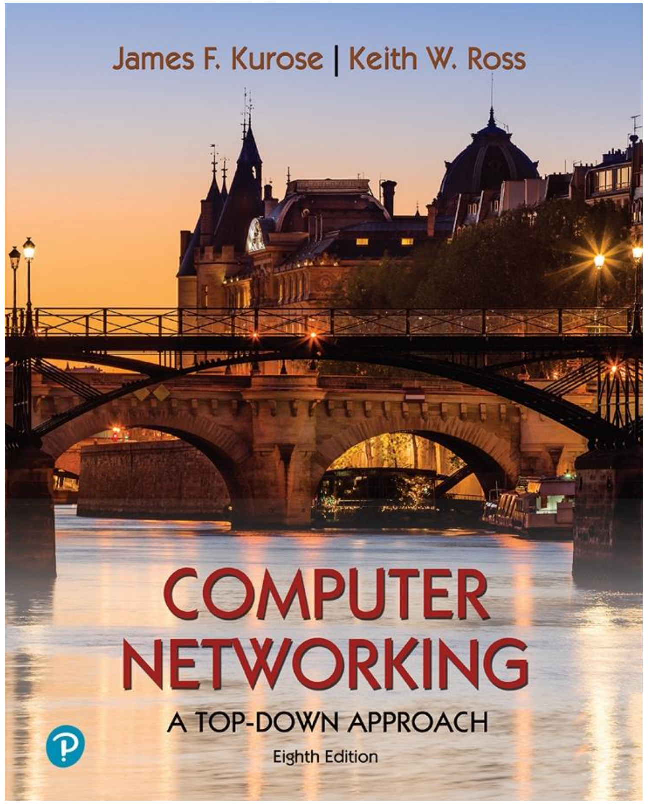
Chapter 1
Introduction

Yaxiong Xie

Department of Computer Science and Engineering
University at Buffalo, SUNY

Adapted from the slides of the book's authors



Computer Networking: A Top-Down Approach
8th edition
Jim Kurose, Keith Ross
Pearson, 2020

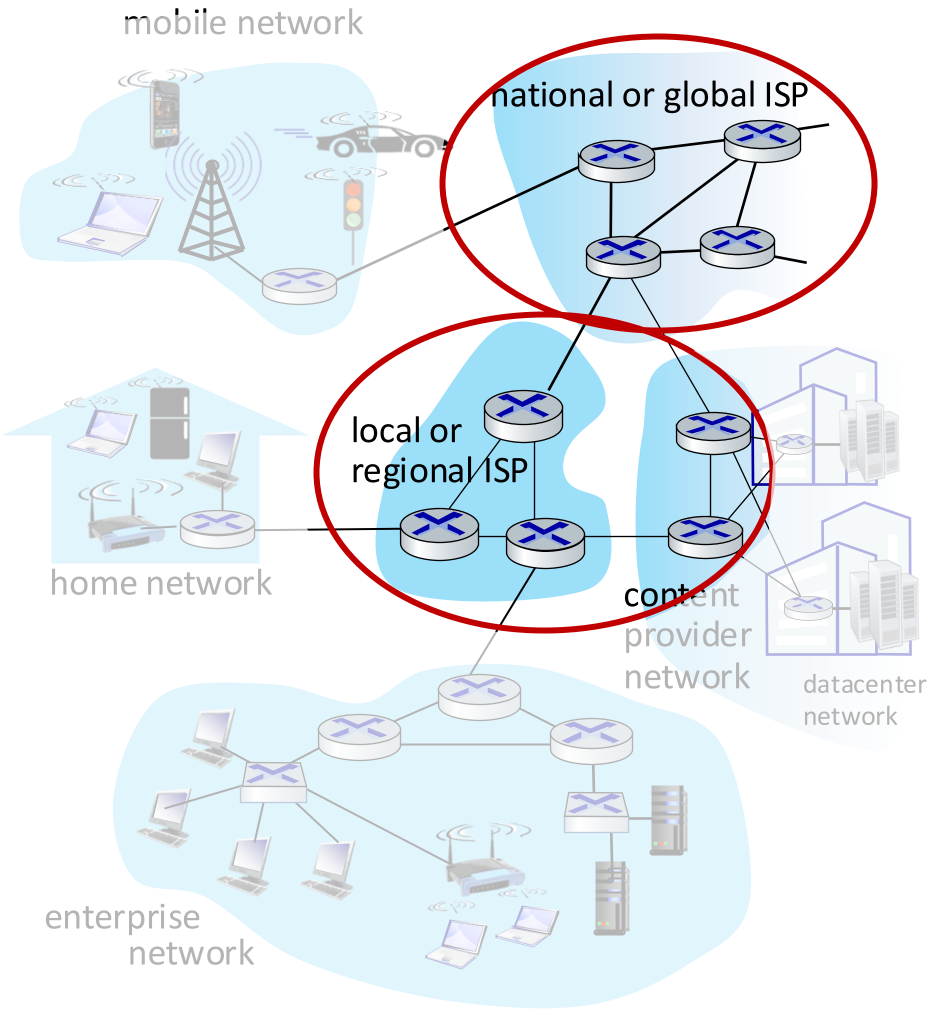
Chapter 1: roadmap

- What *is* the Internet?
- What *is* a protocol?
- Network edge: hosts, access network, physical media
- Network core: internet structure, routing and forwarding
- Performance: loss, delay, throughput
- Security
- Protocol layers, service models
- History

The network core

- mesh of interconnected routers

Ok, but WHY?

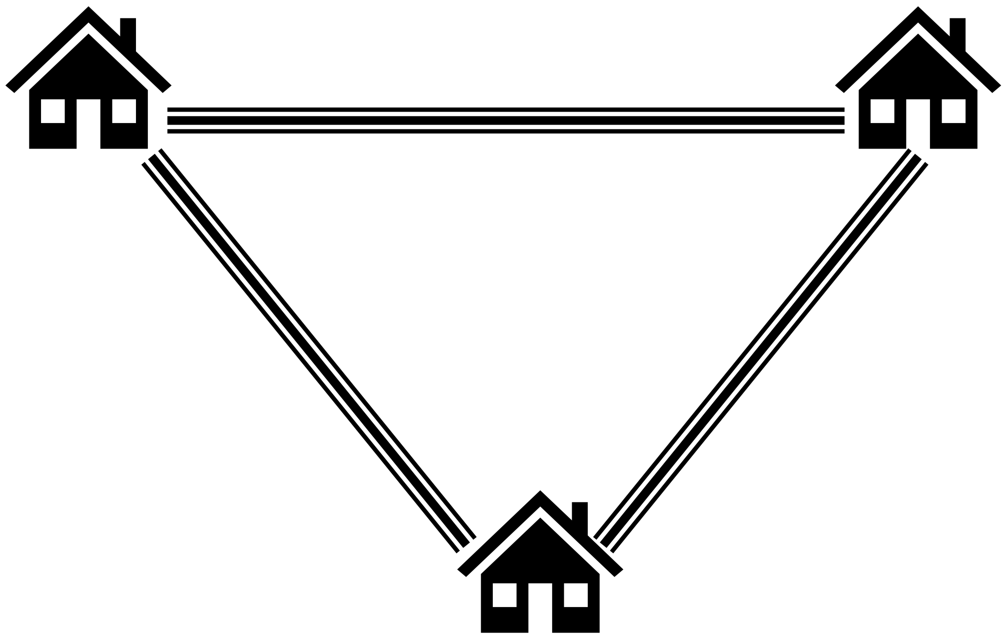


The network core

- mesh of interconnected routers

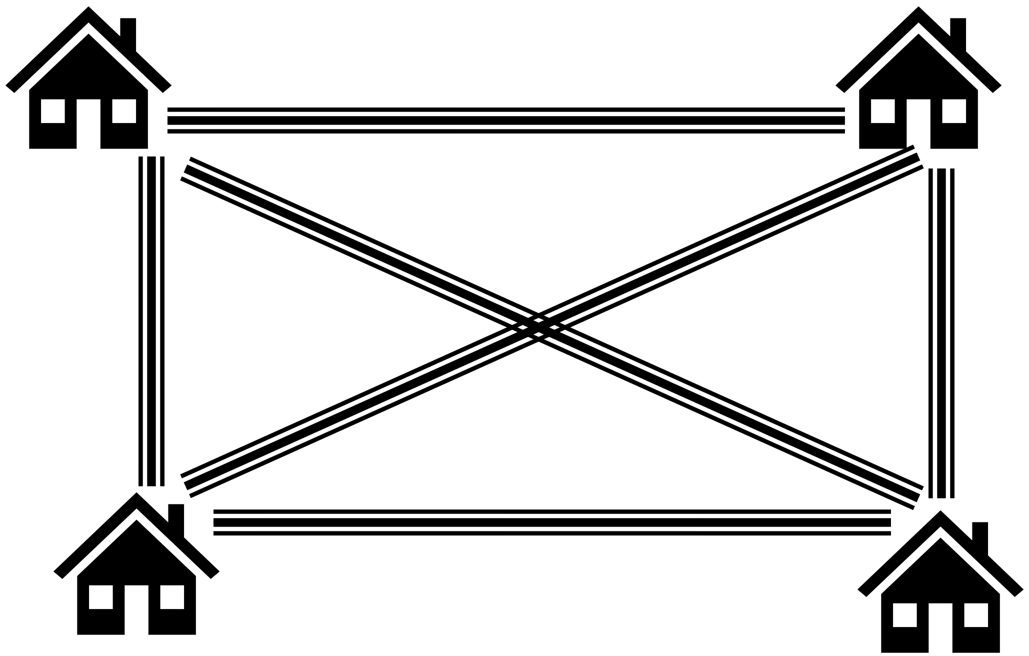
The network core

- mesh of interconnected routers



The network core

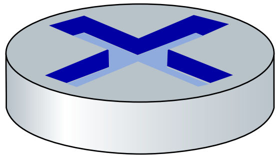
- mesh of interconnected routers

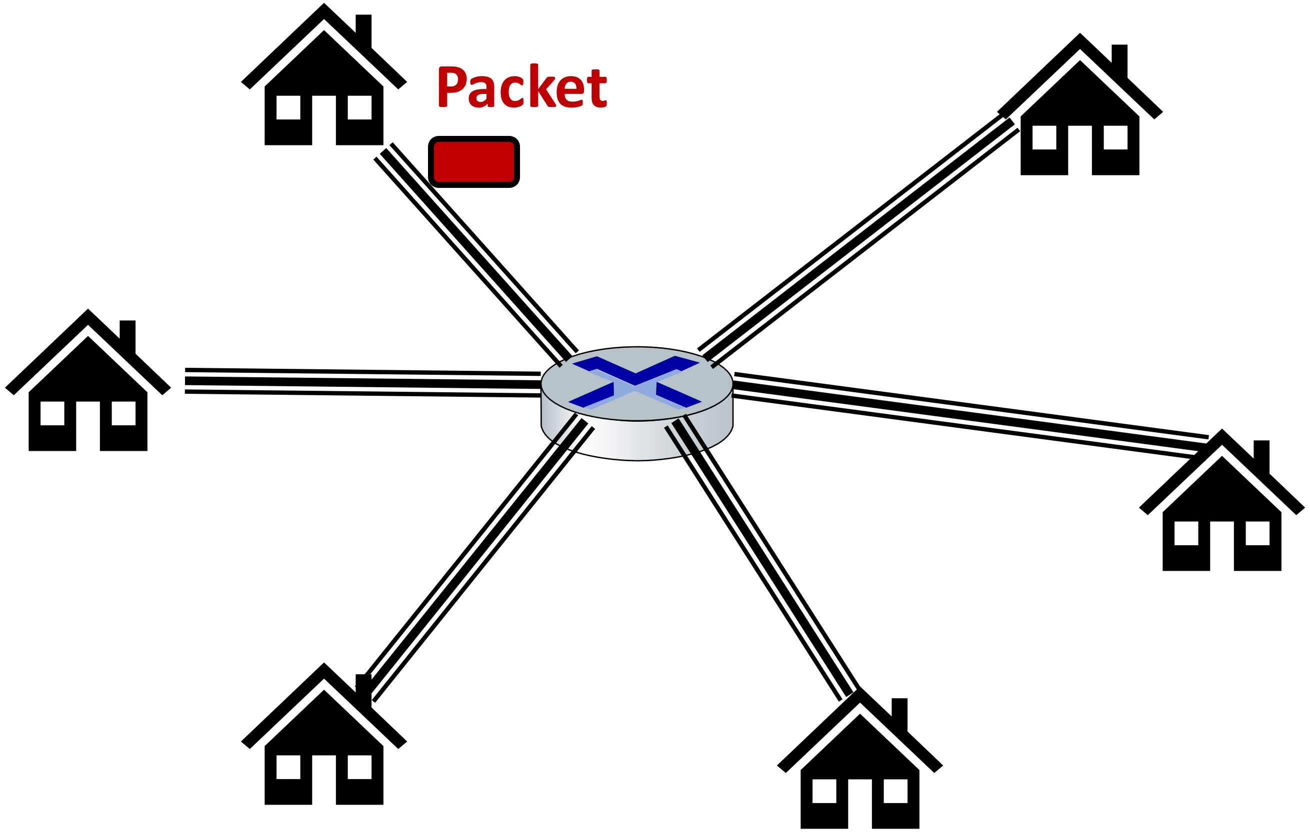


The network core

- mesh of interconnected routers

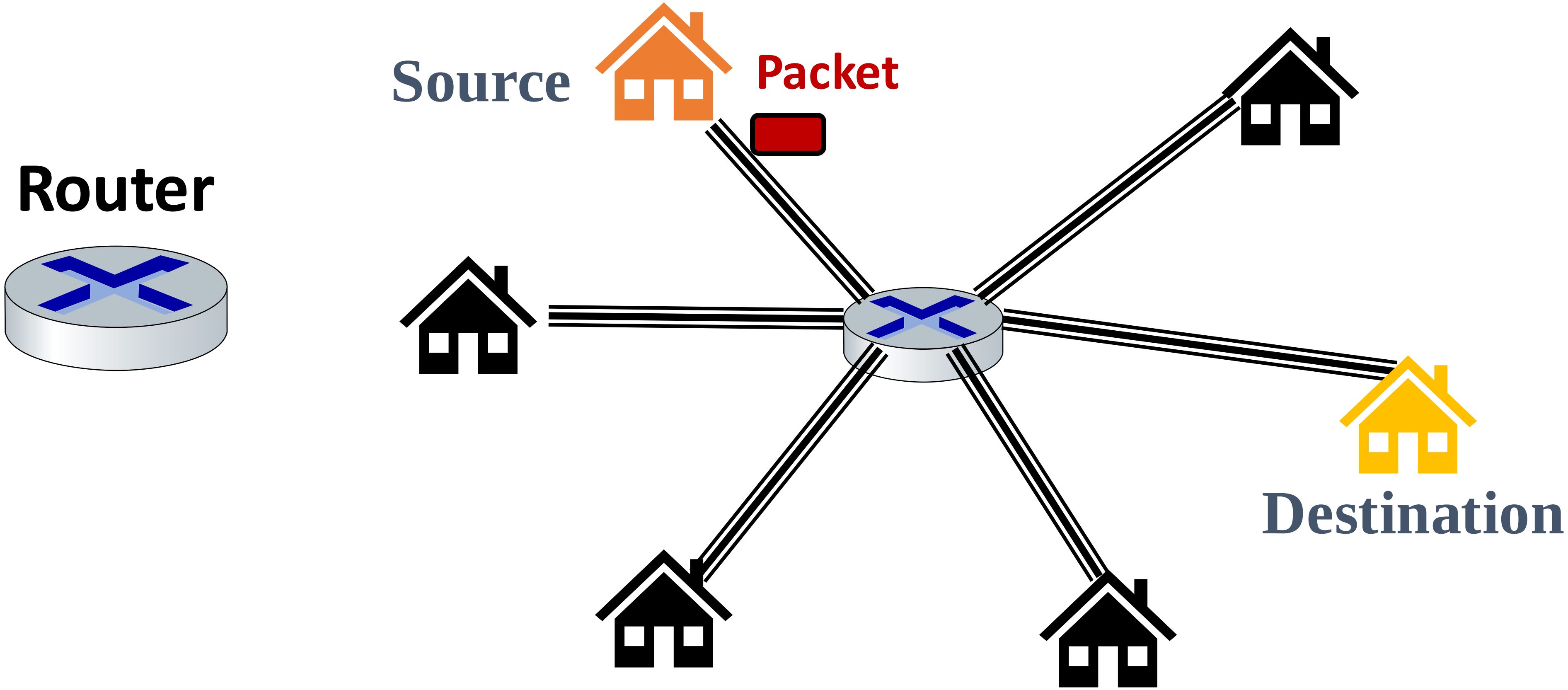
Router





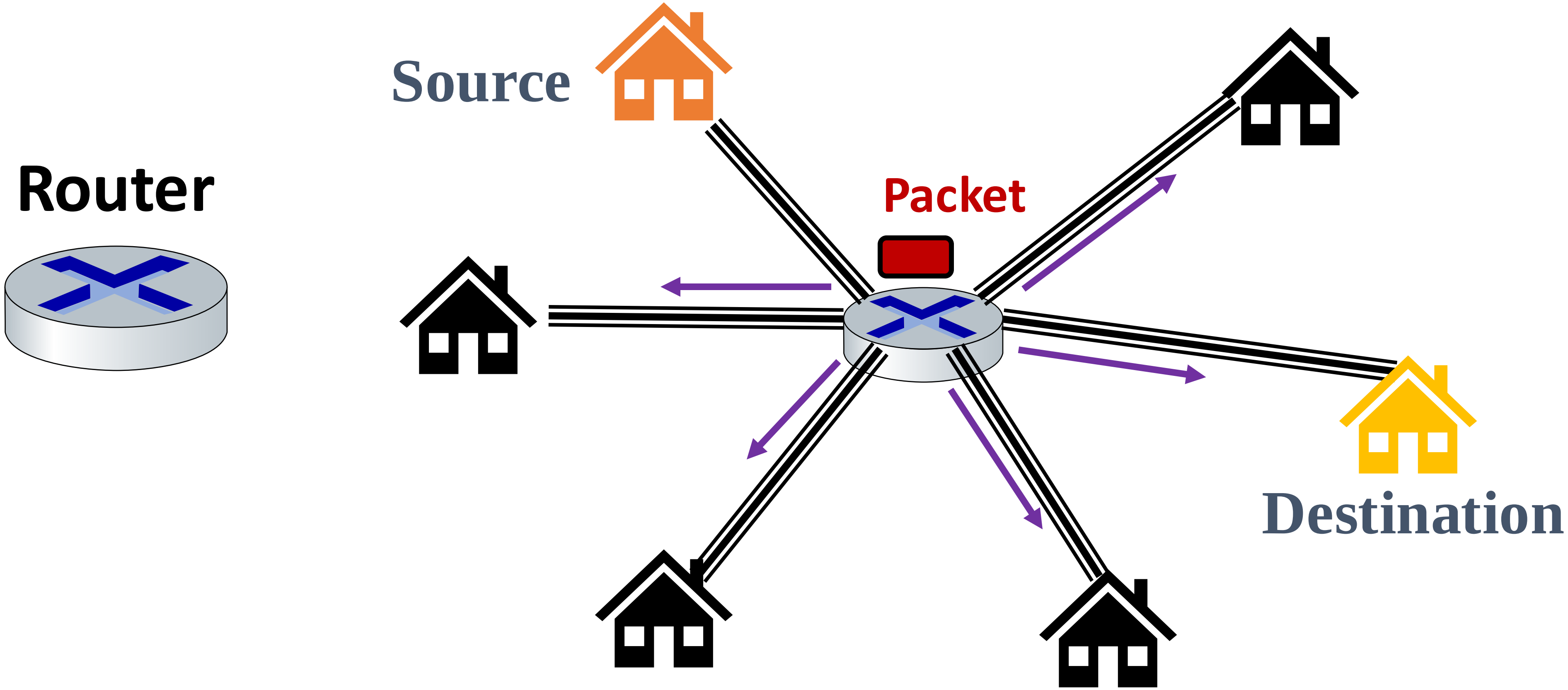
The network core

- mesh of interconnected routers



The network core

- mesh of interconnected routers

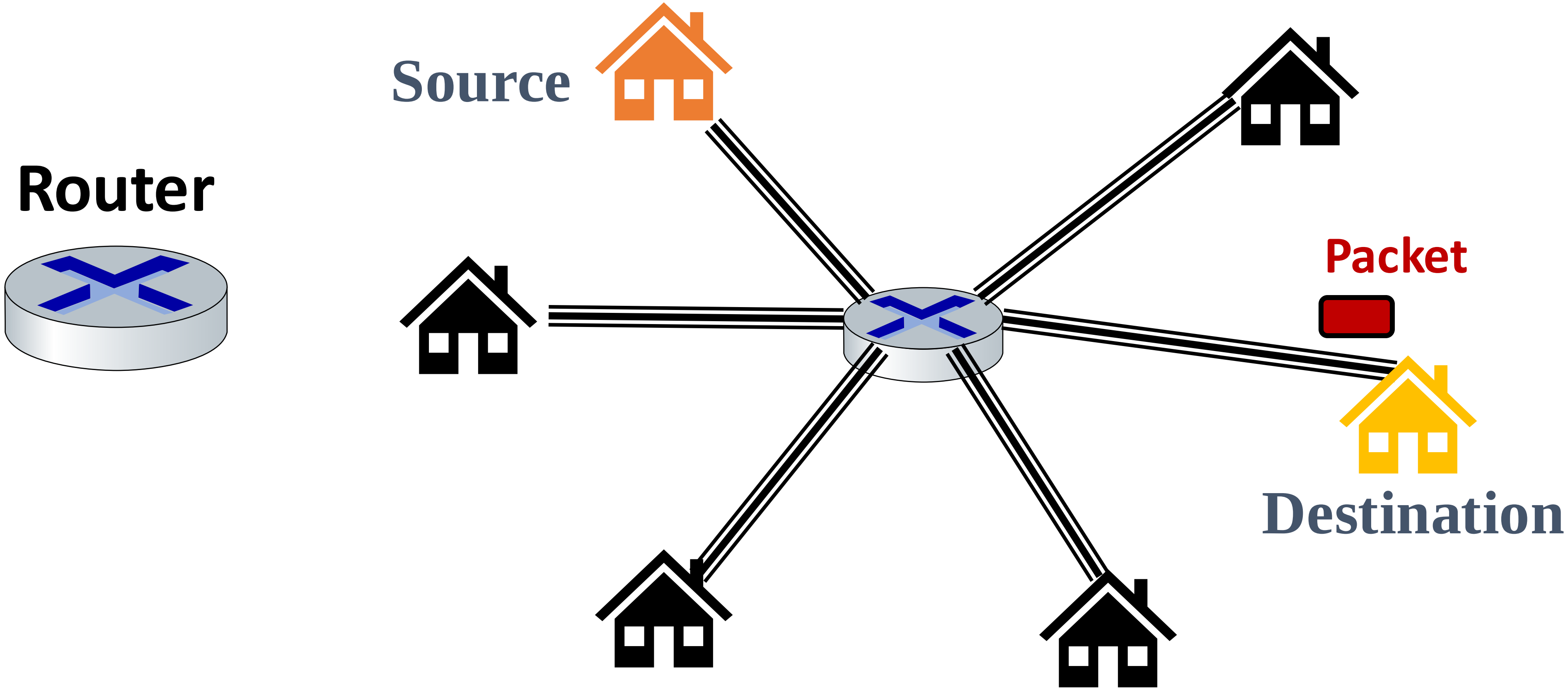


Forwarding:

- aka "switching"
- *local* action: move arriving packets from router's input link to appropriate router output link

The network core

- mesh of interconnected routers

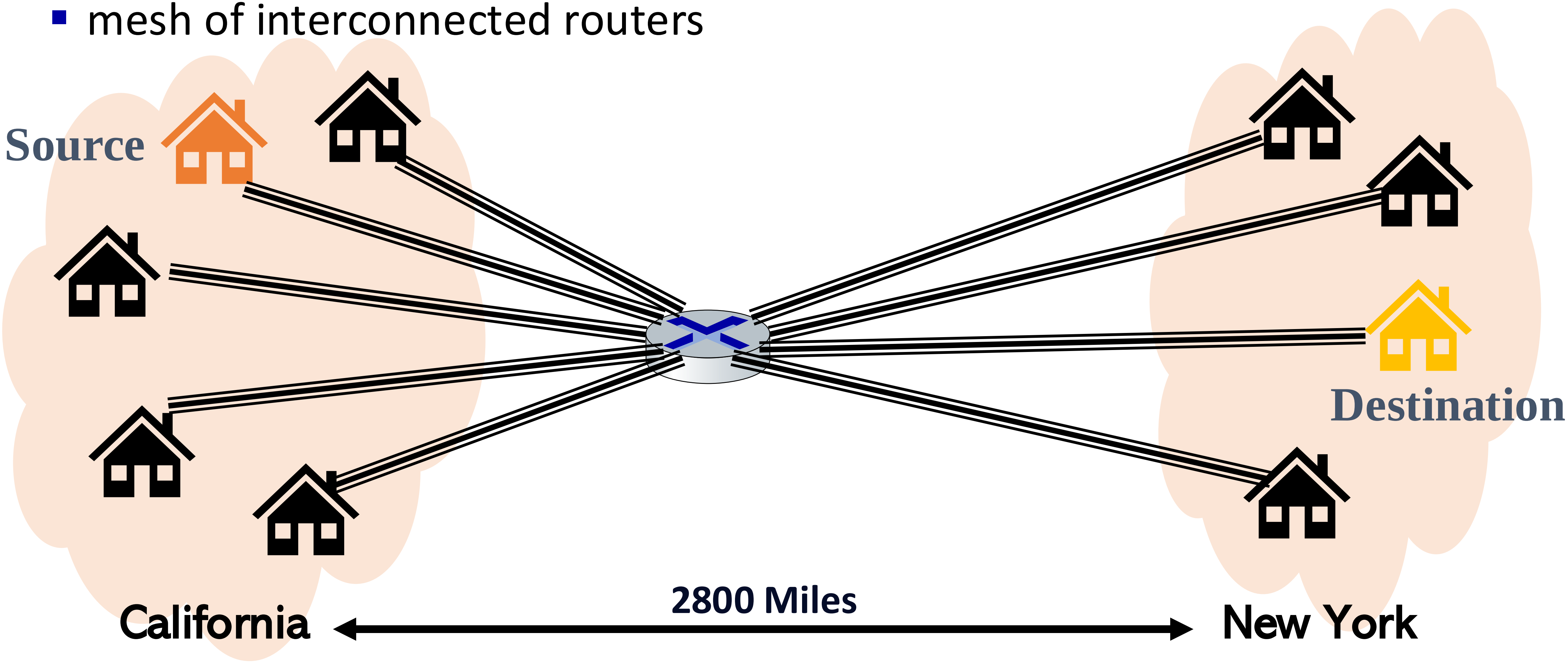


Forwarding:

- aka "switching"
- *local* action: move arriving packets from router's input link to appropriate router output link

