A Differentiated Services Implementation for High-Performance TCP Flows

Volker Sander*† Ian Foster*‡ Alain Roy‡ Linda Winkler*

Abstract

The IETF's recent differentiated services architecture, which specifies a scalable mechanism for treating packets differently, offers new opportunities for building end-to-end quality of service (QoS) systems. However, it also introduces new challenges. In particular, it is not clear whether TCP's flow and congestion control mechanisms work well with the mechanisms used for end-to-end QoS. For that reason it is essential to analyze whether the existing differentiated services mechanisms can be used with standard TCP implementations or whether it is necessary to wait for upcoming features introduced in future modified versions of TCP. The Globus Architecture for Reservation and Allocation (GARA) supports flow-specific QoS specification, immediate and advance reservation, and online monitoring and control of both individual resources and heterogeneous resource ensembles. Using GARA, we evaluated actual differentiated services mechanisms provided by Cisco routers. We present the results of this evaluation and discuss their impact on the performance of popular TCP implementations.

Keywords: Differentiated Services, Quality of Service, Transmission Control Protocol

*Mathematics and Computer Science Division, Argonne National Laboratory, Argonne, IL 60439, U.S.A.
†Central Institute for Applied Mathematics, Forschungszentrum Juelich GmbH, 52425 Juelich, Germany
‡Department of Computer Science, The University of Chicago, Chicago, IL 60637, U.S.A.

1 Introduction

Network quality of service (QoS) can be addressed by a variety of mechanisms. However, the actual deployment of those mechanisms in the current structure of the Internet is rare. Network layer mechanisms such as ATM assume a homogeneous infrastructure between two end-systems, which for economic reasons may not exist. The Integrated Services (IS) Architecture [3] does not depend on homogeneous infrastructure but often requires that the end-systems have a specialized kernel installed. Furthermore, it requires a per-flow handling on every networking device, which results in scalability problems. Recently the IETF has defined a new QoS framework that addresses these concerns: the Differentiated Services (DS) [2] architecture.

The DS architecture defines an architecture for implementing scalable service differentiation in the existing Internet. Edge routers are configured to mark packets for a desired class. Interior routers treat packets based on their aggregates (classes), an approach that eliminates the scalability problem introduced by the IS model. The DS model defines different ways [13, 11] in which a marked packet should be treated by an interior router, based on the associated class. This specification is called per-hop behavior (PHB). In summary, an implementation of the DS architecture has to provide edge routers that are capable of marking packets per flow and interior routers that implement the related PHBs.

Several recent studies [5, 22, 23] show that it is difficult to guarantee requested throughput for TCP flows, because of the flow and congestion control [20, 1] mechanisms used by by TCP, which result, for example, in bursty traffic. Therefore, researchers have proposed both changes to the DS architecture and modifications to the TCP protocol (e.g., two-window TCP [5]). However, no study has analyzed high-speed flows (10 MB/s or more) when using commodity router hardware that provides buffer capabilities of up to several megabytes.

Using Cisco's implementation of DS mechanisms, we performed a careful evaluation of high-bandwidth TCP performance and used our results to draw conclusions about how to configure DS to provide a premium service in this situation. This work was performed in the context of the Globus Architecture for Reservation and Allocation (GARA) [8], which builds on these mechanisms to deliver per-flow, advance reservation, end-to-end quality of service.

2 The Differentiated Services Architecture

2.1 Overview

The Differentiated Services architecture [2] is a reaction to earlier per-flow-based network quality of service architectures such as RSVP. In contrast to these QoS architectures, DS focuses on defining the behavior of aggregates. Packets are identified by simple markings that indicate which aggregate behavior they should be given. In the core of the network, routers need not determine which flow a packet is part of, only which aggregate behavior should be used.

DS pushes complexity to the edges of the network: packets are marked either by applications or by edge routers. If edge routers mark packets, they may choose to do so on a per-flow basis or on any other criteria. Packets may be marked only when they are "within profile"—that is, when the sender is sending within predetermined limits such as bandwidth or time

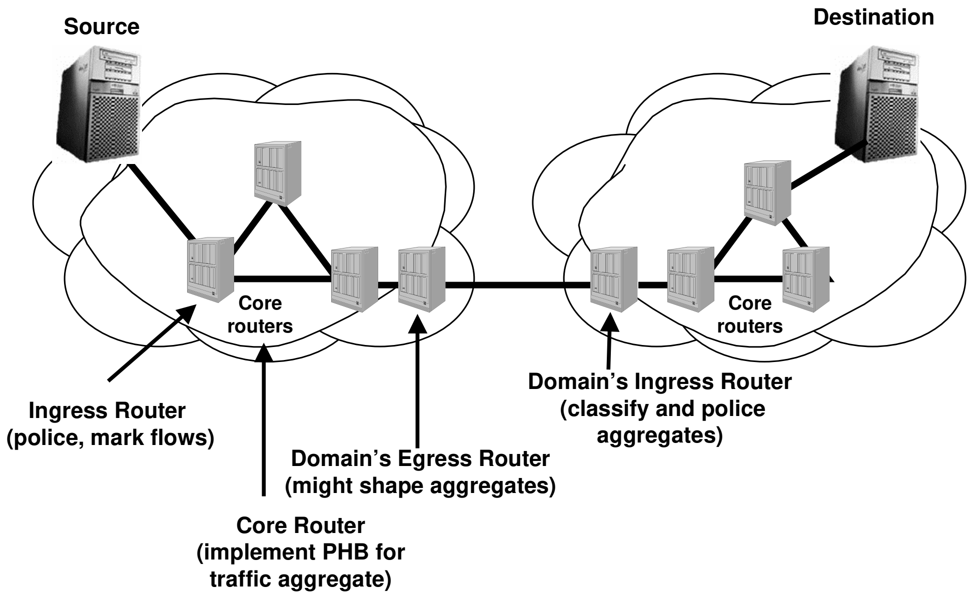

Figure 1: A network flow going through two DS domains. The ingress router is responsible for policing and marking of packets. The interior, or core, router handles the traffic based on the aggregate's PHB. The domain's egress router might enforce the shape of the traffic of the aggregates. The domain's ingress router does policing based on aggregates.

of day. In contrast to this complexity in the edge routers, routers in the core of the network provide service based only on these markings. A particular marking on a packet indicates a *per-hop behavior* (PHB). Currently, the IETF's Differentiated Services Working Group has specified multiple PHBs [13, 11]. In this paper, we consider the Expedited Forwarding (EF) PHB.

Figure 1 illustrates the DS architecture in a two-domain environment.

2.2 Cisco's Implementation of Differentiated Services

We use a DS implementation based on Cisco 7500 series routers, which support the EF PHB, with two mechanisms:

- Committed Access Rate (CAR) is used on the ingress ports of edge routers to mark and police the flows for which premium bandwidth is required. It is also used on the ingress router of a domain to police the premium aggregate. CAR currently does not support specific DS packet markings (as defined in [17]) but instead uses the IP precedence field to mark packets belonging to EF flows.

- Weighted Fair Queuing (WFQ) is used on the egress port of edge routers and in interior routers. WFQ ensures that in periods of congestion—namely, when packets get queued in the router because the output link does not provide the capacity for delivering them immediately—each IP precedence class receives at least the fraction of the output bandwidth defined by the weight defined for that class. Cisco's VIP2-50 implementation allows premium classes to be assigned up to 99% of the available link bandwidth. It is important to note that WFQ has an active effect only when there is congestion. When the interface is not congested, queues can use any available bandwidth, regardless of their weight.

