NCTUns Emulation Testbed for Evaluating Real-life Applications over WiMAX Networks

Shie-Yuan Wang, C.C. Lin, P.H. Koo, and Y.M. Huang
Department of Computer Science
National Chiao Tung University, Taiwan
Email: {shieyuan, linjc, phku, yuming}@cs.nctu.edu.tw

Jen-Shun Yang
Information and Communications Research Laboratory
Industrial Technology Research Institute, Taiwan
Email: jsyang@itri.org.tw

***Abstract*—In this paper, we design and implement a hybrid WiMAX network testbed that integrates the NCTUns network emulator with a real-life WiMAX network. This hybrid WiMAX network testbed can seamlessly connect virtual networks (emulated by NCTUns) to a real-life WiMAX network. Thus, in this testbed real-life applications on real machines can exchange data with those on emulated (virtualized) network nodes. With this unique feature, our developed testbed provides a powerful, high-fidelity, and convenient testing environment for evaluating the performances and behaviors of real applications and devices in WiMAX networks. In this paper, the architecture and methodology of the developed testbed are explained, which are general and can collaborate with any other IP-based networks. In addition, an example study on this testbed is also presented to demonstrate its usage and capabilities.**

I. Introduction

The IEEE 802.16 WiMAX [1] technology has gained great attention in recent years. Operators in more than 140 countries [2] have deployed this next-generation network for providing better broadband communication quality. The deployment of such emerging broadband wireless networks creates opportunities for new applications and service models (e.g., broadband video multicasting and video-based online shopping). It can be foreseen that more and more new applications and services will be provided on these commercial broadband mobile networks. As a result, performance and reliability evaluation of these new applications and services will be highly desired. To satisfy such needs, in this paper we design and implement a flexible, high-fidelity, and convenient WiMAX network testbed for application and service providers.

Our implemented testbed uses the NCTUns network simulator/emulator [3] to emulate the operation of virtual networks and exploits its unique features to seamlessly connect with a component in a real-life WiMAX network, e.g., the Access Service Network (ASN) gateway and the Connectivity Service Network (CSN) gateway. The proposed network testbed is thus called hybrid because it comprises virtual networks and a real-life WiMAX network. On this testbed, applications to be tested can be directly run on real devices (e.g., WiMAX MSs) and be evaluated under various network conditions set by the emulated virtual networks.

We implemented this testbed in the M-Taiwan WiMAX Application Lab (MTWAL), which is the first application (and Proof-of-Concept) lab, selected by WiMAX Forum. MTWAL is established at the Industrial Technology Research Institute (ITRI) campus in Hsinchu, Taiwan. The lab deploys a mobile WiMAX experimental network, which so far comprises four base stations in the ITRI campus and is expected to cover other main districts in Hsinchu (e.g., the Hsinchu Science Park, National Chiao Tung University, and National Tsing Hua University) in the near future. The objective of MTWAL is to provide an environment for creating innovative mobile applications and facilitating interoperability testing of equipments from different vendors. It also allows for analysis of user behavior and network performance. On top of the experimental network of MTWAL, we connects the NCTUns network emulator with the ASN gateway of the MTWAL network.

Our proposed network testbed has several advantages. First, it allows users to flexibly create different network topologies and conditions between the ASN gateway and application servers, which greatly decreases the time and efforts required for setting up a testing case. Second, by exploiting the proxy ARP mechanism, the virtual networks emulated by NCTUns (and real devices behind this virtual network) can seamlessly connect with the real WiMAX network. Third, our methodology for connecting the virtual network and the real network is general. The virtual network created by the network emulator can connect with any other IP-based networks to together form a high-fidelity and high flexibility network testbed. In this paper, we first explain the design and implementation of this network testbed. An example study on video streaming traffic over the WiMAX network using this testbed is also presented to demonstrate its usage and capabilities.

The rest of this paper is organized as follows. In section II, we survey related works. In section III, we first briefly explain NCTUns and its emulation methodology and then present the architecture and methodology of our developed hybrid network testbed. In section IV, we demonstrate an example video streaming study over the proposed network testbed to show its capability and advantage. Finally, we conclude this paper in section V.

