Datacenter Networks

Justine Sherry & Peter Steenkiste

15-441/641

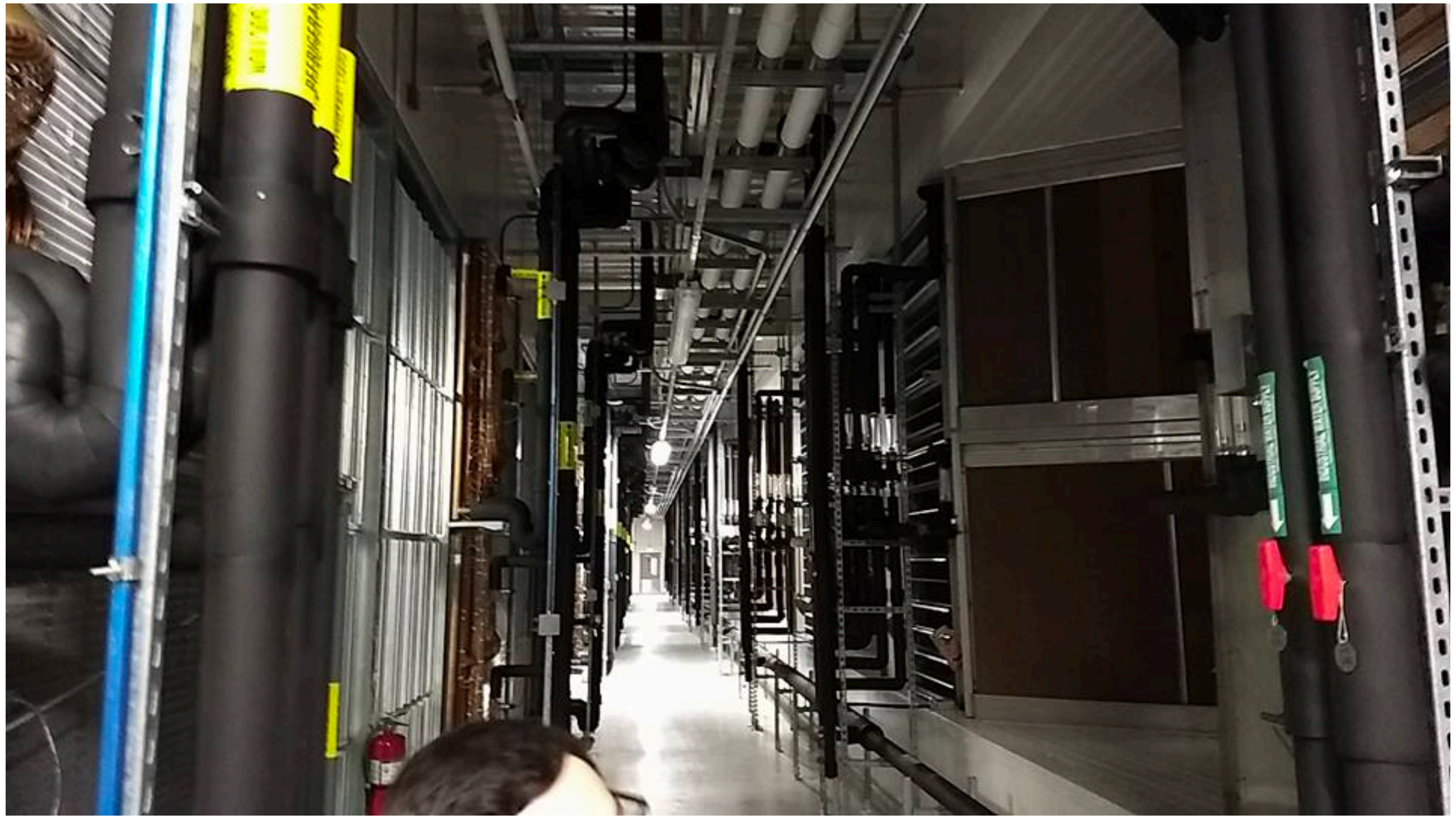
Carnegie Mellon University

My trip to a Facebook datacenter last year.

(These are actually stock photos because you can't take pics in the machine rooms.)

Receiving room: this many servers arrived *today*

Upstairs: Temperature and Humidity Control

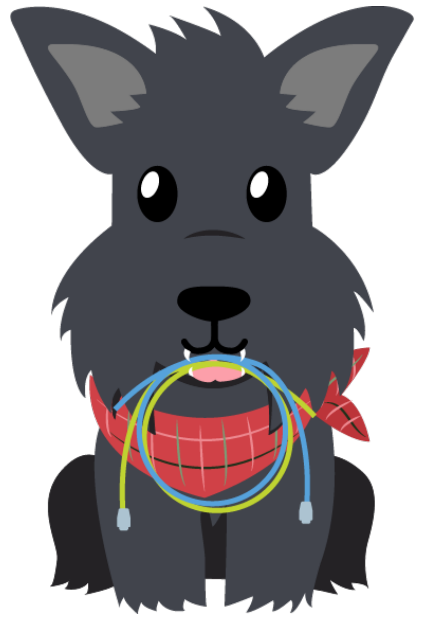


Upstairs: Temperature and Humidity Control

so many fans

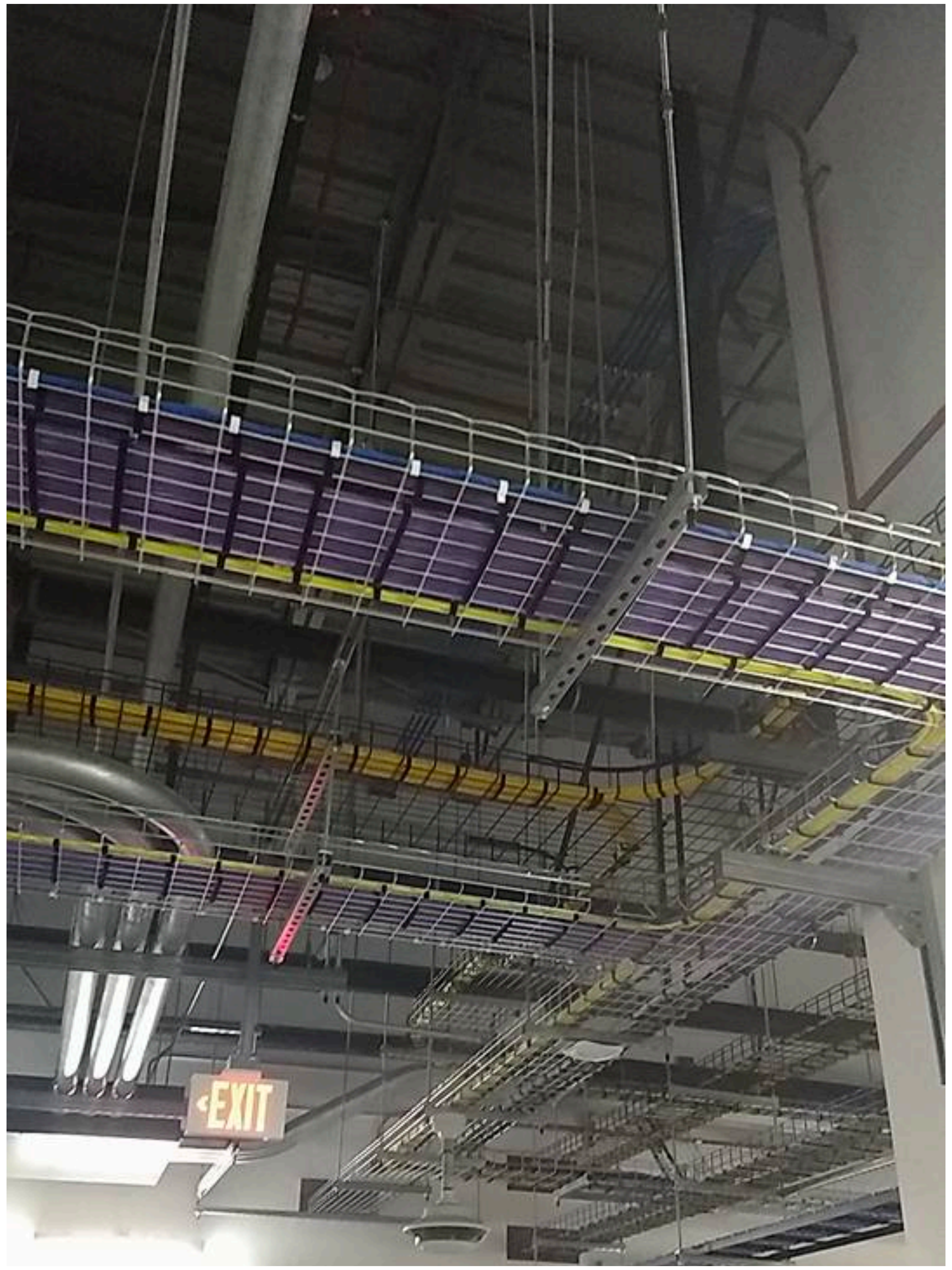
Why so many servers?

- **Internet Services**
  - Billions of people online using online services requires lots of compute… somewhere!
  - Alexa, Siri, and Cortana are always on call to answer my questions!
- **Warehouse-Scale Computing**
  - Large scale data analysis: billions of photos, news articles, user clicks — all of which needs to be analyzed.
  - Large compute frameworks like MapReduce and Spark coordinate tens to thousands of computers to work together on a shared task.



A very large network switch

Cables in ceiling trays run everywhere



How are datacenter networks different from networks we’ve seen before?

- **Scale**: very few local networks have so many machines in one place: 10’s of thousands of servers — and they all work together like one computer!
- **Control**: entirely administered by one organization — unlike the Internet, datacenter owners control every switch in the network **and** the software on every host
- **Performance:** datacenter latencies are 10s of us, with 10, 40, even 100Gbit links.

How do these factors change how we *design* datacenter networks?

How are datacenter networks different from networks we've seen before?

There are *many* ways that datacenter networks differ from the Internet. Today I want to consider these three themes:

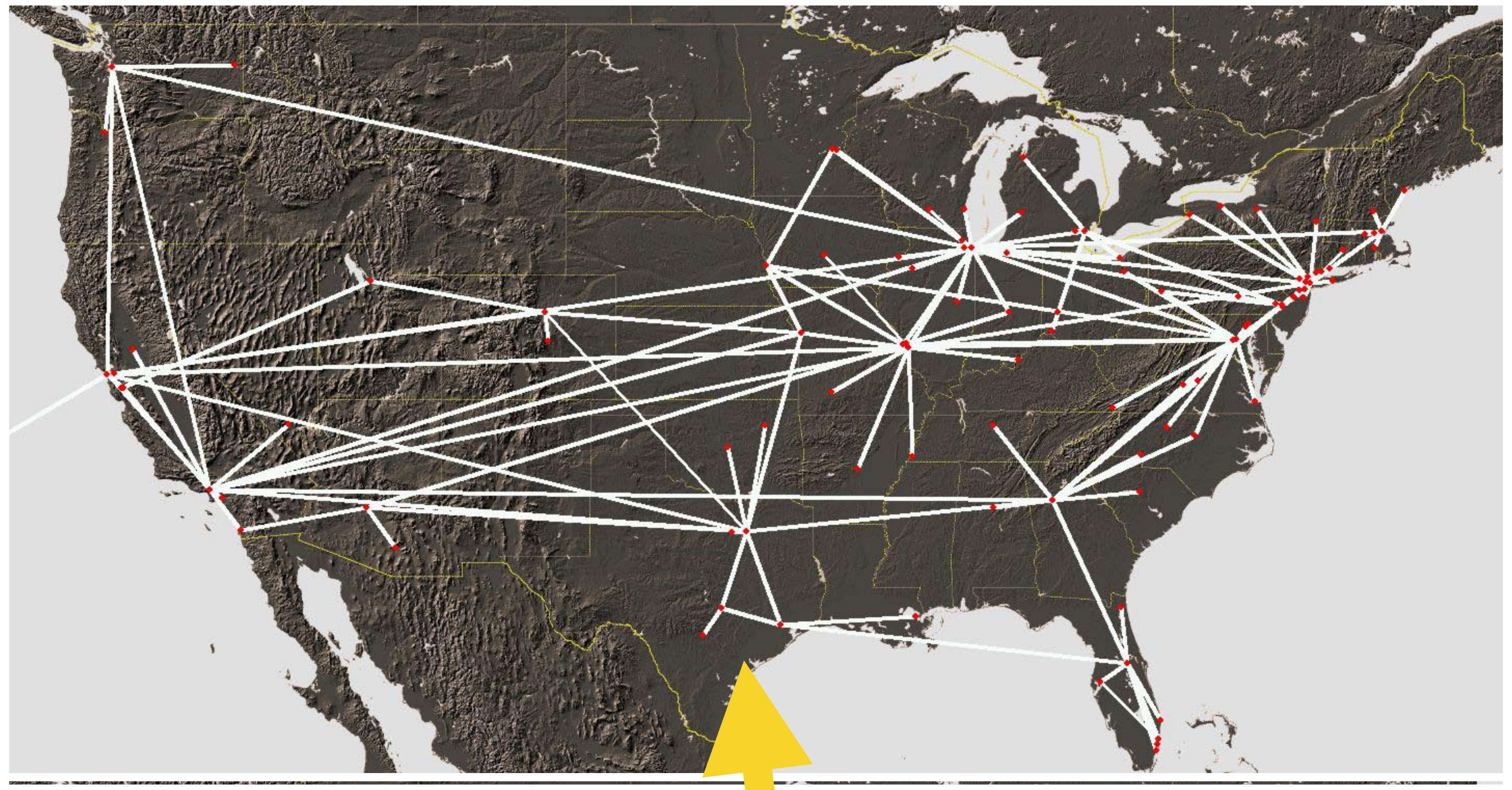
1. Topology

2. Congestion Control

3. Virtualization

Network topology is the arrangement of the elements of a communication network.

Wide Area Topologies



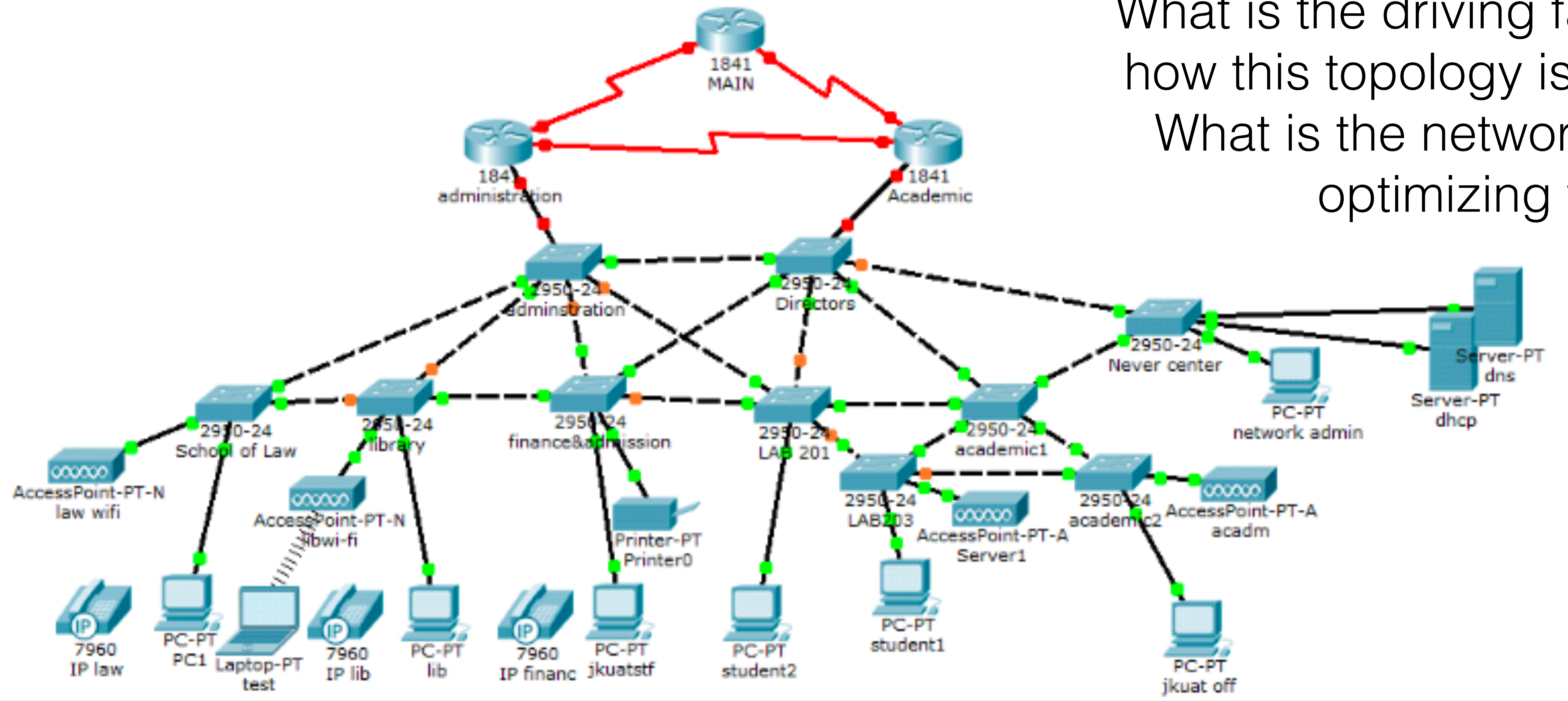
AT&T's Wide Area Backbone, 2002

This is called a "hub and spoke"

Google's Wide Area Backbone, 2011

Every city is connected to at least two others. Why?

A University Campus Topology



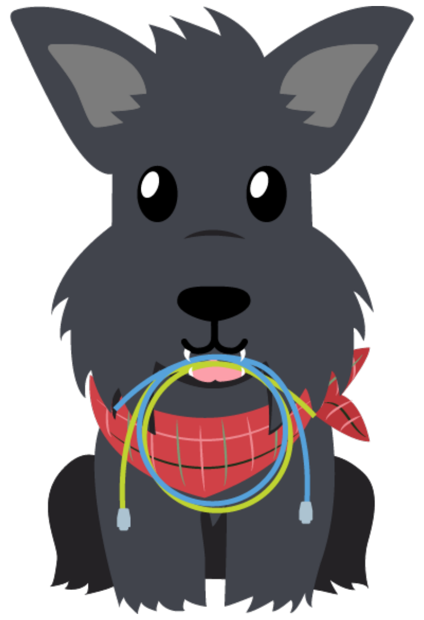
What is the driving factor behind how this topology is structured? What is the network engineer optimizing for?

You're a network engineer…

- …in a warehouse-sized building… with 10,000 computers…
- **What features do you want from your network topology?**

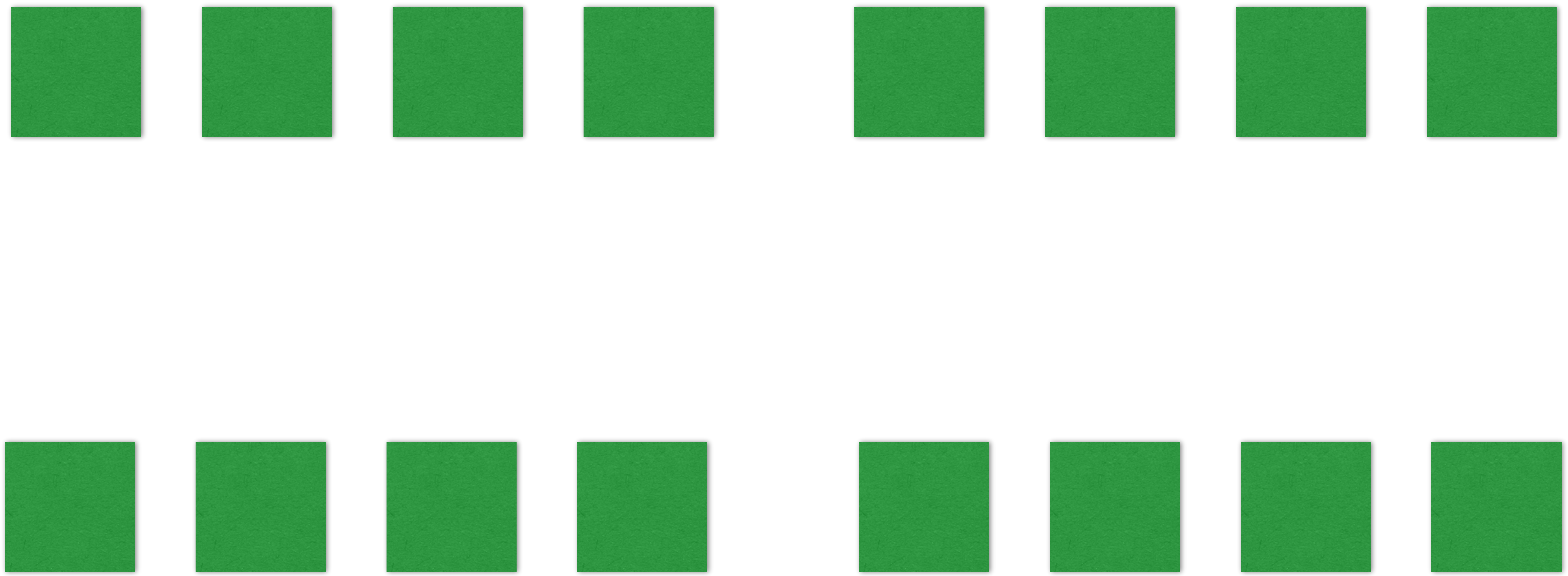
Desirable Properties

- **Low Latency:** Very few “hops” between destinations
- **Resilience:** Able to recover from link failures
- **Good Throughput:** Lots of endpoints can communicate, all at the same time.
- **Cost-Effective:** Does not rely too much on expensive equipment like very high bandwidth, high port-count switches.
- **Easy to Manage:** Won’t confuse network administrators who have to wire so many cables together!



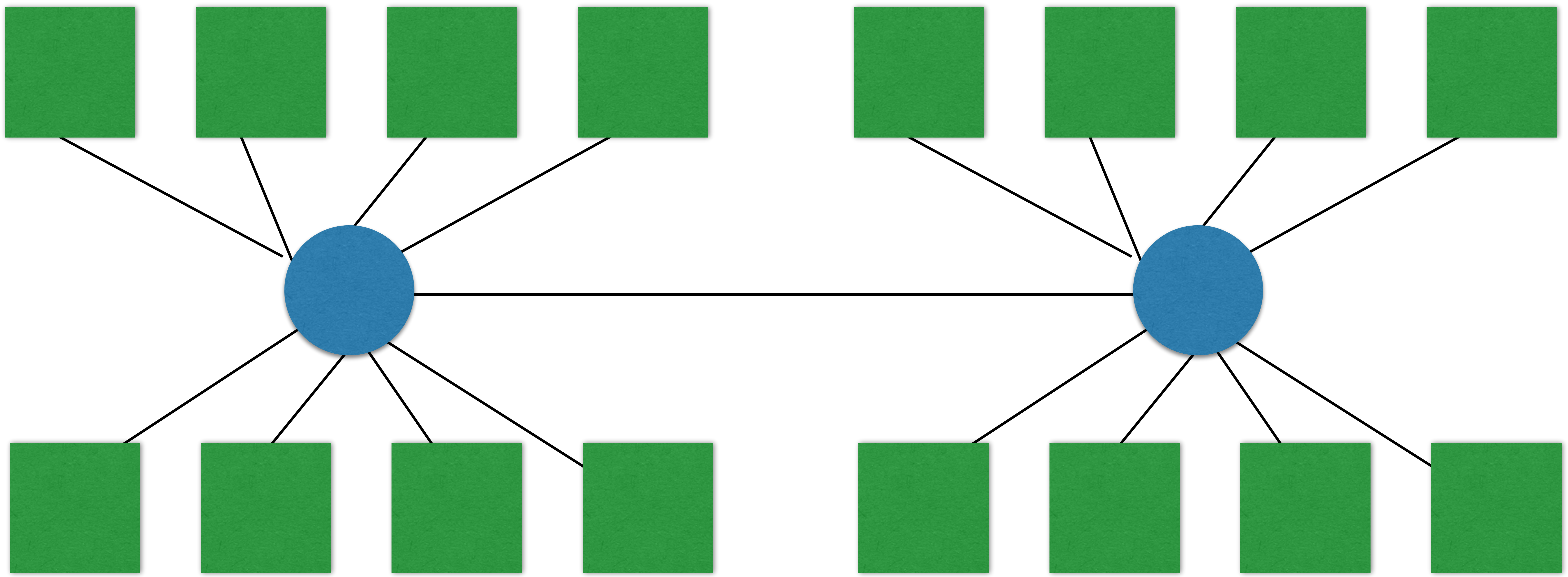
Activity

- We have 16 servers. You can buy as many switches and build as many links as you want. How do you design your network topology?



