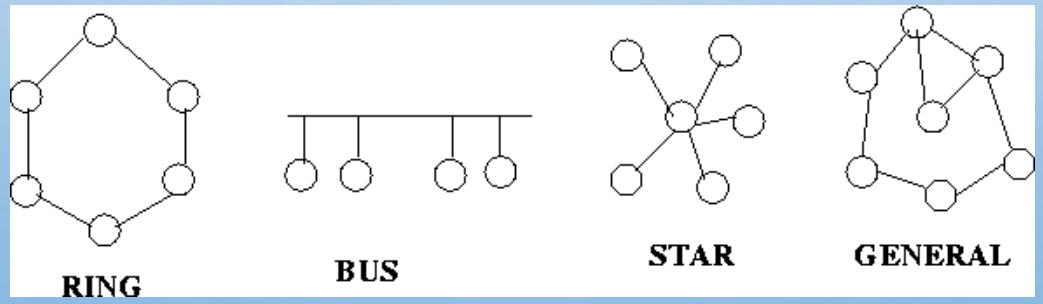
LECTURE 8 INTRODUCTION TO NETWORKING

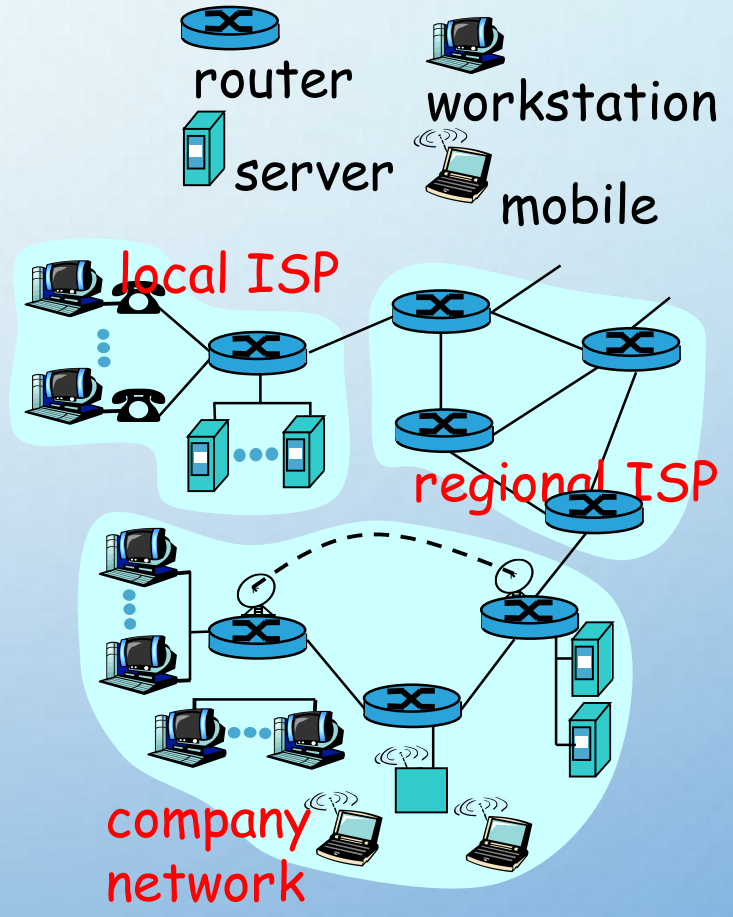
What is a Computer Network

- A computer network consists of a set of end-systems/hosts and switches/routers connected by communication links. It supports communications among entities in the network.
  - Many types
    - ❖ network on chip, system area network, person area network, local area network, data center network, wide area network
  - Many topologies (how the entities are connected in the network)



Elements of a Network

- Hosts, end-systems
  - pc's, workstations, servers
  - PDA's, phones, toasters
  - CPUs, memory controllers
  - running network applications
  - Operate at the edge of the network
- Routers, switches, and middleboxes:
  - Forward/filter packets (chunks) of data thru network
  - Operate at the core of the network
- Communication links
  - Point-to-point, multiaccess
  - fiber, copper, radio, satellite

