クラウドコンピューティング基礎論

第11回

創造情報•小林克志

ikob@acm.org

Course Outline

- Administrivia
- Cloud computing
- Service reliability
- Scale-up / Scale-out
- Distributed data stores
- Global services
- Datacenter networkings (1)
- Datacenter networkings (2)
- Network performance
- User experiences
- Network latencies
- Advanced topics

Outline

- Administravia
- Homework review
- Advanced Topic

  Datacenter Networks
- NS2 again

  Simulate TCP incast

Final report (revised)

- Choose one network latency topic surveyed in:
  Briscoe, Bob, et al. "Reducing Internet Latency: a survey of techniques and their merits." IEEE Communications Surveys & Tutorials.

- Write review of **all of the cited papers** in the chosen topic, but excluding [202]
  The review must include both strong and weak points of the papers.

- Submit via course Web.
  - Submission due date is July 31.
    - **<u>If you will graduate on this August, submit it before July 20.</u>**
  - The review must be 2-4 pages long and written either in English or Japanese.

最終レポート(修正あり)

- 以下のサーベイ論文からネットワーク遅延に関するトピックを一つ選択せよ:

  Briscoe, Bob, et al. "Reducing Internet Latency: a survey of techniques and their merits." IEEE Communications Surveys & Tutorials.

- 選択したトピックで引用されている**全ての論文**をレビューすること。
  レビューには論文の良い点、悪い点両方を含むこと。

- 講義 Web ページから提出すること。
  - 締め切りは 7 月 31 日（火）
    - **今年 8 月卒業予定のものは、7/20 までに提出すること。**
  - レビューは 2-4ページにまとめ、英語あるいは日本語で記述すること。

Outline

- Administravia
- Homework review
- Advanced Topic

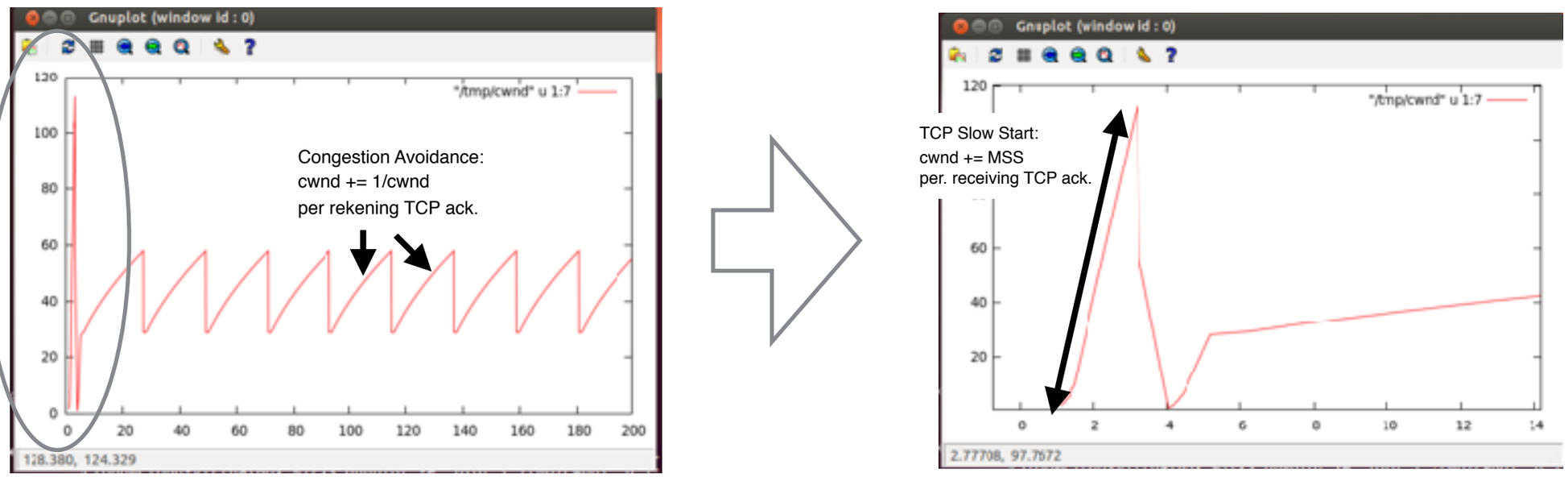
  Datacenter Networks
- NS2 again

  Simulate TCP incast

Today’s quiz1/2

- Goal: To understand slow-start and congestion avoidance behaviors.
- After download “ex1-sample.tcl” script from the course Web, Run it using “ns”
- Plot the dependency of ***cwnd*** on simulation time.
- Upload captured snapshot via the course web **during this class**.
- In case of using docker, you are recommended to run gnuplot on server side.

```
ns@ns-VirtualBox:~/Ex$ grep "1  0  5  0  cwnd" ex1-sample.tcp > /tmp/cwnd
ns@ns-VirtualBox:~/Ex$ gnuplot -e "plot \"/tmp/cwnd\" u 1:7 w l; pause -1"
```



Today’s quiz 2/2

- In ex1-sample.tcl, the queue size of the bottleneck link is bigger than optimal.
  How much queue size, C, is optimal in the number of pkts ?
  Note that the size of 1pkt is 1000Bytes.
    - The end-to-end throughput should be more than or equal to the bottleneck link capacity.
    - In terms of reducing latency, queue size should be small as possible.
- Plot the dependency of ***cwnd*** on simulated time.
- Upload captured snapshot via the course web **during this class**.

Optimal Queue size ?

- Too big queue size at bottle neck in ex1-sample.tcl
- How much is the Optimal Queue size, C ? (1000Bytes = 1pkt)
- C = BW x RTT = 0.5Mbps x 100ms = 0.5Mbps x 100ms / 1000Bytes/pkt = **<u>6.25 pkt</u>**

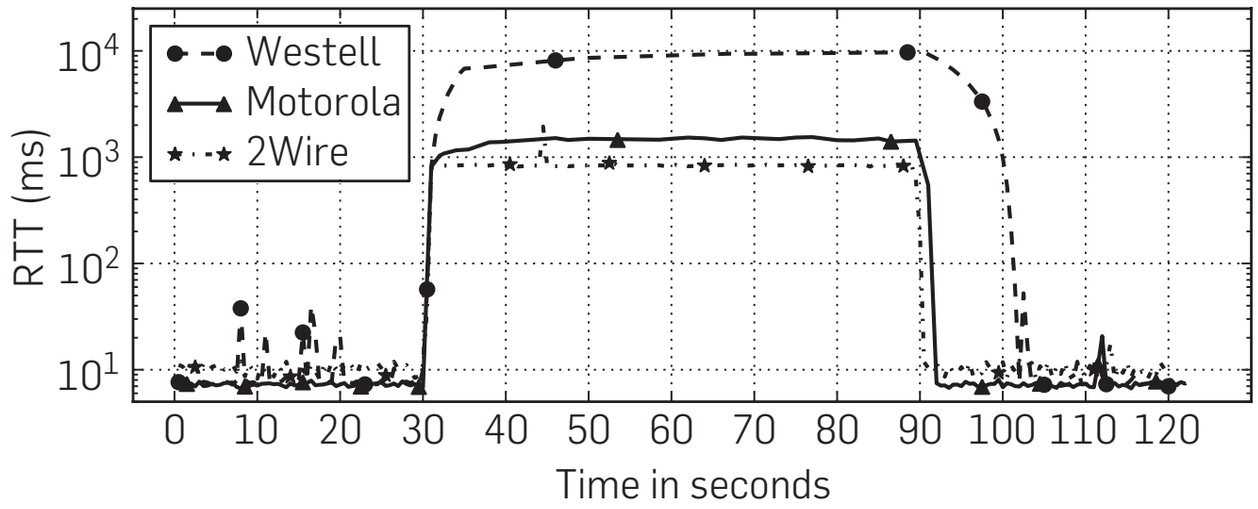
```
25 # Dumbbell topology
26 #
27 # s0              r0
28 #   \            /
29 #    \          /
30 #     n0------n1
31 #    /          \
32 #   /            \
33 # s1              r1
34 #
……..
41
42 $ns duplex-link $s0 $n0 10Mb 10ms DropTail
43 $ns duplex-link $s1 $n0 10Mb 10ms DropTail
44
45 $ns duplex-link $n0 $n1 0.5Mb 30ms DropTail
46 $ns duplex-link-op $n0 $n1 queuePos 0.5
47
48 $ns duplex-link $n1 $r0 10Mb 10ms DropTail
49 $ns duplex-link $n1 $r1 10Mb 10ms DropTail
50
……….
56
57 $ns queue-limit $s0 $n0 50
58 $ns queue-limit $s1 $n0 50
```

← Queue size definition at the bottleneck.

