CSCI 381/780
Cloud Computing

Data Center Networking

Jun Li
Queens College

Building a scalable data center network with commodity devices

Data center networking

- Ever increasing scale
  - Google has 2.5 million servers in 2021
  - Microsoft doubles its number of servers in 14 months
- The expansion rate exceeds Moore’s Law

Data center networking

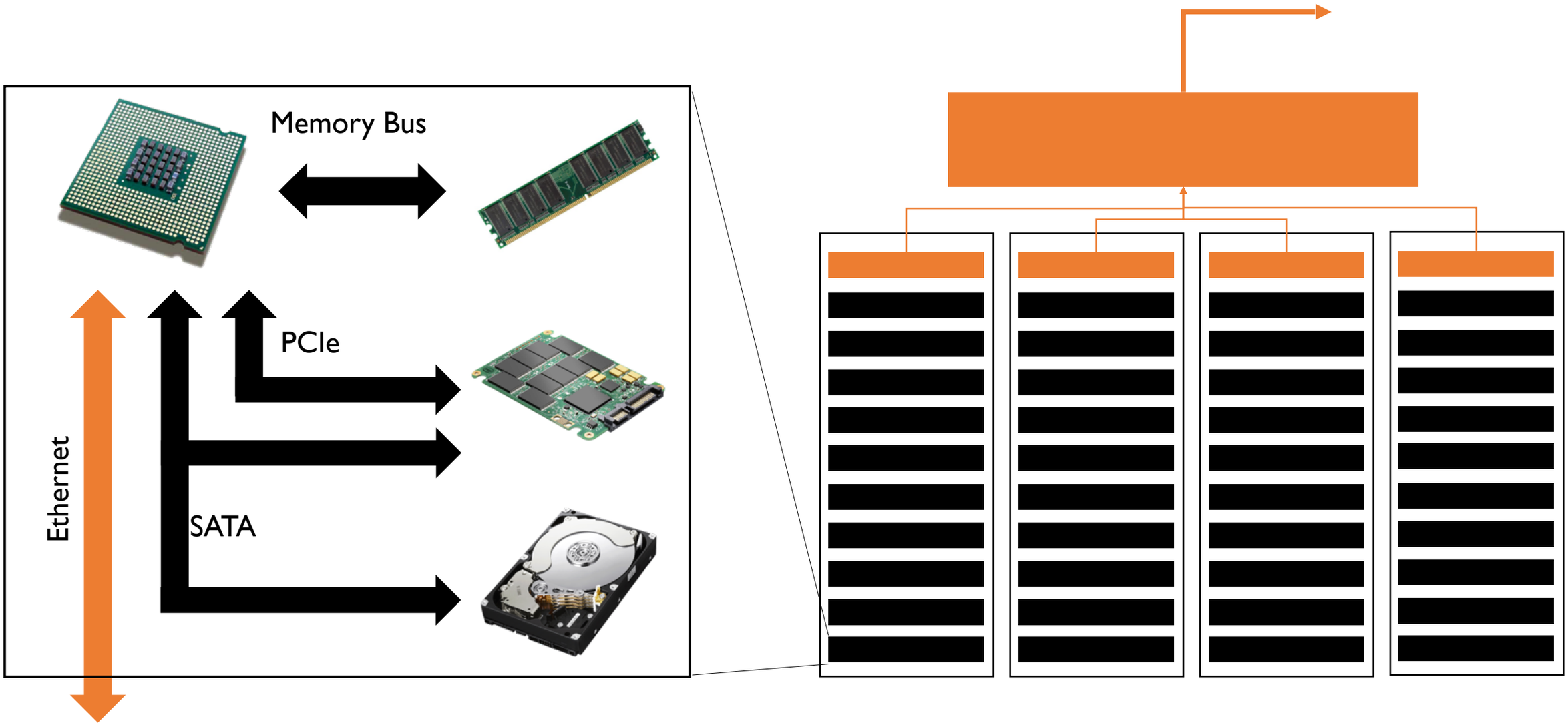
- Network capacity: Bandwidth hungry data-centric applications
  - Data shuffling in MapReduce/Spark
  - Data replication/re-replication in distributed file systems
  - Disseminating virtual machines

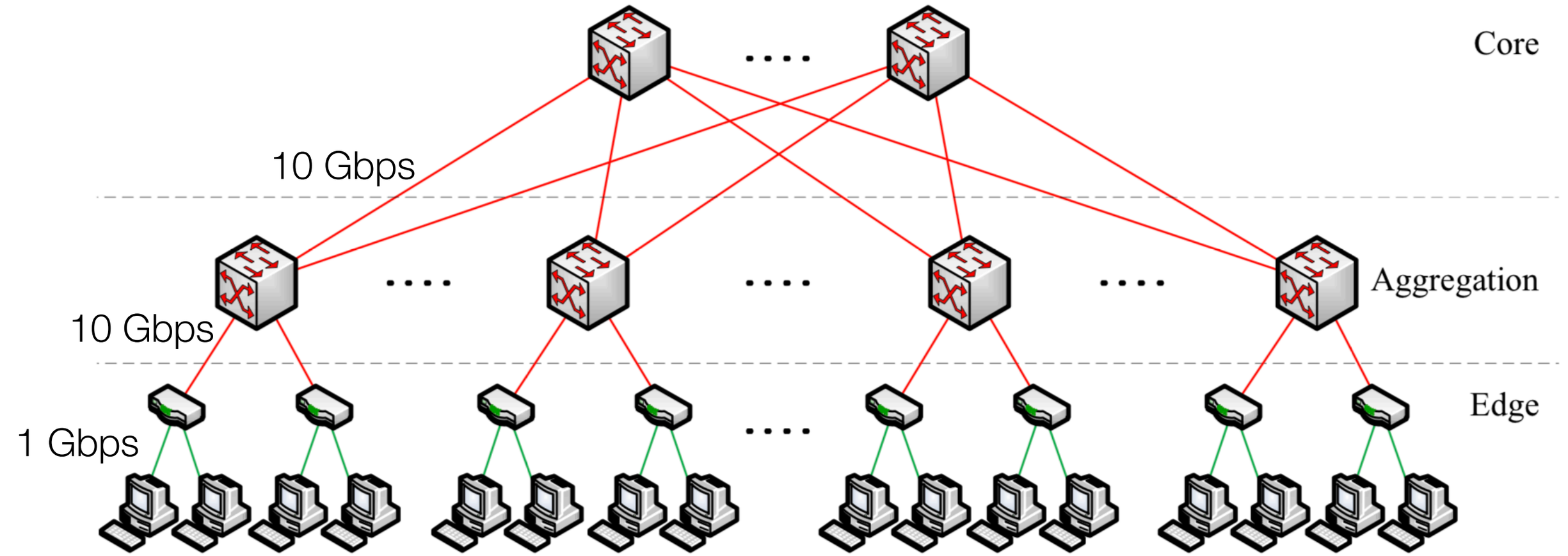
Data center networking

- Cost: Using high-end switches/routers to scale up is costly

- Fault-tolerance: When data centers scale, failures become the norm

Data center network





a traditional data center network topology

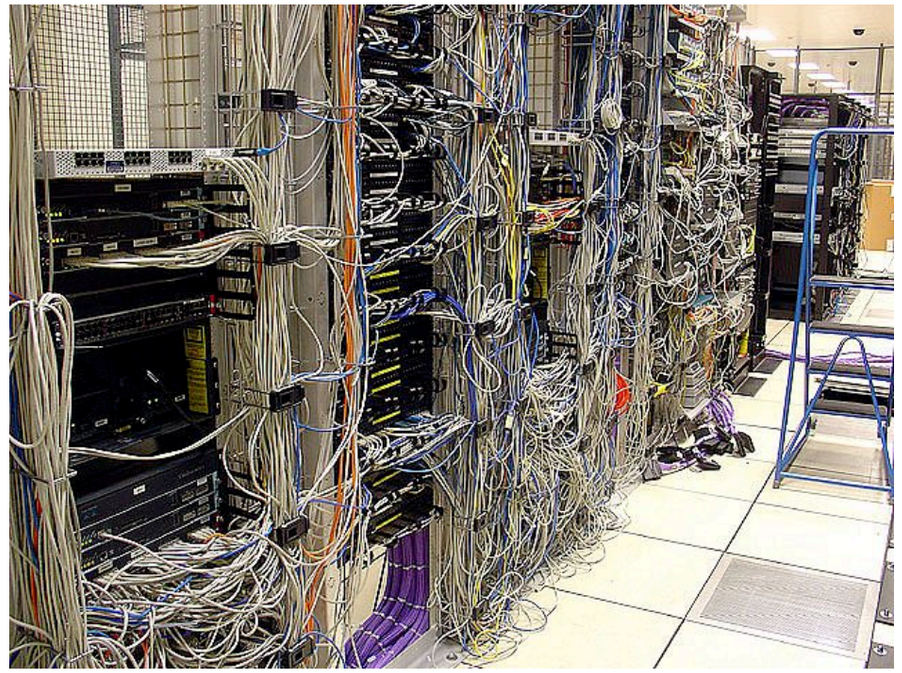
Topology

- 2 layers: 5K to 8K hosts
- 3 layer: >25K hosts
- Switches:
  - Leaves: have N GigE ports (48-288) + N 10 GigE uplinks to one or more layers of network elements
  - Higher levels: N 10 GigE ports (32-128)

Cost

- Edge: $7,000 for each 48-port GigE switch (in 2008)
- Aggregation and core: $700,000 for 128-port 10GigE switches

Year	10 GigE	Hosts	Cost/ GigE
2002	28-port	4,480	$25.3K
2004	32-port	7,680	$4.4K
2006	64-port	10,240	$2.1K
2008	128-port	20,480	$1.8K



- Cabling costs are not considered!

