Transport Layer's Approaches for TCP Incast Problem at Data Center Networks

Irfan Riaz Shohab, Muhammad Younas, Ramzan Talib, Umer Sarwar

Abstract ---Data Centers have become very popular for storage a huge amount /volumes of data. Many companies like Amazon, Google, Microsoft, IBM, Yahoo, and Face Book use the data center for storage, Web Search, E-Commerce, and large scale computation. High Speed Links, Low Propagation Delay, Limited sizes Switch Buffer are the main characteristic of Data Center. Data Center in these days have hundreds of thousands of server, storage across many thousands of machines.TCP is the most popular transport layer protocol that is currently used in the internet. Data Center faces the different set of problem than internet. TCP Incast is the main problem that the data centers faces.TCP incast problem is refer to the TCP throughput collapse. When the multiple data senders simultaneously respond to a single receiver, that is called many to one communication pattern, the burst data overload the buffer of receiver's switch. This causes the throughput collapse that degrades the performance and packets loss. That is so-called TCP incast problem. Many techniques, approaches and algorithms have been introduced for resolving the through put collapse problem of TCP. Due to the switching to cloud computing culture or cluster base switching system those are data centers. The researchers join the race of research to avoid the throughput collapse issue. Multilayer approaches & techniques are proposed for incast problem. Transport layer is the most important layer in the TCP/IP structure. This layer has the capabilities to handle or control to congestion. In this research work techniques for incast problem at Transport Layer level are discussed. In this research work attempt is made to discuss available possible solutions for researchers who want to work on the incast problem at Transport layer.

Index Terms— "Congestion"; "Cloud computer"; "Cluster base system"; "Data Center"; "Goodput"; "throughput collapse"

1 INTRODUCTION

As we have discussed is our previous research work. Irfan et al.,[1] and Cerf & Icahn [2] described that TCP is the one core protocols for the internet protocol .TCP are the currently most popular transport protocol that is used in internet industry. It is the backbone of the today computing industry. Transmission Control Protocol (TCP) is design to operate over the wide variety of the network Topologies and provide the Connection-Oriented services with guaranteed delivery and sequentially. Institute of Electrical and Electronic Engineers (IEEE) published a paper in May 1974 with a title of A Protocol for Packet Network Intercommunication. The authors are Vint Cerf and Bob Kahn. They describe an internetworking protocol for sharing resources using packet-switching technique between the nodes. Model's central control component was the Transmission Control Program that incorporates both connection-oriented links and datagram services between the hosts. Later on Transmission Control Program was divided into a modular architecture consisting of Transmission Control Protocol at connection oriented layer and the internet protocol at internetworking Layer .The model became known informally as TCP/IP.

- *Irfan Riaz Shohab is a student of MS in Computer Science College of Computer Science and Information Studies, Government College University, Faisalabad, Pakistan & also working as a Communication Engineer (Satellite) at Dunya NEWS TV Faisalabad Bureau. E-mail: shohabsons@gmail.com 03338369095*
- *Muhammad Younas is working as a Lecturer in College of Computer Science and Information Studies, Government College University, Faisalabad, Pakistan. E-mail: younas.76@gmail.com*
- *Ramzan Talib is working as a Principal (Associate Professor) in College of Computer Science and Information Studies, Government College University, Faisalabad, Pakistan. E-mail: ramzan.talib@gcuf.edu.pk*
- *M.Umer Sarwar is working as a Assistant Professor in College of Computer Science and Information Studies, Government College University, Faisalabad, Pakistan. E-mail: sarwaroner@gmail.com*

Gosai et al., [3] described that TCP use the flow control on end to end so that sender sends the data to fast for the TCP receiver to receive and process the data reliably. TCP use the Sliding Window for flow control protocol .The receiver specifies in the receive window field about the amount of receiving data that is willing to buffer the data. The main aspect of the TCP is congestion control.TCP uses the number of mechanisms to achieve high performance and avoid congestion collapse. The Modern implementation of TCP uses Slow Start, Congestion Avoidance, Fast-Retransmits and Fast Recovery Algorithms. Sender employs retransmissions that is based on the estimated time out (RTT) between the sender and receiver and also see the variance in the round trip time. Congestion Control or throughput has always been the focus of researchers since communication started. Congestion can be define as a network state, result of over load network resources like switches/ routers, resulting to unexpected behavior of network with the users which causes the lost or delay of packets. The solution of this problem of the congestion is congestion control mechanism. Congestion Control is the mechanism that gives us the solution to share the network resources among the existing competing traffics.

TCP divide the congestion control protocol into two main categories.

A; Single Rate Congestion Protocol that include the Rate Base Approach

1. Rate adaption Protocol
2. Low Delay Base Adaption Protocol
3. TCP Friendly Rate Control Protocol
4. TCP Emulation at Receivers.

B; Single Rate Congestion Protocol that includes the Window Base.

5. Random Listening Algorithms
6. Linear Listening Algorithms

7. Multicast TCP
8. Nominee Base Congestion Avoidance
9. Pragmatic General Multicast Congestion Control.

C; Multicast Rate Congestion Control Protocol include the Rate Base Approach

1. Receiver Driven layered congestion Control
2. Fair Layer Increase/Decrease with Dynamic Layering.
3. Layer Transmission Scheme
4. TCP Friendly Transport Protocol
5. Multicast Loss Delay Base Adaption Algorithms

D; Multicast Rate Congestion Control Protocol include the Window Base Approach

1. Rainbow

E; Categorization of Congestion Control Schemes is given below.

1. Window Base or Rate Base
2. Unicast or Multi Cast
3. Single Rate or Multi Cast
4. Peer to Peer or Network Layer Supports

F; TCP use the following main Congestion Control Algorithms

1. Slow Start
2. Congestion Avoidance
3. Fast Retransmission
4. Fast Recovery: TCP Reno
5. TCP New Reno
6. TCP Sack
7. FACK
8. TCP Vegas

Algorithms are shown in figure 1a.

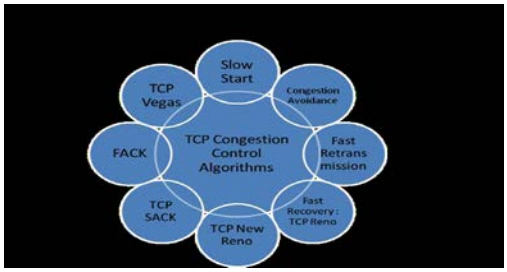


Fig: 1a Congestion Control Algorithms [3]

In [4] Wu et al., describe that TCP is one of the most popular Transport Protocol that is currently used in the network. It is also used in the data center networks. However, due to the incast problem and other issue it violates those assumptions on which the TCP was originally designed. TCP incast issue potentially arises in many application of typical data centers. For example, in Cluster Storage, when storage node respond to request for data, in web search, when many worker respond near simultaneously to search queries.TCP incast attract the attention of many researcher, interest as the development of data center and cloud computing.

In [5] Gibson et al., and In [33] Aized Amin Soofi et al, state that Data Center use the cluster base storage system and in cloud computing data is store at data centers. Data Stripping Technique is used in the cluster base storage system. In the data stripping technique the data is stripped over the multiple network storage nodes as shown in figure 1b. Concept behind this technique is segmenting logically sequential data, simply saying consecutive segment are store on different physical storage device. It is useful that a processing device access to data more quickly than a single device can provide or multiple segment can be access concurrently.

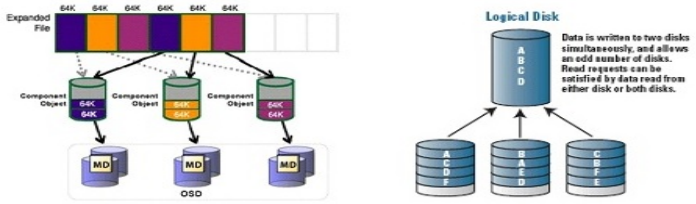


Fig 1b [6]

In [7] Zhang et al., define This problem arises because the client simultaneously reads fragments of data block from multiple sources that together send enough data to over load the switch buffer on the client link . These Storage systems consist of smaller set of storage servers that stored the data in spread form across these servers to increase performance and reliability. First time Incast word in a communication is used by Nagle.

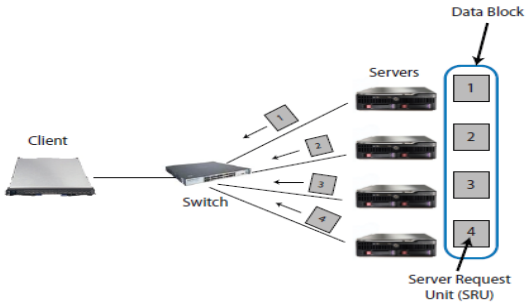


Fig 2a Zhang et al.[7]

In [8] Phanishayee et al., describe that Data block are stripe over a number of servers, such that each server store a fragment of data block, named as a server request unit (SRU) as shown in figure 2a. A client requesting the data block send the request packets to all of storage servers containing data for that particular block. The client request the next block only after it has received all the data for current block, it refer to the Synchronized Read. In [9] Zhang et al., define that Multiple path, small propagation delay, mixture of long and short flow, low latency. Small switch buffer size is the unique feature of Data Center.

