Building Predictable Services Using AF

Wu-chang Feng
Department of EECS
University of Michigan
Ann Arbor, MI 48105
Phone: (734) 763-5363 Fax: (734) 763-4617
E-mail: wuchang@eecs.umich.edu

Abstract

The IETF is considering a more evolutionary approach for providing service differentiation in the Internet based on using the type-of-service (ToS) bits in the IP header. Through the Differentiated Services (DIFFSERV) working group, a small set of building blocks are being defined which give the network the ability to provide predictable services to applications. One of the mechanisms being considered for deployment is the Assured Forwarding (AF) Per-Hop Behavior (PHB). AF specifies a low-loss forwarding behavior for packets which are marked with it. This article surveys some of the problems in implementing AF correctly in the network and in building predictable services on top of AF. This article also surveys a number of simple mechanisms which have been proposed to address these problems.

1 Introduction

In order to provide immediate support for quality of service over the Internet, the Internet Engineering Task Force (IETF) is considering an evolutionary approach to provide service differentiation using the type-of-service bits (ToS) bits in the IP header [1, 2]. Through the Differentiated Services (DIFFSERV) working group, these bits have been renamed the "DS field" and the functions associated with them have been redefined. The crux of the DIFFSERV approach is to standardize a simple set of mechanisms for handling packets with different priorities as encoded in the DS field. Figure 1 shows the basic architecture of the DIFFSERV approach. As the figure shows, traffic conditioners such as shapers, DS-markers, and droppers are placed at the edges of the network. Given this functionality at the edge, the interior routers then use the priority markings of packets in order to deliver differentiated services to various packets. This provides a very basic QoS architecture in which most of the complexity is pushed to the edges of the network where it is the most scalable.

Because of the limited amount of bits available for use in the DS field, the DIFFSERV working group has defined a small set of building blocks which can be used to construct a number of

