Editor: Harrick Vin
University of Texas at Austin

Towards a Better-Than-Best-Effort Forwarding Service for Multimedia Flows

Kevin Jeffay
University of North Carolina at Chapel Hill

My colleagues and I at the University of North Carolina at Chapel Hill (UNC) conduct research about the management of multimedia and more general continuous-media flows across the Internet. We focus on network support for (highly) interactive applications where real-time transmission is essential for the correct operation of the application, such as tele-immersion and interactive distributed virtual environments. To be effective, these applications typically require the ability to transmit data continuously at an application-specific, uniform rate.

The requirement of continuous transmission puts many multimedia applications at odds with current and proposed Internet network management practices. Whereas other network applications such as Web browsers rely on the underlying network transport protocol, most notably TCP, for congestion control and avoidance, real-time interactive multimedia applications eschew TCP in favor of the User Datagram Protocol (UDP). The reliance on UDP by interactive multimedia applications typically results from a conscious trade-off between reliable, in-order data delivery and the ability to exercise control (however effective) over the reaction to packet loss and the variation in the delay of transmissions. Thus, there exists a fundamental tension between the current practice of relying on end-systems to limit the rate at which they inject traffic into the network and the desire of many multimedia applications to transmit data continuously at a uniform rate. TCP connections detect packet loss in the network, treat it as a signal of congestion, and react by reducing their transmission rate. While this particular reaction to packet loss is unsuitable for applications requiring uniform or quasi-uniform-rate transmissions, it is essential for the health of the Internet. It ensures a crude form of fair sharing of link capacity between TCP connections and ensures that a pathological condition called congestion collapse does not occur. These are vital attributes given that TCP traffic accounts for approximately 97 percent of all the bytes transmitted over the Internet today.

UDP applications, such as high-bandwidth multimedia applications, have the potential to starve TCP connections of bandwidth. During congestion, queues of packets build up in routers and switches. Packets are dropped ("lost") when the queues overflow. A UDP application that does not respond to packet loss as an indicator of congestion will continue transmitting data into queues that are overflowing. TCP connections that share these queues will react to packet loss and reduce their transmissions. If the buffer space in routers previously consumed by these TCP connections are now consumed by packets from nonresponsive applications, then the situation spirals. Router queues remain full and TCP connections, although transmitting at lower rates, still lose packets and therefore continue to reduce their transmission rates until they effectively stop transmitting.

Our research attempts to develop network resource-management mechanisms that strike a balance between the conflicting requirements of continuous transmission and congestion control. TCP connections require isolation or protection from nonreactive applications (or more generally, from all non-TCP applications), and interactive UDP-based multimedia applications require the ability to transmit data continuously with (ideally) low end-to-end latency. All of these goals can be met in a tunable manner by reserving bandwidth in the network for either individual connections or collections of connections, and explicitly allocating network bandwidth on a

Published in: IEEE Multimedia, Volume 6, Number 4,
October-December 1999, pages 84-88.

packet-by-packet basis by scheduling packets across network links. However, this approach is complex and requires close coordination among, and integration between, all routers (and hence all network service providers) along the path from sender to receiver.

We are investigating an alternate approach based on queue management rather than packet scheduling. The practice of manipulating queues to bias the performance of network connections is known as active queue management. We are developing lightweight active queue management schemes for routers that (1) protect TCP connections from starvation by "aggressive" or nonresponsive UDP applications, and (2) reserve router capacity for UDP applications that require continuous transmission. The goal is not to provide guarantees of performance, but rather to provide a so-called "better-than-best-effort" forwarding service for multimedia flows that results in better throughput and latency for multimedia flows in times of congestion. The challenge is to provide a better-than-best-effort service without unnecessarily degrading TCP performance. We focus on a fundamentally best-effort service because we recognize that multimedia applications do not, in general, require or need performance guarantees. Most applications are capable of adapting to modest changes in network conditions but have minimal bandwidth and latency requirements that must be respected. Moreover, we are interested in determining how close one can come to achieving the performance of packet scheduling schemes with simpler and cheaper mechanisms.