In [10] Chen et al., 2009 state that In High bandwidth, Low latency data center networks, when the multiple data senders simultaneously respond to a single receiver, that is called many to one communication pattern, the burst data overload the buffer of receiver's switch. This causes the throughput collapse that degrades the performance and packets loss. That is so-called TCP incast problem and this concept is shown in figure 2b.

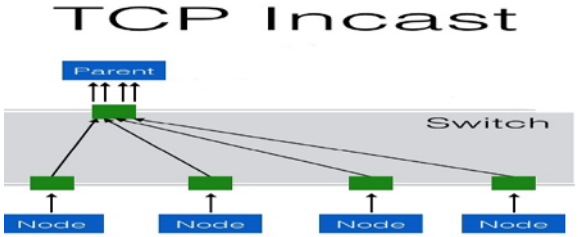


Fig 2B: TCP Incast [11]

It has become a new hot research topic in these days due to the industry and many organizations keeping a deep interest in switching to the cloud culture. TCP in data center has been a major concern recently because it leads to impairments such as TCP incast. With the cloud computing culture it become a important issue for the researcher because conventional TCP do not work properly. Data center face a different set of challenges then the internet. Many techniques, approaches and algorithms have been introduced for resolving the through put collapse problem of TCP. Due to the switching to cloud computing culture or cluster base switching system that is data centers. The researchers join the race of research to avoid the throughput collapse issue. There is still need of improvement in this field. Many possible solutions have been proposed from the aspects of multiple layers in which application layer, transport layer as well as link layer are also include.

There are many techniques for controlling TCP throughput collapse ranging from replacing TCP to alerting switches and routers. A major problem arises when conflicting goals such as congestion control and maximizing link utilization are address. In such scenarios it is real hard to reach a tradeoff between these two. So it is real important that whatever strategy is used these two goals should not affect each other drastically. Aprot from maximization of utilization and congestion control it is also important that the link also converges to it fair share.

2. Methodology

The collection principle through which we evaluated study sources is based on the research skills and experience of the authors and in order to choose these sources we have considered certain limitation: studies included in the selected sources must be interrelated to our problem of incast problem at data center networks and these sources must be web-available.

The review protocol is developed by using keywords "Congestion"; "Cloud computer"; "Cluster base system"; "Data Center"; "Good put"; "throughput collapse" and the following list of sources have been well thought-out to conduct the efficient review: IEEE, Elsevier Ltd, Springer, IT Professional Magazine, and Academia.edu, International journal computer science and mobile computing and different international journal of computer science.

Another step in the search process is executed by searching the correlated work area of the chosen papers to develop the review strength by validating that no cooperative reference is fails to notice during the explore process. Once the sources had been defined, it was compulsory to describe the process and the criteria for study selection and evaluation.

The inclusion standard for this study is harshly limited to studies that contain data center and incast problem concerning to data center networks, high speed network, cloud computing networks, cluster base storage system and is appropriate for further development of this incast issues.

3. TRANSPORT LAYER'S SOLUCTIONS FOR TCP INCAST PROBLEM AT DATA CENTER NETWORKS

Transport layer guarantees that information is conveyed without error, should be in sequence, and with no losses or duplication. Transport layer protocol provides the services of message segmentation, message acknowledgement, message traffic control and session multiplexing. [12]

Transport layer delivered the end to end or host to host communication services for application within a layered architecture of network component and protocol. Transport layer gives the services such as flow control, multiplexing, reliability, and connection oriented data stream. Transmission control protocol (TCP) is well known transport protocol. Addressing mechanism, packetizing issue like encapsulation and de capsulation is also discussed at transport layer. Congestion error and flow control is also discussed at transport layer. It also supports the connectionless and connection-oriented services. [13]

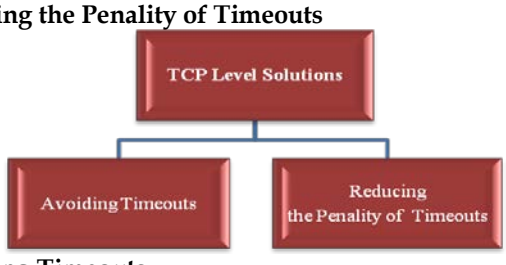
Many researchers contribute their effort to minimize the incast problem at Data center. Some of the solutions are highlighted in this section. Researchers have already proposed many solutions to avoid the incast problem, of course their solution solves the incast problem from different aspects such as tuning parameter or design new design approaches. Different solution is proposed at data link layer, Transport and network layers. We are going to explore the Transport layer approaches that do not require any modification at any element of network. [14] Data is stored at data center networks and read or accessed by the user on demand over the network, and that network could be wide area networks. Many company like Google, Hotmail, Amazone.com, IBM, Microsoft are providing the service to access, store, and process the data. To increase the performance and reliability of the system data is distributed over multiple servers. Stripping of data of at multiple nodes causes a fruitful factor of decrease latency of data center.

In[8] Phanishayee et al., described the TCP level solutions that are described given below.

3.1) TCP Level Solutions;

3.1.1) Avoiding Timeouts

3.1.2) Reducing the Penality of Timeouts



3.1.1) Avoiding Timeouts.

In [8] Phanishayee et al, describe that TCP Timeouts are the acentry cause of incast problem that effect the throughput at data center networks. The basic concept behinde the design of the TCP level soluction is to reduce the panalty and number of

timeouts. Phanishayee devided different type of approaches to avoiding the timeouts into following three categories.

3.1.1.1) Alternative TCP implementations –Reno, NewReno, Sack.

3.1.1.2) Addressing the Lack of Sufficent feed back-Limit Transmit & Reduced Duplicate ACK Threshold.

3.1.1.3) Disabling TCP Slow Start.

We show the approaches to avoid the Timeouts in following figure 3.

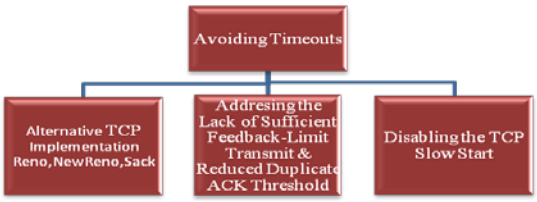


Fig: 3 Soluction for avoiding the timeouts

In [8] Phanishayee et al., divide the type of timeouts into three categories that cannot be avoided by the most flavor of TCP.

1) Full Window Lost
2) Last Packet Lost
3) Lost Retransmission

These types are shown in following figure 4.

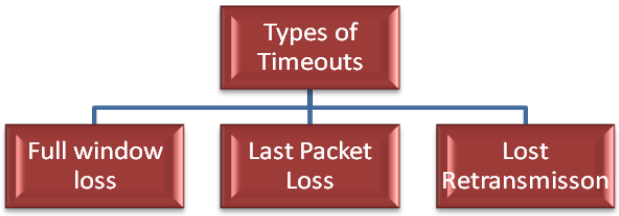


Fig:4 Different type of Time outs

1) Full Window Lost;

In Full window lost case as shown from its name the complete window of the dara is lost and no feed back is available that is used in recovery for TCP.

2) Last Packet Lost;

In Last Packet Lost case the last packets is lost in the SRU and data in the block is not available for data driven recovery.

3) Lost Retransmission;

These mechanisms will occure when the retransmitted packets is also lost or dropped. Since there is no method is available to informe the sender wether this retransmission packet is lost.

3.1.1.1) Alternative TCP implementations –Reno, NewReno, Sack.

Phanishayee state that we can avoid the timeout with alternative use of TCP implementation. Many flavor of TCP are useful to reduce the timeouts that give us the fruitful result to minimize the effect of incast at data center. There are many development phases of TCP like Reno, New Reno, Sack have the capabilities to avoid the timeouts. Each of them has a modified or advance technique that is useful to avoid the timeouts.

In [15] Atul et al., Describe in his research work that Slow start use sender base flow control. It controls the rate of sender on the basis of Acknowledgement that is received from the receiver. It maintains the congestion window and its value are used to control the rate of the sender. The value of the congestion window is incremental after receiving acknowledge. Value of congestion window is one at starting level that is too much slow to obtain the pet rate of data transmission. TCP Reno is new comer that overcomes this issue/problem. TCP Reno uses Fast Recovery mechanism that uses the large window size and giving us the fruitful effect on the higher data rate. In [16] Floyd & Henderson describe in his work that TCP Reno has not strong ability to avoid the timeouts. Main weak point of TCP Reno is that it does not properly recover multiple losses in a window. Phanishayee define in his work that when size of window is 6 if the first two packets of window are lost then TCP Reno will experience a timeouts. The performance of Reno is very beneficial over the TCP when the losses of packets are very small. But in the case of lose of multiple packets in the single window its performance will suffer and its behavior is like the previous version of TCP Tahoe. So we can say it is fruitful for us in the case of single packet loss detection. Main advantages of TCP Reno are its characteristics of avoidance of congestion and utilization of bandwidth. So it also gives us some fruitful result to mitigate the effect of incast problem than the older version of TCP.

