Utilizing Datacenter Networks: Centralized or Distributed Solutions?

Costin Raiciu

Department of Computer Science

University Politehnica of Bucharest

We've gotten used to great applications

Enabling Such Apps is Hard

- Apps
  - Process huge amounts of data
  - Are fast
  - Are reliable
- One machine is not enough
  - Limited reliability, speed
- Super computers are expensive
- Use *many commodity machines* instead …

Data Centers Rule the World

Cloud computing

- Economies of scale: networks of tens thousands of hosts

Datacenter apps support web search, online stores

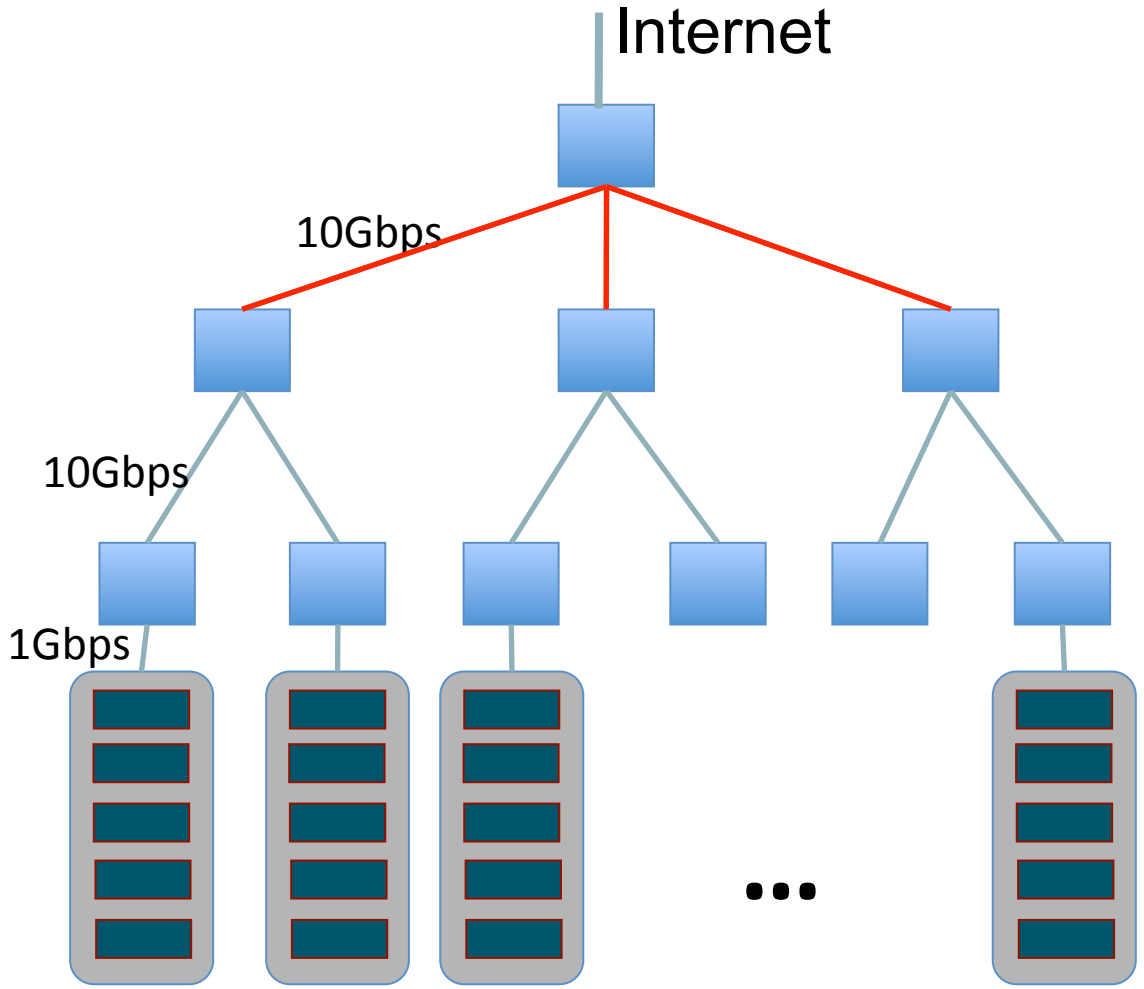
- Web search, GFS, BigTable, DryadLINQ, MapReduce
- Dense traffic patterns
- Intra datacenter traffic is increasing in volume

Flexibility is Important in Data Centers

- Apps distributed across thousands of machines.
- Flexibility

  want any machine to be able to play any role.

Traditional Data Center Topology



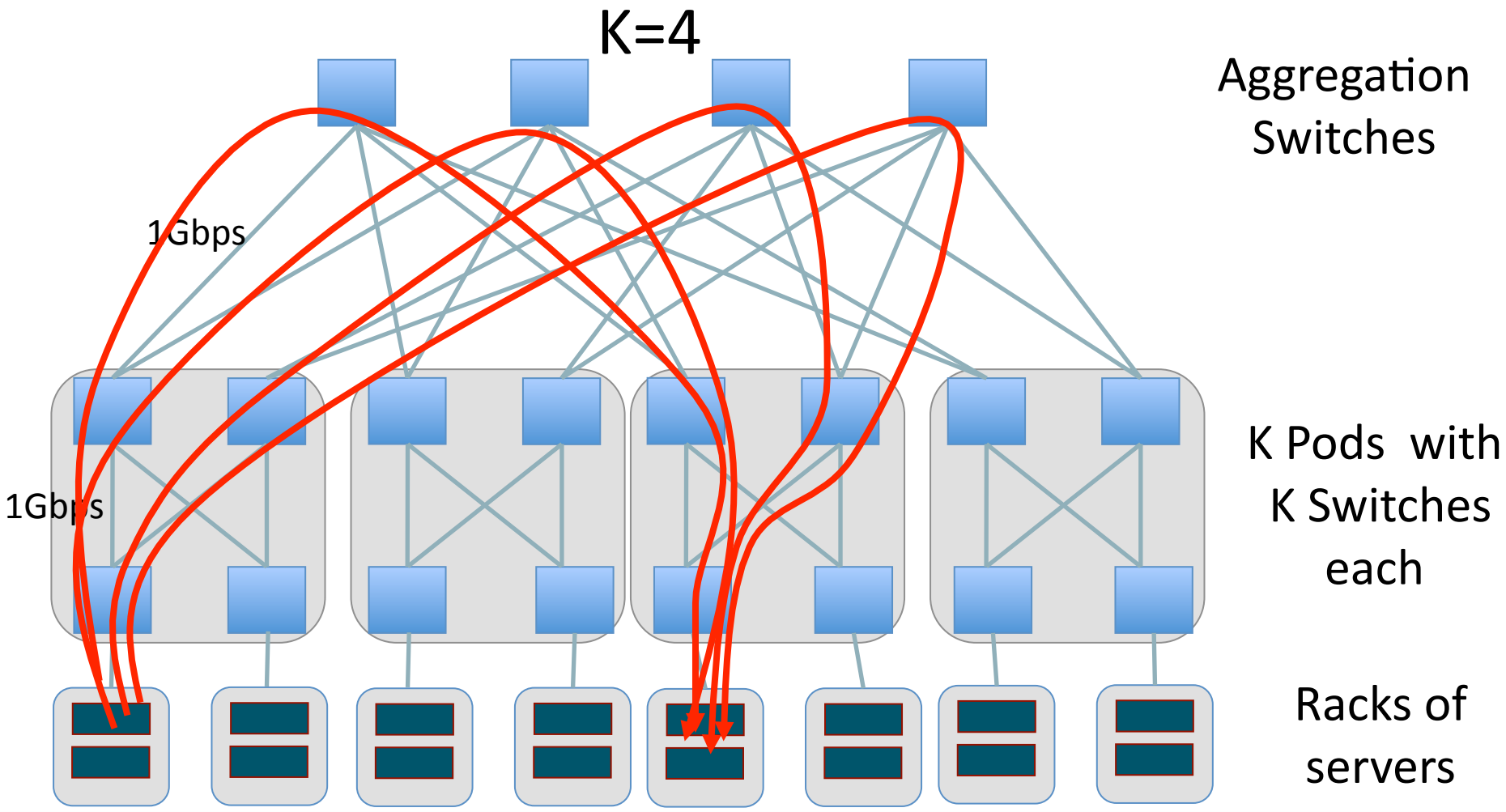
Core Switch

Aggregation Switches

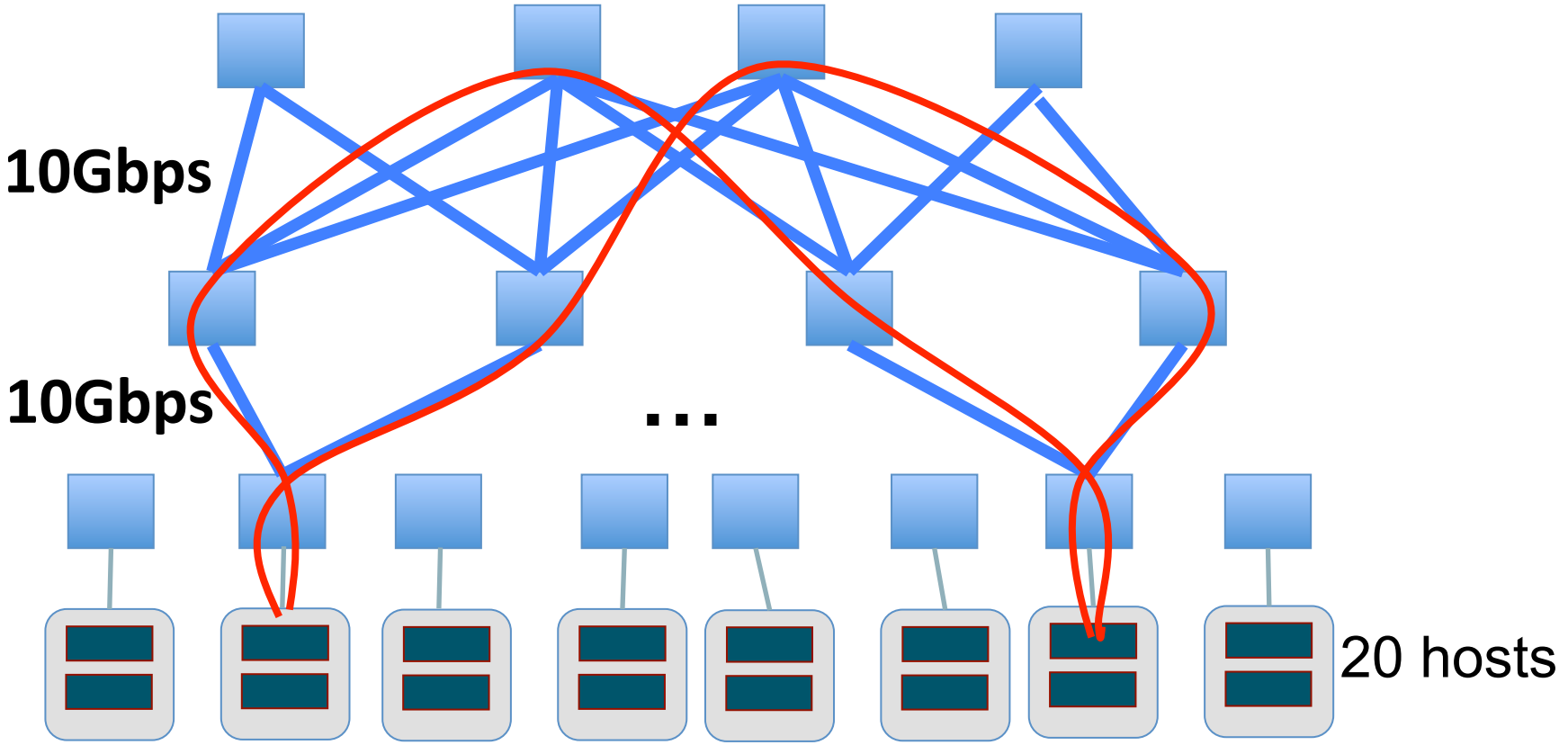
Top of Rack Switches

Racks of servers

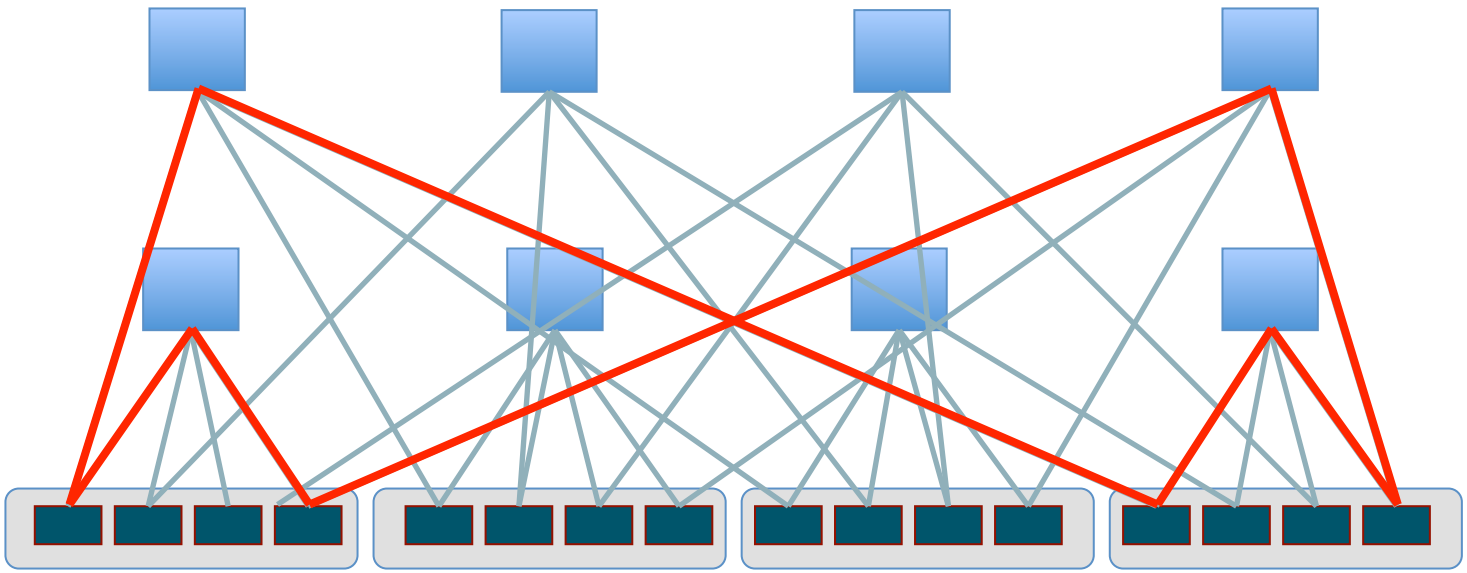
Fat Tree Topology [Fares et al., 2008; Clos, 1953]



VL2 Topology [Greenberg et al, 2009, Clos topology]



BCube Topology [Guo et al, 2009]



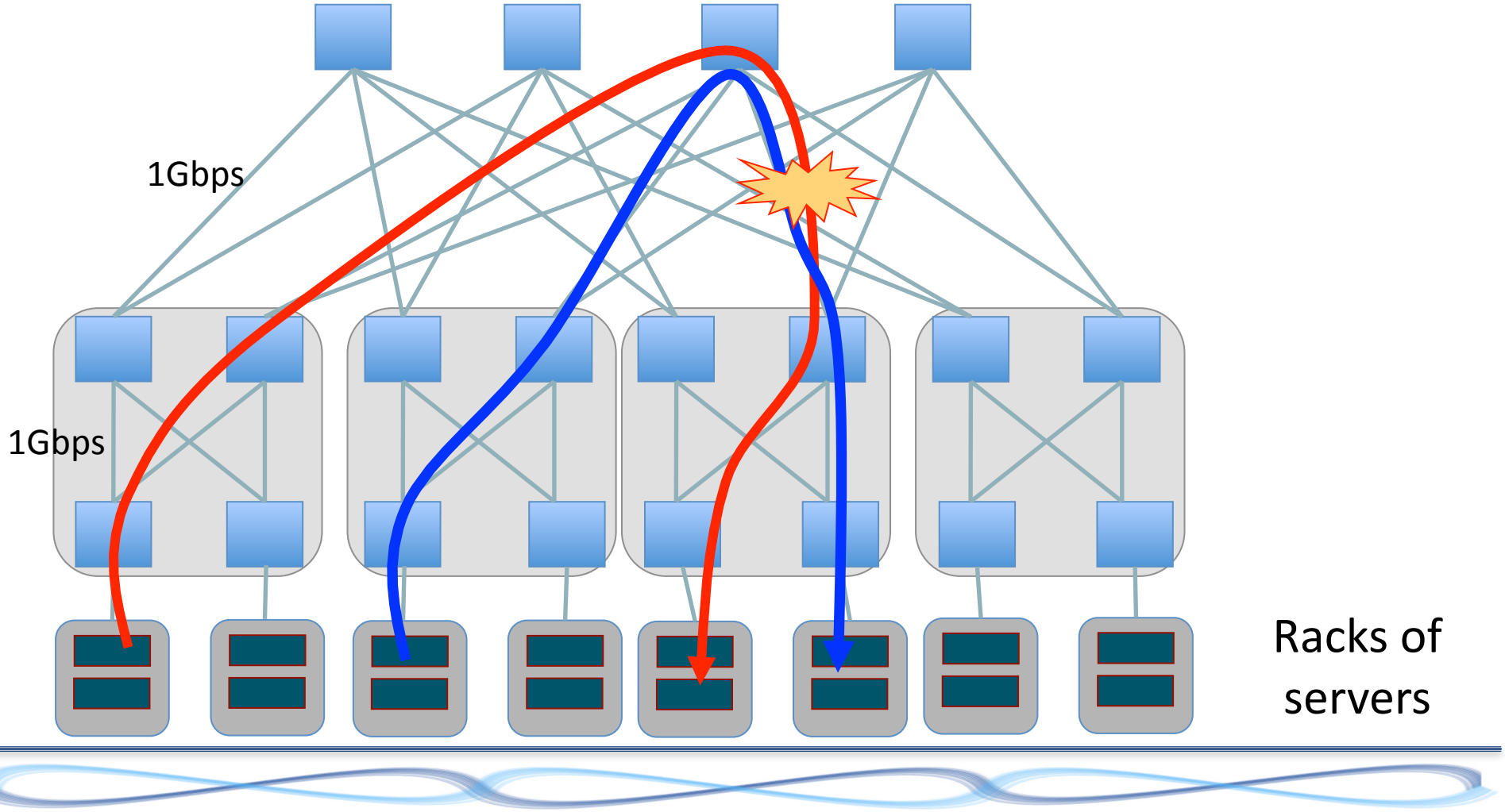
How Do We Route Packets in Data Centers?

- Traditional Routing
  - Spanning Tree Protocol - kills all redundancy
- Instead datacenters use one of the following techniques:
  - Multiple VLANs
  - OSPF
  - TRILL (new IETF standard)

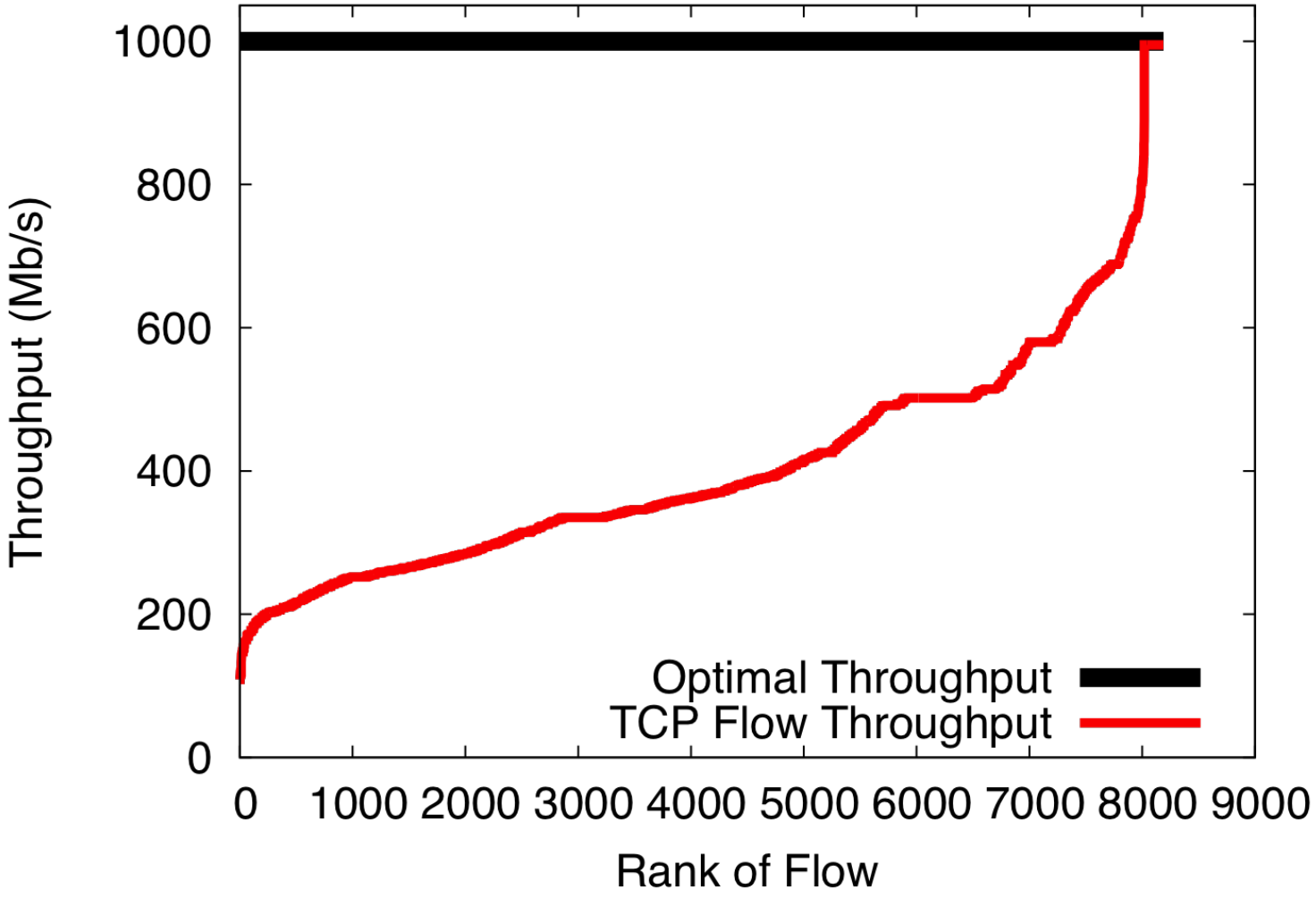
How Do We Use this Capacity?