Bufferbloat

- Increase latency with growing packet buffer sizes on home routers.
    - Pointed out earlier days (RFC970: On Packet Switches With Infinite Storage)
    - Decrease cost of memory leads too-much packet buffers.
- Degrade UX of interactive applications
    - more than 10sec. latency in US CATV.
        - CATV：Limited upstream. Large file uploads suppress UX of interactive applications.
    - WiFi ：Dynamic rate adaptation and ARQ on MAC layer lead larger latency.

Figure 13. Different buffer sizes across modems lead to wide disparities in latencies when the upstream link is busy. (BISmark)



*Gettys, Jim, and Kathleen Nichols. "Bufferbloat: Dark buffers in the internet." *Queue* 9.11 (2011): 40.
**Sundaresan, Srikanth, et al. "Measuring home broadband performance." *Communications of the ACM* 55.11 (2012): 100-109.

Today’s assignment

- Add one more FTP application from node s0 to r0 into the sample script ex1-sample.tcl. (with optimal queue size)
  - Hint: Replace CBR application and UDP agent with FTP and TCP, respectively.
- Plot **cwnd** dependency of two TCP connections’ on the simulated time, and upload it.
- Submit via the course Web.

本日の課題

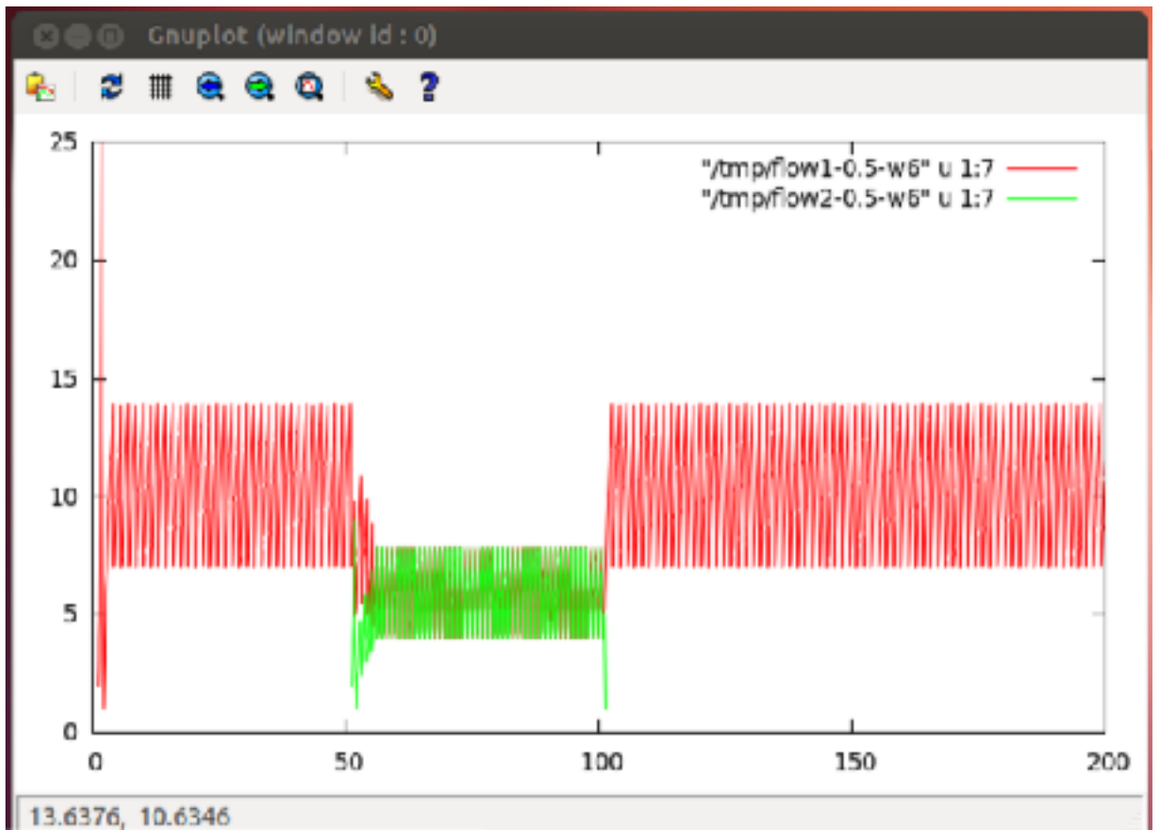
- サンプルスクリプト ex1-sample.tcl を変更し、s0 -> r0 でデータ転送するもう一つの FTP アプリケーションを追加せよ。
  （最適なキュー長を利用すること）
  - ヒント：CBR アプリケーションおよび UDP agent をそれぞれ FTP および TCP に置き換えれば良い。

