International Journal of Networks and Communications 2012, 2(5): 62-74
DOI: 10.5923/j.ijnc.20120205.01

Easing Congestion in Computer Networks using the Receiver-Window Modification (RWM) Scheme

Visvasuresh Victor Govindaswamy[1,*], Gergely Záruba[2], G. Balasekaran[3]

[1]College of Science, Technology, Engineering, and Mathematics, Texas A&M University at Texarkana, Texas, 75503, United States
[2]College of Computer Science and Engineering, The University of Texas at Arlington, Texas, 76019, United States
[3]Human Bioenergetics Laboratory, Nanyang Technology University, Singapore, 639798, Singapore

Abstract Random Early Detection (RED), Adaptive RED (ARED), BLUE and other Active Queue Management (AQM) queues have been proposed to replace drop-tail queues for Transmission Control Protocol/Internet Protocol (TCP/IP) networks to ease network congestion. Together with the Explicit Congestion Notification (ECN) scheme, they had shown some promise over the drop tail queues. However, the timeout mechanism or the duration of the reception of three duplicate acknowledgements (ACKs), due to early-dropped packets, delays the response time of TCP in reducing the network congestion. Moreover, using ECN has its downsides: i) its messages may get lost; and ii) TCP implementations at the source, router and the destination have to be ECN-compliant (which presents a significant problem in today's implementations). To minimize these problems, this paper presents a novel AQM modification scheme called Receiver-Window Modification (RWM), which can be used together with any AQM queue for congestion avoidance in packet switched networks, especially at ingress and gateway routers. RWM improves on the average queue sizes, one-way end-to-end packet delays, delay jitter, throughput and number of dropped packets. It also overcomes the dependency of these AQM queues on the queues of downstream routers since ECN messages may get dropped or delayed due to congestion at these routers. We carry out extensive NS2 simulations to show our results and to support our claims.

Keywords TCP, Active Queue Management (AQM), Buffer Management, Random Early Detection (RED), Congestion Control, Congestion Avoidance

1. Introduction

TCP (Transmission Control Protocol) is the major connection oriented *transport layer protocol* used in today's Internet[1] and is tailored to provide reliability and transmission rate control to applications using flow and congestion controls. Despite these controls, congestion still occurs and these controls lead to busty Internet traffic. In the presence of bursty Internet traffic, routers along the path may buffer packets in their queues to absorb/reduce the burstiness. However, these may often lead to buffer overloads, i.e. congestion, at these routers along the transmission path. If the buffer of a router is saturated then the router may drop packets determined by its congestion control policy. TCP's congestion control can introduce a high variance resulting in a high delay jitter in the current buffer sizes. On the other hand, if the maximum buffer size of routers is increased, the TCP senders will be notified about congestion much later.

* Corresponding author:
lovebat814@yahoo.com (Visvasuresh Victor Govindaswamy)
Published online at http://journal.sapub.org/ijnc

This will lead to an increase in the flow's end-to-end delay. This delay is the sum of two components: i) the transmission, propagation and processing delays which can be considered fixed delays; ii) the queuing delays at the routers which are variable. Hence, the greater the current queue size, the greater the queuing delay will be, increasing the end-to-end packet delay. On the other hand, if the queue size is too small, packets will be dropped regularly due to buffer overflows. Hence, there is a need for solutions to determine an optimal or sub-optimal queue sizes.

The research area dealing with the above phenomenon is called TCP Queue Management (QM). The aim of QM is to minimize the congestion at the queue. QM approaches can be divided into two categories: i) passive queue management algorithms, where the queue of the routers needs to fill up before corrective action is taken; and ii) active queue management (AQM)[2], where preventive action is taken before the queue is saturated. AQM algorithms can again be subdivided into:

- Load-based AQM approaches, including Random Exponential Marking (REM)[3], Adaptive Virtual Queue (AVQ)[4] and PI controller[5] algorithms. Load based approaches are out of the focus of this paper. They are merely mentioned for completeness of discussion.

•Queue-based AQM, including Random Early Detection (RED)[6], Stabilized RED (SRED)[7], Flow RED (FRED)[8], Adaptive RED (ARED)[9], and BLUE[10].

Although these AQMs have, overall, reduced the congestion, the reduction is not much in most cases. This paper deals with this problem, presenting a novel approach called Receiver Window Modification (RWM) for congestion avoidance in TCP/IP networks, thereby aiding existing AQMs to reduce the congestion levels further. In AQMs, such as RED, ARED and BLUE, the timeout mechanism or the duration of the reception of three duplicate acknowledgements (ACKs), due to early-dropped packets, delays the response time of the TCP congestion scheme in reducing the network congestion. To reduce the timeouts due to these early-dropped packets, Explicit Congestion Notification (ECN)[11][12] was proposed. ECN is used to notify the sender TCP process about congestion, before it actually happen thus preventing unnecessary packet drops. However, using ECN has its downsides: i) its messages may get lost; and ii) TCP implementations at both the source and the destination have to be ECN-compliant which presents a significant problem in today's implementations. To minimize these problems, several schemes, such as[13], [14],[15] and[16] have been proposed. In[13], a hop-by-hop rate-based congestion control scheme has been proposed that matches the sending rate of a connection to the service rate observed at the downstream node. Each network node needs to measure and collect buffer occupancies for each connection per out-going link and feedback these information periodically to all the nearest upstream network nodes sending traffic via that link. In[14], the authors have proposed eXplicit Control Protocol, XCP, to replace TCP in the network. XCP-compliant routers inform the senders about the degree of congestion by putting control state within XCP-compliant packets. In[15], an end-to-end rate-based flow control scheme (called EXACT) has been proposed, whereby a flow's allowed rate is explicitly conveyed from intermediate routers to the end-hosts within each data packet's special control header. In a similar manner,[16] has developed a proportionally-fair congestion control algorithms (both hop-by-hop as well as end-to-end) with the MAC constraint being imposed in the form of a channel access time constraint. All these schemes are too costly since they need the replacement of routers, senders and hosts along the path the packet travels through, which presents a significant problem in today's implementations. Eventually, sooner or later these schemes require the replacement of the entire present network. This is not possible since it will be too costly to implement such a drastic change. Our argument is to improve on the present day infrastructure by minimizing the cost. By simply changing the ingress and gateway routers which are the two main areas of congestion[17], we can drastically reduce the congestion levels within the network.

Our scheme, RWM, which can be used together with RED, ARED, BLUE or other AQM queues, is used for congestion avoidance in packet switched networks, especially at the ingress and gateway routers. RWM works with RED, ARED and BLUE algorithms together relaxing their need for a modified TCP layer at servers and clients since it does not require modification to TCP implementations at servers or clients, i.e., no "RWM-compliancy" is needed. Our approach is specifically designed for ingress and gateway routers (see item 3. of Section 5 with regards to core routers) which are areas of heavy congestion. The main idea of our approach is to restrict the TCP transmission window with the flow control window instead of the congestion control window, thus controlling the transmission window with a finer granularity. There is no need to change the entire network or the entire path from the sender to the receiver. Results from simulations show that the RWM modified queues improve on the average queue size, the one-way packet delays, number of dropped packets and throughput as compared to RED, RED-ECN, ARED, ARED-ECN, BLUE and BLUE-ECN queues, especially in paths that have non-ECN compliant routers which is a true reflection of present-day networks. Since the number of packet drops is reduced, the number of timeouts or the duration of the reception of three duplicate acknowledgements (ACKs) is reduced or avoided leading to reduced packet delay. A summary version of this paper has been published as a conference proceeding[18]. We had also performed mathematical modeling on the RWM scheme[19].

The rest of the paper is organized as follows: Section 2 briefly covers the necessary background information and previous work. In Section 3, the RWM scheme is introduced; simulation details, results as well as analysis are covered in Section 4. The paper is concluded and future research directions are outlined in Section 5.

2. Background Information

This section gives an overview of the various major topics involved or related in AQM research. Sections 2.1, 2.2, 2.3 and 2.4 provide brief insight into TCP, RED, ARED and BLUE AQMs respectively while section 2.5 provides an overview of the ECN mechanism. Section 2.6 concludes Section 2.

