Sensitive Information in a Wired World

CPSC 457/557, Fall 2013

Lectures 6 and 7; Sept 17 and 19, 2013

1:00-2:15 pm; AKW 400

http://zoo.cs.yale.edu/classes/cs457/fall13/

Internet History

- Late 1960s and early 1970s: ARPANET
  - US Department of Defense
  - Connects small ARPA-sponsored data networks
  - Ground breaking testbed for network ideas and designs
- Early 1980s: Other wide-area data networks are established (*e.g.*, BITNET and Usenet).
- Late 1980s and early 1990s:
  - “ARPANET” fades out.
  - US Gov’t sponsors NSFNET, which connects large regional networks.
  - Commercial data networks become popular (*e.g.*, Prodigy, Compuserve, and AOL).
- Mid-1990s: Unified “Internet”

Internet Protocols Design Philosophy

- Ordered set of goals:
  1. multiplexed utilization of **existing networks**
  2. survivability in the face of failure
  3. support multiple types of communications service
  4. accommodate a variety of network types
  5. permit distributed management of resources
  6. cost effective
  7. low effort to attach a host
  8. account for resources
- Not all goals have been met

Packets!

- Basic decision: use packets not circuits (Kleinrock)
- Packet (*a.k.a.* datagram)

Dest Addr	Src Addr	payload

  - self contained
  - handled independently of preceding or following packets
  - contains destination and source **internetwork** address
  - may contain processing hints (*e.g.*, QoS tag)
  - **no delivery guarantees**
    - net may drop, duplicate, or deliver out of order
    - reliability (where needed) done at higher levels

Telephone Network

- Connection-based
- Admission control
- Intelligence is "in the network"
- Traffic carried by relatively few, "well-known" communications companies

Internet

- Packet-based
- Best effort
- Intelligence is "at the endpoints"
- Traffic carried by many routers, operated by a changing set of "unknown" parties

Technology Advances

	1981	2006	Factor
MIPS	1	50,000	50,000
$/MIPS	$100K	$0.02	5,000,000
DRAM Capacity	128KB	4GB	30,000
Disk Capacity	10MB	750GB	75,000
Network B/W	9600b/s	40Gb/s	4,000,000
Address Bits	16	64	4
Users/Machine	10s	<=1	<0.1

- Expensive machines, cheap humans
- Cheap machines, expensive humans
- (Almost) free machines, really expensive humans, and communities

The Network *is* the Computer

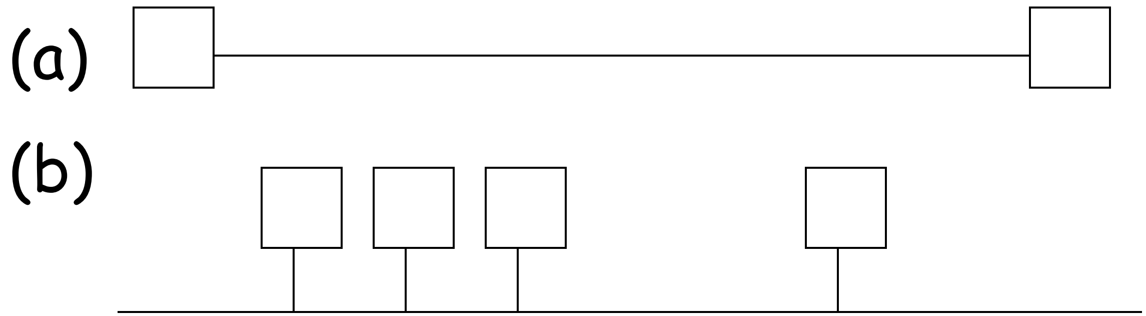
- Relentless decentralization
  - "Smaller, cheaper, more numerous"

    mainframe → mini → PC → palms → ubiquitous/ embedded
  - More computers → more data communication
- (Shifting) reasons computers talk to each other
  - Efficient sharing of machine resources
  - Sharing of data
  - Parallel computing
  - *Human* communication

The Network *is* the computer (continued)

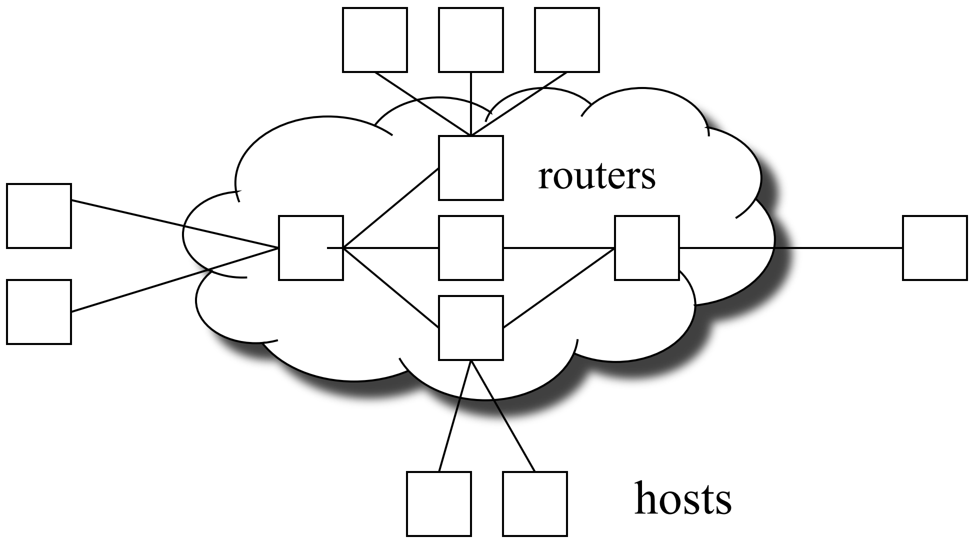
- Networks are everywhere and they are converging
  - SAN, LAN, MAN, WAN
  - All converging towards a similar technology
  - Sensor nets
- New chapter of every aspect of computer science
  - Re-examine virtually all the issues in the context of distributed systems or parallel systems
- This is only the beginning.

Directly Connected



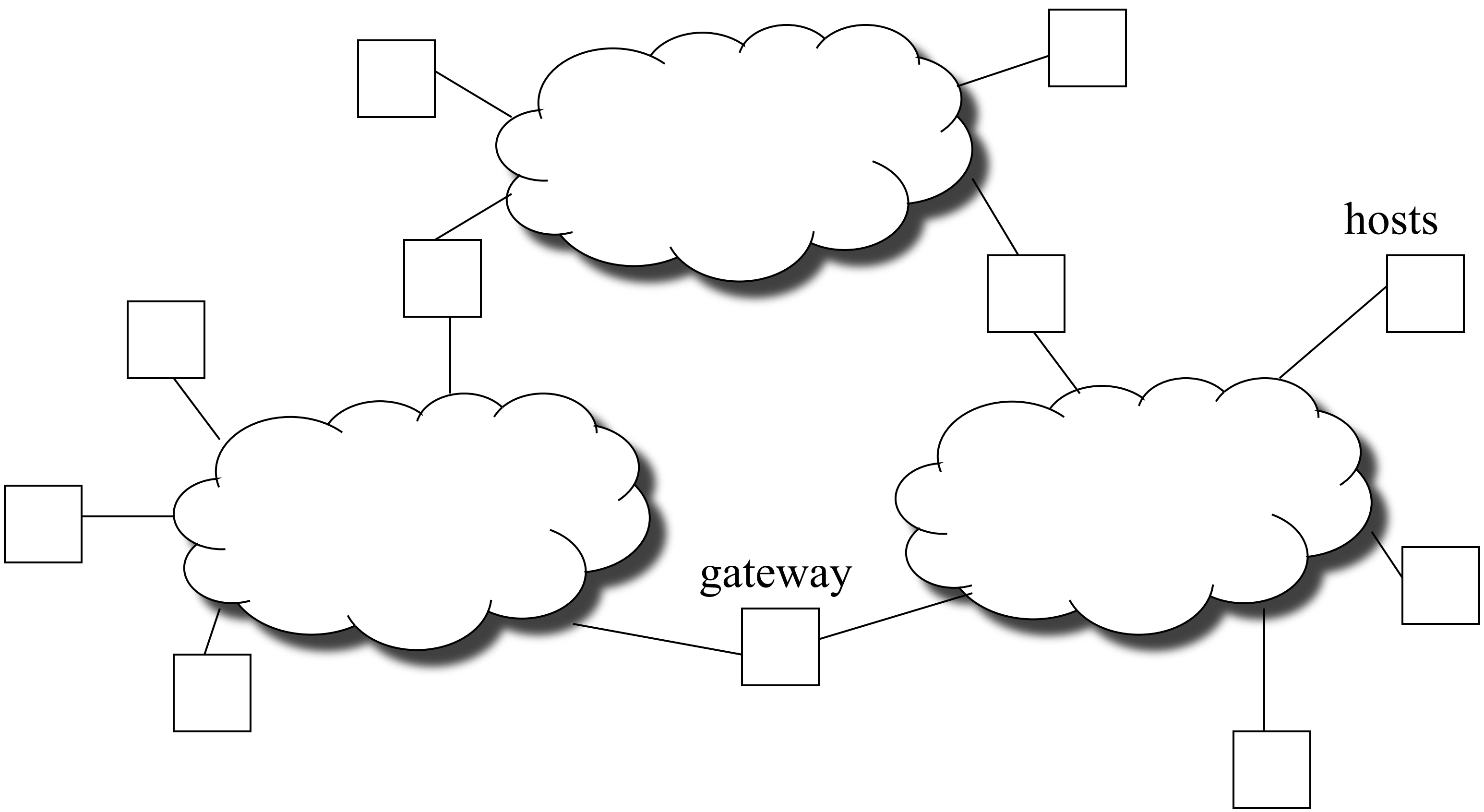
- (a) Point-to-point: *e.g.*, ATM
- (b) Multiple-access: *e.g.*, Ethernet
- Can't build a network by requiring *all* nodes to be directly connected to each other; need scalability with respect to the number of wires or the number of nodes that can attach to a shared medium

