High-Fidelity Link Shaping

Roman Chertov, Kevin Almeroth
Department of Computer Science
University of California, Santa Barbara
Santa Barbara, CA 93106-5110 USA
Email: {rchertov, almeroth}@cs.ucsb.edu

***Abstract*—Network testbeds are designed to act as highly flexible experimentation platforms, suitable for a broad range of network experiments. One of the key requirements of network testbeds is to provide a link shaping ability to the experimenter. In this work, we conduct an in-depth overview of three link shaping approaches and provide experimental results of their performance in terms of achieving desired link properties and emulation artifacts. The results indicate that a transparent delay node running a kernel-level Click modular router significantly outperforms the other two methods.**

***Index Terms*—emulation, testbeds, link-shaping**

I. INTRODUCTION

Network testbeds are an invaluable tool for conducting experiments with real hardware and software in a controlled, repeatable, and easily manageable environment. Typically, testbeds employ an emulation component to control the delay and bandwidth of the experimental links. On most testbeds, the nodes are only a few milliseconds away from each other and the link speed is either 100 or 1000 Mbps. Such an arrangement is acceptable when testing protocols in a LAN environment; but is insufficient for protocols that need to be tested in networks that have long distance links and experience much higher delays. For instance, networks with Geostationary Earth Orbit (GEO) satellite links can experience a delay of 125 ms per hop. Additionally, networks with satellite links can have a significant variation in the throughput capacity. Hence, providing the capability to vary the delay and bandwidth is paramount when conducting experiments that require an emulated Wide Area Network (WAN).

Typically, artificial propagation delays and bandwidth restrictions are created via emulation tools. Using emulation tools to change the physical link properties is commonly referred to as *link shaping*. The accuracy and fidelity of the tools is critical to a majority of networking experiments. If an emulation tool induces losses or heavy jitter when such events are not desired or expected in the experiment, the resulting data and its subsequent analysis will be flawed. For instance, losses due to emulation artifacts will impact congestion control protocol studies as the protocols will react to losses that should not occur. High variability in delay or jitter is undesirable and can impact experiments with real time protocols such as Voice over IP (VoIP) as well as affect round trip time calculations.

In the networking literature, common tools include DummyNet [9], NIST-Net [8], Click modular router [7], and Netpath [1]. All of these tools have the common property that they need to be used as transparent bridges between two nodes to provide the desired link characteristics. Naturally, some of the tools offer higher fidelity than others. For example, shaping methods such as DummyNet [9] and Linux traffic control (tc) have been found to induce emulation artifacts due to bursty behavior as well as not always being true to the desired properties [6], [1], [4].

Emulation testbeds can range from very small lab networks to large facilities such as Emulab [10]. Testbeds can also contain a wide selection of hardware with varying capabilities. Such variability can lead to cases where some of the tools are more appropriate than others. Hence, in this paper, we are primarily interested in comparing different *shaping methods, rather than comparing the individual tools*. Link shaping over Ethernet can be provided with three methods: (1) a transparent delay node, (2) rate limiting at the transmitting network node, and (3) IEEE 802.3x Ethernet flow control.

To conduct a qualitative comparison of the three different shaping approaches, we utilize a measurement tool capable of microsecond level precision. The tool allows us to ascertain if the desired delay has been achieved and how much the delay varies. We also measure inter-frame gaps to gauge the presence of jitter in Constant Bit Rate (CBR) flows. Finally, we record packet losses to determine if the tool is dropping packets when loss is not desired. We first perform a comparison between the three different methods, then we choose the best method to shape a 1 Gbps Ethernet output link of a Cisco 2851 router to provide a 125 ms delay and a 155 Mbps bandwidth limit.

The remainder of this paper is organized as follows. Section II provides a detailed overview of link shaping methods. Section III provides the details of our experimental methodology, tools, and the testbed. Section IV provides the calibration results of our measurement tool. Section V discusses the results obtained with various link shaping methods. Finally, Section VI concludes this paper and summarizes our findings.

II. LINK SHAPING TAXONOMY

This section provides an in depth overview of the three Ethernet link shaping approaches: transparent delay nodes, rate limited output links, and IEEE 802.3x flow control via Ethernet pause frames.

A. Transparent Delay Node

Transparent bridging is a popular link shaping approach. An extra delay node that runs a link emulator is used to pass Ethernet frames from one node to another. When the delay node receives a frame, it first sends the frame through a rate limiter, then it delays the frame by the specified amount. The architecture of a delay node is illustrated in Figure 1. If the ingress rate exceeds the specified bandwidth, the delay node first queues the frames to ensure that the egress interface maintains the desired rate. If the ingress rate does not decrease, the queue overflows, and the incoming frames get dropped. From the point of view of the non-delay network nodes, the frames are either lost or delayed.

Popular examples of such link emulators are DummyNet [9], NIST-Net [8], and the Click modular router [7]. Click modular router has a powerful multi-threading feature and can be configured to utilize multiple CPUs to ensure that flows in one direction do not affect the flows in the other direction. The major drawback of link emulators is that they require an additional node, and hence an increase in configuration complexity as transparent bridges have to be created.

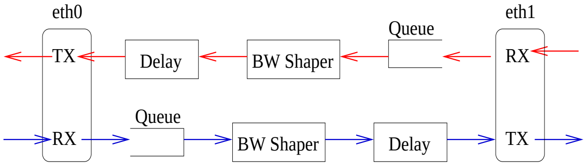

Fig. 1. Transparent bridge link shaper.

B. Rate Limiting the Output Link

Another way to provide the desired link characteristics is to rate limit the output interface of the node itself. For example, Linux supports shaping via Class Based Queuing (CBQ) [6] and Cisco routers provide a `rate-limit` command [5]. The obvious drawback of this approach is the fact that the node must provide the rate limiting capability. Providing rate limiting taxes system resources and can potentially lead to emulation artifacts. The ability to perform only output link rate limiting makes it impossible to add artificial propagation packet delays. If the delays cannot be achieved on the node itself, a transparent delay bridge is required.