2.1. Transmission Control Protocol

Transmission Control Protocol (TCP) is the commonly used protocol which, along with the Internet Protocol (IP), sends data in the form of packets between computers over the internet. In common TCP implementations congestion control employs indirect feedback[20], i.e., the transmitting node will assume that a packet was dropped due to congestion if no acknowledgment has been received within a time period or negative acknowledgment (ACK) has been received in the form of duplicate positive ACKs. The transmission rate of the TCP sender is then drastically reduced to help the routers to recover. More precisely, in the present network, TCP congestion control consists of two main phases called *Congestion Avoidance* (CA) and *Slow Start* (SS)[21]. The CA algorithm slows down the

transmission rate of segments, by reducing the allowable number of pending or unacknowledged segments on the network. Here, the congestion window that determines the number of pending packets will be halved. To increase the transmission rate the SS algorithm is then invoked to exponentially and after reaching a threshold, linearly increase the load on the network until congestion occurs again, i.e., a packet is dropped again. This drop is detected again by TCP either in a form of a timeout by the transmit timer or by the reception of three duplicate ACKs. This unique combined behavior of the CA and SS algorithms may lead to what is known as the *Global Synchronization Problem*[23]. This problem occurs whenever multiple flows lose packets at the bottleneck router, leading to the flows restarting concurrently, resulting in reduced aggregate throughput. It has also been shown that the sudden changes in the TCP transmission window caused by the interaction of SS and CA create a chaotic process introducing self-similarity in the traffic[23]. Present day, queue management solutions, such as RED[24], use more sophisticated solutions in the queues of routers to determine which packets to drop thus trying to compensate for the Global Synchronization Problem. We give a brief description of applying techniques in the rest of this remaining section.

2.2. Random Early Detection

Random Early Detection (RED)[24] prevents global synchronization by sacrificing a specific data flow whenever the average queue size increases to a point where the overflow may soon occur. It picks a flow fairly and discards a packet, forcing the flow into congestion avoidance and hence, decreasing its aggregate transmission rate. The probability that the gateway chooses a particular flow to mark or drop during congestion is roughly proportional to that flow's share of the bandwidth at the router. In short, it is valid to say that if a flow has a large fraction of the recently dropped packets, then it has also most likely received a large portion of the recent bandwidth. RED also avoids biasness against bursty traffic.

There are two parts to the RED algorithm; the estimation of average queue size and the decision on which packet to drop (see pseudo-codes in Figures 1 and 2 respectively). In Figure 1, the RED gateway uses an exponential weighted moving average low-pass filter to calculate the average queue size. The weight w_q determines the time constant of the low-pass filter and is set to 0:001 as in[24]. The purpose of using a low-pass filter is to make the network more capable of accommodating bursty traffic rather than shaping bursty traffic to accommodate the needs of the network. RED also uses FIFO scheduling due to FIFO's: i) low overhead, ii) lack of scaling problems, and iii) reduction of weight of the tail of the delay distribution. As shown in Figure 2, RED has two thresholds, the minimum (MIN_{th}) and maximum (MAX_{th}). If the average queue size of the queue is smaller than MIN_{th}, then the arriving segment is accepted. If the average queue size is between MIN_{th} and MAX_{th}, then the segment is dropped with packet-marking probability P_a. P_a is a linear function of the average queue size, possibly increasing to a pre-determined maximum value P_{max}. As average queue size varies from MIN_{th} to MAX_{th}, the P_a varies linearly from 0 to P_{max} with $P_a = P_{max}(avg - MIN_{th})/(MAX_{th} - MIN_{th})$. However, if the calculated average queue size is larger than MAX_{th}, then incoming segments are dropped with probability one. RED's behaviour results in a larger virtual queue size for short-term bursts compared to long-term bursts. However, despite the above-mentioned advantages, there are some well-documented problems with RED[7],[9],[25],[26]. The level of congestion with its immediate links (neighbours), compounded with its complex parameter tuning[26] induces large variations in the queue size. The queue size is near MIN_{th}, whenever the link is lightly congested or the P_{max} is high. However, the queue size increases to around MAX_{th} with an increase in the congestion levels or if P_{max} is set to a low value. Both these situations result in a degraded throughput, i.e., RED may underperform even when compared to the Drop Tail mechanism[20]. RED may also introduce jitter into non-bursty streams. All these disadvantages have generated more research using RED as a starting point for further improvements[3],[9],[10].

For each packet arrival
If the queue is not empty
$avg = (1- w_q)\ avg + w_q q$
Else using a table lookup
$avg = (1- w_q)^{(time-q\_time)/s}\ avg$
Where
avg: average queue size
w_q: time constant
q: current queue size
s: typical transmission time
time: current time
q_time: start of the queue idle time

Figure 1. Average queue size estimation algorithm

For each packet arrival
Calculate avg
If $MIN_{th} \leq avg < MAX_{th}$
Calculate P_a
Mark/drop the arriving packet with P_a
Else if $MAX_{th} < avg$
Drop the arriving packet
Else
Accept the arriving packet
Where
avg: average queue size
MIN_{th}: minimum queue threshold
MAX_{th}: maximum queue threshold
P_a: marking/dropping probability

Figure 2. Dropped packet selection algorithm

There are a number of variations of RED, such as Adaptive RED (ARED)[9], Random Exponential Marking (REM)[3], BLUE[10], Balanced RED[27] and RED with Preferential Dropping[28]. We are going to briefly outline ARED and BLUE in subsequent sections as they are the most advanced AQM schemes available today.

2.3. Adaptive Random Early Detection (ARED)

Adaptive RED (ARED)[9] is one of the many versions of RED with modifications to the original Adaptive RED proposal[26]. It is geared to achieve a predefined target queue length and hence lower packet loss and minimum variance in queuing delay. More importantly, it aims to solve the complex parameter setting problem in RED.

In ARED, P_{max} is not constant but is changed (adapted) at 0.5s intervals to keep the average queue around (MIN_{th} + MAX_{th})/2. The pseudo code of P_{max} adaptation is shown in Figure 3 which uses an additive increase multiplicative decrease (AIMD) technique for adapting P_{max}.

Execute the following at every interval of 0.5 seconds:
If (avg > target and $P_{max} < 0.5$)
$P_{max} = P_{max} + \alpha$
Else if (avg < target and $P_{max} > 0.01$)
$P_{max} = P_{max} * \beta$
Where
avg: average queue size
target: desired queue size
P_{max}: maximum probability
α: min(0.01, P_{max}/4)
β: 0.9

Figure 3. Algorithm to adapt P_{max}

2.4. BLUE

The BLUE active queue management algorithm[1] uses packet loss and link utilization history to manage congestion by detecting and adjusting the rate of packets being dropped or marked. It uses the probability, P_a, to mark by using ECN or to drop queued packets. P_a is increased whenever packets are dropped from the queue and decreased when the link is underutilized.

Upon packet loss:
If ((now-last_update) > freeze_time) then
$P_a := P_a + \delta 1$
Last_update = now
Upon link idle:
If ((now-last_update) > freeze_time) then
$P_a := P_a - \delta 2$
Last_update = now
Where
P_a: marking/dropping probability
$\delta 1$: amount of increase by P_a
$\delta 2$: amount of decrease by P_a
now: current time
last_update: last time P_a was changed
freeze_time: amount of time for P_a to take effect

Figure 4. BLUE algorithm

The amount of increase by P_a is denoted by δ_1 while δ_2 represents its amount of decrease. This update on P_a takes place, whenever the queue length exceeds a certain threshold *L*, at the rate of 1/*freeze_time*. The parameter *freeze_time* represent the time interval between successive updates on P_a. Figure 4 shows a sample pseudo code of the BLUE algorithm.

2.5. Explicit Congestion Notification

In Explicit Congestion Notification (ECN) scheme [12][13], whenever the average queue size is between MIN_{th} and MAX_{th}, RED is modified by an ECN-compliant router to marking instead of dropping packets. An ECN-compliant TCP source, on receiving the marked ACK packets echoed from an ECN-compliant TCP receiver, will behave in a manner similar to as if it had encountered a dropped packet. The response of TCP source to the congestion should happen at most once per round trip time (RTT).