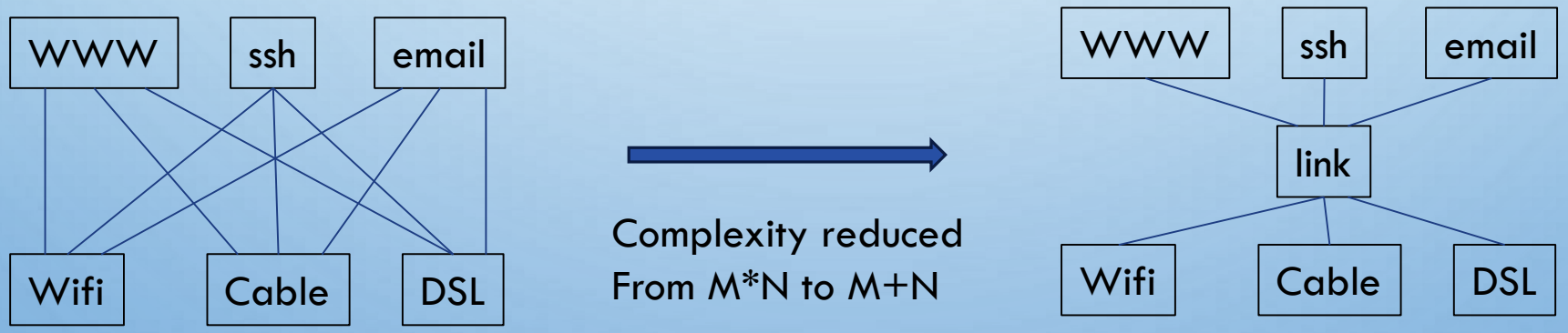


Challenges in the Networking System

- What problems need to be solved for us to achieve the Internet as it is today?
  - Dealing with heterogeneity in the networking environment
    - ❖ different types of media: wired, wireless, satellite, radio, etc
    - ❖ Dealing with different types of networks: telephone, data, ATM, fiber channel, Cable network, DSL
  - Complex network functionality
    - ❖ Reliability, routing, network congestion, speed-mismatch between communicating parties (flow control), naming and addressing, fragmentation and reassembly
- What are the common techniques in computing to deal with complexity and heterogeneity in Computer Science?

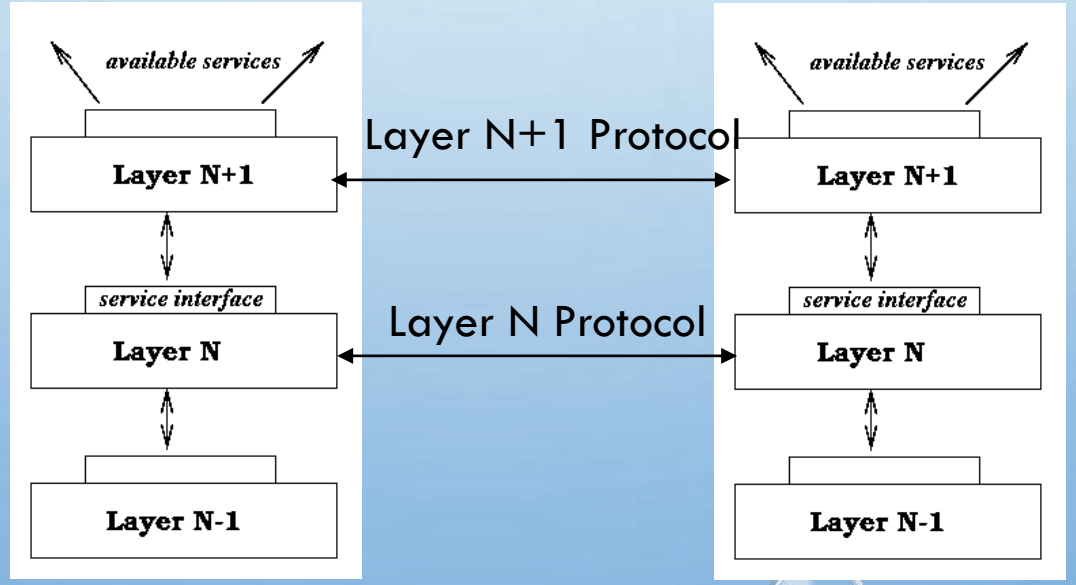
Challenges in Networking Software

- What the common techniques in computing to deal with complexity and heterogeneity?
  - **Divide-and-conquer**: partitioning the software to smaller pieces such that each piece becomes more manageable.
  - **Abstraction**:
- Result: **Layered architecture** in networking software



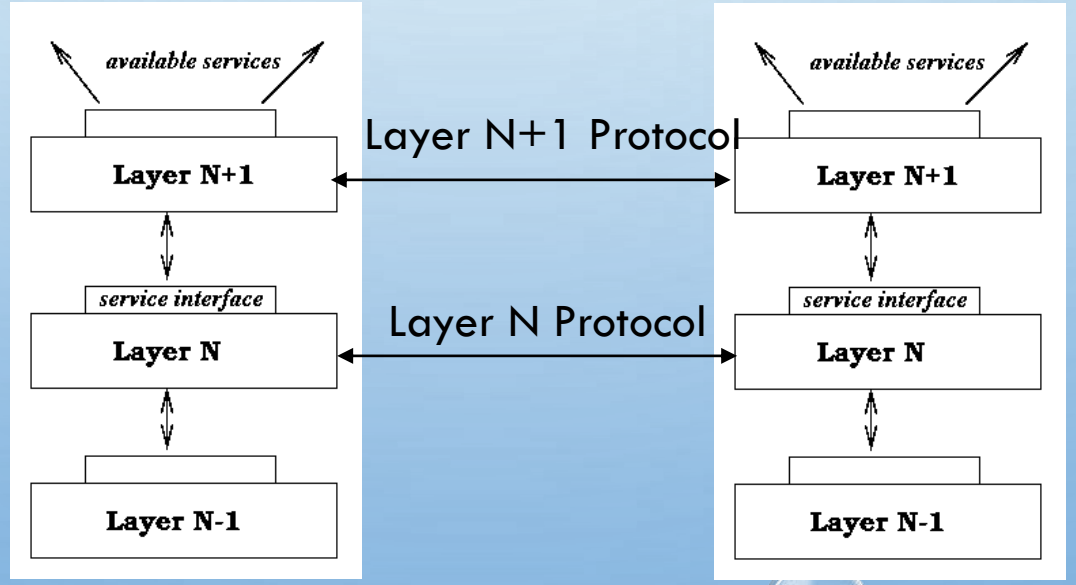
Layered Architecture

- *Layering* simplifies the architecture of a complex system
- Layer N relies on *services* from layer N-1 to provide a *service* to layer N+1
- *Interfaces* define the services offered
- Service required from a lower layer is independent of the implementation
  - Layer N change doesn't affect other layers
  - Information/complexity hiding



Layered Architecture

- Entities in the same layer on different hosts interact (**peer entities**) through a **protocol**.
  - An entity can be an IO chip, a process, a procedure, etc
  - A protocol governs how communication should happen between entities in the same layer on different hosts.
- Entities on the same machine between two adjacent layers interact through the service interface.