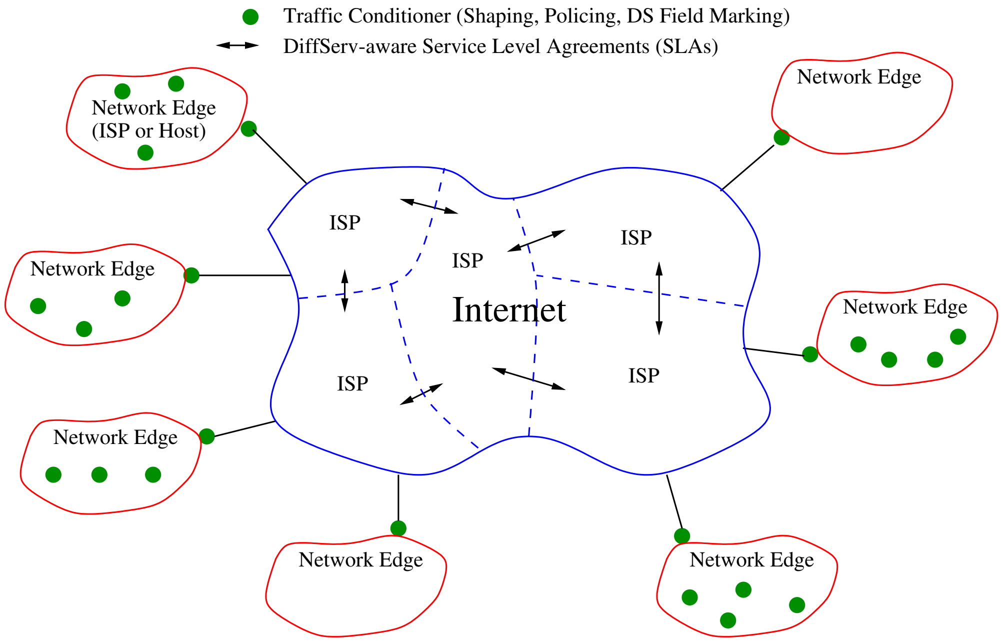


Figure 1: DiffServ architecture

services. These building blocks, called per-hop behaviors (PHBs), are encoded in the Differentiated Services Codepoint (DSCP) part of the DS field and specify the forwarding behavior that each packet receives from individual routers in the Internet. When used on an end-to-end basis, it is envisioned that these building blocks can be used to construct a variety of services which are able to support a range of emerging applications. Among the initial PHBs being standardized are the Expedited Forwarding (EF) and the Assured Forwarding (AF) PHBs [3, 4]. The EF PHB specifies a forwarding behavior in which packets see a very small amount of loss and a very low queueing delay. In order to ensure every packet marked with EF receives this service, EF requires every router to allocate enough forwarding resources so that the rate of incoming EF packets is always less than or equal to the rate at which the router can forward them. In order to preserve this property on an end-to-end basis, EF requires that traffic be shaped and reshaped in the network. The AF PHB group, on the other hand, specifies a forwarding behavior in which packets see a very small amount of loss. The AF PHB group consists of four, independently forwarded classes. Within each class, two or three drop preference levels are used to differentiate between flows in the class. The idea behind AF is to preferentially drop best-effort packets and packets which are outside of their contract when congestion occurs. By limiting the amount of AF traffic in the network and by managing the best-effort traffic appropriately, routers can then ensure low loss behavior to packets marked with the AF PHB.

Through the use of strict profiles and trafic shaping, it is relatively clear how to build to build predictable end-to-end services such as the virtual leased line service using the EF PHB. On the other hand, since the AF PHB is more loosely defined it is unclear how it can be used to build predictable end-to-end services. The basic idea behind AF is to mark packets at the edge of the network based on a negotiated profile. If packets from a flow are within this profile, they are AF-marked and thus have their priorities increased. Within the network, routers and switches give AF-marked packets preferential treatment when congestion occurs and queues build up. In this situation, non-priority packets are dropped in order to accommodate high priority, AF packets. While the mechanisms for AF are relatively straight-forward, it is not clear whether AF can be used to provide predictable services. This article addresses two key issues in building predictable services with the AF PHB. In Section 2, we examine issues in implementing the AF PHB accurately in the network. Given a proper implementation of AF, Section 3 addresses some of the challenges which must be addressed in order to build end-to-end services using AF. Section 4 concludes the article.

2 Implementing AF

The general idea behind AF is to preferentially drop non-priority packets in times of congestion. There have been numerous proposals for doing this. Each proposal focuses on one of two goals: 1) protecting the network from unnecessary packet loss and 2) protecting priority packets over an arbitrary arrival pattern of priority and non-priority packets. This section summarizes these approaches and suggests a queue management algorithm which does both.

2.1 Enhancements to Red

One of the more popular approaches for implementing AF is to use mechanisms based on Random Early Detection (RED) queue management [5, 6]. In normal best-effort networks, RED is used to prevent unnecessary packet loss. The idea behind active queue management algorithms such as RED is to detect incipient congestion *early* and to convey congestion notification to the end hosts so that they can reduce their transmission rates before queue overflow and sustained packet loss occur. RED performs this task by maintaining an exponentially weighted moving average (EWMA) of the queue length which it uses to detect congestion. RED detects increases in the average queue length and uses it to determine whether or not to drop or ECN-mark a packet. A connection receiving congestion notification in the form of an ECN mark, cuts its congestion window in half or sets it to 1 as it would have if it had detected a packet loss. The marking/dropping probability of RED is a simple function of the average queue length, an initial probability parameter (max_p), a minimum threshold (min_{th}), and a maximum threshold (max_{th}). When the average queue length is less than or equal to min_{th}, this probability is 0. The probability linearly increases from 0 to max_p as the average queue length goes from min_{th} to max_{th}. When the average queue length exceeds max_{th}, all packets are dropped or marked. Using this algorithm, RED has been able to effectively reduce unnecessary packet loss over traditional drop-tail queues.

