A Study of ECN Effects on Long-Lived TCP Connections Using Red and Drop Tail Gateway Mechanisms

MOHAMMED M. KADHUM and SUHAIDI HASSAN
Graduate Department of Computer Science, College of Arts and Sciences, Universiti Utara Malaysia
06010 UUM Sintok, MALAYSIA
{ kadhum@ss.uum.edu.my | Suhaidi@uum.edu.my}

Abstract

Rapid growth of Internet traffic and the increase of new user applications authorize the development of new internet infrastructure. Congestion remains the major problem that affects the Internet service quality. Avoiding packet drops keeps network bandwidth and permits congestion signals to be propagated faster. Sending congestion information is essential to the network performance. Using Explicit Congestion Notification (ECN) to notify the source about the network congestion can result in sending congestion signal faster so that the sender can reduce its congestion window sooner which leads to better network utilization. When ECN is enabled on routers, they mark packets instead of dropping them. ECN mechanism does not require creation of additional transfer at the router and can be applied in the data path of routers. . In this article, we study the behavior of ECN-capable TCP and examine the effect of ECN on long-lived TCP connections using Random Early Detection (RED) and Drop tail gateway mechanisms. We estimate the gain introduced by ECN, in terms of throughput, with different sets of number of TCP users and a point of congestion. Our analysis and simulations results show that the use of ECN over long-lived TCP connections sharing a bottleneck can improve the overall throughput, having less loss, less delay time, and better network utilization

1. Introduction

In recent years, significant amount of investments have been made in the planning and development of computer networks. The rabid growth of the Internet provide a good opportunity for creating new mechanisms for Internet infrastructure to service the increase of new applications, such as web surfing, desktop sharing and video conferencing. The delay variations in network system influence both real-time and non-real-time applications. In an acknowledgement and time-out-based congestion control mechanism, e.g., TCP, performance is related to the delay-bandwidth product of the connection [1]. Furthermore, TCP round-trip time (RTT) measurements are sensitive to delay variations, which may cause wrong timeouts and retransmissions [1].

1.1. The Congestion Issues in Computer Networks

In the design of network one faces many problems, among which are: design; capacity allocation, routing procedure and traffic control procedure [1] [2].
A computer network is a collection of resources which has a finite capacity which causes users to compete for the network resources such as buffers, transmission bandwidth and processor time. The limitation of capacity can result in a degradation of performance of the system to the point that the throughput of the system goes to zero. If the network is overloaded, throughput degradation becomes unavoidable. Networks cannot afford to accept all the traffic that is offered to them without some control.

Therefore, there must be regulations which rule the receipt of traffic from outside and manage the flow inside the network. Congestion problems can appear when the load on the network (the number of packets sent to the network) is greater than the capacity of the network (the number of packets a network can handle) which cause data loss, large delays in data transmission, and a large variance in these delays.

978-1-4244-2328-6/08/$25.00 © 2008 IEEE

Congestion reduces the effective utilization of network resources and causes degradation in the performance experienced by the network users. Therefore, it is desirable to minimize the occurrence of congestion conditions in a network to optimize the utilization of network resources and to provide the network users with acceptable levels of performance.

For a long time, controlling or avoiding congestion is a crucial problem in network design and operation. The availability of cheaper buffers and faster links and processors, in the future, will not completely alleviate the network congestion problem due to mismatching of link speeds, higher offered traffic loads; unforeseeable traffic patterns, large transient loads, and greater degrees of statistical multiplexing will continue to act as sources of congestion [3].

There is a number of schemes have been proposed for network congestion control, and the research for new schemes continues due to the requirements for congestion control schemes that make it difficult to get a satisfactory solution., and network policies that affect the design of a congestion scheme (s). Thus, a scheme developed for one network, traffic pattern, or service requirements may not work on another network. For example, many of schemes developed in the past for best-effort data networks will not work satisfactorily for multi-class IP

Therefore, the study of congestion and congestion management in computer communication networks will continue to be an important research area.

1.2. Congestion in TCP-based Networks

TCP (Transport Control Protocol) is the transport protocol that is responsible of the majority of the Internet traffic. TCP have many goals, some of which are:

- Adapt the transmission rate of packets to the available bandwidth.
- Avoid congestion at the network.
- Create a reliable connection by retransmitting lost packets.

In order to control the transmission rate, the number of packets that have not been received is bounded by a parameter called a congestion window (*cwnd*). The source is forced to wait and stop transmissions the number of packets that it had transmitted and have not been acknowledged reaches *cwnd.*

In most TCP-based networks, TCP uses packet loss as congestion indications. When the queue at the gateway is overflowed, packets are dropped. The TCP source infers the presence of congestion in the network when detecting dropped packets either from the receipt of three duplicate acknowledgements (ACKs) or after the timeout of a retransmit timer, and responds to a dropped packet by reducing the congestion window [4].Sender flow control is achieved with TCP conventionally injecting packets into networks based on feedback of congestion state of the network.

Dropping packets only when the queue overflows and having TCP react only to such losses, results in significant transfer delay as a consequence of the packet retransmission, unnecessarily many packet losses and unfairness due to synchronization effects [5].