- **Need to distribute flows across available paths.**
- Basic solution: Random Load Balancing.
  - Use Equal-Cost Multipath (ECMP) routing (OSPF, TRILL)
    - Hash to a path at random.
  - Sources randomly pick a VLANs.
    - In practice sources have multiple interfaces – pick a random source address for the flow

Collisions



Single-path TCP collisions reduce throughput



How bad are collisions?

- Capacity wasted (worst case):
  - FatTree – 60%
  - BCube – 50%
  - VL2 – 25%

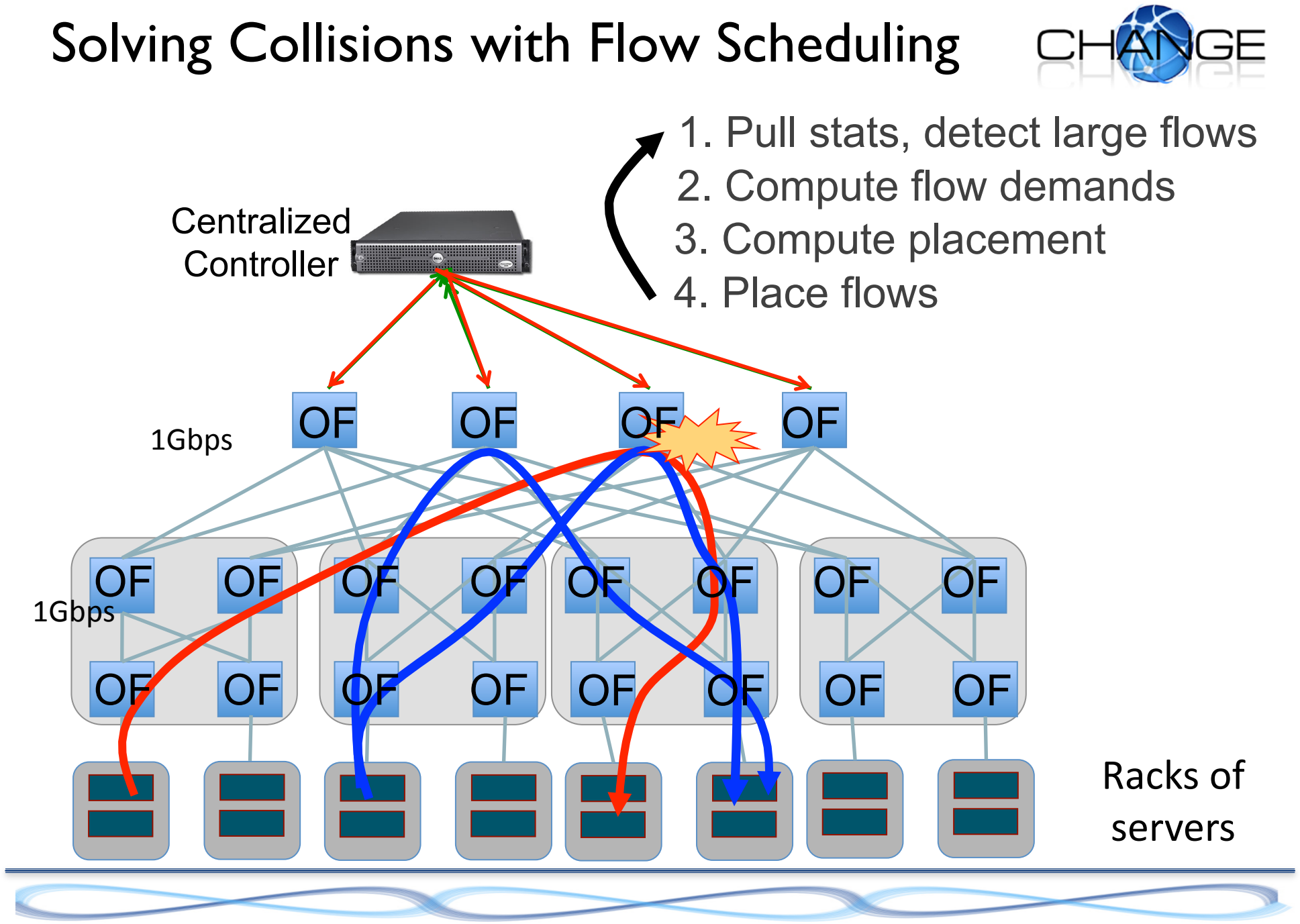
How do we address this problem?

- I will discuss two solutions
  - Flow scheduling
  - Multipath TCP

Flow Scheduling

Hedera – Fares et al. NSDI 2010

Solving Collisions with Flow Scheduling



Hedera Main Idea

- Schedule elephant flows
  - They carry most of the bytes
- ECMP deals with short flows

Detecting Elephants

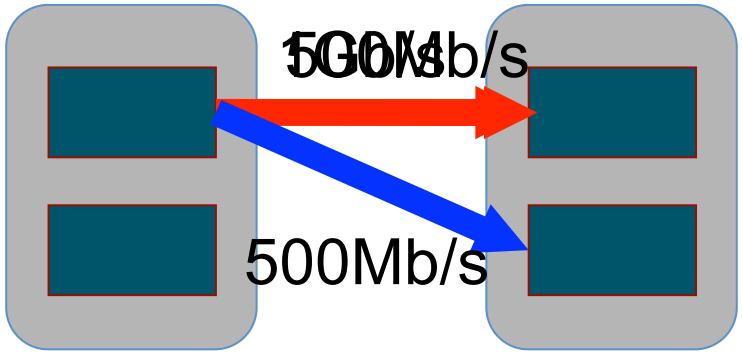
- Pull edge switches for byte counts
  - Flows exceeding 100Mb/s are large
- What if only short flows?
  - ECMP should be good enough

Demand Estimation

- Current flow rates are a poor indicator of flow demand
  - Network could be the bottleneck

- Hedera's approach: what would this flow get if the network was not a bottleneck?

Demand estimation: simple example



- General Approach: Iterative algorithm

Allocating Flows to Paths

- Multi-Commodity Flow Problem
  - Single path forwarding
    - Expressed as Binary Integer Programming
    - NP-Complete
    - Solvers give exact solution but are impractical for large networks

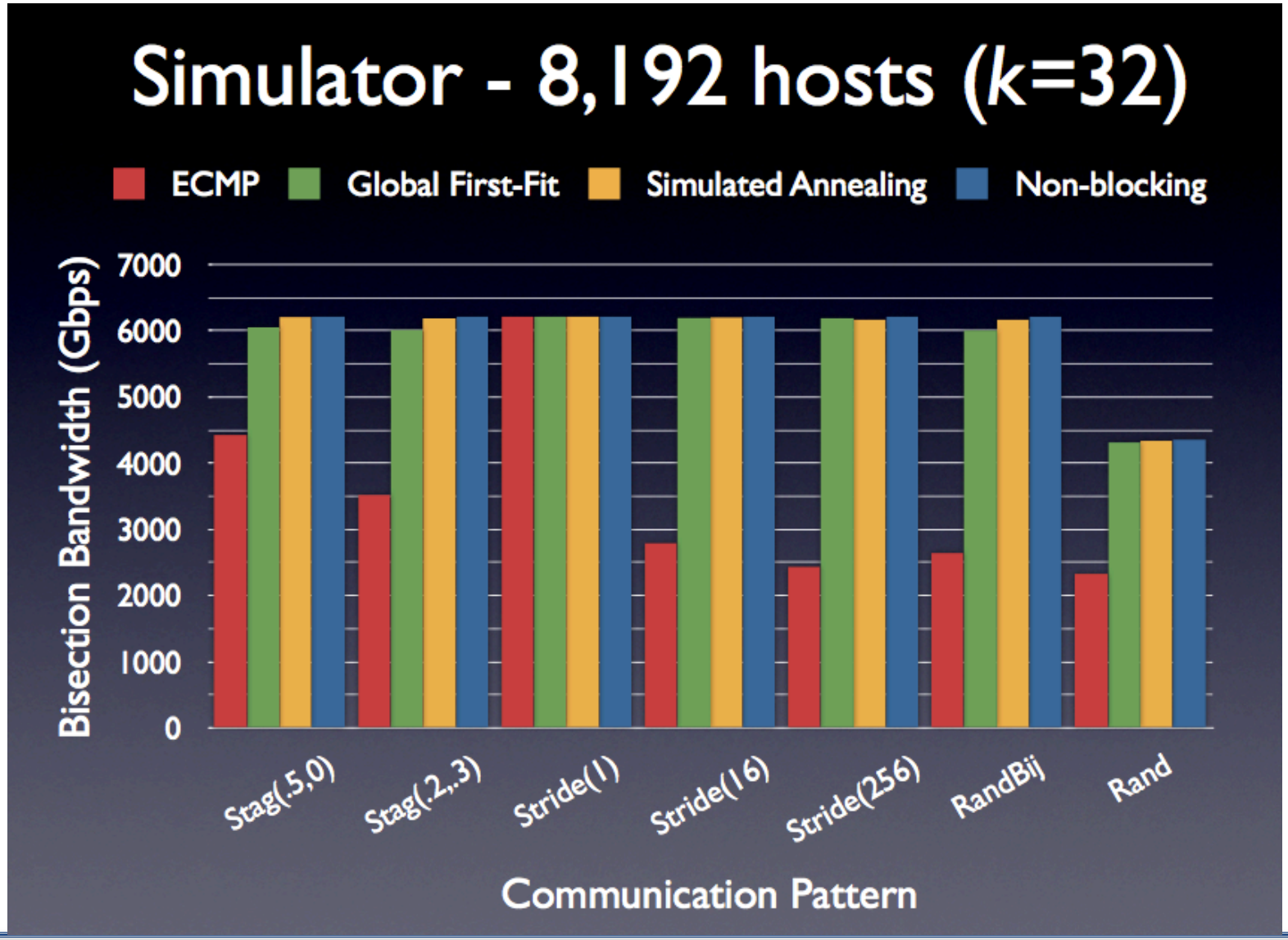
Approximating Multi-Commodity Flow

- Global First Fit
  - Linearly search all paths until one that can accommodate the traffic is found
  - Flows placed upon detection, are not moved
- Simulated Annealing
  - Probabilistic search for good solutions that maximize bisection bandwidth

Fault Tolerance

- Scheduler failure
  - all soft state, just fall back to ECMP
- Link, switch failures
  - Portland notifies the scheduler

Does it work?

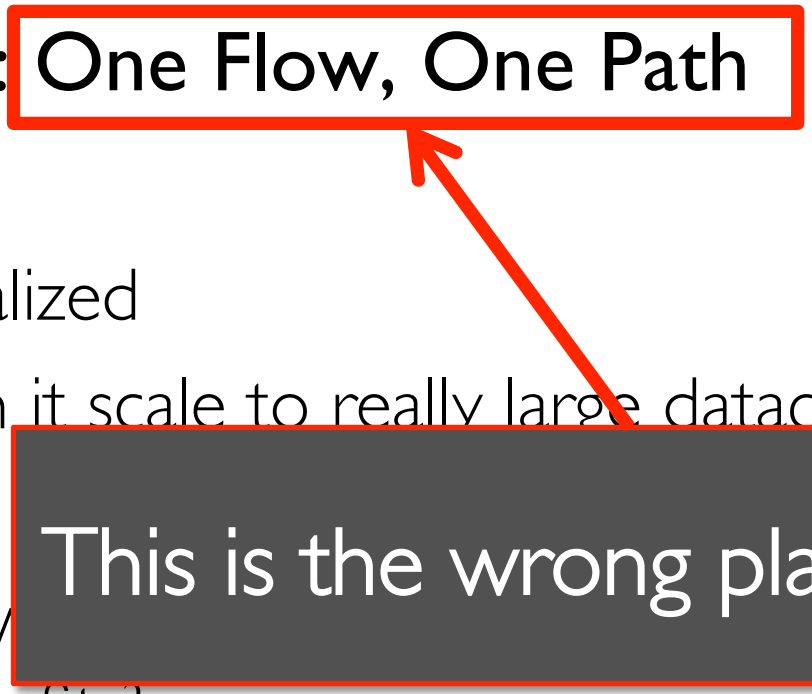

Simulator - 8,192 hosts (k=32)
ECMP
Global First-Fit
Simulated Annealing
Non-blocking
Bisection Bandwidth (Gbps)
7000
6000
5000
4000
3000
2000
1000
0
Stag(.5,0)
Stag(.2,.3)
Stride(1)
Stride(16)
Stride(256)
RandBij
Rand
Communication Pattern

Hedera: One Flow, One Path

- Centralized
  - Can it scale to really large datacenters?
- Needs a very tight control loop
  - How often does it need to run to achieve these benefits?
- Strong assumption:

  traffic is always bottlenecked by the network
  - What about app-bound traffic, e.g disk reads/writes?

Hedera: One Flow, One Path



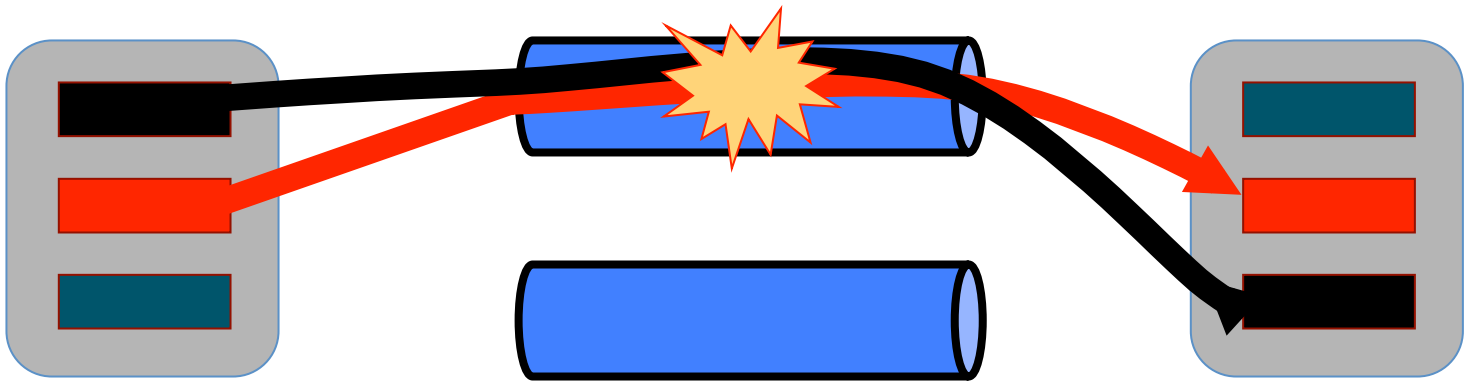
- Centralized
  - Can it scale to really large datacenters? **MAYBE**
- Needs
  - Hov … se benefits?
- Strong assumption: **Only Hosts Know**

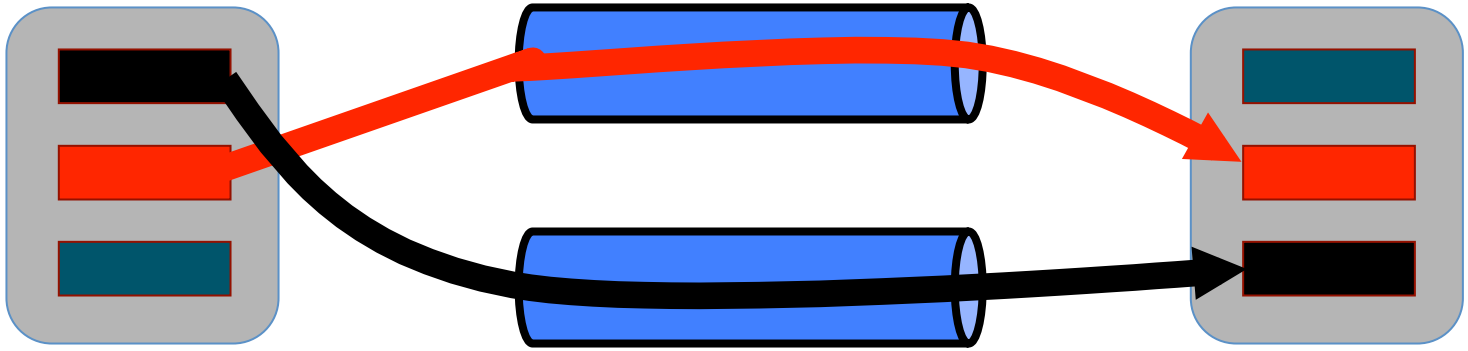
  traffic is always bottlenecked by the network
  - What about app-bound traffic, e.g disk reads/writes?

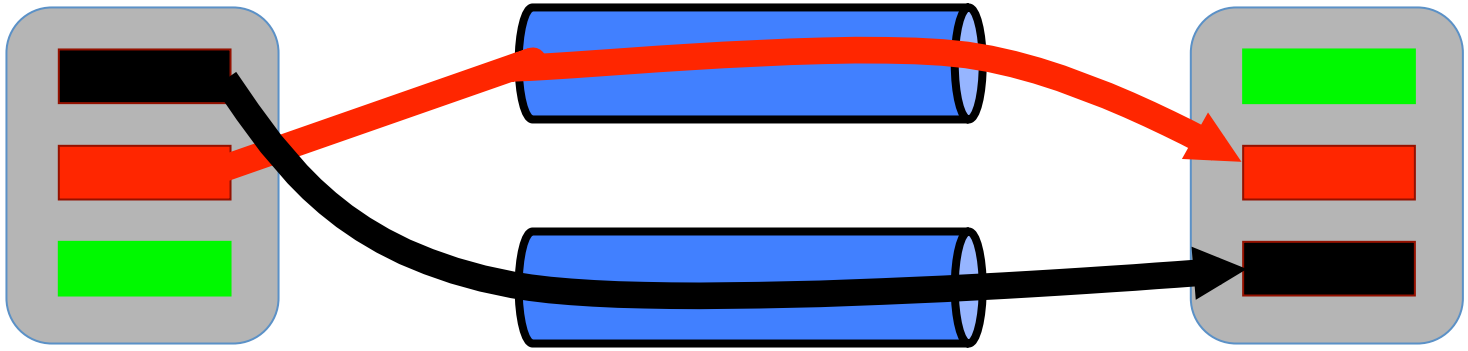
Multipath topologies need multipath transport