C. Pause Frames

The IEEE 802.3x standard, which specifies flow control for Ethernet, can be used to rate limit and create delay on a link. The IEEE standard specifies that Ethernet pause frames can be sent to a sender from a receiver to temporarily halt transmissions. A pause frame specifies a time during which the sender should not send any packets. The time is specified in quanta values, where a single quanta pauses the sender by the time it takes to transmit 512 bits. For example, if a receiver on 1 Gbps link sends a frame with a quanta value of 195, the sender must block for 100 microseconds[1]. Hence, the rate of the pause frames and their quanta values dictate the resulting bandwidth and delay. Obviously, the pause frame approach is based on the fact that the hardware to be paused is capable of dealing with microsecond precision timers and supports flow control.

III. EXPERIMENTAL SETUP

To test which link shaping approach will produce the fewest number of emulation artifacts, we created a small testbed, shown in Figure 2. The testbed has one Cisco 2851 router and two Dell Symmetric MultiProcessor (SMP) PCs. One PC acts as a traffic source/sink and has two quad-core 1.86 GHz Intel Xeon processors. The other PC acts as a bridge/packet counter and has two dual-core AMD Opteron 2212 processors. Additionally, the PC with the AMD processors has the node interleaving memory option enabled. The equipment is directly connected as shown in Figure 2 using Gigabit Ethernet.

In our experiments, our ultimate goal is to shape the output link of the Cisco 2851 router to 155 Mbps with 125 ms of delay, while keeping emulation artifacts at a minimum. We chose the bandwidth limit of 155 Mbps and the propagation delay of 125 ms because these parameters correspond to a very long-range optical link, for example, a free space laser communication link between satellites. Prior to experimenting with this scenario, we need to closely examine the three link shaping methods described previously.

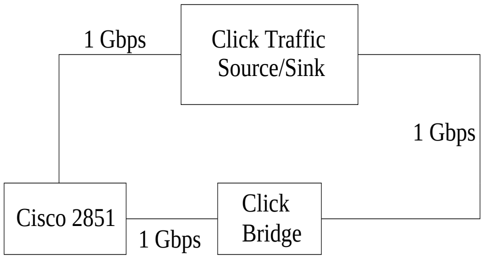

Fig. 2. Testbed topology with a commercial Cisco router and SMP Dell PCs.

A. Traffic Generator/Analyzer

To gauge the effectiveness of each method, we used a high precision packet generator/analyzer. The tool is called the Black Box Profiler (BBP) [2]. The layout of the BBP is shown in Figure 3. The BBP is configured to act as a traffic source and as a traffic sink. The device driver is modified to embed timestamps into packets as they are sent and received. The traffic flow over the network was configured such that the packet path would originate and finally terminate at the BBP. As the frames originate and terminate at the BBP node, no clock synchronization is required to obtain the packet time in the network with microsecond-level precision. Additionally, it is possible to compute the inter-frame gap as the frames leave and enter the system.

The ability of the BBP to compute the delay of packets in the network and the inter-frame gaps allows us to gauge the performance of a link shaping scheme. For instance, if the overall packet delay is much larger than expected or there is a significant variance in the delay, the link shaping scheme is not performing well and is producing emulation

---

[1] $\frac{195*512b}{1e9\ bps} \times \frac{1e6\ \mu s}{sec} = 100\ \mu s.$

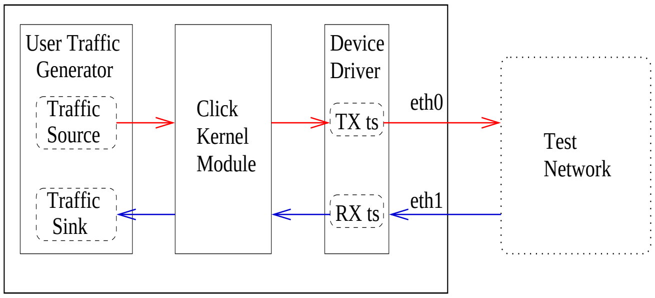

Fig. 3. Layout of the BBP traffic generator/logger.

artifacts. In addition, the inter-frame gap is an indication of whether the desired rate is achieved or not, and how much jitter an emulation tool induces. Ideally, the inter-frame gap for a Constant Bit Rate (CBR) flow is constant.

In our experiments, we created a CBR stream of 1,000,000 UDP packets. We varied the packet sizes and packet rates to explore a wide variety of configurations. Using identically sized packets simplifies fidelity testing of each link emulation method as there is no variance in the packet delays. We also repeat the same experiment five times and report the results across all of the experimental runs.

B. Shaping Implementations

This section provides an overview and configuration of the link shaping methods that we have implemented, evaluated, and compared.

1) Transparent Delay Node: A node acting as a transparent Ethernet bridge can delay and rate limit frames from one interface before passing them to the other interface. Figure 1 demonstrates such functionality. In our experiments, we configured the Click modular router to perform this task. We also configured the Click modular router to utilize two CPUs, such that each port is assigned to a dedicated CPU. Assigning a CPU per port removes the possibility of heavy congestion on one port affecting the traffic on the other port. To provide link shaping in the Click modular router, we utilized the *LinkUnqueue* element.

Besides using a kernel-level Click module to perform the shaping, we were interested in comparing it to *LinkEm*, a link emulation utility developed by MIT Lincoln Labs. Unlike the Click modular router, *LinkEm* runs in user-level mode and bridges two network interfaces by relying on a raw socket to capture Ethernet frames. Running a bridge in user-level can be detrimental to high speed packet forwarding, as packets have to be moved from kernel space to user space, hence sacrificing efficiency. On the other hand, running a user-level program is far simpler than using a kernel-level module and can be very attractive to inexperienced users. In addition, *LinkEm* provides a significant library of satellite link shaping models, hence we were interested in including it in our evaluation study.

As stated in Section II-A, the main drawback of a transparent delay node is the fact that extra hardware is needed and that frame drops occur at a delay node and not at the device whose link is being shaped. In a heavy congestion scenario, it does not matter if either the router or the delay node drops the frames. However, in a low load but bursty scenario, the delay node might drop more frames if it has smaller buffers than the router. Alternatively, the delay node can drop fewer frames, if its buffers are larger than that of the router. This effect can possibly be eliminated by profiling a router first to ascertain its buffer sizes [3].

