

CPE348: Introduction to Computer Networks

Lecture #3: Chapter 1.2

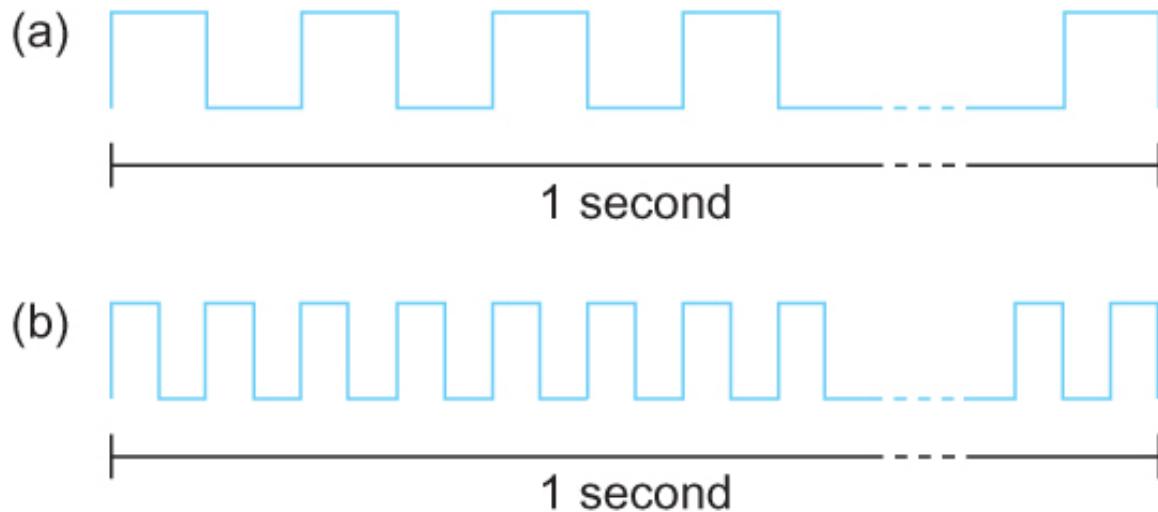


Jianqing Liu
Assistant Professor of Electrical and Computer
Engineering, University of Alabama in Huntsville
jianqing.liu@uah.edu
<http://jianqingliu.net>

Metrics and Performance

- Bandwidth
 - Width of the frequency band
 - Number of bits per second that can be transmitted over a communication link
- 1 Mbps: 1×10^6 bits/second
- 1×10^{-6} seconds to transmit each bit or imagine that a timeline, now each bit occupies 1 micro second space.
- On a 2 Mbps link the width is 0.5 micro second.
- Smaller the width more will be transmitted per unit time.

Bandwidth



Bits transmitted at a particular bandwidth can be regarded as having some width:

- (a) bits transmitted at 1Mbps (each bit 1 μ s wide);
- (b) bits transmitted at 2Mbps (each bit 0.5 μ s wide).

Bandwidth

Bandwidth, throughput, (perceived) data rate

What are the differences?

Delay

- Latency = Propagation + transmit + queue
- Propagation = distance/speed of light*
- Transmit = size/bandwidth

- One bit transmission => propagation is important
- Large bytes transmission => bandwidth is important

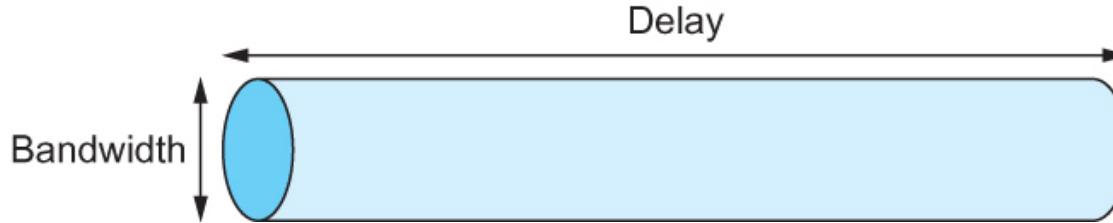
- *Unless the speed of transmission is otherwise specified, the speed of light is 3×10^8 meter/second.

Round-Trip-Time (RTT)

- **RTT**
 - the length of time it takes for a signal to be **sent** plus the length of time it takes for an acknowledgement of that signal to be **received**.
- **1 cross country RTT is approximately 100 milliseconds**

Delay X Bandwidth

- We think of the channel between a pair of processes as a hollow pipe
 - Latency (delay) - length of the pipe and
 - Bandwidth (transmission rate) - the width of the pipe
- Delay of 50 ms and bandwidth of 45 Mbps
 - ⇒ 50×10^{-3} seconds $\times 45 \times 10^6$ bits/second
 - ⇒ 2.25×10^6 bits = 281.25×10^3 Bytes = 274.66 KB data.



Network as a pipe

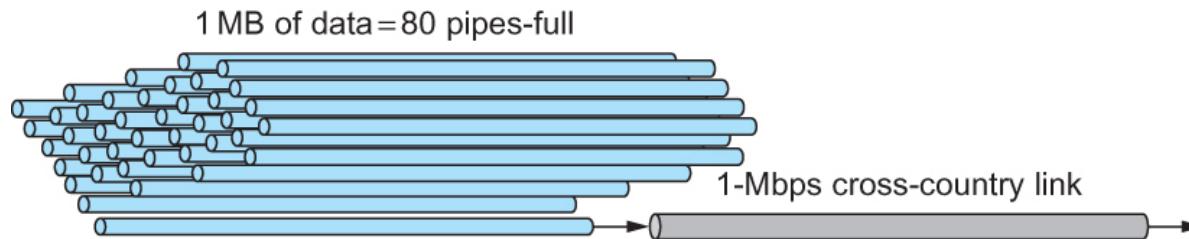
Delay X Bandwidth

- Relative importance of bandwidth and latency depends on application
 - For large file transfer, bandwidth is critical
 - For small messages (HTTP, SMS, etc.), latency is critical

Delay X Bandwidth

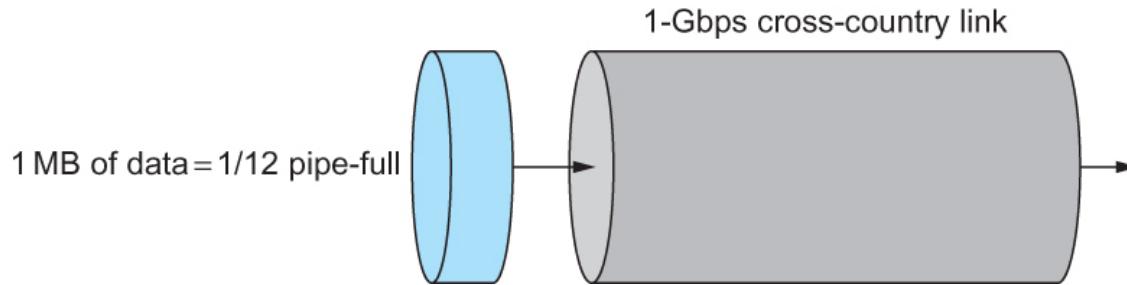
How many bits the sender must transmit before the first bit arrives at the receiver if the sender keeps the pipe full.

Relationship between bandwidth and latency



$$1 \text{ pipe-full} = \text{Delay} \times \text{BW} = 1 \times 10^6 \text{ (0.1)} = 100,000 \text{ bits}$$

$$1 \text{ MB} = 1024(1024)^*8 \text{ bits} = 8.3886 \times 10^6 \text{ bits} = 83.9 \text{ pipe-fulls}$$



$$1 \text{ pipe-full} = \text{Delay} \times \text{BW} = 1 \times 10^9 \text{ (0.1)} = 1 \times 10^8 \text{ bits}$$

$$1 \text{ MB} = 1024(1024)/8 \text{ bits} = 8.3886 \times 10^6 \text{ bits} = 0.0839 \text{ pipe-fulls}$$

A 1-MB file would fill the 1-Mbps link 80 times,
but only fill the 1-Gbps link 1/12 of one time

Other Metrics

- Reliability
 - How a connection, a device or a service is resilient to disruptions.
- Packet loss
 - Due to errors in link, buffer overflow in network devices (e.g., routers)
 - Will be covered in Chapter 6
- Number of hops

Summary

- A layered architecture for computer network;
- A network application case study from top-bottom;
- Metrics and performance for computer networks;

Chapter 2: Getting Connected

→ Links → ex. last-mile or backbone links

- can be classified by the medium they use (m)
- can be characterized by: freq. (Hz), speed of light (m/s), & wavelength
- can eval. performance using Shannon-Hartley Theorem:
 - ↳ finds channel capacity / throughput as an upper bound of the data rate over that medium.

$$C = B \log_2 \left(1 + \frac{S}{N} \right)$$

→ B is bandwidth (Hz)
→ S is signal power at receiver
→ N is noise power at receiver.

→ Encoding

→ NRZ

- ↳ con: baseline wander w/ too many consec. 1's or 0's, which affects the average.

- ↳ con: clock recovery w/ too many consec. 1's or 0's.

→ NRZI

- ↳ solves NRZ issue w/ consec. 1's, but not consec. 0's.

→ Manchester

- ↳ pro: solves consec. 1's and 0's problem.

- ↳ con: doubles the rate. Rcvr. has half time to detect each pulse of the signal.

↳ i.e. it consumes more bandwidth.

→ UB/SB

- ↳ inserts extra bit to break up 1's and 0's then transmitted

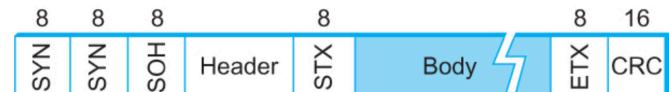
using NRZI

- ↳ 80% effective

→ Byte - Oriented Framing

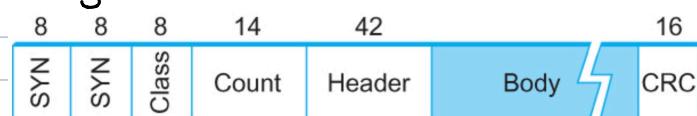
→ BISYNC

- ↳ uses DLE (data link escape) to precede any data portion equal to DLE or ETX.



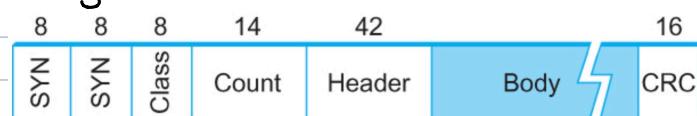
→ PPP

- ↳ payload is negotiated. default 1500 bytes → use demultiplexing
- ↳ LCP (link control prot.) negotiates some field sizes



→ DDCMP

- ↳ count holds # of bytes in frame body
- ↳ if count corrupted, get a framing error (Usually back-to-back frames reqd.)



→ Bit-Oriented Framing: HDLC

→ beginning seq: 0111110
↳ also sent when idle

→ after 5 consec 1's if next is:

0 → stuffed, discard (0111110)

10 → end of frame marker (0111110)

11 → error, must get seq. again before it can rcv. again.
(0111111)



→ Error Detection - add redundant info to determine if errors have been introduced.

- * *know how to do* { → CRC (HDLC, DDCMP, CSMA/CD, Token Ring) - XOR
→ 2D Parity (BISYNC)
↳ guaranteed to catch all 3-bit errors. Can get most 4-bit.
→ Checksum (IP) → Used at network level.
↳ Carryout added to LSB.
↳ Uses 1's complement

* Why is error detection implemented in dif. OSI layers?

For CRC

Properties of Generator Polynomial

It is possible to detect the following types of errors by a $C(x)$ with degree r

All single-bit errors, as long as the x^r and x^0 terms have nonzero coefficients.

All double-bit errors, as long as $C(x)$ has a factor with at least three terms.

Any odd number of errors, as long as $C(x)$ contains the factor $(x+1)$.

Any "burst" error (i.e., sequence of consecutive error bits) for which the length of the burst is less than r bits. (Most burst errors of larger than r bits can also be detected.)

→ Reliable Transmission

- ↳ send an ACK (acknowledgement) to show frame was rcv'd. successfully.
↳ timeout if ACK is not rcv'd.
↳ Using ACK and timeout combo = ARQ

→ ARQ

→ Stop-and-Wait - use frame # (0 or 1) to avoid confusion.

↳ sending rate = bits per frame / time per frame = RTT

↳ 1 outstanding frame at a time → poor use of capacity.

→ Sliding Window

→ On sender: LFS - LAR \leq SWS

→ on rcvr: LAF - LFR \leq RWS

↳ largest acceptable frame

→ to check:

→ Seq. # \leq LFR or Seq. # $>$ LAF

↳ discard. outside rcvng window

→ LFR $<$ Seq. # \leq LAF

↳ Accept and send an ACK

→ Seq. Num to ACK = largest Seq. # not yet ACKed.

↳ if higher Seq. # recv'd., holds ACK until
Seq. Num to ACK is recv'd.

↳ then adjust window w/ Seq. Num to ACK = LFR.

→ Can be improved w/ NAKs (negative ACKs), or
additional or selective ACKs. (solicit, duplicate, explicit)

→ Use SWS = Delay \times Bandwidth / frame size to keep pipe full.

→ RWS should not be greater than SWS.

↳ if RWS = SWS

SWS \leftarrow (MaxSeqNum + 1) / 2 or MaxSeqNum \geq 2 * SWS - 1

↳ if RWS $<$ SWS

MaxSeqNum \leq 2 * SWS - 1

→ Ethernet → uses CSMA/CD, uses ALOHA as root prot.

↳ classic ethernet has an upper limit to its length → 500 m coax

Chapter 2 Terminology

- Encoding - converting a string of binary data to a stream of pulse signal
- Modulation - modifying (freq., amp., and phase) of the base-band signals to the carrier frequency.
- NRZ (non-return to zero) - a bit encoding scheme that encodes a 1 as a high signal and 0 as the low sig.
- NRZI (non-return to zero inverted) - a bit encoding scheme that makes a transition from the current signal encode to a 1 and stays at current sig. to encode a 0.
- Manchester - a bit encoding scheme that transmits the XOR of the clock and the NRZ-encoded data.
Used on Ethernet.
0: low → high transition
1: high → low transition.
- 4B/5B - a type of bit encoding scheme used in Fiber Distributed Data Interface (FDDI), in which every 4 bits of data are transmitted as a 5-bit seq.
- Frame - another name for packet, usually used in ref. to packets sent over a single link rather than a whole network.
 - ↳ Framing - imp. problem about how rcv. detects the beginning and ending of a frame.
- BISYNC (Binary Synchronous Communication) - a byte-oriented link-level protocol dev'd. by IBM in the 60's.
- PPP (Point-to-Point Protocol) - data link protocol usually used to connect computers over a dial-up line.
- DDCMP (Digital Data Communication Message Protocol) - a byte-oriented link-level protocol used in DECnet.
- HDLC (High-Level Data Link Control) - an ISO-standard link-level protocol. It uses bit stuffing to solve the framing problem.
- ARQ (Automatic repeat request) - retransmit if don't rcv. an ACK before timeout.
 - ↳ Stop-and-Wait and Sliding Window are ex. ARQ protocols.

→ Ethernet - popular LAN technology that uses CSMA/CD and has a bandwidth of 10 Mbps. Ethernet is just a passive wire - all aspects of Ethernet transmission are imp. by host adaptors.

→ CSMA/CD (Carrier Sense Multiple Access w/ Collision Detection)

- ↳ CS - all nodes can distinguish b/w a busy + active link.
- ↳ MA - a set of nodes can send + recv. frame over a shared link.
- ↳ CD - a node listens as it transmits, so it can detect when a frame it is transmitting has collided w/ one transmitted by another node.

CPE348: Introduction to Computer Networks

Lecture #4: Chapter 2.1



Jianqing Liu
Assistant Professor of Electrical and Computer
Engineering, University of Alabama in Huntsville
jianqing.liu@uah.edu
<http://jianqingliu.net>

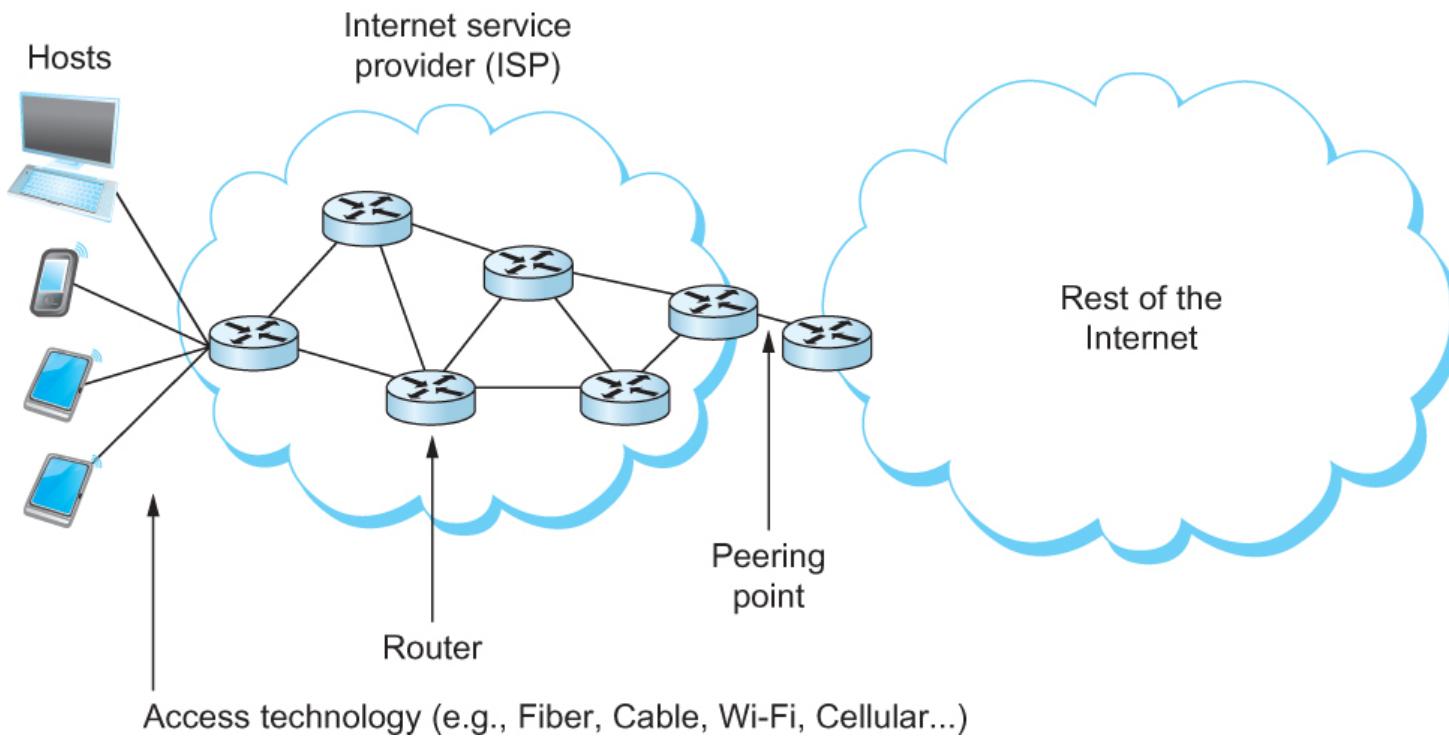
Chapter Outline

- Perspectives on Connecting nodes
- Encoding
- Framing
- Error Detection
- Reliable Transmission
- Ethernet and Multiple Access Networks
- Wireless Networks

Chapter Goal

- Exploring different **communication medium**
- Understanding the issue of **encoding** bits
- Discussing techniques to **detect transmission errors**
- Discussing how to make links **reliable**
- Introducing **Media Access Control Problem**
- Introducing Wireless Networks

Perspectives on Connecting



- ❖ “Last-mile” connection: access technology
- ❖ Backbone connection: Internet technology

What are communication medium?

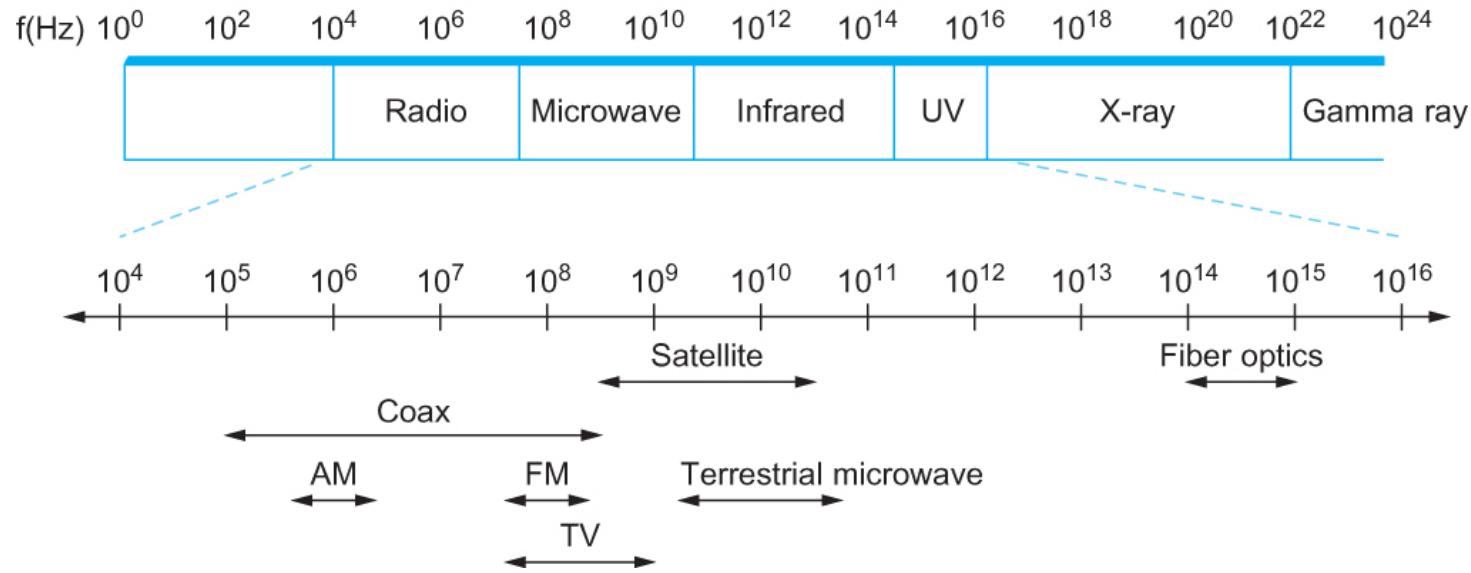
- All links rely on electromagnetic radiation propagating through a medium.
- Links can be classified by the medium they use
 - Coaxial cable (e.g., TV)
 - Optical fiber (e.g., Internet)
 - Air/free space (e.g., Bluetooth)
 - Visible light (e.g., barcode scanner)



How to characterize a link

- Frequency (in Hz)
 - WiFi/Bluetooth/Microwave Oven: 2.4 GHz
 - 4G/LTE: 2.5 GHz – 2.7 GHz
- Speed of light (in m/s)
 - How fast the light is travelled.
- Wavelength (in meter)
 - Speed of light divided by frequency gives the wavelength.

Radio frequency of a link



Electromagnetic spectrum

Performance metrics for a link

- How to evaluate the performance of a link?
- Channel capacity/throughput: Shannon-Hartley Theorem
- $C = B * \log_2(1+S/N)$
 - Where B = 3000Hz is the bandwidth
 - S is the signal power at the receiver
 - N the noise power at the receiver
- It is an upper bound of the data rate over that communication medium.
- How can we get 56kbps? Example

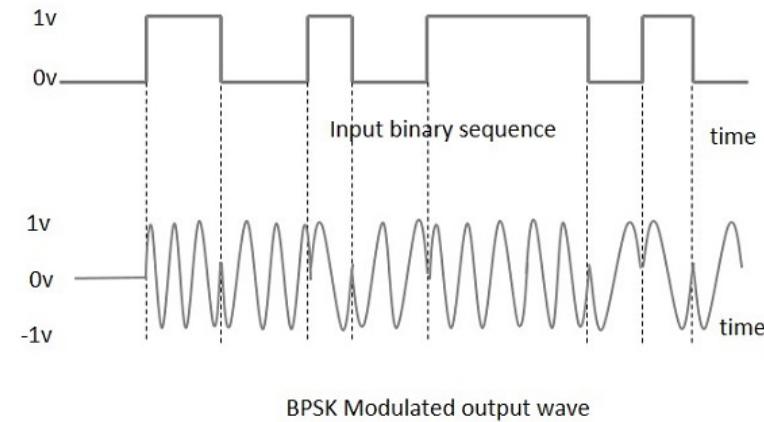
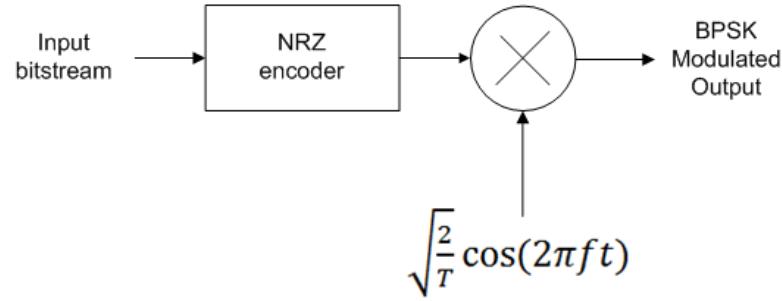
Some Examples

Service	Bandwidth (typical)
Dial-up	28–56 kbps
ISDN	64–128 kbps
DSL	128 kbps–100 Mbps
CATV (cable TV)	1–40 Mbps
FTTH (fibre to the home)	50 Mbps–1 Gbps

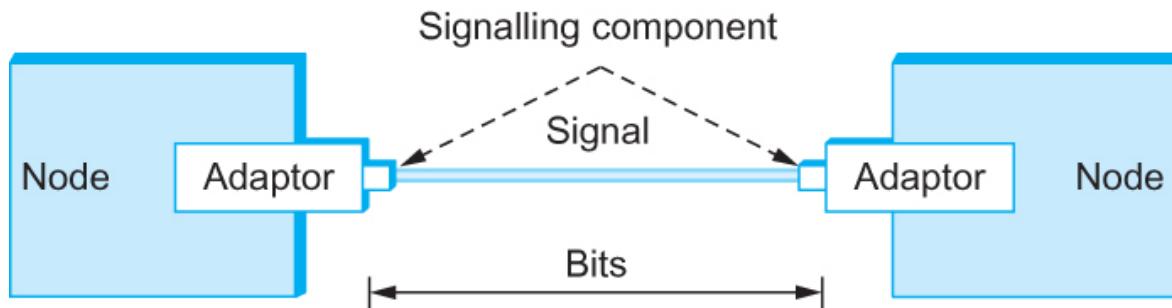
Common services available to connect your home

Encoding & Modulation

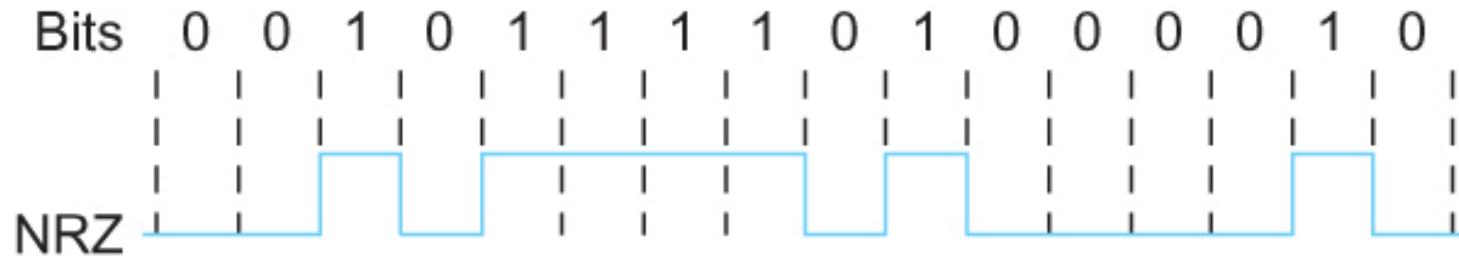
- Converting a stream of binary data to a stream of pulse signal is called *encoding*.
- Modifying (frequency, amplitude, and phase) the base-band signals to the carrier frequency is called *modulation*.



Encoding



Signals travel between signaling components; bits flow between adaptors



NRZ encoding of a bit stream

Encoding

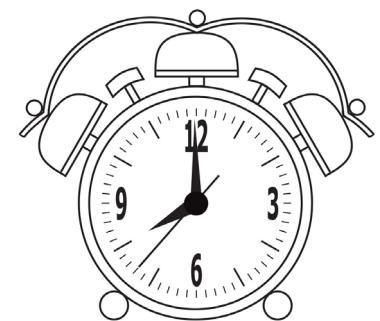
- Problem with NRZ - **Baseline wander**
 - The receiver keeps an average of the signals it has seen so far to distinguish between low and high signal
 - When a signal is significantly lower than the average, it is 0, else it is 1
 - Too many consecutive 0's and 1's cause this average to change, making it difficult to detect

Problem of consecutive 0's or 1's

Encoding

- Problem with NRZ - **Clock recovery**
 - Both the sending and decoding process is driven by a clock
 - Frequent transition from high to low or vice versa are necessary to enable clock recovery
 - The sender and receiver have to be precisely synchronized

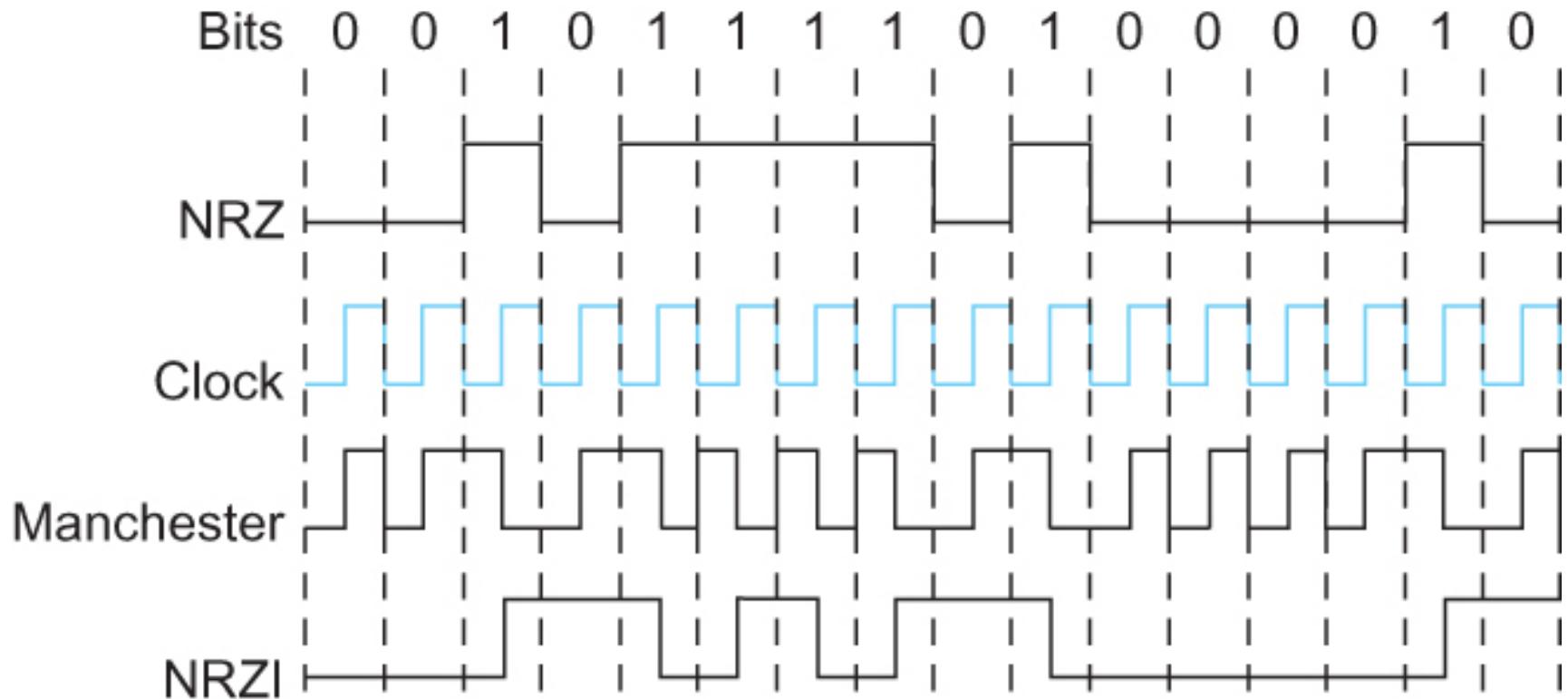
Problem of consecutive 0's or 1's



Encoding

- NRZI - Non Return to Zero Inverted
 - Sender makes a transition from the current signal to encode 1 and stay at the current signal to encode 0
 - Transition occurs on rising clock edge
- Solve for consecutive 1's, but does not solve problem with consecutive 0's

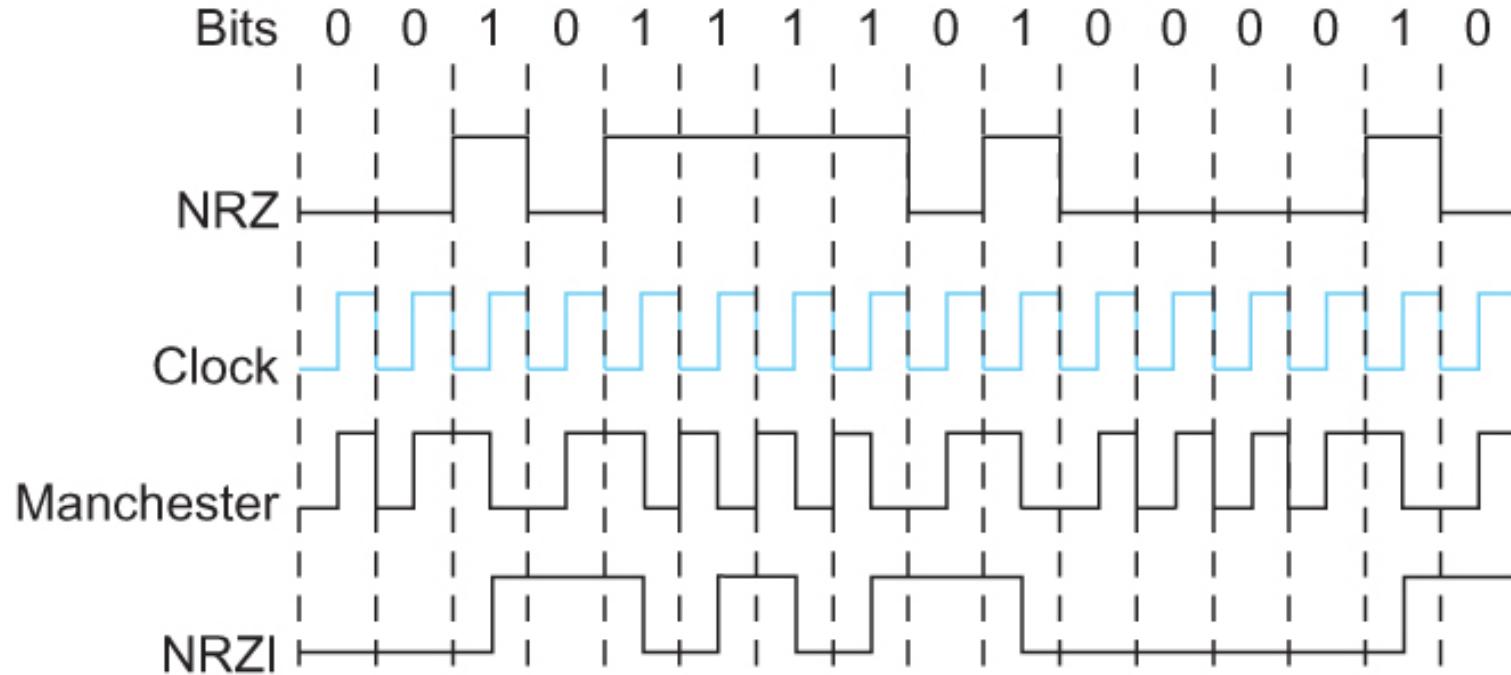
NRZI Waveform



Encoding

- Manchester encoding
 - Merging the clock with signal by transmitting Ex-OR of the NRZ encoded data and the clock
 - In Manchester encoding
 - 0: low → high transition
 - 1: high → low transition

Encoding



Different encoding strategies

Encoding

- Pros & Cons with Manchester encoding
 - **Solves** consecutive 1's and 0's problem
 - **But, doubles** the rate
 - Which means the receiver has half of the time to detect each pulse of the signal
 - In other words, **consuming more bandwidth**

Encoding

- 4B/5B encoding
 - Insert extra bits into bit stream so as to break up the long sequence of 0's and 1's
 - Every 4-bits of actual data are encoded in a 5- bit code that is transmitted to the receiver
 - Then, transmitted using NRZI
 - 80% efficient

Encoding

■ 4B/5B encoding mapping

Data Transmit

0000 → 11110

0001 → 01001

0010 → 10100

..

..

1111 → 11101

Other 5B codes

16 left (16 used for data)

11111 – when the line is idle

00000 – when the line is dead

00100 – to mean halt

13 left : 7 invalid, 6 for various control signals

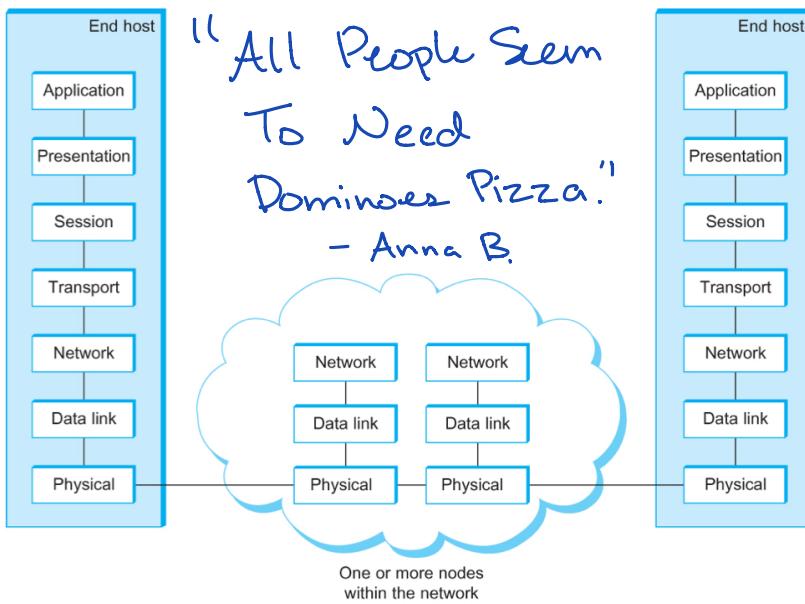
Encoding

■ 4B/5B encoding

4-bit Data	5-bit code	4-bit Data	5-bit code
0000	11110	1000	10010
0001	01001	1001	10011
0010	10100	1010	10110
0011	10101	1011	10111
0100	01010	1100	11010
0101	01011	1101	11011
0110	01110	1110	11100
0111	01111	1111	11101

Chapter 1: Foundation Problem: Building a Network

OSI 7-layer Model



* 348 is OSI going bottom-up

→ lower layer serves the one above it.
→ encompasses rules, algorithms, + protocols.

→ Rules: format, naming, addressing, etc.
→ Algorithms: functions
→ Protocols: a collection of agreements (incl. rules + algs) for a conversation.

→ Specified in the header of each layer

Know Network Applications

- WWW / internet
- email
- online social networks
- streaming videos
- file sharing
- instant messaging

* KB vs. kb

KB = 1024 bytes
kb = 1000 bits

Application Protocols

- URL - uniform resource locator
- HTTP - Hyper text transfer protocol
- TCP - Transmission Control protocol

Links - physical connection

- has a bit-rate or capacity
- has a propagation delay
- transfer time on link:
- can be shared by multiplexing, de-multiplexing, and Synchronous time-division multiplexing

bit / bit-rate + prop. delay

* Know difference w/ bandwidth, throughput, and (perceived) data rate
↳ perceived data rate = how much data the app. is actually reading/transmitting

Delay

- Latency = propagation + transmit + queue
- Propagation = distance / speed of light
- Transmit = size / bandwidth

know how to
use to find
total delay

- ★ for 1 bit transmissions - prop. time is important
- ↳ for large byte transmissions - bandwidth is important
- ★ think of latency (delay) as the length of the pipe and of bandwidth (transmission rate) as the width of the pipe.
- RTT = $\left(\frac{\text{distance}}{c} \times 2 \right) + \text{any processing that happens along the way}$
- see delay \times bandwidth product.
- Throughput = TransferSize / TransferTime
- TransferTime = RTT + $1/\text{Bandwidth} \times \text{TransferSize}$

Chapter 1 Terminology

- OSI Model - Open Systems Interconnection Model p. 32
- Protocol - a collection of agreements (including rules + algorithms) for a conversation.
- Host - comp. attached to one or more networks that supports users and runs application programs.
- Switch - a network node that forwards packets from inputs to outputs based on header info. in each packet.
 - ↳ differs from a router mainly b/c it usually does not interconnect networks of different types
- Intertetwork (internet) - a collection of (poss. heterogenous) (h-to-h) packet-switching networks interconnected by routers.
 - ↳ different from def. for Internet which uses TCP/IP architecture
- Router (gateway) - network node connected to 2 or more networks that forwards packets from one network to another.
 - ↳ different from a bridge, repeater, and switch.
- Routing - process by which nodes exchange topological info. to build correct forwarding table.
 - ↳ see forwarding, link state, and distance vector.
- Broadcast - a method of deliv. a packet to every host on a network or internet. Can be imp. in hardware (Ethernet) or software (IP broadcast).
- Unicast - sending a packet to a single dest. host.
- Multicast - a special form of broadcast where packets are deliv. to a specified subgroup of network hosts.
- Bandwidth (data rate) - measure of capacity of a link or connection, usually given in bits/second.
- Throughput - observed rate of data sent through a channel
- Delay x Bandwidth Product - product of a network's RTT and Bandwidth. Gives a measure of how much data can be in transit on the network. Aka. how many bits the sender must transmit before the first bit arrives at the receiver if the sender keeps the pipe full.

$$1 \text{ pipe full} = \text{Delay} \times \text{Bandwidth} = \text{RTT} \times \text{Bandwidth}$$

CPE348: Introduction to Computer Networks

Lecture #2: Chapter 1.1



Jianqing Liu
Assistant Professor of Electrical and Computer
Engineering, University of Alabama in Huntsville
jianqing.liu@uah.edu
<http://jianqingliu.net>

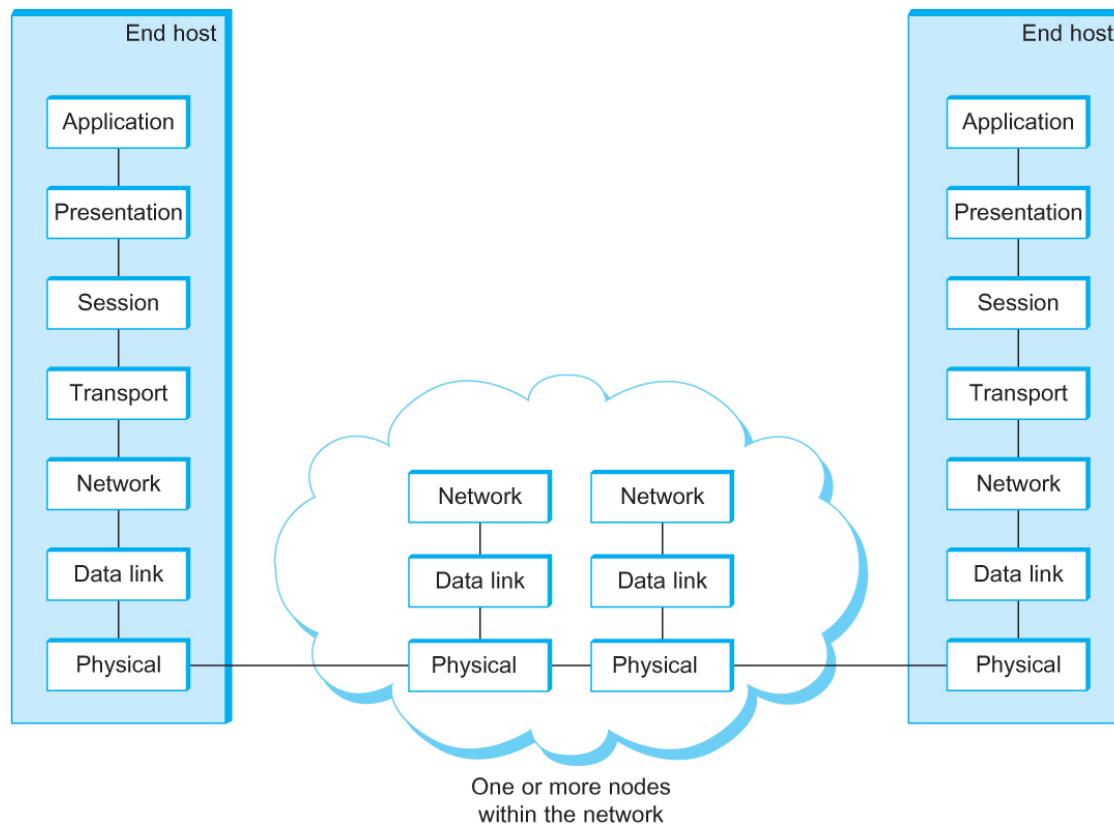
Chapter Outline

- Network Architecture
- Applications
- Metrics and Performance

Chapter Goal

- Overview of the **network architecture**
- Briefly explain how to develop a network application following a **top-down approach**
- Given several **metrics** that will be used to evaluate the performance of computer network

OSI Model



The OSI 7-layer Model
OSI – Open Systems Interconnection

OSI Model

- It is the **foundation** of any computer network;
- It **partitions** a computer network into **abstraction layers**;
- A lower layer serves the layer above it;
- It encompasses a wide range of **rules**, **algorithms** and **protocols**;

OSI model

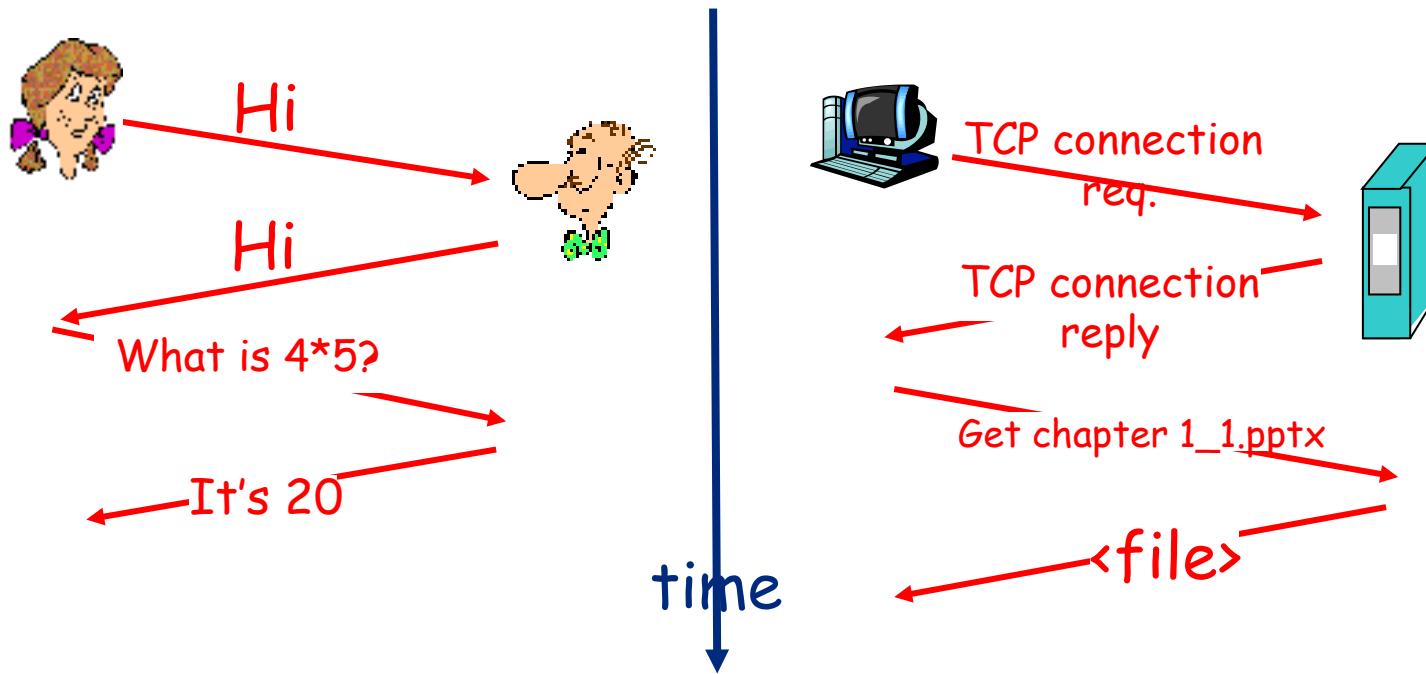
by layer

7. **Application layer** [hide]
NNTP · SIP · SSI · DNS · FTP · Gopher · HTTP · NFS · NTP · SMPP · SMTP · SNMP · Telnet · DHCP · Netconf · *more....*
6. **Presentation layer** [hide]
MIME · XDR · ASN.1
5. **Session layer** [hide]
Named pipe · NetBIOS · SAP · PPTP · RTP · SOCKS · SPDY
4. **Transport layer** [hide]
TCP · UDP · SCTP · DCCP · SPX
3. **Network layer** [hide]
IP (IPv4 · IPv6) · ICMP · IPsec · IGMP · IPX · AppleTalk · X.25 PLP
2. **Data link layer** [hide]
ATM · ARP · IS-IS · SDLC · HDLC · CSLIP · SLIP · GFP · PLIP · IEEE 802.2 · LLC · MAC · L2TP · IEEE 802.3 · Frame Relay · ITU-T G.hn DLL · PPP · X.25 LAPB · Q.921 LAPD · Q.922 LAPF
1. **Physical layer** [hide]
EIA/TIA-232 · EIA/TIA-449 · ITU-T V-Series · I.430 · I.431 · PDH · SONET/SDH · PON · OTN · DSL · IEEE 802.3 · IEEE 802.11 · IEEE 802.15 · IEEE 802.16 · IEEE 1394 · ITU-T G.hn PHY · USB · Bluetooth · RS-232 · RS-449

Wiki Screenshot

Rules, Algorithms, Protocols?

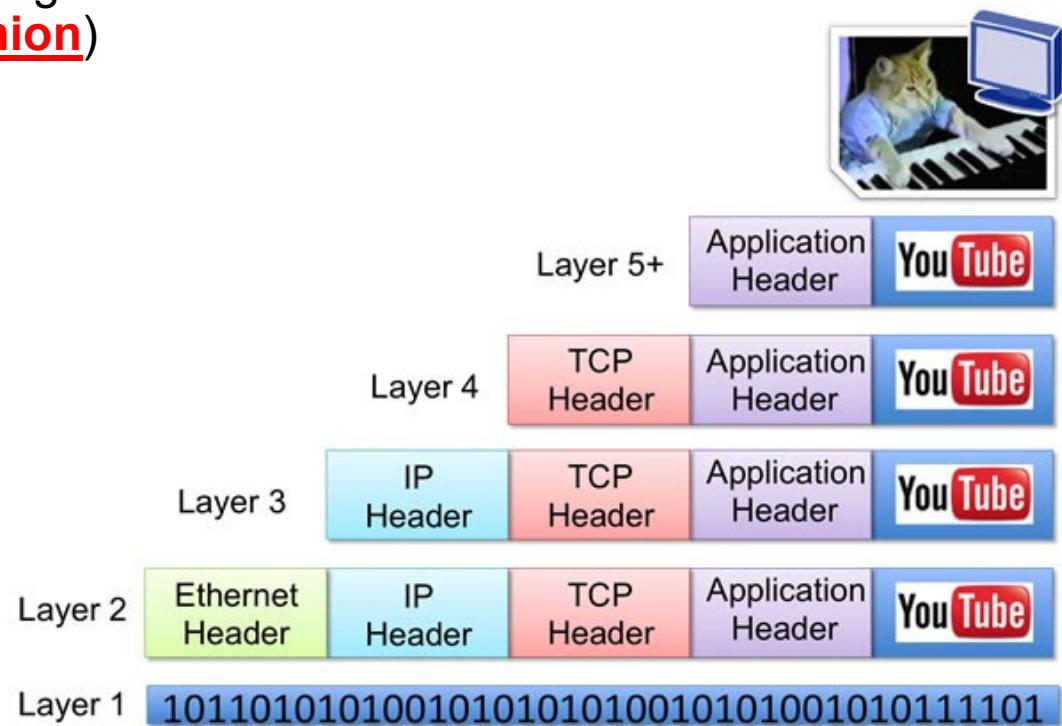
a human conversation and a computer conversation:



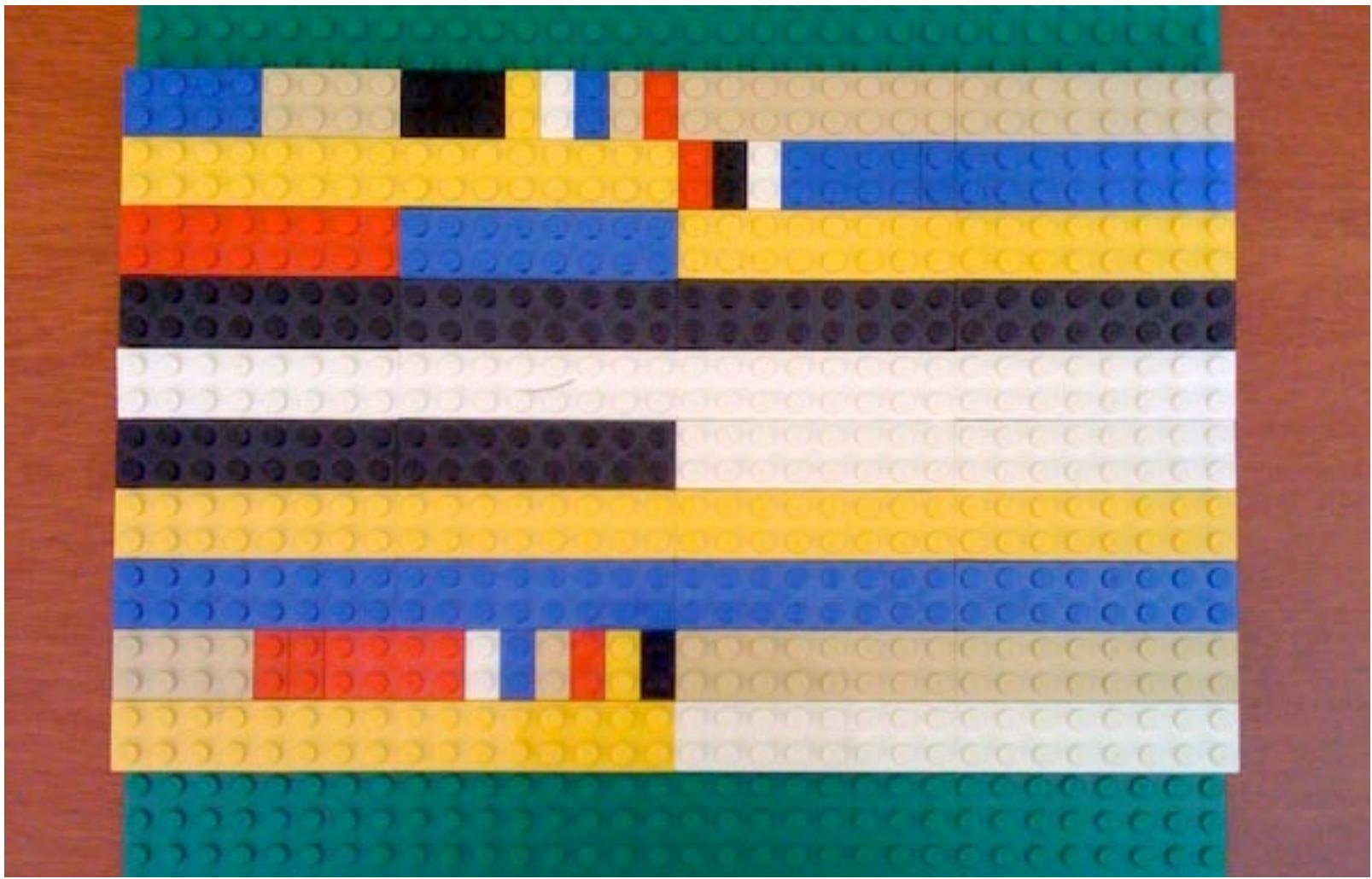
- **Rules:** the format, naming, addressing, etc.
- **Algorithms:** functions
- **Protocols:** A collection of agreements, including rules and algorithms, for a conversation

Rules, Algorithms, Protocols?

- **Rules, Algorithms and Protocols**: are specified in the header of each OSI layer;
- **Send host** goes from top to bottom, encapsulates data; whereas **receive host** goes from bottom to top, decapsulates data. (**onion**)



e.g., TCP/IP Header in Lego



Network Architecture

This course is about to study **Rules, Algorithms and Protocols** of each OSI layer, using a **bottom-up approach**.

First, let's take an alternative view in terms of developing a **network application**, using a **top-bottom approach**.

Applications

- Most people know about the Internet (a computer network) through applications
 - World Wide Web (www.espn.com)
 - Email (outlook)
 - Online Social Network (facebook)
 - Streaming Video (youtube)
 - File Sharing (dropbox)
 - Instant Messaging (whatsapp)
 - ...

