A Network Emulator on the NetFPGA Platform

Seok Hong Min, Jae Yong Lee, Byung Chul Kim
Dept. of Infocomm Eng., Chungnam National University, Daejeon, 305-764, Korea
{minsh, jyl, byckim}@cnu.ac.kr

Abstract—**Network emulators play an important role when researchers want to evaluate the performance of newly designed protocols or network mechanisms instead of deploying them in real networks, because network emulators can provide appropriate network situations, (for example, delay, bottleneck bandwidth and packet loss) needed for experiments through easy control 'knobs'. In this paper, we have implemented a network emulator on the NetFPGA platform that support exact emulation functions by hardware-accelerated packet processing. We show the pipeline architecture of the NetFPGA emulator and explain its component function. Through various performance experiments, we show that the NetFPGA network emulator has more accurate emulation performance compared to software-based network emulators such as the Dummynet [1] and the NISTnet [2].**

Keywords- Network Emulator, NetFPGA Platform, emulation performance

I. INTRODUCTION

When network researchers want to evaluate the performance of newly designed protocols or network mechanisms, they need to deploy them in real large-scale networks and do experiments they want to perform. However, it is usually very difficult to adjust network environments for experiments as they want, because it is hard to manage the real network characteristics appropriate for their experiments. In this case, network emulators can do appropriate functions by emulating a real network situations. The network emulators can control the bottleneck bandwidth, passing delay, and packet loss probability in order to emulate real network conditions.

The Dummynet [1] and the NISTnet [2] are widely used network emulators that can be installed and operated in ordinary PCs. However, the main drawback of these emulators is their performance limitations caused by software processing of control actions in rather low-performance PC platforms. Software packages running on PCs cannot guarantee exact packet transmission timing and cannot provide full line rate performance. They reveal marginal performance with almost 100% CPU utilization for two NICs with line rate 1 Gbps. Furthermore, the bottleneck of PCI bus performance on PC is another main reason for performance degradation of network emulators.

In this paper, we have implemented a network emulator on the NetFPGA platform that support exact emulation functions by hardware-accelerated packet processing. We utilized the NetFPGA reference pipeline [3] and the packet generator pipeline [4] in order to organize the architecture of the implemented network emulator. Various experiments show that the NetFPGA network emulator has more accurate emulation performance compared to software-based network emulators such as the Dummynet [1] and the NISTnet [2]. One can utilize our network emulator in experiments such as high-speed TCP protocols in Internet and satellite link that has large delay.

The paper is organized as follows. After Introduction, we explain related works in section II. We present the overall architecture of network emulator and detailed design of each component in section III. In section IV, performance comparison with software-based emulator and various experiments are shown. We conclude the paper in section V.

II. RELATED WORK

A. NetFPGA Platform

The NetFPGA platform is a network hardware accelerator that can handle packet processing at line rate without CPU participation. The PC plug-in card provides four ports of Gigabit Ethernet and includes local Static RAM (SRAM) and Dynamic RAM (DRAM) for local processing. The NetFPGA directly handles all data-path switching, routing, and processing of Ethernet and Internet packets. All the other control-path functions are handled by software.

The gateware of the NetFPGA is designed in a modular fashion to allow users to modify or reconfigure modules to implement other useful devices. Basically, a network emulator needs to have at least three traffic control functions, i.e., the control of bottleneck bandwidth, the control of passing delay, and the control of packet loss probability. We can implement an efficient network emulator by using the design of the reference pipeline of the NetFPGA as shown in Figure 1 [3], which is comprised of eight receive queues, eight transmit queues, and user data path which includes input arbiter, output port lookup and output queues.

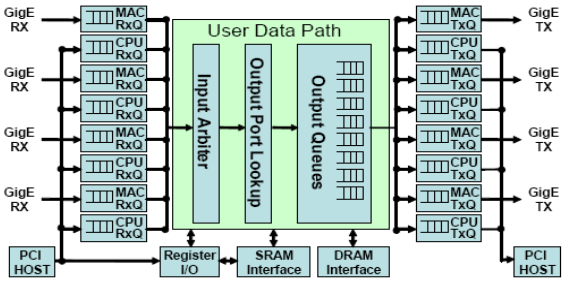

Figure 1. NetFPGA Reference Pipeline [3]

B. *Network Emulator*

Network emulation is a technique where the properties of an existing, planned network are emulated in order to assess performance, predict the impact of change without large-scale network deployment. Network emulation differs from simulation in that a network emulator appears to be a network; end-systems such as computers can be attached to the emulator and will behave as if they are attached to a network. Network simulators are typically programs which run on a single computer, take an abstract description of the network traffic and yield performance statistics. The network emulator incorporate a variety of network attributes into its emulation model including the round-trip time across the network (latency), the amount of available bandwidth, a given degree of packet loss, duplication of packets, reordering packets, and/or the severity of network jitter. A high performance network emulator plays a very important role for various network experiments of network researchers.