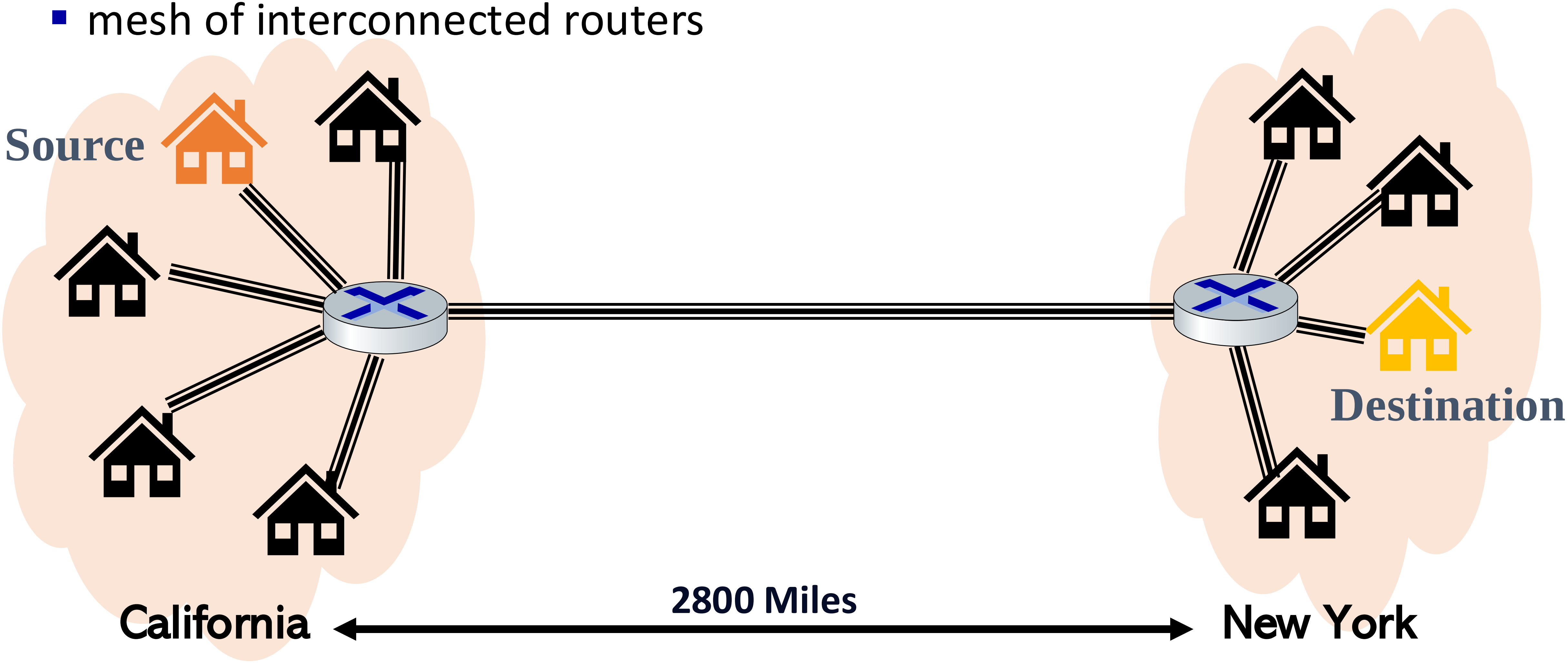
The network core

- mesh of interconnected routers



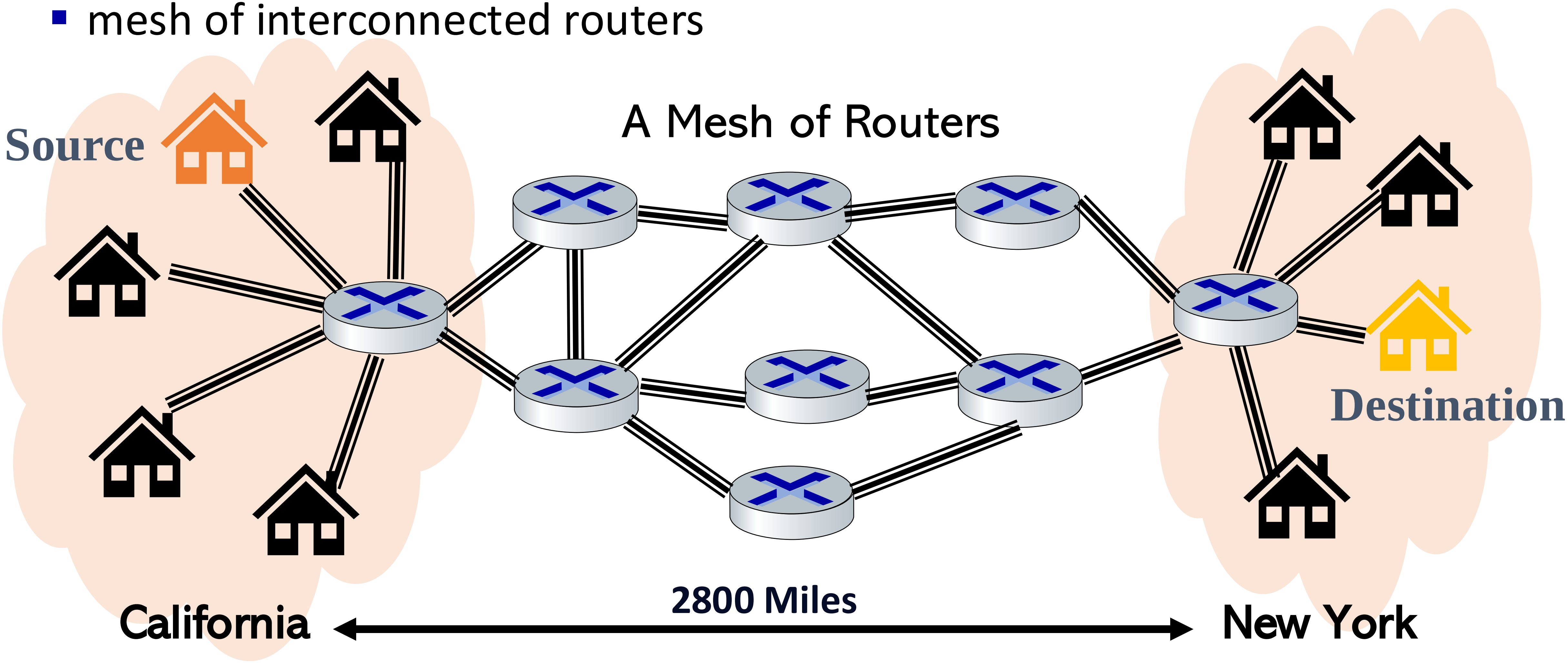
The network core

- mesh of interconnected routers



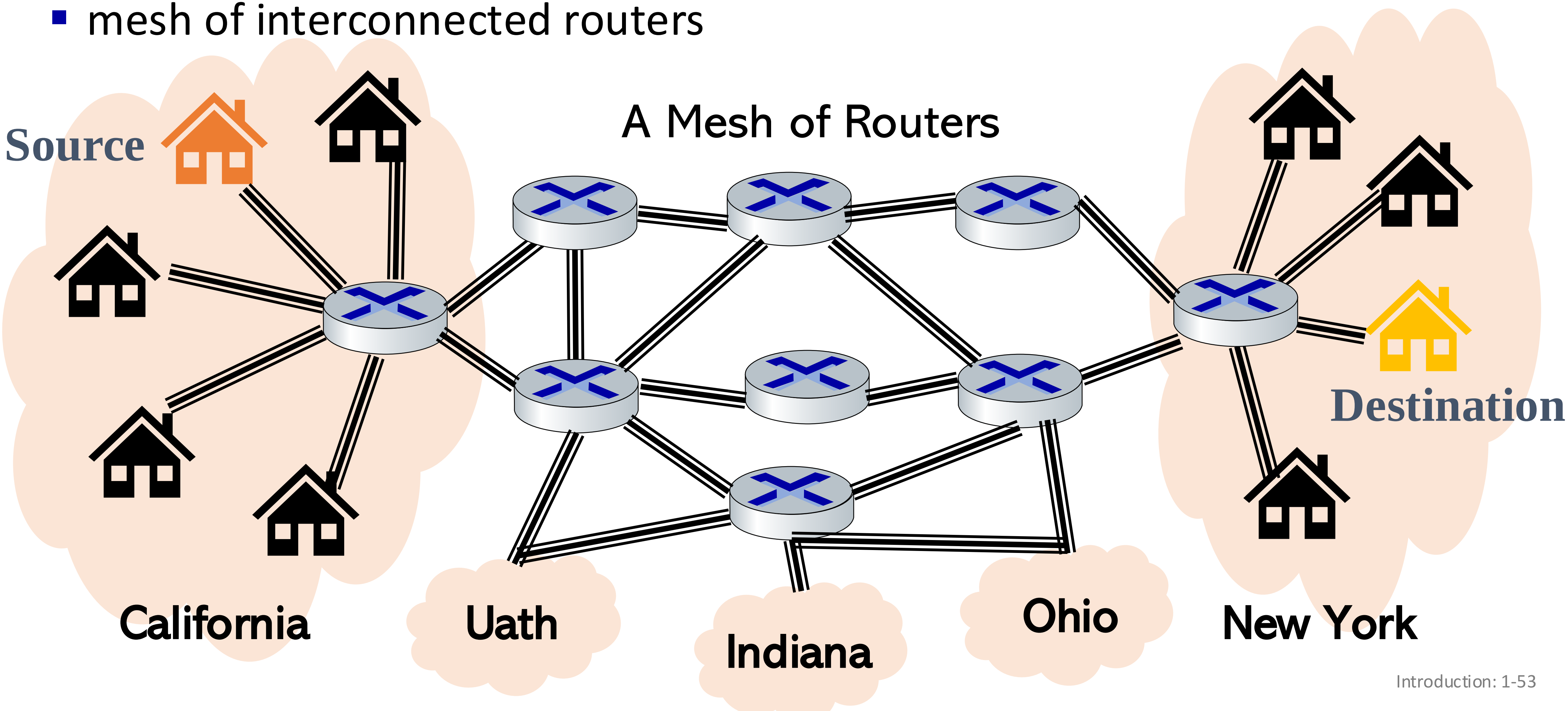
The network core

- mesh of interconnected routers



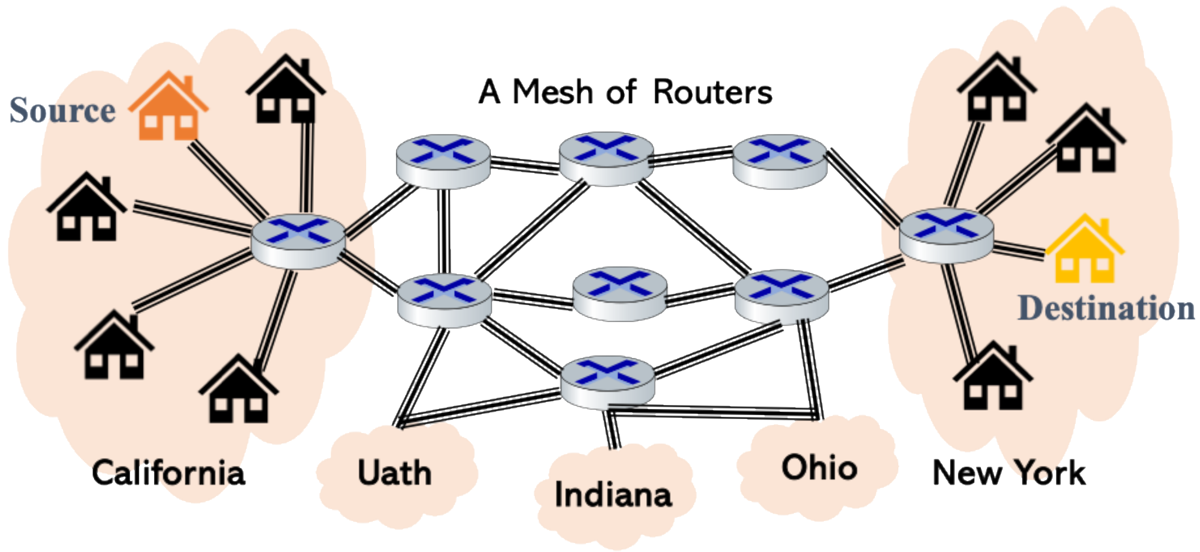
The network core

- mesh of interconnected routers



The network core

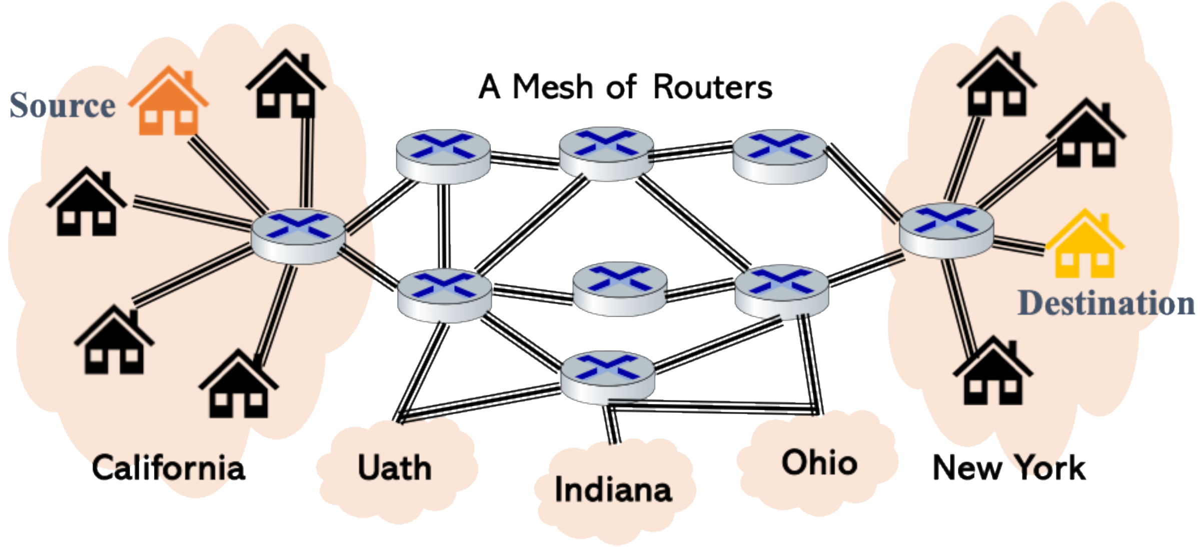
- mesh of interconnected routers



Aren't they very similar to each other?

The network core – Routing

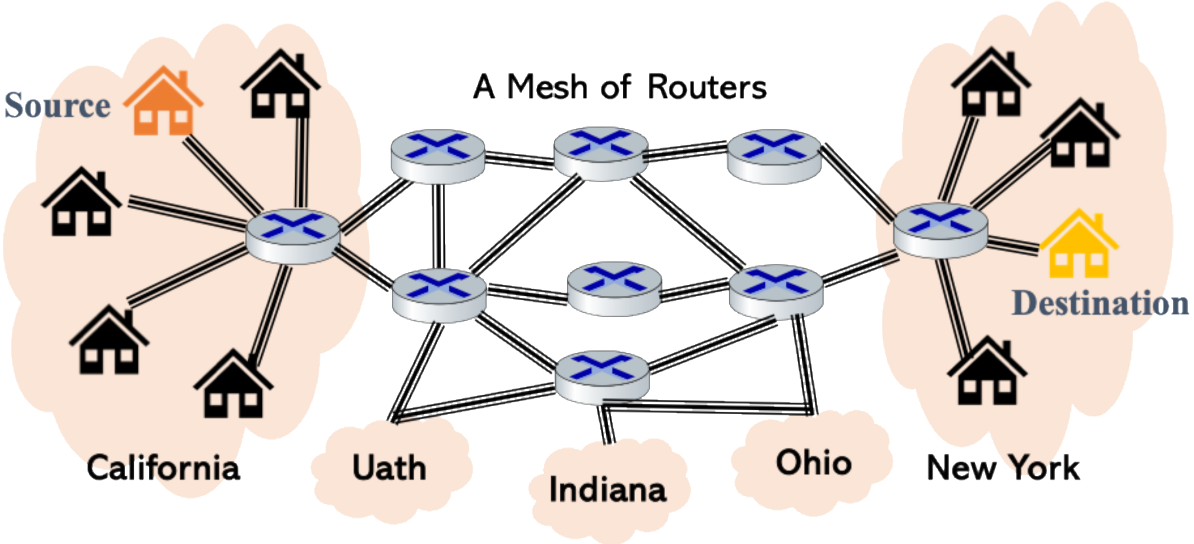
- Routing: Finding the Correct/Optimal path from source to destination



What's a correct/optimal path?

The network core – Routing

- Routing: Finding the Correct/Optimal path from source to destination



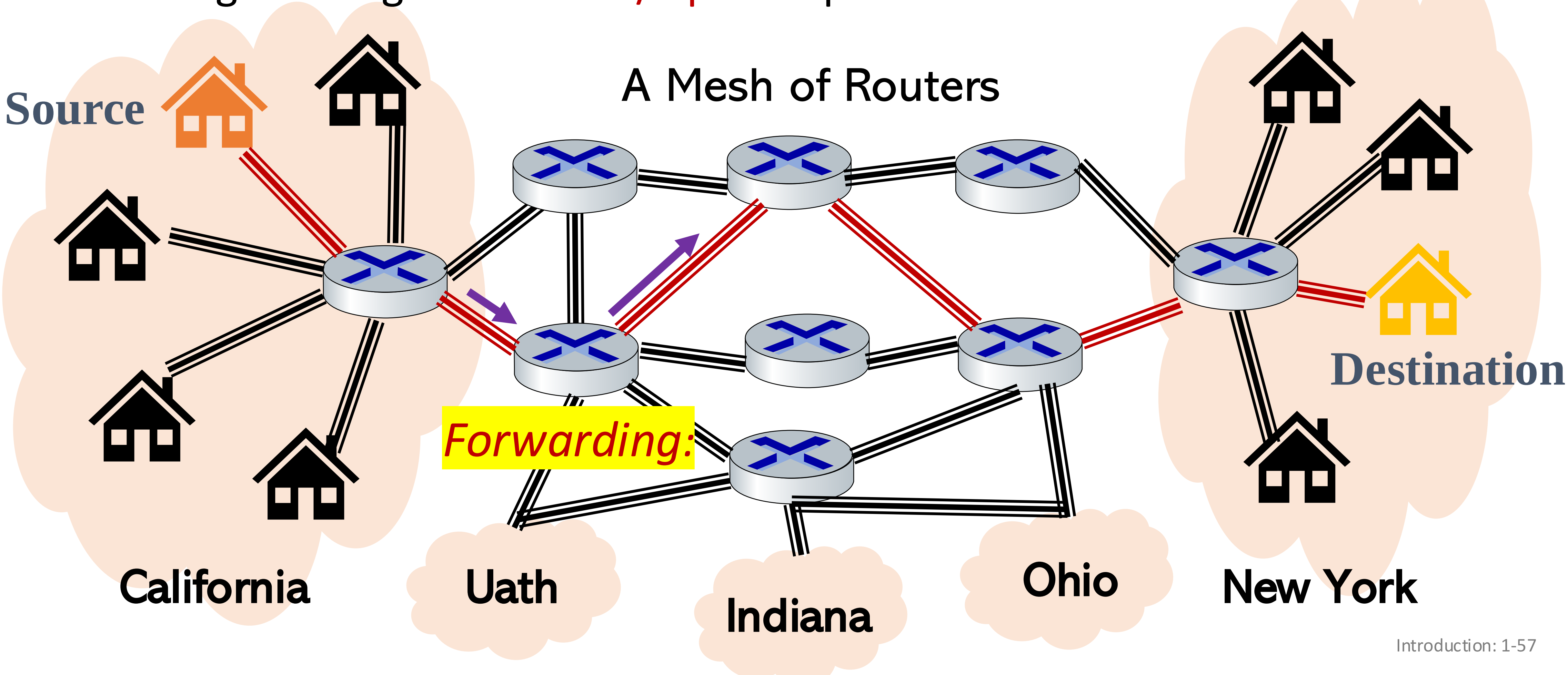
Routing Algorithm

❖ Shortest distance?

❖ Shortest distance?
❖ Cheapest without tolls?
❖ Best views?

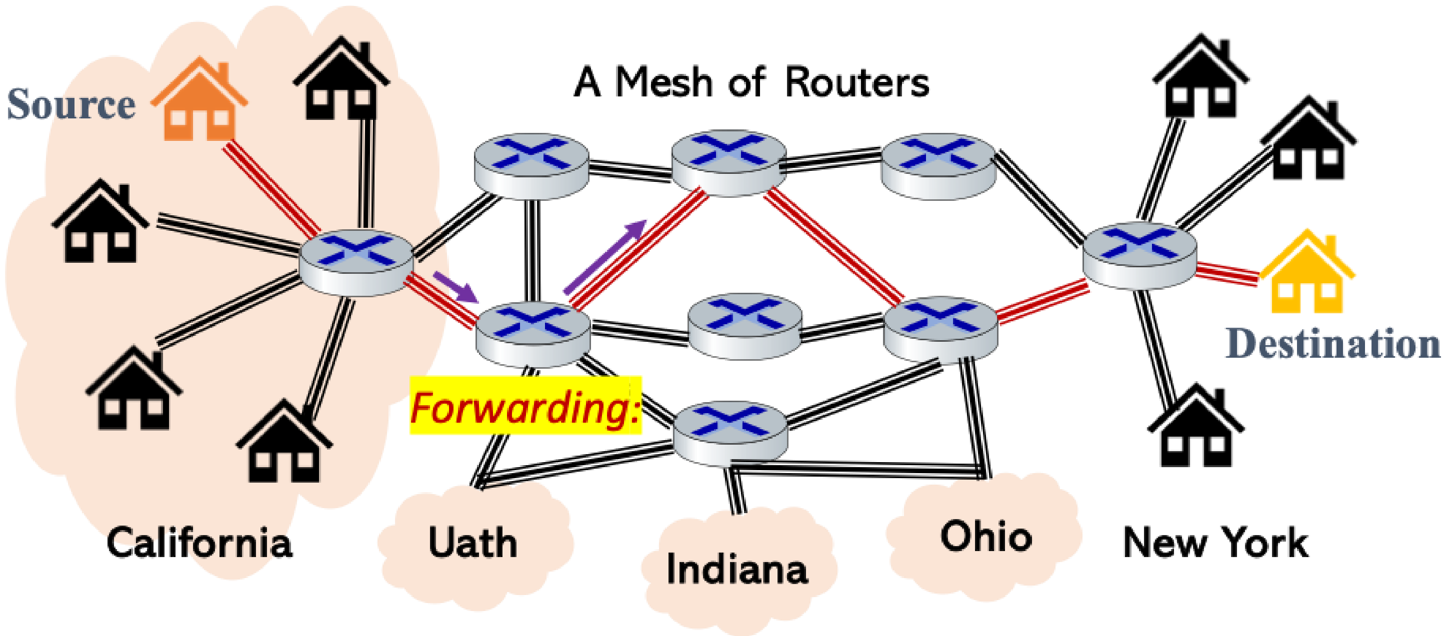
The network core – Routing

- Routing: Finding the Correct/Optimal path from source to destination



The network core – Routing

- Routing: Finding the Correct/Optimal path from source to destination



Forwarding:

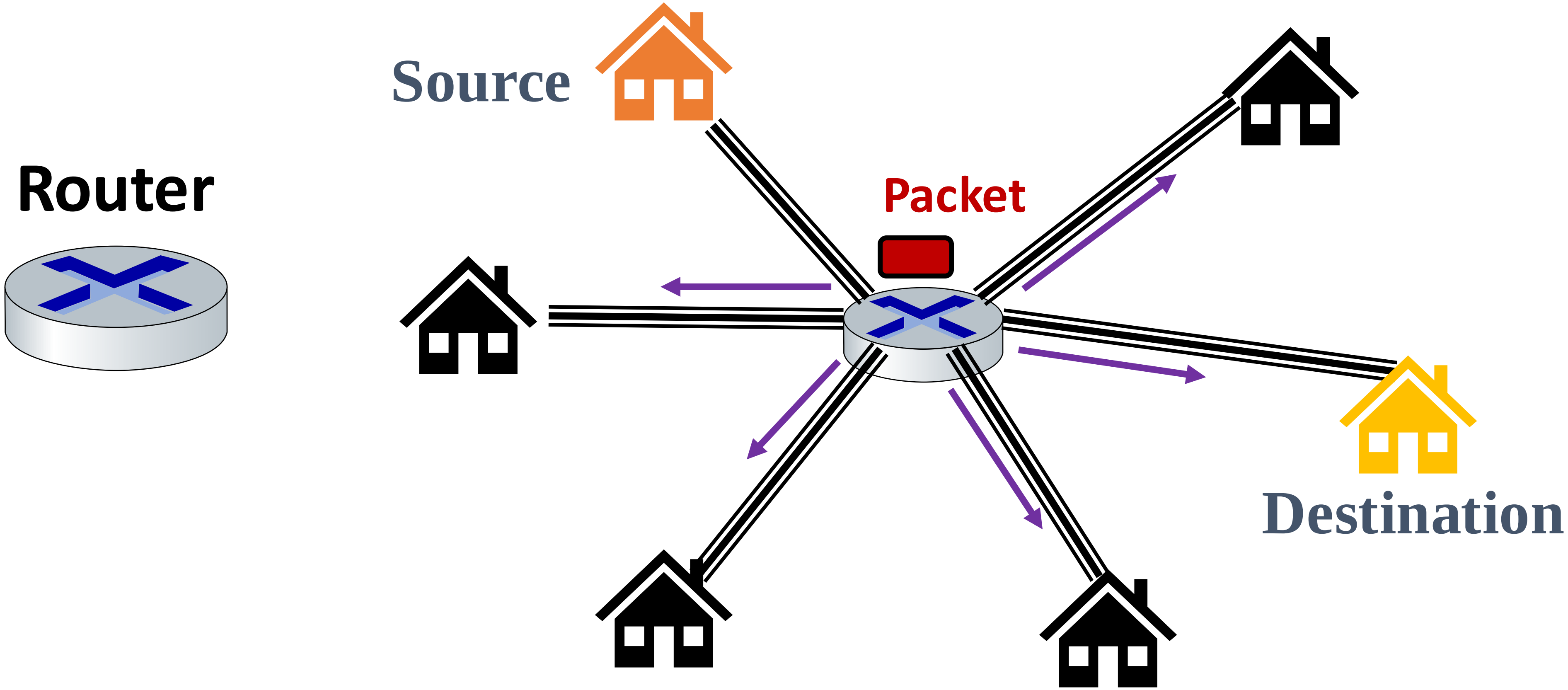
- *local* action: move arriving packets from router's input link to appropriate router output link

Routing:

