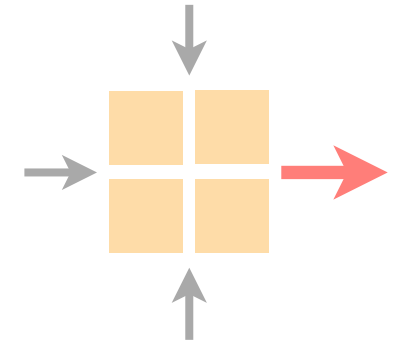


# Communication Networks

Spring 2019



Laurent Vanbever

[nsg.ee.ethz.ch](http://nsg.ee.ethz.ch)

ETH Zürich (D-ITET)

February 25 2019

Materials inspired from Scott Shenker & Jennifer Rexford

Last week on  
**Communication Networks**

# Communication Networks

## Part 1: General overview



### #1 What is a network made of?

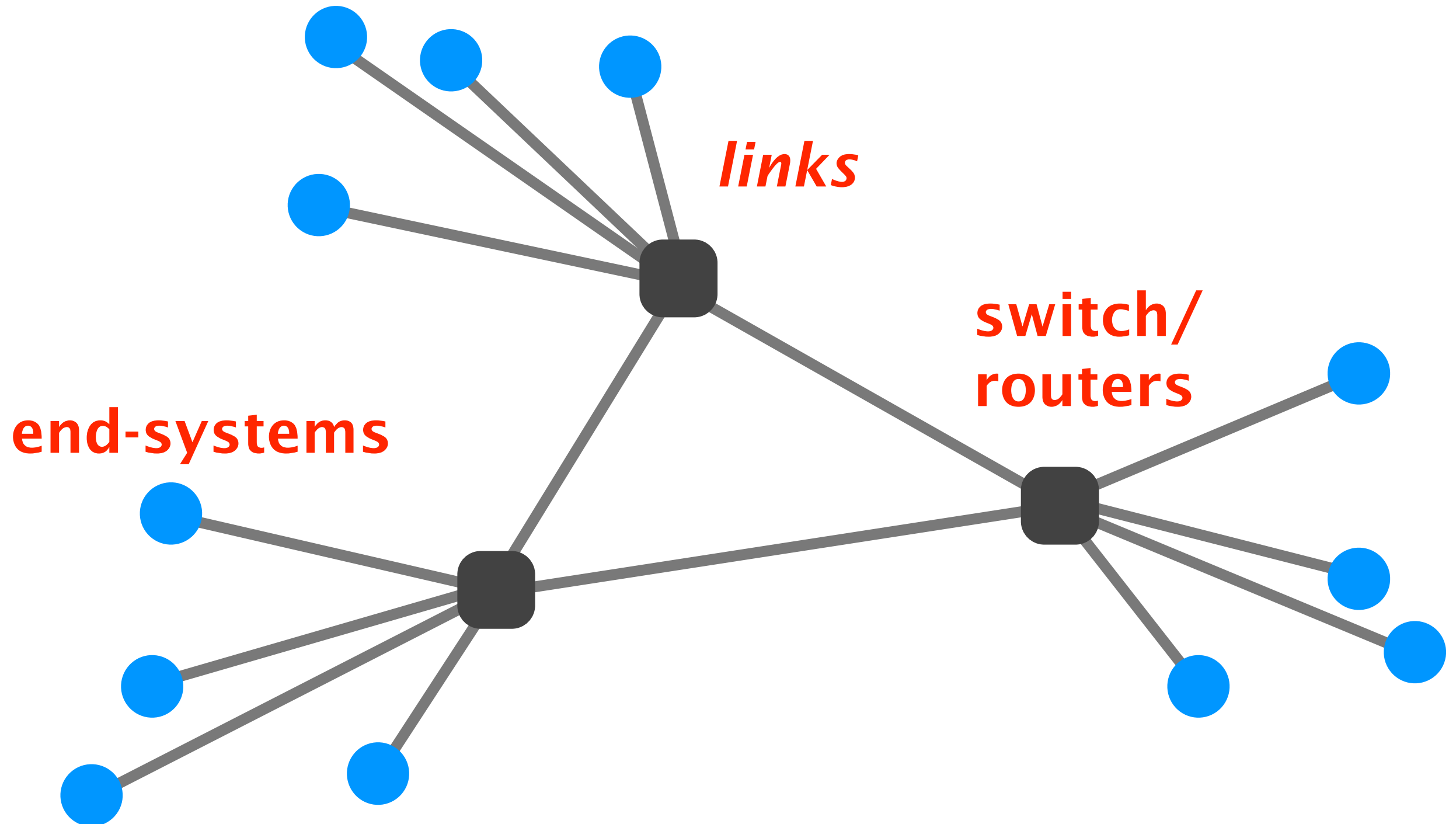
How is it shared?

How is it organized?

How does communication happen?

How do we characterize it?

Networks are composed of three basic components



# Communication Networks

## 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?

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

In practice, the approaches are implemented using  
circuit-switching or packet-switching



Reservation

On-demand

implem.

circuit-switching

packet-switching

# Pros and cons of circuit switching

## advantages

predictable performance

simple & fast switching  
once circuit established

## disadvantages

inefficient if traffic is bursty or short

complex circuit setup/teardown  
which adds delays to transfer

requires new circuit upon failure



# Pros and cons of packet switching

## advantages

efficient use of resources

simpler to implement  
than circuit switching

route around trouble

## disadvantages

unpredictable performance

requires buffer management and  
congestion control

# Communication Networks

## Part 1: General overview



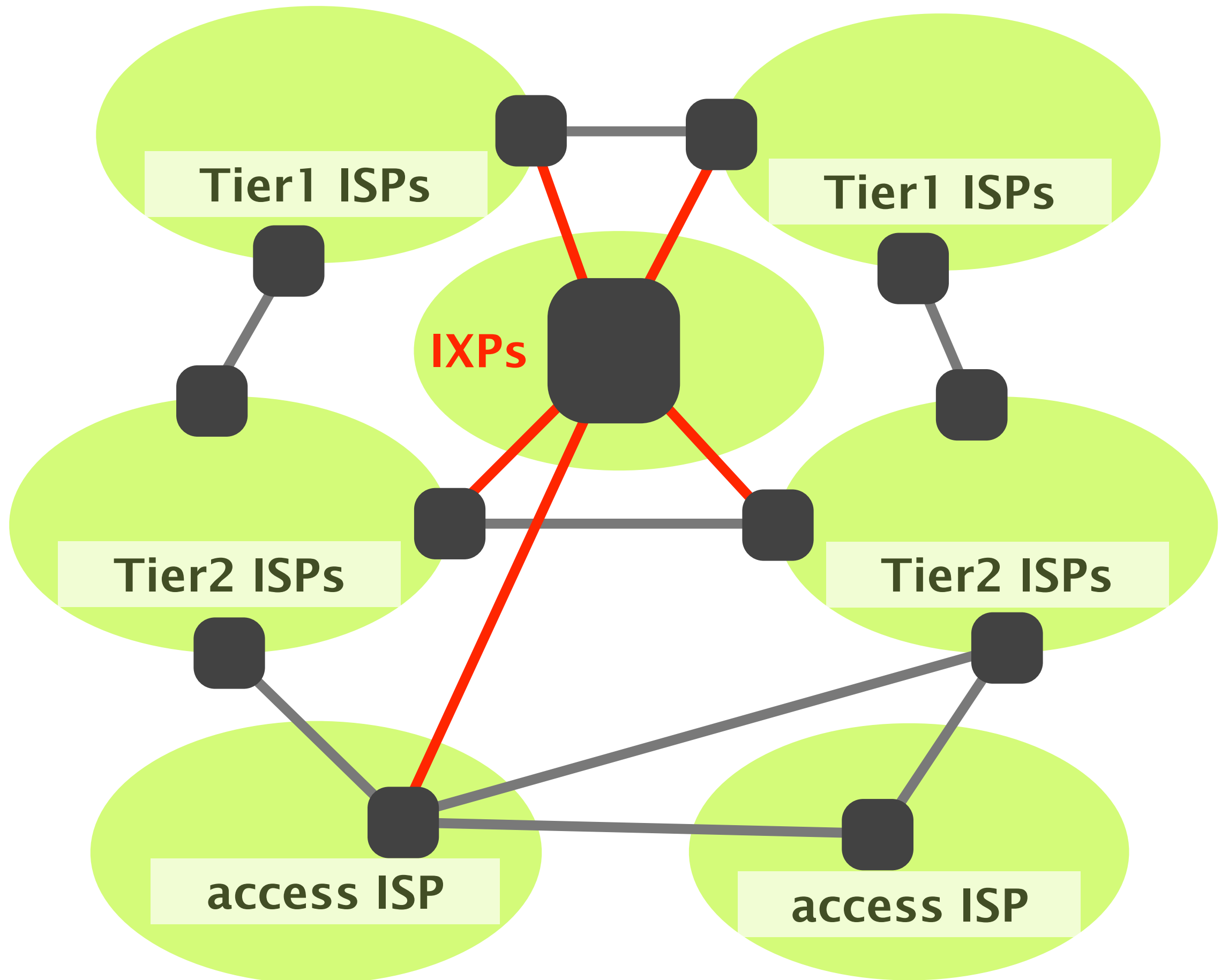
What is a network made of?

How is it shared?

#3 **How is it organized?**

How does communication happen?

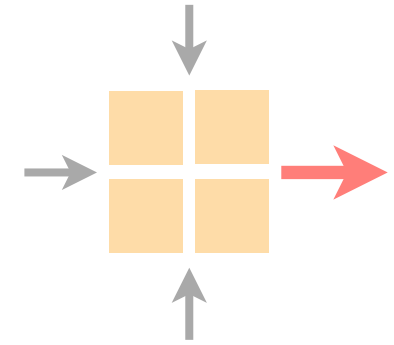
How do we characterize it?



**This week on**  
**Communication Networks**

# Communication Networks

## Part 1: General overview



What is a network made of?

How is it shared?

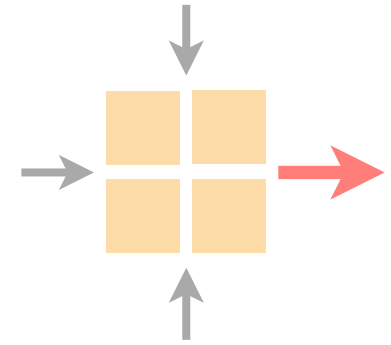
How is it organized?

#4      How does communication happen?

#5      How do we characterize it?

# Communication Networks

## 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?

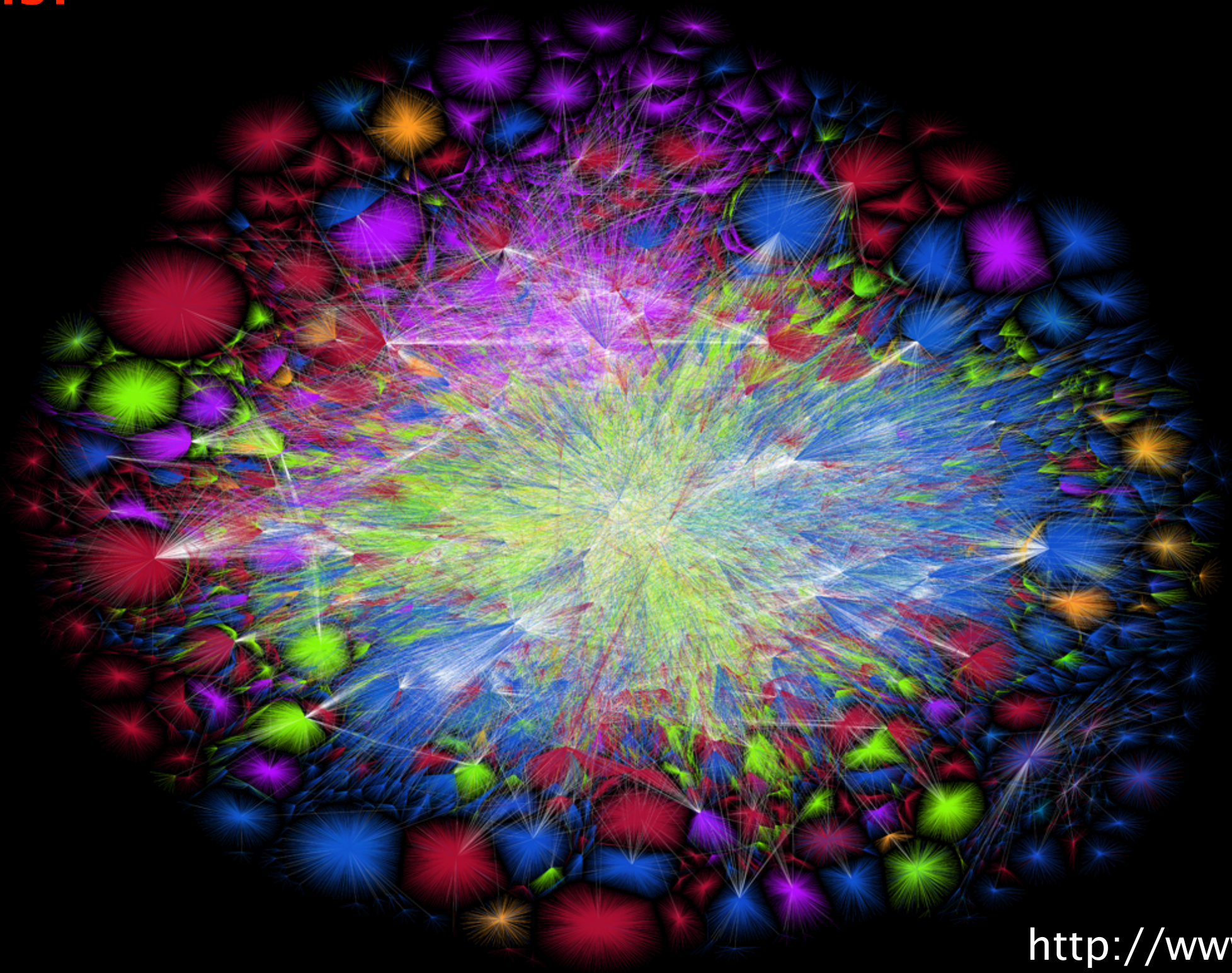
The Internet should allow

**processes on different hosts**  
to exchange data

everything else is just commentary...

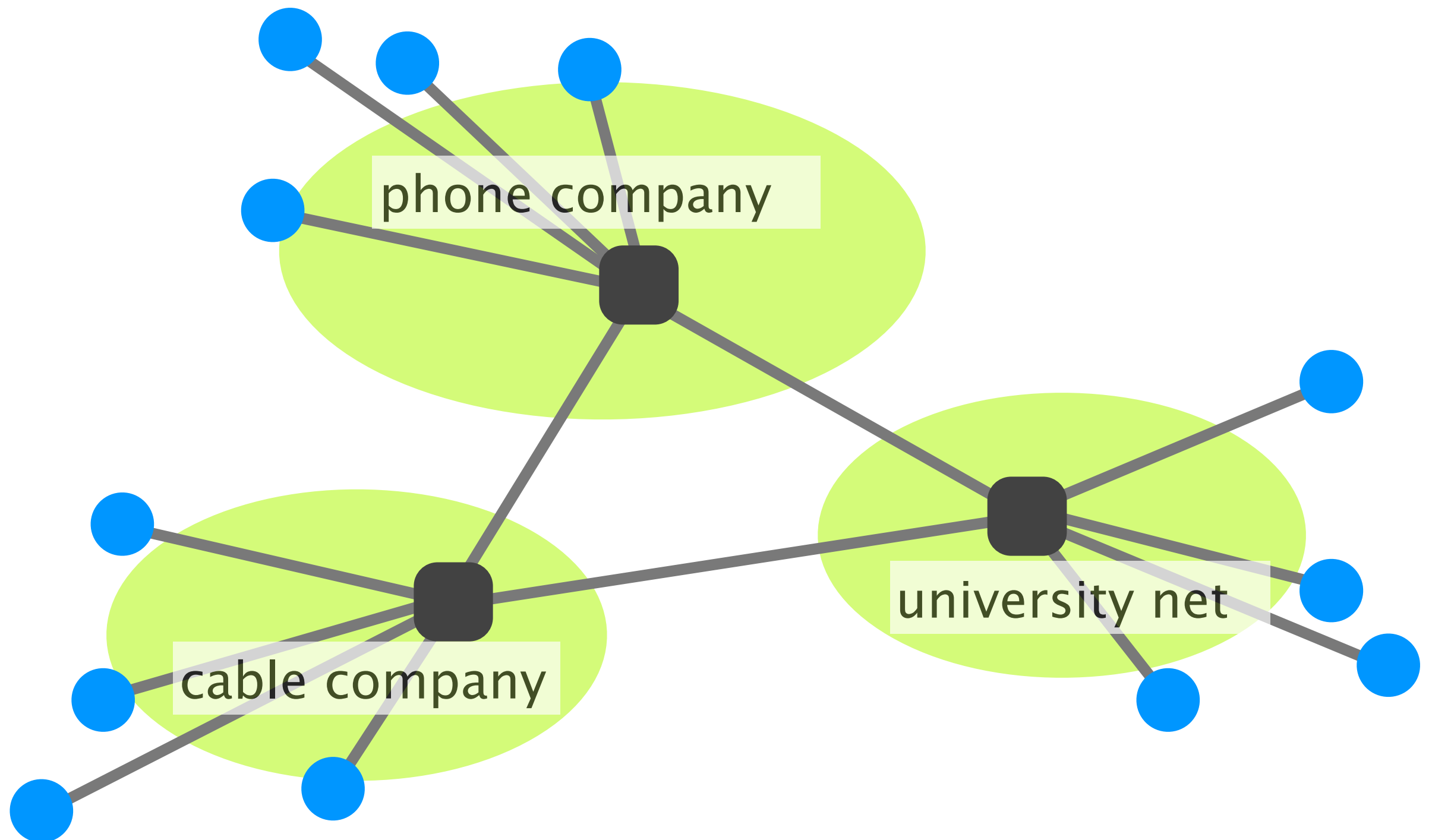


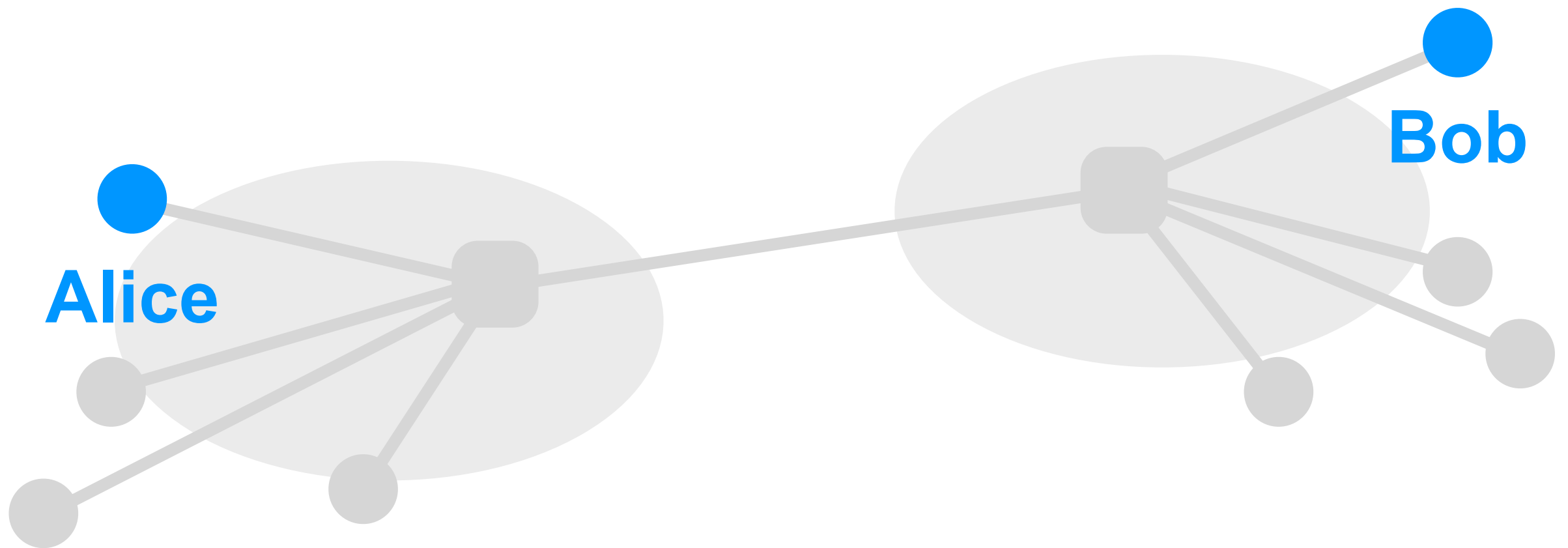
How do you exchange data in a network as complex  
as **this?**



<http://www.opte.org>

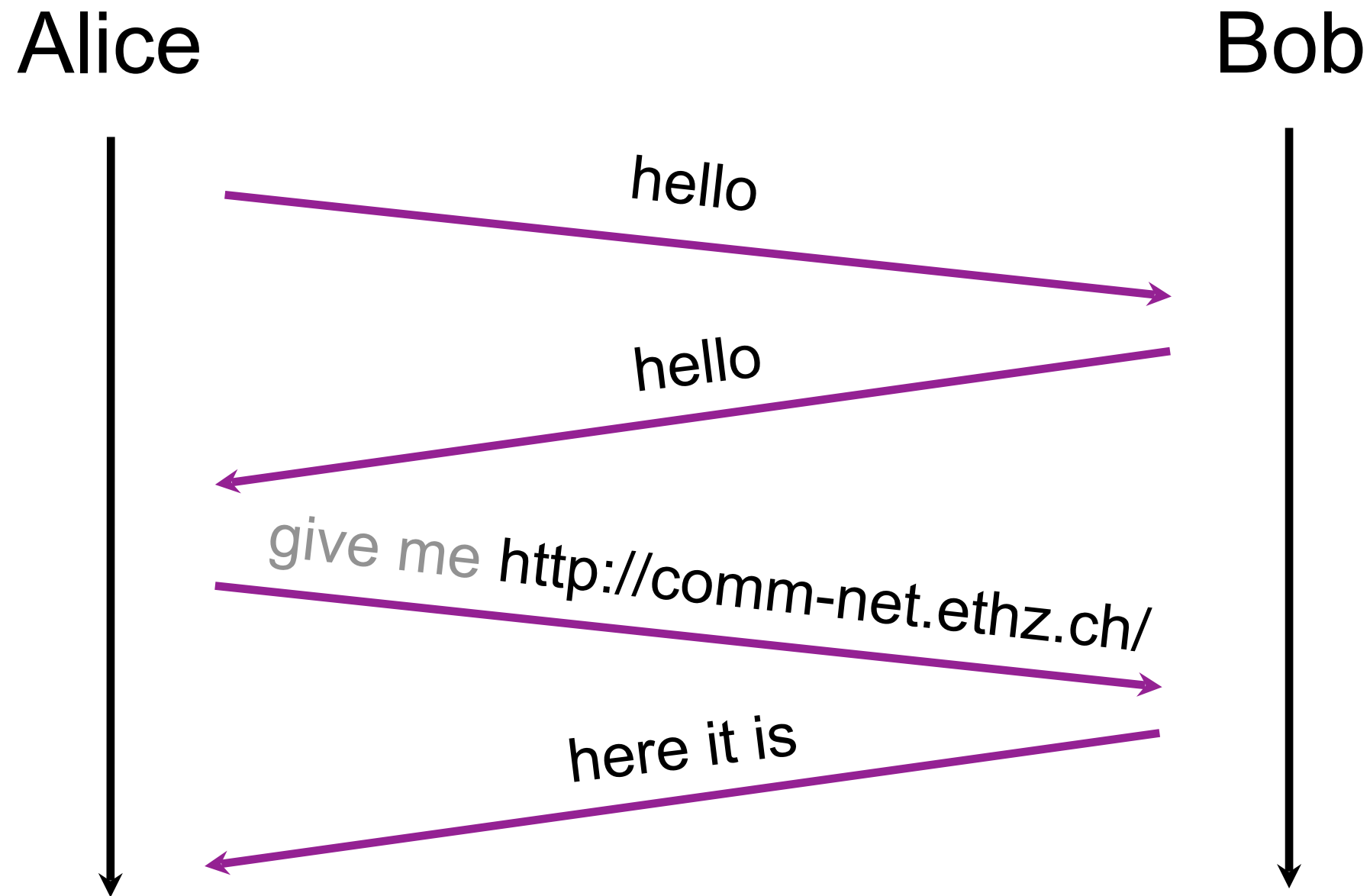






To exchange data, Alice and Bob use  
a set of network protocols

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

### WoW server

```
while (...) {  
    message = ...;  
    send(message, ...);  
}
```



**Alice**

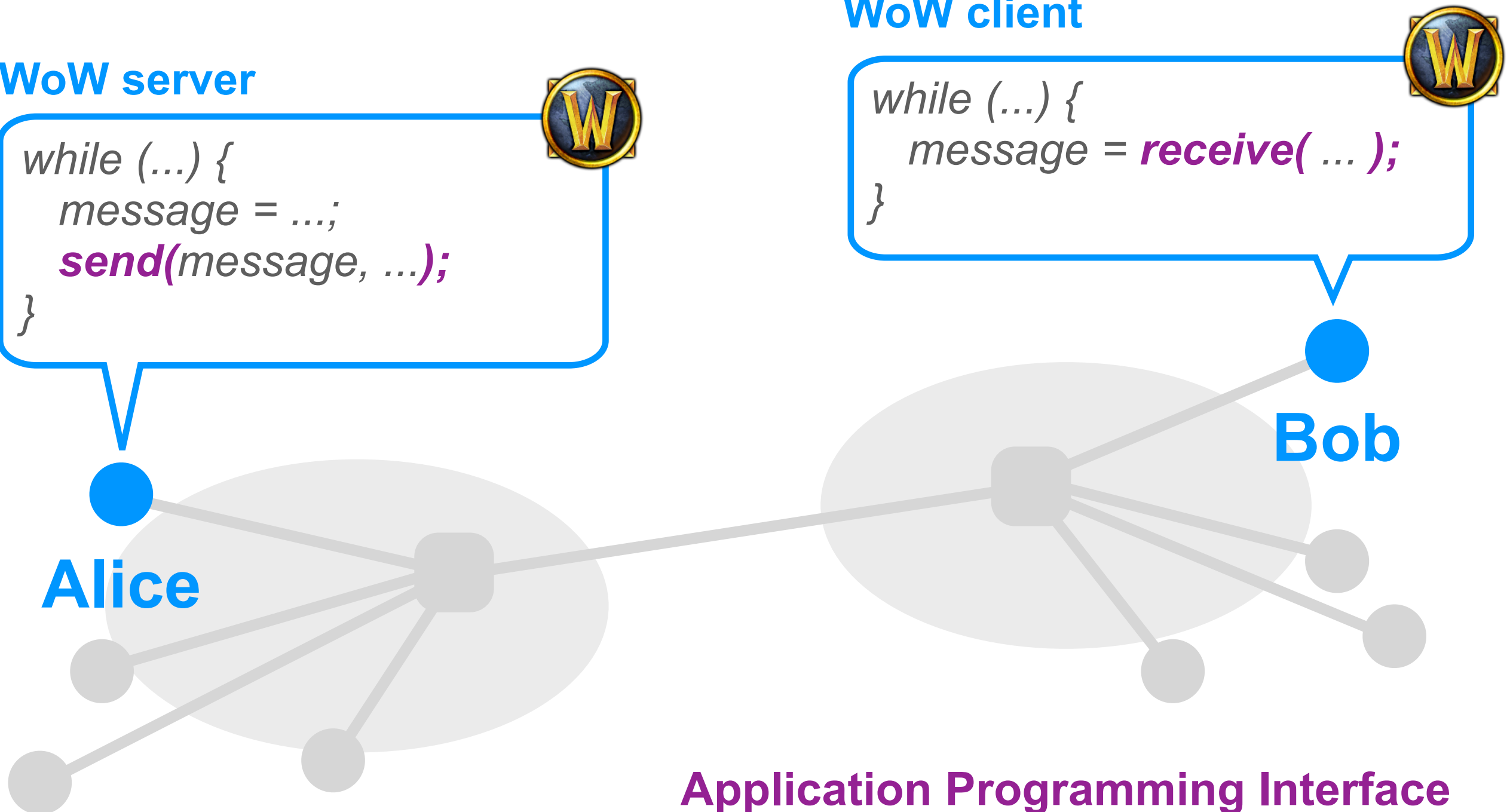
### WoW client

```
while (...) {  
    message = receive( ... );  
}
```



**Bob**

**Application Programming Interface**



In practice, there exists **a lot** of network protocols.  
How does the Internet organize **this**?