- ２つの TCP コネクションの cwnd の時間変化をプロット、アップロードする。

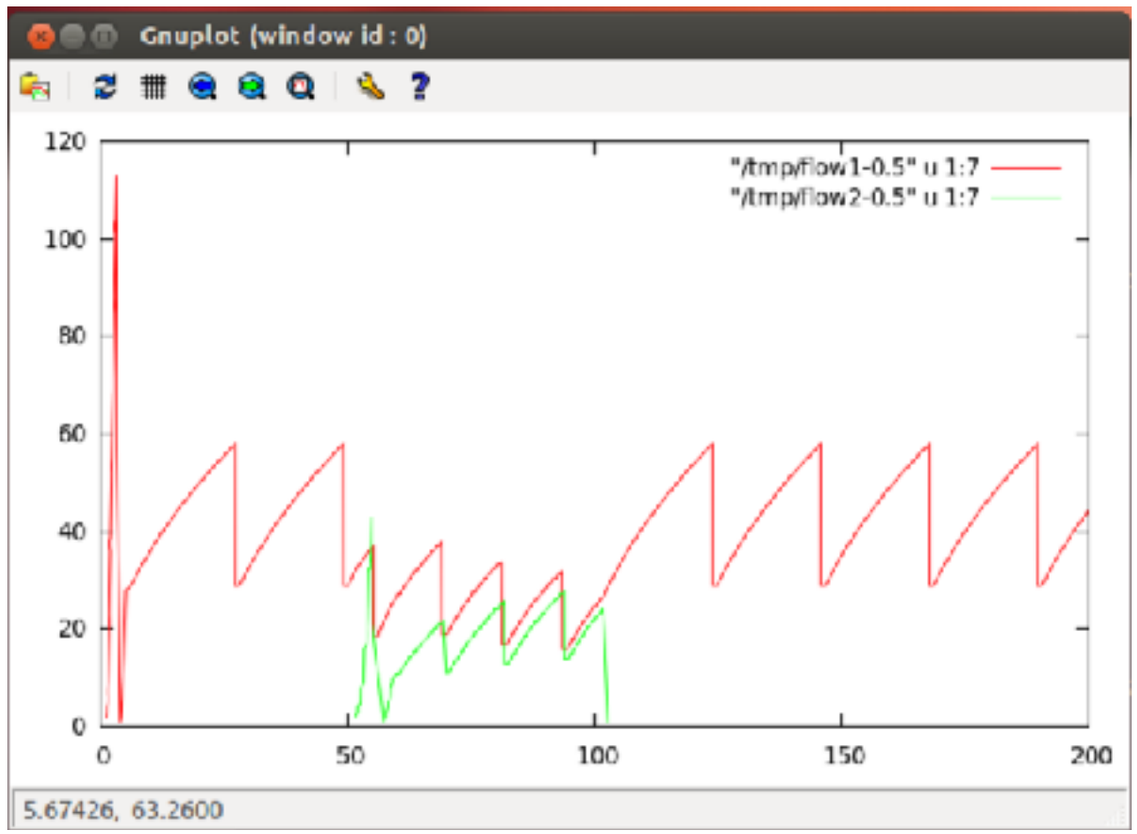
- 講義　Web から提出すること。

TCP flows share a bottleneck

- Add TCP flows instead of UDP



Bottleneck queue : Optimal



Bottleneck queue : Original

Outline

- Administravia
- Homework review
- Advanced Topic

  QUIC
- Analyzing TCP stream with Wireshark

Have you ever seen …

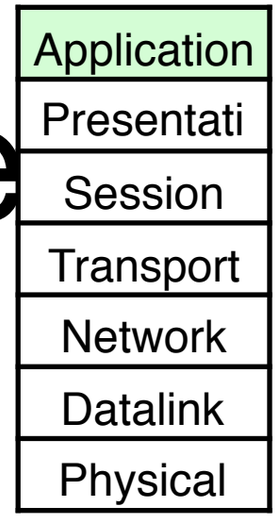
amazon.co.jp

何かお探しですか？

申し訳ございません。お探しのページは見つかりませんでした。

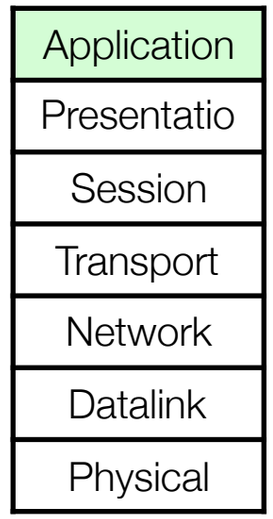
► ホームページへ戻る

Latency of network service caused by …

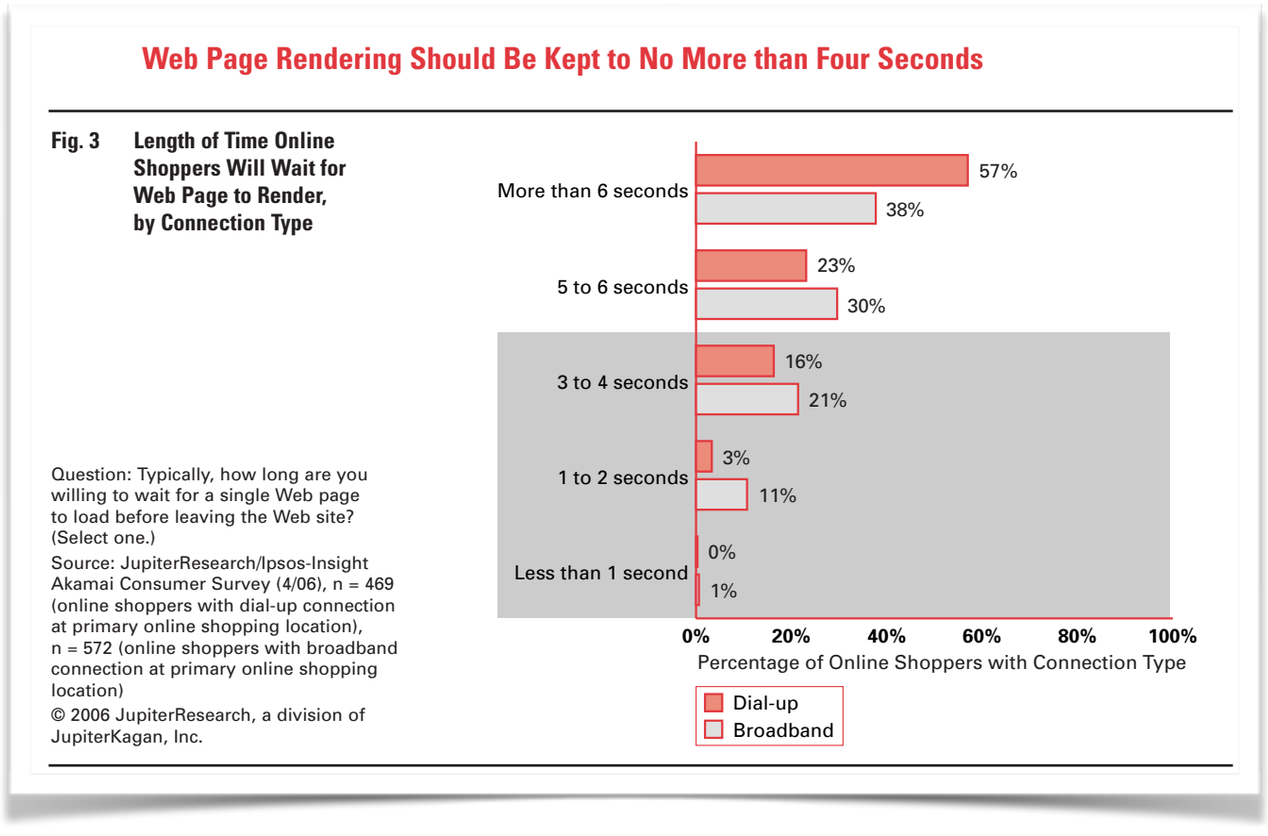


- Intra-node (server, client) :
  - Can be reduced by improving node performance and/or software, but not discussed in this class.
- Client - Server communication :
  - Global network, or the Internet
- Inter-node in server cluster :
  - Depending on communication in DC network

Four/Two second rule in Web services



- < 4 sec. render completion limit to keep customers' attention @ 2006
  - The limit is decreasing, e.g., 2 sec.@2013
- Poor satisfaction decreases customer loyalty and revisiting.
- To miss opportunity, and to lost revenue on e-Commerce.

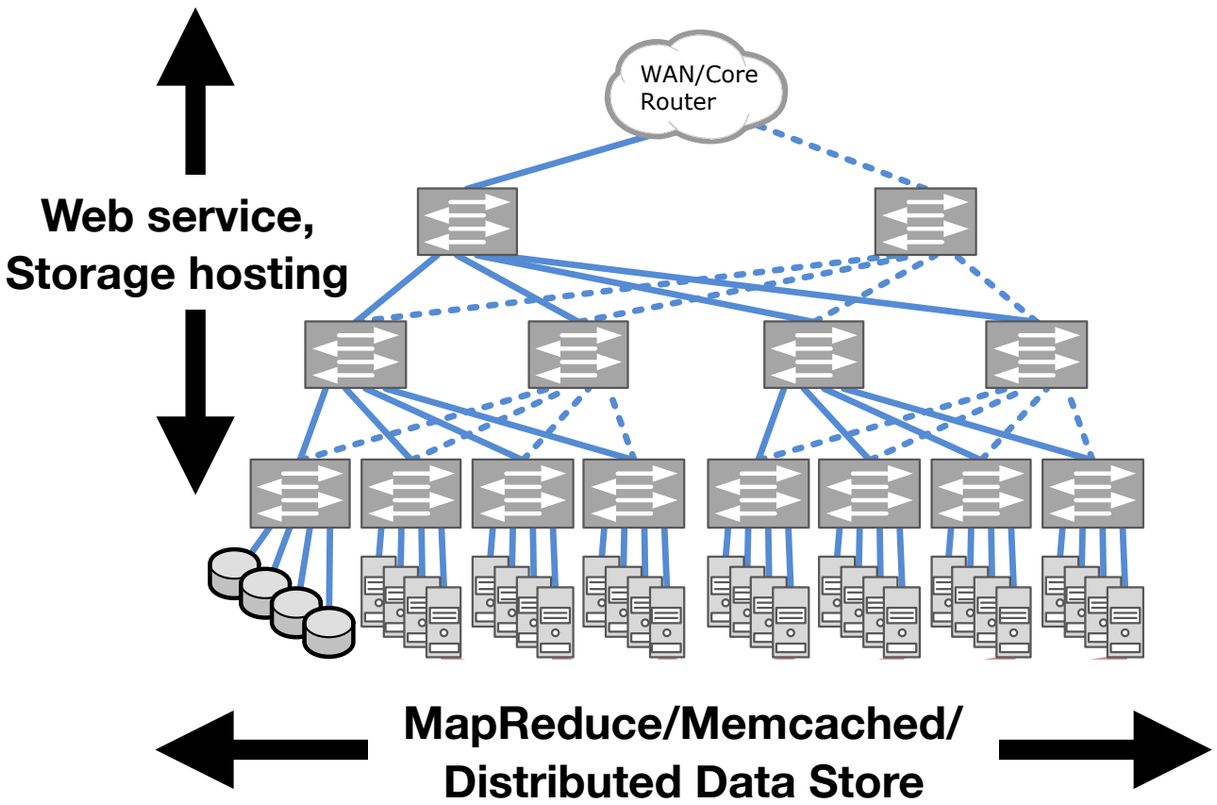


Jupiter Research, "RETAIL WEB SITE PERFORMANCE Consumer Reaction to a Poor Online Shopping Experience", 2006

