International Journal of Computer Applications (0975 – 8887)
Volume 15– No.2, February 2011

An Experimental Analysis of Random Early Discard (RED) Queue for Congestion Control

Md. Shohidul Islam
Dept. of CSE
DUET, Gazipur-1700, Bangladesh

Md. Niaz Morshed
Dept. of CSE
DUET, Gazipur-1700, Bangladesh

SK. Shariful Islam
Dept. of CSE
DUET, Gazipur-1700, Bangladesh

Md. Mejbahul Azam
Dept. of CSE
DUET, Gazipur-1700, Bangladesh

ABSTRACT
Active Queue Management (AQM) is receiving wide attention as a promising technique to prevent and avoid congestion collapse in packet-switched networks. By providing advanced warning of incipient congestion, end nodes can respond to congestion before router buffer overflows and hence ensure improved performance. Random Early Discard (RED) is an IETF recommended active queue management scheme that is expected to provide several Internet performance advantages such as minimizing packet loss and router queuing delay, avoiding global synchronization of sources, guaranteeing high link utilization and fairness. It tends to drop packets from each connection in proportion to the transmission rate the flow has on the output link. It does not minimize the number of dropped packets as expected, but it manages to achieve improved performance when compared to the Tail Drop. In this paper, extensive experimental analysis has been carried out on RED using Network Simulator (NS-2) in relation to congestion control and decision has been settled where RED can perform better.

Keywords
AQM, RED, Congestion, NS-2.

1. INTRODUCTION
RED is the first active queue management algorithm proposed for deployment in TCP/IP networks. Transmission Control Protocol (TCP) includes eleven variants-Tahoe, FullTcp, TCP/Asym, Reno, Reno/Asym, Newreno, Newreno/Asym, Sack1, Fack, Vegas and VegasRBP as implemented in NS-2 [8]. In the traditional tail drop algorithm, a router or other network component buffers as many packets as it can, and simply drops the ones it cannot buffer. If buffers are constantly full, the network is congested. Tail drop distributes buffer space unfairly among traffic flows. RED monitors the average queue size and drops packets based on statistical probabilities [3]. If the buffer is almost empty, all incoming packets reaccepted. As the queue grows, the probability for dropping an incoming packet grows too. When the buffer is full, the probability has reached 1 and all incoming packets are dropped [2].

2. PERFORMANCE ANALYSIS
2.1 Variation of Threshold over Simulation Periods
Simulation has been started with minimum threshold 15 and maximum threshold 40. Average queue size lies between min and max threshold. The minimum threshold (min_{th}) has been varied each time and the number of packets was counted at destination node during entire simulation period in connection with several TCP variants whose amount was as in Table 1 to Table 4.

Table 1. No. of received packet for various TCP variants with respect to threshold for simulation time 70s.

TCP variants	Threshold				
	15	20	25	30	35
Reno	863	1192	845	701	729
Newreno	702	773	782	784	751
Vegas	851	778	691	685	615
Fack	809	731	723	624	764
Sack1	864	877	789	827	785

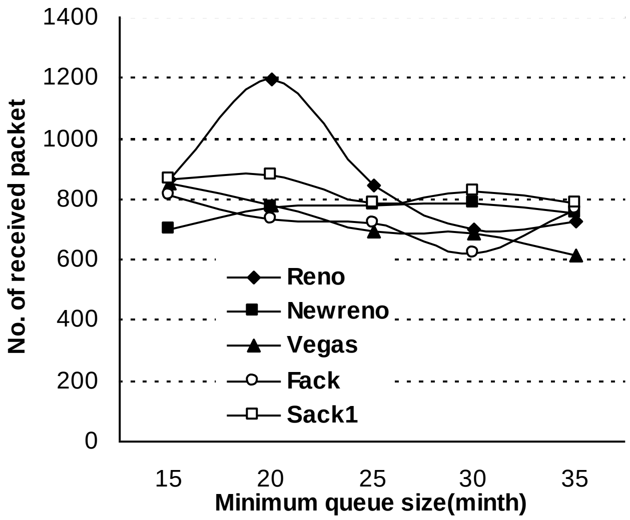

Figure 1.Graph of received packet for various TCP variants with respect to threshold for simulation time 70s.

It is observed that RED queue is very important for controlling the congestion. It can handle the congestion if user can tune the min_{th} and max_{th} perfectly. RED queue has been monitored very carefully and it is founds that if the min_{th} is increased then the packet drop decreases. RED queue was also applied against various TCP versions- Reno, Newreno, Fack,Vegas and Sack1.When the min_{th} was increased and other RED parameters then the number of packets successfully received apparently increased as demonstrated in Table I, Table II, Table III and Table IV and the corresponding figures for simulation time 70s, 140s, 210s and 280s respectively.