1.3. Active Queue Management (AQM)

The Active Queue Management (AQM) mechanisms, such as Random Early Detection (RED) [7], were proposed for solving the abovementioned problems. They detect congestion before gateway queues overflow to avoid the condition of having a massive congestion. AQM mechanisms have better control of bottleneck queues that aim to avoid congestion collapse [4] [6]. The use of AQM can result in reducing queuing delay in addition to better control of the queue, fewer packet drops, and better fairness due to fewer synchronization effects. Queue management mechanisms provide the following benefits many of which are inter-related:

- Early congestion detection
- Early congestion notification
- Congestion avoidance capability
- Less buffer overflows
- Less drop trains
- Easier TCP recovery
- Better throughput proportional drop fairness
- Less packet loss
- Less global synchronization
- Higher link utilization
- Less delay variation (small delay jitter)
- Lower queuing and End-to-End delay
- Avoidance of gateway buffer monopolization

1.4. Random Early Detection (RED)

RED [7] was proposed by Floyd and Jacobson in 1993. It is an efficient mechanism for congestion avoidance at the gateway, in assistance with network transport protocols. RED was originally designed to solve the lock-out and full queue problems associated with tail drop queues, with less attention to the response time. RED is a global-state, probabilistic, proactive queue management policy. It is a development over Early Random Drop [ref] in terms of distinguishing constant congestion from transient upsurges, distinguishing between different congestion levels, and adjusting the packet dropping probability based on the severity of the congestion. This is because of the *Drop Activation Function* that it uses for congestion detection as well as the *Drop Probability Function* used for congestion notification, and queue size control. RED detects incipient congestion and provides feedback to end hosts by dropping the packets. The motivation behind RED is to keep the queue size small, reduce burstiness and solve the problem of global synchronization [l]. RED drops packets before the physical queue is full. It works base on an average queue length that is calculated using an exponential weighted average of the instantaneous queue length. RED drops packets with certain probability depending on the average length of the queue. The drop probability increases from 0 to maximum drop probability (max_p) as average queue size increases from minimum threshold (min_{th}) to maximum threshold (max_{th}). If average queue size goes above (max_{th}) , all packets are dropped.

1.5. Drop Tail

The conventional technique for managing router queue length is Drop Tail, which drops packets only in the event of a buffer overflow. In Drop Tail technique, a maximum queue size is set for each queue which is usually the physical buffer size. The arriving packets are accepted since the maximum queue size is not reached yet. Once the maximum queue length is reached, the following incoming packets are dropped until the queue size decreases as a result of the leave of a packet from the head of the queue.

Drop Tail technique has two main disadvantages, as explained in [8]:

i. Lock-outs

Lock-out relates to the situation where a single connection or some connections dominate the buffer, preventing the other connections from using it. This occurrence is due to of determinism, synchronization or other timing effects. This condition can be caused by high data-rate or unresponsive flows.

ii. Full Queues

As the Drop Tail rule does not take any action before the queue size grows to its maximum, it allows the router queues inside the network to maintain a full or almost full state over long periods of time].

It is obvious that Lock-out gives unfair advantage to high data-rate and unresponsive flows over low data-rate and responsive flows.

1.6. Explicit Congestion Notification (ECN)

An extension to RED is to mark the IP header instead of dropping packets (when the average queue size is between minth and maxth, above maxth the packets are dropped as before). Cooperating end systems would then use this as a signal that the network is congested and slow down [9]. This is known as Explicit Congestion Notification (ECN). ECN, which has been proven to be a better way of delivering congestion information to the source host [9], has a better transfer delay for short-lived flows than packet drop schemes [10] [11]. In addition to reducing the number of timeouts for TCP flows, ECN mechanism does not require creation of additional transfer at the router and can be easily implemented in the data path of routers; it requires setting of a single bit.

ECN uses two bits in the IP header to carry information which indicates the router that the packet is ECN-capable or not (ECT1, ECT0 and Not-ECT codepoints). This information allows a congested router to mark the packet (by set EC codepoint) instead of dropping it as an indication of congestion. Also, ECN uses two bits in TCP header, ECE bit (ECE-Echo) for negotiating Ecn-capability and inform the sender about the congestion, CWR bit is used to enable the TCP receiver to determine when to stop setting the ECN-Echo or whether it has reduced its congestion window (Figure 1).

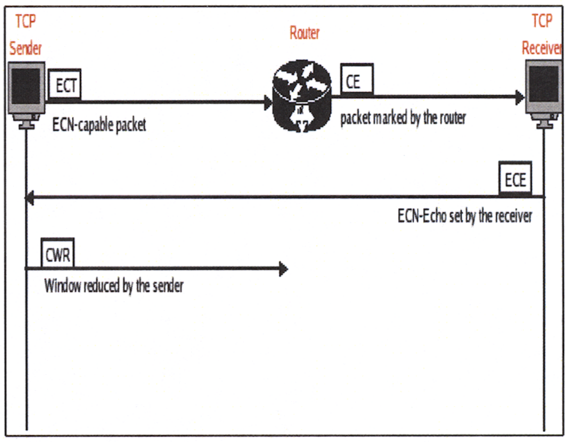

TCP Sender
Router
TCP Receiver
ECT
ECN-capable packet
CE
packet marked by the router
ECE
ECN-Echo set by the receiver
CWR
Window reduced by the sender

Figure 1. ECN in action

The performance of TCP congestion control is affected by the feedback delay involved in the congestion information reaching the source host from the bottleneck node [10].

Generally ECN implementations use the mark-tail strategy [9], i.e. when congestion is detected the router marks the incoming packets that have just entered the buffer of each router. However, a received marked packet can experience a queuing delay until all earlier buffered packets have been transmitted.

Mark-front strategy mechanism [12] was proposed to reduce the queuing delay by marking a packet at the time it is sent, which provide a faster congestion information delivery and reflecting the up-to-date congestion information.

Nevertheless, as a packet can still have a queuing delay in the buffer at each transit (intermediate nodes); the congestion information enclosed in an incoming packet cannot be directly transferred to its destination [13].