Switched Network



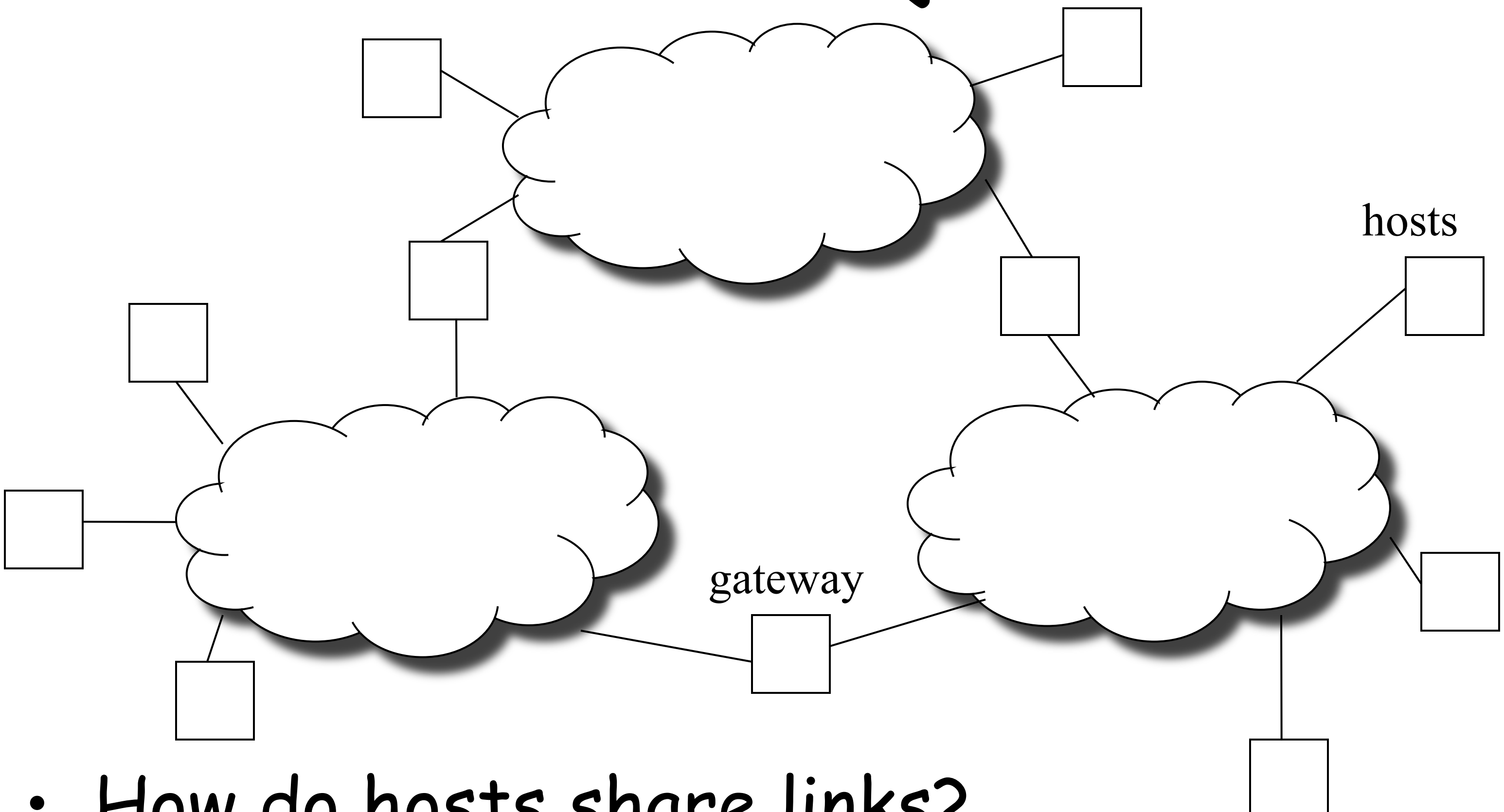
- Circuit switching vs. packet routing
- Hosts vs. “the network,” which is made of routers
- Nice property: scalable aggregate throughput

Interconnection of Networks



Recursively build larger networks

Some Hard Questions

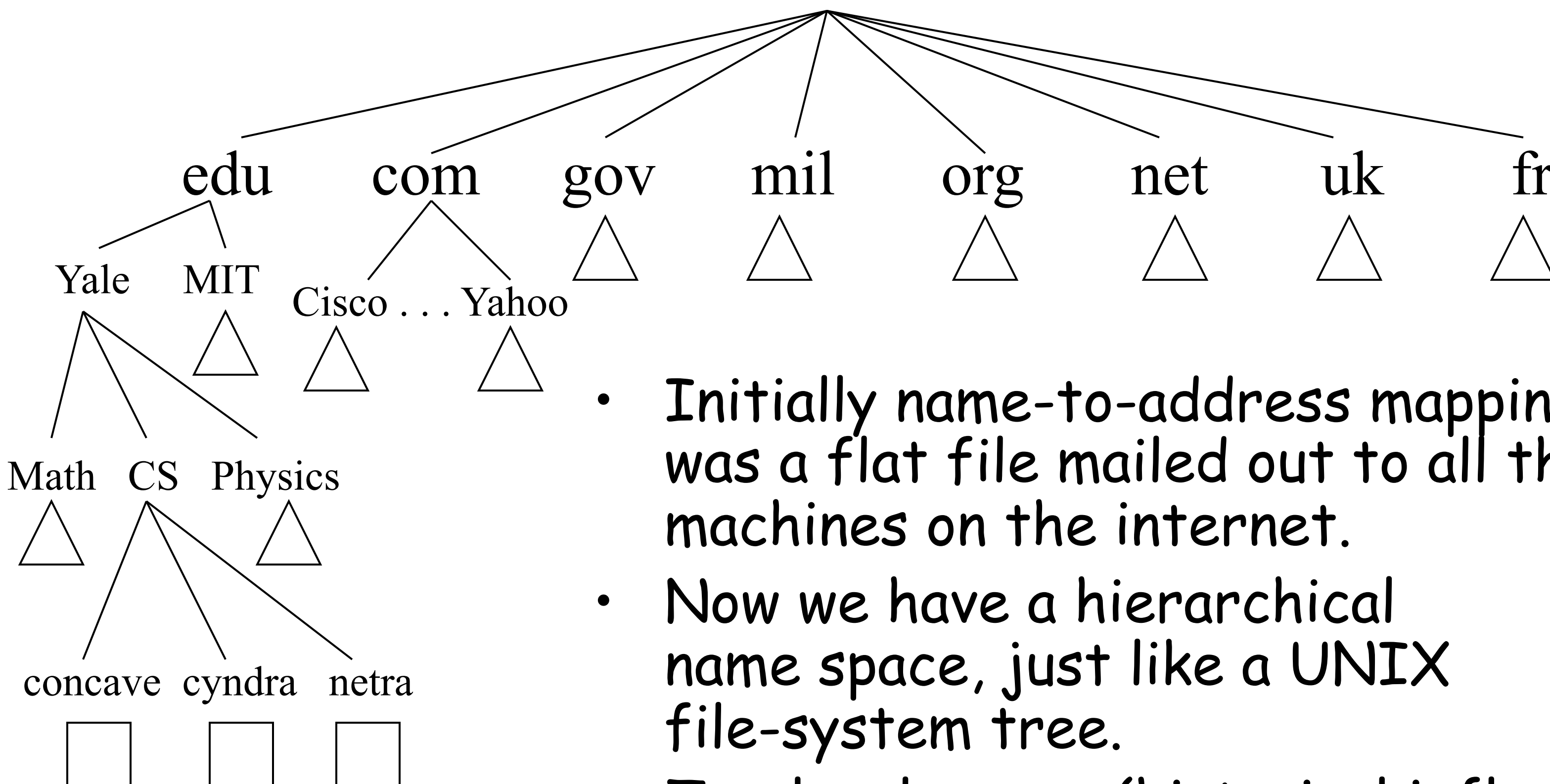


- How do hosts share links?
- How do you name and address hosts?
- Routing: given a destination address, how do you get to it?

