Active Queue Management for Web Traffic

Mark Claypool, Robert Kinicki, Matthew Hartling
CS Department at Worcester Polytechnic Institute
Worcester, MA 01609, USA
{claypool,rek}@cs.wpi.edu

Abstract

The focus in Active Queue Management (AQM) research has been on traffic that is long-lived, insensitive to delay, and often has large TCP congestion windows. However, the majority of the flows on the Internet today are Web flows, which are short-lived, more sensitive to delay, and typically have small TCP congestion windows. In particular, short-lived flows with small congestion windows frequently suffer from timeouts when encountering packet loss. This paper presents and evaluates a new AQM scheme, SHort-lived flow friendly RED (SHRED), targeted at providing better network performance for short-lived Web traffic. Using an edge hint to indicate the congestion window size in each packet sent by the flow source or by an edge router, SHRED preferentially drops packets from short-lived Web flows less often than packets from long-lived flows. A wide-range of simulation results and analysis demonstrate that SHRED protects short-lived flows from unnecessary timeouts, improving the response time characteristics for Web traffic when the router becomes congested, while not degrading overall network performance.

1 Introduction

The continually growing World Wide Web is significantly changing the nature and characteristics of Internet traffic. An Internet traditionally made up of file transfer and email applications is being overwhelmed by newer, more-sophisticated and more-demanding Web-based applications that require a range of network services, from small object transfers such as advertisements and buttons, to retrieval of streamed multimedia. Whereas applications such as file transfers are concerned with throughput and packet loss rates, applications such as Web browsers that frequently request many short downloads care about response time and throughput.

Active Queue Management (AQM) research has focused on evaluating the benefits of congestion control algorithms by analyzing the performance of traffic mixes dominated by long-lived TCP flows. Most of the improvements and variations proposed for Random Early Detection (RED) [6] have been evaluated using predominately long-lived flows. However, [3] showed that RED provides little benefit when traffic consists solely of short-lived Web flows, and under highly congested circumstances RED can perform worse than Drop Tail. This paper proposes an AQM solution based on RED, but applicable to other AQMs, that seeks to improve Web traffic performance.

A typical Web page is composed of text, pictures, advertisements and other embedded Web objects. Often each Web object is transfered over a separate TCP connection. Even when there are multiple objects per TCP connection, most Web TCP connections tend to be short-lived flows [10]. Transmitting only a small number of packets, these short-lived flows spend most of their lifetime in TCP's slow-start phase with small TCP window sizes. In contrast, long-lived flows transmit a larger number of packets predominantly during TCP's congestion avoidance phase with larger TCP window sizes.

Small TCP window sizes can significantly affect short-lived flows in several ways [8]. First, a small TCP window yields a lower transmission rate. Second, short-lived flows are more sensitive to packet drops because small windows increase the likelihood that a packet drop will result in a retransmission timeout that adds significant delay to the transfer of a small object. Finally, TCP's fast retransmit is ineffective at preventing retransmission timeouts when a packet is dropped at the beginning and end of the transmission. In general, past work in Web flows and AQM [2, 17, 19, 20] does not support a continuum of treatments for short- to long-lived flows and/or requires per-flow state at the router.

This paper proposes a SHort-lived flow variant of RED, called *SHRED*, that provides preferential dropping for flows with small windows. Sources mark each packet with its current window size, allowing SHRED to drop packets from flows with small TCP windows with a lower probability. This reduces the negative effects of small windows sizes, protects short-lived flows from low transmission rates, and provides fairer bandwidth allocation among flows. Although SHRED is implemented based on RED, the core ideas in SHRED can be applied to most other AQM approaches.

We use NS-2 simulations in this investigation to compare the performance of SHRED, RED and Drop Tail under traffic scenarios that include Web traffic only, mixed Web and FTP traffic, and FTP only traffic. Additionally, we briefly discuss a Web traffic generator developed to facilitate properly analyzing AQMs with Web traffic.

Our analysis shows that: for Web only traffic, SHRED performs slightly better than Drop Tail for low to moderate congestion levels, whereas RED performs worse than Drop Tail; RED always performs better than Drop Tail for mixed traffic environments; and SHRED performs significantly better than RED and Drop Tail with mixed traffic and Web only traffic for moderate to high levels of congestion.

This paper is structured as follows: Section 2 describes the SHRED architecture and algorithm details; Section 3 describes the experiments designed to evaluate SHRED, RED and Drop Tail over a range of traffic conditions; Section 4 analyzes the performance of SHRED and compares it to RED and Drop Tail; Section 5 discusses some issues to address in deploying SHRED; and Section 6 summarizes our conclusions and presents possible future work.