Multipath transport enables better topologies

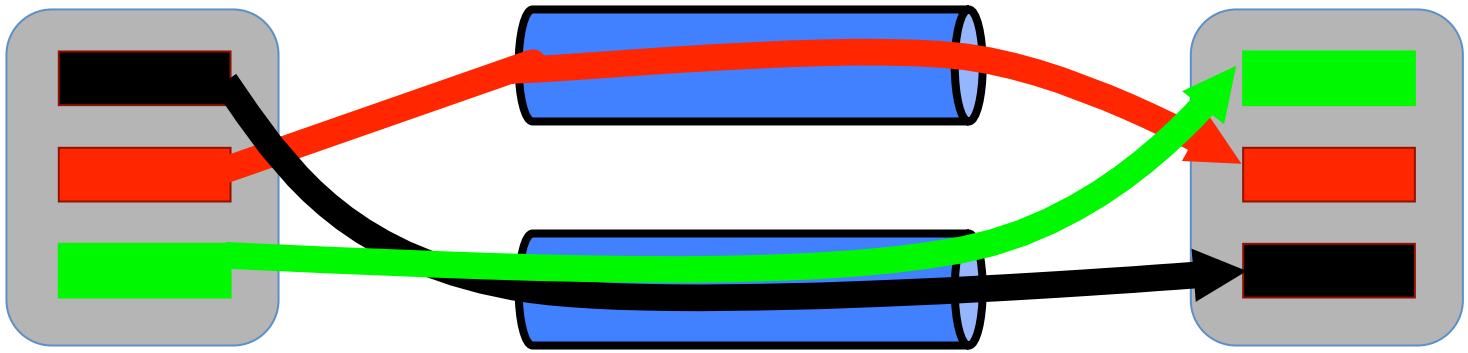
Collision



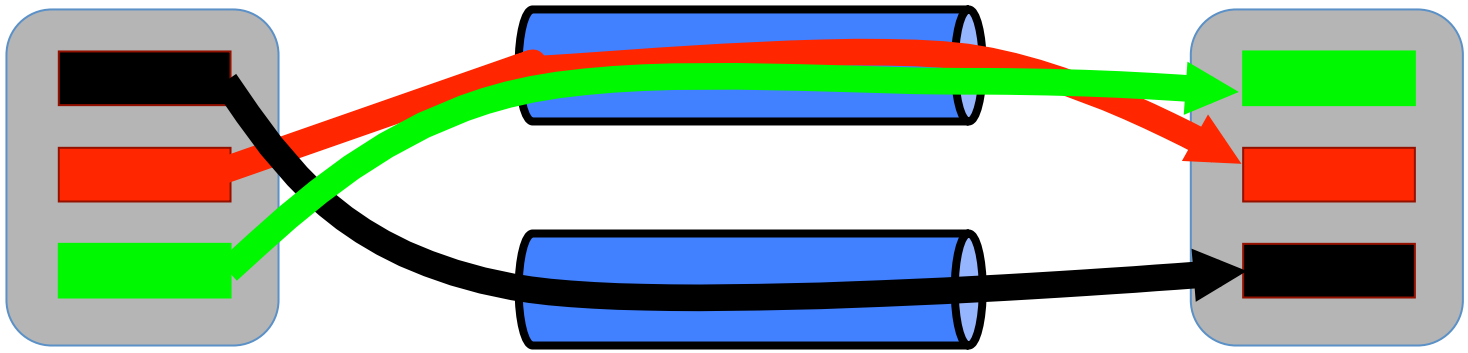


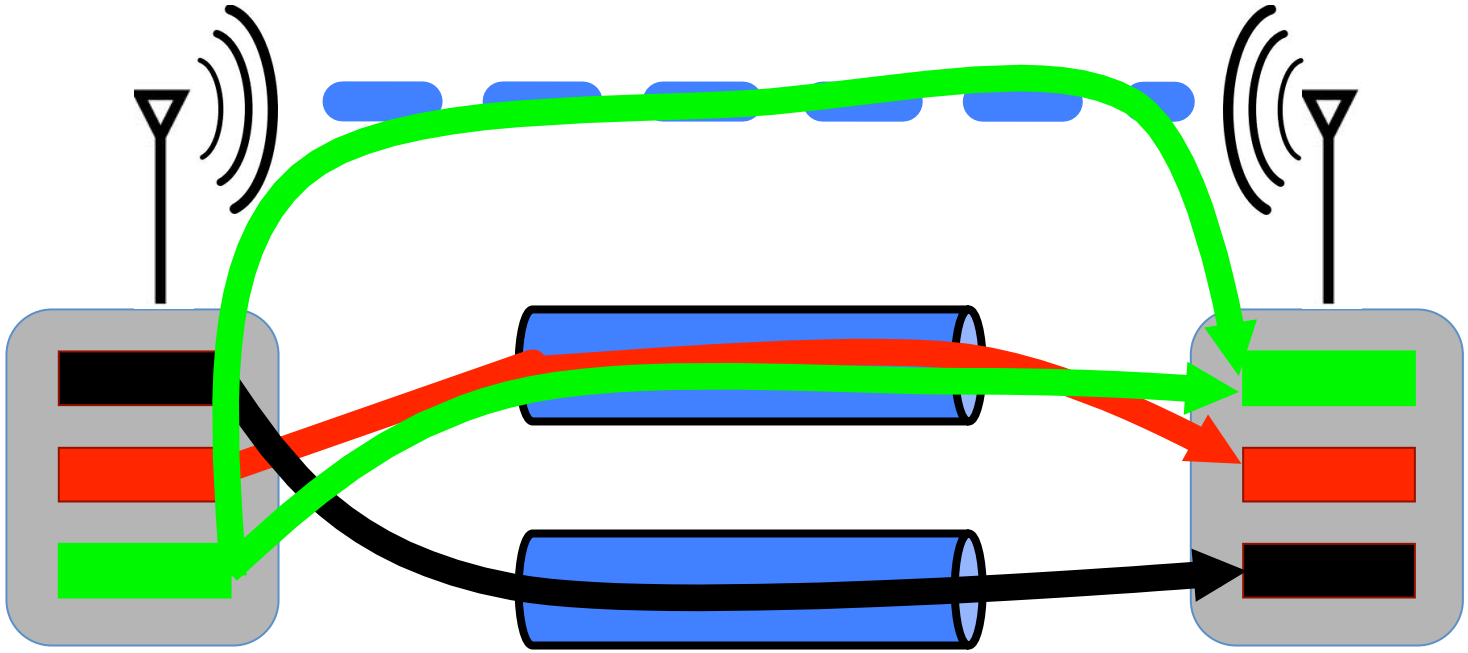


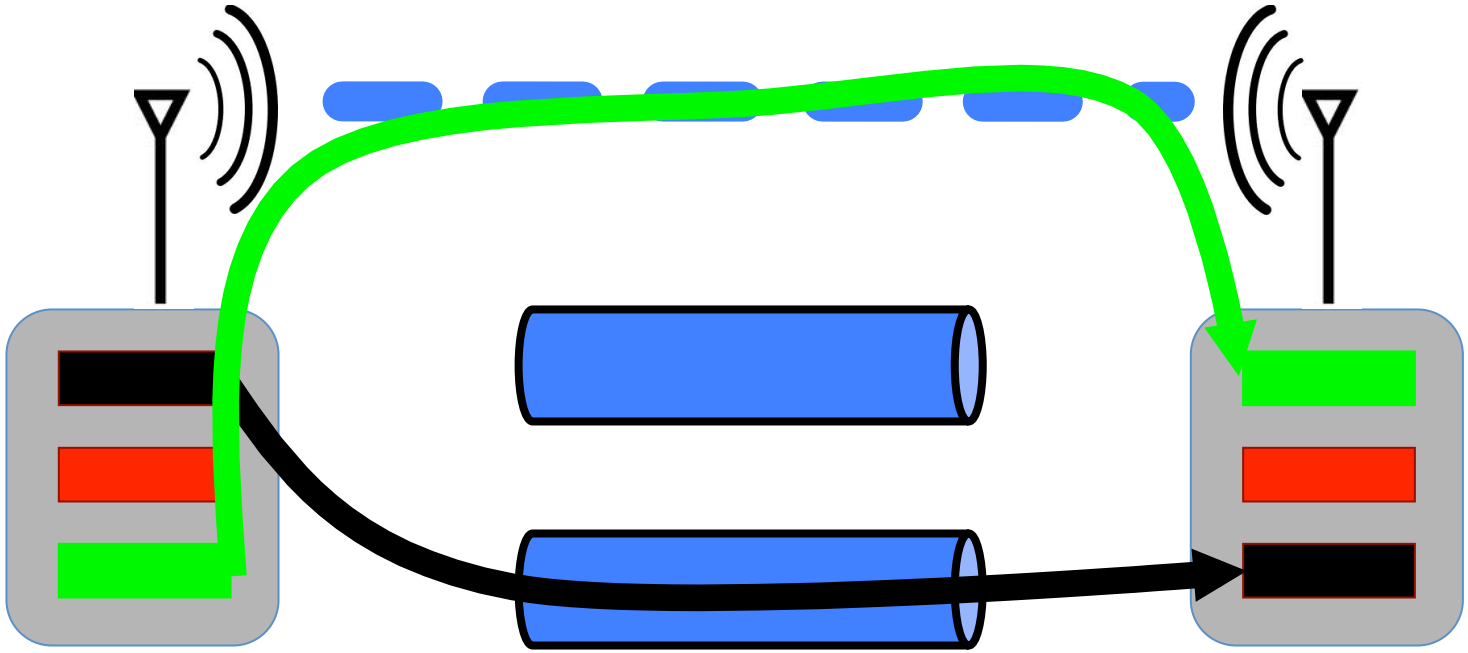
Not fair



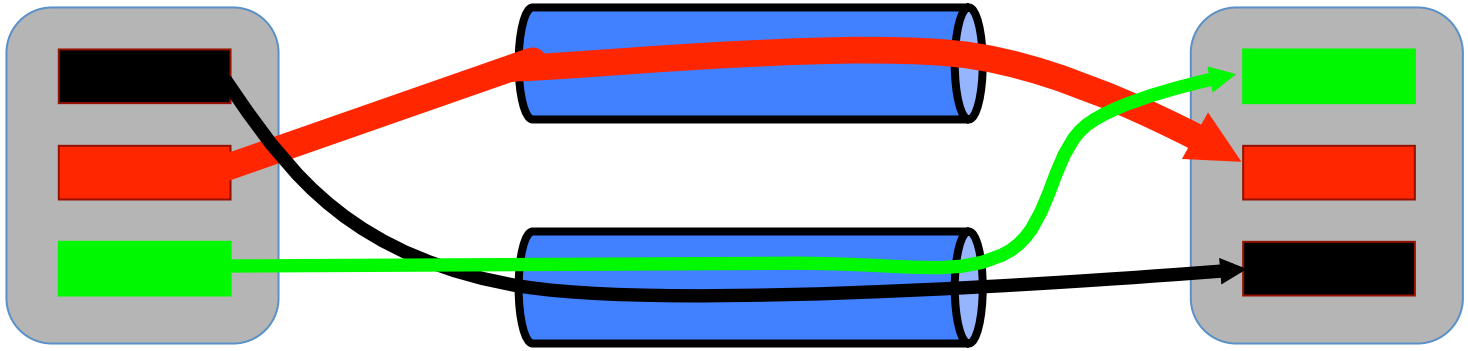
Not fair



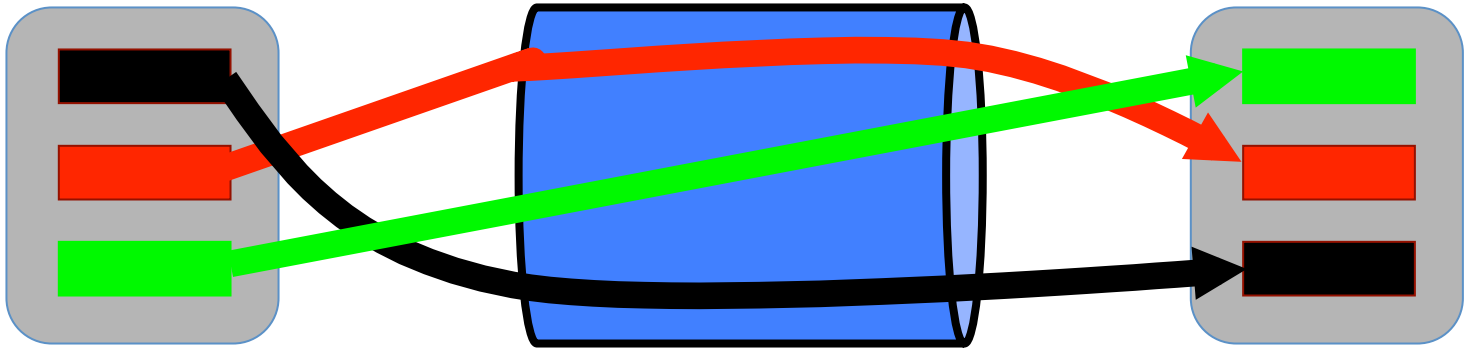




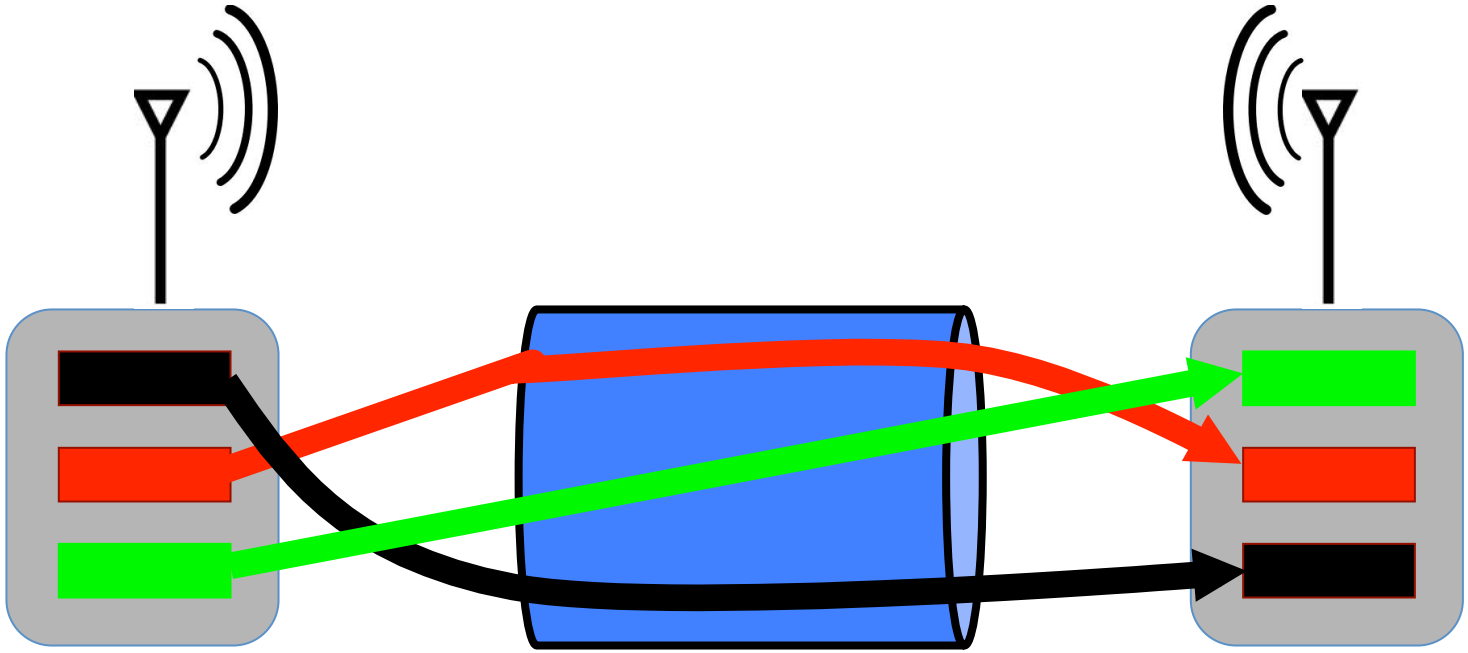
No matter how you do it,
mapping each flow to a path is the wrong goal



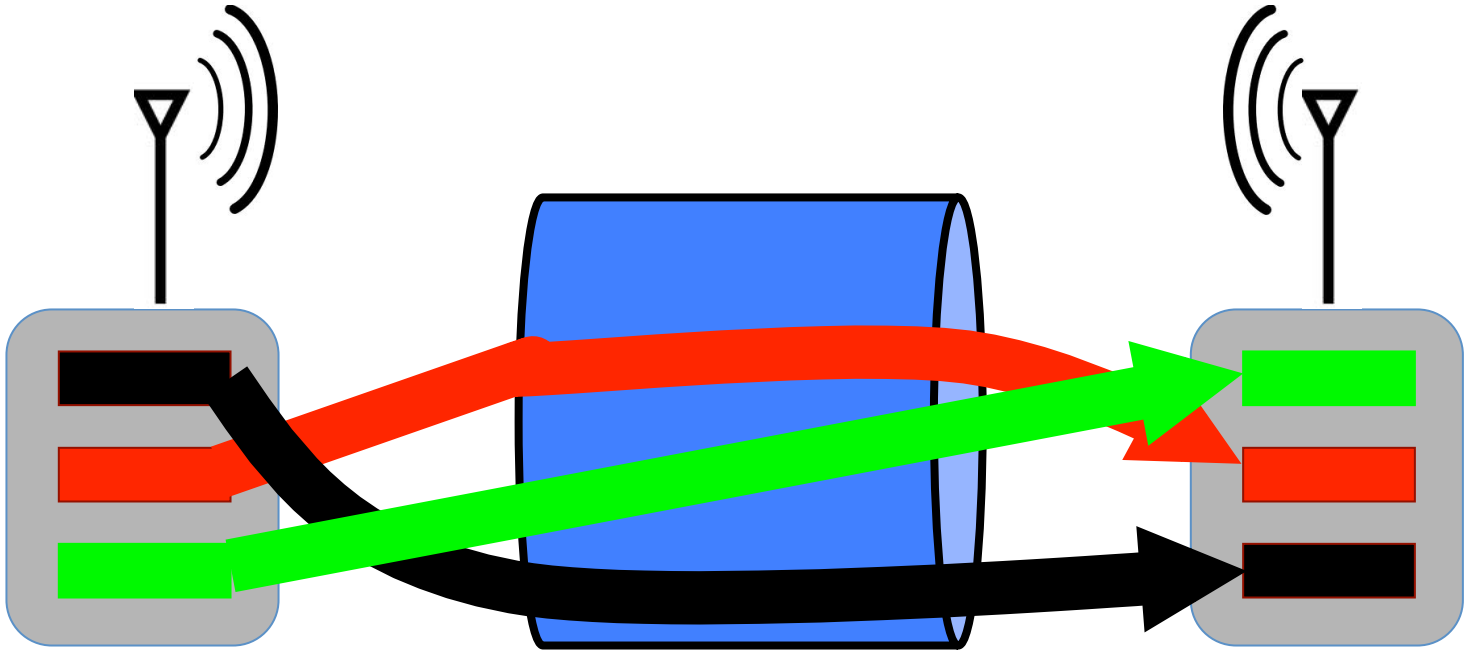
Instead, we should **pool** capacity from different links



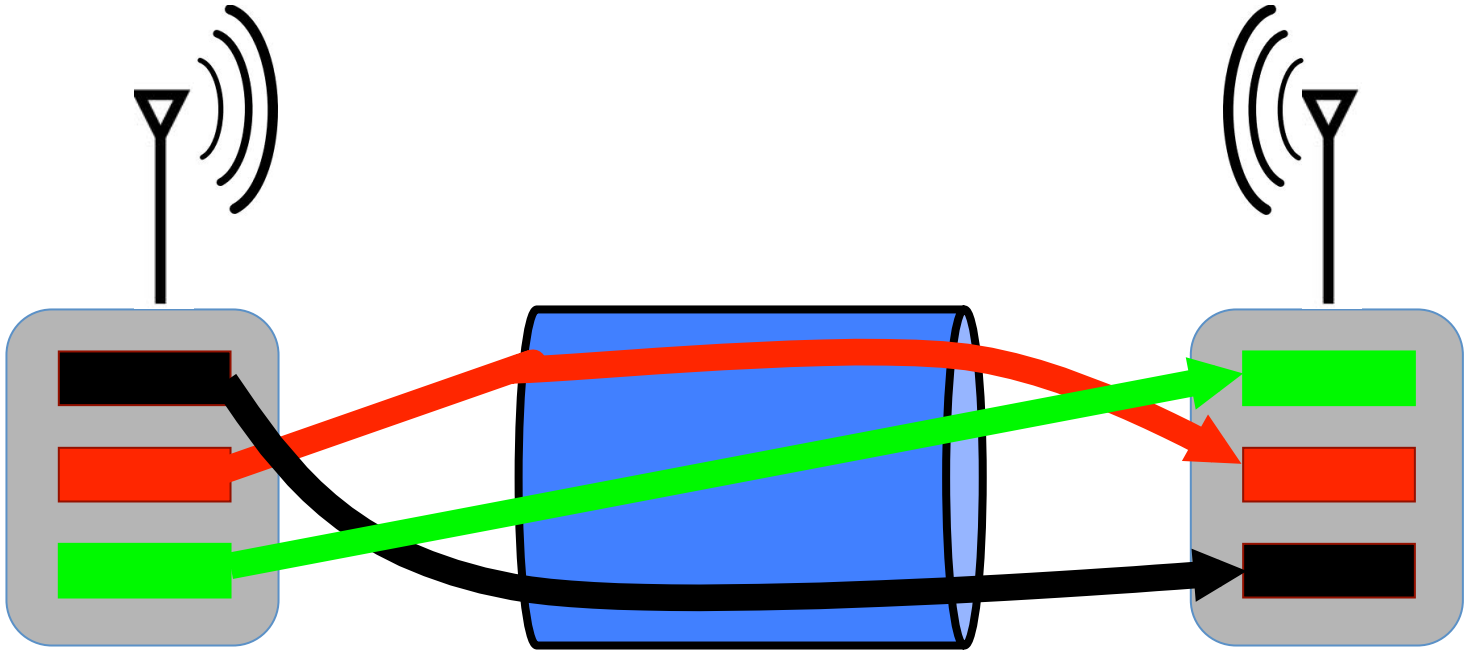
Instead, we should **pool** capacity from different links