2) Router Rate Limiting: The Cisco 2851 router has an ability to limit the output and input rates of an interface via the `rate-limit` command. The router however does not have a feature to add artificial delays. Hence, an additional delay node is still required. However, in this case, frames will be dropped on the router if the rate is exceeded. In our experiments, we tested both the router's ability to rate limit its output without using the delay node as well as experiments with the delay node. Testing without the delay node is necessary to ascertain the performance of a rate limiter as the extra delay node can produce an additional measurement noise.

3) Ethernet Pause Frames: The IEEE 802.3x standard specifies a flow control mechanism via Ethernet control frames. One node can send a frame to another node to pause its transmission for a specified duration of time. Changes in the duration and the rate of the pause frames can be used to achieve a desired link bandwidth. Additionally, the pause duration can be set to the desired link delay. The problem with using pause frames is that they can introduce burstiness into the packet flow, as the link operates in an on/off mode; however, if only a coarse-grained emulation is necessary, it should be possible to achieve the desired delay and bandwidth without noticeable artifacts.

Pause frames require no changes in the Cisco router that we used, and all of the frame drops will occur at the router, which is an ideal scenario. However, a pause frame generating node is required. Figure 4 demonstrates the layout of the generator that we implemented. Just as in the transparent delay node case, we used the Click modular router and configured it to act as a bridge. In addition to bridging, we created a pause frame generator that emits Ethernet pause frames with a specific quanta and at a given constant rate[2]. The pause frame generator emits pause frames in the opposite direction of the measured packet flow, hence switching off the transmitting interface on the router for a specified duration. In our experimental topology depicted in Figure 2, the node labeled as *Click Bridge* can be configured to run as a pause frame generator.

IV. CALIBRATION

Prior to conducting our comparison experiments, we performed a calibration test of the BBP on our traffic generation PC. To perform the calibration, we connected two Intel Pro cards with a single cable and ran a series of tests where we varied the packet size and rate. The purpose of the test was to ascertain how much overhead the system adds to the overall packet delay. Ideally, the system should add no delay and

---

[2] *EtherPauseSource* element in the current Click source tree.

TABLE I
NIC-TO-NIC: INTER-FRAME GAPS (μs) FOR 64-, 800- AND 1518-BYTE ETHERNET FRAMES

	64 bytes			800 bytes			1518 bytes		
Rate	10^{th}	50^{th}	90^{th}	10^{th}	50^{th}	90^{th}	10^{th}	50^{th}	90^{th}
8000	123	125	127	123	125	127	123	125	127
40000	23	25	27	19	25	31	20	25	31
80000	7	12	18	8	12	16	11	12	14
120000	5	8	12	6	8	11			
200000	4	5	6						

TABLE II
NIC-TO-NIC: PACKET DELAYS (μs) FOR 64-, 800- AND 1518-BYTE ETHERNET FRAMES

	64 bytes			800 bytes			1518 bytes		
Rate	10^{th}	50^{th}	90^{th}	10^{th}	50^{th}	90^{th}	10^{th}	50^{th}	90^{th}
8000	6	7	8	14	15	16	22	23	24
40000	6	7	8	14	15	20	22	23	28
80000	6	7	10	15	16	17	24	29	34
120000	6	7	10	15	16	18			
200000	6	7	8						

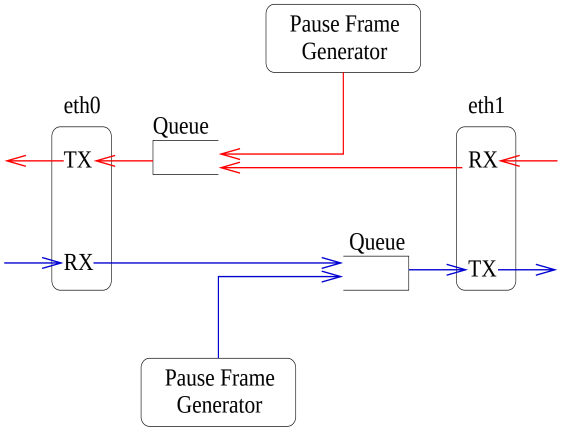

Fig. 4. Transparent bridge and IEEE 802.3x Ethernet pause frame generator.

the packet delay would be equal to the transmission delay. Also, we wanted to understand whether there were significant variations in any added delay. If the delay variation was low, the additional delay could be treated as a constant.

R. Chertov *et al.* showed that the BBP can perform well under a wide variety of loads, without inducing a significant level of measurement noise [2], [3]. In addition, the previous results showed that the noise variance is low. However, since the experimental platform in the previous studies had different hardware, it was of interest to perform the calibration tests on the new platform.

To conduct the calibration, we created UDP packets such that the resulting Ethernet frame sizes were 64-, 800-, and 1518-bytes. For each packet size, we set the packet rate to 8000, 40000, 80000, 120000, and 200000 packets per second. In cases where the resulting bandwidth is greater than the available bandwidth, we do not run the experiment. Such an arrangement allows us to explore a wide variety of byte and packet rates.

Tables I and II show the inter-frame gaps and packet delays in μs for the 10^{th}, 50^{th}, and 90^{th} percentiles, respectively. In cases where the byte rate exceeded link capacity, we have left the table entries blank. Inter-frame gaps shown in Table I represent the time between frames as they enter the BBP. Ideally, the difference between the 10^{th} and 90^{th} percentiles should be small, and the mean should be the same as the computed theoretical μs inter-frame gap for a given rate, which is computed as:

$$\frac{1}{packet\_rate} \times \frac{1e6\ \mu s}{sec}.$$

The data in Tables I and II indicates that the system performed exceptionally well: the difference between the 10^{th} and 90^{th} percentiles is very small. Additionally, the 50^{th} percentile for inter-frame gaps is the same as the theoretical value.

The delay that packets experience when going over the cable between the two network cards should ideally be equal only to the transmission delay over a Gigabit link. Any additional delay besides that value implies artifacts in the system. In the case of the BBP, the additional delay arises from the fact that the packet delays include the transmission over the PCI-E bus and the network card. However, if the difference between the 10^{th} and 90^{th} percentiles is small, then the additional noise can be treated as a constant. The data in Table II implies that the additional delay in the system does not vary significantly and can, therefore, be treated as a constant.

The results described above demonstrate that the BBP system provides a good level of precision, sufficient to conduct evaluation studies of various link shaping techniques. Additionally, no unexpected packet loss was detected, meaning that the system is capable of maintaining very high packet rates without losing packets.

