Published by :
http://www.ijert.org

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 5 Issue 11, November-2016

Quelling TCP Throughput Collapse in Data Center Networks using TCP Lite and RIO Mechanism

R. Amrutha[1],
[1]Assistant Professor,
Department of ECE, CKCET,
Cuddalore, India.

S. Nithya[2]
[2]Assistant Professor,
Department of ECE, CKCET,
Cuddalore, India.

***Abstract*— Data centre 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. The major causes of TCP Incast are the basic TCP which is used as a transport protocol and the congestion that happens at switch. In this paper, we use TCP Lite as transport protocol instead of basic TCP and also an active queue management scheme called Random early detection Input/Output with Coupled average queue (RIO-C) is incorporated to the incast scenario. This controls the congestion that occurs due to barrier synchronized transmission. The performance analysis of RIO-C mechanism is done on the basis of metrics such as bytes received, throughput, packet loss and number of retransmissions**.

Keywords*— *DCN;TCP; TCP Incast;TCP Lite; RIO-C

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 high bandwidth, low latency and restricted switch buffer size. Due to this, cloud computing services make use of data center for cluster storage large scale general computation and web search. 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. 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 fast retransmit, congestion avoidance and slow start whereas in TCP variants additional features are available. The general characteristics of Active Queue Management scheme are discussed in [13].

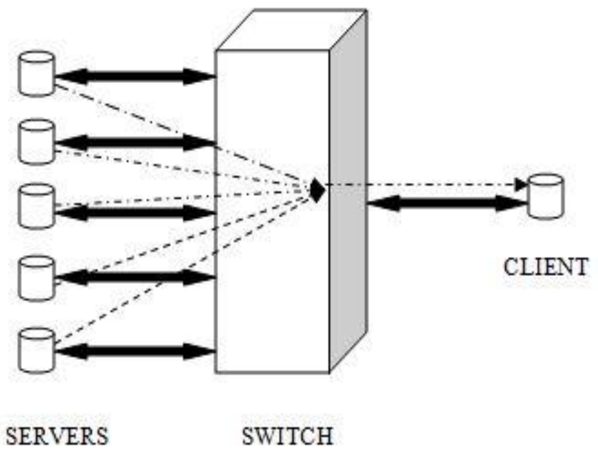

Fig.1. A Typical Incast Scenario

In [14], the authors have analyzed several active queue management algorithms with respect to their abilities of identifying and restricting disproportionate bandwidth usage, maintaining high resource utilization and their deployment complexity. The author also compares the performance BLUE, SFB, FRED and CHOKe using RED and Drop Tail as the evaluation baseline.

The authors of [15] have presented a simulation based comparison and evaluation of four popular queue management schemes: Stochastic Fair Queuing (SFQ), Random Early Detection (RED), Random Exponential Marking (REM) and Drop tail in terms of packet drop rate and delay.

In this paper, the TCP Incast scenario is designed and analyzed with RIO-C mechanism 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 RIO-C mechanism. Section IV deal with simulation parameter and methodology. Section V presents the results and discussions. Section VI deals with Conclusions.

II. TCP INCAST PROBLEM

Incast has exactly the reverse meaning of broadcast. During broadcast, one node transmits data to multiple other nodes. While during incast multiple nodes transmit data 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.

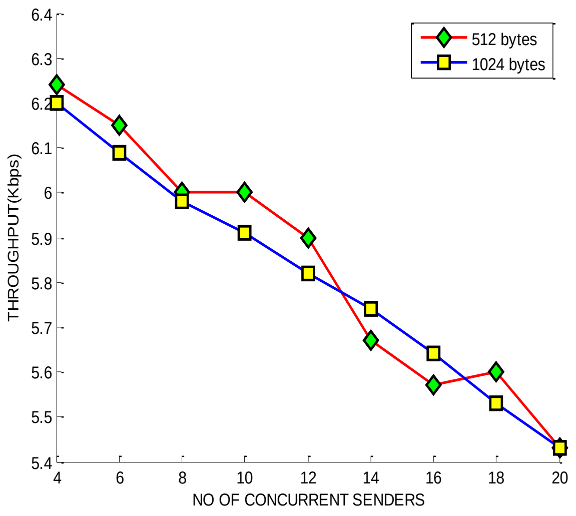

Fig.2. TCP Incast - Throughput collapse

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 LITE & RIO-C MECHANISM

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.