There have been a number of proposals for adapting the RED algorithm in order to implement the AF PHB. These proposals include RED with In/Out (RIO) [7], Enhanced RED (ERED) [8, 12], and Cisco's Weighted RED (WRED) [9]. These algorithms work by selectively applying RED's early detection mechanism based on the the priority marking of the packet. In each scheme, when congestion occurs and the EWMA slowly grows, non-priority and lower priority packets are discarded first. This mechanism preferentially delivers congestion notification to best-effort and low priority sources causing them to reduce their transmission rates. In contrast, flows which use AF-marking receive significantly smaller amounts of congestion notification and can continue to send at a higher rate.

While RED-based mechanisms can effectively implement the AF PHB in many situations, they suffer from an inability to completely protect priority packets. There are several key problems with using RED. One of the problems is that the performance and behavior of RED is extremely sensitive to the parameters it is configured with. For example, a misparameterized RED queue can fail to control the transmission rates of the best-effort and low priority sources, allowing them to fill the queue with packets and thus, cause priority packets to be dropped. Another problem is that RED's congestion detection mechanism is based on the calculation of an average queue length. Since the instantaneous queue length can change quite dramatically over short time intervals, queue overflow and loss of priority packets can occur before the increase in average queue length can trigger RED's congestion control mechanisms.

2.2 Protecting priority packets

While building AF implementations using RED allows the network to optimize the performance of non-priority packets, the AF PHB becomes meaningless when priority packets are not given sufficient protection from non-priority packets. In order to address this problem, a number of queue management algorithms have been proposed which advocate the strict allocation of buffer resources to packets marked with AF [10, 11]. The idea behind these algorithms is to control the sharing of buffer resources between priority and non-priority packets in order to minimize the number of priority packets which are lost.

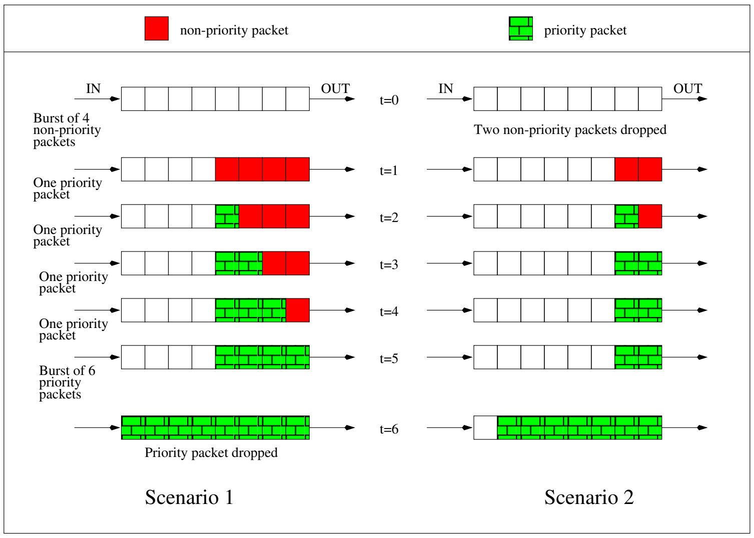


Figure 2: Protective Buffer Management

Figure 2 shows an example of how the queue management scheme impacts the forwarding behavior of priority and non-priority packets. The figure shows how a single packet arrival pattern can induce entirely different behavior out of a queue based on the queue management scheme used. In this scenario, a burst of 4 non-priority packets is followed by four priority packets spaced one transmission slot apart from each other. These packets are then followed by a burst of 6 priority packets. Scenario 1 shows the behavior of the queue when it is configured to allow half of the queue to be occupied by non-priority packets before they are dropped. In this scenario, all 4 non-priority packets in the initial burst are allowed to be enqueued and each one is forwarded as each of the next 4 priority packets are enqueued. When the burst of 6 priority packets arrives, however, the queue overflows and a *priority* packet is dropped. Thus, as the figure shows, by servicing non-priority packets earlier in time, the queue unknowingly causes subsequent priority packets to be dropped due to buffer overflow. Scenario 2 shows the behavior the queue when it is configured to allow only a quarter of the queue to be occupied by non-priority packets. As the figure shows, the queue drops two non-priority packets out of the initial burst of non-priority packets. By doing so, there is room left in the queue to accomodate the subsequent burst of priority packets.

