Extensions to FreeBSD Datacenter TCP for Incremental Deployment Support

Midori Kato
<katoon@sfc.wide.ad.jp>
DCTCP has been available since FreeBSD 11.0!!
FreeBSD DCTCP highlight

What is the average transmission time of 10 800KB transfers while 2 long flows are running in the same link?

- TCP (NewReno): 252.7 msec
- DCTCP: 82.5 msec
- One-sided DCTCP: 89.4 msec

Using BSD DCTCP achieves faster data transfer than using TCP not only in the fully deployed network but also in the partially deployed network.
Outline

1. DCTCP introduction
   - What is DCTCP?
   - What benefit can you receive by using DCTCP?
   - Necessary network equipment for DCTCP

2. FreeBSD DCTCP feature

3. How to configure DCTCP on FreeBSD
DCTCP (Datacenter TCP) [*]

- A proposed TCP variant that solves performance impairments in the data center network

**Traditional TCP problem**
- Concurrent interactive short flows show long data transmission time in a traffic mix of short flows and long flows
  - Short flows: database query and response
  - Long flows: bulk transfer for server migration

**DCTCP contribution**
1. Maintain low and predictable latency for short flows
2. Absorb traffic bursts
3. Maintain high throughput for long flows

### Data Center Network

**Data Center Network**

- **Servers**

**The Internet**

- **Internet**

<table>
<thead>
<tr>
<th></th>
<th>Data center network</th>
<th>The Internet</th>
</tr>
</thead>
<tbody>
<tr>
<td>Where is the destination host?</td>
<td>Single domain</td>
<td>Multiple domains</td>
</tr>
<tr>
<td>Sensitive to what?</td>
<td>Data transmission time</td>
<td>Throughput</td>
</tr>
</tbody>
</table>
Data Center Network

Feature of data center network
• Easy to optimize the network equipment and operation requirements of running services
• Short delay and no data re-transmission

<table>
<thead>
<tr>
<th></th>
<th>Data center network</th>
<th>The Internet</th>
</tr>
</thead>
<tbody>
<tr>
<td>Where is the destination host?</td>
<td>Single domain</td>
<td>Multiple domains</td>
</tr>
<tr>
<td>Sensitive to what?</td>
<td>Data transmission time</td>
<td>Throughput</td>
</tr>
</tbody>
</table>
DCTCP approach

- Leverage ECN (Explicit Congestion Notification) to the network
ECN (Explicit Congestion Notification)

- ECN is a traditional active queue management scheme
  - Provide auxiliary information for TCP congestion control

- ECN motivation
  - make hosts transfer data without packet losses

- Network equipments supporting ECN are needed
  - L3 switches/routers and servers
    - Check the configuration of network equipments
    - Many of servers unset ECN as default
### ECN mechanism

#### Traditional queuing management

- **Window size calculation with a packet loss**
  \[ W \leftarrow W / 2 \]

#### Queuing management with ECN

- **Window size calculation with ECN reception**
  \[ W \leftarrow W / 2 \]

- **L3 switch**
  - Set ECN when the queue length exceeds the threshold

- **Sending hosts**
  - Halve the window size by receiving a ECN packet
    - Window size: the total amount of data in byte sent by the host

- **ECN Echo**

- **Lost packet**

- **ECN mark**
**DCTCP mechanism**

- Use ECN for the precise estimation of the potential congestion
- **L3 switch**
  - Set ECN when the queue length exceeds the threshold
- **Sending hosts**
  - Calculate the window size based on the fraction of ECN in a previous window size

Window size calculation with ECN reception

\[ F \leftarrow \text{frac. Of ECNs in } W \]
\[ W \leftarrow W \times F / 2 \]
Testbed

- Hosts: four x86 machines
  - FreeBSD—CURRENT 10.0
  - Two dual-core Intel Xeon E5240 CPUs @ 3 GHz
  - 16GB RAM
  - Four ports Intel PRO/1000 1G Ethernet card
- L3 Switch: Cisco Nexus 3548
- Threshold of queue length: 10 packets
- Traffic generator: flowgrind [*]