Type of Communication Services

- Connection-oriented service & connectionless service -- order of packets
  - Connection-oriented -- like the telephone.
    - Establish the connection, use the connection, then release the connection. Even when multiplexed, minimal header information
    - The packets received are in the same order as the packets sent.
  - Connectionless (datagram) -- like the postal system.
    - Each message carries its destination's address and is routed to the destination independently.
    - The packets received may not be in the same order as the packets send.
- Reliable and unreliable – loss or no loss of packets
  - reliable: all packets sent are received correctly
  - unreliable: packets sent may not be received

Type of Communication Services

- Four combinations of service types:
  - Reliable connection-oriented
  - Reliable connectionless
  - Unreliable connection-oriented
  - Unreliable connectionless
- Example:
  - send 1 2 3 4 5
  - receive: 1 2 3 4 5
    1 3 2 5 4
    1 2 4
    3 1 4
- Many applications use reliable connection-oriented service. Why not just do reliable connection-oriented all the time?

ISO/OSI Reference Model

- ISO: International Standards Organization
- OSI: Open Systems Interconnection.
- Seven layers ISO/OSI reference model:

Application		Application
Presentation		Presentation
Session		Session
Transport		Transport
Network	Network	Network
Data Link	Data link	Data link
Physical	Physical	Physical

7-Layer ISO/OSI Reference Model

- Physical layer: how to transfer bits over a physical link
  - conversion of bits into signals, what is 0, 1? How long does a bit lasts? How many pins in a connector?
- Data link layer: how to transfer frames between directly connected entities
  - A frame is a sequence of bits as one unit
- Network layer: how to send a packet to the destination that may not be directly connected (routing)
- Transport layer: how to achieve end to end communication.
  - Lowest layer with end-to-end communication, not run on routers.

7-Layer ISO/OSI Reference Model

- Session layer: allows users to establish session, enhanced services.
  - Checkpointing.
- Presentation layer: provides general solutions to users.
  - Compression, syntax conversion, cryptography
- Application layer: variety of protocols that are commonly used.
  - All network applications.

TCP/IP Reference Model

- The 7-layer OSI/ISO reference model is well designed but does not have real implementation.
  - Internet had already been built when the 7-layer model was developed. Internet does not follow this 7-layer model.
- TCP/IP reference model:

  Application layer (ssh, FTP SMPT, DNS, NNTP, HTTP) --- application, presentation, session

  Transport layer (TCP, UDP)

  Internet layer (IP) – network

  Host to Network layer (Ethernet, FDDI, X.25) -- data link, physical

TCP/IP Reference Model

Application layer (HTTP, SSH, FTP SMPT, DNS, NNTP)

Interface – socket programming: (socket, TCP_send, TCP_recv, UDP_send, UDP_recv) ------- OS interface (system calls)

Transport layer (TCP, UDP) -- performed at end hosts

- Major function: end-to-end communicate
- TCP (transmission control protocol): reliable connection-oriented service
- UDP (user datagram protocol): unreliable connectionless service

Interface: (IP_send, IP_recv)

Internet layer (IP) – unreliable connectionless service (best-effort service) -- performed at routers and end-hosts

- Major function: Routing – packet can reach destination multiple hops away
- A packet switching network based on connectionless communication. Hosts send packets into the network and then the packets travel independently to their destinations.
- Format conversion: for different networks.
- Packet format and protocol: IP

Interface – network dependent, Ethernet: (Eth_send, Eth_recv) ---- HW/SW interface

Host to Network layer (Ethernet, FDDI, X.25)

- Major function: Communication between directly connected devices
- Undefined, rely on the existing technology - must be able to send IP packets over this layer.

Networked Applications

- Application layer --- this is where network applications are!

Above OS, where you write your program

Interface – socket programming (OS system call interface)

Transport layer (TCP, UDP)

Internet layer (IP)

Host to Network layer (Ethernet, FDDI, X.25)

Below OS, either OS software or hardware (normally, you do not touch these layers).

- Network applications: Use TCP or UDP (through socket programming) to perform communication between programs (application) on different machines.
  - A network application usually has its own protocol, which decides how the programs in the application will communicate.