3 TCP Overview

Establishing end-to-end QoS for high-performance TCP flows in a DS environment is a challenge. This section gives an overview of the dynamics of TCP flows and discusses the nature of associated challenges.

3.1 Flow Control

TCP implements flow control mechanisms to prevent a sender from transmitting data faster than its receiver can handle. These mechanisms constrain the maximum amount of outstanding data (i.e., data sent but not acknowledged) to be less than or equal to a specified window size. The sliding window protocol [21, 4] used to implement flow control uses two windows:

- The receiver advertises the window that corresponds to the amount of free space in the receiving buffer, relative to the acknowledged data. The receiver guarantees that it is able to buffer the data indicated by the window. The maximum offered window size can be set by the application.

- Based on the offered window size and on other information (e.g., the available local buffer space, the first unacknowledged segment, and the outstanding segments), the sender computes the usable (i.e., the actual) window size.

3.2 Congestion Control

Wide-area tests have shown that if a sender immediately injects multiple segments into the network up to the window size advertised by the receiver, the application throughput, often called goodput, can be reduced [12] in the presence of competing traffic (congestion).

Therefore, TCP has been modified to avoid the injection of a full offered window at the beginning of a session. Instead, TCP starts with a small usable window size and then increases this window size based on the successful receipt of segments. Another TCP sender window is introduced, the congestion window (cwnd), which can reduce the usable window size; it is the minimum of the previous value and the cwnd. The operation of the sender is split into two phases:

- Within the slow-start phase the cwnd is increased with every acknowledged segment, until it reaches a system-wide maximum threshold or the transmitter detects that packets were dropped. Dropped packets are determined either by timing out or by receiving a specific amount of duplicate acknowledgments [20, 1]. Starting with a cwnd size of 1 or 2, the slow-start phase doubles the cwnd-size with every round trip of a full window.

- Within the congestion control phase the cwnd is increased only by one with every full-window round trip. Again this phase is limited by a system-wide threshold and by the detection of dropped packets.

The process is described more formally in [14].

To emphasize the effect of these mechanisms, we note that the goodput of a TCP application is limited to the actual window size divided by round trip time (bandwidth*delay product [21]). The impact of a single drop on the short-term TCP goodput is enormous. The macroscopic behavior of these mechanisms is described in [15, 14]. The general impact of the window size on the goodput is evaluated in [18, 16]. Statistical analysis for the impact on a DS implementation is discussed in [19, 23].

4 Experimental Studies

We report on experiments designed to examine a DS implementation based on commodity networking products.

4.1 Experimental Configuration

Our experimental configuration, illustrated in Figure 2, comprises a laboratory testbed at Argonne National Laboratory (the Globus Advance Reservation Network Testbed: GARNET) connected to a number of remote sites, including Lawrence Berkeley National Laboratory (LBNL). Connectivity to LBNL is provided by the Energy Sciences Network (ESnet) DS testbed. GARNET allows controlled experimentation with basic DS mechanisms; the wide-area extensions allow for more realistic operation, albeit with a small number of sites. Notice that end-system resources are located in different domains; hence, we must deal with distributed authentication and authorization.

Cisco Systems 7507 routers are used for all experiments. Within GARNET, these are connected by OC3 ATM connections; across wide-area links, they are connected by VCs of varying capacity. We are restricted to these relatively slow speeds because the 7507 cards do not implement CAR and WFQ at speeds faster than OC3. End-system computers are connected to routers by either switched Fast Ethernet or OC3 connections.

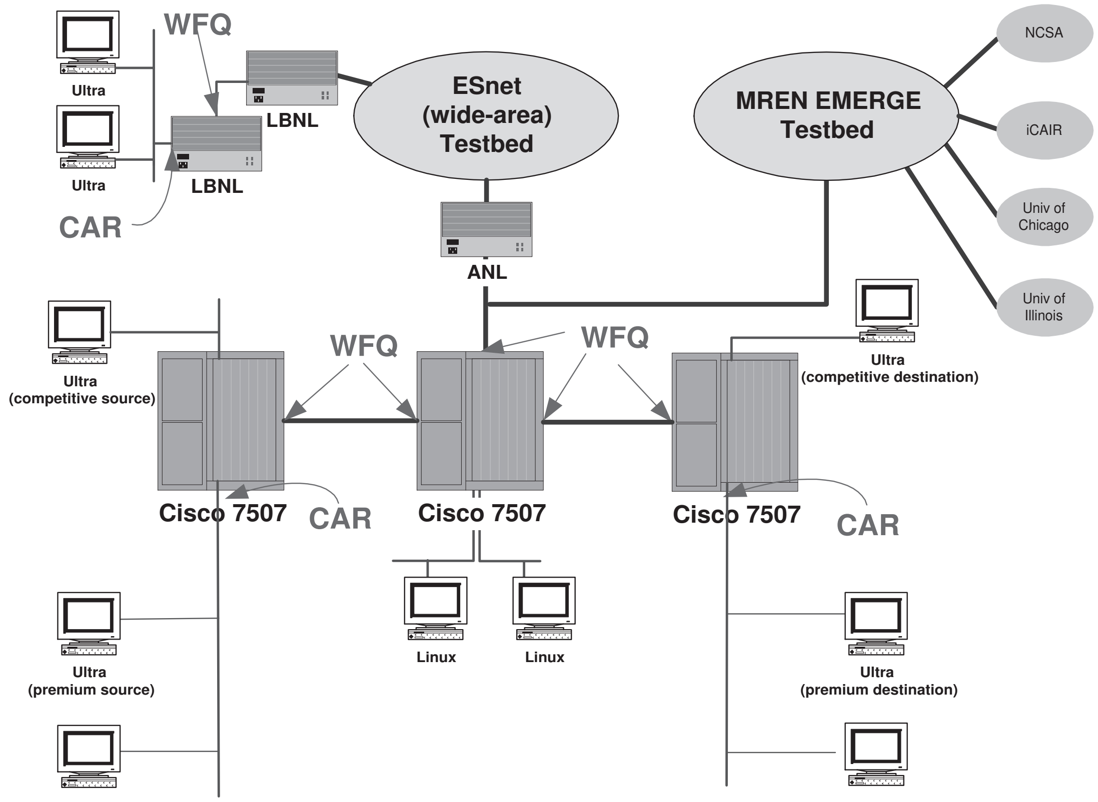

Figure 2: The experimental configuration used in this work, showing our local GARNET testbed and its extensions to remote sites connected via ESnet and the Metropolitan Research and Education Network (MREN).

4.2 Evaluation Tools

The analysis required appropriate measurement methods. It was necessary to use a tool that was capable of producing a network flow (TCP and/or UDP) with specified characteristics such as bandwidth or burstiness. Also needed was a tool for creating a well-defined amount of competing traffic, to create some congestion on the internal network.

- *TCP stream generator:* we use a TCP traffic generator capable of generating a flow with a predetermined rate. This generator operates by adapting the frequency of write() calls to achieve the desired rate. Note that this adapts only the frequency with which the transmitter fills the socket buffer, and not the actual transmission rate. The host's TCP stack is still responsible for transmitting those packets onto the network media, by using TCP's flow and congestion control mechanisms. However, this behavior reflects exactly the challenge of QoS-aware TCP communication.