V. EXPERIMENTAL RESULTS

This section provides the experimental comparison of the three link shaping methods. We compare the shaping methods based on the presence of the following emulation artifacts: loss, high jitter, and incorrect link shaping. A method which exhibits minimal emulation artifacts is then selected to carry out our original goal to shape an output link of a Cisco 2851 router to provide a 125 ms propagation delay and a bandwidth limit of 155 Mbps.

A. *Transparent Delay Node*

As was mentioned in Section III-B1, we chose two tools to emulate a link, *LinkEm* and the Click modular router. We used the topology shown in Figure 2, except without the Cisco router. Removing the Cisco router was necessary to measure the delay node only without the additional noise introduced by the router.

For both tools, we conducted two sets of tests. In the first test, the node adds no delay and purely bridges. The purpose of the test is to ascertain the level of additional packet delay due to bridging and to determine the variability of the inter-frame gaps due to processing at the delay node. In the second test, the node is configured to provide a link delay of 125 ms and maintain a 1 Gbps bandwidth. The test is aimed at determining if the desired link delay can be maintained, and if the variance of the inter-frame gaps increases. Finally, both tests must produce no packet loss. Packet loss is considered an artifact in such a scenario.

1) LinkEm:

LinkEm is simple to use. It runs at the user-level and has a wide variety of satellite link models, hence, we were interested if it could perform at Gigabit rates without introducing artifacts. At first, we configured *LinkEm* to produce zero delay and ran a single UDP flow with varying packet sizes and rates. To ensure that *LinkEm* would not suffer from scheduling artifacts, we configured it to run at priority level -20 via the Linux “`nice`” command. Table III, Table IV, and Table V show the results for loss ratios, inter-frame gaps, and packet delays, respectively.

One important result is that at packet rates larger than 40 kilo-packets per second (Kpps), *LinkEm* starts to lose packets. We used “`ifconfig -s`” to determine if the losses were occurring on the ingress or the egress interfaces. The ingress interface reported zero loss. On the other hand, the egress interface did not transmit all of the packets that were received, meaning that the packets were lost in transit between the interfaces.

The results also reveal a large degree of packet jitter (variance in inter-frame gaps) and variance in the packet delays. Delay data for 120 Kpps and above indicates a significant increase in delay compared to lower packet rates. The additional delay is most likely due to queuing delays, since the tool cannot forward packets fast enough, hence leading to queue buildups. The experimental data indicates that *LinkEm* adds a significant number of emulation artifacts and is not an appropriate tool for high data rates.

Next, we configured *LinkEm* to emulate a 125 ms delay and no loss. As before, we ran the same set of experiments. Tables VI, VII, VIII show the results for packet loss ratios, inter-frame gaps, and delays, respectively.

One important observation of these results is that every single experiment resulted in some level of packet loss. The inter-packet gap values are similar to the previous experiments when zero delay was used. The packet delays are close to the desired 125 ms delay for packet rates under 120 Kpps. For rates 120 Kpps and above, the packet delay is larger than 125 ms. Just as before, the increase in delay is most likely due to queuing delay.

The experiments with 0 ms and 125 ms delays revealed that *LinkEm* adds a significant number of emulation artifacts, which include packet loss, jitter, and delay. Hence, *LinkEm* is not a good choice for high fidelity, high data rate emulation experiments.

2) Click Modular Router:

Since *LinkEm* runs in user-space, we were interested in using the Click modular router kernel module. In order to take advantage of the multiple cores available on the node, we configured Click to assign each packet path to a separate CPU. As the packet handling code operates in the kernel, there is no overhead of copying data from the kernel to the user-space. Hence, we expect the Click modular router to provide low jitter, low packet delay, and no loss.

Tables IX and Table X show the results with a Click bridge providing no emulated delay. As there was no packet loss, we did not include a loss ratio table. The data for the inter-frame gaps and delays reveals that the difference between the 10^{th} and 90^{th} percentiles has only slightly increased from the calibration values in Section IV. However, the level of noise induced by the Click modular router is significantly smaller than the noise induced by *LinkEm*.

The results for inter-frame gaps and packet delays with Click, configured to provide 125 ms of delay, are shown in Tables XI and XII. There was no loss of packets; the queue size was set to the bandwidth delay product. The data for inter-frame gaps and packet delays indicates relatively small differences between the 10^{th} and 90^{th} percentiles. Also, it can be seen that the packets did experience the 125 ms delay with a variance of a fraction of a millisecond. Since the desired delay of 125 ms is overwhelmingly larger than the sub millisecond variance, Click’s emulation performance is therefore quite good. The Click modular router performed significantly better than *LinkEm* by providing the desired link delays and not introducing packet losses. Hence, in the remainder of this paper, we will only use Click for link delay and shaping.

To further ascertain the applicability of the Click modular router as a link shaper, we conducted bandwidth shaping experiments. We modified the configuration file to provide a rate limit of 155 Mbps. This value is equivalent to the speed of an OC-3 link. To gauge the effectiveness of bandwidth limiting, we ran the same set of experiments as before.

Table XIII lists the inter-frame gaps. The theoretical minimum inter-frame gaps corresponding to using the entire 155 Mbps link capacity are 3.30, 41.28, and 78.34 for 64-, 800-, and 1518-byte Ethernet frames respectively. The maximum packet rate for 64-byte frames at 155 Mbps could not be achieved, as it is over 300 Kpps and the logging tool cannot handle more than 200 Kpps. However, for 800- and 1518-byte frames, the maximum rate had been achieved. The data in Table XIII further indicates that the desired link shaping was achieved.