2 SHRED

2.1 Short-lived Flow Issues

For both congestion and flow control, TCP uses a congestion window (*cwnd*) to limit a flow's sending rate. TCP initializes the value of *cwnd* to one at the beginning of the connection and, each round-trip time either doubles cwnd (during slow start) or increases cwnd by one (during congestion avoidance). However, since short-lived flows send only a few packets for each connection, they typically have small cwnds. Therefore, TCP flows with small cwnds receive a lower effective data rate than TCP flows with larger cwnds since the smaller window-sized flows require more round-trip times to send and acknowledge packets.

A second problem with short-lived TCP flows is that TCP's fast retransmit is ineffective when the cwnd is less than four. Fast retransmit requires the receipt of three duplicate acknowledgments (acks) by the TCP sender to trigger retransmission of a lost packet instead of waiting for a full retransmission timeout (RTO) period, often at least one second. TCP flows with a small cwnds are unable to send enough packets to generate three duplicate acks. Moreover, loss of the first packet (a SYN packet) costs small cwnd flows even more time since the initial retransmission timeout (ITO) is typically conservatively set to three seconds.

A third problem with short-lived TCP flows occurs at the end of a TCP connection. If one of the last three packets is dropped, the TCP source does not send enough additional packets to trigger three duplicate acks and an RTO occurs. Typically, the last three packets consist of the last two data packets and the TCP FIN packet. Therefore, for small Web objects all packet drops at a congested router will result in a costly RTO. Even for larger objects many dropped packets still result in an RTO. Thus, short-lived Web flows are more likely to suffer significant performance degradation when their packets are dropped than are long-lived flows.

2.2 SHRED Algorithm

The fundamental idea of SHRED is to use a lower drop probability for flows with small cwnds and have the drop probability increase linearly with a flow's increased relative cwnd size. As in the core-edge architecture used by Core Stateless Fair Queuing [18], an edge hint provides flow information, such as sending rate, to core routers to more efficiently manage congestion. Edge hints eliminate the need for per-flow state information in core routers. With SHRED, the sending host or the edge router inserts the current value of cwnd, a TCP state variable, into the IP packet header. The SHRED core router extracts the cwnd value ($cwnd_{sample}$) from the arriving packet and computes an exponentially weighted average cwnd ($cwnd_{avg}$) as shown:

$$cwnd_{avg} = (1 - w_c) cwnd_{avg} + (w_c) cwnd_{sample} \quad (1)$$

where w_c is set to 0.002, the same setting as originally recommended for RED's w_q. [1]

$cwnd_{avg}$ is biased towards flows that send more packets because $cwnd_{avg}$ is updated for each packet arrival. However, flows that send more packets have higher cwnds, and a higher $cwnd_{avg}$ results in an even lower drop probability for packets with smaller cwnds. This is the desired behavior.

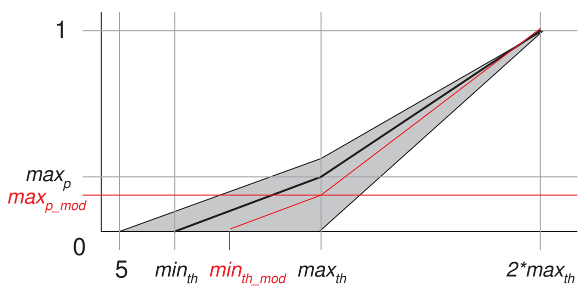


Figure 1: Computing Drop Probability in SHRED.

In SHRED, the packet drop probability is adjusted from that of gentle RED based on the ratio of $cwnd_{sample}$ to $cwnd_{avg}$. As depicted in Figure 1, based

[1] http://www.aciri.org/floyd/REDparameters.txt

on this cwnd ratio, SHRED modifies min_{th} and max_p such that the slope of the drop probability line remains the same for a given set of RED parameters:

$$min_{th-mod} = min_{th} + (max_{th} - min_{th}) \times (1 - \frac{cwnd_{sample}}{cwnd_{avg}}) \quad (2)$$

If the cwnd ratio is equal to 1, min_{th-mod} remains at min_{th}. If the cwnd ratio is greater than 1, min_{th-mod} is set lower than min_{th}. If the cwnd ratio is less than 1, min_{th-mod} is set to a value between min_{th} and max_{th}. Based on recommendations,[2] the lower bound for min_{th-mod} is set at five packets. The upper bound for min_{th-mod} is asymptotic to max_{th}.

