Datacenter Networks

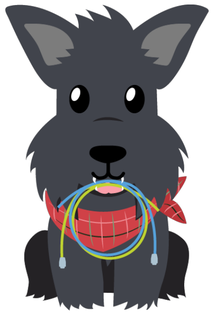
Justine Sherry & Peter Steenkiste

15-441/641

Carnegie Mellon University

Administrivia

- P3 due today at 5PM
  - Unusual deadline to give you time for Carnival :-)
- I officially have funding for summer TAs — please ping me again if you were interested in curriculum development (ie redesigning P3)
- Guest Lecture next week from Jitu Padhye from Microsoft Azure!

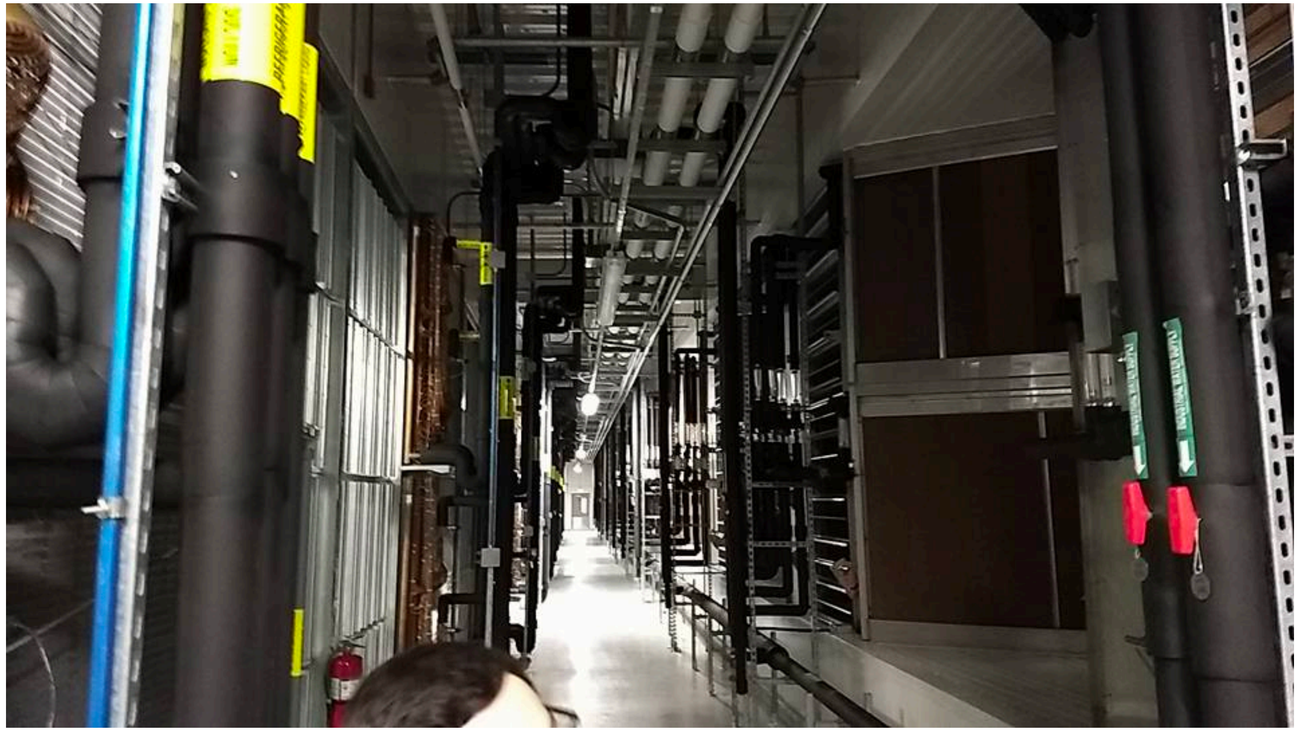


My trip to a Facebook datacenter last year.

(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

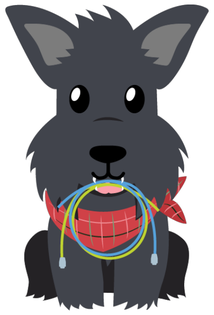


Upstairs: Temperature and Humidity Control

so many fans

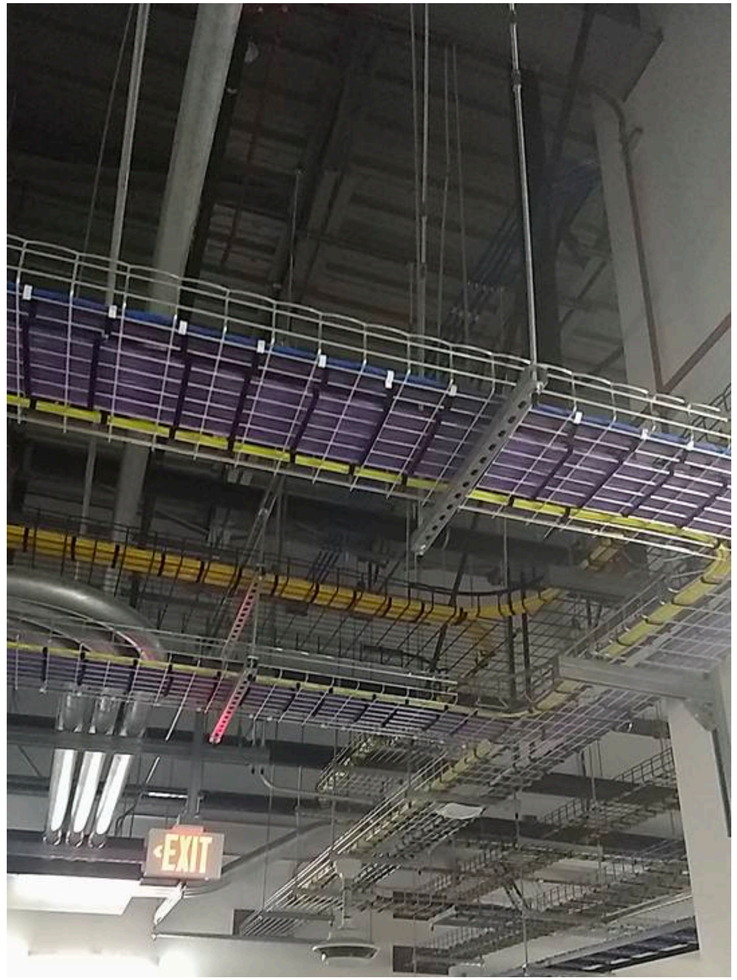
Why so many servers?

- **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?

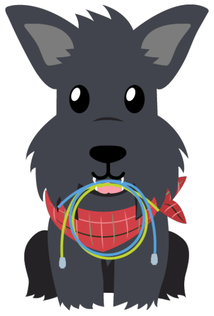
How are datacenter networks different from networks we've seen before?

