

# Measurement, App Layer

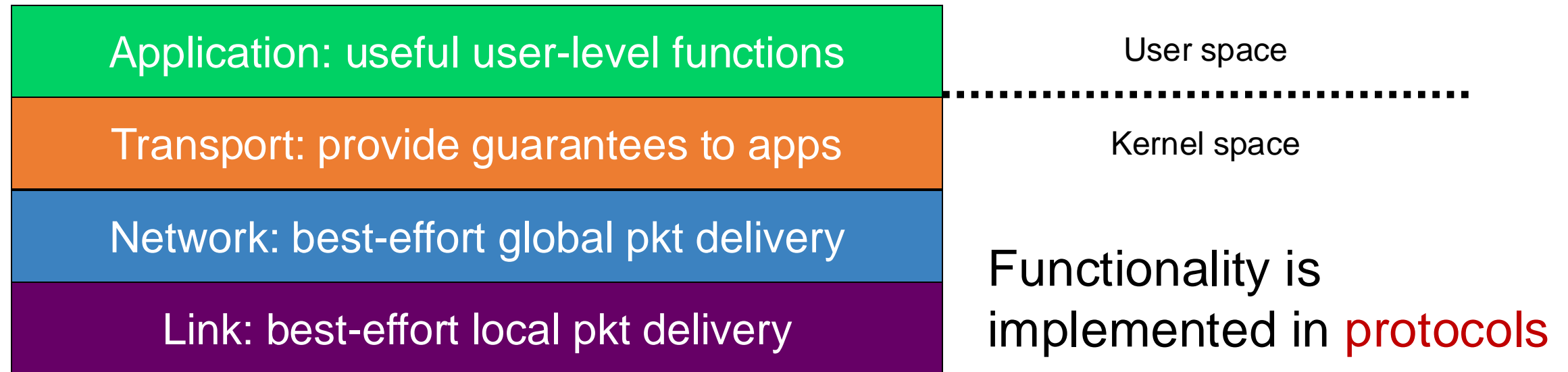
Lecture 3

<http://www.cs.rutgers.edu/~sn624/352-F24>

Srinivas Narayana

# Review

- Switching: Circuit, Message, Packet
- Layering: Modularity



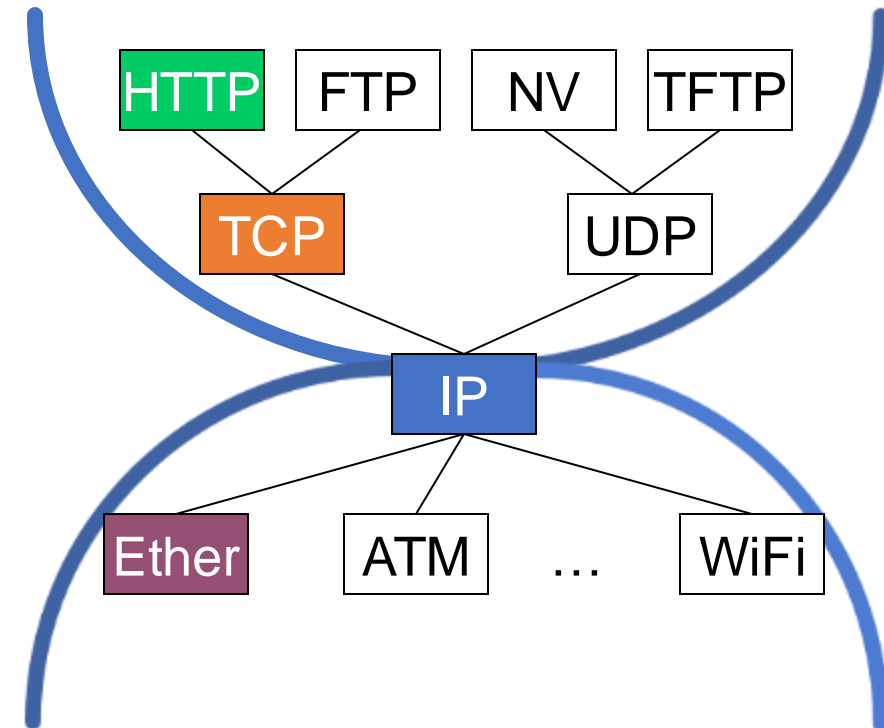
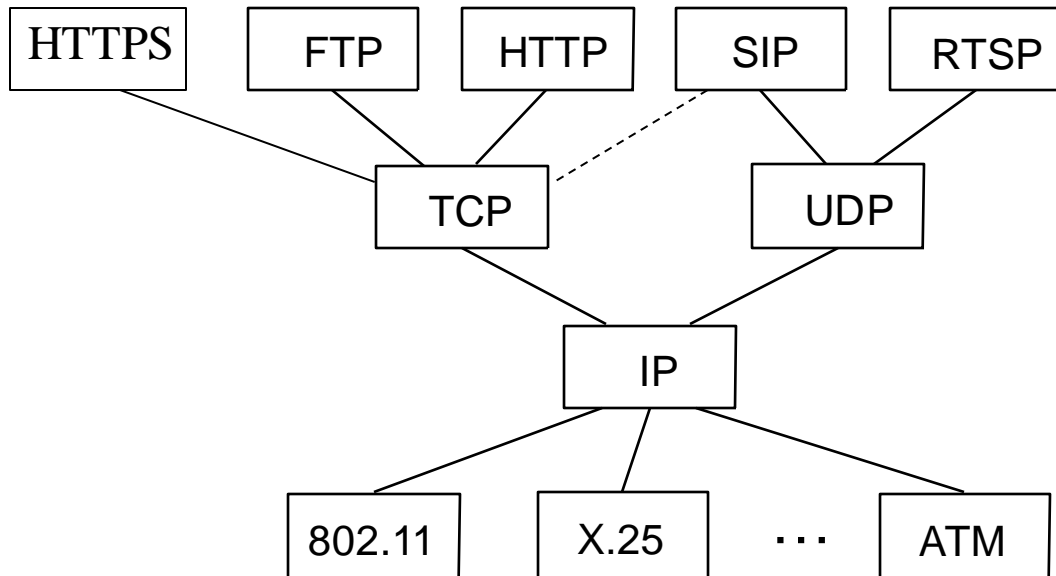
# Protocols: The “rules” of networking

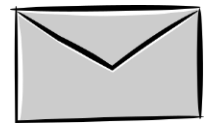
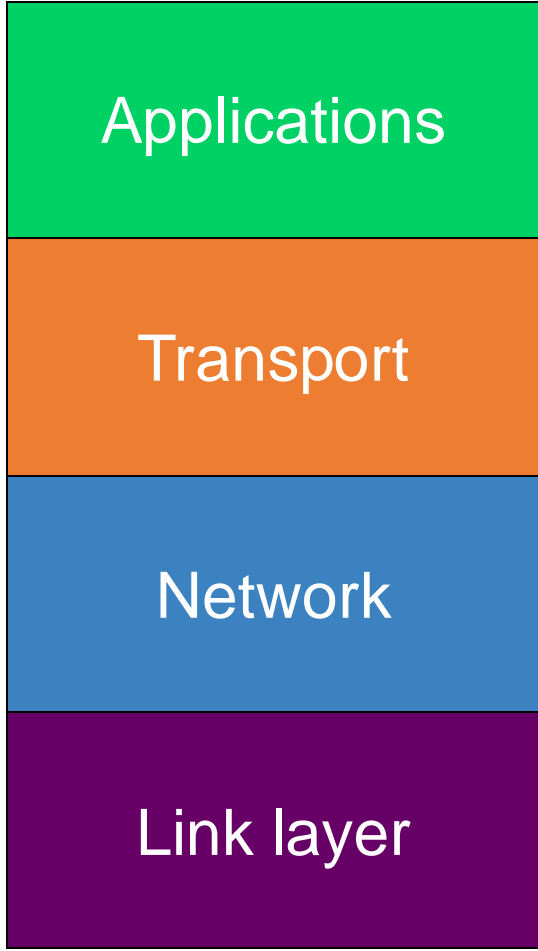
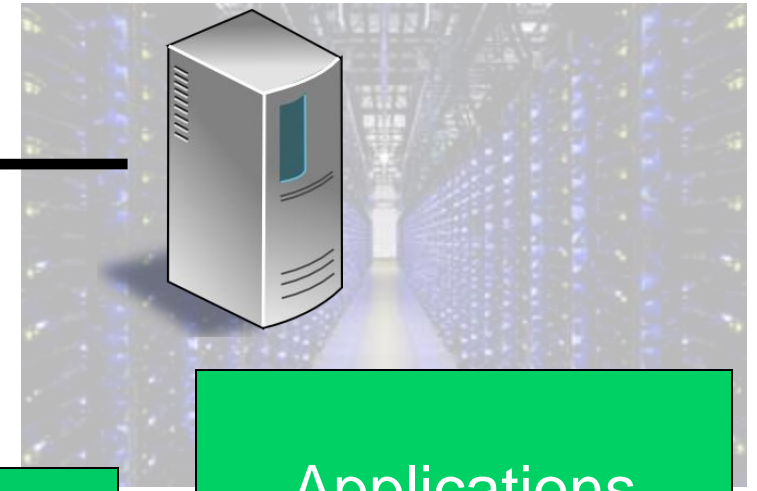
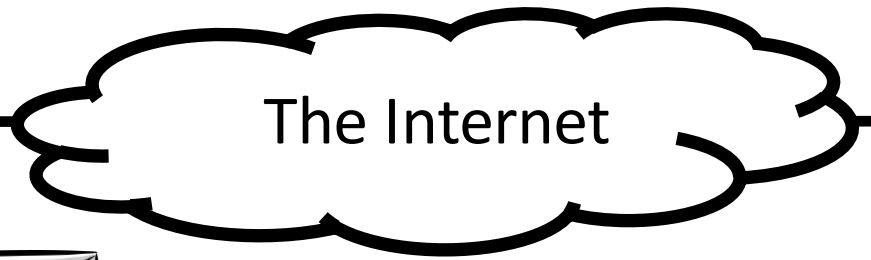
- Protocols consist of two things
- **Message format**
  - structure of messages exchanged with an endpoint
- **Actions**
  - operations upon receiving, or not receiving, messages
- Example of a Zoom conversation:
  - Message format: English words and sentences
  - Actions: when a word is heard, say “yes”; when nothing is heard for more than 3 seconds, say “can you hear me?”

