International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 2 Issue 6, June - 2013

An Effective Congestion Control Mechanism For Communication Netwroks

J. Aruna S. Vishnupriyan

Abstract

Congestion is one of the major issues in the communication networks. One of the approaches towards congestion is Network Assisted Congestion Control. According to this approach the router provides feedback about congestion to the end system. Explicit Congestion Notification (ECN) provides end to end notification without dropping packets. eXplicit Control Protocol (XCP) uses ECN bits for feedback to control congestion. But it uses multiple bits in the header which faces certain obstacles in the implementation in the communication network. In this paper we propose a protocol called Reduced Explicit Congestion Notification (REN).It uses only two bits and provides better results compared to existing protocol XCP. Moreover the objective of this paper is to compare the existing eXplicit Control Protocol with the proposed Reduced Explicit Congestion Notification protocol and to provide the analysis report of the performance of the system.

1. Introduction

Congestion means high rate of input to a router than at the output. Congestion occurs due to many reasons. Congestive collapse is a condition in which a packet switched computer network can reach, when a little or no useful communication is happening due to congestion. The main effects of the congestion are delay and losses. Problems due to congestion are excessive queuing delays, wasted network capacity on retransmission, wasteful retransmit of prematurely time out packets. Problem of congestion can be solved before the congestion takes place or after it occurs. The method of handling congestion after the occurrence of it is termed as congestion control. Congestion control is the mechanism by which the network bandwidth is distributed across multiple end to end connections. The objective of congestion control is to limit the delay and buffer overflow caused by network congestion and to provide trade-off between efficient and fair resource allocation. The congestion control involves detecting the occurrences of congestion and limiting the sending rate of packets.

The two approaches towards the congestion control are end to end congestion control and Network Assisted Congestion Control. In the end to end approach, the congestion can be found from end to end system loss and delay. This approach is used by TCP. In the Network Assisted Congestion Control, router gives feedback about the congestion to the end system.TCP is the basic end to end transport protocol of the internet and its congestion control algorithm is fundamental to efficient, high-performance and stable network. However, Tcp becomes inefficient and prone to instability regardless of the queuing schemes. Traditional TCP congestion is widely used to control the loss of packet and prevent the congestion breakdown. But when the bandwidth delay increases, TCP becomes very unstable and surges easily. To address the limitation of end to end congestion control schemes, the explicit network feedback is used. The traditional congestion notification schemes as Active Queue Management and Explicit Congestion Notification proposals are successful in reducing the loss rate and the queue size in the network, they still falls short in achieving high utilization in the communication networks. eXplicit Control Protocol address this

problem by having routers which estimates the fair flow rate and send this rate back to the sender and achieves good performance. But the eXplicit Control Protocol is difficult to deploy in the networks .This is because the protocol uses multiple bits in the header. The performance of any protocol utilizing IP header bits to carry congestion information may be seriously crippled in communication networks without proper compensation against error-caused loss. It is a challenging task to use IP header bits to carry congestion information. The major source of performance degradation is the decoupling of efficiency and fairness. Efficiency involves the aggregate traffic behaviour when the input traffic rate equals the link capacity, no queue builds and utilization is optimal. Where, the fairness involves the relative throughput of flows sharing a link. A scheme is fair when the flows sharing a link have the same throughput irrespective of congestion. To address these issues, Reduced Explicit Congestion Notification protocol provides better results by using congestion header which carries only two bits and it also solves the performance degradation problem by decoupling efficiency and fairness effectively. The main goal of this paper is to deploy the existing eXplicit Control Protocol (XCP) and Reduced Explicit Congestion Notification (REN) protocol in Network Simulator and to create the graphs with appropriate parameters. With the help of graphs the performance of two protocols are need to compare.