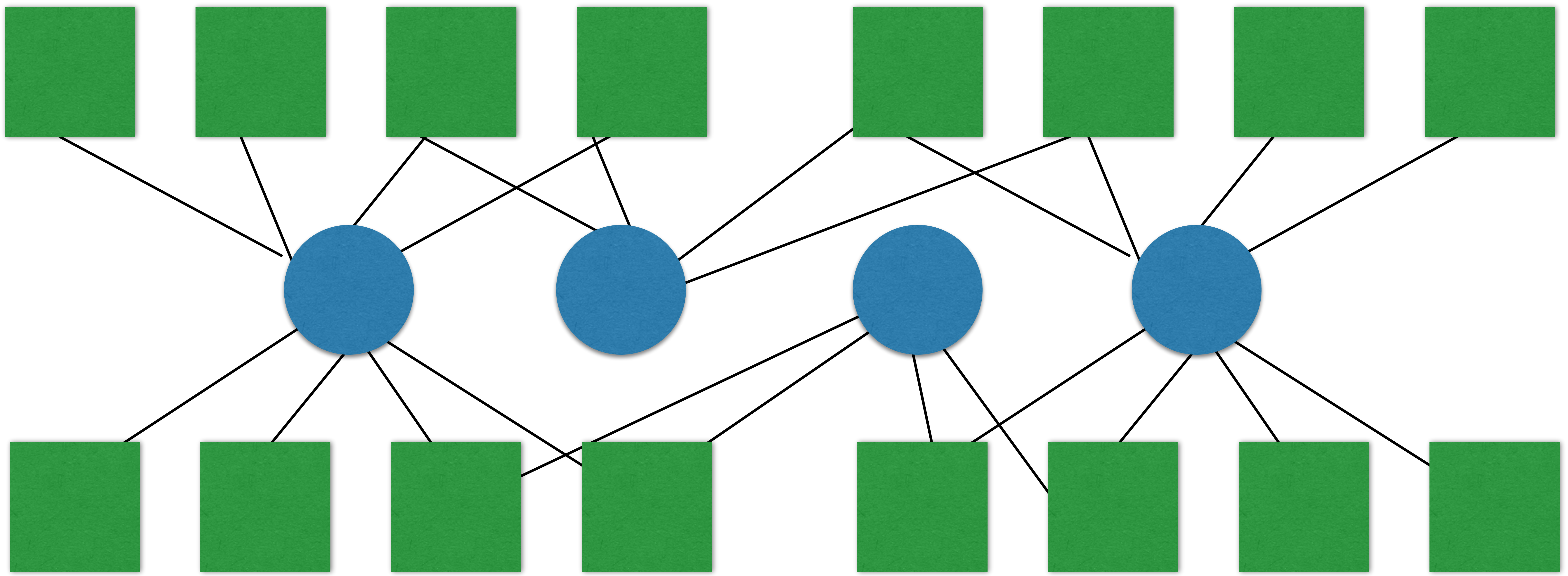
Activity

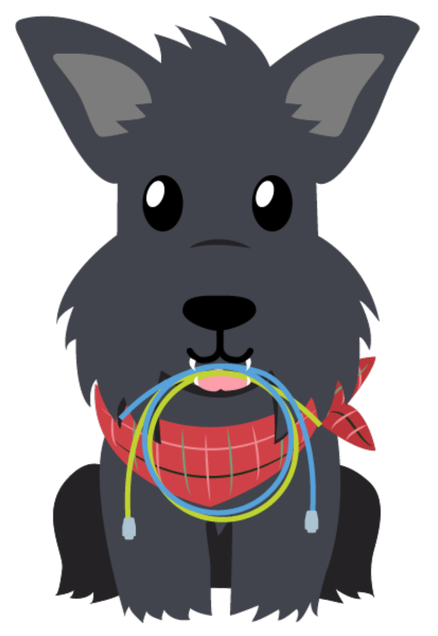
- We have 16 servers. You can buy as many switches and build as many links as you want. How do you design your network topology?



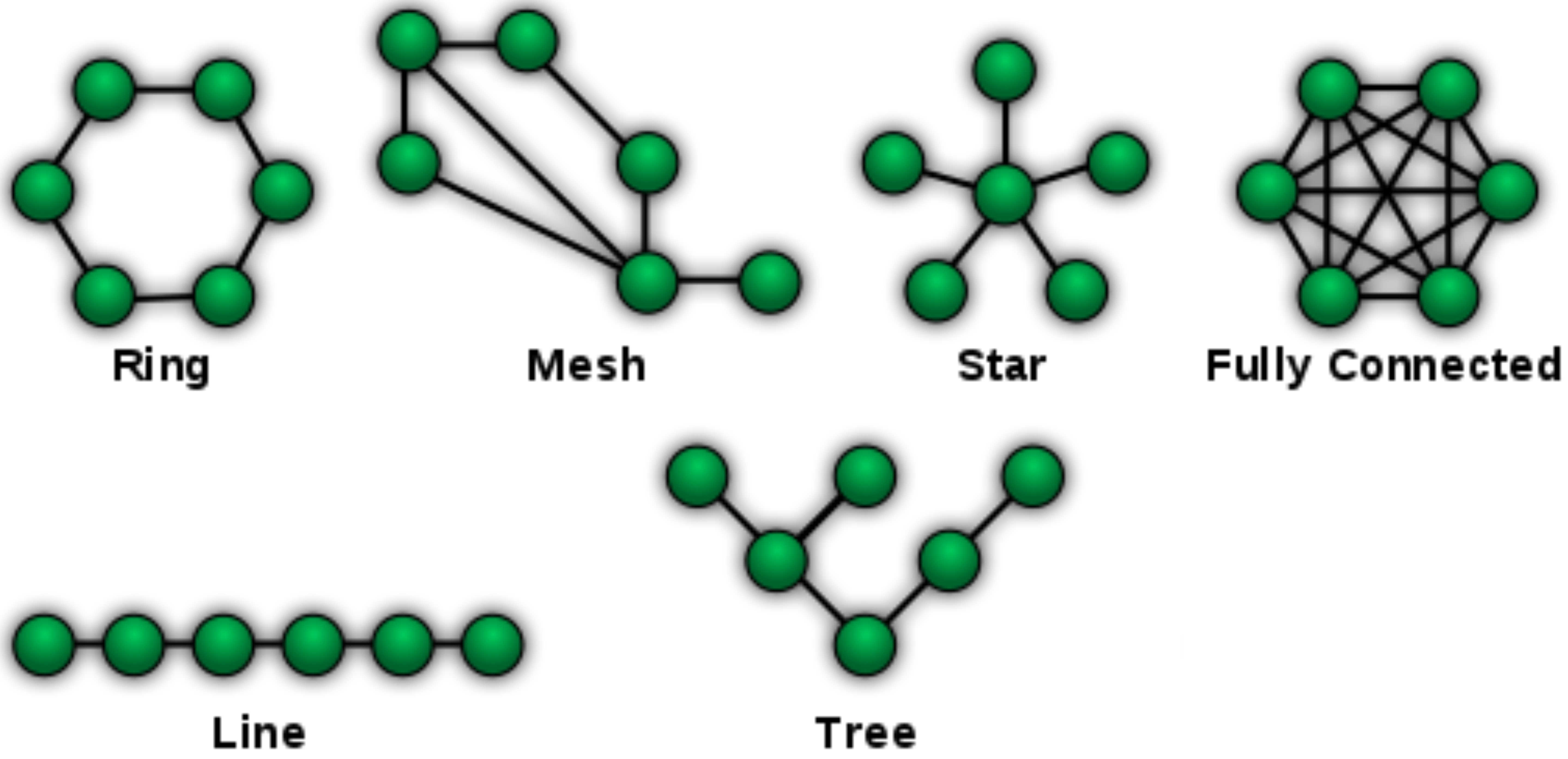
Activity

- We have 16 servers. You can buy as many switches and build as many links as you want. How do you design your network topology?





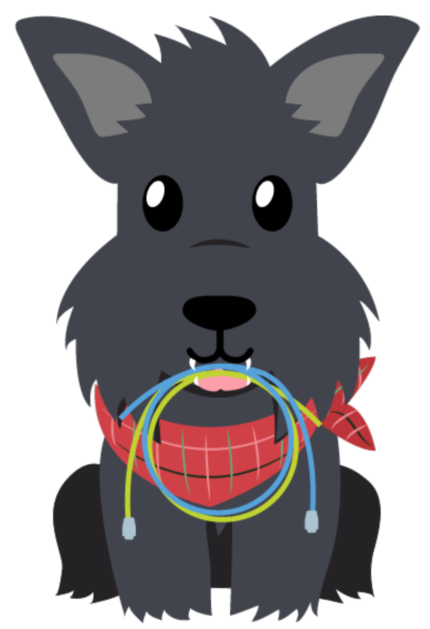
A few “classic” topologies…



What kind of topology are your designs?

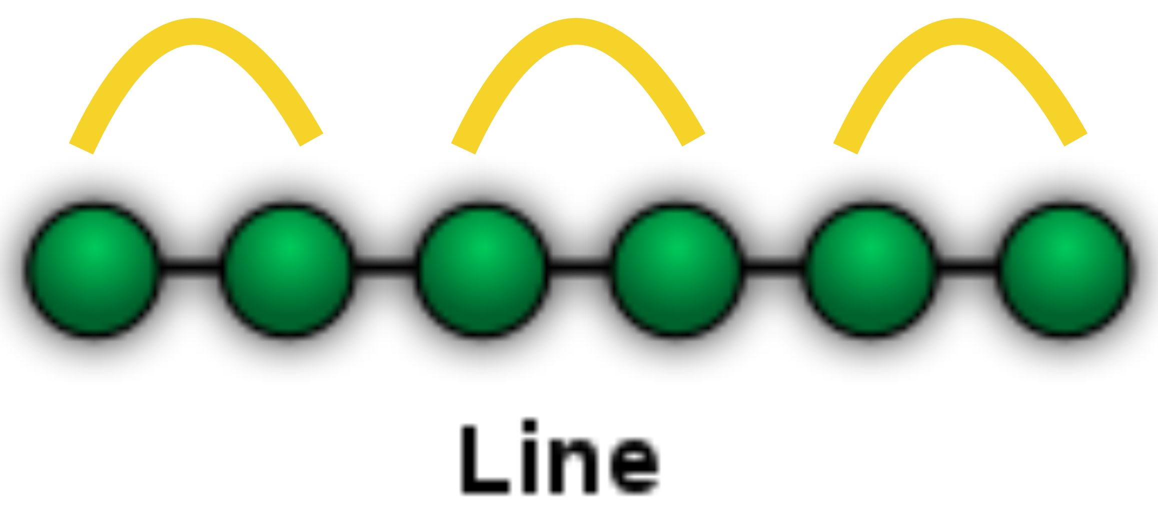
Line Topology

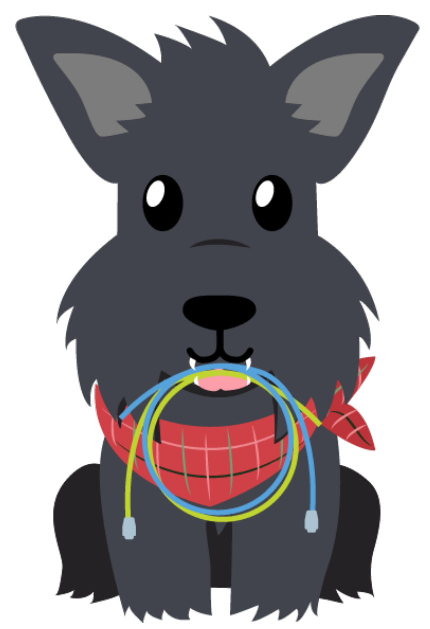
- Simple Design (Easy to Wire)
- Full Reachability
- Bad Fault Tolerance: any failure will partition the network
- High Latency: O(n) hops between nodes
- “Center” Links likely to become bottleneck.



Line Topology

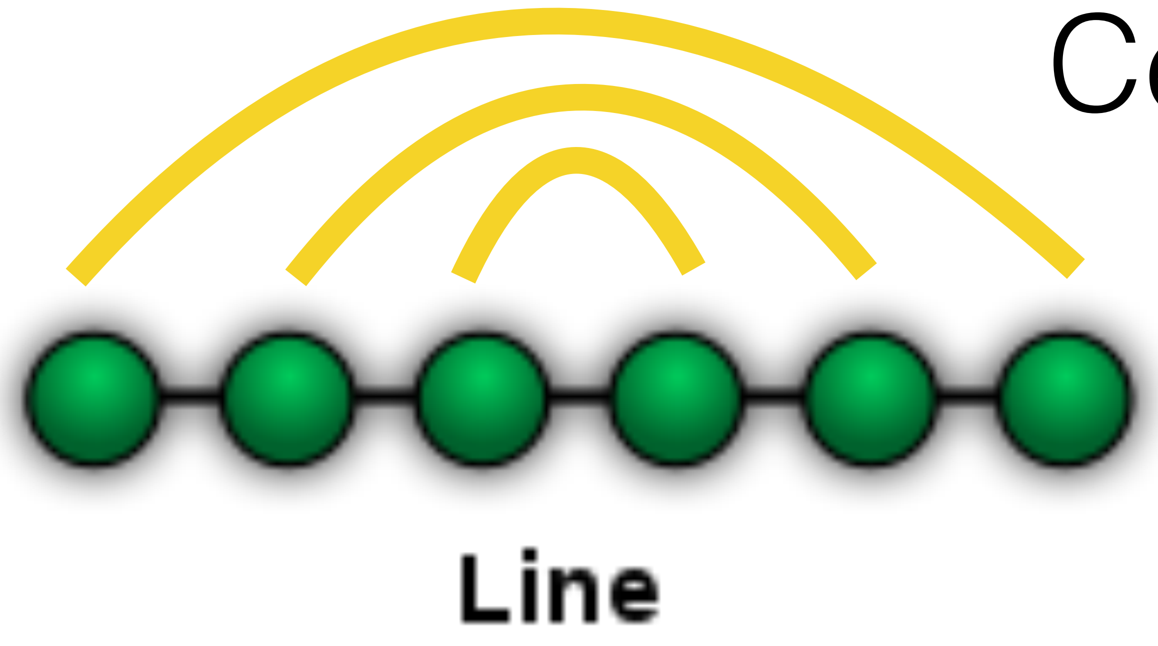
- Simple Design (Easy to Wire)
- Full Reachability
- Bad Fault Tolerance: any failure will partition the network
- High Latency: O(n) hops between nodes
- “Center” Links likely to become bottleneck.





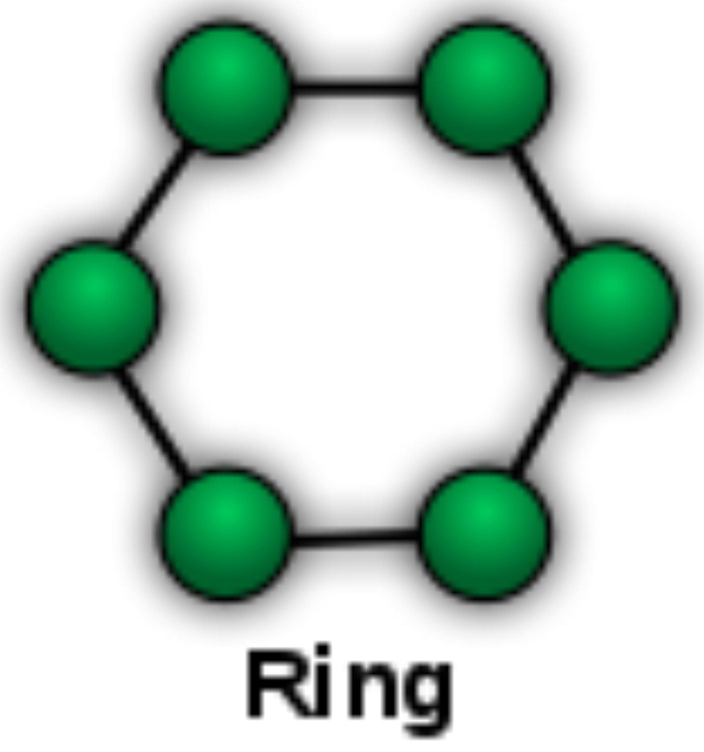
Line Topology

- Simple Design (Easy to Wire)
- Full Reachability
- Bad Fault Tolerance: any failure will partition the network
- High Latency: O(n) hops between nodes
- “Center” Links likely to become bottleneck.



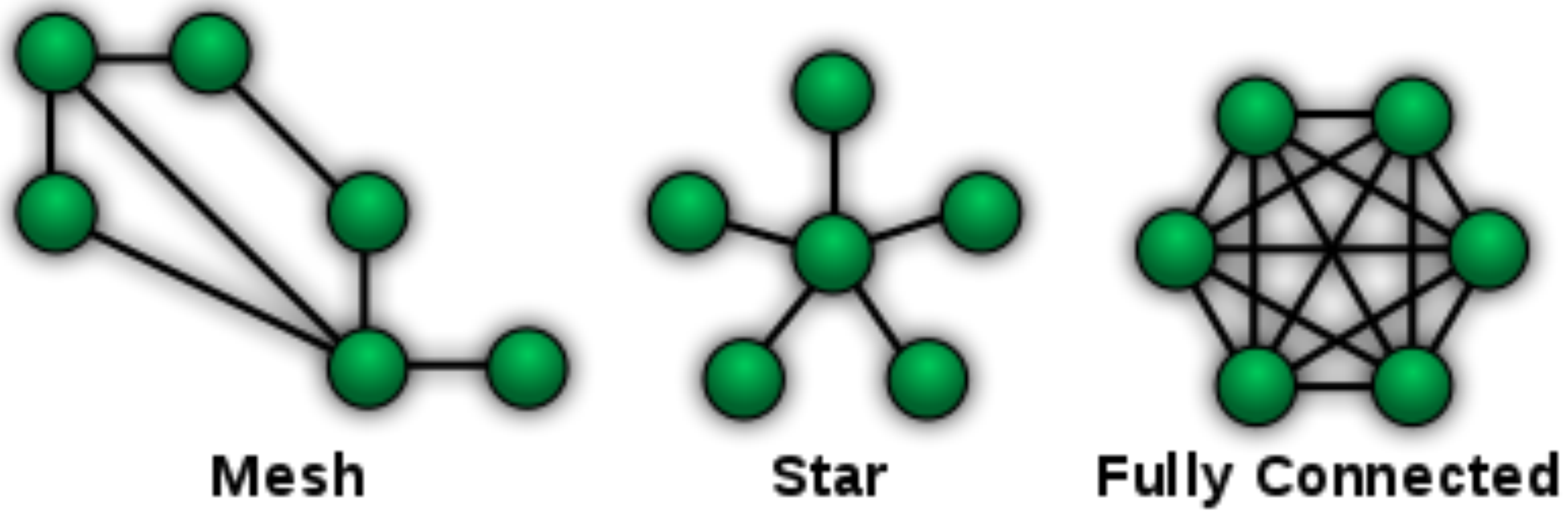
Center link has to support 3x the bandwidth!

Ring Topology

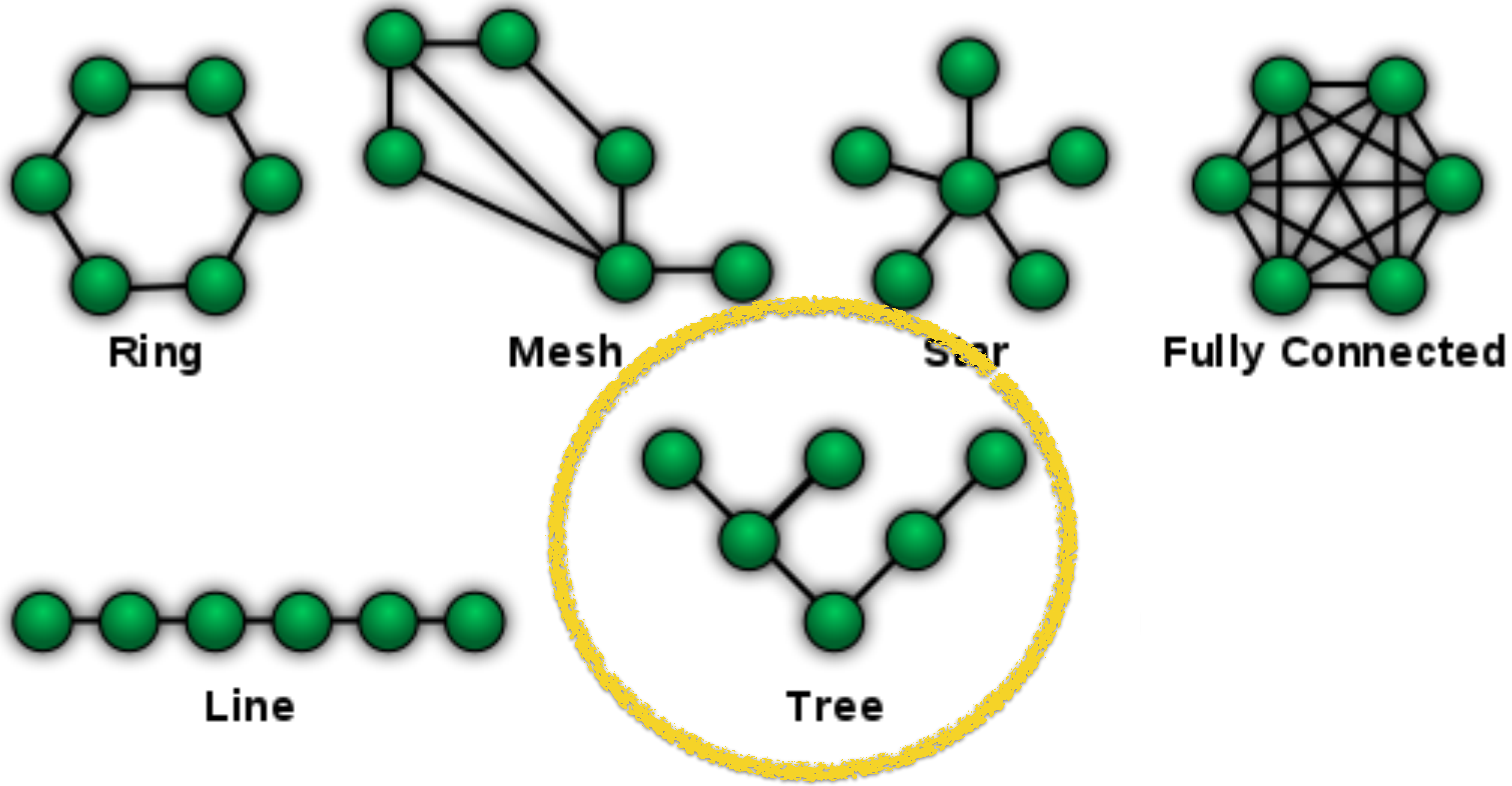


- Simple Design (Easy to Wire)
- Full Reachability
- Better Fault Tolerance (Why?)
- Better, but still not great latency (Why?)
- Multiple paths between nodes can help reduce load on individual links (but still has some bad configurations with lots of paths through one link).