Table 2. Received packet for various TCP variants with respect to threshold for simulation time 140s

TCP variants	Minimum Threshold				
	15	20	25	30	35
Reno	1448	1540	1311	1772	1377
Newreno	1452	1454	1493	1622	1541
Vegas	1335	1582	1350	1480	1541
Fack	1499	1786	1253	2381	1429
Sack1	1503	1379	1602	1365	1182

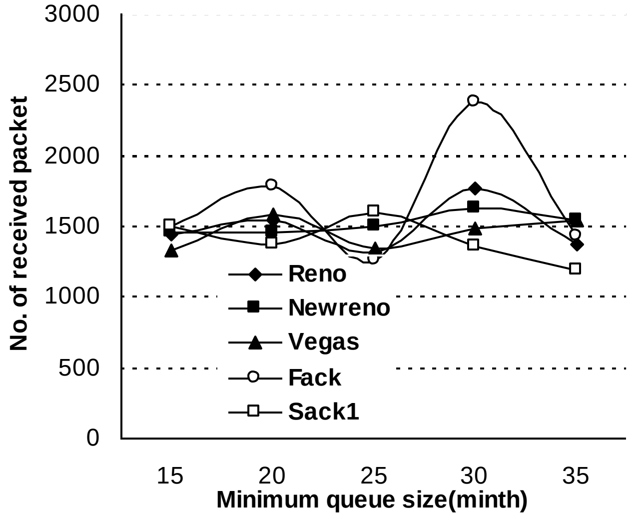

Figure 2.Graph of received packet for various TCP variants with respect to threshold for simulation time 140s.

Table 3. Received packet for various TCP variants with respect to threshold for simulation time 210s.

TCP variants	Minimum Threshold				
	15	20	25	30	35
Reno	2686	2638	2375	1949	2300
Newreno	2697	2545	2013	2173	2303
Vegas	2249	2275	2294	2428	2197
Fack	2792	2463	2908	2127	2369
Sack1	2274	2406	2192	2546	2068

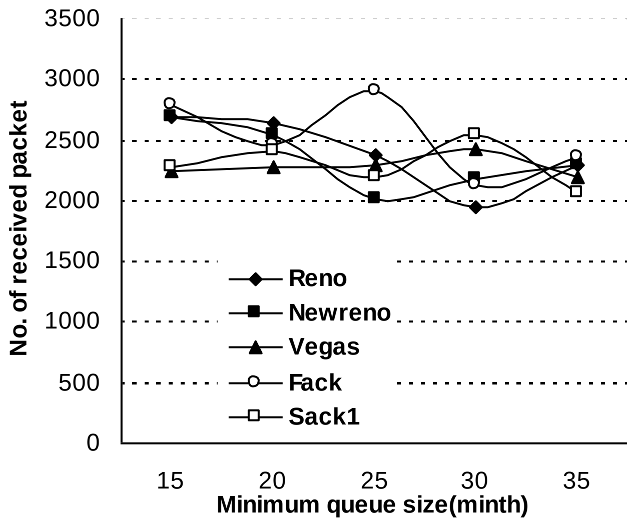

Figure 3.Graph of received packet for various TCP variants with respect to threshold for simulation time 210s.

Table 4. Received packet for various TCP variants with respect to threshold for simulation time 280s

TCP variants	Minimum Threshold				
	15	20	25	30	35
Reno	3140	3403	3311	3321	2900
Newreno	3384	3227	3205	3263	2926
Vegas	2628	2743	2778	2539	2791
Fack	3541	3083	2852	2682	4292
Sack1	3889	3214	3053	3236	3402

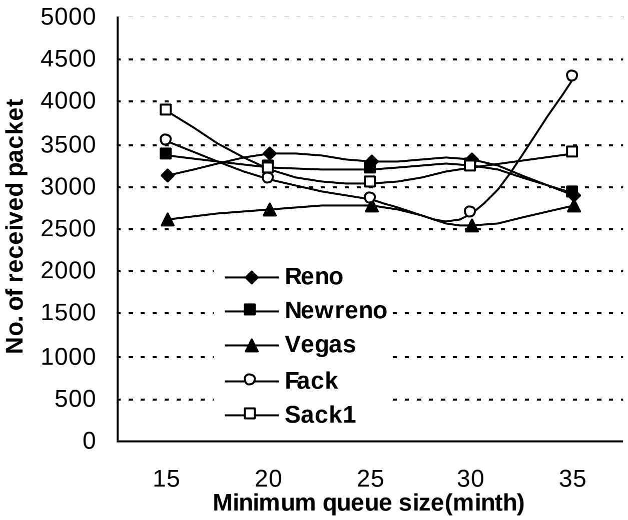

Fig 4.Graph of received packet for various TCP variants with respect to threshold for simulation time 280s.