Instead, we should **pool** capacity from different links



Instead, we should **pool** capacity from different links



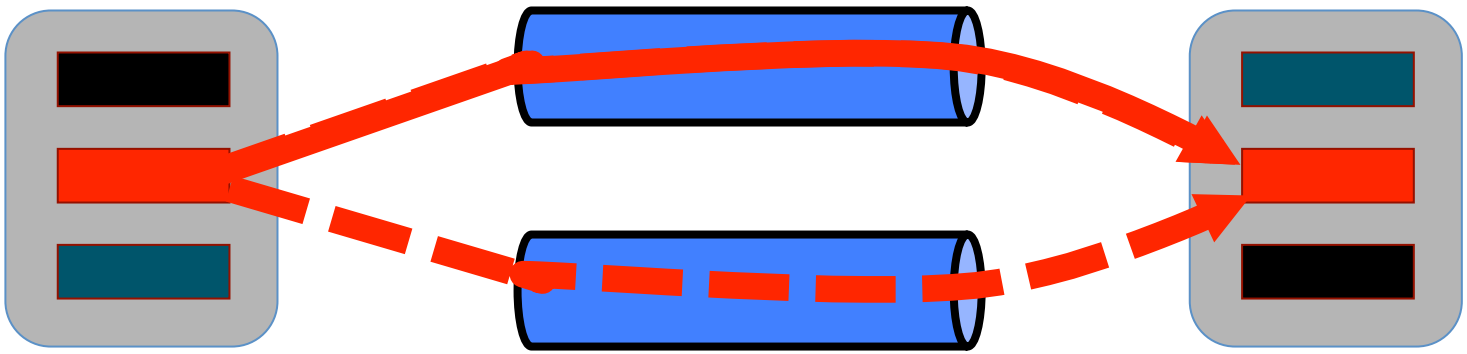
Multipath Transport

Multipath Transport can pool datacenter networks

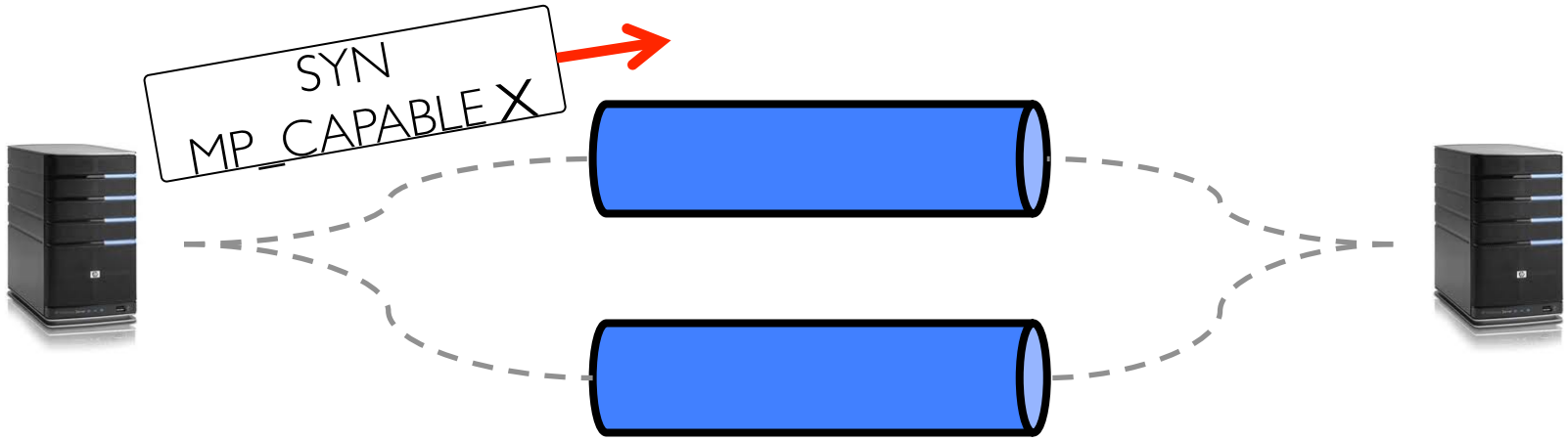
- Instead of using one path for each flow, use many random paths
- Don't worry about collisions.
- Just don't send (much) traffic on colliding paths

Multipath TCP Primer [IETF MPTCP WG]

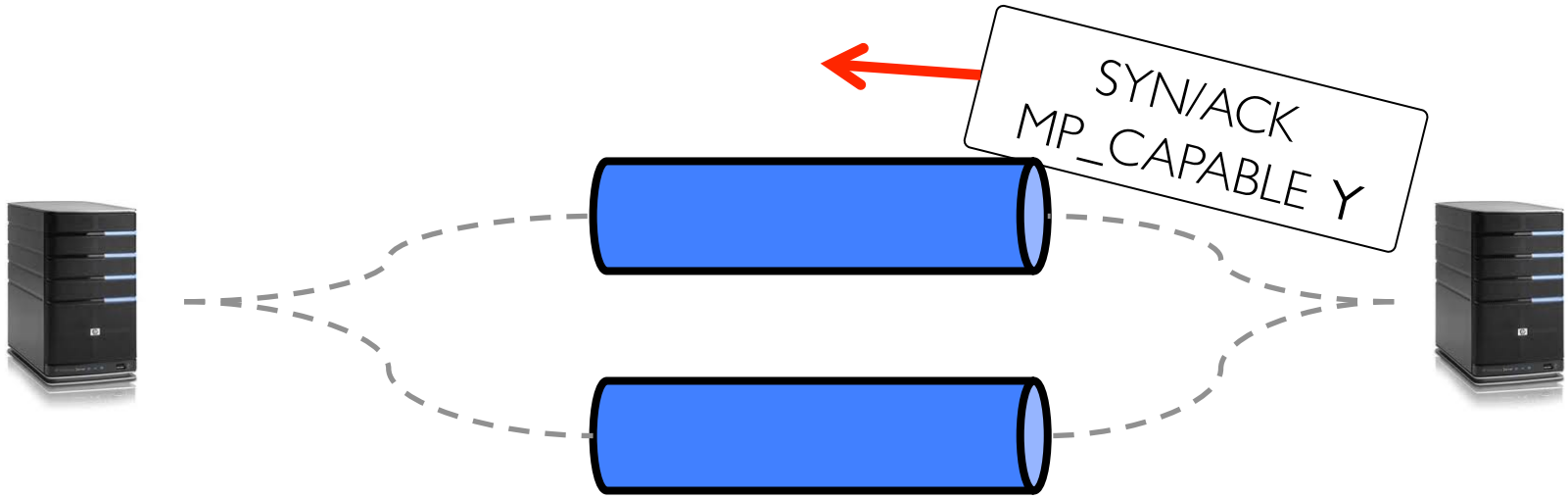
- MPTCP is a drop in replacement for TCP
  - Works with unmodified applications
  - Over the existing network



