ECE 650
Systems Programming & Engineering

Spring 2018

Networking Introduction

Tyler Bletsch
Duke University

Slides are adapted from Brian Rogers (Duke)

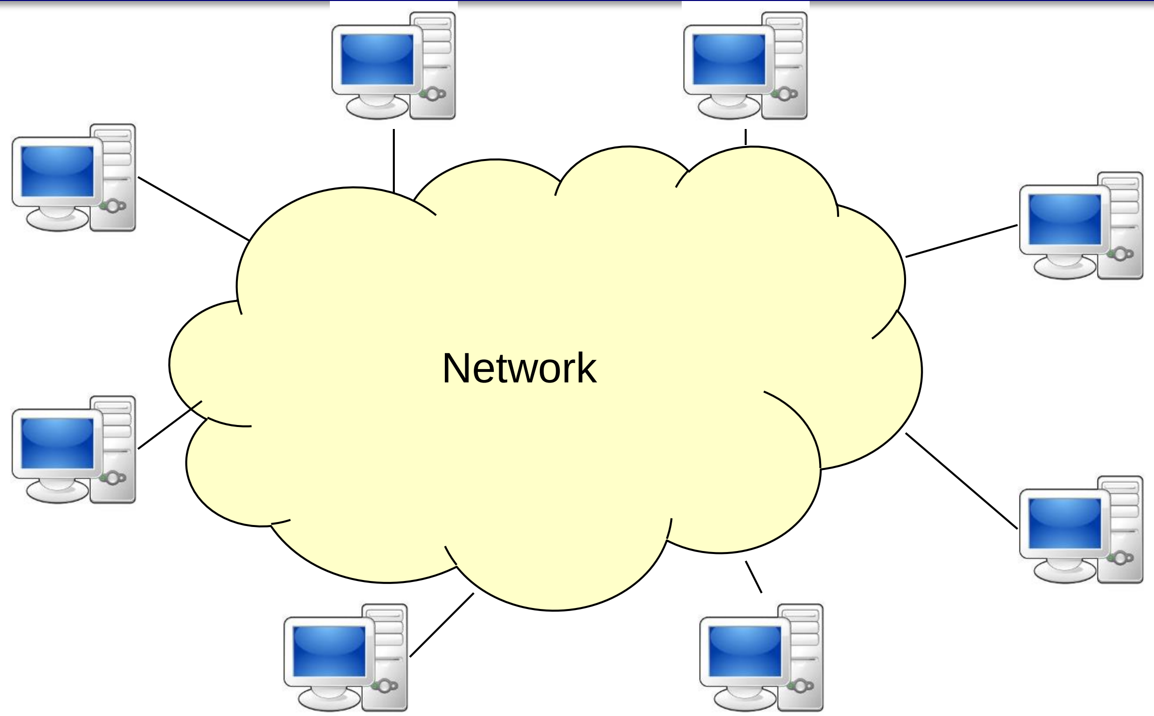
Computer Networking

- A background of important areas was covered in 550
  - What is a network
  - 7 layer OSI networking stack
  - IP and routing
  - TCP sessions
- We will cover more depth on these topics and more
- Homework 3 will be network focused
- May want to review this background material

What is a Computer Network?

- A network is a group of interconnected computers
- Motivations for computer networks
  - Share resources:
    - Files, information, databases (and remote data access)
    - Compute resources (distributed computing)
    - Devices (e.g. printers)
  - Communication (any-to-any) between users & applications
  - Separate client and server
  - Connection to Internet network is now an important part of a PC

Internet



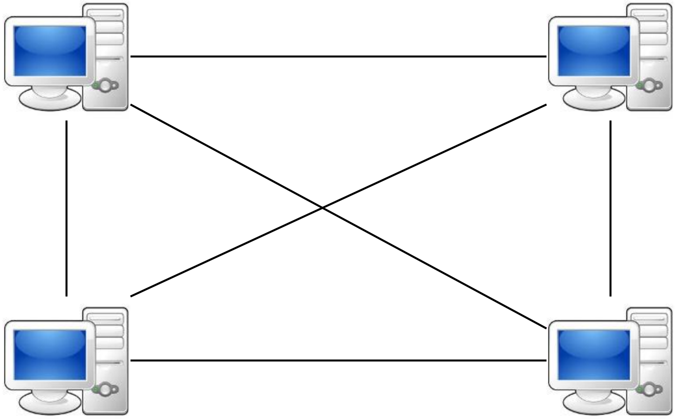
- We are familiar with "network endpoints":
  - PCs, servers, mobile devices, etc.
- Network goal is for any-to-any communication

Network Links

- At the lowest level we have links
- DSL, T1, T3, Fiber, etc.
- Characterized by
  - Bit rate (e.g. 100 Mbps, 1Gbps)
  - Propagation delay (latency; mostly a function of distance)
  - Transfer time on a link = #bits / bit rate + propagation delay

