EVENT	0	NODE	1	0	0	0	0	MOBILITY	0	0	0	275	C:\Users\Navya\Documents\NetSim_13.0.26_64_std_default\src\Simulation\Mobility\Mobility.c
4	TIMER_EVENT	0	NODE	2	0	0	0	0	MOBILITY	0	0	0	275	C:\Users\Navya\Documents\NetSim_13.0.26_64_std_default\src\Simulation\Mobility\Mobility.c
6	TIMER_EVENT	1000000	NODE	1	0	0	0	0	MOBILITY	0	0	3	492	C:\Users\Navya\Documents\NetSim_13.0.26_64_std_default\src\Simulation\Mobility\Mobility.c
7	TIMER_EVENT	1000000	NODE	2	0	0	0	0	MOBILITY	0	0	4	492	C:\Users\Navya\Documents\NetSim_13.0.26_64_std_default\src\Simulation\Mobility\Mobility.c
8	TIMER_EVENT	2000000	NODE	1	0	0	0	0	MOBILITY	0	0	6	492	C:\Users\Navya\Documents\NetSim_13.0.26_64_std_default\src\Simulation\Mobility\Mobility.c
9	TIMER_EVENT	2000000	NODE	2	0	0	0	0	MOBILITY	0	0	7	492	C:\Users\Navya\Documents\NetSim_13.0.26_64_std_default\src\Simulation\Mobility\Mobility.c
10	TIMER_EVENT	3000000	NODE	1	0	0	0	0	MOBILITY	0	0	8	492	C:\Users\Navya\Documents\NetSim_13.0.26_64_std_default\src\Simulation\Mobility\Mobility.c
11	TIMER_EVENT	3000000	NODE	2	0	0	0	0	MOBILITY	0	0	9	492	C:\Users\Navya\Documents\NetSim_13.0.26_64_std_default\src\Simulation\Mobility\Mobility.c
12	TIMER_EVENT	4000000	NODE	1	0	0	0	0	MOBILITY	0	0	10	492	C:\Users\Navya\Documents\NetSim_13.0.26_64_std_default\src\Simulation\Mobility\Mobility.c
13	TIMER_EVENT	4000000	NODE	2	0	0	0	0	MOBILITY	0	0	11	492	C:\Users\Navya\Documents\NetSim_13.0.26_64_std_default\src\Simulation\Mobility\Mobility.c
5	TIMER_EVENT	5000000	NODE	1	0	1	1	0	APPLICATION		1460	0	144	C:\Users\Navya\Documents\NetSim_13.0.26_64_std_default\src\Simulation\Application\Database_F
14	TIMER_EVENT	5000000	NODE	1	0	0	0	0	MOBILITY	0	0	12	492	C:\Users\Navya\Documents\NetSim_13.0.26_64_std_default\src\Simulation\Mobility\Mobility.c
15	TIMER_EVENT	5000000	NODE	2	0	0	0	0	MOBILITY	0	0	13	492	C:\Users\Navya\Documents\NetSim_13.0.26_64_std_default\src\Simulation\Mobility\Mobility.c
16	APPLICATION_	5000000	NODE	1	0	1	1	0	APPLICATION	0	1460	5	142	C:\Users\Navya\Documents\NetSim_13.0.26_64_std_default\src\Simulation\Application\Application
20	TRANSPORT_O	5000000	NODE	1	0	1	2	0	UDP	0	1460	16	122	C:\Users\Navya\Documents\NetSim_13.0.26_64_std_default\src\Simulation\Application\APP_OUT.c
21	NETWORK_OU	5000000	NODE	1	0	1	1	0	IPV4	0	1468	20	87	C:\Users\Navya\Documents\NetSim_13.0.26_64_std_default\src\Simulation\UDP\Send_User_Datagra
23	TIMER_EVENT	5000000	NODE	1	1	0	DSR_RREQ	0	IPV4	IP_UNKNOWN	32	21	646	C:\Users\Navya\Documents\NetSim_13.0.26_64_std_default\src\Simulation\IP\IP_Routing.c
24	MAC_OUT	5000000	NODE	1	1	0	0	0	WLAN	0	32	23	115	C:\Users\Navya\Documents\NetSim_13.0.26_64_std_default\src\Simulation\ARP\ReadArpTable.c
25	MAC_OUT	5000000	NODE	1	2	0	DSR_RREQ	0	WLAN	CS	32	24	109	C:\Users\Navya\Documents\NetSim_13.0.26_64_std_default\src\Simulation\IEEE802_11\CSMACA.c
26	MAC_OUT	5000060	NODE	1	2	0	DSR_RREQ	0	WLAN	IEEE802_11_EV	32	25	149	C:\Users\Navya\Documents\NetSim_13.0.26_64_std_default\src\Simulation\IEEE802_11\CSMACA.c
27	MAC_OUT	5000080	NODE	1	2	0	DSR_RREQ	0	WLAN	IEEE802_11_EV	32	26	247	C:\Users\Navya\Documents\NetSim_13.0.26_64_std_default\src\Simulation\IEEE802_11\CSMACA.c
28	MAC_OUT	5000100	NODE	1	2	0	DSR_RREQ	0	WLAN	IEEE802_11_EV	32	27	299	C:\Users\Navya\Documents\NetSim_13.0.26_64_std_default\src\Simulation\IEEE802_11\CSMACA.c
29	MAC_OUT	5000120	NODE	1	2	0	DSR_RREQ	0	WLAN	IEEE802_11_EV	32	28	299	C:\Users\Navya\Documents\NetSim_13.0.26_64_std_default\src\Simulation\IEEE802_11\CSMACA.c
30	MAC_OUT	5000140	NODE	1	2	0	DSR_RREQ	0	WLAN	IEEE802_11_EV	32	29	299	C:\Users\Navya\Documents\NetSim_13.0.26_64_std_default\src\Simulation\IEEE802_11\CSMACA.c
31	MAC_OUT	5000160	NODE	1	2	0	DSR_RREQ	0	WLAN	IEEE802_11_EV	32	30	299	C:\Users\Navya\Documents\NetSim_13.0.26_64_std_default\src\Simulation\IEEE802_11\CSMACA.c
32	MAC_OUT	5000180	NODE	1	2	0	DSR_RREQ	0	WLAN	IEEE802_11_EV	32	31	299	C:\Users\Navya\Documents\NetSim_13.0.26_64_std_default\src\Simulation\IEEE802_11\CSMACA.c
33	MAC_OUT	5000200	NODE	1	2	0	DSR_RREQ	0	WLAN	IEEE802_11_EV	32	32	299	C:\Users\Navya\Documents\NetSim_13.0.26_64_std_default\src\Simulation\IEEE802_11\CSMACA.c
34	MAC_OUT	5000220	NODE	1	2	0	DSR_RREQ	0	WLAN	IEEE802_11_EV	32	33	299	C:\Users\Navya\Documents\NetSim_13.0.26_64_std_default\src\Simulation\IEEE802_11\CSMACA.c
35	MAC_OUT	5000240	NODE	1	2	0	DSR_RREQ	0	WLAN	IEEE802_11_EV	32	34	299	C:\Users\Navya\Documents\NetSim_13.0.26_64_std_default\src\Simulation\IEEE802_11\CSMACA.c
36	MAC_OUT	5000260	NODE	1	2	0	DSR_RREQ	0	WLAN	IEEE802_11_EV	32	35	299	C:\Users\Navya\Documents\NetSim_13.0.26_64_std_default\src\Simulation\IEEE802_11\CSMACA.c
37	PHYSICAL_OUT	5000260	NODE	1	1	0	DSR_RREQ	0	WLAN	IEEE802_11_PH	72	36	391	C:\Users\Navya\Documents\NetSim_13.0.26_64_std_default\src\Simulation\IEEE802_11\IEEE802_11_

Figure 9-43: NetSim writes the file name and line number of the source code in Event Trace

9.3.4 Viewing & Accessing variables

Viewing variables while debugging code

To see the value of a variable, when debugging the mouse over the variable name in the code. A text box with variable contents appears. If the variable is a structure and contains other variables, then click on the plus sign which is there to the left of the text box. Users can pin the variable to watch by clicking on the pin icon to the right of that variable in the text box.



```
    It processes Data Packets, Route Request Packets, Route Reply Packets, Route Error
    Packets, Ack Packet
TIMER_EVENT -
    It Process the DSR Route Request timeout and the DSR Maintenance Buffer timeout.
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

*/
_declspec(dllexport) int fn_NetSim_DSR_Run()
{
    set_dsr_curr();
    if (!isDsrConfigured(pstruEventDetails->nDeviceId, dsr_get_curr_if()))
        return -1;
    switch(pstruEventDetails->nEventType)
    {
    case NETWORK_OUT_EVENT:
        switch(pstruEventDetails->nSubEventType)
        {
        case 0: //Packet from upper layer or Network in
        default:
            {
                if(!isDSRPACKET(pstruEventDetails->pPacket))
                {
                    NETSIM_IPAddress n1,n2;
                    NETSIM_IPAddress dest = pstruEventDetails->pPacket->pstruNetworkData->szDestIP;
```

Name	Lang
libDSR.dll!fn_NetSim_DSR_Run(...) Line 68	C
NetworkStack.dll!fnCallProtocol(int nProtocol) Line 246	C
libIP.dll!fn_NetSim_IP_Run(...) Line 316	C
NetworkStack.dll!fnCallProtocol(int nProtocol) Line 246	C
NetworkStack.dll!fn_NetSim_Stack_NetworkOut(...) Line 467	C
NetworkStack.dll!fn_NetSim_Stack_EventHandler(...) Line 221	C
NetworkStack.dll!fn_NetSim_Stack_InitiallizeSimulation(char * pszAppPath) Line 993	C

Figure 9-44: Viewing variables while debugging code

Adding the variable to watch

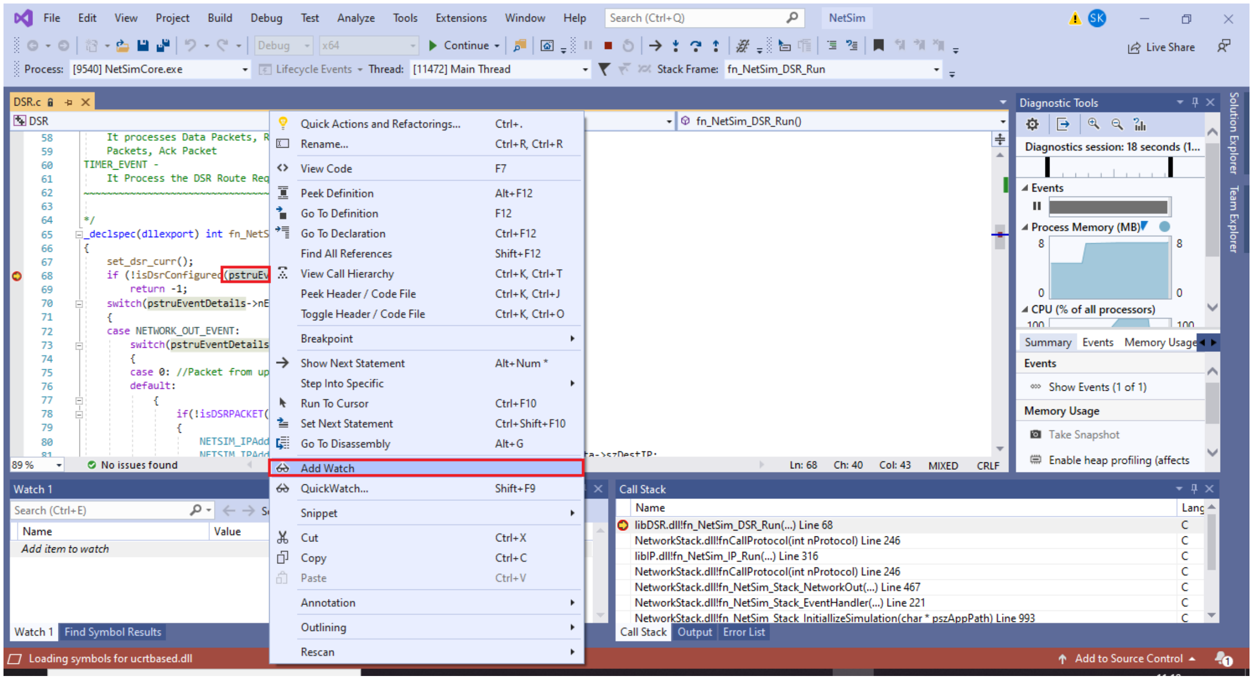

Quick Actions and Refactorings... Ctrl+.
Rename... Ctrl+R, Ctrl+R
View Code F7
Peek Definition Alt+F12
Go To Definition F12
Go To Declaration Ctrl+F12
Find All References Shift+F12
View Call Hierarchy Ctrl+K, Ctrl+T
Peek Header / Code File Ctrl+K, Ctrl+J
Toggle Header / Code File Ctrl+K, Ctrl+O
Breakpoint
Show Next Statement Alt+Num *
Step Into Specific
Run To Cursor Ctrl+F10
Set Next Statement Ctrl+Shift+F10
Go To Disassembly Alt+G
Add Watch
QuickWatch... Shift+F9
Snippet
Cut Ctrl+X
Copy Ctrl+C
Paste Ctrl+V
Annotation
Outlining
Rescan

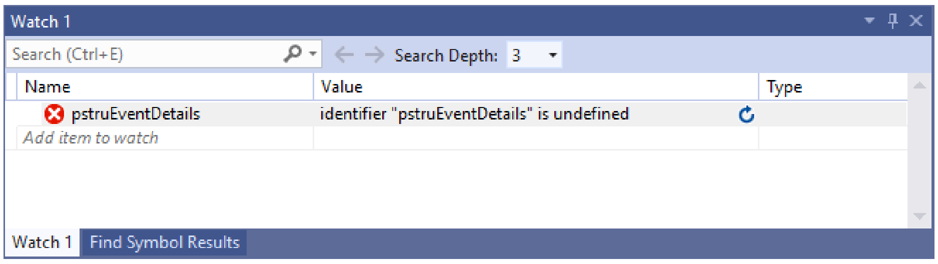
Figure 9-45: Adding the variable to watch

Watch the change in the variable as the code progress by right clicking on the variable & clicking on "**add watch**" tab. This is useful if to continuously monitor the change in the variable as the code progresses.

Viewing external variables

During the process of debug users would come across variables that are defined outside the source file being built as a .dll. Such variables cannot be viewed directly when added in the watch tab, as this would throw the error.

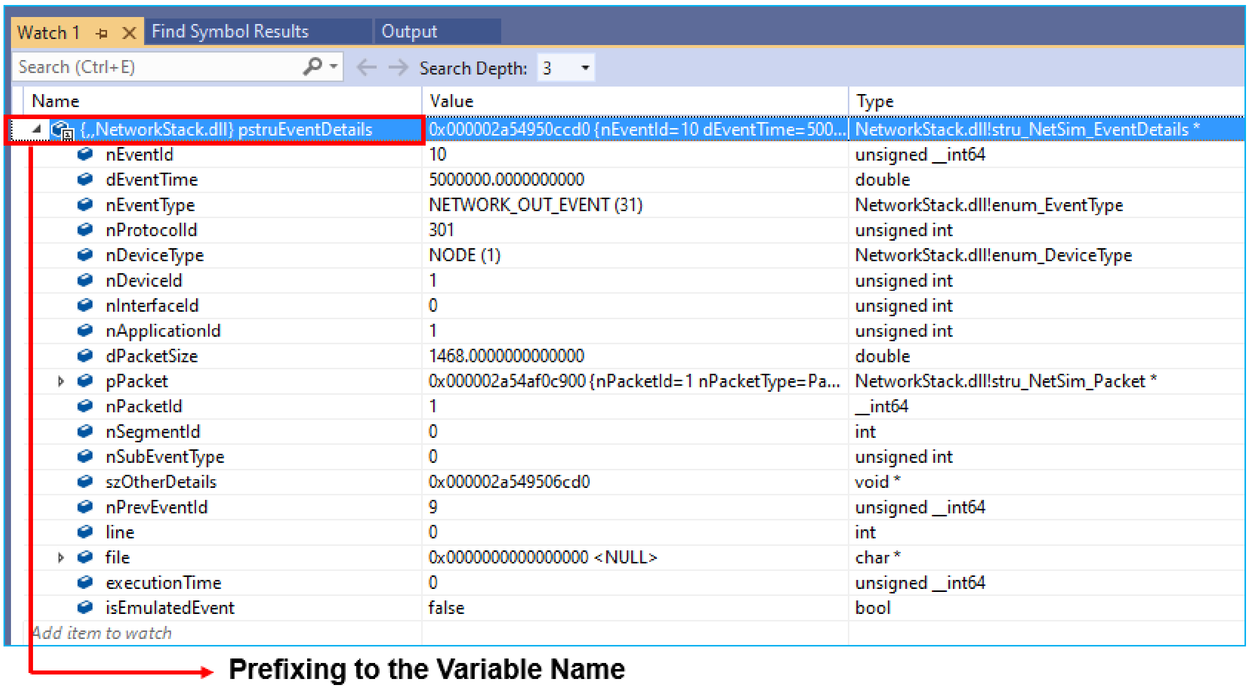
Error: Identifier “pstruEventDetails” is undefined



Name	Value	Type
pstruEventDetails	identifier "pstruEventDetails" is undefined	
Add item to watch		

Figure 9-46: Viewing external variables

In the watch window, the variable which the user has to watch should be edited by double clicking on it and prefixing {,,**NetworkStack.dll**} to the variable name and pressing enter. (The name of the respective file in which the variable is defined should be mentioned - in this case **NetworkStack.dll**).



Name	Value	Type
{,,NetworkStack.dll} pstruEventDetails	0x000002a54950ccd0 {nEventId=10 dEventTime=500...	NetworkStack.dll!stru_NetSim_EventDetails *
nEventId	10	unsigned __int64
dEventTime	5000000.0000000000	double
nEventType	NETWORK_OUT_EVENT (31)	NetworkStack.dll!enum_EventType
nProtocolId	301	unsigned int
nDeviceType	NODE (1)	NetworkStack.dll!enum_DeviceType
nDeviceId	1	unsigned int
nInterfaceId	0	unsigned int
nApplicationId	1	unsigned int
dPacketSize	1468.0000000000000	double
pPacket	0x000002a54af0c900 {nPacketId=1 nPacketType=Pa...	NetworkStack.dll!stru_NetSim_Packet *
nPacketId	1	__int64
nSegmentId	0	int
nSubEventType	0	unsigned int
szOtherDetails	0x000002a549506cd0	void *
nPrevEventId	9	unsigned __int64
line	0	int
file	0x0000000000000000 <NULL>	char *
executionTime	0	unsigned __int64
isEmulatedEvent	false	bool
Add item to watch		

Figure 9-47: Variable In the watch window

Accessing External Variables

Each protocol in NetSim has a separate Dll file which contains the variables and functions which can be shared. In case of cross layer protocol implementations variables of one protocol may have to be accessed from another Dll.

An example is given below showing how Physical layer parameters of devices running

IEEE802.11 can be accessed in the Network Layer with DSR protocol configured.

The variable **battery** is defined in a structure **stru_802_11_Phy_Var** which is part of **IEEE802_11_Phy.h** file**.** So the user will have to access a pointer of type **stru_802_11_Phy_Var**. In the header file where the structure definition is given, the following line of code must be written –

#ifndef SHARE_VARIABLE

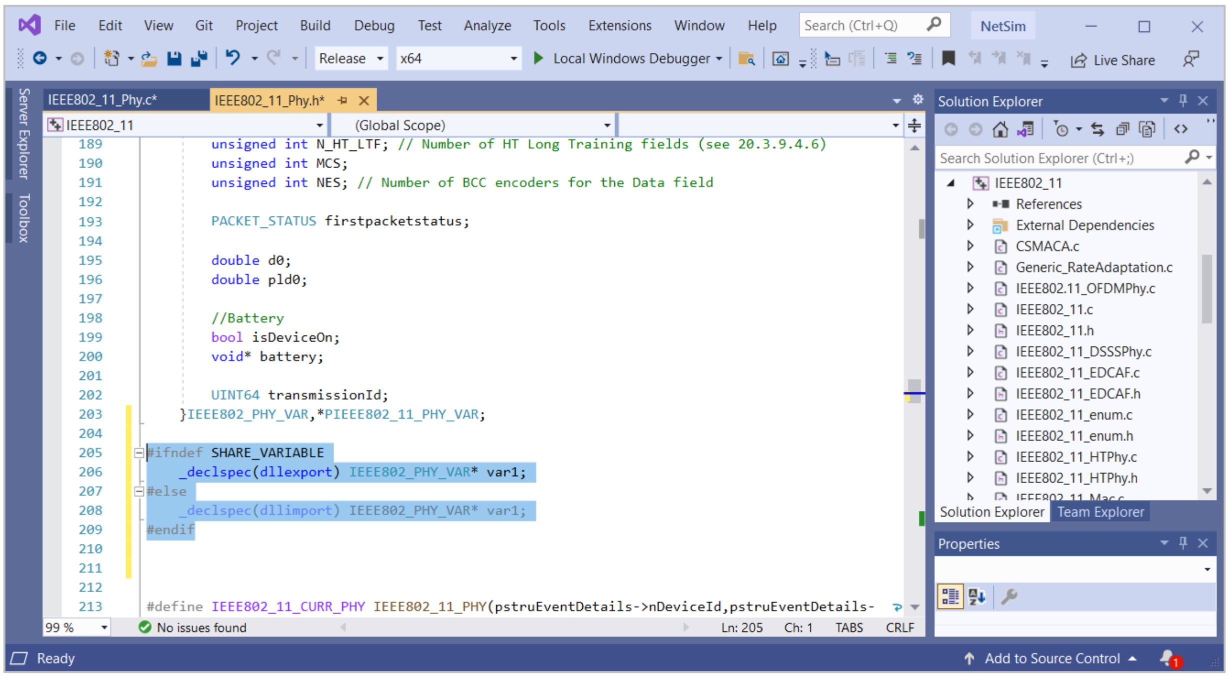
  _declspec(dllexport) IEEE802_PHY_VAR *var1;

#else

  _declspec(dllimport) IEEE802_PHY_VAR *var1;

#endif

In the example, the code line must be written in IEEE802_11_Phy.h file present inside IEEE802_11 folder.



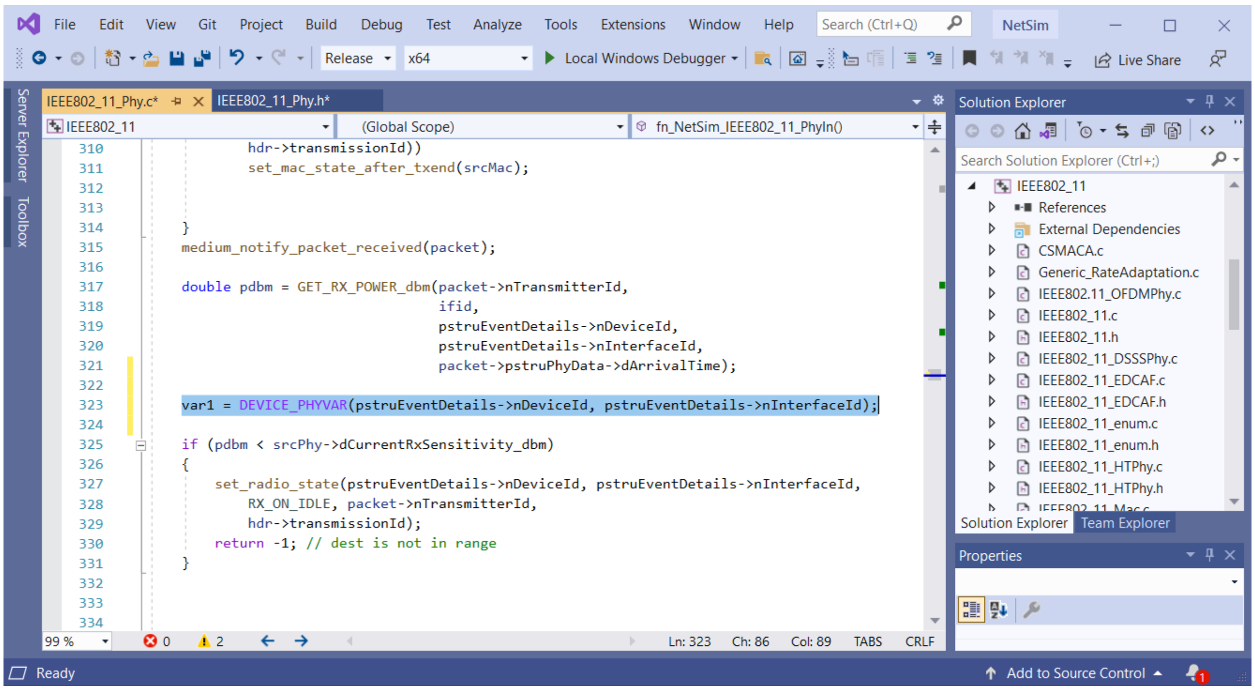
```
	unsigned int N_HT_LTF; // Number of HT Long Training fields (see 20.3.9.4.6)
	unsigned int MCS;
	unsigned int NES; // Number of BCC encoders for the Data field

	PACKET_STATUS firstpacketstatus;

	double d0;
	double pld0;

	//Battery
	bool isDeviceOn;
	void* battery;

	UINT64 transmissionId;
}IEEE802_PHY_VAR,*PIEEE802_11_PHY_VAR;

#ifndef SHARE_VARIABLE
	_declspec(dllexport) IEEE802_PHY_VAR* var1;
#else
	_declspec(dllimport) IEEE802_PHY_VAR* var1;
#endif



#define IEEE802_11_CURR_PHY IEEE802_11_PHY(pstruEventDetails->nDeviceId,pstruEventDetails-
```

Figure 9-48: Code Modification done in IEEE802_11_Phy.h file present inside IEEE802_11 folder

In the main function where a user wishes to find the dReceivedPower_mw, the variable must be assigned the respective value. In the above case, the following line of code must be written inside fn_NetSim_IEEE802_11_PhyIn() function in IEEE802_11_Phy.c file present inside IEEE802_11 folder.

var1 = DEVICE_PHYVAR(pstruEventDetails->nDeviceId,pstruEventDetails->nInterfaceId);

Note that the parameters given in the macro or any function which assigns a value to the variable must be defined beforehand in the code. Here nDeviceId and nInterfaceId are defined beforehand.



```
            hdr->transmissionId))
            set_mac_state_after_txend(srcMac);


    }
    medium_notify_packet_received(packet);

    double pdbm = GET_RX_POWER_dbm(packet->nTransmitterId,
                                   ifid,
                                   pstruEventDetails->nDeviceId,
                                   pstruEventDetails->nInterfaceId,
                                   packet->pstruPhyData->dArrivalTime);

    var1 = DEVICE_PHYVAR(pstruEventDetails->nDeviceId, pstruEventDetails->nInterfaceId);

    if (pdbm < srcPhy->dCurrentRxSensitivity_dbm)
    {
        set_radio_state(pstruEventDetails->nDeviceId, pstruEventDetails->nInterfaceId,
            RX_ON_IDLE, packet->nTransmitterId,
            hdr->transmissionId);
        return -1; // dest is not in range
    }
```

Figure 9-49: Code Modification done in IEEE802_11_Phy.c file present inside IEEE802_11 folder

The IEEE802_11 project must be built and the resulting libIEEE802.11.dll file which gets created in the bin_x64 folder of NetSim's current workspace.

C:\Users\PC\Documents\NetSim\Workspaces\<Your default workspace>\bin_x64 for 64-bit

The Object file IEEE802_11.lib which is also got created in the bin_x64 folder located in the current workspace path

<C:\Users\PC\Documents\NetSim\Workspaces\<Your default workspace>\bin_x64 >

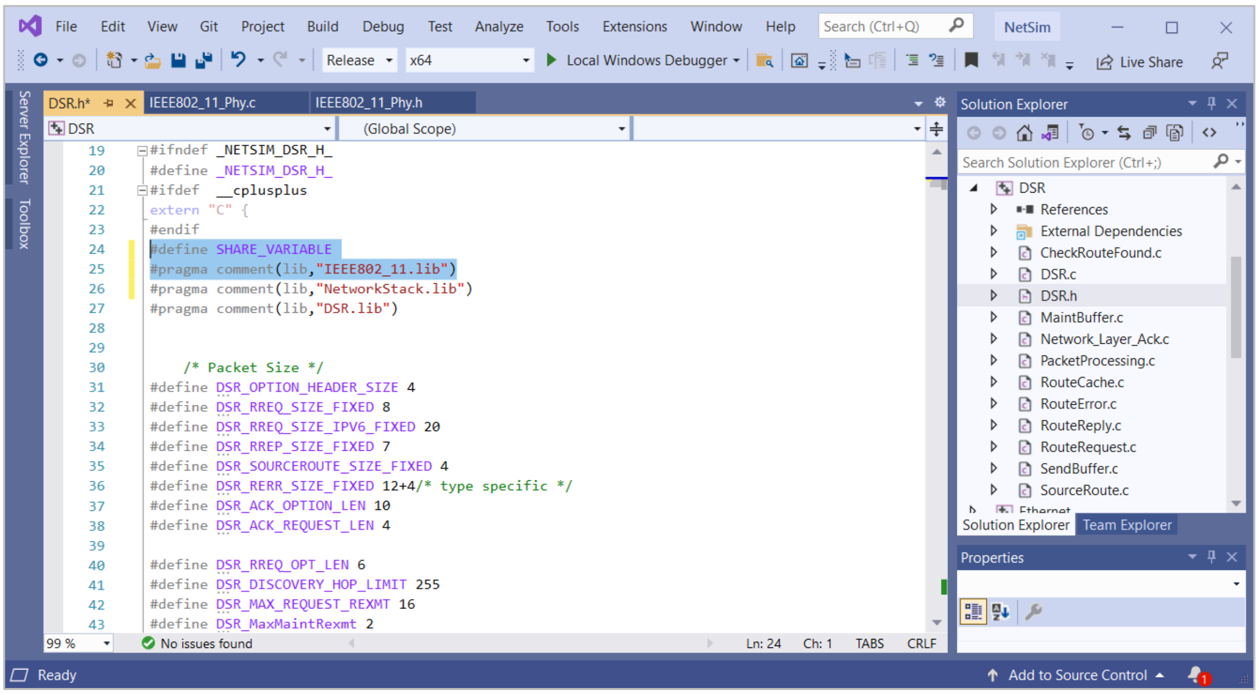
Place the created IEEE802_11.lib inside simulation **lib x64** or **lib** folder as shown in below path.

<C:\Users\PC\Documents\NetSim\Workspaces\<Your default workspace\src\Simulation\

lib_x64>for 64-bit

Now expand the DSR project in solution explorer. For accessing the IEEE802_11 variable, the following lines must be added in DSR.h file.

#define SHARE_VARIABLE
#pragma comment(lib,"IEEE802_11.lib")



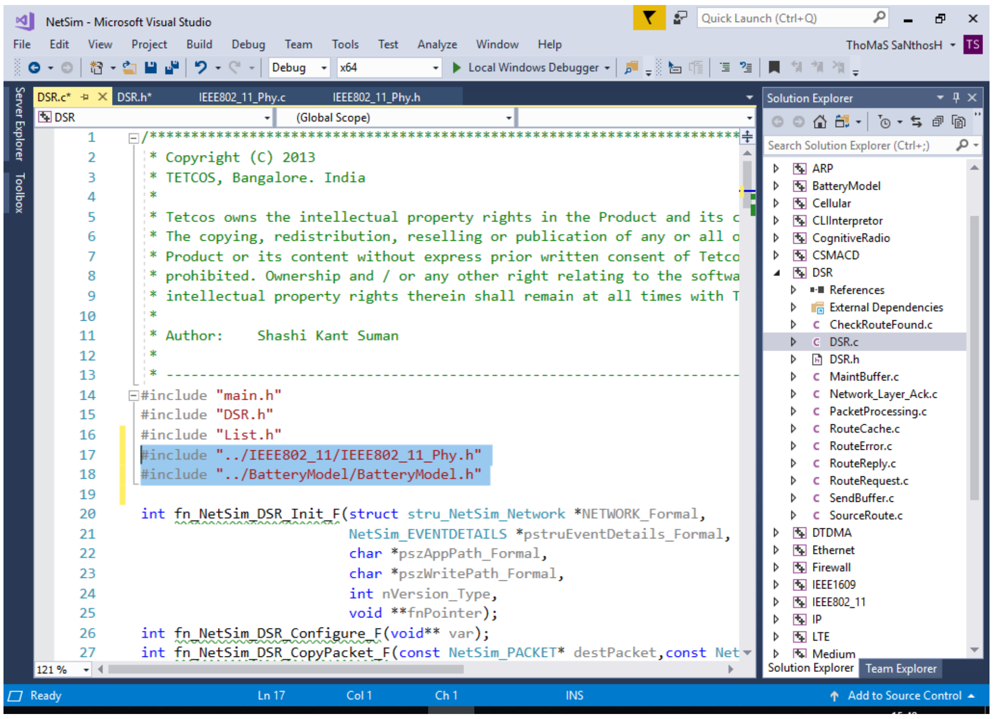
```
#ifndef _NETSIM_DSR_H_
#define _NETSIM_DSR_H_
#ifdef __cplusplus
extern "C" {
#endif
#define SHARE_VARIABLE
#pragma comment(lib,"IEEE802_11.lib")
#pragma comment(lib,"NetworkStack.lib")
#pragma comment(lib,"DSR.lib")


    /* Packet Size */
#define DSR_OPTION_HEADER_SIZE 4
#define DSR_RREQ_SIZE_FIXED 8
#define DSR_RREQ_SIZE_IPV6_FIXED 20
#define DSR_RREP_SIZE_FIXED 7
#define DSR_SOURCEROUTE_SIZE_FIXED 4
#define DSR_RERR_SIZE_FIXED 12+4/* type specific */
#define DSR_ACK_OPTION_LEN 10
#define DSR_ACK_REQUEST_LEN 4

#define DSR_RREQ_OPT_LEN 6
#define DSR_DISCOVERY_HOP_LIMIT 255
#define DSR_MAX_REQUEST_REXMT 16
#define DSR_MaxMaintRexmt 2
```

Figure 9-50: Accessing the IEEE802_11 variable, Modification done in DSR.h file

Add the following lines of code to the DSR.c file as shown below.

#include "../IEEE802_11/IEEE802_11_Phy.h"

#include "../BatteryModel/BatteryModel.h"



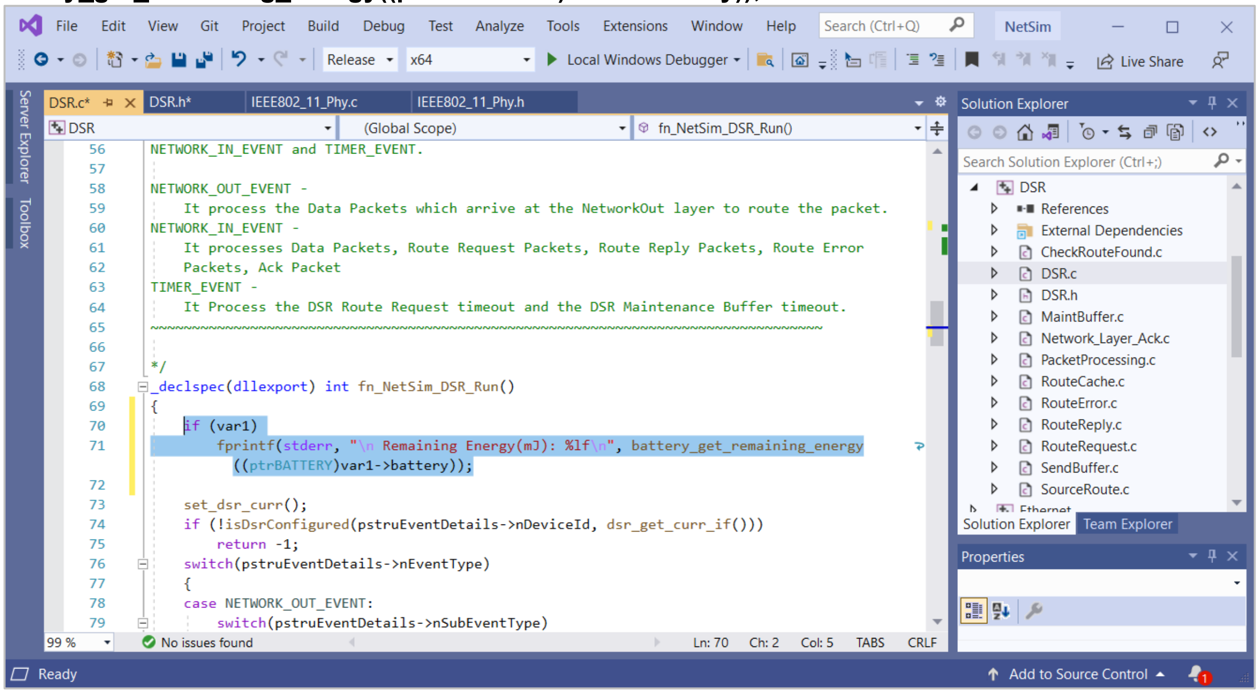
```
/*****************************************************************************
 * Copyright (C) 2013
 * TETCOS, Bangalore. India
 *
 * Tetcos owns the intellectual property rights in the Product and its c
 * The copying, redistribution, reselling or publication of any or all o
 * Product or its content without express prior written consent of Tetco
 * prohibited. Ownership and / or any other right relating to the softwa
 * intellectual property rights therein shall remain at all times with T
 *
 * Author:    Shashi Kant Suman
 *
 * ---------------------------------------------------------------------------
#include "main.h"
#include "DSR.h"
#include "List.h"
#include "../IEEE802_11/IEEE802_11_Phy.h"
#include "../BatteryModel/BatteryModel.h"

int fn_NetSim_DSR_Init_F(struct stru_NetSim_Network *NETWORK_Formal,
                         NetSim_EVENTDETAILS *pstruEventDetails_Formal,
                         char *pszAppPath_Formal,
                         char *pszWritePath_Formal,
                         int nVersion_Type,
                         void **fnPointer);
int fn_NetSim_DSR_Configure_F(void** var);
int fn_NetSim_DSR_CopyPacket_F(const NetSim_PACKET* destPacket,const Net
```

Figure 9-51: Add the following lines of code to the DSR.c file in DSR Project

In the fn_NetSim_DSR_Run() function add the following lines of code to print the value of dReceivedPower_mw variable from DSR project.

if (var1)

fprintf(stderr, "\n Remaining Energy(mJ): %lf\n" ,battery_get_remaining_energy((ptrBATTERY)var1->battery));



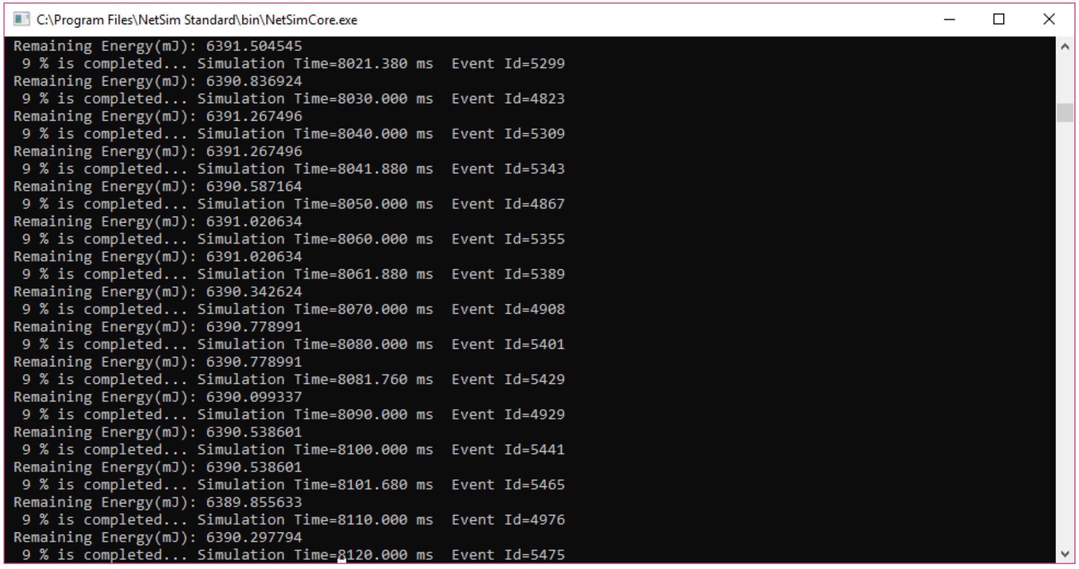
```
NETWORK_IN_EVENT and TIMER_EVENT.

NETWORK_OUT_EVENT -
    It process the Data Packets which arrive at the NetworkOut layer to route the packet.
NETWORK_IN_EVENT -
    It processes Data Packets, Route Request Packets, Route Reply Packets, Route Error
    Packets, Ack Packet
TIMER_EVENT -
    It Process the DSR Route Request timeout and the DSR Maintenance Buffer timeout.
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

*/
_declspec(dllexport) int fn_NetSim_DSR_Run()
{
    if (var1)
        fprintf(stderr, "\n Remaining Energy(mJ): %lf\n", battery_get_remaining_energy
          ((ptrBATTERY)var1->battery));

    set_dsr_curr();
    if (!isDsrConfigured(pstruEventDetails->nDeviceId, dsr_get_curr_if()))
        return -1;
    switch(pstruEventDetails->nEventType)
    {
    case NETWORK_OUT_EVENT:
        switch(pstruEventDetails->nSubEventType)
```

Figure 9-52: Code Related to Remaining Energy for nodes

The DSR project must be built and the resulting libDSR.dll file gets created in the bin_x64 folder of NetSim’s current workspace path

< C:\Users\PC\Documents\NetSim\Workspaces\<Your default workspace>\bin_x64> for 64-bit.

When a scenario is run, the remaining energy of the node will be printed to the simulation console as shown below.



C:\Program Files\NetSim Standard\bin\NetSimCore.exe

```
Remaining Energy(mJ): 6391.504545
 9 % is completed... Simulation Time=8021.380 ms  Event Id=5299
Remaining Energy(mJ): 6390.836924
 9 % is completed... Simulation Time=8030.000 ms  Event Id=4823
Remaining Energy(mJ): 6391.267496
 9 % is completed... Simulation Time=8040.000 ms  Event Id=5309
Remaining Energy(mJ): 6391.267496
 9 % is completed... Simulation Time=8041.880 ms  Event Id=5343
Remaining Energy(mJ): 6390.587164
 9 % is completed... Simulation Time=8050.000 ms  Event Id=4867
Remaining Energy(mJ): 6391.020634
 9 % is completed... Simulation Time=8060.000 ms  Event Id=5355
Remaining Energy(mJ): 6391.020634
 9 % is completed... Simulation Time=8061.880 ms  Event Id=5389
Remaining Energy(mJ): 6390.342624
 9 % is completed... Simulation Time=8070.000 ms  Event Id=4908
Remaining Energy(mJ): 6390.778991
 9 % is completed... Simulation Time=8080.000 ms  Event Id=5401
Remaining Energy(mJ): 6390.778991
 9 % is completed... Simulation Time=8081.760 ms  Event Id=5429
Remaining Energy(mJ): 6390.099337
 9 % is completed... Simulation Time=8090.000 ms  Event Id=4929
Remaining Energy(mJ): 6390.538601
 9 % is completed... Simulation Time=8100.000 ms  Event Id=5441
Remaining Energy(mJ): 6390.538601
 9 % is completed... Simulation Time=8101.680 ms  Event Id=5465
Remaining Energy(mJ): 6389.855633
 9 % is completed... Simulation Time=8110.000 ms  Event Id=4976
Remaining Energy(mJ): 6390.297794
 9 % is completed... Simulation Time=8120.000 ms  Event Id=5475
```

Figure 9-53: Remaining energy of the node printed in NetSim console

9.3.5 Print to console window in NetSim

Users can try printing the Device ID, Application ID, Duplicate Ack Count etc.