IP Addresses and Host Names

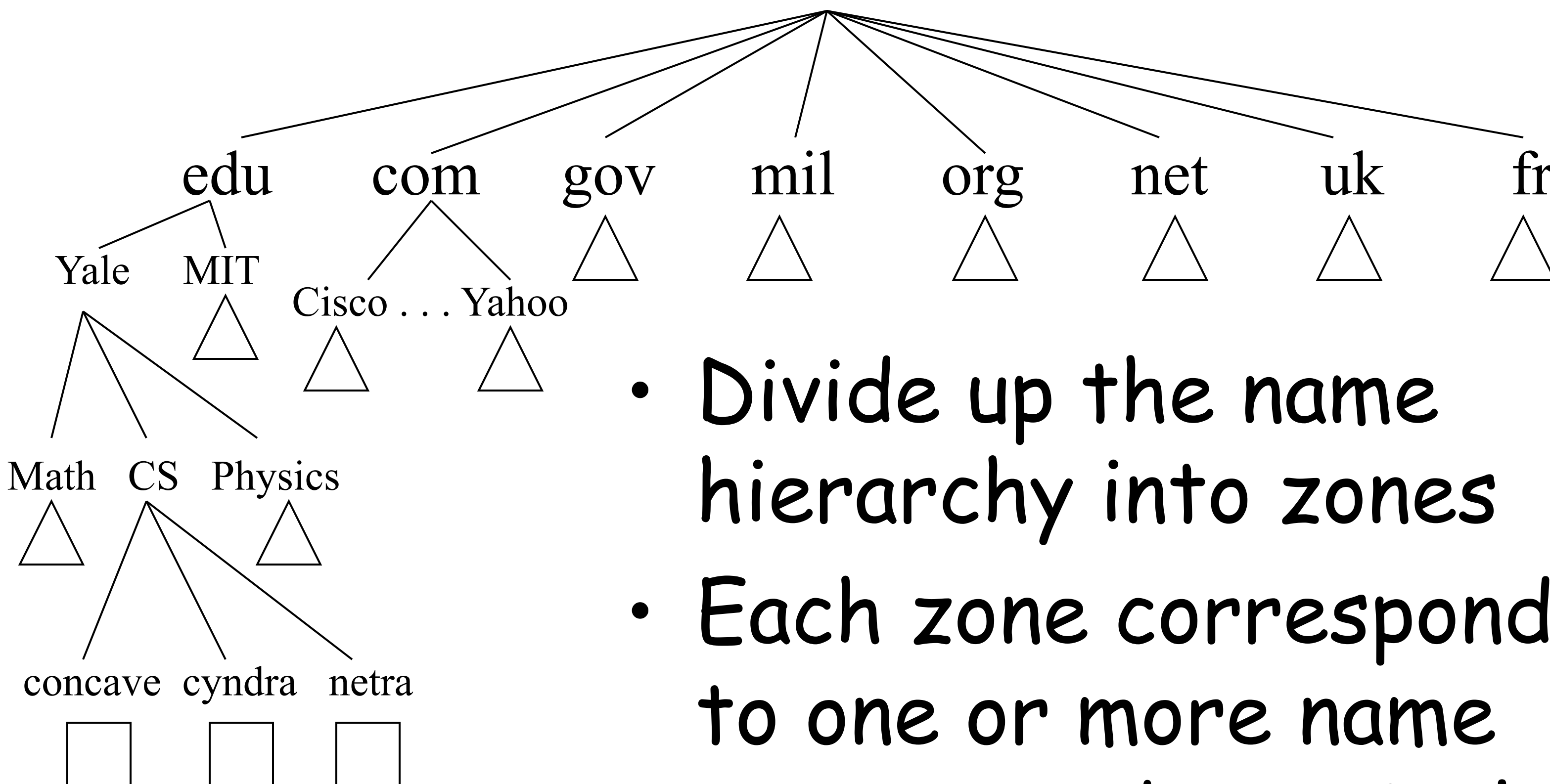
- Each machine is addressed by an integer, its **<u>IP address</u>**, written down in a “dot notation” for “ease” of reading, such as `128.36.229.231`
- IP addresses are the universal IDs that are used to name everything
- For convenience, each host also has a human-friendly host name. For example, `128.36.229.231` is `concave.cs.yale.edu.`
- Question: how do you translate names into IP addresses?

Domain Hierarchy



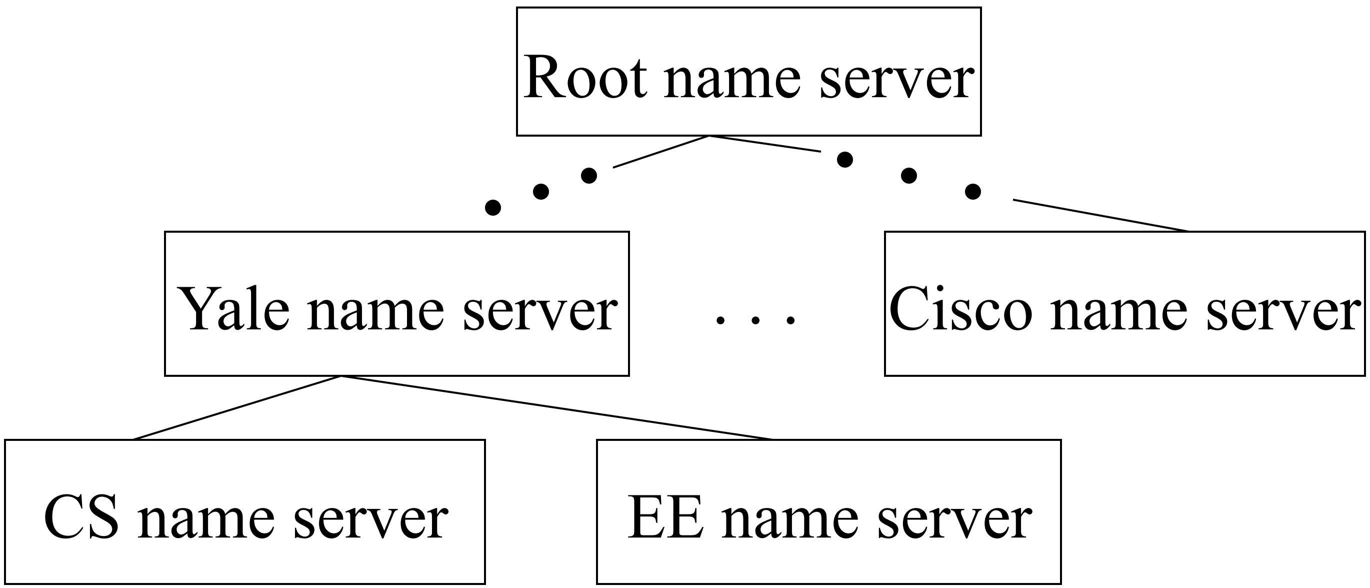
- Initially name-to-address mapping was a flat file mailed out to all the machines on the internet.
- Now we have a hierarchical name space, just like a UNIX file-system tree.
- Top-level names (historical influence): heavily US-centric, government-centric, and military-centric view of the world.

DNS Zones and Name Servers



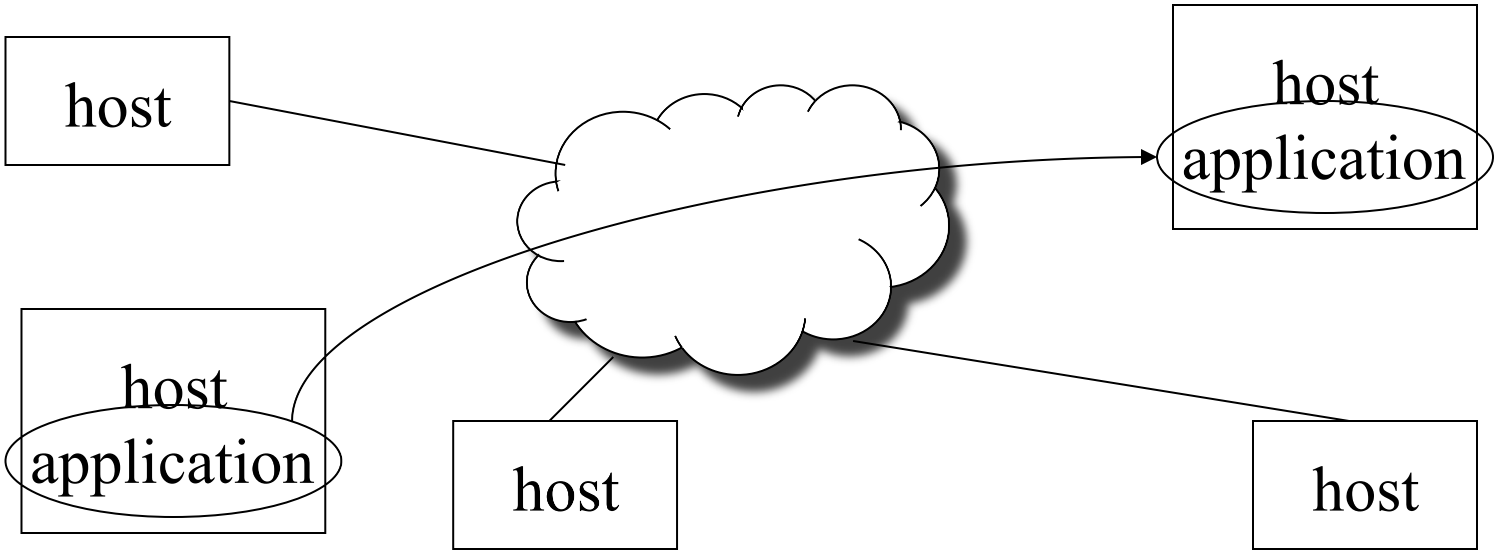
- Divide up the name hierarchy into zones
- Each zone corresponds to one or more name servers under a single administrative control

Hierarchy of Name Servers



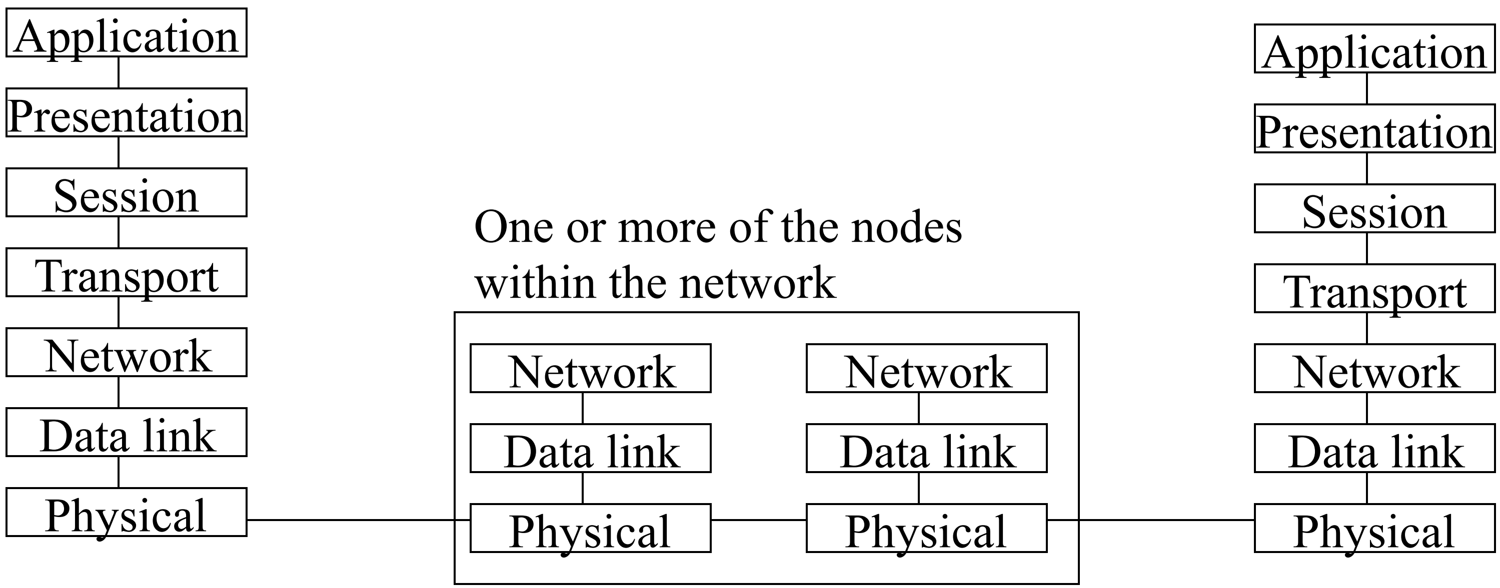
- Clients send queries to name servers
- Name servers reply with answers or forward request to other name servers
- Most name servers also perform lookup caching

