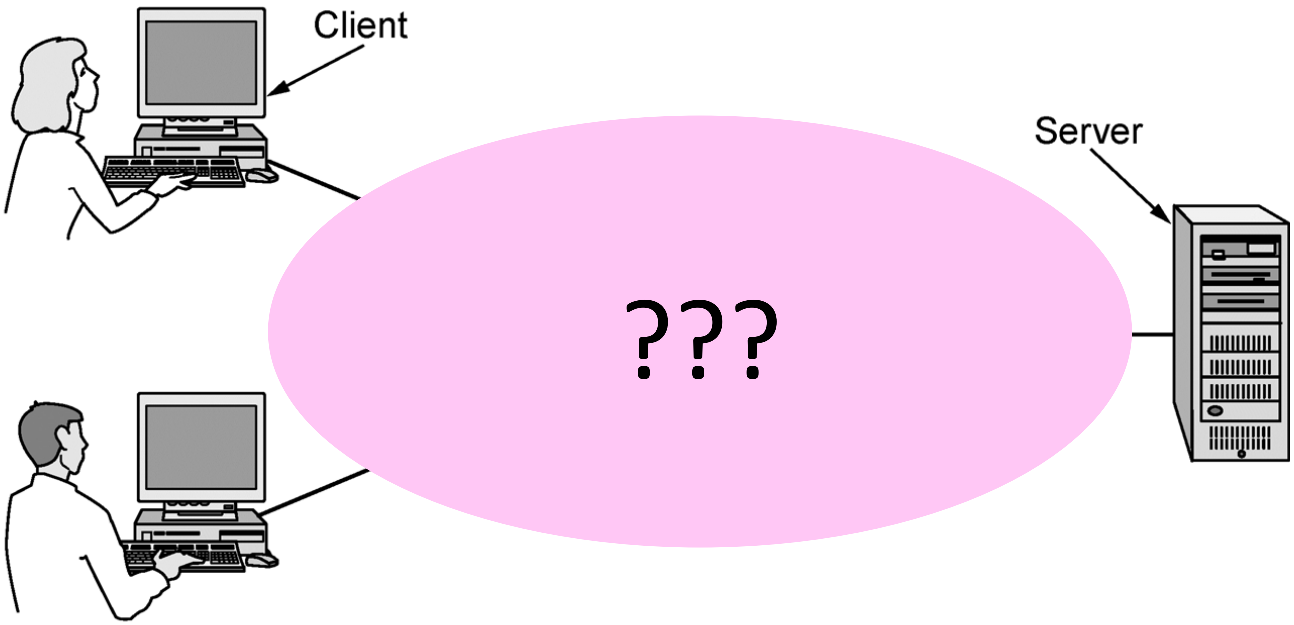
Introduction to Computer Networks

Arvind Krishnamurthy

Computer Science & Engineering

UNIVERSITY of WASHINGTON

Focus of the course



Focus of the course (2)

- Three “networking” topics:

Distributed systems
Networking
Communications

- We’re in the middle

The Main Point

1. To learn how the Internet works »
   – What really happens when you "browse the web"?
   – What are TCP/IP, DNS, HTTP, NAT, VPNs, 802.11 etc. anyway?

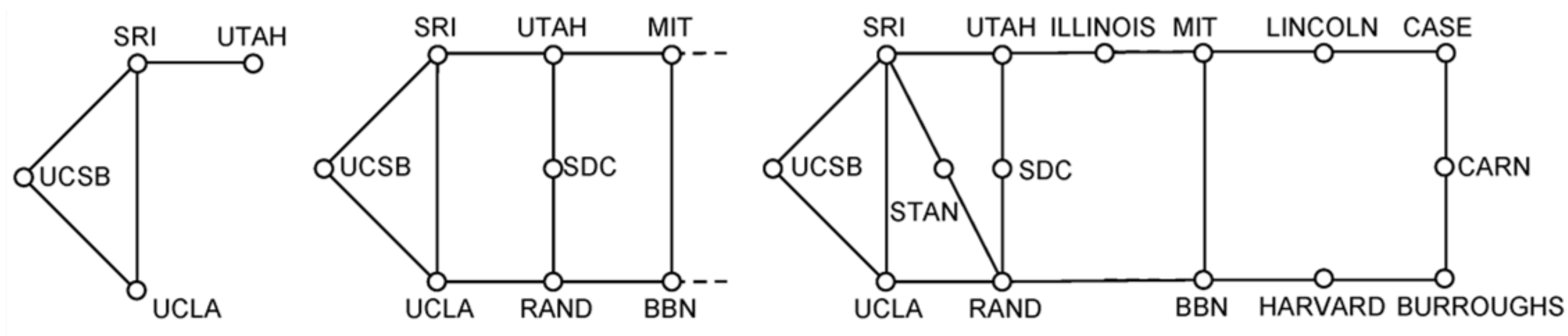
2. To learn the fundamentals of computer networks

Why learn about the Internet?

1. Curiosity »
2. Impact on our world »
3. Job prospects!

From this experimental network …

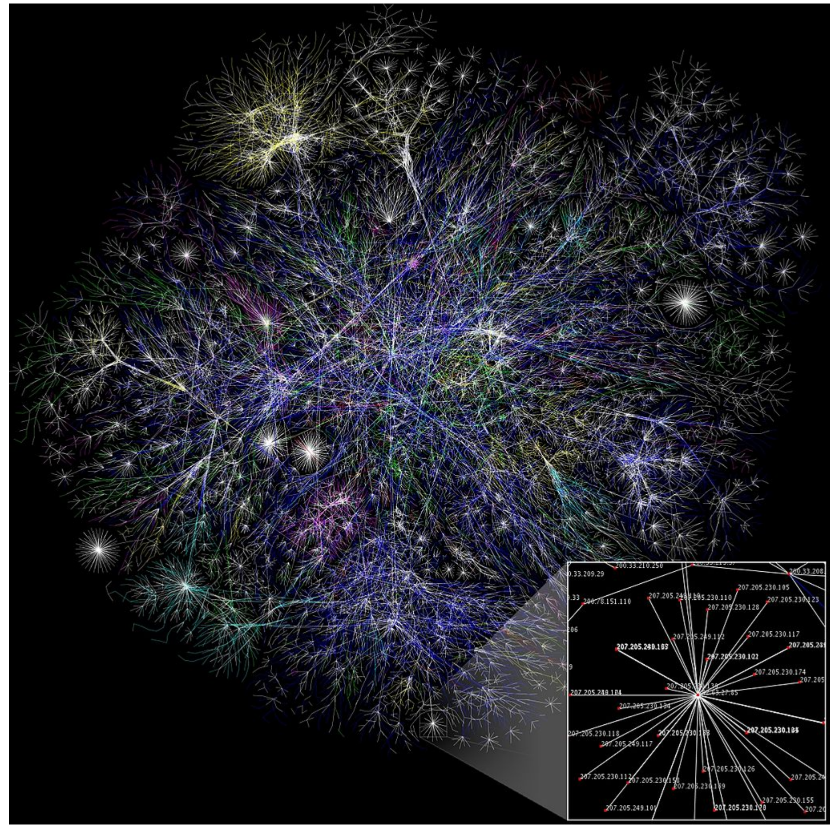
ARPANET ~1970



(a) Dec. 1969. (b) July 1970. (c) March 1971.

To this!
Internet ~2005

- An everyday institution used at work, home, and on-the-go
- Visualization contains millions of links



Attribution: By The Opte Project [CC-BY-2.5], via Wikimedia Commons

Question

- What do you think are the issues that one has to tackle to grow from a small network to an extremely large network?

Internet – Societal Impact

- An enabler of societal change
  - Easy access to knowledge
  - Electronic commerce
  - Personal relationships
  - Discussion without censorship

Internet – Economic impact

- An engine of economic growth
  - Advertising-sponsored search
  - "Long tail" online stores
  - Online marketplaces
  - Crowdsourcing

The Main Point (2)

1. To learn how the Internet works
2. To learn the fundamentals of computer networks
   – What hard problems must they solve?
   – What design strategies have proven valuable?

Why learn the Fundamentals?

1. Apply to all computer networks
2. Intellectual interest »
3. Change / reinvention »

Fundamentals – Intellectual Interest

- Example key problem: Reliability!
  - Any part of the Internet might fail
  - Messages might be corrupted
  - So how do we provide reliability?

Fundamentals – Intellectual Interest (2)

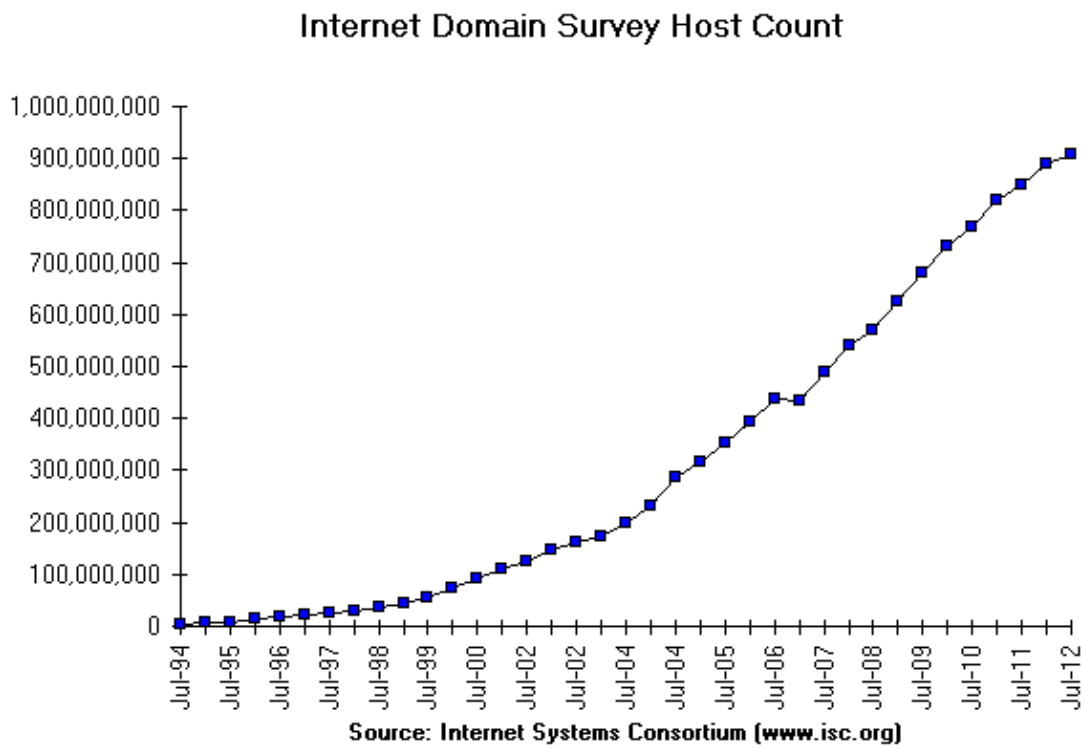
Key problem	**Example solutions**
Reliability despite failures	Codes for error detection/correction (§3.2, 3.3) Routing around failures (§5.2)
Network growth and evolution	Addressing (§5.6) and naming (§7.1) Protocol layering (§1.3)
Allocation of resources like bandwidth	Multiple access (§4.2) Congestion control (§5.3, 6.3)
Security against various threats	Confidentiality of messages (§8.2, 8.6) Authentication of communicating parties (§8.7)