2. Explicit control protocol (xcp)

2.1. Details about xcp

XCP is a feedback congestion control protocol that uses direct, explicit, router feedback to avoid congestion in the network. It makes use of Explicit Congestion Notification (ECN) bits.ECN provides end to end notification about occurrences of congestion without dropping packets.The end points consist of TCP source and sink agents using XCP as their congestion control mechanisms, the intermediate node or router writes feedback in each packet header about data capacity. When this packet reaches the receiver, the data capacity value is sent as reverse feedback to the sender. The sender upon receiving this reverse feedback value, adjusts its sending rate by increasing or decreasing its congestion window sizes as the case may be.

2.2. Architecture of eXplicit control protocol (xcp)

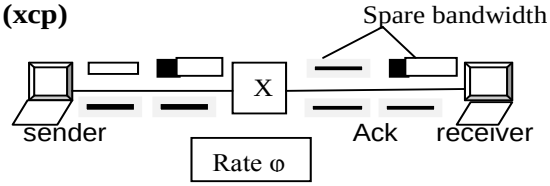

Figure 1.Architecture of xcp

The sender involves in sending the data packets along with the congestion header. The router involves in calculating the rate value, spare bandwidth value and attaches that information with the congestion header. The receiver upon receiving, it copies that information into the Acknowledgment (ACK). Those ACK packets will be sent to the sender, according to the information in the ACK packets, the sender increases or decreases its congestion window.

2.3. Xcp framework

The XCP framework consist of following

- The congestion header.
- The XCP Sender.
- The XCP Receiver.
- The XCP Router.
- The Efficiency Controller.
- The Fairness Controller.

2.3.1. The congestion header. Each XCP packet carries a congestion header which is used to communicate each flows state to routers and feedback from the routers on to the receiver. The field H_cwnd is the sender's current congestion window. The H_rtt is the sender's current RTT (Round Trip time) estimate. H_feedback takes positive or negative values and is initialized by the sender, router along the path modify this field to directly control the congestion window of the sender.

H_cwnd(Set to sender's current congestion window)
H_rtt(Set to sender's current round trip time)
H_feedback(Initialize to demands)

Figure 2.The congestion header

2.3.2 Xcp sender. On packet departure, the sender attaches a congestion header to the packet and sets the H_cwnd to its current congestion window. In the first packet of the flow, H_rtt is set to zero in order to indicate the router that the sender not yet has a valid RTT estimate. The sender initializes the H_feedback field to request its desired window increase. Whenever a new acknowledgement arrives, positive feedback increases the sender congestion window and negative feedback reduces it.

2.3.3. Xcp receiver. An XCP receiver is similar to a TCP receiver except that when acknowledging a packet, it copies the congestion header from the data packet to its acknowledgment.

2.3.4. Xcp router. The job of an XCP router is to compute the feedback to cause the system to converge to optimal efficiency and fairness. To compute the feedback, an XCP router uses Efficiency Controller and Fairness Controller.

2.3.5. Efficiency controller. The Efficiency Controller's purpose is to maximize links utilization while minimizing drop rate. As XCP is window-based, Efficiency Controller computes a desired increase or decrease in the number of bytes. The feedback is computed using the following formula.

$$\Phi=\alpha*d*S-\beta*Q \qquad (1)$$

Where α, β are constant values and is set to 0.4, 2.66, S is spare bandwidth value and Q is persistent queue value.

Φ is then used as feedback to add or subtract bytes. The equation makes the feedback proportional to spare bandwidth. When S>0, the link is underutilized so the router will send positive feedback. When S<0, the link is congested and so the router will send negative feedback. To achieve efficiency, we allocate the aggregate feedback to single packets as H_feedback.

2.3.6. Fairness controller. It uses Additive Increase Multiplicative Decrease (AIMD) to promote fairness. If $\varphi>0$, allocate it so that increase in throughput of all the flows are same. If $\varphi<0$, allocate it so that decrease of all the flows are proportional to its current throughput. When $\varphi=0$, bandwidth shuffling is used (i.e.) simultaneous allocation and deal location of bandwidth such that the total traffic rate does not change.