In [16] Floyd & Henderson also describe the new modified version of TCP Reno that is TCP New Reno. It has some modification in Reno that is useful for Incast problem and it also overcome the drawback of TCP Reno. It uses the advance retransmission algorithms in TCP Reno that is useful to overcome the problem of TCP Reno. So TCP New Reno is the modified version of the TCP Reno that has most fruitful effect to minimize the effect of Incast problem.

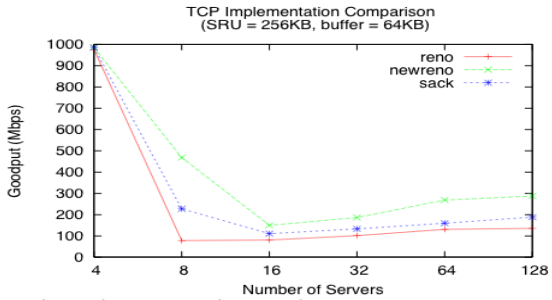
TCP NewReno has the ability to detect the multiple losses of packets in the single window, so it is more efficient in detection of multiple losses of packets than TCP Reno. It has also ability to enter in the fast retransmission mode as the Reno enter in this mode but it has a most important benefits that it does way out this features in the case of partial ACK. In this case it transmits packet on immediate basis that is shown in the partial ACK. TCP NewReno also give us positive result that is useful for congestion control as well as incast problem. But TCP NewReno has a week point that it take one RTT to detect the lost of packet.

TCP SACK is the modified version of the TCP NewReno. Its uses the Selective Acknowledgement approach to overcome the weak point of the TCP Reno and TCP NewReno. TCP SACK has the ability to detecting multiple packets loss and retransmission of multiple packets lost per RTT. It has the slow start and fast re-transmits ability of Reno and also has coarse grained timeout part of Tahoe. So it contains the both benefits of TCP Reno and Tahoe. In [15] Atul et al., describes in his research work that TCP SACK protocol uses the Selective Acknowledgement algorithms. In this algorithm maintains a queue that has a list of received and missing segments. The beauty of TCP SACK is that sender retransmits the miss-

ing segment without waiting the retransmission timeouts. The next block is only sent by the sender when all the segments are acknowledged at the sender side.

In [17] Kevin and Sally Describe that TCP SACK has the ability to sent the specific lost packet in a window. TCP SACK uses the selective Acknowledgement approach to point out the lost packet in the window that required to be resent.

So we can say that with the alternative implement of TCP protocol it has a fruitful effect on the Congestion control as well as the positive effect to mitigate the effect of incast problem. The effect of good put of different flavor of TCP like Reno, New Reno, and SACK is shown in figure no 5 given below. Figure shows that TCP Reno and TCP SACK have as strong effect on good put than the Reno. So it is also helpful to mitigate the effect of incast problem.



Fig=5 Alternative TCP Implementations –Reno, New Reno, SACK

Phanishayee also did the simulation for the different flavor of TCP about the distribution of duplicate acknowledgement Received at a Timeouts (DART); his simulation result is shown in figure no 6 given below. Figure 6 a show the DART distribution for the TCP Reno and figure b show for New Reno. This DART is reorder for a 20s run with 16 servers, 64 packet switch buffer and size of SRU size is 256 Kbytes. He also categorizes of timeouts events in the table shown in figure 7. Simulation result in the table of figure of 7 shows that in the case of New Reno the number of timeouts is lower per data block.

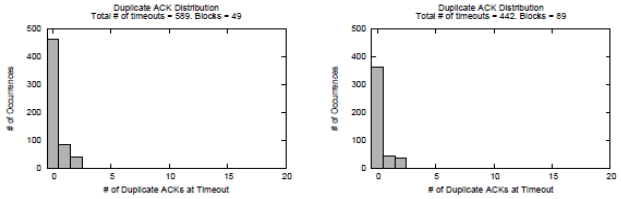


Fig: 6 Distributions of Duplicate Acknowledgements Received at a Timeouts

	Reno	NewReno
data blocks transmitted	49	89
timeout events	589	442
full window losses	464	362
lost retransmits	61	2
lost retransmits when DART $\geq da_{thresh}$	0	0
lost retransmits when DART $< da_{thresh}$	61	2
last packets dropped	2	5

Fig: 7 categorizing timeouts event under different TCP Flavor

3.1.1.2) Addressing the Lack of Sufficient Feed-Back-Limited Transmit and Reduced Duplicate ACK Threshold;

In [8] Phanishayee described in his research that When a many number of packets are lost in a large window size or when a window size is small of a flow and large number of packets are lost, Limit Transmit make an effort to ensure that for triggering the 3 duplicate ACK the enough packets are sent that is necessary to enter fast recovery and fast retransmission. On the other hand we can decrease the duplicate acknowledgement threshold from 3 to 1 to robotically / automatically activate the fast recovery and fast retransmission upon receiving any duplicate acknowlegement.

Figure 8 given below show that this approach did not provide a fruitful result for throughput over the NewReno Flavor of TCP.

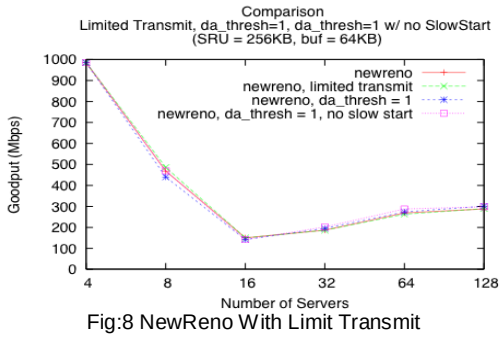


Fig:8 NewReno With Limit Transmit

As shown in simulation result in figure 8 that the Limit Transmit has not such strong effect on goodput.Both graph of NewReno with and without limit transmit are on the same level. It has not positive result on the goodput because when we are going to transmit the data block it still requires at least one timeout. In [18] M.Allman et al., dscribe in his research work of Limit transmit that in this approach the sender sends the new data segmant in the reaction of the first two duplicate acknowledgement that sender receive from the receiver. Due to transmit these segments TCP recovers the single lost segment with the help of fast retransmit algorithms. Thus the ratio of recovering the lost segments in a large window size will increase. It is also informative in the case of with or without TCP SACK.

3.1.1.3) Disabling TCP Slow Start;

In [8] Phanishayee et al., and in [19] Padhye et al., described that Slow start is another flavor of TCP that use the sender base flow control mechanism. In this slow start mechanism the rate of sennder is control on the base of acknowledgement that is received from the receiver side. The size of congestion is increase by on the base of receiver's acknowledgement like 1,2,4,8,16,32 …….. as shown in figure 9.

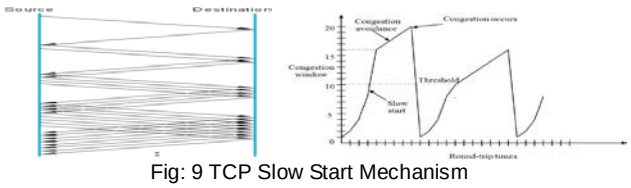


Fig: 9 TCP Slow Start Mechanism

Congestion window is effectively double at receiving every RTT.Due to this we have to hace the packet loss and packet drops. The acknowledgement of sender has to wait timeouts and thus reduce the size of congestion window. As shown in

figure 10 at start level of slow start the rate is incremented exponentially.This exponentially increment causes the congestion.

Phanishayee describe in his research article that disabling the TCP Slow stat is much useful to avoid the congestion and also helpful to avoid timeouts and finally this technique is useful to overcome incast problem. When disable the TCP Slow Start to avoid the network congestion created by the exponentially increasing the window size of flow to find out the capacity of link subsequent a timeout.

