Data Center TCP
(DCTCP)

Mohammad Alizadeh, Albert Greenberg, David A. Maltz, Jitendra Padhye Parveen Patel, Balaji Prabhakar, Sudipta Sengupta, Murari Sridharan

Microsoft Research  Stanford University
Data Center Packet Transport

• Large purpose-built DCs
  – Huge investment: R&D, business

• Transport inside the DC
  – TCP rules (99.9% of traffic)

• How’s TCP doing?
TCP in the Data Center

• We’ll see TCP does not meet demands of apps.
  – Suffers from bursty packet drops, Incast [SIGCOMM ‘09], ...
  – Builds up large queues:
    ➢ Adds significant latency.
    ➢ Wastes precious buffers, esp. bad with shallow-buffered switches.

• Operators work around TCP problems.
  – Ad-hoc, inefficient, often expensive solutions
  – No solid understanding of consequences, tradeoffs
Roadmap

• What’s really going on?
  – Interviews with developers and operators
  – Analysis of applications
  – Switches: shallow-buffered vs deep-buffered
  – Measurements

• A systematic study of transport in Microsoft’s DCs
  – Identify impairments
  – Identify requirements

• Our solution: Data Center TCP
Case Study: Microsoft Bing

• Measurements from 6000 server production cluster

• Instrumentation passively collects logs
  – Application-level
  – Socket-level
  – Selected packet-level

• More than **150TB** of compressed data over a month
Partition/Aggregate Application Structure

- Time is money
  - Strict deadlines (SLAs)

- Missed deadline
  - Lower quality result

1. Art is a lie...
2. The chief...
3. ...

Deadline = 250ms
Deadline = 50ms
Deadline = 10ms

TLA
Deadline = 250ms
1. Art is a lie...
2. ...

Deadlines are useless. They can only give you answers.”

Worker Nodes
Generality of Partition/Aggregate

- The foundation for many large-scale web applications.
  - Web search, Social network composition, Ad selection, etc.

- Example: Facebook

Partition/Aggregate ~ Multiget
- Aggregators: Web Servers
- Workers: Memcached Servers
Workloads

• Partition/Aggregate (Query)

• Short messages [50KB-1MB] (Coordination, Control state)

• Large flows [1MB-50MB] (Data update)

→ Delay-sensitive

→ Delay-sensitive

→ Throughput-sensitive
Impairments

• Incast

• Queue Buildup

• Buffer Pressure
Incast

- Synchronized mice collide.  
  ➢ Caused by Partition/Aggregate.

Worker 1

Worker 2

Worker 3

Worker 4

Aggregator

RTO_{min} = 300 ms

TCP timeout
Incast Really Happens

- Requests are jipered over 10ms window.
- Jipering switched off around 8:30 am.

Jittering 99.9th percentile is being tracked. Percentiles.
Queue Buildup

- Big flows buildup queues.
  ➢ Increased latency for short flows.

- Measurements in Bing cluster
  ➢ For 90% packets: RTT < 1ms
  ➢ For 10% packets: 1ms < RTT < 15ms
Data Center Transport Requirements

1. High Burst Tolerance
   – Incast due to Partition/Aggregate is common.

2. Low Latency
   – Short flows, queries

3. High Throughput
   – Continuous data updates, large file transfers

The challenge is to achieve these three together.
Tension Between Requirements

High Throughput
High Burst Tolerance

Low Latency

Deep Buffers:
- Queuing Delays Increase Latency
- Reduced $RTO_{\text{min}}$ (SIGCOMM ‘09)
- Doesn’t Help Latency

Shallow Buffers:
- Bad for Burst & Throughput

Objective:
Low Queue Occupancy & High Throughput

AQM – RED:
- Avg Queue Not Fast Enough for Incast
The DCTCP Algorithm
Review: The TCP/ECN Control Loop

Sender 1

ECN Mark (1 bit)

ECN = Explicit Congestion Notification

Sender 2

Receiver

16
**Small Queues & TCP Throughput: The Buffer Sizing Story**

- Bandwidth-delay product rule of thumb:
  - A single flow needs $C \times RTT$ buffers for **100% Throughput**.
Small Queues & TCP Throughput: The Buffer Sizing Story

• Bandwidth-delay product rule of thumb:
  – A single flow needs $C \times RTT$ buffers for 100% Throughput.

• Appenzeller rule of thumb (SIGCOMM ’04):
  – Large # of flows: $C \times RTT / \sqrt{N}$ is enough.
Small Queues & TCP Throughput: The Buffer Sizing Story

• Bandwidth-delay product rule of thumb:
  – A single flow needs $C \times RTT$ buffers for 100% Throughput.

• Appenzeller rule of thumb (SIGCOMM ‘04):
  – Large # of flows: $C \times RTT/\sqrt{N}$ is enough.

• Can’t rely on stat-mux benefit in the DC.
  – Measurements show typically 1-2 big flows at each server, at most 4.
Small Queues & TCP Throughput: The Buffer Sizing Story

• Bandwidth-delay product rule of thumb:
  – A single flow needs $C \times RTT$ buffers for 100% Throughput.

• Appenzeller rule of thumb (SIGCOMM ‘04):
  – Large # of flows: $C \times RTT / \sqrt{N}$ is enough.

• Can’t rely on stat-mux benefit in the DC.
  – Measurements show typically 1-2 big flows at each server, at most 4.

