Less is More: Trading a little Bandwidth for Ultra-Low Latency

Mohammad Alizadeh, Abdul Kabbani, Tom Edsall, Balaji Prabhakar, Amin Vahdat,Masato Yasuda

Past Beliefs

- Network goodness is measured in bandwidth
  - Circuit to packet switching
  - TCP
  - Bandwidth Provisioning

- This is good for throughput-oriented applications

Motivation

- Latency-sensitive applications should not be ignored
  - High frequency trading
  - High performance computing
  - Search
- How can latency-oriented and throughput-oriented apps share the network?

Where is Latency an issue?

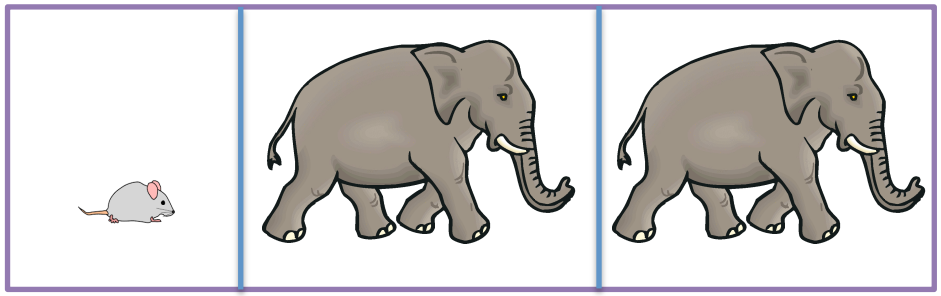
- NIC and End Hosts
  - Kernel bypass
  - Zero Copy
- Switches: Queuing Delays are still an issue

HULL: High Bandwidth Utra-Low Latency

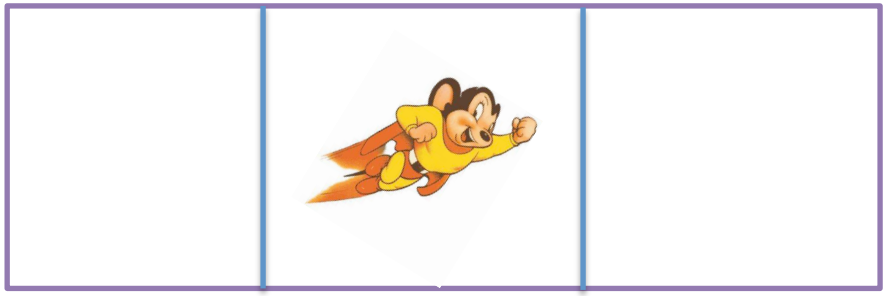
Solution: Predict queue occupancy and prevent congestion

Result: empty queues

Queue Without HULL



Queue With HULL



How do we get empty queues?

1. DCTCP: Flexible response to congestion
2. Phantom Queue: Predict Congestion before it happens
3. Packet Pacing: Control burstiness

DCTCP

- Set ECN Marking threshold at switch queue
- Back off is now proportional to fraction of marked packets

% of Packets Marked	TCP	DCTCP
10%	Cuts packets sent by 50%	Cuts packets sent by 5%
60%	Cuts packets sent by 50%	Cuts packets sent by 30%

DCTCP is Better, but Insufficient

Good

- Reduces fluctuation in throughput
  - 94% vs. 75% average throughput
- This reduces latency from 10 ms to 100us