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

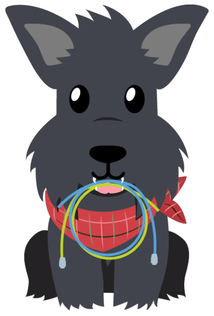
1. Topology

2. Congestion Control

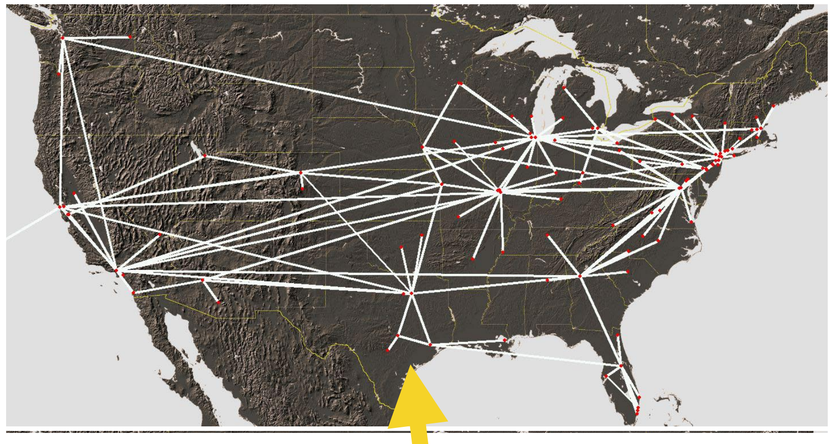
3. Naming and Addressing



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

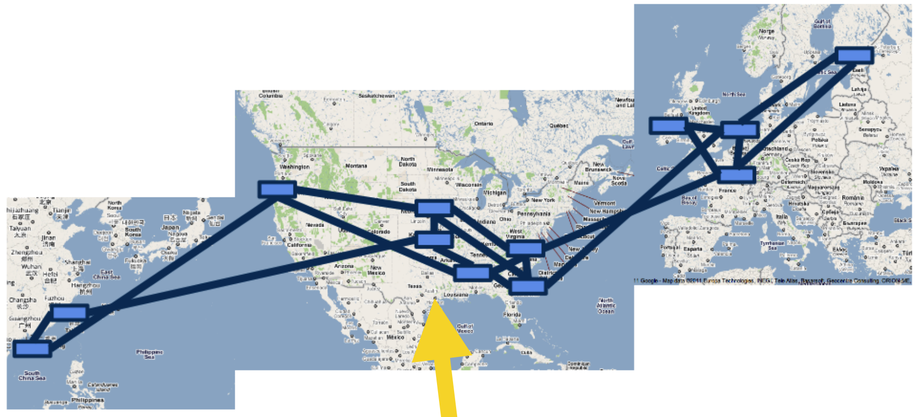


Wide Area Topologies



AT&T's Wide Area Backbone, 2002

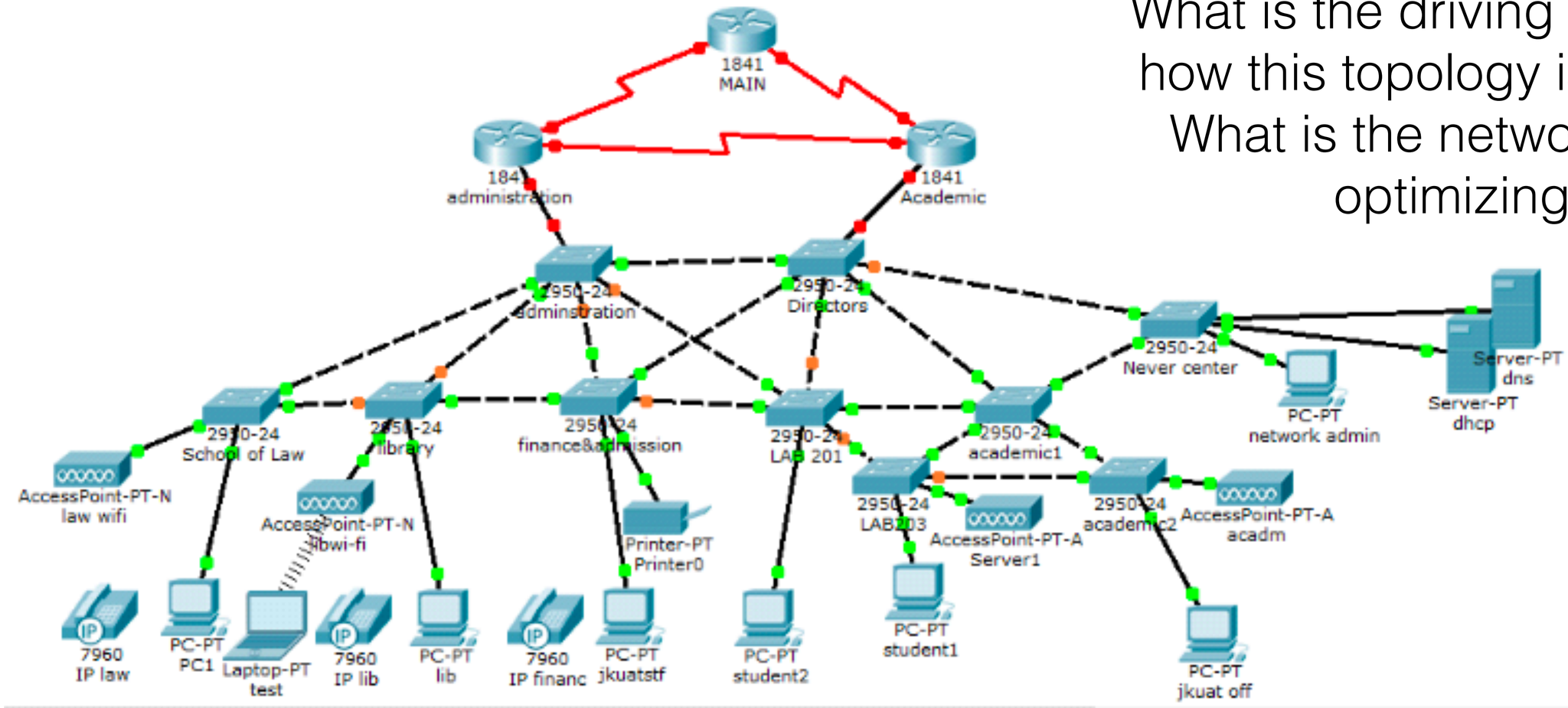
This is called a "hub and spoke"



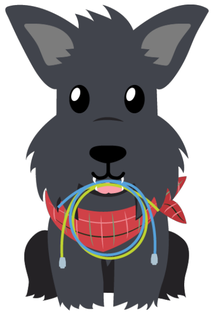
Google's Wide Area Backbone, 2011

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?