Networked Applications

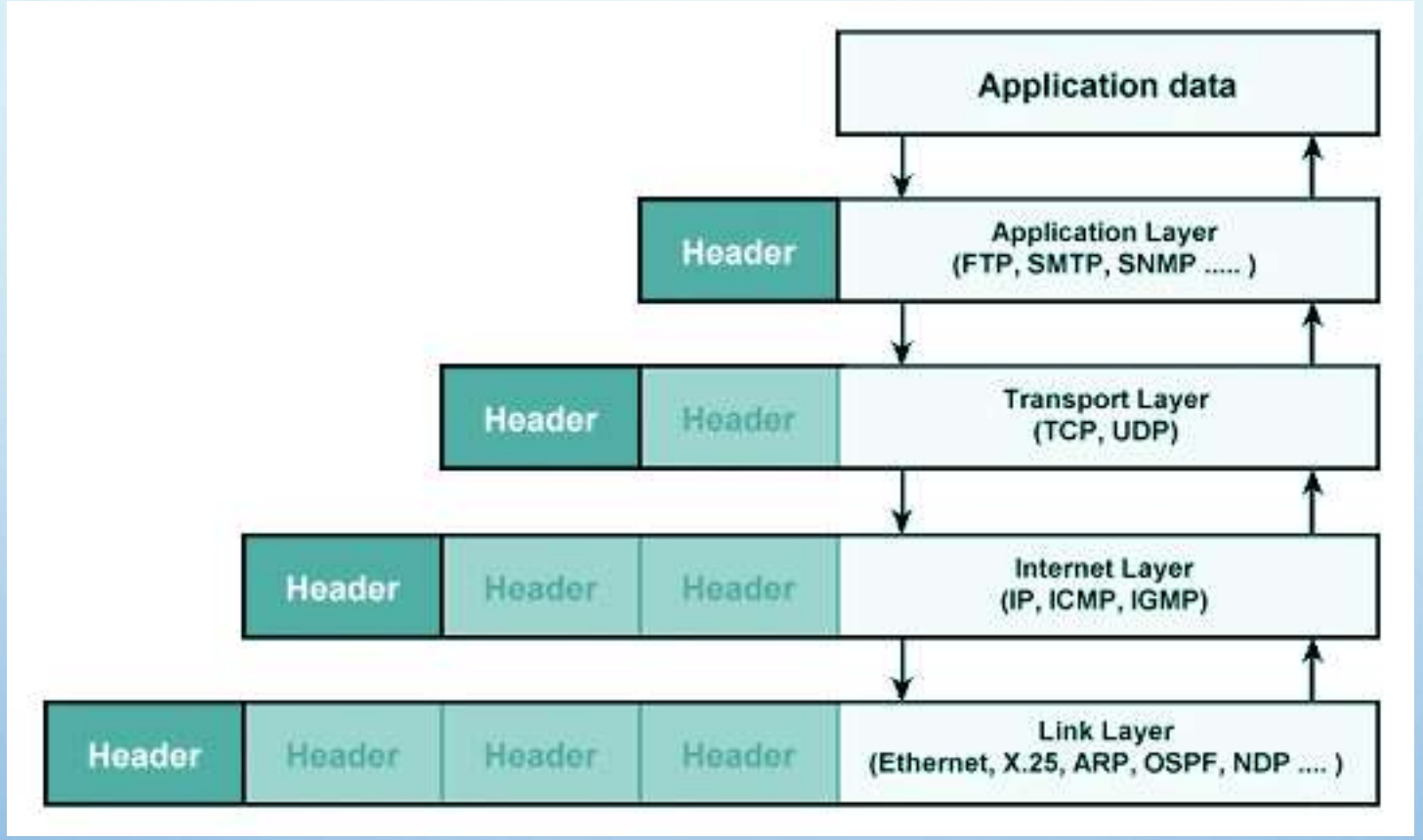
- Application layer --- this is where network applications are!
  - Need to have a way to identify another service/application on another machine
    - This is done through the pair: (IP address, port number)
  - IPv4 addresses are 32-bit binary numbers
    - Often represented as dotted decimal notation – 4 decimal values, each representing 8 bits in the range 0 to 255
      - Example linprog1.cs.fsu.edu, 128.186.120.188
    - IPv4 is supposed to be replaced by IPv6, but is still going strong
  - IPv6 addresses are 128 bits binary numbers
  - Some machines have both IPv4 and IPv6 addresses

Identifying the service (process) on a computer: Port number

- The port number is a 16-bit number
  - 0 to 1023 are reserved ports to well known services
  - Example: ssh (22), http(80)
- 1024-65535 are ephemeral ports that you can use for your application.
  - For the security reason, many ports on most of machines are blocked by the firewall, so to open to public, you need to work with the system administrator.
  - On linprog, use port 50000 and above for your application.
- You can try to connect to a service using telnet from linprog
  - Try 'telnet www.cnn.com 80' and then 'GET / HTTP/1.1'.

Passing data through TCP/IP layers

Sender



Receiver

Each layer adds a header to the data. E.g. port number is in the transport layer header, IP address is in the Internet header. Headers introduce overheads

Hosts/end points run all 4 layers; routers/middleboxes may only have 2 layers (Internet and Link layer).

How are some key issues addressed on the Internet?

- Unlike a machine, the Internet cannot be rebooted: whatever is introduced on the Internet should work all the time, under any condition!

  - Reliability
  - Routing
  - Decide the data rate to send

Reliability

- Reliability: whatever sent by the sender is whatever received by the receiver
  - When the receiver receives a packet, it needs to have the ability to decide whether the data is corrupted or not!
  - Sender needs to be able to decide whether the data it sent has been received!

- Performed in two layers on the Internet: the link layer and the transport layer

Reliability

- How can the receiver decide whether the packet it received is corrupted or not?
  - Add redundancy in the communication to perform checking
    - Example: Assume that the sender will data:10, 20, 30, 40. In addition to the data, it would also send a checksum of the data. To send 10, 20, 30, 40, the sender actually sends 10, 20, 30, 40, (100). That 100=10+20+30+40 is called checksum. When the receiver receives a packet, it computes the checksum to see if it matches that one in the packet and consider only the ones with matched checksum to be valid data.
  - How the checksum is computed decides the capability for error detection and correction.
  - In practice, cyclic redundance check (CRC) has been standardized and widely used.
    - CRC adds the checksum to make the data+checksum is divisible by a (binary) number called generator (sender adds the checksum), receiver checks the packet by dividing data+checksum by the generator to check data validity.

