End-to-End Traffic Shaping

Felipe Maya, Luiz Magalhães
Mestrado em Engenharia de Telecomunicações – Universidade Federal Fluminense (UFF)
{fmay, schara}@midiacom.uff.br

Abstract

This work is an investigation on how end-to-end traffic shaping affects communication. While studying rate-based congestion control, we developed a linux module capable of measuring available bandwidth between two end points, and of pacing packets according to the available bandwidth. The module can be used for congestion control of non regulated protocols, such as UDP. This paper describes the module and preliminary results.

1 Introduction

The transmission of data in networks where bandwidth is dynamically allocated, e.g. the Internet, can result in an unstable behavior when there is significant traffic, leading to overflowing queues on routers and to packet loss. The conventional FIFO policy on routers will discard packets that arrive last when buffers are full. On the other hand, if it were possible to obtain information on measurements such as bandwidth, link capacity and network usage, it would be possible to use the Internet in a more efficient manner, by dynamically controlling flows at the source. This would be a form of congestion avoidance, where packets violating the measured bandwidth would be dropped at the source, not wasting network bandwidth needlessly on the path to the choke point.

This work describes a Linux module that measures available bandwidth in a path, and sends packets at regular intervals, with the constraint of timer granularity, to match the available end-to-end bandwidth, joining the concepts of congestion control and pacing in a traffic shaping module.

2 Flow Control Mechanisms

The proposed mechanism for flow control (or traffic shaping) is composed of two separate phases. The first is the active probing, where probe packets are sent in the network to gather delay measurements, and the second, the rate change phase, is charged with producing the estimates to control data transmission.

2.1 Active Probing

The calculation of the available bandwidth on the traffic shaping module is based on a series of techniques that actively measure the number of bits that can be sent in an end-to-end path for a determined time period.

The technique used for active probing is the Train of Packet Pairs (ToPP), which is a friendly measurement to estimate the available bandwidth in a congested link. It sends a sequence of ICMP packet pairs of the type Timestamp and Timestamp Reply Message (according to RFC 792), types 13 and 14 [4], where the header carries the time when the packet left the source, the time when the packet arrived at the destination and the time the packet left the destination. Each timestamp corresponds to the time in milliseconds since midnight, used as the epoch. Besides these fields, information using microsecond accuracy is added by our module if present, because it is needed for fine grained measurements (for use with high speed links).

The ToPP sequence used in this traffic shaping module is composed of a series of packet pairs with different time spacing, as shown in (figure 1)

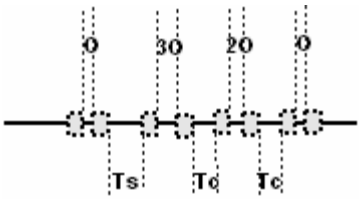

Figure 1. ToPP Sequence

The main difference from standard ToPP it that the traffic shaping module varies the rate O for each pair in the sequence. Therefore, it is possible to calculate estimates for each sequence and not wait until n sequences are received to make the estimate. The new ToPP sequence depends on the sum of the RTTs Ts and each pair is sent at each Tc.

2.2 Variation on the Transmission Rate

When the timestamps from the ToPP sequence are received, the transmission rate is estimated based on calculations of error on the delay [2]. When the rate is estimated, the module identifies which queue is associated with that end-to-end path, and uses this as the pacing rate for that queue. The traffic shaping uses a dequeue function, which examines each queue in turn, looking for packets that should be sent at the current time, as shown in (figure 2).

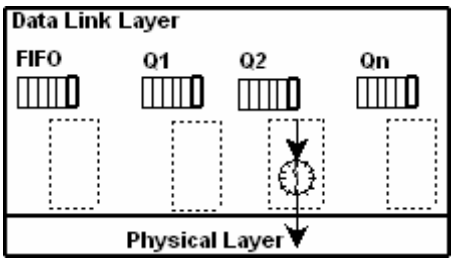

Figure 2. Dequeueing Mechanims

The first queue uses a conventional FIFO, and packets from that queue will be chosen whenever a packet is available. The inspection continues on each queue, selecting packet which are ready to be sent.

There is a dependency between the timer granularity of the system and the rate that packets will be dequeued. The queues are examined at each clock tick, so all rates will be divisors of the basic rate, which is to send one packet every clock tick.

For each queue, the maximum bandwidth reachable can vary dynamically depending on the estimates of the active probes, and will follow equation (1), where k is the number of interrupts estimated to give the required rate, HZ is the interrupt frequency and P the size of the packet:

$$MaxRate = P^* (k * HZ) \quad (1)$$

When k equals 1, the system is working at its maximum rate (which may be below the actual rate in the network), and sends a packet at each interrupt.