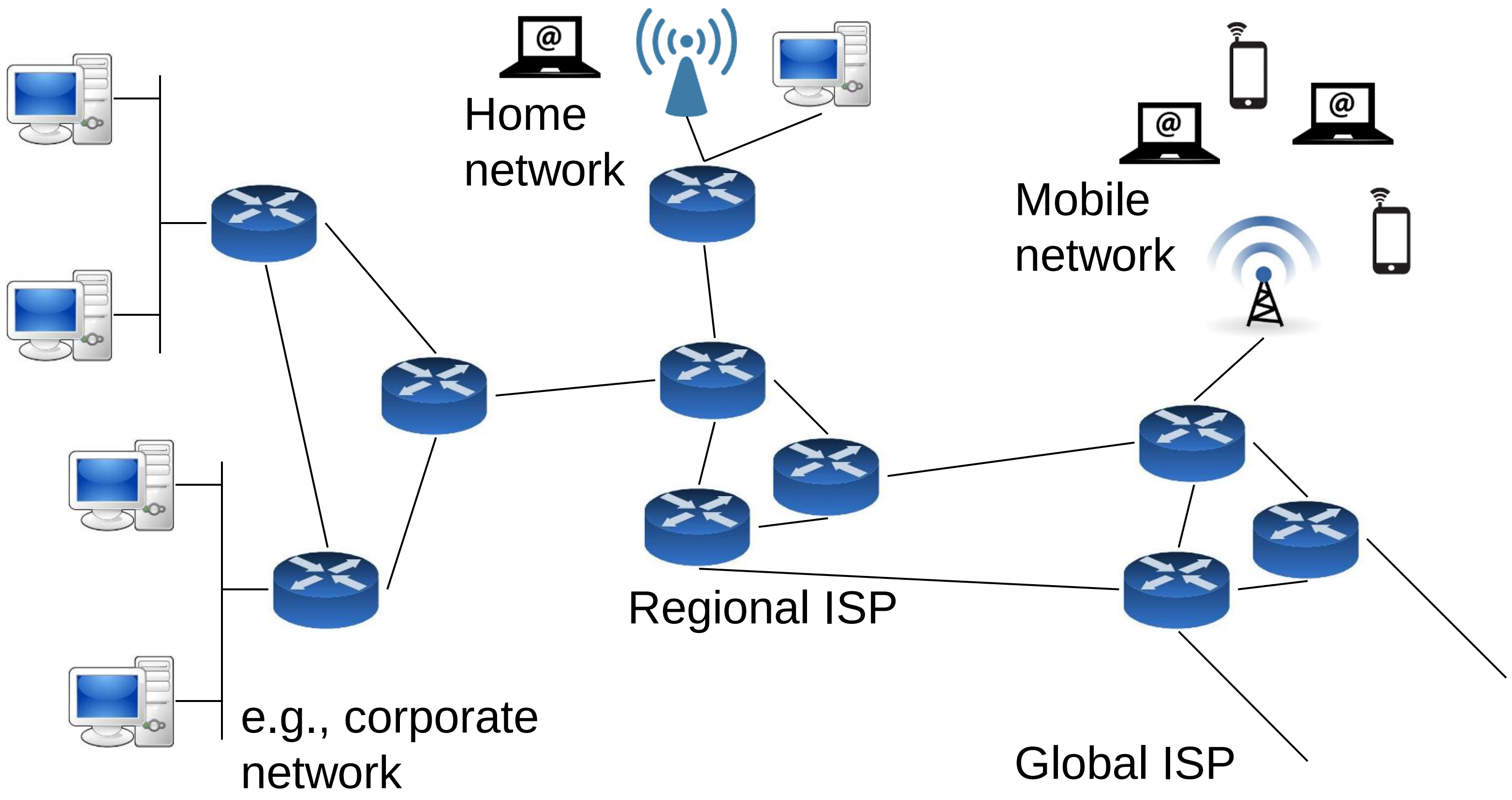
Connectivity in the Internet

- A point-to-point mesh?
- Clearly not sustainable for large networks
  - N^2 links required
  - Add new endpoint: new link added to all existing endpoints



Network Structure

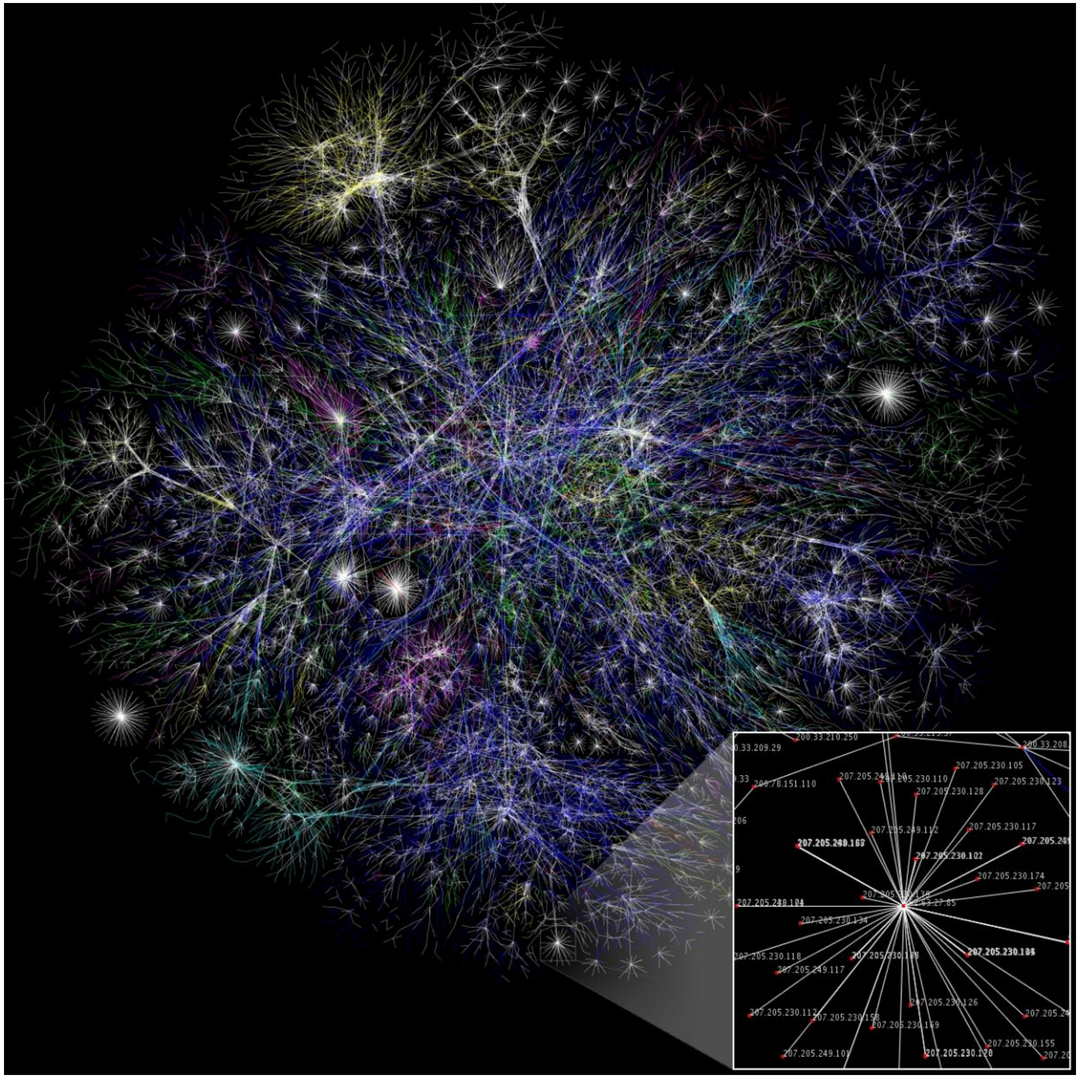
- Need to share infrastructure!
- Routers and switches (intermediate nodes) allow sharing



Internet Backbone

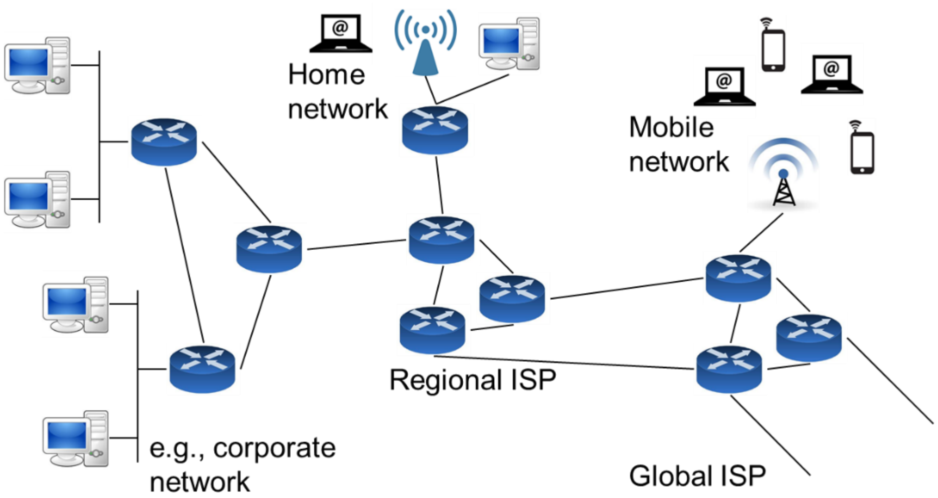
- From Wikipedia:

- Due to sharing, we get a structure that looks like this
- Localized “stars” connected to others