Fundamentals – Reinvention

- The Internet is constantly being re-invented!
  - Growth over time and technology trends drive upheavals in Internet design and usage »
- Today's Internet is different from yesterday's
  - And tomorrow's will be different again
  - But the fundamentals remain the same

Fundamentals – Reinvention (2)

- At least a billion Internet hosts and growing …



Fundamentals – Reinvention (3)

- Examples of upheavals in the past 1-2 decades

Growth / Tech Driver	**Upheaval**
Emergence of the web	Content Distribution Networks
Digital songs/videos	Peer-to-peer file sharing
Falling cost/bit	Voice-over-IP calling
Many Internet hosts	IPv6
Wireless advances	Mobile devices

Not a Course Goal

- To learn IT job skills
  - How to configure equipment
    - e.g., Cisco certifications
  - But course material is relevant, and we use hands-on tools

Course Mechanics

- Course Administration
  - Everything you need to know will be on the course web page:
    http://www.cs.washington.edu/461/
- Teaching Assistants:
  - Qiao Zhang
  - Danyang Zhuo

Course Logistics

1. Reading
2. Projects/Homeworks: 55%
3. Mid-term/final: 45%

Introduction to Computer Networks

Uses of Networks (§1.1)

Computer Science & Engineering

UNIVERSITY of WASHINGTON

Example Uses of Networks

- Work:
  - Email, file sharing, printing, …
- Home:
  - Movies / songs, news, calls / video / messaging, e-commerce, …
- Mobile:
  - Calls / texts, games, videos, maps, information access …

Example Uses of Networks

- Work:
  - Email, file sharing, printing, …
- Hom
  - M … s /
    vi … merce, …
- Mobile:
  - Calls / texts, games, videos, maps, information access …

What do these uses tell us about why we build networks?

For User Communication

- From the telephone onwards:
  - VoIP (voice-over-IP)
  - Video conferencing
  - Instant messaging
  - Social networking

→What is the metric we need to be optimizing for these uses?

For Resource Sharing

- Many users may access the same underlying resource
  - E.g., 3D printer, search index, machines in the cloud

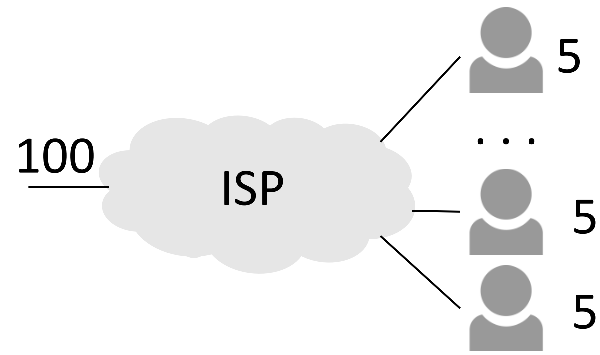
→ More cost effective than dedicated resources per user
  - Even network links are shared via statistical multiplexing »

Statistical Multiplexing

- Sharing of network bandwidth between users according to the statistics of their demand
  - (Multiplexing just means sharing)
  - Useful because users are mostly idle and their traffic is bursty
- Key question:
  - How much does it help?

Statistical Multiplexing (2)

- Example: Users in an ISP network
  - Network has 100 Mbps (units of bandwidth)
  - Each user subscribes to 5 Mbps, for videos
  - But a user is active only 50% of the time …

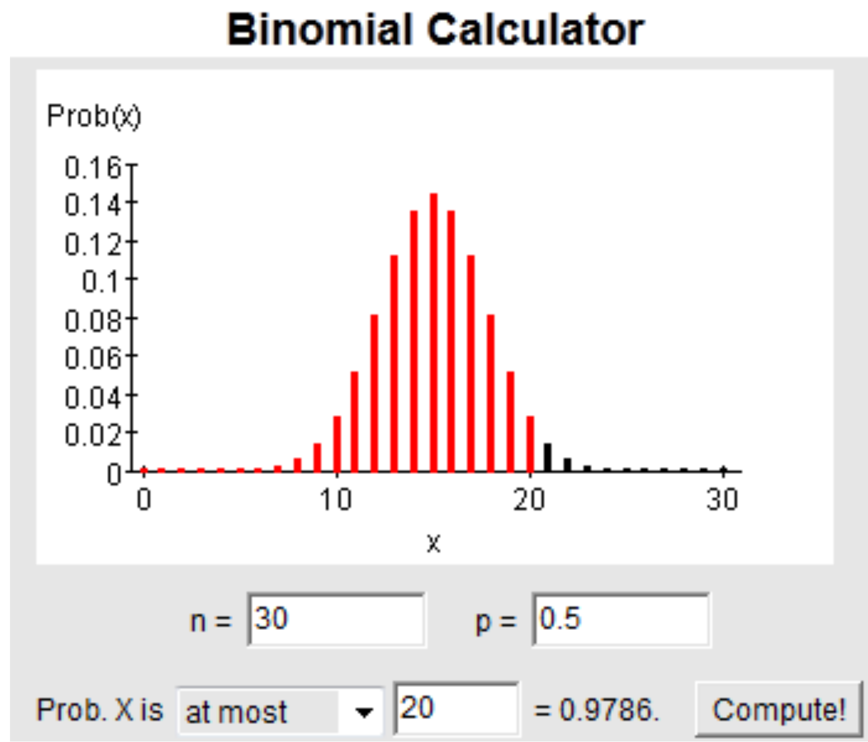


- How many users can the ISP support?
  - With dedicated bandwidth for each user:
  - Probability all bandwidth is used:

Statistical Multiplexing (3)

- With 30 users, still unlikely (2% chance) to need more than 100 Mbps!
  - Binomial probabilities

→ Can serve more users with the same size network
  - Statistical multiplexing gain is 30/20 or 1.5X
  - But may get unlucky; users will have degraded service



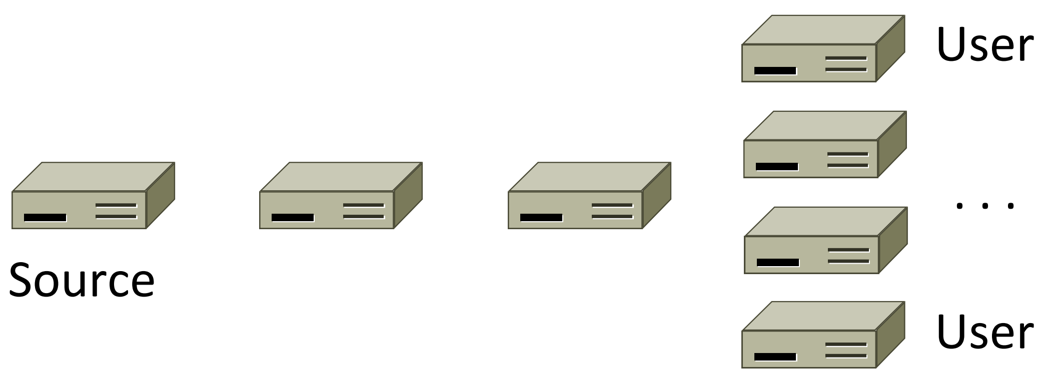
For Content Delivery

- Same content is delivered to many users
  - Videos (large), songs, apps and upgrades, web pages, …

→What is the metric that we want to optimize in such cases?

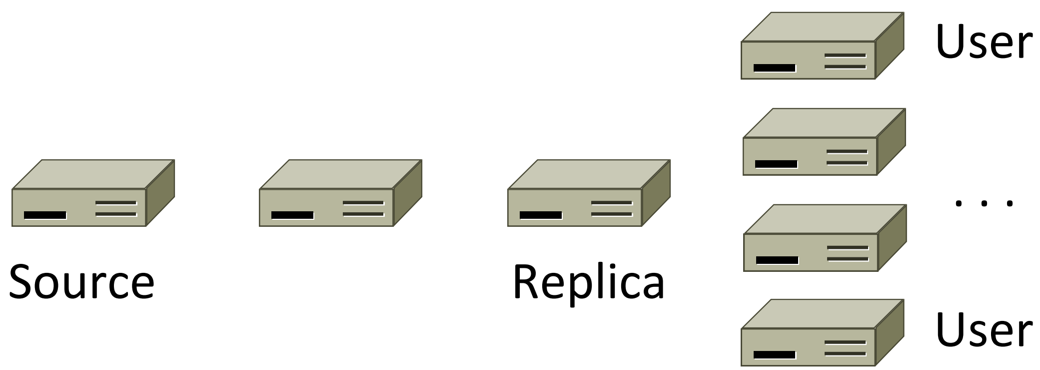
Content Delivery (2)

- Sending content from the source to 4 users takes 4 x 3 = 12 “network hops” in the example



Content Delivery (3)

- But sending content via replicas takes only 4 + 2 = 6 "network hops"