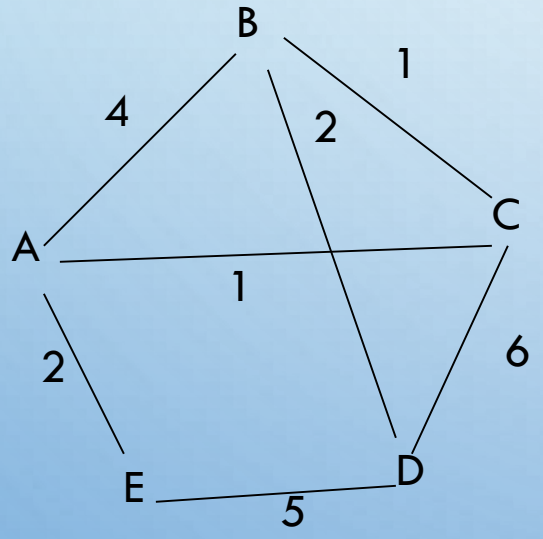
Reliability

- How can the sender decide whether the data it sent has been received?
  - The only way is to receive an acknowledgement from the receiver
  - This requires the sender/receiver to behave in certain way
    - Receiver needs to send ACK when it gets a packet
    - The sender needs to be able to timeout if it does not receive the ACK from the receiver
  - Needs to use a protocol to achieve this.

- Reliability is done at the link layer and TCP?

Routing

- Problem: When a router receives a packet that is going to destination: FSU 128.186.*.*, where should it forward the packet?



B gets a packet that goes to E, what to do next?

The routing issue is addressed in the Internet layer (IP). There are different types of routing algorithms in IP. The routing must be able to handle nodes dynamically joining and leaving the network.

- The major goal of an Internet routing algorithm is to find the shortest path to the destination.

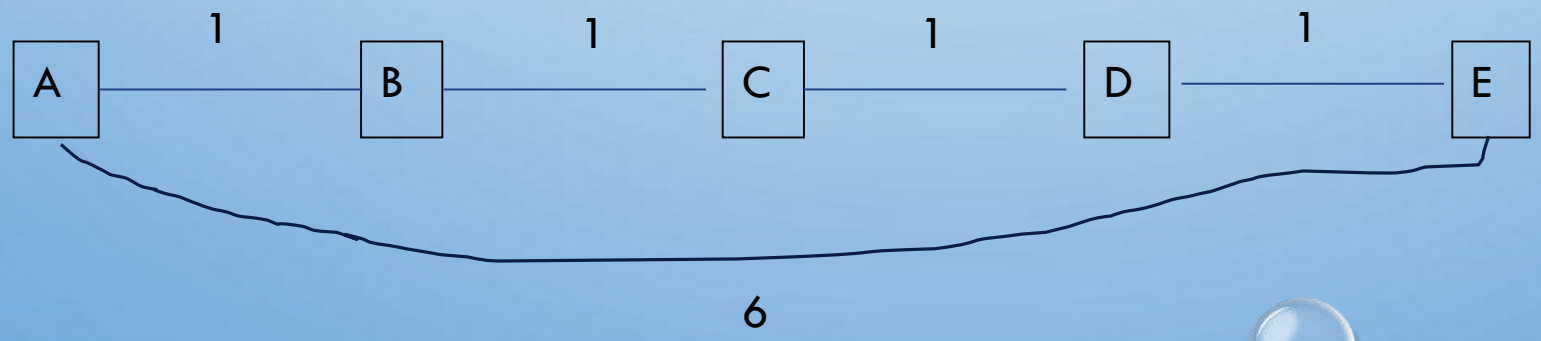
One commonly used routing algorithm is called distance vector routing (RIP protocol)

Distance Vector Routing – distributed shortest path algorithm

- Each router maintains a routing table. Each node that the router knows how to go to has an entry (you can consider the routing table is indexed by the destination)
  - Initially, each router only knows its neighbor (by using a link protocol)
- Each entry has two parts: the next hop to the destination and the distance (cost) to the destination.
- Periodically, each router exchanges its routing table with its neighbors.
- Router updates its entry (trying to find the shortest path to every destination) based on the routing table from its neighbors. The goal is to dynamically maintain the routing table where the next hop leads to the shortest path to the destination