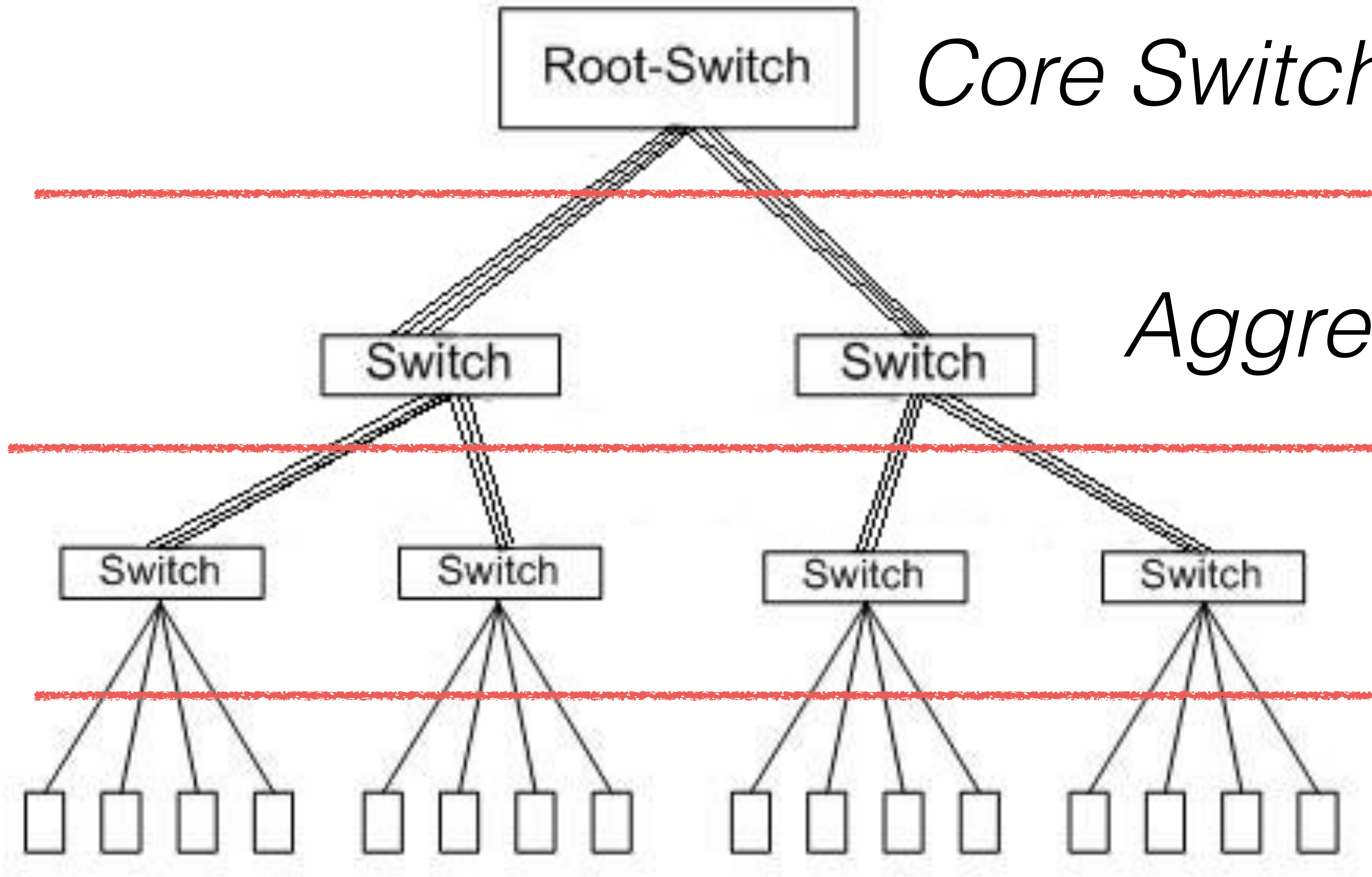
What would you say about these topologies?



In Practice:
Most Datacenters Use Some Form of a Tree Topology



Classic "Fat Tree" Topology



Core Switch (or Switches)

Aggregation Switches

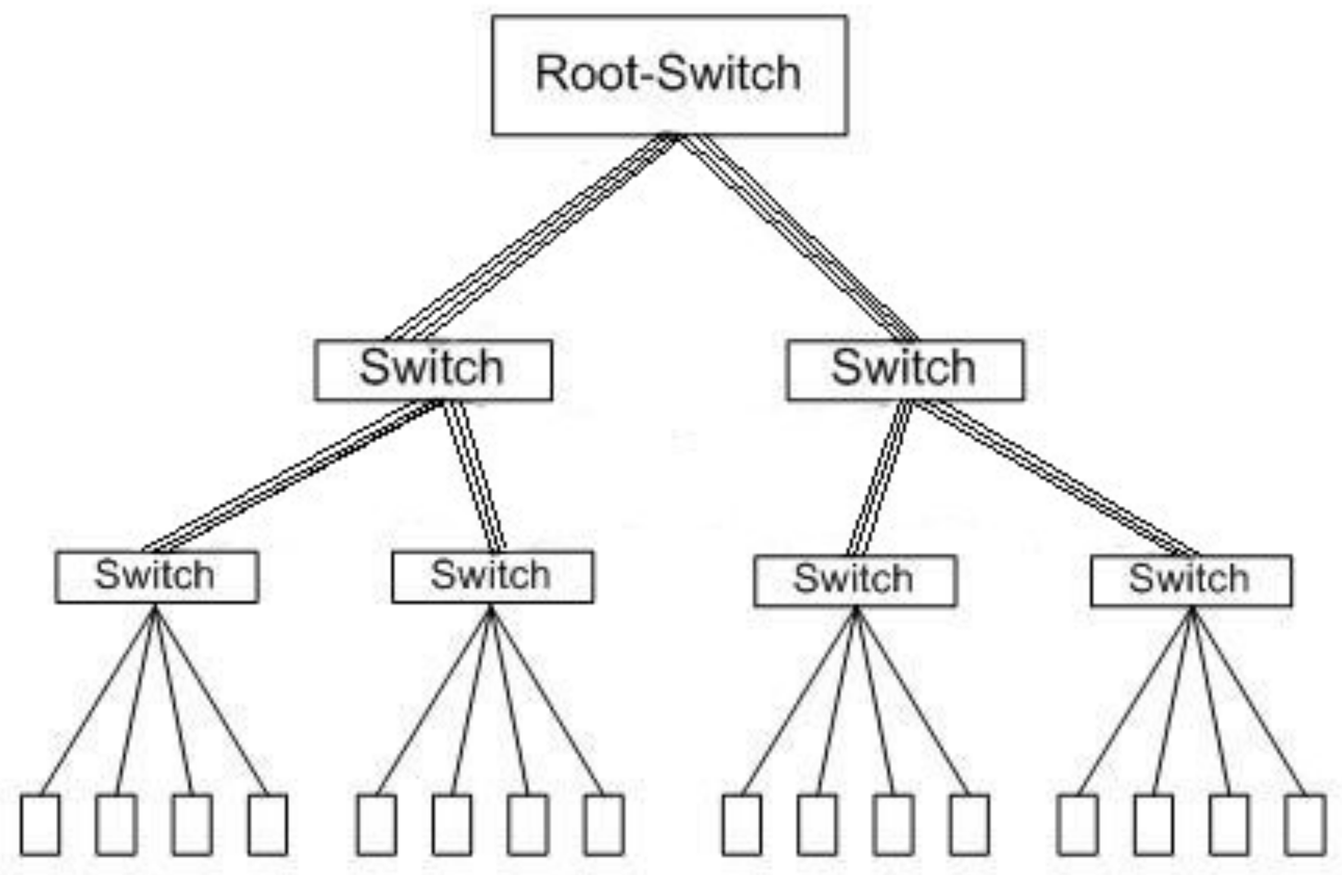
Access (Rack) Switches

Servers

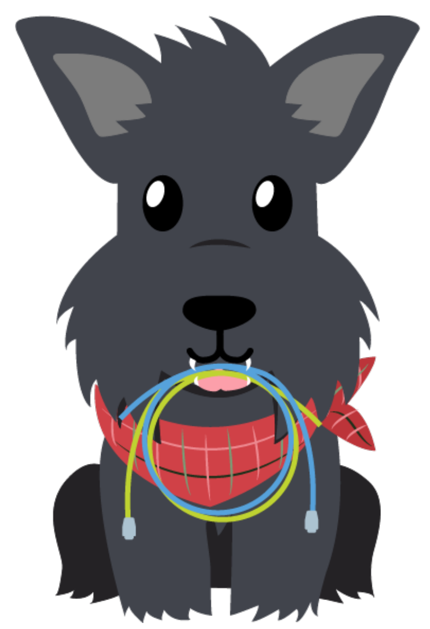
Higher bandwidth links

More expensive switches

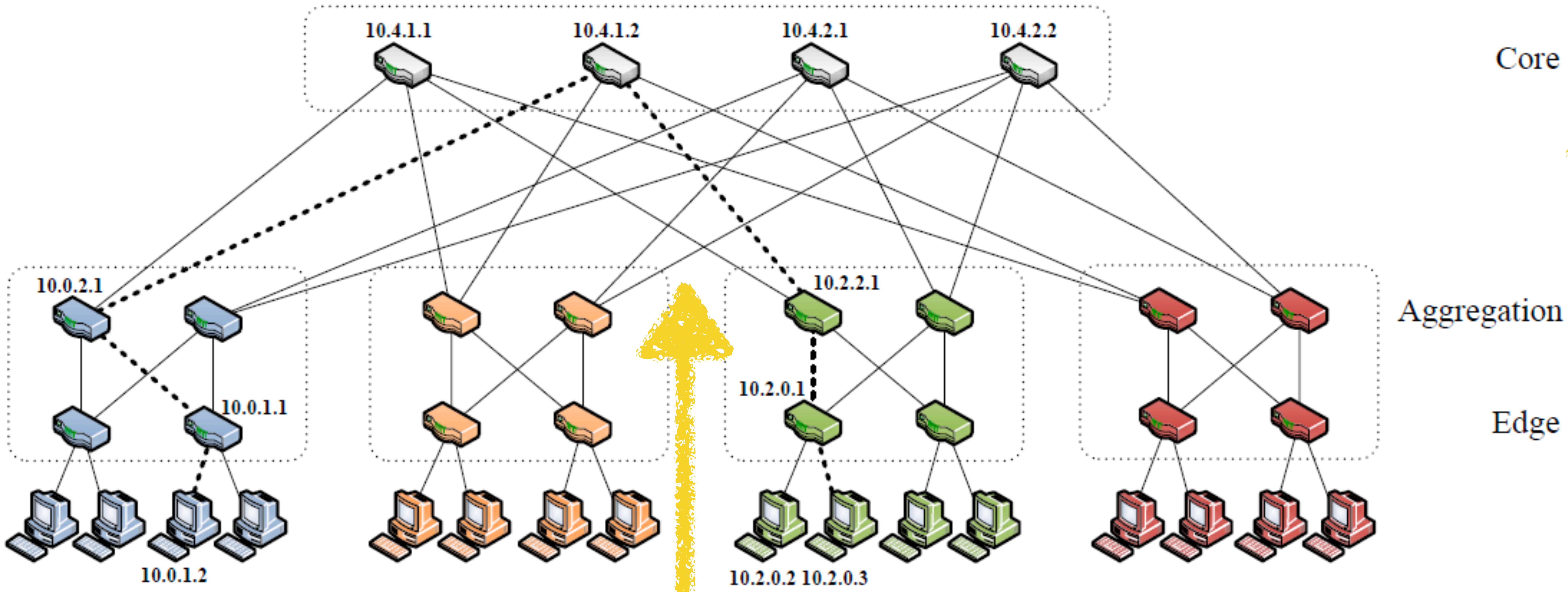
Classic "Fat Tree" Topology



- **Latency:** O(log(n)) hops between arbitrary servers
- **Resilience:** Link failure disconnects subtree — link failures "higher up" cause more damage
- **Throughput:** Lots of endpoints can communicate, all at the same time — due to a few expensive links and switches at the root.
- **Cost-Effectiveness:** Requires some more expensive links and switches, but only at the highest layers of the tree.
- **Easy to Manage:** Clear structure: access -> aggregation -> core

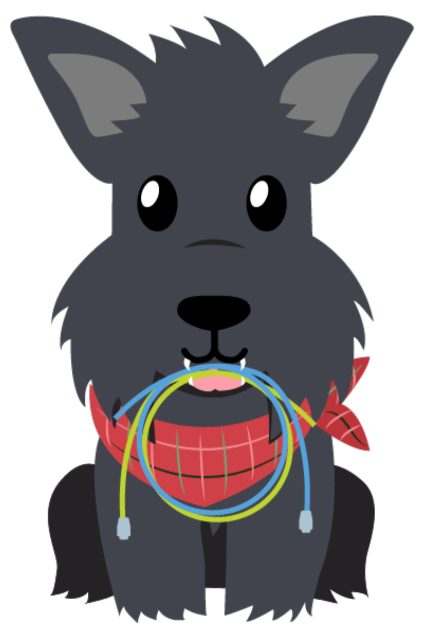


Modern Clos-Style Fat Tree

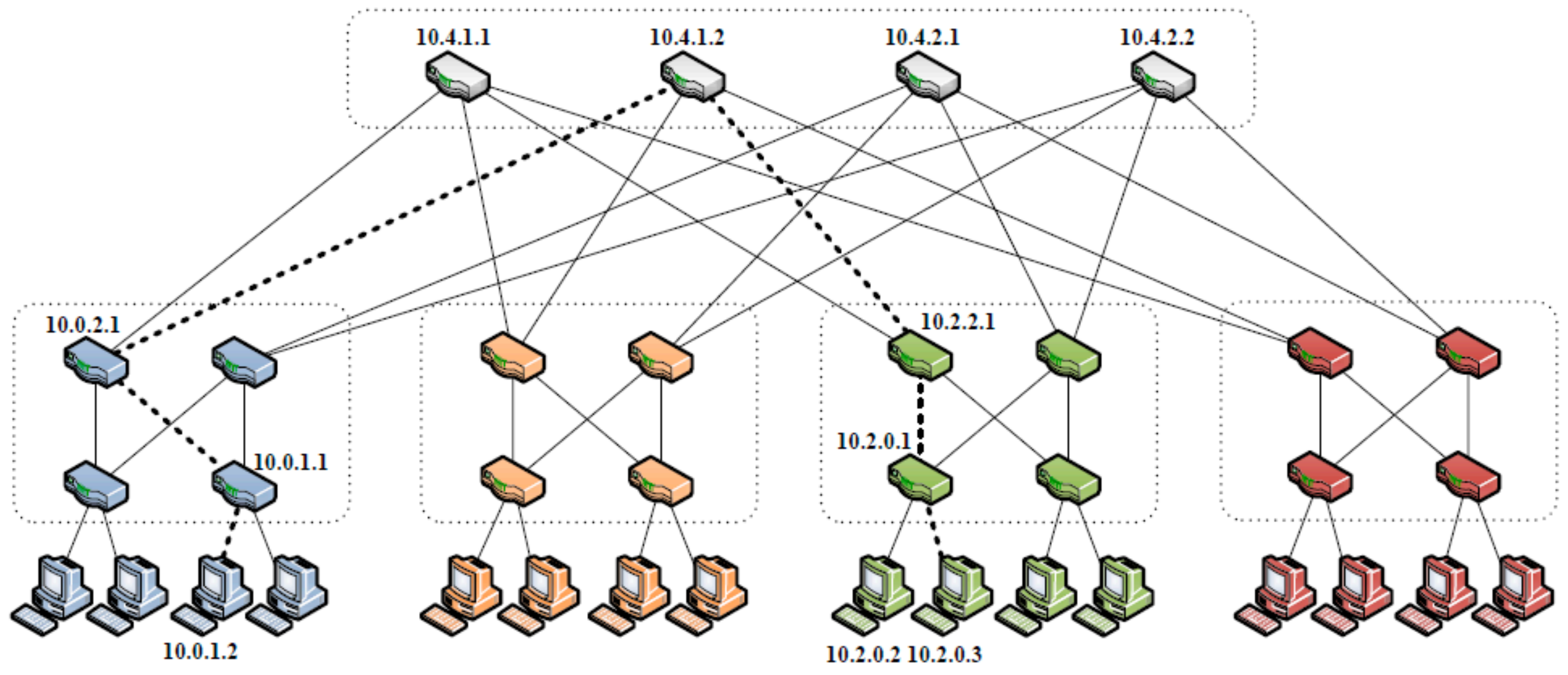


Aggregate bandwidth increases — but all switches and are simple/ relatively low capacity

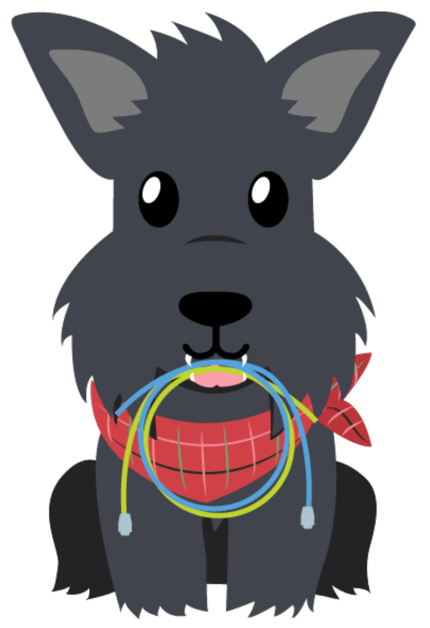
Multiple paths between any pair of servers



Modern Clos-Style Fat Tree



- **Latency:** O(log(n)) hops between arbitrary servers
- **Resilience:** Multiple paths means any individual link failure above access layer won't cause connectivity failure.
- **Throughput:** Lots of endpoints can communicate, all at the same time — due to many cheap paths
- **Cost-Effectiveness:** All switches and links are relatively simple
- **Easy to Manage:** Clear structure… but more links to wire correctly and potentially confuse.



How are datacenter networks different from networks we've seen before?

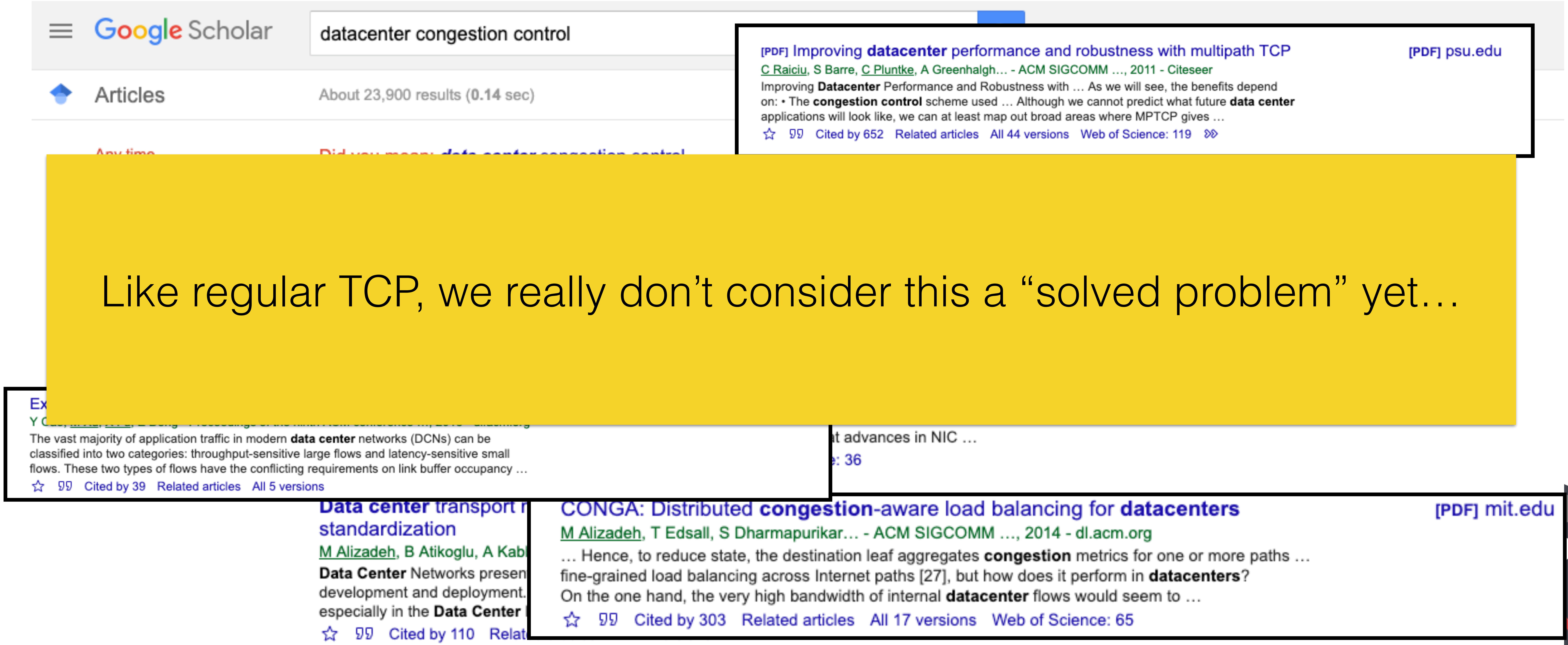
There are *many* ways that datacenter networks differ from the Internet. Today I want to consider these three themes:

1. Topology ✓

2. Congestion Control

3. Virtualization

Datacenter Congestion Control



Like regular TCP, we really don't consider this a "solved problem" yet…

As you work on your CP3 — how might your design change if you were aiming for deployment in a datacenter rather than on the Internet?

Just one of many problems: Mice, Elephants, and Queueing

Short messages

(e.g., query, coordination) → **Low Latency**

Large flows

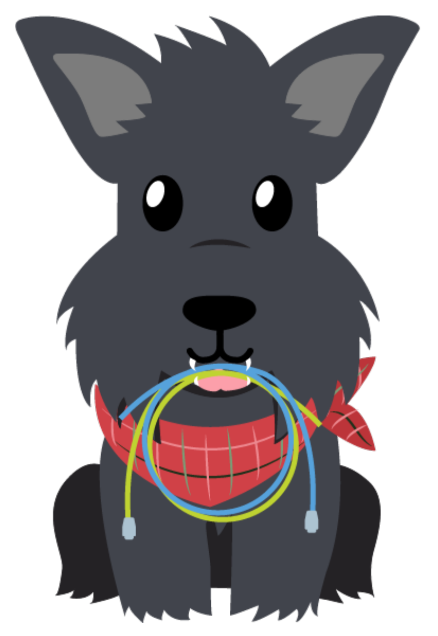
(e.g., data update, backup) → **High Throughput**

Think about applications: what are "mouse" connections and what are "elephant" connections?