SHOP BY

CATEGORY

Cisco Nexus 2000 Switches (8)
Cisco Nexus 3000 Switches (54)
Cisco Nexus 5000 Switches (24)
Cisco Nexus 7000 Switches (41)
Cisco Nexus 9000 Switches (70)
Huawei CloudEngine 12800 Data Center Switches (60)
Huawei CloudEngine 16800 Data Center Switches (16)
Huawei CloudEngine 5800 Data Center Switches (34)
Huawei CloudEngine 6800 Data Center Switches (47)
Huawei CloudEngine 7800 Data Center Switches (3)
Huawei CloudEngine 8800 Data Center Switches (9)
Dell Networking S4048-ON Switches (2)
D-Link Data Center Switches (8)
Dell EMC PowerSwitch S series 1GbE switches (5)
Extreme Data Center Switches (1)

PRICE

US$0.00 US$91,584.99

382 products OK

IN STOCK

Yes (19)

Data Center Switches

382 Products found **SORT BY:** Sold Quantity **SHOW** 20 View as

N3K-C3548P-10GX

4.9/5.0 22 Reviews

Condition: Brand New Sealed

Nexus 3548-X Switch 48 SFP+ ports, Enhanced

~~US$28,573.00~~

US$13,535.00 (53% OFF)

Quote | Help

Wishlist Compare

N5K-C5672UP

5/5.0 22 Reviews

Condition: Brand New Sealed

Nexus 5672UP Chassis, 1RU, 32 p 10-Gbps SFP+, 16 Unified Ports, 6p 40G QSFP+

~~US$47,622.00~~

US$15,540.00 (67% OFF)

Quote | Help

Wishlist Compare

N5K-C56128P

5/5.0 19 Reviews

Condition: Brand New Sealed

Nexus 56128P Chassis, 2RU, 48x 10-Gbps SFP+, 4 x 40G QSFP+ Fixed Ports (Base)

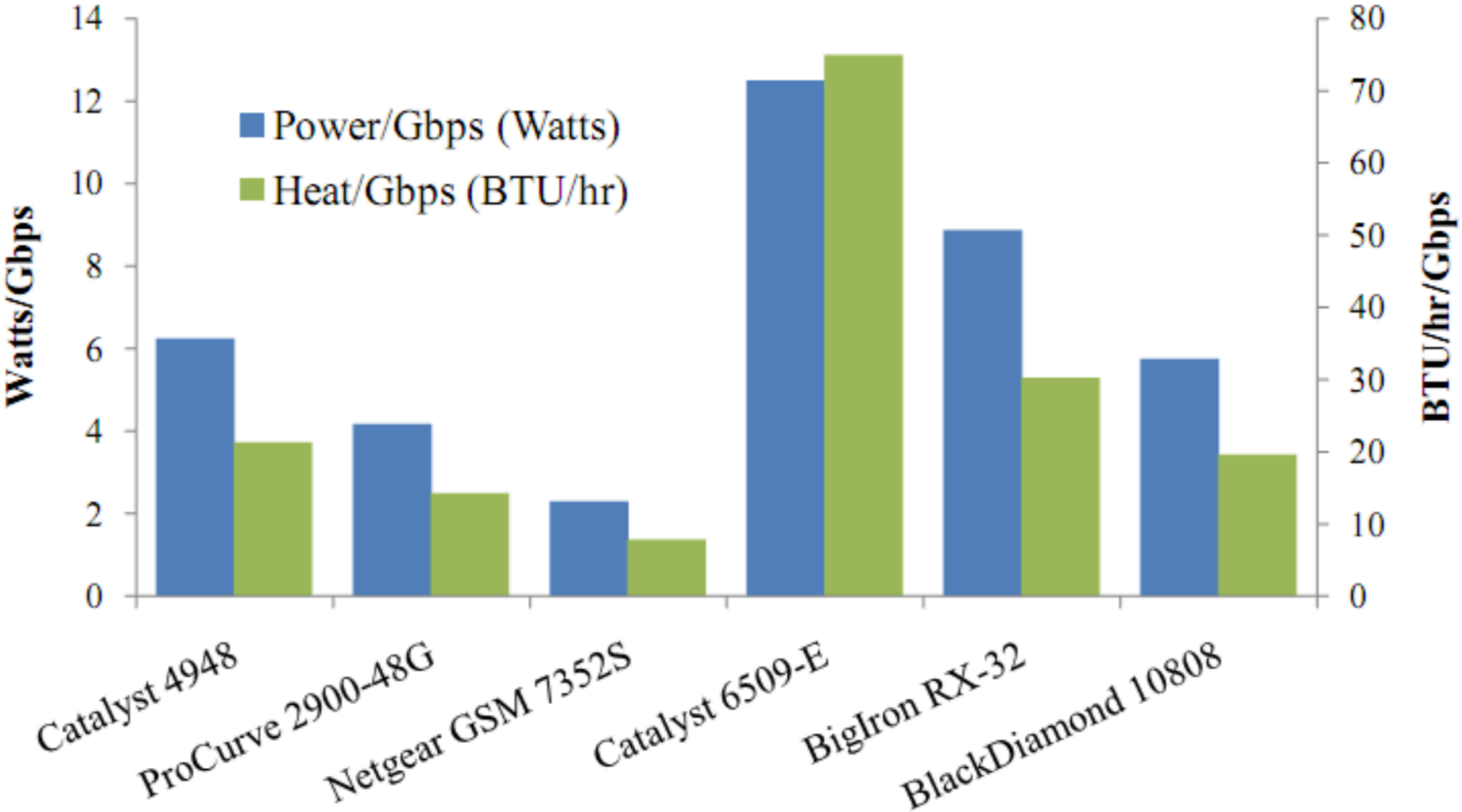
~~US$37,695.00~~

US$15,872.00 (58% OFF)

Quote | Help

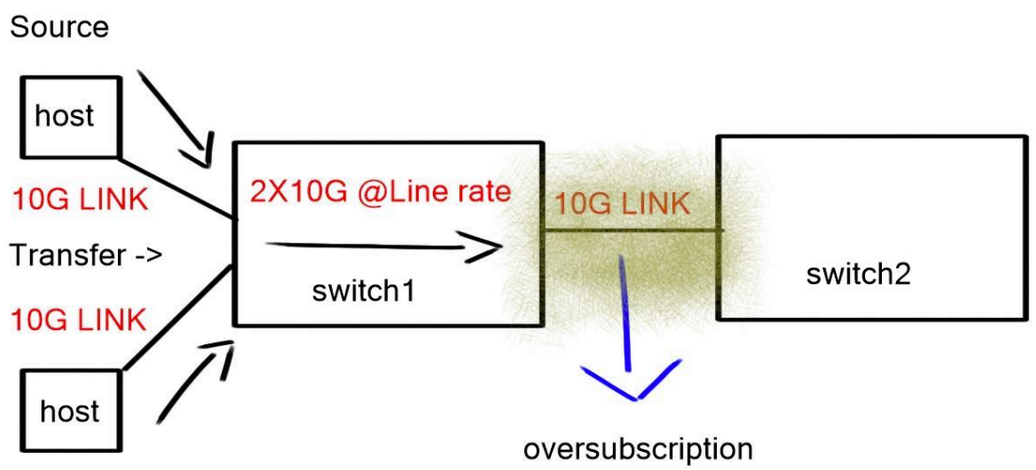
Wishlist Compare

Cost of maintaining switches



Oversubscription

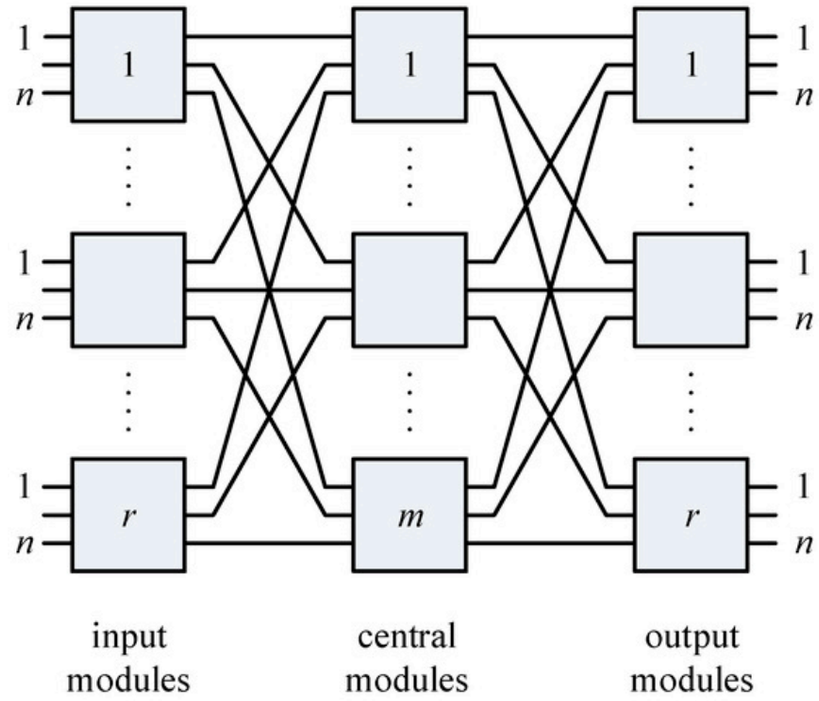
- Ratio of the worst-case achievable aggregate bandwidth among the end hosts to the total bisection bandwidth of a particular communication topology
- Lower the total cost of the design
- Typical designs: factor of 2.5:1 (400 Mbps) to 8:1(125 Mbps)