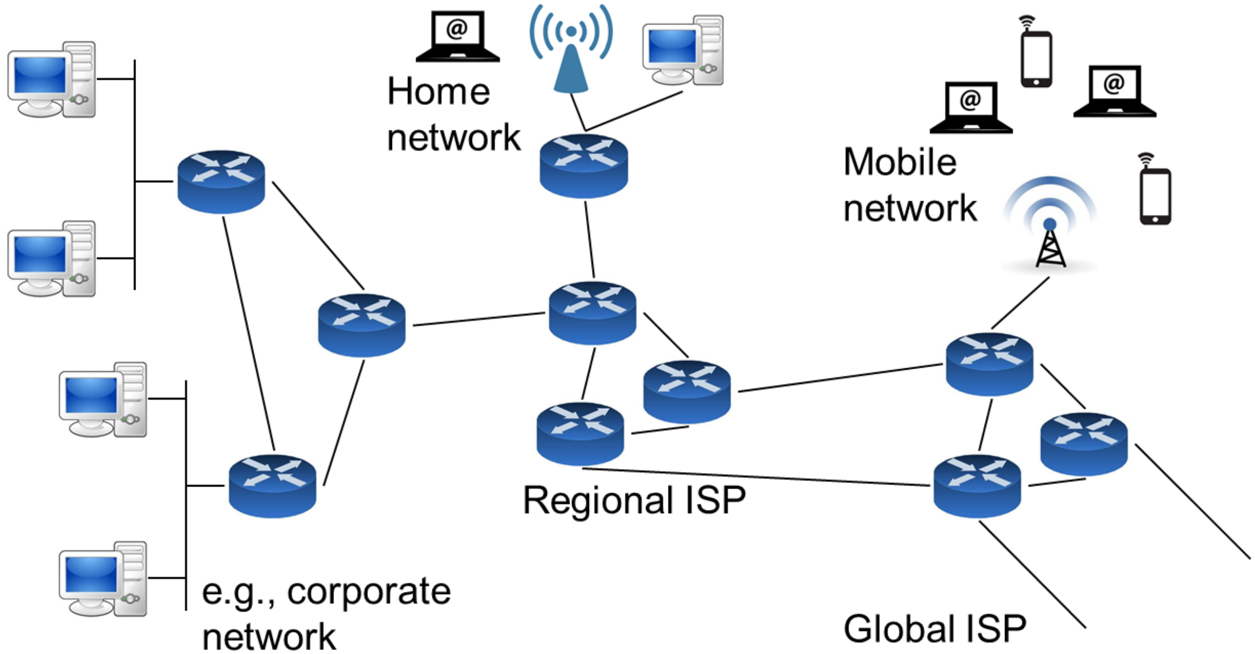
Usage Models

- Network endpoints run application programs
  - Web browser, email client, ftp, ssh, etc.
- Client / Server model
  - Client endpoints requests a service from a server
  - E.g. client / server web page service
- Peer-to-peer (P2P)
  - Direct client communication (e.g. Skype, BitTorrent)



Network Backbone

- Network of interconnected routers is the internet core
- Key questions:
  - How is data transferred between endpoints through the network?
  - How are the network links shared for communication?

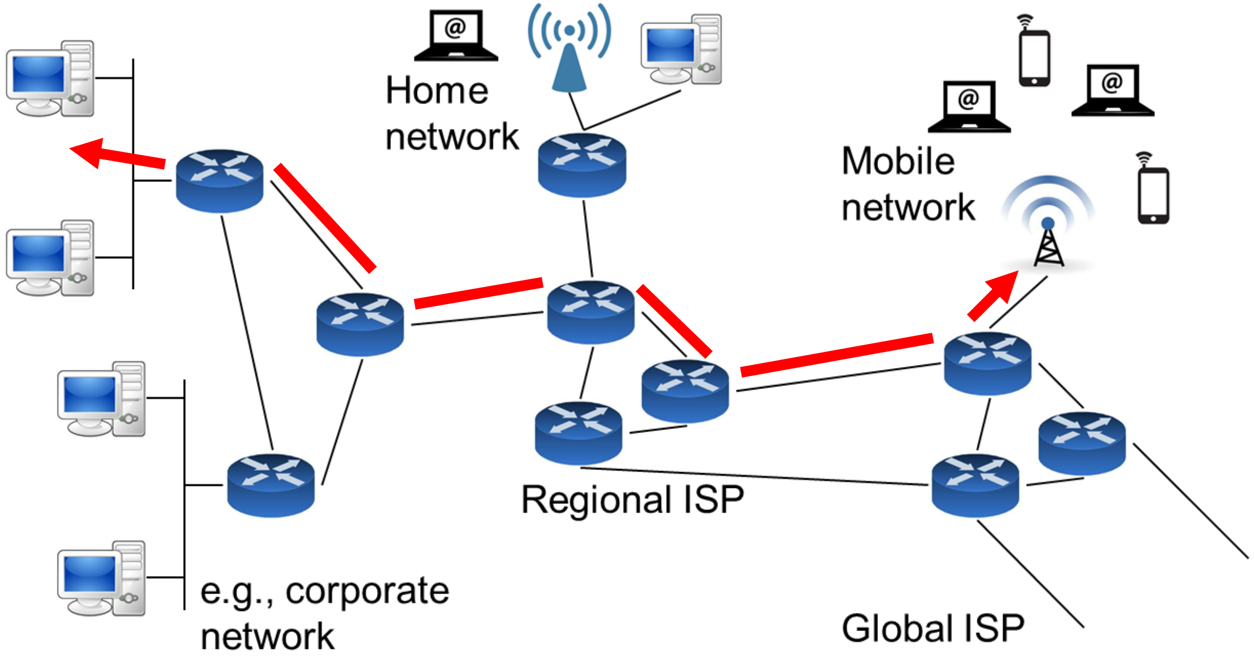


Two Sharing Strategies

- Circuit Switching
  - Create & allocate dedicated path for a transmission
    - From one endpoint to another through a series of routers / switches
  - This is how the old telephone network operates
- Packet Switching
  - Divide each message up into a sequence of packets
  - Packets sent from one network node (e.g. router) to the next
  - Each router decides the destination for the next "hop"
  - Eventually, packets of the message should arrive at destination

Circuit Switching

- Reserve end-to-end resources for each transmission
  - Link bandwidth, router resources
  - Performance guaranteed
  - Requires a setup process



Circuit Switching Process

1. Establish the end-to-end circuit
   - "Dialing" in phone network
2. Communication
   - Send information through network
3. Close circuit ("tear down")
   - Deallocate resources

- If no end-to-end circuit can be established
  - E.g. due to lack of resources available
  - Re-try is required (e.g. busy signal on phone)