HOW STANDARDS PROLIFERATE:  
(SEE: A/C CHARGERS, CHARACTER ENCODINGS, INSTANT MESSAGING, ETC.)

SITUATION:  
THERE ARE  
14 COMPETING  
STANDARDS.

14?! RIDICULOUS!  
WE NEED TO DEVELOP  
ONE UNIVERSAL STANDARD  
THAT COVERS EVERYONE'S  
USE CASES.



SOON:

SITUATION:  
THERE ARE  
15 COMPETING  
STANDARDS.



# 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

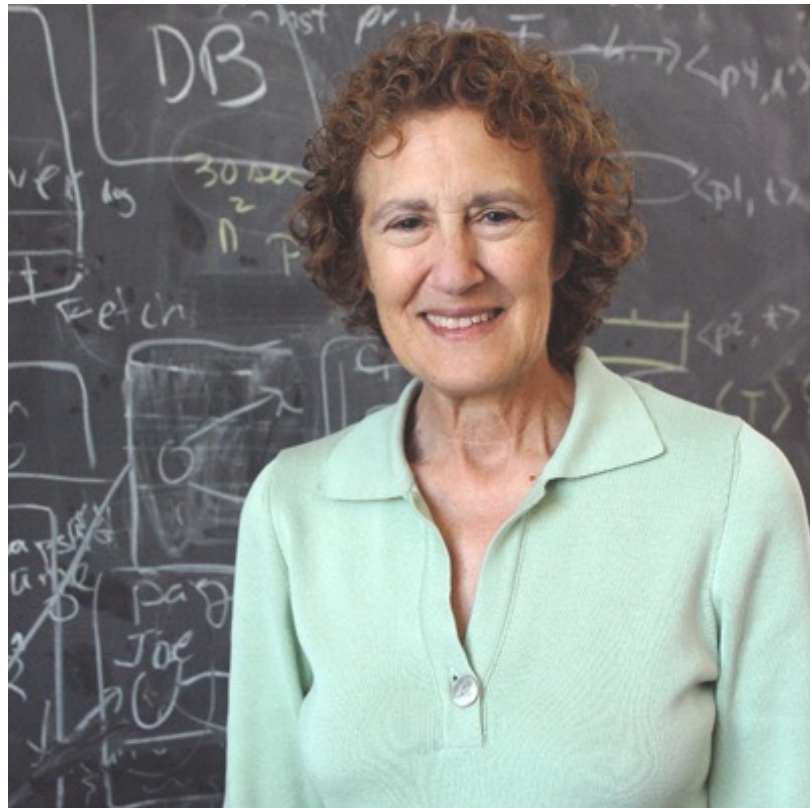


Photo: Donna Coveney

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...

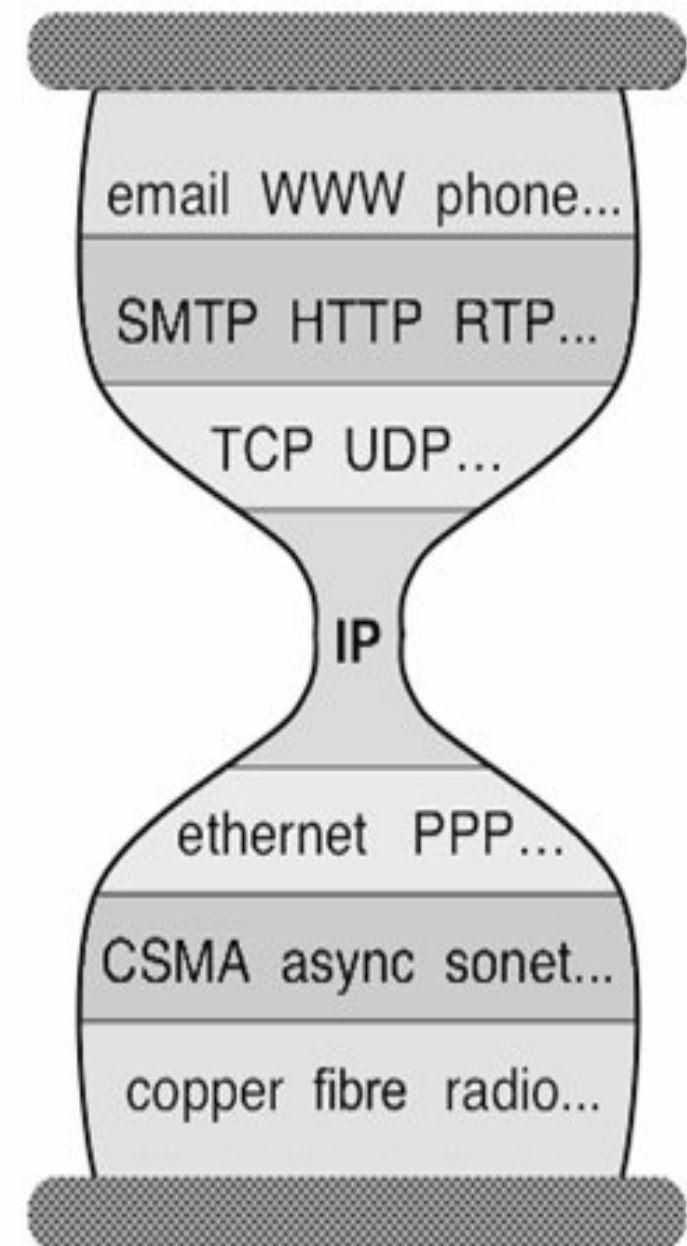
**Best-effort global packet delivery**

...built on...

**Best-effort local packet delivery**

...built on...

**Physical transfer of bits**



Each layer has a unit of **data**

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

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

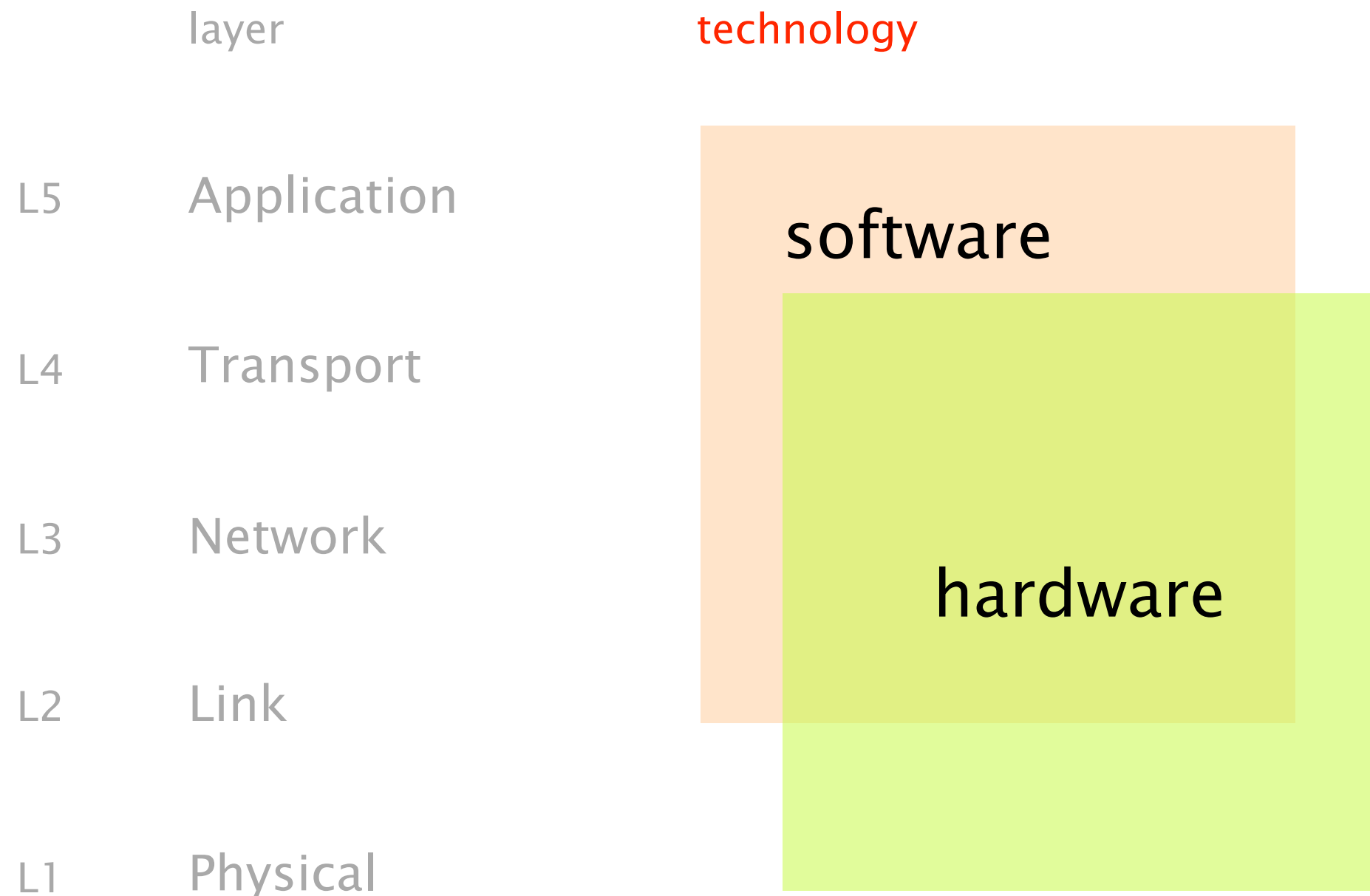
	layer	protocol
L5	Application	HTTP, SMTP, FTP, SIP, ...
L4	Transport	TCP, UDP, SCTP
L3	Network	IP
L2	Link	Ethernet, Wifi, (A/V)DSL, WiMAX, LTE, ...
L1	Physical	Twisted pair, fiber, coaxial cable, ...



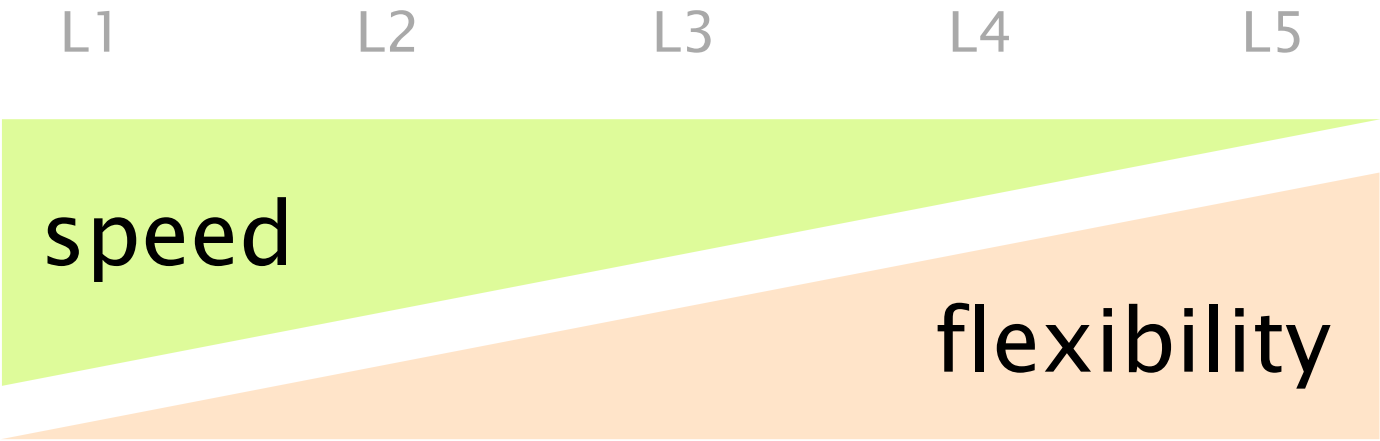
# The Internet Protocol (IP) acts as an unifying, network, layer

	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 is implemented with different protocols  
and technologies

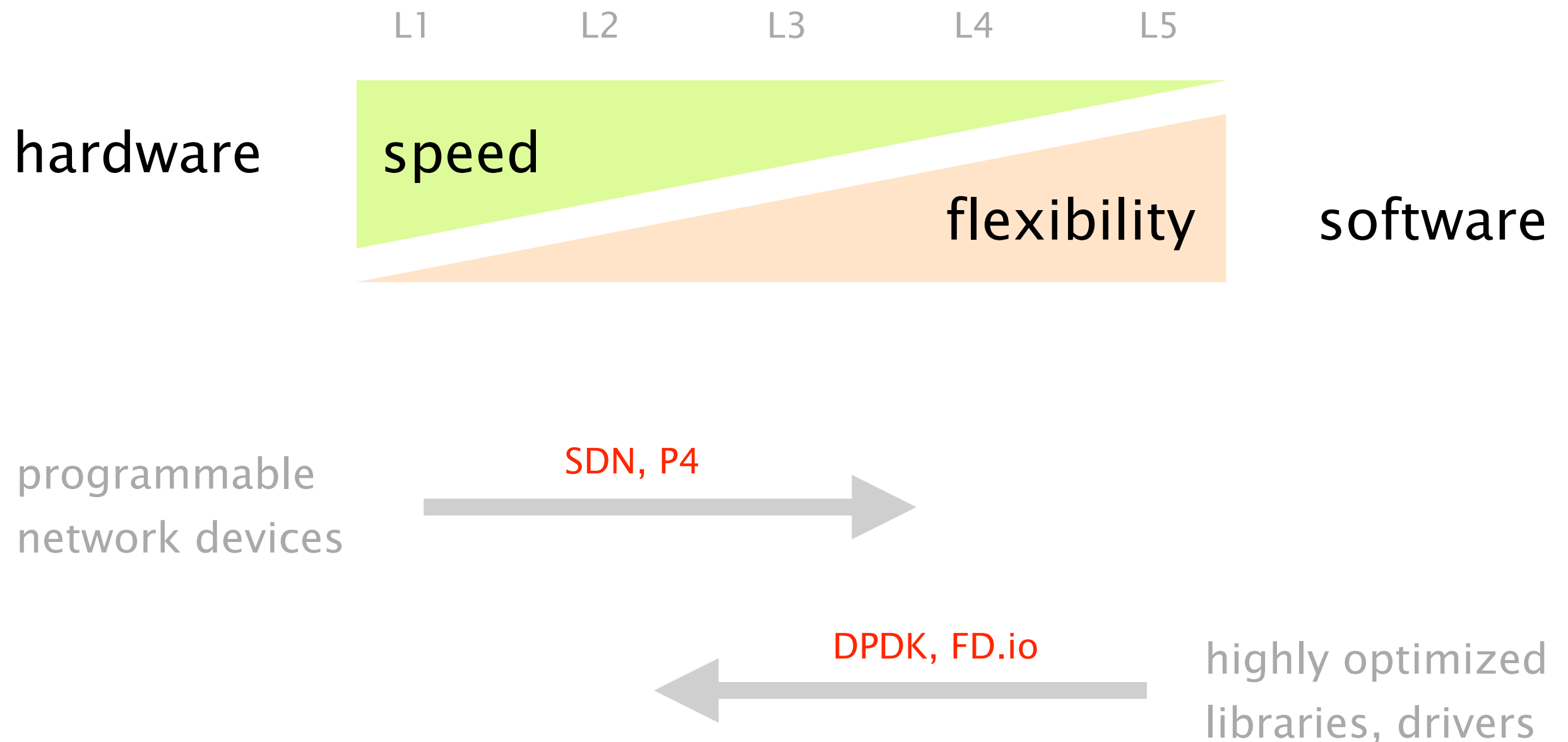


hardware



software

# Software and hardware advancements



## SHARE



SHARE  
1123



TWEET



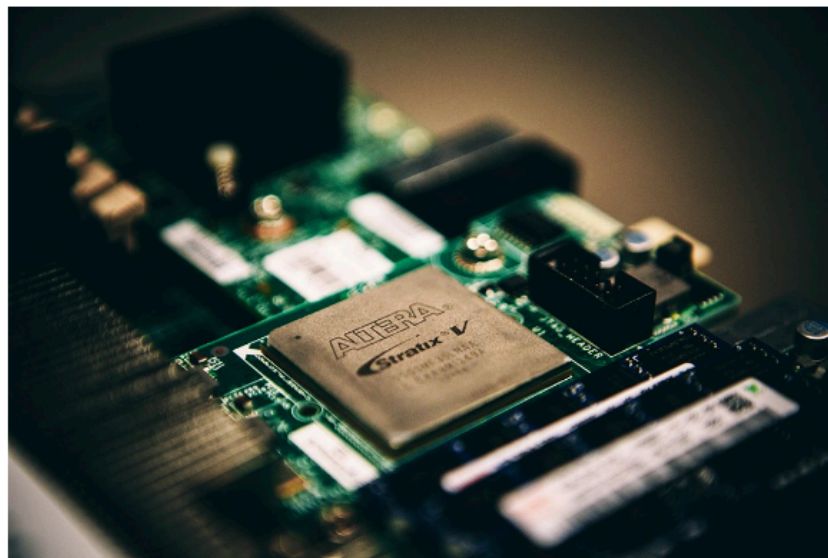
COMMENT  
103



EMAIL

ROBERT MCMILLAN BUSINESS 06.16.14 6:30 AM

# MICROSOFT SUPERCHARGES BING SEARCH WITH PROGRAMMABLE CHIPS



Microsoft

## 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



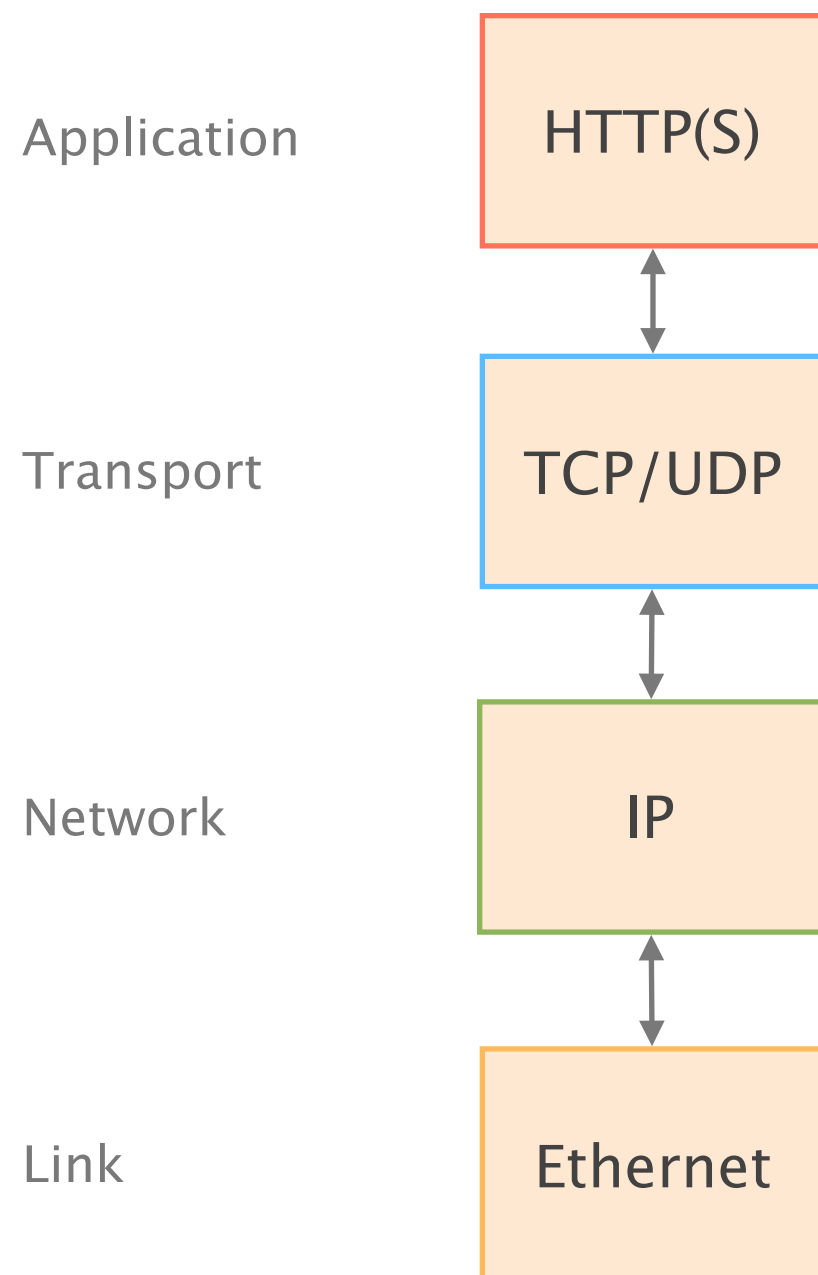
MORE STORIES

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

<https://www.wired.com/2014/06/microsoft-fpga/>  
them with a new kind of computer processor.

