Enhanced Mechanisms for Explicit Congestion Notification in TCP/IP Networks

by

Frank Akujobi

B.Sc. (1999)

A thesis submitted to the
Faculty of Graduate Studies and Research
in partial fulfillment of the requirements for the degree of

Masters of Applied Science

at the

Ottawa-Carleton Institute for Electrical and Computer Engineering
Department of Systems and Computer Engineering

Carleton University
Ottawa, Ontario, Canada

December 19, 2002

©Frank Akujobi, 2002

National Library
of Canada

Acquisitions and
Bibliographic Services

395 Wellington Street
Ottawa ON K1A 0N4
Canada

Bibliothèque nationale
du Canada

Acquisitions et
services bibliographiques

395, rue Wellington
Ottawa ON K1A 0N4
Canada

Your file Votre référence

Our file Notre référence

The author has granted a non-exclusive licence allowing the National Library of Canada to reproduce, loan, distribute or sell copies of this thesis in microform, paper or electronic formats.

L'auteur a accordé une licence non exclusive permettant à la Bibliothèque nationale du Canada de reproduire, prêter, distribuer ou vendre des copies de cette thèse sous la forme de microfiche/film, de reproduction sur papier ou sur format électronique.

The author retains ownership of the copyright in this thesis. Neither the thesis nor substantial extracts from it may be printed or otherwise reproduced without the author's permission.

L'auteur conserve la propriété du droit d'auteur qui protège cette thèse. Ni la thèse ni des extraits substantiels de celle-ci ne doivent être imprimés ou autrement reproduits sans son autorisation.

0-612-79725-2

Canada

The undersigned recommend to the Faculty of Graduate Studies and Research
acceptance of the thesis

**Enhanced Mechanisms for Explicit Congestion Notification in
TCP/IP Networks**

Submitted by

Frank Akujobi
B.Sc.

In partial fulfillment of the requirements for the degree of
Masters of Applied Science

A handwritten signature in black ink, appearing to read 'D. H. ...', positioned above a horizontal dashed line.

Thesis Supervisor

A handwritten signature in black ink, appearing to read 'S. ...', positioned above a horizontal dashed line.

Chair, Department of Systems and Computer Engineering

Carleton University

Abstract

In this thesis the suitability of Backward Explicit Congestion Notification (BECN) for IP networks is evaluated extensively. The BECN mechanism has previously been used in non-IP networks, but there has been limited experimental investigation into the application of the BECN scheme as congestion control mechanism in TCP/IP networks. In this thesis, an enhanced algorithm for BECN which uses Internet Control Message Protocol (ICMP) Source Quenches (ISQs) for backward notification in IP networks is considered. A mathematical model is developed to analyze and predict the throughput and queue variation for TCP flows in Explicit Congestion Notification (ECN) and BECN capable networks while comparative performance evaluation of Random Early Detection (RED), ECN and our enhanced BECN mechanism is undertaken to validate the model. This thesis also undertakes performance evaluation of a suggested Multi-level Backward Explicit Congestion Notification (MECN) mechanism in comparison to the ECN and BECN mechanisms with long-lived TCP traffic.

The impact of two-way traffic and a lossy reverse path on performance of the ECN and BECN mechanisms is extensively investigated and results showed that two-way traffic and loss on the reverse path can result in significant queue fluctuations for BECN due to loss of ISQ messages. A lossy reverse path did not significantly affect queue fluctuations for ECN due to reliable delivery of congestion notification via ECN Echo ACKs (ECE-ACKs).

Deductions from these results led to the proposal and evaluation of a novel algorithm (ECN+BECN mechanism) which combines the merits of the ECN and BECN mechanisms. Experimental results show that the combined mechanism benefits from BECN's early notification under heavy congestion and ECN's reliable delivery of congestion notification. It is also shown that the ECN+BECN scheme can significantly reduce the ICMP Source Quench reverse traffic in a network compared to the BECN scheme.

Acknowledgments

I am indebted to my parents and entire family for their encouragements and enduring support throughout the duration of my research work and studies.

I am grateful to my thesis supervisor, Professor Ioannis Lambadaris for his guidance and useful insight which contributed significantly to the success of this research work. His indepth knowledge of issues around congestion control in IP networks proved very helpful during this work.

I am equally grateful to Nabil Seddigh and Biswajit Nandy of Tropic Networks Inc. in Ottawa for their useful ideas and suggestions which led to the detailed simulation experiments carried out.

I also wish to thank Rupinder Makkar and my colleagues at the Broadband Networks Laboratory and Advanced Optical Network Laboratory, Carleton University for their moral support, encouragements and help during this research work.

Finally, I gratefully acknowledge the financial support of Nortel Networks, Communications and Information Technology Ontario (CITO), Mathematics of Information Technology and Complex Systems (MITACS) and the government of Ontario towards the completion of this degree.

Contents

Abstract	ii
Acknowledgments	iii
Table of Contents	iv
List of Figures	viii
List of Tables	xii
List of Abbreviations	xiii
1 Introduction	1
1.1 Background and Motivation	1
1.2 Objective of the Thesis	3
1.3 Thesis Outline	4
1.4 Thesis Contributions	5
2 Congestion Control in TCP/IP Networks	7
2.1 Random Early Detection	8
2.2 Explicit Congestion Notification	9
2.2.1 Functional Description of ECN	9

3	Backward ECN (BECN) for Congestion Control in TCP/IP Networks	12
3.1	Introduction	12
3.2	Issues with use of BECN for TCP/IP networks	14
3.2.1	Concerns with use of BECN	14
3.2.2	Benefits of using BECN	15
3.3	Enhanced BECN algorithm	16
3.3.1	Behavior of a BECN-capable router.	16
3.3.2	Behavior of BECN-capable TCP end hosts	17
3.4	Mathematical model and predictions for TCP throughput and queue dynamics in ECN/BECN-capable networks	19
3.4.1	TCP throughput for ECN and BECN capable connections	19
3.4.2	RED-ECN/BECN Feedback Control System	23
3.4.3	Variance of average queue, $\bar{q}(p)$	24
3.5	Simulation Setup	25
3.5.1	Simulation Topology	25
3.5.2	Queue parameters	26
3.5.3	Traffic sources	27
3.5.4	Performance Metrics	28
3.6	Experiments and Results	29
3.6.1	Competing long-lived BECN/ECN flows with plain TCP flows - [Fig. 3.8 and Fig. 3.9]	31
3.6.2	Homogeneous long-lived flows - [Fig. 3.10 - Fig. 3.14]	32
3.6.3	Short-lived web transfers - [Fig. 3.15 - Fig. 3.18]	33
3.6.4	Large round-trip times (RTT) - [Fig. 3.19 - Fig. 3.28]	37
3.6.5	ISQ reverse traffic analysis - [Fig. 3.29]	39
3.7	Conclusion and Future work	40

4	Multi-level BECN (MECN) mechanism	42
4.1	Introduction	42
4.2	Suggested MECN algorithm	43
4.2.1	Behavior of a MECN-capable router	43
4.2.2	Behavior of MECN-capable TCP end hosts	45
4.3	Simulation Setup	46
4.3.1	Simulation Topology and parameters	46
4.3.2	Performance Metrics	46
4.4	Experiments and Results	46
4.4.1	Homogeneous long-lived flows - [Fig. 4.1 - Fig. 4.3]	47
4.4.2	Large round-trip times (RTT) - [Fig. 4.4 - Fig. 4.7]	48
4.5	Conclusion	49
5	Impact of two-way traffic and a lossy reverse path on the behavior of BECN, ECN and RED	51
5.1	Introduction	51
5.2	Simulation Setup	52
5.2.1	Simulation Topology	53
5.2.2	Queue parameters	55
5.2.3	Traffic sources	55
5.2.4	Performance Metrics	55
5.3	Experiments and Results	56
5.3.1	Experiment with small round-trip time (RTT) - [Fig. 5.3 - Fig. 5.13]	56
5.3.2	Experiment with large round-trip times - [Fig. 5.13 - Fig. 5.20]	61
5.3.3	Lossy reverse path - [Fig. 5.21 - Fig. 5.32]	63
5.4	Conclusion and Future work	67

6	ECN+BECN mechanism for Explicit Congestion Notification	69
6.0.1	Introduction and motivation	69
6.1	Proposed ECN+BECN algorithm	70
6.1.1	Behavior of an ECN+BECN-capable router	71
6.1.2	Behavior of ECN+BECN-capable TCP end hosts	72
6.2	Simulation Setup	74
6.2.1	Simulation Topology	74
6.2.2	Performance Metrics	74
6.3	Experiments and Results	74
6.3.1	Competing long-lived ECN+BECN flows with plain TCP flows - [Fig. 6.1 - Fig. 6.2]	75
6.3.2	Homogeneous long-lived flows - [Fig. 6.3 - Fig. 6.6]	76
6.3.3	Experiment with two-way traffic - [Fig. 6.7 - Fig. 6.8]	77
6.3.4	Lossy reverse path - [Fig. 6.9 - Fig. 6.13]	79
6.3.5	Short-lived web transfers - [Fig. 6.14 - Fig. 6.18]	82
6.4	Conclusion and Future work	84
7	Summarized conclusion and Future Work	86

List of Figures

2.1	RED drop probability as a function of average queue size	9
3.1	Evolution of window size for Explicit Congestion Notifications	20
3.2	Packets sent during an EN period	21
3.3	Average throughput ($\sigma = 0.5$)	25
3.4	Average throughput ($\sigma = 0.8$)	25
3.5	Average queue ($\sigma = 0.5$)	26
3.6	Average queue ($\sigma = 0.8$)	26
3.7	Network Topology 1	27
3.8	Goodput gain over TCP	30
3.9	Percentage loss	30
3.10	Goodput - Homogeneous flows	30
3.11	Loss - Homogeneous flows	30
3.12	Bottlebeck utilization	33
3.13	Queue - 15 ECN FTP	33
3.14	Queue - 15 BECN FTP	34
3.15	Average object transfer delay	34
3.16	HTTP data packet loss	34
3.17	Queue - ECN web + 15 FTP	34
3.18	Queue - BECN web + 15 FTP	35

3.19	Goodput - Homogeneous flows	35
3.20	Loss - Homogeneous flows	35
3.21	Bottlebeck utilization	35
3.22	45 Plain TCP FTP (large RTT)	36
3.23	45 ECN FTP (large RTT)	36
3.24	45 BECN FTP (large RTT)	37
3.25	Average object transfer delay	37
3.26	HTTP data packet loss	38
3.27	25 ECN FTP + web (RTT)	38
3.28	25 BECN FTP + web (RTT)	39
3.29	ISQ reverse traffic	39
4.1	Goodput - Homogeneous flows	47
4.2	Loss - Homogeneous flows	47
4.3	Bottlebeck utilization	48
4.4	Goodput - Homogeneous flows	48
4.5	Loss - Homogeneous flows	49
4.6	Bottlebeck utilization	49
4.7	45 MECN FTP flows (large RTT) - 40KB threshold	50
5.1	Network Topology 2	54
5.2	Network Topology 3	54
5.3	Average goodput	57
5.4	Average congestion window	57
5.5	Mean of actual queue size	58
5.6	Variance of actual queue size	58
5.7	Bottleneck utilization	58

5.8	Data packet loss	58
5.9	25 background TCP flows	59
5.10	25 background ECN flows	59
5.11	25 background BECN flows	59
5.12	ISQ loss	59
5.13	Percentage of ISQ on reverse path	61
5.14	Average goodput	61
5.15	Data packet loss	62
5.16	Congestion window	62
5.17	Bottleneck utilization	62
5.18	30 background RED flows	62
5.19	30 background ECN flows	63
5.20	30 background BECN flows	63
5.21	Average goodput	64
5.22	Loss on reverse path-BECN	64
5.23	Loss on reverse path-ECN	65
5.24	Mean of average queue	65
5.25	Variance of average queue	65
5.26	Average congestion window	65
5.27	Effect of ISQ loss on BECN window - 10% lossrate	66
5.28	Effect of ISQ loss on BECN queue - 10% lossrate	66
5.29	Effect of ECE ACK loss on ECN window - 10% lossrate	66
5.30	Effect of ECE ACK loss on ECN queue - 10% lossrate	66
5.31	Effect of ACK loss on TCP window - 10% lossrate	67
5.32	Effect of ACK loss on TCP queue - 10% lossrate	67
6.1	Average goodput gain over TCP	76

6.2	Percentage loss	76
6.3	Bottlebeck utilization	77
6.4	ISQ reverse traffic	77
6.5	Queue - 15 ECN FTP	78
6.6	Queue - 15 ECN+BECN (20KB) FTP	78
6.7	Variance of actual queue size - two-way traffic	78
6.8	ISQ loss - two-way traffic	78
6.9	Effect of ISQ loss on BECN window - 10% lossrate	80
6.10	Effect of ISQ loss on BECN queue - 10% lossrate	80
6.11	Effect of ISQ loss on ECN+BECN window - 10% lossrate	80
6.12	Effect of ISQ loss on ECN+BECN queue - 10% lossrate	80
6.13	Average goodput - lossy reverse path	81
6.14	Average object transfer delay	81
6.15	HTTP packet loss	83
6.16	Fairness - Object transfer delay	83
6.17	Mean of queue - web transfers	84
6.18	Variance of queue - web transfers	84

List of Tables

3.1	WEB Traffic Parameters	28
6.1	WEB Traffic Parameters	82

List of Abbreviations

ACKs	Acknowledgements
BECN	Backward Explicit Congestion Notification
CE bit	Congestion Experienced bit
CE packet	Packet with Congestion Experienced bit set
CITO	Communications and Information Technology Ontario
<i>cwnd</i>	Congestion Experienced bit
CWR bit	Congestion Window Reduced bit
CWR packet	Packet with Congestion Window Reduced bit set
ECE-ACKs	ECN Echo ACKs
ECN	Explicit Congestion Notification
ECN+BECN	Combined ECN and BECN mechanism
ECT	ECN Capable Transport
ENP	Explicit Notification period
FIFO	First-In-First-Out
FTP	File Transfer Protocol
HTTP	Hyper Text Transfer Protocol
ICMP	Internet Control Message Protocol
IPV4	IP version 4
IPV6	IP version 6

TOS	Type of Service
TC	Traffic Class
ISQs	ICMP Source Quenches
<i>maxth</i>	Maximum RED threshold
MECN	Multi-level Backward Explicit Congestion Notification
<i>Midthresh</i>	MECN threshold
<i>minth</i>	Minimum RED threshold
MITACS	Mathematics of Information Technology and Complex Systems
NS	Network Simulator
QoS	Quality of Service
RED	Random Early Detection
RTO	Retransmission Timeout
RTT	Round-Trip Time
SQuID	Source Quench Introduced Delay
<i>ssthresh</i>	Slow Start threshold
TDA	Triple Duplicate Acknowledgements
TO	Timeouts

Chapter 1

Introduction

1.1 Background and Motivation

Over the years, the Internet has evolved into a global heterogeneous network that is used for mission-critical and leisure purposes. Its successful evolution is in part due to flow and congestion avoidance mechanisms that aim to avoid “congestion collapse” [4][8]. Sender flow control is achieved with TCP conservatively injecting packets into the network based on feedback of the congestion state of the network. The TCP sender considers packet loss as an indication of congestion and relies on three duplicate acknowledgements or a retransmission timeout (RTO) to confirm occurrence of a loss. The sender responds to this feedback by limiting its packet injection rate into the network.