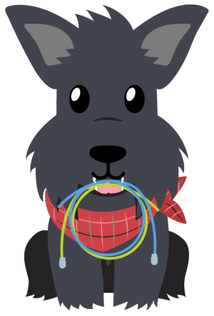


You’re a network engineer…

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

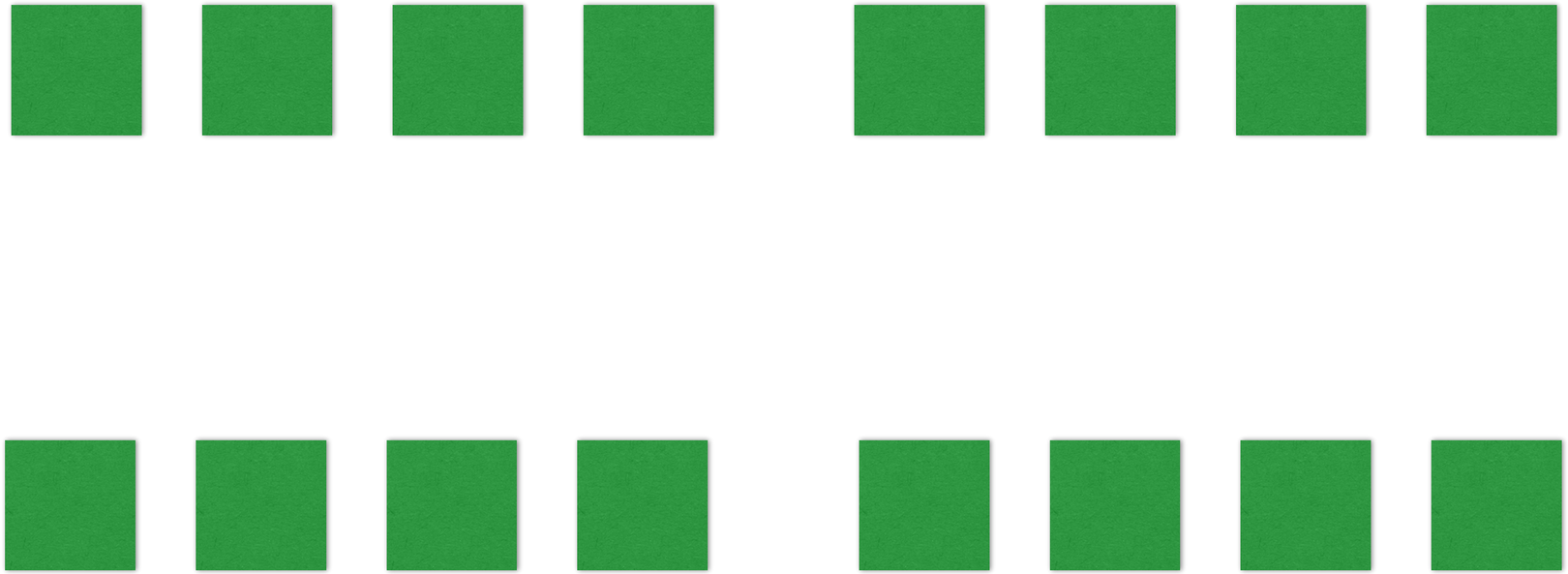
Desirable Properties

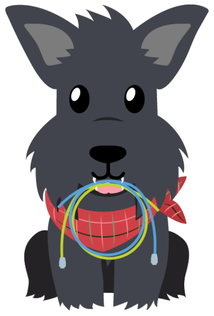
- **Low Latency:** Very few “hops” between destinations
- **Resilience:** Able to recover from link failures
- **Good Throughput:** Lots of endpoints can communicate, all at the same time.
- **Cost-Effective:** Does not rely too much on expensive equipment like very high bandwidth, high port-count switches.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



Activity

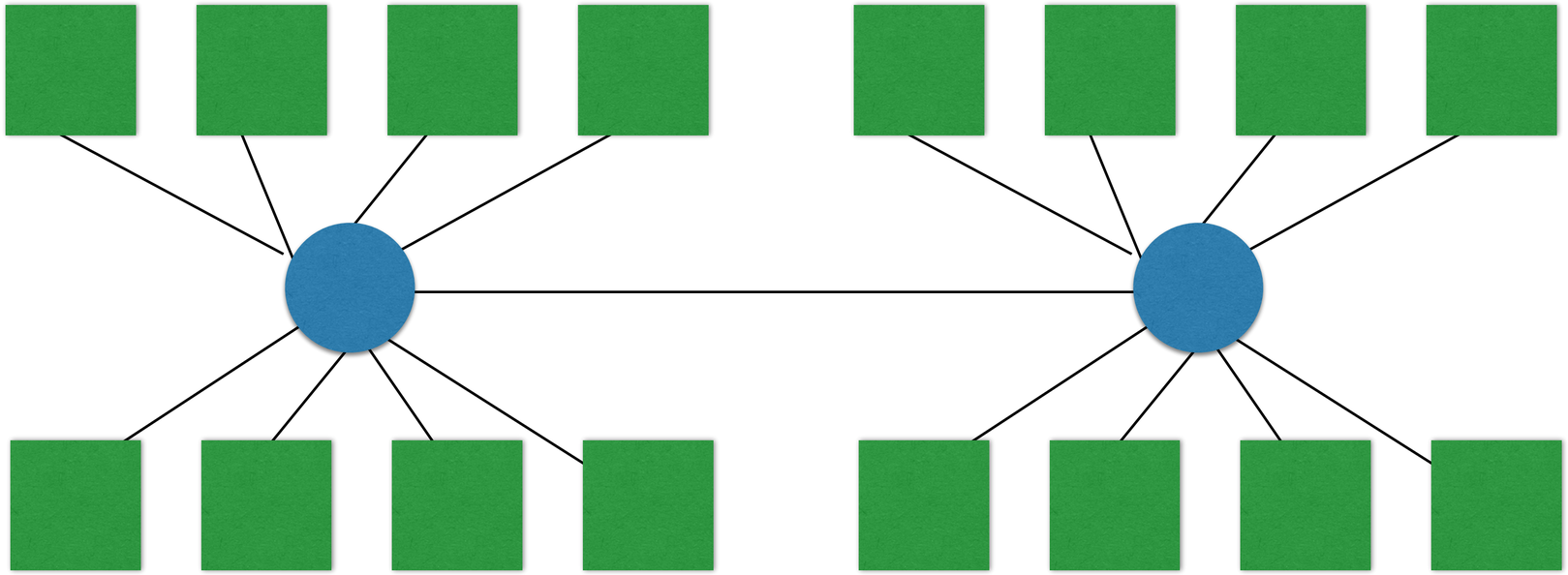
- We have 16 servers. You can buy as many switches and build as many links as you want. How do you design your network topology?

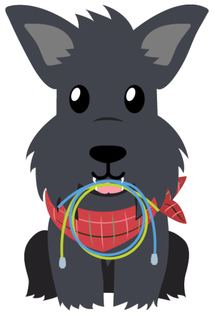




Activity

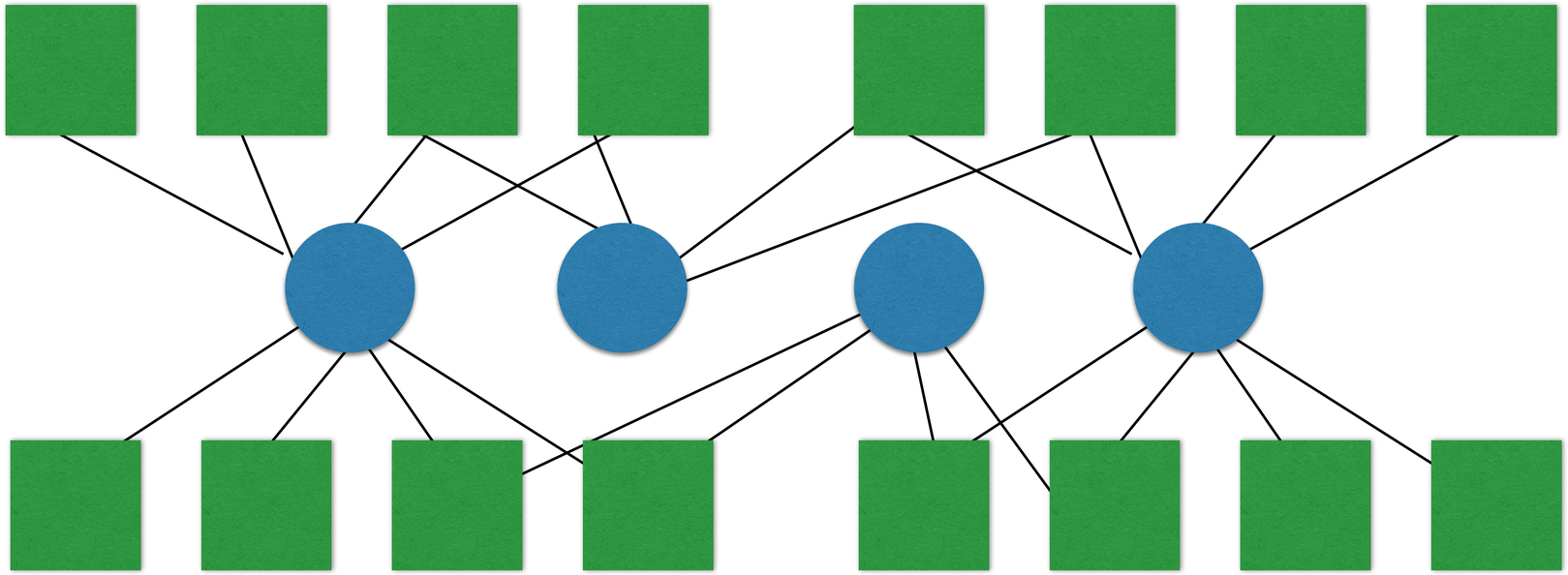
- We have 16 servers. You can buy as many switches and build as many links as you want. How do you design your network topology?