Application Protocol

- URL
 - Uniform resource locator
 - <http://www.cs.princeton.edu/~llp/index.html>
- HTTP
 - Hyper Text Transfer Protocol
- TCP
 - Transmission Control Protocol
- 17 messages for one URL request
 - 6 to find the IP (Internet Protocol) address
 - 3 for connection establishment of TCP
 - 4 for HTTP request and acknowledgement
 - Request: I got your request and I will send the data
 - Reply: Here is the data you requested; I got the data
 - 4 messages for tearing down TCP connection

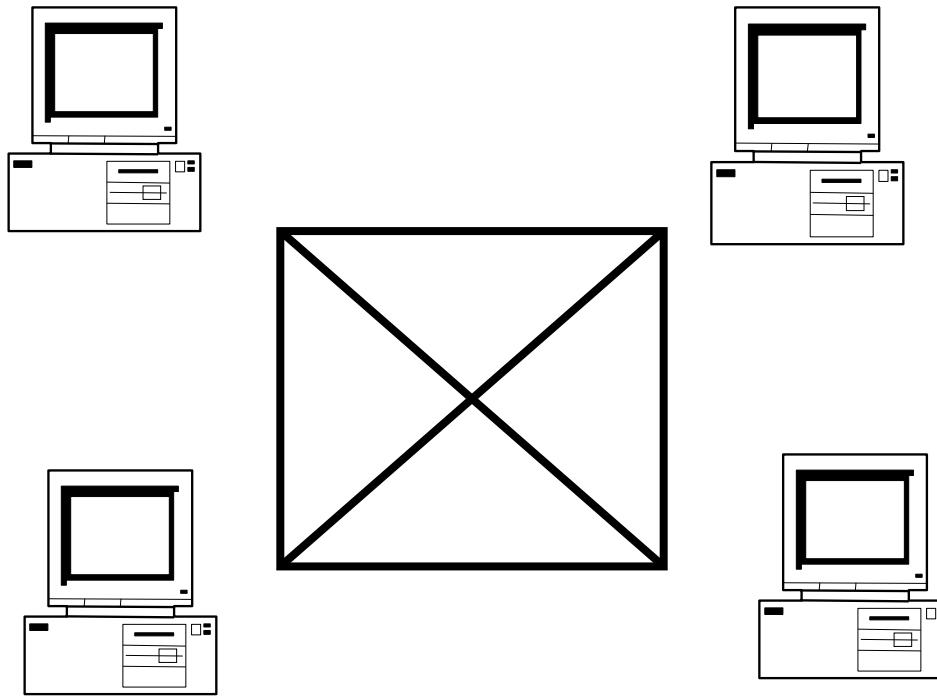
Connectivity



Link

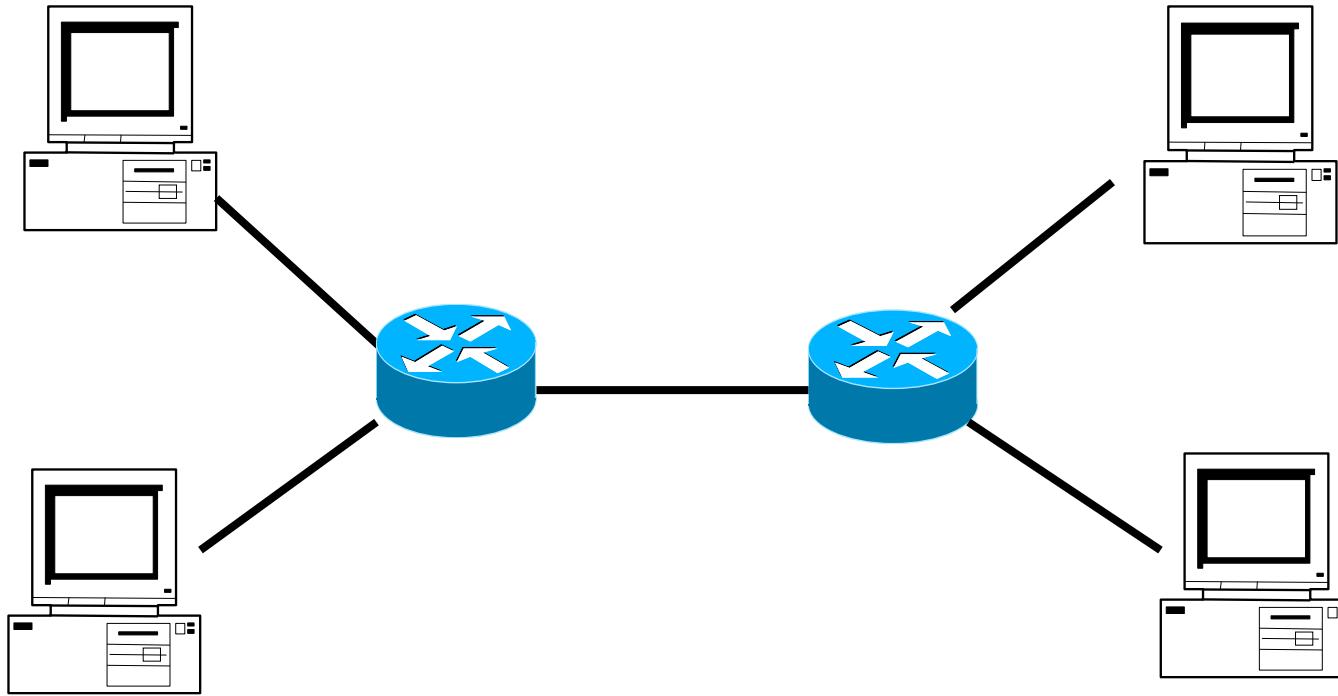
- Fiber optics, ethernet, wireless, cable,...
- Characterized by
 - Capacity or bit-rate (1.5 Mb/s, 100Mb/s, ...)
 - Propagation delay (10us, 10ms, 100ms, ..)
 - Transfer time on a link = #bit/bit-rate + propagation delay
 - ...

Connectivity



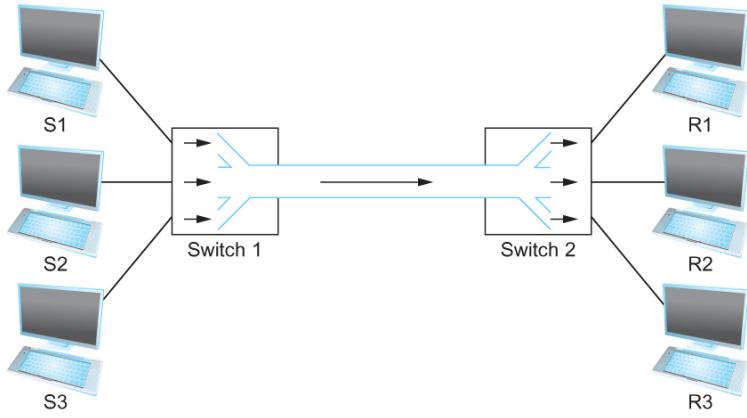
A mesh network requires N^2 connections → too costly

Connectivity



A shared infrastructure

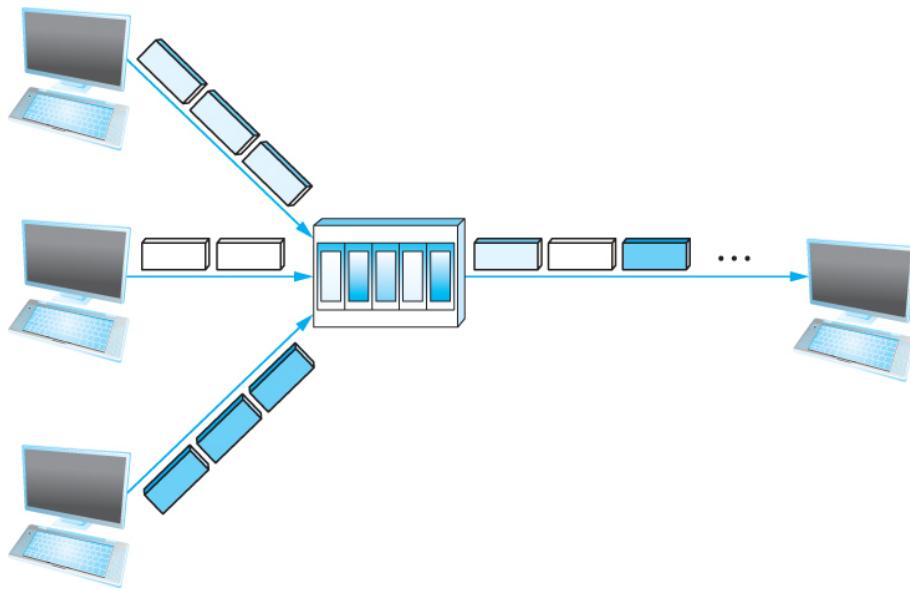
Cost-Effective Resource Sharing



Multiplexing multiple logical flows over a single physical link

- Resource: links and nodes
- How to share a link?
 - Multiplexing
 - De-multiplexing
 - Synchronous Time-division Multiplexing
 - Time slots/data transmitted in predetermined slots

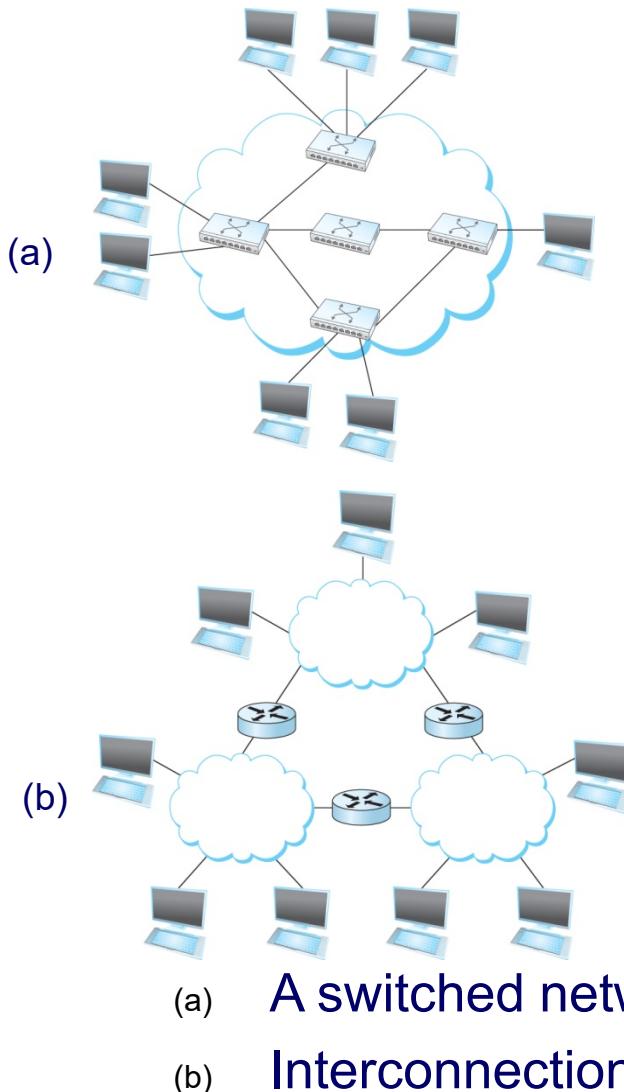
Cost-Effective Resource Sharing



A switch multiplexing packets from multiple sources onto one shared link

- FDM: Frequency Division Multiplexing
- Statistical Multiplexing
 - Data is transmitted based on demand of each flow.
 - What is a flow?
 - Packets vs. Messages
 - FIFO, Round-Robin, Priorities (Quality-of-Service (QoS))
 - Congested?
- LAN, MAN, WAN
- SAN (System Area Networks)

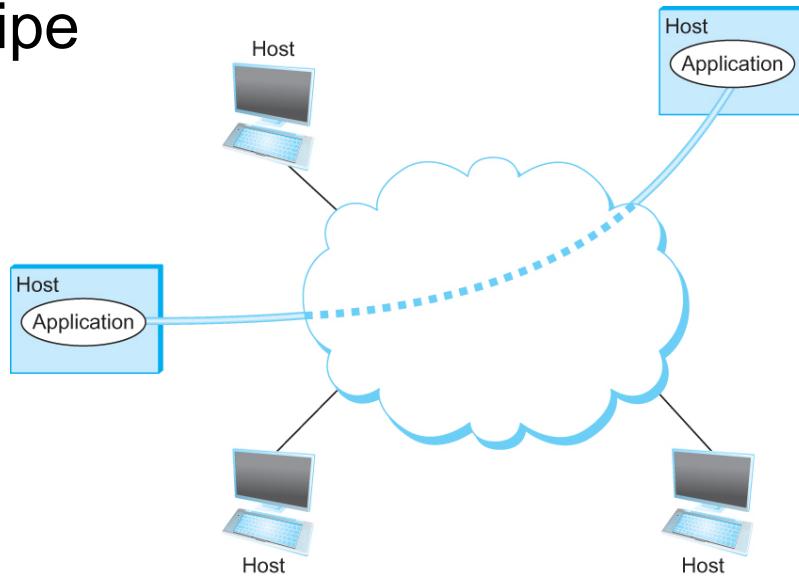
Connectivity



- Terminologies (contd.)
 - Cloud
 - Hosts
 - Switches
 - internetwork
 - Router/gateway
 - Host-to-host connectivity
 - Address
 - Routing
 - Unicast/broadcast/multicast

Support for Common Services

- Logical Channels (via TCP)
 - Application-to-Application communication path or a pipe



Process communicating over an abstract channel

Reliability

- Network should hide the errors
- Bits are lost
 - Bit errors (1 to a 0, and vice versa)
 - Burst errors – several consecutive errors
- Packets are lost (Congestion)
- Links and Node failures
- Messages are delayed
- Messages are delivered out-of-order
- Third parties eavesdrop

CPE348: Introduction to Computer Networks

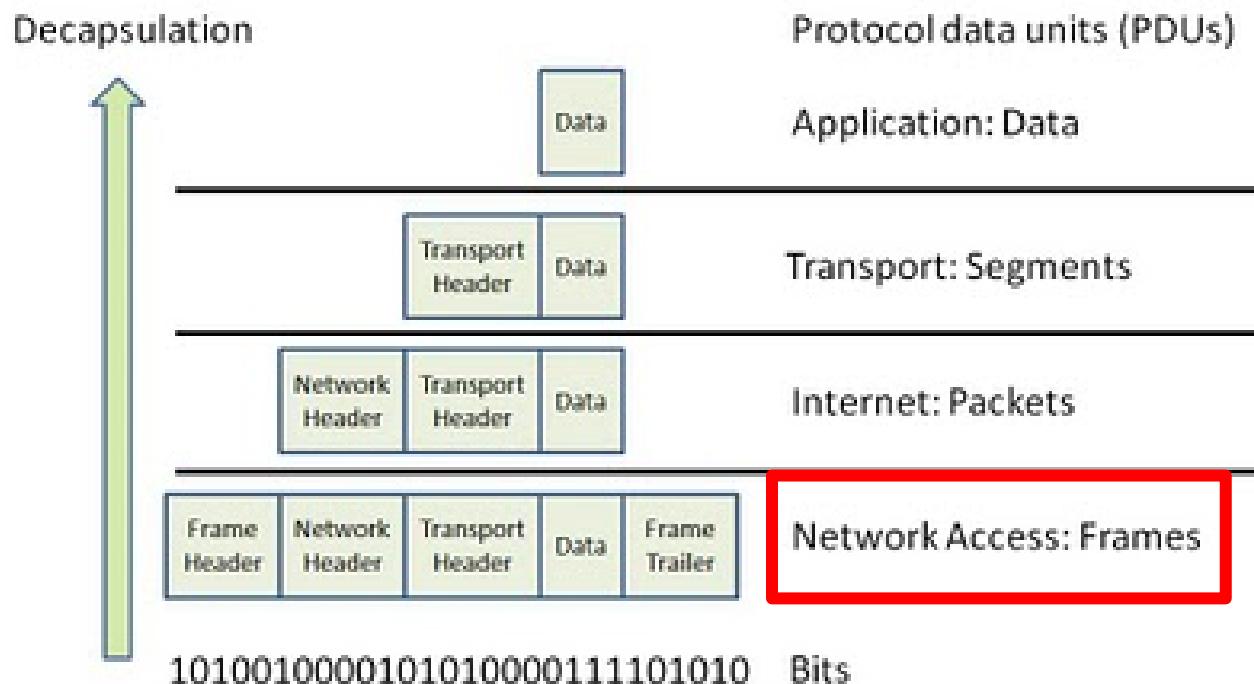
Lecture #5: Chapter 2.2



Jianqing Liu
Assistant Professor of Electrical and Computer
Engineering, University of Alabama in Huntsville
jianqing.liu@uah.edu
<http://jianqingliu.net>

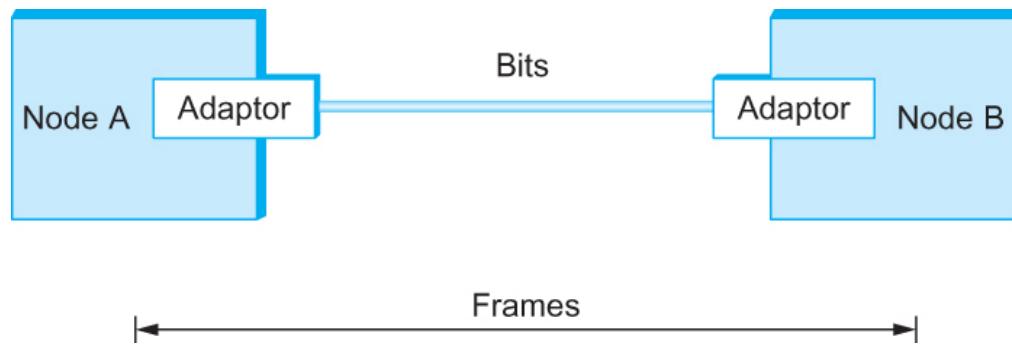
Framing

- What is the name of a data unit?



Framing

- Blocks of data (i.e., **frames**), not bit streams, are exchanged between nodes.
- It is the **network adaptor** that enables the nodes to exchange frames.



Bits flow between adaptors, frames between hosts

Framing

- What is important?
 - determining where the frame begins and ends
 - Distinguishing data payload and header



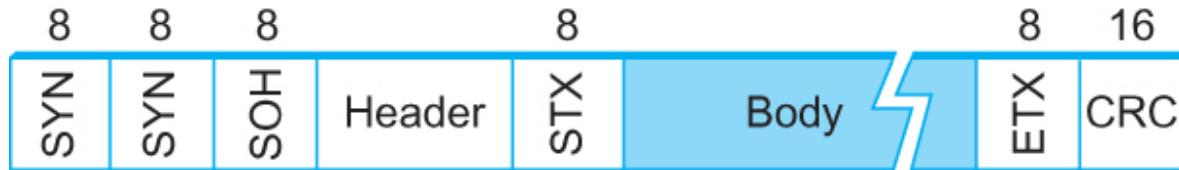
Framing

- Byte-oriented Protocols
 - To view each frame as a collection (i.e., **multiplier**) of bytes (characters) rather than bits.
 - Three examples
 - BISYNC (Binary Synchronous Communication) Protocol
 - PPP (Point-to-Point Protocol)
 - DDCMP (Digital Data Communication Protocol)

Framing

- BISYNC – sentinel approach
 - Frames transmitted beginning with leftmost field
 - Begin with a SYN (synchronize) character
 - Data portion is between special sentinel character STX (start of text) and ETX (end of text)
 - SOH : Start of Header
 - DLE : Data Link Escape
 - Used to precede a data portion that is equivalent to ETX
 - Used to precede a data portion that is equivalent to DLE
 - CRC: Cyclic Redundancy Check

Framing

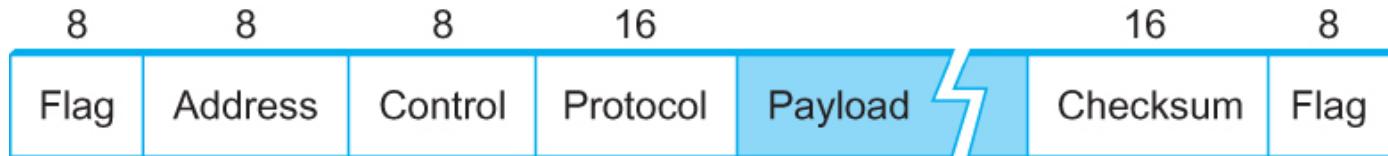


BISYNC Frame Format

Framing

- Point-to-point protocol (PPP)
 - Special start of text character denoted as Flag
0 1 1 1 1 1 1 0
 - Address, control : default numbers
 - Protocol for demux : IP / IPX
 - Payload : negotiated (default:1500 bytes)
 - Checksum : for error detection
 - Several field sizes negotiated – Link control protocol (LCP)

Framing

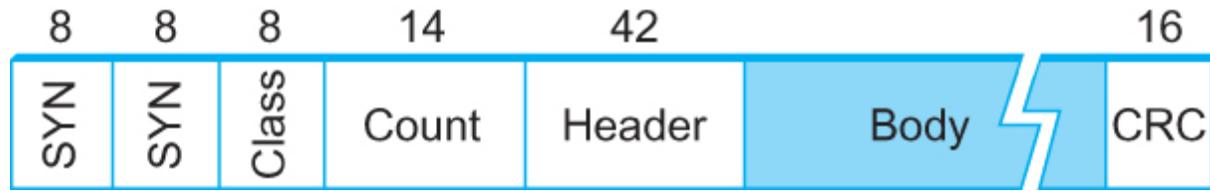


PPP Frame Format

Framing

- DDCMP (Digital Data Communication Message Protocol)
 - *count* : how many bytes are contained in the frame body
 - If *count* is corrupted
 - Framing error
 - Usually results in back-to-back frames received in error

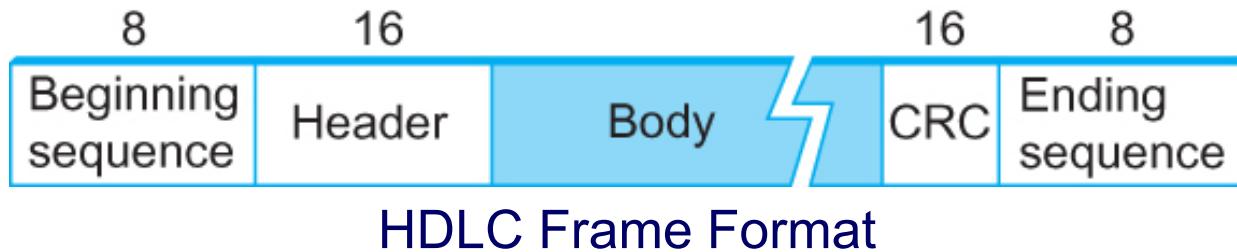
Framing



DDCMP Frame Format

Framing

- Bit-oriented Protocol
 - HDLC : High Level Data Link Control
 - Beginning and Ending Sequences
 - 0 1 1 1 1 1 0
 - Send 0 1 1 1 1 1 0 when idle



Framing

- HDLC Protocol – Sending Side
 - any time five consecutive 1's transmitted from the body of the message
 - The sender inserts 0 before transmitting the next bit (**bit stuffing**)



Framing

- HDLC Protocol - On the receiving side
 - Receive 5 consecutive 1's
 - Next bit 0 : Stuffed, so discard it, continue with data
 - Next bit 1 : Either End of the frame marker,
Or Error introduced in the bitstream
 - Look at the next bit after the 6th 1
 - If 0 (01111110) → End of the frame marker
 - If 1 (01111111) → Error, discard the whole frame
 - The receiver needs to wait for next 01111110 before it can start receiving again

Error Detection

- Bit errors are introduced into frames
 - Because of electrical interference
 - Because of thermal noises
- Need a mechanism to
 - Detect any errors
 - Notify of or correct errors

Error Detection

- Two approaches when the recipient detects an error
 - Notify the sender that the message was corrupted, so the sender can send again.
 - Repeat re-transmission till correctly receiving it or time expiration
 - Using some error **correct detection** and **correction** algorithm, the receiver reconstructs the message

Error Detection

- Common techniques for detecting transmission error
 - CRC (Cyclic Redundancy Check)
 - Used in HDLC, DDCMP, CSMA/CD, Token Ring
 - Two Dimensional Parity (BISYNC)
 - Checksum (IP)

Error Detection is implemented in different OSI layers!

Why?

Error Detection

- Basic Idea of Error Detection
 - Add redundant information to a frame that can be used to determine if errors have been introduced
 - Extreme Case: Transmitting two complete copies of data
 - Identical → No error
 - Differ → Error
 - Poor Scheme? – not very efficient
 - n bit message, n bit redundant information
 - Error can go undetected
 - In general, we can provide strong error detection technique
 - k redundant bits, n bits message, $k \ll n$
 - In Ethernet, a frame carrying up to 12,000 bits of data requires only 32-bit CRC

Error Detection

- Extra bits are redundant
 - Derived from the original message using some **algorithms**
 - Both the sender and receiver know the algorithm

Sender

m	r
----------	----------

Receiver

m'	r'
------------------------	------------------------

Receiver receives m' and r' computes r using m'

If the computed r matches the received r' , then message is assumed to be error free

Two-dimensional parity

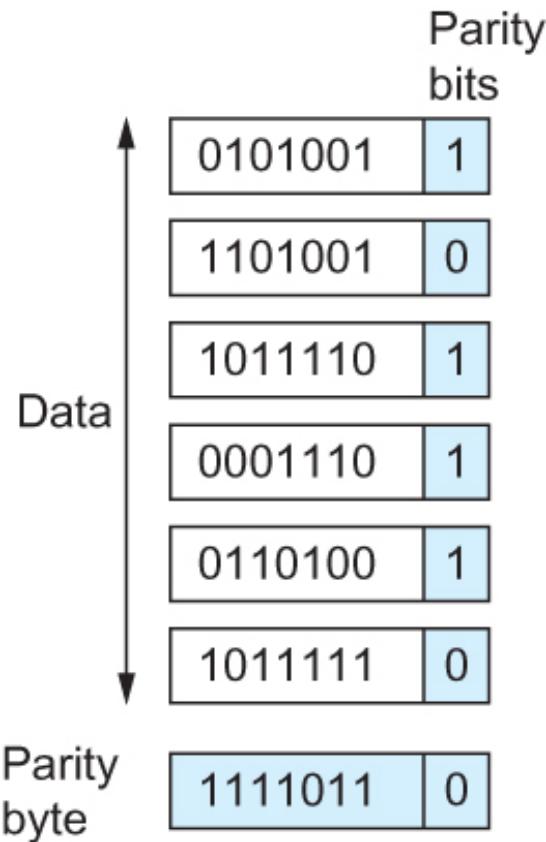
- Two-dimensional parity is exactly what the name suggests.
- One-dimensional parity is,
 - adding one extra bit to a 7-bit code
 - Odd parity sets the eighth bit to 1 if needed to give an odd number of 1s in the byte, and
 - Even parity sets the eighth bit to 1 if needed to give an even number of 1s in the byte



Two-dimensional parity

- Two-dimensional parity catches all 1-, 2-, and 3-bit errors and most 4-bit errors
- Or, how many bit errors does the 2-D parity code guarantee to detect?
- In the following example, 42 bits of data and 14 bits of parity (redundant bits)

Two-dimensional parity



Two Dimensional Parity (using even parity)

Internet Checksum Algorithm

- Used at the **network level**
- Add up all the words that are transmitted and then transmit the result of that sum
 - The result is called the checksum
- The receiver performs the same calculation on the received data and compares the result with the received checksum

Internet Checksum Algorithm

- Consider the data being checksummed as a sequence of 16-bit integers.
- Add them together using 16-bit ones complement and then take the ones complement of the result.
- Not a good detector of bit errors – however it is easy to implement in software

**Why not use the same
error detection technique in Layer 2?**

Internet Checksum Algorithm

- In ones complement arithmetic, a negative integer $-x$ is represented as the complement of x :
 - Each bit of x is inverted.
- When adding numbers in ones complement arithmetic, a carryout from the most significant bit needs to be added to the result.

Internet Checksum Algorithm

- Consider, for example, the addition of -5 and -3 in ones complement arithmetic on 4-bit integers
 - $+5$ is 0101, so -5 is 1010; $+3$ is 0011, so -3 is 1100
- If we add 1010 and 1100 ignoring the carry, we get 0110
- In ones complement arithmetic, the fact that this operation caused a carry from the most significant bit causes us to increment the result, giving 0111, which is the ones complement representation of -8 (obtained by inverting the bits in 1000), as we would expect

Cyclic Redundancy Check (CRC)

- Reduce the number of extra bits and maximize protection
- Given a bit string 110001 we can associate a polynomial on a single variable x for it.

$1*x^5 + 1*x^4 + 0*x^3 + 0*x^2 + 0*x^1 + 1*x^0 = x^5 + x^4 + 1$ and the degree is 5.

A k -bit frame has a maximum degree of $k-1$

- Let $M(x)$ be a message polynomial and $C(x)$ be a generator polynomial – pulled from a table.

Cyclic Redundancy Check (CRC)

- Let $M(x)/C(x)$ leave a remainder of 0.
- When $M(x)$ is sent and $M'(x)$ is received we have $M'(x) = M(x) + E(x)$
- The receiver computes $M'(x)/C(x)$ and if the remainder is nonzero, then an error has occurred.
- The only thing the sender and the receiver should know is $C(x)$.

Cyclic Redundancy Check (CRC)

- Let $M(x)$ be a frame with m bits and let the generator polynomial have less than m bits say equal to k .
- Let r be the degree of $C(x)$ where $r = k-1$
- Append r zero bits to the low-order end of the frame, so it now contains $m+r$ bits and corresponds to the polynomial $x^r M(x)$.

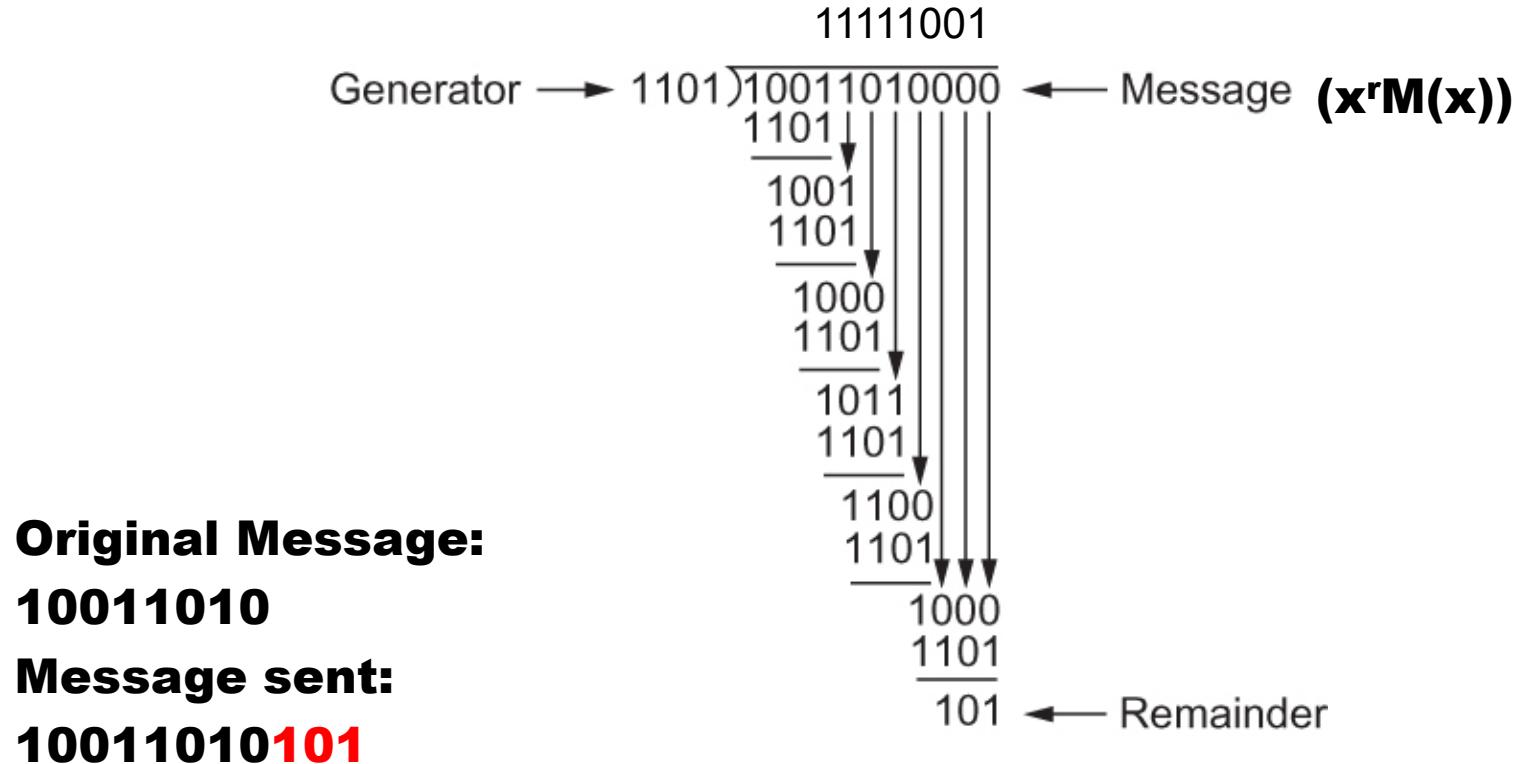
If $M(x) = x^5 + x^4 + 1$ and $C(x) = x^3 + x^2 + 1 \rightarrow r=3$

Then $x^r M(x) = x^8 + x^7 + x^3$

Cyclic Redundancy Check (CRC)

- Divide the bit string corresponding to $x^r M(x)$ by the bit string corresponding to $C(x)$ using modulo 2 division.
- Subtract the remainder (which is always r or fewer bits) from the string corresponding to $x^r M(x)$ using modulo 2 subtraction (addition and subtraction are the same in modulo 2). XOR
- The result is to be transmitted. Call it polynomial $M'(x)$.

Cyclic Redundancy Check (CRC)



CRC Calculation using Polynomial Long Division

Cyclic Redundancy Check (CRC)

- Properties of Generator Polynomial
 - It is possible to detect the following types of errors by a $C(x)$ with degree r
 - All single-bit errors, as long as the x^r and x^0 terms have nonzero coefficients.
 - All double-bit errors, as long as $C(x)$ has a factor with at least three terms.
 - Any odd number of errors, as long as $C(x)$ contains the factor $(x+1)$.
 - Any “burst” error (i.e., sequence of consecutive error bits) for which the length of the burst is less than r bits. (Most burst errors of larger than r bits can also be detected.)

Cyclic Redundancy Check (CRC)

- Six generator polynomials that have become international standards are:
 - $\text{CRC-8} = x^8 + x^2 + x + 1$
 - $\text{CRC-10} = x^{10} + x^9 + x^5 + x^4 + x + 1$
 - $\text{CRC-12} = x^{12} + x^{11} + x^3 + x^2 + x + 1$
 - $\text{CRC-16} = x^{16} + x^{15} + x^2 + 1$
 - $\text{CRC-CCITT} = x^{16} + x^{12} + x^5 + 1$
 - $\text{CRC-32} = x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1$

CPE348: Introduction to Computer Networks

Lecture #6: Chapter 2.3



Jianqing Liu
Assistant Professor of Electrical and Computer
Engineering, University of Alabama in Huntsville
jianqing.liu@uah.edu
<http://jianqingliu.net>

Reliable Transmission - Review

- CRC, checksum are used to detect errors.
- If errors,
 - corrupted frames must be **discarded**;
 - correct frames must be **recovered** or retransmitted.



Reliable Transmission - Motivation

- Error correction code (ECC) like interleaving and Turbo code typically has high overhead (but still in use)!
- An alternative: **re-transmission**
 - A link-level mechanism: **Acknowledgement**



Reliable Transmission

- An *acknowledgement* (ACK) is a small control frame saying that it has received the earlier frame.
 - ACK frame only has header (no data)
 - ACK and non-ACK
- The receipt of an ACK indicates that the prior frame was successfully delivered.

Reliable Transmission

But, the ACK could get lost/corrupted too!

Reliable Transmission

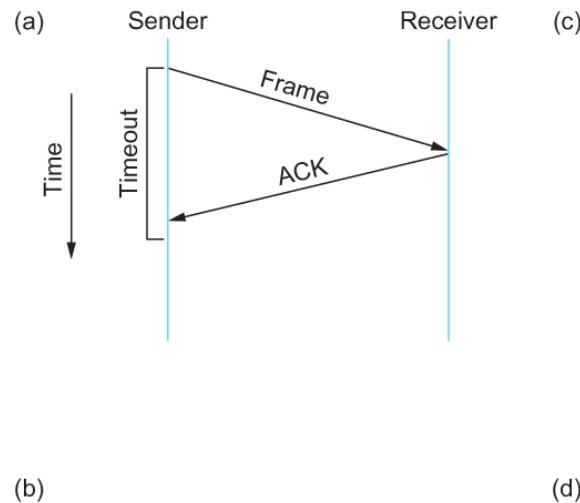
- A link-level mechanism: **Timeout**
 - If ACK is not received after a while, then the sender retransmits the original frame.

The combination of **ACK** and **Timeout** to implement reliable delivery is called **Automatic Repeat reQuest (ARQ)**.

ARQ - Stop and Wait Protocol

- One exemplary ARQ protocol - stop-and-wait
 - After transmitting one frame, the sender waits for an **ACK** before transmitting the next one.
 - If the ACK does not arrive after **Timeout**, the sender retransmits the prior frame.

ARQ - Stop and Wait Protocol

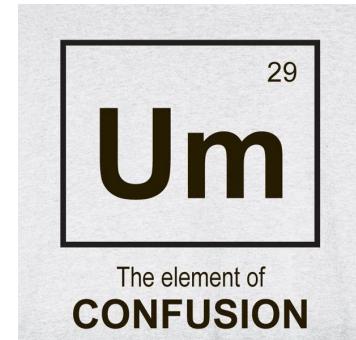


Timeline showing four different scenarios for the stop-and-wait algorithm.

(a) The ACK is received before the timer expires; (b) the original frame is lost; (c) the ACK is lost; (d) the timeout fires too soon

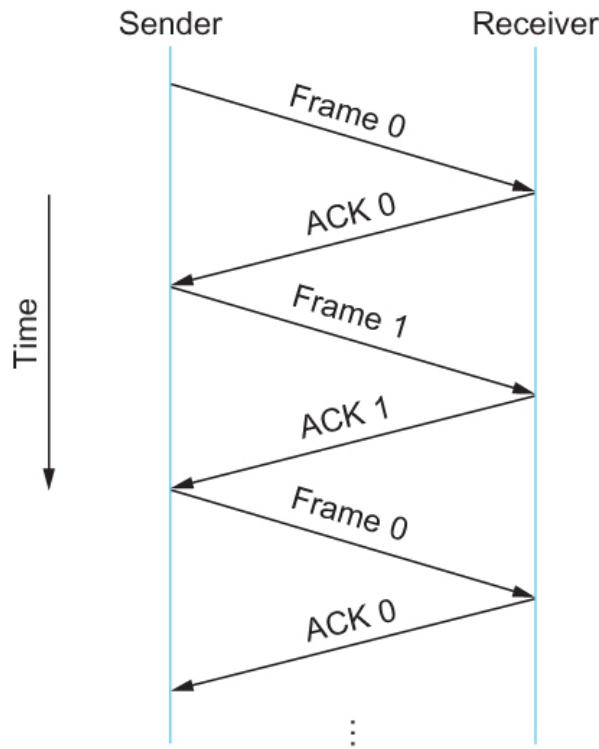
Stop and Wait Protocol – issue 1

- **Misunderstanding:** when the ACK is lost or delayed
 - The sender re-transmits;
 - The receiver gets a duplicate copy, which causes confusion



Stop and Wait Protocol – issue 1

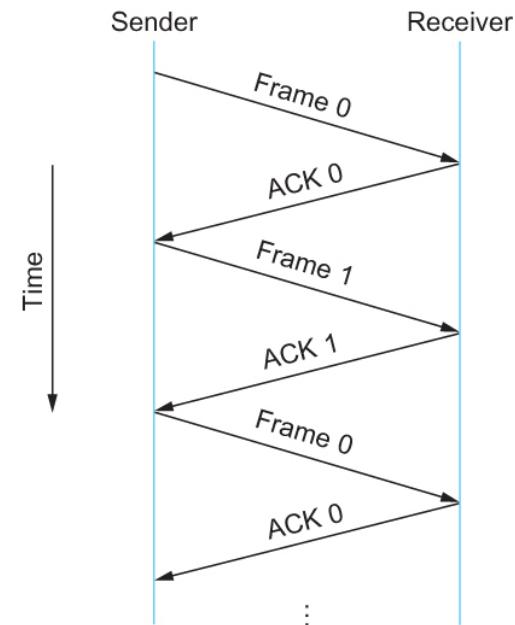
- Solution
 - Use 1 bit sequence number (0 or 1) to represent a frame



Stop and Wait Protocol – issue 2

- **Efficiency**: the sender has only one outstanding frame on the link at a time
 - poor utilization of channel capacity
 - Sending rate = (bits per frame)/(time per frame = 1 RTT)

- **Question**: what is the effective data rate in this diagram?



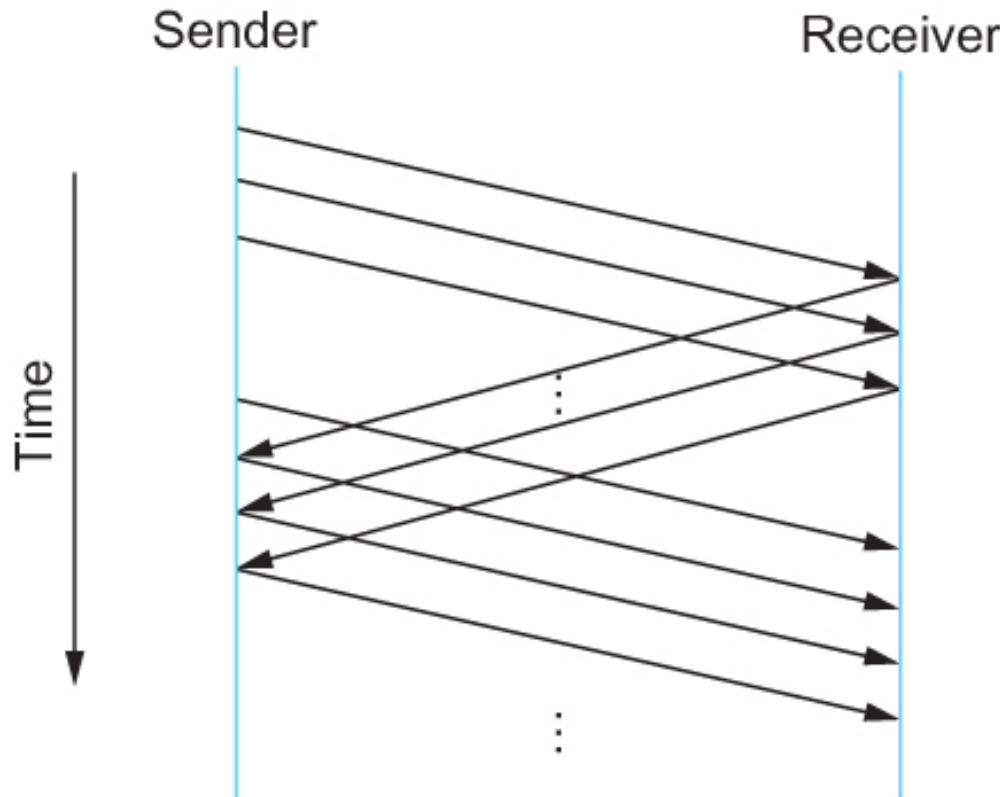
Stop and Wait Protocol – issue 2

- Question cont'

Consider sending a 1 KB frame over a 1.5 Mbps link with a 45 ms RTT

- Since the sender can send only one frame per RTT
 - Maximum Sending rate is
$$\text{Bits per frame} \div \text{Time per frame} = 1024 \times 8 \div 0.045 = 182 \text{ Kbps}$$
Or about one-eighth of the link's capacity
- An alternative solution
 - delay \times bandwidth product:
$$1.5E6 \times 0.045 = 67,500 \text{ bits} = 8,437 \text{ Bytes}$$
, or about eight-times of the frame size

Sliding Window Protocol



The solution to improving efficiency of
stop-and-wait protocol

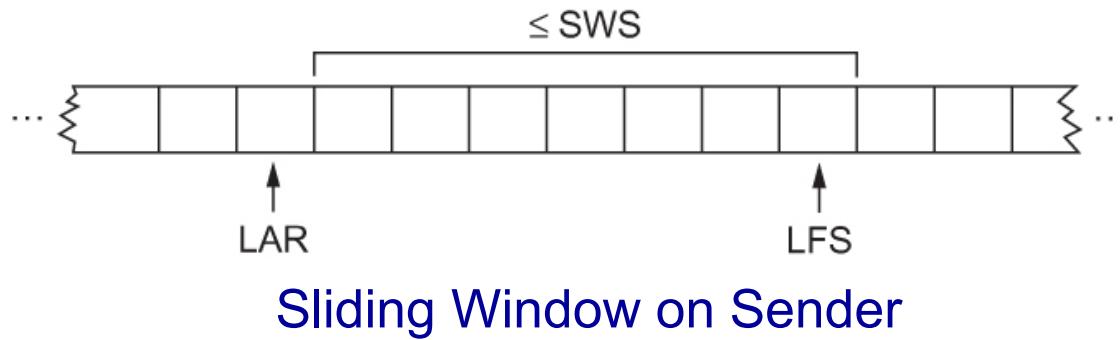
Sliding Window Protocol - Sender

- Sender assigns a **sequence number** to a frame
- Sender maintains three pointers
 - Sending Window Size (**SWS**)
 - Last Acknowledgement Received (**LAR**)
 - Last Frame Sent (**LFS**)

Sliding Window Protocol - Sender

- One property of these pointers:

$$LFS - LAR \leq SWS$$



- LAR moves right as ACKs received
- LFS moves right as frames are sent

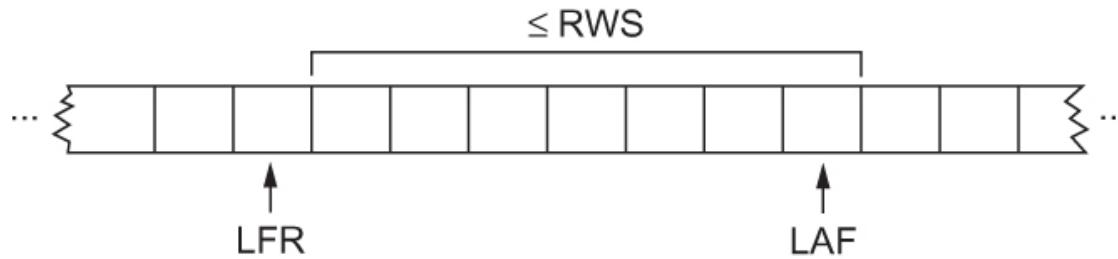
Sliding Window Protocol - Rcvr

- Receiver maintains three pointers
 - Receiving Window Size (**RWS**)
 - Largest Acceptable Frame (**LAF**)
 - Last Frame Received (**LFR**)

Sliding Window Protocol - Rcvr

- Also, one property of these pointers

$$\text{LAF} - \text{LFR} \leq \text{RWS}$$



Sliding Window on Receiver

Sliding Window Protocol - Rcvr

- When a frame arrives, what does the receiver do?

Check

- If $\text{SeqNum} \leq \text{LFR}$ or $\text{SeqNum} > \text{LAF}$
 - Discard it (the frame is outside the receiver window)
- If $\text{LFR} < \text{SeqNum} \leq \text{LAF}$
 - Accept it
 - Send an ACK

Sliding Window Protocol - Rcvr

- Keep a pointer **SeqNumToAck**
 - Denote the largest SeqNum *not yet ACKed*,
 - All frames with SeqNum less than SeqNumToAck have been received
- Even if higher-numbered packets have been received, the receiver holds the ACK till the frame SeqNumToAck is received.
- The receiver then sets
 - $LFR = \text{SeqNumToAck}$ and adjusts
 - $LAF = LFR + RWS$

Sliding Window Protocol – example

For example, suppose $LFR = 1$ and $RWS = 4$

(i.e. the last ACK that the receiver sent was for seq. no. 1 and $SeqNumToAck$ is set to 2)

→ $LAF = 5$

If frames 3 and 4 arrive before frame 2, they will be buffered because they are within the receiver window

But no ACK will be sent since frame 2 is yet to arrive

Frames 3 and 4 are out of order

Frame 2 arrives (it is late because it was lost first time and had to be retransmitted)

Receiver Acknowledges Frame 4

Receiver bumps LFR to 4

Receiver moves LAF to 8 ($LAF = LFR + RWS$)

Receiver sets $SeqNumToAck$ to 5

Sliding Window Protocol – issue 1

- When timeout occurs,
 - Sender waits and is unable to advance its window
- When the packet loss occurs, this scheme is no longer keeping the pipe full
 - The longer it takes to notice that a packet loss has occurred, the more severe the problem becomes
- How to improve this
 - Negative Acknowledgement (NAK)
 - Additional Acknowledgement
 - Selective Acknowledgement

Efficiency!

Sliding Window Protocol – issue 1

- Negative Acknowledgement (NAK)
 - Receiver sends NAK for frame 2 when frame 3 arrive (**solicitate**)
- Additional Acknowledgement
 - Receiver sends additional ACK for frame 1 when frame 3 arrives (**duplicate**)
- Selective Acknowledgement
 - Receiver will acknowledge exactly those frames it has received, rather than the highest number frames (**explicit**)
 - Receiver will acknowledge frames 3 and 4
 - Sender knows frame 2 is lost

Sliding Window Protocol – issue 2

How to select the window size

- SWS is easy to compute
 - Use $\text{Delay} \times \text{Bandwidth}/(\text{frame size})$ – keeps the pipe full
- RWS can be anything
 - Two common settings
 - RWS = 1, **OK!**
No buffer for frames that arrive out of order
 - RWS = SWS, **OK!**
The rcvr buffers frames that sender transmits
 - RWS > SWS. **NO!**
Why?

Sliding Window Protocol – issue 2

Before we present the issue,
let's first look into the SeqNum!

- Frame sequence number
 - Specified in the header
 - Finite size
 - 3 bits: eight possible sequence number: 0, 1, 2, 3, 4, 5, 6, 7
 - $\text{MaxSeqNum} = 8$
 - Wrap around – reuse sequence numbers

Sliding Window Protocol – issue 2

How does the MaxSeqNum affect SWS and RWS?

- Question: $SWS + 1 \leq \text{MaxSeqNum}$, is this sufficient?
 - Depends on RWS
 - If RWS = 1, then sufficient
 - If RWS = SWS, then not good enough
- Example, we have 4 sequence numbers: 0, 1, 2, 3
RWS = SWS = 3
Sender sends 0, 1, 2
Receiver receives 0, 1, 2 and ACKs 0, 1, 2
ACK (0, 1, 2) are lost
Sender re-transmits 0, 1, 2
Receiver is expecting 3, 0, 1

Confusion!

Sliding Window Protocol – issue 2

- To avoid confusion,
 - If $RWS = SWS$ (remember makes no sense for $RWS > SWS$)
 $SWS < (\text{MaxSeqNum} + 1)/2$ or $\text{MaxSeqNum} > 2*SWS - 1$
 - If $RWS < SWS$, then MaxSeqNum may be less than $2*SWS - 1$

In general, SWS should be no more than
a half of MaxSeqNum, but depends!

CPE348: Introduction to Computer Networks

Lecture #6: Chapter 2.4



Jianqing Liu
Assistant Professor of Electrical and Computer
Engineering, University of Alabama in Huntsville
jianqing.liu@uah.edu
<http://jianqingliu.net>

Ethernet – history

- Most successful networking technology of last 20 years.
- Developed in the mid-1970s by Palo Alto Research Centers (PARC), 10-Mbps Ethernet standard in 1978.
- Now, it has been extended to 100-Mbps version called Fast Ethernet, 1000-Mbps version called Gigabit Ethernet, 10 Gbps, 100 Gbps and more.



Ethernet – key technology

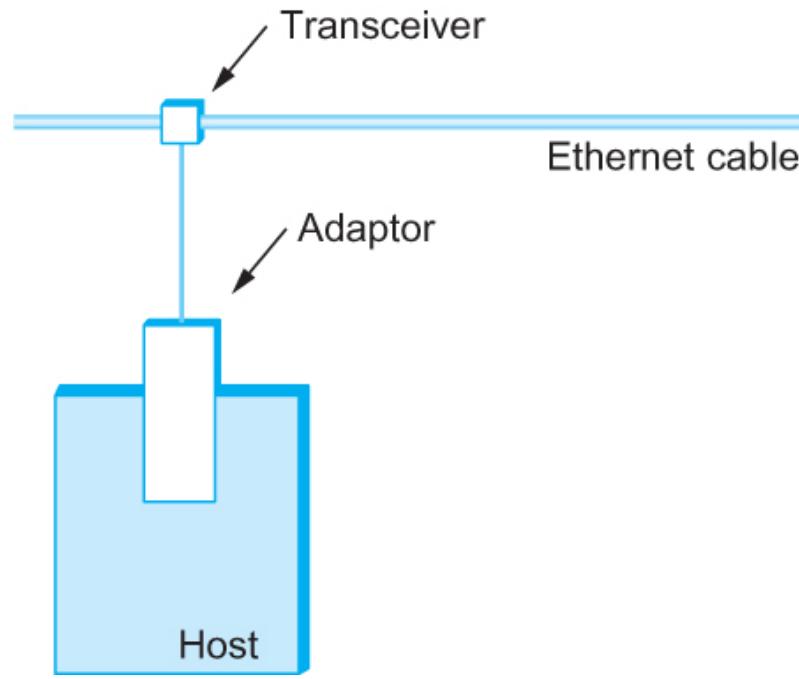
- Uses **CSMA/CD** technology
 - **Carrier Sense Multiple Access** with **Collision Detection**.
 - MA: A set of nodes send and receive frames over a shared link.
 - CS: all nodes can distinguish between an idle and a busy link.
 - CD: a node listens as it transmits and can therefore detect when a frame it is transmitting has collided with a frame transmitted by another node.
- Uses **ALOHA** (packet radio network) as the root protocol
 - Developed at the University of Hawaii to support communication across the Hawaiian Islands.
 - If link is idle, transmit the packet

Ethernet - adaptor

- A classical Ethernet segment is implemented on a coaxial cable of up to 500 m.
- A transceiver tapes the cable, transmits and receives signal.
- The transceiver is connected to an Ethernet adaptor which is plugged into the host.
- The protocol is implemented on the adaptor.



Ethernet - adaptor

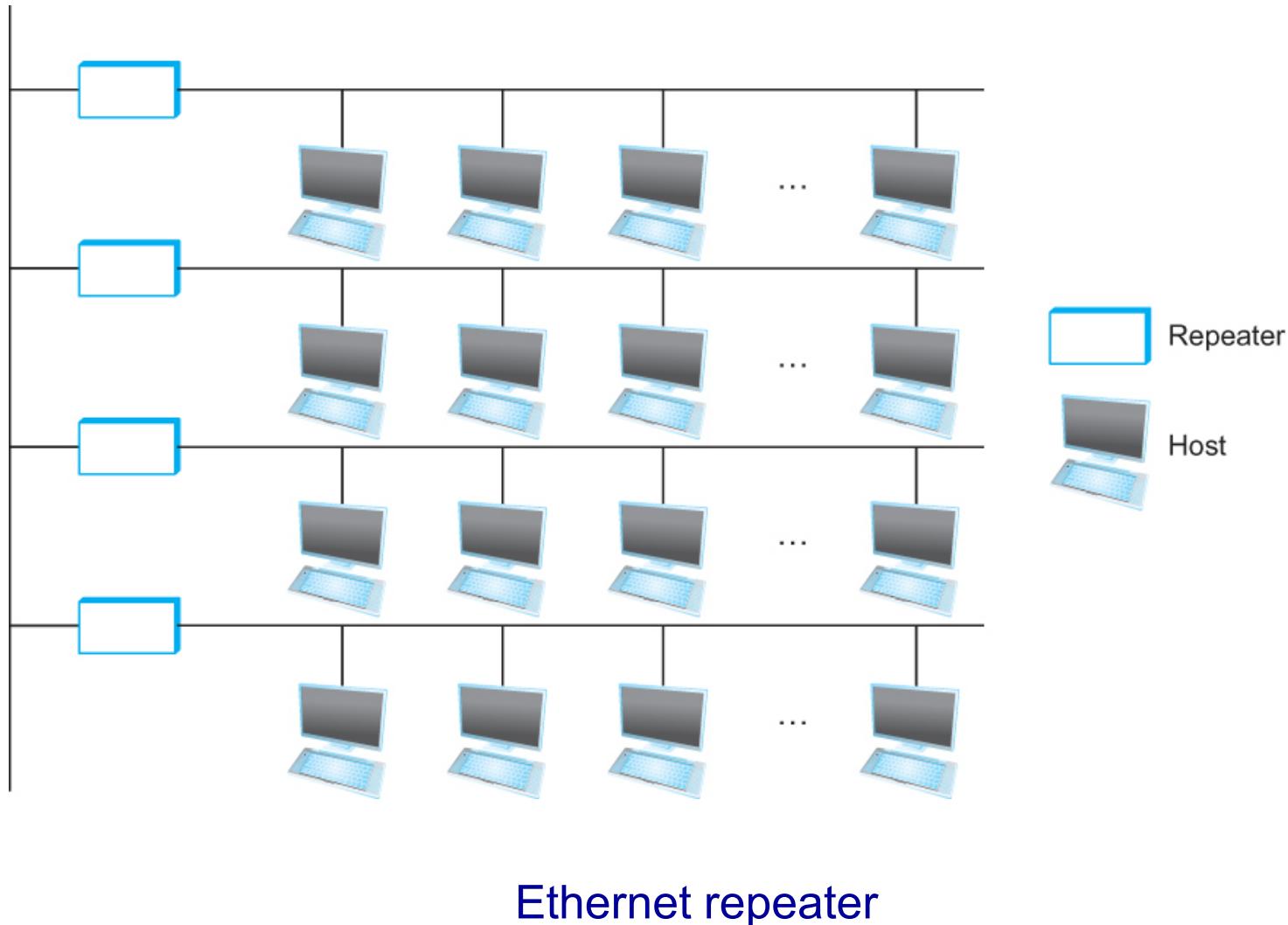


Ethernet transceiver and adaptor

Ethernet – repeater

- Multiple Ethernet segments can be joined together by *repeaters*.
- A *repeater* is a device that **relays** digital signals.
- No more than four repeaters may be positioned between any pair of hosts.
 - A classical Ethernet has a total reach of only 2500 m.
 - Maximum of 1024 hosts
- Modern Ethernets
 - use category 5 twisted copper pair (cat 6 or cat 7 for 10Gbps)
 - Use optical fibers
 - Can be longer than 500 meters between repeaters

Ethernet – repeater

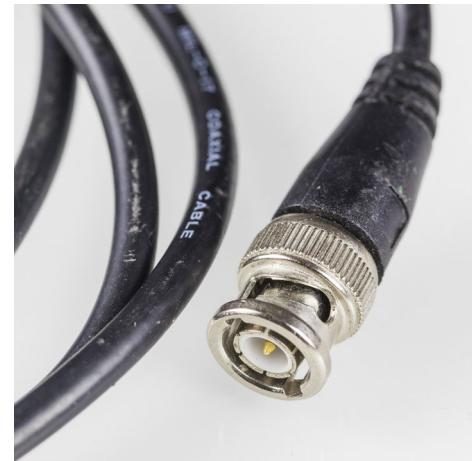


Ethernet – cont'

- Any signal placed on the Ethernet by a host is **broadcast over the entire network**
 - Signal is propagated in both directions.
 - Hosts can detect the signal from the cable.
- Classical Ethernet uses **Manchester encoding** scheme.
- Higher speed Ethernets use 4B/5B or 8B/10B encoding

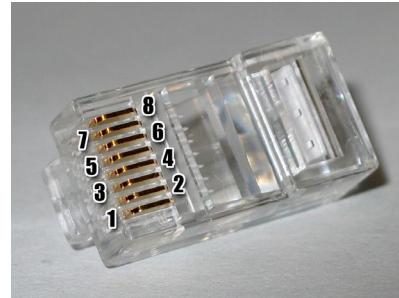
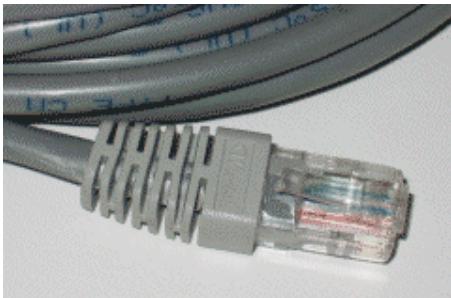
Ethernet – cont'

- New Technologies in Ethernet
 - Instead of using coax cable, an Ethernet can be constructed from a thinner cable known as **10Base2**
 - 10 means the network operates at 10 Mbps
 - Base means the cable is used in a baseband system
 - 2 means that a given segment can be no longer than 200 m



Ethernet – cont'

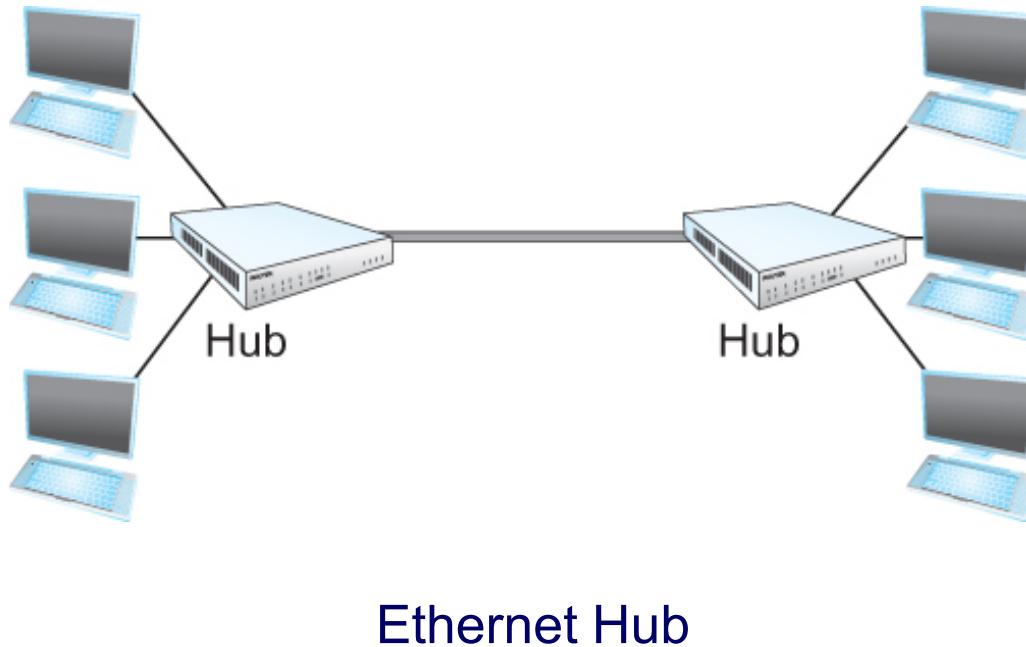
- New Technologies in Ethernet
 - Another cable technology is **10BaseT**
 - T stands for twisted pair
 - Limited to 100 m in length
 - With 10BaseT, the repeater has multiple outputs, called **Hub**



TIA/EIA-568 T568A termination

Pin	Pair	Wire ^[e]	Color
1	3	tip	white/green
2	3	ring	green
3	2	tip	white/orange
4	1	ring	blue
5	1	tip	white/blue
6	2	ring	orange
7	4	tip	white/brown
8	4	ring	brown

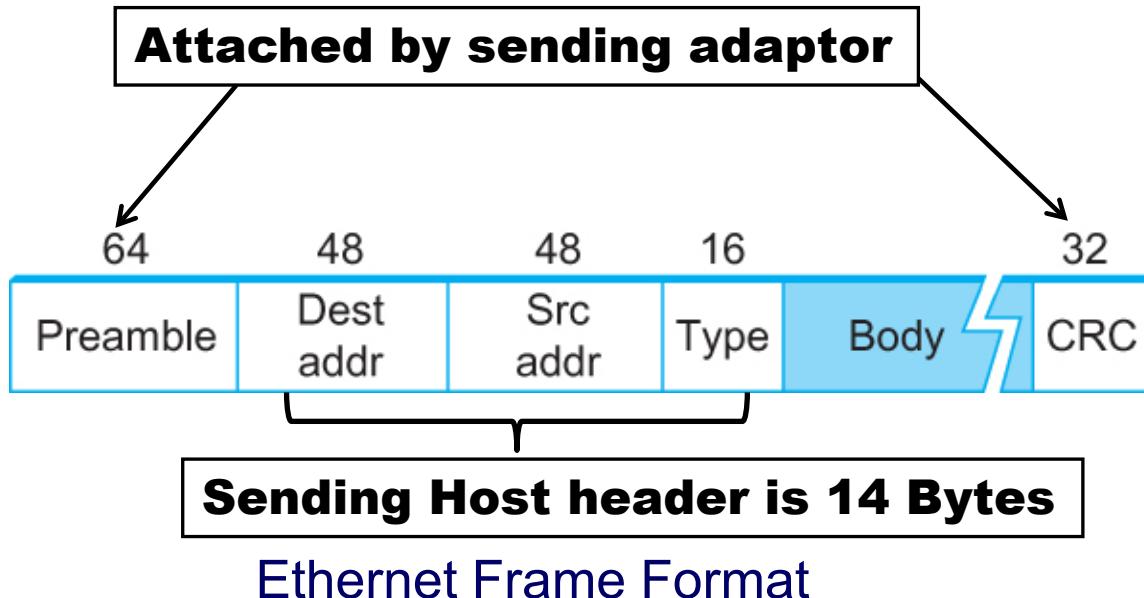
Ethernet – cont'



Ethernet – frame

- Frame format
 - Preamble (64bit or 8 Bytes): allows rcvr to synchronize.
 - Src and Dst Address (48bit or 6 Bytes each).
 - Packet type (16bit or 2 Bytes): acts as demux key to identify the higher level protocol.
 - Data (up to 1500 bytes)
 - Minimally a frame must contain at least 46 bytes of data. WHY?
 - Frame must be long enough to detect collision, but no more than 1,500 bytes to avoid always occupying the line.
 - CRC (32bit or 4 Bytes)

Ethernet – frame



[Link to Slide 22](#)

Ethernet – address

- Each host on an Ethernet has a unique Ethernet Address (e.g., IP, MAC address).
- The address belongs to the adaptor, not the host.



