Computer Networks and Internet Technology

2021W703033 VO Rechnernetze und Internettechnik
Winter Semester 2021/22

Jan Beutel

universität innsbruck

With materials and inspiration from Scott Shenker, Jennifer Rexford, Phil Levis, Adrian Perrig and Laurent Vanbever

Communication Networks and Internet Technology

Part 1: General overview

#1 What is a network made of?

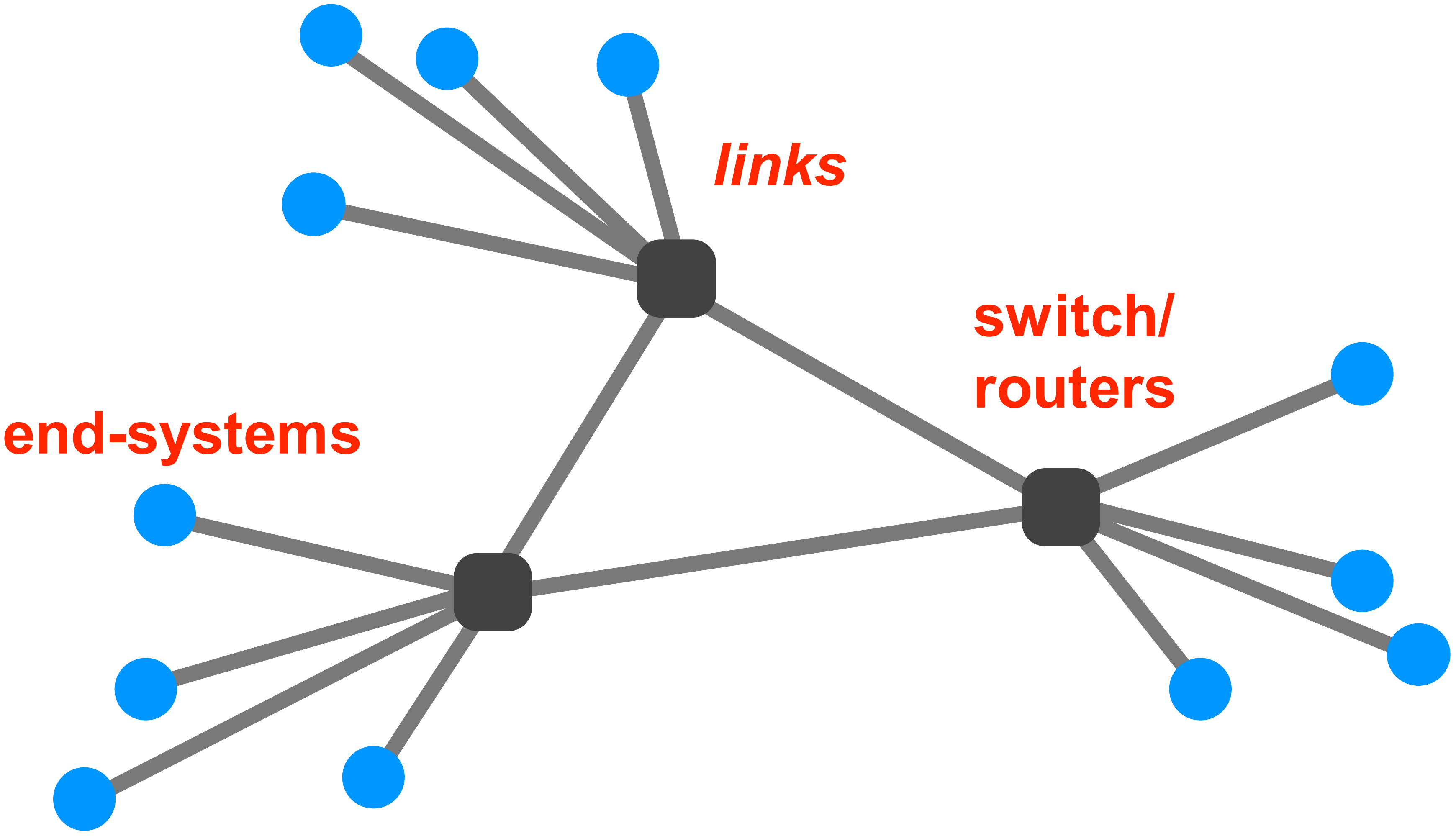
How is it shared?

How is it organized?

How does communication happen?

How do we characterize it?

Networks are composed of three basic components



Communication Networks and Internet Technology

Part 1: General overview

What is a network made of?

#2 How is it shared?

How is it organized?

How does communication happen?

How do we characterize it?

So far, we’ve been discussing what the "last mile" of the Internet looks like

What about the rest of the network?

3 must-have requirements of a good network topology

Tolerate failures

several paths between each source and destination

Possess enough sharing to be feasible & cost-effective

number of links should not be too high

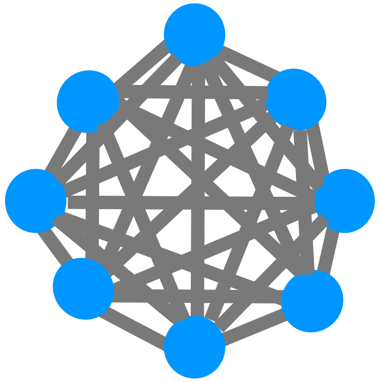
Provide adequate per-node capacity

number of links should not be too small

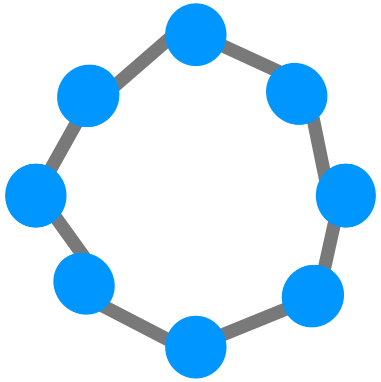
Compare these three designs in terms of
sharing, resiliency, and per-node capacity

design

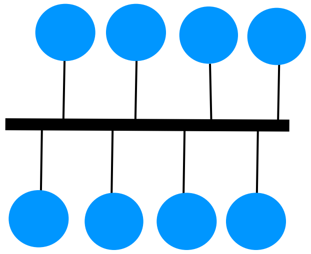
full-mesh



chain



bus

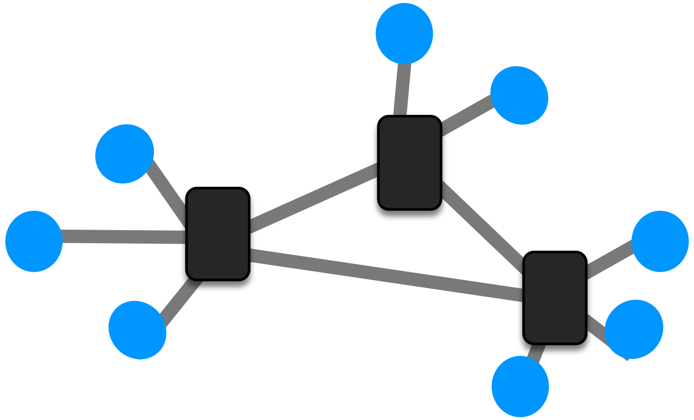


advantages

disadvantages

Switched networks provide
reasonable and flexible compromise

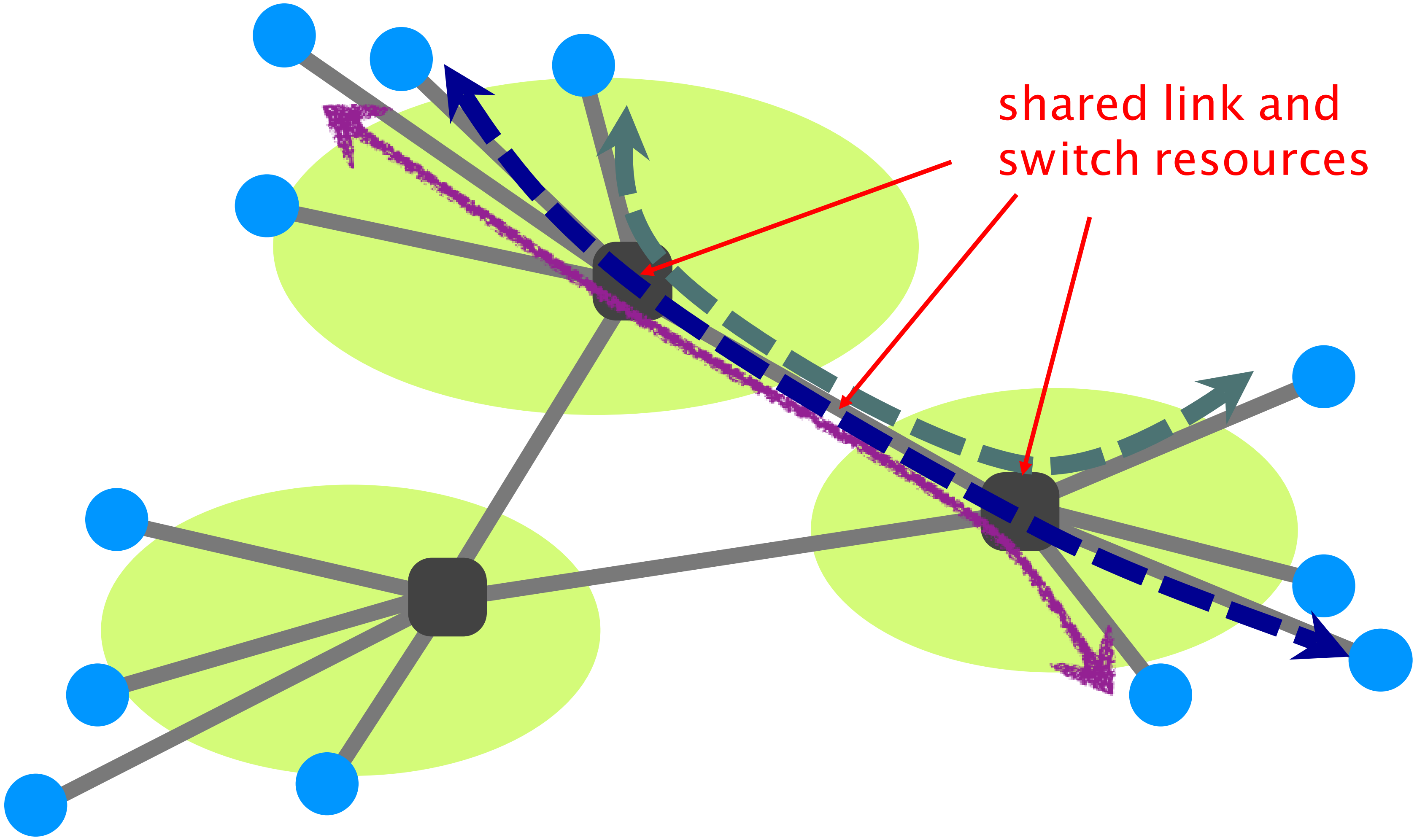
design switched



advantages sharing and per-node capacity can be adapted to fit the network needs

disadvantages require smart devices to perform: forwarding, routing, resource allocation

Links and switches are shared between flows



Communication Networks and Internet Technology

This weeks lecture

There exist two approaches to sharing:
reservation and on-demand

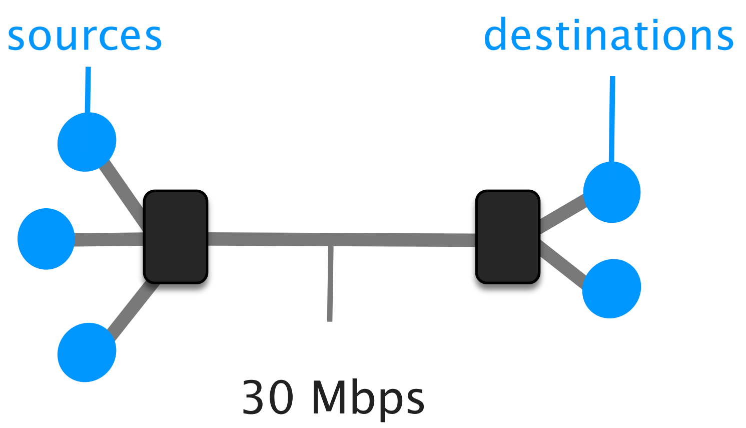
	Reservation	On-demand
principle	reserve the bandwidth you need in advance	send data when you need

Both are examples of statistical multiplexing

	Reservation	On-demand
multiplexing	at the flow-level	at the packet-level

Between reservation and on-demand:

Which one do you pick?

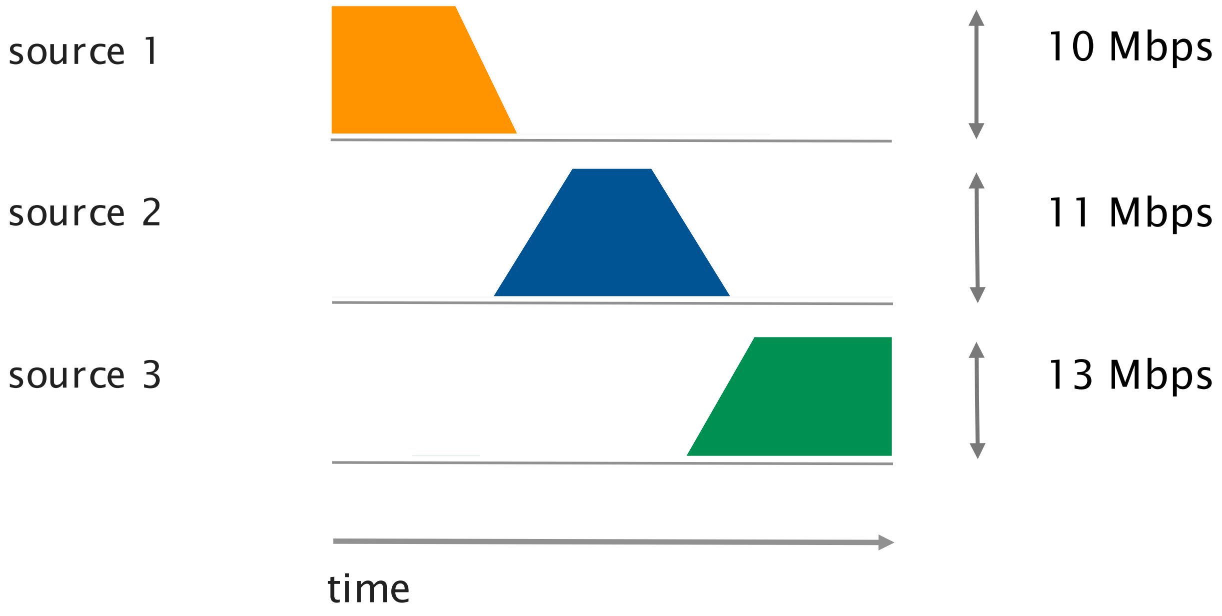


Consider that each source needs 10 Mbps

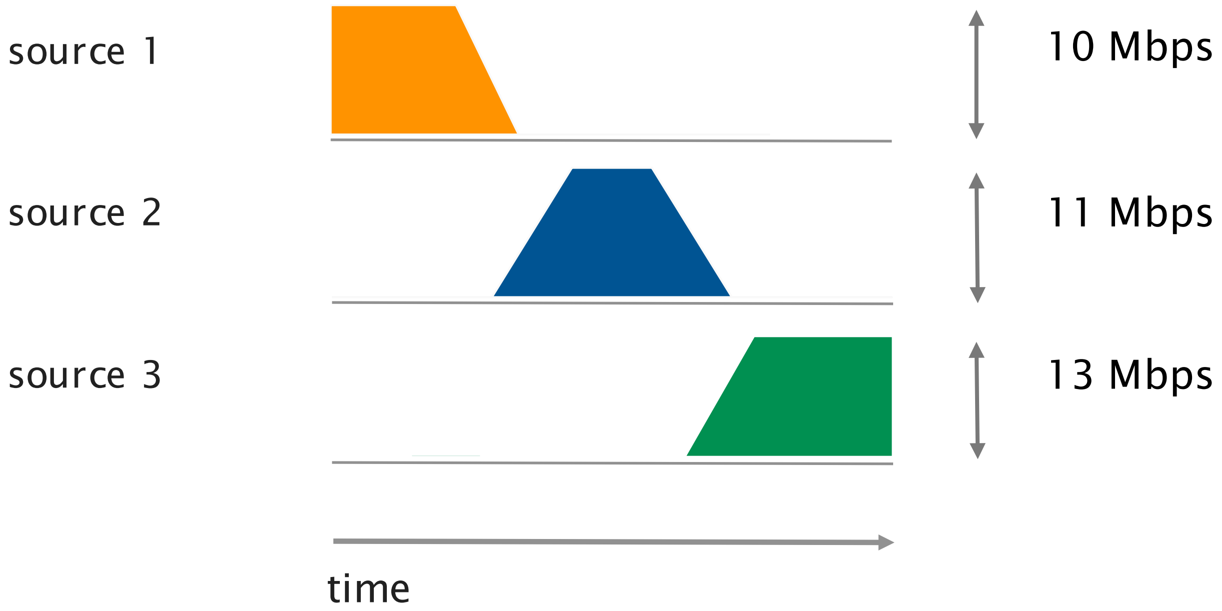
What do they get with:

- reservation
- on-demand

Assume the following peak demand and flow duration



Assume the following peak demand and flow duration



What does each source get with reservation and on-demand?

first-come first-served

equal (10 Mbps)

Peak vs average rates

Each flow has	Peak rate	P
	Average rate	A

Reservation must reserve P, but level of utilization is A/P

P=100 Mbps, A=10 Mbps, level of utilization=10%

On-demand can usually achieve higher level of utilization

depends on degree of sharing and burstiness of flows

Ultimately, it depends on the application

Reservation makes sense when P/A is small

voice traffic has a ratio of 3 or so

Reservation wastes capacity when P/A is big

data applications are bursty, ratios >100 are common

Reservation makes sense when P/A is small

voice traffic has a ratio of 3 or so

Reservation wastes capacity when P/A is big

data applications are bursty, ratios >100 are common

That’s why the phone network used reservations

… and why the Internet does not!

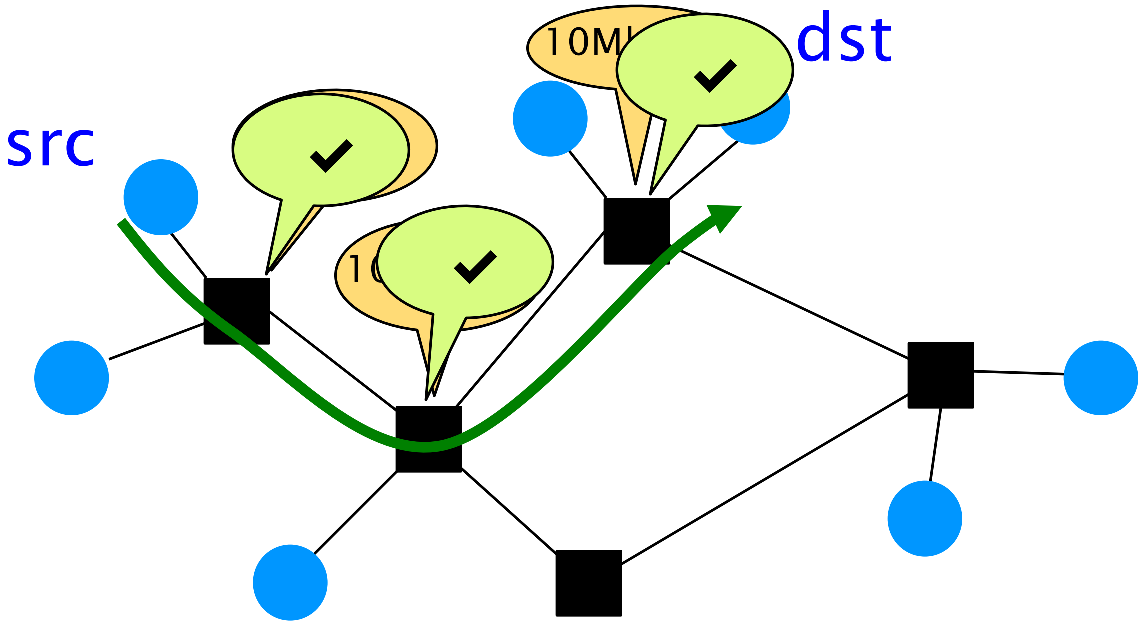
The two approaches are implemented using circuit-switching or packet-switching, respectively

	Reservation	On-demand
implementation	circuit-switching	packet-switching

	Reservation	On-demand
implementation	circuit-switching	packet-switching

Circuit switching relies on

the Resource Reservation Protocol



(1) src sends a reservation request for 10Mbps to dst

(2) switches "establish a circuit"

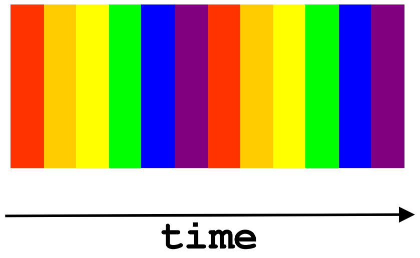
(3) src starts sending data

(4) src sends a "teardown circuit" message

There exist many kinds of circuits

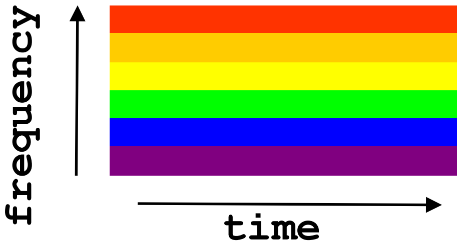
Time-based multiplexing

- divide time in slots
- allocate one slot per circuit

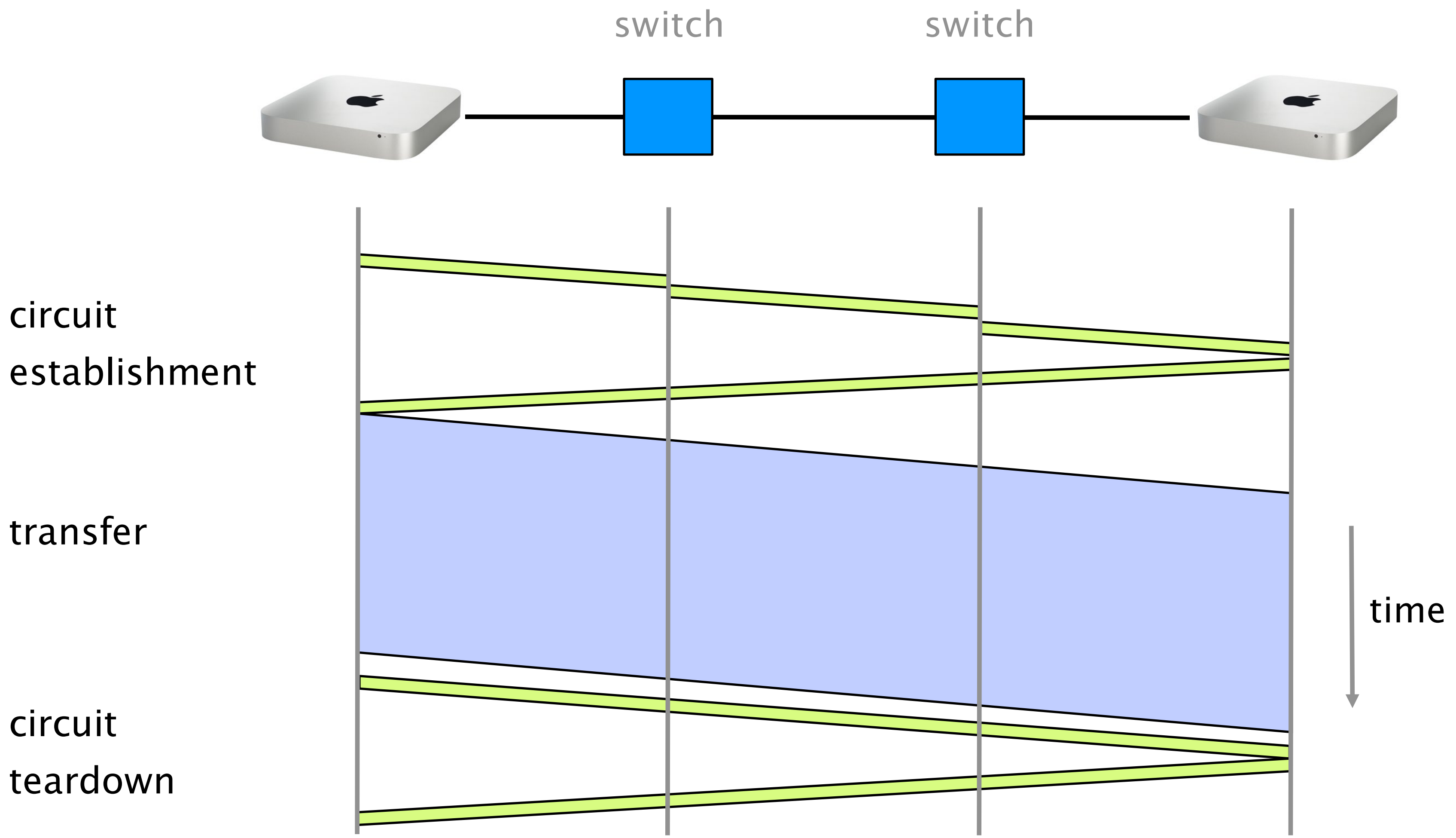


Frequency-based multiplexing

- divide spectrum in frequency bands
- allocate one band per circuit



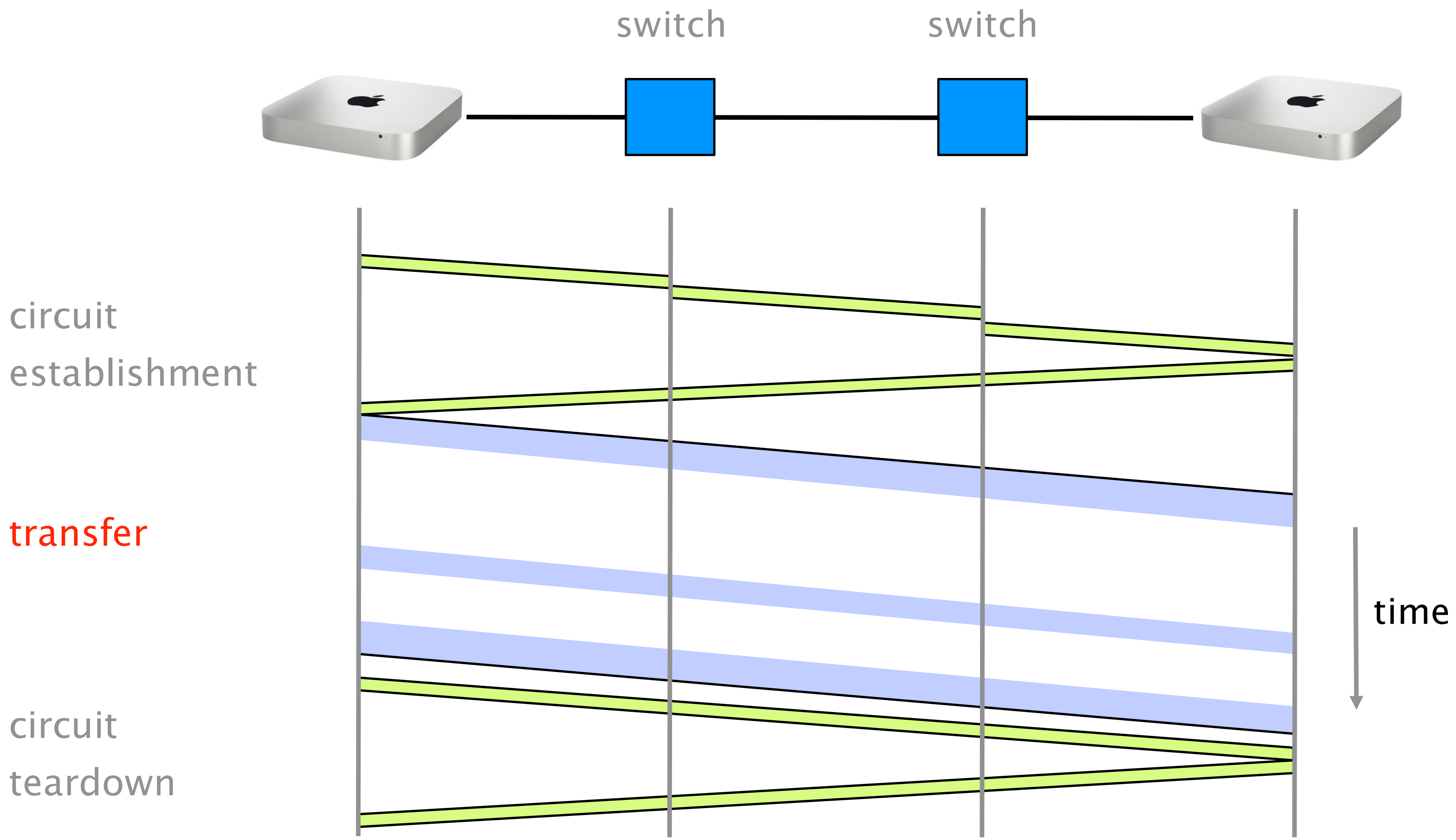
Let's walk through example of data transfer using circuit switching



The efficiency of the transfer depends on how utilized the circuit is once established

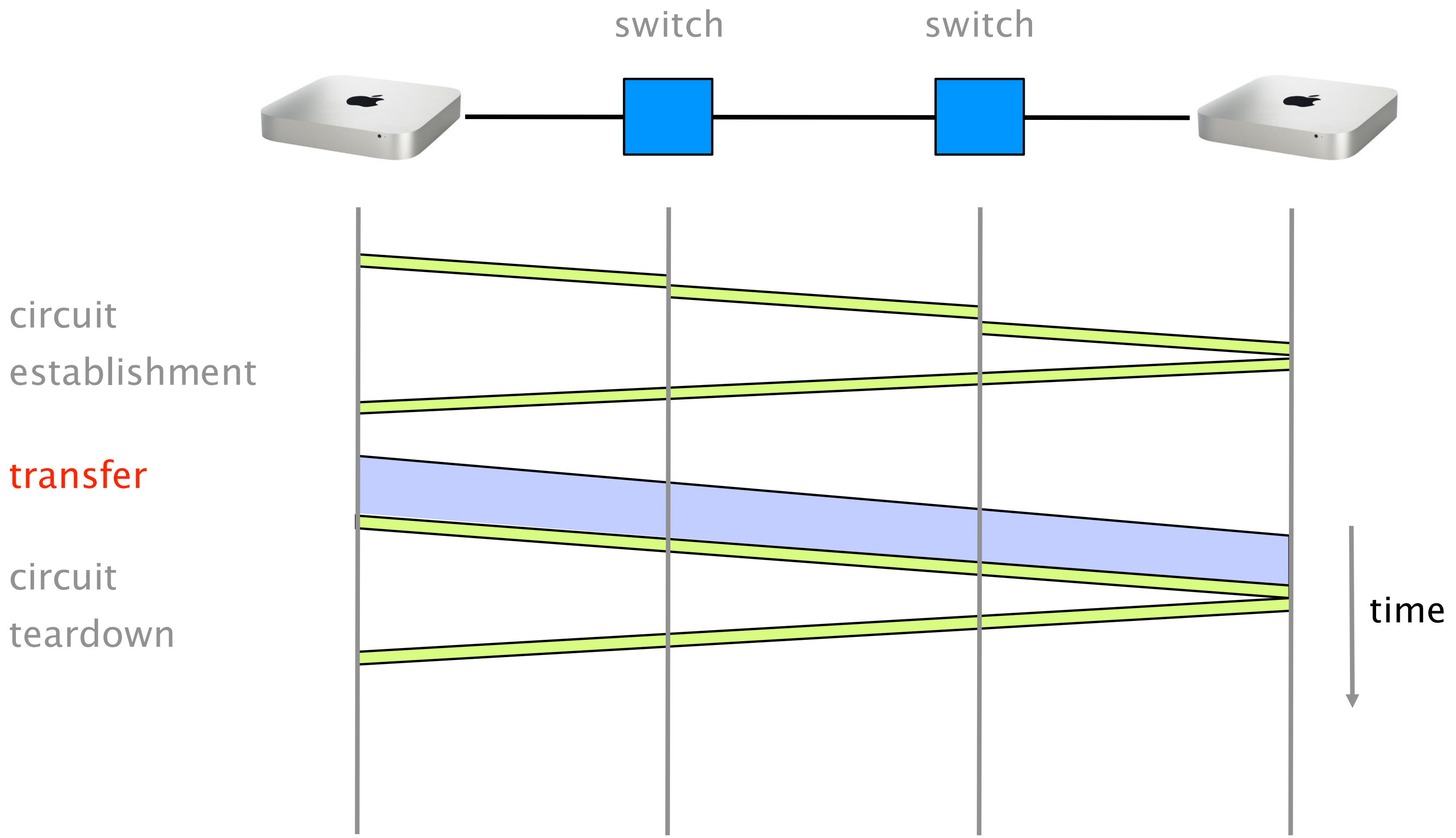
This is an example of poor efficiency

The circuit is mostly idle due to traffic bursts

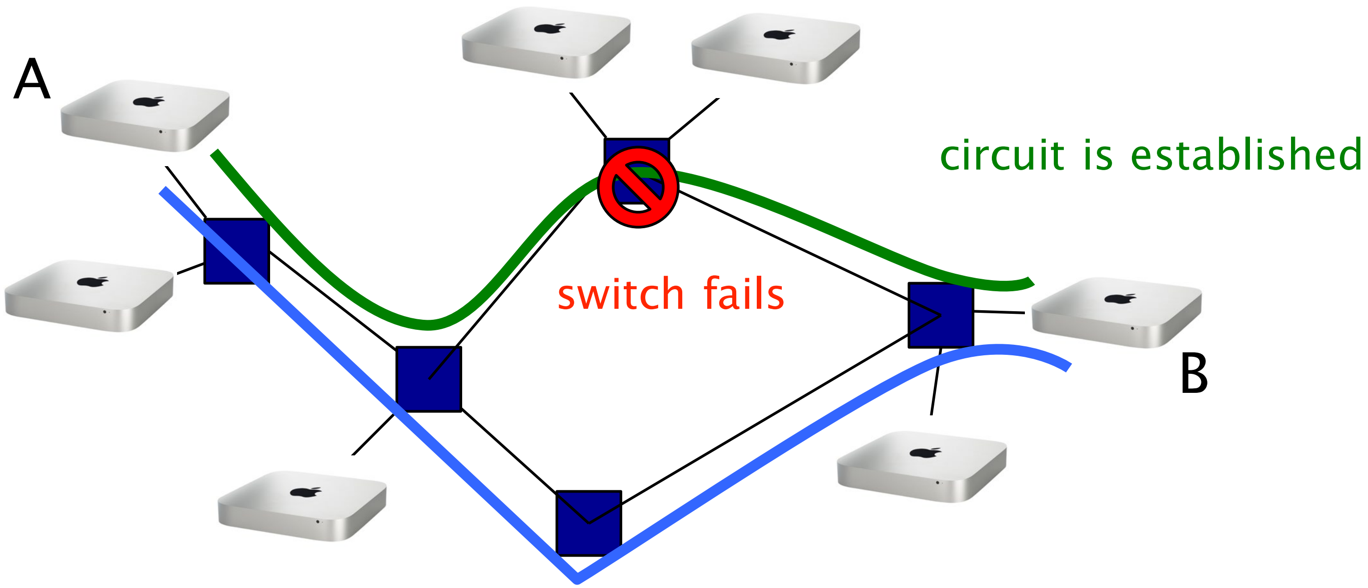


This is another example of poor efficiency

The circuit is used for a short amount of time



Another problem of circuit switching is that it doesn't route around trouble



A is forced to signal a new circuit to restore communication

Pros and cons of circuit switching

advantages	disadvantages
predictable performance	inefficient if traffic is bursty or short
simple & fast switching once circuit established	complex circuit setup/teardown which adds delays to transfer
	requires new circuit upon failure

What about packet switching?

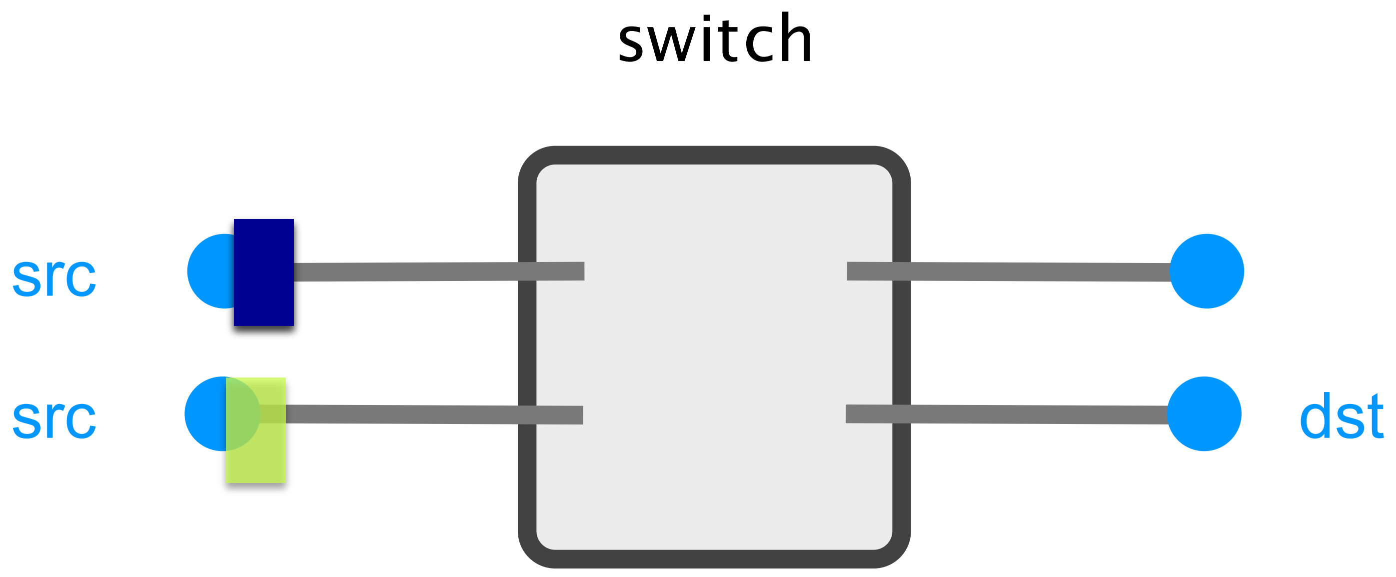
Reservation

On-demand

circuit-switching

packet-switching

In packet switching,
data transfer is done using independent packets

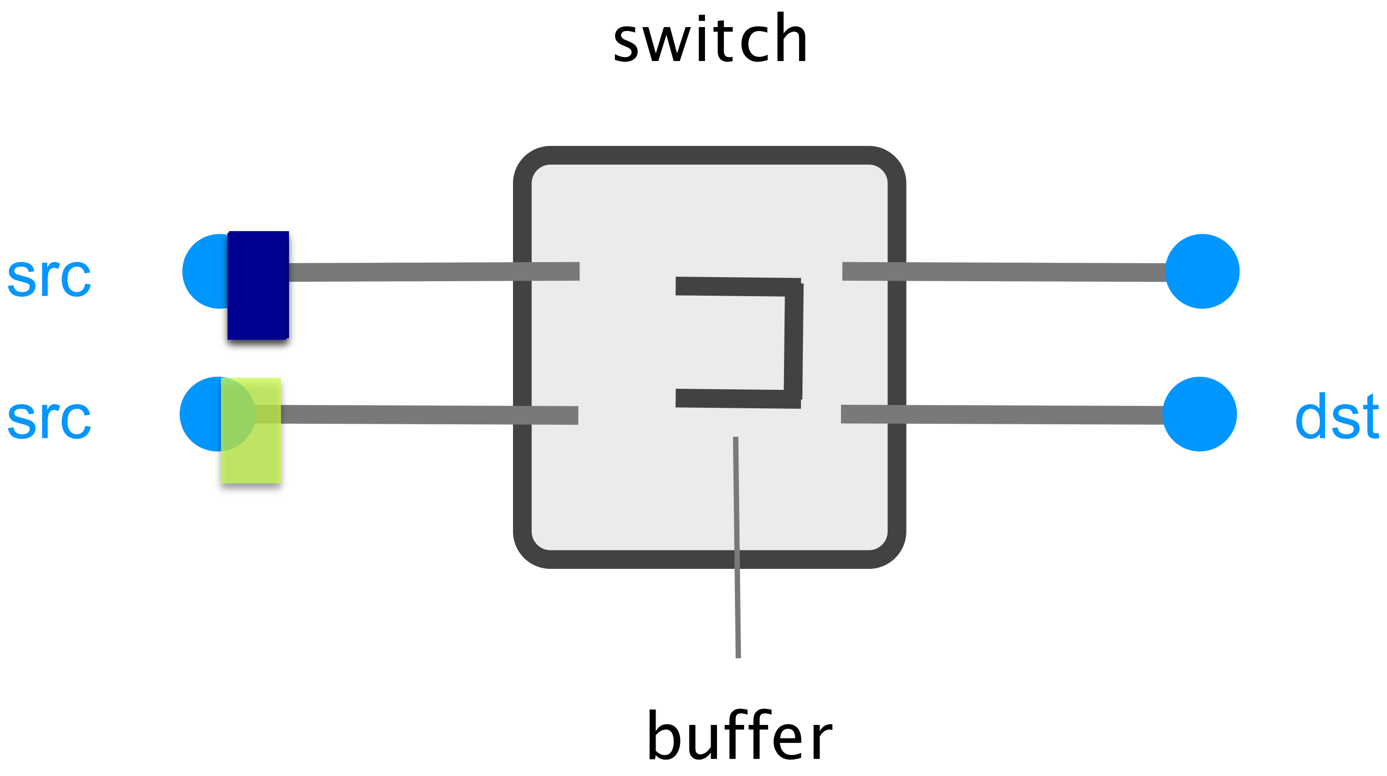


Each packet contains a destination (dst)

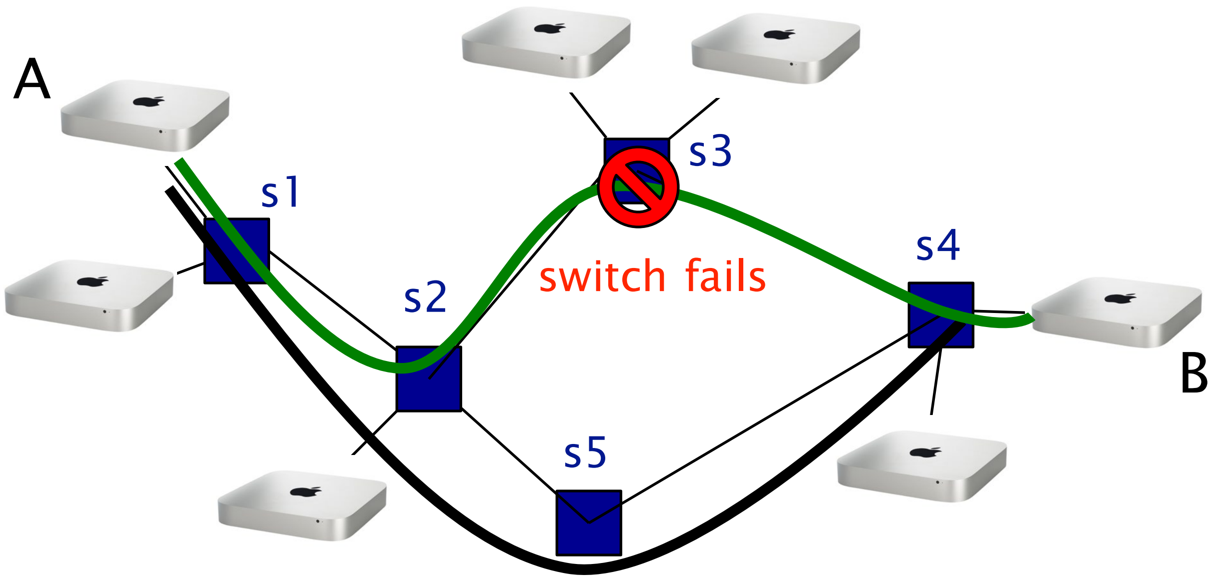
Since packets are sent without global coordination, they can “clash” with each other

To absorb transient overload,
packet switching relies on buffers

To absorb transient overload,
packet switching relies on buffers



Packet switching routes around trouble



route is recomputed on the fly by s2

Pros and cons of packet switching

advantages

efficient use of resources

simpler to implement

route around trouble

disadvantages

unpredictable performance

requires buffer management and congestion control

Packet switching beats circuit switching
with respect to *resiliency* and *efficiency*

Internet ♥ packets

Packet switching will be our focus for the rest of the course

Communication Networks and Internet Technology

Part 1: Overview

What is a network made of?