Workloads in DC networ

Applicatio
Presentati
Session
Transport
Network
Datalink
Physical

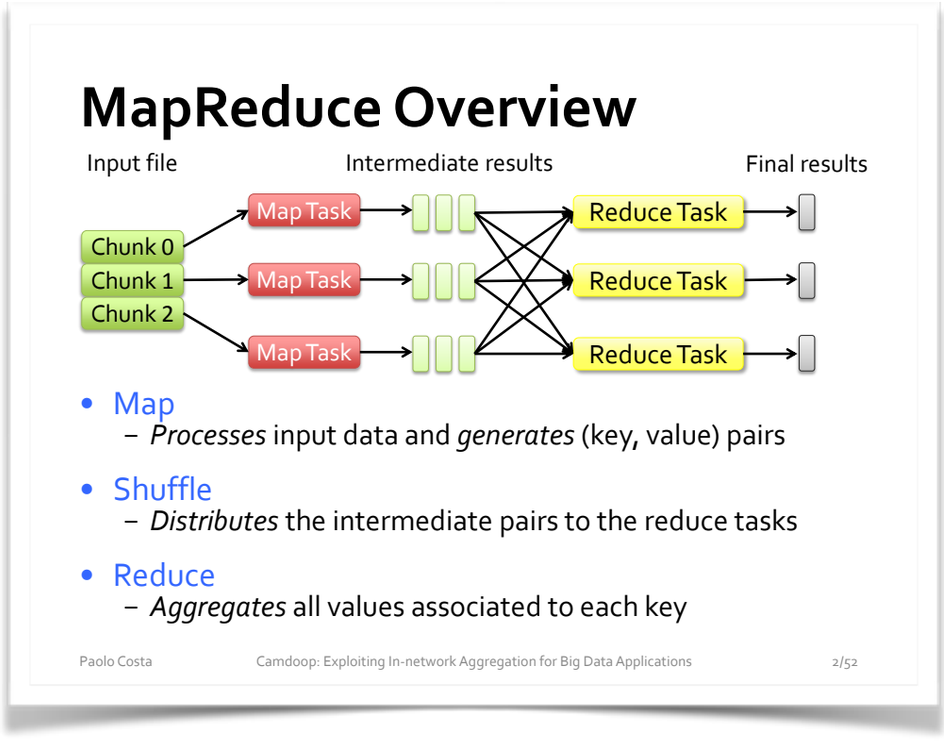
- Traffic Types:
  - External, or North-South:
    - Web services
    - Cloud storages
    - SQL DB
  - Internal, or East-West:
    - MapReduce
    - Memcached
    - Distributed Data Store.
- Workloads are increasing, and are shifting
  from : North-South
  to : East-West traffic.



PANDUIT, Flexible Modular Data Center Architecture Simplifies Operations, 2012

MapReduce

- Parallel and distributed processing framework
  - Implementation: Apache Hadoop by Java
    - Cloud API : Amazon Elastic MapReduce (EMR)
- Many similar ideas with different words, such as : Scatter/Gather (MPI), Partition/Aggregate (Microsoft), Divide/Conquer

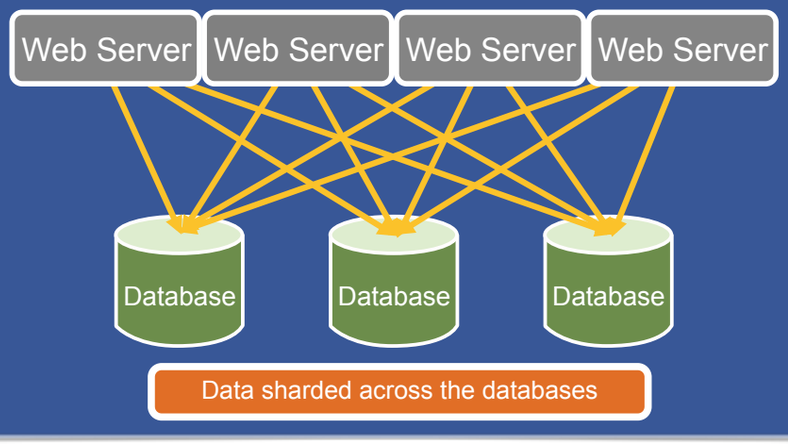


memcache

- Simple in memory cache for distributed systems used such in Facebook, Twitter.
  - key-value store using server memory only.
    - set(), get(), delete()…
    - If store is full, purge data, e.g., LRU (Least Recently Used) manner.

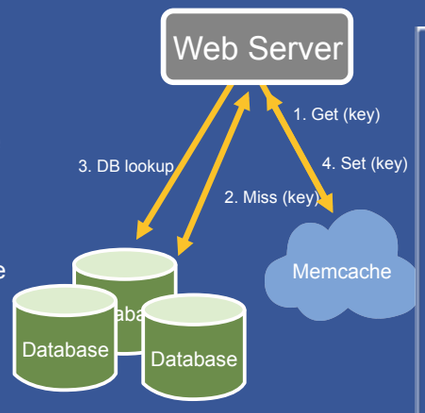
Pre-memcache

Just a few databases are enough to support the load



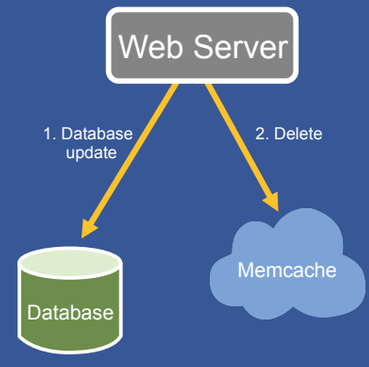
Need more read capacity

- Two orders of magnitude more reads than writes
- Solution: Deploy a few memcache hosts to handle the read capacity
- How do we store data?
  - Demand-filled look-aside cache
  - Common case is data is available in the cache



Handling updates

- Memcache needs to be invalidated after DB write
- Prefer deletes to sets
  - Idempotent
  - Demand filled
- Up to web application to specify which keys to invalidate after database update



Nishtala, Rajesh, et al. "Scaling Memcache at Facebook." Usenix NSDI. 2013.