- *global* action: determine source-destination paths taken by packets

Packet Switching VS Circuit Switching

- Forward is also called switching

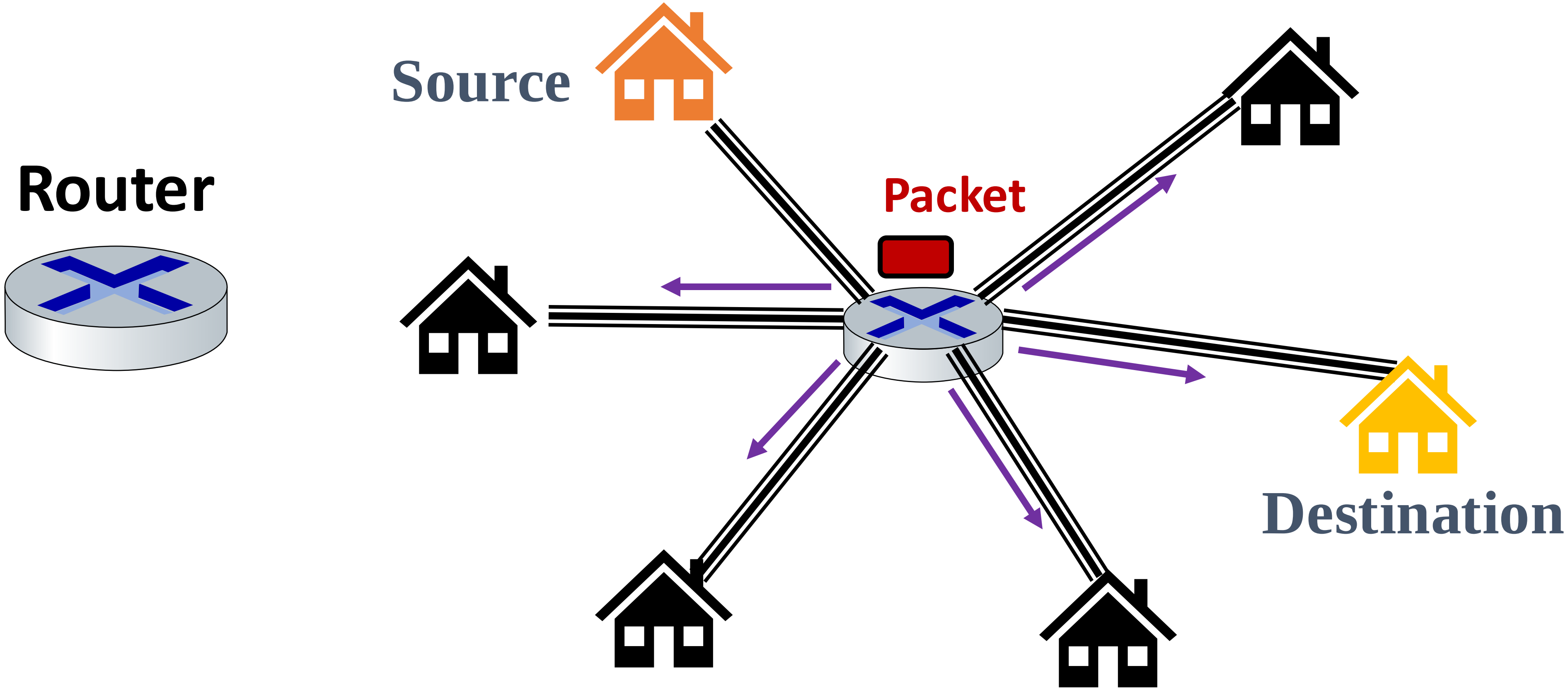


Forwarding:

- aka "switching"
- *local* action: move arriving packets from router's input link to appropriate router output link

Packet Switching

- Forward is also called switching

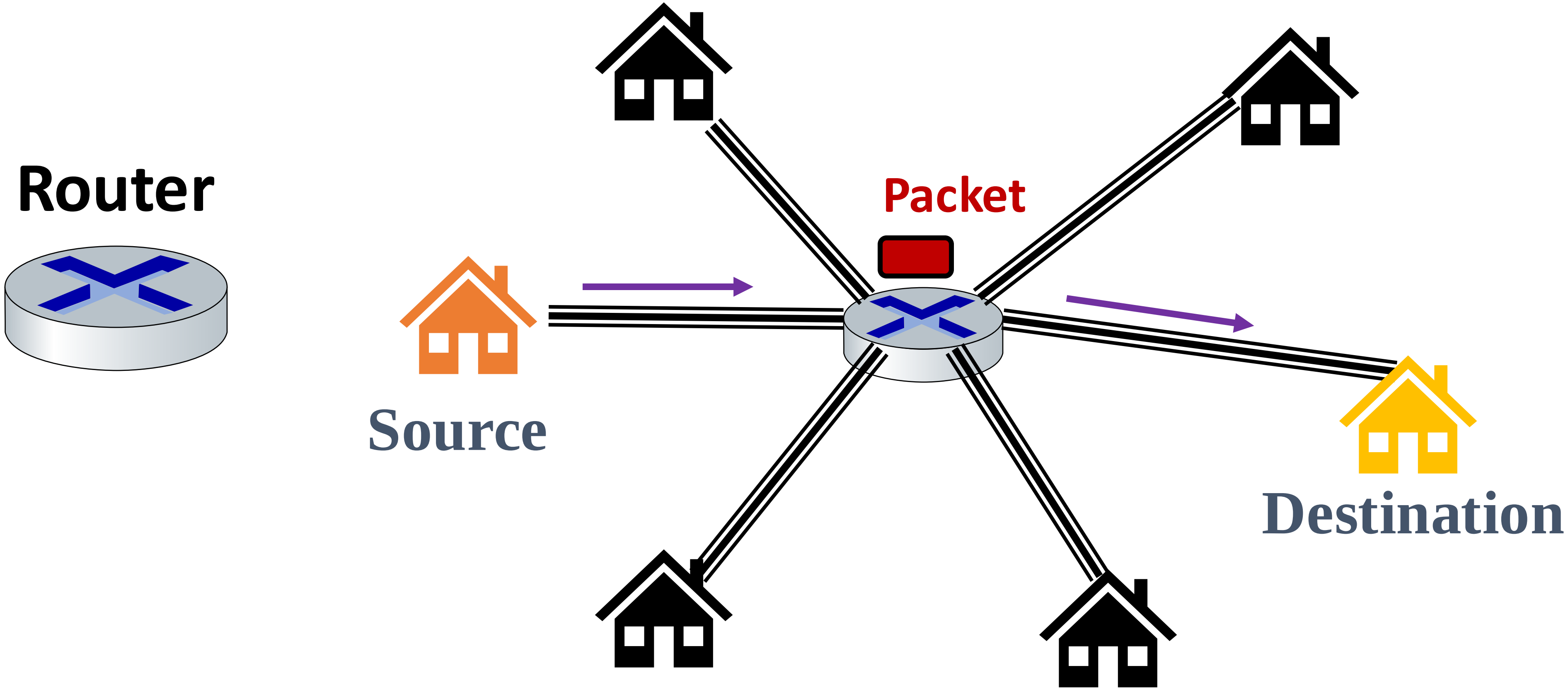


Forwarding:

- aka "switching"
- *local* action: move arriving packets from router's input link to appropriate router output link

Packet Switching

- Forward is also called switching

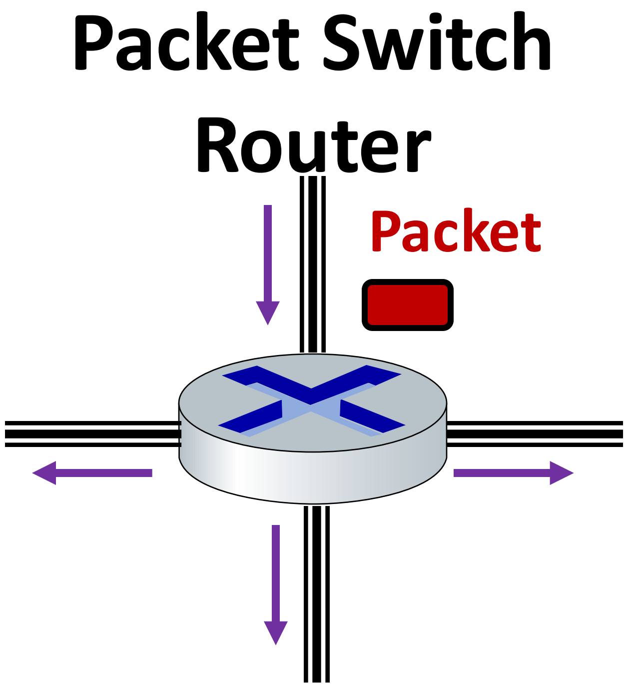


Forwarding:

- aka "switching"
- *local* action: move arriving packets from router's input link to appropriate router output link

Packet-switching: store-and-forward

- Forward is also called switching
  - *store and forward: entire* packet must arrive at router before it can be transmitted on next link

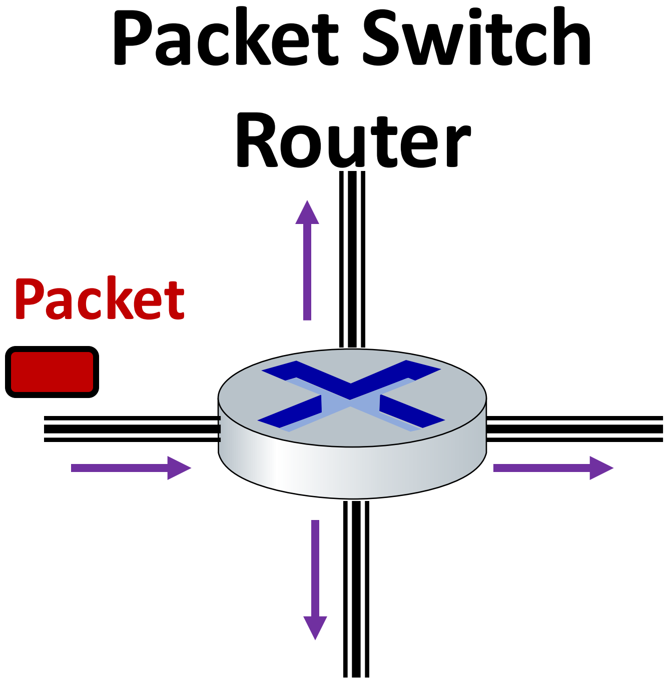


Forwarding:

- aka "switching"
- *local* action: move arriving packets from router's input link to appropriate router output link

Packet-switching: store-and-forward

- Forward is also called switching
  - *store and forward: entire* packet must arrive at router before it can be transmitted on next link

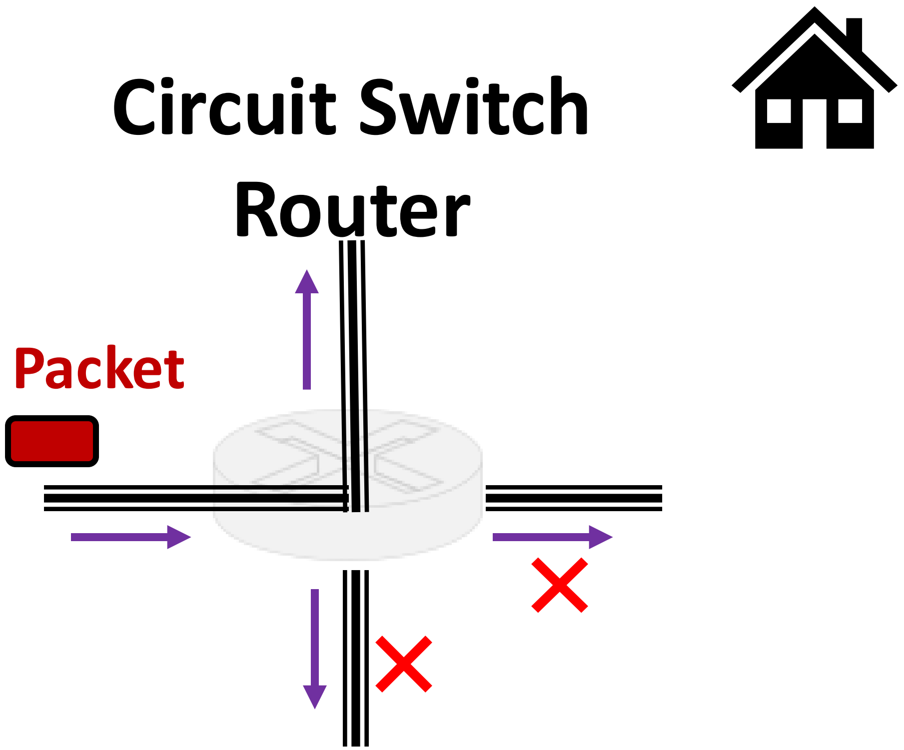


Forwarding:

- aka "switching"
- *local* action: move arriving packets from router's input link to appropriate router output link

Circuit switching

- Forward is also called switching

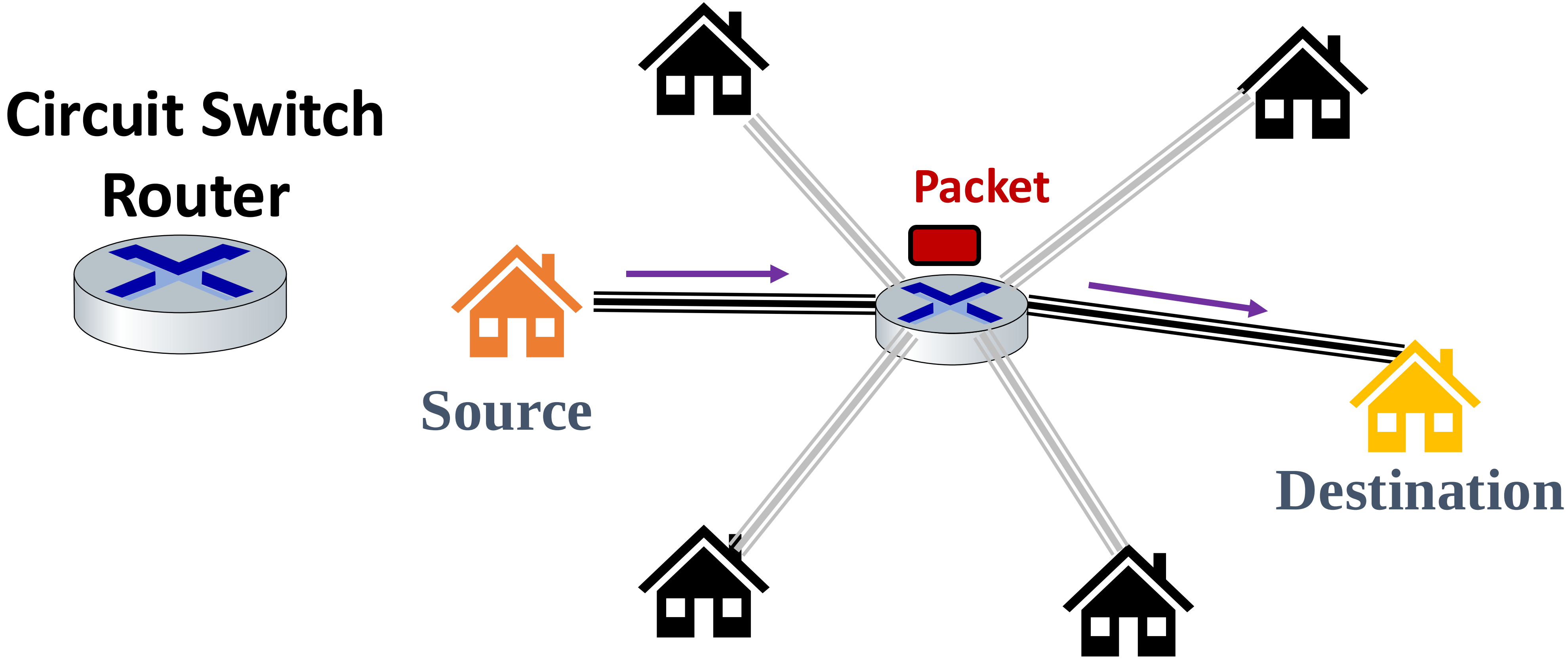


Forwarding:

- aka "switching"
- *local* action: move arriving packets from router's input link to appropriate router output link

Circuit switching

- Forward is also called switching



Forwarding:

- aka "switching"
- *local* action: move arriving packets from router's input link to appropriate router output link

Internet Core: Packet Switching

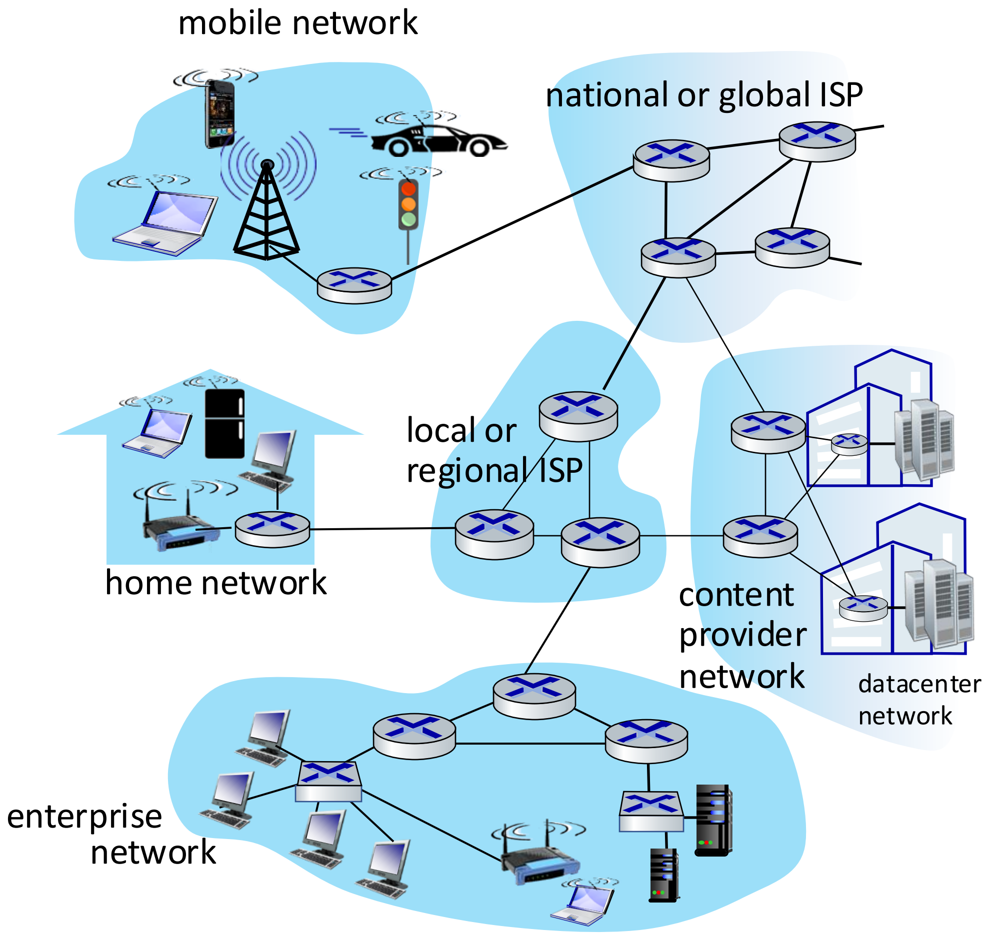
each end-end data stream divided into *packets*

- users A through C packets *share* network resources
- each packet uses full link bandwidth
- resources used *as needed*

- each packet has a "header" (containing e.g., destination address) in addition to "payload" (data)
- Store and Forward (requires buffer and introduces delay)

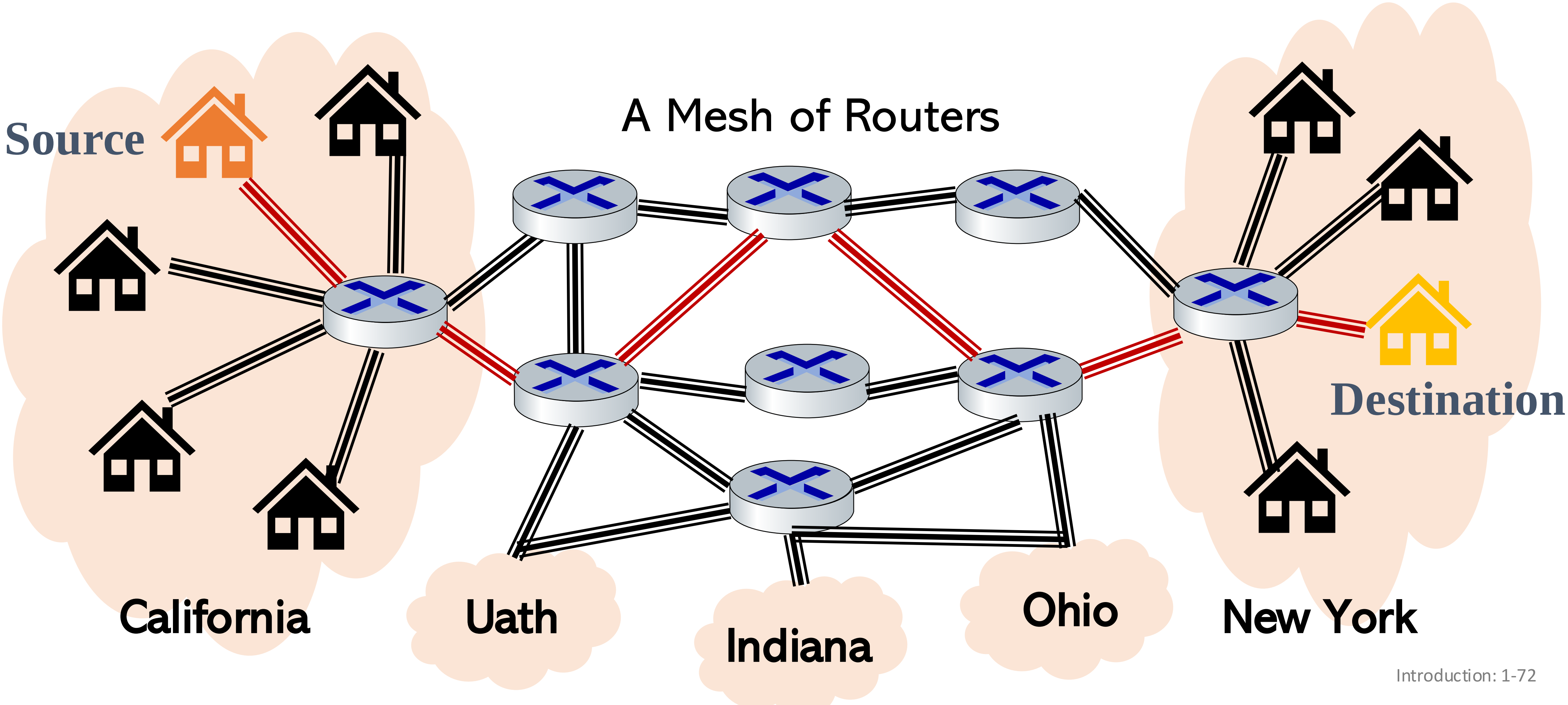
Internet structure: a "network of networks"

- hosts connect to Internet via access Internet Service Providers (ISPs)
- access ISPs in turn must be interconnected
  - so that *any* two hosts *(anywhere!)* can send packets to each other
- resulting network of networks is very complex
  - evolution driven by economics, national policies

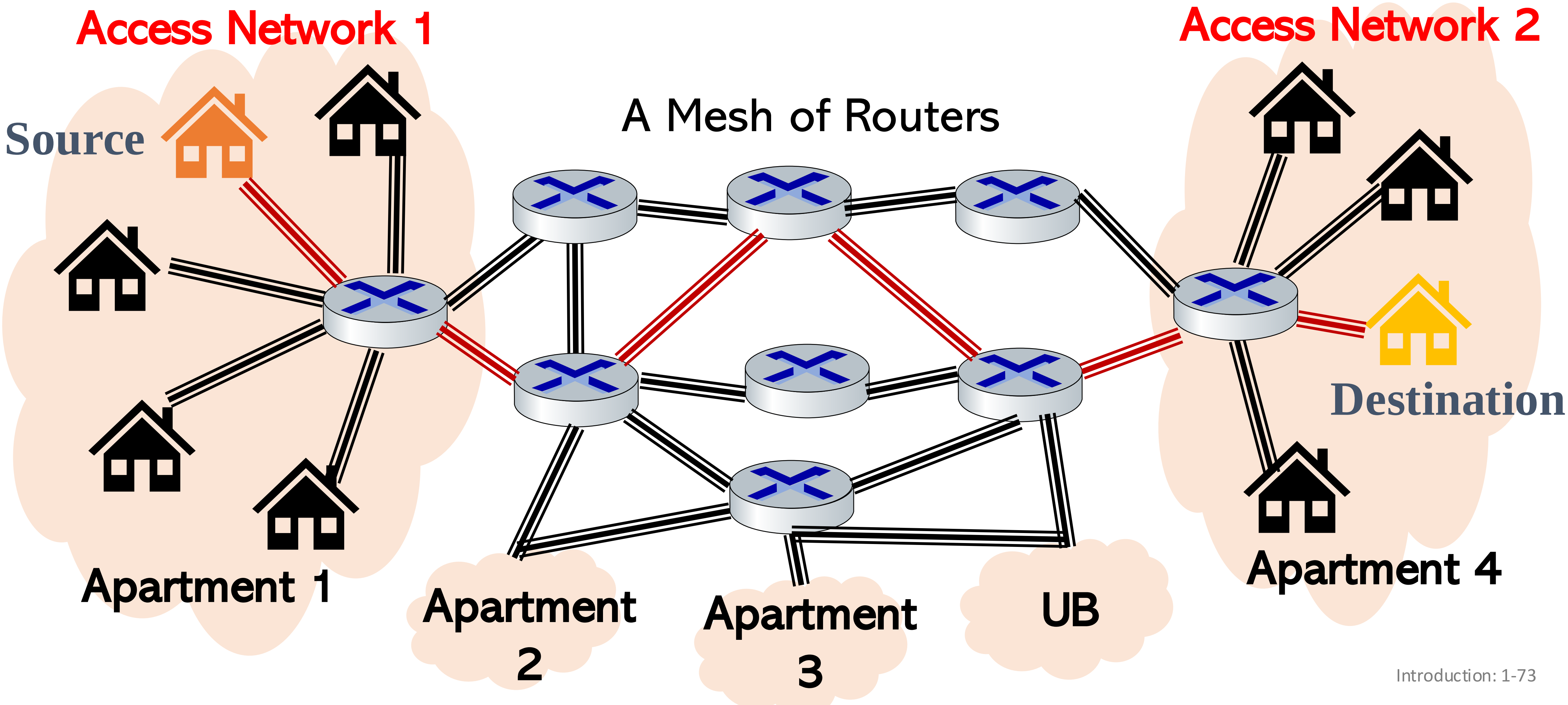


Let's take a stepwise approach to describe current Internet structure

Internet structure: a "network of networks"

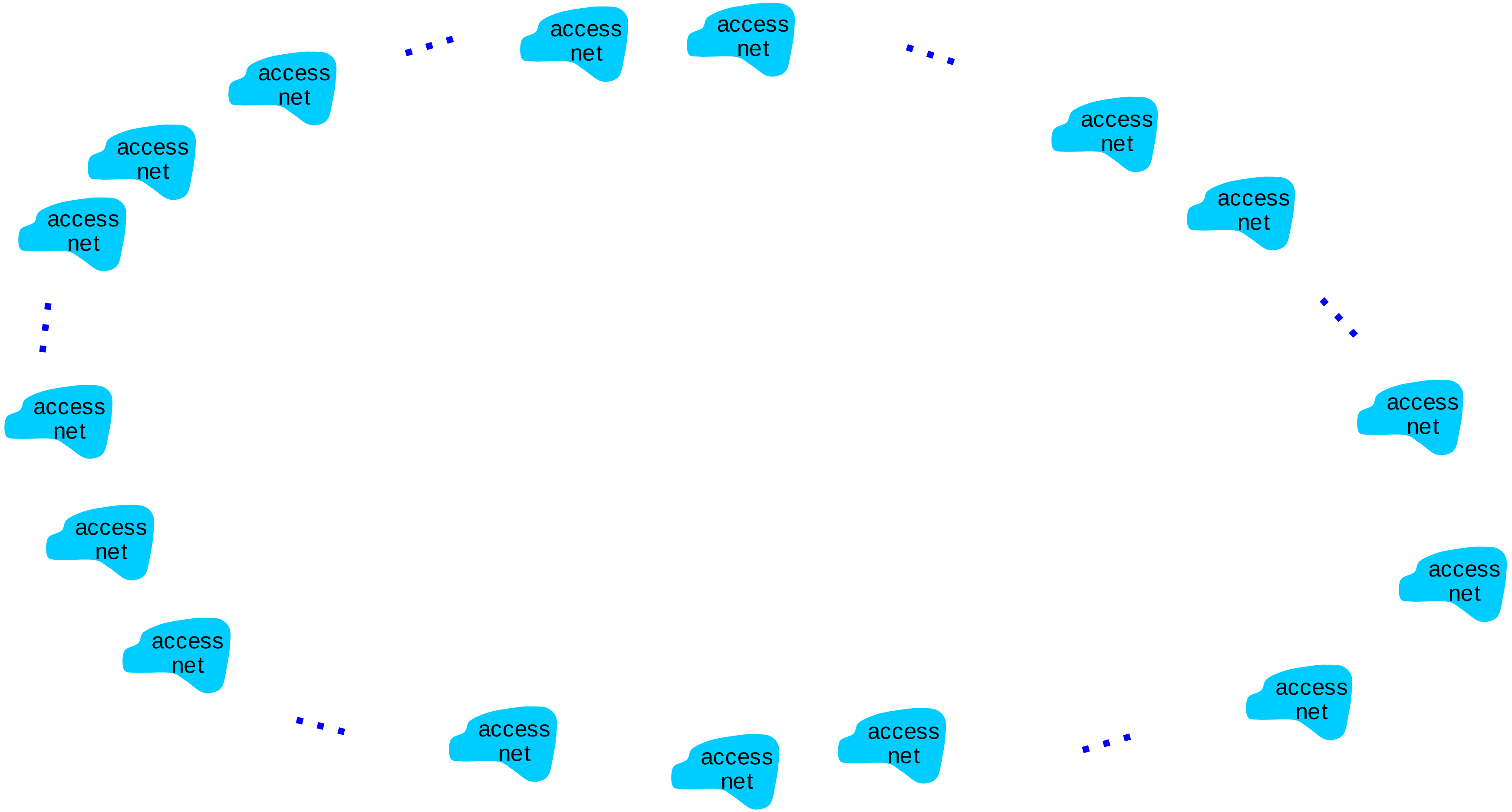


Internet structure: a "network of networks"