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



your laptop

Application

HTTP(S)

Transport

TCP/UDP

Network

IP

Link

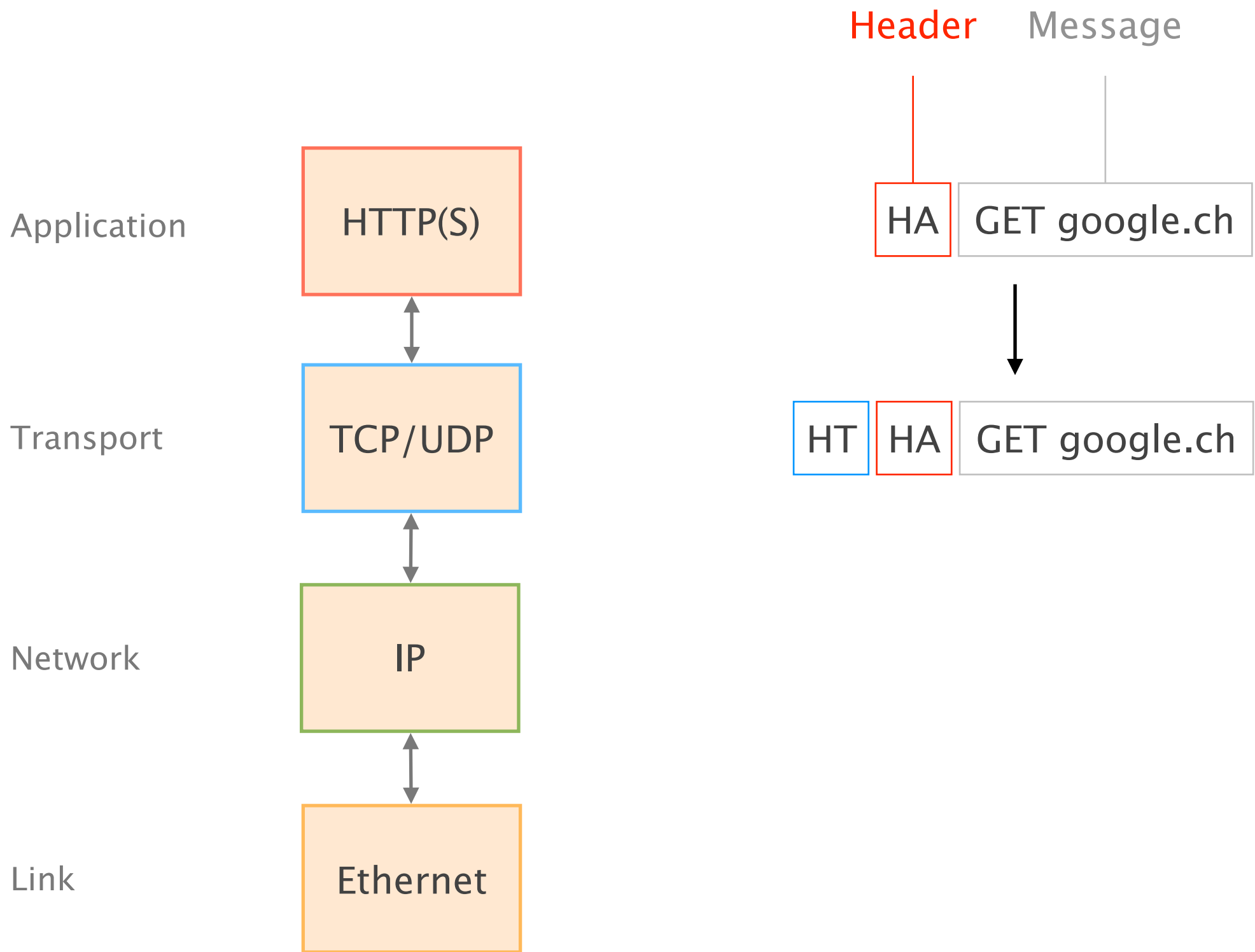
Ethernet

Header

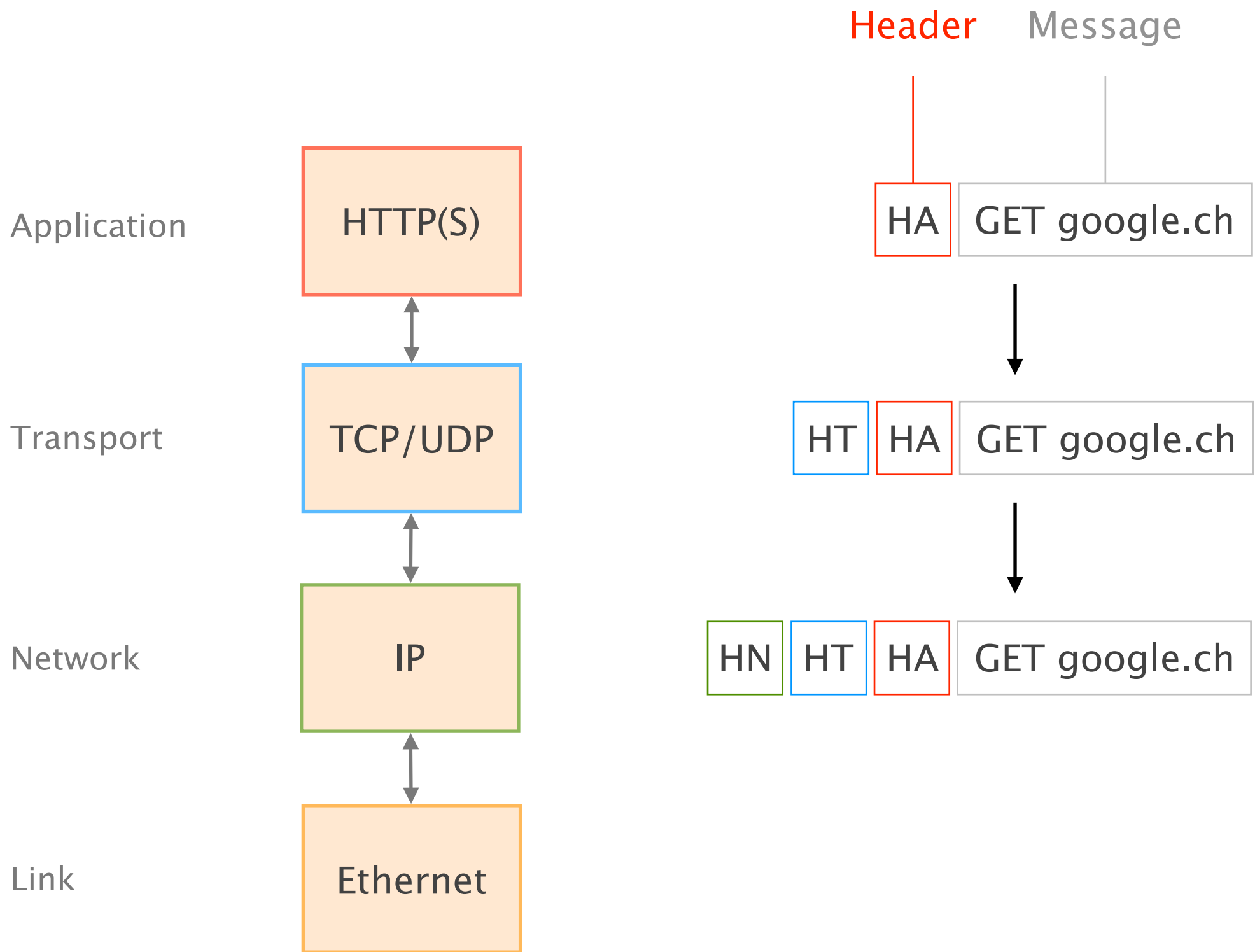
Message

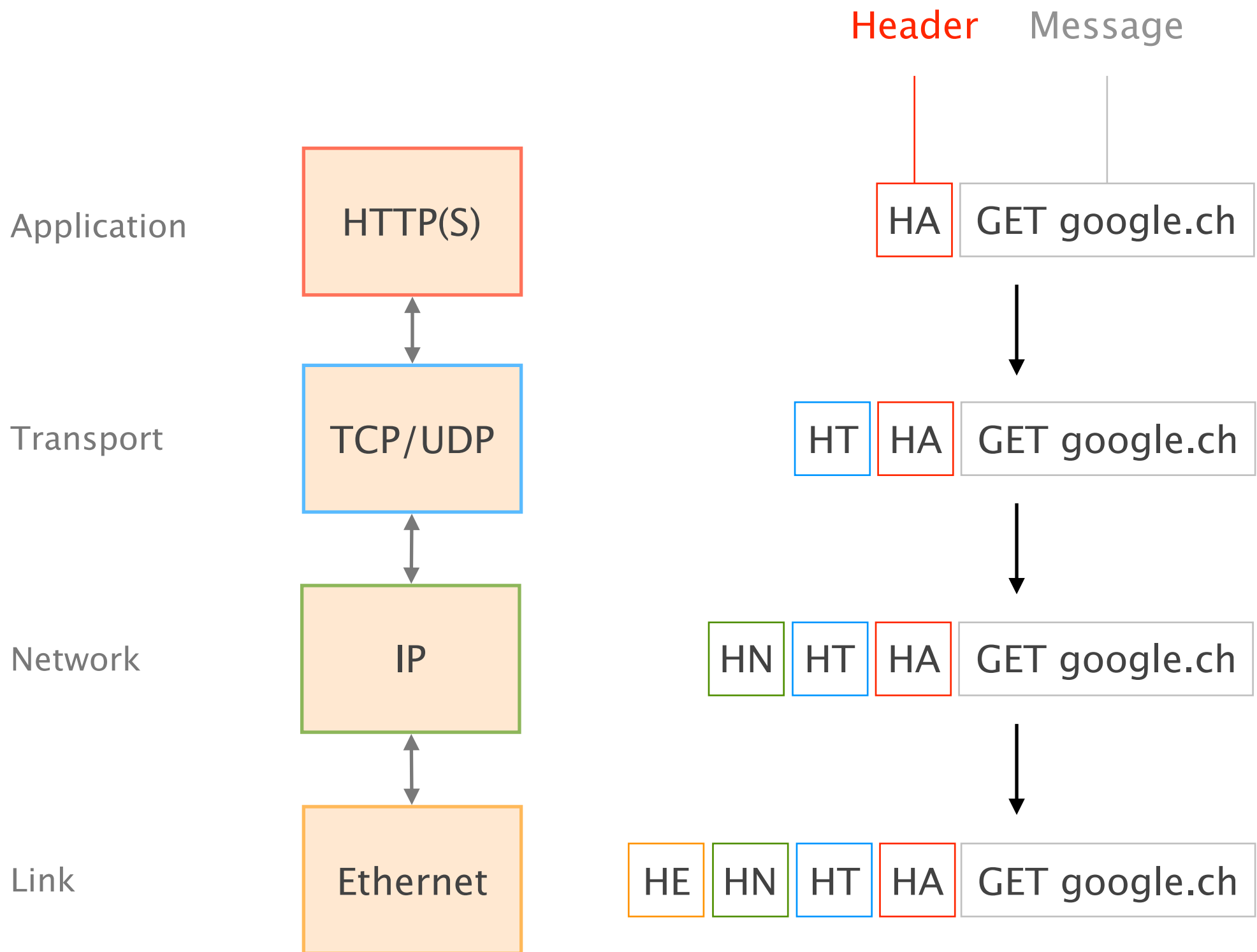
HA

GET google.ch

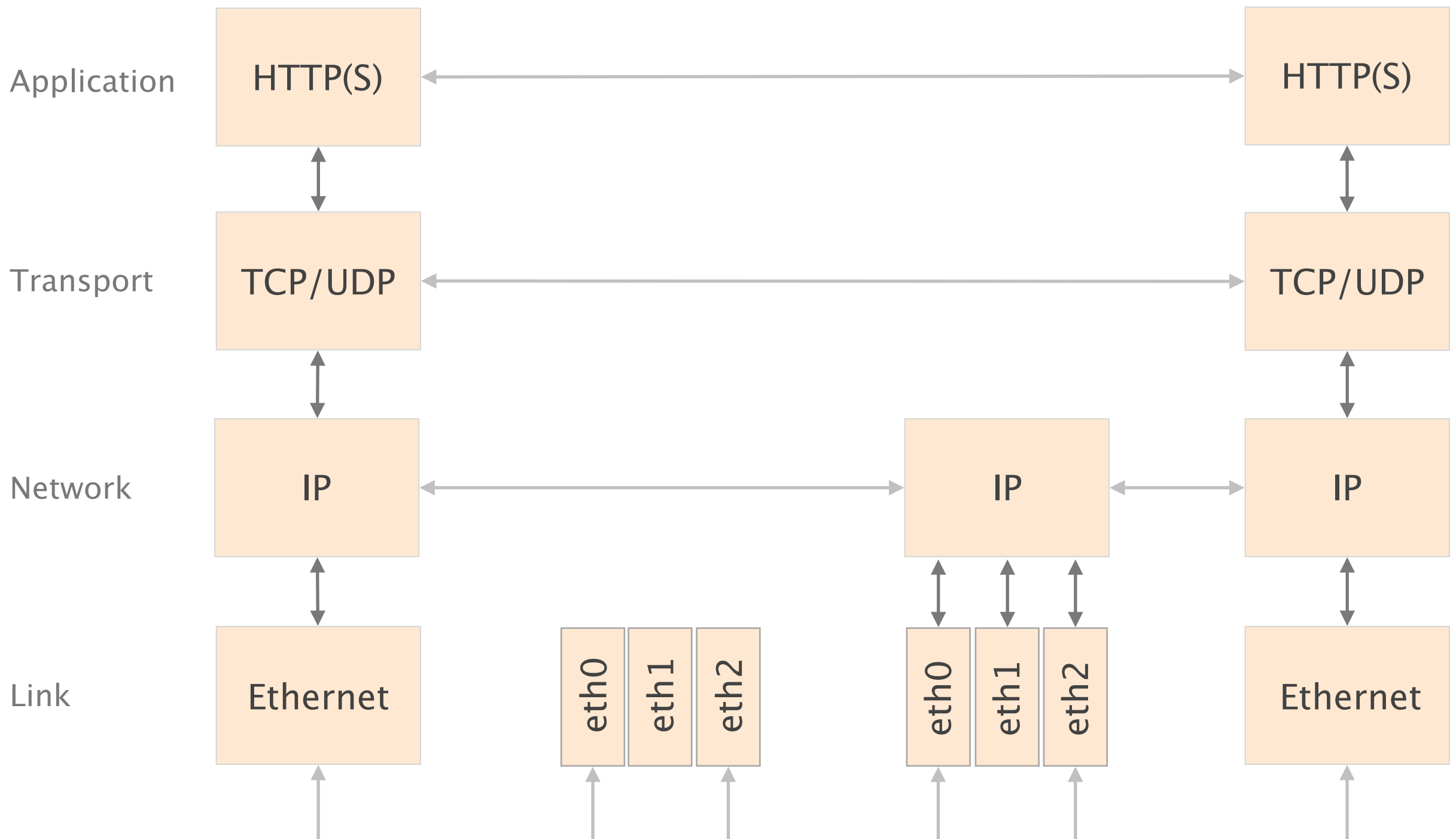




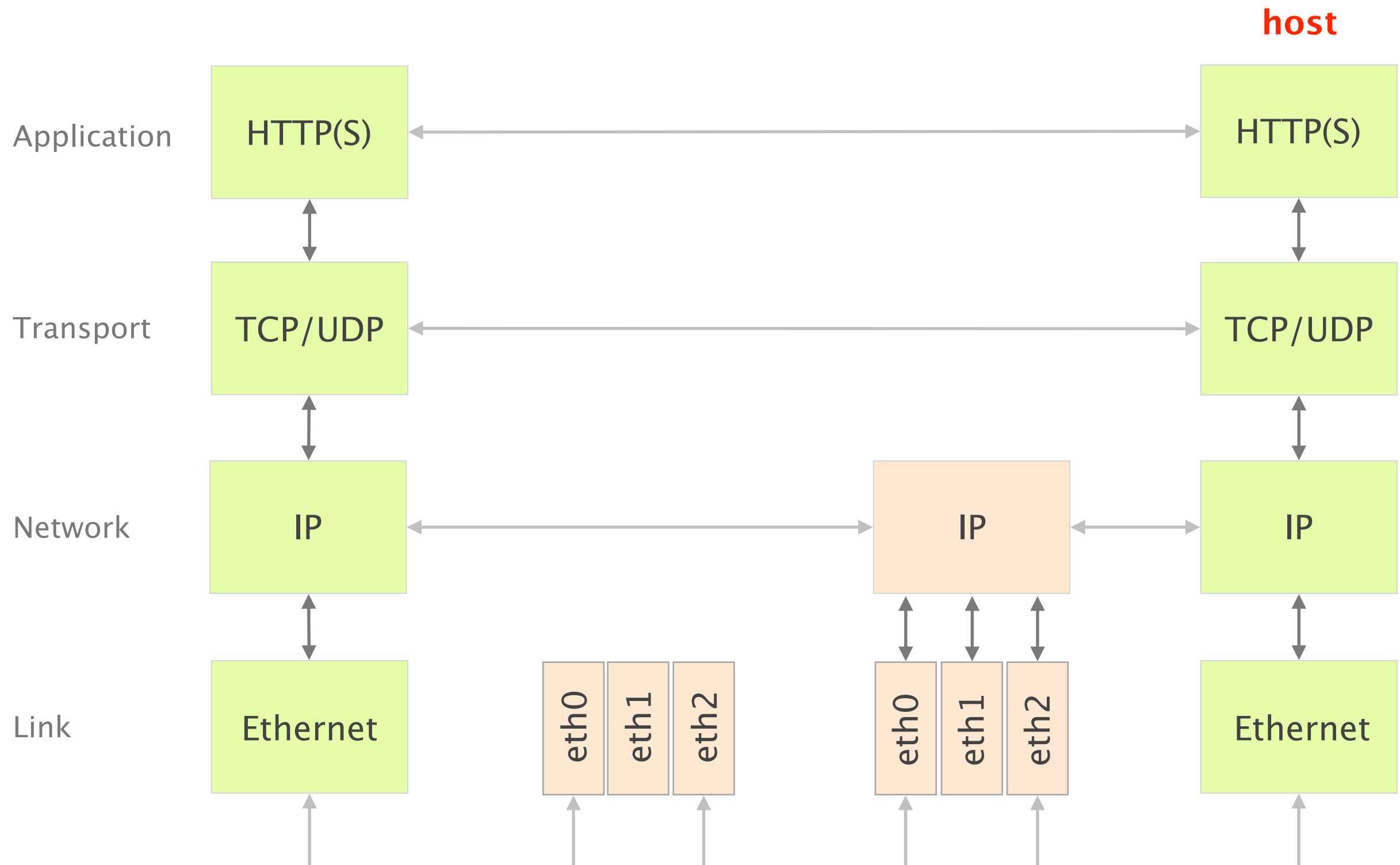




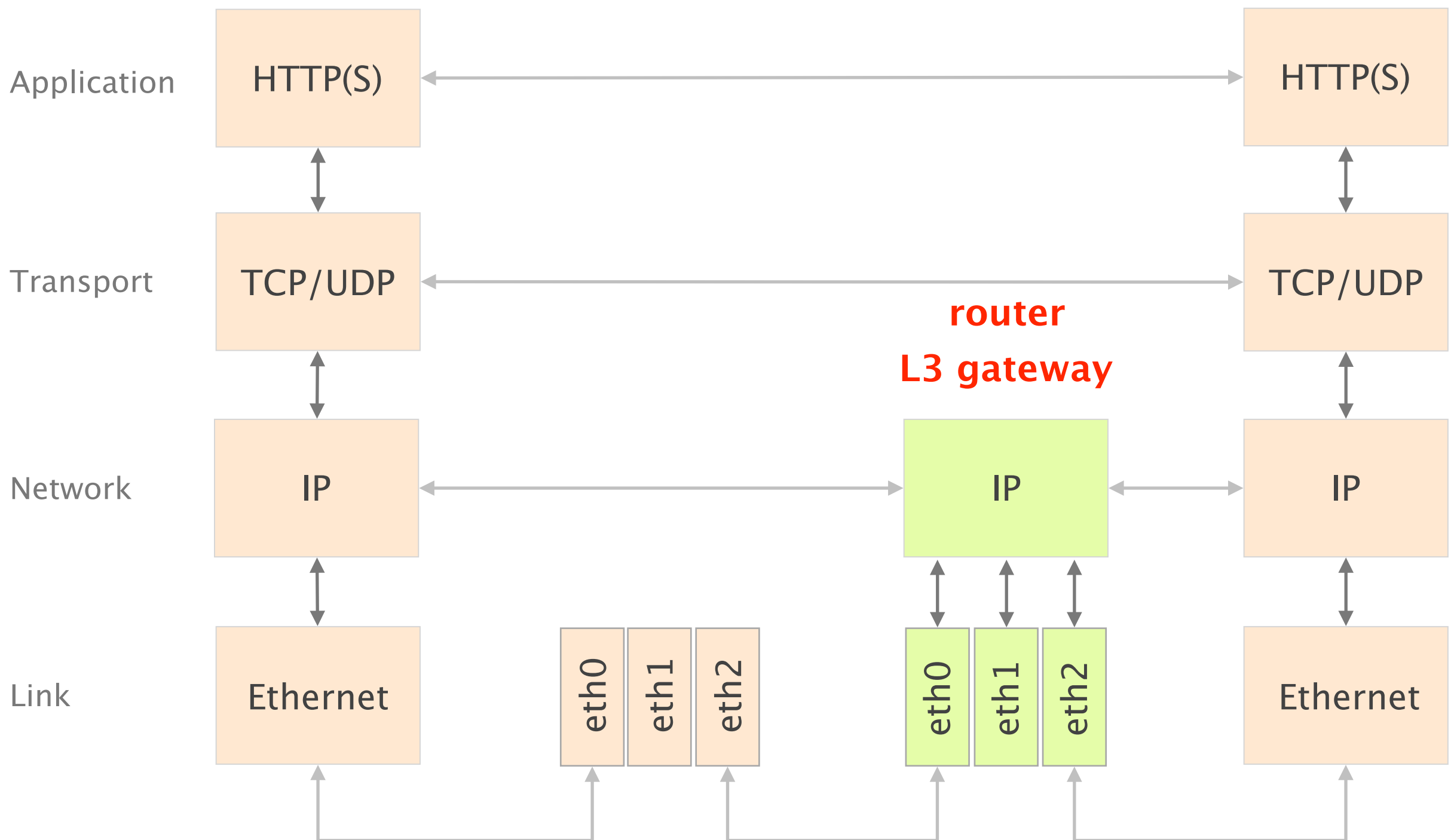
In practice, layers are distributed on every network device



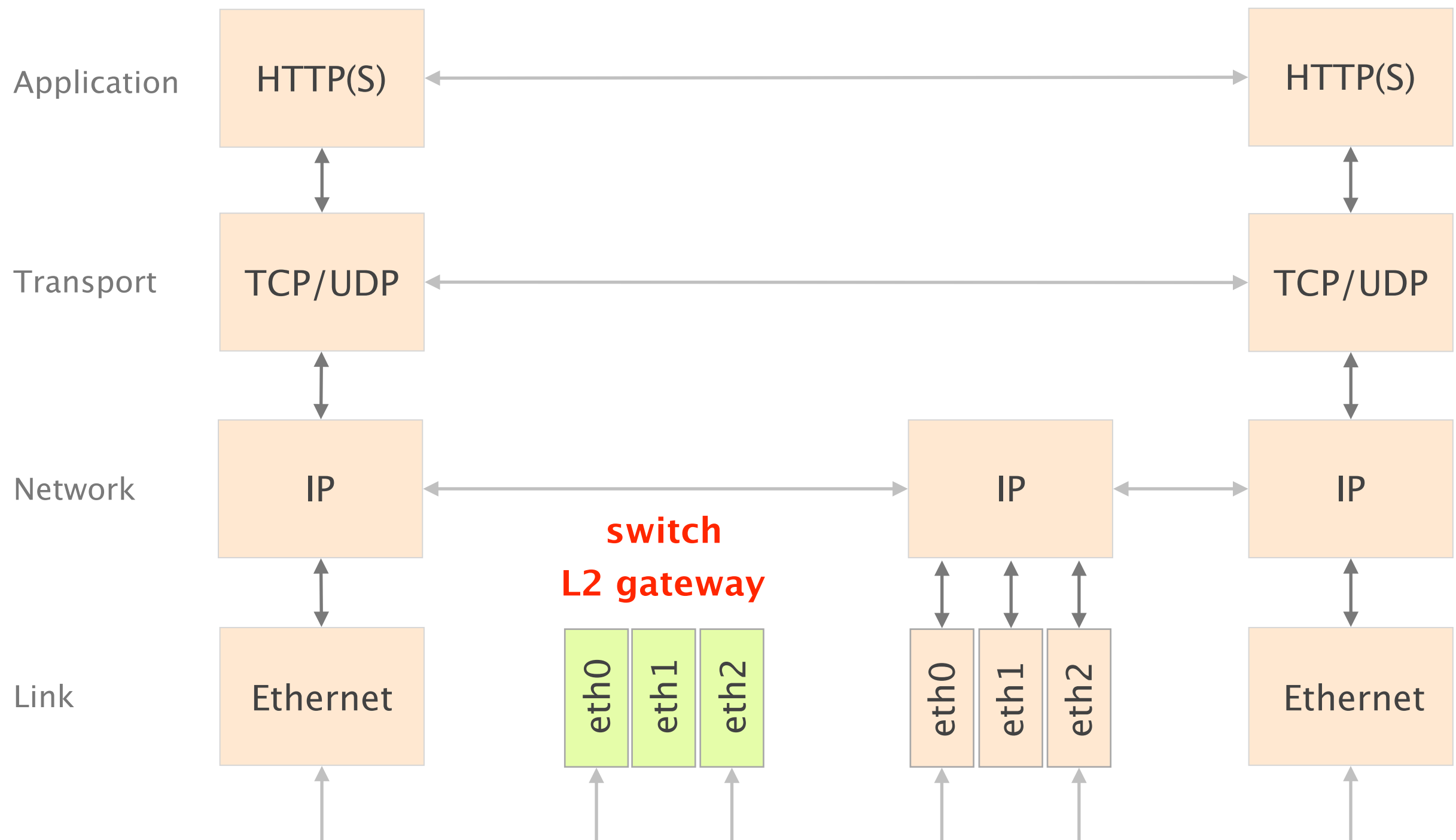
Since when bits arrive they must make it to the application, all the layers exist on a host



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

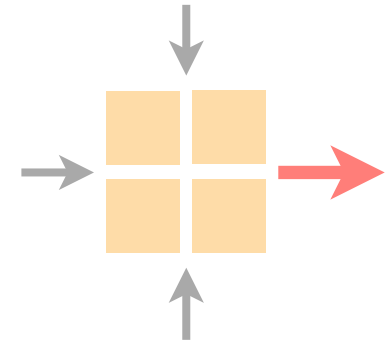
on a host, using Wireshark

<https://www.wireshark.org>



# Communication Networks

## Part 1: General overview



What is a network made of?

How is it shared?

How is it organized?

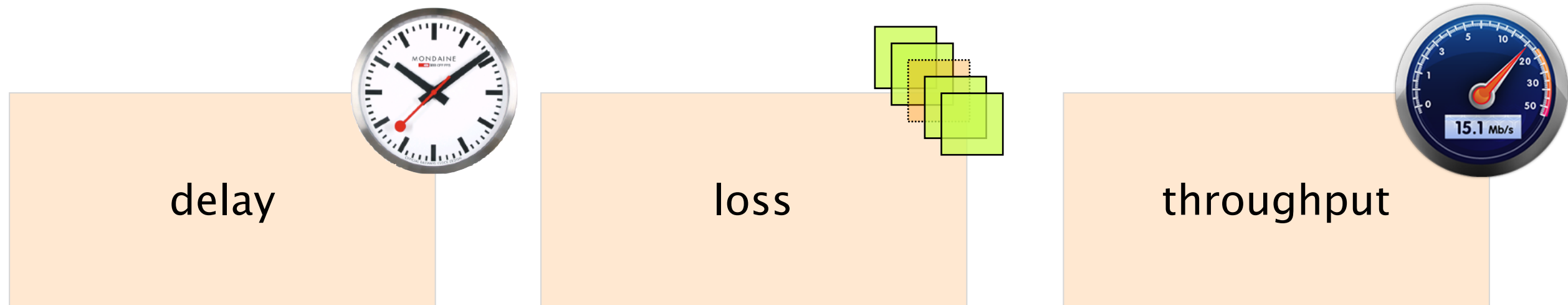
How does communication happen?

#5

How do we characterize it?