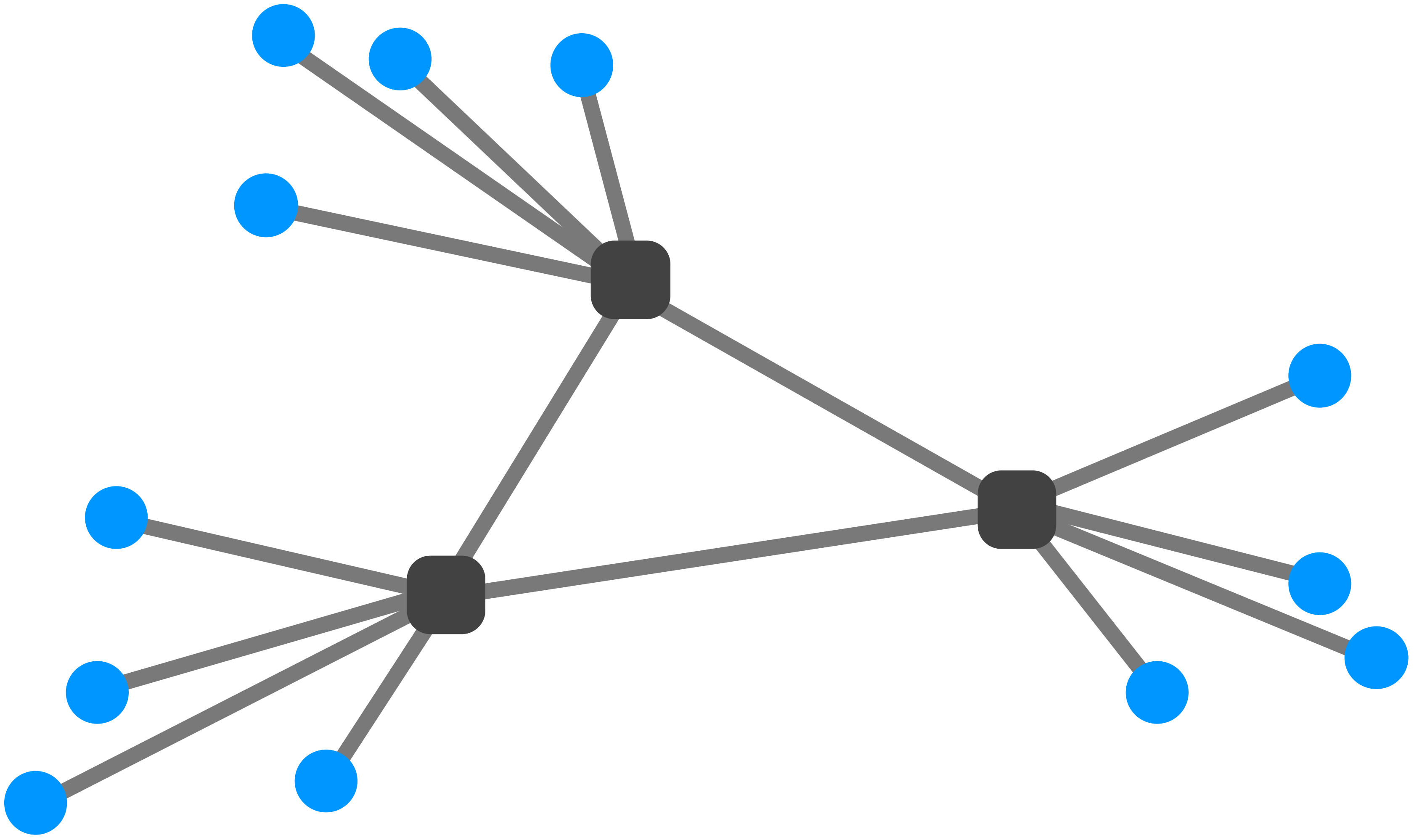
How is it shared?

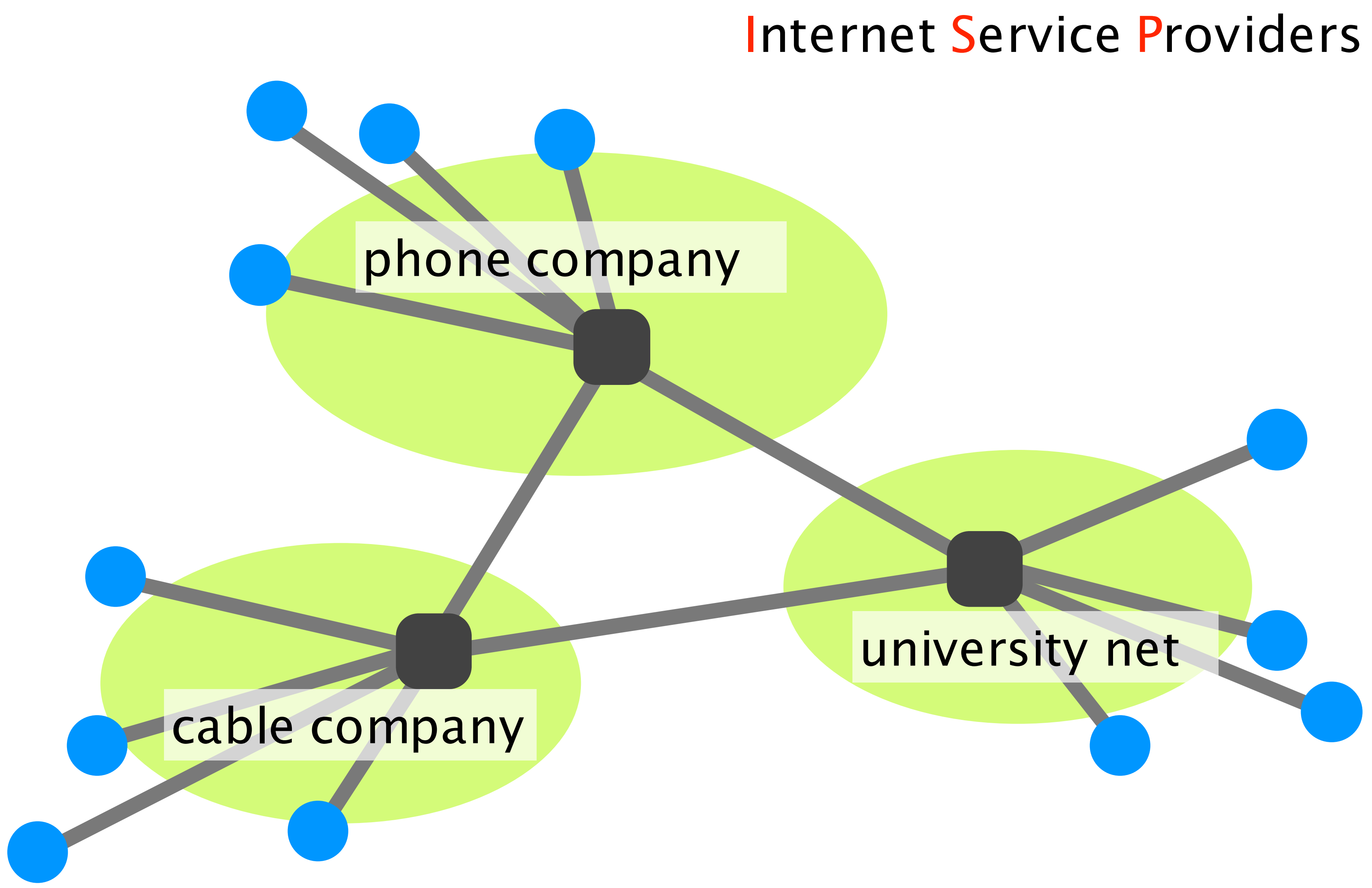
#3 How is it organized?

How does communication happen?

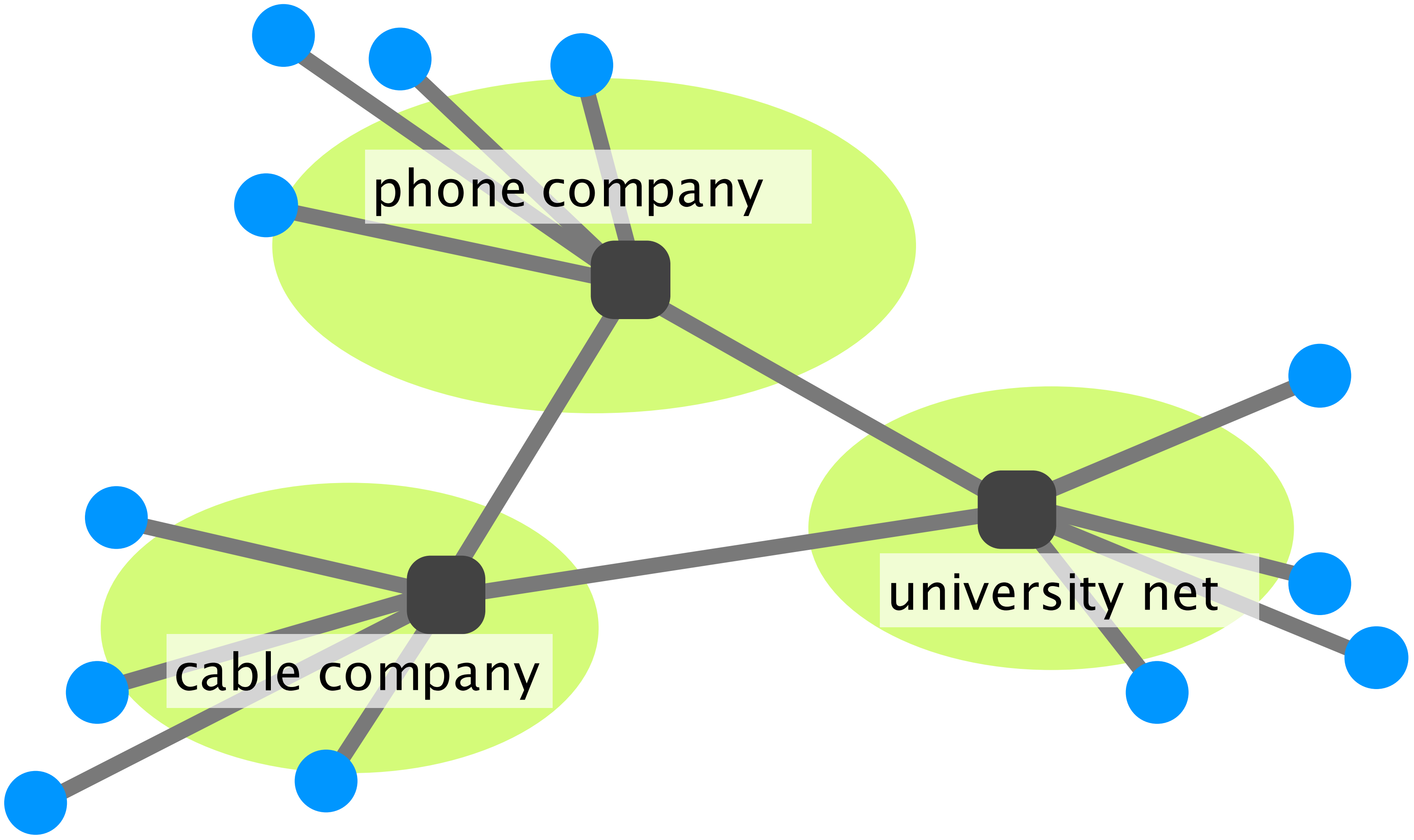
How do we characterize it?

The Internet is a network of networks

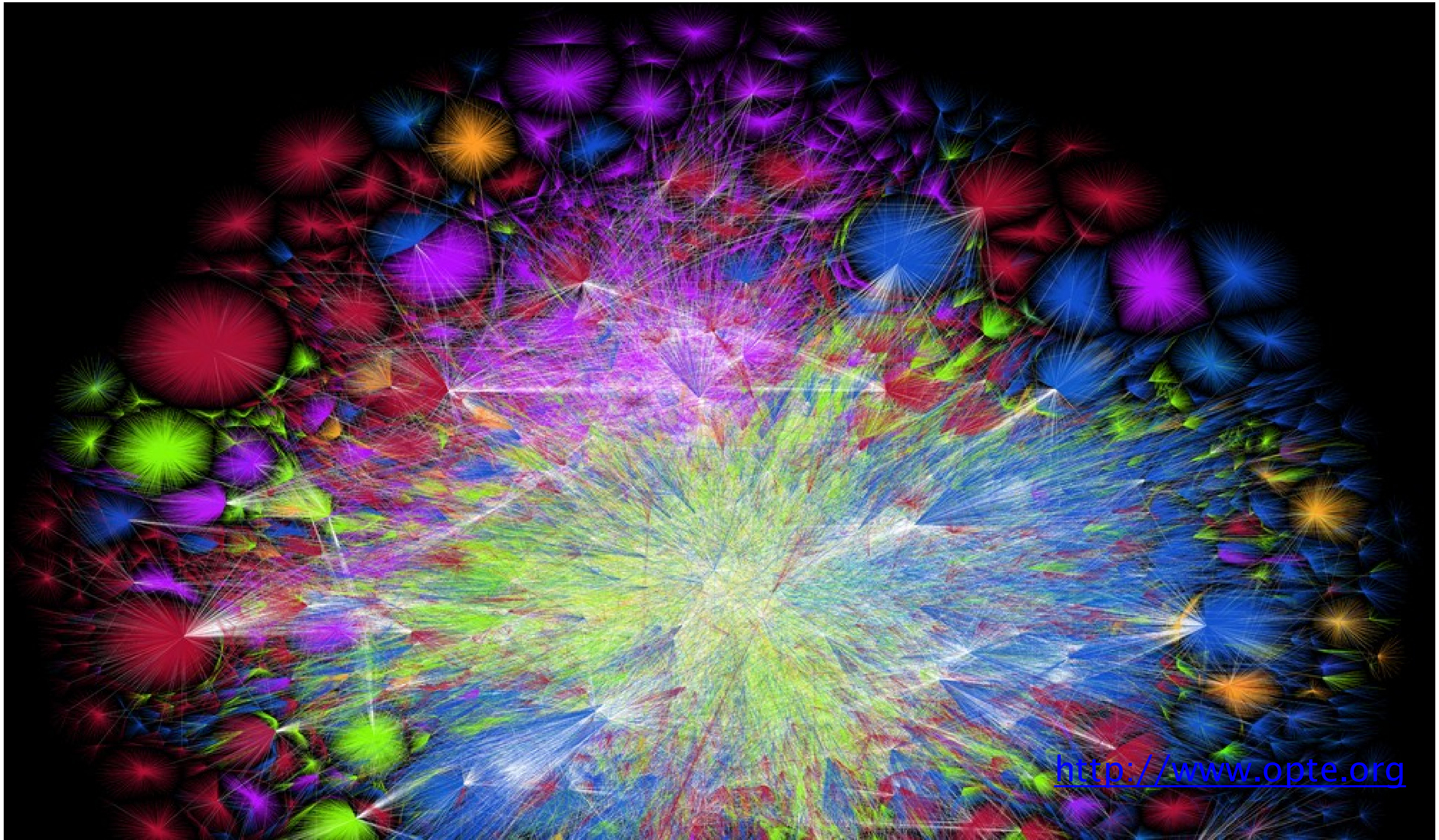


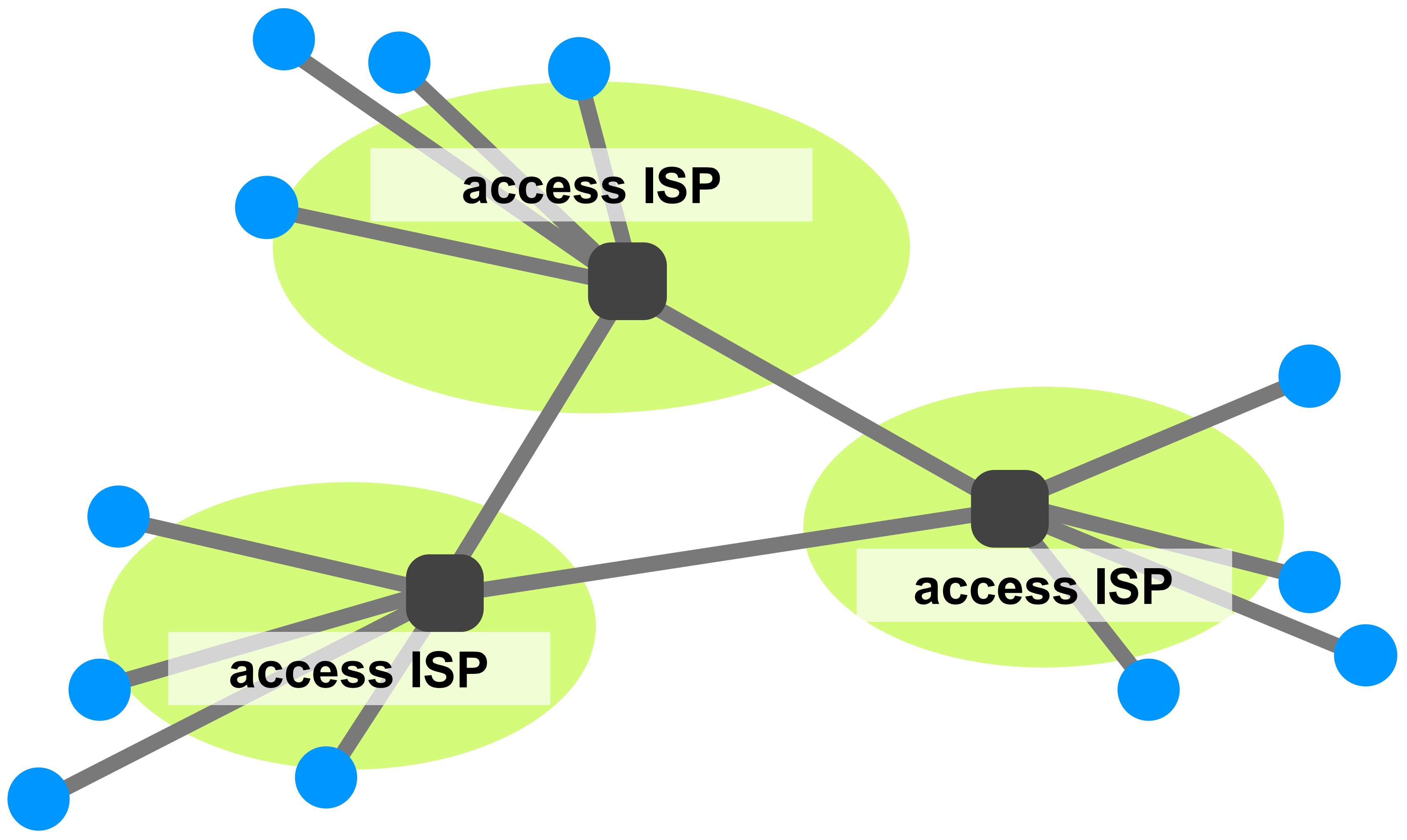

Internet Service Providers
phone company
university net
cable company

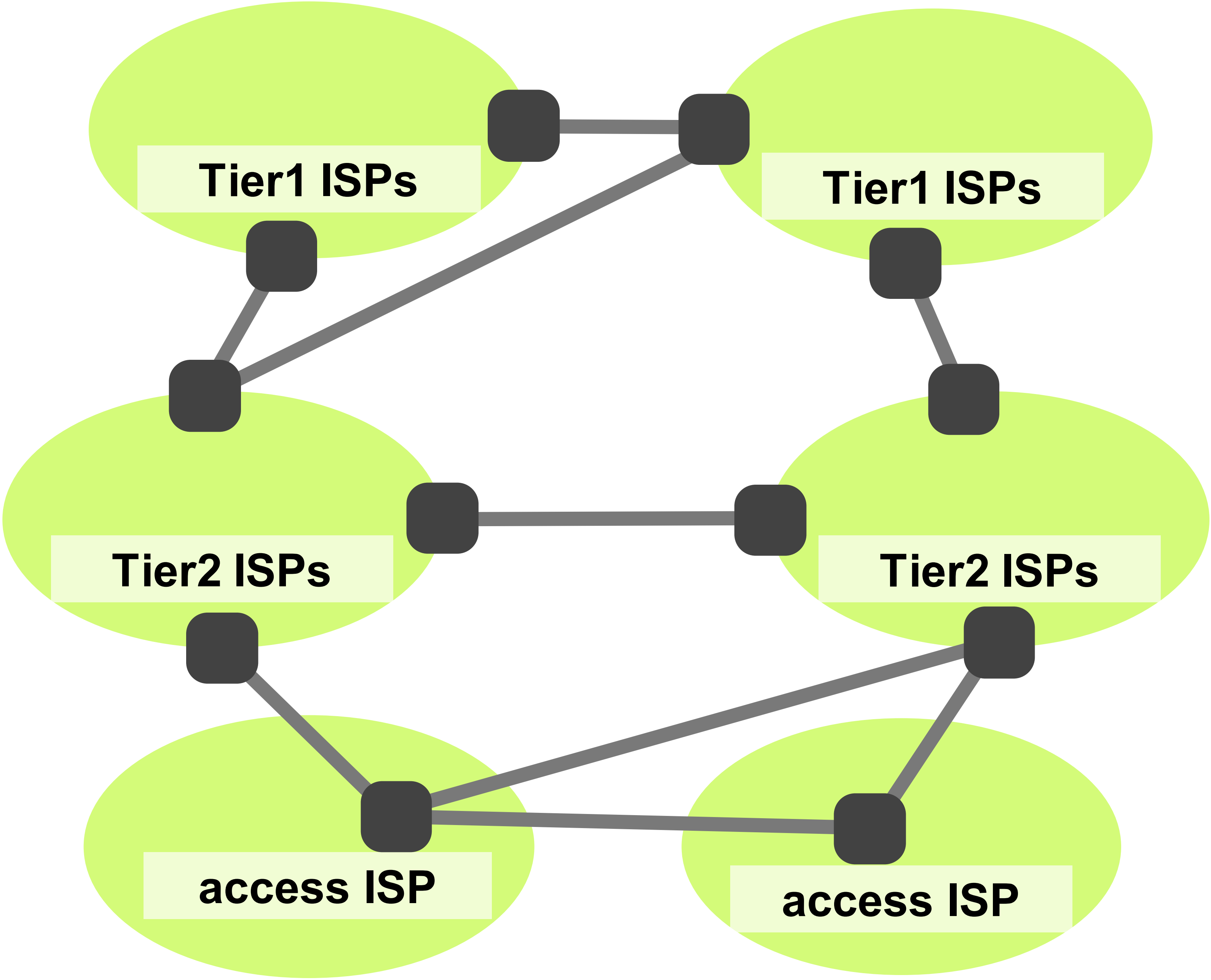
So far our view of the Internet



The Internet is a tad more complex…

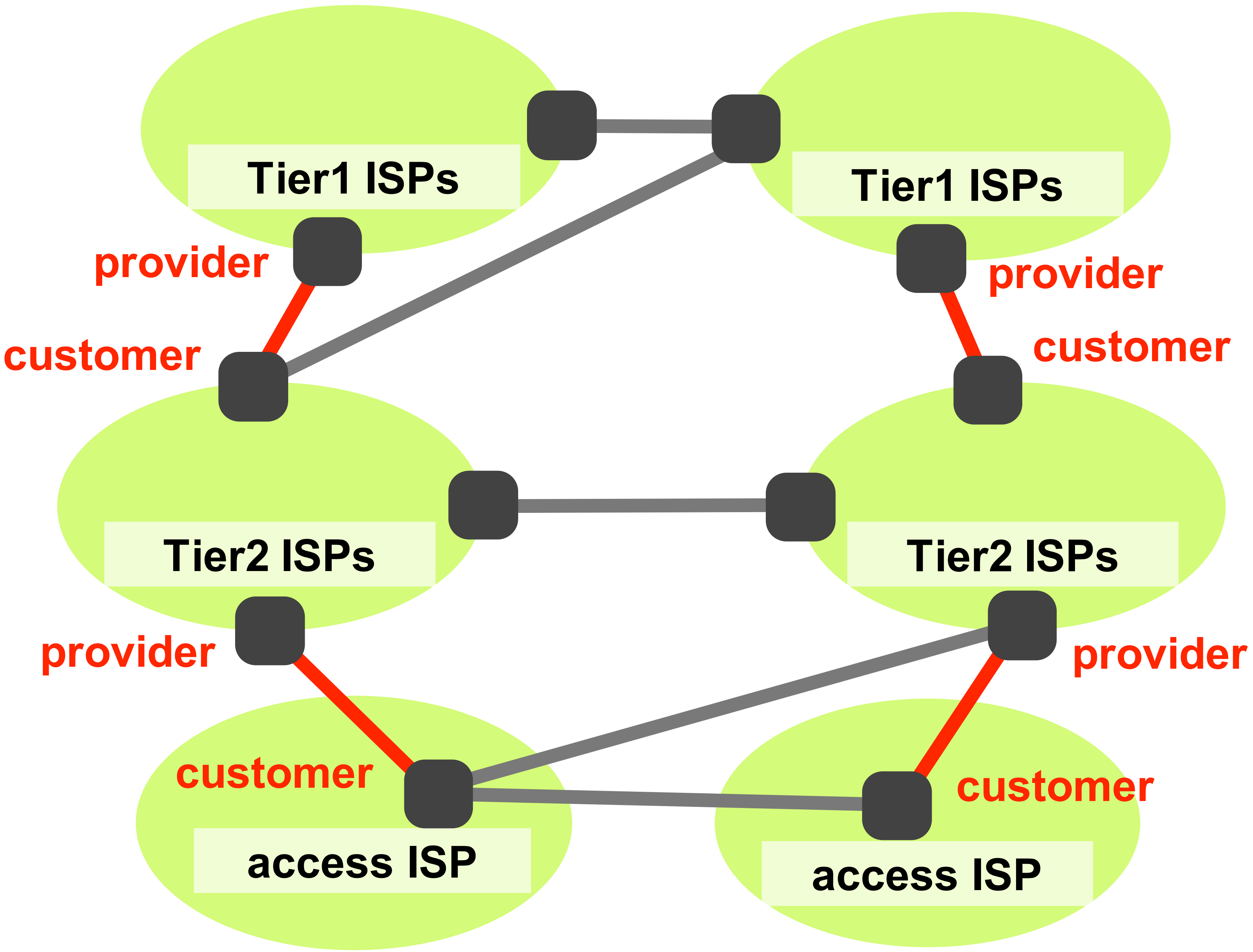



access ISP
access ISP
access ISP


Tier1 ISPs
Tier1 ISPs
Tier2 ISPs
Tier2 ISPs
access ISP
access ISP

The Internet has a hierarchical structure

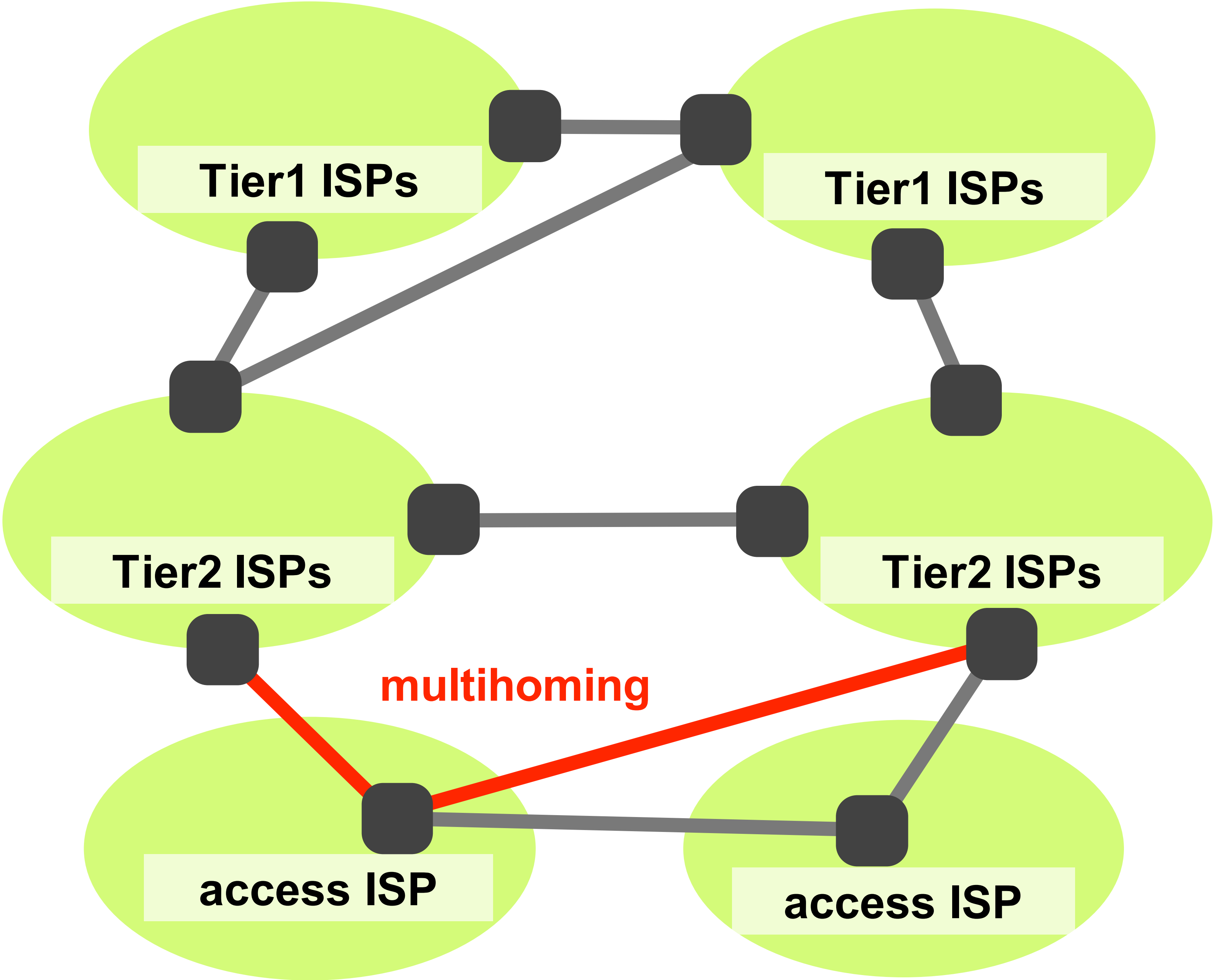
Tier-1 international	have no provider
Tier-2 national	provide transit to Tier-3s have at least one provider
Tier-3 local	do not provide any transit have at least one provider



Tier1 ISPs
Tier1 ISPs
provider
provider
customer
customer
Tier2 ISPs
Tier2 ISPs
provider
provider
customer
customer
access ISP
access ISP

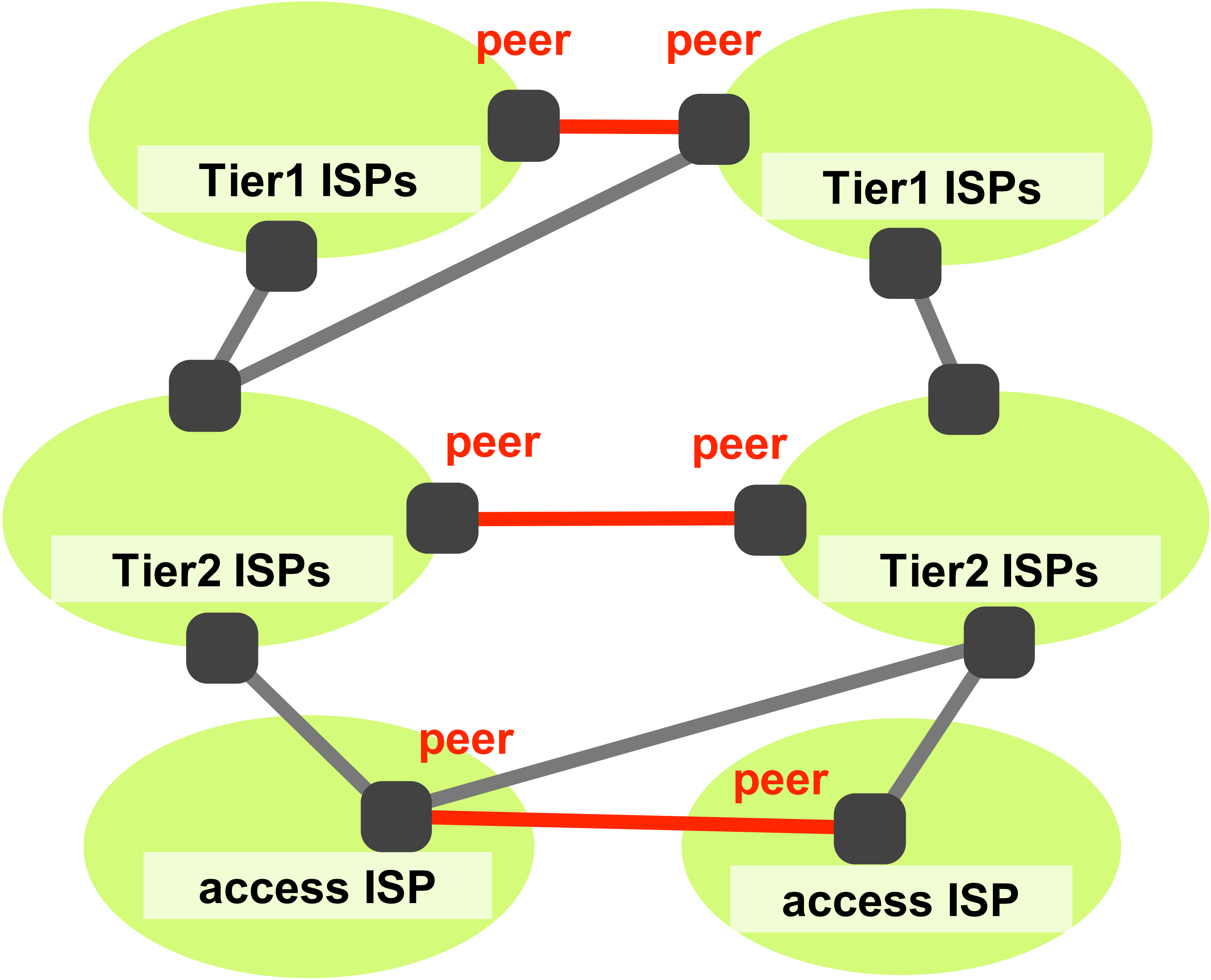
The distribution of networks in Tiers is extremely skewed towards Tier-3s

	total	~60,000 networks
Tier-1 international	have no provider	~12
Tier-2 national	provide transit to Tier-3s have at least one provider	~1,000s
Tier-3 local	do not provide any transit have at least one provider	85-90%


Tier1 ISPs
Tier1 ISPs
Tier2 ISPs
Tier2 ISPs
multihoming
access ISP
access ISP

Some networks have an incentive to connect directly,
to reduce their bill with their own provider

This is known as "peering"



peer
peer
Tier1 ISPs
Tier1 ISPs
peer
peer
Tier2 ISPs
Tier2 ISPs
peer
peer
access ISP
access ISP

Interconnecting each network to its neighbors one-by-one is not cost effective

Physical costs

of provisioning or renting physical links

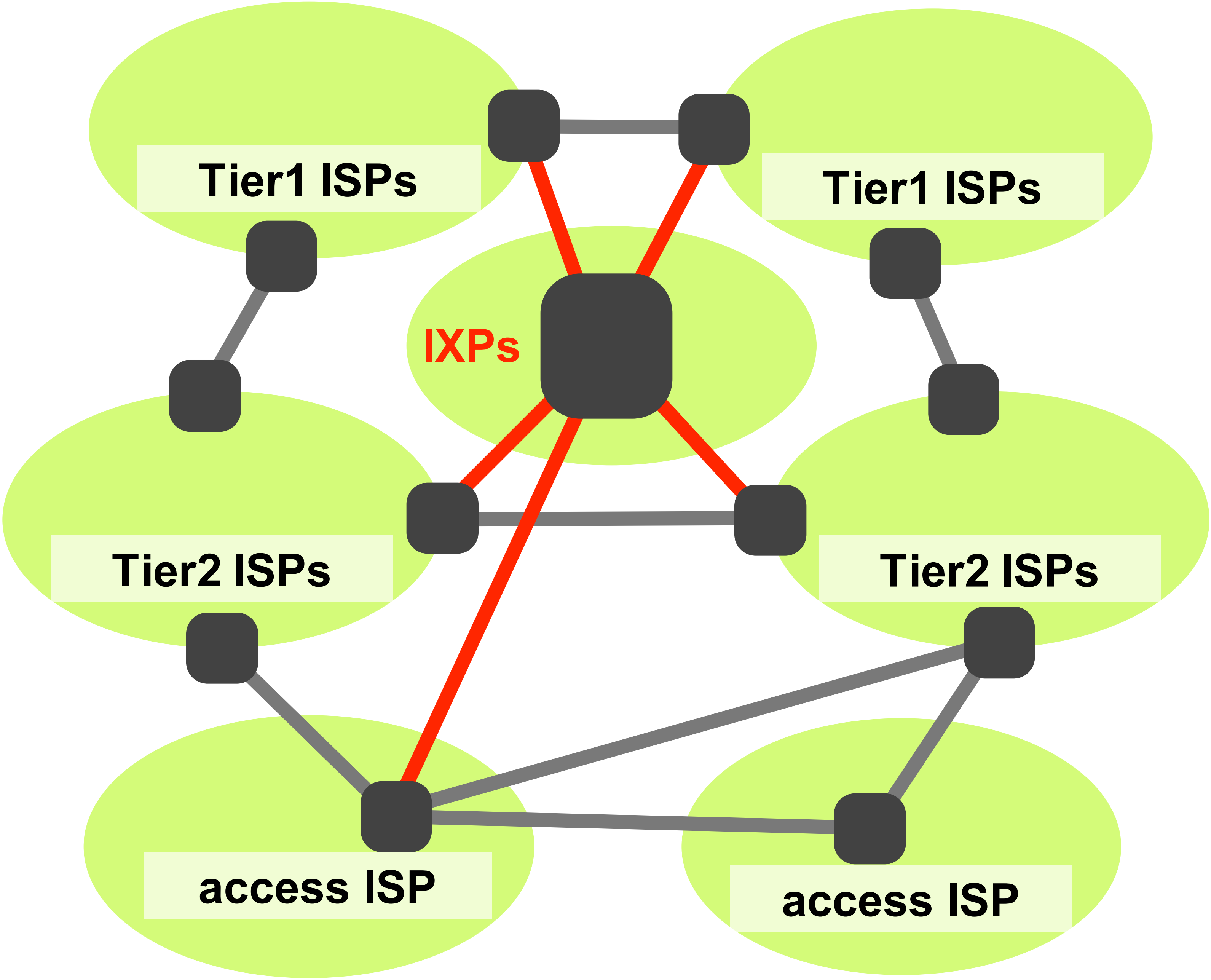
Bandwidth costs

a lot of links are not necessarily fully utilized

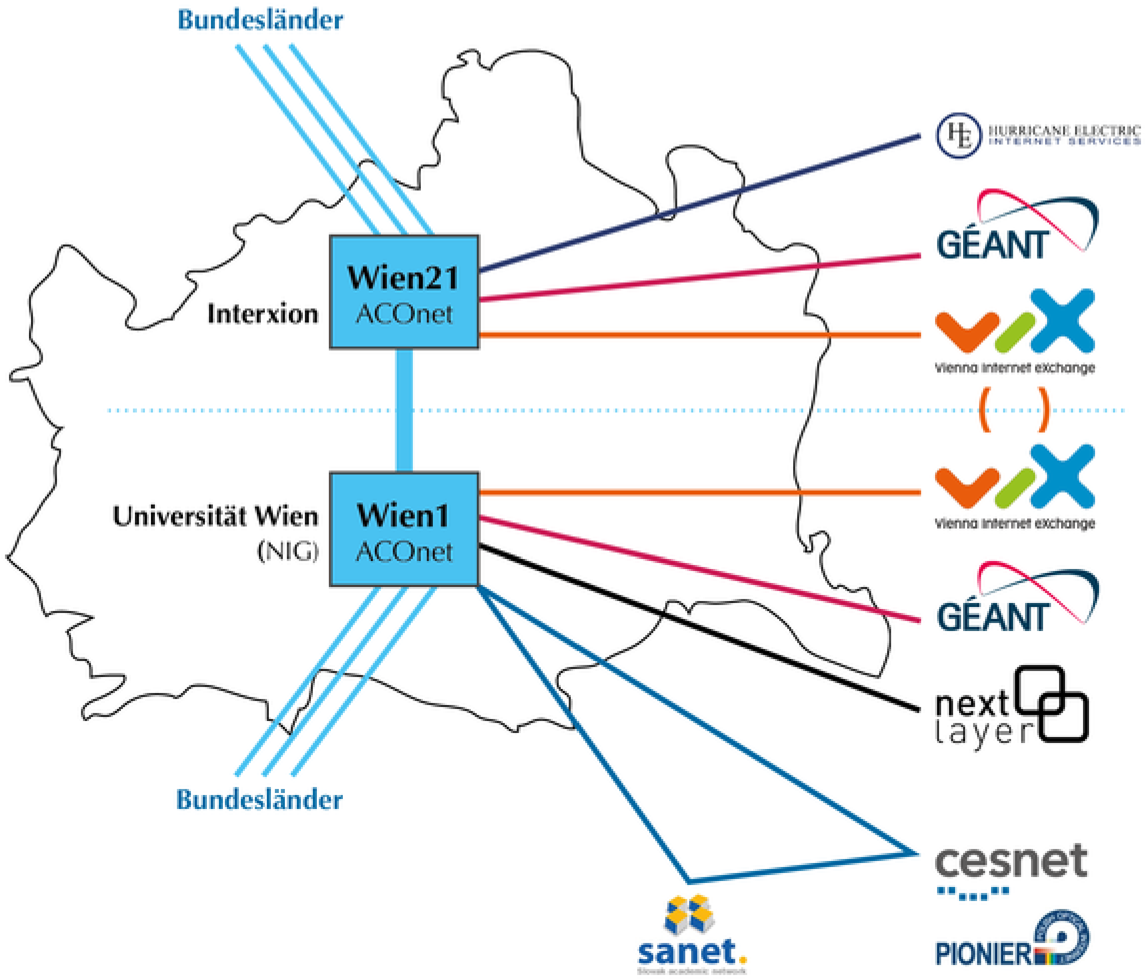
Human costs

to manage each connection individually

Internet eXchange Points (IXPs) solve these problems
by letting *many* networks connect in one location


Tier1 ISPs
Tier1 ISPs
IXPs
Tier2 ISPs
Tier2 ISPs
access ISP
access ISP

Let's explore our network environment



Communication Networks and Internet Technology

Part 1: General overview

What is a network made of?

