International Journal of Control Theory and Applications

ISSN : 0974–5572

© International Science Press

Volume 9 • Number 40 • 2016

# Survey on TCP Incast Problem in Datacenter Networks

**B. Thiruvenkatam[a] and M.B. Mukesh Krishnan[a]**

[a]*SRM University*

*E-mail: thiruvenkatam.b@ktr.srmuniv.ac.in, Corresponding Author*

***Abstract :*** Theexplosive growth ofdata generation necessitatesthe requirement for large size of storage to keep the data available in Datacenter. The data management costshas led to a fast evolution of distributed file systems and distributed storage. Inaddition, growingreal-time data is changing the programming paradigms and driving the need for faster hardware. Traditional storage networks and costs are growing exponentially to handle massive amount of data. Thisdrives new requirements to manage bursts of traffic in distributed networking environment. Hence Clusterbased storage systems are becoming an increasingly important in Datacenter. Traditional network designs can be massively oversubscribed in data center with the underlying concept of distributed storage.

Storage data input/output must be flexible and ensure reliable transmission to prevent loss or corruption.TCP is a transport layer protocol used by applications that require guaranteed delivery and provides both timeouts and retransmissions.TCP is used in applications that requires bulk storagetransfer in the modern fastest datacenters. TCP provides low latency, high bandwidth and synchronized parallelapplications. Modern datacenters requires cluster based storage environment where one client requesting data from plenty of servers as a synchronized read. This paper analysis the various ways through which the TCP incast problem can be mitigated.

***Keywords:*** *Incast, RTO, Datacenter, RTT, ECN.*

## 1. INTRODUCTION

### 1.1. TCP Incast

TCP Incast is a disastrous TCP throughput downfall that happens in a high-bandwidth less latency networks during multiple synchronized servers sending data to a common receiver in parallel. This type of many to one traffic pattern is quite nature in the Map Reduce, and search based applications in modern data centers.

In a clustered storage system, the client application requires a data set that may be stripped across various storage servers. The next data set request can be initiated only when all the other servers provided their portion of the data. Such a synchronized work load generally results inpackages over filled memories of the client port in the switch and will result in many packet losses. When the packet loss is high, TCP may encountered a time out that lasts the minimum of 200 ms determined as TCP minimum retransmission time out (RTO min).

TCP Incast issue is become inevitable in many datacenter applications like batch processing jobs and in particularly Map Reduce based applications. In Map Reduce, the intermediate key value pairs from many Mappers are transferred to appropriate Reducers during the Shuffle stage.

TCP incast issue starts when a parent server requests a data to cluster of the nodes received the request simultaneously. The cluster of nodes will in turn respond synchronously to the single parent. This will cause a micro burst of many machines sending TCP data streams simultaneously to one machine.

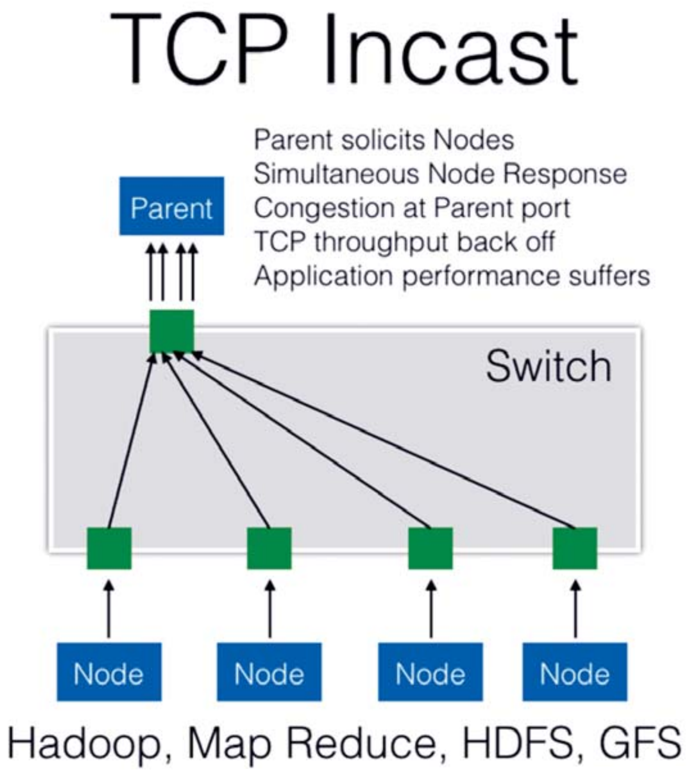
TCP Incast
Parent solicits Nodes
Simultaneous Node Response
Congestion at Parent port
TCP throughput back off
Application performance suffers
Parent
Switch
Node
Node
Node
Node
Hadoop, Map Reduce, HDFS, GFS

**Figure 1**

**Referred from http:** //bradhedlund.com/2011/05/01/tcp-incast-and-cloud-application-performance/

## 2. MODELING AND SOLVING TCP INCAST PROBLEM IN DATACENTER NETWORKS

In datacenter networks TCP incast problem creates catastrophic good put drop. It is identified that two type of time outs, Block Tail time out (BTTO) and Block Head Time out (BHTO) are the main cause for significant good put collapse. BTTO caused by special incast communicationpattern, it happens when the number of sender is less and caused by one of the last three packets in a block of sender is dropped. It is the inherent properties of TCP incast that if all the blocks of the current block of data are not completed the sender will not initiate new block of data.

BHTO occurs during large number of senders.When sending block of data, few senders complete their task and will wait for other to finish by not utilizing any bandwidth. Since the inherent properties of TCP window size determines the number of blocks being sent and received, there is a possibility of whole window can lost when the number of senders are high and more over when large amount of data was sent to small Ethernetswitch buffer, it will cause more packet drops and leads to Time out.

### 2.1. TCP incast scenario

If the senders are more, each sender sends onepacket and the bottleneck buffer size will obviously be exhausted. This causes the major packet drops and hence the TCP incast situation.