Have you ever tried to play a video game while your roommate is torrenting?

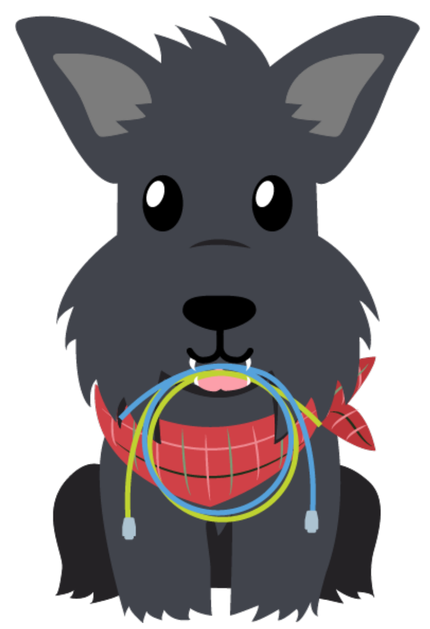
Small, latency-sensitive connections

Long-lived, large transfers

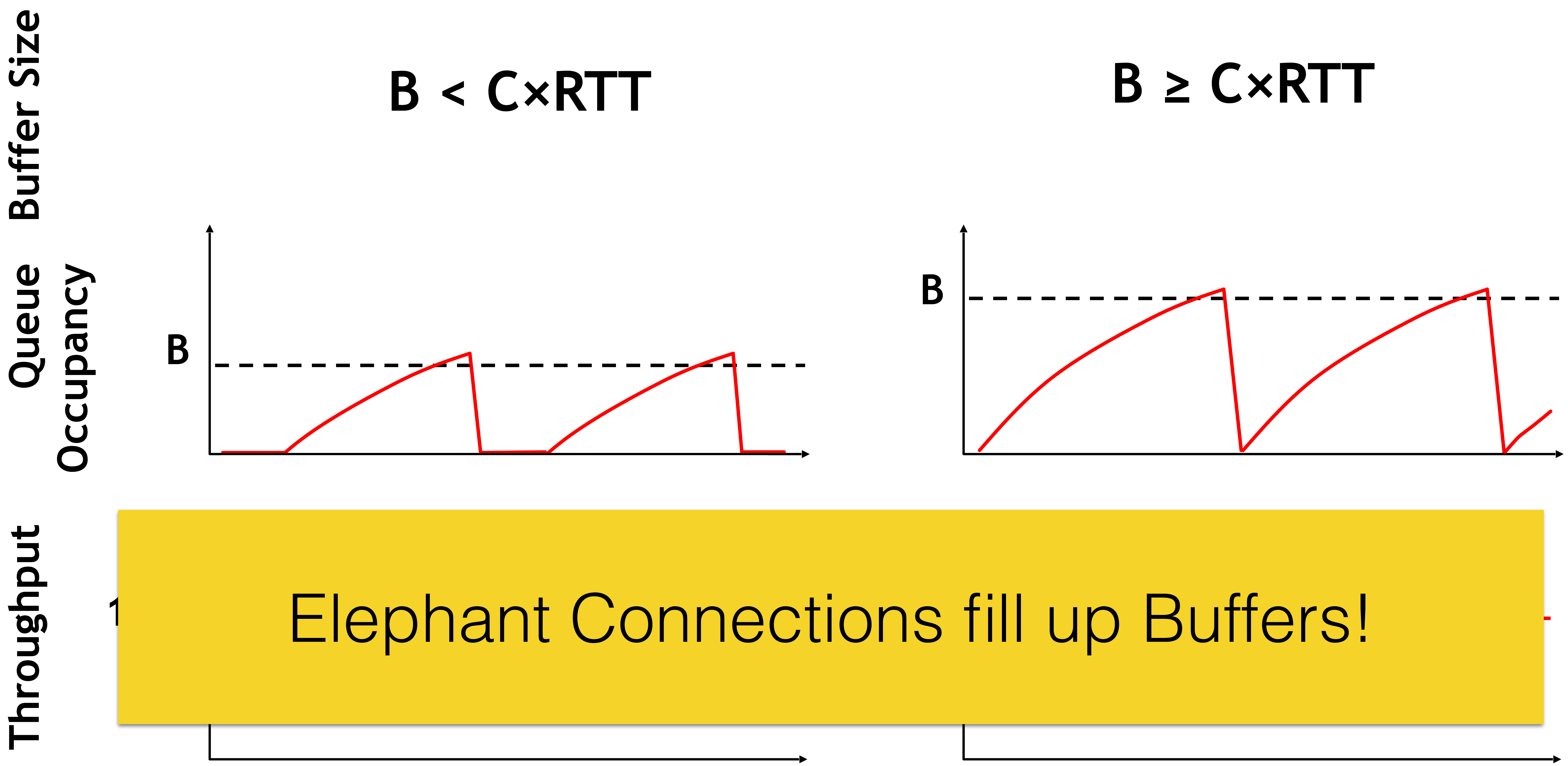


In the Datacenter

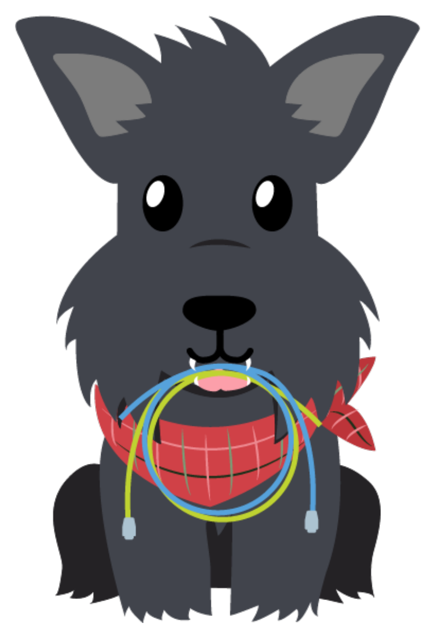
- Latency Sensitive, Short Connections:
  - How long does it take for you to load google.com? Perform a search? These things are implemented with short, fast connections between servers.
- Throughput Consuming, Long Connections:
  - Facebook hosts billions of photos, YouTube gets 300 hours of new videos uploaded every day! These need to be transferred between servers, thumbnails and new versions created and stored.
  - Furthermore, everything must be backed up 2-3 times in case a hard drive fails!



TCP Fills Buffers — and needs them to be big to guarantee high throughput.

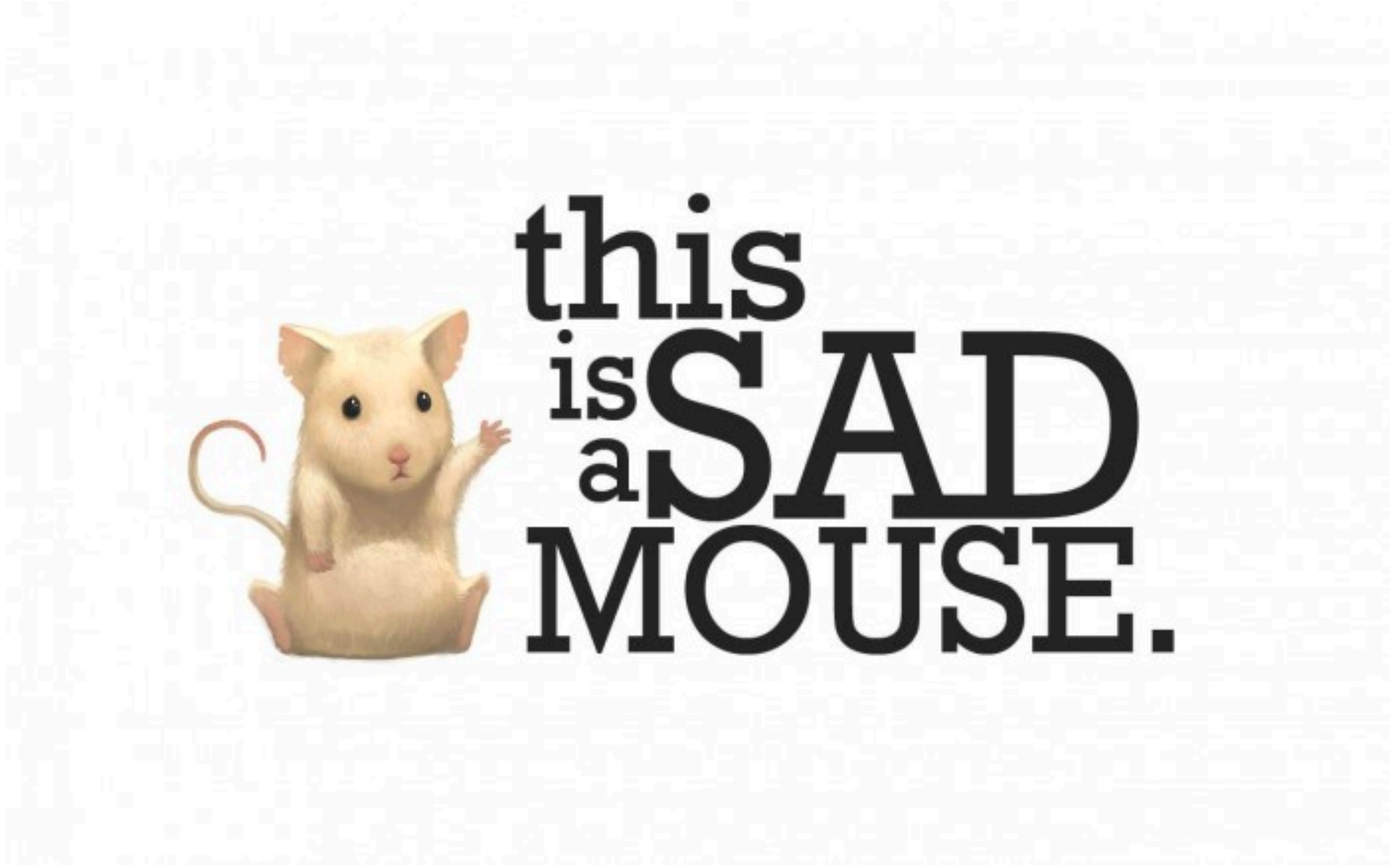


Elephant Connections fill up Buffers!



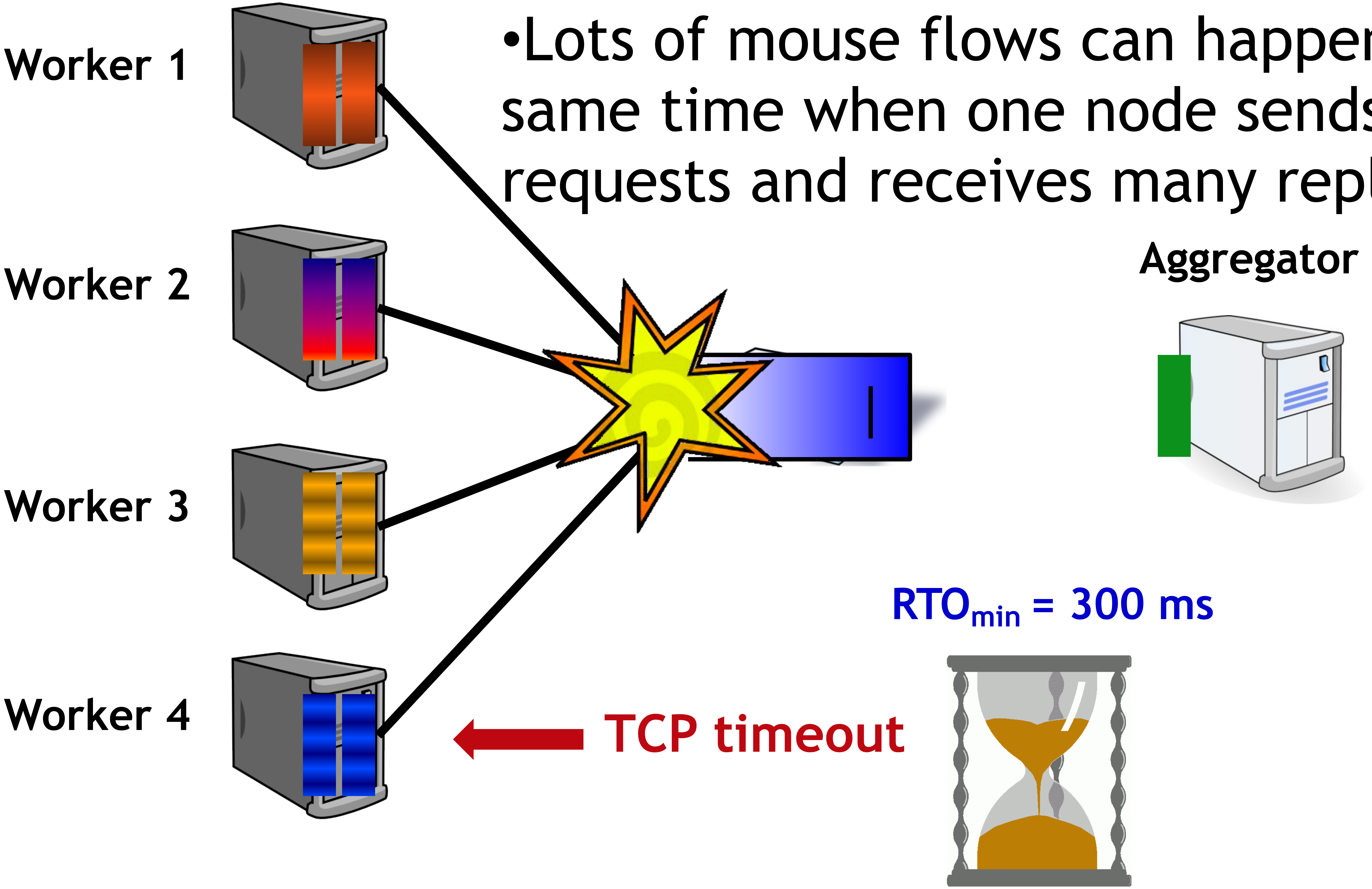
Full Buffers are Bad for Mice

- Why do you think this is?
- Full buffers increase latency! Packets have to wait their turn to be transmitted.
    - Datacenter latencies are only 10s of microseconds!
- Full buffers increase loss! Packets have to be retransmitted after a full round trip time (under fast retransmit) or wait until a timeout (even worse!)



Incast: Really Sad Mice!

- Lots of mouse flows can happen at the same time when one node sends many requests and receives many replies at once!



When the queue is already full, even more packets are lost and timeout!

How do we keep buffers empty to help mice flows — but still allow big flows to achieve high throughput?

Ideas?

A few approaches

- **Microsoft [DCTCP, 2010]:** Before they start dropping packets, routers will “mark” packets with a special congestion bit. The fuller the queue, the higher the probability the router will mark each packet. Senders slow down proportional to how many of their packets are marked.

- **Google [TIMELY, 2015]:** Senders track the latency through the network using very fine grained (nanosecond) hardware based timers. Senders slow down when they notice the latency go up.

Why can’t we use these TCPs on the Internet?

I can't wait to test your TCP implementations next week!

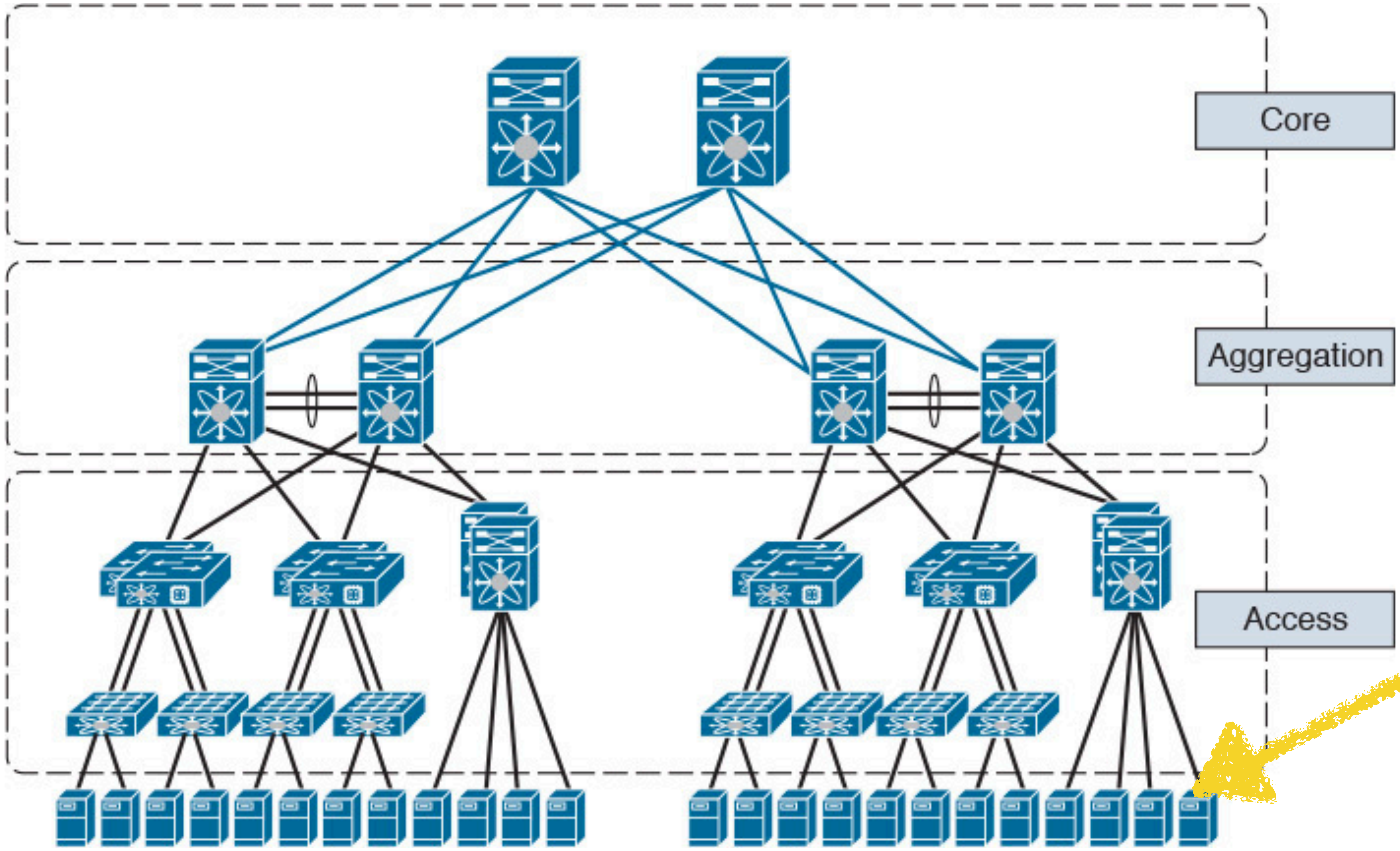
How are datacenter networks different from networks we’ve seen before?

There are *many* ways that datacenter networks differ from the Internet. Today I want to consider these three themes:

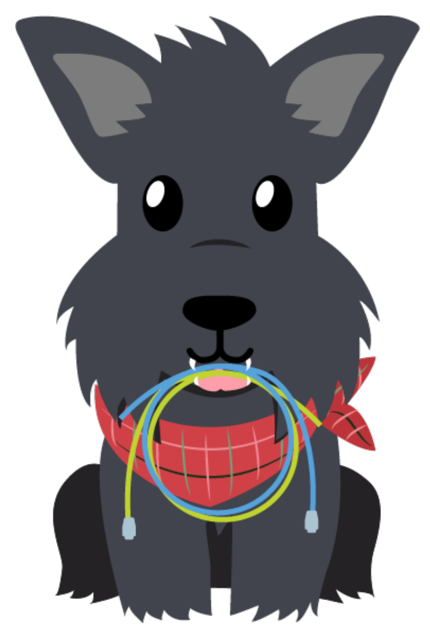
1. Topology ✓
2. Congestion Control ✓
3. Virtualization

Later…

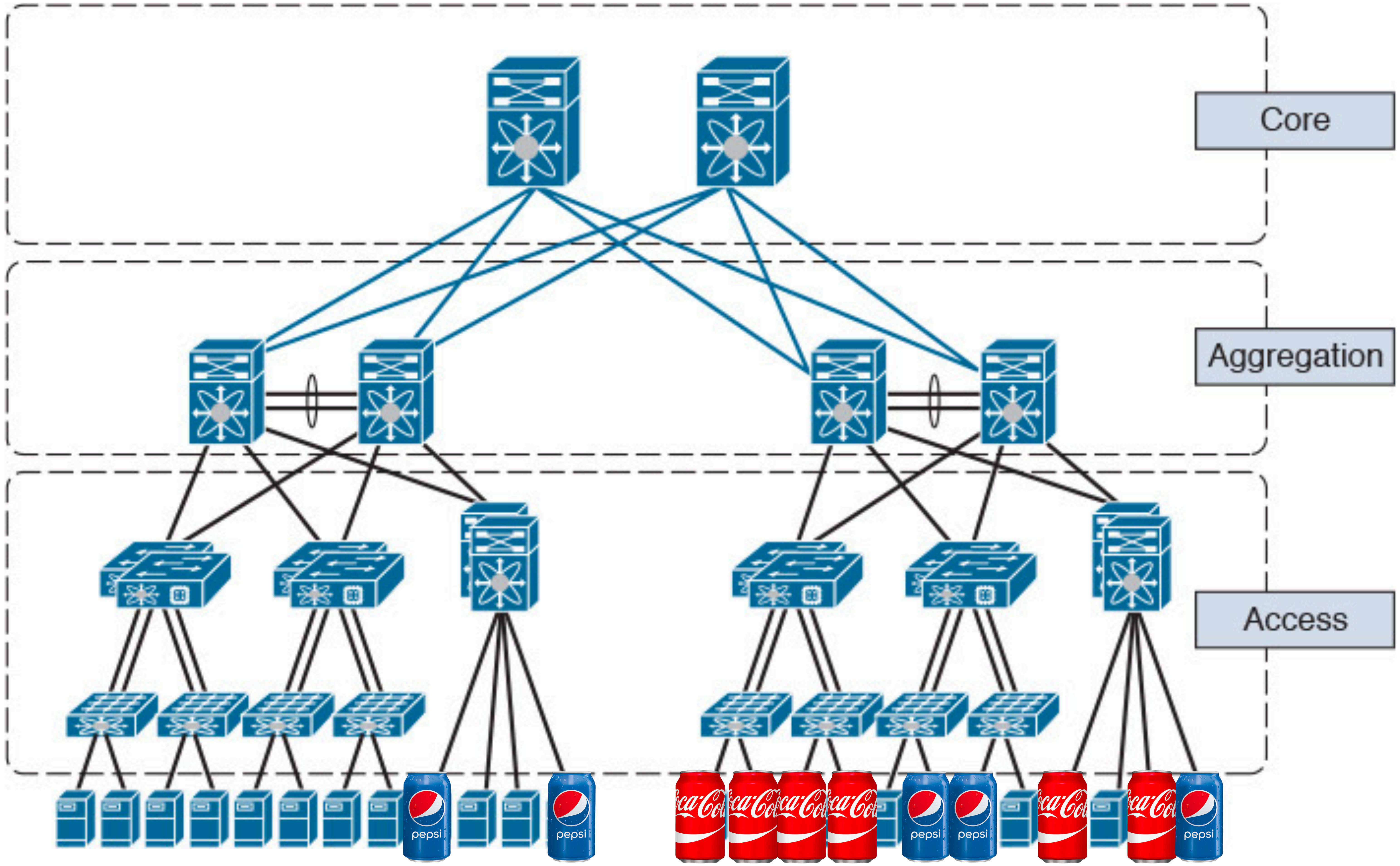
Imagine you are AWS or Azure



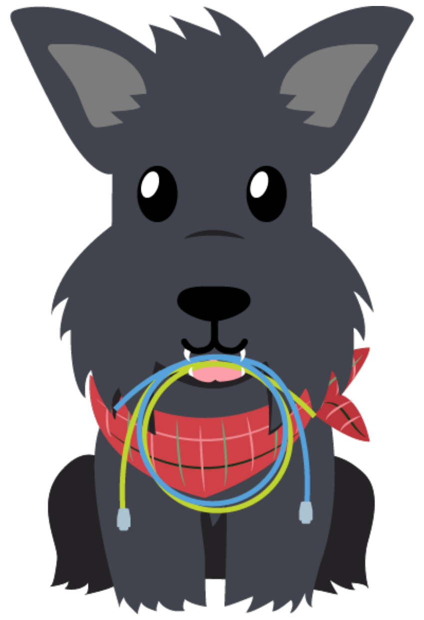
You rent out these servers



Imagine you are AWS or Azure



Meet your new customers



Um… hey….!