How is it shared?

How is it organized?

#4 How does communication happen?

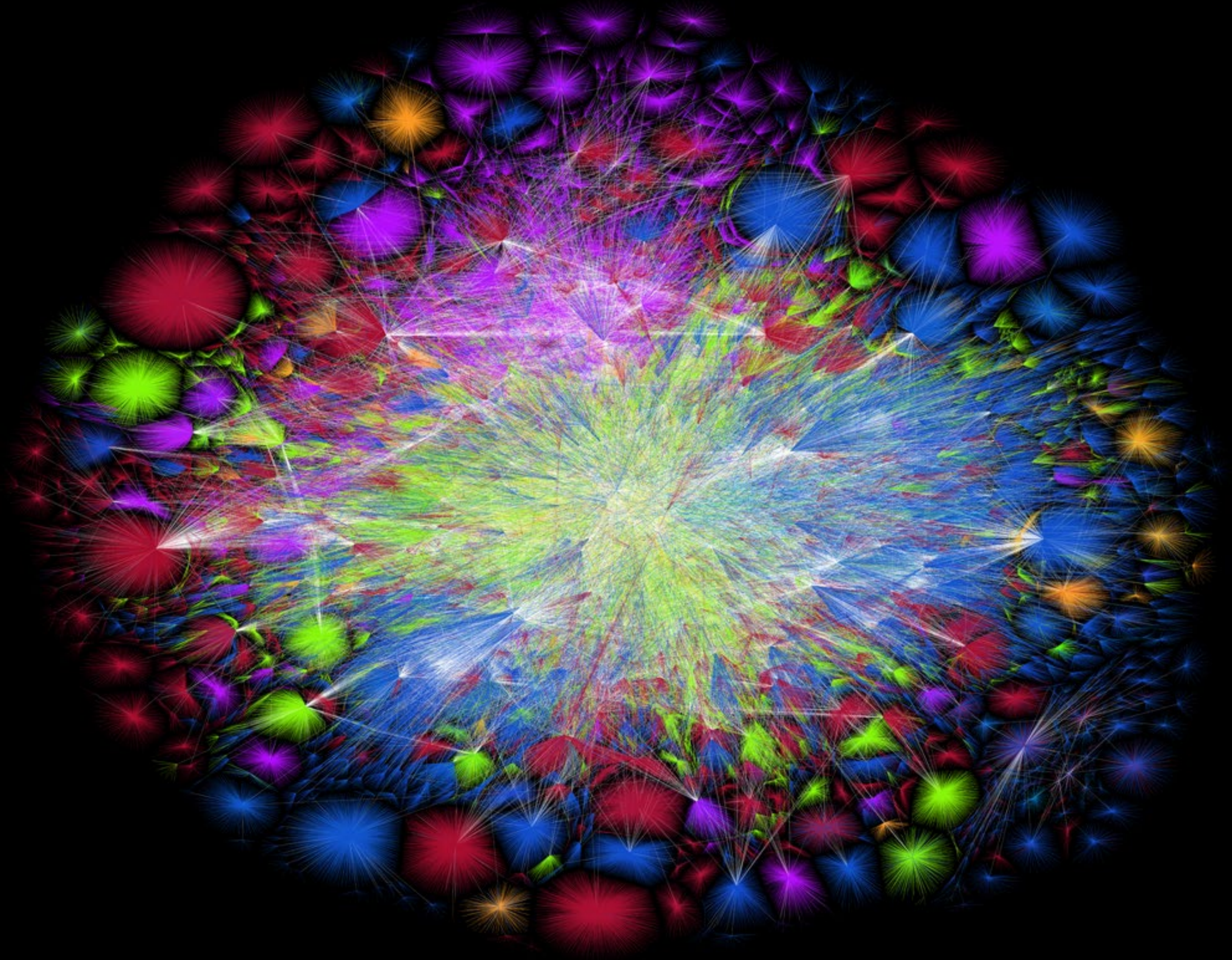
How do we characterize it?

The Internet should allow

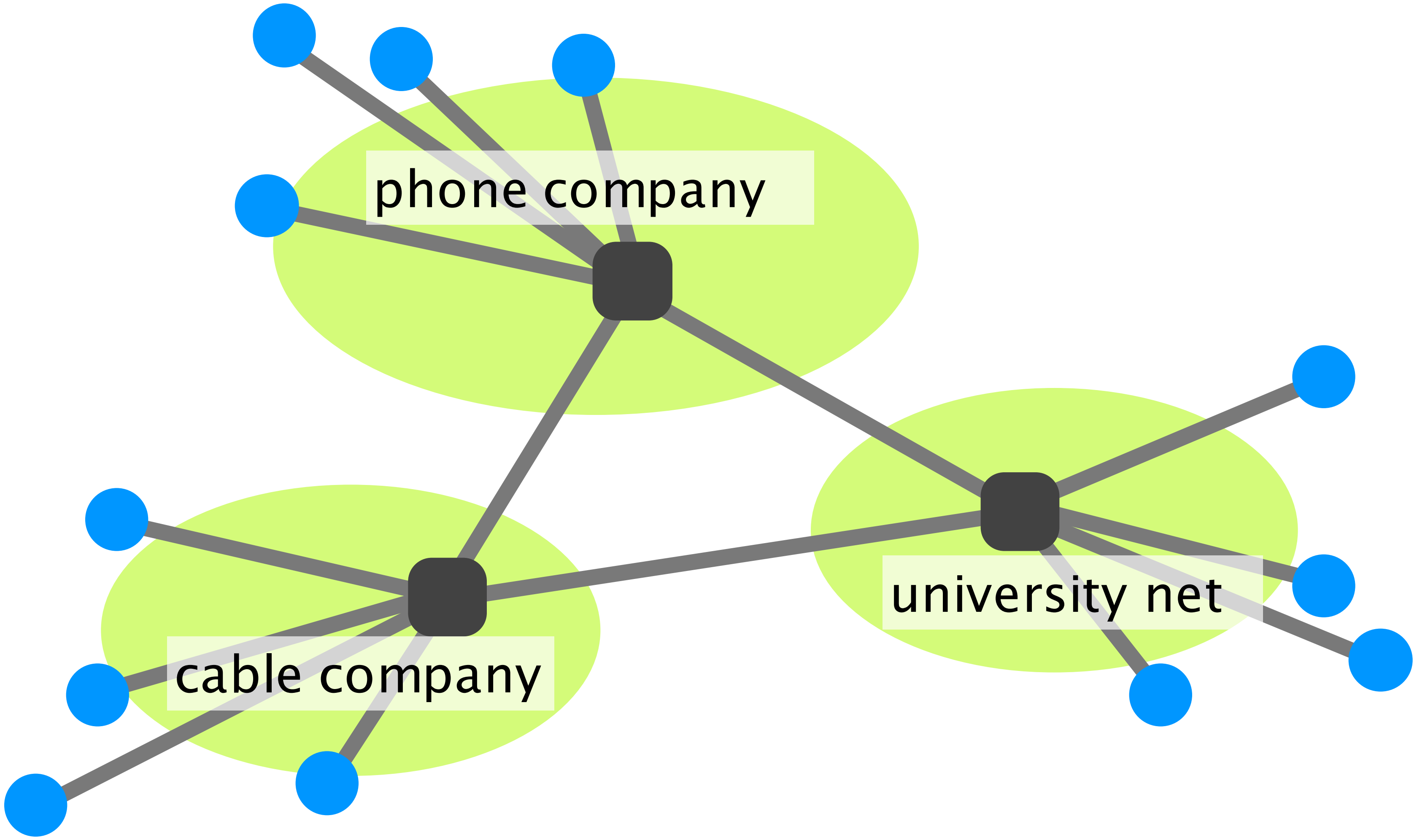
processes on different hosts
to exchange data

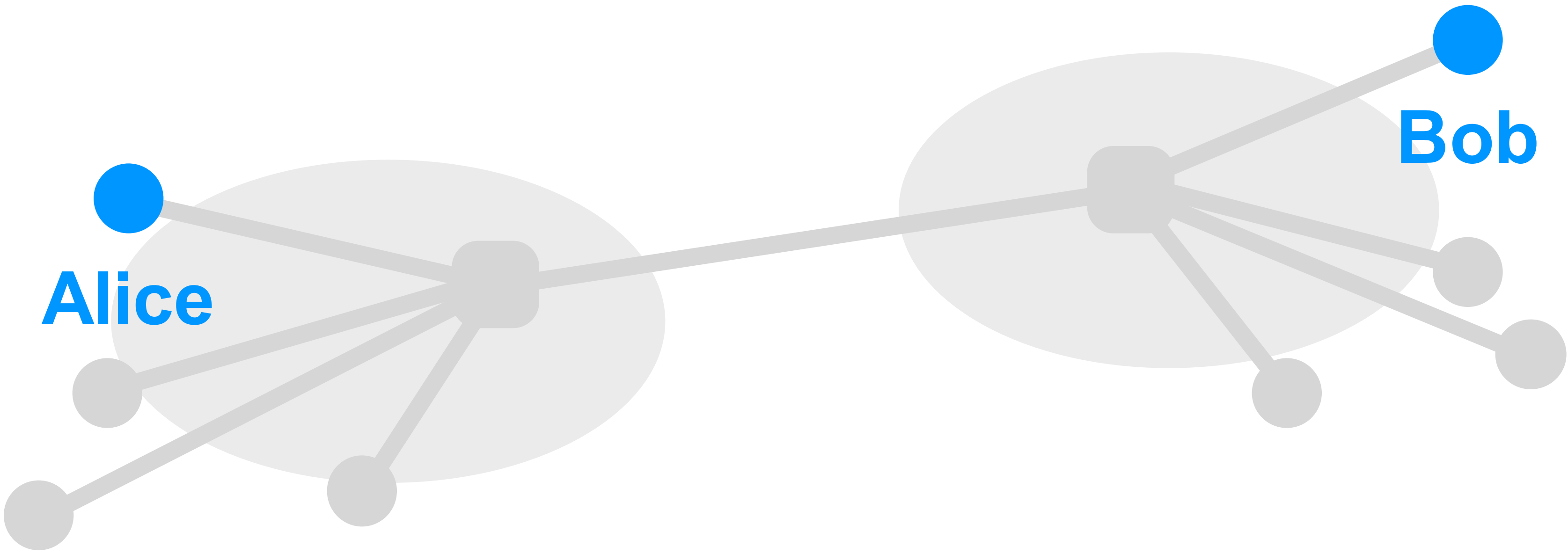
everything else is just commentary…

How do you exchange data in a network as complex as **this?**



http://www.opte.org

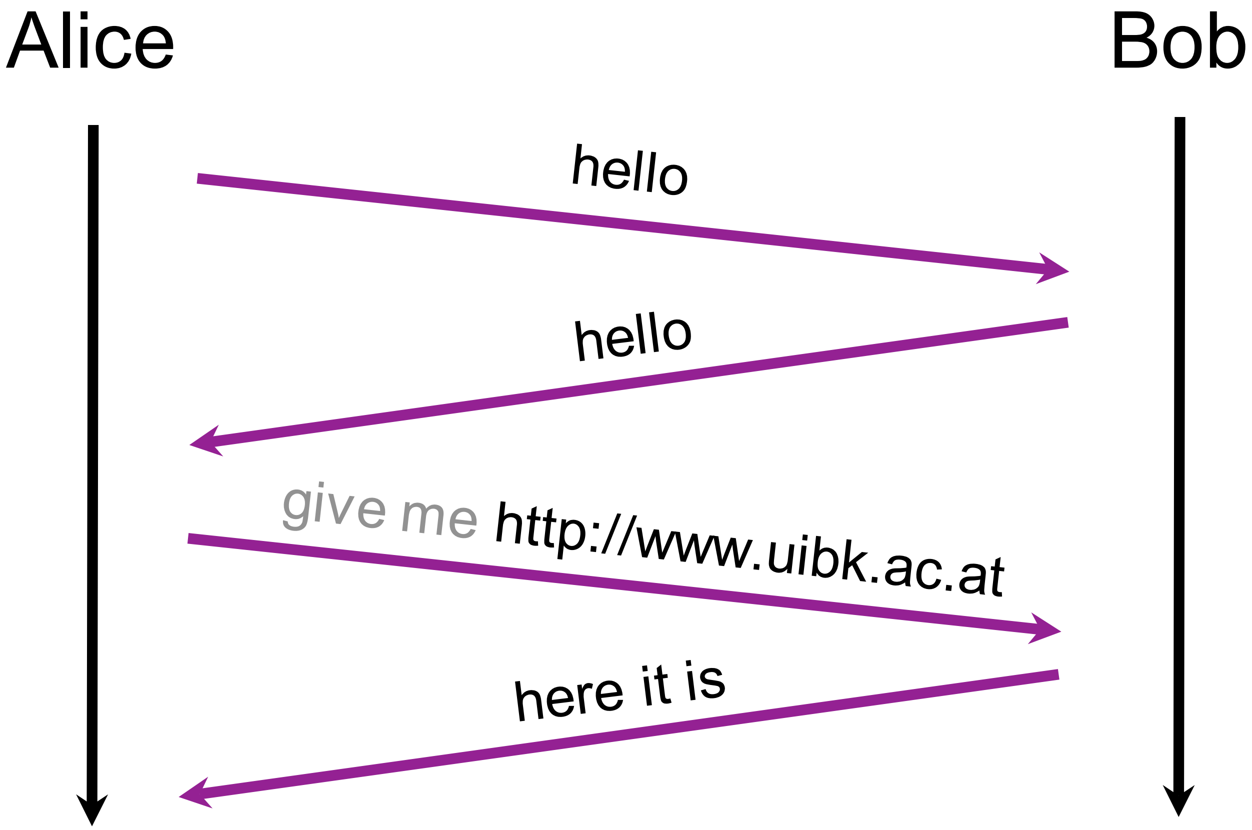

phone company
university net
cable company


Alice
Bob

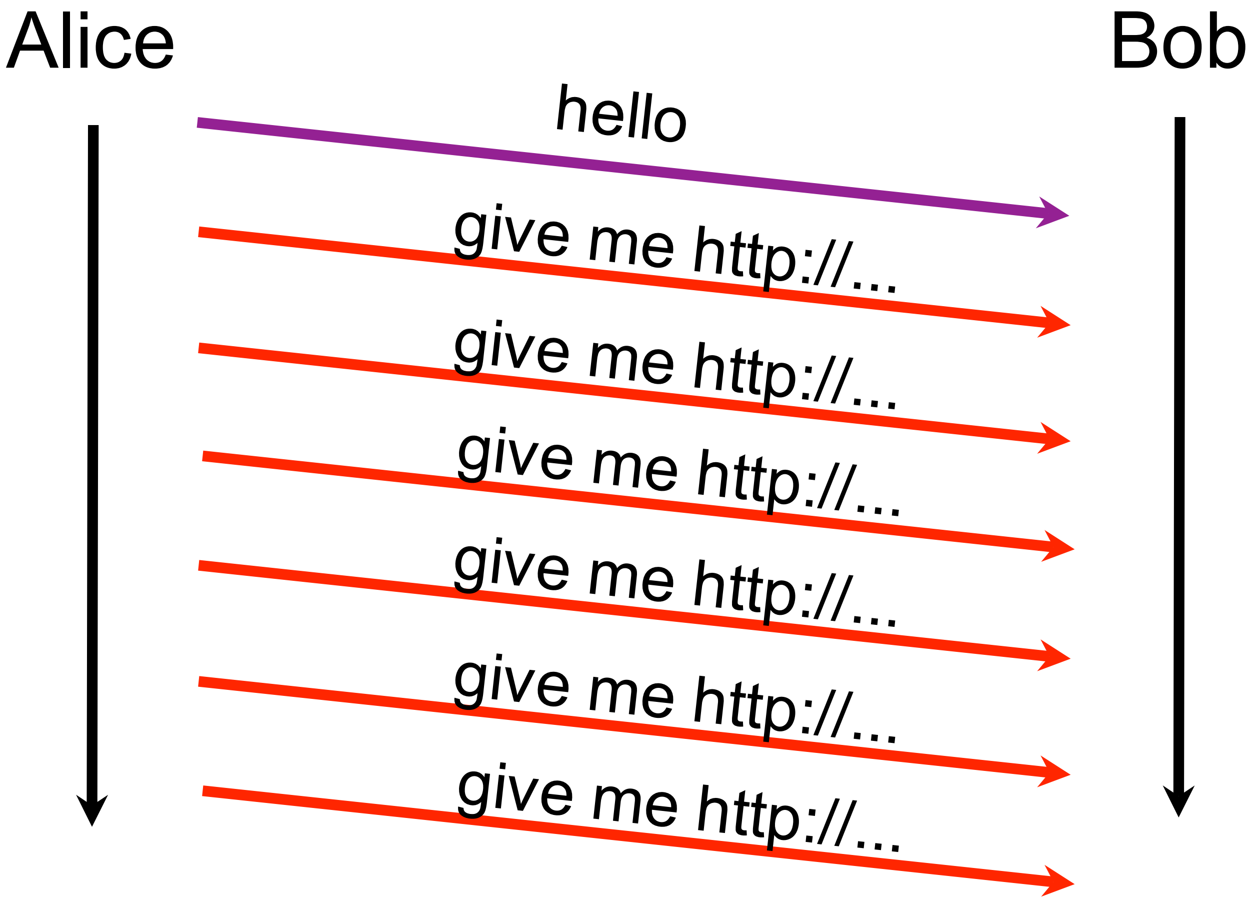
To exchange data, Alice and Bob use a set of network protocols

A protocol is like a conversational convention:

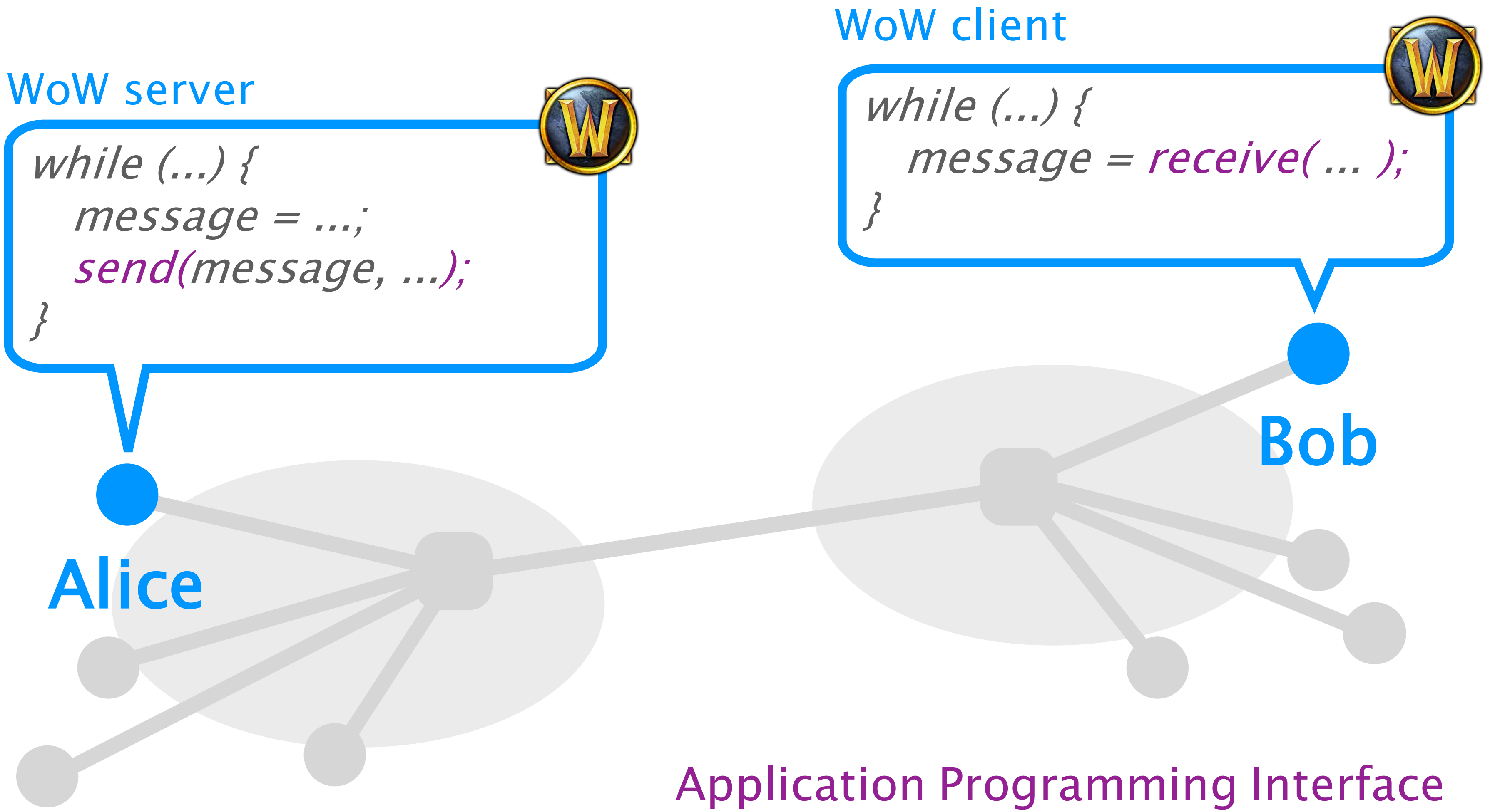
who should talk next and how they should respond



Sometimes implementations are not compliant…

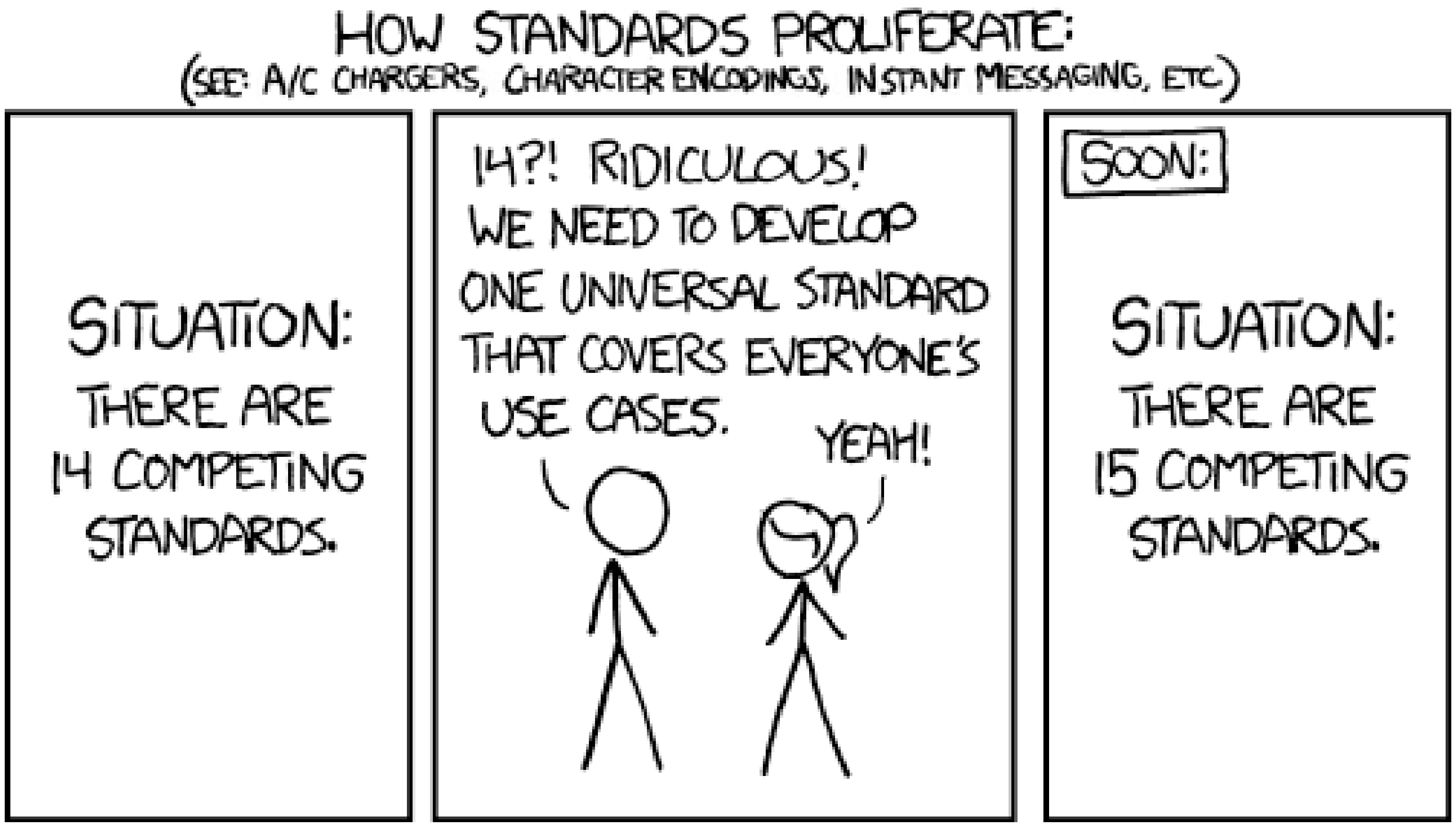


Each protocol is governed
by a specific interface



In practice, there exists **a lot** of network protocols.

How does the Internet organize **this?**



https://xkcd.com/927/

Modularity is a key component of any good system

Problem

can't build large systems out of spaghetti code

hard (if not, impossible) to understand, debug, update

need to bound the scope of changes

evolve the system without rewriting it from scratch

Solution

Modularity is how we do it

…and understand the system at a higher-level

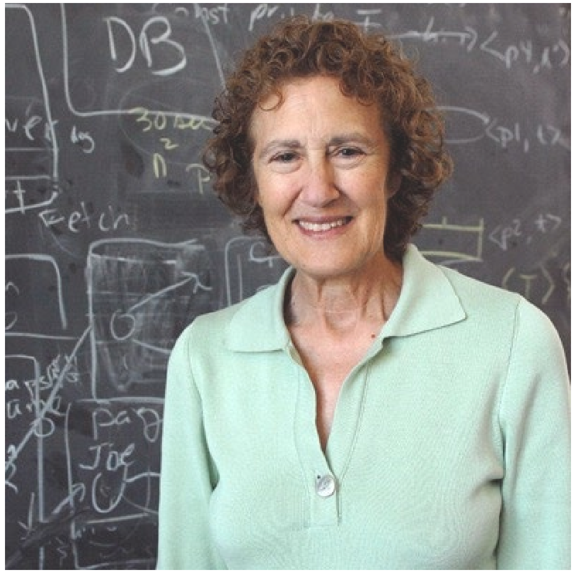


Photo: Donna Coveney

Modularity,
based on abstraction,
is *the* way things get done

— *Barbara Liskov, MIT*

To provide structure to the design of network protocols, network designers organize protocols in layers

and the network hardware/software that implement them

Internet communication can be decomposed
in 5 independent layers (or 7 layers for the OSI model)

	layer
L5	Application
L4	Transport
L3	Network
L2	Link
L1	Physical

Each layer provides a service to the layer above

	layer	service provided:
L5	Application	network access
L4	Transport	end-to-end delivery (reliable or not)
L3	Network	global best-effort delivery
L2	Link	local best-effort delivery
L1	Physical	physical transfer of bits

Each layer provides a service to the layer above
by using the services of the layer directly below it

Applications

...built on...

Reliable (or unreliable) transport

...built on...

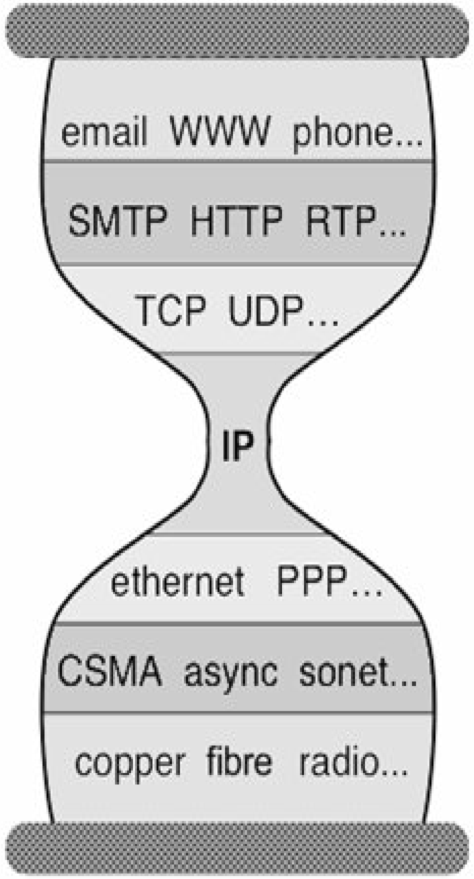
Best-effort global packet delivery

...built on...

Best-effort local packet delivery

...built on...

Physical transfer of bits



Each layer (except for L3) is implemented with different protocols

	layer	protocol
L5	Application	HTTP, SMTP, FTP, SIP, …
L4	Transport	TCP, UDP, SCTP
L3	Network	IP
L2	Link	Ethernet, Wifi, (A/V)DSL, WiMAX, LTE, …
L1	Physical	Twisted pair, fiber, coaxial cable, …

The Internet Protocol (IP) acts as an unifying, network, layer

	layer	protocol
L5	Application	HTTP, SMTP, FTP, SIP, …
L4	Transport	TCP, UDP, SCTP
L3	Network	IP
L2	Link	Ethernet, Wifi, (A/V)DSL, Cable, LTE, …
L1	Physical	Twisted pair, fiber, coaxial cable, …

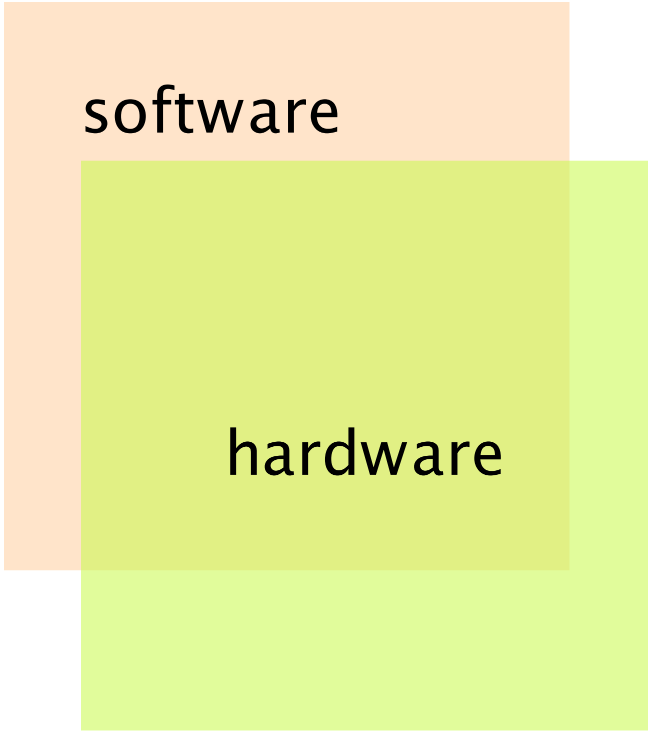
Each layer has a unit of data

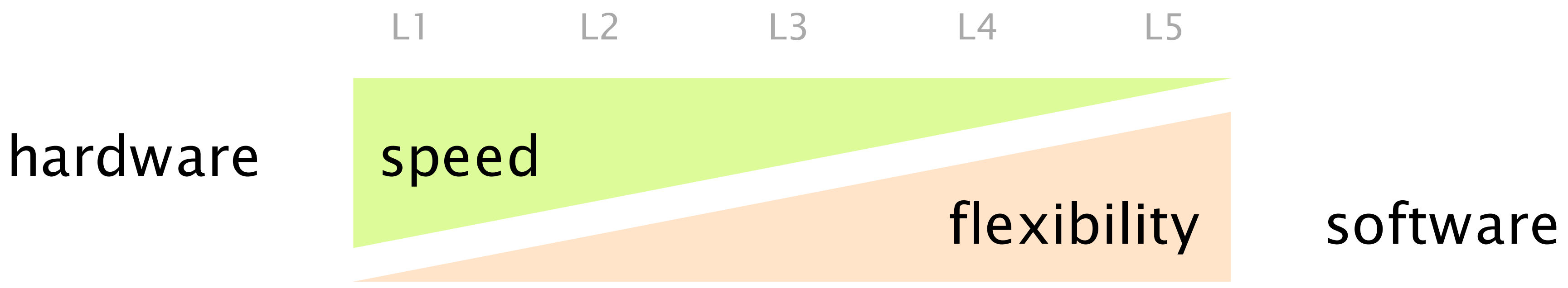
	layer	role
L5	Application	exchanges messages between processes
L4	Transport	transports segments between end-systems
L3	Network	moves packets around the network
L2	Link	moves frames across a link
L1	Physical	moves bits across a physical medium

Each layer is implemented with different protocols
and technologies

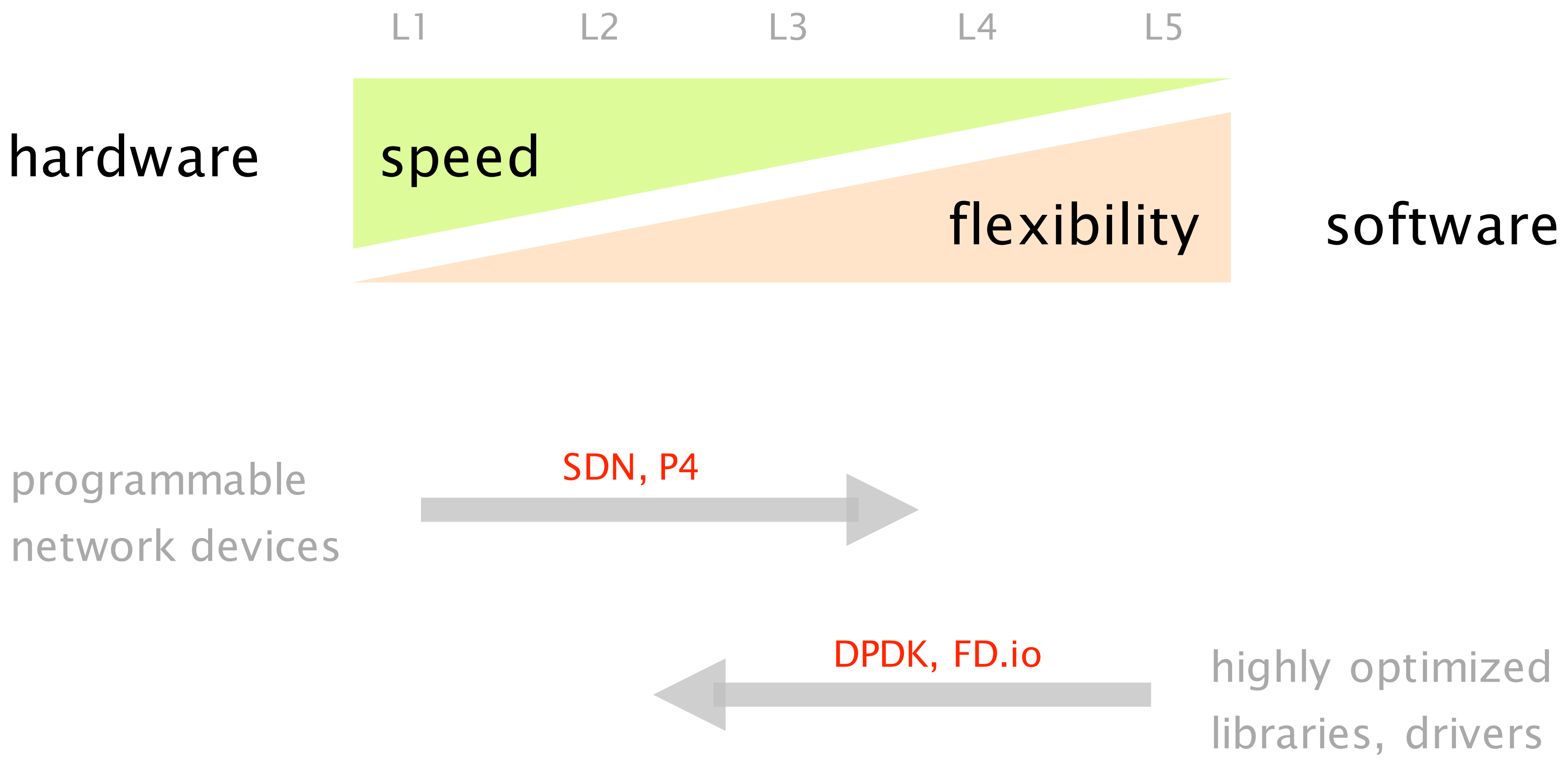
	layer
L5	Application
L4	Transport
L3	Network
L2	Link
L1	Physical

technology




L1
L2
L3
L4
L5
hardware
speed
flexibility
software

Software and hardware advancements



Microsoft Supercharges Bing Search With Programmable Chips

Secure https://www.wired.com/2014/06/microsoft-fpga/

WIRED — Microsoft Supercharges Bing Search With Programmable Chips — SUBSCRIBE

BUSINESS | CULTURE | DESIGN | GEAR | SCIENCE | SECURITY | TRANSPORTATION

SHARE

SHARE 1123

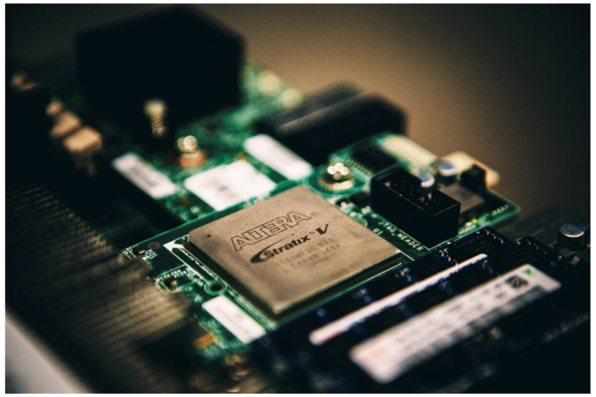
TWEET

COMMENT 103

EMAIL

ROBERT MCMILLAN BUSINESS 06.16.14 6:30 AM

MICROSOFT SUPERCHARGES BING SEARCH WITH PROGRAMMABLE CHIPS



Microsoft

DOUG BURGER CALLED it Project Catapult.