Circuit Switching Networks

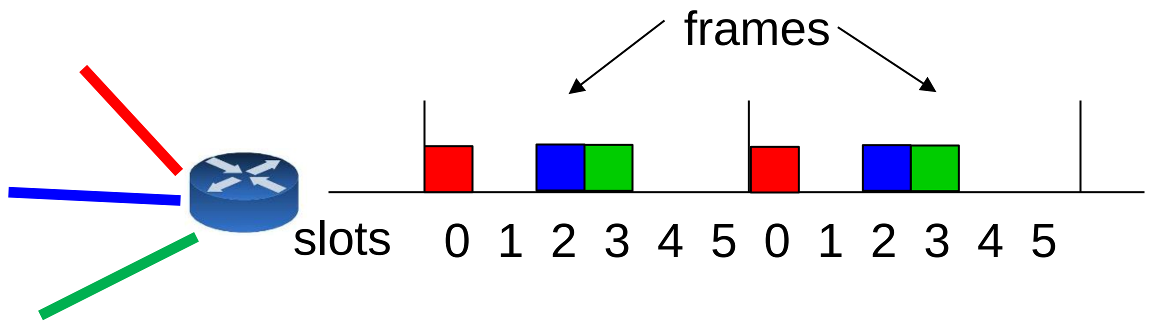
- Often not efficient
  - Capacity of circuit is allocated for entire duration of connection
  - The transmission often does not fully utilize channel for duration
- Delay is required to establish the circuit
- Network is transparent to users after circuit is established
  - Like having a dedicated wire to the target endpoint
- Data may be transmitted at fixed rate w/ propagation delay

Multiplexing

- Routers & links can carry multiple communications
  - E.g. if each communication uses only a fraction of total bandwidth
- Need a mechanism to divide network resources into pieces
- How can we divide link bandwidth into pieces? Multiplexing!
  - Frequency division multiplexing (FDM)
  - Time division multiplexing (TDM)
  - Code division multiplexing (often used in cellular technology)
- Motivation
  - Carry multiple signals on a single medium
  - More efficient use of transmission medium

Time division multiplexing (TDM)

- Divide time into frames; frames into slots
- Each transmission stream gets a relative slot position within a frame



- Requires synchronization between sender and receiver

Frequency division multiplexing (FDM)

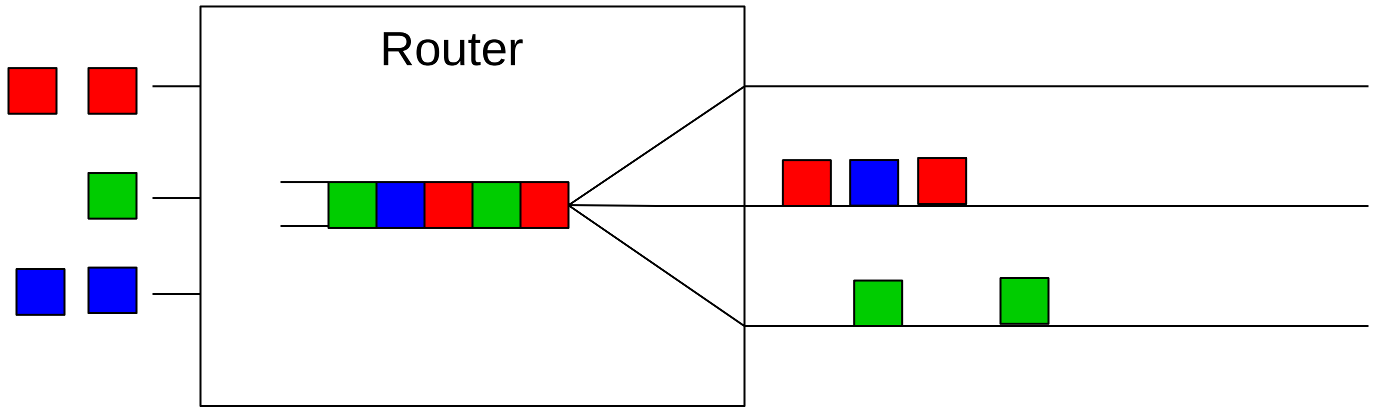
- Separate frequency spectrum of the medium
  - Into non-overlapping, smaller frequency bands
- A channel is allocated to a smaller frequency band
  - Has access to that frequency band for the entire life of circuit

- Can combine TDM + FDM
  - Use FDM to divide frequency spectrum
  - Use TDM to time-slice channels across slots within each band

Packet Switching

- Break information in small chunks: packets
- Each packet forwarded independently
  - Must add metadata to each packet
- Allows statistical multiplexing
  - \+ High utilization
  - \+ Very flexible
  - Fairness not automatic
  - Highly variable queueing delays
  - Different paths for each packet

Packet Switched Routers



- Multiplex w/ queue(s) in the router
- Demultiplex with packet header info:
  - Destination endpoint

Sample Packet Format

- Highly simplified example – we will look more closely later
- Header
  - Source Address (SA)
  - Destination Address (DA)
  - Sequence number (which packet index within a transmission)?
- Data (or payload)
- Trailer: e.g. CRC for error detection

1010	0110	0001	1010100010110001	010110
SA	DA	SEQ	Payload	CRC

Packet Routing

- Store & Forward Routing
  - Entire packet must arrive at router before next hop
  - Each router adds delay to the packet transmission latency
- Cut-through Routing
  - Pieces of a packet may be forwarded onto next hop right away
  - More difficult to manage packet transmission

Packet Routing

- Queues introduce new effects:
  - Variable delay
    - Delay = queueing delay + propagation delay + transmission delay + processing delay
  - Packet loss
    - When packet arrive to a router with a full queue, they are dropped
- Ordering is impacted:
  - Packets of a stream may arrive at destination endpoint out of order
  - May take different paths through network

Comparison

Circuit Switching	Packet Switching
Constant delay	Variable delay
In-order packet arrival	Out-of-order packet arrival
Inefficient use of bandwidth	Efficient use (sharing) of bandwidth
Simple routing	Complex routing
Quality is "all or nothing"	"Graceful" degradation of quality
Low complexity of control	High complexity of control

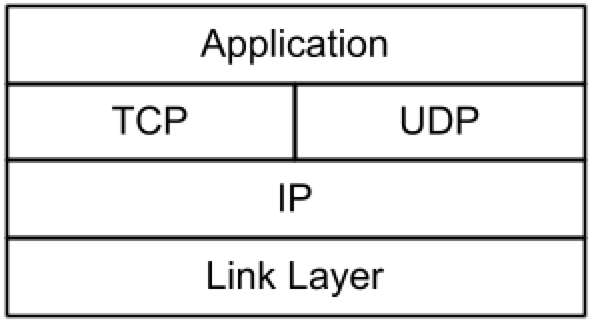
Managing Complexity

- Let's turn attention back to the endpoints
  - Now that we briefly understand what the network looks like

- Very large number of computers
- Incredible variety of technologies
  - Each with very different constraints
- No single administrative entity
- Evolving demands, protocols, applications
  - Each with very different requirements!
- How do we make sense of all this?

