Abstract

NCTUns is a network simulator and emulator. By using a novel kernel re-entering simulation methodology, it provides many unique advantages over traditional network simulators and emulators. For this reason, since its public release on November 1, 2002, as of February 16, 2005, more than 2,745 people from 80 countries in the world have registered at its web site (http://NSL.csie.nctu.edu.tw/nctuns.html) and downloaded it.

NCTUns is open-source and runs on Linux. It can simulate many different types of networks. It has many high-quality journal and conference papers supporting its simulation methodology, design, and implementation. After five years of development, it has become a high-quality and useful product and has received many academic honors. In this paper, we give a brief introduction to NCTUns. More information is available at its web site.

Keyword: network simulation, network emulation

1. Introduction

Network simulators implemented in software are valuable tools for researchers to develop, test, and diagnose network protocols. Simulation is economical because it can carry out experiments without the actual hardware. It is flexible because it can, for example, simulate a link with any bandwidth and propagation delay. Simulation results are easier to analyze than experimental results because important information at critical points can be easily logged to help researchers diagnose network protocols.

Network simulators, however, have their limitations. A complete network simulator needs to simulate networking devices (e.g., hosts and routers) and application programs that generate network traffic. It also needs to provide network utility programs to configure, monitor, and gather statistics about a simulated network. Therefore, developing a complete network simulator is a large effort. Due to limited development resources, traditional network simulators usually have the following drawbacks:

• Simulation results are not as convincing as those produced by real hardware and software equipment. In order to constrain their complexity and development cost, most network simulators usually can only simulate real-life network protocol implementations with limited details, and this may lead to incorrect results.

• These simulators are not extensible in the sense that they lack the standard UNIX POSIX application programming interface (API). As such, existing or to-be-developed real-life application programs cannot run normally to generate traffic for a simulated network. Instead, they must be rewritten to use the internal API provided by the simulator (if there is any) and be compiled with the simulator to form a single, big, and complex program.

To overcome these problems, Wang invented a kernel re-entering simulation methodology [1, 2] and used it to implement the Harvard network simulator [3]. Later on, Wang further improved the methodology and used it to design and implement the NCTUns network simulator and emulator [4].
Due to its unique features, NCTUs has received many academic honors. For example, it is selected by both ACM MobiCom 2002 [5] and MobiCom 2003 [6] international conferences as a research demonstration. The IEEE Network Magazine reports this tool in its July 2003 issue [7]. The IEEE MASCOTS 2004 international conference selects it as a tutorial [8]. The IEEE vehicular technology society selects it as a full-day tutorial at its workshop [9]. The IEEE INFOCOM 2005 international conference selects it as a demonstration [10]. An invited paper will be published by Wiley to illustrate the unique capabilities of NCTUs [11]. In addition, an invited book chapter will be published by Nova Science Publishers, Inc. in one of its new book to introduce the unique emulation capability of NCTUs [12].

In the rest of the paper, we will briefly present the unique features of NCTUs.

2. Unique and Important Features

It can be used as an emulator. An external host in the real world can exchange packets (e.g., set up a TCP connection) with nodes (e.g., host, router, or mobile station) in a network simulated by NCTUs. Two external hosts in the real world can also exchange their packets via a network simulated by NCTUs. This feature is very useful as the function and performance of real-world devices can be tested under various simulated network conditions.

It directly uses the real-life Linux’s TCP/IP protocol stack to generate high-fidelity simulation results. By using a novel kernel re-entering simulation methodology, a real-life UNIX (e.g., Linux) kernel’s protocol stack can be directly used to generate high-fidelity simulation results.

It can use any real-life existing or to-be-developed UNIX application program as a traffic generator program without any modification. Any real-life program can be run on a simulated network to generate network traffic. This enables a researcher to test the functionality and performance of a distributed application or system under various network conditions. Another important advantage of this feature is that application programs developed during simulation studies can be directly used on real-world UNIX machines and deployed in the real world after simulation studies are finished. This eliminates the time and effort required to port a simulation prototype to a real-world implementation if traditional network simulators are used.