Ethernet – address

- MAC address is typically printed in a human readable format
 - As a sequence of **six numbers** separated by colons.
 - Each number is given by a pair of hexadecimal digits, one for each of the 4-bit nibbles in the byte
 - For example, 8:0:2b:e4:b1:2 is
 - 00001000 00000000 00101011 11100100 10110001 00000010

Ethernet – address

- MAC-48 is now being called EUI-48 (Extended Unique Identifier)
- 48 bits, so number of addresses: $2^{48} = 281.47E12!$
- It is projected to be exhausted in 2100!

Ethernet – address

IP address:

- *Unicast* address – one-to-one addressing
- *broadcast* address – one-to-all addressing
- *multicast* address – one-to-many addressing

We will elaborate on next
chapter
– network layer technology

Ethernet – access protocol

- It transmits the frame immediately when the line is idle.
- It holds the transmission when the line is busy.

Ethernet – access protocol – issue

- No coordination, so it is possible for two (or more) adaptors far-away to transmit at the same time,
- When this happens, the two (or more) frames are said to *collide* on the network.



Ethernet – access protocol – issue

How does CSMA/CD come into play?

- When an adaptor detects its frame colliding with another,
 - it first transmits a frame of 32-bit jamming sequence.
 - it then stops transmission.

Ethernet – access protocol – issue

- Worst case collision: two hosts are at opposite ends of the Ethernet.



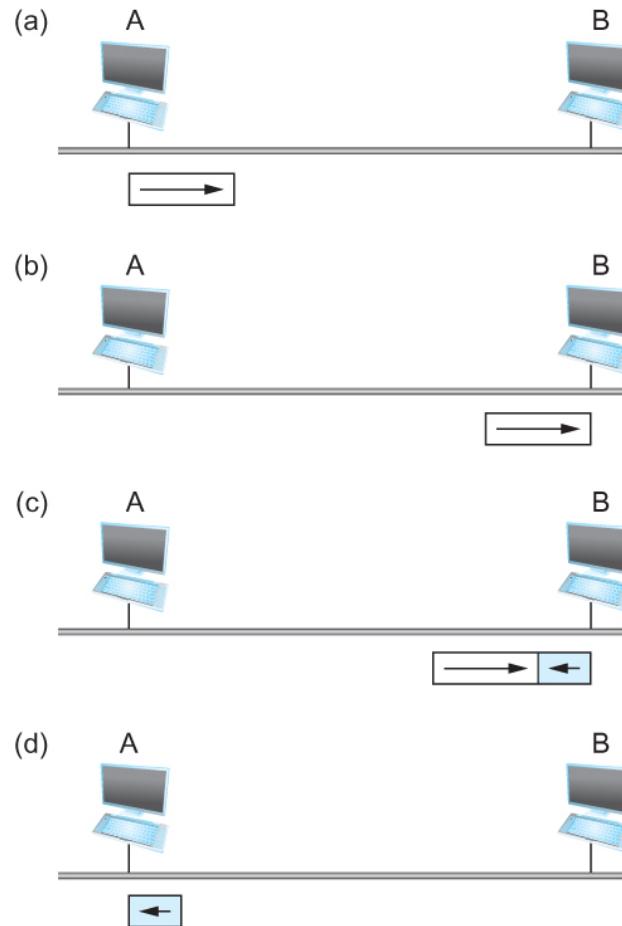
- Let's first put the design rule here:
 - Every Ethernet frame must be at least **512** bits (64 bytes) long.
 - 14 bytes of header + 46 bytes of data + 4 bytes of CRC
 - Not include the preamble of 8 Bytes – otherwise minimum frame is 576 bits

Ethernet – access protocol – issue

Why 512 bits? Why is its length limited to 2500 m?

- The farther apart, the longer it takes for transmitting a frame.
- On 10 Mbps Ethernet,
 - Round trip delay of transmitting 512-bit frame is 51.2 μ s for 2500 meter length and 4 repeaters

Ethernet – access protocol – issue



Worst-case scenario: (a) A sends a frame at time t ; (b) A's frame arrives at B at time $t + d$; (c) B begins transmitting at time $t + d$ and collides with A's frame; (d) B's runt (32-bit) frame arrives at A at time $t + 2d$.

Ethernet – access protocol

- Once detecting a collision,
 - transmission is stopped,
 - wait a certain amount of time,
 - try again.
- After several trials,
 - double waiting time,
 - try again

Exponential Backoff

Ethernet – access protocol

- At first collision,
 - the adaptor delays either 0 or 51.2 μ s, selected at random.
- At second collision,
 - it waits 0, 51.2, 102.4, 153.6 μ s (selected randomly);
 - This is $k * 51.2$ for $k = 0, 1, 2, 3$.
- At third collision,
 - it waits $k * 51.2$ for $k = 0 \dots 2^3 - 1$ (selected at random).

Ethernet – access protocol

- In general, the algorithm
 - randomly selects a k between 0 and $2^n - 1$;
 - and waits for $k * 51.2 \mu\text{s}$;
 - n is the number of collisions experienced so far.
- After a successful transmission,
 - n may be reset to 0
 - or reduced by some factor (1, $\frac{1}{2}$, $\frac{1}{4}$, etc)

Ethernet – some experience

- Ethernets work best under **lightly loaded** conditions.
 - Under heavy loads (typically $>30\%$ utilization), too much of the network's capacity is wasted by collisions.
- Most Ethernets are used in a **conservative way**.
 - Have fewer than 200 hosts connected to them which is far fewer than the maximum of 1024.
- Hosts can plug-and-play.

CPE348: Introduction to Computer Networks

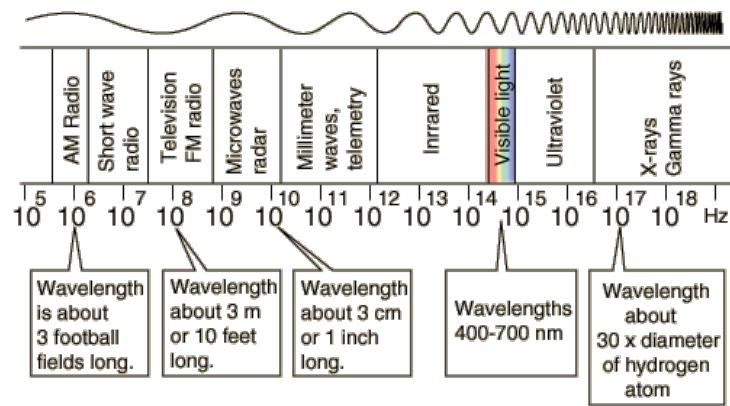
Lecture #7: Chapter 2.5



Jianqing Liu
Assistant Professor of Electrical and Computer
Engineering, University of Alabama in Huntsville
jianqing.liu@uah.edu
<http://jianqingliu.net>

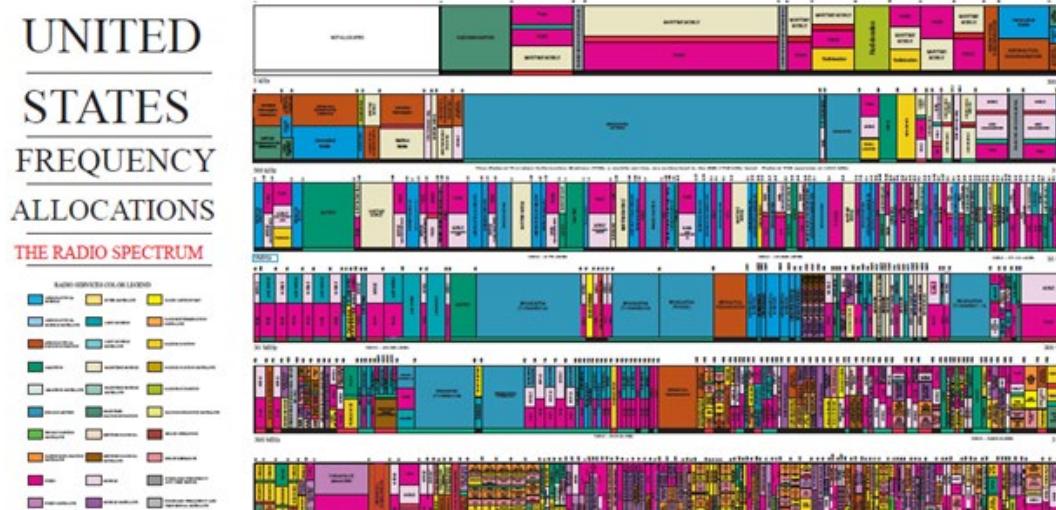
Wireless Links

- Wireless data transmission is based on **EM wave** propagation in the **free space**.



Wireless Links

- A Specific frequency band is allocated to a specific entity.
- These allocations are determined by FCC (Federal Communications Commission) in USA.



Wireless Networks

- Several wireless networks
 - Bluetooth (802.15)
 - Wi-Fi (more formally known as 802.11)
 - 2G, 3G, 4G/LTE cellular systems and beyond



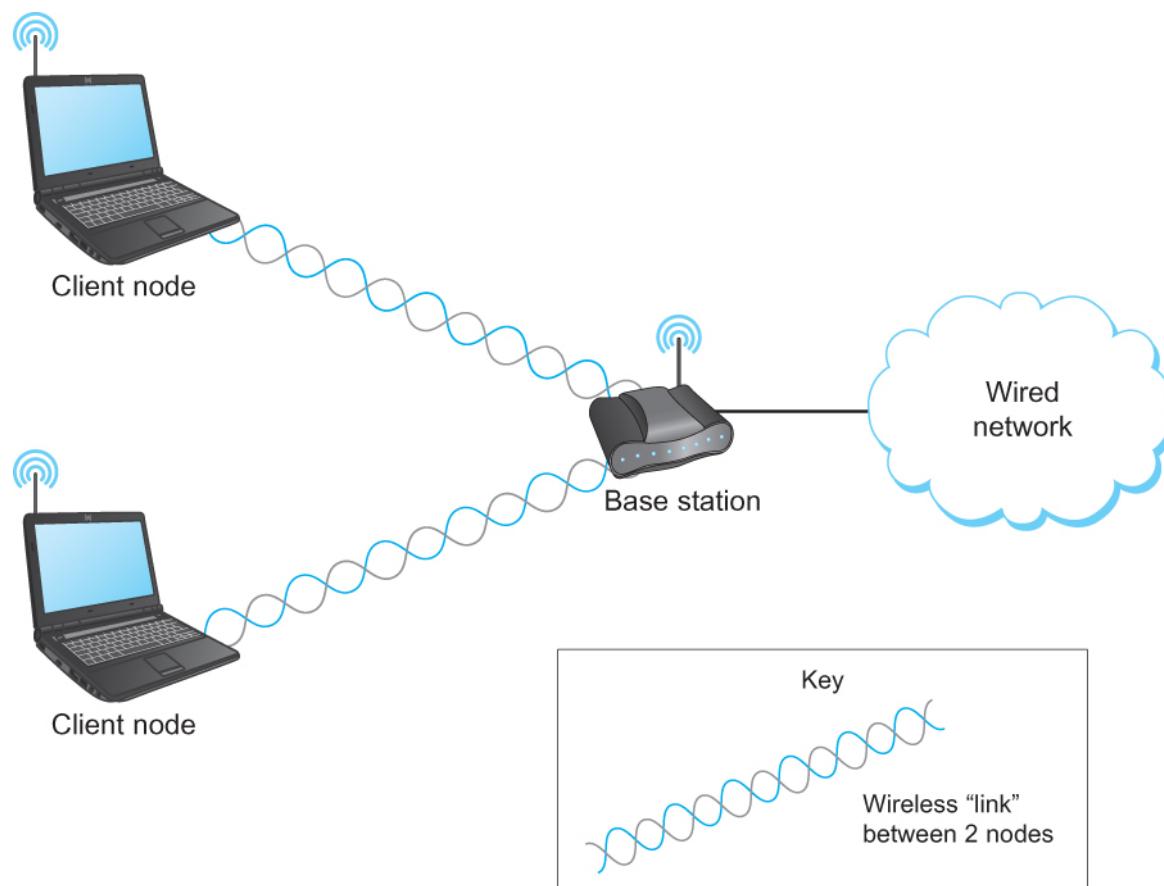
Wireless Networks

- Considerations upon designing or updating a wireless system:
 - Connectivity
 - data rate (uplink & downlink)
 - Latency
 - Energy efficiency
- How to achieve these specs:
 - Architecture design
 - Resource allocation (bandwidth, power, time, device)
 - Scheduling and control design

Capability	5G target
Peak data rate	20 Gbit/s
User experienced data rate	1 Gbit/s
Latency	1 ms
Mobility	500 km/h
Connection density	$10^6/\text{km}^2$
Energy efficiency	Equal to 4G
Spectrum efficiency	3–4x 4G
Area traffic capacity	1000 (Mbit/s)/m ²

Computer Engineer!

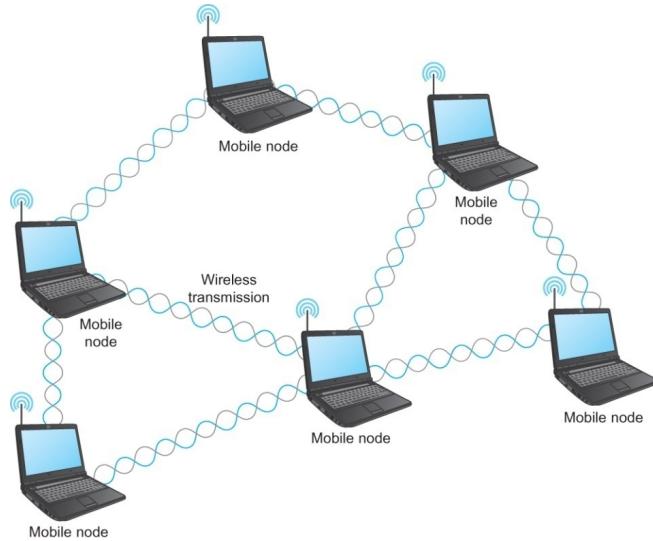
Wireless Networks – centralized



A wireless network using a base station, e.g. cellular

Wireless Networks – distributed

- Mesh or Ad-hoc network
 - Nodes are peers
 - Messages may be forwarded via a chain of peer nodes
 - Multiple paths are available



A wireless mesh network, e.g. wireless sensor network

Wireless Networks

Too many wireless networks!

Let's just study WiFi as an example!



IEEE 802.11 – overview

- Also known as Wi-Fi
- 802.11 defines a suite of protocols to build a wireless local area network (WLAN)
- Its version evolves to support different applications.



IEEE 802.11 – history

- Original 802.11 standard defined two radio-based physical layer standard
 - One using the frequency hopping
 - Over 79 1-MHz-wide frequency bandwidths
 - Second using direct sequence
 - Using 11-bit chipping sequence
 - Both standards run in the 2.4-GHz and provide up to 2 Mbps

IEEE 802.11 – history

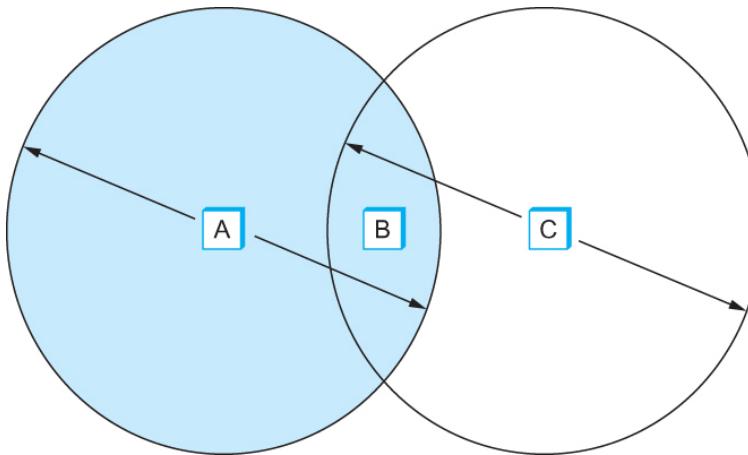
- Then physical layer standard **802.11b** was added
 - Using a variant of direct sequence 802.11b provides up to 11 Mbps
 - Uses license-exempt 2.4-GHz band
- Then came **802.11a** which delivers up to 54 Mbps using OFDM (Orthogonal FDM)
 - Runs on license-exempt 5-GHz band – less interference
- Then came **802.11g** which is backward compatible with 802.11b
 - Uses 2.4 GHz band, OFDM and delivers up to 54 Mbps
- Then came **802.11n** which delivers up to 600 Mbps
 - Uses multiple antennas – MIMO (multiple input multiple output)
- Story continues...

IEEE 802.11 – Collision Avoidance

Let's look into an important issue in WiFi networks!

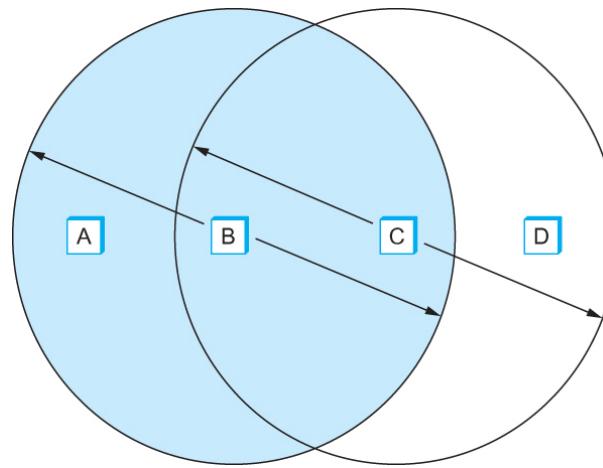
IEEE 802.11 – Collision Avoidance

- Suppose both A and C want to communicate with B
 - A and C are unaware of each other
 - These two frames collide with each other at B
 - But unlike an Ethernet, neither A nor C is aware of this collision
- A and C are said to *hidden nodes* with respect to each other – see next slide



IEEE 802.11 – Collision Avoidance

- Another problem called *exposed node* problem occurs
 - Suppose B is sending to A. Node C is aware of this communication
 - Suppose C wants to transmit to node D.
 - It would be a mistake for C to conclude that it cannot transmit.
 - Waste of resources!



IEEE 802.11 – Collision Avoidance

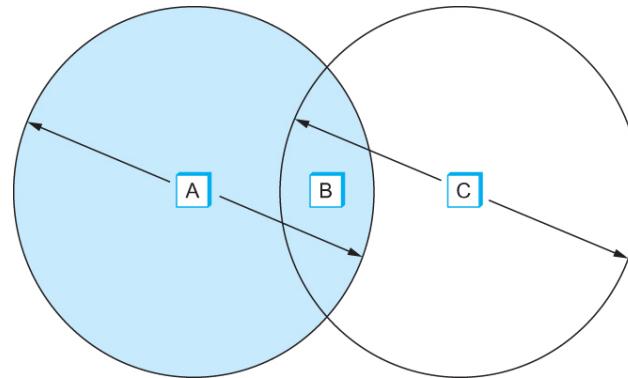
- 802.11 addresses these two problems with an algorithm called Multiple Access with Collision Avoidance (**MACA**).
 - Sender and receiver exchange control frames with each other before data communications;
 - This exchange informs all nearby nodes that a transmission is about to begin;
 - Sender transmits a *Request to Send (RTS)* frame to the receiver.
 - Includes a field that indicates how long the sender wants to hold the medium
 - Includes length of the data frame to be transmitted
 - Receiver replies with a *Clear to Send (CTS)* frame
 - This frame echoes the length field back to the sender

IEEE 802.11 – Collision Avoidance

- Any node that sees the CTS frame knows that
 - it is close to the receiver, therefore
 - cannot transmit for the period of time it takes to send a frame of the specified length
- Any node that sees the RTS frame but not the CTS frame
 - is not close enough to the receiver to interfere with it, and
 - so is free to transmit to a node other than the node originating the RTS

IEEE 802.11 – Collision Avoidance

- If two or more nodes detect an idle link and try to transmit an RTS frame at the same time
 - Their RTS frame will collide with each other
 - So the senders realize the collision when they do not receive the CTS frame after a period of time
 - In this case, they each wait a random amount of time before trying again.
 - The amount of time a given node delays is defined by the same *exponential backoff* algorithm used on the Ethernet.



IEEE 802.11 – Collision Avoidance

802.11 does not support collision detection! Why?

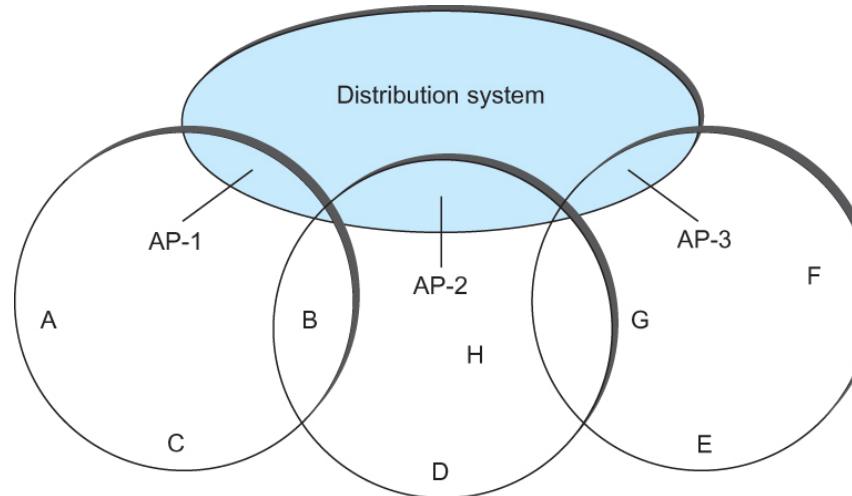
IEEE 802.11 – Distribution System

- WiFi devices are free to move around
- To deal with this mobility and partial connectivity,
 - Some nodes are stable, like an **anchor**, which are connected to a wired network infrastructure
 - they are called *Access Points (AP)* and they are connected to each other by a so-called *distribution system*



IEEE 802.11 – Distribution System

- Three local area networks (LANs) severed by three Aps;
- APs are connected to the distribution system (in most cases Ethernet).

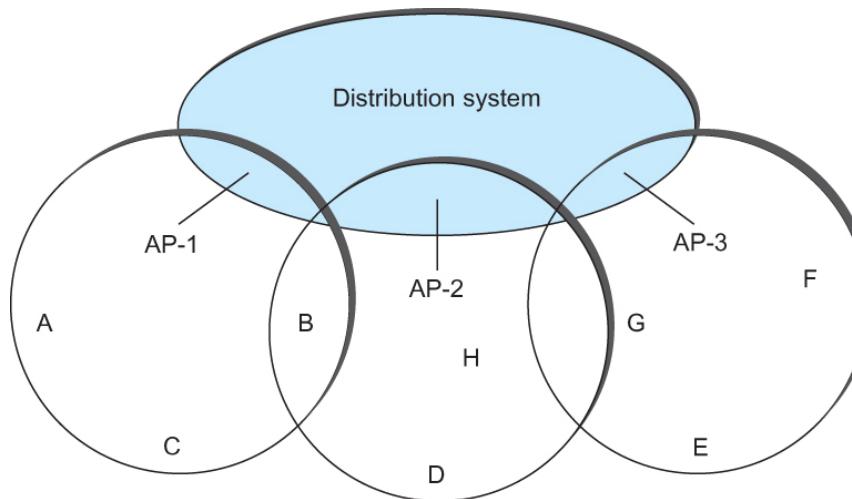


Access points connected to a distribution network

IEEE 802.11 – Distribution System

- For example, if A tries to talk with E,

A first sends a frame to its AP-1 which forwards the frame across the distribution system to AP-3, which finally transmits the frame to E



A -> AP-1 -> AP-3 -> E

IEEE 802.11 – Distribution System

When a node is moving, how to associate with an appropriate AP?
This is called, **handover!**

- *Active scanning*
 - The node initiates a *Probe* frame
 - All APs within reach reply with a *Probe Response* frame
 - The node selects one of the APs, based on signal strength, and sends that AP an *Association Request* frame
 - The AP replies with an *Association Response* frame

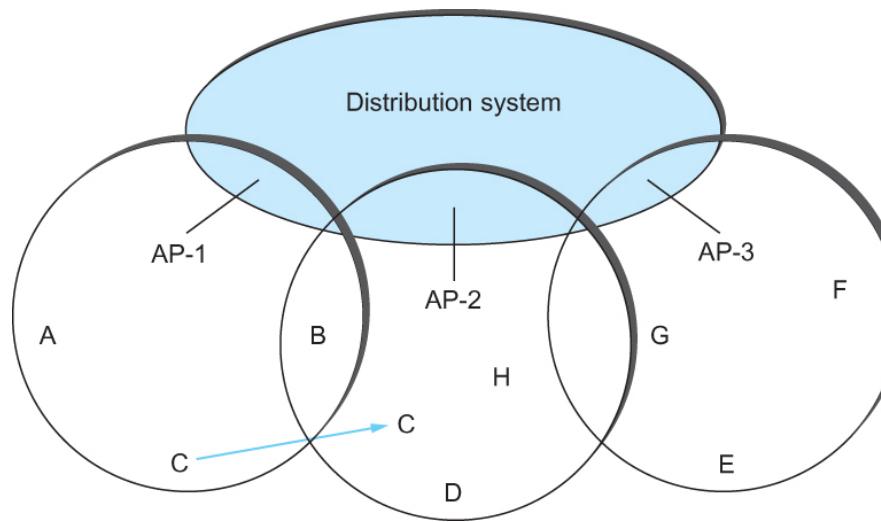


IEEE 802.11 – Distribution System

- Active scanning – node is actively searching for an access point
- Passive scanning
 - performed by access points
 - AP's periodically send Beacon Frames
 - AP's advertise their capabilities in Beacon Frames
 - Nodes can decide to change AP's based on Beacon Frames

IEEE 802.11 – Distribution System

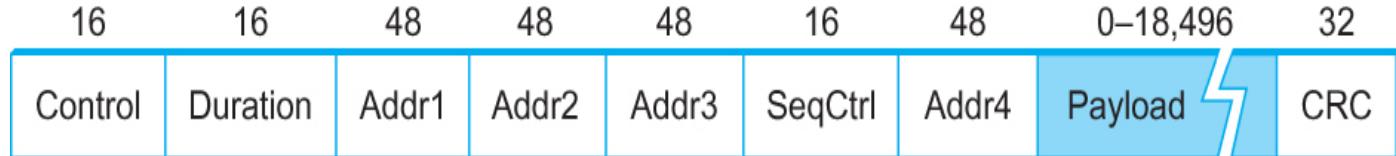
- For example



Node Mobility

IEEE 802.11 – Frame Format

- Source and Destination addresses: each 48 bits
- Data: up to 2312 bytes
- CRC: 32 bit
- Control field: 16 bits
 - Contains three subfields (of interest)
 - 6 bit **Type** field: indicates whether the frame is an RTS or CTS frame or being used by the scanning algorithm
 - A pair of 1 bit fields : called **ToDS** and **FromDS**



Frame Format

Chapter Summary

- Physical layer (L1) and Link layer (L2) techniques.
- We looked into five key issues in L1&L2
 - Encoding
 - Framing
 - Error Detecting
 - Reliability
 - Multiple Access Links
 - Ethernet
 - Wireless 802.11

CPE348: Introduction to Computer Networks

Lecture #8: Chapter 3.1



Jianqing Liu
Assistant Professor of Electrical and Computer
Engineering, University of Alabama in Huntsville
jianqing.liu@uah.edu
<http://jianqingliu.net>

Chapter 3 – Internetworking

- How do we build networks of global scale?
- How do we interconnect different types of networks to build a large global network?

Chapter Outline

- Switching and Bridging (L2)
- Basic Internet Protocol (IP) (L3)
- Routing (L3)

Chapter Goal

- Understanding the functions of switches, bridges and routers
- Discussing Internet Protocol (IP) for interconnecting networks
- Understanding the concept of routing

Switching and Forwarding

- Store-and-Forward Switches
- Bridges and Extended LANs
- Cell Switching
- Segmentation and Reassembly

Switching and Forwarding

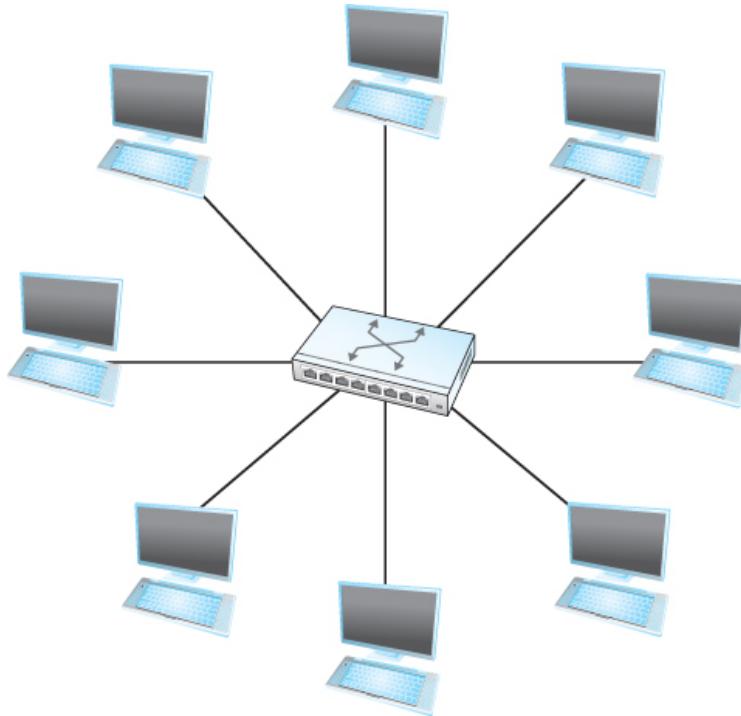
- Switch

- A **mechanism**: that allows us to interconnect links to form a large network
- A **device**: which transfers packets from an input to one or more outputs



Switching and Forwarding

Switches add the star topology to the point-to-point link, bus (Ethernet), and ring (802.5) topologies



Switching and Forwarding

Properties of this star topology

- Switch have a **fixed** number of **inputs and outputs** – limits the number of hosts supported by a switch
- Large networks have **interconnected switches**
- Use **point-to-point** links between switches and hosts
- Additional hosts do not impact performance of current hosts

Switching and Forwarding

- A switch is connected to a set of links
- For each link, the switch runs the link protocol to communicate with that node
- Switches receive incoming packets on one of its links and transmit them on a different link(s)
 - This function is referred as ***switching and forwarding***

Switch is a L2 device!

Switching and Forwarding

How does the switch decide which output port to place each packet on?

- Two common approaches
 - *Datagram* or *Connectionless* approach
 - *Virtual circuit* or *Connection-oriented* approach
- A third approach *source routing* is less common
- Specified at the header of the packet

Switching and Forwarding

Let's first clear some terminologies!

connectionless

connection-oriented

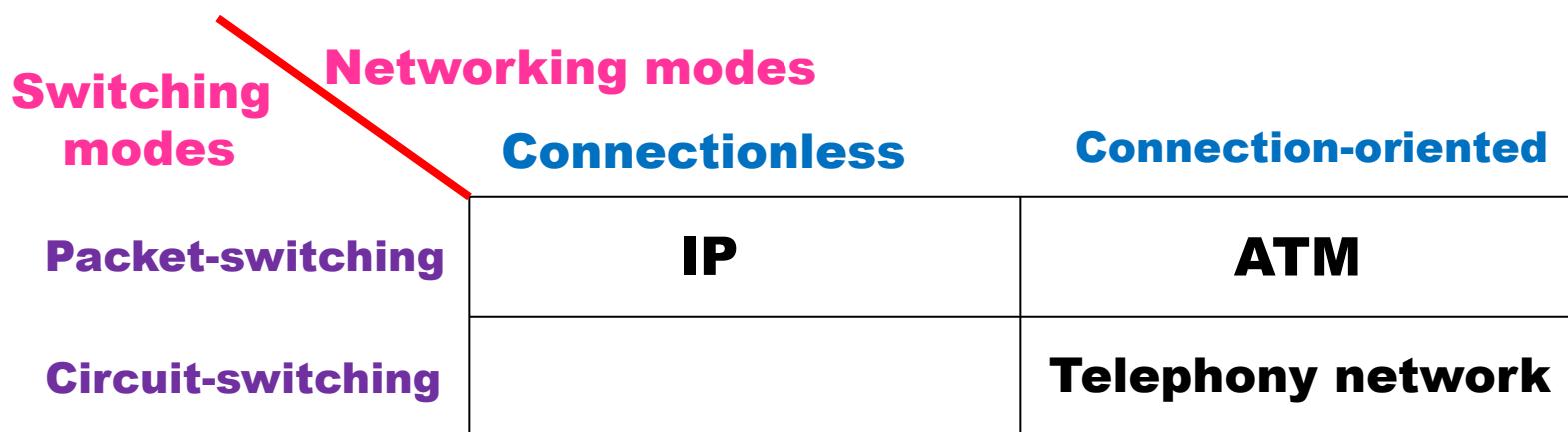
packet-switching

circuit-switching

Switching and Forwarding

Overview

- Applications at endpoints send data without warning in CL networks;
- CO networks need a connection setup phase;



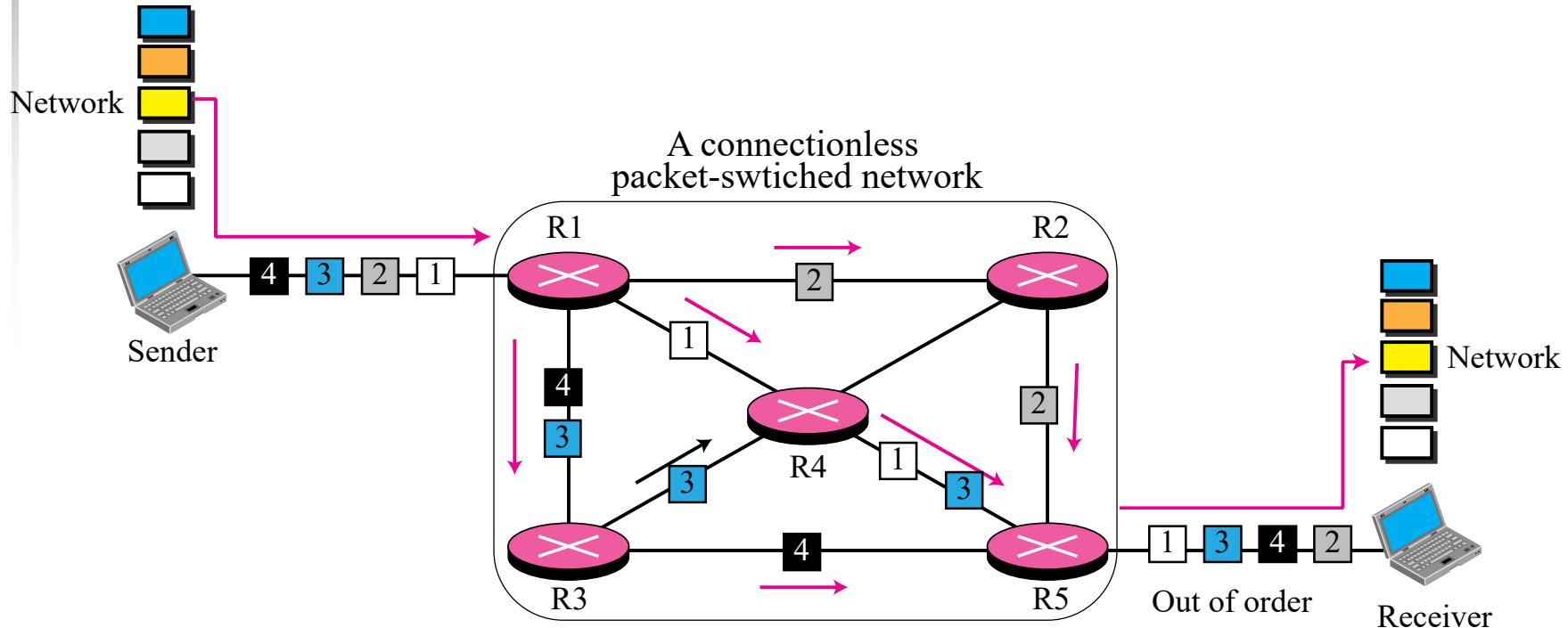
Switching and Forwarding

Overview

- In circuit switching, the whole message is sent from the source to the destination without being divided into packets.
- In packet switching, the message is first divided into manageable packets at the source before being transmitted. The packets are assembled at the destination.

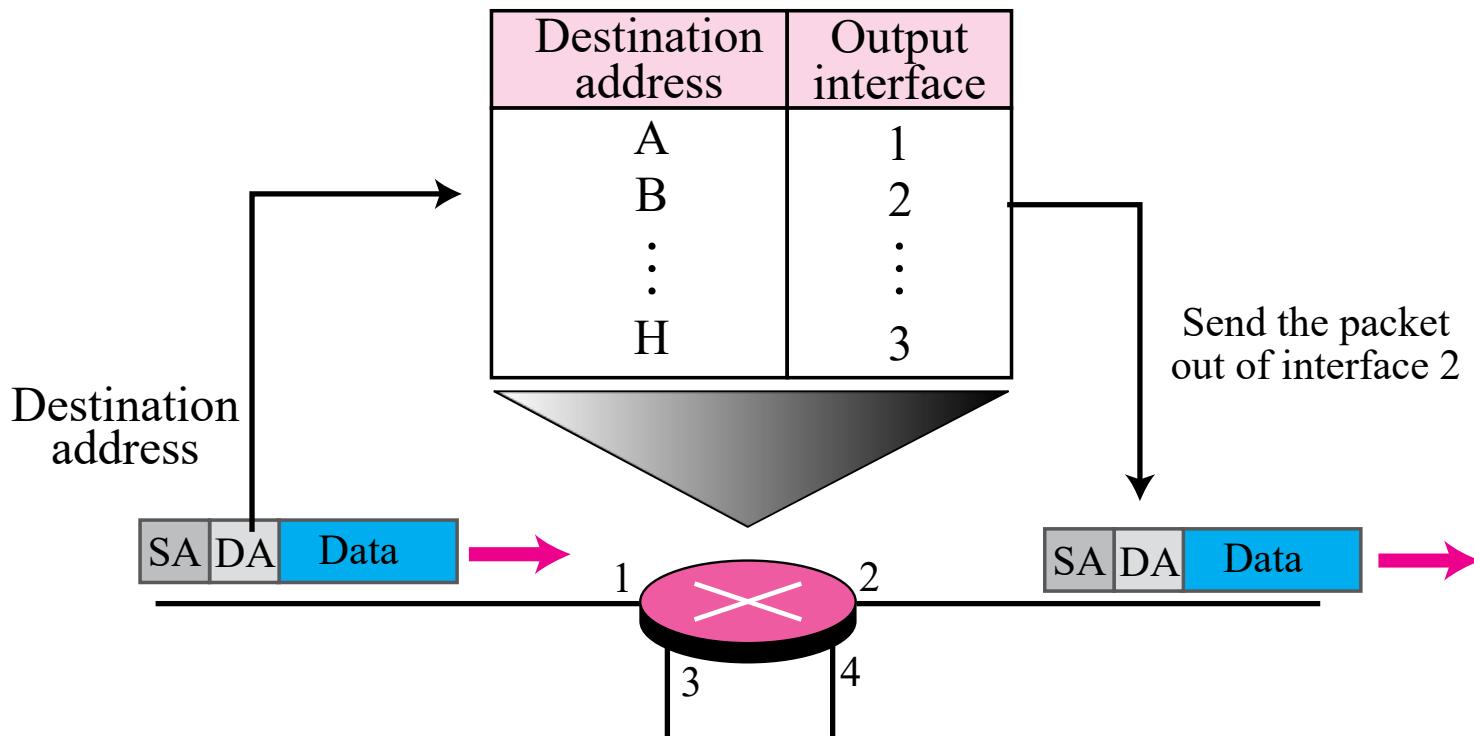
Switching and Forwarding

Example: A connectionless packet-switched network



Switching and Forwarding

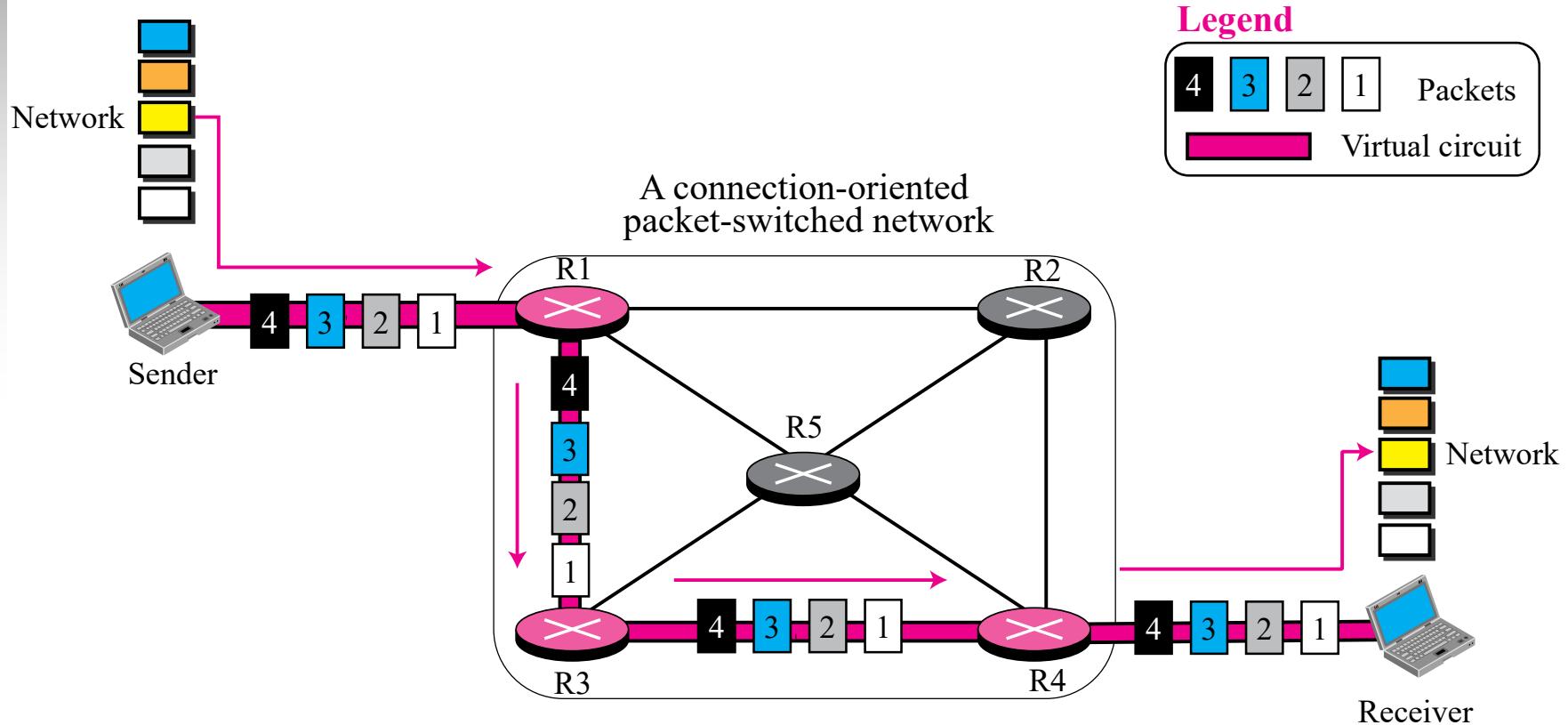
Example: A connectionless packet-switched network



Forwarding decision is based on destination address.

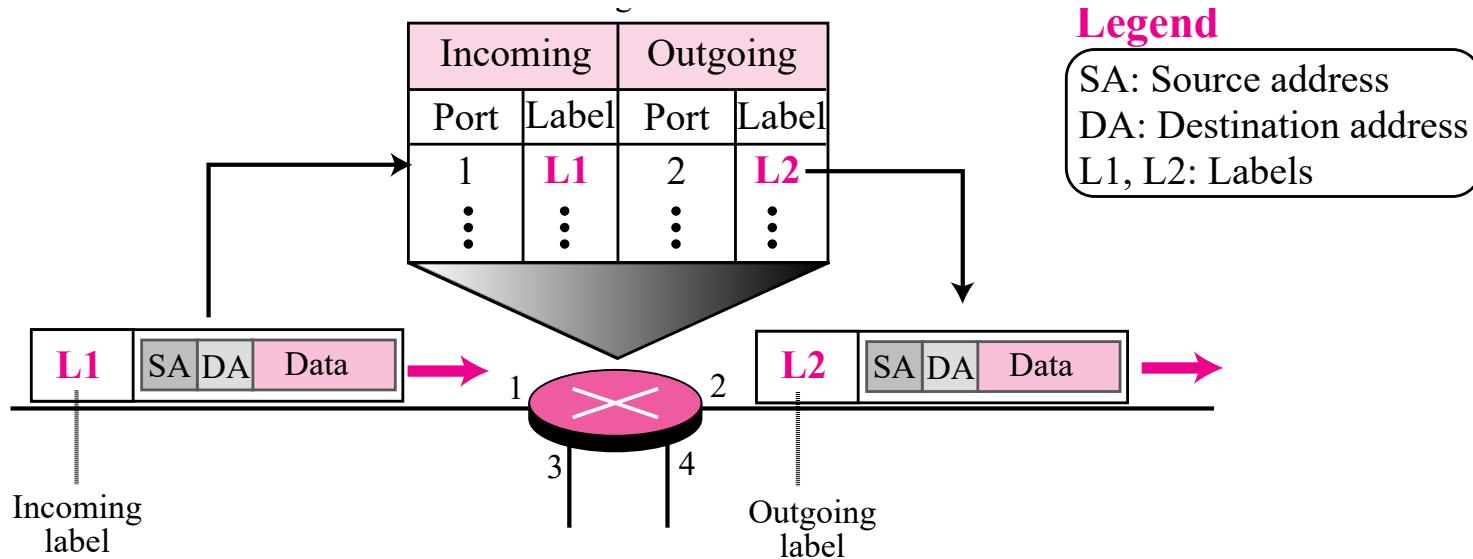
Switching and Forwarding

Example: A connection-oriented packet-switched network



Switching and Forwarding

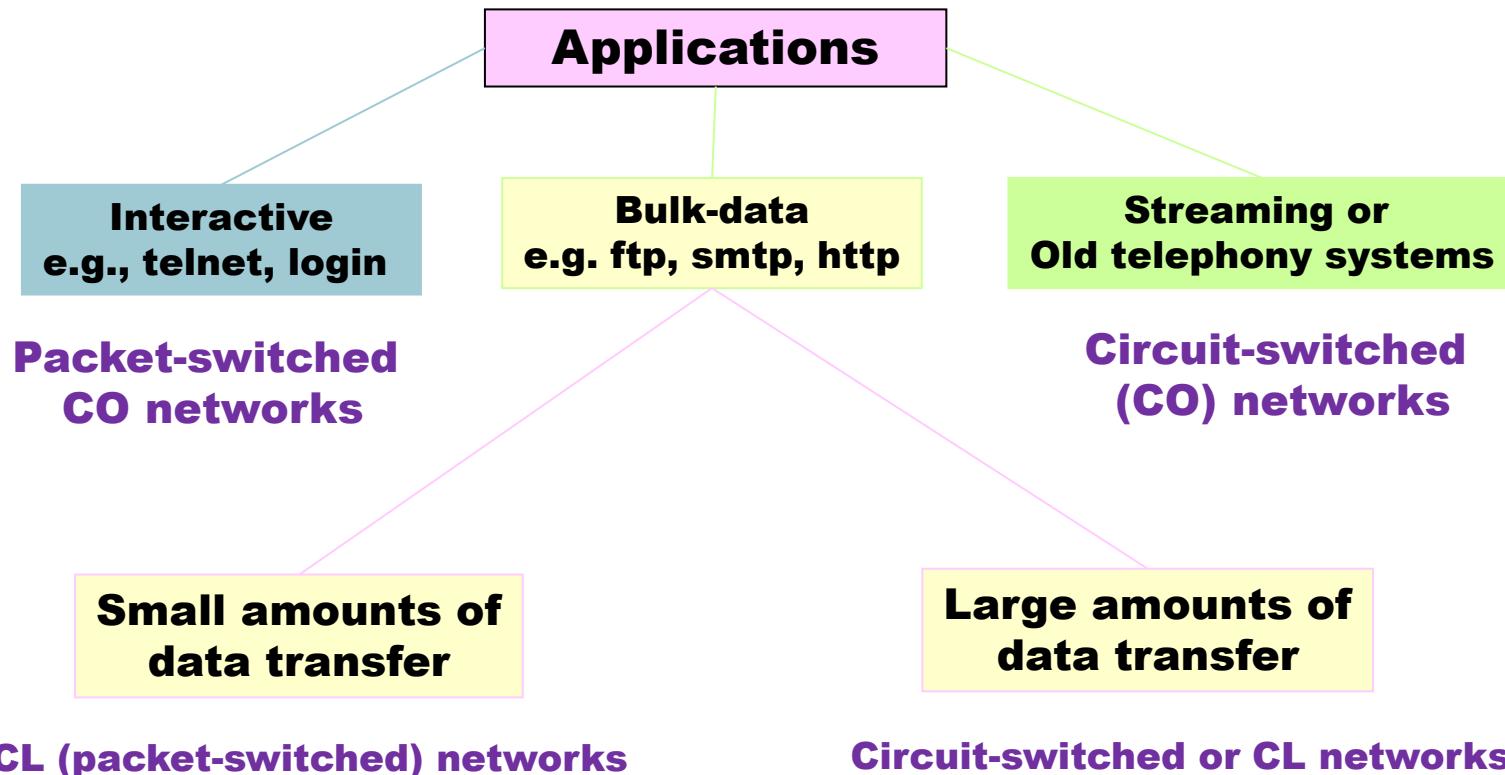
Example: A connection-oriented packet-switched network



Forwarding decision is based on the label.

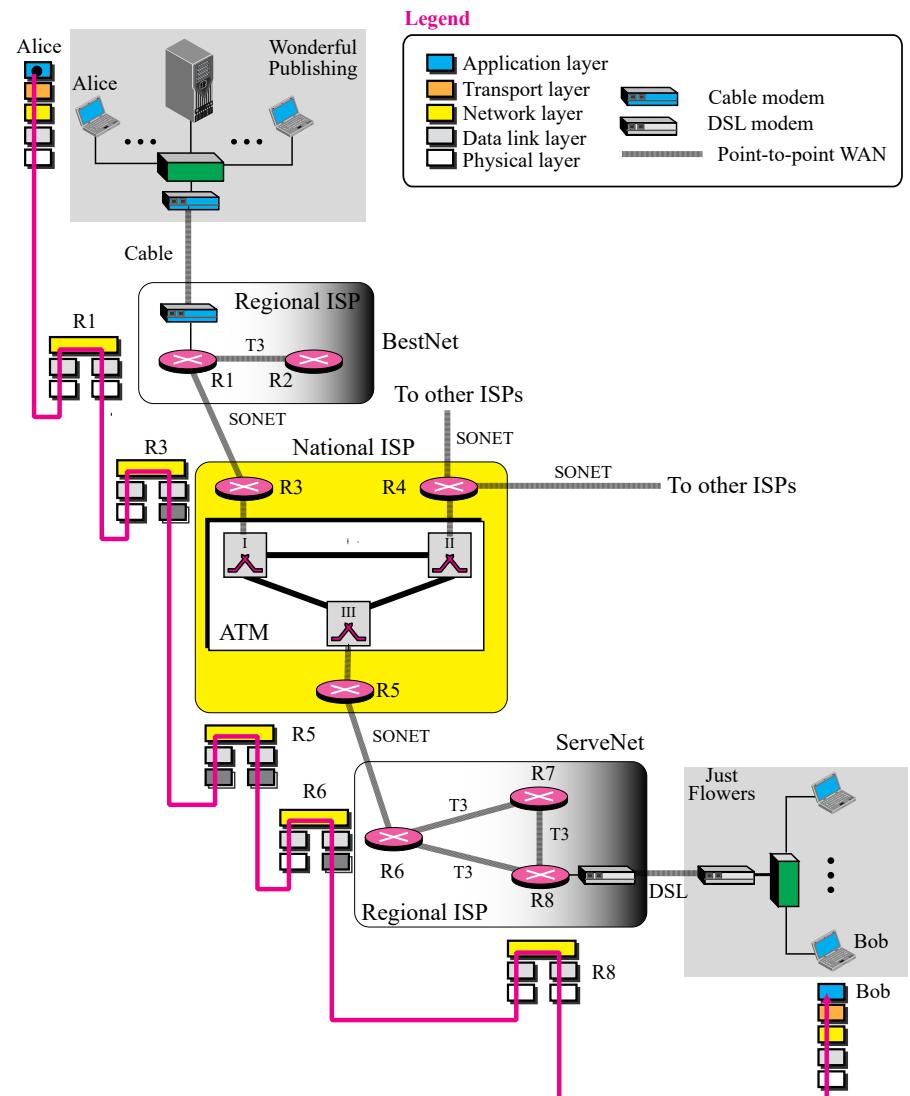
Switching and Forwarding

The operation mode is determined by specific applications!



