Performance Analysis of TCP Incast with TCP Lite and Abstract TCP

R.Amrutha
Department of Electronics and communication Engineering
SRM University,
Kattankulathur, Tamilnadu 603203, India
amruthaammu27@gmail.com

V.Nithya
Department of Electronics and communication Engineering
SRM University,
Kattankulathur, Tamilnadu 603203, India
nithyaamirtha@gmail.com

***Abstract*—Data center networks widely uses TCP for communication between the servers as it provides reliability and congestion control. However, TCP does not work well for certain type of communication patterns. One such a pattern is barrier synchronized many to one communication. During this type of data transmissions, multiple servers simultaneously transmits data to a single client and also the client cannot issues its next request until it receives all the data from current transmission. This will leads to severe throughput collapse in the networks which is called as TCP Incast. There are numerous other variants of TCP which will provides better congestion avoidance than TCP. In this paper, we analyze the performance of TCP Incast using QualNet simulator with different TCP variants namely TCP Lite and Abstract TCP. The performance analysis is based on the metrics such as bytes received, throughput, packet loss and number of retransmission**.

Keywords—TCP; TCP Incast; TCP Lite; Abstract TCP

I. INTRODUCTION

Data center is a restricted access area containing automated system that constantly monitors server activity, web traffic and network performance. The main qualities of a data center networks are low latency, high bandwidth and restricted switch buffer size. Due to this, cloud computing services make use of data center for large scale general computation, web search and cluster storage. TCP is used for data transmission between the servers as it provides more benefits in terms of both performance and cost wise. This is because TCP is the existing technology with good reliability and congestion control features. In certain specific applications like Map reduce, the performance of TCP is found to be very poor when barrier synchronized many to one communication takes place. During this communication pattern, congestion happens when the network traffic exceeds the switch buffer size.

TCP Incast is the recently found problem which degrades the performance of data center networks. There have been many proposed solution for Incast. Incast was first reported in the design of scalable storage architecture by D. Nagleet al [1]. They found that multiple packet loss and timeouts happens during barrier synchronized many to one communication. They mitigate the incast congestion by reducing the client's buffer size. The root cause for TCP Incast is time outs and limited buffer size of the switch. Two main approaches that have been proposed to reduce the incast problem are disabling the slow start to avoid retransmission timeout [2] and reducing the retransmission time out (RTO) from millisecond to microsecond [3]. But none of them helped. The mathematical model of [4] provides better understanding of TCP incast which paved way to find many solutions.

Furthermore solutions like DCTCP [5], IATCP [6] and LTTP [7] provide congestion control algorithms for the data center networks. DCTCP uses explicit congestion notification (ECN) to detect the network congestion and provide window based control methods. IATCP is a rate based congestion control algorithm which controls the total number of packets injected to meet the bandwidth delay product (BDP) of the network. In LTTP, TCP transport protocol is totally replaced with UDP. As UDP is connectionless unreliable protocol, Transmission friendly rate control (TFRC) and Luby transform (LT) codes are incorporated to provide congestion control and reliability. There are some other protocols which prefer congestion avoidance than congestion control as it is more appealing than recovering the lost packets. In ICTCP [8], congestion avoidance is carried out in receiver side as the knowledge about available bandwidth and throughput of all the connection is known at the receiver.

The theoretical model from [9] indicates that the throughput collapse is mainly caused by two types of time outs namely BHTO and BTTO. BTTO happens at the tail of the data packets and dominate the throughput whereas BHTO happens at the head of the data packets and governs the throughput. Solution for this timeout problem is provided in [10] by employing a simple drop tail queue management scheme called GIP at the switch.

In [11], detailed study about congestion problem in Adhoc networks is carried out with existing TCP variants like TCP Reno, TCP Tahoe and TCP Lite. A comparative analysis between all the TCP variants is provided in [12]. This helps in identifying the advantage and disadvantages among the TCP variants namely TCP Tahoe, TCP Reno, TCP New Reno, TCP Lite, TCP Vegas, TCP West woods and TCP Fack.

Some of the recently developed TCP variants provide better congestion control than TCP. This is because the basic TCP comprises of only slow start, congestion avoidance and fast retransmit whereas in TCP variants additional features are available.