II. Related Work

Several network testbeds for evaluating WiMAX network performances have been proposed in the literature. The WiMAX Extension to Isolated Research Data networks

978-1-4244-8016-6/10/$26.00 ©2010 IEEE

(WEIRD) [8] [9] is the first research WiMAX testbed in Europe [10]. WEIRD has been deployed in four countries in Europe and used to demonstrate new applications that can be used in the WiMAX technology. WEIRD supports different scenarios such as environmental monitoring, video/voice streaming from medical equipments, and general VoIP, video conferencing applications.

In [11], Hu et al. develop a physical-layer testbed using signal-level equipments. Several signal-level performances, such as signal spectrum, Error Vector Magnitude (EVM) can be measured on this testbed. In addition, several commercial companies (such as Alvarion, Alcatel-Lucent, and Agilent) have proposed their own solutions for a variety of WiMAX network testings. However, none of these existing testbeds can provide both flexibility and fidelity at the same time, as compared with our developed NCTUns-based testbed. In our developed testbed, any real-life network applications can be tested under various complex network conditions emulated by the NCTUns network emulator. Such a feature, to the best of the authors' knowledge, is unique among existing network testbeds in the literature and can substantially reduce the efforts required to configure and control test cases.

III. Design and Implementation

A. Overview of NCTUns

NCTUns [3] [4] is an open-source network simulator/emulator with a unique kernel re-entering simulation/emulation methodology [5]. NCTUns allows real-world application programs to run on simulated/emulated networks without modifications. In addition, by exploiting pseudo device drivers in operating systems, NCTUns is capable of using the kernel-space network protocol stack to simulate the behaviors of the transport layer and the network layer (e.g., TCP/UDP/IP). The architecture of NCTUns is shown in Figure 1 and is briefly explained below. NCTUns is mainly composed of five components: 1) Graphical User Interface (GUI), 2) Job Dispatcher, 3) Coordinator, 4) Simulation Engine, 5) Applications, and 6) Kernel Patches.

The GUI program is an integrated tool that allows users to create network simulation cases in an intuitive manner. After creating a simulation case, one can run the simulation case using the GUI program. The GUI program will first consult the Job Dispatcher program to know which simulation server that it manages is available for running simulation. Upon obtaining the IP address of an available simulation server, the GUI program will compress the whole simulation case into a tarball file and transmit it to the Coordinator program on the chosen simulation server. The Coordinator program then will fork a Simulation Engine process to run the received simulation case. After the simulation is completed, the simulation results will be collected by the Coordinator and sent back to the GUI program for post-simulation analyses. Recall that NCTUns can use real-life application programs to generate traffic and use the kernel-space TCP/IP protocol stack to simulate ("run") the packet processing of the transport and network layers.

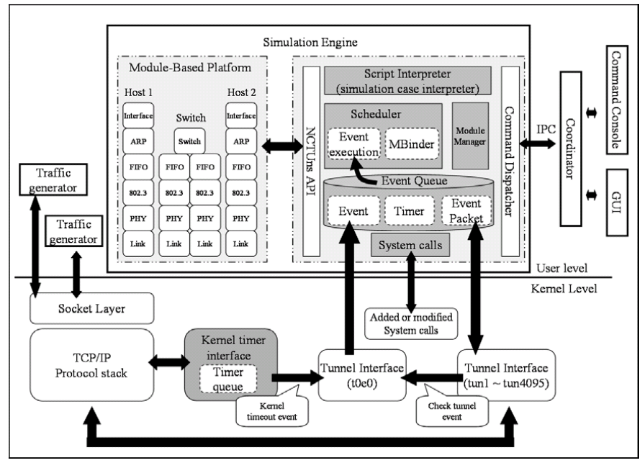

Fig. 1. The architecture of the NCTUns network simulator and emulator

Thus, application programs and kernel patches are independent components separated from the Simulation Engine.