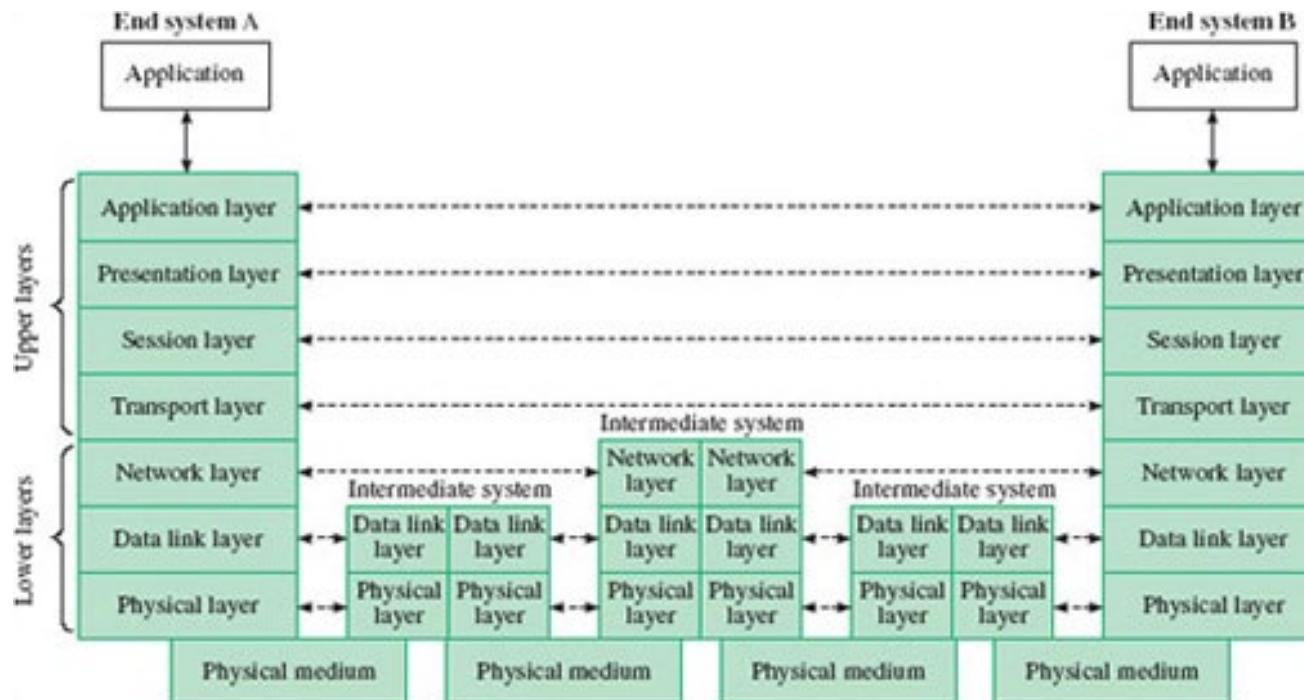
Switching and Forwarding

An overview of the Internet



Switching and Forwarding

An overview of the Internet using the OSI model



CPE348: Introduction to Computer Networks

Lecture #9: Chapter 3.2



Jianqing Liu
Assistant Professor of Electrical and Computer
Engineering, University of Alabama in Huntsville
jianqing.liu@uah.edu
<http://jianqingliu.net>

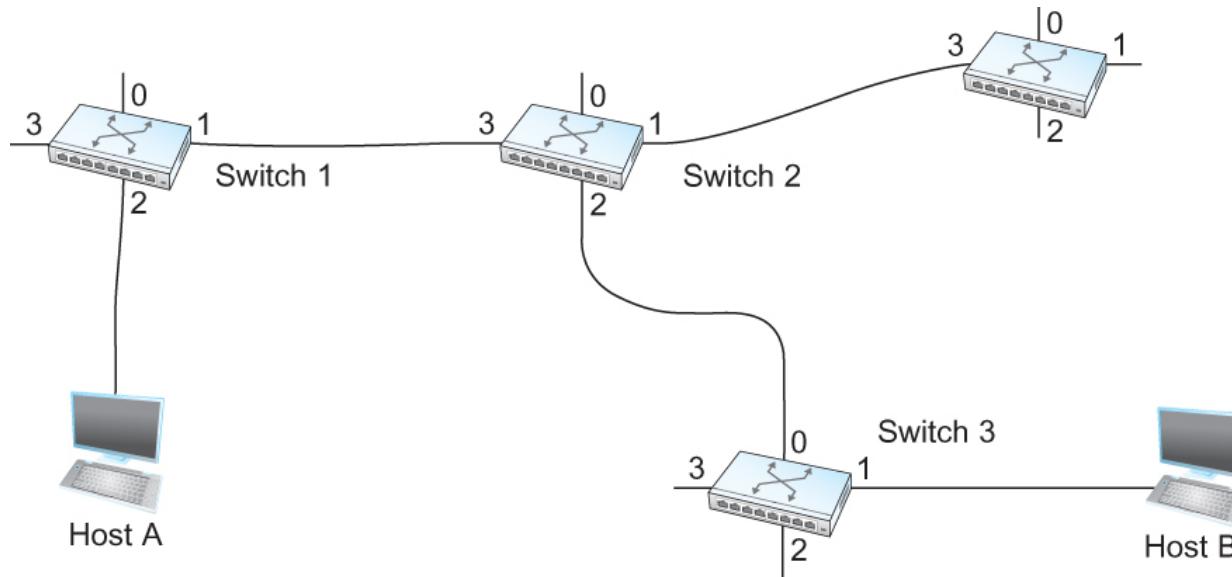
Virtual Circuit Switching

Virtual Circuit Switching

- Similar to circuit switching
- Widely used technique for packet switching
- Uses the concept of *virtual circuit (VC)*
- Provides **connection-oriented** service
- First set up a virtual connection from the source host to the destination host and then send the data

Virtual Circuit Switching

- Host A wants to send packets to host B



Virtual Circuit Switching

Two-stage process

- Connection setup
- Data Transfer
- Connection setup
 - Establish “**connection state**” or “**label**” in each of the switches between the source and destination hosts
 - The connection state for a single connection consists of an entry in the “**VC table**” in each switch

Virtual Circuit Switching

One entry in the VC table on a single switch contains

- A virtual circuit identifier (VCI)
 - Uniquely identifies the connection at this switch for the link
 - Carried inside the header of the packets that belong to this connection
- An incoming interface on which packets for this VC arrive at the switch
- An outgoing interface in which packets for this VC leave the switch

VC table in a Switch

Incoming Interface	Incoming VCI	Outgoing Interface	Outgoing VCI
3	11	2	7

Virtual Circuit Switching

Note:

- There may be **many virtual connections** established in the switch at one time
- Incoming and outgoing **VCI** values are **not** generally the **same**
- Whenever a new connection is created, we need to assign a **new VCI** for that connection on each link that the connection will traverse
 - We need to ensure that the chosen VCI on a given link is not currently in use on that link by some existing connection.

Virtual Circuit Switching

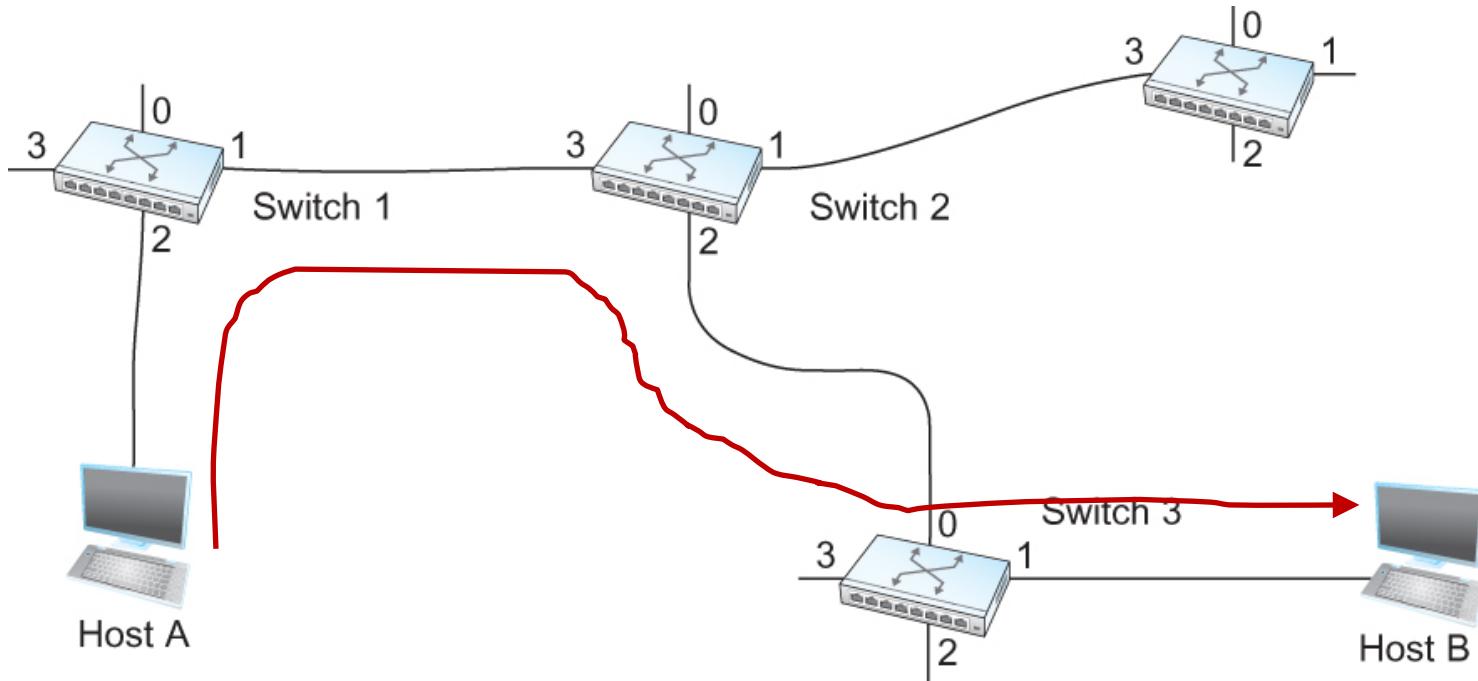
Two broad classes of approach to establishing a virtual circuit

- Network Administrator will configure the state (**centralized**)
- A host can send messages into the network to cause the state to be established (**distributed**)

Virtual Circuit Switching

Let's assume that a network administrator wants to manually create a new virtual connection from host A to host B

- First the administrator identifies a path through the network from A to B



[return](#)

Virtual Circuit Switching – centralized

The administrator then picks a VCI value that is currently *unused* on each link for the connection

- For our example,
 - Suppose the VCI value 5 is chosen for the link from host A to switch 1
 - 11 is chosen for the link from switch 1 to switch 2
 - So the switch 1 will have an entry in the VC table

Switch 1 VC table

Incoming Interface	Incoming VCI	Outgoing Interface	Outgoing VCI
2	5	1	11

Virtual Circuit Switching – centralized

Similarly, suppose

- VCI of 7 is chosen to identify this connection on the link from switch 2 to switch 3
- VCI of 4 is chosen for the link from switch 3 to host B
- Switches 2 and 3 are configured with the following VC table

Switch 2 VC table

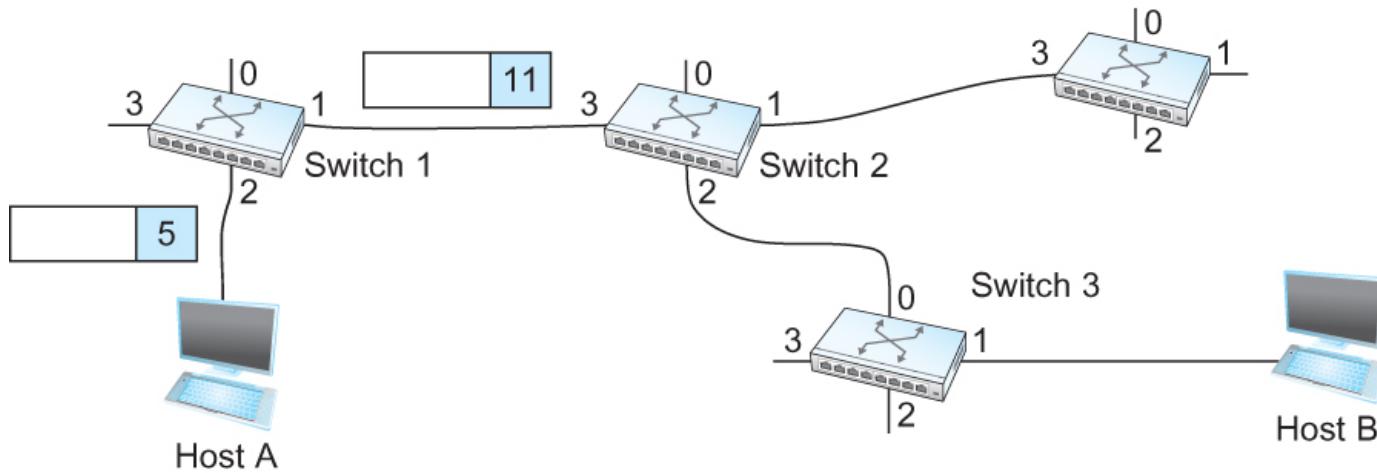
Incoming Interface	Incoming VCI	Outgoing Interface	Outgoing VCI
3	11	2	7

Switch 3 VC table

Incoming Interface	Incoming VCI	Outgoing Interface	Outgoing VCI
0	7	1	4

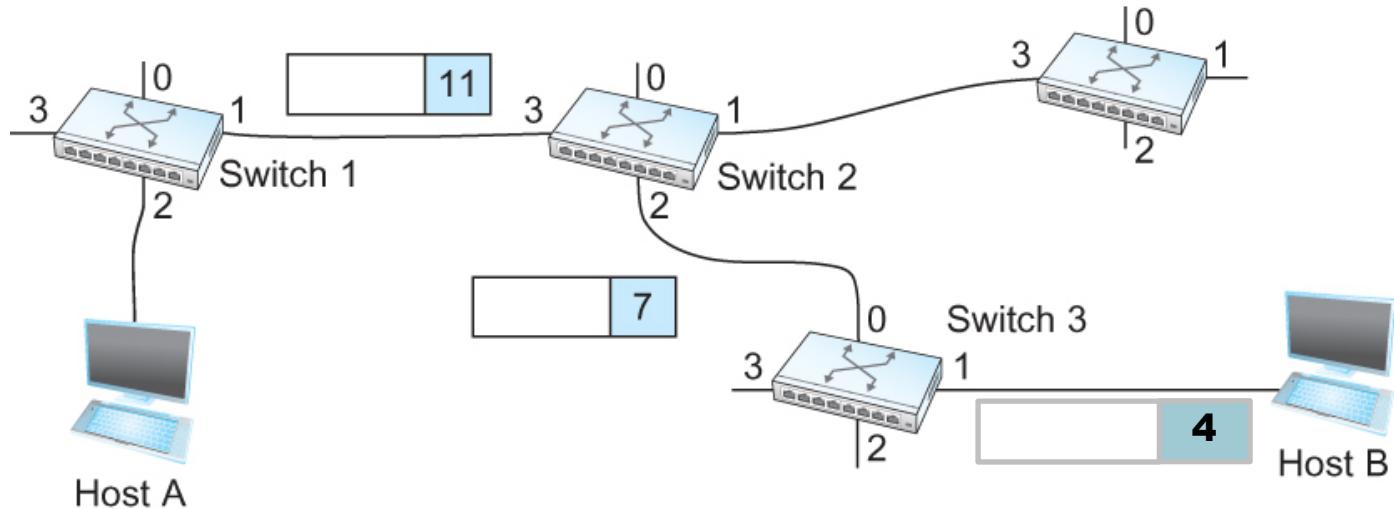
Virtual Circuit Switching – centralized

- For any packet that A wants to send to B, A puts the VCI=5 in the header of the packet and sends it to switch 1.
- Switch 1 receives a packet of VCI=5 on interface 2, and it searches for the appropriate VC table entry.
- The table entry on switch 1 tells the switch the output interface=1 and VCI=11.



Virtual Circuit Switching – centralized

- Packet will arrive at switch 2 on interface 3 bearing VCI=11
- Switch 2 looks up interface 3 and VCI=11 in its VC table and sends the packet on to switch 3 after updating the VCI=7
- This process continues until it arrives at host B with the VCI=4 in the packet

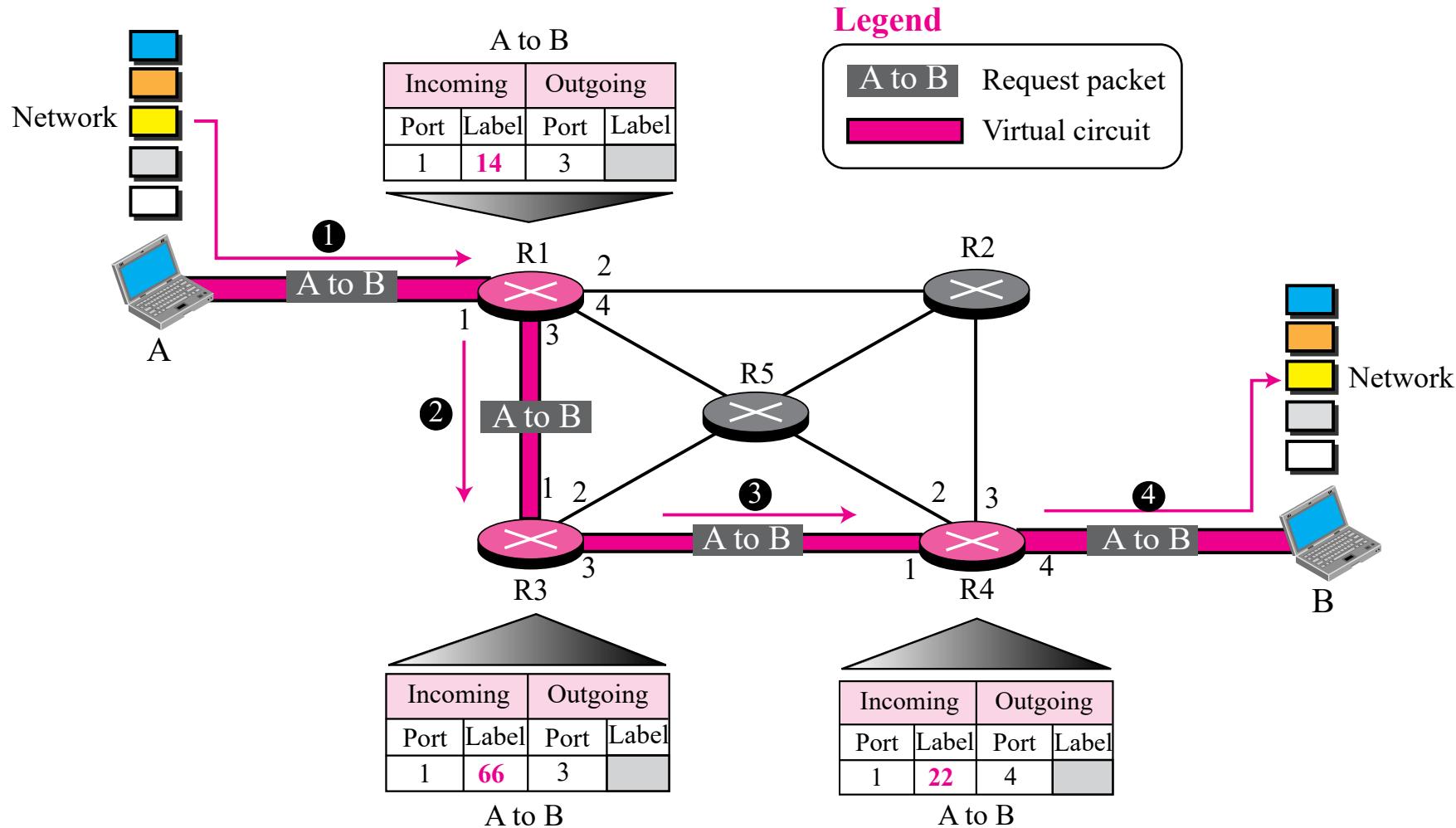


Virtual Circuit Switching – centralized

- In real networks of a large number of switches, the burden of configuring VC tables become excessive!!!
- A user-centric, on-demand approach to configuring VC tables is favorable!

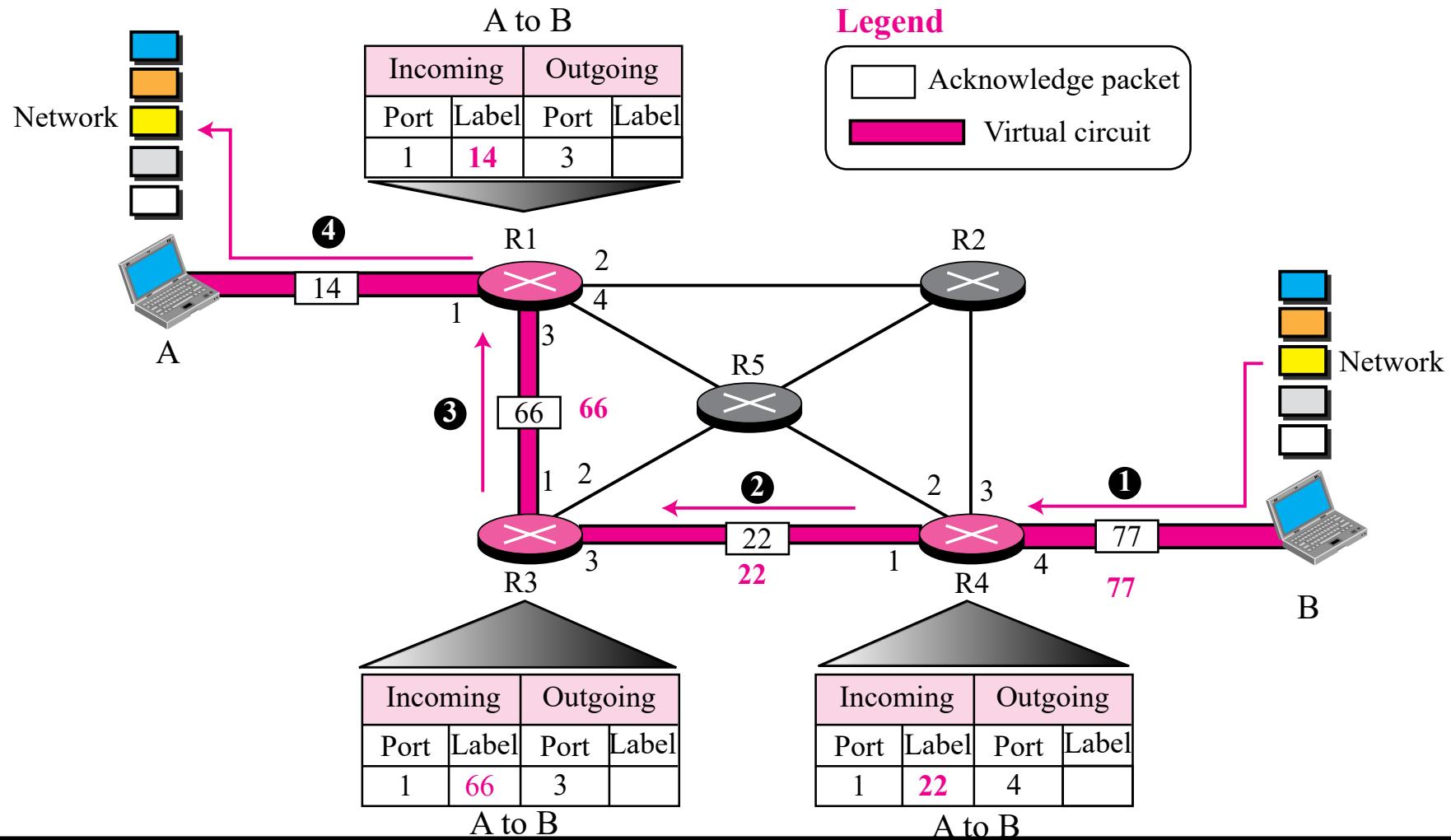
Virtual Circuit Switching – distributed

Overview: Part1 – Sending a *request packet*



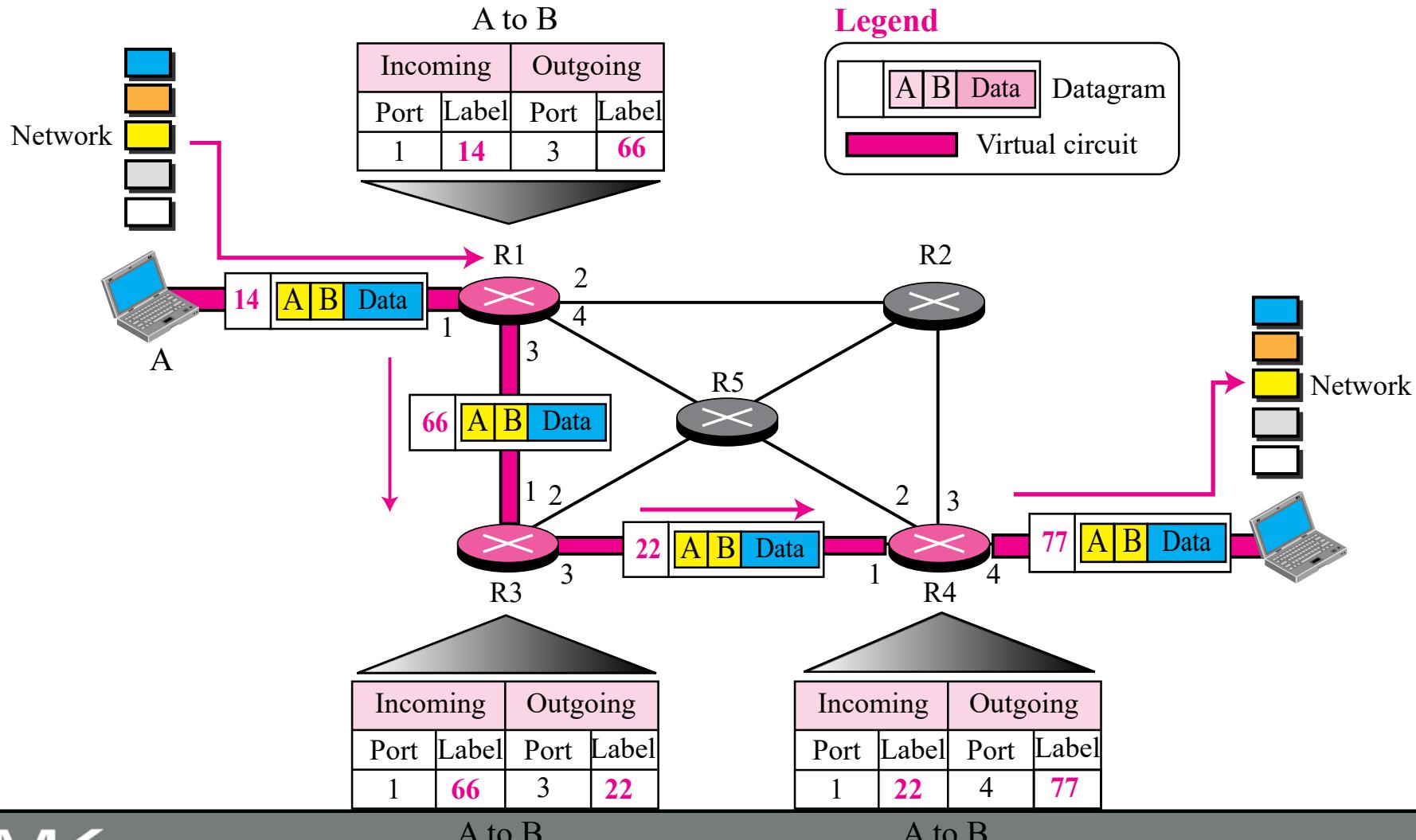
Virtual Circuit Switching – distributed

Overview: Part2 – Setup an *acknowledgement*



Virtual Circuit Switching – distributed

Overview: After Part 1 and Part 2, data communications



Virtual Circuit Switching – distributed

- When host A no longer wants to send data to host B, it tears down the connection by sending a teardown message to switch 1
- The switch 1 removes the relevant entry from its table and forwards the message to the other switches in the path which similarly delete the appropriate table entries

Virtual Circuit Switching – distributed

- Characteristics of setting up a Virtual Circuit
 - Connection request requires at least **one RTT of delay**, so there is a delay before data is sent.
 - If a switch or a link in a connection fails, the connection is broken and a new one will need to be established.
 - The issue of how a switch decides which link to forward the connection request on has similarities with the function of a routing algorithm. (We will discuss it in the next chapter.)

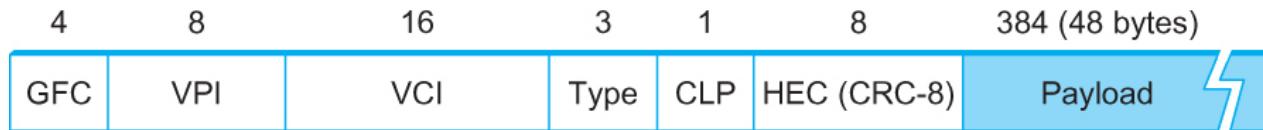
Asynchronous Transfer Mode

- ATM (Asynchronous Transfer Mode)
 - Based on virtual circuit switching
 - Connection-oriented packet-switched network
 - Packets are called cells (53 bytes total)
 - 5 byte header + 48 byte payload



Asynchronous Transfer Mode

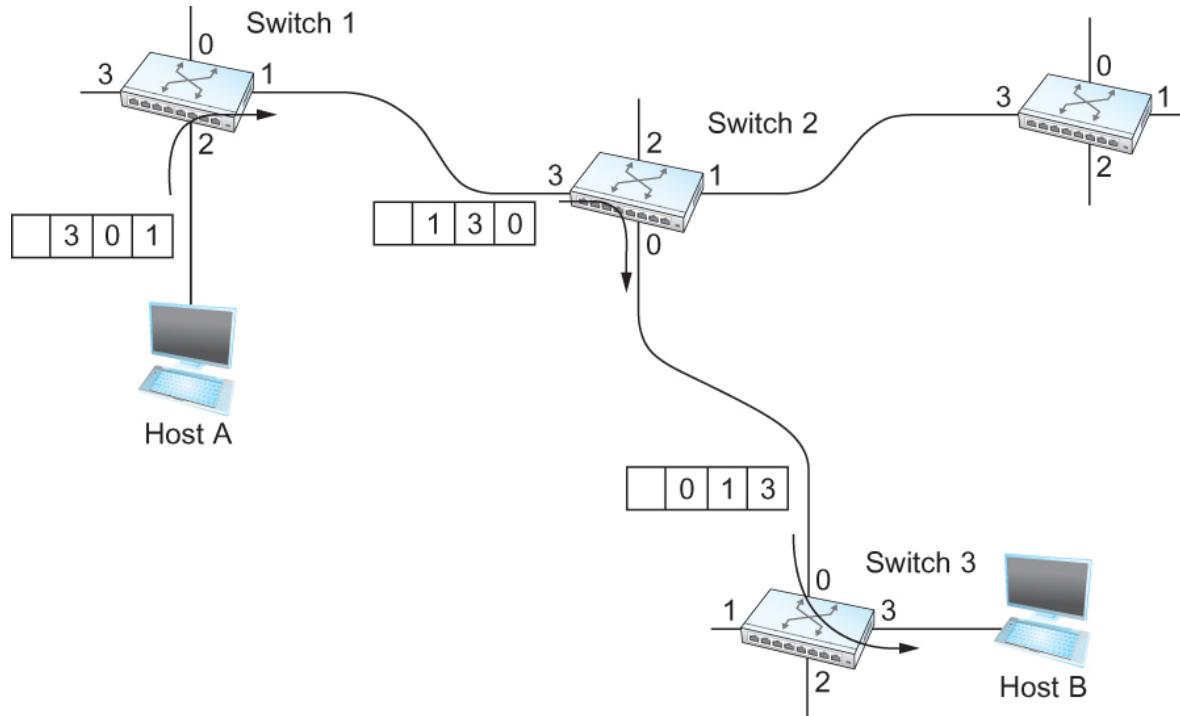
- ATM Cell Structure
 - User-Network Interface (UNI) (Host-to-Switch format)
 - GFC: Generic Flow Control
 - VPI: Virtual Path Identifier
 - VCI: Virtual Circuit Identifier
 - Type: management, congestion control
 - CLP: Cell Loss Priority
 - HEC: Header Error Check (CRC-8)



- Network-Network Interface (NNI) (Switch-to-Switch format)
 - GFC becomes part of VPI field

Source Routing

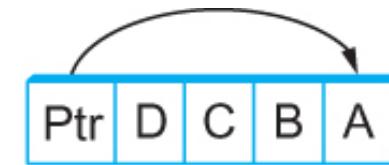
- Source Routing
 - All the information about network topology that is required to switch a packet across the network is provided by the source host
 - Header contains next port to use by receiving switch



Source Routing

- Three other approaches in Source Routing

Header entering switch



Header leaving switch



(a)



(b)



(c)

(a) – header is rotated

(b) – header is stripped

(c) – pointer is used

CPE348: Introduction to Computer Networks

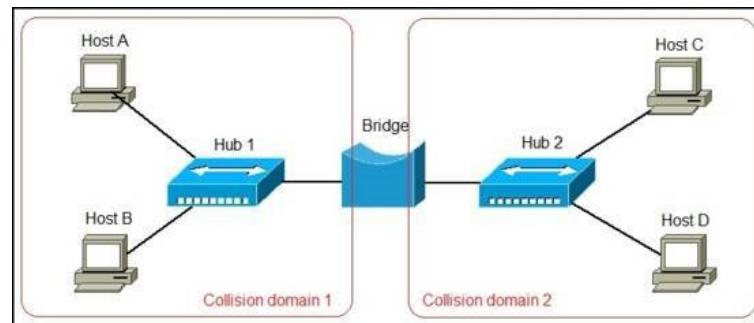
Lecture #10: Chapter 3.3



Jianqing Liu
Assistant Professor of Electrical and Computer
Engineering, University of Alabama in Huntsville
jianqing.liu@uah.edu
<http://jianqingliu.net>

Bridges and LAN Switches

- Bridges and LAN Switches
 - Class of switches that is used to forward packets between shared-media LANs
 - Connect a pair of Ethernets
 - **One approach is put a repeater in between them**
 - It might exceed the physical limitation of the Ethernet
 - No more than a total of 2500 m in length is allowed
 - **Alternatively, put a node between the two Ethernets**
 - This node is called a **Bridge**
 - A collection of LANs connected by one or more bridges is usually said to form an **Extended LAN** (split domains)

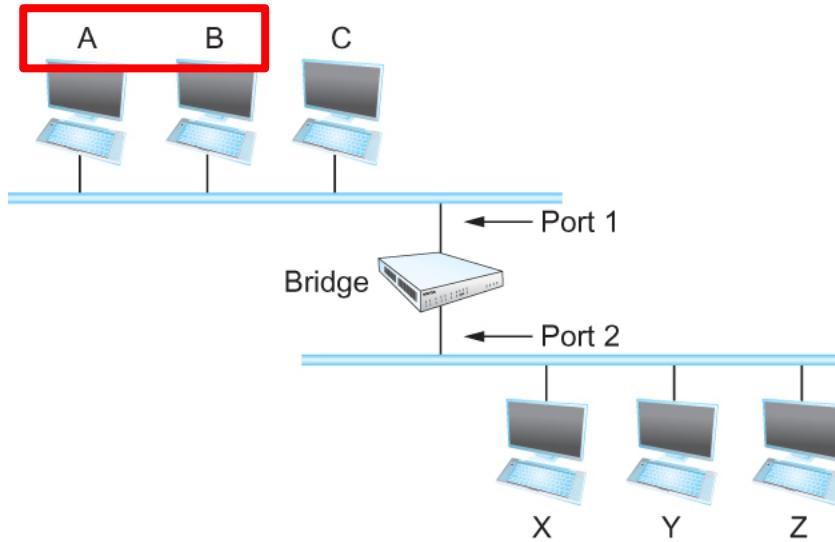


Bridges and LAN Switches

- Simplest Strategy for Bridges
 - Accept LAN frames on their inputs and forward them out to **all** other outputs
 - Used by early bridges
- Learning Bridges
 - Observe that there is no need to forward all the frames that a bridge receives

Bridges and LAN Switches

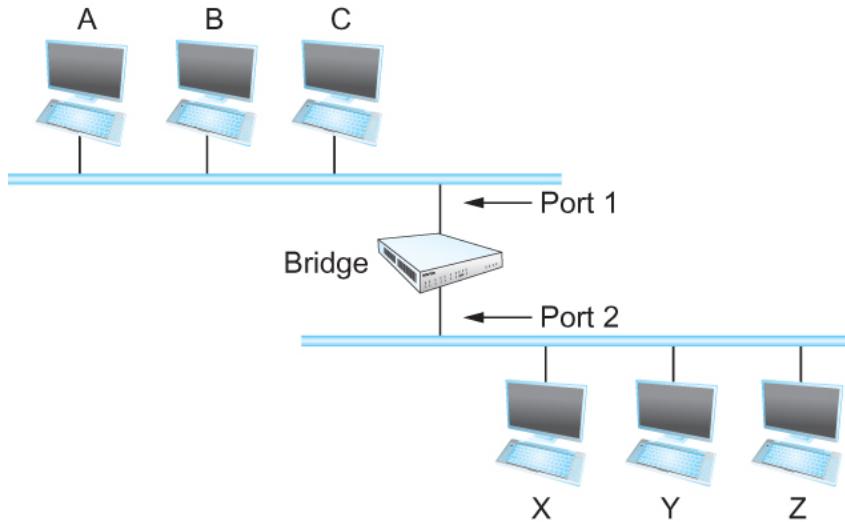
- Consider the following figure
 - When a frame from host A that is addressed to host B arrives on port 1, there is no need for the bridge to forward the frame out over port 2.



- How does a bridge come to learn on which port the various hosts reside?

Bridges and LAN Switches

- Solution
 - Download a table into the bridge

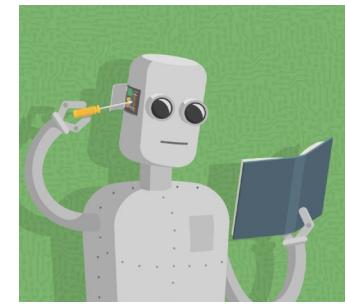


- Who does the download?
 - Human - Too much work for maintenance

Bridge Routing Table	
Host	Port
A	1
B	1
C	1
X	2
Y	2
Z	2

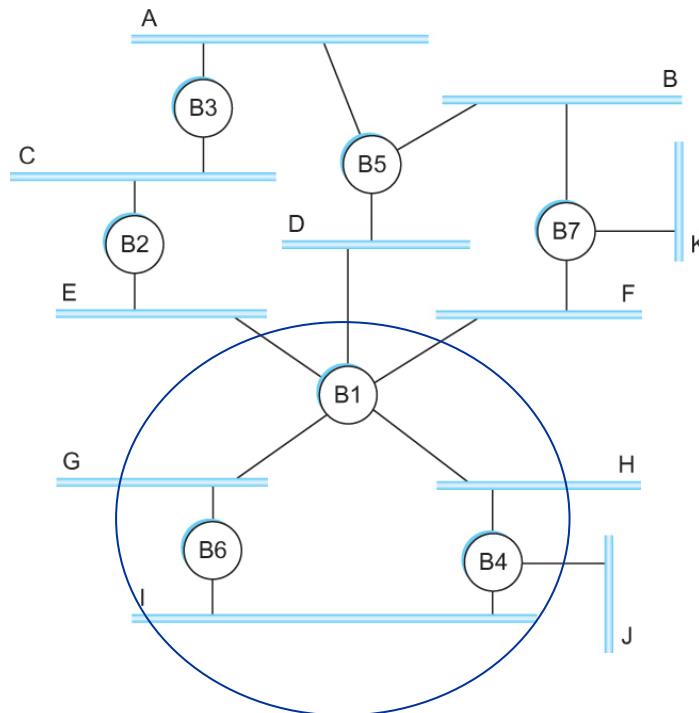
Bridges and LAN Switches

- Can the bridge learn this information by itself?
 - Yes – Learning Bridge
- How?
 - Each bridge inspects the source address in all the frames it receives
 - Record the information at the bridge and build the table
 - When a bridge first boots, this table is empty
 - Entries are added over time
 - A timeout is associated with each entry and the bridge discards the entry after a specified period of time
 - Due to node mobility
- If the bridge receives a frame that is addressed to host not currently in the table
 - Forward the frame out on all other ports



Bridges and LAN Switches

- Strategy works fine if the extended LAN does not have a loop in it
- Why? Frames potentially loop through the extended LAN forever



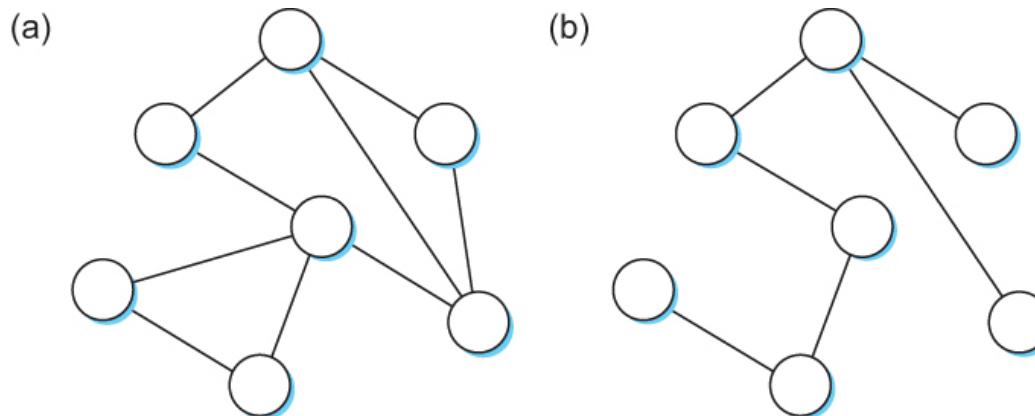
- Bridges B1, B4, and B6 form a loop

Bridges and LAN Switches

- How does an extended LAN come to have a loop in it?
 - Loops are built into the network to provide redundancy in case of failures
- Solution
 - Distributed Spanning Tree Algorithm

Spanning Tree Algorithm

- Think of the extended LAN as being represented by a graph that possibly has loops (cycles)
- A spanning tree is a sub-graph of this graph that covers all the vertices but contains no cycles
 - throw out some of the edges



Example of (a) a cyclic graph; (b) a corresponding spanning tree.

Spanning Tree Algorithm

- IEEE 802.1 specification for LAN bridges is based on this algorithm
- Each bridge decides the ports over which it is and is not willing to forward frames
 - In a sense, it is by removing ports from the topology

Spanning Tree Algorithm

- Algorithm is dynamic
 - The bridges are always prepared to reconfigure themselves into a new spanning tree if some bridges fail
- Main idea
 - Each bridge selects the ports over which they will forward the frames

Spanning Tree Algorithm

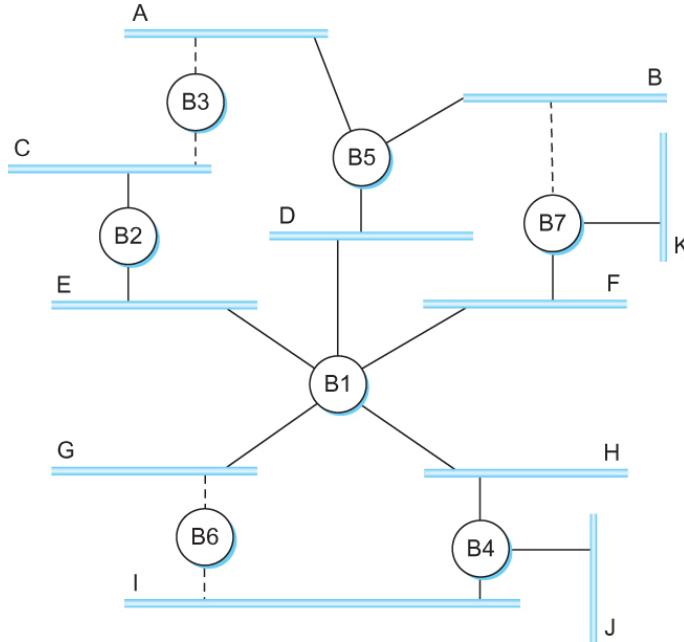
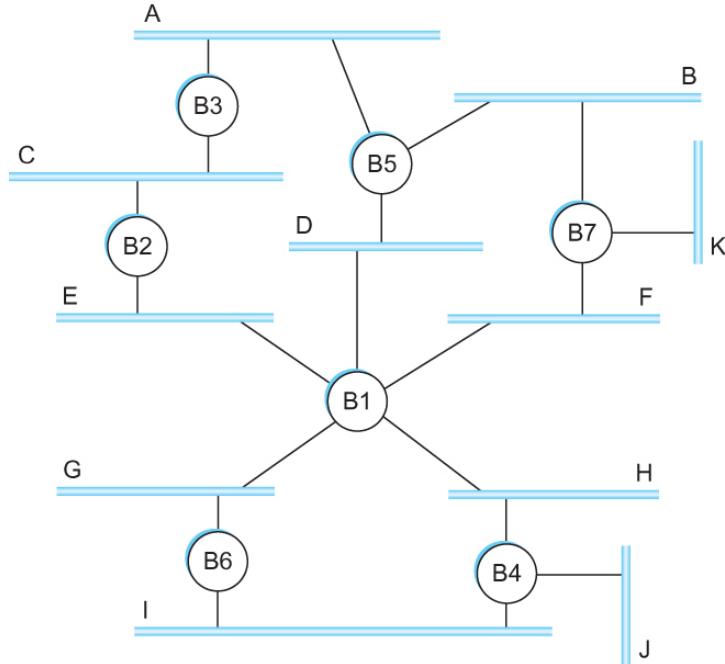
- Spanning Tree Algorithm selects ports as follows:
 - Each bridge has a unique **identifier** (B1, B2, B3,...).
 - Elect the bridge with the smallest id as the **root** of the spanning tree
 - The **root** bridge always forwards frames out over all of its ports
 - Each bridge computes the **shortest path** to the root and notes which of its ports is on this path
 - Finally, all the bridges connected to a given LAN elect a **single designated bridge** that will be responsible for forwarding frames toward the root bridge

Spanning Tree Algorithm

- Each LAN's designated bridge is the one that is closest to the root
- If two or more bridges are equally close to the root,
 - Then select bridge with the smallest id

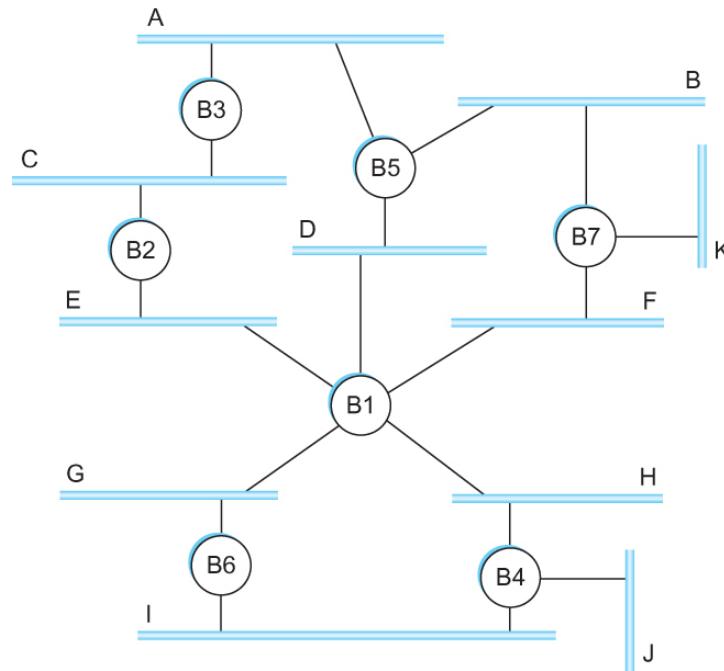
Spanning Tree Algorithm

- B1 is the root bridge
- B3 and B5 are connected to LAN A, and B5 is the designated bridge
- B5 and B7 are connected to LAN B, and B5 is the designated bridge
- B3 and B2 are connected to LAN C, and B2 is the designated bridge



Spanning Tree Algorithm

- ## ■ Consider the network re-boot:



- All bridges would start off by claiming to be the root

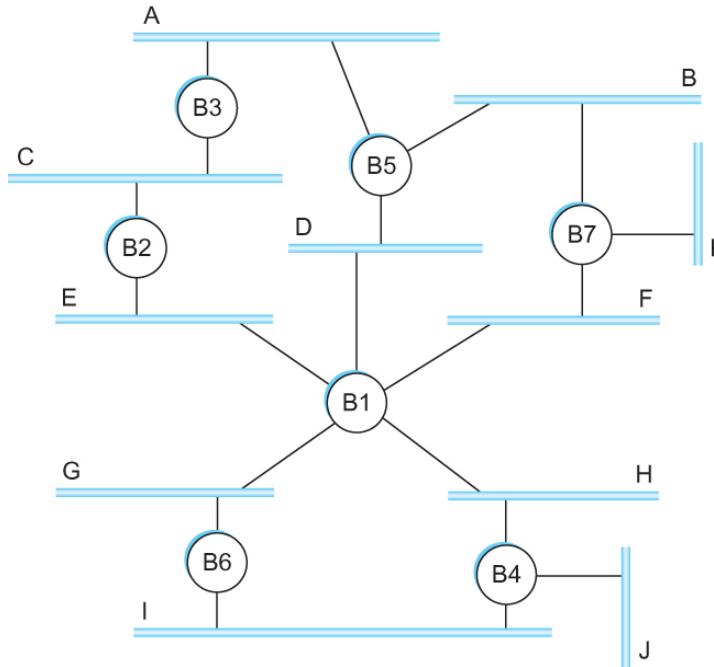
Spanning Tree Algorithm

- Initially each bridge thinks it is the root, so it sends a **configuration message**
- Upon receiving a configuration message, the bridge checks to see if the new message is *better* than the current best configuration message
- The new configuration is better than the currently recorded information if
 - It identifies a root with a smaller id or
 - It identifies a root with an equal id but with a shorter distance or
 - The root id and distance are equal, but the sending bridge has a smaller id

Spanning Tree Algorithm

Example:

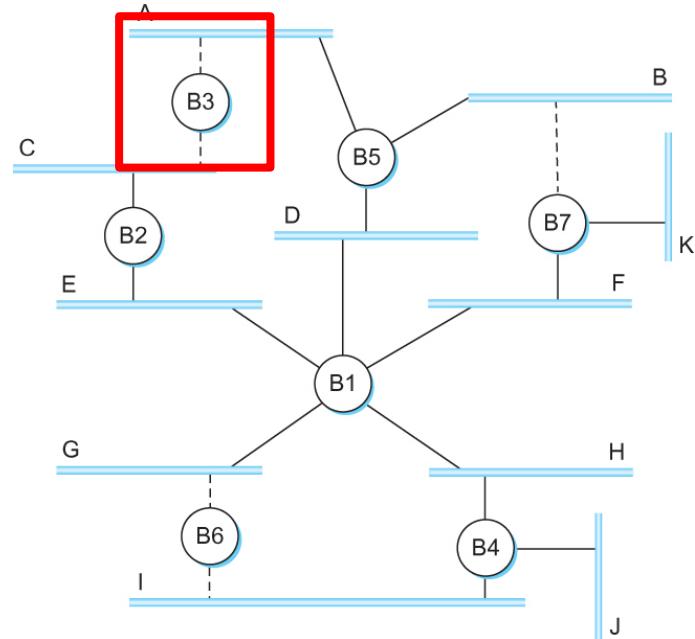
- Denote a configuration message from node X in which it claims to be distance d from the root node Y as (Y, d, X)



- Consider the activity at node B3

Spanning Tree Algorithm

- B3 receives (B2, 0, B2)
- Since $2 < 3$, B3 accepts B2 as root
- B3 adds 1 to the distance advertised by B2 and sends (B2, 1, B3) to B5
- Meanwhile B2 accepts B1 as root because it has the lower id and it sends (B1, 1, B2) toward B3
- B5 accepts B1 as root and sends (B1, 1, B5) to B3
- B3 accepts B1 as root and it notes that both B2 and B5 are closer to the root than it is.
 - Thus B3 stops forwarding messages on both its interfaces
 - This leaves B3 with both ports not selected



Spanning Tree Algorithm

- Even after the system has stabilized, the root bridge continues to send configuration messages periodically
 - Other bridges continue to forward these messages
- When a bridge fails, the downstream bridges will not receive the configuration messages
- After waiting a specified period of time, they will once again claim to be the root and the algorithm starts again

CPE348: Introduction to Computer Networks

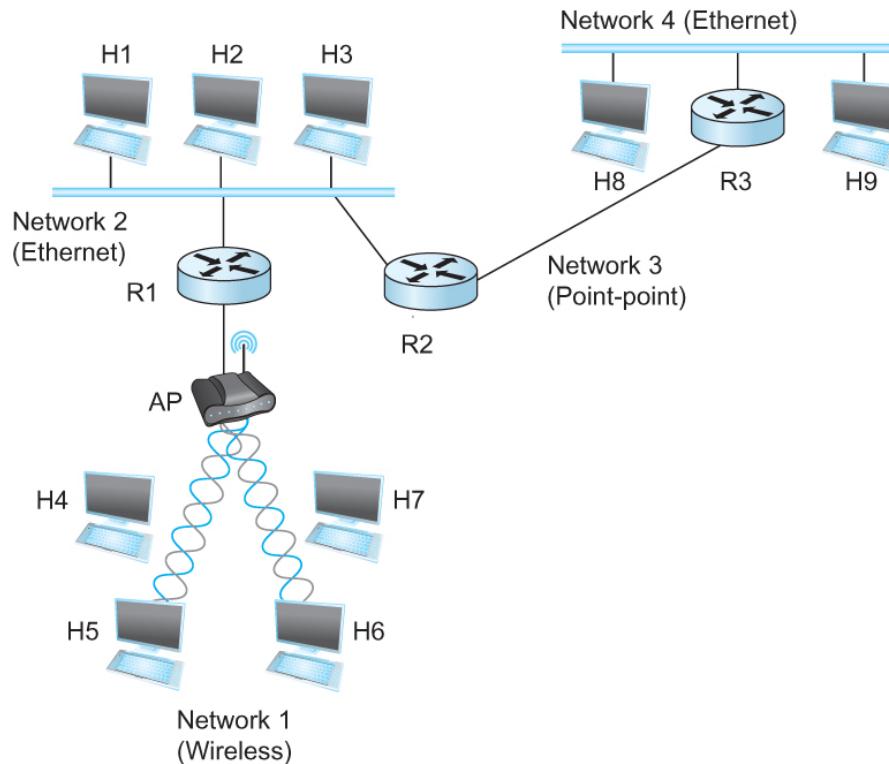
Lecture #11: Chapter 3.4



Jianqing Liu
Assistant Professor of Electrical and Computer
Engineering, University of Alabama in Huntsville
jianqing.liu@uah.edu
<http://jianqingliu.net>

Internetworking

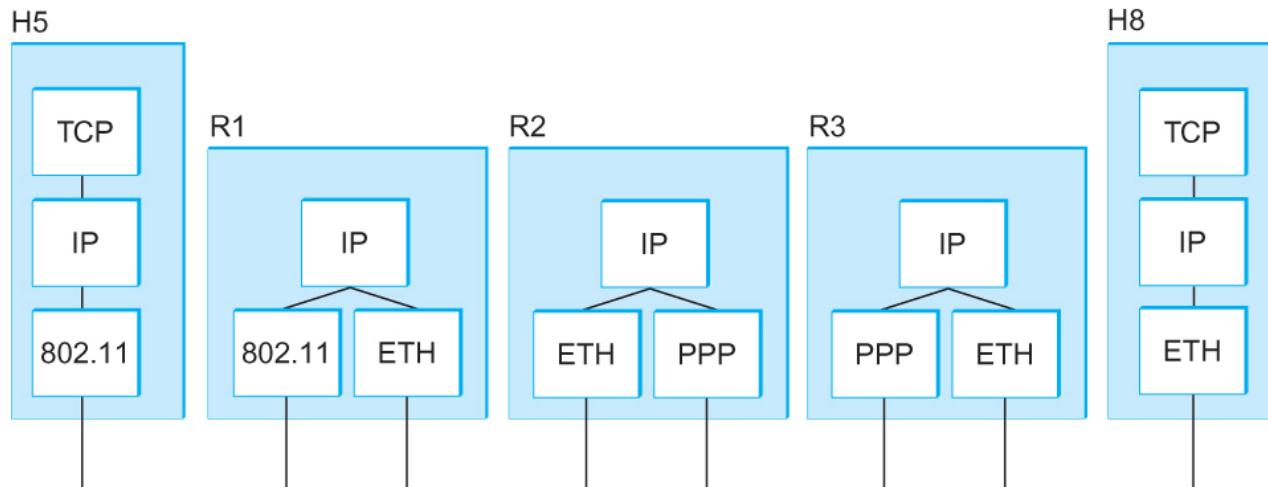
- What is internetwork
 - An arbitrary collection of networks interconnected to provide host-host packet delivery service



A simple internetwork where H represents hosts and R represents routers

Internetworking

- What is IP
 - IP stands for Internet Protocol
 - Key tool used today to build scalable, heterogeneous internetworks
 - It runs on all the nodes in a collection of networks and defines the infrastructure that allows these nodes and networks to function as a single logical internetwork



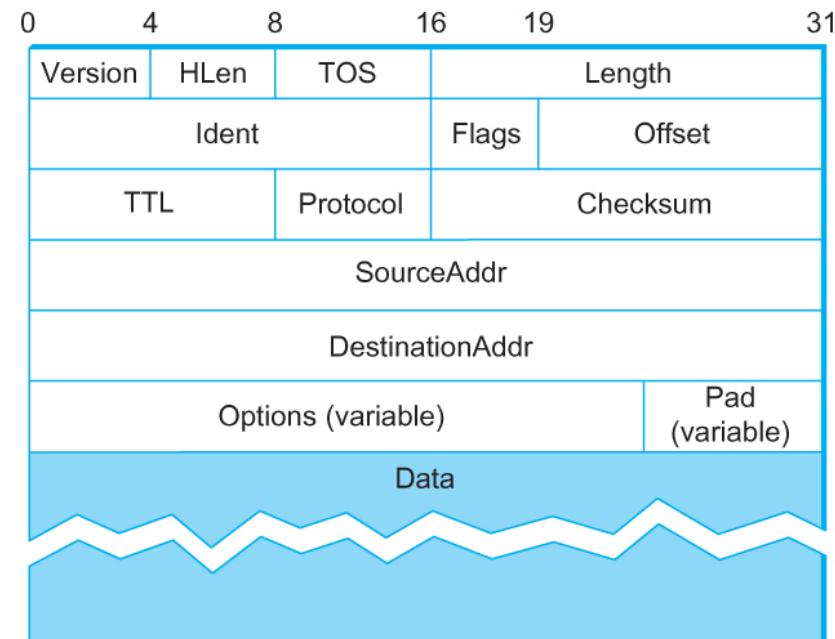
A simple internetwork showing the protocol layers

IP Service Model

- **Packet Delivery Model**
 - Connectionless model for data delivery
 - Best-effort delivery (unreliable service)
 - packets are lost
 - packets are delivered out of order
 - duplicate copies of a packet are delivered
 - packets can be delayed for a long time
- **Global Addressing Scheme**
 - Provides a way to identify all hosts in the network

Packet Format (IPv4)

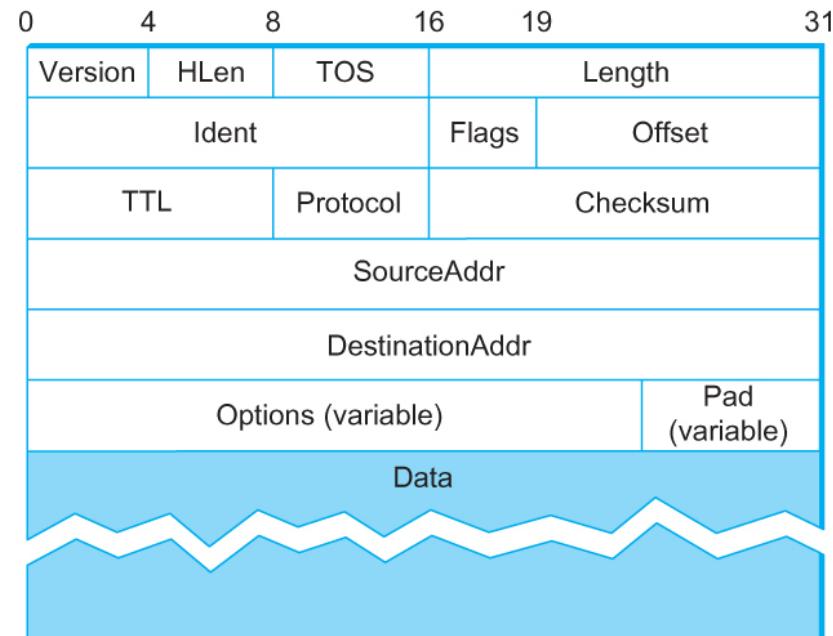
- Version (4 bits): IPv4/6
- Hlen (4 bits): number of 32-bit words in header. 5 words typically without Options
- TOS (8 bits): type of service (not widely used)
- Length (16 bits): number of bytes in this datagram – maximum datagram size is 65535 Bytes
- Ident (16 bits): used by fragmentation
- Flags/Offset (16 bits): used by fragmentation



[<Return>](#)

Packet Format (IPv4)

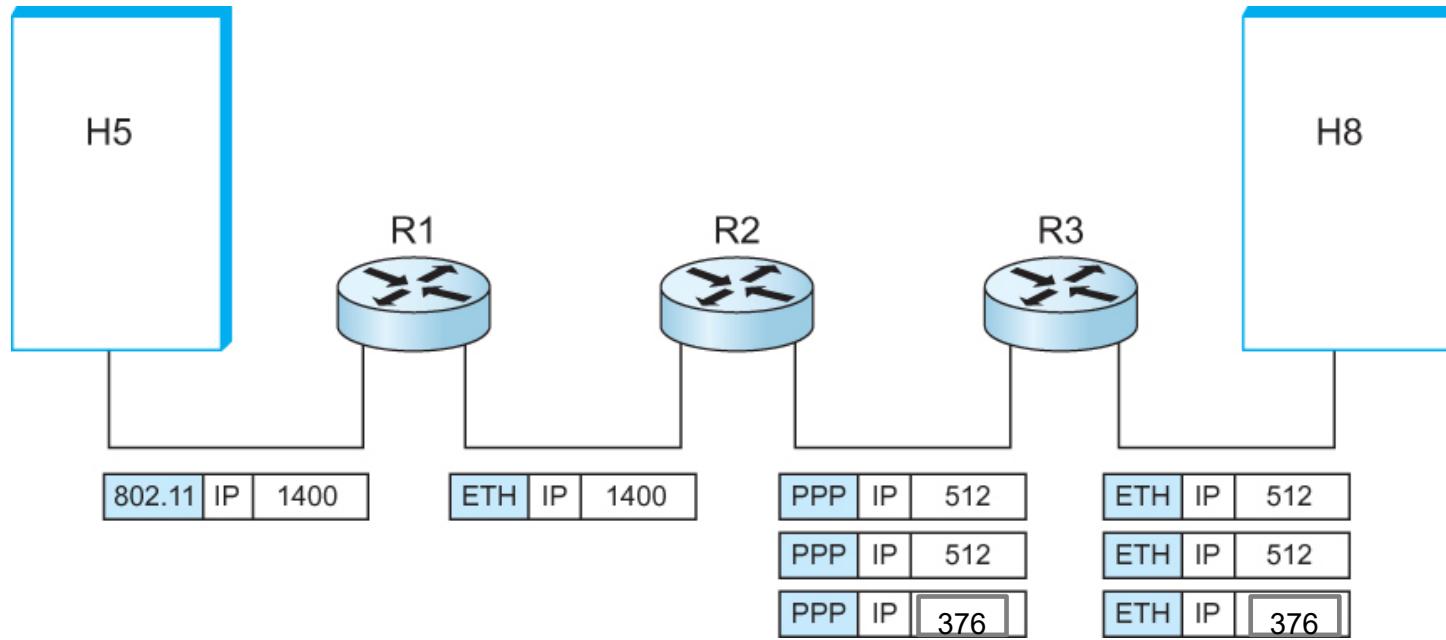
- TTL (8 bits): number of hops this datagram has traveled – typically it is a countdown timer
- Protocol (8): demux key (TCP=6, UDP=17)-higher level protocol using the packet info
- Checksum (16 bits): of the header only (IP checksum algorithm)
- DestAddr & SrcAddr (32 bits each, IPv6 has 128)
- Options/Pad (32 bits) – rarely used



IP Fragmentation and Reassembly

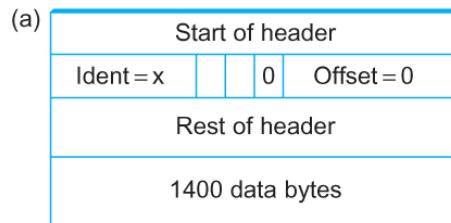
- Each network has some MTU (Maximum Transmission Unit)
 - Ethernet (1500 bytes)
- Strategy
 - Fragmentation occurs in a router when it receives a datagram
 - Reassembly is done at the receiving host
 - All the fragments carry the same identifier in the *Ident* field
 - Fragments are self-contained datagrams
 - Fragments re-encapsulate each IP datagram
 - IP does not recover from missing fragments

IP Fragmentation and Reassembly

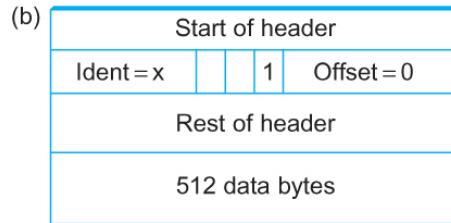


IP datagrams traversing the sequence of physical networks

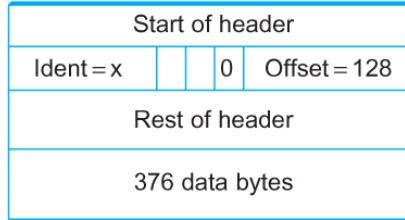
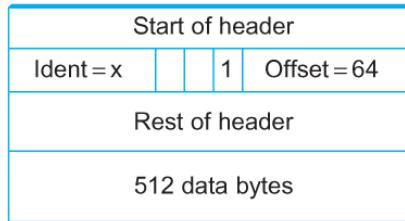
IP Fragmentation and Reassembly



(a) Total packet size = 1420 bytes
(20 bytes for header, 1400 for data)



(b) Total size of packets = $20*3 + 1400 = 1460$ Bytes
(3 headers @ 20 bytes each plus data)



Header fields used in IP fragmentation. (a) Unfragmented packet; (b) fragmented packets.