Network emulation can be accomplished by introducing a device on the LAN that alters packet flow characteristics in a way that imitates the behavior of application traffic in the environment being emulated. This device may be either a general-purpose computer running software to perform the network emulation such as the Dummynet [1] and the NISTnet [2], or a dedicated emulation device. In this paper, we implement a high performance network emulation device by using the NetFPGA platform on a PC.

III. NETWORK EMULATOR ARCHITECTURE

In this section, we explain the overall pipeline architecture of the implemented NetFPGA emulator and its functions. In Figure 2, we represent the overall pipeline architecture of our network emulator. We utilized the NetFPGA reference pipeline [3] and the packet generator pipeline [4] in order to organize the architecture of the emulator. There are 3 main emulation function modules in our emulator, i.e. , probability random packet drop module, delay module, and bandwidth limiter module. We explain the detailed structure of the emulator components as follows.

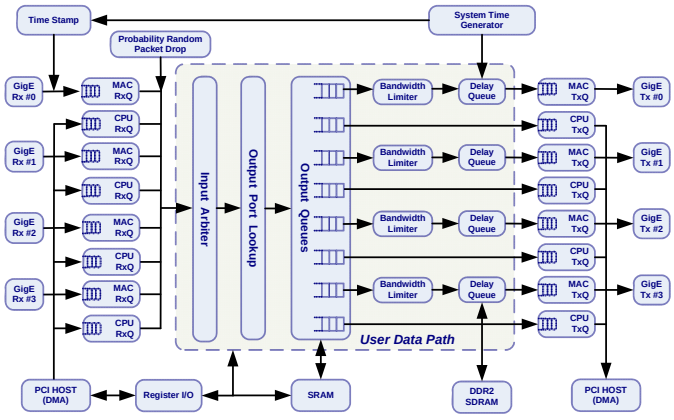

Figure 2. Overall architecture of NetFPGA network emulator

A. *Input Arbiter/Output Port Lookup/Output queues*

The input arbiter module decides the next Rx queue to be serviced, pulls and passes its packet to the next module in the pipeline. The output port lookup module decides the destination output port of the packet. After that decision is made, the packet is then handed down to an output queue module which stores the packet in its buffer corresponding to the output port until the Tx queue is ready to accept the packet for transmission. The role of these three modules is the same as the reference pipeline of the NetFPGA. The next modules after the output queues perform the various network emulation functions.

B. *Bandwidth limiter module*

Each of the four reference output queues to the Ethernet ports has the bandwidth limiter module in order to emulate the bottleneck bandwidth of each output Ethernet port individually. We utilize a token bucket model to control the output bandwidth exactly from 0 to 1 Gbps. Figure 3 show the token bucket model that generates tokens according to the bandwidth limit value and transmits arriving packets utilizing the same amount of tokens as the packet length. If there is no token, packet transmission is blocked until the required amount of tokens are generated.

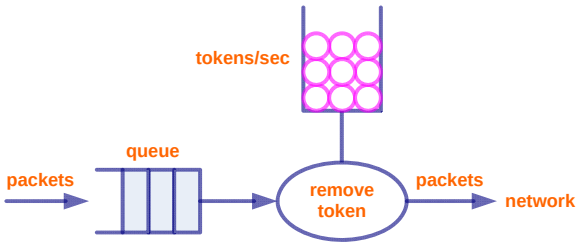

Figure 3. Token bucket model for bandwidth limiter

C. *Delay queue/ Packet delay module*

In order to give some latency to each packet before it is transmitted to the Ethernet output, we put delay queues and packet delay modules to each output queue. We can control the packet latency from 0 ns to 2^{32} ns by attaching 'timestamp' of system clock unit to each packet. Actually, the delay value of each packet is rather limited to somewhat small value, because there is basically built-in memory limitation in the NetFPGA platform. Thus, when the packet input rate is assumed to be 1 Gbps, the maximum delay we can assign to each packet is limited to about 500 msec. However, when the average bandwidth of a flow is rather limited upto one hundred Mbps, we can control the packet latency almost as we want. Further study is necessary for longer latency emulation using the NetFPGA platform.

D. *Random packet drop module*

One of main network emulation function is the control of packet loss probability occurred in the usual networks. We can control the packet loss probability in our emulator from 0.001% to 100%. We used a 21-bit LFSR (Linear Feedback

Shift Register) having polynomial as shown in (1) to implement a pseudo random number generator having the characteristics of uniform random variables.

$$P(x) = x^{21} + x^{20} + x + 1 \qquad (1)$$

For example, when we set the packet loss probability to *p*, we generate a random number from the pseudo random number generator that generates a number from 0 to 1 for each packet. If the generated number is less than *p*, the packet is dropped, otherwise, it is normally transmitted. We can change the random drop property to arbitrary burst drop property by changing the random variable characteristics.