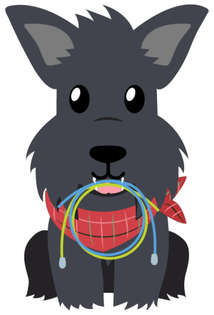




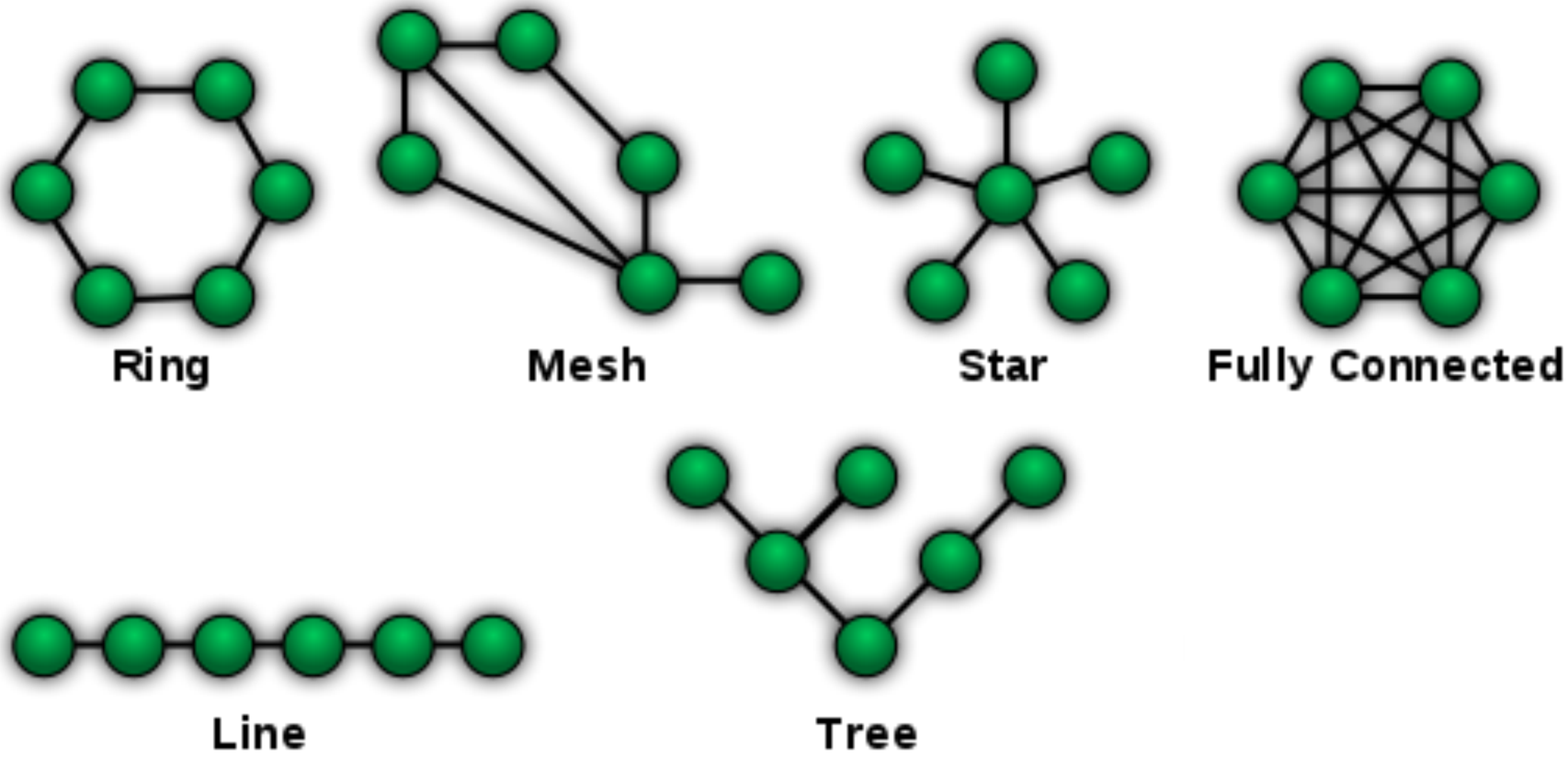
Activity

- 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?

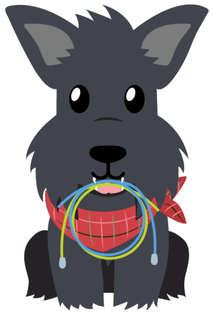




A few “classic” topologies…

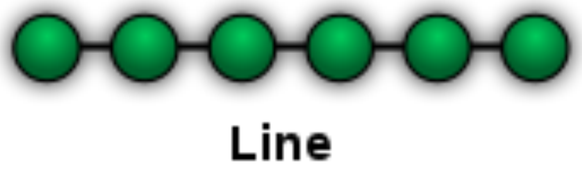


What kind of topology are your designs?

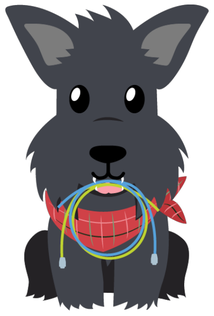


Line Topology

- 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.

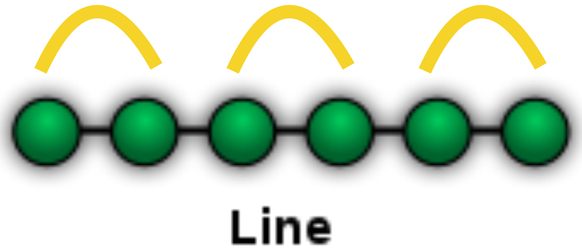


Line

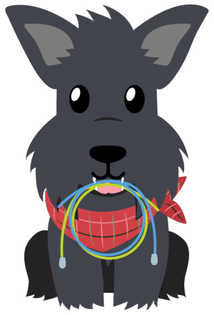


Line Topology

- 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.

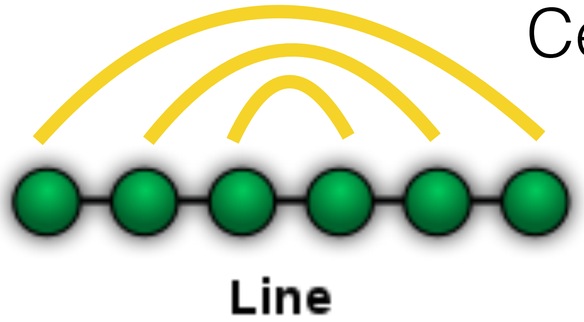


Line

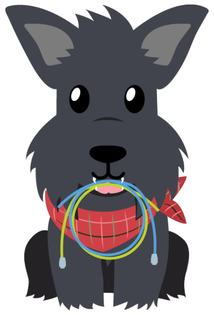


Line Topology

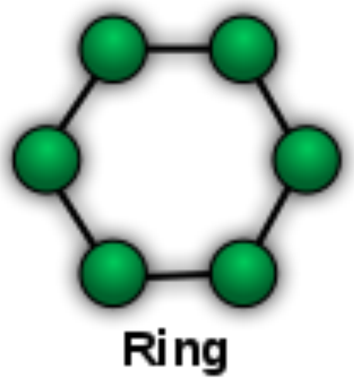
- Simple Design (Easy to Wire)
- Full Reachability
- Bad Fault Tolerance: any failure will partition the network
- High Latency: O(n) hops between nodes
- “Center” Links likely to become bottleneck.



