Challenges for TCP Protocols in Data Center Network: A Survey

Sivasankari M  
Department of Computer Science & Engineering  
Dynamic Safety Pte Ltd  
Tamilnadu

Hariharan S  
Department of Electronics & Communication Engineering  
Annamalai University, Chidambaram.  
Tamilnadu

Abstract— Data Center plays a significant role in data storage. Data Center Networks (DCN) have some unique features like reducing the low transmission of data, automatic scaling, low barrier switches compared with other networks. To adopt the extraordinary communication environment of DCNs, a vital issue for data center operators is to upgrade the deadline unstable service execution through enhancing the present system protocols, such as Transmission Control Protocol (TCP). In recent years various solutions have been proposed for improvement of TCP in Data Center Network. This paper presents a detailed study of existing TCP protocols with its merits and demerits related with DCN and also discuss the issues of TCP in data center networks like TCP Incast, Timeout problems, Queue buildup in switches, and delay in data transmission.

Keywords—Data center Network; ; TCP; Timeout; Latency

I. INTRODUCTION

The Data Center is a facility for housing computational and capacity frameworks interconnected through a similar network called Data Center Network (DCN). Due to an enormous development in the computational power, storage capacity and the number of inter-associated servers, the DCN withstand the challenges concerning productivity, dependability, what’s more, versatility. In spite of Transmission Control Protocol (TCP) is a time-tested transport protocol in the Internet, DCN difficulties, for example, lacking buffer space in switches, and bandwidth restrictions have incited to propose systems that enhance TCP execution or plan new transport ways for DCN. In recent years, the global data center business is extending quickly. Different online administrations facilitated by data center have invaded each part of everyday lives. Keeping in mind the end goal to keep up high client encounter, data center network operators, for example, Google, Amazon, and Microsoft, continue discovering approaches to diminish the reaction time. As a general rule, a client demand may bring about many streams to be created instantly at the back-end at a portion of data center networks. For better intuitiveness, these back-end flows are allotted with different communication due dates ranging from 10 ms to 100ms [1, 2]. In this communication procedure, if few flows miss their deadlines, their conveyed information would not be acknowledged by an aggregator (server for gathering reactions or appropriating demands). Other than bringing clients poor experience, missing the inactivity confinement of an administration is considered as a contravention of SLA (Service Level Agreement) [1].

Data center networks adopt the extraordinary communication environment, due to the vital issue for data center operators is to upgrade the deadline unstable service execution through enhancing the present system protocols, such as TCP [13, 14, 24]. To adapt to the relative low data transfer capacity and high latency in the routine WAN, TCP chooses conservative Additive Increase (AI), to recognize the most extreme accessible transmission capacity over the congestion avoidance phase. This issue observes, and recent work has examined new protocols fitting better in data center environments. However, none of these works have concentrated on the effect of AI congestion avoidance on the inefficiency issue of transmission capacity recognition in transport protocols for data center networks [1, 3]. Figure 1 shows the architecture of data center.

II. TCP RELATED ISSUES IN DATA CENTER NETWORK

The performance of TCP is not reasonable in data center networks due to its unique features when compared to
other IP networks. In this section, to discuss the significant issues that TCP faces when deployed in data center networks as shown in Figure 2.

**A. TCP Incast**

The execution of TCP is not acceptable in a data center because of its exceptional components when contrasted with other IP networks. TCP Incast is one of the significant execution issues in data center networks [1]. It is a tragic throughput crumple that happens when an expansive number of servers send information at the same time to a private beneficiary with high data transfer capacity and low round outting time. It has been characterized as the unyielding conduct of TCP that outcomes in under-use of the connection limit in numerous to one correspondence designs [2, 11]. TCP Incast was first found in the conveyed stockpiling framework. [4,21]. In this numerous to one correspondence design, the customer sends boundary synchronized information demands (i.e., the client won’t send data demands until the greater part of the senders finished the immediate needs) utilizing a substantial, consistent piece size to different servers through a switch for expanding the execution and steady quality. [17] Every server stores a piece of information square, which is allowed to as Server Request Unit (SRU). Figure 3 shows the incast problem of TCP.