The two scenarios above clearly demonstrate the need for strictly allocating buffer resources to priority packets based on their arrival rate. While this strict allocation can protect priority packets, without the use of active queue management techniques, these algorithms can be detrimental to non-priority packets. In particular, the queue management algorithms act as a drop-tail queue towards non-priority packets.

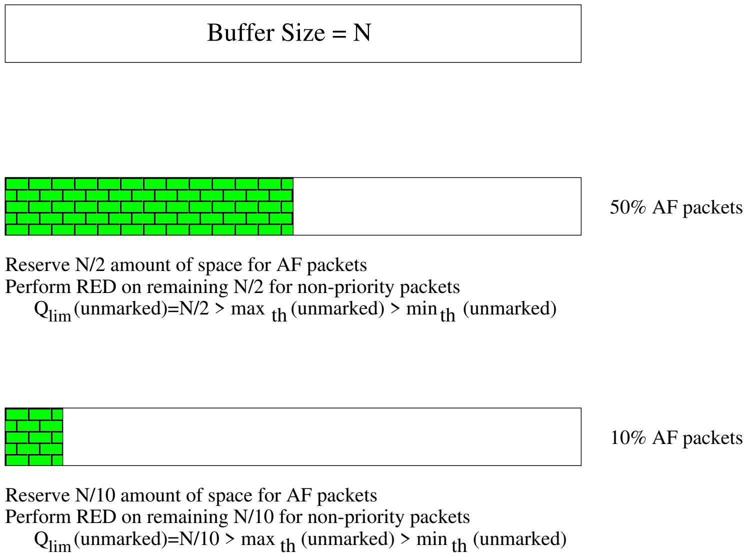


Figure 3: Protective Buffer Management

2.3 Protective RED

Given the issues described above, it is relatively easy to construct a protective, congestion-sensitive, queue management algorithm to support AF. By allocating a fixed amount of the buffer space to AF packets based on the overall arrival rate of AF packets and by performing RED queue management on non-priority packets with a maximum queue size that is equal to the buffer space left, priority packets can be protected while maximizing the performance of unmarked packets. Note that while priority and non-priority packets share the same FIFO queue, the hard limits on buffer occupancy effectively partition the buffer resources.

Figure 3 shows an example of a protective RED queue management algorithm. As the figure shows, a fixed amount of the buffer space is allocated for priority packets depending on the overall rate of AF traffic allowed through the queue. For example, if $X\%$ of the link's bandwidth is allocated to AF, $X\%$ of the buffer space is set aside for only priority packets. While non-priority packets are not allowed to occupy more than $1 - X\%$ of the buffer space, priority packets are allowed to occupy the entire buffer space if necessary. In order to improve the packet loss rates and performance of the non-priority packets, the RED algorithm is simply applied to the rest of the queue based only on the queue occupancy of non-priority packets. By adaptively changing the allocation based on the admitted AF traffic, this queue management algorithm maximizes the performance of non-priority packets while still protecting priority packets from packet loss.