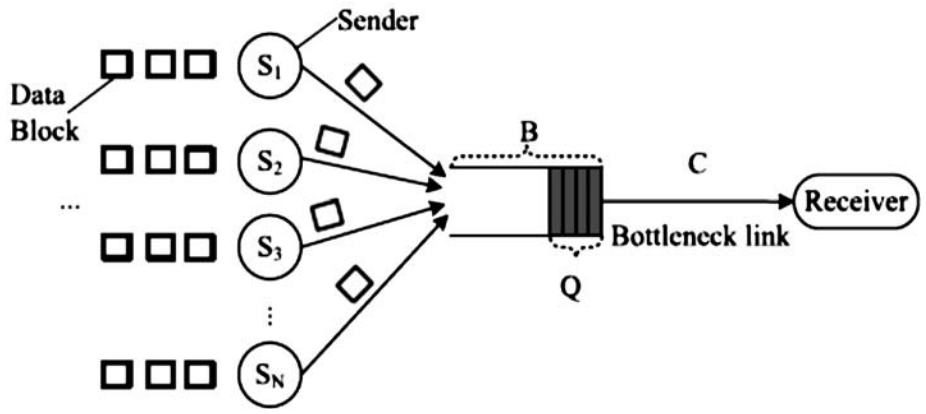

**Figure 2**

## 2.2. TCP incast issue mitigation using PRIORITY

TCP Incast problem can be resolved using priority based solution otherwise called as PRIN.This method reduces the send window of each connection at the start of block to avoid Block Head Time Out. In other words the congestion window size of every flow is shrink at the start of every data block to avoid Block Head Time out (BHTO) and configuring high priorityinthe threelast packets and to prevent Block Tail Time Out (BTTO) at the end of block. In order for successful transmission of the last three packets high priority has been set. By virtue of TCP, three packets priority will be set to high, as generally three duplicated ACKs will be needed to trigger the TCP fast retransmission.

PRIN is implemented using the existing properties of IEEE 802.1p differentiated code point, Type of service. IEEE 802.1P has been implemented in many switches and it is used along with the IEEE 802.1q for VLAN.

## 2.3. Implementation

Linux Kernel using the PRIN algorithm. TCP header is modified in the kernel to achieve the TCP incast congestion. TCP/UDP Socket interface between the application and TCP is used to enable and disable the implementation algorithm.

## 2.4. Testing

PRIN Performance is analyzed in test bed with many servers and in Gigabit Ethernet switches and concluded that all Timeouts (TO) have been averted and improves the Good put and confirmed that BHTO and BTTO are the main reason for TCP good put collapse.

NS-2 simulation platform is used to perform the validation of the PRIN model.

## 2.5. Advantages

PRIN mechanism provides high good put and higher percentage of link utilization with or without background traffic as compared to TCP New Reno a popular standard TCP implementation. There is many other UDP based application that runs in the datacenter network causing background traffic. The average number of Timeout (TO) in every data block issmall in PRIN.

PRIN mechanism supports Multi-Hop Technology where the Bottleneck link is not necessarily the final hop that connects to the receiver instead the bottleneck link can be connected to the other switch. PRIN mechanism works well irrespective of where and how the congestion link is located.

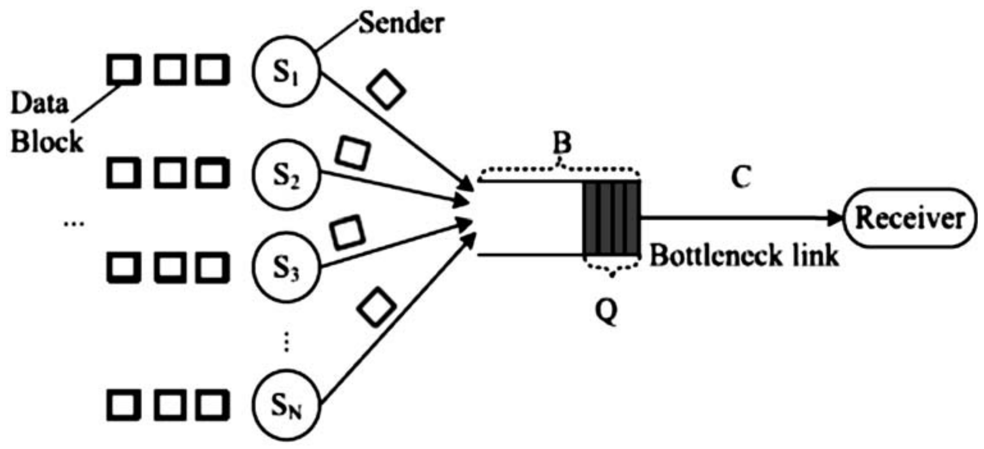

**Figure 3**

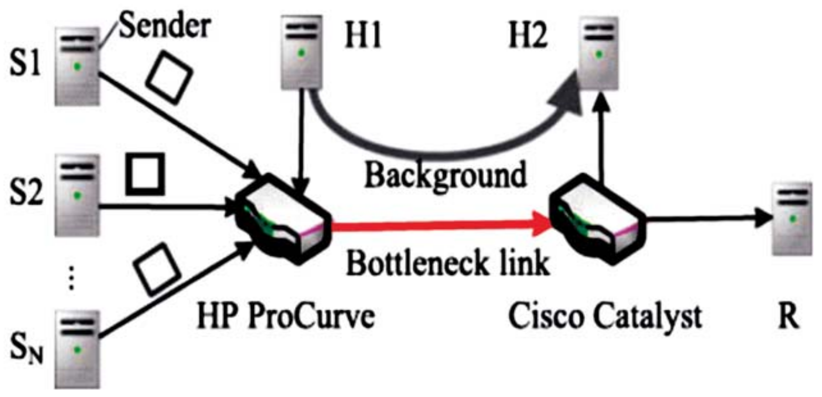

**Figure 4**

### 2.6. Single Hop topology

1. In Map Reduce based applications in datacenters, Reducer takes the data from mappersthat should not locate in the common rack as the reducer and one intermediate connection link is congested.

### 2.7. Limitation

This method assumed to support to less number of senders as the buffer size is smaller than the number of senders. Requires TCP header modification in Linux kernel and may require Open source platform support to achieve the PRIN implementation.

PRIN mechanism does not support implementation in windows operating system, associated driver software and its interface.

TCP slow start after the ACK's are lost is not considered in the model as it is assumed to have negligible impact.