978-1-4799-8553-1/15/$31.00 © 2015 IEEE

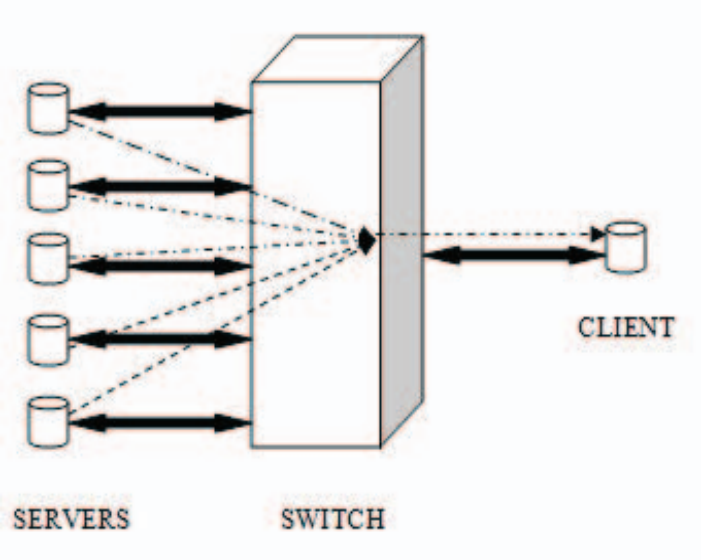

Fig.1. A Typical Incast Scenario

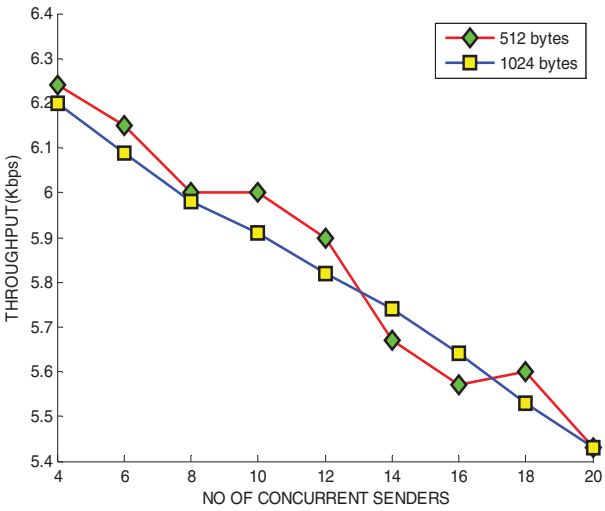

Fig.2. TCP Incast - Throughput collapse

In this paper, the TCP Incast scenario is designed and analyzed with different TCP variants namely TCP Lite and Abstract TCP using QualNet simulator.

The rest of the paper is organized as follows. Introduction of TCP Incast problem is given in section II. Section III describes the TCP Lite and Abstract TCP. Section IV deal with simulation parameter and methodology. Section V presents the results and discussions. Conclusions are shown in section VI.

II. TCP INCAST PROBLEM

Incast has exactly the reverse meaning of broadcast. During broadcast, one node sends out messages to multiple other nodes. While during incast multiple nodes send messages to the same node. Client requests data from multiple servers and the servers upon receiving the request from the client transmit the data simultaneously to the receiver. Due to its limitation on the switch buffer size, not all packets can go through at same time which causes congestion and leads to dramatic decrease in throughput.TCP Incast occurs mainly due to two types of time outs namely block head time out which occurs when the number of concurrent servers are small and block tail time out [9] which occurs when the number of concurrent servers are larger. It can be solved by either reducing the number of packet loss or by improving the quick recovery of lost packets.

Fig.1 shows the typical incast scenario in which multiple servers simultaneously transmitting to the client.Fig.2 shows the graph in which throughput is plotted against the number of concurrent senders/servers for different workloads. Here the workload is as follows. Each server transmits 100 packets of size 512 bytes and 1024 bytes to the client. File transport protocol is used to transmit data packets between the servers and the client as this application generates TCP traffic. This graph shows that the throughput dramatically decreases with the number of concurrent servers which symbolizes the incast scenario.

III. TCP VARIANTS

The congestion control mechanism provided by the basic TCP is not enough to handle the network load in all kind of communication pattern. Therefore it has been modified time to time as per need. The modified versions of TCP are called as TCP variants. Some of the variants are TCP Reno, TCP New Reno, TCP SACK, TCP Vegas, TCP Lite and Abstract TCP. In our work, we considered TCP Lite and Abstract TCP as the remaining variants are already discussed in detail.