3. Reduced explicit congestion notification protocol

The proposed protocol is Reduced Explicit Congestion Notification (REN) protocol.REN remains a window based protocol and is designed to regulate the congestion window. This is an extension of the existing TCP protocol. REN uses different congestion control policies according to the level of congestion in the network. They are

1. Multiplicative Increase (MI) in the low-load region.
2. Additive Increase (AI) in the high-load region.
3. Multiplicative Decrease (MD) in the over-load region.

3.1. Multiplicative increase

If the link is underutilized which means traffic in the network is less and so the receiver involves in sending two bits (00). If the sender receives (00) as the feedback value it multiplicatively increases its sending rate.

3.2. Additive increase

In the case of occurrences of high traffic in the network, in this situation the receiver involves in sending the feedback value (01). Upon receiving the feedback value the sender additively increases its sending rate.

3.3. Multiplicative decrease

In the case of occurrences of heavy traffic in the network, the receiver involves in sending the feedback value (11). Upon receiving the feedback the sender multiplicatively decreases ifs sending rate.

3.4. Architecture of Reduced Explicit Congestion Notification Protocol

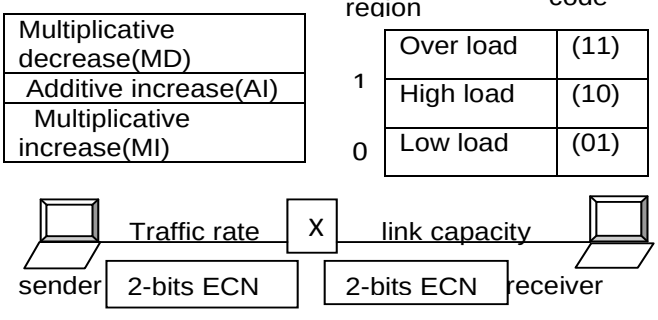

Figure 2. Architecture of ren

The sender sends the data packet through the router to the destination node. When the data packet reaches the router, it calculates the load factor for all flows. Based on the load factor value, it maps the congestion region. The receiver sends feedback via acknowledgment packet to the sender. Upon receiving it, the sender increases or decreases its transmission rate according to the feedback obtained. The whole process is continuously repeated and the transmission rates are changed according to the congestion identified.

3.5. Modules of reduced explicit congestion notification

The proposed protocol REN has three modules

- The REN Sender.
- The REN Queue.
- The REN Receiver.

3.5.1. REN sender. This module works on the sender's side. It is performs the following

- Initializing ECN values.
- Obtain the load factor from ACK packets.
- Apply appropriate congestion policy.

The congestion control policies are Multiplicative Increase for low load region, Additive Increase for high load region and Multiplicative Decrease for over load region.

3.5.2. Ren queue. Upon arrival of each packet REN capable router performs the following

- Computes load factor of each of its link.
- Maps each computed load factor to one of the three congestion levels.
- Updates ECN bits in the packet header.

More congested downstream router can further change the level of congestion by overwriting it.

3.5.3. Ren receiver. The receiver signals the sender with the congestion information. REN Receiver is responsible for sending the appropriate feedback value according to the traffic levels in the network.

3.6. Implementation details of ren. The main design goals of REN are to decouple efficiency and fairness and to use link load factor as the congestion signal. The key points are congestion control parameter setting and handling round trip time.

3.6.1. Load factor transition point. REN separates the network load condition into three regions. The load factor transition point is the boundary between the low load, high load and over load regions. The choice of transition point represents a trade off between achieving high link utilization and responsiveness to congestion.

The load factor should satisfy the following conditions,

- The transition point should be sufficiently high to enable the system to obtain high utilization.
- After the flows perform Multiplicative Decrease from an overloaded state, the Multiplicative Decrease should force the system to enter the high load state not low load state.
- The Multiplicative Increase in the case of low load region should lift the system into high load state, but not the overloaded state.