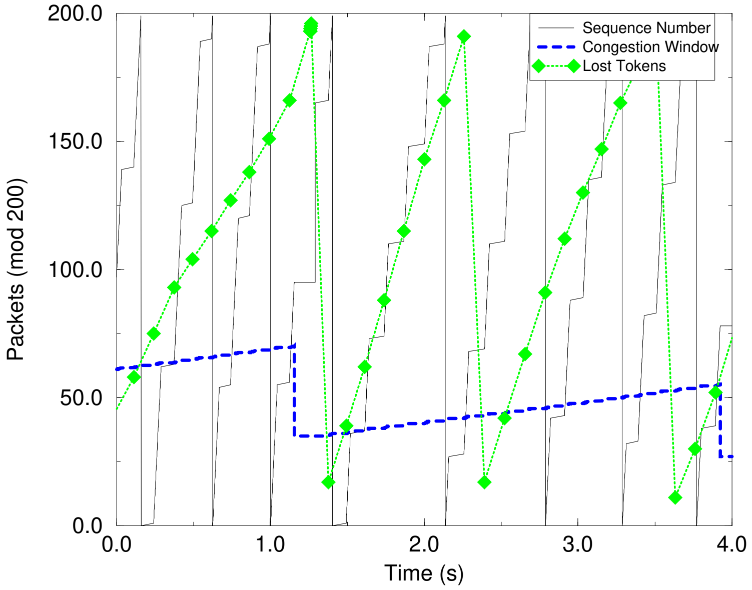


Figure 4: Packet trace of a policed, $4Mbs$ connection

3 Building AF Services

While correctly implementing the AF PHB in the network is fairly easy, building predictable, end-to-end services based on AF is not. This section examines several issues in implementing assured bandwidth services using AF. In particular, it examines two problems with the current AF architecture: 1) unpredictability in performance due to interactions with TCP and 2) unpredictability in performance due to overmarking. These problems prevent the construction of reliable and predictable end-to-end services using AF.

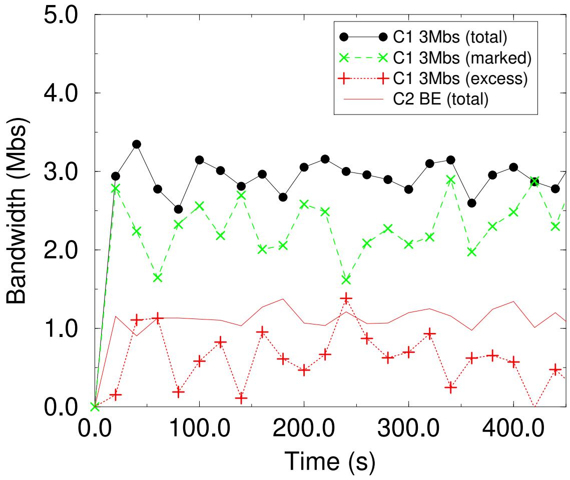
3.1 TCP Interactions

Because an overwhelming majority of applications use TCP and its congestion control algorithms, AF and its marking mechanisms must interact with TCP in a predictable manner. One of the problems with AF-style marking in the network is that it can interact rather poorly with TCP. The crux of the problem is the fact that the rate-control being done at the end hosts through TCP congestion control and windowing is disjunct and non-cognizant of the rate control being done in the network through the use of marking mechanisms.