Global Addresses

- Properties
 - globally unique
 - hierarchical: network + host
 - 4 Billion IP address, half are A type, 1/4 is B type, and 1/8 is C type
- Format



(a) Class A, (b) Class B, (c) Class C

- Dot notation
 - 10.3.2.4
 - 128.96.33.81
 - 192.12.69.77

Global Addresses

Class	1 st Octet Decimal Range	1 st Octet High Order Bits	Network/ Host ID (N=Network, H=Host)	Default Subnet Mask	Number of Networks	Hosts per Network (Usable Addresses)**	
A	1 – 126*	0	N.H.H.H	255.0.0.0	126 (2^7 – 2)	16,777,214 (2^{24} – 2)	
B	128 – 191	10	N.N.H.H	255.255.0.0	16,382 (2^{14} – 2)	65,534 (2^{16} – 2)	
C	192 – 223	110	N.N.N.H	255.255.255.0	2,097,150 (2^{21} – 2)	254 (2^8 – 2)	
D	224 – 239	1110		Reserved for Multicasting			
E	240 – 254	1111		Experimental; used for research			

Note: *Class A addresses 127.0.0.0 to 127.255.255.255 cannot be used and are reserved for **loopback** and **diagnostic** functions.

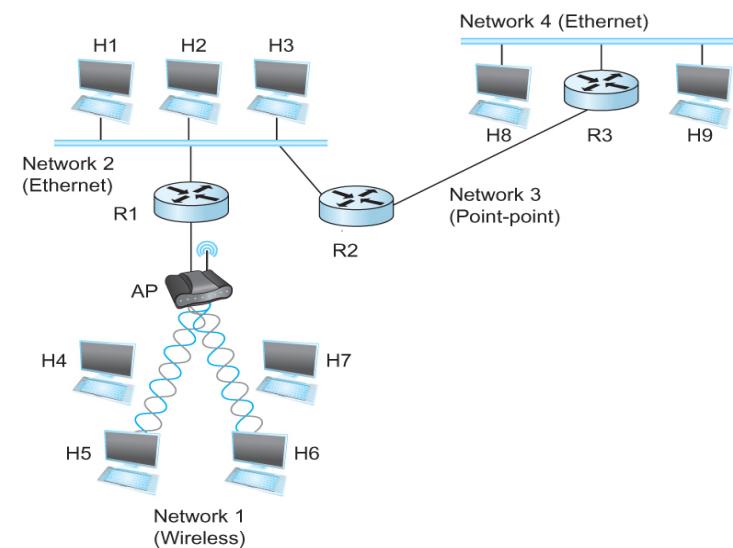
** host 255 is for broadcast, host 0 is not a valid host – identifies the network

IP Datagram Forwarding

- Strategy
 - every datagram contains destination's address
 - if directly connected to destination network, then forward to host
 - if not, then forward to some router
 - forwarding table maps network number into next hop
 - each host has a **default router**
 - each router maintains a forwarding table

- Example (router R2)

NetworkNum	NextHop
1	R1
2	Interface 1
3	Interface 0
4	R3



Subnetting

- Take one IP network number and break it up into subnets
- Adds another level to address hierarchy: *subnet*
- **Subnet masks** define variable partition of host part of class A and B addresses
- Subnets visible only within site of base IP network
- **Reason:** Allows for smaller number of hosts to be handled more efficiently
 - If a network is to connect 300 hosts
 - Need 2 class C networks – requires 2 Network addresses in tables
 - 1 class B network – requires one network address, but wastes over 65,000 hosts

Subnetting

- Assign a Class B network to a group that creates a subnet 192.11.0.0 to 192.11.255.255
- Router advertises network address 192.11.0.0/16 (/16 indicates first 16 bits as the network address)
 - Subnet mask is 255.255.XXX.0
 - The third octet of mask determines how the Class B network is subnetted

Start Address	End Address	Subnet Mask	Subnet IP
192.11.128.0	192.11.255.255	255.255.128.0	192.11.128.0
192.11.64.0	192.11.127.255	255.255.192.0	192.11.64.0
192.11.16.0	192.11.31.255	255.255.240.0	192.11.16.0
192.11.48.0	192.11.63.255	255.255.240.0	192.11.48.0

Subnetting

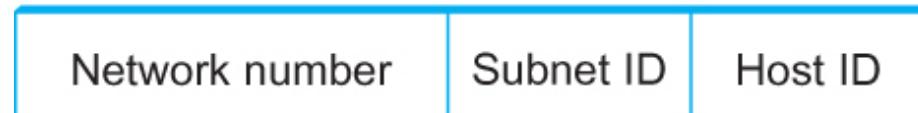
- Subnets visible only within site.
- Subnetted address is calculated by **bit-wise AND** operation between IP address and subnet mask.



Class B address

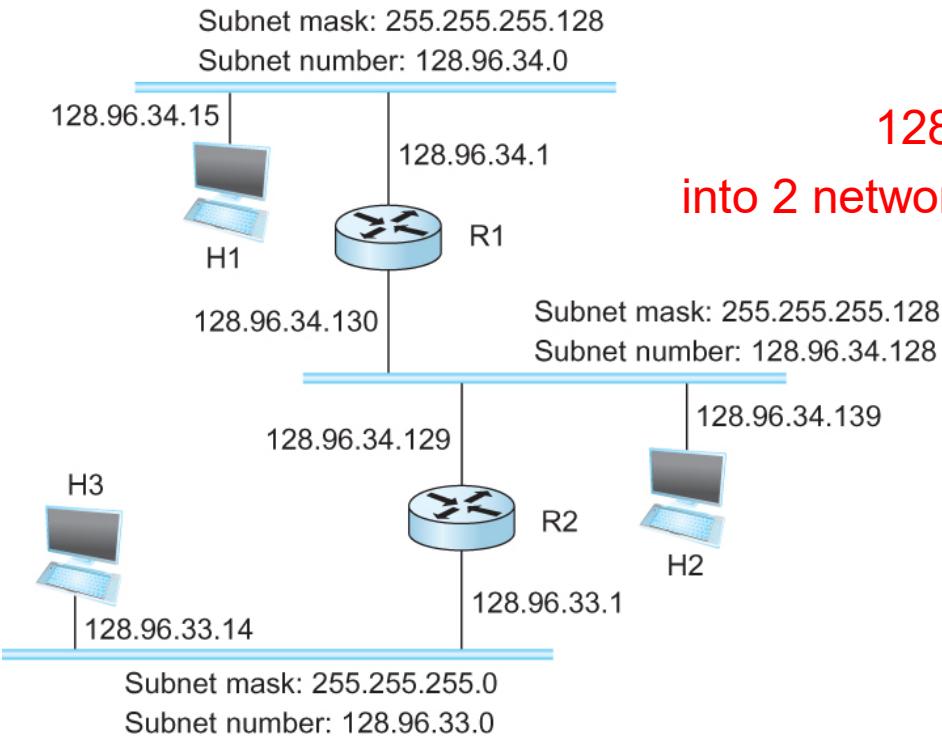


Subnet mask (255.255.255.0)



Subnetted address

Subnetting



128.96.34 is split
into 2 networks with 126 hosts each

- Forwarding Table at Router R1

SubnetNumber	SubnetMask	NextHop
128.96.34.0	255.255.255.128	Interface 0
128.96.34.128	255.255.255.128	Interface 1
128.96.33.0	255.255.255.0	R2

Subnetting

Forwarding Algorithm

```
D = destination IP address
for each entry < SubnetNum, SubnetMask, NextHop>
    D1 = SubnetMask & D
    if D1 = SubnetNum
        if NextHop is an interface
            deliver datagram directly to destination
        else
            deliver datagram to NextHop (a router)
```

Subnetting

Notes

- Would use a default router if nothing matches
- Subnets not visible from the rest of the Internet

Classless Addressing

- Classless Inter-Domain Routing (CIDR)
 - A technique that addresses two scaling concerns in the Internet
 - The growth of backbone routing table
 - Potential exhaustion of the 32-bit address space
 - Addresses assignment efficiency
 - IP address structure forces us to hand out network address space in fixed-size chunks
 - A network with two hosts needs a class C address
 - Address assignment efficiency = $2/255 = 0.78e-2$
 - A network with 256 hosts needs a class B address
 - Address assignment efficiency = $256/65535 = 0.39e-2$

Classless Addressing

Example:

- If a company has, say 16 class C network numbers assigned to it (4064 hosts),
 - Every Internet backbone router needs 16 entries in its routing tables for them
 - Even if the path to every one of class C networks is the same
- If we had assigned a class B address to them
 - The same routing information can be stored in one entry
 - But Efficiency(% used) = $16 \times 256 / 65,536 = 6.25\%$

Classless Addressing

- CIDR tries to **balance** the desire to minimize the number of routes that a router needs to know against the need to hand out addresses efficiently.
- CIDR uses aggregate routes
 - Uses a single entry in the forwarding table to tell the router how to reach a lot of different networks

Classless Addressing

- Consider a company with 16 class C network numbers.
- Instead of handing out 16 addresses at random, hand out a block of contiguous class C addresses
- Suppose we assign the class C network numbers from 192.4.16 through 192.4.31
- Observe that top 20 bits of all the addresses in this range are the same (11000000 00000100 0001)

Classless Addressing

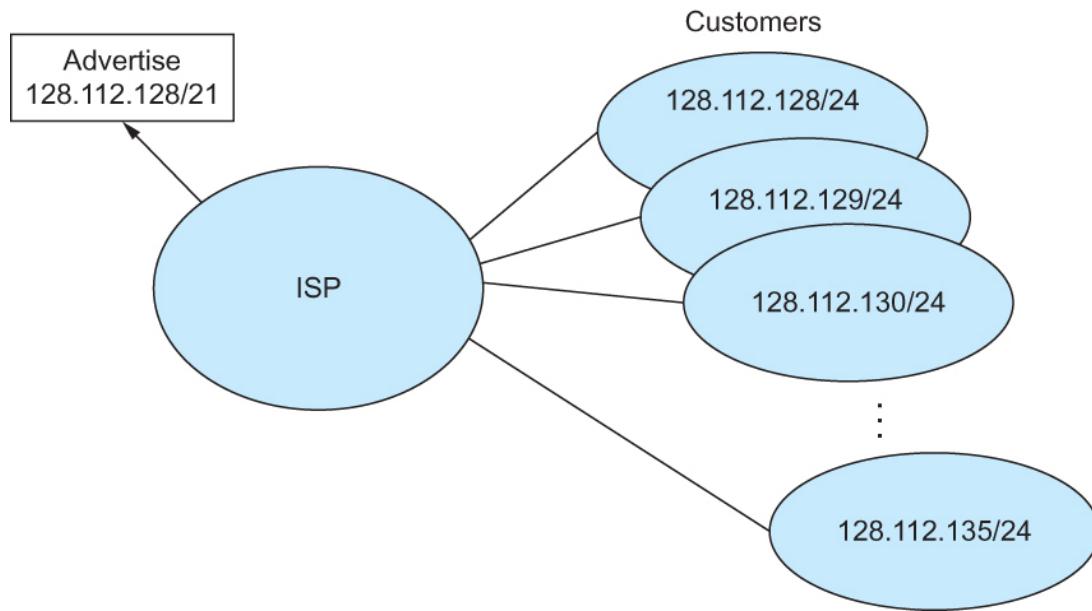
- The convention is to place a /X after the prefix where X is the prefix length in bits
- In this example, the 20-bit prefix for all the networks 192.4.16 through 192.4.31 is represented as **192.4.16/20**

Subnetting

- Want a network router to handle 8 class C networks
192.11.64.XXX to 192.11.71.XXX
- Router advertises network address 192.11.64/21
 - Subnet mask is 255.255.248.0
 - The third octet of IP addresses are masked with 11111000 (248)

Third Octet	3 rd Octet Binary	Masked	Subnet IP
63	00111111	00111000	192.11.56.0
64	01000000	01000000	192.11.64.0
71	01000111	01000000	192.11.64.0
72	01001000	01001000	192.11.72.0

Classless Addressing



Route aggregation with CIDR

Classless Addressing

- It is also possible to have prefixes in the forwarding tables that overlap
- For example, we might find both 171.69 (a 16 bit prefix) and **171.69.10** (a 24 bit prefix) in the forwarding table of a single router
- A packet destined to 171.69.10.5 clearly matches both prefixes - The rule is based on the “longest match”!

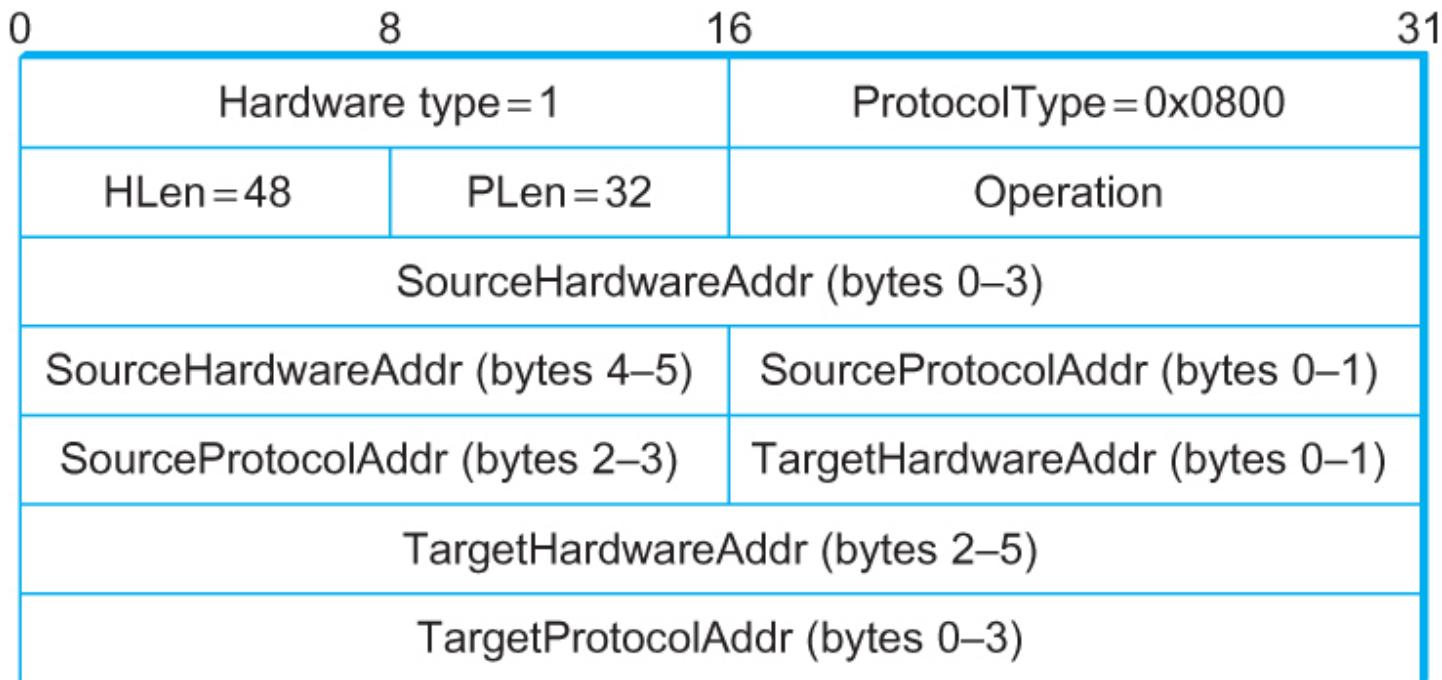
Address Translation Protocol (ARP)

- Map IP addresses into physical (MAC) addresses
- ARP (Address Resolution Protocol)
 - table of IP to physical address bindings
 - broadcast request if IP address not in table
 - target machine responds with its physical address
 - table entries are discarded if not refreshed

Rachel

ریتاشل

ARP Packet Format



- **HardwareType:** type of physical network (e.g., Ethernet)
- **ProtocolType:** type of higher layer protocol (e.g., IP)
- **HLEN & PLEN:** length of physical(MAC) and protocol(IP) addresses
- **Operation:** ARP request or ARP response
- **Source/Target Physical(Ethernet)/Protocol(IP) addresses**

Host Configurations

- When a computer is firstly connected to the Internet,
 - A unique IP address is assigned
 1. Either automatically,
 2. Or, its OS manually configures it
- Drawbacks of manual configuration
 - A lot of work to configure all the hosts in a large network
 - Configuration process is error-prone

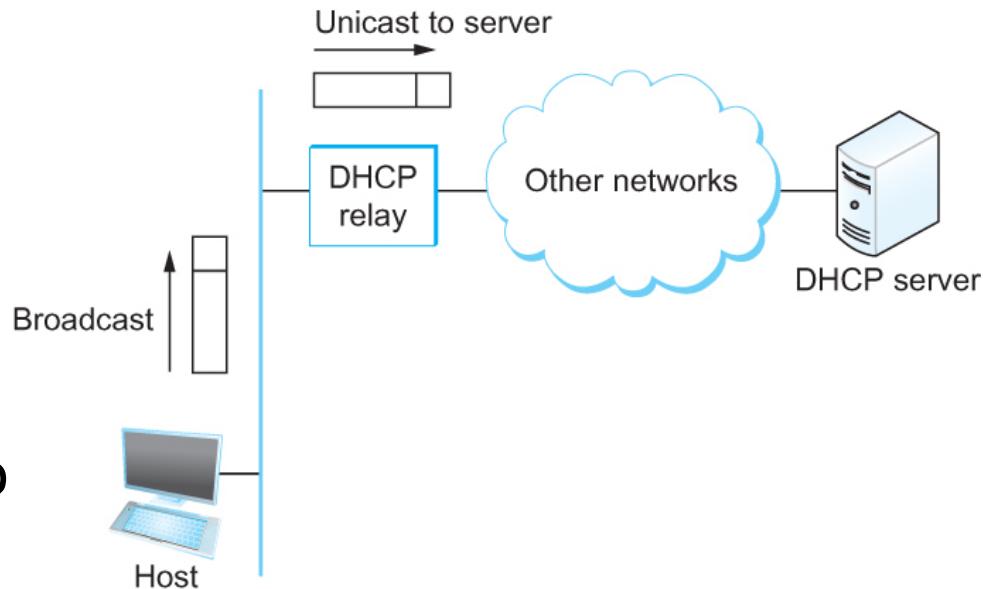
Dynamic Host Configuration Protocol (DHCP)

- DHCP server is responsible for providing automatic IP configuration to hosts
 - There is at least one DHCP server for an administrative domain
 - DHCP server maintains a pool of available addresses
 - DHCP leases an address to a host. Host must renew the lease periodically, in case of disconnected.



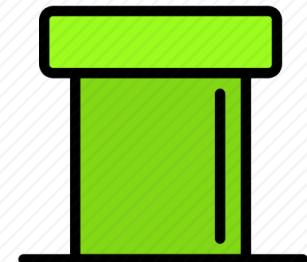
DHCP

- New host sends **DHCPDISCOVER** message to a special IP address (**255.255.255.255**)
- DHCP relay agent unicasts the message to DHCP server and waits for the response

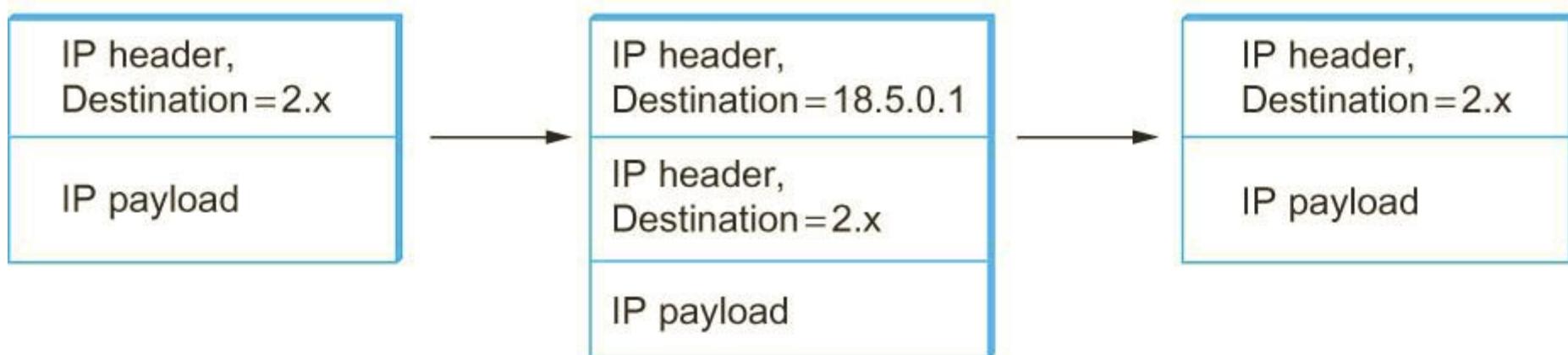
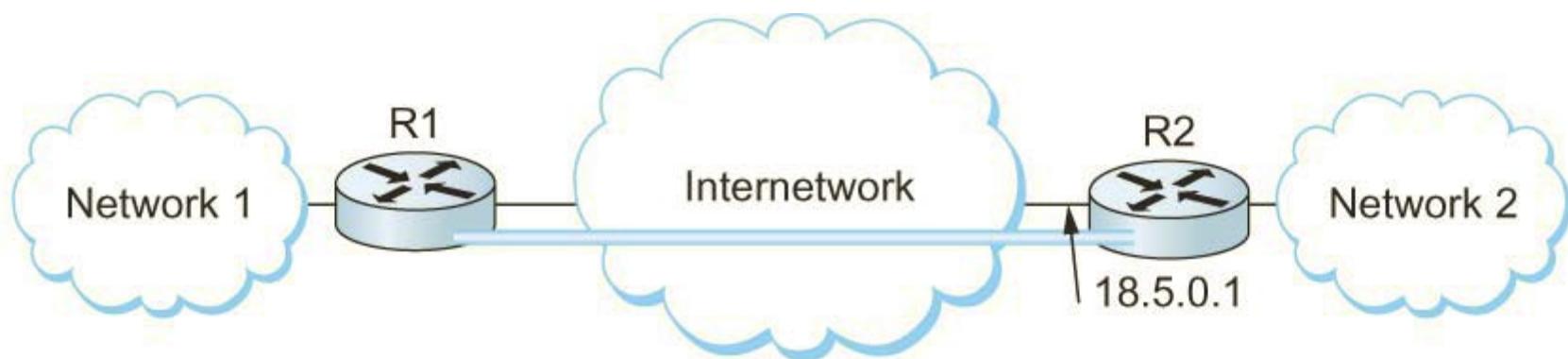


Virtual Networks and Tunnels

- VPN – virtual Private Network
 - Use Virtual point-to-point links on a shared network
 - For IP, use a concept called tunneling
- IP Tunnel
 - Router to Router transmission of an IP Packet
 - It encapsulates the entire IP packet from source to destination.



Virtual Networks and Tunnels



IP Tunnels

- IP Tunnel Advantages
 - Provides security of transmissions
 - Can carry packets from protocols different from IP
 - Can force a packet to be delivered to a particular destination – used with mobile hosts
- IP Tunnel Disadvantages
 - Longer packets are created
 - More work is required at the edge router

CPE348: Introduction to Computer Networks

Lecture #12: Chapter 3.5



Jianqing Liu
Assistant Professor of Electrical and Computer
Engineering, University of Alabama in Huntsville
jianqing.liu@uah.edu
<http://jianqingliu.net>

Routing

Forwarding, Switching or Routing

- Forwarding:
 - one device sending a datagram to the next one **in the path to** the destination
- Switching:
 - moving a datagram from one interface to another **within** a device
- Routing:
 - a specific process in a **layer-3 device** to decide what to do with a layer-3 packet

Routing

- Forwarding table vs Routing table
 - Forwarding table
 - It contains the mapping from a network number to an outgoing interface and some MAC information, such as Ethernet Address of the next hop
 - Routing table
 - It contains mapping from network numbers to next hops

Routing

(a)

Prefix/Length	Next Hop
18/8	171.69.245.10

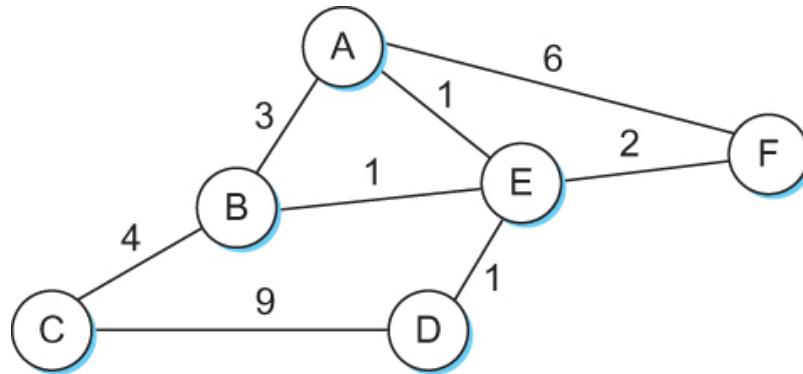
(b)

Prefix/Length	Interface	MAC Address
18/8	if0	8:0:2b:e4:b:1:2

Example rows from (a) routing and (b) forwarding tables

Routing

- Network as a Graph



- The basic problem of routing is to find the **lowest-cost path** between any two nodes
- The term – “cost” – has many different meanings:
 - Latency/delay
 - Number of hops
 - Error probability
 - ...

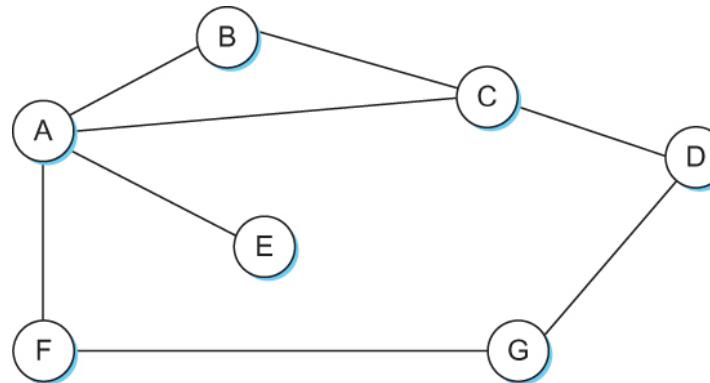
Routing

Is there any algorithm that can calculate the shortest path?

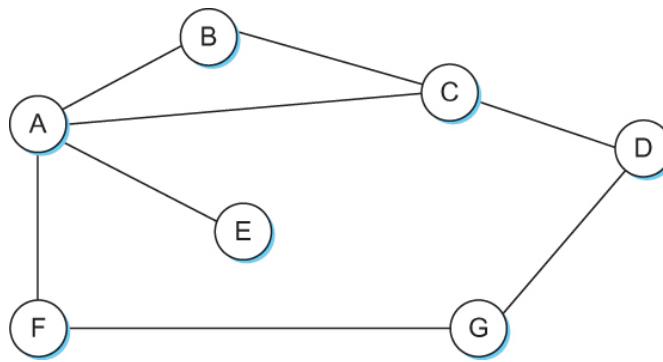
- Centralized approach:
 - A central controller calculates all shortest paths and load them into these routers. **Pros & Cons?**
- Distributed approach:
 - Two main classes of protocols
 - Distance Vector
 - Link State

Distance Vector Routing Algorithm

- Each node constructs a vector containing the “costs” to all other nodes
- It then distributes that vector to its immediate neighbors
- Open question: how could a node know the cost?



Distance Vector - Example

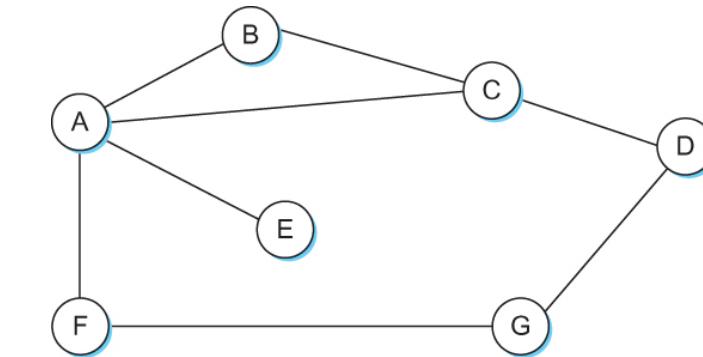


Information Stored at Node	Distance to Reach Node						
	A	B	C	D	E	F	G
A	0	1	1	∞	1	1	∞
B	1	0	1	∞	∞	∞	∞
C	1	1	0	1	∞	∞	∞
D	∞	∞	1	0	∞	∞	1
E	1	∞	∞	∞	0	∞	∞
F	1	∞	∞	∞	∞	0	1
G	∞	∞	∞	1	∞	1	0

Initial distances stored at each node (global view)

Use hop count as the cost

Distance Vector - Example



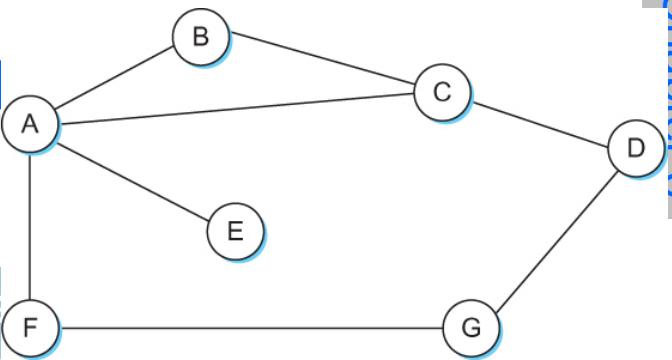
Destination	Cost	NextHop
B	1	B
C	1	C
D	∞	—
E	1	E
F	1	F
G	∞	—

Initial routing table at node A

Distance Vector - Exam

Information Stored at Node	Distance to Reach Node						
	A	B	C	D	E	F	G
A	0	1	1	∞	1	1	∞
B	1	0	1	∞	∞	∞	∞
C	1	1	0	1	∞	∞	∞
D	∞	∞	1	0	∞	∞	1
E	1	∞	∞	∞	0	∞	∞
F	1	∞	∞	∞	∞	0	1
G	∞	∞	∞	1	∞	1	0

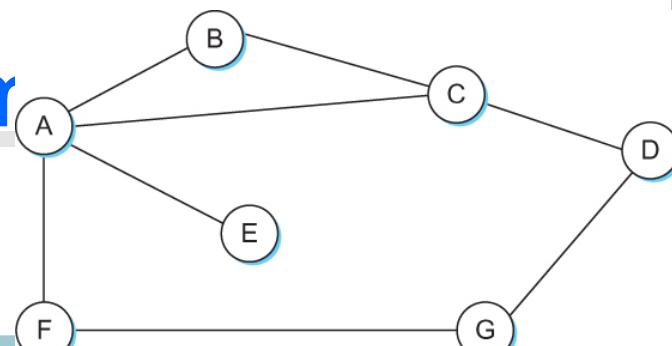
Information Stored at Node	Distance to Reach Node						
	A	B	C	D	E	F	G
A	0	1	1	2	1	1	2
B	1	0	1	2	2	2	∞
C	1	1	0	1	2	2	2
D	2	2	1	0	∞	2	1
E	1	2	2	∞	0	2	∞
F	1	2	2	2	2	0	1
G	2	∞	2	1	∞	1	0



Distances stored at each node after 1 update (global view)

Distance Vector - Exam

Information Stored at Node	Distance to Reach Node						
	A	B	C	D	E	F	G
A	0	1	1	2	1	1	2
B	1	0	1	2	2	2	∞
C	1	1	0	1	2	2	2
D	2	2	1	0	∞	2	1
E	1	2	2	∞	0	2	∞
F	1	2	2	2	2	0	1
G	2	∞	2	1	∞	1	0

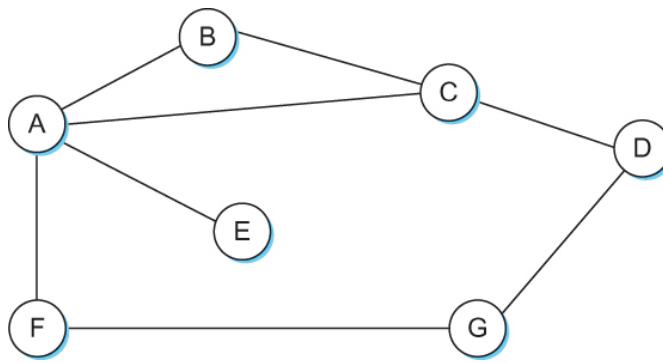


The graph shows a network of 7 nodes (A, B, C, D, E, F, G) connected as follows: A is connected to B, C, and F; B is connected to A and C; C is connected to A, B, D, and E; D is connected to C and G; E is connected to C; F is connected to A and G; and G is connected to D and F.

Information Stored at Node	Distance to Reach Node						
	A	B	C	D	E	F	G
A	0	1	1	2	1	1	2
B	1	0	1	2	2	2	3
C	1	1	0	1	2	2	2
D	2	2	1	0	3	2	1
E	1	2	2	3	0	2	3
F	1	2	2	2	2	0	1
G	2	3	2	1	3	1	0

Distances stored at each node after 2 updates (global view)

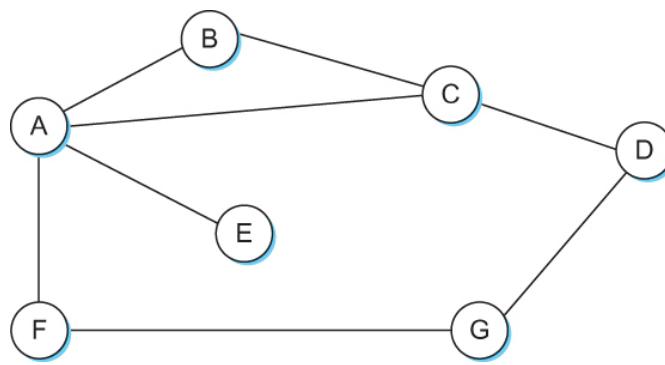
Distance Vector - Example



Information Stored at Node	Distance to Reach Node						
	A	B	C	D	E	F	G
A	0	1	1	2	1	1	2
B	1	0	1	2	2	2	3
C	1	1	0	1	2	2	2
D	2	2	1	0	3	2	1
E	1	2	2	3	0	2	3
F	1	2	2	2	2	0	1
G	2	3	2	1	3	1	0

Final distances stored at each node (global view)

Distance Vector - Example



Destination	Cost	NextHop
B	1	B
C	1	C
D	2	C
E	1	E
F	1	F
G	2	F

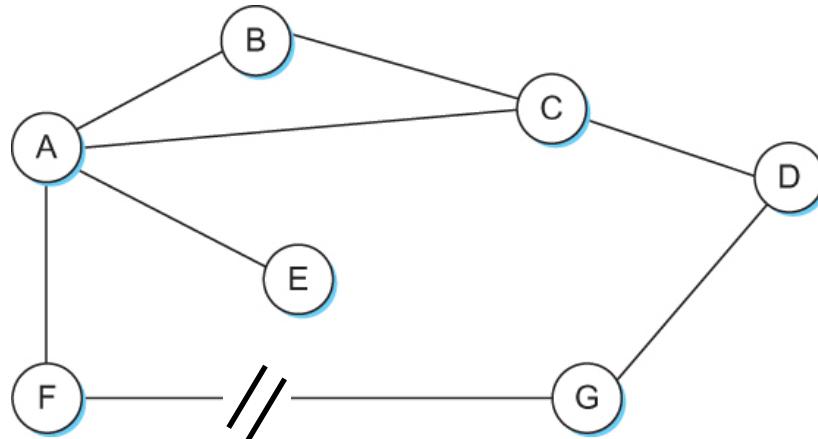
Final routing table at node A

Distance Vector Routing Algorithm

- The distance vector routing algorithm is also called as Bellman-Ford algorithm
- Every T seconds each router sends its table to its neighbor, which then updates its table based on the new information

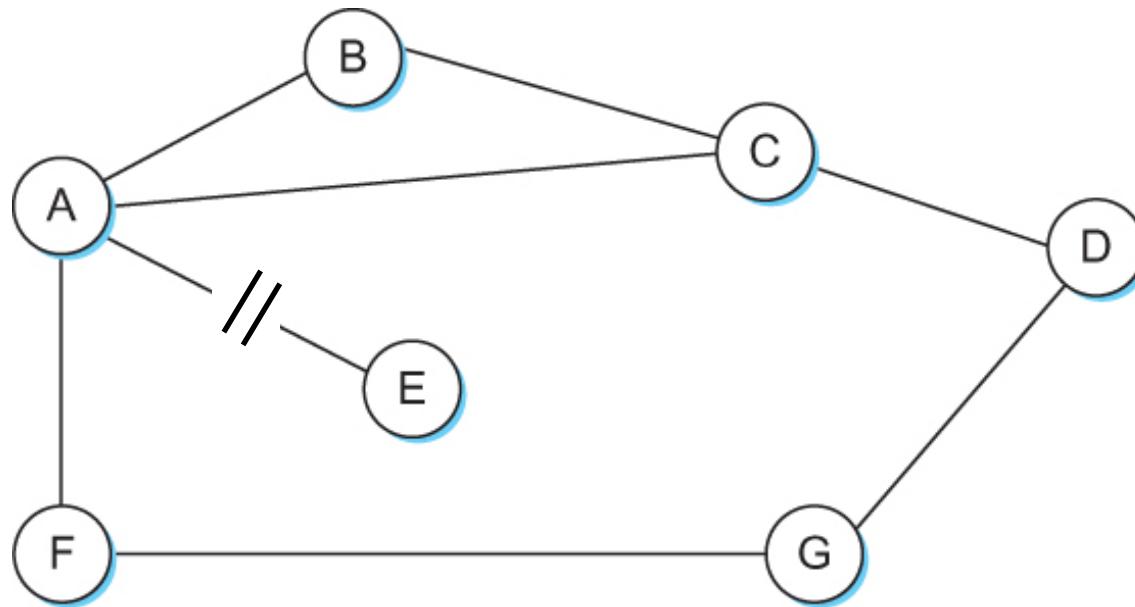
DVR Algorithm – issue

- When a node detects a link failure
 - F detects that link to G has failed
 - F sets distance to G to infinity and sends update to A
 - A sets distance to G to infinity since it uses F to reach G
 - A receives periodic update from C with 2-hop path to G
 - A sets distance to G to 3 and sends update to F
 - F decides it can reach G in 4 hops via A



DVR Algorithm – issue

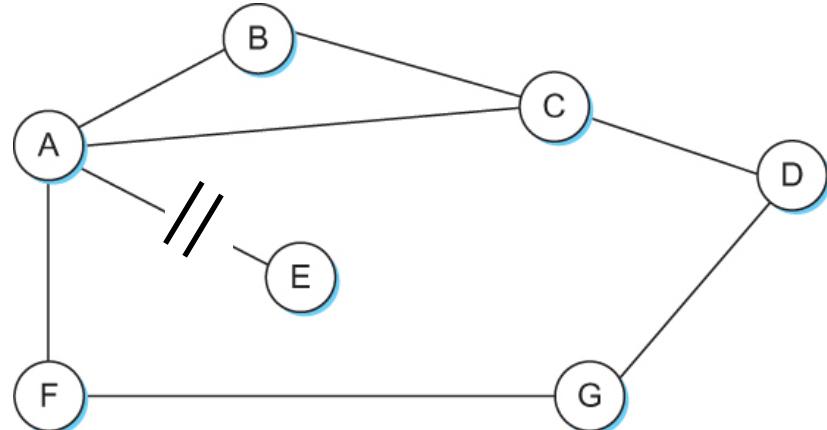
- Another example:
 - Suppose the link from A to E goes down
 - In the next round of updates, A advertises a distance of infinity to E, but B and C advertise a distance of 2 to E



DVR Algorithm – issue

- Depending on the exact timing of events, the following might happen
 - Node B, upon hearing that E can be reached in 2 hops from C, concludes that it can reach E in 3 hops and advertises this to A
 - Node A concludes that it can reach E in 4 hops and advertises this to C
 - Node C concludes that it can reach E in 5 hops; and so on.

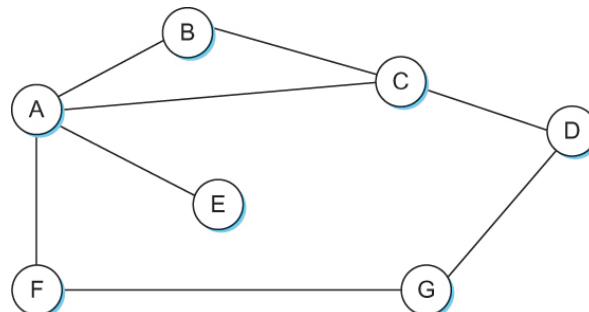
Count-to-infinity problem!



Count-to-infinity Problem – remedy

- Use an upper bound to force stop – i.e. 16
- Another technique to improve the time to stabilize routing is called *split horizon*
 - When a node sends a routing update to its neighbors, it does not send those routes it learned from each neighbor back to that neighbor

For example, if B has the route (E, 2, A) in its table, then it knows it learned this route from A, and so whenever B sends a routing update to A, it does not include the route (E, 2) in that update



Link State Routing

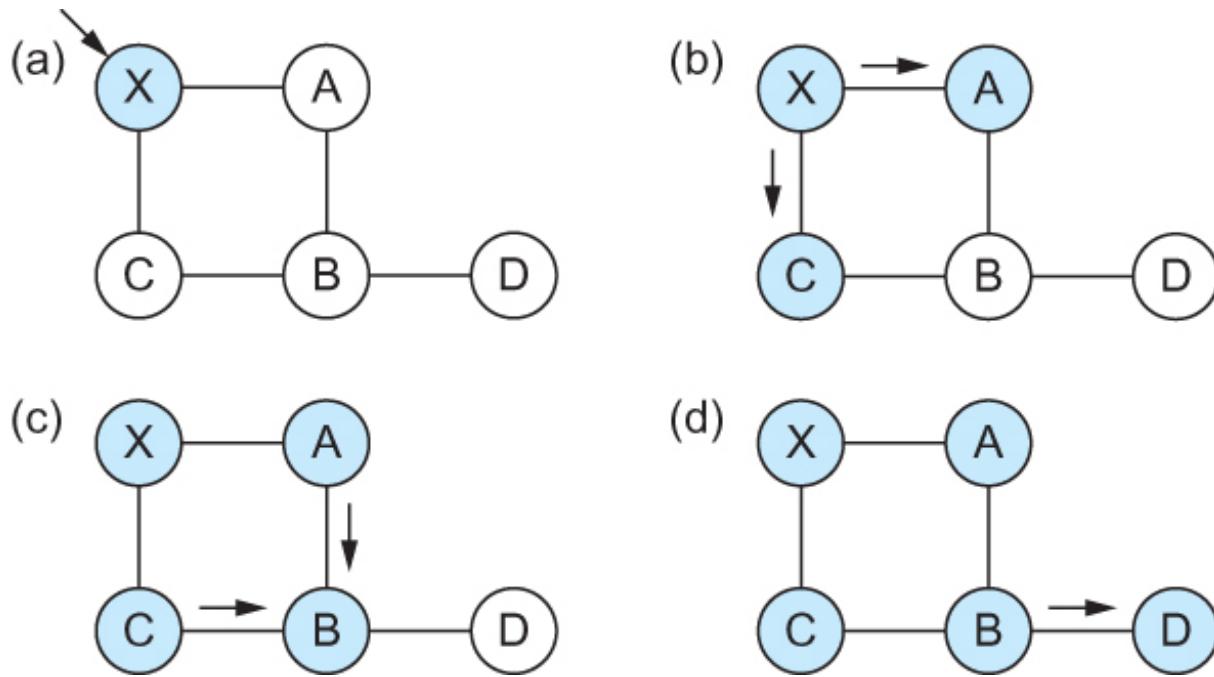
Strategy: Send to all nodes (not just neighbors) information about directly connected links (not entire routing table).

- Link State Packet (LSP)
 - ID of the node that created the LSP
 - cost of link to each directly connected neighbor
 - sequence number (SEQNO)
 - time-to-live (TTL) for this packet
- Reliable Flooding
 - forward LSP to all nodes but one that sent it
 - start SEQNO = 0 and increment
 - decrement TTL of each stored LSP;
 - discard when TTL=0



Flood Link State Packets

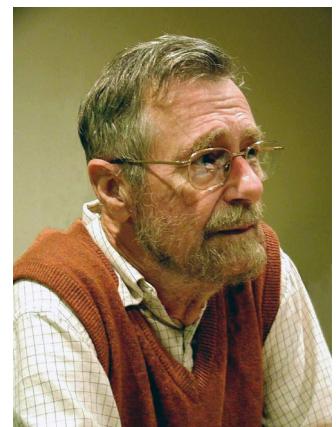
Reliable Flooding



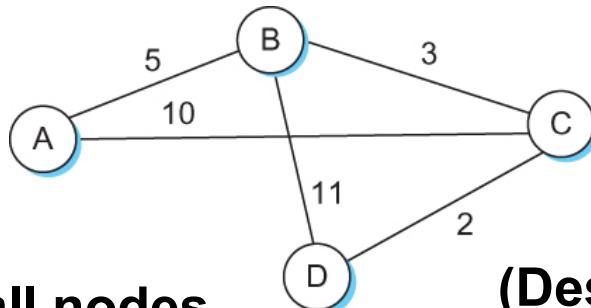
Flooding of link-state packets. (a) LSP arrives at node X; (b) X floods LSP to A and C; (c) A and C flood LSP to B (but not X); (d) flooding is complete

Link State Routing

- Each router computes its routing table from the LSP's it has collected.
 - The calculation is based on the **Dijkstra's algorithm** a.k.a., *forward search algorithm*
 - i. Each router maintains two lists, known as **Tentative** and **Confirmed**
 - ii. Each of these lists contains a set of entries of the form (Destination, Cost, NextHop)
 - iii. Algorithm converges when **Tentative** list become null



Shortest Path Routing-Forward Search



D has LSP's of all nodes

(Destination, Cost, NextHop)

Step	Confirmed	Tentative	Comments
1	(D,0,–)		Since D is the only new member of the confirmed list, look at its LSP.
2	(D,0,–)	(B,11,B) (C,2,C)	D's LSP says we can reach B through B at cost 11, which is better than anything else on either list, so put it on Tentative list; same for C.
3	(D,0,–) (C,2,C)	(B,11,B)	Put lowest-cost member of Tentative (C) onto Confirmed list. Next, examine LSP of newly confirmed member (C).
4	(D,0,–) (C,2,C)	(B,5,C) (A,12,C)	Cost to reach B through C is 5, so replace (B,11,B). C's LSP tells us that we can reach A at cost 12.
5	(D,0,–) (C,2,C) (B,5,C)	(A,12,C)	Move lowest-cost member of Tentative (B) to Confirmed, then look at its LSP.
6	(D,0,–) (C,2,C) (B,5,C)	(A,10,C)	Since we can reach A at cost 5 through B, replace the Tentative entry.
7	(D,0,–) (C,2,C) (B,5,C) (A,10,C)		Move lowest-cost member of Tentative (A) to Confirmed, and we are all done.

Distance-Vector vs Link-State

- **Distance-Vector (e.g. RIP):**
 - Each node talks with its neighbors only
 - Sends all information it knows - known distances to other nodes
 - Speed of **convergence is slower** than LS
 - **Stabilization may not occur** – count to infinity
 - Simple algorithm
- **Link-State (e.g. OSPF):**
 - Each node talks to all other nodes
 - Sends what it knows for sure - State of its directly connected links
 - **Stabilizes quickly** and it responds rapidly to network changes
 - Low traffic generation
 - Storage at each node is large
 - Uses reliable flooding of packets

Metrics – Cost of Links

- Assign 1 to all links – hop count
- Latency – take into account delay of the link
- Capacity – what is BW of each link
- Current Load – increase cost as load increases
- Queue length (average value between updates)

- Metrics are fixed by administrators – not dynamically changing due to stability issues.

More Practices

- What if you are given a network graph of bilateral links?
- What if a link fails when the algorithm is running?

Appendix – pseudo codes of Dijkstra's algorithm

- Dijkstra's Algorithm - Assume non-negative link weights
 - N : set of nodes in the graph
 - $L(i, j)$: the non-negative cost associated with the edge between nodes $i, j \in N$ and $L(i, j) = \infty$ if no edge connects i and j
 - Let $s \in N$ be the starting node which executes the algorithm to find shortest paths to all other nodes in N
 - Two variables used by the algorithm
 - M : set of nodes incorporated so far by the algorithm
 - $C(n)$: the cost of the path from s to each node n
 - The algorithm

$M = \{s\}$

For each n in $N - \{s\}$

$C(n) = L(s, n)$

while ($N \neq M$)

$M = M \cup \{w\}$ such that $C(w)$ is the minimum
for all w in $(N - M)$

For each n in $(N - M)$

$C(n) = \text{MIN } (C(n), C(w) + L(w, n))$

Appendix – pseudo codes of Dijkstra's algorithm

■ The algorithm

- 1) Initialize the **Confirmed** list with an entry for myself; this entry has a cost of 0
- 2) For the node just added to the **Confirmed** list in the previous step, call it node **Next**, select its LSP
- 3) For each neighbor (Neighbor) of **Next**, calculate the cost (Cost) to reach this Neighbor as the sum of the cost from myself to Next and from Next to Neighbor
 - a) If Neighbor is currently on neither the **Confirmed** nor the **Tentative** list, then add (Neighbor, Cost, Nexthop) to the **Tentative** list, where Nexthop is the direction I go to reach Next
 - b) If Neighbor is currently on the **Tentative** list, and the Cost is less than the currently listed cost for the Neighbor, then replace the current entry with (Neighbor, Cost, Nexthop) where Nexthop is the direction I go to reach Next
 - c) If Neighbor on **Confirmed** list, then ignore it
- 4) If the **Tentative** list is empty, stop. Otherwise, pick the entry from the **Tentative** list with the lowest cost, move it to the **Confirmed** list, and return to Step 2.

CPE348: Introduction to Computer Networks

Lecture #13: Chapter 4.1



Jianqing Liu
Assistant Professor of Electrical and Computer
Engineering, University of Alabama in Huntsville
jianqing.liu@uah.edu
<http://jianqingliu.net>

Chapter 4 – Advanced Networking

- How do we build a system that can
 - handle hundreds of thousands of **networks**,
 - host billions of **end nodes**?
- How to enhance the functionalities of Internet?

