Recitation II: DCTCP
Plan

* Recitation Qs
* Background on DCs
* Queue game
* DCTCP

Logistics

* DPPR due 3/18
* Volunteers?
1. What is the goal of DCTCP?
   * Better net utilization in DC

2. How does DCTCP differ from TCP?
   * Management of buffers

3. Why does DCTCP differ from TCP?
   * Different setting
     * Control endpoints
     * Very low RTT
Why is this a great paper?

- Simple, important (?) problem

- Elegant solution
  - No new hardware
  - Simple changes to endpoints

- Works in practice (e.g. Cisco)
Types of Flows in Data Center

- Query traffic \( \sim 2 \text{ KB} \)
- Short message traffic \( \sim 1 \text{ MB} \) \( \Rightarrow \) Small, latency sensitive
- Background traffic \( \sim 50 \text{ MB} \) \( \Rightarrow \) big, throughput is important

E.g. Search

```
"Cheese shops"
```

Deadline = 100 ms

```
Cache
Language processor
Spell checker
Maps
```

Deadline = 50 ms

Deadline = 10 ms

```
DB_1
DB_2
DB_3
... DB_n
```

Updates
Why do they exist?

- If 10 FEs want to send to same DB at same time
  \[\text{in-rate} \gg \text{out rate} \Rightarrow \text{packets need to go somewhere}\]
- Memory is $$\Rightarrow \text{All ports share same queue buffer}

What happens when queue fills up?
- Dropped packets = worse search results (concurrency times out)

Even before queue fills up, can be problematic
- Latency grows?
Queue Game

Switch
(Amir)

Buffer

Sent

Server

- Students: Write S/L in buffer (if space) or in DROPPED bin (if no space)
- Amir: Move packet from buffer to sent box

Trial run: One sender, every 3-5 seconds

Show what this looks like in queue
For real now...

- Two flows, both send every 1-2 seconds

- One long flow, sending as quickly as possible
  - Some packets drop but throughput is good
  - How does TCP handle this?
    - Try again.

- One long flow, one short flow
  - Queue length grows \( \Rightarrow \) latency?
    - "better buffer"

- Everyone sends at same time
  - Lots of packets drop ("incastr")

\( \Rightarrow \) What could we do to fix this problem?

\( \Rightarrow \) What does DCTCP do to fix this problem?
Switch
As soon as queue grows beyond threshold, $k$
small
Switch sets flag on each packet $\text{SECN}$

Normal TCP: Wait until queue is full

Sender
Uses ECN flags to estimate whether queue size is $> k$

- If so, gradually back off (adjust window size)
- If not, continue as normal

Normal TCP: Cut window in half
Why would this not work on Internet?

- Practical: Need to modify both ends of the connection.

- Convergence time depends on RTT:
  - Small in DC (0.1 ms)
  - Big in Internet (50 ms)

  Time for new flow to get its fair share of bandwidth.
  
  3-4x slower than TCP

- Feedback is too slow... by the time sender gets ACK, queue may be empty.
  - Instantaneous queue length not a good signal (e.g., traffic on road 8 hrs away)

- Less clarity about types of Slow (?)