DC network issues

- TCP incast
- Buffer build up
- Shared buffer

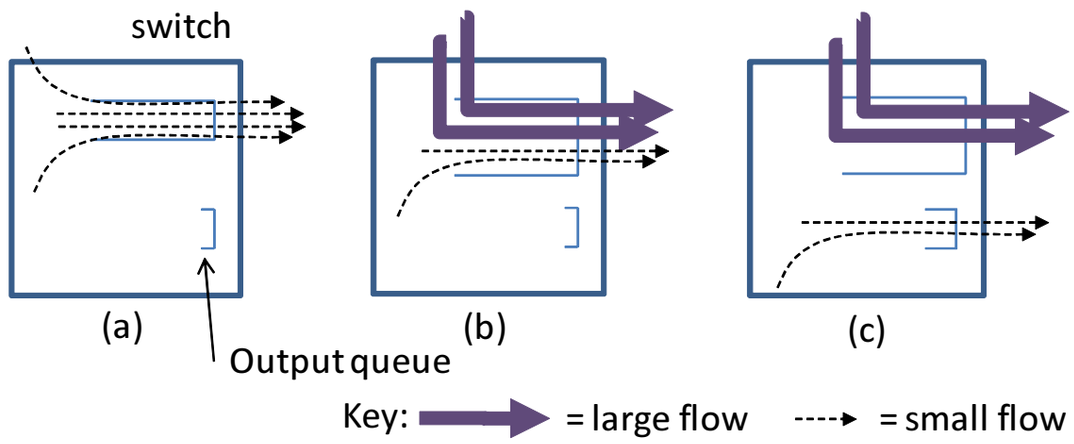
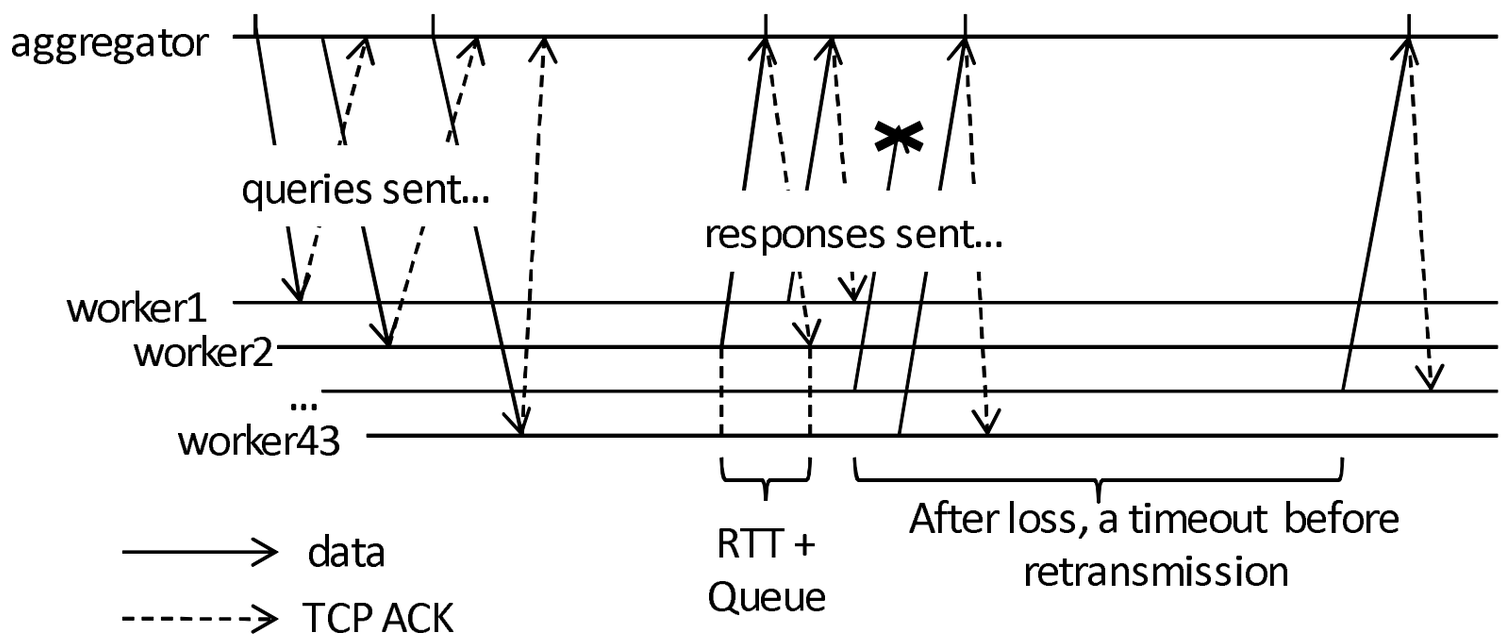


Figure 6: Three ways in which flows interact on a multi-ported switch that result in performance problems.

* Alizadeh, Mohammad, et al. "Data center tcp (dctcp)." *ACM SIGCOMM Computer Communication Review* 40.4 (2010): 63-74.

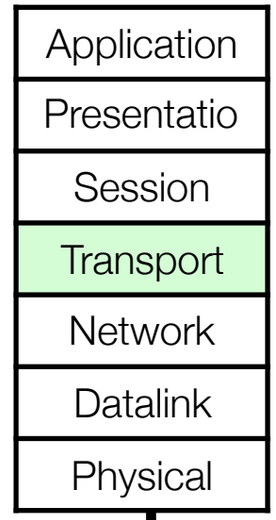
TCP incast

- On Partition/Aggregation type process
  - Request to / from N-servers.
- On-line service, e.g., SNS, imposes strict deadline even for such process.
  - Longer process worsen UX by longer waiting until render completion.
- Synched responses from many workers congest at aggregation points.
  - Even if short message sizes, longer completion time due to longer Retransmit Time Out (RTO), 200ms(min.). (RFC1122)



*Phanishayee, Amar, et al. "Measurement and Analysis of TCP Throughput Collapse in Cluster-based Storage Systems." FAST. Vol. 8. 2008.
**Chen, Yanpei, et al. "Understanding TCP incast throughput collapse in datacenter networks." Proceedings of the 1st ACM workshop on Research on enterprise networking. ACM, 2009.
***Nishtala, Rajesh, et al. "Scaling memcache at facebook." Proceedings of the 10th USENIX conference on Networked Systems Design and Implementation. USENIX Association, 2013.
Alizadeh, Mohammad, et al. "Data center tcp (dctcp)." *ACM SIGCOMM Computer Communication Review* 40.4 (2010): 63-74.

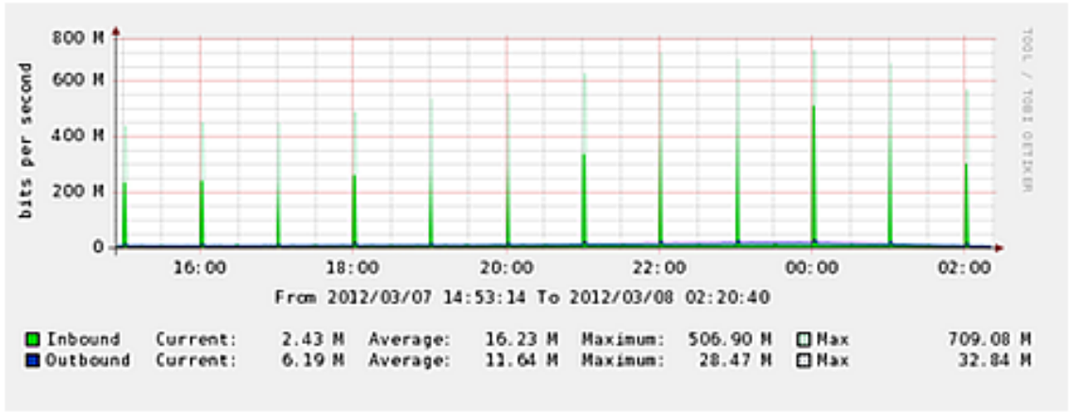
Loss detection and retransmission in TCP



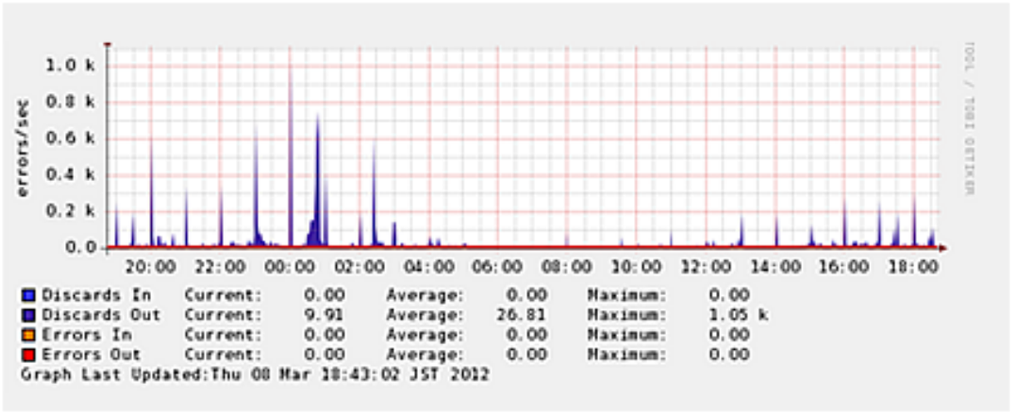
- Retransmit Time Out (RTO)
  When Data Ack. is not received until RTO*,
  - retransmit the segment that is regarded as loss
  - ssthresh = cwnd / 2, cwnd = min_cwnd.
  - RTO is derived by measured RTT.
    min_RTO :
    - 200msec on Linux default
    - ~~1~~05msec on Google DC intra-traffic