Table XIV shows that for 800-, and 1518-byte packets the delays are larger than the specified 125 ms. This result occurs because the packets experienced queuing delay due to a reduction of bandwidth, as shown in Figure 1. The size of the queue can be adjusted to any value. If the queue is very small, then the total delay would be close to the target value. However, if the queue is small, the link shaper might drop packets that come in bursts. Ideally, *the queue size should*

TABLE III

LINKEM 0 MS DELAY: PACKET LOSS RATIOS (%) FOR 64-, 800- AND 1518-BYTE ETHERNET FRAMES

Rate	64	800	1518
8000	0.0000	0.0000	0.0000
40000	0.0000	0.0000	0.0000
80000	2.4797	1.3296	3.4842
120000	11.6460	12.0584	
200000	37.6603		

TABLE IV

LINKEM 0 MS DELAY: INTER-FRAME GAPS (μs) FOR 64-, 800- AND 1518-BYTE ETHERNET FRAMES

	64 bytes			800 bytes			1518 bytes		
Rate	10^{th}	50^{th}	90^{th}	10^{th}	50^{th}	90^{th}	10^{th}	50^{th}	90^{th}
8000	99	145	151	99	145	151	99	144	151
40000	5	22	45	7	22	43	11	12	49
80000	5	10	30	6	9	30	11	12	14
120000	4	7	20	6	7	15			
200000	4	6	12						

TABLE V

LINKEM 0 MS DELAY: PACKET DELAYS (μs) FOR 64-, 800- AND 1518-BYTE ETHERNET FRAMES

	64 bytes			800 bytes			1518 bytes		
Rate	10^{th}	50^{th}	90^{th}	10^{th}	50^{th}	90^{th}	10^{th}	50^{th}	90^{th}
8000	31	51	71	47	67	87	64	85	104
40000	34	49	62	57	72	86	178	228	284
80000	41	56	5108	65	78	89	246	289	1056
120000	45	60	4276	77	90	799			
200000	2101	2880	3613						

TABLE VI

LINKEM 125 MS DELAY: PACKET LOSS RATIOS (%) FOR 64-, 800- AND 1518-BYTE ETHERNET FRAMES

Rate	64	800	1518
8000	1.1798	1.1767	1.1807
40000	0.7923	0.7986	0.2206
80000	10.8434	15.4696	21.2297
120000	28.9757	41.1499	
200000	57.9171		

TABLE VII

LINKEM 125 MS DELAY: INTER-FRAME GAPS (μs) FOR 64-, 800- AND 1518-BYTE ETHERNET FRAMES

	64 bytes			800 bytes			1518 bytes		
Rate	10^{th}	50^{th}	90^{th}	10^{th}	50^{th}	90^{th}	10^{th}	50^{th}	90^{th}
8000	98	144	152	98	144	152	98	144	152
40000	4	23	46	7	25	44	11	13	48
80000	4	12	30	6	12	28	11	13	23
120000	4	10	19	6	12	25			
200000	3	10	27						

TABLE VIII

LINKEM 125 MS DELAY: PACKET DELAYS (μs) FOR 64-, 800- AND 1518-BYTE ETHERNET FRAMES

	64 bytes			800 bytes			1518 bytes		
Rate	10^{th}	50^{th}	90^{th}	10^{th}	50^{th}	90^{th}	10^{th}	50^{th}	90^{th}
8000	125040	125060	125080	125059	125079	125097	125073	125093	125115
40000	125040	125060	125078	125063	125081	125098	125190	125247	125298
80000	125051	125074	130723	125080	125150	126242	125275	125995	126623
120000	128122	128343	130169	125745	125887	126296			
200000	128105	129286	131500						

be the same capacity as on the device whose link is being emulated.

Table XIII demonstrates the percentile statistics for the inter-frame gap data. However, it is sometimes valuable to look at the raw data. Figure 5 demonstrates a small window of sequential packets and their inter-frame gap values, after link shaping has been applied to a 1518-byte flow at 80 Kpps. The plot looks like a scatter plot, as no packets with consecutive sequence numbers were received. The inter-frame gaps in the graph have a maximum variation of 10 μs and are roughly centered around the desired 78 μs line, meaning that the link shaper has performed an adequate task of evenly spacing the frames in time. This indicates that using the Click modular router as a transparent link shaping bridge will yield excellent performance with negligible emulation artifacts.

B. Router Rate Limiting

Output link shaping can be performed on a testbed node itself. In our case, we utilize a Cisco 2851 router. Output link shaping can potentially reduce experimental complexity, as no extra link shaping hardware is required, and there is no need to match the size of the link shaping node's queue to that of the router.

Prior to conducting an experiment with a rate limiter, we conducted a test where we measured the inter-frame gaps and packet delays with a baseline configuration. The router under test was a Cisco 2851 as shown in Figure 2. For the

TABLE IX

CLICK 0 MS BRIDGE: INTER-FRAME GAPS (μs) FOR 64-, 800- AND 1518-BYTE ETHERNET FRAMES

	64 bytes			800 bytes			1518 bytes		
Rate	10^{th}	50^{th}	90^{th}	10^{th}	50^{th}	90^{th}	10^{th}	50^{th}	90^{th}
8000	121	125	129	121	124	129	121	126	129
40000	21	25	30	12	26	31	13	25	33
80000	4	12	16	7	12	19	11	13	14
120000	3	8	14	6	7	12			
200000	2	5	9						

TABLE X

CLICK 0 MS BRIDGE: PACKET DELAYS (μs) FOR 64-, 800- AND 1518-BYTE ETHERNET FRAMES

	64 bytes			800 bytes			1518 bytes		
Rate	10^{th}	50^{th}	90^{th}	10^{th}	50^{th}	90^{th}	10^{th}	50^{th}	90^{th}
8000	34	35	38	56	58	60	78	79	82
40000	34	36	41	57	70	85	79	93	108
80000	22	27	32	48	55	62	90	96	105
120000	20	23	29	47	53	61			
200000	22	26	31						

TABLE XI

CLICK 125 MS BRIDGE: INTER-FRAME GAPS (μs) FOR 64-, 800- AND 1518-BYTE ETHERNET FRAMES

	64 bytes			800 bytes			1518 bytes		
Rate	10^{th}	50^{th}	90^{th}	10^{th}	50^{th}	90^{th}	10^{th}	50^{th}	90^{th}
8000	111	126	141	111	127	141	111	126	141
40000	21	25	29	20	24	29	13	24	34
80000	2	12	22	7	12	20	11	13	14
120000	2	8	16	6	7	12			
200000	2	4	10						