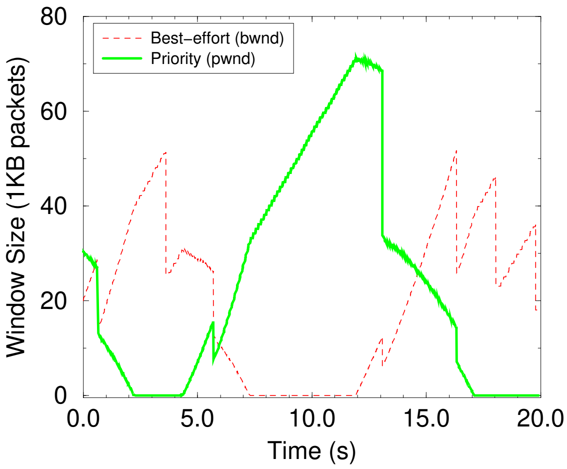
One of the proposed ways of marking using AF is to police the sources at the edge of the network using a token bucket. By specifying a token bucket with a maximum burst size and an overall rate of token generation, the worst-case arrival rates of priority packets at intermediate nodes can be accurately determined. In addition, the use of token buckets allows simple aggregation to be done across AF flows. Unfortunately, while token buckets have several salient features, it has been shown that a token bucket must be extremely deep in order to perform well with TCP [8]. The problem is that TCP is ACK-triggered. That is, the sender usually does not send a packet unless it receives an acknowledgment from previously transmitted data. As TCP awaits the receipt of an acknowledgment to send more data, the token bucket being used to police the source can easily overflow and tokens can be lost. TCP is inherently bursty, sending packets in bunches with significant gaps between bursts. Gaps in packet transmissions and acknowledgments can be caused by ACK compression, TCP's congestion recovery mechanisms, and congestion itself. Figure 4 shows a packet trace which demonstrates this problem. The figure plots the congestion window of a connection, its sequence number (modulo 200) and the number of lost tokens (modulo 200). In this experiment, an AF connection is marked by a token bucket with depth $25KB$ and a token generation rate of $4Mbs$. In order to ensure that no tokens are lost in this configuration, an acknowledgment must arrive at the source every $50ms$. Unfortunately, as the figure shows, through the course of the connection, gaps in the acknowledgment stream eventually develop causing the TCP sender to often freeze, as seen in the behavior of the sequence number plot. These gaps cause a significant amount of tokens to be lost throughout the trace. Note that gaps are especially problematic since they regenerate every round-trip time. Gaps in the acknowledgment stream cause gaps in packet transmission which cause subsequent gaps in the acknowledgment stream for TCP.

One way to solve this problem is to use extremely deep token buckets. For large token buckets to work, the depth of the bucket must be large enough to accomodate the largest gap in the acknowledgment scheme. Unfortunately, large token buckets increase the burstiness of the AF-marked packet stream. By allowing large bursts of AF-marked packets into the network, queue management schemes (such as protective RED) at intermediate routers must allocate a significant amount of buffer space to accomodate bursts in priority packets. While this can be done, allocating such a large amount can cause the buffer to be significantly underutilized. In addition, allocating more room for priority packets severely degrades the performance of non-priority packets sharing the same queue.

Another way of solving the token loss problem is to use markers which look at very large time intervals. For example, the Assured Service approach [7] attempts to do marking based on a time interval which captures the period of TCP's congestion window oscillations. The Constant Bit Rate service implementation described in [12] attempts to do probabilistic marking based on bandwidth measurements which also stretch across larger time intervals. While such marking policies can sometimes be successful, they have a significant problem in that the marking behavior