A. TCP Lite

TCP Lite retains the basic principle of TCP such as slow start, congestion avoidance and fast retransmission. However it has some other additional features like fast recovery, big window and protection against wrapped sequence numbers. Due to these two options it provides congestion avoidance and band utilization. In basic TCP, two possible issues that concerns using TCP options are overhead and compatibility. TCP Lite easily overcomes these issues by using PAWS and nagle algorithm. TCP Lite provides better way for fast retransmission when compared with other TCP variants.

B. Abstract TCP

This TCP variant is based on TCP Reno. It simplifies and omits some features of RENO to improve the performance of the transport protocol. The mechanism that followed by Abstract TCP does not includes Nagle algorithm and keep alive probe option. It provides better congestion control.

IV. SIMULATION PARAMETERS

The parameter setting for the experiments are illustrated in Table I. All the simulation work is carried out using QualNet Simulator. A network with S number of servers and a

978-1-4799-8553-1/15/$31.00 © 2015 IEEE

client is designed. Wired link is provided between the servers and the client via a switch. Generic FTP is considered, as this application transfers data packets from servers to client over a TCP based networks. In our experiment, same amount of data will be transmitted to the client from the multiple servers. Simulation is carried out by varying the TCP variants for each value of S. Parameters that are analyzed for evaluation includes throughput, bytes received, packet loss and number of retransmission.

TABLE I. SIMULATION PARAMETERS

Network Properties	
Simulation time	120s
Terrain	1500*`1500m^2
Node placement	Random
Seed	1
Network Topology	
No of server(S)	2,4,6...14
No of client	1
Link type	Wired
Link Data Rate	10 Mbps
Application	Generic FTP
No of packets transmitted by each server(N)	100
Packet size	512 bytes
Protocols	
Routing protocol	Bellman Ford
MAC Protocol	Abstract MAC
Transport Protocol	TCP Lite , Abstract TCP

V. RESULTS AND DISCUSSION

A. Bytes Received Analysis

It denotes the total number of packets received by the client. From Fig.3 it can be seen that TCP Lite delivers more number of packets to the client, when compared with Abstract TCP. In Abstract TCP, the largest window that can be used is 65535 bytes as its TCP header uses a 16 bit field to report the receive window size to the server. TCP Lite has a new TCP option called window scale option which allows windows larger than 65535 bytes. Due to this option, the client receives more number of bytes when TCP Lite is used as transport protocol in incast scenario. The graph shows that the amount of bytes received decreases with number of concurrent servers in TCP Lite whereas in Abstract TCP it fluctuates.

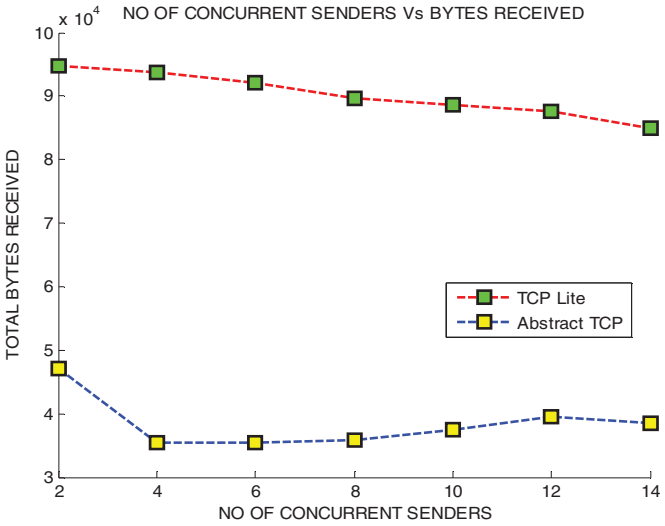

Fig.3. No of Concurrent Senders Vs Bytes Received

B .Packet Loss Analysis

It is defined as the discarding of data packets in a network when a device (switch) is overloaded. It is calculated by subtracting the total number of packets received by the client from the total number of packets sent by the server [11]. The packet loss is analyzed with varying TCP Lite and Abstract TCP transport protocol. TCP Lite consists of keep alive probe which is used to identify the failure links in a network. This method will reduce the packet loss that occurs due to link failure by routing the data via another link. Therefore, in TCP Lite packet loss occurs only when the buffer gets overloaded. Abstract TCP experiences more number of packet loss as it does not have keep alive probe option. Packet loss should be less for transport protocol to perform better. According to our simulation results from Fig.4, better performance is shown by TCP Lite as it has minimum packet loss.

