Experimenting with Buffer Sizes in Routers

Neda Beheshti
Stanford University
nbehesht@stanford.edu

Jad Naous
Stanford University
jnaous@stanford.edu

Yashar Ganjali
University of Toronto
yganjali@cs.toronto.edu

Nick McKeown
Stanford University
nickm@stanford.edu

ABSTRACT

Recent theoretical results in buffer sizing research suggest that core Internet routers can achieve high link utilization, if they are capable of storing only a handful of packets. The underlying assumption is that the traffic is non-bursty, and that the system is operated below 85-90% utilization.

In this paper, we present a test-bed for buffer sizing experiments using NetFPGA [2], a PCI-form factor board that contains reprogrammable FPGA elements, and four Gigabit Ethernet interfaces. We have designed and implemented a NetFPGA-based Ethernet switch with finely tunable buffer sizes, and an event capturing system to monitor buffer occupancies inside the switch. We show that reducing buffer sizes down to 20-50 packets does not necessarily degrade system performance.

Categories and Subject Descriptors

C.2.1 [**Computer-Communication Networks**]: Network Architecture and Design

General Terms

Experimentation, Performance

1. INTRODUCTION

Traditionally, it has been assumed that the required buffer size in a network is controlled by the delay-bandwidth product rule. According to this rule, with flows having an average round trip time of 100ms, a router must be able to buffer about a million packets in order to fully utilize a 10Gb/s bottleneck link. As the capacity of the network grows exponentially over time, based on the delay-bandwidth product rule, so will the buffer size requirements of core routers. Keeping up with that growth has huge implications for the design of core Internet routers.

Appenzeller *et al.* showed that we can reduce the buffer size by a factor of $\sqrt{N}$ without any degradation in performance; where N is the number of long-lived flows going through the router [1]. Thus, if the 10Gb/s linecard has 10,000 flows, we could reduce the buffers to 10,000 packets [3]. In [4] and [5], it is suggested that reducing the size down to only 20-50 packets would still result in high throughput, if the arriving traffic is paced out, either by users or by network. This result is based on the observation that the inherent bursts in the traffic will be spread out when traffic moves from slow access links to much faster core links.

ANCS'07, December 3–4, 2007, Orlando, Florida, USA.
ACM 978-1-59593-945-6/07/0012.

In this paper, we present a test-bed for experiment with buffer sizing in routers. The test-bed is based on NetFPGA; a PCI-form factor board that contains reprogrammable FPGA elements, and four Gigabit Ethernet interfaces, customized for buffer sizing experiments. The designed NetFPGA switch has finely tunable buffer sizes. One can control the buffer sizes (in bytes or packets) with high precision, without the worry of hidden buffers in the system. An event capturing system is designed which records every drop and store event inside the switch with high resolution time stamps.

2. NETFPGA SWITCH

Fig. 1(a) shows a block diagram of the NetFPGA platform. The custom four-port learning switch is an output queued Ethernet switch which implements switching through the SRAM. An `event_capture` module is designed to snoop signals in the switch and record the precise time when they pulse. The events are aggregated into an event packet and then sent out through a specified port.

The block diagram in Fig. 1(b) shows the major blocks of the switch. The bold arrows specify the path of packet data. Packets arrive from the MAC into four receive queues. The `input_manager` implements a round-robin arbiter that selects a receive queue to read a packet from and sends the packet to the user data path.

The user data path contains the main functionality of the switch. The packet's output queue is looked up in the `MAC_table`, and then the packet is passed to the `store_egress` module which stores it in SRAM. The `send_egress` module implements a round-robin arbiter which waits for a packet in the SRAM output queues, then sends it out to the `output_manager` which in turn passes it out to the correct output port. The `event_capture` module snoops on signals from `send_egress` and `store_ingress` indicating packet storage, packet removal and packet drops. The time when these signals occur is stored along with the length of the packet and the output queue affected in an event record. The event records are aggregated into an event packet that is stored in an output queue on the FPGA (not in SRAM). This queue is serviced in round-robin order along with the SRAM output queues by the `send_egress module`.

Using the management interface of the switch, it is possible to change the size of the queues, and to impose limits on the number of packets stored in the output queues.