- Fast Retransmit / FastRecovery
  - If receiving three same ack., then the consecutive packet is considered as a loss.
    If retransmit is success,
    ssthresh = cwnd / 2, cwnd = ssthresh + 3 * MSS

2. Short Burst Traffic対策
   短期間に大量のトラフィックが発生するとShort Burst Trafficと呼ばれる現象が発生することがあります。Short Burst Trafficとは、短期間に大量のトラフィックが発生することにより通信許容量を超えてしまう現象です。
   下の図は実際にShort Burst Trafficが発生したときの、とあるL2スイッチの特定ポートのステータスです。この図では1Gbpsのインタフェースに対して700Mbpsの通信量でdiscard(破棄)が発生しているのがわかります。ここでは1Gbpsのキャパシティに対して700Mbpsと、まだ300Mbpsの帯域に余裕があるように見えても実際はL2スイッチのバッファキャパシティを超えている状況です。



700 Mbps



704 discard / sec

Short Burst Traffic が発生したときのL2スイッチポートの状況

一般的なL2スイッチは、受信したフレームを一旦バッファに入れてから少しづつ転送（スイッチング）する仕組みを取ります。上記の状況ではL2スイッチのバッファに収まりきらないほどの通信が瞬間的に大量に発生したため、バッファに収まらない分がdiscardとしてカウントされました。
この問題に対して我々は2つの対策を取りました。

1. バッファサイズが大きいL2スイッチへの入れ替え
2. Short Burst Trafficの制御が難しいサーバをネットワーク的に分離することで他のサーバに影響を及ぼさないようにする

How eliminates delays in TCP incast ?

- Reduce TCP minRTO
    - minRTO : 1s (RFC6298), 200ms (Linux).
    - Also such timeout processes rely on OS tick granularity, because it is scheduled by OS.
  - Pros: Only change server’s OS parameters, but not networks.
  - Cons: Another risk on global communications.
- Increase packet buffer of SWs
    - Pros: Only change network devices.
    - Cons: Additional delay, such as buffer buildup, buffer pressure.
- Eliminate burst responses by scheduling/spreading requests …
    - Pros: Not change infrastructures.
    - Cons: Modify applications.
- Modify TCP spec.
    - Tail Loss Probe

Ethernet SW with large buffer

S-Series S60 High-Performance Access Switch

The Force10 S-Series S60 is a high-performance 1/10 GbE access switch optimized for lowering operational costs at the network edge. The S60 answers the key challenges related to network congestion in data center ToR (Top-of-Rack) and service provider aggregation deployments. As the use of bursty applications and services continue to increase, huge spikes in network traffic that can cause network congestion and packet loss, also become more common. The S60 is equipped with the industry's largest packet buffer (1.25 GB) and the ability to "tune" buffering for specific application needs, enabling it to deliver lower application latency and maintain predictable network performance even when faced with significant spikes in network traffic. Providing 48 line-rate GbE ports and up to four optional 10 GbE uplinks in just 1-RU, the S60 conserves valuable rack space. Further, the S60 design delivers unmatched configuration flexibility, high reliability, and power and cooling efficiency to reduce costs.

Arista 7048T-A: 48-port 1GbE- 4-port 10GbE switch

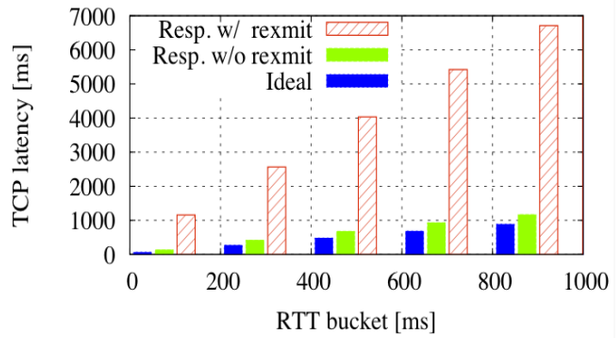
Deep Buffer Architecture

The Arista 7048 switches are designed to handle fair allocation of bandwidth to all traffic flows in the datacenter. With 768 MB of packet buffer memory, the architecture scales in congestion scenarios and all ports can simultaneously buffer up to 50ms of traffic without any drops. The deep buffer architecture of the 7048 also provides a platform for lossless asymmetric connections when 10GbE nodes communicate with 1GbE nodes in the datacenter.

TCP Tail Loss Probe (TLP)

Losses hurt Web latency

- Lossy responses last 10 times longer than lossless ones.
- 6.1% responses and 30% of TCP connections experience losses.



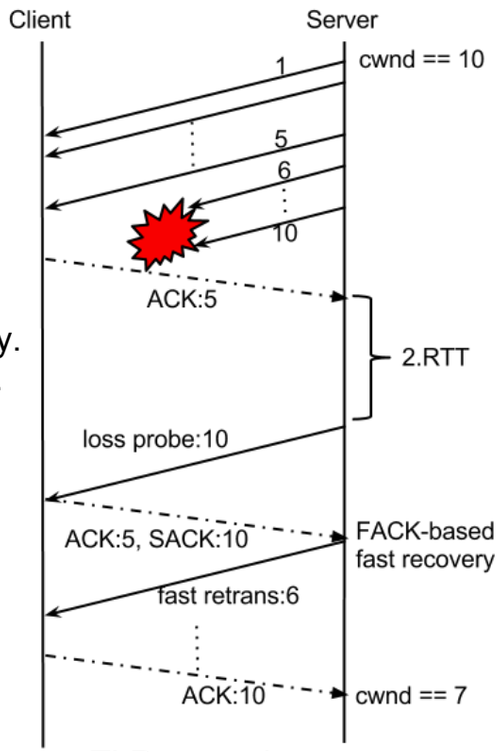
- Problem: timeouts are expensive for short flows
  - RTO is primary recovery mode for Web traffic
  - Normalized RTO values (#RTTs)

50%ile	75%ile	90%ile	95%ile	99%ile
5	12	29	54	214

Tail Loss Probe (TLP)

Key idea: convert RTOs to fast recovery.

- Transmit loss probe after approx. 2. RTT in absence of ACKs.
- Retransmit last packet (or new if available) to trigger fast recovery.



TLP example

TLP idea has been merged Recent Ack.(RACK) proposal as draft-cheng-tcpm-rack-02

Minimum RTO

Most TCPs have min RTO of 200 ms .. 1 sec; why?

- Delayed ACKs: typical delays: 40 ms .. 200 ms [400x RTT]
- RFC minimum RTO of 1 sec [10,000x RTT]
  - But switches don't have 1 s. of buffer; hosts don't delay ACKs by 1 s.

Google experience, incast research [1] [2] [3]: lower timeouts help app latency

- Quicker RTO, TLP: simple way to vastly reduce latency ~40x (200ms -> 5ms)
- Google uses 5ms internally since 2013 ([3] mentions 5ms as well)

Neal Cardwell et al., “TCP Options for Low Latency: Maximum ACK Delay and Microsecond Timestamps”, IETF 97

Buffer build up / Shared buffer

- Buffer build up
    - Deadline sensitive applications' packets, e.g. MapReduce, blocked by Long-lived flows packets, such as back-up, VM migration, occupying SW packet buffer.
        - Greedy TCP behavior, TCP designed to use network capacity in high efficiency.
- Shared Buffer
    - On some affordable SW, packet buffer shared by more than one SW ports. Not dedicated packet buffer for each port.
    - Flow blocking issue similar to buffer build up, even on different physical ports.