C. Throughput Analysis

It is the ratio of total number of packets received from all the servers to the finishing time of the last sender [4]. In this analysis, it is observed as expected that the throughput dramatically decreases with the number of concurrent servers for both TCP Lite and Abstract TCP as we are considering for TCP Incast scenario. But the throughput values are higher for TCP Lite in comparison to Abstract TCP. Fig.5 shows similar response as that of Fig.3. This is because of the reason that in our work we use the parameter total number of bytes received

978-1-4799-8553-1/15/$31.00 © 2015 IEEE

to calculate the throughput value. As the throughput and the amount of bytes received are directly proportional, we got similar graphical response. But the resultant values obtained are different for throughput and amount of bytes received.

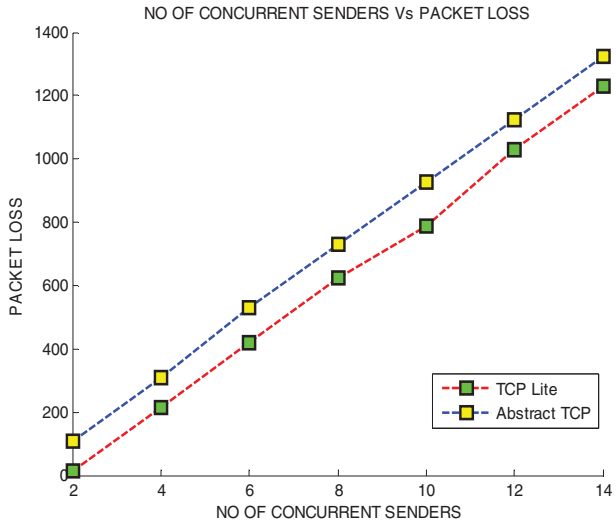

Fig.4. No of Concurrent Senders Vs Packet Loss

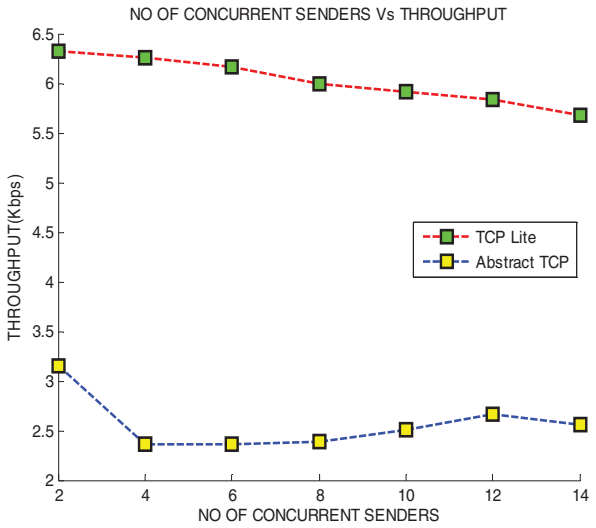

Fig.5. No of Concurrent Senders Vs Throughput

D. Retransmission analysis

Whenever a server transmits packet to the client, it retains a copy of the packets it sent until the client has acknowledged that it received it. If no such acknowledgement is received within a certain period of time, timeout occurs which leads to retransmission of the packet. In our analysis the number of retransmission increases with the number of concurrent servers in Abstract TCP whereas in TCP Lite the number of retransmission increases slightly for lower number of concurrent servers and it almost reduces to zero for higher number of concurrent servers. From the packet loss analysis, it can be clearly seen that Abstract TCP experiences more number of packet loss. If the packet loss is high, the number of retransmission will also be high and it will lead to severe traffic.TCP incast scenario already have enough traffic due to barrier synchronized communication pattern. If Abstract TCP is used as transport protocol in that scenario, it will further degrade the network performance. From Fig.6 it is very clear that TCP Lite performs well.