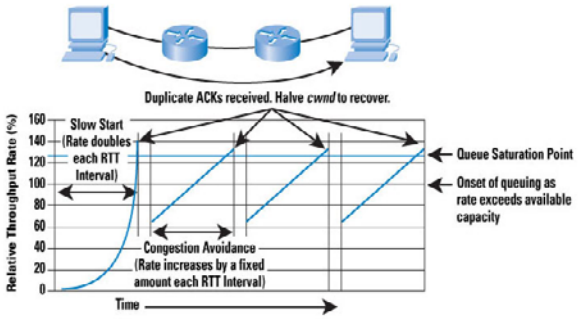


Fig: 10 Slow Starts and Congestion Avoidance [20]

3.1.2) Reducing the Penalty Timeouts;

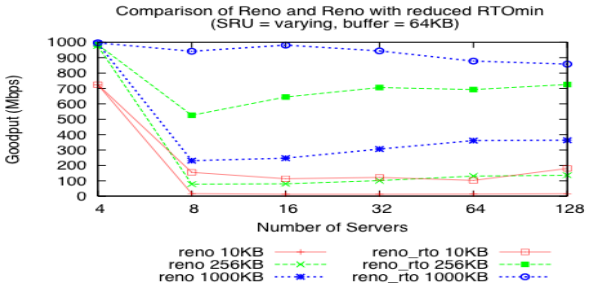
In [8] Phanishayee et al., described in his research paper that plummeting the consequence of timeout we can avoid the effect of incast problem at data center networks. Amar said that there are many timeouts like full window loss and lost retransmission, we cannot avoid these timeouts. The basic idea behinde the approach of Amar is that reduced the time that is spent for waiting timeouts. This approach gives us the useful result in the form of improved goodput. It also has some weakpoint in the form of pre-meture timeouts. In [21] Mark and Vern et al., defined that if we reduced the timeout then the risk of premature timeout will increase especially when our network is wide area network. If we are going to implement this technique we never forget the premuture timeouts.

Phanishayee describe that RTO (retransmission timeouts) is the amout of time a flow has to wait before retransmitting a lost packet without mechanism of fast retransmission by the three duplicate acknowledgement.Premature timeouts are due to the effect of estimating the value of RTO. According to the research work of Phanishayee there are two negative effects of premature timeouts.

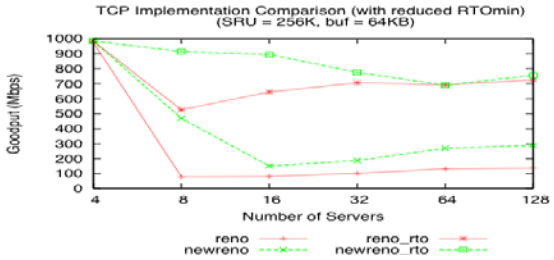
1) It causes to a factitious retransmission.
2) TCP reduces its value of slow start threshold by the half and due to this enter to the state of slow start but there is lost of packets. Due to no congestion , TCP thus would misjudge the ability of the link and throughput would undergo.

In [22] Pasi and Alexey describe that the value of RTOmin that is mostly used for the different falvor of TCP is 200ms. This value of RTOmin is much greater than Round Trip Time (RTT). This large value of RTOmin inflict a enormous throughput penalty, because time of transfer for each data block is drastically low than RTO min.

The simulation work of Phanishayee show that if there are 8 to 32 servers and reduced the value of RTOmin from 200ms to 200us, it give us the fruitful result on the goodput. The goodput improves by the order of magnitude as shown in figure 11.



(a) Varying SRU sizes



(b) Different TCP implementations

Fig: 11 simulation result of reducing the RTOmin

Figure 11a shows that when reduced the value of RTOmin the value of goodput improves, this simulation is done with the Reno falvor of TCP. In this simulation auther also vary the size of SRU and see the effect on the goodput. In figure 11b same simulation is done for the NewReno flavor of TCP. Simulation result shows that with reducing the RTOmin we have fruitful improvement in the goodputs.

Phanishayee also state that this approach also has a little bit implementation issue. To setting the value of RTOmin at low value we have to face noteworthy implementation, safety and generality issue.

Implementation Issue;

In [23] Vern and Mark describe that when reducing the value of RTOmin ,it also reqiure TCP clock granularity of 100us ansd this value is according to the algorithm of standard RTO estimating. Some oprating systems are not able to implement this value of clock granularity. Linux use the TCP clock granularity of 10ms.

In [24] Mohit and Peter described that If we are going to change the value TCP timer in microseconds that we have to do some changes or supports from the hardware level. Most oprating systems have not this ability.

Safety and Generality;

If we achieve the satisfactory desireable value of TCP timer then Reducing the value of timmer is also detrimental particularly in the situation of server client communication pattern in

the wide area network or when the servers communicate with clients.Also remember that if we reduced the value of RTO min at low level then we have to face the premature timeouts. The small value to RTOmin causes a unauthentic retransmission.

3.2) Ren's Model of Transport Layer's Approaches for incast Problem;

In [25] Renet al., defined the approaches to migitigate the effect of incast problem at datacenter networks in his research work. He devides approaches into three categories in his research work.

Transport Layer's Approaches for incast Problem;

3.2.1) Modifying the TCP parameters while maintenance the TCP protocol unaffected.

3.2.2) Designing the Enhanced TCP Protocol.

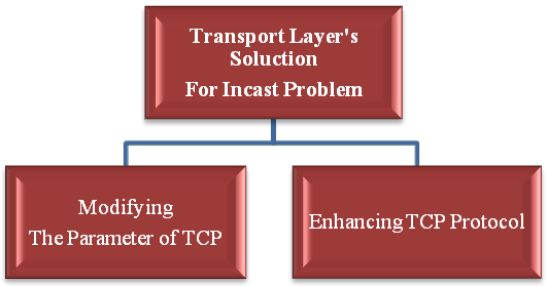


Fig:12 Transport Layer's Soluction For Incats Problem

3.2.1 Modifying TCP Parameters;

Devide this category into three phases.

3.2.1.1) Removing Binary Exponential Backoff

3.2.1.2) Disabling the Delayed ACK

3.2.1.3) Reducing the minimum RTO timmer

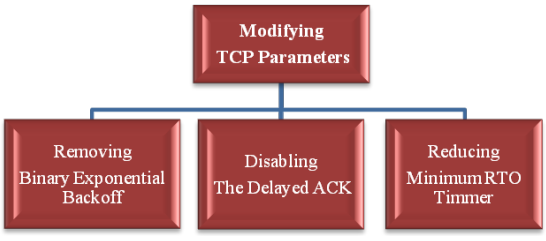


Fig:12 Different Approaches For Modifying the TCP Parameters

3.2.1.1) Removing Binary Exponential Backoff;

In [25] Ren et al., defined that removing the binary exponential backoff (BEB) is most advantageous on the goodputs. It is also useful to mitigate the effect of incast problem at datacenter networks. During the time of waiting for the stalled server to receiver the link is underutilized .Thus it causes the harsh droping of throughputs. Many charecteristics of data center networks like limited packets to send, application's synchronized read and packets being transferred by store and forward rather than broadcasting are different from the Ethernet.Thus, removing the binary exponential backoff is appropriate in the TCP of data center networks.

In [26] Hongyun et al., describe in his research work that in case of brutal congestion in excess of the 50% retransmission of TCP timeouts can raise BEB in the data center. In his research work Hongyun prof with simulation and investigation that removing the algorithm of binary exponential backoff from the TCP does not move forward the commencement of incast problem at data cener networks. Instead the removal of binary exponential backoff is useful to reduce the effect of incast problem. In some cases its benefits are perceptible with large SRU size and lower RTT min.

Hongyun state in hid work that Binary Exponential Backoff algorithm bring into play to control the interval between the successive retransmission in the case of numerous timeouts occure. A server has to face a timeouts is stalled even though other server can utilize the accessible bandwidth to complete transmistting their segment of the block. So client has to wait till the stalled server recovers from its timeouts and send its remaining component of block. Link of the client may be absolutely ideal during the waiting time that is due to stalled. Thus the comprehensive throughput over the client link to switch is significantly dropped. Binary Exponential Backoff algorithm of TCP is purely a classical Ethernet protocol. In the of Ethernet protocol enviroment the comunication medium is share between the physically spread comunication nodes. In this enviroment Nodes broadcast the packets of traffice on the channel to communictae each other. If multiple Nodes broadcast at the same time then the collision of packets occure. It is also possible that same nodes conflict again due to unsuccessful retransmission.In this situation Binary Exponential Backoff algorithm is most useful to control the interval between consective retransmission. The fuctionality of data center is different from the classical Ethernat. In the Data center network packets are sent by the mechanism of store and forward instead of broadcasting the packets. Hongyun did the simulation for both situations with or without the Binary Exponential Backoff algorithm. He validate that if we remove the Binary Exponential technique from the TCP in the case of data center network, it has a fruitful result on the goodput.

Simulation result of removing the Binary Exponential Backoff algorithm is shown in figure 13. As shown in given below figure when remove the Binary Exponential Backoff algorithm its goodput will improve as shown in figure at the TOP level graph.Removing a lgorithms ofthe Binary Exponential Backoff is also useful when we change the size of SRU and vary value of RTTmin. **2**

Removing BEB is useful in the TCP traffice brustiness. Figure 14 show the simulation result of TCP with and wothout BEB. The dotted line in the upper side show the time when the request for data block start at time t and lower star dotted line show the time a packet reaches at switch. Simulation result show that removing the BEB smothes the interval time between the two consecutive requests. Figure 15 show the result of difference of completion times of transfer of data block for all TCP connections. Result show that difference of time when different TCP complete send the data block is very small when BEB is removed.