Application-Level Abstraction



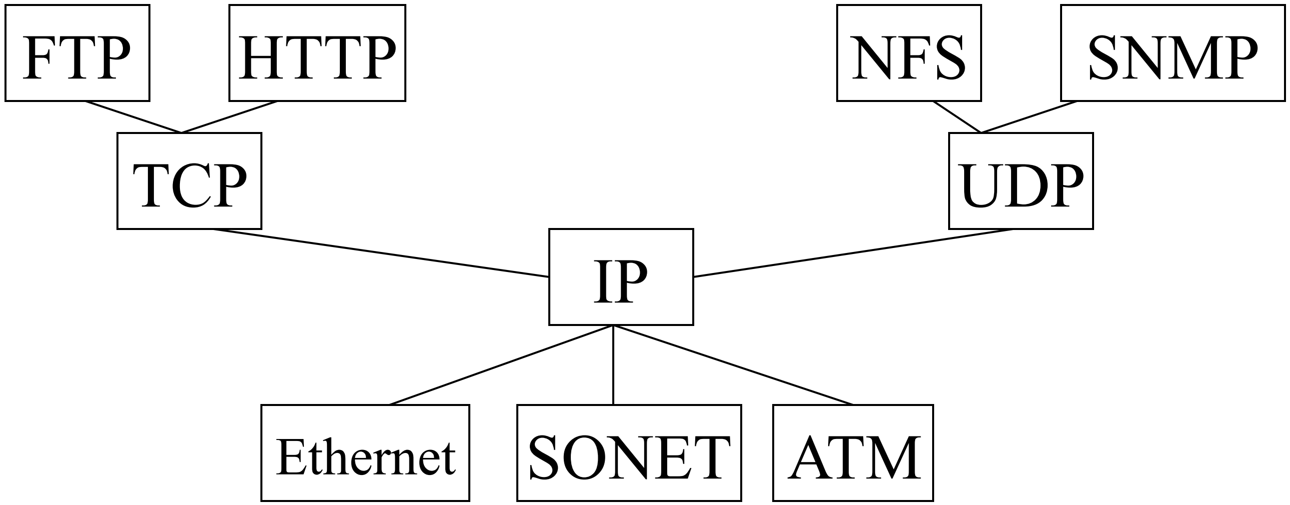
- What you have: hop-to-hop links, multiple routes, packets, can be potentially lost, can be potentially delivered out-of-order
- What you may want: application-to-application (end-to-end) channel, communication stream, reliable, in-order delivery

OSI Architecture



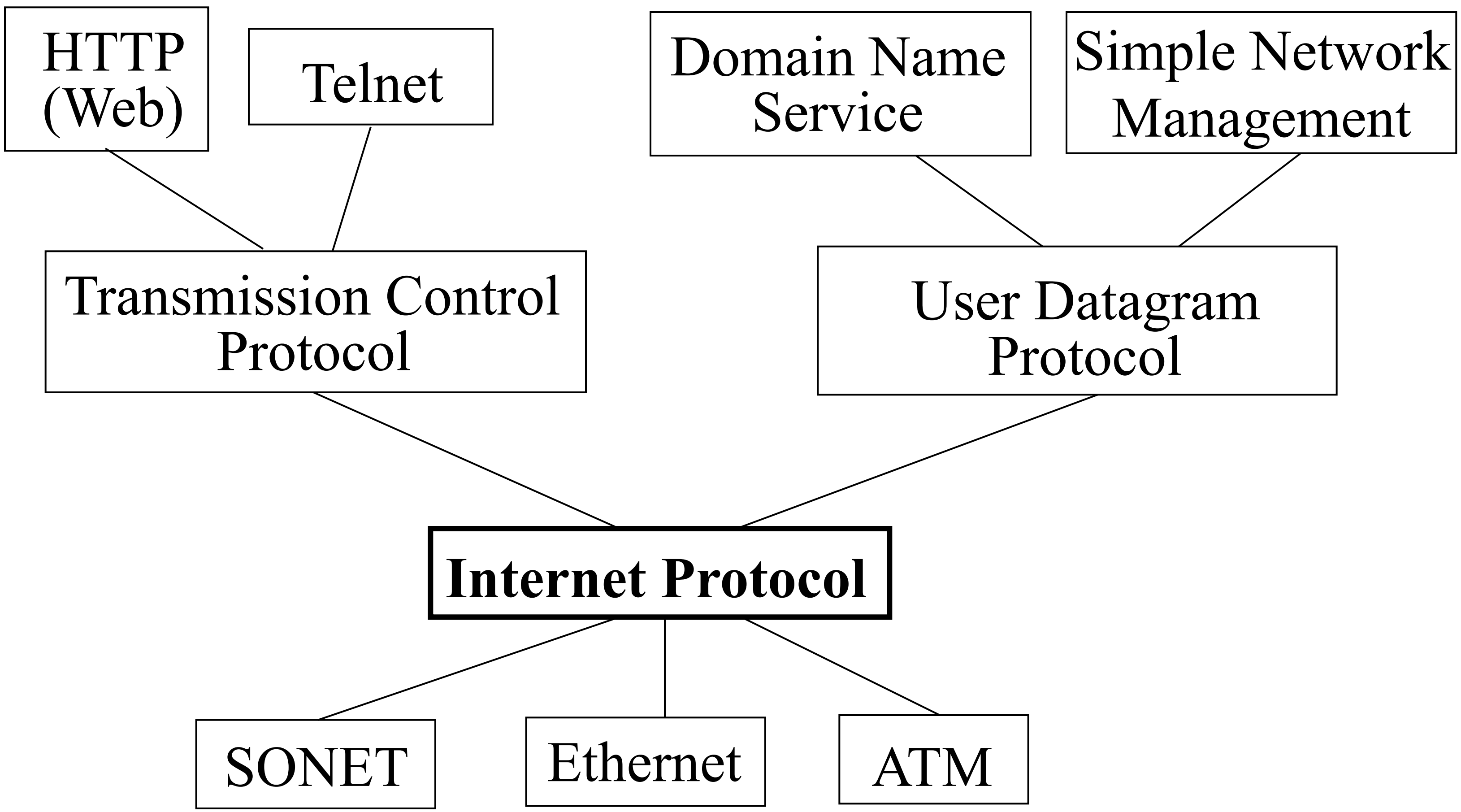
- Physical: handles *bits*
- Data link: provides "*frames*" abstraction
- Network: handles hop-to-hop routing, at the unit of *packets*
- Transport: provides process-to-process semantics such as in-order-delivery and reliability, at the unit of *messages*
- Top three layers are not well-defined, all have to do with application level abstractions such as transformation of different data formats

Reality: the "Internet" Architecture

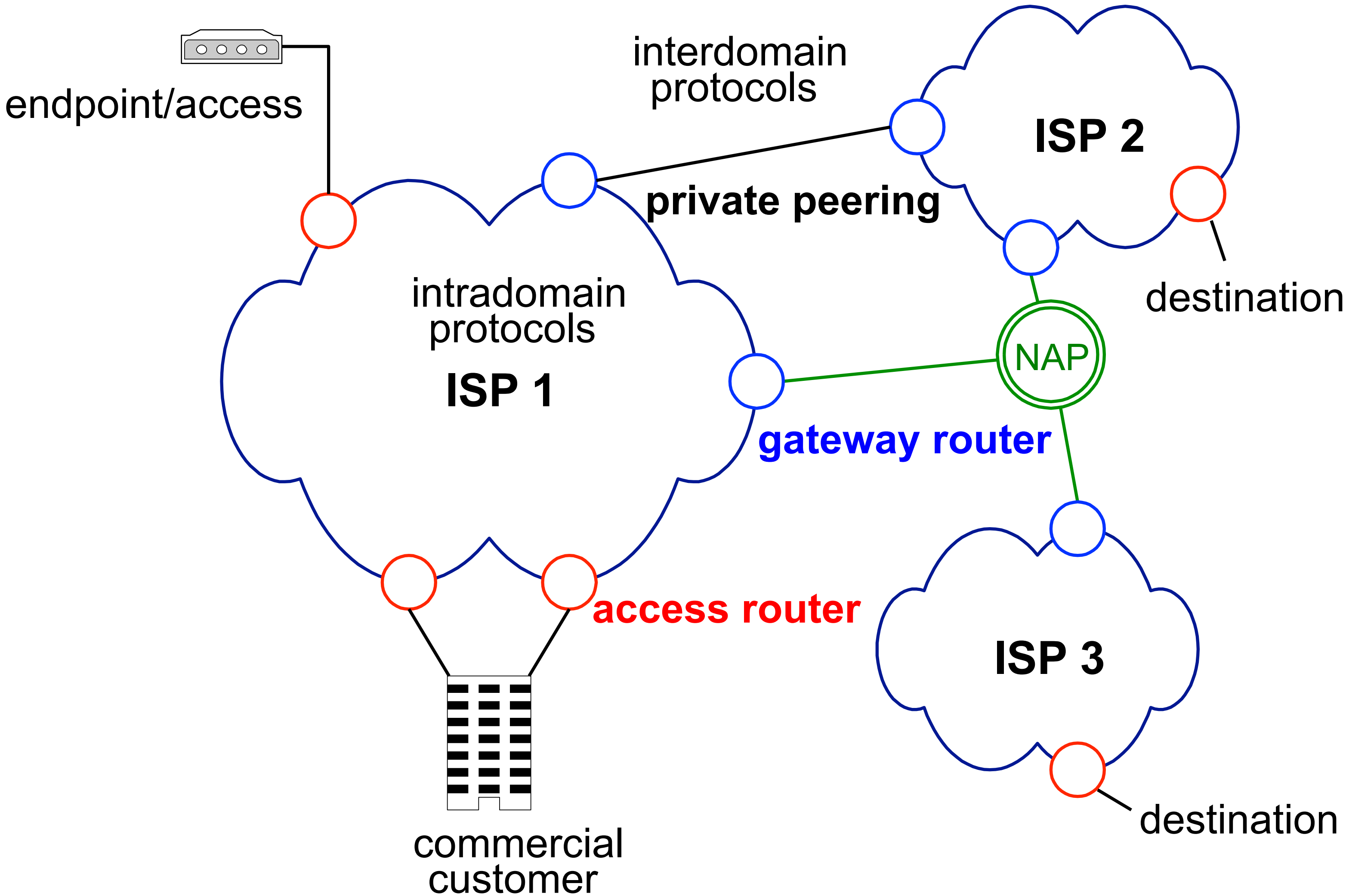


- Protocols: abstract objects that make up a layer
- Lowest level: hardware specific, implemented by a combination of network adaptors and OS device drivers
- IP (Internet Protocol): focal point of the architecture, provides host-to-host connection, defines common methods of exchanging packets
- TCP (transmission Control Protocol): reliable, in-order stream
- UDP (User Datagram Protocol): unreliable messages (maybe faster)
- On top of those are the application protocols
- Not strictly layered, "hour-glass shape," implementation-centric