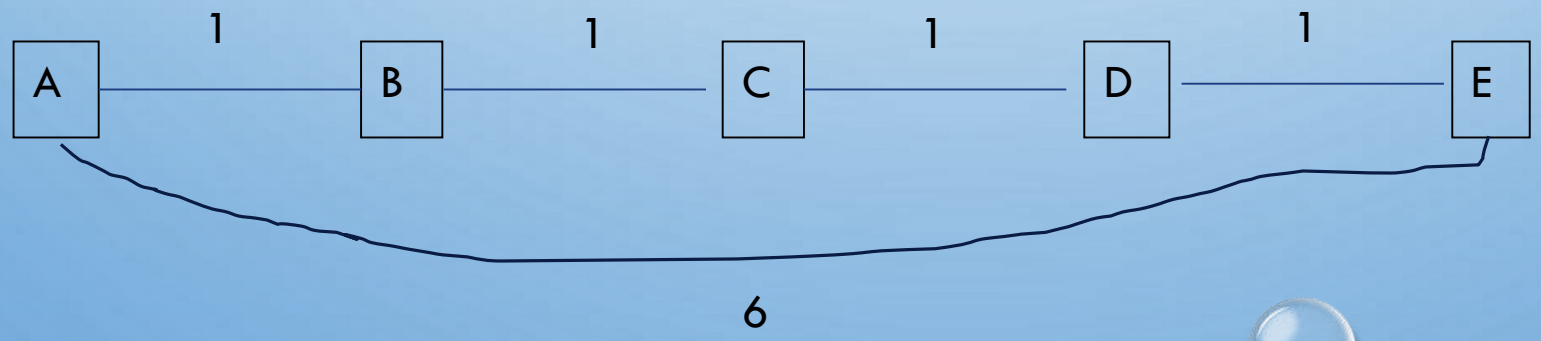
Distance Vector routing algorithm (initial)

	A	B	C	D	E
A	(0, -)	(1, A)	-	-	(6, A)
B	(1, B)	(0, -)	(1, B)	-	-
C	-	(1, C)	(0, -)	(1, C)	-
D	-	-	(1, D)	(0, -)	(1, D)
E	(6, E)	-	-	(1, E)	(0, -)



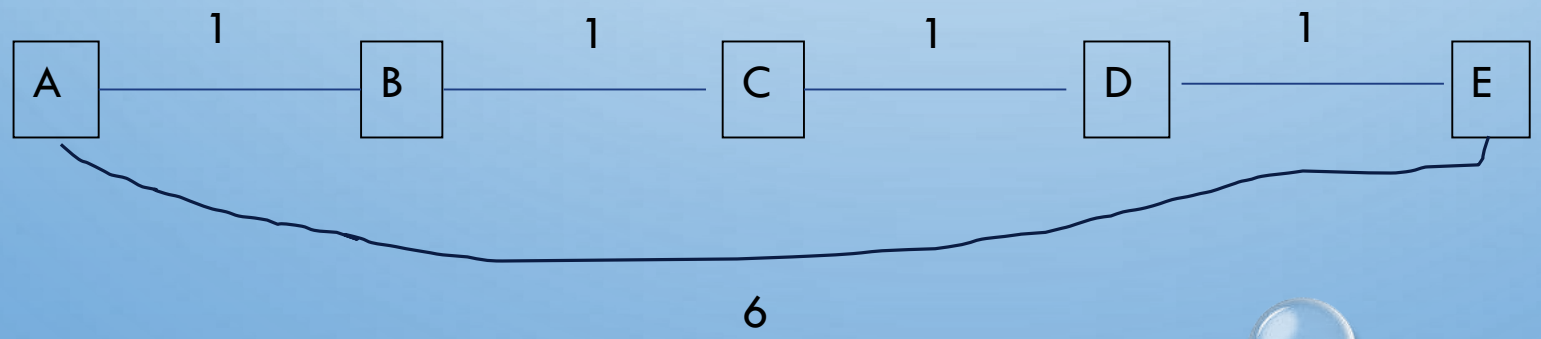
Distance Vector routing algorithm - Step 1

	A	B	C	D	E
A	(0, -)	(1, A)	(2, B)	(7, E)	(6, A)
B	(1, B)	(0, -)	(1, B)	(2, B)	(7, A)
C	(2, B)	(1, C)	(0, -)	(1, C)	(2, D)
D	(7, E)	(2, C)	(1, D)	(0, -)	(1, D)
E	(6, E)	(7, A)	(2, E)	(1, E)	(0, -)



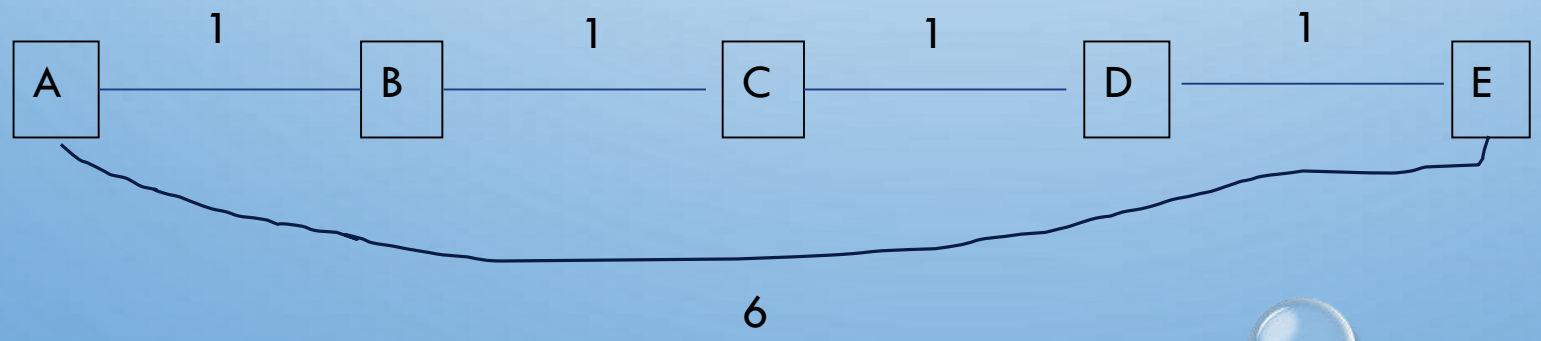
Distance Vector routing algorithm - Step 2