The load factor is the ratio of demand and capacity

i.e. Load Factor=Demand/Capacity

Load Factor = Arrival traffic + queue size/link bandwidth*tρ

3.6.2. Congestion control parameter settings. Using Multiplicative Increase for congestion control is often fraught with the danger of in-stability due to its large variations over short time scales. To maintain stability and to make sure that the aggregate rate of the REN flows using Multiplicative Increase does not overshoot the link capacity. Similarly, to achieve fairness, we need to make sure that a flow enters the Additive Increase phase before the link gets congested. In order to satisfy these criteria, we need an appropriate choice of Multiplicative Increase, Additive Increase and Multiplicative Decrease parameters that can achieve high utilization while maintaining stability, fairness and small persistent queues. To simplify the discussion, considering a single link shared by flows, whose RTTs are equal to the link load factor estimation period i.e. RTT = tρ such that all the flows have synchronous feedback and their intervals are also in synchronous with the link load factor estimation. At any time t, REN sender performs one of the three actions based on the value of the encoded load factor sent by the network,

$$MI: Cwnd(1+rtt) = Cwnd(t) + (1+\xi) \quad (2)$$

$$AI: Cwnd(1+rtt) = Cwnd(t) + \alpha \quad (3)$$

MD: Cwnd (1+rtt) = Cwnd (t) + β (4)

We require the Multiplicative Increase of the congestion window to be proportional to 1-ρ^ where ρ^ represents the current load factor. During the Multiplicative Increase phase, the current sending rate of each flow is proportional to the current load factor ρ^. Consequently we obtain, ξ=k (1-ρ^)/ρ^, where k = 0.25 (for stability).

3.6.3. Handling rtt with parameter scaling. In the case of different RTT, we need to scale the congestion control parameters used by the end-hosts according to their RTTs. To handle the case of flows with different RTTs, we set the scaled Multiplicative Increase/Additive Increase parameters ξs and αs as follows

$$\xi s \leftarrow (1+\xi)^{rtt/t\rho}-1 \quad (5)$$

$$\alpha s \leftarrow \alpha, rtt/t\rho \quad (6)$$

Multiplicative Decrease is not affected by the length of RTT. Hence the β value needs not to be scaled with RTT of the flows. However to avoid over-reaction of the congestion signal, a flow should perform a Multiplicative Decrease at most once during an estimation interval tρ

Table 1.Parameter value setting of REN

Symbols	Value	Meaning
tρ	200ms	The link factor measurement interval
tq	10ms	The link queue sampling interval
γ	0.98	The link target utilization
kq	0.5	How fast to drain the link queue
k	0.25	How fast to probe the available bandwidth
α	1.0	The AI parameter
β	0.875	The MI parameter

4. Results and discussion

In this paper, in order to reveal that the proposed protocol outperforms the Existing protocol we use the following characteristics such as End to End delay, Jitter, throughput and packet delivery ratio of both the eXplicit Control Protocol and Reduced Explicit Congestion Notification protocol are compared.

4.1. Comparison of end to end delay

End to End delay refers to the time taken for a packet to be transmitted across a network from source to destination.

4.1.1. End to end delay of xcp.

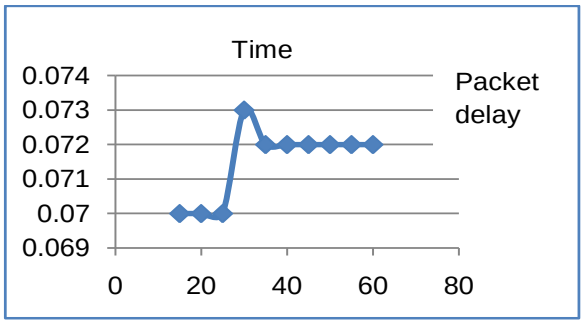

Figure 4.End to end delay of xcp

The End to End delay of eXplicit Control Protocol (XCP) seems to be less at the starting phase i.e. at the beginning of sending packet, as the time increases it getting increases and reaches the constant state.