**B. TCP Timeouts**

By trial, TCP timeouts are the underlying driver of the incast issue in data center networks [1]. TCP timeouts happen because of quick information transmissions from various servers which effectively stuffs the switch buffers bringing about overwhelming packet misfortunes. These timeouts force delays of many milliseconds in networks with round trip time of 10 or 100 microseconds can diminish the throughput by 90 percent [23]. Moreover, the never-ending timeouts can harm the execution of data center applications [3]. In the data center, the timeouts are predominantly created by the loss of packets from the end of data blocks, from the beginning of data blocks, and the loss of retransmitted packets. Figure 4 shows the causes of TCP timeouts.

**C. Latency**

Another issue is latency in a Data Center Network. The data center TCP flow characteristics are long queuing delay in switches; this is the main reason for latency in Data Center Network [5, 6].Data Center Networks hold two types of TCP flow: short lived and long lived, with a range that was typically arranging from 2KB to 100MB [4]. In this, short-lived flows are latency aware, while long-lived flows are latency unaware that can transfer significant traffic, which basis to grow the bottleneck queue otherwise the packets get dropped. Finally, when long and short flows divide the similar bottleneck queue, the short flows enlarged practice latency because of queue build up by large flows [1, 5]. The result is that huge amount of packet drops and repeated retransmissions. As an effect of consecutive retransmissions due to packet drops, TCP sender requires decreasing the size of the congestion window and wants more Round Trip Time (RTT) to finish the flow [9, 18].In addition to validating packet drops, the retransmission timer wants to be larger than the largest attainable RTT, which is too lengthly for a TCP flow to reach its deadline [1, 4]. Moreover, in Data Center Networks the greater part of the traffic is burst, and consequently, short-lived packets in TCP flows get dropped frequently dropped [19, 20].

**D. Queue Buildup**

Due to the different nature of cloud applications, little traffic, medium traffic and large traffic co-exist in a Data Center Network. The permanent and self-centered
nature of massive traffic makes the network to the point of excessive congestion and overflows the bottleneck buffer [7, 12]. Thus, when both small traffic and huge traffic cross through the same route, the small traffic performance is considerably affected due to the presence of the massive traffic [2, 4].

The following kinds of small traffic are degraded due to the existence of huge traffic:

- **Huge traffic occupies the most of the buffer, so the small traffic gets dropped for a high probability of packets.** TCP Incast has the similar problem because the small traffic performance is hugely affected by repeated packet losses and also the timeouts.

- **None of the packets are lost from the small traffic; increased queuing delays are suffered from queue behind the packets of huge traffic.** This problem has been named as Queue build-up. Minimizing the queue occupancy in the Data Center Network switches are the only solution for Queue build-up problem.

Most of the existing TCP alternatives take up next approach towards congestion control so that it does not succeed to decrease the queue occupancy [25]. A practical approach is desired to reduce the queue occupancy and defeat the problem of queue build-up.

### III. VARIOUS TRANSPORT PROTOCOLS FOR DATA CENTER NETWORK

#### A. Transmission Control Protocol

Data Center plays a significant role in the field of influence and for storing a large volume of data’s and database in various services and functions in business. Data Center Networks differs from other networks because DCN was having several to single connection prototype with the high data transmission rate. DCN reduce the low barrier switches, low transmission of data, automatic scaling and use various tree architecture. Normal TCP suffers from these Data center Networks [1, 2, 16, and 21]. So, now a day diplomatic solutions have been offer for resolving the problem of regular TCP in Data Center Networks. To present conservative solutions for existing transport layer protocol.