Due to its unique architecture, application programs running on NCTUns can exchange data with those on real machines seamlessly. That is, the applications running on NCTUns are not aware that they are running on nodes in a simulated network and those running on real machines are not aware that they are transmitting/receiving data to/from application programs in a simulated network. Therefore, NCTUns can be used as a network emulator [1] without the need of modifications. The emulation of NCTUns uses a kernel module (called the network emulation module and abbreviated as NEM) to perform network address translation when needed. The details about the methodology of the NCTUns network emulation are available in [6] [7]

B. Developed Hybrid Network Testbed

In this paper, we built a hybrid WiMAX network testbed on top of NCTUns and the experimental WiMAX network of MTWAL. As shown in Figure 2, the architecture of our implemented hybrid network testbed can be divided into two parts. The first part is a real network, which comprises real WiMAX Mobile Stations (MSs), Base Stations (BSs), ASN gateways, CSN gateways, and real application servers. The second part is the emulated network, which is created and emulated by the NCTUns network emulator. Real machines can attach themselves to the emulated network by connecting to the emulated machine and run real applications on them.

Note that the NCTUns emulation should operate on the virtual network that it creates. Before the emulation starts, NCTUns will first create a virtual node in its emulated network to represent each external (real) machine. Each virtual node is assigned a virtual IP address (which is used in the emulated network), which corresponds to the real IP address of the interface on the corresponding external (real) machine. Each

[1]Network emulation is defined as a network simulation involved with real machines and the simulation speed of an emulation is set to be as fast as the real-world clock.

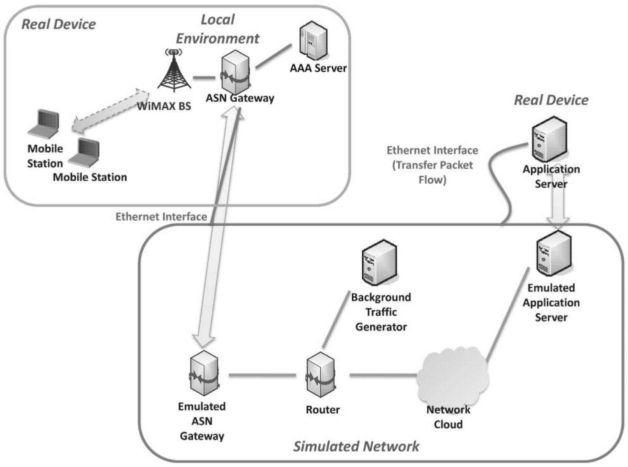

Fig. 2. The architecture of our proposed testbed

external application program is required to exchange data using its assigned virtual IP address. For example, suppose that an external application program intends to transmit data to its peer application program running on a real machine that is assigned the real IP address 140.113.100.100 and a virtual IP address 1.0.2.1. In this condition, this external application should transmit data to 1.0.2.1 rather than 140.113.100.100.

To achieve this, NCTUns employs the kernel-space NEM to perform IP address translation for packets sent from/transmitted to external machines. NEM exploits the existing kernel-space Netfilter facility to intercept the packets sent from/transmitted to external machines. On one hand, upon intercepting a packet sent from external machines and destined to a node with a virtual IP address, NEM first translates the (real) source IP addresses of the intercepted packet into the corresponding virtual IP addresses and then puts it back into the IP-layer routine that processes the reception of an IP packet. Afterward, this IP packet will be processed as a normal packet in the emulated virtual network.

On the other hand, upon intercepting a packet sent from a node with a virtual IP address and destined to an external machine, NEM should first replace the virtual source and destination IP addresses of the intercepted packet with the corresponding real IP addresses and then put it into the related IP-layer routine to proceed the transmission of this IP packet. Due to this design, when using the current NCTUns emulation methodology, an external sending program should know the virtual address of its corresponding receiving program and use it as the destination IP address to transmit its packets. Such a design, however, has two drawbacks explained below.

First, application programs should use virtual IP addresses to exchange data, which is counter-intuitive to users that have been accustomed to use real IP addresses. Second, due to the use of virtual IP addresses, these external application programs should be run on machines that are directly connected with the NCTUns network emulator. This is because, if there are intermediate machines involved, one should properly configure the routing tables of these machines for the used virtual IP addresses. Such a requirement, however, cannot be met in many existing networks. For example, in WiMAX network, the data packets of an MS is transmitted to the ASN gateway over tunnels, after being received by the serving BS. In this network architecture, it is infeasible to configure routes for inexistent virtual IP address on BSs and thus the tunnel mechanism will fail.