Center link has to support 3x the bandwidth!

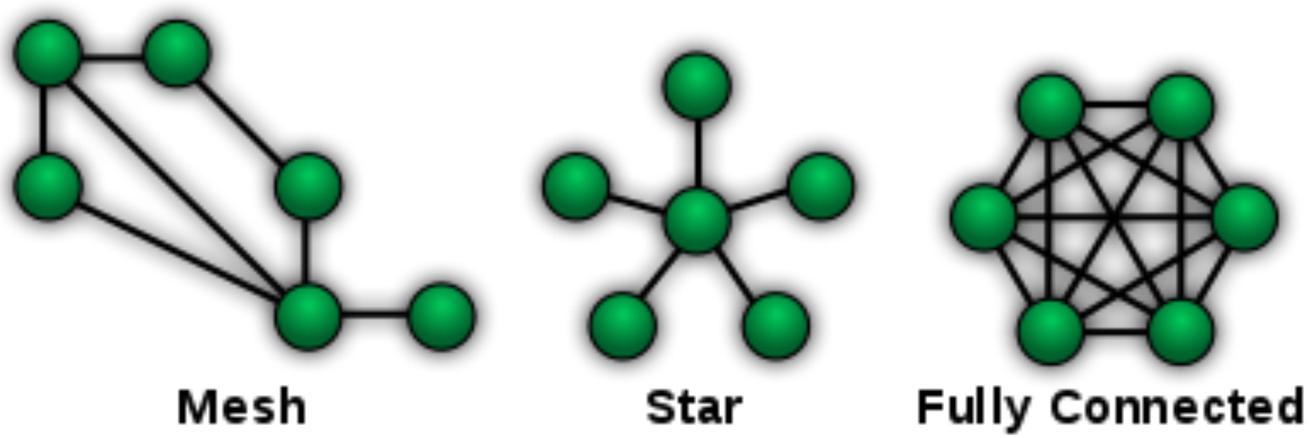


Ring Topology

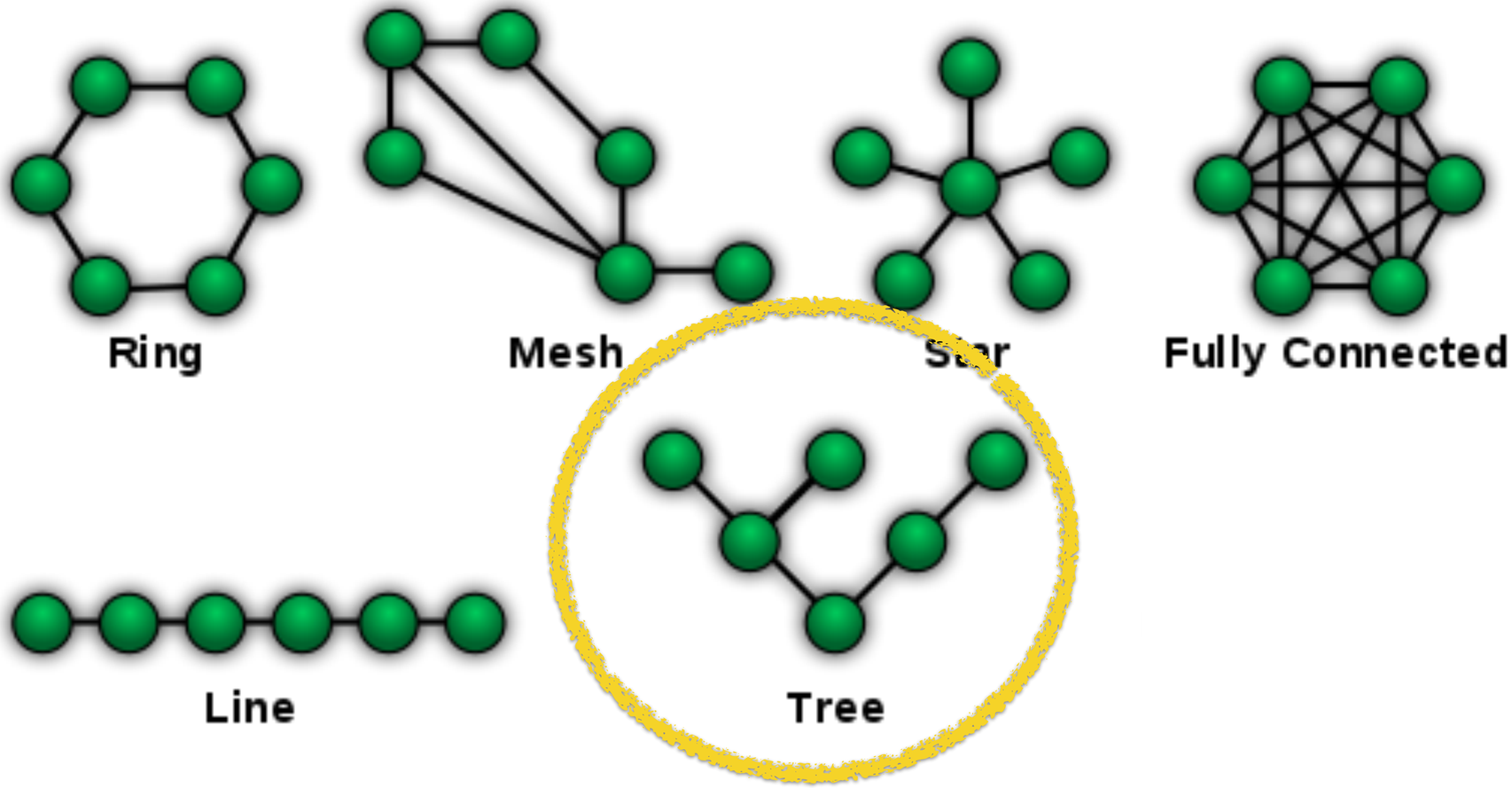


- Simple Design (Easy to Wire)
- Full Reachability
- Better Fault Tolerance (Why?)
- Better, but still not great latency (Why?)
- Multiple paths between nodes can help reduce load on individual links (but still has some bad configurations with lots of paths through one link).