#### B. Data Center Transmission Control Protocol (DCTCP)

DCTCP is a kind of TCP protocol for data center networks. DCTCP proposed for producing multi-bit feedback to the end hosts within the network using specific Explicit Congestion Notification (ECN) method. To judge DCTCP at 1Gbps to 10Gbps speeds using artifact, shallow buffered switches. TCP proposed model of DCTCP delivers the same or higher throughput than TCP, whereas discrimination of less buffer space [1]. DCTCP additionally provides high burst tolerance and low latency for data transmission from one end to another end. DCTCP permits the applications to handle this background traffic, without impacting foreground traffic [1, 10].

The DCTCP algorithm has two main components [1, 2]:

- **Easy marking at the Switch:** Very straightforward and dynamic queue management method is employed from DCTCP. If the queue occupancy is above fixed point upon its receiving, the arriving packet is marked with code point or else, it is not corrected. This scheme ensures that sources are rapidly notified of the queue exceed. Implementing the most new switches can be re-purposed for DCTCP in RED marking scheme.

- **Receiver side ECN:** The only variation between a DCTCP and TCP receiver is the transmission back to the sender information in the code points. A sequence of ACK packets in ECN-Echo flag is set from the receiver. Otherwise, the congestion notification has been received from the sender (through the congestion window receiver). A DCTCP convey the exact series of marked packets back to the sender from the receiver.

Following are shows the Benefits of DCTCP:

- **Queue Buildup:** DCTCP senders start responding as quickly as queue length on an interface exceeds. Decreasing the queuing delay on congested switch ports. To reduce the impact of large flows on the finishing point time of small flows and also available in more buffer space.

- **Buffer pressure:** DCTCP also resolves the buffer pressure difficulty, because a congested ports queue length does not raise extremely large.

- **Incast:** In incast scenario, it is not easy to handle, where a huge number of corresponding small flows hit the same queue. If the number of small flows is so high, that still one packet from both flows is sufficient to overcome the buffer on a synchronized burst. [2]

#### C. Dead-Line Aware Transmission Control Protocol (D'TCP)

Deadline-Aware TCP (D'TCP), is a kind of Transmission Control Protocol, that handles bursts, in deadline-aware and without delay.

D'TCP makes two assurances:

- D'TCP permits to utilize a distributed and direct approach for bandwidth allocation.

- Traditional congestion avoidance algorithm along with ECN reaction is used for D'TCP.

A gamma-correction function is used to transform from deadlines to congestion window [4]. Performance evaluation
of D²TCP reduces the missed deadlines contrasted with DCTCP. [1, 4]. Following are the features of D²TCP:

- **D²TCP diminishes the missed deadlines by 20 percentage contrasted with DCTCP, which requires less than 100 extra lines of kernel code.**
- **Accomplishes high transmission capacity as TCP for foundation streams without corrupting On-Line Data Interchange (OLDI) execution.**
- **Background flow achieves high bandwidth.**
- **Diminishes the fraction of missed deadlines contrasted with DCTCP and deadline aware (D³) by 75 percentage and 50 percentage.**
- **OLDI’s provide more time for real computation in missed deadlines for D³.**

**D. Incast Congestion Control for Transmission Control Protocol (ICTCP)**

The problem of TCP Incast is discussed in detail by the center of attention on the connection between TCP sender and receiver, throughput for given RTT and receiver window. The TCP receiver part can know the throughput of all TCP links and the accessible bandwidth [3, 21]. Further, the receiver can control the burst ones of all the corresponding senders by changing the receiver window capacity of each TCP connection and also the congestion control window for receiver side should be made based on the each flow response time and delay independently. Based on these interpretation, the proposed incast congestion control for TCP avoid the TCP incast congestion for receiver side. Minimizing the packet loss before incast congestion is the objective of ICTCP. And another to recover after the loss happens [3, 15]. Consider this, receiver side achieves congestion control, ICTCP computes the obtainable bandwidth on the network boundary and present allowance for all incoming connections that increase the receiver window for superior throughput. The estimation of throughput based on not accessible flow with RTT and link capacity, Effective bandwidth, minimum-latency are progressed with network throughput. Sender side congestion control window for TCP, this approach adjusts the rate of the variation among the deliberate and predictable per-connection for the amount of data transfer over the estimated throughput based on the receiver window, as well as the final hop obtainable bandwidth to the receiver [22]. When the ratio is small, ICTCP enhances it receiver window. Otherwise, the rate was decreased.