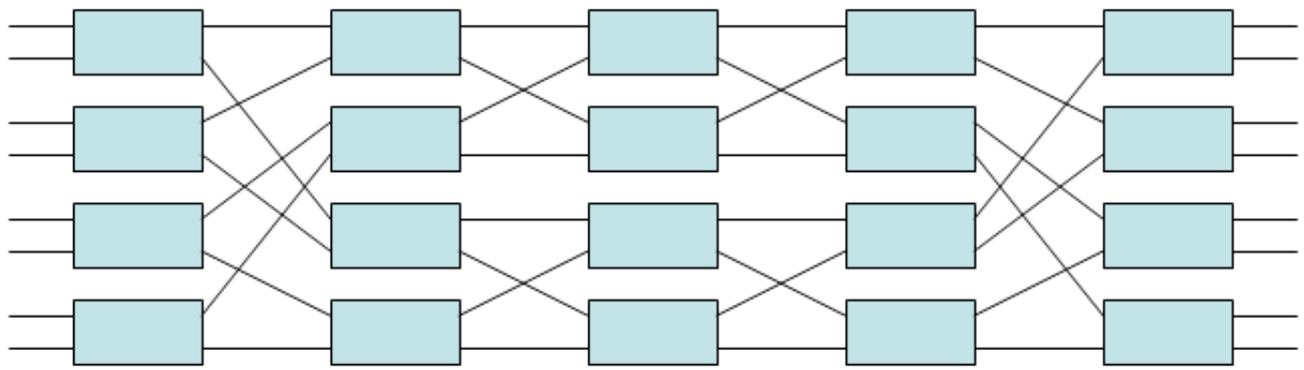


Clos network (Fat-tree)

- Adopt a special instance of a Clos network

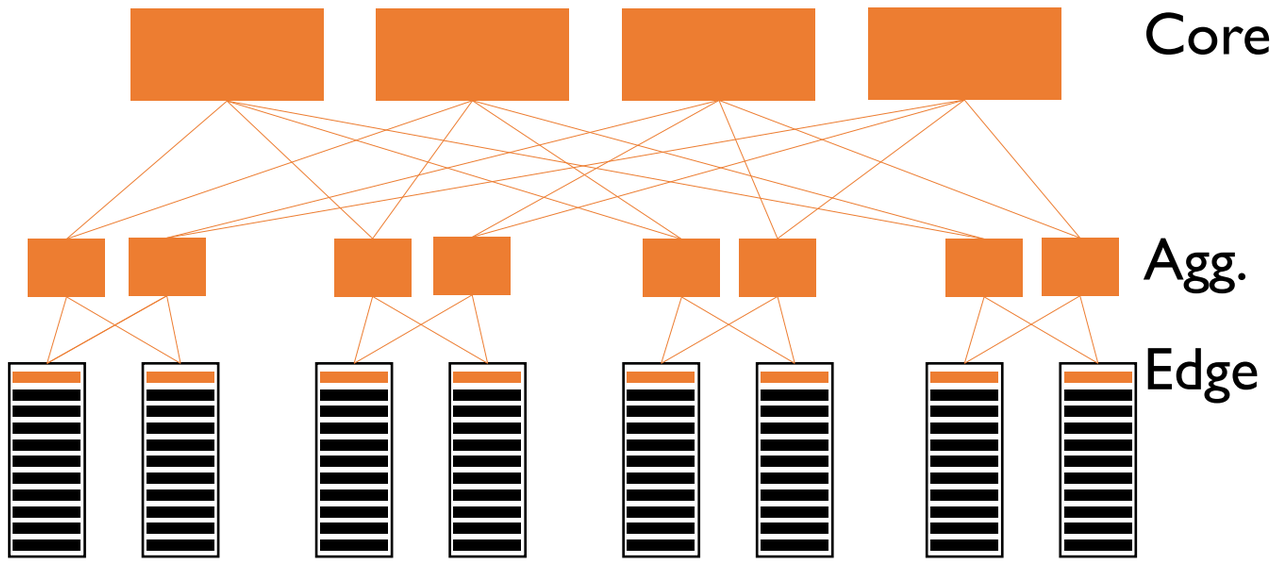
- Similar trends in telephone switches led to designing a topology with high bandwidth by interconnecting smaller commodity switches.





Clos topology (Fat-tree)

- Clos topology
  - Cheaper
  - Easier to scale
  - NO/low oversubscription
  - Higher path diversity
  - …



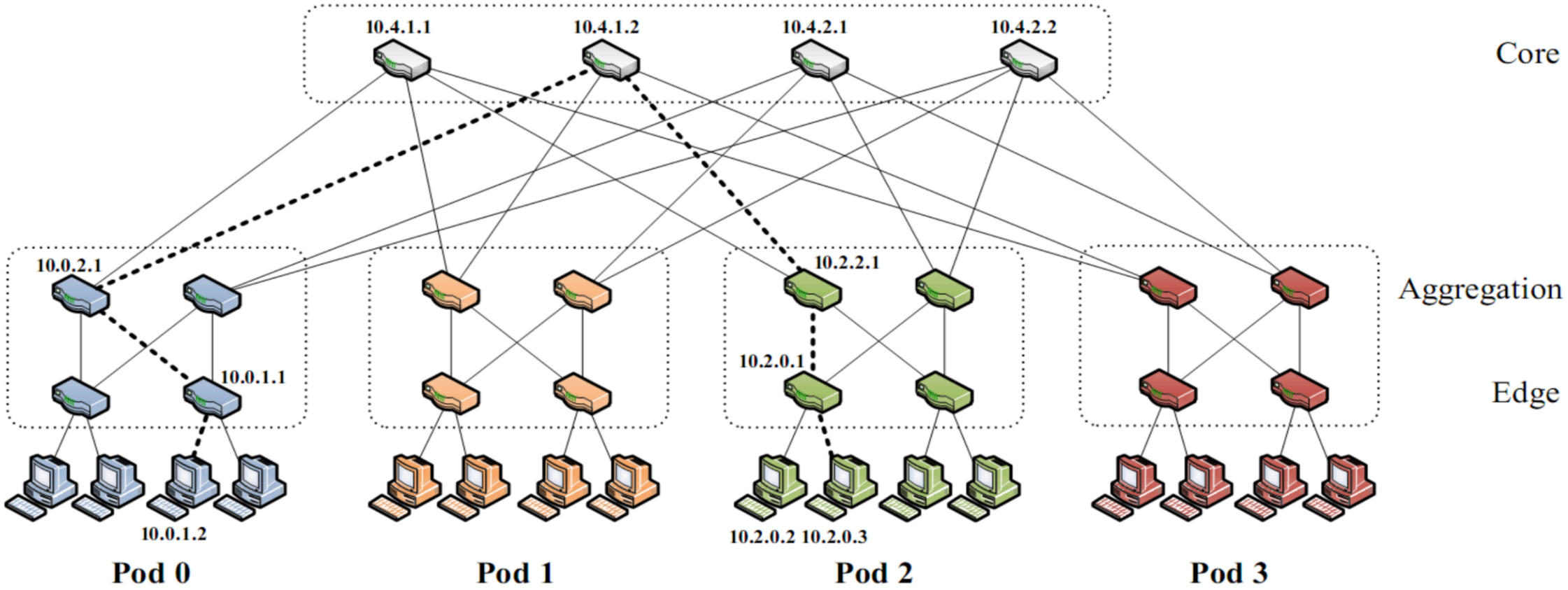
k=4 Fat-Tree

$(k/2)^2$

$k/2$

$k/2$

$(k/2)^2$



Fat-tree-based DC architecture

- Inter-connect racks (of servers) using a fat-tree topology
- K-ary fat-tree: three-layer topology (edge, aggregation and core)
  - each pod consists of $(k/2)^2$ servers & 2 layers of k/2 k-port switches
  - each edge switch connects to k/2 servers & k/2 aggr. switches
  - each aggr. switch connects to k/2 edge & k/2 core switches
  - $(k/2)^2$ core switches: each connects to k pods