Simulation result shown in figure 16 show that the when we

increase the size of SRU or reducing the value of RTTmin there is no noticeable effect of removing the Binary Exponential Backoff on the traffice of TCP burstiness. Result are similer with or without the removing the BEB.

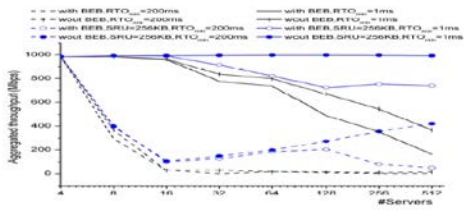


Fig:13 Comparison of throughput when buffer size of 64

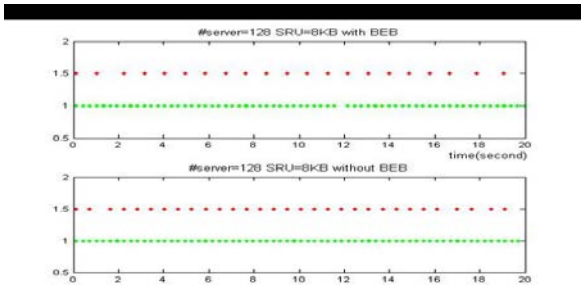


Fig:14 TCP traffice brustiness

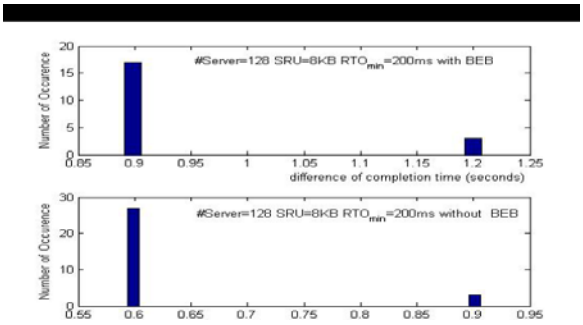


Fig:15 disribucion of different of ttransfer Complection time

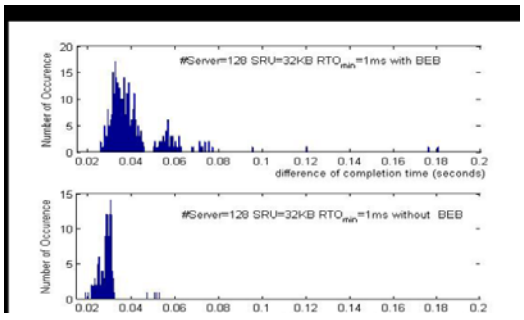


Fig:16 Distribuction with lower RTTmin & larger Sru size

edgement of TCP attempts to reduce the volume of the traffic of acknowledgement by having a receiver acknowledge just each other packets. Vijay define that if there is no packet after the received packet, the receiver has to wait up to the delayed acknowledgement timeout threshold earlier than transfer an acknowledgement. Previous work of different scholer show that in the case of cluster base storage systems with 3 to 5 servers and having barrier-synchronized request work loads, the mechanism of delay acknowledgement can proceed as a tiny timeouts. So it has a reduced effect on the goodput but not a calamitously low throughput in the convinced loss patterns. The acknowledgement procedure with and without delay is shown in figure 17. In the case of delay acknowledgement when the size of the window in small, it has a lower inpact of the throughtput as well as negative effect on the loss recoveryn in th form of slower loss recovery. But in the case of when disabling the Delay ACk, the acknowledgement is sent immediately after the receiving the packet. It enables the sender to grow up its window of TCP and activating the data driven recovery for packet 2. When we are using the delay ACK mechanism the receiver has to wait up to 40ms to send the acknowledgement. It causes the delay in recovery process. But this delay is not as high as in the normal or full 200ms RTO. The default delyed Acknowledgement minimum in the case of Linux is 40ms that is still large in the case of when we compared it with the RTTs in the data centers.It causes the low throughput in the case of 3 to 5 simultaneous respond senders. Microsecond retransmission timeouts have different transactions with mechanism of delay acknowledegement. The dely ACK timmer of receiver should always be let off before the firing of retransmission timmer of the sender to put a stop to the sender from timing out waiting for an ACK that is just delay. Modern systems have the ability to handle this situation with the help of setting the value of ACK timmer to 40ms that is asfe under the 200ms value of RTO min as shown in figure 18. In the case of Microsecond granularity retransmission would occasionally occurrence of an unnecessary timeout when communicationg with unmodified hosts in which the RTO is lower then 40ms. Vijay demonstrated in his work that abolishing the RTOmin has a slight impact on performance of volume data transfer in the wide area flows.

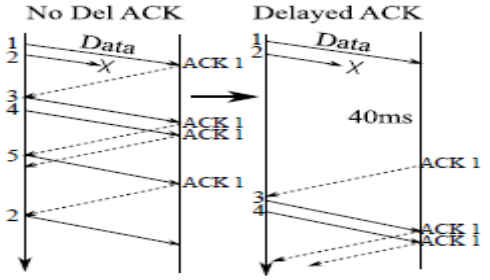


Fig:17 Acknowledgement procedure with and wothout Delay

2

3.2.1.2 Disabling the delayed ACK

In[27] Vijay et al., defined in his work that accroding to the R.T.Bradin In [28] the approach or method of delyed acknowl-

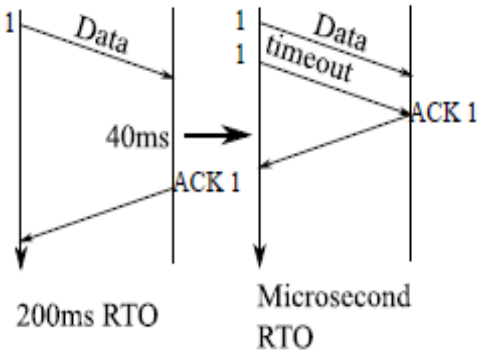


Fig: 18 ACK with microsecond RTO & 200ms RTO

Figure 19 show the effect of Delay ACK disable and also the effect of enabling on the goodput. As shown in the figure the upper line of the red color of Delay ACK disable has the improved effect on goodput.In this simulation the block size is 1MB and buffer is 32KB. Vijay used the 200us value of timmer for delayed ACKs and also used the default 40ms value of delayed ACK. Further than the 8 servers, Client that have a 200us delayed ACK timmer have to face a15-30Mbps lower throughput as compared to the throughput of disable of the delayed acknowledegment. Different value of RTT & RTOmin in different enviroment is shown in figure 20.

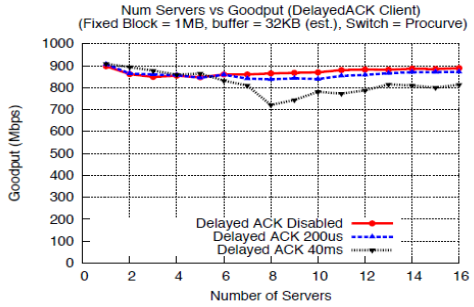


Fig :19 Effect of Delayed ACK Disabled on the Goodput

Scenario	RTT	OS	TCP RTO_{min}
WAN	100ms	Linux	200ms
Datacenter	<1ms	BSD	200ms
SAN	<0.1ms	Solaris	400ms

Fig:20 Values of RTTs & RTOmin in Different Enviroments

3.2.1.3 Reducing The minimum RTO Timer;

In [29] Y.Chen et al., and in [25] Ren et al., described in his research work that In the environment of WAN the default design value of the TCP minimum RTO timer is 200ms. If we reduced the default value of RTO timer from 200ms to 200us, its has a improved/fruitful impact on the throughput. The simulation result of Yen shown in firgure 20. As shown in figure the curve of graph grow up when the value of timer is near about 200us, Its is at the starting level of the figure.But Yen also described that reducing the value of RTO timer has some implementaion issues. Its requires the TCP clock granularity of 100us.According to the algorithms of estimating the value of standard RTO many oprating systems like the Linux and BSD TCP currently have not ability to provide the this fine grained timer. So it is too difficult to implement this approach. There is another issue of implementation that to reducing the value of timer is also destructive in the case of when the servers communicate with the client in the wide area network (WAN) enviroment. The summary of this approach is that trim down the value of timer mean high value of goodput as shown in given below figure 20.

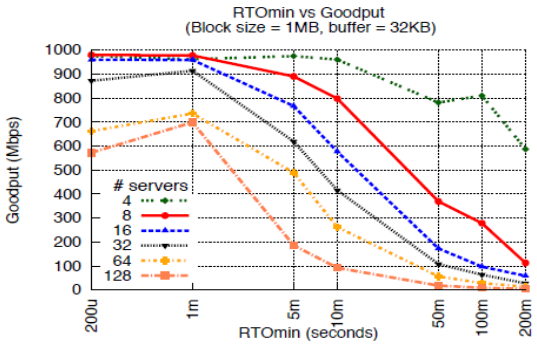


Fig:20 Effect of RTOmin on the Goodput(Mbps)

3.2.2) Designing the Enhanced TCP Protocols;

Environment and requireemnt are the main things that force the researchers to develop the different flavor of the Transmission Control Protocol. There are many flavors of TCP that are specially design for the Data Center Networks. In thhis section we will discuss those special protocols that are most effetive to mitigate the effect of the Incast problem at data center networks.