To print to console: Print node positions in MANET

Open Mobility Project, and in Mobility.c file go to fn_NetSim_Mobility_Run() function. Inside the default case add following codes

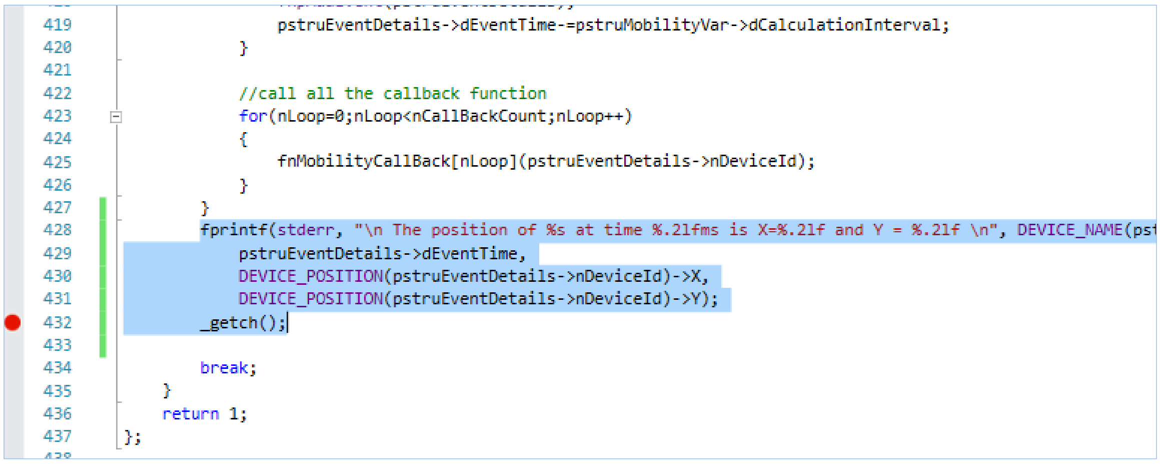
fprintf(stderr,"\n The position of %s at time %.2lfms is X=%.2lf and Y = %.2lf \n",DEVICE_NAME(pstruEventDetails->nDeviceId),

pstruEventDetails->dEventTime,

DEVICE_POSITION(pstruEventDetails->nDeviceId)->X,

DEVICE_POSITION(pstruEventDetails->nDeviceId)->Y);

_getch();



```
                pstruEventDetails->dEventTime-=pstruMobilityVar->dCalculationInterval;
            }

            //call all the callback function
            for(nLoop=0;nLoop<nCallBackCount;nLoop++)
            {
                fnMobilityCallBack[nLoop](pstruEventDetails->nDeviceId);
            }
        }
        fprintf(stderr, "\n The position of %s at time %.2lfms is X=%.2lf and Y = %.2lf \n", DEVICE_NAME(pst
            pstruEventDetails->dEventTime,
            DEVICE_POSITION(pstruEventDetails->nDeviceId)->X,
            DEVICE_POSITION(pstruEventDetails->nDeviceId)->Y);
        _getch();

        break;
    }
    return 1;
};
```

Figure 9-54: Code for Print node positions in MANET

Building Mobility project creates libMobility.dll inside the binary folder(bin_x64) of NetSim's current workspace path

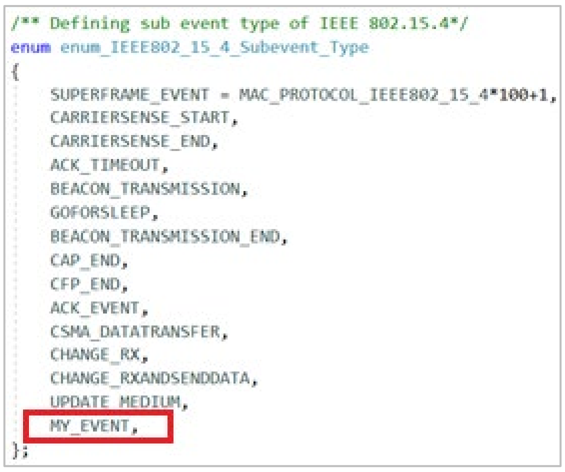
<C:\Users\PC\Documents\NetSim\Workspaces\<Your default workspace>\bin_x64 > for 64-bit.

Create a scenario in MANET and configure the mobility model of the nodes. During simulation users can notice that the positions of the nodes are displayed in the console w.r.t. the simulation time.

9.4 Creating a new packet and adding a new event in NetSim

In this example we show how users can create their own packet & event in 802.15.4 Zigbee. The same methodology can be applied to any network / protocol.

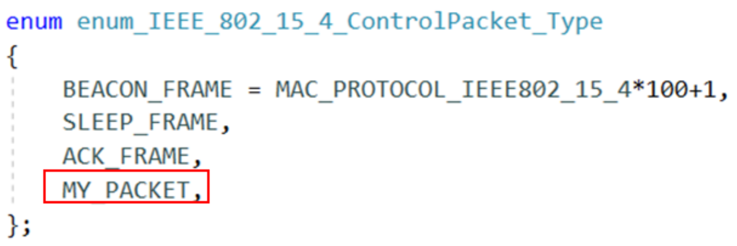
1. Open the Source codes in Visual studio using the NetSim.sln file.
2. Go to ZigBee project and Open 802_15_4.h file and add a subevent called "MY_EVENT" inside enum_IEEE802_15_4_Subevent_Type as shown below.



```
/** Defining sub event type of IEEE 802.15.4*/
enum enum_IEEE802_15_4_Subevent_Type
{
    SUPERFRAME_EVENT = MAC_PROTOCOL_IEEE802_15_4*100+1,
    CARRIERSENSE_START,
    CARRIERSENSE_END,
    ACK_TIMEOUT,
    BEACON_TRANSMISSION,
    GOFORSLEEP,
    BEACON_TRANSMISSION_END,
    CAP_END,
    CFP_END,
    ACK_EVENT,
    CSMA_DATATRANSFER,
    CHANGE_RX,
    CHANGE_RXANDSENDDATA,
    UPDATE_MEDIUM,
    MY_EVENT,
};
```

Figure 9-55: Add “MY_EVENT” inside enum_IEEE802_15_4_Subevent_Type

3. To add a new packet in NetSim first user has to initialize their new packet name inside 802_15_4.h file. Let us assume the new packet be “MY_PACKET” and it is a control packet. So, user has to define it inside the following enum as shown below Figure 9-57



```
enum enum_IEEE_802_15_4_ControlPacket_Type
{
    BEACON_FRAME = MAC_PROTOCOL_IEEE802_15_4*100+1,
    SLEEP_FRAME,
    ACK_FRAME,
    MY_PACKET,
};
```

Figure 9-56: Add “MY PACKET” inside enum_IEEE802_15_4_ControlPacket_Type

4. We assume that MY_PACKET has the same fields as a Zigbee Ack and hence we are adding the following ack frame to 802_15_4.h file (Add this code just above the enum enum_IEEE_802_15_4_ControlPacket_Type {} defenition):

```
        struct stru_My_Frame
        {
                int nBeaconId;
                int nSuperFrameId;
                int nBeaconTime;
                double dPayload;
                double dOverhead;
                double dFrameSize;
        };
    typedef struct stru_My_Frame MY_FRAME;
        enum enum_IEEE_802_15_4_ControlPacket_Type
        {
```

5. Open 802_15_4.c file, go to the case TIMER_EVENT and add the following code to the subevent type.

```
case SUBEVENT_GETLINKQUALITY:
{
--------------------------
}
break;
case MY_EVENT:
{
//my event//
     fprintf(stderr, "My_event");
     pstruEventDetails->dEventTime = pstruEventDetails->dEventTime + 1 * SECOND;
     pstruEventDetails->nDeviceId = nGlobalPANCoordinatorId;
     pstruEventDetails->nInterfaceId = 1;
     pstruEventDetails->nEventType = TIMER_EVENT;
     pstruEventDetails->nSubEventType = MY_EVENT;
     pstruEventDetails->nProtocolId = MAC_PROTOCOL_IEEE802_15_4;
     fnpAddEvent(pstruEventDetails);
     fn_NetSim_WSN_MY_PACKET();
//my event//
}
break;
```

Here we are adding a new event inside the timer event, and this event will occur every 1 second in the GlobalPANCoordinator. i.e., sink node. In this event, fn_NetSim_WSN_MY_PACKET() is called as explained in step 5.

6. Inside 802_15_4.c file, add the following code at the end of the file for sending ack (broadcast):

```
int fn_NetSim_WSN_MY_PACKET()
{
      double dTime;
      NETSIM_ID nDeviceId = pstruEventDetails->nDeviceId;
      NETSIM_ID nInterfaceId = pstruEventDetails->nInterfaceId;
      IEEE802_15_4_MAC_VAR *pstruMacVar =
DEVICE_MACVAR(nDeviceId, nInterfaceId);
      IEEE802_15_4_PHY_VAR *pstruPhyVar = DEVICE_PHYVAR(nDeviceId,
nInterfaceId);
      NetSim_PACKET *pstruPacket = pstruEventDetails->pPacket;
      NetSim_PACKET *pstruAckPkt;
      MY_FRAME *pstruAck;
```

```
        dTime = pstruEventDetails->dEventTime;

        // Create MY_Frame
        pstruAckPkt = fn_NetSim_Packet_CreatePacket(MAC_LAYER);
        pstruAckPkt->nPacketType = PacketType_Control;
        pstruAckPkt->nPacketPriority = Priority_High;
        pstruAckPkt->nControlDataType = MY_PACKET;
        pstruAck = fnpAllocateMemory(1, sizeof(MY_FRAME));

        // Update packet fields
        pstruAckPkt->nSourceId = nDeviceId;
        pstruAckPkt->nTransmitterId = nDeviceId;
        pstruAckPkt->nReceiverId = 0;
        add_dest_to_packet(pstruAckPkt, pstruAckPkt->nReceiverId);
        pstruAckPkt->pstruMacData->Packet_MACProtocol = pstruAck;
        pstruAckPkt->pstruMacData->dArrivalTime = dTime;
        pstruAckPkt->pstruMacData->dStartTime = dTime;
        pstruAckPkt->pstruMacData->dEndTime = dTime;
        pstruAckPkt->pstruMacData->dPacketSize =
 pstruAckPkt->pstruMacData->dOverhead;
        pstruAckPkt->pstruMacData->nMACProtocol =
MAC_PROTOCOL_IEEE802_15_4;
        pstruAckPkt->nPacketId = 0;
        strcpy(pstruAckPkt->szPacketType, "MY_PACKET");
//to see the packet in animation
// Add SEND ACK subevent
        pstruEventDetails->dEventTime = dTime;
        pstruEventDetails->dPacketSize =
pstruAckPkt->pstruMacData->dPacketSize;
        pstruEventDetails->nSubEventType = 0;
        pstruEventDetails->nEventType = PHYSICAL_OUT_EVENT;
        pstruEventDetails->pPacket = pstruAckPkt;
        fnpAddEvent(pstruEventDetails);

        //Free the packet
```

```
            fn_NetSim_Packet_FreePacket(pstruPacket);
            pstruPacket = NULL;
            return 0;
        }
```

7. Inside the above function NetSim API *fn_NetSim_Packet_CreatePacket(MAC_LAYER);* is used. This is the API which creates a new packet in NetSim. Since in this example, new packet is created in MAC layer, it is passed as an argument. Users can give the respective Layer name for creating packets in any other layers. In the above code users can see the following line:

strcpy(pstruAckPkt->szPacketType, "MY_PACKET");

This is used visualize the packet transmission in the packet animation.

8. In 802_15_4.c file, goto fn_NetSim_Zigbee_Init() function and add the following code in red color to call the timer_event. i.e. MY_EVENT

```
_declspec (dllexport) int fn_NetSim_Zigbee_Init()
    {
    //MY_EVENT
pstruEventDetails->nDeviceId = nGlobalPANCoordinatorId;
        pstruEventDetails->nInterfaceId = 1;
pstruEventDetails->dEventTime = pstruEventDetails->dEventTime;
        pstruEventDetails->nEventType = TIMER_EVENT;     pstruEventDetails-
>nSubEventType = MY_EVENT;
pstruEventDetails->nProtocolId = MAC_PROTOCOL_IEEE802_15_4;
        fnpAddEvent(pstruEventDetails);
//MY_EVENT
return fn_NetSim_Zigbee_Init_F();
}
```

In the above function, subevent type, “MY_EVENT” is called. So this function calls the MY_EVENT, timer event to execute.

9. fn_NetSim_Zigbee_Trace() is an API to print the trace details to the event trace. So inside 802_15_4.c file add the following lines of code in red color inside fn_NetSim_Zigbee_Trace(int nSubEvent) as shown below:-

```
_declspec (dllexport) char *fn_NetSim_Zigbee_Trace(int nSubEvent)
{
    if (nSubEvent == MY_EVENT)
    return "MY_EVENT";
    return (fn_NetSim_Zigbee_Trace_F(nSubEvent));
    }
```

10. Save the code and build Zigbee project, libZigBee.dll will get created in the bin folder of NetSim's current workspace path.

< C:\Users\PC\Documents\NetSim\Workspaces\<Your default workspace>\bin_x64 > for 64-bit.

11. Create a basic scenario in WSN with 2 sensors and 1 sink node.
12. While creating the application between two sensors, set

    Application Type - Sensor App

    Interarrival Time - 5000
13. Run simulation for 10 seconds.
14. Play packet animation. Here users can see that Sink node broadcasts "MY_PACKET" to the sensor.

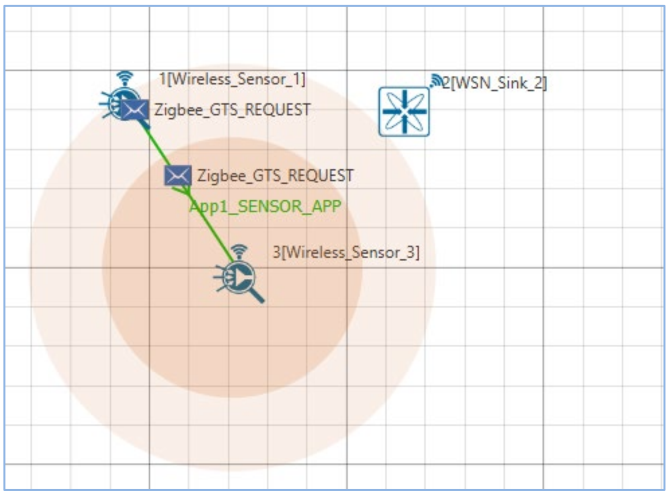
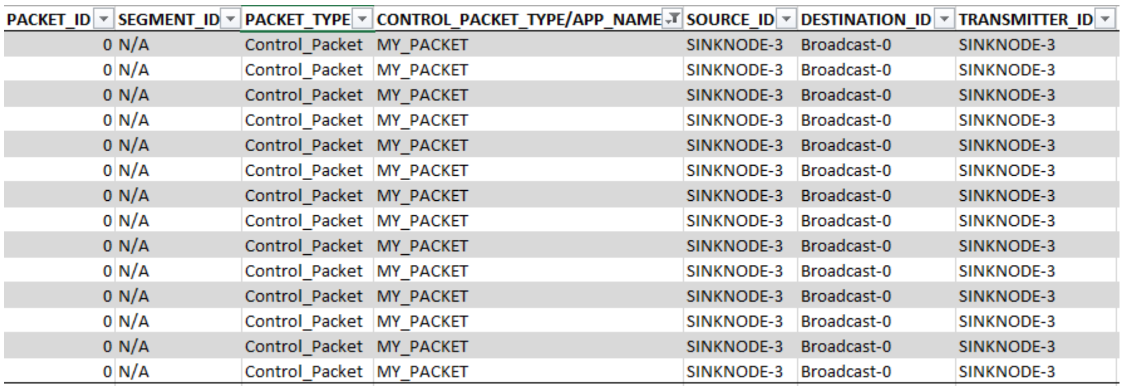


Figure 9-57: In animation window Sink node broadcasts "MY_PACKET" to the sensor

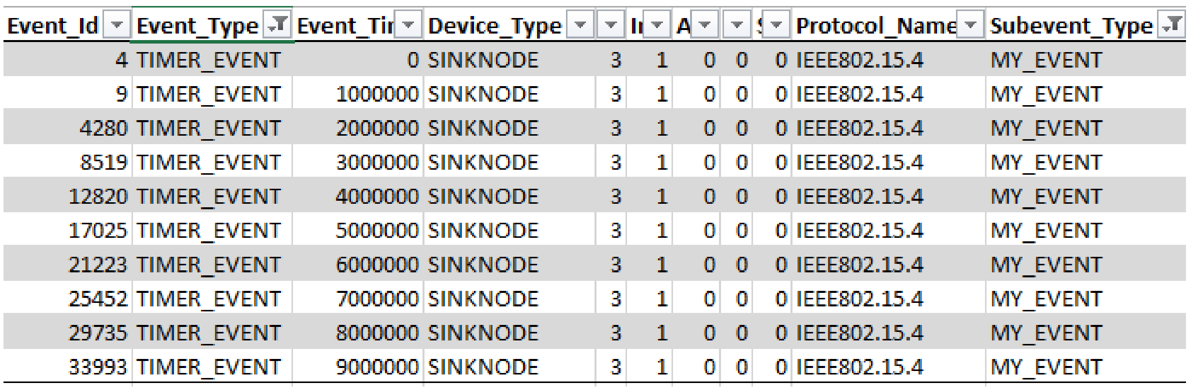
15. Also, open packet trace and users can filter the control packet and see all the packet details of "MY_PACKET" written in the packet trace.



PACKET_ID	SEGMENT_ID	PACKET_TYPE	CONTROL_PACKET_TYPE/APP_NAME	SOURCE_ID	DESTINATION_ID	TRANSMITTER_ID
0	N/A	Control_Packet	MY_PACKET	SINKNODE-3	Broadcast-0	SINKNODE-3
0	N/A	Control_Packet	MY_PACKET	SINKNODE-3	Broadcast-0	SINKNODE-3
0	N/A	Control_Packet	MY_PACKET	SINKNODE-3	Broadcast-0	SINKNODE-3
0	N/A	Control_Packet	MY_PACKET	SINKNODE-3	Broadcast-0	SINKNODE-3
0	N/A	Control_Packet	MY_PACKET	SINKNODE-3	Broadcast-0	SINKNODE-3
0	N/A	Control_Packet	MY_PACKET	SINKNODE-3	Broadcast-0	SINKNODE-3
0	N/A	Control_Packet	MY_PACKET	SINKNODE-3	Broadcast-0	SINKNODE-3
0	N/A	Control_Packet	MY_PACKET	SINKNODE-3	Broadcast-0	SINKNODE-3
0	N/A	Control_Packet	MY_PACKET	SINKNODE-3	Broadcast-0	SINKNODE-3
0	N/A	Control_Packet	MY_PACKET	SINKNODE-3	Broadcast-0	SINKNODE-3
0	N/A	Control_Packet	MY_PACKET	SINKNODE-3	Broadcast-0	SINKNODE-3
0	N/A	Control_Packet	MY_PACKET	SINKNODE-3	Broadcast-0	SINKNODE-3
0	N/A	Control_Packet	MY_PACKET	SINKNODE-3	Broadcast-0	SINKNODE-3
0	N/A	Control_Packet	MY_PACKET	SINKNODE-3	Broadcast-0	SINKNODE-3

Figure 9-58: Filter the Packet Type to control packet and See "MY_PACKET" in Packet Trace

16. To analyze the "MY_EVENT" users can open event trace and filter the subevent type as "MY_EVENT". Here users can analyze that the event occurs for every 1 seconds.



Event_Id	Event_Type	Event_Tir	Device_Type		I	A		S	Protocol_Name	Subevent_Type
4	TIMER_EVENT	0	SINKNODE	3	1	0	0	0	IEEE802.15.4	MY_EVENT
9	TIMER_EVENT	1000000	SINKNODE	3	1	0	0	0	IEEE802.15.4	MY_EVENT
4280	TIMER_EVENT	2000000	SINKNODE	3	1	0	0	0	IEEE802.15.4	MY_EVENT
8519	TIMER_EVENT	3000000	SINKNODE	3	1	0	0	0	IEEE802.15.4	MY_EVENT
12820	TIMER_EVENT	4000000	SINKNODE	3	1	0	0	0	IEEE802.15.4	MY_EVENT
17025	TIMER_EVENT	5000000	SINKNODE	3	1	0	0	0	IEEE802.15.4	MY_EVENT
21223	TIMER_EVENT	6000000	SINKNODE	3	1	0	0	0	IEEE802.15.4	MY_EVENT
25452	TIMER_EVENT	7000000	SINKNODE	3	1	0	0	0	IEEE802.15.4	MY_EVENT
29735	TIMER_EVENT	8000000	SINKNODE	3	1	0	0	0	IEEE802.15.4	MY_EVENT
33993	TIMER_EVENT	9000000	SINKNODE	3	1	0	0	0	IEEE802.15.4	MY_EVENT

Figure 9-59: Filter Subevent type to “MY_EVENT”

9.5 NetSim API’s

NetSim provides a wide variety of APIs for protocol developers. These are available in

1. **packet.h** – Packet related APIs
   - Create a new packet.
     - fn_NetSim_Packet_CreatePacket_dbg(int nLayer,int line,const char* file);
   - Copy a packet into a new packet.
     - fn_NetSim_Packet_CopyPacket_dbg(const NetSim_PACKET* pstruPacket,int line,const char* file);
   - Create error in packet.
     - fn_NetSim_Packet_DecideError(double dBER, long double dPacketSize);
   - Free a packet
     - fn_NetSim_Packet_FreePacket_dbg(NetSim_PACKET**pstruPacket,int line,char* file);
2. **stack.h** – Network / device / link and event related APIs
   - Calculate distance between nodes.
     - fn_NetSim_Utilities_CalculateDistance(NetSim_COORDINATES* coordinate1,NetSim_COORDINATES* coordinates2);
   - Stores the event details. Only one-time memory is allocated. Most used variable
     - struct stru_NetSim_EventDetails* pstruEventDetails;
   - Retrieve values from xml file.
     - GetXmlVal(void* var,char* name,void* xmlNode,int flag, XMLDATATYPE type);
3. **list.h** -- Optimized list operation calls since NetSim uses lists extensively.
   - Add elments in list.

- list_add(void** list,void* mem,size_t offset,int (*check)(void* current,void* mem));

- Sorting the list
    - list_sort(void** list,size_t offset,int (*check)(void* current, void* mem));

4. **IP_Addressing.h** – For setting & getting IP address per the appropriate format.

- Set Ip address of any node.
    - NETSIM_IPAddress
- Checking ip address is broadcast or multicast.
    - isBroadcastIP(NETSIM_IPAddress ip);
    - isMulticastIP(NETSIM_IPAddress ip);

For detailed help please refer the appropriateheader (.h) files inside:/NetSim_Standard/src/simulation/include or read through the doxygen source code documentation available inside Home Page under Documentation > source code Help

- Include all the header (.h) files from the include folder.
- NetworkStack.lib is a "import library" file and has the definitions for the functions present in the NetworkStack.dll
- When developing new protocols users should create their own protocol.h and declare all the protocol specific variables here. Stack & packet related variables should be used from stack.h and packet.h

<u>NetSim Network Stack calling individual Protocol.</u>

Every protocol should provide the following APIs as hooks to the network stack:

- **int** (*fn_NetSim_protocol_init)(**conststruct** stru_NetSim_Network*,**conststruct** stru_NetSim_EventDetails*,**constchar*,constchar*,int,constvoid****);
- Using this API the stack passes all the relevant pointers to variables, paths etc needed for the protocol. Inside this function a) local variables should be initialized, b) Initial events if any should be written, eg: Hello packet in RIP, STP in Ethernet c) File pointers for reading & writing protocol_specific_IO files.
- **int** (*fn_NetSim_protocol_Configure)(**conststruct** stru_NetSim_Network*,**int** nDeviceId, **int** nINterfaceID, **int** nlayertype, fnpAllocateMemory, fnpFreeMemory, fpConfigLog);
- The stack calls this API when reading the config file. Upon reaching the appropriate protocol definition in the XML file, the stack calls this and passes all these pointers to the protocol.
- **int** (*fn_NetSim_protocol_run)(): This is called by the stack to run the protocol
- **char*** (*fn_NetSim_protocol_trace)(**int**): This called by the stack to write the event trace

- **int**(*fn_NetSim_protocol_CopyPacket)(**const**NetSim_PACKET* pstruDestPacket,**const** NetSim_PACKET* pstruSrcPacket):
- This is for copying protocol specific parameters / data into the packed
- int (*fn_NetSim_protocol_FreePacket)(**const** NetSim_PACKET* pstruPacket): The this to free the protocol specific parameters / data in the packet
- (*fn_NetSim_protocol_Metrics)(**const** FILE* fpMetrics): This is to write the metrics file upon completion of the simulation
- **int** (*fn_NetSim_protocol_Finish)(): To release all memory after completion
- char* (*fn_NetSim_protocol_ConfigPacketTrace)(**constvoid*** xmlNetSimNode); To configure the packet packet trace in terms of the parameters to be logged
- char* (*fn_NetSim_protocol_WritePacketTrace)(**const** NetSim_PACKET*); To configure the event trace in terms of the parameters to be logged.

10 Advanced Features

10.1 Random Number Generator and Seed Values

NetSim includes protocol and traffic models which include stochastic behavior. Typical examples are a) wi-fi node's random back-off after collisions, and b) packet error decision by comparing a random number chosen between 0 and 1 against the packet error probability.

NetSim uses an in-built linear congruential Random Number Generator (RNG) to generate the randomness. The RNG uses two seeds values to initialize the RNG. Having the same set of seed values ensures that for a particular network configuration the same output results will be got, irrespective of the system or time at which the simulation is run. This ensures repeatability of experimentation.

Modifying the seed value will lead to the generation of a different set of random numbers and thereby lead to a different sequence of events in NetSim. Therefore, the results are dependent on the initial seeding of the RNG. Because a particular random seed can potentially result in an anomalous, or non-representative behavior, it is important for each network scenario to be simulated with several random number seeds, to ascertain typical performance.

10.2 Confidence in simulation results and error bars

Since NetSim's models include stochastic behavior, results are dependent on the initial seeding of the random number generator. Because a particular random seed selection can potentially result in an anomalous, or non-representative behavior, it is important for each model configuration to be exercised with several random number seeds, to be able to determine standard or typical behavior.

The field of statistics provides methods for calculating confidence in an estimate, based on a trial or series of random trials. To calculate confidence intervals, users can do the following:

- Run N simulations for the same model configuration, with a different initial seed (for the random number generator) for each run.
- For any output metric X, calculate the mean (average) $\bar{X}$ of the N samples.

$$\bar{X} = \frac{1}{N}(X_1 + X_2 + \cdots + X_N)$$

- Calculate the standard deviation σ of the N samples.

$$\sigma = \sqrt{\left(\frac{1}{N-1}\right) \times \Sigma\,(X_i - \bar{X})^2}$$

- Confidence interval limits can be expressed as

$$\Theta_{\text{lower}} = \bar{X} - z_\alpha \times \left(\frac{\sigma}{\sqrt{(N)}}\right),\ \Theta_{\text{upper}} = \bar{X} + z_\alpha \times \left(\frac{\sigma}{\sqrt{(N)}}\right)$$

  This statement can be thought of as assigning a probability to the condition that the true mean μ is within a particular distance of the random sample $\bar{X}$, as shown below

$$Prob\left[\bar{X} - z_\alpha \times \left(\frac{\sigma}{\sqrt{(N)}}\right) < \mu < \bar{X} + z_\alpha \times \left(\frac{\sigma}{\sqrt{(N)}}\right)\right] = \alpha$$

Confidence level (α)	z_α
99%	2.575
98%	2.327
95%	1.960
90%	1.645
80%	1.282

Table 10-1:Value of $\boldsymbol{Z_\alpha}$ for different confidence intervals

- The above expression for confidence assumes the distribution of the output metric is Normal, which is true (from the Central Limit Theorem) if the number of runs $N \geq 30$. If the number of repetitions is less than 30 then it is better to use the t-statistic based confidence-interval (details can be found in standard statistics textbooks).

Error bars are plotted as a vertical bar centered at the mean and ending at the upper and lower confidence limits

10.3 Interfacing MATLAB with NetSim (Std/Pro versions)

NetSim provides run-time interfacing with MATLAB so that users do not have to rewrite code in C for features that are already available in MATLAB and instead simply reuse MATLAB code. Lot of work related to machine learning, artificial intelligence and specialized mathematical algorithms which can be used for networking research, can be carried out using existing MATLAB code.

NETSIM
MATLAB
Figure 10-1: Interfacing MATLAB with NetSim

This interfacing feature can be used to either replace an existing functionality in NetSim or to incorporate additional functionalities supported by MATLAB. Any existing command/function/algorithm in MATLAB or a MATLAB M-script can be used.

In general, the following are done when a user interfaces NetSim to MATLAB:

- Initialize a MATLAB engine process in parallel with NetSim,
- Execute MATLAB workspace commands,
- Pass parameters to MATLAB workspace,
- Read parameters from MATLAB workspace,
- Generate dynamic three-dimensional plots,
- Make calls to functions that are part of MATLAB M-scripts or .m files, and
- Terminate the MATLAB engine process at NetSim simulation end.

Guidelines to interface NetSim with MATLAB

- Analyze what parameters the function or code in MATLAB expects as input.
- Identify the relevant variables in NetSim to be passed as input to MATLAB.
- Make calls from relevant places of NetSim source code to
    - Pass parameters from NetSim to MATLAB.
    - To read computed parameters from MATLAB workspace
- Identify and update the appropriate simulation variables in NetSim.

NetSim offers two different methods to interact with MATLAB during runtime.

- NetSim-MATLAB Socket Interface
- NetSim-MATLAB COM Interface

The first is an inbuilt socket interface that offers simplified NetSim API's that can be used for interactions with MATLAB. The second method involves importing MATLAB libraries, including MATLAB headers, and using MATLAB Engine APIs for all interactions with MATLAB. The former is a lot simpler since no additional settings will be required in the source code project settings. The later involves multiple configuration steps but offers more flexibility.

10.3.1 NetSim-MATLAB Socket Interface

NetSim-MATLAB Socket Interface was introduced in NetSim v13.0 to reduce the complexity involved in the NetSim-MATLAB COM Interface.

NetSim provides several inbuilt APIs that can be called from the underlying protocol C source codes to interact with MATLAB. Following is some of the APIs with syntax and description:

NetSim API's to interact with MATLAB	Description
netsim_matlab_interface_configure(char* appPath)	Starts a MATLAB engine process and adds the appPath to the top of the search path for current MATLAB session.
netsim_matlab_interface_close();	Sends exit command to MATLAB to terminate the session.

netsim_matlab_send_ascii_command(char* format, ...)	Sends commands, variables and values as string to MATLAB.
netsim_matlab_get_value(char* out, int outLen, char* name, char* type)	Gets the value of a MATLAB variable as a string. Currently supports double data type only.

Table 10-2: MATLAB Engine API functions

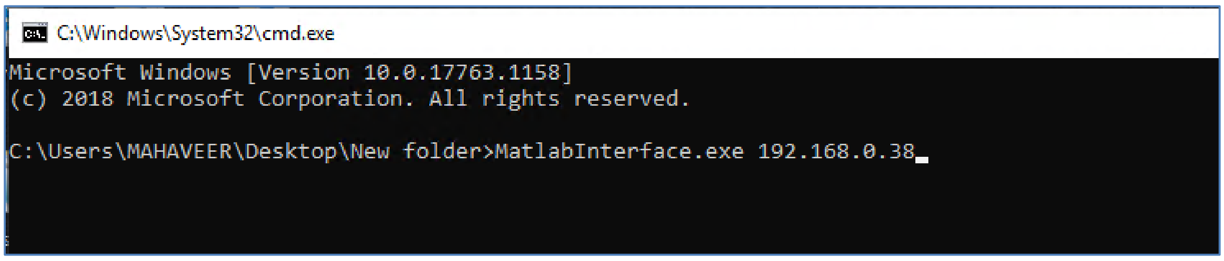
These functions are defined as part of the *NetSim_utility.h* file which is part of the Include directory of NetSim Source Codes. This header can be included in the C files where these APIs are to be called from.

10.3.1.1 Prerequisites for MATLAB Socket Interfacing

- An installed version of MATLAB R2021(a) or lower version in the same system where NetSim is installed.

Note:

- *If the MATLAB is installed in different System in the same Network. Copy MATLAB_Interface.exe present in bin folder from installed directory (C:\Program Files\NetSim\Standard_v13_1\bin) to the system where MATLAB is installed.*
- *Execute the below command in the command prompt where MATLAB_Interface.exe is placed followed by IP address of the system where the NetSim is installed.*



```
C:\Windows\System32\cmd.exe
Microsoft Windows [Version 10.0.17763.1158]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Users\MAHAVEER\Desktop\New folder>MatlabInterface.exe 192.168.0.38
```

Figure 10-2: Command to start the MATLAB interface

- Registration of MATLAB as a COM server by one of the following methods
  - Start Command Prompt as administrator and execute the following command:

matlab -regserver

Note: *If you have multiple versions of MATLAB installed on your computer, the best practice is to run the matlab command from the matlabroot folder.*

- Enter the following command in the command line of MATLAB version that you want to interface with NetSim:

matlab -regserver

10.3.1.2 Implement Weibull Distribution of MATLAB without using .m file

In this example we will replace the default Rayleigh Fading (part of the path loss calculation) used in NetSim, with a Fading Power calculated using the Weibull Distribution from MATLAB Socket Interfacing.

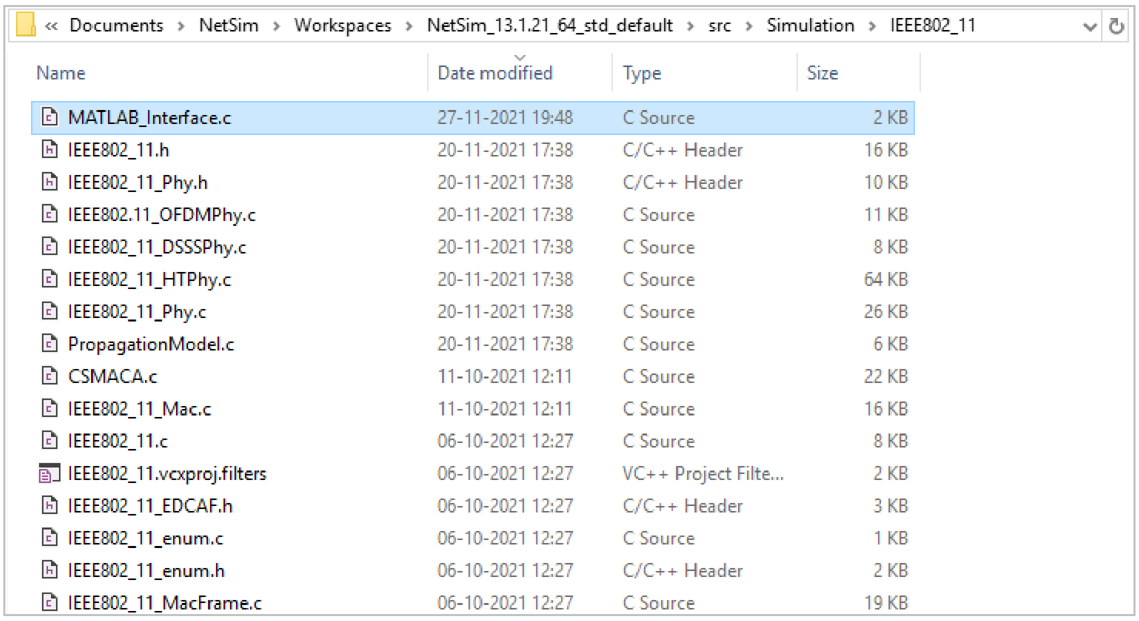
"This example uses 64bit version of NetSim setup with MATLAB Engine"

Procedure

1. Create a MATLAB_Interface.c file inside the IEEE802_11 folder which can be found in the current workspace location of NetSim that you are running, and it would be something like "C:\Users\PC\Documents\NetSim\Workspaces\<Yourdefaultworkspace>\src\Simulation\IEEE 802_11". Write the following code inside MATLAB_Interface.c file.

```
/*
*
* This is a simple program that illustrates how to call the MATLAB
* Engine functions from NetSim C Code.
*
*/
#include<windows.h>
#include<stdlib.h>
#include<stdio.h>
#include<string.h>
#include "main.h"
#include "Stack.h"
#include"NetSim_utility.h"
#include "direct.h"

char buf[BUFSIZ];
double  status;
double* result;
double fn_netsim_matlab_init()
{
        netsim_matlab_interface_configure("");
        return 0;
}
double fn_netsim_matlab_run()
```

```
{

//write your own implementation here

        int weibull_noncentrality = 1, weibull_scale = 2;

        //use ProbDistUnivParam() function for matlab 2016 or lower

        //sprintf_s(buf, BUFSIZ, "h=ProbDistUnivParam('weibull',[%d %d])",

        //weibull_noncentrality, weibull_scale);//

        //use makedist() function for matlab 2017 or Higher

        sprintf_s(buf,BUFSIZ,"h=makedist('weibull',%d,%d)", weibull_noncentrality,

        weibull_scale);

        netsim_matlab_send_ascii_command(buf);

        sprintf_s(buf,BUFSIZ,"i=random(h)");//


        netsim_matlab_send_ascii_command(buf);

        netsim_matlab_get_value(buf,BUFSIZ,"i","double");

        double result = atof(buf);

        return result;

}

double fn_netsim_matlab_finish()

{

fprintf(stderr, "\nPress any key to close MATLAB\n");

_getch();

netsim_matlab_interface_close();

return 0;

}
```

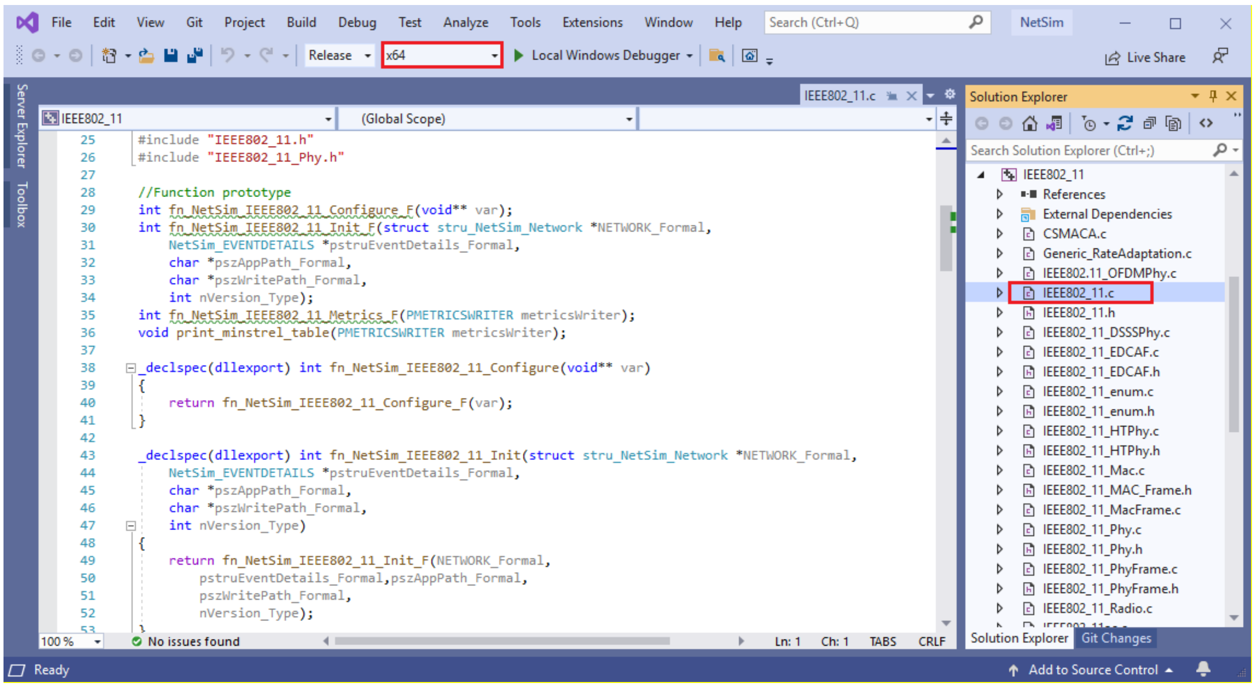


« Documents › NetSim › Workspaces › NetSim_13.1.21_64_std_default › src › Simulation › IEEE802_11

Name	Date modified	Type	Size
MATLAB_Interface.c	27-11-2021 19:48	C Source	2 KB
IEEE802_11.h	20-11-2021 17:38	C/C++ Header	16 KB
IEEE802_11_Phy.h	20-11-2021 17:38	C/C++ Header	10 KB
IEEE802.11_OFDMPhy.c	20-11-2021 17:38	C Source	11 KB
IEEE802_11_DSSSPhy.c	20-11-2021 17:38	C Source	8 KB
IEEE802_11_HTPhy.c	20-11-2021 17:38	C Source	64 KB
IEEE802_11_Phy.c	20-11-2021 17:38	C Source	26 KB
PropagationModel.c	20-11-2021 17:38	C Source	6 KB
CSMACA.c	11-10-2021 12:11	C Source	22 KB
IEEE802_11_Mac.c	11-10-2021 12:11	C Source	16 KB
IEEE802_11.c	06-10-2021 12:27	C Source	8 KB
IEEE802_11.vcxproj.filters	06-10-2021 12:27	VC++ Project Filte...	2 KB
IEEE802_11_EDCAF.h	06-10-2021 12:27	C/C++ Header	3 KB
IEEE802_11_enum.c	06-10-2021 12:27	C Source	1 KB
IEEE802_11_enum.h	06-10-2021 12:27	C/C++ Header	2 KB
IEEE802_11_MacFrame.c	06-10-2021 12:27	C Source	19 KB

Figure 10-3: Create a MATLAB_Interface.c inside the IEEE802_11 folder

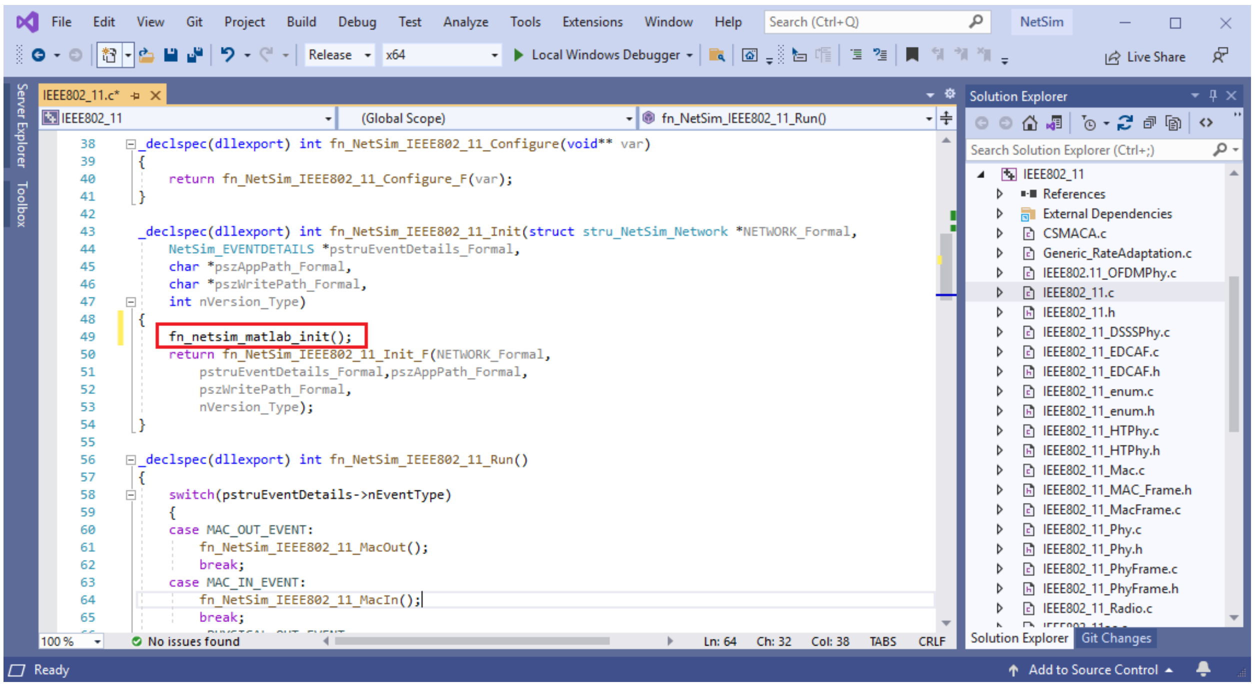
2. Now open the code and configure Visual studio Software Platform to x64.
3. Right click on “IEEE802_11 Project” present in “Solution Explorer” window and select Add > Existing Item > select the MATLAB_Interface.c
4. MATLAB_Interface.c file contains the following API functions.
   I. **netsim_matlab_interface_configure()** under **fn_netsim_matlab_init()** - waits to connect MATLAB after running MATLAB interface.exe i.e “Open MATLAB Interface” option in the NetSim UI.
   II. **netsim_matlab_send_ascii_command() and netsim_matlab_get_value()** inside **fn_netsim_matlab_run()** -relevant variables in NetSim to be passed as input to MATLAB and vice versa.
   III. **netsim_matlab_interface_close();** inside **fn_netsim_matlab_finish()** - Send the quit command to MATLAB to terminate the connection.
5. In the Solution Explorer under **IEEE802_11** project double click on the **IEEE802_11.c** file.



```
#include "IEEE802_11.h"
#include "IEEE802_11_Phy.h"

//Function prototype
int fn_NetSim_IEEE802_11_Configure_F(void** var);
int fn_NetSim_IEEE802_11_Init_F(struct stru_NetSim_Network *NETWORK_Formal,
    NetSim_EVENTDETAILS *pstruEventDetails_Formal,
    char *pszAppPath_Formal,
    char *pszWritePath_Formal,
    int nVersion_Type);
int fn_NetSim_IEEE802_11_Metrics_F(PMETRICSWRITER metricsWriter);
void print_minstrel_table(PMETRICSWRITER metricsWriter);

_declspec(dllexport) int fn_NetSim_IEEE802_11_Configure(void** var)
{
    return fn_NetSim_IEEE802_11_Configure_F(var);
}

_declspec(dllexport) int fn_NetSim_IEEE802_11_Init(struct stru_NetSim_Network *NETWORK_Formal,
    NetSim_EVENTDETAILS *pstruEventDetails_Formal,
    char *pszAppPath_Formal,
    char *pszWritePath_Formal,
    int nVersion_Type)
{
    return fn_NetSim_IEEE802_11_Init_F(NETWORK_Formal,
        pstruEventDetails_Formal,pszAppPath_Formal,
        pszWritePath_Formal,
        nVersion_Type);
```

