Mohd. Aftab Alam
02013302717

EXPERIMENT 1

Aim :

Introduction to Network Simulator 2 (NS2). # Downloading and Installation of NS2.

Introduction to Computer Network Laboratory. # Introduction to Discrete Event Simulation.

Discrete Event Simulation Tools – NS2/NS3, OMNet++.

Theory :

Introduction to NS2 :

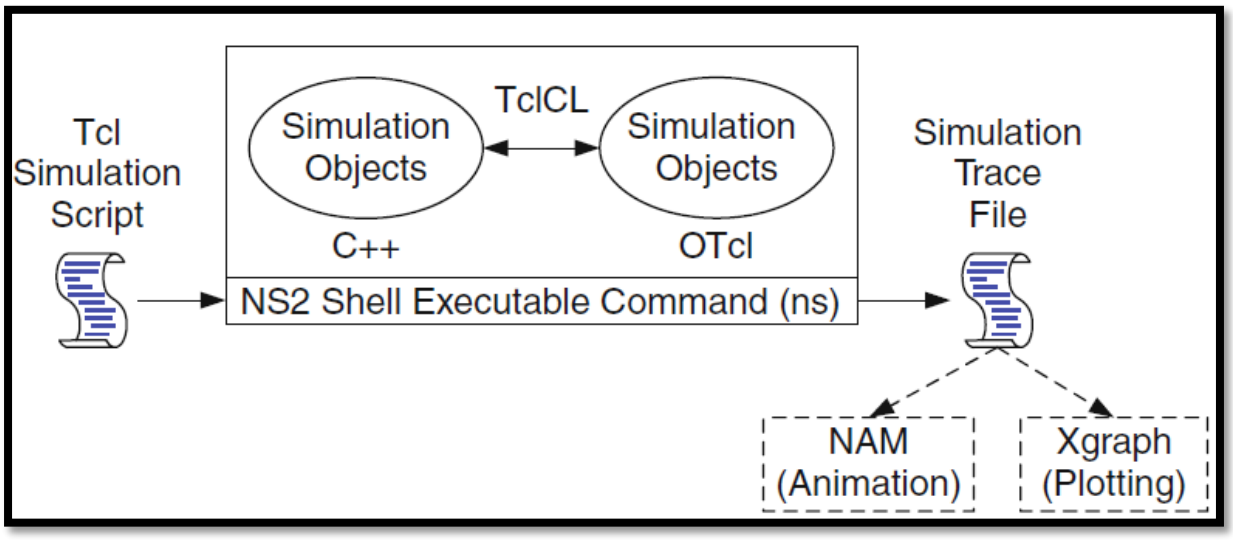
NS2 stands for Network Simulator Version 2. It is an open-source event-driven simulator designed specifically for research in computer communication networks. It is simply an event driven simulation tool. Useful in studying the dynamic nature of communication networks. Simulation of wired as well as wireless network functions and protocols (routing algorithms, TCP, UDP) can be done using NS2. In general, NS2 provides users with a way of specifying such network protocols and simulating their corresponding behaviours.

Features of NS2 :

- It is a discrete event simulator for networking research
- It provides substantial support to simulate bunch of protocols like TCP, FTP, UDP, https and DSR
- It simulates wired and wireless network
- It is primarily UNIX based
- Uses Tcl as its scripting language
- OTcl : Object Oriented support
- TclCL : C++ and OTcl linkage
- Discrete event scheduler

Basic Architecture of NS2 :

NS2 consists of two key languages : C++ and Object-oriented Tool command language (OTcl). While the C++ defines the internal mechanism of the simulation objects, the OTcl sets up simulation by assembling and configuring the objects as well as scheduling discrete events. The C++ and the OTcl are linked together using TclCL.



Mohd. Aftab Alam
02013302717

Downloading and Installation of NS2 :

Following are the steps to install NS2 on UNIX based system :

Downloading :

Download ns-allinone-2.35 from here :

https://sourceforge.net/projects/nsnam/files/allinone/ns-allinone-2.35/

It gets downloaded into your `'/home/your_user_name/Downloads'` directory.

Copy it to `/opt` folder by the following command :

```
cp /home/user_name/Downloads/ns-allinone-2.35.tar.gz /opt/
```

Installation :

Open Terminal and Execute the following commands :

```
1. sudo apt-get update
2. sudo apt-get dist-upgrade
3. sudo apt-get update
4. sudo apt-get gcc
5. sudo apt-get install build-essential autoconf automake
6. sudo apt-get install tcl8.5-dev tk8.5-dev
7. sudo apt-get install perl xgraph libxt-dev libx11-dev libxmu-dev
```

Execute the following commands for Extraction :

```
1. tar -zxvf ns-allinone-2.35.tar.gz
2. cd ns-allinone-2.35
3. ./install
```

Open `bashrc` file to Set the Environment Variables. Execute following command on terminal :

```
sudo gedit ~/.bashrc
```

Copy the following lines at the end of the file :

```
# LD_LIBRARY_PATH
OTCL_LIB=/opt/ns-allinone-2.35/otcl-1.14/
NS2_LIB=/opt/ns-allinone-2.35/lib/
USR_Local_LIB=/usr/local/lib/
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:$OTCL_LIB:$NS2_LIB:$USR_Local_LIB
# TCL_LIBRARY
TCL_LIB=/opt/ns-allinone-2.35/tcl8.5.10/library/
USR_LIB=/usr/lib/
export TCL_LIBRARY=$TCL_LIBRARY:$TCL_LIB:$USR_LIB
```

Mohd. Aftab Alam
02013302717

```
# PATH

XGRAPH=/opt/ns-allinone-2.35/xgraph-12.2/:/opt/ns-allinone-
2.35/bin/:/opt/ns-allinone-2.35/tcl8.5.10/unix/:/opt/ns-allinone-
2.35/tk8.5.10/unix/

NS=/opt/ns-allinone-2.35/ns-2.35/

NAM=/opt/ns-allinone-2.35/nam-1.15/

export PATH=$PATH:$XGRAPH:$NS:$NAM

#----
```

At last Execute the following commands :

1. source ~/.bashrc
2. ns

If you receive **"%"** sign, it means that NS is installed successfully.