**E. Low Latency Data Center Transport (L²DCT)**

L²DCT, a one of the data center transport protocol implemented in existing Transmission Control Protocol. Decreasing the short flows ending time is a main objective of the protocol. [1]. L²DCT does not require any modification to router hardware or application software. L²DCT has back off method for setting the transport layer window size. Additive increase mechanism is the main part of the L²DCT [8]. Using this method, the amount of data are previously sent, the end user makes use of the information conditional from Explicit Congestion Notification (ECN) regulated by their window size setting. The persistent increase in short flows and traditional back off automatically allows these flows to finish quickly. The goal of L²DCT is reducing the flow completion times for short flows. Application software or router hardware does not need any modification. And long flows are not affected by this method. Explicit Congestion Notification (ECN) adapting the flow rate to the amount of network congestion. This characteristic generally maintained by the installed router base [8, 21]. In data center traffic for data transfer phase, deadlines are meeting the end user. To missing the deadline rate is constantly smaller through L²DCT protocol, compared to previous deadline aware handles the data center transmission control protocols [1].

**F. Adaptive Acceleration Data Center Transmission Control Protocol (A²DTCP)**

A²DTCP is to avoid congestion with the AI mechanism, increase the flow sending rate steadily when large link bandwidth is available, plenty of bandwidth resource will leave unused [1]. As a result in long flow completion time and more bandwidth resource will leave unused. The overall results in long flow completion time and high deadline missing ratio. Another one, MI mechanism could attain much extra available bandwidth in the equal amount of time compared to AI. However, its drawback is noticeable as well. When congestion is already heavy, MI may inject excess packets into the network thereby make worse the presented congestion. Therefore, a new window rising mechanism should take on their advantages and avoid their unfavorable effects. More particularly, in a case of light congestion, the new method increases the CW assertively to make use of available bandwidth as quickly as possible. When the available bandwidth is insufficient, the window rising should be slowed down to avoid congestion. Based on the above study, this protocol should satisfy, the following [1]:

- **Let as many flows as probable meet their deadlines,**
- **Decrease the completion time of deadline-sensitive flows,**
- **Attain the obtainable bandwidth as quickly as possible in the congestion avoidance phase,**
- **Be companionable with existing switch hardware,**
- **and be responsive to the legacy TCP.**

Friendly Performance with Non-Deadline Traffic Make sure that the acceleration of critical flows by A²DTCP does not extensively delay the non-deadline flows when evaluated with other protocols. A²DTCP achieves this by nice consideration of the mixture of the flow urgent degree and network congestion level. In particular, even with a high urgent degree, as long as there is available bandwidth, the non-deadline flows still have a chance to gain needed bandwidth. To show this property, mix the deadline and non-deadline traffic in the data-intensive environment with the network topology. For evaluating the performance of A²DTCP in the multi-hop network, two general network topologies in data centers are selected here. Note that the topology of OLDI atmosphere is multi-hop too, due to its
design in fabric switch; however, the flow could only be unproductive at the Top of the Rack (ToR) switches.

IV. SUMMARY

To discuss throughout the paper, some alterations have been proposed to the novel design of TCP protocol. However, there is a sensitive need to optimize further the various TCP performance discussed above. Table 1 summarizes the comparative study of various TCP Protocol proposed for Data Center Networks. Apart from the innovation of the proposed TCP protocol, the table also highlights the advantages and shortcoming of each protocol. Table 2 summarize in which protocols need modifications in the sender, receiver and switches are arranged in a complex manner. The summary also includes the problems amongst TCP Incast, TCP Timeout, Queue build-up and Latency is alleviating by each TCP protocols.