3.2.2.1) Data Center TCP (DCTCP);

In [30] Muhammad Alizadeh et als.,develpoed a new protocol named as a DCTCP and defined in his research work about DCTCP that The main objective of data center transmission control protocol (DCTCP) is to attain sky-scraping burst tolerance, high throughput and low latency with low buffered switches in a ordinary data center networks. DCTCP is design to maneuver with small queue without any throughtput loss. Simple marking scheme is used at switches in the data center networks. Simple marking scheme activate the packets's Congestion ExperienceD (CE) codepoint immediately when the capacity of buffer exceeds a fixed minute threashold. The source of the DCTCP acts in the response by reducing the size of window by the factor that depends on the portion of the marked packets. If the value of fraction is large then its means that decrease factor is also bigger. In other words DCTCP uses the Explicit Congestion Notification (ECN) to providing the multi-bit feedback to the end hosts in the network.Remember that key contribuction is not control law. It is the work of drawing the information of multibit feed back from the single bit sequence of marks. DCTCP wants the network to make available only single bit feedback, Due to this we are able to reuse already available machinery of the ECN in the modern switches and TCP stacks. The information of path delay can be

view as a implicit multi bit feedback. In the case of high data rates with low latency network selecting the queue buildup in the buffered switches can be tremendously noisy.

Algorithm of DCTCP;

There are three main component of DCTCP, detail are given below.

1) Simple Marking at the Switch:

DCTCP make use of simple scheme of active queue management.There is only one factor, K marking threshold. The value of the K that is threshold with the queueing occupancy. In genral when the packet is arrieved at the switch, the value of K marking threshold is compaired with queue occupancy.If the queue occupancy is greater than K then packet is marked with the CE code point otherwise not marked. This scheme guarantee that the senders are rapidly give notice of queue overshoot. Modern switches implement the RED marking scheme. It can be re-purposed for DCTCP. Simply we required setting both the low and high thresholds to K and marking is based on the instantanious instead of average queue length.

2) ECN-Echo at the Receiver Side;

The way in which the information in the CE codepints is conveyed back to the sender is only difference between DCTCP receiver and the TCP receiver. The receiver activates the ECN-Echo flag in the series of acknowledgement packets until it receives the verification from the sender that the congestion notification has been received in the case of RFC 3168. The receiver in the DCTCP struggles to send the precise information back to the sender about the accurate sequence of the marked packets. This can be done with the simplest way to ACK every packet, by setting the flag of ECN-Echo only in the one situation when the packets have a marked CE-codepoint.

Due to the multiple reasons the use of delay ACK is very imperative. Reducing the load at data center network is one the reason. To used the delay ACK ,the receiver of the DCTCP uses the inconsequential two state machine that is shown in the figure 22 to decide the whether to set the ECH-Echo bit. The state communicates to whether the last received packet was marked with CE codepoint or not. So the sender has the information about how many each acknowledgement envelops.

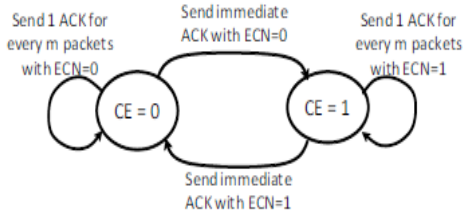


Figure: 21 Two State Machine

3) Controller at the Sender Side:

As shown from its name it performs the controlling function at the sender side. Sender keeps up an estimate of the fraction of packets that are marked. It is called as α. The value of α is updated once for every window of data, equcation is given below.

$$\alpha \leftarrow (1-g) \times \alpha + g \times F,$$

F = fraction of the packets that was marked in the last window of data.

g = is the weight to the new samples against the past in the estimation of α. Its value is between the one and zero.

When the queue length is higher than the K, the sender received every packet with marks. The value of α is approximates the likelihood that the queue size is greater than K. There are two possibilites of α.

If

α near to zero show that the level of congestion is low.

α naer to one show that the level of congestion is high.

How to react after receiving the ACK with ECN-Echo falg set is the only one difference between DCTCP sender and TCP sender. All the old attribute of TCP like the additive increase in congestion avoidness or recovery from packets losss and slow start are left unchanged. TCP always cuts its window size by the factor of 2 in the reponse to a marked ACK. But in the case of DCTCP, it uses the value of α for resizing the window size.Equcation is shown in given below.

$$cwnd \leftarrow cwnd \times (1 - \alpha/2).$$

If the value of α is near to the zero (0) ,its mean low level of congestion and resize the window size slightly.When the Value of α is near to one (1) ,its mean the level of congestion is high and resize the window size by cutting down its Value to half. When the size of queue is exceeding K, DCTCP start the resizing process. Figure 22 show the effectiveness of DCTCP in achieving full goodput taking the small spacing in the buffer of switch as compared to TCP. As shown in figure 22 TCP required the high capacity in the switch buffer.

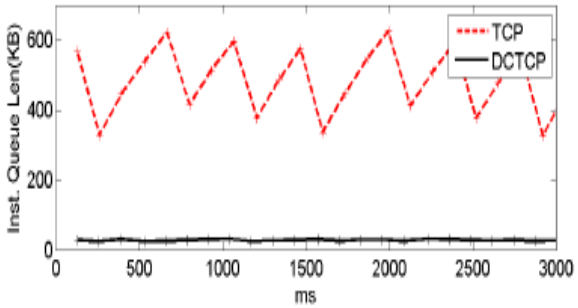


Figure :22 TCP and DCTCP comparison for switch buffer

DCTCP improve the impairments like queeu build up, buffer pressure and incast problem.

1) Queue Build Up;

DCTCP sender reacts immediately when the size of queue going to exceed the value of K.

This is helpful in queue dely in the case of congested switch ports.

2) Buffer Pressure;

DCTCP also give us the fruitfull result to resolve the issue of pressure on the buffer, because the value of queue length of the congested port does not rise extraordinarily large.

3) Incast;

DCTCP also give us the most fruitfull result to mitigate the effect of the incast at data center network.Because there is no pressure on the buffer of the switch then no buffer overloading, so no throughput collapse.

3.2.2.2 Incast Congestion Control for TCP (ICTCP);

In [31] Wu Haitao et al., developed a new flavor of TCP named as a Incast Congestion Control for TCP (ICTCP). His main idea is to design an incast congestion control for TCP approach on the side of receiver. Haitao described in his research work that this method regulates the receive window of TCP proactively before occurance of the packets loss. Due to the implementation of this approach, we accomplish almost the zero timeouts and high throughput.This research work focus on the avoiding the packet loss before the incast congestion that is most benificial for us than the recovery after the lost, because the recovery after loss has a huge cost. This soluction modifies the receiver of the TCP that's why it is a better soluction than those require the modification on both sides of TCP and require some changes on switches as well as the stack TCP protocol. Haitao selected the receiver side because it has the information about the throughput of all the connections of TCP and available bandwidth. The receiver side of the TCP adjusts the size of the receiving window of the TCP connection. In other words the resizing of window is performed on the receiver side.

This new protocol has the compatibilities of the entire old version or flavor of the TCP and not requires anu modification at switch and TCP Stack. It also has the abilities to handle the future high bandwidth and low latency networks. The basic thinking of this apaproach is to avoid the unnecessary overflow of the buffer, so we reduced the timeout and saves preventable retransmission. Haitao describe in his research work that the receive window of TCP can be used to suffocate the throughput of TCP. Result of simulatio is shown in figure no 23. This approach is useful to control the incast problem or congestion. Basically the design receive window is for flow control. This scheme is a window based congestion control algorithms. The most important benefit of this algorithm of incast congestion control at receiver side is that receiver has the information about how much goodputs it has acheived and also has the information about the bandwidth is left. He summarized the observation of the ICTCP into the 3 parts.

First, the value of existing bandwidth at receiver is the indicator to the receiver to perform the congestion control. If the receiver of the TCP require to increase the receive window of TCP, it should also forecast whether there is sufficient obtainable bandwidth to maintain the enhancement.

Second, The dynamics of congestion control of one TCO connection can be considerd as a control system. Feedback of delay is the RTT of that TCP connection. When the receive window of TCP is accustomed, It gets the one RTT time at least prior to the data packets subsequent the newly adjusted receive window arrive. The period of control should be bigger then one RTT time.

Third, This incast congestion control scheme should regulate the size of window according to condition of link congestion and alos application requirement. When the bandwidth is available then the receive window size should not limit the TCP goodput and should smother TCP goodput earlier than the happening of the incast congestion.