# The protocols of the Internet

- Standardized by the Internet Engineering Task Force (IETF)
  - through documents called **RFCs** (“Request For Comments”)

- Layering of protocols

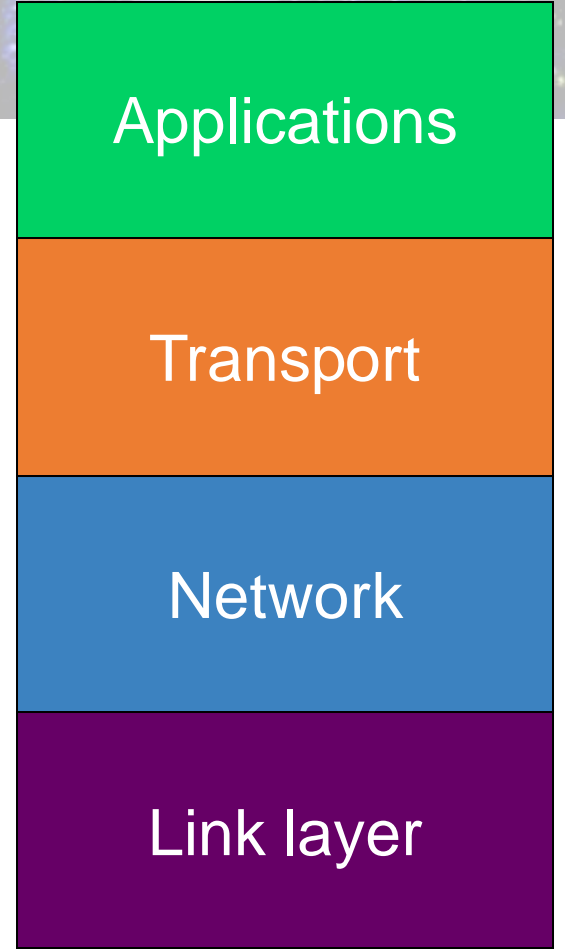
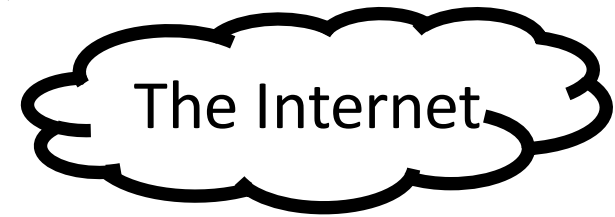


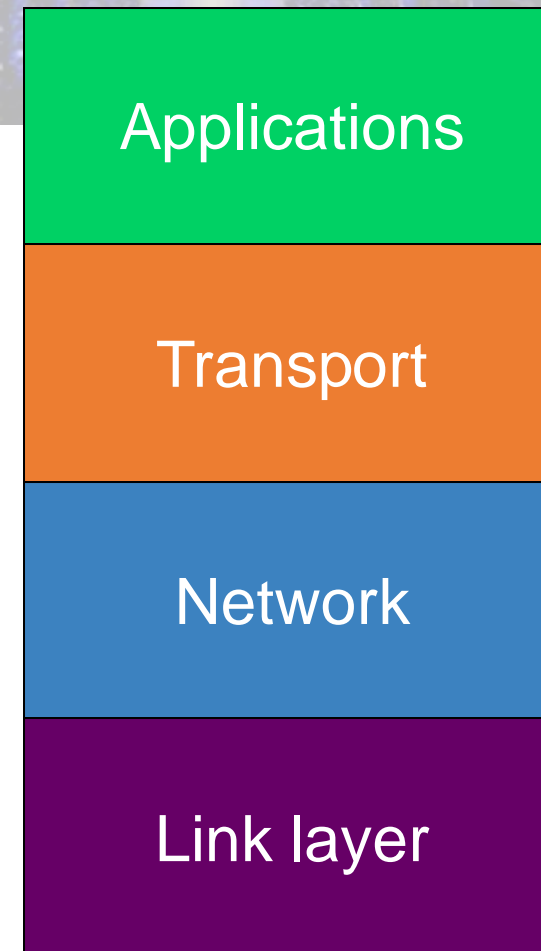
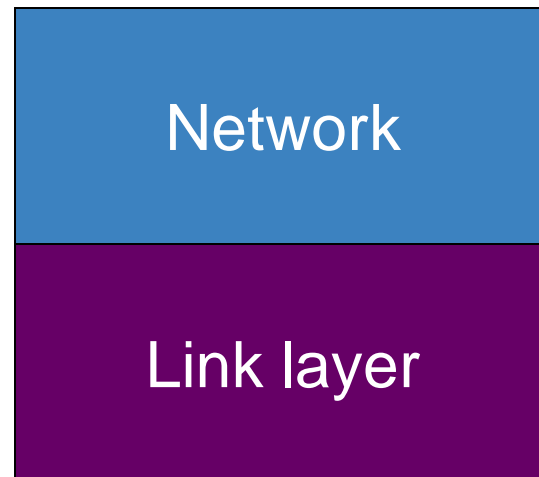
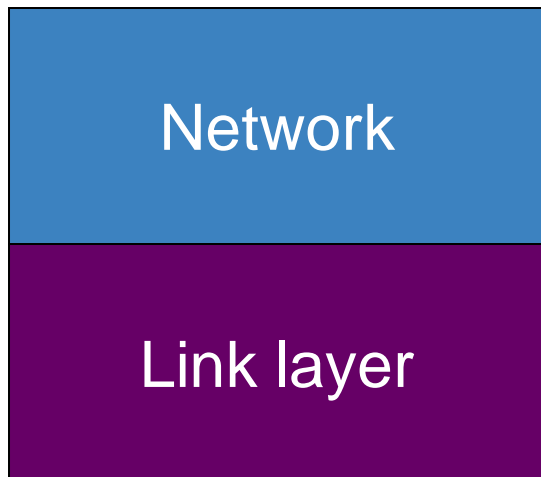
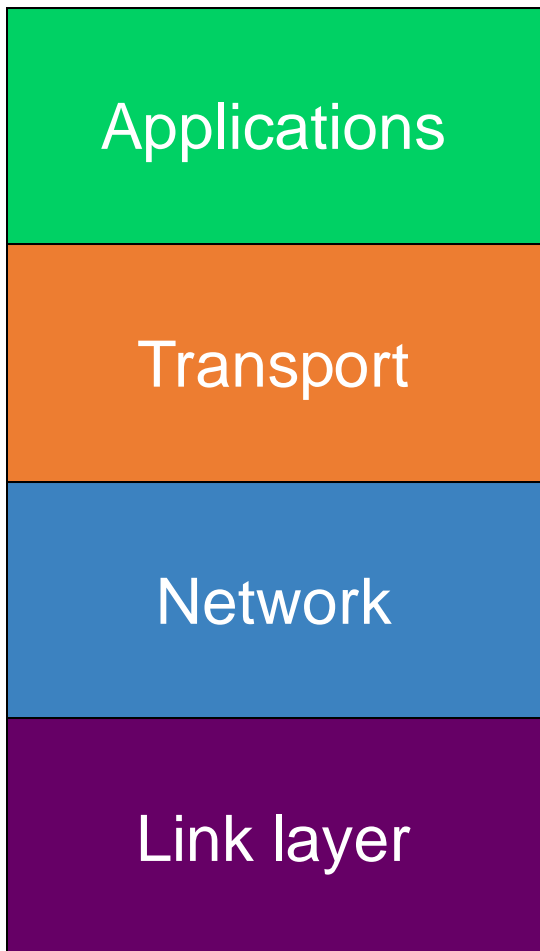
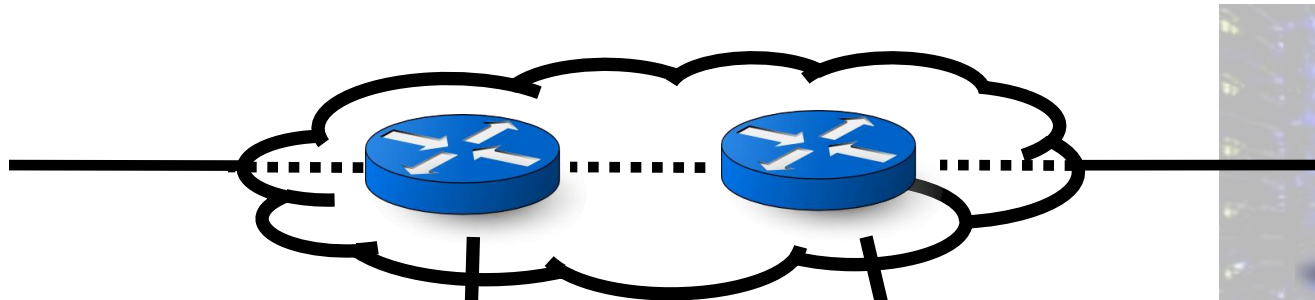