Burger works inside Microsoft Research–the group where the tech giant explores blue-sky ideas–and in November 2012, he pitched a radical new concept to Qi Lu, the man who oversees Microsoft's Bing web search engine. He wanted to

MOST POPULAR

MOBILE
Android Can't Compete With iMessage. Google Is Changing That
DAVID PIERCE

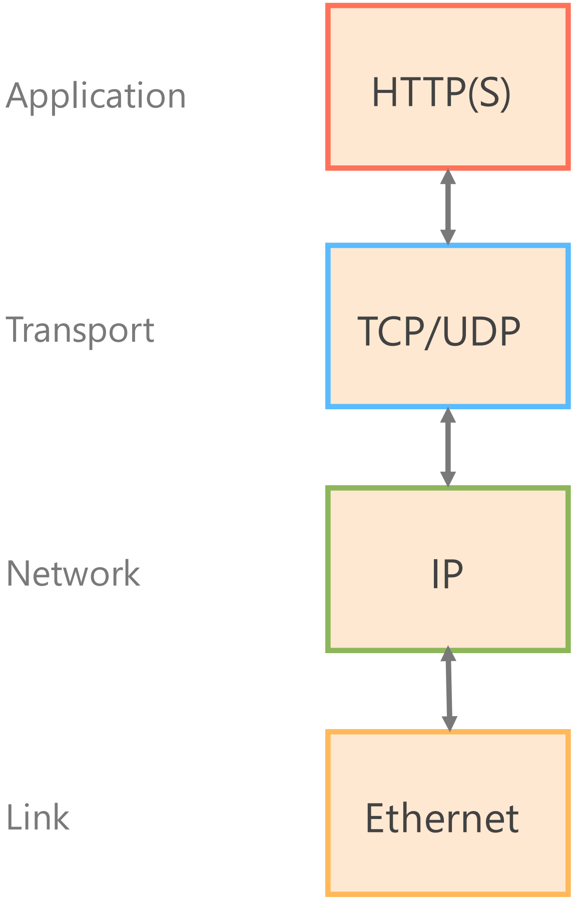
SCANDALS
Google Accuses Uber of Stealing Its Self-Driving Car Tech
ALEX DAVIES

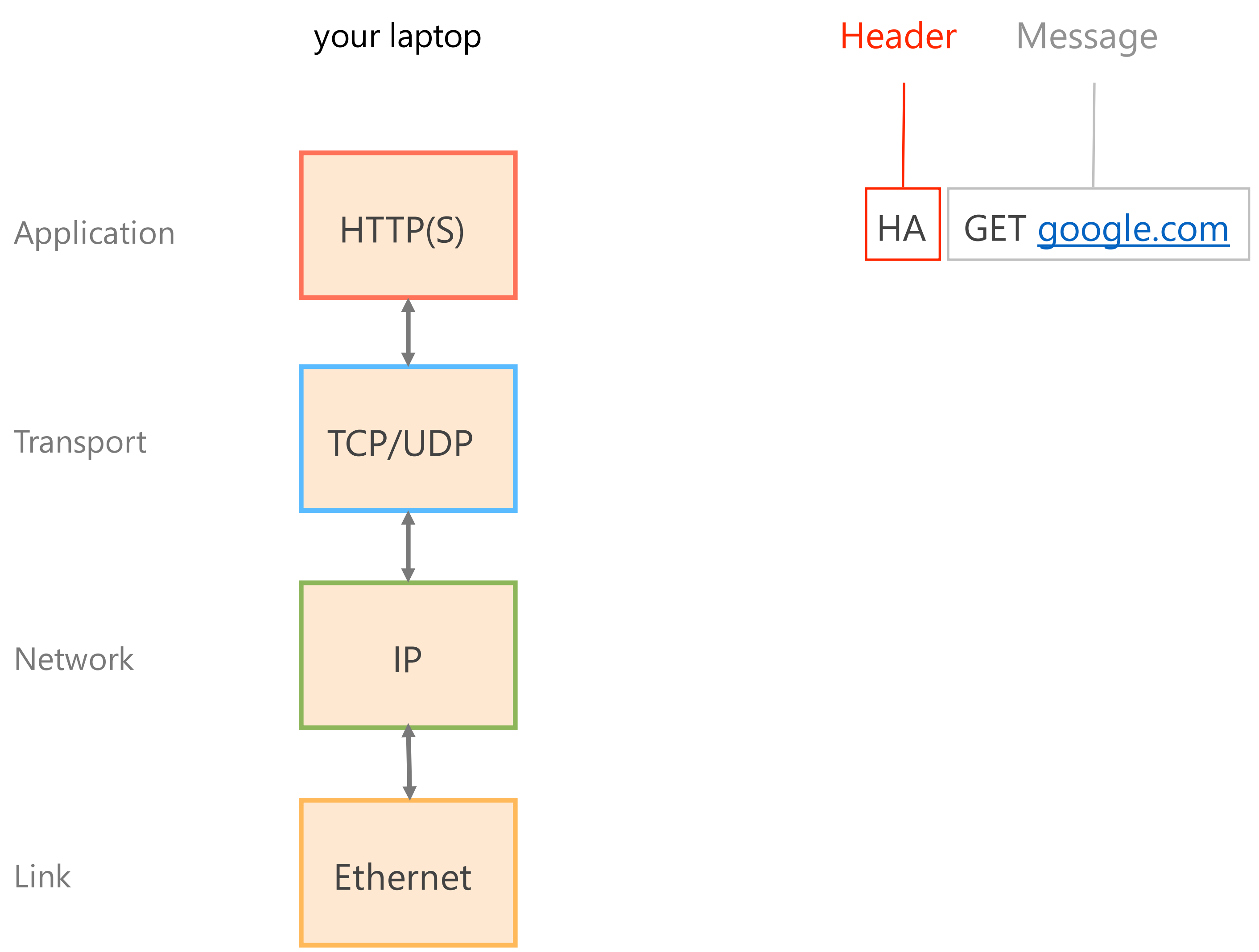
PRODUCT REVIEW
Review: Microsoft Surface Studio
DAVID PIERCE

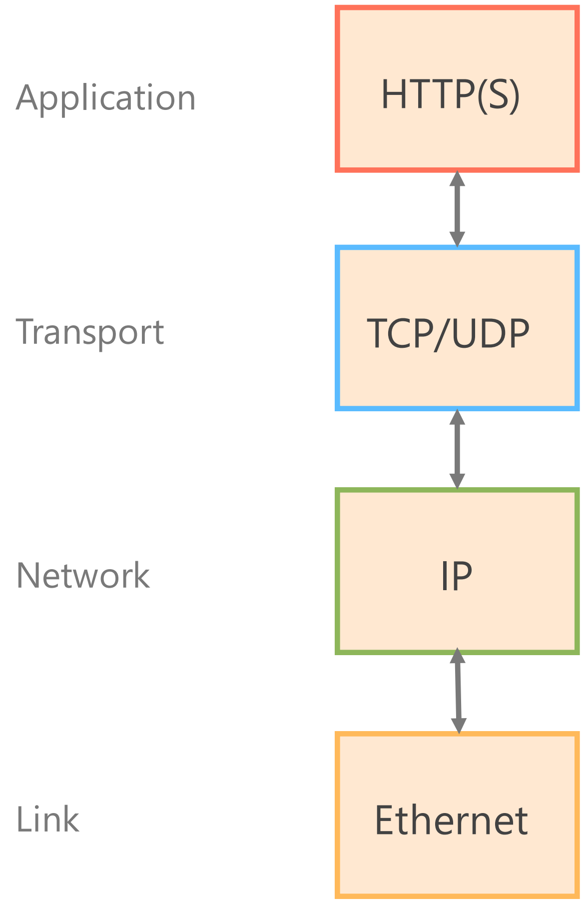
MORE STORIES

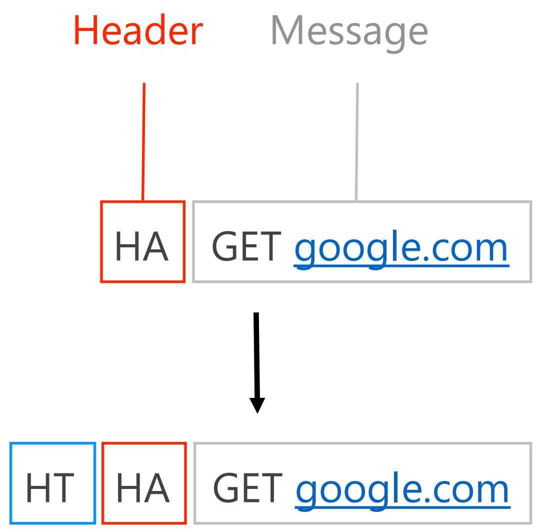
https://www.wired.com/2014/06/microsoft-fpga/

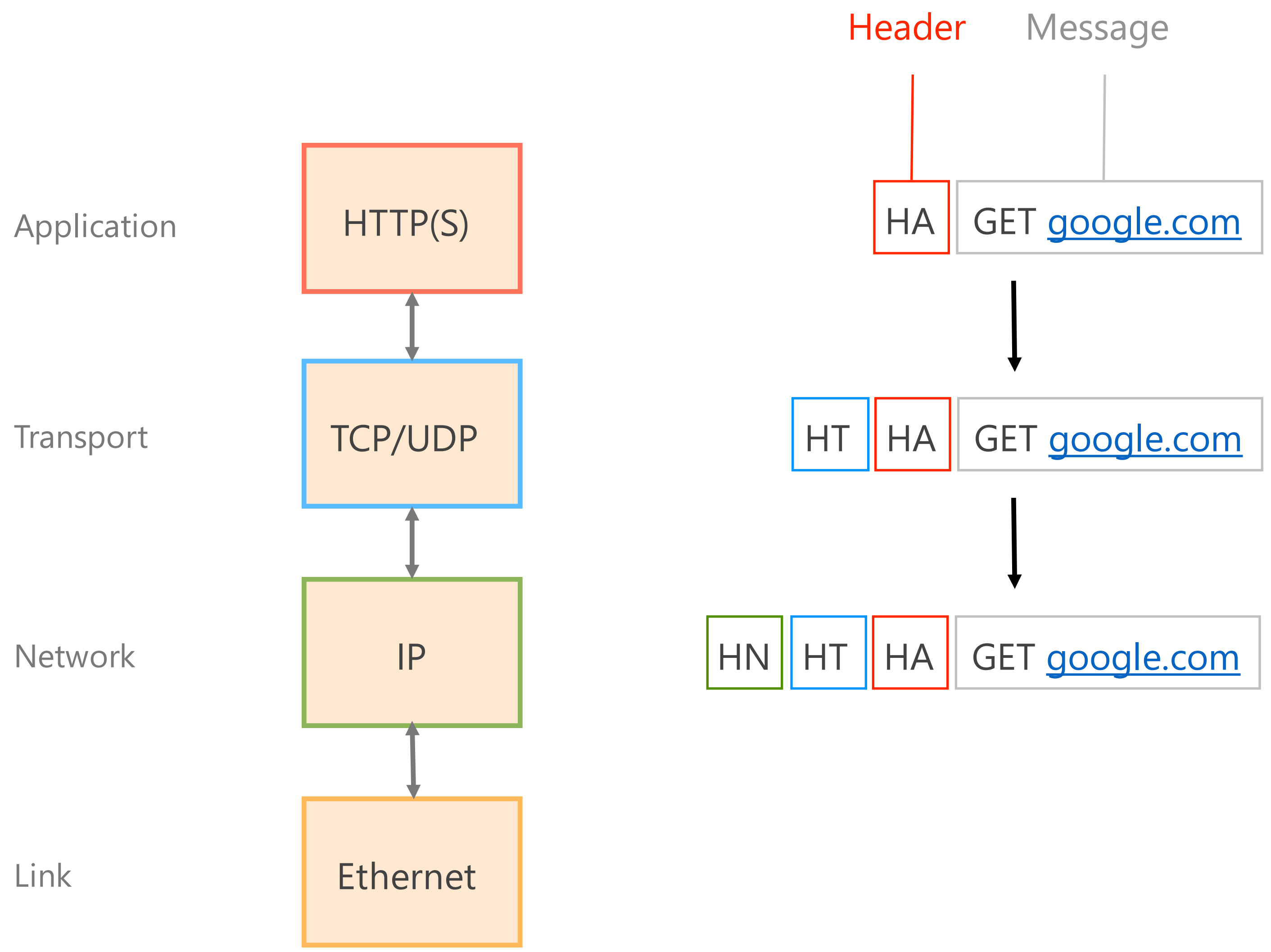
Each layer takes messages from the layer above, and *encapsulates* with its own header and/or trailer

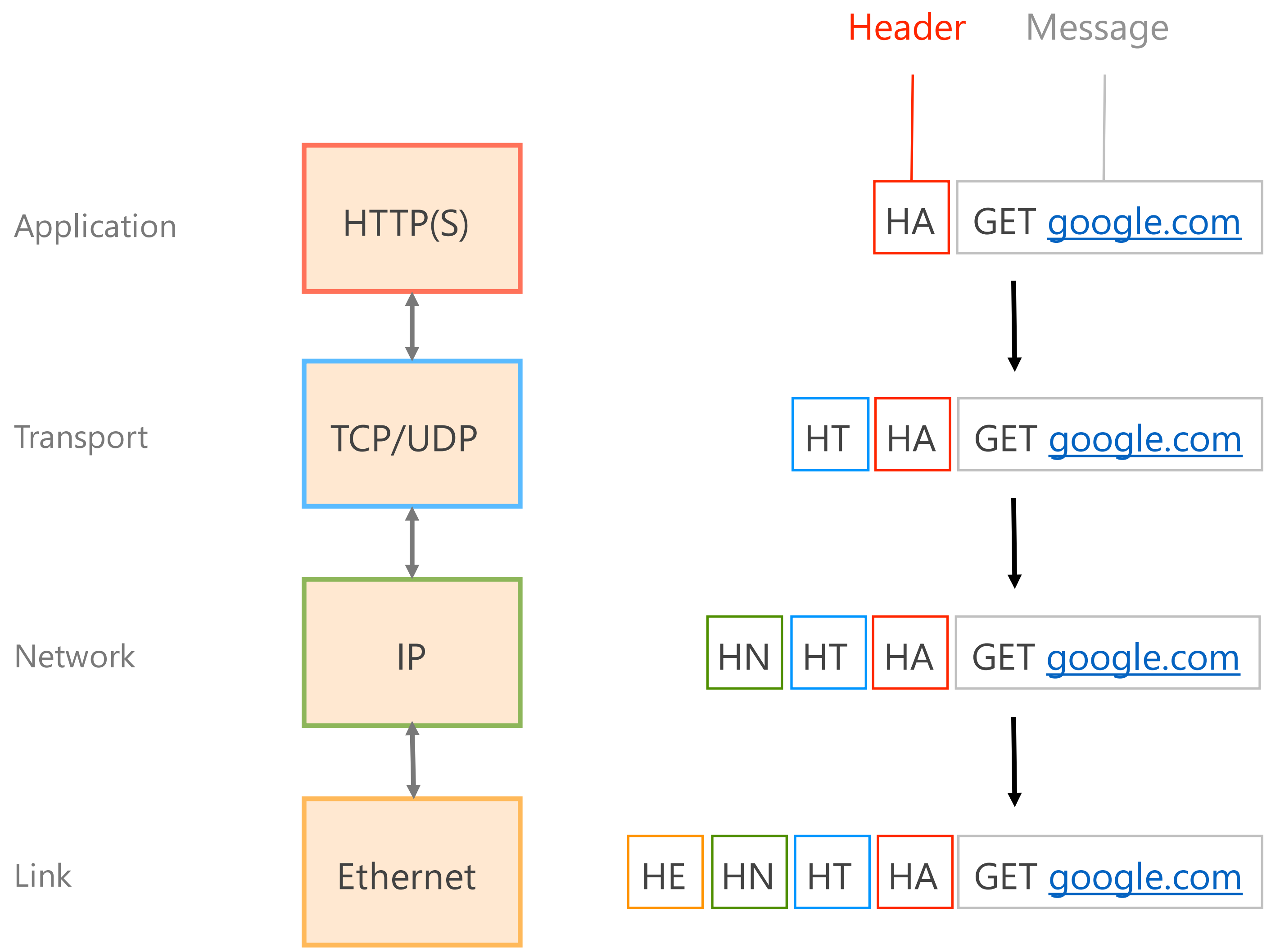



your laptop
Application
Transport
Network
Link
HTTP(S)
TCP/UDP
IP
Ethernet
Header
Message
HA
GET google.com

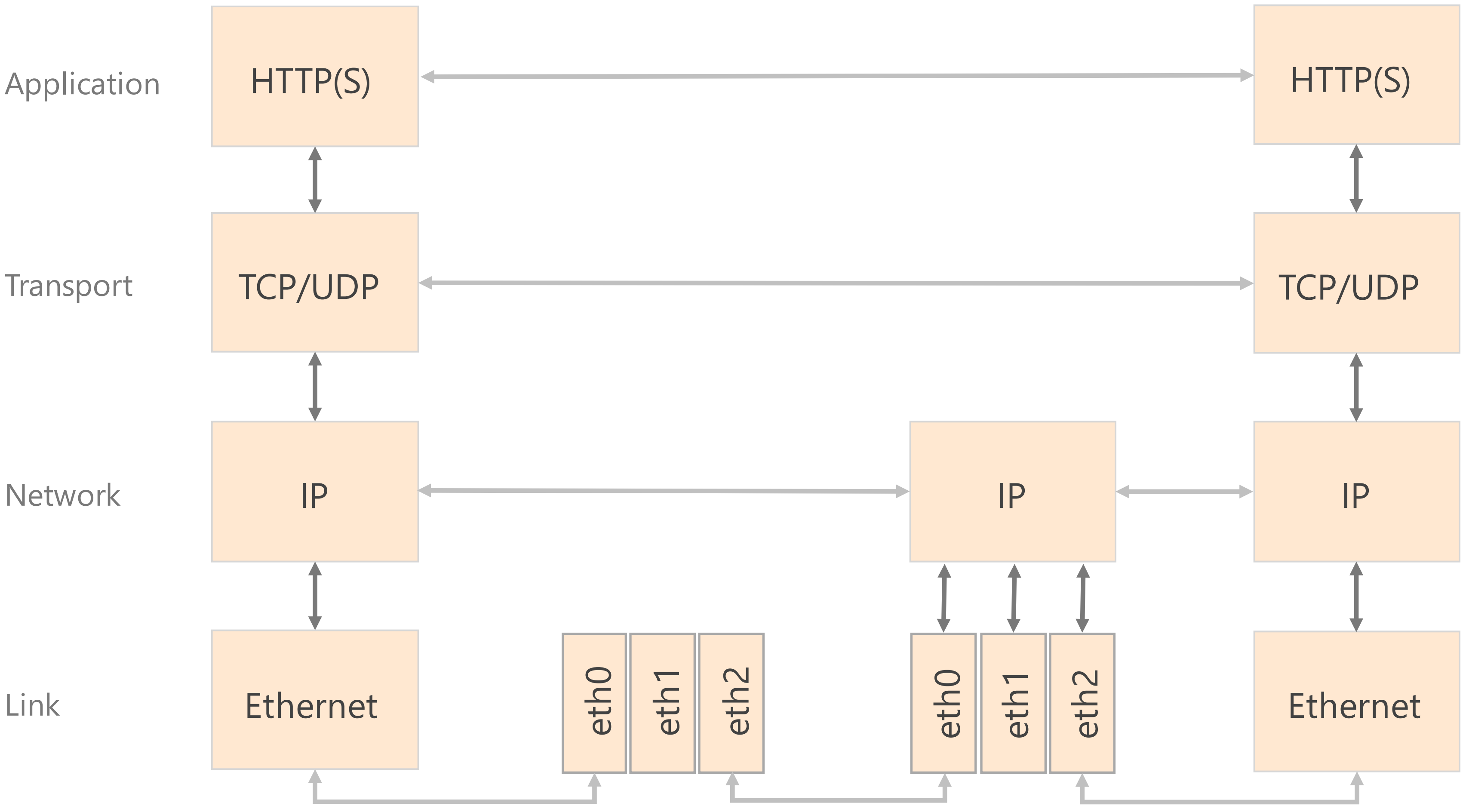

Application
HTTP(S)
Transport
TCP/UDP
Network
IP
Link
Ethernet


Header
Message
HA
GET google.com
HT
HA
GET google.com

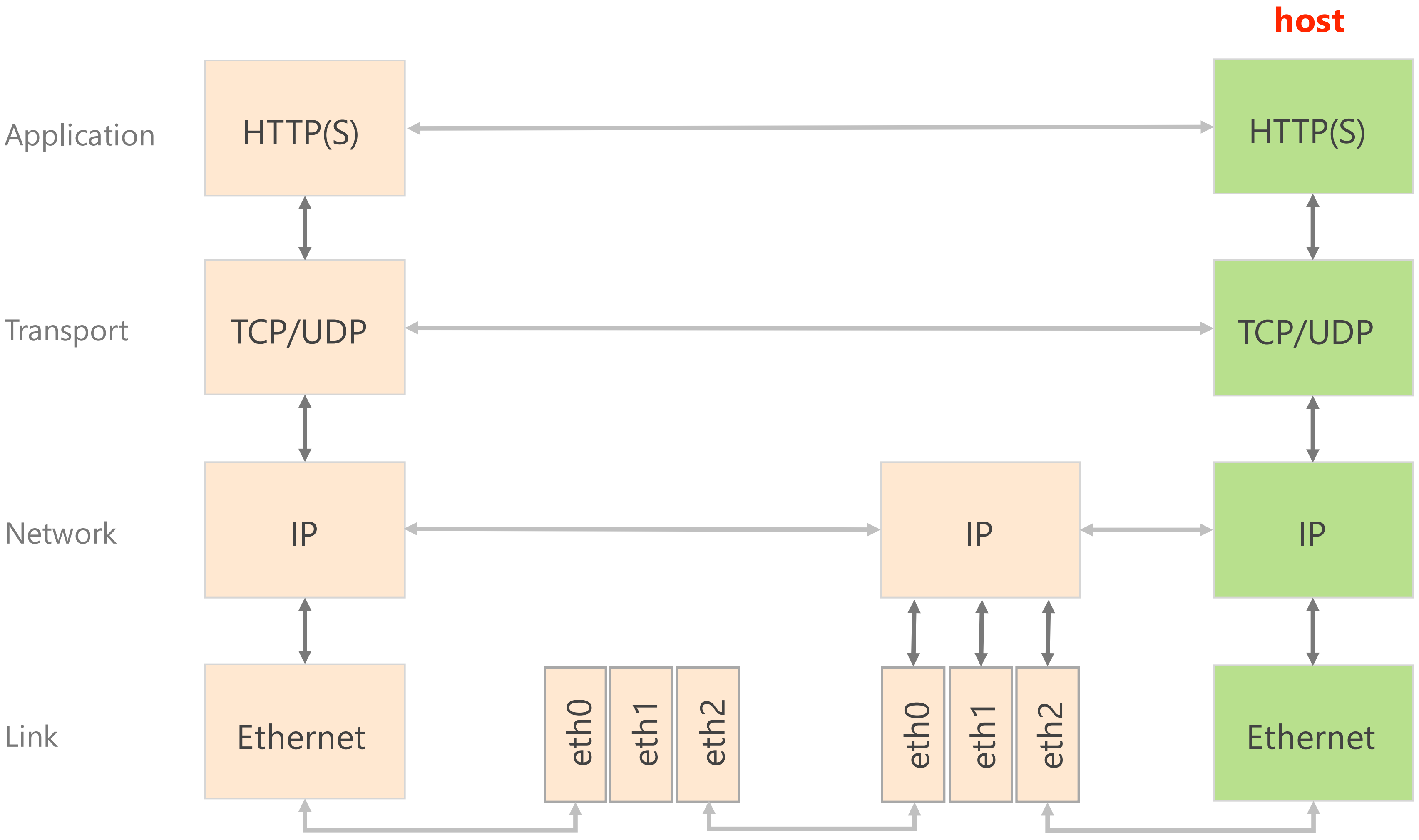

Application
Transport
Network
Link
HTTP(S)
TCP/UDP
IP
Ethernet
Header
Message
HA
GET google.com
HT
HA
GET google.com
HN
HT
HA
GET google.com


Header
Message
Application
HTTP(S)
HA
GET google.com
Transport
TCP/UDP
HT
HA
GET google.com
Network
IP
HN
HT
HA
GET google.com
Link
Ethernet
HE
HN
HT
HA
GET google.com

In practice, layers are distributed on every network device

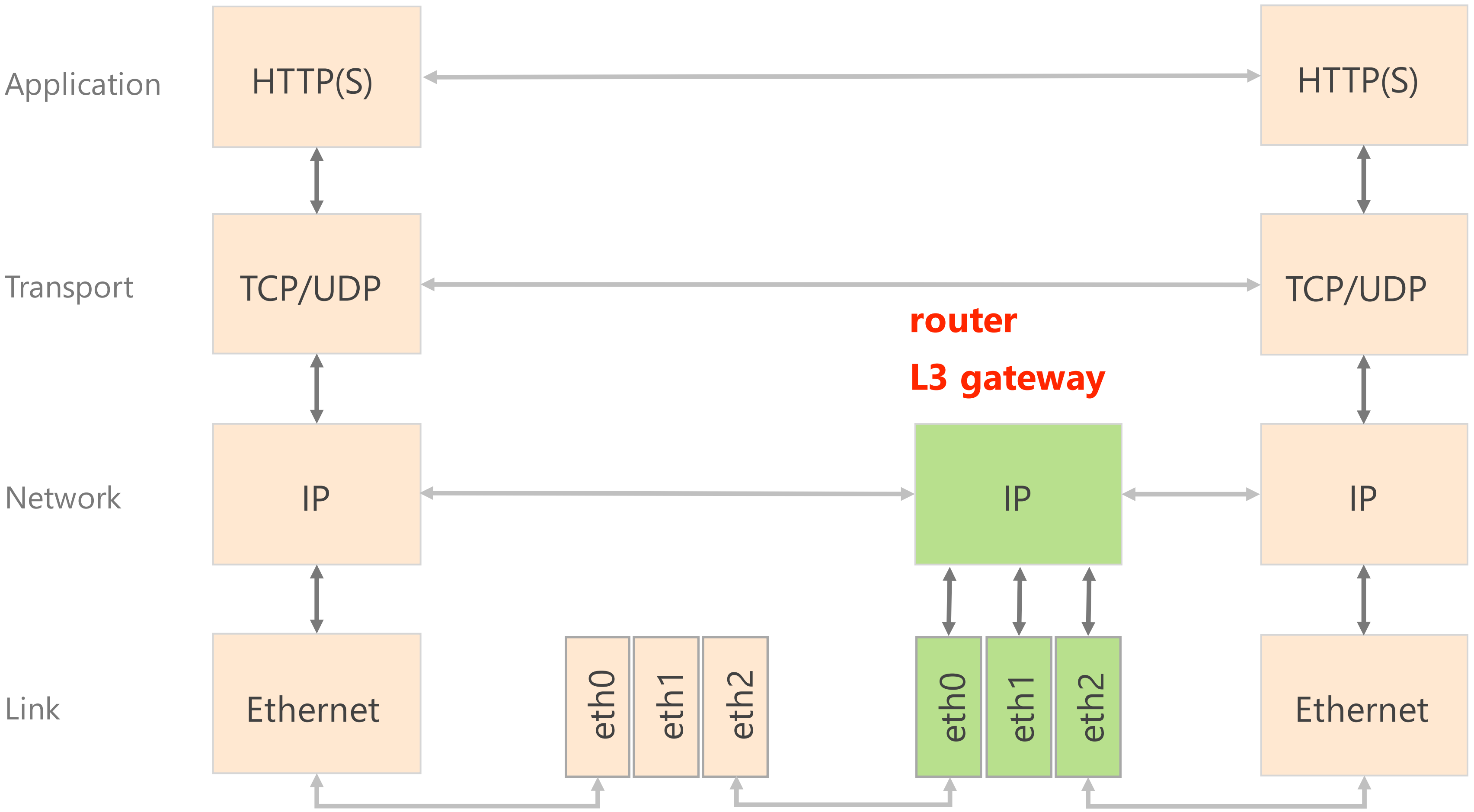


Since when bits arrive they must make it to the application, all the layers exist on a host

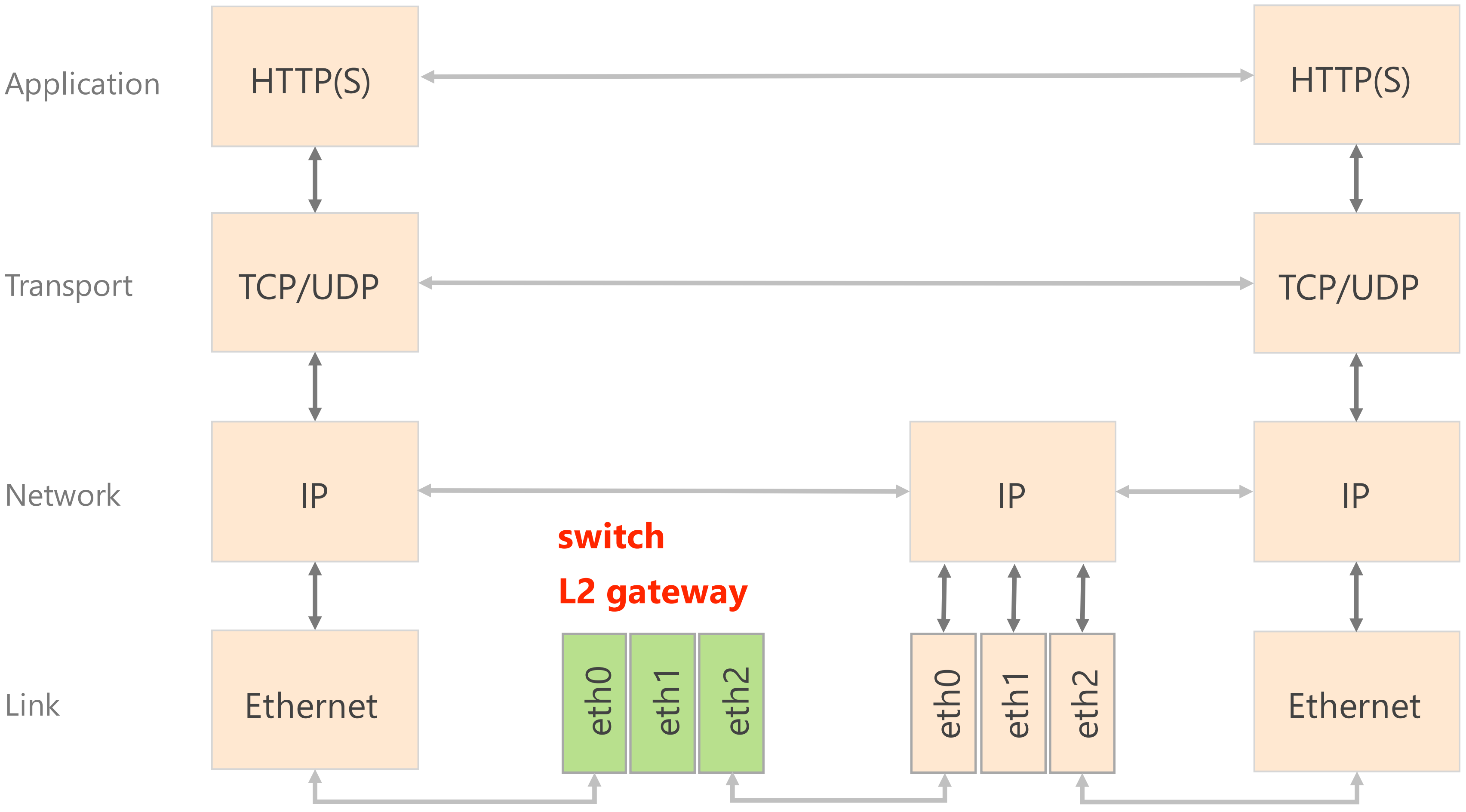


Routers act as L3 gateway

as such they implement L2 and L3



Switches act as L2 gateway
as such they only implement L2



Let’s see how it looks like in practice

on a host, using Wireshark https://www.wireshark.org

Communication Networks and Internet Technology

Part 1: General overview

What is a network made of?

How is it shared?

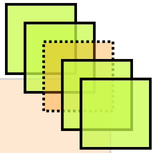
How is it organized?

How does communication happen?

#5 How do we characterize it?

A network *connection* is characterized by its delay, loss rate and throughput

delay



loss

throughput

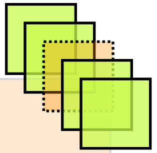
How long does it take for a packet to reach the destination

What fraction of packets sent to a destination are dropped?

At what rate is the destination receiving data from the source?

A network *connection* is characterized by its delay, loss rate and throughput

delay



loss

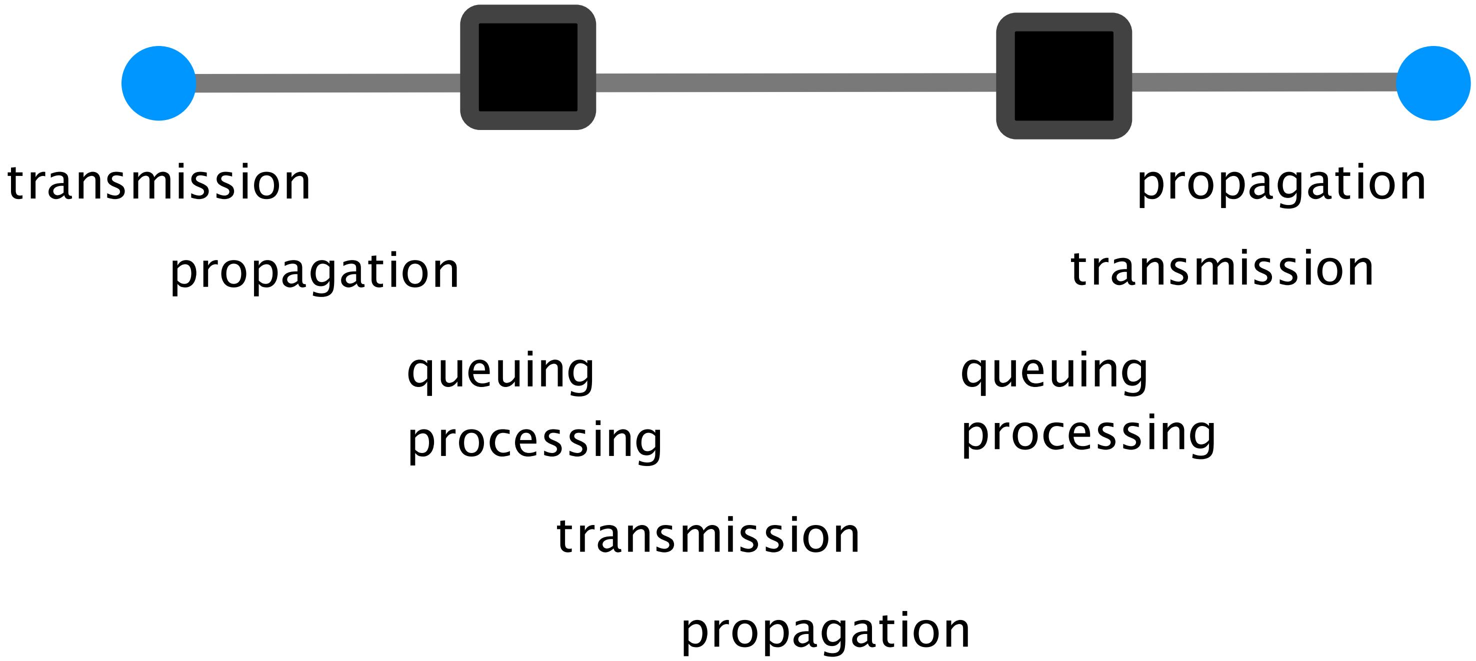
throughput

Each packet suffers from several types of delays at *each node* along the path

	transmission	delay	due to link properties
⊕	propagation	delay	
⊕	processing	delay	due to traffic mix & switch internals
⊕	queuing	delay	
=	total	delay	

Overall, the main culprits for the overall delay are the transmission, propagation and queuing delays

	transmission	delay	
$\oplus$	propagation	delay	
$\oplus$	processing	delay	*tend to be tiny*
$\oplus$	queuing	delay	
$=$	total	delay	


transmission
propagation
queuing
processing
transmission
propagation
queuing
processing
transmission
propagation

The transmission delay is the amount of time required to push all of the bits onto the link

$$\underset{[\text{sec}]}{\text{Transmission delay}} = \frac{\text{packet size}}{\text{link bandwidth}} \quad \frac{[\#\text{bits}]}{[\#\text{bits/sec}]}$$

Example

$$\frac{1000 \text{ bits}}{100 \text{ Gbps}} \quad 10 \text{ ns}$$

The propagation delay is the amount of time required for a bit to travel to the end of the link

$$\text{Propagation delay [sec]} = \frac{\text{link length [m]}}{\text{propagation speed [m/sec]}}$$

(fraction of speed of light)

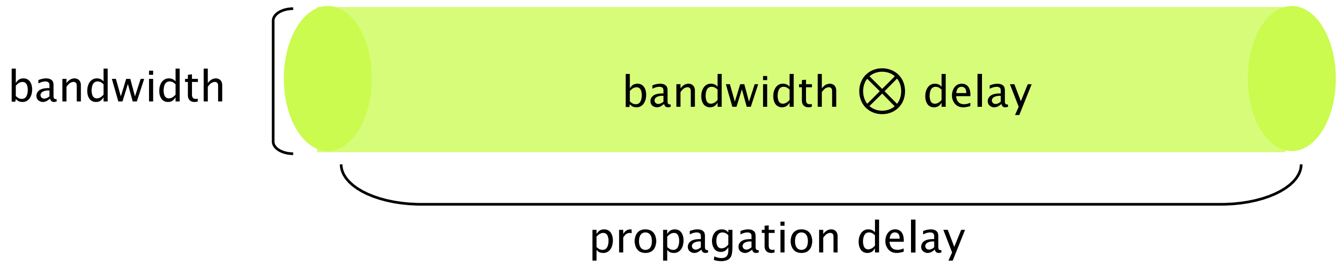
Example

$$\frac{30\,000 \text{ m}}{2\text{x}10^8 \text{ m/sec}} \quad 150\ \mu\text{sec}$$

(speed of light in fiber)

A network *link* is characterized by its bandwidth and propagation delay

link bandwidth	# bits sent per sec
propagation delay	time for 1 bit to move through the link (sec)
bandwidth ⊗ delay	# of bits in-flight at any time



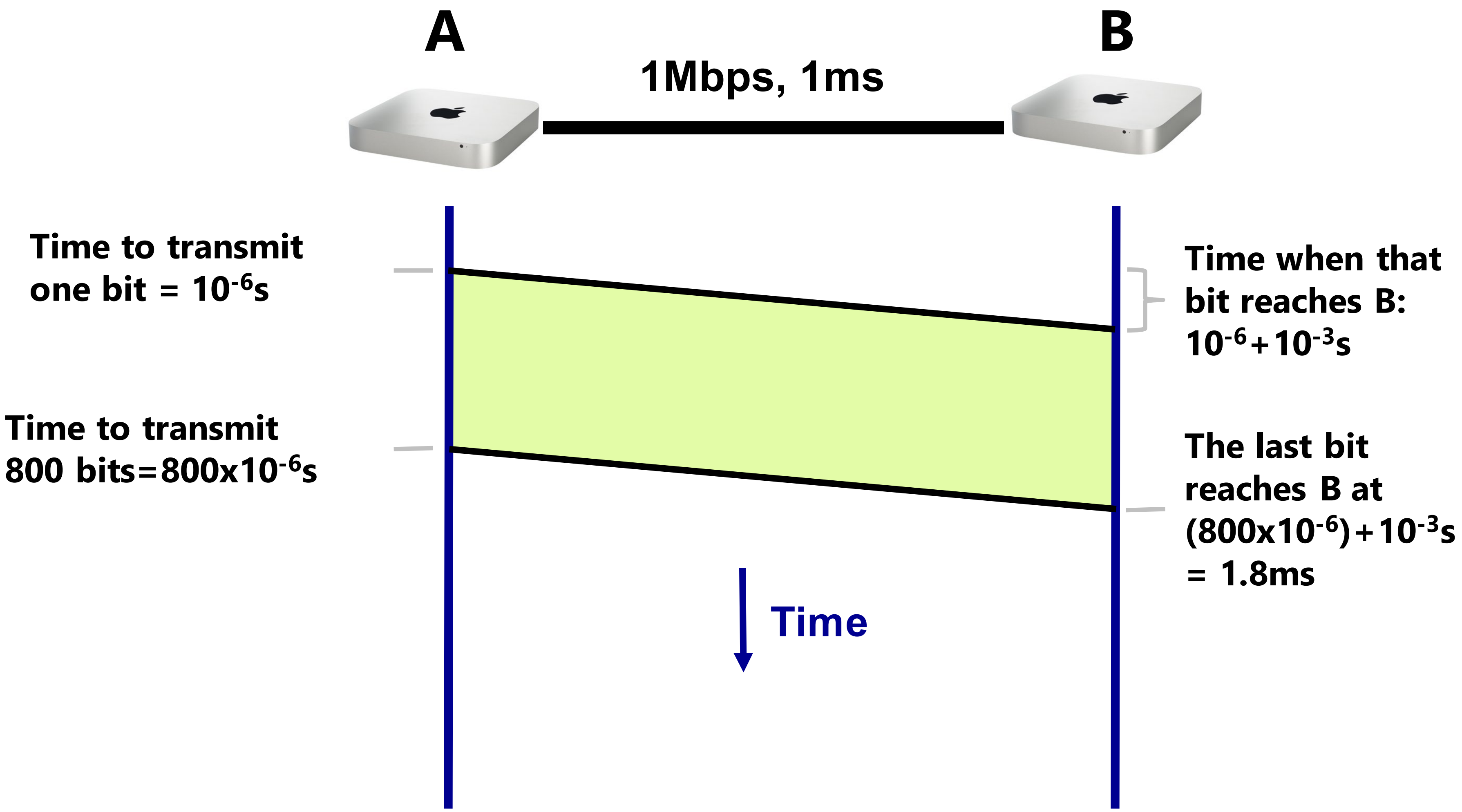
In practice, BDP can be huge

same city, over slow link		cross continent, over fast link	
bandwidth	100 Mbps	bandwidth	10 Gbps
delay	0.1 ms	delay	10 ms
BDP	10,000 bits	BDP	10^8 bits
	1.25 KBytes		12.5 MBytes

How long does it take for a packet to travel from A to B?

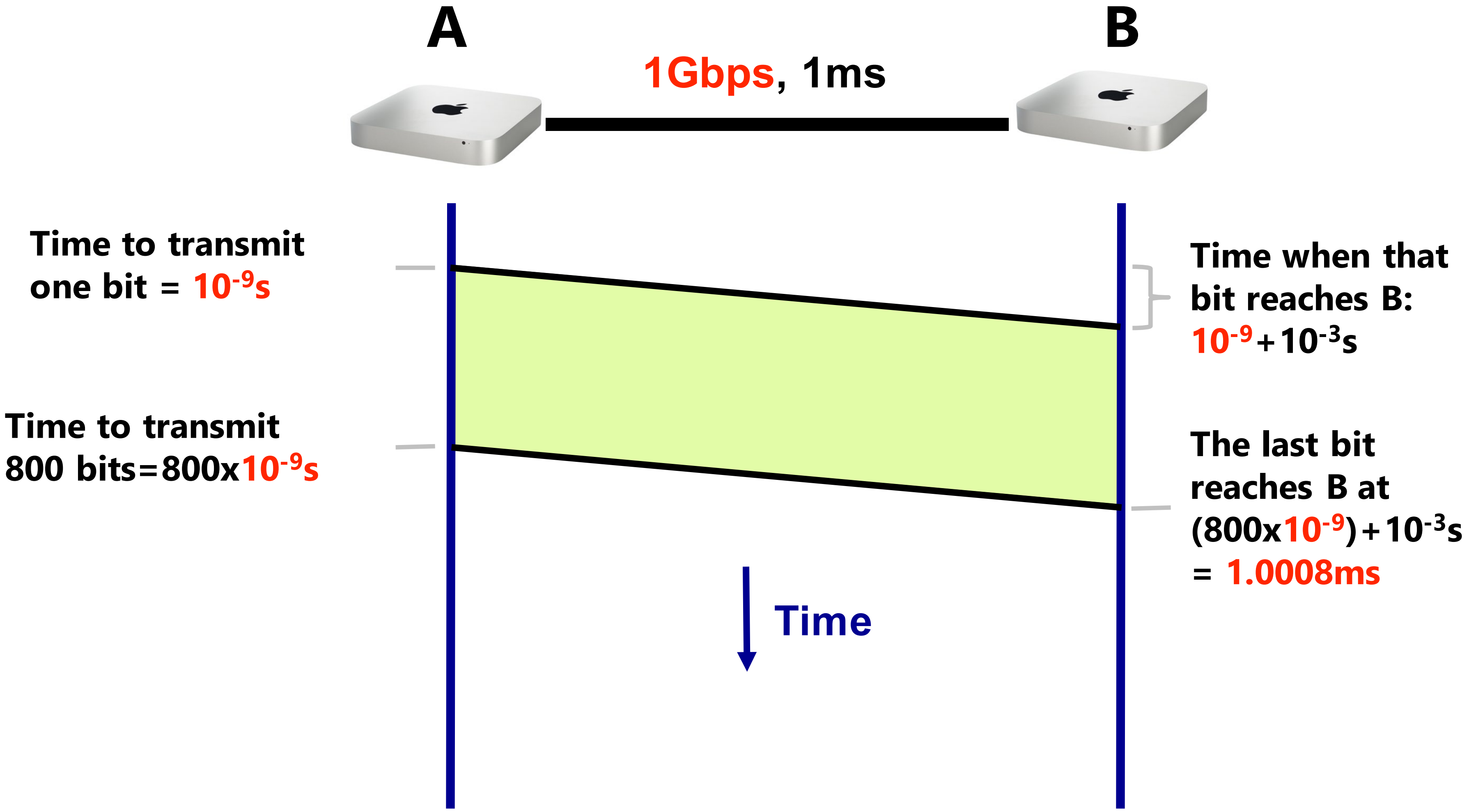
(not considering queuing for now)

How long does it take to exchange 100 Bytes packet?

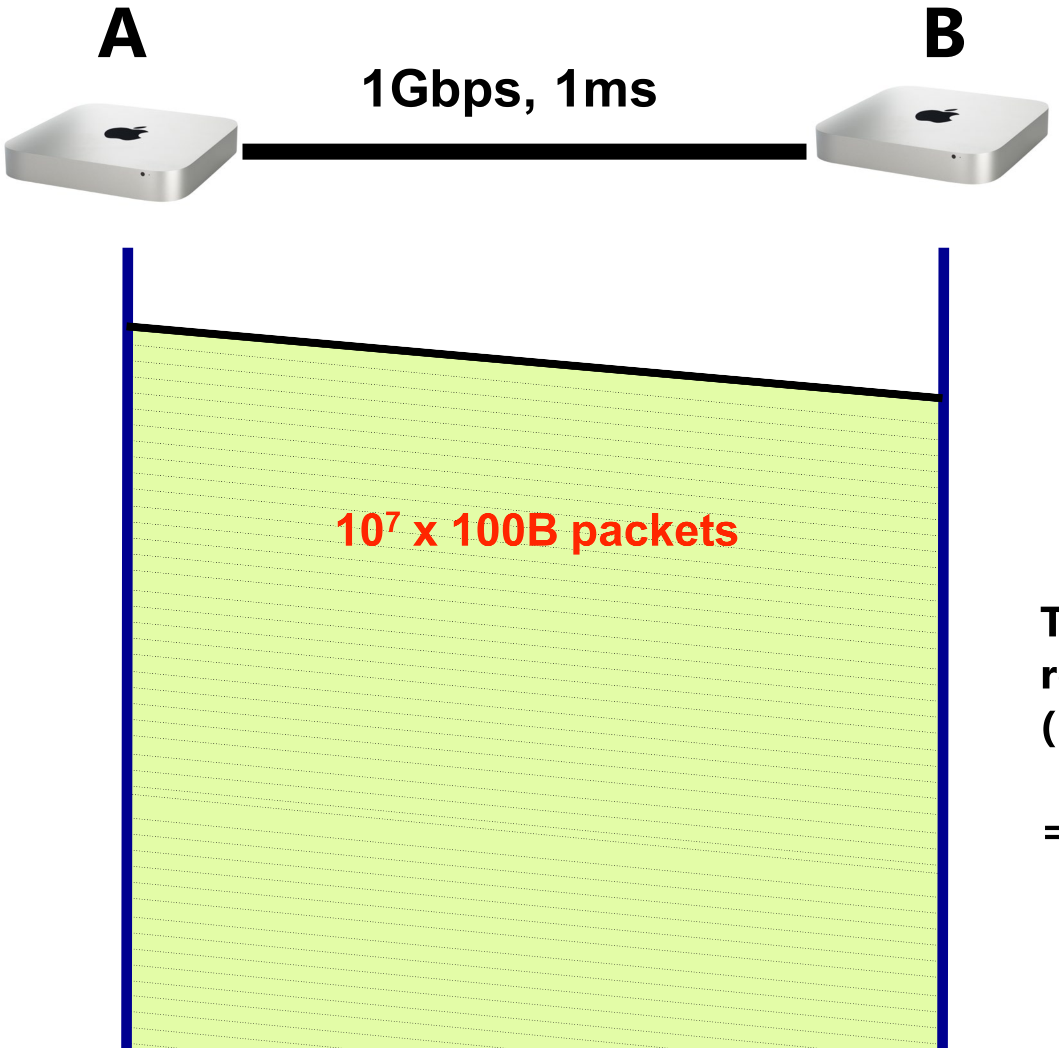


If we have a 1 Gbps link,

the total time decreases to 1.0008ms



If we now exchange a 1GB file
split in 100B packets



The last bit reaches B at
$(10^7 \times 800 \times 10^{-9}) + 10^{-3}$s
= 8001ms

Different transmission characteristics imply different tradeoffs in terms of which delay dominates

10^7x100B	pkt	1Gbps link	transmission delay dominates
1x100B	pkt	1Gbps link	propagation delay dominates
1x100B	pkt	1Mbps link	both matter

In the Internet, we **can't know** in advance which one matters!