Active queue management

The default "best-effort" packet-forwarding service of the Internet Protocol (IP) is typically implemented in routers with a single, fixed-size, first-in, first-out (FIFO) queue shared by all flows. The queue exists simply to provide capacity for tolerating variability ("bursts") in the arrival rate of packets. Router implementations using a simple fixed size FIFO queue typically drop any packet that arrives at an already full queue. We call this behavior drop-tail packet discarding. However, this behavior has two well-documented problems. First, in some situations flows can be "locked out," a condition in which a small subset of the flows sharing the outbound link monopolize the queue during periods of congestion. Flows generating packets at a high rate fill up the queue such that packets from flows generating packets at substantially lower rates have a higher probability of arriving at the queue when it is full and being discarded. This has the effect of providing feedback to the "wrong" sources. The locked out flows receive feedback that they should reduce their transmission rate while the "aggressive," high-packet-rate flows receive less feedback and do not reduce their rate appropriately.

The second problem is that latency is increased for all flows when the queue remains full or nearly full. Simply making the queue shorter will decrease the latency but negates the possibility of accommodating brief bursts of traffic without dropping packets. For TCP-responsive flows, these latency problems can be addressed by selectively dropping packets before the queue fills. Since TCP flows decrease the load they generate in response to drops, dropping packets "early" should have the effect of eventually slowing or ceasing the growth of router queues.

The Internet community is promoting a specific form of active queue management for routers known as random early detection, or RED. Under RED, packets will randomly be dropped from a non-full queue with the probability of a packet being dropped at any given time being a function of the average length of the queue in the recent past. RED avoids lockout by dropping packets uniformly from all flows. By providing early notification of congestion, RED has been shown to result in shorter average queue lengths in routers.

Although RED improves the network performance of TCP flows, they are still susceptible to starvation in the presence of nonresponsive UDP flows. The starvation phenomenon has led to the call for routers to explicitly recognize nonresponsive flows and to place them in a "penalty box." Nonresponsive flows would be penalized by some draconian action such as having all their packets dropped during times of congestion. Such proposals are, on the one hand, seemingly necessary to protect TCP from nonresponsive flows and to provide an incentive for applications to respond to packet loss as an indicator of congestion. On the other hand, the penalty box concept does not bode well for continuous media applications that, while potentially adaptive, do not or can not adapt in a manner that mimics TCP's response to congestion.

Class-based thresholds

We are experimenting with extensions to RED and other active queue management schemes that will isolate TCP flows from non-TCP flows while at the same time ensuring that continuous media applications receive a tunable share of network

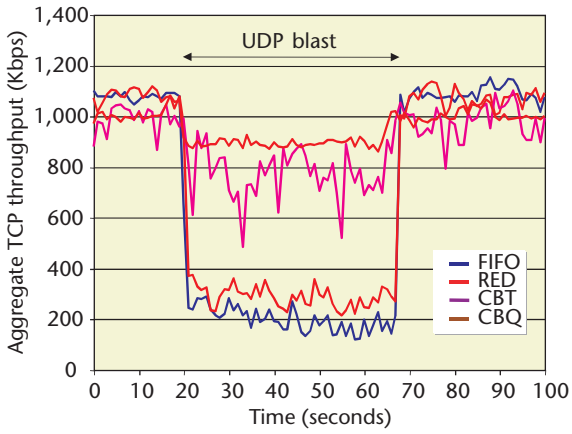

Figure 1. TCP throughput under each queue management/packet scheduling scheme.

resources. Our approach is to isolate TCP flows from the effects of all other flows by constraining the average number of non-TCP packets that may reside simultaneously in a router's queue. We also want to isolate classes of non-TCP traffic from one another, specifically isolating certain continuous media traffic from all other traffic. To do this we tag packets of designated continuous media streams before they reach the router so that they can be classified appropriately. How this tagging is accomplished and the issues raised by tagging flows lie outside the scope of this work. However, our operating assumption is that these flows are either self-identified at the end system or identified by network administrators as part of a larger architecture for differentiated network services.