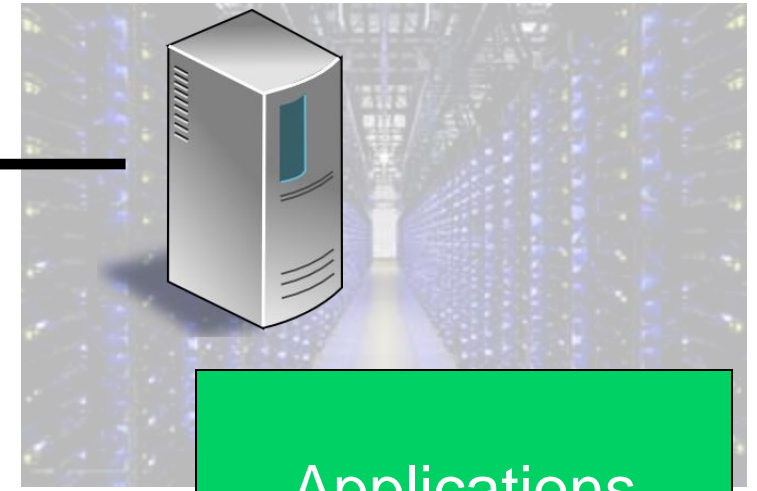
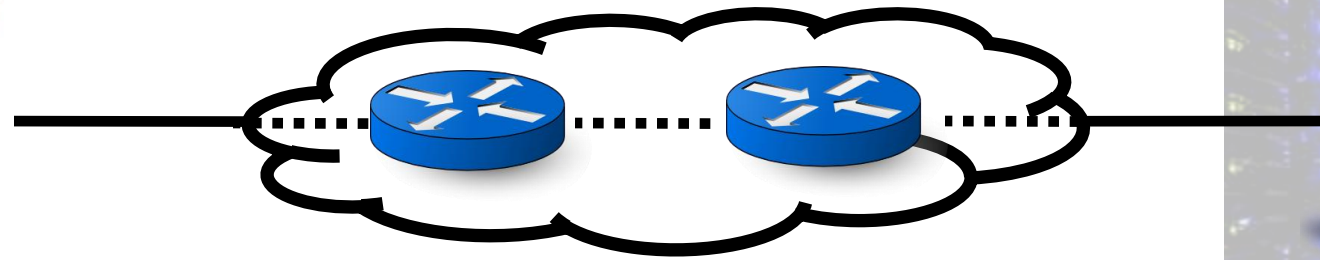
Packet starts as an app message



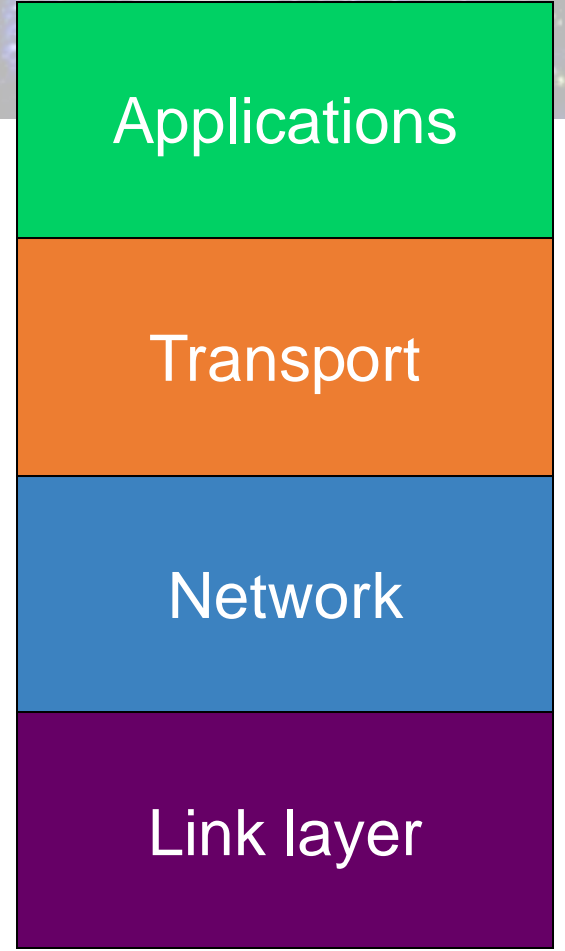
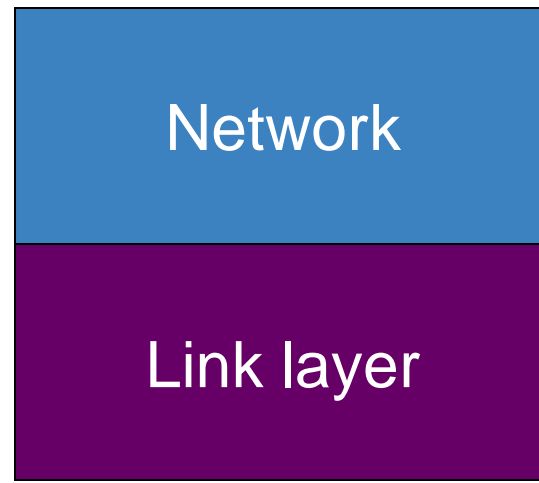
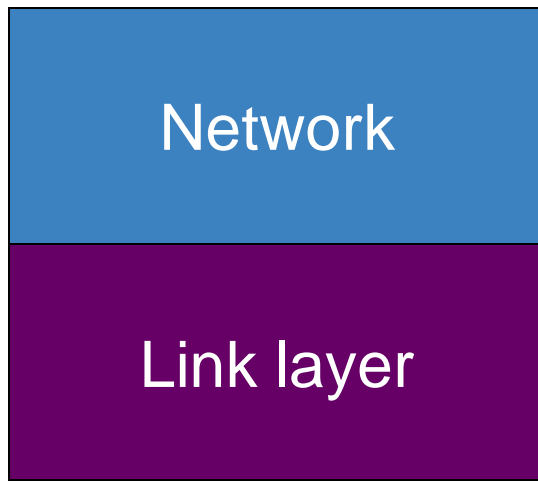
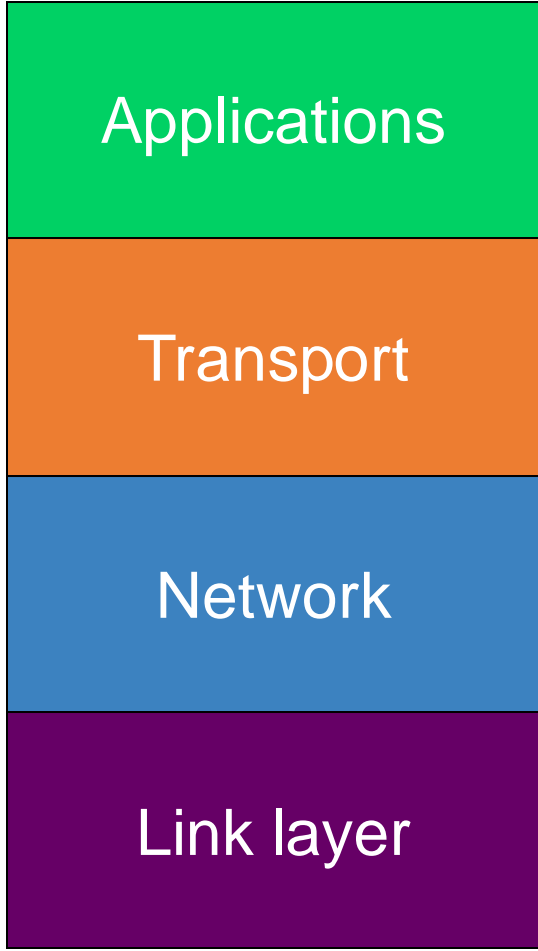
Packet takes on headers at each layer