Random Early Detection with In/Out Bit which is a Multilevel RED Random Early Detection is incorporated in our TCP Incast scenario to control the network congestion. The incoming packets are marked through packet marking algorithm on the basis of Service Level Agreement between the customer and service-provider. The marking is done in two-colour (DP0 and DP1) level referring drop precedence.

RIO is a Multiple Average Multiple threshold (MAMT) variant of multilevel RED and can operate in both two and three colour modes. Two simple approaches to the MAMT variant of MRED are RIO-C (RED with In/Out and Coupled average Queues) and RIO-DC (RED with In/Out and Decoupled average Queues).

RIO-C derives its name from the coupled relationship of the average queue calculation. The queue length for packets of different colours can be calculated by adding its average queue to the average queues of colours of lower drop precedence. This scheme implies that the dropping probability for packets with higher drop precedence is dependent on the buffer occupancy of packets having lower drop precedence. Thus, there is an assumption that it is better to drop low priority packets for high priority packets.

A. RIO-C Algorithm

STEP 1: Strict priority scheduling is considered as Network scheduling.

STEP 2: The incoming packets are marked through "packet marking algorithm" on the basis of Service Level agreement between the client and server.

STEP 3: The marking is done in two-color level as

Yellow: Level Low Drop- DP0- High Priority

Green: Level High Drop- DP1 – Low Priority

STEP 4: Minimum and Maximum threshold value should be given separately for green and yellow packets.

STEP 5: Now RIO-C algorithm is carried out for yellow and green packets.

STEP 6: Consider the parameters average queue length, min threshold and max threshold to establish the following three zones.

- If average queue length < minimum threshold

  Normal operation-packets are accepted

- else if average queue length > maximum threshold

  Congestion control region – arriving packets will be dropped.

- else if mini thrsh < average queue length> max thrsh

  Congestion avoidance region- packets are discarded.

Where average queue length` denotes the Average of queue length reached by this queue during simulation.

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 client is designed. Wired link is provided between the servers and the client via a switch. Bottleneck link is created at the client side by reducing the data rate to 10Kbps. 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 and the results are analyzed on the basis of metrics such as throughput, bytes received, packet loss and packet drop.

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...20
No of client	1
Link type	Wired
Link Data Rate	10 Mbps
Application	Generic FTP
No of packets transmitted by each server	100
Packet size	512 bytes
Protocols	
Routing protocol	Bellman Ford
MAC Protocol	Abstract MAC
Transport Protocol	TCP Lite
`Scheduler & Queues	
Scheduler	Strict Priority
Queue	RIO-C
RIO-C Parameters	
Color mode	Two color
Color-1	Green
Color -2	Yellow
Yellow profile Minimum threshold	100
Yellow profile Maximum threshold	200
Green profile Minimum threshold	50
Green profile Maximum threshold	100

V. RESULTS AND DISCUSSION

A. Bytes Received Analysis

Total bytes received are defined as the total number of packets received by the client from the multiple server nodes. From Fig.3 it can be seen that if Active queue management (Random early detection with input/output-coupled) scheme is incorporated to TCP Incast scenario, more number of packets are delivered to the client when compared with general scenario without RIO-C. This is because of the reason that RIO-C mechanism reacts before the congestion happens. The graph shows that the amount of bytes received by increases with the number of concurrent senders for incast scenario with RIO-C.