	A	B	C	D	E
A	(0, -)	(1, A)	(2, B)	(3, C)	(6, A)
B	(1, B)	(0, -)	(1, B)	(2, B)	(3, D)
C	(2, B)	(1, C)	(0, -)	(1, C)	(2, D)
D	(3, B)	(2, C)	(1, D)	(0, -)	(1, D)
E	(6, E)	(3, C)	(2, E)	(1, E)	(0, -)



Distance Vector routing algorithm - Step 3

	A	B	C	D	E
A	(0, -)	(1, A)	(2, B)	(3, C)	(4, D)
B	(1, B)	(0, -)	(1, B)	(2, B)	(3, D)
C	(2, B)	(1, C)	(0, -)	(1, C)	(2, D)
D	(3, B)	(2, C)	(1, D)	(0, -)	(1, D)
E	(4, B)	(3, C)	(2, E)	(1, E)	(0, -)



Decide the data rate to send

- The optimal data rate to send is dynamically changing on the Internet. How to decide the rate to send in such a condition?
- At a given time, there is always an ideal rate (which may be unknown to the sender), the sender would try to approximate that rate
- This is done in TCP.

- Consider this game: there is a number that is fixed, you need to guess what it is. Every time you guess, an oracle will tell you whether this number is found, or too small, or too large. What is the best strategy to guess the number?

Decide the data rate to send

- TCP slow start
  - Start with one segment per round trip, double the number of segments every round trip (when acknowledgement is received).
  - At some point, TCP would send too fast, and the segment would be lost (not acknowledge, timeout), then halve the number of segments and start AIMD (additive increment and multiplicative decrement)
- TCP AIMD (additive increment and multiplicative decrement)
  - Add one segment every round trip
  - If not acknowledge, halve the number of segments

Network performance metrics

- Latency and bandwidth
  - Hardware latency and bandwidth
  - End-to-end latency and bandwidth
- Network latency is the amount of time it takes for data to travel from one point to another across a network.
  - Units: second (s), millisecond ($ms = 10^{-3}s = 0.001s$), microseconds ($\mu s = 10^{-6}s = 0.000001s$), etc
- Bandwidth: the rate that data can be injected to the network (and transferred).
  - Units: the network bandwidth is measured in bits per second. E.g. 1000 bps or 1kbps.

Network performance metrics

- Bandwidth units (bits per seconds)
  - B and b:
    - ❖ B – bytes: data size or memory size is measured in bytes (B).
      - ❑ Example: 10KB ($10\ kilobytes = 10 \times 10^3\ bytes = 10{,}000\ bytes$), 100MB ($100\ megabytes = 100 \times 10^6\ bytes = 100{,}000{,}000\ bytes$), 16GB ($16\ gigabytes = 10 \times 10^9\ bytes = 16{,}000{,}000{,}000\ \ bytes$)
    - ❖ b – bits: network bandwidth is measured in bits per seconds
    - ❖ 1 Byte = 8 bits
    - ❖ Pay attention to the b/B in the spec: 1Gbps is different from 1GBps
      - ❑ Example 100Mbps Ethernet: 100,000,000 bits per second; 1800GBps NVLink: 1,800,000,000,000 Bytes per second
  - Most wired desktops are connected over 1Gbps links (1,000,000,000 bits per second)
    - ❖ Faster ones can be 10Gbps, 100Gbps, or more
    - ❖ Slower ones may be 100Mbps (100,000,000 bits per second)
  - Wireless Ethernet (Wifi): Wi-Fi 6 (802.11ax) 2021, up to 10Gbps. This is shared by all users on the network.
- You have a 1Gbps Internet service, what is the minimum amount of time to download a 10GB movie?