- *UDP traffic generator:* This traffic generator (created by Andy Adamson, University of Michigan) divides the flow into intervals of one second, in which UDP datagrams are transmitted at fixed frequency until a given per-second rate is achieved. Having achieved the desired amount of data per second, the transmission stops until the next interval starts. If the rate is not achieved, the size of the datagram used for the next second is increased by two bytes. If the sender has already submitted this amount of data for a second, it stops transmission until the next interval starts.

4.3 Evaluation of CAR

Our first experiments were designed to evaluate the impact of the policing performed by Committed Access Rate (CAR) on TCP flows. To this end, we sent flows over an otherwise uncongested link, with CAR applied at the ingress point.

4.3.1 Short-Term Transfers

As stated above, TCP reacts when dropped packets are detected. If the transmitter recognizes a lost packet, it falls into either the congestion control phase or the slow start phase, depending on the number of contiguous packets lost. This behavior dramatically affects TCP performance. The first experiment was designed to illustrate this impact, by demonstrating the behavior of a relatively small TCP transfer (16.7 MB), with different transmission rates in combination with CAR.

The results displayed in Figure 3 were gathered over several sessions by transmitting a TCP stream at a prespecified rate and altering the CAR configuration. The impact of the token bucket depth[1] is recognizable. Increasing the normal burst size results in a higher short-term bandwidth, until CAR's exceeding policy comes into play. Note that the increase of the bandwidth from 4700 KB/s to 5200 KB/s corresponds to the transmission time and the difference in the bucket depth. Having additional tokens for 1.8 MB allows a stream to exceed its limit by 500 KB/s for nearly 4 seconds. Overall, the transmitter is transferring 16.7 MB, which also can be done in less than 4 seconds. For an exact calculation it should be mentioned that the transmitter was using the path MTU discovery. For that reason the segment size used was 1460 bytes.

This experiment demonstrates effectively the behavior of TCP's slow-start feature. As soon as the token bucket is empty, the router starts dropping packets and often drops consecutive packets. The TCP transmitter recognizes that consecutive packets are lost and reacts to this by shrinking the congestion window to two, which has an enormous short-term impact on the actual throughput.

From this, one can conclude that exceeding the actual rate limit causes TCP to decrease performance drastically. Transmitting packets faster than the QoS rate limit has a negative impact on overall performance. Note that TCP might submit a whole socket buffer in one burst as permitted by the offered receiver window and the congestion window. The actual CAR configuration must be able to handle those bursts without dropping packets. Currently the normal burst size is limited to 2 MB. This limit has the side effect of shrinking the

[1] Cisco has implemented the rate policies with a token bucket mechanism. The depth specifies the maximum amount of tokens a token bucket can store.

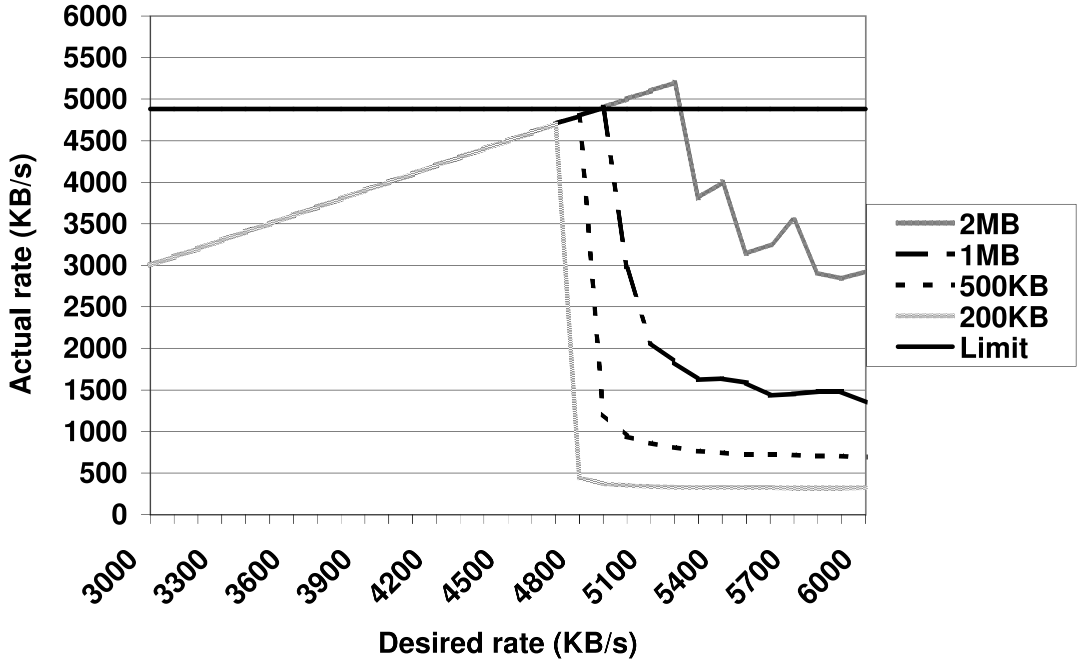

Figure 3: Received throughput of different short-term TCP streams with an active CAR configuration on the ingress site. The token bucket depth was varied between 200 KB and 2 MB. The average rate limit used for configuring CAR was constant.

maximum supported window size to 2 MB. Cisco has mentioned that future versions of CAR might be able to handle deeper token buckets.

4.3.2 Long-Term Transfers

Besides the short-term behavior of TCP streams, it is important to analyze a stream with a longer duration. For that reason several long-term TCP sessions were monitored. Figure 4 represents an example for the behavior of a TCP session exceeding its rate limit. In this case the normally constant throughput starts oscillating. As soon as CAR starts dropping packets, the transmitter reacts with TCP's slow-start feature. This reduces the transmission rate drastically. As the router's token bucket begins filled again, the transmitter is able to exceed its limits again (if enforced by the application) and hence resumes its oscillating behavior.

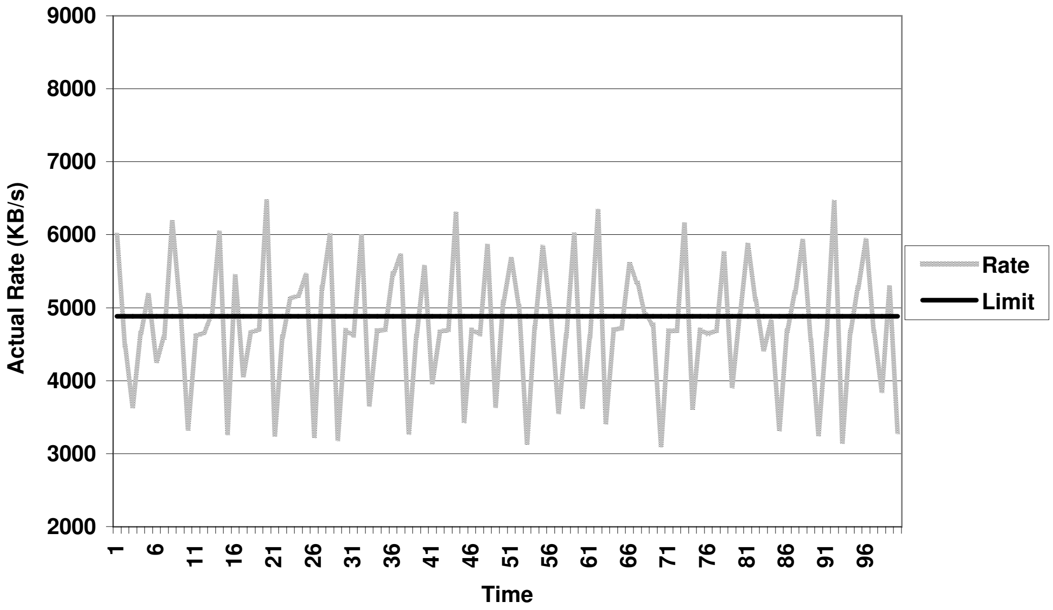

Figure 4: Received throughput of a TCP stream with an underestimated active CAR configuration on the ingress site. The attempted transmission rate was 6 MB/s. CAR was configured with a bucket depth of 2 MB.