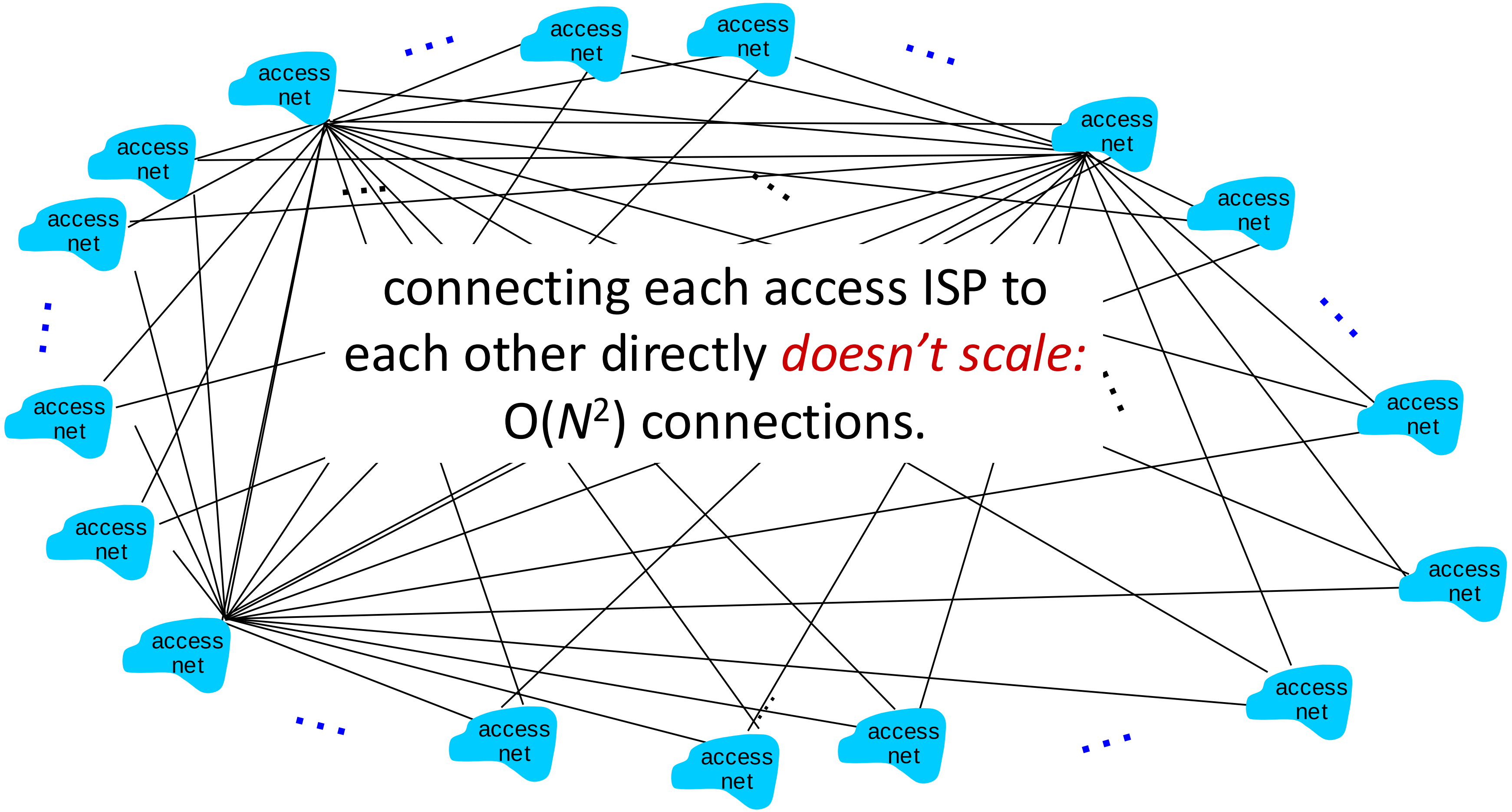
Internet structure: a "network of networks"

Question: given *millions* of access ISPs, how to connect them together?



Internet structure: a "network of networks"

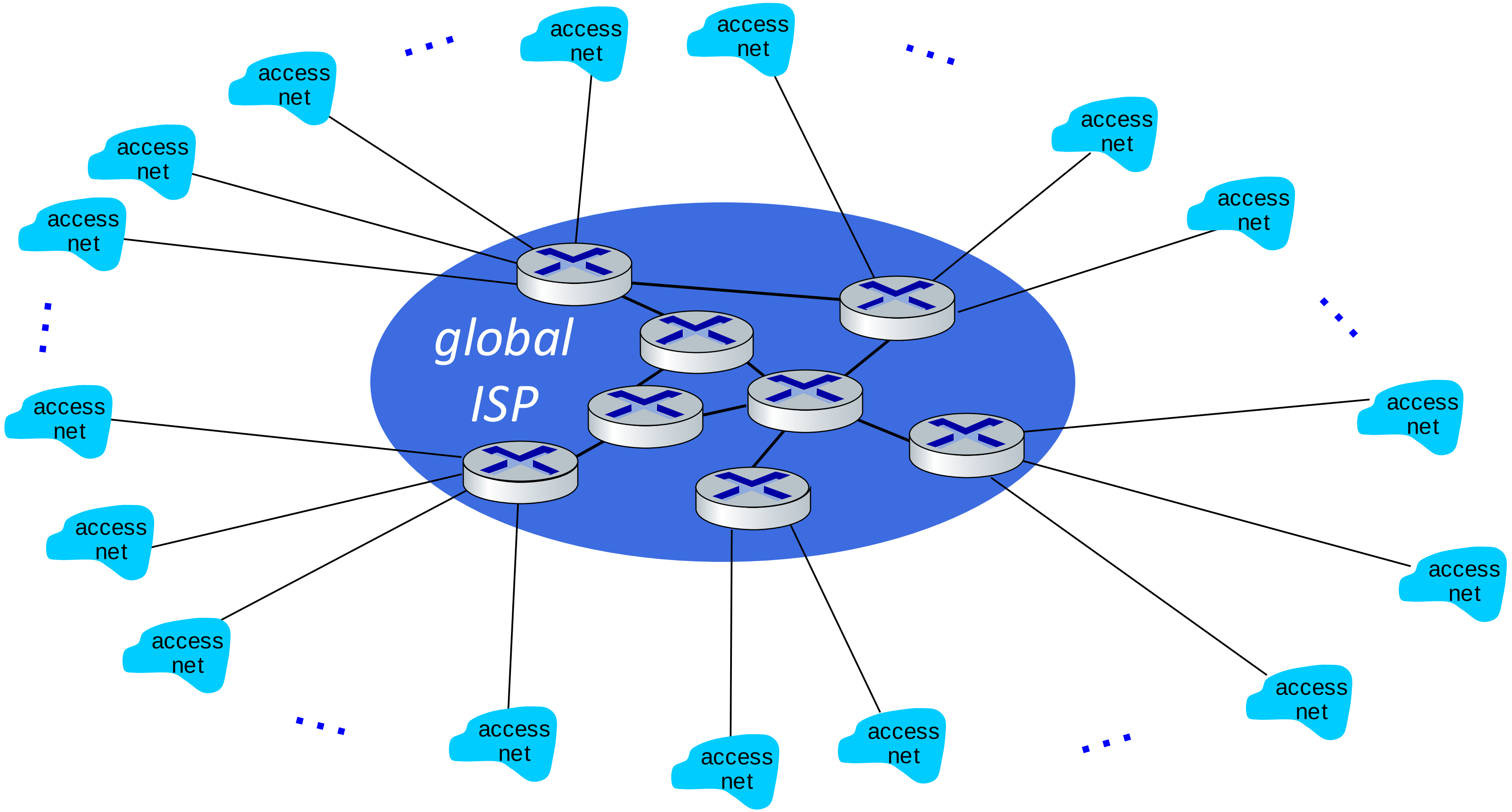
Question: given *millions* of access ISPs, how to connect them together?



connecting each access ISP to each other directly *doesn't scale:* O(N^2) connections.

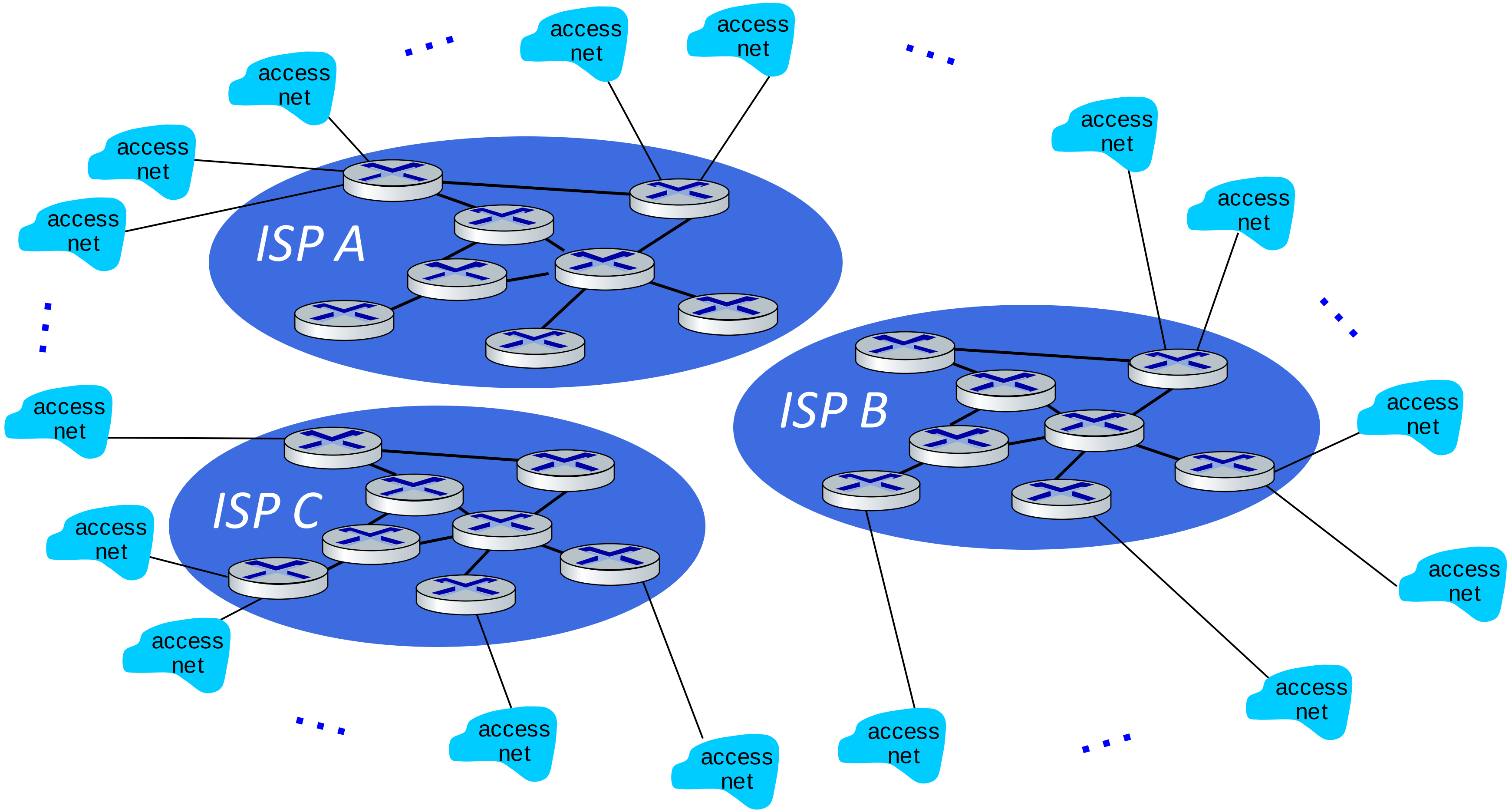
Internet structure: a "network of networks"

Option: connect each access ISP to one global transit ISP?
Customer and provider ISPs have economic agreement.



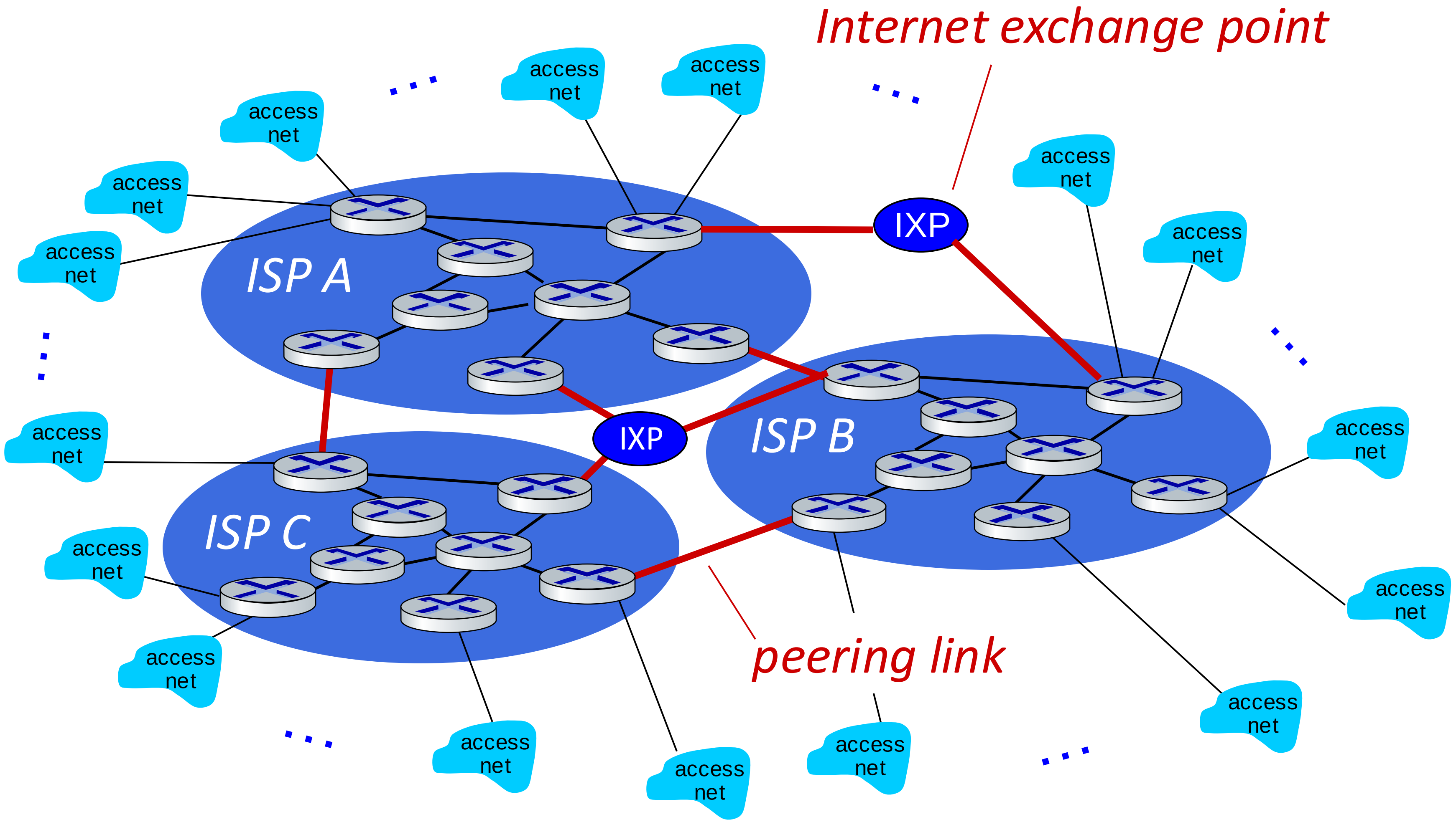
Internet structure: a "network of networks"

But if one global ISP is viable business, there will be competitors ….



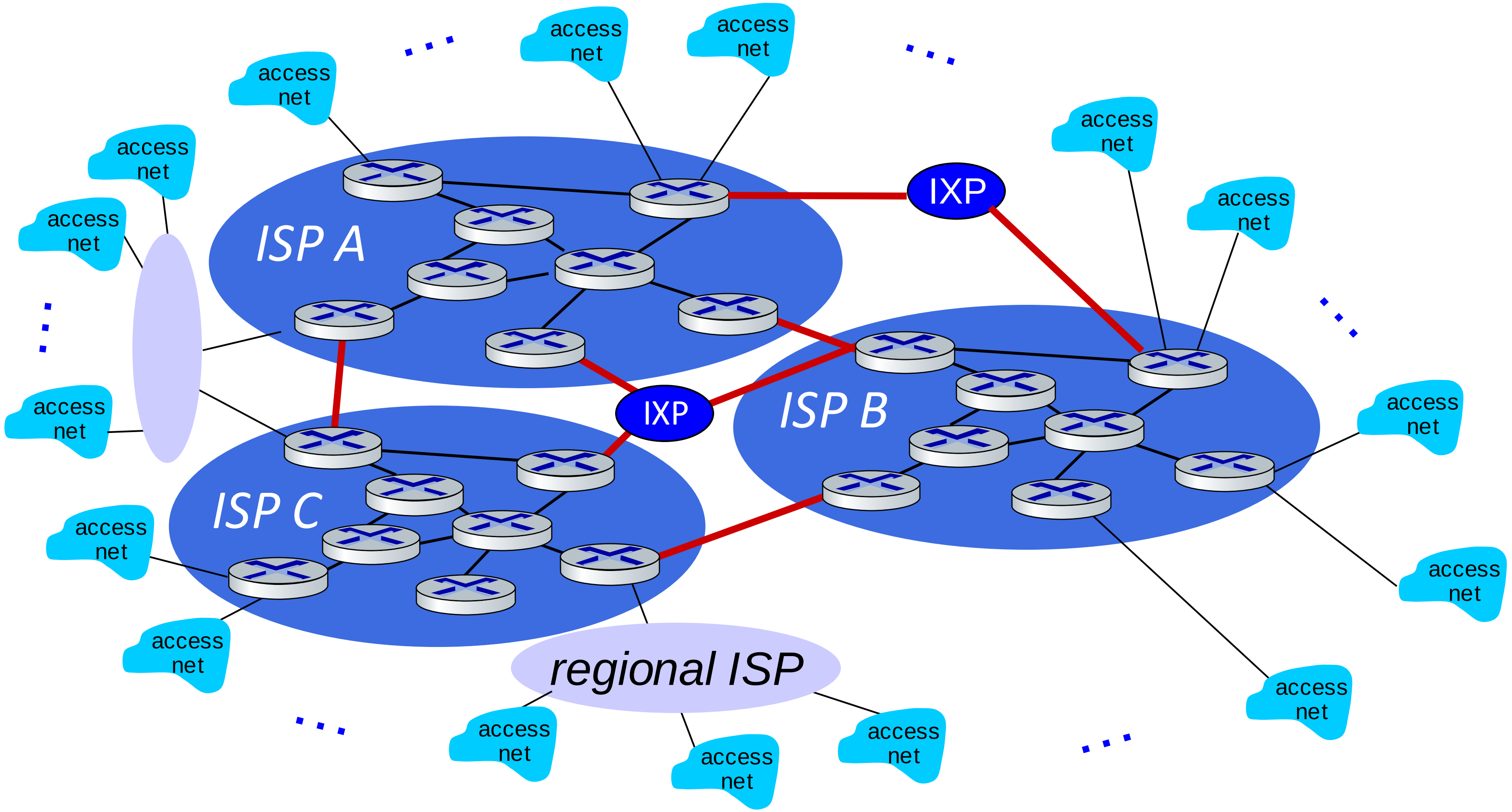
Internet structure: a "network of networks"

But if one global ISP is viable business, there will be competitors …. who will want to be connected



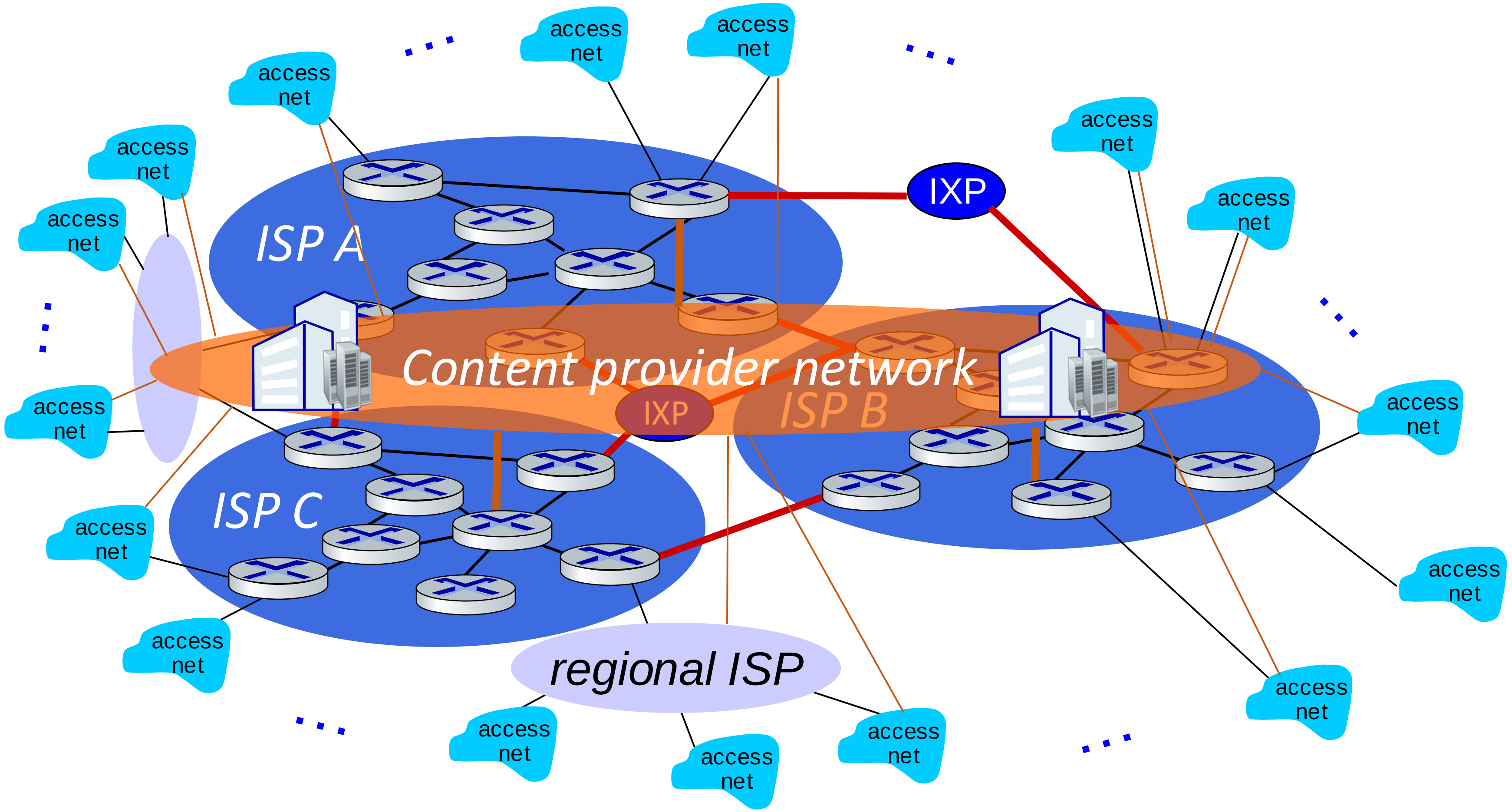
Internet structure: a "network of networks"

… and regional networks may arise to connect access nets to ISPs

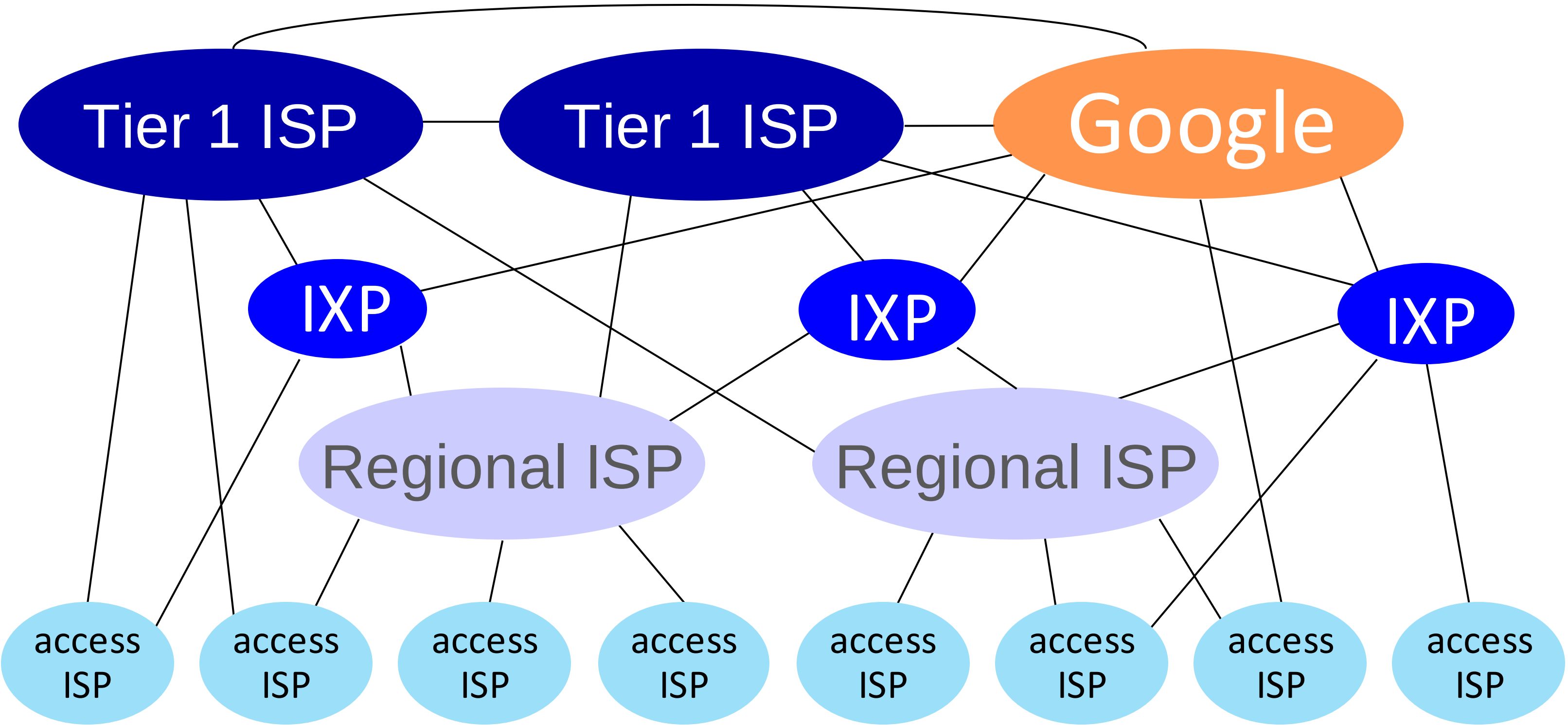


Internet structure: a "network of networks"