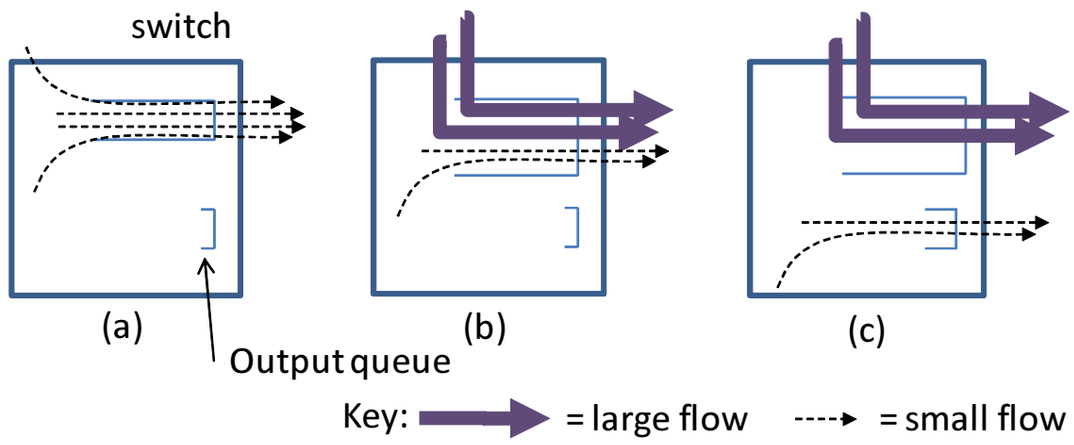


Figure 6: Three ways in which flows interact on a multi-ported switch that result in performance problems.

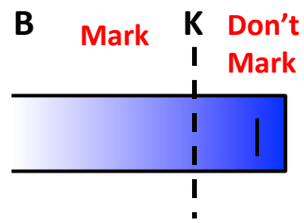
* Alizadeh, Mohammad, et al. "Data center tcp (dctcp)." *ACM SIGCOMM Computer Communication Review* 40.4 (2010): 63-74.

Data Center TCP (DCTCP)*

- Prevent the followings by maintaining SW queue at small size on DC network.
  - Buffer Build UP / Shared Buffer
- Congestion control taking advantages of ECN (Explicit Congestion Notification)
  - Windows Server 2012

Switch side:

– Mark packets when **Queue Length > K.**



Sender side:

– Maintain running average of ***fraction*** of packets marked **(α)**.

In each RTT:

$$F = \frac{\#\ of\ marked\ ACKs}{Total\ \#\ of\ ACKs} \qquad \alpha \leftarrow (1-g)\alpha + gF$$

➢ **Adaptive window decreases:** $Cwnd \leftarrow (1-\frac{\alpha}{2})Cwnd$

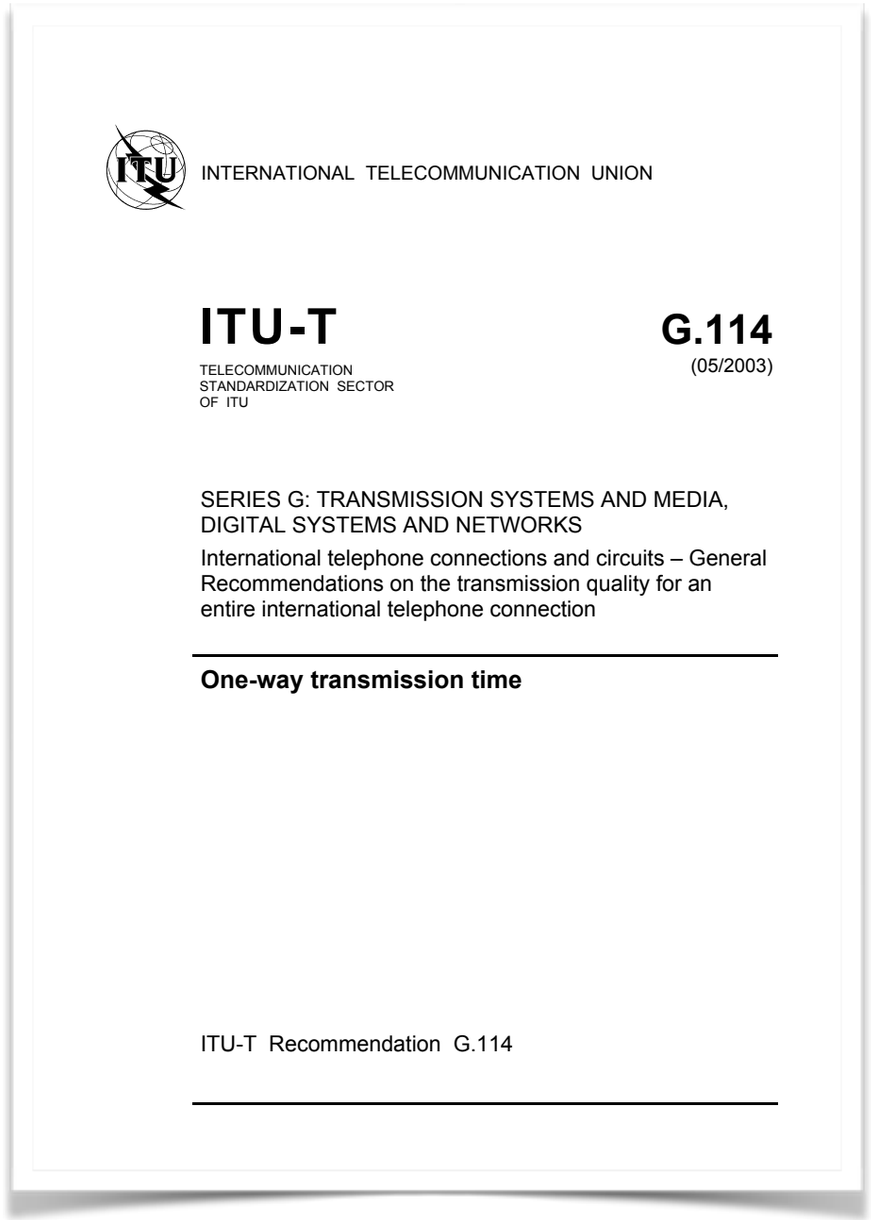
– Note: decrease factor between 1 and 2.

)." *ACM SIGCOMM Computer Communication Review* 40.4 (2010): 63-74.

Latency at Voice over IP (VoIP)

Application
Presentati
Session
Transport
Network
Datalink
Physical

- < 150msec. for interactive communication by ITU-T G.114.
  - One-way, Mouth-to-ear.
- VoIP Deployed 16M@2013 by NTT.
  - Prioritized class in on Diffserv CoS (Class of Service, RFC2475)
    - NGN (Next Generation Network)
    - NTT announces ISDN (INS) will terminated at 2025.



INTERNATIONAL TELECOMMUNICATION UNION

ITU-T

TELECOMMUNICATION STANDARDIZATION SECTOR OF ITU

G.114

(05/2003)

SERIES G: TRANSMISSION SYSTEMS AND MEDIA, DIGITAL SYSTEMS AND NETWORKS

International telephone connections and circuits – General Recommendations on the transmission quality for an entire international telephone connection