Layering in the IP Protocols

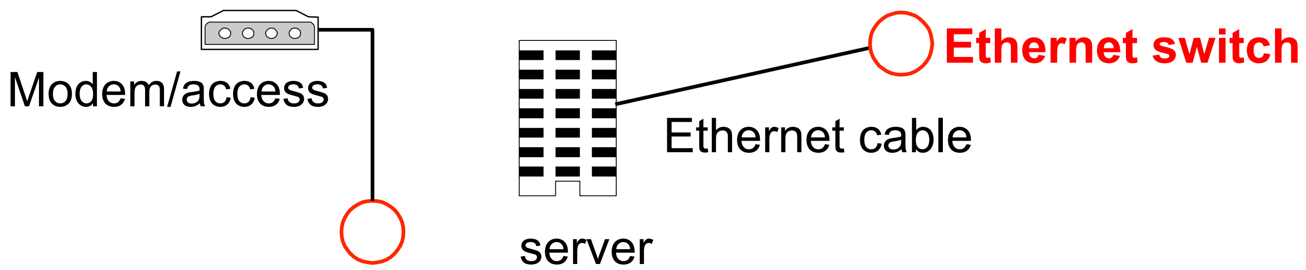


Internet Architecture



The Physical Layer

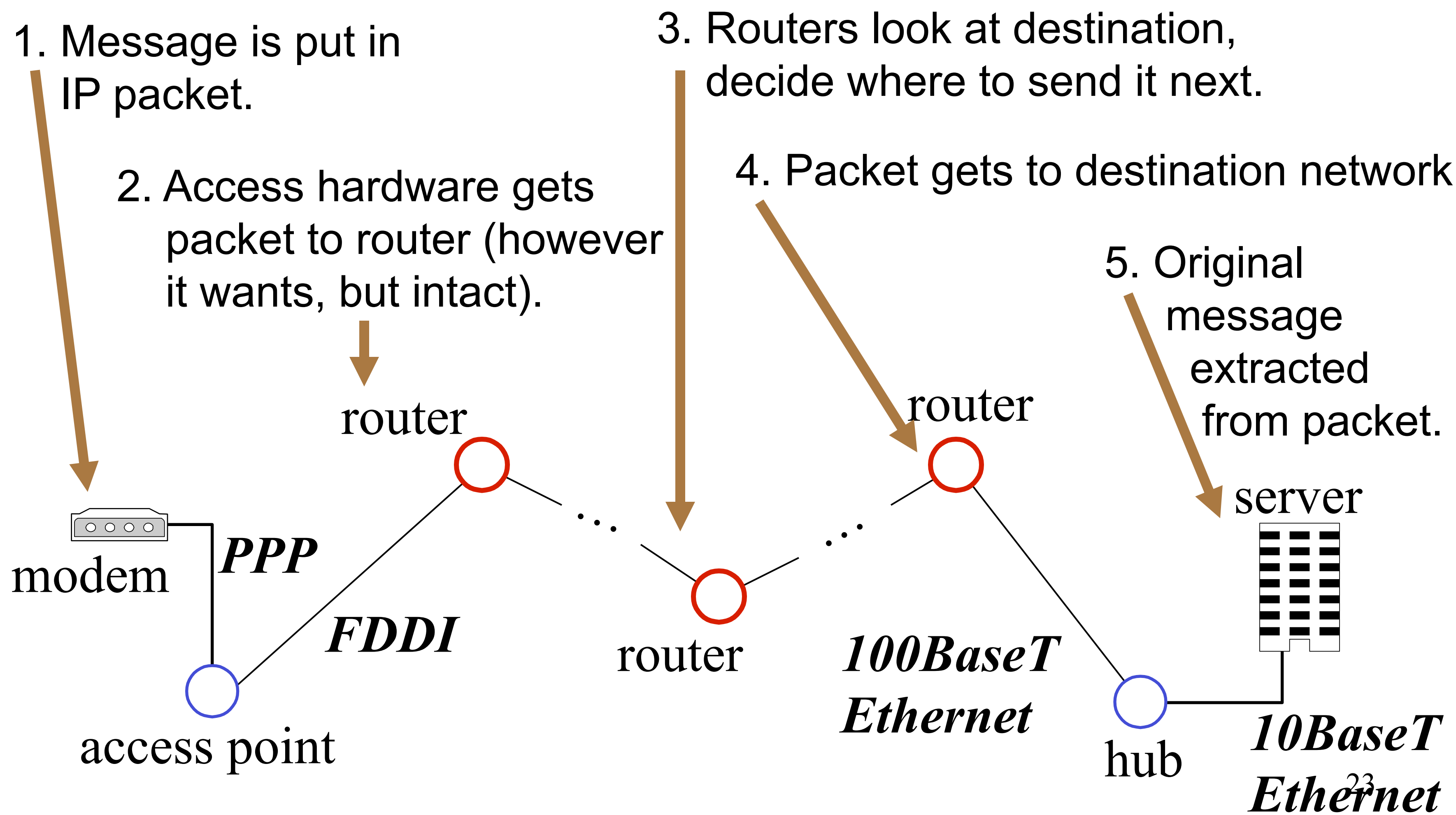
- A network spans different hardware.



- Physical components can work however they want, as long as the **interface between them is consistent**.
- Then, different hardware can be connected.

The Role of the IP Layer

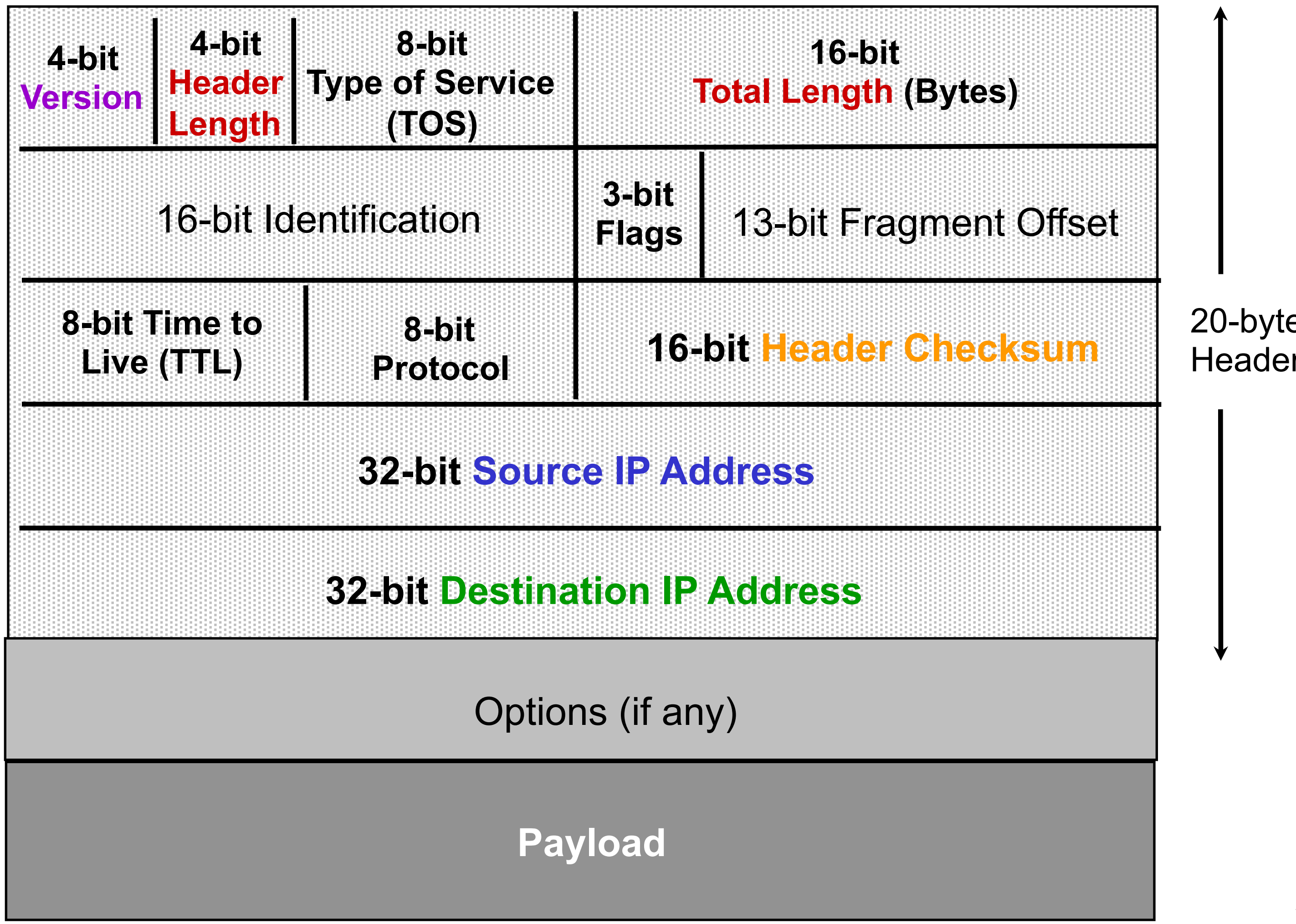
- **Internet Protocol (IP):** *gives a standard way to "package" messages across different hardware types.*