Once min_{th-mod} is adjusted, max_p is modified so as to maintain the slope of the original RED drop probability line:

$$max_{p-mod} = max_p \times \frac{max_{th} - min_{th-mod}}{max_{th} - min_{th}} \quad (3)$$

Given min_{th-mod} and max_{p-mod}, SHRED computes the packet drop probability:

$$p_b = max_{p-mod} \times \frac{q_{avg} - min_{th-mod}}{max_{th} - min_{th-mod}} \quad (4)$$

with the remainder of the algorithm implementation identical to RED.

3 Experiments

This section discusses the experimental methodology used to evaluate SHRED, Drop Tail and RED.

3.1 Web Traffic Generation

Typically, Web traffic is simulated using one or more Web servers and a fixed number of Web clients. Web clients generate load on a router from empirically or mathematically derived parameters that may include: reply size, response size, think time, inter-object arrival time, and the number of objects per Web page.

When the number of Web clients stays fixed, increased router congestion causes increased Web response times. This, in turn, reduces the client's request rate which effectively decreases the load generated by individual clients and the total network load seen by the router. While this accurately mimics the behavior of individual Web clients, it does not necessarily provide an accurate network representation from the router's perspective. Many Web servers, especially those from a content distribution network, serve only part of a Web page and do not receive subsequent requests from the same client [10]. Clients often retrieve only one or two pages from a Web server before going to another Web site and so do not generate subsequent requests to the same server. Moreover, during flash crowds, the rate of users initially connecting is not influenced by the response time. Since existing ns-2 Web traffic models all decrease their offered load during congestion, we developed a Web traffic model that produces a fixed offered load through the router, independent of the level of congestion.

Our traffic generator sends Web pages, composed of multiple Web objects, to a traffic sink. We assume a model where each object in the Web page is a separate TCP connection.[3] The page generation rate is determined based on the previous page size and the aggregate load, in bits per second, specified during the simulation setup. The Web traffic generator sends all objects in a Web page and then waits for an amount of time determined by the page generation rate before sending the next page.

Since the model constructed for this research is a best case scenario in terms of Web page response time in that the primary object and all secondary objects are downloaded in parallel, actual response time differences due to AQM strategies (see Section 4) should be more dramatic than the simulation results presented here.

The Web traffic generated within ns-2 for this study uses the built in Pareto II function to generate Web reply object sizes. Based on previous findings in [15] and [7], 1.2 is used as the Pareto shape parameter with 10 KBytes as the average size parameter. The minimum and maximum object sizes are set to 12 Bytes and 2 MBytes, respectively. All the simulations used for the experiments use 1 object per page unless otherwise noted. Section 4.5 shows separate simulations with the number of objects set to 1, 1-8 objects, 1-16 objects, and 1-32 objects to evaluate the impact of number of objects per page on response time.

3.2 Metrics

We use both user-centric measurements that allow us to study evaluate the impact of Drop Tail, RED and SHRED on Web traffic and network-centric measurements that allow us to evaluate the impact of the same AQM techniques on overall network dynamics and long-lived traffic.

For user-centric measurements, we define object transmission time as the time taken to transfer a single Web object from a server to the client and Web response time as the time taken to download all objects in a Web page. While users certainly desire low response times,

[2] http://www.aciri.org/floyd/REDparameters.txt

[3] While HTTP/1.1 [16] enables multiple objects to be transferred over a single connection, in times of heavy server load, Web servers may close even persistent connections [14]. Moreover, [1] found nearly half of all TCP connections to a Web server use multiple connections. Lastly, many Web servers deliver only part of a Web page and so do not receive subsequent requests from the same client.

users also desire low object transmission times since this indirectly results in a low response times. Moreover, typical Web browsers can display part of a page when complete objects are downloaded even if the entire page is composed of multiple objects. Routers should try to minimize object transmission time, not only to provide better end-user performance, but also to reduce the number of flows through the router.

For network-centric measurements, we evaluate the percentage of packets dropped per flow for both Web and FTP traffic and goodput and Jain's Fairness index [12] for FTP traffic.

3.3 Experiment Setup

The network configuration used for all the experiments in this investigation is shown in Figure 2. The base RED parameters were set according to the latest recommended values,[4] and the Drop Tail queue size is the same as that for RED. Although ns-2 RED simulations often set queue thresholds based on packets, since many Web reply objects are smaller than a single packet, byte-based queue thresholds were used in this research. All the experiments were run for 160 seconds with measurements taken after the first 20 seconds for experiments using FTP traffic to allow long-lived flows to stabilize before collecting performance data. A few select tests were run to show that the results presented here are stable under longer durations, but shorter tests were used in light of the number of experiments required. For all experiments, both the FTP traffic and the Web traffic used TCP Reno.