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

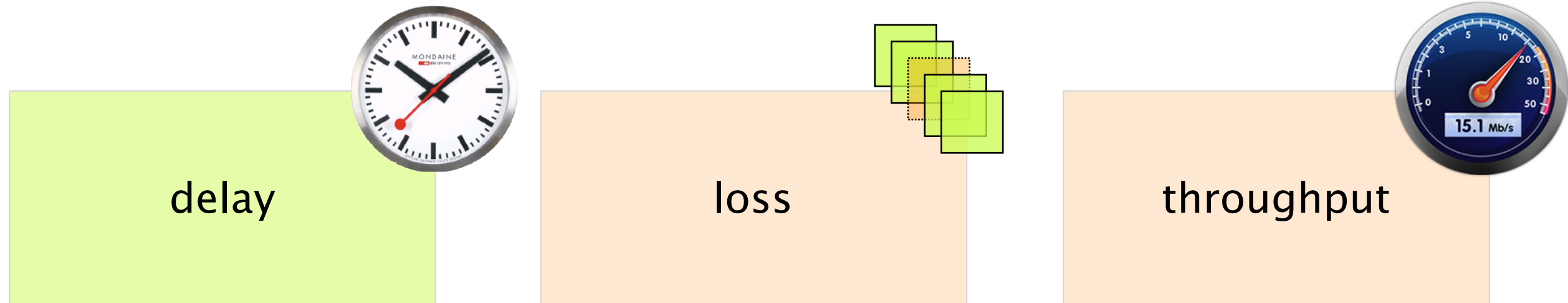


How long does it take for a packet to reach the destination

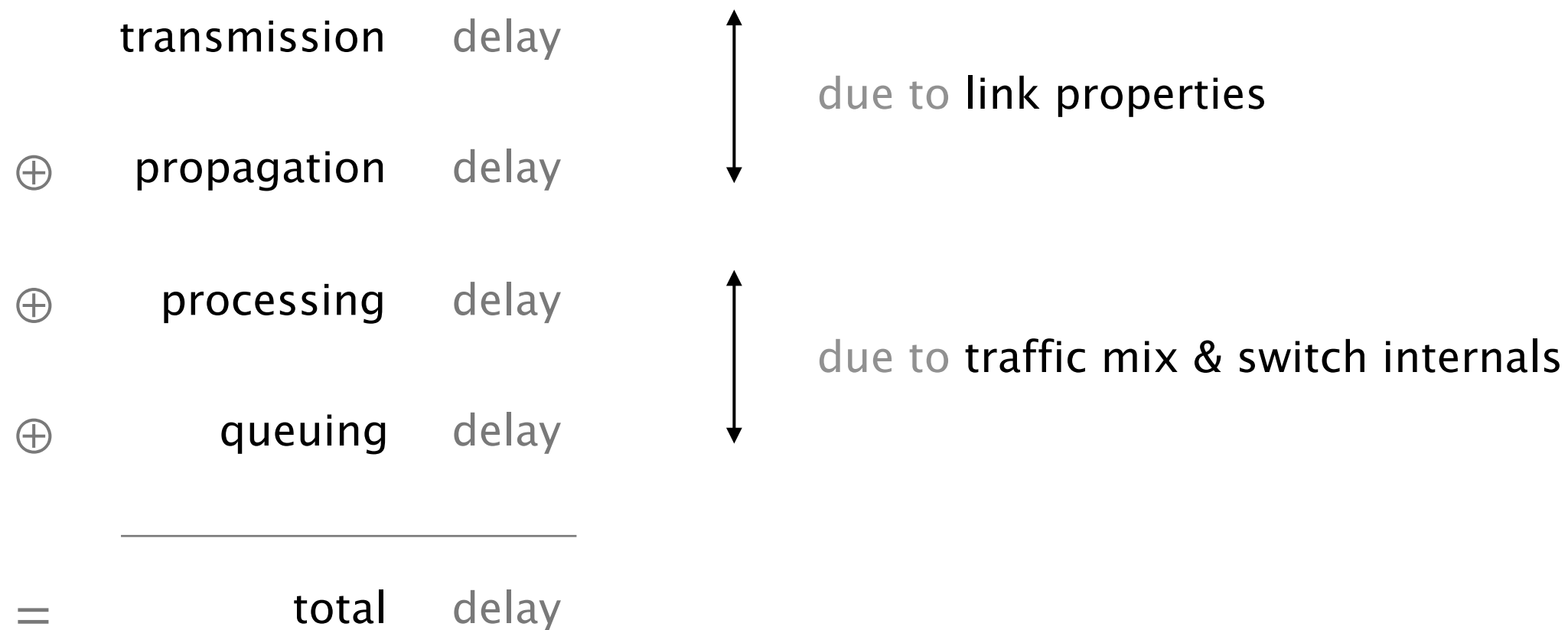
What fraction of packets sent to a destination are dropped?

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

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

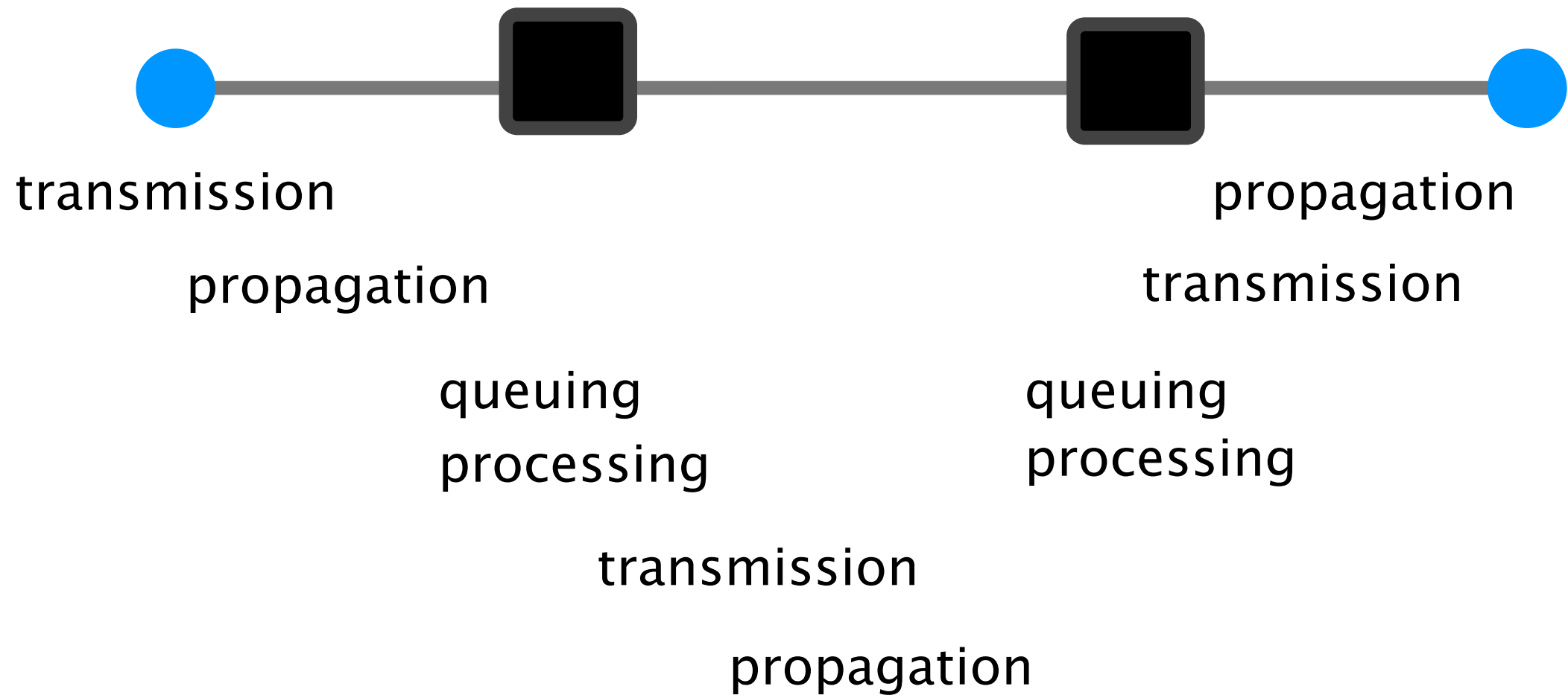


Each packet suffers from several types of delays  
at *each node* along the path



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

	transmission	delay	
⊕	propagation	delay	
⊕	processing	delay	<i>tend to be tiny</i>
⊕	queuing	delay	
<hr/>			
=	total	delay	



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

$$\begin{array}{lcl} \text{Transmission delay} & = & \frac{\text{packet size}}{\text{link bandwidth}} \\ [\text{sec}] & & \begin{array}{l} \text{[#bits]} \\ \text{[#bits/sec]} \end{array} \end{array}$$

$$\begin{array}{lcl} \text{Example} & & \frac{1000 \text{ bits}}{100 \text{ Mbps}} \\ & & 10 \mu\text{sec} \end{array}$$

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

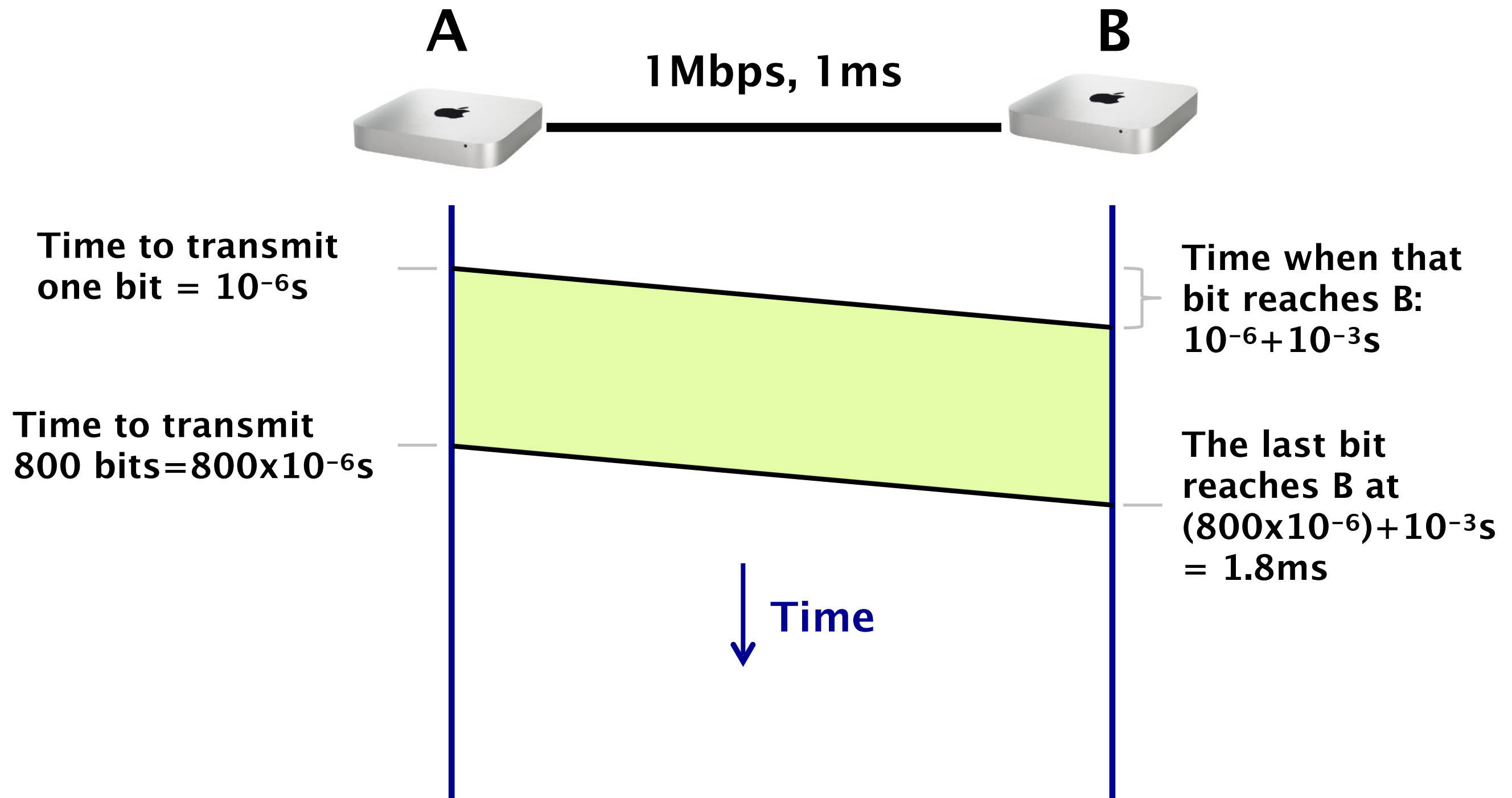
$$\begin{array}{lcl} \text{Propagation delay} & = & \frac{\text{link length}}{\text{propagation speed}} \\ \text{[sec]} & & \begin{array}{l} \text{[m]} \\ \text{[m/sec]} \\ \text{(fraction of speed of light)} \end{array} \end{array}$$

$$\begin{array}{lcl} \text{Example} & & \frac{30\,000\text{ m}}{2 \times 10^8\text{ m/sec}} \\ & & \begin{array}{l} \text{(speed of light in fiber)} \\ 150\text{ }\mu\text{sec} \end{array} \end{array}$$

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**



Time to transmit  
one bit =  **$10^{-9}s$**

Time to transmit  
800 bits =  $800 \times 10^{-9}s$

Time when that  
bit reaches B:  
 **$10^{-9} + 10^{-3}s$**

The last bit  
reaches B at  
 $(800 \times 10^{-9}) + 10^{-3}s$   
= **1.0008ms**

↓ Time

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



$10^7 \times 100\text{B}$  packets

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

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

$10^7 \times 100\text{B}$	pkt	1 Gbps link	transmission delay dominates
$1 \times 100\text{B}$	pkt	1 Gbps link	propagation delay dominates
$1 \times 100\text{B}$	pkt	1 Mbps link	both matter

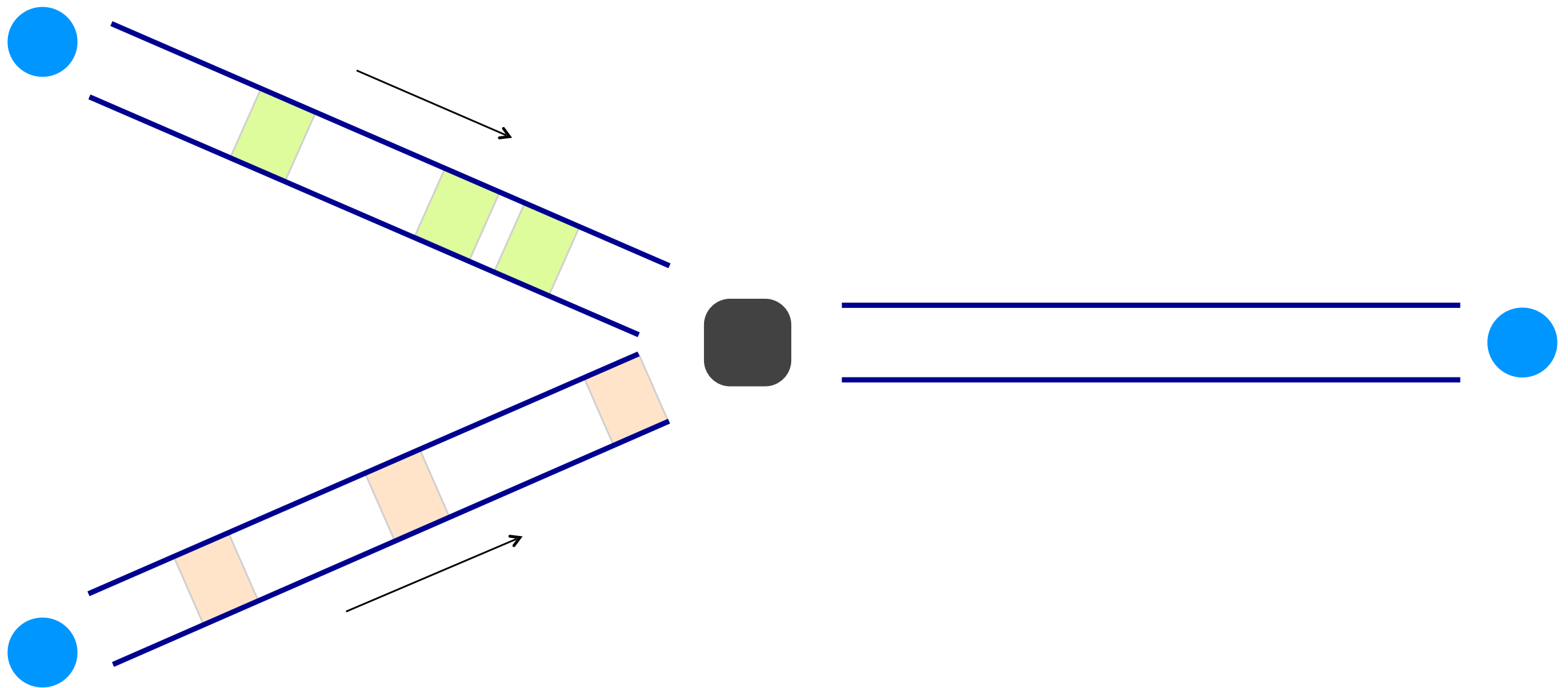
In the Internet, we **can't know** in advance which one matters!

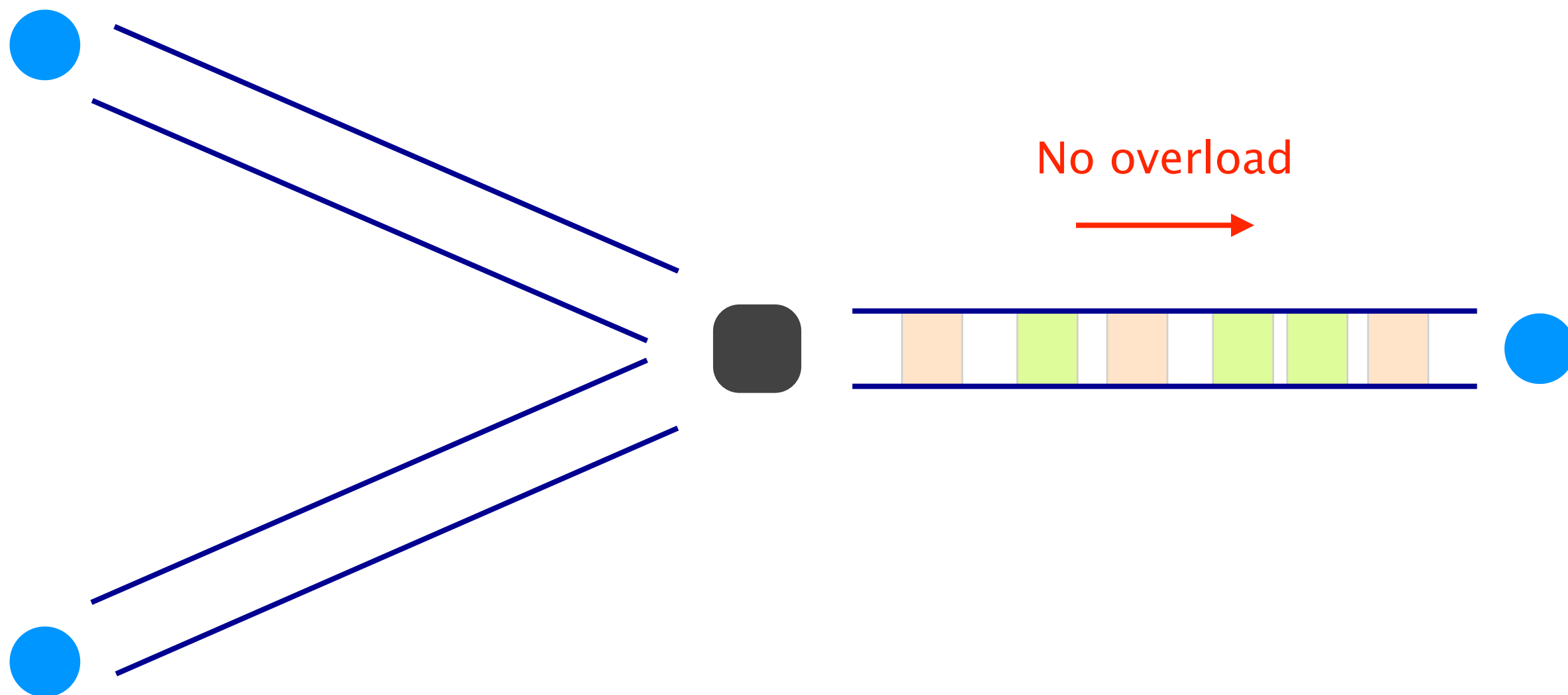
The queuing delay is the amount of time a packet waits (in a buffer) to be transmitted on a link

Queuing delay is the hardest to evaluate  
as it varies from packet to packet

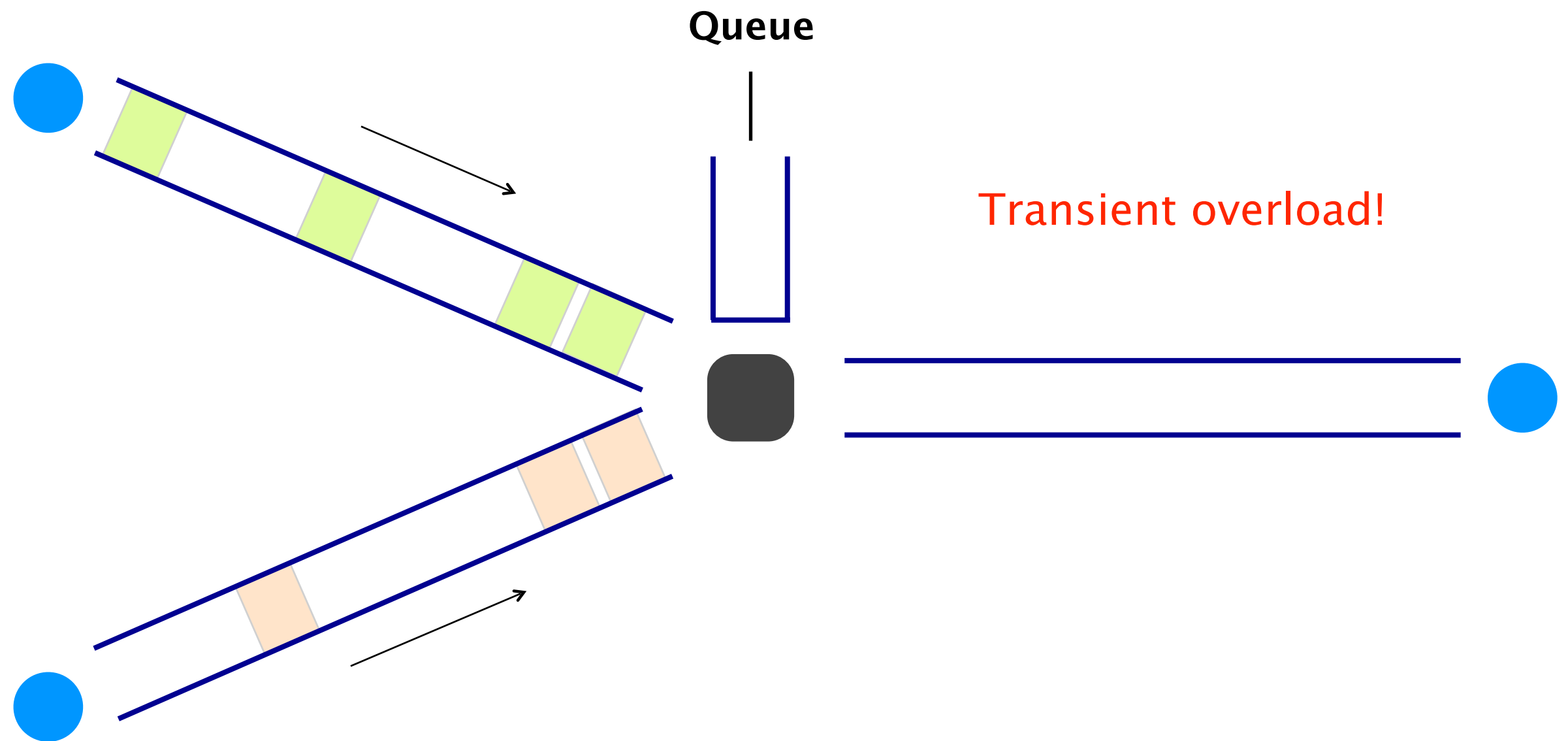
It is characterized with statistical measures  
*e.g.*, average delay & variance, probability of exceeding  $x$