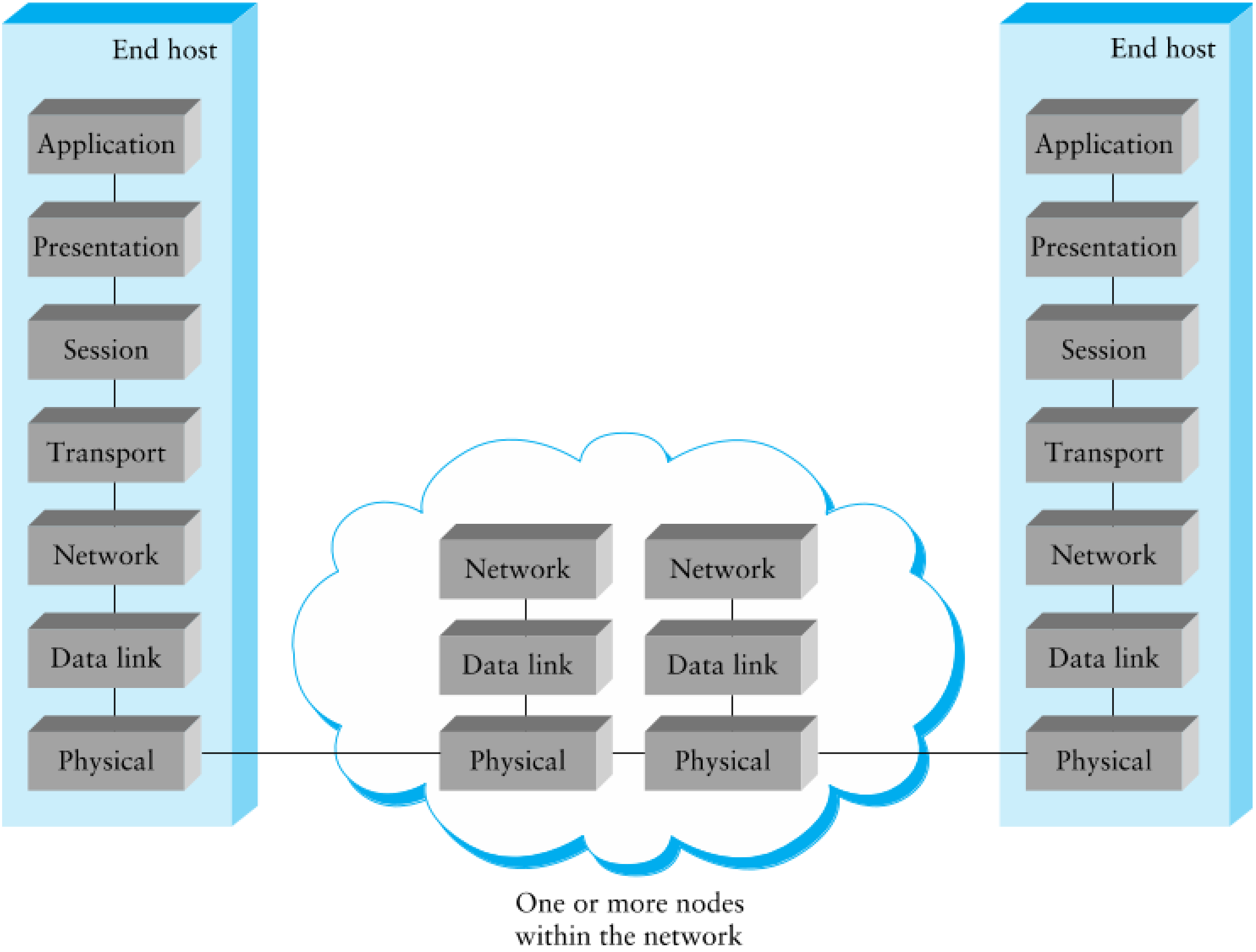
Layering

- We see layers of abstraction

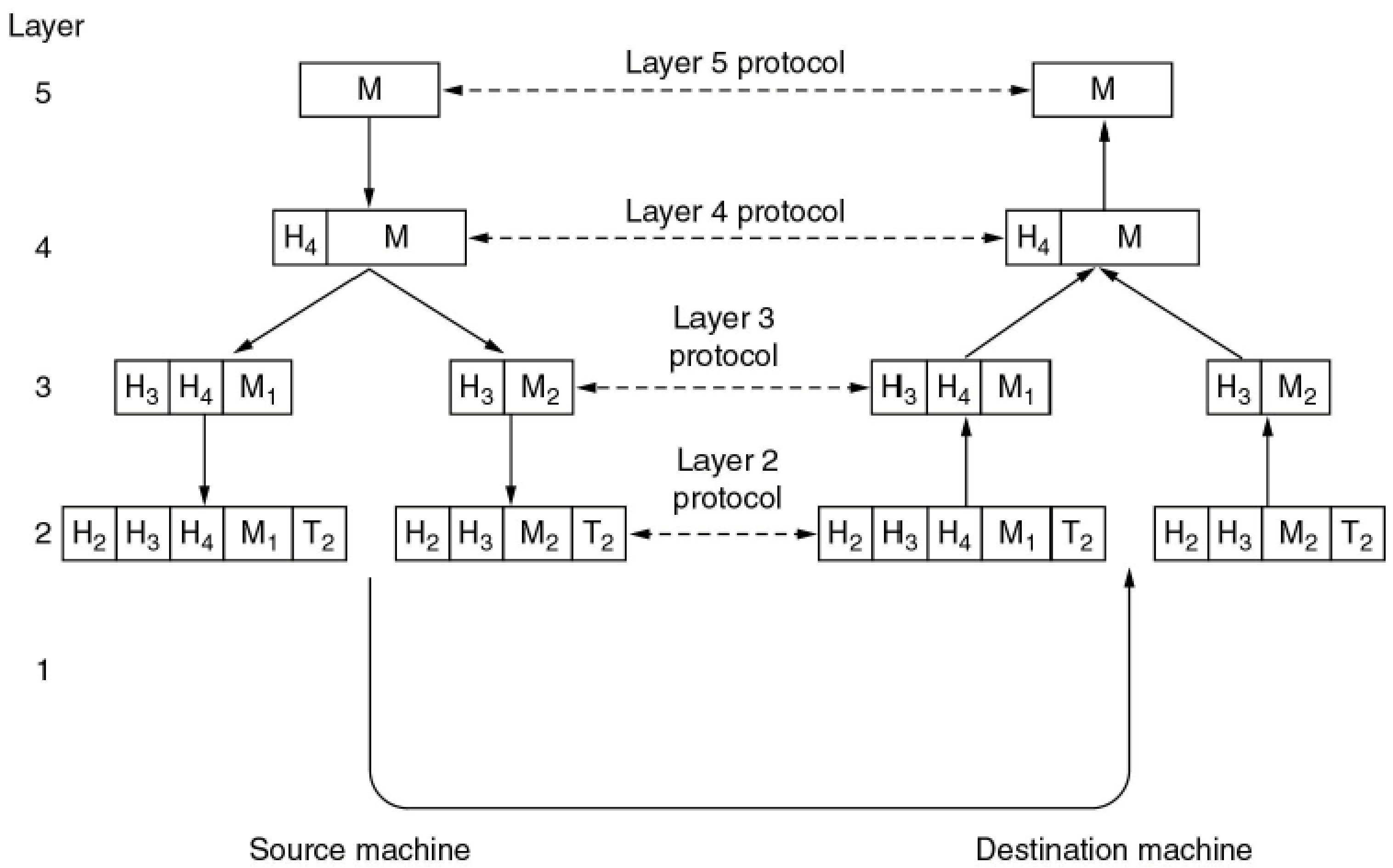


- Separation of concerns
  - Break problem into separate parts
  - Solve each one independently
  - Tie together through common interfaces: abstraction
  - Encapsulate data from layer above inside data from layer below
  - Allow independent evolution