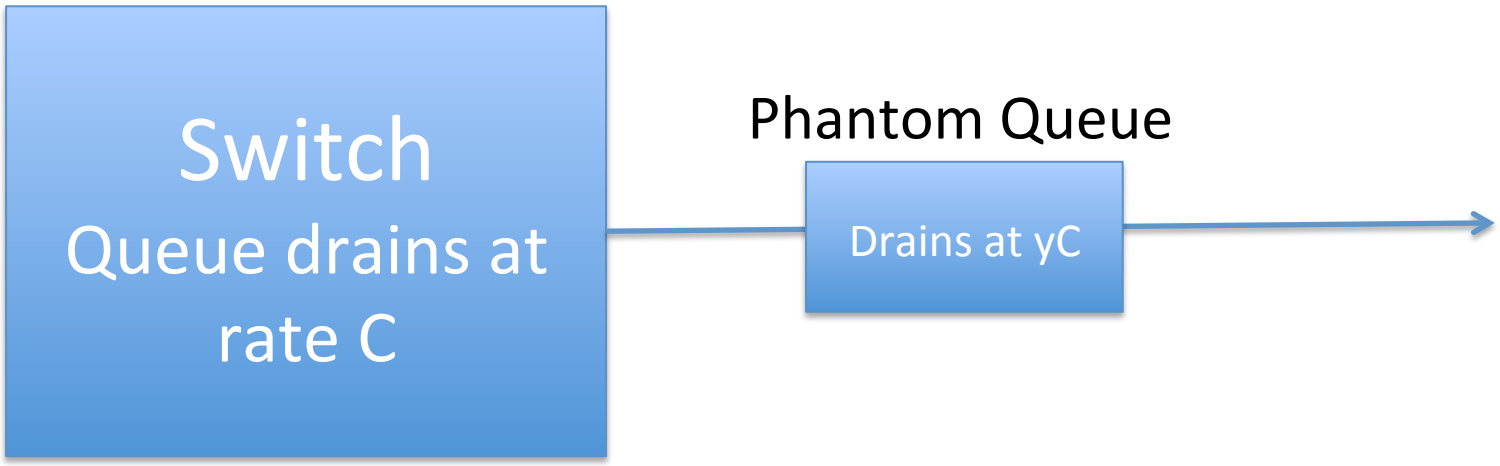
Insufficient

- For our applications, we want to approach 0us latency

Issue: Detects congestion already it is already happens

Phantom Queues

- What we want: signal congestion before it occurs
- How: Keep track of the rate of a switch's queue drains

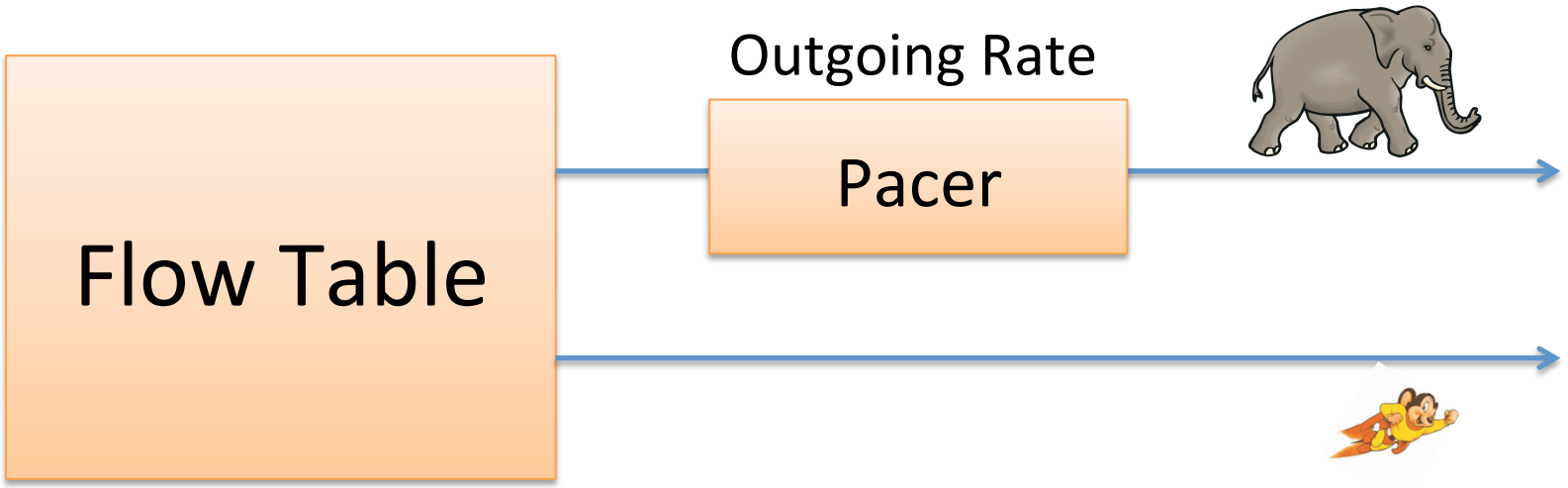


Still not good enough...