Figure 10-4: Solution Explorer double click on the IEEE802_11.c file

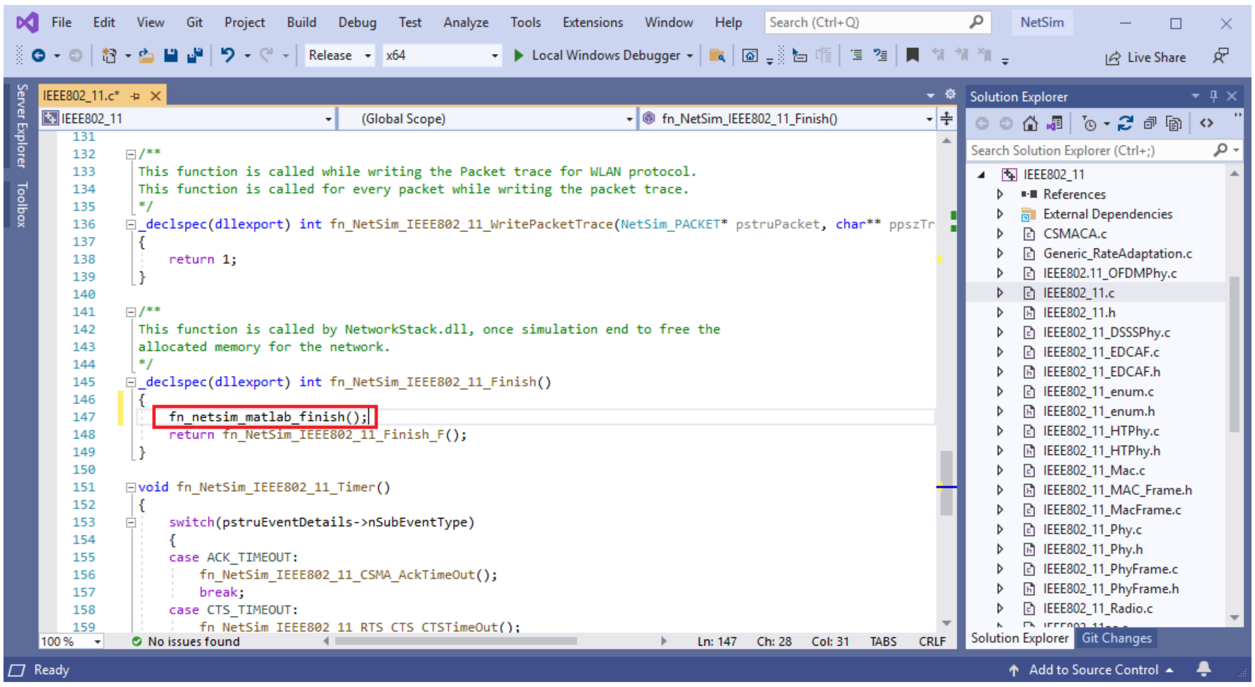
6. Add a call to **fn_netsim_matlab_init();** inside the fn_NetSim_IEEE802_11_Init() function.



```
_declspec(dllexport) int fn_NetSim_IEEE802_11_Configure(void** var)
{
    return fn_NetSim_IEEE802_11_Configure_F(var);
}

_declspec(dllexport) int fn_NetSim_IEEE802_11_Init(struct stru_NetSim_Network *NETWORK_Formal,
    NetSim_EVENTDETAILS *pstruEventDetails_Formal,
    char *pszAppPath_Formal,
    char *pszWritePath_Formal,
    int nVersion_Type)
{
    fn_netsim_matlab_init();
    return fn_NetSim_IEEE802_11_Init_F(NETWORK_Formal,
        pstruEventDetails_Formal,pszAppPath_Formal,
        pszWritePath_Formal,
        nVersion_Type);
}

_declspec(dllexport) int fn_NetSim_IEEE802_11_Run()
{
    switch(pstruEventDetails->nEventType)
    {
    case MAC_OUT_EVENT:
        fn_NetSim_IEEE802_11_MacOut();
        break;
    case MAC_IN_EVENT:
        fn_NetSim_IEEE802_11_MacIn();
        break;
```

Figure 10-5: Added a fn_netsim_matlab_init(); inside the fn_NetSim_IEEE802_11_Init() function

7. Similarly add a call to **fn_netsim_matlab_finish();** inside the **fn_NetSim_IEEE802_11_Finish()** function.



```
/**
This function is called while writing the Packet trace for WLAN protocol.
This function is called for every packet while writing the packet trace.
*/
_declspec(dllexport) int fn_NetSim_IEEE802_11_WritePacketTrace(NetSim_PACKET* pstruPacket, char** ppszTr
{
    return 1;
}

/**
This function is called by NetworkStack.dll, once simulation end to free the
allocated memory for the network.
*/
_declspec(dllexport) int fn_NetSim_IEEE802_11_Finish()
{
    fn_netsim_matlab_finish();
    return fn_NetSim_IEEE802_11_Finish_F();
}

void fn_NetSim_IEEE802_11_Timer()
{
    switch(pstruEventDetails->nSubEventType)
    {
    case ACK_TIMEOUT:
        fn_NetSim_IEEE802_11_CSMA_AckTimeOut();
        break;
    case CTS_TIMEOUT:
        fn_NetSim_IEEE802_11_RTS_CTS_CTSTimeOut();
```

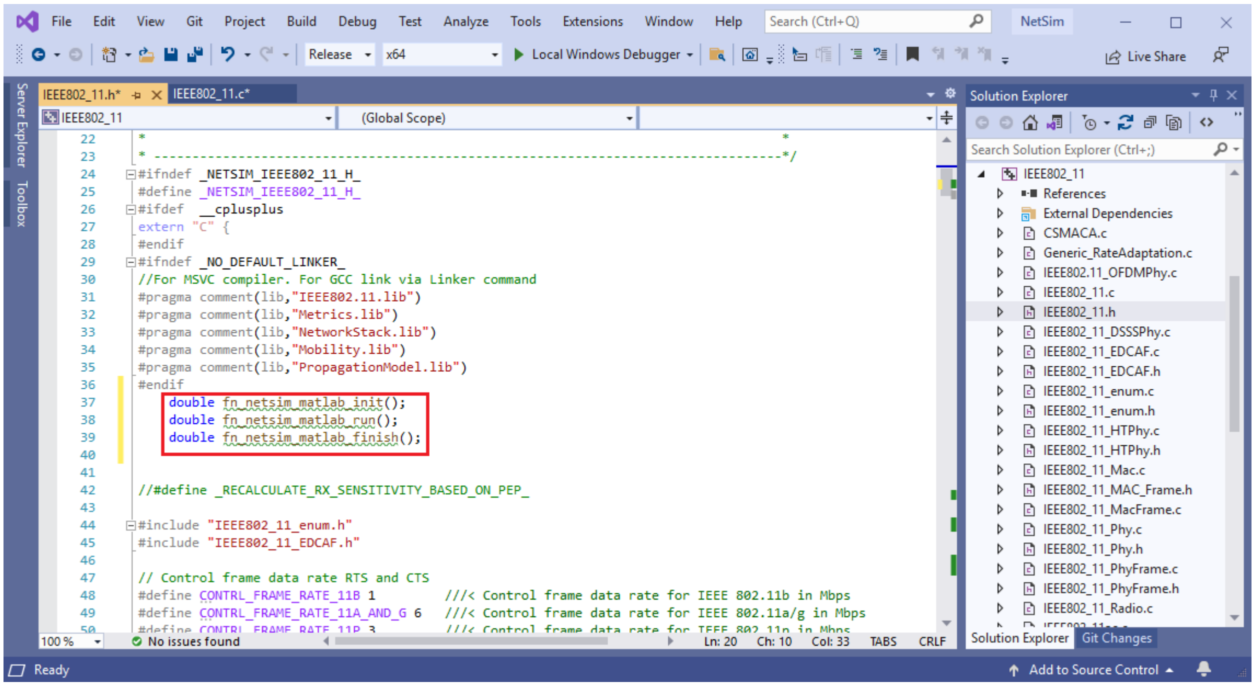
Figure 10-6: Added a fn_netsim_matlab_finish(); inside the fn_NetSim_IEEE802_11_Finish() function

8. In the Solution Explorer under IEEE802_11 project double click on the IEEE802_11.h file. Add definitions of the following functions.

   double fn_netsim_matlab_init();

   double fn_netsim_matlab_run();

   double fn_netsim_matlab_finish();



```
*                                                                          *
* -------------------------------------------------------------------------*/
#ifndef _NETSIM_IEEE802_11_H_
#define _NETSIM_IEEE802_11_H_
#ifdef  __cplusplus
extern "C" {
#endif
#ifndef _NO_DEFAULT_LINKER_
//For MSVC compiler. For GCC link via Linker command
#pragma comment(lib,"IEEE802.11.lib")
#pragma comment(lib,"Metrics.lib")
#pragma comment(lib,"NetworkStack.lib")
#pragma comment(lib,"Mobility.lib")
#pragma comment(lib,"PropagationModel.lib")
#endif
    double fn_netsim_matlab_init();
    double fn_netsim_matlab_run();
    double fn_netsim_matlab_finish();


//#define _RECALCULATE_RX_SENSITIVITY_BASED_ON_PEP_

#include "IEEE802_11_enum.h"
#include "IEEE802_11_EDCAF.h"

// Control frame data rate RTS and CTS
#define CONTRL_FRAME_RATE_11B 1        ///< Control frame data rate for IEEE 802.11b in Mbps
#define CONTRL_FRAME_RATE_11A_AND_G 6  ///< Control frame data rate for IEEE 802.11a/g in Mbps
```

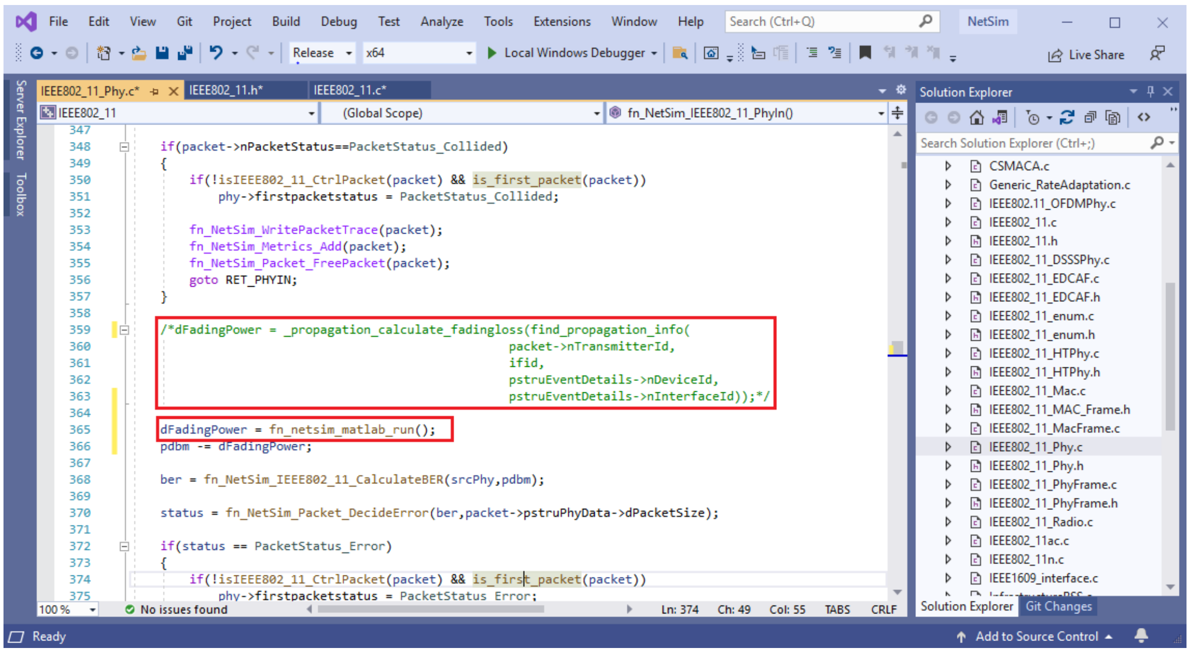
Figure 10-7: Added Matlab definitions in IEEE802_11.h file

9. In the Solution Explorer under IEEE802_11 project double click on the IEE802_11_PHY.c file.
10. Inside **fn_Netsim_IEEE802_11_PHYIn ()** function comment the lines,

dFadingPower = propagation_calculate_fadingloss(propagationHandle,packet->nTransmitterId,ifid,pstruEventDetails->nDeviceId, pstruEventDetails->nInterfaceId);

11. Make a call to the **fn_netsim_matlab_run()** function by adding the following line,

dFadingPower = fn_netsim_matlab_run();

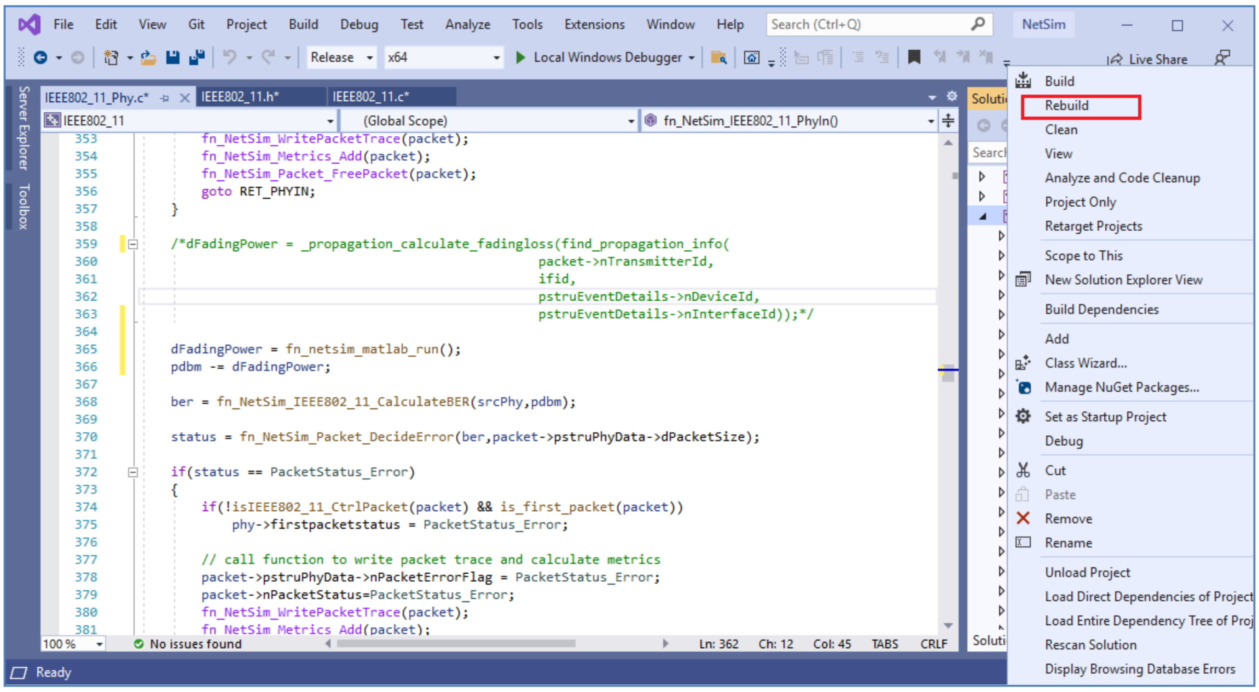


```
    if(packet->nPacketStatus==PacketStatus_Collided)
    {
        if(!isIEEE802_11_CtrlPacket(packet) && is_first_packet(packet))
            phy->firstpacketstatus = PacketStatus_Collided;

        fn_NetSim_WritePacketTrace(packet);
        fn_NetSim_Metrics_Add(packet);
        fn_NetSim_Packet_FreePacket(packet);
        goto RET_PHYIN;
    }

    /*dFadingPower = _propagation_calculate_fadingloss(find_propagation_info(
                                                        packet->nTransmitterId,
                                                        ifid,
                                                        pstruEventDetails->nDeviceId,
                                                        pstruEventDetails->nInterfaceId));*/

    dFadingPower = fn_netsim_matlab_run();
    pdbm -= dFadingPower;

    ber = fn_NetSim_IEEE802_11_CalculateBER(srcPhy,pdbm);

    status = fn_NetSim_Packet_DecideError(ber,packet->pstruPhyData->dPacketSize);

    if(status == PacketStatus_Error)
    {
        if(!isIEEE802_11_CtrlPacket(packet) && is_first_packet(packet))
            phy->firstpacketstatus = PacketStatus_Error;
```

Figure 10-8: Call to the fn_netsim_matlab_run() function

Note: *In MATLAB Socket Interfacing, compilation of MATLAB Engine is not required since the MATLAB APIs are already called in function.*

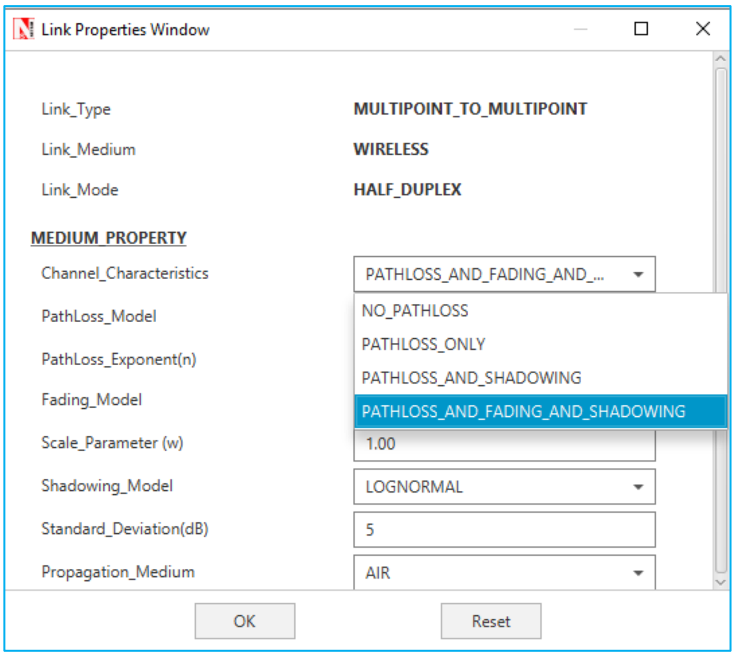
12. Now Right Click on IEEE802_11 project and select Rebuild.



```
        fn_NetSim_WritePacketTrace(packet);
        fn_NetSim_Metrics_Add(packet);
        fn_NetSim_Packet_FreePacket(packet);
        goto RET_PHYIN;
    }

    /*dFadingPower = _propagation_calculate_fadingloss(find_propagation_info(
                                                        packet->nTransmitterId,
                                                        ifid,
                                                        pstruEventDetails->nDeviceId,
                                                        pstruEventDetails->nInterfaceId));*/

    dFadingPower = fn_netsim_matlab_run();
    pdbm -= dFadingPower;

    ber = fn_NetSim_IEEE802_11_CalculateBER(srcPhy,pdbm);

    status = fn_NetSim_Packet_DecideError(ber,packet->pstruPhyData->dPacketSize);

    if(status == PacketStatus_Error)
    {
        if(!isIEEE802_11_CtrlPacket(packet) && is_first_packet(packet))
            phy->firstpacketstatus = PacketStatus_Error;

        // call function to write packet trace and calculate metrics
        packet->pstruPhyData->nPacketErrorFlag = PacketStatus_Error;
        packet->nPacketStatus=PacketStatus_Error;
        fn_NetSim_WritePacketTrace(packet);
        fn_NetSim_Metrics_Add(packet);
```

Figure 10-9: Rebuild IEEE802_11 project

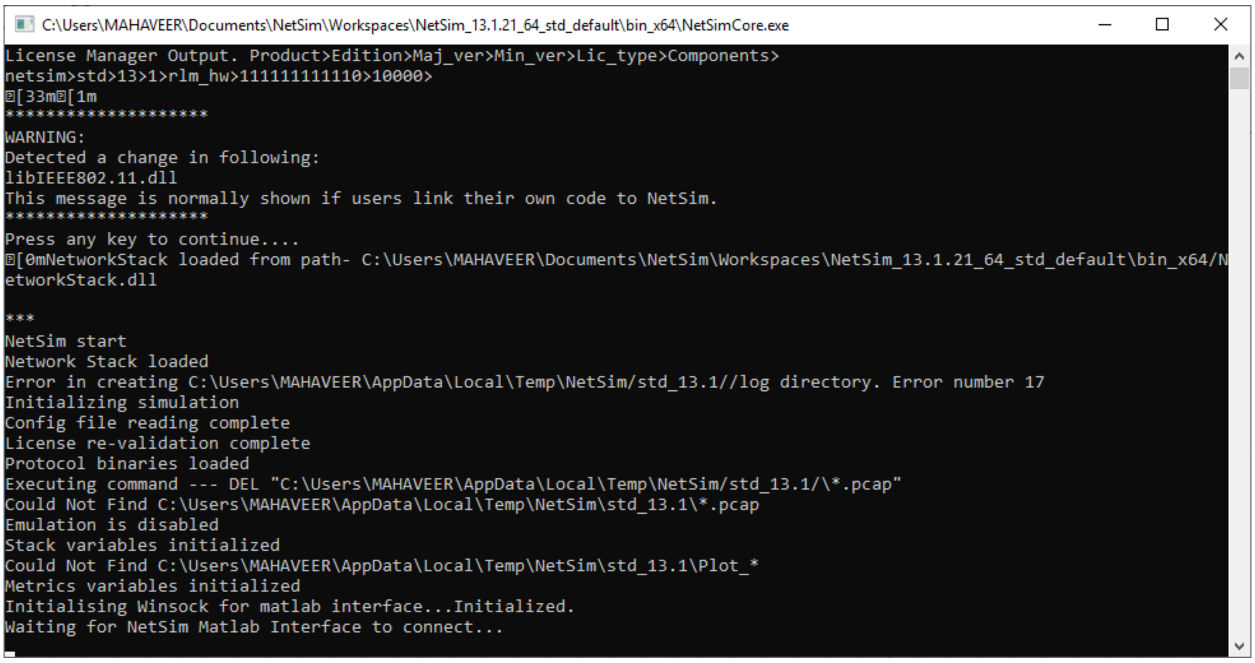
13. A new libIEEE802.11.dll gets created in the binx64 folder of NetSim's current workspace path C:\Users\PC\Documents\NetSim\Workspaces\<Your default workspace>\bin_x64> for 64-bit. For more information, follow steps provided in Section 9.1 "Writing your own code".
14. Run NetSim in Administrative mode. Create a Network scenario involving IEEE802_11 say MANET, right click on the Adhoc link and select properties. Make sure that the Channel Characteristics is set to PathLoss and Fading and Shadowing. Perform the simulation.



Link_Type	MULTIPOINT_TO_MULTIPOINT
Link_Medium	WIRELESS
Link_Mode	HALF_DUPLEX
MEDIUM_PROPERTY	
Channel_Characteristics	PATHLOSS_AND_FADING_AND_...
PathLoss_Model	
PathLoss_Exponent(n)	
Fading_Model	
Scale_Parameter (w)	1.00
Shadowing_Model	LOGNORMAL
Standard_Deviation(dB)	5
Propagation_Medium	AIR

Figure 10-10: Wireless Link Properties Window in MANET

15. NetSim Simulation console waits for MATLAB Interface process to connect.



```
C:\Users\MAHAVEER\Documents\NetSim\Workspaces\NetSim_13.1.21_64_std_default\bin_x64\NetSimCore.exe
License Manager Output. Product>Edition>Maj_ver>Min_ver>Lic_type>Components>
netsim>std>13>1>rlm_hw>111111111110>10000>
[33m[1m
********************
WARNING:
Detected a change in following:
libIEEE802.11.dll
This message is normally shown if users link their own code to NetSim.
********************
Press any key to continue....
[0mNetworkStack loaded from path- C:\Users\MAHAVEER\Documents\NetSim\Workspaces\NetSim_13.1.21_64_std_default\bin_x64/N
etworkStack.dll

***
NetSim start
Network Stack loaded
Error in creating C:\Users\MAHAVEER\AppData\Local\Temp\NetSim/std_13.1//log directory. Error number 17
Initializing simulation
Config file reading complete
License re-validation complete
Protocol binaries loaded
Executing command --- DEL "C:\Users\MAHAVEER\AppData\Local\Temp\NetSim/std_13.1/\*.pcap"
Could Not Find C:\Users\MAHAVEER\AppData\Local\Temp\NetSim\std_13.1\*.pcap
Emulation is disabled
Stack variables initialized
Could Not Find C:\Users\MAHAVEER\AppData\Local\Temp\NetSim\std_13.1\Plot_*
Metrics variables initialized
Initialising Winsock for matlab interface...Initialized.
Waiting for NetSim Matlab Interface to connect...
```

Figure 10-11: NetSim Simulation console waits for MATLAB Interface process to connect

16. If MATLAB is installed in the same system where NetSim is installed. MATLAB Interface process can be initiated directly from the design window of NetSim.

- Go to “Options” in design window and select the Open MATLAB Interface option as shown below:

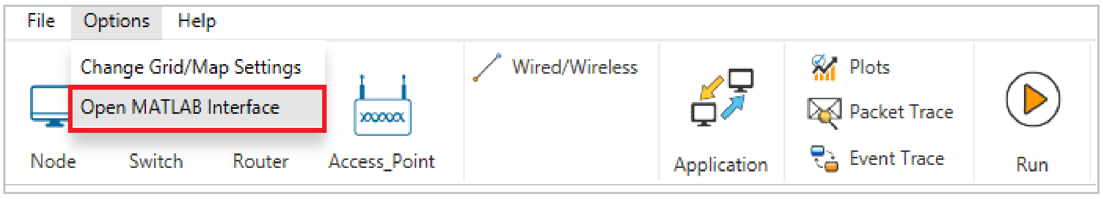
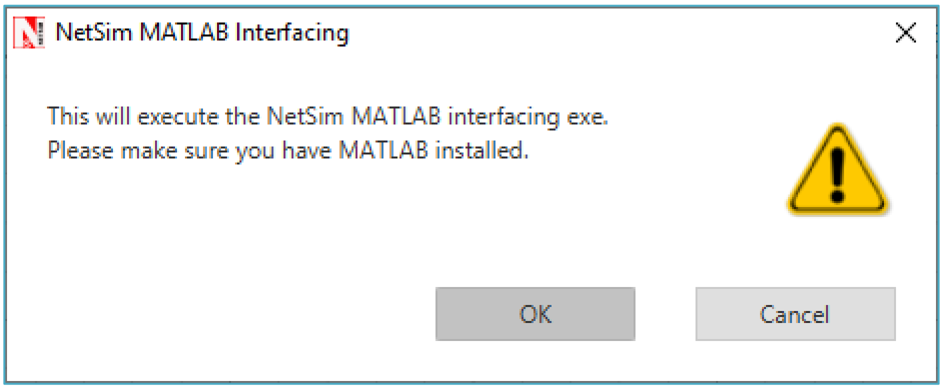


Figure 10-12: Option to open MATLAB Interface

- Click on the OK button when the below window is displayed. This opens the MATLAB Session.

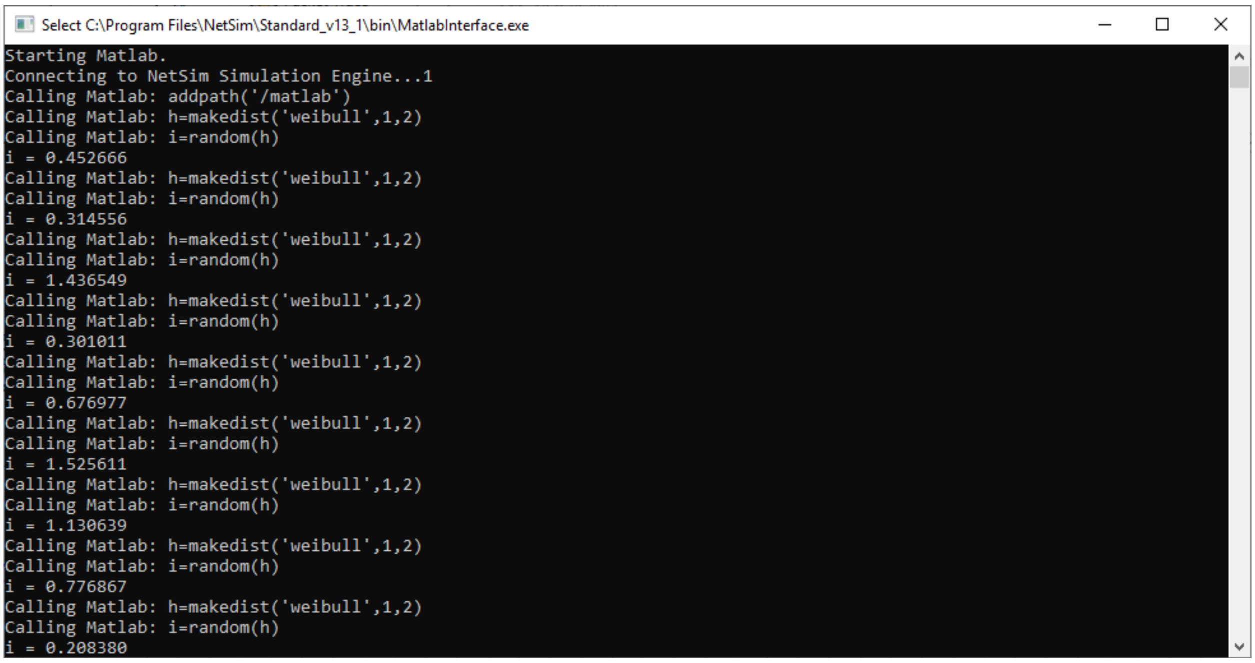


NetSim MATLAB Interfacing

This will execute the NetSim MATLAB interfacing exe.
Please make sure you have MATLAB installed.

Figure 10-13: NetSim MATLAB Interfacing warning message

17. After clicking on “OK”, MATLAB interface will get initiated. MATLAB Command window will open after which the simulation NetSim starts to simulate. During the simulation communication between MATLAB and NetSim will established.
18. The dFading power value will be displayed in the MATLAB Console window as below. At simulation end the MATLAB Interface process will get terminated.



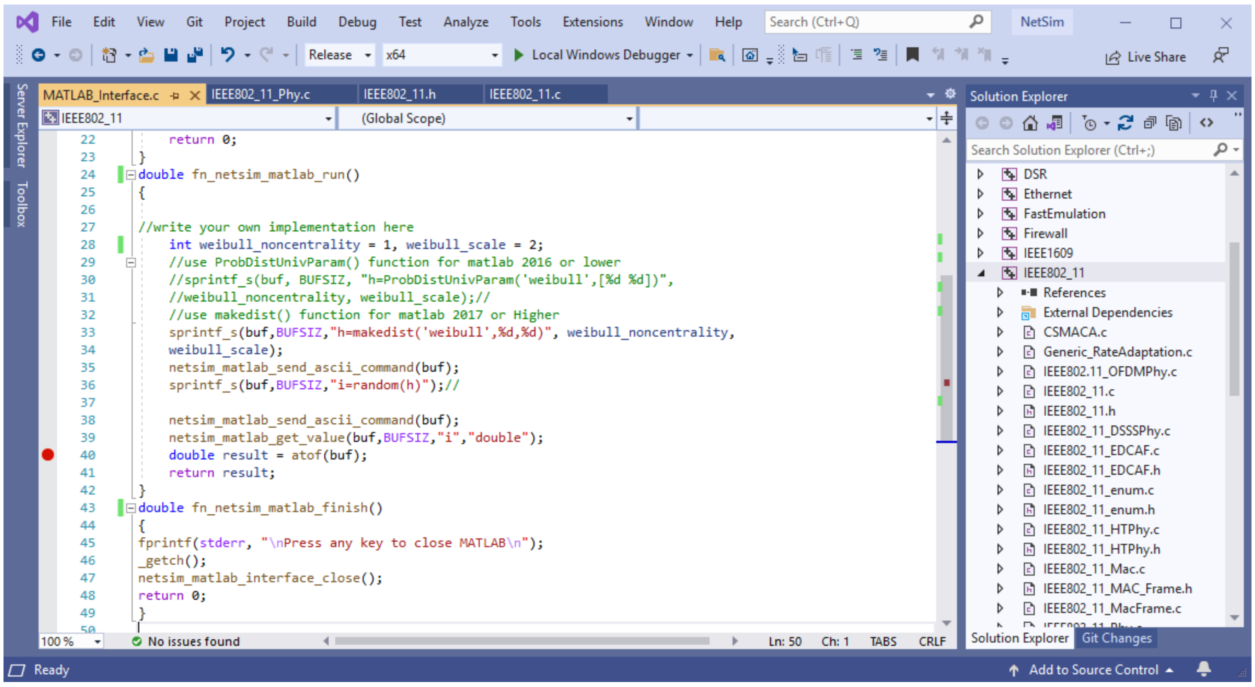
Select C:\Program Files\NetSim\Standard_v13_1\bin\MatlabInterface.exe

```
Starting Matlab.
Connecting to NetSim Simulation Engine...1
Calling Matlab: addpath('/matlab')
Calling Matlab: h=makedist('weibull',1,2)
Calling Matlab: i=random(h)
i = 0.452666
Calling Matlab: h=makedist('weibull',1,2)
Calling Matlab: i=random(h)
i = 0.314556
Calling Matlab: h=makedist('weibull',1,2)
Calling Matlab: i=random(h)
i = 1.436549
Calling Matlab: h=makedist('weibull',1,2)
Calling Matlab: i=random(h)
i = 0.301011
Calling Matlab: h=makedist('weibull',1,2)
Calling Matlab: i=random(h)
i = 0.676977
Calling Matlab: h=makedist('weibull',1,2)
Calling Matlab: i=random(h)
i = 1.525611
Calling Matlab: h=makedist('weibull',1,2)
Calling Matlab: i=random(h)
i = 1.130639
Calling Matlab: h=makedist('weibull',1,2)
Calling Matlab: i=random(h)
i = 0.776867
Calling Matlab: h=makedist('weibull',1,2)
Calling Matlab: i=random(h)
i = 0.208380
```

Figure 10-14: Runtime MATLAB interfacing window with I value

10.3.1.3 Debug and understand communication between NetSim and MATLAB

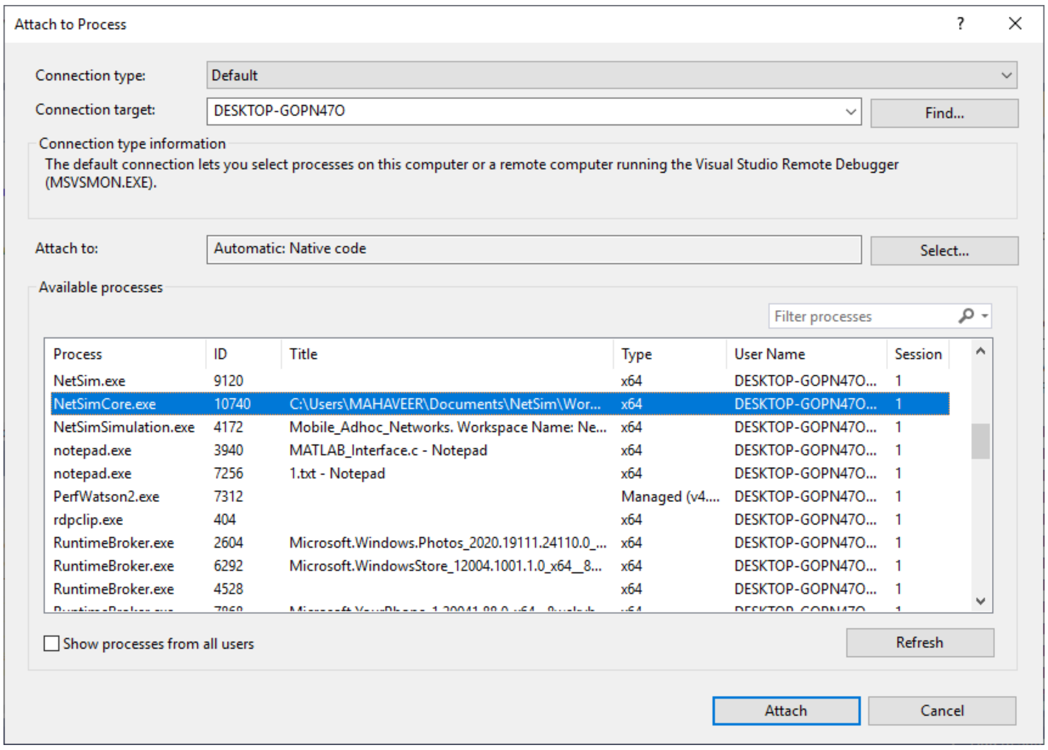
1. In the Solution Explorer under IEEE802_11 project double click on MATLAB_Interface.c file and place a breakpoint inside the **fn_netsim_matlab_run()** function before the return statement.



```
	return 0;
}
double fn_netsim_matlab_run()
{

//write your own implementation here
	int weibull_noncentrality = 1, weibull_scale = 2;
	//use ProbDistUnivParam() function for matlab 2016 or lower
	//sprintf_s(buf, BUFSIZ, "h=ProbDistUnivParam('weibull',[%d %d])",
	//weibull_noncentrality, weibull_scale);//
	//use makedist() function for matlab 2017 or Higher
	sprintf_s(buf,BUFSIZ,"h=makedist('weibull',%d,%d)", weibull_noncentrality,
	weibull_scale);
	netsim_matlab_send_ascii_command(buf);
	sprintf_s(buf,BUFSIZ,"i=random(h)");//

	netsim_matlab_send_ascii_command(buf);
	netsim_matlab_get_value(buf,BUFSIZ,"i","double");
	double result = atof(buf);
	return result;
}
double fn_netsim_matlab_finish()
{
fprintf(stderr, "\nPress any key to close MATLAB\n");
_getch();
netsim_matlab_interface_close();
return 0;
}
```

Figure 10-15: Place a breakpoint inside the fn_netsim_matlab_run() function

2. Rebuild the code.
3. Now run the NetSim Scenario. The simulation window stops for user interrupt.
4. In Visual studio, go to Debug =>Attach to Process.
5. From the list of Processes select NetSimCore.exe and click on Attach.



Attach to Process

Connection type: Default

Connection target: DESKTOP-GOPN47O

Connection type information
The default connection lets you select processes on this computer or a remote computer running the Visual Studio Remote Debugger (MSVSMON.EXE).

Attach to: Automatic: Native code

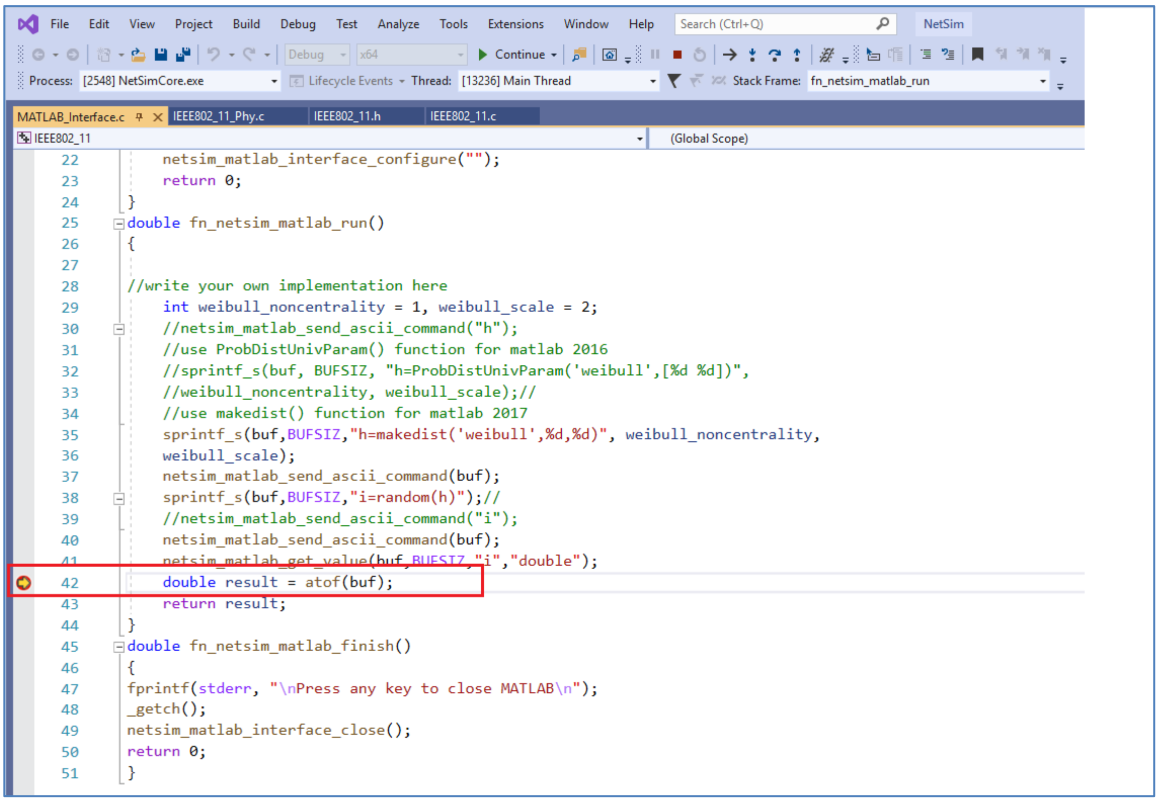
Available processes

Process	ID	Title	Type	User Name	Session
NetSim.exe	9120		x64	DESKTOP-GOPN47O...	1
NetSimCore.exe	10740	C:\Users\MAHAVEER\Documents\NetSim\Wor...	x64	DESKTOP-GOPN47O...	1
NetSimSimulation.exe	4172	Mobile_Adhoc_Networks. Workspace Name: Ne...	x64	DESKTOP-GOPN47O...	1
notepad.exe	3940	MATLAB_Interface.c - Notepad	x64	DESKTOP-GOPN47O...	1
notepad.exe	7256	1.txt - Notepad	x64	DESKTOP-GOPN47O...	1
PerfWatson2.exe	7312		Managed (v4....	DESKTOP-GOPN47O...	1
rdpclip.exe	404		x64	DESKTOP-GOPN47O...	1
RuntimeBroker.exe	2604	Microsoft.Windows.Photos_2020.19111.24110.0_...	x64	DESKTOP-GOPN47O...	1
RuntimeBroker.exe	6292	Microsoft.WindowsStore_12004.1001.1.0_x64__8...	x64	DESKTOP-GOPN47O...	1
RuntimeBroker.exe	4528		x64	DESKTOP-GOPN47O...	1

☐ Show processes from all users

Figure 10-16: Select NetSimCore.exe in Attach to Process Window

6. Now go to the Simulation window and press Enter. Simulation will pause at Waiting for matlab interface to connect, go to Design window/GUI -> Options -> Open MATLAB interface ->click on OK
7. MATLAB Command Window will start and breakpoint in Visual Studio gets triggered.

