**INTRODUCTION:-**

*ns-3* has been developed to provide an open, extensible network simulation platform, for networking research and education. In brief, *ns-3* provides models of how packet data networks work and perform, and provides a simulation engine for users to conduct simulation experiments. Some of the reasons to use *ns-3* include

to perform studies that are more difficult or not possible to perform with real systems, to study system behavior in a highly controlled, reproducible environment, and to learn about how networks work. Users will note that the available model set in *ns-3* focuses on modeling how Internet protocols and networks work, but *ns-3* is not limited to Internet systems; several users are using *ns-3* to model non-Internet-based systems.

Many simulation tools exist for network simulation studies. Below are a few distinguishing features of *ns-3* in contrast to other tools.

* *ns-3* is designed as a set of libraries that can be combined together and also with other external software libraries. While some simulation platforms provide users with a single, integrated graphical user interface environment in which all tasks are carried out, *ns-3* is more modular in this regard. Several external animators and data analysis and visualization tools can be used with *ns-3*. However, users should expect to work at the command line and with C++ and/or Python software development tools.
* *ns-3* is primarily used on Linux or macOS systems, although support exists for BSD systems and also for Windows frameworks that can build Linux code, such as Windows Subsystem for Linux, or Cygwin. Native Windows Visual Studio is not presently supported although a developer is working on future support. Windows users may also use a Linux virtual machine.

 A user will be more productive with ns-3 for the following reasons:

* Ns-3 is actively maintained with an active, responsive users mailing list, while ns-3 is only lightly maintained and has not seen significant development in its main code tree for over a decade.
* Ns-3 provides features not available in ns-3, such as a implementation code execution environment (allowing users to run real implementation code in the simulator)

**INSTALLATION PROCEDURE OF NS3:-**

**Setting up wsl on windows10:**

1. Go to **Settings > Update & Security  > For Developers**. And turn on the developer mode.
2. Go to **Control Panel > Programs > Programs and Feature**. In the left sidebar click on **Turn on Windows features on or off**.  Check the option for **Windows Subsystem for Linux**.
3. WSL is now enabled. **Reboot**to proceed to the next step.
4. After rebooting, open Microsoft Store and find Ubuntu. Install it

**Setting up work environment:**

1. Launch Ubuntu. It could be done either by typing **Ubuntu**on search bar, or typing **ubuntu** on command line.
2. Set up basic information. Password etc.
3. Perform basic update and upgrade before doing anything else.

sudo apt-get update && apt-get upgrade

1. Install **Desktop Environment.** Even though Ubuntu WSL only offer command line as user interface, but installing Xfce can make it easier to navigate for most user.
2. Install the latest [VcXsrv](https://sourceforge.net/projects/vcxsrv/" \t "_blank)version.
3. Install **Xfce desktop**.

sudo apt-get install xfce-terminal xfce

1. Specify the display server.
2. add DISPLAY=:0.0 to ~/.bashrc
3. then run **source ~/.bashrc**

type in the terminal **export DISPLAY=:0.0**

1. Open **XLaunch**, choose “One large window” or “One large window without titlebar” and set the “display number” to 0. Other settings leave as default and finish the configuration.

11.Run the following command in the terminal.

startxfce4

12.Wait a moment until **Xfce** desktop is showing up in **XLaunch**.

13.Now that the desktop environment is ready to use, close **Xlaunch**. We don’t need it yet to installing NS3.

**Installation of NS3:-**

1. Perform update again. Just in case.

sudo apt-get update

1. Use the following command to install all the required dependencies:
   1. GCC & GCC compiler

sudo apt-get install gcc g++ python

* 1. Python

sudo apt-get install python-dev

* 1. Qt Framework

sudo apt-get install qt4-dev-tools

* 1. Version control and repository management

sudo apt-get install git mercurial

* 1. Compilation package

sudo apt-get install cmake libc6-dev libc6-dev-i386 g++-multilib

* 1. Debugging tools

sudo apt-get install gdb valgrind

* 1. GNU scientific library

sudo apt-get install gsl-bin libgsl0-dev libgsl2

* 1. Parser generator

sudo apt-get install flex bison libfl-dev

* 1. Packet capturing tools

sudo apt-get install tcpdump wireshark

* 1. Database support

sudo apt-get install sqlite sqlite3 libsqlite3-dev

* 1. XML support

sudo apt-get install libxml2 libxml2-dev

* 1. GTK library

sudo apt-get install libgtk2.0-0 libgtk2.0-dev

* 1. OS based virtualization

sudo apt-get install vtun lxc

* 1. SC modification tools

sudo apt-get install uncrustify

* 1. Graphical tools (*execute one line at a time*)
  2. sudo apt-get install doxygen graphviz imagemagick
  3. sudo apt-get install texlive texlive-extra-utils texlive-latex-extra texlive-font-utils dvipng

sudo apt-get install python-sphinx dia

* 1. Python library

sudo apt-get install python-pygraphviz python-kiwi python-pygoocanvas libgoocanvas-dev

* 1. C++ library

sudo apt-get install libboost-signals-dev libboost-filesystem-dev

* 1. High performance computing

sudo apt-get install openmpi-bin openmpi-common openmpi-doc libopenmpi-dev

* 1. Plotting engine

sudo apt-get install gnuplot plotdrop

1. For downloading and installing NS3 create a new directory called s*oftware* under the *Home* directory. It’s not required actually, it just personal thing to keep things neat and clean.
2. cd ~

mkdir software

1. At the time of writing this, the latest stable version of NS3 is ns-3.28 which are availabe for download at NS3 Releases. Now download the file and extract it under *software*directory.
2. cd software
3. wget https://www.nsnam.org/release/ns-allinone-3.28.tar.bz2

tar xvjf ns-allinone-3.28.tar.bz2

6. The output is a directory named **ns-allione-3.28**. And for the sake of simplicity the directory will renamed.

mv ns-allinone-3.28 ns3

cd ns3

ls

7. Now the only thing to do is build NS3 from ground up using the following command:

./build.py --enable-tests --enable-examples

8. Some extra *waf* configuration.

CXXFLAGS="-O3" ./waf configure

**PROBLEM STATEMENT:-**

**Realtime-udp-echo using Network Topology:**

This script shows that UDP flows from n0 to n1 and back to the drop tail queues. Tracing queues and packet receptions to the file “udp-echo.tr”

n0 n1 n2 n3

| | | |

=================

LAN

**OUTPUT EXPLANATION:-**

Here, initially we have node n0 has client to make node n1 also client we create a udp echo server application on n1, Once it is done create a udp echo client application to send udp datagram from node n0 to n1.The echo application starts at time 2.0sec where the packets starts transferring here the node n2 and n3 acts as a server where it can only receive they packets can’t be sent.

N0 acts as client and send packets to n1 at time 2.0sec and further it moves to remaining nodes. Once the packet is reached to the server, the echo application starts i.e, the packets are sent back to the client.

This process continues until the time reaches to 10sec.