… and content provider networks (e.g., Google, Microsoft, Akamai) may run their own network, to bring services, content close to end users



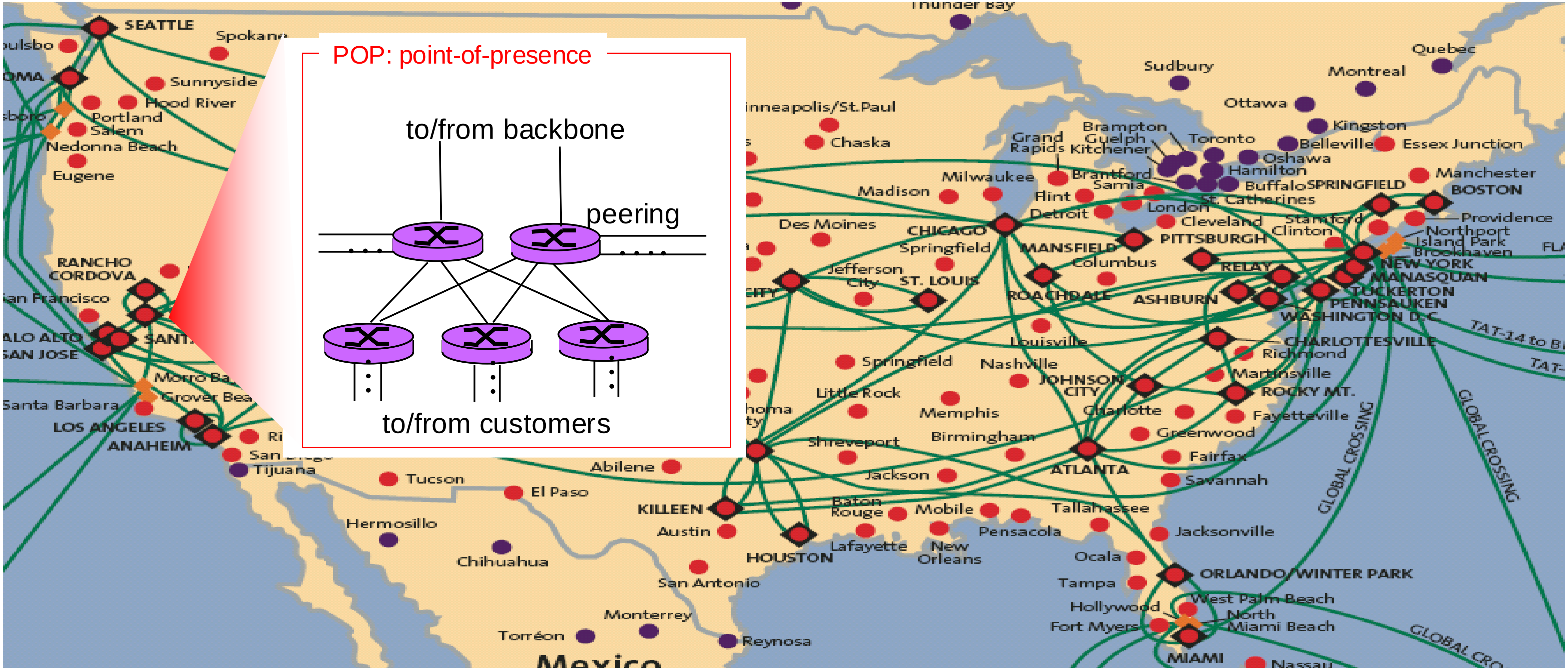
Internet structure: a "network of networks"



At "center": small # of large but well-connected networks

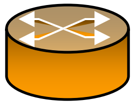
- "tier-1" commercial ISPs (e.g., Level 3, Sprint, AT&T, NTT), national & international coverage
- content provider networks (e.g., Google, Facebook): private network that connects its data centers to Internet, often bypassing tier-1, regional ISPs

Tier-1 ISP: e.g., Sprint/T-Mobile



POPs from different Tier-1 ISP connect to each other at IXPs – residing at a building like this in London

Internet Core Routers (including those at POPs/IXP)

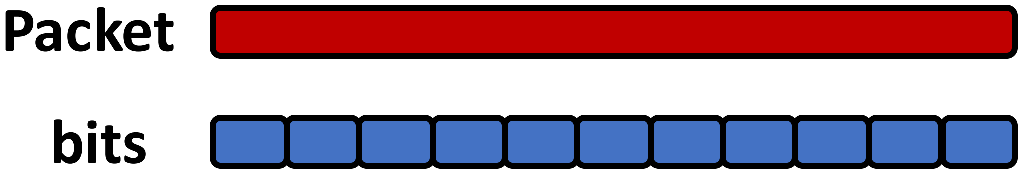


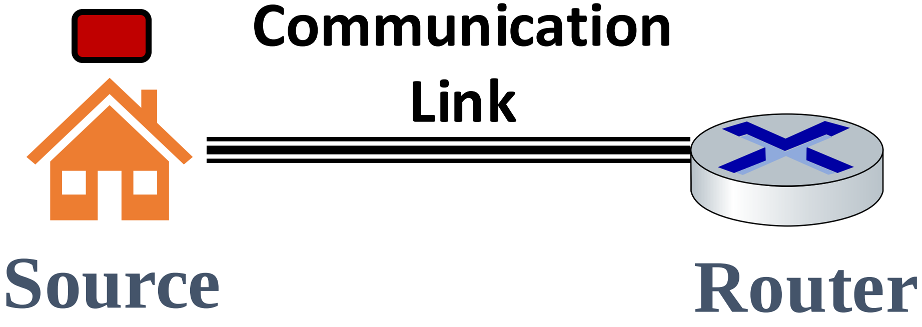
Router on "paper"

Chapter 1: roadmap

- What *is* the Internet?
- What *is* a protocol?
- Network edge: hosts, access network, physical media
- Network core: internet structure, routing and forwarding
- **Performance: loss, delay, throughput**
- Security
- Protocol layers, service models
- History

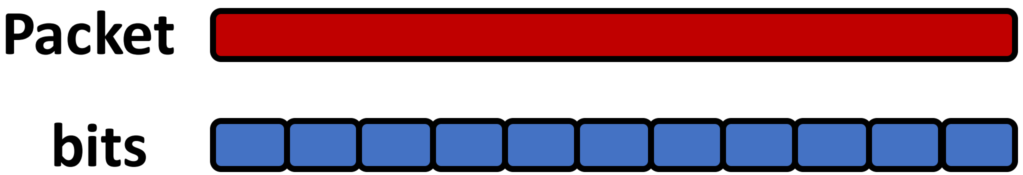
How to send a packet via network

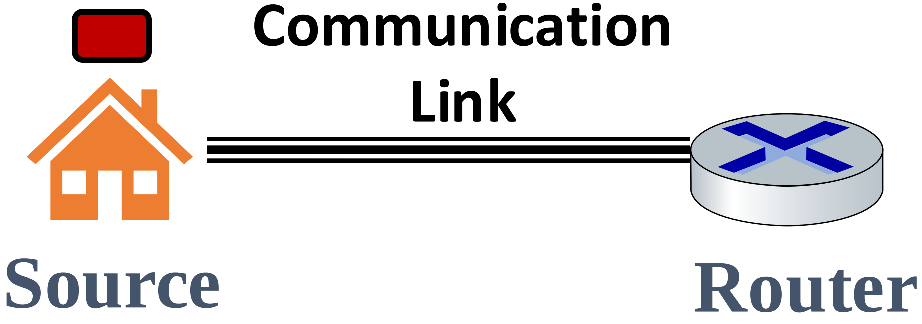




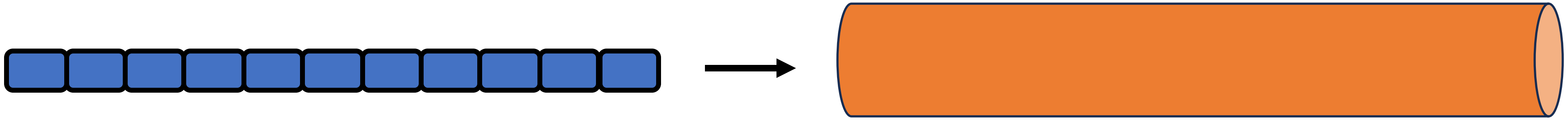
Step 1: Transmit the packets into the link

How to send a packet via network



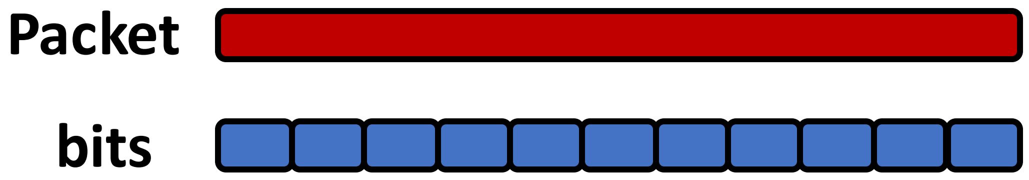


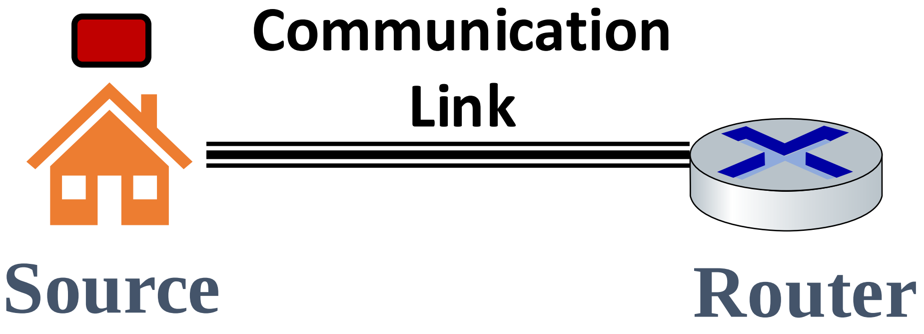
Step 1: Transmit the packets into the link



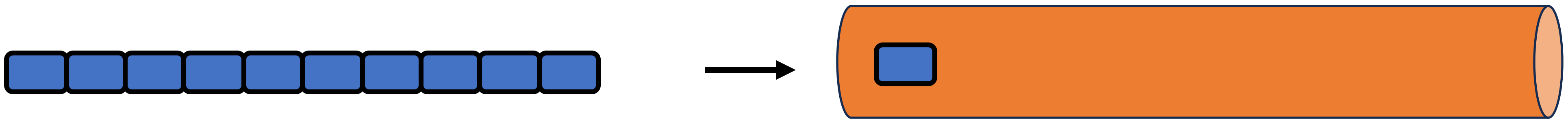
Link capacity: 10 bit/sec

How to send a packet via network

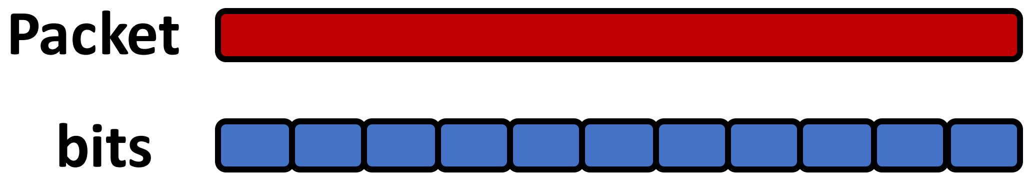


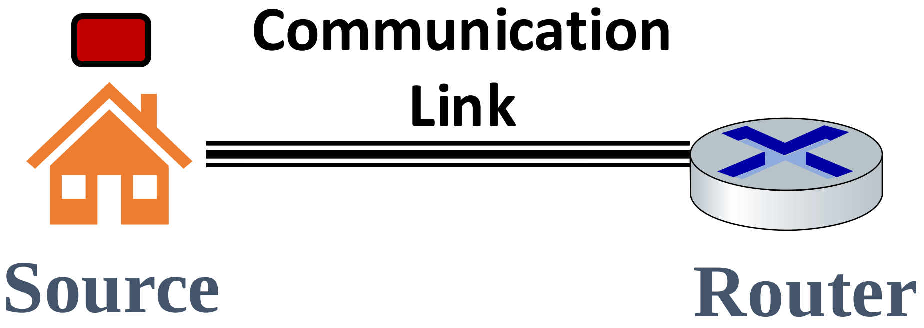


Step 1: Transmit the packets into the link

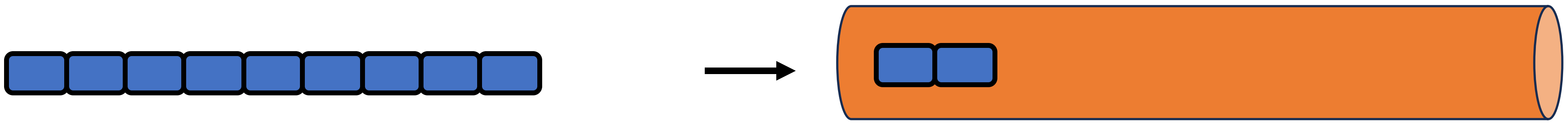


How to send a packet via network



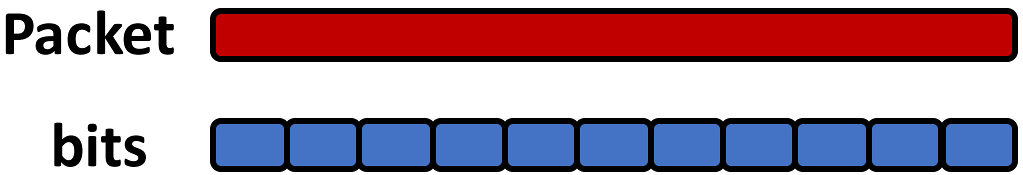


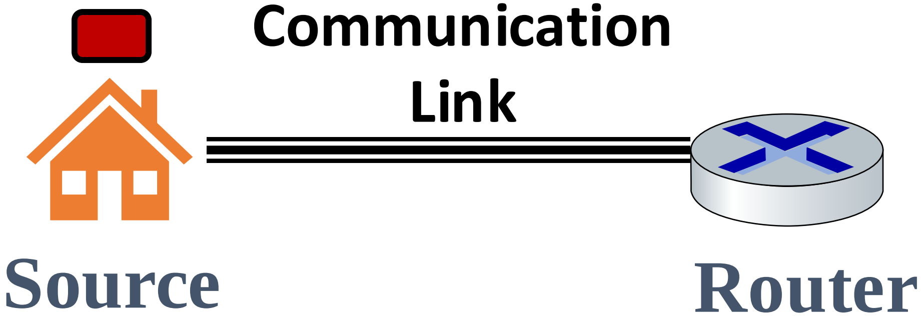
Step 1: Transmit the packets into the link



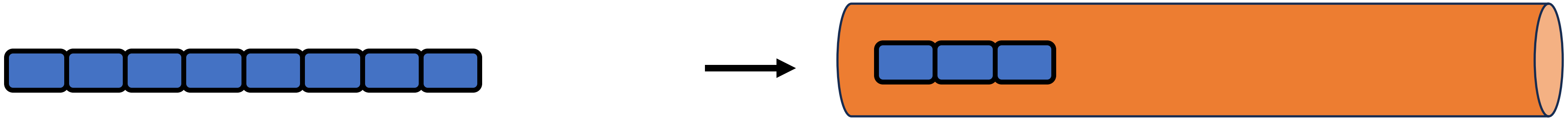
Link capacity: 10 bit/sec

How to send a packet via network



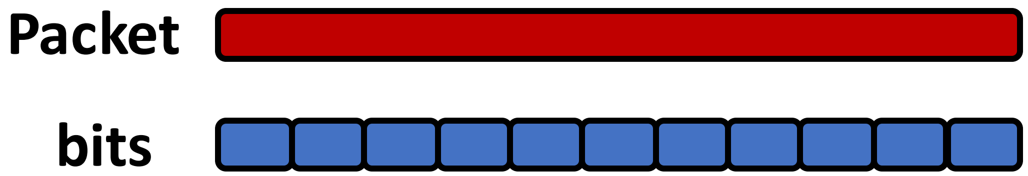


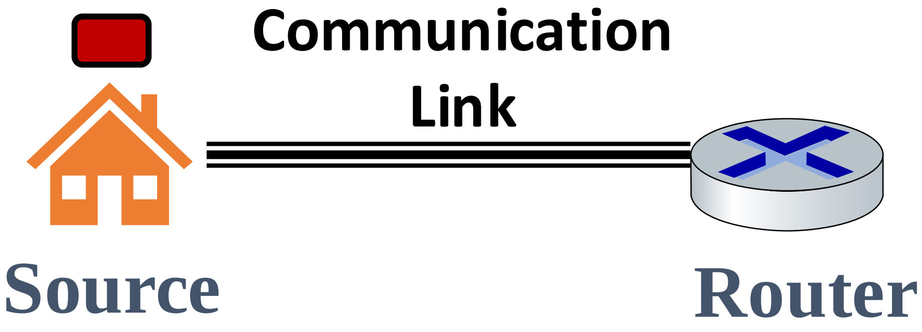
Step 1: Transmit the packets into the link



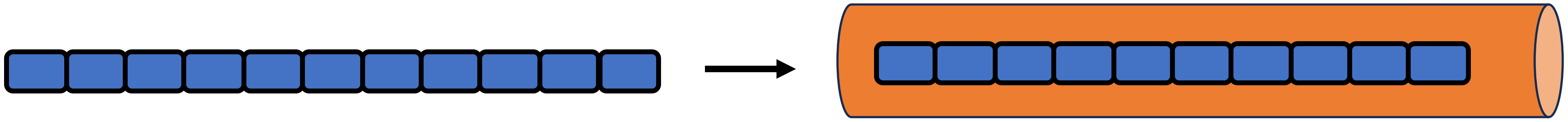
Link capacity: 10 bit/sec

How to send a packet via network



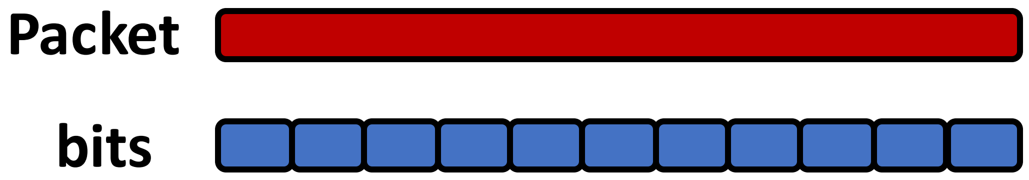


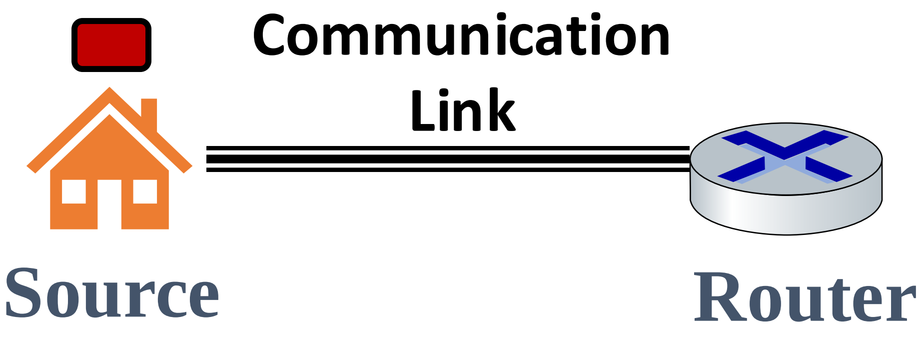
Step 1: Transmit the packets into the link



Link capacity: 10 bit/sec

How to send a packet via network

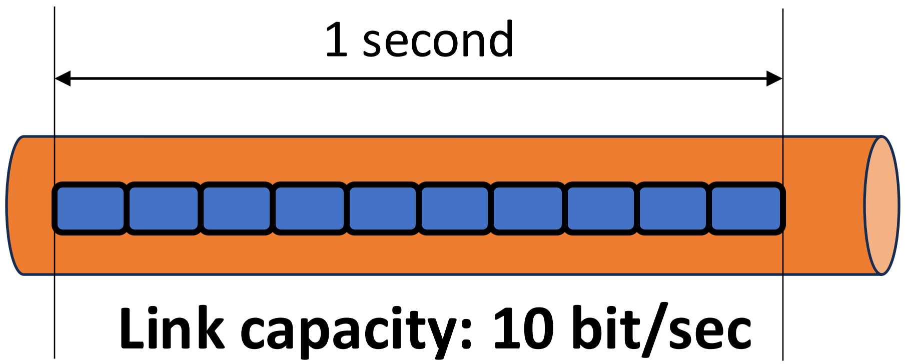




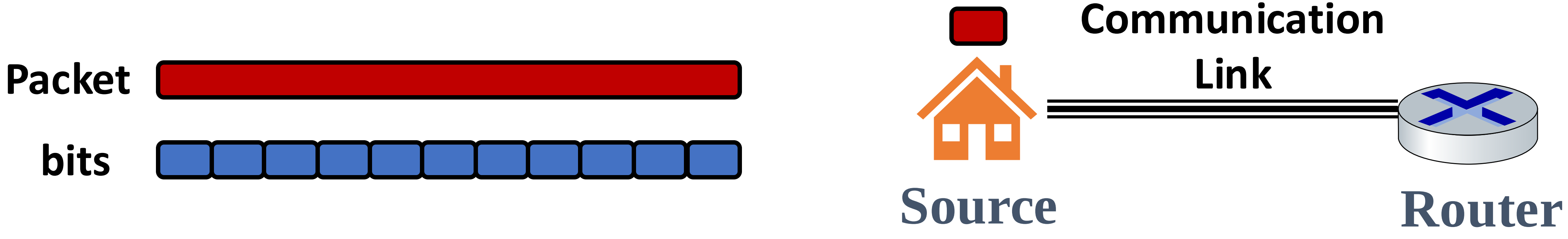
Step 1: Transmit the packets into the link

d_{trans}: transmission delay:

- L: packet length (bits)
- R: link *transmission rate (bps)*
- $d_{trans} = L/R$



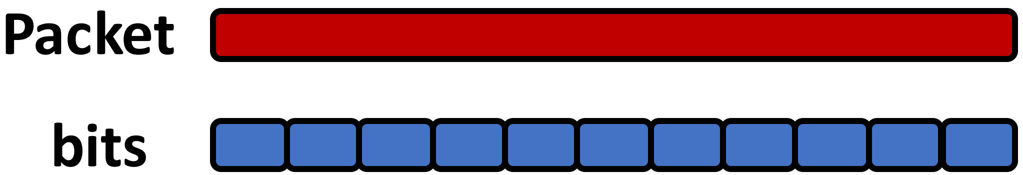
How to send a packet via network

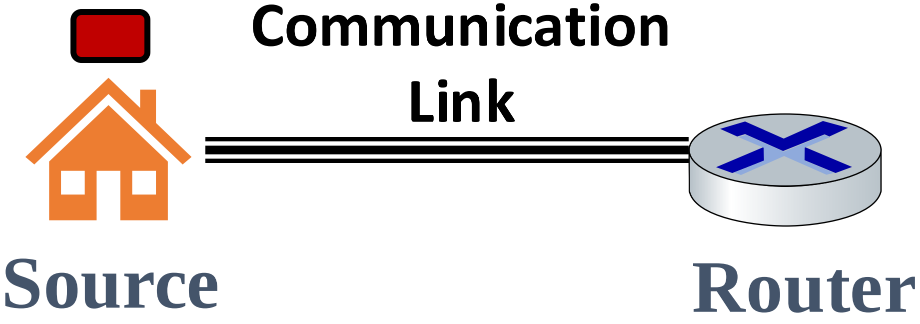


Step 1: Transmit the packets into the link

Step 2: The packet bits propagates to the router

How to send a packet via network



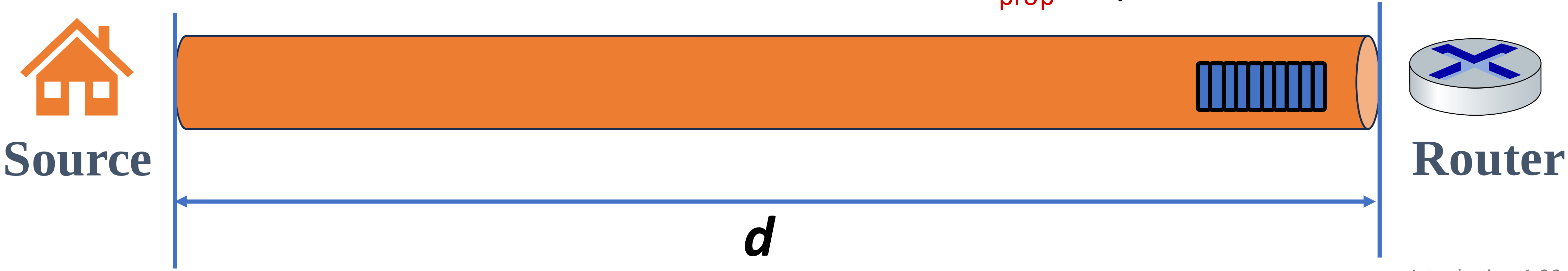


Step 1: Transmit the packets into the link

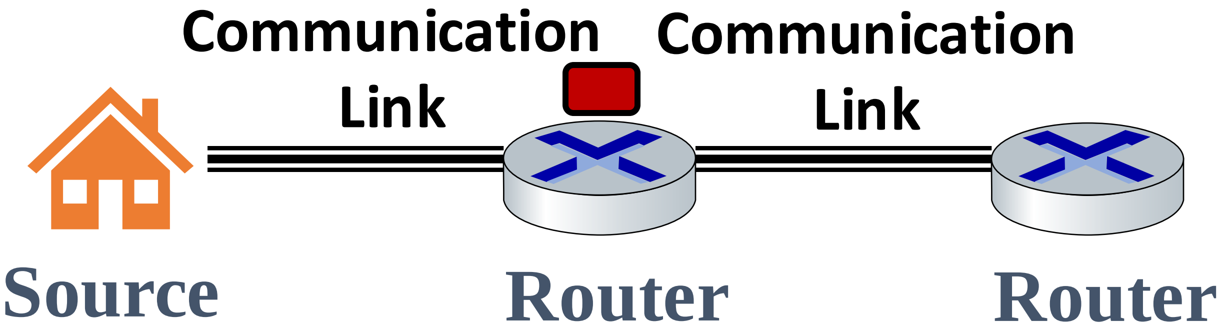
Step 2: The packet bits propagates to the router

d_{prop}: propagation delay:

- d: length of physical link
- s: propagation speed (~2×10^8 m/sec
- $d_{prop} = d/s$