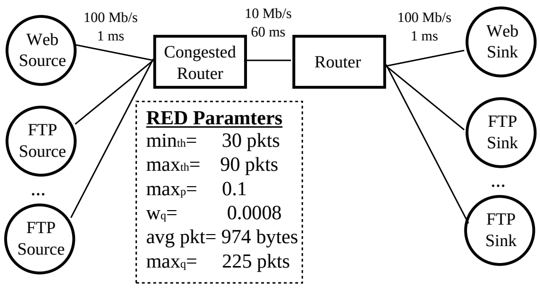
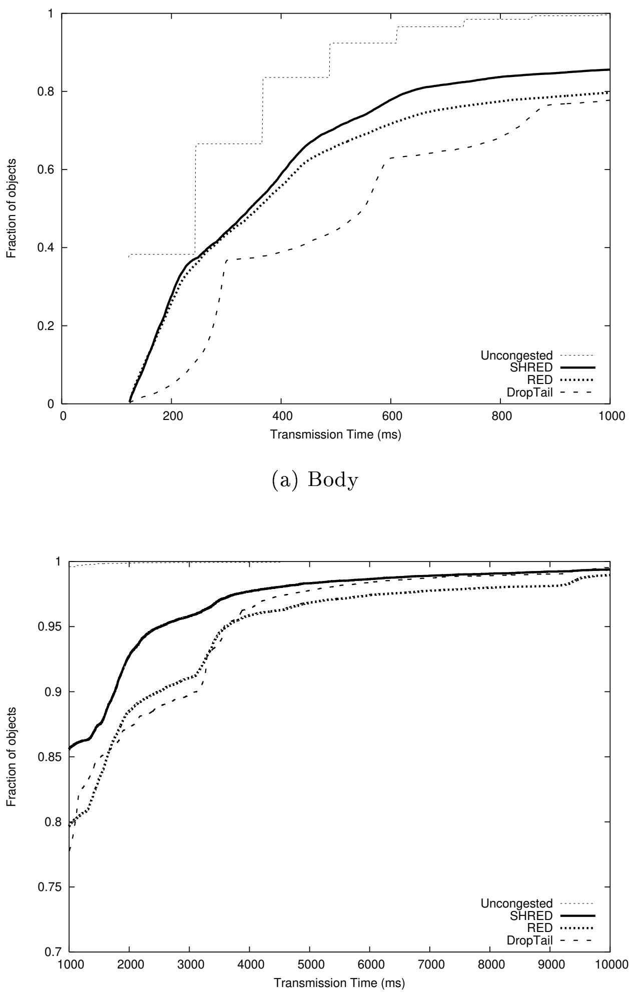


Figure 2: Network Configuration.

4 Analysis

The first set of *Web-only* experiments allows AQM performance evaluation similar to that in [3]. The second set of *Web-mixed* experiments analyzes performance with both Web and long-lived traffic. MCI backbone measurements [13] show that 75% of the flows and bytes are Web, and Sprint backbone measurements [4] show 90% of the flows send fewer than 20 packets. Thus, in the second set of experiments, the number of FTP flows is fixed at 10 while the Web-traffic load is varied between 40 and 80 percent of the bottleneck bandwidth (10 Mbps). The third set of *FTP-mixed* experiments fixes the Web traffic load at 50 percent and varies the number of FTP flows between 0 and 40. The last set of *FTP-only* experiments shows the impact of SHRED on only long-lived flows.