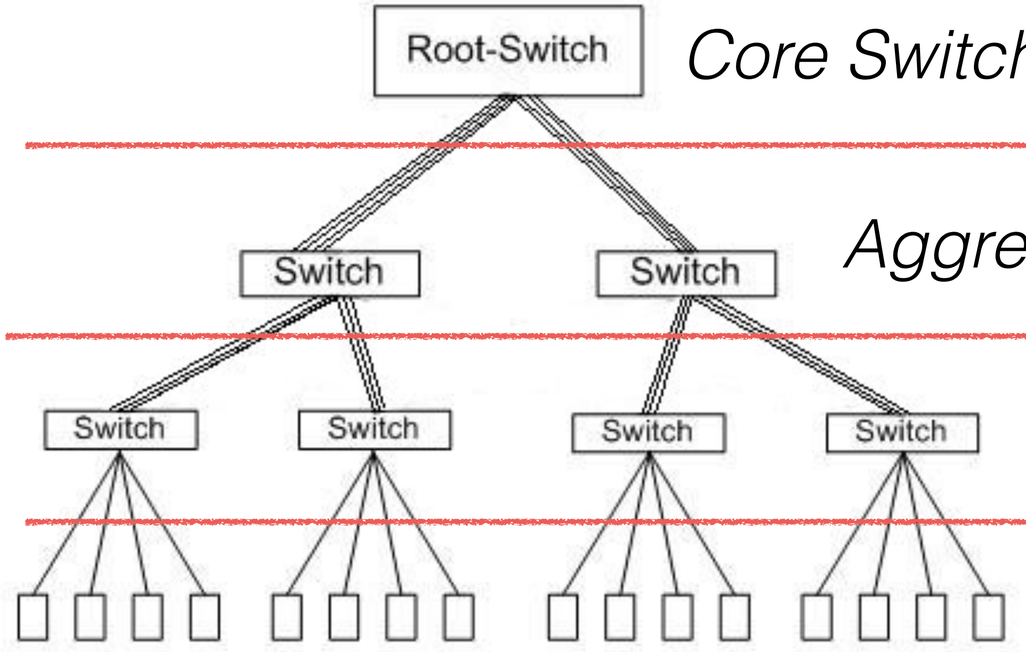
What would you say about these topologies?



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



Classic “Fat Tree” Topology



Core Switch (or Switches)

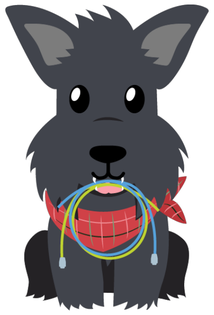
Aggregation Switches

Access (Rack) Switches

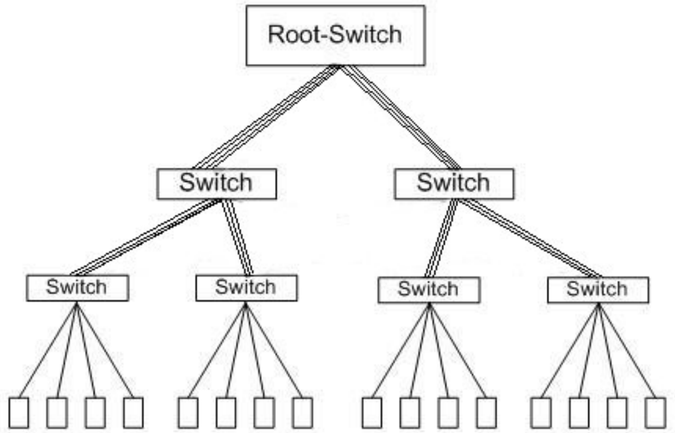
Servers

Higher bandwidth links

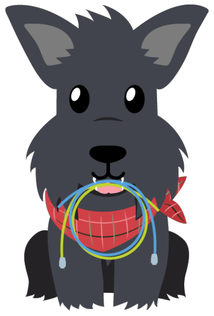
More expensive switches



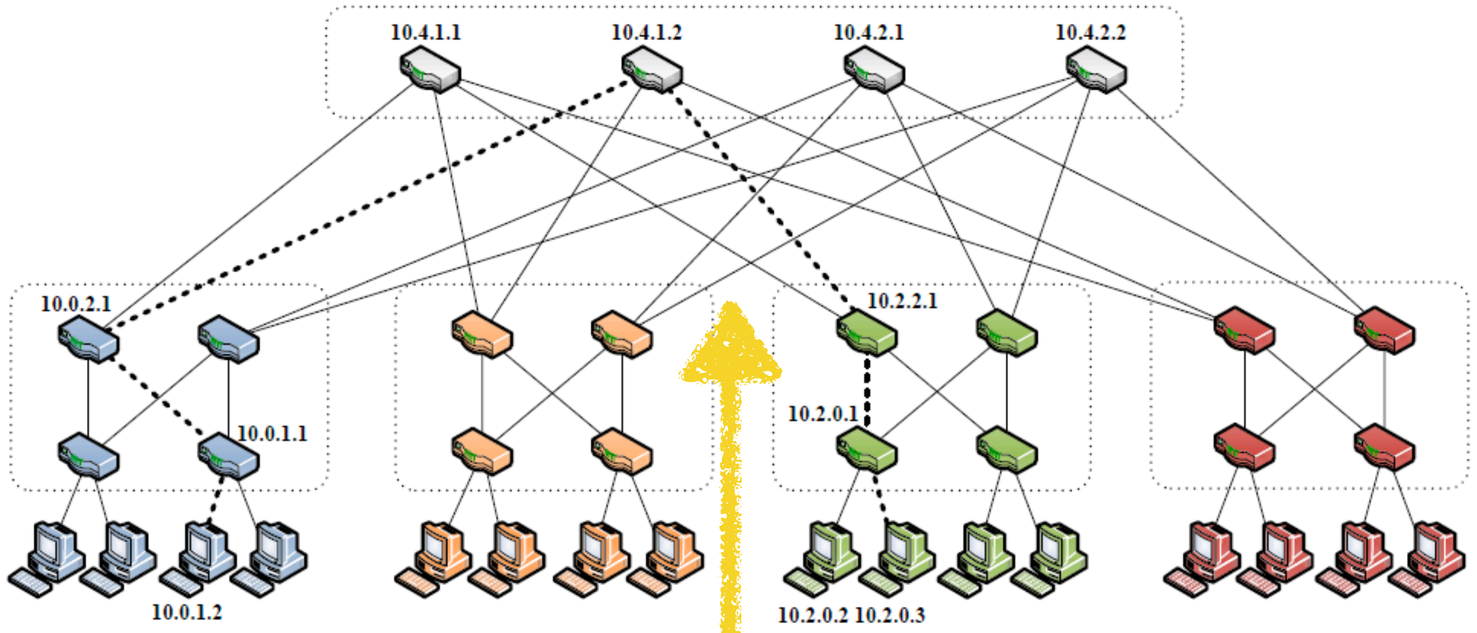
Classic “Fat Tree” Topology



- **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



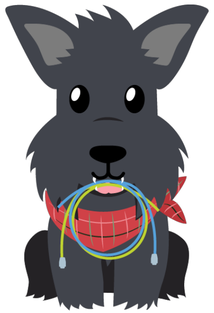
Core

Aggregation

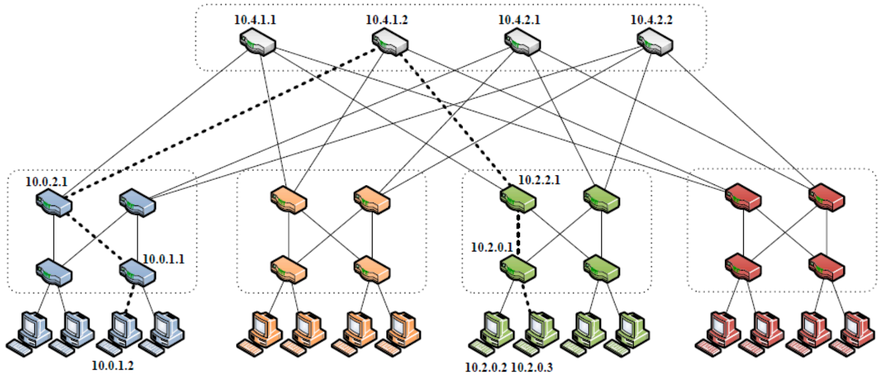
Edge

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

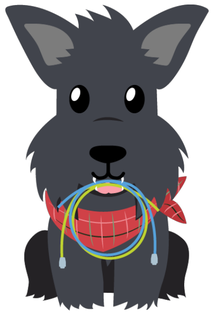
Multiple paths between any pair of servers