A TCP sender reacts to an ECN flagged segment by halving both the congestion window *cwnd* and the slow-start threshold *ssthresh* at most once per RTT, while ignoring succeeding ECNs within that particular RTT. This is to ensure that the TCP source does not reduce its window repeatedly within that particular RTT. However, when the ECN-compliant TCP sender receives three duplicate ACKs without any ECNs, it will undergo Fast Retransmit and Fast Recovery procedures (see TCP Reno[20]). On the other hand, if the ECN-compliant TCP sender receives three duplicate ACKs after a reaction to an ECN notification within the RTT, it will not react since it has already reacted to the ECN notification.

A typical sequence of events is as follows. After positive negotiation between the sender and the receiver about using ECN, the sender sets the ECN-Capable Transport (ECT) codepoint in the packet's IP header. When an ECN-capable router detects congestion, it selects a packet. If the packet had its ECT codepoint set, instead of dropping it, the router sets the packet's Congestion Experienced (CE) codepoint in its IP header, to indicate to its sender of the impending congestion. On receiving this packet, the TCP receiver sets the ECN-Echo (ECE) flag in its next ACK packet to the TCP sender. When the TCP sender receives this ACK packet, it will react as if it had encountered a packet dropped. It will also set the Congestion Window Reduced (CWR) flag in the TCP header of the next packet it sends to the receiver, acknowledging the reception of the ACK with its ECE flag set and that it had reacted to the impending congestion.

2.6. Conclusions

AQM solutions use packet drops at router queues to manipulate the TCP sender into decreasing its transmission rate. RED, ARED and BLUE queues prevent a queue in any intervening router from reaching its limit by dropping packets early. With ECN-compliant RED, ARED or BLUE queues, packets are not unnecessarily dropped but marked as if they were dropped. We denote these types of queues as RED-ECN, ARED-ECN and BLUE-ECN respectively.

Using ECN, a queue accelerates the detection of congestion by the source, without having to wait to detect a dropped packet in the form of a timeout from the transmit timer or on receiving three duplicate ACKs. However, as a downside: i) ECN messages may get lost; and ii) TCP

implementations at the router, source and destination have to be ECN-compliant which presents a significant problem in today's implementations. Currently there is no practical benefit in setting ECN bits in Internet packets; as this requires ECN capable routers, at least at bottleneck points such as ingress and gateway routers, servers and clients throughout a network. In[29], experiments were conducted using TBIT (the TCP Behaviour Inference Tool) showing that in Sept. 13, 2000 results, 21 out of 26447 (0.07%) sites positively responded with an appropriate SYN/ACK. In March 2002, only 7 out of 12364 sites (0.05%) responded positively. These tests included nearly all types of operating systems, including Windows and Linux, and they showed the unpopularity of ECN. It is very difficult to force end-users to switch to ECN-compliant machines and it is only sensible to change the gateway and ingress routers which can be easily done by network administrators.

In the next section, we will introduce our scheme called the receiver-window modification scheme.

3. Receiver-Window Modification (RWM)

We are proposing to use both flow and congestion control feedbacks to reduce congestion at routers. In this paper, we will deal with congestion occurring especially at the ingress and gateway routers which are two major congestion areas within the network (see item 3. of Section 5 with regards to core routers). Instead of setting ECN related bit in the chosen packets of RED, ARED and BLUE queues, we propose the use of receiver window modification (RWM) at these routers. RWM sets the receiver window field to one maximum segment size (MSS) in the ACK packets[1] that are going towards the sender from the receiver, instead of early-dropping or marking the packets at the queue. The only exception that it will not modify the receiver window field is when the field has a value of 0. This occurs when a TCP application wants to tell its peer not to send any more data. We denote RWM modification to AQM schemes by affixing "RWM" after the name of the AQM, thus we will be talking about RED-RWM, ARED-RWM and BLUE-RWM respectively.

In the case of RED-RWM and ARED-RWM, the field is set to 1, instead of early-dropping or marking the packets, only if the average queue size is between MIN_{th} and MAX_{th}. However, if the average queue size is greater than MAX_{th}, the packet is not only dropped but the field in the ACK is also set to 1. The idea here is to reduce the rate as soon as possible without waiting for the timeout mechanism or the duration of the reception of three duplicate ACKs to take effect or elapse, thus cutting down on the number of packets that would have otherwise been dropped. The threshold values used are the same as in RED and ARED queues. In the case of RED-RWM and ARED-RWM, the recommended P_{max} is bounded from 0.3 to 1 respectively.

In the case of BLUE-RWM, it uses the probability, P_a, to modify the ACK packets instead of marking or early-dropping of queued packets. P_a is increased whenever ACK packets are modified or/and data packets are dropped from the queue due to overflow and decreased when the link is underutilized. If the queue is continually dropping packets due to buffer overflow, BLUE-RWM increments P_a, thus increasing the rate at which it modifies the field to 1. Conversely, if the queue becomes empty or if the link is idle, BLUE-RWM decreases P_a, thereby decreasing the rate of ACK modification. This effectively allows BLUE-RWM to "learn" the correct rate it needs to modify the field. The freeze_time is randomized to avoid global synchronization.

Upon receiving the modified ACK packet, the sender will transmit the minimum of the congestion and the advertised received window sizes. Whenever the TCP header of an ACK packet is modified, the checksum in the TCP header needs to be adjusted for error control.

The advantages of RWM queues are that they work with the current implementation of TCP in end systems and do not require changes to the existing network schemes except at the ingress and gateway routers where the RWM scheme is to be implemented. Hence, they overcome the disadvantage of the ECN queues since in ECN, TCP/IP stacks at both ends require modifications to be "ECN-compliant" in addition to the router. Since, the feedback loop extends from the RWM queue on the router to the sender; the response of the algorithm is determined by the delay between the router and the sender rather than the RTT of the connection, which is true in the case for the queues using the ECN mechanism. This implies that the longer the RTT is, the greater the advantage that RWM has over ECN queues. The delayed response of the sender to an ECN is further compounded if the congestion worsens at the downstream routers where ECN messages could be delayed or worse, dropped. Moreover, the response time will also be heavily dependent on the types of queue implementations at these routers. Presently, almost all the routers along a path use the drop tail queues. We will show, in Section 4, that introducing RED, ARED and BLUE AQM queues at congested routers, such as at the ingress and gateway routers, in the present network will only increase the response time when compared to the RWM scheme. The queues using the RWM mechanism enjoy the advantage of faster response times to the impending congestion since the notification of congestion arrives at the source quicker when compared to those using ECN mechanism. A theoretical model of the RWM scheme and its analysis has been presented in[27]. Here, we had 1) provided a weak convergence theorem for the number of connections; thus leading the way to a more relaxed configuration of RED-RWM gateway parameters, 2) used a Monte Carlo simulation method to compare and verify outcomes of our mathematical model with those of NS2 simulation experiments and 3) proved mathematically that the average queue size is approximately the minimum

[1] Some researchers claim that routers in general should not look at layer-4 headers as their function is limited to layer-3. Yet, enabling them to do so can provide significant benefits (see e.g., NAT).

threshold parameter of the RED-RWM queue.

In the next section, we carry out extensive NS2 simulations to show our results and to support our claims that the RWM scheme improves popular queue-based AQM.

4. Simulation Results

We have conducted an extensive simulation experiment to evaluate and compare AQM algorithms with and without ECN, and with our proposed RWM scheme. Our simulations were based on an extended and corrected NS2[30] simulator since the receiver window–congestion window interaction in NS2 does not follow a common linux-type TCP implementation.