Chapter Outline

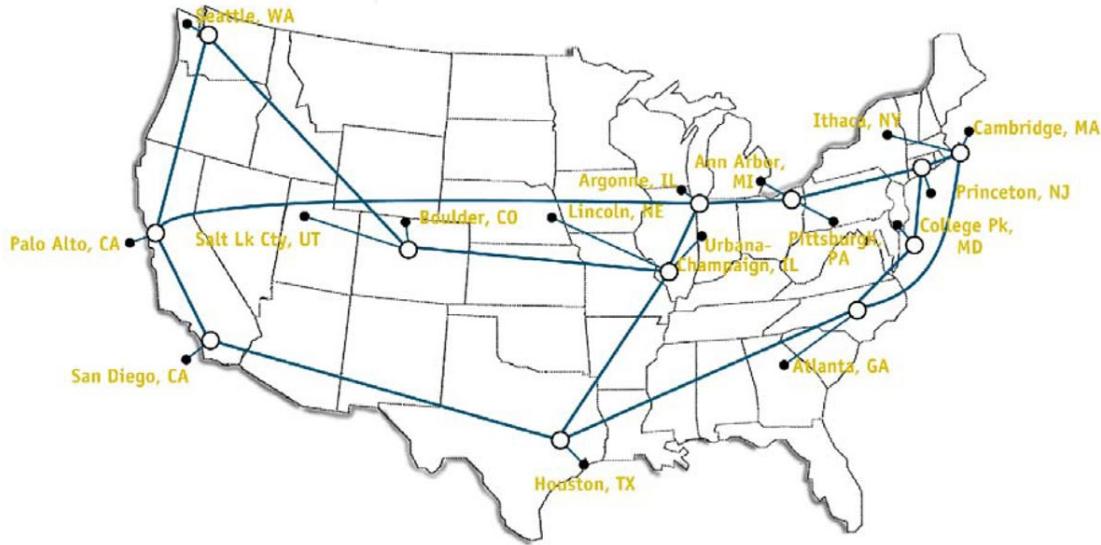
- Global Internet
- Multicast
- Mobile IP

Chapter Goal

- Understanding the scalability of routing in the Internet
- Discussing IPv6
- Understanding the concept of multicasting
- Discussing Mobile IP

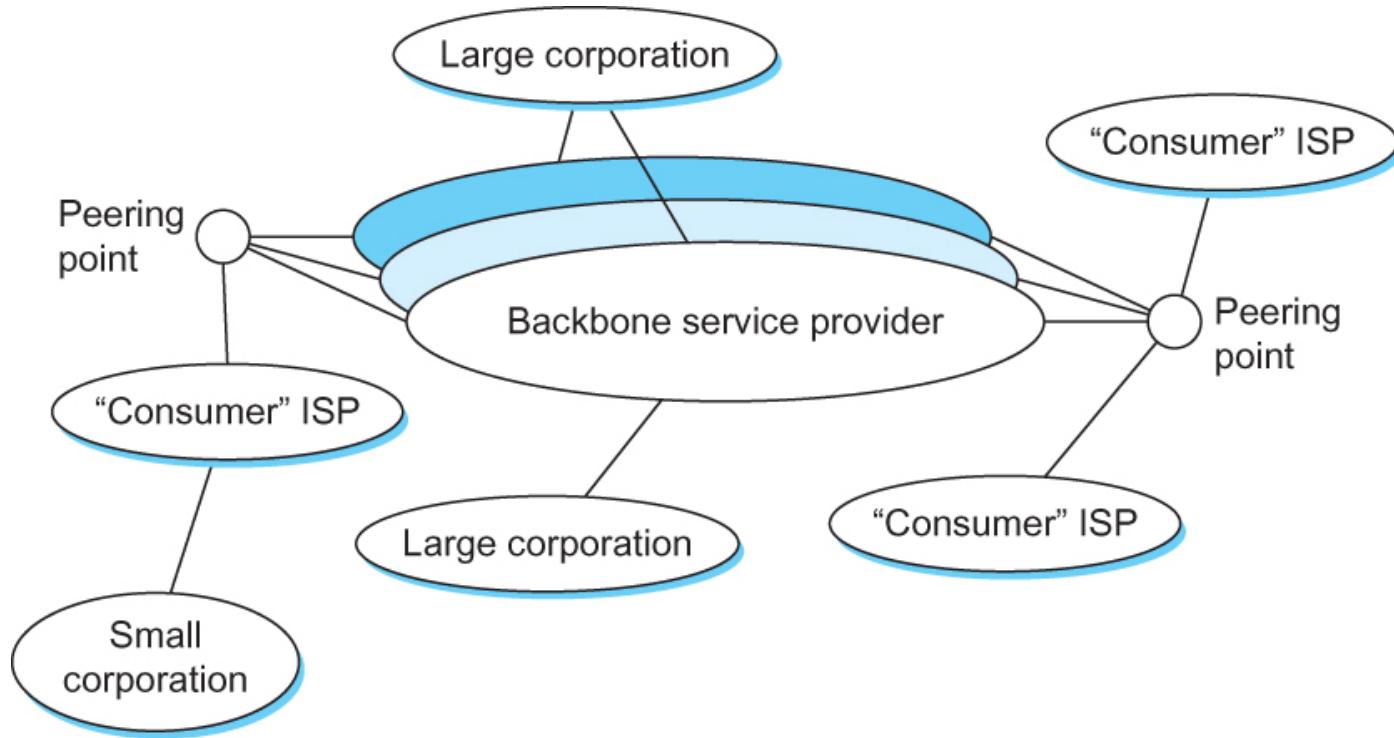
The Global Internet – history

NSFNET T3 Network 1992



National Science Foundation Network (NSFNET) program
1988-1992

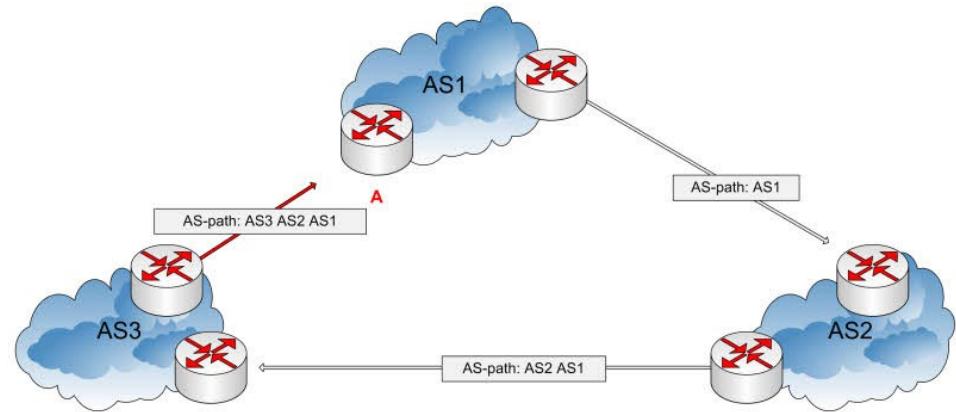
The Global Internet



Global Internet has a **tree structure** and operated by **multiple service providers**.

Interdomain Routing

- Internet is organized by interconnected autonomous systems.
- Autonomous System (AS)
 - corresponds to an administrative domain
 - examples: University, company, backbone network



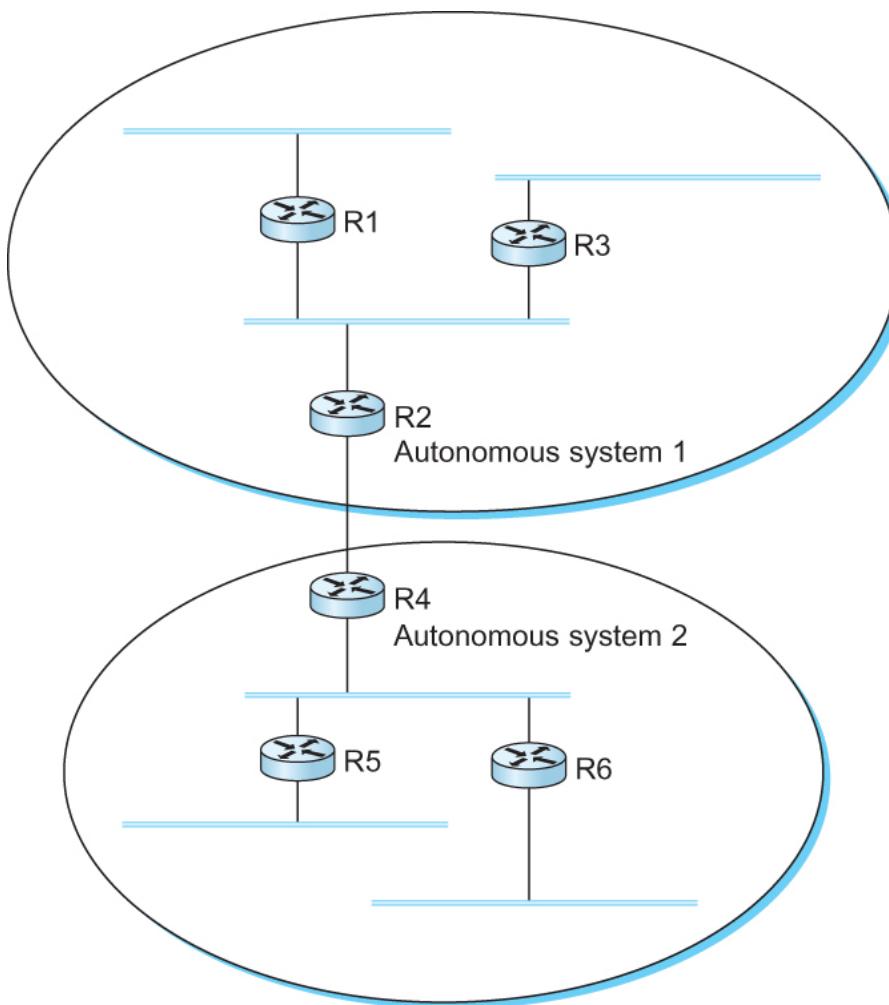
Interdomain Routing

What is the difference between a ISP and an AS?

ISP: Cogent, AT&T, etc.

AS: UAH, Intel, etc.

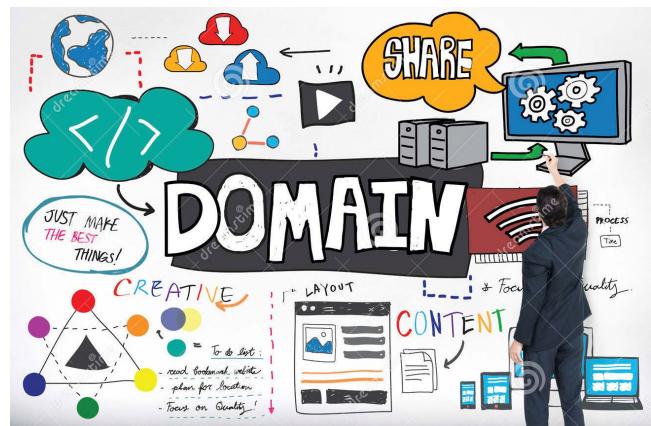
Interdomain Routing



A network with two autonomous systems

Route Propagation

- Idea: a hierarchical way to disseminate routing information to a large internet.
 - Improves scalability **WHY NOT** if non-hierarchical?
- Divide the routing problem into:
 - Routing within a single AS - **Intra-domain routing protocol**
 - Routing between ASs - **Inter-domain routing protocol**

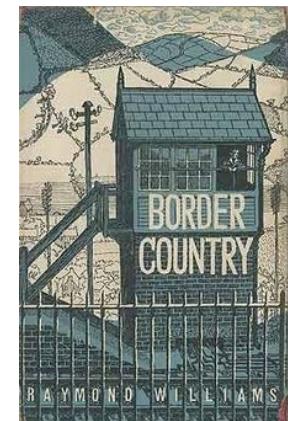


Intra-domain routing

We've studied it in the previous chapter!

Inter-domain routing

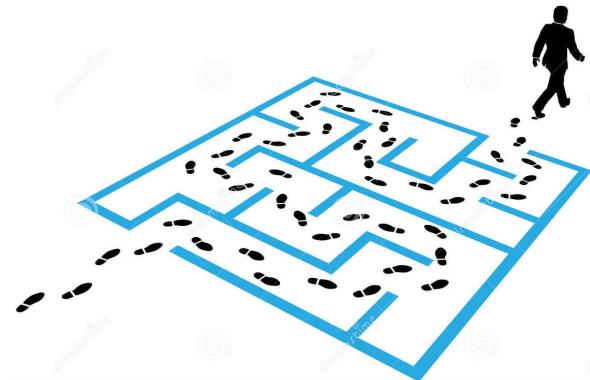
- Inter-domain Routing Protocols
 - Exterior Gateway Protocol (EGP) – first attempt
 - Forced a tree-like topology onto the Internet
 - Did not allow for the topology to become general
 - Tree-like structure: a single backbone and ASs are connected only as parents and children and not as peers.
 - Border Gateway Protocol (BGP) – replaces EGP
 - Assumes that the Internet is an arbitrarily interconnected set of ASs.



BGP: Border Gateway Protocol

The goal of BGP

- To find **one path** to the dest. that is **loop free**
 - Reachability more than Optimality
 - Optimal path is hard to find. **WHY?**



BGP: Border Gateway Protocol

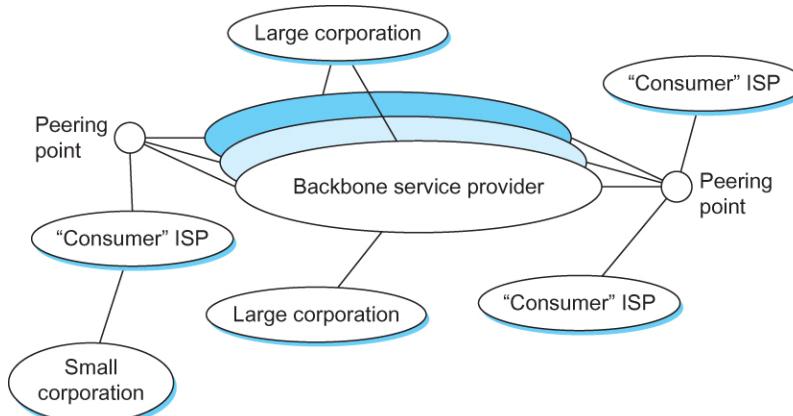
Design foundations for BGP:

- Define **local traffic** as traffic that originates at or terminates on nodes within an AS.
- Define **transit traffic** as traffic that passes through an AS.

BGP: Border Gateway Protocol

Depending on the traffic type:

- *Stub AS*: only connect to one other AS; only carry **local traffic** (e.g., *small corporation*).
-
- *Multihomed AS*: connect to more than one other AS; only carry **local traffic** (e.g., *large corporation*)
- *Transit AS*: connect to more than one other AS; carry both **transit and local traffic** (e.g., *backbone providers*)



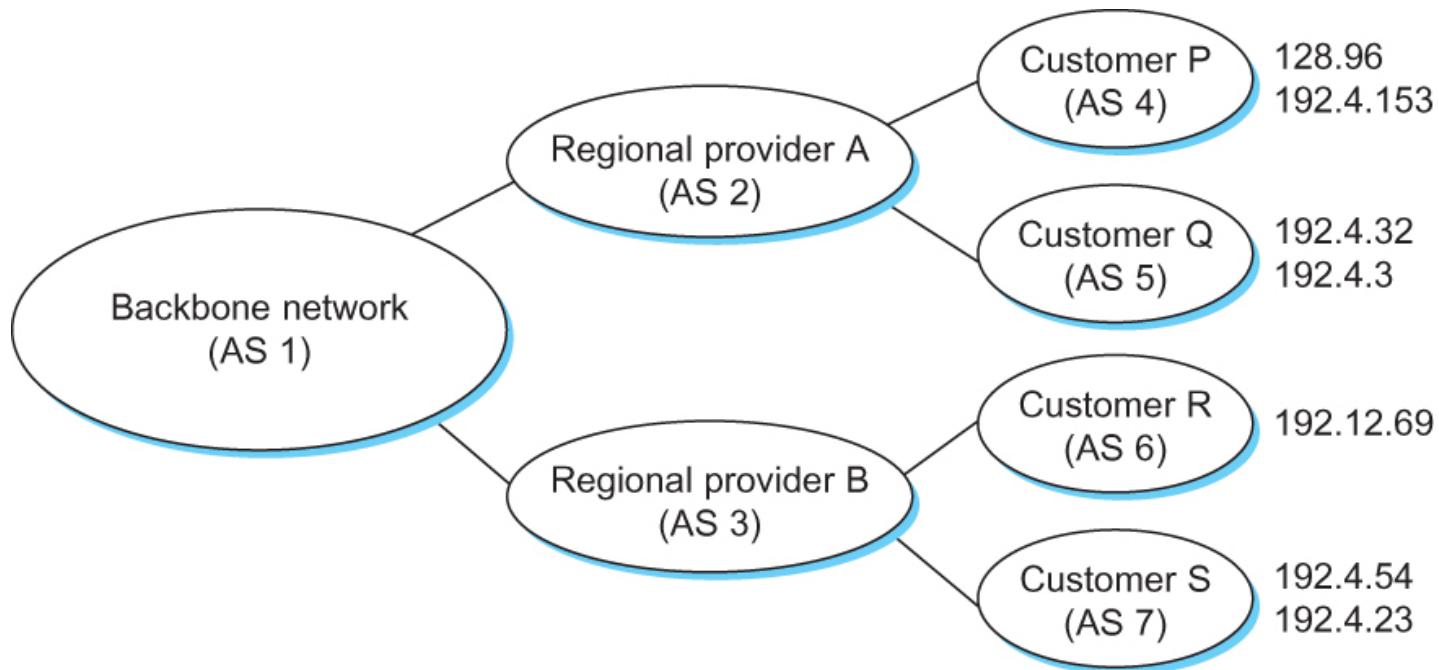
BGP: Border Gateway Protocol

Each AS has:

- One BGP *speaker* that advertises:
 - local networks
 - other reachable networks (transit AS only)
 - gives *path* information – uses a path vector
- One or more *border gateways*
 - routers through which packets enter and leave the AS

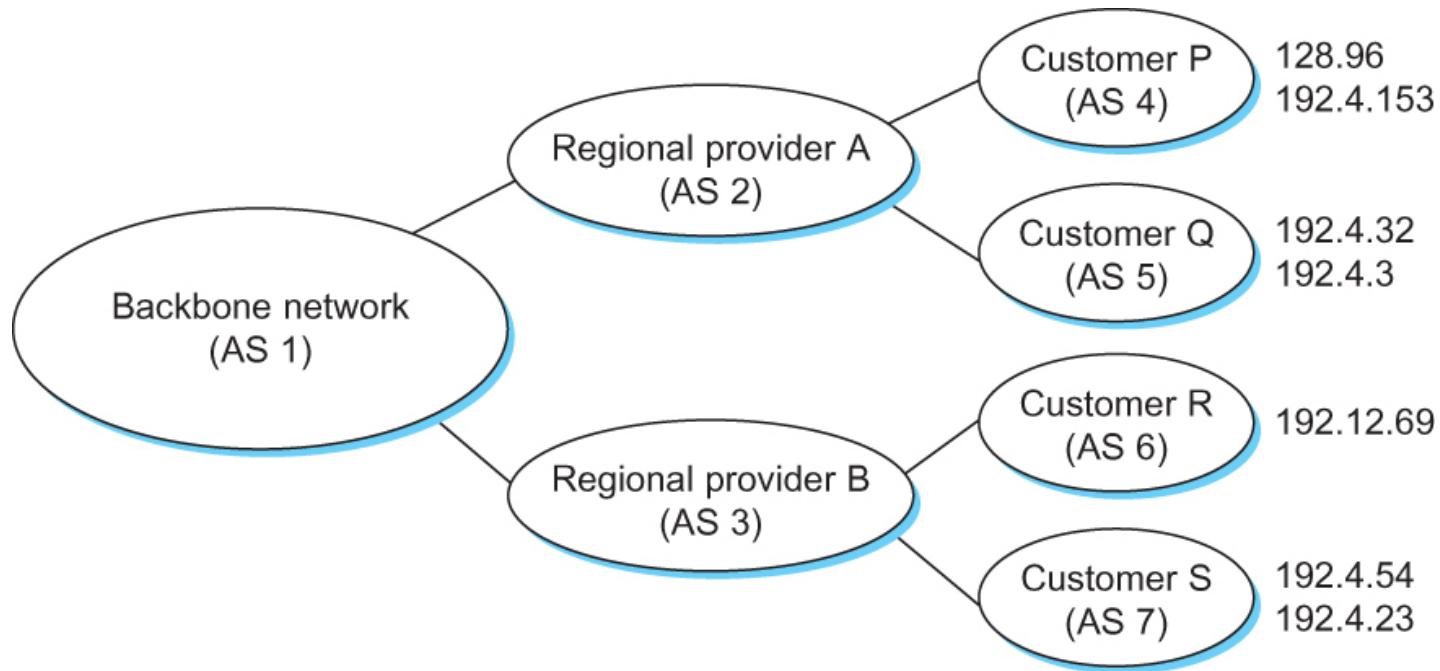
BGP: Example

- Speaker for AS 2 advertises reachability to
 - Networks 128.96, 192.4.153, 192.4.32, and 192.4.3, directly from AS 2.

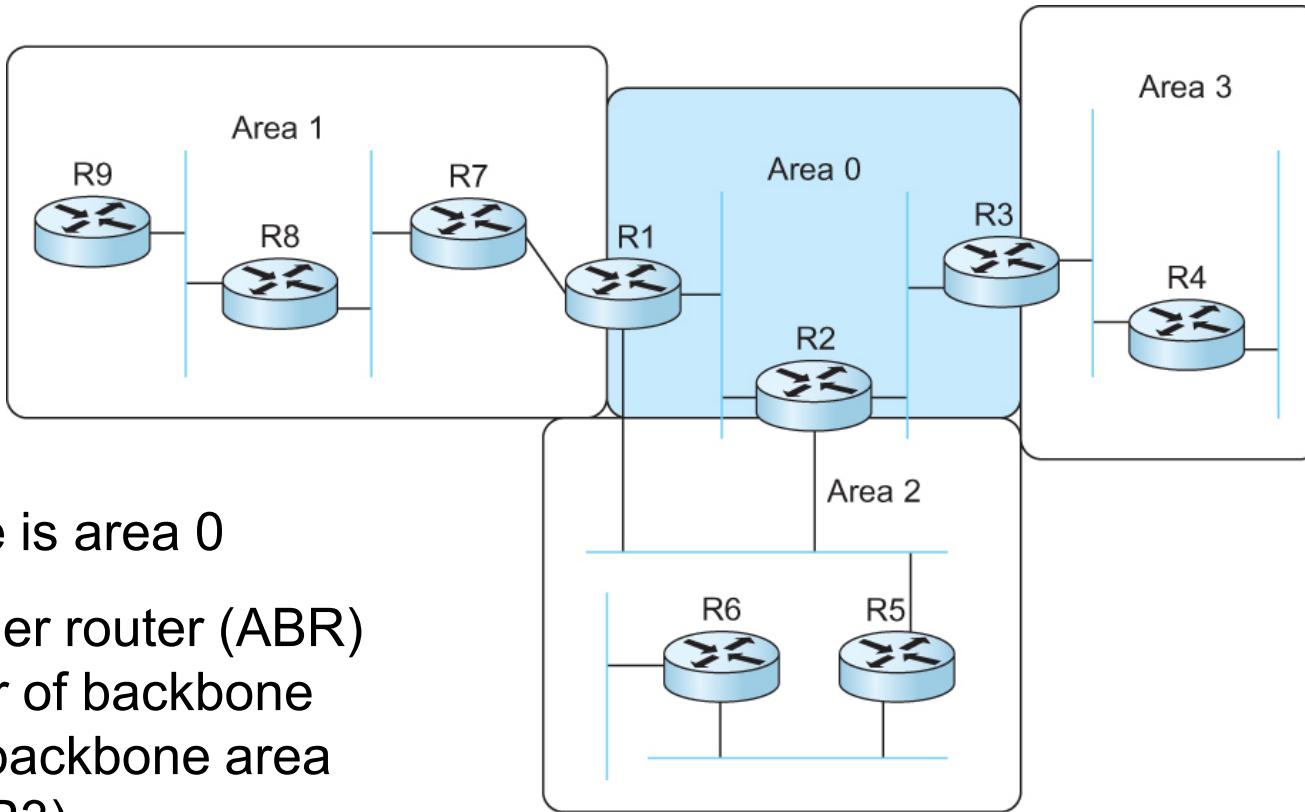


BGP: Example

- Speaker for AS 1 advertises reachability to
 - Networks 128.96, 192.4.153, 192.4.32, and 192.4.3 along the path <AS 1, AS 2>.



BGP: router area



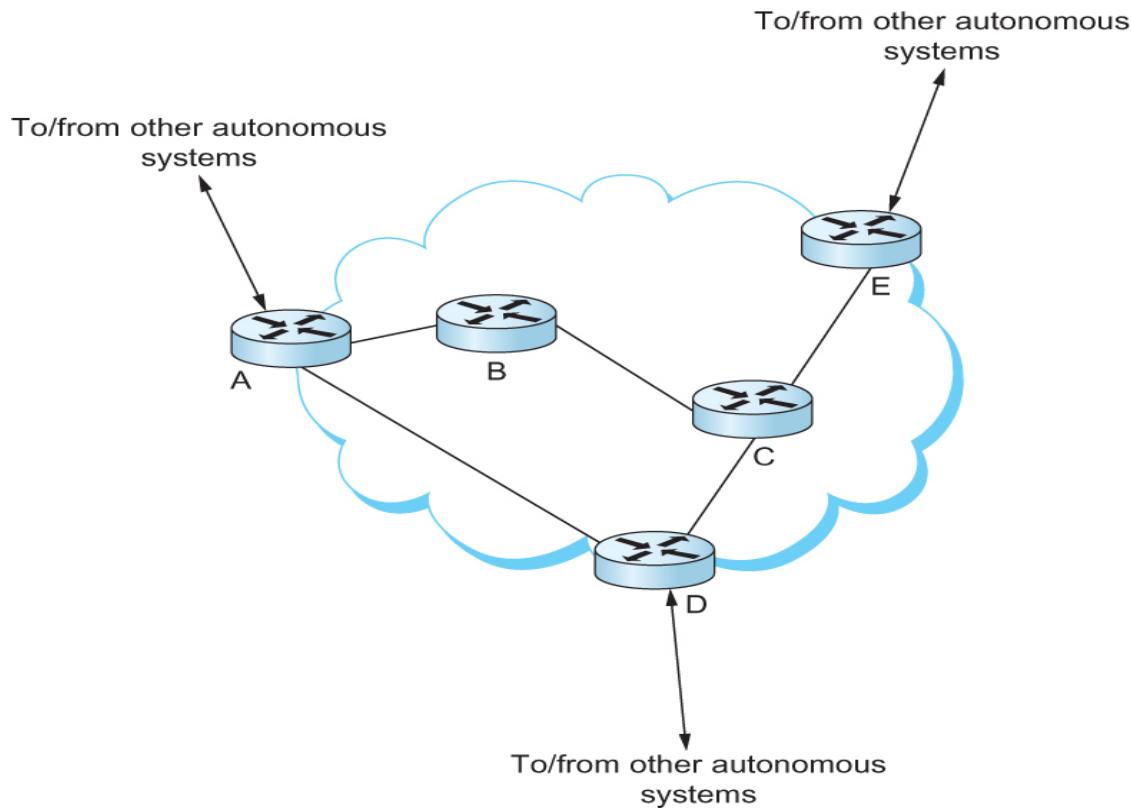
Backbone is area 0

Area border router (ABR)
– member of backbone
and non-backbone area
(R1, R2, R3)

ABR's send other areas
link-state information on
all networks in their area

A domain divided into areas

Integrating Interdomain and Intradomain Routing



All routers run iBGP (interior BGP) to border routers;
Border routers (A, D, E) also run eBGP (Exterior BGP) to other ASs.

Integrating Interdomain and Intradomain Routing

Prefix	BGP Next Hop
18.0/16	E
12.5.5/24	A
128.34/16	D
128.69./16	A

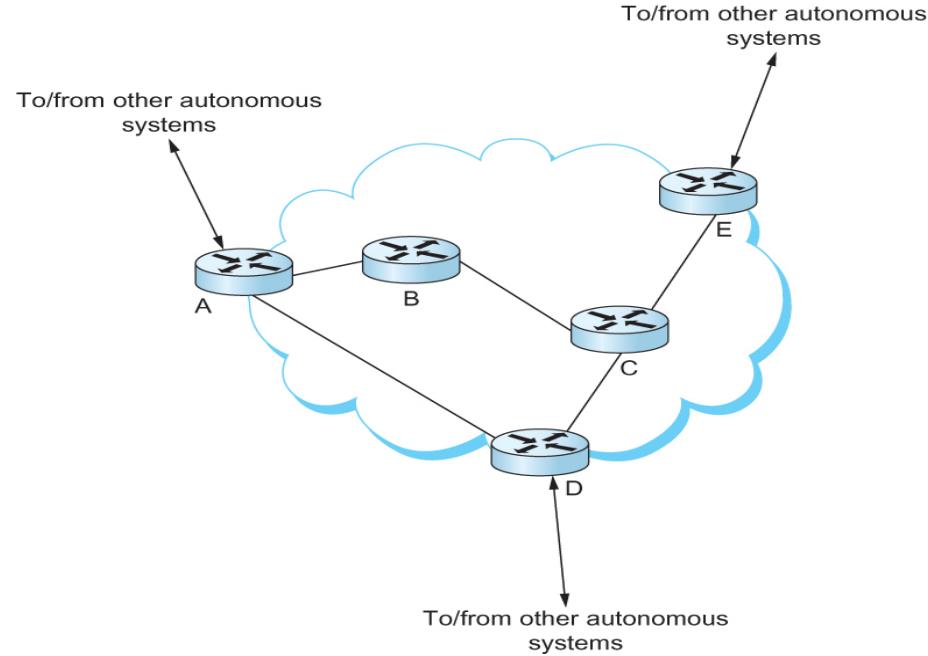
BGP table for the AS

Router	IGP Path
A	A
C	C
D	C
E	C

IGP table for router B

Prefix	IGP Path
18.0/16	C
12.5.5/24	A
128.34/16	C
128.69./16	A

Combined table for router B



BGP routing table, IGP(IBGP) routing table, and combined table at router B

Further Reading

Keycdn: <https://tools.keycdn.com/bgp-looking-glass>

Ping, DNS lookup, BGP looking glass, etc.

Cogent: <http://www.cogentco.com/en/network/looking-glass>

Internet service provider

Its served ASs, network map, BGP looking glass, etc.

CPE348: Introduction to Computer Networks

Lecture #14: Chapter 4.2



Jianqing Liu
Assistant Professor of Electrical and Computer
Engineering, University of Alabama in Huntsville
jianqing.liu@uah.edu
<http://jianqingliu.net>

Outline

- IPv6
- Multicast
- Mobile IP

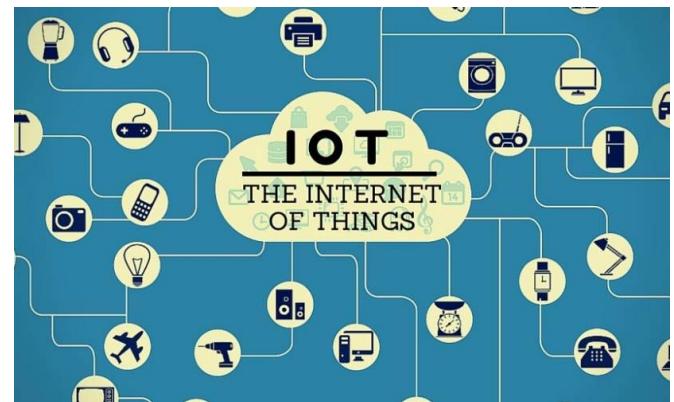
Next Generation IP (IPv6)

IPv6 Major Features

- 128-bit addresses
- Multicast
- Real-time service
- Authentication and security
- Auto-configuration – (no need for DHCP)
- Enhanced routing functionality, including support for mobile hosts

IPv6 Addresses

- Classless addressing/routing (similar to CIDR)
- Notation: x:x:x:x:x:x:x:x (x = 16-bit hex number)
 - contiguous 0s are compressed: 47CD::A456:0124
 - IPv6 compatible IPv4 address: ::FFFF:128.42.1.87
- Address assignment
 - provider-based
 - geographic



Internet Multicast (Multicast on IP)

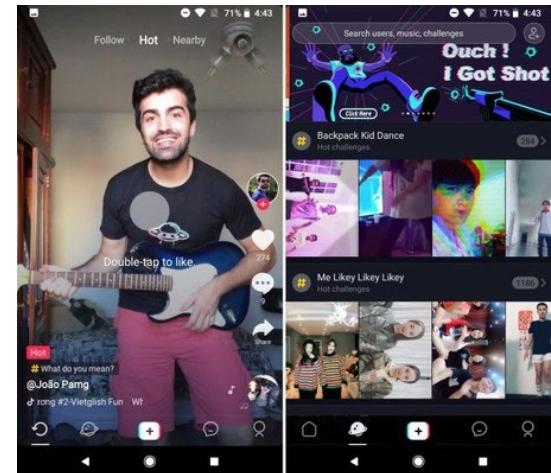
What is the difference between unicast,
multicast and broadcast?



IP Multicast Overview

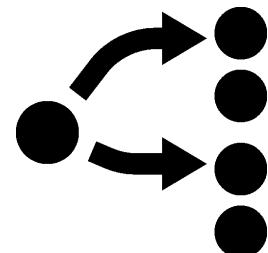
- One-to-many
 - Radio station broadcast
 - Transmitting news, stock-price
 - Software updates to multiple hosts

- Many-to-many
 - Multimedia teleconferencing
 - Online multi-player games
 - Distributed simulations



IP Multicast Overview

- If no support of multicast
 - A source sends a separate packet to each destination
 - More bandwidth
 - Redundant traffic, especially concentrated near the source
 - The source keeps (IP) addresses of all destinations
 - Storage overhead
 - Scalability



IP Multicast Overview – Facts

- Members of the group could be anywhere in the Internet;
- Members join and leave the group;
- A member could be in different groups;
- Senders and receivers could be distinct: a sender need not be a group member.

IP Multicast Overview – Design Objectives

- Source:
 - A host only transmits a **single copy** of data to the multicast group
- Address:
 - Use a **group address** instead of many unicast addresses
- Group Management:
 - Maintain **dynamic members** in a multicast group

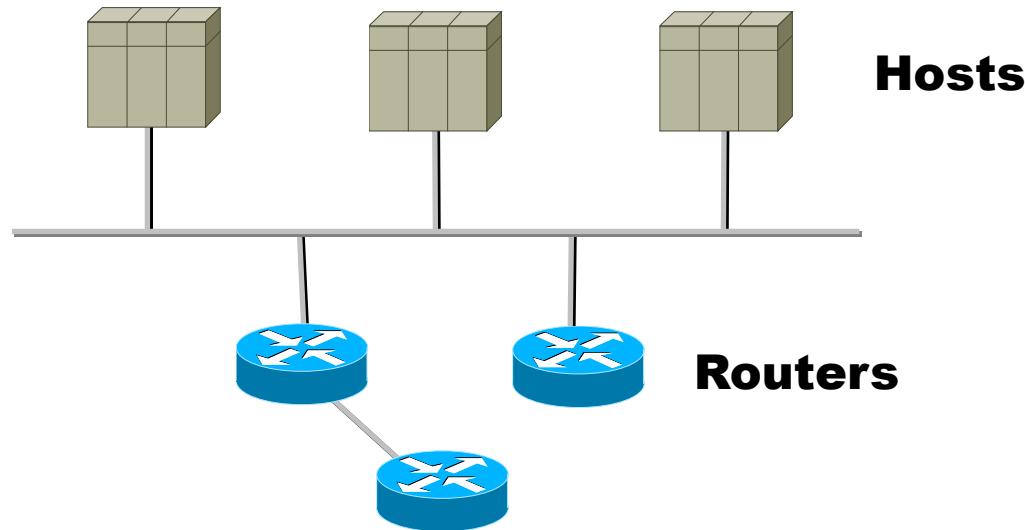
What is the design to fulfill such goals?

IP Multicast Overview

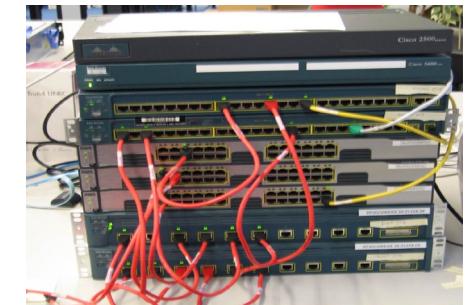
The edge router is the field of design!

Router here is NOT the WiFi router/access point!

Host-to-Router Protocols (IGMP)



Multicast Routing Protocols (PIM)

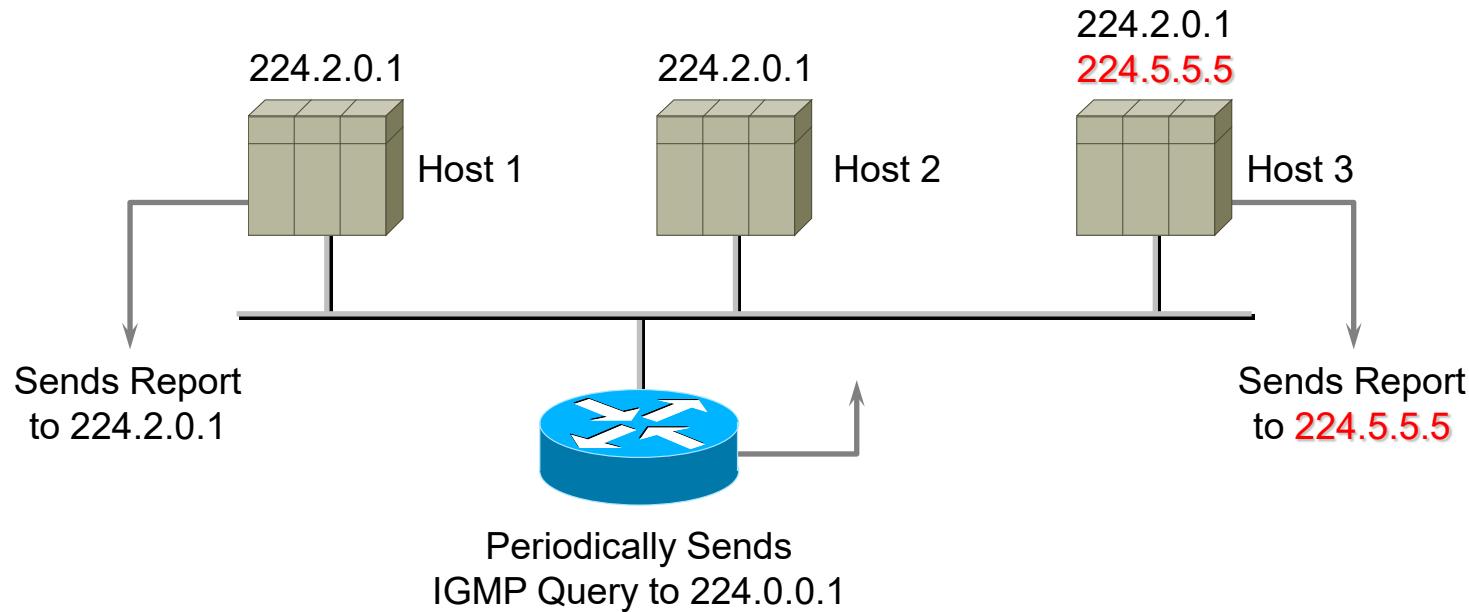


Internet Group Management Protocol (IGMP)

- IP multicast protocol used in IPv4
 - IPv6 has a similar one: [Multicast Listener Discovery \(MLD\)](#) protocol
- IGMP evolves from [v1](#) to [v3](#) for better efficiency (e.g., redundant message, latency)

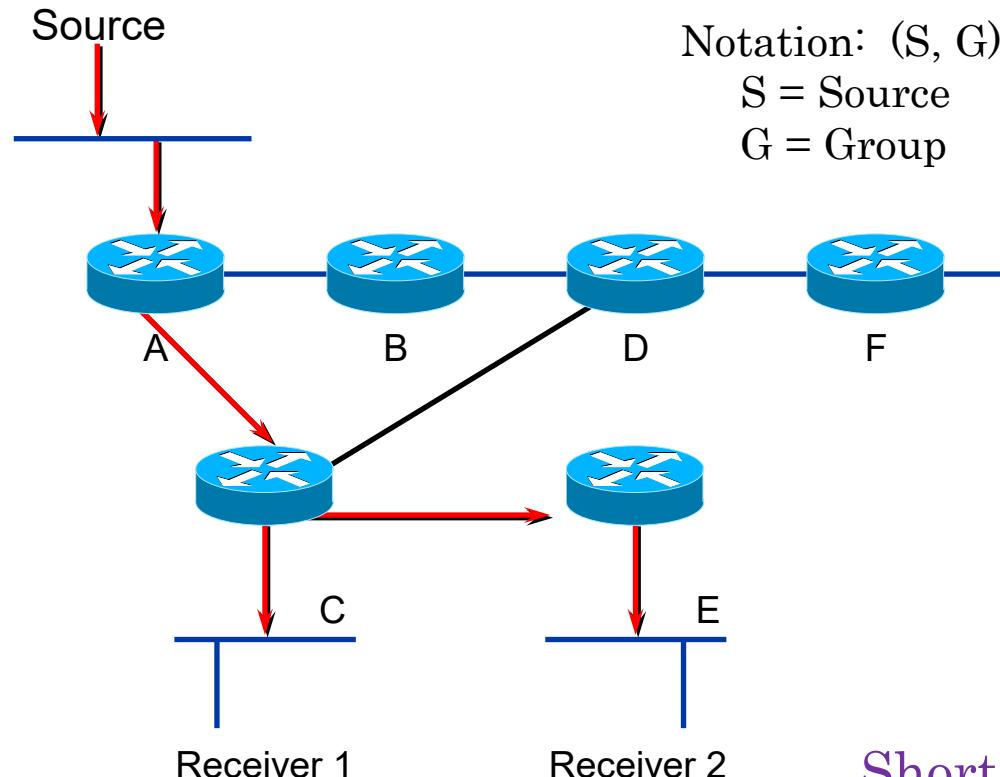
Internet Group Management Protocol (IGMP)

■ IGMP v1 Queries & Reports



Multicast Routing Protocols

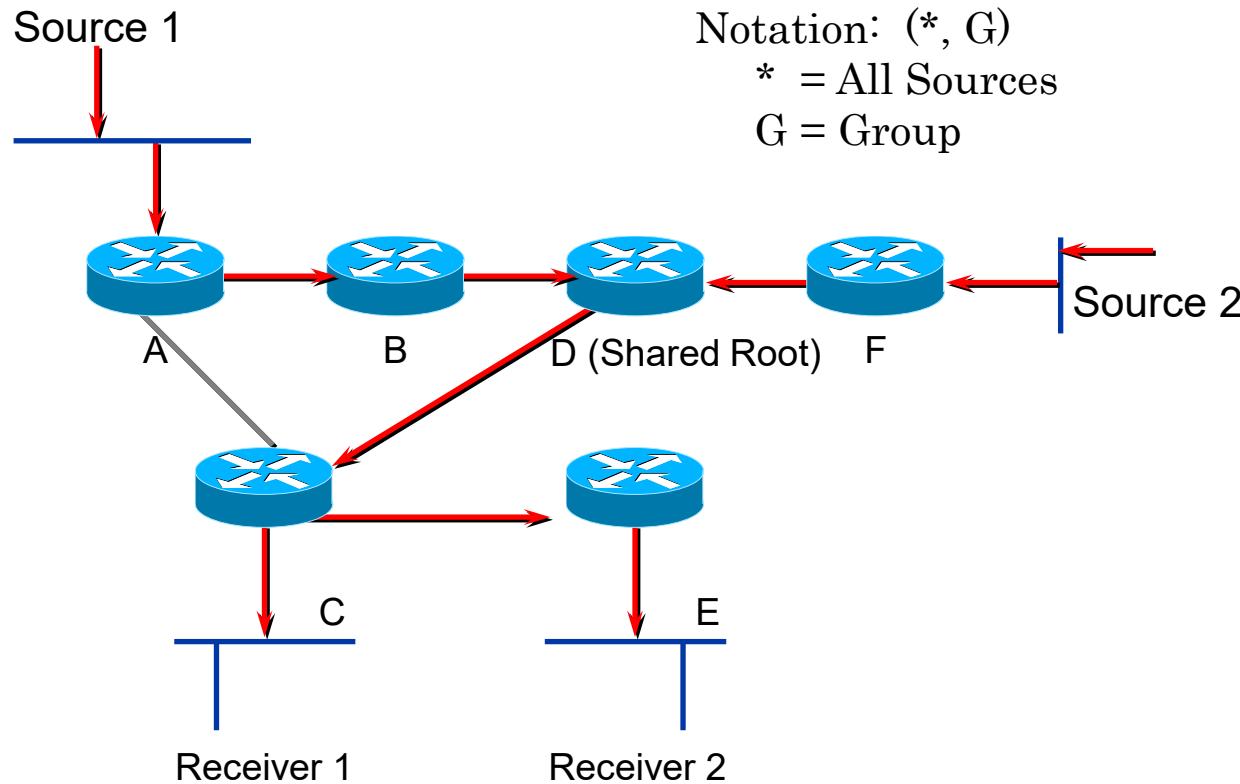
- **Goal:** forwarding multicast packets between routers



Shortest path or source distribution tree

Multicast Routing Protocols

- Shared distribution tree (multiple sources)



Multicast Routing Protocols – analysis

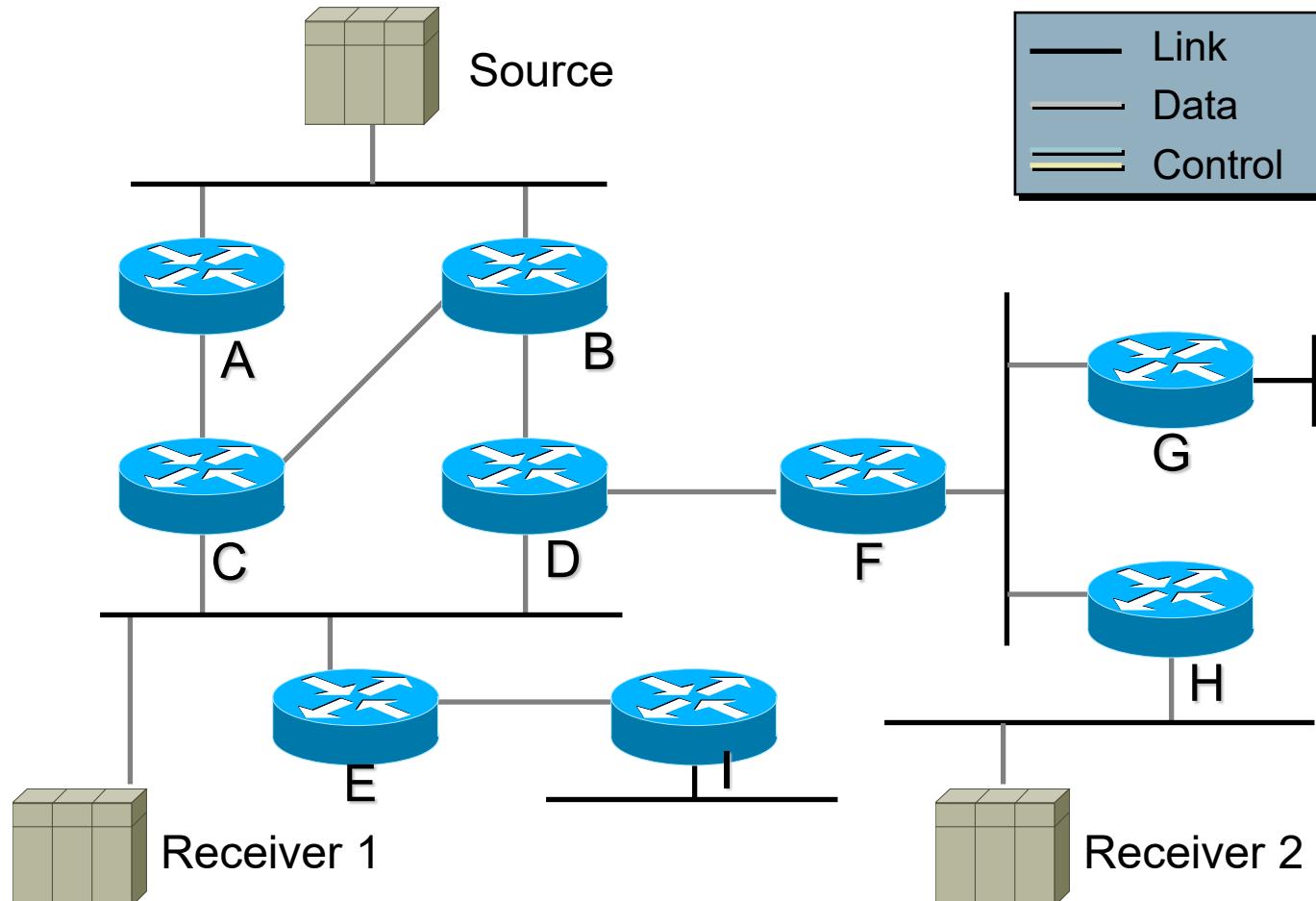
- Distribution Trees
 - Source distribution tree
 - Uses **more memory** $O(S \times G)$ but you get **optimal** paths from source to all receivers, **minimizes delay**
 - Shared distribution tree
 - Uses **less memory** $O(G)$ but you may get **suboptimal** paths from source to all receivers, may **introduce delay**
- Protocols
 - Protocol Independent Multicast (PIM)
 - Distance Vector Multicast Routing Protocol (DVMRP)

Multicast Routing Protocols – PIM

- PIM: Protocol Independent Multicast
 - Dense Mode
 - Broadcast and prune
 - Assumes dense group membership
 - Source initiated
 - Sparse Mode
 - Explicit join
 - Sparsely distributed group membership
 - Destination initiated