4.1 Web-only Experiments



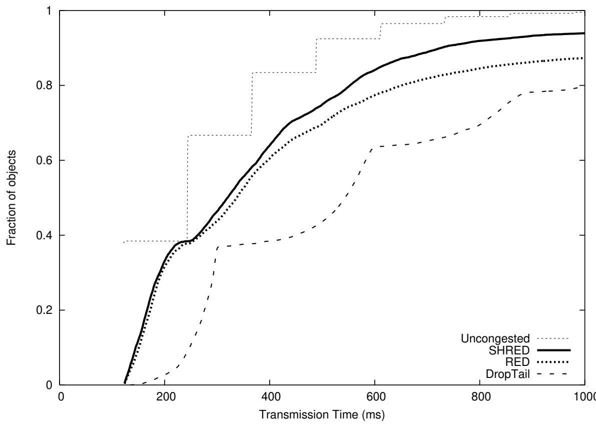
(a) Body

(b) Tail

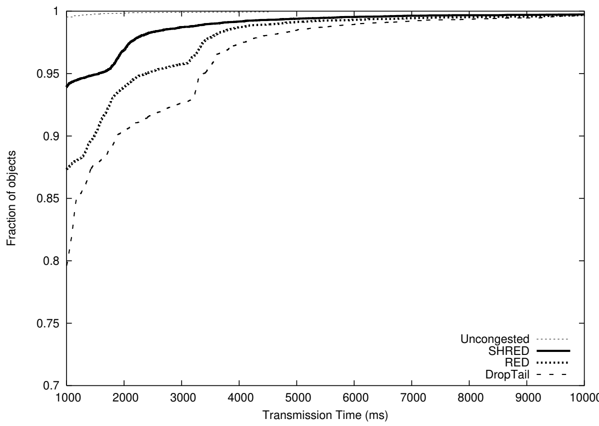
Figure 3: Transmission Time CDF for 100 percent load (average of 140 active flows)

To provide an upper bound on Web traffic performance, we obtained object transmission times for a simulated uncongested router of capacity 100 Mbps that

[4]See http://www.aciri.org/floyd/REDparameters.txt and [5]. Although the *gentle* setting is recommended for RED routers, it was not a factor in our experiments since average queue sizes remained below the maximum threshold.



(a) Body



(b) Tail

Figure 4: Transmission Time CDF for 70 percent Web load (average of 84 active flows) and 10 FTP flows

experienced little queuing delay and no packet drops. The cumulative density function of transfer times for the uncongested router has "steps" in Figure 3 due to an initial window size of 1 and the TCP slow-start phase. Specifically, an object small enough to fit in a single packet is transmitted in one round-trip time (RTT); an object as large as two or three packets is transmitted in two RTTs; and an object as large as four to seven packets is transmitted in three RTTs. This effect continues throughout the slow-start phase.

In Figure 3, at 100 percent Web-only load (95% and 105% Web loads are given in [9]), Drop Tail begins to degrade as the queue is frequently full and the queuing delay for all packets becomes large. With a queue size of 225 packets, a packet takes approximately 175 ms to traverse the queue. Adding this queuing delay to the 120 ms RTT, objects consisting of a single packet take roughly 300 ms to transfer; objects of two or three packets in length take 600 ms to transfer; and objects between 4 and 7 packets in length take 900 ms to transfer. Furthermore, packet drop bursts induced by Drop Tail make it difficult for the small-windowed TCP flows to employ fast retransmit. At 100 percent load, RED performs slightly better than Drop Tail, while SHRED has the best performance.

4.2 Web-mixed Experiments

Figure 4 shows the performance of SHRED for the Web-mixed experiments with 70% Web load and 10 FTP flows (Web loads of 60% and 80% are given in [9]). SHRED consistently provides a substantial performance improvement over both Drop Tail and RED. The SHRED CDF is much closer to the uncongested router CDF than either RED or Drop Tail. Analysis of the CDF tails suggest that at least 5 to 10 percent of RED's performance degradation is due to RTOs (between 1 and 2 seconds) and an additional 2 to 5 percent degradation is from ITOs. Drop Tail suffers a performance degradation of at least 5 to 8 percent due to RTOs and an additional 4 to 10 percent degradation from ITOs. SHRED largely avoids performance degradation due to ITOs and is less affected by RTOs than both RED and Drop Tail. Lastly, contrary to results in Section 4.1 and [3], RED, running a more realistic mixture of short-lived and long-lived flows, performs consistently better than Drop Tail for all loads.

Figure 5 summarizes the SHRED performance benefit over RED and Drop Tail for the mixed traffic experiments. SHRED provides a 12 to 17 percent performance improvement over Drop Tail and this advantage increases slightly as the Web traffic load increases. The RED improvement over Drop Tail remains fairly constant around 8 percent regardless of the load.

Figure 6 shows the percent drops for nine of the Web-mixed experiments. Each bar in the graph separates out the Web traffic drops from FTP traffic drops for a given experiment. For each three column group, the first column is the SHRED drop percentage while the second and third column are drop percentages for RED and Drop Tail, respectively. SHRED drops fewer Web packets than either RED or Drop Tail and drops packets more proportionally to the actual traffic mix. SHRED yields the fewest drops at all three load levels.

4.3 FTP-mixed Experiments

The results from the FTP-mixed experiments are similar to the results from the Web-mixed experiments. Due to space constraints, we only summarize these experiments, with more results in [9]. Figure 7 summarizes the performance benefit of RED and SHRED over Drop Tail for the FTP-mixed experiments for 10 to 50 FTP flows and a Web load of 50%.