Experiment

- **Incast**
  - Objective: evaluate the TCP/DCTCP performance of burst transfers
  - Scenarios: run 10 flows simultaneously
    - Transfer size: 10 - 800KB
  - Metric: avg. of data transmission time for short flows

- **Bulk transfer**
  - Objective: Evaluate the TCP/DCTCP performance by running a mix of short and long flows
  - Scenarios: starts 10 short flows 500 milliseconds after 2 long flows run
    - Transfer size of short flows: 10 - 800KB
    - Transfer size of long flows: 40MB
  - Metric: avg. of data transmission time for short and long flows
Result of Incast experiments

- DCTCP flows reduce 5 milliseconds of data transmission time at most
Result of bulk transfer experiments

- With background flows running, DCTCP short flows reduces
  - 10KB transfer time by 32.1 milliseconds
  - 800KB transfer time by 170.2 milliseconds
Summary of DCTCP

Q. What is DCTCP?
A. DCTCP is a proposed TCP variant using ECN for data center network

Q. What benefit can you receive by using DCTCP?
A. You can reduce 170 milliseconds of data transmission time for 800KB transfer in a mix traffic of short and long flows

Q. What are necessary network equipments for DCTCP?
A. L3 switch, routers and servers supporting ECN
BSD DCTCP = DCTCP + α

α means
1. Incremental deployment support
2. Initial window size calculation for performance tuning
Why is incremental deployment support important? [1/2]

Q. Do you recognize the difference between two example?
A. YES for us but NO for servers
Why is incremental deployment support important? [2/2]

Using DCTCP on a one-sided server is a possible situation in the network where appliance servers are running.
One-sided DCTCP Problem

(worst case) Trans. Time of one-sided DCTCP >> trans. Time of two-sided DCTCP

Solution

Support compatibility with ECN
- See our paper for detail

Result

One-sided DCTCP remarks 90% of similar performance to two-sided DCTCP
- Incast: +0.09 milliseconds
- Bulk transfer: +6.9 msec milliseconds
BSD DCTCP for initial window size calculation

- Selectable parameter considering the trade-off between latency and throughput

<table>
<thead>
<tr>
<th></th>
<th>slowstart = 0</th>
<th>slowstart = 1 (recommended[*])</th>
</tr>
</thead>
</table>
| Benefit | • Higher throughput | • Short latency during data transmission  
          |                       | • Friendly to competitive running flows |
| Flaws   | • Unfriendly to competitive running flows | • Longer data transmission time |

[*] Stephen Bensley, Lars Eggert, Dave Thaler  
“Microsoft’s Datacenter TCP (DCTCP): TCP Congestion Control for Datacenters”  
IETF, draft-bensley-tcpm-dctcp-03.txt
How to configure DCTCP on FreeBSD

1. Load DCTCP module
   # kldload cc_dctcp

2. Check available congestion control algorithms
   # sysctl net.inet.tcp.cc.available
      newreno, dctcp

3. Enable ECN
   # sysctl -w net.inet.tcp.ecn.enable=1

4. Enable DCTCP
   * Support for incremental DCTCP deployment is included
     # sysctl -w net.inet.tcp.cc.algorithm=dctcp

5. Tune DCTCP parameter (optional)
   - For faster data transfer (default)
     # sysctl -w net.inet.tcp.cc.dctcp.slowstart=0
   - For short latency
     # sysctl -w net.inet.tcp.cc.dctcp.slowstart=1
Summary of BSD DCTCP

BSD DCTCP = DCTCP + $\alpha$

- BSD DCTCP minimizes the performance penalty even in the partial use.
- BSD DCTCP has an optional configuration considering trade-off between latency and throughput.
Acknowledgement

Many thanks to

- BSD developers
  - Hiren Panchasara
  - Lawrence Stewart
  - Grenville Armitage
- KEIO University
  - Hideyuki Tokuda
  - Kenjiro Cho
  - Rod Van Meter
- NetApp Germany
  - Lars Eggert
  - Alexander Zimmermann
  - Richard Scheffenegger
- Cisco
  - Lucien Avramov
- IETF
  - Mirja Kuhlewind
  - Michio Honda