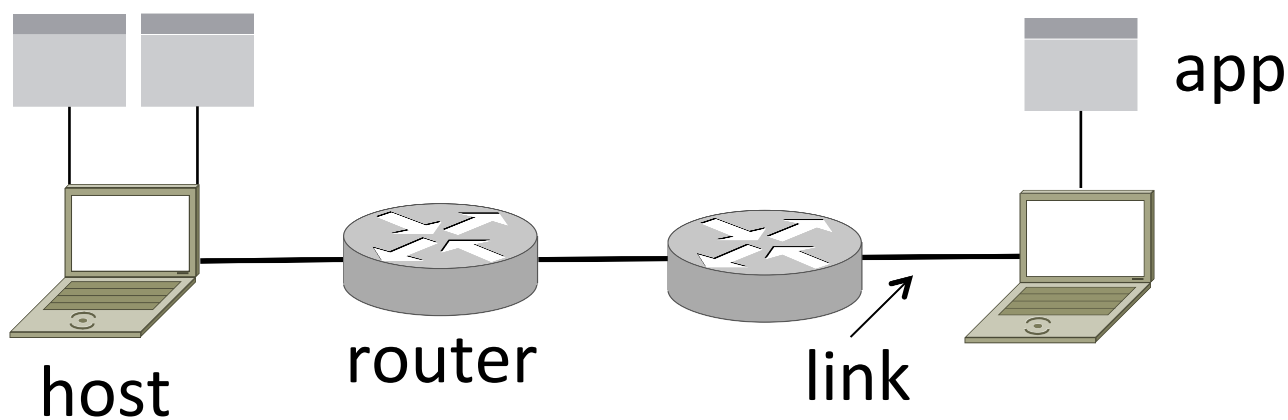
Introduction to Computer Networks

Network Components (§1.2)

Computer Science & Engineering

UNIVERSITY of WASHINGTON

Parts of a Network



Component Names

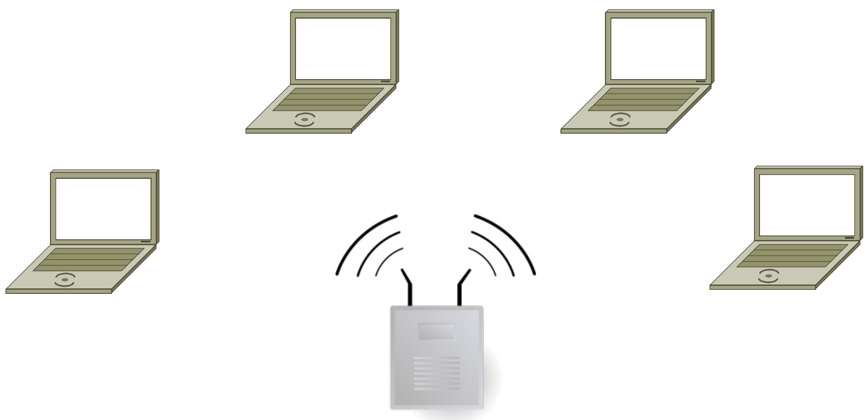
Component	**Function**	**Example**
Application, or app, user	Uses the network	Skype, iTunes, Amazon
Host, or end-system, edge device, node, source, sink	Supports apps	Laptop, mobile, desktop
Router, or switch, node, intermediate system, ...	Relays messages between links	Access point, cable/DSL modem
Link, or channel	Connects nodes	Wires, wireless

Types of Links

- Full-duplex
  - Bidirectional
- Half-duplex
  - Bidirectional
- Simplex
  - unidirectional

Wireless Links

- Message is <u>broadcast</u>
  - Received by all nodes in range
  - Not a good fit with our model



Example Networks

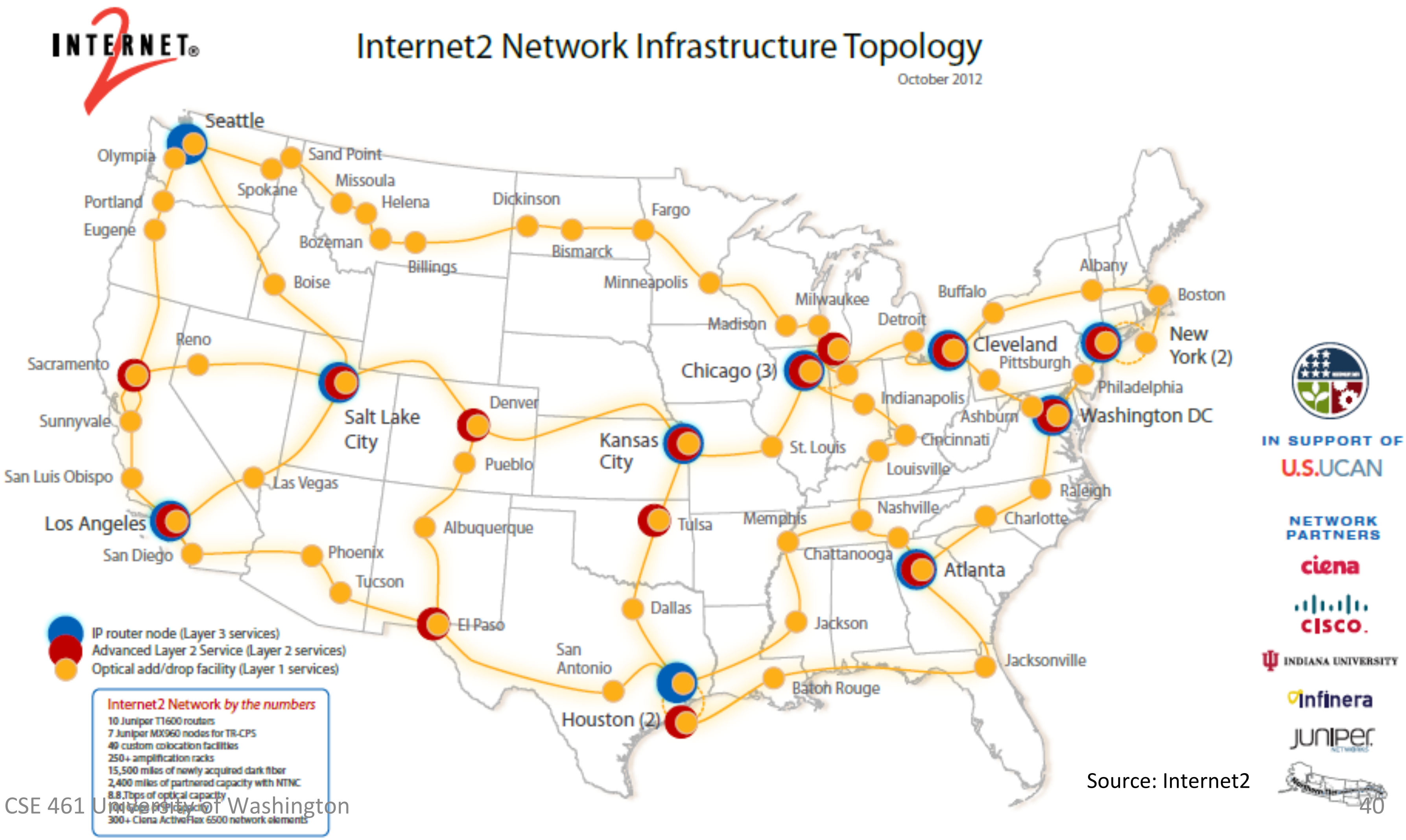
- WiFi (802.11)
- Enterprise / Ethernet
- ISP (Internet Service Provider)
- Cable / DSL
- Mobile phone / cellular (2G, 3G, 4G)
- Bluetooth
- Telephone
- Satellite ...

Network names by scale

Scale	Type	Example
Vicinity	PAN (Personal Area Network)	Bluetooth (e.g., headset)
Building	LAN (Local Area Network)	WiFi, Ethernet
City	MAN (Metropolitan Area Network)	Cable, DSL
Country	WAN (Wide Area Network)	Large ISP
Planet	The Internet (network of all networks)	The Internet!

Internetworks

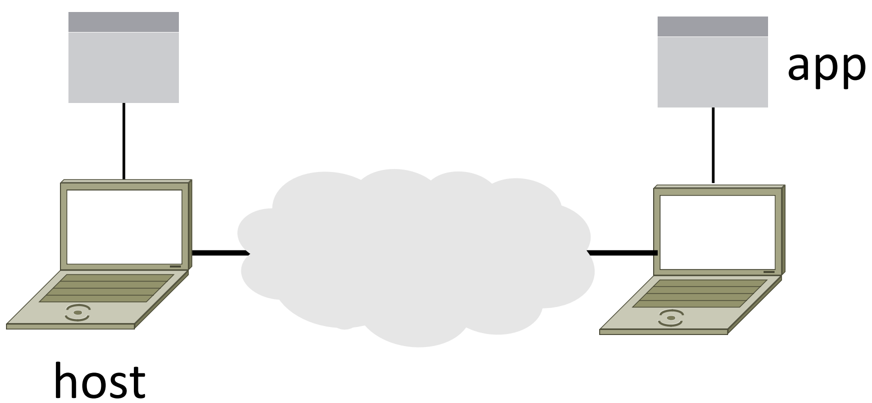
- An internetwork, or internet, is what you get when you join networks together
  - Just another network
- The Internet (capital "I") is the internet we all use



Source: Internet2

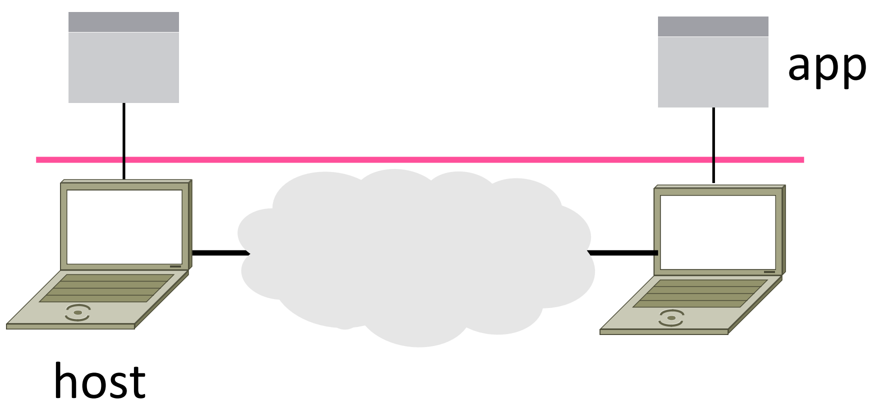
Key Interfaces

- Between (1) apps and network, and (2) network components
  - More formal treatment later on