The conclusion is that exceeding the actual rate limit causes long-term TCP traffic to oscillate. This is mainly because of the slow start/congestion avoidance feature and the bucket depth. The greater the bucket depth, the greater the variation.

4.4 Evaluation of WFQ

As stated before, Weighted Fair Queuing (WFQ) is conceptionally well suited for supporting the IETF's EF PHB. This section describes experiments done to evaluate the functionality of WFQ. Because we are focusing on end-to-end QoS, we were more interested in analyzing whether a specific TCP goodput of a high-end application could be guaranteed under congestion than analyzing the jitter of a UDP stream.

We used the following test suite to create well-defined congestion. There was a TCP stream trying to send a unidirectional stream of 10 MB/s. The source was a host on the left-hand site of GARNET; the destination was a host on the right-hand site (see Figure 2 for details). The edge routers were configured to mark packets of that stream and limit its rate to 11 MB/s. To emulate the behavior of a wide-area TCP application, we used a window size of 1 MB. Note that we were also required to increase the maximum congestion window size of our Solaris 7 end-system to 1 MB because the default limitation is 262 KB.

After the TCP stream was running for a while, a UDP stream was started between the other Sun workstations, across the same link. Every 0.1 ms, packets of 500 bytes were submitted, resulting in an application bandwidth of 5 MB/s.

To create congestion, we shrank the PVC between the interior router and the egress router to 12.5 MB/s (100 Mbps). Note that this bandwidth includes all overheads of an ATM link; hence, the actual supported IP bandwidth is much lower, depending on the actual packet size.

4.4.1 Impact of Congestion on a TCP Flow without QoS

We performed several experiments. The first was done to demonstrate the behavior of commodity routers without WFQ. We examined this non-DS case in order to compare it with the DS case when there are no DS flows, because we want to be sure that the DS mechanisms do not impact the best-effort traffic. Figure 5 shows two graphs:

- The amount of outstanding data is displayed as upper line. It indicates the number of packets that are not yet acknowledged. At 4 seconds the congestion started. The amount of outstanding data immediately increased to its limit of 1 MB; thereafter it decreased to 500–600 KB. The peak at 11 seconds was caused by a miscalculation due to CPU scheduling. It has no corresponding entry in the round-trip graph described below.

- The graph of the round-trip time (rtt) needs to be read carefully. TCP implementations do not acknowledge every received segment. The delayed acknowledgment mechanism delays this until the acknowledgment can be piggy-backed, or a timer expires (50 ms in the testbed). For that reason, the minimum value of this graph represents the actual round-trip time, but the maximum value indicates the value that is of interest for calculating the bandwidth-delay product for optimizing the window size. The graph itself correlates to the outstanding data. Without congestion the actual round-trip time is as low as expected. As congestion started, the router started queuing packets. This increased the round-trip time enormously. Under congestion the round-trip time was still extremely high. As expected, TCP did almost no piggy-backing of acknowledgments, because the actual transmission time was larger than the timeout of 50 ms. The important fact to note is that at the time the congestion occurred, some packets seemed to get dropped. The graph indicates that the TCP flow was going into the into the slow-start phase. As a consequence, the transmitter backed off its transmission speed drastically. One can see that TCP went to slow-start mode because the amount of outstanding data was reduced to a very small value (two segments). Now there was no longer congestion (the UDP traffic was submitting only 5 MB/s). As long as TCP was in slow start mode, it increased its window size exponentially. Having reached half of the former window size of 1 MB, it went into the congestion control phase. Now the

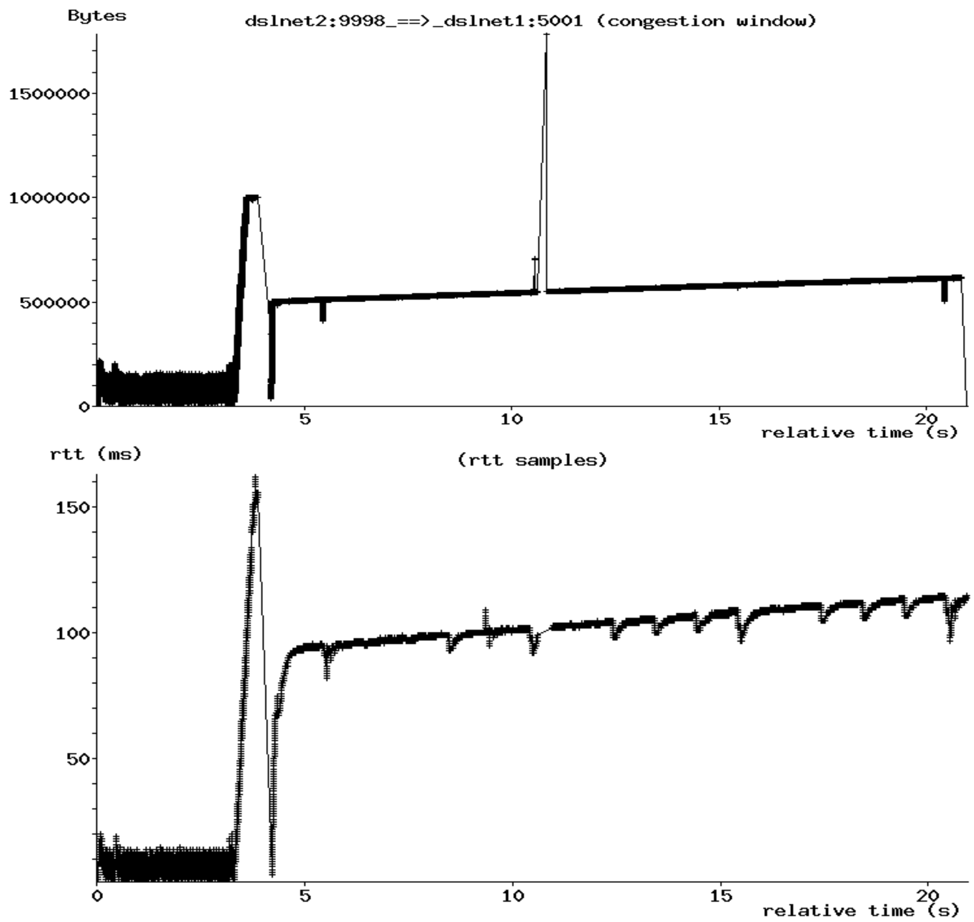

Figure 5: Performance achieved for a TCP flow under congestion on GARNET. We demonstrate the behavior of our experiment without any WFQ configuration. See text for details.

window size increased only linearly. As a consequence the throughput was increasing continuously, but only by a small amount. This created more and more congestion on the network, which led to a higher round-trip time that slowed the increase of the window size permanently.