After estimating head marking (a modification of routers behaviour which allows to faster propagating congestion signals) on a single congested link and more complex link, Malowidzki [14] inferred that either the RED mechanism should be tuned or it should be replaced by a new AQM-type approach since the aggregate goodput of all TCP sources was the same for both ECN and non-ECN cases.

A study done by Kadhum and Hassan [15] proved that ECN improves the performances of short TCP sessions in RED network in case of having variation of files size and number of senders sharing one bottleneck; it is also shown that ECN increases the throughput and generally reduces the delay. The study infers that ECN is much more powerful than the simple packet drop indication.

While long-lived connections as the amount of multimedia traffic and ftp over the Internet have increased, TCP is still the dominant traffic force on the Internet, and is likely to remain so for the foreseeable future. The primary reason is the advent of the World Wide Web: the growing number of Internet users, the widespread availability of easy-to-use Web browsers, and the proliferation of Web sites with rich multimedia content combine to contribute to the exponential growth of Internet TCP traffic. Web caching and content distribution networks help to soften this impact [3, 14, 15], but the overall growth is still dramatic.

As a result, the performance and TCP behaviour is a main concern. Throughput is the significant performance measure for long-lived TCP connections. In this paper, we study the effect of ECN on long-lived TCP connections interacting with RED and then we compare ECN-RED with RED and Drop Tail gateways.

2. The Simulation Experiments Set-Up

We ran a series of simulation experiments to evaluate the performance of TCP using ECN-RED comparing with RED and Doptail and estimate the gain introduced by ECN, in terms of throughput with 10-30 TCP streams and one point of congestion. All tests employed TCP-Reno. We performed the simulations using ns-2 simulator [16] version 2.29. We use gnuplot [17] to produce the graphs. We run the simulations for the following scenarios with ECN-RED, RED, and Doptail respectively:

2.1. Scenario 1

The topology in Figure. 2 consists of 10 TCP sources feeding into a single congested link through one router to the destination (sink). The capacity of the bottle link is 0.7Mb with a 20ms delay to the destination. Other input links that join this link have 10Mb with delay that is chosen at random, uniformly distributed between 1ms and 5ms. The whole simulation duration is 50 seconds. We created 10 unlimited FTP connections throughout the simulations. These connections start randomly, the starting time is uniformly distributed between 0 and 7. We record the info of the evolution of the window size of all connection at granularity of 0.03sec.

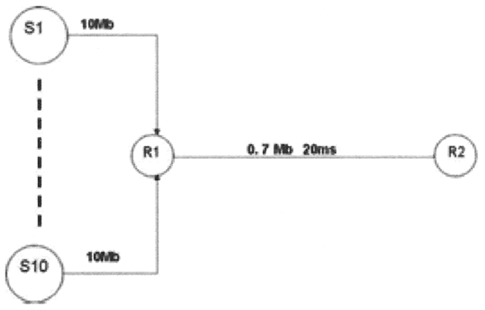


Figure 2. Simulation scenario of 10 users

The simulations use window size of 32 KB and packet size of 552 bytes. We use queue size of 100 packets at the gateway. For this scenario, we use RED with automatic parameter configuration. For the simulations in this paper, when using RED with ECN, the RED gateway set ECN bit to 1 in the packet header as an indication of congestion when the maximum threshold is reached in the buffer and set ECN bit to zero when using RED without ECN.

2.2. Scenario 2

The topology in Figure. 3 consists of 30 TCP sources sharing one congested link through one router to their destination (sink). The capacity of the bottle link is 0.7Mb with a 20ms delay to the destination. Other links have 10Mb with delay that is chosen at random, uniformly distributed between 1ms and 5ms. The whole simulation duration is 50 seconds. We use 30 unlimited FTP TCP connections that start randomly within the first 0.2 s. The timer granularity is 0.03sec and the window size of 32 KB. The packet size of 552 bytes and a queue size of 100 packets at the gateway.

In RED gateway, the minimum threshold for the average queue size is set to 60 packets, and the maximum threshold is set to 80 packets. The RED gateway drops all arriving packets when the average queue size exceeds the maximum threshold.

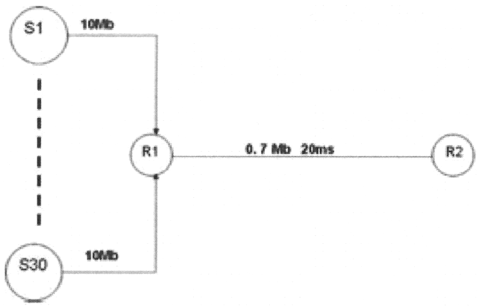


Figure 3. Simulation scenario of 30 users

3. Results and Results and Discussions

In this section, we exhibit the advantages of using ECN over long-lived TCP connections.

3.1. 10 users scenario

We discuss the results of the first set of tests, which were performed for 10 users (Figure. 2). In the use of Drop Tail gateway, we see from Figure. 4 (we plot windows of the first three connections for more clarity) and Figure. 5 that there is a high level of synchronization between the window sizes.