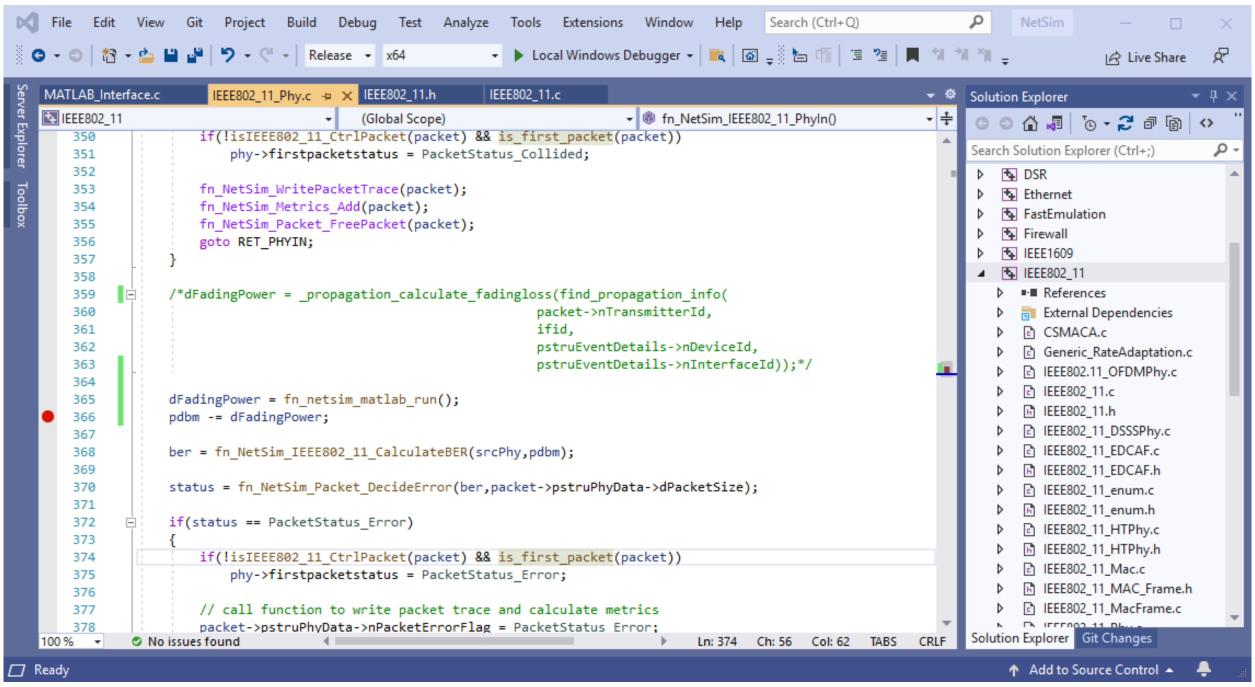


File Edit View Project Build Debug Test Analyze Tools Extensions Window Help
Search (Ctrl+Q)
NetSim
Debug
x64
Continue
Process: [2548] NetSimCore.exe
Lifecycle Events
Thread: [13236] Main Thread
Stack Frame: fn_netsim_matlab_run
MATLAB_Interface.c
IEEE802_11_Phy.c
IEEE802_11.h
IEEE802_11.c
IEEE802_11
(Global Scope)

```
    netsim_matlab_interface_configure("");
    return 0;
}
double fn_netsim_matlab_run()
{

//write your own implementation here
    int weibull_noncentrality = 1, weibull_scale = 2;
    //netsim_matlab_send_ascii_command("h");
    //use ProbDistUnivParam() function for matlab 2016
    //sprintf_s(buf, BUFSIZ, "h=ProbDistUnivParam('weibull',[%d %d])",
    //weibull_noncentrality, weibull_scale);//
    //use makedist() function for matlab 2017
    sprintf_s(buf,BUFSIZ,"h=makedist('weibull',%d,%d)", weibull_noncentrality,
    weibull_scale);
    netsim_matlab_send_ascii_command(buf);
    sprintf_s(buf,BUFSIZ,"i=random(h)");//
    //netsim_matlab_send_ascii_command("i");
    netsim_matlab_send_ascii_command(buf);
    netsim_matlab_get_value(buf,BUFSIZ,"i","double");
    double result = atof(buf);
    return result;
}
double fn_netsim_matlab_finish()
{
fprintf(stderr, "\nPress any key to close MATLAB\n");
_getch();
netsim_matlab_interface_close();
return 0;
}
```

Figure 10-17: Breakpoint gets triggered in Visual Studio.

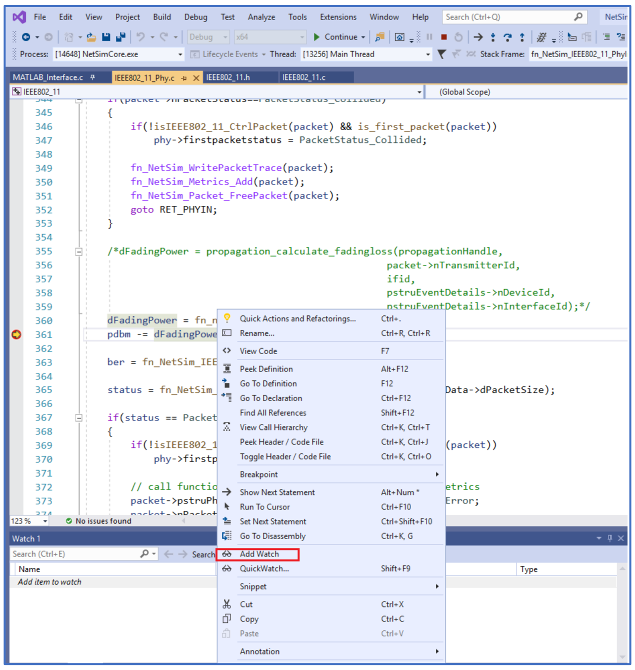
8. Now place another breakpoint after the line dFadingPower = fn_netsim_matlab_run()



```
        if(!isIEEE802_11_CtrlPacket(packet) && is_first_packet(packet))
            phy->firstpacketstatus = PacketStatus_Collided;

        fn_NetSim_WritePacketTrace(packet);
        fn_NetSim_Metrics_Add(packet);
        fn_NetSim_Packet_FreePacket(packet);
        goto RET_PHYIN;
    }

    /*dFadingPower = _propagation_calculate_fadingloss(find_propagation_info(
                                            packet->nTransmitterId,
                                            ifid,
                                            pstruEventDetails->nDeviceId,
                                            pstruEventDetails->nInterfaceId));*/

    dFadingPower = fn_netsim_matlab_run();
    pdbm -= dFadingPower;

    ber = fn_NetSim_IEEE802_11_CalculateBER(srcPhy,pdbm);

    status = fn_NetSim_Packet_DecideError(ber,packet->pstruPhyData->dPacketSize);

    if(status == PacketStatus_Error)
    {
        if(!isIEEE802_11_CtrlPacket(packet) && is_first_packet(packet))
            phy->firstpacketstatus = PacketStatus_Error;

        // call function to write packet trace and calculate metrics
        packet->pstruPhyData->nPacketErrorFlag = PacketStatus_Error;
```

Figure 10-18: Added Breakpoint under fn_netsim_matlab_run()

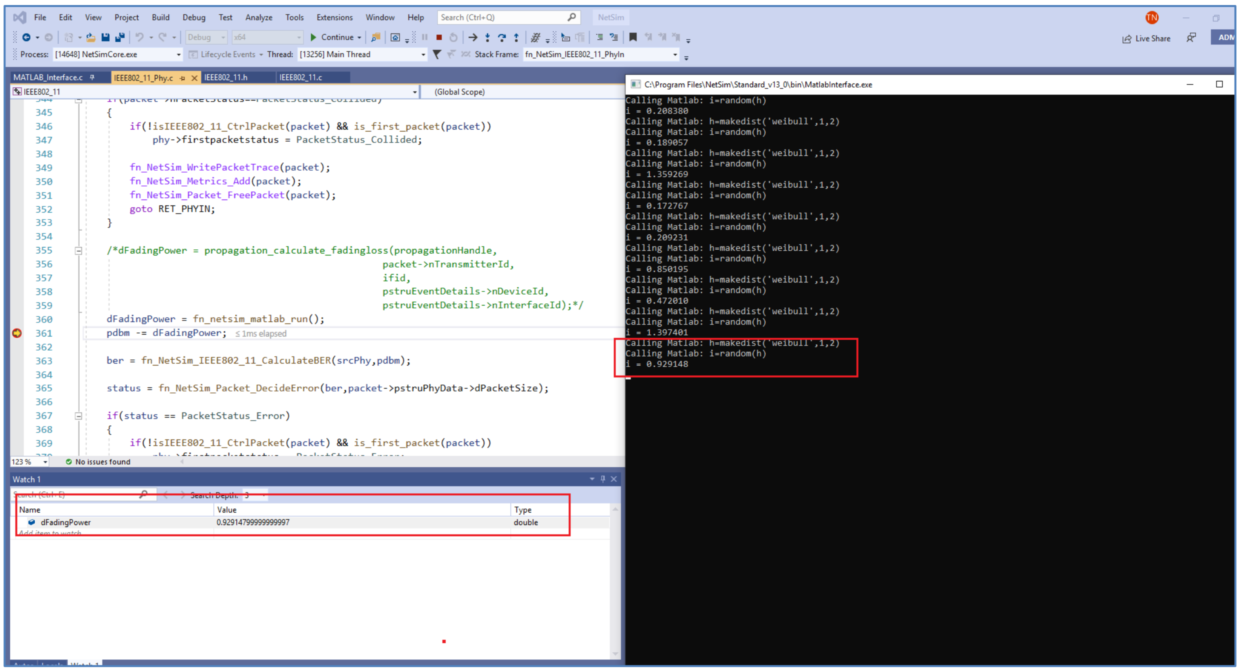
9. Add the variable dFadingPower in IEEE802_11_Phy.c file, to watch. For this, right click on the variable dFadingPower and select “Add Watch” option. You will find a watch window containing the variable name and its value in the bottom left corner.



```
    {
        if(!isIEEE802_11_CtrlPacket(packet) && is_first_packet(packet))
            phy->firstpacketstatus = PacketStatus_Collided;

        fn_NetSim_WritePacketTrace(packet);
        fn_NetSim_Metrics_Add(packet);
        fn_NetSim_Packet_FreePacket(packet);
        goto RET_PHYIN;
    }

    /*dFadingPower = propagation_calculate_fadingloss(propagationHandle,
                                            packet->nTransmitterId,
                                            ifid,
                                            pstruEventDetails->nDeviceId,
                                            pstruEventDetails->nInterfaceId);*/

    dFadingPower = fn_n
    pdbm -= dFadingPowe

    ber = fn_NetSim_IEE

    status = fn_NetSim_                          Data->dPacketSize);

    if(status == Packet
    {
        if(!isIEEE802_1                          (packet))
            phy->firstp

        // call functio                          etrics
        packet->pstruPh                          Error;
```

Figure 10-19: Right Click on dFadingPower and Select “Add Watch” option

10. Now when debugging (say by pressing F5 each time) you will find that the watch window displays the value of dFadingPower whenever the control reaches the recently set breakpoint. You will also find that the value of dFadingPower in the Visual Studio Watch window and the value of i in the MATLAB Console window appear to be similar.



```
    {
        if(!isIEEE802_11_CtrlPacket(packet) && is_first_packet(packet))
            phy->firstpacketstatus = PacketStatus_Collided;

        fn_NetSim_WritePacketTrace(packet);
        fn_NetSim_Metrics_Add(packet);
        fn_NetSim_Packet_FreePacket(packet);
        goto RET_PHYIN;
    }

    /*dFadingPower = propagation_calculate_fadingloss(propagationHandle,
                                                       packet->nTransmitterId,
                                                       ifid,
                                                       pstruEventDetails->nDeviceId,
                                                       pstruEventDetails->nInterfaceId);*/
    dFadingPower = fn_netsim_matlab_run();
    pdbm -= dFadingPower;  ≤ 1ms elapsed

    ber = fn_NetSim_IEEE802_11_CalculateBER(srcPhy,pdbm);

    status = fn_NetSim_Packet_DecideError(ber,packet->pstruPhyData->dPacketSize);

    if(status == PacketStatus_Error)
    {
        if(!isIEEE802_11_CtrlPacket(packet) && is_first_packet(packet))
```

Name	Value	Type
dFadingPower	0.92914799999999997	double

C:\Program Files\NetSim\Standard_v13_0\bin\MatlabInterface.exe

```
Calling Matlab: i=random(h)
i = 0.208380
Calling Matlab: h=makedist('weibull',1,2)
Calling Matlab: i=random(h)
i = 0.189057
Calling Matlab: h=makedist('weibull',1,2)
Calling Matlab: i=random(h)
i = 1.359269
Calling Matlab: h=makedist('weibull',1,2)
Calling Matlab: i=random(h)
i = 0.172767
Calling Matlab: h=makedist('weibull',1,2)
Calling Matlab: i=random(h)
i = 0.209231
Calling Matlab: h=makedist('weibull',1,2)
Calling Matlab: i=random(h)
i = 0.850195
Calling Matlab: h=makedist('weibull',1,2)
Calling Matlab: i=random(h)
i = 0.472010
Calling Matlab: h=makedist('weibull',1,2)
Calling Matlab: i=random(h)
i = 1.397401
Calling Matlab: h=makedist('weibull',1,2)
Calling Matlab: i=random(h)
i = 0.929148
```

Figure 10-20: The Visual Studio Watch window and the value of i in the Runtime MATLAB window are similar

10.3.1.4 Implement Weibull Distribution of MATLAB in NetSim using .m file

Procedure:

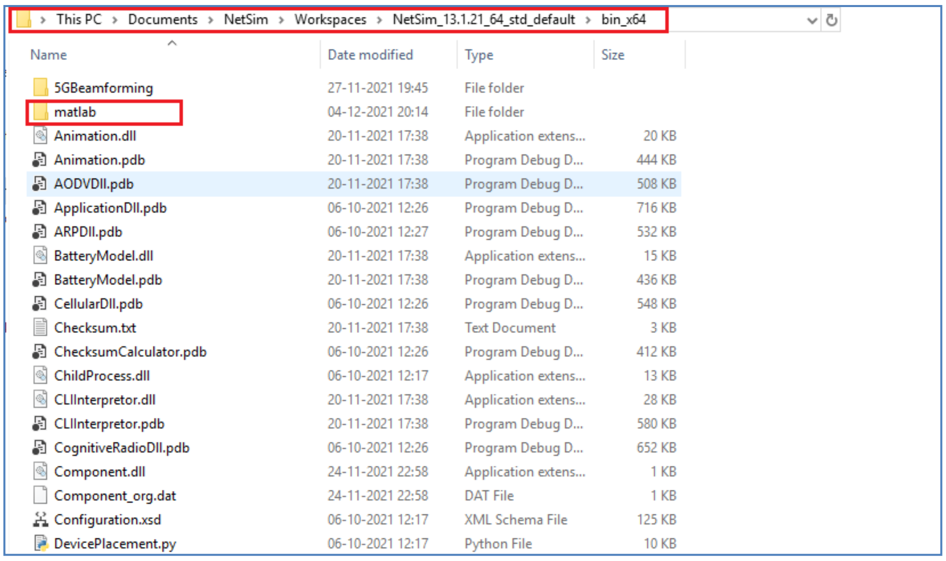
1. Create the weibull_distribution.m file inside <bin_x64 folder of where workspace is created>

C:\Users\PC\Documents\NetSim\Workspaces\<Your default workspace>\bin_x64 for 64 bit.

The weibull_distribution.m file contains the following code:

```
%use ProbDistUnivParam() function for matlab 2016
%h=ProbDistUnivParam('weibull',[noncentrality,scale]);
%use makedist() function for matlab 2017
h=makedist('weibull',noncentrality,scale);
 i=random(h,1);
WLAN=I;
```

- Create matlab folder inside working workspace bin_x64 folder

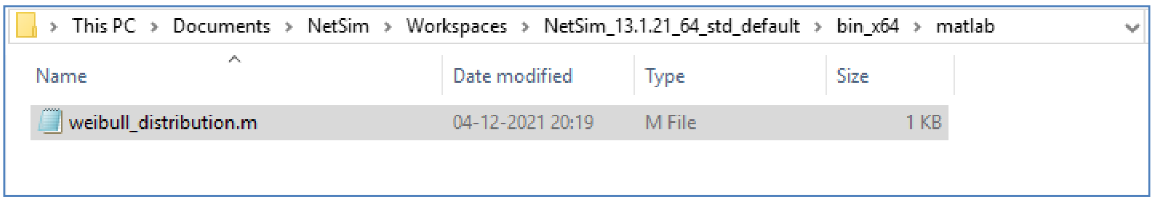


This PC > Documents > NetSim > Workspaces > NetSim_13.1.21_64_std_default > bin_x64

Name	Date modified	Type	Size
5GBeamforming	27-11-2021 19:45	File folder	
matlab	04-12-2021 20:14	File folder	
Animation.dll	20-11-2021 17:38	Application extens...	20 KB
Animation.pdb	20-11-2021 17:38	Program Debug D...	444 KB
AODVDll.pdb	20-11-2021 17:38	Program Debug D...	508 KB
ApplicationDll.pdb	06-10-2021 12:26	Program Debug D...	716 KB
ARPDll.pdb	06-10-2021 12:27	Program Debug D...	532 KB
BatteryModel.dll	20-11-2021 17:38	Application extens...	15 KB
BatteryModel.pdb	20-11-2021 17:38	Program Debug D...	436 KB
CellularDll.pdb	06-10-2021 12:26	Program Debug D...	548 KB
Checksum.txt	20-11-2021 17:38	Text Document	3 KB
ChecksumCalculator.pdb	06-10-2021 12:26	Program Debug D...	412 KB
ChildProcess.dll	06-10-2021 12:17	Application extens...	13 KB
CLIInterpretor.dll	20-11-2021 17:38	Application extens...	28 KB
CLIInterpretor.pdb	20-11-2021 17:38	Program Debug D...	580 KB
CognitiveRadioDll.pdb	06-10-2021 12:26	Program Debug D...	652 KB
Component.dll	24-11-2021 22:58	Application extens...	1 KB
Component_org.dat	24-11-2021 22:58	DAT File	1 KB
Configuration.xsd	06-10-2021 12:17	XML Schema File	125 KB
DevicePlacement.py	06-10-2021 12:17	Python File	10 KB

Figure 10-21: Bin folder of x64 of current workspace

- Place the weibull_distribution.m file inside matlab folder.



This PC > Documents > NetSim > Workspaces > NetSim_13.1.21_64_std_default > bin_x64 > matlab

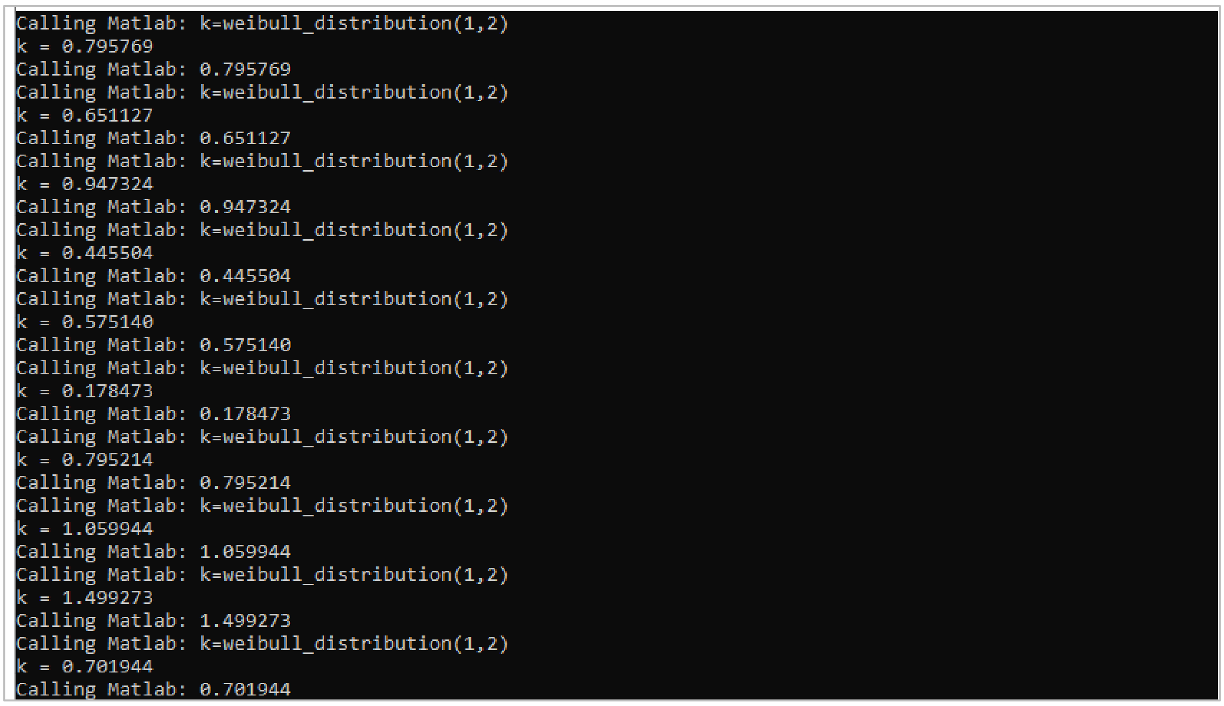
Name	Date modified	Type	Size
weibull_distribution.m	04-12-2021 20:19	M File	1 KB

Figure 10-22: Place Weibull_distribution.m file inside matlab folder

2. You will have to create a MATLAB_Interface.c file in the IEEE802_11 folder similar to the previous example.The functions **fn_netsim_matlab_init()** and **fn_netsim_matlab_finish()** will remain the same. Modify the function **fn_netsim_matlab_run()** that is part of MATLAB_Interface.c which was used in the previous example as shown below:

```
double fn_netsim_matlab_run()
{
        //write your own implementation here
        int weibull_noncentrality = 1, weibull_scale = 2;
        netsim_matlab_send_ascii_command(buf);
        sprintf_s(buf,BUFSIZ, "k=weibull_distribution(%d,%d)", weibull_noncentrality,
weibull_scale);
        netsim_matlab_send_ascii_command( buf);
        netsim_matlab_get_value(buf, BUFSIZ, "k", "double");
        double result = atof(buf);
        return result;
        }
```

3. Follow the similar steps 2 to 17 as explained in the section "Implement Weibull Distribution of MATLAB in NetSim without using .m file".
4. You will find that once the Simulation is run MATLAB Command Window starts to execute and dfading power value will be printed based on the weibull_distribution.m file.



```
Calling Matlab: k=weibull_distribution(1,2)
k = 0.795769
Calling Matlab: 0.795769
Calling Matlab: k=weibull_distribution(1,2)
k = 0.651127
Calling Matlab: 0.651127
Calling Matlab: k=weibull_distribution(1,2)
k = 0.947324
Calling Matlab: 0.947324
Calling Matlab: k=weibull_distribution(1,2)
k = 0.445504
Calling Matlab: 0.445504
Calling Matlab: k=weibull_distribution(1,2)
k = 0.575140
Calling Matlab: 0.575140
Calling Matlab: k=weibull_distribution(1,2)
k = 0.178473
Calling Matlab: 0.178473
Calling Matlab: k=weibull_distribution(1,2)
k = 0.795214
Calling Matlab: 0.795214
Calling Matlab: k=weibull_distribution(1,2)
k = 1.059944
Calling Matlab: 1.059944
Calling Matlab: k=weibull_distribution(1,2)
k = 1.499273
Calling Matlab: 1.499273
Calling Matlab: k=weibull_distribution(1,2)
k = 0.701944
Calling Matlab: 0.701944
```

Figure 10-23: Runtime MATLAB interfacing window with k value

10.3.2 NetSim-MATLAB COM Interface

NetSim-MATLAB COM Interface was introduced in NetSim v9. This allows runtime interaction between NetSim and MATLAB taking advantage of rich MATLAB Engine and matrix API's.

MATLAB functions can be called from NetSim's underlying protocol C source codes using MATLAB APIs. Following is some of the MATLAB Engine API functions that can be used from NetSim C source codes:

MATLAB Engine API Functions	Description
Engine	Type for MATLAB engine
engOpen	Start MATLAB engine session
engOpenSingleUse	Start MATLAB engine session for single, nonshared use
engClose	Quit MATLAB engine session
engEvalString	Evaluate expression in string
engGetVariable	Copy variable from MATLAB engine workspace
engPutVariable	Put variable into MATLAB engine workspace
engGetVisible	Determine visibility of MATLAB engine session
engSetVisible	Show or hide MATLAB engine session
engOutputBuffer	Specify buffer for MATLAB output

Table 10-3: MATLAB Engine API functions

In addition to these API's there are several C Matrix API's of MATLAB that can be used.

10.3.2.1 Pre-requisites for NetSim-MATLAB COM Interface

- MATLAB Interfacing requires an installed version of MATLAB. Engine API functions cannot be run on a machine that only has the MATLAB Runtime.
- Both NetSim and MATLAB should use the same build; 64-bit.
- Make sure that the following path is added to the environment variable *PATH*: *<Path where MATLAB is installed>\bin\win64*
- If the machine has more than one MATLAB installed, the directory for the target platform must be ahead of any other MATLAB directory (for instance, when compiling a 64-bit application, the directory in the MATLAB 64-bit installation must be the first one on the PATH).
- Registration of MATLAB as a COM server by one of the following methods
  - Start Command Prompt as administrator and execute the following command:
    matlab -regserver

Note:

- *If you have multiple versions of MATLAB installed on your computer, the best practice is to run the matlab command from the matlabroot folder.*
- Enter the following command in the command line of MATLAB version that you want to interface with NetSim:
  matlab -regserver

10.3.2.2 Implement Weibull Distribution of MATLAB without using .m file

In this example we will replace the default Rayleigh Fading (part of the path loss calculation) used in NetSim, with a Fading Power calculated using the Weibull Distribution from MATLAB.

***Note**: This example uses 64-bit version of NetSim and MATLAB.*

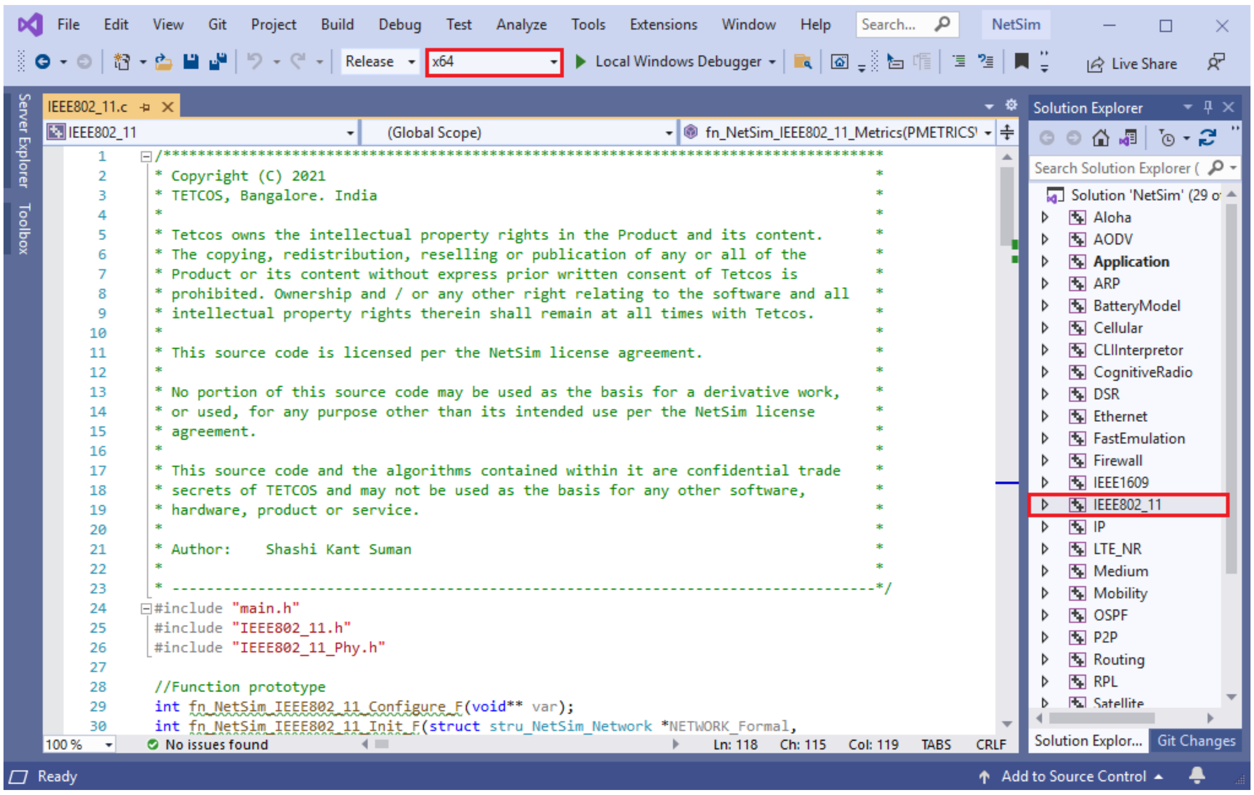
Procedure:

1. Create a MATLAB_Interface.c file inside the IEEE802_11 folder which can be found in the current workspace location of NetSim that you are running, and it would be something like "C:\Users\PC\Documents\NetSim\Workspaces\<Your default workspace>\src\Simulation\IEEE802_11" For more information on NetSim workspace refer Section 4 "Workspaces and Experiments". Write the following code inside the MATLAB_Interface.c file:

```
/*
*
* This is a simple program that illustrates how to call the MATLAB
* Engine functions from NetSim C Code.
*
*/
```

```
#include<windows.h>
#include<stdlib.h>
#include<stdio.h>
#include<string.h>
#include"engine.h"
#include"mat.h"
#include"mex.h"
#include"main.h"

char buf[BUFSIZ];
Engine *ep;
int status;
mxArray *h = NULL, *i = NULL, *j = NULL, *k = NULL;
mxArray *out;
double *result;
double fn_netsim_matlab_init()
{
  /*
  * Start the MATLAB engine
  */
  fprintf(stderr, "\nPress any key to Initialize MATLAB\n");
  _getch();
  if(!(ep = engOpen(NULL))) {
        MessageBox((HWND)NULL, (LPCWSTR)"Can't start MATLAB engine",
                (LPCWSTR) "MATLAB_Interface.c", MB_OK);
        exit(-1);
  }

  engEvalString(ep, "desktop");

  return 0;
}

double fn_netsim_matlab_run()
{

  //write your own implementation here
```

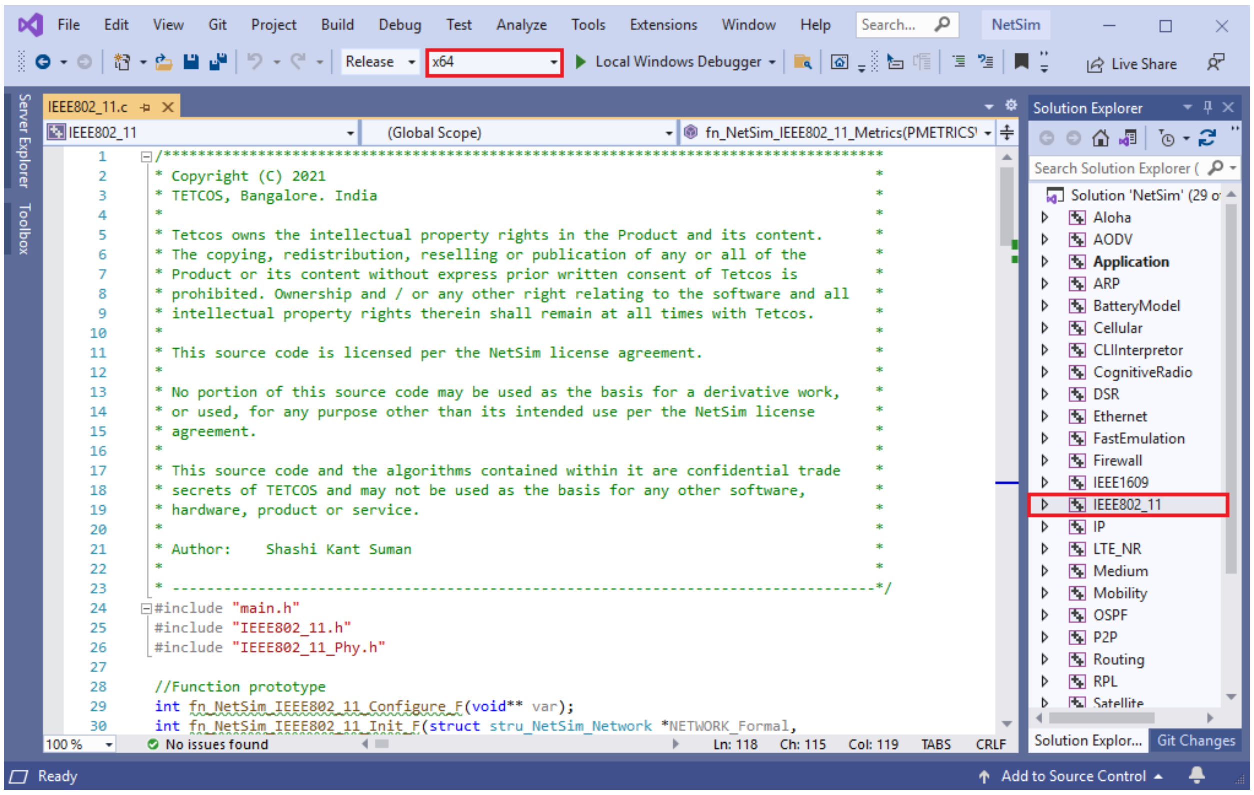
```
      int weibull_noncentrality = 1, weibull_scale = 2;

      engPutVariable(ep, "h", h);
      //use ProbDistUnivParam() function for matlab 2016
      /*sprintf_s(buf, BUFSIZ, "h=ProbDistUnivParam('weibull',[%d %d])",
      weibull_noncentrality, weibull_scale);*/
      //use makedist() function for matlab 2017 or higher
      sprintf_s(buf, BUFSIZ, "h=makedist('weibull',%d,%d)", weibull_noncentrality,
      weibull_scale);

      status = engEvalString(ep, buf);

      engPutVariable(ep, "i", i);
      sprintf_s(buf, BUFSIZ, "i=random(h,1)");

      status = engEvalString(ep, buf);
      out = engGetVariable(ep, "i");
      result = mxGetPr(out);

      return *result;
    }

    double fn_netsim_matlab_finish()
    {
      fprintf(stderr, "\nPress any key to close MATLAB\n");
      _getch();
      status = engEvalString(ep, "exit");
      return 0;
}
```



```
/************************************************************************************
 * Copyright (C) 2021                                                              *
 * TETCOS, Bangalore. India                                                        *
 *                                                                                 *
 * Tetcos owns the intellectual property rights in the Product and its content.    *
 * The copying, redistribution, reselling or publication of any or all of the      *
 * Product or its content without express prior written consent of Tetcos is       *
 * prohibited. Ownership and / or any other right relating to the software and all *
 * intellectual property rights therein shall remain at all times with Tetcos.     *
 *                                                                                 *
 * This source code is licensed per the NetSim license agreement.                  *
 *                                                                                 *
 * No portion of this source code may be used as the basis for a derivative work,  *
 * or used, for any purpose other than its intended use per the NetSim license     *
 * agreement.                                                                      *
 *                                                                                 *
 * This source code and the algorithms contained within it are confidential trade  *
 * secrets of TETCOS and may not be used as the basis for any other software,      *
 * hardware, product or service.                                                   *
 *                                                                                 *
 * Author:    Shashi Kant Suman                                                    *
 *                                                                                 *
 * ---------------------------------------------------------------------------------*/
#include "main.h"
#include "IEEE802_11.h"
#include "IEEE802_11_Phy.h"

//Function prototype
int fn_NetSim_IEEE802_11_Configure_F(void** var);
int fn_NetSim_IEEE802_11_Init_F(struct stru_NetSim_Network *NETWORK_Formal,
```

Figure 10-24: Create a MATLAB_Interface.c inside the IEEE802_11 folder

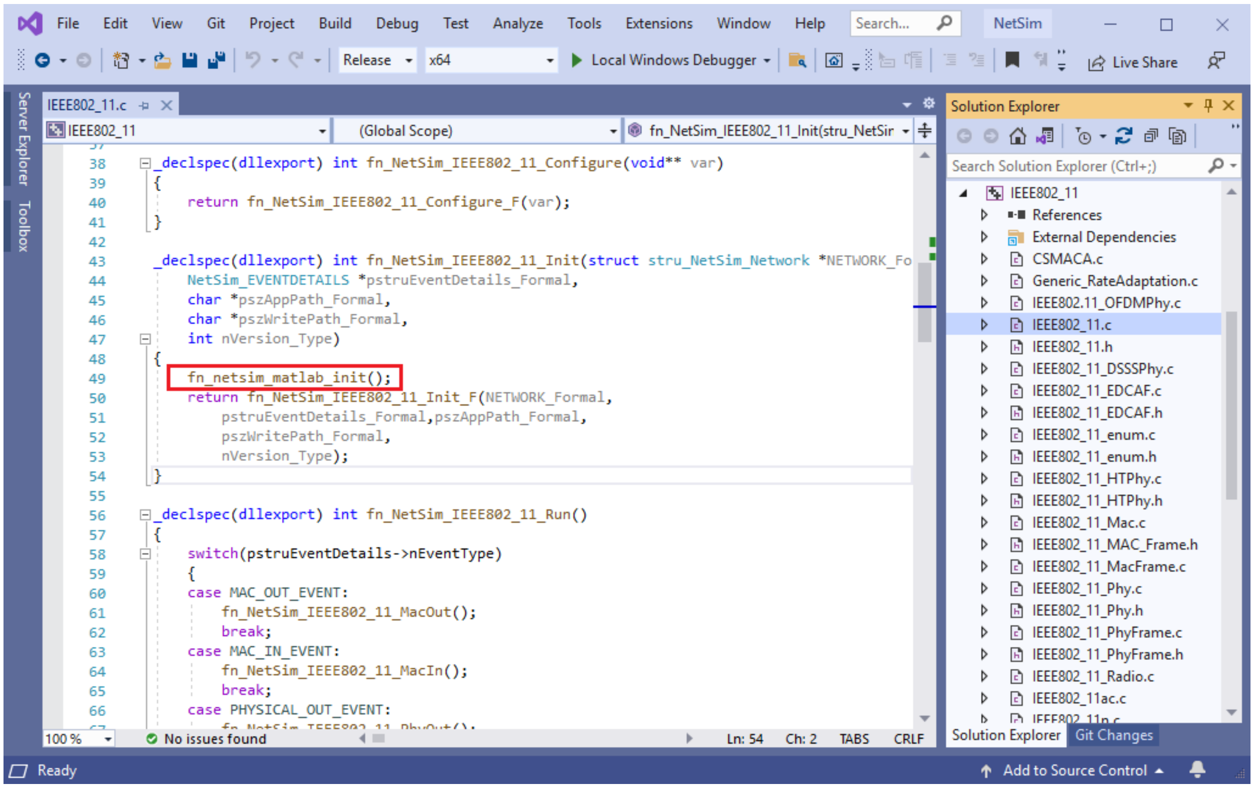
2. Now open the code and you can configure Visual studio platform to x64 to build 64 bit Dll files.
3. Right click on “IEEE802_11 Project” present in “Solution Explorer” window and select Add → Existing Item and select the MATLAB_Interface.c file.
4. MATLAB_Interface.c file contains the following functions.
    i. **fn_netsim_matlab_init()** - Opens the MATLAB Engine
    ii. **fn_netsim_matlab_run()** - Communicates with MATLAB Command Window
    iii. **fn_netsim_matlab_finish()** - Closes the MATLAB Engine
5. In the Solution Explorer under IEEE802.11 project Double click on the IEEE802_11.c file



```
/***************************************************************************
 * Copyright (C) 2021                                                      *
 * TETCOS, Bangalore. India                                                *
 *                                                                         *
 * Tetcos owns the intellectual property rights in the Product and its content. *
 * The copying, redistribution, reselling or publication of any or all of the *
 * Product or its content without express prior written consent of Tetcos is *
 * prohibited. Ownership and / or any other right relating to the software and all *
 * intellectual property rights therein shall remain at all times with Tetcos. *
 *                                                                         *
 * This source code is licensed per the NetSim license agreement.          *
 *                                                                         *
 * No portion of this source code may be used as the basis for a derivative work, *
 * or used, for any purpose other than its intended use per the NetSim license *
 * agreement.                                                              *
 *                                                                         *
 * This source code and the algorithms contained within it are confidential trade *
 * secrets of TETCOS and may not be used as the basis for any other software, *
 * hardware, product or service.                                           *
 *                                                                         *
 * Author:    Shashi Kant Suman                                            *
 *                                                                         *
 * ----------------------------------------------------------------------------*/
#include "main.h"
#include "IEEE802_11.h"
#include "IEEE802_11_Phy.h"

//Function prototype
int fn_NetSim_IEEE802_11_Configure_F(void** var);
int fn_NetSim_IEEE802_11_Init_F(struct stru_NetSim_Network *NETWORK_Formal,
```

Figure 10-25: Solution Explorer double click on the IEEE802_11.c file

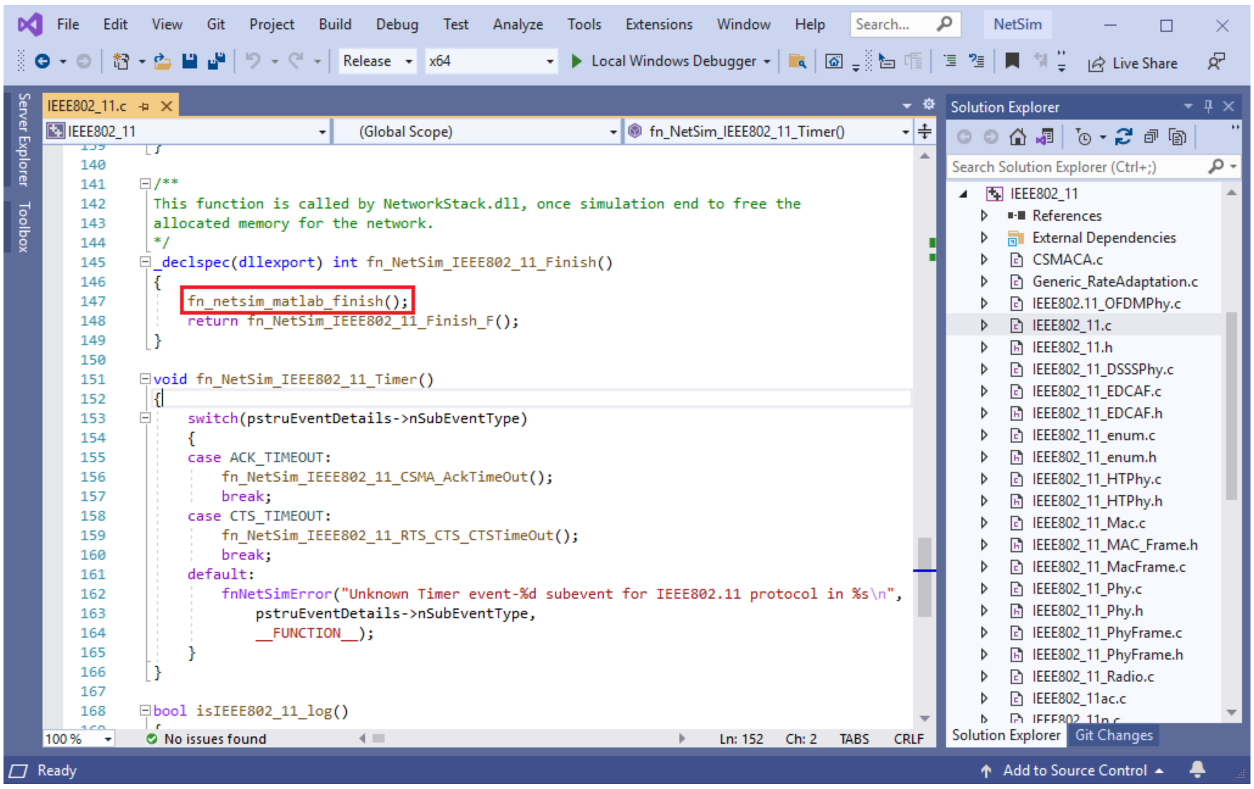
6. Add a call to **fn_netsim_matlab_init();** inside the fn_NetSim_IEEE802_11_Init() function.



```
_declspec(dllexport) int fn_NetSim_IEEE802_11_Configure(void** var)
{
    return fn_NetSim_IEEE802_11_Configure_F(var);
}

_declspec(dllexport) int fn_NetSim_IEEE802_11_Init(struct stru_NetSim_Network *NETWORK_Fo
    NetSim_EVENTDETAILS *pstruEventDetails_Formal,
    char *pszAppPath_Formal,
    char *pszWritePath_Formal,
    int nVersion_Type)
{
    fn_netsim_matlab_init();
    return fn_NetSim_IEEE802_11_Init_F(NETWORK_Formal,
        pstruEventDetails_Formal,pszAppPath_Formal,
        pszWritePath_Formal,
        nVersion_Type);
}

_declspec(dllexport) int fn_NetSim_IEEE802_11_Run()
{
    switch(pstruEventDetails->nEventType)
    {
    case MAC_OUT_EVENT:
        fn_NetSim_IEEE802_11_MacOut();
        break;
    case MAC_IN_EVENT:
        fn_NetSim_IEEE802_11_MacIn();
        break;
    case PHYSICAL_OUT_EVENT:
```

Figure 10-26: Added a fn_netsim_matlab_init(); inside the fn_NetSim_IEEE802_11_Init() function

7. Similarly add a call to **fn_netsim_matlab_finish();** inside the **fn_NetSim_IEEE802_11_Finish()** function.



```
/**
This function is called by NetworkStack.dll, once simulation end to free the
allocated memory for the network.
*/
_declspec(dllexport) int fn_NetSim_IEEE802_11_Finish()
{
    fn_netsim_matlab_finish();
    return fn_NetSim_IEEE802_11_Finish_F();
}

void fn_NetSim_IEEE802_11_Timer()
{
    switch(pstruEventDetails->nSubEventType)
    {
    case ACK_TIMEOUT:
        fn_NetSim_IEEE802_11_CSMA_AckTimeOut();
        break;
    case CTS_TIMEOUT:
        fn_NetSim_IEEE802_11_RTS_CTS_CTSTimeOut();
        break;
    default:
        fnNetSimError("Unknown Timer event-%d subevent for IEEE802.11 protocol in %s\n",
            pstruEventDetails->nSubEventType,
            __FUNCTION__);
    }
}

bool isIEEE802_11_log()
```