Figure 5 demonstrates that commodity networking hardware does have a significant amount of buffer capacity. It is not a problem for the router to buffer more than 500 KB of data. For that reason it is quite important to evaluate the impact of a WFQ configuration emulating the IETF's EF PHB.

4.4.2 Impact of Congestion on a Non-Premium TCP Flow with Standard WFQ Buffer Limits

The following graph corresponds to the same experiment without any premium traffic. This time, however, WFQ was configured to provide under congestion 98% of the output bandwidth to the premium class. The goal was to demonstrate the impact of a standard WFQ configuration on best effort traffic. Our hope was that enabling WFQ would not have any impact on best-effort traffic.

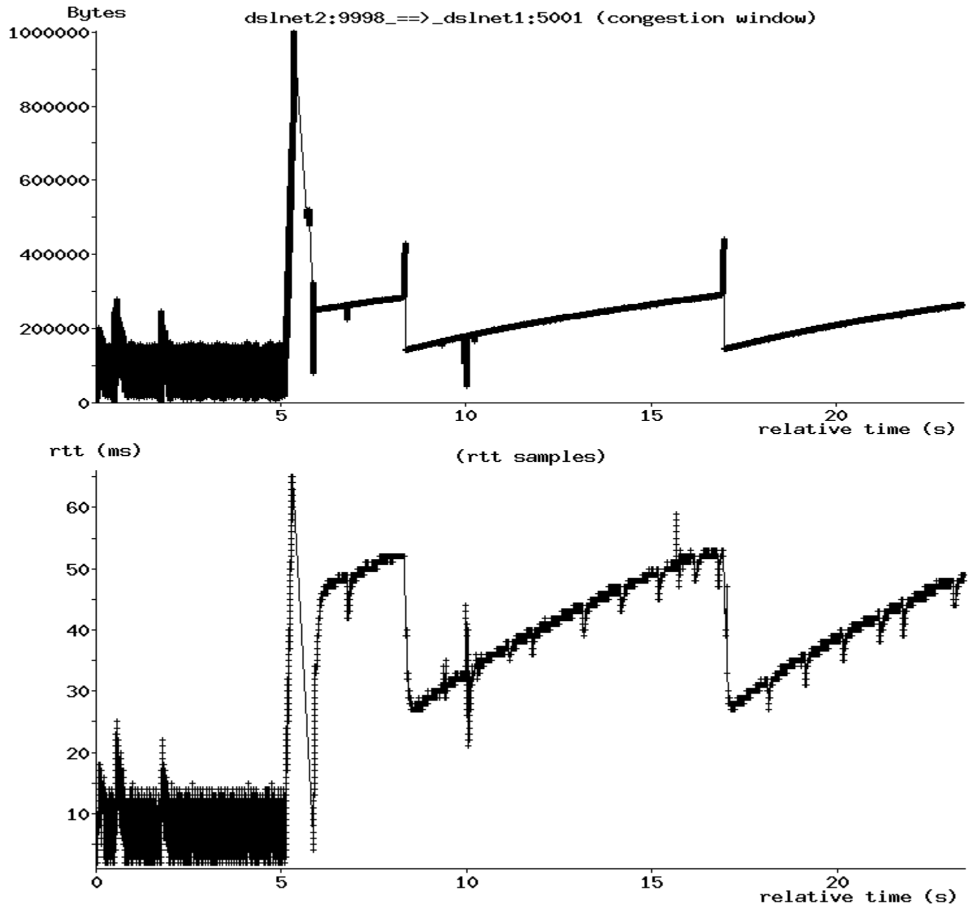

Figure 6: Performance achieved for a TCP flow under congestion on GARNET. We demonstrate the behavior of our experiment without any WFQ configuration. See text for details.

Figure 6 shows two graphs:

- The amount of outstanding data looks different this time. Whereas the configuration without WFQ was able to keep a 1 MB window outstanding over several hundreds of milliseconds, this setup was not able to handle this burst. Though the number of outstanding bytes quickly reached the 1 MB limit, the actual amount of data buffered correctly was less than 550 KB. This is indicated by the amount of outstanding data in the following congestion control phase, which is half of the former congestion window size. Even an amount of 550 KB was not handled properly over this period. Because TCP was now increasing the actual window size only linearly—before it went to slow-start mode again, it was increasing the size exponentially—it gives a better overview of the amount of data buffered correctly. In this case, it falls into the congestion control phase again at 300 KB, which reflects the amount of buffer space available under this condition.

- For the round-trip time, results are similar to the amount of outstanding data. Because of the smaller amount of buffer space, the actual round-trip time is significantly less. The maximum round-trip time handled correctly is approximately 52 ms. Assuming that the 100 Mbps PVC goodput capacity of less than 11 MB/s (86 Mbps; after SONET, ATM, and protocol overhead), and assuming that the amount of time the acknowledgment packet needs is 2 ms, the overall amount of buffer space is around 530 KB. Nearly two-thirds of this buffer space is available to the TCP flow: 350 KB. The remaining difference between this rough calculation and the observed amount is caused by the fact that the UDP packets contain fewer packet headers and by the fact that the WFQ implementation handles its memory management in chunks of 512 bytes. This causes an internal fragmentation of the available queue space.

The reason for this behavior is that Cisco's WFQ implementation divides nearly the whole available amount of queue space to the aggregate specific queues based on their weight. Hence, a best-effort aggregate with a guarantee of 2% of bandwidth under congestion receives only a small amount of queue space. Following the idea of a premium service, it does not make sense to provide a significant amount of queue space to the premium flow, because it would result in a potentially higher latency. Consequently, an implementation of the EF PHB should change the default buffering dramatically by providing a significant amount of queue space to the best-effort queue.

A proper configuration could reproduce the same behavior observed with the non-WFQ configuration, even with assigning 99% to the premium queue. Figure 7 illustrates the result with a correct WFQ configuration. It agrees with the result illustrated by Figure 5.

4.4.3 Impact of Congestion on a Premium TCP Flow with Correct Configuration

We next demonstrate that the proposed implementation of the EF PHB really provides a premium channel. WFQ guaranteed 99% of the available bandwidth to the premium class. For test purposes, the policy on the ingress site allowed 90% of the available bandwdith. In a real environment, however, the absolute amount of allowed premium bandwidth would be less. Figure 8 shows two graphs:

- The amount of outstanding data increases only slightly (<10%) when congestion startes at 5.5 seconds. Note that the scale of this graph differs from the previously shown outstanding data graphs, because this configuration does not introduce a buffering of segments inside the router.