Recently, Random Early Detection (RED) [6] has been recommended [14] as the active queue management scheme for use on the Internet. More recently, Explicit Congestion Notification (ECN) was proposed for TCP/IP networks as a means of explicitly notifying end-hosts of network congestion by marking, instead of dropping packets [11]. Recent studies show that RED with ECN support gives definite improvement in time delays for interactive traffic over packet drop schemes [9][19].

In this thesis, the use of a Backward Explicit Congestion Notification (BECN) mechanism to inform the sender of the congestion situation in the network is examined. This mechanism uses Internet Control Message Protocol (ICMP) Source Quenches for backward notification. The ICMP Source Quench (ISQ) messaging was originally standardized as the mechanism of choice for notifying an end host of network congestion [1]. However, while the messaging for ICMP Source Quenches (ISQs) was defined, the conditions for ISQ generation at the router and the appropriate reaction at the end host were not implemented in a standardized way nor could they be considered mature algorithms. RFC 1812 [3] deprecates generation of ISQ messages from a router but specifies that a router that originates ISQs must be able to limit the rate at which they are generated. Floyd in [9], gives due consideration to ISQs as a mechanism for explicitly notifying the TCP sender about congestion. However, for a number of reasons explained in subsequent sections, ISQs were considered unsuitable for congestion notification. Nevertheless, we believe that in particular network scenarios BECN might have an advantage over other congestion notification schemes.

An enhancement to an originally proposed BECN algorithm [7] is provided in this thesis, and performance evaluation of the enhanced BECN is carried out with FTP bulk transfers and HTTP-like workloads.

To achieve effective use of network resources and improved performance in IP networks the concept of multi-level backward congestion notification with the use of ICMP source quench packets was proposed in [7], however authors in [7] did not show experimental evaluation of the proposed mechanism. This thesis evaluates the performance of a modified scheme for multi-level BECN (MECN) which also uses ICMP Source Quenches to carry information on the severity of congestion back to the sender so that the sender responds in accordance with the level of congestion in the network.

There has been limited experimental investigation into the impact of both packet

loss on the reverse path and two-way traffic on the performance of the ECN and BECN mechanisms. Two-way traffic is a more realistic form of traffic as obtained on the Internet and studies [25] show that impact of loss of ACKs on the reverse path of data flow cannot be ignored in wireless networks. In [10], concerns were raised as to how the loss of BECN ISQs on a congested reverse path would impact the performance of BECN. Hence a study of the impact of two-way traffic and a lossy reverse path on TCP traffic becomes important. This thesis, using simulations investigates the impact of two-way traffic and a lossy reverse path on the performance of BECN in comparison to the ECN and RED mechanisms.

Based on deductions from results obtained in the experimental study of the congestion notification mechanisms, this thesis proposes a novel mechanism (ECN+BECN mechanism) which combines the merits of both the ECN and BECN mechanisms for congestion notification in TCP/IP networks.

1.2 Objective of the Thesis

The objective of this thesis is to provide an extensive study of congestion notification schemes for congestion control in a predominantly TCP/IP network. The work includes modifications to existing algorithms and a proposal for novel mechanisms for explicit congestion notification which based on our experimental results offer improved performance. The outcome from this work is expected to serve as a relevant basis for recommendations to the IETF Working Group working on Internet Traffic Engineering.

Specifically this thesis accomplishes the following:

- Development of a mathematical model to analyse throughput and queue dynamics for TCP flows in ECN and BECN capable networks.
- Experimental simulations that validate the predictions based on the developed

mathematical model.

- Performance evaluation of the RED, ECN and BECN mechanisms with both long and short-lived TCP traffic using performance metrics such as average TCP goodput, percentage packet loss, bottleneck link utilization, web object average transfer delay, fairness in object transfer delay.
- Performance evaluation of the multi-level BECN (MECN) mechanism in comparison the ECN and BECN mechanisms using both long-lived and short-lived TCP traffic.
- Investigation of the impact of two-way traffic and a lossy reverse path on the behavior of a TCP connection in a RED, ECN, or BECN enabled network.
- Proposal and performance analysis of a novel mechanism (ECN+BECN mechanism) that combines the salient features of both ECN and BECN mechanisms.

1.3 Thesis Outline

Rest of the thesis is organized as follows:

Chapter 2: gives an introduction into the Random early detection (RED) and the Explicit Congestion Notification (ECN) scheme.

Chapter 3: presents a developed mathematical model for TCP throughput and queue dynamics in ECN and BECN networks. A comparative experimental evaluation of the Explicit Congestion Notification (ECN) scheme and an enhanced Backward ECN (BECN) scheme is also done. Details of the simulation environments used, traffic models and topologies with relevant justifications is provided. The performance metrics used for evaluating the algorithms are also defined.

Chapter 4: A multi-level BECN (MECN) scheme is suggested and comparative performance evaluation of this mechanism is done.

Chapter 5: contains further investigation into the behavior of RED, ECN and BECN in the presence of two-way traffic and a lossy reverse path. Justification for these experiments are made along with detailed experimental evaluation.

Chapter 6: presents a proposed novel algorithm for congestion notification which combines the benefits of both ECN and BECN mechanisms. Using simulations this thesis shows that the novel mechanism can offer improved performance for TCP connections in comparison to both the ECN and BECN schemes.

Chapter 7: discusses the findings in previous chapters and provides a summarized conclusion for this work. Ideas for possible future work in this area are also presented.

1.4 Thesis Contributions

The following are the main contributions of this thesis:

- Extensive experimental comparative evaluation of the RED, ECN, BECN and MECN mechanism using topologies and traffic models that have not been used for such evaluations in the past.
- Findings from this work resulted in a paper “BECN for Congestion Control in TCP/IP Networks - Study and Comparative Evaluation” IEEE/Globecom 2002, Taipei, Taiwan.
- Study into the impact of two-way traffic and a lossy reverse path on the behavior of the RED, ECN and BECN mechanisms. Such a comparative study has not been carried out in the past.

- Proposal of a novel mechanism for explicit congestion notification which combines the salient features of both the ECN and BECN mechanisms. Simulations were used to show that the mechanism results in improved performance for TCP connections for the scenarios considered.
- Analytic and experimental results with the proposed novel mechanism resulted in a second paper “Congestion Control in TCP/IP networks: A combined ECN and BECN approach” submitted to the IEEE INFOCOM Conference 2003, San Francisco, California, USA.

Chapter 2

Congestion Control in TCP/IP Networks

In complex networks, congestion and traffic control is a vital issue which if not properly handled can lead to “congestion collapse” [4][8]. Most networks therefore have congestion control and avoidance mechanisms that aim to ensure proper management of congestion.

In IP networks, congestion management is handled by the transport layer protocol running over IP. TCP is the predominant transport protocol today [32] and infers congestion in the network by detecting packet drops within the network [35]. Congestion avoidance algorithms [8][14] for TCP are then invoked to alleviate the congestion situation. The TCP protocol permits a sending host to inject packets into the network at a high rate (slow start state) until it detects a packet loss. The sending host then “backs off” by moving into a “congestion avoidance state” where it reduces its rate of injecting packets into the network. TCP infers a packet loss by the receipt of 3 duplicate acknowledgements (ACKs) or by the elapse of a pre-determined TCP retransmission timeout period.

Studies [14] have shown that while the TCP congestion avoidance mechanisms are

necessary, they are not sufficient to provide good service in all circumstances, as there is a limit to how much control can be accomplished by endpoint hosts only. Some mechanism within the network (essentially the network routers) is required to complement TCP's end-to-end congestion avoidance mechanism.

Queue management algorithms within routers is one vital class of router algorithms related to congestion control [14], as they manage the length of packets in the router queues by dropping packets when appropriate. Traditionally, a router reacts to congestion by dropping an arriving packet in the absence of buffer space. This is referred to as the Tail Drop queue management algorithm and has a number of inherent drawbacks [14]. Drawbacks with the Tail Drop mechanism has led to the introduction of active queue management schemes in routers which provide a number of advantages for responsive flows [14]. One such scheme is the Random Early Detection (RED) queue management scheme [6].

2.1 Random Early Detection

The Random Early Detection (RED) active queue management scheme detects congestion before it actually occurs based on a running average queue size and drops packets probabilistically before the queue actually overflows thus notifying TCP sending hosts of congestion before deep congestion occurs. RED operates by maintaining two configurable queue thresholds - minimum threshold (*minth*) and maximum threshold (*maxth*). As the average queue length varies between *minth* and *maxth*, the probability of dropping an arriving packet increases linearly from 0% towards a configurable maximum drop probability *maxp* (Fig. 2.1). Beyond *maxth*, the drop probability is 100%. This pattern of dropping packets ensures that the probability of dropping packets from a particular flow is proportional to the bandwidth occupied by that flow. Hence more aggressive TCP hosts are caused to invoke their congestion avoidance algorithms that will ease conges-

Figure 2.1: RED drop probability as a function of average queue size

tion at the gateway. The RED active queue management scheme addresses the problems of the tail drop mechanisms and has been recommended as the default mechanism for managing queue length in FIFO queues [14].

2.2 Explicit Congestion Notification

Explicit Congestion Notification is an extension proposed to RED which probabilistically marks a packet instead of dropping it when the average queue size is between *minth* and *maxth* [11]. When the average queue size exceeds the *maxth* arriving packets are dropped with a 100% probability just as in RED. This mechanism significantly reduces packet drops within the network compared to the RED mechanism and studies [9][19] have shown that it offers definite improvement in delays for short-lived TCP traffic connections like telnet applications.

2.2.1 Functional Description of ECN

The ECN mechanism requires the use of some fields within the IP and TCP headers of a TCP packet to achieve congestion notification. Bit 6 and 7 in the IPV4 TOS field or

bit 10 and 11 in the IPV6 TC field are designated as the ECN fields. For IPV4, bit 6 is designated as the ECN Capable Transport (ECT) bit which is set to indicate ECN capability while bit 7 is designated as the Congestion Experienced (CE) bit which is set probabilistically by a router to indicate a congestion situation. Also, two new flags in the Reserved field of the TCP header are used. Bit 9 in the Reserved field is designated as the ECN-Echo (ECE) flag while bit 8 is designated as the Congestion Window Reduced (CWR) flag. The usage of the ECN fields and the functional behavior of ECN-capable TCP hosts in a congestion situation are explained as follows:

1. In the connection setup phase, the source and the destination TCP hosts exchange information about their desire and/or capability to use ECN. This is done by setting both the ECN-Echo flag and the CWR flag in the SYN packet of the initial connection phase by the sender. On receipt of this SYN packet, the receiver will set the ECN-Echo flag in the SYN-ACK response. Once this agreement has been reached, the sender will set the ECT bit in the IP header of subsequent data packets for that flow injected into the network, to indicate to the network that it is capable and willing to participate in the ECN congestion control mechanism.
2. When a router experiences congestion it sets the CE bit in the IP header of an arriving packet (CE packet) if the ECT bit is already set on the packet. When such a packet reaches the receiver, the receiver responds by setting the ECN-Echo flag in TCP header of the next outgoing ACK (ECE-ACK) for the flow. The receiver continues to do this in subsequent ACKs till it receives from the sender an indication that it (the sender) has responded to the congestion notification.
3. Upon receipt of the ECE-ACK, the TCP sender triggers its congestion avoidance algorithm by halving its congestion window (*cwnd*) and updating its *ssthresh*. The sender then sets the CWR bit on the next outgoing packet (CWR packet) to inform

the receiver that it has reacted to the receiver's notification of congestion. Upon receipt of the CWR packet the receiver halts the sending of congestion notifications (ECE-ACKs) to the sender, but continues to send normal ACKs for packets received according to the TCP mechanism.

Chapter 3

Backward ECN (BECN) for Congestion Control in TCP/IP Networks

3.1 Introduction

In this chapter an enhanced algorithm for Backward Explicit Congestion Notification (BECN) which uses Internet Control Message Protocol (ICMP) Source Quenches (ISQs) for backward congestion notification in IP networks is proposed and its suitability for congestion control in IP networks is evaluated. A mathematical model is developed to analyze throughput and queue variation for TCP flows in ECN and BECN capable networks while comparative evaluation of the RED, ECN and BECN mechanisms is done using simulations to better understand the mechanisms as well as verify the developed mathematical models.

The ICMP Source Quench (ISQ) message format was originally defined in [1]. In [2] it is documented that a disadvantage of the ISQ mechanism is that its details are

discretionary stating that it is impossible for the end-system user to be sure about the conditions under which the ISQ was generated since ISQs did not carry this information themselves. RFC 896 [4] discusses in general terms approaches for generating ISQs and reacting to them. Among the approaches is one that considers generating ISQs adaptively before the queue is full. RFC 1016 [5] described Source Quench Introduced Delay (SQuID) where ISQs were to be generated based on threshold levels of the physical queue in the router. Packets are clocked by the sender based on inter-arrival times adjusted in response to the ISQ arrival rate. More recently, [15] explored the use of Source Quenches for controlling unresponsive sources that inject more than their fair share bandwidth into the network.

There has also been proposals for using backward notification within ATM networks [16]. In [17] it was affirmed that though the indiscriminate use of backward notification can cause problems in ATM networks, it might help reduce the feedback time for paths with large delays.

The proposal for BECN in IP networks [7] provided guidelines for generating ISQs and responding to them in a TCP/IP network. According to [7], a BECN TCP sender responds to ISQ congestion notification by halving its TCP window. During the research work for this thesis it was observed through simulation that with this proposal the BECN sender also starts increasing its window upon receipt of the next ACK after a window reduction. The immediate reaction of increasing rate of injecting packets into the network makes the proposed BECN algorithm unduly aggressive. In this thesis, a modification to the BECN algorithm is suggested and performance of the enhanced BECN algorithm is evaluated in comparison to the ECN and RED schemes.

3.2 Issues with use of BECN for TCP/IP networks

In this section, issues that have been raised concerning the use of the BECN mechanism for congestion control in TCP/IP networks are discussed.