IP Connectionless Paradigm

- No error detection or correction for packet data
  - Higher-level protocol can provide error checking
- Successive packets may not follow the same path
  - Not a problem as long as packets reach the destination
- Packets can be delivered out-of-order
  - Receiver can put packets back in order (if necessary)
- Packets may be lost or arbitrarily delayed
  - Sender can send the packets again (if desired)
- No network congestion control (beyond “drop”)
  - Send can slow down in response to loss or delay

IP Packet Structure

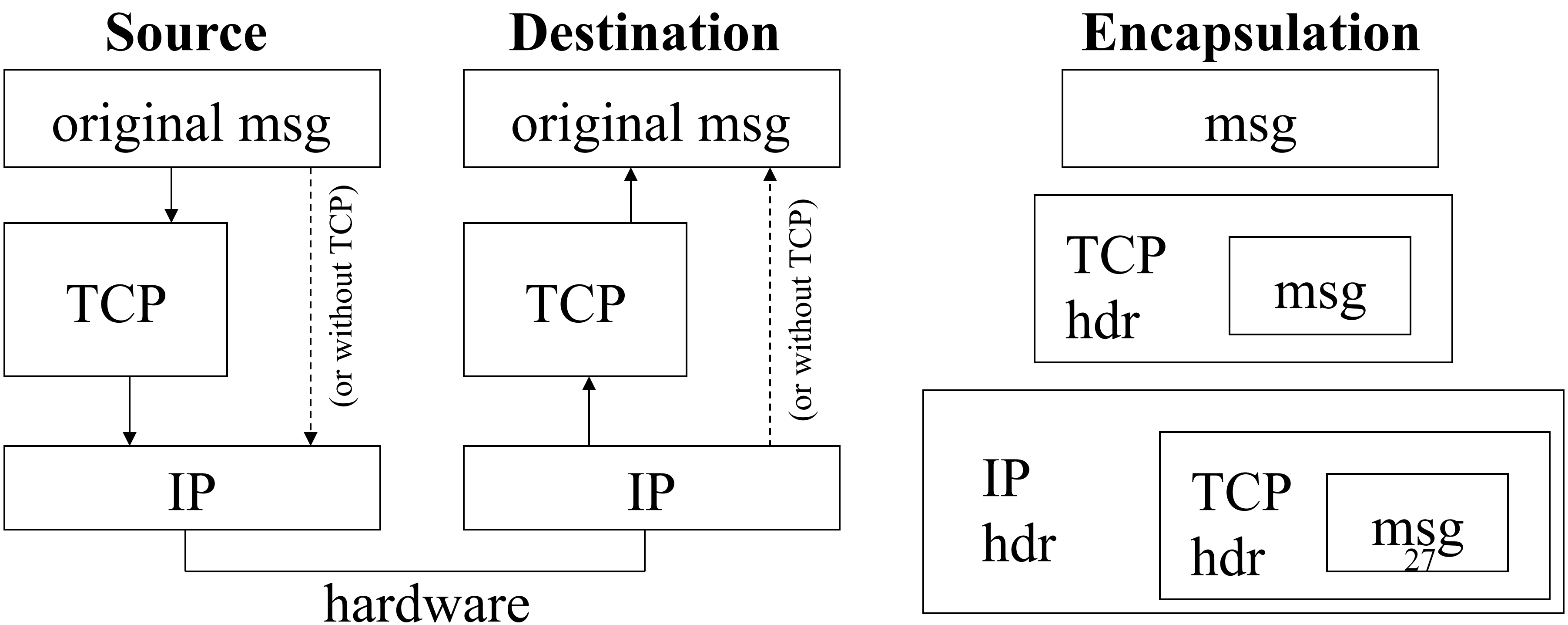


Main IP Header Fields

- **Version number** (e.g., version 4, version 6)
- **Header length** (number of 4-byte words)
- **Header checksum** (error check on header)
- **Source** and **destination** IP addresses
- Upper-level protocol (e.g., TCP, UDP)
- **Length** in bytes (up to 65,535 bytes)
- IP options (security, routing, timestamping, etc.)
- TTL (prevents messages from looping around forever; packets "die" if they "get lost")

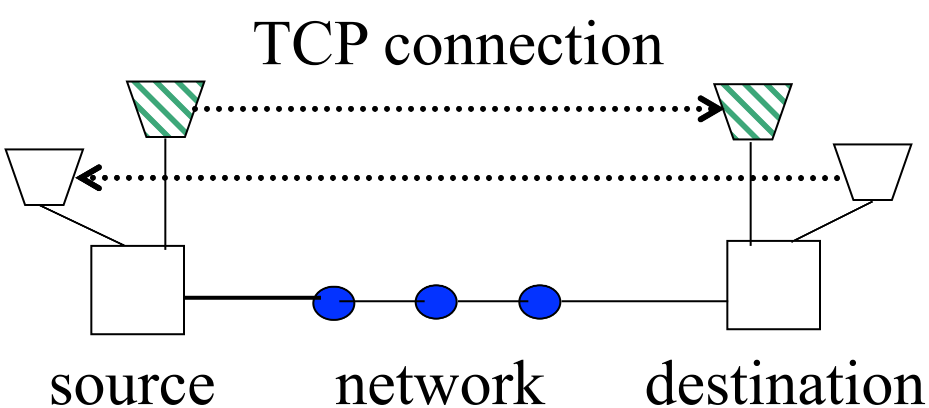
Adding Some Functionality

- More guarantees, *e.g.*, that packets go in order, require more work at both ends.
- Solution: add another layer (*e.g.*, TCP)

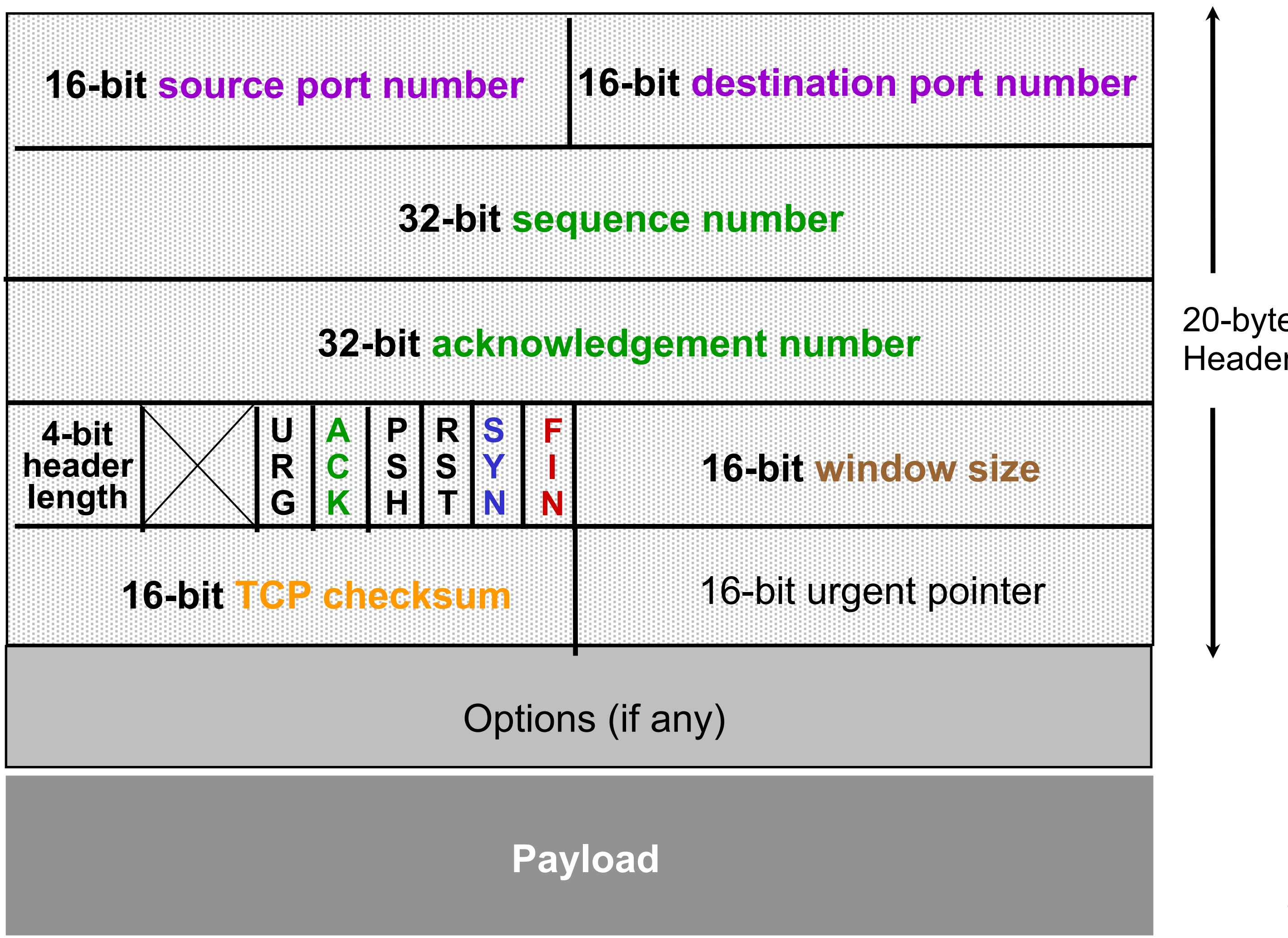


Transmission Control Protocol (TCP)

- Byte-stream socket abstraction for applications
- **Retransmission** of lost or corrupted packets
- **Flow-control** to respond to network congestion
- Simultaneous transmission in both directions
- **Multiplexing** of multiple logical connections