OSI Reference Model



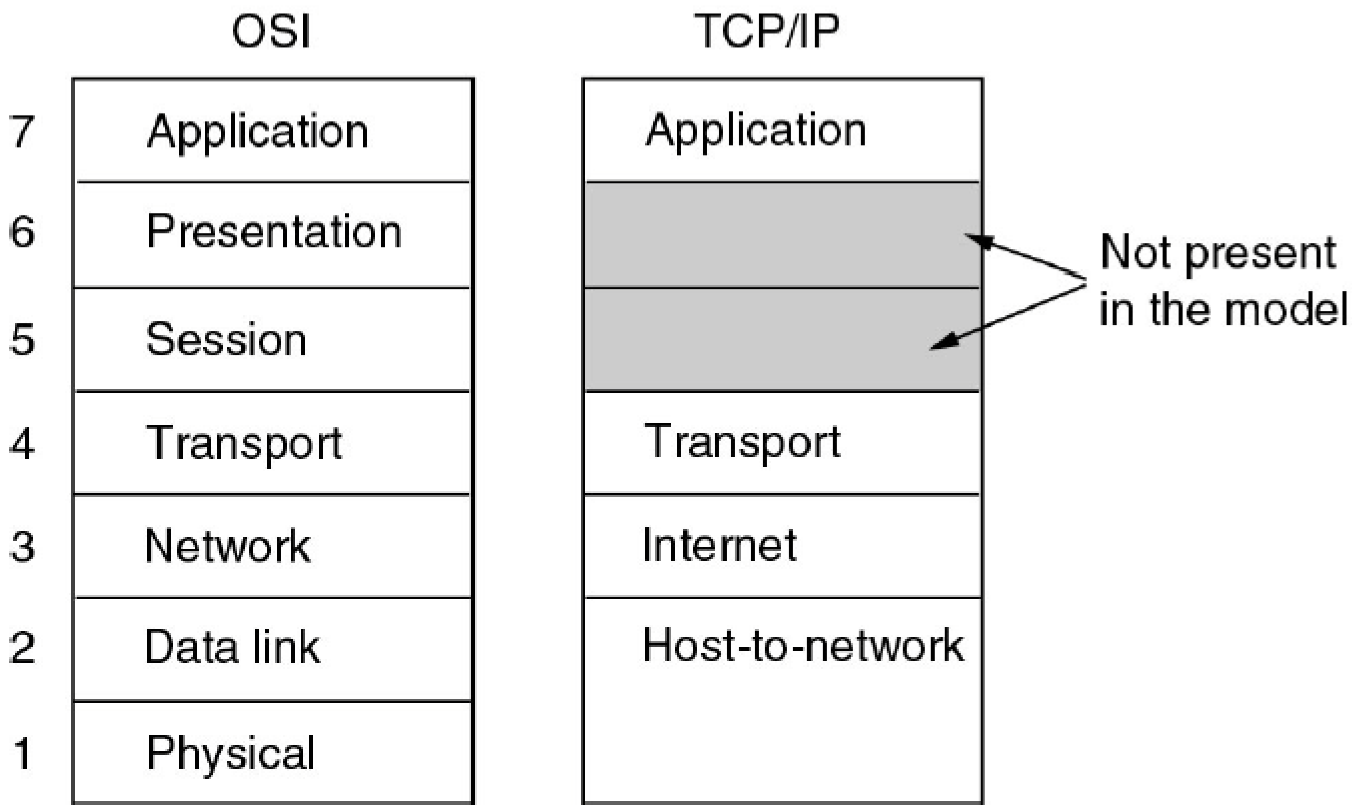
Protocol Hierarchies - Example



Protocol

- Each abstraction layer communicates via a protocol
- Protocols define:
  - Message format
  - Order of messages sent / received
  - Actions to take on message transmission / receipt

TCP/IP Model


OSI
TCP/IP
7
Application
6
Presentation
5
Session
4
Transport
3
Network
2
Data link
1
Physical
Application
Transport
Internet
Host-to-network
Not present
in the model

Layer 1 & 2

- Layer 1: Physical Layer
  - Encoding of bits to send over a single physical link

Examples: Ethernet, 802.11 WiFi
(the part of the spec that says how to send bits)

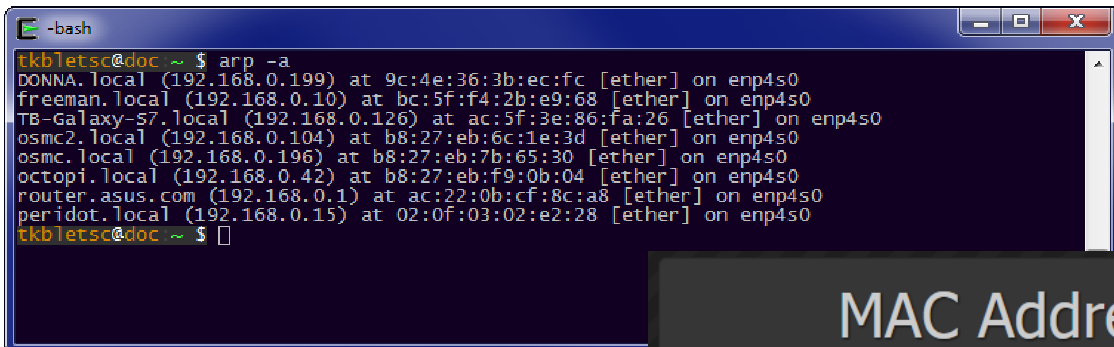
- Layer 2: Link Layer

Examples: Ethernet, 802.11 WiFi
(the part of the spec that how to send packets to a host on this network)

  - Framing and transmission of a collection of bits into individual messages sent across a single subnetwork (one physical topology)
  - Provides local addressing (MAC)
  - May involve multiple physical links
  - Often the technology supports broadcast: every "node" connected to the subnet receives
  - Examples:
    - Modern Ethernet
    - WiFi (802.11a/b/g/n/etc)
  - MAC address is 48-bit value burned into network card; globally unique
    - First 3 bytes are assigned to manufacturer (OUI: Organizationally Unique Identify)

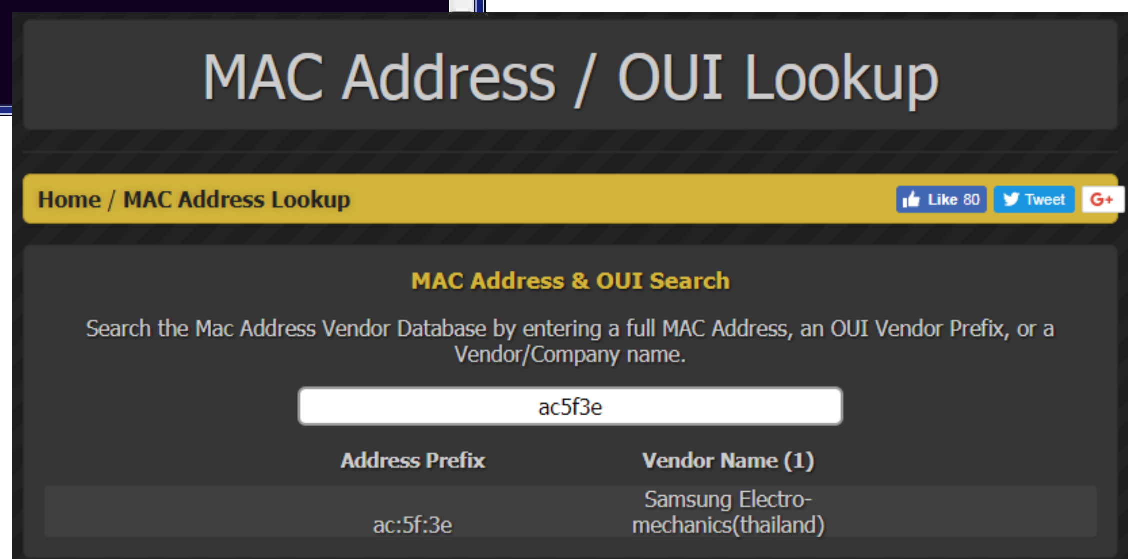
Layer 1/2 demo: ARP

- Address Resolution Protocol (ARP): how we figure out the layer 2 address (MAC address) for a given layer 3 address (IP address)
  - Can inquire to see known MAC addresses
  - Can use OUI (first 3 bytes) to check manufacturer of devices!