Queuing delay depends on the traffic pattern

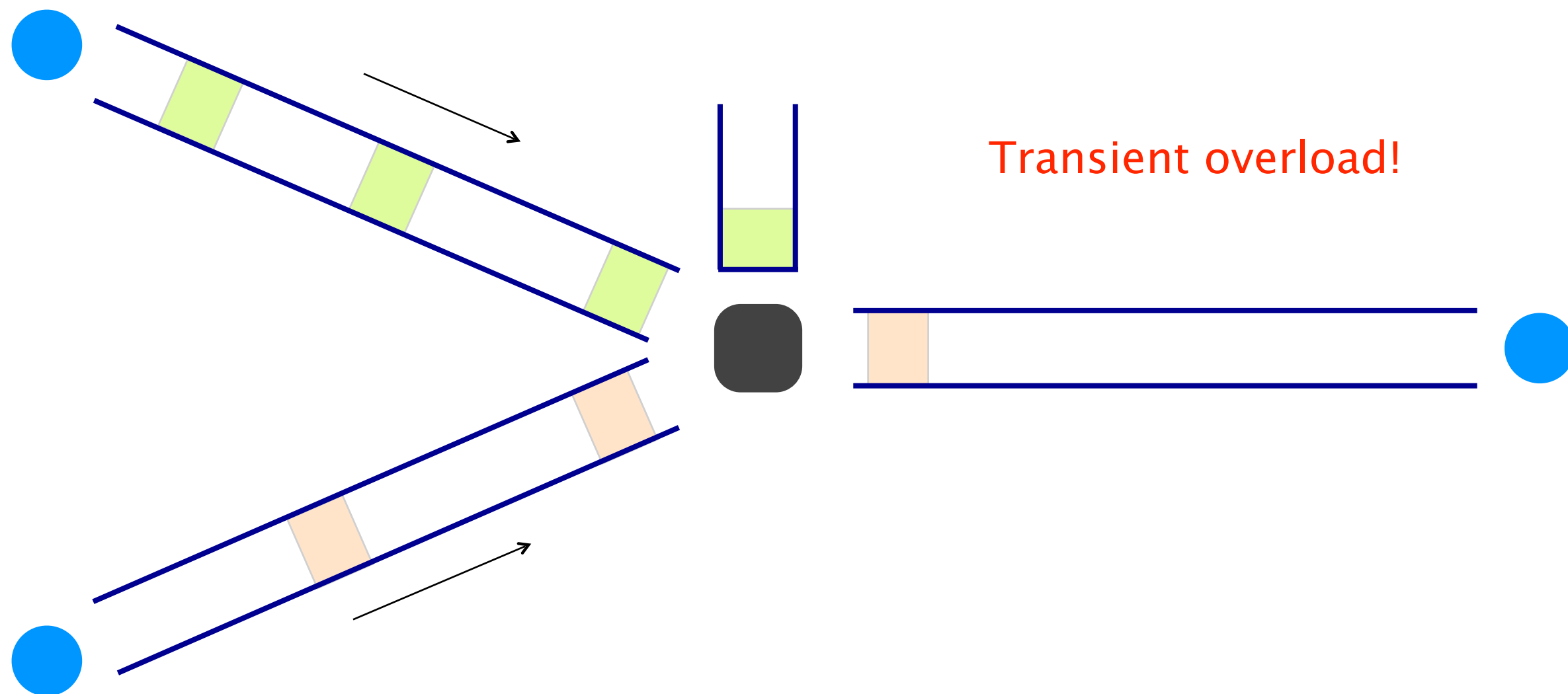


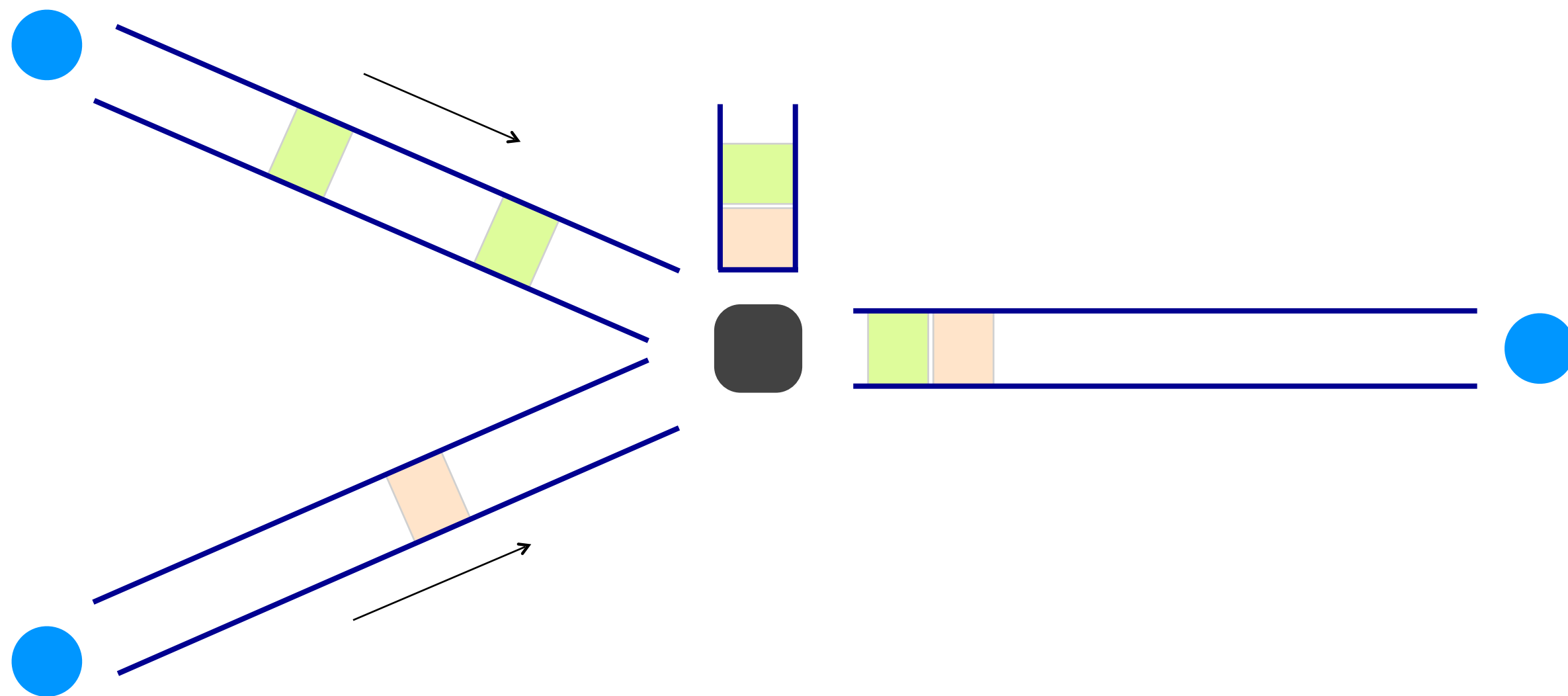


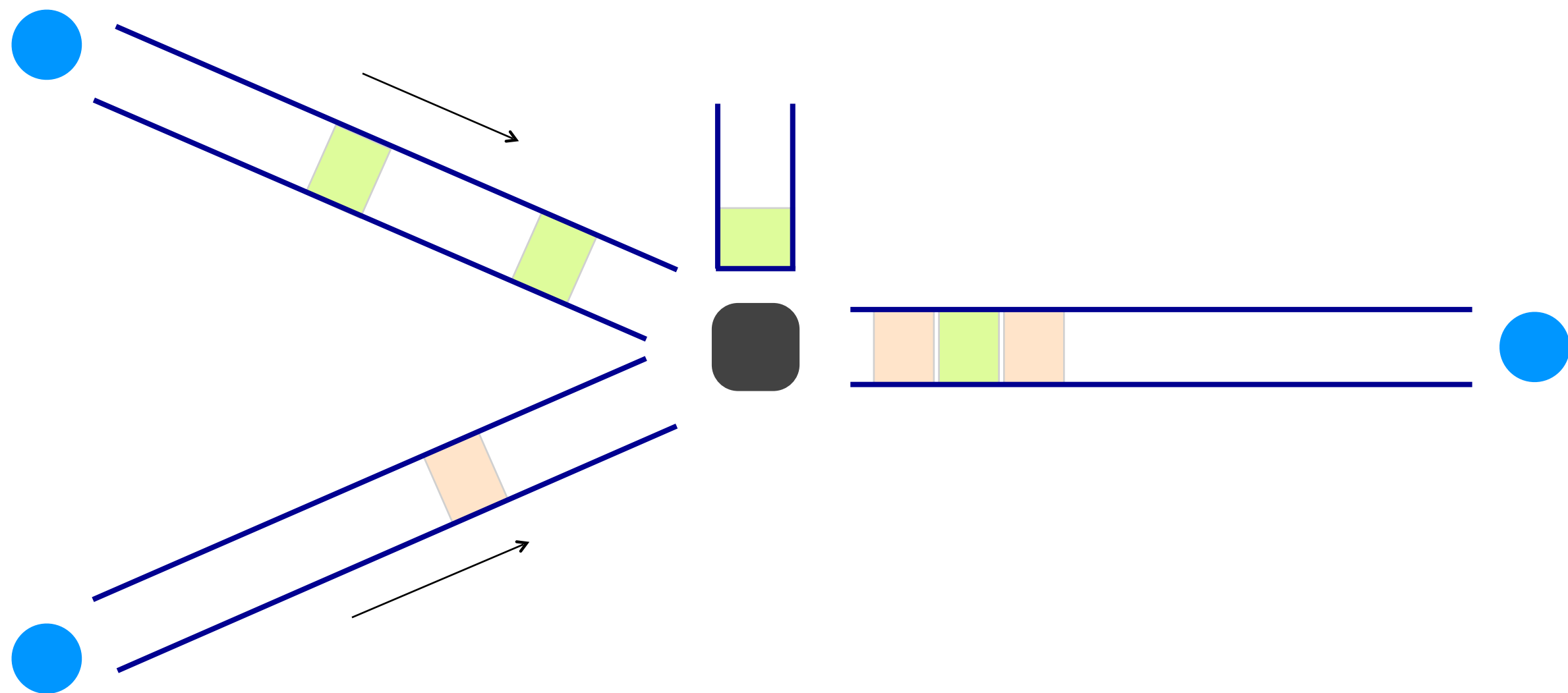
Queuing delay depends on the traffic pattern

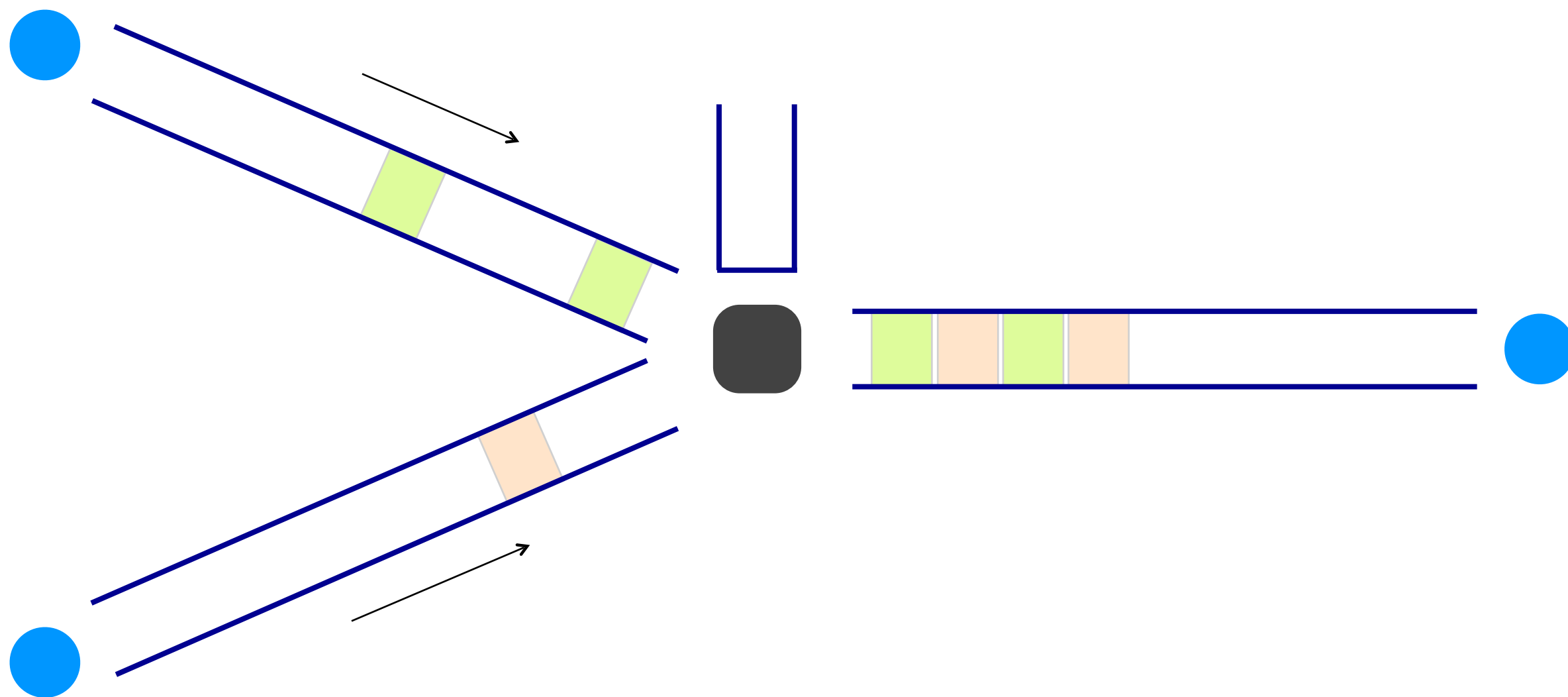




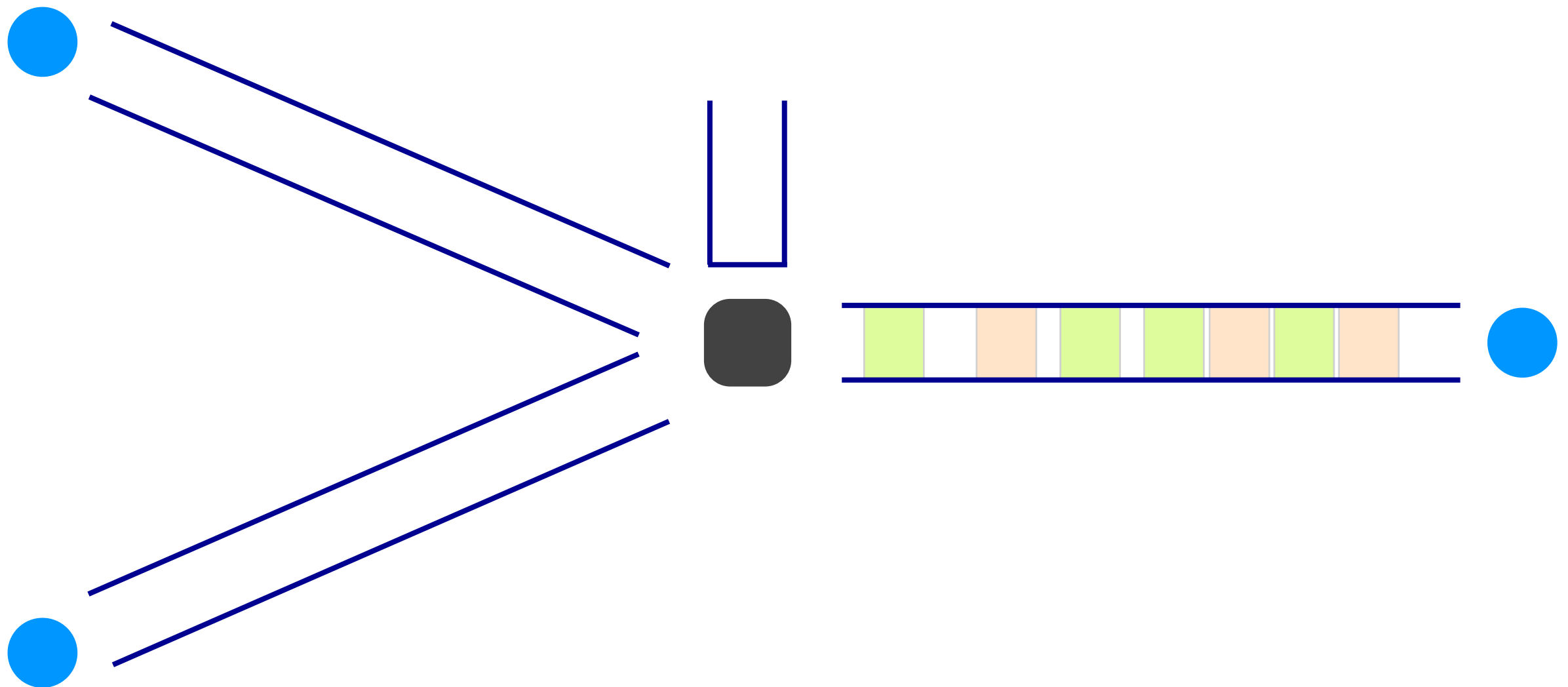








Queues absorb transient bursts,  
but introduce queueing delays



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	$La$	[bit/sec]
traffic intensity	$La/R$	

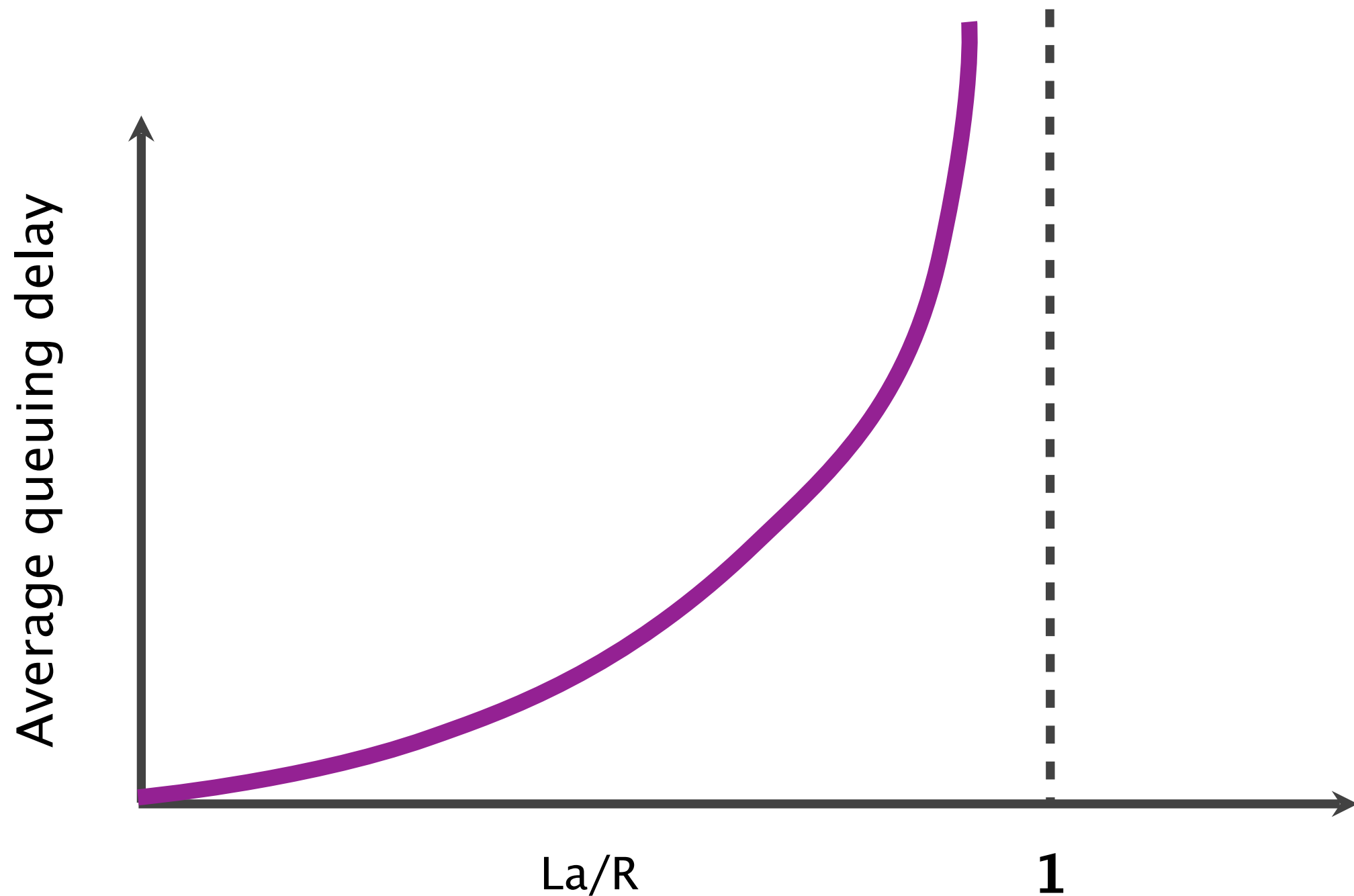
When the **traffic intensity is  $>1$** , the queue will increase without bound, and so does the queuing delay

Golden rule

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



When the **traffic intensity** is  $\leq 1$ ,  
queueing delay depends on the burst size

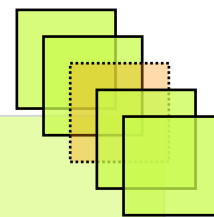


A network *connection* is characterized by its delay, loss rate and throughput

delay



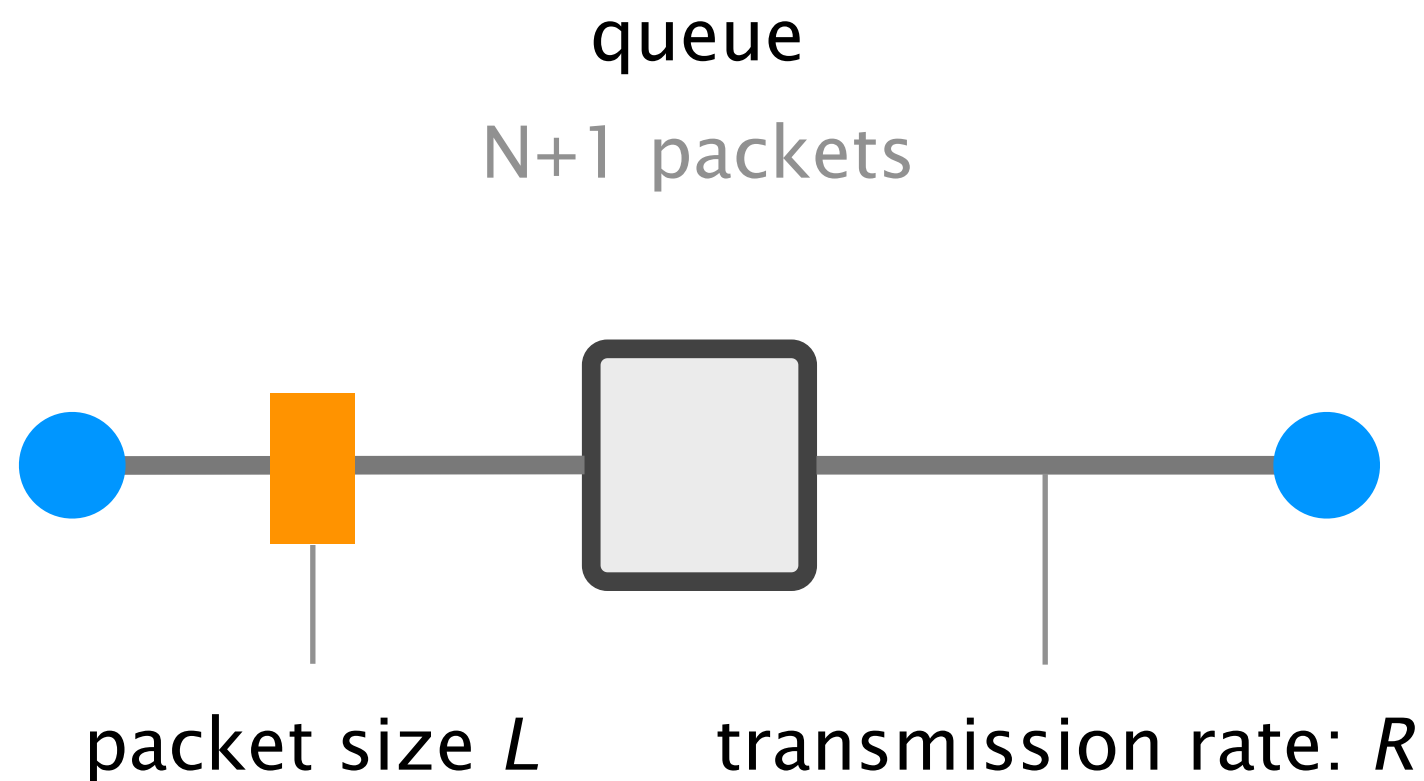
loss



throughput

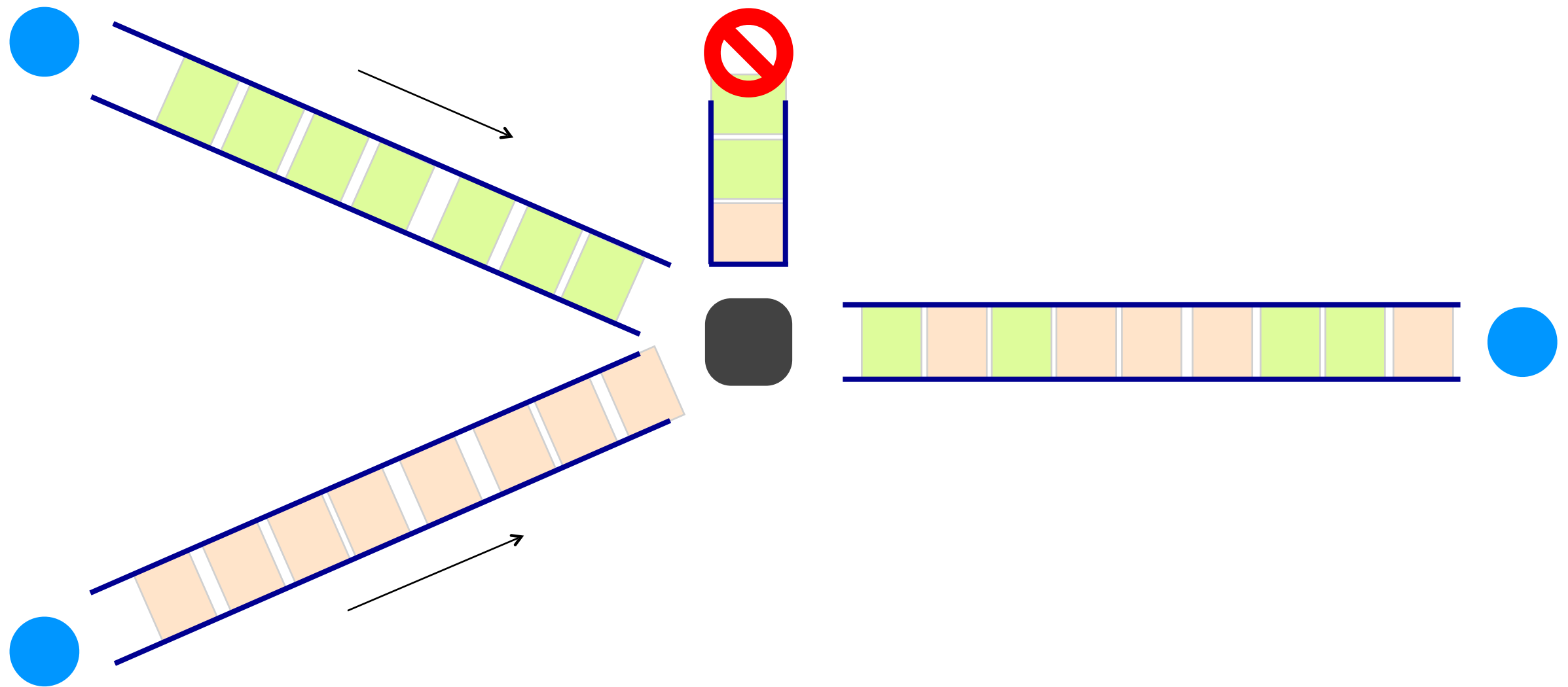


In practice, queues are not infinite.  
There is an upper bound on queuing delay.