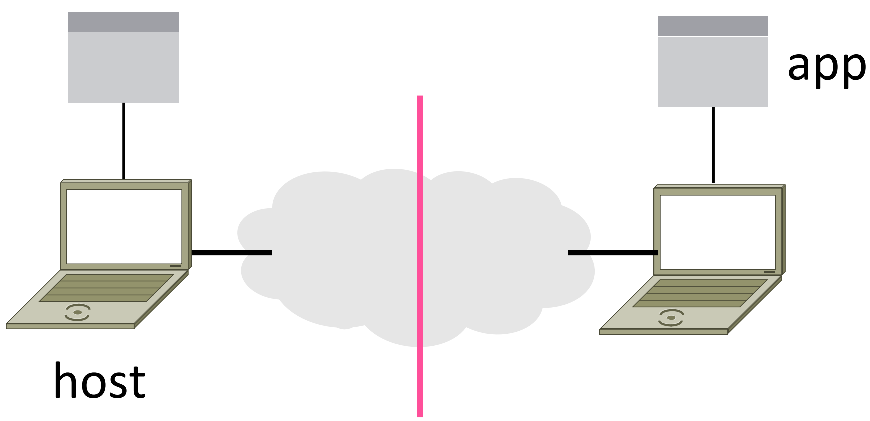
Key Interfaces (2)

1. Network-application interfaces define how apps use the network
   - <u>Sockets</u> are widely used in practice



Key Interfaces (3)

2. Network-network interfaces define how nodes work together
   – Traceroute can peek in the network



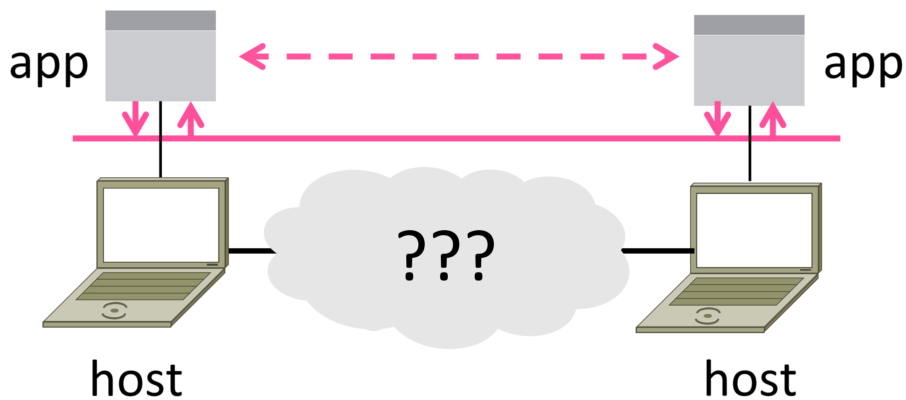
Introduction to Computer Networks

Peeking inside the Network
with Traceroute

Computer Science & Engineering
UNIVERSITY of WASHINGTON

Network Service API Hides Details

- Apps talk to other apps with no real idea of what is inside the network
  - This is good! But you may be curious …

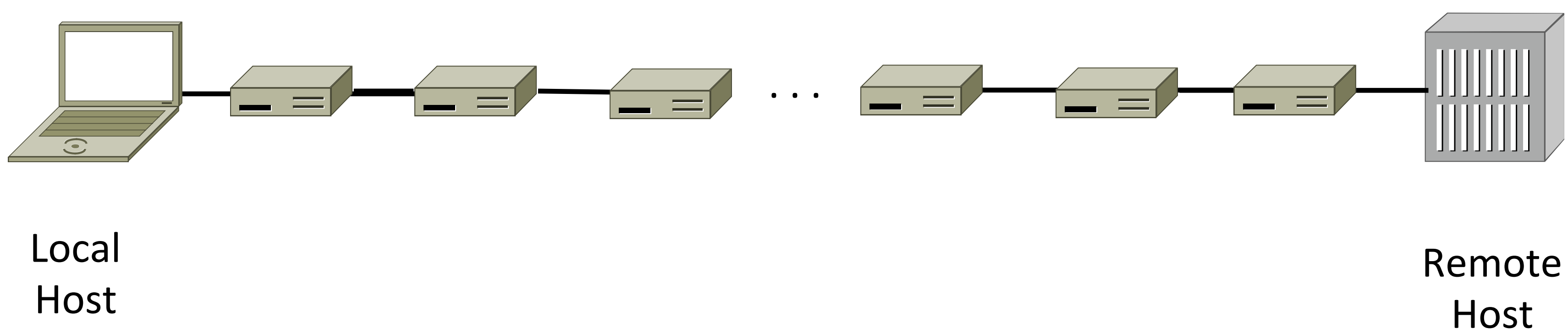


Traceroute

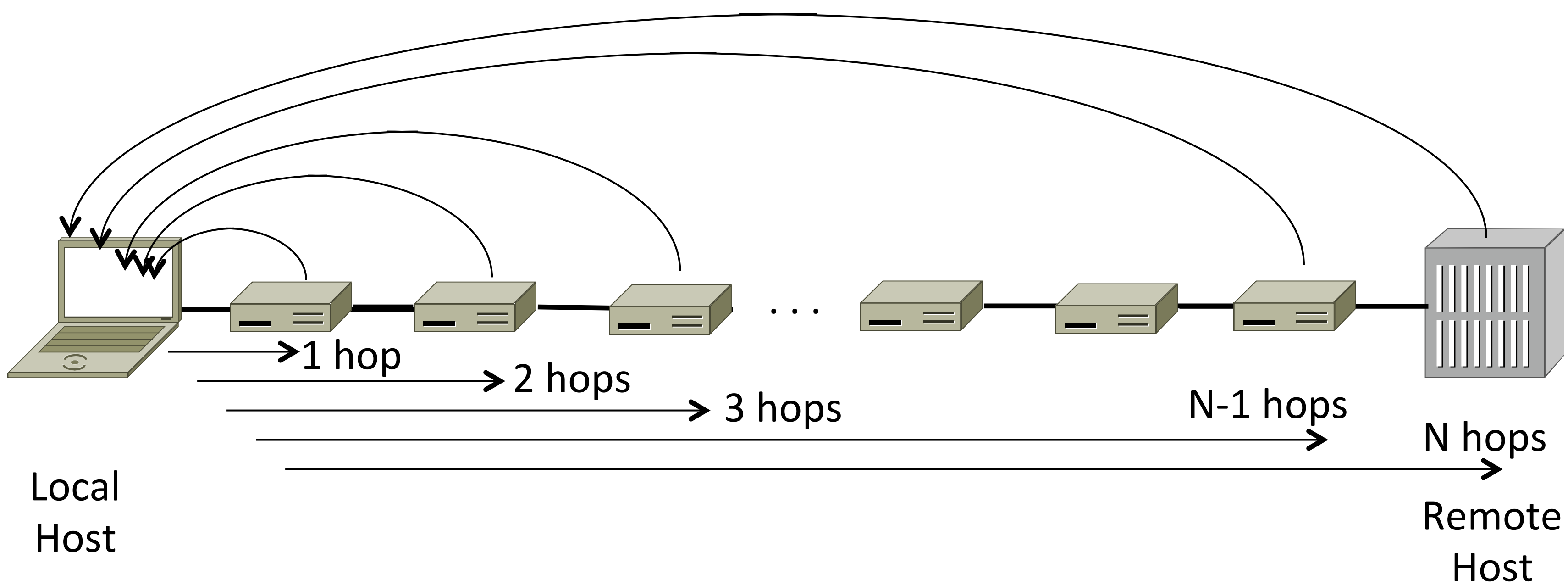
- Widely used command-line tool to let hosts peek inside the network
  - On all OSes (tracert on Windows)
  - Developed by Van Jacobson ~1987
  - Uses a network-network interface (IP) in ways we will explain later

Traceroute (2)

- Probes successive hops to find network path



Traceroute (3)