TABLE XII

CLICK 125 MS BRIDGE: PACKET DELAYS (μs) FOR 64-, 800- AND 1518-BYTE ETHERNET FRAMES

	64 bytes			800 bytes			1518 bytes		
Rate	10^{th}	50^{th}	90^{th}	10^{th}	50^{th}	90^{th}	10^{th}	50^{th}	90^{th}
8000	125083	125152	125224	125107	125176	125247	125127	125196	125267
40000	125034	125047	125061	125058	125071	125084	125081	125093	125106
80000	125029	125035	125043	125052	125058	125066	125094	125101	125110
120000	125026	125031	125038	125052	125057	125065			
200000	125030	125036	125043						

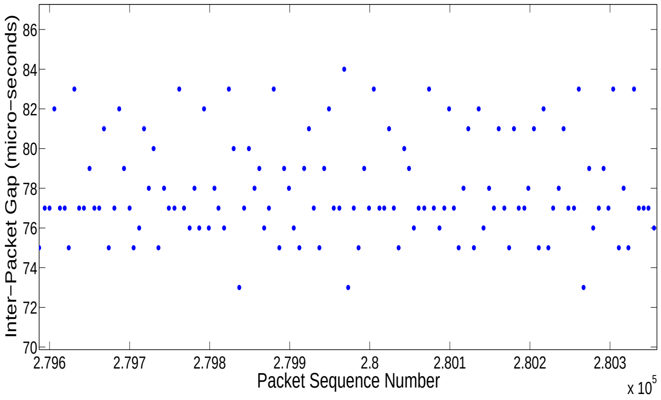

Fig. 5. Resulting inter-packet gaps after 155 Mbps rate limit via Click

experiments in this section, we removed the link shaping node, and connected the Click traffic source/sink to the router directly. Table XV and Table XVI present the results for inter-frame gaps and packet delays, respectively. The results indicate that the router does not add a significant amount of delay when compared to the calibration data. Also, there is very little variance in the inter-frame gaps for packets coming from the router, meaning that packet forwarding functions well with small delays.

Next, we enabled a rate limiter on the interface via the `rate-limit` command. We specified an output limit of 155 Mbps and set the burst size to the minimum allowed setting of 77500 bytes. We opted for the smallest burst size to ensure that inter-frame gaps have the smallest degree of variance. Table XVII and Table XVIII demonstrate the results for inter-frame gaps and packet delays, respectively.

As was stated in Section V-A2, the theoretical minimum inter-frame gaps corresponding to using the entire link capacity are 3.30, 41.28, and 78.34 for 64-, 800- and 1518-byte Ethernet frames, respectively. One observation from the data is that even the 90^{th} percentiles are below the target values. However, the mean values (not shown) for the inter-packet delays are close to the ideal values. Figure 6 demonstrates the reason for such statistics. Even though the mean inter-frame gaps are close to the ideal target, the analysis of the individual packets shows a high degree of burstiness by the shaping mechanism. The router drops a sequence of packets and then transmits a burst of queued packets. This behavior results in a very large gap between the first packet of a current burst and the last packet of a previous burst. Hence, this behavior explains the L-shaped lines seen on the graph.

Even though the router has achieved the desired link rate limits, it has induced a large degree of burstiness in the traffic. Such burstiness can be detrimental for studies of real-time protocols such as VoIP. Additionally, the router does

TABLE XIII
Click 125 ms 155 Mbps bridge: inter-frame gaps (μs) for 64-, 800- and 1518-byte Ethernet frames

	64 bytes			800 bytes			1518 bytes		
Rate	10^{th}	mean	90^{th}	10^{th}	mean	90^{th}	10^{th}	mean	90^{th}
8000	111	127	141	112	126	141	14	135	152
40000	20	25	30	38	42	46	72	80	86
80000	3	12	22	37	42	48	71	80	85
120000	2	8	16	36	41	48			
200000	2	3	10						

TABLE XIV
Click 125 ms 155 Mbps bridge: packet delays (μs) for 64-, 800- and 1518-byte Ethernet frames

	64 bytes			800 bytes			1518 bytes		
Rate	10^{th}	50^{th}	90^{th}	10^{th}	50^{th}	90^{th}	10^{th}	50^{th}	90^{th}
8000	125084	125156	125230	125142	125209	125280	125204	125272	125343
40000	125038	125051	125065	135574	135594	135615	145105	145138	145172
80000	125032	125038	125047	135577	135594	135612	145112	145143	145176
120000	125028	125033	125040	135579	135595	135612			
200000	125034	125041	125048						

TABLE XV
Cisco 2851 1 Gbps: inter-frame gaps (μs) for 64-, 800- and 1518-byte Ethernet frames

	64 bytes			800 bytes			1518 bytes		
Rate	10^{th}	50^{th}	90^{th}	10^{th}	50^{th}	90^{th}	10^{th}	50^{th}	90^{th}
8000	123	125	127	123	125	127	123	125	127
40000	21	25	29	23	25	27	18	25	31
80000	5	12	18	8	12	17	11	13	14
120000	4	9	12	6	7	12			
200000	2	4	11						

TABLE XVI
Cisco 2851 1 Gbps: packet delays (μs) for 64-, 800- and 1518-byte Ethernet frames

	64 bytes			800 bytes			1518 bytes		
Rate	10^{th}	50^{th}	90^{th}	10^{th}	50^{th}	90^{th}	10^{th}	50^{th}	90^{th}
8000	16	17	18	33	33	35	48	49	50
40000	16	17	21	33	33	34	49	50	53
80000	15	17	21	33	35	39	56	60	348
120000	14	16	19	35	38	43			
200000	15	19	23						

TABLE XVII
Cisco 2851 155 Mbps rate limit: inter-frame gaps (μs) for 64-, 800- and 1518-byte Ethernet frames

	64 bytes			800 bytes			1518 bytes		
Rate	10^{th}	50^{th}	90^{th}	10^{th}	50^{th}	90^{th}	10^{th}	50^{th}	90^{th}
8000	123	125	127	122	125	128	122	125	128
40000	20	25	30	22	25	28	20	25	31
80000	6	12	19	6	14	16	11	13	14
120000	4	7	14	6	7	14			
200000	4	6	8						

TABLE XVIII
Cisco 2851 155 Mbps rate limit: packet delays (μs) for 64-, 800- and 1518-byte Ethernet frames

	64 bytes			800 bytes			1518 bytes		
Rate	10^{th}	50^{th}	90^{th}	10^{th}	50^{th}	90^{th}	10^{th}	50^{th}	90^{th}
8000	19	20	21	35	36	37	51	52	53
40000	19	20	24	35	36	38	53	54	57
80000	18	20	25	34	38	41	58	63	68
120000	17	21	25	37	40	43			
200000	418	434	471						

not provide a mechanism to delay the packets, thus requiring additional delay hardware regardless.