E. Control registers

We can control each of the emulation hardware modules by setting up parameter values in controlling registers via software. The main input registers in our network emulator are the delay register, the bottleneck bandwidth register, and the loss probability register.

IV. EXPERIMENTATION

In this section, we compare the emulation performance of our NetFPGA network emulator with the software emulator 'Dummynet' [1]. We select for comparison the three performance measure, bottleneck bandwidth, passing delay, and loss probability. Overall, we can see that the performance of the NetFPGA emulator is more accurate and stable. The performance of the Dummynet has much more fluctuation and unstable characteristics. We can get the hardware acceleration benefit of the NetFPGA in the network emulator implementation.

A. Emulation of bottleneck bandwidth

Figure 4 shows that the bottleneck bandwidth performance of the two emulators. As the configuration value of bottleneck bandwidth increases up to 1 Gbps, the NetFPGA emulator can emit exact throughput almost up to 1 Gbps, but the output of the Dummynet is saturated around 700 or 800 Mbps. In the Dummynet case, the software processing can not follow the input rate 1 Gbps.

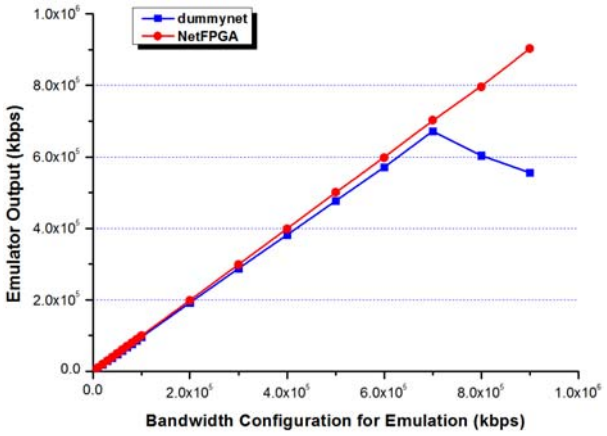

Figure 4. Bottlencek Bandwidth emulation of the two network emulators

B. Emulation of packet delay

In Figure 5, we measure the delay emulation performance of the two emulators. We use 1000 ping packets to measure the delay emulation using 10 msec delay configuration. The Figure shows that the delay value of the NetFPGA emulator is very accurate according to the configured value, but the delay performance of the Dummynet has more diverse variation

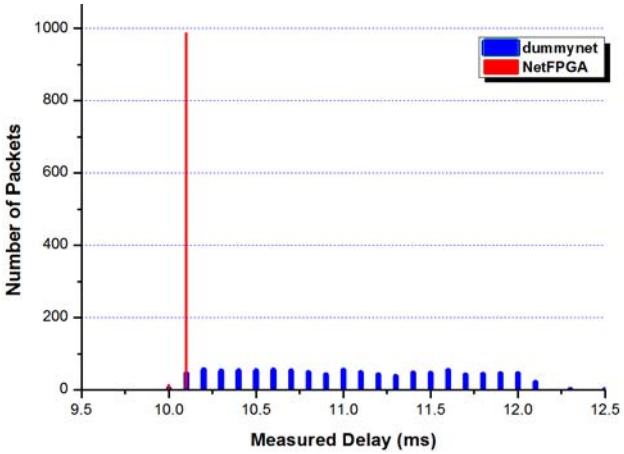

Figure 5. Delay emulation of the two network emulators

C. Emulation of packet loss

In Table 1, we show the emulation performance of packet loss probability of the two emulators. The NetFPGA emulator can control exact loss probability following the configured values. The Table shows exact mean value and very small variance of loss probability. But, the loss probability of the Dummynet does not follow the configured values and shows large variance in loss probability.

TABLE I. PACKET LOSS PROBABILITY EMULATION OF THE TWO NETWORK EMULATORS

Loss Probability Configuration		0.1%	1%	5%	10%
NetFPGA Result	mean	0.100062	1.00111	5.004734	10.11184
	variance	4.26e-11	2.39e-10	7.63e-10	2.22e-09
Dummynet Result	mean	0.099667	0.997294	4.939374	9.754387
	variance	9.72e-11	2.96e-10	2.21e-09	6.93e-09

D. Application to TCP throughput performance

As an application of our NetFPGA network emulator, we measure the throughput performance of TCP/Reno according to varying round trip time (RTT) and packet loss probability. The experiment topology is shown in Figure 6. We use two Linux PCs for the TCP sender and the receiver host and 'iperf'