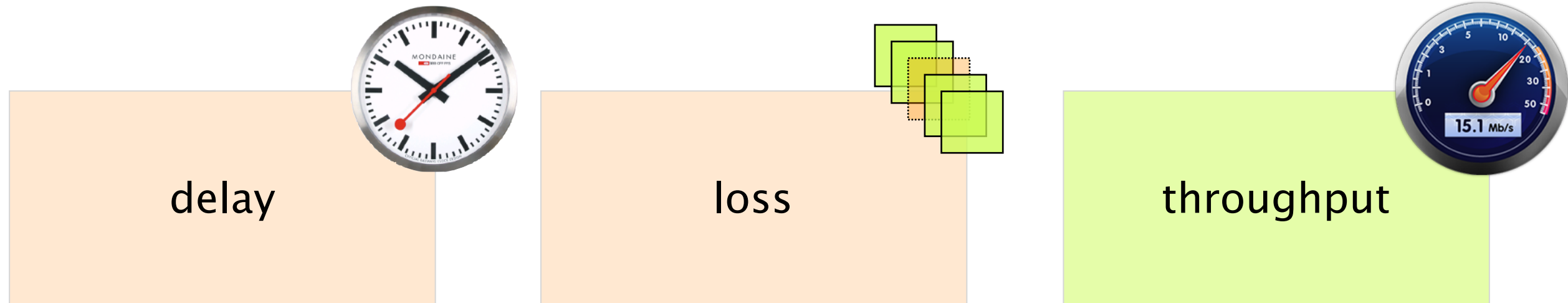


queuing delay upper bound:  $N \cdot L / R$

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



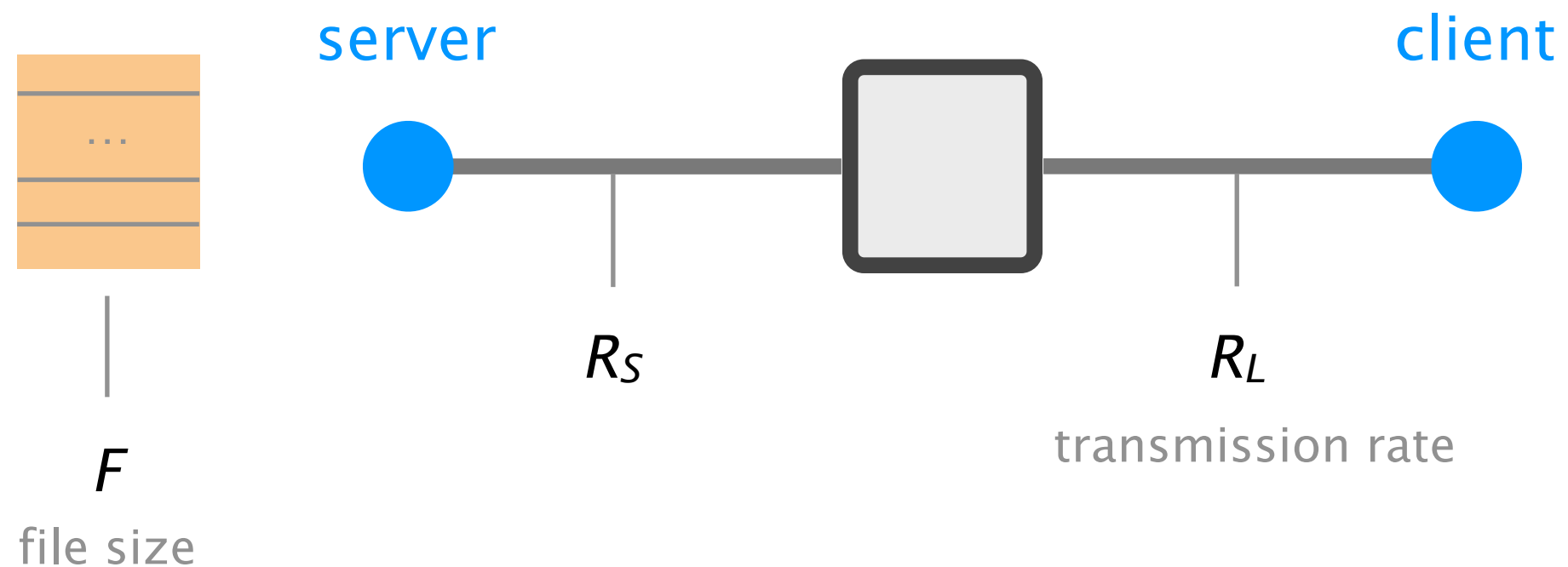
A network *connection* is characterized by its delay, loss rate and throughput



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

$$\begin{array}{lcl} \text{Average throughput} & = & \frac{\text{data size}}{\text{transfer time}} \\ \text{[#bits/sec]} & & \begin{array}{l} \text{[#bits]} \\ \text{[sec]} \end{array} \end{array}$$

To compute throughput, one has to consider the bottleneck link

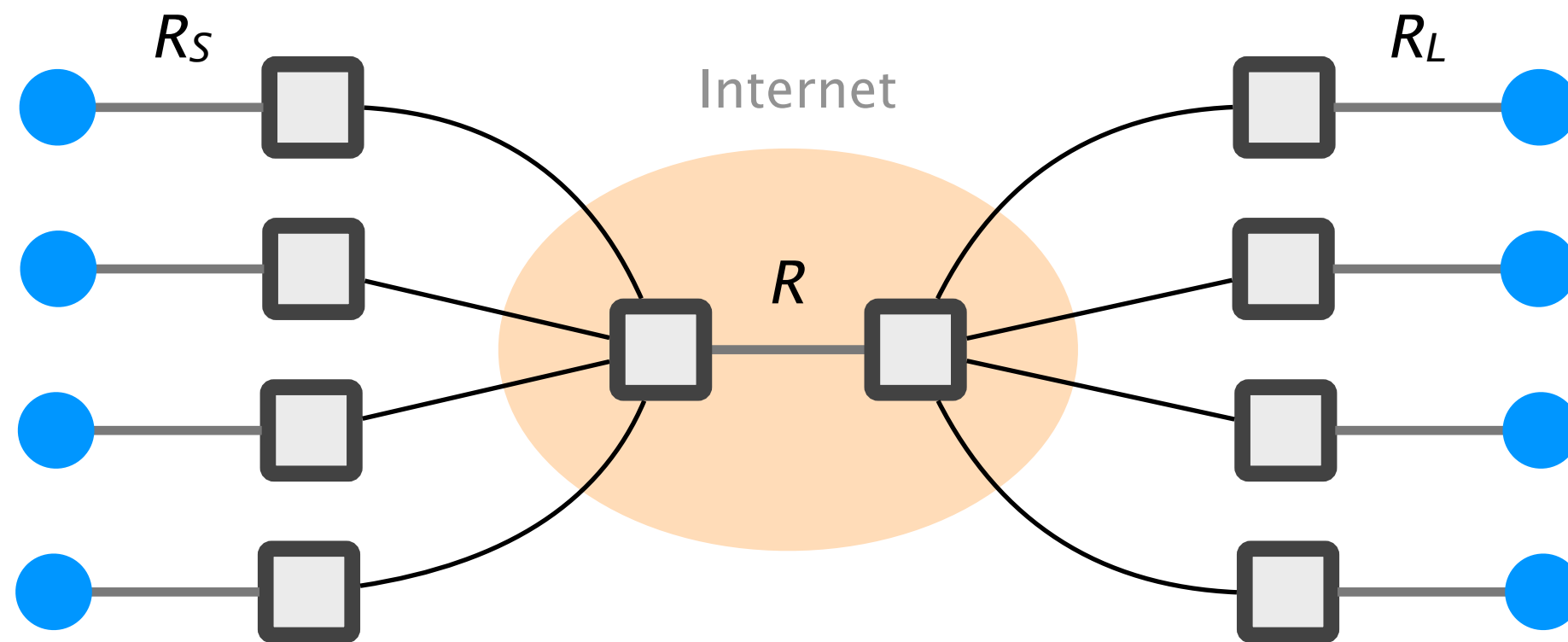


Average throughput

$$\min(R_S, R_L)$$

= transmission rate  
of the bottleneck link

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



if  $4 * \min(R_S, R_L) > R$

the bottleneck is now in the core,  
providing each download  $R/4$  of throughput

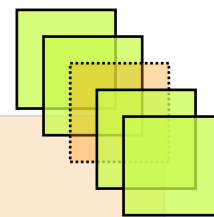


A network *connection* is characterized by its delay, loss rate and throughput

delay



loss

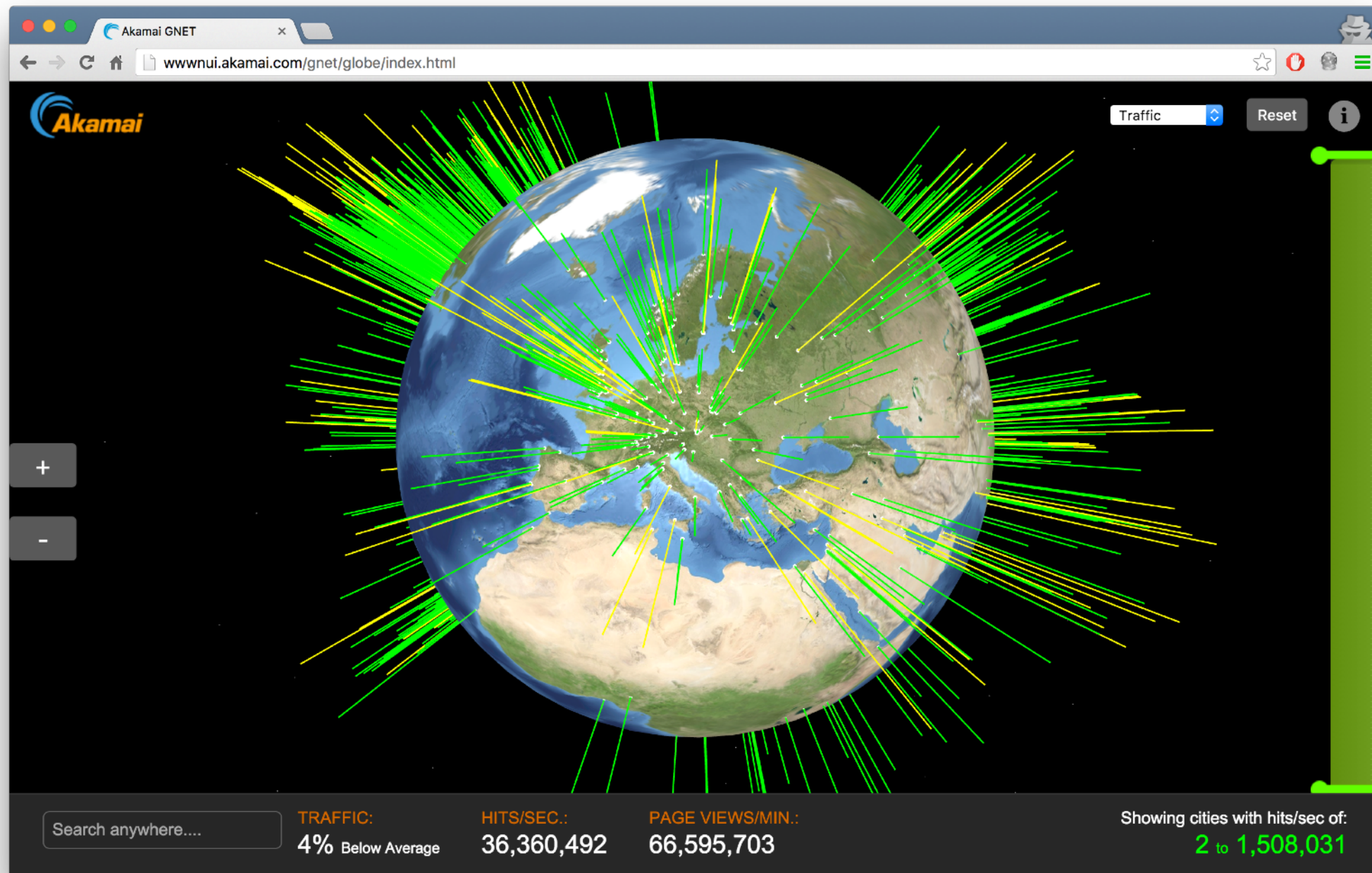


throughput



As technology improves, throughput increase &  
delays are getting lower except for propagation  
(speed of light)

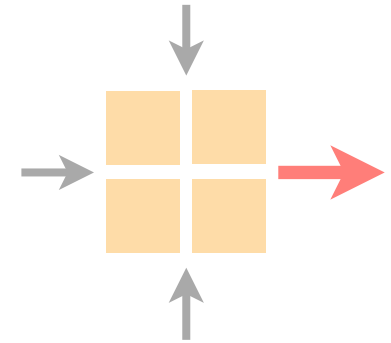
Because of propagation delays,  
Content Delivery Networks move content closer to you



<http://wwwnui.akamai.com/gnet/globe/index.html>

# Communication Networks

## Part 1: General overview



What is a network made of?

How is it shared?

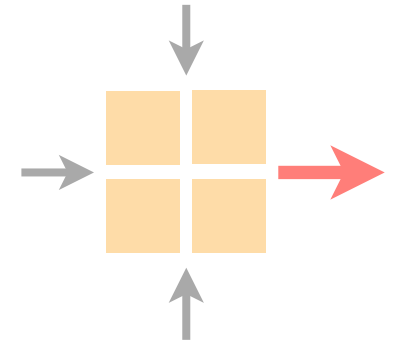
How is it organized?

How does communication happen?

How do we characterize it?

# Communication Networks

## Part 2: Concepts

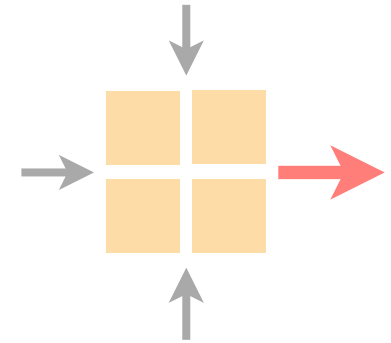


routing

reliable  
delivery

# Communication Networks

## Part 2: Concepts



routing

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

reliable  
delivery

How do you ensure reliable transport  
on top of best-effort delivery?

This week

routing

Next week

reliable  
delivery

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


Think of IP packets as envelopes



Packet



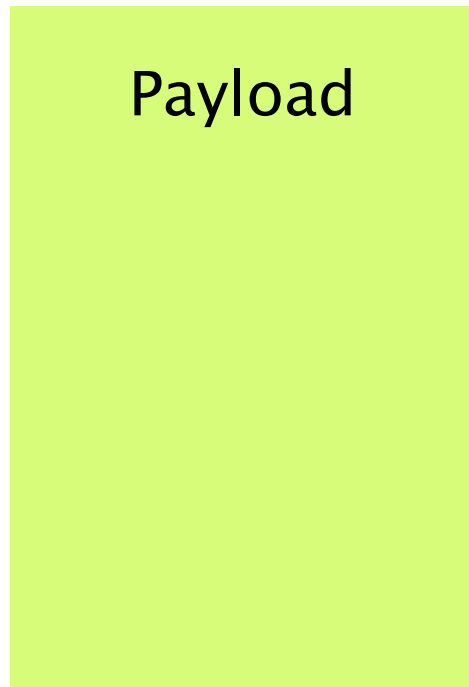
Like an envelope,  
packets have a header



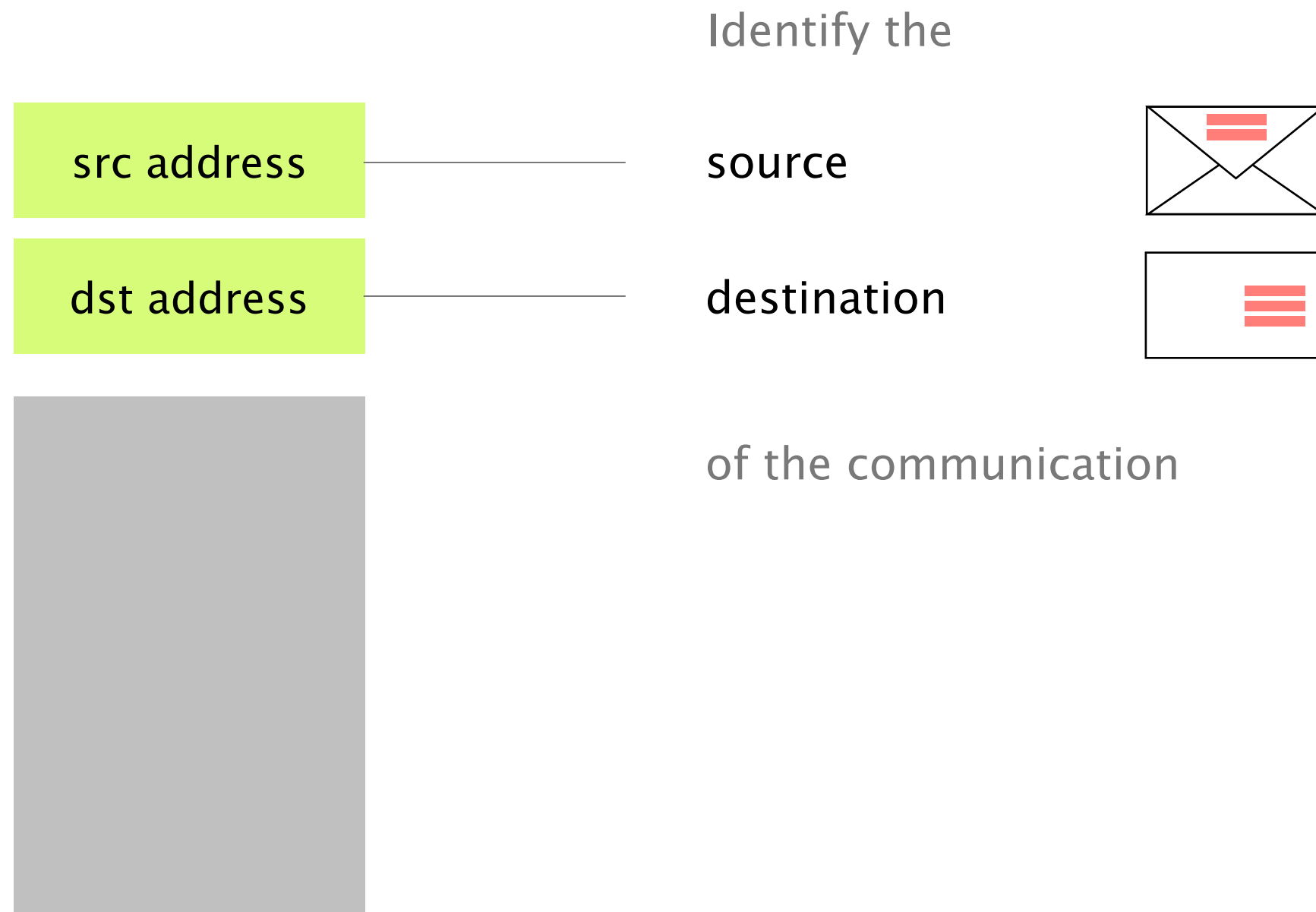
Header

The diagram illustrates a packet structure. It consists of two vertically stacked rectangular boxes. The top box is light green and contains the word 'Header'. The bottom box is gray and is empty, representing the data payload. The boxes are aligned to the left and have a consistent width.