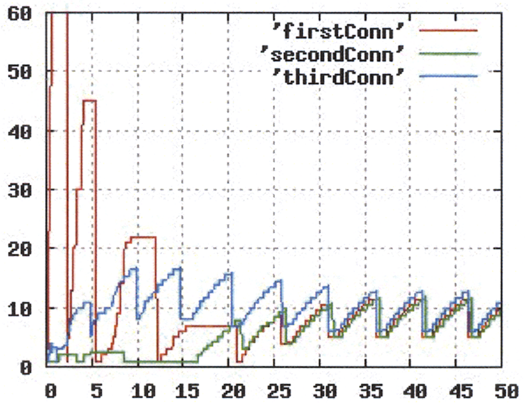


Figure 4. Window size of the first three connections

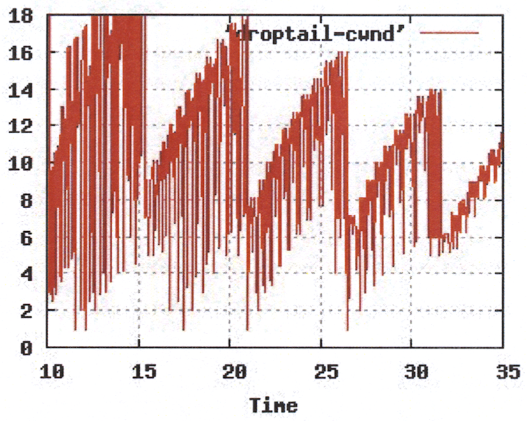


Figure 5. Window size of all TCP connections

They all lose packets at the same time. Moreover, we have high oscillations of the queue sizes that correspond to those of the windows, and the average queue size is around 75 packets as shown in Figure 6. This means that there is an additional average queuing delay which equals:

$$D_q = \frac{75*592*8}{700*10^3} = 507.42\,ms$$

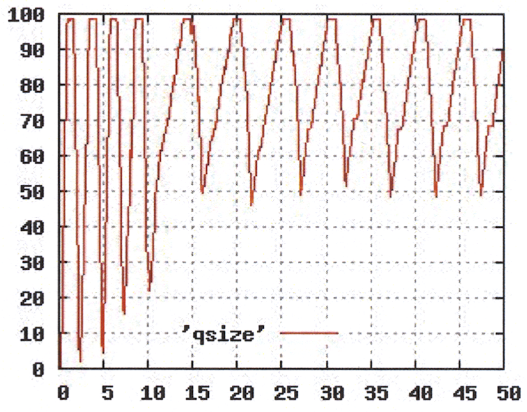


Figure 6. Queue size evolution

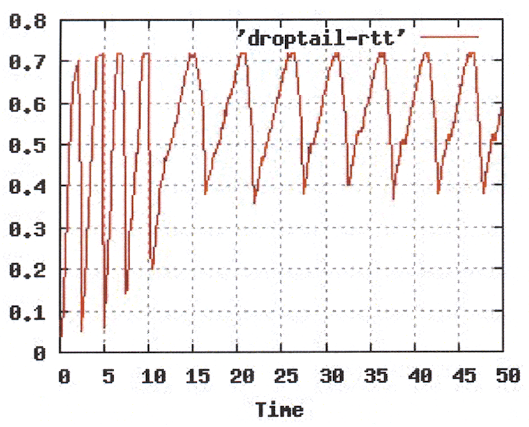


Figure 7. RTT fluctuation using Drop Tail

Figure. 7 shows that connections with Drop Tail notice a greater variability in RTTs.

In case of using RED, we see from Figure. 8 and Figure. 9 that there is no synchronization between the window sizes. We observe that instead of the large oscillations of the window size, we now get much faster and smaller variation in window size.

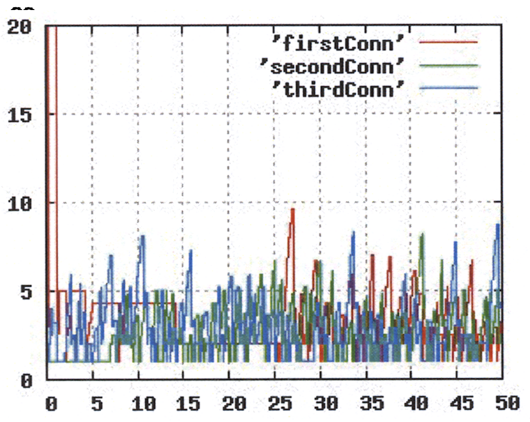


Figure 8. Window size of the first three connections

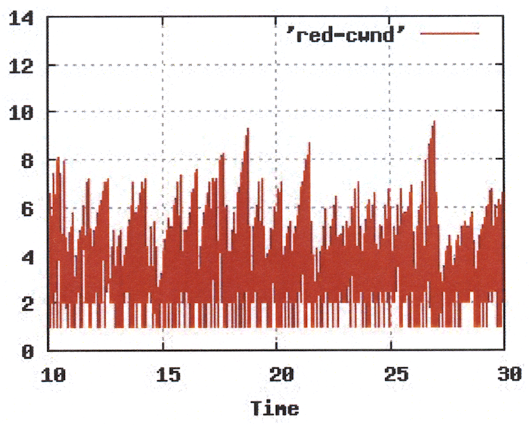


Figure 9. Window size of all TCP connections

As shown in Figure. 10, we have small oscillations in the queue size and the average queue size is much lower than in the Drop Tail case. It is around 8 (instead of 75 in the Drop Tail case). Thus, the average delay of the connections thus also smaller, it is:

$$D_q = \frac{8*592*8}{700*10^3} = 54.12\,ms$$

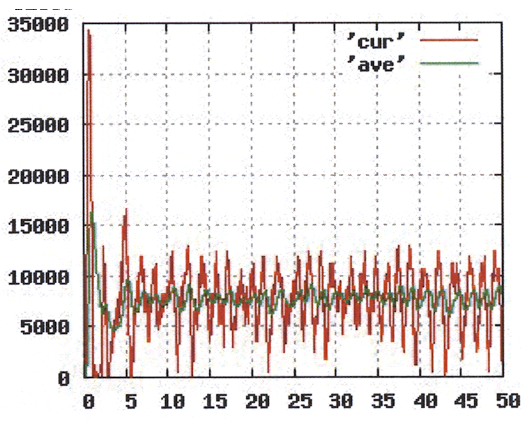


Figure 10. Current and Average queue size evolution in RED

RED did allow the queue to grow during the transient spike at the begging of the connection, which shows that short bursts are indeed not penalized with RED.