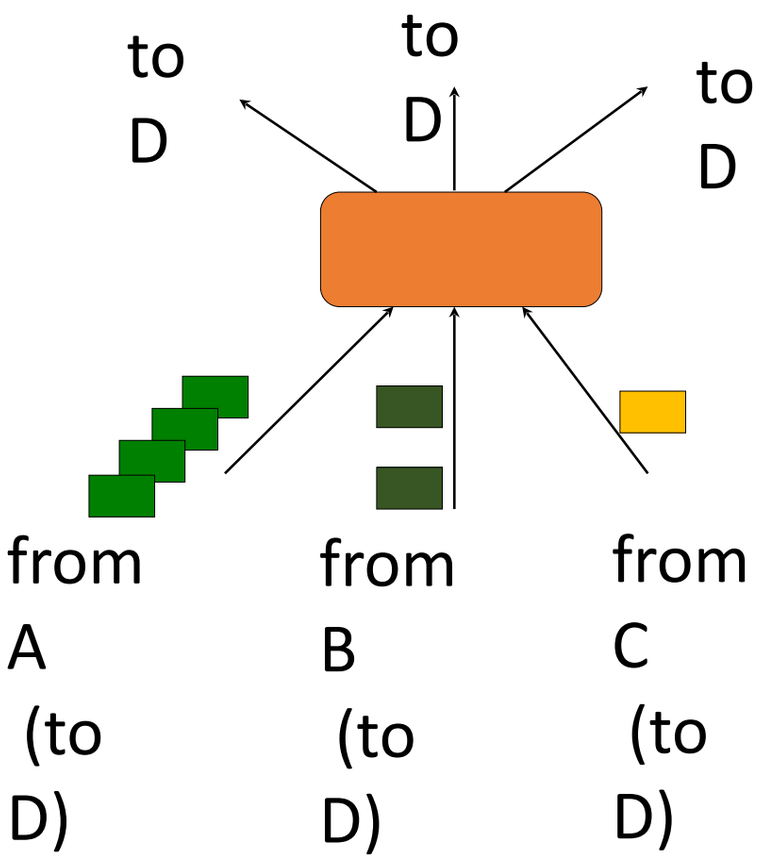
Fat-tree-based DC architecture

- Why Fat-tree?
  - Fat-tree has identical bandwidth at any bisections
  - Each layer has the same aggregated bandwidth
- Can be built using cheap devices with uniform capacity
  - Each port supports same speed as end host
  - All devices can transmit at line speed if packets are distributed uniform along available paths
- Great scalability: k-port switch supports $k^3/4$ servers

Problems of Fat-Tree

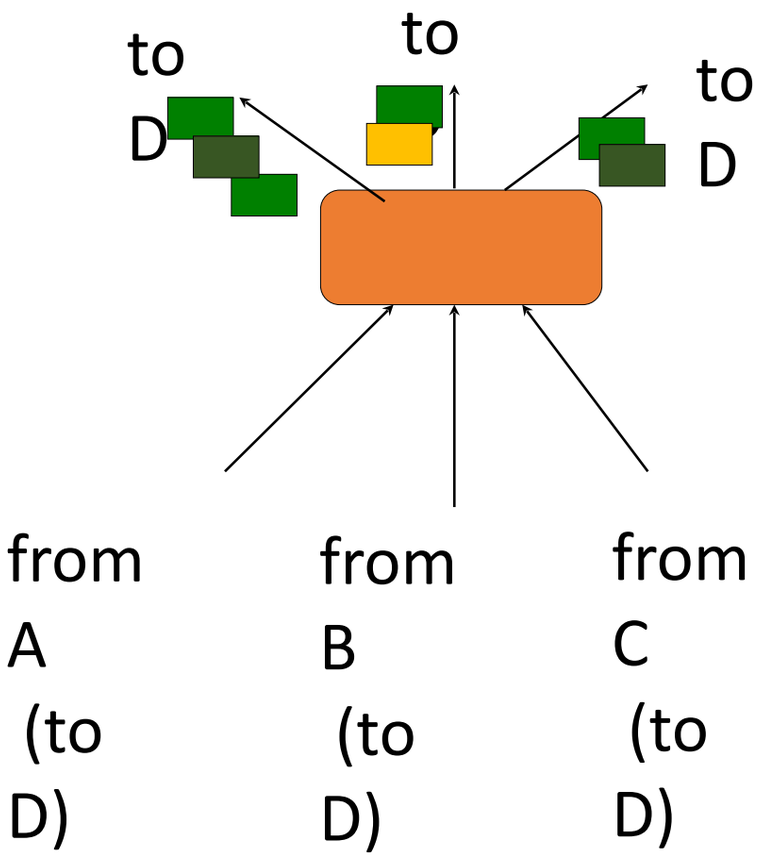
- What routing protocols should we run on these switches?
- Layer 2 switch algorithm: data plane flooding!
- Layer 3 IP routing:
  - shortest path IP routing will typically use only one path despite the path diversity in the topology
  - if using equal-cost multi-path routing at each switch independently and blindly, packet re-ordering may occur; further load may not necessarily be well-balanced
  - Aside: control plane flooding!

Forwarding



Per-packet load balancing

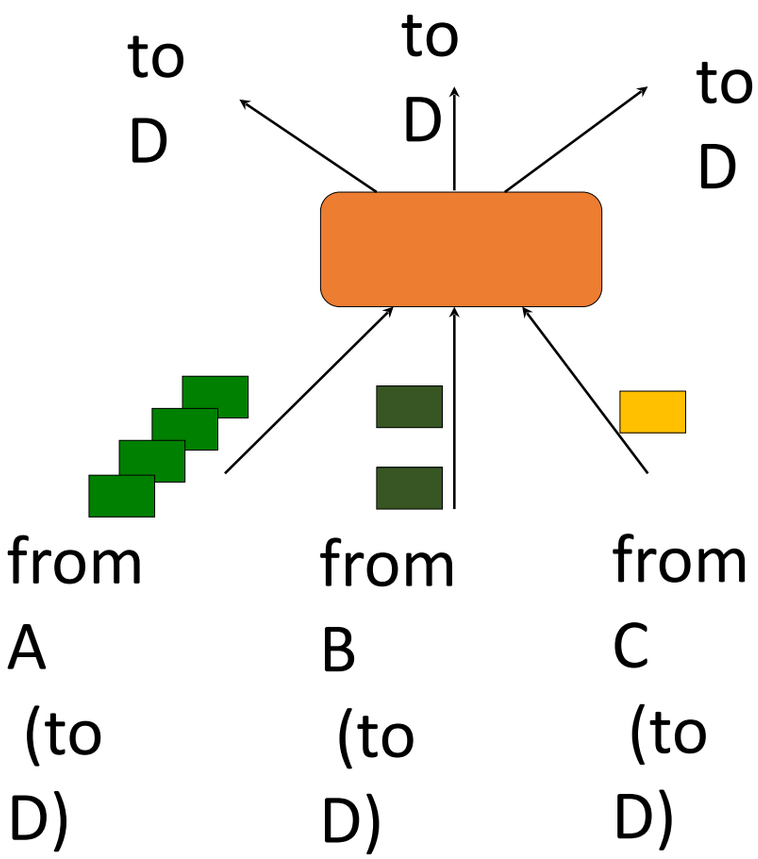
Forwarding



Traffic well spread (even w/ elephant flows)

BUT Interacts poorly w/ TCP

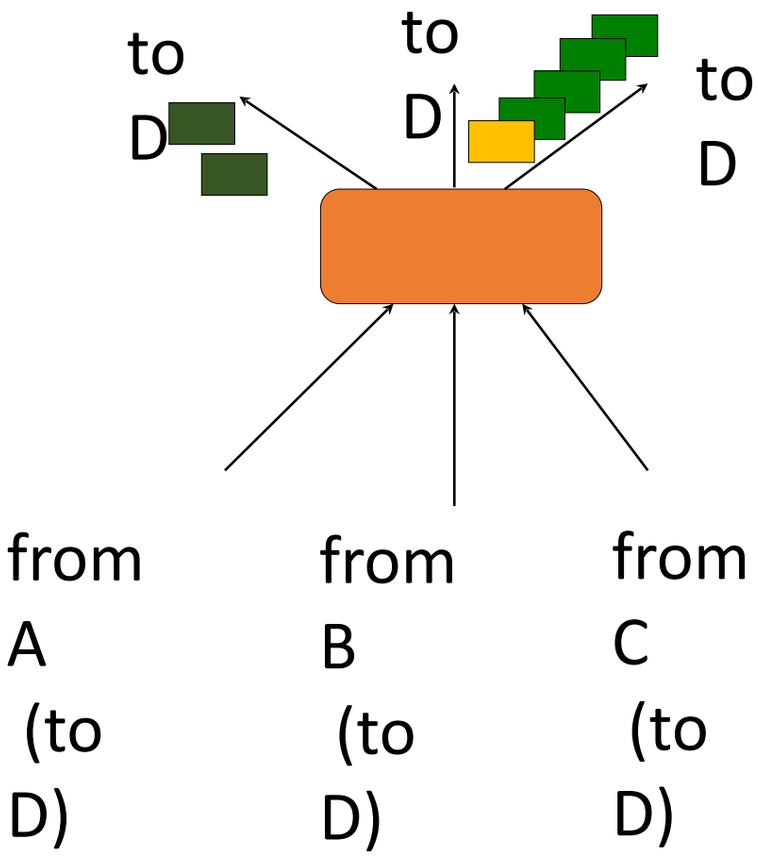
Forwarding



Per-flow load balancing (ECMP, “Equal Cost Multi Path”)

E.g., based on (src and dst IP and port)