(a) Bandwidth graph



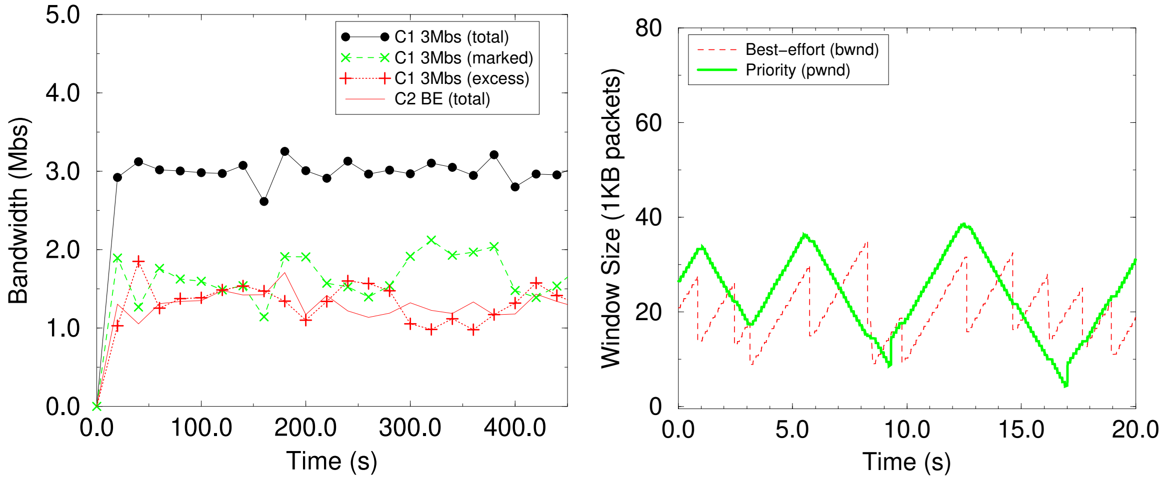
(b) Window traces of $3Mbs$ connection

Figure 5: Bandwidth sharing example

of the sources is hard to predict. For the Assured Service approach, it is extremely difficult for the network and the marker to determine the time interval of TCP's congestion window oscillations. The marking behavior of the CBR service implementation is also hard to predict since it depends on the bandwidth measurement intervals of the service. As a result, it is extremely difficult to aggregate such services predictably in the network. Without stricter bounds on the marking behavior (as with token buckets), the arrival pattern of AF packets at intermediate nodes cannot be characterized. Allowing arbitrary amounts of transient bursts in AF-marked packets puts a lot of pressure on the queue management algorithm being used in the network. In the absence of any shaping mechanisms for AF traffic, low loss can only be ensured to AF-marked packets when a large amount of buffer space is allocated to the high priority packets. If all of the sources are allowed to burst out a large number of priority packets, not even protective RED can prevent the loss of priority packets.

3.2 Overmarking

Another, more subtle problem with the currently proposed AF-marking mechanisms is that they can potentially overmark the packets of a connection. In an ideal scenario, a connection that has its packets striped accross two priorities should receive a fair share of the best-effort bandwidth in addition to the bandwidth received due to priority packets. Unfortunately, when AF-marking is done externally, this can not occur since priority packets are sent *in place of* best-effort packets rather then *in addition to* best-effort packets. Figure 5(a) shows an example of a scenario using



(a) Bandwidth graph (b) Window trace of $3Mbs$ connection

Figure 6: Bandwidth sharing example using an integrated approach

the Constant Bit Rate marker where one source is adaptively AF-marked to maintain a $3Mbs$ throughput. The figure shows the bandwidth of a best-effort source, the total bandwidth of the $3Mbs$ connection, and the number of priority and non-priority packets the $3Mbs$ source sends to achieve its target bandwidth. Note that the $3Mbs$ source gets much less than a fair share of the best-effort, non-priority bandwidth. In order to maintain its target, it must AF-mark a large number of its packets. Figure 5(b) shows the effective window traces of the $3Mbs$ connection. These traces plot the number of outstanding priority and non-priority packet the connection has. As the figure shows, the connection does not compete fairly for a share of the best-effort bandwidth when it sends priority packets.

The fact that a significant amount of additional packets are marked in order to achieve a particular bandwidth sharing configuration is detrimental to the performance of both the Assured and the Constant Bit Rate services. As an increasing number of AF-marked packets are allowed into the network, the performance of the queue management schemes in the network deteriorates. For example, as described in the previous section, a larger amount of buffer space must be allocated to priority packets in order to guarantee low loss. In addition, when a queue is constantly filled with priority packets, it soon degenerates into a normal drop-tail queue.