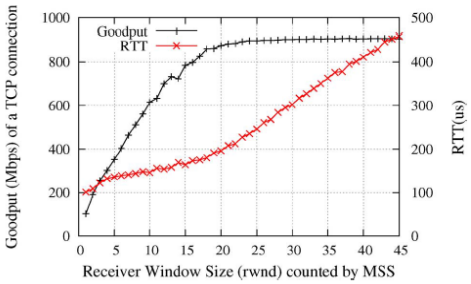


Figure 23

Algorithms of ICTCP,

Algorithms of Incast congestion for TCP has following main component

1) Control Trigger; Available Bandwidth;

ICTCP algorithm is also fruitful in the case when the receiver has the multiple interfaces and this approach is independently peformed by the connections of each interface. The link capacity of the interface on receiver is C, the bandwidth of the total incomming traffice on the interface is BWt, and these include all type of traffice like broadcast, multicast, unicast of TCP and UDP. The available bandwidth on the interface BWt as,

$$\mathrm{BW_A} = \max(0, \alpha * C - \mathrm{BW_T})$$

Where α □ [0,1] is a parameter to absorb potential oversubscribed bandwidth during window adjustment. We have the fixed setting value with α = 0.9. The available bandwidth BWt is used for increase the size of the window for higher throughput. For estimating the bandwidth that is available on the interface, he devide the time into slots. Each slot contains two subslots with the same length of T. This time slots mechnism is described in this figure 24. First timeslots is used to measure all received traffic for each network interface. Second time slots is used to calculate the available bandwidth for window increase. In the first time slots the value of the received window cannot be increased but it can decrease if required. In the second Time slots the window size can be increased.But rememer this increase in the receive window should not increase the available bandwidth that is calculated in the first time slot.

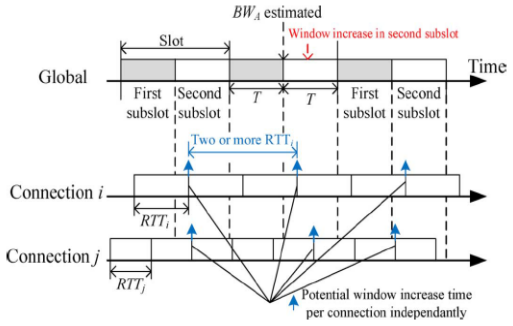


Figure 24

2) Per Connection Control Interval: 2*RTT;

No extra TCP ACK packets are produced for the resizing the receive window. Each connection regulates the receiving window when there is ACK transfer out on that connection. So no traffice is exhausted. When an ACK is sent out, the data packet related to that ACK reaches on RTT later. The latency on the feedback is one RTT time of each TCP connection. To estimate the goodput of TCP connection resizing the receive window, the minimum time sacle is an RTT for that connection. So in the ICTCP control interval for a TCP connection is 2*RTT. Required one RTT latency for that used to window to take outcome. One supplementary RTT is required to measure the achieved goodput with that newly adjusted window.

3) Window adjustment on Single Connection;

The expected throughput/goodput of connection is obtained as

$$b_{i,\text{new}}^{\text{m}} = \max(b_i^{\text{s}}, \beta * b_{i,\text{old}}^{\text{m}} + (1-\beta) * b_i^{\text{s}}).$$

$$b_i^{\text{e}} = \max(b_i^{\text{m}}, \text{rwnd}_i/\text{RTT}_i)$$

Where bm is the measured throughput

bi is the expected throughput

db is the ratio of throughput difference of measured and expected throughput over the expected one for connection i,

$$d_i^b = (b_i^{\text{e}} - b_i^{\text{m}})/b_i^{\text{e}}.$$

The window adjustment is given below.

$d^b <= \gamma 1$	Increase receive window if there is enough quota of available bandwidth on the network interface. Decrease the quota correspondingly if the receive window is increased
$d^b > \gamma 1$	Decrease receive window by one MSS2 if this condition holds for three continuous RTT. The minimal receive window is 2*MSS.
$\gamma 1 < d^b < \gamma 2$	Keep current receive window

4) Fairness Controller for multiple Connectios;

When the receiver of TCP senses that BWa is smaller than the threshold, ICTCP start to reduce the window of receiver to avoid the the congestion of some selected connection. In the case of multiple TCP active connection at the same time, then we need such method that can attain fair sharing of all the connection without affecting the goodput. ICTCP does not regulate receive window for the flows with RTT larger then 2ms. So the fairness is only measured among low latency flows.

The software stack of ICTCP is shown in figure 25.

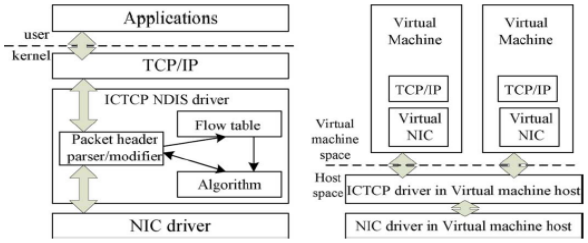


Figure 25 ICTCP implementation

The effect of ICTCP on the throughput is shown in figure 26 with different point of view.

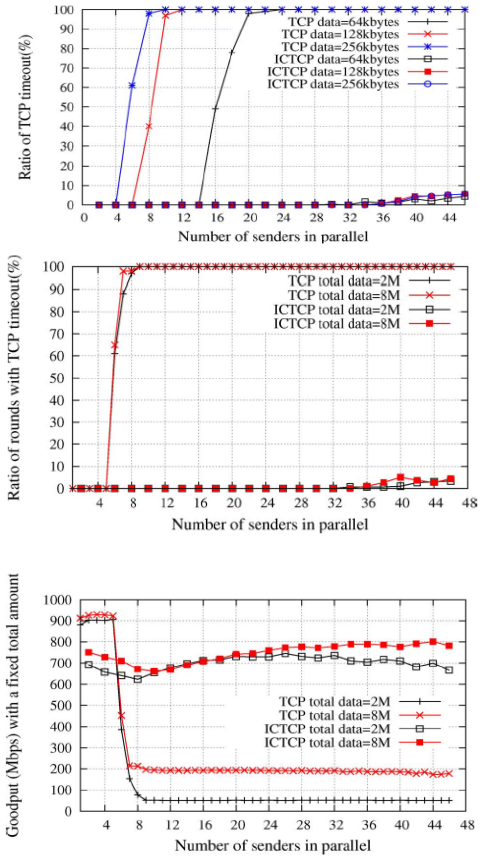


Figure 26: Different effects of ICTCP on goodput and Timeouts

3.2.2.3) Multipath TCP;

In [32] Raiciu et al., described in his reseach work that Multipath TCP expands TCP so that a single connection can be exposed across multiple network paths. In the initial SYN ex-

change the MPTCP support is consulted and client discovered any additional IP addresses the server may have. Additional subflows can then be among the same pair of IP addresses as the first subflow, but remember by using unlike ports.When the set up process of the subflow has completed then the TCP Stack of the sending host devide the data across the subflow.Each sub flow maintain its own sequence space and control its own congestion window.

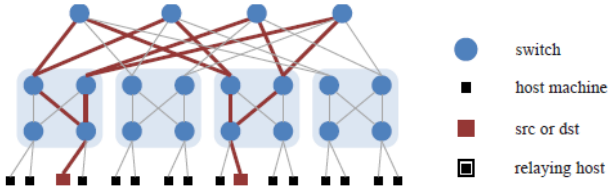


Figure: 27 MPTCP subflow

MPTCP has following advantages.

1) Better Aggregate Throughput;
2) Better Fairness;
3) Better Robustness;

Effect of subflow on the throughput shown in figure 28.

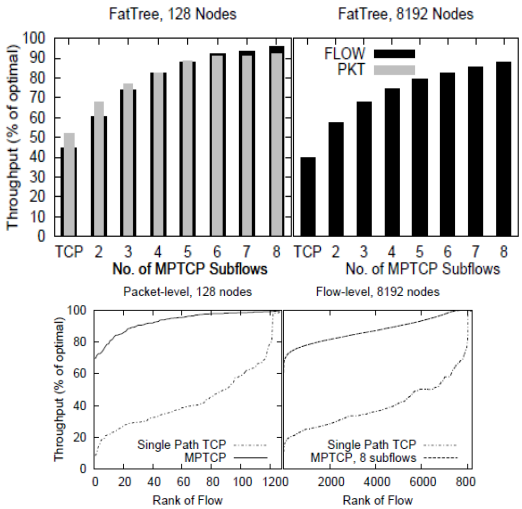


Figure: 28 effect of MPTCP subflow on the throughput

4. CONCLUSION;

In research of this paper we discuss in detail the TCP incast throughput collapse problem in Data Center Networks at Transport layer. This research work has two phases, in first phase we discuss the incast problem in detail, and how the incast problem is generated. In second phase we discussed the available solution that is useful to mitigate the effect of incats problem at Transport layer. In this paper we summarized all the available solution of Incast problem at Transport layer perspective .We explored that there are two main approach, avoiding timeouts and reducing the penalty of timeouts that are useful to mitigate the effect of incast at data center networks from an Transport layer perspective. Many researchers contribute their efforts but further research is requiring at level of avoiding timeouts. This approach will give us the fruitful result from Transport layer perspective. We divide in our research work the solution of incast at Transport layer into the two categories, modifying the TCP parameter and Enhanced TCP protocol. Multiple techniques are proposed at multiple layers and still further research is required at enhanced TCP protocol at transport layer.