Routers have network and link layers too!

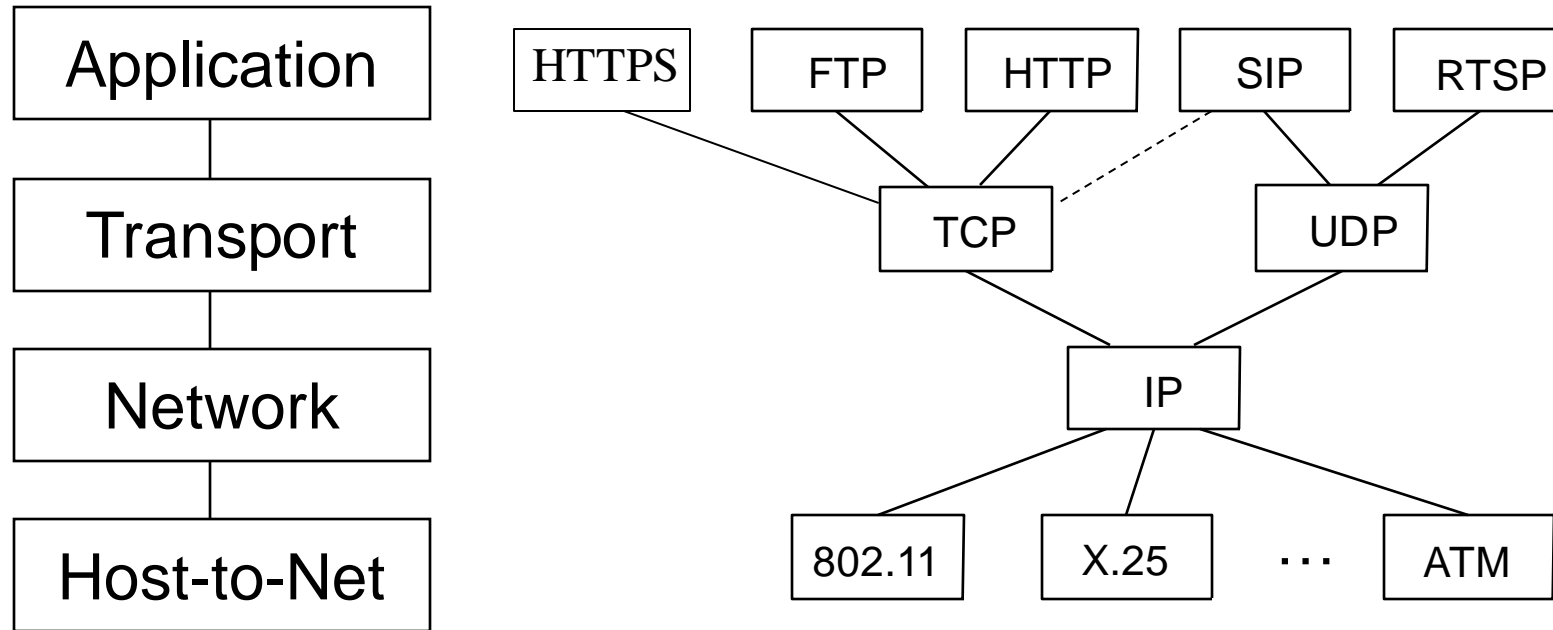


# Layering

- Communication over the Internet is a complex problem.
- Layering simplifies understanding, testing, maintaining
- Easy to improve or replace protocol at one layer without affecting others