To solve these problems, we modified NEM to make it capable of detecting the real IP addresses of external machines. We first modified the GUI program so that users can use it to specify the real IP addresses of external machines. The GUI program will then generate the emulation configuration file for NEM, before the emulation starts. The emulation configuration file describes the specified real IP addresses and their corresponding virtual IP addresses. If NEM detects an incoming packet with its source and destination IP addresses matching the specified real IP addresses, it will first replace the source and destination IP addresses of this packet with the corresponding virtual IP addresses and then put it back into the IP-layer receive routine.

In contrast, if NEM detects an outgoing packet with its source and destination IP addresses matching the virtual IP addresses specified in the emulation configuration file, it will first replace the virtual IP address of the packet with the corresponding real IP addresses and then transmits the packet out. Using this design, in an emulation real applications can exchange data using intuitive real IP addresses. In addition, real applications are not required to run on machines directly connected with the NCTUns emulator. Instead, they can also be run on machines that are multi-hop away from the NCTUns emulator. Furthermore, due to the use of real IP addresses, intermediate machines now knows how to route these packets to the NCTUns emulator.

Our proposed network testbed has several advantages. First, it allows users to flexibly create different network topologies and conditions between the ASN gateway and application servers, which greatly decreases the time and efforts required for setting up a testing case. Second, by exploiting the proxy ARP (Proxy Address Resolution Protocol) mechanism, the virtual networks emulated by NCTUns (and real devices connecting to the NCTUns emulation server) can seamlessly connect with a real-life WiMAX network. The attachment of such virtual networks does not hinder the operation of the original WiMAX network. Only the packets that are destined to the nodes in the virtual networks or the real devices connecting with the virtual networks will be redirected to the NCTUns network emulator. Packets that are not destined to the virtual networks will be forwarded using their original routes.

IV. Case Study: Evaluating Quality of Transmitted Video Streams

In this section, we demonstrate how to study transmitted video stream quality on the developed testbed. We first explain how to configure an emulation case on this testbed in Section IV-A and then present the performance results in Section IV-B.

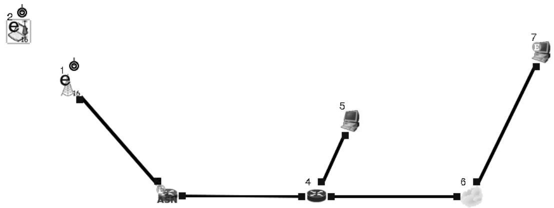

Fig. 3. The corresponding virtual network created by NCTUns

A. Emulation Experiment Setup

The topology of the emulation experiment in this section is similar to that shown in Fig. 2, except that in this experiment, we used only one WiMAX MS. Without loss of generality, in this experiment we let all real machines be in the subnet of 192.168.1.0/24. (In our testbed, real machines can also be assigned public IP addresses.) The MS is assigned a real IP address of 192.168.1.1 and its corresponding virtual IP address is 1.0.1.1. The machine running the NCTUns emulator has two interfaces. One (called eth0) connects with the ASN gateway and is assigned the real IP address of 192.168.1.100. The other (called eth1) connects with the application server (i.e., the machine running the application server program) and is assigned the real IP address of 192.168.1.200. The application server is assigned the real IP address of 192.168.1.10. Real applications are run on the real MS and the real application server.

The corresponding virtual topology created using the GUI program is shown in Fig. 3 In this topology node 1 represents the real MS and node 2 represents the real BS (which are not directly connected to the NCTUns emulation machine but shown in the virtual emulated topology for the integrity of the whole experimental network). Node 6 represents the real ASN gateway and node 10 represents the real application server. These two machines are directly connected with the NCTUns emulation machine. One sees that in the virtual emulated network, the WiMAX core network connects with the application server via a Wide Area Network (WAN) module, which allows users to specify the quality (such as the distributions of the packet loss rate and the packet re-ordering probability) of the core network using pre-defined mathematical distributions.

The next step is ito configure the routing table of the NCTUns emulation machine. The route commands required in this example case are shown as follows:

- route del -net 192.168.1.0/24 dev eth1
- route add 192.168.1.200 dev eth1

The first route command deletes the default route for the packets destined to the 192.168.1 subnet, which is undesired in our emulation experiment. The second route command adds an explicit route for the packets destined to the 192.168.1.200 host.