Like an envelope,  
packets have a payload



# The header contains the metadata needed to forward the packet

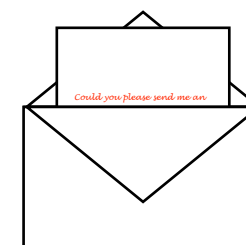


# The payload contains the data to be delivered

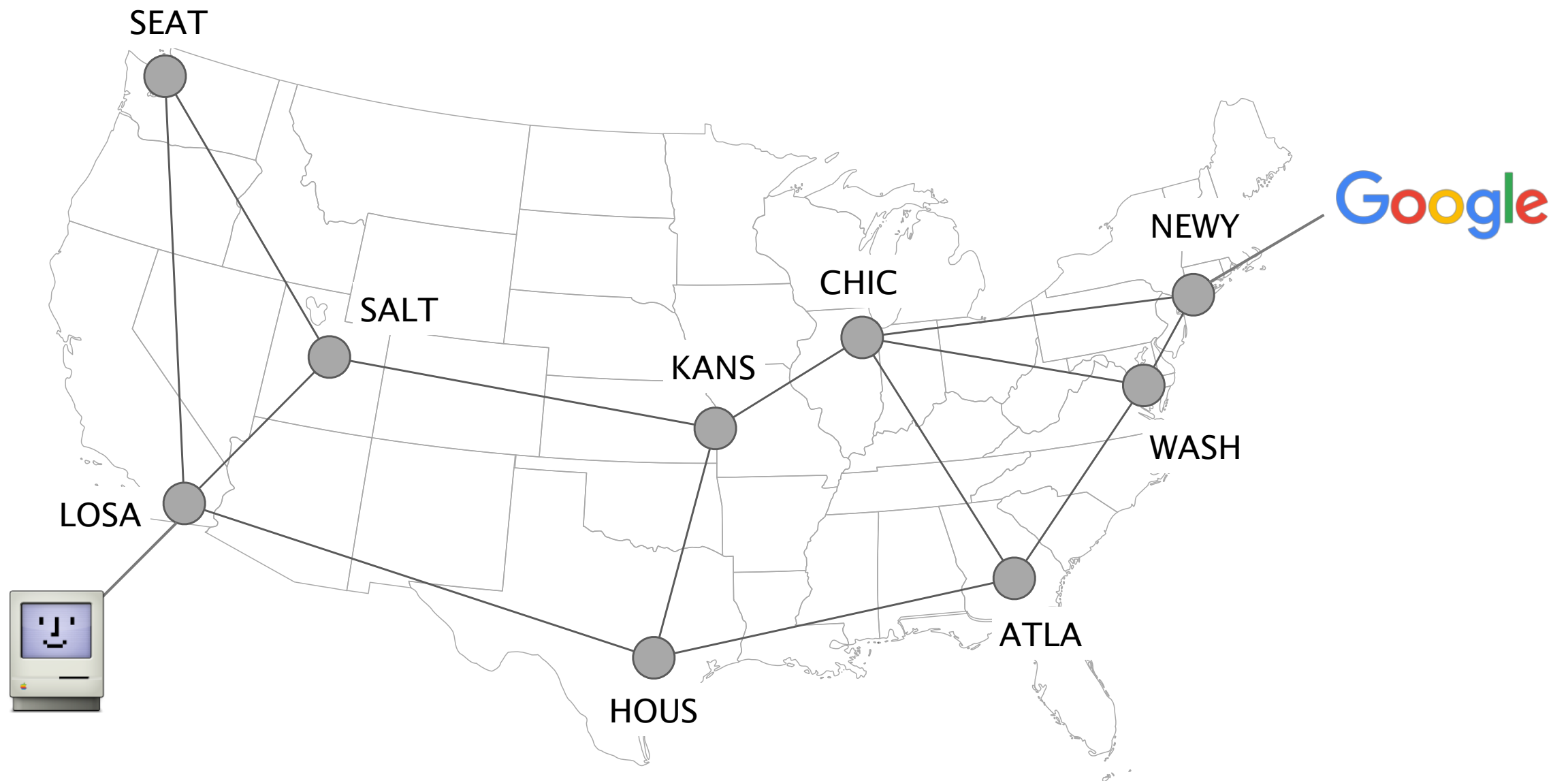


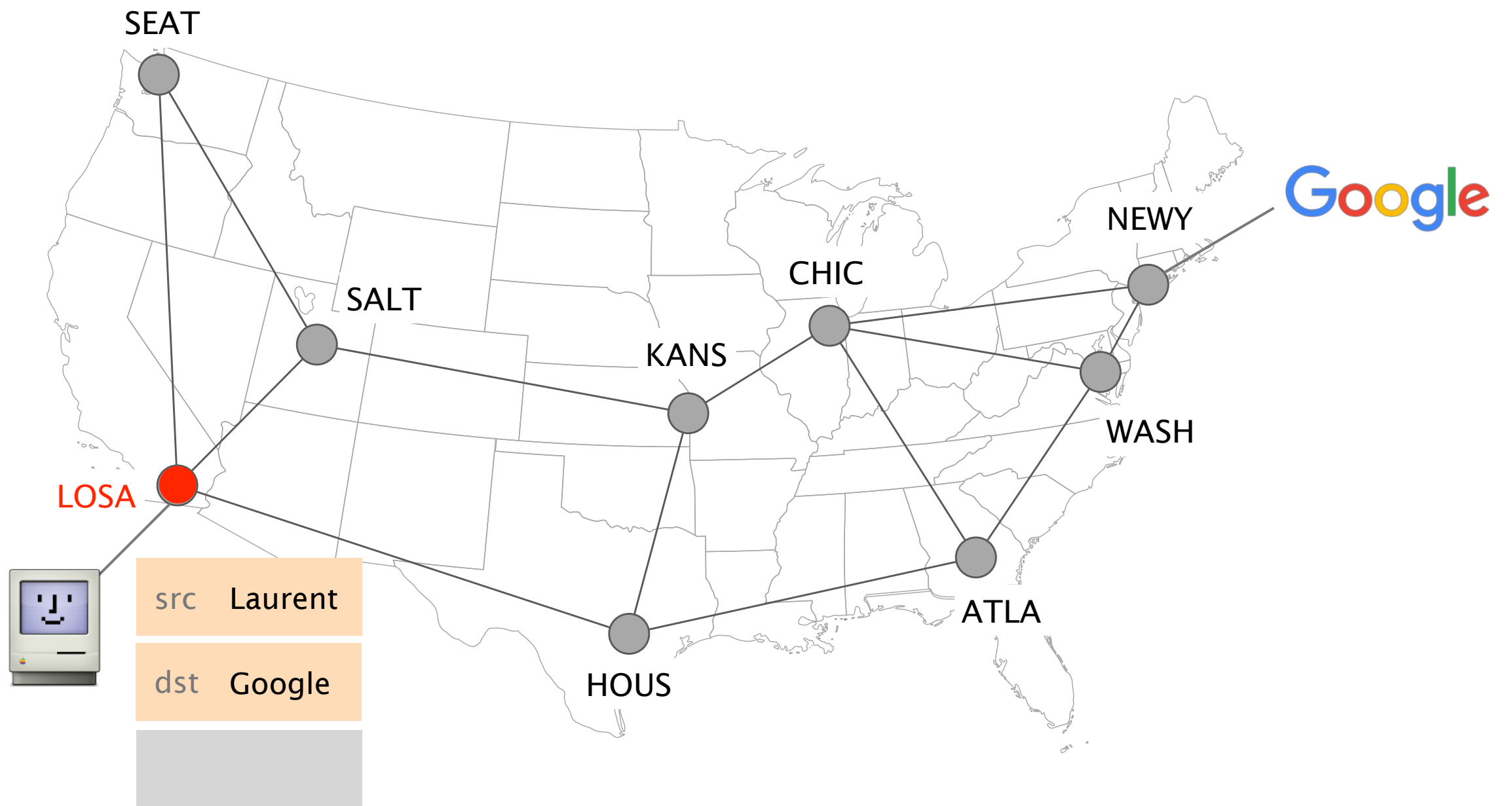
Payload

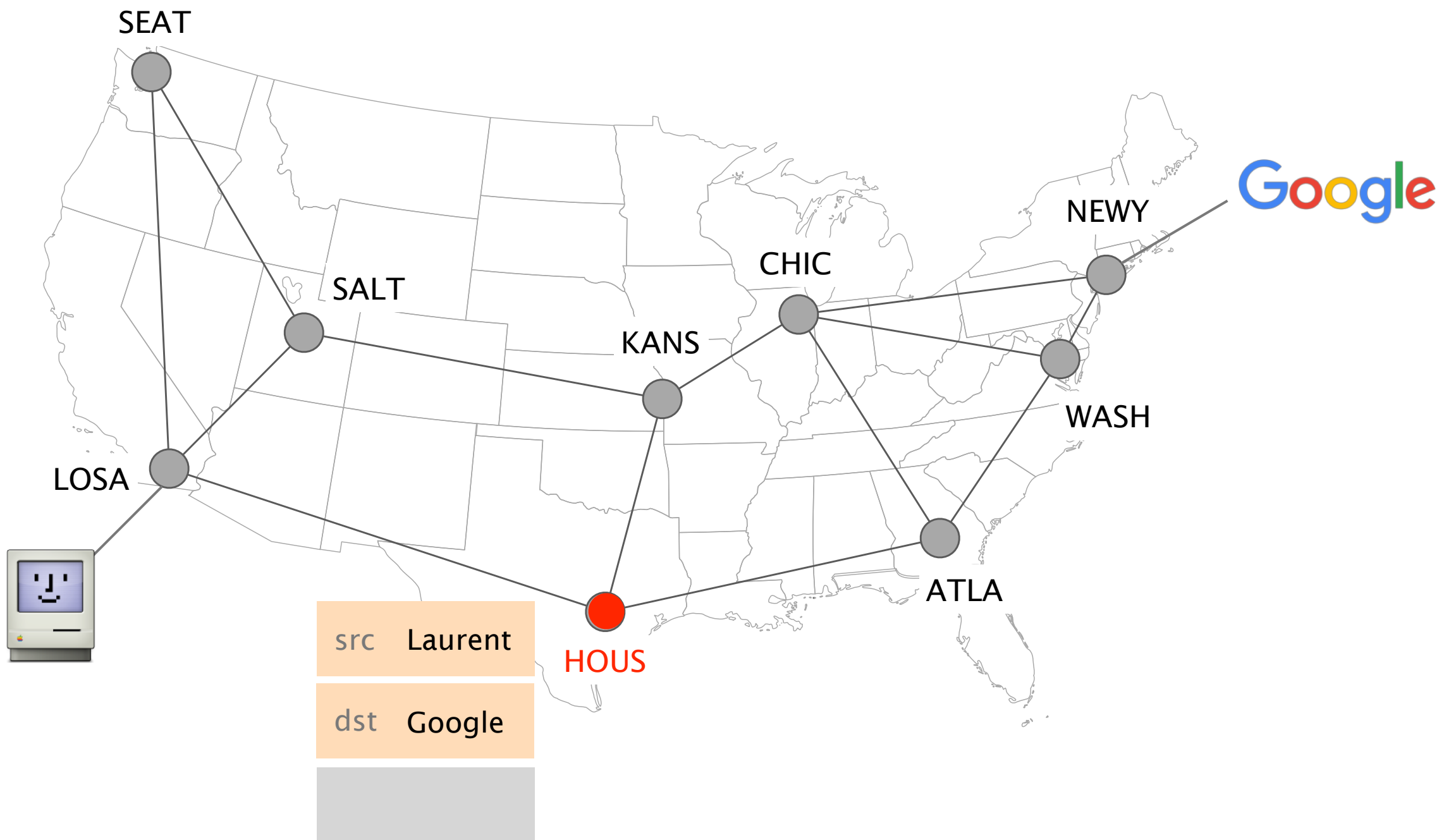
```
<html><head>
<meta http-equiv="content-type" content="text/html; charset=UTF-8">
<title>Google</title>
</head><body>
  
  <form action="/search" name=f>
    <input name=hl type=hidden value=en>
    <input name=q size=55 title="Google Search" value="">
    <input name=btnG type=submit value="Google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

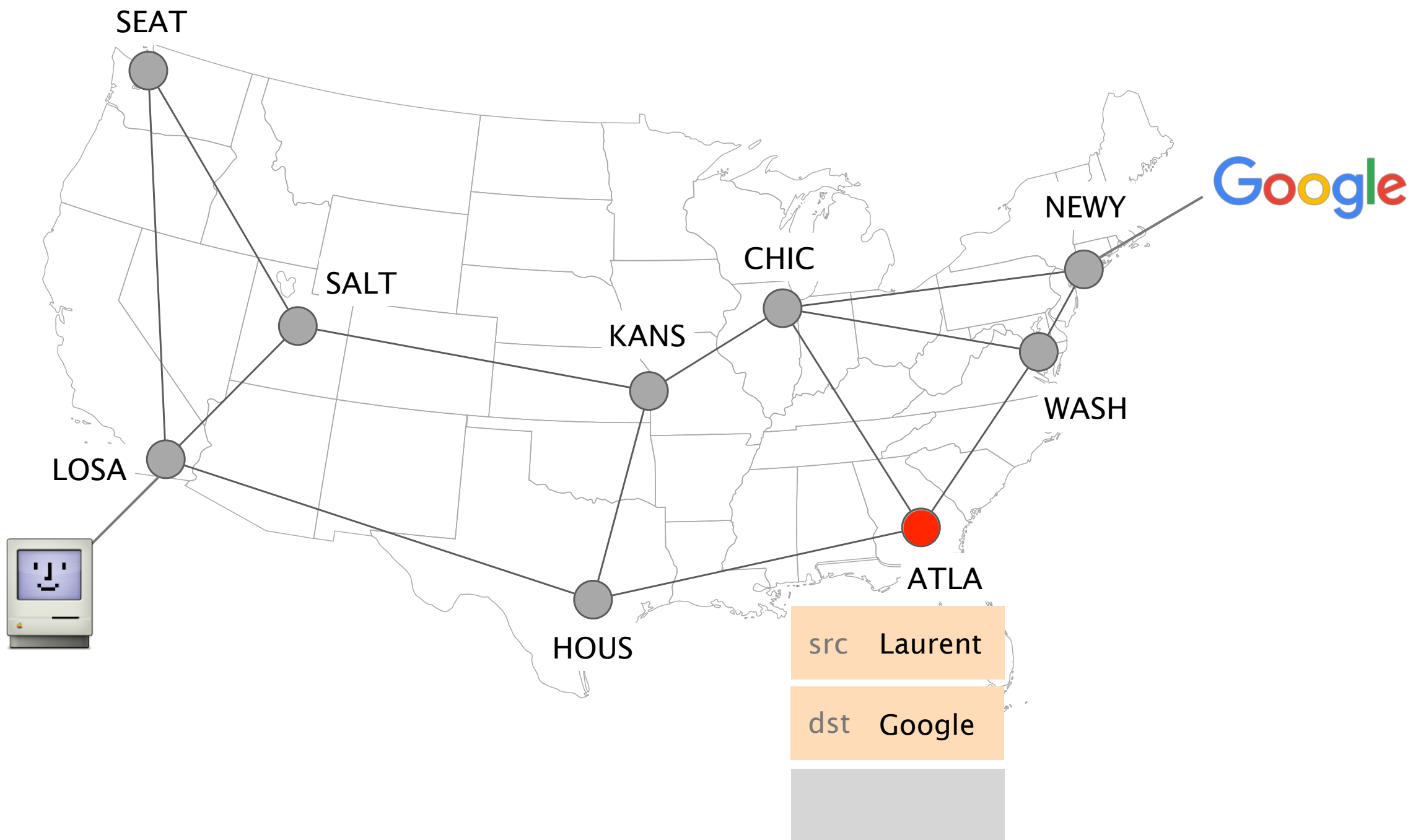


Routers forward IP packets hop-by-hop towards their destination

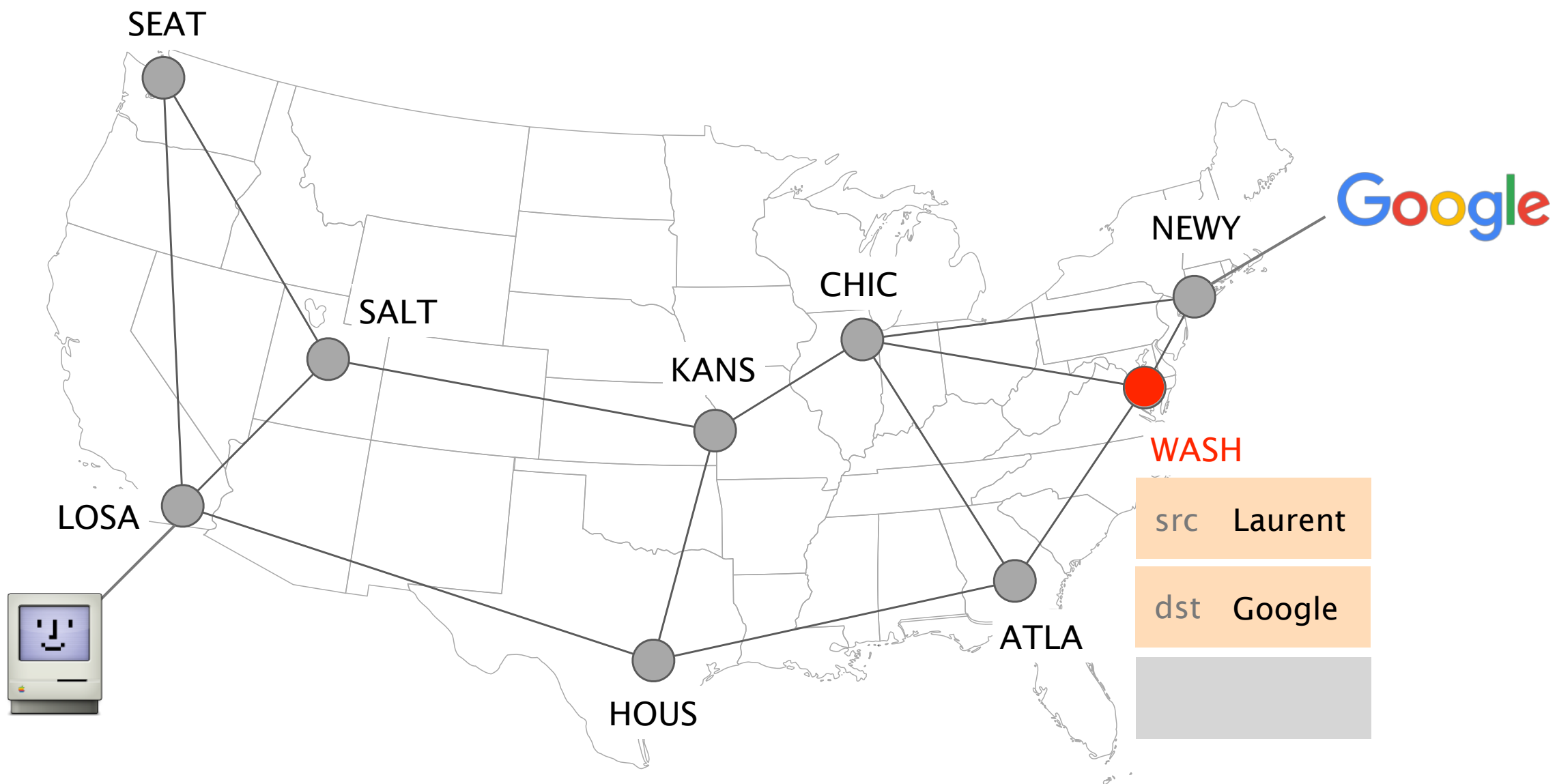


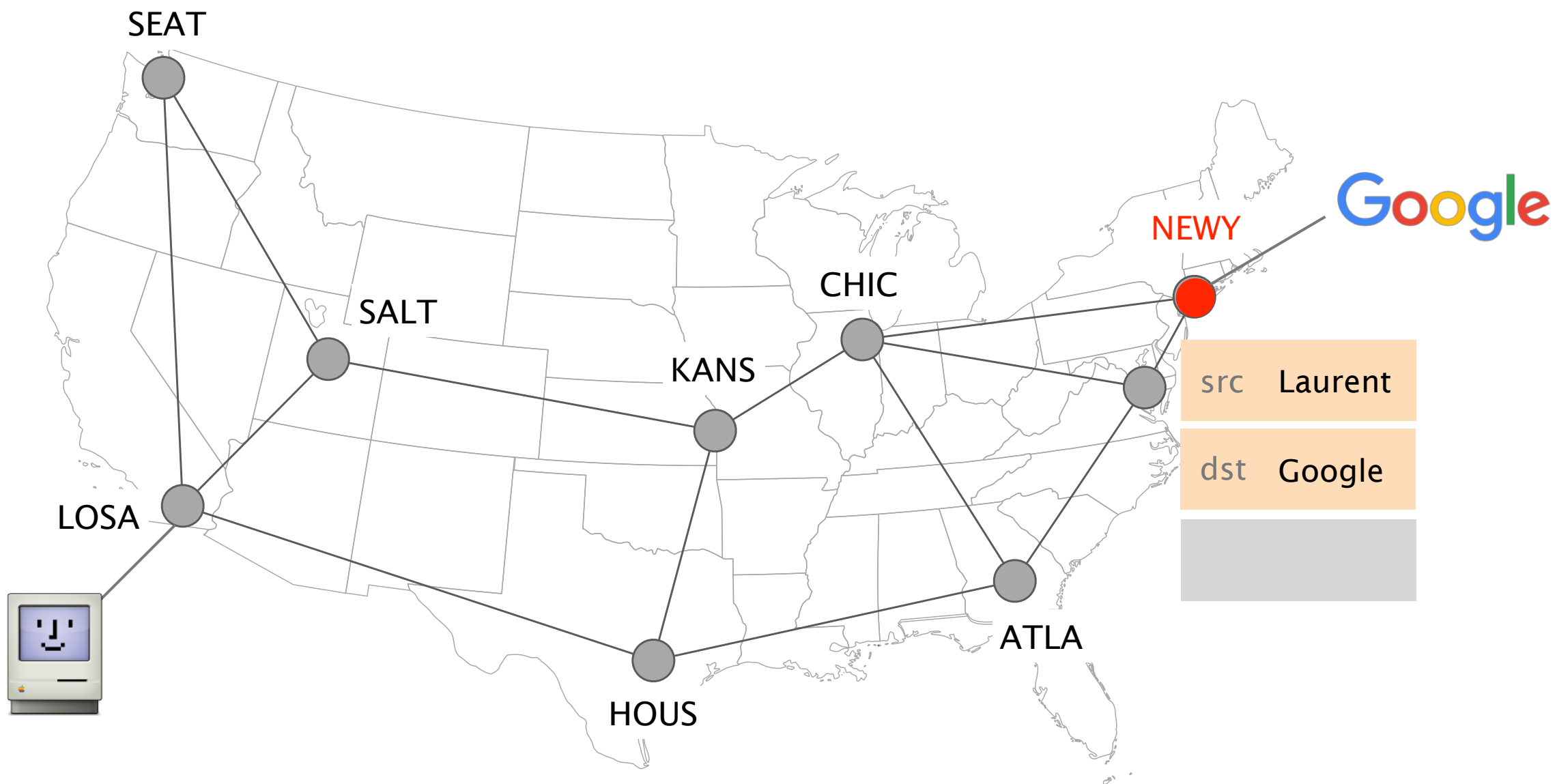


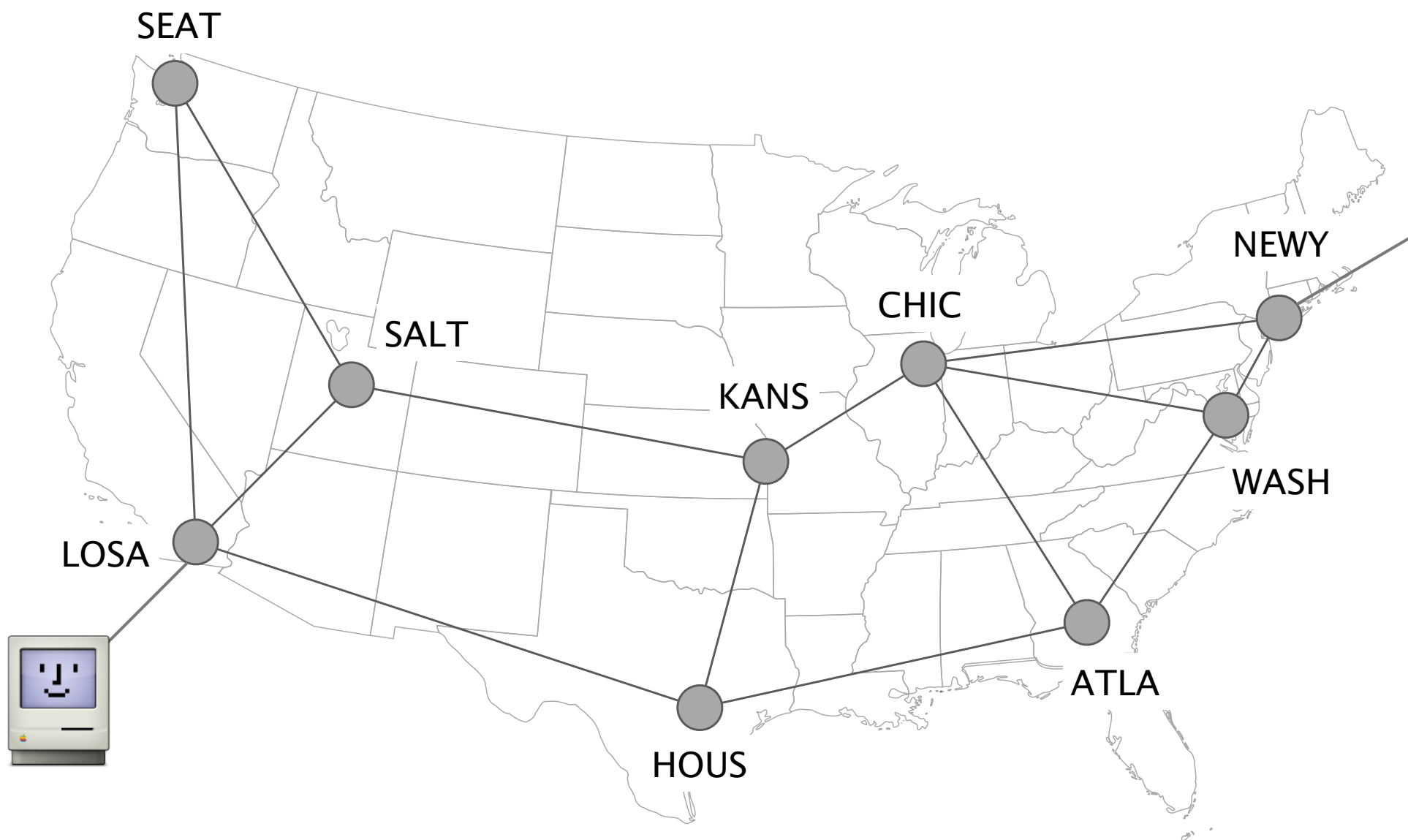










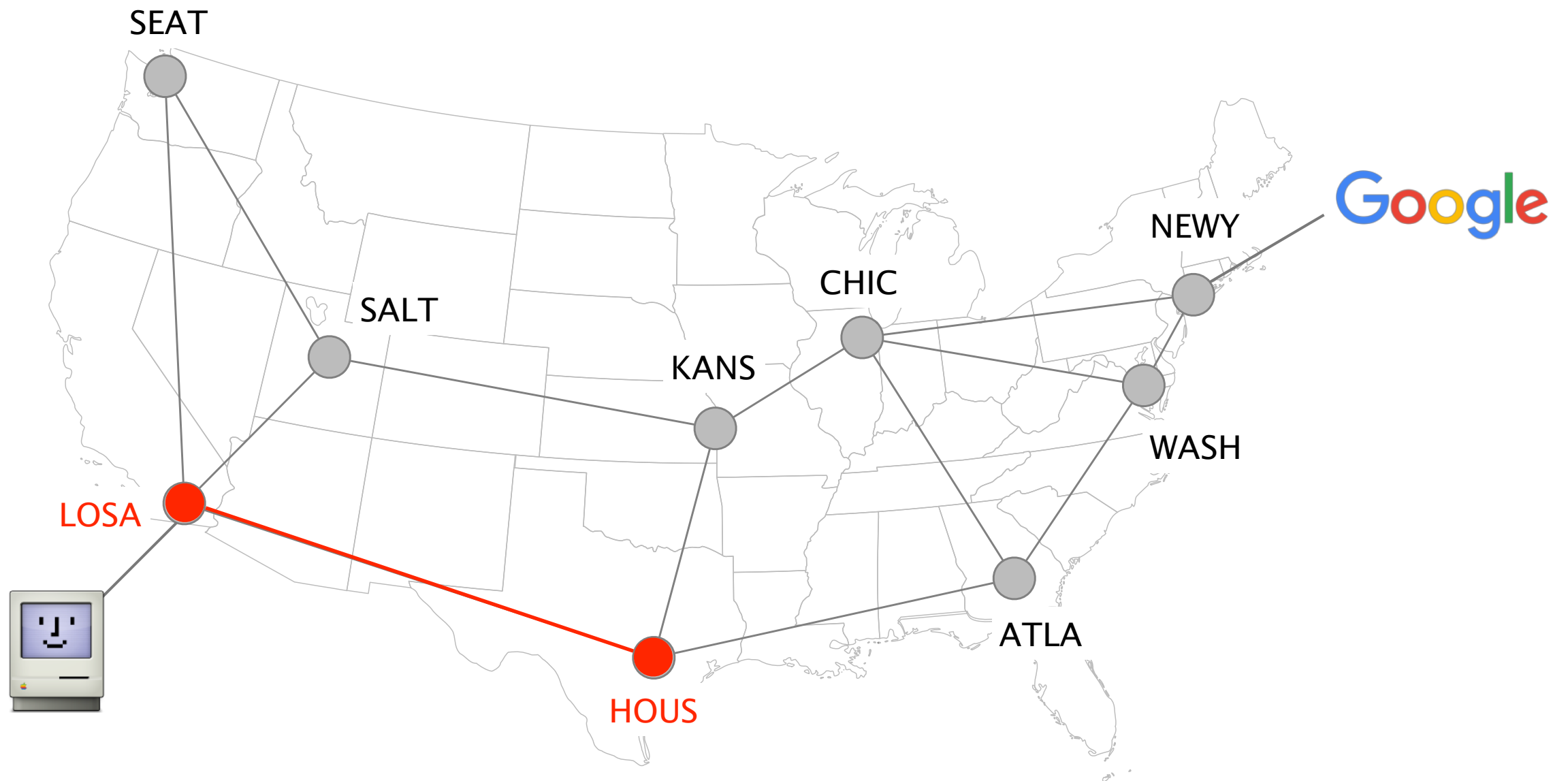


Google

src Laurent

dst Google

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



LOSA

IF#2

IF#4

Data-Plane

IF#1

IF#3

HOUS

IF#2

IF#4

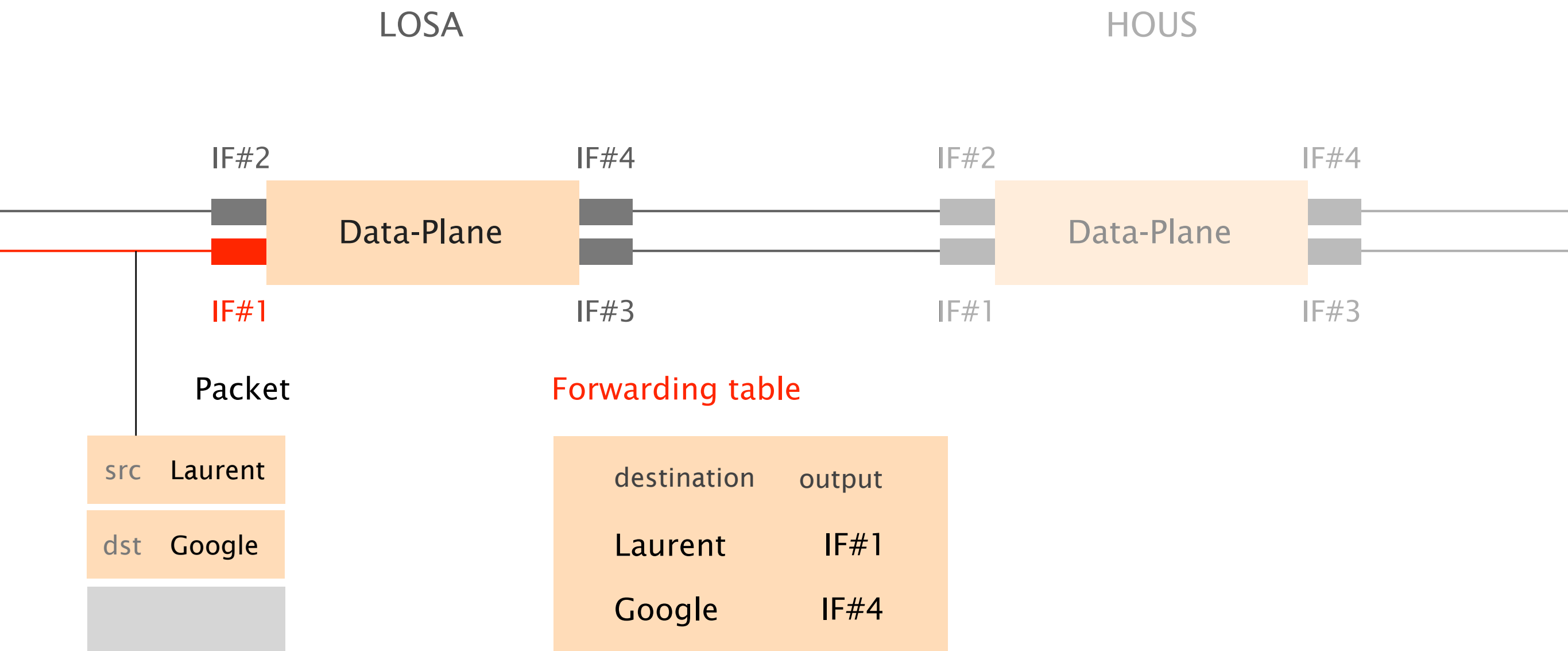
Data-Plane

IF#1

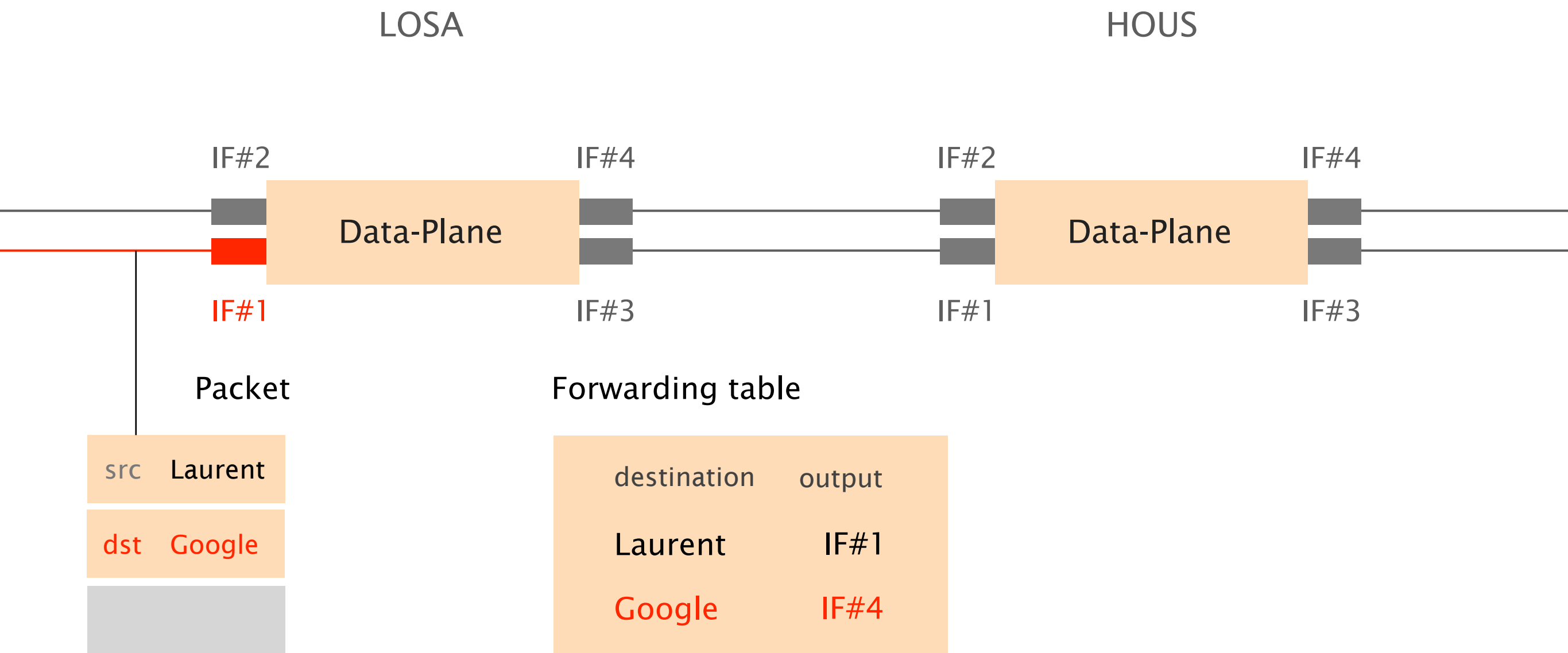
IF#3



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

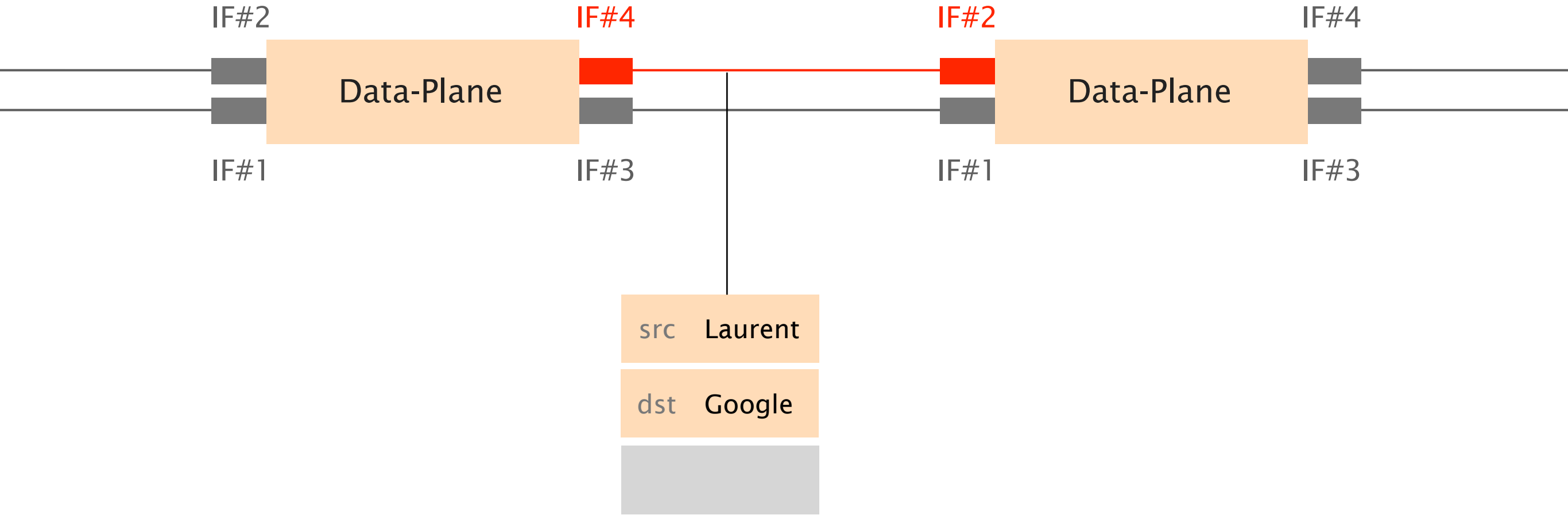


Here, the packet should be directed to **IF#4**



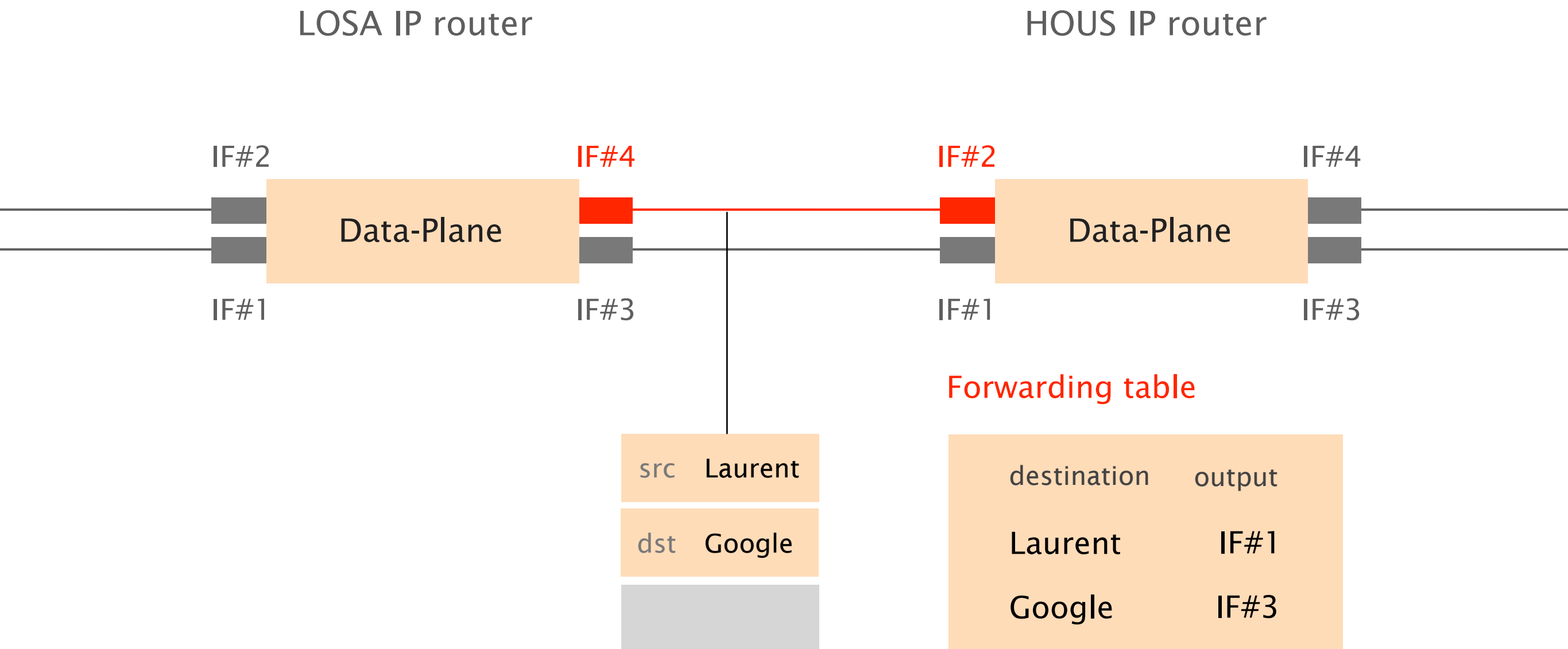
LOSA IP router

HOUS IP router



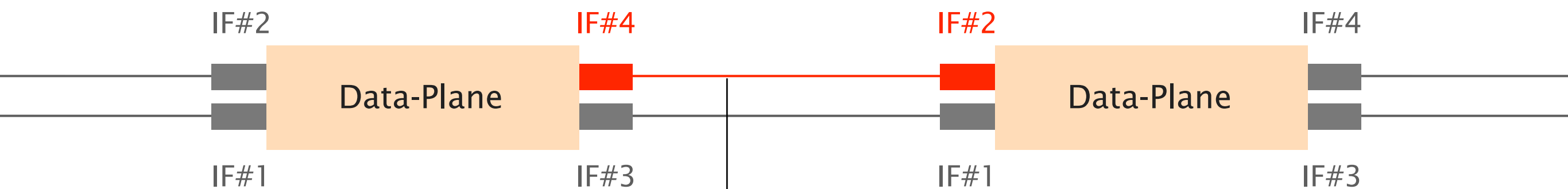