Real Rule of Thumb:
Low Variance in Sending Rate $\Rightarrow$ Small Buffers Suffice
Two Key Ideas

1. React in proportion to the *extent* of congestion, not its *presence*.
   ✓ Reduces *variance* in sending rates, lowering queuing requirements.

<table>
<thead>
<tr>
<th>ECN Marks</th>
<th>TCP</th>
<th>DCTCP</th>
</tr>
</thead>
<tbody>
<tr>
<td>1011110111</td>
<td>Cut window by 50%</td>
<td>Cut window by 40%</td>
</tr>
<tr>
<td>000000000001</td>
<td>Cut window by 50%</td>
<td>Cut window by 5%</td>
</tr>
</tbody>
</table>

2. Mark based on *instantaneous* queue length.
   ✓ Fast feedback to better deal with bursts.
Data Center TCP Algorithm

Switch side:

- Mark packets when Queue Length > K.

Sender side:

- Maintain running average of fraction of packets marked ($\alpha$).

In each RTT:

$$F = \frac{\# \text{ of marked ACKs}}{\text{Total \# of ACKs}}$$

$$\alpha \leftarrow (1 - g)\alpha + gF$$

- Adaptive window decreases:

$$Cwnd \leftarrow (1 - \frac{\alpha}{2})Cwnd$$

- Note: decrease factor between 1 and 2.
DCTCP in Action

Setup: Win 7, Broadcom 1Gbps Switch
Scenario: 2 long-lived flows, K = 30KB
Why it Works

1. High Burst Tolerance
   ✓ Large buffer headroom → bursts fit.
   ✓ Aggressive marking → sources react before packets are dropped.

2. Low Latency
   ✓ Small buffer occupancies → low queuing delay.

3. High Throughput
   ✓ ECN averaging → smooth rate adjustments, low variance.
Analysis

• How low can DCTCP maintain queues without loss of throughput?
• How do we set the DCTCP parameters?

➢ Need to quantify queue size oscillations (Stability).
Analysis

• How low can DCTCP maintain queues without loss of throughput?
• How do we set the DCTCP parameters?

➤ Need to quantify queue size oscillations (Stability).

![Graph showing window size and time with formulas](image)

- Window Size
  - $W^* + 1$
  - $W^*$
  - $(W^* + 1)(1 - \alpha/2)$

- Time

- Packets sent in this RTT are marked.
Analysis

• How low can DCTCP maintain queues without loss of throughput?
• How do we set the DCTCP parameters?

➤ Need to quantify queue size oscillations (Stability).

\[ K > \frac{1}{7} C \times RTT \]

85% Less Buffer than TCP
Evaluation

• Implemented in Windows stack.
• Real hardware, **1Gbps and 10Gbps** experiments
  – 90 server testbed
  – Broadcom Triumph 48 1G ports – 4MB shared memory
  – Cisco Cat4948 48 1G ports – 16MB shared memory
  – Broadcom Scorpion 24 10G ports – 4MB shared memory
• Numerous micro-benchmarks
  – Throughput and Queue Length
  – Multi-hop
  – Queue Buildup
  – Buffer Pressure
  – Fairness and Convergence
  – Incast
  – Static vs Dynamic Buffer Mgmt
• Cluster traffic benchmark
Cluster Traffic Benchmark

• Emulate traffic within 1 Rack of Bing cluster
  – 45 1G servers, 10G server for external traffic

• Generate query, and background traffic
  – Flow sizes and arrival times follow distributions seen in Bing

• Metric:
  – Flow completion time for queries and background flows.

We use $\text{RTO}_{\text{min}} = 10\text{ms}$ for both TCP & DCTCP.
Baseline

Background Flows

Flow Completion Time (ms)

<table>
<thead>
<tr>
<th>Flow Size</th>
<th>DCTCP</th>
<th>TCP</th>
</tr>
</thead>
<tbody>
<tr>
<td>10-100KB</td>
<td>9</td>
<td>16</td>
</tr>
<tr>
<td>100KB-1MB</td>
<td>13</td>
<td>22</td>
</tr>
<tr>
<td>1-10MB</td>
<td>63</td>
<td>64</td>
</tr>
<tr>
<td>&gt;10MB</td>
<td>182</td>
<td>182</td>
</tr>
</tbody>
</table>

Query Flows

Query Completion Time (ms)

<table>
<thead>
<tr>
<th>Percentile</th>
<th>DCTCP</th>
<th>TCP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mean</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>95th</td>
<td>5</td>
<td>7</td>
</tr>
<tr>
<td>99th</td>
<td>19</td>
<td>28</td>
</tr>
<tr>
<td>99.9th</td>
<td>40</td>
<td>68</td>
</tr>
</tbody>
</table>
Low latency for short flows.
Low latency for short flows.
High throughput for long flows.
Baseline

- Low latency for short flows.
- High throughput for long flows.
- High burst tolerance for query flows.
Scaled Background & Query

10x Background, 10x Query

Completion Time (ms)

- DCTCP/ShallowBuf
- TCP/ShallowBuf
- TCP-RED/ShallowBuf
- TCP/DeepBuf

Short messages

Query
Conclusions

• DCTCP satisfies all our requirements for Data Center packet transport.
  ✓ Handles bursts well
  ✓ Keeps queuing delays low
  ✓ Achieves high throughput

• Features:
  ✓ Very simple change to TCP and a single switch parameter.
  ✓ Based on mechanisms already available in Silicon.