[2CEIT503 COMPUTER NETWORKS]

Practical: 6

AIM- Study and installation of Network Simulator.

Department of Computer Engineering/Information Technology

Introduction

The network simulator is discrete event packet level simulator. The network simulator covers a very large number of application of different kind of protocols of different network types consisting of different network elements and traffic models. Network simulator is a package of tools that simulates behavior of networks such as creating network topologies, log events that happen under any load, analyze the events and understand the network. Well the main aim of our first experiment is to learn how to use network simulator and to get acquainted with the simulated objects and understand the operations of network simulation and we also need to analyze the behavior of the simulation object using network simulation.

Platform required to run network simulator

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

Backend Environment of Network Simulator

Network Simulator is mainly based on two languages. They are C++ and OTcl. OTcl is the object oriented version of Tool Command language. The network simulator is a bank of different network and protocol objects. C++ helps in the following way:

- It helps to increase the efficiency of simulation.
- It is used to provide details of the protocols and their operation.
- OTcl helps in the following way:
- 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 platforms and it is flexible for integration
- Tcl is very easy to use and it is available in free
- Developed by UC Berkeley
- Maintained by USC
- Popular simulator in scientific environment
- Other popular network simulators – Glomosim: UCLA, CMU; ParseC, Mobile Simulation mostly – OPNET: commercial software, graphical interface, not free; –
- Others: commercial ones, not free, e.g. IBM TPNS

NS2 Goals

- To support networking research and education
- Protocol design, traffic studies, etc.
- Protocol comparison;

- New architecture designs are also supported.
- To provide collaborative environment
- Freely distributed, open source;
- Increase confidence in result

Two Languages:

C++, OTcl OTcl: short for MIT Object Tcl, an extension to Tcl/Tk for object-oriented programming.

- Used to build the network structure and topology which is just the surface of your simulatoion;
- Easily to configure your network parameters;
- Not enough for research schemes and protocol architecture adaption.
- Two Languages (Con't) C++: Most important and kernel part of the NS2
- To implement the kernel of the architecture of the protocol designs;
- From the packet flow view, the processes run on a single node;
- To change or "comment out" the existing protocols running in NS2;
- Details of your research scheme.

Why 2 Languages?

- 2 requirements of the simulator – Detailed simulation of Protocol: Run-time speed; – Varying parameters or configuration: easy to use.
- C++ is fast to run but slower to code and change;
- OTcl is easy to code but runs slowly

Protocols/Models supported by NS2

- Wired Networking – Routing: Unicast, Multicast, and Hierarchical Routing, etc. – Transportation: TCP, UDP, others; – Traffic sources: web, ftp, telnet, cbr, etc. – Queuing disciplines: drop-tail, RED, etc. – QoS: IntServ and Diffserv Wireless Networking
- Ad hoc routing and mobile IP
- Sensor Networks(hmmm) – SensorSim: built up on NS2, additional features, for TinyOS

NS2 Models

• Traffic models and applications: Web, FTP, telnet, constant-bit rate(CBR) • Transport protocols: Unicast: TCP (Reno,Vegas), UDP Multicast • Routing and queuing: Wired routing, Ad Hoc routing. • Queuing protocols: RED(Random Early Drop), drop-tail • Physical media: Wired (point-to-point, LANs), wireless, satellite

Researches based on NS2

• Intserv/Diffserv (QoS) • Multicast: Routing, Reliable multicast • Transport: TCP Congestion control • Application: Web caching Multimedia • Sensor Networks: LEACH, Directed Diffusion, etc. Most are routing protocols.

NS2 Components

• NS2: the simulator itself, now version: ns-2.26 We will work with the part mostly. • NAM: Network animator. Visualized trace tool(not really). Nam editor: GUI interface to generate ns scripts Just for presentation now, not useful for research tracing. • Pre-processing: Traffic and topology generators • Post-processing: Simple trace analysis, often in Awk, Perl(mostly), or Tcl

Installation of NS2.

Installation Steps for ns-2.35 on Ubuntu above 18.00

Step 1: Download the ns-allinone-2.35.

Step 2:

Step 3:

Step 4:

…….

Step n: After installation of ns2 and nam Check following command on terminal

ns

% ctrl+c

nam