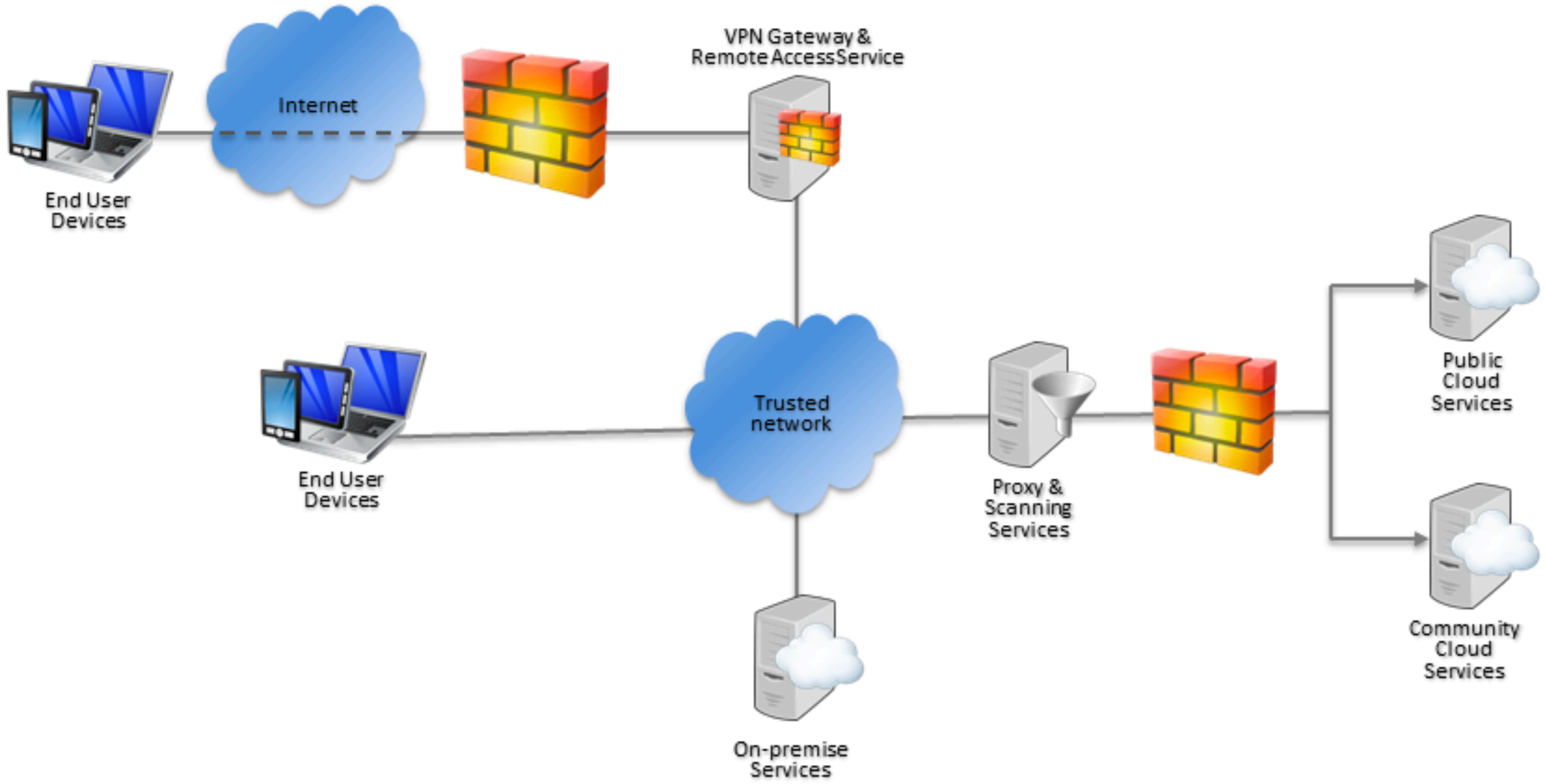
I'm gonna DDoS your servers and knock you offline!

I have a new 0day attack and am going to infiltrate your machines!

Isolation: the ability for multiple users or applications to share a computer system without interference between each other

Here comes the new kid…

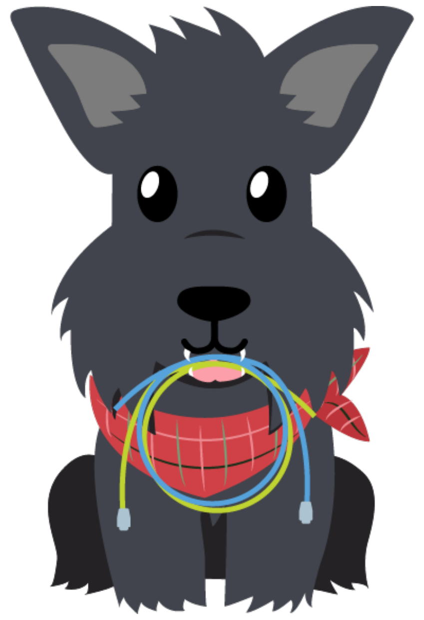
I want to move my servers to your cloud, but I have a complicated set of firewalls and proxies in my network — how do I make sure traffic is routed through firewalls and proxies correctly in your datacenter?



Emulation: the ability of a computer program in an electronic device to emulate (or imitate) another program or device

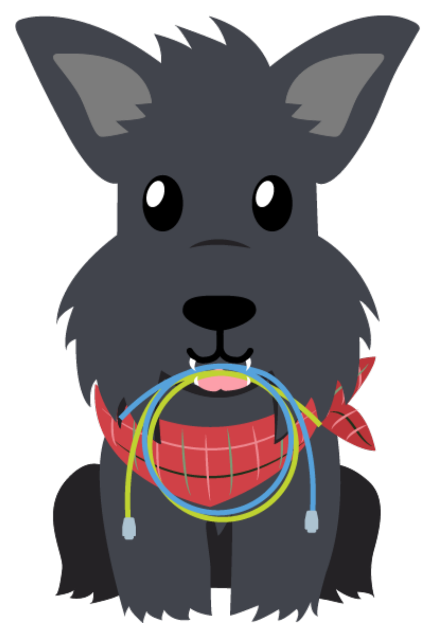

SO TELL ME WHAT YOU WANT, WHAT YOU REALLY, REALLY WANT

virtualization refers to the act of creating a virtual (rather than actual) version of something, including virtual computer hardware platforms, storage devices, and computer network resources.



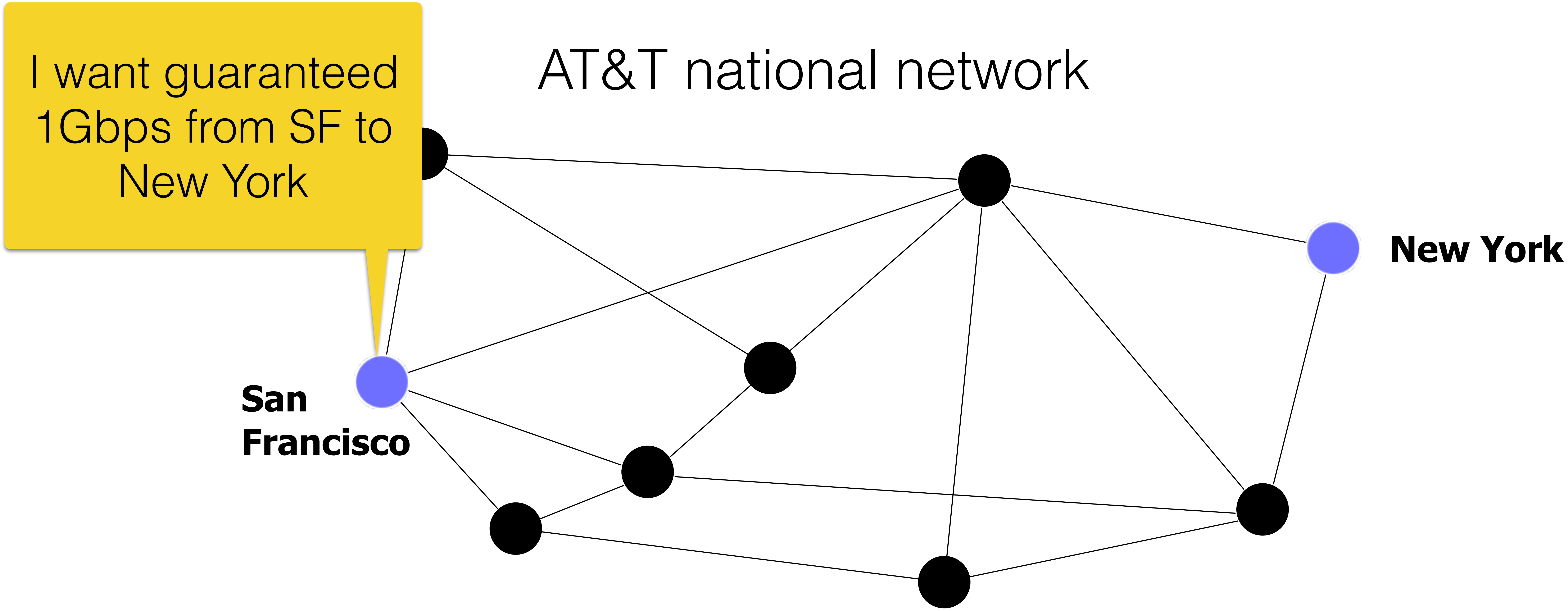
Virtualization provides *isolation* between users and *emulation* for each user — as if they each had their own private network.

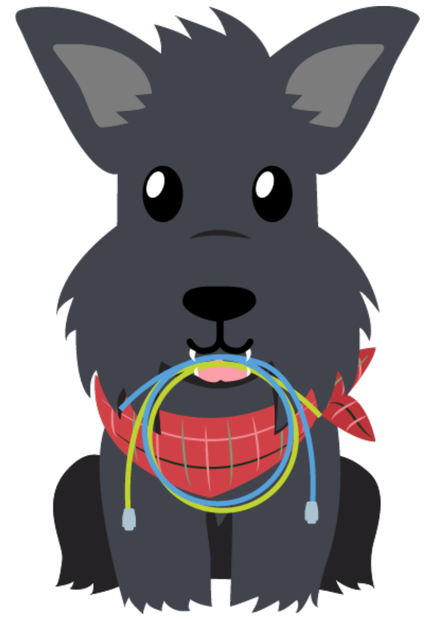
Makes a shared network feel like everyone has their own personal network.



Virtualization in Wide Area Networks: MPLS

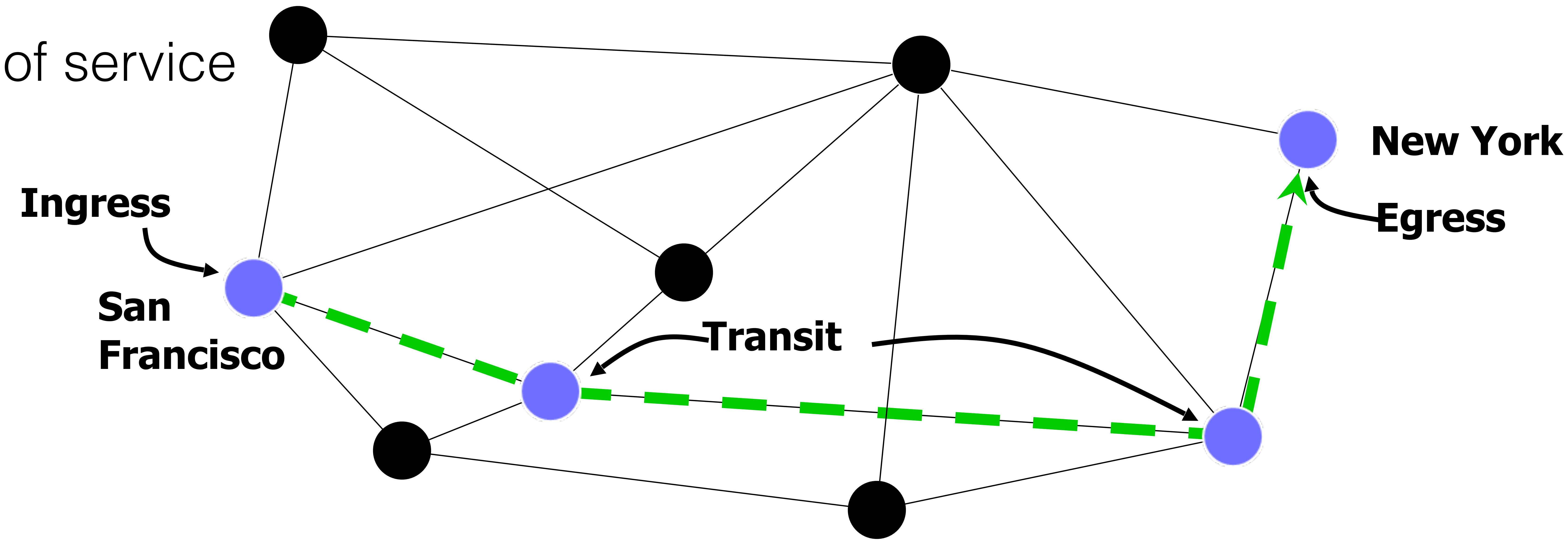
Wide Area Virtualization: MPLS





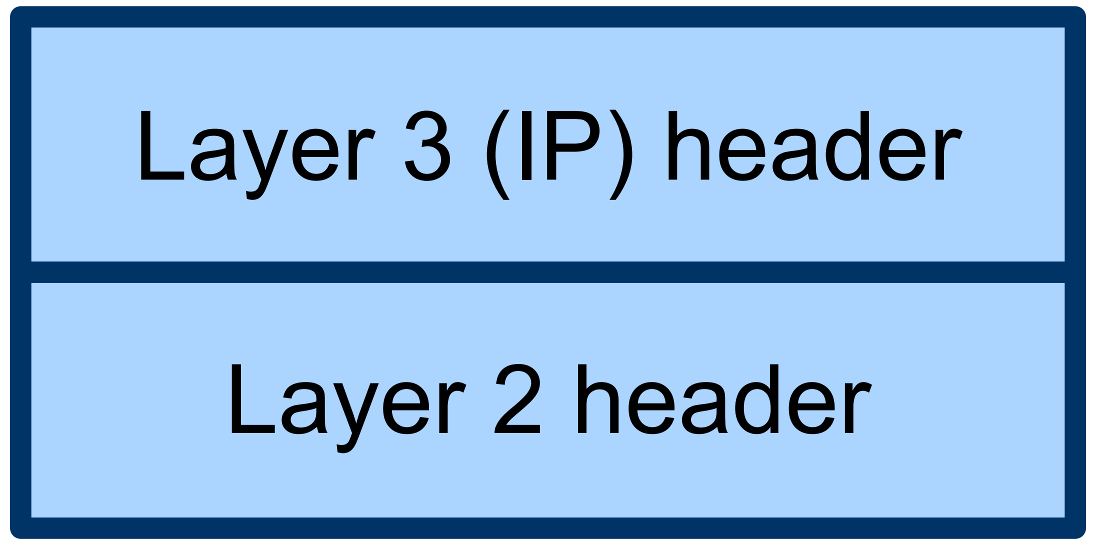
Label Switched Path (LSP)

- Fixed, one-way path through interior network
- Driven by multiple forces
  - Traffic engineering
  - High performance forwarding
  - VPN
  - Quality of service

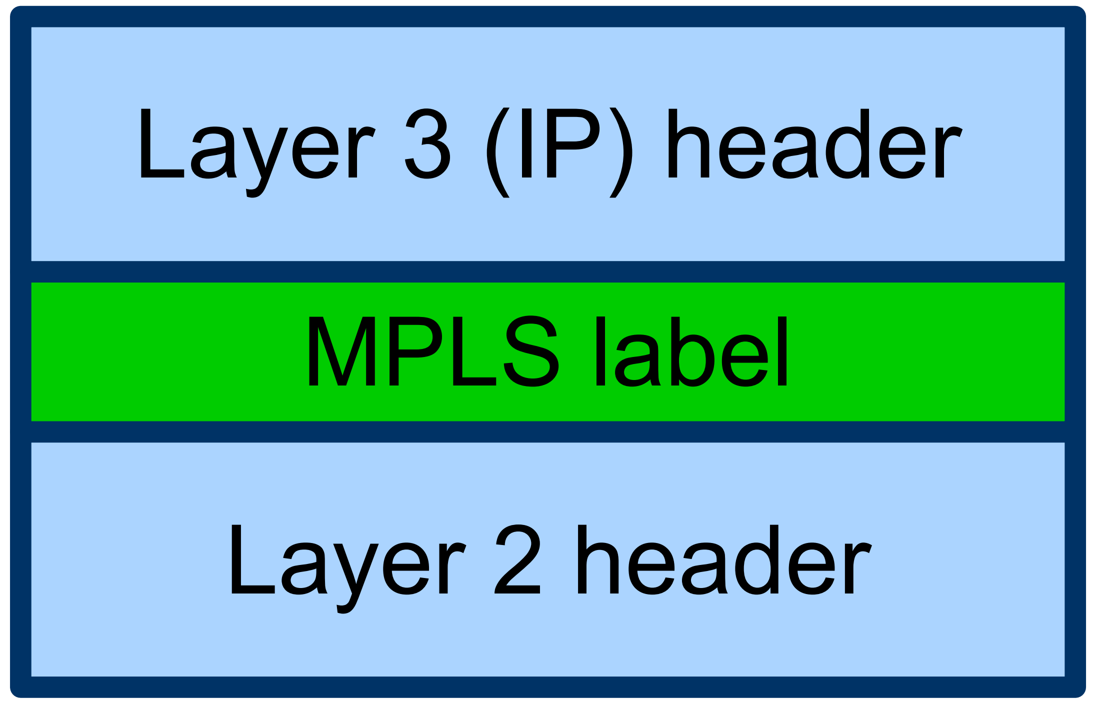


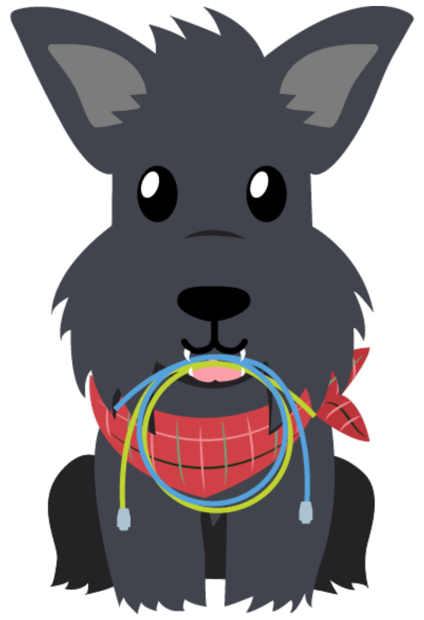
Label Switching: Just add a new header!

- Key idea “virtual circuit”
  - Remember circuit switched network?
  - Want to emulate a circuit.