4.1.2. End to end delay of reduced explicit congestion notification protocol.

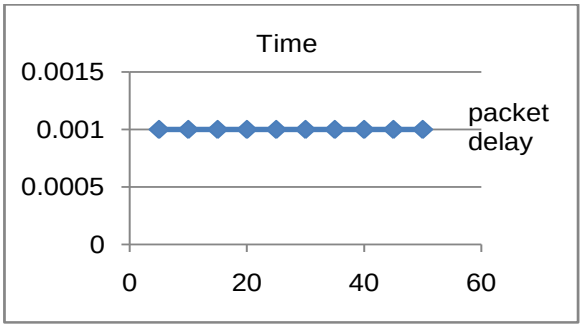

Figure 5.End to end delay of ren

The graph reveals that the End to End delay of REN is constant and it seems to be less when compare to XCP. Hence the packets will reach the destination node without any much delay.

4.2. Comparison of jitter

Jitter represents the variance of delay of transmitted packet.

4.2.1. Jitter of explicit control protocol (xcp).

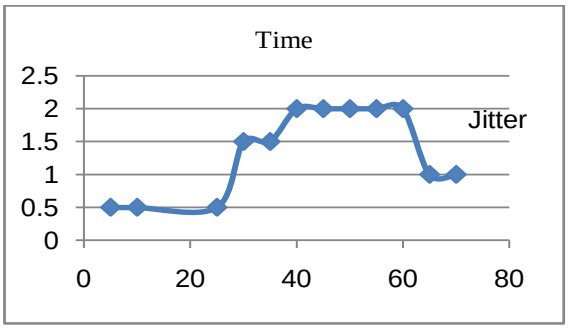

Figure.6 Jitter of xcp

The jitter increases step by step and it reaches an increased steady state, as the time increases it reaches the reduced steady state.

4.2.2. Jitter of reduced explicit congestion notification (ren).

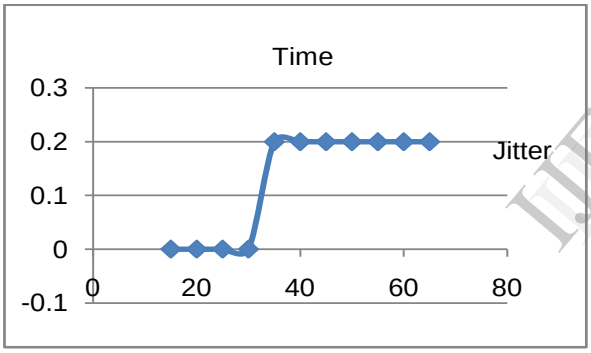

Figure.7 Jitter of ren

The Jitter of Reduced Explicit Congestion Notification seems to be less at the starting phase and as the time increase it increases slightly and reaches a steady state. When comparing the jitter of REN and XCP the jitter of REN seems to be less so that the packets will the reach the destination node in time without much delay.

4.3. Comparison of throughput

The throughput of the existing eXplicit Control Protocol in the single bottleneck topology is 3305.10.The throughput of the proposed Reduced Explicit Congestion Notification in the single bottleneck topology is 45623.76.Comparing the throughput of both the protocols clearly; the proposed protocol performs well when compared to existing protocol.

4.4. Comparison of Packet Delivery fraction

4.4.1. Packet delivery ratio of xcp. The ratio between the packets received and the packet sent is known as packet delivery ratio. The numbers of packets sent and received are calculated using trace file. The following output is obtained for packet delivery fraction of XCP protocol.

At link 22-----23 TCP Packet Delivery Fraction: 99.919679.

At link 2 ------ 3 TCP Packet Delivery Fraction: 99.919567.

At link 4 ------ 5 TCP Packet Delivery Fraction: 100.000000.

At link 6 ------ 7 TCP Packet Delivery Fraction: 99.949173.

At link 8 ------ 9 TCP Packet Delivery Fraction: 100.000000.

At link 12------13 TCP Packet Delivery Fraction: 99.809433.

At link 14------15 TCP Packet Delivery Fraction: 99.755292.

At link 16------17 TCP Packet Delivery Fraction: 99.983065.

At link 18------19 TCP Packet Delivery Fraction: 99.461207.

At link 20------21 TCP Packet Delivery Fraction: 100.000000.

At link 24------25 TCP Packet Delivery Fraction: 100.000000.