## 3. SA-TCP - TO MITIGATE TCP INCAST IN DATACENTER NET WORKS

SA-TCP - Stochastic Adjustment TCP discuss about stochastically adjusting the congestion window to avoid synchronization during parallel flows.

In general Datacenter Networks are low propagation delays limited size switch buffers and high speed links. TCP in general is the most popular transport protocol used in internet and best suited for WAN application. Datacenter networks are facing more challenges in data transmission when compare to internet like microsecond RTTs (Round Trip Time), decreased amount of multiplexing. Because of the unique work load scale like barrier-synchronized request are common in datacenter environments such as file system reads and writes the same TCP when in DATACENTER networks has performance bottleneck.

In LINUX like contemporary operating systems the default Retransmission Time Out value would be set to 200ms as it is a reasonable value for WAN, but two or three orders of magnitude greater than the average round trip time in networks used in datacenter and it has negative impact in Datacenter Networks.

Since the number of concurrent sender's goes high, the work load overflow the buffer memory size atthe bottleneck switch leads to packet losses and subsequent TCP retransmissions. It is important to consider the TCP transfer mode and controlling the TCP congestion window. TCP congestion control window will be stochastically adjusted through an congestion avoidance algorithm to prevent many parallel flows so as to control the TCP incast issue.

Heavy traffic and network status in datacenter center networks create a precondition for incast congestion that barrier synchronized many to one traffic pattern is general in datacenter networks that mainly created by Map Reduce like application.

In TCP Communication pattern "Incast" scenario, communication from multiple servers to the same client is always synchronized in general. In this type of situation when congestion happens all the senders reduce the congestion window so that less amount of data will be sent to the client and increases the congestion window when there is no congestion in the network so that high amount of data will be sent at the same time.

## 3.1. Design

Congestion window is indicated as cwnd which can be stochastically increased and decreased that leads to design the TCP incast mitigation technique by using "Additive Increase and Multiplicative Decrease"(AIMD) – AIMD is a feedback control algorithm identified as best fit for TCP Congestion Avoidance. It usually combines the linear growth of the congestion control window with an exponential reduction during congestion time.

In SA-TCP algorithm if the packet loss is not identified when ACK arrives, the cwnd will additively increase in a stochastic step size that is stochastic cwnd increase and when there is packet loss identified the cwnd will be multiplicatively decrease.

## 3.2. Evaluation Testing

The performance of SA-TCP is evaluated through NS-2 simulation.

## 3.3. Assumptions

In this method TCP property of slow start phase is disabled as it causes the congestion window to increase exponentially and quickly and that will reduce the efficiency of stochastically modifying congestion window. In order to have AIMD (Additive Increase and Multiplicative Decrease)properties the TCP slow start phase has to be disabled.

## 3.4. Advantages

1. Most of the other famous TCP mitigation techniques uses ECN (Explicit Congestion Notification) enabled switches. But ECN is not supported across globe in datacenter network hardware and switches already placed in existing datacenter. This method does not uses Explicit Congestion Notification.

2. Compared to most famous method of TCP mitigation DCTCP (Datacenter TCP), this method needs modification only on the sender side and not in both sender and receiver.
3. This method does not require both the sender and receiver connected to a common switch and that the bottleneck link is not necessarily the final hop to the receiver there could be many bottleneck link and hence this method supports multiple topology and increased traffic.
4. This method supports LINUX and other open source platform and not like other
5. Famous TCP mitigation techniques (DCTCP, ICTCP) that are proposed by Microsoft developed windows operating system.
6. This method gives solutions on the bases of Transport Layer in TCP/IP stack and does not require modification on Application Layer.

### 3.5. Disadvantages

1. Need further investigation with more number of senders and heavy network traffic.
2. It is an assumption that TCP slow start disablement willnot have any bad effect.
3. SA TCP method considers only the synchronized flow of data transfer and does not discuss much about switch buffer size and assumed to have less volume of data traffic.

## 4. FLOW AWARE CONGESTION CONTROL

With the growth of cloud computing and various applications that need to be supported in datacenter networks the inter server network traffic has increased dramatically. In Datacenter the servers are generally placed adjacent when compare to internet and hence the inter server traffic will dramatically increase among and characterized as high speed and low latency networks. This type of traffic pattern in Datacenter networks are different from internet traffic.

It is essential to consider the data flow characteristics under incastscenario and look for congestion control in datacenter networks to flows that have other characteristics.

In Datacenter Long lived flow and throughput sensitive flows are account for less than 10% of all flows but carries more than 80% of the entire total traffic. Examples of such a long lived flows are live migration, software updates and distributed data processing applications. Whereas the short lived flows correspond to the query searching and remote procedure call -RPC

The short lived flows affecting the long living flows and particularly under TCP incast the long lived flows are affected more severally.

Identifying the intended target flows from many other flows is a challenge since the number of flows a switch can calculate per traffic flow is limited due to hardware components. It is essential to identifying the long lived flows as targets for which a value -DSCP - Differentiated services code point is assigned to a dedicated prototype switch using Open v Switch (OVS) and modifies the Explicit Congestion Notification (ECN) code point of target flows at the hardware level.

A unique DSCP value is assigned to theintended target flow and modifies the DSCP value if we find the target flow under certain conditions. Then the Congestion Experienced (CE) code point on the target will be discarded if the DSCP value matches. Incast will be hidden by autonomous network switch from theintended target flows. This method is widely used in most commercial switches.

Explicit Congestion Notification (ECN) is a TCP/IP extension that can notify the congestion to end nodes and ECN has greatly been used in Internet and in modern web servers. Shallow buffered switcheswith trade ASIC (Application Specific Integrated control) have been widely used in internet.