# This course has layers



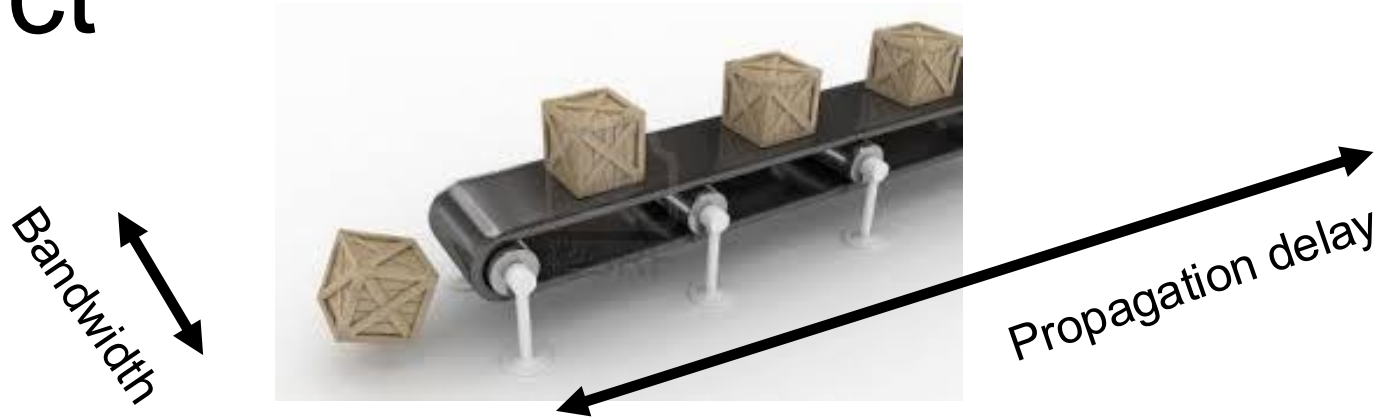
# Measuring the Internet

Speed, by any other name

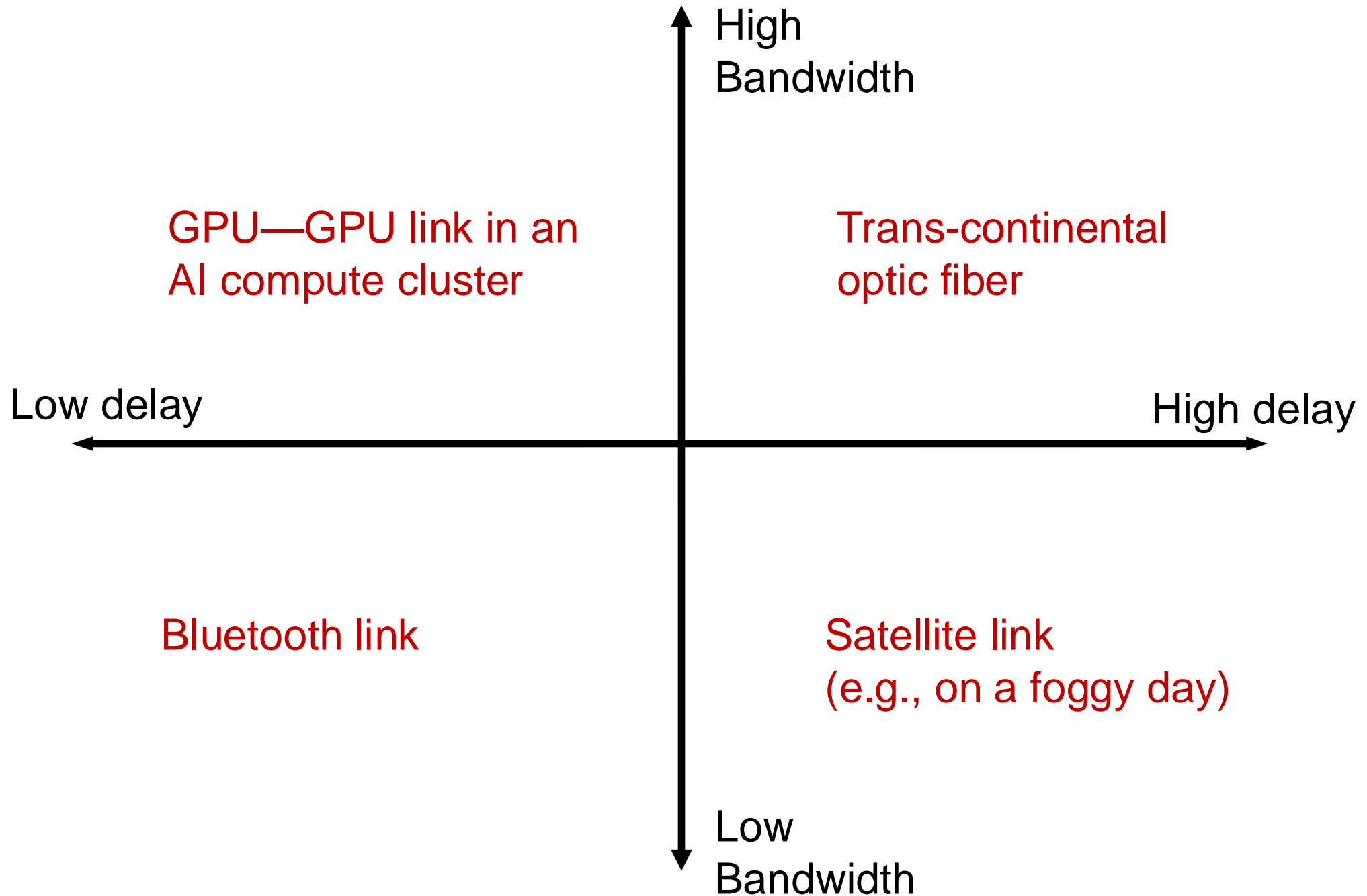
# What exactly do we mean by speed?

- A packet consists of many bits, including header and data
  - **Packet size**: length of the packet (bits or bytes) incl. header and data
- **Bandwidth**: For a single link, amount of data it can transmit per unit time (bits/second or Bytes/second or packets/second)
- **Total packet delay**: time from the first bit@sender to the last bit@receiver

# Bandwidth and delay are related but distinct



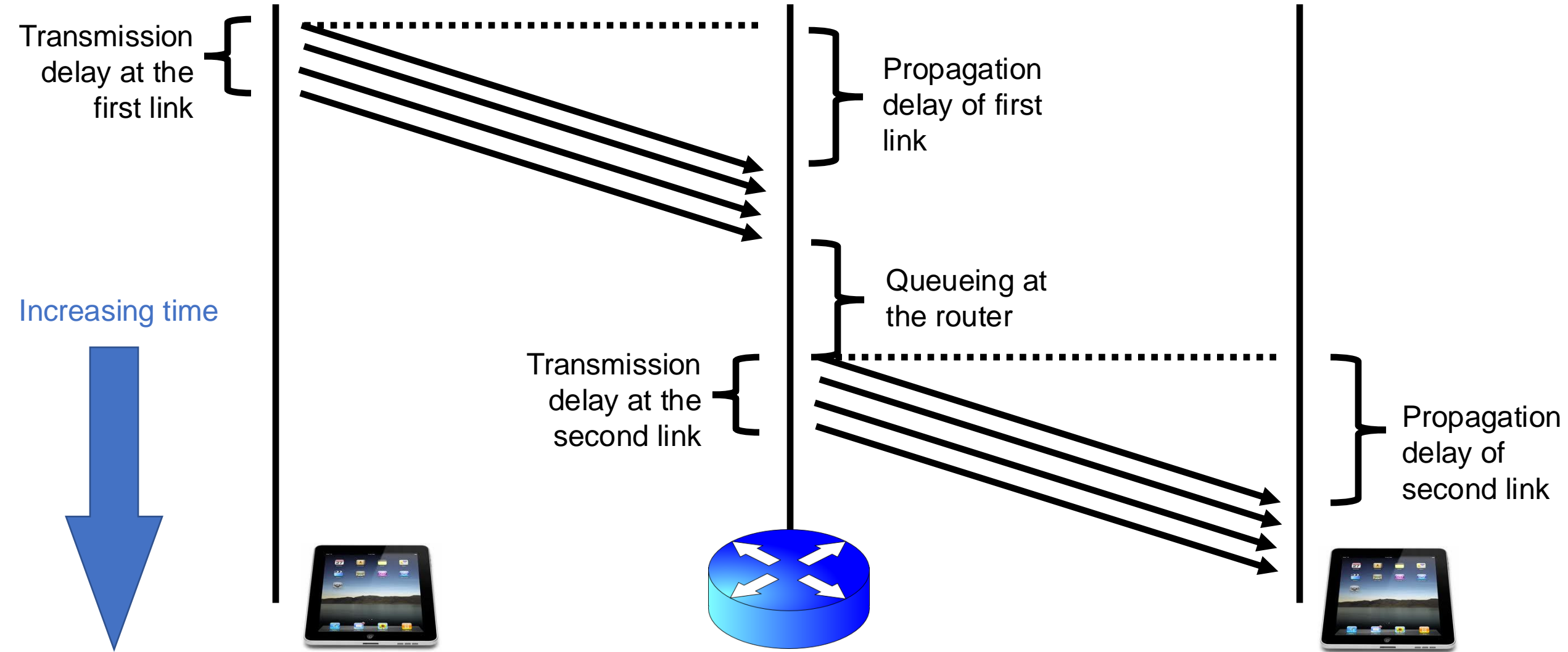
- Total packet delay = time for a box to travel the length of the belt
- Bandwidth = the number of boxes put on the belt per minute (“rate”)



# Total Packet Delay has a few pieces

- **Propagation delay:** Time needed to move one bit across (second)
  - Imposed by the communication medium; depends on the link “length”
- **Transmission delay:** Time from first bit@sender to last bit@sender
  - Determined by link bandwidth and packet size
  - Packet size / link bandwidth
- **Queueing delay:** Time that a packet waits for transmission
  - Determined by contention for the link
- **Total packet delay** = propagation delay + queueing delay + transmission delay for a single packet

# Visualizing the components of delay



# Bandwidth and delay demo

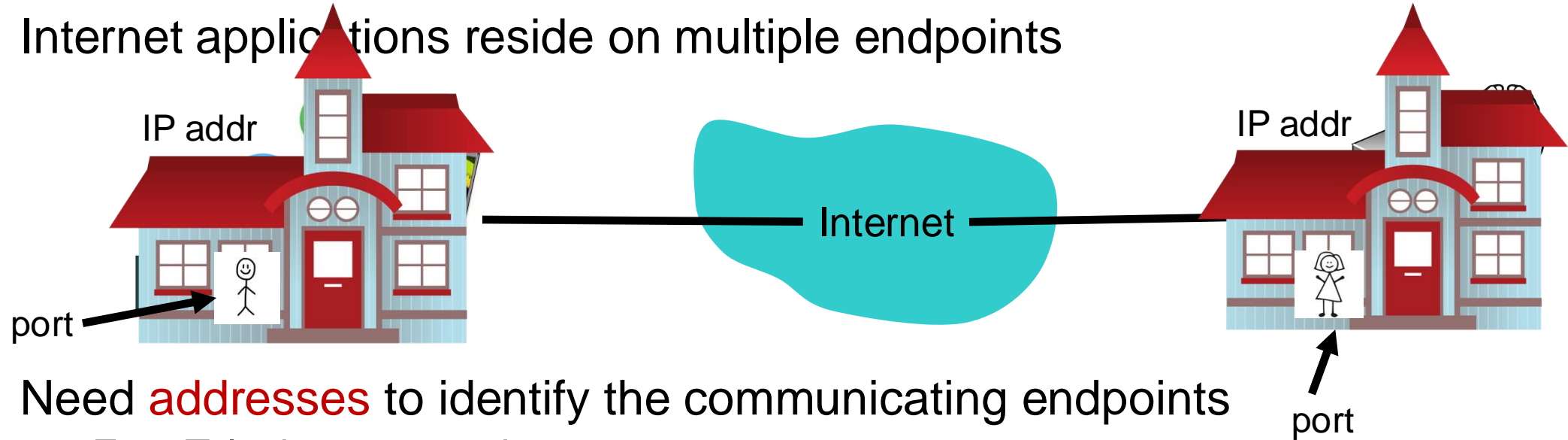
- Throughput (related to bandwidth)
  - `iperf -s #` at the destination
  - `iperf -c <destination> #` at the source,
  - e.g., `iperf -c localhost`
- (total) delay
  - `ping <destination>`
  - e.g., `ping google.com`
- (you can try it!)