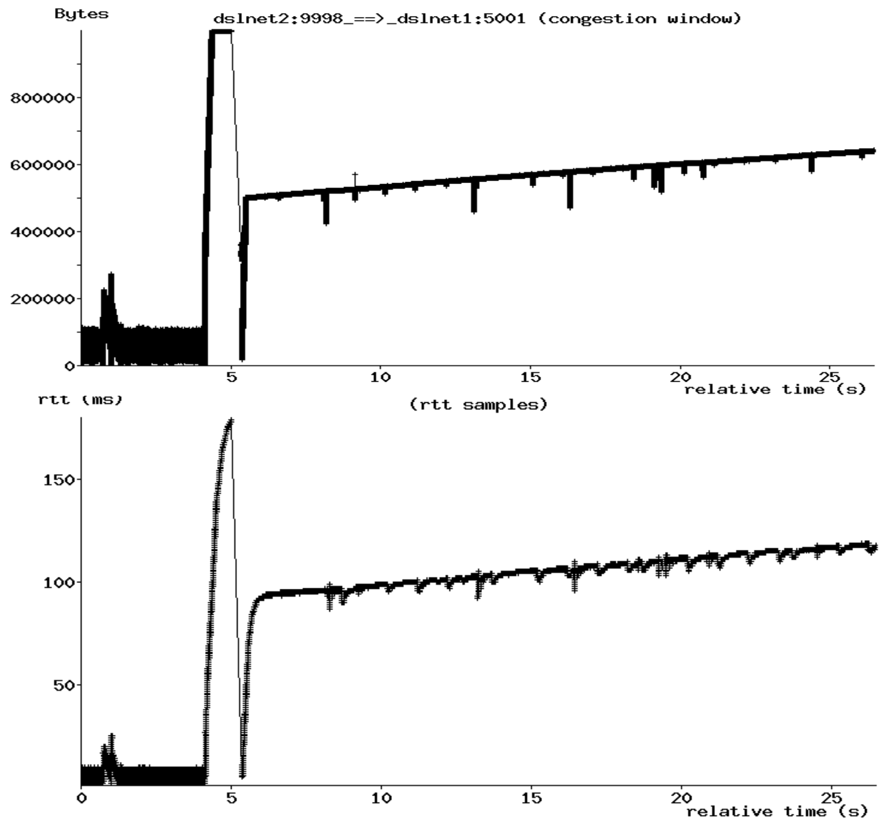

Figure 7: Performance achieved for a TCP flow under congestion on GARNET. We demonstrate that a correct WFQ configuration does not have any impact on congestion within the best effort class.

- The round-trip time of the premium traffic is almost unaffected by the congestion. We see only a minor increase from 3 ms to 4 ms, and this increase results in part from the granularity of the graphing tool, which is able to display round-trip times only in units of milliseconds.

Figure 9 displays the actual average throughput over the time. As one can see, it is not influenced by the congestion.

5 Proposed Differentiated Services Configuration

Based on the experiments we propose the following DS configuration: CAR and WFQ should be used for QoS enforcement, as described above. Flow specifications supplied to CAR should use a bandwidth computed from the user-specified required bandwidth, taking into account

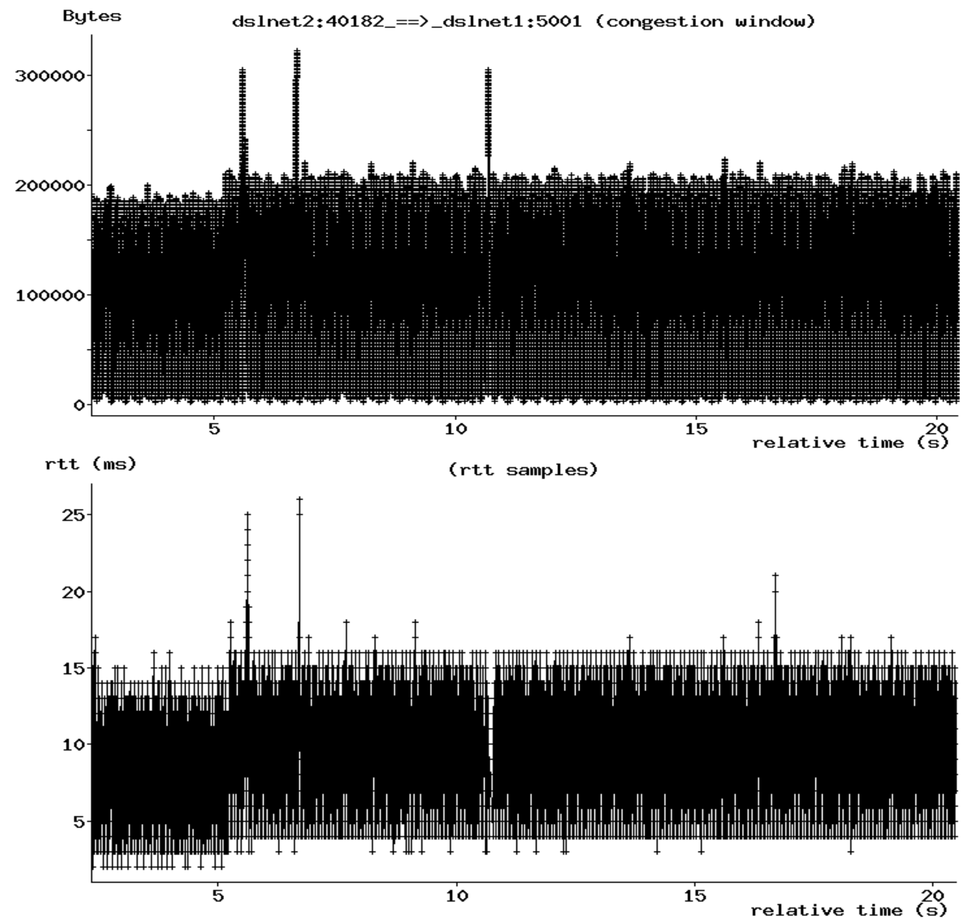

Figure 8: Performance achieved for a TCP flow under congestion on GARNET. We demonstrate the behavior of a premium high-bandwidth TCP stream. See text for details.

packet headers (note that this requires packet size information); packets that exceed the rate-limit should be dropped. The burst size parameter should be set to the bandwidth (in bytes/second) multiplied by the assumed maximum round trip time.

To implement the EF PHB without traffic shaping, we recommend assigning 99% of the available bandwidth of the slowest link (BW) to the premium class of WFQ. This is not as extreme as it may seem at first glance. Because the edge routers are using CAR to limit the rate of premium flows, premium traffic will never actually take 99% of the bandwidth except, perhaps, for very short bursts. However, providing 99% to the queue for premium traffic means that WFQ will serve packets very quickly after they arrive in the queue, thus minimizing the queuing and therefore decreasing the delay that the premium traffic encounters.

We emphasize that guaranteeing a smaller amount of bandwidth to the premium aggregate

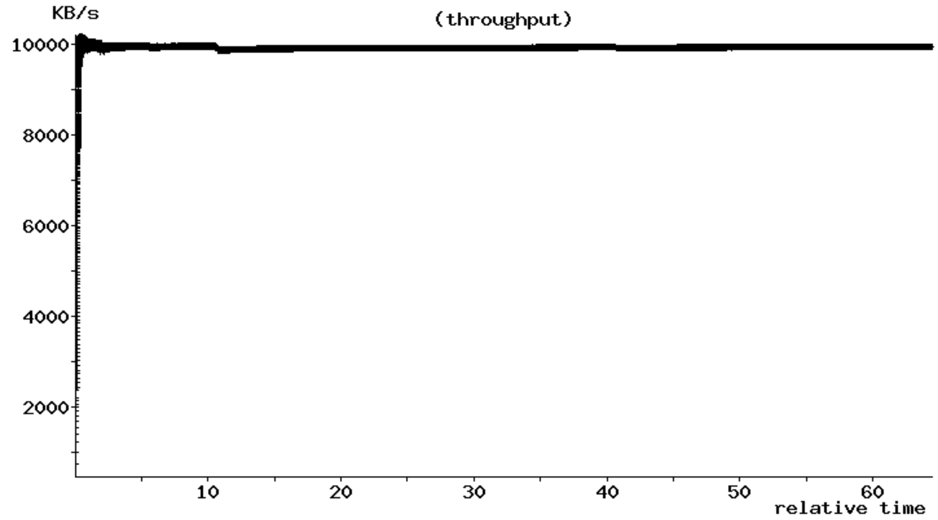

Figure 9: Throughput achieved for a TCP flow under congestion on GARNET. We demonstrate that the achieved throughput remains constant, even when the network is congested. See text for details.

would not prevent a premium flow from using up to 99% of the available bandwidth if there is no congestion or rate-limiting. This is because WFQ comes into play only if there is congestion and the proposed configuration does not rely on traffic shaping yet. We will now show that this proposed configuration does, in fact, minimize the queuing time for packets.