3.2.1 Concerns with use of BECN

1. There is concern that no standard algorithm exists for response of a TCP source to an ISQ nor are the conditions for ISQ generation well defined. BECN algorithm in [7] addresses this by defining conditions for ISQ generation and TCP source response. More details are given in subsequent sections.
2. There is the issue of how reliable the use of BECN ISQs is for congestion notification especially in scenarios where BECN ISQ messages are lost on the reverse path [10]. We point out that during persistent congestion this condition is no worse than loss of an ECN-Echo ACK [11] in the case of ECN since ISQs continue to be generated irrespective of whether a previous one was sent. The BECN sender only responds once per window. For very temporal congestion situations more work is required to evaluate the impact of loss of an ISQ message.
3. In [10], the amount of extra reverse network traffic generated by the BECN ISQ messages was a concern. This was a valid concern that existed with drop-tail buffer management as during times of congestion lots of ISQs are generated. However in the BECN proposal [7], BECN ISQs are generated only when the computed RED probability requires a packet dropping or marking. Results in this chapter show that for the scenarios considered, the contribution of ISQs to reverse traffic in a BECN capable network does not significantly impact performance of BECN.
4. There is also a real concern about how router performance would be degraded due to the overhead of generating packets on the reverse path. While this thesis does

not do a direct quantitative evaluation of how reverse packet generation impacts router performance, chapter six suggests a mechanism that further reduces the percentage of reverse packets generated with the BECN mechanism.

5. It has been argued that BECN is non-generic for multicasting as there can be receiver or sender based congestion control [10]. However, it has also been pointed out that with sender-based multicast congestion control, the BECN feedback mechanism is more scalable than earlier proposals for feedback control in multicast environments since it is provided by the router not by all the receivers in a multicast session [12].
6. The issue of ATM's "beat-down" problem has also been raised with regards to IP BECN. It is worth pointing out that this is really a problem of traffic that passes through multiple hops versus traffic that passes through fewer hops. This issue is present even for TCP flows in a RED or ECN enabled network and was dealt with in [13]. As a result, it is not considered in this thesis.

The concerns with IP BECN need further investigation to better understand them. To date, there has been limited quantitative investigation into the performance of IP BECN in comparison with other IP Congestion control mechanisms.

3.2.2 Benefits of using BECN

1. BECN enjoys all the advantages of ECN over TCP with RED. This stems from the fact that for both ECN and BECN, packets are marked probabilistically and not dropped. Such advantages include lower loss rates, reduction in number of TCP timeouts and retransmissions, faster congestion notification, and lower packet delay.
2. BECN uses existing network layer signalling and does not require the use of any transport layer protocol for congestion notification. It is therefore protocol inde-

pendent and can be used by other transport protocols such as UDP. Also, there may be value in providing a common mechanism for notifying all transport protocols about congestion.

3. BECN provides faster congestion notification as compared to the ECN mechanism. This could be particularly useful in networks with large delay such as satellite networks. There is a clear need to investigate the possibility of a BECN advantage in this scenario.
4. Finally, the BECN scheme allows the development in the future of multi-level congestion feedback schemes. Till now, this has not been possible since both the duplicate ACKs and ECN schemes cannot carry multi-level congestion feedback notification. However, with use of ISQs there is possibility of such a mechanism.

3.3 Enhanced BECN algorithm

In this section, we describe our improved BECN algorithm explaining the guidelines for the behavior of all hosts in a BECN-capable TCP/IP network.

3.3.1 Behavior of a BECN-capable router.

If (*arriving packet causes the average queue size to go above maximum RED threshold*)

{

check IP header of packet and drop the packet just like an ECN packet;

if (*ECT¹ bit was marked in the IP header*)

¹The ECN-Capable Transport (ECT) bit is set in the IP header of an ECN or BECN IP packet [11] for identification at the router.


```

{
    send an ISQ due to a dropped packet back to the source;
}
} else if (arriving packet causes average queue to go between
minimum and maximum RED thresholds){
    if (RED rules chooses this packet for marking and ECT bit
is set) and if (packet is not already marked)
    {
        mark the packet (CE2 bit) and send an ISQ due to a
        marked packet back to the source;
    }
    else if (RED chooses this packet and ECT bit is not set)
        drop the packet;
}

```

3.3.2 Behavior of BECN-capable TCP end hosts

BECN TCP end hosts do not need initial end-to-end negotiations to establish BECN capability as the TCP receiver is not involved in the congestion notification mechanism. The TCP sender on receipt of an ISQ due to a marked packet sets its congestion window and *ssthresh* to one-half of current congestion window and waits a round-trip time (RTT) before it starts increasing the window. An ISQ due to a dropped packet causes the sender to set its congestion window and *ssthresh* to one-half of current congestion window and follows the TCP congestion control algorithm thereafter. The sender does not react to ISQs more than once per RTT.

²The Congestion Experienced (CE) bit is set in the IP header of an ECN or BECN IP packet [11] as an indication of congestion.

Our suggested algorithm introduces some modifications to the original BECN algorithm [7]. It ensures that the BECN sender is not unduly aggressive by creating a delay of one RTT before sender starts to increase its congestion window after a window reduction. This gives the network time to alleviate the state of congestion before packet injection rate is increased.

The modified algorithm requires the use of a single bit to differentiate between an ISQ due to a marked packet and an ISQ due to a dropped packet. An ISQ due to a marked packet would have this bit set so that the sender on receipt of the ISQ detects it should wait an RTT before increasing its window according to the TCP algorithm. When the bit is unset, the sender assumes the ISQ was generated due to a dropped packet. It halves its congestion window and starts increasing the window according to the TCP algorithm similar to ECN. Experimental investigation showed that this response due to a dropped packet gave improved results for BECN. It was observed that the option of waiting an RTT before increasing the window in this case degraded TCP throughput for BECN and so was not followed. The advantage for BECN here is early notification of packet drops since it does not wait for duplicate acknowledgements before responding to congestion.

This thesis proposes the use of a single bit in the 32 bit unused field in the ICMP Source Quench packet header for the ISQ differentiation. A bit within the fifth octet of the header can be used.

3.4 Mathematical model and predictions for TCP throughput and queue dynamics in ECN/BECN-capable networks

In this section some analytical models are developed which give better understanding of the impact of BECN's early congestion notification due to ISQs on the performance of BECN in comparison to ECN. These models provide mathematical justification for some of the benefits of a BECN-based congestion control mechanisms in IP networks. Simulation results in later sections of this thesis offer experimental validation of the models.

3.4.1 TCP throughput for ECN and BECN capable connections

This thesis extends the stochastic model of TCP congestion control developed in [33] and [28] for a saturated TCP sender to include early notification due to Backward Explicit Congestion Notification (BECN). With the addition of Explicit Congestion Notification (ECN) schemes to TCP, a congestion situation is indicated by either of the following three mechanisms: Explicit Notification (EN) due to ECN or BECN, triple duplicate acknowledgements (TDA) or timeouts (TO). Results in [9] and [19] show that Explicit Notification schemes significantly reduce occurrence of timeouts and triple duplicate ACKs hence in our analysis we modify the TCP model in [33] to deal with only explicit congestion notification due to ECN or BECN. This presumes that RED parameters are also set such that average queue size, $\bar{q}$ is maintained within RED thresholds. As in [33] our model focuses on TCP's congestion avoidance mechanism where the TCP sender increases its TCP congestion window size W by $\frac{1}{W}$ each time an ACK packet is received.

Figure 3.1: Evolution of window size for Explicit Congestion Notifications

Conversely, when congestion notification is achieved due to ECN or BECN the sender reduces its TCP window by one-half.

Using the same notations as in [33], we assume that a packet is marked in a round independently of any packets marked in other rounds and that packet markings are correlated among the back-to-back transmissions within a round. We define p to be the probability that a packet is marked given that it is the first packet in its round or the preceeding packet in its round is not marked. Fig. 3.1, also used in [33] shows a sample evolution of the congestion window size for the explicit notification schemes. Between two explicit notifications, the window increases by $\frac{1}{b}$ packets per round where b is the number of packets acknowledged by a received ACK. As in [33], $b = 2$ is used in our analysis. We define an EN period (ENP) to be the period between two explicit congestion notifications (Fig. 3.1). For the i -th ENP we define Y_i to be the number of packets sent in the period, A_i the duration of the period, and W_i the window size at the end of the period. $B(p)$ is defined as the number of packets sent into the network per unit of time (or throughput of the connection). Considering $\{W_i\}_i$ to be a Markov regenerative process with rewards $\{Y_i\}_i$ as in [33] it can be shown that

$$B = \frac{E[Y]}{E[A]} \quad (3.1)$$

Figure 3.2: Packets sent during an EN period

Consider the EN period shown in Fig. 3.2 (also used in [33]). We denote by α_i the first marked packet in an ENP and by X_i the round where this marking occurs (Fig. 3.2). We also define a parameter σ for ECN and BECN networks which is a fraction of the number of packets sent into the network in an additional round after a packet marking and before an explicit congestion notification is received by the sender. Hence, after packet α_i is marked by the congested router, $\sigma(W_i - 1)$ more packets are sent in an additional round before an explicit congestion notification is received by sender, where

$$\sigma = \begin{cases} 1 & \text{if ECN;} \\ 0 \leq \sigma < 1 & \text{if BECN.} \end{cases} \quad (3.2)$$

We use $\sigma = 1$ for ECN and $\sigma < 1$ for BECN to model BECN's earlier notification which ensures that with BECN, less packets are injected into the network after a packet marking compared to ECN (i.e $\sigma(W_i - 1) < W_i - 1$). Thus, a total of $Y = \alpha + \sigma W - \sigma$ packets are sent in $X_i + 1$ rounds. It follows that

$$E[Y] = E[\alpha] + \sigma E[W] - \sigma \quad (3.3)$$

The exact value of σ for any BECN network would be dependent on the location of the congested router from which BECN congestion notifications are sent. The determination

of σ values is beyond the scope of this thesis but we believe its bounds are as defined in (3.2). To derive $E[\alpha]$, consider the random process $\{\alpha_i\}_i$, where α_i is the number of packets sent in an EN period up to and including the first packet that is marked. Using derivations in [33] it can be shown that

$$E[\alpha] = \frac{1}{p} \quad (3.4)$$

$$E[W] = \sqrt{\frac{8}{3bp}} \quad (3.5)$$

$$E[A] = RTT(\sqrt{\frac{4}{3p}} + 1) \quad (3.6)$$

Hence,

$$E[Y] = \frac{1 - \sigma p}{p} + \sigma \sqrt{\frac{4}{3p}} \quad (3.7)$$

It follows that

$$B(p) = \frac{\frac{1 - \sigma p}{p} + \sigma \sqrt{\frac{4}{3p}}}{RTT(\sqrt{\frac{4}{3p}} + 1)} \quad (3.8)$$

Using (3.8) we develop a prediction about TCP throughput as a function of p for ECN connections as well as for BECN connections using arbitrarily chosen values of σ (0.5 and 0.8). For simplicity we assume that link propagation delay accounts for the bulk of network round-trip time (RTT), hence we use an $RTT = 88ms$ corresponding to link propagation delays used in our simulation (Fig. 3.7). Fig. 3.3 - Fig. 3.4 based on equation (3.8) show plots of TCP throughput $B(p)$ in an ECN or BECN network as a function of packet marking probability p for different values of σ (0.5, 0.8). Values of p are varied to model different levels of congestion. *The figures predict that the average throughput for long-lived TCP flows in an ECN or BECN capable network should be similar for levels of congestion during which average queue size is within RED thresholds.* Experimental results in section 3.6.2, 5.3.1 and 6.3.2 validate this prediction.

3.4.2 RED-ECN/BECN Feedback Control System

We model n ECN or BECN capable TCP flows sharing a single RED-ECN/BECN router as in [34] and determine the steady state values for packet marking probability p_o and average queue q_o when the network is in equilibrium. We use the same assumptions in [34] except that instead of probabilistic packet droppings, we deal with packet markings. At the RED-ECN/BECN router, the marking probability rule $H(\bar{q})$ is defined as:

$$H(\bar{q}) = \begin{cases} 0 & 0 \leq \bar{q} < q_{min}; \\ \frac{\bar{q} - q_{min}}{q_{max} - q_{min}} p_{max} & q_{min} \leq \bar{q} < q_{max}; \\ 1 & q_{max} \leq \bar{q} < Q. \end{cases} \quad (3.9)$$

From derivations in [34] and assuming 100% link utilization, two relationships between p and $\bar{q}$ result:

$$p = H(\bar{q}) \quad (3.10)$$

$$\bar{q} = c[B_{RTT}^{-1}(p, \frac{c}{n}) - RTT_o] \quad (3.11)$$

where RTT_o is the total propagation and transmission time on a path excluding the queue waiting time.

Using (3.10) and (3.11) we obtain a prediction for the equilibrium point (p_o, q_o) for n ECN connections as well as n BECN connections for σ values of 0.5 and 0.8. The predictions are based on a line speed $c = 10Mbps$ and $n = 15flows$. In Fig. 3.5 - Fig. 3.6 we use equation (3.10) and (3.11) to determine the equilibrium point for 15flows based on a line speed of 10Mbps for different values of σ (0.5, 0.8). The figures predict that the difference in average queue size at the equilibrium point for both ECN and BECN mechanisms should be insignificant. Simulation results in section 3.6.2, 5.3.1 and 6.3.2 validate this prediction.

3.4.3 Variance of average queue, $\bar{q}(p)$

Taking the variance of (3.11)

$$Var(\bar{q}(p)) = c^2 Var(B_{RTT}^{-1}(p, \frac{c}{n})) \quad (3.12)$$

Using notations as in [34] it can be shown that

$$B_{RTT}^{-1}(p, \frac{c}{n}) = RTT(p, \frac{c}{n}) = \frac{\frac{1-\sigma p}{p} + \sigma \sqrt{\frac{4}{3p}}}{\frac{c}{n}(\sqrt{\frac{4}{3p}} + 1)} \quad (3.13)$$

Assuming p to be a random variable uniformly distributed between 0 and 0.1, it can be shown that $(E[B_{RTT}^{-1}(p, \frac{c}{n})])^2$ can be written in the form

$$(E[B_{RTT}^{-1}(p, \frac{c}{n})])^2 = \sigma^2 \kappa_0 \quad (3.14)$$

where κ_0 is a constant w.r.t ECN and BECN.

Also it can be shown that $E[B_{RTT}^{-1}(p, \frac{c}{n})]^2$ can be written in the form

$$E[B_{RTT}^{-1}(p, \frac{c}{n})]^2 = \sigma^2 \kappa_1 + \sigma \kappa_2 + \kappa_3 \quad (3.15)$$

where κ_1 , κ_2 , and κ_3 are constants w.r.t ECN and BECN.

Hence $Var(B_{RTT}^{-1}(p, \frac{c}{n}))$ would be of the form

$$Var(B_{RTT}^{-1}(p, \frac{c}{n})) = \sigma^2 \kappa_4 + \sigma \kappa_5 + \kappa_6 \quad (3.16)$$

where κ_4 , κ_5 , and κ_6 are constants w.r.t ECN and BECN.