The queuing delay is the amount of time a packet waits (in a buffer) to be transmitted on a link

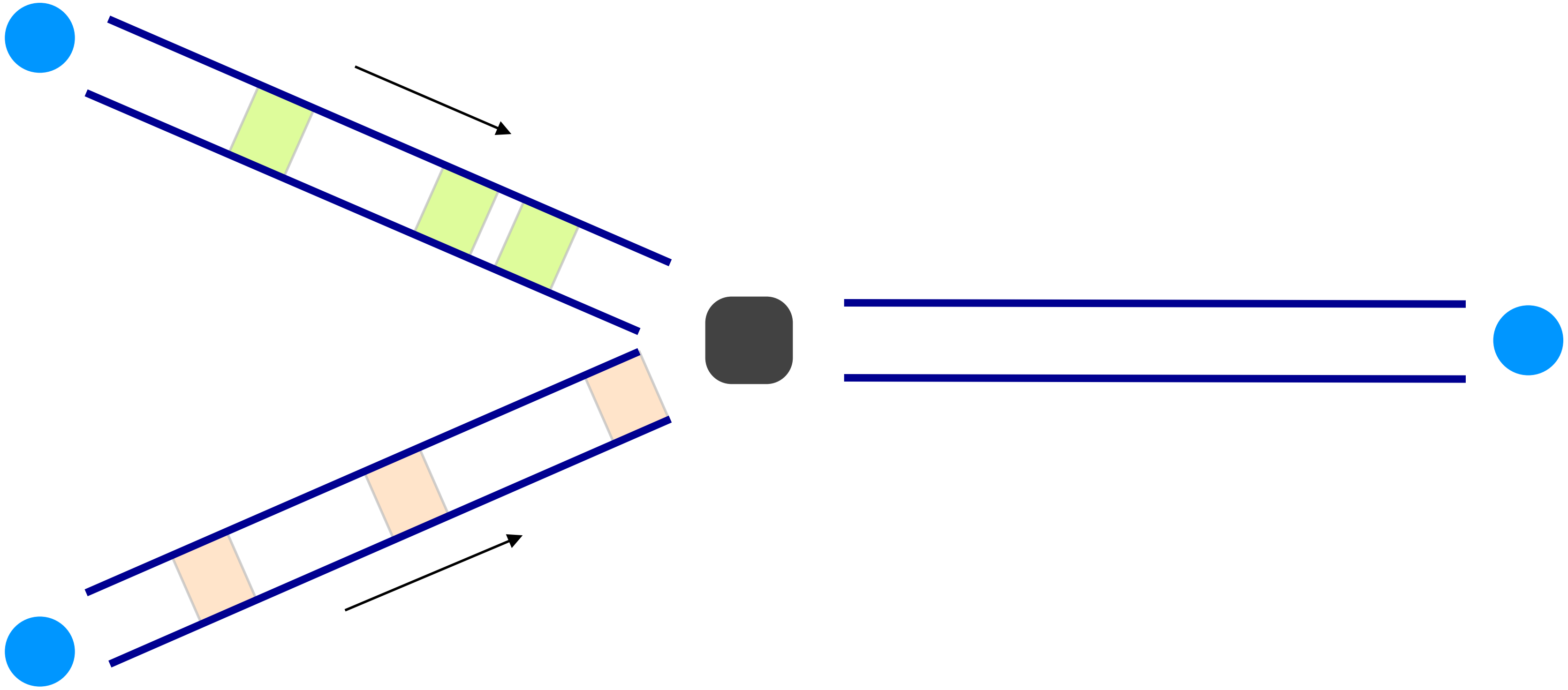
Queuing delay is the hardest to evaluate

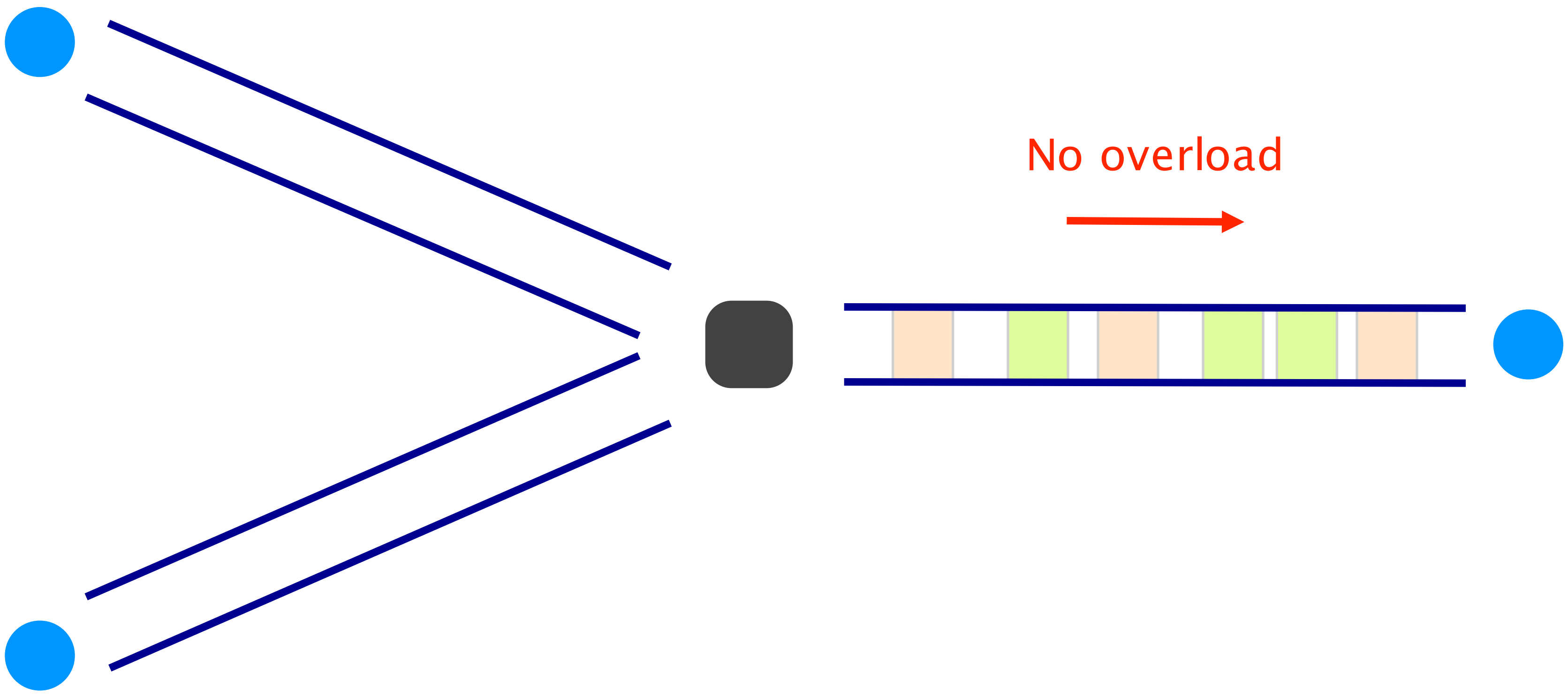
as it varies from packet to packet

It is characterized with statistical measures

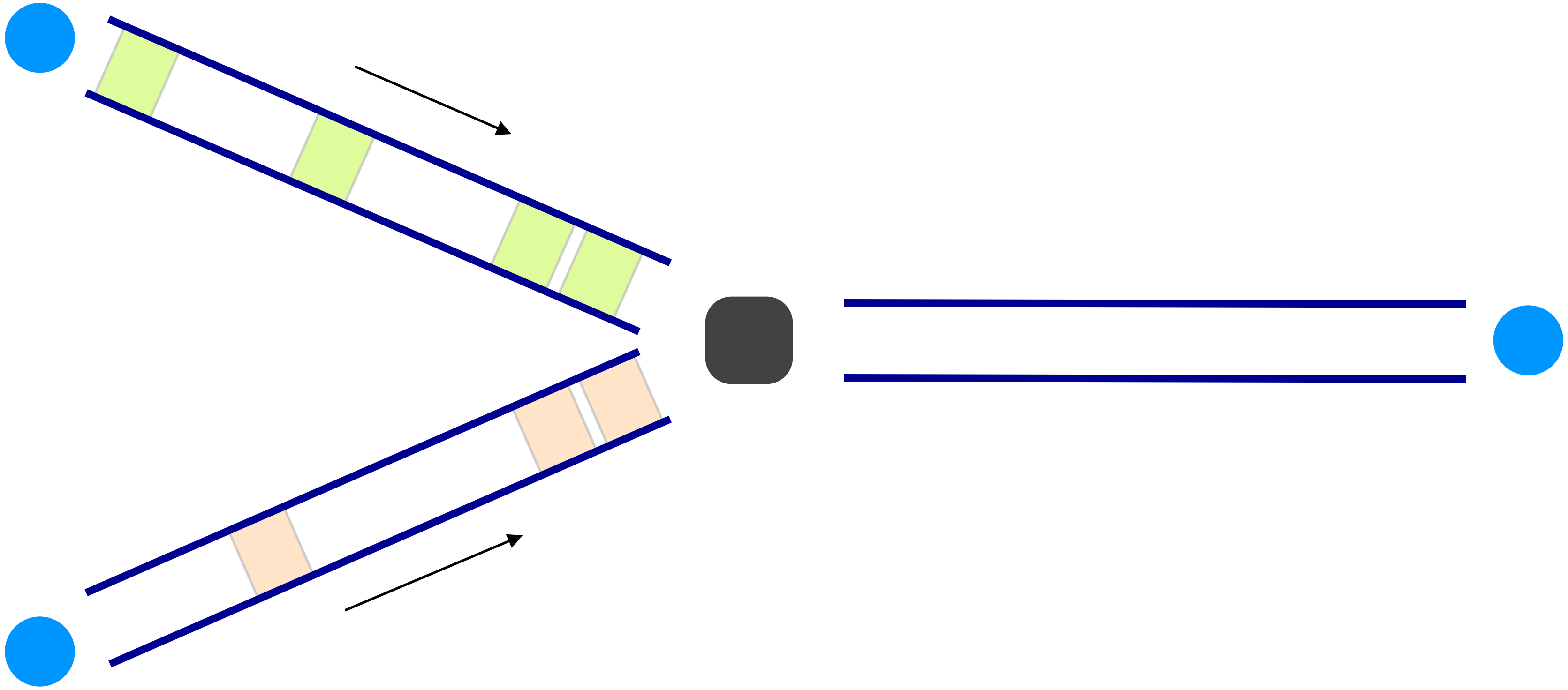
e.g., average delay & variance, probability of exceeding *x*

Queuing delay depends on the traffic pattern

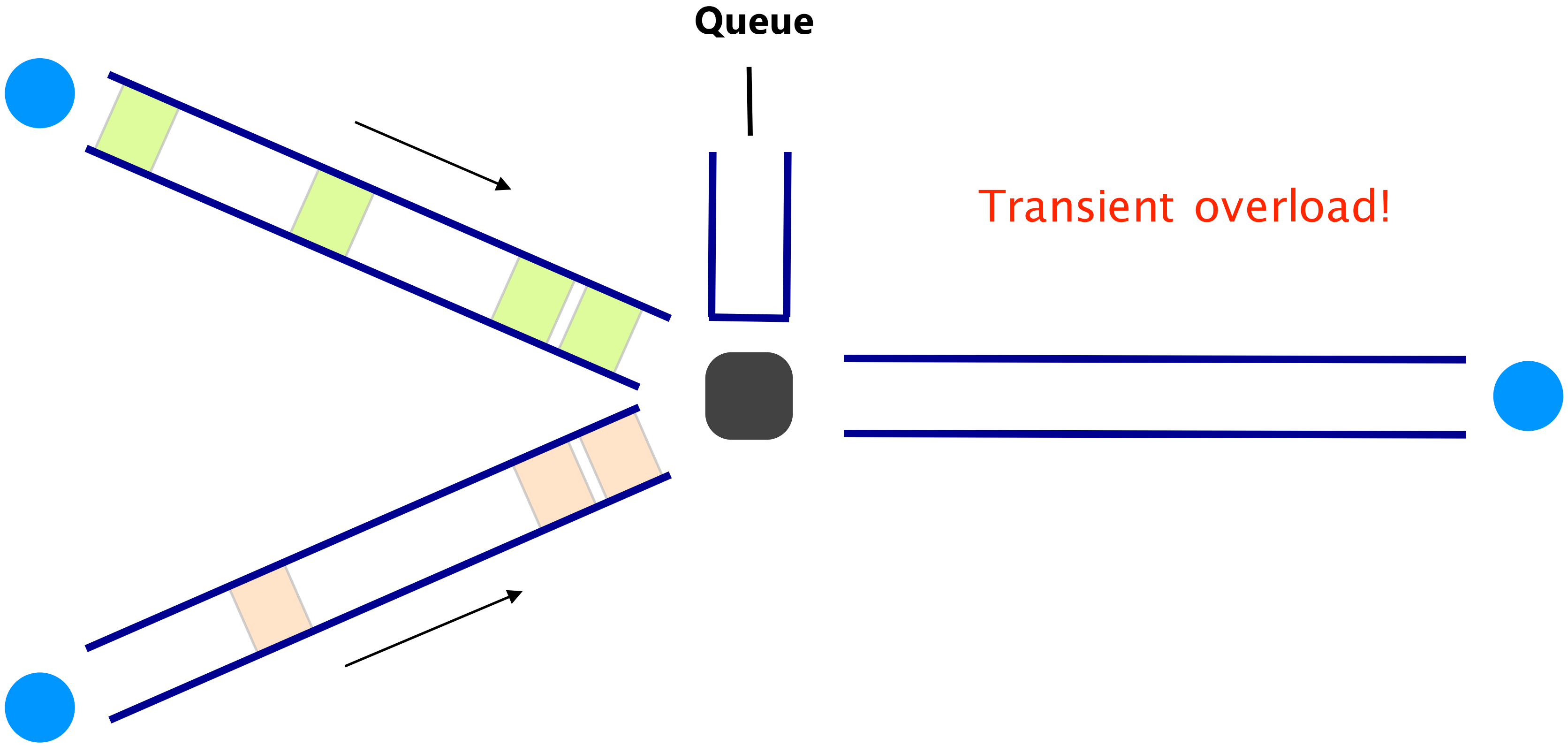


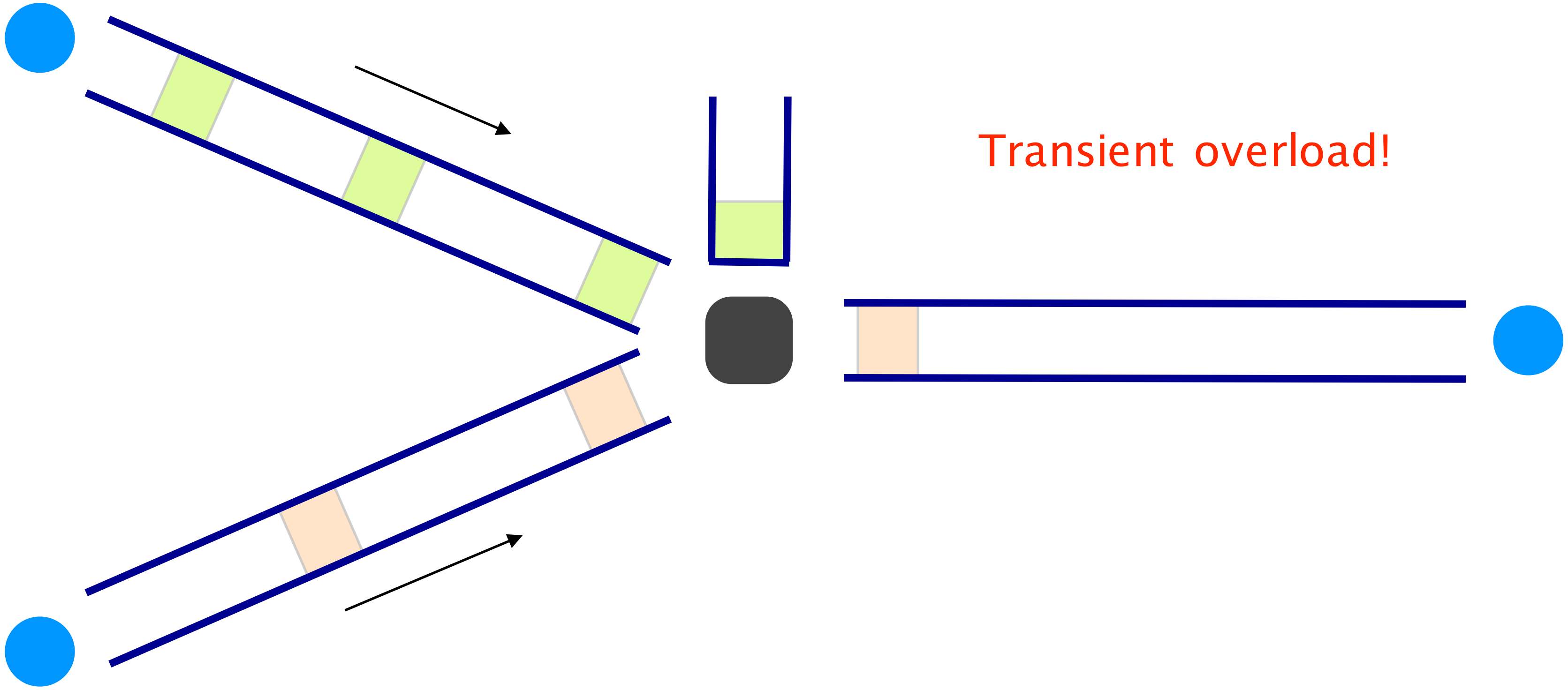

No overload

Queuing delay depends on the traffic pattern

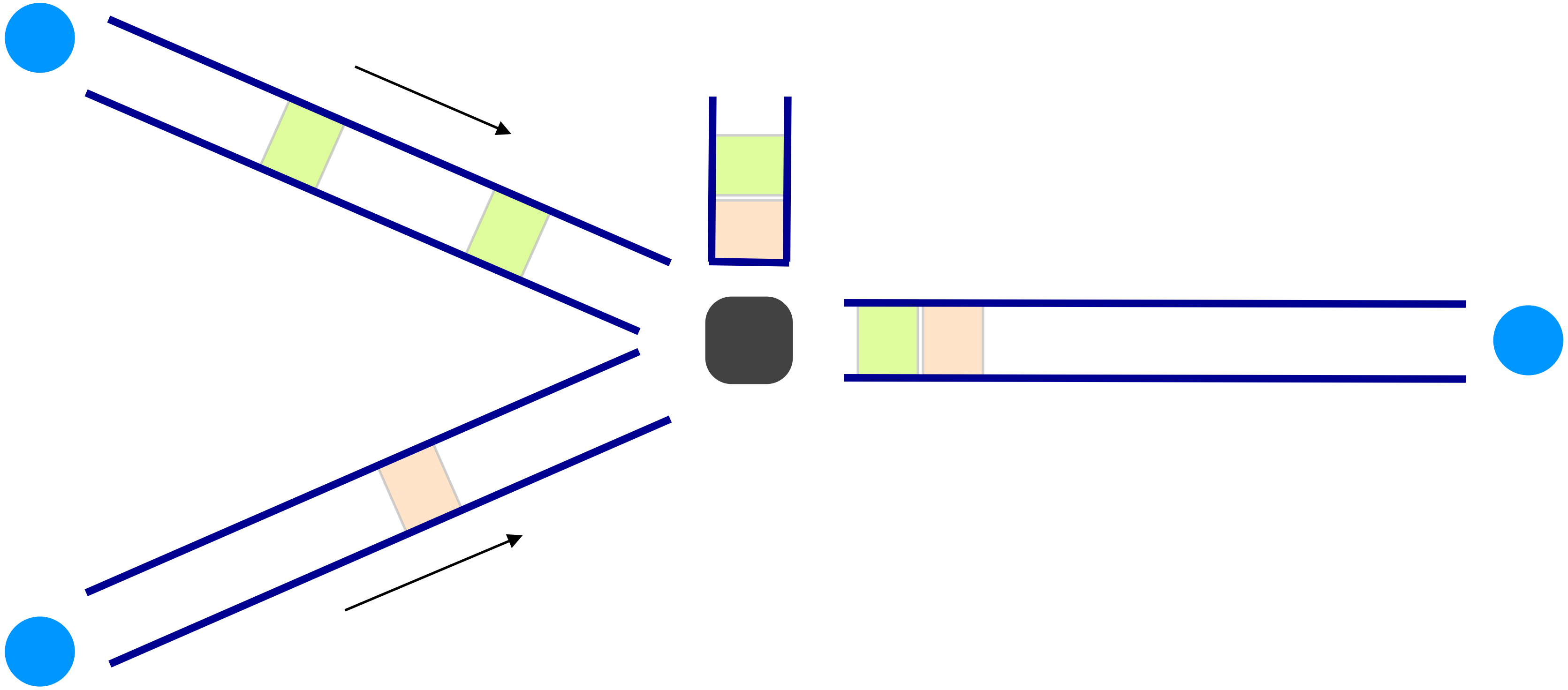


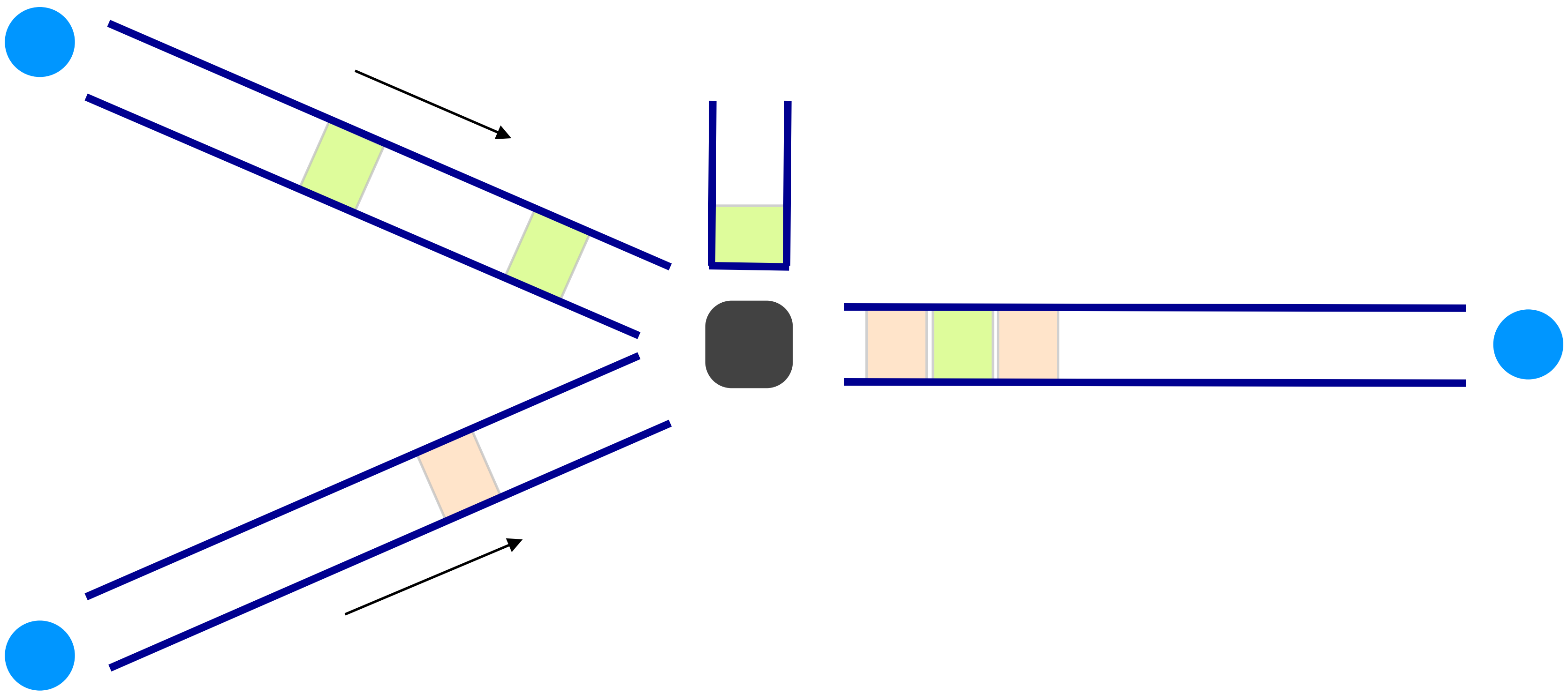
Queuing delay depends on the traffic pattern

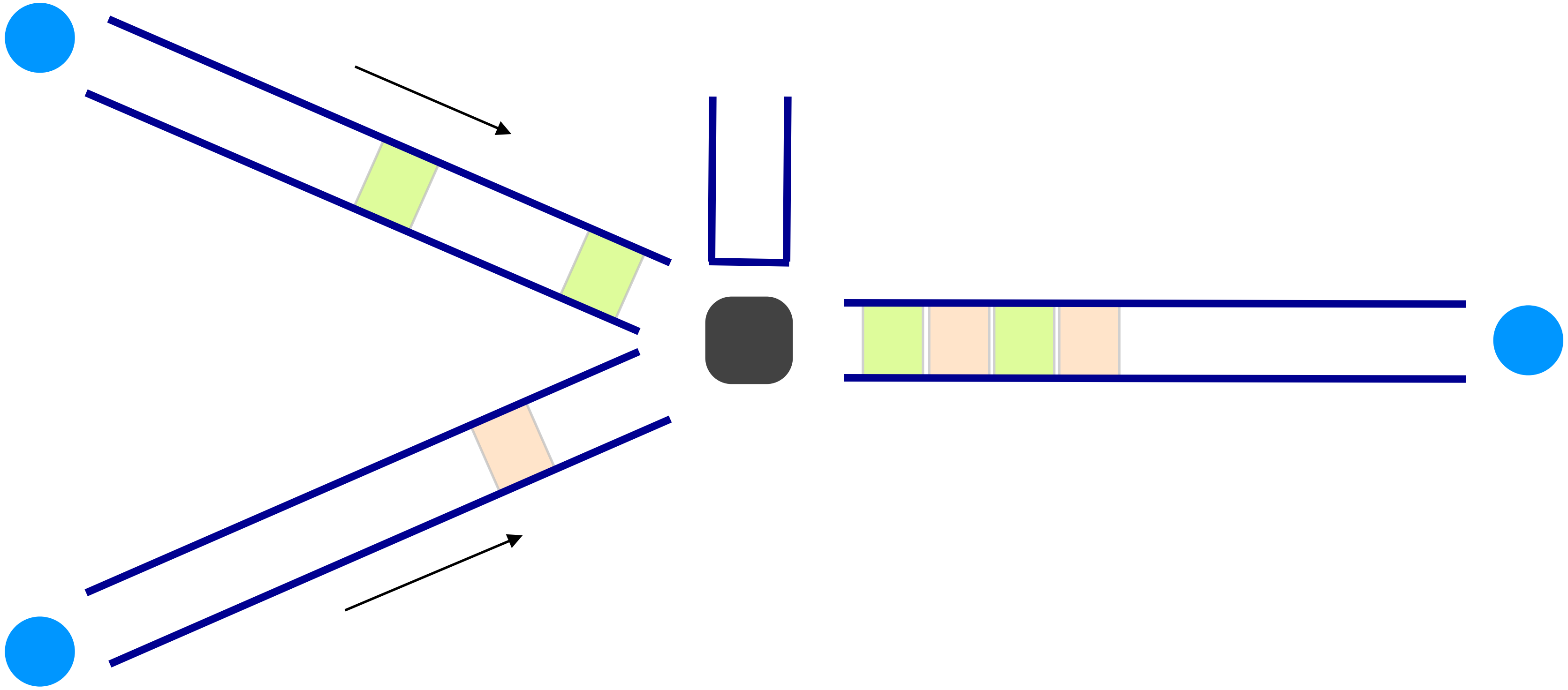




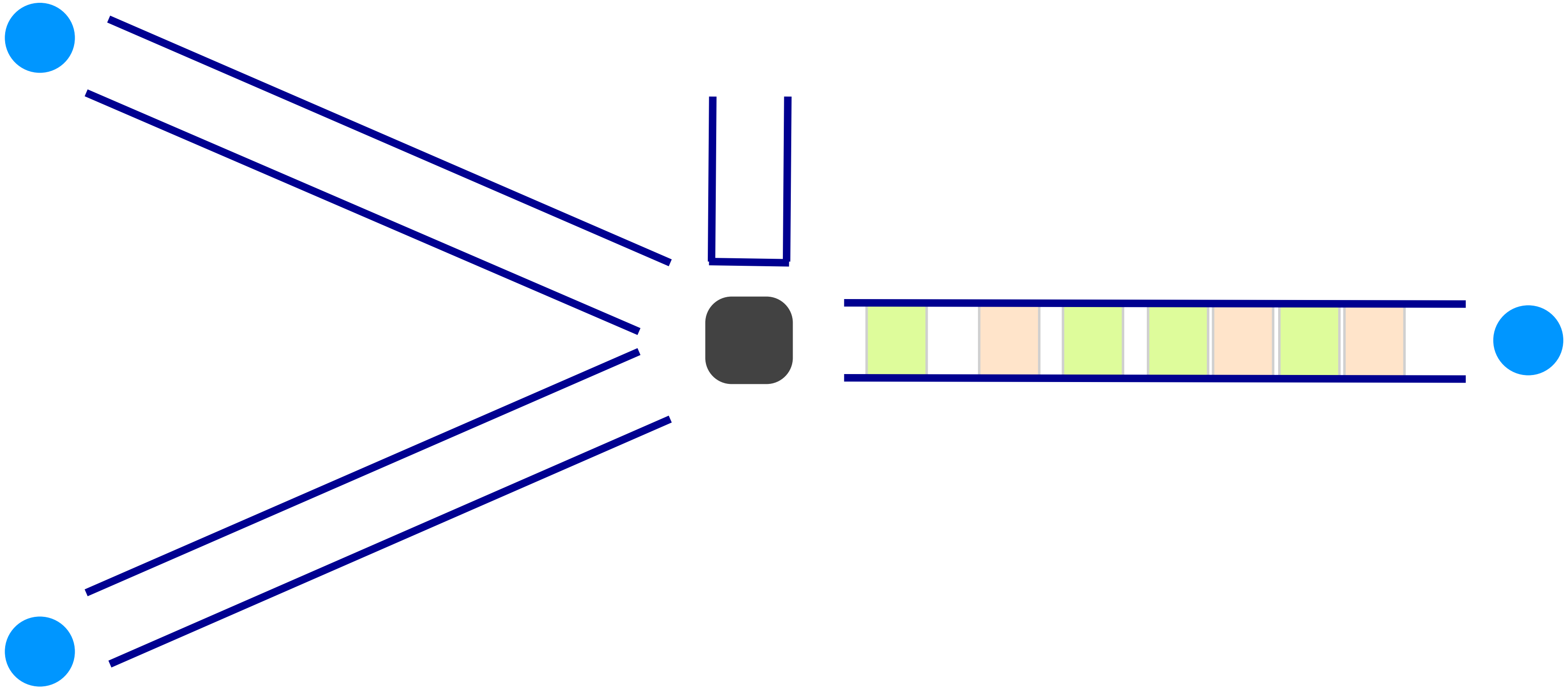
Transient overload!







Queues absorb transient bursts,
but introduce queueing delays



The time a packet has to sit in a buffer before being processed depends on the traffic pattern

Queueing delay depends on:

- arrival rate at the queue
- transmission rate of the outgoing link
- traffic burstiness

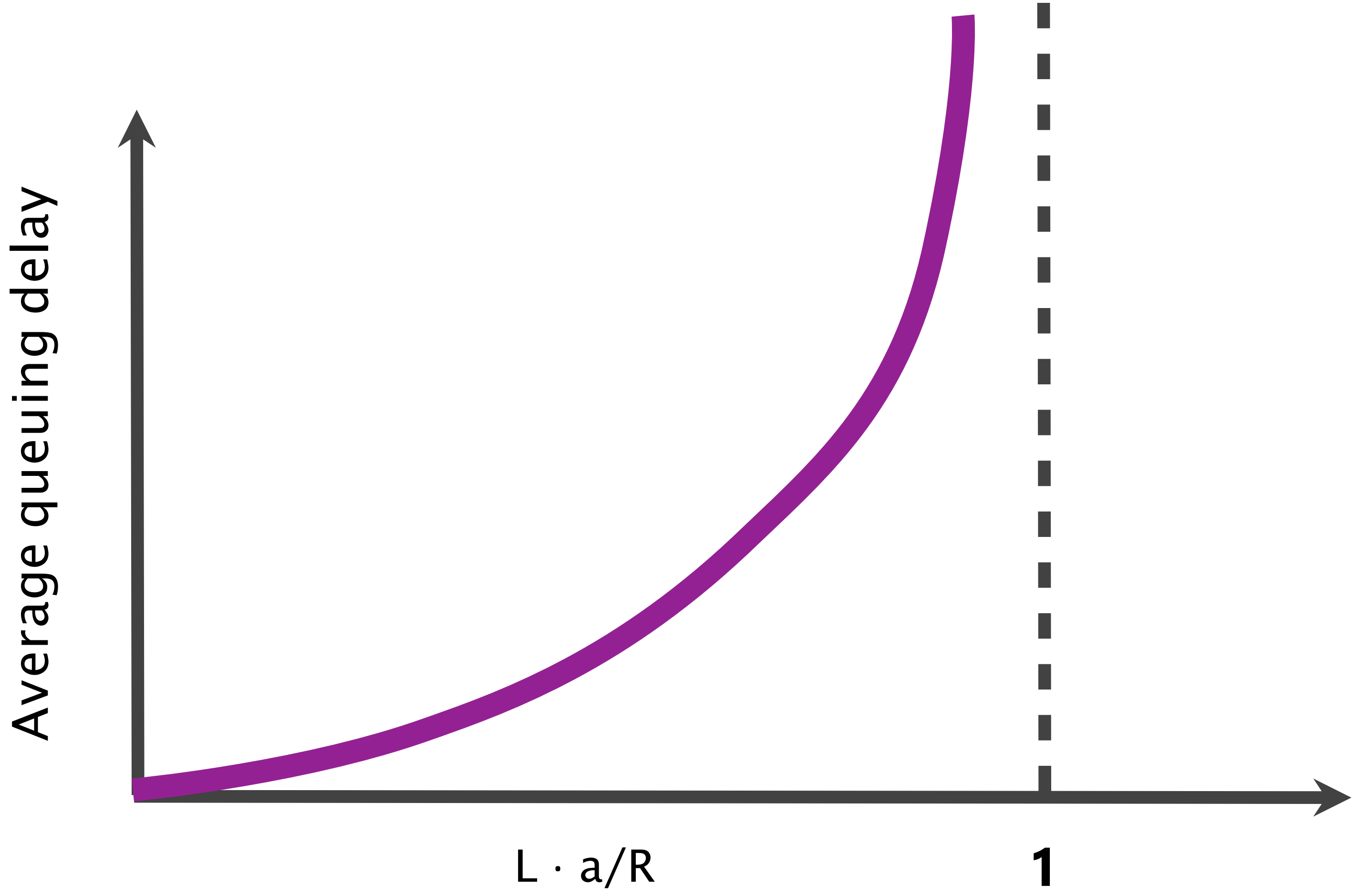
average packet arrival rate	a	[packet/sec]
transmission rate of outgoing link	R	[bit/sec]
fixed packets length	L	[bit]
average bits arrival rate	$L \cdot a$	[bit/sec]
traffic intensity	$L \cdot a/R$	

When the traffic intensity is >1, the queue will increase without bound, and so does the queuing delay

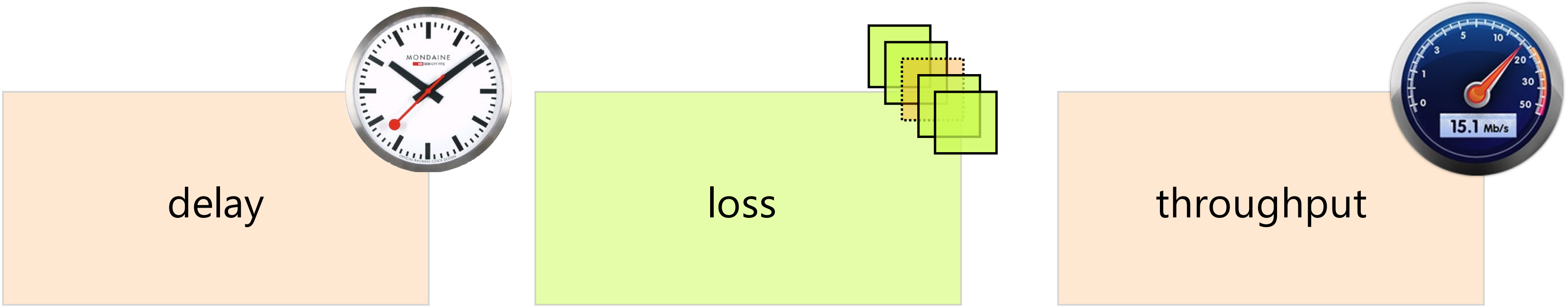
Golden rule

Design your queuing system,
so that it operates far from that point

When the traffic intensity is <=1,
queueing delay depends on the burst size

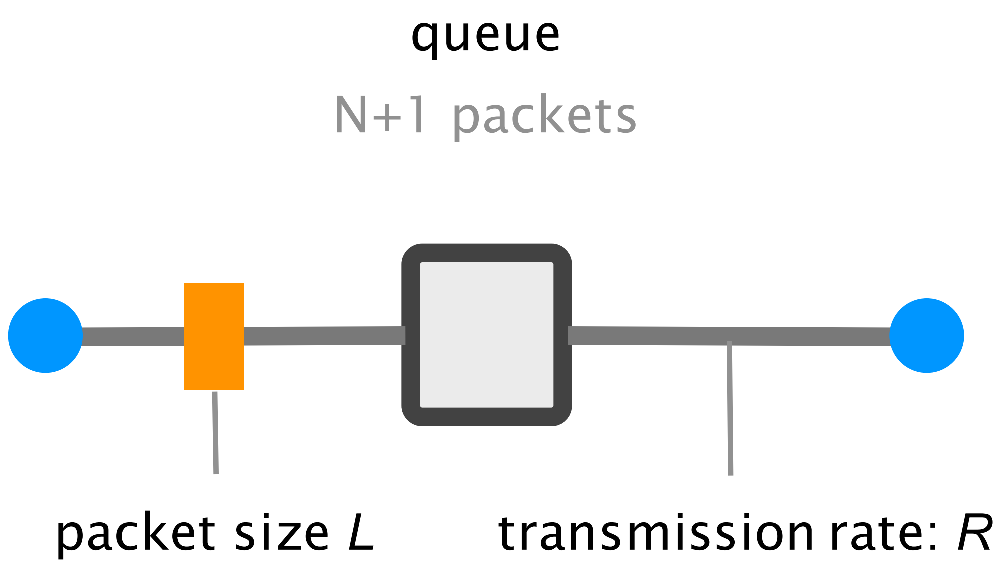


A network *connection* is characterized by its delay, loss rate and throughput



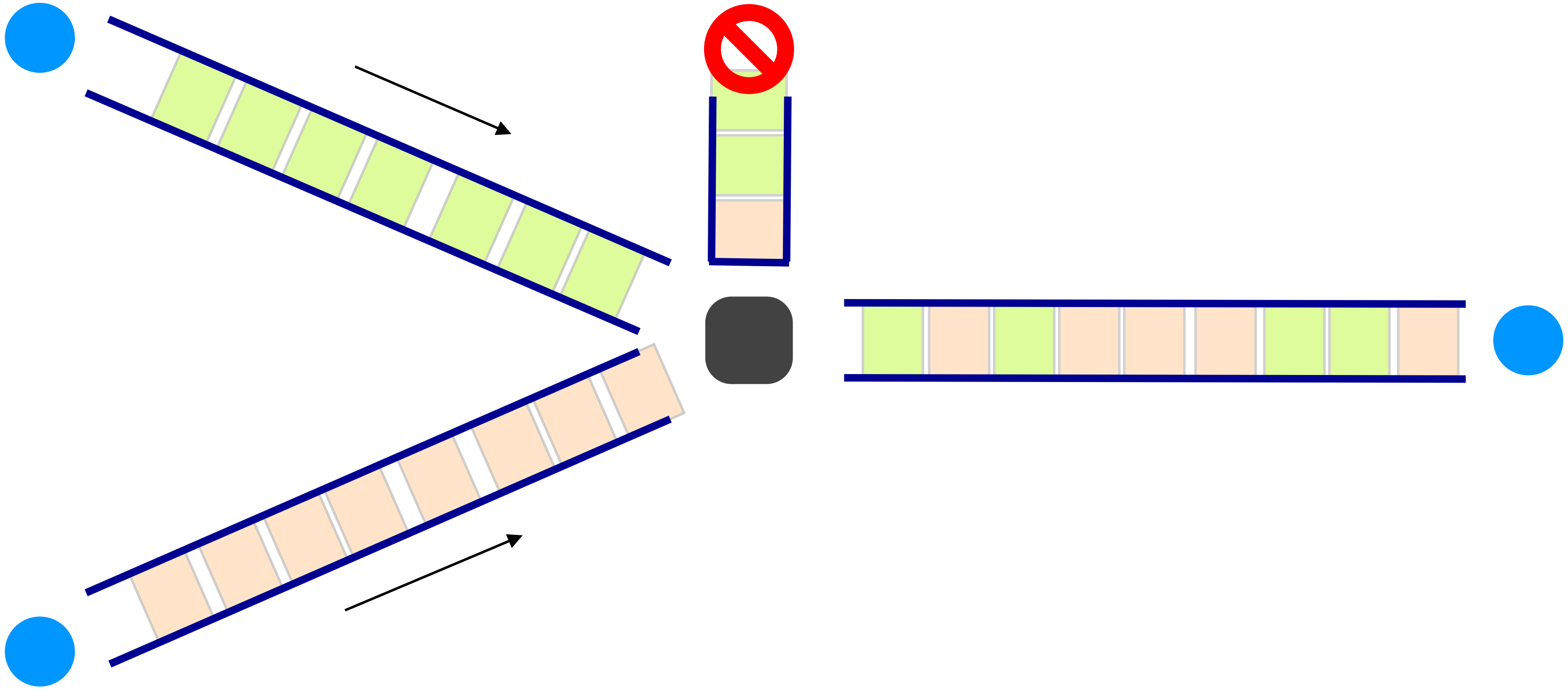
In practice, queues are not infinite.