All other interior interfaces should exactly guarantee 99% of the rate BW. In doing this, we can now calculate the maximum queuing time introduced for the premium traffic. Let $x*BW$ be the amount of premium bandwidth allowed by the policy (e.g, the average rate limit of CAR), where $0 < x < 1$. Furthermore, let rtt be the round-trip time with no competing traffic and qt the maximum queueing time for the premium traffic, introduced by bursts. Note that qt is the time the last packet of a burst remains in the router queue until it gets transmitted. Assuming that there is no additional queuing delay for the acknowledgments, we can calculate the maximum estimated round-trip time used by CAR to calculate the token bucket depth with $rtt+qt$. Assuming this round-trip time CAR actually limits the maximum size of a burst to $x * BW * (rtt + qt)$. Because we can guarantee a premium bandwidth of $0.99 * BW$, we can calculate the queuing time with

$$\frac{x * BW * (rtt + qt)}{0.99 * BW} = qt.$$

Assuming $x < 0.99$, we get

$$qt = \frac{rtt * x}{0.99 - x}.$$

For example, limiting the allowed premium bandwidth to 33% of the network capacity ($x = 0.33$) results in a maximum increase of latency of $\frac{rtt}{2}$.

The buffer size allocated to the WFQ queues should leave a significant amount of space. This could be used on demand by either the best-effort class or the premium class. With this configuration we minimize the impact on handling traditional best-effort traffic under congestion. It is important to note that the usage of the overall queue limit is available only with the standard tail-drop behavior of the class of queue, and not with activating WRED on WFQ queues.

6 Application Example

We have constructed our DS resource manager to support two classes [9] of premium service: a foreground service, for latency- and jitter-sensitive flows (e.g., multimedia streaming and control), and a background service, for long-lived high bandwidth but latency-insensitive flows (e.g., bulk data transfer operations). The resource manager changes background reservations dynamically as foreground reservations come and go, generating callbacks to the application when a reservation changes. This strategy allows bulk data transfers to coexist with multimedia flows. The amount of bandwidth available for background reservations over a particular time period can then be controlled via policy mechanisms. We report results with this approach below. Our prototype supports multiple foreground reservations but initially only a single background reservation; the extensions required to support multiple background flows are not complex.

Figure 10 shows the bandwidth delivered to the foreground, background, and best-effort flows during the experiment. We configured GARNET to create a 45 Mb/s premium channel in a 100 Mb/s network. We then created five distinct flows: a bulk data transfer, operating as a "background" flow; a competing 80 Mb/s best-effort UDP flow (a traffic generator submitting 1,000 byte packets every 100 μs); and three independent, short-lived foreground flows with immediate reservations. In this experiment we used a simple data transfer program, which adapted its transmission rate to the rate assigned to the background class.

7 Related Work

The general problem of QoS implementation and management is receiving increased attention (see [10]). Some groups have investigated the use of DS mechanisms (e.g., [23, 19]) and its impact on TCP. Sahu et al. [19], in particular, present a quantitative comparison of different DS router mechanisms. However, none of the studies has explicitly addressed flows consuming a goodput of 10 MB/s or higher and using commodity router hardware that provides buffer capabilities of up to several megabytes. Further information about the QoS framework offered by GARA, including the combination of Network and CPU reservations

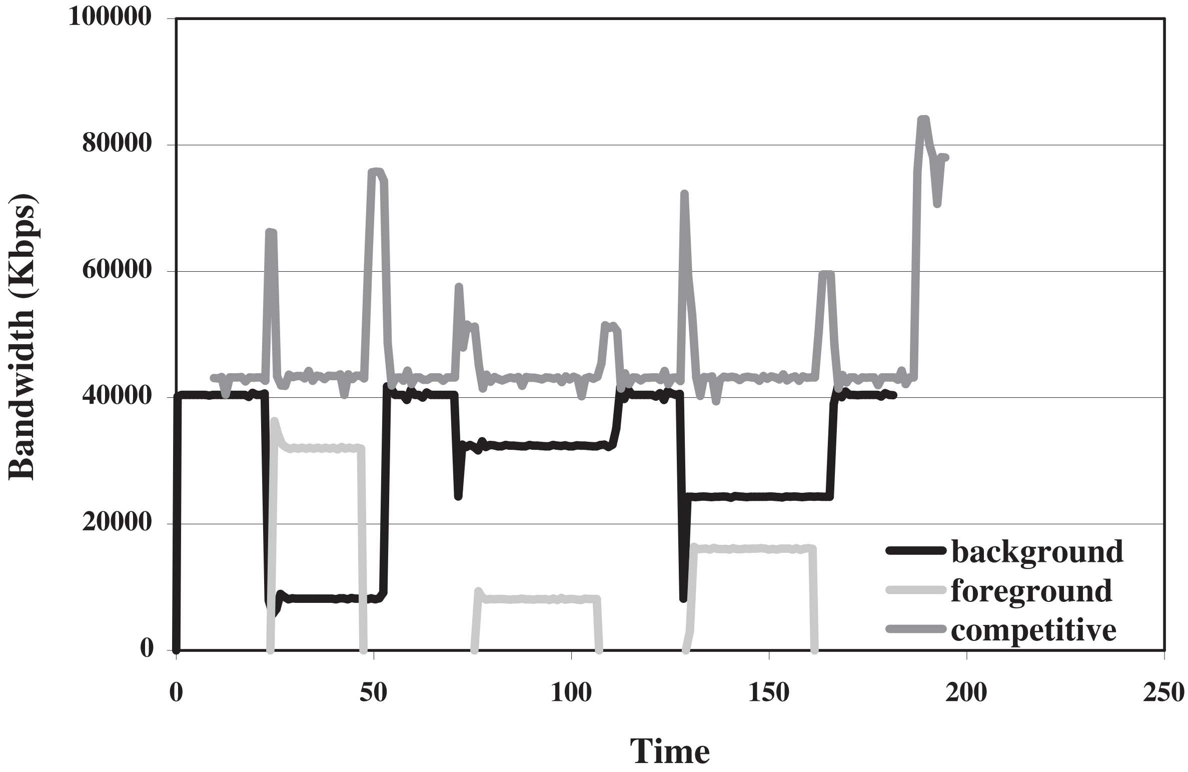

Figure 10: Performance achieved for a mixture of premium and best effort services on GARNET. We demonstrate that a bulk-transfer (background) application is able to exploit unused premium traffic without affecting foreground reservations. See text for details.

and wide-area experiments, can be found in [9].

The pan-European research network and its task force Testing Advanced Networking Technologies (TF-TANT) are currently evaluating DS mechanisms using commodity hardware. TF-TANT addresses aspects similar to those discussed in this paper but focuses on flows with lower bandwidth requirements. The results have not yet been published but are available on the Web [6].