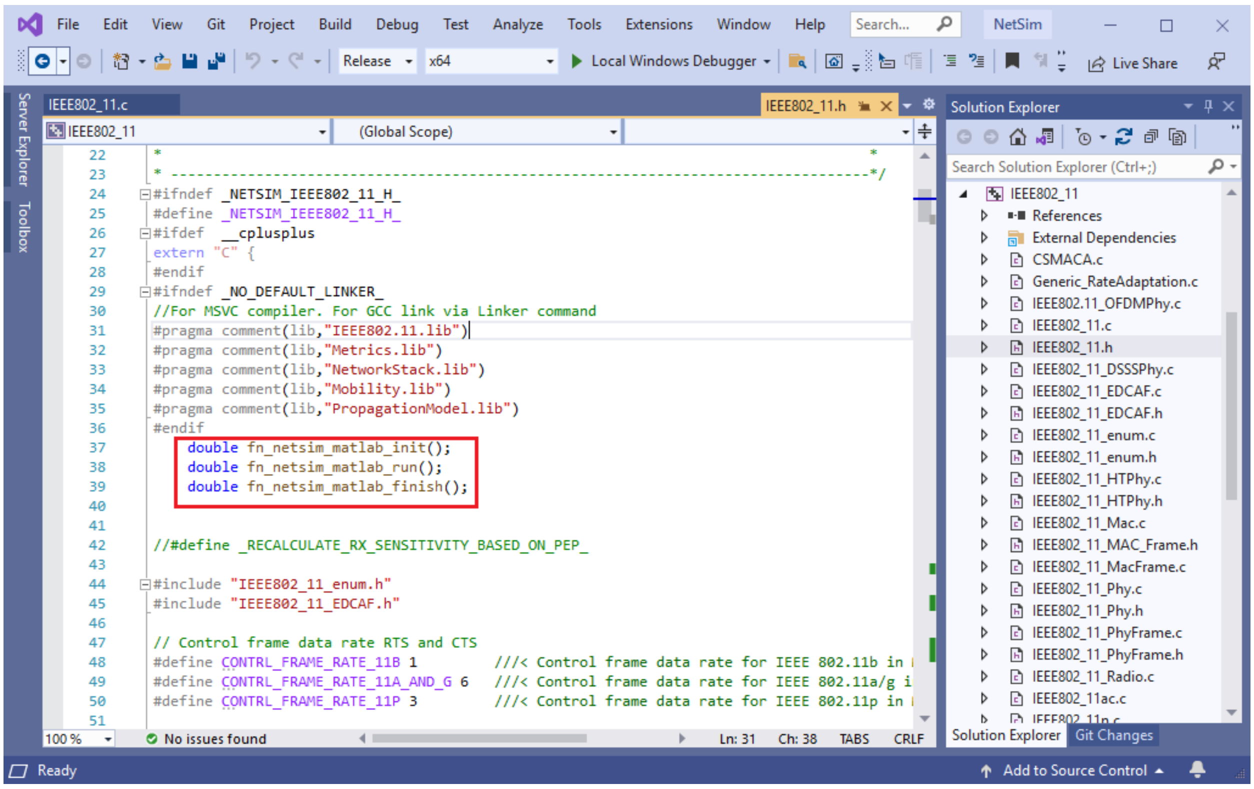
Figure 10-27: Added a fn_netsim_matlab_finish(); inside the fn_NetSim_IEEE802_11_Finish() function

8. In the Solution Explorer under IEEE802_11 project Double click on the IEEE802_11.h file. Add definitions of the following functions.

   double fn_netsim_matlab_init();

   double fn_netsim_matlab_run();

   double fn_netsim_matlab_finish();



```
 *                                                                            *
 * ---------------------------------------------------------------------------*/
#ifndef _NETSIM_IEEE802_11_H_
#define _NETSIM_IEEE802_11_H_
#ifdef  __cplusplus
extern "C" {
#endif
#ifndef _NO_DEFAULT_LINKER_
//For MSVC compiler. For GCC link via Linker command
#pragma comment(lib,"IEEE802.11.lib")
#pragma comment(lib,"Metrics.lib")
#pragma comment(lib,"NetworkStack.lib")
#pragma comment(lib,"Mobility.lib")
#pragma comment(lib,"PropagationModel.lib")
#endif
    double fn_netsim_matlab_init();
    double fn_netsim_matlab_run();
    double fn_netsim_matlab_finish();


//#define _RECALCULATE_RX_SENSITIVITY_BASED_ON_PEP_

#include "IEEE802_11_enum.h"
#include "IEEE802_11_EDCAF.h"

// Control frame data rate RTS and CTS
#define CONTRL_FRAME_RATE_11B 1         ///< Control frame data rate for IEEE 802.11b in
#define CONTRL_FRAME_RATE_11A_AND_G 6   ///< Control frame data rate for IEEE 802.11a/g i
#define CONTRL_FRAME_RATE_11P 3         ///< Control frame data rate for IEEE 802.11p in

```

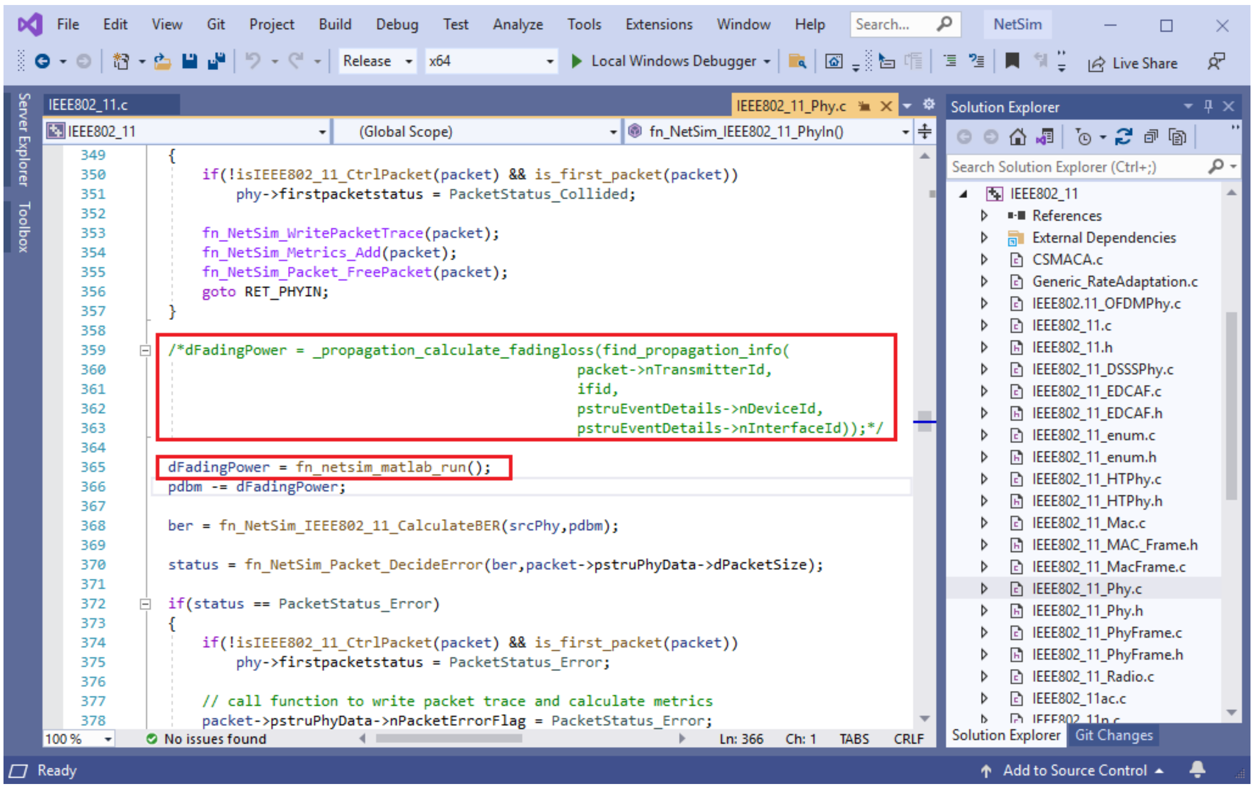
Figure 10-28: Added Matlab definitions in IEEE802_11.h file

9. In the Solution Explorer under IEEE802_11 project Double click on the IEEE802_11_PHY.c file
10. Inside **fn_Netsim_IEEE802_11_PHYIn ()** function comment the lines,

    dFadingPower = propagation_calculate_fadingloss(propagationHandle,

    packet->nTransmitterId,ifid,pstruEventDetails->nDeviceId, pstruEventDetails->nInterfaceId);
11. Make a call to the **fn_netsim_matlab_run()** function by adding the following line,

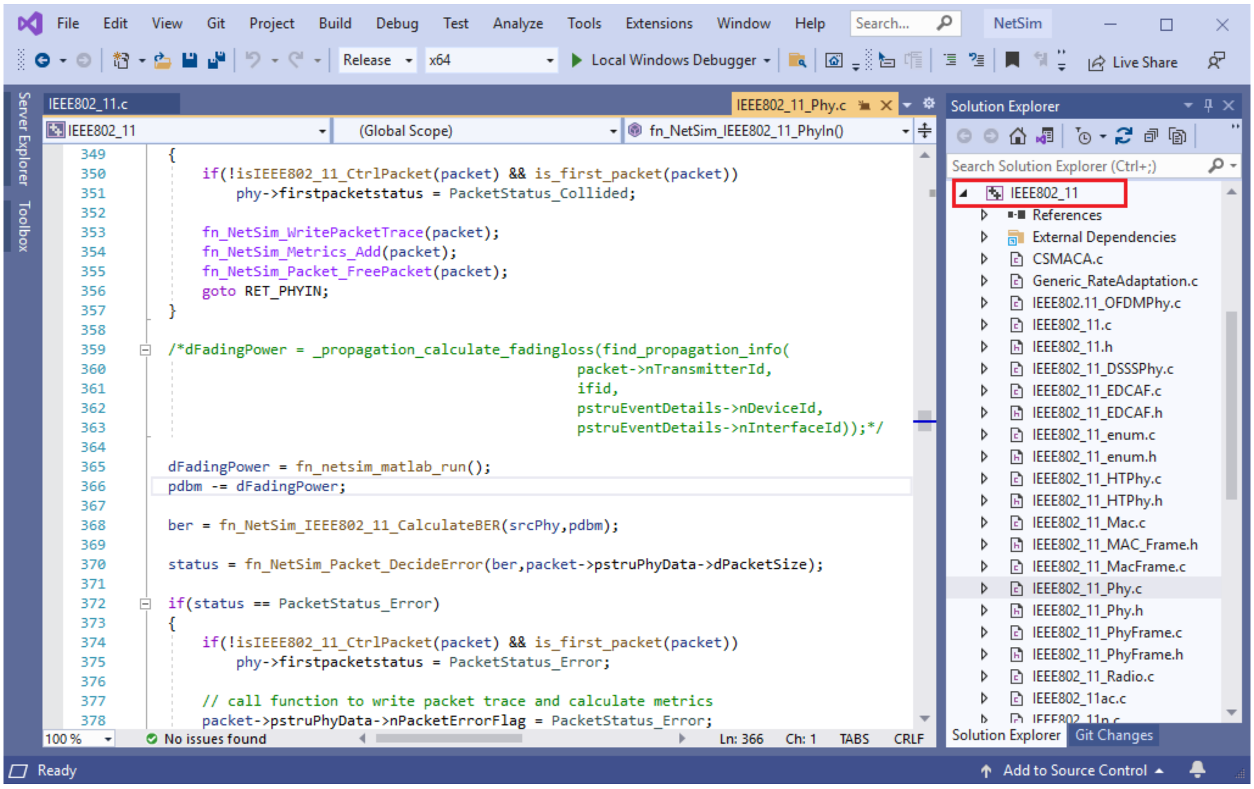
    dFadingPower = fn_netsim_matlab_run();



```
{
    if(!isIEEE802_11_CtrlPacket(packet) && is_first_packet(packet))
        phy->firstpacketstatus = PacketStatus_Collided;

    fn_NetSim_WritePacketTrace(packet);
    fn_NetSim_Metrics_Add(packet);
    fn_NetSim_Packet_FreePacket(packet);
    goto RET_PHYIN;
}

/*dFadingPower = _propagation_calculate_fadingloss(find_propagation_info(
                                                packet->nTransmitterId,
                                                ifid,
                                                pstruEventDetails->nDeviceId,
                                                pstruEventDetails->nInterfaceId));*/

dFadingPower = fn_netsim_matlab_run();
pdbm -= dFadingPower;

ber = fn_NetSim_IEEE802_11_CalculateBER(srcPhy,pdbm);

status = fn_NetSim_Packet_DecideError(ber,packet->pstruPhyData->dPacketSize);

if(status == PacketStatus_Error)
{
    if(!isIEEE802_11_CtrlPacket(packet) && is_first_packet(packet))
        phy->firstpacketstatus = PacketStatus_Error;

    // call function to write packet trace and calculate metrics
    packet->pstruPhyData->nPacketErrorFlag = PacketStatus_Error;
```

Figure 10-29: Call to the fn_netsim_matlab_run() function

12. To compile a MATLAB engine application in the Microsoft Visual Studio (2019) environment, Right click on the IEEE802_11 project and select PROPERTIES in the solution explorer. Once this window has opened, make the following changes:



```
{
    if(!isIEEE802_11_CtrlPacket(packet) && is_first_packet(packet))
        phy->firstpacketstatus = PacketStatus_Collided;

    fn_NetSim_WritePacketTrace(packet);
    fn_NetSim_Metrics_Add(packet);
    fn_NetSim_Packet_FreePacket(packet);
    goto RET_PHYIN;
}

/*dFadingPower = _propagation_calculate_fadingloss(find_propagation_info(
                                                packet->nTransmitterId,
                                                ifid,
                                                pstruEventDetails->nDeviceId,
                                                pstruEventDetails->nInterfaceId));*/

dFadingPower = fn_netsim_matlab_run();
pdbm -= dFadingPower;

ber = fn_NetSim_IEEE802_11_CalculateBER(srcPhy,pdbm);

status = fn_NetSim_Packet_DecideError(ber,packet->pstruPhyData->dPacketSize);

if(status == PacketStatus_Error)
{
    if(!isIEEE802_11_CtrlPacket(packet) && is_first_packet(packet))
        phy->firstpacketstatus = PacketStatus_Error;

    // call function to write packet trace and calculate metrics
    packet->pstruPhyData->nPacketErrorFlag = PacketStatus_Error;
```

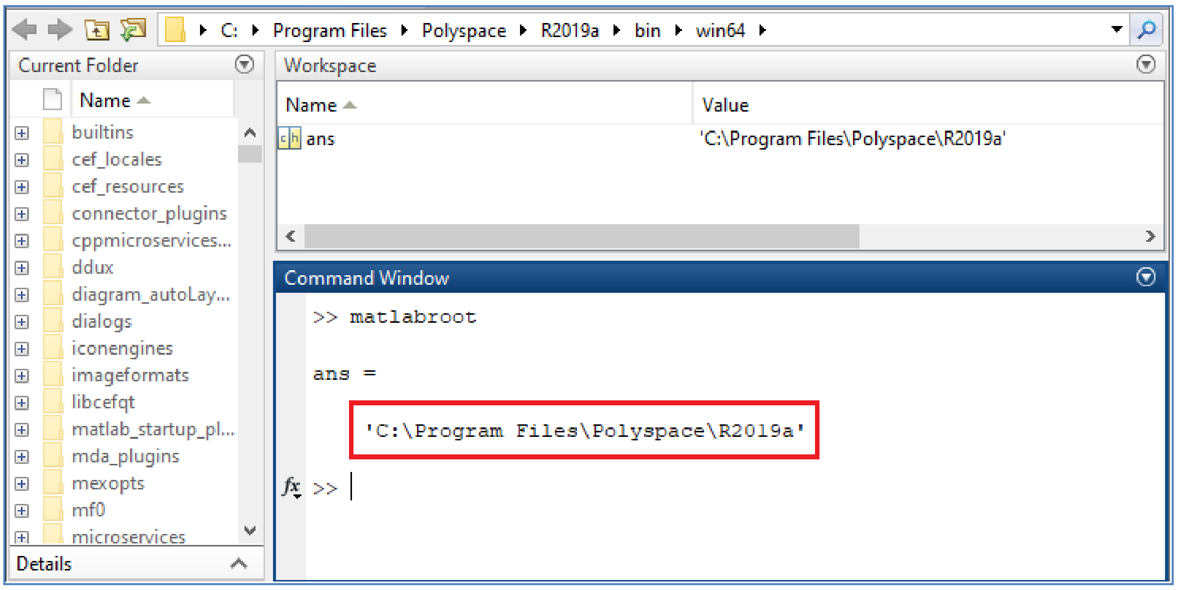
Figure 10-30: Right click on the IEEE802_11 project and select Properties

13. Under C/C++ → General, add the following directory to the field ADDITIONAL INCLUDE DIRECTORIES:

    <Path where MATLAB is installed>\extern\include

NOTE: *To determine path where MATLAB is installed, entering the following command in the MATLAB command prompt:*

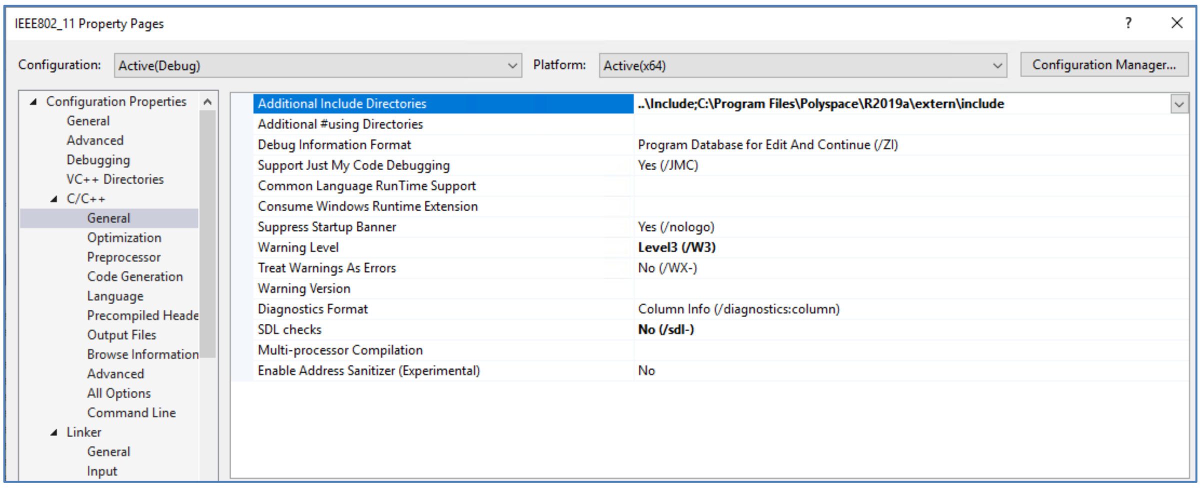
matlabroot



Name	Value
ans	'C:\Program Files\Polyspace\R2019a'

```
>> matlabroot

ans =

    'C:\Program Files\Polyspace\R2019a'

fx >>
```

Figure 10-31: MATLAB Command Prompt

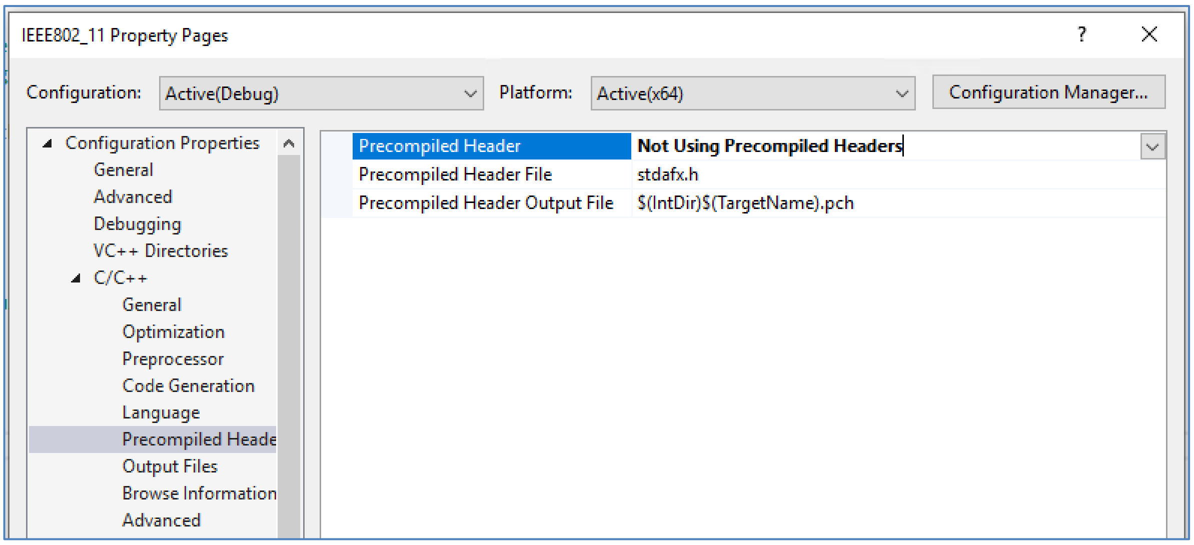


Configuration: Active(Debug)   Platform: Active(x64)

Additional Include Directories	..\Include;C:\Program Files\Polyspace\R2019a\extern\include
Additional #using Directories	
Debug Information Format	Program Database for Edit And Continue (/ZI)
Support Just My Code Debugging	Yes (/JMC)
Common Language RunTime Support	
Consume Windows Runtime Extension	
Suppress Startup Banner	Yes (/nologo)
Warning Level	**Level3 (/W3)**
Treat Warnings As Errors	No (/WX-)
Warning Version	
Diagnostics Format	Column Info (/diagnostics:column)
SDL checks	**No (/sdl-)**
Multi-processor Compilation	
Enable Address Sanitizer (Experimental)	No

Figure 10-32: Determine MATLAB Path in Additional Include Directories

14. Under C/C++ → Precompiled Headers, set PRECOMPILED HEADERS as "Not Using Precompiled Headers".


IEEE802_11 Property Pages

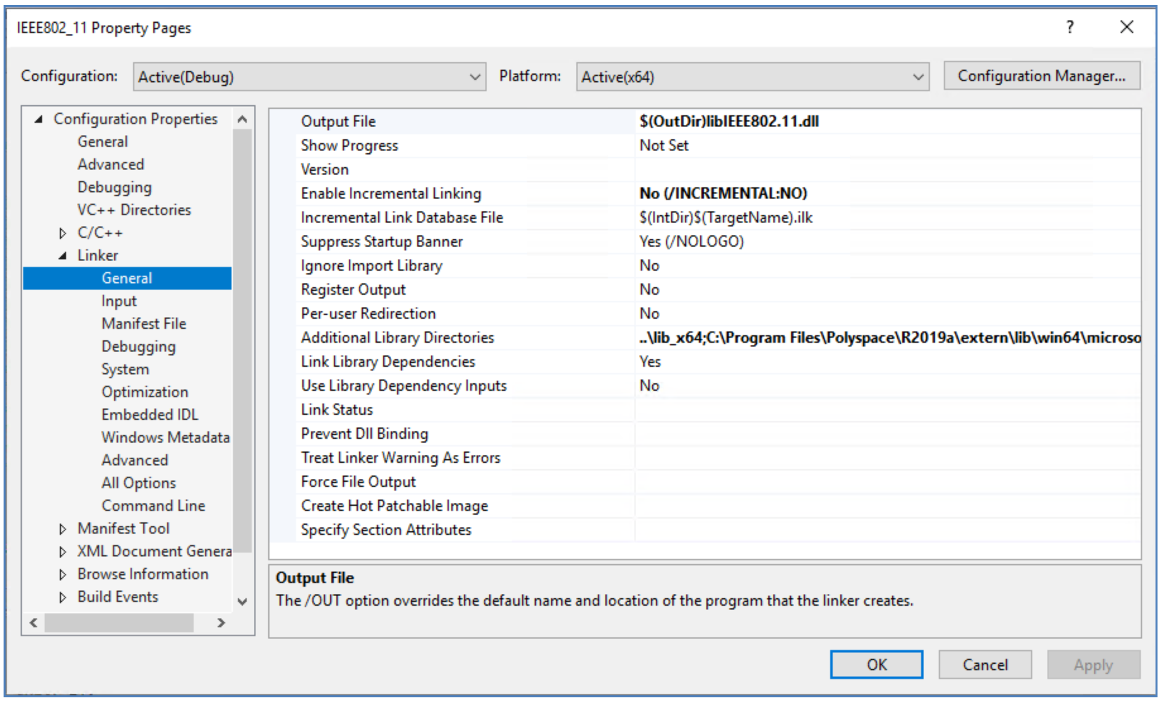
Configuration: Active(Debug)   Platform: Active(x64)   Configuration Manager...

Precompiled Header	**Not Using Precompiled Headers**
Precompiled Header File	stdafx.h
Precompiled Header Output File	$(IntDir)$(TargetName).pch

Figure 10-33: Select Precompiled Header as “Not Using Precompiled Headers”

15. Under Linker → General, add the directory to the field ADDITIONAL LIBRARY DIRECTORIES:

<Path where MATLAB is installed>\extern\lib\win64\microsoft


IEEE802_11 Property Pages

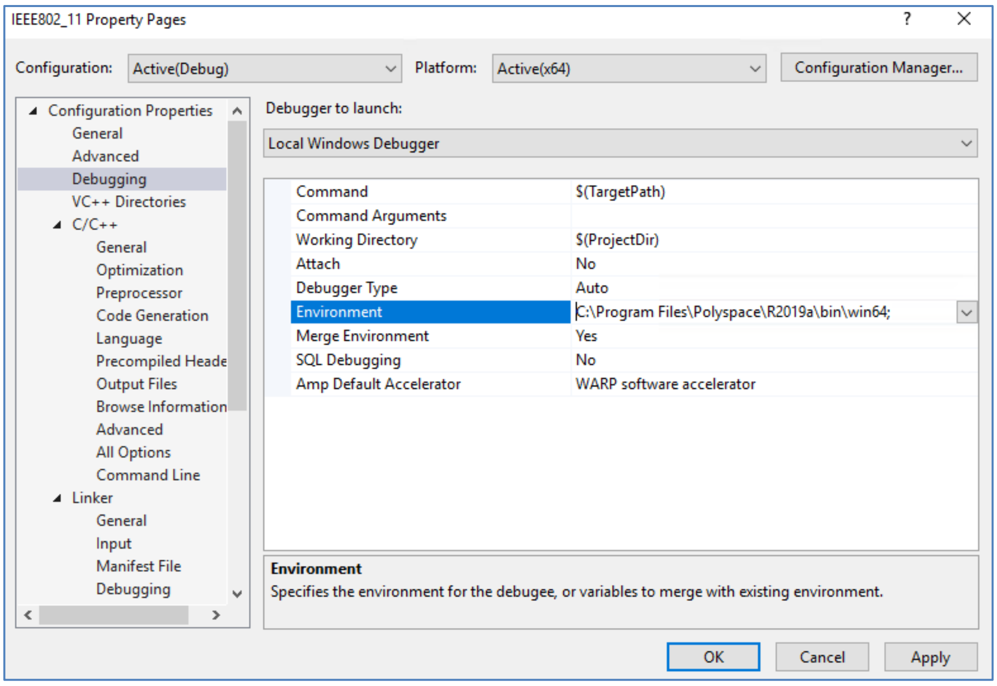
Configuration: Active(Debug)   Platform: Active(x64)   Configuration Manager...

Output File	**$(OutDir)libIEEE802.11.dll**
Show Progress	Not Set
Version	
Enable Incremental Linking	**No (/INCREMENTAL:NO)**
Incremental Link Database File	$(IntDir)$(TargetName).ilk
Suppress Startup Banner	Yes (/NOLOGO)
Ignore Import Library	No
Register Output	No
Per-user Redirection	No
Additional Library Directories	**..\lib_x64;C:\Program Files\Polyspace\R2019a\extern\lib\win64\microso**
Link Library Dependencies	Yes
Use Library Dependency Inputs	No
Link Status	
Prevent Dll Binding	
Treat Linker Warning As Errors	
Force File Output	
Create Hot Patchable Image	
Specify Section Attributes	

Output File
The /OUT option overrides the default name and location of the program that the linker creates.

Figure 10-34: SetAdditional Library Directories to <Path where MATLAB is installed>\extern\lib\win64\microsoft

16. Under Configuration Properties →Debugging, Add the following Target path in the ENVIRONMENT: <Path where MATLAB is installed>\bin\win64



Configuration: Active(Debug) Platform: Active(x64)

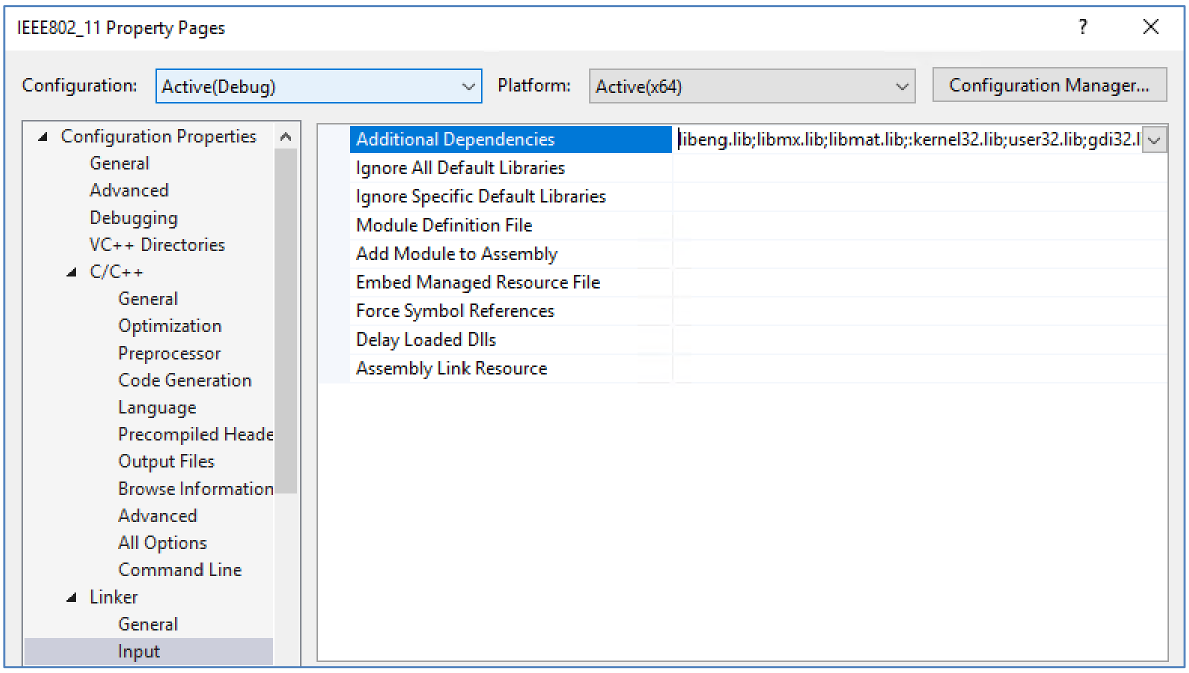
Debugger to launch: Local Windows Debugger

Command	$(TargetPath)
Command Arguments	
Working Directory	$(ProjectDir)
Attach	No
Debugger Type	Auto
Environment	C:\Program Files\Polyspace\R2019a\bin\win64;
Merge Environment	Yes
SQL Debugging	No
Amp Default Accelerator	WARP software accelerator

Environment
Specifies the environment for the debugee, or variables to merge with existing environment.

Figure 10-35: Set Environment to <Path where MATLAB is installed>\bin\win64

17. Under Linker → Input, add the following names to the field marked ADDITIONAL DEPENDENCIES: libeng.lib;libmx.lib;libmat.lib; and Click on Apply and OK.



Configuration: Active(Debug) Platform: Active(x64)

Additional Dependencies	libeng.lib;libmx.lib;libmat.lib;;kernel32.lib;user32.lib;gdi32.l
Ignore All Default Libraries	
Ignore Specific Default Libraries	
Module Definition File	
Add Module to Assembly	
Embed Managed Resource File	
Force Symbol References	
Delay Loaded Dlls	
Assembly Link Resource	

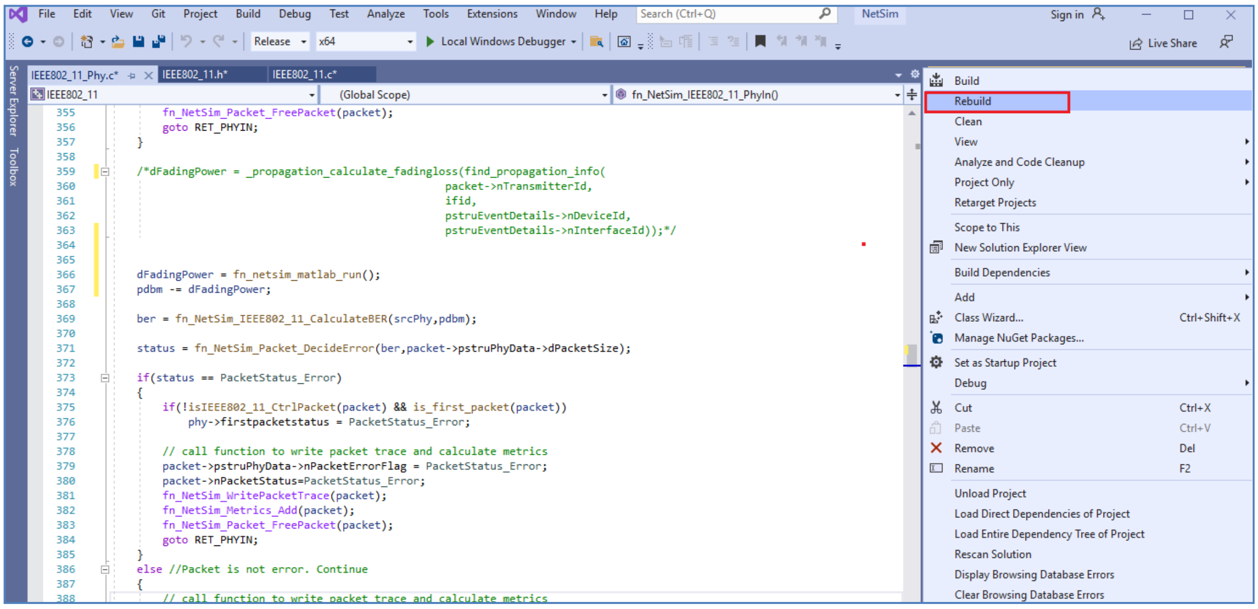
Figure 10-36: Set Additional Dependencies as libeng.lib;libmx.lib;libmat.lib;

18. Now Right Click on IEEE802_11 project and select Rebuild.

Note:

- *Before Rebuilding make sure you have access to IEEE802.11 folder, if the access denied while rebuilding go to* "C:\Users\PC\Documents\NetSim\Workspaces\<Yourdefaultworkspace>\src\Simulation\IEEE802_11" *path..*

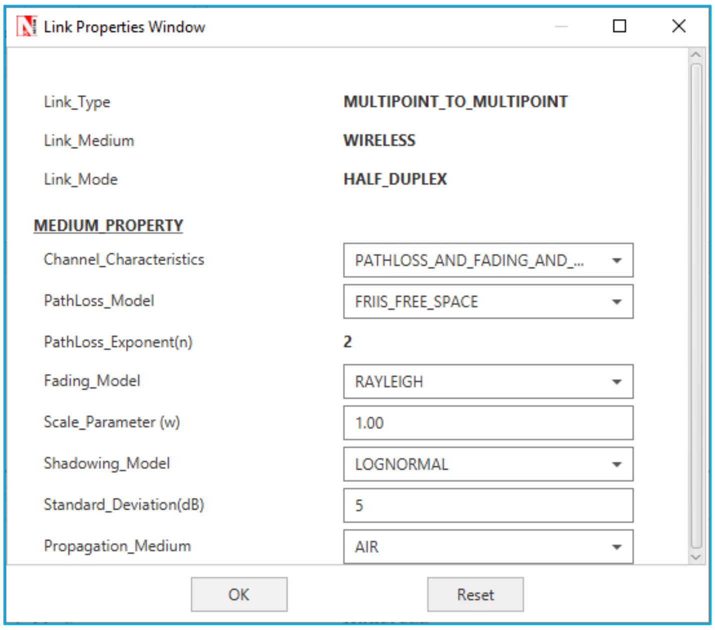
- *Right click on IEEE802.11 folder, select on properties and deselect "Read only" attribute and click on apply.*



```
        fn_NetSim_Packet_FreePacket(packet);
        goto RET_PHYIN;
    }

    /*dFadingPower = _propagation_calculate_fadingloss(find_propagation_info(
                                                    packet->nTransmitterId,
                                                    ifid,
                                                    pstruEventDetails->nDeviceId,
                                                    pstruEventDetails->nInterfaceId));*/


    dFadingPower = fn_netsim_matlab_run();
    pdbm -= dFadingPower;

    ber = fn_NetSim_IEEE802_11_CalculateBER(srcPhy,pdbm);

    status = fn_NetSim_Packet_DecideError(ber,packet->pstruPhyData->dPacketSize);

    if(status == PacketStatus_Error)
    {
        if(!isIEEE802_11_CtrlPacket(packet) && is_first_packet(packet))
            phy->firstpacketstatus = PacketStatus_Error;

        // call function to write packet trace and calculate metrics
        packet->pstruPhyData->nPacketErrorFlag = PacketStatus_Error;
        packet->nPacketStatus=PacketStatus_Error;
        fn_NetSim_WritePacketTrace(packet);
        fn_NetSim_Metrics_Add(packet);
        fn_NetSim_Packet_FreePacket(packet);
        goto RET_PHYIN;
    }
    else //Packet is not error. Continue
    {
        // call function to write packet trace and calculate metrics
```

Figure 10-37: Rebuild IEEE802_11 project

19. A new libIEEE802.11.dll gets created in the bin_x64 folder of NetSim's current workspace path <C:\Users\PC\Documents\NetSim\Workspaces\<Your default workspace>\bin_x64>. For more information, follow steps provided in Section 9.1 "Writing your own code".
20. Run NetSim in **Administrative** mode. Create a Network scenario involving IEEE802_11 say MANET, right click on the Adhoc link and select properties. Make sure that the Channel Characteristics is set to PathLoss and Fading and Shadowing.



Link Properties Window

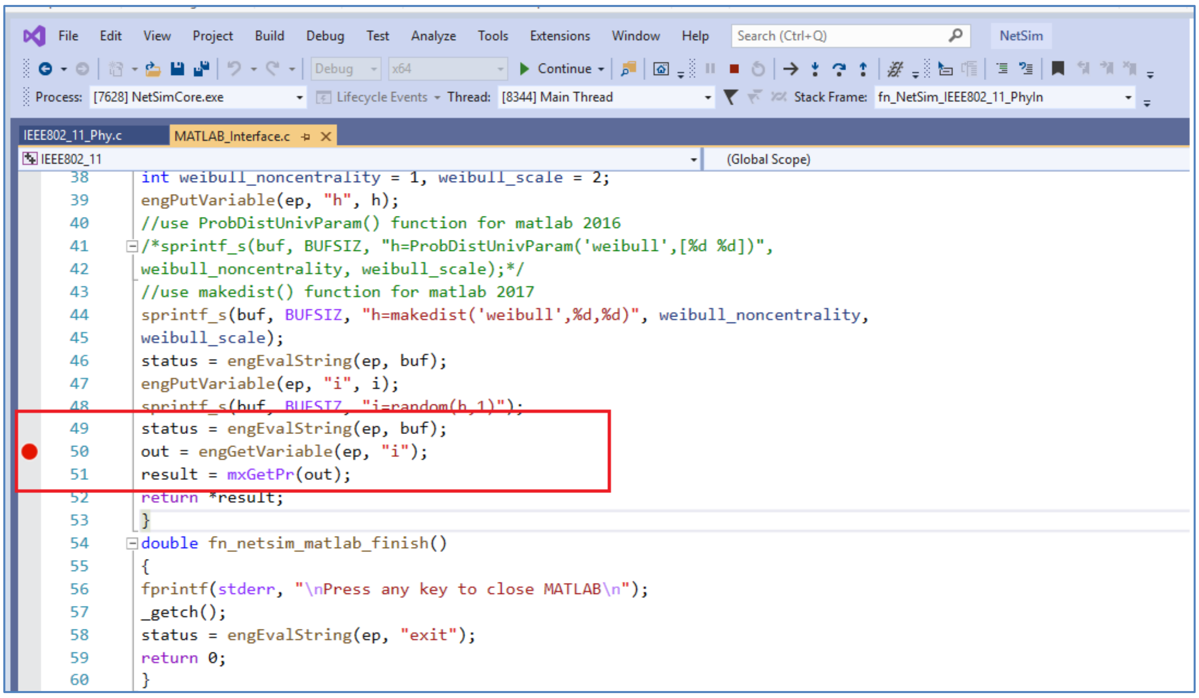
Link_Type	MULTIPOINT_TO_MULTIPOINT
Link_Medium	WIRELESS
Link_Mode	HALF_DUPLEX
MEDIUM PROPERTY	
Channel_Characteristics	PATHLOSS_AND_FADING_AND_...
PathLoss_Model	FRIIS_FREE_SPACE
PathLoss_Exponent(n)	2
Fading_Model	RAYLEIGH
Scale_Parameter (w)	1.00
Shadowing_Model	LOGNORMAL
Standard_Deviation(dB)	5
Propagation_Medium	AIR

Figure 10-38: Wireless Link Properties Window

21. Perform Simulation. You will be observed to initiate the MATLAB in NetSim console window After clicking on enter, MATLAB command window starts and gets closed once the simulation is over.

10.3.2.3 Debug and understand communication between NetSim and MATLAB

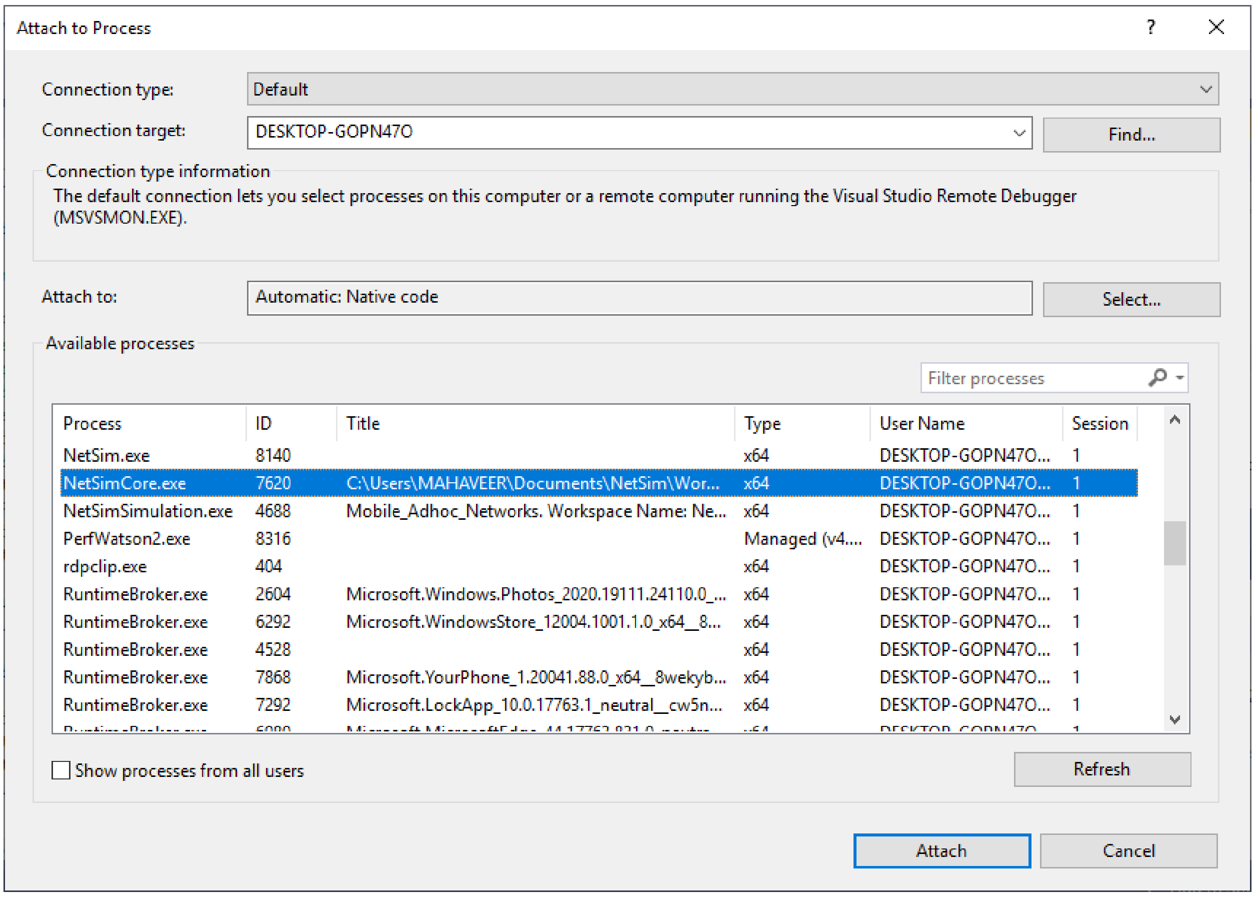
1. In the Solution Explorer under IEEE802_11 project double click on MATLAB_Interface.c file and place a breakpoint inside the **fn_netsim_matlab_run()** function before the return statement as shown in below screenshot.



```
int weibull_noncentrality = 1, weibull_scale = 2;
engPutVariable(ep, "h", h);
//use ProbDistUnivParam() function for matlab 2016
/*sprintf_s(buf, BUFSIZ, "h=ProbDistUnivParam('weibull',[%d %d])",
weibull_noncentrality, weibull_scale);*/
//use makedist() function for matlab 2017
sprintf_s(buf, BUFSIZ, "h=makedist('weibull',%d,%d)", weibull_noncentrality,
weibull_scale);
status = engEvalString(ep, buf);
engPutVariable(ep, "i", i);
sprintf_s(buf, BUFSIZ, "i=random(h,1)");
status = engEvalString(ep, buf);
out = engGetVariable(ep, "i");
result = mxGetPr(out);
return *result;
}
double fn_netsim_matlab_finish()
{
fprintf(stderr, "\nPress any key to close MATLAB\n");
_getch();
status = engEvalString(ep, "exit");
return 0;
}
```

Figure 10-39: Place a breakpoint inside the fn_netsim_matlab_run() function

2. Rebuild the code.
3. Now run the NetSim Scenario. The simulation window stops for user interrupt.
4. In Visual studio, go to Debug → Attach to Process.
5. From the list of Processes select NetSimCore.exe and click on Attach.