Modern Clos-Style Fat Tree



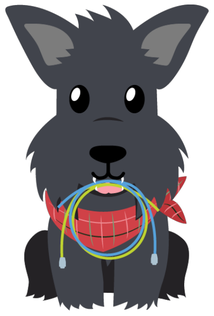
- **Latency:** O(log(n)) hops between arbitrary servers
- **Resilience:** Multiple paths means any individual link failure above access layer won't cause connectivity failure.
- **Throughput:** Lots of endpoints can communicate, all at the same time — due to many cheap paths
- **Cost-Effectiveness:** 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. Naming and Addressing



Datacenter Congestion Control

Google Scholar | datacenter congestion control

Articles | About 23,900 results (0.14 sec)

[PDF] Improving **datacenter** performance and robustness with multipath TCP [PDF] psu.edu
C Raiciu, S Barre, C Pluntke, A Greenhalgh… - ACM SIGCOMM …, 2011 - Citeseer
Improving **Datacenter** Performance and Robustness with … As we will see, the benefits depend on: • The **congestion control** scheme used … Although we cannot predict what future **data center** applications will look like, we can at least map out broad areas where MPTCP gives …
☆ Cited by 652 Related articles All 44 versions Web of Science: 119

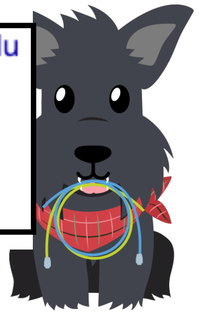
Like regular TCP, we really don't consider this a "solved problem" yet…

Ex
The vast majority of application traffic in modern **data center** networks (DCNs) can be classified into two categories: throughput-sensitive large flows and latency-sensitive small flows. These two types of flows have the conflicting requirements on link buffer occupancy …
☆ Cited by 39 Related articles All 5 versions

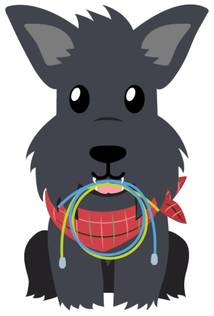
t advances in NIC …
: 36

Data center transport
standardization
M Alizadeh, B Atikoglu, A Kab
Data Center Networks presen
development and deployment.
especially in the **Data Center**
☆ Cited by 110 Relat

CONGA: Distributed **congestion**-aware load balancing for **datacenters** [PDF] mit.edu
M Alizadeh, T Edsall, S Dharmapurikar… - ACM SIGCOMM …, 2014 - dl.acm.org
… Hence, to reduce state, the destination leaf aggregates **congestion** metrics for one or more paths … fine-grained load balancing across Internet paths [27], but how does it perform in **datacenters**? On the one hand, the very high bandwidth of internal **datacenter** flows would seem to …
☆ Cited by 303 Related articles All 17 versions Web of Science: 65



How many of you chose the datacenter as your Project 2 Scenario? How did you change your TCP?



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

Short messages

(e.g., query, coordination)

Low Latency

Large flows

(e.g., data update, backup)

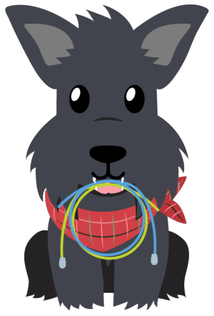
High Throughput

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?

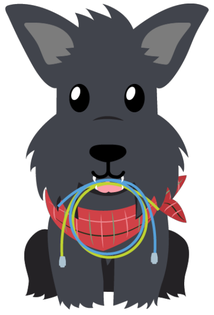
Small, latency-sensitive connections

Long-lived, large transfers

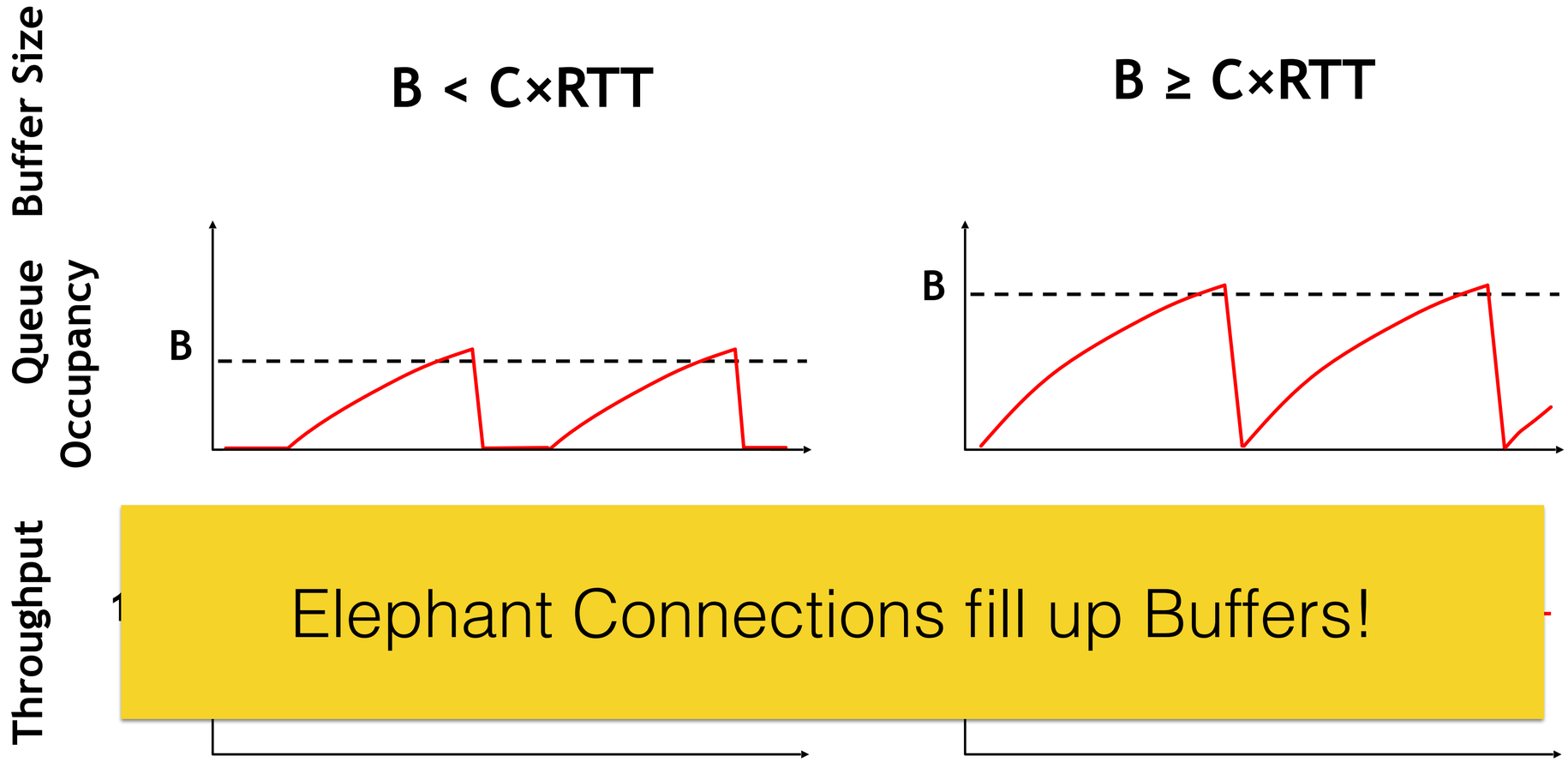


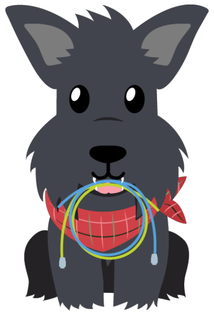
In the Datacenter

- 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!