3.3 Possible Solutions

In order to address some of the problems outlined above, several efforts to integrate the AF-marking with the congestion control mechanisms at the end host have been examined. One proposal is to adapt TCP's acknowledgment-triggered transmission mechanism. For example, in the token bucket approach, adding a fine-grained timer which triggers the transmission of segments in TCP can be used to eliminate the effects of gaps in the acknowledgment stream. Another approach would be to shape the outgoing segments or the incoming acknowledgments in an attempt to reduce the burstiness of TCP. To minimize the effects of overmarking, AF-marking can be integrated into the congestion control mechanisms of TCP [12]. Instead of sending priority packets in place of non-priority packets, TCP can layer priority packet transmissions on top of its normal transmission of best-effort packets. Figure 6 shows the same scenario as shown in Figure 5 using a Constant Bit Rate marker integrated into TCP. As Figure 6(a) shows, the $3Mbs$ connection marks significantly less packets. In contrast to Figure 5(a), the $3Mbs$ connection is able to compete fairly for the best-effort bandwidth in the network. Finally, Figure 6(b) shows the window traces of the $3Mbs$ connection. In contrast to Figure 5(b), the $3Mbs$ connection is able to send priority packets in addition to non-priority packets rather than in place of non-priority packets.

4 Conclusion

This article has demonstrated several problems in the design and implementation of AF and services based on AF. In particular, it has shown that without strictly allocating buffer resources for AF packets, it is easy to construct a packet arrival scenario in which priority packets do not receive better performance than non-priority packets. While a strict allocation of resources can protect priority packets, such a scheme can be detrimental to the performance of non-priority packets. To address this, an adaptation to RED which strictly allocates resources to priority packets while performing RED on the remaining resources is described.

While accurately implementing AF in the network is one issue, this article has also described some of the problems with building predictable services on top of AF. In particular, it has shown that AF can interact poorly with the rate control mechanisms at the end hosts, causing performance to deteriorate. Without addressing these problems, it is unlikely that AF will provide end hosts any sort of predictable bandwidth guarantees. In order to improve the performance of AF, a number of mechanisms have been described which adapt better to the rate control being done at the end host with the AF-marking being done in the network.

References

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

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

[3] V. Jacobson, K. Nichols, and K. Poduri, "An Expedited Forwarding PHB," *Internet Draft draft-ietf-diffserv-phb-ef-01.txt*, November 1998.

[4] J. Heinanen, F. Baker, W. Weiss, and J. Wroclawski, "Assured Forwarding PHB Group," *Internet Draft draft-ietf-diffserv-af-04.txt*, January 1999.

[5] S. Floyd and V. Jacobson, "Random Early Detection Gateways for Congestion Avoidance," *ACM/IEEE Transactions on Networking*, vol. 1, no. 4, pp. 397–413, August 1993.

[6] R. Braden, D. Clark, J. Crowcroft, B. Davie, S. Deering, D. Estrin, S. Floyd, V. Jacobson, G. Minshall, C. Partridge, L. Peterson, K. Ramakrishnan, S. Shenker, J. Wroclawski, and L. Zhang, "Recommendations on Queue Management and Congestion Avoidance in the Internet," *RFC 2309*, April 1998.

[7] D. Clark and W. Fang, "Explicit Allocation of Best Effort Packet Delivery Service," *IEEE/ACM Transactions on Networking*, vol. 6, no. 4, , August 1998.

[8] W. Feng, D. Kandlur, D. Saha, and K. Shin, "Understanding TCP Dynamics in an Integrated Services Internet," in *Proc. of NOSSDAV '97*, May 1997.

[9] Cisco Systems Inc., "Advanced QoS Services for the Intelligent Internet," White Paper, July 1998.

[10] I. Cidon, R. Guerin, and A. Khamisy, "Protective Buffer Management Policies," *IEEE/ACM Transactions on Networking*, vol. 2, no. 3, , June 1994.

[11] R. Guerin, S. Kamat, V. Peris, and R. Rajan, "Scalable QoS Provision Through Buffer Management," in *Proceedings of ACM SIGCOMM*, September 1998.

[12] W. Feng, D. Kandlur, D. Saha, and K. Shin, "Adaptive Packet Marking for Providing Differentiated Services in the Internet," in *Proc. of ICNP '98*, October 1998.