Once flows are tagged, a router will maintain simple queue statistics for a limited number of classes of traffic. For simplicity, in the following we assume three traffic classes: TCP, marked non-TCP, and all others. The throughput of traffic classes will be constrained during times of congestion by limiting the average number of packets each traffic class can have enqueued in the router. When a packet arrives at the router, it is classified into one of the three traffic classes. TCP packets are subject to the RED algorithm. For the non-TCP traffic classes, a separate weighted average number of enqueued packets from each class is maintained. When a non-TCP packet arrives, the weighted average for the appropriate class is updated and compared against a maximum threshold for the class. If the class average exceeds the threshold, the incoming packet is dropped, otherwise it is enqueued. Although packets are classified, there is still only one queue of packets, shared by all traffic classes, per outbound link in the router, and all packets are enqueued and dequeued in a FIFO manner.

We call our scheme class-based thresholds (CBT). CBT provides isolation between traffic classes by maintaining separate thresholds for each class. These thresholds effectively allocate each class a portion of the queue's capacity and ensure this capacity is available to the class independent of the transmission rates of other classes.

To demonstrate the effectiveness of CBT, we empirically compare the performance of CBT to simple FIFO (drop-tail) queuing and RED. We would naturally expect FIFO queuing to result in the worst performance for all flows. The best possible performance results when packets are explicitly scheduled. Therefore, we also compare CBT to a packet scheduling scheme called class-based queuing (CBQ). In CBQ, link bandwidth (as opposed to queue elements) is explicitly allocated to traffic classes, and classes are segregated into separate queues and scheduled according to their bandwidth allocations. Since traffic classes no longer share a common queue, all traffic classes are completely isolated from one another, and, if provisioned correctly, flows such as continuous media flows should experience no drops. While a packet scheduler such as CBQ can in fact provide performance guarantees to traffic classes, it is considerably more complicated to implement than an active queue management scheme for a FIFO queue. The interesting question is how closely a simpler active queue management scheme such as CBT can approximate the performance of CBQ.

Figure 1 illustrates aggregate TCP performance under FIFO, RED, CBT, and CBQ. In this demonstration a router is configured with a 10-Mbps outbound link that is the bottleneck link in the network. Five separate experiments are run, each using a different active queue management (or packet scheduling) scheme. In the CBT and CBQ implementations there are three classes of traffic: TCP, marked non-TCP, and other traffic. Initially, a group of approximately 3,000 users browsing the Web share the 10 Mbps link. A sufficient number of browsers are active at any one time to keep the average link utilization at approximately 95 percent. In addition, there exists a set of six marked, non-TCP audio/video flows originating from end systems participating in a set of videoconferences (each generating 220 Kbps of UDP traffic).

At time 20, a single high-bandwidth UDP application (the "UDP blast") commences transmission. Instead of sharing link capacity fairly with the TCP

connections, the UDP application causes congestion and starves the TCP connections of bandwidth. The introduction of the unresponsive application causes the network to become saturated and hence causes packet loss. The TCP connections react to the loss by reducing their transmission rates and hence consuming fewer buffers in the router. The UDP application also sees a high rate of packet loss but nonetheless does not adapt its transmissions. Under FIFO and RED, the buffers in the router remain full. All network connections continue to experience packet loss, and TCP performance plummets as connections are starved of bandwidth. CBT and CBQ do show a decrease in TCP throughput, but the decrease is on the order of 10 percent for packet scheduling and 20 percent for CBT. This is because both methods constrain the amount of bandwidth that the unresponsive flow can consume.