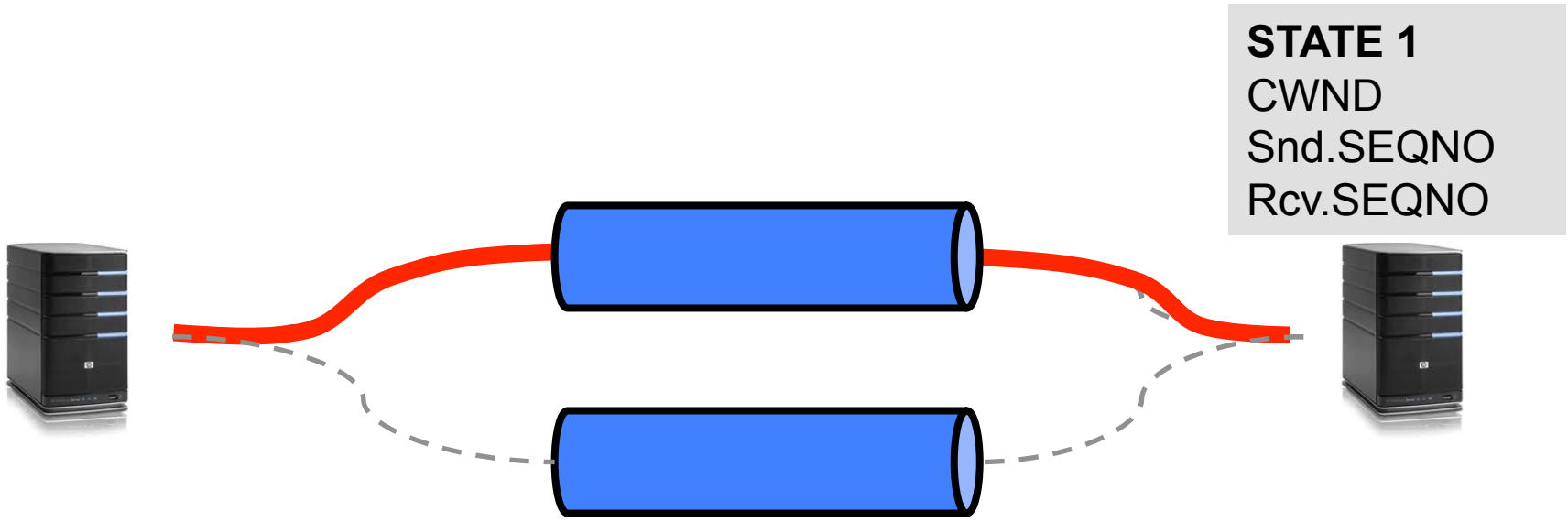
MPTCP Operation



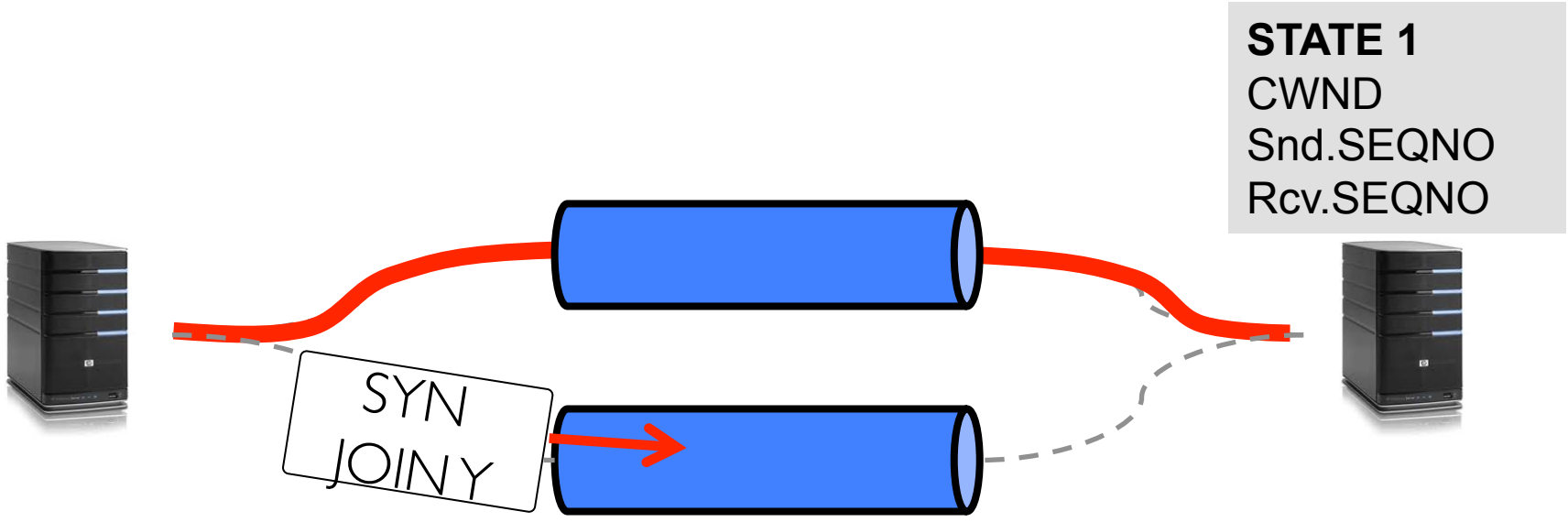
MPTCP Operation



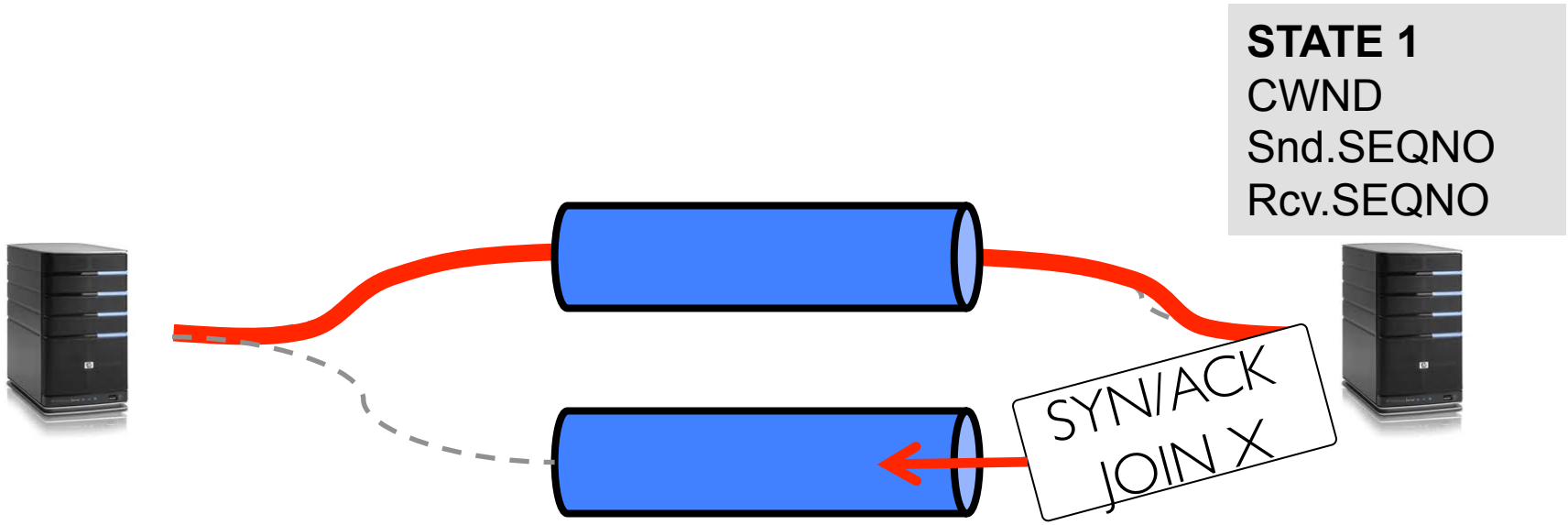
MPTCP Operation



MPTCP Operation



MPTCP Operation

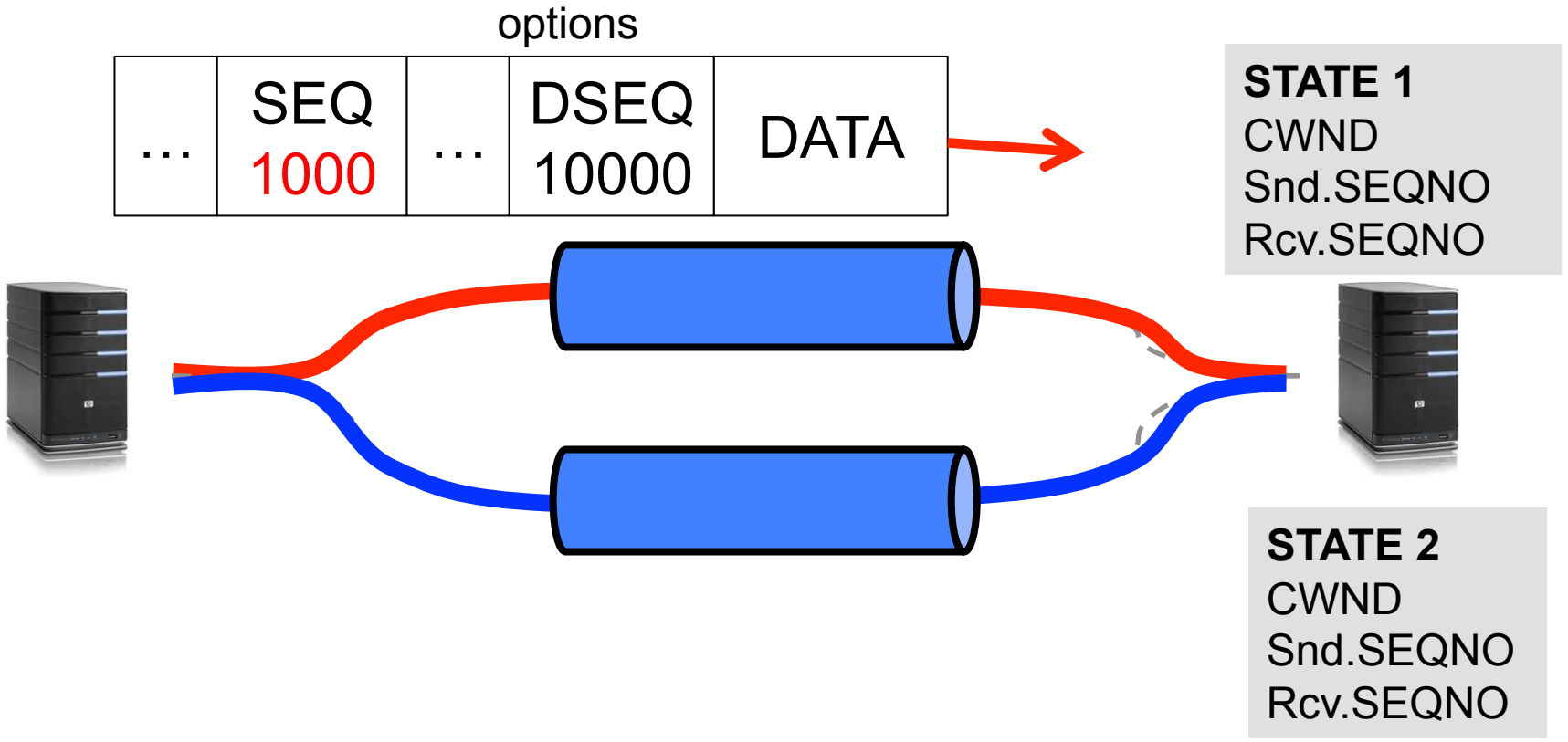


MPTCP Operation

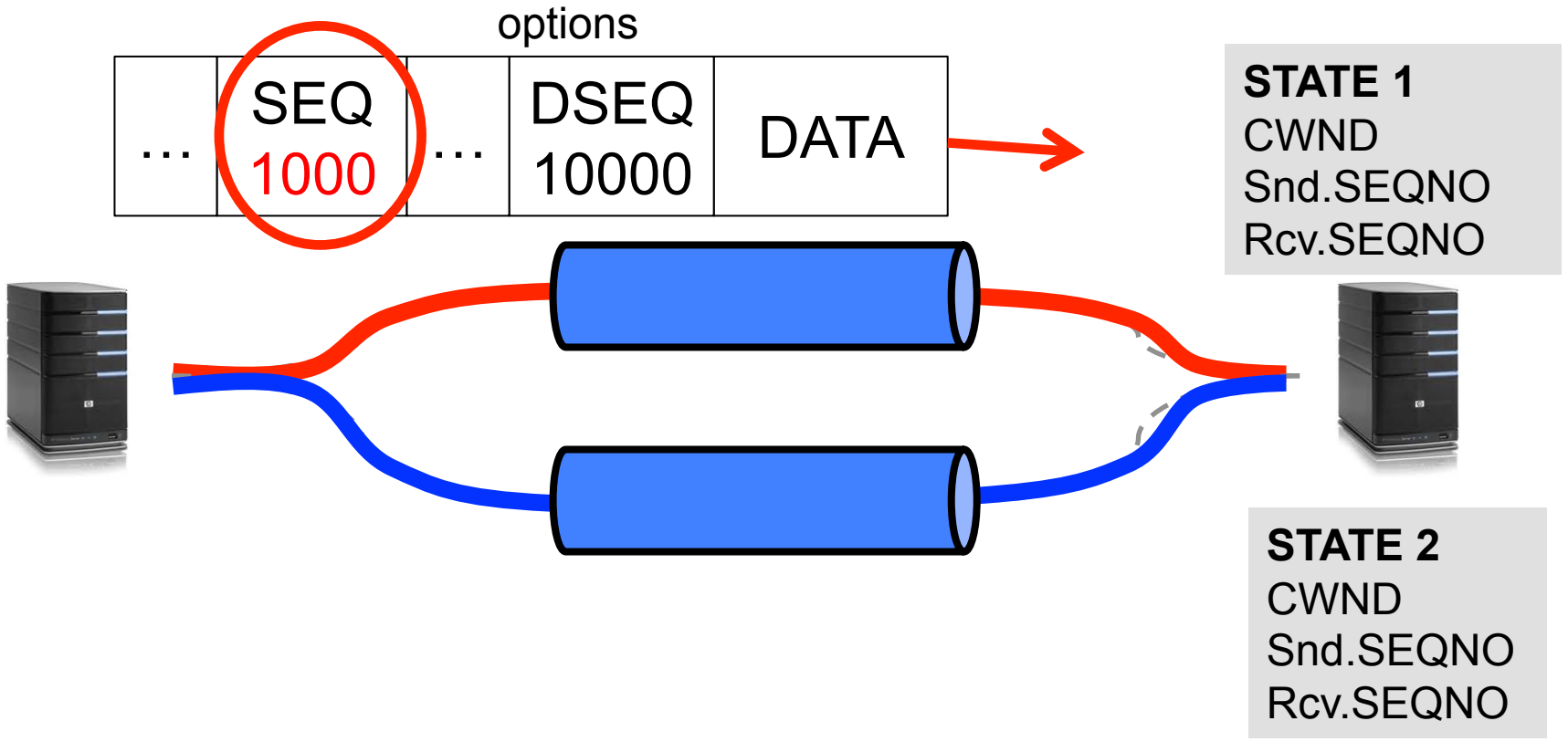
STATE 1
CWND
Snd.SEQNO
Rcv.SEQNO

STATE 2
CWND
Snd.SEQNO
Rcv.SEQNO

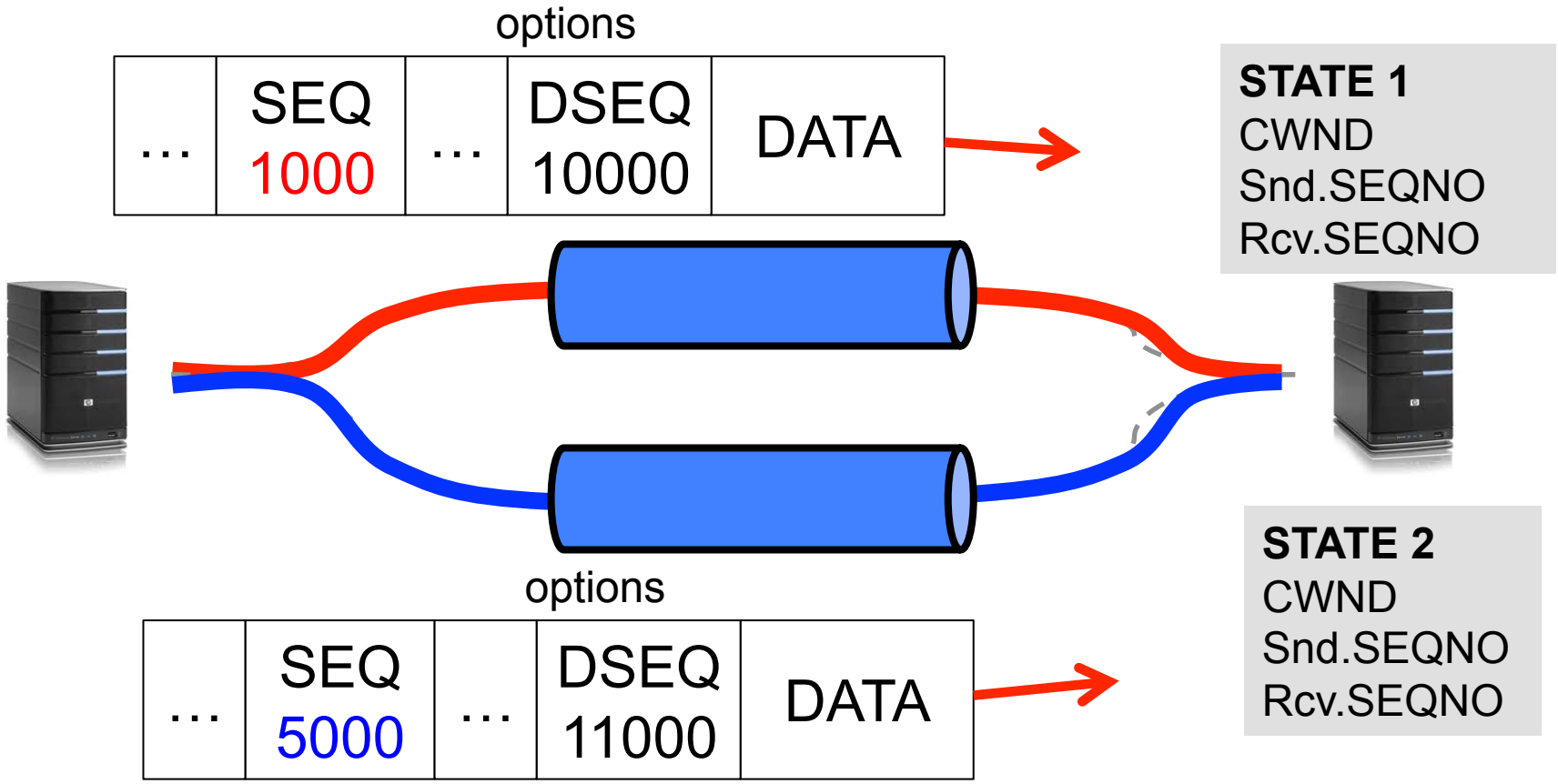
MPTCP Operation



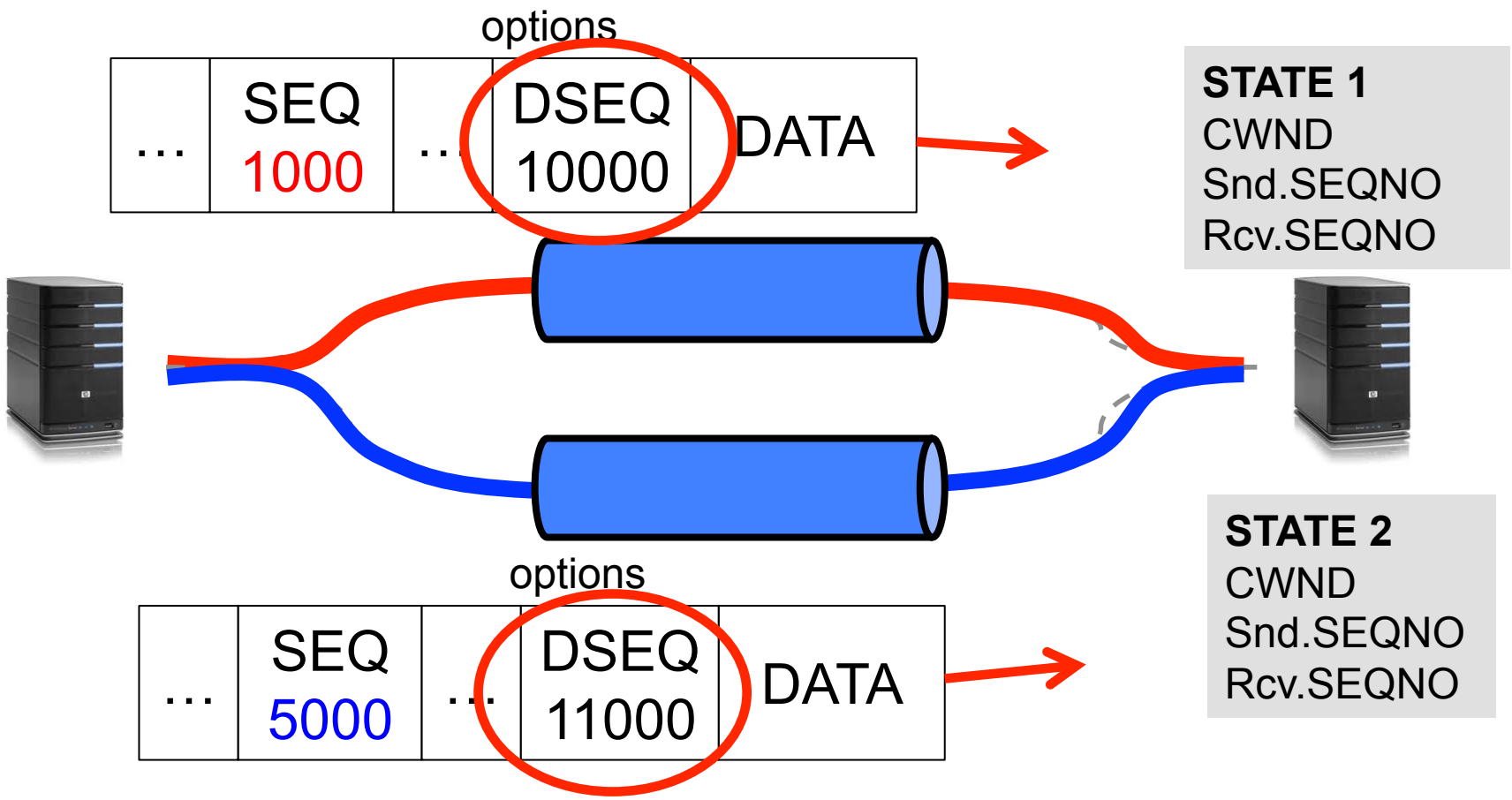
MPTCP Operation



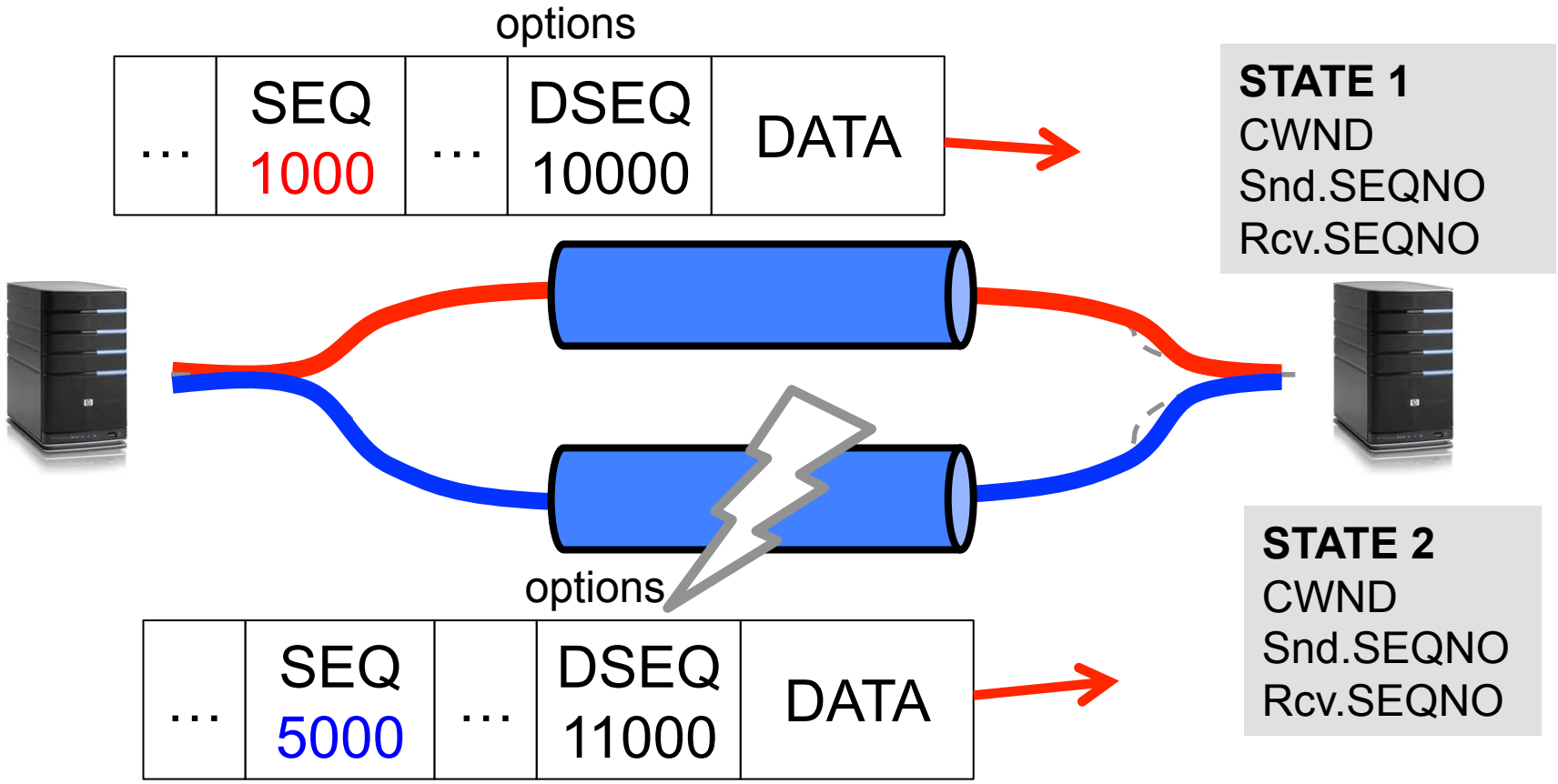
MPTCP Operation



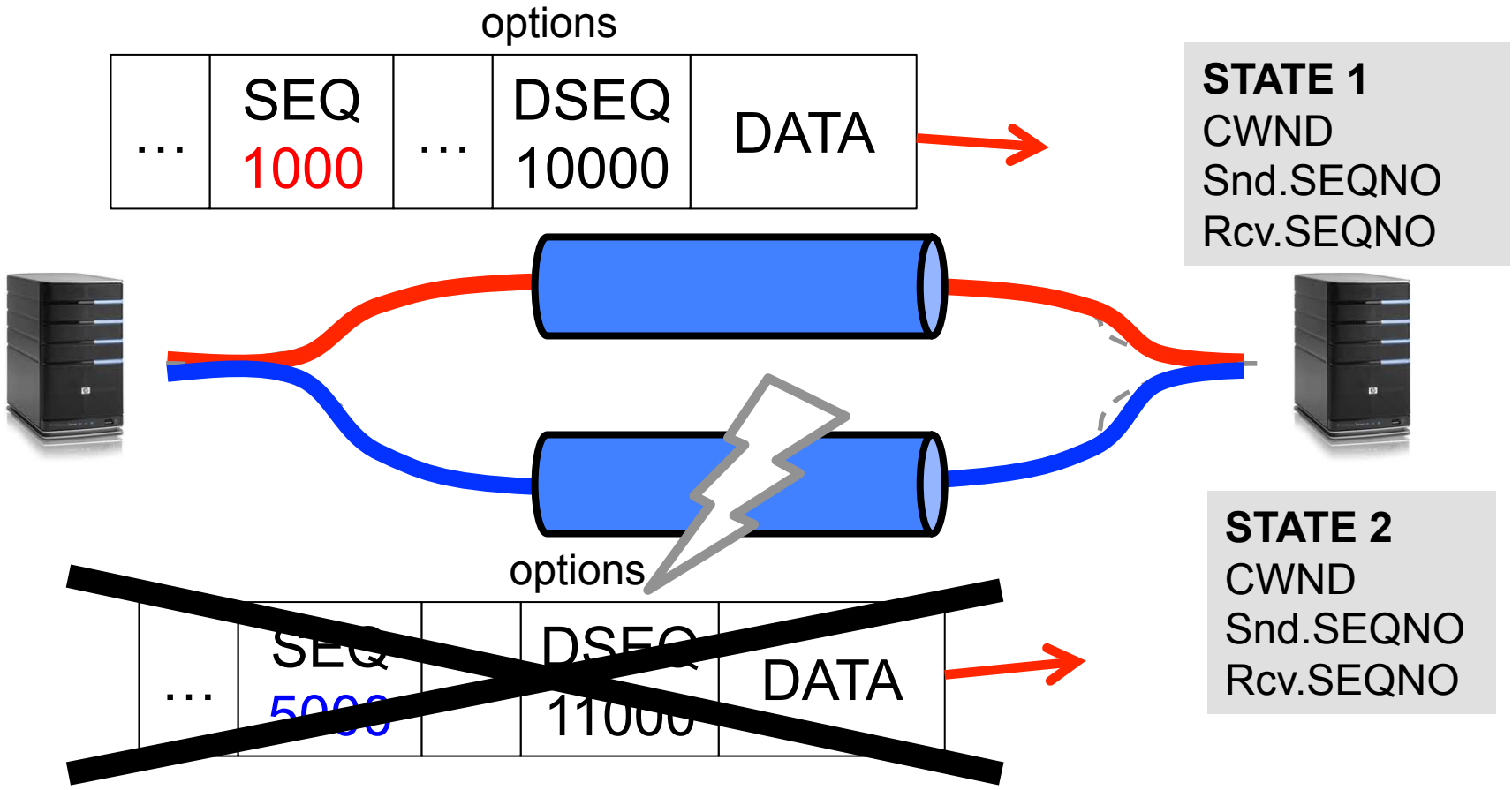
MPTCP Operation



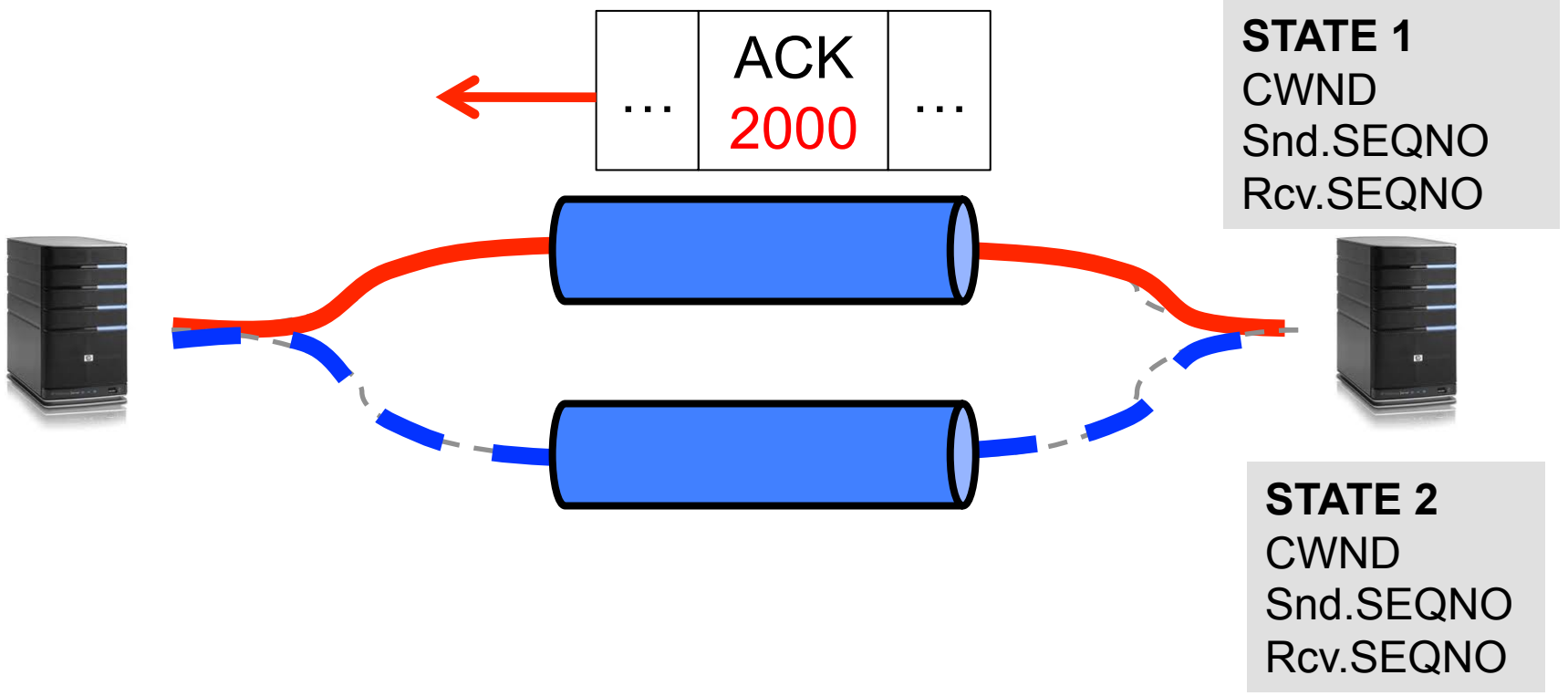
MPTCP Operation



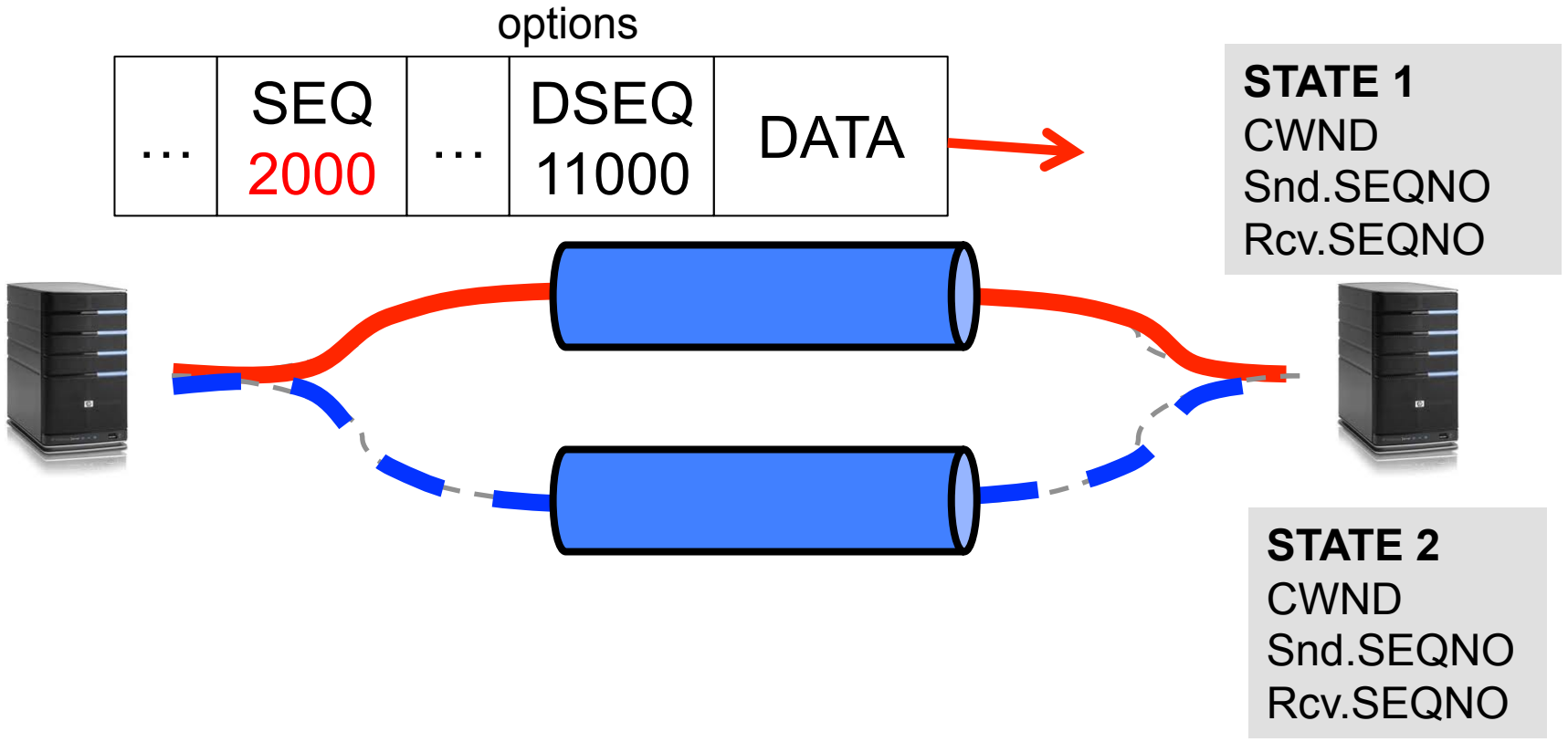
MPTCP Operation



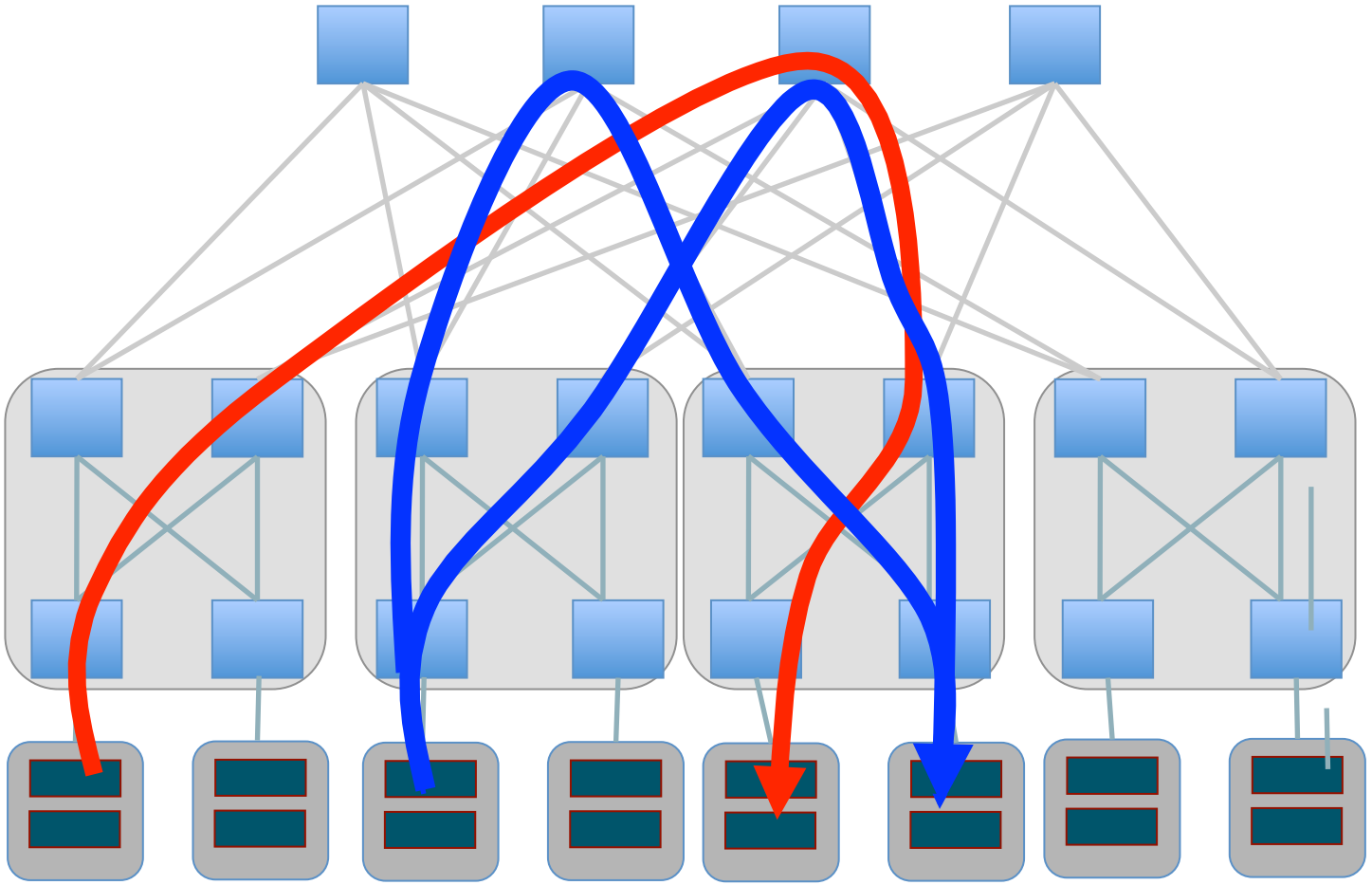
MPTCP Operation



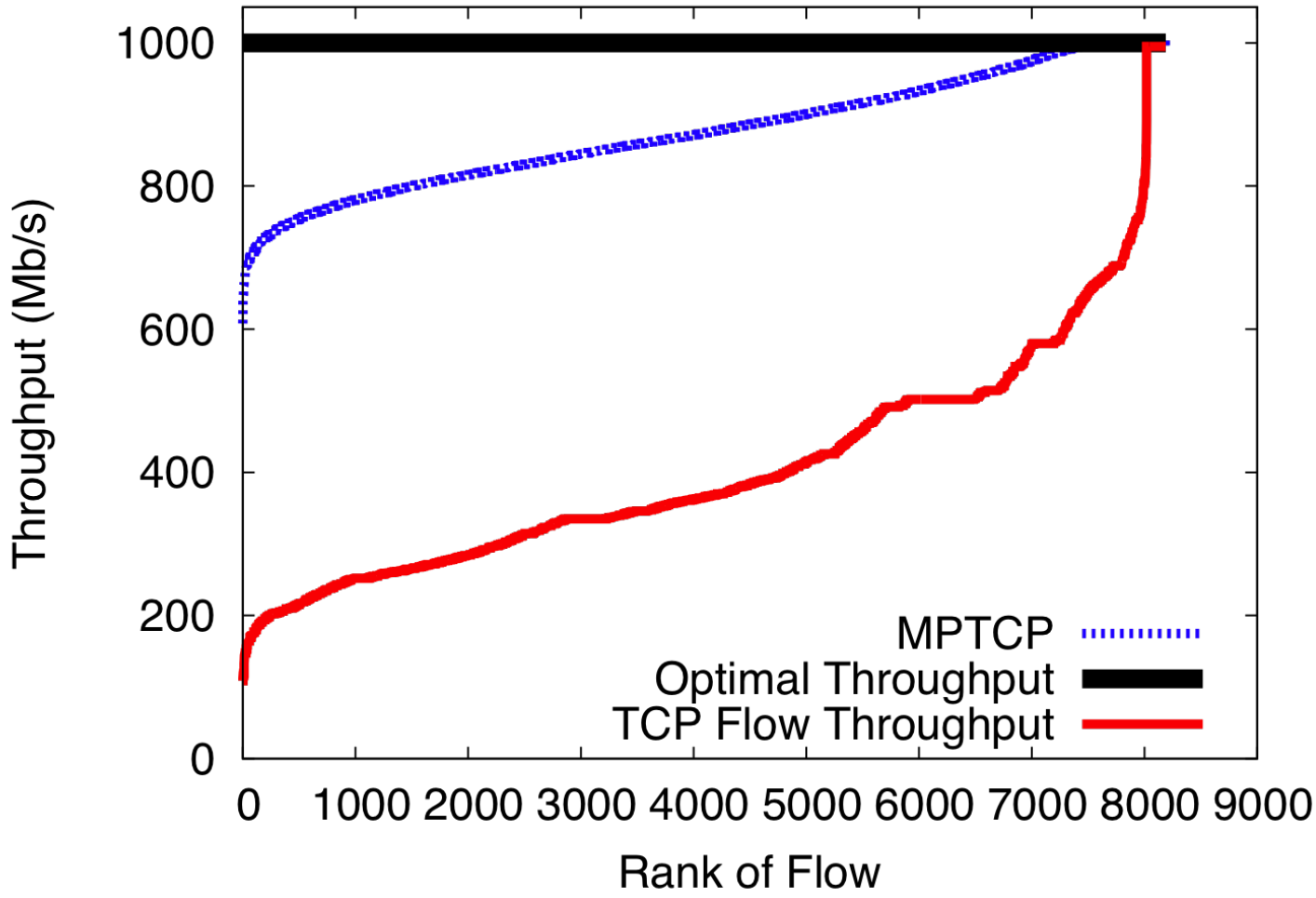
MPTCP Operation



Multipath TCP: Congestion Control [NSDI, 2011]



MPTCP better utilizes the FatTree network



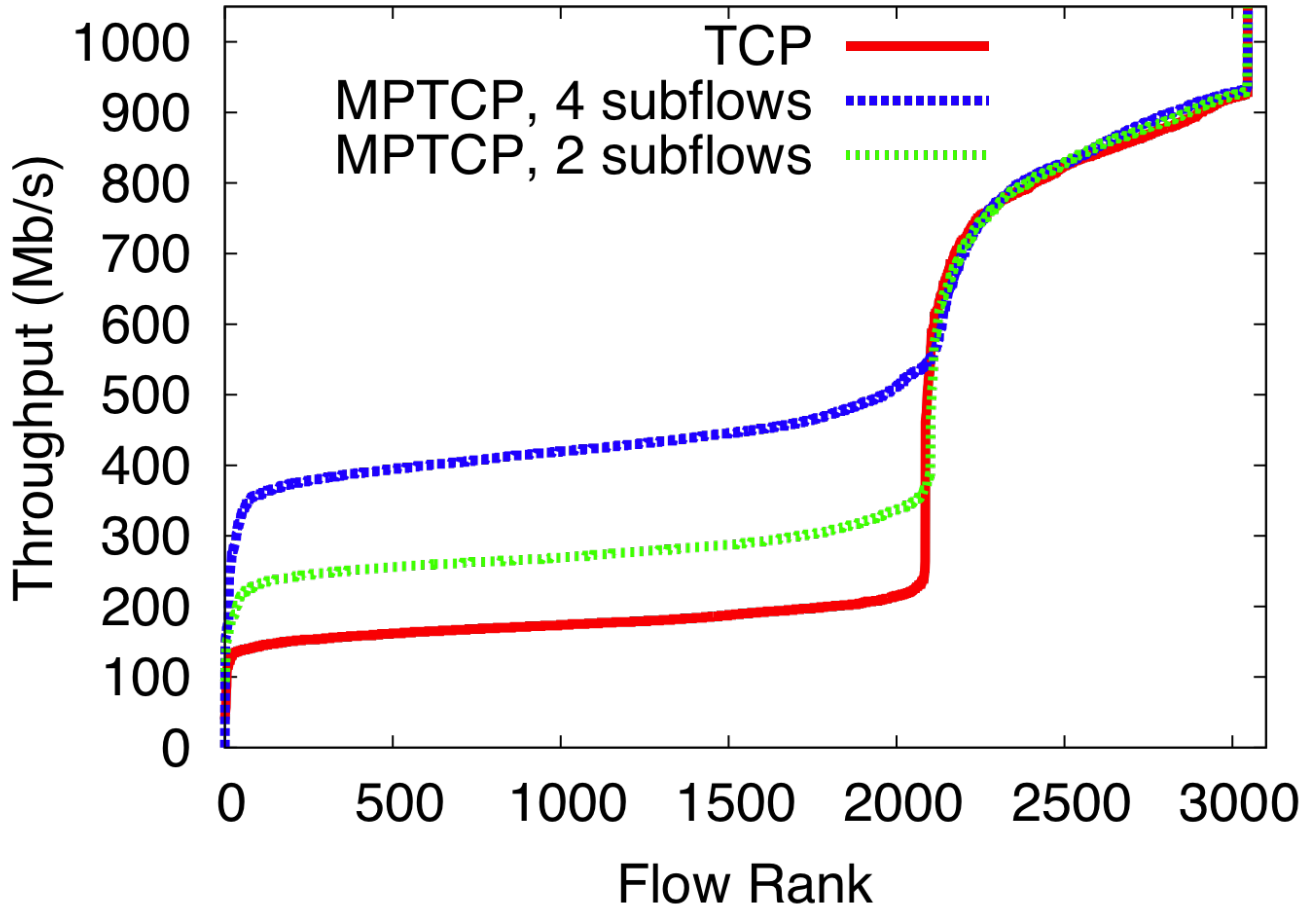
MPTCP on EC2

- Amazon EC2: infrastructure as a service
  - We can borrow virtual machines by the hour
  - These run in Amazon data centers worldwide
  - We can boot our own kernel