We then need to set up the Proxy ARP to make the ASN gateway redirect the traffic destined to the application server to the NCTUns emulation machine. The required steps for configuring the Proxy ARP are shown in the following:

- route add 192.168.1.10 dev eth1
- echo "1" > /proc/sys/net/ipv4/conf/eth0/proxy_arp
- echo "1" > /proc/sys/net/ipv4/conf/eth1/proxy_arp

By using the Proxy ARP mechanism, the ASN gateway will think that the MAC address associated with the IP address of 192.168.1.10 is used by the "eth1" interface on the NCTUns emulation machine. Thus, when it forwards packet destined to 192.168.1.10, it will send these packets to the NCTUns emulation. Such traffic redirection is desired in our emulation experiment.

We finally enable the IP forwarding capability of the NCTUns emulation machine, which is accomplished by the following command:

- echo "1" > /proc/sys/net/ipv4/ip_forward

After finishing the configuration of an emulation case, one can start the emulation through the GUI program. After the emulation starts, the WiMAX MS will attach itself to the MTWAL WiMAX experimental network and exchange data with the application server through the MTWAL network. When the data packets pass through the ASN gateway, due to the use of the proxy ARP protocol, these packets will be redirected to the NCTUns emulation machine and enter the virtual emulated network. When passing through the WAN module, these packets will be processed (e.g., dropped, delayed, or/and re-ordered) based on the network quality specified in the WAN module. When leaving the WAN module and arriving at the virtual application server node, these packets will be transmitted to the real application server via the wired line connecting these two machines.

Performance results can be generated by the protocol modules of the NCTUns network emulator and real applications on end hosts. In the following section, we present the performance results logged by real applications as an example.

B. Performance Results

In this section, we studied the performance of UDP video streaming traffic under various network conditions. The performance metric for the UDP video streaming traffic is the PSNR values of the original frames and reconstructed frames during the streaming. These PSNR values are computed using the "MSU Video Quality Measurement Tool" software, and we use "VLC media player" for UDP video streaming. The video source is a clip titled "Porsche 911 Turbo vs Nissan GT-R vs Audi R8 V10," public at YouTube website. The clip is encoded in the following setting:

- Encapsulation: MPEG-TS
- Resolution: 640 x 360
- Video codec: DixX 3, 1024 Kb/sec, 29.97 frame/sec
- Audio codec: MPEG Audio, 128Kb/sec, 2 channels, sampling rate: 44100

Four network conditions are considered in this study: link bandwidth, packet loss rate, constant-bitrate background traffic, and exponentially distributed background traffic. Fig. 4

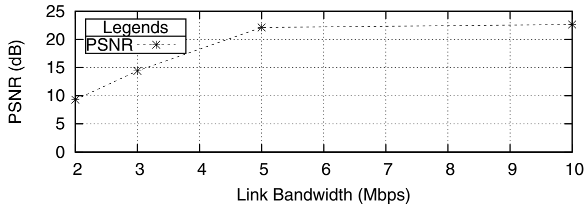

Fig. 4. PSNR over different link bandwidths

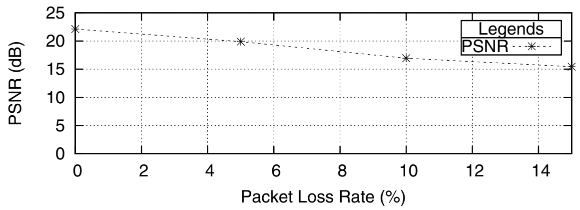

Fig. 5. PSNR over different packet loss rates

shows the PSNR results over different link bandwidths. The value increases as the link bandwidth increases, which is expected. One sees that the PSNR does not increase significantly after the link bandwidth exceeds 5 Mbps, but stay at the value of about 22 dB. This indicates that a PSNR of 22 dB can be achieved in such an environment. Generally speaking, the video quality is acceptable if the PSNR is higher than 20 dB.

Fig. 5 shows the PSNR over different packet loss rates. The PSNR gradually decreases as the packet loss rate increases. These results can be a reference for future application developers who want to provide such streaming services over WiMAX networks.

Fig. 6 and 7 show the PSNR over constant-bit-rate and exponentially distributed background traffic, respectively. Constant-bit-rate traffic can be seen as simpler and basic traffic, while exponentially distributed traffic is more complex and closer to real traffic. The figures show that the PSNR drops significantly after the mean load is larger than 3 Mbps in both cases. This value indicates the maximum background load that such video streaming can tolerate.