Since variance is a non-negative variable,

$$Var(B_{RTT}^{-1}(p, \frac{c}{n}))_{BECN} < Var(B_{RTT}^{-1}(p, \frac{c}{n}))_{ECN}. \quad (3.17)$$

for all values of $\kappa_4, \kappa_5, \kappa_6$ where $Var(B_{RTT}^{-1}(p, \frac{c}{n}))_{BECN}$ and $Var(B_{RTT}^{-1}(p, \frac{c}{n}))_{ECN}$ are the $Var(B_{RTT}^{-1}(p, \frac{c}{n}))$ for BECN and ECN respectively. It therefore follows from equation (3.12) and (3.17) that with the BECN mechanism, *due to early notification the variance*

Figure 3.3: Average throughput ($\sigma = 0.5$)

Figure 3.4: Average throughput ($\sigma = 0.8$)

in queue size should be less than that of the ECN mechanism for long-lived TCP transfers. Also for networks with a large bandwidth-delay product this difference in queue variation should increase since σ becomes even smaller for BECN. Experimental results in section 3.6. 2, 3.6.4, 5.3.1, 5.3.2, 6.3.2 and 6.3.5 validate these predictions.

3.5 Simulation Setup

Study and comparative evaluation of the modified BECN and ECN mechanisms is done using the Network Simulator (ver 2.1b8a) [21]. In this section, the network scenarios, simulation parameters and performance metrics used in evaluating the algorithms are described.

3.5.1 Simulation Topology

Fig. 3.7 illustrates Network 1 a single bottleneck topology that we use in most of our simulations. The bottleneck bandwidth is 10Mbps with a propagation delay of 40ms.

Figure 3.5: Average queue ($\sigma = 0.5$)

Figure 3.6: Average queue ($\sigma = 0.8$)

All other links have 100Mbps capacity with 2ms propagation delay. Link capacities of 10Mbps and 100Mbps are common in Local Area Networks while the link propagation delays were chosen to ensure that the bandwidth-delay product permitted the range of network loading used in our experiments. Nodes $Fc(1)..Fc(n)$ serve as FTP clients with nodes $Fs(1)..Fs(n)$ as corresponding FTP servers. Hosts $Wc(1)..Wc(m)$ serve as web clients with corresponding $Ws(1)..Ws(m)$ hosts serving web servers. Traffic flow is from servers to clients.

3.5.2 Queue parameters

RED queue management was used in all simulations with or without support for ECN or BECN. Guidelines by Floyd [6] are followed in setting the RED parameters as follows: $min_{th} = 15KB$, $max_{th} = 3 * min_{th}$, $buffer\ size = 2 * max_{th}$, $max_p = 0.1$, $w_q = 0.002$. Byte-based dropping for RED was used.

Figure 3.7: Network Topology 1

3.5.3 Traffic sources

In the experiments two types of traffic sources were used:

Long-lived TCP traffic sources

FTP traffic model in the Network Simulator (NS) is used with an infinite amount of data to send. TCP type is NewReno with a data packet size of 1000 bytes and ACK packet size of 40 bytes. The TCP clock granularity is set to 100ms. Though a number of older systems use 500ms as TCP clock granularity, most current systems have no problem coping with 100ms. The maximum TCP congestion window was set to 100KB to ensure that in all experiments the TCP transfers were in the order of the bottleneck bandwidth-delay product of the link. Delayed ACKs are not used in the experiments.

Short-lived web-traffic sources

Here the built-in web-traffic model in NS with parameters in Table 3.1 was used.

The intersession time, number of sessions, session sizes, and inter-object time were chosen to ensure that several sessions were active throughout simulation time, while

Table 3.1: WEB Traffic Parameters

Parameter value	Distribution	Mean	Shape
Inter-session time	Exponential	0.5s	-
Session size	Constant	10 pages	-
Inter-page time	ParetoII	10s	2
Pagesize	ParetoII	3 objects	1.5
Inter-object time	ParetoII	0.1s	1.5
Object size	ParetoII	12 packets	1.2
Number of sessions	Constant	1000	-

other parameters were chosen based on recommendations in [18].

3.5.4 Performance Metrics

The following performance metrics were used for evaluation:

Goodput for a TCP flow

Computed based on the number of data packets received by the receiver. The number of ACKs received by the sender within simulation time is used for the computation.

Average goodput

For a number of TCP flows this is the average of their individual goodputs.

Web object transfer delay

For a transferred web object, this is the interval between the time a web client makes an initial request (GET message) and the time the server receives the ACK to the last data packet for the object requested by client. In the NS webtraffic model, computing the

object transfer time requires identification of the two separate flow pairs used in NS to represent the GET flow and the data transfer flow.

Percentage loss

Measures the ratio of the number of packets dropped at the bottleneck link to the total number of packets injected into the bottleneck link for a particular flow or set of flows.

Bottleneck utilization (U)

Utilization is computed using the following formula:

$$U = \frac{(8 * 1000 * n) + (8 * 40 * m) + (8 * 40 * p)}{c} \quad (3.18)$$

where n, m and p are the number of data, ACKs, and ICMP packets respectively that successfully traverse the bottleneck link and are received. c is the capacity of the bottleneck link. For experiments with one-way traffic m and p would be zero.

Percentage ISQ reverse traffic

computed as the number BECN ISQs generated in the reverse direction of data flow as a ratio of the total number of packets in the reverse direction.

3.6 Experiments and Results

In this section, five sets of experiments are described and explanations for the observed behavior of the algorithms (RED, ECN and the modified BECN) is presented. All simulations are run for 500s and data for results were collected after a period of 100s to avoid bias due to initial conditions. For each of the algorithms, the same seed was used in all random number generators to ensure all algorithms are tested based on same input variables.

Figure 3.8: Goodput gain over TCP

Figure 3.9: Percentage loss

Figure 3.10: Goodput - Homogeneous flows

Figure 3.11: Loss - Homogeneous flows

3.6.1 Competing long-lived BECN/ECN flows with plain TCP flows - [Fig. 3.8 and Fig. 3.9]

This experiment captures the scenario where some Internet users decide to use either the BECN or ECN algorithm for congestion control in a Internet where RED is widely deployed. Flows are described as “plain TCP” if they are treated based on pure RED algorithm at the routers. ECN and BECN flows are treated based on the ECN and enhanced BECN algorithms respectively. In this experiment, a number of either BECN or ECN FTP flows compete with an equal number of plain TCP FTP flows. The total number of competing flows is varied using 8, 12, 16, 18, 20, 26, 30, 36 and 40 flows (i.e. $n = 8, 12, 16, 18, 20, 26, 30, 36, 40$ in section 5A), to observe performance under different levels of congestion. The start time for all flows is a random variable uniformly distributed between 0s and 5s. Fig. 3.8 shows the measured gain in goodput for BECN and ECN over plain TCP flows.

Fig. 3.8 shows that the BECN algorithm offers upto 43% gain in average goodput under high congestion over plain TCP flows. Recall that unlike plain TCP packets, the BECN packets are not dropped probabilistically but marked. As congestion increases plain TCP flows experience more packet drops while the BECN flows continue to send packets successfully across the network. Under high congestion while BECN suffers a 0.07% packet loss, plain TCP flows suffer 5.21% loss (Fig. 3.9) - thus, higher goodput for BECN. For same reasons, ECN offers upto 31.5% gain in average goodput during high congestion with a loss of 0.18%. BECN therefore offers greater gain over plain TCP during high congestion as *it suffers less losses due to early notification compared to ECN*.

However, during very low congestion we observe that ECN offers higher gain over plain TCP compared to BECN. The reason is that during low congestion, the dropping and marking probability is small, therefore plain TCP flows not only experience less drops but also do not respond to packet drop notification early since they have to wait

for duplicate acknowledgements. On the other hand, BECN senders are quenched quickly and this reduces their average goodput in this case. ECN senders in this scenario are also quenched but not as early as the BECN senders since they have to wait more than half RTT before responding to congestion notification due to a marked packet. The advantage of BECN's early notification becomes more obvious under higher congestion as explained above.

3.6.2 Homogeneous long-lived flows - [Fig. 3.10 - Fig. 3.14]

In this experiment, the flows are called "homogeneous" as all flows in a particular experiment are either plain TCP, ECN capable or BECN capable. Three sets of simulations are done in the experiment. In each simulation, FTP flows are either all plain TCP, all ECN-capable or all BECN-capable. The FTP flows start randomly within the initial 5s of simulation. The number of FTP connections is varied using 10, 15, 25, 30, 35, 40, and 45 flows (i.e. $n = 10, 15, 25, 30, 35, 40, 45$; $m = 0$ in section 5A). Average goodput for the TCP flows is computed. Fig. 3.10 - Fig. 3.14 summarizes the observed results. We noted that though plain TCP homogeneous flows experience higher losses - 4.6% (Fig. 3.11), their average queue size (31392 bytes) remains less than that of BECN (39247 bytes) or ECN (40034 bytes) under high congestion (45 background flows). This is because arriving plain TCP packets are dropped probabilistically not marked as with BECN and ECN packets. However, the plain TCP flows achieve between 97.44% and 99.7% utilization on the bottleneck link (Fig. 3.12), and their average goodput remain comparable to that of homogeneous BECN and ECN flows (Fig. 3.10). This is due not only to the improved TCP NewReno fast recovery mechanism which ensures that when multiple packets are lost from a single window of data, TCP can recover without a re-transmission timeout [20], but also because all flows in individual sets of simulations are same type, and therefore, equally competing for available bandwidth. It is noted that

Figure 3.12: Bottlebeck utilization

Figure 3.13: Queue - 15 ECN FTP

Fig. 3.10 also validates our prediction in section 3.4.1 from the developed mathematical model as it shows that the average goodput for TCP flows in an all ECN or BECN network is similar under all levels of congestion used in this experiment.

However, it was observed that due to earlier notification BECN flows experience less fluctuations in queue size with 15 background flows (Fig. 3.13 and Fig. 3.14) compared to ECN flows, even though their average queue sizes are about the same. This also validates our prediction in section 3.4.2 and 3.4.3. As a result, even though packets are not dropped for either BECN or ECN flows (Fig. 3.11), the BECN flows achieve higher bottleneck link utilization (Fig. 3.12). With higher congestion (45 background flows), BECN flows suffer slightly lower losses - 0.99% compared to the ECN flows - 1.19% (Fig. 3.11).

3.6.3 Short-lived web transfers - [Fig. 3.15 - Fig. 3.18]

This test case assesses the performance of BECN and ECN with short-lived web-traffic workloads. In this experiment, there are either 10 BECN or ECN capable web servers and

Figure 3.14: Queue - 15 BECN FTP

Figure 3.15: Average object transfer delay

Figure 3.16: HTTP data packet loss

Figure 3.17: Queue - ECN web + 15 FTP

Figure 3.18: Queue - BECN web + 15 FTP

Figure 3.19: Goodput - Homogeneous flows

Figure 3.20: Loss - Homogeneous flows

Figure 3.21: Bottleneck utilization

Figure 3.22: 45 Plain TCP FTP (large RTT)

Figure 3.23: 45 ECN FTP (large RTT)

10 BECN or ECN capable web clients, while homogeneous BECN or ECN background FTP connections respectively are varied using 5, 10, 15, 20, 25, 30, and 35 flows (i.e. $n = 5, 10, 15, 20, 25, 30, \text{ and } 35$; $m = 10$ in section 5A), to establish different levels of congestion. The FTP flows start randomly within the initial 5s of simulation while the web-traffic connections start after 50s. The average object transfer delay for the web transfers is recorded. Fig. 3.15 - Fig. 3.18 show the observed results. Fig. 3.16 shows that the burstiness of the web-traffic workloads causes packet loss for both BECN and ECN connections especially during high congestion. Contributors to transfer delay Fig. 3.15 include queue size (delay), packet sending rate, and packet loss which leads to TCP timeout and retransmission. Fig. 3.17 and Fig. 3.18 show that the queue size for BECN is slightly lower than that of ECN with 15 background FTP flows. For short-lived connections, the BECN advantage of early notification would be enjoyed only if the BECN ISQ reaches the sender while there are still a good number of outstanding packets to be sent. The occurrence of such a situation clearly depends on the all the factors that determine the network condition at that particular time. It can be observed that while

Figure 3.24: 45 BECN FTP (large RTT)

Figure 3.25: Average object transfer delay

ECN suffers 5.09% loss with 35 background flows, BECN suffers 3.76% loss (Fig. 3.16). BECN also offers upto 8.1% reduction in the average web object transfer time (with 10 background flows) compared to ECN (Fig. 3.15).

3.6.4 Large round-trip times (RTT) - [Fig. 3.19 - Fig. 3.28]

In this section, the experiments in section 3.6.2 and section 3.6.3 are repeated using plain TCP, ECN and BECN test flows with propagation delay on the bottleneck link increased to 250ms. Roundtrip delays of 500ms are common with geostationary satellite links. Fig. 3.19 - Fig. 3.28 summarizes the observed results.

For experiments with only homogeneous long-lived flows, it is observed that due to early notification, the BECN flows are able to maintain their queue size below maximum RED threshold (Fig. 3.24), resulting in a 0% packet loss with 45 background flows (Fig. 3.20). The ECN flows wait more than half RTT before responding to congestion thus it is observed that their queue size occasionally exceeds the maximum threshold (Fig. 3.23) resulting in 0.77% loss (Fig. 3.20). These results further validate predictions

Figure 3.26: HTTP data packet loss

Figure 3.27: 25 ECN FTP + web (RTT)

in section 3.4.3. In this scenario, the BECN algorithm therefore offers better network utilization (Fig. 3.21) and 20% gain in average goodput over ECN (Fig. 3.19) under high congestion (45 background flows). Recall that packet marking probability is proportional to average queue size and halving the congestion window reduces packet injection rate which directly affects utilization of the bottleneck link. It is also observed in this scenario that plain TCP NewReno flows have lower average and instantaneous queues compared to ECN flows (Fig. 3.22 and Fig. 3.23) and achieve higher utilization on the bottleneck link (Fig. 3.21). Though plain TCP flows experience slightly greater packet drops - only 0.86% (Fig. 3.20), they achieve average goodput 6.6% higher than that of ECN flows (Fig. 3.19) with 45 background flows. In addition to the higher probability of quenching the ECN flows due to higher queue size, the TCP flows enjoy the NewReno fast recovery advantage and even have a small probability of experiencing packet drops due to their lower queue size (Fig. 3.22).