Attach to Process

Connection type: Default

Connection target: DESKTOP-GOPN47O

Connection type information
The default connection lets you select processes on this computer or a remote computer running the Visual Studio Remote Debugger (MSVSMON.EXE).

Attach to: Automatic: Native code

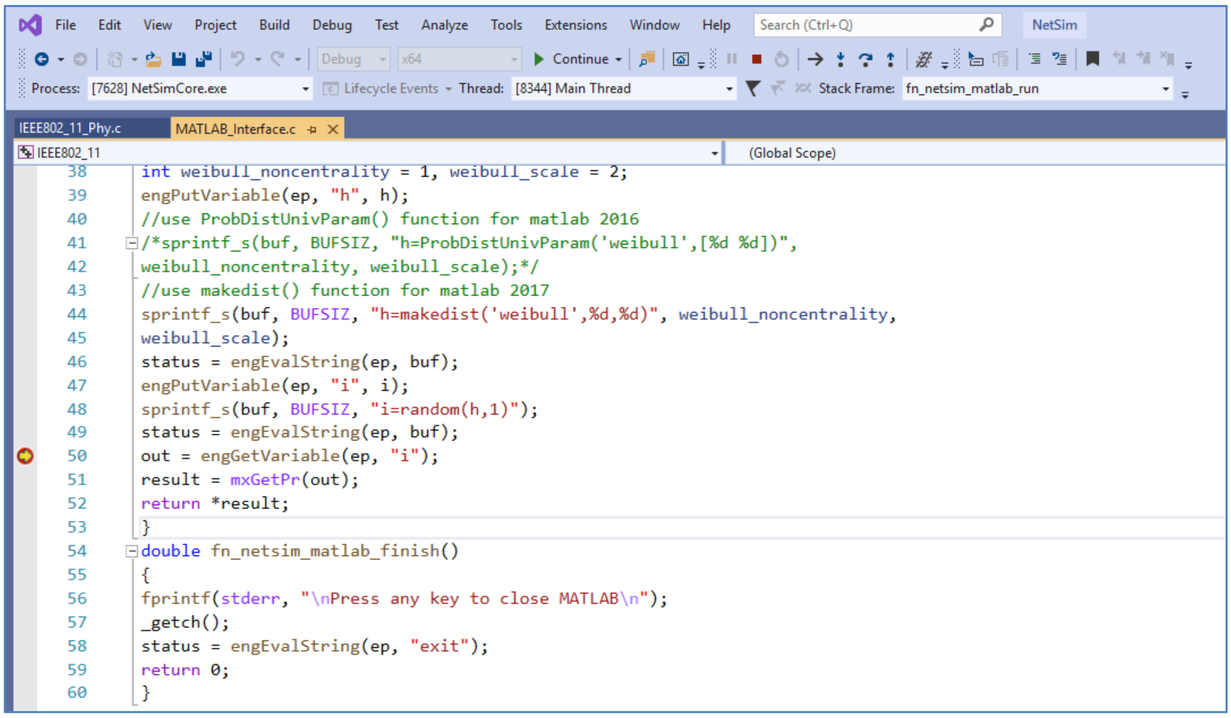
Available processes

Process	ID	Title	Type	User Name	Session
NetSim.exe	8140		x64	DESKTOP-GOPN47O...	1
NetSimCore.exe	7620	C:\Users\MAHAVEER\Documents\NetSim\Wor...	x64	DESKTOP-GOPN47O...	1
NetSimSimulation.exe	4688	Mobile_Adhoc_Networks. Workspace Name: Ne...	x64	DESKTOP-GOPN47O...	1
PerfWatson2.exe	8316		Managed (v4....	DESKTOP-GOPN47O...	1
rdpclip.exe	404		x64	DESKTOP-GOPN47O...	1
RuntimeBroker.exe	2604	Microsoft.Windows.Photos_2020.19111.24110.0_...	x64	DESKTOP-GOPN47O...	1
RuntimeBroker.exe	6292	Microsoft.WindowsStore_12004.1001.1.0_x64__8...	x64	DESKTOP-GOPN47O...	1
RuntimeBroker.exe	4528		x64	DESKTOP-GOPN47O...	1
RuntimeBroker.exe	7868	Microsoft.YourPhone_1.20041.88.0_x64__8wekyb...	x64	DESKTOP-GOPN47O...	1
RuntimeBroker.exe	7292	Microsoft.LockApp_10.0.17763.1_neutral__cw5n...	x64	DESKTOP-GOPN47O...	1

☐ Show processes from all users

Figure 10-40: Select NetSimCore.exe in Attach to Process Window

6. Now go to the Simulation window and press Enter.
7. MATLAB Command Window and MATLAB Desktop Window will start and breakpoint in Visual Studio gets triggered.

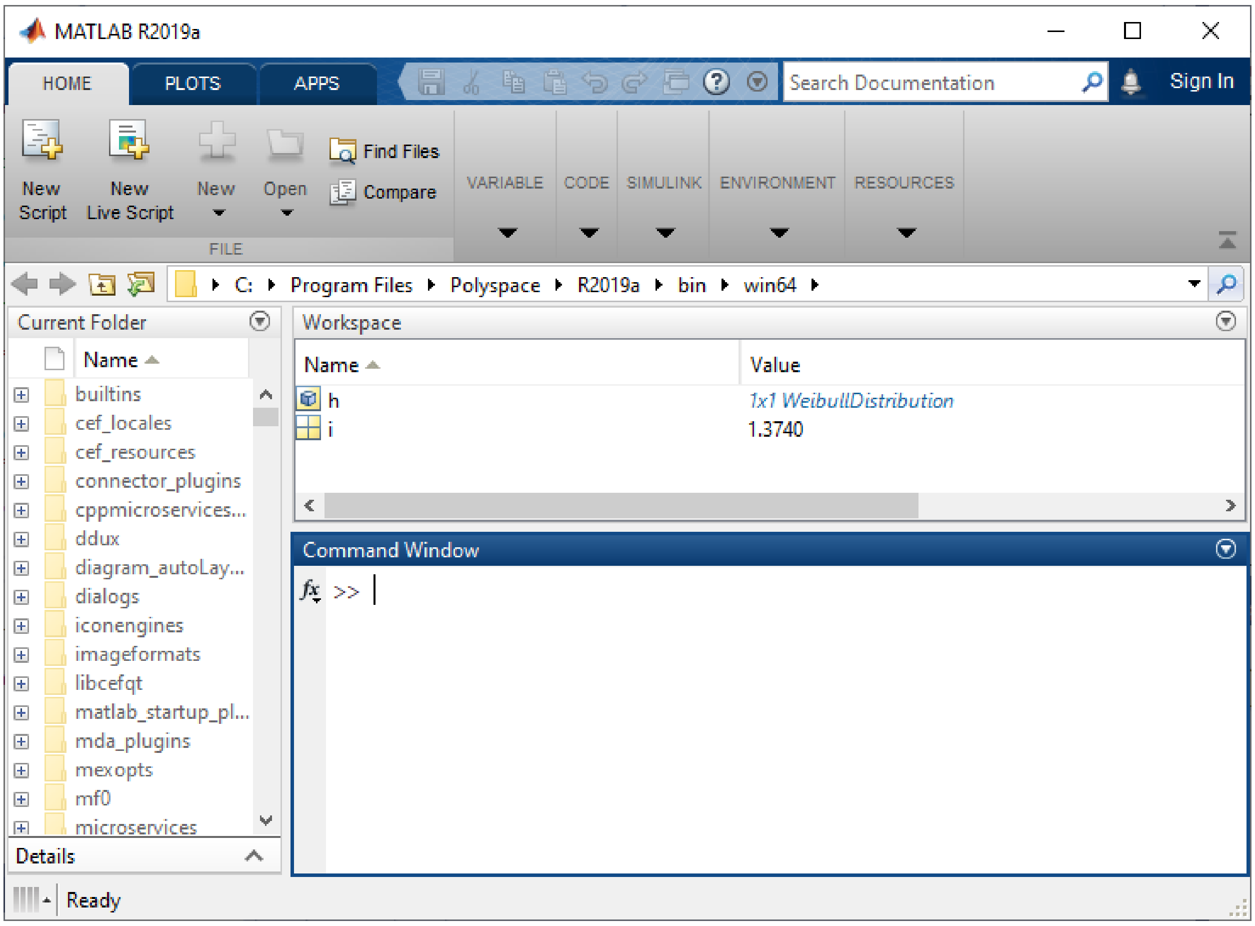


Process: [7628] NetSimCore.exe   Thread: [8344] Main Thread   Stack Frame: fn_netsim_matlab_run

```
int weibull_noncentrality = 1, weibull_scale = 2;
engPutVariable(ep, "h", h);
//use ProbDistUnivParam() function for matlab 2016
/*sprintf_s(buf, BUFSIZ, "h=ProbDistUnivParam('weibull',[%d %d])",
weibull_noncentrality, weibull_scale);*/
//use makedist() function for matlab 2017
sprintf_s(buf, BUFSIZ, "h=makedist('weibull',%d,%d)", weibull_noncentrality,
weibull_scale);
status = engEvalString(ep, buf);
engPutVariable(ep, "i", i);
sprintf_s(buf, BUFSIZ, "i=random(h,1)");
status = engEvalString(ep, buf);
out = engGetVariable(ep, "i");
result = mxGetPr(out);
return *result;
}
double fn_netsim_matlab_finish()
{
fprintf(stderr, "\nPress any key to close MATLAB\n");
_getch();
status = engEvalString(ep, "exit");
return 0;
}
```

Figure 10-41: Once Simulation Start and breakpoint gets triggered in Visual Studio

8. Now when debugging (say, by pressing F5 each time) you will find the computation taking place in the MATLAB Workspace.

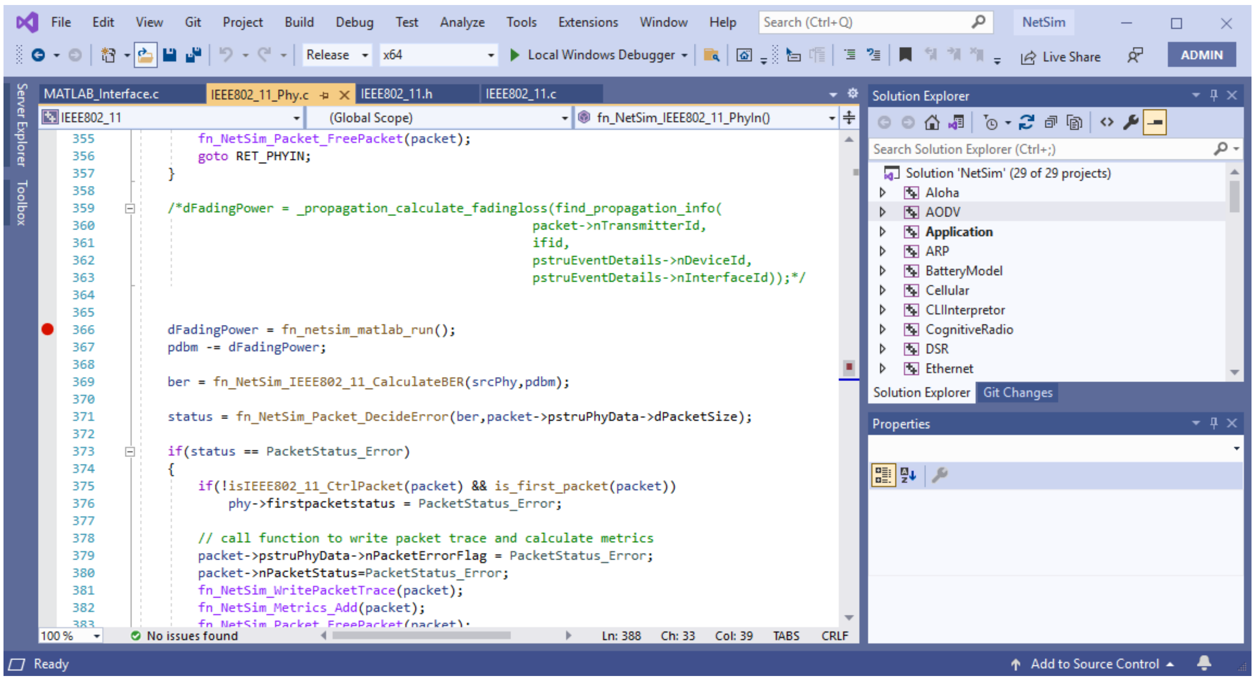


Name	Value
h	1x1 WeibullDistribution
i	1.3740

```
fx >>
```

Figure 10-42: MATLAB Workspace

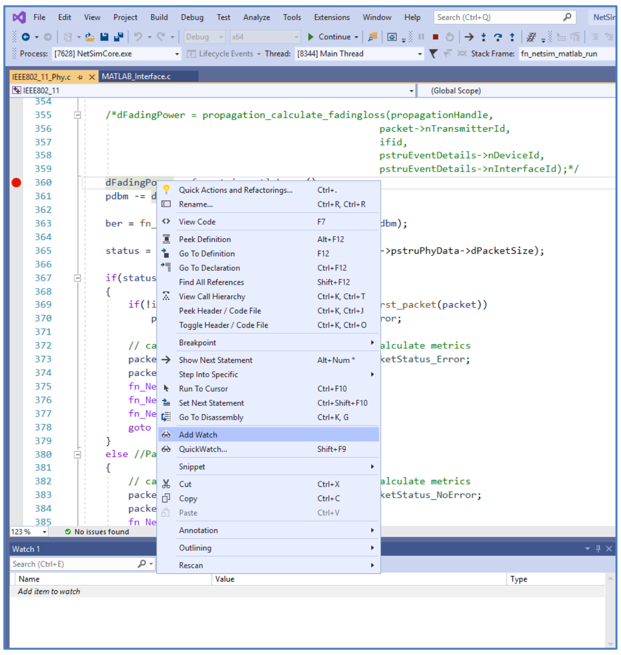
9. This value of i obtained from MATLAB is used to calculate fading power instead of the already available models in NetSim.
10. Now place another breakpoint after the line dFadingPower = fn_netsim_matlab_run()



```
            fn_NetSim_Packet_FreePacket(packet);
            goto RET_PHYIN;
        }

        /*dFadingPower = _propagation_calculate_fadingloss(find_propagation_info(
                                                    packet->nTransmitterId,
                                                    ifid,
                                                    pstruEventDetails->nDeviceId,
                                                    pstruEventDetails->nInterfaceId));*/


        dFadingPower = fn_netsim_matlab_run();
        pdbm -= dFadingPower;

        ber = fn_NetSim_IEEE802_11_CalculateBER(srcPhy,pdbm);

        status = fn_NetSim_Packet_DecideError(ber,packet->pstruPhyData->dPacketSize);

        if(status == PacketStatus_Error)
        {
            if(!isIEEE802_11_CtrlPacket(packet) && is_first_packet(packet))
                phy->firstpacketstatus = PacketStatus_Error;

            // call function to write packet trace and calculate metrics
            packet->pstruPhyData->nPacketErrorFlag = PacketStatus_Error;
            packet->nPacketStatus=PacketStatus_Error;
            fn_NetSim_WritePacketTrace(packet);
            fn_NetSim_Metrics_Add(packet);
            fn_NetSim_Packet_FreePacket(packet);
```

Figure 10-43: Added Another breakpoint after the line dFadingPower = fn_netsim_matlab_run()

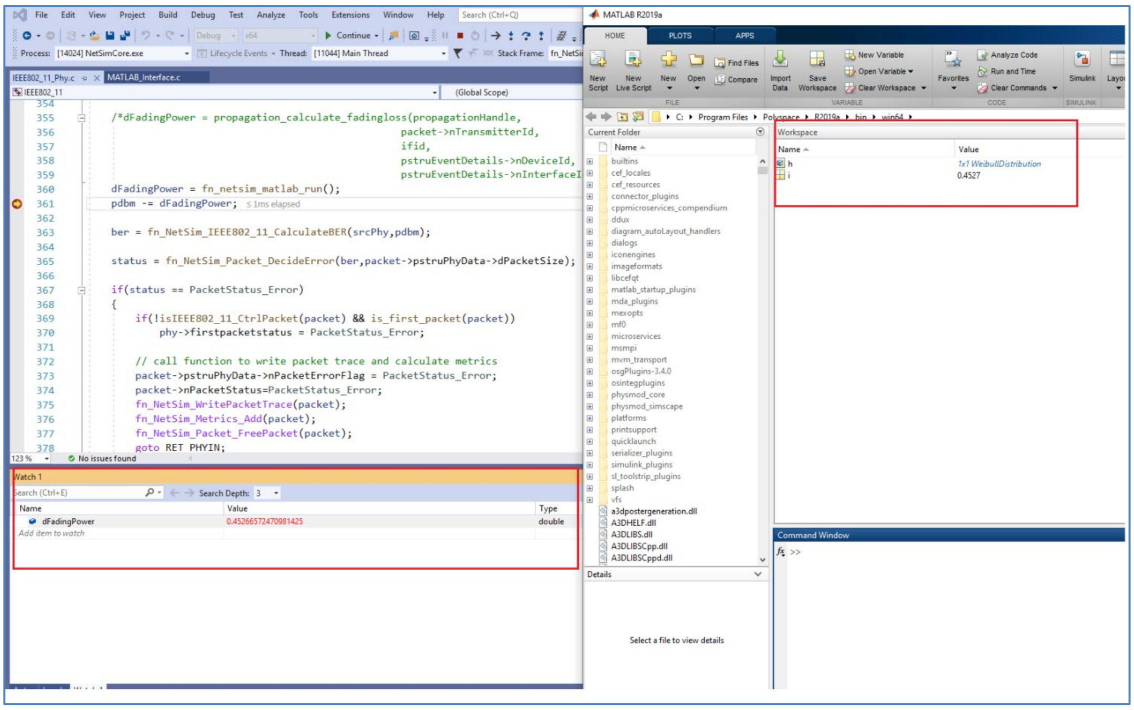
11. Add the variable dFadingPower in IEEE802_11_Phy.c file, to watch. For this, right click on the variable dFadingPower and select “Add Watch” option. You will find a watch window containing the variable name and its value in the bottom left corner.



```
        /*dFadingPower = propagation_calculate_fadingloss(propagationHandle,
                                                    packet->nTransmitterId,
                                                    ifid,
                                                    pstruEventDetails->nDeviceId,
                                                    pstruEventDetails->nInterfaceId);*/

        dFadingPo
        pdbm -= d

        ber = fn_                                   dbm);

        status =                                    ->pstruPhyData->dPacketSize);

        if(status
        {
            if(!i                                   rst_packet(packet))
                p                                   ror;

            // ca                                   alculate metrics
            packe                                   ketStatus_Error;
            packe
            fn_Ne
            fn_Ne
            fn_Ne
            goto
        }
        else //Pa
        {
            // ca                                   alculate metrics
            packe                                   ketStatus_NoError;
            packe
            fn Ne
```

Figure 10-44: Right Click on dFadingPower and Select “Add Watch” option

12. Now when debugging (say by pressing F5 each time) you will find that the watch window displays the value of dFadingPower whenever the control reaches the recently set breakpoint. You will also find that the value of dFadingPower in the Visual Studio Watch window and the value of i in the MATLAB workspace window are similar.



```
/*dFadingPower = propagation_calculate_fadingloss(propagationHandle,
                                                   packet->nTransmitterId,
                                                   ifid,
                                                   pstruEventDetails->nDeviceId,
                                                   pstruEventDetails->nInterfaceI
dFadingPower = fn_netsim_matlab_run();
pdbm -= dFadingPower;  ≤ 1ms elapsed

ber = fn_NetSim_IEEE802_11_CalculateBER(srcPhy,pdbm);

status = fn_NetSim_Packet_DecideError(ber,packet->pstruPhyData->dPacketSize);

if(status == PacketStatus_Error)
{
    if(!isIEEE802_11_CtrlPacket(packet) && is_first_packet(packet))
        phy->firstpacketstatus = PacketStatus_Error;

    // call function to write packet trace and calculate metrics
    packet->pstruPhyData->nPacketErrorFlag = PacketStatus_Error;
    packet->nPacketStatus=PacketStatus_Error;
    fn_NetSim_WritePacketTrace(packet);
    fn_NetSim_Metrics_Add(packet);
    fn_NetSim_Packet_FreePacket(packet);
    goto RET PHYIN;
```

Process: [14024] NetSimCore.exe Thread: [11044] Main Thread

Watch 1

Name	Value	Type
dFadingPower	0.45266572470981425	double

Workspace

Name	Value
h	1x1 WeibullDistribution
i	0.4527

Figure 10-45: The Visual Studio Watch window and the value of i in the MATLAB workspace window are similar.

10.3.2.4 Implement Weibull Distribution of MATLAB in NetSim using .m file:

Procedure:

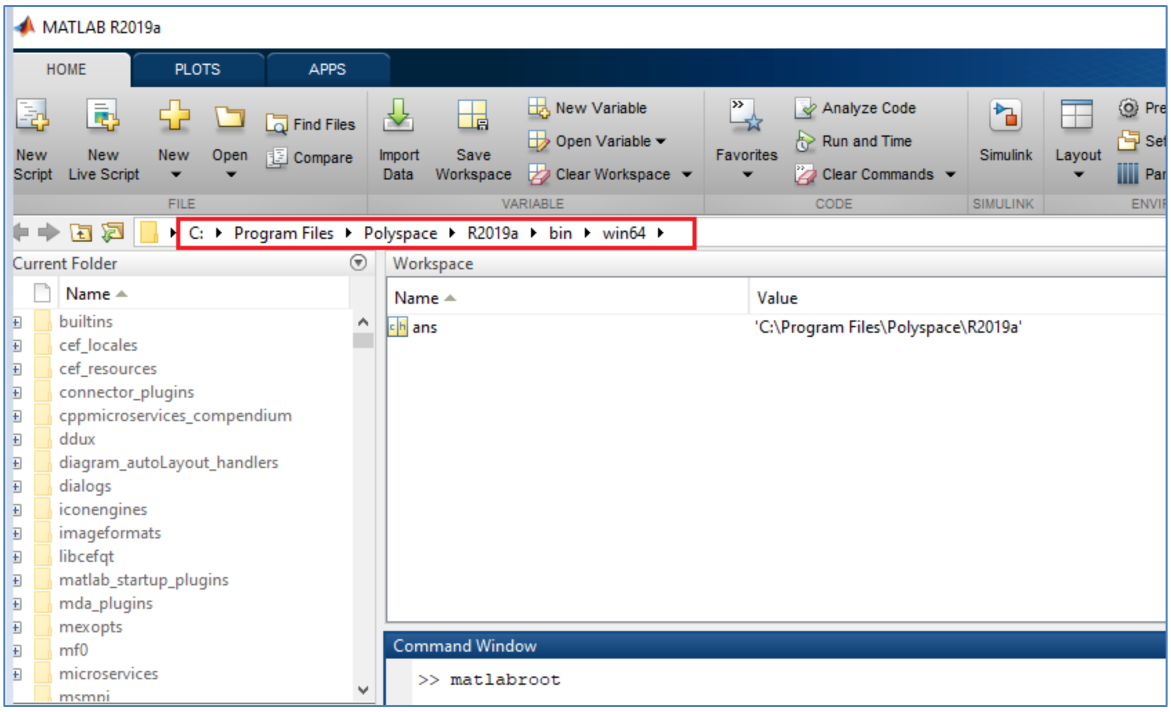
1. Create a file named weibull_distribution.m file inside <Path where MATLAB is installed>. The weibull_distribution.m file contains the following code:

```
function WLAN= weibull_distribution(noncentrality,scale)
%use ProbDistUnivParam() function for matlab 2016

%h=ProbDistUnivParam('weibull',[noncentrality,scale]);
%use makedist() function for matlab 2017

h=makedist('weibull',noncentrality,scale);
i=random(h,1);
WLAN=i;
```

2. Place this file in the MATLAB's default working directory. This will usually be MATLAB's root directory or the bin folder in MATLAB's installation path.

NOTE: *To determine path where MATLAB is installed, entering the following command in the MATLAB command prompt:*

```
matlabroot
```



Name	Value
ans	'C:\Program Files\Polyspace\R2019a'

```
>> matlabroot
```

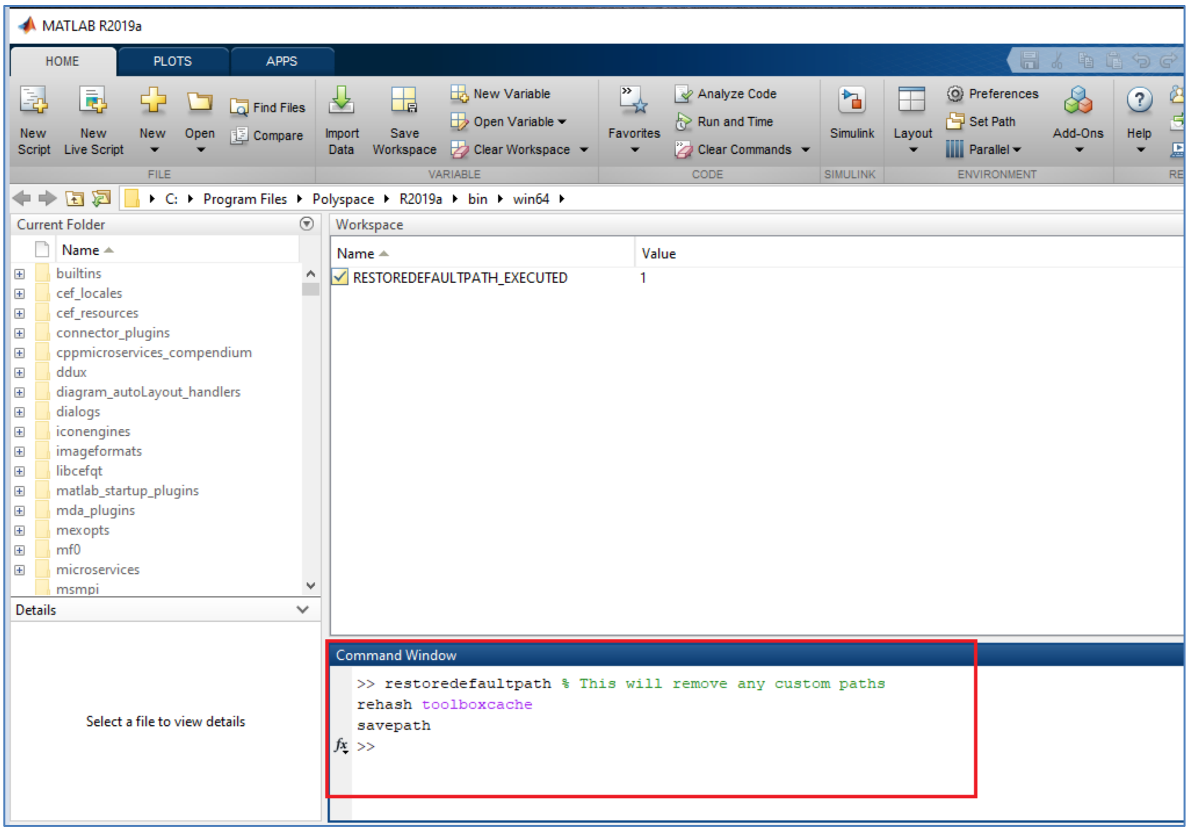
Figure 10-46: MATLAB Command Prompt

- *Execute the below command in MATLAB command window to clear cache.*

  restoredefaultpath % This will remove any custom paths

  rehash toolboxcache

  savepath



Name	Value
RESTOREDEFAULTPATH_EXECUTED	1

```
>> restoredefaultpath % This will remove any custom paths
rehash toolboxcache
savepath
fx >>
```

Figure 10-47: MATLAB command window to clear cache

3. You will have to create a MATLAB_Interface.c file in the IEEE802_11 folder similar to the previous example. The functions **fn_netsim_matlab_init()** and **fn_netsim_matlab_finish()** will remain the same. Modify the function **fn_netsim_matlab_run()** that is part of MATLAB_Interface.c which was used in the previous example as shown below:

```
double fn_netsim_matlab_run()
{
  //write your own implementation here
  int weibull_noncentrality = 1, weibull_scale = 2;
  engPutVariable(ep, "h", h);
  sprintf_s(buf,BUFSIZ, "k=weibull_distribution(%d,%d)", weibull_noncentrality, weibull_scale);
  status = engEvalString(ep, buf);
  out = engGetVariable(ep, "k");
  result = mxGetPr(out);
  return *result;
}
```

4. Follow steps 2 to 17 as explained in the section “Implement Weibull Distribution of MATLAB in NetSim without using .m file” above.
5. A call to the weibull_distribution () function inside the weibull_distribution.m file is made, and weibull_noncentrality and weibull_scale parameters are passed from NetSim.
6. Right Click on IEEE802_11 project and select Rebuild.
7. A new libIEEE802.11.dll will get created in the bin_x64 folder of NetSim’s current workspace path <C:\Users\PC\Documents\NetSim\Workspaces\<Your default workspace>\bin_x64> for 64-bit.

   Open NetSim in Administrative mode. Create a Network scenario involving IEEE802_11 say MANET, right click on the properties, make sure that the Channel Characteristics is set to PathLoss and Fading and Shadowing.
8. You will find that once the Simulation is run MATLAB Command Window starts and gets closed once the Simulation is over. You can also debug the code to understand the communication between NetSim and MATLAB as explained in the DEBUGGING section above.

10.3.2.5 Plot a histogram in MATLAB per a Weibull distribution (using .m file)

Procedure:

1. Create a file NETSIM_MATLAB.m file containing the following code, Place this file in the MATLAB's default working directory. This will usually be MATLAB's root directory or the bin folder in MATLAB's installation path.

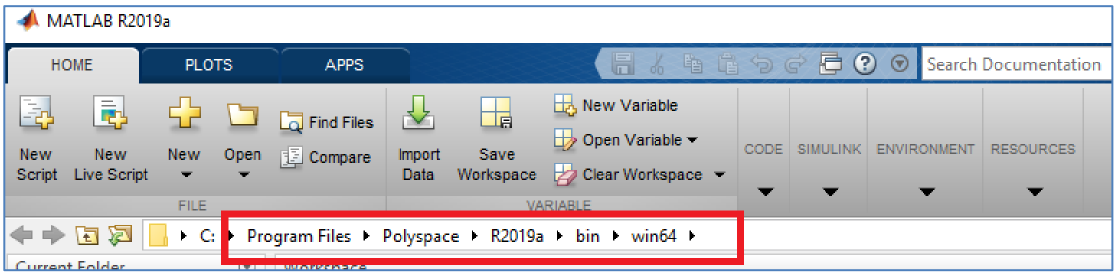


Figure 10-48: MATLAB's default working directory path

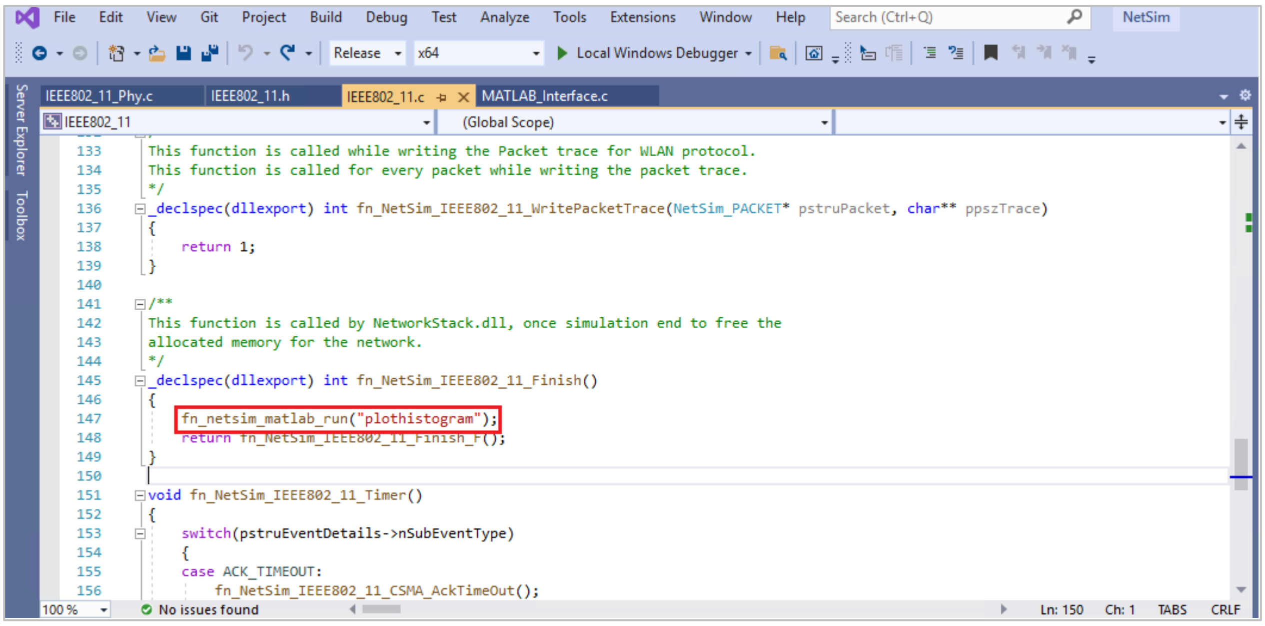
```
function WLAN=NETSIM_MATLAB(choice,varargin)

switch(choice)

        case'weibull'

        %use ProbDistUnivParam function for matlab 2016

        %h=ProbDistUnivParam('weibull',[varargin{1},varargin{2}]);

        %use makedist function for matlab 2017

        h=makedist('weibull',varargin{1}, varargin{2});

   logpath=strcat(varargin{3},'\plotvalues.txt');

        i=random(h,1);

        fid = fopen(logpath,'a+');

        fprintf(fid,'%f',i);

        fprintf(fid,'\r\n');

        fclose('all');

        WLAN=i;

        case'plothistogram'

   logpath=strcat(varargin{1},'\plotvalues.txt');

        fid=fopen(logpath);

        mx=fscanf(fid,'%f');
        hist(mx);
        fclose('all');
    WLAN=0;
end
```

2. Modify the function fn_netsim_matlab_run() that is part of MATLAB_Interface.c which was used in the previous example.

```
double fn_netsim_matlab_run(char* arr)
{
        //write your own implementation here
        int weibull_noncentrality = 1, weibull_scale = 2;
        if (strcmp(arr, "weibull") == 0)
        {
                engPutVariable(ep, "h", h);
                //engPutVariable(ep, "logpath=%s", pszIOLogPath);
                //sprintf_s(buf, BUFSIZ, "logpath=%s", pszIOLogPath);
                //status = engEvalString(ep, buf);
                sprintf_s(buf, BUFSIZ, "h=NETSIM_MATLAB('weibull',%d,%d,'%s')",
                        weibull_noncentrality, weibull_scale, pszIOLogPath);
                status = engEvalString(ep, buf);
                out = engGetVariable(ep, "h");
                result = mxGetPr(out);
                return *result;
        }
        else if (strcmp(arr, "plothistogram") == 0)
        {
                //engPutVariable(ep, "logpath=%s", pszIOLogPath);
                sprintf(buf, "NETSIM_MATLAB('plothistogram','%s')", pszIOLogPath);
                engEvalString(ep, buf);
                status = engEvalString(ep, buf);
                return 0;
        }
        else
                return 0;
}
```

Follow steps 2 to 17 as explained in the section on “Implement weibull Distribution of MATLAB in NetSim without using .m file” above.

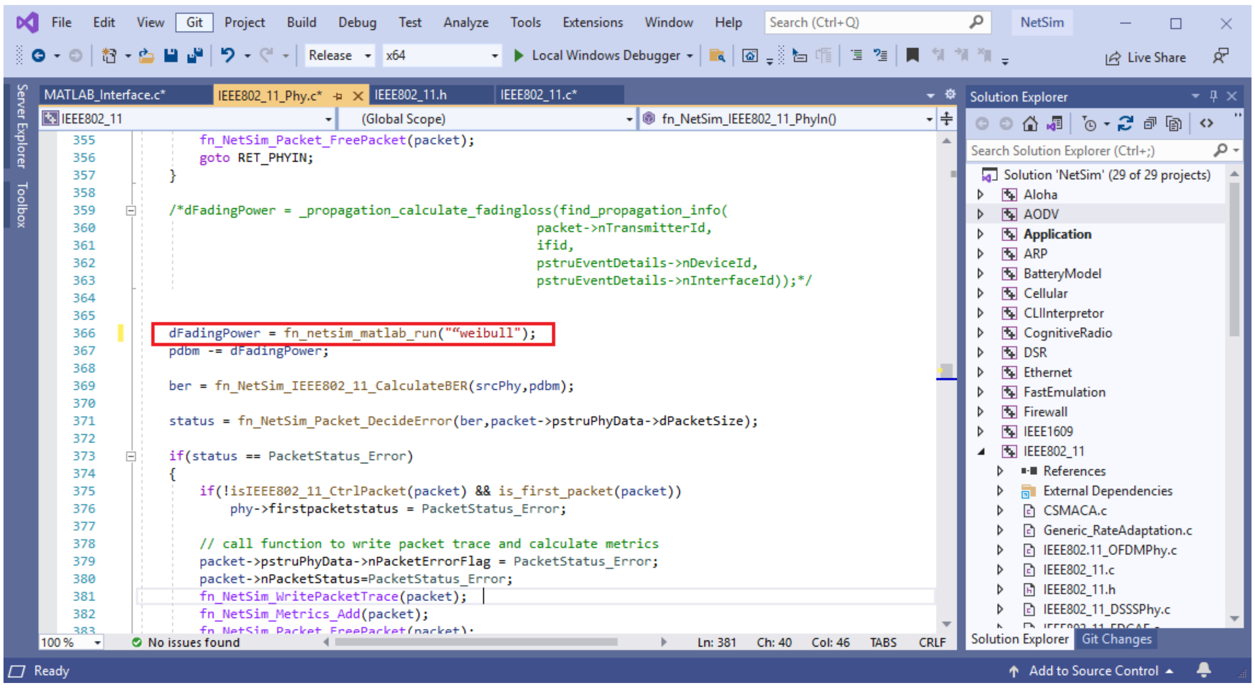
- A call to the NetSim_MATLAB() function inside the NetSim_MATLAB.m file is made, for fading power calculation with parameters distribution(‘weibull’), weibull_noncentrality and weibull_scale parameters are passed from NetSim.
- A call to the NetSim_MATLAB() function inside the NetSim_MATLAB.m file is made, for plotting histogram for the values generated by MATLAB.
- Also add the following call to fn_netsim_matlab_run() function to plot the histogram before closing the MATLAB Engine.



```
This function is called while writing the Packet trace for WLAN protocol.
This function is called for every packet while writing the packet trace.
*/
_declspec(dllexport) int fn_NetSim_IEEE802_11_WritePacketTrace(NetSim_PACKET* pstruPacket, char** ppszTrace)
{
    return 1;
}

/**
This function is called by NetworkStack.dll, once simulation end to free the
allocated memory for the network.
*/
_declspec(dllexport) int fn_NetSim_IEEE802_11_Finish()
{
    fn_netsim_matlab_run("plothistogram");
    return fn_NetSim_IEEE802_11_Finish_F();
}

void fn_NetSim_IEEE802_11_Timer()
{
    switch(pstruEventDetails->nSubEventType)
    {
    case ACK_TIMEOUT:
        fn_NetSim_IEEE802_11_CSMA_AckTimeOut();
```

Figure 10-49: Added the parameter "plothistogram" in fn_netsim_matlab_run() function to get histograms

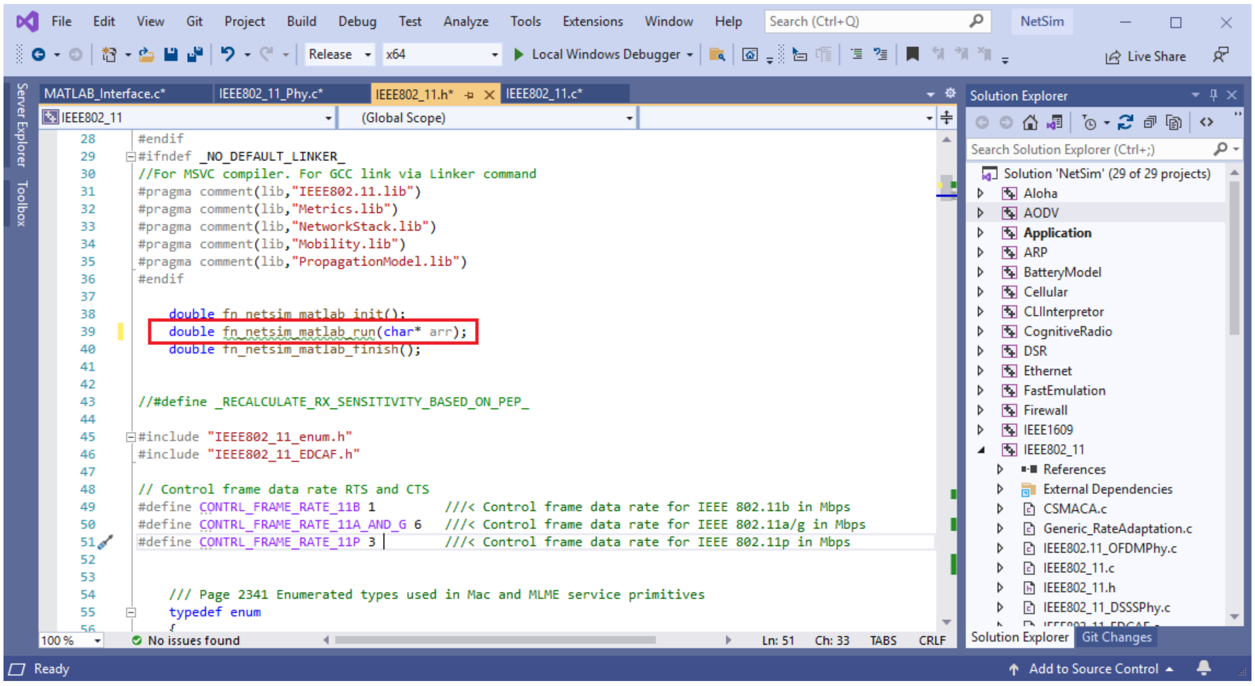
- Similarly in the call made to **fn_netsim_matlab_run()** function in IEEE802_11_Phy.c file add the parameter “weibull” as shown below.



```
            fn_NetSim_Packet_FreePacket(packet);
            goto RET_PHYIN;
        }

        /*dFadingPower = _propagation_calculate_fadingloss(find_propagation_info(
                                                        packet->nTransmitterId,
                                                        ifid,
                                                        pstruEventDetails->nDeviceId,
                                                        pstruEventDetails->nInterfaceId));*/


        dFadingPower = fn_netsim_matlab_run("“weibull");
        pdbm -= dFadingPower;

        ber = fn_NetSim_IEEE802_11_CalculateBER(srcPhy,pdbm);

        status = fn_NetSim_Packet_DecideError(ber,packet->pstruPhyData->dPacketSize);

        if(status == PacketStatus_Error)
        {
            if(!isIEEE802_11_CtrlPacket(packet) && is_first_packet(packet))
                phy->firstpacketstatus = PacketStatus_Error;

            // call function to write packet trace and calculate metrics
            packet->pstruPhyData->nPacketErrorFlag = PacketStatus_Error;
            packet->nPacketStatus=PacketStatus_Error;
            fn_NetSim_WritePacketTrace(packet);
            fn_NetSim_Metrics_Add(packet);
```

Figure 10-50: Added the parameter “weibull” in fn_netsim_matlab_run() function in IEEE802_11_Phy.c file

- Also modify the function definition of **fn_netsim_matlab_run()** function in IEEE802_11.h file as shown below:



```
#endif
#ifndef _NO_DEFAULT_LINKER_
//For MSVC compiler. For GCC link via Linker command
#pragma comment(lib,"IEEE802.11.lib")
#pragma comment(lib,"Metrics.lib")
#pragma comment(lib,"NetworkStack.lib")
#pragma comment(lib,"Mobility.lib")
#pragma comment(lib,"PropagationModel.lib")
#endif

    double fn_netsim_matlab_init();
    double fn_netsim_matlab_run(char* arr);
    double fn_netsim_matlab_finish();


//#define _RECALCULATE_RX_SENSITIVITY_BASED_ON_PEP_

#include "IEEE802_11_enum.h"
#include "IEEE802_11_EDCAF.h"

// Control frame data rate RTS and CTS
#define CONTRL_FRAME_RATE_11B 1        ///< Control frame data rate for IEEE 802.11b in Mbps
#define CONTRL_FRAME_RATE_11A_AND_G 6  ///< Control frame data rate for IEEE 802.11a/g in Mbps
#define CONTRL_FRAME_RATE_11P 3        ///< Control frame data rate for IEEE 802.11p in Mbps


    /// Page 2341 Enumerated types used in Mac and MLME service primitives
    typedef enum
```

Figure 10-51: Modify the Function definition of fn_netsim_matlab_run() function in IEEE802_11.h file

- Right Click on IEEE802_11 project and select Rebuild will create a new libIEEE802.11. in the bin_x64 folder of NetSim’s current workspace path < C:\Users\PC\Documents\NetSim\Workspaces\<Your default workspace>\bin_x64 > for 64-bit.