Figure. 11 indicates that connections with RED see a small variability in RTTs than Drop Tail case.

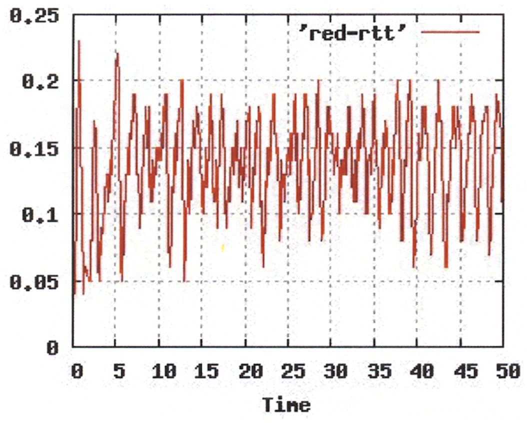


Figure 11. RTT fluctuation using RED

When we used ECN with RED, as shown in figure 12 and figure 13, we noticed that the ECN capable RED shows much improvement over the other schemes regarding synchronization and variation between the window sizes. ECN-RED can recover multiple drops gracefully and keep the window oscillating in a suitable saw tooth fashion, and the window swings are not drastic.

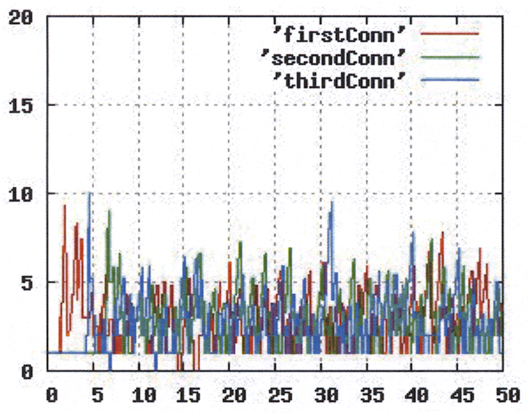


Figure 12. Window size of the first three connections

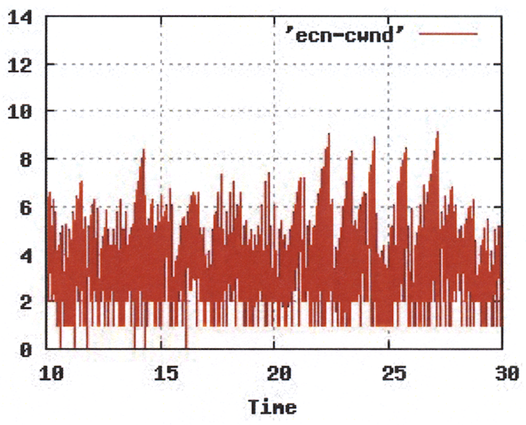


Figure 13. Window size of all TCP connections

From Figure 14, we can realize that the average queue size is lesser fairly than in the RED case. It is around 7 (instead of 8 in RED and 75 in the Drop Tail cases). Therefore, the average delay of the connections is smaller, it is:

$$D_q = \frac{7*592*8}{700*10^3} = 47.36ms$$

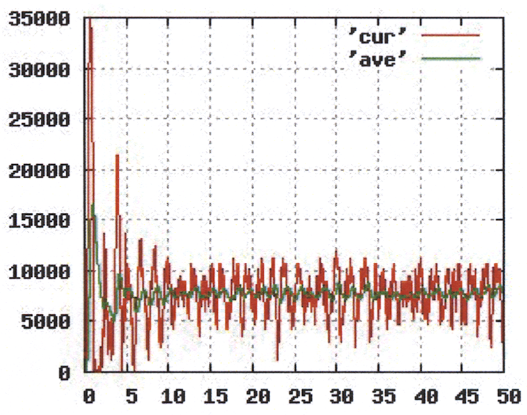


Figure 14. Current and Average queue size evolution in ECN-RED

Figure 15 illustrates that connections with RED observe a lower changeability in RTTs than RED case.

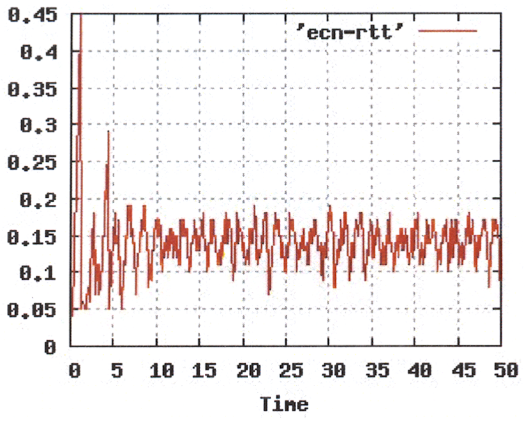


Figure 15. RTT fluctuation using ECN-RED

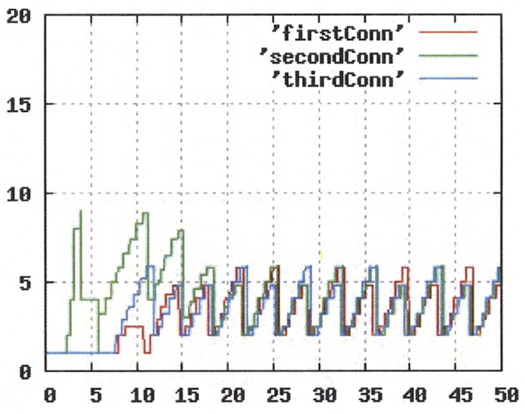


Figure 17. Window size of the first three connections

In figure 16, the results show that ECN provides better throughput than RED and Drop Tail due to the use of ECN as an early warning sign of congestion so that the sender will reduce its congestion window as soon as possible which make the network stable and that leads to better network utilization.