Differences between NS2 & NS3 :

Sr. No.	Property	NS2	NS3
1.	Support	NS2 is not actively maintained and no longer support provided	NS3 is actively maintained with very good support. It is not backward compatible with NS2
2.	Architecture	NS2 is made with C++ and OTcl language. Tcl is used as scripting language for network simulation	NS3 is made with C++ and Python. Instead of scripting language, there is object oriented language and another option is Python
3.	Programming Language	C++, OTcl, Tcl script	C++, Python
4.	Recompilation	Recompilation is long process and many times fail	Recompilation is fast process and done in single command
5.	Post Analysis Supported File Formats	1. **.nam** for network animation 2. **.tr** for trace analysis 3. **.xg** for xgraph	1. **.xml** for network animation 2. **.tr** for trace analysis 3. **.pcap** for wireshark analysis
6.	Network Visualization	**nam** as network animator	1. **PyViz** as python visualizer 2. **NetAnim** as network animator
7.	Others	Only simulation	1. With simulation there is emulation also 2. DCE environment is provided which is used to simulate Linux kernel network stack

Mohd. Aftab Alam
02013302717

Introduction to Computer Network Laboratory :

The purpose of computer network laboratory is to learn the basic idea about open source network simulator NS2/NS3/other simulators and how to download, install and work with them.

Objectives :

- It helps to increase the efficiency of simulation.
- It is used to provide the details of the protocols and their operation.
- It is used to reduce packet and event processing time.

OTcl helps in the following ways :

- With the help of OTcl we can describe different network topologies.
- It helps us to specify the protocols and their applications.
- It allows fast development.
- Tcl is compatible with many platform and it is flexible for integration.
- Tcl is very easy to use and it is available for free.

Platform required to run network simulator :

- Unix and Unix like systems
- Linux (use Fedora or Ubuntu versions)
- Free BSD
- SunOS / Solaris
- Windows 95 / 98 / NT / 2000 / XP

Introduction to Discrete Event Simulation :

1. **Topology Definition :** To ease the creation of basic facilities and define their interrelationships, NS3 has a system of containers and helpers that facilitates this process.

2. **Model Development :** Models are added to simulation (UDP, IPv4) most of the time this is done using helpers.

3. **Node & Link Configuration :** Models set their default values, most of the time this is done using the attribute system.

4. **Execution :** Simulation facilities generate events, data requested by the user is logged.

5. **Performance Analysis :** After the simulation is finished and data is available as a time-stamped event trace. This data can then be statistically analysed with tools like R to draw conclusions.

6. **Graphical Visualization :** Raw data collected in a simulation can be graphed using tools like gnuplot, matplotlib or XGRAPH.

Discrete Event Simulation Tools :

Network Simulator 2 (NS2) :

NS2 is one of the most popular open source network simulators. The original NS is a discrete event simulator targeted at networking research. Brief introduction to NS2 is given below. NS2 is the second version of NS (Network Simulator). NS is originally based on REAL network simulator. The first version of NS was developed in 1989 and evolved a lot over the past few years. The current NS project is supported through DARPA. The current second version NS2 is widely used in academic research and it has a lot of packages contributed by different non-benefit groups

Mohd. Aftab Alam
02013302717

Network Simulator 3 (NS3) :

Similar to NS2, NS3 is also an open sourced discrete-event network simulator which targets research and educational use. NS3 is licensed under the GNU GPLv2 license, and is available for research and development. NS3 is designed to replace the current popular NS2. However, NS3 is not an updated version of NS2 since NS3 is a new simulator and it is not backward-compatible with NS2. The basic idea of NS3 comes from several different network simulators including NS2, YANS, and GTNetS.

OMNet ++ :

OMNeT++ is also a public-source, component-based network simulator with GUI support. Its primary application area is communication networks. OMNeT++ has generic and flexible architecture which makes it successful also in other areas like the IT systems, queuing networks, hardware architectures, or even business processes as well.

It is a component-based architecture. Components are also called modules and are programmed in C++. The components are then assembled into larger components and models by using a high-level language. Its function is similar to that of OTcl in NS2 and Python in NS3.

Since OMNeT++ is designed to provide a component-based architecture, the models or modules of OMNeT++ are assembled from reusable components. Modules are reusable and can be combined in various ways which is one of the main features of OMNeT++.

OMNeT++ **components** include :

1. Simulation kernel library
2. Compiler for the NED topology description language (nedc)
3. Graphical network editor for NED files (GNED)
4. GUI for simulation execution, links into simulation executable (Tkenv)
5. Command-line user interface for simulation execution (Cmdenv)
6. Graphical output vector plotting tool (Plove)
7. Graphical output scalars visualization tool (Scalars)
8. Model documentation tool (opp_neddoc)
9. Utilities (random number seed generation tool, makefile creation tool, etc.)
10. Documentation, sample simulations, etc.