Application Layer

# App-layer communication

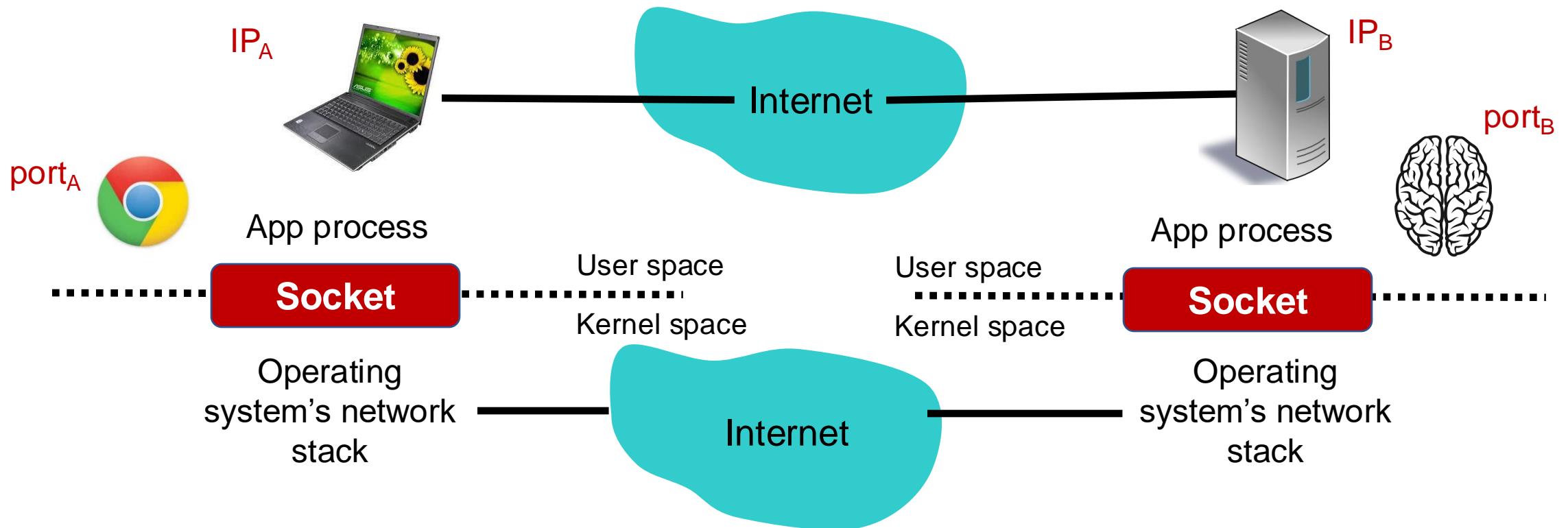
- Internet applications reside on multiple endpoints



- Need **addresses** to identify the communicating endpoints
  - E.g., Telephone network: xxx-yyy-zzzz
- Internet: **Internet Protocol (IP) addresses**
  - IPv4 (32 bits) 128.6.24.78
  - IPv6 (128 bits) 2001:4000:A000:C000:6000:B001:412A:8000
- Which app on each endpoint? **Port number**

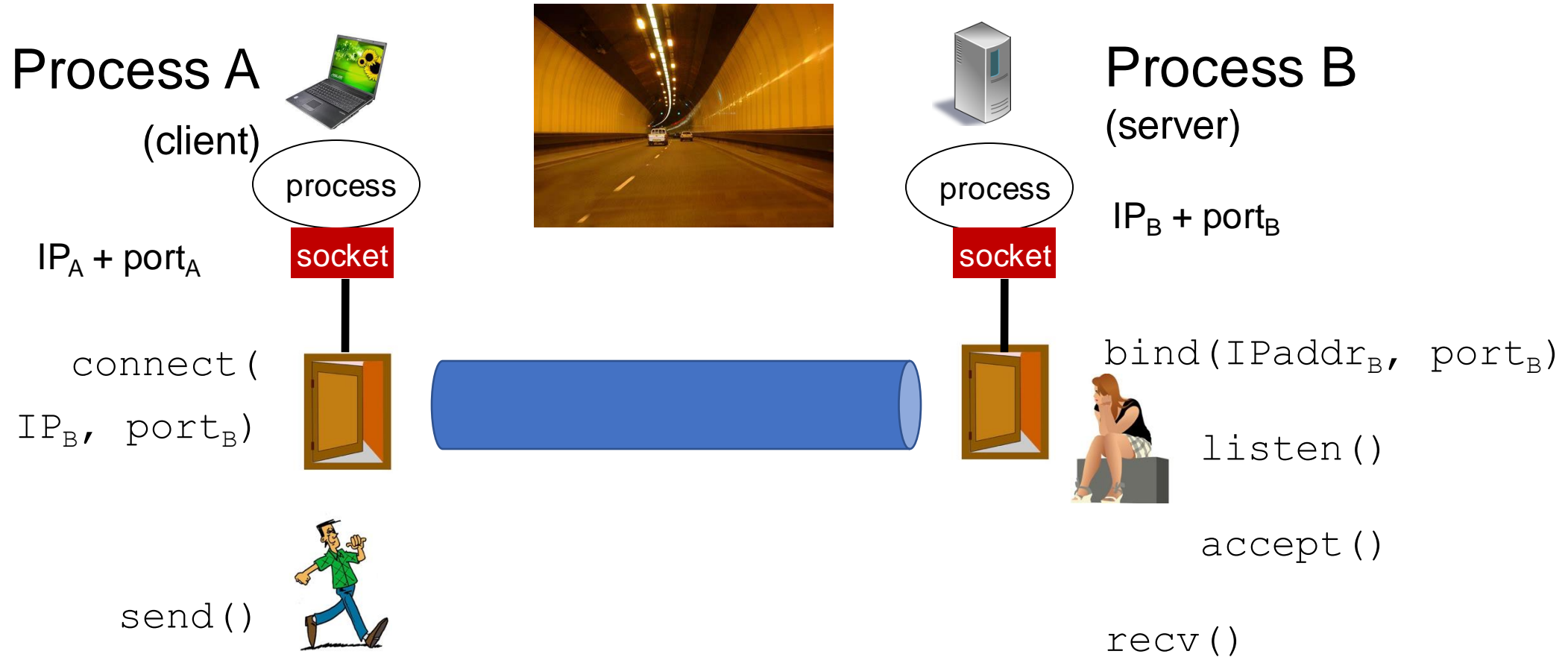
# How are addresses used?