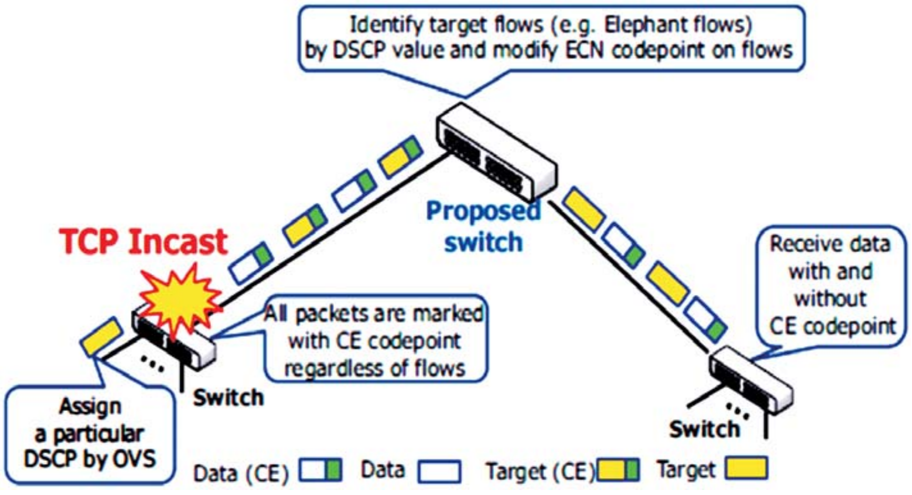
Identify target flows (e.g. Elephant flows) by DSCP value and modify ECN codepoint on flows
Proposed switch
TCP Incast
Receive data with and without CE codepoint
All packets are marked with CE codepoint regardless of flows
Switch
Switch
Assign a particular DSCP by OVS
Data (CE)
Data
Target (CE)
Target

**Figure 5**

## 4.1. Implementation

The Explicit Congestion Notification mechanism is slightly modified for target flows (Long-lived) in such a way that CE (Congestion Experienced) code point of packets belongs to intended target flows to hide the congestion from the receiver of such flows so that the sending rate of sender will not be reduced.

Target flow identification and congestion control mechanism are implemented on separate switches. Differentiated services code point (DSCP) has been implemented in all modern commercial switches. A particular DSCP value will be assigned for a target flow (Long lived) and modify the value under certain conditions. The CE (Congestion Experienced) code point on the target will be removed if the DSCP value matches. Incast will be hide out by autonomous network switch through the target flows.

## 4.2. Testing

To induce incast, the generation time of short lived and long lived flows are synchronized. Generally servers are inter connected by an Ethernet switch.

## 4.3. Model Test bed

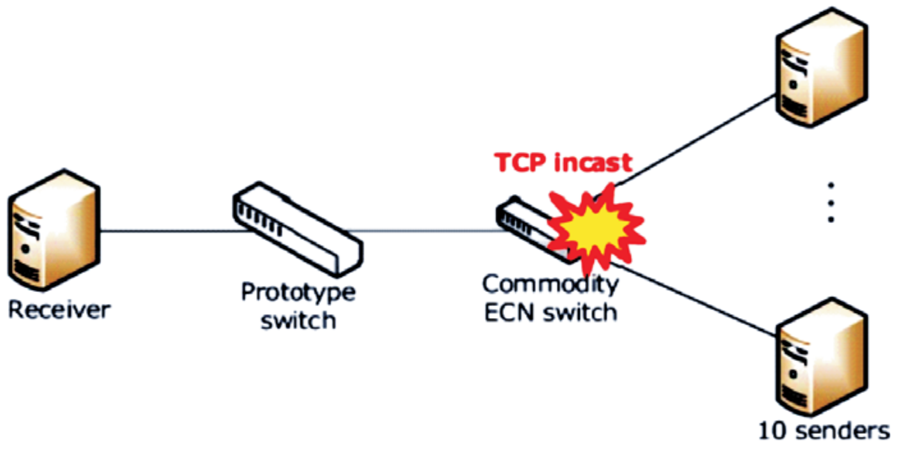

**Figure 6**

All sender nodes are connected to market available ECN switch and the prototype switch was connected between the ECN switch and the receiver. ECN function was enabled in ECN switch where TCP incast occurred.

### 4.4. Advantages

1. Throughput of multiple long lived flows is high when comparing to other techniques for TCP incast congestion control.
2. The implementation for flow aware congestion control never requires modification in the TCP stack though it uses ECN.
3. In SDN (Software Defined Networking) a central controller is used for network management, but it is difficult to rapidly handle the target flows from TCP incast by the centralized controller. Since incast is autonomously handled by network switch, this method overpowers the weakness of SDN approach.
4. ECN (Explicit Congestion notification) mechanism avoids unnecessary packet drops and provides high throughput in highly congested network.
5. In ECN, the congestion notification is accurately received by the TCP source and the connection doesn't remain idle, while waiting for a TCP retransmission timer to expire, after a packet has been discarded.

### 4.5. Limitation

1. A dedicated switch used to identify the target flow requires changes at the hardware level in datacenter networking and incur additional implementation cost and maintenance.
2. Inherent disadvantages of ECN is that, the ECN messages (Source Quench messages and TCP ACK packets with ECN field set ) may get dropped by the network before it reaches the TCP source.

## 5. PAC- PROACTIVE ACKNOWLEDGMENT CONTROL FOR TCP INCAST CONGESTION

In low latency, shallow buffered and High-bandwidth datacenter environments, TCP is more prone to incast congestion when more senders sending data synchronously to a receiver. TCP incast will create hundreds of milliseconds delay and high percentage of throughput degradation leads to affecting the application performance, especially those involving barrier-synchronized communications such as Map Reduce, Spark, Dryad and large scale web application.

In general the TCP incast mitigation techniques fall under two categories as window based solution and recovery based solution. In window based solution like DCTCP and ICTCP the congestion window will be adjusted to avoid overfilling switch memory buffer and packet losses. In recovery based solutions the RTOmin (Retransmission Time Out) focuses on minimizing the impact after packet loss. It is the properties of TCP that can retransmit lost packets immediately without leaving the link to get underutilized.

In PAC (Proactive Acknowledgment Control) TCP incast congestion is implemented at the receiver .The ACK is considered as not only the acknowledgement of received packets but also as a prompt for new packets. PAC proactively interrupts and releases ACK's in a way that the traffic fully utilizes the bottleneck link without creating TCP incast collapse. As TCP, a inbuilt self-clocking protocol it relies on the arrival of ACK packets to indicate that the network can accept more packets and maintaining uninterrupted transmission to fully utilize the link capacity. Proactive Acknowledgement Control uses the switch buffer memory size value as the initial standard threshold to modulate the ACK control pattern. The main idea of PAC algorithm is to wait to send back outstanding ACKs as long as the ACK triggered in-flight traffic doesn't exceed the threshold and will be able to optimized utilization of the bottleneck link without overloading the switch buffer.