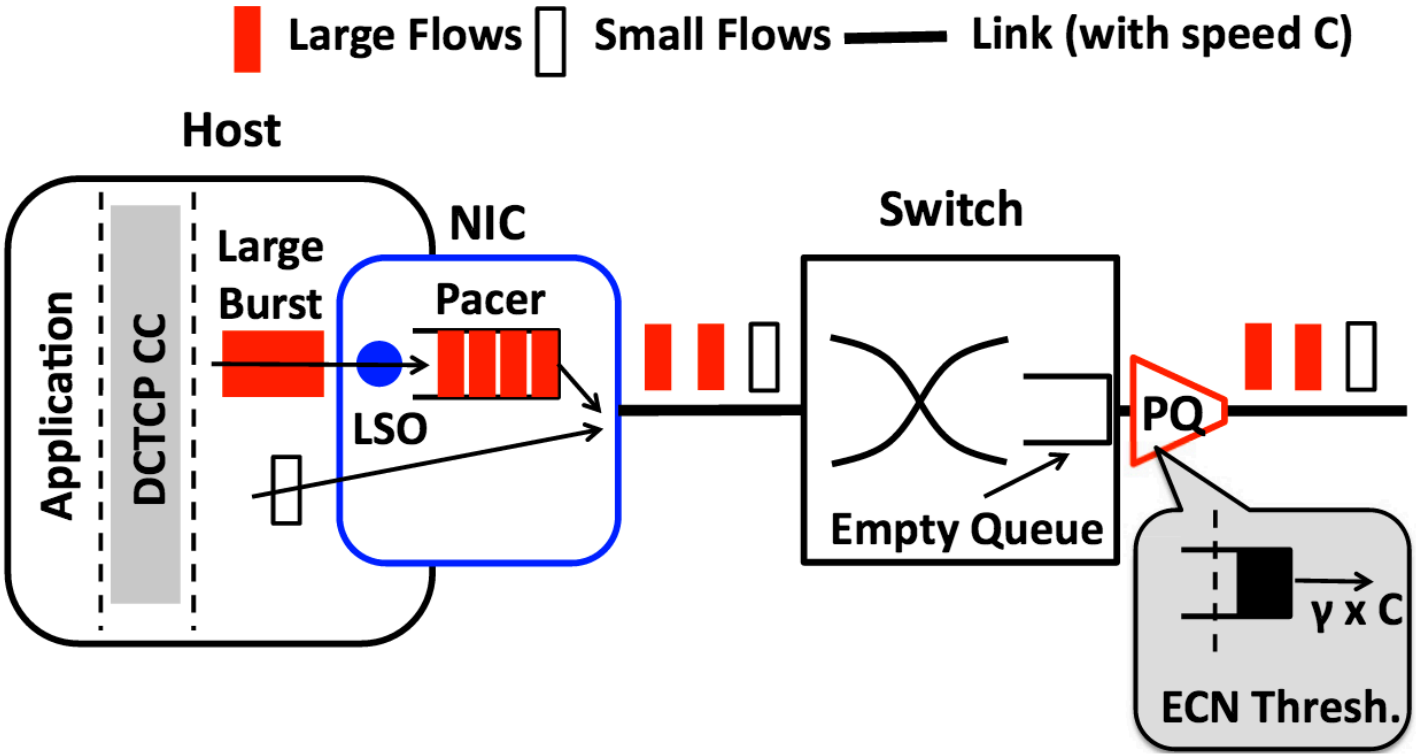
- Issue: Bursty traffic
  - Slow Start
  - NIC optimizations to reduce CPU utilization
- Why this is bad: Queue can still get congested
  - Phantom queue is marking all packets with ECN!