3. BUFFER SIZING EXPERIMENTS

Experiment Setup- Fig. 2 depicts the setup of the test-bed network. In this network, a single point of congestion is

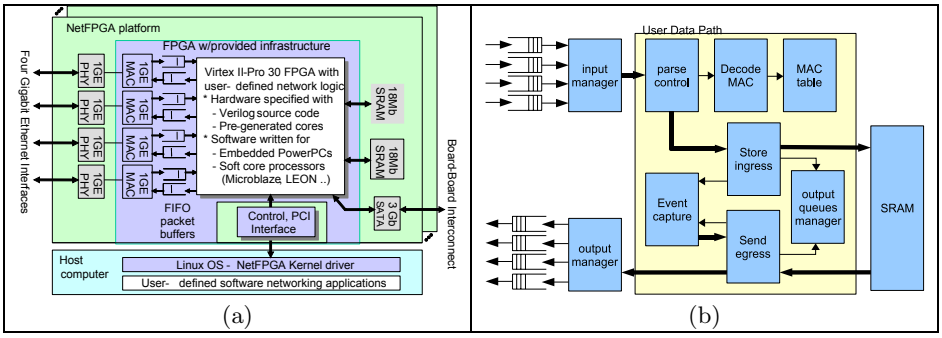

Figure 1: **(a) NetFPGA 2.0 block diagram, (b) Switch with event capture subsystem.**

formed, where packets from several TCP servers go through the NetFPGA switch, and share a bottleneck link towards their destinations. TCP traffic is generated by the Spirent Communication's Avalanche and Reflector boxes [1]. The Avalanche box creates a number of clients which request for downloading files from TCP servers. The TCP servers belong to 99 different sub-networks, and are connected to the output ports of the Reflector via emulated 20Mb/s access links. The aggregated traffic is then carried to the NetFPGA switch over three 1Gbps links, and from the switch to the client network over a 1Gbps bottleneck link. The average round trip time delay of TCP flows is set to 80ms.

Experiment Results- Fig. 3 shows the throughput of the

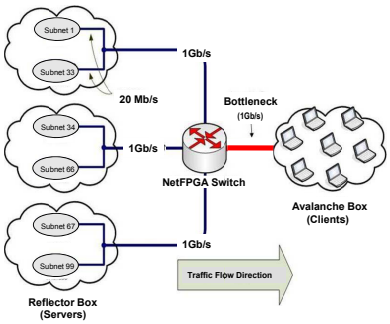

Figure 2: **Experiment setup**

bottleneck link in a time interval of 320 seconds, and with three different buffer sizes. The size of buffer in the designed NetFPGA switch is accurately controllable and can be reduced to zero. As the graph shows, the link utilization drops down by about 17%, when the router reduces the buffer size from 300 packets to only 5 packets. As discussed before, high utilization on the bottleneck link can be achieved with very small buffers if the TCP traffic is not overly bursty. Access links of limited capacity eliminate possible bursts in the traffic. In the next set of experiments, throughput is measured as a function of access bandwidth (right figure) with 50-packet buffers. We observe that a very small access capacity limits the throughput in the access links, and reduces the aggregate throughput. On the other hand, when the access link capacity gets close to the core link capacity, some portion of the throughput is lost due to a more bursty traffic on the shared link. The multiplexing of different flows

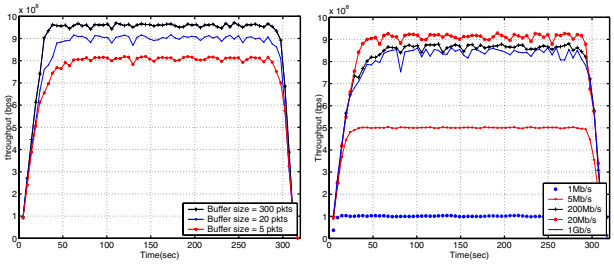

Figure 3: **Throughput vs. time, as a function of buffer size (left), and access bandwidth (right).**

in this experiment, however, doesn't let the throughput drop down drastically, even when the access link capacity is equal to the core capacity.

The results of our experiments on the topology of Fig. 2 show that when the TCP traffic comes from slow access links (having an access to core capacity ratio of 2% in our setup), 90% utilization is achieved with 20-packet buffers. Under the same conditions, the buffer sizing rule-of-thumb would have overestimated the required buffer size to be 10,000 packets for achieving 100% throughput.

4. CONCLUSIONS

The results of our experiments in a NetFPGA-based testbed show that reducing buffer sizes down to 20-50 packets does not necessarily degrade the bottleneck link utilization. These results are inline with small buffer sizing theories. The designed and implemented NetFPGA Ethernet switch has finely tunable buffer sizes, and an event capturing system that monitors buffer occupancies inside the switch with 16ns time resolution. Both the above features are extremely important in any buffer sizing experiment, but do not exist in current commercial routers.

5. REFERENCES

[1] Avalanche traffic generator. http://www.spirentcom.com.
[2] NetFPGA. http://yuba.stanford.edu/NetFPGA/.
[3] G. Appenzeller, I. Keslassy, and N. McKeown. Sizing router buffers. In *Proceedings of the ACM SIGCOMM*, pages 281–292, 2004.
[4] M. Enachescu, Y. Ganjali, A. Goel, N. McKeown, and T. Roughgarden. Routers with very small buffers. In *Proceedings of the IEEE INFOCOM*, Barcelona, Spain, April 2006.
[5] G. Raina, D. Towsley, and D. Wischik. Part II: Control theory for buffer sizing. *ACM/SIGCOMM Computer Communication Review*, 35(3):79–82, July 2005.