- Open NetSim in Administrative mode. Create a Network scenario involving IEEE802_11 say MANET, Right click on Adhoc link or Wireless link. Make sure that the Channel Characteristics is set to PathLoss and Fading and Shadowing.
- You will find that once the Simulation is run MATLAB Command Window starts and once the Simulation is over a histogram is displayed in MATLAB for the values that were generated using weibull distribution.

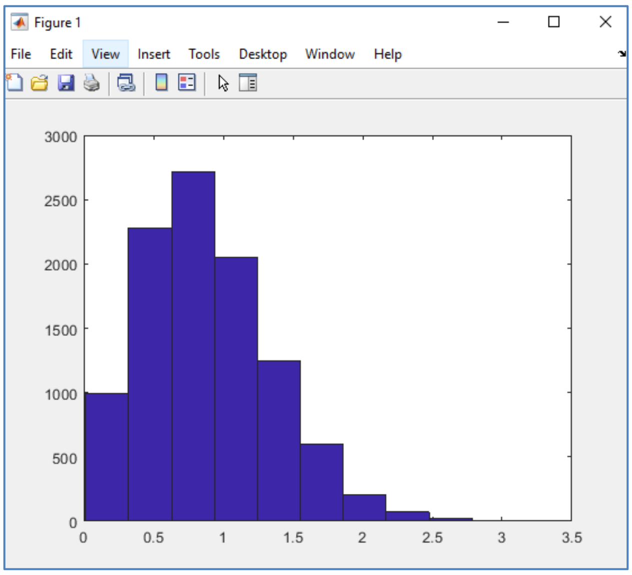


Figure 10-52: Histogram plot is displayed in MATLAB

- The graph and the MATLAB windows get closed once you press any key.

You can also debug the code to understand the communication between NetSim and MATLAB as explained in the DEBUGGING section above.

10.4 Interfacing tail with NetSim

What is a tail command?

The **tail** command is a command-line utility for outputting the last part of files given to it via standard input. It writes results to standard output. By default, tail returns the last ten lines of each file that it is given. It may also be used to follow a file in real-time and watch as new lines are written to it.

PART 1:

Tail options

- The following command is used to log the file.

```
tail " path_to_file " -f
```

where -f option is used to watch a file for changes with the tail command pass the -f option. This will show the last ten lines of a file and will update when new lines are added. This is commonly used to watch log files in real-time. As new lines are written to the log the console will update will new lines.

- If users do not want the last ten lines of the file, then use the following command.

tail -n 0 " path_to_file " –f

where –n option is used to show the last n number of lines.

- If you want to open more than 1 file then use the following command

tail –n 0 " path_to_file " " path_to_file " –f

PART 2:

Steps to log NetSim files using tail console.

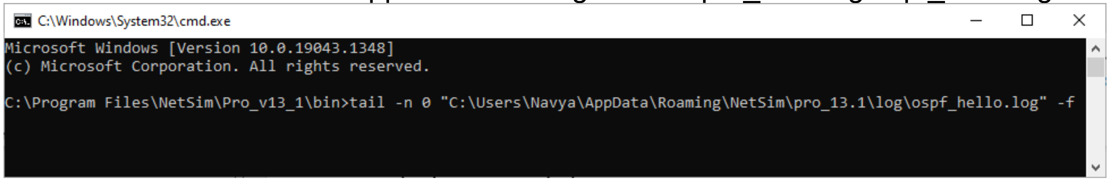
Note: *Before Executing below steps, User need to generate/Simulate a scenario of a network to which tail command is being executed.*

- Open command window from Install directory path of Netsim (<C:\Program Files\NetSim\Pro_v13_1\bin) which contains tail.exe
- Type the following command to open ospf_hello.log.txt file and press enter.

  tail -n 0 "<NetSim_IOPath>\log\ospf_hello.log" -f

  For example,

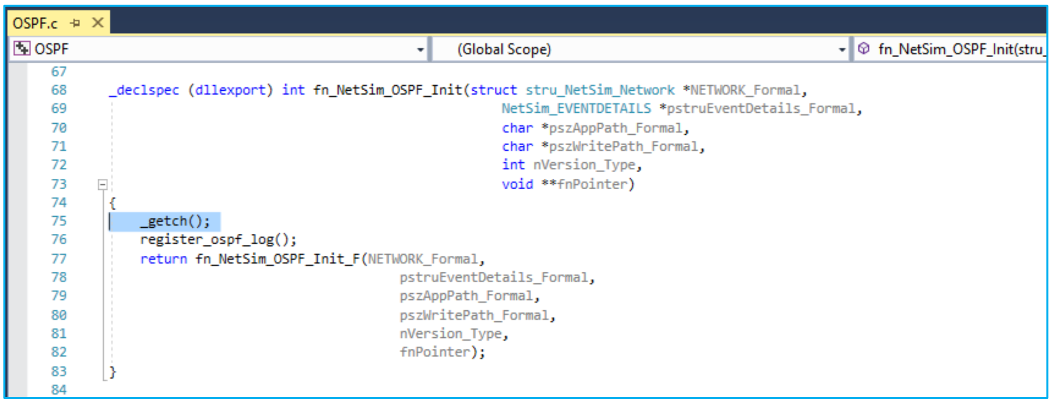
  tail -n 0 "C:\Users\Admin\AppData\Roaming\NetSim\pro_13.1\log\ospf_hello.log" -f



```
Microsoft Windows [Version 10.0.19043.1348]
(c) Microsoft Corporation. All rights reserved.

C:\Program Files\NetSim\Pro_v13_1\bin>tail -n 0 "C:\Users\Navya\AppData\Roaming\NetSim\pro_13.1\log\ospf_hello.log" -f
```

Figure 10-53: Enter the ospf_hello.log.txt file path is Command Prompt

- Open solution file and add the following line in fn_NetSim_OSPF_Init() function in ospf.c file present inside OSPF project



```
_declspec (dllexport) int fn_NetSim_OSPF_Init(struct stru_NetSim_Network *NETWORK_Formal,
                                              NetSim_EVENTDETAILS *pstruEventDetails_Formal,
                                              char *pszAppPath_Formal,
                                              char *pszWritePath_Formal,
                                              int nVersion_Type,
                                              void **fnPointer)
{
    _getch();
    register_ospf_log();
    return fn_NetSim_OSPF_Init_F(NETWORK_Formal,
                                 pstruEventDetails_Formal,
                                 pszAppPath_Formal,
                                 pszWritePath_Formal,
                                 nVersion_Type,
                                 fnPointer);
}
```

Figure 10-54: Add the following line in fn_NetSim_OSPF_Init() function in ospf.c file present inside OSPF project

- Rebuild the project.

- Upon rebuilding, **libOSPF.dll** will get created in the bin folder of NetSim's current workspace path <C:\Users\PC\Documents\NetSim\Workspaces\<Your default workspace>\bin_x64 > for 64-bit
- Create a scenario in NetSim as per the screenshot below and run simulation.

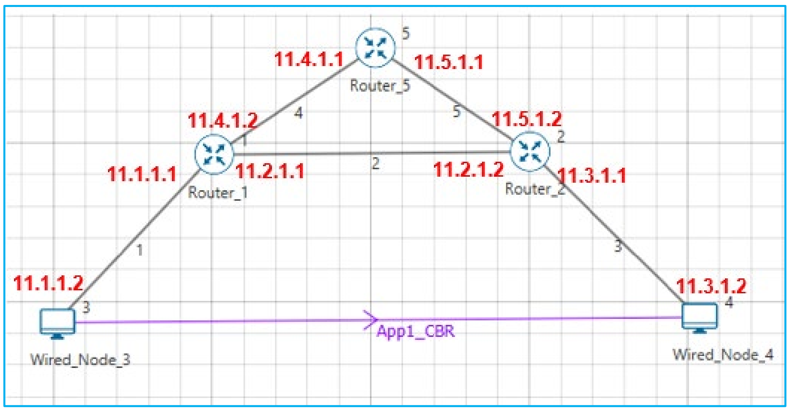
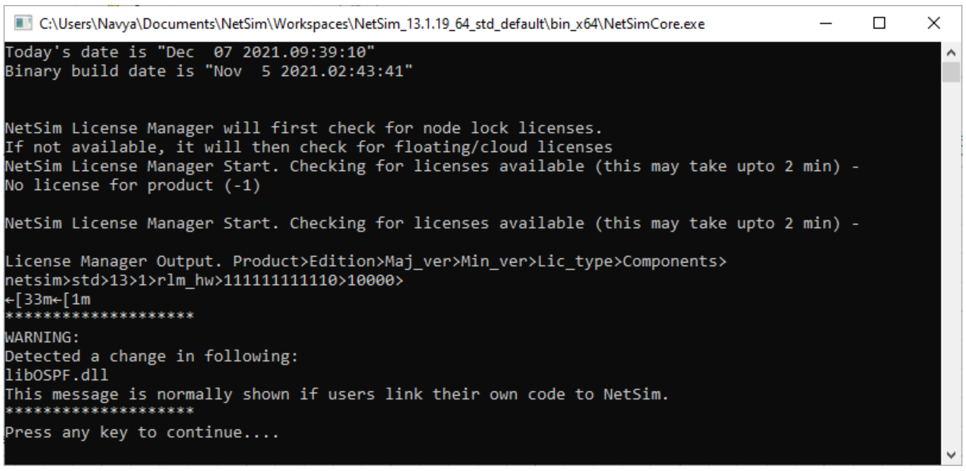


Figure 10-55: Network Topology

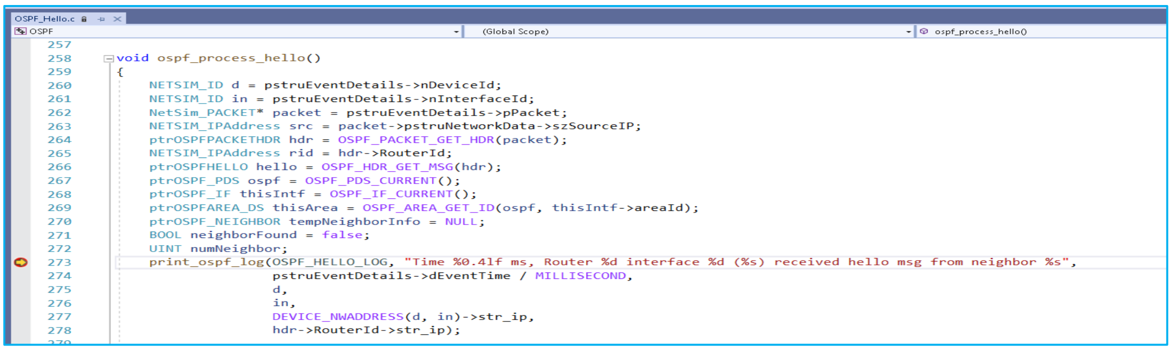
- In the console window user would get a warning message shown in the below screenshot Figure 10-56 (because of changed DLL) and then the simulation will pause for user input (because of _getch() added in the init function)



```
C:\Users\Navya\Documents\NetSim\Workspaces\NetSim_13.1.19_64_std_default\bin_x64\NetSimCore.exe
Today's date is "Dec  07 2021.09:39:10"
Binary build date is "Nov  5 2021.02:43:41"


NetSim License Manager will first check for node lock licenses.
If not available, it will then check for floating/cloud licenses
NetSim License Manager Start. Checking for licenses available (this may take upto 2 min) -
No license for product (-1)

NetSim License Manager Start. Checking for licenses available (this may take upto 2 min) -

License Manager Output. Product>Edition>Maj_ver>Min_ver>Lic_type>Components>
netsim>std>13>1>rlm_hw>111111111110>10000>
←[33m←[1m
********************
WARNING:
Detected a change in following:
libOSPF.dll
This message is normally shown if users link their own code to NetSim.
********************
Press any key to continue....
```

Figure 10-56: Modified Project DLL Warning Message in NetSim Console

- In Visual Studio, put break point inside all the functions in OSPF_Hello.c file present inside OSPF project.
- Go to "Debug->Attach to Process" in Visual studio and attach to NetSimCore.exe.
- Press enter in the command window. Then control goes to the project and stops at the break point in the source code as shown below Figure 10-57.

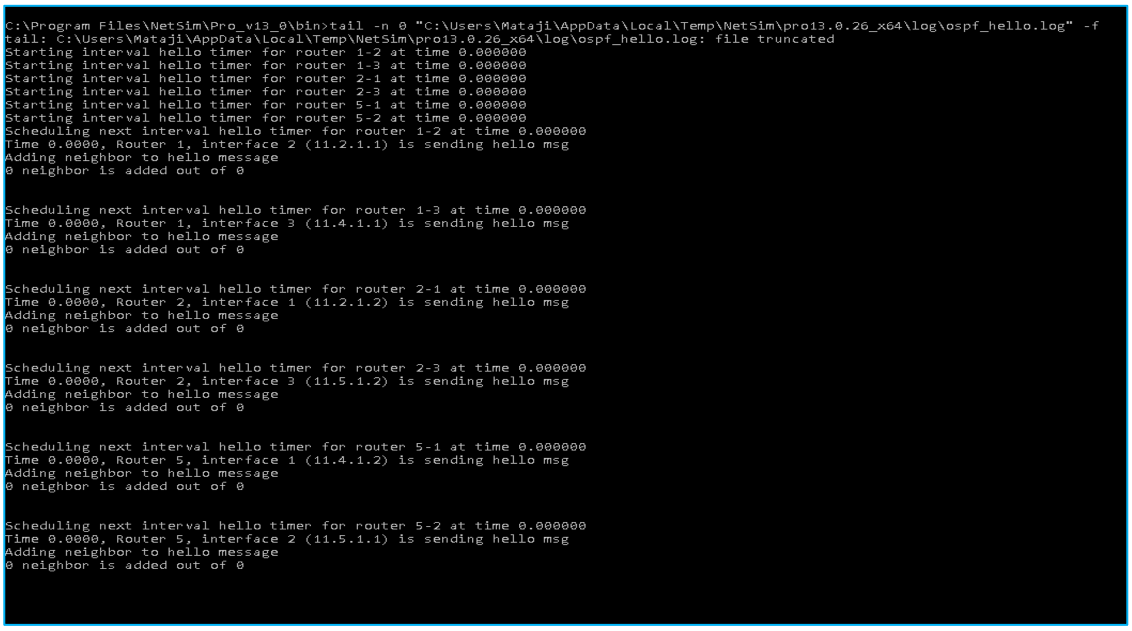


OSPF_Hello.c | OSPF | (Global Scope) | ospf_process_hello()

```c
void ospf_process_hello()
{
    NETSIM_ID d = pstruEventDetails->nDeviceId;
    NETSIM_ID in = pstruEventDetails->nInterfaceId;
    NetSim_PACKET* packet = pstruEventDetails->pPacket;
    NETSIM_IPAddress src = packet->pstruNetworkData->szSourceIP;
    ptrOSPFPACKETHDR hdr = OSPF_PACKET_GET_HDR(packet);
    NETSIM_IPAddress rid = hdr->RouterId;
    ptrOSPFHELLO hello = OSPF_HDR_GET_MSG(hdr);
    ptrOSPF_PDS ospf = OSPF_PDS_CURRENT();
    ptrOSPF_IF thisIntf = OSPF_IF_CURRENT();
    ptrOSPFAREA_DS thisArea = OSPF_AREA_GET_ID(ospf, thisIntf->areaId);
    ptrOSPF_NEIGHBOR tempNeighborInfo = NULL;
    BOOL neighborFound = false;
    UINT numNeighbor;
    print_ospf_log(OSPF_HELLO_LOG, "Time %0.4lf ms, Router %d interface %d (%s) received hello msg from neighbor %s",
                   pstruEventDetails->dEventTime / MILLISECOND,
                   d,
                   in,
                   DEVICE_NWADDRESS(d, in)->str_ip,
                   hdr->RouterId->str_ip);
```

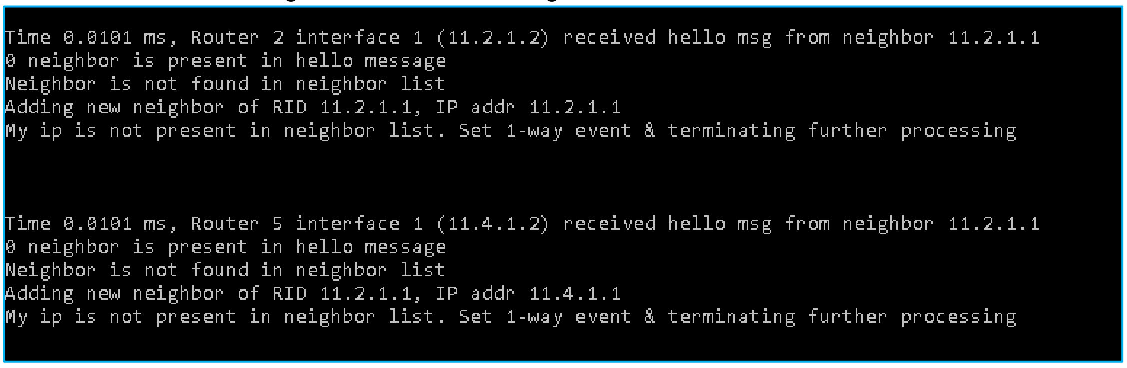
Figure 10-57: Control goes to the project and stops at the break point in the source code.

- Once after pressing enter in command window, check the tail console to watch the ospf_hello.log would look like the following screenshot Figure 10-57.
- Below Screenshot shows that scheduling time of hello interval for each Routers connected in Network.



```
C:\Program Files\NetSim\Pro_v13_0\bin>tail -n 0 "C:\Users\Mataji\AppData\Local\Temp\NetSim\pro13.0.26_x64\log\ospf_hello.log" -f
tail: C:\Users\Mataji\AppData\Local\Temp\NetSim\pro13.0.26_x64\log\ospf_hello.log: file truncated
Starting interval hello timer for router 1-2 at time 0.000000
Starting interval hello timer for router 1-3 at time 0.000000
Starting interval hello timer for router 2-1 at time 0.000000
Starting interval hello timer for router 2-3 at time 0.000000
Starting interval hello timer for router 5-1 at time 0.000000
Starting interval hello timer for router 5-2 at time 0.000000
Scheduling next interval hello timer for router 1-2 at time 0.000000
Time 0.0000, Router 1, interface 2 (11.2.1.1) is sending hello msg
Adding neighbor to hello message
0 neighbor is added out of 0


Scheduling next interval hello timer for router 1-3 at time 0.000000
Time 0.0000, Router 1, interface 3 (11.4.1.1) is sending hello msg
Adding neighbor to hello message
0 neighbor is added out of 0


Scheduling next interval hello timer for router 2-1 at time 0.000000
Time 0.0000, Router 2, interface 1 (11.2.1.2) is sending hello msg
Adding neighbor to hello message
0 neighbor is added out of 0


Scheduling next interval hello timer for router 2-3 at time 0.000000
Time 0.0000, Router 2, interface 3 (11.5.1.2) is sending hello msg
Adding neighbor to hello message
0 neighbor is added out of 0


Scheduling next interval hello timer for router 5-1 at time 0.000000
Time 0.0000, Router 5, interface 1 (11.4.1.2) is sending hello msg
Adding neighbor to hello message
0 neighbor is added out of 0


Scheduling next interval hello timer for router 5-2 at time 0.000000
Time 0.0000, Router 5, interface 2 (11.5.1.1) is sending hello msg
Adding neighbor to hello message
0 neighbor is added out of 0
```

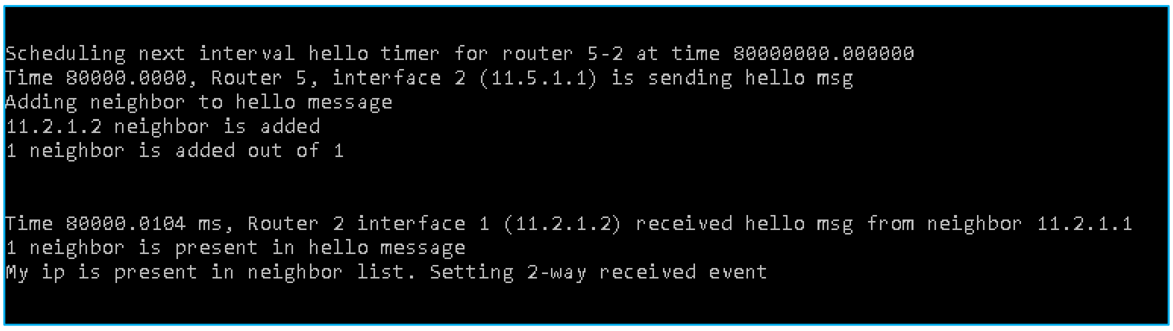
Figure 10-58: Scheduling hello interval for Routers.



```
Time 0.0101 ms, Router 2 interface 1 (11.2.1.2) received hello msg from neighbor 11.2.1.1
0 neighbor is present in hello message
Neighbor is not found in neighbor list
Adding new neighbor of RID 11.2.1.1, IP addr 11.2.1.1
My ip is not present in neighbor list. Set 1-way event & terminating further processing



Time 0.0101 ms, Router 5 interface 1 (11.4.1.2) received hello msg from neighbor 11.2.1.1
0 neighbor is present in hello message
Neighbor is not found in neighbor list
Adding new neighbor of RID 11.2.1.1, IP addr 11.4.1.1
My ip is not present in neighbor list. Set 1-way event & terminating further processing
```

Figure 10-59: Adding new neighbor and terminating process if it is 1-way event

- Above Screenshot shows of adding neighbor to hello message, at Time 0.0101 ms, Router 5 interface 1 (11.4.1.2) received hello msg from neighbor 11.2.1.1.



```
Scheduling next interval hello timer for router 5-2 at time 80000000.000000
Time 80000.0000, Router 5, interface 2 (11.5.1.1) is sending hello msg
Adding neighbor to hello message
11.2.1.2 neighbor is added
1 neighbor is added out of 1


Time 80000.0104 ms, Router 2 interface 1 (11.2.1.2) received hello msg from neighbor 11.2.1.1
1 neighbor is present in hello message
My ip is present in neighbor list. Setting 2-way received event
```

Figure 10-60: Adding Neighbor and Performing 2-way Event if neighbor is present in Router’s list.

Above Screenshot indicates the Router 2’s interface 1 (11.2.1.2) has received hello message from neighbor 11.2.1.1, if neighbor is present in Router’s list 2-way event will be performed.

- Similarly, users can debug the code and observe how the OSPF tables get filled.
- Users can also open multiple files by using the command given in **Part 1.**

10.5 Adding Custom Performance Metrics

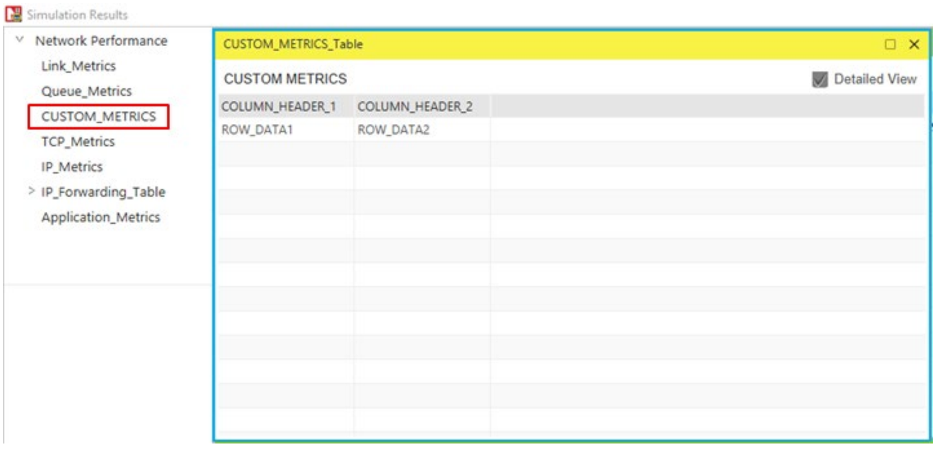
NetSim allows users to add additional metrics tables to the Simulation Results window in addition to the default set of tables that are available at the end of any simulation. This can be achieved by editing the source codes of the respective protocol.

General format to add custom metrics in Result window:

Every protocol has a main C file which contains a Metrics () function. For E.g., TCP project will have a TCP.c file, UDP will have an UDP.c file etc. In the following example we have added a new table as part of TCP protocol. TCP.c file contains fn_NetSim_TCP_Metrics() function where code related to custom metrics is added as shown below:

```
_declspec(dllexport) int fn_NetSim_TCP_Metrics(PMETRICSWRITER metricsWriter)
{
//CUSTOM METRICS
  //Set table name
  PMETRICSNODE table = init_metrics_node(MetricsNode_Table, "CUSTOM METRICS",
  NULL);
  //set table headers
  add_table_heading(table, "COLUMN_HEADER_1", true, 0);
  add_table_heading(table, "COLUMN_HEADER_2", false, 0);
  //Add table data
  add_table_row_formatted(false, table, "%s,%s,", "ROW_DATA1","ROW_DATA2");
  PMETRICSNODE menu = init_metrics_node(MetricsNode_Menu,"CUSTOM_METRICS",
  NULL);
  //Add table to menu
  add_node_to_menu(menu, table);
  //Write to Metrics file
  write_metrics_node(metricsWriter, WriterPosition_Current, NULL, menu);
  delete_metrics_node(menu);
  //CUSTOM METRICS
  return fn_NetSim_TCP_Metrics_F(metricsWriter);
```

}



CUSTOM METRICS

COLUMN_HEADER_1	COLUMN_HEADER_2
ROW_DATA1	ROW_DATA2

Figure 10-61: Added Custom metrics table to the Simulation Results window

For illustration, an example regarding Wireless Sensor Network is provided. In this example, parameters such as Sensor Node Name, Residual Energy, State (On/Off) and turn–off time are tracked and added to a new table in the Simulation Results window.

Refer Section 9.1 on writing your own code, for more information.

After loading the source codes in Visual Studio, perform the following modifications:

Step 1: Copy the provided code at the top in 802_15_4.h file.

```
double NetSim_Off_Time[100]; //Supports upto Device ID 100. Array size can be increased for
higher number of Devices/Device ID's
```

Step 2:

Add the header file in 802_15_4.c file.

```
#include "../BatteryModel/BatteryModel.h"
```

Step 3:

Copy the below code (in red colour) in 802_15_4.c file (inside fn_NetSim_Zigbee_Metrics() function)

```
/** This function write the metrics in metrics.txt */
_declspec(dllexport) int fn_NetSim_Zigbee_Metrics(PMETRICSWRITERmetricsWriter)
 {
//CUSTOM METRICS
   ptrIEEE802_15_4_PHY_VAR phy;
   ptrBATTERY battery;
   char radiostate[BUFSIZ];
   NETSIM_ID nDeviceCount = NETWORK->nDeviceCount;
   //Set table name
```

```
    PMETRICSNODE table = init_metrics_node(MetricsNode_Table,
"NODE_FAILURE_METRICS", NULL);
    //set table headers
    add_table_heading(table, "Node Name", true, 0);
    add_table_heading(table, "Status(ON/OFF)", true, 0);
    add_table_heading(table, "Residual_Energy (mJ)", true, 0);
    add_table_heading(table, "Time - Turned OFF (microseconds)", false, 0);
    for (int i = 1; i <= nDeviceCount; i++)
    {
        sprintf(radiostate, "ON");
        phy = WSN_PHY(i);
        if (strcmp(DEVICE(i)->type, "SENSOR"))
            continue;
        if (WSN_MAC(i)->nNodeStatus == 5 || phy->nRadioState==RX_OFF)
            sprintf(radiostate, "OFF");
        //Add table data
        add_table_row_formatted(false, table, "%s,%s,%.2lf,%.2lf,", DEVICE_NAME(i), radiostate,
battery_get_remaining_energy((ptrBATTERY)phy->battery), NetSim_Off_Time[i]);
    }
    PMETRICSNODE menu = init_metrics_node(MetricsNode_Menu, "CUSTOM_METRICS",
NULL);
    add_node_to_menu(menu, table);
    write_metrics_node(metricsWriter, WriterPosition_Current, NULL, menu);
    delete_metrics_node(menu);
    //CUSTOM METRICS
return fn_NetSim_Zigbee_Metrics_F(metricsWriter);
}
```

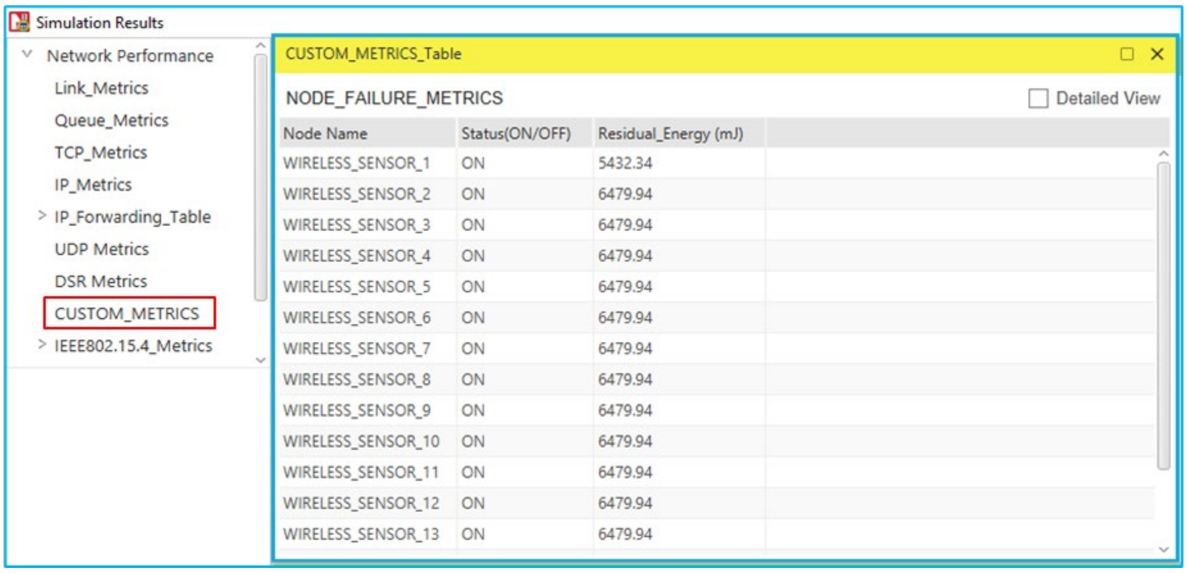
Step 4:

Copy the below code (in red colour) at the end of ChangeRadioState.c file.

```
if(isChange)
        {
                phy->nOldState = nOldState;
                phy->nRadioState = nNewState;
        }
        else
        {
                phy->nRadioState = RX_OFF;
                WSN_MAC(nDeviceId)->nNodeStatus = OFF;
                NetSim_Off_Time[nDeviceId] = ldEventTime;
        }
        return isChange;
        }
```

Step 5:

Build DLL with the modified code and run a Wireless Sensor Network scenario. After Simulation, user will notice a new Performance metrics named “Custom Metrics” is added. The newly added NODE_FAILURE_METRICS table is shown below Figure 10-62.



NODE_FAILURE_METRICS

Node Name	Status(ON/OFF)	Residual_Energy (mJ)
WIRELESS_SENSOR_1	ON	5432.34
WIRELESS_SENSOR_2	ON	6479.94
WIRELESS_SENSOR_3	ON	6479.94
WIRELESS_SENSOR_4	ON	6479.94
WIRELESS_SENSOR_5	ON	6479.94
WIRELESS_SENSOR_6	ON	6479.94
WIRELESS_SENSOR_7	ON	6479.94
WIRELESS_SENSOR_8	ON	6479.94
WIRELESS_SENSOR_9	ON	6479.94
WIRELESS_SENSOR_10	ON	6479.94
WIRELESS_SENSOR_11	ON	6479.94
WIRELESS_SENSOR_12	ON	6479.94
WIRELESS_SENSOR_13	ON	6479.94

Figure 10-62: Added Custom metrics table to the Simulation Results window

10.6 Simulation Time and its relation to Real Time (Wall clock)

The notion of time in a simulation is not directly related to the actual time that it takes to run a simulation (as measured by a wall-clock or the computer's own clock), but is a variable maintained by the simulation program. NetSim uses a virtual clock which ticks virtual time. Virtual time starts from zero progresses as a positive real number.

Time is as a global parameter. All components of the network share the same time throughout the simulation independently of where they are physically located or how they are logically connected to the network. There are not differences among the local time of the communication network components.

This virtual time is referred to as simulation time to clearly distinguish it from real (wall-clock) time. NetSim is a discrete event simulator (DES), and in any DES, the progression of the model over simulation time is decomposed into individual events where change can take place. The flow of time is only between events and is not continuous. Therefore, simulation time is not allowed to progress during an event, but only between events. In fact, the simulation time is always equal to the time at which the current event occurs. Thus, simulation time can be viewed as a variable that "jumps" to track the time specified for each new event.

The answer to the question "Will NetSim run for 10 seconds if Simulation time is set to 10 sec?" is, the simulation may take more than 10 seconds (Wall clock) if the network scenario is very large and heavy traffic load. It may take a much shorter time (wall clock) for small networks with low traffic loads.

Note that when running in "Emulation mode" simulation time and wall clock will be exactly synchronized since it involves the transfer of real packets across the virtual network in NetSim.

In NetSim, the current simulation time can be got using -pstruEventDetails->dEventTime

10.7 Adding Custom Plots

NetSim's plot option can be used to obtain Link and application throughput plots, which can be accessed from the results dashboard after the simulation. In addition, TCP Congestion Window plots and Buffer occupancy plots can be obtained by enabling the respective option in the device properties.

Users can also log additional parameters with respect to time and get them plotted in NetSim results dashboard. Following are some of the API's which are part of NetSim_Plots.h file that can be used for this purpose:

- fn_NetSim_Install_Metrics_Plot() - This function creates a plot log file and returns a value of type PNETSIMPLOT which can be stored in a pointer and be used for adding values using add_plot_data_formatted(). This function can generally be called at simulation start. This function can be called one time for each plot that is to be generated.

  For Eg: If a plot is to be generated for each node. Then this function needs to be called the number of device times.

- add_plot_data_formatted() - This function can be used to add values to the plot log created using the call to fn_NetSim_Install_Metrics_Plot(). This function needs to be called each time you want to add new values to the plot log file. The call should be made at appropriate section of code where the value being plotted changes with time.

10.7.1 Plotting SNR for each UE-gNB pair in 5G NR

SNR measured by UE's from each gNB can be logged and plotted as part of NetSim results window without having to use additional tools. Following is one such example where we log the SNR for each UE-gNB pair and obtain plots at the end of the simulation.

Step 1: Open NetSim source code in NetSim current workspace. For more information, please refer section “4.12 How does a user open and modify source codes”.

Step 2: Go to LTE_NR project through the solution explorer and open the **LTE_NR.c** file. In the function **fn_NetSim_LTE_NR_Init()**, modify code as shown below:

```
_declspec(dllexport) int fn_NetSim_LTE_NR_Init()
{
    //custom plot
    int ret = fn_NetSim_LTE_NR_Init_F();
    for (NETSIM_ID r = 0; r < NETWORK->nDeviceCount; r++)
    {
        for (NETSIM_ID rin = 0; rin < DEVICE(r + 1)->nNumOfInterface; rin++)
        {
            if (!isLTE_NRInterface(r + 1, rin + 1))
                continue;
            ptrLTENR_PROTODATA data = LTENR_PROTODATA_GET(r + 1, rin + 1);
            switch (data->deviceType)
            {
            case LTENR_DEVICETYPE_UE:
            {
                for (NETSIM_ID r1 = 0; r1 < NETWORK->nDeviceCount; r1++)
                {
                    for (NETSIM_ID rin1 = 0; rin1 < DEVICE(r1 + 1)->nNumOfInterface; rin1++)
                    {
                        if (!isLTE_NRInterface(r1 + 1, rin1 + 1))
                            continue;
                        ptrLTENR_PROTODATA data = LTENR_PROTODATA_GET(r1 + 1, rin1 + 1);
                        switch (data->deviceType)
                        {
                        case LTENR_DEVICETYPE_GNB:
                        {
```

```
                    char heading[BUFSIZ], plotname[BUFSIZ];

                    sprintf(heading, "UE_%d_GNB_%d_SNR", r + 1, r1 + 1);

                    sprintf(plotname, "plot_UE_%d_GNB_%d_SNR", r + 1, r1 + 1);

                    fn_NetSim_Install_Metrics_Plot(Plot_Custom, "LTE_NR SNR Plot", heading,
"SNR(dB)", 1, plotname);

                }

                break;

                default:

                    break;

                }

            }

        }

        break;

    default:

        break;

    }

    break;

    }

  }

 }

 return ret;

 //custom plot

}
```

Step 3: In the file LTENR_GNBRRC.c go to the function fn_NetSim_LTENR_RRC_GENERATE_UE_MEASUREMENT_REPORT() and add the lines of code highlighted in red as shown below:

```
void fn_NetSIM_LTENR_RRC_GENERATE_UE_MEASUREMENT_REPORT()

{
 NETSIM_ID d = pstruEventDetails->nDeviceId;

 NETSIM_ID in = pstruEventDetails->nInterfaceId;
```

```
    ptrLTENR_UERRC ueRRC = LTENR_UERRC_GET(d, in);

    ptrLTENR_GNBRRC gnbRRC = LTENR_GNBRRC_GET(ueRRC->SelectedCellID, ueRRC-
>SelectedCellIF);

    ptrLTENR_RRC_UE_MEASUREMENT_REPORT report = NULL;

    ptrLTENR_RRC_UE_MEASUREMENT_REPORT temp = NULL;

    ptrLTENR_GNBPHY phy = NULL;

    for (NETSIM_ID r = 0; r < NETWORK->nDeviceCount; r++)

    {

        for (NETSIM_ID rin = 0; rin < DEVICE(r + 1)->nNumOfInterface; rin++)

        {

            if (!isLTE_NRInterface(r + 1, rin + 1))

                continue;

            ptrLTENR_PROTODATA data = LTENR_PROTODATA_GET(r + 1, rin + 1);

            switch (data->deviceType)

            {

            case LTENR_DEVICETYPE_GNB:

                temp = MEASUREMENT_REPORT_ALLOC();

                temp->ueID = d;

                temp->cellID = r + 1;

                temp->cellIF = rin + 1;
                temp->rs_type = RS_TYPE_SSB;

                temp->reportAmount = ReportAmount_r1;

                temp->reportInteval = gnbRRC->ueMeasReportInterval;

                phy = LTENR_GNBPHY_GET(r + 1, rin + 1);

                temp->sinr = LTENR_PHY_RRC_RSRP_SINR(r + 1, rin + 1, d, in);

                //custom plot

                char plotname[BUFSIZ];

                sprintf(plotname, "%s\\plot_UE_%d_GNB_%d_SNR.txt",pszIOPath, d, r + 1);

                FILE* fp = fopen(plotname, "a+");

                if (fp)

                {

                    fprintf(fp, "%lf,%lf\n",pstruEventDetails->dEventTime,temp->sinr);

                    fclose(fp);

                }

                //custom plot

                LIST_ADD_LAST((void**)&report, temp);

                break;

            default:

                break;

            }
```

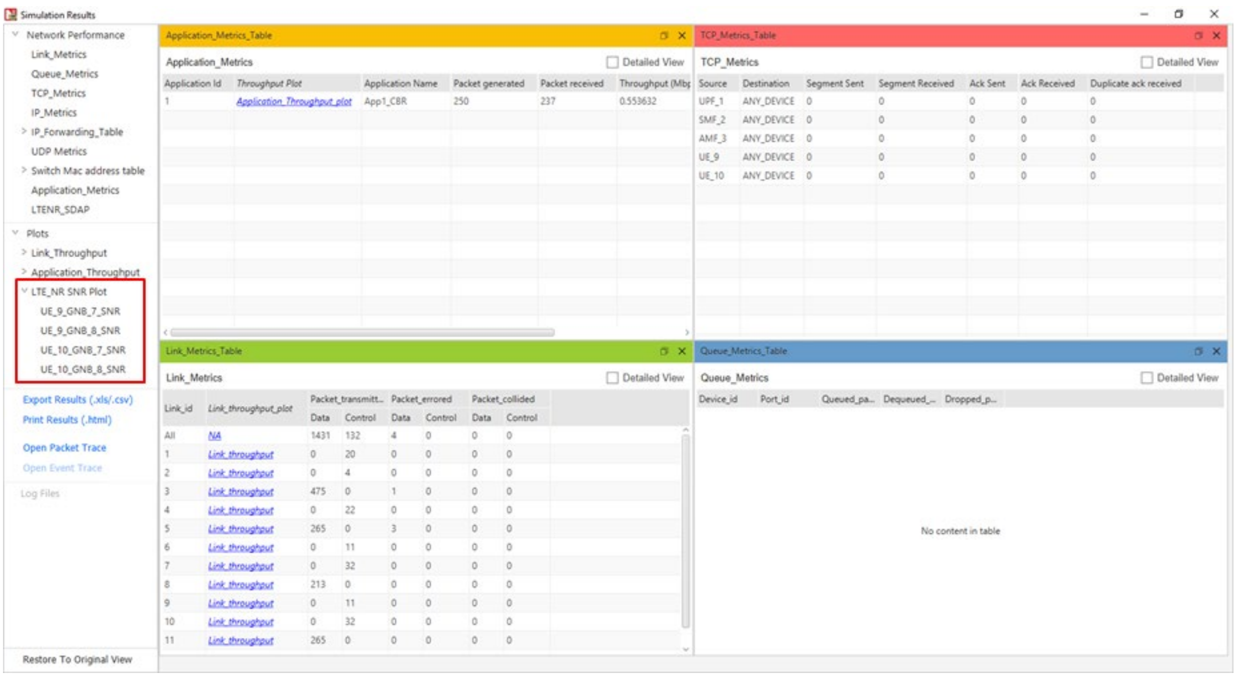
```
        }

    }

    ptrLTENR_RRC_Hdr hdr = calloc(1, sizeof * hdr);

    hdr->msg = report;

    hdr->msgType = LTENR_MSG_RRC_UE_MEASUREMENT_REPORT;

    hdr->SenderID = d;

    hdr->SenderIF = in;

    fn_NetSIm_LTENR_RRC_ADD_HDR_INTO_PACKET(pstruEventDetails->pPacket, hdr, ueMEASID,
LTENR_MSG_RRC_UE_MEASUREMENT_REPORT);

    LTENR_CallPDCPOut();

}
```

Step 4: In **LTENR.c** add the following lines **#include "NetSim_Plot.h".**

Step 5: Save the changes and right-click on the LTE_NR module in the solution explorer and select Rebuild.

Step 6: Upon a successful build, NetSim will automatically update the modified binaries in the respective binary folder.

Step 7: Now on running any simulation in LTE/5G NR networks, you will get individual SNR plots for each UE-GNB/UE-ENB pair, in the NetSim Metrics window under Plots ->LTE_NR SNR Plot shown below Figure 10-63.