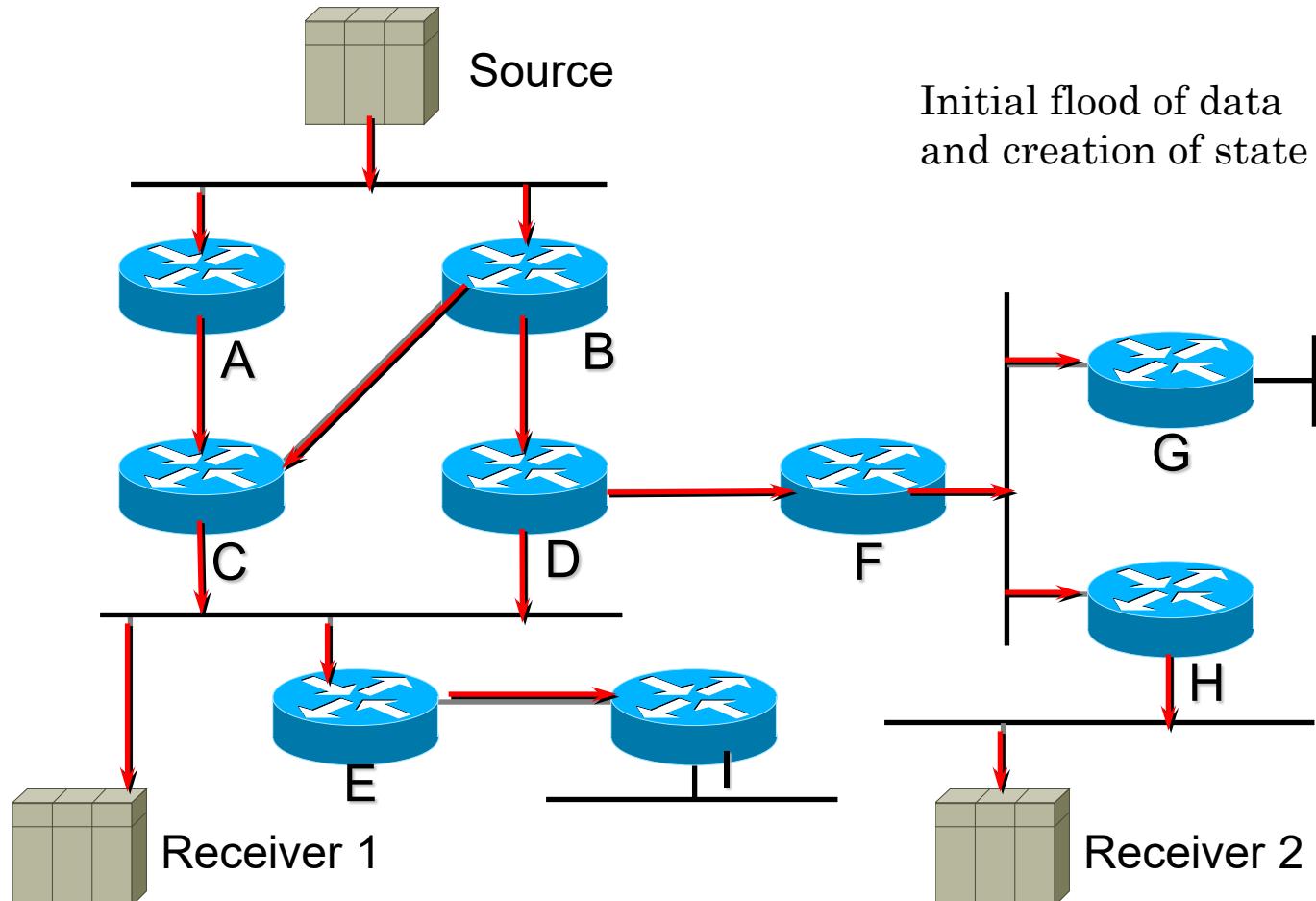
Multicast Routing Protocols – PIM

- PIM – Dense Mode



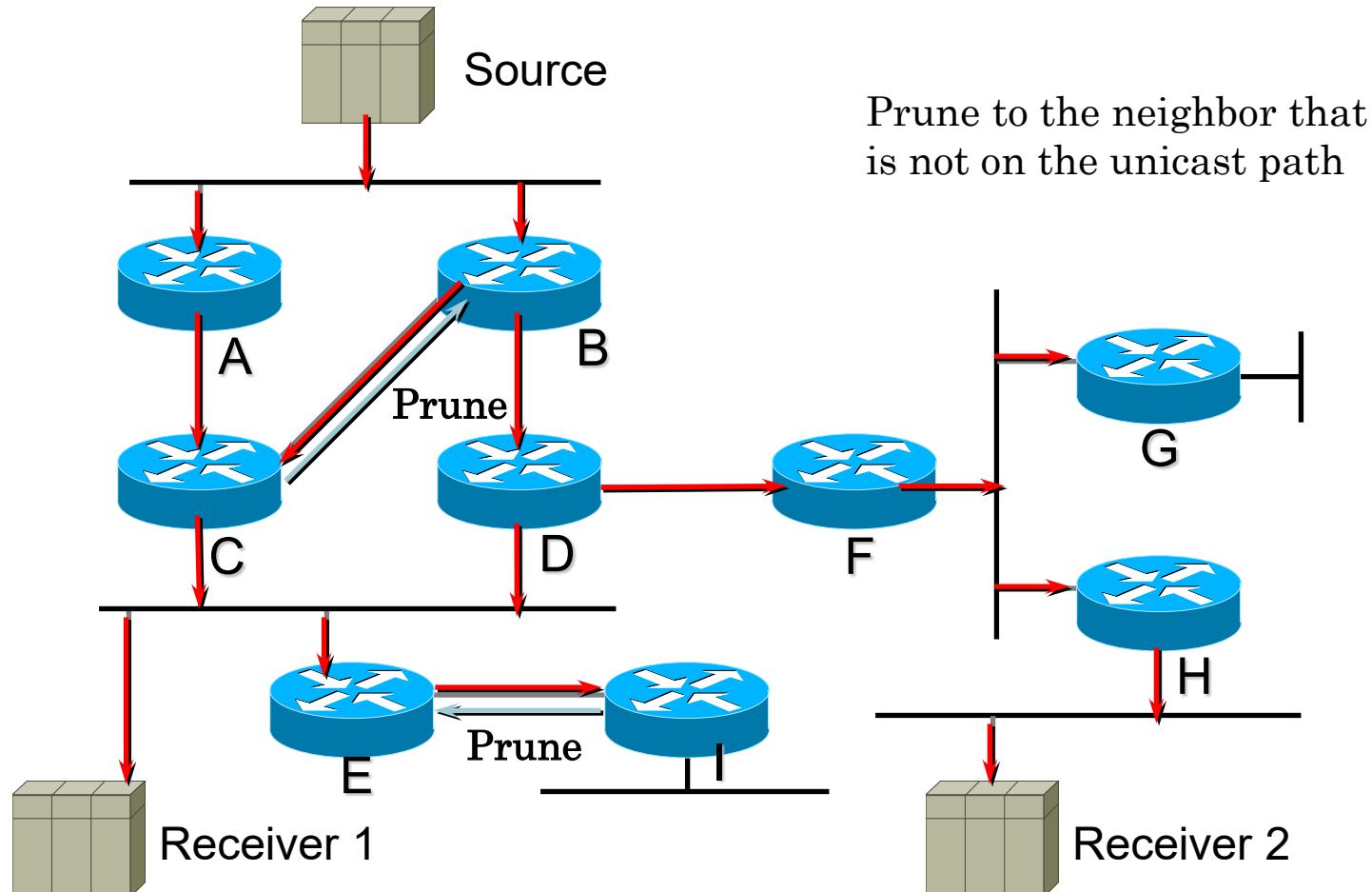
Multicast Routing Protocols – PIM

- PIM – Dense Mode



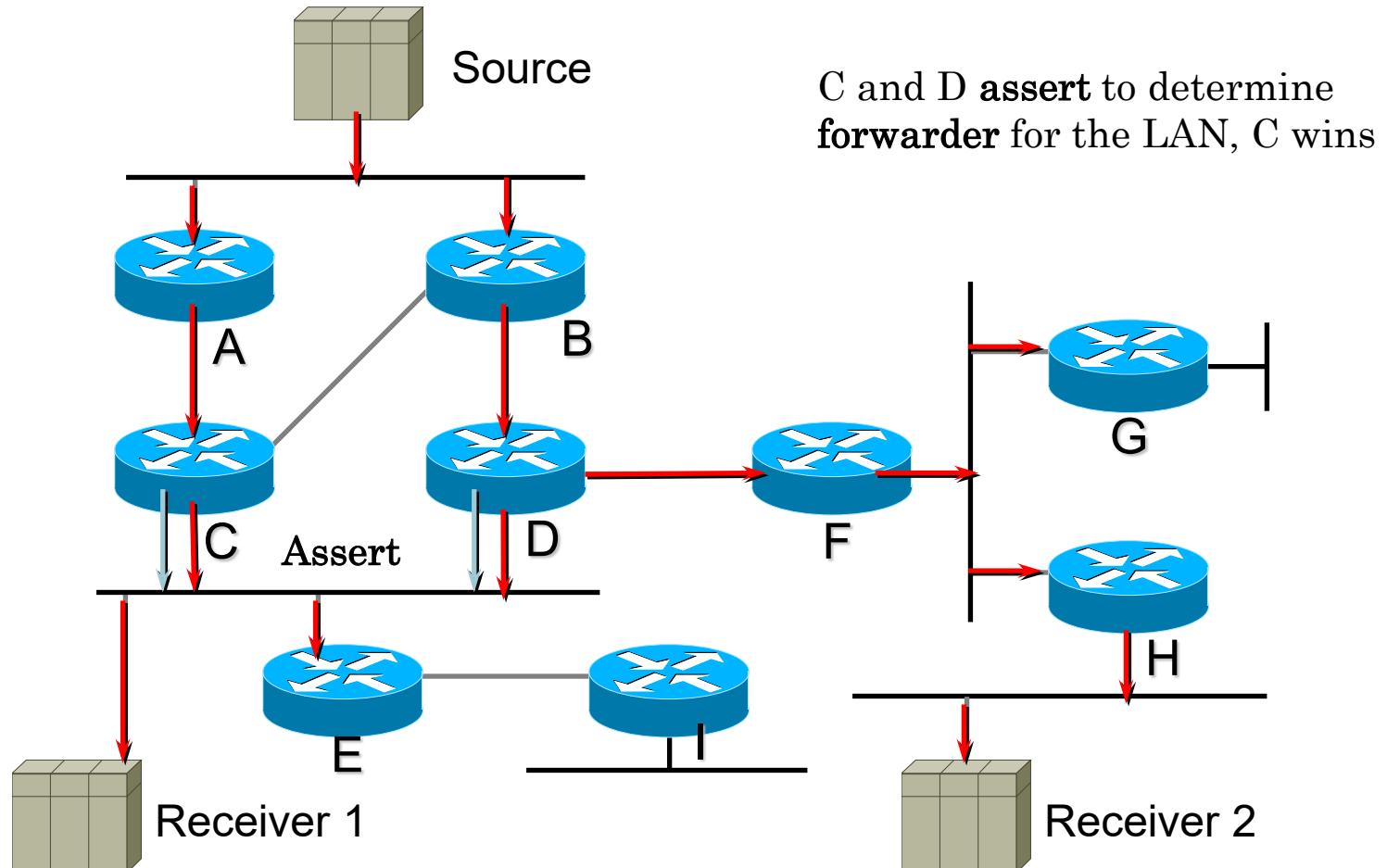
Multicast Routing Protocols – PIM

- PIM – Dense Mode



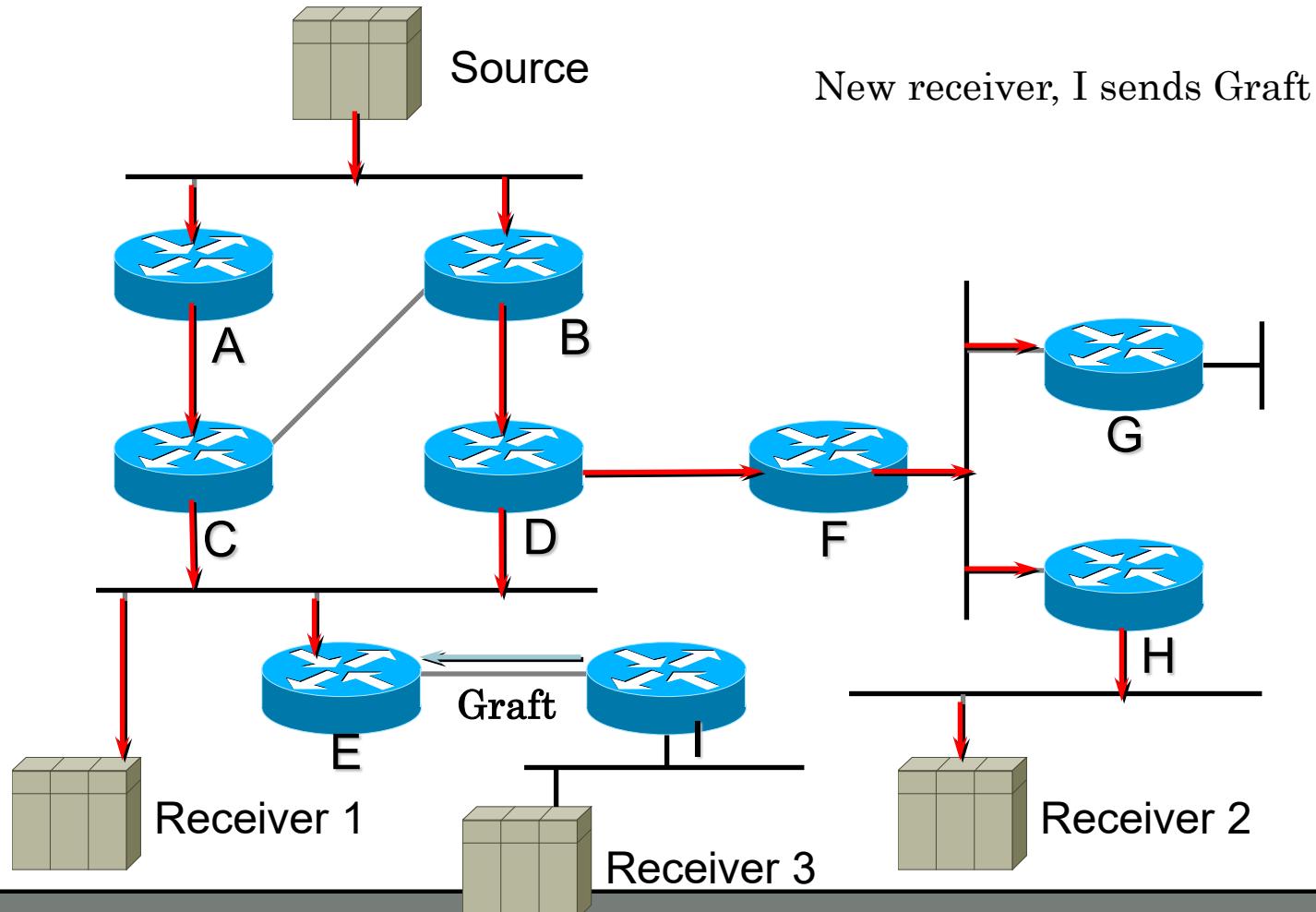
Multicast Routing Protocols – PIM

- PIM – Dense Mode



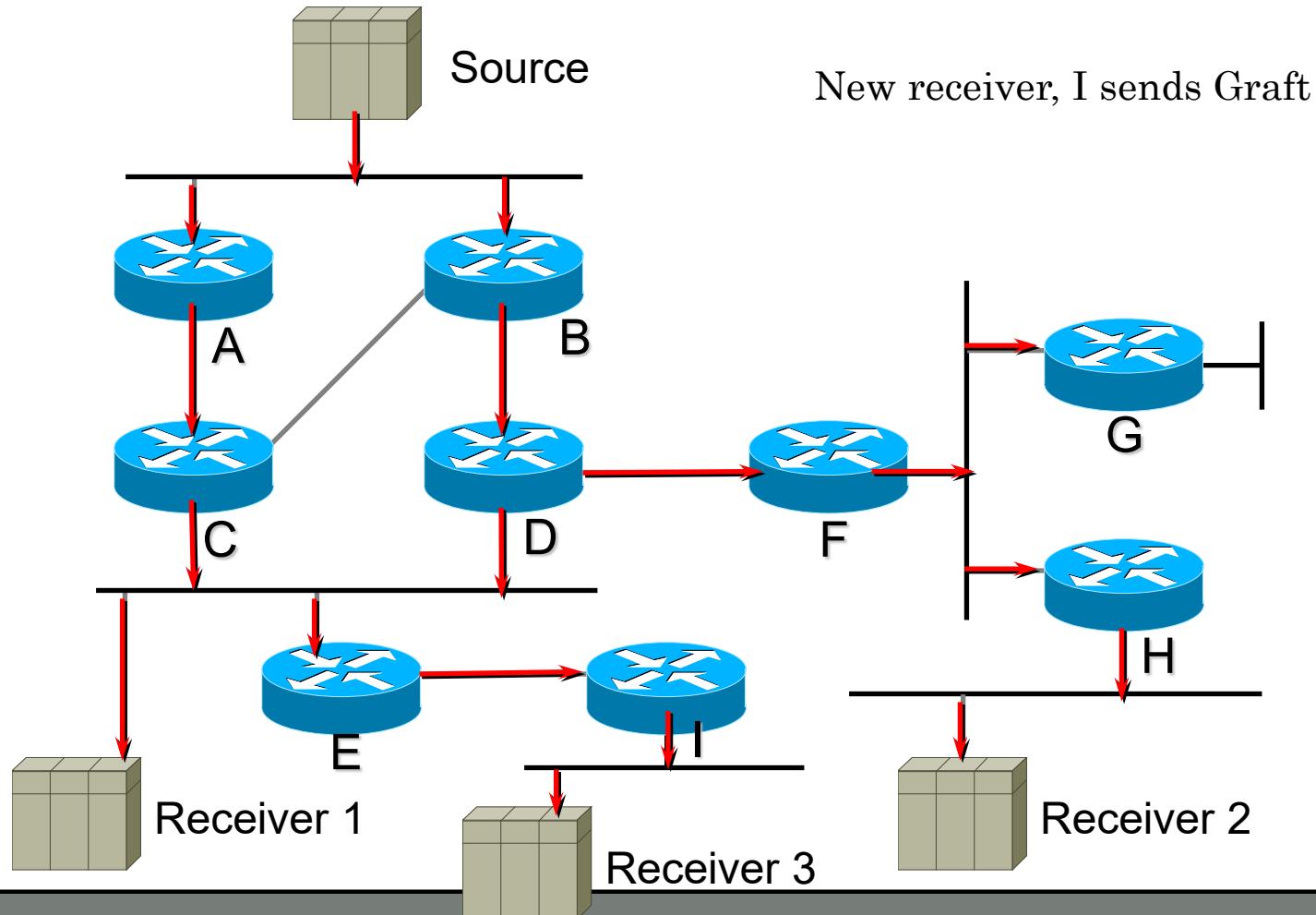
Multicast Routing Protocols – PIM

- PIM – Dense Mode



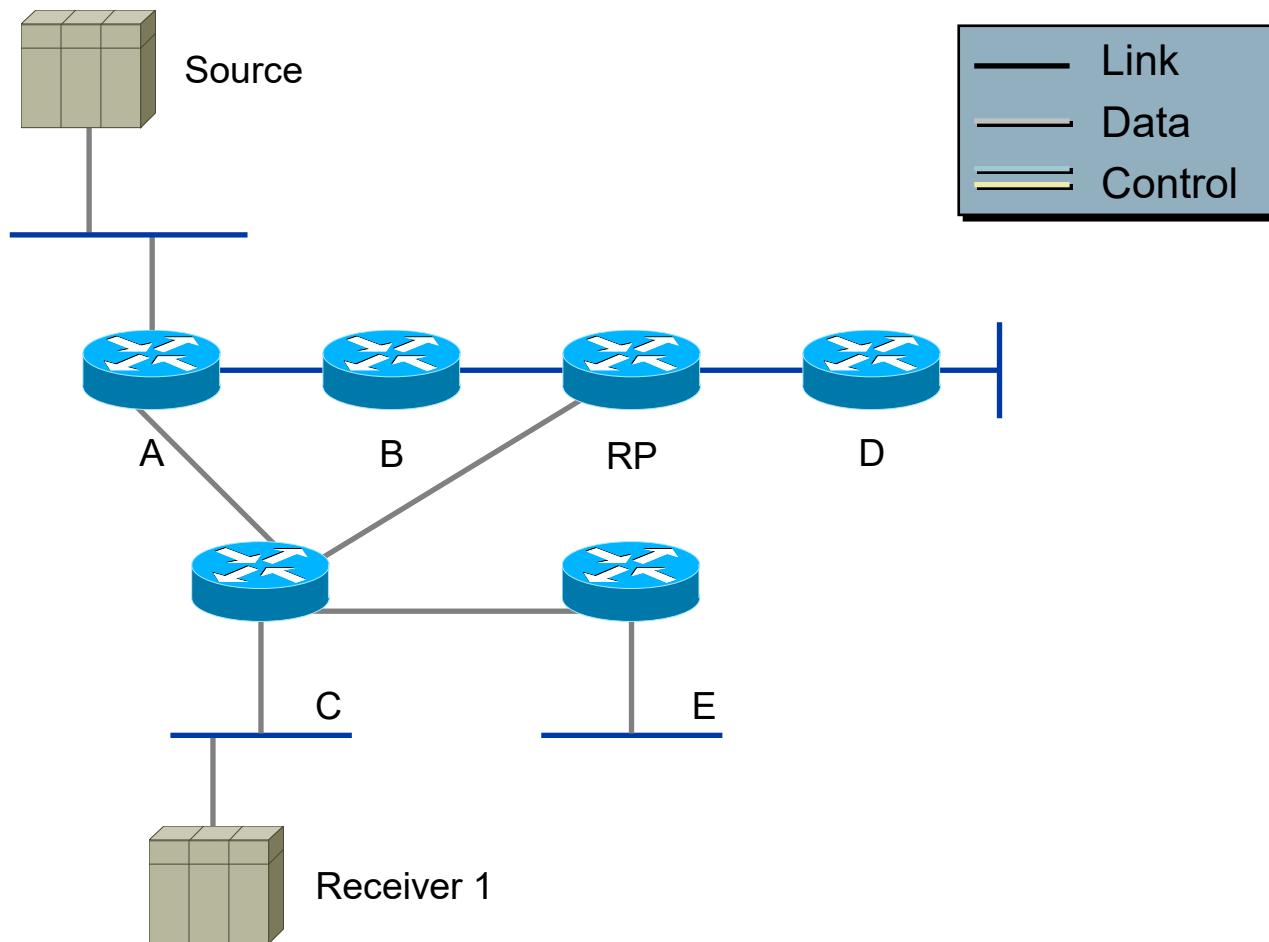
Multicast Routing Protocols – PIM

- PIM – Dense Mode



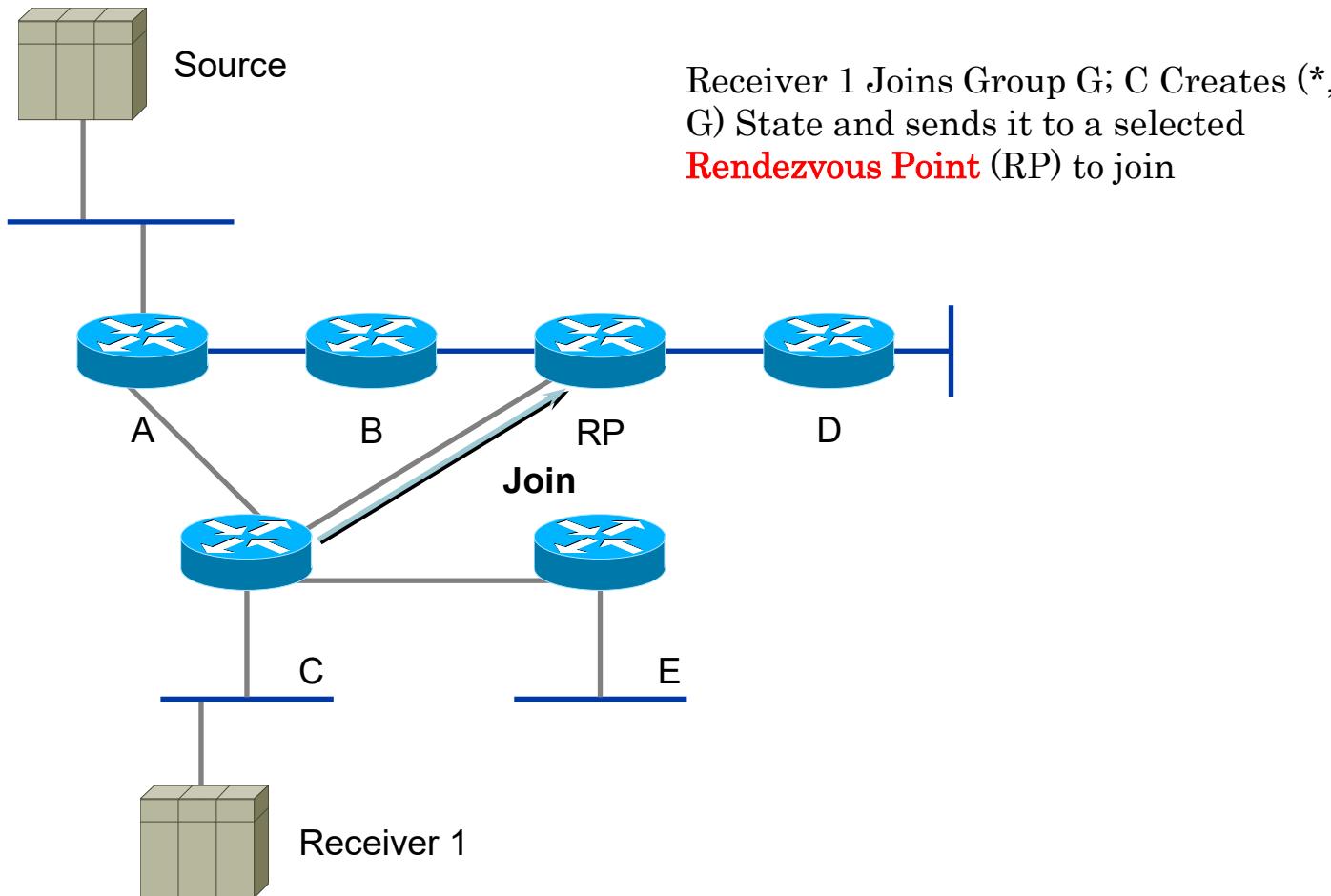
Multicast Routing Protocols – PIM

- PIM – Sparse Mode



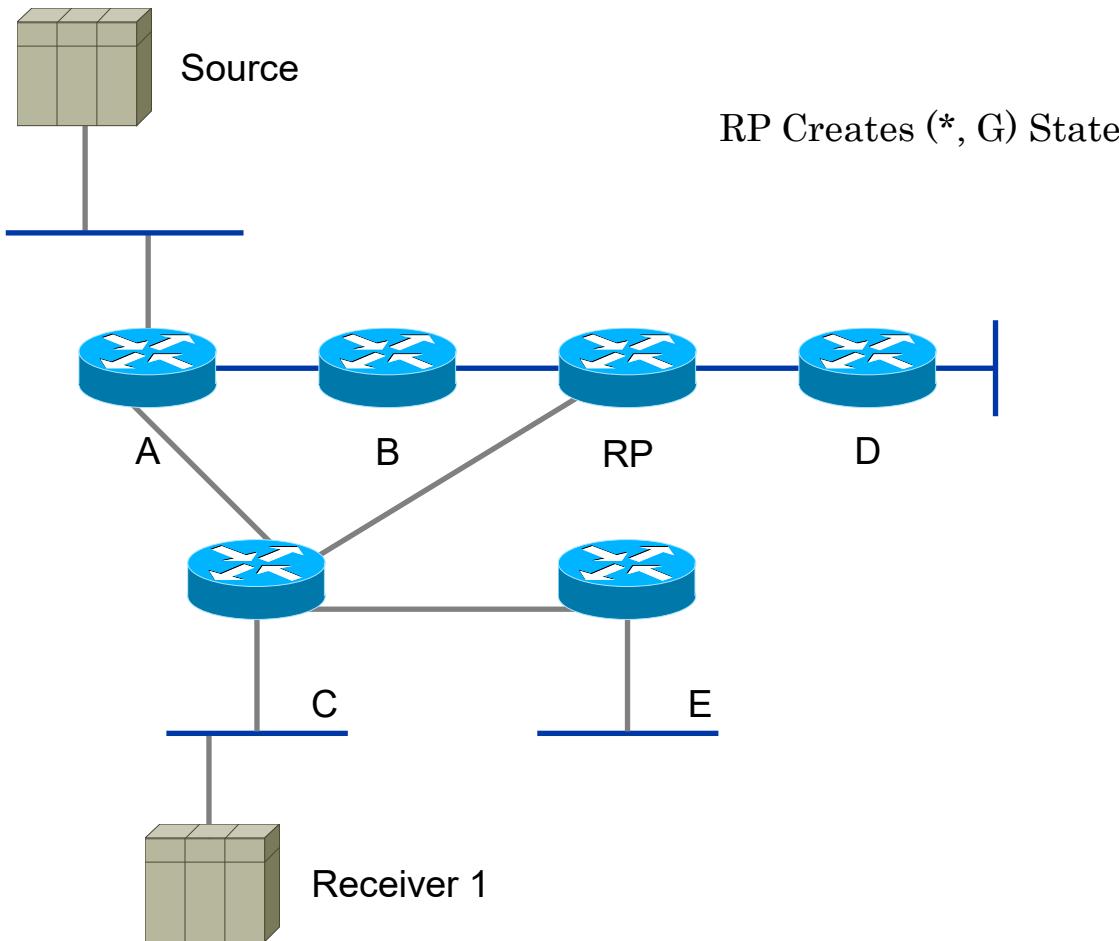
Multicast Routing Protocols – PIM

■ PIM – Sparse Mode



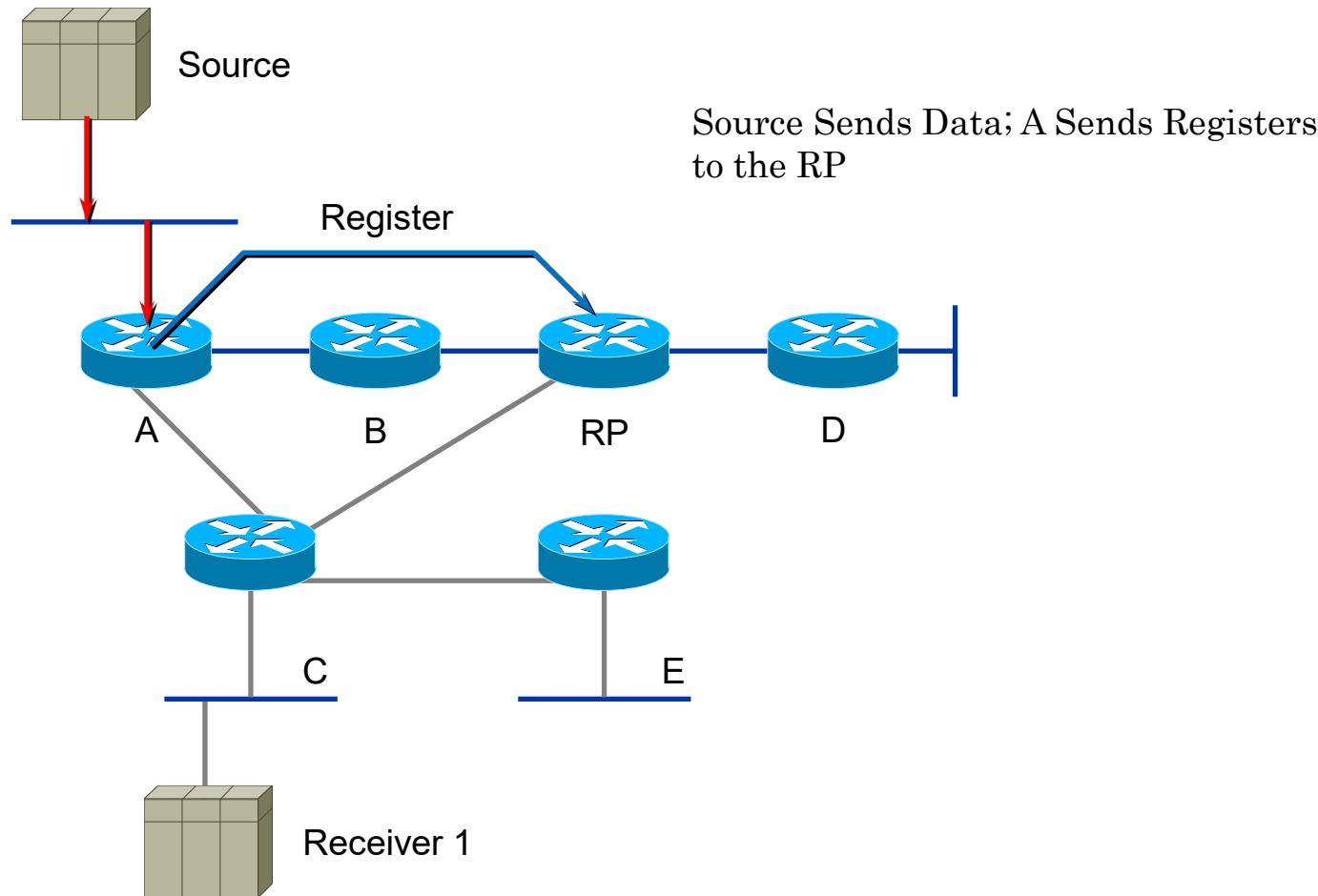
Multicast Routing Protocols – PIM

- PIM – Sparse Mode



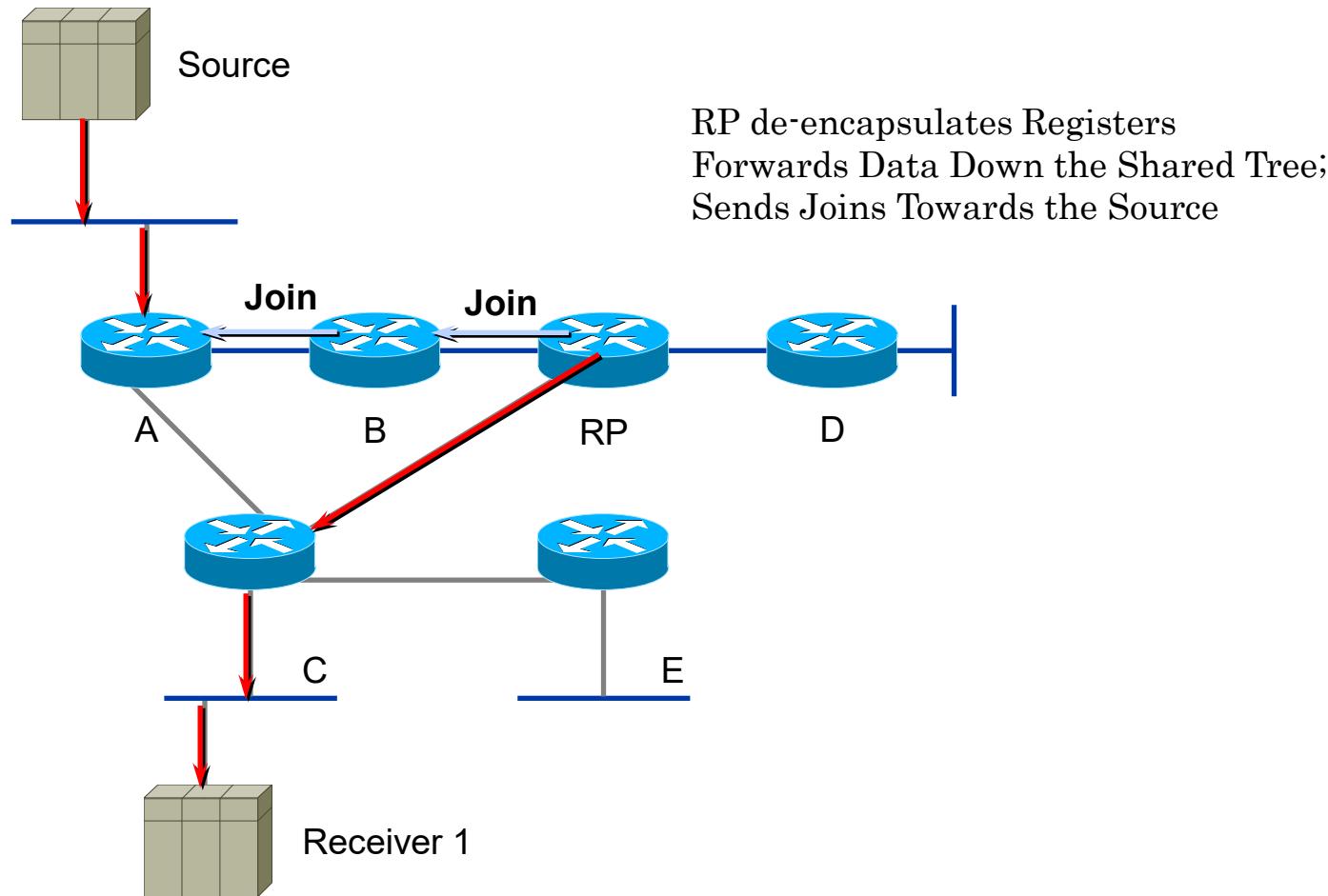
Multicast Routing Protocols – PIM

■ PIM – Sparse Mode



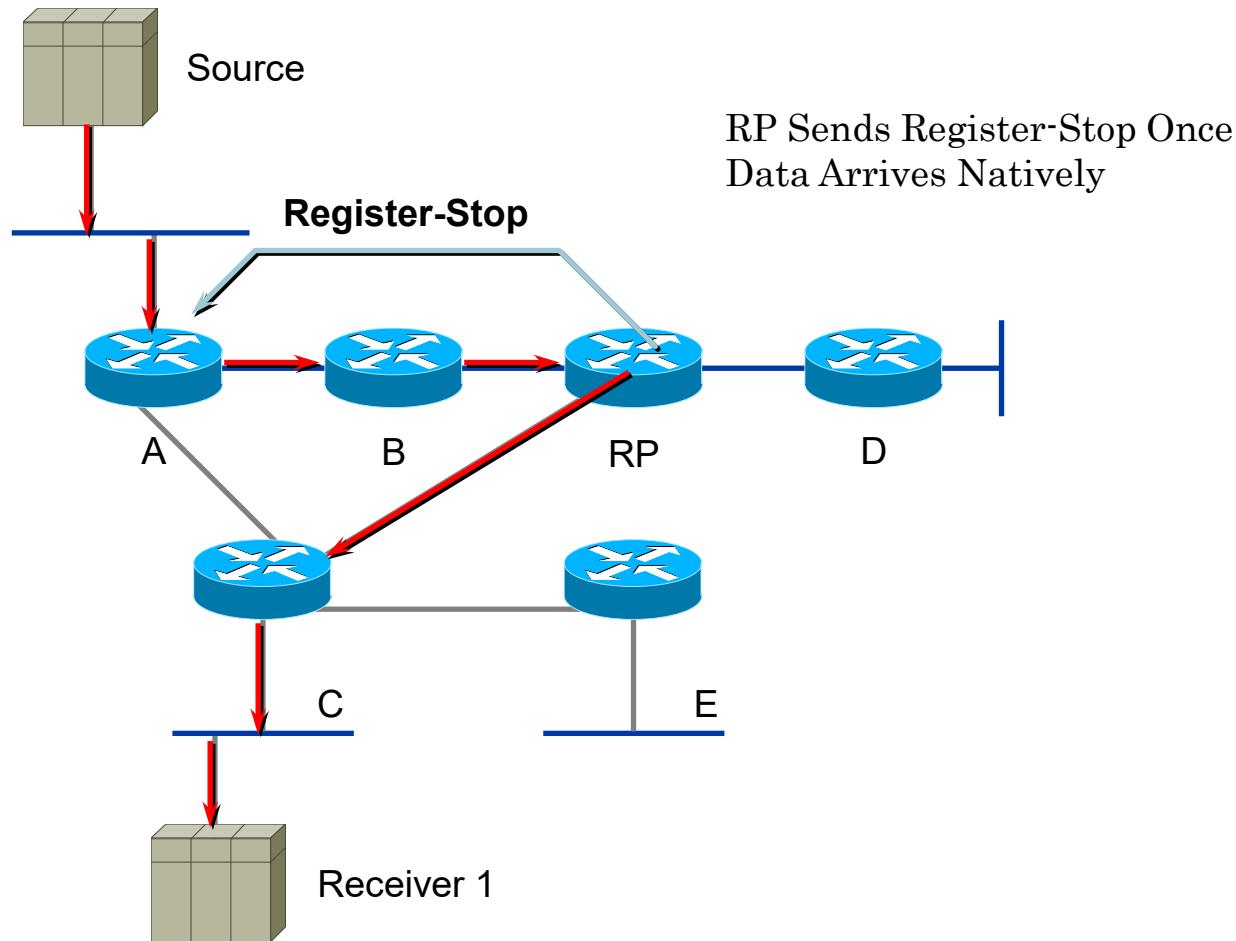
Multicast Routing Protocols – PIM

■ PIM – Sparse Mode



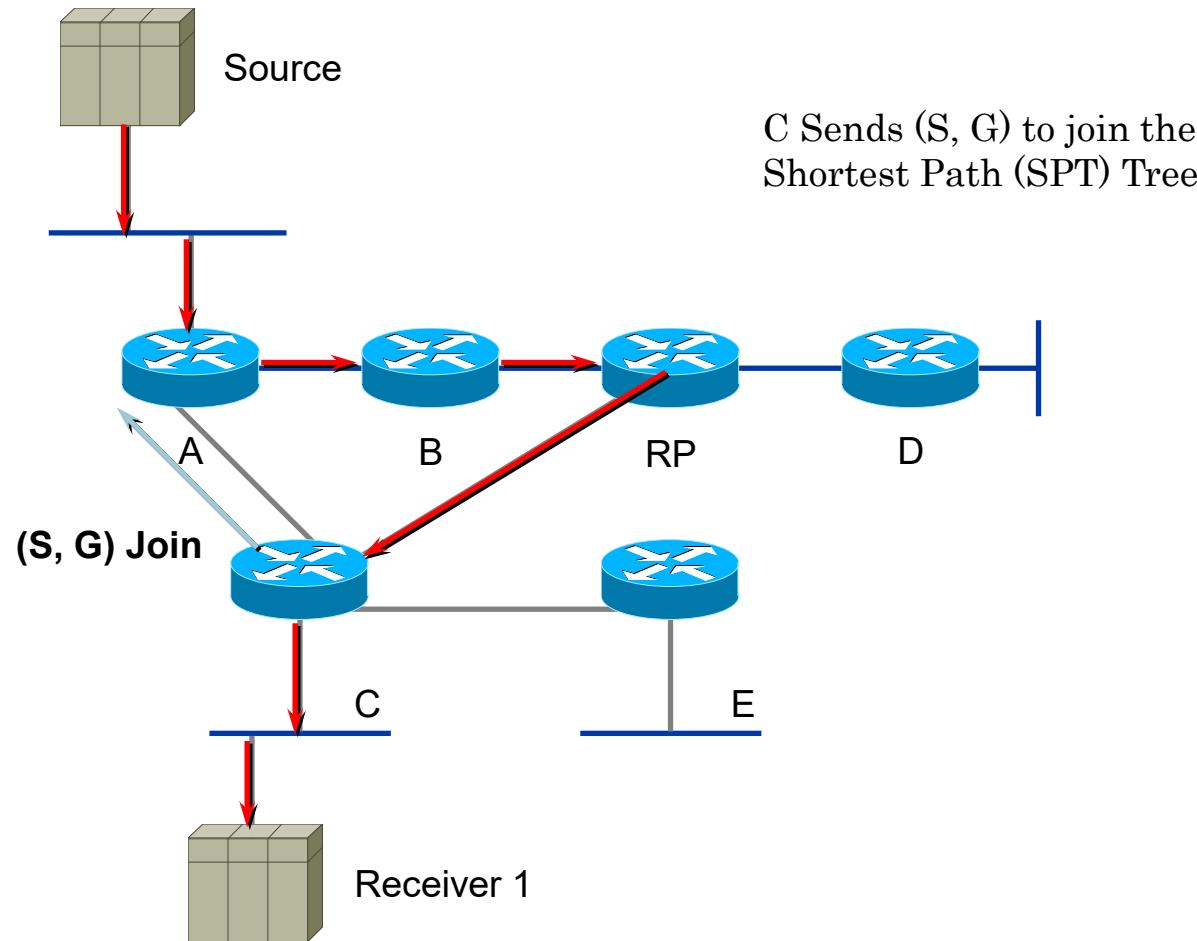
Multicast Routing Protocols – PIM

- PIM – Sparse Mode



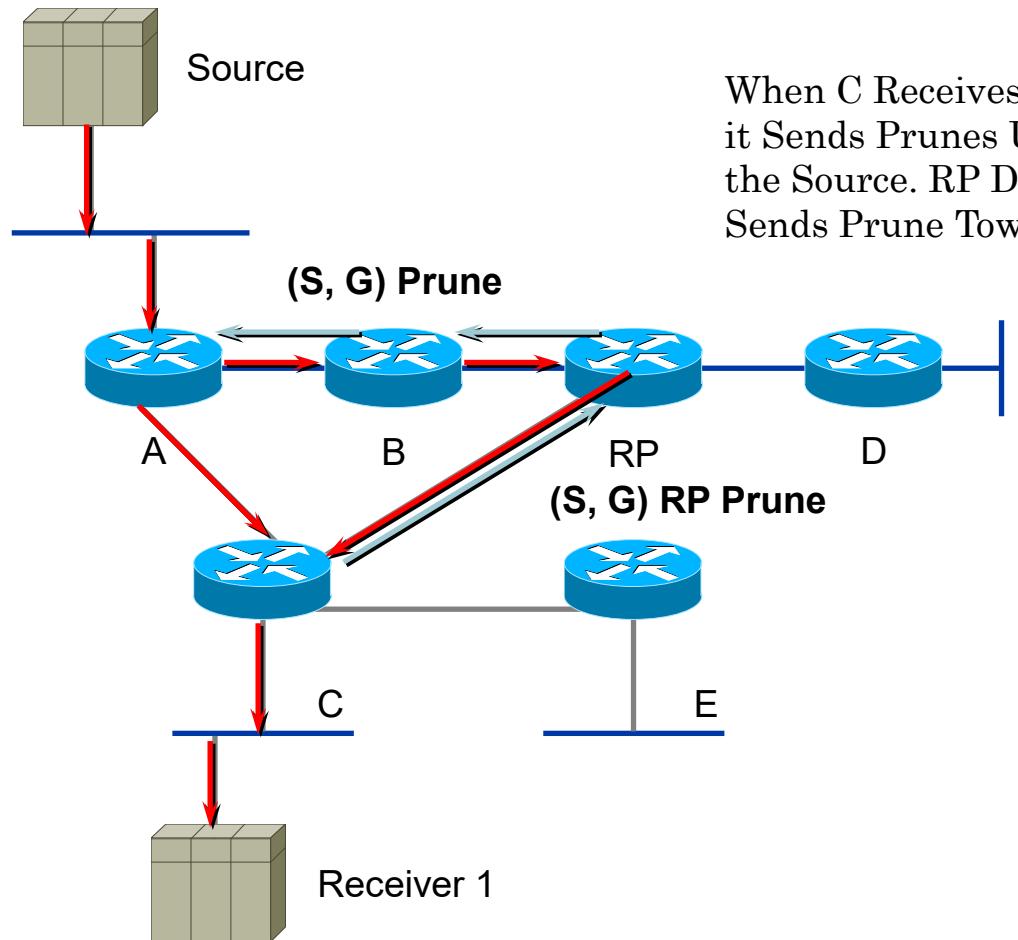
Multicast Routing Protocols – PIM

■ PIM – Sparse Mode



Multicast Routing Protocols – PIM

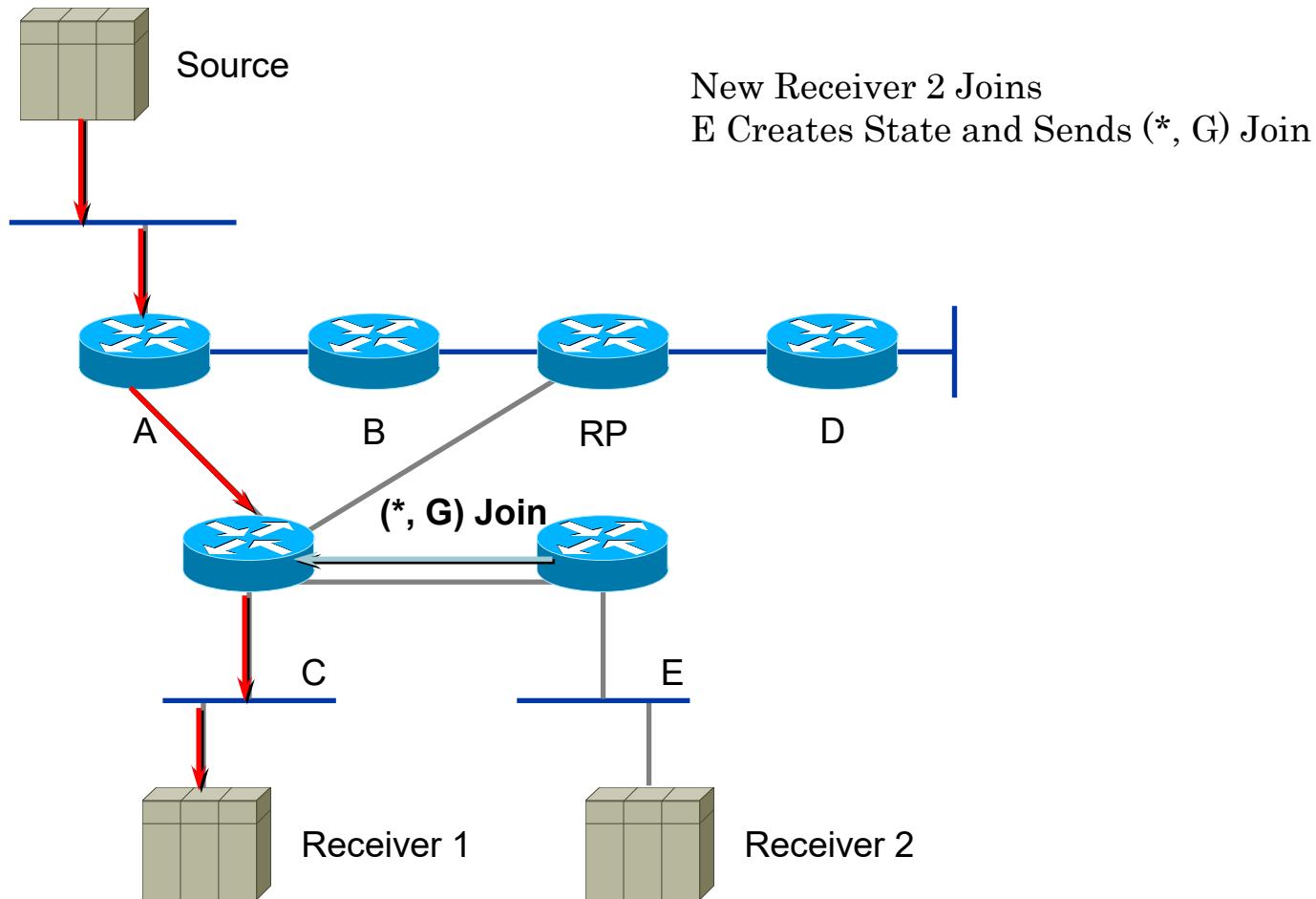
■ PIM – Sparse Mode



When C Receives Data Natively, it Sends Prunes Up the RP tree for the Source. RP Deletes (S, G) and Sends Prune Towards the Source

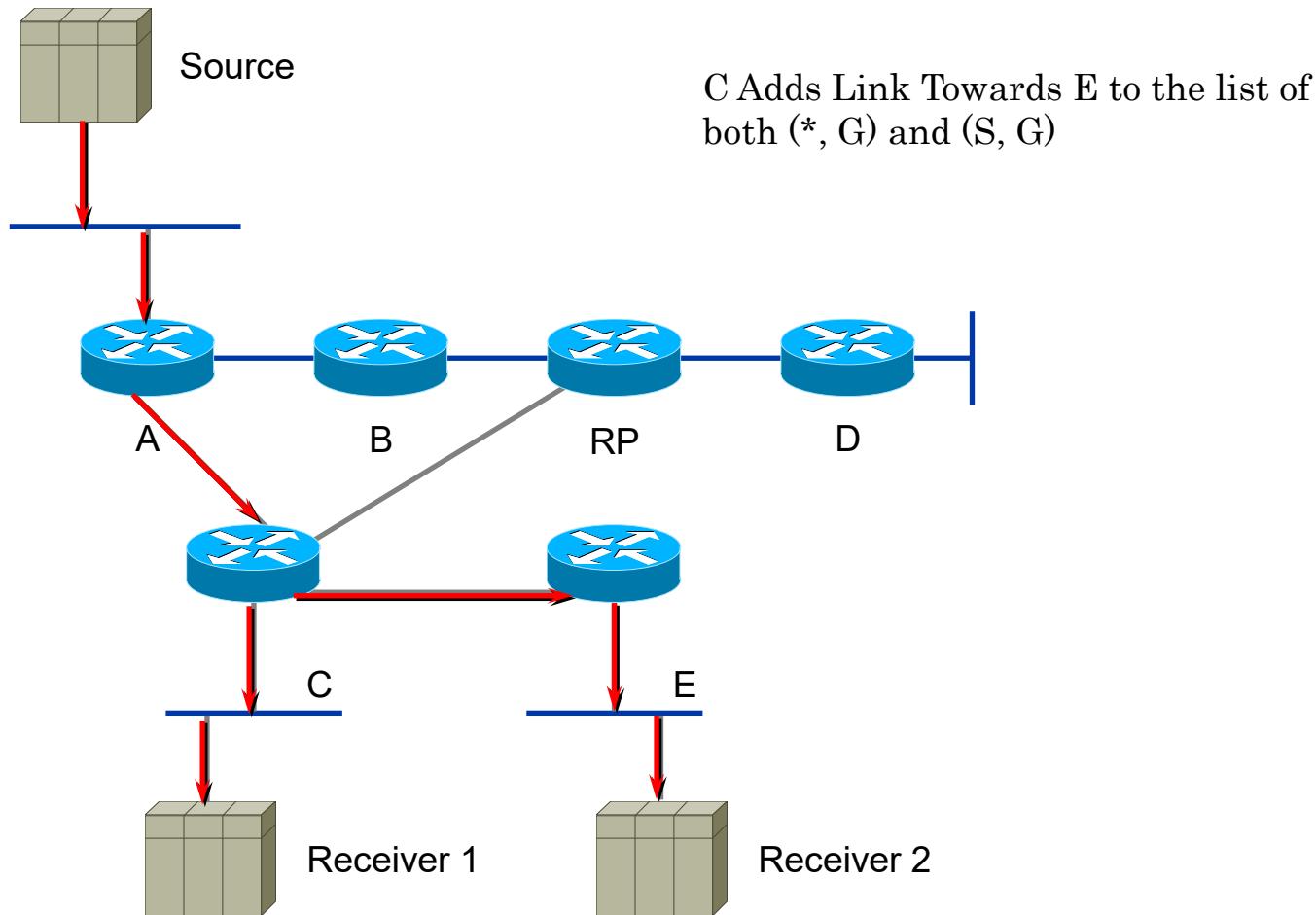
Multicast Routing Protocols – PIM

■ PIM – Sparse Mode



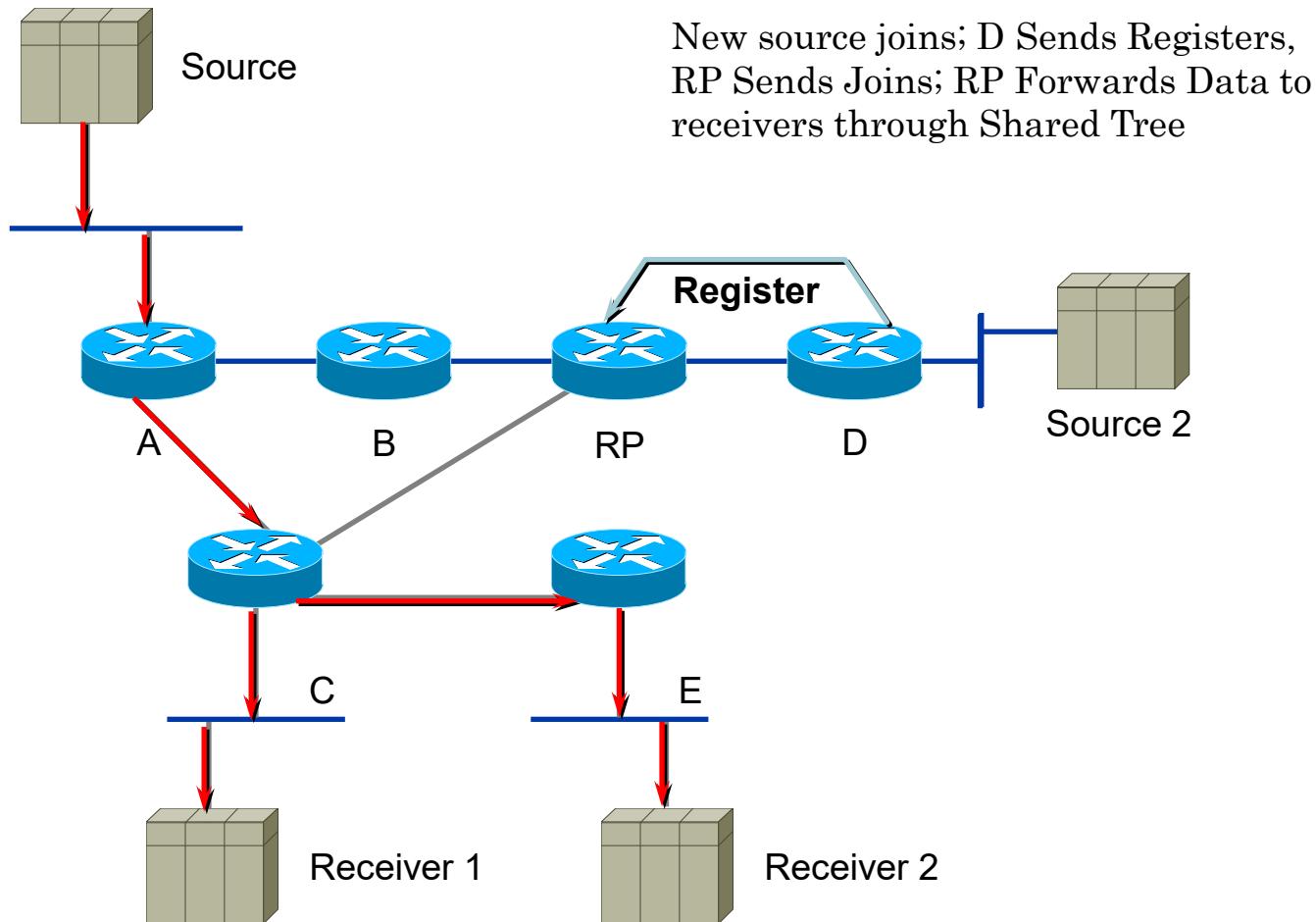
Multicast Routing Protocols – PIM

■ PIM – Sparse Mode



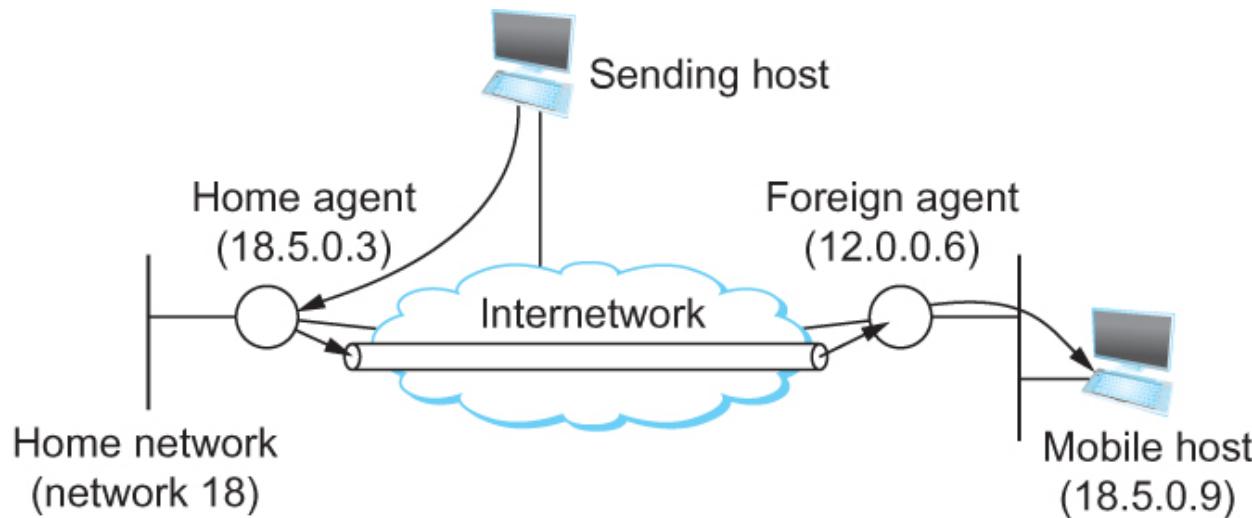
Multicast Routing Protocols – PIM

■ PIM – Sparse Mode



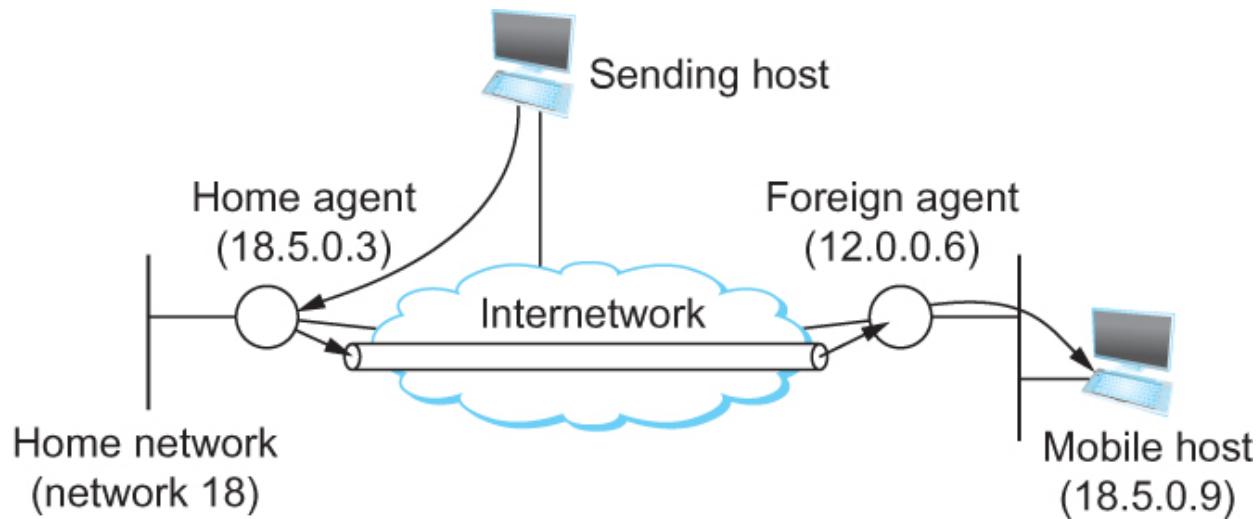
Routing for Mobile Hosts

- Mobile IP
 - *home agent*
 - Router located on the home network of the mobile hosts
 - *home address*
 - The permanent IP address of the mobile host.
 - Has a network number equal to that of the home network and thus of the home agent
 - *foreign agent*
 - Router located on a network to which the mobile node attaches itself when it is away from its home network



Routing for Mobile Hosts

- Mobile host connects to Foreign Agent
- Mobile host provides the address of the Home Agent
- Foreign Agent sends a care-of-address to the Home Agent



Routing for Mobile Hosts

- Problem of delivering a packet to the mobile node
- How does the home agent intercept a packet that is destined for the mobile node?
 - Proxy ARP –
 - same as ARP, except Home Agent inserts the Mobile Hosts IP address
 - Nodes on the network associate the MAC address of the Home Agent with the Mobile Hosts IP address
- How does the home agent then deliver the packet to the foreign agent?
 - IP tunnel
 - Care-of-address

Routing for Mobile Hosts

- Problem of delivering a packet to the mobile node
 - How does the foreign agent deliver the packet to the mobile node?
 - Foreign Agent receives the IP packet for the Mobile Host with the IP address of the Mobile Host
 - Foreign Agent recognizes the IP address as a registered mobile node and delivers the packet to the hardware address of the Mobile Node

Routing for Mobile Hosts

- Route optimization in Mobile IP
 - The route from the sending node to mobile node can be significantly sub-optimal – want to avoid the Home Agent
 - One extreme example
 - The mobile node and the sending node are on the same network, but the home network for the mobile node is on the far side of the Internet
 - Triangle Routing Problem

Routing for Mobile Hosts

- Route optimization in Mobile IP - solution
 - Let the sending node know the care-of-address of the mobile node. The sending node can create its own tunnel to the foreign agent
 - Home agent sends binding update message
 - The sending node creates an entry in the binding cache
 - The binding cache may become out-of-date
 - The mobile node moved to a different network
 - Foreign agent sends a binding warning message

Chapter Summary – Advanced Networking

- BGP
- IPv6: addressing
- Multicast: IGMP and PIM
- Mobile IP

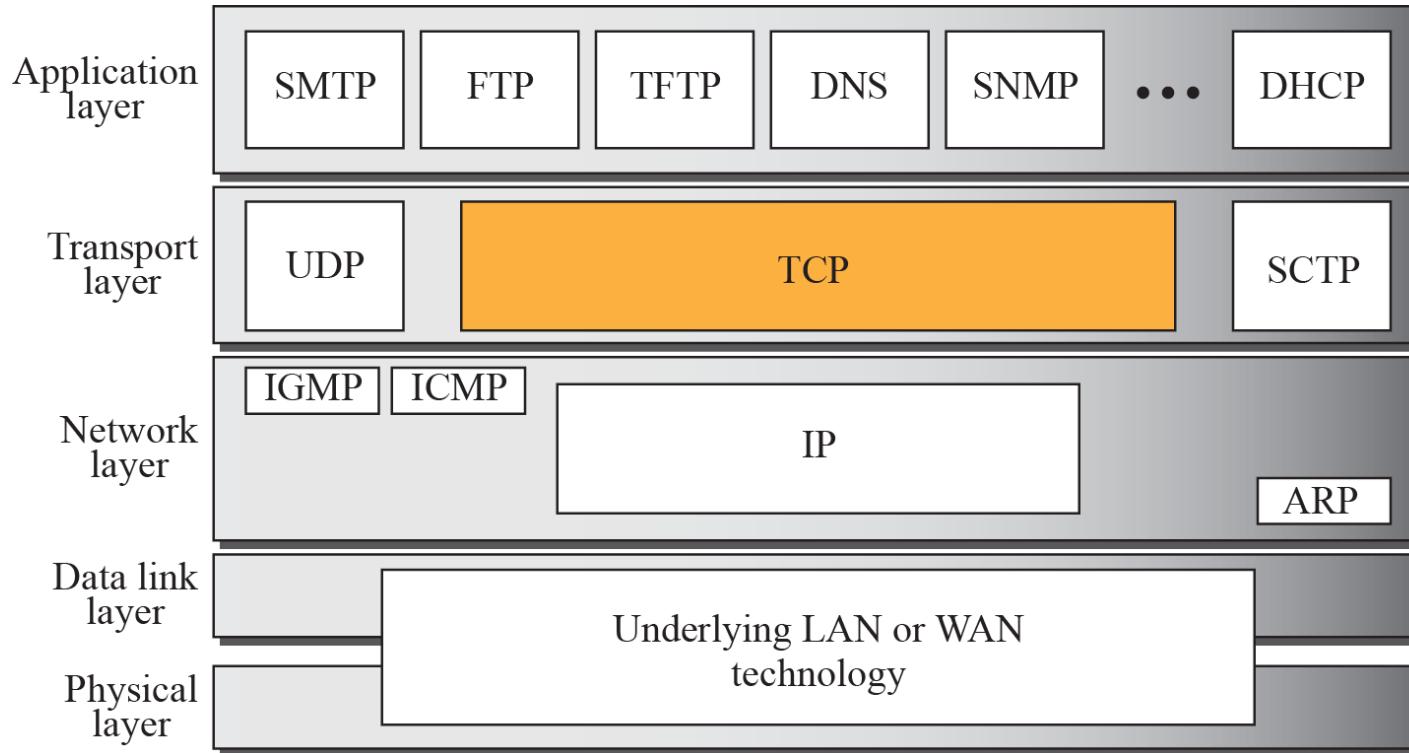
CPE348: Introduction to Computer Networks

Lecture #15: Chapter 5.1



Jianqing Liu
Assistant Professor of Electrical and Computer
Engineering, University of Alabama in Huntsville
jianqing.liu@uah.edu
<http://jianqingliu.net>

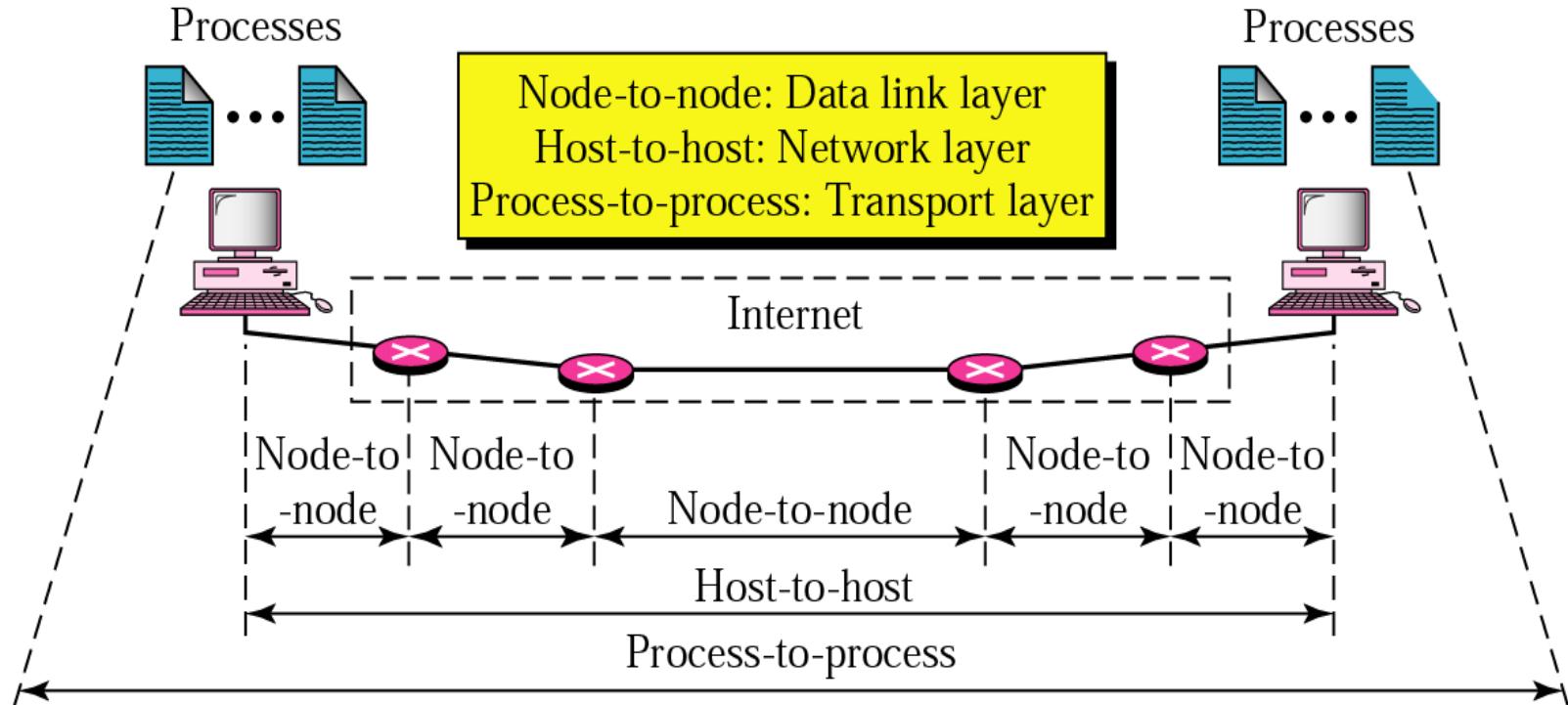
Chapter 5 – Overview



Chapter 5 – Outline

- End-to-end processes to consider:
 - Simple Demultiplexer (UDP)
 - Reliable Byte Stream (TCP)
 - Request/Reply Protocol (RPC)
 - Multimedia Specific Protocol (RTP)