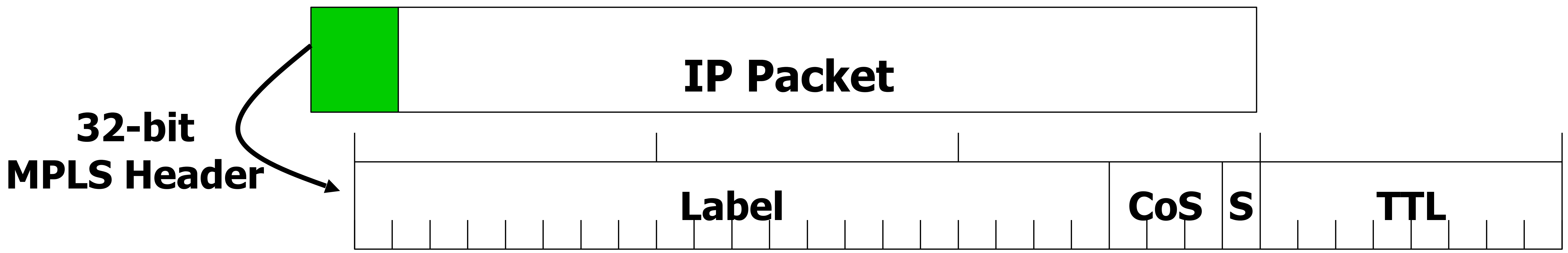


- Packets forwarded by “label-switched routers” (LSR)
  - Performs LSP setup and MPLS packet forwarding
  - Label Edge Router (LER): LSP ingress or egress
  - Transit Router: swaps MPLS label, forwards packet

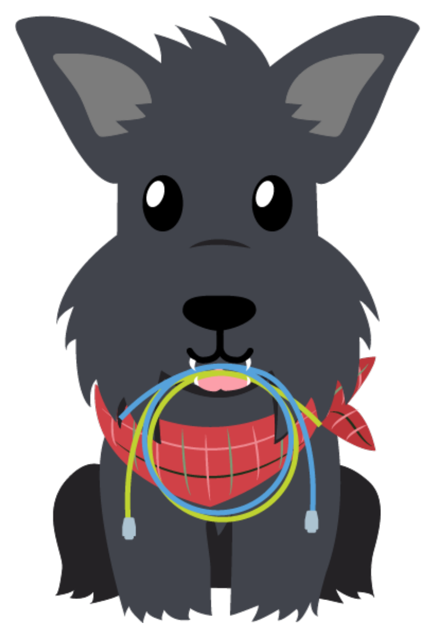




MPLS Header

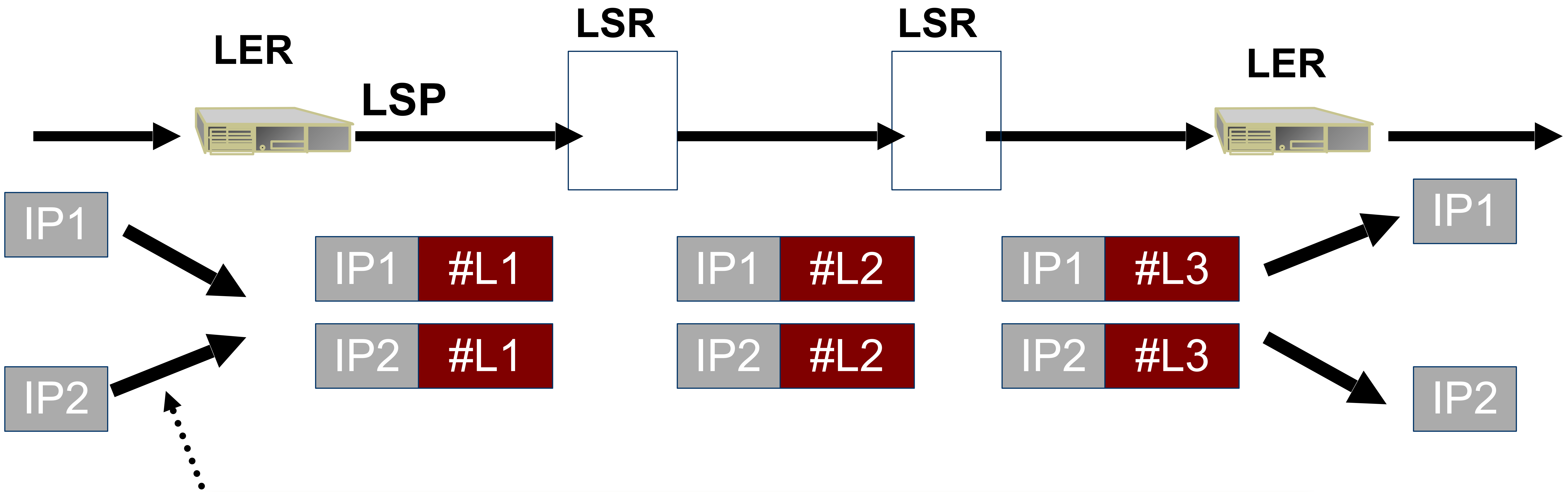


- IP packet is encapsulated in MPLS header
  - Label
  - Class of service
  - Stacking bit: if next header is an MPLS header
  - Time to live: decremented at each LSR, or pass through
- IP packet is restored at end of LSP by egress router
  - TTL is adjusted, transit LSP routers count towards the TTL
- MPLS is an optimization – does not affect IP semantics



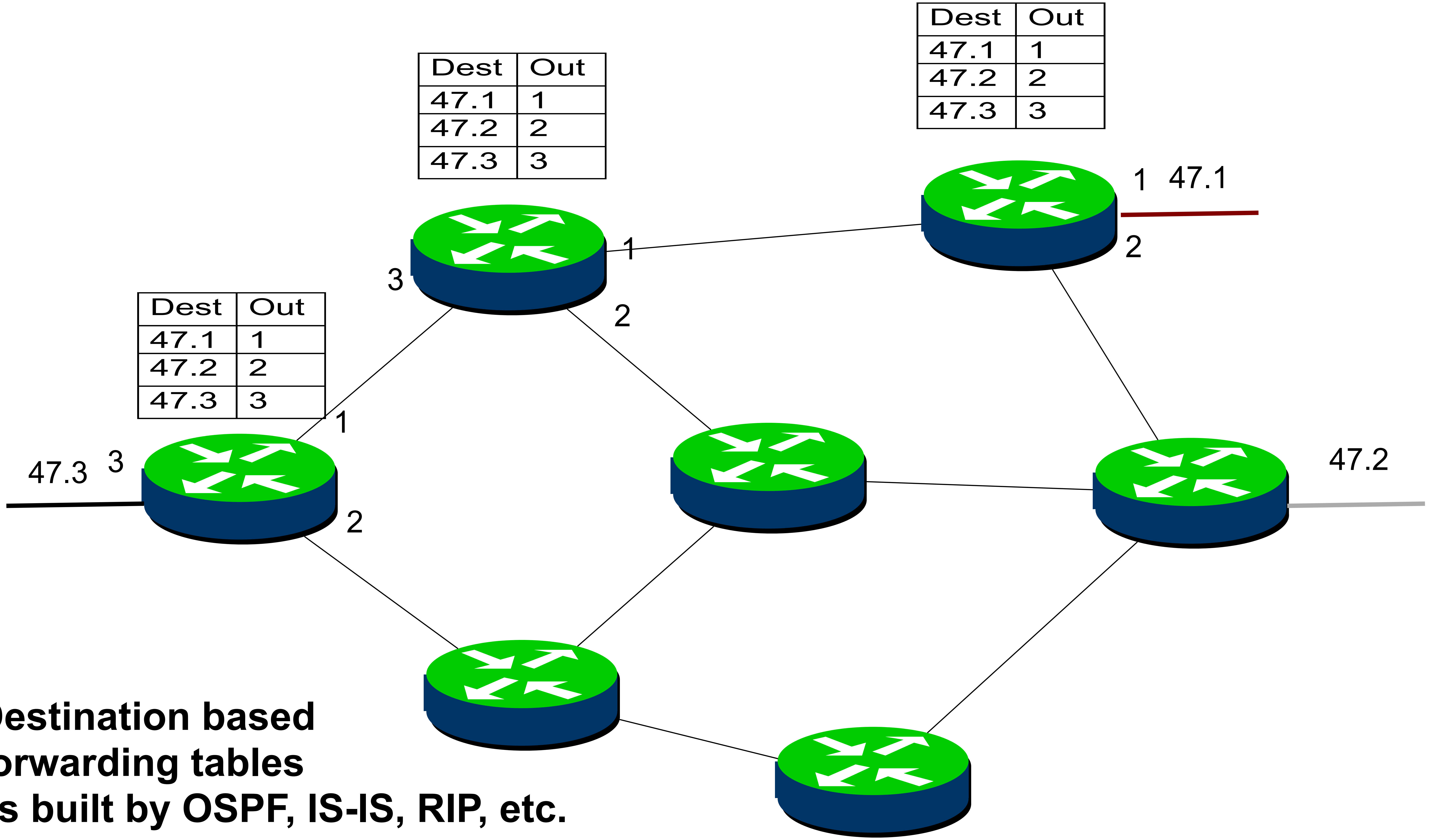
Forwarding Equivalence Classes

FEC = "A subset of packets that are all treated the same way by a LSR"



Packets are destined for different address prefixes, but can be mapped to common path

MPLS Builds on Standard IP



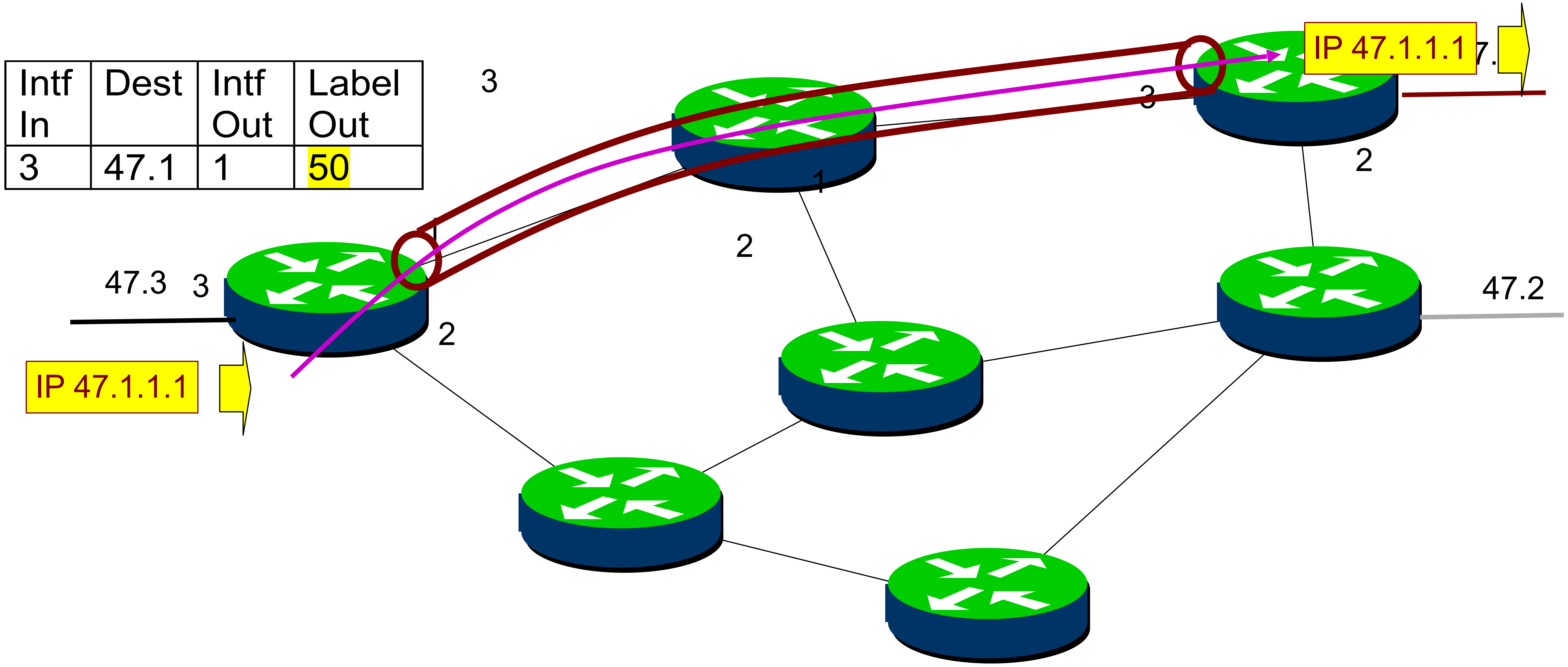
Destination based forwarding tables as built by OSPF, IS-IS, RIP, etc.

Label Switched Path (LSP)

Intf In	Dest	Intf Out	Label Out
3	47.1	1	50

Intf In	Label In	Dest	Intf Out	Label Out
3	50	47.1	1	40

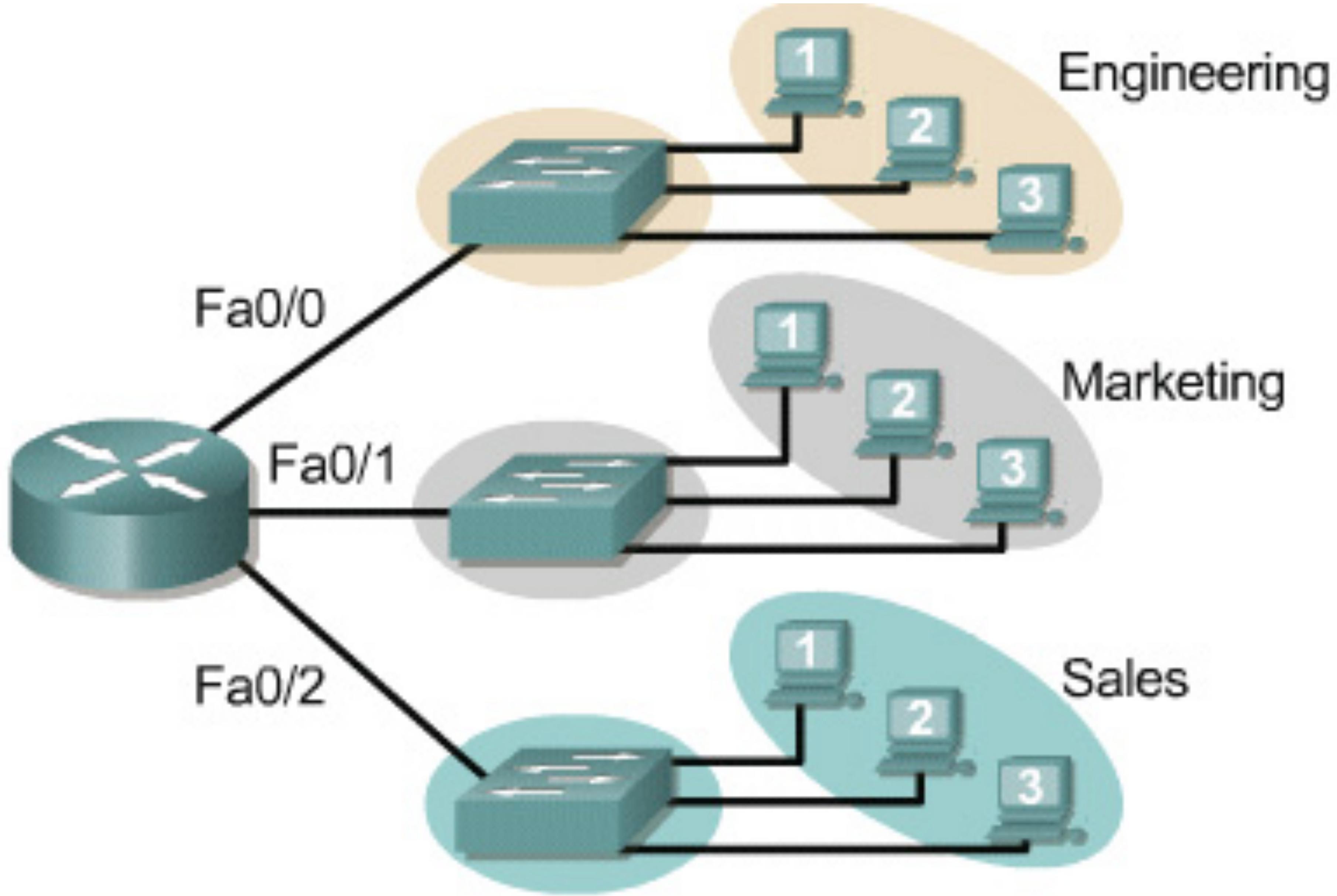
Intf In	Label In	Dest	Intf Out
3	40	47.1	1



Virtualization in Local Area Networks: “Virtual LANs”

Broadcast domains with VLANs and routers

Layer 3 routing allows the router to send packets to the three different broadcast domains.



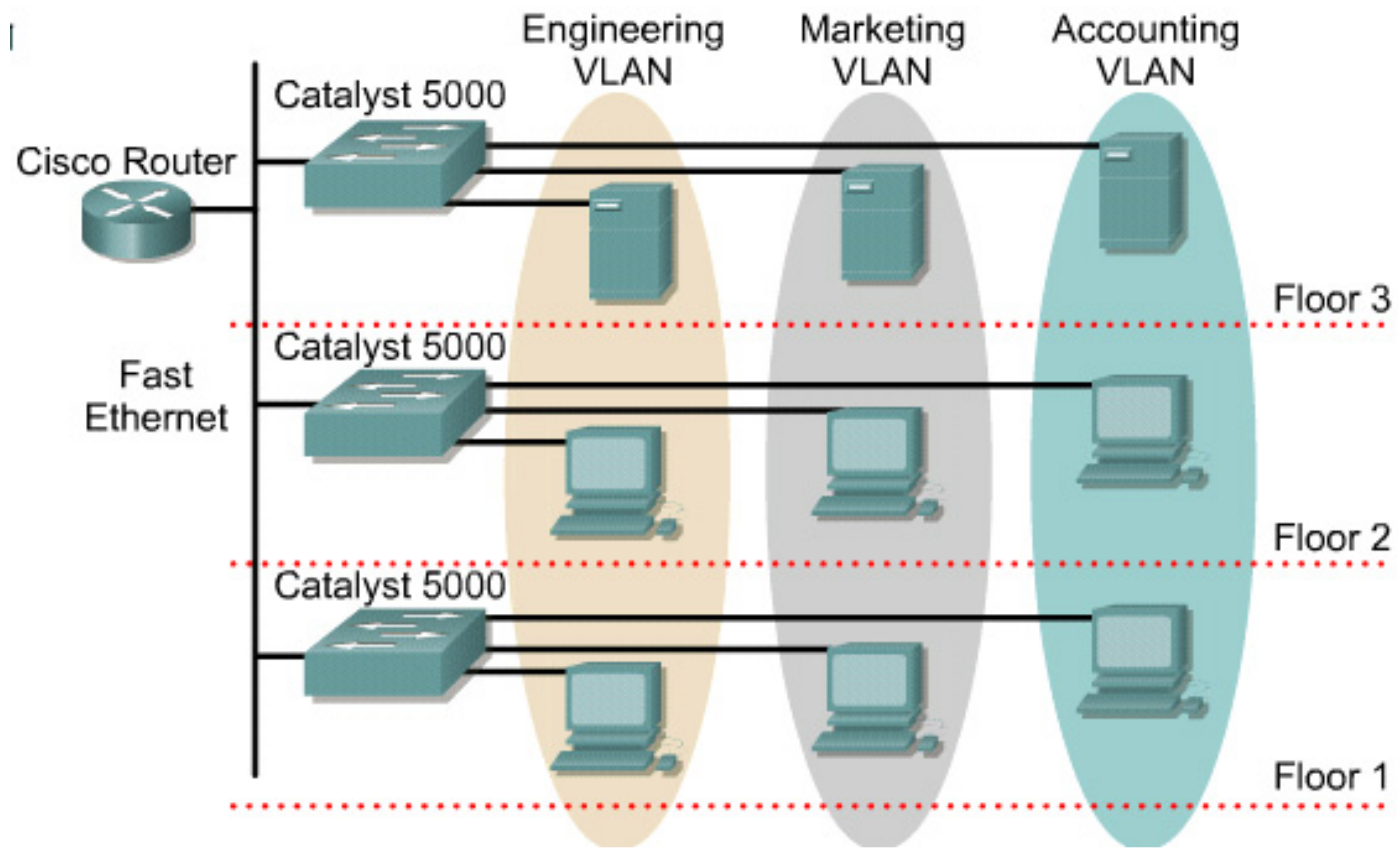
Three switches and one router could be used without VLANs:

- Switch for Engineering
- Switch for Sales
- Switch for Marketing
- Each switch treats all ports as members of one broadcast domain
- Router is used to route packets among the three broadcast domains

VLAN introduction

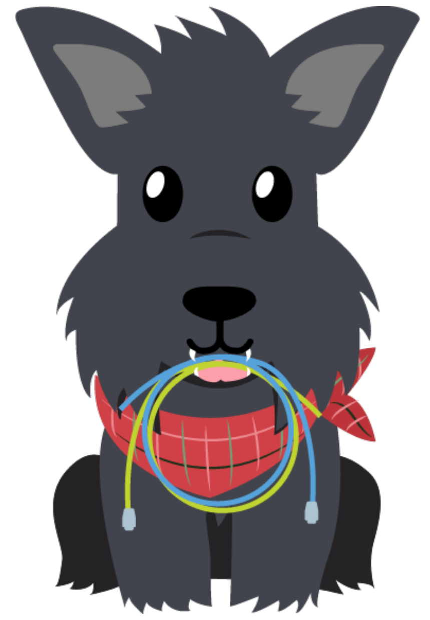
VLANs function by logically segmenting the network into different broadcast domains so that packets are only switched between ports that are designated for the same VLAN.

Routers in VLAN topologies provide broadcast filtering, security, and traffic flow management.



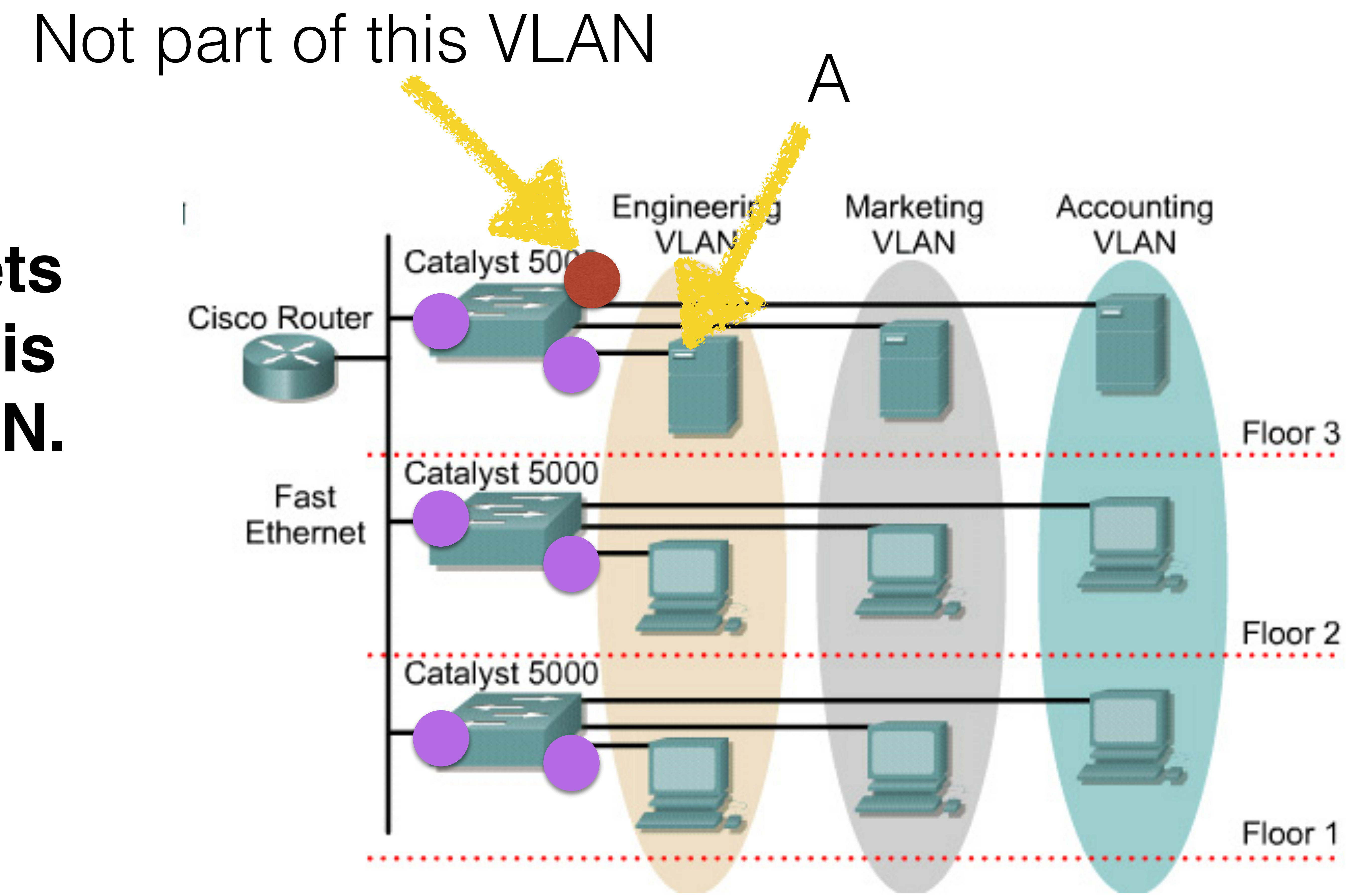
How do we achieve this? Headers!

MPLS Wraps entire packet in a new header to give a “label”.
VLANs add a new field to Ethernet specifying the VLAN ID.