In addition to protecting TCP, CBT also isolates the marked non-TCP multimedia flows from the nonresponsive flows in the "other" class. This can be seen from the measured drop-rates for marked multimedia packets during the period of the UDP blast shown in Table 1. Both FIFO and RED produced extremely high drop-rates for multimedia while CBT produced drop-rates of just over 1 percent. No multimedia packets are lost under CBQ, as CBQ is providing a guaranteed forwarding service to these applications. However, CBT closely approximates the performance of CBQ. In particular, it results in a drop rate for marked multimedia flows that is quite likely to be tolerated by the applications.

Table 1 also compares the average end-to-end network latency of marked multimedia packets under each scheme. Latency is highest under simple FIFO queuing because the queue is consistently full. RED and CBT maintain shorter average queues, and this is reflected in their lower average latencies. CBQ provides the best average latency, as it maintains a separate queue for packets from each traffic class.

Table 1. Comparison of drop rates and network latency for tagged continuous media traffic under the various queue management alternatives.

Queue Management Scheme	Packet Drop-Rate for Marked Flows	Average Latency of Marked Flows
FIFO	32.4%	63.2 ms
RED	30.0%	26.2 ms
CBT	1.3%	28.4 ms
Packet Scheduling (CBQ)	0.0%	5.7 ms

Summary

A salient requirement of interactive multimedia applications is that they transmit data continuously at uniform rates with minimum possible end-to-end delay. The majority of these applications do not require hard and fast guarantees of network performance, but, the current best-effort forwarding model of the Internet is frequently insufficient for realizing these requirements. Worse still, the requirement of uniform-rate transmission puts many multimedia applications at odds with current and proposed Internet network management practices that assume or require TCP-like reactions to packet loss. The Internet only works as well as it does because the vast majority of packets transmitted on the network are controlled by (loosely speaking) the same congestion control and avoidance algorithm. Multimedia flows present a potential threat to this regime given their potentially high bandwidth requirements (either individually or in aggregate). Moreover, making these flows react to congestion in a TCP-like manner is not likely to work for all applications.

We are investigating a form of segregation between TCP and non-TCP continuous media flows. This will allow the research community to develop congestion control and avoidance mechanisms for continuous media applications inherently suited to the performance requirements of these applications without being burdened by compatibility with TCP. We are investigating router-based active queue management, specifically, the use of queue occupancy thresholds to isolate TCP flows and to provide a better-than-best-effort forwarding service for flows in need of uniform-rate transmissions. Our current scheme, class-based thresholds, relies on a packet marking mechanism such as those proposed for realizing differentiated services on the Internet. CBT, when combined with existing active router queue management schemes such as RED, provides performance for TCP that approximates that achievable under a packet scheduling scheme and acceptable performance for multimedia flows. CBT is a simple and efficient mechanism with implementation complexity and run-time overhead comparable to that of RED. **MM**

Acknowledgements

The research described here has been performed

by the Distributed and Real-Time Systems group at the University of North Carolina at Chapel Hill (see http://www.cs.unc.edu/Research/dirt). Mark Parris is the inventor of the CBT concept and performed the experiments described in the section "Class-based thresholds." Other team members and contributors include Don Smith, David Ott, Michele Clark, Jan Borgersen, Arun Moorthy, Mikkel Christiansen, Ramkumar Parameswaran, Gerardo Lamastra, Felix Hernandez, and Karim Mardini.

Support for this research has come from the National Science Foundation (grants CDA-9624662, CCR 95-10156, and IRIS-9508514), the National Institute of Health (National Center for Research Resources Award RR02170), the IBM Corporation, the Intel Corporation, Cabletron Incorporated, Advanced Networks & Services Inc., and the North Carolina Networking Initiative.

Readers may contact Jeffay at the Department of Computer Science, University of North Carolina at Chapel Hill, Chapel Hill, NC 27599-3175, e-mail jeffay@cs.unc.edu.