For experiments with short-lived web transfers, it is observed that for same reasons as explained above the average queue for BECN is less than that of ECN with 25 background

Figure 3.28: 25 BECN FTP + web (RTT)

Figure 3.29: ISQ reverse traffic

flows (Fig. 3.27 and Fig. 3.28) and BECN flows suffer packet losses 88.3% less than ECN flows (Fig. 3.26). The BECN algorithm in this case offers an average transfer delay 6.4% less than that of ECN with 25 background flows (Fig. 3.25).

3.6.5 ISQ reverse traffic analysis - [Fig. 3.29]

In this section, the experiments in section 3.6.2 with only homogeneous FTP flows are repeated using 40ms propagation delay on the bottleneck link. The experiments are re-done with propagation delay on the bottleneck link increased to 250ms. The ratio of the number of ISQs generated to the total number of packets received in the reverse direction of traffic is computed in each case. Fig. 3.29 shows that with 40ms bottleneck link propagation delay the ratio of number of ISQs generated under high congestion is only 5.4% of total number of packets in reverse direction. ICMP Source Quench packets are 56 bytes in size (including the IP header) for IPV4 and would therefore consume even less percentage of reverse bandwidth compared to much larger TCP packets in a real life network. Fig. 3.29 also shows that with larger roundtrip time the computed

ratio of number of generated ISQs under high congestion is much less - only 0.64%. This is because with an increase in bandwidth-delay product, the bottleneck utilization and average queue size reduces, and this directly impacts probability of marking packets. The network topology (Fig. 3.7) used in this simulation experiment is by no mean representative of the Internet and so further work with more realistic network scenarios is required to better understand the impact of ISQ reverse traffic on BECN performance.

3.7 Conclusion and Future work

This chapter suggested modifications to a proposed BECN algorithm for IP [7] and used simulations to explore the performance of a modified BECN algorithm. First, we modeled TCP throughput and RED queue behavior in BECN and ECN capable networks and validated this model through simulation experiments. We showed mathematically that due to the advantage of early notification, the BECN mechanism results in less queue variations compared to ECN (3.17). Other deductions from our model predict that for paths without large delays the TCP throughput for flows in a BECN capable network should be about same as for flows in an ECN capable network (Fig. 3.3 - Fig. 3.4). These predictions were validated using simulations.

The results in this chapter suggest that the enhanced BECN mechanism for congestion control in TCP/IP networks can offer improved performance over the ECN mechanism for both long-lived and short-lived TCP transactions. On links that have a high bandwidth-delay product, BECN offers significant improvement over the ECN mechanism in terms of average goodput for long-lived traffic and packet loss, transfer delay for short-lived workloads.

Therefore, it is concluded that the BECN mechanism is a viable scheme for congestion control in TCP/IP networks and can be used to offer improved quality of service on links with a high bandwidth-delay product.

There is interest in further examining the performance of the enhanced BECN in the presence of multiple congested gateways and two-way bulk TCP transfers. There is also interest in evaluating multiple-level BECN (MECN) under a variety of scenarios and workloads. Investigating a mechanism that combines both the ECN and BECN mechanism to give a more robust scheme for congestion control is also under consideration.

Chapter 4

Multi-level BECN (MECN) mechanism

4.1 Introduction

The congestion notification mechanisms considered so far (BECN, ECN and RED packet drop schemes) have been described as “binary notification schemes” [7] since the sender only gets to be notified that the network is congested or not congested. However with the BECN mechanism more quantitative information about the level of congestion in the network can be sent to the TCP sender so that sender reduces packet injection rate more accurately according to the perceived level of congestion in the network. This is called the multi-level BECN (MECN) mechanism. The concept of multi-level backward congestion notification with the use of ICMP source quench packets was originally proposed in [7]. In [7], the weight of the congestion level sent back to the sender is a multiplicative factor of the marking probability and is stored in the 32-bit unused field of the ISQ. With this scheme, when the average queue size is at the minimum RED threshold a weight of 1% is sent to the sender while when the average queue size is at the maximum threshold a

weight of 99% is sent. In case of a packet drop, a weight of 100% is sent. The sender responds by reducing its congestion window and *ssthresh* by half if the received weight is greater than or equal to 50% otherwise the congestion window is linearly decremented by 1. Authors in [7] did not show experimental evaluation of the proposed mechanism but anticipated that the MECN mechanism would result in effective use of the network resources and improved performance since MECN enabled TCP hosts would only “back off” according to the perceived level of congestion in the network.

In this chapter we suggest a modified scheme for multi-level BECN (MECN) which also uses ICMP Source Quenches to carry information on the severity of congestion back to the sender. With the modified scheme a congested router does not need to compute the weight of the congestion level but sends *a weight based on a configurable queue threshold*. Better details of the mechanism is given in the next section.

4.2 Suggested MECN algorithm

In this section, a modified MECN algorithm is described and guidelines for the behavior of all hosts in a MECN-capable TCP/IP network is explained.

4.2.1 Behavior of a MECN-capable router

For a MECN-capable router a new threshold *midthresh* is defined, henceforth referred to as “MECN threshold”. The MECN threshold has a value between the minimum and maximum RED thresholds and is used as a basis for defining low and high congestion. When the average queue size is below the MECN threshold the router infers low congestion while high congestion is detected when average queue size is above MECN threshold. The ISQs sent from the router carry information on whether congestion level is low or high. A new bit (*MECN bit*) is defined within the 32-bit ICMP packet header which

is set or unset by the router to notify the source of perceived high or low congestion respectively.

For a new arriving packet, the MECN-capable router behaves as follows:

```
If (arriving packet causes the average queue size to go above
maximum RED threshold)
{
  check IP header of packet and drop the packet just like an
  ECN packet;
  if (ECT bit was marked in the IP header)
  {
    send an ISQ due to a dropped packet back to the source;
  }

} else if (arriving packet causes average queue to go between
minimum RED threshold and the MECN threshold){
  if (RED rules chooses this packet for marking and ECT bit
is set) and if (packet is not already marked)
  {
    mark the packet (CE bit) and send an ISQ due to a
    marked packet back to the source with the MECN bit
    unset;
  }

} else if (arriving packet causes average queue to go between
the MECN threshold and maximum RED threshold){
```

```

if (RED rules chooses this packet for marking and ECT bit
is set) and if (packet is not already marked)
{
    mark the packet and send an ISQ due to a
    marked packet back to the source with the MECN bit set;
}

else if (RED chooses this packet and ECT bit is not set)
drop the packet;
}

```

This router behavior ensures three levels of congestion notification scheme - low congestion with packet marking, high congestion with packet marking, and high congestion with packet drops when average queue exceeds *maxthresh*. It is possible to define more thresholds for more levels of notification but in this chapter we are only interested in investigating and evaluating a simple three-level notification scheme to enable better observation of the impacts of multi-level notification on TCP performance.

4.2.2 Behavior of MECN-capable TCP end hosts

MECN TCP end hosts do not need initial end-to-end negotiations to establish MECN capability as the TCP receiver is not involved in the congestion notification mechanism. The TCP sender on receipt of an ISQ due to a marked packet with a set MECN bit reduces its congestion window and *ssthresh* to *one-half of current congestion window* and waits a full roundtrip time (RTT) before it starts increasing the window. If the ISQ due to the marked packet has an unset MECN bit the TCP sender reduces its congestion window and *ssthresh* to *80% of current congestion window* instead. An ISQ due to a dropped packet causes the sender to set its congestion window and *ssthresh* to one-

half of current congestion window and follows the TCP congestion avoidance algorithm thereafter. The sender does not react to ISQs more than once per RTT.

The MECN algorithm just like the enhanced BECN algorithm described in section 3.3 requires the use of a single bit to differentiate between an ISQ due to a marked packet and an ISQ due to a dropped packet.

4.3 Simulation Setup

In this section the network scenarios, simulation parameters and performance metrics used in evaluating the MECN mechanism is described.

4.3.1 Simulation Topology and parameters

Fig. 3.7 in section 3.5.1 illustrates the topology that was used in simulations in this section. RED queue management with the same parameters specified in section 3.5.2 was used for the router queue while the long-lived traffic source described in section 3.5.3 was used.

4.3.2 Performance Metrics

The same performance metrics defined in section 3.5.4 were used for evaluation.

4.4 Experiments and Results

In this section, two sets of experiments are described and explanations for the observed behavior of our MECN algorithm is presented. The experiments are similar to those carried out for BECN and ECN in section 3.6.2 and 3.6.4. All results are shown alongside previously obtained results for the BECN and ECN mechanisms to facilitate comparative

Figure 4.1: Goodput - Homogeneous flows

Figure 4.2: Loss - Homogeneous flows

evaluation. The same set of seeds used for the BECN and ECN experiments are used in all random number generators to ensure the MECN algorithm is tested based on same input variables. In all experiments three results for MECN are shown each corresponding to simulations using a different configured MECN threshold to enable observation of the impact of varying the MECN threshold on TCP performance. The three thresholds used are 20KB, 30KB and 40KB.

4.4.1 Homogeneous long-lived flows - [Fig. 4.1 - Fig. 4.3]

This experiment is similar to that carried out in section 3.6.2 except that in this case the FTP flows are all MECN-capable. Fig. 4.1 - Fig. 4.3 summarizes the observed results. We observe that due to the aggressive behavior of MECN with a threshold of 40KB, percentage packet loss increases steadily with congestion when number of active FTP flows exceed 25 (Fig. 4.2) and this results in a decreasing bottleneck utilization as congestion and packet drops increase (Fig. 4.3). Recall that a MECN flow reduces its congestion window by only 20% as long as the average queue size of the congested router

Figure 4.3: Bottleneck utilization

Figure 4.4: Goodput - Homogeneous flows

in below the configured MECN threshold. With 45 FTP flows 2.4% loss is recorded for MECN with 40KB threshold (Fig. 4.2) compared to 1.19% and 0.99% loss for ECN and BECN respectively. However with a MECN threshold of 20KB and 30KB we observe that the percentage loss recorded for MECN reduces to 0.98% and 1.1% respectively (Fig. 4.2). The average goodput for MECN remains similar to that obtained for BECN and ECN and does not seem to be affected by the configured MECN threshold (Fig. 4.1). MECN injects more packets into the network than both BECN and ECN hence despite greater packet loss, a good number of packets still reach the receiver and are acknowledged.

4.4.2 Large round-trip times (RTT) - [Fig. 4.4 - Fig. 4.7]

In this section, the experiment in section 3.6.4 is repeated using MECN test flows with propagation delay on the bottleneck link increased to 250ms. Fig. 4.4 - Fig. 4.7 summarizes the observed results.

Fig. 4.4 shows that the average goodput for MECN with 20KB, 30KB and 40KB MECN thresholds remain similar to that of BECN while 0% percentage packet loss

Figure 4.5: Loss - Homogeneous flows

Figure 4.6: Bottleneck utilization

(Fig. 4.5) is recorded for MECN (with all the three thresholds) just like observed for BECN under same conditions. Even though MECN is more aggressive under low congestion compared to the ECN and BECN mechanisms we observe that the average queue size for MECN with 40KB threshold remains lower than the maximum RED threshold under high congestion (45 FTP flows) for paths with large bandwidth-delay product (Fig. 4.7) resulting in the observed 0% loss. However Fig. 4.6 also shows that due to the aggressive behavior of MECN the bottleneck utilization for MECN with 40KB and 30KB thresholds is 2.7% and 0.43% higher than that of BECN respectively under high congestion (45 FTP flows). With a 20KB threshold the utilization under low congestion is also higher but becomes similar to that of BECN under heavier congestion (Fig. 4.6) when MECN becomes less aggressive.

4.5 Conclusion

Results from experiments carried out in this chapter show that for the MECN algorithm considered in this thesis there was no observed significant improvement compared to the

Figure 4.7: 45 MECN FTP flows (large RTT) - 40KB threshold

BECN mechanism in terms of average goodput and bottleneck utilization for connections with small round trip times. In fact depending on the configured MECN threshold, significant levels of packet drops could result due to the aggressive behavior of the MECN algorithm as defined in this thesis. For paths with a large bandwidth-delay product it is observed that MECN can offer slight improvement in the utilization of the bottleneck link but this does not result in any significant improvement in goodput over the BECN mechanism. Despite these results, the opportunity for multi-level notification as provided by MECN is of great value and improving the response of end hosts to multi-level congestion notifications is still a relevant issue for further research.

Chapter 5

Impact of two-way traffic and a lossy reverse path on the behavior of BECN, ECN and RED

5.1 Introduction

Studies have revealed that the dynamics of more realistic two-way traffic in a network can significantly impact the performance of congestion avoidance algorithms [22]. Authors in [23] showed that due to the interactions of data and ACK packets, two-way traffic can result in ACK-compression which gives rise to rapid fluctuations in queue length. In [22], it was found that with congestion avoidance algorithms, two-way traffic with window flow control can cause severe burstiness in data traffic and bunching of ACK arrivals. It was also noted in [22] that it is the clustering of ACKs which get queued behind data packets that reinforces the pattern of burstiness from one data window to the next and that this results from relying on the “self clocking” nature of ACKs to time transmission of data packets. It was further observed that ACKs flowing along with

data packets of an opposite direction connection do not act like a well-behaved clock and this could break the “self clocking” nature of the overall network [22]. An impact of these observations associated with two-way traffic is the degradation of throughput [22]. Floyd S. in [29] affirms that with RED gateways and the introduction of two-way traffic in a network with multiple congested gateways, measured throughput is acceptable for a connection with a priority service which gives higher priority to ACK packets over data packets [29], while with FIFO service in the gateways the resulting throughput is not acceptable. It was therefore concluded in [29] that it is not the two-way traffic itself that causes degradation of performance for connections with multiple congested gateways, but specifically the compressed ACKs which result in increased burstiness of traffic.

It has also been shown that TCP throughput can be significantly degraded in networks with sporadic high error rates as common with the wireless medium [24], [25], [26], [27]. Authors in [25] observed that while previous studies in this area assume that data packets are lost on such networks and ACKs are not lost, such an assumption is not valid as the effects of ACK loss due to errors on the reverse channel cannot be ignored unless the bit-error rate on the reverse channel is of the order of $10E-6$.

There has been limited experimental evaluation of the impact of both packet loss on the reverse path and two-way traffic on the performance of the ECN and BECN mechanisms. In [10], concern was raised as to how the loss of BECN ISQs on a congested reverse path would impact the performance of BECN. In this chapter, using simulations the impact of two-way traffic and a lossy reverse path on the performance of BECN in comparison to the ECN and RED mechanisms is evaluated.

5.2 Simulation Setup

In this section the network scenarios, simulation parameters and performance metrics used in evaluating the algorithms are described.

5.2.1 Simulation Topology

Here the two network topologies (Network 2 and Network 3) used in our experiments are described.

