Improving Data Centre Performance using Multipath TCP (work in progress)

Mark Handley
Costin Raiciu
Christopher Pluntke
Adam Greenhalgh
Sebastien Barre
Data Centres are Interesting!

As a real problem:
- Networks of tens of thousands of hosts (big money).
- Distributed apps, dense traffic patterns (GFS, BigTable, Dryad, MapReduce)

As a research problem:
- We get to determine the topology, routing, and end-system behaviour as a unified system.
Location independence

- Apps distributed across thousands of machines.
- Want any machine to be able to play any role.

But:
- Traditional data centre topologies are tree based.
- Don’t cope will with non-local traffic patterns.

Much recent research on better topologies.
Traditional data centre topology

- Core Switch
- Aggregation Switches
- Top of Rack Switches
- Racks of servers

Diagram:
- Top of Rack Switches
- 1Gbps
- 10Gbps
- 10Gbps
Fat Tree topology [Fares, 2008]

K=4

Aggregation
Switches

K Pods with K Switches
each

Racks of servers

1Gbps
1Gbps
VL2 topology [Greenberg et al, 2009]

Link-state network carrying only LAs (e.g., 10/8)

Internet

D_N/2 x Intermediate Switches

Int

D_N/2 x 10G

Aggr

D_N/2 x 10G

D_I x Aggregate Switches

2 x 10G

D_I x Aggregate Switches

D_N/4 x ToR Switches

ToR

20 Servers

Fungible pool of servers owning AAs (e.g., 20/8)
BCube topology [Guo et al, 2009]
So many paths, so little time…

- **Need to distribute flows across paths.**

- **Basic solution: Valiant Load Balancing.**
  - Use Equal-Cost Multipath (ECMP) routing.
    - Hash to a path at random.
  - Or, use many differently rooted VLANs.
    - End-host hashes to a VLAN; determines path.
Collisions

Racks of servers

1Gbps

1Gbps
Multipath TCP
Set up multiple subflows between the same pair of endpoints.

Client

Server

Stripe data from one connection across both paths.

Load balances between access links.
Sending simultaneously across more than one path can **balance load and pool resources.**

[Kelly & Voice, Key, Massoulie & Towsley]

Each path runs its own congestion control, to detect and respond to the congestion it sees.

But *link* the congestion control parameters, so as to move traffic away from the more congested paths.
Multipath TCP in Data Centres

- VLB suffers from collisions.
  - Especially on FatTree, BCube.
  - If two flows share a link, each suffers 50%, some other path ends up underused.

- Multipath TCP
  - Uses more paths.
  - Is no more aggressive in aggregate than a single TCP
  - Moves traffic away from congestion.

- Can MP-TCP self-optimize data-centre traffic?
Intuition

With Multipath TCP we can explore many paths:

- Don’t worry about collisions.
- Just don’t send (much) traffic on colliding paths
Multipath TCP in the Fat Tree Topology

K=32 (8K hosts, 256 Paths between endpoints)
Performance depends on topology

FatTree

VL2

BCube
Multipath TCP improves Fairness

FatTree

VL2

BCube
How many MP-TCP subflows are needed?
Centralized Scheduling

- Without TCP, it’s really hard to utilize FatTree.

- Hedera uses a centralized scheduler and flow switching.
  - Start by using VLB
  - Measure all flow throughput periodically.
  - Any flow using more than 10% of its interface rate is explicitly scheduled onto an unloaded link.

How does centralized scheduling compare with MP-TCP?
Simulation bottleneck

- Fluid models can’t capture all the details (RTO, slowstart, etc) that we need to understand to model the behaviour of centralized scheduling.
- Want accurate TCP model at packet-level with 1000 hosts transmitting at 1Gb/s.
  - Aggregate rate: 1Tb/s
- We wrote our own simulator: htsim
MP-TCP vs Centralized Dynamic Scheduling

![Bar Chart]

Throughput (% of max)

- VLB
- 1s
- 500ms
- 100ms
- 10ms
- MTCP

First Fit Scheduler
Can’t we just use many TCP connections?

Loss rate of MP-TCP ("linked") vs multiple uncoupled TCP flows

Retransmit timeouts with MP-TCP ("linked") vs uncoupled TCP flows
Conclusions

- Multipath TCP seems a really good fit to proposed modern data centre topologies.
  - Improved throughput
  - Improved fairness
  - More robust than centralized scheduling

- To do: understand the end-host performance limitations with many subflows.