Left: ARP listing for my home server

Below: Lookup of manufacturer of the "TB-Galaxy-S7" device



http://www.whatsmyip.org/mac-address-lookup/

Layer 3

Example: Internet Protocol (IP)
(how to send packets between networks)

- Bridges multiple "subnets" to provide end-to-end connectivity between nodes
- Provides global addressing (IP addresses)
- Only provides best-effort delivery of data
  - No retransmissions, etc.
- Works across different link technologies

Below: Diagnostic tool showing the the IP addresses passed on the way from my home to duke.edu

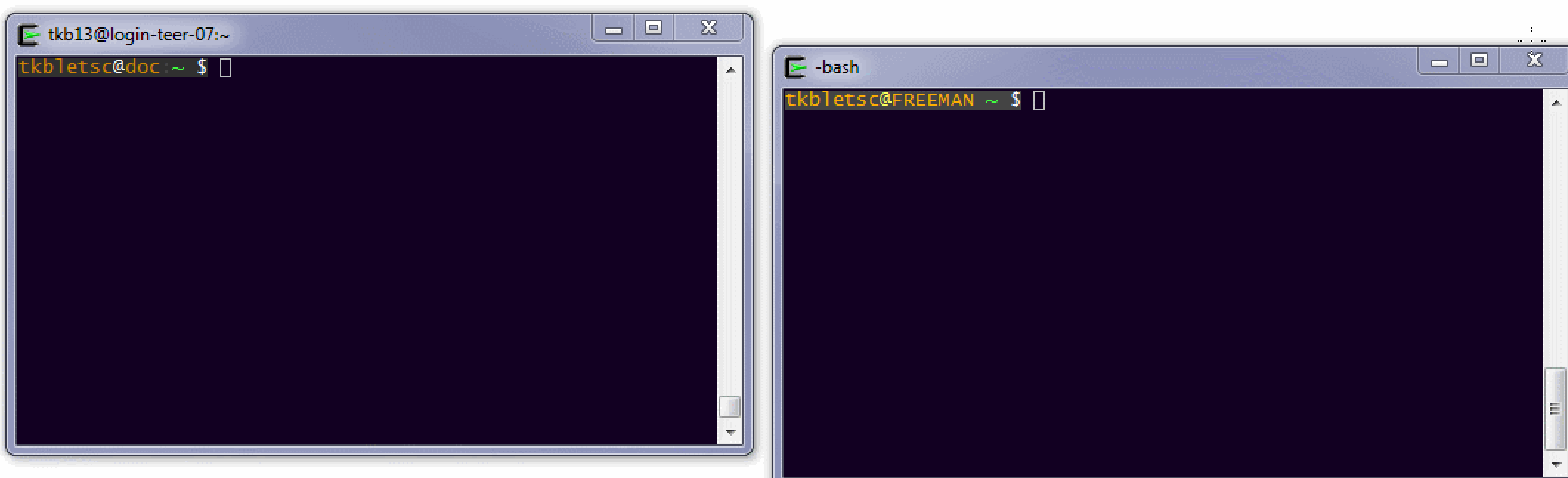
```
-bash
trtkbletsc@FREEMAN ~ $ tracert duke.edu

Tracing route to duke.edu [152.3.72.197]
over a maximum of 30 hops:

  1    <1 ms    <1 ms    <1 ms  router.asus.com [192.168.0.1]
  2    13 ms    12 ms     9 ms  cpe-174-99-64-1.nc.res.rr.com [174.99.64.1]
  3    31 ms    32 ms    34 ms  cpe-174-111-104-017.triad.res.rr.com [174.111.104.17]
  4    16 ms    13 ms    10 ms  cpe-024-025-062-000.ec.res.rr.com [24.25.62.0]
  5    21 ms    15 ms    23 ms  24.93.67.200
  6    23 ms    23 ms    23 ms  gig9-0-0.clmascmhe-rtr1.sc.rr.com [24.93.64.41]
  7    24 ms    15 ms    15 ms  24.93.67.205
  8    20 ms    34 ms    24 ms  cpe-024-074-247-067.carolina.res.rr.com [24.74.247.67]
  9    25 ms    19 ms    21 ms  rrcs-24-172-68-247.midsouth.biz.rr.com [24.172.68.247]
 10    19 ms    37 ms    15 ms  rrcs-98-101-20-135.midsouth.biz.rr.com [98.101.20.135]
 11    23 ms    18 ms    23 ms  rrcs-24-172-64-46.midsouth.biz.rr.com [24.172.64.46]
 12     *        *        *     Request timed out.
 13     *        *        *     Request timed out.
 14     *        *        *     Request timed out.
 15     *        *        *     Request timed out.
 16    28 ms    25 ms    24 ms  rsv-152-3-72-254.oit.duke.edu [152.3.72.254]
 17    27 ms    29 ms    24 ms  152.3.72.197

Trace complete.
tkbletsc@FREEMAN ~ $
```

Layer 4

Example: TCP/UDP
(how to establish a logical channel, maybe even a reliable channel)

- End-to-end communication between processes
- Different types of services provided:
  - UDP: unreliable datagrams
  - TCP: reliable byte stream
- "Reliable" = keeps track of what data were received properly and retransmits as necessary
- This is the layer that applications talk with

Below: Sending data between two computers via a raw TCP socket using the 'netcat' (nc) tool.

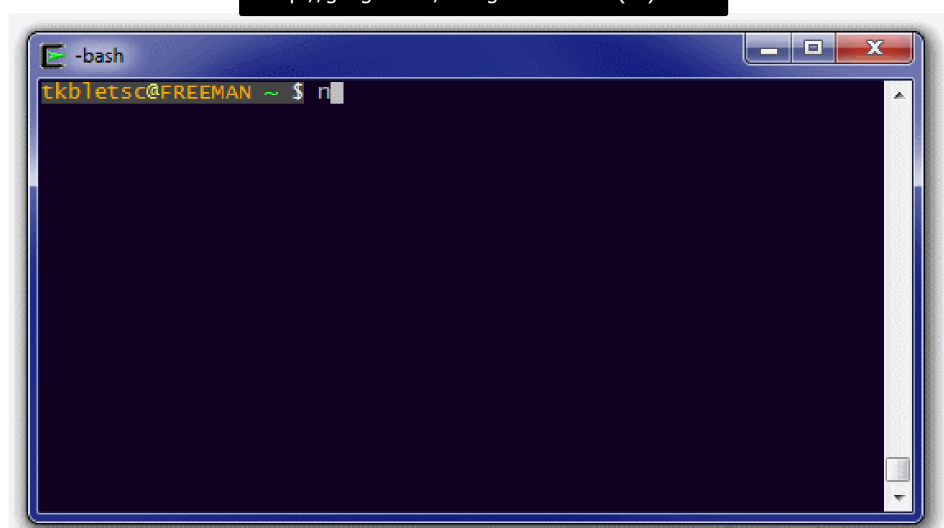


Layer 5

Example: HTTP, lots more
(fulfill the actual use case)