### 5.1. Explanation of the concept

In the Available Switch buffer memory size is the initial threshold in-flight traffic volume to be zero, ACK queues to be empty, There will not be traffic in the network and no ACK required at the receiver.

Once PAC receiver receives new packets from the networktraffic, it will update the current traffic and the threshold value then the newly generated ACKs will be place in the ACK queue. When there is aremaining ACK in the ACK queues to be sent back then it has to check whether releasing the new ACK may cause the current traffic to exceed the threshold. In that condition PAC will hold the ACK till there is required network space to absorb the potential traffic triggered by the new ACK before sending it back.

PAC leverages ECN, used in many modern datacenter switches, to address the persistent congestion in core networks.

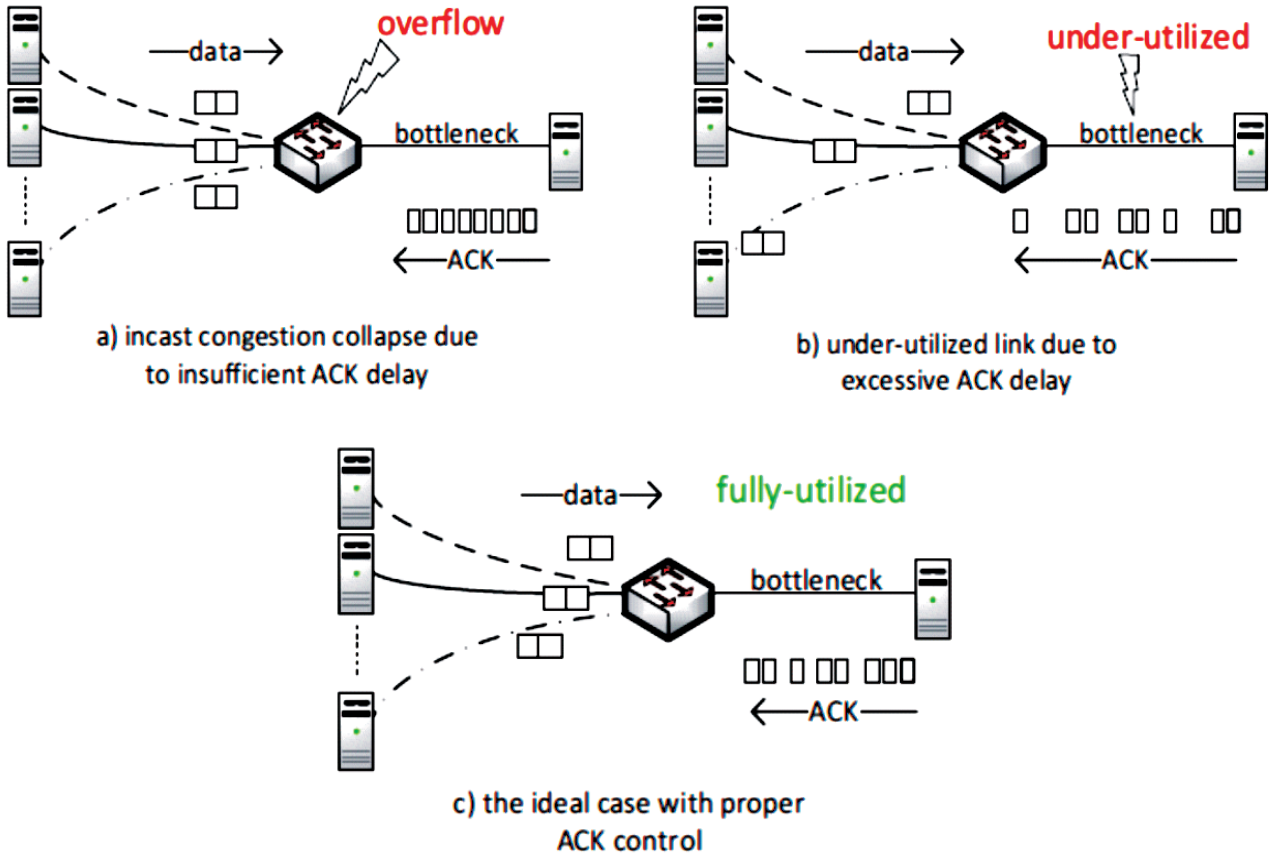
data
overflow
bottleneck
ACK
a) incast congestion collapse due to insufficient ACK delay
data
under-utilized
bottleneck
ACK
b) under-utilized link due to excessive ACK delay
data
fully-utilized
bottleneck
ACK
c) the ideal case with proper ACK control

**Figure 7**

### 5.2. Existing Solution to incast problem –Window Based solution Approach

The Key idea of this approach is to adjust the congestion window to control the current traffic so that it won’t over fill the switch buffer.

However the congestion window based designs are basically constrained because it can’t reduce the receive window size less than 1 MSS (Maximum Segment Size) and hence number of senders that will be in support is limited .Ex-DCTCP and ICTCP mitigation techniques.

### 5.3. Existing Solution to incast problem - Recovery Based solution Approach

Recovery based solution primarily focus on minimizing the impact after packet loss. TCP retransmission time out mechanism with high resolution timer of existing TCP characteristics is used to retransmit lost packets immediately without leaving the network link idle for very long time.

Though the recovery based solution handle larger number of senders to an extent but requires no major modifications on existing TCP/IP stack that makes very difficult to deploy in real production Datacenters.

### 5.4. Implementation

PAC is implemented as a NDIS -Network Driver Interface Specification through system programming, filter driver for Windows platforms and as an open SourceKernel module for Linux platform. The NDIS driver is located between the Network Interface card (NIC) driver and the TCP/IP protocol stack.