There is an upper bound on queuing delay.



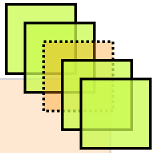
queuing delay upper bound: $N \cdot L/R$

If the queue is persistently overloaded,
it will eventually drop packets (loss)



A network *connection* is characterized by its delay, loss rate and throughput

delay 

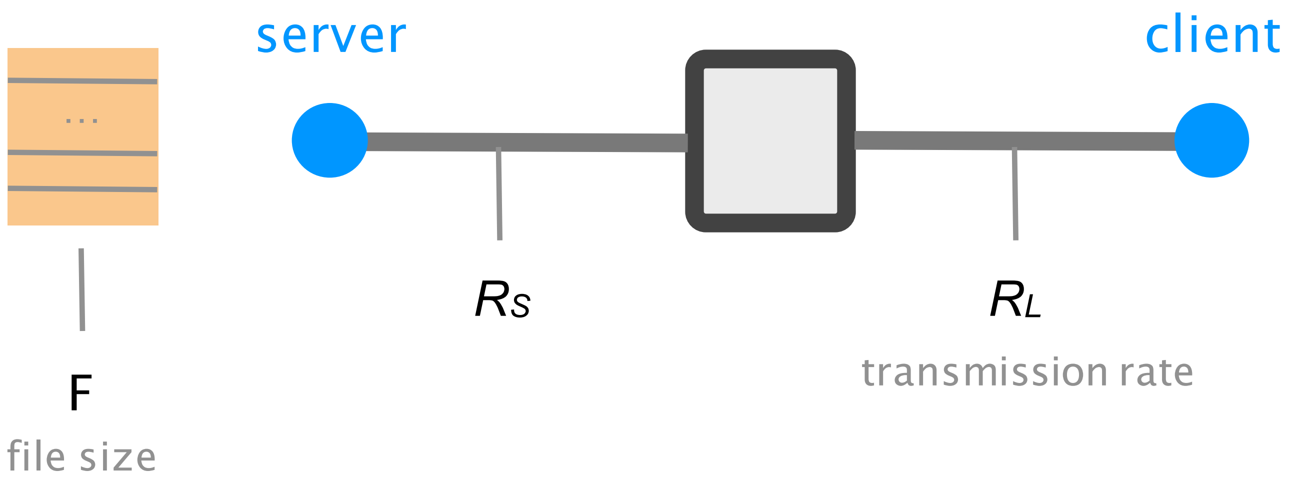
loss 

throughput 

The throughput is the instantaneous rate at which a host receives data

$$\underset{[\#bits/sec]}{\text{Average throughput}} = \frac{\text{data size} \quad [\#bits]}{\text{transfer time} \quad [sec]}$$

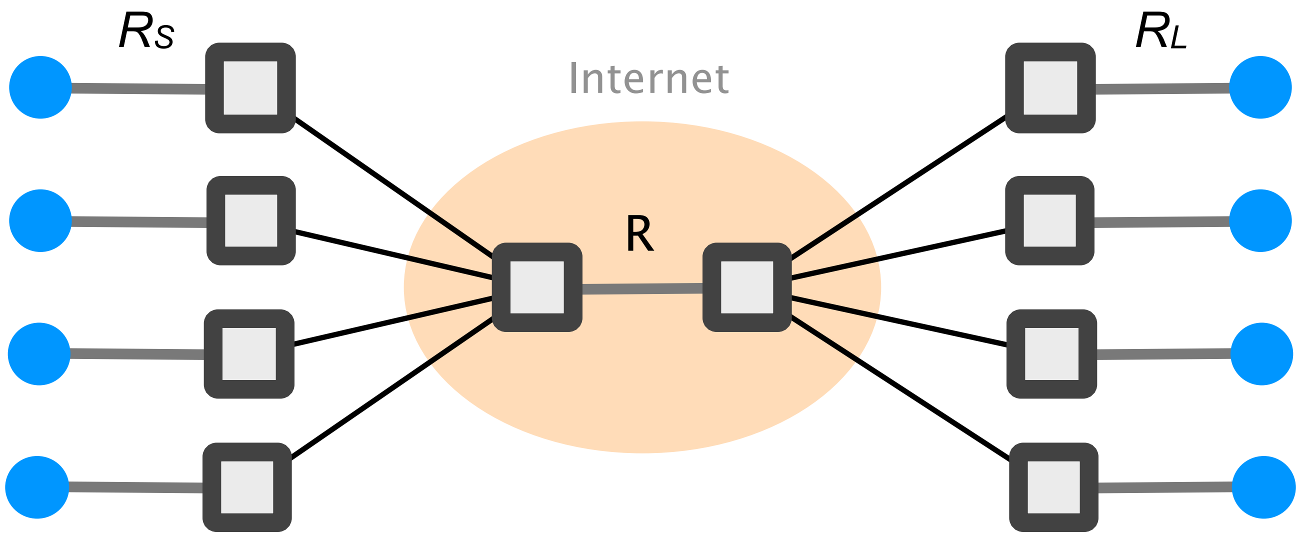
To compute throughput, one has to consider the bottleneck link



Average throughput $\min(R_S, R_L)$

= transmission rate of the bottleneck link

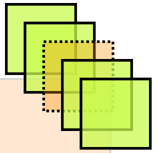
To compute throughput, one has to consider the bottleneck link… and the intervening traffic



if $4 \cdot \min(R_S, R_L) > R$ the bottleneck is now in the core, providing each download R/4 of throughput

A network *connection* is characterized by its delay, loss rate and throughput

delay

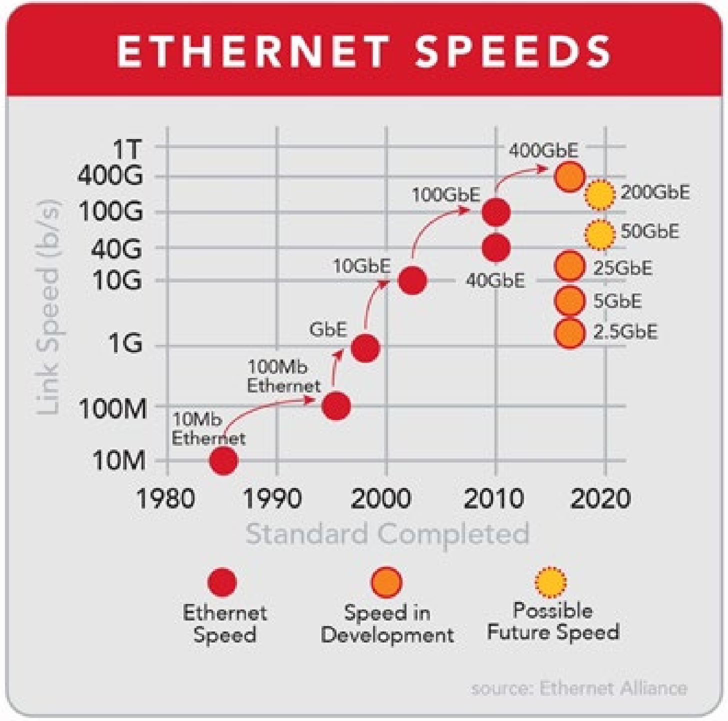


loss

throughput

As technology improves, throughput increase & delays are getting lower except for propagation

(speed of light)



source: ciena.com

Because of propagation delays,

Content Delivery Networks move content closer to you

Communication Networks and Internet Technology

Part 1: General overview

What is a network made of?

How is it shared?

How is it organized?

How does communication happen?

How do we characterize it?

The basic protocols underlying the Internet
are *intuitive*

The principles behind the Internet are *more about architecture than engineering*

Principles

Interconnect many different networks

Ethernet, Optical Fibers, wireless, …

Scale to the entire world

both geographically and numerically

Tolerate and recover from failures

both constant and inevitable

The principles behind the Internet are

more about architecture than engineering

Architecture

Engineering

what tasks get done

and *where*

how tasks get done

The principles behind the Internet are

more about architecture than engineering

Architecture

Engineering

what tasks get done

and *where*

in the network?

in the hosts?

how tasks get done

with what technology?

Network engineering is all about optimization and balancing tradeoffs

Goals	Speed	Quality of Service
	Cost	Security
	Port density	…
	Reliability	

Too small timers will cause unnecessary retransmissions,
too large timers will slow down the communication

The “right” value depends on the network conditions

Protocols have to be flexible and adapt to them

Communication Networks and Internet Technology

Part 2: Concepts

routing

reliable delivery

Communication Networks and Internet Technology

Part 2: Concepts

routing	reliable delivery
How do you guide IP packets from a source to destination?	How do you ensure reliable transport on top of best-effort delivery?

This week

routing

How do you guide IP packets
from a source to destination?

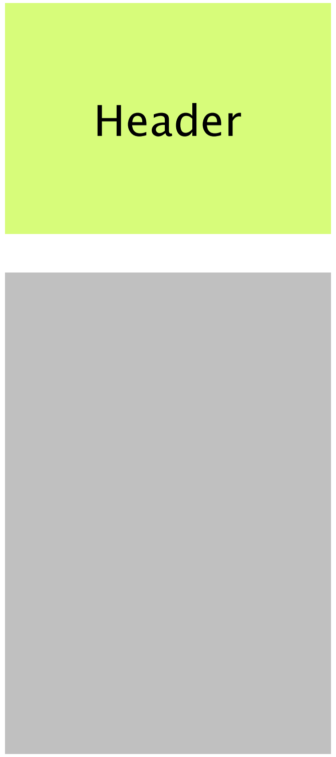
Next week

reliable
delivery

Think of IP packets as envelopes

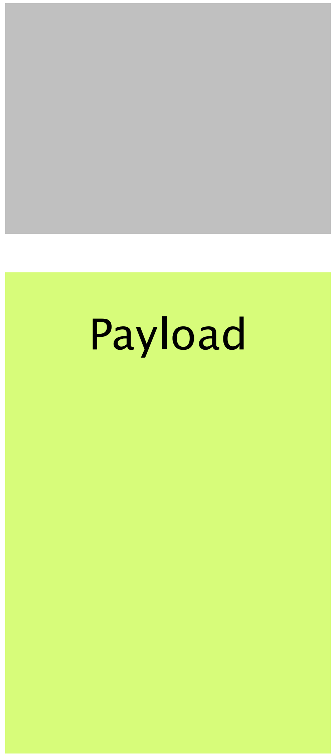
Like an envelope,

packets have a header

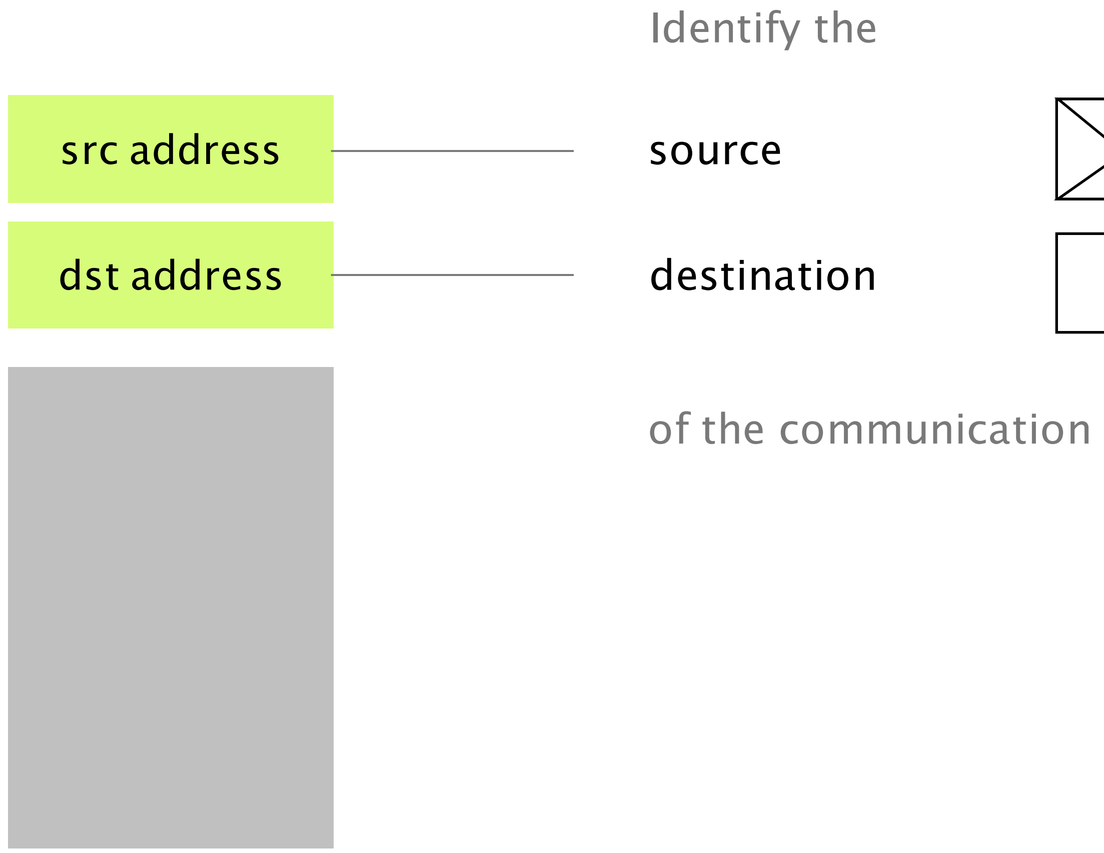


Like an envelope,

packets have a payload

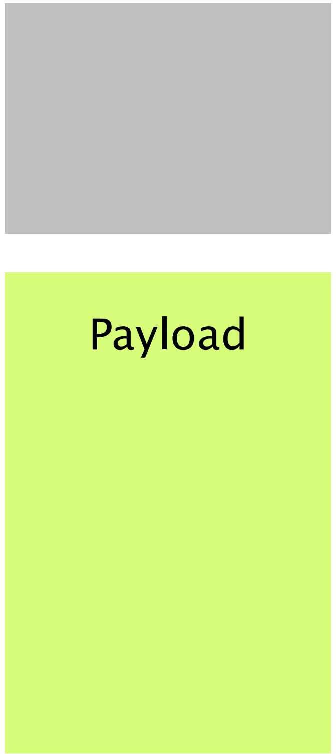


The **header** contains the **metadata** needed **to forward** the packet

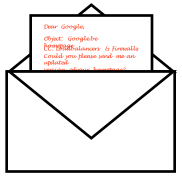


The payload contains

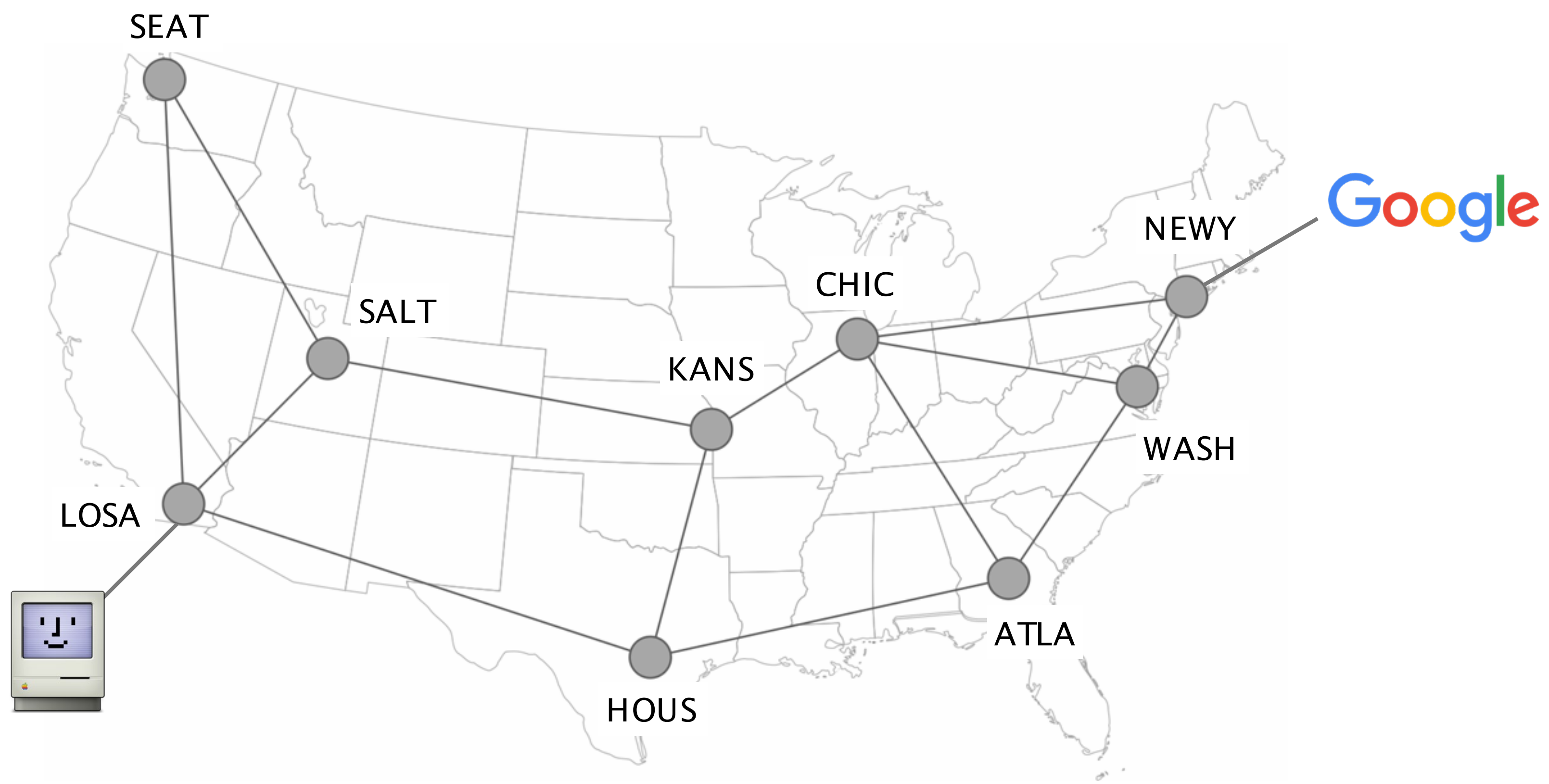
the data to be delivered

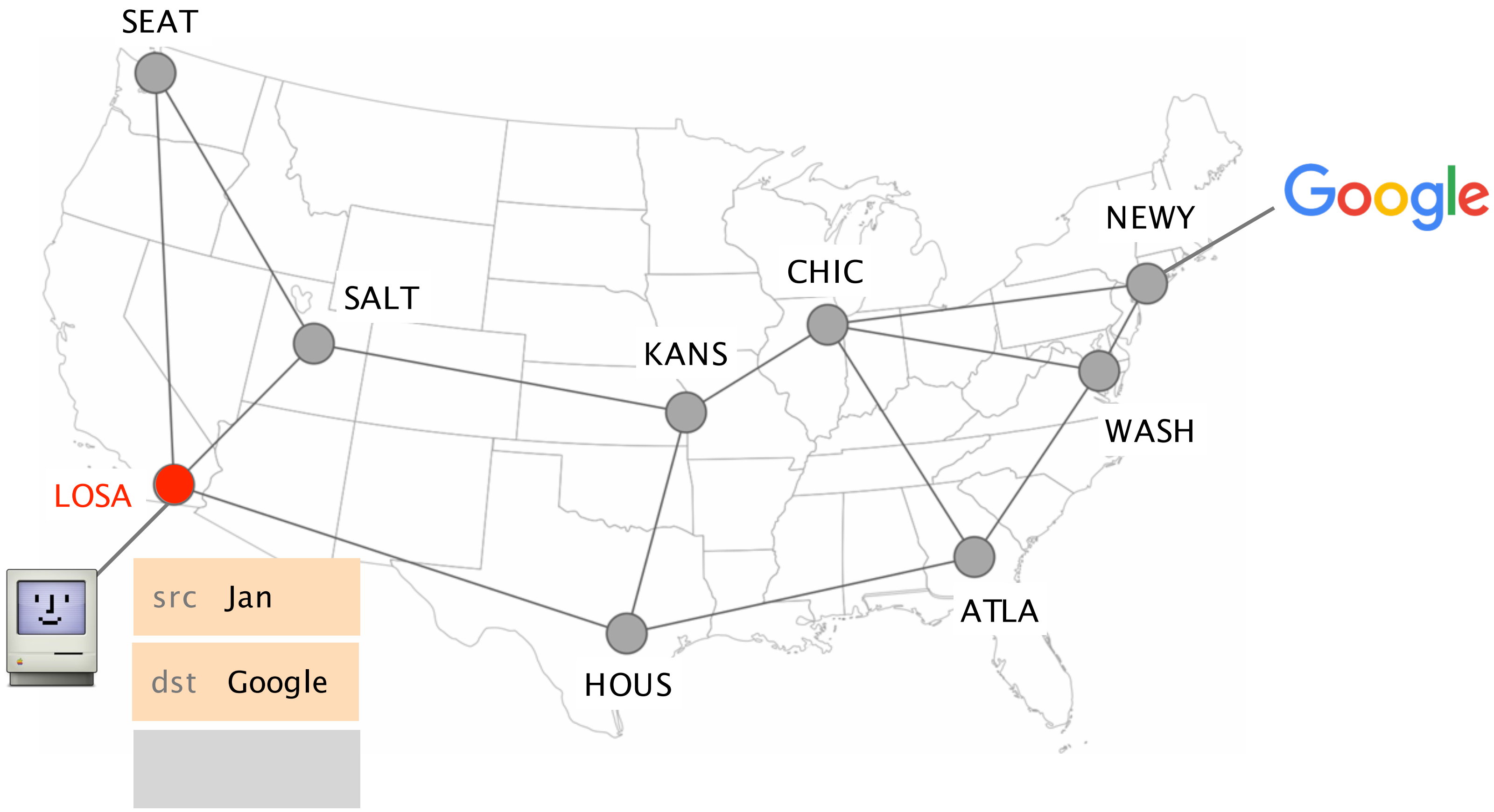


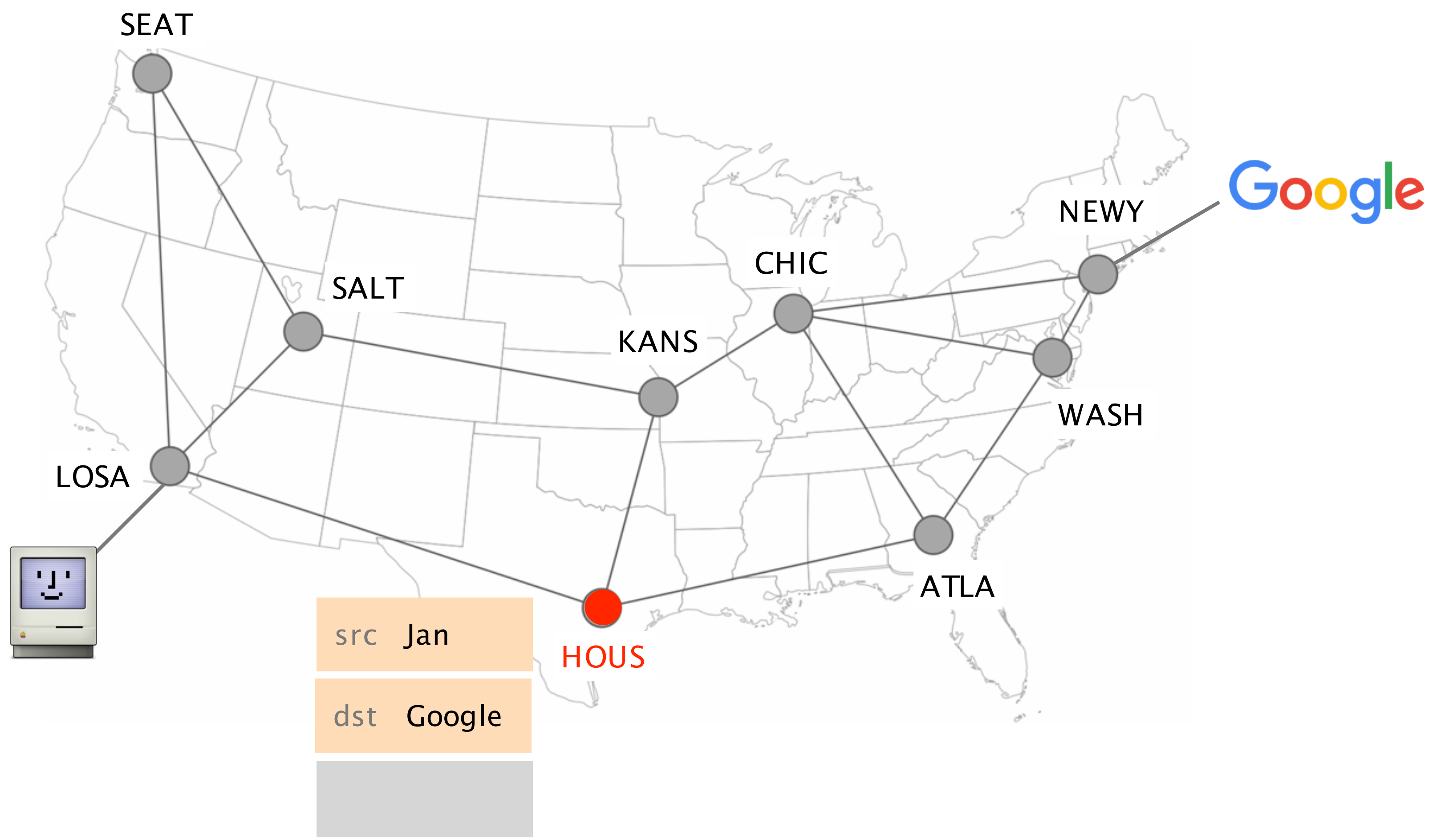

```html
<html><head>
<meta http-equiv="content-type" content="text/html; charset=UTF-8">
<title>Google</title>
</head><body>
  <img alt="Google" height=110 src="images/logo.gif" width=276>
  <form action="/search" name=f>
   <input name=hl type=hidden value=en>
   <input name=q size=55 title="Google Search" value="">
   <input name=btnG type=submit value="Google Search">
   <input name=btnI type=submit value="I'm Feeling Lucky">
 </form>
</body></html>
```

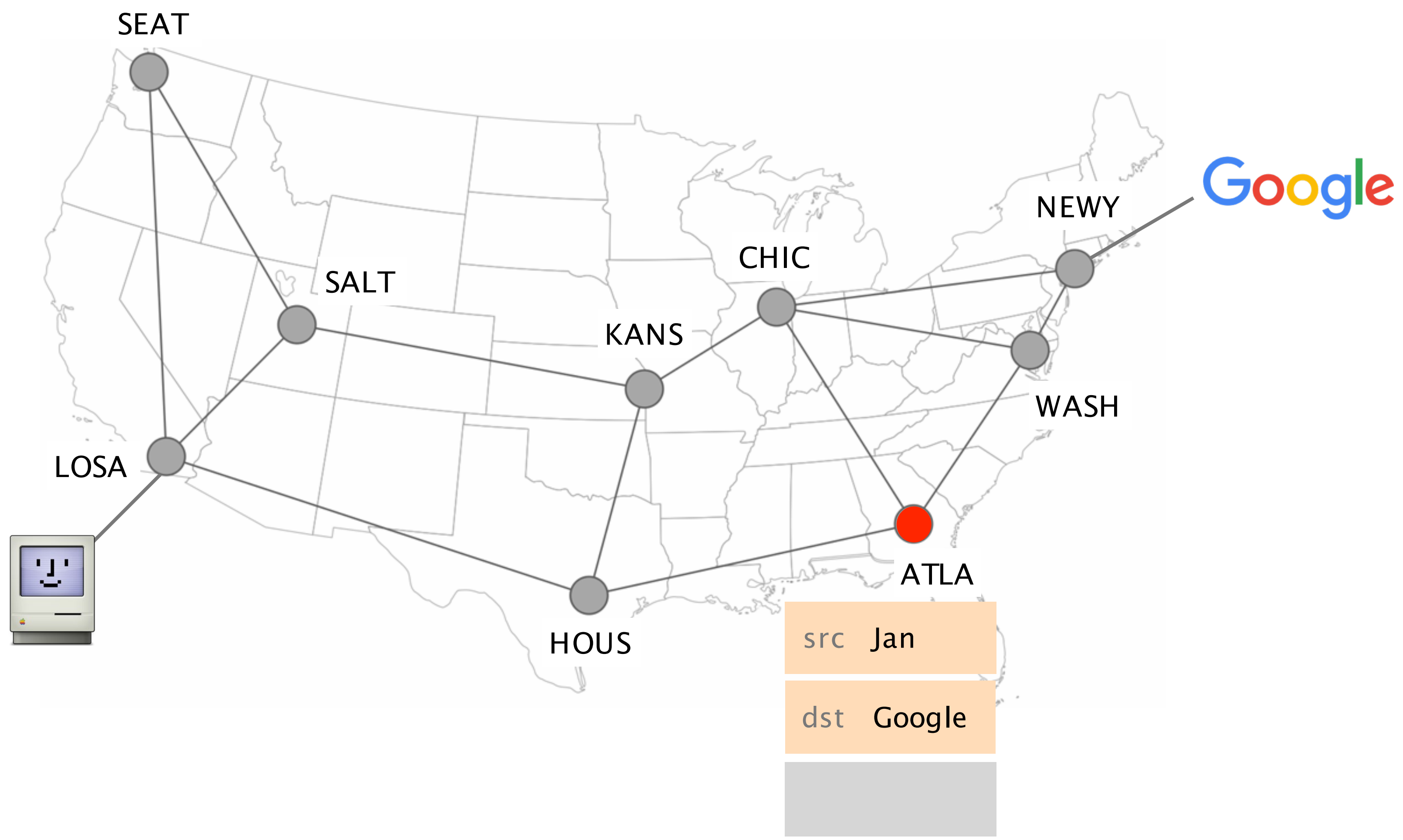


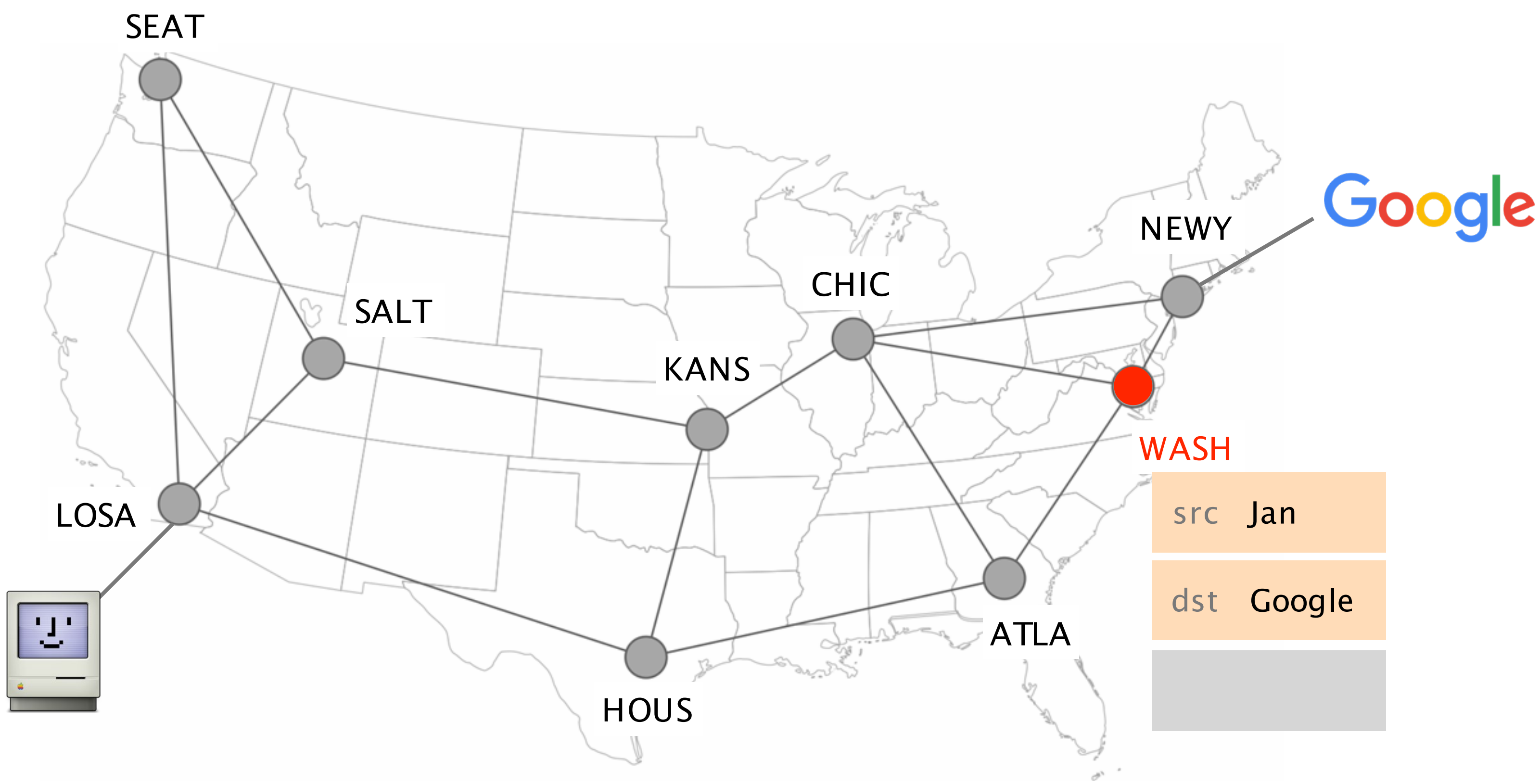
Routers forward IP packets **hop-by-hop** towards their destination