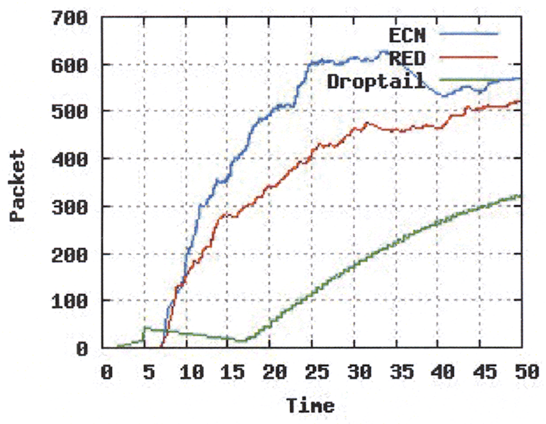


Figure 16. Throughput of ECN-RED, RED and Drop Tail

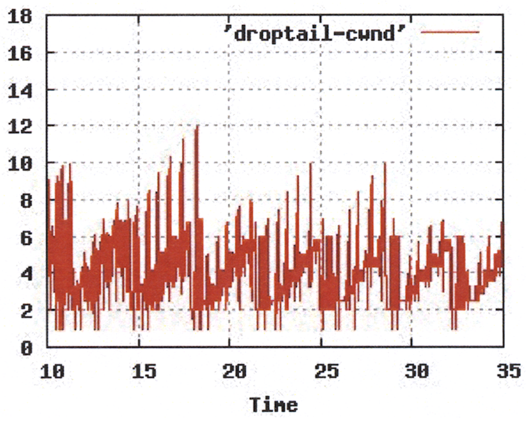


Figure 18. Window size of all TCP connections

3.2. 30 users scenario

We discuss the results of the second scenario, in which we simulate 30 TCP connections sharing one bottleneck (Figure. 3). In the use of Drop Tail gateway, as shown in figure 17 and figure 18, there is a high level of synchronization between the window sizes due to losing packets simultaneously, but the fluctuation is less than the Drop Tail-10 users' case. It is almost 6 (in stead of 12 (Figure. 4)).

The average queue size is around 80 packets as exhibited in Figure 19. In this case the average queuing delay equals:

$$D_q = \frac{80*592*8}{700*10^3} = 541.25ms$$

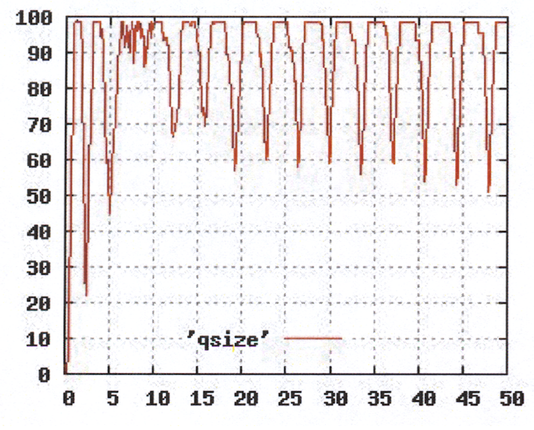


Figure 19. Queue size evolution

Figure 20 shows the variability in RTTs that connections see with the use of Drop Tail.

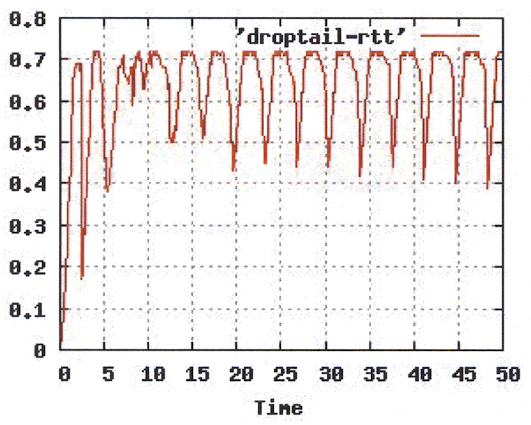


Figure 20. RTT fluctuation using Drop Tail

When we used RED in this scenario RED, we got improvement over Drop Tail and even over RED in first scenario. Figure 21 and Figure 22 illustrates that there is no synchronization between the window sizes and the oscillation of the window size is lower.