- Communication of whatever you want
- Can use whatever transport(s) is(are) convenient/appropriate
- Freely structured
- Examples:
  - Skype (UDP)
  - SMTP = email (TCP)
  - HTTP = web (TCP)
  - Online games (TCP and/or UDP)

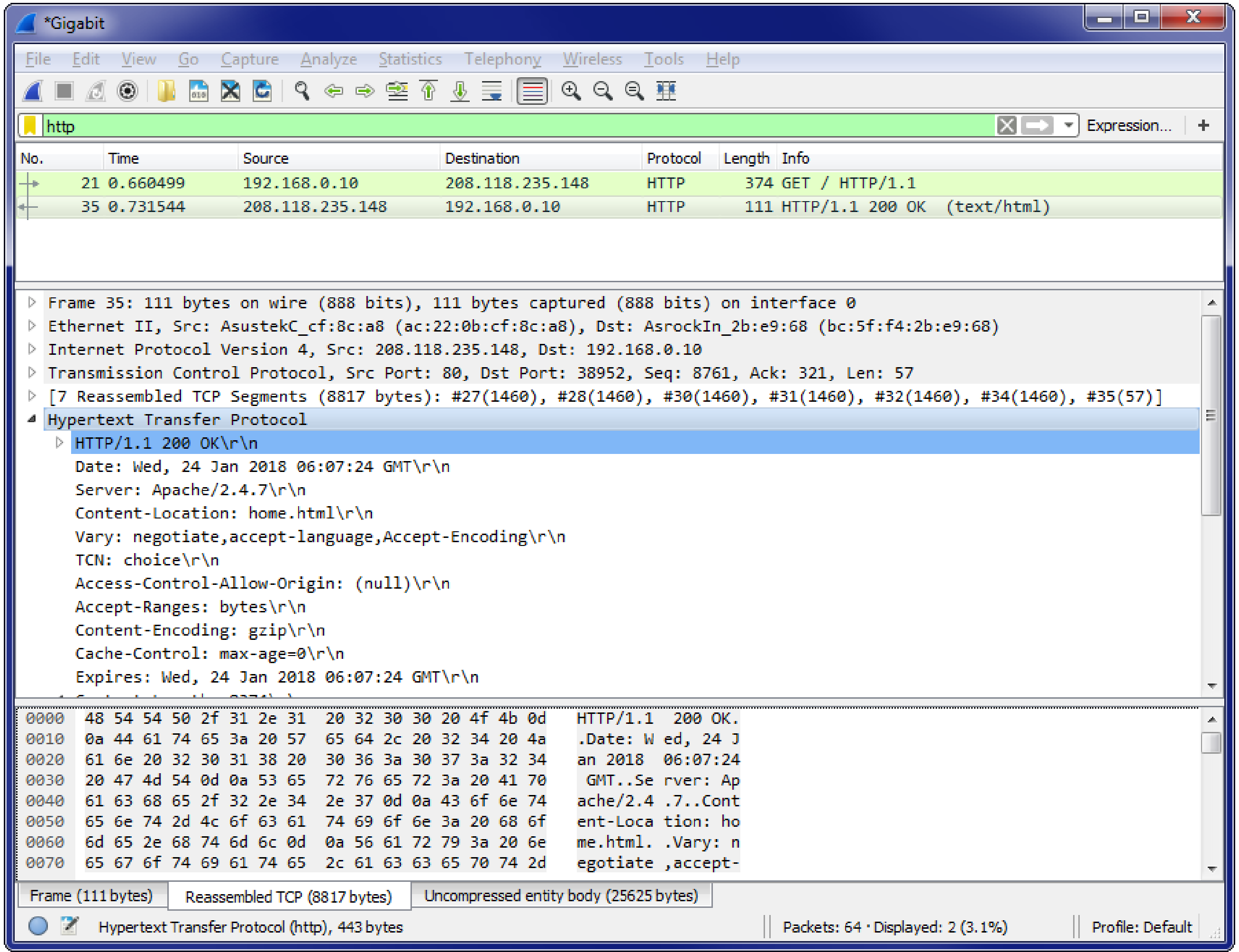
Below: Manually speaking HTTP to request http://google.com/ using the 'netcat' (nc) tool.



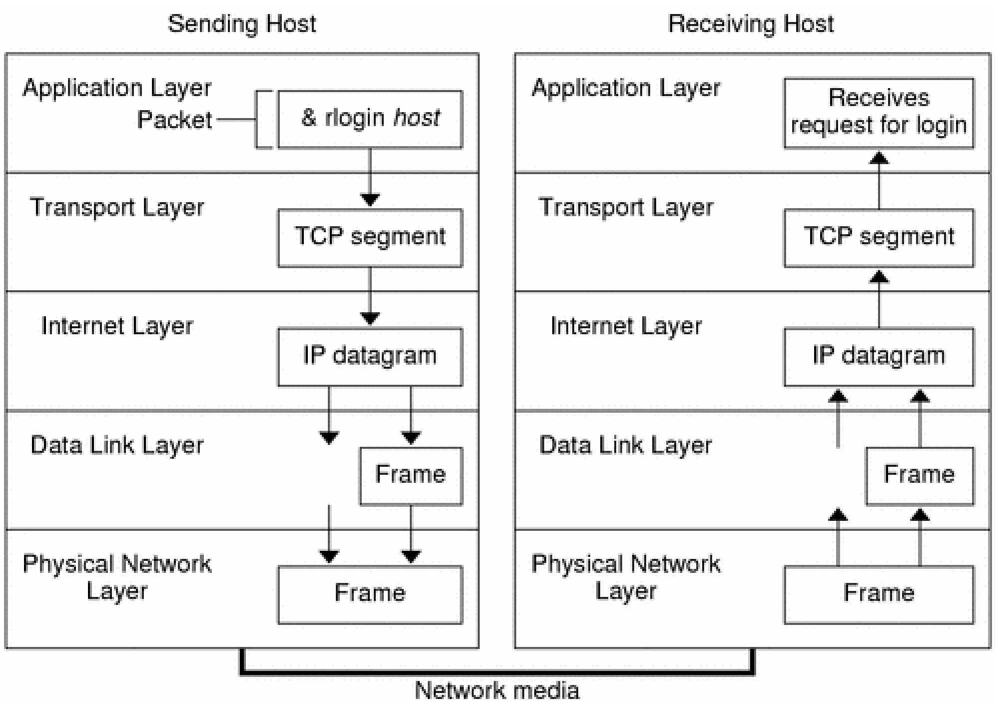
Demo: Wireshark

- Can observe packets in transit with network sniffer, e.g. Wireshark

Below: Trace of a Firefox request for http://www.gnu.org/



Summary



- Get http://pics.com/dog.jpg (overall goal)
- Transport dog.jpg data stream reliably
- Send packets of data stream across world to pics.com
- Send packet to router on my network; I assume it can eventually reach pics.com
- Put electrical pulses on wire that represent the packet