- **Socket**: abstraction (API) of the Internet for applications



App-layer connection is a 4-tuple:  $(IP_A, port_A, IP_B, port_B)$

# Socket system calls

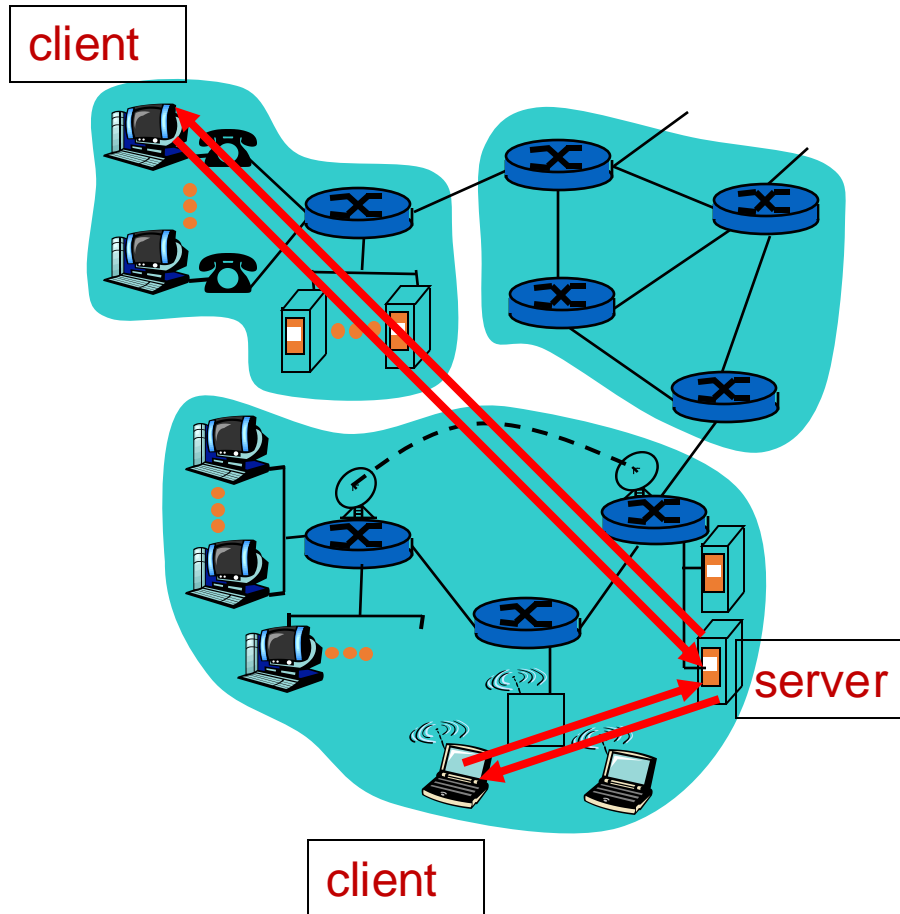


# Seeing app-layer connections

- `netstat`
- `ss`

# Common Architectures of Applications

# Client-server architecture



## Server:

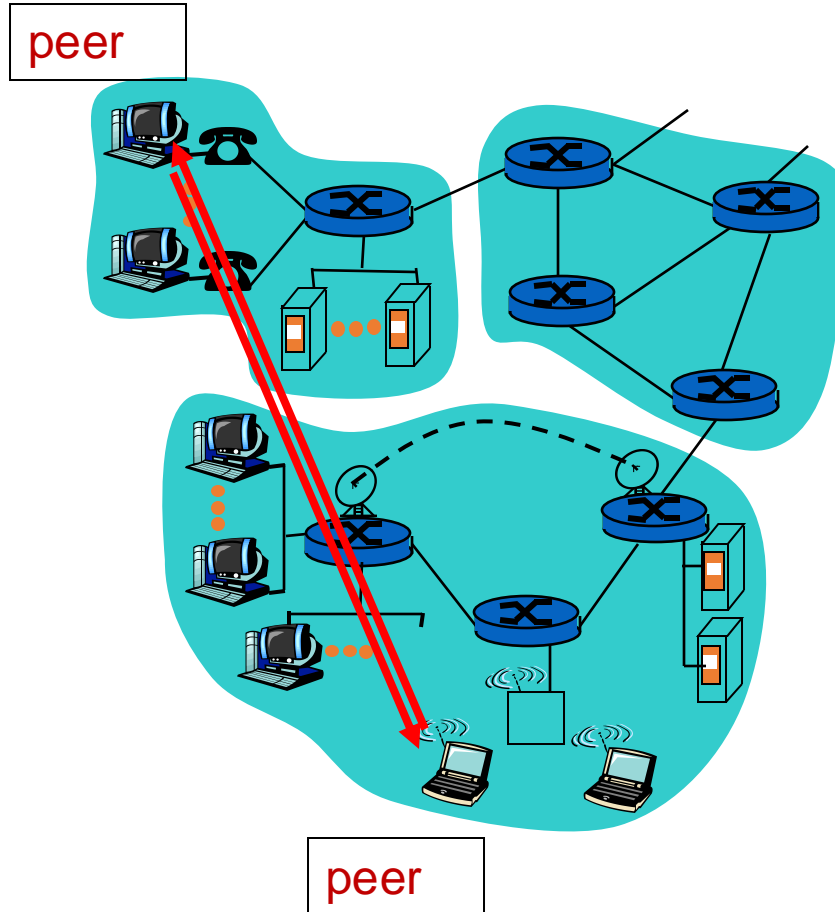
- Always-on endpoint
- Provides a “service” to the world
- Typically, a permanent IP address
- Compute clusters to scale to many users

## Clients:

- A “customer” of the server
- May be intermittently connected
- May have dynamic IP addresses
- Typically, do not communicate directly with other clients

- The web and most mobile apps use a client-server architecture

# Peer-to-peer (P2P) architecture



- **Peers:**
  - Intermittently connected hosts
  - Directly talking to each other
- Little to no reliance on always-up servers
  - Examples: BitTorrent
- Today, many applications use a **hybrid** model
  - Example: (webRTC) Google meet, Facebook messenger, ...



# Going forward: A few app-layer protocols

- Domain Name System
- The web
- Streaming video

# Domain Name System

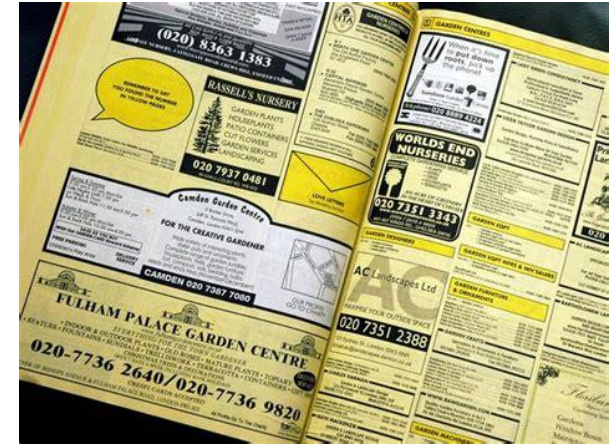
You have my name. Can you  
lookup my address?

# Domain Name System (DNS)

- Problem: We need an easier way to remember IP addresses
  - Average brain can easily remember 7 digits for a few names
  - On average, IP addresses have 12 digits
- Solution:
  - Use alphanumeric names to refer to hosts.
  - Called **host names** or **domain names** (e.g.: cs.rutgers.edu)
  - We need a **directory (address book)**
  - A service to map alphanumeric host names to binary IP addresses
  - We call this process **Address Resolution**

# Types of Directories

- Directories map a *name* to an *address*
- Simplistic designs
  - Central directory
  - Ask everyone (e.g., flooding)
  - Tell everyone (e.g., push to a file like /etc/hosts)
- Scalable distributed designs
  - Hierarchical namespace (e.g., Domain Name System (DNS))
  - Flat name space (e.g., Distributed Hash Table)



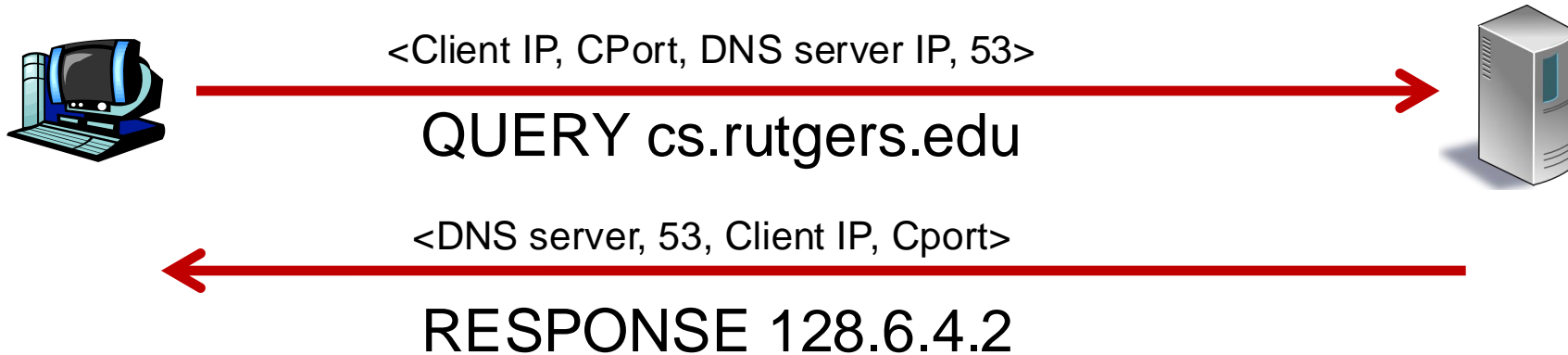
# Simple DNS

- What if every endpoint has a local directory?
- `/etc/hosts.txt`
  - How things worked in the early days of the Internet!
- What if endpoints changed addresses? How do you keep this up to date?