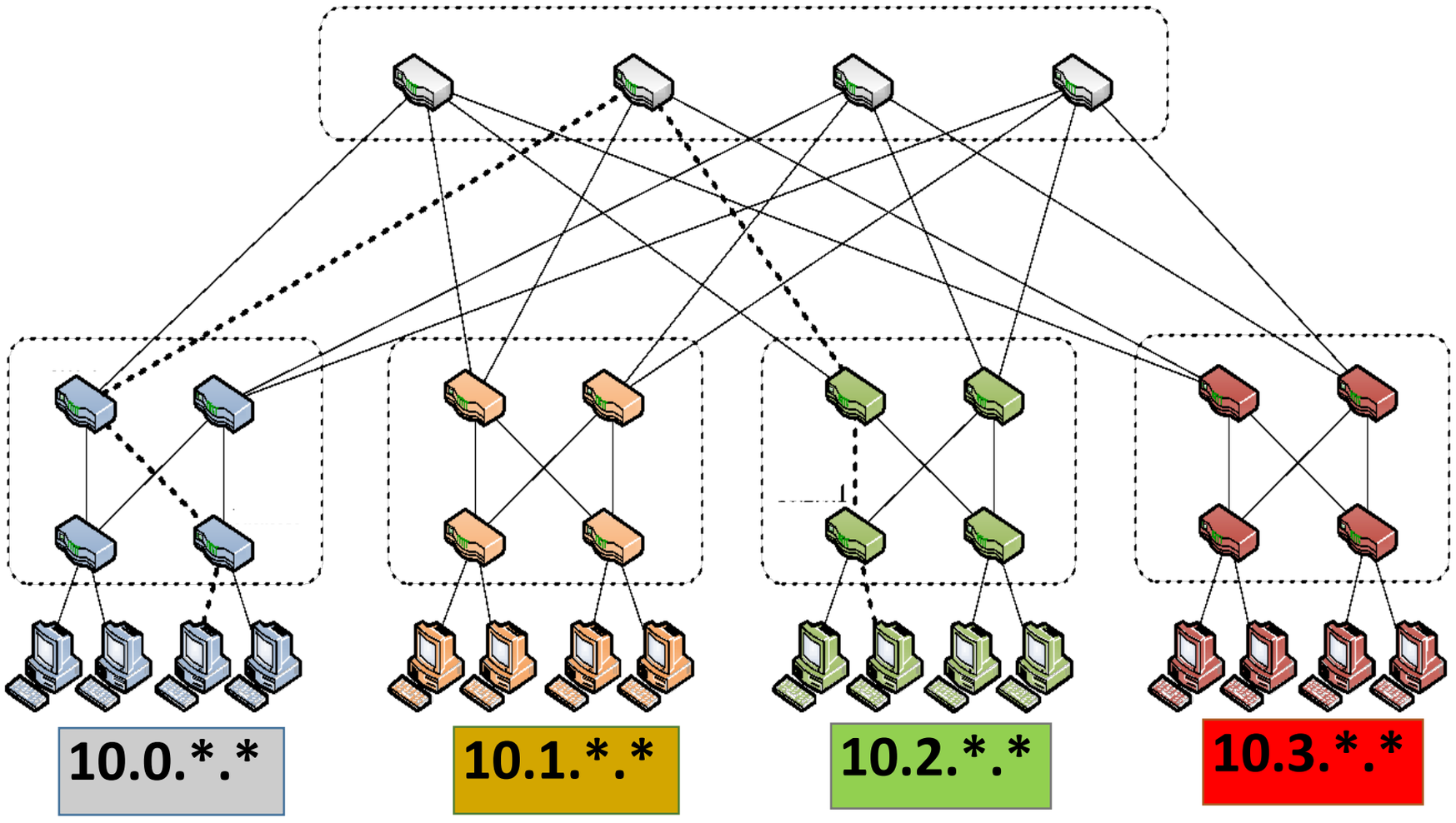
Forwarding



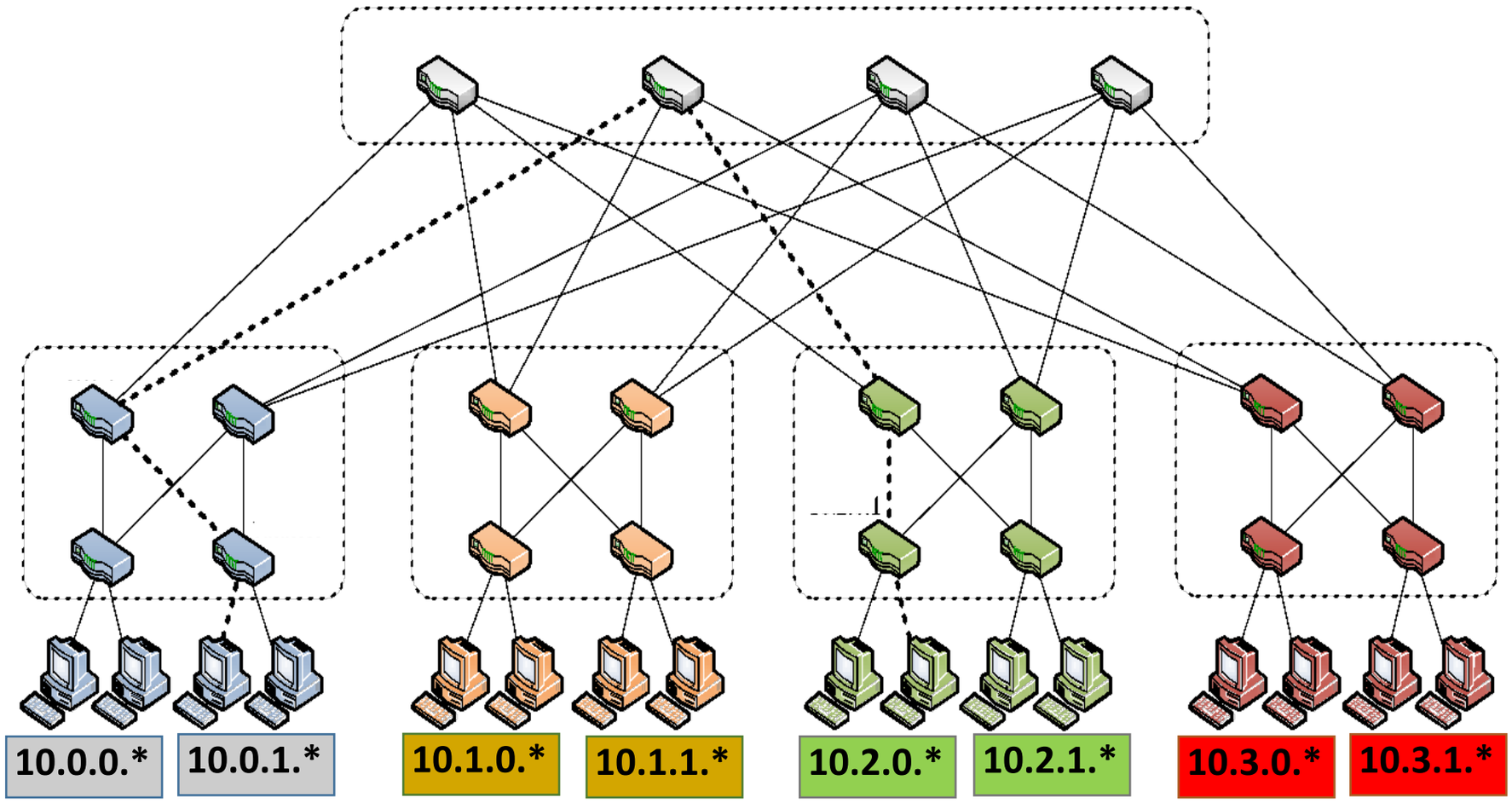
A flow follows a single path (→ TCP is happy)