3 Experimental Results

Our experiments were carried out in the network shown on figure 3. It consists of two end-systems connected to a 100 Mbps Ethernet network and two routers with a band-limited 28Kbps link (burst 4750 bytes, latency 50 ms). The end systems run Linux with kernel 2.6.17 compiled with the variable HZ set to HZ_100, which results in a clock tick of 1ms (timer interrupts occur at 1ms, which gives the minimum period between two events).

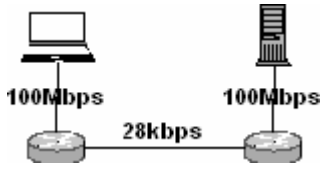

Figure 3. Test Network.

It should be pointed out that the timer granularity is extremely important in these experiments and to rate-based mechanisms in general. The timer granularity sets the minimum interval between the transmission of two packets, and also the maximum rate attainable by a protocol, or in this case, a traffic shaper, that can send only a packet at

each timer interrupt. Modern systems allow timer granularity of 1ms (down from 20 ms which was the default in earlier kernels), but for better performance hardware assistance may be necessary.

To test our mechanism the client/server application TFTP (Trivial File Transfer Protocol), which uses UDP and no congestion control, was used. We show the results for four runs, with and without the traffic shaping module.

Without the module:

```
Sent 1289616 bytes in 417.2
seconds [24730 bit/s]
Sent 1289616 bytes in 420.6
seconds [24528 bit/s]
Sent 1289616 bytes in 415.0
seconds [24858 bit/s]
Sent 1289616 bytes in 427.8
seconds [24116 bit/s]
```

With the module:

```
Sent 1289616 bytes in 402.3
seconds [25648 bit/s]
Sent 1289616 bytes in 404.5
seconds [25504 bit/s]
Sent 1289616 bytes in 400.1
seconds [25785 bit/s]
Sent 1289616 bytes in 406.2
seconds [25398 bit/s]
```

The first set has an average of 24,6Kbps, and the second 25,6Kbps. It is a 4% increase in throughput, even though probe packets are being sent to measure available bandwidth.

4 Conclusions and Future Work

While it is in its early stages, this investigation shows promise, specially in using traffic shaping techniques at the end-point to work as a congestion control mechanism for applications that may be badly behaved and cause congestion, specially those that use UDP.

We showed that TFTP transfers can actually benefit from the module, which may be explained by smaller losses at the routers, which in the long run make the process run faster, though it is bandwidth limited on the origin.

The module works with any protocol, and measures the available bandwidth for all active destinations, so it can also be used in conjunction with TCP. We plan to study the interactions between TCP's congestion control and the pacing algorithm, and see if improvement can also be realized for TCP. It may be that improvements will only be seen on the network, and performance at the end-points may be worse. Both congestion avoidance and TCP pacing showed promise, but did not behave well in an uncontrolled environment.

The need for better timers may also be a hurdle to be overtaken. It is expected that timer granularity will be finer as processor clock speeds keep on improving, and the current tendency of making smarter network cards [8], offloading network protocol processing from the main CPU to the NIC can also help this effort.

5 References

[1] Vikram Visweswaraiah, John Heidemann. Rate Based Pacing for TCP. 1997. http://www.isi.edu/lsam/publications/rate_based_pacing/

[2] Luiz C. Schara Magalhães. PhD thesis: A Transport Layer Approach to Host Mobility., University of Illinois at Urbana-Champaign, 2005

[3] E. Kohler, M. Handley, and S. Floyd. Designing DCCP: Congestion control without reliability. Available at http://www.icir.org/kohler/dccp/, May 2003.

[4] Prasad, R.S., Murray M., Dovrolis, C., Clay K.: Bandwidth estimation: metrics, measurement techniques, and tools. IEEE Network Magazine, Volume: 17, Issue: 6 pages: 27- 35, 2003

[5] Hu, N., Steenkiste: Evaluation and Characterization of Available Bandwidth Probing Techniques. IEEE Journal on Selected Areas in Communication 21 Volume:21, Issue:6 pages: 879- 894, 2003

[6] Strauss, J., Katabi, D., Kaashoek, F.: A Measurement Study of Available Bandwidth Estimation Tools. In: ACM SIGCOMM Internet Measurement Workshop. Pages: 39 - 44 , 2003

[7] Andreas Johnsson. On the Comparison of Packet Pair and Packet Train Measurements. http://winternet.sics.se/workshops/sncnw2003/proceedings/25T-posterSNCNW03.pdf

[8] Killer NIC Overview - www.killernic.com/KillerNic/KillerOverview.aspx