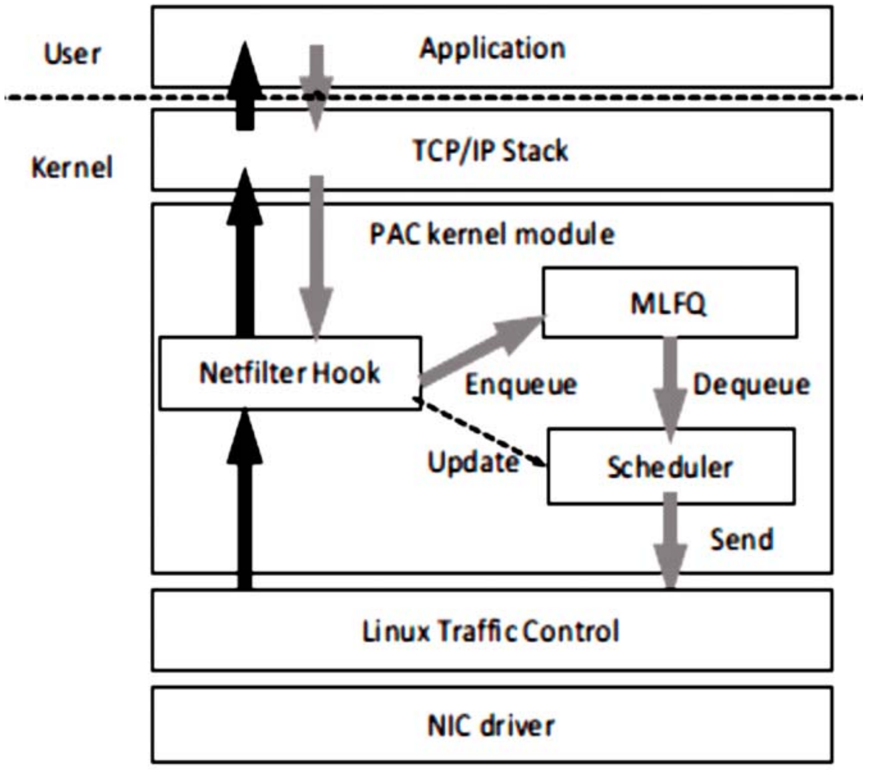

**Figure 8**

1. In virtualized environments PAC resides in Hypervisor.

#### *5.4.1. A hypervisor*

It is otherwise called as virtual machine manager, a program that makes multiple operating systems to share and utilize the single hardware host. Every operating system will behave as if it has the host'smemory, processor, and other resources all in itself. But, the hypervisor is actually controlling the host processor and resources, allocating the resources that are needed for each operating system and making sure that the virtual machines cannot disrupt each other.

## 5.5. Advantages

This method supports more number of senders (more than 1000) than its contemporary methods like DCTCP and ICTCP

1. It supports multiple platforms like windows and LINUX
2. It does not require any modification to TCP protocol stack
3. No additional hardware or dedicated switches required to implement this method.
4. Does not require any system modification to implement this method
5. Supports heterogeneous network traffic.
6. It does not affect the normal behavior of the TCP.
7. Readily deployable to production datacenters.

## 5.6. Limitations

1. It uses Switch buffer memory size as the initial threshold value to modulate the ACK control rhythm. Switches of various buffer size be deployed in Datacenter which again depends on ToR (Top of the Rack) and EoR (End of the Row) Datacenter design.
2. It leverages ECN -Explicit congestion Notification when severe congestion happened in network core. Inherent disadvantages of Explicit congestion notification is that, the ECN messages like Source Quench
3. Messages or TCP ACK packets with the ECN field set mightget dropped by the network before it reaching the TCP source.
4. Since placed as a NDIS for windows and implemented in kernel module, general system performance may get degraded.

## 6. MAINTAINING THE TCP INCAST CONGESTIONIN MODERN DATA CENTER NETWORKS - M21TCP-A

**Table 1**

| *Parameter* | *Default* |
|---|---|
| SUR size | 256 KB |
| Maximum Segment Size | 576 bytes |
| Link Bandwidth | 1 Gbps |
| Link delay | 25 us |
| TCP Variant | New Reno |
| Device Transmit Buffer Size | 128 KB |
| Retransmission Time Out (RTO) | 200ms |
| Switch Buffer Size | 64 KB |
| Limited Transmit | Disabled |
| Switch Queue | Droptail |

Many Datacenter applications are completely depends on TCP/IP protocol suite for reliable data transport. When it is true that it is successful in many internet applications but it does not integrate with and perform seamlessly in the datacenter networks. One of the important problems that comes in data center networks where multiple servers communicate concurrently and effectively in parallel with one single client through single switch.

A new variety of TCP called Many to one TCP is developed with the intention that the network manageable switch can be utilized to inform other parallel servers of how many other servers are communicating simultaneously. Each sender will limit its congestion window size based on the feedback from the manageable switch. This new approach is evaluated against Random Early Discard, Explicit congestion notification that are the most successful congestion control methods for incast mitigation.

## 6.1. Default Network Parameters

The partition aggregate workflow pattern consists of a multi layered hierarchical structure approach where the end line of computations and responses on one layer has impact on computations and response on its higher layer.

After each worker completes a task, the results will be sent back to the aggregator or parent node that compiles the results in to a final response.

Partition/Aggregate work flow designs are one of the “many to one communications” workload patterns that are affected by a communication problem commonly found in datacenteridentified as incast congestion.