Finally the observation was for entire simulation duration when threshold is increased then variation occurs in received packet among various TCP variants. It was noticed that at the time of empty queue all arriving packets are received. When average queue size exceeds max threshold or less than minimum threshold then packets are dropped which is shown in above all tables and corresponding figures**.**

2.2 RED Performance with TCP and UDP

From figure 5, it is evident that received packet for TCP is greater than that of UDP. The performance of TCP is greater than UDP. By using RED model it was observed that congestion control in TCP is much more than UDP. So decision came to light that RED model control the congestion accurately. To compare the performance it is found that TCP is better than UDP because packet received is higher in it with respect to UDP. That is why packet loss is lower in TCP. For packet drop, it is clear that packet drop is higher in UDP than TCP and also occurs more congestion in it. It is possible to control congestion in TCP using RED model.

Table 5. Performance of RED with UDP and TCP in terms of packet receiving

Time	Packet received for UDP at min_{th}				
	15	20	25	30	35
70s	672	794	757	792	746
140s	1299	1228	1181	1487	1339
210s	1998	1800	2129	2088	1961
280s	2586	2698	2633	2793	2785

Time	Packet received for TCP at min_{th}				
	15	20	25	30	35
70s	569	663	636	541	834
140s	1354	1606	1437	1659	1612
210s	2726	2374	2421	2247	2414
280s	2451	3282	3694	2830	3435

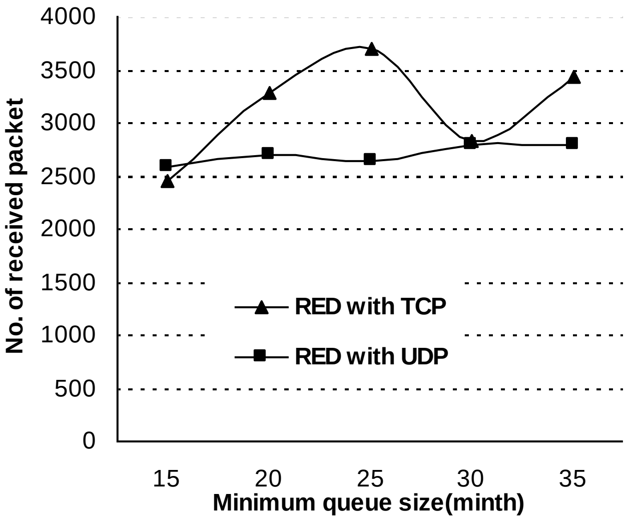

Figure 5.RED performance with TCP and UDP in terms of packet receiving for simulation time 280s.

Table 6. Performance of RED with UDP and TCP in terms of packet dropping

Time	Packet dropped for UDP at min_{th}				
	15	20	25	30	35
70s	25	67	24	131	30
140s	122	106	58	118	32
210s	242	112	429	161	432
280s	372	359	696	349	354

Time	Packet dropped for TCP at min_{th}				
	15	20	25	30	35
70s	0	0	0	0	0
140s	24	13	8	7	5
210s	36	32	31	25	17
280s	71	48	47	36	12

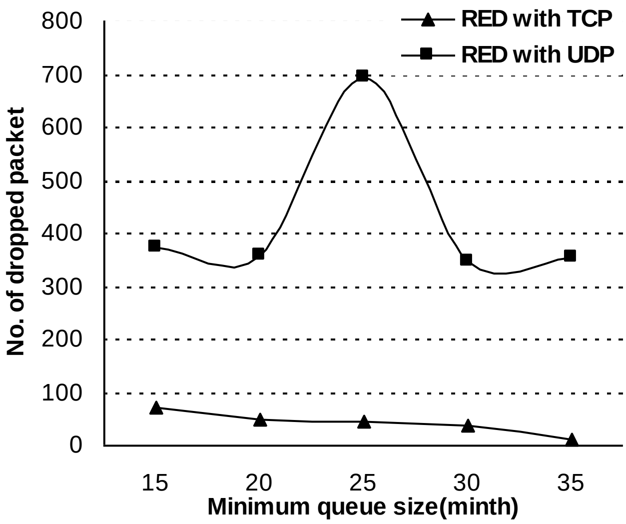

Figure 6. RED performance with TCP and UDP in terms of packet dropping for simulation time 280s.

2.3 Comparison of RED Algorithm with Drop Tail

In scenario with RED algorithm slightly more packets were sent through the network. What is more interesting the proportion between number of received TCP packets and number of received UDP packets was a little shifted. When RED algorithm was used less UDP packets and at the same time, more TCP packets were sent (1. 7% more TCP packets sent). This slightly lessens the unfairness in allocation of bandwidth among responsive and non-responsive flows.