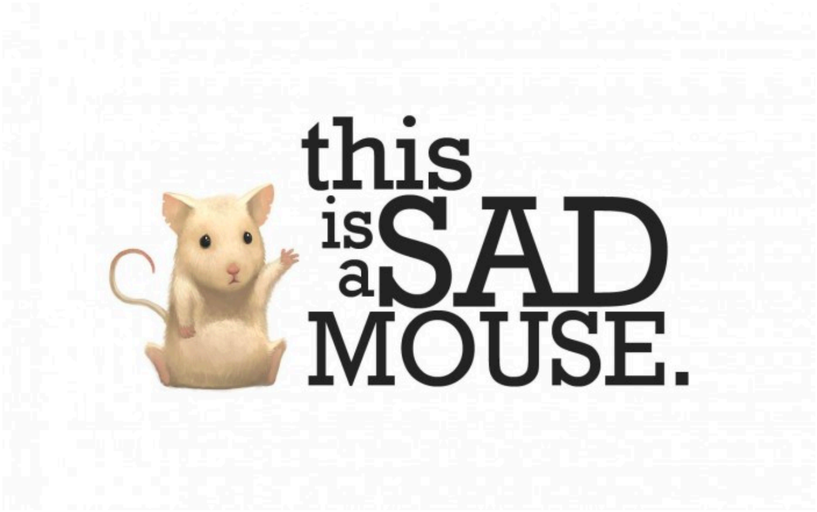
TCP Fills Buffers — and needs them to be big to guarantee high throughput.

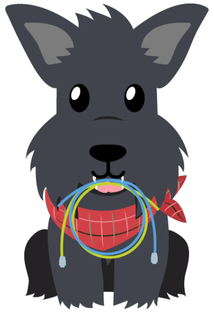




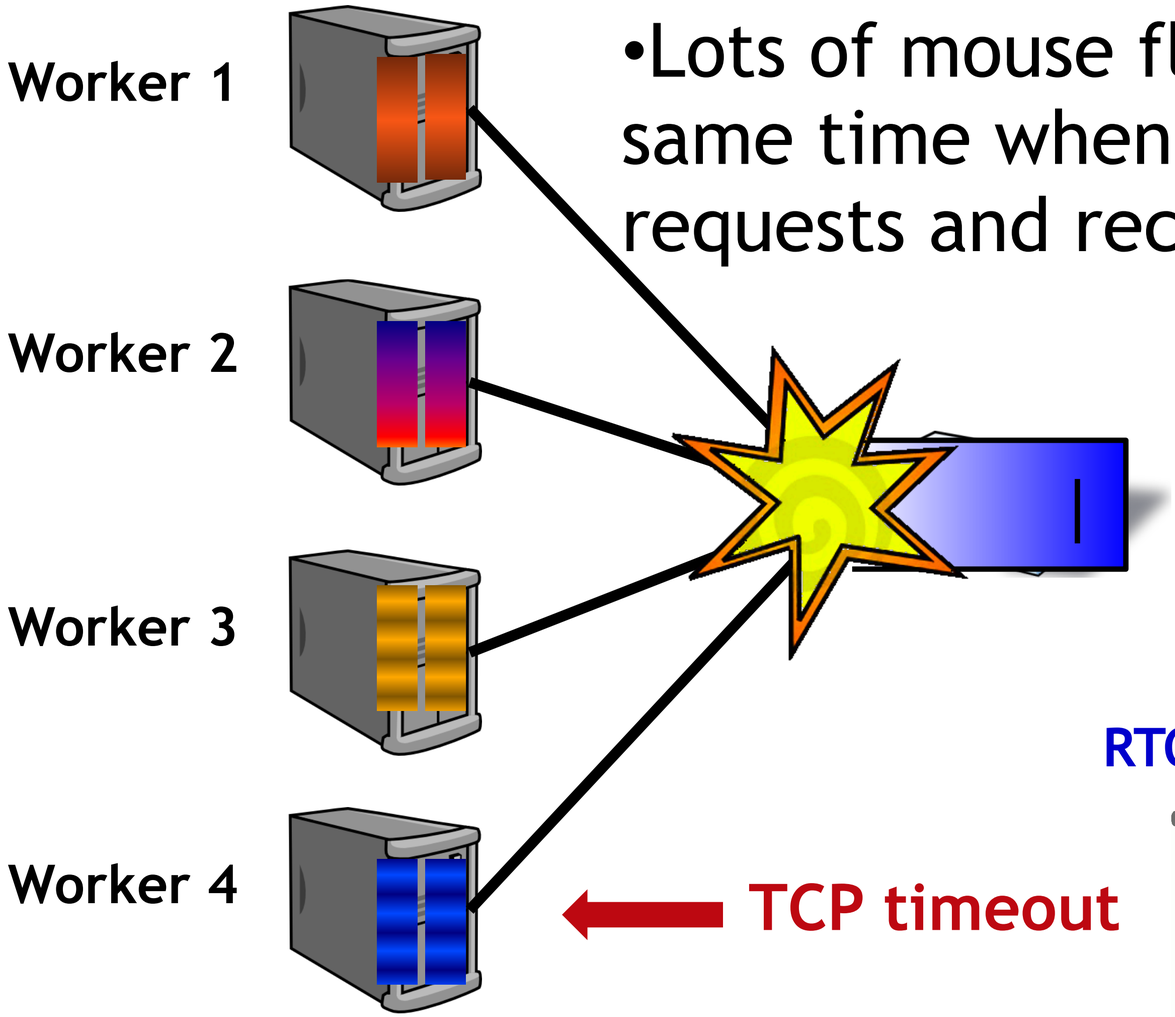
Full Buffers are Bad for Mice

- 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!)



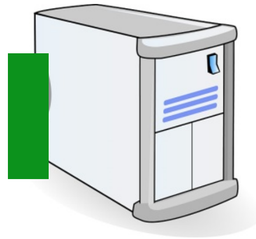


Incast: Really Sad Mice!



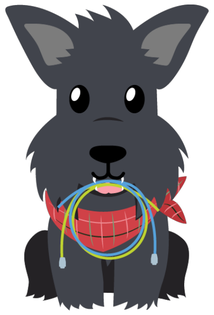
- Lots of mouse flows can happen at the same time when one node sends many requests and receives many replies at once!

Aggregator

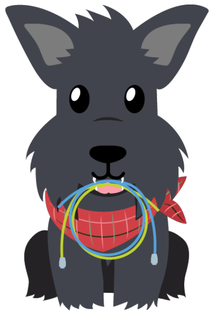


RTO_{min} = 300 ms

TCP timeout

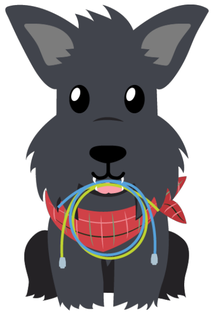


When the queue is already full, even more packets are lost and timeout!



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

Ideas?



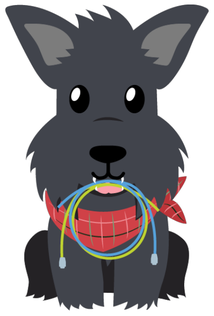
A few approaches

- **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?

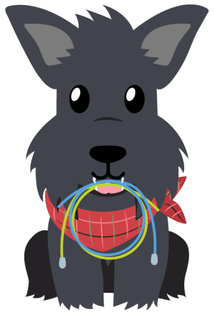
I can’t wait to test your TCP implementations next week!



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. Naming and Addressing **THURSDAY**



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 naming and addressing…