5 References

[1] Irfan Riaz Shohab et al, International Journal of Computer Science and Mobile Computing, Vol.3 Issue.4, April- 2014, pg. 459-474

[2] Cerf, V. G., & Icahn, R. E. (2005). A protocol for packet network intercommunication. ACM SIGCOMM Computer Communication Review, 35(2), 71-82.

[3] Gosai, A. M., Goswami, B. H., & Kar, U. Evolution of Congestion Control Mechanisms for TCP and Non TCP Protocols.

[4] Wu, H., Feng, Z., Guo, C., & Zhang, Y. (2013). ICTCP: incast congestion control for TCP in data-center networks. IEEE/ACM Transactions on Networking (TON), 21(2), 345-358.

[5] Gibson, G. A., Nagle, D. F., Amiri, K., Butler, J., Chang, F. W., Gobioff, H., ... & Zelenka, J. (1998, October). A cost-effective, high-bandwidth storage architecture. In ACM SIGPLAN Notices (Vol. 33, No. 11, pp. 92-103). ACM.

[6]Adaptec." What RAID level is Right for me",[online]1996, https://www.adaptec.com/en-us/_common/compatibility/_education/raid_level_compar_wp.htm,(Acceesed:Nov 2013)

[7] Zhang, J., Ren, F., & Lin, C. (2011, April). Modeling and understanding TCP incast in data center networks. In INFOCOM, 2011 Proceedings IEEE (pp. 1377-1385). IEEE.

[8] Phanishayee, A., Krevat, E., Vasudevan, V., Andersen, D. G., Ganger, G. R., Gibson, G. A., & Seshan, S. (2008, February). Measurement and Analysis of TCP Throughput Collapse in Cluster-based Storage Systems. In FAST (Vol. 8, pp. 1-14).

[9] Zhang, J., Ren, F., & Lin, C. (2013). Survey on transport control in data center networks. Network, IEEE, 27(4).

[10] Chen, Y., Griffith, R., Liu, J., Katz, R. H., & Joseph, A. D. (2009, August). Understanding TCP incast throughput collapse in data-center networks. InProceedings of the 1st ACM workshop on Research on enterprise networking(pp. 73-82). ACM.

[11] Brad Hedlund,"TCP Incast and Cloud Application Performance",[online] May 1,2011, http://bradhedlund.com/2011/05/01/tcp-incast-and-cloud-application-performance/,(Accessed:Oct 23,2013)

[12] Microsoft Support"The OSI model's seven layers definition and explained",[online]27Feb 2002, https://suport.microsoft.com/kb/103884,(Acceesed:Feb2014)

[13] Wiki depidia"Transport Layer",[online]27Feb 2013, https:/en.wikidepia.org/wiki/transportlayer ,(Acceesed:Feb2013)

[14] Raiciu, C., Barre, S., Pluntke, C., Greenhalgh, A., Wischik, D., & Handley, M. (2011, August). Improving datacenter performance and robustness with multipath tcp. In ACM SIGCOMM Computer Communication Review (Vol. 41, No. 4, pp. 266-277). ACM

[15]Academia.edu." Evolution of Congestion Control Mechanismsfor TCP and Non TCP Protocols ",[online]2013, http://www.academia.edu/4668667/Evolution_of_Congestion_Control_Mechanisms_for_TCP_and_Non_TCP_Protocols,(Acceesed: Jan 20014).

[16] S. Floyd and T. Henderson. The NewRenoModification to TCP's Fast Recovery Algorithm.RFC 2582 (Experimental), April 1999.Obsoleted by RFC 3782

[17] Kevin Fall and Sally Floyd. Simulation-based comparisons of Tahoe, Reno and SACK TCP.Computer Communication Review, 26(3):5–21, July 1996.

[18] M. Allman, H. Balakrishnan, and S. Floyd. Enhancing TCP's Loss Recovery Using Limited Transmit. RFC 3042 (Proposed Standard), January2001.

[19] Padhye, Jitendra, Victor Firoiu, Don Towsley, and Jim Kurose. "Modeling TCP throughput: A simple model and its empirical validation." In ACM SIGCOMM Computer Communication Review, vol. 28, no. 4, pp. 303-314. ACM, 1998

[20]"CISCO."giga bit tcp" ,[online]2012 ,http://www.cisco.com/web/about/ac123/ac147/archeive_ issue/ipj-92.gigabit_tcp.html ,(Acceesed: Jan 20014).

[21] Mark Allman and Vern Paxson. On estimating end-to-end network path properties.SIGCOMM Comput. Commun. Rev., 31(2supplement):124–151, 2001

[22] Pasi Sarolahti and Alexey Kuznetsov. Congestion control in Linux TCP. In Proceedings of the FREENIX Track: 2002 USENIX AnnualTechnical Conference, pages 49–62, Berkeley,CA, USA, 2002. USENIX Association.

[23] Vern Paxson and Mark Allman. Computing TCP's Retransmission Timer. RFC 2988.

[24] Mohit Aron and Peter Druschel. Soft timers:efficient microsecond software timer support for network processing. ACM Trans. Comput.Syst., 18(3):197–228, 2000.

[25] Ren, Yongmao, Yu Zhao, Pei Liu, Ke Dou, and Jun Li. "A survey on TCP Incast in data center networks." International Journal of Communication Systems(2012).

[26] Hongyun Zheng Changjia Chen Chunming Qiao Understanding the Impact of Removing TCP Binary Exponential Backoff in Data Centers "2011 Third International Conference on Communications and Mobile Computing

[27] Vijay Vasudevan,, Amar Phanishayee, Hiral Shah, Elie Krevat, David G. Andersen, Gregory R. Ganger, Garth A. Gibson, and Brian Mueller. "Safe and effective fine-grained TCP retransmissions for datacenter communication." InACM SIGCOMM computer communication review, vol. 39, no. 4, pp. 303-314. ACM, 2009.

[28] R. T. Braden. Requirements for InternetHosts|Communication Layers. Internet Engineering Task Force, Oct. 1989. RFC 1122.

[29]Y. Chen, R. Griffith, J. Liu, R. H. Katz, and A. D. Joseph,"Understanding tcp incast throughput collapse in datacenter networks,"in WREN'09: Proceedings of the 1st ACM workshop on Research on enterprise networking. New York, NY, USA: ACM, 2009, pp. 73–82.

[30] M. Alizadeh, B. Atikoglu, A. Kabbani, A. Lakshmikantha, R. Pan, B.Prabhakar, and M. Seaman, "Data Center Transport Mechanisms:Congestion Control Theory and IEEE Standardization," in the 46th Annual Allerton Conference, Illinois, USA, Sep. 2008, pp. 1270–1277

[31]Wu, Haitao, Zhenqian Feng, Chuanxiong Guo, and Yongguang Zhang. "ICTCP: incast congestion control for TCP in data-center networks." IEEE/ACM Transactions on Networking (TON) 21, no. 2 (2013): 345-358.

[32]Raiciu, Costin, Sebastien Barre, Christopher Pluntke, Adam Greenhalgh, Damon Wischik, and Mark Handley. "Improving datacenter performance and robustness with multipath tcp." In ACM SIGCOMM Computer Communication Review, vol. 41, no. 4, pp. 266-277. ACM, 2011.

[33] Aized Amin Soofi et al, International Journal of Computer Science and Mobile Computing, Vol.3 Issue.3, March- 2014, pg. 15-21 © 2014, IJCSMC All Right

5.BIOGRAPHY;

- ***Irfan Riaz Shohab*** *is a student of MS in Computer Science College of Computer Science and Information Studies, Government College University, Faisalabad, Pakistan & also working as a Communication Engineer (Satellite) at Dunya NEWS TV Faisalabad Bureau. E-mail: shohabsons@gmail.com 03338369095*
- ***Muhammad Younas*** *is working as a Lecturer in College of Computer Science and Information Studies, Government College University, Faisalabad, Pakistan. E-mail: younas.76@gmail.com*
- ***Ramzan Talib*** *is working as a Principal (Associate Professor) in College of Computer Science and Information Studies, Government College University, Faisalabad, Pakistan. E-mail: ramzan.talib@gcuf.edu.pk*
- ***M.Umer Sarwar*** *is working as a Assistant Professor in College of Computer Science and Information Studies, Government College University, Faisalabad, Pakistan. E-mail: sarwaroner@gmail.com*