Network 2

Fig. 5.1 illustrates Network Topology 2 used in experiments with two-way traffic. All outer links have 100Mbps capacity with 2ms propagation delay while the bottleneck links (link 1 and link 3) each have capacities of 10Mbps with a propagation delay of 15ms. For simulations with larger round-trip time, the propagation delay on the bottleneck links is increased to 120ms. Link 2 also has a bandwidth of 10Mbps with 2ms propagation delay. Link capacities of 10Mbps and 100Mbps are common in Local Area Networks while the link propagation delays were chosen to ensure that the bandwidth-delay product permitted the range of network loading used in our experiments. FTP traffic flow from nodes $S(i)$ to $R(i)$ and nodes $X(i)$ to $Y(i)$ serve as background traffic that congest both forward and reverse paths respectively. The FTP traffic flows from host A to host AA and host B to host BB serve as forward and reverse flows respectively. The forward flow is used as test flow.

Network 3

Fig. 5.2 illustrates Network Topology 3 used in experiments with a lossy reverse path. The bottleneck bandwidth is 0.5Mbps with a propagation delay of 10ms while the outer links have 10Mbps capacity with 2ms propagation delay. A single sender (Host A) sends traffic to a single receiver (Host B). Data traffic flow is in one direction and this single flow is used as our test flow. The reverse link between router R1 and Host A is designed to be lossy, hence cause packet drops even in the absence of congestion.

Figure 5.1: Network Topology 2

Figure 5.2: Network Topology 3

5.2.2 Queue parameters

RED queue management was used in all gateways with or without support for BECN or ECN.

For Network 2, guidelines by Floyd [6] were followed in setting the RED parameters as follows: $minth = 15KB$, $maxth = 3*minth$, $buffer\ size = 2*maxth$, $maxp = 0.1$, $wq = 0.002$. For Network 3, the parameters used were: $minth = 5KB$, $maxth = 3*minth$, $buffer\ size = 2*maxth$, $maxp = 0.1$, $wq = 0.002$. Byte-based dropping for RED with FIFO service was used. Priority service [29] is not used in the gateways.

5.2.3 Traffic sources

In our experiments with Network 2, long-lived TCP traffic sources were emulated using the FTP traffic model in the Network Simulator (NS) with an infinite amount of data to send. TCP type is NewReno with a data packet size of 1000 bytes and ACK packet size of 40 bytes. The maximum TCP congestion window for the background flows was set to 100KB while that of the test flows was set to 500KB to ensure that in all experiments the TCP transfers were in the order of the bottleneck bandwidth-delay product of the link.

For experiments with Network 3, the same traffic source model described above is used and the maximum TCP congestion window is set to 100KB for the single connection.

5.2.4 Performance Metrics

The same performance metrics defined in section 3.5.4 were used for evaluation.

5.3 Experiments and Results

In this section, three sets of experiments carried out are described and explanations for the observed behavior of the algorithms (BECN, ECN and RED) is presented. All simulations are run for 500s and data for results were collected after a period of 100s to avoid bias due to initial conditions. For each of the algorithms, the same seed was used in all random number generators to ensure all algorithms are tested based on same input variables.

5.3.1 Experiment with small round-trip time (RTT) - [Fig. 5.3 - Fig. 5.13]

Network topology 2 (Fig. 5.1) is used in this experiment. In this experiment, three sets of simulations are done and in each simulation, FTP flows are either all plain TCP, all BECN-capable or all ECN-capable. The forward and reverse FTP connections are subjected to varied levels of congestions. The number of background flows for each congested link is varied using 5, 8, 10, 13, 15, 18, 23 and 25 flows to observe performance under different levels of congestion. The start times for all background flows is a random variable uniformly distributed between 0s and 5s, while start times for the forward and reverse flows varies uniformly between 20s and 25s. We measure the average goodput for the forward connection, the average value of the TCP congestion window of the forward connection, the mean and variance of the actual queue length for the gateway at the congested reverse link (both gateways were observed to have similar queue statistics for all experiments with Network topology 1 since the network is symmetrical), utilization on the congested forward link, and the percentage loss of packets (data, ACKs, ISQs) for our test flow (forward connection). Loss of data packets is monitored on the forward congested link (link 3 in Fig. 5.1) while loss of ACKs and ISQs are monitored on the

Figure 5.3: Average goodput

Figure 5.4: Average congestion window

reverse congested link (link 1 in Fig. 5.1). For the simulation with BECN-capable flows the percentage of ISQ reverse traffic is also recorded. Fig. 5.3 - Fig. 5.13 summarize the observed results. An interesting observation with the results is that the measured average goodput for BECN, ECN and plain TCP flows appear to be directly proportional to the measured average congestion window under all levels of congestion (Fig. 5.3 and Fig. 5.4).

Under low congestion with 5 background flows, the average congestion window for BECN is 703.1 bytes (4.17%) larger than that of ECN (Fig. 5.4), thus offering an average goodput 6.97% higher (Fig. 5.3). We ascribe the slightly better performance of BECN to its lower variance in queue length (Fig. 5.6) reducing the impact of ACK compression which affects the clocking of data packets with window flow control mechanisms and degrades performance of TCP throughput [22] [29]. However ECN, BECN, and plain TCP all achieve similar levels of utilization on the congested forward link (Fig. 5.7). With more than 8 background flows we observe in Fig. 5.5 that the mean queue length for plain TCP is lower than that of both BECN and ECN. Congestion avoidance for

Figure 5.5: Mean of actual queue size

Figure 5.6: Variance of actual queue size

Figure 5.7: Bottleneck utilization

Figure 5.8: Data packet loss

Figure 5.9: 25 background TCP flows

Figure 5.10: 25 background ECN flows

Figure 5.11: 25 background BECN flows

Figure 5.12: ISQ loss

plain TCP is based on RED dropping and this results in lower queue size compared to the explicit congestion notification schemes. It also results in 1.46% loss in data packets for the plain TCP flow while BECN and ECN flows record 0% loss under low congestion - 8 background flows (Fig. 5.8). BECN achieves 0% ISQ packet loss (Fig. 5.12) at this level of congestion while the measured variance in queue length (Fig. 5.6) show that ECN experiences upto 24.5% greater variance in queue length over BECN. This further validates deductions from the mathematical model developed in section 3.4.3. Performance of delay-sensitive applications can be particularly affected by significant variation in queue delay.

Under medium level congestion (15 and 18 background flows), it is observed that both the loss of ISQs on the reverse path (Fig. 5.12) and the contribution of BECN ISQs to reverse traffic (Fig. 5.13) becomes crucial to BECN performance. The ISQs add to BECN traffic and since they get delayed in the queues and sometimes lost, congestion notification and control is delayed resulting in a higher mean queue length for BECN compared to ECN in this scenario (Fig. 5.5). However, results show that the average goodput for BECN is not degraded compared to that of ECN (Fig. 5.3).

Under high congestion (25 background flows), the contribution of the BECN ISQs to BECN traffic (Fig. 5.13) reduces as the number of data packets and ACK packets that traverse the network also increase with congestion. BECN experiences 0.475% loss of ISQs (Fig. 5.12) and this causes significant variation in BECN queue size upto 55.2% greater than that of ECN (Fig. 5.6) resulting in 0.41% packet loss for BECN (Fig. 5.8). Plain TCP maintains the lowest queue size (Fig. 5.5) as congestion avoidance for plain TCP is based on RED packet dropping unlike the explicit notification schemes. Remarkably, though the loss of BECN ISQs cause fluctuations in queue size that can impact performance of delay-sensitive applications, the average congestion window and goodput for long-lived FTP flows (Fig. 5.3 and Fig. 5.4) seem not to be affected significantly for the

Figure 5.13: Percentage of ISQ on reverse path

Figure 5.14: Average goodput

scenario considered here. This observation suggests that though some ISQs are lost on the reverse path, since ISQs continue to be generated irrespective of whether a previous one was sent, congestion notification is eventually achieved even though not as early as would be experienced under periods of lower congestion. Further investigation to gain more insight into the effects of the loss of reverse flowing packets is carried out in section 5.3.3.

5.3.2 Experiment with large round-trip times - [Fig. 5.13 - Fig. 5.20]

In this section, the experiments in section 5.3.1 are repeated with propagation delay on the congested links of Network topology 2 (Fig. 5.1) increased to 120ms. The number of background flows for the experiments is varied using 10, 15, 18, 20, 23, 25 and 30 flows. Fig. 5.13 - Fig. 5.20 summarize the observed results. We observe that due to early notification, the BECN flows are able to maintain their queue size below maximum RED threshold (Fig. 5.20), resulting in a 0% packet loss (Fig. 5.15) even under high congestion

Figure 5.15: Data packet loss

Figure 5.16: Congestion window

Figure 5.17: Bottleneck utilization

Figure 5.18: 30 background RED flows

Figure 5.19: 30 background ECN flows

Figure 5.20: 30 background BECN flows

(with 30 background flows). The ECN flows wait more than half RTT before responding to congestion thus we observe that their queue size occasionally exceeds the maximum threshold (Fig. 5.19) resulting in 0.602% loss (Fig. 5.15). The BECN algorithm therefore offers better network utilization (Fig. 5.17) and 12.5% gain in average goodput over ECN (Fig. 5.14) under high congestion (25 background flows).

5.3.3 Lossy reverse path - [Fig. 5.21 - Fig. 5.32]

This test case investigates the impact of loss of packets flowing in reverse direction (ISQs, ECN-Echo ACKS, and ordinary ACKs) on the behavior of BECN, ECN and plain TCP. The issue of packet loss on the reverse path in a network can be crucial to TCP performance especially in wireless networks [25]. A simple topology - Network Topology 3 (Fig. 5.2) is used with a single TCP connection and a lossy reverse path to enable proper observation of the impact of losing reverse flowing packets on TCP performance. Three sets of simulations are done in the experiment. In each simulation, the FTP test flow is either plain TCP, BECN-capable or ECN-capable. The loss rate

Figure 5.21: Average goodput

Figure 5.22: Loss on reverse path-BECN

on the lossy link is varied using loss rates of 0%, 2%, 4%, 6%, 8%, 10% to observe performance with different loss rates on the reverse path. The start times for the test flow varies uniformly between 0s and 5s. The average goodput for the test flow, the mean and variance of the queue size for the gateway, and the percentage loss of packets in the reverse direction are measured while the TCP congestion window is monitored. Fig. 5.21 - Fig. 5.32 summarize the observed results. The results show that the percentage

loss for ISQs, ECN-Echo ACKs and ordinary ACKs increase with increase in loss rates (Fig. 5.22 and Fig. 5.23). It is observed that for ECN, the loss of ECN-Echo ACKs have insignificant impact on the mean and variance of the average queue size (Fig. 5.24, Fig. 5.25 and Fig. 5.30) as well as on the average goodput of the test flow (Fig. 5.21) since the results obtained with the ECN flow is similar to those obtained from simulations with the plain TCP flow (Fig. 5.21, Fig. 5.24, Fig. 5.25, and Fig. 5.32) which suffers only from loss of ordinary ACKs. This is because the ECN-Echo ACKs used for congestion notification are continuously generated until an ECN CWR data packet is received, thus the ECN sender is reliably notified of congestion. This reasoning is also supported in

Figure 5.23: Loss on reverse path-ECN

Figure 5.24: Mean of average queue

Figure 5.25: Variance of average queue

Figure 5.26: Average congestion window

Figure 5.27: Effect of ISQ loss on BECN window
- 10% lossrate

Figure 5.28: Effect of ISQ loss on BECN queue
- 10% lossrate

Figure 5.29: Effect of ECE ACK loss on ECN window
- 10% lossrate

Figure 5.30: Effect of ECE ACK loss on ECN queue
- 10% lossrate

Figure 5.31: Effect of ACK loss on TCP window Figure 5.32: Effect of ACK loss on TCP queue -
- 10% lossrate 10% lossrate

(Fig. 5.29) where it is observed that though several ECN-Echo ACKs are lost, the ECN congestion window still gets halved just like that of the plain TCP flow (Fig. 5.31). For BECN, we observe that the mean and variance of queue size increases with increase in loss rates over 4% (Fig. 5.24 and Fig. 5.25). Fig. 5.27 and Fig. 5.28 shows that this is due to loss of ISQs which result in temporal surges in the BECN congestion window since congestion notification is not achieved early.

5.4 Conclusion and Future work

In summary, the experiments with two-way traffic show that due to the relative delay in congestion notification ECN results in greater fluctuations in queue size under low and medium congestion levels compared to BECN thus further validating our developed mathematical model in section 3.4 (see equation 3.17). Results from the experiment with a lossy reverse path show that loss of ISQs on the reverse path can delay congestion notification for BECN and result in significant fluctuations in queue size. However for

ECN, loss of ECN-Echo ACKs does not significantly impact performance since the ECN-Echo ACKs are continuously generated and eventually delivered.

For future work, there is interest in researching ways of improving the robustness of BECN in the presence of a lossy reverse path. The next chapter details research work done in this area.

Chapter 6

ECN+BECD mechanism for Explicit Congestion Notification

6.0.1 Introduction and motivation

Research results presented in previous chapters show that while BECN enjoys the benefits of early notification over ECN it also suffers from the impact of loss of ISQs on the reverse path. On the other hand, the ECN mechanism enjoys the reliable delivery of congestion notification and is not adversely affected by a lossy reverse path. Given that both BECN and ECN mechanisms have certain unique advantages, the question arises as to whether or not there is any benefit in combining the two mechanisms for improved congestion control in TCP/IP networks. A combined approach would be useful if it could take advantage of BECN's performance enhancements while mitigating the issues associated with ISQ loss and the additional network load generated by ISQ packets.

One approach to combining ECN and BECN would be for RED to always generate ISQ packets and mark the ECN CE bit whenever it seeks to notify the sender of incipient congestion. In this case, BECN would be the primary mechanism for congestion notification with ECN as the backup mechanism. While this would address the issue of

ISQ loss, it would not address the issue of additional network load on the network.

A second approach to combining ECN and BECN would be for both mechanisms to co-exist and be used for different purposes. ECN would be used for low to medium level congestion while BECN would be used for the medium to high level congestion. When BECN is generated, ECN congestion control mechanisms would be used as a backup in case of ISQ loss. The benefit of this solution is that it reduces ISQ traffic on the network. Such a scheme would need to have a configurable threshold which would trigger BECN ISQ generation by the router in addition to ECN marking of the data packet.

This chapter presents a novel algorithm for congestion control that focuses on the second approach of combining the ECN and BECN schemes (called ECN+BECN mechanism) as it better addresses the issue of reducing ISQ reverse network traffic and ISQ loss on the reverse path. Simulations are performed to facilitate a comparative evaluation of the ECN+BECN, ECN and BECN mechanisms.