We compare AQM schemes on the simple network topology of 4 sources connected to a sink via 4 routers as shown in Figure 5; the nearest router to the sources, i.e. the ingress router, employs the AQM to be investigated while the other 3 routers contain Drop Tail queues. The first set of experiments, called Scenario 1, we compare the RED, RED-ECN and RED-RWM queues. In the second set, Scenario 2, we compare the ARED, ARED-ECN and ARED-RWM queues while in the third, Scenario 3, we compare BLUE, BLUE-ECN and BLUE-RWM queues. In each scenario, we first compare the average queue sizes of the AQM schemes and then, keeping the same topology and AQM parameters, we vary the number of sources to investigate their performance at increasing levels of congestion by collecting essential data and analysing them in terms of i) delay jitter (the variation in the time taken for packets to be transmitted from the source to the destination in a network.), ii) average packet delay, iii) number of packets dropped and iv) throughput. In Scenario 4, we compare all the RWM queues in terms of these 4 metrics.

Here, we are simulating the effect of introducing RWM-compliant ingress or gateway routers in the present-day network which comprises of Drop Tail queues. Since it is too costly to change the entire network, we recommend finding a practical solution which minimizes cost and at the same time enhances the flows using the network by reducing their average packet delay, number of packet drops and delay jitter while improving their throughput. We are targeting ingress and gateway routers as they are a major source of congestion and not the core routers as these are over-provisioned with high speed links that experiences lesser congestion[17]. Different and complex topologies comprising of ingress and gateway routers containing these queues and core routers with Drop Tail queues were experimented and their outcomes were similar to those that were obtained with the simple topology shown in Figure 5. Hence, we are using this simple topology here for ease of understanding. The subsequent subsections explain the details and discuss the results of the different simulation scenarios. We had used TCP Reno as most modern TCP's are "Reno" based and had used RED, ARED and BLUE as these are the more popular AQM schemes that had been proposed to replace the Drop Tail queues. Since TCP is chaotic in nature[23], we ran each experiments several times using different seeds and averaged the results to get more accurate readings.

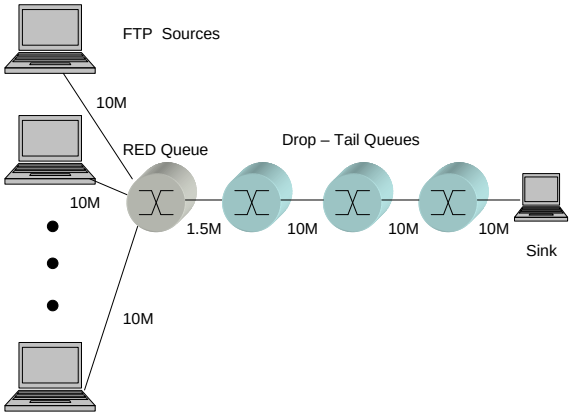

Figure 5. Simulation network topology

4.1. Scenario 1 – RED AQM

We first study the performance of TCP with RED, RED-ECN and RED-RWM. During the interval (0, 150) seconds, four File Transfer Protocol (FTP) connections are started at time t = 0s with the size of the RED queue set at 35 packets. The RED parameters were set as follows:

- Queue weight given to current queue size sample = 0.002
- Minimum threshold of average queue size (MIN_{th}) = 5
- Maximum threshold of average queue size (MAX_{th}) = 15
- Max probability of dropping a packet = 1/linterm, where linterm = 3

Simulation results were obtained for RED, RED-ECN and RED-RWM AQMs employed in the first router. Figure 6 depicts the average queue size versus the simulation time of the investigated AQMs in a congested state. It shows that the RED-RWM reduces the average queue size by about 25% compared to RED and RED-ECN. The reasons are as follows: In the case of the RED-RWM queue, its feedback loop is much shorter since we are modifying the receiver window field to one maximum segment size (MSS) in the ACK packets that are going towards the sender from the receiver, instead of early-dropping or marking the packets at the queue. This modification occurs as soon as the average queue size is above the minimum threshold (MIN_{th}) of 5 packets. On the contrary, the feedback loop for the RED-ECN is much longer, nearly a round trip time, thereby delaying the TCP senders' ability to respond to the impending congestion in a timely manner. As a result, more packets enter the network, causing increases in the average queue sizes of the RED-ECN and the Drop Tail queues. These increases, in turn, are responsible for marked packets getting dropped by these queues or delayed as a result of being queued for a longer period of time. These drops and delays further hinder the TCP senders' ability to react in a timely manner to ease the congestion resulting in more packets entering the already congested network. Since the feedback loop is extremely short, unlike in RED-ECN, the RED-RWM queue is thus able to keep its average queue size as close as possible to the minimum

threshold. Hence it is able to control the congestion better than RED-ECN. As for RED, packets are forcedly dropped whenever the average queue size is greater than MAX_{th} and randomly dropped whenever the average queue size is between MIN_{th} and MAX_{th}. In comparison, in RED-ECN and RED-RWM queues, packets are only forcedly dropped when their average queue sizes are greater than MAX_{th}. Hence, RED experiences the greatest number of packet drops among the three. The duration of the reception of three duplicate ACKs, due to these dropped packets, severely delays the response time of the TCP congestion scheme causing the larger average queue sizes when compared to RED-RWM. A similar observation was also noticed when we had used timeouts in our experiments. As seen in Figure 6, the RED-ECN and RED have slightly similar plots though for different reasons.

Next, keeping the same topology and RED parameters but varying the number of sources, we investigated the performance of RED, RED-ECN and RED-RWM by collecting essential data and analyzing them in terms of i) delay jitter , ii) average packet delay, iii) number of packets dropped and iv) throughput. Figures 7, 8, 9 and 10 show the respective results obtained for 1000 seconds of simulation time.

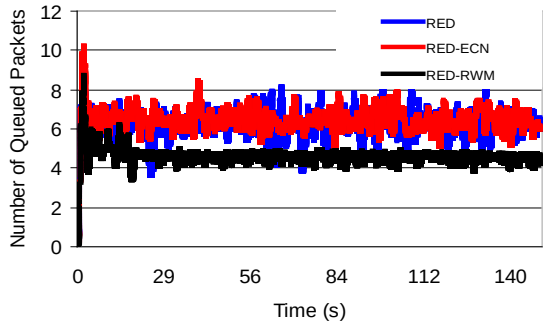

Figure 6. Average queue sizes for RED, RED-ECN and RED-RWM

Larger average queues mean greater buffering delay. When the buffering delay is increased, the corresponding round-trip times increase and cause the aggregate TCP behaviour to be less aggressive. Likewise, when the buffering delay is decreased, the corresponding round-trip times decrease and cause the aggregate TCP behaviour to be more aggressive. Hence, smaller average queue sizes imply smaller queuing delay and as a result, there is an improvement in the one-way packet delay for the flows using the RED-RWM queue as compared to those obtained using RED-ECN and RED queues, as can be seen in Figure 7. Meanwhile, RED performs slightly better than RED-ECN since RED experiences delays mainly due to lost packets and RED-ECN experiences delays due to, not only from lost packets, but also from marked packets experiencing drops and/or congestions in the downstream routers. As the congestion increases, RED-ECN increasingly randomly marks the packets while RED increasingly randomly drops them. These randomly marked packets experience congestion in the downstream routers which may cause a bigger delay when compared to the duration of the reception of the three duplicate ACKs due to RED's randomly dropped packets. Hence, RED initially performs much better than RED-ECN when the number of sources is just small enough to cause congestion within the network. As the number of sources is increased, the number of RED packet drops is only slightly greater than those of RED-ECN, resulting in RED performing only slightly better than RED-ECN in terms of average packet delay. Since its average queue length is the least, the average packet delay for flows using the RED-RWM queue is 15-20% less than those using RED-ECN and RED queues. Moreover, using the RWM queue significantly and consistently reduces the variation of the delay for its flows by about 35% as shown in Figure 8 since the variation in the average queue size of the RED-RWM queue (see Figure 6), as well as those of the Drop Tail queues at the downstream routers, is most stable among the three.