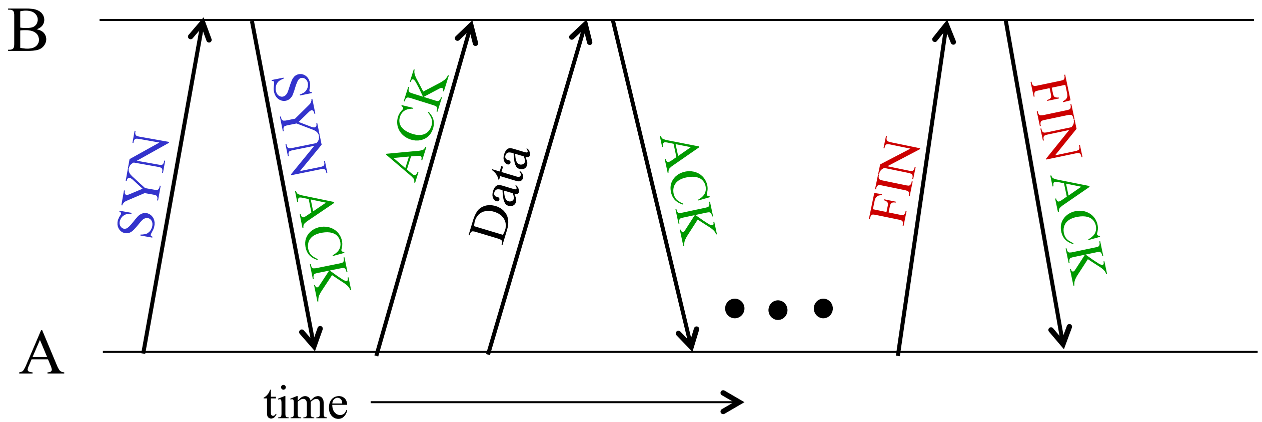


TCP Header



16-bit source port number
16-bit destination port number
32-bit sequence number
32-bit acknowledgement number
4-bit header length
URG
ACK
PSH
RST
SYN
FIN
16-bit window size
16-bit TCP checksum
16-bit urgent pointer
Options (if any)
Payload
20-byte Header

Establishing a TCP Connection



- Three-way handshake to establish connection
  - Host A sends a SYN (open) to the host B
  - Host B returns a SYN acknowledgement (ACK)
  - Host A sends an ACK to acknowledge the SYN ACK
- Closing the connection
  - Finish (FIN) to close and receive remaining bytes (and other host sends a FIN ACK to acknowledge)
  - Reset (RST) to close and not receive remaining bytes

Lost and Corrupted Packets

- Detecting corrupted and lost packets
  - Error detection via **checksum** on header and data
  - Sender sends packet, sets timeout, and waits for ACK
  - Receiver sends ACKs for received packets
- Retransmission from sender
  - Sender retransmits lost/corrupted packets
  - Receiver reassembles and reorders packets
  - Receiver discards corrupted and duplicated packets

Packet loss rates are high (*e.g.*, 10%), causing significant delay (especially for short Web transfers)!

TCP Flow Control

- Packet loss used to indicate network congestion
  - Router drops packets when buffers are (nearly) full
  - Affected TCP connection reacts by backing off
- **Window-based** flow control
  - Sender limits number of outstanding bytes
  - Sender reduces **window size** when packets are lost
  - Initial slow-start phase to learn a good window size
- TCP flow-control header fields
  - **Window size** (maximum # of outstanding bytes)
  - **Sequence number** (byte offset from starting #)
  - **Acknowledgement number** (cumulative bytes)

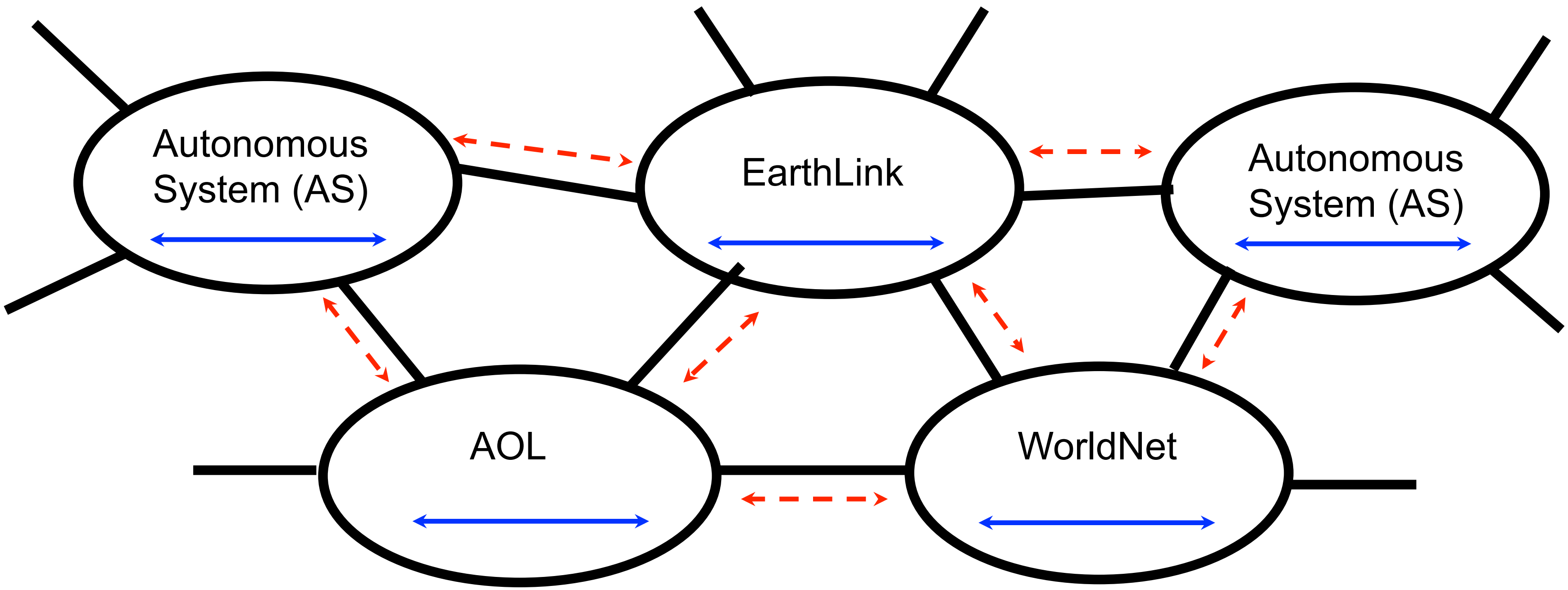
User Datagram Protocol (UDP)

- Some applications do not want or need TCP
  - Don’t need recovery from lost or corrupted packets
  - Don’t want flow control to respond to loss/congestion
- Fraction of UDP packets is rapidly increasing
  - Commonly used for multimedia applications
  - UDP traffic interferes with TCP performance
  - But, many firewalls do not accept UDP packets
- Dealing with the growth in UDP traffic
  - Pressure for applications to apply flow control
  - Future routers may enforce “TCP-like” behavior
  - Need better mathematical models of TCP behavior

Getting from A to B: Summary

- Need IP addresses for:
  - Self (to use as source address)
  - DNS Server (to map names to addresses)
  - Default router to reach other hosts (e.g., gateway)
- Use DNS to get destination address
- Pass message through TCP/IP handler
- Send it off! **Routers** will do the work:
  - Physically connecting different networks
  - Deciding where to next send packets **(HOW??)**

Connecting Networks



Autonomous System: A collection of IP subnets and routers under the same administrative authority.

Interior Routing Protocol (*e.g.*, Open Shortest Path First)

Exterior Routing Protocol (*e.g.*, Border Gateway Protocol)

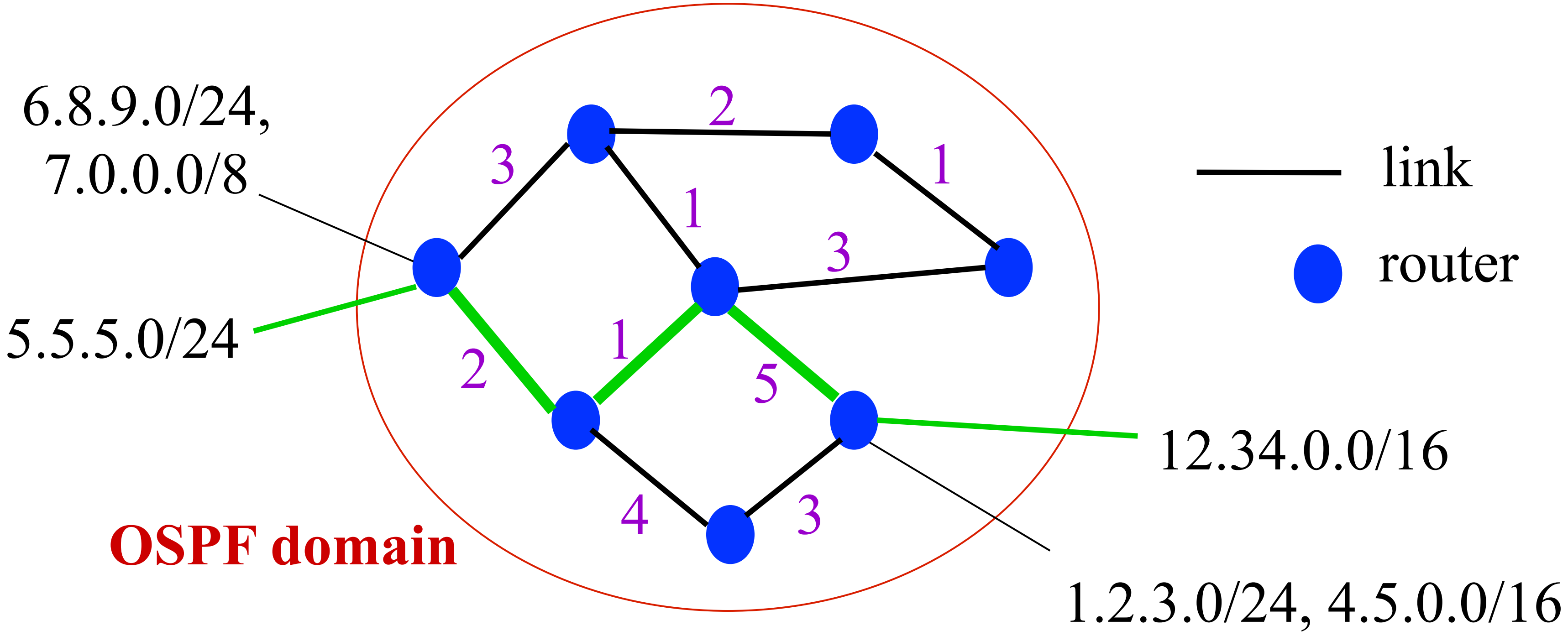
Where to Go Next

- Routers contain a **forwarding table** that pairs destination with next hop (on what physical wire to send msg.).
- The table gets populated with information learned **internally** (*e.g.*, OSPF) and **externally** (*e.g.*, BGP).
- OSPF and BGP are protocols that communicate ***knowledge about destinations*** between routers.