V. CONCLUSION

Data centers have become an economic infrastructure for hosting a various range of cloud applications and for storing a large amount of data. Data center flows classically operate using the TCP as it is an older technology that provides consistent and planned bidirectional delivery of a stream of bytes from one application to the other application residing on the similar or two different machines. Most of these applications use many-to-one communication pattern to achieve performance efficiency. However, TCP is set up of data center networks; it is not capable of providing high throughput and the main problem of TCP Incast, Timeout, and latency. For this reason, need to redesign TCP primarily to handle the traffic in data center networks. In this paper, to present a detailed survey of recent times proposed transport protocols purposely to mitigate the TCP problems in the data center.
Table I. List of Protocols with their merits and demerits.

<table>
<thead>
<tr>
<th>PROTOCOL</th>
<th>ADVANTAGES</th>
<th>LIMITATIONS</th>
</tr>
</thead>
</table>
| DCTCP    | • To achieve the performance benefits of major modification TCP and ECN.  
          • It tries to avoid packet loss.  
          • DCTCP have the properties of Stability, convergence, and fairness to suitable for Data Center Network. | • DCTCP performance fails TCP Incast more than 35 nodes send in one aggregator. |
| ICTCP    | • ICTCP does not need any alteration at the sender side, only at the receiver side.  
          • TCP receive windows are efficiently corrected before packet loss occurs.  
          • TCP performance for TCP incast is improved by ICTCP in data-center networks. | • The major concern of the ICTCP shows that they achieve almost zero timeout and high throughput, the scalability but does not tell about how to handle incast congestion when there are a huge number of flows because of per-flow congestion control for ICTCP.  
          • ICTCP assume same switches in sender and receiver side so does not know how much buffer space use in ICTCP. |
| D²TCP    | • D²TCP easily deploys to avoid TCP Incast, Queue build-up, and high burst tolerance because it is built upon based on DCTCP.  
          • And also, it is deadline-aware, compared with DCTCP the missed deadlines minimize up to 75%. | • D²TCP are accurately similar to those of DCTCP.  
          • Same worker nodes in parallel are the major concern of the scalability. More research is required to conclude whether D²TCP can beat the TCP Outcast problem. |
| L²DCT    | • L²DCT same behavior as DCTCP.  
          • L²DCT, minimizing the flow completion times and solves the latency issues in data centers.  
          • It does not need any alteration at the receiver side; only at the sender side have the modification. | • L²DCT does not have the deadline awareness and no modifications in hardware switches.  
          • It does not have any flow size information.  
          • L²DCT does not give superior precedence to more urgent flow. |
| A²DTCP   | • A²DTCP is a deadline aware protocol which considers both network traffic and late transmission of data.  
          • By utilizing network traffic with a versatile increment rate that differs amongst existing approach, A²DTCP speed up the data transfer capacity discovery accordingly accomplishing high transmission capacity use effectiveness. | • A²DTCP does not design of end-to-end congestion detecting scheme to avoid the deployment hardness of ECN mechanism. |

Table II. Summary of TCP Protocols for Data Center Network.

<table>
<thead>
<tr>
<th>Proposed protocols for TCP</th>
<th>TCP Modification</th>
<th>Solve TCP Incast</th>
<th>Solve TCP time outs</th>
<th>Solve TCP Latency</th>
<th>Solve TCP Queue-Buildup</th>
<th>Deadline Aware</th>
</tr>
</thead>
<tbody>
<tr>
<td>DCTCP</td>
<td>Sender and Receiver side</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>ICTCP</td>
<td>Receiver side</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>D²TCP</td>
<td>Sender and Receiver side</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>L²DCT</td>
<td>Sender side</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>A²DCTCP</td>
<td>Sender and Receiver side</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
</tbody>
</table>
REFERENCE