It can use any real-life UNIX network configuration and monitoring tools. For example, the UNIX route, ifconfig, netstat, tcpdump, traceroute commands can be run on a simulated network to configure or monitor the simulated network.

In NCTUs, the setup and usage of a simulated network and application programs are exactly the same as those used in real-world IP networks. For example, each layer-3 interface has an IP address assigned to it and application programs directly use these IP addresses to communicate with each other. For this reason, any person who is familiar with real-world IP networks can easily learn and operate NCTUs in a few minutes. For the same reason, NCTUs can be used as an educational tool to teach students how to configure and operate a real-world network.

It can simulate fixed Internet, Wireless LANs, mobile ad hoc (sensor) networks, GPRS networks, and optical networks. A wired network is composed of fixed nodes and point-to-point links. Traditional circuit-switching optical networks and more advanced Optical Burst Switching (OBS) networks are also supported. A wireless networks is composed of IEEE 802.11 (b) mobile nodes and access points (both the ad-hoc mode and infrastructure mode are supported). GPRS cellular networks are also supported.

It can simulate various networking devices. For example, Ethernet hubs, switches, routers, hosts, IEEE 802.11 (b) wireless stations and access points, WAN (for purposely delaying/dropping/reordering packets), Wall (wireless signal obstacle), GPRS base station, GPRS phone, GPRS GGSN, GPRS SGSN, optical circuit switch, optical burst switch, QoS DiffServ interior and boundary routers, etc.

It can simulate various protocols. For example, IEEE 802.3 CSMA/CD MAC, IEEE 802.11 (b) CSMA/CA MAC, learning bridge protocol, spanning tree protocol, IP, Mobile IP, DiffServ (QoS), RIP, OSPF, UDP, TCP, RTP/RTCP/SDP, HTTP, FTP, Telnet, etc.

Its simulation speed is high. By combining the kernel re-entering methodology with the discrete-event simulation methodology, a simulation job can be finished quickly.

Its simulation results are repeatable. If the chosen random number seed for a simulation case is fixed, the simulation results of a case are the same across different simulation runs even though there are some other activities (e.g., disk I/O) occurring on the simulation machine.

It provides a highly-integrated and professional GUI environment. This GUI can help a user (1) draw network topologies, (2) configure the protocol modules used inside a node, (3) specify the moving paths of mobile nodes, (4) plot network performance graphs, (5) play back the animation of a logged packet transfer trace, etc. All these operations can be easily and intuitively done with the GUI.
Its simulation engine adopts an open-system architecture and is open source. By using a set of module APIs provided by the simulation engine, a protocol module writer can easily implement his (her) protocol and integrate it into the simulation engine. NCTUns uses a simple but effective syntax to describe the settings and configurations of a simulation job. These descriptions are generated by the GUI and stored in a suite of files. Normally the GUI will automatically transfer these files to the simulation engine for execution. However, if a researcher wants to try his (her) novel device or network configurations that the current GUI does not support, he (she) can totally bypass the GUI and generate the suite of description files by himself (herself) using any text editor (or script program). The non-GUI-generated suite of files can then be manually fed to the simulation engine for execution.

It supports remote and concurrent simulations. NCTUns adopts a distributed architecture. The GUI and simulation engine are separately implemented and use the client-server model to communicate. Therefore, a remote user using the GUI program can remotely submit his (her) simulation job to a server running the simulation engine. This scheme can easily support the cluster computing model in which multiple simulation jobs are performed concurrently on different server machines. This can increase the total simulation throughput.

It supports tactics military mobile ad hoc network simulations and intelligent transportation systems simulations. In addition to supporting traditional IP network simulations, NCTUns supports tactics military mobile ad hoc network simulations in which an mobile node (e.g., a tank) can actively change its location, moving speed, and moving direction based on the current environment. NCTUns has also been used to design protocols for vehicle-formed mobile ad hoc networks on the roads, which is a new and important application in future intelligent transportation systems.

3. Conclusions

In this paper, we briefly present the development history and important features of the NCTUns 2.0 network simulator and emulator. More information is available in its web site at http://NSL.csie.nctu.edu.tw/nctuns.html.

References