8 Conclusions and Future Work

We have presented a quantitative evaluation of DS implementation for high-performance TCP-flows and demonstrated that end-to-end QoS can be delivered to such flows if DS mechanisms are configured carefully. The basic challenge is dealing with the burstiness introduced by TCP's sliding window. This must be addressed by appropriate policies at the edge routers; these policies must support bursts correlating to the TCP window size. Burstiness also introduces problems on the interior interfaces, because the available bandwidth might be exceeded by accumulating aggregate bursts. For that reason the implementation of the EF PHB should avoid queuing by overprovisioning the guaranteed amount of bandwidth for the premium class as much as possible.

We have shown that CAR and WFQ can be used to implement a DS architecture. Policing at edge routers is done based on the applied bandwidth and the estimated round-trip time. The EF PHB is implemented by guaranteeing 99% of the available bandwidth to the premium class, which minimizes the maximum latency of premium traffic. We have demonstrated that the impact of this configuration on best-effort traffic in the absence of premium traffic is negligible.

In future work we plan to improve the proposed implementation by introducing Distributed Traffic Shaping (DTS) and by analyzing threshold dropping for premium flows exceeding its policy limit at the edge. We will also analyze the impact of priority queuing as an alternative to WFQ.

Acknowledgments

We gratefully acknowledge assistance given by Robert Olsen who provided additional information about Cisco's DS implementation, by Andy Adamson who wrote the UDP traffic generator, and by Jun Wang who assisted in our measurement effort. Numerous discussions with our colleagues Gary Hoo, Bill Johnston, Carl Kesselman, and Steven Tuecke have helped shape our approach to quality of service. We also thank Cisco Systems for an equipment donation that allowed the creation of the GARNET testbed. This work was supported in part by the Mathematical, Information, and Computational Sciences Division subprogram of the Office of Advanced Scientific Computing Research, U.S. Department of Energy, under Contract W-31-109-Eng-38; by the Defense Advanced Research Projects Agency under contract N66001-96-C-8523; by the National Science Foundation; and by the NASA Information Power Grid program.

References

[1] M. Allman, V. Paxson, and W. Stevens. TCP Congestion Control. *Internet RFC 2581*, 1997.

[2] S. Blake, D. Black, M. Carlson, M. Davies, Z. Wang, and W. Weiss. An Architecture for Differentiated Services. *Internet RFC 2475*, 1998.

[3] R. Braden, D. Clark, and S. Shenker. Integrated Services in the Internet Architecture: an Overview. RFC 1633, Internet Engineering Task Force, 1994.

[4] D. Comer. *Internetworking with TCP/IP*. Prentice-Hall International Editions, 1988.

[5] W. Feng, D. Kandlur, D. Saha, and K. Shin. Adaptive Packet Marking for Providing Differentiated Services in the Internet. In *Proceedings of the International Confrerence on Network Protocols*. 1998.

[6] Editor: Tiziana Ferrari. Differentiated Services Experiment Report. Technical report, CNAF, 1999. `http://www.cnaf.infn.it/˜ferrari/tfng/ds/rep1-del.doc`.

[7] I. Foster and C. Kesselman, editors. *The Grid: Blueprint for a Future Computing Infrastructure*. Morgan Kaufmann Publishers, 1999.

[8] I. Foster, C. Kesselman, C. Lee, R. Lindell, K. Nahrstedt, and A. Roy. A Distributed Resource Management Architecture that Supports Advance Reservations and Co-Allocation. In *Proceedings of the International Workshop on Quality of Service*, pages 27–36, 1999.

[9] I. Foster, A. Roy, V. Sander, and L. Winkler. End-to-End Quality of Service for High-End Applications. Technical report, Mathematics and Computer Science Division, Argonne National Laboratory, Argonne, 1999. `http://www.mcs.anl.gov/qos`.

[10] Roch Guérin and Henning Schulzrinne. Network quality of service. In *[7]*, pages 479–503.

[11] J. Heinanen, T. Finland, F. Baker, W. Weiss, and J. Wroclawski. Assured Forwarding PHB Group. *Internet RFC 2597*, 1999.

[12] V. Jacobson and M. Karels. Congestion Avoidance and Control. In *Proceedings of the SIGCOMM*. 1988.

[13] V. Jacobson, K. Nichols, and K. Poduri. An Expedited Forwarding PHB. *Internet RFC 2598*, 1999.

[14] T. Lakshman, U. Madhow, and B. Suter. Window-based Error Recovery and Flow Control with a Slow Acknowledgement Channel: A Study of TCP/IP Performance. In *Proceedings of the IEEE INFOCOM*. 1997.

[15] M. Mathis, J. Semke, and J. Mahdavi. The Macroscopic Behavior of the TCP Congestion Avoidance Algorithm. In *Proceedings of ACM SIGCOMM, volume 27, number 3*. 1997.

[16] M. Mathis, J. Semke, and J. Mahdavi. Automatic TCP Buffer Tuning. In *Proceedings of ACM SIGCOMM, volume 28, number 4*. 1998.

[17] K. Nichols, S. Blake, F. Baker, and D. Black. Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers. *Internet RFC 2474*, 1998.

[18] R. Nitzan and B. Tierney. Experiences with TCP/IP over an ATM OC12 WAN. LBNL Report, Lawrence Berkeley National Laboratory, April 1999.

[19] Sambit Sahu, Don Towsley, and Kim Kurose. A Quantitative Study of Differentiated Services for the Internet. UMass CMPSCI Technical report, University of Massachusetts, Sept. 1999.

[20] W. Stevens. TCP Slow Start, Congestion Avoidance, Fast Retransmit, and Fast Recovery Algorithms. *Internet RFC 2001*, 1997.

[21] W. Stevens. *TCP/IP Illustrated, Vol. 1 The Protocols.* Addison-Wesley, 1997.

[22] I. Yeom and N. Reddy. Realizing Throughput Guarantees in a Differentiated Services Network. In *Proceedings of the ICMCS, June 99.* 1999.

[23] Ikjun Yeom and A. L. Narasimha Reddy. Modeling TCP Behavior in a Differentiated-Services Network. Technical report, TAMU ECE, 1999.

9 Vitae

Volker Sander is a visiting scientist at Argonne National Laboratory's Distributed Systems Laboratory (DSL). He has been actively researching issues in metacomputing since 1995 at Forschungszentrum Juelich GmbH. Focal point of his current research field is Quality of Service, Remote I/O, and Security.

Dr. Ian Foster is Senior Scientist and Associate Director in the Mathematics and Computer Science Division at Argonne National Laboratory, and also Associate Professor of Computer Science at The University of Chicago. His research interests include parallel and distributed computing, and computational science. He has written 4 books and over 100 technical articles in these and related areas.

Alain Roy is a graduate student from the University of Chicago. He has been actively researching issues in Quality of Service since 1997. He current research is with GARA, a system that allows users to make advanced reservations for not only networks, but for CPUs, disks, and even graphics pipelines.

Linda Winkler is a Senior Network Engineer at Argonne National Laboratory's Mathematics and Computer Science Division. She is currently serving as the MREN Technical Director and is a primary member of the STARTAP engineering team. Her focus since 1995 has been in the area of interconnectivity and interoperability of wide-area research networks in support of advanced scientific and engineering applications.