As expected, in Figure 9, when the path is lightly congested, both the topologies using RED-ECN and RED perform as well as that of RED-RWM. However, as the flow path gets increasingly congested, the throughput of the flows using both RED-ECN and RED drops significantly and then tapers off at around 93% when the path becomes fully saturated. The throughput of the flows using the RED-ECN queue is greatly affected by the number of dropped packets whenever 1) their marked packets get dropped or delayed by downstream routers and 2) its average queue size exceeds MAX_{th}. As for the flows using the RED queue, their throughput is greatly affected by the number of randomly and forcedly dropped packets. As the links get saturated, i.e. when the average queue size is greater than MAX_{th}, the plots of RED-ECN and RED behave similarly since they both forcibly drop packets. RED-RWM performs the best since it drops the least number of packets as seen in Figure 10. In Figure 7, the average packet delay for the flows using the RED-ECN and RED queues are much greater than those of RED-RWM queue as the number of sources are increased. This delay also affects the throughput of the flows since fewer packets are transmitted and received in a given time period and hence, flows using the RED-RWM queue easily outperform those using RED and RED-ECN queues.

In Figure 10, flows using the RED-RWM queue drop the least number of packets. In the beginning, the plots for both the RED-RWM and RED-ECN were similar. This was expected since the network is lightly congested, RED-ECN is able to control the congestion as well as RED-RWM. However, as explained earlier, as the congestion increases in the RED-ECN queue as well as in the downstream routers, the number of dropped packets by the flows using the RED-ECN queues is increased. It starts to drop just as many as the RED queue and hence, the plots of RED-ECN and RED become nearly identical. Since it modifies instead of early dropping, flows using the RED-RWM queue drops less packets than those using the RED queue, whereas the disadvantages of ECN, mentioned in Section 3, especially its dependence on the performance of downstream routers,

leads to the flows using the RED-ECN queue dropping more packets than the flows using the RED-RWM queue. Most of the packets that are dropped by flows using the RED-RWM queue occur at the Drop Tail queues of the downstream routers and not at the RED-RWM queue.

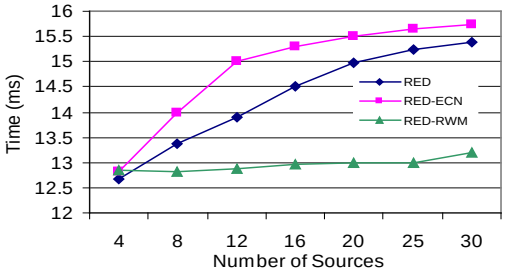

Figure 7. Average packet delay for RED, RED-ECN and RED-RWM

The results from these figures confirm that RED-RWM outperforms both RED and RED-ECN in all of the above metrics.

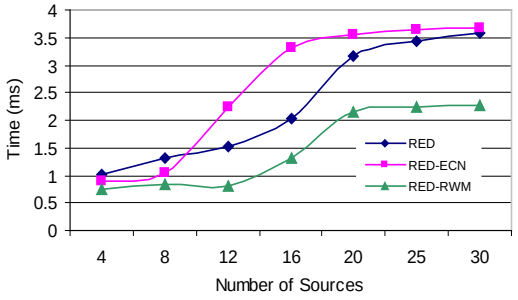

Figure 8. Delay Jitter for RED, RED-ECN and RED-RWM

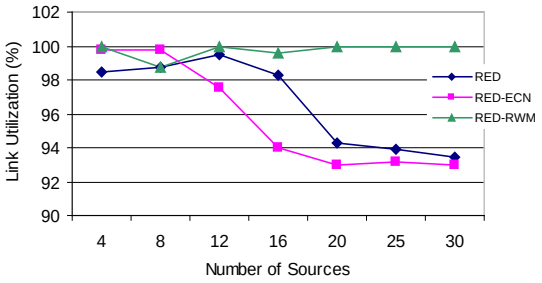

Figure 9. Throughput for RED, RED-ECN and RED-RWM

4.2. Scenario 2 – ARED AQM

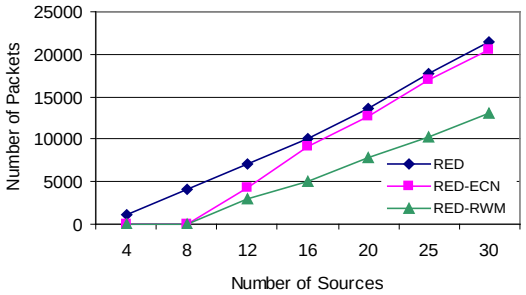

Figure 10. Total packet drops for RED, RED-ECN and RED-RWM

Here, we investigate the performance of ARED, ARED-ECN and ARED-RWM queues using the network topology and the RED parameters of Scenario 1. In Figure 11, the average queue sizes for ARED, ARED-ECN and ARED-RWM are compared. Generally, it can be observed that the average queue sizes for ARED-RWM is smaller than those of ARED and ARED-ECN. The average queue sizes for ARED-RWM are about 50% smaller compared to those of ARED-ECN and ARED. The reasons for this variation between ARED-RWM, ARED and ARED-ECN can be explained in a similar manner to the previous scenario. They are due to the faster response time to the impending congestion since the notification of congestion for ARED-RWM queue arrives at the source faster when compared to the ECN mechanism as in the case of ARED-ECN, and to the response time for the three duplicate ACKs or the TCP timeout mechanism as in the case of ARED. This aggressive reduction in the number of packets, which are being sent into the network with the ARED-RWM queue at the ingress router, enables smaller queue sizes, not only in the ingress router but also in all downstream routers. The ARED-ECN and ARED queues have slightly similar plots though for different reasons, as mentioned in the previous section. Just like the plots of the average queue sizes of RED and RED-ECN (see Figure 6), those of ARED and ARED-ECN are comparable and semi-overlapping. Hence, for both RED and ARED algorithms, the ECN mechanism provides only slight improvements in reducing the congestion at the ingress router when it is introduced in a network consisting of routers with drop-tail queue. Hence, we believe that ECN should be introduced in every hosts and routers within the network or not at all. However, the former is too costly to be implemented.

Keeping the same topology and RED parameters but varying the number of sources and increasing the duration of the experiments to 1000s, ARED, ARED-ECN and ARED-RWM queues were then compared on their one-way packet delay, delay jitter, number of packet drops and throughput.

By reducing the congestion in the network, ARED-RWM queue enhances the flows by reducing their average packet delay (see Figure 12), delay jitter (see Figure 13) and number of packets drops (see Figure 15) while increasing their throughput (see Figure 14) when compared to those flows using the ARED-ECN and ARED queues. As in Scenario 1, the smallest average queue sizes for the ARED-RWM queue in the ingress router and the Drop Tail queues in the downstream routers imply smaller queuing delays, resulting in improvements in the one-way packet delays for the flows when compared to those obtained using ARED-ECN and ARED queues, as can be seen in Figure 12. By having an average queue size of 5 packets, which is equivalent to the minimum threshold (MINth), ARED-RWM reduces the delay better than ARED-ECN and ARED. Hence, it is able to control congestion better by reducing the number of packets from entering and flooding the network. The average packet delay due to the presence of ARED-RWM is 7-10% better than that of ARED and ARED-ECN. Similar to the reasoning why RED is better than RED-ECN in the previous scenario,

ARED is only slightly better than ARED-ECN in terms of average packet delay.

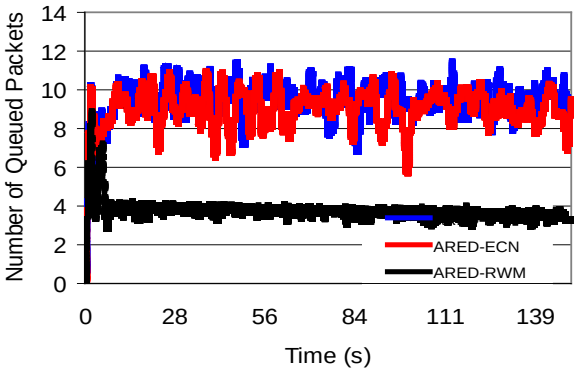

Figure 11. Average queue sizes for ARED, ARED-ECN and ARED-RWM

In Figure 13, ARED-RWM significantly and consistently reduces the variation of the delay for its flows by about 133% since from Figure 11; the variation in its average queue size is the most stable among the three. ARED-RWM's ability to reduce and maintain the number of packets near the minimum threshold (MIN_{th}) enables the variation in the average queue sizes for the Drop Tail queues at the downstream routers to a minimum.