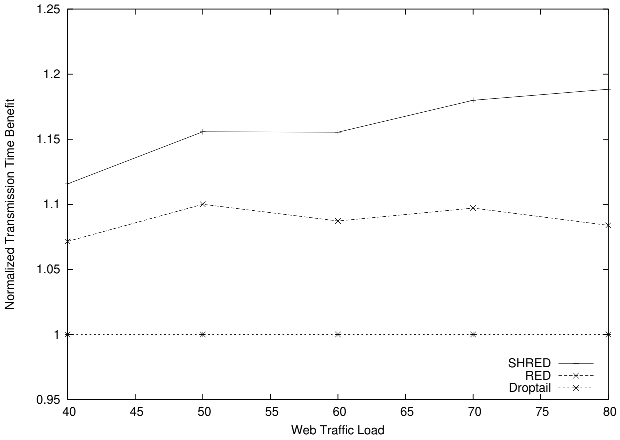


Figure 5: Normalized Transmission Time Benefit for Web-mixed

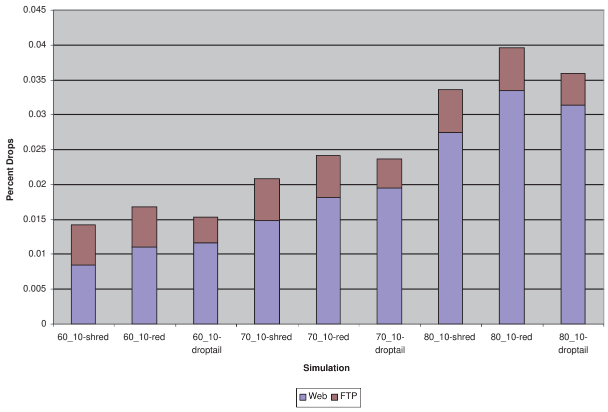


Figure 6: Percent Drops for Web-mixed

SHRED provides an increase in benefit versus Drop Tail with an increase in FTP load and SHRED performs significantly better than RED at higher levels of congestion. While RED performs consistently better than Drop Tail, the benefit of SHRED to Web traffic widens with an increase in the number of flows.

4.4 FTP-only Experiments

Although SHRED does not seek to improve performance of long-lived flows, it should not have a negative impact versus either Drop Tail or RED in the presence of only long-lived traffic. We ran experiments with 10 to 100 FTP flows (no Web flows) under RED, SHRED and Drop Tail, and compute the total goodput and Jain's fairness for all flows. Table 1 shows that SHRED's performance is similar to Drop Tail and slightly better than RED for most of the experiments. Table 2 shows that SHRED yields a slightly higher Jain's Fairness value than both RED and Drop Tail by providing a more uniform window size for all flows.

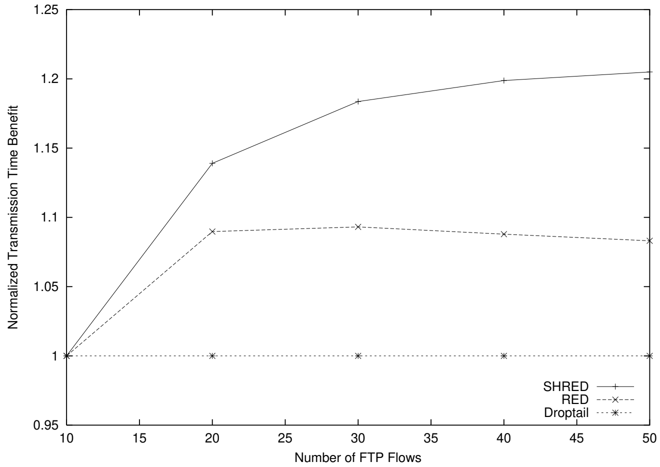


Figure 7: Normalized Transmission Time Benefit for FTP-mixed

4.5 Web Response Times

Since a Web page is not entirely usable until all its objects have been downloaded, Web response time is a more useful measure of performance for users than transmission time. While [7] consider between 1-3 and 2-7 objects per page, recent measurements [14] imply that today's Web pages have considerably more objects per page. Thus, we re-ran the Web-only and Web-mixed experiments with a uniform random distribution of 1 to 8, 1 to 16, and 1 to 32 objects per page.