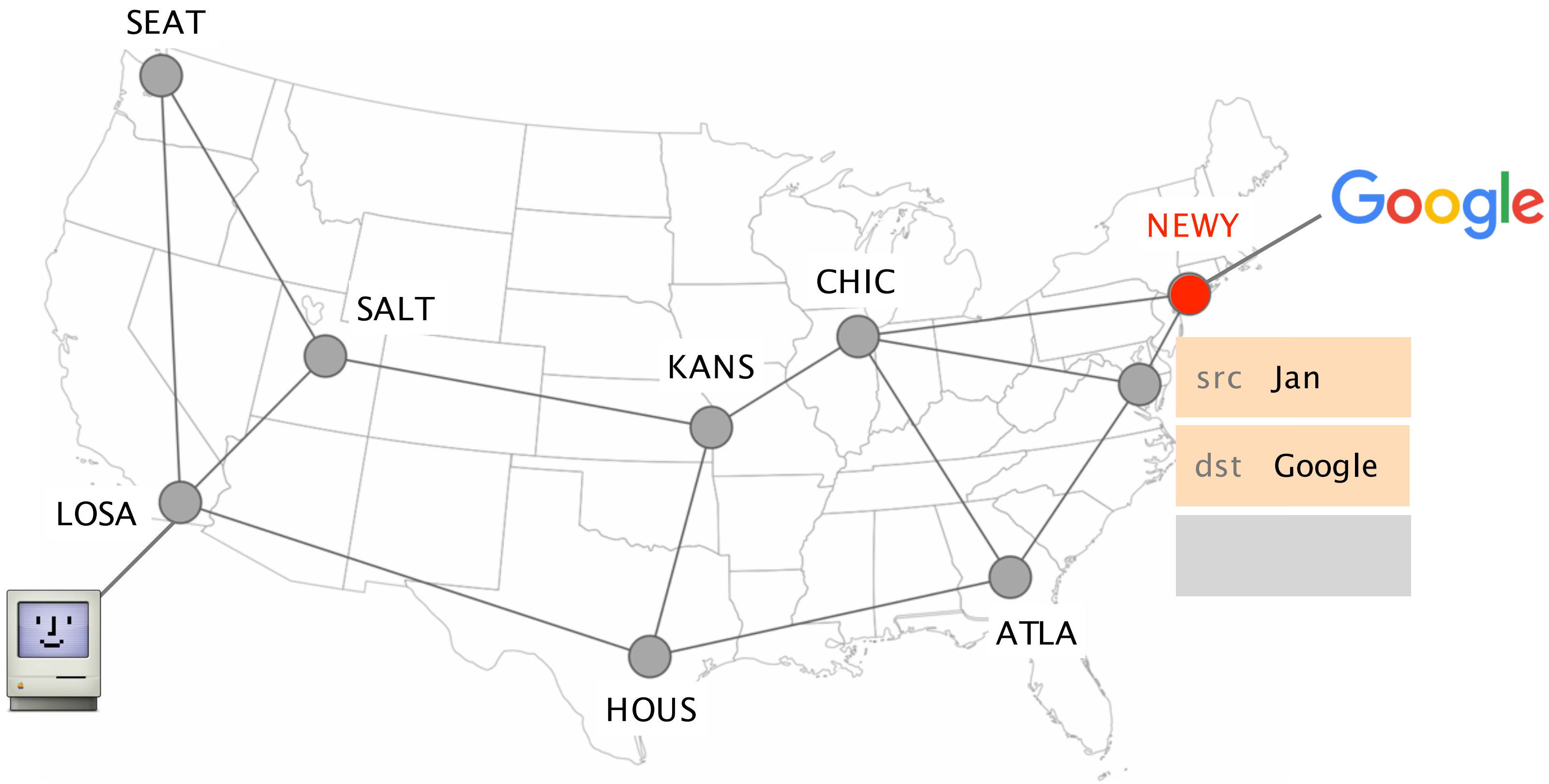


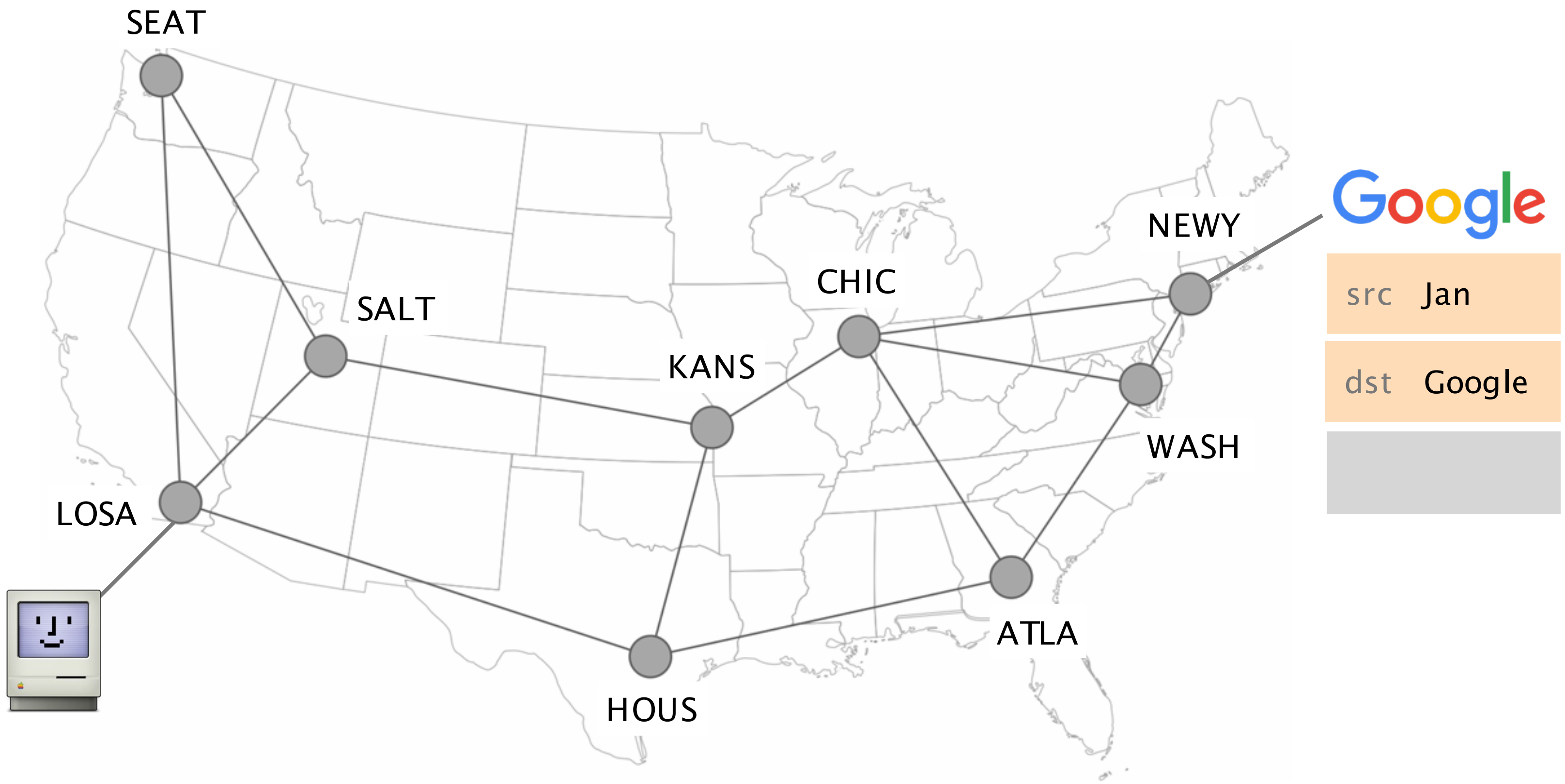

SEAT
SALT
KANS
CHIC
NEWY
WASH
ATLA
HOUS
LOSA
Google
src Jan
dst Google


SEAT
SALT
KANS
CHIC
NEWY
Google
WASH
LOSA
ATLA
HOUS
src Jan
dst Google


SEAT
SALT
LOSA
KANS
CHIC
NEWY
WASH
ATLA
HOUS
Google
src Jan
dst Google

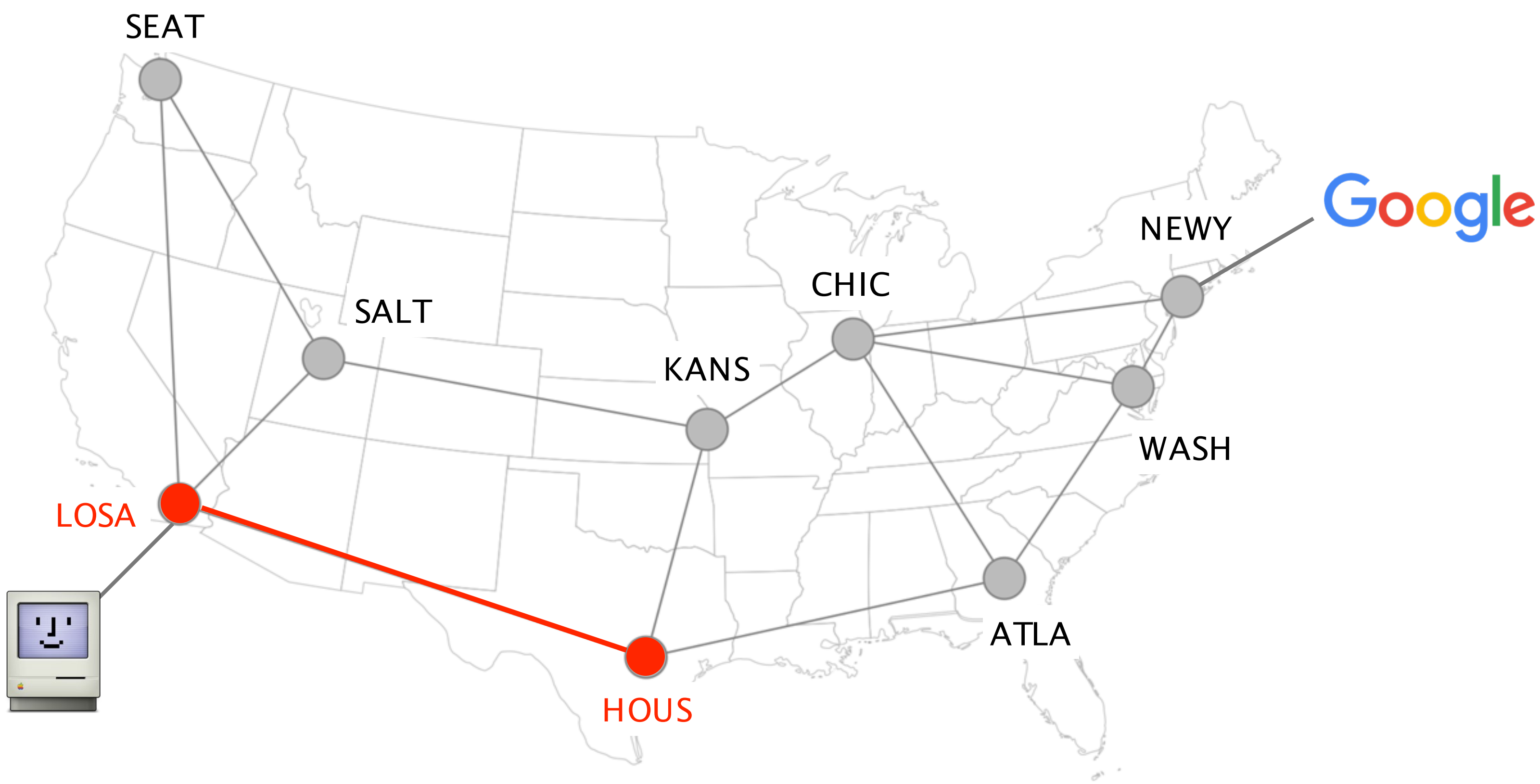

SEAT
SALT
KANS
CHIC
NEWY
WASH
LOSA
HOUS
ATLA
src Jan
dst Google

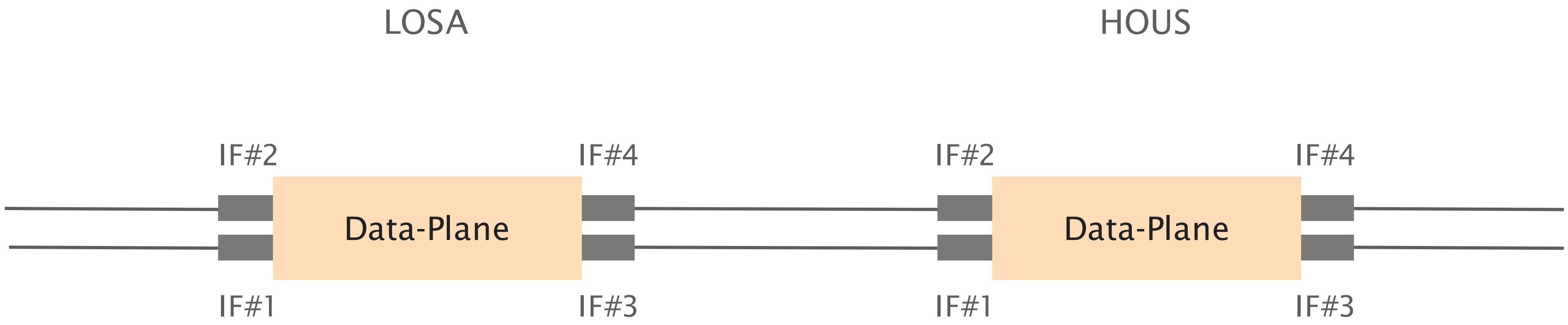

SEAT
SALT
KANS
CHIC
NEWY
LOSA
HOUS
ATLA
Google
src Jan
dst Google



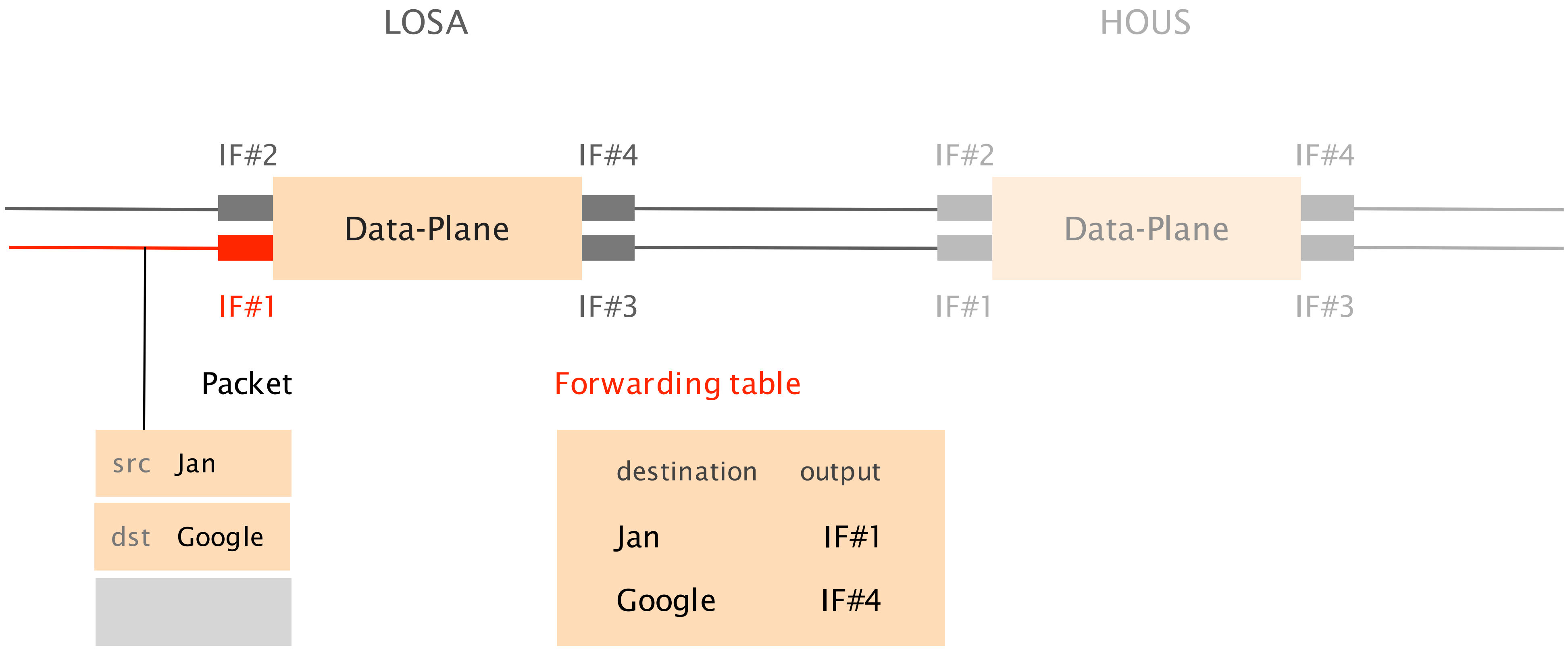
SEAT
SALT
LOSA
KANS
HOUS
CHIC
NEWY
WASH
ATLA
Google
src Jan
dst Google

Let's zoom in on what is going on between **two adjacent routers**




LOSA
HOUS
IF#2
IF#4
IF#2
IF#4
Data-Plane
Data-Plane
IF#1
IF#3
IF#1
IF#3

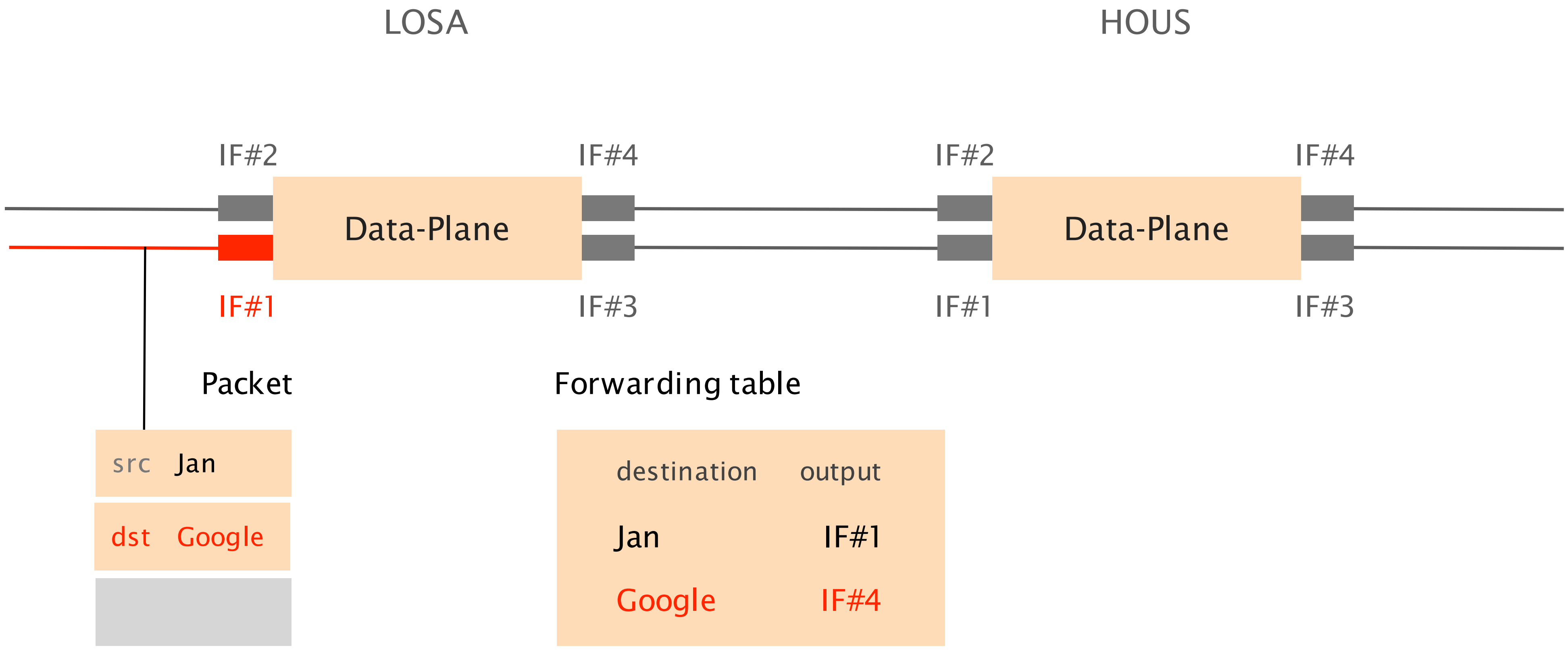
Upon packet reception, routers locally look up their forwarding table to know where to send it next

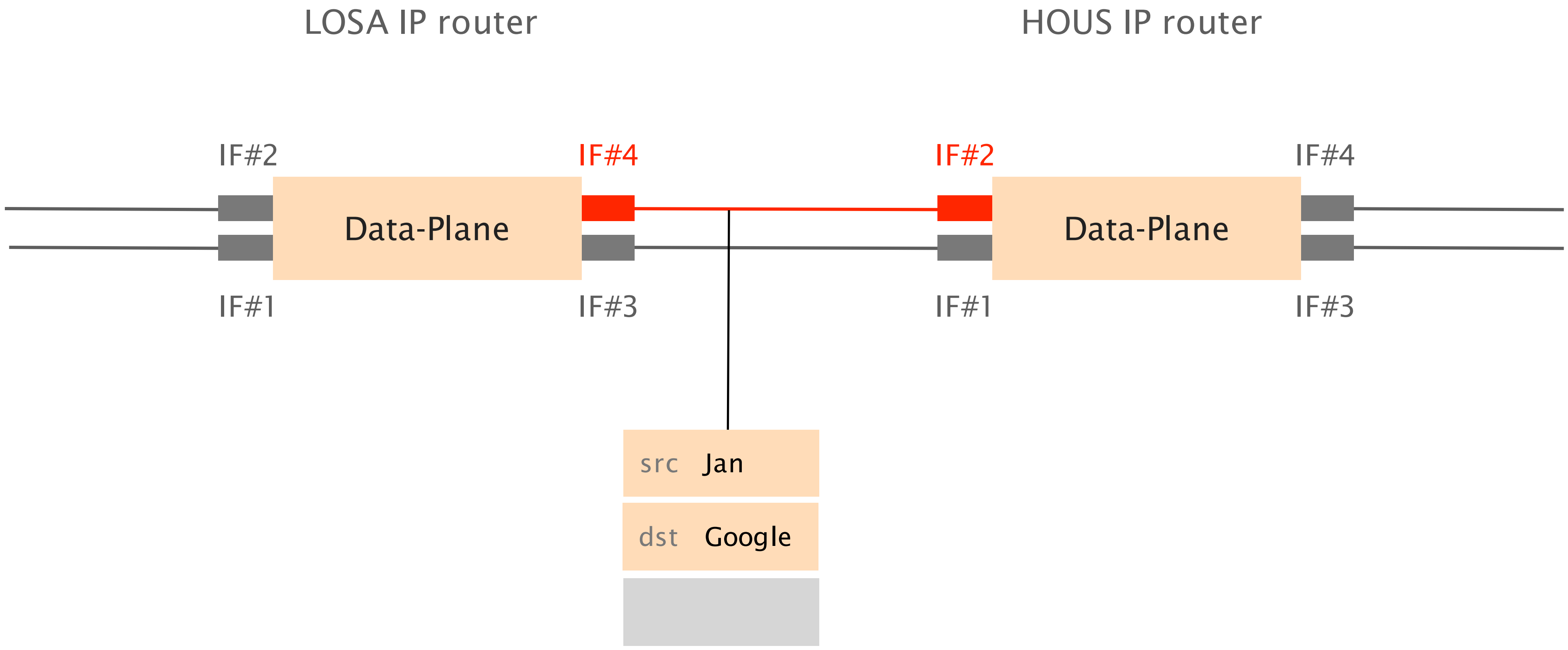


Forwarding table

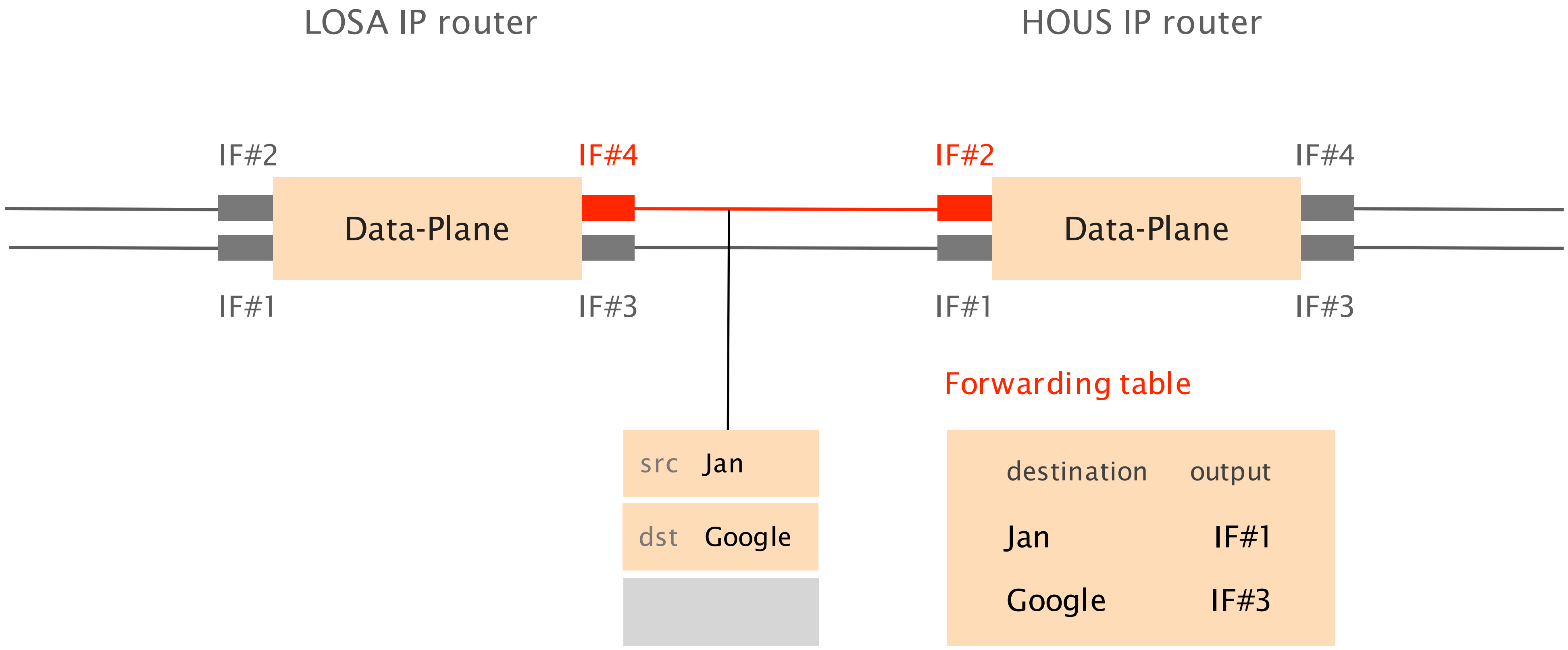
destination	output
Jan	IF#1
Google	IF#4

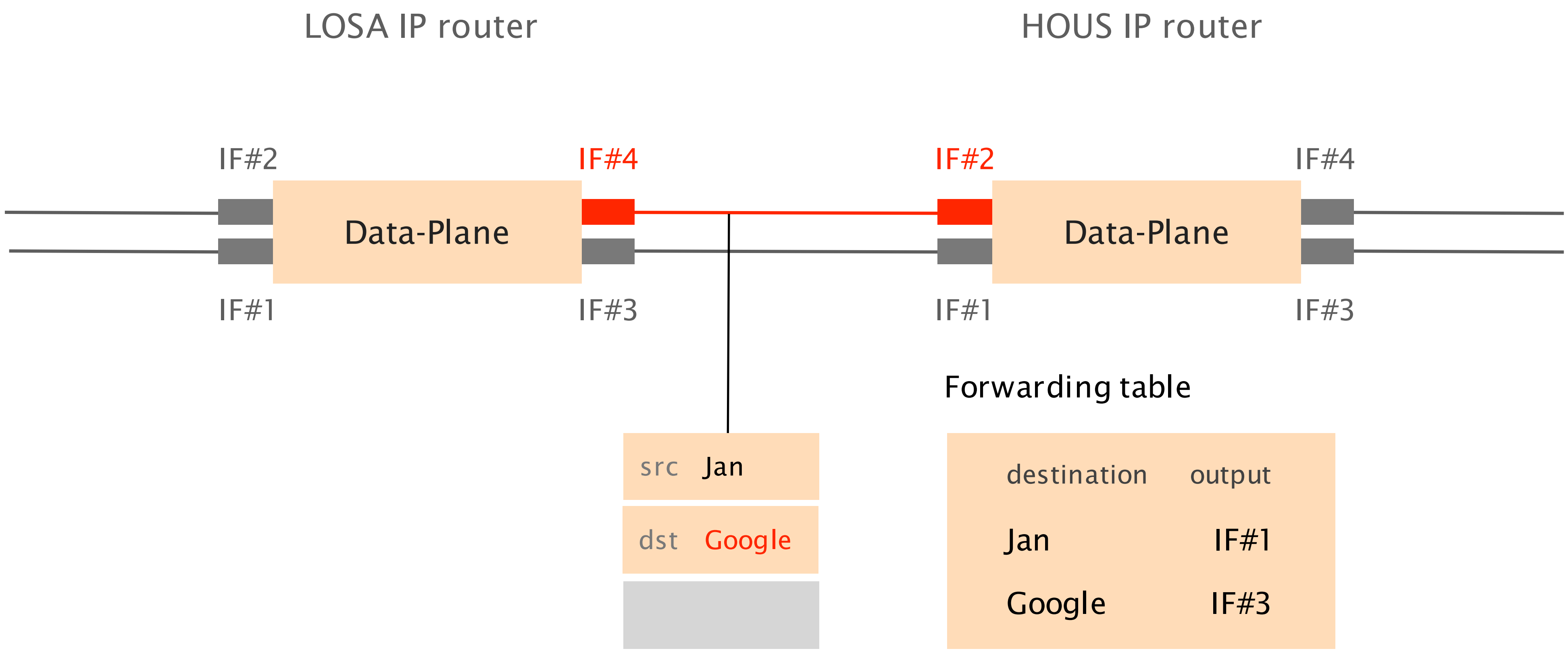
Here, the packet should be directed to IF#4

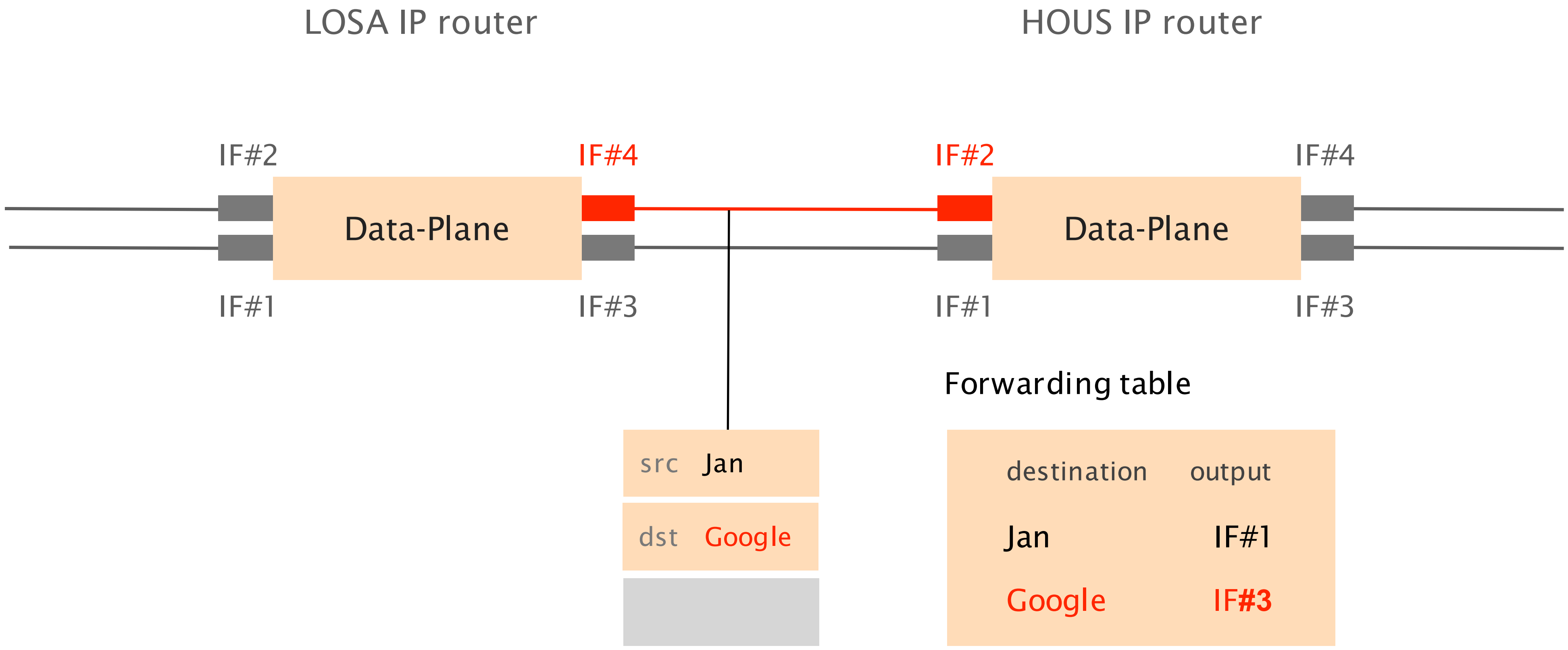


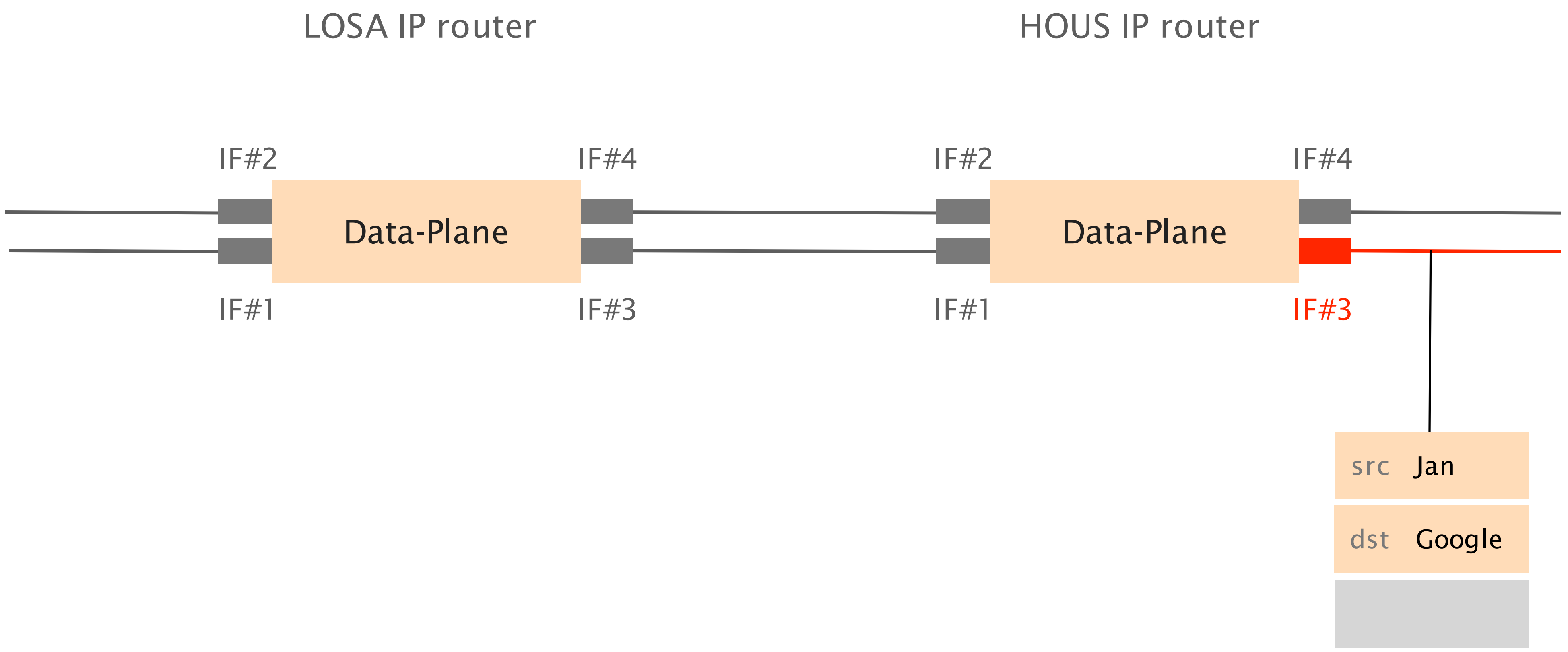

LOSA IP router
HOUS IP router
IF#2
IF#4
IF#2
IF#4
Data-Plane
Data-Plane
IF#1
IF#3
IF#1
IF#3
src Jan
dst Google

Forwarding is repeated at each router,
until the destination is reached

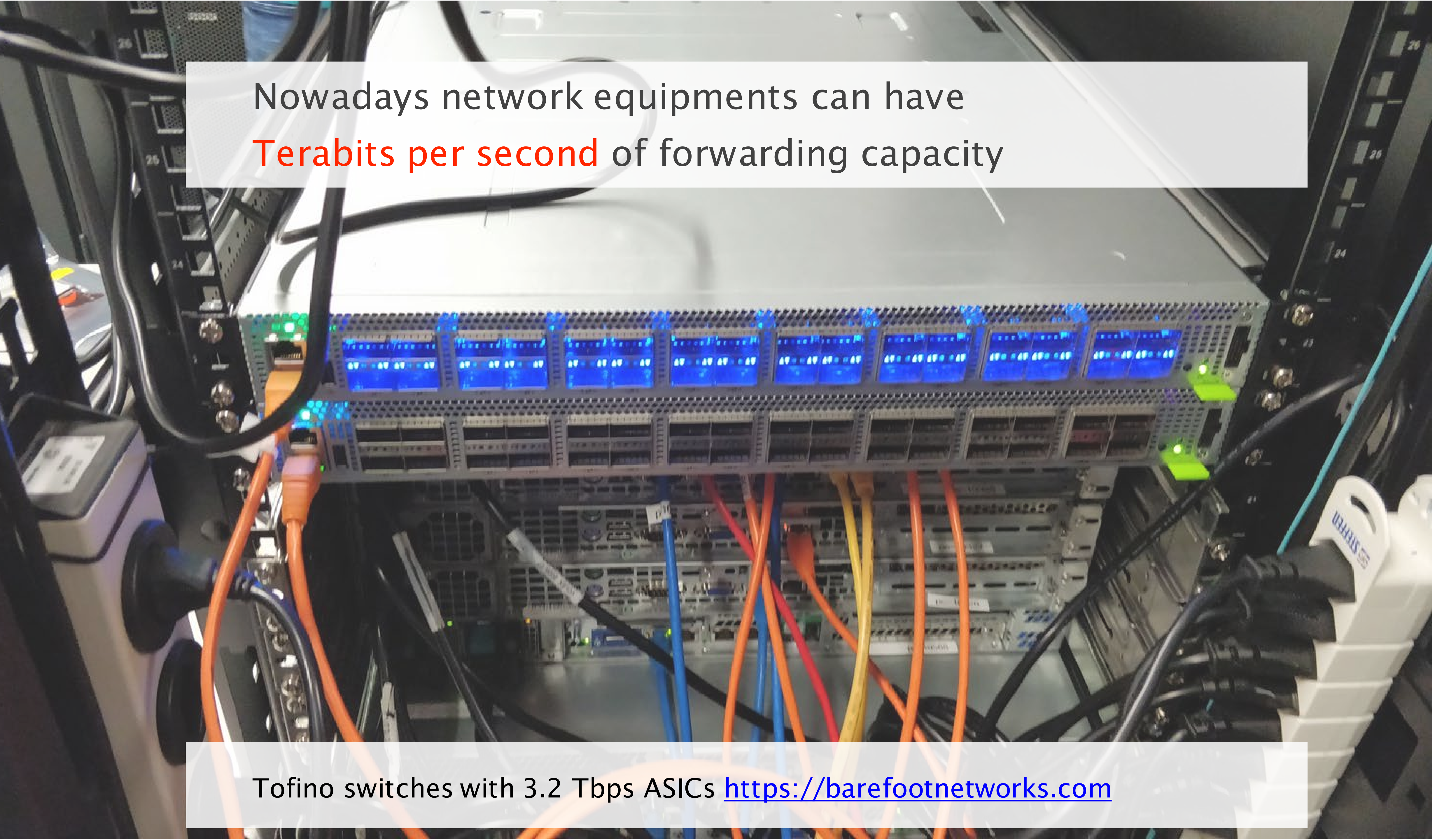



LOSA IP router
HOUS IP router
IF#2
IF#4
IF#2
IF#4
Data-Plane
Data-Plane
IF#1
IF#3
IF#1
IF#3
Forwarding table
src Jan
dst Google
destination output
Jan IF#1
Google IF#3


LOSA IP router
HOUS IP router
IF#2
IF#4
IF#2
IF#4
Data-Plane
Data-Plane
IF#1
IF#3
IF#1
IF#3
Forwarding table
src Jan
dst Google
destination output
Jan IF#1
Google IF#3



LOSA IP router
HOUS IP router
IF#2
IF#4
IF#2
IF#4
Data-Plane
Data-Plane
IF#1
IF#3
IF#1
IF#3
src Jan
dst Google



Nowadays network equipments can have
Terabits per second of forwarding capacity

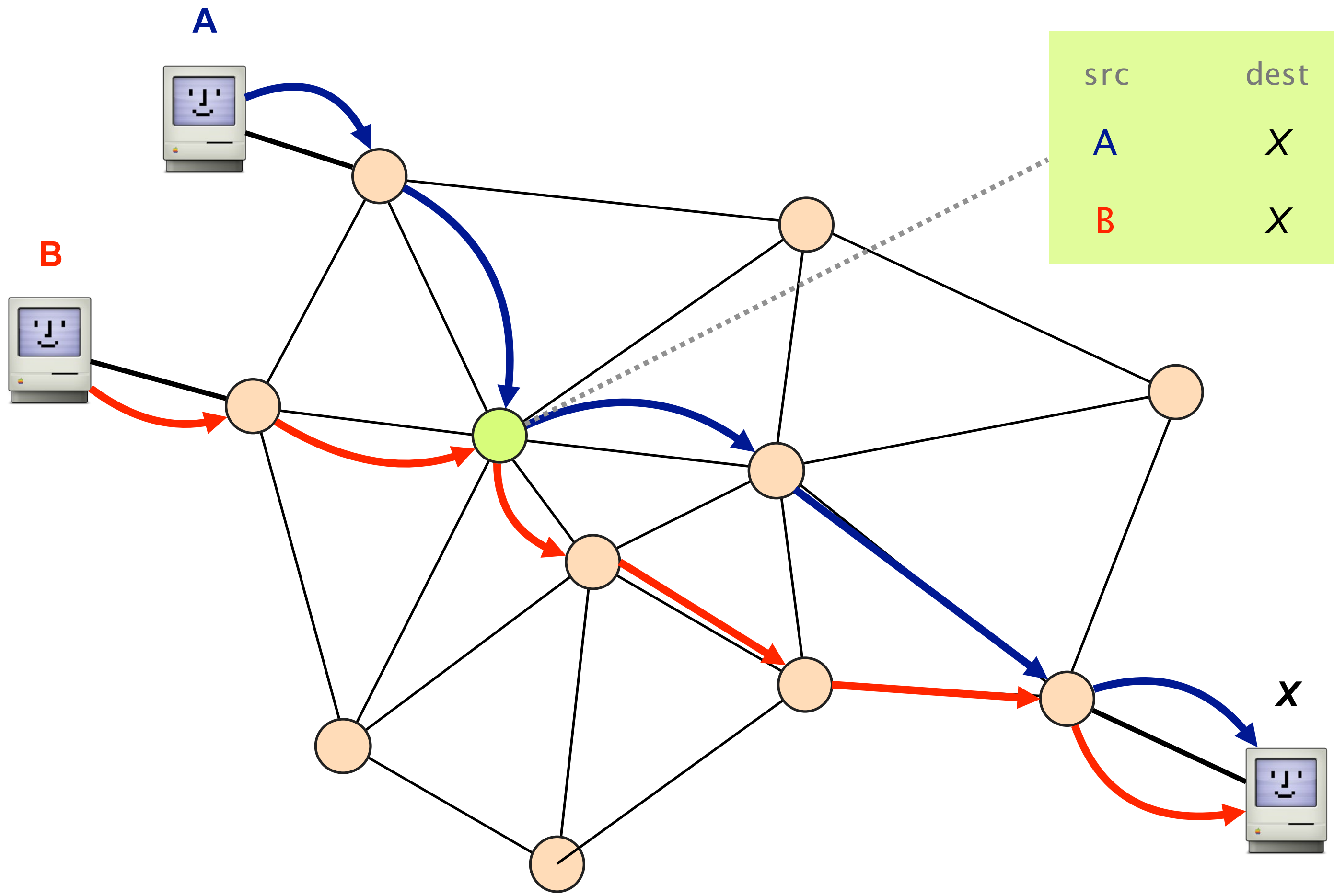
Tofino switches with 3.2 Tbps ASICs https://barefootnetworks.com

Forwarding decisions necessarily depend on the destination, but can also depend on other criteria

criteria		
	destination	mandatory (why?)
	source	requires n^2 state
	input port	traffic engineering
	+any other header	

With **source- & destination-based routing**,

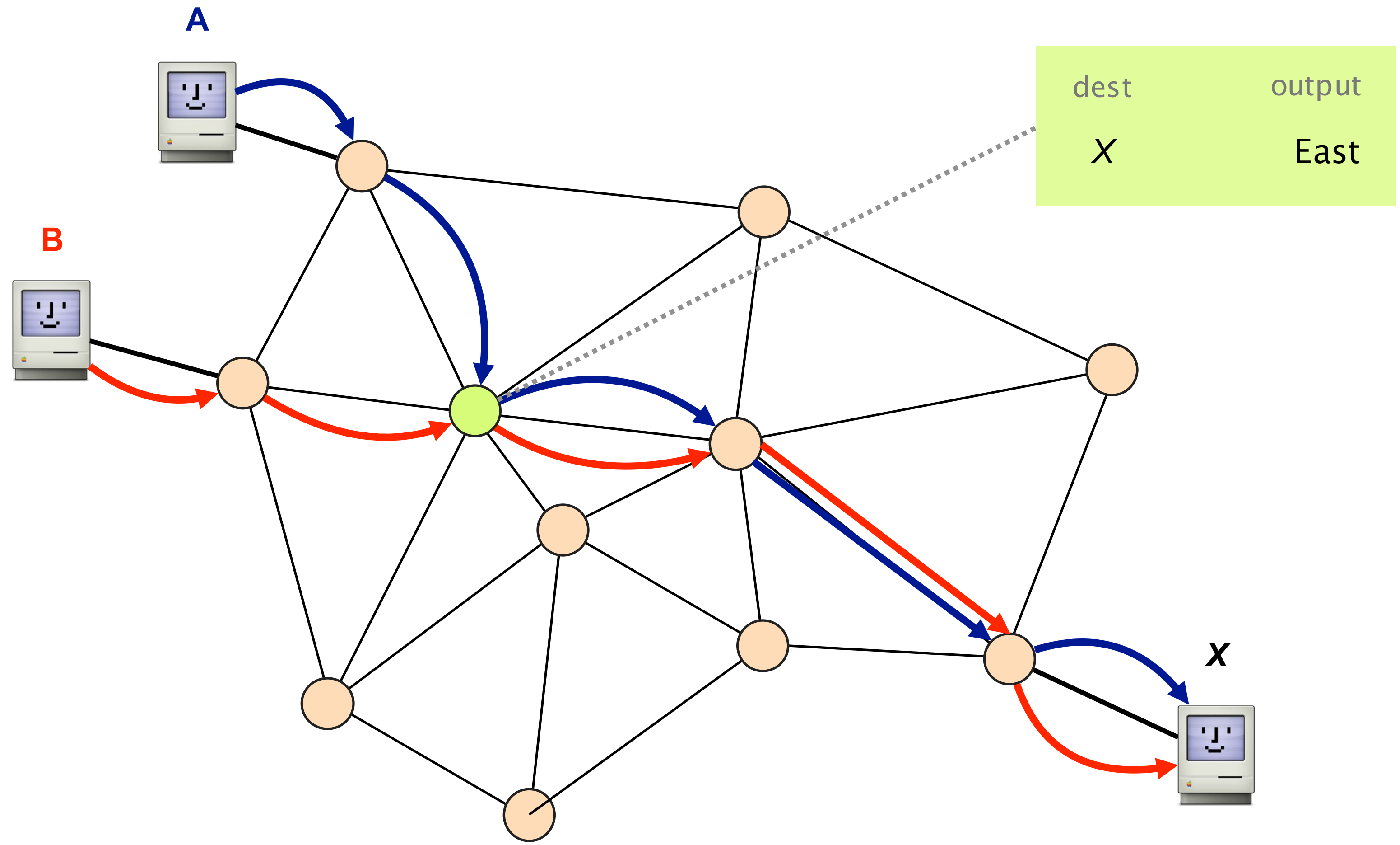
paths from different sources can differ



src	dest	output
A	*X*	East
B	*X*	South-East

With **destination-based routing**,

paths from different source coincide once they overlap



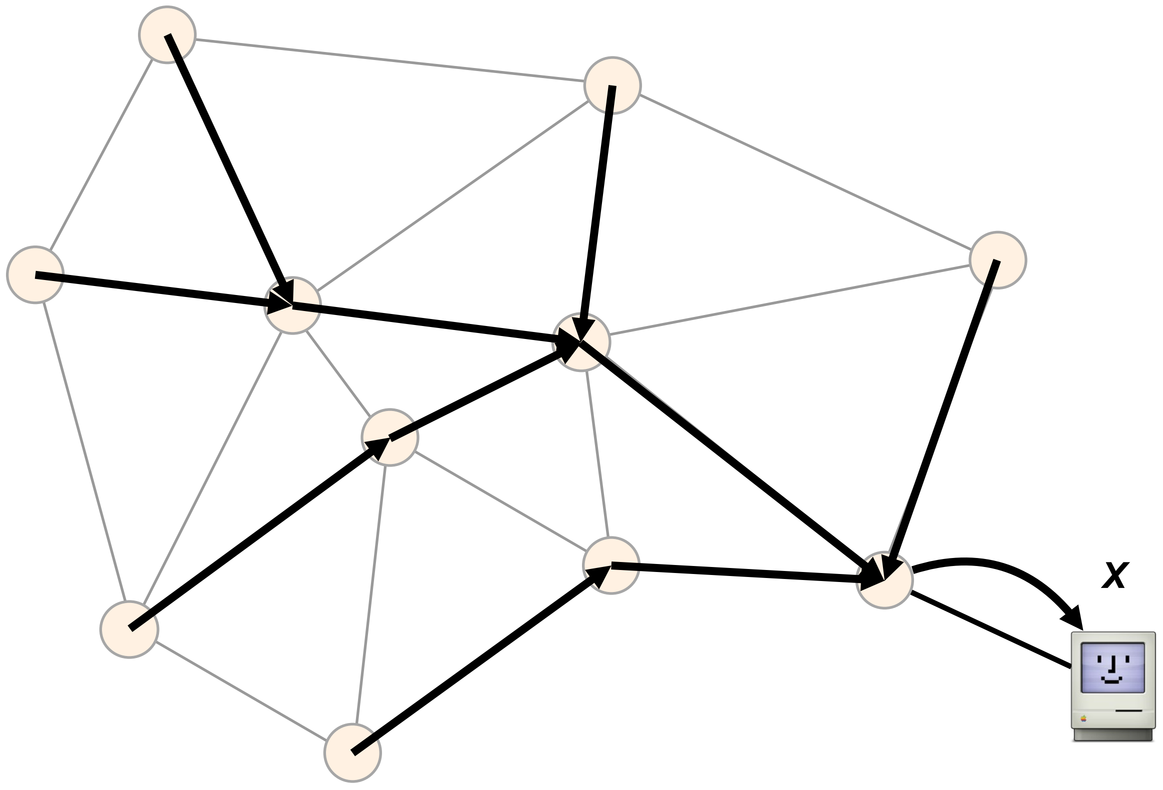
Once path to destination meet,
they will *never* split

Set of paths to the destination

produce a spanning tree rooted at the destination:

- cover every router exactly once
- only one outgoing arrow at each router

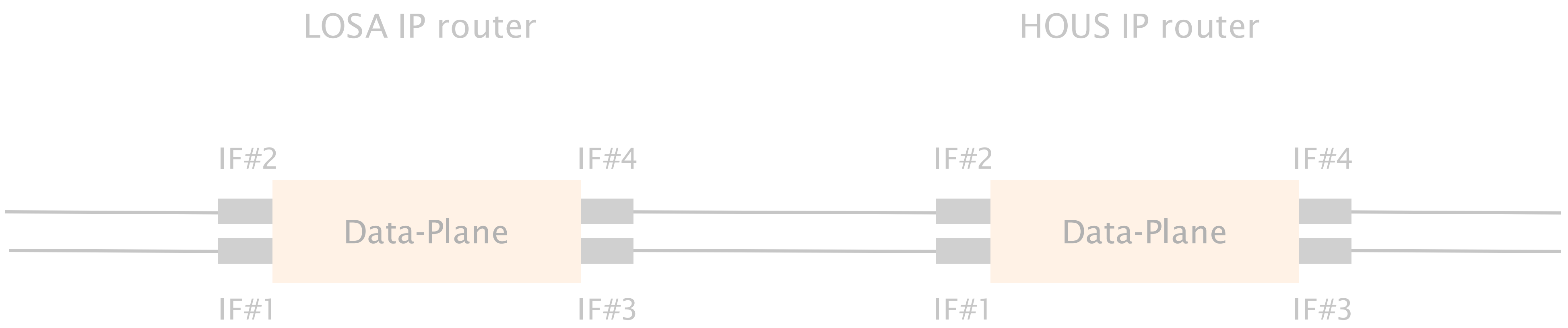
Here is an example of a spanning tree for destination *X*



In the rest of the lecture,
we'll consider destination-based routing

the default in the Internet

Where are these forwarding tables coming from?