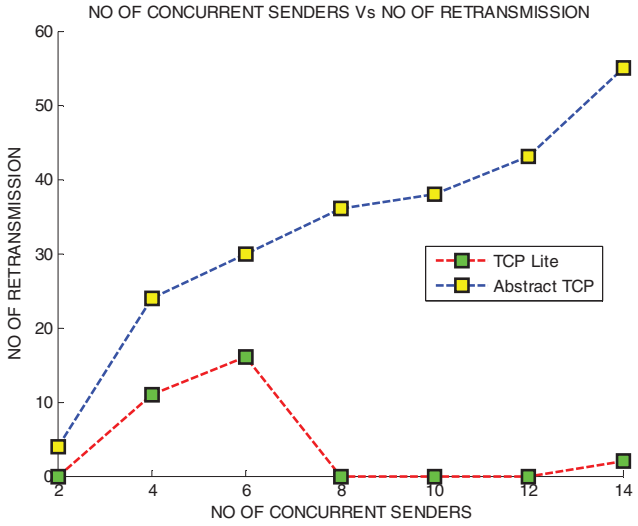

Fig.6. No of Concurrent Senders Vs No of Retransmission

VI.CONCLUSIONS

In this paper, the TCP Incast scenario is designed for different workloads using QualNet simulator. This paper also analyzed the performance of TCP Incast based on the parameters such as bytes received, throughput, packet loss and number of retransmission with different TCP variants. In order to provide a better communication, a transport protocol should have high throughput, minimum packet loss and minimum number of retransmissions. From our simulation results it is very clear that TCP Lite outperforms Abstract TCP as it achieves the above criterion.

REFERENCES

[1] D. Nagle, D. Serenyi, and A. Matthews, “The Panasas activeScale storage cluster - delivering scalable high bandwidth storage,” in *Proc. ACM/IEEE SC2004 Conference*, 2004.

[2]. A. Phanishayee, E. Krevat, V. Vasudevan, D. G. Andersen, G. R. Ganger, G. A. Gibson, and S. Seshan, "Measurement and analysis of TCP throughput collapse in cluster-based storage systems," in Proc. Of FAST, 2008.

[3] V. Vasudevan, A. Phanishayee, H. Shah, E. Krevat, D. G. Anderson, G. R. Ganger, G. A. Gibson, and B. Mueller, “Safe and effective finegrained TCP retransmissions for datacenter communication,” in *Proc. ACM SIGCOMM*, 2009.

978-1-4799-8553-1/15/$31.00 © 2015 IEEE

[4]. Y. Chen, R. Griffith, J. Liu, R. H. Katz, and A. DJoseph, "Understanding TCP Incast throughput collapse in data center networks," in Proc. Of ACM WREN, 2009.

[5]. M. Alizadeh, A. Greenberg, D. A. Maltz, J. Padhye, P. Patel, B. Prabhakar, S. Sengupta, and M. Sridharan, "Data center TCP (DCTCP)," in*Proc. ACM SIGCOMM 2010*, New York, NY, USA, 2010, pp. 63–74.

[6] Jaehyun Hwang , Joon Yoo and Nakjung Choi," IA-TCP: A rate based Incast-Avoidance Algorithm for TCP in data center networks" in *IEEE ICC 2012* - Communication QoS, Reliability and Modeling Symposium

[7]. Changlin Jiang, Dan Li, *Member, IEEE,* and Mingwei Xu, *Member, IEEE*, "LTTP: An LT-Code Based transport protocol for many-to-one communication in data centers" in IEEE Journal on Selected areas in Communications, vol. 32, no. 1, January 2014.

[8] H. Wu, Z. Feng, C. Guo, and Y. Zhang, "ICTCP: Incast congestion control for TCP in data center networks," in *Proc. ACM CoNEXT*, 2010.

[9]. J. Zhang, F. Ren, and C. Lin, "Modeling and understanding TCP incast in data center networks," in *Proc. IEEE INFOCOM 2011*, Apr. 2011,pp. 1377 –1385..

[10]. Jiao Zhang, Fengyuan Ren, Li Tang, and Chuang Lin Dept. of Computer Science and Technology, Tsinghua University, Beijing, China. Taming TCP Incast throughput collapse in data center networks 2013.

[11] Poonam Tomar and Prashant Panse, "A Comprehensive Analysis and Comparison of TCP Tahoe, TCP Reno and TCP Lite" in (IJCSIT) International Journal of Computer Science and Information Technologies, Vol. 2 (5) , 2011, 2467-2471.

[12]. Komal Zaman, Muddesar Iqbal, Muhammad Shafiq, Azeem Irshad and Saqib Rasool**, "** A Survey: variants of TCP in Ad-hoc networks" in ACSIJ Advances in Computer Science: an International Journal, Vol. 2, Issue 5, No.6 , November 2013.

978-1-4799-8553-1/15/$31.00 © 2015 IEEE