Suboptimal load-balancing; elephants are a problem

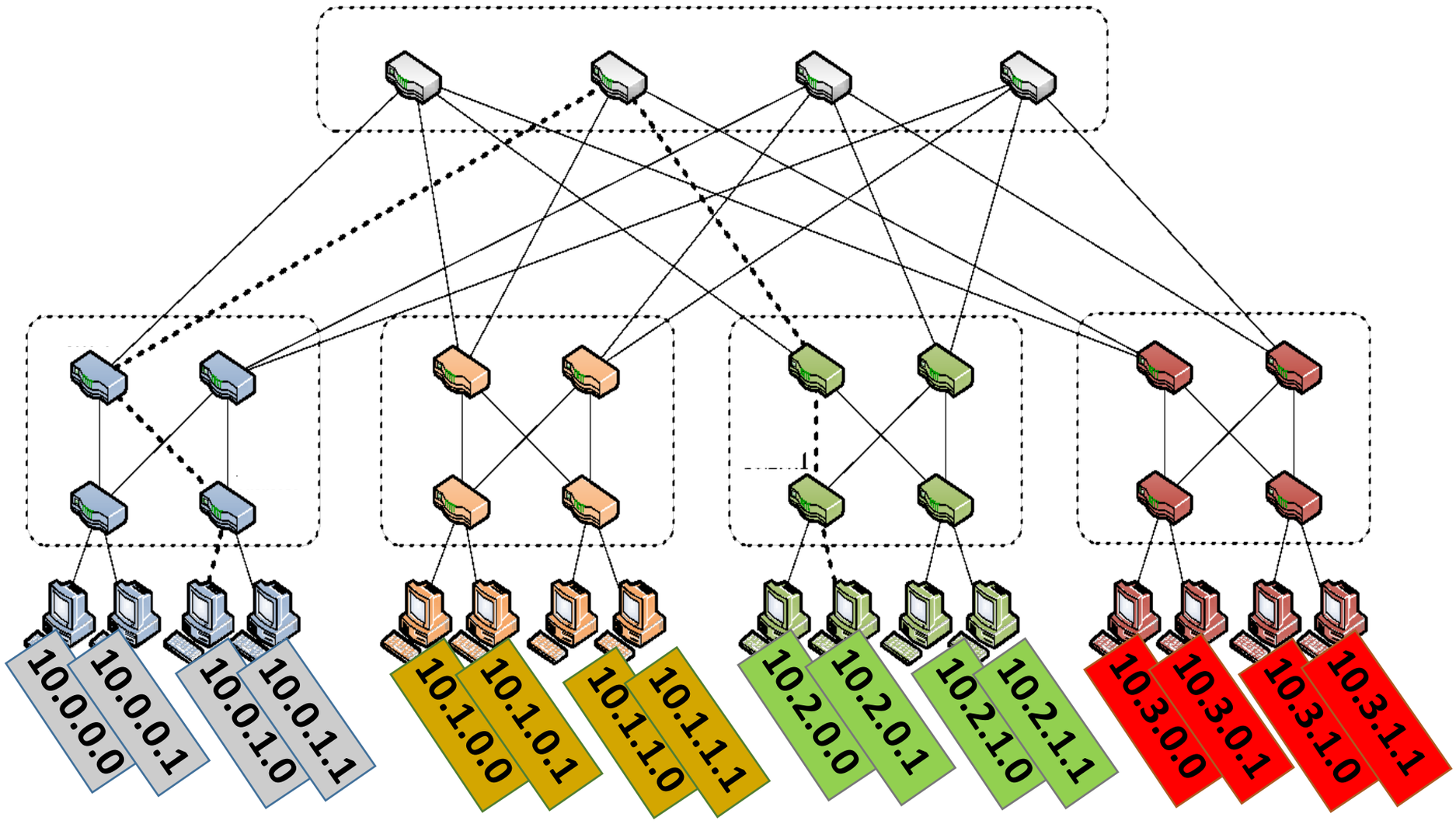
Solution 1: Topology-aware addressing



Solution 1: Topology-aware addressing



Solution 1: Topology-aware addressing



Solution 1: Topology-aware addressing

- Addresses embed location in regular topology
- Maximum #entries/switch: k (= 4 in example)
  - Constant, independent of #destinations!
- No route computation / messages / protocols
  - Topology is hard-coded, but still need localized link failure detection
- Problems?
  - VM migration: ideally, VM keeps its IP address when it moves
  - Vulnerable to (topology/addresses) misconfiguration

Diffusion Optimizations

- 1. Flow classification: Denote a flow as a sequence of packets with the same entries; pod switches forward subsequent packets of the same flow to same outgoing port, and periodically reassign a minimal number of output ports

- Eliminates local congestion

- Assign traffic to ports on a per-flow basis instead of a per-host basis, Ensure fair distribution on flows

Diffusion Optimizations

- 2. Flow scheduling: Pay attention to routing large flows, edge switches detect any outgoing flow whose size grows above a predefined threshold, and then send notification to a central scheduler. The central scheduler tries to assign non-conflicting paths for these large flows.

- Eliminates global congestion

- Prevent long lived flows from sharing the same links – Assign long lived flows to different links

Solution 2: Centralize + Source Routes

- Centralized “controller” server knows topology and computes routes
- Controller hands server all paths to each destination
  - O(#destinations) state per server, but server memory cheap (e.g., 1M routes x 100B/route=100MB)
- Server inserts entire path vector into packet header (“source routing”)
  - E.g., header=[dst=D | index=0 | path={S5,S1,S2,S9}]
- Switch forwards based on packet header
  - index++; next-hop = path[index]

Solution 2: Centralize + Source Routes

- #entries per switch?
  - None!
- #routing messages?
  - Akin to a broadcast from controller to all servers
- Pro:
  - Switches very simple and scalable
  - Flexibility: end-points control route selection
- Cons:
  - Scalability / robustness of controller (SDN issue)
  - Clean-slate design of everything

Software Defined Networking (SDN)

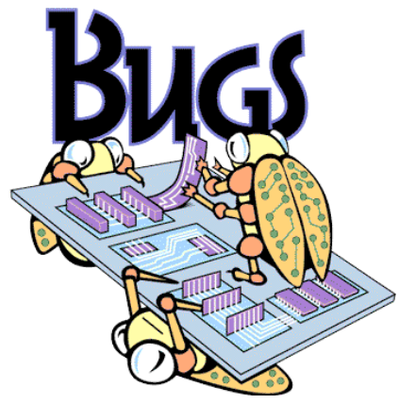
The Internet: A Remarkable Story

- Tremendous success
  - From research experiment to global infrastructure
- Brilliance of under-specifying
  - Network: best-effort packet delivery
  - Hosts: arbitrary applications
- Enables innovation in applications
  - Web, P2P, VoIP, social networks, virtual worlds
- But, change is easy only at the edge… ☹

Networks are Hard to Manage

- Operating a network is expensive
  - More than half the cost of a network
  - Yet, operator error causes most outages

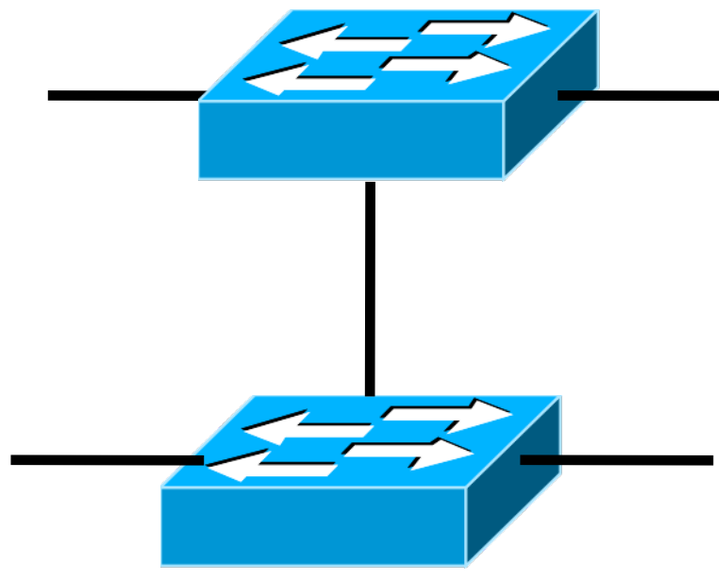
- Buggy software in the equipment
  - Routers with 20+ million lines of code
  - Cascading failures, vulnerabilities, etc.



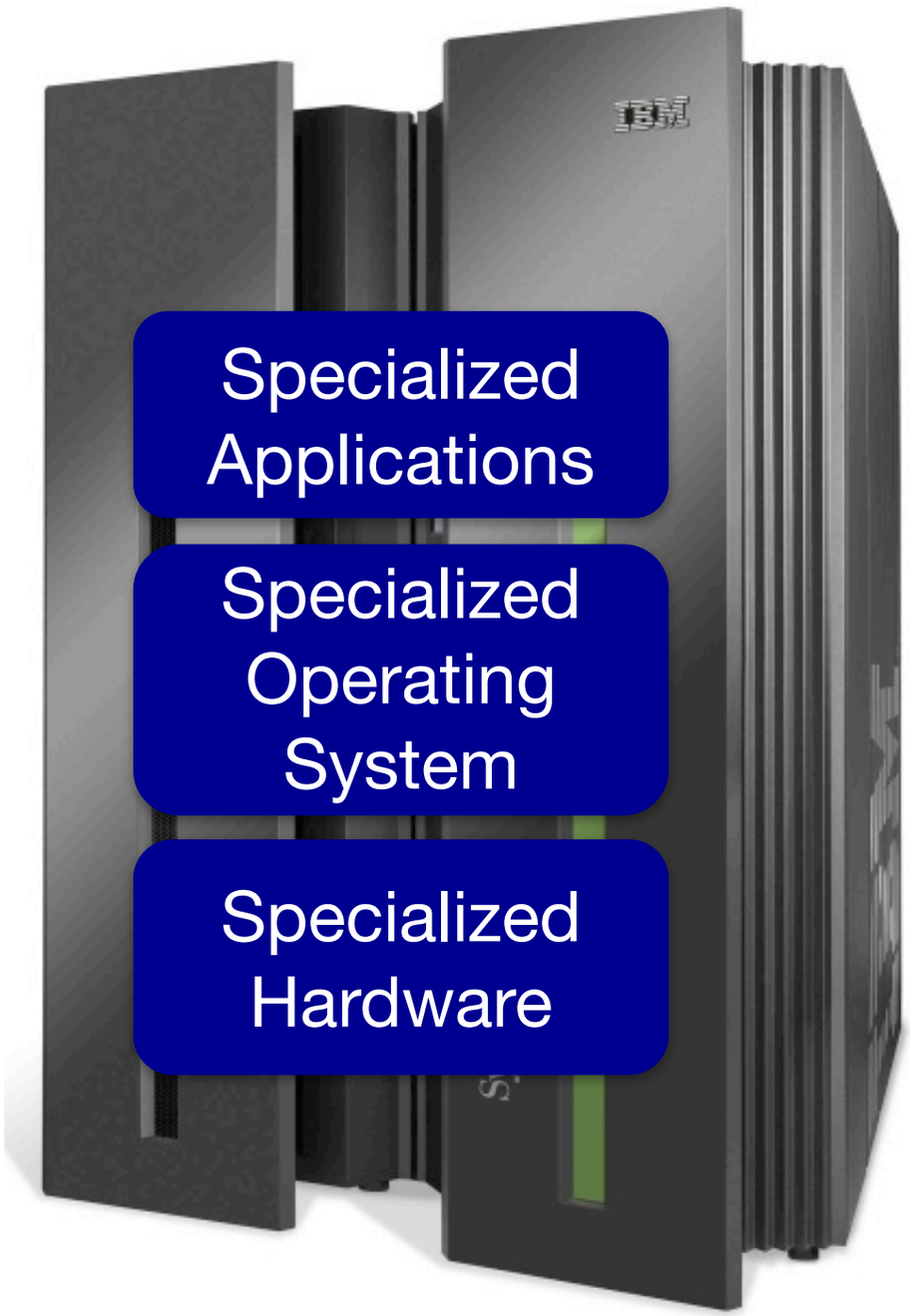
- The network is “in the way”
  - Especially a problem in data centers
  - … and home networks