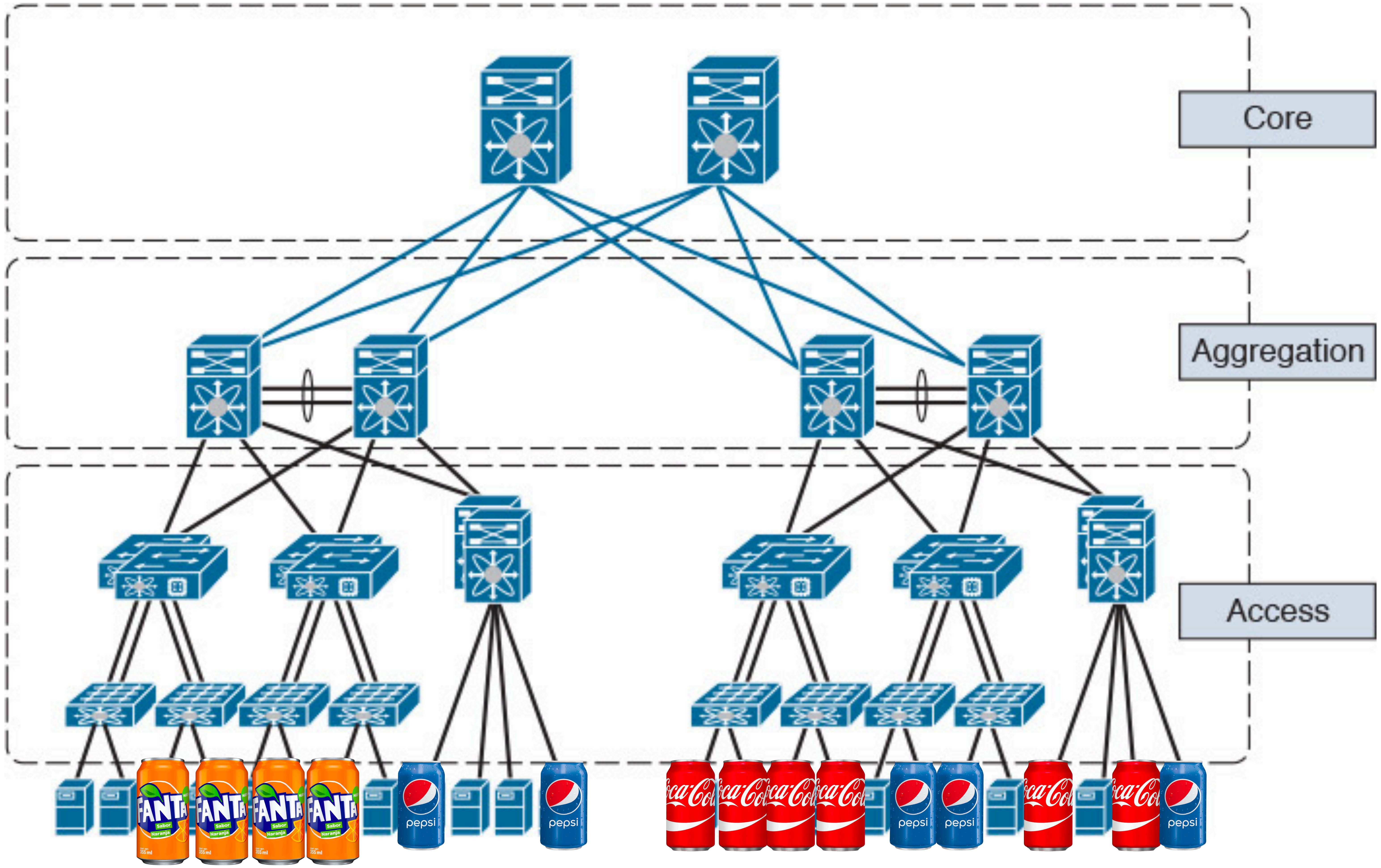
How do I let A broadcast to all other engineering nodes?

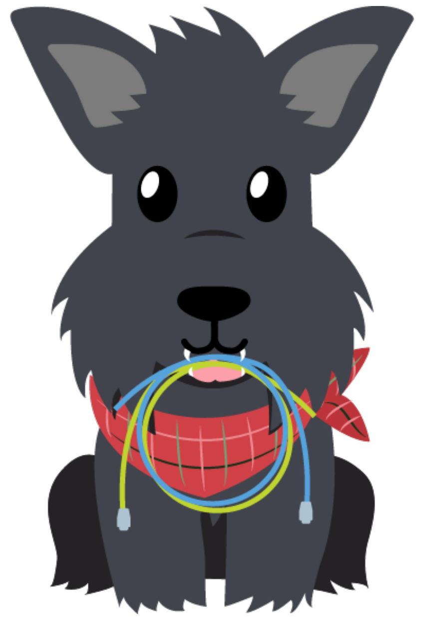
Broadcast packets on any port that is part of a the VLAN.



Back to our Datacenter

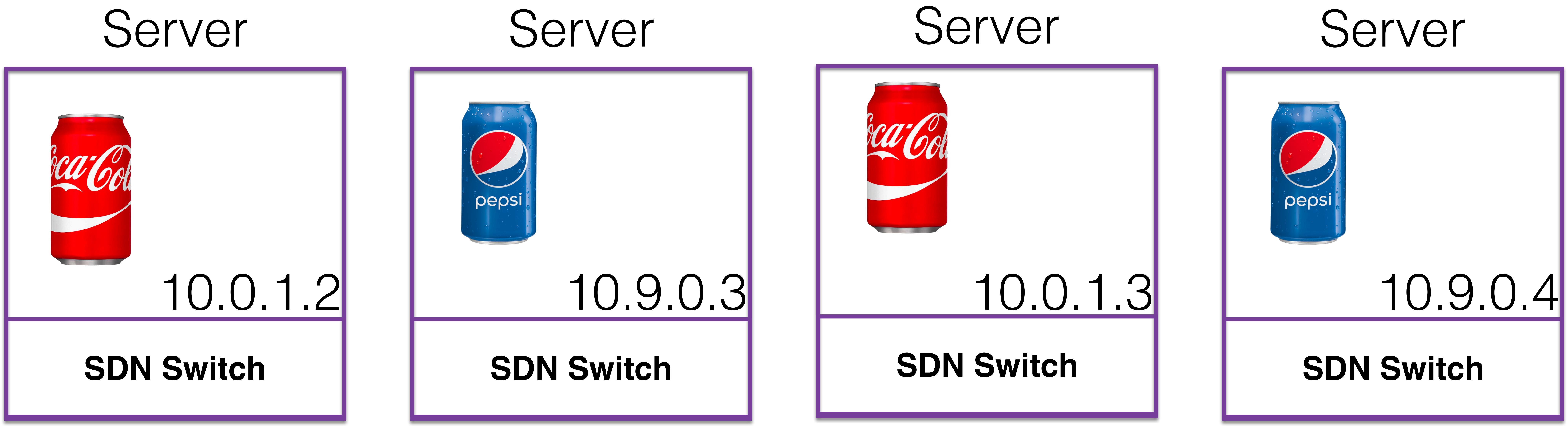
Back to our Datacenter



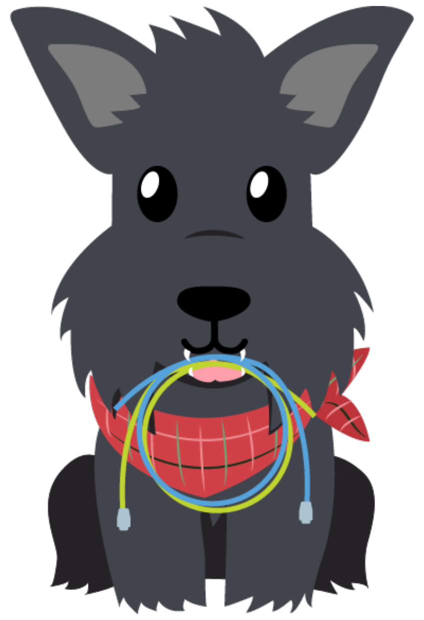


Knowing what you know now, how would you isolate Coke and Pepsi from each other?

SDN Switch at Every Server

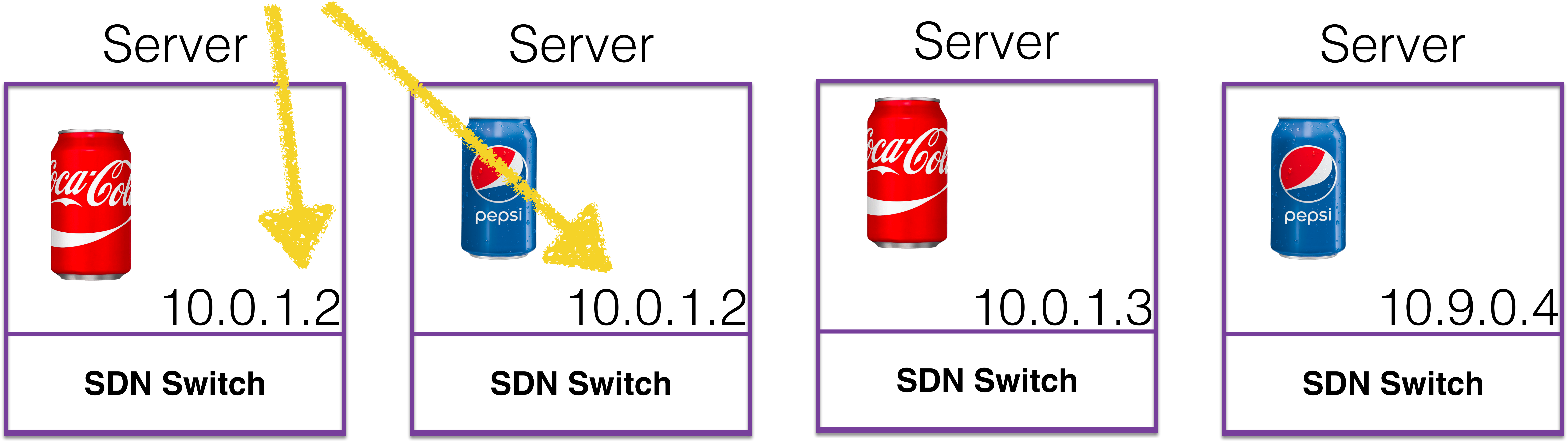


Each server has its own private, virtual address within the Virtual Network for each client.

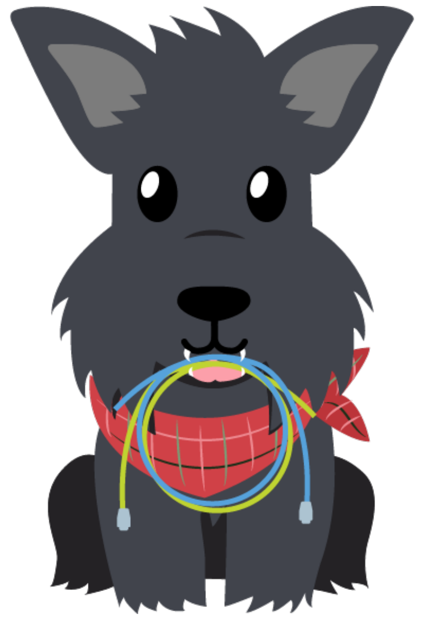


SDN Switch at Every Server

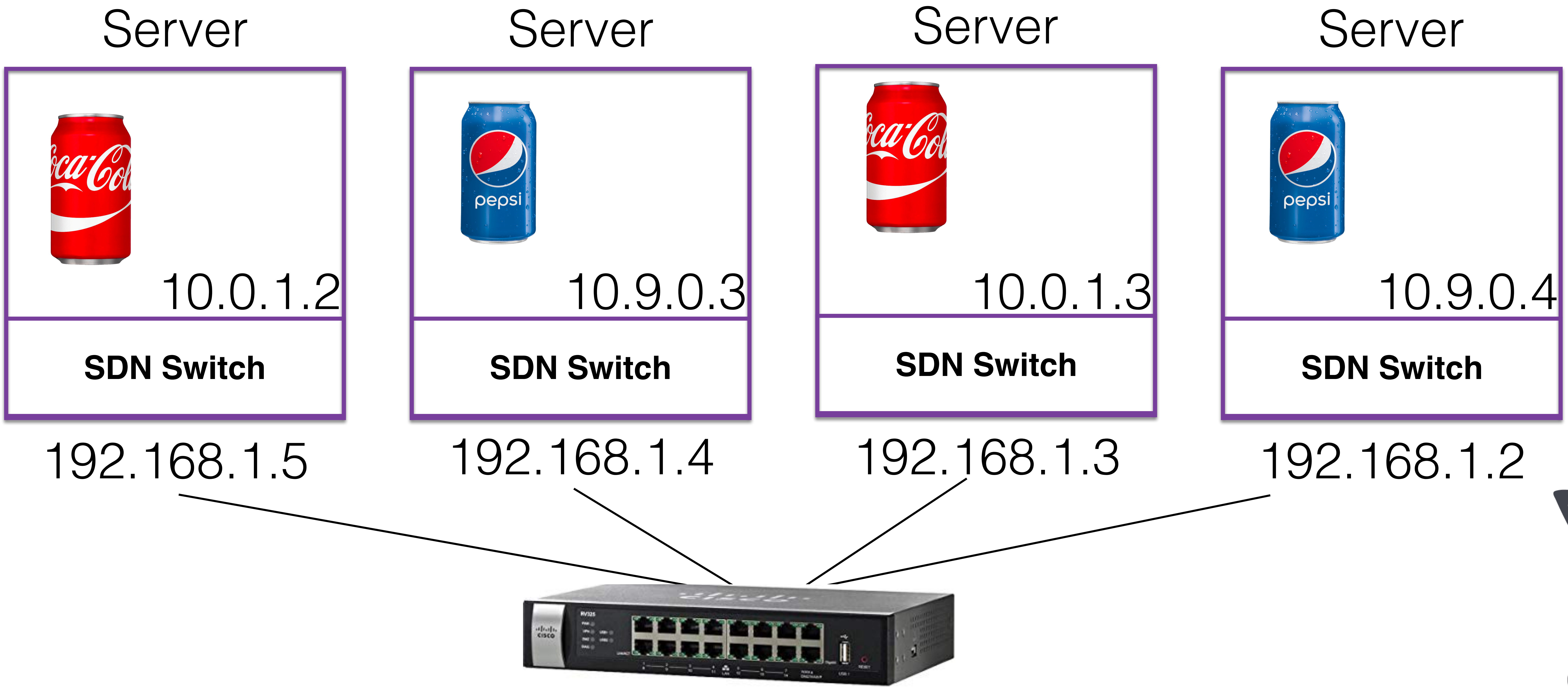
Okay to use the same address — these servers are on virtual networks.

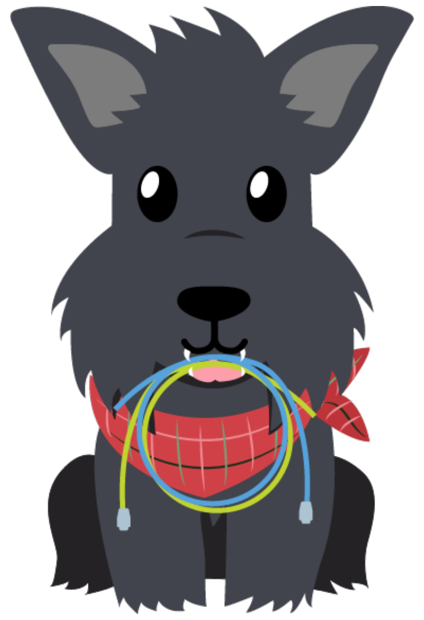


Each server has its own private, virtual address within the Virtual Network for each client.