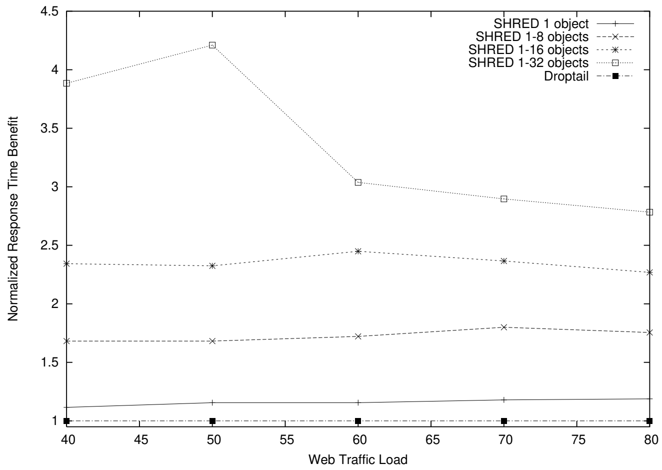


Figure 8: Web Response Time Benefit for Web-mixed

Figure 8 summarizes the response time benefit of SHRED over Drop Tail for the Web-mixed experiments for 40% to 80% Web load and 10 FTP flows. Since our Web traffic generator transmits all objects concurrently,

Flows	SHRED	RED	Drop Tail
10	9.4	9.6	10.0
20	9.7	9.7	10.0
30	9.8	9.8	9.9
40	9.7	9.7	9.9
50	9.7	9.7	9.8
60	9.6	9.6	9.8
70	9.5	9.6	9.7
80	9.5	9.3	9.6
90	9.4	8.7	9.6
100	9.3	8.4	9.5

Table 1: Goodput (Mbps)

Flows	SHRED	RED	Drop Tail
10	1.000	0.997	0.991
20	0.999	0.996	0.990
30	0.999	0.997	0.994
40	0.999	0.997	0.989
50	0.999	0.996	0.991
60	0.999	0.995	0.993
70	0.998	0.994	0.988
80	0.998	0.993	0.990
90	0.999	0.989	0.989
100	0.998	0.988	0.990

Table 2: Jain's Fairness

Drop Tail routers see traffic as a burst of packets, consisting of a large number of objects belonging to a page, that are more likely to overflow the router queue. Unlike the Web-only results, the response time benefit increases approximately linearly as the number of objects increases and remains constant over all loads except for the experiments with 1-32 objects per page. SHRED provides over 60 percent response time benefit over Drop Tail for Web pages with 1-8 objects and over 140 percent response time benefit over Drop Tail for pages with 1-16 objects. For pages with 1-32 objects and loads of 40% or 50%, SHRED provides response time benefits greater than 250 percent. For higher loads, the SHRED response time benefit reduce somewhat to between 175 to 200 percent.

5 Discussion of Deployment

The benefits of SHRED can be realized by incorporating SHRED into the next generation of Internet routers. This section discuss practical deployment issues facing SHRED.

The cwnd edge hint can be implemented in IPv4 using the TOS byte in the IP header. The TCP source maintains cwnd in bytes, but can map it to a value in packets using TCP's maximum segment size (or any mapping where a value of 1 yields the lowest drop probability and a value of 255 yields the highest). Using the TOS byte gives 255 possible values for cwnd. For IPv6, the cwnd edge hint can be implemented using an extended header.

SHRED supports incremental deployment. Only users and Internet Service Providers (ISPs) seeking improved Web performance need to upgrade. If the TCP source does not support SHRED, the SHRED router uses normal RED dropping probabilities. Although there far more TCP sources than routers, source operating systems are easier to update since popular operating systems issue updates multiple times a year. An alternative to upgrading TCP in all end-host operating systems would be to place a packet classifier at an edge router that keeps per flow TCP state and inserts cwnd hints into each packet.

Unfortunately, using cwnd hints provides opportunities for misbehaving hosts to deliberately keep their cwnd hint low to get reduced drop rates from SHRED routers. Stagnant cwnd hints could be policed by the first downstream router from the source keeping per flow TCP state since all packets and TCP acknowledgments for the source must pass through that router.

Another possible source of misbehaving is between an ISP and a backbone provider, where the ISP could set low cwnd hints such that all packets receive a lower dropping probability, yielding better overall performance for network traffic for the ISP. However, as backbone router's are typically over-provisioned to avoid packet drops [11], such false hints would not significantly affect performance.

6 Conclusions

This paper presents SHort-lived flow friendly RED (SHRED), an AQM mechanism designed to improve response time for short-lived Web traffic. Using a cwnd hint from a TCP source, SHRED computes the cwnd ratio of an arriving packet to the cwnd average and reduces the probability of dropping packets during the sensitive period when a flow's cwnd is small. We ran extensive simulation experiments modeling Web-only traffic, FTP-only traffic, and mixes of Web and FTP traffic to evaluate the performance of SHRED versus RED and Drop Tail routers.

