Roll. No.: A022	Name: Kartik Padave
Sem/Year: VII/4	Batch: 1
Date of Experiment: 13/08/2022	Date of Submission: 27/08/2022
Grade --	

Study of ns2

What is NS2?

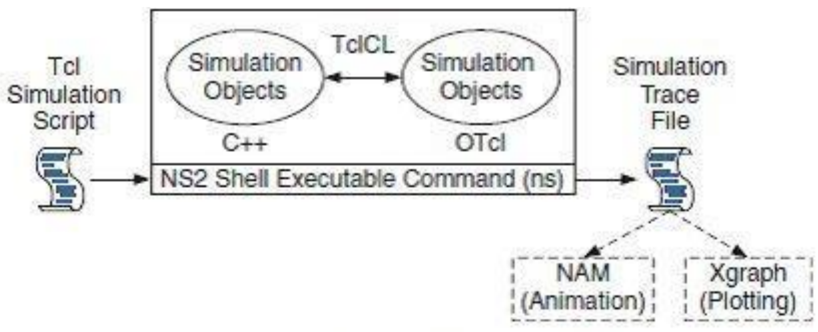
ns (from network simulator) is a name for a series of discrete event network simulators, specifically ns-1, ns-2, and ns-3. All are discrete-event computer network simulators, primarily used in research and teaching. The first version of ns, known as ns-1, was developed at Lawrence Berkeley National Laboratory (LBNL) in the 1995-97 timeframe by Steve McCanne, Sally Floyd, Kevin Fall, and other contributors. This was known as the LBNL Network Simulator and derived in 1989 from an earlier simulator known as REAL by S. Keshav. Ns-2 began as a revision of ns-1. From 1997 to 2000, ns development was supported by DARPA through the VINT project at LBL, Xerox PARC, UCB, and USC/ISI. In 2000, ns-2 development was supported through DARPA with SAMAN and through NSF with CONSER, both at USC/ISI, in collaboration with other researchers including ACIRI.

Features of NS2:

1. It is a discrete event simulator for networking research.
2. It provides substantial support to simulate bunch of protocols like TCP, FTP, UDP, https and DSR.
3. It simulates wired and wireless network.
4. It is primarily Unix based.
5. Uses TCL as its scripting language.
6. Otcl: Object oriented support
7. Tclcl: C++ and otcl linkage
8. Discrete event scheduler

Basic Architecture

NS2 consists of two key languages: C++ and Object-oriented Tool Command Language (OTcl). While the C++ defines the internal mechanism (i.e., a backend) 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



Basic architecture of NS.

Simulation workflow

The general process of creating a simulation can be divided into several steps:

1. *Topology definition:* To ease the creation of basic facilities and define their interrelationships, ns-3 has a system of containers and helpers that facilitates this process.
2. *Model development:* Models are added to simulation (for example, UDP, IPv4, point-to-point devices and links, applications); most of the time this is done using helpers.
3. *Node and link configuration:* models set their default values (for example, the size of packets sent by an application or MTU of a point-to-point link); 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 analyzed with tools like R to draw conclusions.
6. *Graphical Visualization*: Raw or processed data collected in a simulation can be graphed using tools like Gnuplot, matplotlib or XGRAPH.

Conclusion

Hence, we were able to perform the experiment.