Inside the ‘Net: A Different Story…

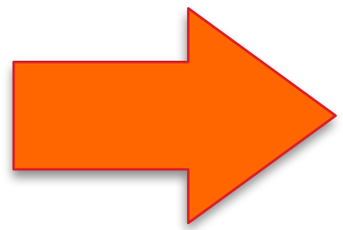
- Closed equipment
  - Software bundled with hardware
  - Vendor-specific interfaces
- Over specified
  - Slow protocol standardization
- Few people can innovate
  - Equipment vendors write the code
  - Long delays to introduce new features

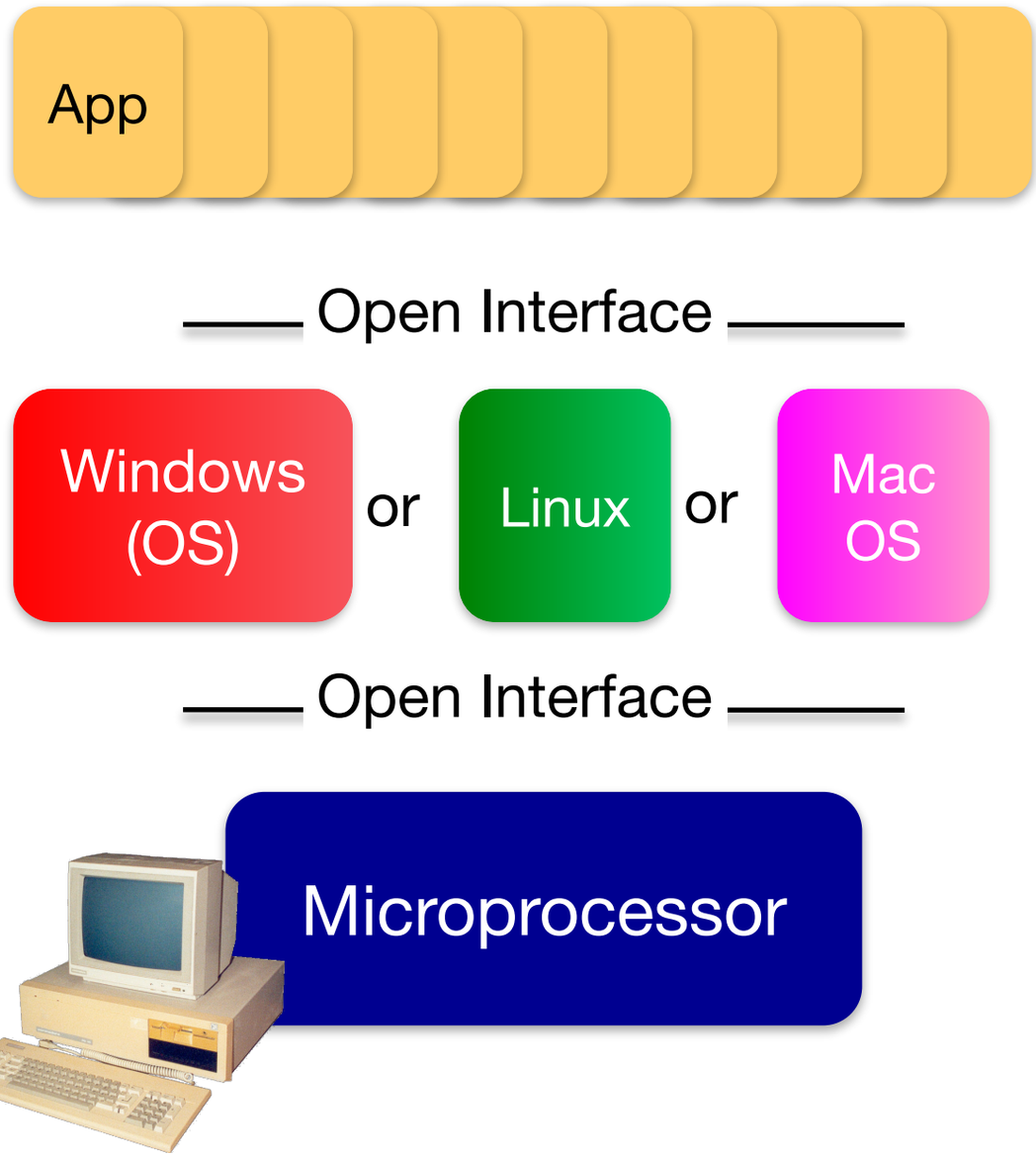


Impacts performance, security, reliability, cost…



Vertically integrated
Closed, proprietary
Slow innovation
Small industry





Horizontal
Open interfaces
Rapid innovation
Huge industry

Routing, management, mobility management, access control, VPNs, …

Feature Feature

OS

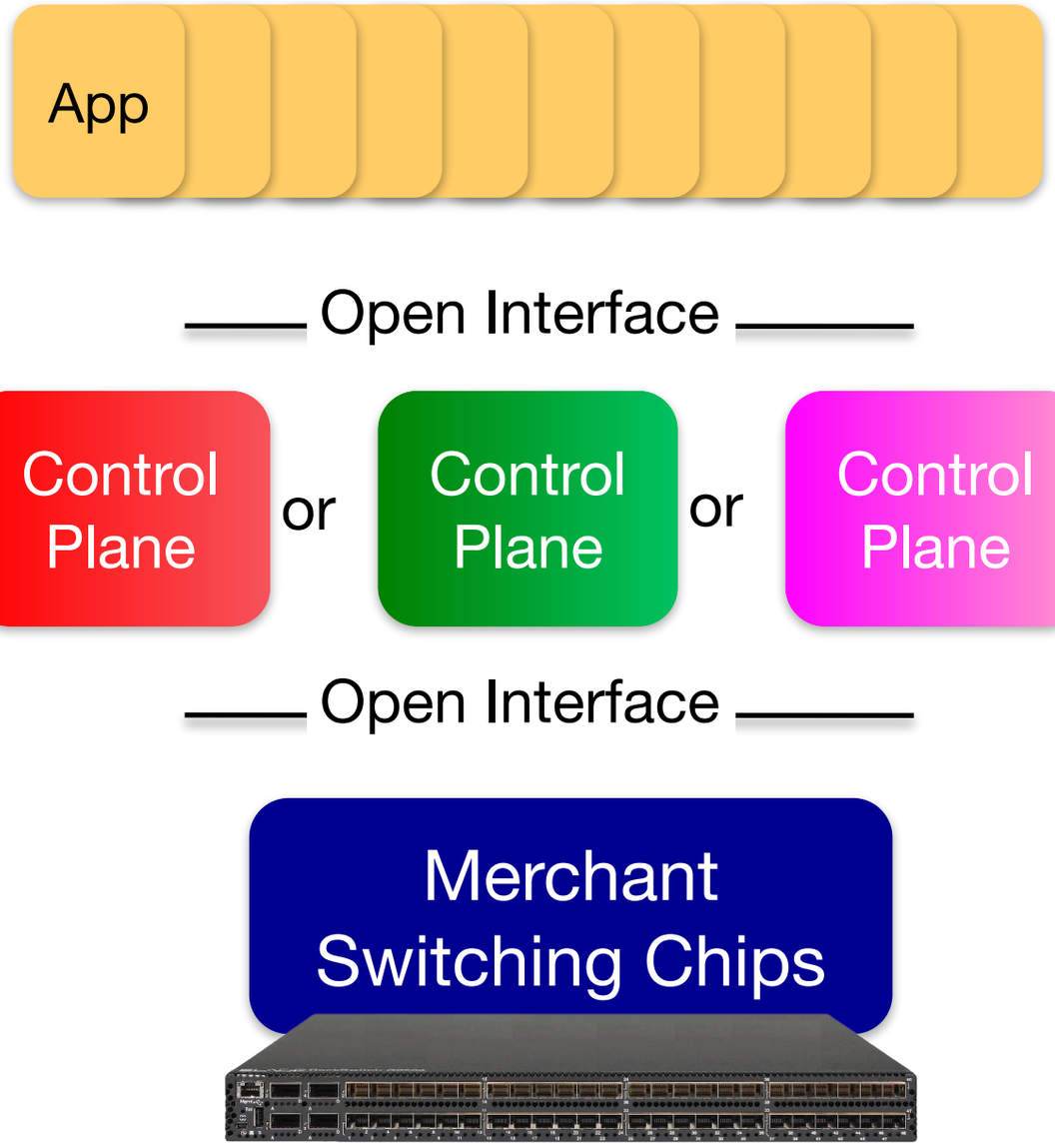
Custom Hardware

Million of lines of source code — 6,000 RFCs