C. Ethernet Pause Frames

The final shaping method available to us is the IEEE 802.3x pause frame approach. As was discussed in Section III-B3, pause frames can be used to induce delays and limit the traffic rate. Because of how pause frames operate, there is a limit as to how much packet delay they can induce. On a 1 Gbps link, the maximum achievable delay is 33.553 ms[3]. Since pause frames are sent periodically, there can be cases when packets do not experience the desired delay. Such situations arise when packets are sent to the router between two successive pause frames (*i.e.*, when the link is not paused). Additionally, using pause frames induces heavy jitter. This result is due to the fact that the link operates in the on/off state, hence making the inter-frame gaps non-constant for a CBR flow. When conducting this experiment, we configured the Click

[3] $\frac{65535 \times 512\ b}{1e9\ bps} \times \frac{1000\ ms}{1 sec} = 33,553\ ms.$

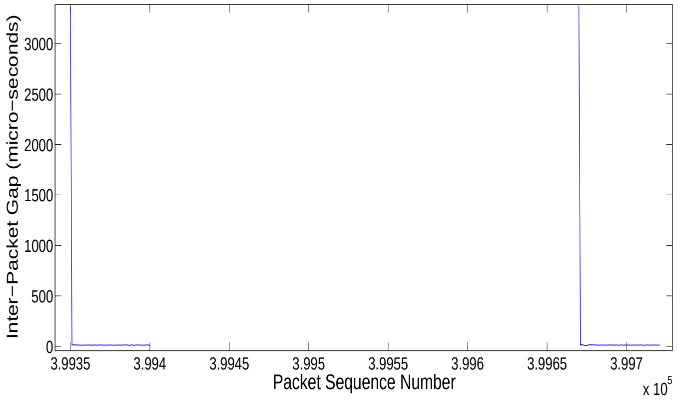

Fig. 6. Resulting inter-frame gaps after 155 Mbps rate limit via `rate-limit` on Cisco 2851.

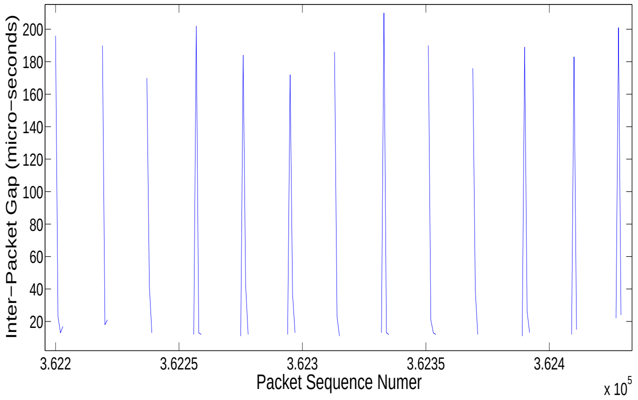

Fig. 7. Resulting inter-frame gaps after 155 Mbps rate limit via pause frames

modular router to generate pause frames with a quanta delay value of 390. This value is equivalent to a 200 μs delay on a 1 Gbps link. We also set the pause frame rate to 4231 packets per second. This rate is necessary to pause a 1 Gbps link to produce a 155 Mbps bandwidth limit. Since we had to use a hidden node to generate pause frames, we configured it to perform a 125 ms packet delay as the pause frames induced a delay of at most 200 μs.

The results for the inter-frame gaps and packet delays are shown in Table XIX and Table XX. As expected, the inter-frame gaps exhibit a very large degree of variance, and the 50^{th} percentiles are much smaller than the ideal inter-frame gaps. Also, the average inter-packet gap values are smaller than the ideal (not shown), meaning that the desired rate limit was not achieved. The delay values, on the other hand, are satisfactory.

Figure 7 provides additional insight about the inter-frame gaps. The figure shows a series of 1518-byte packets with the source rate of 80 Kpps. As expected, the inter-frame gaps vary from a low value, being 20 in this case, to 200 μs (pause frame delay). Even though the data exhibits a large degree of jitter, the pause frame method produces less jitter than the router rate limit approach.

Ultimately, the pause frame method did not provide us with the desired results, as the bandwidth reduction did not meet our specified target. The complexity of the pause frame method rivals the complexity of the delay node method, as in both cases, an intermediate node is required. A drawback of the pause frame method is the fact that the router has to constantly process pause frames, thus potentially taking away resources from regular data traffic processing. The applicability of this method only makes sense if the router can correctly process pause frames to meet the desired bandwidth limit, and if the experiment requires the packets to be dropped on the router instead of at the delay node.

D. The Best Shaping Method

Finally, we present the results of our original goal to shape a Cisco 2851 1 Gbps Ethernet link to provide a maximum bandwidth of 155 Mbps and a propagation delay of 125 ms, with minimal emulation artifacts. For this experiment, we chose to use the Click transparent delay node, as it proved to be superior over the others approaches. Figure 2 shows the topology used for this set of experiments. The Click shaper was configured to add 125 ms of delay and reduce the link bandwidth to 155 Mbps. The Cisco router was configured not to perform any rate limiting. Additionally, the Click delay node was configured to use a 256-slot based output queue. We chose a higher value than reported by the router's "`show interface`" command in order to include the effect of intermediate buffering [3].

Tables XXI and Table XXII demonstrate the results for inter-frame gaps and packet delays, respectively. The data indicates that the desired inter-frame gaps are achieved, and the difference between the 10^{th} and 90^{th} percentiles is small. Additionally, the packet delays are 125 ms except for cases when the queuing delay due to bandwidth shaping results in additional delays.

Figure 8 shows the inter-frame gaps for a flow of 1518-byte packets with a source rate of 80 Kpps. Even though there are some packets that have a 140+ μs inter-frame gap, the majority of the packets are around the desired 78 μs line. This indicates that the delay node was successfully used in conjunction with a Cisco 2851 router to emulate a link delay and rate limit the bandwidth. Unlike the previous methods, this method did not produce undesired artifacts and kept jitter to a minimum. The results indicate that if emulation fidelity is a priority, then the extra complexity due to the kernel-level Click transparent delay node is justified.