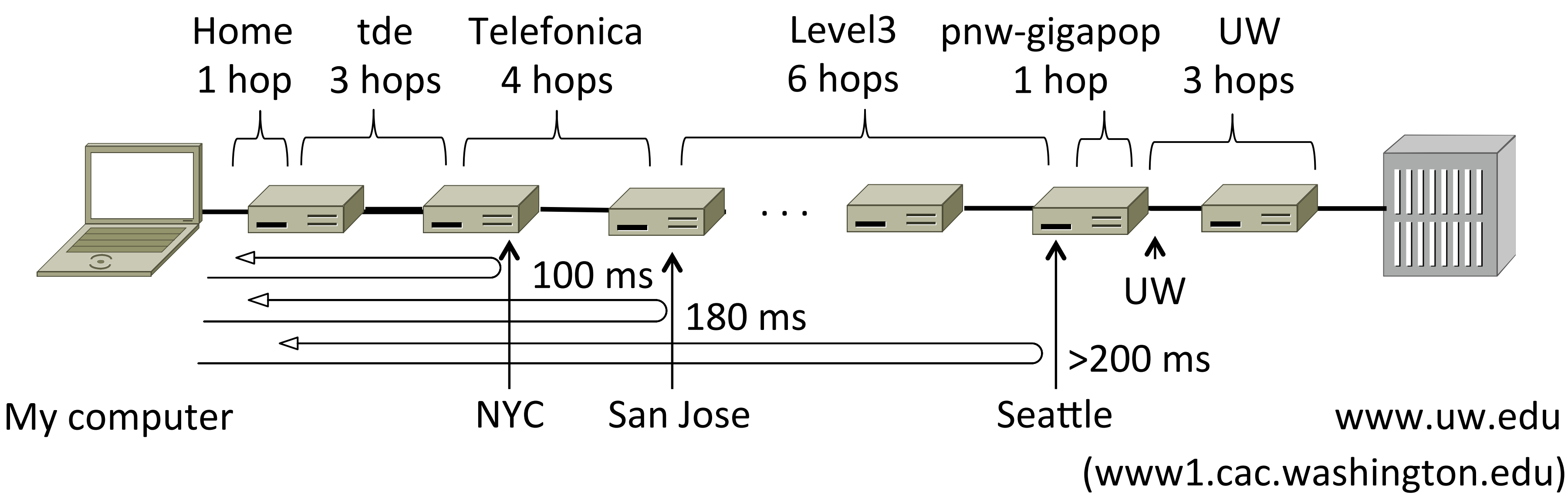
Using Traceroute

Administrator: Command Prompt

```
C:\Users\djw>tracert www.uw.edu

Tracing route to www.washington.edu [128.95.155.134]
over a maximum of 30 hops:

  1     1 ms    <1 ms     2 ms  192.168.1.1
  2     8 ms     8 ms     9 ms  88.Red-80-58-67.staticIP.rima-tde.net [80.58.67.88]
  3    16 ms     5 ms    11 ms  169.Red-80-58-78.staticIP.rima-tde.net [80.58.78.169]
  4    12 ms    12 ms    13 ms  217.Red-80-58-87.staticIP.rima-tde.net [80.58.87.217]
  5     5 ms    11 ms     6 ms  et-1-0-0-1-101-GRTBCNES1.red.telefonica-wholesale.net [94.142.103.20
5]
  6    40 ms    38 ms    38 ms  176.52.250.226
  7   108 ms   106 ms   136 ms  xe-6-0-2-0-grtnycpt2.red.telefonica-wholesale.net [213.140.43.9]
  8   180 ms   179 ms   182 ms  Xe9-2-0-0-grtpaopx2.red.telefonica-wholesale.net [94.142.118.178]
  9   178 ms   175 ms   176 ms  te-4-2.car1.SanJose2.Level3.net [4.59.0.225]
 10   190 ms   186 ms   187 ms  vlan80.csw3.SanJose1.Level3.net [4.69.152.190]
 11   185 ms   185 ms   187 ms  ae-82-82.ebr2.SanJose1.Level3.net [4.69.153.25]
 12   268 ms   205 ms   207 ms  ae-7-7.ebr1.Seattle1.Level3.net [4.69.132.50]
 13   334 ms   202 ms   195 ms  ae-12-51.car2.Seattle1.Level3.net [4.69.147.132]
 14   195 ms   196 ms   195 ms  PACIFIC-NOR.car2.Seattle1.Level3.net [4.53.146.142]
 15   197 ms   195 ms   196 ms  ae0--4000.iccr-sttlwa01-02.infra.pnw-gigapop.net [209.124.188.132]
 16   196 ms   196 ms   195 ms  v14000.uwbr-ads-01.infra.washington.edu [209.124.188.133]
 17     *        *        *     Request timed out.
 18   201 ms   194 ms   196 ms  ae4--583.uwar-ads-1.infra.washington.edu [128.95.155.131]
 19   197 ms   196 ms   195 ms  www1.cac.washington.edu [128.95.155.134]

Trace complete.
```

Using Traceroute (2)

- ISP names and places are educated guesses



Traceroute to another commercial webserver

```
-bash-3.1$ traceroute www.nyse.com
traceroute to www.nyse.com (209.124.184.150), 30 hops max, 40 byte packets
 1  acar-hsh-01-vlan75.cac.washington.edu (128.208.2.100)  0.327 ms  0.353 ms  0.392 ms
 2  uwcr-hsh-01-vlan3904.cac.washington.edu (205.175.110.17)  0.374 ms  0.412 ms  0.443 ms
 3  uwcr-hsh-01-vlan1901.cac.washington.edu (205.175.103.5)  0.595 ms  0.628 ms  0.659 ms
 4  uwbr-ads-01-vlan1902.cac.washington.edu (205.175.103.10)  0.445 ms  0.472 ms  0.501 ms
 5  ccar1-ads-ge-0-0-0-0.pnw-gigapop.net (209.124.176.32)  0.679 ms  0.747 ms  0.775 ms
 6  a209.124.184.150.deploy.akamaitechnologies.com.184.124.209.in-addr.arpa (209.124.184.150)  0.621 ms  0.456 ms  0.419 ms
```

What is going on?

```
-bash-3.1$ nslookup www.nyse.com
Name:   a789.g.akamai.net
Address: 209.124.184.137
```

Introduction to Computer Networks

The Socket API

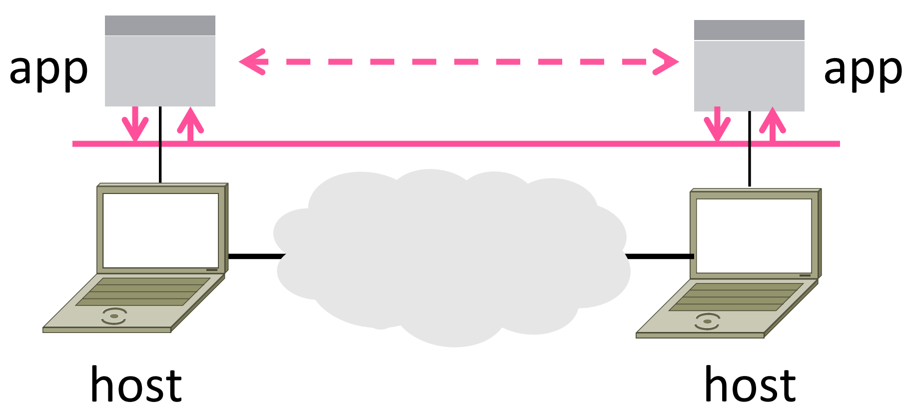
(§1.3.4, 6.1.2-6.1.4)

Computer Science & Engineering

UNIVERSITY of WASHINGTON

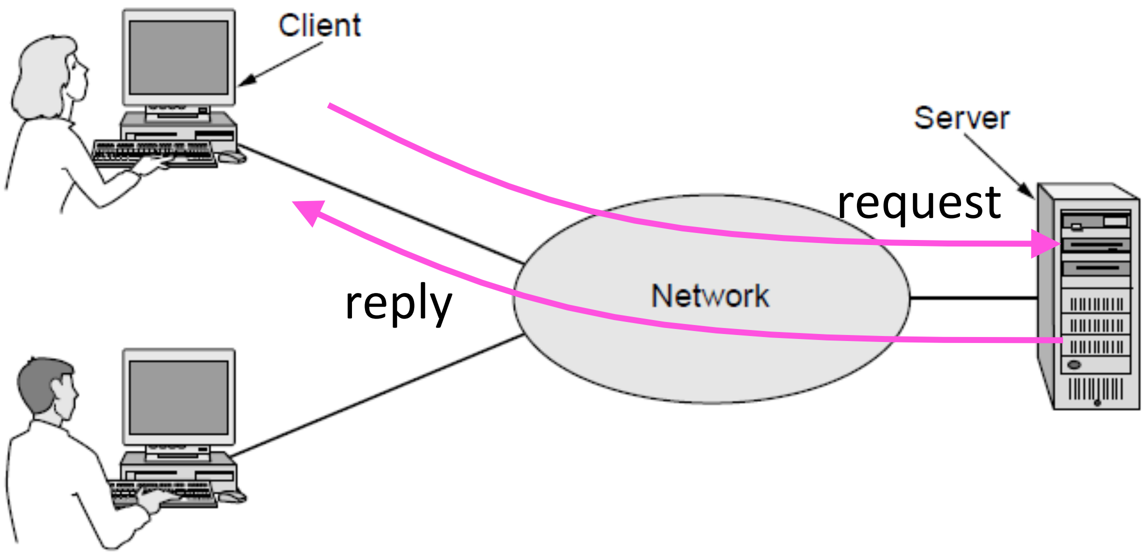
Network-Application Interface

- Defines how apps use the network
  - Lets apps talk to each other via hosts; hides the details of the network



Motivating Application

- Simple client-server setup



Motivating Application (2)

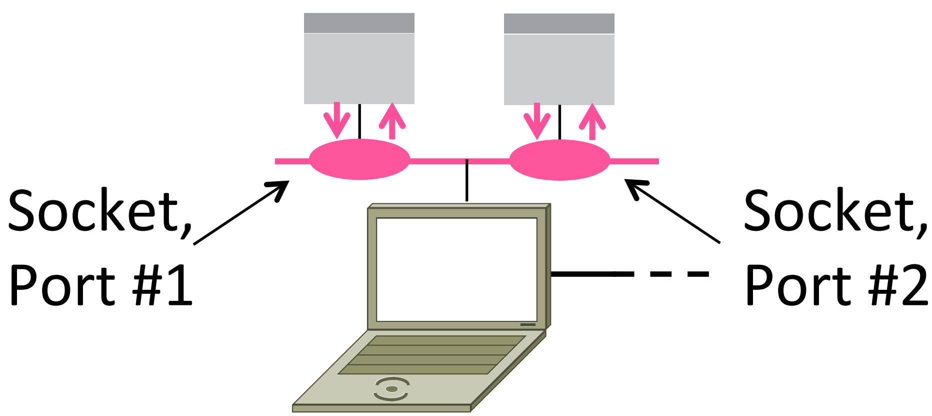
- Simple client-server setup
  - Client app sends a request to server app
  - Server app returns a (longer) reply
- This is the basis for many apps!
  - File transfer: send name, get file (§6.1.4)
  - Web browsing: send URL, get page
  - Echo: send message, get it back
- Let's see how to write this app …

Socket API

- Simple abstraction to use the network
  - The network service API used to write all Internet applications
  - Part of all major OSes and languages; originally Berkeley (Unix) ~1983
- Supports two kinds of network services
  - Streams: reliably send a stream of bytes »
  - Datagrams: unreliably send separate messages. (Ignore for now.)
  - Question: when would you use streams vs. datagrams?