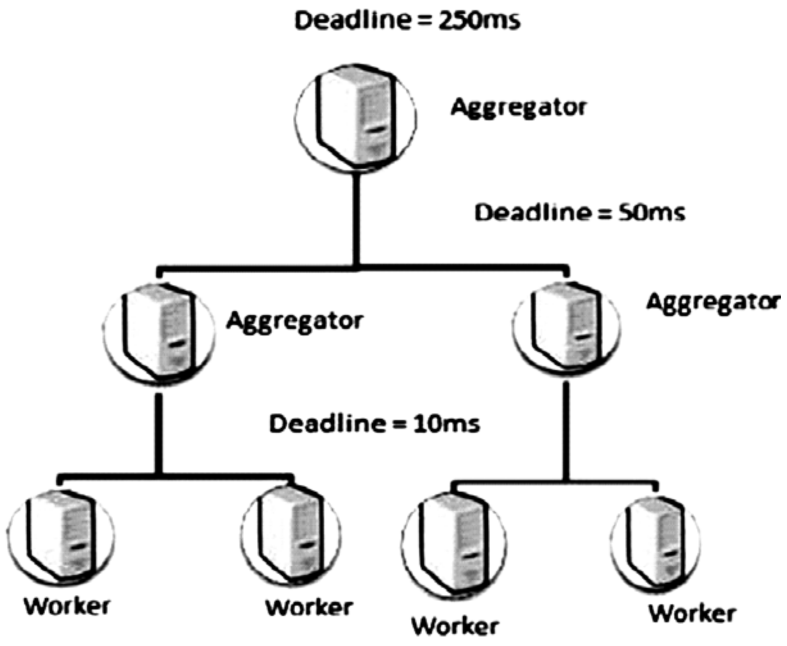

**Figure 9: Partition Aggregate Workflow**

## 7. CONCEPT

In M21TCP-A the switch informs each sender about  the total number of parallel senders by setting a TCP filed in every packet called “number of senders” that passes through it. The protocol leverages the idea of router based flow control concepts focused to arrest the root cause of incast: overflow of buffer memory size caused by high traffic. An additional filed will be added in the TCP /IP header about the number of senders in each flow that the switch allocates.

## 7.1. TCP Incast problem - Work load characterizations

In cluster based storage workloads where effectively parallel concurrent senders communicate with a client through bottleneck client where TCP incast congestion occurs in barrier synchronized many to one communication patterns like the Partition/aggregate pattern. The work load considered is motivated by distributed cluster based storage systems and bulk block transfers in batch processing systems like Map Reduce.

## 7.2. M21TCP Algorithm has three main components

### 7.2.1. Router /Switch Operation

Router/Switch that supports many to one TCP operation allocates a number of senders to each flow by counting the number of flows currently traversing the interface. This sender number is valid for the next Round Trip Time. Many to one router does this by keeping the list of all the different flows with varying flow parameters in its buffer memory and identifying new flows by comparing the incoming packets to the list of already achieved flow parameters.

### 7.2.2. Receiver Operations

Many to one TCP receiver sends the flow/sender number received in a packet back to the sender by encoding it in the ACK packet of the IP or TCP header. It can either encode the sender number information in an additional IP or TCP header filed. In the case of TCP’s properties like delayed ACKs the sender value in the latest received packet is used in the ACK field. The ACK that contains the number of senders encoded in an additional TCP header field is sent from the receiver to sender.

### 7.2.3. Sender Operation

The difference between normal TCP sender and the many to one TCP sender is that the Many to one sender should support an additional 32 bit TCP filed or an additional IP field. Both TCP and IP fields are used to obtain the result in M21TCP algorithm.

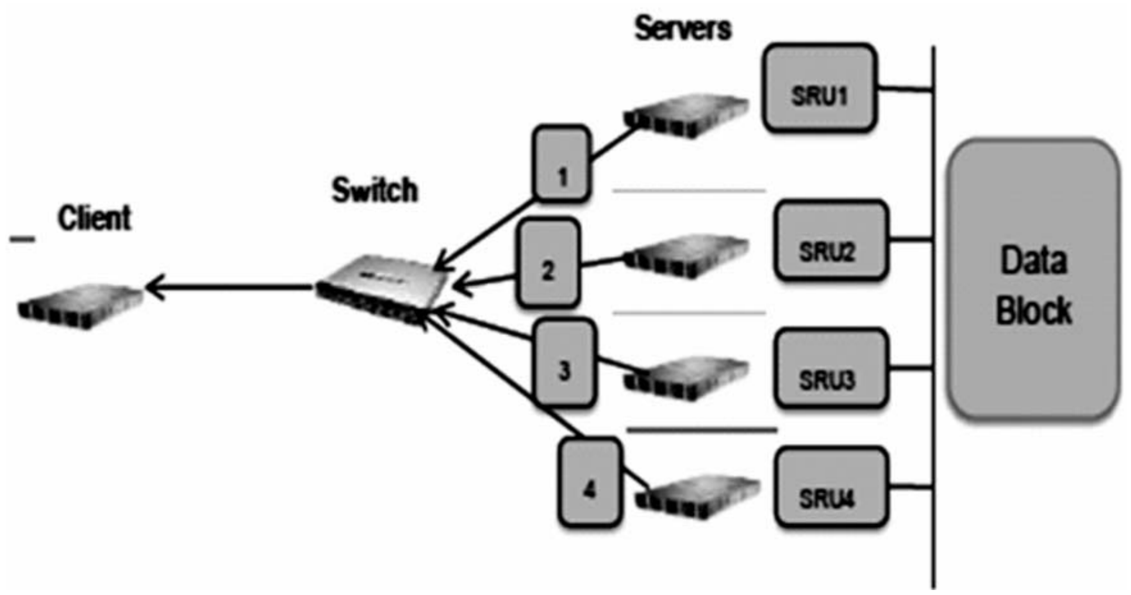

**Figure 10: The classical incast scenario showing multiple servers communicating with a single client through a bottleneck link**

### 7.2.4. Advantages

1. M21TCP -An algorithm utilizes the features of both network layer and transport layer and mitigating incast at granular by considering pack level data.
2. Effective use of TCP/IP Stack, both TCP and IP fields are used to obtain the results.
3. Leveraging the idea of router /Switch based flow control proposals
4. Sending server is used to calculate the Congestion window size.
5. This algorithm is compatible with any other congestion algorithm as the senders only set a maximum that the congestion window must not supersede.
6. Incoming request queue length or average queue length and no probabilistic methods are not considered on packet drops.