6.1 Proposed ECN+BECN algorithm

As mentioned previously, it is anticipated that combining BECN and ECN mechanisms can result in a robust performance-enhanced congestion control for IP networks. There could be different ways of combining the two schemes but the scheme described in this section is intended to apply ECN to low congestion scenarios and BECN to high congestion scenarios. A new configurable threshold between RED's *minth* and *maxth* is required in order to identify the point at which BECN is triggered. Thus, in one extreme, the new threshold could be set equal to *minth*, thus making the combined scheme operate exactly like BECN alone. In the other extreme, the new threshold could be set to equal *maxth*, thus making the combined scheme operate like ECN alone. In the following subsections, the proposed mechanism is described in more details and guidelines for the behavior of all hosts in an ECN+BECN capable network are explained.

6.1.1 Behavior of an ECN+BECN-capable router

For an ECN+BECN-capable router a new threshold *becnthresh* is defined, henceforth referred to as “BECN threshold”. The BECN threshold has a value between the minimum and maximum RED thresholds and is used as a basis for determining when ISQs should be generated for marked and/or dropped packets. When the average queue size is below the BECN threshold the router behaves as a pure ECN router marking packets based on RED probability and forwarding such packets. Under heavier congestion when the average queue size exceeds the BECN threshold ISQs are also generated for every marked and/or dropped packet in addition to the ECN forwarding. Hence the router essentially behaves as a pure ECN router under low congestion and under heavier congestion both ECN and BECN mechanisms are used for congestion control.

For a new arriving packet, the ECN+BECN-capable router behaves as follows:

If (*arriving packet causes the average queue size to go*
above maximum RED threshold)
{
 drop the packet just like an ECN packet;
 if (*ECT¹ bit was marked in the IP header*)
 {
 send an ISQ due to a dropped packet back
 to the source;
 }
} else if (*arriving packet causes average queue to go*
between minimum RED threshold and becnthresh){
 if (*RED rules chooses this packet for marking and ECT*

¹The ECN-Capable Transport (ECT) bit is set in the IP header of an ECN or BECN IP packet [11] for identification at the router.

```

    bit is set) and if (packet is not already marked)
    {
        mark the packet (CE2 bit) but do not send an ISQ;
    }
} else if (arriving packet causes average queue to go
between becnthresh and maximum RED threshold){
    if (RED rules chooses this packet for marking and ECT
bit is set) and if (packet is not already marked)
    {
        mark the packet (CE bit) and send an ISQ due to a
        marked packet back to the source;
    }
    else if (RED chooses this packet and ECT bit is not set)
        drop the packet;
}

```

6.1.2 Behavior of ECN+BECN-capable TCP end hosts

ECN+BECN TCP end hosts perform initial end-to-end negotiations to establish ECN capability just like pure ECN hosts. At the same time, the sender is responsive to both ECN and BECN congestion notifications. Under low congestion when the average queue is below *becnthresh* the end hosts respond to ECN signalling as pure ECN hosts. When congestion gets heavier and the average queue size exceeds *becnthresh*, RED's selection of a packet for marking would also cause an ISQ to be sent. As with ECN, when the marked packet arrives at the receiver, an ECE-ACK is generated to

²The Congestion Experienced (CE) bit is set in the IP header of an ECN or BECN IP packet [11] as an indication of congestion.

the sender. Whichever notification (ISQ or ECN-Echo ACK) reaches the sender first should cause a window reduction. The first packet to be sent after window reduction will have Congestion Window Reduced (CWR) bit set to stop the receiver from sending more ECE-ACKs. Hence, the sender on receipt of an ISQ due to a marked packet before the corresponding ECE-ACK, reduces its congestion window and *ssthresh* to one-half of current congestion window. It sends a CWR packet as allowed by the reduced window and waits a full roundtrip time (RTT) after window reduction before it starts increasing the window. An ISQ due to a dropped packet also causes the sender to set its congestion window and *ssthresh* to one-half of current congestion window and follow the TCP congestion avoidance algorithm thereafter. If for some reason the ECE-ACK reaches the sender before the BECN ISQ, sender responds like a pure ECN sender. The sender does not respond to congestion notification more than once per RTT whether due to ISQs, ECE-ACKs or duplicate ACK's. The advantage of using the BECN scheme under a configurable level of congestion is *early notification* especially for congestion levels that lead to packet drops since the sender does not wait for duplicate acknowledgements or a timeout before responding to congestion. Also the use of ECN in addition to BECN ensures that for scenarios where BECN ISQs are lost the ECN-Echo ACKs guarantees reliable delivery of congestion notification.

The ECN+BECN algorithm also requires the use of a single bit to differentiate between an ISQ due to a marked packet and an ISQ due to a dropped packet. An ISQ due to a marked packet would have this bit set so that the sender detects it should wait a full RTT before increasing its window according to the TCP algorithm. When the bit is unset, the sender assumes the ISQ was generated due to a dropped packet. It halves its congestion window and starts increasing the window according to the TCP algorithm similar to ECN.

In chapter three the use of a single bit in the 32 bit unused field in the ICMP source

quench packet header for the ISQ differentiation was proposed.

6.2 Simulation Setup

Study and evaluation of the ECN+BECN mechanism is done using the Network Simulator (ver 2.1b8a) [21]. In this section, the network scenarios, simulation parameters and performance metrics used in evaluating the algorithm are described.

6.2.1 Simulation Topology

The same network topologies (Network 1, Network 2 and Network 3) described previously in section 3.5.1 and 5.2.1 as well as their respective network parameters were used in this section for simulations.

6.2.2 Performance Metrics

Fairness Index for web object transfer delay

For N transferred objects the Fairness Index (FI) [30] for transfer delay is computed using the following formula:

$$FI = \frac{(\sum_{i=1}^N T_i)^2}{N \sum_{i=1}^N T_i^2} \quad (6.1)$$

where T_i is the transfer delay incurred by object i .

Other performance metrics used in evaluation have been previously defined in section 3.5.4.

6.3 Experiments and Results

In this section, five sets of experiments are described and explanations for the observed behavior of the ECN+BECN algorithm is presented. The same set of seeds used for

previously done BECN and ECN experiments are used in all random number generators to ensure the ECN+BECN algorithm is tested based on same input variables. For experiments in section 6.3.1, 6.3.2, 6.3.3, 6.3.5 two results for ECN+BECN are shown each corresponding to simulations using two different values for *becnthresh* (20KB and 40KB) to enable observation of the impact of varying *becnthresh* on TCP performance. Previously obtained results for ECN and BECN are also shown alongside results from this experiment to facilitate comparative evaluation.

6.3.1 Competing long-lived ECN+BECN flows with plain TCP flows - [Fig. 6.1 - Fig. 6.2]

Network Topology 1 (Fig. 3.7) is used in this experiment. A number of ECN+BECN FTP flows compete with an equal number of plain TCP FTP flows. The total number of competing flows is varied using 8, 12, 16, 18, 20, 26, 30, 36, 40 and 46 flows, to observe performance under different levels of congestion. The start time for all flows is a random variable uniformly distributed between 0s and 5s. Fig. 6.1 shows the measured gain in average goodput for ECN+BECN flows over plain TCP flows while Fig. 6.2 shows the measured losses. Previously obtained results for ECN and BECN are also shown.

First, it is observed in Fig. 6.1 that due to early notification BECN and ECN+BECN with 20KB *becnthresh* attain 45.97% and 45.43% gain in average goodput over plain TCP respectively under high congestion (46 flows). ECN and ECN+BECN with 40KB *becnthresh* attain only upto 34.58% gain and 40.48% gain respectively. Second, it is observed in Fig. 6.2 that BECN and ECN+BECN with 20KB *becnthresh* due to early congestion notification experience a loss of only 0.41% and 0.42% respectively under high congestion while ECN and ECN+BECN with 40KB *becnthresh* experience higher losses of 0.57% and 0.46% respectively.

Figure 6.1: Average goodput gain over TCP

Figure 6.2: Percentage loss

6.3.2 Homogeneous long-lived flows - [Fig. 6.3 - Fig. 6.6]

Network Topology 1 (Fig. 3.7) is also used in this experiment. The flows are called “homogeneous” as all flows in the experiment are ECN+BECN capable. The FTP flows start randomly within the initial 5s of simulation. The number of FTP connections is varied using 10, 15, 25, 30, 35, 40, and 45 flows. The average goodput for the flows, percentage packet loss, bottleneck utilization and percentage ISQ reverse traffic are measured. Fig. 6.3 - Fig. 6.6 summarizes the observed results. The results show that with a lower *becnthresh* (20KB) the ECN+BECN mechanism performs much like BECN as expected achieving a bottleneck utilization of between 99.2% and 99.9% (Fig. 6.3). With a higher *becnthresh* (40KB) ECN+BECN behaves more like ECN achieving a lower utilization of between 96.8% and 99.85% (Fig. 6.3). First, it is observed that *the percentage ISQ reverse traffic is significantly reduced compared to BECN (Fig. 6.4) since ISQs are generated only when average queue exceeds becnthresh*. With 40KB *becnthresh* we measure upto 38.8% reduction (Fig. 6.4) in ISQ reverse traffic under high congestion - 45 FTP flows however the average goodput and loss percent for ECN+BECN flows show

Figure 6.3: Bottleneck utilization

Figure 6.4: ISQ reverse traffic

no significant improvement over ECN and BECN for this test case. Second, compared to ECN it is observed that due to early ISQ notification ECN+BECN with 20KB *becnthresh* experienced lower variation in queue size (Fig. 6.5 and Fig. 6.6). This result supports earlier deductions from the developed mathematical model in section 3.4.3.

6.3.3 Experiment with two-way traffic - [Fig. 6.7 - Fig. 6.8]

Previous studies have shown that the dynamics of more realistic two-way traffic in a network can significantly impact the performance of congestion avoidance algorithms [22] and in chapter five the impact of this on BECN, ECN and RED were evaluated. In this experiment a comparative study of the behavior of the ECN+BECN algorithm in the presence of two-way traffic is done to understand how the algorithm offers improved performance over ECN and BECN mechanisms in this scenario. Network topology 2 (Fig. 5.1) is used in this experiment and all FTP flows are ECN+BECN capable. The forward and reverse FTP connections are subjected to varied levels of congestions. The number of background flows for each congested link is varied using 5, 8, 10, 13, 15, 18,

Figure 6.5: Queue - 15 ECN FTP

Figure 6.6: Queue - 15 ECN+BECN (20KB) FTP

Figure 6.7: Variance of actual queue size - two-way traffic

Figure 6.8: ISQ loss - two-way traffic

23 and 25 flows to observe performance under different levels of congestion. The start times for all background flows is a random variable uniformly distributed between 0s and 5s, while start times for the forward and reverse flows varies uniformly between 20s and 25s. The variance of the actual queue length for the gateway at the congested reverse link and the percentage loss of ISQ packets on the reverse congested link are measured. Fig. 6.7 and Fig. 6.8 summarizes our observed results.

Previous results (section 5.3.1) from experiments with ECN and BECN show that under low and medium congestion (between 5 and 18 background flows), ECN results in greater fluctuation in queue size due to its relative delay in responding to congestion. Under high congestion (25 background flows) BECN too suffers loss of ISQs (Fig. 6.8) thus delaying congestion notification. This results in greater variation in queue size for BECN (Fig. 6.7). However, results with the ECN+BECN mechanism show that depending on the configured *becnthresh* it is possible to circumvent both of these observed undesirable behaviors in ECN and BECN. Fig. 6.7 shows that with a 20KB *becnthresh* the variation in queue size is lower than that obtained for both ECN and BECN. This is essentially due to *early BECN notification* combined with *reliable delivery of congestion notification* (via ECN-Echo ACKs) even if BECN ISQs are lost under high congestion. Hence the ECN+BECN mechanism combines the advantages of both ECN and BECN schemes in this scenario.

6.3.4 Lossy reverse path - [Fig. 6.9 - Fig. 6.13]

This test case investigates the impact of loss of ISQ packets on the behavior of ECN+BECN by controlling the loss rate on the reverse traffic path. We use a simple topology - Network Topology 3 (Fig. 5.2) with a single ECN+BECN connection and a lossy reverse path to enable detailed observation of the impact of losing reverse flowing packets on the performance of ECN+BECN. The loss rate on the lossy link is varied using loss rates

Figure 6.9: Effect of ISQ loss on BECN window
- 10% lossrate

Figure 6.10: Effect of ISQ loss on BECN queue
- 10% lossrate

Figure 6.11: Effect of ISQ loss on ECN+BECN window
- 10% lossrate

Figure 6.12: Effect of ISQ loss on ECN+BECN queue
- 10% lossrate

Figure 6.13: Average goodput - lossy reverse path

Figure 6.14: Average object transfer delay

of 0%, 2%, 4%, 6%, 8%, 10% to observe performance with different loss rates on the reverse path. The start times for the test flow varies uniformly between 0s and 5s. We measure the average goodput for the test flow, the average queue size for the gateway and monitor the TCP congestion window. Fig. 6.9 - Fig. 6.13 summarizes our observed results along with previously obtained results for ECN and BECN in section 5.3.3.

First, Fig. 6.13 shows that for the four algorithms, average goodput decreased steadily with increase in loss rate and no significant difference in average goodput was recorded between the algorithms. Second, Fig. 6.9 shows that due to loss of ISQs the BECN congestion window experiences temporal surges since congestion notification is not achieved early. Fig. 6.10 also shows that this effect causes considerable fluctuation in average queue size. ECN however is less severely affected due to reliable delivery of ECN-Echo ACKs. Third, Fig. 6.11 and Fig. 6.12 for our experiment with the ECN+BECN mechanism shows that the ECN+BECN scheme enjoys this merit of ECN and hence is not adversely affected by loss of BECN ISQs unlike pure BECN.

6.3.5 Short-lived web transfers - [Fig. 6.14 - Fig. 6.18]

This test case assesses the performance of ECN+BECN with short-lived web-traffic workloads. In the experiment, there are 10 ECN+BECN-capable web servers and 10 ECN+BECN-capable web clients, while homogeneous ECN+BECN background FTP connections are varied using 5, 10, 15, 20, 25, 30, and 35 flows, to establish different levels of congestion. The FTP flows start randomly within the initial 5s of simulation while the web-traffic connections start after 50s. Short-lived traffic sources were emulated using the built-in web-traffic model in NS with parameters in Table 1. According

Table 6.1: WEB Traffic Parameters

Parameter value	Distribution	Mean	Shape
Intersession time	Exponential	0.5s	-
Session size	Constant	10 pages	-
Inter-page time	ParetoII	10s	2
Pagesize	ParetoII	3 objects	1.5
Inter-object time	ParetoII	0.1s	1.5
Object size	Constant	8 packets	-
Number of sessions	Constant	1000	-

to Internet traffic studies in [31] the average size for an HTTP object is between 8KB - 10KB hence we used a constant web object size of 8KB in our experiments with HTTP traffic. Having all objects with same size in a particular experiment permitted measurement of the fairness index for object transfer time. The intersession time, number of sessions, session sizes, and inter-object time were chosen to ensure that several sessions were active throughout simulation time, while other parameters were chosen based on recommendations in [18].