How to send a packet via network

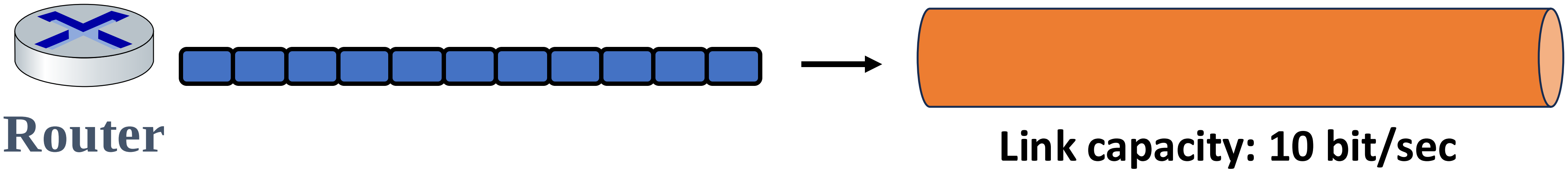


Step 1: Transmit the packets into the link

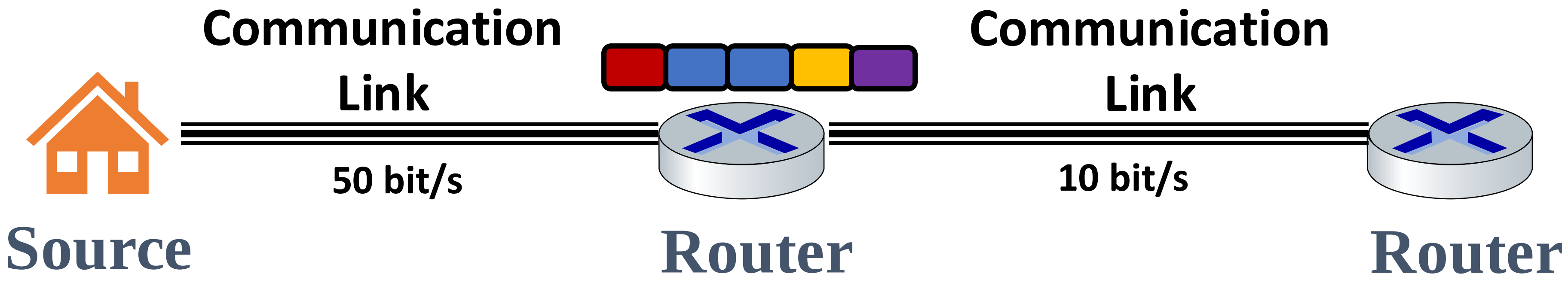
Step 2: The packet bits propagates to the router

d_{trans}: transmission delay:

- *L*: packet length (bits)
- *R*: link *transmission rate (bps)*
- $d_{trans} = L/R$



How to send a packet via network



Key point:

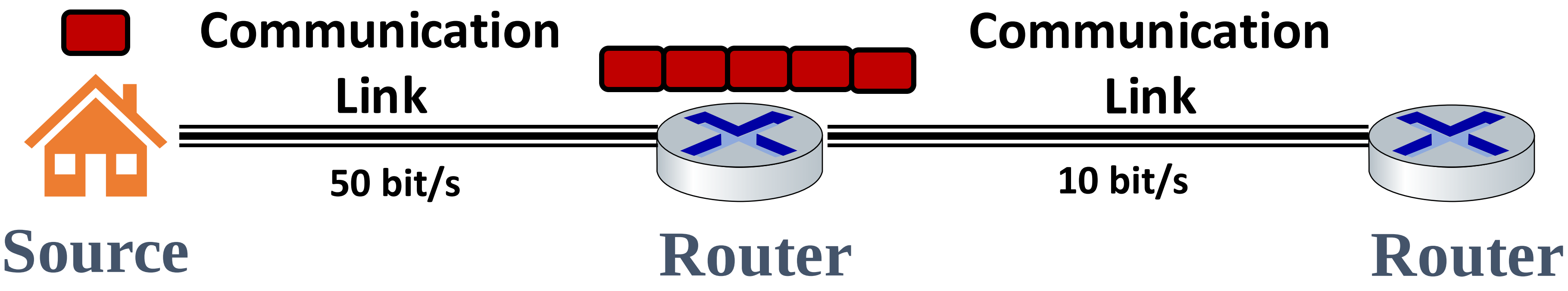
- Router takes transmission delay to transmit a packet to the link
- The packet may arrive faster than the packets get out of the router
- The later arrived packets must wait at the router until all the packets arriving before it are transmitted into the link

d_{queue}: queueing delay

- time waiting at output link for transmission
- depends on congestion level of router

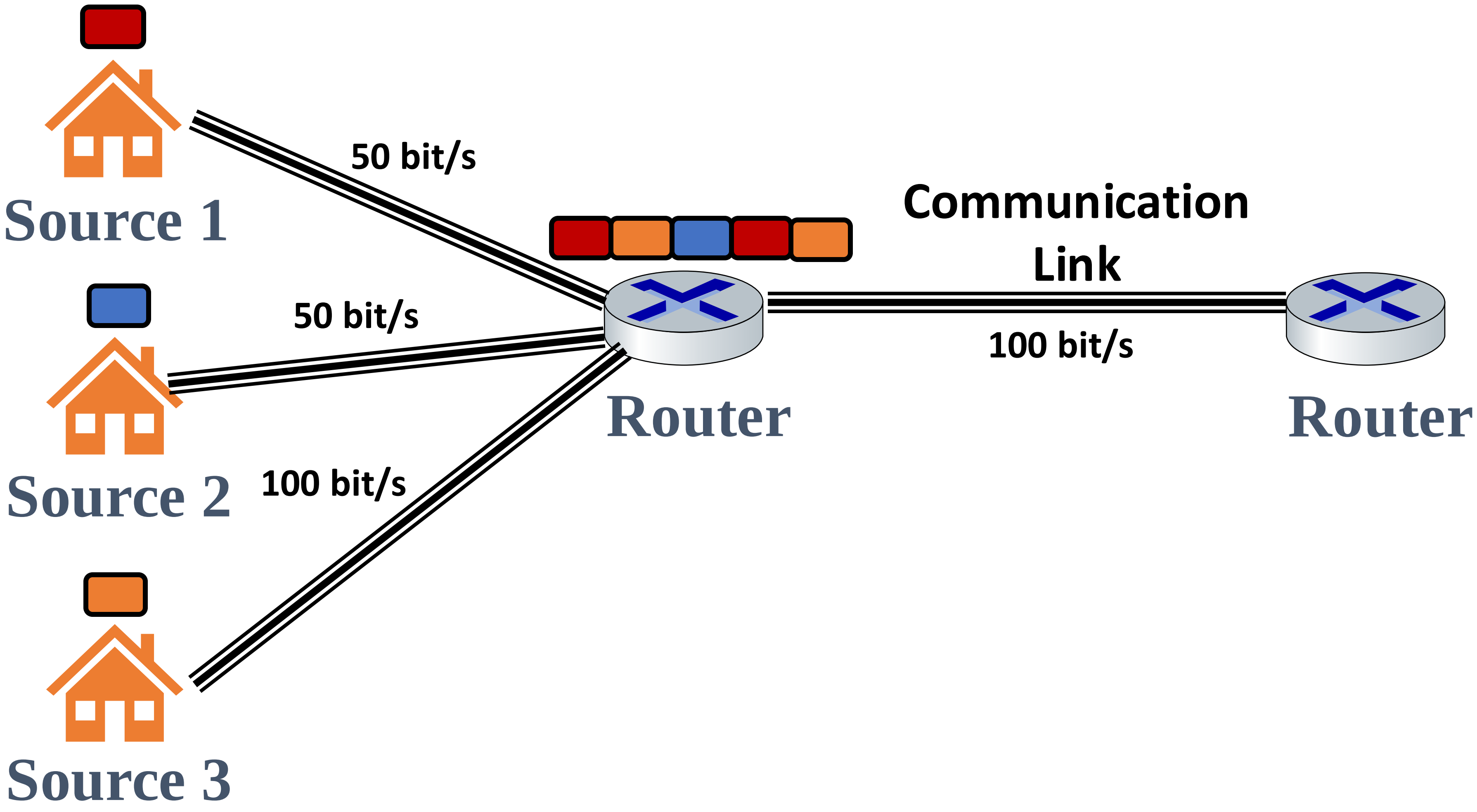
How to send a packet via network

Various reasons of queuing inside the router



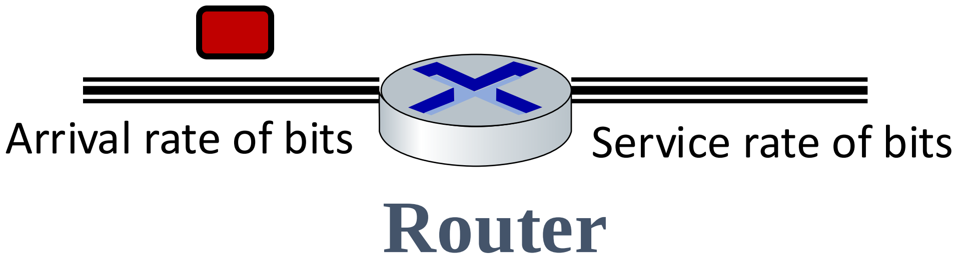
How to send a packet via network

Various reasons of queuing inside the router



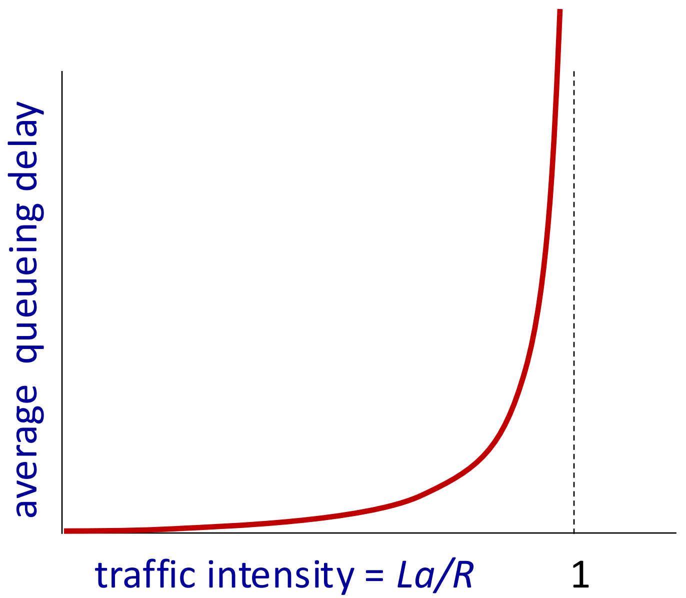
Packet queueing delay (revisited)

- *a:* average packet arrival rate
- *L:* packet length (bits)
- *R:* link bandwidth (bit transmission rate)



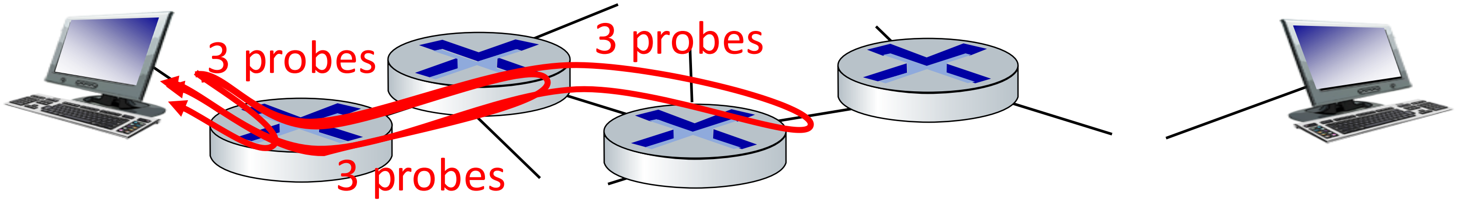
$$\frac{L \cdot a}{R} : \frac{\text{arrival rate of bits}}{\text{service rate of bits}}$$ *"traffic intensity"*

- *La/R* ~ 0: avg. queueing delay small
- *La/R* -> 1: avg. queueing delay large
- *La/R* > 1: more "work" arriving is more than can be serviced - average delay infinite!



“Real” Internet delays and routes

- what do “real” Internet delay & loss look like?
- **traceroute** program: provides delay measurement from source to router along end-end Internet path towards destination. For all *i:*
  - sends three packets that will reach router *i* on path towards destination (with time-to-live field value of *i*)
  - router *i* will return packets to sender
  - sender measures time interval between transmission and reply



Real Internet delays and routes

traceroute: gaia.cs.umass.edu to www.eurecom.fr

3 delay measurements from gaia.cs.umass.edu to cs-gw.cs.umass.edu

3 delay measurements to border1-rt-fa5-1-0.gw.umass.edu

```
1  cs-gw (128.119.240.254)  1 ms  1 ms  2 ms
2  border1-rt-fa5-1-0.gw.umass.edu (128.119.3.145)  1 ms  1 ms  2 ms
3  cht-vbns.gw.umass.edu (128.119.3.130)  6 ms 5 ms 5 ms
4  jn1-at1-0-0-19.wor.vbns.net (204.147.132.129)  16 ms 11 ms 13 ms
5  jn1-so7-0-0-0.wae.vbns.net (204.147.136.136)  21 ms 18 ms 18 ms
6  abilene-vbns.abilene.ucaid.edu (198.32.11.9)  22 ms  18 ms  22 ms
7  nycm-wash.abilene.ucaid.edu (198.32.8.46)  22 ms  22 ms  22 ms
8  62.40.103.253 (62.40.103.253)  104 ms 109 ms 106 ms
9  de2-1.de1.de.geant.net (62.40.96.129)  109 ms 102 ms 104 ms
10  de.fr1.fr.geant.net (62.40.96.50)  113 ms 121 ms 114 ms
11  renater-gw.fr1.fr.geant.net (62.40.103.54)  112 ms  114 ms  112 ms
12  nio-n2.cssi.renater.fr (193.51.206.13)  111 ms  114 ms  116 ms
13  nice.cssi.renater.fr (195.220.98.102)  123 ms  125 ms  124 ms
14  r3t2-nice.cssi.renater.fr (195.220.98.110)  126 ms  126 ms  124 ms
15  eurecom-valbonne.r3t2.ft.net (193.48.50.54)  135 ms  128 ms  133 ms
16  194.214.211.25 (194.214.211.25)  126 ms  128 ms  126 ms
17  * * *
18  * * *
19  fantasia.eurecom.fr (193.55.113.142)  132 ms  128 ms  136 ms
```

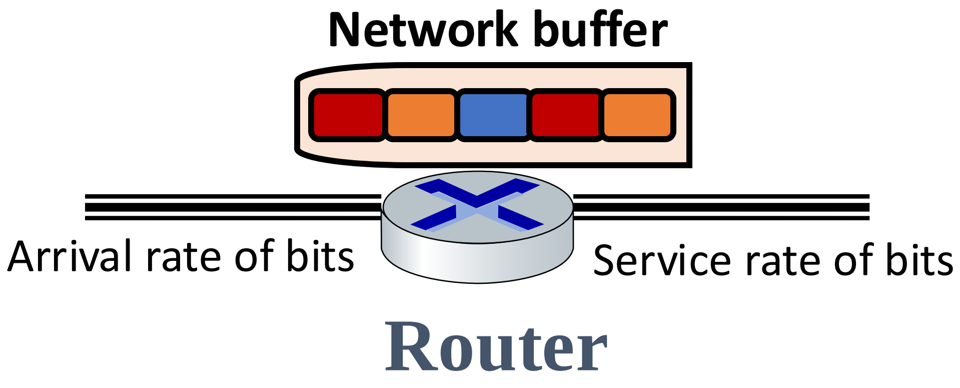
trans-oceanic link (between hop 7 and hop 8)

* means no response (probe lost, router not replying)

* Do some traceroutes from exotic countries at www.traceroute.org

Packet loss

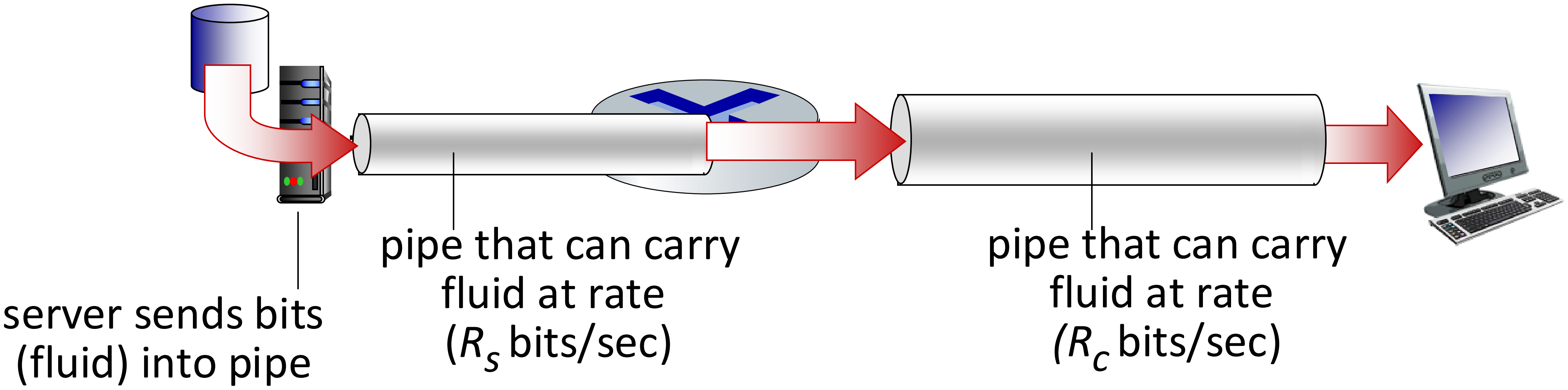
- queue (aka buffer) preceding link in buffer has finite capacity
- packet arriving at a full queue dropped (aka lost)
- lost packet may be retransmitted by previous node, by source end system, or not at all



* Check out the Java applet for an interactive animation (on publisher's website) of queuing and loss

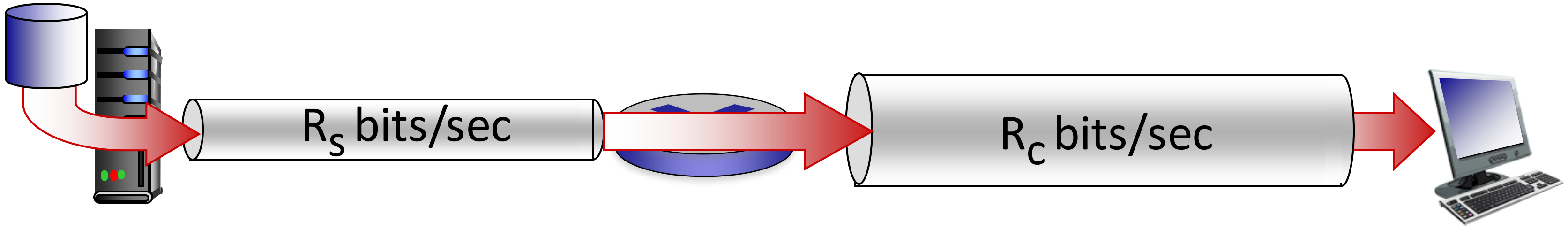
Throughput

- *throughput:* rate (bits/time unit) at which bits are being sent from sender to receiver
  - *instantaneous:* rate at a given point in time
  - *average:* rate over longer period of time

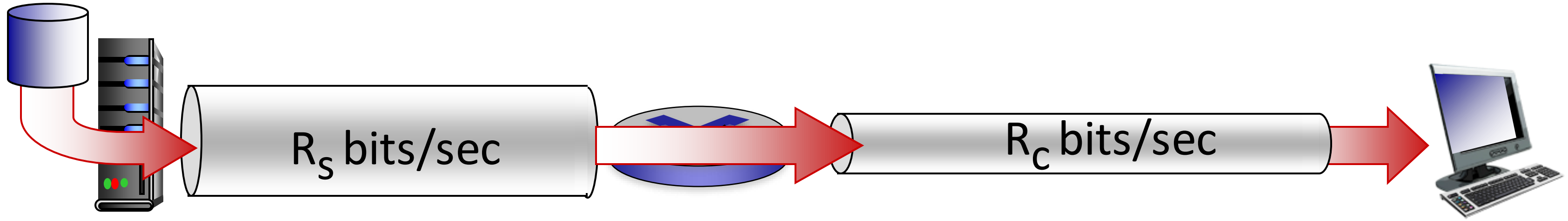


Throughput

$R_s < R_c$ What is average end-end throughput?



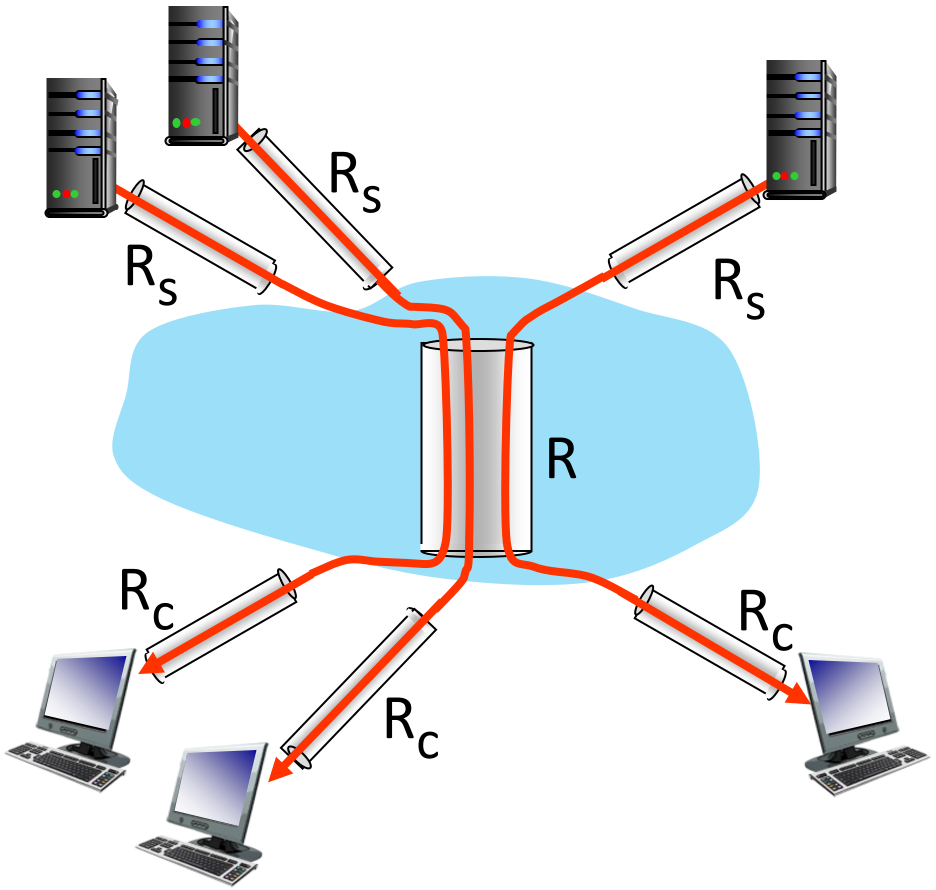
$R_s > R_c$ What is average end-end throughput?



bottleneck link

link on end-end path that constrains end-end throughput

Throughput: network scenario



10 connections (fairly) share backbone bottleneck link *R* bits/sec

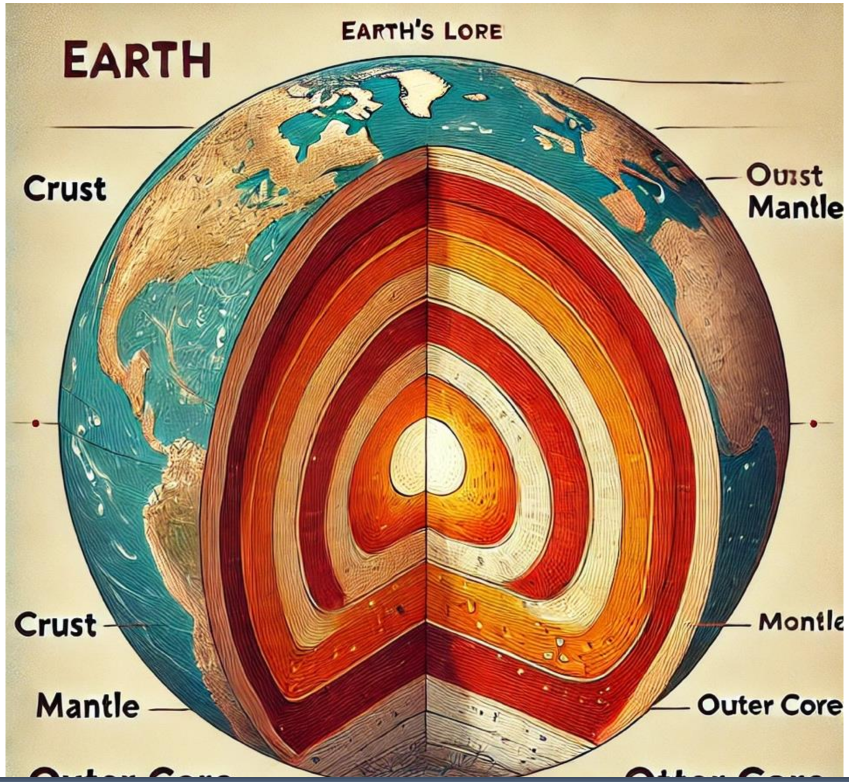
- per-connection end-end throughput: $min(R_c,R_s,R/10)$
- in practice: R_c or R_s is often bottleneck

* Check out the online interactive exercises for more examples: http://gaia.cs.umass.edu/kurose_ross/

Chapter 1: roadmap

- What *is* the Internet?
- What *is* a protocol?
- Network edge: hosts, access network, physical media
- Network core: packet/circuit switching, internet structure
- Performance: loss, delay, throughput
- Protocol layers, service models
- Security
- History

Layers in Computer Networks

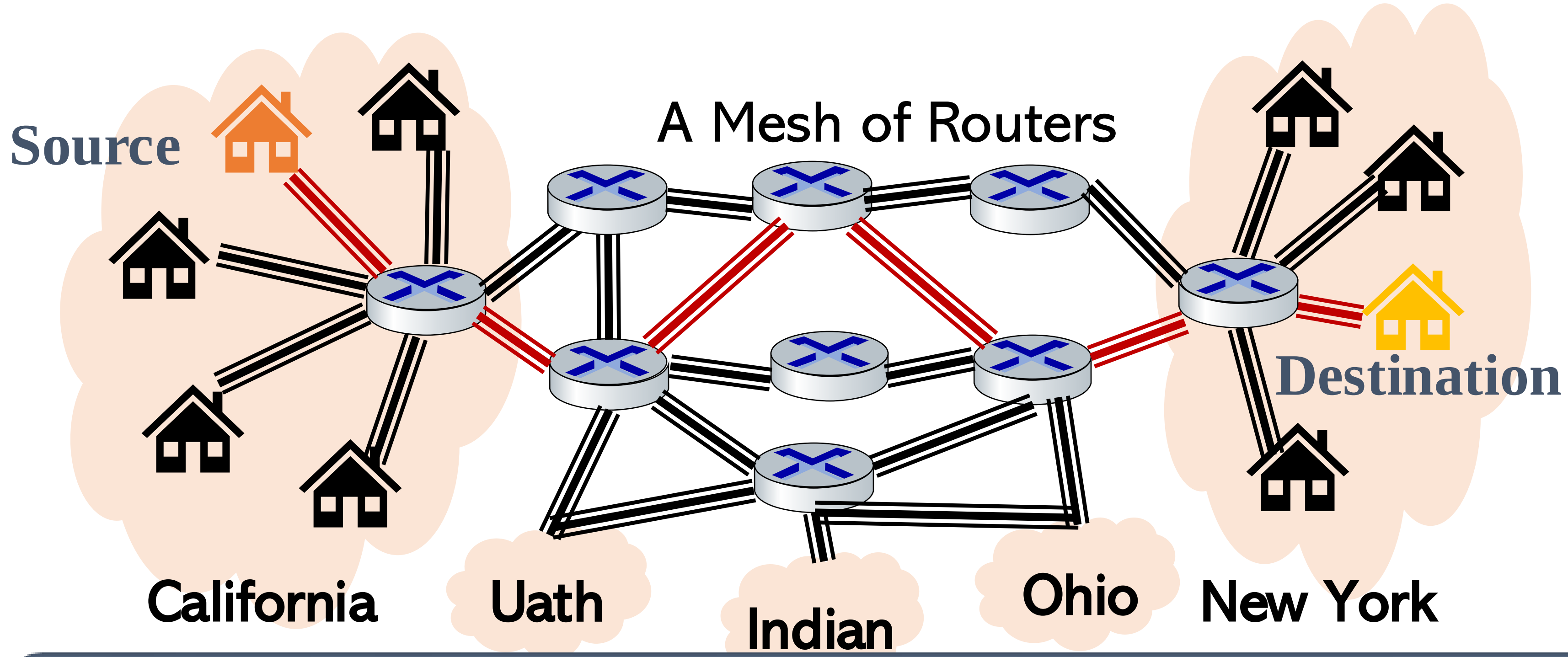


What's Layers in Computer Networks?
Why we need layers?