Application_Metrics_Table

Application_Metrics ☐ Detailed View

Application Id	Throughput Plot	Application Name	Packet generated	Packet received	Throughput (Mbp
1	Application_Throughput_plot	App1_CBR	250	237	0.553632

TCP_Metrics_Table

TCP_Metrics ☐ Detailed View

Source	Destination	Segment Sent	Segment Received	Ack Sent	Ack Received	Duplicate ack received
UPF_1	ANY_DEVICE	0	0	0	0	0
SMF_2	ANY_DEVICE	0	0	0	0	0
AMF_3	ANY_DEVICE	0	0	0	0	0
UE_9	ANY_DEVICE	0	0	0	0	0
UE_10	ANY_DEVICE	0	0	0	0	0

Link_Metrics_Table

Link_Metrics ☐ Detailed View

Link_id	Link_throughput_plot	Packet_transmitt...		Packet_errored		Packet_collided	
		Data	Control	Data	Control	Data	Control
All	NA	1431	132	4	0	0	0
1	Link_throughput	0	20	0	0	0	0
2	Link_throughput	0	4	0	0	0	0
3	Link_throughput	475	0	1	0	0	0
4	Link_throughput	0	22	0	0	0	0
5	Link_throughput	265	0	3	0	0	0
6	Link_throughput	0	11	0	0	0	0
7	Link_throughput	0	32	0	0	0	0
8	Link_throughput	213	0	0	0	0	0
9	Link_throughput	0	11	0	0	0	0
10	Link_throughput	0	32	0	0	0	0
11	Link_throughput	265	0	0	0	0	0

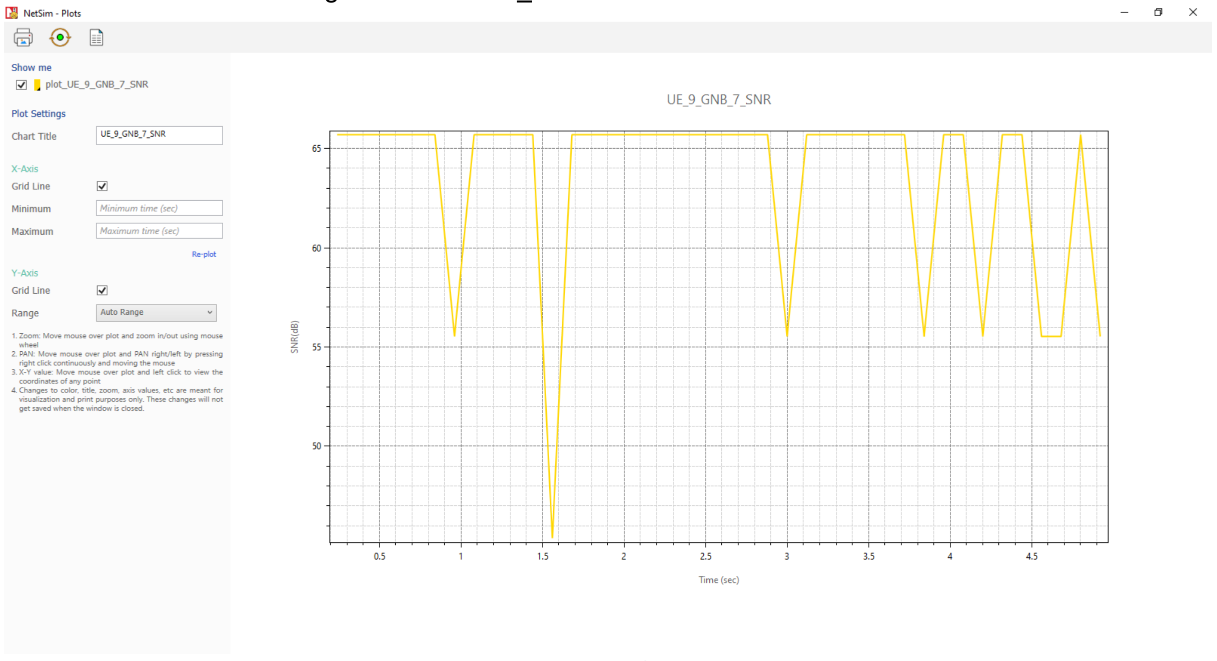
Queue_Metrics_Table

Queue_Metrics ☐ Detailed View

Device_id	Port_id	Queued_pa...	Dequeued_...	Dropped_p...

No content in table

Figure 10-63: LTE_NR SNR Plots in Result Window



☑ plot_UE_9_GNB_7_SNR

Chart Title: UE_9_GNB_7_SNR

X-Axis

Grid Line ☑

Minimum: Minimum time (sec)

Maximum: Maximum time (sec)

Y-Axis

Grid Line ☑

Range: Auto Range

1. Zoom: Move mouse over plot and zoom in/out using mouse wheel
2. PAN: Move mouse over plot and PAN right/left by pressing right click continuously and moving the mouse
3. X-Y value: Move mouse over plot and left click to view the coordinates of any point
4. Changes to color, title, zoom, axis values, etc are meant for visualization and print purposes only. These changes will not get saved when the window is closed.

Figure 10-64: SNR Plot

The above results are based on the Handover in 5G NR Experiment which is part of NetSim v13.1 experiment manual. The plot shows how the SNR drops as UE 3 moves away from GNB 1.

10.8 Environment Variables in NetSim

1. NETSIM_PACKET_FILTER = <filter_string> //used by NetSim developers to debug. Emulator code to passes filter string to   windivert. See windivert doc for more information.
2. NETSIM_EMULATOR_LOG = <log_file_path> // Used by Real time sync function to log get event and add event. Used by NetSim developers to debug.

3. NETSIM_EMULATOR = 1 // Set by application dll or user to notify NetSim internal modules to run in emulation mode
4. NETSIM_CUSTOM_EMULATOR = 1 // To notify NetworkStack to not load emulation dll and to only do time sync.
5. NETSIM_SIM_AREA_X = <int> // Area used by Mobility functions for movement of device. Set by config file parser or user.
6. NETSIM_SIM_AREA_Y = <int> // Same as above
7. NETSIM_ERROR_MODE = 1 // if set then windows won't popup gui screen for error reporting on exception.
8. NETSIM_BREAK = <int> // Event id at which simulation will break and wait for user input.

Equivalent to -d command in CLI mode.

9. NETSIM_AUTO = <int> // If set NetSim will not ask for keypress after simulation. //Useful to run batch simulations.
10. NETSIM_IO_PATH = <path> // IO path of NetSim from where it will read Config file and write output file. Equivalent to -IOPATH command in CLI mode.
11. NETSIM_MAP = 1 // Set by Networkstack to inform other modules that simulation is running per map view.
12. NETSIM_ADVANCE_METRIC // If set, NetSim provides a set of extra metrics.
    In application metrics, you can see duplicate packets received.
13. NETSIM_CONFIG_NAME = <FILE NAME> // Config file name. This file must present in IOPath. If not set default value is Configuration.netsim
14. NETSIM_NEG_ID = 1 // If set, then control packets will have negative id.
15. NETSIM_PACKET_DBG = 1 // If set, then Simulation engine will log the packet creation and freeing
16. NETSIM_MEMCHECK = 1 // If set, then simulation will enable memory check.
17. NETSIM_MEMCHECK_1 = x   // Lower event id
18. NETSIM_MEMCHECK_2 = x   // Upper event id

10.9 Best practices for running large scale simulations

As we scale simulations, the number of events processed and the memory consumed increase. Simulation scale can be defined in terms of:

1. Number of Nodes
   - End nodes
   - Intermediate devices
2. Total traffic in the network

- Number of traffic sources
- Average generation rate per source

3. Simulation time

The simulators performance is additionally affected by:

- Protocols running
- Network Parameters such as Topology, Mobility, Wireless Channel etc.
- Enabling/Disabling - Animation, Plots, Traces and Logs
- External Interfacing – MATLAB, Wireshark, SUMO

NetSim GUI limitation on total number of Nodes is as follows:

- NetSim Academic – 100
- NetSim Standard – 500
- NetSim Professional – No software limit

Recommended best practices for running large scale simulations are:

- Run 64-bit Build of NetSim
- Use the latest Windows 10 Build.
- Use a system running a high-end processor with minimum 32 GB RAM
- Disable Animation – NetSim writes one file per node and windows limits the number of simultaneously opened files to 512.
- Disable plots, traces and logs to speed up the simulation.
- If plots are enabled NetSim writes one file for each link and for each application. Therefore, it is recommended users only select those links/applications for which they wish to plot output performance. Plots for all other applications/links should be disabled.
- NetSim writes one packet trace and one event trace file per simulation. If users wish to open this file in MS-Excel, please note Excel's limit of 1 Million rows.
- Packet trace and Event trace can be disabled to speed up the simulation.
- Running simulations via CLI mode will save memory.

10.10 Batch experimentation and automated simulations

NetSim Batch Automation allows users to execute a series of simulations without manual intervention. Consider a requirement, where a user wishes to create and simulate hundreds of network scenarios and store and analyse the performance metrics of each simulation run. It is impossible to do this manually using the GUI. This requirement can be met using NetSim Batch Automation script which runs NetSim via CLI.

A related requirement of advanced simulation users is a *multi-parameter sweep*. When you *sweep* one or more parameters, you change their values between simulation runs, and compare and analyze the performance metrics from each run.

The documentation and codes for batch experimentation script is available TETCOS - https://www.tetcos.com/file-exchange.html

11 NetSim Emulator

NOTE: *Emulator will be featured in NetSim only if license for Emulator Add-on is available*

11.1 Introduction

A network simulator mimics the behavior of networks but cannot connect to real networks. NetSim Emulator enables users to connect NetSim simulator to real hardware and interact with live applications.

11.1.1 Simulating and Analyzing Emulation Examples

To simulate the different types of Emulations Examples such as PING (both one-way and two-way communications), Video (one-way communication), File transfer using FileZilla, Skype etc.

1. Refer to the Emulation Technology Library document, which explains the following:
   i. Introduction to Emulation.
   ii. How to set up and configure Emulation Server in NetSim
   iii. NetSim Emulation Features with added examples
   iv. Latest FAQs
2. To access the Emulation Technology Library document,
   i. You can access from the Technology Libraries link present under Documentation in NetSim Homescreen
   ii. From the Help Menu inside the design window, choose Technology Libraries Manuals → Emulation.

12 Troubleshooting in NetSim

This section discusses some common issues and solutions:

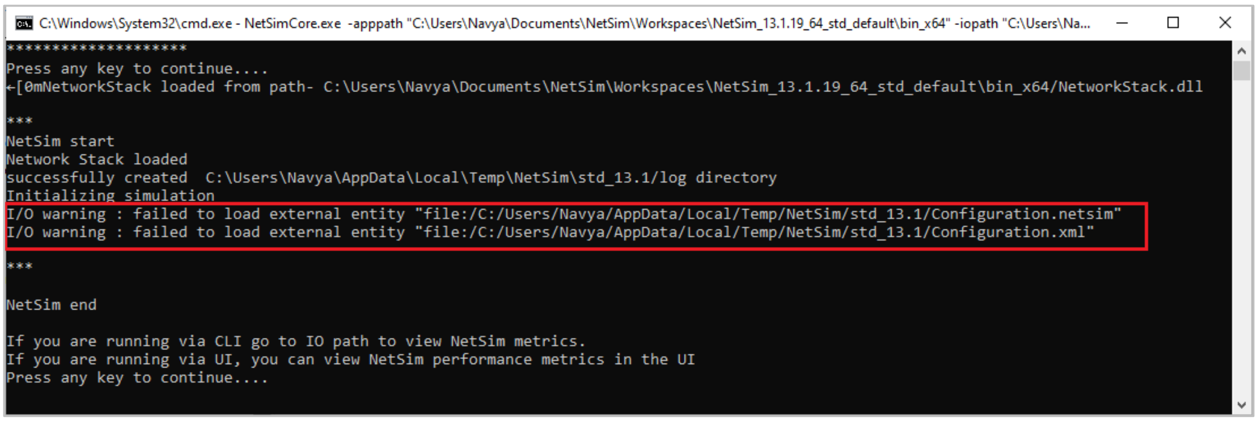
12.1 CLI mode

While running NetSim via CLI for the scenarios described in the Configuration file, you may bump into a few problems. While running NetSim via CLI, try to ensure that there are no errors in the Configuration.netsim file. The ConfigLog.txt file written to the windows temp path would show errors, if any, found by NetSim's config parser.

12.2 Warnings when running CLI mode

12.2.1 I/O warning

Reason: While typing the CLI command if you enter the wrong I/O Path, or if there is no Configuration.netsim file then the following error is thrown



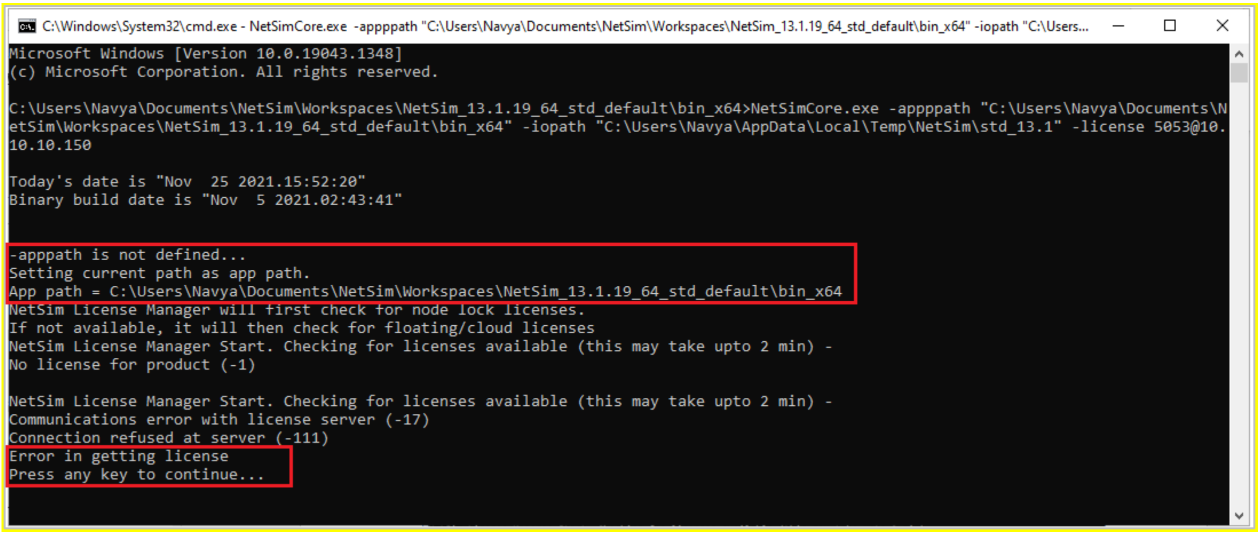
```
C:\Windows\System32\cmd.exe - NetSimCore.exe  -apppath "C:\Users\Navya\Documents\NetSim\Workspaces\NetSim_13.1.19_64_std_default\bin_x64" -iopath "C:\Users\Na...
********************
Press any key to continue....
←[0mNetworkStack loaded from path- C:\Users\Navya\Documents\NetSim\Workspaces\NetSim_13.1.19_64_std_default\bin_x64/NetworkStack.dll

***
NetSim start
Network Stack loaded
successfully created  C:\Users\Navya\AppData\Local\Temp\NetSim\std_13.1/log directory
Initializing simulation
I/O warning : failed to load external entity "file:/C:/Users/Navya/AppData/Local/Temp/NetSim/std_13.1/Configuration.netsim"
I/O warning : failed to load external entity "file:/C:/Users/Navya/AppData/Local/Temp/NetSim/std_13.1/Configuration.xml"

***

NetSim end

If you are running via CLI go to IO path to view NetSim metrics.
If you are running via UI, you can view NetSim performance metrics in the UI
Press any key to continue....
```

Figure 12-1: I/O warning displayed in CLI mode

Solution*:* Check and correct the I/O path.

12.2.2 Error in getting License displayed

Reason: Unable to communicate with the license server



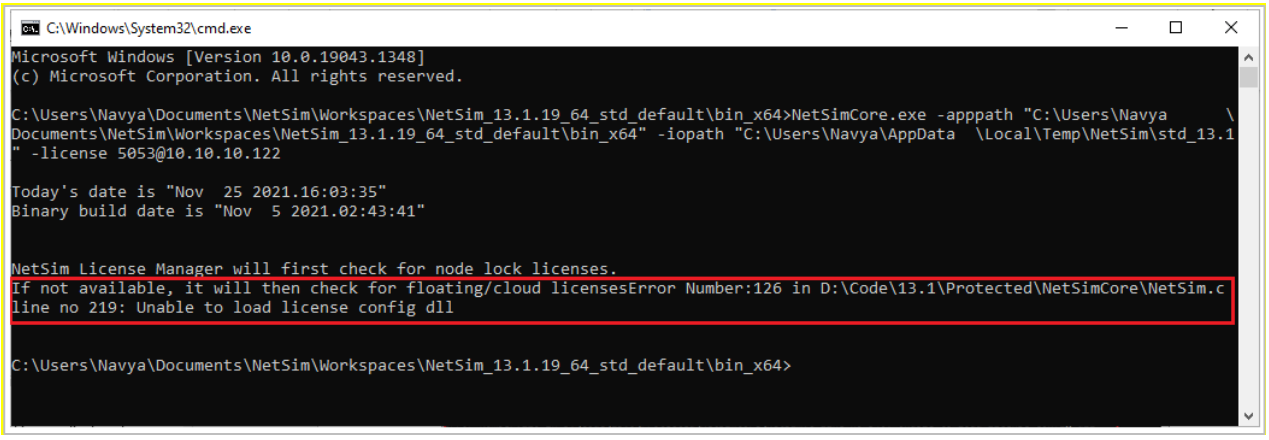
```
C:\Windows\System32\cmd.exe - NetSimCore.exe -appppath "C:\Users\Navya\Documents\NetSim\Workspaces\NetSim_13.1.19_64_std_default\bin_x64" -iopath "C:\Users...
Microsoft Windows [Version 10.0.19043.1348]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Navya\Documents\NetSim\Workspaces\NetSim_13.1.19_64_std_default\bin_x64>NetSimCore.exe -appppath "C:\Users\Navya\Documents\N
etSim\Workspaces\NetSim_13.1.19_64_std_default\bin_x64" -iopath "C:\Users\Navya\AppData\Local\Temp\NetSim\std_13.1" -license 5053@10.
10.10.150

Today's date is "Nov  25 2021.15:52:20"
Binary build date is "Nov  5 2021.02:43:41"


-apppath is not defined...
Setting current path as app path.
App path = C:\Users\Navya\Documents\NetSim\Workspaces\NetSim_13.1.19_64_std_default\bin_x64
NetSim License Manager will first check for node lock licenses.
If not available, it will then check for floating/cloud licenses
NetSim License Manager Start. Checking for licenses available (this may take upto 2 min) -
No license for product (-1)

NetSim License Manager Start. Checking for licenses available (this may take upto 2 min) -
Communications error with license server (-17)
Connection refused at server (-111)
Error in getting license
Press any key to continue...
```

Figure 12-2: Error in getting license

Solution: In this example, license server IP address is 10.10.10.122 but it is entered as 10.10.10.150. This is a case where the server IP address is wrong. The same error message is shown for wrong port number, wrong tag name like–apppath, -iopath, -license …, etc. Therefore, this message is also shown if –appppath is typed instead of –apppath. Users are advised to check the command line arguments carefully.

12.2.3 Unable to load license config dll(126)

Reason: Apppath and I/O path have white spaces



```
C:\Windows\System32\cmd.exe
Microsoft Windows [Version 10.0.19043.1348]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Navya\Documents\NetSim\Workspaces\NetSim_13.1.19_64_std_default\bin_x64>NetSimCore.exe -apppath "C:\Users\Navya      \
Documents\NetSim\Workspaces\NetSim_13.1.19_64_std_default\bin_x64" -iopath "C:\Users\Navya\AppData  \Local\Temp\NetSim\std_13.1
" -license 5053@10.10.10.122

Today's date is "Nov  25 2021.16:03:35"
Binary build date is "Nov  5 2021.02:43:41"


NetSim License Manager will first check for node lock licenses.
If not available, it will then check for floating/cloud licensesError Number:126 in D:\Code\13.1\Protected\NetSimCore\NetSim.c
line no 219: Unable to load license config dll


C:\Users\Navya\Documents\NetSim\Workspaces\NetSim_13.1.19_64_std_default\bin_x64>
```

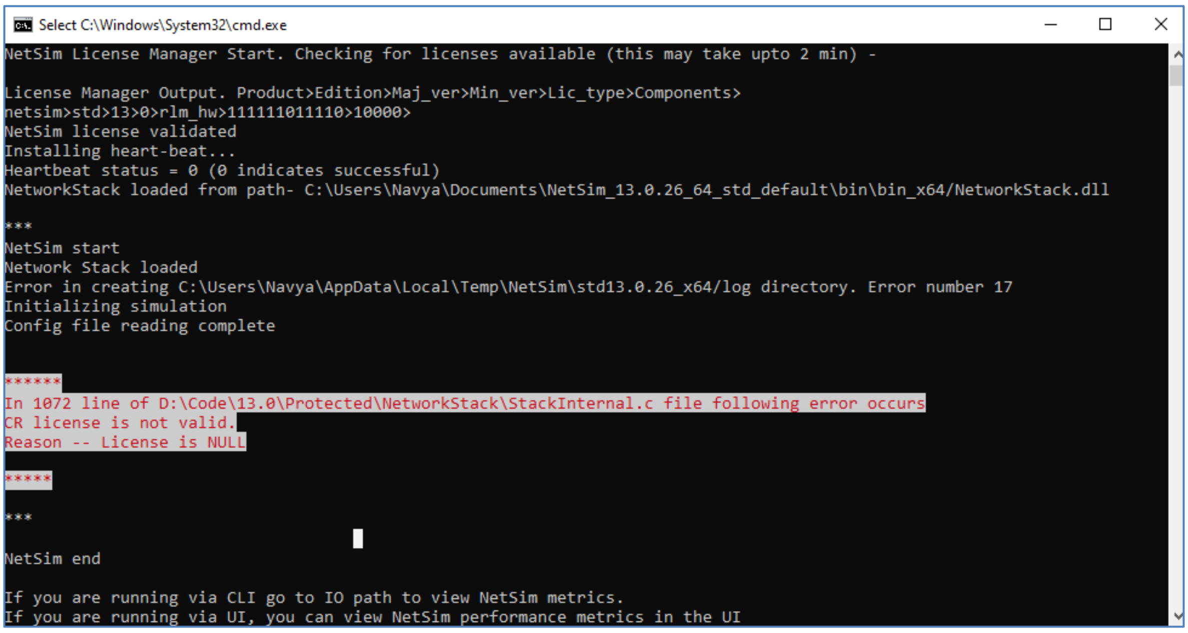
Figure 12-3: Unable to load license config dll

Solution*:* 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.

12.2.4 “License is NULL” error in CLI mode

Simulation does not commence. “License is NULL" is displayed in the command prompt.

Example:



```
Select C:\Windows\System32\cmd.exe
NetSim License Manager Start. Checking for licenses available (this may take upto 2 min) -

License Manager Output. Product>Edition>Maj_ver>Min_ver>Lic_type>Components>
netsim>std>13>0>rlm_hw>111111011110>10000>
NetSim license validated
Installing heart-beat...
Heartbeat status = 0 (0 indicates successful)
NetworkStack loaded from path- C:\Users\Navya\Documents\NetSim_13.0.26_64_std_default\bin\bin_x64/NetworkStack.dll

***
NetSim start
Network Stack loaded
Error in creating C:\Users\Navya\AppData\Local\Temp\NetSim\std13.0.26_x64/log directory. Error number 17
Initializing simulation
Config file reading complete


******
In 1072 line of D:\Code\13.0\Protected\NetworkStack\StackInternal.c file following error occurs
CR license is not valid.
Reason -- License is NULL

*****

***


NetSim end

If you are running via CLI go to IO path to view NetSim metrics.
If you are running via UI, you can view NetSim performance metrics in the UI
```

Figure 12-4: “License is NULL” error in CLI mode

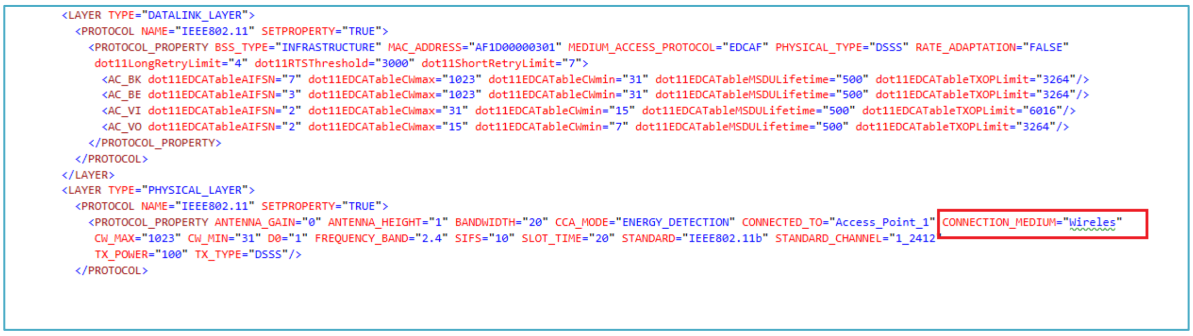
Solution: This indicates unavailability of licenses for the network (component) being simulated.

12.3 Configuration.netsim

12.3.1 Invalid attribute in configuration file attributes

Specific attributes in the Configuration file are highlighted with zigzag lines.

***Reason**:* If invalid input is given in the Configuration file, then the corresponding attribute is highlighted as blue lines as shown in the figure given below Figure 12-5.



```
<LAYER TYPE="DATALINK_LAYER">
  <PROTOCOL NAME="IEEE802.11" SETPROPERTY="TRUE">
    <PROTOCOL_PROPERTY BSS_TYPE="INFRASTRUCTURE" MAC_ADDRESS="AF1D00000301" MEDIUM_ACCESS_PROTOCOL="EDCAF" PHYSICAL_TYPE="DSSS" RATE_ADAPTATION="FALSE"
     dot11LongRetryLimit="4" dot11RTSThreshold="3000" dot11ShortRetryLimit="7">
      <AC_BK dot11EDCATableAIFSN="7" dot11EDCATableCWmax="1023" dot11EDCATableCWmin="31" dot11EDCATableMSDULifetime="500" dot11EDCATableTXOPLimit="3264"/>
      <AC_BE dot11EDCATableAIFSN="3" dot11EDCATableCWmax="1023" dot11EDCATableCWmin="31" dot11EDCATableMSDULifetime="500" dot11EDCATableTXOPLimit="3264"/>
      <AC_VI dot11EDCATableAIFSN="2" dot11EDCATableCWmax="31" dot11EDCATableCWmin="15" dot11EDCATableMSDULifetime="500" dot11EDCATableTXOPLimit="6016"/>
      <AC_VO dot11EDCATableAIFSN="2" dot11EDCATableCWmax="15" dot11EDCATableCWmin="7" dot11EDCATableMSDULifetime="500" dot11EDCATableTXOPLimit="3264"/>
    </PROTOCOL_PROPERTY>
  </PROTOCOL>
</LAYER>
<LAYER TYPE="PHYSICAL_LAYER">
  <PROTOCOL NAME="IEEE802.11" SETPROPERTY="TRUE">
    <PROTOCOL_PROPERTY ANTENNA_GAIN="0" ANTENNA_HEIGHT="1" BANDWIDTH="20" CCA_MODE="ENERGY_DETECTION" CONNECTED_TO="Access_Point_1" CONNECTION_MEDIUM="Wireles"
     CW_MAX="1023" CW_MIN="31" D0="1" FREQUENCY_BAND="2.4" SIFS="10" SLOT_TIME="20" STANDARD="IEEE802.11b" STANDARD_CHANNEL="1_2412
     TX_POWER="100" TX_TYPE="DSSS"/>
  </PROTOCOL>
```

Figure 12-5: Invalid attribute in configuration file

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.

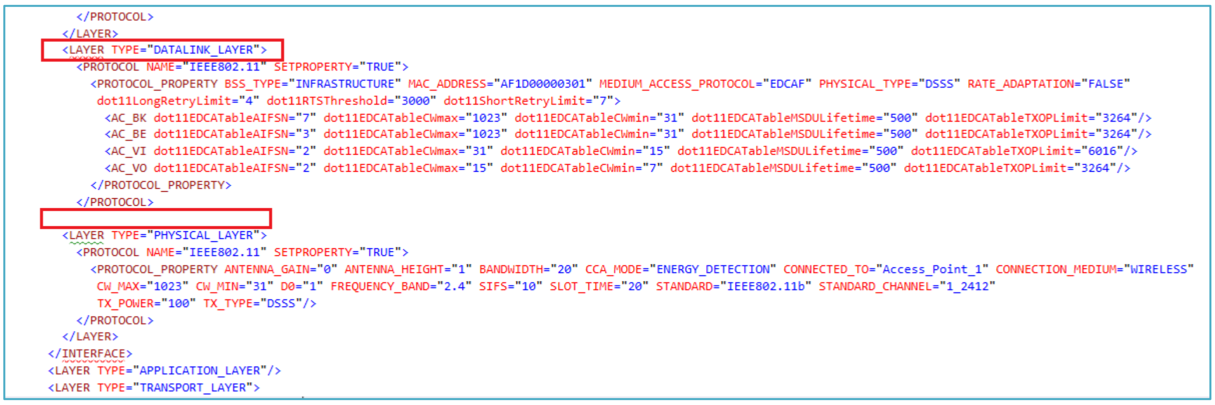
Note: *If the schema file and the configuration file are not present in the same folder, the zigzag lines won’t appear. So, place the Configuration file and Schema File in the same location or change the path of schema file in the configuration file.*

12.3.2 Error in tags in configuration file attributes

Simulation does not commence, and error is displayed at the command prompt. Also, red lines appearing at the tag specifying the Layer in the Configuration file

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

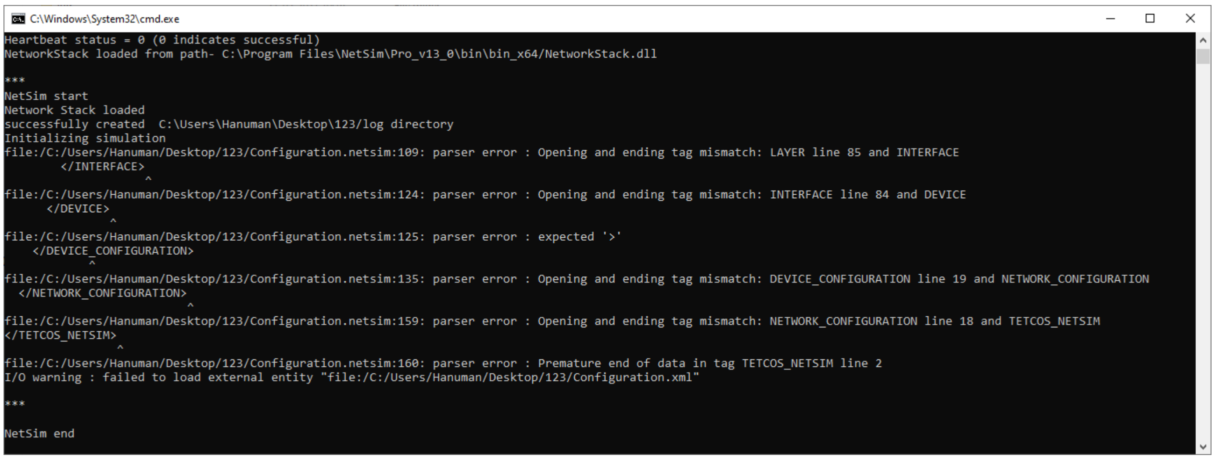
Example: If the closing tag is not specified for the Data link Layer, then the zigzag lines appear at the starting tags of Data link Layer and the Network Layer.



```
        </PROTOCOL>
      </LAYER>
      <LAYER TYPE="DATALINK_LAYER">
        <PROTOCOL NAME="IEEE802.11" SETPROPERTY="TRUE">
          <PROTOCOL_PROPERTY BSS_TYPE="INFRASTRUCTURE" MAC_ADDRESS="AF1D00000301" MEDIUM_ACCESS_PROTOCOL="EDCAF" PHYSICAL_TYPE="DSSS" RATE_ADAPTATION="FALSE"
           dot11LongRetryLimit="4" dot11RTSThreshold="3000" dot11ShortRetryLimit="7">
            <AC_BK dot11EDCATableAIFSN="7" dot11EDCATableCWmax="1023" dot11EDCATableCWmin="31" dot11EDCATableMSDULifetime="500" dot11EDCATableTXOPLimit="3264"/>
            <AC_BE dot11EDCATableAIFSN="3" dot11EDCATableCWmax="1023" dot11EDCATableCWmin="31" dot11EDCATableMSDULifetime="500" dot11EDCATableTXOPLimit="3264"/>
            <AC_VI dot11EDCATableAIFSN="2" dot11EDCATableCWmax="31" dot11EDCATableCWmin="15" dot11EDCATableMSDULifetime="500" dot11EDCATableTXOPLimit="6016"/>
            <AC_VO dot11EDCATableAIFSN="2" dot11EDCATableCWmax="15" dot11EDCATableCWmin="7" dot11EDCATableMSDULifetime="500" dot11EDCATableTXOPLimit="3264"/>
          </PROTOCOL_PROPERTY>
        </PROTOCOL>

      <LAYER TYPE="PHYSICAL_LAYER">
        <PROTOCOL NAME="IEEE802.11" SETPROPERTY="TRUE">
          <PROTOCOL_PROPERTY ANTENNA_GAIN="0" ANTENNA_HEIGHT="1" BANDWIDTH="20" CCA_MODE="ENERGY_DETECTION" CONNECTED_TO="Access_Point_1" CONNECTION_MEDIUM="WIRELESS"
           CW_MAX="1023" CW_MIN="31" D0="1" FREQUENCY_BAND="2.4" SIFS="10" SLOT_TIME="20" STANDARD="IEEE802.11b" STANDARD_CHANNEL="1_2412"
           TX_POWER="100" TX_TYPE="DSSS"/>
        </PROTOCOL>
      </LAYER>
    </INTERFACE>
    <LAYER TYPE="APPLICATION_LAYER"/>
    <LAYER TYPE="TRANSPORT_LAYER">
```

Figure 12-6: Error in tags in configuration file

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



C:\Windows\System32\cmd.exe

```
Heartbeat status = 0 (0 indicates successful)
NetworkStack loaded from path- C:\Program Files\NetSim\Pro_v13_0\bin\bin_x64/NetworkStack.dll

***
NetSim start
Network Stack loaded
successfully created  C:\Users\Hanuman\Desktop\123/log directory
Initializing simulation
file:/C:/Users/Hanuman/Desktop/123/Configuration.netsim:109: parser error : Opening and ending tag mismatch: LAYER line 85 and INTERFACE
        </INTERFACE>
                    ^
file:/C:/Users/Hanuman/Desktop/123/Configuration.netsim:124: parser error : Opening and ending tag mismatch: INTERFACE line 84 and DEVICE
      </DEVICE>
               ^
file:/C:/Users/Hanuman/Desktop/123/Configuration.netsim:125: parser error : expected '>'
    </DEVICE_CONFIGURATION>
            ^
file:/C:/Users/Hanuman/Desktop/123/Configuration.netsim:135: parser error : Opening and ending tag mismatch: DEVICE_CONFIGURATION line 19 and NETWORK_CONFIGURATION
  </NETWORK_CONFIGURATION>
                          ^
file:/C:/Users/Hanuman/Desktop/123/Configuration.netsim:159: parser error : Opening and ending tag mismatch: NETWORK_CONFIGURATION line 18 and TETCOS_NETSIM
</TETCOS_NETSIM>
                ^
file:/C:/Users/Hanuman/Desktop/123/Configuration.netsim:160: parser error : Premature end of data in tag TETCOS_NETSIM line 2
I/O warning : failed to load external entity "file:/C:/Users/Hanuman/Desktop/123/Configuration.xml"

***

NetSim end
```

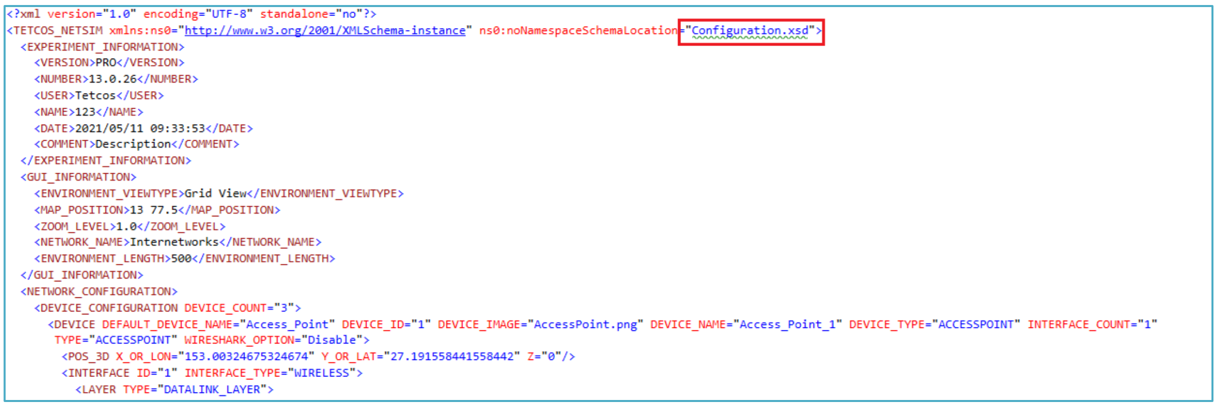
Figure 12-7: NetSim Run through CLI and following error displayed in the command prompt

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

12.3.3 Error lines in configuration.xsd in the Configuration file

Blue lines appear at configuration.xsd in the Configuration file.

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



```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<TETCOS_NETSIM xmlns:ns0="http://www.w3.org/2001/XMLSchema-instance" ns0:noNamespaceSchemaLocation="Configuration.xsd">
  <EXPERIMENT_INFORMATION>
    <VERSION>PRO</VERSION>
    <NUMBER>13.0.26</NUMBER>
    <USER>Tetcos</USER>
    <NAME>123</NAME>
    <DATE>2021/05/11 09:33:53</DATE>
    <COMMENT>Description</COMMENT>
  </EXPERIMENT_INFORMATION>
  <GUI_INFORMATION>
    <ENVIRONMENT_VIEWTYPE>Grid View</ENVIRONMENT_VIEWTYPE>
    <MAP_POSITION>13 77.5</MAP_POSITION>
    <ZOOM_LEVEL>1.0</ZOOM_LEVEL>
    <NETWORK_NAME>Internetworks</NETWORK_NAME>
    <ENVIRONMENT_LENGTH>500</ENVIRONMENT_LENGTH>
  </GUI_INFORMATION>
  <NETWORK_CONFIGURATION>
    <DEVICE_CONFIGURATION DEVICE_COUNT="3">
      <DEVICE DEFAULT_DEVICE_NAME="Access_Point" DEVICE_ID="1" DEVICE_IMAGE="AccessPoint.png" DEVICE_NAME="Access_Point_1" DEVICE_TYPE="ACCESSPOINT" INTERFACE_COUNT="1"
       TYPE="ACCESSPOINT" WIRESHARK_OPTION="Disable">
        <POS_3D X_OR_LON="153.00324675324674" Y_OR_LAT="27.191558441558442" Z="0"/>
        <INTERFACE ID="1" INTERFACE_TYPE="WIRELESS">
          <LAYER TYPE="DATALINK_LAYER">
```

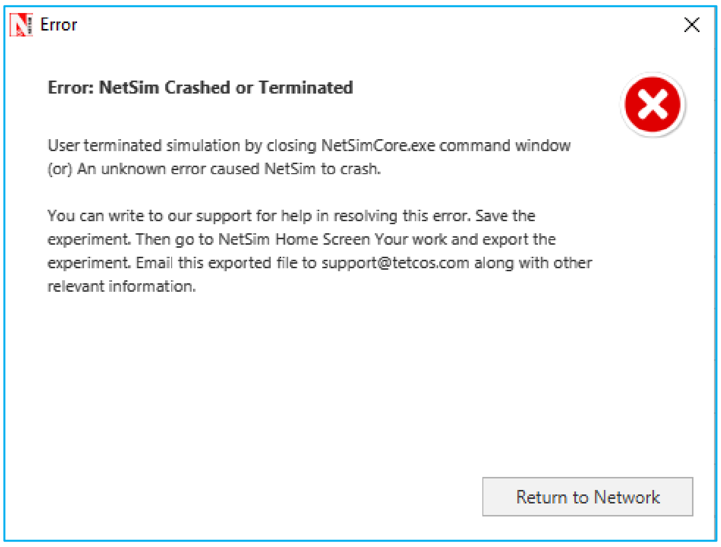
Figure 12-8: Error lines in configuration.xsd in the Configuration file

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

12.4 Simulation terminates and "NetSim Backend has stopped working" displayed

Simulation terminates and exhibits unpredictable behavior. An error message stating, "NetSim Crashed or Terminated" is thrown.

<u>Example:</u>



Error

Error: NetSim Crashed or Terminated

User terminated simulation by closing NetSimCore.exe command window (or) An unknown error caused NetSim to crash.

You can write to our support for help in resolving this error. Save the experiment. Then go to NetSim Home Screen Your work and export the experiment. Email this exported file to support@tetcos.com along with other relevant information.

Figure 12-9: NetSim Crashed or Terminated window

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

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

12.5 Licensing

12.5.1 No License for product (-1) error

NetSim dongle is running in the server system. When running the NetSim in the Client system showing “No License for product (-1)” error.

Possible Reasons:

1. Firewall in the client system is blocking the Network traffic.
2. No network connection between Client and Server.
3. License Server is not running in the Server system.

Solution:

1. The installed firewall may block traffic at 5053 port used for licensing. So, either the user can stop the firewall, or may configure it to allow port 5053.
2. Contact the Network-in-charge and check if the Server system can be pinged from client.
3. Check whether License Server is running in the Server system or not.

12.6 Troubleshooting VANET simulations that interface with SUMO

12.6.1 Guide for Sumo

- Link for the Sumo Website - http://www.dlr.de/ts/en/desktopdefault.aspx/tabid-9883/16931_read-41000/for help related to Sumo.
- In case sumo Configuration files do not open, Right click on any Sumo Configuration file, go to properties→open with→sumo.
- While Running NetSim Vanet Simulation – If any message pops up as **“SUMO_HOME” Not found**→ Go to My computer → System Properties → Advanced system settings → Environment Variables. Add an Environment variable as “SUMO_HOME”.
- Sumo Configuration File must contain the paths of the Vehicle routes and Networks file.
- Set the exact End Time for Sumo Simulation in Sumo Configuration File.

12.6.2 Guide for Python

- Any Python 2.7 version Installer would work fine for running simulations.
- If you have installed python by an external Installer, make sure the Python Path is set. It would be set automatically by python installer that comes with NetSim.
- In case **“Pywin 32**” is not getting installed, or during simulation, error occurs as “**win32 modules not found**” try the code below (Run it as a python Code).

```
import sys
from _winreg import *
# tweak as necessary
version = sys.version[:3]
installpath = sys.prefix
regpath = "SOFTWARE\\Python\\Pythoncore\\%s\\" % (version)
installkey = "InstallPath"
pythonkey = "PythonPath"
pythonpath = "%s;%s\\Lib\\;%s\\DLLs\\" % (
    installpath, installpath, installpath
)
def RegisterPy():
    try:
        reg = OpenKey(HKEY_CURRENT_USER, regpath)
    except EnvironmentError as e:
        try:
            reg = CreateKey(HKEY_CURRENT_USER, regpath)
            SetValue(reg, installkey, REG_SZ, installpath)
            SetValue(reg, pythonkey, REG_SZ, pythonpath)
            CloseKey(reg)
        except:
            print "*** Unable to register!"
            return
        print "--- Python", version, "is now registered!"
        return
    if (QueryValue(reg, installkey) == installpath and
        QueryValue(reg, pythonkey) == pythonpath):
        CloseKey(reg)
        print "=== Python", version, "is already registered!"
        return
    CloseKey(reg)
    print "*** Unable to register!"
    print "*** You probably have another Python installation!"
if __name__ == "__main__":
   RegisterPy()
```

12.6.3 VANET Simulation

i. Changing Vehicle (Node) Names, Moving or deleting vehicles etc. are disabled in Vanets Simulation.
ii. On running simulation, Backend calls Python file.
iii. NetSim protocol engine waits for the Pipes connection to be established.

12.6.4 Python

- SUMO_HOME Environment variable is checked. If Environment variable is not present, Error is displayed as "key interrupt error" in SUMO_HOME.
- Python File waits for Pipes connection. ("waiting for pipes to connect").
- It reads initial data as GUI enable/disable from simulation engine.
- "Checking sumo" is printed. If the environment variable SUMO_HOME points to wrong directory, error is displayed.
- Sumo Simulation is started where Sumo Binary is checked (To check Sumo.exe or Sumo GUI are working in the system or not). Then a TCP connection is made
- A while loop runs – It follows the following procedure.
  i. Send Garbage value to Backend to clear pipe buffers (pipes).
  ii. Read Vehicle name from NetSim (pipes).
  iii. Compare with each vehicle present in Sumo. If vehicle is present –Then write confirmation (pipes) and read its position from NetSim (2pipes for X and Y coordinates). Also, sumo is stepped forward for every first vehicle in the list of current vehicles in sumo.
- If vehicle not present, fail ('f') is sent.
- Pipes and TCP closed.

12.6.5 NetSim Core Protocol Library

- After establishing the connection, NetSim VANET Library checks for GUI flag, and sends '1' if animation status is online.
- As simulation proceeds, NetSim VANET library sends vehicle name to python, and receives XY positions, which are passed from python.
- Positions are updated and simulation proceeds.

13 NetSim Videos

In order to have a better understanding of NetSim, users can access YouTube channel of Tetcos at www.youtube.com/tetcos and check out the various videos available.

14 R&D projects in NetSim

Example R & D projects in NetSim is available in www.tetcos.com/file-exchange.

15 NetSim FAQ/Knowledgebase

NetSim knowledgebase with hundreds on FAQs on how NetSim works is available at https://tetcos.freshdesk.com/support/home

List of known issues in v13.1 is available at

https://support.tetcos.com/support/solutions/articles/14000101817-list-of