Forwarding is repeated at each router,  
until the destination is reached



LOSA IP router

HOUS IP router



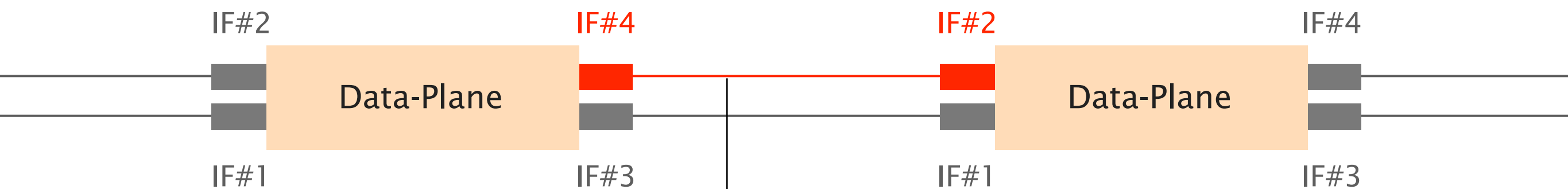
src	Laurent
dst	Google

Forwarding table

destination	output
Laurent	IF#1
Google	IF#3

LOSA IP router

HOUS IP router



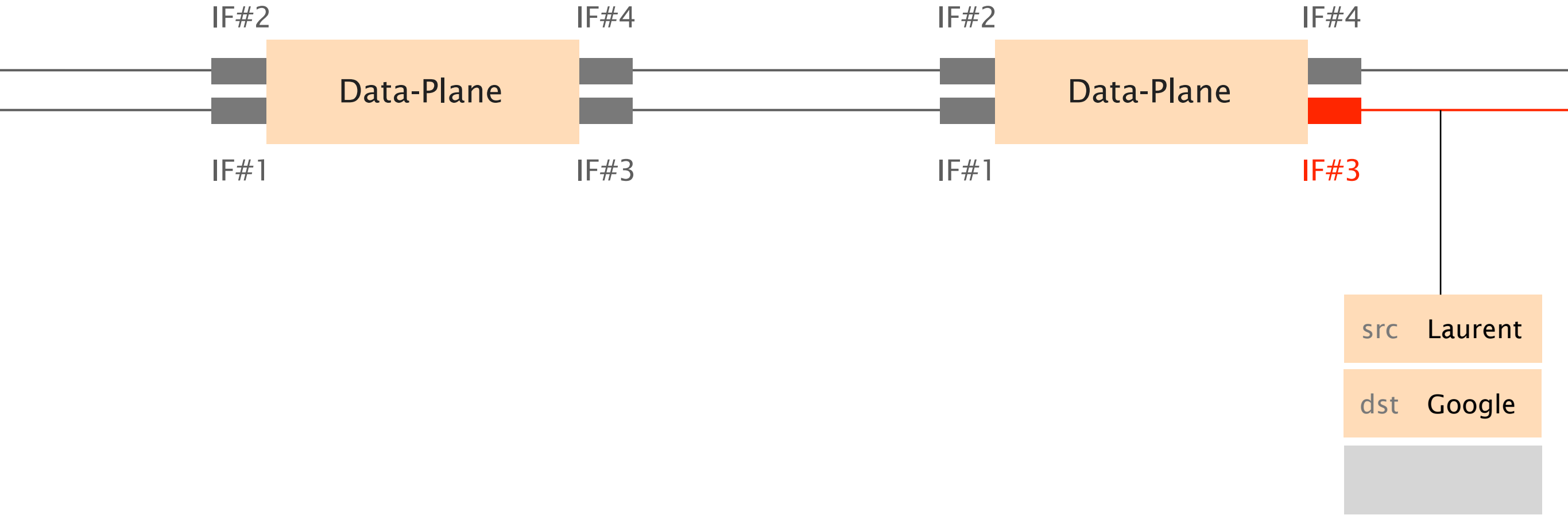
src	Laurent
dst	Google

Forwarding table

destination	output
Laurent	IF#1
Google	IF#3

LOSA IP router

HOUS IP router



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

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

the default in the Internet

# Where are these forwarding tables coming from?

LOSA IP router



Forwarding table

destination	output
Laurent	IF#1
Google	IF#4

HOUS IP router



Forwarding table

destination	output
Laurent	IF#1
Google	IF#3





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

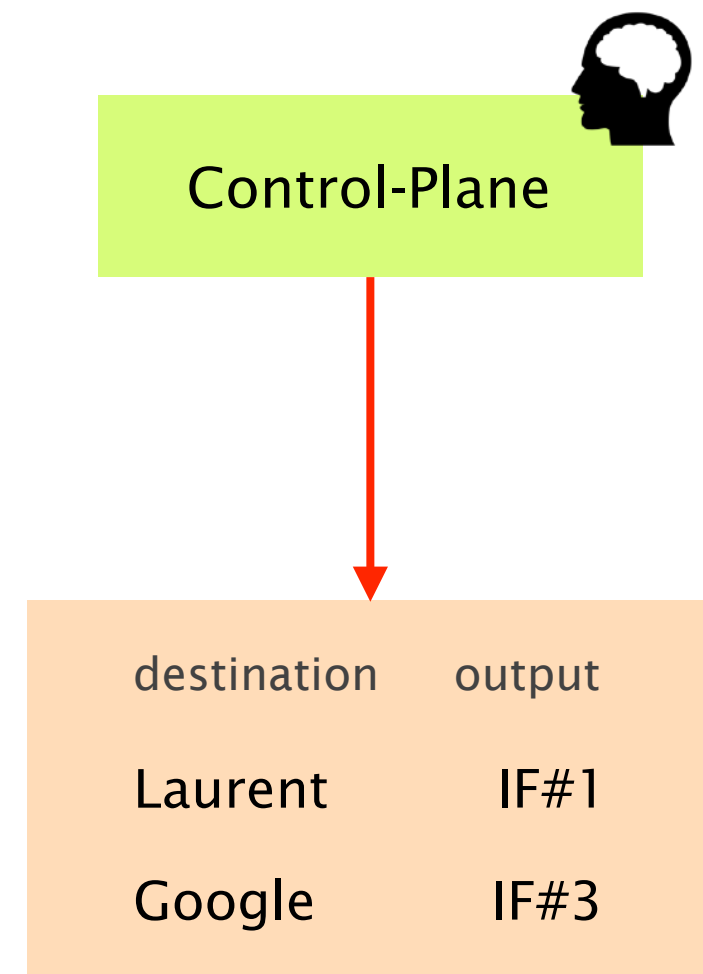
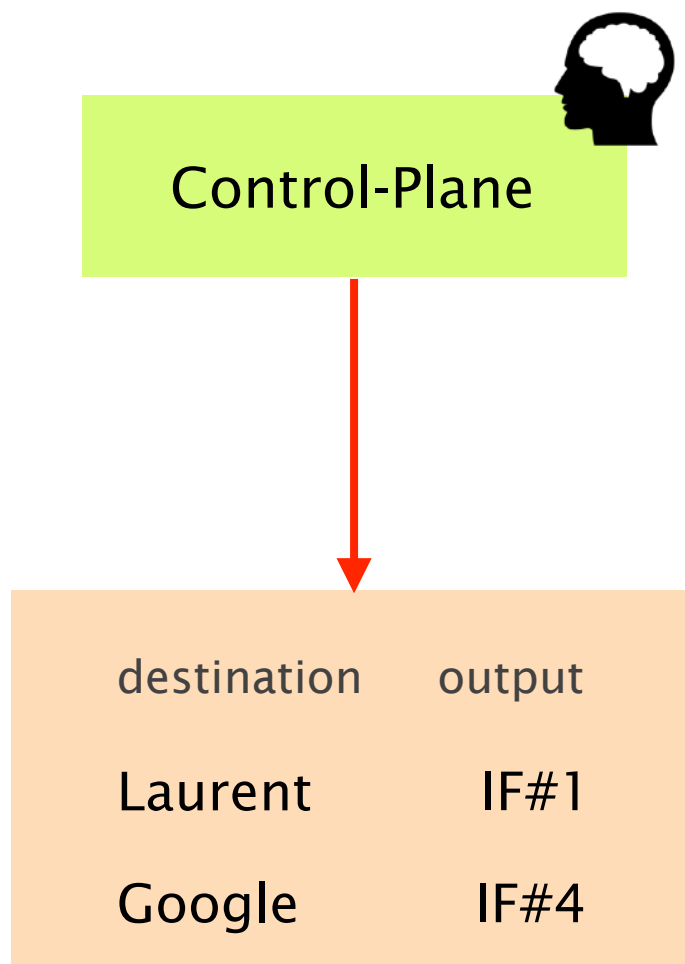
Routing

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  
always

timescale

nanoseconds

10s of ms  
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

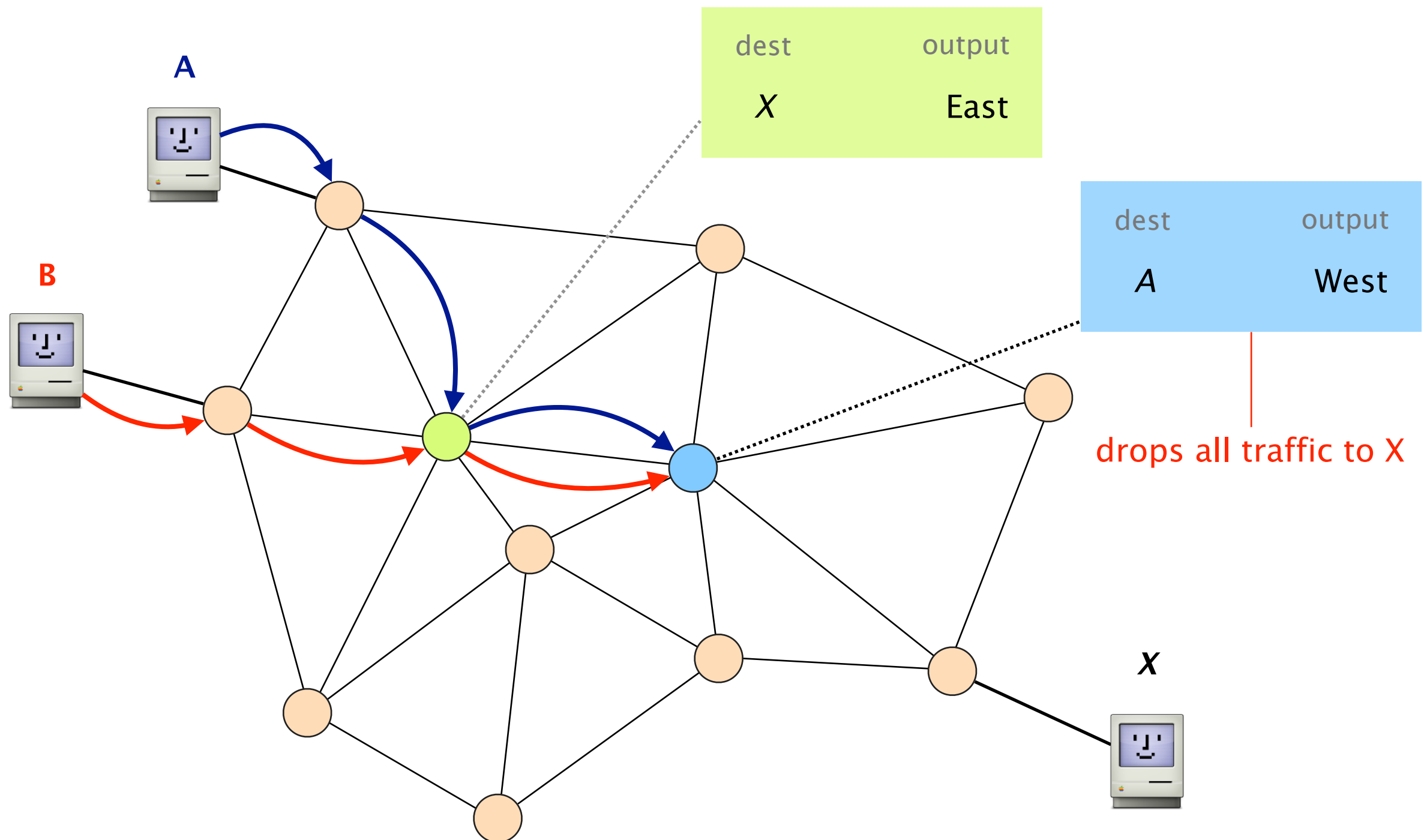
sufficient and necessary condition

Theorem

a global forwarding state is valid if and only if

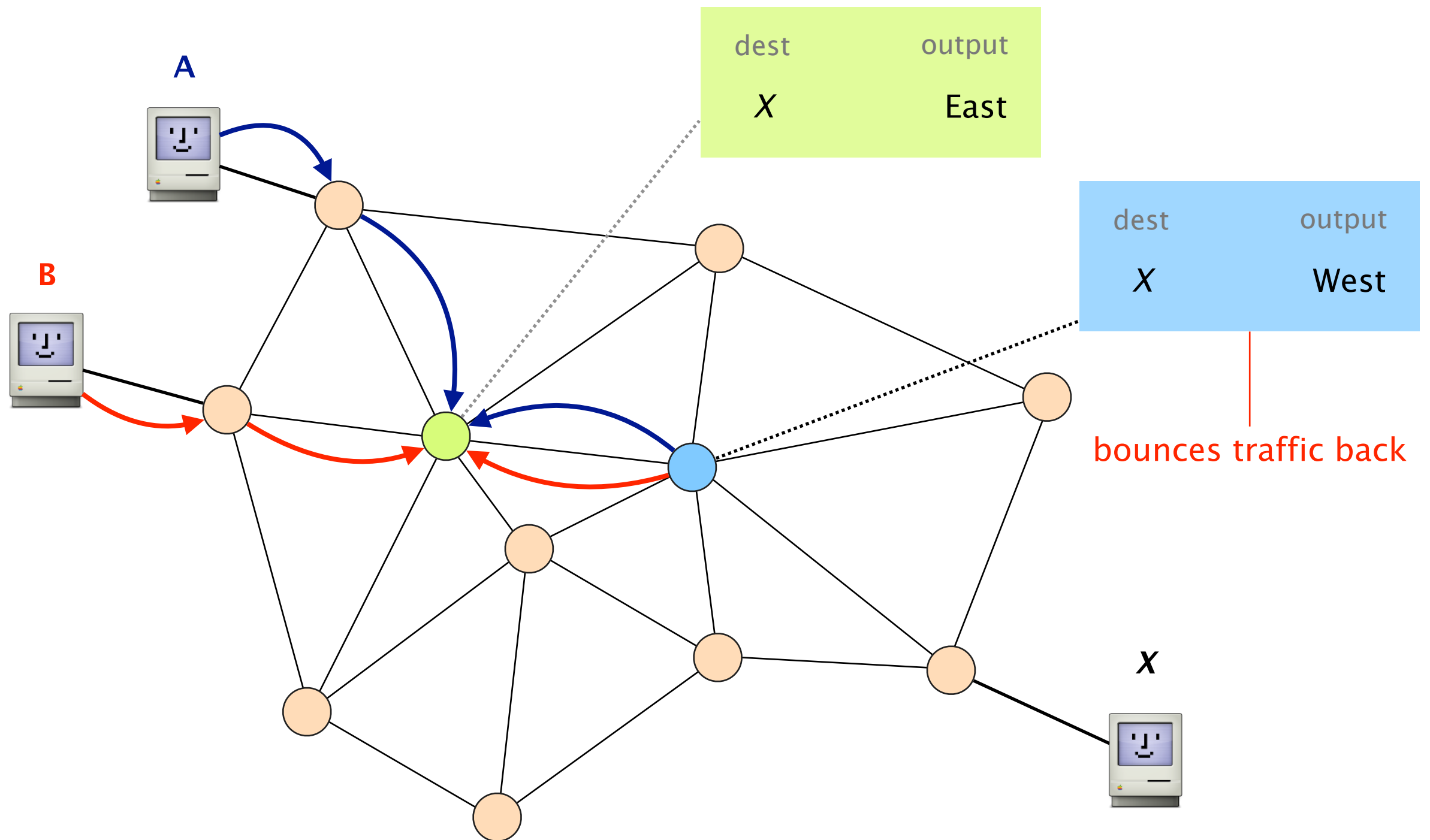
- there are no dead ends  
no outgoing port defined in the table
- there are no loops  
packets going around the same set of nodes

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**



# 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