Layers in Computer Networks

- In computer networking, **layers** refer to different levels of abstraction that help in designing, implementing, and troubleshooting communication systems.
- The idea behind layering is to break down complex networking functions into smaller, manageable parts.
- Each layer performs a specific role and interacts with the layers directly above and below it.

Example: Two types of Computer Networks

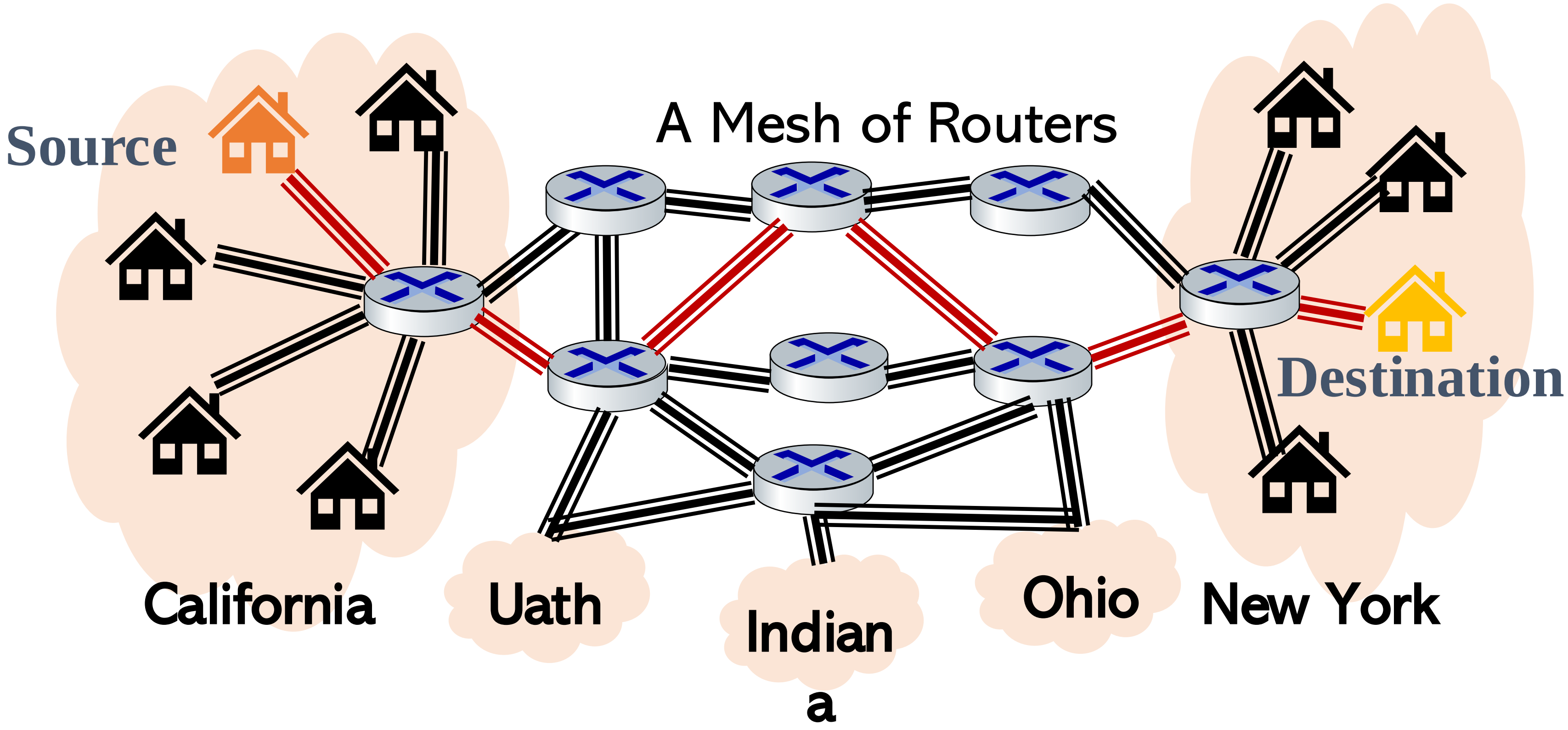


The Network ask: Give me

- **Data you want to transmit**
- **The IP of the destination**
- **Which router the packet should travel**
- **What happens if the packets are dropped**
- **How should I tell you if the**

Of course you can tell the network the answers after you taking this course ☺ !

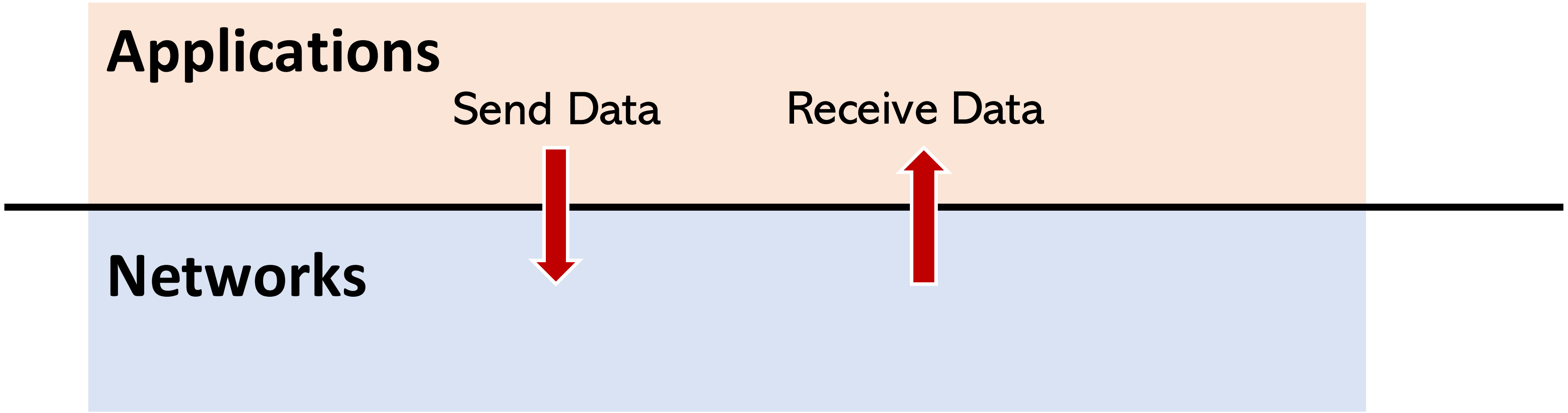
Example: Two types of Computer Networks



Another Network Says

- **Don't worry, just give me the data and the destination I will handle the other things for you!**

Abstractions make things much simpler



Example: Program Developing

Wi-Fi

Qualcomm
BROADCOM
intel

Cellular

Qualcomm
ERICSSON
MEDIATEK
SAMSUNG

Example: Program Developing

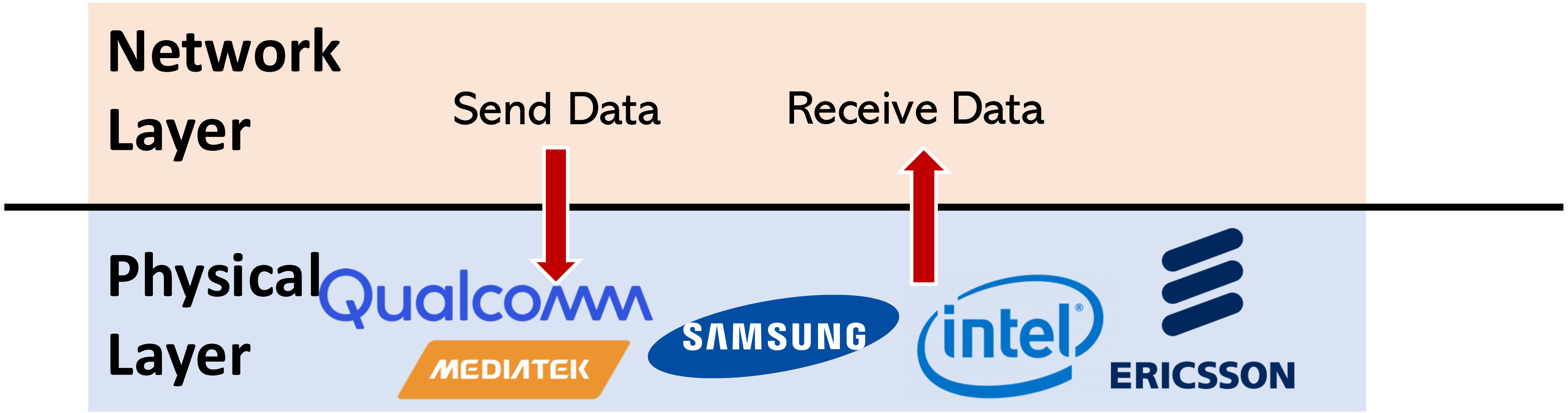
Wi-Fi

Qualcomm
BROADCOM
intel

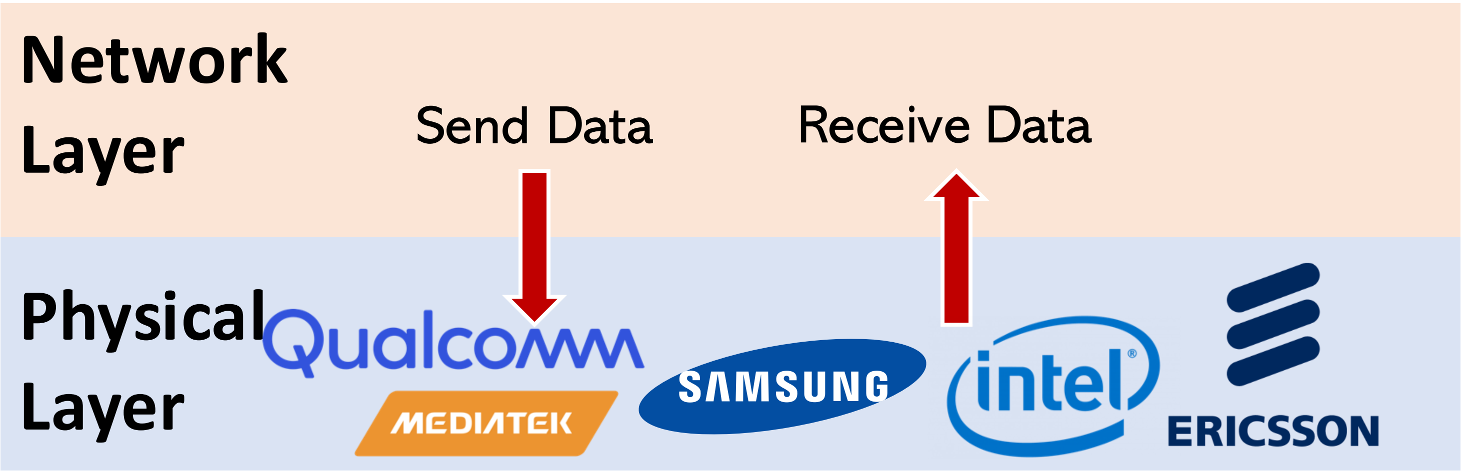
Cellular

Qualcomm
ERICSSON
MEDIATEK
SAMSUNG

Abstractions enables interoperability



Example: Technology Innovation

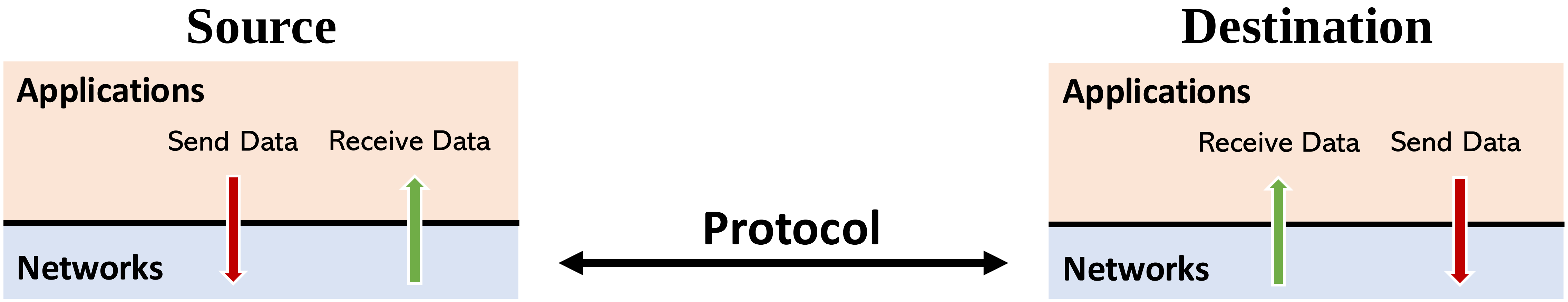


200Mbps ⟶ 2 Gbps

Benefit of Layers

- Modular Design: Makes networking easier to understand and manage.
- Interoperability: Ensures different systems and technologies can communicate.
- Abstraction: Allows developers to focus on specific functions without worrying about the entire network.
- Innovation: Easy to deploy new technology without changing the whole network stack
- Troubleshooting: Simplifies diagnosing issues by isolating problems within specific layers.

Structure of the layer design

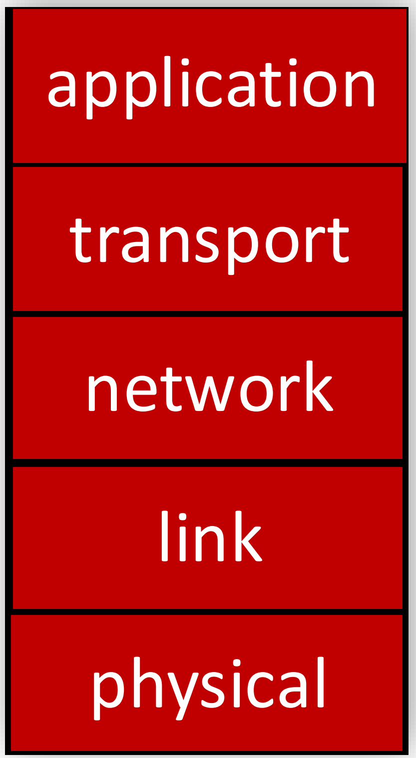


- **Service:** **What** a layer does
- **Service interface:** **How to access** the service
  - Interface for the layer **above**

- **Protocol interface:** **How peers communicate** to implement service
  - Set of rules and formats that govern the communication **between two Internet hosts**

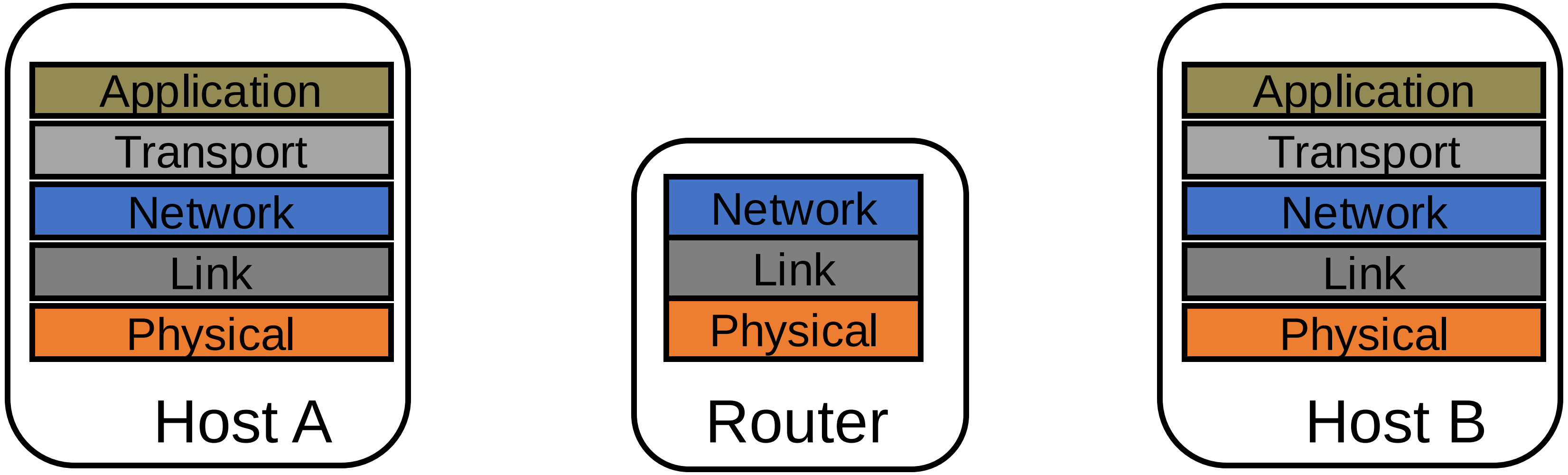
Layered Internet protocol stack

- *application:* supporting network applications
  - HTTP, IMAP, SMTP, DNS
- *transport:* process-process data transfer
  - TCP, UDP
- *network:* routing of datagrams from source to destination
  - IP, routing protocols
- *link:* data transfer between neighboring network elements
  - Ethernet, 802.11 (WiFi), PPP
- *physical:* bits “on the wire”



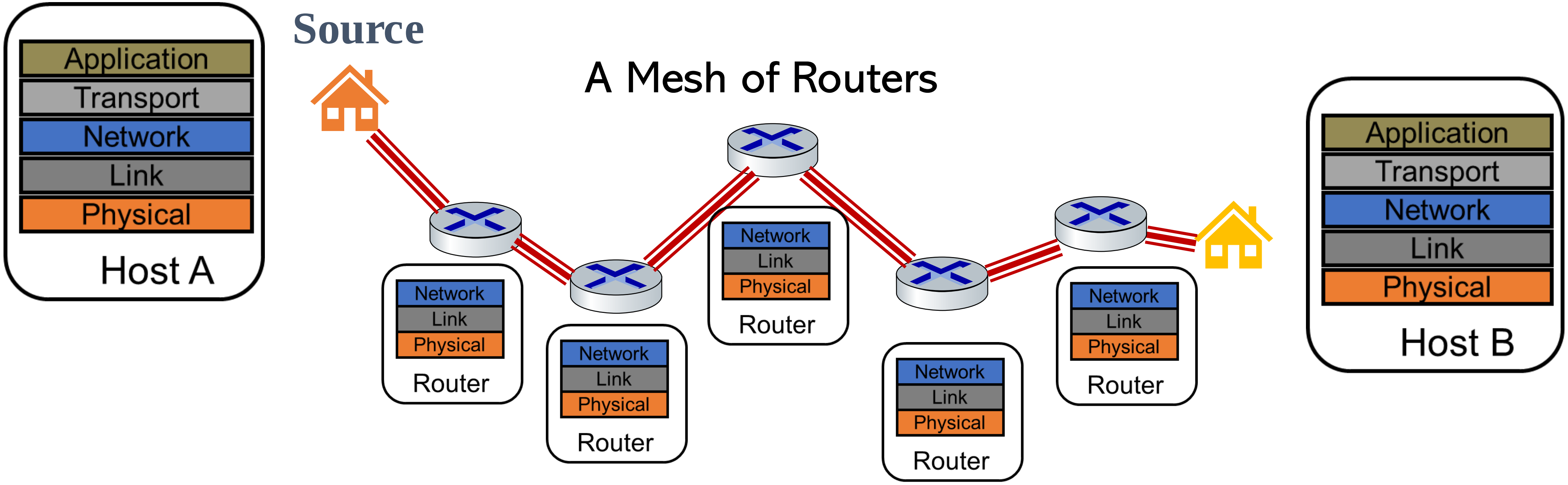
Layers inside network

- Five layers
  - **Lower three layers** are implemented **everywhere**
  - **Top two layers** are implemented **only at end hosts**
    - **Their protocols are *end-to-end***



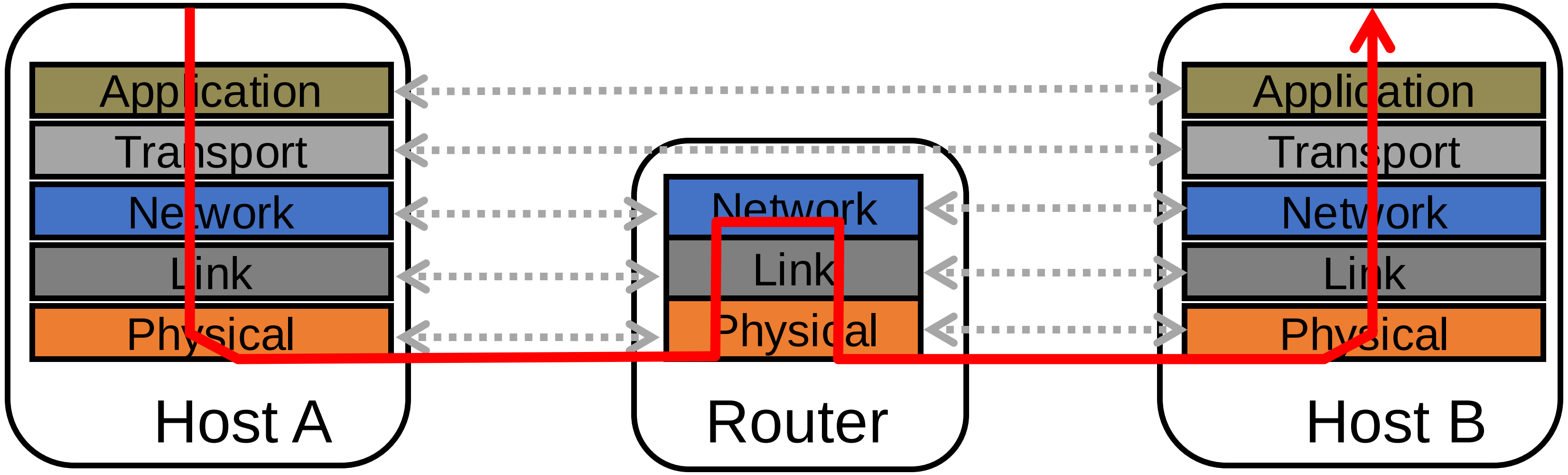
Layers inside network

- Five layers
  - **Lower three layers** are implemented **everywhere**
  - **Top two layers** are implemented **only at end hosts**
    - **Their protocols are *end-to-end***



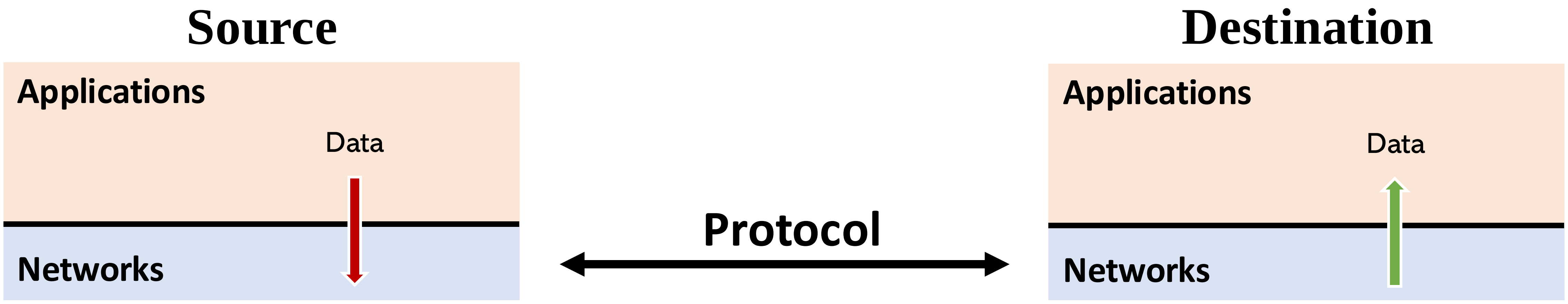
Physical path across the Internet

- Communication **goes down to physical network**
- Then from **network peer to peer**
- Then **up to the relevant layer**