Billions of gates — Bloated — Power Hungry

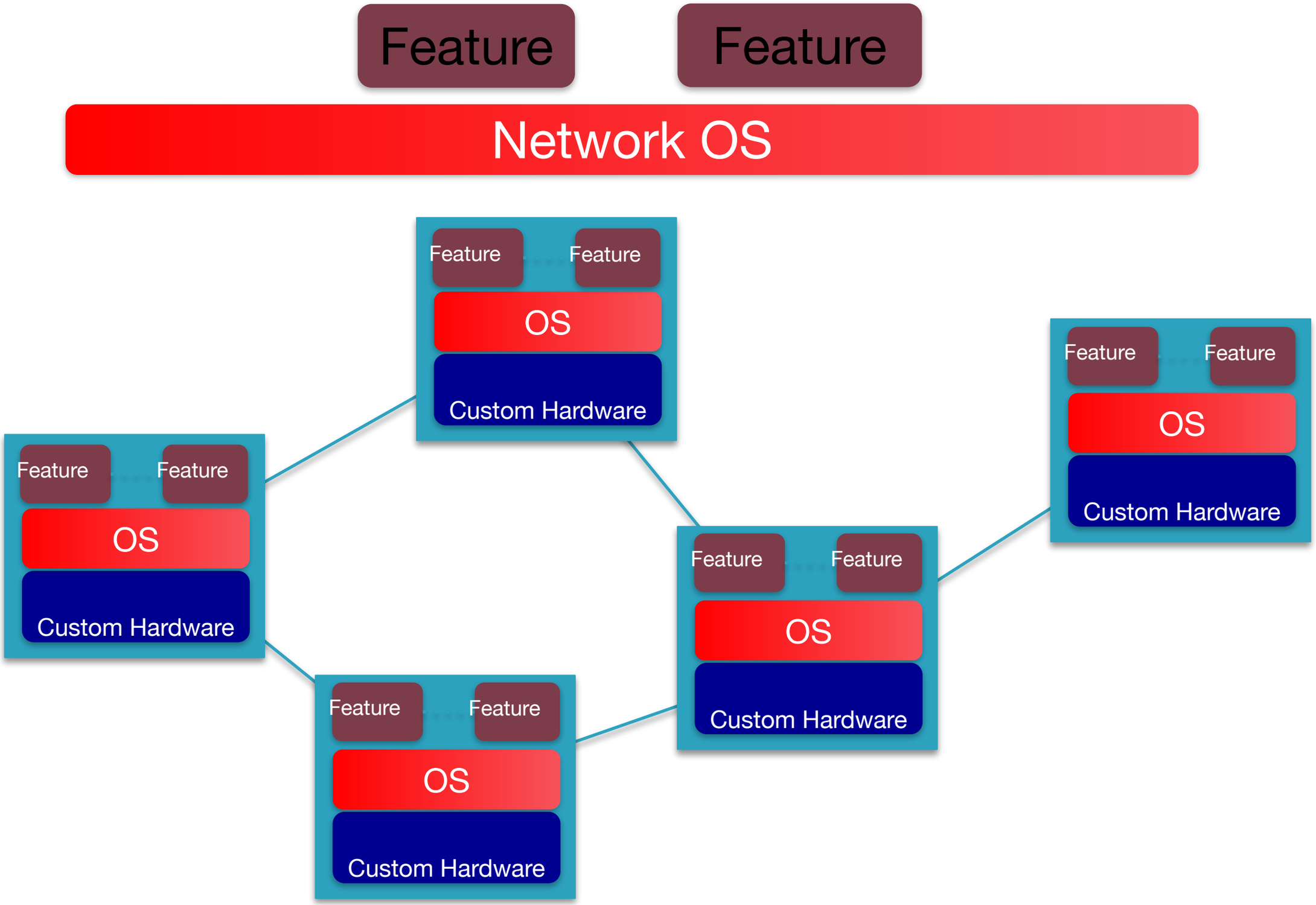
- Vertically integrated, complex, closed, proprietary
- Networking industry with “mainframe” mind-set

Vertically integrated
Closed, proprietary
Slow innovation

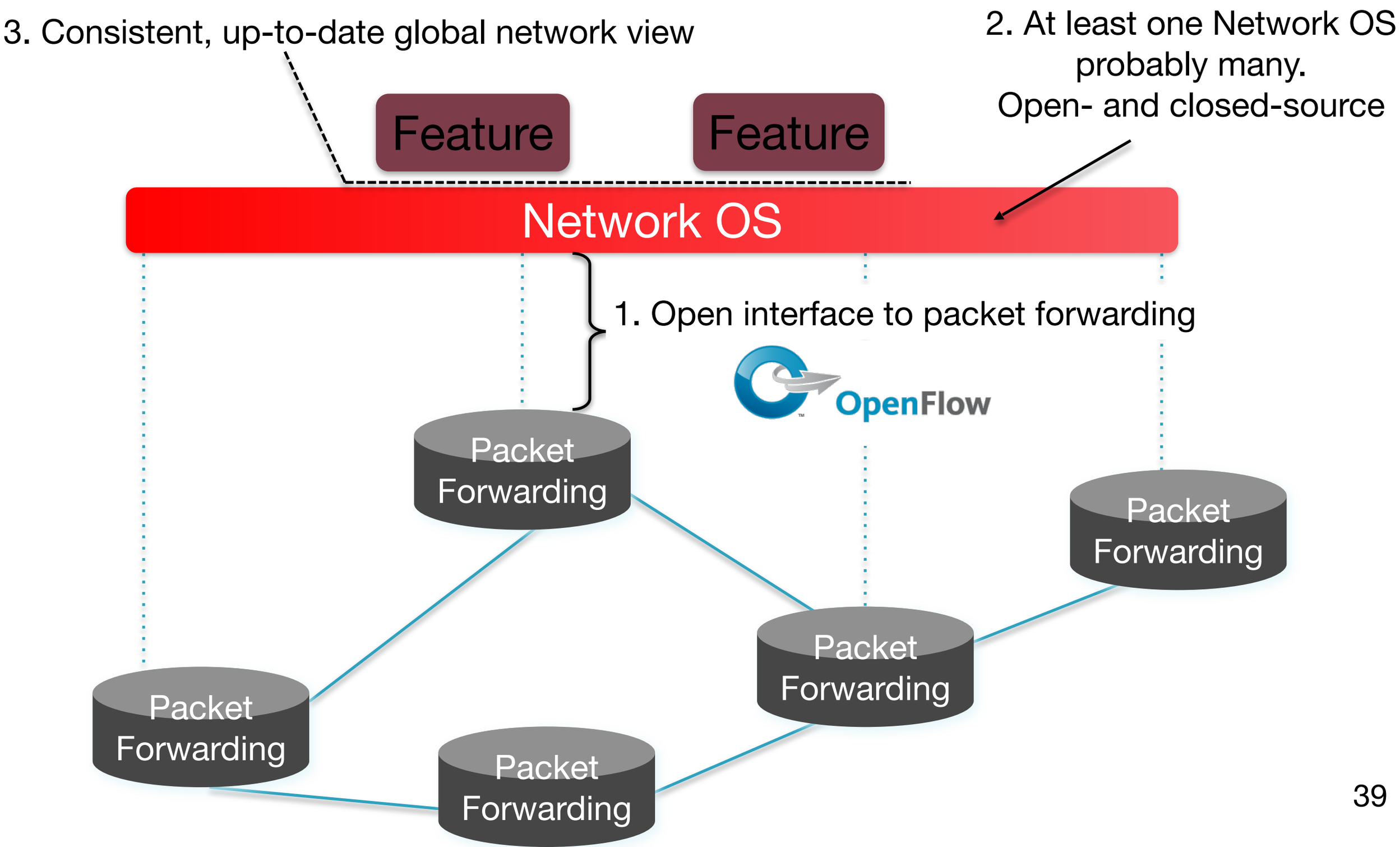


Horizontal
Open interfaces
Rapid innovation

The network is changing



Software Defined Network (SDN)



Network OS

Network OS: distributed system that creates a consistent, up-to-date network view

- Runs on servers (controllers) in the network
- NOX, ONIX, Trema, Beacon, Maestro, … + more

Uses forwarding abstraction to:

- Get state information **from** forwarding elements
- Give control directives **to** forwarding elements