- A few availability zones have multipath topologies
  - 2-8 paths available between hosts not on the same machine or in the same rack
  - Available via ECMP

Amazon EC2 Experiment

- 40 medium CPU instances running MPTCP
- For 12 hours, we sequentially ran all-to-all *iperf* cycling through:
  - TCP
  - MPTCP (2 and 4 subflows)

MPTCP improves performance on EC2



Where do MPTCP's benefits come from?

Allocating Flows to Paths

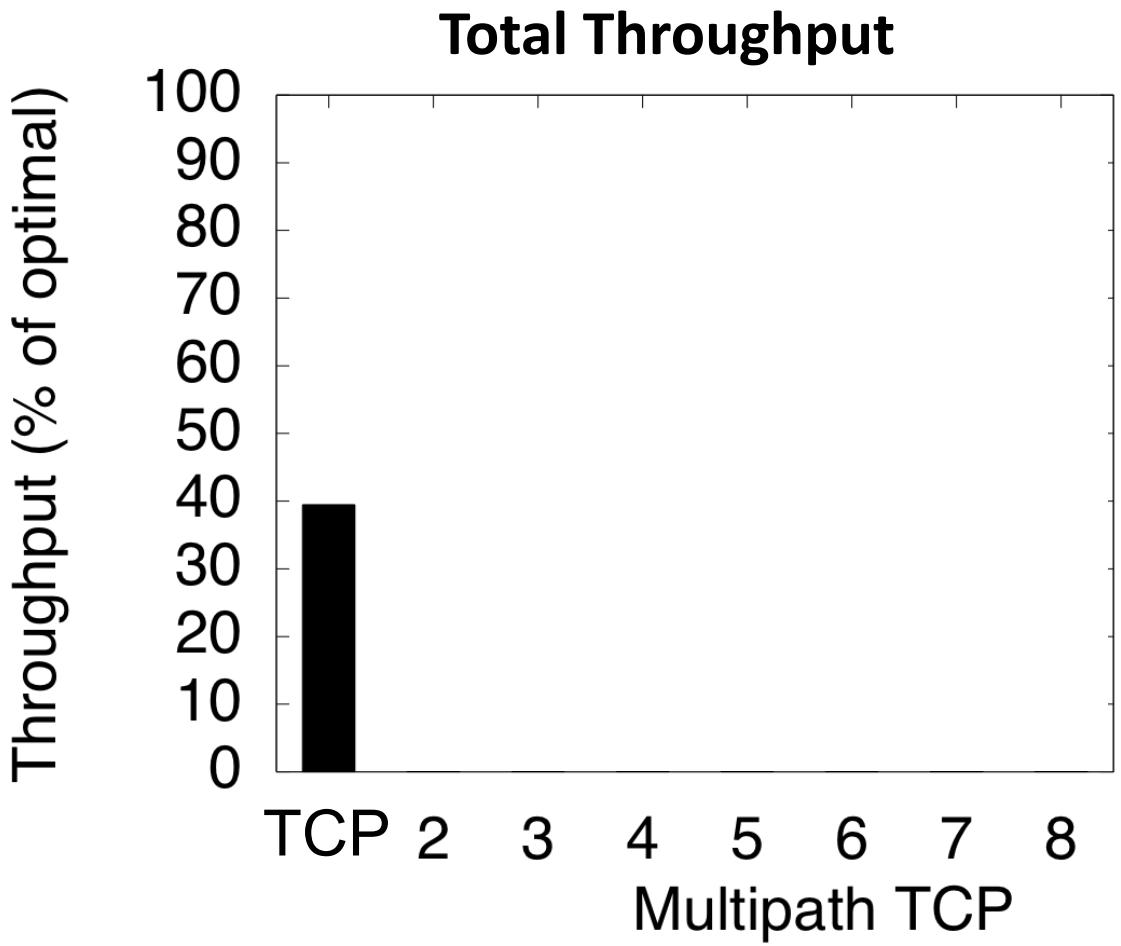
- Multi-Commodity Flow Problem
  - Single path forwarding
    - Expressed as Binary Integer Programming
    - NP-Complete
    - Solvers give exact solution but are impractical for large networks

Allocating Flows to Paths

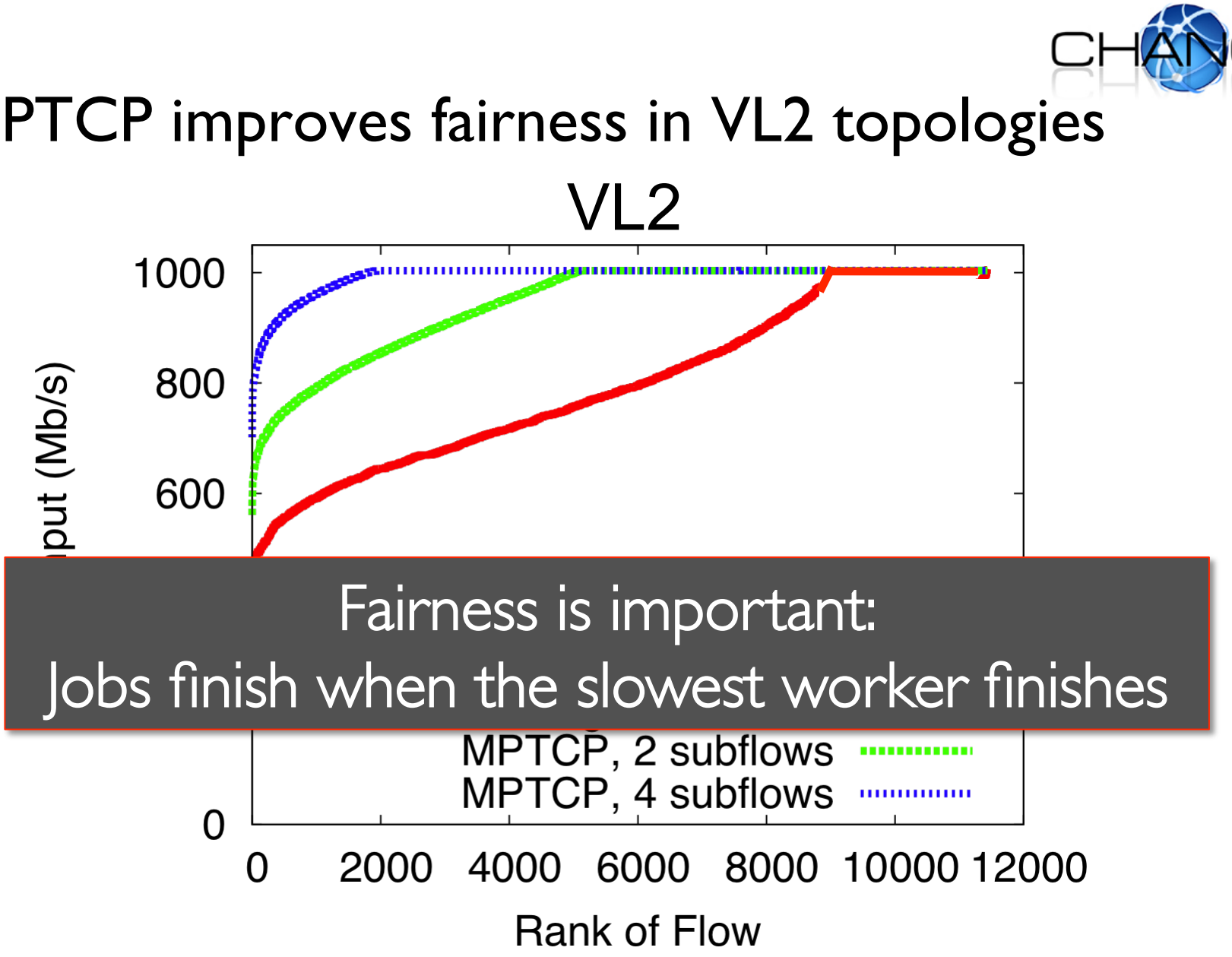
- Multi-Commodity Flow Problem
  - Single path forwarding
    - Expressed as Binary Integer Programming
    - NP-Complete
    - Solvers give exact solution but are impractical for large networks
  - Multipath forwarding
    - Expressed as Linear Programming problem
    - Solvable in **polynomial** time

- How many subflows are needed?
- How does the topology affect results?
- How does the traffic matrix affect results?