<b>nowski Maciej</b> Gorn Kryst- tr 11 610 41 <b>nowski Mieczyslaw</b> Lehens- mittel Hopfenstr 91 522 47 <b>nowski Mieczyslaw R.</b> Snausteyer 28 415 65 <b>nowski Stanislaw</b> Mechani- er Bahnhofstr 2 596 78 <b>nowski Stanislaw</b> Desin- kt. Hausrennng, Sienastr 45 599 82 <b>nowski Stanislaw</b> Dr. med. nowskastr 13 826 08 <b>nowski, Szymon</b> Verteilungs- stelle Siena Ponnaska Edo-Mech- ackstr 673 03 <b>nowski Tadeusz</b> Lestrzawa- n Pus-Kl-Str 13 936 45 <b>nowski W.</b> Eisenwarenwerk, Snaust 8 614 03 <b>nowski W.</b> Eisenw.-Verk. Hahernallee 12a 436 86 <b>nowski Wacław</b> + Nordbad- keo 130 442 17 <b>nowski Zdzislaw</b> + Co. arschauer Müllabfuhr Ka. Mac- wicz-Str 3/5 10 30 53 <b>nowski Zygmunt</b> Ing. Miko- waszstr 41 832 44 <b>onko H. u. Wojciechowski</b> , Bauing. GmbH Krucast 8 881 84 <b>onko Henryk</b> Ing. Roemer- Parkwastr 7 11 17 14 <b>on Adam</b> Dr. med. I. innere nch. Radomer Str 43 979 69 <b>on Stanislaw</b> Kinderkon- d. Bldg. I Markthalle 151 <b>ynska Eugenia</b> Widokstr 23 643 98 <b>ynski Alfons</b> Feldhern- ee 117a 436 62 <b>ynski Jan</b> Seilenn. Brown- dr 12 636 65 <b>ynski Janusz</b> Kienpore- Sst. Roemstr 28 826 04 <b>owski Adam</b> Ing.-Mech. Ba- rickastr 45 431 48 <b>owicz S.</b> Marschallstr 15 925 80	<b>Zaklad Ubezpie. Spolacznych u.</b> Hauptanstalt f. Sozialversiche- rung <b>Sozialversicherungskasse in</b> <b>Warschau</b> Weichselufer 33 Zentrale * 558 00 Deutscher Kommissar 240 66 Stellvertr. d. Deutsches Kommis- sars 348 48 Deutscher Chefars 628 95 Hausverwaltung 686 99 Zentrale Analit. Laborat. Sonn. u. Feiertage 11-12 558 04 Wirtschaftslager Dorfstr 20 805 13 Schreibmat.-Lager Polnast 34 992 62 Druckerei Litmannstadtstr 32 Snaust 8 627 56 Landgut Grotz Nachtverbindungen (nach 19 Uhr) Weichselufer 35 Rote Flotenz 558 01 Intendant 558 02 Garage 558 03 I. Bezirk Smulikowskistr 1/3 Zentrale * 558 00 Rostgeanstalt Zielast 11 675 78 II. Bezirk Polnast 34 Oberarzt 932 84 Vertrauenssritze 746 47 Büroleiter u. Sekretariat 630 71 Meldebüro u. Intendant 856 57 Referat d. Krankenhausewesens 822 06 Überschwester 744 14 Naturheilstalt 981 66 Chemisches Laboratorium 820 36 III. Bezirk Litmannstadt Str 32 Oberarzt 542 82 Vertrauenssritze 231 16 Büroleiter u. Ref. d. Fachkarte 217 34 Referat d. Hausärzte 345 88 Meldebüro u. Ref. d. Bartei-	<b>Spallinski Mieczyslaw</b> Snaust- kuchstr 1 740 59 <b>Spaltenstein Franciszek</b> Ind- nastr 9 927 27 <b>Sparkasse s. unter Kassel</b> <b>Spartaria Holzindustrie</b> GmbH Blumenstr 4 323 02 <b>Spartaria Holzindustrie</b> GmbH Madalinskistr 87 422 02 <b>Spaalska Jadwiga</b> Rakowiec- kastr 5 425 35 <b>Spasowicz Eugeniusz</b> 6 Sier- pienistr 21 944 47 <b>Spasowiczowa Aniela</b> + Be- amin Bedarskastr 26 238 95 <b>Spaw</b> Stahlkonstruktionswerke Kwiecinski Wl. Pradyskuskistr 17 321 49 <b>Specht Elzbieta</b> Kursstr 108 10 23 49 <b>Specht Willi</b> Ingenieurwesen Marast 6 900 89 <b>Speck Paula</b> Welo. u. Spiritus- sechldg Neue Welt 3 805 72 Orlestr 19 633 14 <b>Spedilio</b> Transportbüro Postpl 9 358 00 <b>Speditionshaus Adolf u. Ed- ard Holler</b> Zweigstelle des Dlugastr 29 11 15 70 <b>Spedo</b> Sped.-Büro Marschallstr 10 692 59 <b>Speich Walter</b> + Ing. Kln. Marast 8 738 24 <b>Speldel Max</b> Beauftragte d. Kom- missar. Verwaltung sichergestellt. Grundstücke I. Warschau Grotz- gustr 2 426 35 <b>Spel</b> + elekt. Anl.-u. Materialver- lager Barozowicz M. Gasewski B. Wapolskistr 9 734 57 <b>Sperling J. &amp; Co.</b> Wagon u. Mo- tallwarenher. GmbH Myznarka- str 30 253 59 <b>Sperling Juliusz</b> Kln. Wagon- her. u. Motallwarenher. GmbH	<b>Spietarski Jan</b> Zahnr. Jave- ryzyskistr 7 723 12 <b>Splawa-Neyman Helena</b> Neue Burgstr 10 998 49 <b>Splawa-Neyman Jan</b> Ing.-Arch. Radomer Str 43 946 28 <b>Spietarski Jan</b> Zahnr. Jave- ryzyskistr 7 723 12 <b>Splawa-Neyman Helena</b> Neue Burgstr 10 998 49 <b>Splawa-Neyman Jan</b> Ing.-Arch. Radomer Str 43 946 28	<b>Spietarski Jan</b> Zahnr. Jave- ryzyskistr 7 723 12 <b>Splawa-Neyman Helena</b> Neue Burgstr 10 998 49 <b>Splawa-Neyman Jan</b> Ing.-Arch. Radomer Str 43 946 28	<b>Spychalski Wit</b> solim. Skastr 1 <b>Spyas Jan</b> Napi- Inh. techn. Hand- skastr 1 <b>Srebrny Kazimi-</b> lusz 16 <b>Srednicka Wlad</b> Kozettmacherin F <b>Srednicki Br. M.</b> we Kolost 10 <b>Srednicki Broni</b> Loki Wroklawstr 1 <b>Srednicki Stani-</b> Kladzestr Targow <b>Srednicki Stanis-</b> str 02 <b>Srednicki Leon</b> str 31 <b>Srocki Stefan</b> Pl <b>Sroczyńska Apol</b> str 20 <b>Sroczyńska Iren</b> <b>Sroczyńska Kar</b> bldg. Inbestr 26 <b>Sroczyński u. He</b> nrl. Nutenlager u schallstr 91 <b>Sroczyński E. S</b> Metallw. Abt. eld nigskenger Str 4/4 <b>Sroczyński J. &amp;</b> med. Laborat. E <b>Sroczyński Jan H</b> ria-Kamienica-Str <b>Sroczyński Kar</b> Leczykstr 4 <b>Sroczyński Karo</b> + Grybowkstr <b>Sroczyński Kazi</b> Kinderarzt Sporta <b>Sroczyński Wito</b> str 2a
---	--	---	--	--	--

# Simple DNS

DOMAIN NAME	IP ADDRESS
spotify.com	98.138.253.109
cs.rutgers.edu	128.6.4.2
www.google.com	74.125.225.243
www.princeton.edu	128.112.132.86



- Key idea: Implement a server that looks up a table.
- Will this scale?
  - Every new (changed) host needs to be entered in this table
  - Performance: can the server serve billions of Internet users
  - Failure: what if the server or the database crashes?
  - How to secure this server?