Table 7. performance of red with drop tail buffer

Packet type	Packet received for	
	Drop Tail	RED
TCP Packets	1310	1332
UDP Packet	1120	1110

Greater end-to-end delay in scenario with tail drop algorithm is a result of heavy load that UDP traffic creates. Queue is maintained in almost full state and cause buffer delay to increase. The use of RED results in keeping the average queue length small and reduces the overall delay as buffer delay is smaller. The only disadvantage of using RED queue management algorithm in case of mixed TCP and UDP traffic is greater number of dropped packets. With only TCP flows present, number of dropped packets is smaller when active queue management is used. Presence of UDP flow causes a state of heavy load in the network. As UDP flows do not respond to congestion indication, more packets have to be dropped to keep the average queue length small.

3. CONCLUSION

Beginning section of the paper aims to find which TCP variant works better with RED as it is known that TCP is the mostly used protocol and it has a lot of variants and among them Reno, Newreon, Vegas, Fack and Sack1 have been considered here. Thereafter, an attempt has been under taken to devise RED performance with UDP and TCP in terms of packet receiving and packet dropping which is followed by performance investigation of RED with its another counter part Drop tail.

4. REFERENCES

[1] S. Floyd. RED: Discussions of setting parameters. http://www.aciri.org/ floyd/REDparameters.txt, 1997

[2] S. Floyd and V. Jacobson. Random early detection gateways for congestion avoidance. IEEE/ACM Transactions on Networking, 1(4):397{413, August 1993

[3] S. Floyd and V. Jacobson. The synchronization of periodic routing messages. IEEE/ACM Transactions on Networking, 2(2):122{136, April 1994.

[4] W. E. Leland, M. S. Taqqu, W. Willinger, and D. V. Wilson. On the self similar nature of Ethernet traffic(extended version). IEEE/ACM Transactions on networking, 2(1), February 1994.

[5] "The ns Manual", The VINT Project, A Collaboration between researchers at UC Berkeley, LBL, USC/ISI, and Xerox PARC. December 13, 2003

[6] Jae Chung and Mark Claypool, "NS by Example".

[7] Behrouz A.Forouzan,Sophia Chung,"Data Communication and Networking" ,New Delhi, third edition,2004.

[8] Hari Balakrishnan, Venkata N. Padmanabhan, Srinivasan Seshan, and Randy H. Katz, "A Comparison of Mechanisms for Improving TCP Performance over Wireless Links", Student Member, IEEE.

[9] M. Miyoshi, M. Sugano and M. Murata,"Performance evaluation of TCP throughput on wireless cellular networks", IEEE Vehicular Technology Conference (VTC) 2001, vol.3, 2001, pp.2177 –2181.

[10] Floyd, S., and Jacobson, V., *On Traffic Phase Effects in Packet-Switched Gateways*, Internetworking Research and Experience, V.3 N.3, September 1992, p.115-156.

[11] Hashem, E., "Analysis of random drop for gateway congestion control", *Report LCS TR-465*, Laboratory for Computer Science, MIT, Cambridge, MA, 1989, p.103.

[12] "NS Simulator for beginners", Lecture notes, 2003-2004, Univ. de Los Andes, Merida, Venezuela and ESSI, Sophia-Antipolis, France, December 4, 2003.

[13] KREMPA "Analysis of RED algorithm with responsive and non responsive flows" Poznan University of Technology Academic journals".

Md. Shohidul Islam completed B.Sc Engineering degree in Computer Science & Engineering under the faculty of Electrical & Computer Engineering from Rajshahi University of Engineering & Technology, Rajshahi-6204, Bangladesh in March, 2007. He is a member of IEEE-USA, ACM-USA, IAENG-Hong Kong, IACSIT-Singapore, IEB-Bangladesh and serving as lecturer of Computer Science & Engineering in Dhaka University of Engineering & Technology, Gazipur-1700, Bangladesh. His major research interest includes Wireless networking, Protocol analysis and Algorithm design.

Niaz Morshed obtained his Bacholor of Science in Engineering degree from Department of Computer Science and Engineering (CSE), Dhaka University of Engineering & Technology (DUET), Gazipur, Bangladesh in 2010. His main research areas include wireless and mobile computing, computer networks.

Sk. Shariful Islam obtained his Bacholor of Science in Engineering degree from Department of Computer Science and Engineering (CSE), Dhaka University of Engineering & Technology (DUET), Gazipur, Bangladesh in 2010. His main research areas include wireless and mobile computing.

Md. Mejbahul Azam obtained his Bacholor of Science in Engineering degree from Department of Computer Science and Engineering (CSE), Dhaka University of Engineering & Technology (DUET), Gazipur, Bangladesh in 2010. His main research areas include TCP/IP and Protocol analysis.