At link 26------27 TCP Packet Delivery Fraction: 99.886557.

At link 28------29 TCP Packet Delivery Fraction: 99.921553.

At link 30------31 TCP Packet Delivery Fraction: 100.000000.

At link 32------33 TCP Packet Delivery Fraction: 100.000000.

At link 34------35 TCP Packet Delivery Fraction: 99.804353.

At link 36------37 TCP Packet Delivery Fraction: 99.966239.

At link 38------39 TCP Packet Delivery Fraction: 99.462510.

At link 40------41 TCP Packet Delivery Fraction: 98.712999.

4.4.2 .Packet delivery ratio of ren. The numbers of packets sent and received are calculated using trace file. The following output is obtained for packet delivery ratio of REN protocol.

At link 22------23 TCP Packet Delivery Fraction: 99.968942.

At link 2 ------ 3 TCP Packet Delivery Fraction: 99.976307.

At link 4 ------ 5 TCP Packet Delivery Fraction: 100.000000.

At link 6 ------ 5 TCP Packet Delivery Fraction: 99.942937.

At link 8 ------ 9 TCP Packet Delivery Fraction: 100.000000.

At link 12----- 13 TCP Packet Delivery Fraction: 99.904489.

At link 14------15 TCP Packet Delivery Fraction: 100.000000.

At link 16------17 TCP Packet Delivery Fraction: 100.000000.

At link 18------19 TCP Packet Delivery Fraction: 100.000000.

At link 20------21 TCP Packet Delivery Fraction: 99.436620.

At link 24------25 TCP Packet Delivery Fraction: 100.000000.

At link 26------27 TCP Packet Delivery Fraction: 99.934692.

At link 28------29 TCP Packet Delivery Fraction: 99.939668.

At link 30------31 TCP Packet Delivery Fraction: 99.955277.

At link 32------33 TCP Packet Delivery Fraction: 100.000000.

At link 34------35 TCP Packet Delivery Fraction: 100.000000.

At link 36------37 TCP Packet Delivery Fraction: 99.665522.

At link 38------39 TCP Packet Delivery Fraction: 100.000000.

At link 40------41 TCP Packet Delivery Fraction: 100.000000.

On comparing the packet delivery ratio of both the XCP and REN protocols, the Packet delivery ratio of REN protocol seems to be higher than that of XCP protocol.

5. References

[1] Hairui Zhou, Guanzhong Dai, Faughong Ye and Huixiang Zhang".Improving xcp to achieve efficient and fair bandwidth allocation", IEEE GLOBECOM, 2009.

[2]Xiao long Homayoun Yousefi zadwh,"Distributed ecn-based congestion control", IEEE ICC, 2009.

[3]Ihsan Ayyub Qazi, Taieb Znati,"Congestion control using efficient explicit feedback", IEEE INFOCOM, 2009.

[4]Jiawei Huang and Jianxin Wang,"An ecn-based congestion control algorithm for tcp enhancement in performance computing and communication", 2009.

[5]Homayoun Yousefi'zadeh, Amir Habibi, Wojtek Furmanski."A performance comparison study of end-to-end congestion control protocol over mimo fading channels", IEEE INCOM 2009.

[6]Li Yun,Zhang Rudi,Liu Zhanjun, Liu Qilie,"An improved tcp congestion control algorithm over mixed wired/wireless Networks",IC-BNMT,2009.

[7]Zhipeng Huang Xialong Li Homayoun Yousefi'zadeh,"Robust ekf-based wireless congestion control", IEEE ICC, 2009.

[8]Christo Wilson, Chris Coakley and Ben Y.Zhao,"Fairness attacks in the explicit control protocol", IEEE, 2009.

[9]Ahmed Khurshid,Md.Humayun Kabir and Md.Anindya Tahsin Prodhan,"An improved tcp congestion control algorithm for wireless networks", IEEE, 2009.

[10]Weirong LIU Min Wu Jun Peng Guojun WANG,"A stable congestion control mechanism in adhoc network", IEEE, 2009.