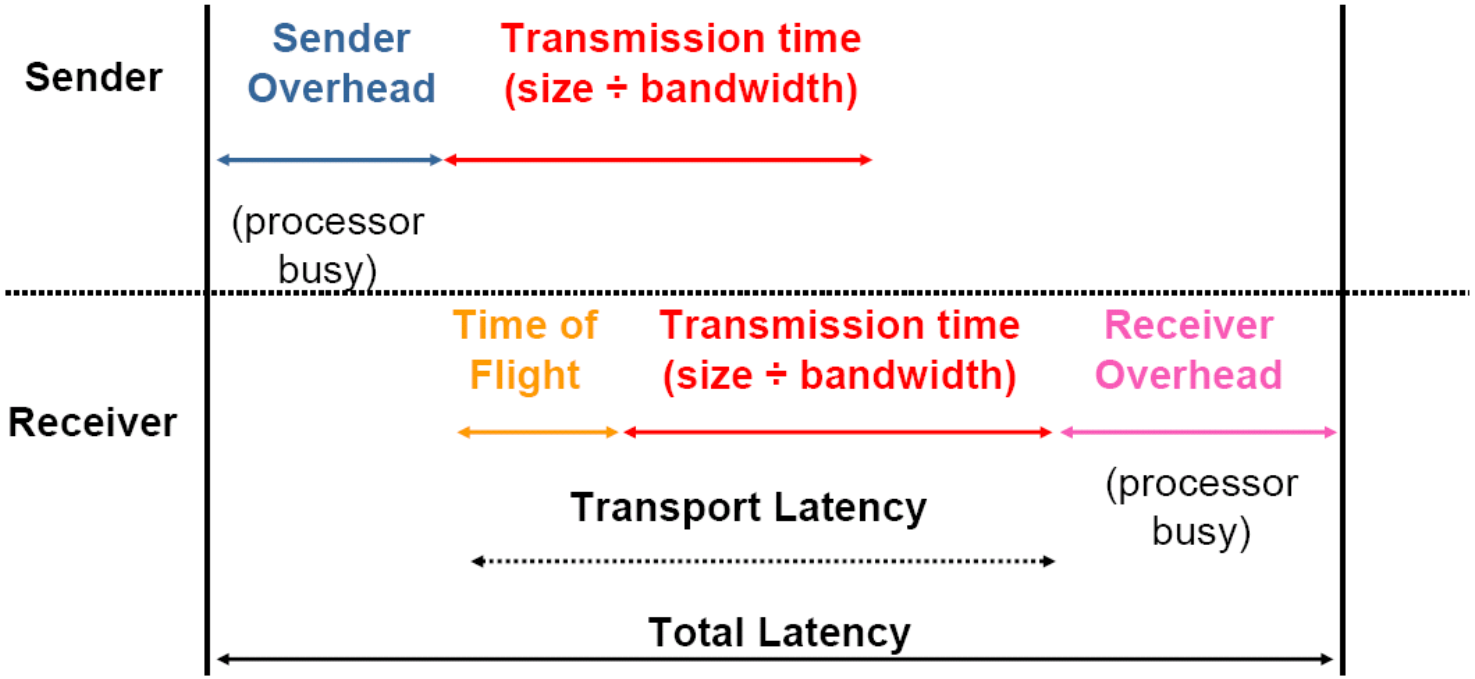
Network performance metrics

- Bandwidth determines the message injection (and transmission) time
  - Let W bits per second be the bandwidth of a link. Given a message of size M bytes (or 8M bits), it will take 8M/W seconds to transmit the message: transmission time = 8M/W
  - What is the transmission time to send 10000 bits over a 1Gbps link?
- Time of flight (propagation delay) is another component of the hardware communication time
  - The time between the first bit is placed on the wire in the sender to the time it reaches the receiver. This time is bounded by the speed of light.
    - If we send a message from Tallahassee to LA (distance roughly 3000 miles), what is the time of flight for this message assuming that the signal propagation speed on the wire is 2/3 of the speed of light (300,000,000 meters per second)?
    - The distance of a geostationary satellite from the earth is 22,236 miles (35,786km), estimate the communication latency when a communication goes through the such a satellite?
      - How does Space X's Starlink make this work?

Network performance metrics

- End-to-end latency: the amount of time from the time the sender calls write() to the time the receiver receives the data.
  - Software time at both sender (OS runs the write() code) and receiver (interrupt and OS runs the read() code) and
  - Hardware time: propagation time (time of flight) + transmission time
    - The propagation time is determined by the physical law (bound by the speed of light).

Network performance metrics



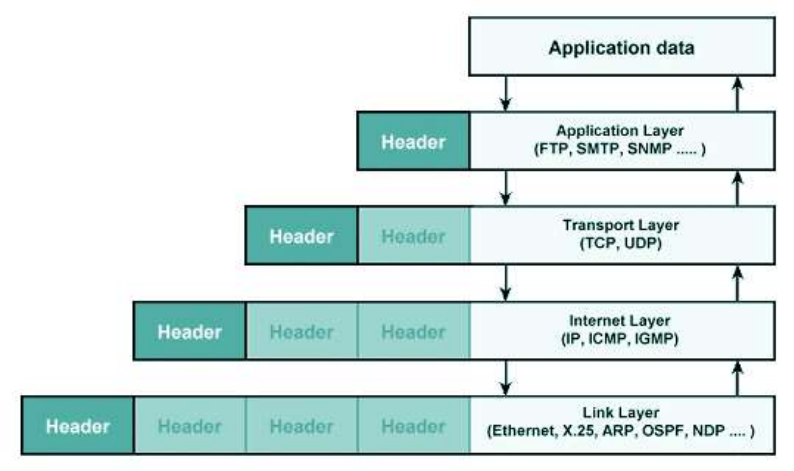
Total Latency = Sender Overhead + Time of Flight + Message Size ÷ BW + Receiver Overhead

Includes header/trailer in BW calculation?

Network performance metrics

- Network users in general care about end-to-end performance
  - Hardware bandwidth (e.g. 100Mbps Ethernet) is different from end-to-end (application level) bandwidth. Why?
- End-to-end bandwidth: the amount of data transferred / the time to transfer the data from sender to receiver.
  - An application: Internet speed test
  - In an Internet speed test, 500MB of data is transferred in 10 seconds over the uplink, what is the uplink bandwidth?

A Question



A computer is connected to a network using a **100 Mbps Ethernet** link. The **minimum Ethernet frame size is 64 bytes,** which includes all headers. The header sizes are:

- **Ethernet header:** 14 bytes
- **IP header:** 20 bytes
- **TCP header:** 20 bytes

Assuming the computer continuously sends **minimum-sized packets,** and there are **no retransmissions or losses,** what is the **maximum end-to-end throughput** (in Mbps) that the computer can achieve at the **application layer**?