Socket API (2)

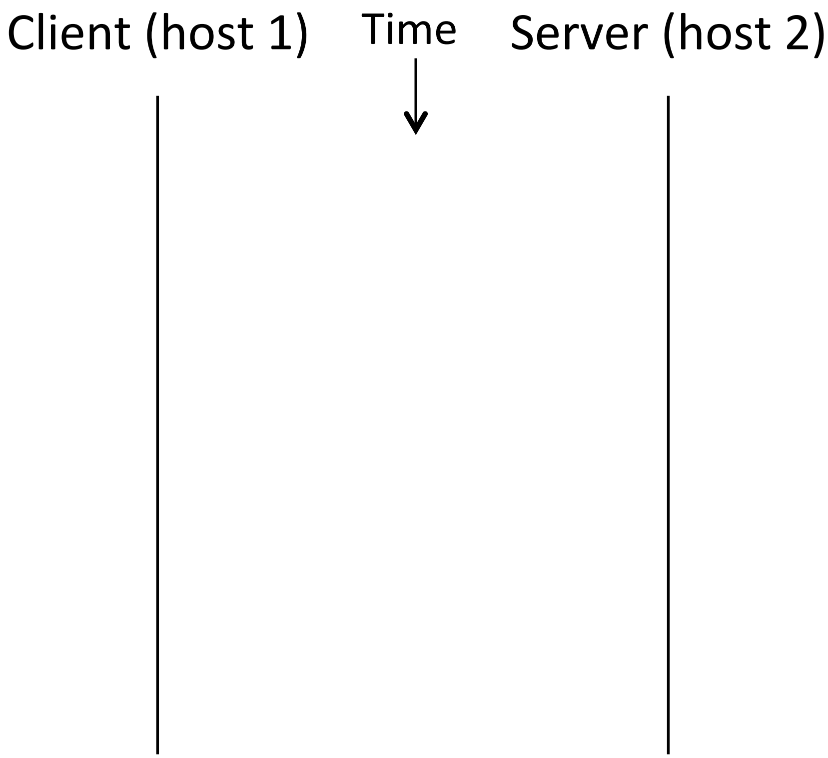
- <u>Sockets</u> let apps attach to the local network at different <u>ports</u>



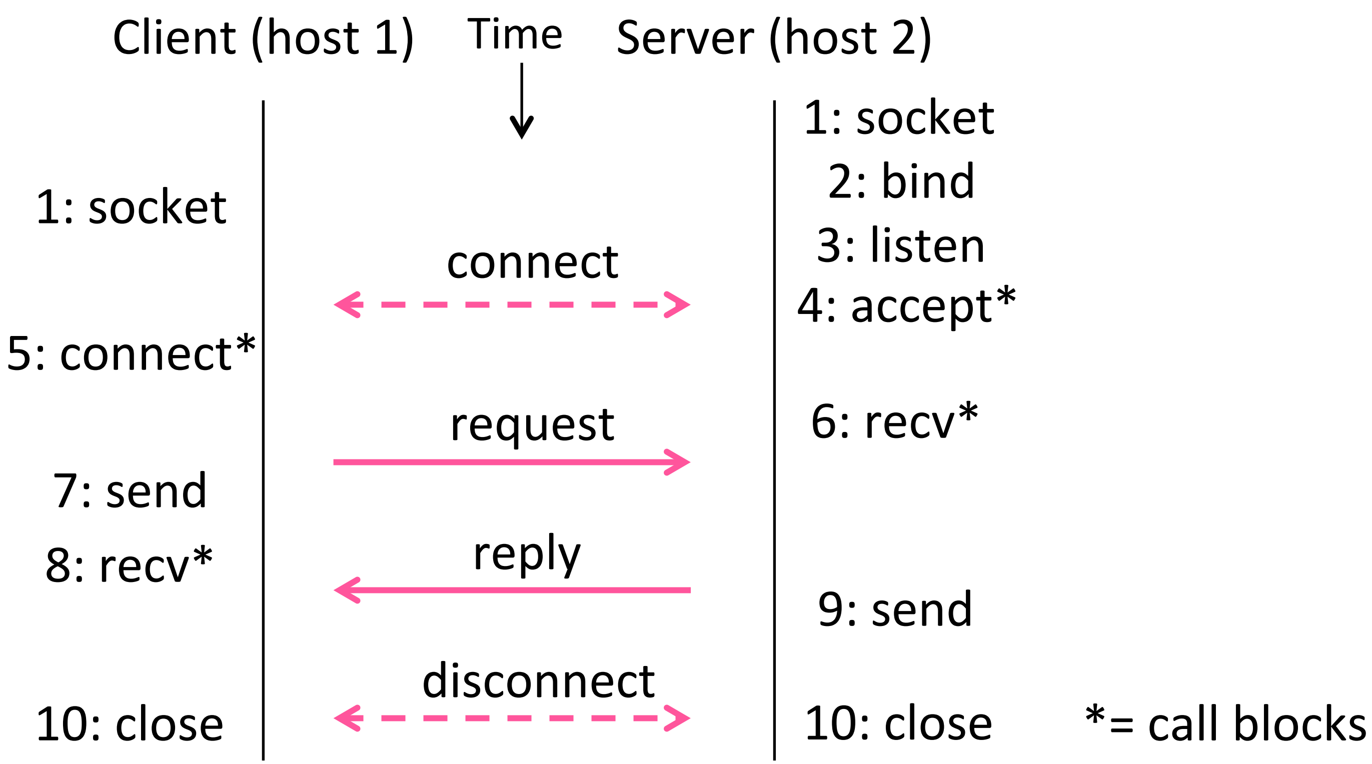
Socket API (3)

Primitive	Meaning
SOCKET	Create a new communication endpoint
BIND	Associate a local address with a socket
LISTEN	Announce willingness to accept connections; give queue size
ACCEPT	Passively establish an incoming connection
CONNECT	Actively attempt to establish a connection
SEND	Send some data over the connection
RECEIVE	Receive some data from the connection
CLOSE	Release the connection

Using Sockets



Using Sockets (2)



Client Program (outline)

```
socket()            // make socket
getaddrinfo()       // server and port name
                    // www.example.com:80
connect()           // connect to server [block]
…
send()              // send request
recv()              // await reply [block]
…                   // do something with data!
close()             // done, disconnect
```

Server Program (outline)

```
socket()            // make socket
getaddrinfo()       // for port on this host
bind()              // associate port with socket
listen()            // prepare to accept connections
accept()            // wait for a connection [block]
…
recv()              // wait for request
…
send()              // send the reply
close()             // eventually disconnect
```

Sample Code

- Python client code:
- http://courses.cs.washington.edu/courses/cse461/15sp/scripts/client_demo.py
- Python server non-threaded code:
- http://courses.cs.washington.edu/courses/cse461/15sp/scripts/server_demo_no_threads.py
- Python server threaded code:
- http://courses.cs.washington.edu/courses/cse461/15sp/scripts/server_demo_with_threads_for_processing.py

Introduction to Computer Networks

Protocols and Layering (§1.3)

Computer Science & Engineering

UNIVERSITY of WASHINGTON

Networks Need Modularity

- The network does much for apps:
  - Make and break connections
  - Find a path through the network
  - Transfers information reliably
  - Transfers arbitrary length information
  - Send as fast as the network allows
  - Shares bandwidth among users
  - Secures information in transit
  - Lets many new hosts be added
  - …

Networks Need Modularity

- The network does much for apps:
  - Make and break connections
  - ork
  -
  - rmation
  - ows
  - s
  - Secures information in transit
  - Lets many new hosts be added
  - …

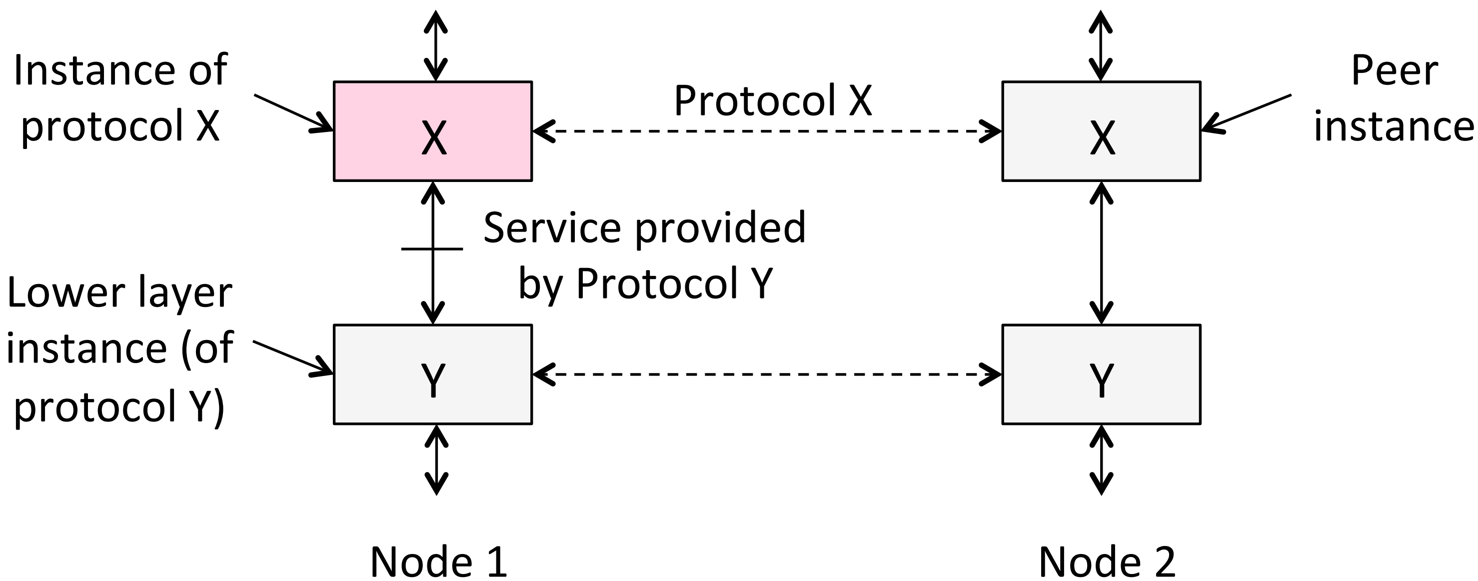
We need a form of modularity, to help manage complexity and support reuse

Protocols and Layers

- Protocols and layering is the main structuring method used to divide up network functionality
  - Each instance of a protocol talks virtually to its peer using the protocol
  - Each instance of a protocol uses only the services of the lower layer