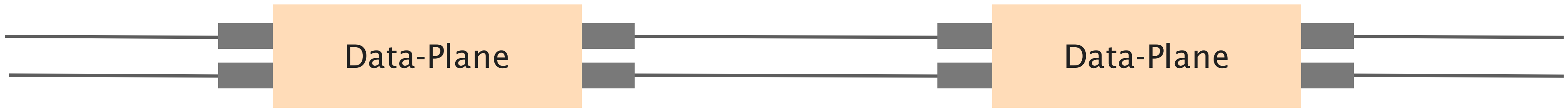


Forwarding table

destination	output
Jan	IF#1
Google	IF#4

Forwarding table

destination	output
Jan	IF#1
Google	IF#3


Data-Plane
Data-Plane

In addition to a data-plane,
routers are also equipped with a control-plane

Think of the control-plane as the router's brain

Roles

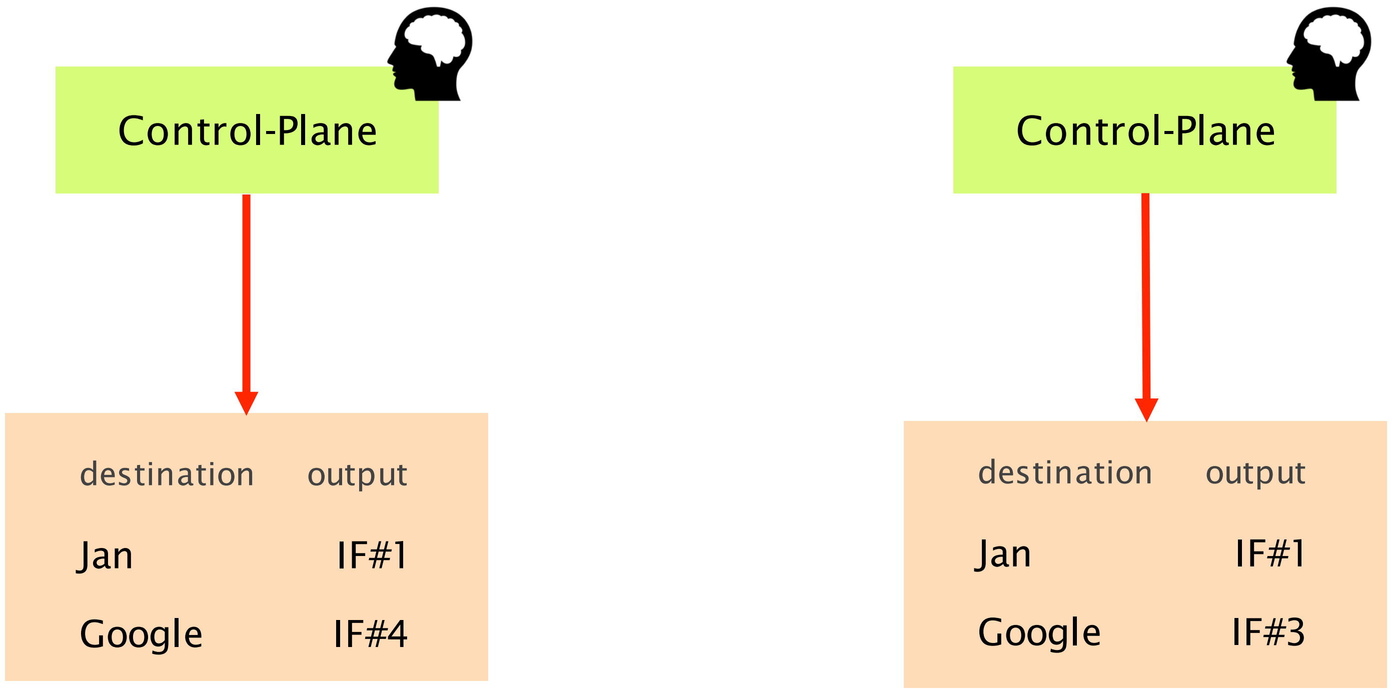
Routing

Configuration

Statistics

…

Routing is the control-plane process that computes and populates the forwarding tables



While forwarding is a *local* process, routing is inherently a *global* process

How can a router know where to direct packets if it does not know what the network looks like?

Forwarding *vs* Routing

summary

	forwarding	routing
goal	directing packet to an outgoing link	computing the paths packets will follow
scope	local	network-wide
implem.	hardware usually	software usually
timescale	nanoseconds	milliseconds (hopefully)

The goal of routing is to compute
valid global forwarding state

Definition

a global forwarding state is valid if

it always delivers packets
to the correct destination

sufficient and necessary condition

Theorem a global forwarding state is valid if and only if

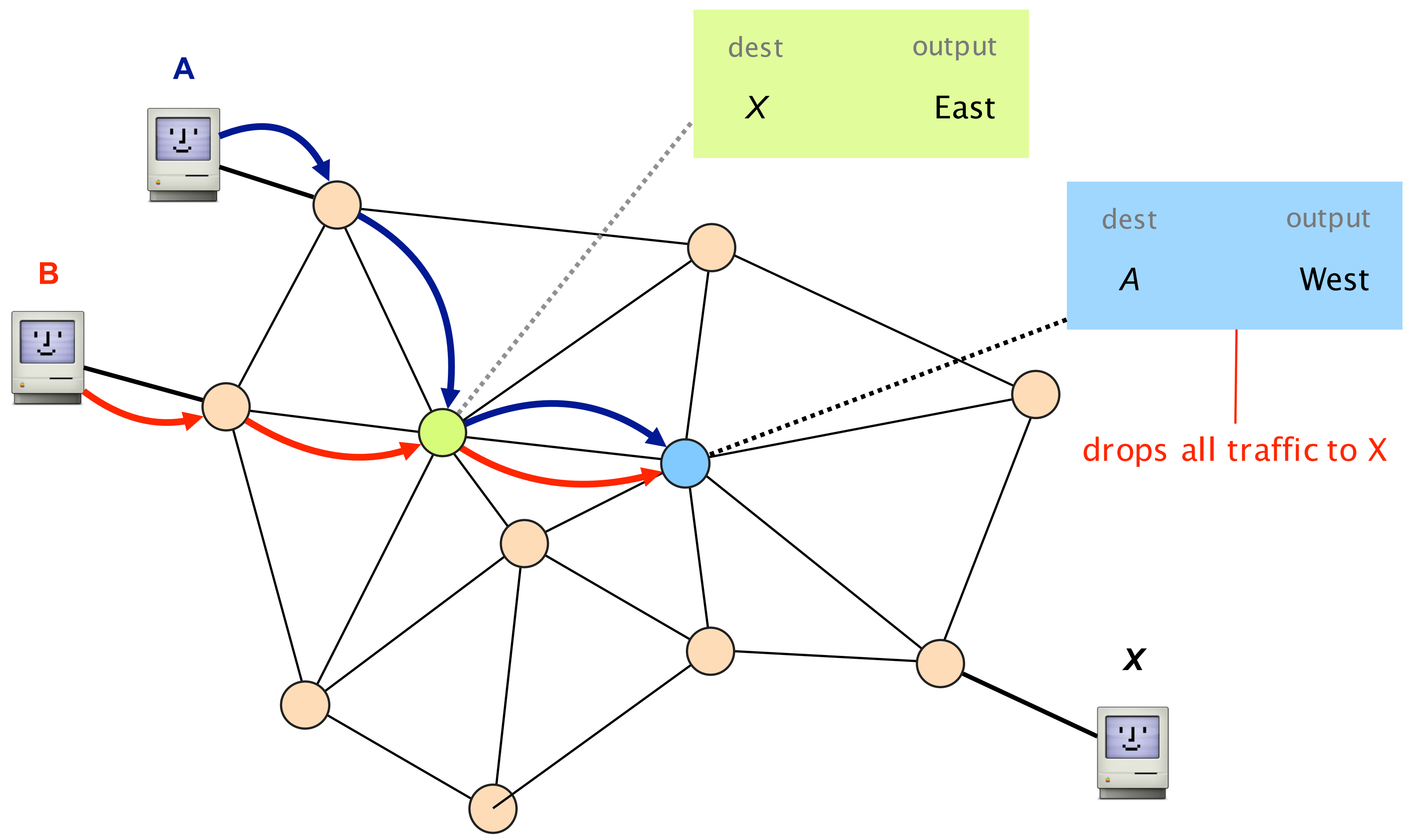
- there are no dead ends

  no outgoing port defined in the table

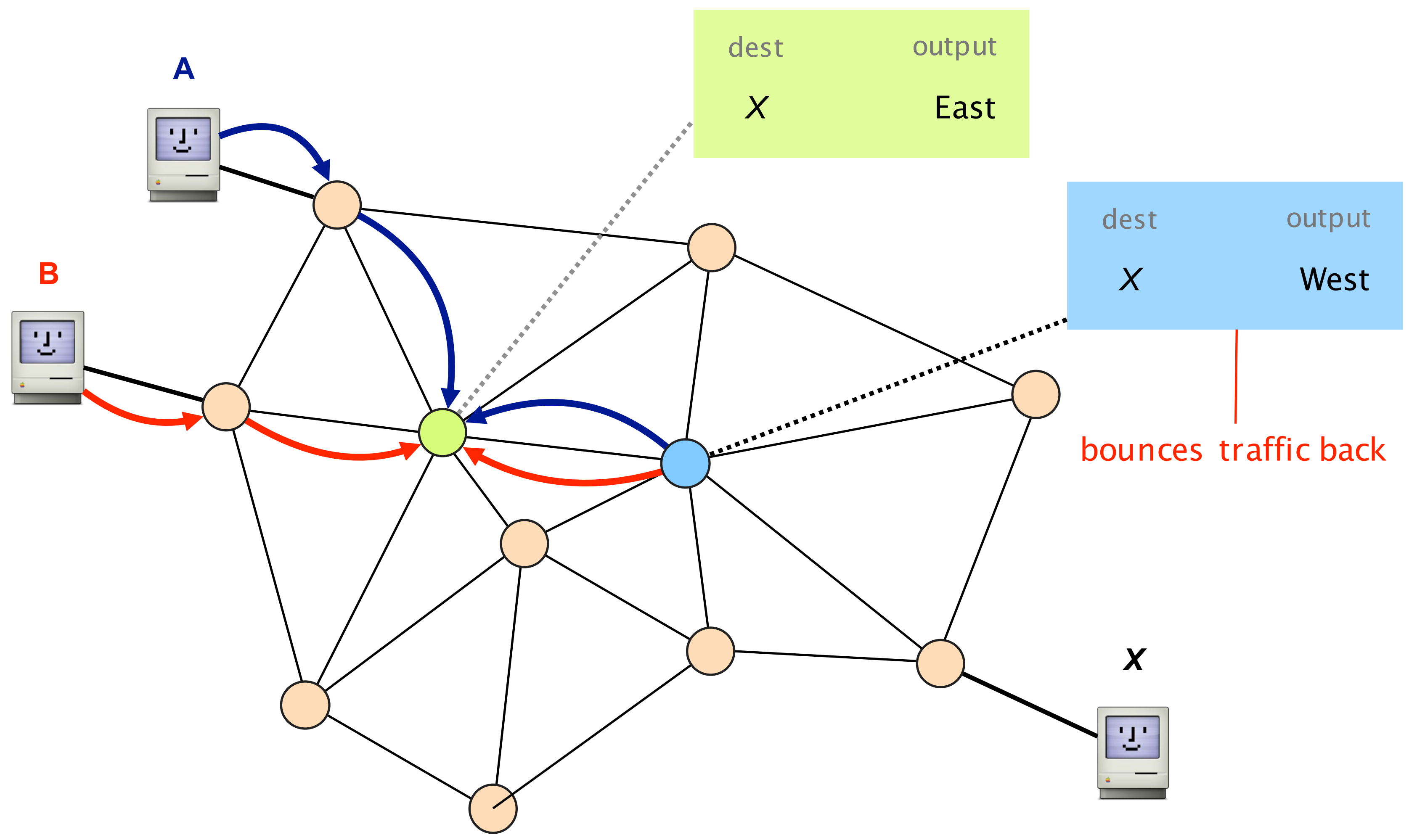
- there are no loops

  packets going around the same set of nodes

A global forwarding state is valid if and only if there are no dead ends



A global forwarding state is valid if and only if there are no forwarding loops



sufficient and necessary condition

Theorem

a global forwarding state is valid **if and only if**

- there are no dead ends

  i.e. no outgoing port defined in the table

- there are no loops

  i.e. packets going around the same set of nodes

Proving the **necessary** condition is **easy**

Theorem

If a routing state is valid
then there are no loops or dead-end

Proof

If you run into a dead-end or a loop
you'll never reach the destination
so the state cannot be correct (contradiction)

Proving the sufficient condition is more subtle

Theorem

If a routing state has no dead end and no loop
then it is valid

Proof

There is only a finite number of ports to visit

A packet can never enter a switch via the same port,
otherwise it is a loop (which does not exist by assumption)

As such, the packet must eventually reach the destination question 1 How do we verify that a forwarding state is valid?

question 2 How do we compute valid forwarding state?

question 1 How do we verify that a forwarding state is valid?

How do we compute valid forwarding state?

Verifying that a routing state is valid is easy

simple algorithm

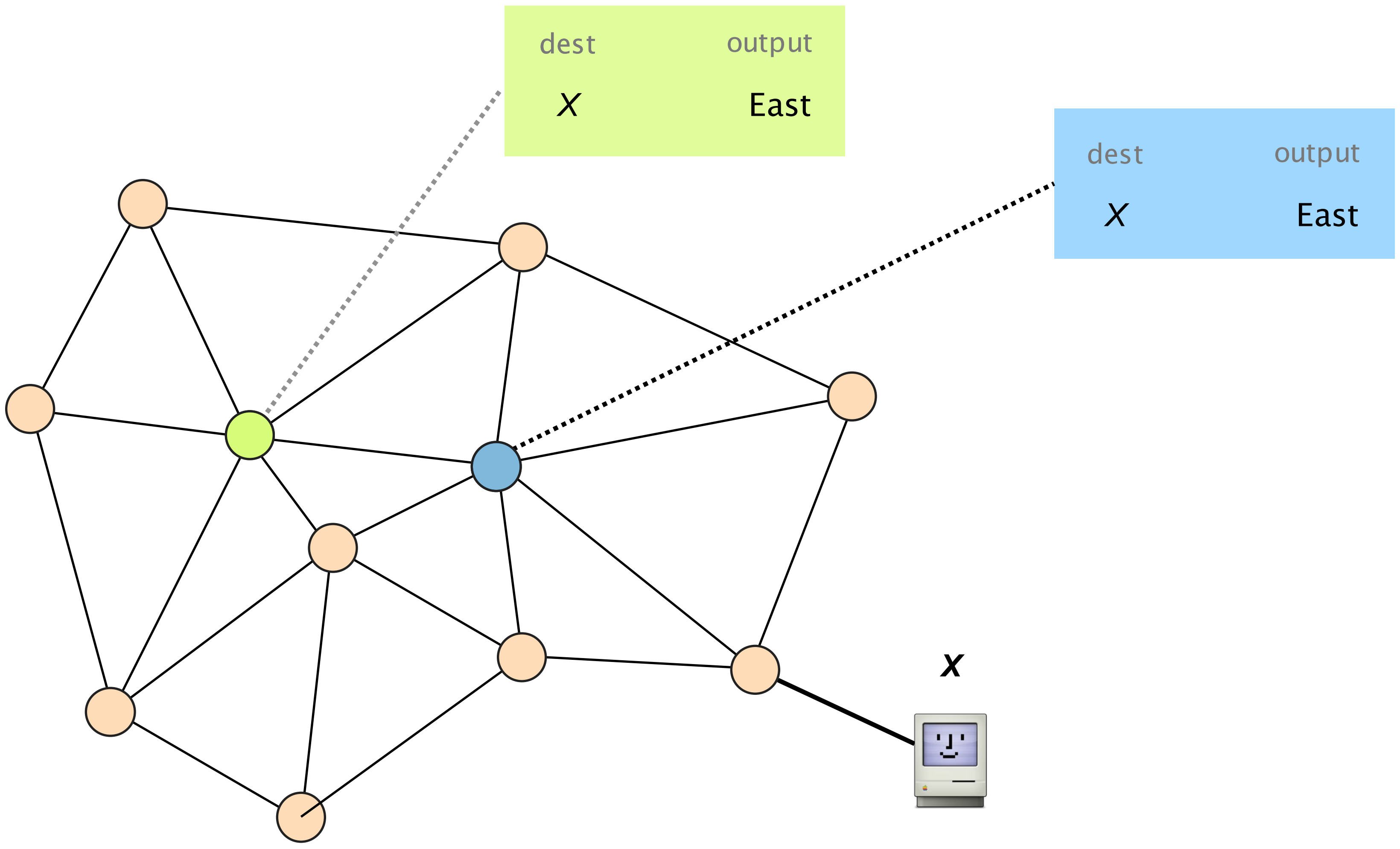
for one destination

Mark all outgoing ports with an arrow

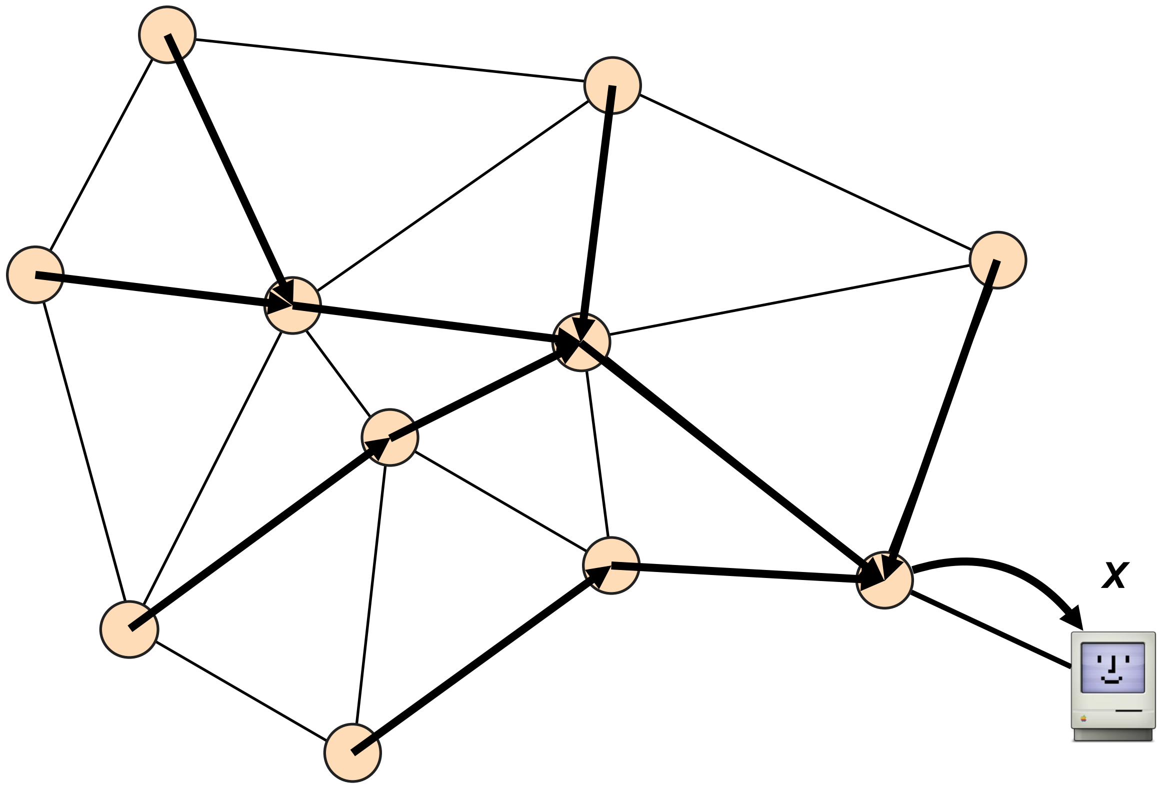
Eliminate all links with no arrow

State is valid *iff* the remaining graph
is a spanning-tree

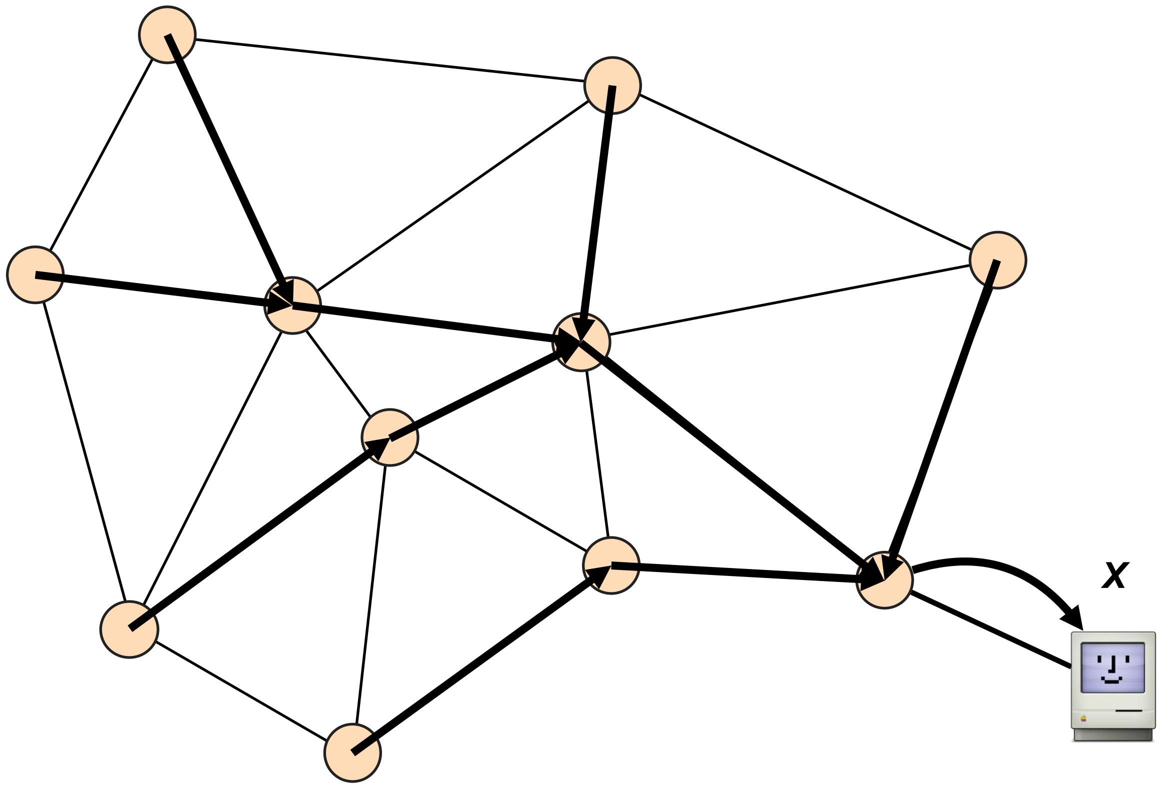
Given a graph with the corresponding forwarding state

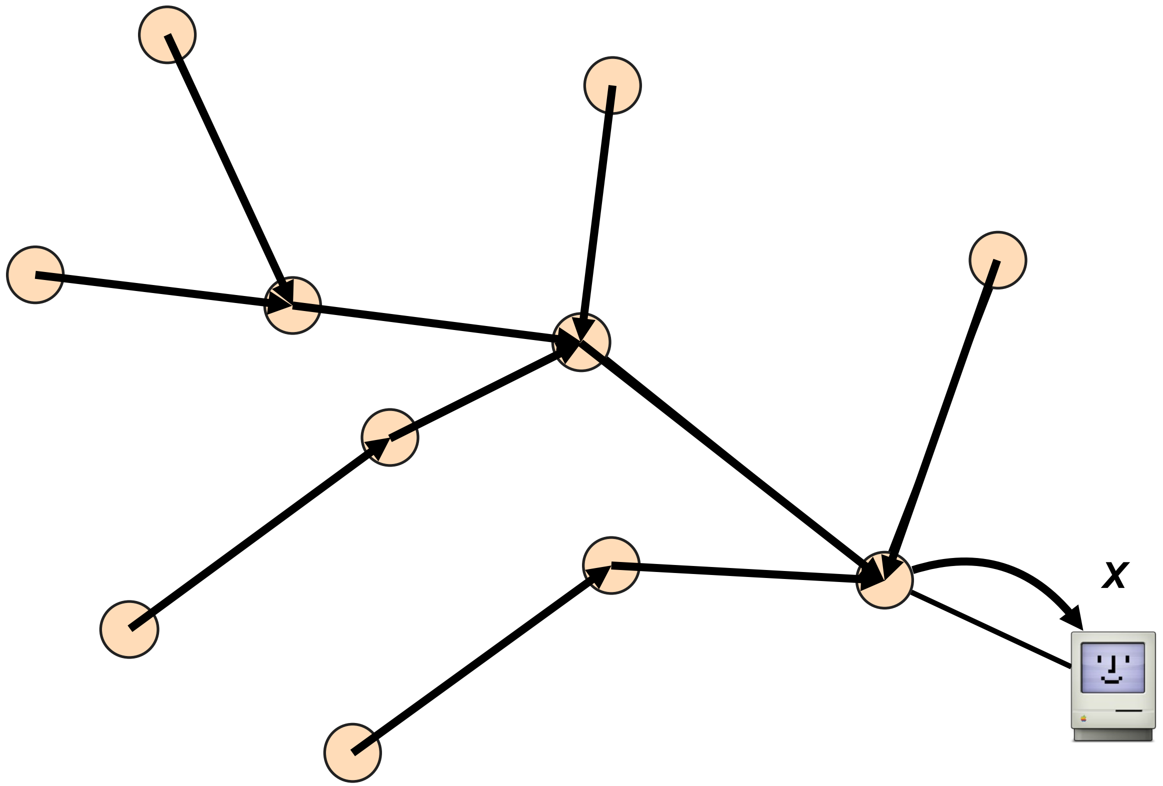


Mark all outgoing ports with an arrow



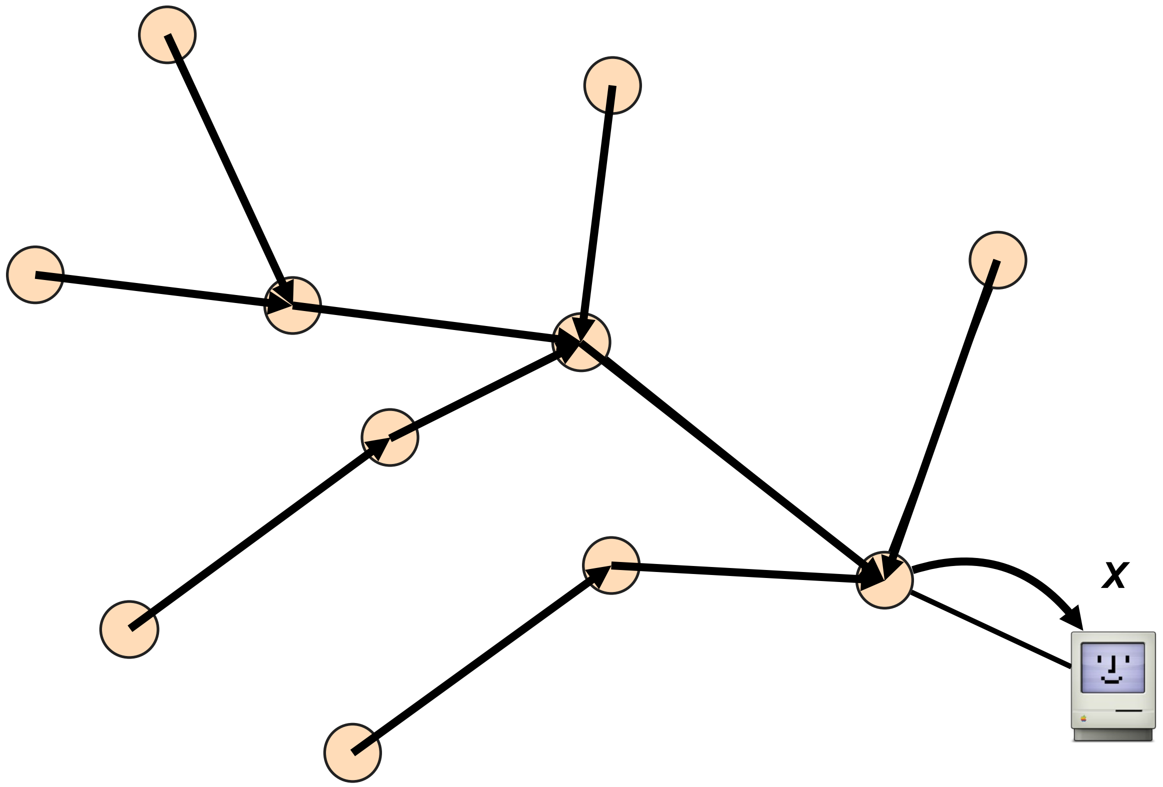
Eliminate all links with no arrow



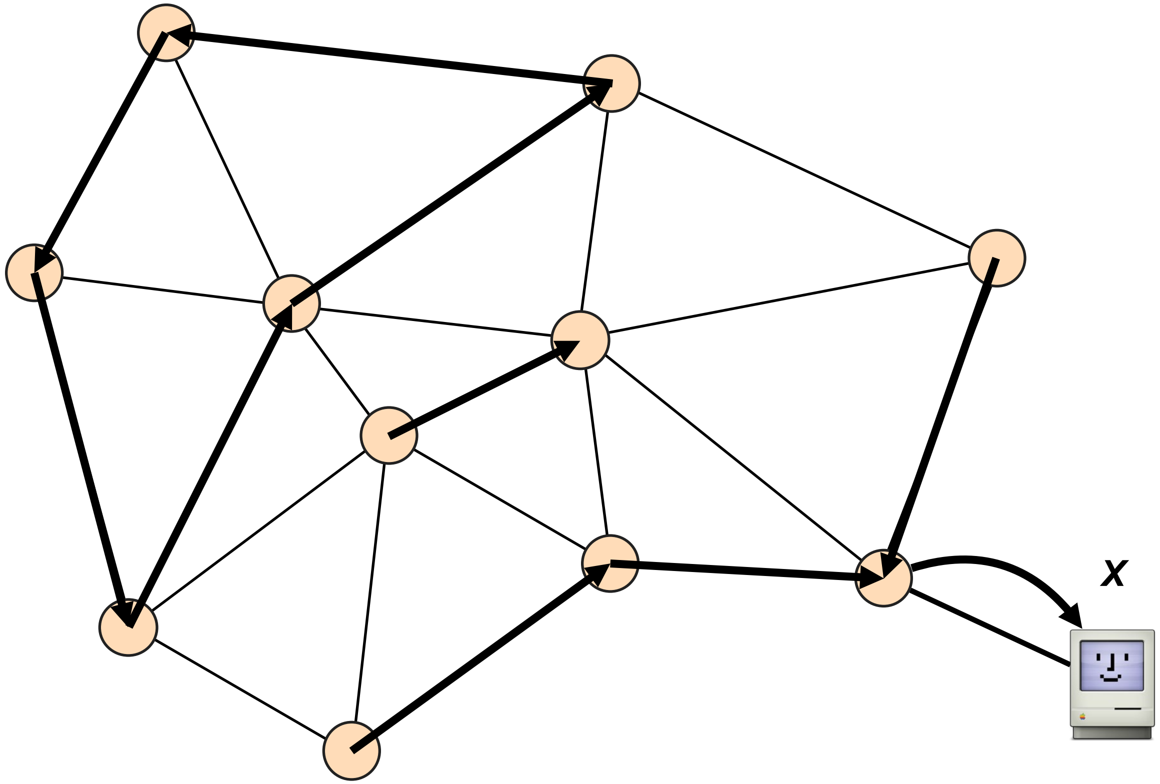

X

The result is a spanning tree.

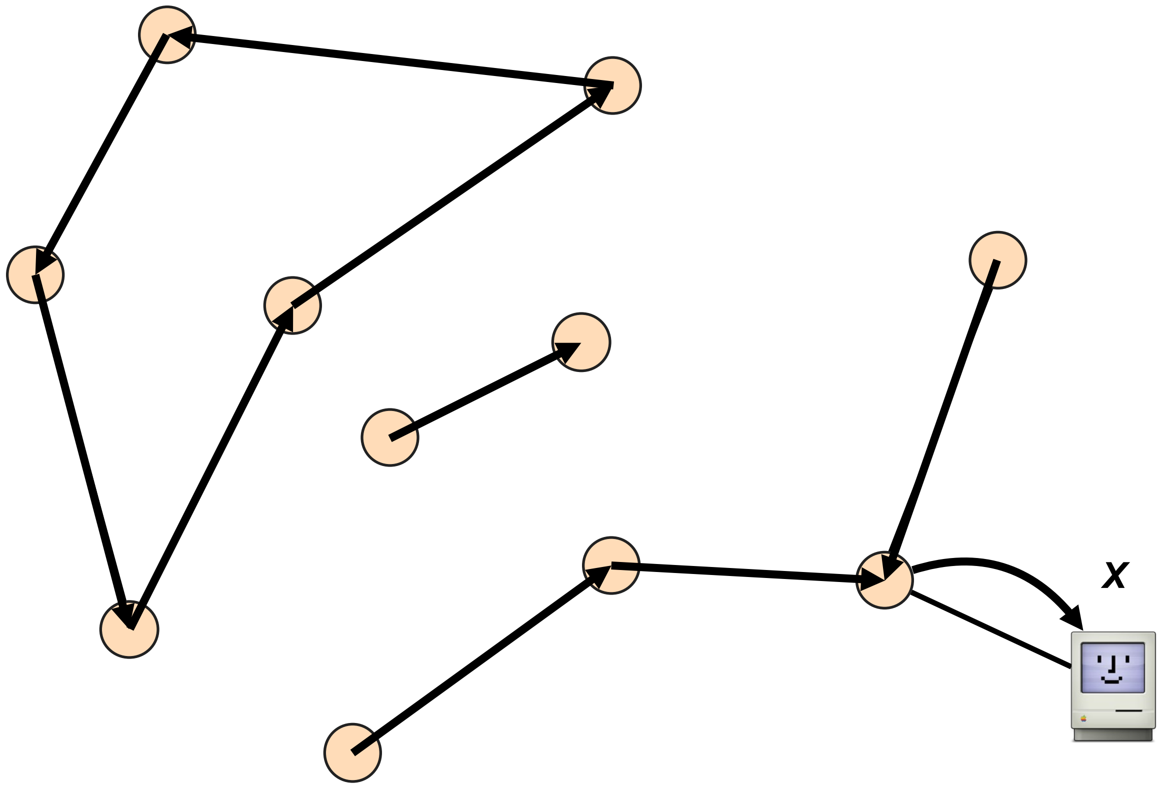
This is a valid routing state



Mark all outgoing ports with an arrow

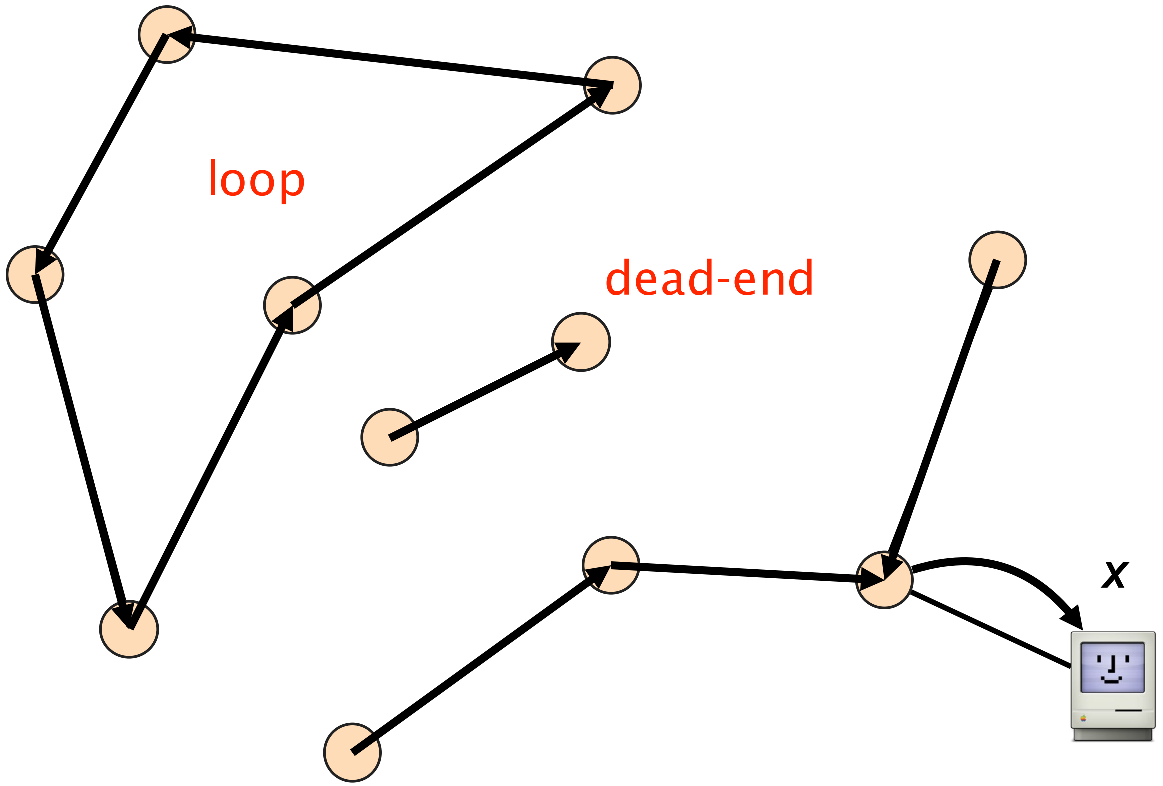


Eliminate all links with no arrow



The result is not a spanning-tree.

The routing state is not valid



How do we verify that a forwarding state is valid?

question 2

How do we compute valid forwarding state?

Producing valid routing state is harder

prevent dead ends

easy

prevent loops

hard

Producing valid routing state is harder

but doable

prevent dead ends	prevent loops
easy	hard

This is the question
you should focus on

Existing routing protocols differ in

how they avoid loops

prevent loops

hard

Essentially,
there are **three ways to compute** valid routing state

	Intuition	Example
#1	Use tree-like topologies	Spanning-tree
#2	Rely on a global network view	Link-State SDN
#3	Rely on distributed computation	Distance-Vector BGP

Communication Networks and Internet Technology

Short Recap on this weeks lecture

Communication Networks and Internet Technology

Part 1: General overview

What is a network made of?

How is it shared?

How is it organized?

How does communication happen?

How do we characterize it?

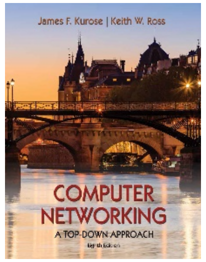
Communication Networks and Internet Technology

Part 2: Concepts

routing	reliable delivery
How do you guide IP packets from a source to destination?	How do you ensure reliable transport on top of best-effort delivery?

Reading: Book Kurose & Ross

Class textbook:
Computer Networking: A Top-Down Approach (8th ed.)
J.F. Kurose, K.W. Ross
Pearson, 2020
http://gaia.cs.umass.edu/kurose_ross



- Week 01
  - 1.1 (The Internet), 1.2 (The Network Edge), 1.3 (The Network Core) and 1.5 (Protocol Layers)
- Week 02
  - 1.4 (Delay, Loss and Throughput), 1.5 (Protocol Layers), 4.2 (What's Inside a Router)

Check Your Knowledge

PROBLEM SOLVING HOME | TRY A RANDOM PROBLEM

INTERACTIVE END-OF-CHAPTER EXERCISES

Supplement to Computer Networking: A Top Down Approach 8th Edition

"Tell me and I forget. Show me and I remember. Involve me and I understand." Chinese proverb

The links below will take you to end-of-chapter exercises where you'll be presented with an exercise whose solution can then be displayed (hopefully *after* you've solved the exercise yourself!). Each of the exercises below is similar to an end-of-chapter problem in the text. Most importantly, you can keep generating new instances of each exercise (and hopefully solving each one!) until you've mastered the material.

You may be interested in other supplemental material (online lectures, powerpoint slides, review questions, Wireshark labs) for our book, available here.

This page replaces the earlier interactive problems page, and includes a number of new problems. We're actively adding new problems here in the summer of 2020. If you've got any comments or suggestions - let us know at kurose@cs.umass.edu

CHAPTER 1: INTRODUCTION

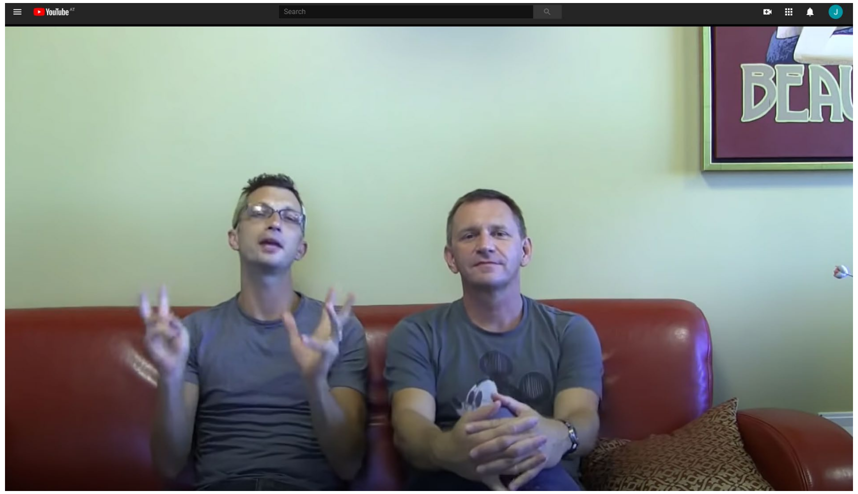
- Circuit Switching
- Quantitative Comparison of Packet Switching and Circuit Switching (similar to Chapter 1, P8, P9)
- Car - Caravan Analogy
- One-hop Transmission Delay (similar to example on pg. 37)
- Queuing Delay
- End-to-End Delay (similar to Chapter 1, P10)
- End-to-End Throughput (similar to Chapter 1, P20, and Figure 1.20)
- The IP Stack and Protocol Layering

http://gaia.cs.umass.edu/kurose_ross/interactive/

Moving Ahead or Hearing Topics from a Different Voice

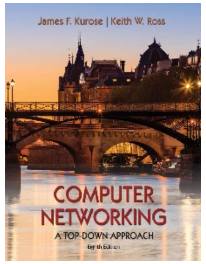
- Phil Levis and Nick McKeown
  Stanford

  [CS144 Introduction to Computer Networking Fall 2016 on YouTube]



- Jim Kurose – UMass
  http://gaia.cs.umass.edu/kurose_ross/videos/1/

Class textbook:
Computer Networking: A Top-Down Approach (8th ed.)
J.F. Kurose, K.W. Ross
Pearson, 2020
http://gaia.cs.umass.edu/kurose_ross



No lecture tomorrow 13:15-14:00h