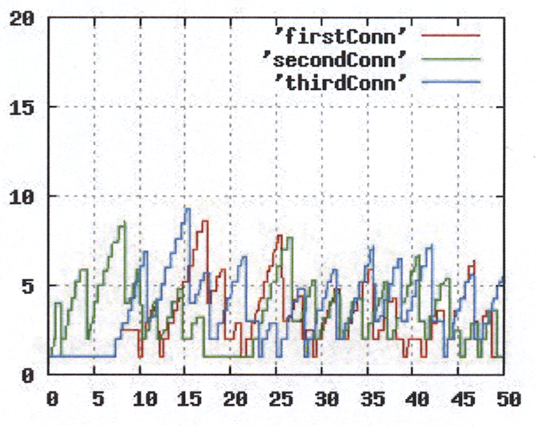


Figure 21. Window size of the first three connections

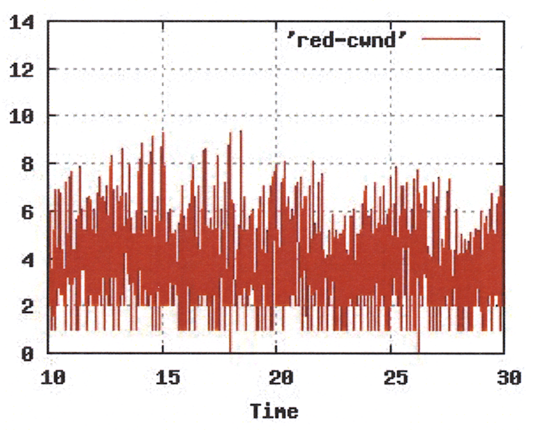


Figure 22. Window size of all TCP connections

In this scenario, we define our own parameters for RED rather than use the default ones. Recall that RED gateway drops all arriving packets when the average queue size exceeds the maximum threshold (which is 80 packets in this scenario). The resulting queue size process is given in Figure 23. We have small oscillations in the queue size and the average queue size is lesser than in the Drop Tail case. It is around 40 (instead of 80 in the Drop Tail case). The average delay of the connections is:

$$D_q = \frac{40*592*8}{700*10^3} = 270.62\,ms$$

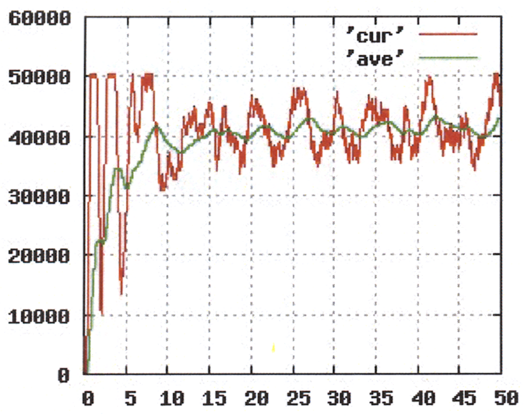


Figure 23. Current and Average queue size evolution in RED

Figure 24 shows the changeability in RTTs that connections see when using RED with fixed parameters for 30 TCP connections.

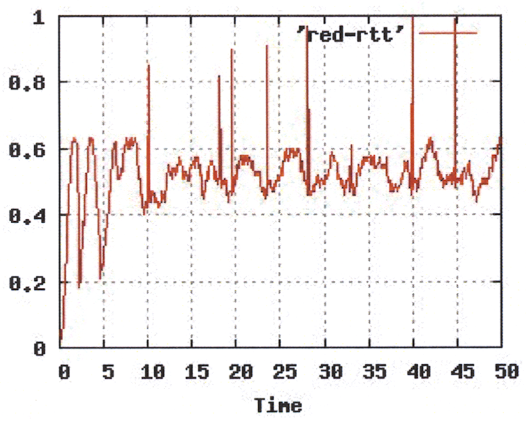
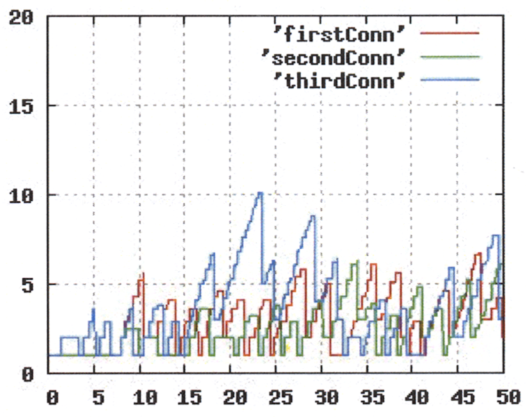


Figure 24. RTT fluctuation using RED

Figure 25 and Figure 26 depict the results gained when we used ECN-RED for 30 users. We observed that the ECN capable RED still shows fairly an improvement. The synchronization and variation between the window sizes are lower than in the RED and Drop Tail cases.



Figurer 25. Window size of the first three connections

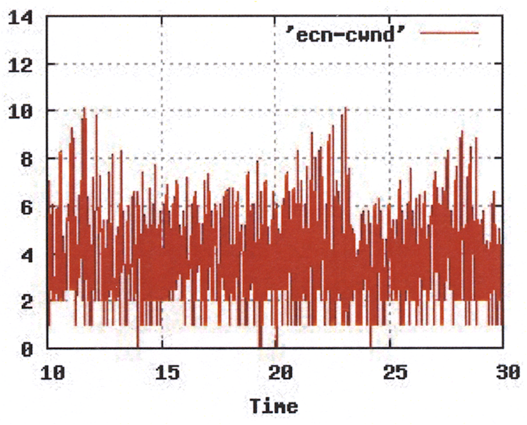


Figure 26. Window size of all TCP connections

As shown in Figure 27, the average queue size is almost the same as the one with RED. Due to the parameters that we chose, the lengths are kept around an average of 40 packets. Hence, the average delay of the connections is:

$$D_q = \frac{40*592*8}{700*10^3} = 270.62\,ms$$

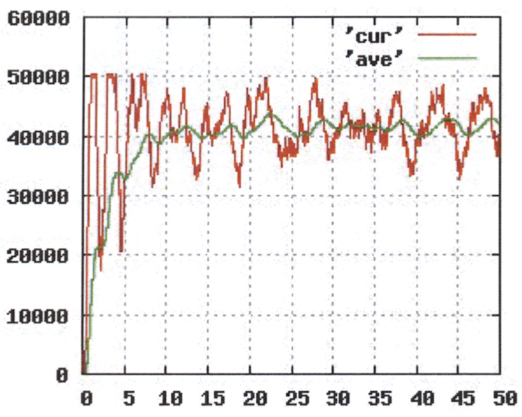


Figure 27. Current and Average queue size evolution in RED

Figure 28 demonstrates that connections when using ECN with RED observe a lower changeability in RTTs than RED case.