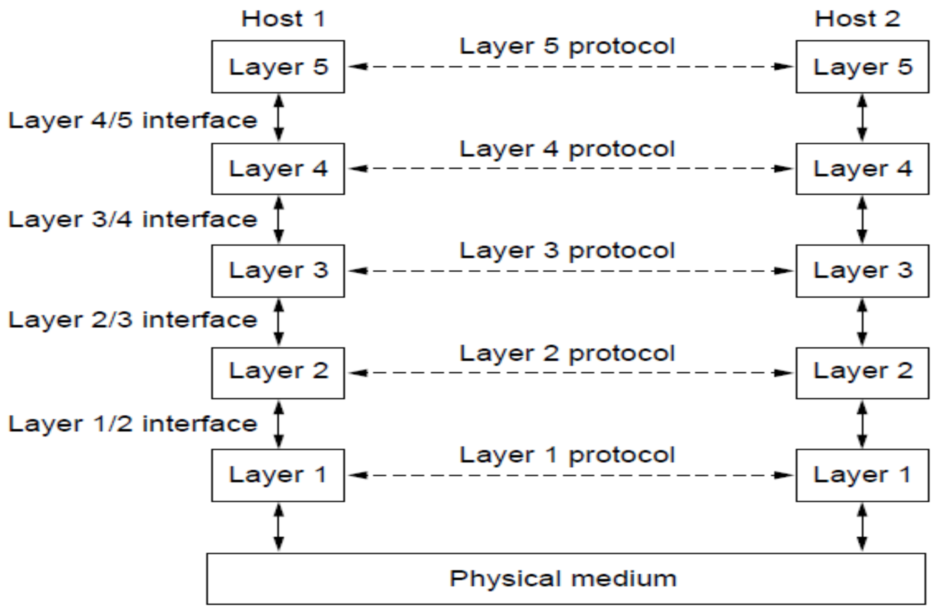
Protocols and Layers (2)

- Protocols are horizontal, layers are vertical



Protocols and Layers (3)

- Set of protocols in use is called a <u>protocol stack</u>



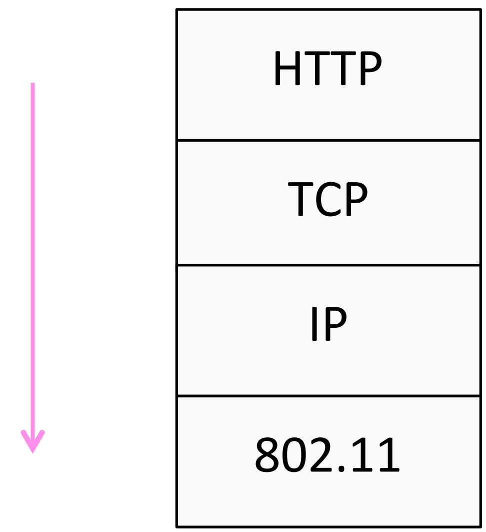
Protocols and Layers (4)

- Protocols you've probably heard of:
  - TCP, IP, 802.11, Ethernet, HTTP, SSL, DNS, … and many more
- An example protocol stack
  - Used by a web browser on a host that is wirelessly connected to the Internet

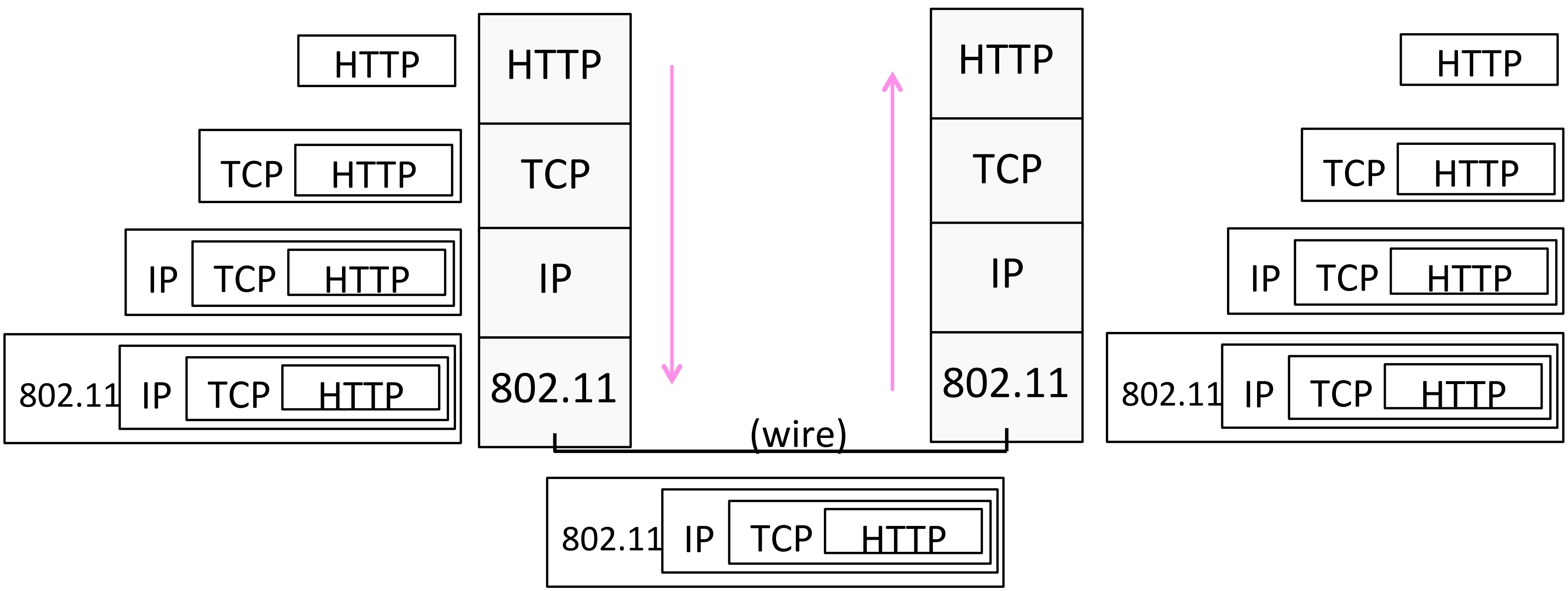
Encapsulation

- <u>Encapsulation</u> is the mechanism used to effect protocol layering
  - Lower layer wraps higher layer content, adding its own information to make a new message for delivery
  - Like sending a letter in an envelope; postal service doesn't look inside

Encapsulation (2)

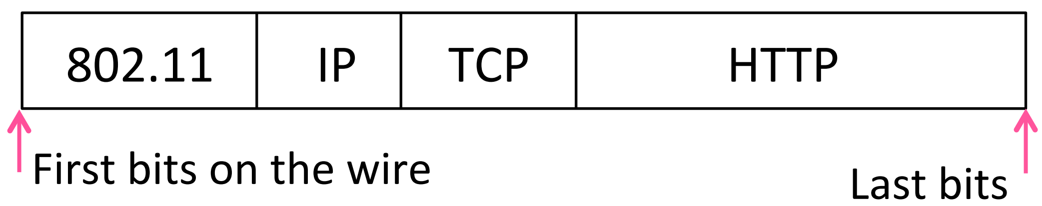


Encapsulation (3)



Encapsulation (4)

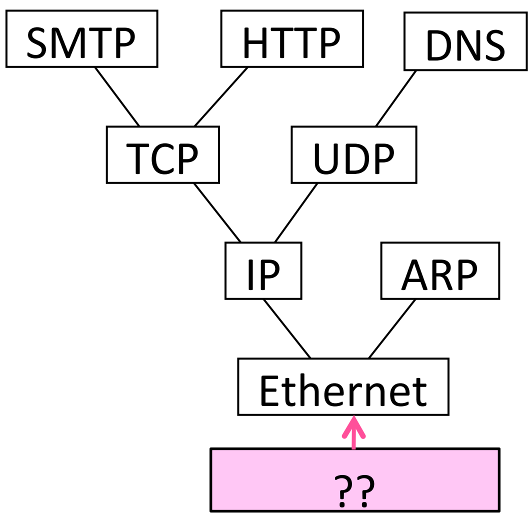
- Normally draw message like this:
  - Each layer adds its own header



- More involved in practice
  - Trailers as well as headers, encrypt/compress contents
  - Segmentation (divide long message) and reassembly