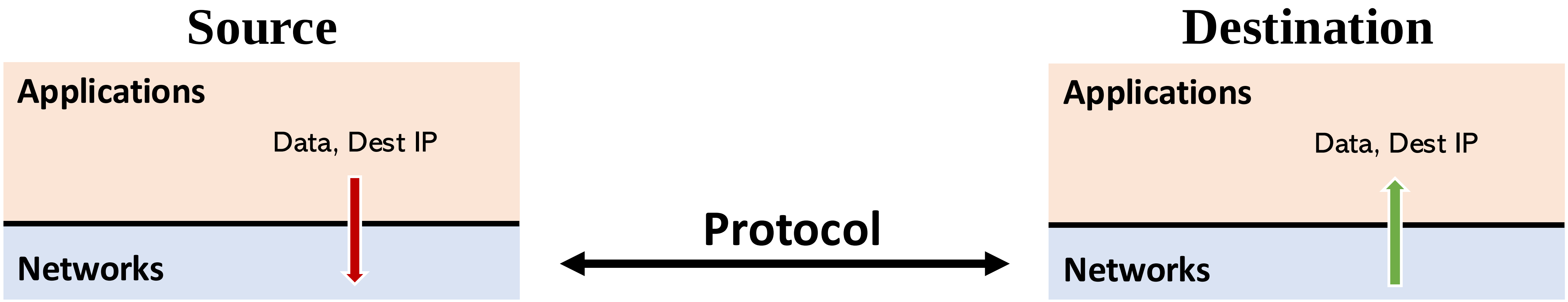
Encapsulation in Networks

- Addressing



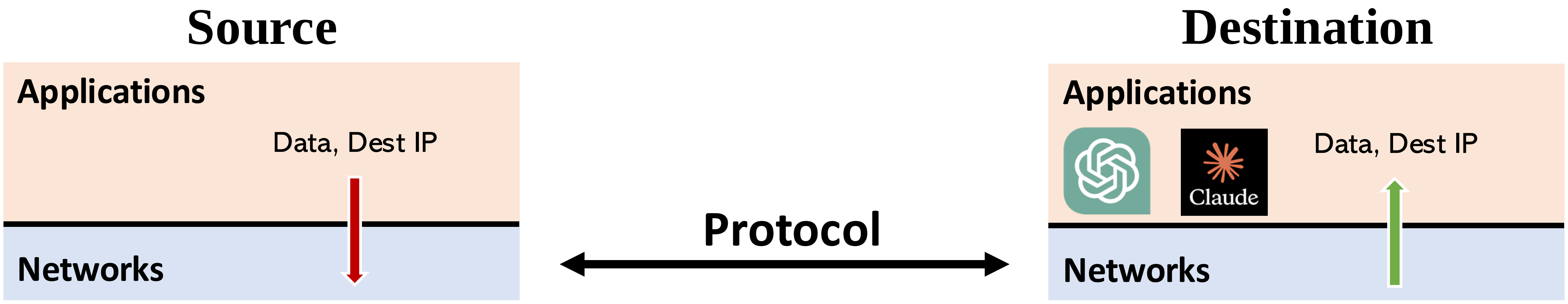
Encapsulation in Networks

- Addressing and routing



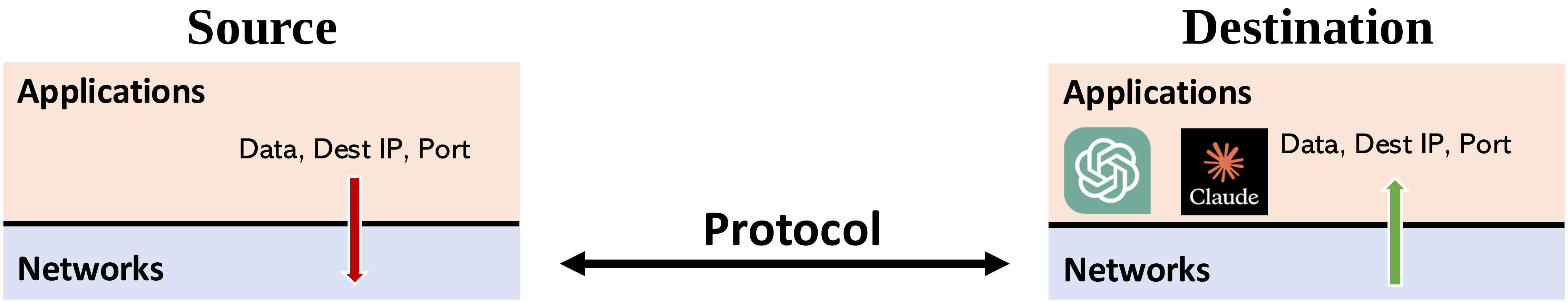
Encapsulation in Networks

- Addressing and routing
- Multiplexing



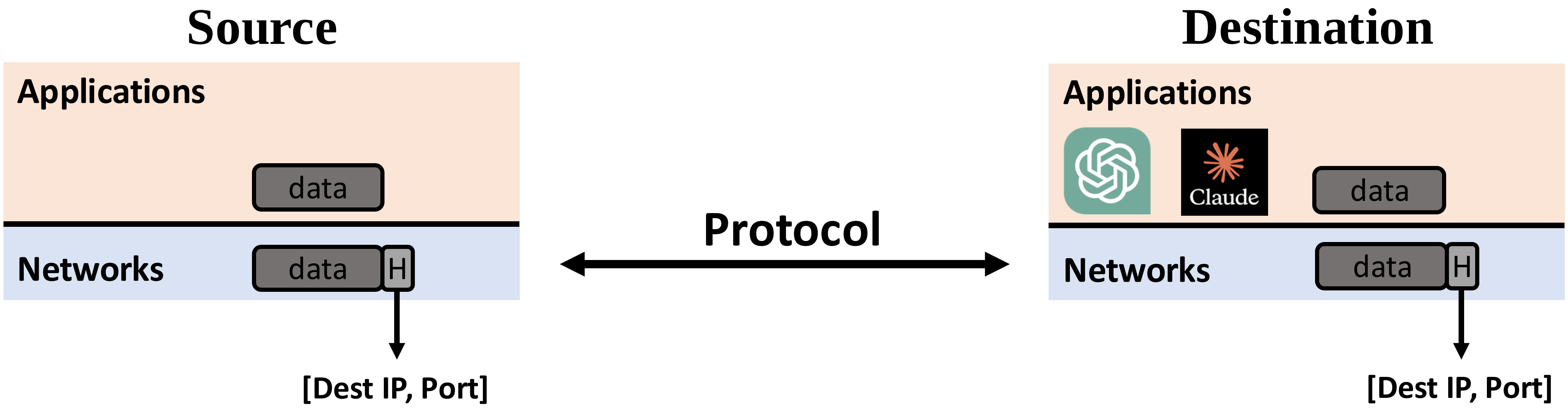
Encapsulation in Networks

- Addressing and routing
- Multiplexing



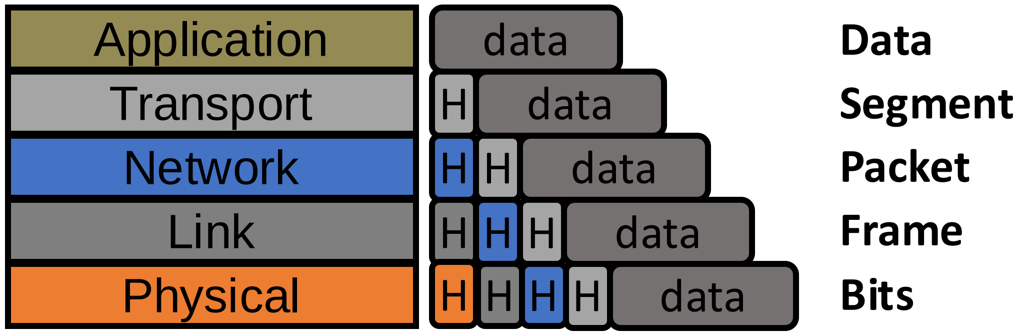
Encapsulation in Networks

- Addressing and routing
- Multiplexing



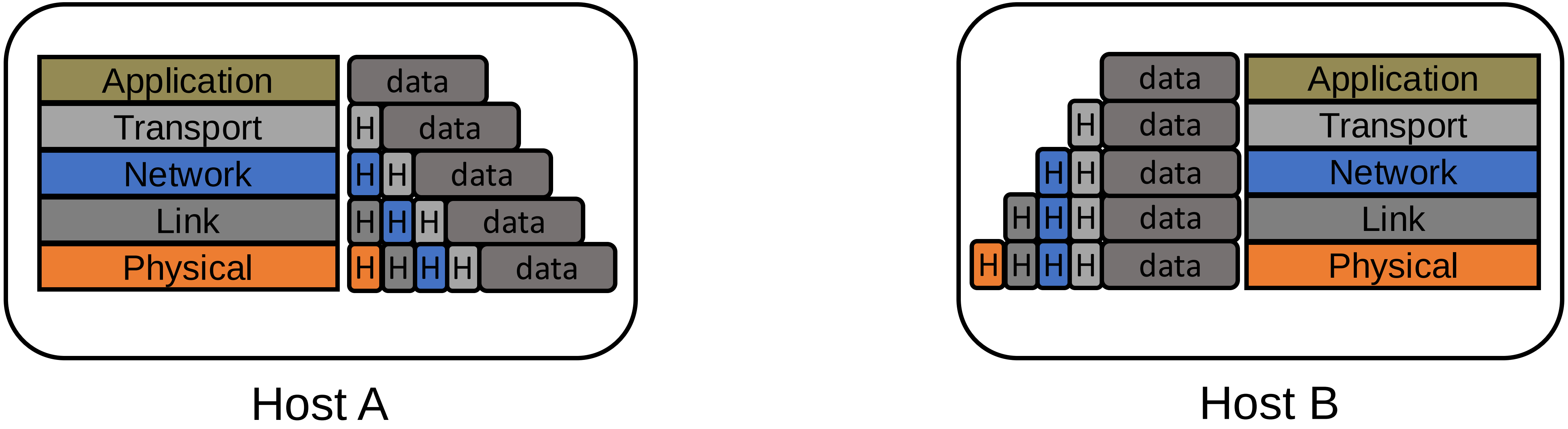
Encapsulation in Networks

- Encapsulation happens at all the layers

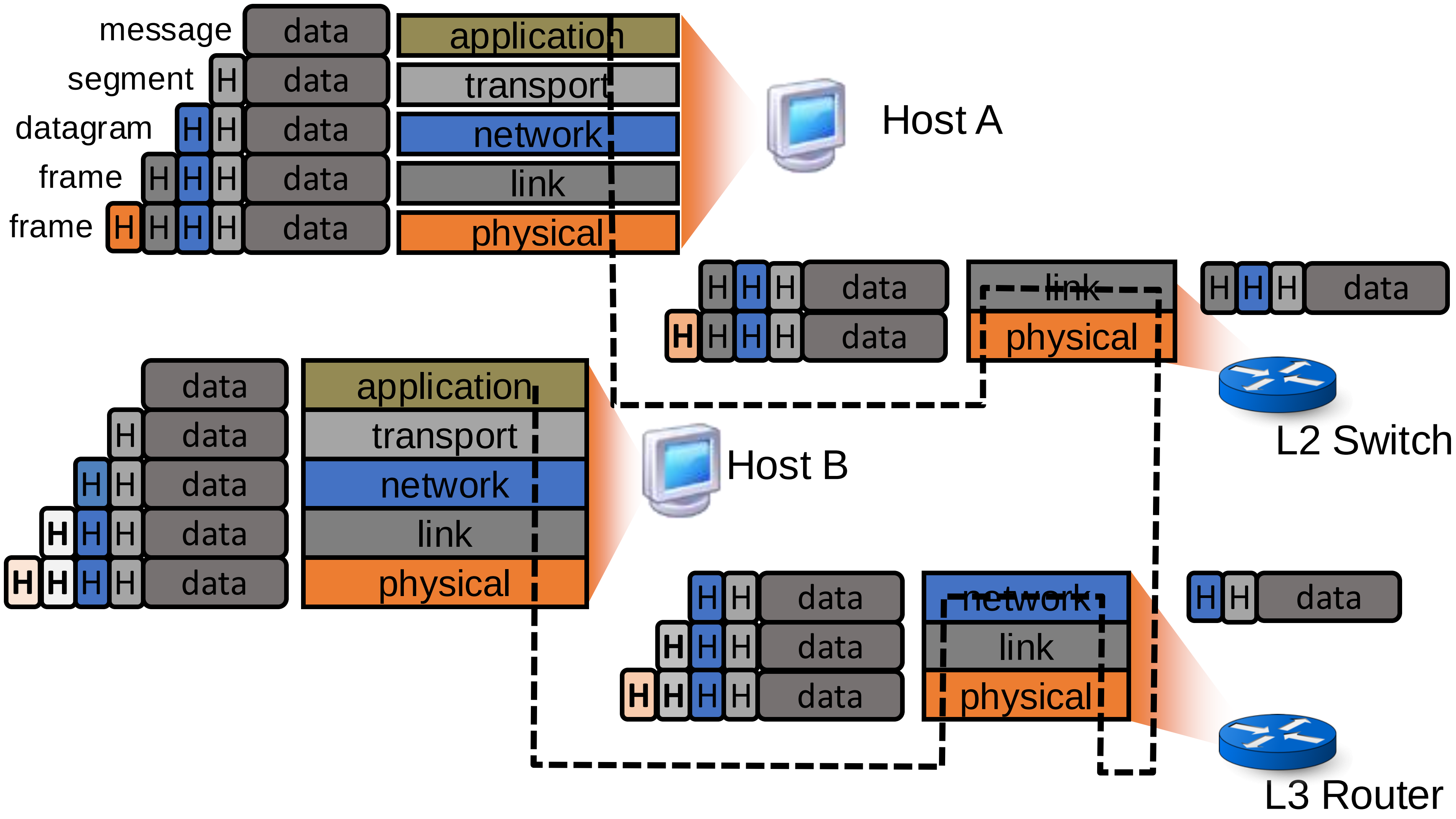


Encapsulation in Networks

- Encapsulation happens at all the layers
- On reception, layer **inspects and removes** its own header
  - Higher layers **don't see** lower layers' headers



Encapsulation in the Internet



Chapter 1: roadmap

- What *is* the Internet?
- What *is* a protocol?
- Network edge: hosts, access network, physical media
- Network core: packet/circuit switching, internet structure
- Performance: loss, delay, throughput
- Protocol layers, service models
- Security
- History

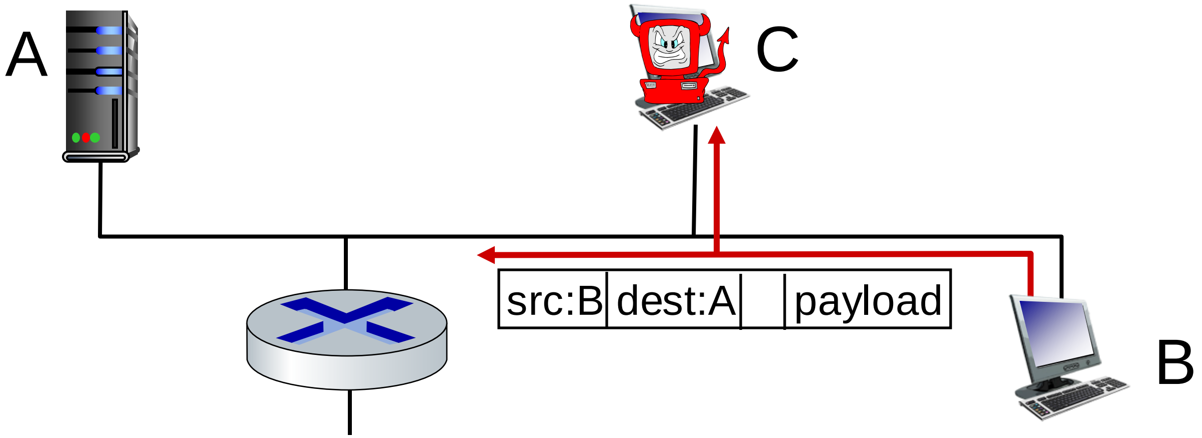
Network security

- Internet not originally designed with (much) security in mind
  - *original vision:* "a group of mutually trusting users attached to a transparent network" ☺
  - Internet protocol designers playing "catch-up"
  - security considerations in all layers!
- We now need to think about:
  - how bad guys can attack computer networks
  - how we can defend networks against attacks
  - how to design architectures that are immune to attacks

Bad guys: packet interception

packet "sniffing":

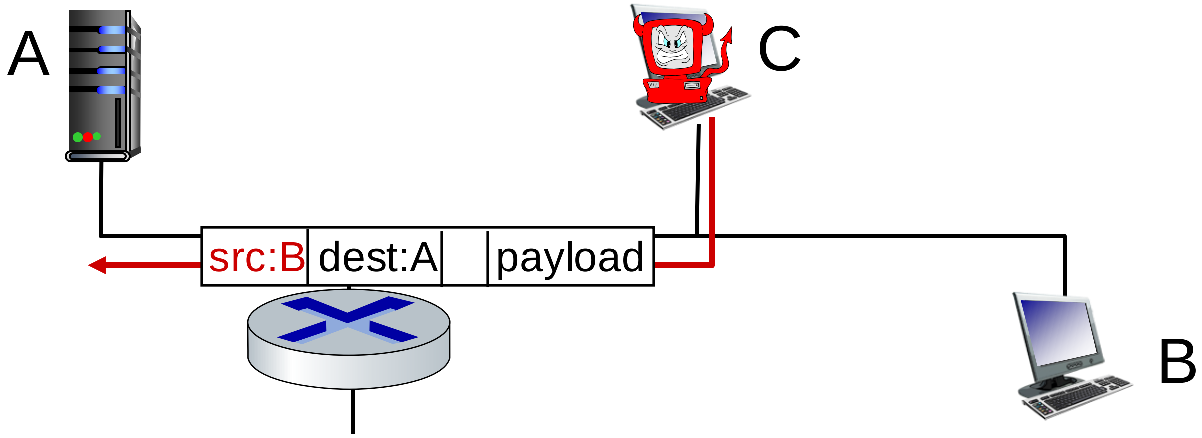
- broadcast media (shared Ethernet, wireless)
- promiscuous network interface reads/records all packets (e.g., including passwords!) passing by



Wireshark software used for our end-of-chapter labs is a (free) packet-sniffer

Bad guys: fake identity

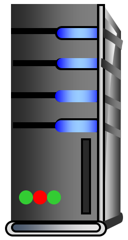
IP spoofing: injection of packet with false source address



Bad guys: denial of service

Denial of Service (DoS): attackers make resource (server, bandwidth) unavailable to legitimate traffic by overwhelming resource with bogus traffic

Client

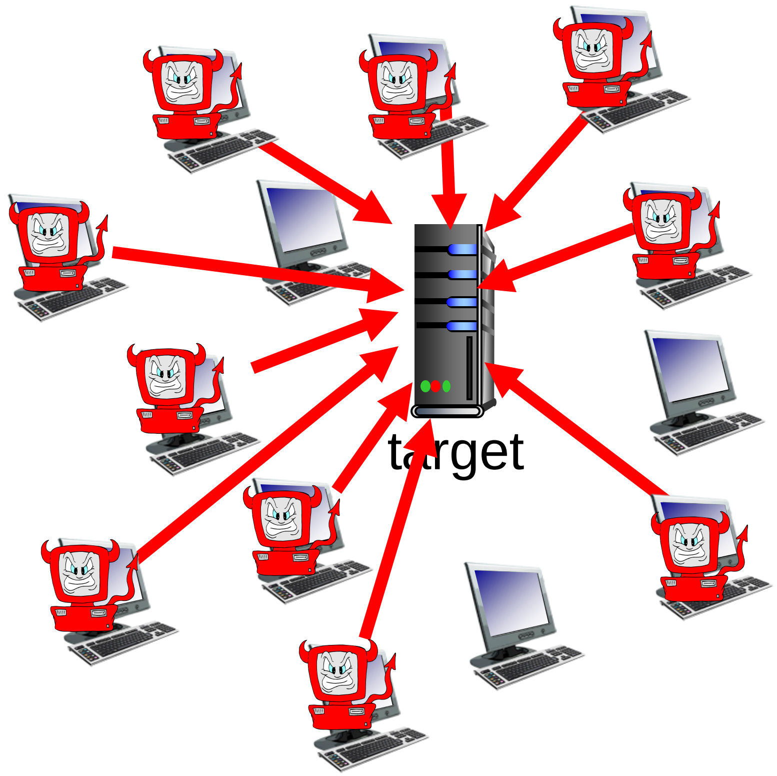


Server

Bad guys: denial of service

Denial of Service (DoS): attackers make resource (server, bandwidth) unavailable to legitimate traffic by overwhelming resource with bogus traffic

1. select target
2. break into hosts around the network
3. send packets to target from compromised hosts



Lines of defense:

- authentication: proving you are who you say you are
  - cellular networks provides hardware identity via SIM card; no such hardware assist in traditional Internet
- confidentiality: via encryption
- integrity checks: digital signatures prevent/detect tampering
- access restrictions: password-protected VPNs
- firewalls: specialized “middleboxes” in access and core networks:
  - off-by-default: filter incoming packets to restrict senders, receivers, applications
  - detecting/reacting to DOS attacks

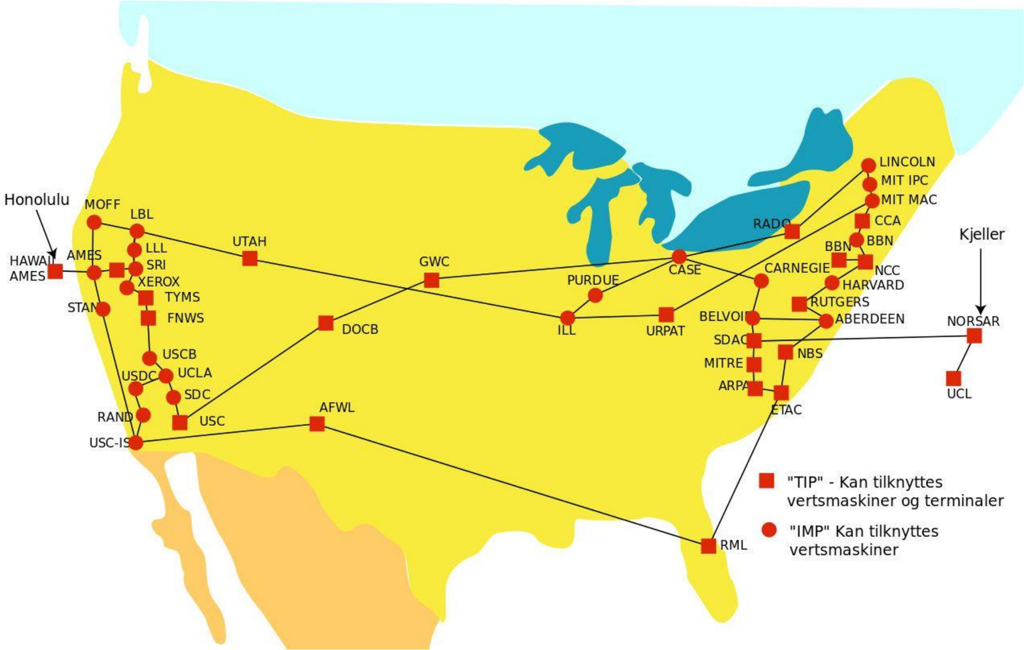
Chapter 1: roadmap

- What *is* the Internet?
- What *is* a protocol?
- Network edge: hosts, access network, physical media
- Network core: packet/circuit switching, internet structure
- Performance: loss, delay, throughput
- Security
- Protocol layers, service models
- History

ARPANET – The Birth of the Internet (1969)

- Funded by DARPA (U.S. Defense Advanced Research Projects Agency).
- First message sent **Oct 29, 1969** between UCLA and Stanford.
- Key technology: **Packet switching**.
- Connected universities and research institutions.

The first message sent over ARPANET happened on Oct. 29, 1969. Charley Kline, who was a student at the University of California Los Angeles (UCLA), tried to log in to the mainframe at the Stanford Research Institute (SRI). He successfully typed in the characters *L* and *O*, but the computer crashed when he typed the *G* of the command `LOGIN`. They were able to overcome the initial crash, however, and had a successful connection that same day.



Expanding ARPANET and TCP/IP (1970s-1980s)

- **1973:** First international connection (UK & Norway).
- **1974: TCP/IP protocol** proposed by **Vinton Cerf & Bob Kahn**.
- **1983:** ARPANET switches to **TCP/IP**, officially creating the Internet.
- **1986: NSFNET** expands network beyond military & academia.

Commercialization and the World Wide Web (1990s)

- **1991:** Tim Berners-Lee develops **World Wide Web (WWW)**.
- **1993:** First web browser **Mosaic** released (led to Netscape).
- **1995:** NSF lifts commercial restrictions → **Birth of Internet Service Providers (ISPs)**.
- **1998:** Google founded, changing web search.

The Dot-Com Boom and Wireless Internet (2000s)

- Rise of **e-commerce (Amazon, eBay, PayPal)**.
- **Wi-Fi, broadband, mobile networks** increase connectivity.
- **2004:** Facebook launches, followed by **social media explosion**.
- **2007:** Apple releases the **iPhone**, boosting mobile Internet.

Modern Internet (2010s-Present)

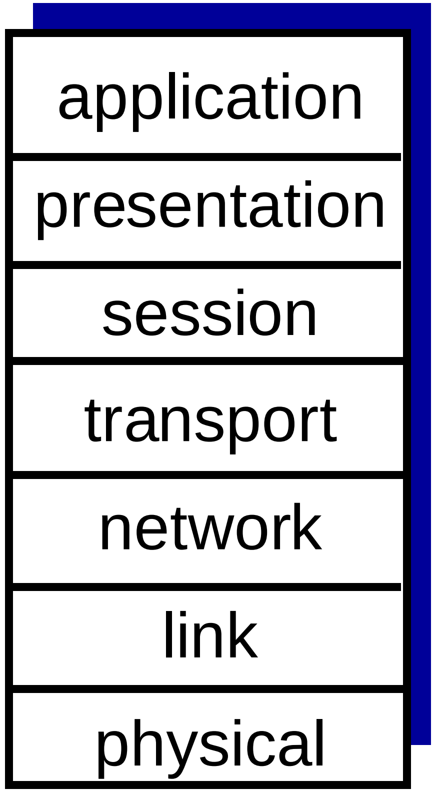
- Cloud computing, streaming services, and 5G.
- **IoT (Internet of Things)** connects smart devices.
- **AI & Machine Learning** reshape content delivery.

Additional Chapter 1 slides

ISO/OSI reference model

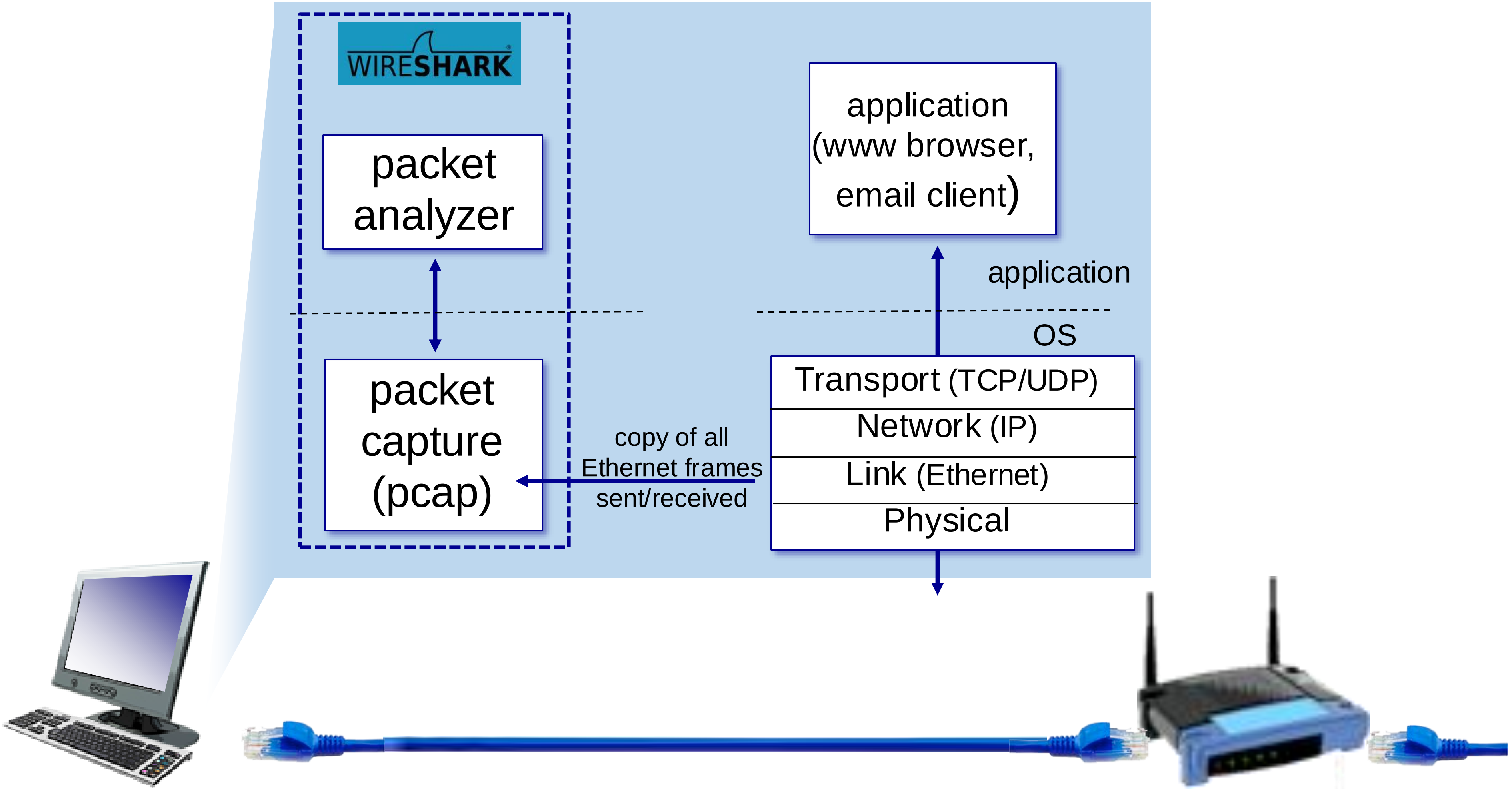
Two layers not found in Internet protocol stack!

- *presentation:* allow applications to interpret meaning of data, e.g., encryption, compression, machine-specific conventions
- *session:* synchronization, checkpointing, recovery of data exchange
- Internet stack "missing" these layers!
  - these services, *if needed,* must be implemented in application
  - needed?



The seven layer OSI/ISO reference model

Wireshark


WIRESHARK
packet analyzer
packet capture (pcap)
copy of all Ethernet frames sent/received
application (www browser, email client)
application
OS
Transport (TCP/UDP)
Network (IP)
Link (Ethernet)
Physical