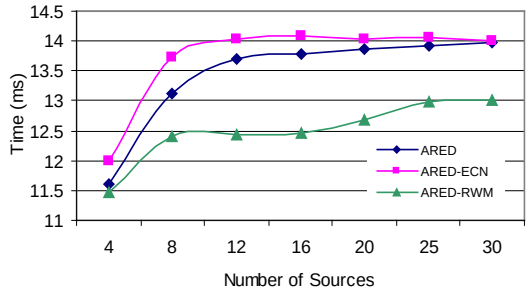

Figure 12. Average packet delay for ARED, ARED-ECN and ARED-RWM

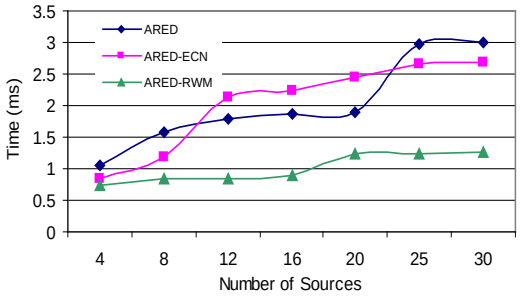

Figure 13. Delay jitter for ARED, ARED-ECN and ARED-RWM

In Figure 14, when comparing throughputs, the plots show a strange behaviour. As the number of sources is increased from 4, all 3 plots drop significantly and increase again afterwards. The reason for this behaviour is because the ARED algorithm used by the ingress queues, adapts too rapidly, resulting in too many packet drops and/or marks as the number of sources is increased. It tries to handle the increasing congestion by behaving too aggressively. Once the network becomes saturated, the plots increase again. This is due to the stabilisation of the ARED's adaptive parameters. Hence, all three are affected by the aggressive nature of the ARED algorithm. Since ARED-RWM keeps the smallest average queue among the three, the ARED algorithm is the least aggressive among the three since the aggressiveness is proportional to the size of the average queue. However, ARED-ECN is further affected by the marked packets getting dropped or suffering delay due to congestion in the downstream routers. These packets are also responsible for the congestion in the downstream routers and hence, more packets are dropped by the Drop Tail packets when compared to those dropped by the downstream routers in the ARED-RWM and ARED topologies. Hence, the number of packet drops for the flows using the ARED-ECN topology is the greatest as the congestion increases as shown in Figure 15.

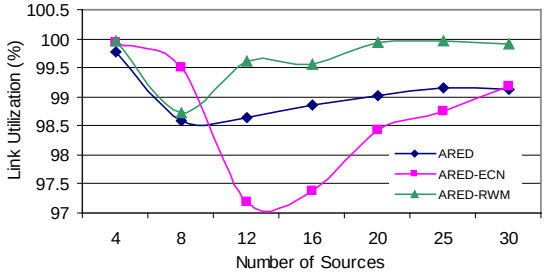

Figure 14. Throughput for ARED, ARED-ECN and ARED-RWM

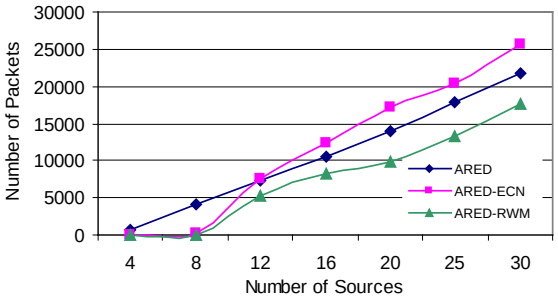

Figure 15. Total packet drops for ARED, ARED-ECN and ARED-RWM

4.3. Scenario 3- BLUE AQM

Here, we investigate the performance of BLUE, BLUE-ECN and BLUE-RWM queues using the network topology of scenario 1. We use the following BLUE parameters[1]:

- Freeze_time = 10ms
- $\delta_1 = 0.02$
- $\delta_2 = 0.002$
- threshold L = 15 packets

The average queue sizes for BLUE, BLUE-ECN and BLUE-RWM are shown in Figure 16. BLUE-RWM is averagely 30-50% better than BLUE and BLUE-ECN. This is due to BLUE-RWM's advantage of faster response times to the impending congestion since its notifications of congestion arrive at the source quicker when compared to those of BLUE-ECN and BLUE. BLUE is better than BLUE-ECN because marked packets by BLUE-ECN get

delayed or dropped at the drop-tail queues of the downstream routers. Hence, BLUE-ECN has the largest average queue size since it uses packet marks to signal congestion. The variation of the average queue sizes using the BLUE algorithm is greater than those of RED and ARED algorithms since the BLUE algorithm uses packet loss and link utilization history to manage congestion while both RED and ARED algorithms use average queue size. Both RED and ARED algorithms tries to maintain a stable average queue size while the BLUE algorithm does not. Using link utilization, the BLUE algorithm is able to react to congestion faster than RED and ARED algorithms. Since RWM enables all three algorithm to react faster, its influence is felt more on RED and ARED queues than on BLUE queues since BLUE queue already reacts faster than both RED and ARED. Hence, the queue size gain is the lowest in BLUE-RWM when compared to RED-RWM and ARED-RWM.

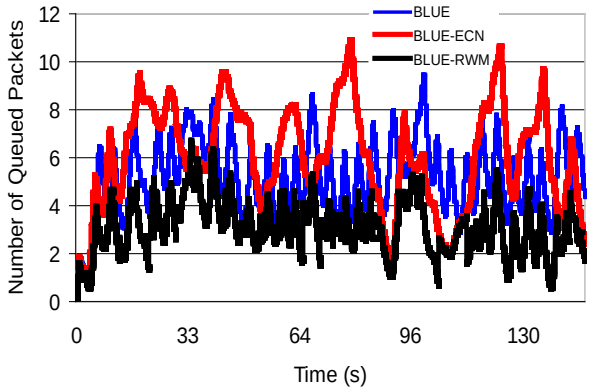

Figure 16. Average queue sizes for BLUE, BLUE-ECN and BLUE-RWM

Keeping the same topology and BLUE parameters but varying the number of sources, BLUE, BLUE-ECN and BLUE-RWM queues are also evaluated in terms of their one-way packet delay, delay jitter, number of packet drops and throughput; Figures 17, 18, 19 and 20 show the respective results. In terms of the packet delay, delay jitter, throughput and number of packet drops, BLUE-RWM significantly outperforms BLUE and BLUE-ECN. The analyses for these are the same as those in the first and second scenarios.

Figure 17 shows that BLUE-RWM performs 2-5% better than BLUE and 2-10% than BLUE-ECN in terms of average packet delay. This is because BLUE-RWM keeps the smallest average queue size in the ingress router and by controlling congestion at the ingress router; it indirectly enables the Drop Tail queues in the downstream routers to maintain the smallest average queue sizes among the three topologies. At low levels of congestion, the performance of BLUE is worse than that of BLUE-ECN but after the number of sources has been increased from 16; BLUE performs better than BLUE-ECN. This is because, when the congestion is saturated, greater number of packets are dropped or marked by BLUE-ECN. Moreover, some of these marked packets get trapped in the buffers of the queues of the downstream routers. As the number of packets being queued at these Drop Tail queues increases, the queuing delay is increased. As a result, these marked packets are increasingly delayed along with the other packets in the queues. Hence, the congestion notifications to the sender are increasingly delayed. These in turn, lead to increased levels of congestion within the network with the introduction of more new packets into the network. On the other hand, the congestion notifications of BLUE, in terms of dropped packets, are more consistent and reliable allowing the sender to reduce the amount of packets being sent into the network. When the congestion levels are low, BLUE-ECN performs better than BLUE since its congestion notifications are not delayed as much as those of BLUE since their delay durations are less than the timeouts or the duration of the reception of three duplicate ACKs.