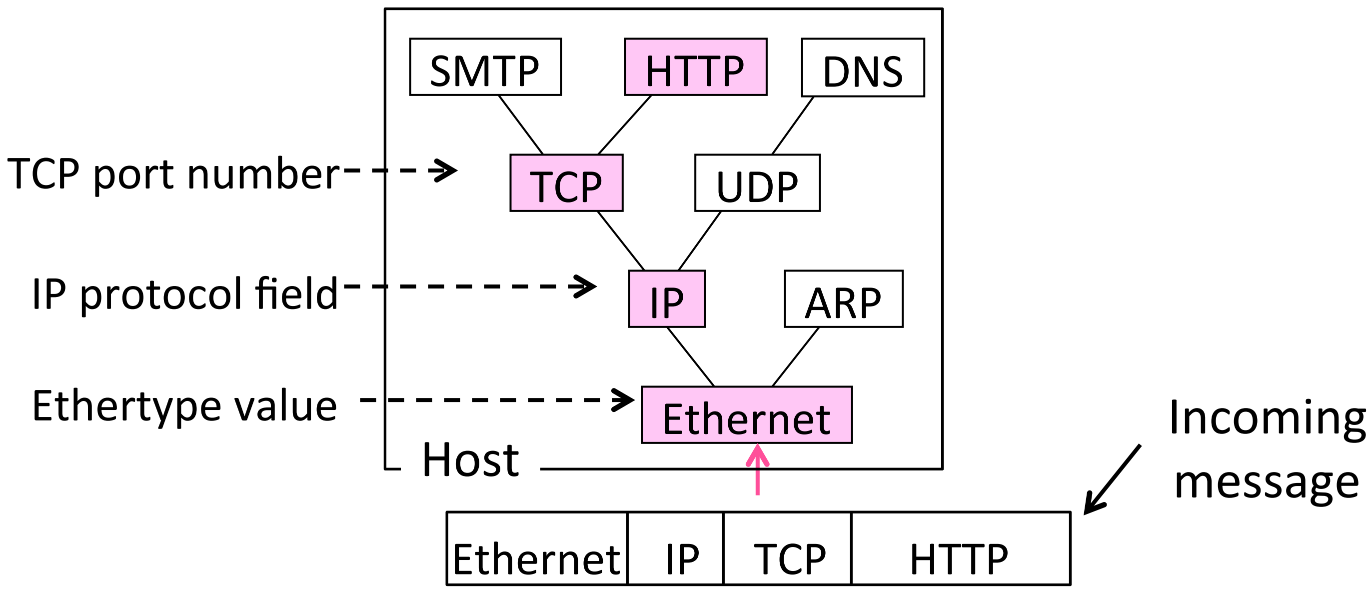
Demultiplexing

- Incoming message must be passed to the protocols that it uses



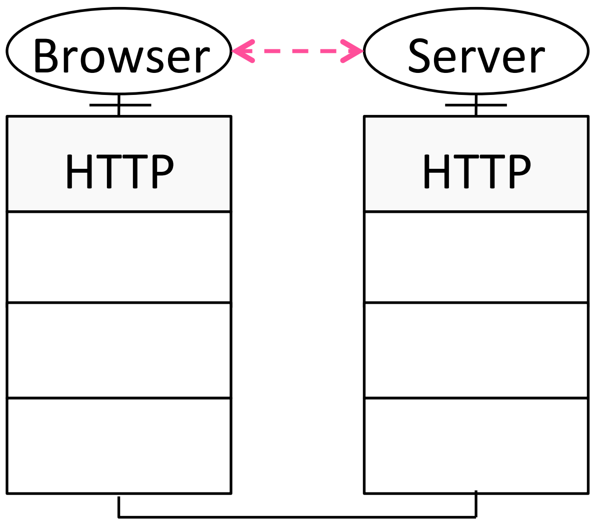
Demultiplexing (2)

- Done with demultiplexing keys in the headers

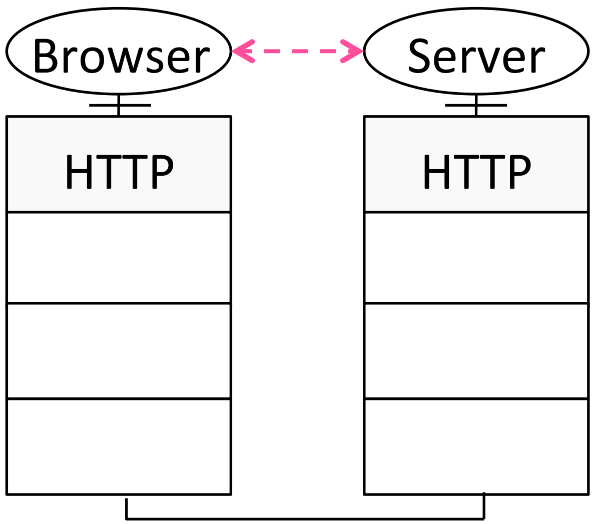


Advantage of Layering

- Information hiding and reuse



or



Advantage of Layering (2)

- Using information hiding to connect different systems

Disadvantage of Layering

- What are the undersirable aspects of layering?

Introduction to Computer Networks

Reference Models (§1.4, 1.6)

Computer Science & Engineering
UNIVERSITY of WASHINGTON

Guidance

- What functionality should we implement at which layer?
  - This is a key design question
  - Reference models provide frameworks that guide us »

OSI “7 layer” Reference Model

- A principled, international standard, to connect systems
  - Influential, but not used in practice. (Woops)

7	Application	– Provides functions needed by users
6	Presentation	– Converts different representations
5	Session	– Manages task dialogs
4	Transport	– Provides end-to-end delivery
3	Network	– Sends packets over multiple links
2	Data link	– Sends frames of information
1	Physical	– Sends bits as signals

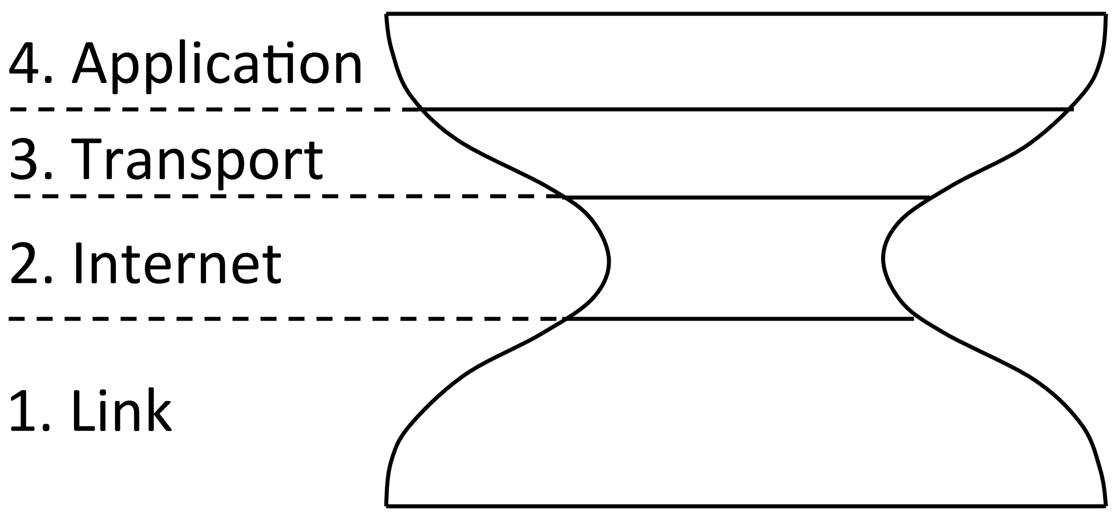
Internet Reference Model

- A four layer model based on experience; omits some OSI layers and uses the IP as the network layer.

4.	Application	– Programs that use network service
3.	Transport	– Provides end-to-end data delivery
2.	Internet	– Send packets over multiple networks
1.	Link	– Send frames over a link

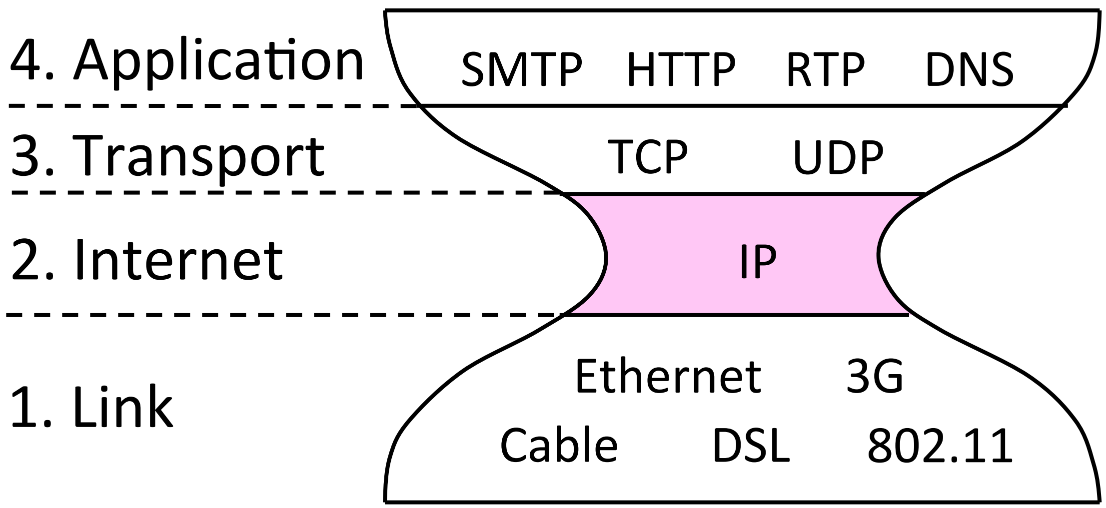
Internet Reference Model (2)

- With examples of common protocols in each layer



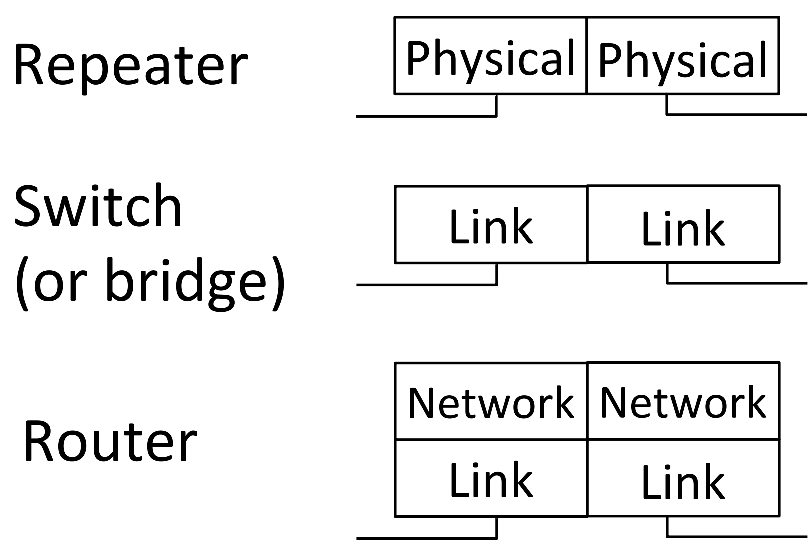
Internet Reference Model (3)

- IP is the "narrow waist" of the Internet
  - Supports many different links below and apps above



Layer-based Names (2)

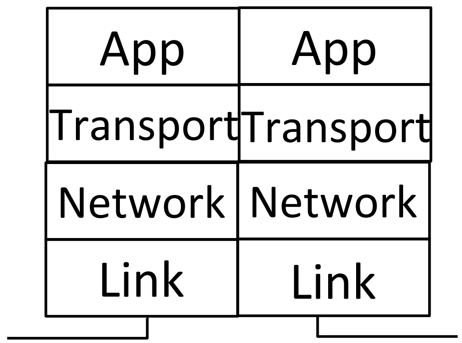
- For devices in the network:



Layer-based Names (3)

- For devices in the network:

Proxy or middlebox or gateway



But they all look like this!

A Note About Layers

- They are guidelines, not strict
  - May have multiple protocols working together in one layer
  - May be difficult to assign a specific protocol to a layer