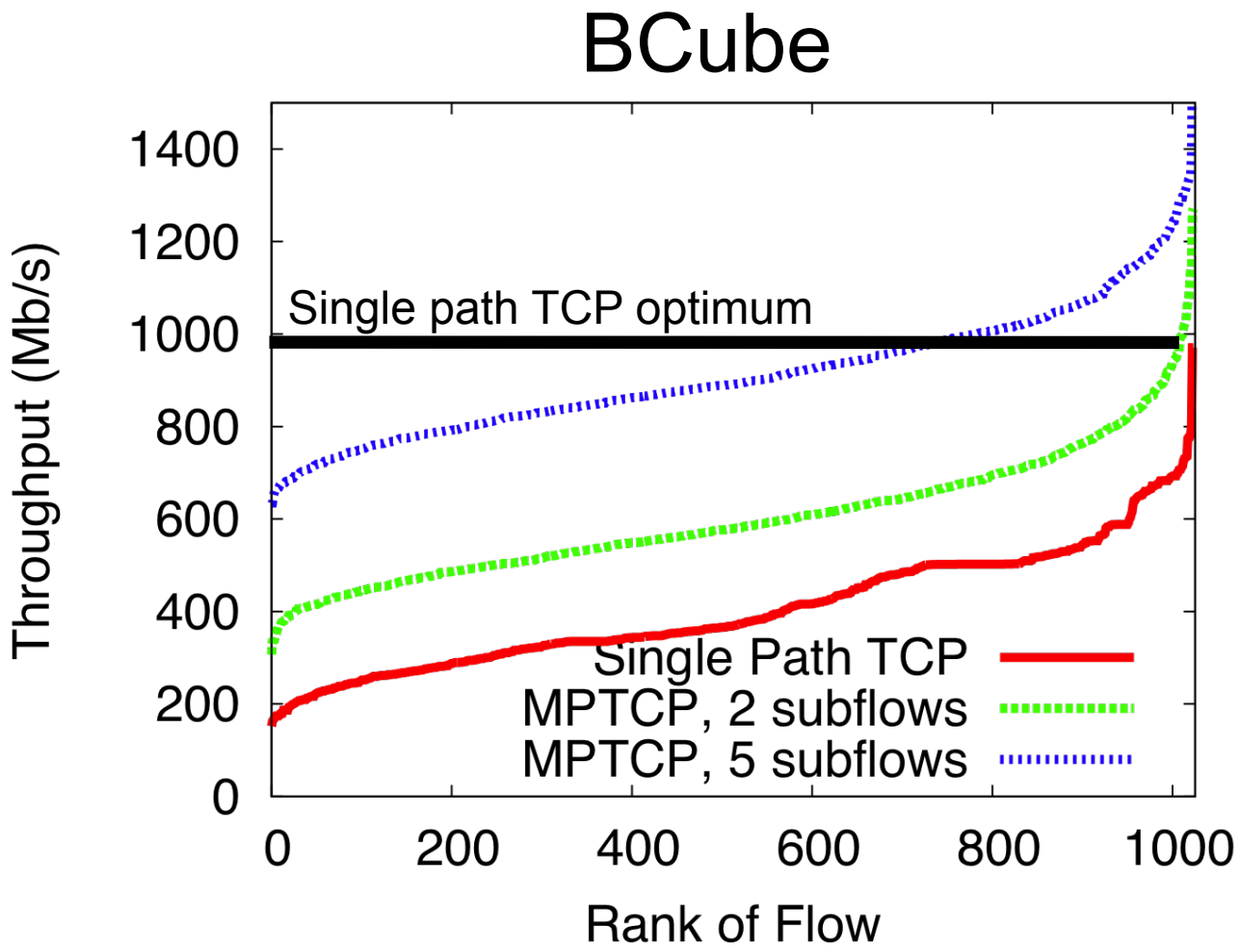
At most 8 subflows are needed



MPTCP improves fairness in VL2 topologies



MPTCP improves throughput and fairness in BCube

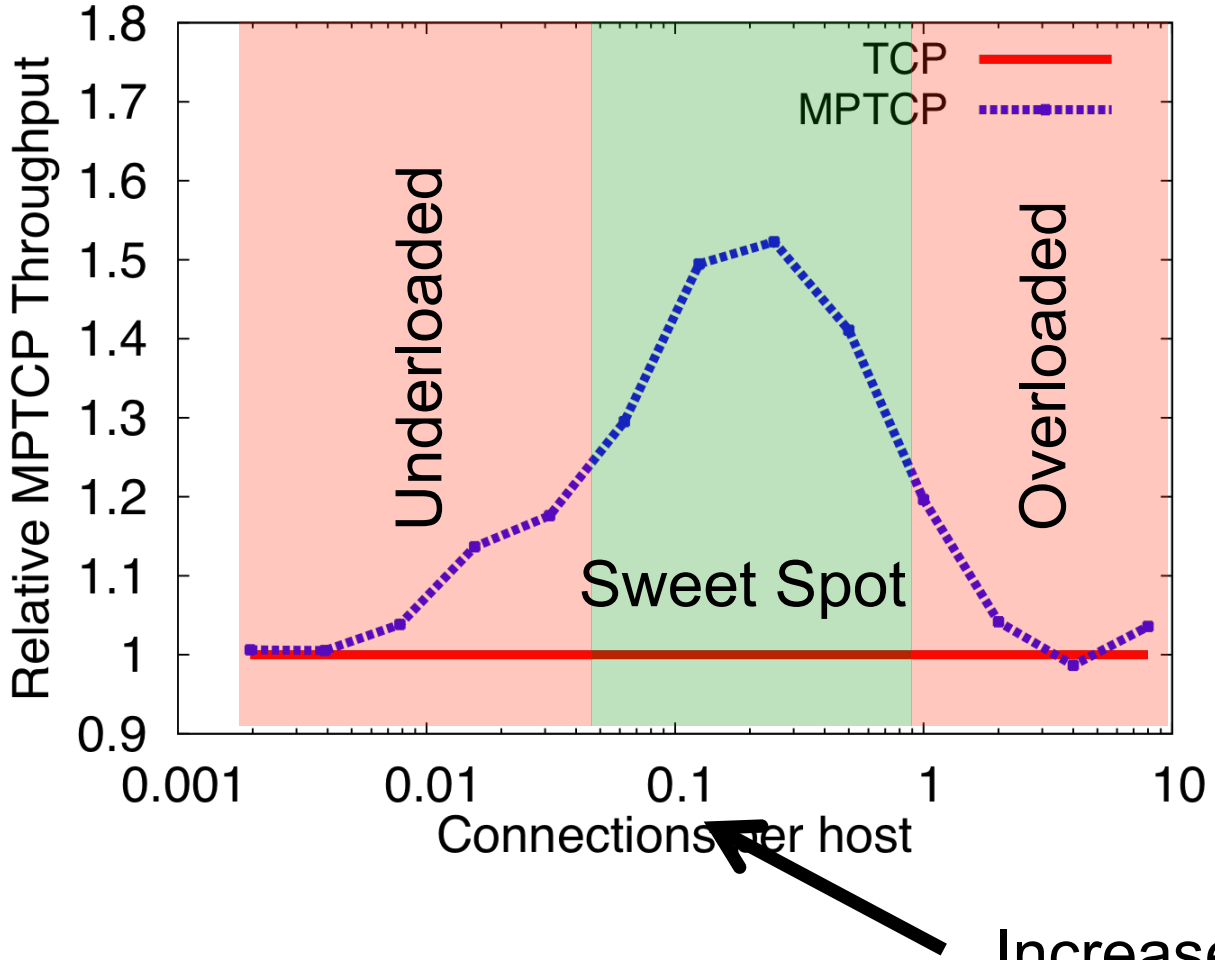


Oversubscribed Topologies

- To saturate full bisectional bandwidth:
  - There must be no traffic locality
  - All hosts must send at the same time
  - Host links must not be bottlenecks

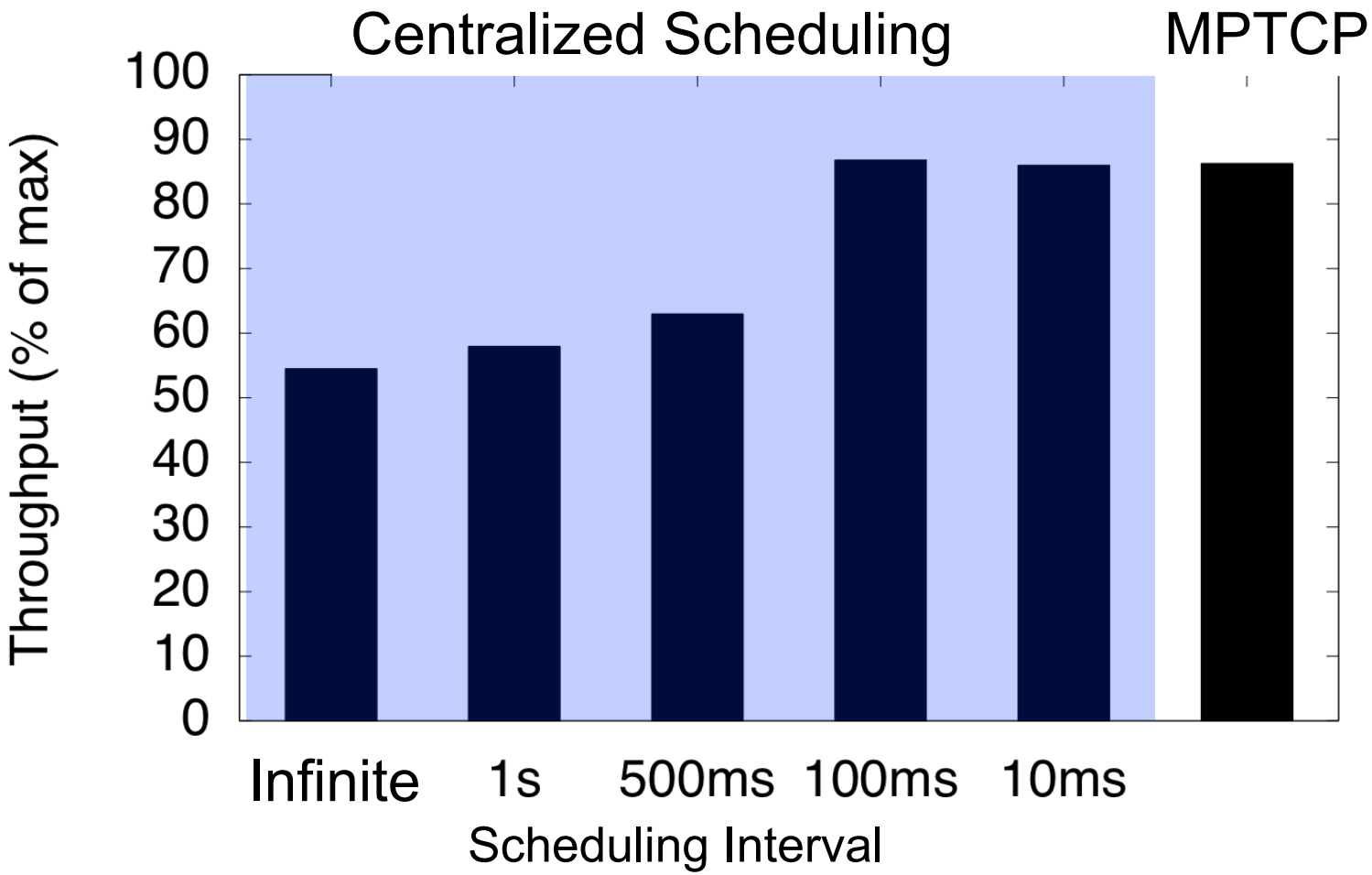
- It makes sense to under-provision the network core
  - This is what happens in practice
  - Does MPTCP still provide benefits?

Performance improvements depend on traffic matrix

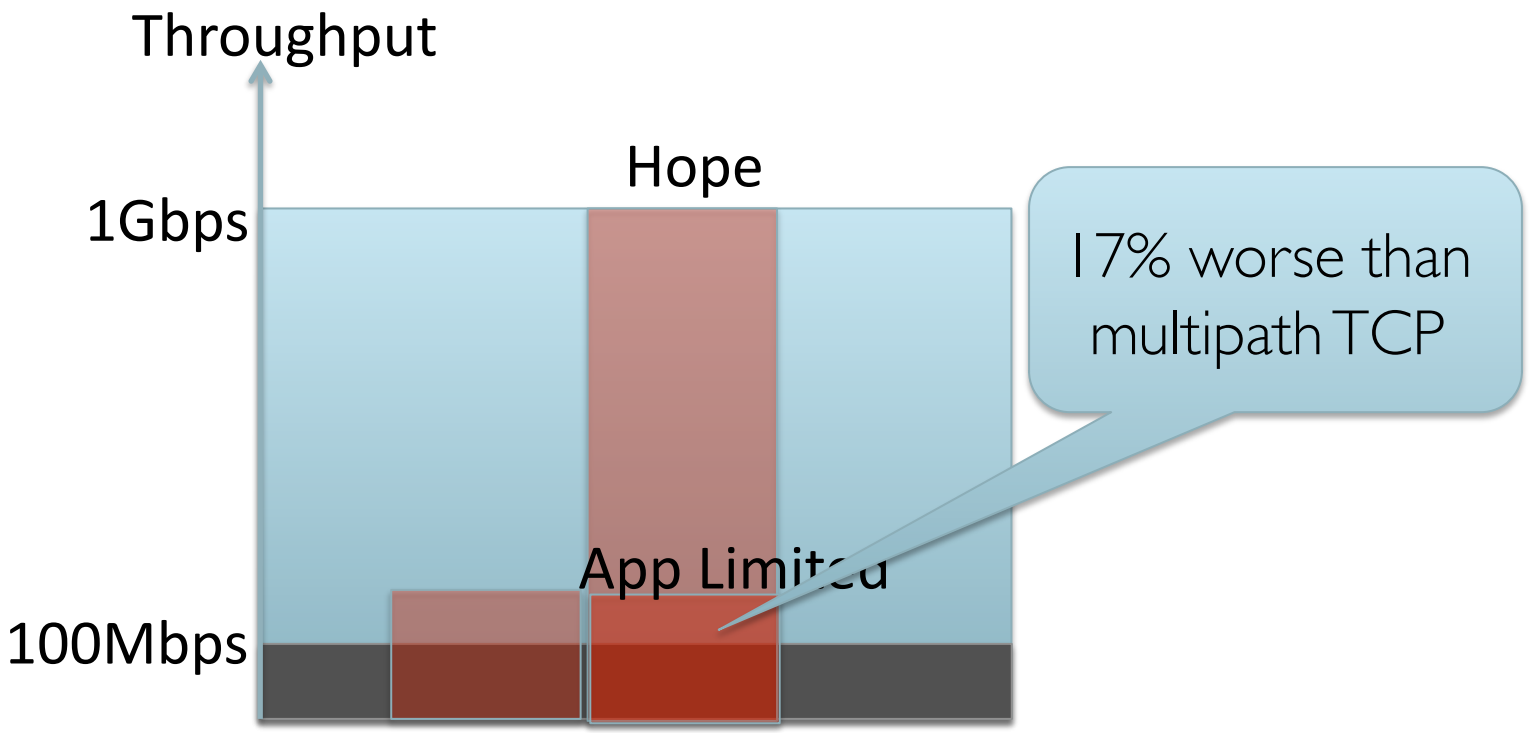


MPTCP vs. Centralized Scheduling

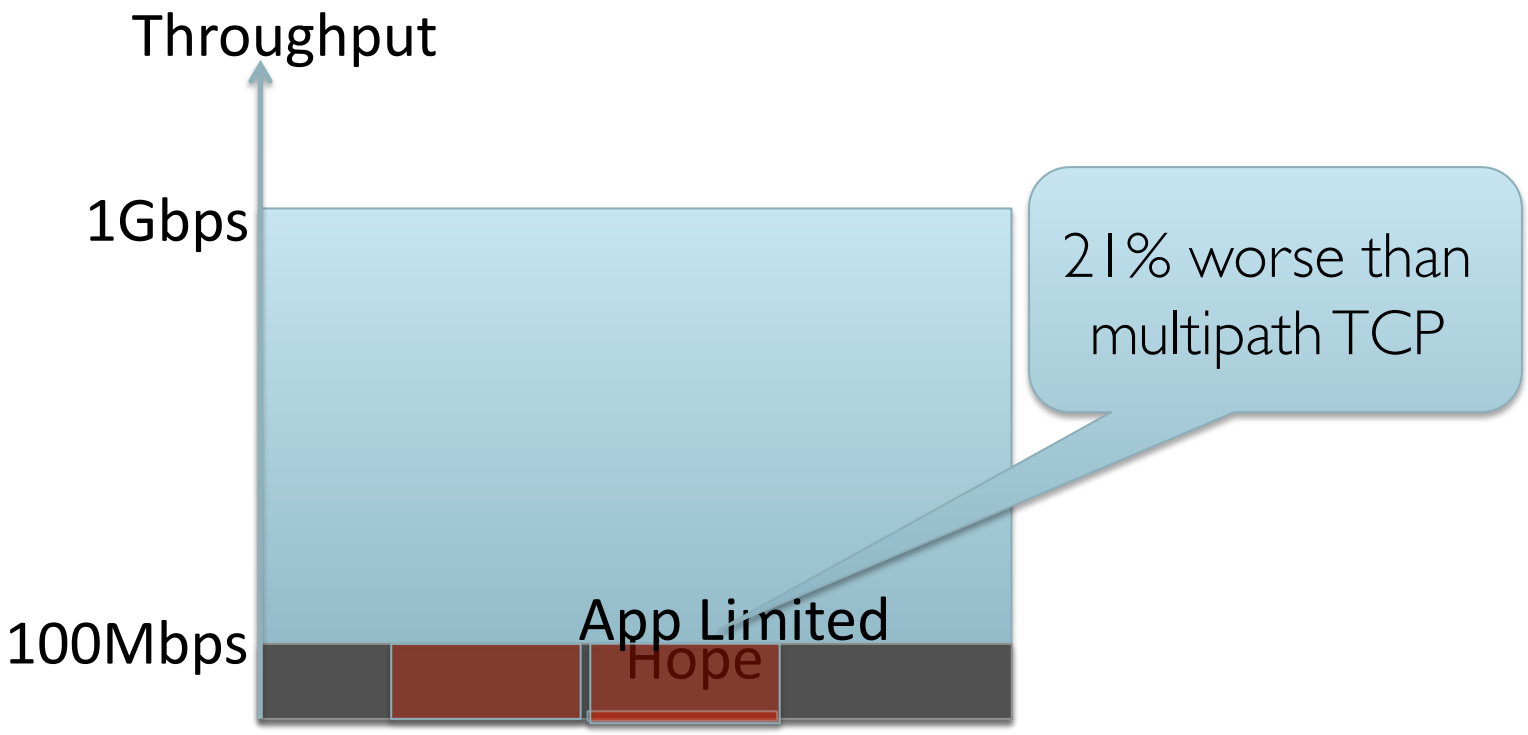
MPTCP vs Hedera First Fit



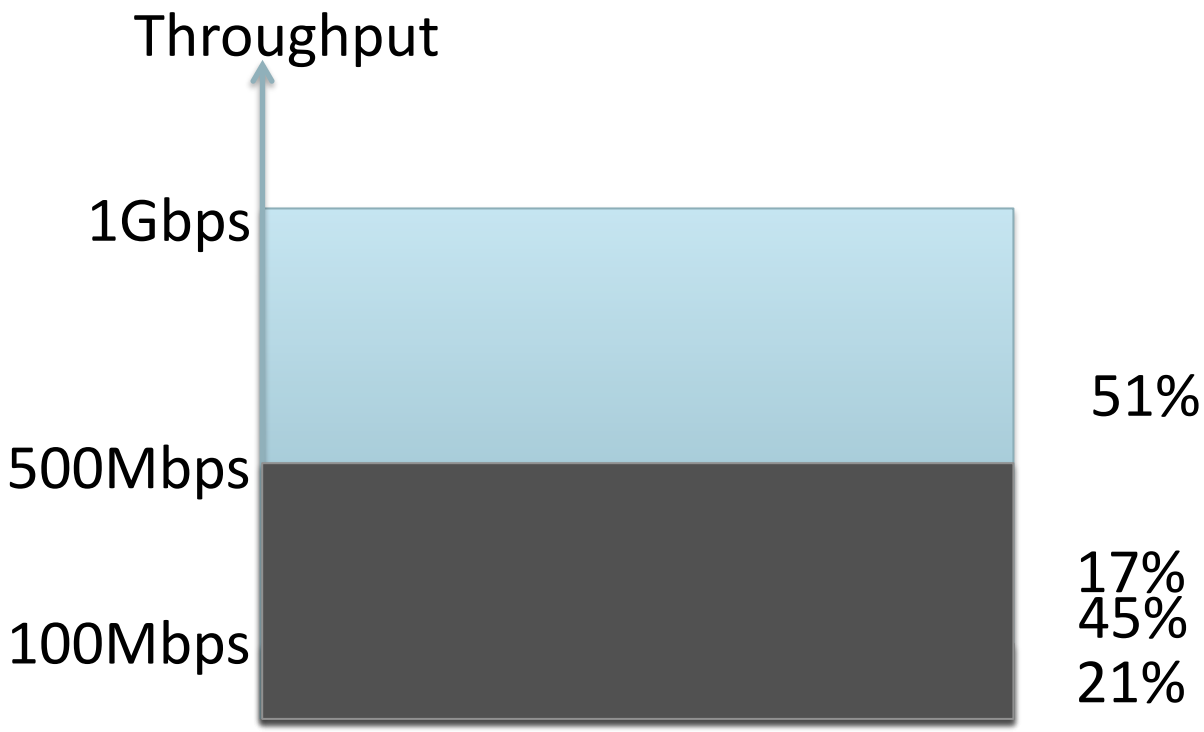
Centralized Scheduling: Setting the Threshold



Centralized Scheduling: Setting the Threshold



Centralized Scheduling: Setting the Threshold



MPTCP vs. Hedera

	MPTCP	HEDERA
Implementation	Distributed	Centralized
Network changes	No	Yes, upgrade all switches to OF
Hardware needed	No	Centralized Scheduler
Software changes	Yes – host stack	No
Scope	Schedules more flows	Large flows only
Convergence Time	Scale Invariant, RTTs	Tight Control Loop Limits Scalability
Fairness	Fair	Less fair