One-way transmission time

ITU-T Recommendation G.114

Outline

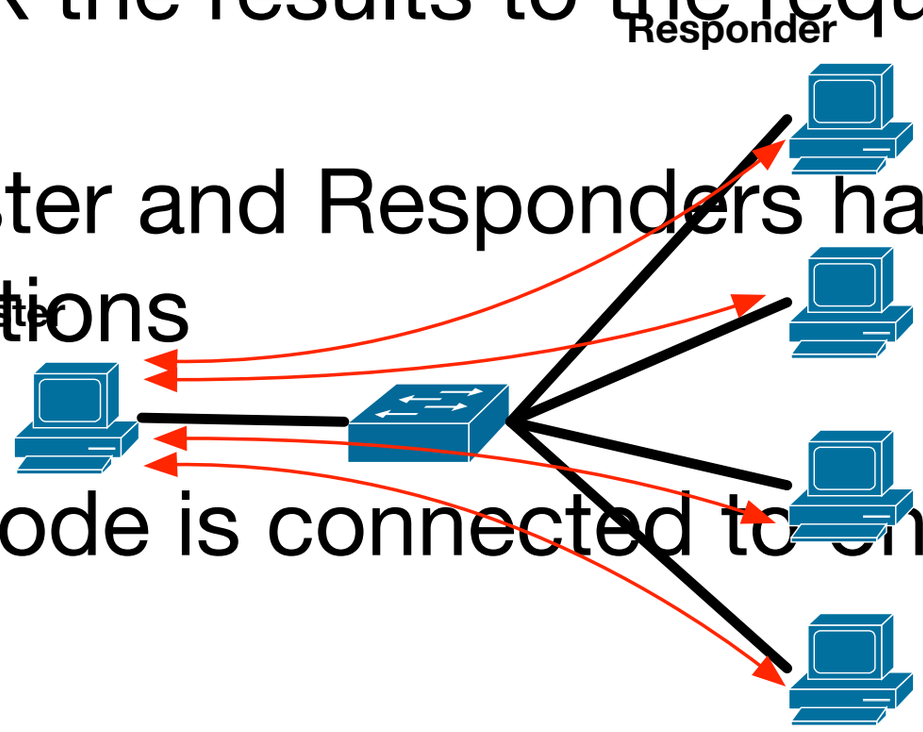
- Administravia
- Homework review
- Advanced Topic

  Datacenter Networks
- NS2 again

  Simulate TCP incast

TCP incast simulation on NS-2

- Scenario :
  - A requester sends a message to responders.
  - After completing the responders' processes, they send back the results to the requester.
    - Requester and Responders have TCP connections
    - Every node is connected to one switch.



ex3-sample.tcl

```
 1 set ns [new Simulator]
 2
 3 Agent/TCP/FullTcp set segsize_ 1460
 4 Agent/TCP/FullTcp set windowInit_ 10
 5 Agent/TCP/FullTcp set tcpTick_ 0.01
 6 set bw 1000Mb
 7 set nums 32
 8 set datasize [expr 1460 * 5]
 9 set reqsize 100
10 set bufsize 128
11 set rto 0.2
12
13 $ns color 1 red
```

TCP MSS : 1460Bytes
TCP initial Window : 10pkts
tcpTick_ (OS scheduler) : 10ms
Link Capacity 1Gbps
responder : 32 servers
Response Data Size : 5 x MSS = 7.3KB
Request Data Size : 100B
Buffer size of SW : 128pkts / port
TCP Retransmission Time Out(RTO)：0.2s

```
60 $ns color 48 violet
61
62 #Tracing
63 set fname ex3-sample
64 set fall [open $fname.tr w]
65 $ns trace-all $fall
66 set fnam [open $fname.nam w]
67 $ns namtrace-all $fnam
68
69 # post processing
70 proc finish {} {
71     global ns fall fnam
72     $ns flush-trace
73     close $fall
74     close $fnam
75     exit 0
76 }
77 # start-all
78 proc produces {size} {
79     global ns nums req res tcps reqsize
80     for {set i 0 } {$i < $nums} {incr i 1} {
81         $tcps($i) listen
82         $req($i) send $reqsize "$res($i) app-recv $i $reqsize"
83         $ns trace-annotate "[$ns now] Start Requester [set i]"
84     }
85 }
```

set trace file

procedure when finishing simulation

Request generator

ex3-sample.tcl

```
 86 #
 87 # application behavior
 88 #
 89 Application/TcpApp instproc stop {} {
 90     [$self agent] close
 91 }
 92 Class Application/MRReq -superclass Application/TcpApp
 93 Class Application/MRRes -superclass Application/TcpApp
 94 Application/MRRes instproc app-recv { node size } {
 95     global ns req datasize
 96     puts "[$ns now] 1 receives $size data from $node"
 97     $self send $datasize "$req($node) app-recv $node $datasize"
 98 }
 99 Application/MRReq instproc app-recv { node size } {
100     global ns
101     puts "[$ns now] requester receives $size data from $node"
102     $self stop
103 }
104 # topology
105 #
106 #    s(0)
107 #      \
108 #       \
109 #        \
110 #         \
111 # s(...)-- n(0)-----q(0)
112 #         /
113 #        /
114 #       /
115 #      /
116 #    s(N)
117 #
```

Close TCP connections

Define Requester, Responder app.

Responder: When receiving request, send back response immediately.

Requester: Do something, when receiving response.

ex3-sample.tcl

```
104 # topology
105 #
106 #    s(0)
107 #      \
108 #       \
109 #        \
110 #         \
111 # s(...)-- n(0)-----q(0)
112 #         /
113 #        /
114 #       /
115 #      /
116 #    s(N)
117 #
118 set n(0) [$ns node]
119 set q(0) [$ns node]
120
121 for {set i 0 } { $i < $nums } {incr i 1} {
122     set s($i) [$ns node]
123     $ns duplex-link $s($i) $n(0) $bw 25us DropTail
124     $ns queue-limit $s($i) $n(0) 1000
125     $ns queue-limit $n(0) $s($i) $bufsize
126 }
127
128 $ns duplex-link $n(0) $q(0) $bw 25us DropTail
129 $ns queue-limit $n(0) $q(0) $bufsize
130 $ns queue-limit $q(0) $n(0) 1000
131 $ns duplex-link-op $n(0) $q(0) queuePos 0.5
132
133 Agent/TCP/FullTcp instproc done {} {
134     global ns
135     puts "[$ns now] TCP proc done called"
136 }
```

← Topology definition

← Message when finishing job

ex3-sample.tcl

```
138 # Transport and Application
139 for {set i 0 } { $i < $nums } {incr i 1} {
140     set tcpq($i) [new Agent/TCP/FullTcp]
141     $tcpq($i) set class_ $i
142
143     $tcpq($i) set minrto_ $rto
144     $tcpq($i) set maxrto_ $rto
145     $ns attach-agent $q(0) $tcpq($i)
146
147     set tcps($i) [new Agent/TCP/FullTcp]
148     $tcps($i) set class_ $i
149
150     $tcps($i) set minrto_ $rto
151     $tcps($i) set maxrto_ $rto
152     $ns attach-agent $s($i) $tcps($i)
153
154     set req($i) [new Application/MRReq $tcpq($i)]
155     set res($i) [new Application/MRRes $tcps($i)]
156
157     $ns connect $tcpq($i) $tcps($i)
158     $req($i) connect $res($i)
159 }
160
161 #
162 # Scenario
163 #
164 $ns at 0 "produces $datasize"
165 $ns at 10.0 "finish"
166
167 $ns run
```

RTO definition

Events definitions

Today’s Class assignment

- Discuss approaches to reduce TCP incast delay.
  Modify ex3-sample.tcl simulation scenario as the followings.
  To reduce the time between request and response delay:

1. By changing the TCP Retransmission Time Out (RTO) parameters.
   The entire process must be no longer than 15ms.

2. By increasing the packet buffer size of the SW device.

- Note: RTO also depends on a OS tick interval.

- Upload the modified NS2 scripts and the simulation outputs on stdout.

本日のクラス課題

- TCP incast による遅延を抑制する２種類の方式を検討する。TCP incast のサンプルスクリプト、ex3-sample.tcl、を以下のシナリオで変更せよ。Request, Response に要する遅延を：

1.TCP 再送遅延時間 (RTO) パラメータの修正によって抑制する。処理完了までの時間は　15ms を超えてはならない。

2. SW のパケットバッファの増大によって抑制する。

- 修正した NS2 スクリプトと出力ログを示せ。

How eliminates delays in TCP incast ?

- **Reduce TCP minRTO**
    - **minRTO : 1s (RFC6298), 200ms (Linux).**
    - **Also such timeout processes rely on OS tick granularity, because it is scheduled by OS.**
  - Pros: Only change server's OS parameters, but not networks.
  - Cons: Another risk on global communications.
- **Increase packet buffer of SWs**
    - Pros: Only change network devices.
    - Cons: Additional delay, such as buffer buildup, buffer pressure.
- Eliminate burst responses by scheduling/spreading requests …
    - Pros: Not change infrastructures.
    - Cons: Modify applications.
- Modify TCP spec.
    - Tail Loss Probe