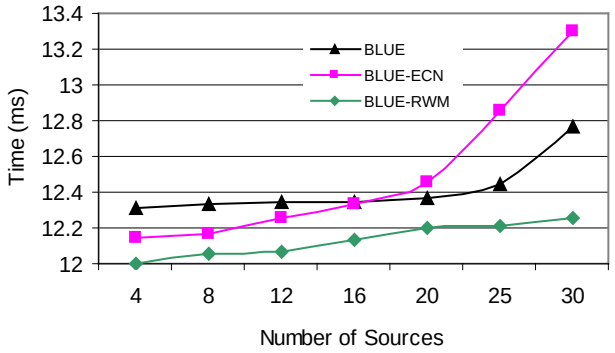

Figure 17. Average packet delay for BLUE, BLUE-ECN and BLUE-RWM

Next, Figure 18 shows that BLUE-RWM performs better than both BLUE and BLUE-ECN in terms of delay jitter. As from Figure 16, the variations in the average queue size for the BLUE-ECN are the largest among the three. These variations which occur at the ingress router are further amplified along the path of the flows. BLUE-RWM improves by 28-70% over BLUE-ECN and 9-29% over BLUE. All three plots show that the jitter rate increases as congestion within the network increases.

In Figure 19, BLUE-RWM improves the throughput by 1-7% over BLUE and BLUE-ECN as the number of sources is increased from 4 to 30. When the number of sources is at 4, BLUE-RWM is only slightly better than the other two since there are low levels of congestion in the network. As the congestion increases, the network consisting of the BLUE-RWM queue benefits from the faster congestion notifications in terms of reduced delay (see Figure 17) due to smaller queuing delay and lesser timeout delays due to smaller number of dropped packets. The throughput of the flows using the BLUE-ECN queue is greatly affected by the number of dropped packets whenever their marked packets get dropped or delayed by the queues in the routers. As for the flows using the BLUE queue, their throughput is greatly affected by the number of randomly and forcedly dropped packets. In Figure 20, not surprisingly, the number of dropped packets is the greatest for BLUE which implies that the throughput for flows using the BLUE queue is lesser than those using the BLUE-ECN queue. As the number of dropped packets for flows using the BLUE-ECN queue

increases as congestion increases, it approaches the performances of BLUE both in terms of throughput and the number of dropped packets. This is expected as congestion reaches saturation; BLUE-ECN behaves just like BLUE since, instead of marking, it drops packets.

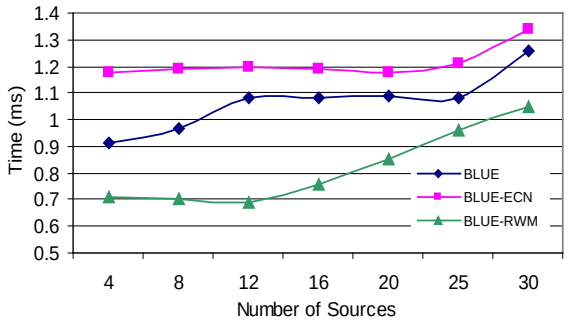

Figure 18. Delay jitter for BLUE, BLUE-ECN and BLUE-RWM

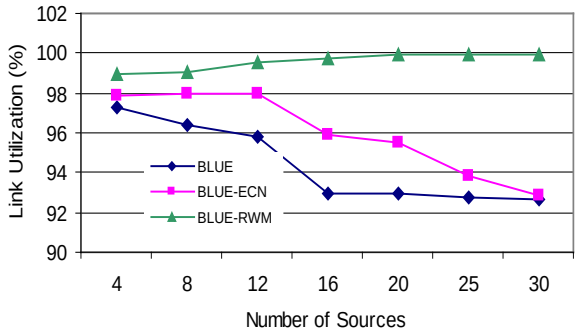

Figure 19. Throughput for BLUE, BLUE-ECN and BLUE-RWM

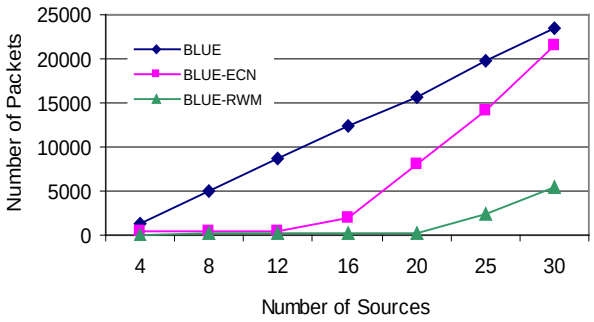

Figure 20. Total packet drops for BLUE, BLUE-ECN and BLUE-RWM

4.4. Comparing RED-RWM, ARED-RWM, and BLUE-RWM

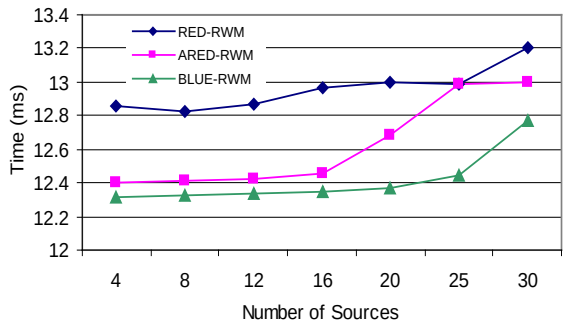

Figure 21. Average packet delay for RED-RWM, ARED-RWM and BLUE-RWM

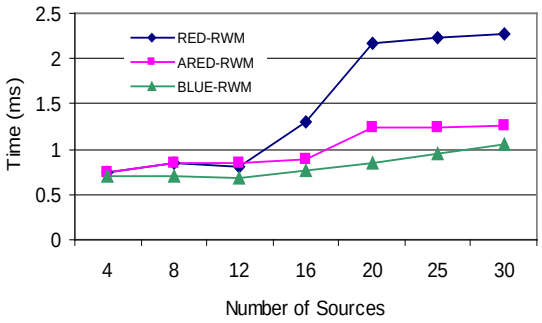

Figure 22. Delay jitter for RED-RWM, ARED-RWM and BLUE-RWM

In this section we are comparing RED-RWM, ARED-RWM, and BLUE-RWM AQMS in terms of average packet delay, delay jitter, number of packet drops and throughput. Figures 21, 22, 23 and 24 show the respective simulation results. Overall, BLUE-RWM outperforms both RED-RWM and ARED-RWM as the number of sources increases.

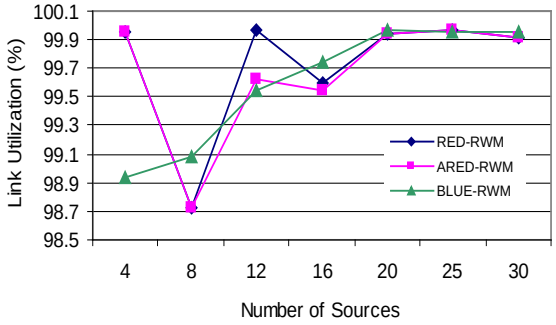

Figure 23. Throughput for RED-RWM, ARED-RWM and BLUE-RWM

In Figure 23, BLUE-RWM initially does not perform as well as RED-RWM and ARED-RWM because of the aggressiveness of the BLUE parameter, P_a. It is increased rapidly, unlike RED-RWM and ARED-RWM, when the number of sources is increased, i.e., whenever packets are marked from the queue and is never decreased, since the link is not underutilized. Hence, it reduces the number of packets entering the network the quickest when compared to the other two. This initially affects its throughput. On the other hand, RED-RWM and ARED-RWM need to wait till their average queue size reaches the minimum threshold of MIN_{th} before marking packets. After the number of sources is increased to 16, all three exhibit high throughputs.