VI. CONCLUSION

In this paper, we focused on three link shaping methods: (1) transparent delay node, (2) rate limiting at the transmitting node, and (3) flow control via Ethernet pause frames. The focus of our study was to determine which method produced the desired delay and bandwidth. In addition, the study also took into consideration variance of inter-frame gaps (jitter) as well as packet losses.

The results of this work are pertinent to network experiments that deal with congestion control, transport protocols, and real-time traffic. Typically, such studies rely on link

TABLE XIX
Cisco 2851 125 ms 155 Mbps pause frames: inter-frame gaps (μs) for 64-, 800- and 1518-byte Ethernet frames

	64 bytes			800 bytes			1518 bytes		
Rate	10^{th}	50^{th}	90^{th}	10^{th}	50^{th}	90^{th}	10^{th}	50^{th}	90^{th}
8000	2	174	238	6	171	227	13	163	220
40000	1	2	162	6	8	186	12	21	194
80000	2	2	18	6	8	186	12	21	194
120000	2	2	22	6	8	186			
200000	2	2	20						

TABLE XX
Cisco 2851 125 ms 155 Mbps pause frames: packet delays (μs) for 64-, 800- and 1518-byte Ethernet frames

	64 bytes			800 bytes			1518 bytes		
Rate	10^{th}	50^{th}	90^{th}	10^{th}	50^{th}	90^{th}	10^{th}	50^{th}	90^{th}
8000	125120	125235	125355	125167	125276	125390	125209	125312	125421
40000	125104	125187	125269	131439	131596	131670	135971	136179	136392
80000	125156	125224	125283	131457	131647	131686	135992	136210	136434
120000	125210	125255	125298	132119	132317	132369			
200000	129110	129203	129314						

TABLE XXI
Click-Cisco 2851 155 Mbps rate limit: inter-frame gaps (μs) for 64-, 800- and 1518-byte Ethernet frames

	64 bytes			800 bytes			1518 bytes		
Rate	10^{th}	50^{th}	90^{th}	10^{th}	50^{th}	90^{th}	10^{th}	50^{th}	90^{th}
8000	111	127	141	111	127	142	14	138	151
40000	20	24	32	38	42	46	72	79	86
80000	2	12	22	36	42	48	72	79	90
120000	2	7	18	36	42	49			
200000	2	3	11						

TABLE XXII
Click-Cisco 2851 155 Mbps rate limit: packet delays (μs) for 64-, 800- and 1518-byte Ethernet frames

	64 bytes			800 bytes			1518 bytes		
Rate	10^{th}	50^{th}	90^{th}	10^{th}	50^{th}	90^{th}	10^{th}	50^{th}	90^{th}
8000	125095	125165	125237	125162	125230	125302	125231	125299	125370
40000	125047	125060	125074	135591	135612	135633	145133	145165	145199
80000	125041	125047	125056	135596	135613	135632	145144	145180	145236
120000	125038	125044	125052	135601	135618	135637			
200000	125045	125052	125064						

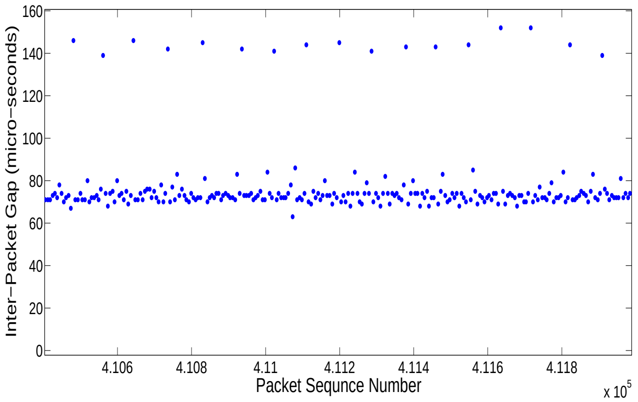

Fig. 8. Resulting inter-frame gaps after 155 Mbps, 125 ms link shaping via the Click modular router.

shaping to create a desired network topology for experimentation. If a link emulator introduces artifacts which affect the experimental results, this can lead to incorrect interpretations of the results and ultimately wrong conclusions.

To obtain the results, we created a variety of constant rate UDP flows and compared the performance of the link shaping methods with each other. The results revealed that the delay bridge using the kernel-level Click modular router is superior to the other two methods.

REFERENCES

[1] S. Agarwal, J. Sommers, and P. Barford. Scalable network path emulation. In *Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS)*, September 2005.

[2] R. Chertov, S. Fahmy, and N. B. Shroff. A black-box router profiler. In *Proc. of Global Internet*, May 2007.

[3] R. Chertov, S. Fahmy, and N. B. Shroff. A device-independent router model. In *Proc. of IEEE INFOCOM*, April 2008.

[4] R. Chertov, S. Fahmy, and N. B. Shroff. Fidelity of network simulation and emulation: A case study of tcp-targeted denial of service attacks. *ACM Transactions on Modeling and Computer Simulation (TOMACS)*, 19(1), December 2008.

[5] Cisco Systems. Commands: rate-limit – set qos-group. http://www.cisco.com/en/US/docs/ios/12_2/qos/command/reference/qrfcmd8.html, 2008.

[6] B. Hubert, et al. Linux Advanced Routing & Traffic Control HOWTO. http://www.tldp.org/HOWTO/Adv-Routing-HOWTO/lartc.qdisc.classful.html, 2002.

[7] E. Kohler, R. Morris, B. Chen, J. Jannotti, and M. F. Kaashoek. The Click modular router. *ACM Transactions on Computer Systems*, 18(3):263–297, August 2000.

[8] NIST Internetworking Technology Group. Nist net home page. http://snad.ncsl.nist.gov/nistnet/, 2008.

[9] L. Rizzo. Dummynet: A simple approach to the evaluation of network protocols. *ACM Communications Review (CCR)*, 27(1), January 1997.

[10] B. White, J. Lepreau, L. Stoller, R. Ricci, S. Guruprasad, M. Newbold, M. Hibler, C. Barb, and A. Joglekar. An integrated experimental environment for distributed systems and networks. In *Proc. of Usenix Operating System Design and Implementation (OSDI)*, December 2002.