V. Conclusion

In this paper, we design and implement a hybrid network testbed that seamlessly connects a real-life network with the virtual networks emulated by the NCTUns network emulator. Due to its unique architecture, our developed network testbed provides both flexibility and fidelity for users. Over this hybrid network testbed, one can evaluate the performances and behaviors of real-life applications using real-life equipments under various network conditions with minimum efforts. The methodology of our developed network testbed can also be extended to support any other real-life IP-based networks such as the 4G LTE network.

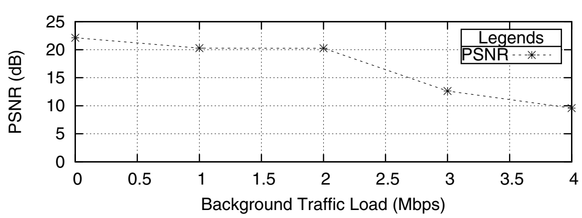

Fig. 6. PSNR over different constant-bit-rate background traffic

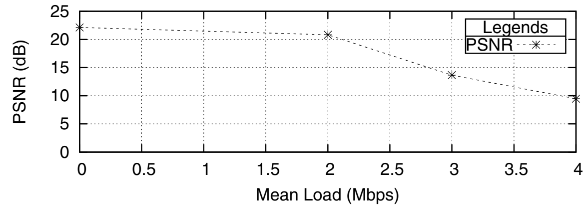

Fig. 7. PSNR over different exponential-bit-rate background traffic

Acknowledgment

This work was sponsored in part by ICL/ITRI project under number H520000560.

References

[1] IEEE 802.16 Working Group, IEEE Standard for Local and Metropolitan Area Networks Part 16: Air Interface for Fixed Broadband Wireless Access Systems. Amendment 2: Physical and Medium Access Control Layer for Combined Fixed and Mobile Operation in Licensed Bands, IEEE Std. 802.16e, December 2005

[2] WiMAX forum, http://www.wimaxforum.org

[3] NCTUns Network Simulator and Emulator, http://NSL.csie.nctu.edu.tw/nctuns.html

[4] S.Y. Wang, C.L. Chou, C.C. Lin, and C.H. Huang, "The Protocol Developer Manual for the NCTUns 6.0 Network Simulator and Emulator," available at http://NSL.csie.nctu.edu.tw/nctuns.html

[5] S.Y. Wang, C.L. Chou, and C.C. Lin, "The Design and Implementation of the NCTUns Network Simulation Engine," Simulation Modelling Practice and Theory, Vol. 15, No. 1, January 2007

[6] S.Y. Wang and C.L. Chou, "NCTUns Simulator for Wireless Vehicular Ad Hoc Network Research," a chapter of the "Ad Hoc Networks: New Research" book, (ISBN: 978-1-60456-895-0, published by Nova Science Publishers in 2008)

[7] S.Y. Wang and Y.M. Huang, "NCTUns Tool for Innovative Network Emulations," a chapter of the "Computer-Aided Design and Other Computing Research Developments" book, (ISBN: 978-1-60456-860-8, published by Nova Science Publishers in 2009)

[8] S. Mignanti, G. Tamea, I. Marchetti, M. Castellano, A. Cimmino, F. Andreotti, M. Spada, P. M. Neves, G. Landi, P. Simoes, and K. Pentikousis, "WEIRD Testbeds with Fixed and Mobile WiMAX Technology for User Applications, Telemedicine, and Monitoring of Impervious Areas," Proc. 4th International Conference on Testbeds and Research Infrastructures for the Development of Networks and Communities (TRIDENTCOM), Innsbruck, Austria, March 2008

[9] WiMAX Extension to Isolated Research Data networks (WIERD) project, http://www.ist-wierd.eu

[10] K. Pentikousis, M. Curado, P. M. Neves, and M. Katz, "Establishing the First European Research WiMAX Testbeds," the European Research Consortium for Informatics and Mathematics (ERCIM) News, No. 73, April 2008

[11] S. Hu, G. Wu, Y. L. Guan, C. L. Law, Y. Yan, and S. Li, "Development and Performance Evaluation of Mobile WiMAX Testbed," IEEE 2007 Mobile WiMAX Symposium, Orlando, Florida, March 2007