### 7.2.5. Limitations

1. Many one to TCP sender needs to support additional 32 bit TCP field or IP filed.
2. This algorithm designed for single path TCP. For network topologies with multiple paths between end hosts additional mechanism will be required.
3. May have additional burden in the router/Switch operations based on the number of senders

**Table 2**
**Comparison of prominent TCP mitigation techniques with key parameters**

| *TCP Incast Mitigation Techniques* | *PRIN Mechanism* | *SA-TCP* | *Flow Aware Congestion control* | *PAC- Proactive Acknowledgement Control* | *M21TCP-A Congestion Control* |
|---|---|---|---|---|---|
| OS Kernel Implementation & System Performance Degradation | Yes | No | No | Yes | No |
| Supports Multiple Hop Technology | Yes | No | No | Yes | No |
| Modification in TCP & IP Header | Yes | No | No | Yes | Yes |
| Number of senders Supports | 100 | >100 | >100 | Very High(1000) | 65 |
| Supports Both Linux and Windows | No | Yes | Yes | Yes | No |
| Switch operation Dependency | Yes | No | No | Yes | Yes |

## 8. CONCLUSION

Datacenters are becoming a cost effective infrastructure for deploying a various range of cloud application and for storing huge volume of data. The applications that are used in the data center networks are diverse in nature and has various performance requirements based on the need. Data center network flows operate using TCP as it is prime technology as it provides reliable and ordered bidirectional delivery of stream of bytes from one application to the other application residing on the same or two different machines with multi rooted tree topology. However when datacenter networks are using TCP it is unable to provide high throughput mainly due to the issue of TCP incast. In this paper, an in-depth survey of recently proposed prominent TCP incast mitigation techniques by highlighting the advantages and limitations have been presented.

## REFERENCES

[1] J. Zhang, F. Ren, and C. Lin,"Modeling and understanding TCP incast in data center networks", Jiao Zhang, Feng Yuan Ren, Member, IEEE, Li Tang, and Chuang Lin, Senior Member, IEEE Transactions on Parallel and Distributed Systems. VOL. 26, NO. 2, February 2015.Digital Object Identifier no. 10.1109/TPDS.2014.2310210

[2] PrasathSree Kumari, Jae-il Jung, Meejeong Lee "An Early Congestion Feedback and Rate Adjustment Schemes for Many-to-one Communication in Cloud-Based datacenter networks. Springer Science + Business Media New York 2015.

[3] Mhamdi L and Adesanmi, A Controlling TCP Incast Congestion in Datacenter Networks. In: IEEE International conference on communications (ICC).IEEE International Conference on Communications (ICC), 08-12 Jun 2015, London, UK

[4] Zhang.J.Ren.F.Tang.L.Lin.C: Modelling and solving TCP Incast problem in datacenter networks.IEEE Trans.Parallel Distribution, Syst 26(2) -2015

[5] Youngman Ren Jun Li1 , Guodong Wang, Lingling Li & Shanshan, SA-TCP –"To Mitigate TCP incast in Datacenter Works", 2014, Second international Conference on Advanced Cloud and Big Data Computer Network Information Center, Chinese Academy of Sciences, Beijing, 100190, China 978-1-4799-8085-7/14 2014 IEEE, DOI 10.1109/.54 163 10.1109/ CBD.2014.55

[6] Chunghan Lee, Yukihiro Nakagawa, Kazuki Hyoudou, Shinji Kobayashi, Osamu Shiraki, Takeshi Shimizu,"Flow-Aware Congestion Control to Improve Throughput under TCP Incast in Datacenter Networks" in Proceedings of the 2015 IEEE 39th Annual International Computers, Software & Applications Conference. 211-8588, Japan, 0730-3157/15 © 2015 IEEE DOI 10.1109/COMPSAC.2015.225.

[7] Wei Bai, Kai Chen, Haitao Wu, Wuwei Lan, Yangming Zha, "PAC: Taming TCP Incast Congestion using Proactive ACK Control", IEEE 22nd International Conference on Network Protocols, , The SING Lab, CSE Department,: Hong Kong University of Science and Technology, 978-1-4799-6204-4/14 2014 IEEE, DOI 10.1109/ICNP, 2014.

[8] Zheng F.Huang Y.Sun D Designing a new TCP based on FAST TCP for datacenter.In:2014, IEEE international Conference of Communications (ICC) pp, 3209-3214, 10-14 June 2014

[9] Adrian S.-W TAM Kang Xi Yang Xu Jonathan Chao,"Preventing TCP Incast Throughput Collapse at the Initiation, Continuation and termination" Department of Electrical and computer Engineering 978-1-4673-1298-1/12. IEEE 2012

[10] Akintomide Dasani Lofti Mhamdi, "Controlling TCP Incast Congestion in Datacenter Networks" School of Electric and Electronic Engineering, University of Leeds, Leeds UK. IEEE International Conference on Communications (ICC), 8-12 Jun2015

[11] Peng Zhang, Hongbo Wang, Shiduan Cheng state Key Laboratory of Networking and Switching Technology. "Shrinking MTU to Mitigate TCP Incast Throughput Collapse in Datacenter Networks" Beijing University of Posts Telecommunications Beijing, China. IEEE 978-0-7695-4357-4/2011/ CMC 2011.68

[12] Maxim Podlesny D.R Cheriton School of Computer Science, University of Waterloo, Carey Williamson, Solving the TCP -Incast Problem with Application -Level Scheduling University of Calgary.IEEE1526-7539/2012

[13] Amar Phanishayee, Elie Krevat, Vijay Vasudevan "Measurement and Analysis of TCP Throughput Collapse in Cluster-Based Storage Systems. Carnegie Mellon University, USENIX Association

[14] D.Nagle D.Serenyi D and Matthews A. The Panasa Active Scale Storage Cluster: Delivering Scalable High Bandwidth Storage. In SC'04: Proceedings of the 2004 ACM/IEEE Conference on Supercomputing.

[15] V. Vasudevan, A.Phanishayee H Shah,E,Krevat D Andersen G Ganger G.Gibson " Solving TCP Incast in Cluster Storage System" In proc , 7th USENIX conference on File and Storage Technologies. San Francisco 2009.

[16] "The Network Simulator"-Ns-2,"http://www.isi.edu/nsnam/ns/

[17] Luwei Cheng, Cho-Li Wang, Francis C.M.Lau "PVTCP: Towards Practical and Effective Congestion Control in Virtualized Datacenters. Department of Computer Science, University of Hong Kong.IEEE 978-1-4799-1270-4/2013

[18] Munir, A, Qazi I.A., Bin Qaisar S: On achieving low latency in datacenters: ln 2013 IEEE International conference on communications (ICC), June 2013.