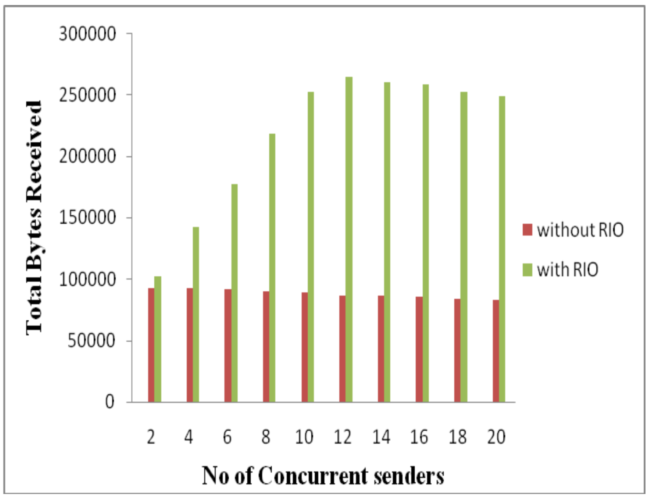

Fig.3. No of Concurrent Senders Vs Bytes Received

B. Throughput Analysis

Throughput is defined the number of successfully transmitted packet from the multiple sender nodes. Generally the throughput decreases with the number of concurrent senders. This behaviour is called as TCP throughput collapse which is a major issue in data center networks. When a queue management scheme (RIO-C) is incorporated, throughput increases with the number of concurrent senders. From Fig.4 it is very clear that TCP throughput collapse is totally overcomed by using active queue management scheme.

C. Packet Loss Analysis

Packet loss is calculated by subtracting the total number of packets received by the client from the total number of packets sent by the server. Fig.5 shows that if Random early detection with input/output is used, the number of packet loss is considerably reduced. This improves the throughput dramatically. The graph shows that the number of packet loss increases with number of concurrent senders for both the scenarios (with and without RIO-C). This is because of the reason that as the number of concurrent senders increases, the input load (number of packets transmitted by the multiple sender nodes to client) to the switch also increases. In order to avoid the congestion at switch corresponding amount of packet should be dropped as per RIO-C mechanism. Thus the packet loss that occurs in incast scenario with RIO-C is due to the intelligent drop of network packets inside a switch with the larger goal of reducing network congestion.