We record the average object transfer delay for the web transfers, fairness in web object transfer times, the percentage loss for HTTP packets, the mean and variance

Figure 6.15: HTTP packet loss

Figure 6.16: Fairness - Object transfer delay

of queue size for the congested router. Fig. 6.14 - Fig. 6.18 show the observed results alongside results for ECN. Fig. 6.15 shows that the burstiness of the web-traffic workloads causes packet loss for both ECN+BECN and ECN connections. Contributors to transfer delay (Fig. 6.14) include queue size (delay), packet sending rate, and packet loss which leads to TCP timeout and retransmission while variations in queue size contribute to unfairness in average transfer delay. First, we observe that due to ISQ early notification ECN+BECN experiences lower mean queue size with less queue fluctuations (Fig. 6.17 and Fig. 6.18) compared to the ECN. Second, in comparison to the ECN+BECN scheme with 20KB *becnthresh*, ECN experiences upto 18.9% greater variance in queue size with 15 background FTP flows (Fig. 6.18) resulting in 14.8% poorer fairness index (Fig. 6.16). Third, as a result of a 4.9% higher mean queue size with 15 background FTP flows (Fig. 6.17) ECN experiences 47.4% greater packet loss (Fig. 6.15) and 7.47% greater average transfer delay (Fig. 6.14) compared to ECN+BECN with a 20KB *becnthresh*.

Figure 6.17: Mean of queue - web transfers

Figure 6.18: Variance of queue - web transfers

6.4 Conclusion and Future work

In this study we proposed the combination of the ECN and BECN mechanisms to offer a robust congestion notification algorithm for TCP/IP networks. Significant observations for experiments in this chapter include the following: (1) For long lived TCP flows, there is a significant improvement in goodput for competing flows under heavy congestion. (2) The ECN+BECN scheme can result in significant reduction in ISQ reverse traffic compared to the BECN mechanism. (3) The combined ECN+BECN scheme results in reduced queue size variation as compared to ECN. (4) Unlike BECN, the ECN+BECN mechanism is not adversely affected by the loss of ICMP Source Quench packets on the reverse path. This is due to reliable congestion notification via ECN-Echo ACKs. (5) Finally, we show that for web traffic, the ECN+BECN mechanism can offer improved average transfer delay and fairness compared to ECN.

Therefore, we conclude that a robust mechanism which combines the merits of BECN and ECN algorithms for congestion control in IP networks can result in improved performance over the individual ECN and BECN schemes in terms of end-to-end metrics

like transfer delay and fairness for webtraffic objects. In addition, there is enhanced performance in terms of network-centric metrics like average queue size and variations in queue size which can impact performance of delay-sensitive applications sharing the same queue.

The results in this chapter warrant further investigation into the combined ECN+BECN scheme. One such area of future work is to study the impact of different *becnthresh* settings and develop guidelines for its preferred values. Finally, the ECN+BECN mechanism would benefit from further study, investigation and prototyping in routers and end systems.

Chapter 7

Summarized conclusion and Future Work

In chapter three of this thesis, modifications to a proposed Backward Explicit Congestion Notification (BECN) algorithm which uses ICMP Source Quench messages for backward congestion notification for IP networks [7] were suggested and simulations were used to explore the performance of the modified BECN algorithm. A mathematical model was also extended to include TCP throughput and RED queue behavior in BECN and ECN capable networks and this model was validated through simulation experiments. It was deduced from our model that due to the advantage of early notification, the BECN mechanism should offer less queue variation compared to ECN. Other deductions from our model predict that for paths without large delays the TCP throughput for flows in a BECN capable network should be about same as for flows in an ECN capable network. These predictions were validated using simulations.

The simulation results in chapter three suggest that the enhanced BECN mechanism for congestion control in TCP/IP networks enjoys the benefits of early congestion notification and can offer improved performance over the ECN mechanism for both long-lived

and short-lived TCP transactions. On links that have a high bandwidth-delay product, BECN offers significant improvement over the ECN mechanism in terms of average goodput for long-lived traffic and packet loss, transfer delay for short-lived workloads.

Therefore, it was concluded that the BECN mechanism is a viable scheme for congestion control in TCP/IP networks and can be used to offer improved quality of service on links with a high bandwidth-delay product.

It was anticipated in [7] that a BECN scheme that also carried information on the severity of congestion in the network so that TCP end hosts respond only according to the perceived level of congestion would result in better use of network resources. In chapter four of this thesis, a modified multi-level BECN (MECN) scheme which delivers congestion notification as well as indicates the severity of congestion to end hosts was suggested and experiments were carried out to evaluate its performance. Results from the experiments showed that for the MECN algorithm considered, there was no observed significant improvement compared to the BECN mechanism in terms of average goodput and bottleneck utilization for connections with small round trip times. In fact depending on the configured MECN threshold, significant levels of packet drops could result due to the aggressive behavior of the MECN algorithm as defined in this thesis. For paths with a large bandwidth-delay product it is observed that MECN can offer slight improvement in the utilization of the bottleneck link but this does not result in any significant improvement in goodput over the BECN mechanism. Despite these results, the opportunity for multi-level notification as provided by MECN is of great value and optimising the response of end hosts to multi-level congestion notifications is still a relevant issue for further research.

Previous studies has shown that the dynamics of two-way traffic can have significant impact on the behavior of congestion avoidance algorithms. Also, valid concerns have been raised about how the loss of BECN ICMP Source Quench congestion notification

messages would impact performance of BECN. In chapter five of this thesis, simulation experiments were done to evaluate the impact of two-way traffic and a lossy reverse path on the performance of BECN, ECN and RED algorithms. The experiments with two-way traffic showed that due to the relative delay in congestion notification ECN results in greater fluctuations in queue size under low and medium congestion levels compared to BECN thus further validating our developed mathematical model. Results from the experiment with a lossy reverse path show that loss of ISQs on the reverse path can delay congestion notification for BECN and result in significant fluctuations in queue size. However for ECN, loss of ECN-Echo ACKs does not significantly impact performance since the ECN-Echo ACKs are continuously generated and eventually delivered.

In view of these results, it can be stated therefore that while BECN enjoys the benefits of early notification over ECN it also suffers from the impact of loss of ISQs on the reverse path. On the other hand, the ECN mechanism enjoys the reliable delivery of congestion notification and is not adversely affected by a lossy reverse path. In other words, both BECN and ECN schemes have their peculiar advantages. This motivated the proposal of a novel mechanism (ECN+BECN mechanism) in chapter six of this thesis which uses both mechanisms for congestion notification and takes advantage of the individual merits of both mechanisms. Experiments were done to evaluate the performance of the ECN+BECN scheme and observations from experiments include the following: (1) For long lived ECN+BECN capable TCP flows, there was a significant improvement in goodput for experiments with competing flows under heavy congestion. (2) The ECN+BECN scheme can result in significant reduction in ISQ reverse traffic compared to the BECN mechanism. (3) The combined ECN+BECN scheme resulted in reduced queue size variation as compared to ECN. (4) Unlike BECN, the ECN+BECN mechanism is not adversely affected by the loss of ICMP Source Quench packets on the reverse path. This is due to reliable congestion notification via ECN-Echo ACKs. (5) Finally, results

showed that for web traffic, the ECN+BECN mechanism can offer improved average transfer delay and fairness compared to ECN.

Therefore, it was concluded that a robust mechanism which combines the merits of BECN and ECN algorithms for congestion control in IP networks can result in improved performance over the individual ECN and BECN schemes in terms of end-to-end metrics like transfer delay and fairness for webtraffic objects. In addition, there is enhanced performance in terms of network-centric metrics like average queue size and variations in queue size which can impact performance of delay-sensitive applications sharing the same queue.

The results in this thesis warrant further investigation into the combined ECN+BECN scheme. One such area of future work is to study the impact of different ECN+BECN thresholds (*becnthresh* settings) and develop guidelines for its preferred values. Finally, the ECN+BECN mechanism would benefit from further study, investigation and prototyping in routers and end systems.

Bibliography

- [1] J. Postel, “*Internet Control Message Protocol*”, RFC 792, September 1981.
- [2] A. Mankin, K. Ramakrishnan, “*Gateway Congestion Control Survey*”, RFC 1254, August 1991.
- [3] J. Baker, “*Requirements for IP Version 4 Routers*”, Internet RFC 1812, June 1995.
- [4] J. Nagle, “*Congestion Control in TCP/IP Internetworks*”, RFC 896, January 1984.
- [5] W. Prue, J. Postel, “*Something a Host Could do with Source Quench: The Source Quench Introduced Delay (SQuID)*”, Internet RFC 1016, July 1987.
- [6] S. Floyd, V. Jacobson, “*Random Early Detection Gateways for Congestion Avoidance*”, IEEE/ACM Transactions on Networking, V.1, N.4, August 1993.
- [7] S. J. Hadi, B. Nandy, N. Seddigh, “*A proposal for Backward ECN for the Internet Protocol (Ipv4/Ipv6)*”, Internet Draft <draft-salim-jhsbnnns-ecn00.txt> July 1998. Available at <http://www.sce.carleton.ca/~seddigh/publications/draft-salim-jhsbnnns-ecn-00.txt>.
- [8] V. Jacobson, M. J. Karels, “*Congestion Avoidance and Control*”, In Proceedings of ACM SIGCOMM '88, Stanford, CA, August 1988.
- [9] S. Floyd, “*TCP and Explicit Congestion Notification*”, ACM Computer Communication review, V.24, N.5, p.10-23, October 1994.
- [10] <ftp://ftp.ee.lbl.gov/email/sf.98may7.txt>
- [11] S. Floyd, Ramakrishnan K, “*A proposal to add Explicit Congestion Notification to IP*”, RFC 2481, January 1999.
- [12] A. Matrawy, I. Lambadaris, C. Huang, “*Multicasting of Adaptively-encoded MPEG4 on a QoS-aware IP Networks*” In Proc. of IEEE ICC 2002.

- [13] S. Floyd, "*Connections with Multiple Congested Gateways in Packet-Switched Networks Part 1: One-way traffic*", Computer Communication Review, Vol. 21, No. 5, pp. 30-47, October 1991.
- [14] B. Braden et al, "*Recommendations on Queue Management and Congestion Avoidance in the Internet*", RFC 2309, April 1998.
- [15] A. Rangarajan, A. Acharya, "*Early Regulation of Unresponsive Best-Effort Traffic*", ICNP 99, October 1999.
- [16] P. Newman, "*Traffic Management for ATM Local Area Networks*", IEEE Communications Magazine, Vol. 32, No. 8, pp. 44-50 August 1994.
- [17] R. Jain, S. Kalyanaraman, R. Viswanathan, "*The OSU Scheme for Congestion Avoidance in ATM Networks: Lessons Learnt and Extensions*", Issue on Traffic Control in ATM Networks, Vol. 31, November 1997.
- [18] A. Feldmann, A. C. Gilbert, P. Huang, W. Willinger, "*Dynamic of IP traffic: A study of the role of variability and the impact of control*", Proceedings of ACM/SIGCOMM, Cambridge, MA, September 1999.
- [19] S. J. Hadi, U. Ahmed, "*Performance Evaluation of Explicit Congestion Notification (ECN) in IP Networks*", RFC 2884, July 2000.
- [20] K. Fall, S. Floyd, "*Simulation-based Comparisons of Tahoe, Reno, and SACK TCP*", Computer Communications Review, July 1996.
- [21] UCB/LBNL/VINT Network Simulator (NS). Available at <http://www.isi.edu/nsnam/ns>
- [22] R. Wilder. K. Ramakrishnan, "*Dynamics of Congestion Control and Avoidance of Two-Way Traffic in an OSI Testbed*", SIGCOMM 91, April 1991.
- [23] L. Zhang and D. Clark, "*Obsevation on the Dynamics of a CongestiControl Algorithm: The effects of Two-Way Traffic*", SIGCOMM 91, Septmber 1991.
- [24] H. Balakrishnan, V. N. Padmanabhan, S. Seshan, and R. H. Katz, "*A comparison of mechanisms for improving tcp performance over wireless links*", IEEE/ACM Transactions on Networking, June 1997.
- [25] F. Anjum and R. Jain, "*Performance of TCP over Lossy Upstream and Downstream Links with Link-level Retransmissions*", Proc. IEEE/ICON September 2000.

- [26] H. Balakrishnan, S. Seshan, and R. H. Katz, "*Improving reliable transport and handoff performance in cellular wireless networks*", ACM wireless Networks, Dec 1995.
- [27] R. Caceres and L. Iftode, "*Improving the performance of reliable transport protocols in mobile computing environment*", IEEE/ JSAC, 1994.
- [28] M. Kwon, S. Fahmy, "*TCP Increase/Decrease Behavior for Explicit Congestion Notification (ECN)*," Proceedings of IEEE ICC (Symposium on High-Speed Networks), volume 4, pp. 2335-2340, April 2002.
- [29] S. Floyd, "*Connections with Multiple Congested Gateways in Packet-Switched Networks Part 2: Two-way traffic*", Computer Communication Review, Vol. 21, No. 5, pp. 30-47, December 1991.
- [30] R. Jain, "*The Art of Computer Systems Performance Analysis: Techniques for Experimental Design, Measurement, Simulation, and Modeling*", Wiley - Interscience, New York, NY, April 1991.
- [31] B. Mah, "*An Empirical Model of HTTP Network Traffic*", Proceedings of INFOCOM '97, Kobe, Japan, April 7-11, 1997.
- [32] Thompson K, Miller G. J, Wilder R, "*Wide-Area Internet Traffic Patterns and Characteristics*" , IEEE Networks magazine, November/December 1997.
- [33] J. Padhye, V. Firoiu, D. Towsley, and J. Kurose, "*Modeling TCP throughput: A simple model and its empirical validation*" in proceedings of the ACM SIGCOMM , September 1998, vol. 28, pp. 303-314, <http://gaia.cs.umass.edu/>.
- [34] V. Firoiu and M. Borden, "*A study of active queue management for congestion control*" in Proceedings of the IEEE INFOCOM, March 2000, <http://www.ieee-infocom.org/2000/papers/405.pdf>.
- [35] M. Allman, V. Paxson, W. Stevens., "*TCP Congestion Control*", RFC 2581, April 1999.
- [36] F. Akujobi, I. Lambadaris, R. Makkar, N. Seddigh, B. Nandy, "*BECN for Congestion Control in TCP/IP Networks: Study and Comparative Evaluation*", IEEE/Globecom 2002, Taipei, Taiwan, November 2002.
- [37] F. Akujobi, R. Makkar, I. Lambadaris, B. Nandy, N. Seddigh, "*Congestion Control in TCP/IP Networks: A combined ECN and BECN approach*" submitted to IEEE/INFOCOM 2003, San Francisco, California, April 2003.