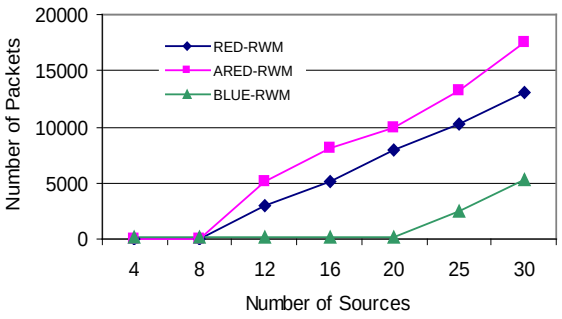

Figure 24. Total packet drops for RED-RWM, ARED-RWM and BLUE-RWM

In all other categories, BLUE-RWM performs the best. The reasons why BLUE-RWM performs better than RED-RWM and ARED-RWM in these categories are due to the fact that BLUE-RWM inherits BLUE ability to perform queue management based directly on packet loss and link utilization rather than on the average queue sizes as in the case of RED-RWM and ARED-RWM. They enable the BLUE algorithm to react to congestion faster than RED and ARED algorithms. RWM causes all three algorithms to react faster. BLUE-RWM reacts faster than RED-RWM and ARED-RWM since the BLUE algorithm already reacts faster than both RED and ARED algorithms. Hence, BLUE-RWM outperforms RED-RWM and ARED-RWM in all the four categories. The experiments also show that there is only slight difference between RED-RWM and ARED-RWM. Using the RWM scheme to manage congestion, allows the ARED algorithm to behave similar to the RED algorithm as congestion notifications when using both algorithms arrive in a similar manner as the congestion increases and since they both use average queue size in their algorithms.

5. Conclusions and Future Work

In this paper we have proposed a modification to existing adaptive queue management protocols (AQM) called Receiver Window Modification (RWM). RWM does not require modification to all end system TCP/IP stacks but can be solely implemented in heavily-congested ingress and gateway routers. This means that RWM does not require both the sources and receivers to be "RWM-compliant" as was the case for ECN-compliant queues. Our RWM scheme helps to reduce the average queue sizes of the RED, ARED and BLUE queues. By reducing the average queue sizes, RWM queues reduce the queuing delay resulting in significant improvements in one-way end-to-end packet delays and number of dropped packets. It is also shown that the performance of RED-ECN, ARED-ECN and BLUE-ECN queues is heavily dependent on the queues of the downstream routers. RWM queues in ingress and gateway routers are not influenced by the number and state of the downstream router as they will piggyback congestion information to the source in the next available acknowledgement packet.

We have used the NS2 simulator to carry out simulation experiments validating the RWM scheme. We are now currently working on research to

1) Implement RWM in Linux based packet routers.

2) Control congestion caused by non-TCP flows in order to complement the gains obtained by using the RWM scheme for TCP flows.

3) Control congestion with an extended-RWM (ERWM) scheme at the edge routers to ensure that at the borders of the network that each flow's packets do not enter the network faster than they are able to leave it. The aim here is to push any complexity in controlling congestion toward the edges of the network by not modifying the core routers and the end systems.

REFERENCES

[1] W. Feng, D. Kandlur, D. Saha, K. Shin, "The BLUE active queue management algorithms", IEEE/ACM Transactions on Networking, vol.10, no.4, pp.513 – 528, 2002.

[2] M. Kwon, and S. Fahmy, "A comparison of load-based and queue-based active queue management algorithms," in Proceedings of the 2002 International Society for Optical Engineering, vol.4866, pp.35-46, 2002.

[3] S. Athuraliya, S.H. Low, V.H. Li, Y. Qinghe, "REM: Active Queue Management", IEEE Network, vol.15, no.3, pp.48-53, 2001.

[4] S. Kunniyur and R. Srikant, "Analysis and Design of an Adaptive Virtual Queue (AVQ) Algorithm for Active Queue Management," in Proceedings of the 2001 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications, pp.123–134, 2001.

[5] C.V. Hollot, V. Misra, D. Towsley and W.B. Gong, "On Designing Improved Controllers for AQM Routers Supporting TCP Flows," in Proceedings of the 2001 INFOCOM, pp.1726-1734, vol.3, 2001.

[6] S. Floyd, "Measurement Studies of End-to-End Congestion Control in the Internet," 2002, http://www.icir.org/floyd/ccmeasure.html.

[7] T.J. Ott, T.V. Lakshman, and L.H. Wong, "SRED: Stabilized RED," in Proceedings of the 1999 IEEE INFOCOM, vol.3, pp.1346-1355, 1999.

[8] D. Lin and R. Morris, "Dynamics of random early detection," in Proceedings of the 1997 ACM SIGCOMM, vol.27, pp.127–136, 1997.

[9] S. Floyd, R. Gummadi, and S. Shenker. "Adaptive RED: an algorithm for increasing the robustness of RED's Active Queue Management", Online Available: http://www.icir.org/floyd.

[10] W. Feng, D. D. Kandlur, D. Saha and K. G. Shin, "BLUE: an alternative approach to active queue management," in Proceedings of the 2001 11th international workshop on Network and operating systems support for digital audio and video, pp.41–50, 2001.

[11] S. Floyd, "TCP and Explicit Congestion Notification," ACM Computer Communication Review, vol.24, pp.10-23, 1994.

[12] K. Ramakrishnan, S. Floyd and D. Black, "The Addition of Explicit Congestion Notification (ECN) to IP," RFC 3168, 2001.

[13] P. P. Mishra and H. Kanakia, "A hop by hop rate based congestion control scheme," in Proceedings of the 1992 ACM SIGCOMM, 1992.

[14] D. Katabi, M. Handley, C. Rohrs, "Internet Congestion Control for Future High Bandwidth-Delay Product Environments," in ACM SIGCOMM, pp.89-102. 2002.

[15] K. Chen, K. Nahrstedt, and N. Vaidya, "The utility of explicit rate-based flow control in mobile ad hoc networks," in Proceedings of the 2004 IEEE Wireless Communications and Networking Conference, vol.3, pp.1921 – 1926, 2004.

[16] Y. Yi, S. Shakkottai, "Hop-by-hop Congestion Control over a Wireless Multi-hop Network," IEEE/ACM Transactions on Networking, vol.15, pp.133 – 144, 2007.

[17] J.F. Kurose, K.W. Ross, Computer Networking: A Top-Down Approach, 4th ed., Addison-Wesley, USA, pp.43, 2007.

[18] V.V. Govindaswamy, G. Zaruba and G. Balasekaran, "Receiver-Window Modified Random Early Detection (RED-RWM) Active Queue Management Scheme: Modeling and Analysis" in Proceedings of the 2006 IEEE International Conference on Communications, vol.1, pp.158–163, 2006.

[19] V.V. Govindaswamy, G. Zaruba and G. Balasekaran, "Analyzing the Receiver Window Modification Scheme of TCP Queues" in Proceedings of the 2006 INFOCOM, 25th IEEE International Conference on Computer Communications. pp.1 – 6, 2006.

[20] V. Jacobson, "Congestion Avoidance and Control," Proceedings of 1988 ACM SIGCOMM, pp.314-329, 1988.

[21] W. Stevens, TCP/IP Illustrated, Volume 1: The Protocols, Addison-Wesley, USA, 1994.

[22] L. Zhang and D. Clark, "Oscillating Behavior of Network Traffic: A Case Study Simulation," in Proceedings of the 1990 Internetworking: Research and Experience, vol.1, pp.101-112, 1990.

[23] A. Veres and M. Boda, "The Chaotic Nature of TCP Congestion Control," in Proceedings of the 2000 INFOCOM, vol.3, pp.1715 - 1723, 2000.

[24] S. Floyd and V. Jacobson, "Random early detection gateways for congestion avoidance," IEEE/ACM Transactions on Networking, vol.1, no.4, pp.397-413, 1993.

[25] M. May, J. Bolot, C. Diot and B. Lyles, "Reasons Not to Deploy RED," in Proceedings of the 1999 7th International Workshop on Quality of Service, pp. 260–262, 1999.

[26] M. May, T. Bonald and J. Bolot, "Analytic Evaluation of RED Performance," in Proceedings of the 2000 INFOCOM, pp. 1415 - 1424, vol.3, 2000.

[27] R. Mahajan, S. Floyd and D. Wetherall, "Controlling high-bandwidth flows at the congested router," in Proceedings of the 2001 9th International Conference on Network Protocols, pp.192-201, 2001.

[28] W. Feng, D.D. Kandlur, D. Saha and K.G. Shin, "A Self-Configuring RED Gateway," in Proceedings of the 1999 INFOCOM, pp.1320-1328, 1999.

[29] TBIT, the TCP Behavior Inference Tool, 2004, Online Available: http://www.icir.org/tbit/ecn-tbit.html.

[30] NS simulator, Online Available: http://www.isi.edu/nsnam /ns/.