Our analysis shows that SHRED produces lower object transmission times than either RED or Drop Tail in both Web-only and mixed traffic environments. SHRED provides over 40 percent improvement in Web response times over Drop Tail for Web-only traffic and between 70 and 300 percent improvement in Web response times over Drop Tail. Moreover the Web traffic performance improvements are achieved without negatively impacting long-lived FTP traffic.

In contrast to results reported in [3] that show little improvement for RED over Drop Tail in Web-only traffic experiments, the mixed traffic simulations here indicate that RED consistently outperforms Drop Tail.

Although SHRED is based on RED, the core principal of SHRED (a cwnd edge hint combined with drop preference by a router) can be applied to most other AQM approaches. One possible extension to SHRED is to incorporate an indicator in the cwnd hint when a flow is nearly finished and to avoid dropping the last few packets of a transfer.

References

[1] M. Allman. A Web Server's View of the Transport Layer. *ACM Computer Communication Review*, 30(4), Oct. 2000.

[2] Carey Williamson and Qian Wu. A Case for Context-Aware TCP/IP. *ACM Performance Evaluation Review*, 29(4):11 – 23, Mar. 2002.

[3] M. Christiansen, K. Jeffay, D. Ott, and F. Smith. Tuning RED for Web Traffic. In *Proceedings of ACM SIGCOMM Conference*, Aug. 2000.

[4] C. Diot, G. Iannaccone, and M. May. Aggregate Traffic Performance with Active Queue Management and Drop from Tail. Technical Report TR01-ATL-012501, Sprint ATL, Jan. 2001.

[5] S. Floyd, R. Gummadi, and S. Shenker. Adaptive RED: An Algorithm for Increasing the Robustness of RED, 2001.

[6] S. Floyd and V. Jacobson. Random Early Detection Gateways for Congestion Avoidance. *IEEE/ACM Transactions on Networking*, Aug. 1993.

[7] L. Guo and I. Matta. The War Between Mice and Elephants. In *Proceedings of the 9th IEEE International Conference on Network Protocols*, Nov. 2001.

[8] J. Hall, I. Pratt, I. Leslie, and A. Moore. The Effect of Early Packet Loss on Web Page Download Times. In *The Passive and Active Measurement Workshop*, Apr. 2003.

[9] M. Hartling, M. Claypool, and R. Kinicki. Active Queue Management for Web Traffic. Technical Report WPI-CS-TR-02-20, CS Department, Worcester Polytechnic Institute, May 2002.

[10] F. Hernandez-Campos, K. Jeffay, and F. Smith. Tracing the Evolution of the Web Traffic: 1995-2003. In *Proceedings of the 11th IEEE/ACM International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems (MASCOTS)*, Oct. 2003.

[11] G. Iannaccone, M. May, and C. Diot. Aggregate Traffic Performance with Active Queue Management and Drop from Tail. *ACM Computer Communication Review*, July 2001.

[12] R. Jain. *The Art of Computer Systems Performance Analysis: Techniques for Experimental Design, Measurement, Simulation, and Modeling.* John Wiley and Sons, Inc., 1991.

[13] k claffy, G. Miller, and K. Thompson. the nature of the beast: recent traffic measurements from an internet backbone. In *Proceedings of ISOC INET '98*, July 1998.

[14] B. Krishnamurthy and C. E. Wills. Analyzing Factors that Influence End-to-End Web Performance. *WWW9 / Computer Networks*, 33(1-6):17–32, 2000.

[15] B. A. Mah. An Empirical Model of HTTP Network Traffic. In *Proceedings of INFOCOM*, pages 592–600, Apr. 1997.

[16] R.Fielding, H. Frystyck, and T. Berners-Lee. Hypertext transfer protocol – http/1.1. *IETF Request for Comments (RFC) 2616*, June 1999.

[17] Shanchieh Jay Yang and Gustavo de Veciana. Size-based Adaptive Bandwidth Allocation: Optimizing the Average QoS for Elastic Flows. In *Proceedings of IEEE INFOCOM*, June 2002.

[18] I. Stoica, S. Shenker, and H. Zhang. *Core*-Stateless Fair Queueing: Achieving Approximately Fair Bandwidth Allocations in High Speed Networks. In *Proceedings of ACM SIGCOMM Conference*, Sept. 1998.

[19] Wael Noureddine and Fouad Tobagi. Improving the performance of interactive tcp applications using service differentiation. In *Proceedings of IEEE INFOCOM*, June 2002.

[20] X. Wu and I. Nikolaidis. On the Advantages of Lifetime and RTT Classification Schemes for TCP Flows. In *Proceedings of IEEE 2003 International Performance, Computing, and Communications Conference (IPCCC)*, Apr. 2003.