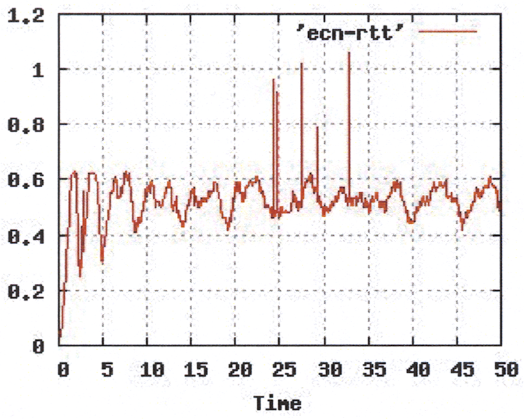


Figure 28. RTT fluctuation using ECN-RED

Figure 29 shows the throughput gained with the use of ECN-RED, RED, and Drop Tail. We can see that ECN provides better throughput than RED and Drop Tail even after increasing the number of users. This means, the use of ECN with RED reduces the number of packets dropped at the gateways due to well-managing of the packets in the queue and sending the congestion signal to the source quickly. Thus, the source halves its congestion window which will reduce packet injection rate that will directly affect the bottleneck link utilization.

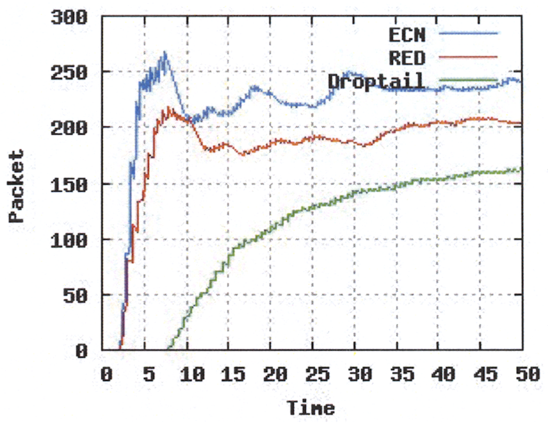


Figure 29. Throughput of ECN-RED, RED and Drop Tail

4. Conclusions

In this paper, we used ns 2.29 to investigate the effect of ECN on long-lived TCP connections for a number of users sharing one bottleneck. We created 2 scenarios and used Drop Tail, RED, and RED-ECN respectively in each scenario. And simulated each of them for five times to assure that the results are coherent. We compare the simulated performance of Drop Tail, RED, and ECN-RED routers. The results show that ECN-RED provides better throughput and less delay time than RED and Drop Tail for heterogeneous TCP flows, ECN is much more powerful than the simple packet drop indication. Our performance study shows that the throughput in all cases is better in the use of ECN-RED gateways.

As a future work, we are going to apply ECN over lossy wireless network, in which the receiver cannot distinguish between the packet loss due to congestion and the loss due to link errors, to see the effect of ECN on wireless network.

References

1. A. Durresi, M. Sridharan, C. Liu, M. Goyal, R. Jain, "Congestion Control using Multilevel Explicit Congestion Notification in Satellite Networks" , 2001.

2. Bensekas, D. and R. Gallager, Dora Networks, Englewood Cliffs, NJ: Pren- tice-Hall, 1987.

3. M. Schwartz, "Telecommunication Networks: Protocols, Modeling and Analysis", Addison-Wesley, 1987

4. Jacobson, V., "Congestion Avoidance and Control", Proceedings of SIGCOMM 1988, August 1988, pp. 314-329. (An updated version of this paper is available by anonymous ftp from ftp.ee.lbl.gov in congavoid.ps.Z.).

5. A. Kuzmanovic, S. Floyd, K. Ramakrishanan, "Adding Explicit Congestion Notification to

(ECN) to TCP's SYN/ACK Packets", IETF Draft, draft-ietf-tcpm-ecnsyn-00.txt, January 2006.

6. B. Braden et al., Recommendations on Queue Management and Congestion Avoidance in the Internet, "RFC 2309", April 1998.
7. Floyd S, Jacobson V. "Random Early Detection Gateways for Congestion Avoidance", IEEE/ACM Transaction on Networking, 1(4):397-413, September 1999.
8. B. Braden, et. al., "RFC2309: Recommendations on Queue Management and Con- gestion Avoidance in the Internet," April 1998
9. S. Floyd, K. Ramakrishanan, "A proposal to add Explicit Congestion Notification to IP", RFC 2481, January 1999.
10. S. Floyd, "TCP and Explicit Congestion Control", ACM Computer Communication review, V.24 N.5, p.10-23, October 1994.
11. J. Hadi, U. Ahmed, "Performance Evaluation of Explicit Congestion Notification (ECN) in IP Networks". RFC 2884, July 2000).
12. LIU, C., and JAIN R., "Improving explicit congestion notification with the mark-front strategy", Compur. Netw, 2001.35, (2-3), pp. 185-201.
13. Byung-Chul Kim and You-Ze Cho, "mark-relay strategy for explicit congestion notification in the internet", V.38 N.12, June 2002.
14. Marek Malowidzki, Simulation-based Study of ECN Performance in RED Networks, SPECTS'03, 2003.
15. M. Kadhum, S. Hassan, "The Effect of ECN on Short TCP Sessions", ICT-MICC, 115, May 2007.
16. http://www.isi.edu/nsnam/ns/doc/index.html
17. http://www.gnuplot.info