[5] software is used for TCP packet generation. To emulate various network environments, we adopt our NetFPGA network emulator. We have measured the TCP/Reno throughputs and compared them with that of the simple TCP/Reno throughput equation [6] obtained by using the simple periodic protocol operation model as the following,

$$Throughput(TCP/Reno) = \frac{1}{RTT}\sqrt{\frac{3}{2p}} \qquad (2)$$

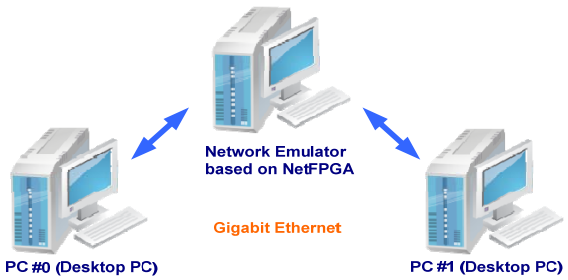

Figure 6. Experiment topology for TCP/Reno performance measurement

In Figure 7, we represent the TCP/Reno throughput measurement results for varying packet loss probability emulation, and compare it with the result of the equation (2). Originally, the equation (2) was obtained using very simple approximation of periodic model and shows accurate result only for the small loss probability. So, the two results show some deviation from each other for large loss probabilities.

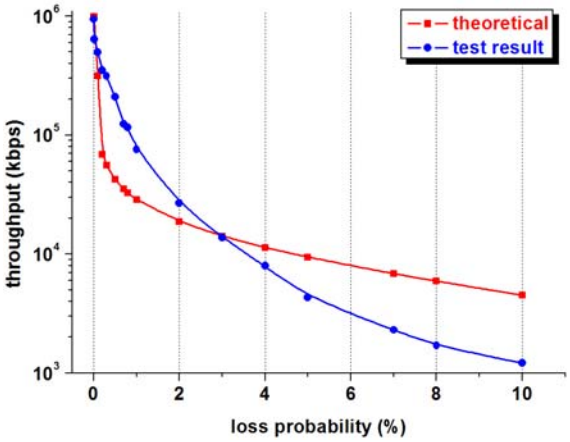

Figure 7. TCP throughput vs. packet loss probability

In Figure 8, we controlled the delay value of the emulator and measured the TCP throughput and compared it with that of the equation (2). In this experiment, we set the configuration delay of the emulator to RTT for convenience, although the actual RTT is different from that of the delay emulation value. The measurement value has very similar tendency with the theoretical result.

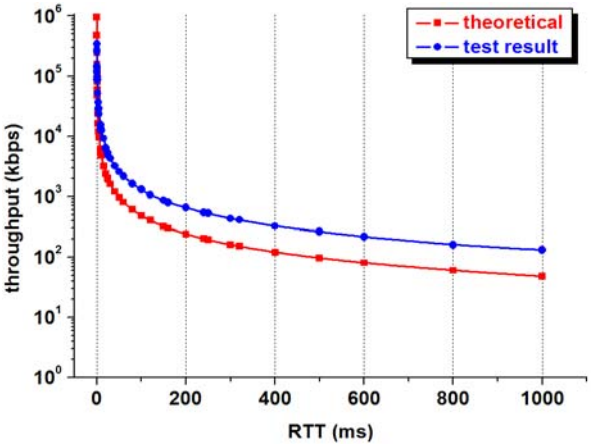

Figure 8. TCP throughput vs. RTT

V. CONCLUSION

In this paper, we have implemented a network emulator on the NetFPGA platform. We have shown its exact emulation performance compared to the software emulator Dummynet. We also show that the TCP throughput performance by using our network emulator.

The NetFPGA network emulator can be utilized for various network experiments such as performance of high-speed TCP protocol variants, emulation of long delay satellite link and etc. There is a limitation in delay emulation due to small memory size for packet storage. Extension of delay range and addition of delay jitter emulation function are for further study.

ACKNOWLEDGMENT

This paper is one of results from the project (2009-F-050-01), “Development of the core technology and virtualized programmable platform for Future Internet” that is sponsored by MKE and KCC. I’d like to express my gratitude for the concerns to support for the research and development of the project.

REFERENCES

[1] Dummynet, http://www.dummynet.com/

[2] NISTnet, http://www-x.antd.nist.gov/nistnet/

[3] NetFPGA,http://netfpga.org/foswiki/bin/view/NetFPGA/OneGig/Guide#Walkthrough_the_Reference_Designs

[4] G. Adam Covington, Glen Gibb, John Lockwood, and Nick McKeown, “A Packet Generator on the NetFPGA Platform”, IEEE Symposium on. Field-Programmable Custom Computing Machines (FCCM), April 2009.

[5] iperf, http://sourceforge.net/projects/iperf/

[6] Matthew Mathis, Jeffery Semke, Jamshid Mahdavi, Teunis Ott, “The Macroscopic Behavior of the TCP Congestion Avoidance Algorithm”, ACM SIGCOMM, vol 27, no 3, July 1997.