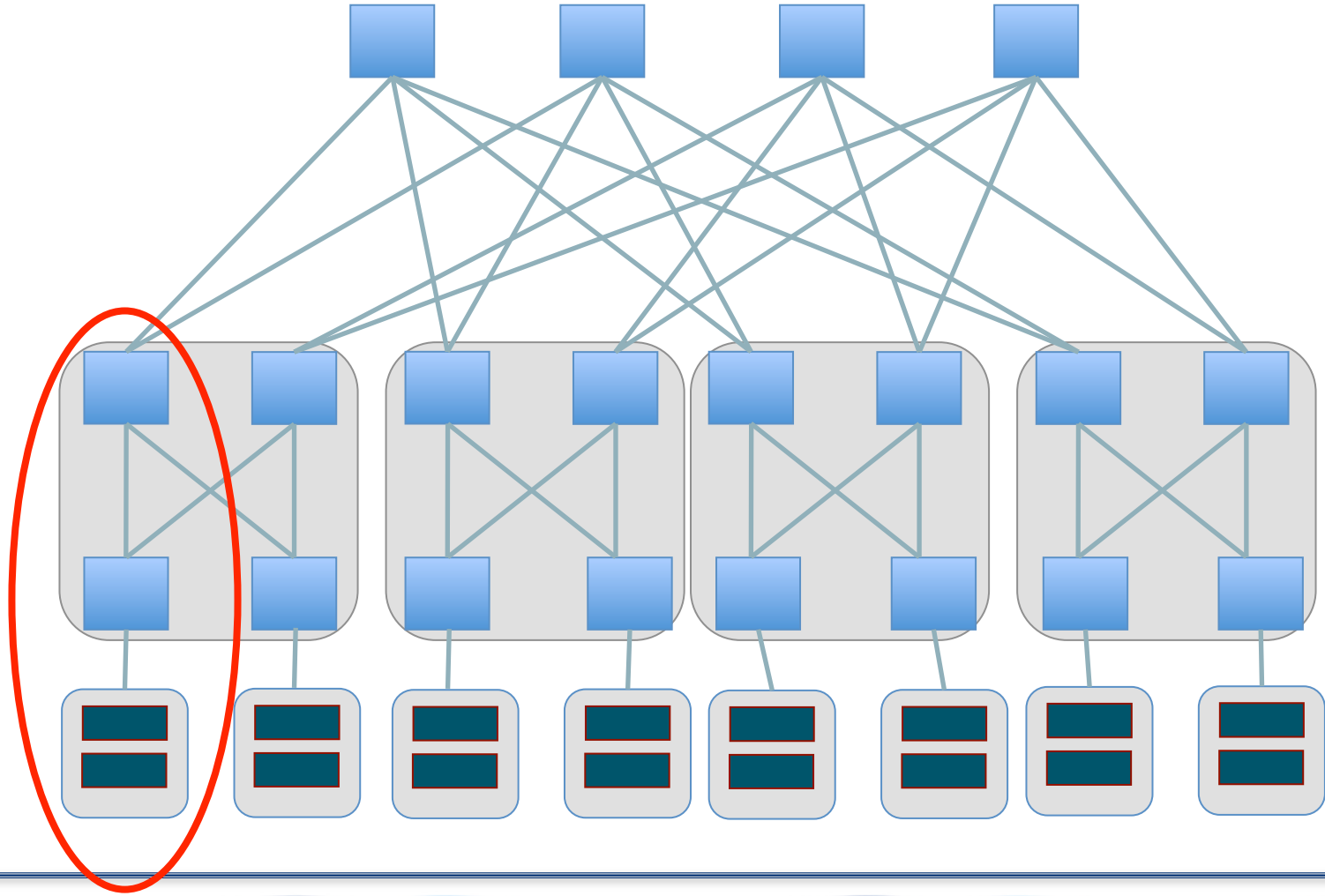
What is an optimal datacenter topology for multipath transport?

In single homed topologies:

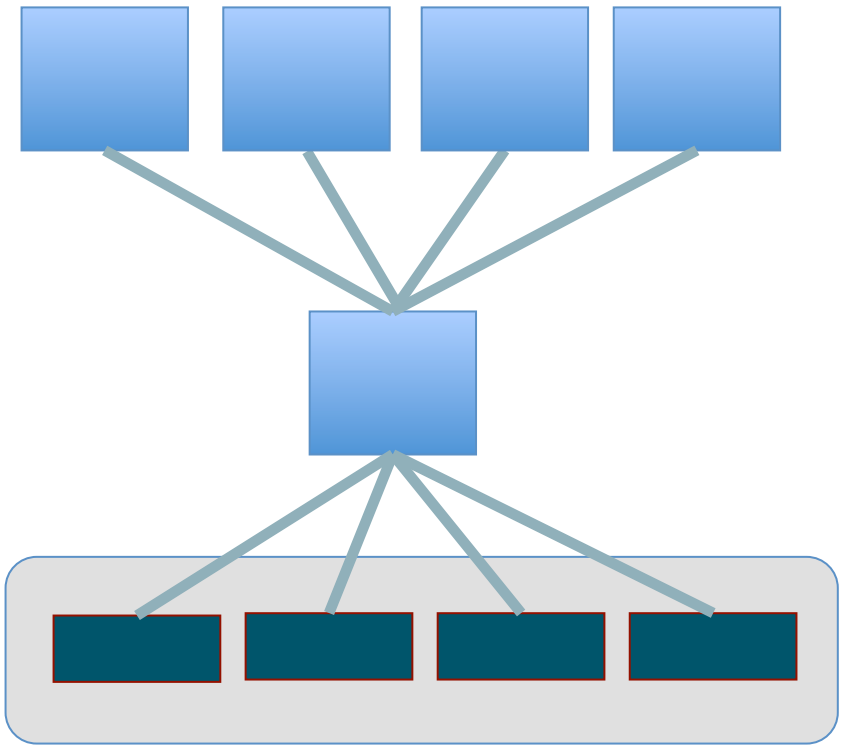
- Hosts links are often bottlenecks
- ToR switch failures wipe out tens of hosts for days

Multi-homing servers is the obvious way forward

Fat Tree Topology



Fat Tree Topology

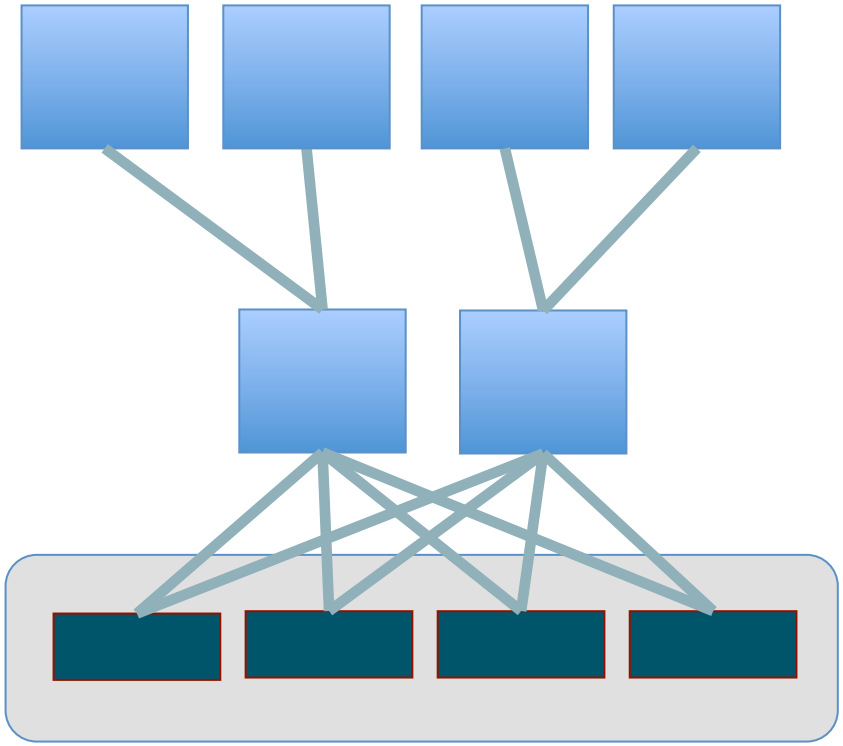


Upper Pod Switch

ToR Switch

Servers

Dual Homed Fat Tree Topology



Upper Pod Switch

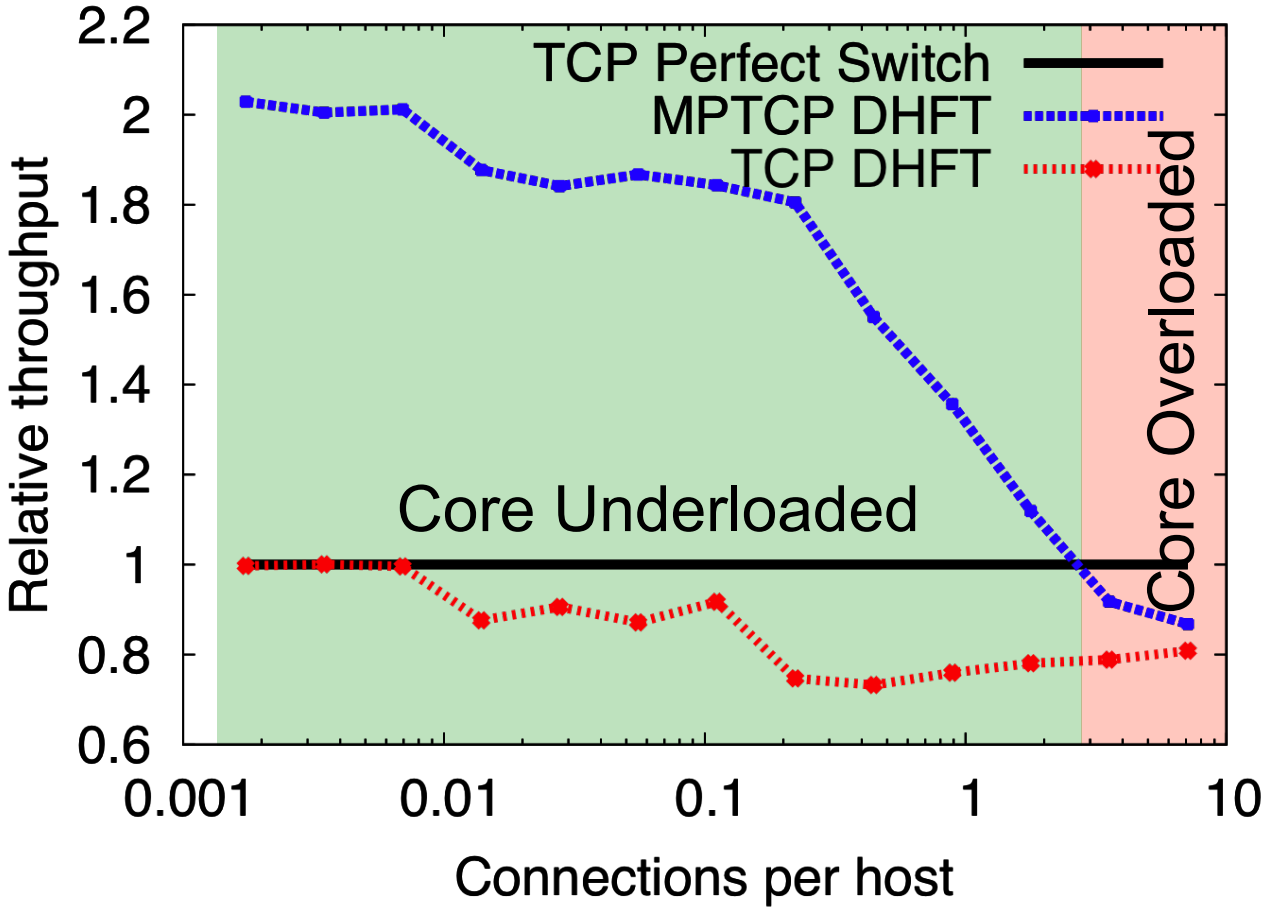
ToR Switch

Servers

Is DHFT any better than Fat Tree?

- Not for traffic matrices that fully utilize the core
- Let's examine random traffic patterns

DHFT provides significant improvements when core is not overloaded



Summary

- *"One flow, one path"* thinking has constrained datacenter design
  - Collisions, unfairness, limited utilization
  - Fixing these is possible, but does not address the **bigger issue**
- Multipath transport enables resource pooling in datacenter networks:
  - Improves throughput
  - Improves fairness
  - Improves robustness
- *"One flow, many paths"* frees designers to consider topologies that offer improved performance for similar cost

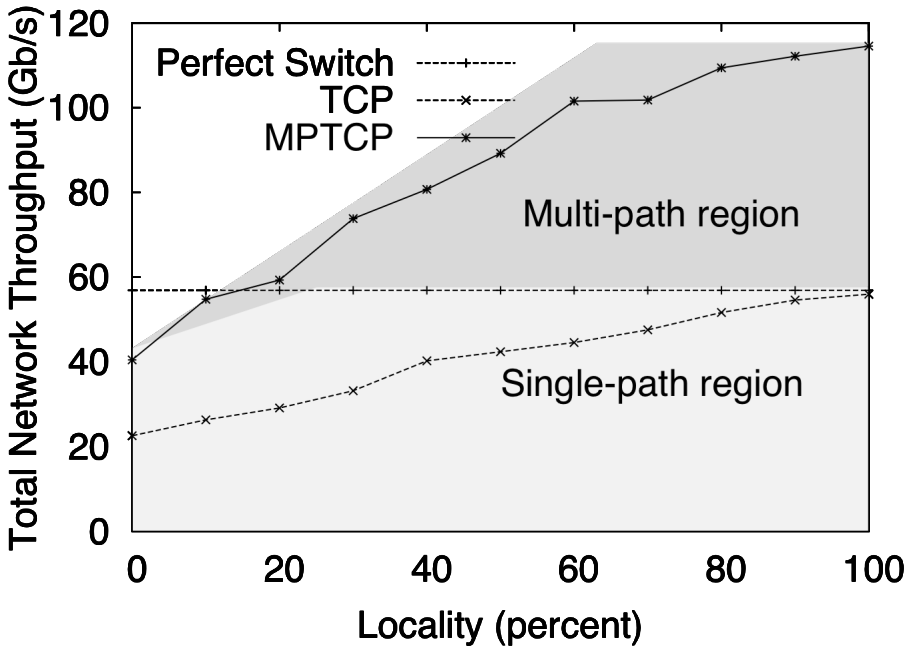
Backup Slides

Effect of MPTCP on short flows

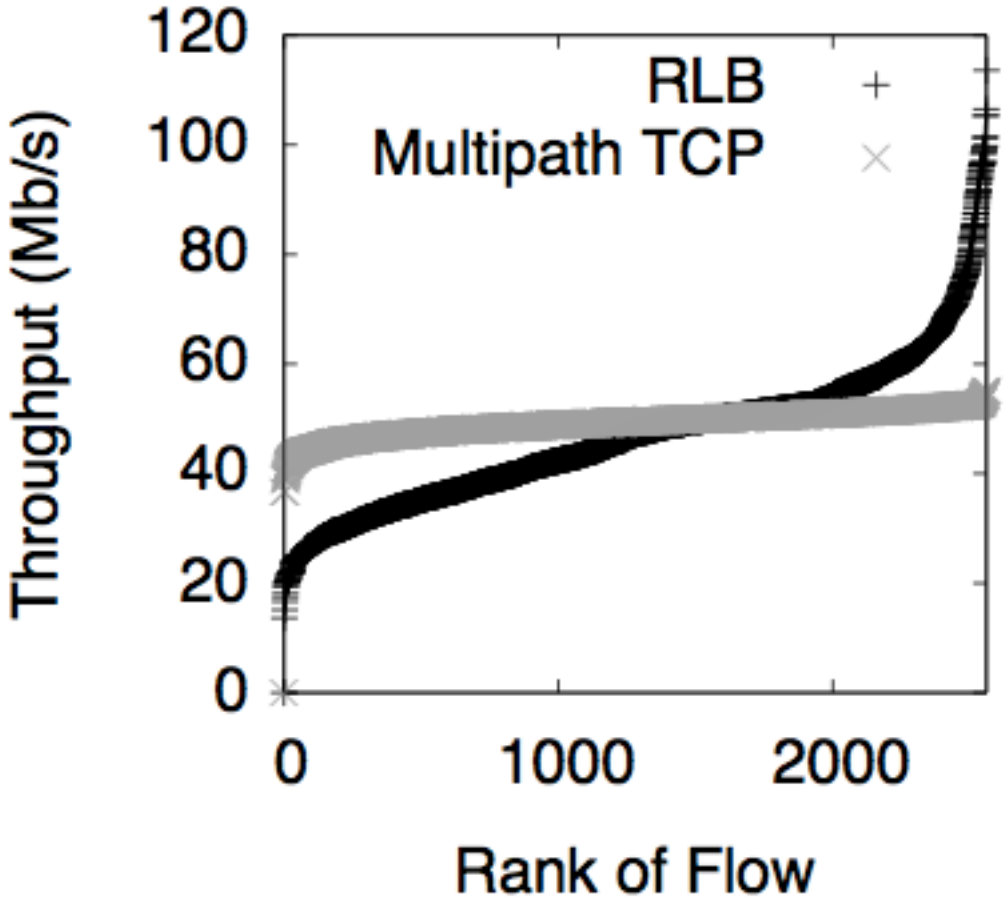
- Flow sizes from VL2 dataset
- MPTCP enabled for long flows only (timer)
- Oversubscribed Fat Tree topology
- Results:

	TCP/ECMP	MPTCP
Completion time:	79ms	97ms
Core Utilization:	25%	65%

Effect of Locality in the Dual Homed Fat Tree



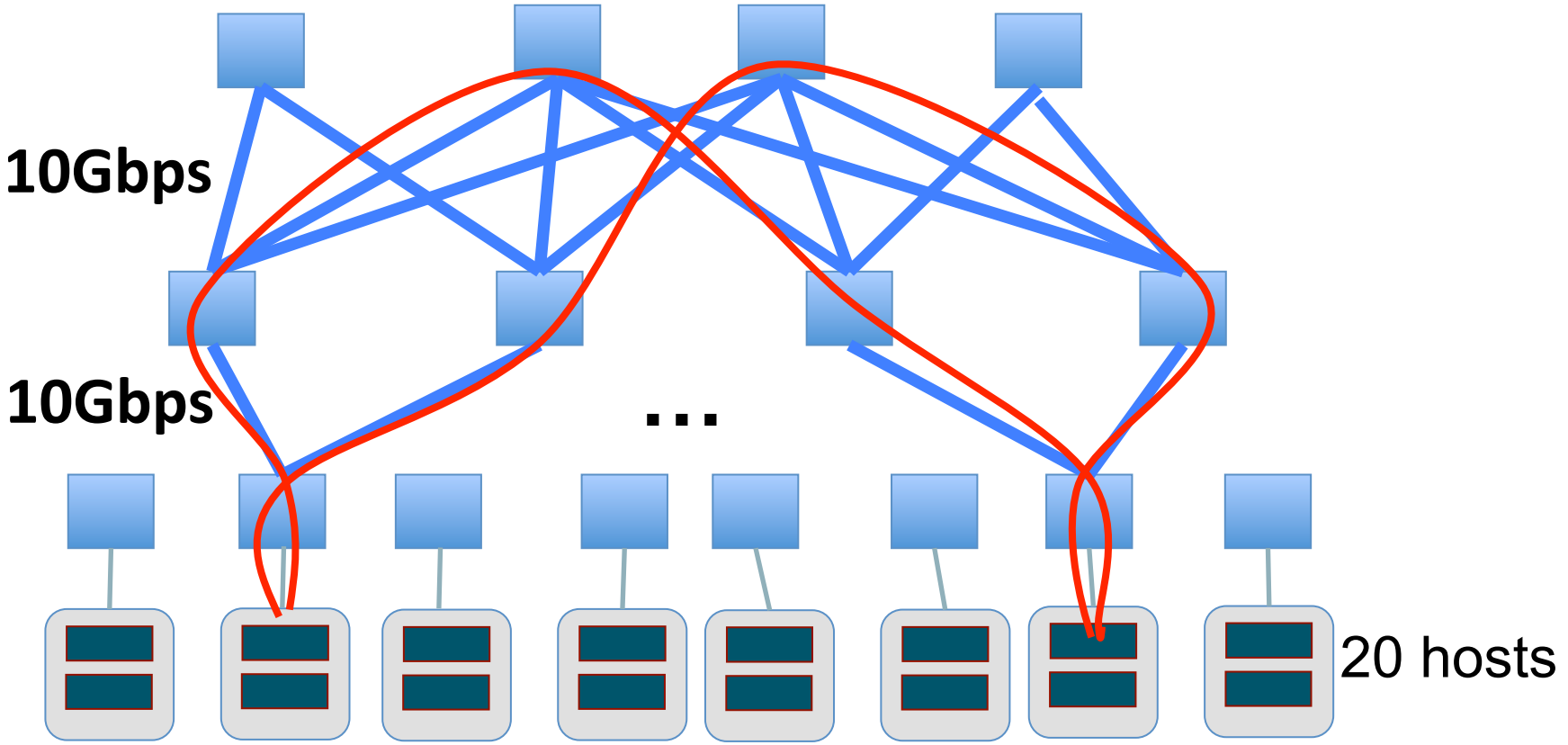
Overloaded Fat Tree: better fairness with Multipath TCP




CHANGE


CHANGE

VL2 Topology [Greenberg et al, 2009, Clos topology]



BCube Topology [Guo et al, 2009]