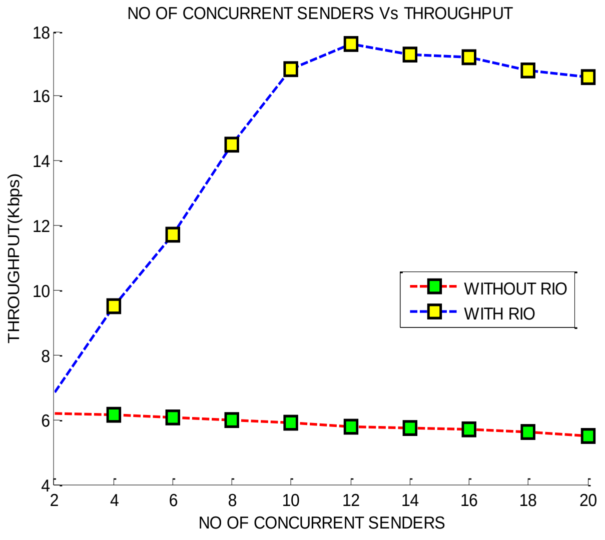

Fig.4. No of Concurrent Senders Vs Throughput

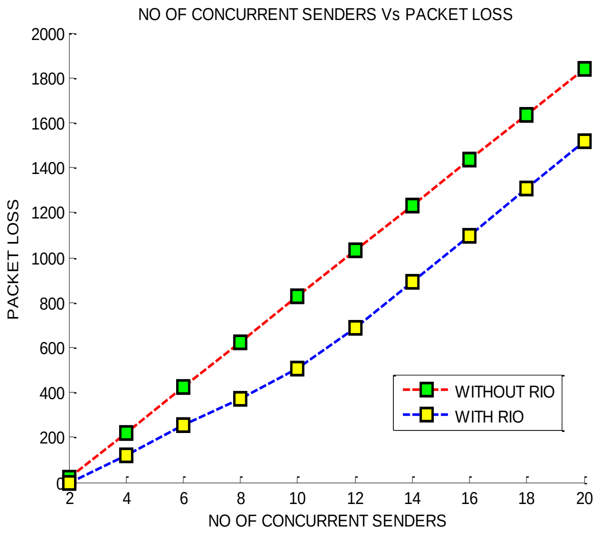

Fig.5. No of Concurrent Senders Vs Packet Loss

D. Packet Drop Analysis

Packet drop denotes the number of packets that dropped during simulation run. In RIO-C mechanism, packets are dropped once the network detects congestion. Generally Packets with low priority are dropped. Here in our scenario green packets are low priority packets. From Fig.6 it is very clear that all the dropped packets are green packets. This implies that the network reacts to the congestion efficiently by dropping low priority packets.

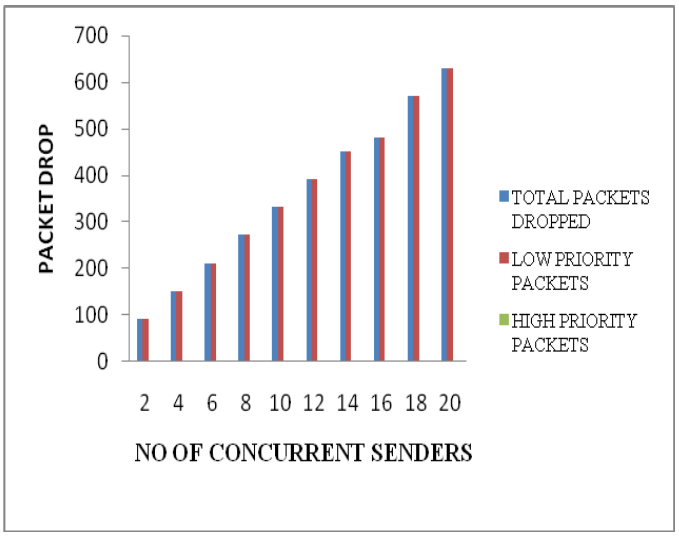

Fig.6. No of Concurrent Senders Vs Packet Drop

VI.CONCLUSIONS

A recently found problem in Data Center Networks is TCP Incast, in which TCP experiences severe throughput collapse in the setting of barrier synchronized many to one workload. The congestion control mechanism provided by the basic TCP is not enough to handle the network load in all kind of communication pattern. In our work, we considered TCP Lite as transport protocol as it provides better communication than basic TCP. Network layer level solution is also provided to the TCP throughput collapse by incorporating Active queue management scheme called Random early detection with input/output- coupled (RIO-C). In RIO-C mechanism, packets are dropped when the buffer gets close to becoming full, with the larger goal of reducing network congestion. From our simulation results, it is very clear that network Throughput is improved when an active queue management scheme is incorporated to TCP incast scenario.

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 fine grained TCP retransmissions for datacenter communication,” in *Proc. ACM SIGCOMM*, 2009.

[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.

[13] Richelle Adams, “Active Queue Management: A Survey” in IEEE Communications surveys & tutorials, vol. 15, no. 3, third quarter 2013, 1425-1476.

[14] G.F.Ali Ahammed and Reshma Banu, “Analyzing the performance of Active Queue Management Schemes” in International Journal of Computer Networks & Communication (IJCNC), vol.2,No.2, March 2010.

[15] P. T. Mahida, Kinjal Patel,Nayan Vanza and Siddharth Patel, “ A Comparative Analysis of Queue Management Techniques using NS-2 Simulator” in International Journal of Computer Applications (0975 – 8887) Volume 65– No.6, March 2013.