Packet Pacing

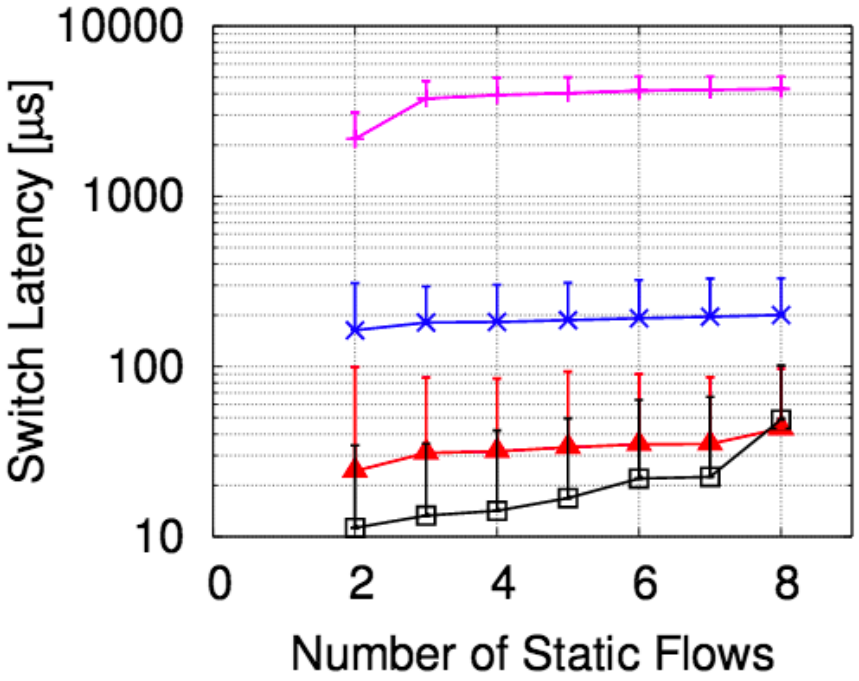
- Pace packets from large flows at the host
- Determine a rate R in which to emit packets

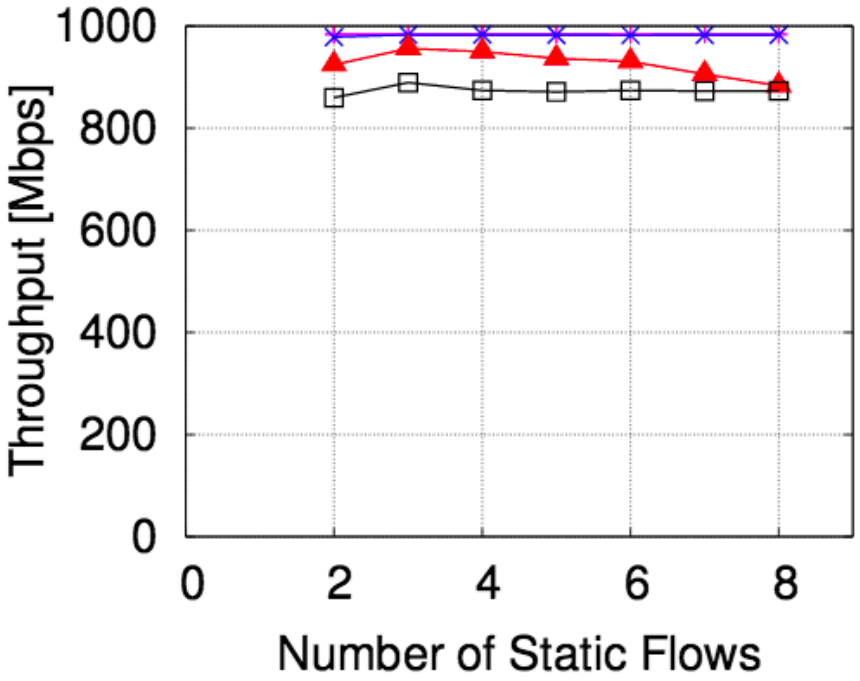


Putting it all Together



Results


Switch Latency [µs]
10000
1000
100
10
0
2
4
6
8
Number of Static Flows


Throughput [Mbps]
1000
800
600
400
200
0
0
2
4
6
8
Number of Static Flows


TCP-Droptail
DCTCP-30K
DCTCP-6K-Pacer
DCTCP-PQ950-Pacer

What is the Innovation Here?

- AQM utilized Virtual Queues to predict congestion
- Software pacing to control burstiness
- DCTCP is not new

Tradeoff between BW and Latency

- The usefulness of this design assumes this trade off is fundamental

...Is it?

What about TCP-QoS

- Already can provide ultra-low latencies and better throughput than HULL
- Argument: applications don't specify priority based on resource requirements
  - Can we just implement this at a lower level and call it a day?