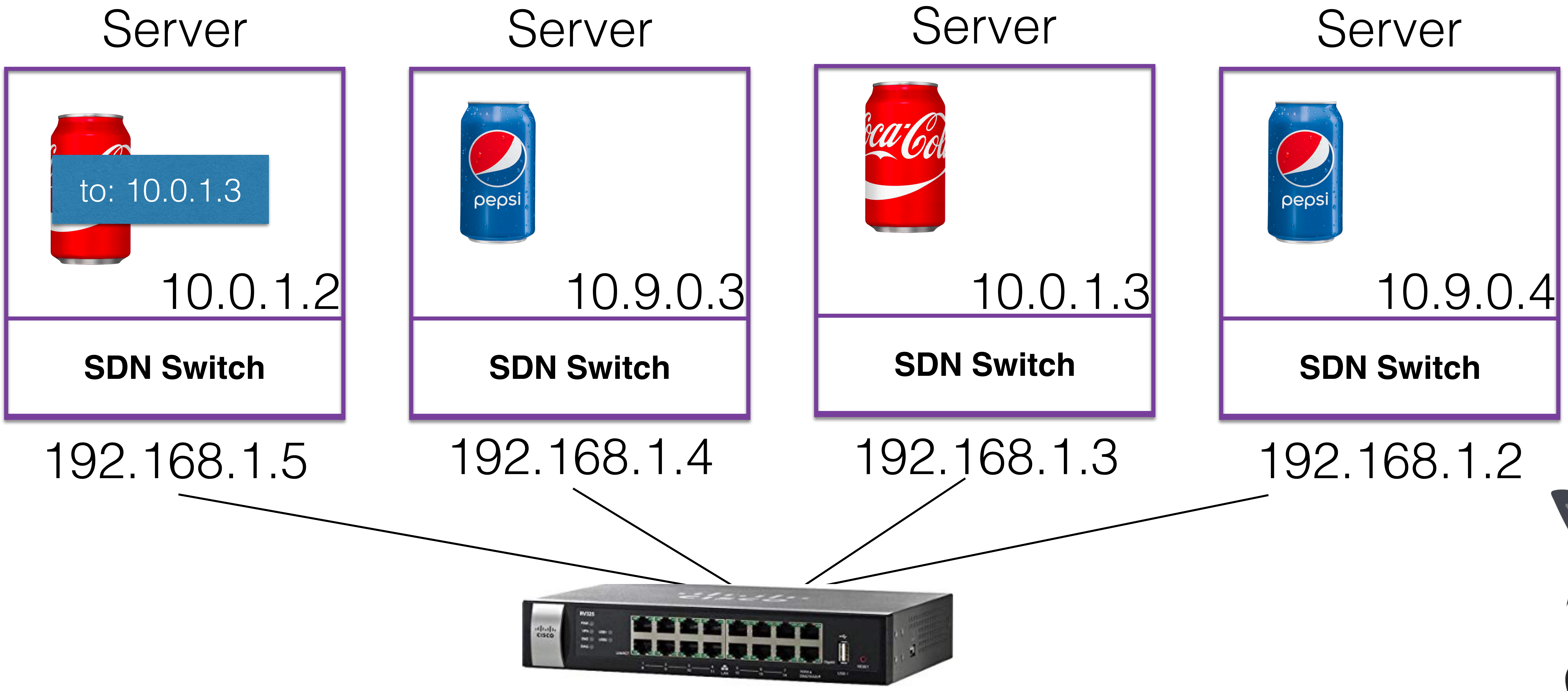


SDN Switch at Every Server

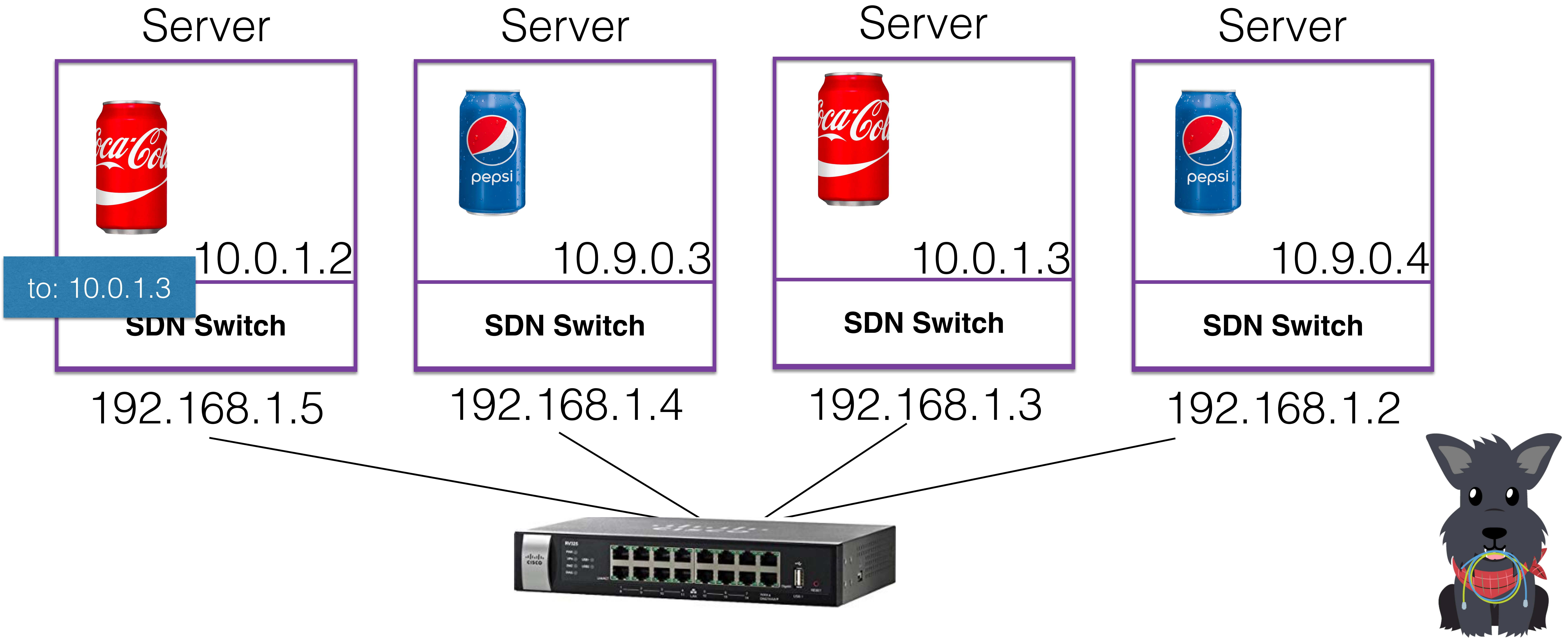




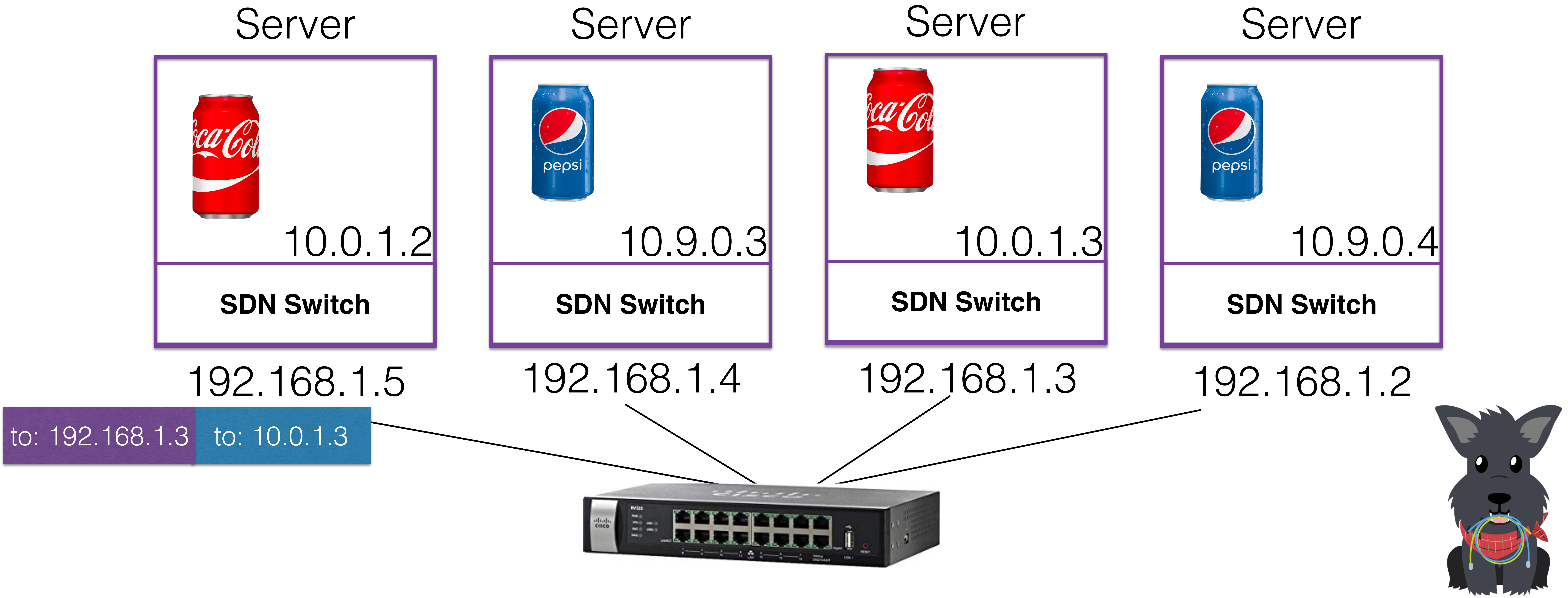
SDN Switch at Every Server



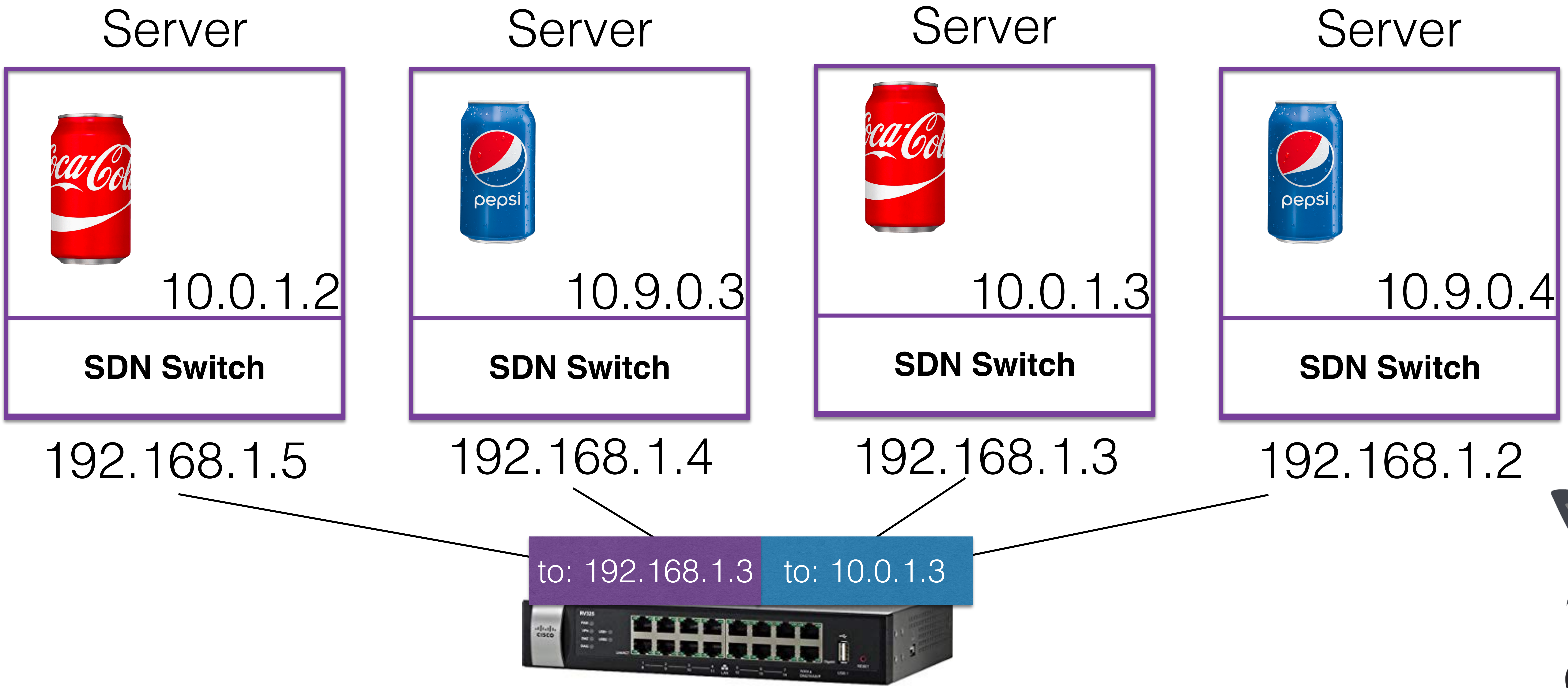
SDN Switch at Every Server

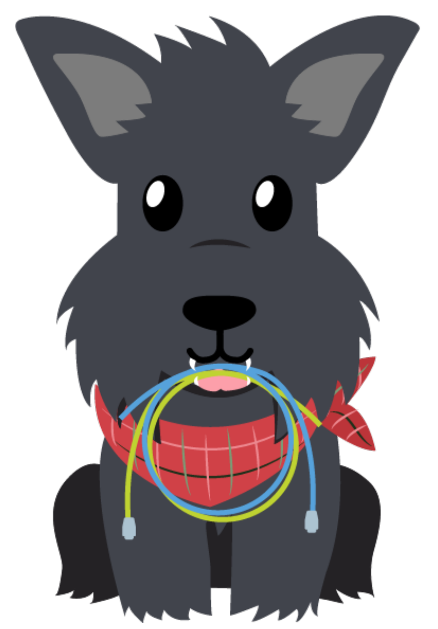


SDN Switch at Every Server

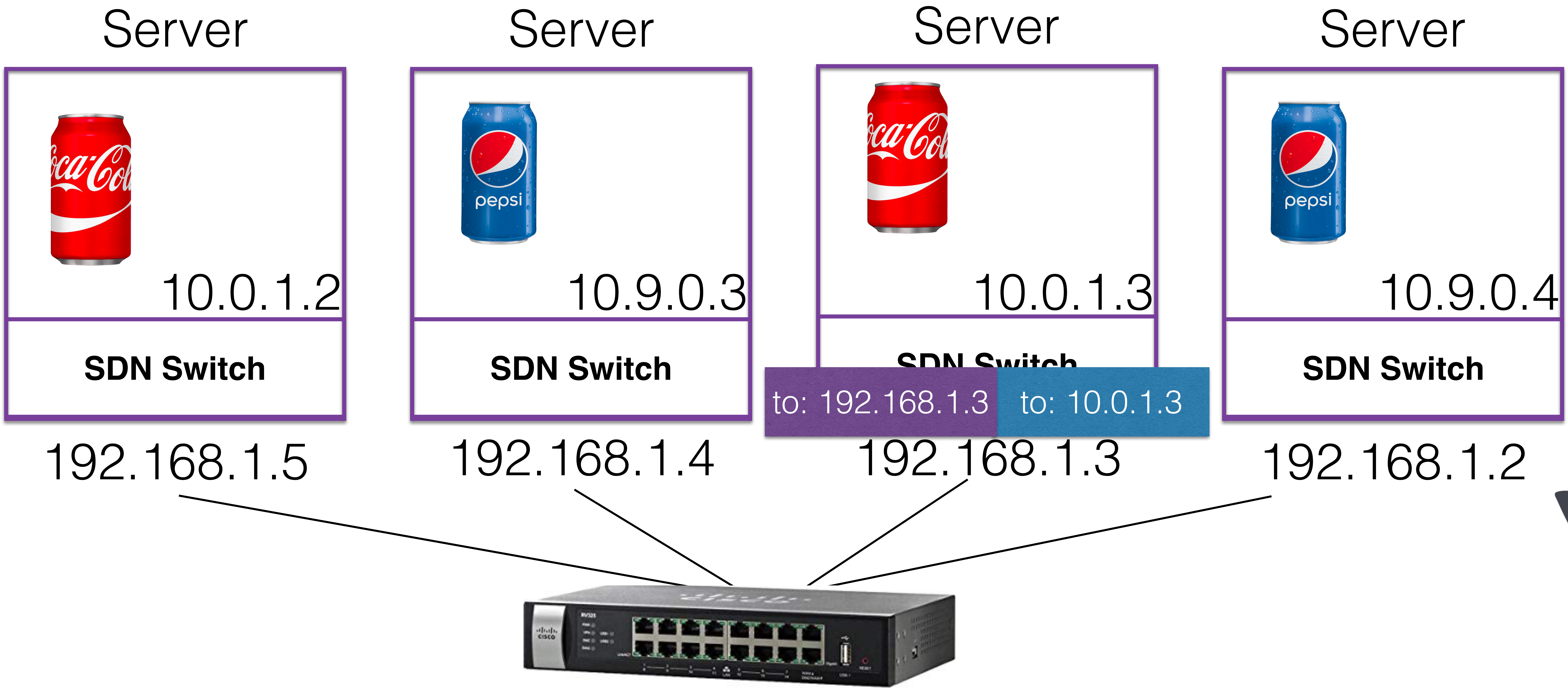


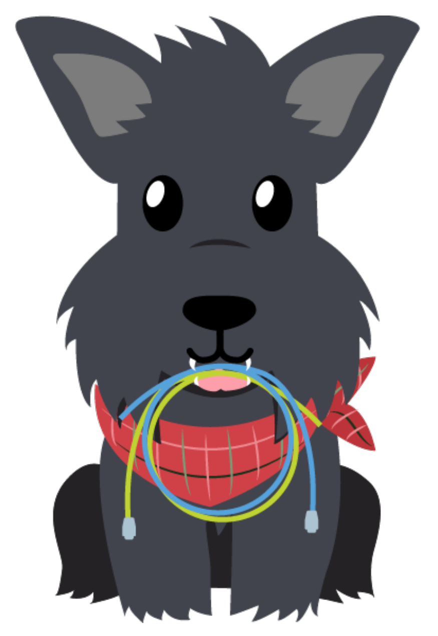
SDN Switch at Every Server



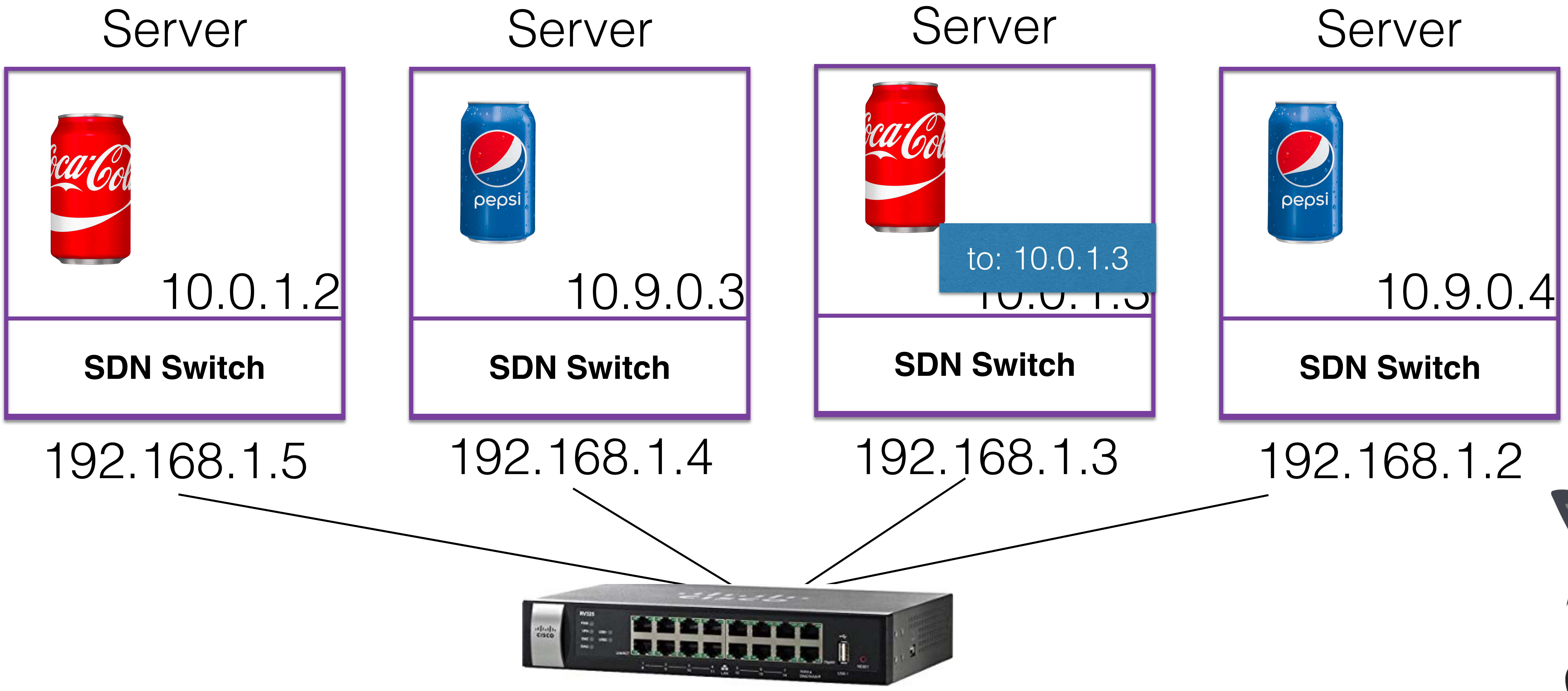


SDN Switch at Every Server

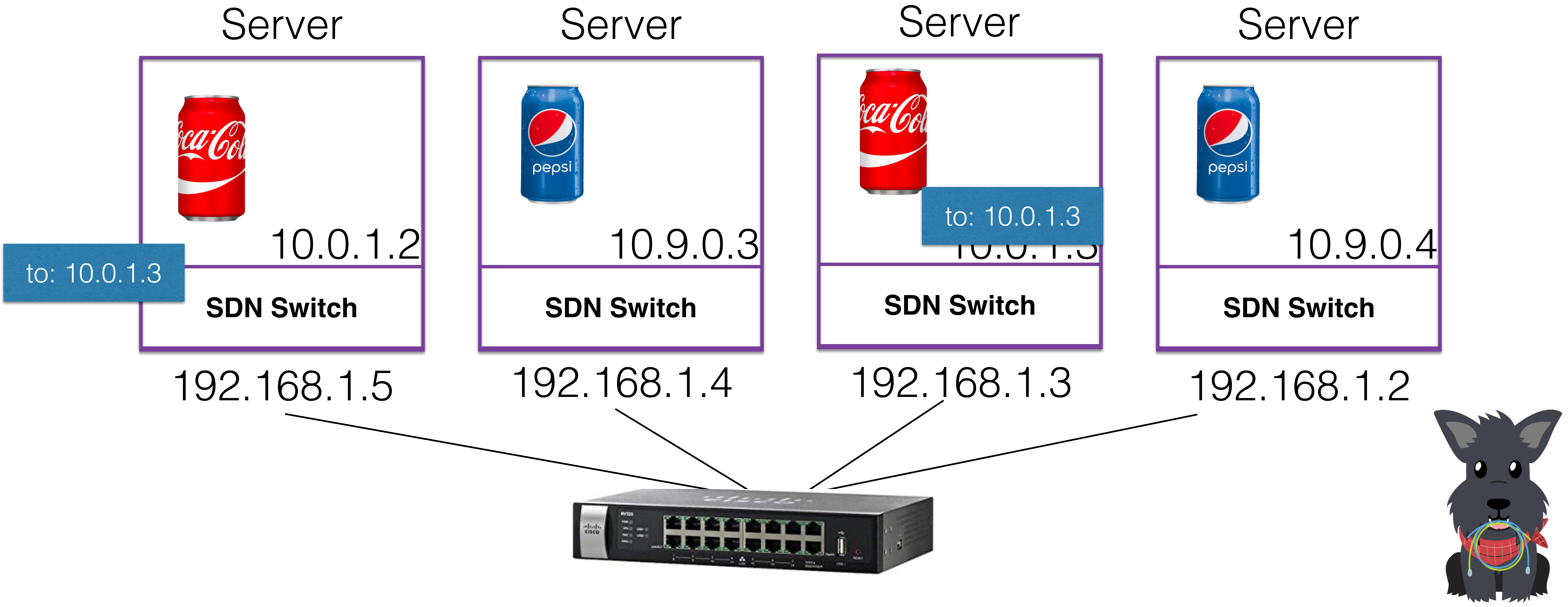




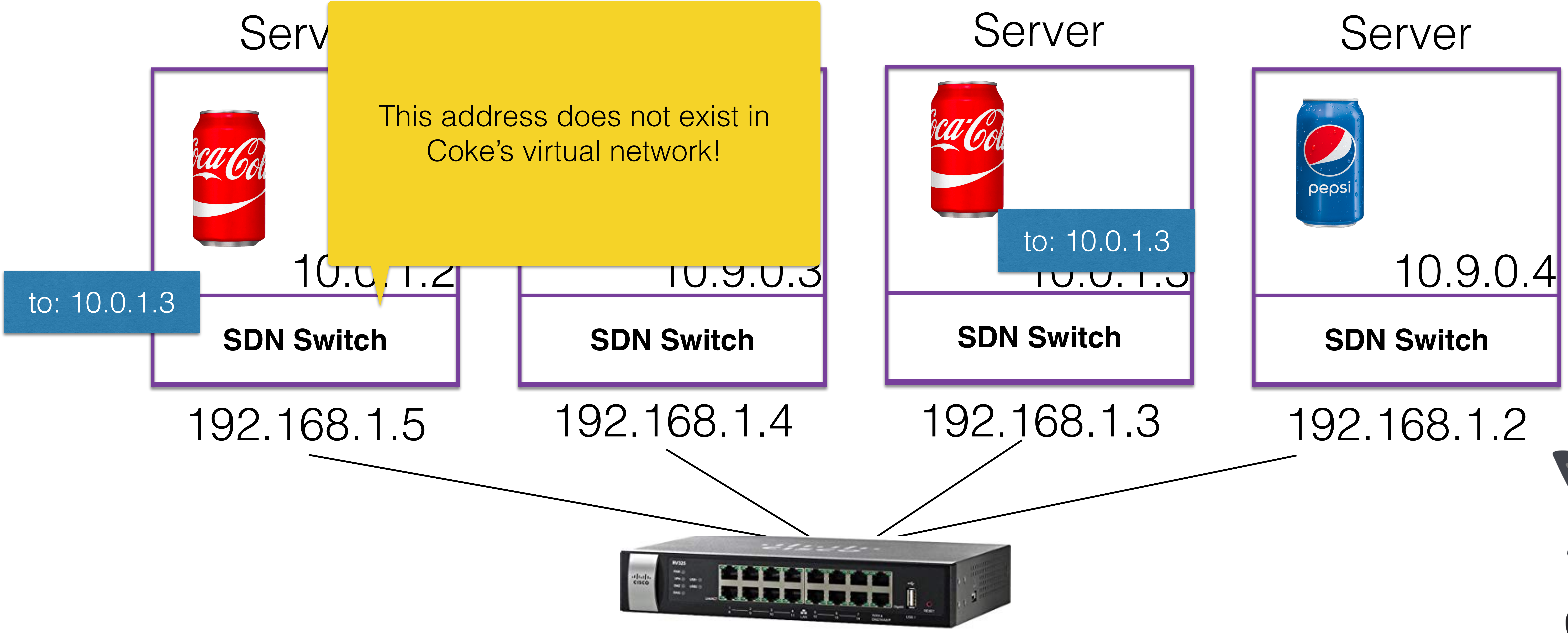
SDN Switch at Every Server



SDN Switch at Every Server



SDN Switch at Every Server

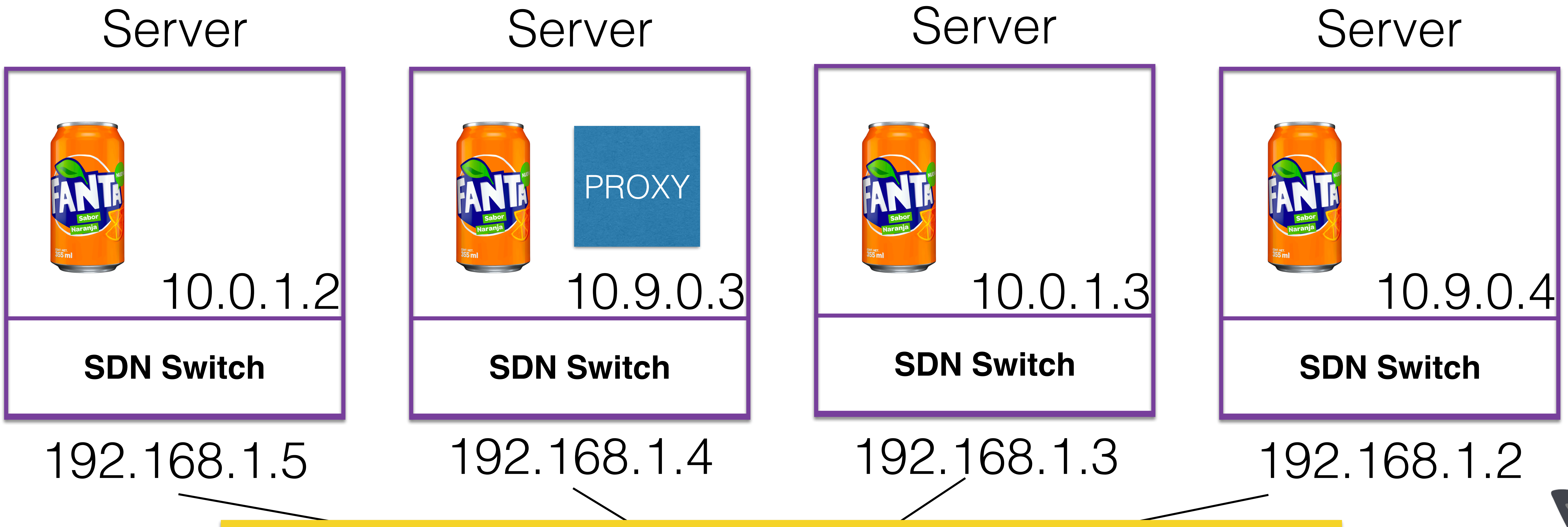


Why implement in software on the host, rather than in real routers/switches like in WANs and LANs?

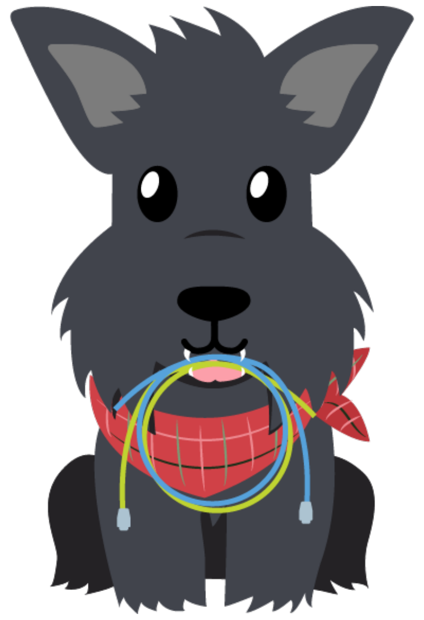
- Easier to update software.
  - Many companies use their own custom protocols/labels to implement their virtual networks.
- There may be multiple clients sharing the same physical server!
  - "On host network"

What about Fanta's Problem?

"I want all traffic between any two nodes to go through my Proxy"



BOOM. Homework question.



Recap: How are datacenter networks different from networks we've seen before?

- **Scale**: very few local networks have so many machines in one place: 10's of thousands of servers — and they are all *working together like one computer!*

- **Control**: entirely administered by one organization — unlike the Internet, datacenter owners control every switch in the network **and** the software on every host

- **Performance:** datacenter latencies are 10s of us, with 10, 40, even 100Gbit links.

These factors change how we design topologies, congestion control, and perform virtualization…

Key Ideas

- Topology: Trees are good!
    - We care about: reliability, available bandwidth, latency, cost, and complexity…
- Congestion Control: Queues are bad!
    - Keeping queue occupancy slow avoids loss and timeouts
- Virtualization: Labels/New Headers are useful!
    - Creating “virtual” networks inside of physical, shared ones provides isolation and can emulate different network topologies without rewiring.