Open Shortest-Path First (OSPF) Routing

- Network is a graph with **routers** and **links**
  - Each unidirectional link has a **weight** (1-63,535)
  - **Shortest-path** routes from sum of link weights
- **Weights** are assigned statically (configuration file)
  - Weights based on capacity, distance, and traffic
  - Flooding of info about weights and IP addresses
- Large networks can be divided into multiple **domains**

Example Network and Shortest Path



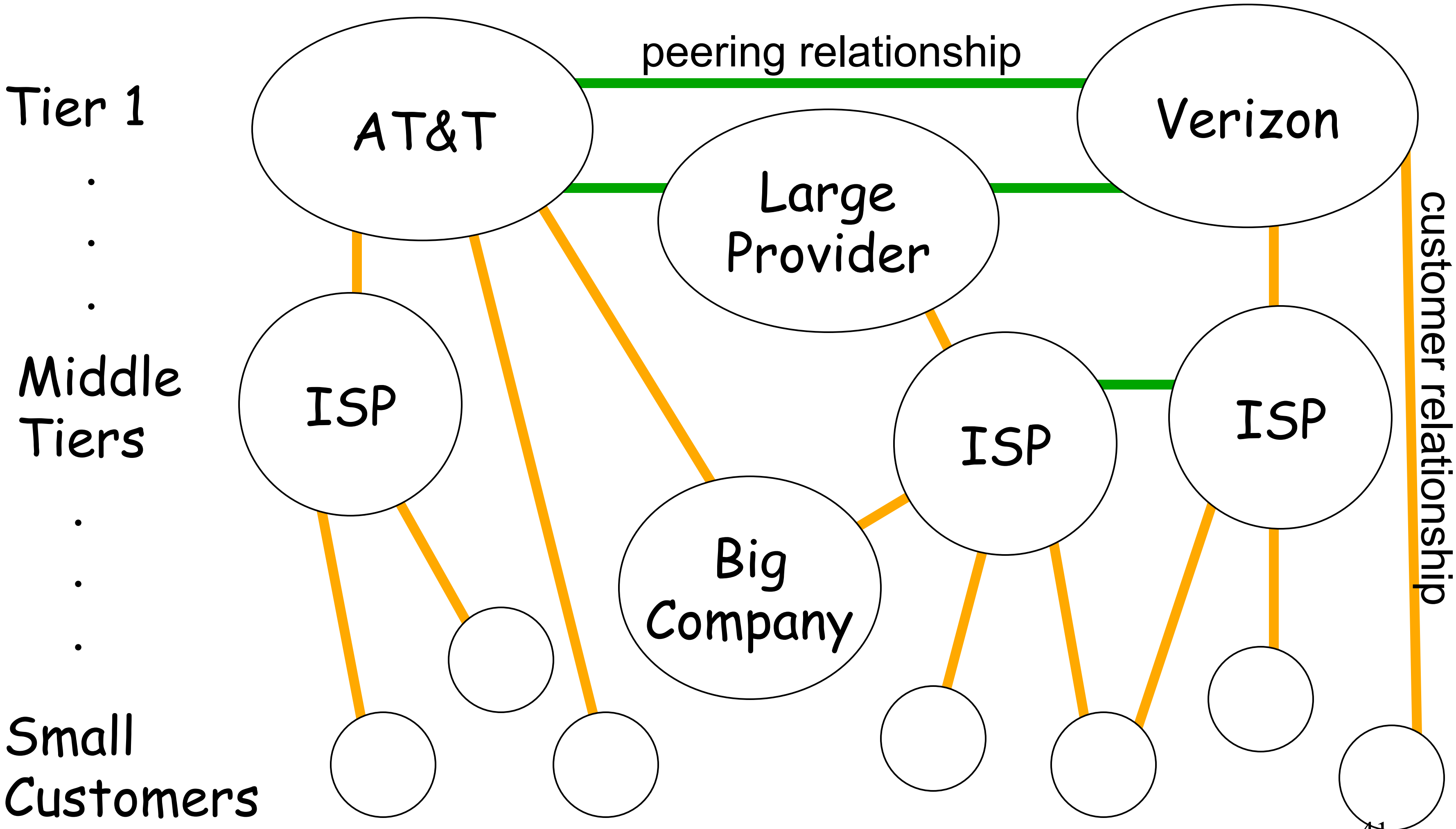
IP Routing in OSPF

- Each router has a **complete view of the topology**
  - Each router transmits information about its links
  - Reliable flooding to all routers in the domain
  - Updates periodically or on link failure/installation
- Each router computes **shortest path(s)**
  - Maintenance of a complete link-state database
  - Execution of Dijkstra's shortest-path algorithm
- Each router constructs a **forwarding table**
  - Forwarding table with next hop for each destination
  - Hop-by-hop routing independently by each router

OSPF Won't Work Between Domains

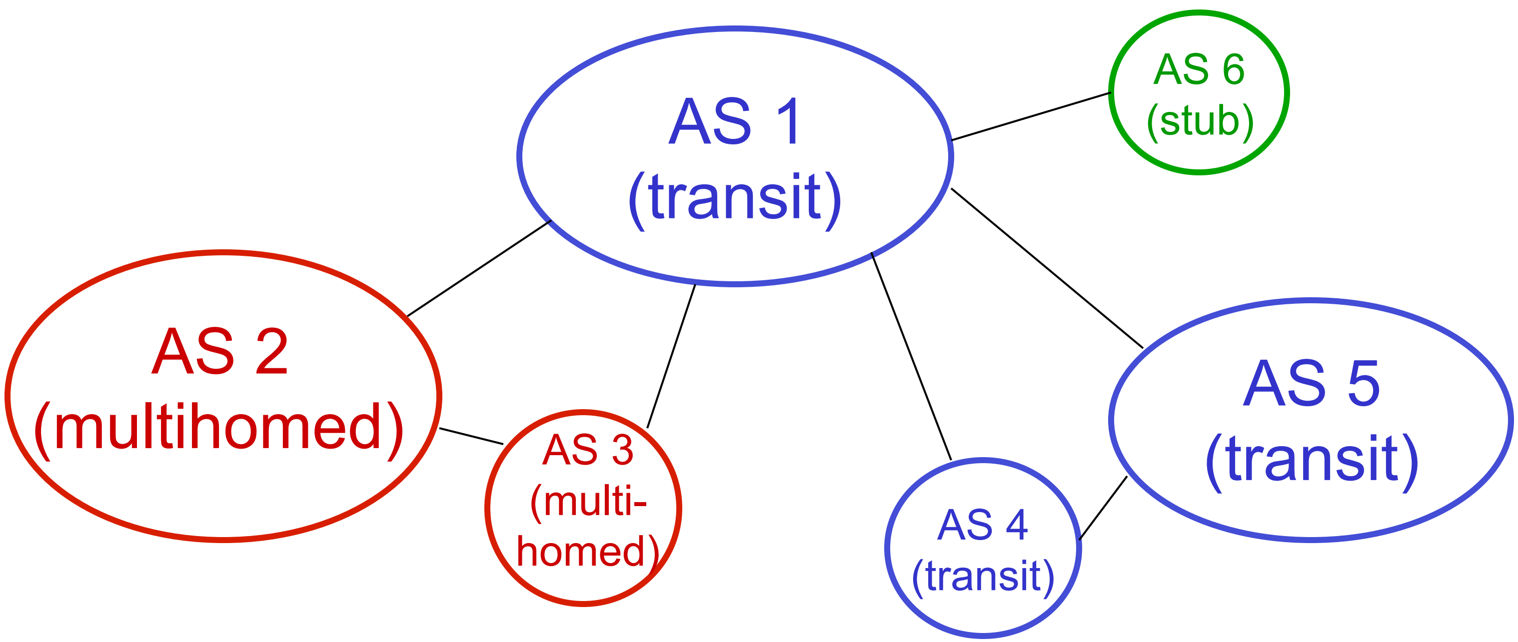
- OSPF nodes are managed by the same authority. They have a common goal (find shortest path).
- Domain is small enough that nodes can flood each other with information.
- Across companies, ***business relationships*** determine routing policy. More complicated!

Business Relationships Connect the Internet



Border Gateway Protocol (BGP)

- BGP routes traffic through a network where the AS's can be connected in any way.
- Three types of AS's: **stub** (local traffic only); **multihomed** (multiple connections but local traffic only); **transit** ("thru" and local traffic).



Border Gateway Protocol (BGP) Concepts

- **Reachability:** from one AS, what other AS's can be reached from it?
- Every AS has a **BGP Speaker** node that advertises its reachability info by sending **complete paths** to reachable networks.
- Given advertised updates, we calculate **loop-free** routes to networks.
- Problem of scale: too many networks; don't know how an AS works, so it's hard to determine cost to send through each.

BGP Preferences

- Nodes have to choose a path from all those advertised by their neighbors.
- BGP table contains all the collected routes and their **local preference**.
- Choose route with highest rank.
- How to set rank?
  - Based on routing policy: prefer customers first, then peers, then upstream providers.
  - Other factors? Geography, special agreements with neighbors.