Transport layer – Overview



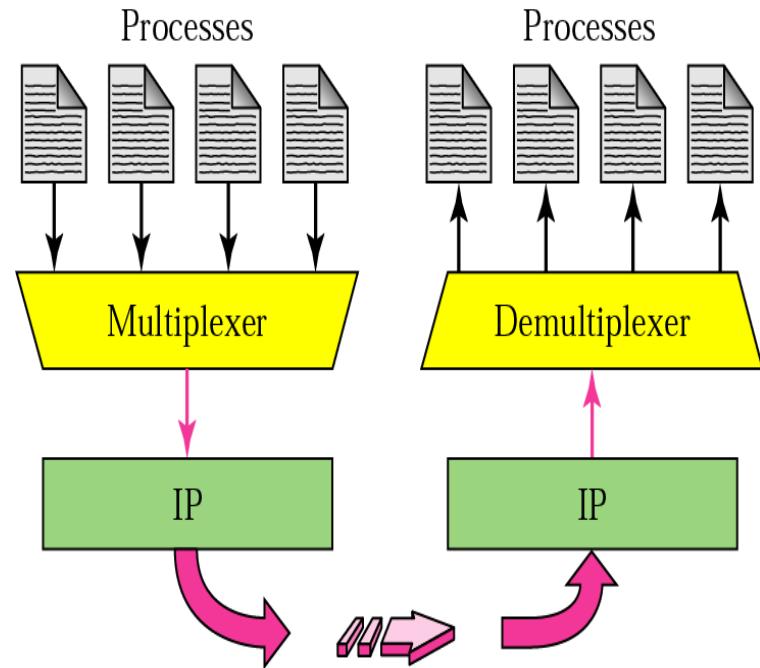
Transport layer – Overview

Multiplexing

Sender may have several processes (e.g., Youtube, WhatsApp) that need to send packets

Demultiplexing

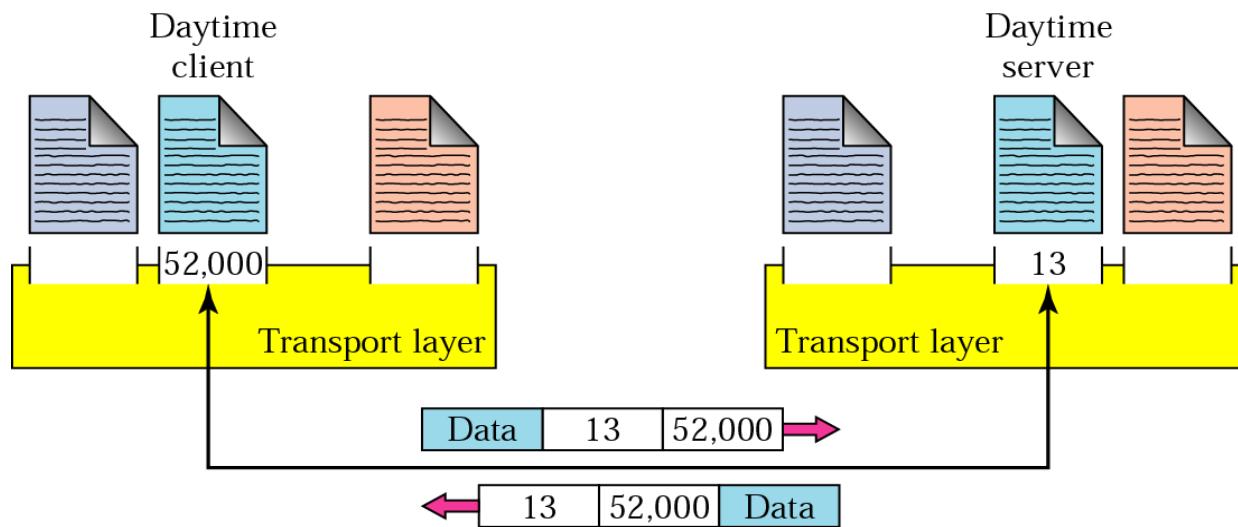
At receiver side, after error checking and header dropping, transport-layer delivers each message to appropriate process



Transport layer – Overview

Addressing

- Data link layer → MAC address
- Network layer → IP address
- Transport layer → **Port number** (choose among multiple processes running on destination host)



Transport layer – Overview

- Port numbers are 16-bit integers (0→65,535)
- Servers use **well known** ports.
 - DNS – port 53 (UDP)
 - FTP – port 21 (TCP)
 - HTTP – port 80 (TCP)
 - Mail service SMTP – port 25 (TCP)
 - SSH – port 22 (TCP)
- Clients use **short-lived** ports
- Server and Client can agree on a new port



If no transport layer protocols

- Typical limitations of the network:
 - Drop messages
 - Reorder messages
 - Limit messages to some finite size
 - Deliver messages after an arbitrarily long delay
- Network is providing a best-effort level of service
 - IP is an example

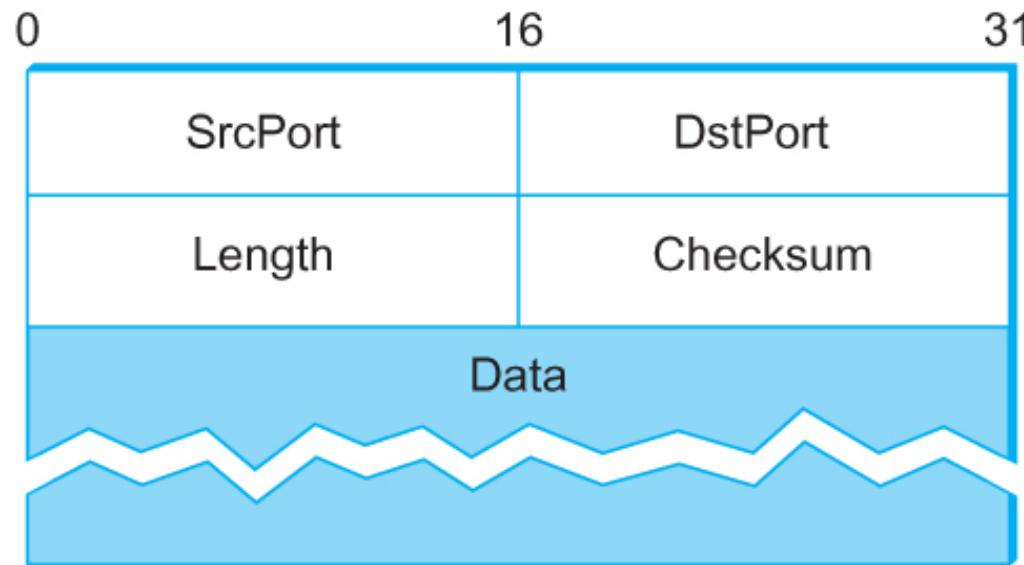
Transport layer – Capability

- A transport protocol promises to
 - Guarantee **message delivery**
 - Deliver messages in the **correct order**
 - Support arbitrarily **large messages**; multiple application **processes** on each host
 - Allow **flow control**, **congestion control** and **QoS provisioning**

Transport layer – Goal

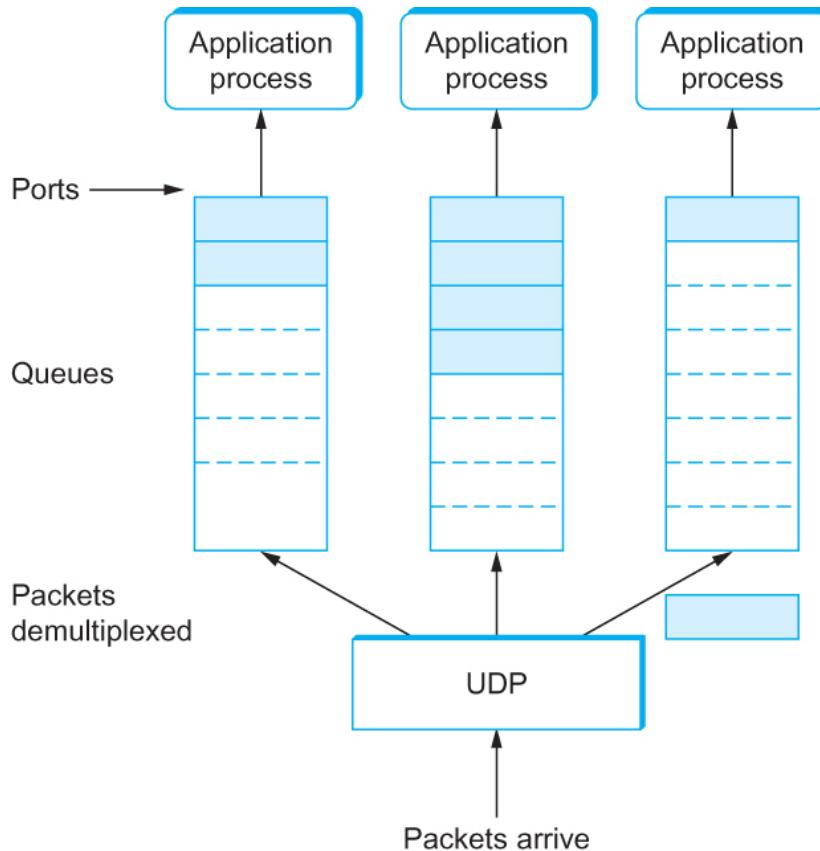
Develop protocols that turn the less-than-desirable underlying network into the high level of service required by application programs!

User Datagram Protocol (UDP)



Format for UDP header

UDP – Simple Demultiplexer



UDP Message Queue

UDP – Properties

- UDP does not have a flow-control mechanism
- UDP does not implement reliable delivery
- But, good for burst data transfer, low-latency, (e.g., live-streaming), loss-tolerating applications



Transmission Control Protocol (TCP)

- TCP must perform functions:
 - Segmentation → breaks message into packets
 - End-to-end error control → since IP is an unreliable Service
 - End-to-end flow control → to avoid host buffer overflow
 - End-to-end congestion control → to avoid network congestion
 - Multiplexing and demultiplexing sessions
- TCP promises to be:
 - Reliable
 - Connection-oriented → virtual circuit
 - Stream-oriented → users exchange streams of data
 - Full duplex → concurrent transfers can take place in both directions
 - Buffered → TCP accepts data and transmits when appropriate

Flow Control vs Congestion Control

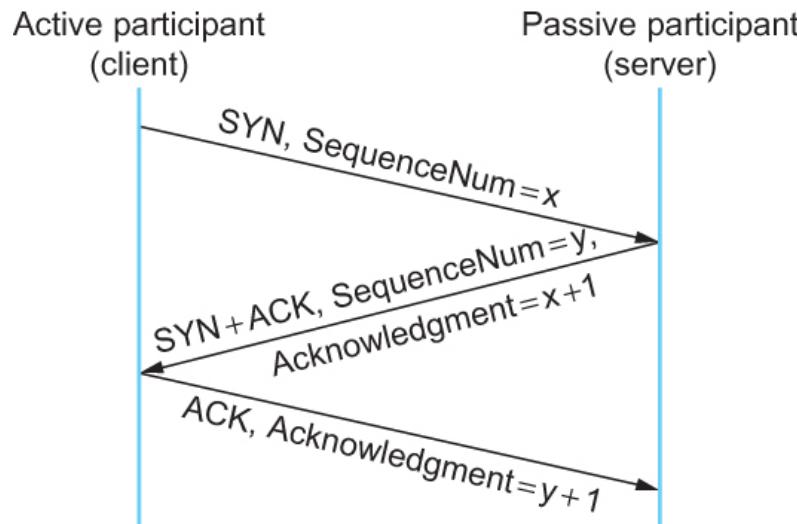
- Flow control - prevent senders from overrunning the capacity of the receivers
- Congestion control - prevent too much data from being injected into the network, thereby causing switches or links to become overloaded

TCP Design – Overview

- Reliable
 - Requires ACK and performs retransmission;
 - If ACK not received, retransmit;
 - After a number of retransmissions, give up;
 - How long to wait for ACK? (next lecture)

TCP Design – Overview

■ Connection-Oriented : Connection Establishment



Timeline for three-way handshake algorithm

x is starting sequence number for client – selected at random

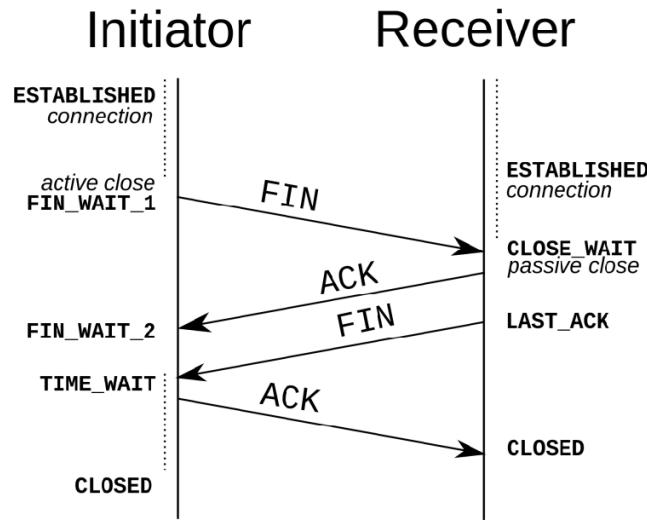
y is starting sequence number for server – selected at random

(want to avoid reusing same sequence numbers to soon)

Ack value identifies **next sequence number expected**

TCP Design – Overview

■ Connection-Oriented : Connection Termination



Timeline for four-way handshake algorithm

Fin: Finish (data with SeqNum = x)

ACK: SeqNum = x+1

FIN: Finish (data with SeqNum = y)

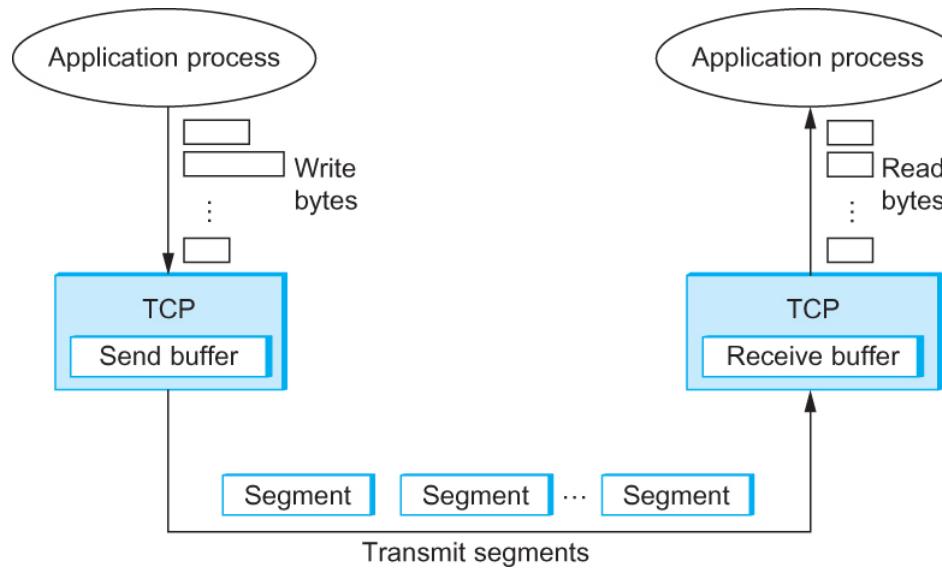
ACK: SeqNum = y+1

TCP Design – Overview

- Correct order
 - Use sequence numbers (next lecture)
 - Associated with every byte that it sends
 - To detect packet loss, reordering and duplicate removal
 - To protect replay attack

TCP Design – Overview

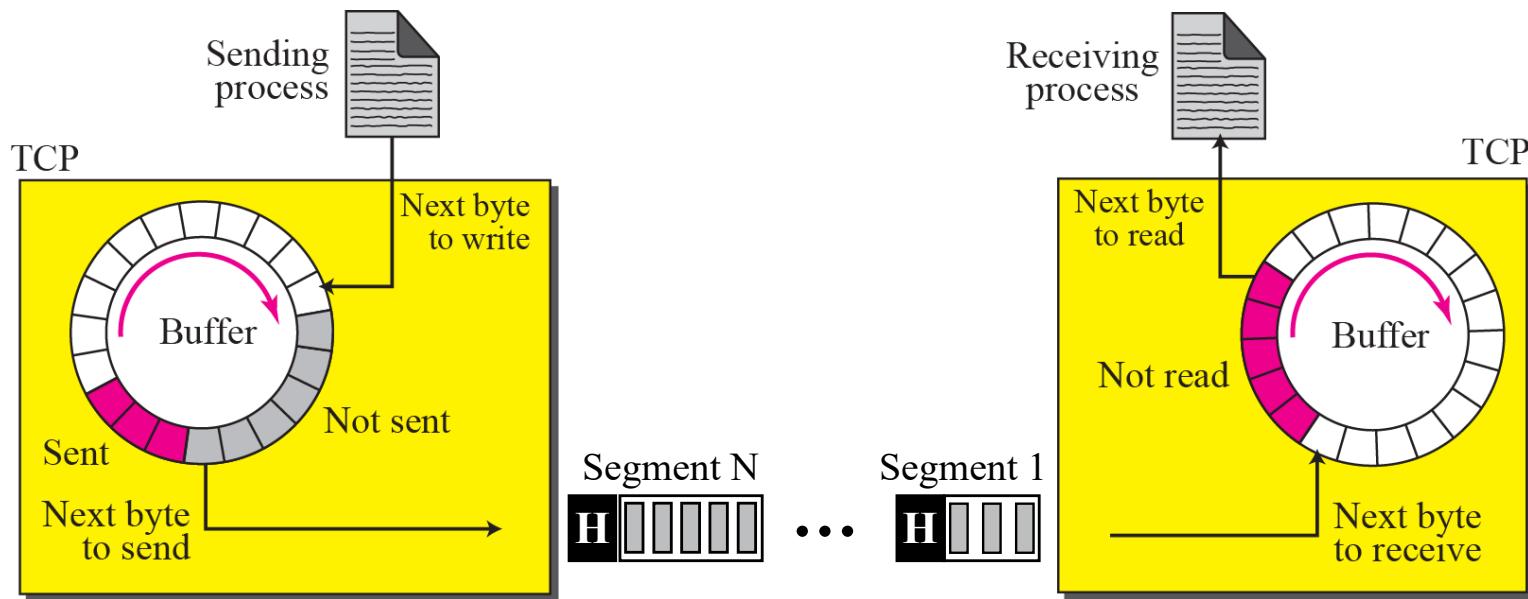
- Congestion and Flow Control
 - Segmentation, buffers and sliding window (future lectures)
 - TCP is a byte-oriented protocol



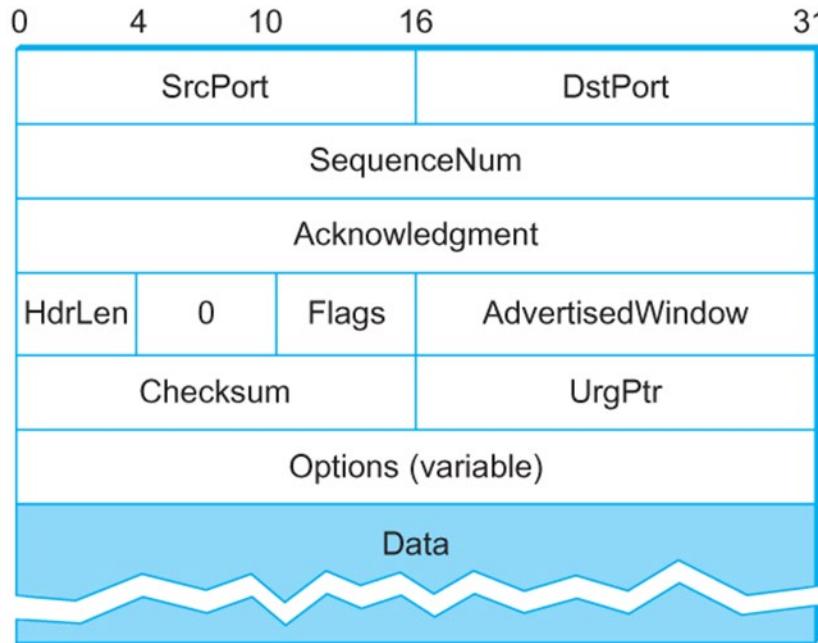
How TCP manages a byte stream.

TCP Design – Overview

- Congestion and Flow Control
 - Segmentation, buffers and sliding window (future lectures)
 - TCP is a byte-oriented protocol

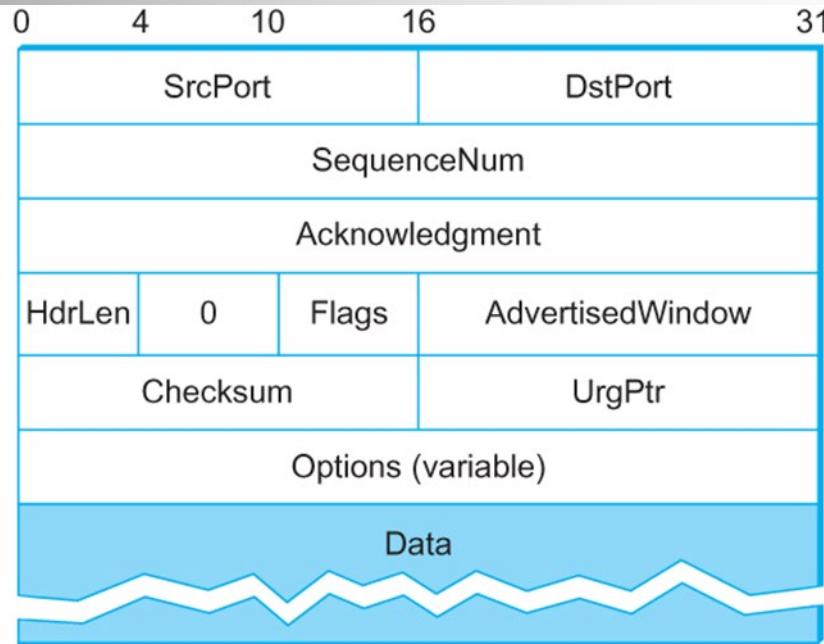


TCP Header



- The SrcPort and DstPort fields identify the source and destination ports, respectively.
- The Acknowledgment, SequenceNum, and AdvertisedWindow fields are all involved in TCP's sliding window algorithm.

TCP Header



- Because TCP is a byte-oriented protocol, each byte of data has a sequence number;
- the SequenceNum field contains the sequence number for the first byte of data carried in that segment.
- The Acknowledgment and AdvertisedWindow fields carry information about the flow of data going in the other direction.

TCP Header

- The Checksum field is used in exactly the same way as for UDP—it is computed over
 - the TCP header,
 - the TCP data, and
 - the pseudo header, which is made up of the source address, destination address, and length fields from the IP header.

CPE348: Introduction to Computer Networks

Lecture #16: Chapter 5.2

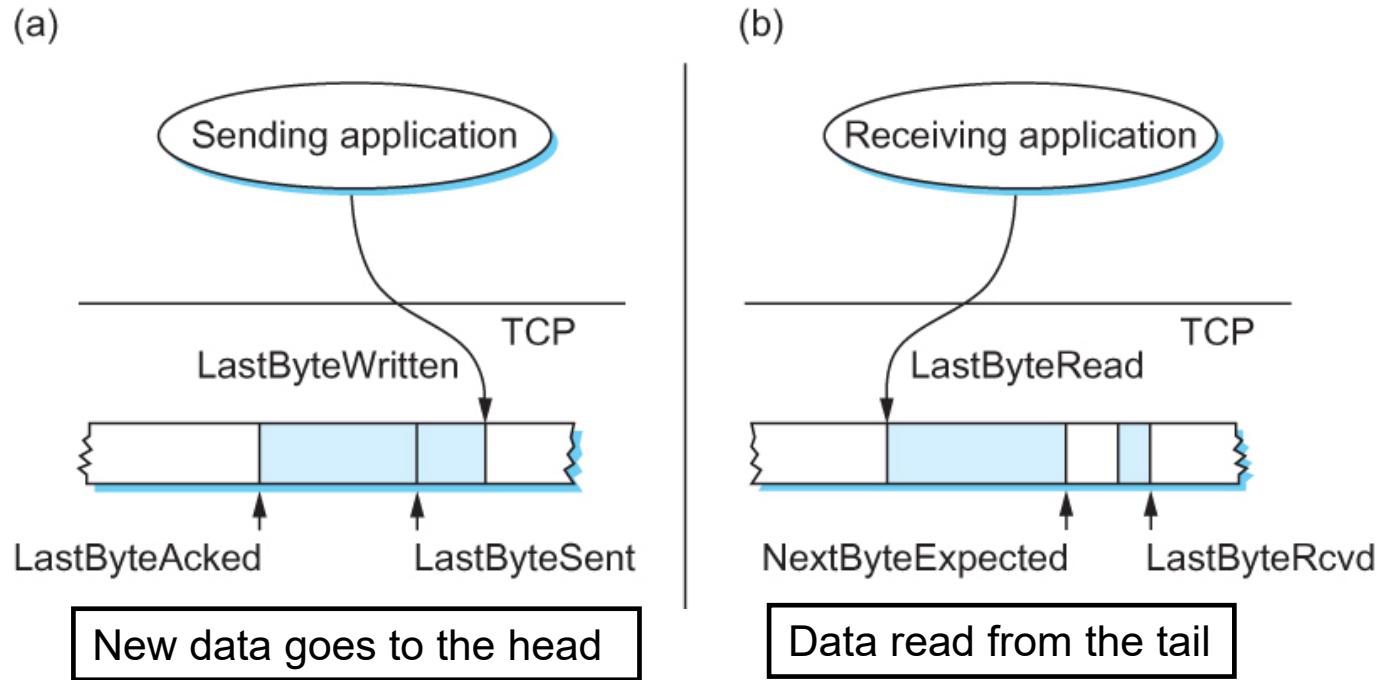


Jianqing Liu
Assistant Professor of Electrical and Computer
Engineering, University of Alabama in Huntsville
jianqing.liu@uah.edu
<http://jianqingliu.net>

Sliding Window Revisited

- TCP's variant of the sliding window algorithm, which serves several purposes:
 - (1) it guarantees the reliable delivery of data,
 - (2) it ensures that data is delivered in order, and
 - (3) it enforces flow control between the sender and the receiver.
- Window size is not fixed
 - Receiver advertises a window size to the sender

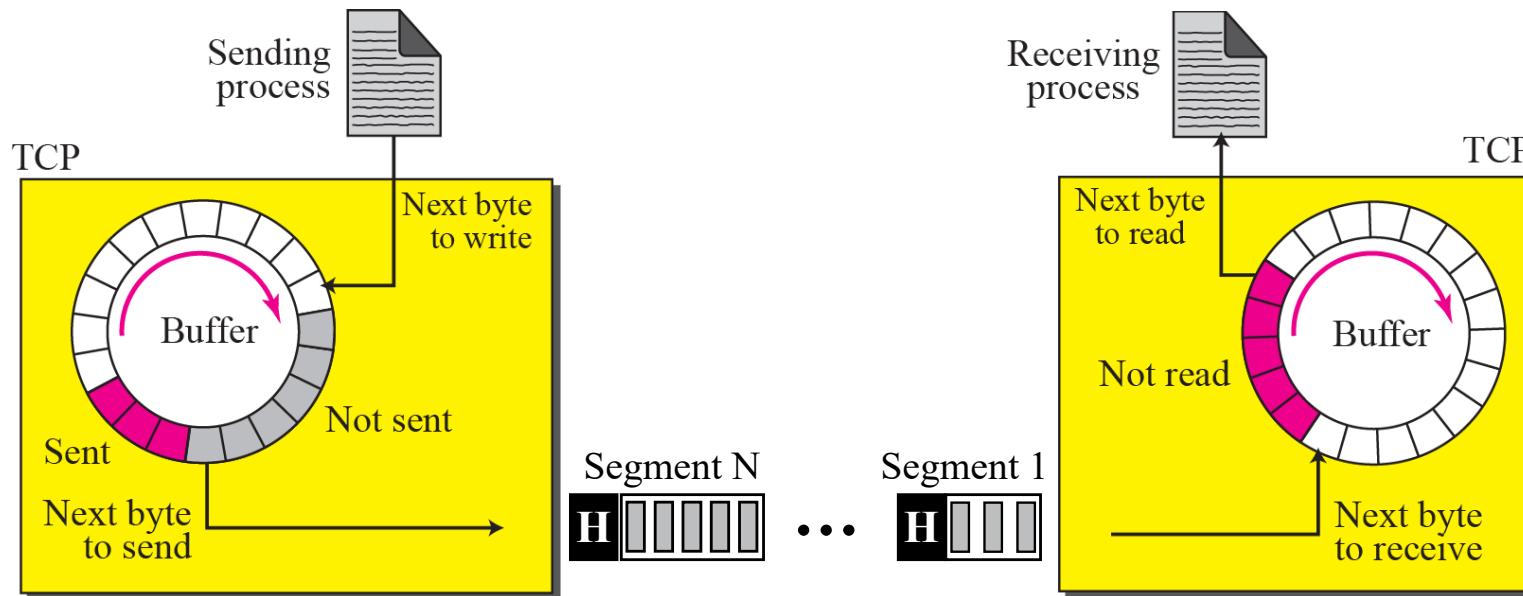
Sliding Window Revisited



Relationship between TCP send buffer (a) and receive buffer (b).

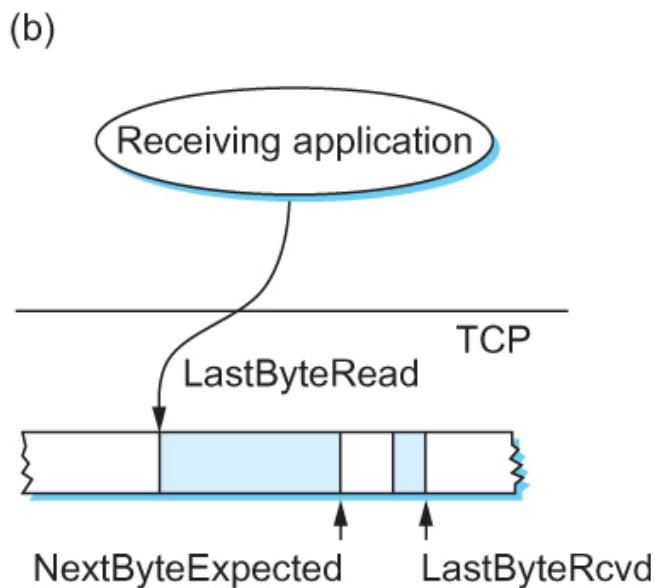
Sliding Window Revisited

- Another angle to look at the TCP sliding window protocol



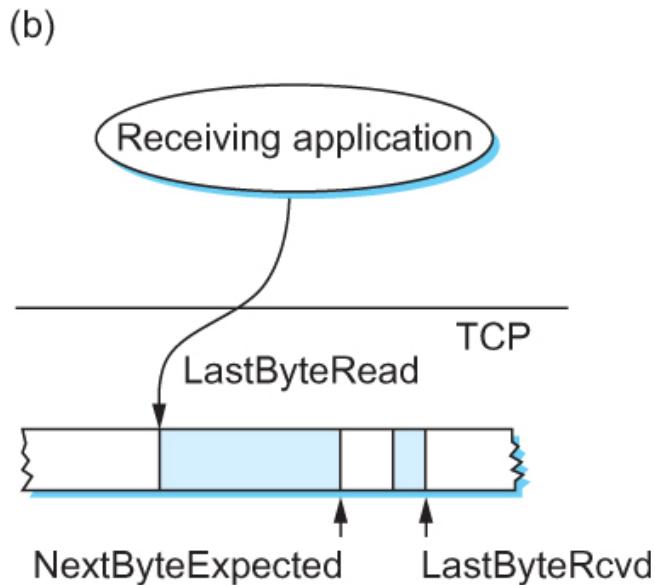
TCP Sliding Window – Rcvr

- Three pointers into receive buffer
 - `LastByteRead`, `NextByteExpected` and `LastByteRcvd`
 - $\text{LastByteRead} < \text{NextByteExpected}$
 - $\text{NextByteExpected} \leq \text{LastByteRcvd} + 1$
 - `NextByteExpected` points to next byte to be received
 - Bytes to left of `LastByteRead` are dropped from buffer



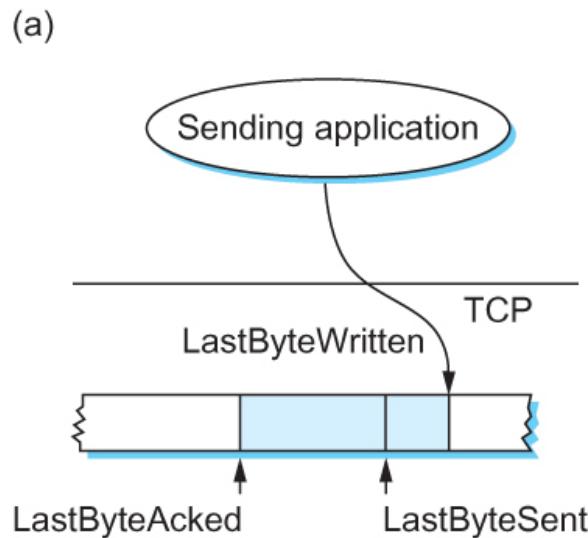
TCP Sliding Window – Rcvr

- Three pointers into receive buffer
 - $\text{LastByteRcvd} - \text{LastByteRead} \leq \text{MaxRcvBuffer}$
 - **AdvertisedWindow** =
$$\text{MaxRcvBuffer} - ((\text{NextByteExpected} - 1) - \text{LastByteRead})$$
 - As data is received, the advertised window decreases
 - As data is read by the process, the advertised window increases



TCP Sliding Window – Tx

- Three pointers into sending buffer
 - `LastByteAcked`, `LastByteSent` and `LastByteWritten`
 - $\text{LastByteAcked} \leq \text{LastByteSent}$
 - $\text{LastByteSent} \leq \text{LastByteWritten}$
 - Acked bytes of data are dropped out of the buffer

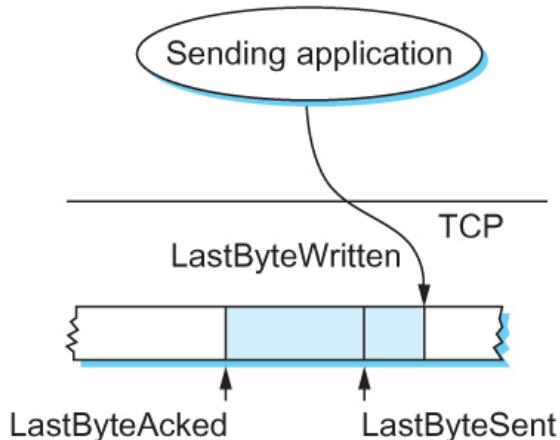


TCP Sliding Window – Tx

- Three pointers into sending buffer
 - $\text{LastByteSent} - \text{LastByteAcked} \leq \text{AdvertisedWindow}$
 - $\text{LastByteWritten} - \text{LastByteAcked} \leq \text{MaxSendBuffer}$
 - EffectiveWindow** =
 $\text{AdvertisedWindow} - (\text{LastByteSent} - \text{LastByteAcked})$
 - If the sending process tries to write y bytes to TCP, but
 $(\text{LastByteWritten} - \text{LastByteAcked}) + y > \text{MaxSendBuffer}$ then TCP blocks the sending process and does not allow it to generate more data.

(a)

If advertised window drops to zero, the sender periodically sends 1 byte segments until a non-zero advertised window size is returned in the ack



TCP Flow Control

Recall what flow control is.

AdvertisedWindow is for flow control purpose!

If AdvertisedWindow $\rightarrow 0$, the sender periodically sends 1 byte segments until a non-zero AdvertisedWindow size is returned in the Ack

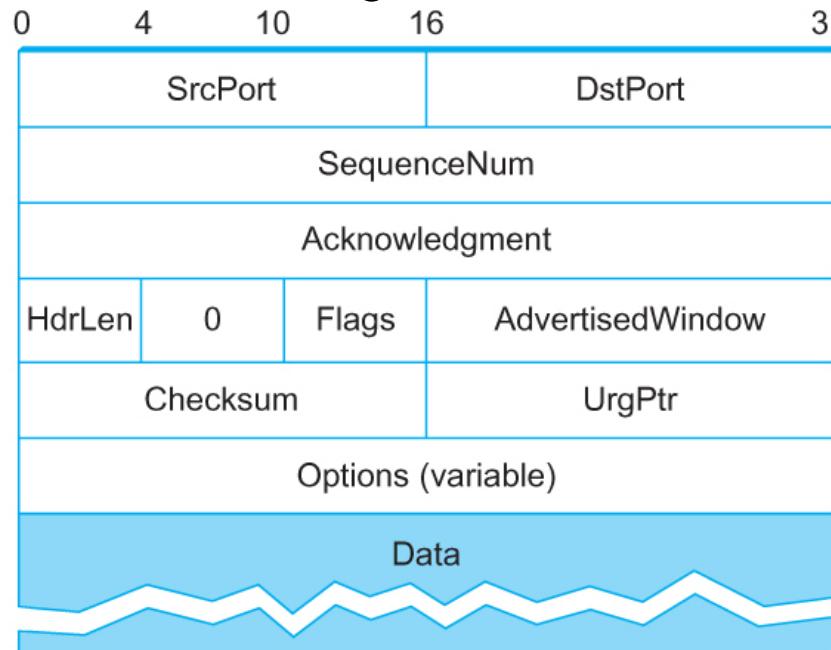
TCP Flow Control

Recall the sliding window protocol on Layer 2.

What is the **relationship** between SWS/RWS and the Sequence Number?

TCP – Sequence Number

- SequenceNum: 32 bits longs
- AdvertisedWindow: 16 bits long
- TCP satisfies the sliding window algorithm requirement that the sequence number space be twice as big as the window size
 $2^{32} \gg 2 \times 2^{16} = 2^{17}$



TCP – Sequence Number

- TCP has 32-bit sequence number space
 - The sequence number used on a given connection might wraparound.
 - How much time does it take for TCP SN to wraparound?
Depending on the data rate!



TCP – Sequence Number Wraparound

Bandwidth	Time until Wraparound
T1 (1.5 Mbps)	6.4 hours
Ethernet (10 Mbps)	57 minutes
T3 (45 Mbps)	13 minutes
Fast Ethernet (100 Mbps)	6 minutes
OC-3 (155 Mbps)	4 minutes
OC-12 (622 Mbps)	55 seconds
OC-48 (2.5 Gbps)	14 seconds

Time until 32-bit sequence number space wraps around.
(2^{32} Bytes = 2^{35} bits = 34.3597 Gbits)

TCP - Keeping the Pipe Full

- 16-bit **AdvertisedWindow** field
 - Theoretically allows for at most 64KB data
- What does **delay x bandwidth** tell us?

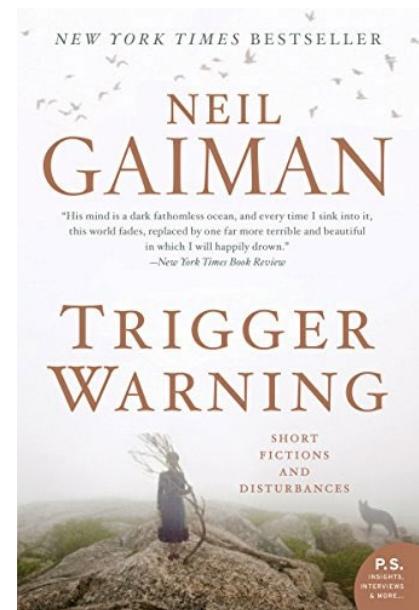
Bandwidth	Delay \times Bandwidth Product
T1 (1.5 Mbps)	18 KB
Ethernet (10 Mbps)	122 KB
T3 (45 Mbps)	549 KB
Fast Ethernet (100 Mbps)	1.2 MB
OC-3 (155 Mbps)	1.8 MB
OC-12 (622 Mbps)	7.4 MB
OC-48 (2.5 Gbps)	29.6 MB

Required window size for 100-ms RTT.

Extensions to TCP allow for the advertised window to indicate X number of bytes

TCP - Triggering Transmission

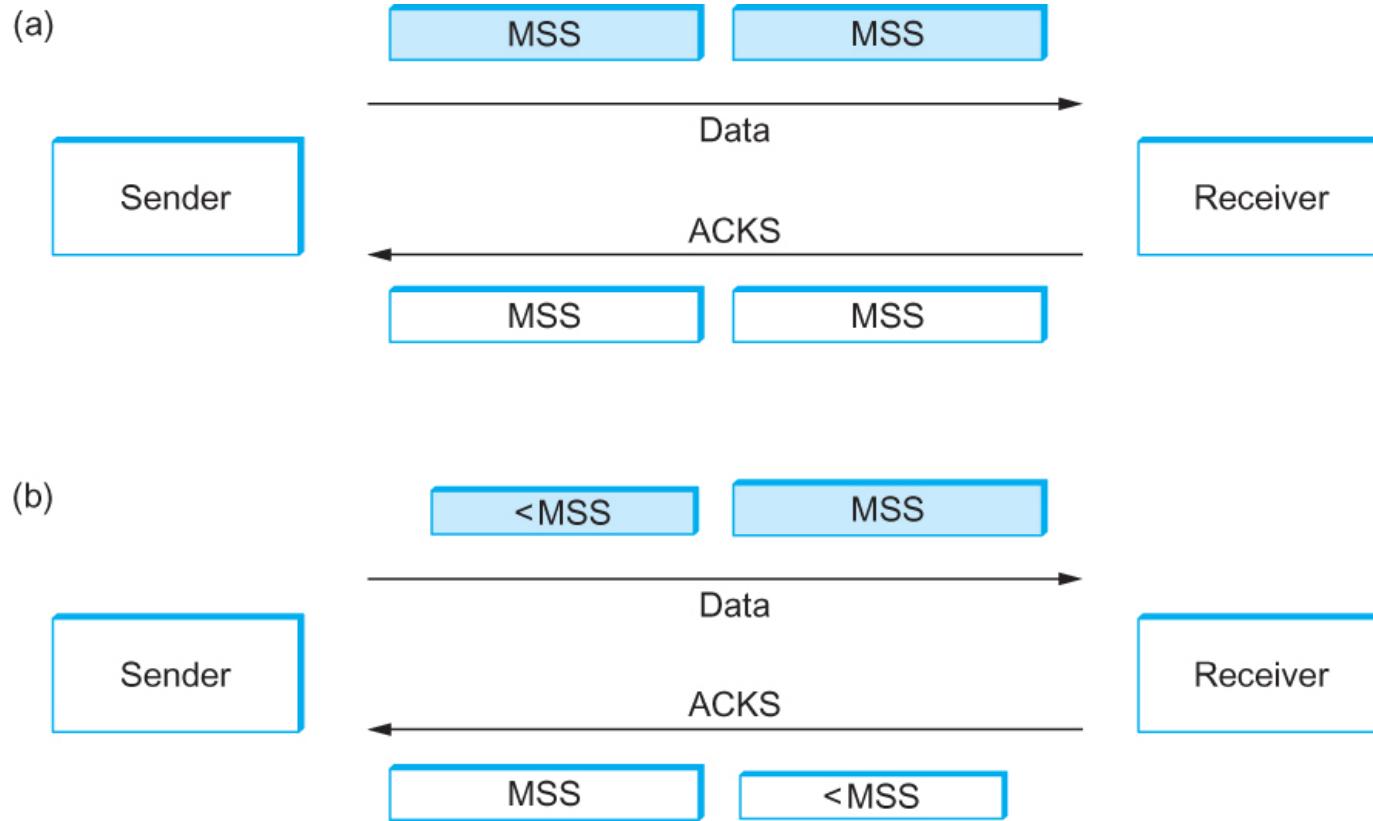
- How does TCP decide to transmit a segment?
 - Application programs write bytes into streams
 - It is up to TCP to decide that it has enough bytes to send a segment



TCP - Triggering Transmission

- What factors governs the decision to transmit
 - TCP has three mechanism to trigger the transmission of a segment
 - 1) TCP maintains a variable for Maximum Segment Size (MSS)
 - TCP sends a segment as soon as it has collected MSS bytes from the sending process
 - MSS is usually set to the size of the largest segment TCP can send without causing local IP to fragment.
 - MSS: use MTU of directly connected network
 - $$\text{MSS} = \text{MTU} - (\text{TCP header} + \text{IP header})$$
 - 2) Sending process has explicitly asked TCP to send it
 - TCP supports push operation
 - 3) When a timer fires
 - Resulting segment contains as many bytes as are currently buffered for transmission

Silly Window Syndrome



Silly Window Syndrome

Silly Window Syndrome

- Sender Aggressively fills any available advertised window size
- If segment size sent is smaller than MSS, that smaller size remains
 - Receiver acks the smaller size
 - Receiver opens up a small window size after a 0 advertised window size
- Only a problem when sender sends a small segment or receiver opens a small window.

Nagle's Algorithm

- If there is data to send but the window is open less than MSS, wait!
- But how long?
 - If we wait too long, long delay!
 - If we wait too short, sending a bunch of tiny packets → waste of resources
- The solution is to introduce a timer and to transmit when the timer expires

Nagle's Algorithm

- We could use a clock-based timer;
- Nagle introduced an elegant **self-clocking** solution
- Key Idea
 - As long as TCP has any data in flight, the sender will eventually receive an ACK
 - This ACK can be treated like a timer firing, triggering the transmission of more data

Nagle's Algorithm

When the application produces data to send

- if both the available data and the window \geq MSS
 - send a full segment
- else
 - if there is unACKed data in flight
 - buffer the new data until an ACK arrives
 - else
 - send all the new data now

TCP - Adaptive Retransmission

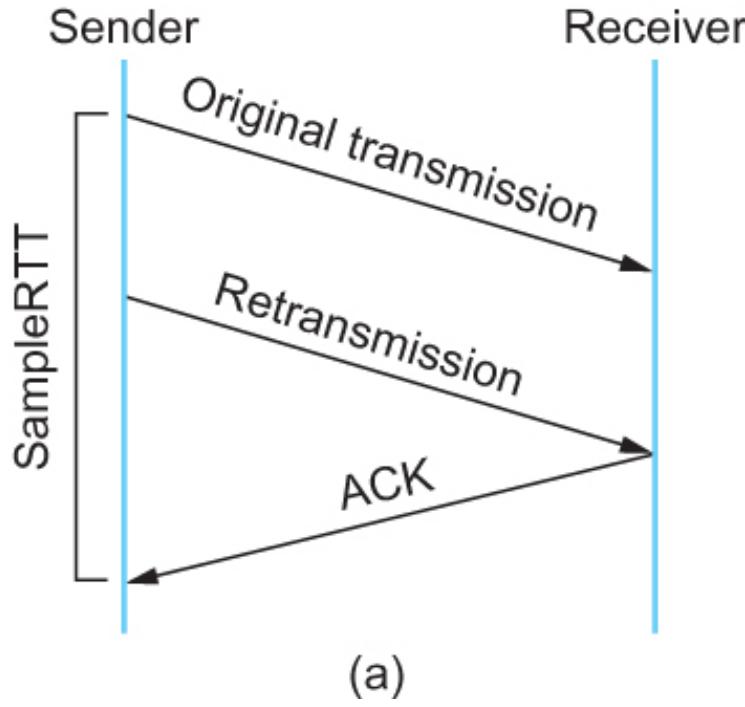
- TCP retransmits a segment if an ACK is not received before a timeout occurs
- Timeout is based on RTT
- RTT can vary, so an adaptive retransmission mechanism is used.



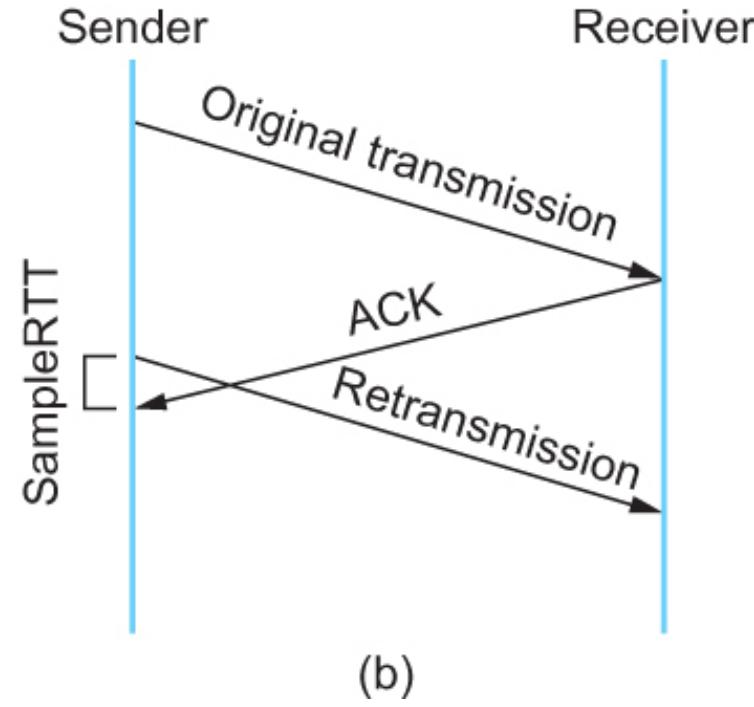
TCP - Adaptive Retransmission

- Original Algorithm
 - Measure **SampleRTT** for each segment/ACK pair
 - Compute weighted average of RTT
 - $\text{EstRTT} = \alpha \times \text{EstRTT} + (1 - \alpha) \times \text{SampleRTT}$
 - α between 0.8 and 0.9
 - Set timeout based on **EstRTT**
 - $\text{TimeOut} = 2 \times \text{EstRTT}$

TCP - Adaptive Retransmission –Problem



Sample RTT too large



Sample RTT too small

Associating the ACK with (a) original transmission versus (b) retransmission

Karn/Partridge Algorithm

- Measure sample RTT for segments sent **only once**
- Do not sample RTT for retransmitted segments
- Double timeout after each retransmission – **exponential backoff**

Jacobson/Karels Algorithm

- Main problem with the original computation is that it does not take variance of Sample RTTs into consideration.
- If the variance among Sample RTTs is small then
 - Estimated RTT can be better trusted
 - No need to multiply this by 2 to compute the timeout

Jacobson/Karels Algorithm

Difference = SampleRTT – EstimatedRTT

EstimatedRTT = EstimatedRTT + ($\delta \times$ Difference)

Deviation = Deviation + δ (|Difference| – Deviation)

TimeOut = $\mu \times$ EstimatedRTT + ($\phi \times$ Deviation)

- where based on experience, μ is typically set to 1
- ϕ is set to 4, and δ is a fraction between 0 and 1
- Thus, when the variance is small, TimeOut is close to EstimatedRTT; a large variance causes the deviation term to dominate the calculation.

CPE348: Introduction to Computer Networks

Lecture #17: Chapter 6



Jianqing Liu
Assistant Professor of Electrical and Computer
Engineering, University of Alabama in Huntsville
jianqing.liu@uah.edu
<http://jianqingliu.net>

TCP Congestion Control Overview

- Early on, we talked about:
 - TCP sliding window **protocol**;
 - TCP adaptive transmission **algorithms**.
- In this lecture, we will talk about:
 - TCP congestion control **protocols**
 - Additive Increase/Multiplicative Decrease
 - Slow Start
 - Fast Retransmit and Fast Recovery

TCP Congestion Control – history

- It was introduced into the Internet in the late 1980s by Van Jacobson, ~ 8 yrs after the TCP/IP had become operational.
- Immediately preceding this time, the Internet was suffering from congestion collapse —
 - hosts would send their packets into the Internet as fast as the **advertised window** would allow
 - congestion would occur at some router, causing pkts drop
 - hosts would time out and retransmit their packets, resulting in even more congestion

TCP Congestion Control – history

- Basic idea
 - Source determines how much capacity is available in the network,
 - it uses the arrival of an ACK as a signal that one of its packets has left the network,
 - source can then insert a new packet into the network.
 - By using ACKs to pace the transmission of packets, TCP is said to be *self-clocking*.

TCP Congestion Control

- Three popular TCP congestion control protocols
 - Additive Increase/Multiplicative Decrease
 - Slow Start
 - Fast Retransmit and Fast Recovery



Additive Increase Multiplicative Decrease

- Source maintains a new variable **CongestionWindow**
- TCP's effective window is revised as follows:
 - **MaxWindow = MIN(CongestionWindow, AdvertisedWindow)**
 - **EffectiveWindow =**
$$\text{MaxWindow} - (\text{LastByteSent} - \text{LastByteAcked}).$$
- That is, **MaxWindow** replaces **AdvertisedWindow**;
- Source is allowed to send no faster than the slowest component—the network or the destination host—can accommodate.



Additive Increase Multiplicative Decrease

- Question is how source comes to learn an appropriate value for **CongestionWindow**.
 - First, source sets it as a default value based its perception to the network.
 - Then,
 - Decrease the **CongestionWindow** when the level of congestion goes up and
 - Increase the **CongestionWindow** when the level of congestion goes down.
 - Taken together, the mechanism is commonly called *additive increase/multiplicative decrease (AIMD)*

Additive Increase Multiplicative Decrease

- The next question is how source comes to learn congestion occurs.
 - Observation: when packets are not delivered, timeout occurs
 - Then, source interprets timeouts as a sign of congestion and reduces the rate at which it is transmitting.
 - When a timeout occurs, the source sets CongestionWindow to **half** of its previous value – **multiplicative decrease**
 - When no timeout occurs, the source increments CongestionWindow by **1** – **additive increase**.

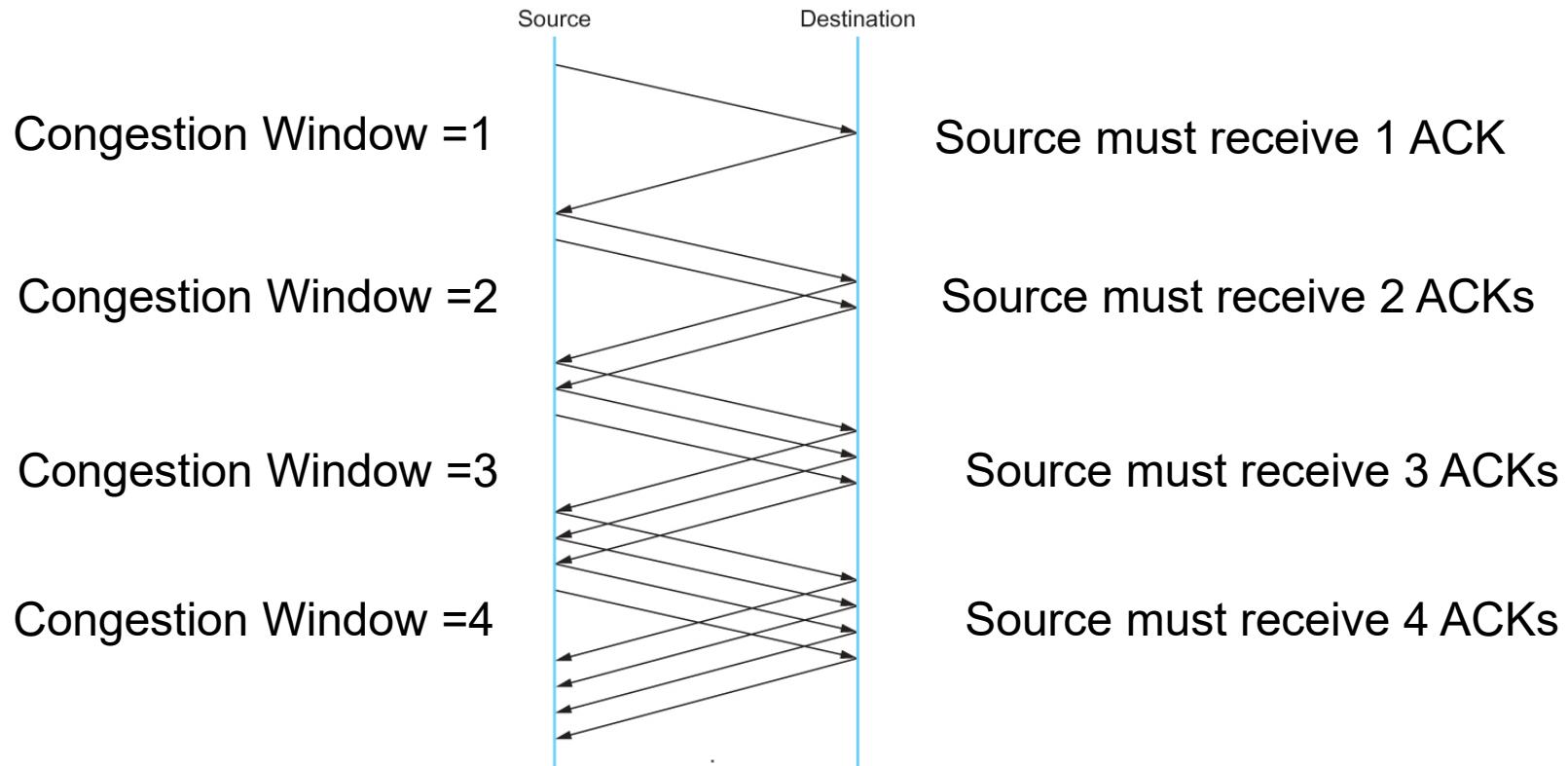
Additive Increase Multiplicative Decrease

- For example
 - CongestionWindow is defined in terms of bytes;
 - Then,
 - Firstly, CongestionWindow is currently set to 16 packets. If a loss is detected, CongestionWindow is set to 8 packets.
 - Additional losses cause CongestionWindow to be reduced to 4, then 2, and finally to 1 packet.
 - CongestionWindow is not allowed to fall below the size of a single packet, or in TCP terminology, the *maximum segment size (MSS)*.



Additive Increase Multiplicative Decrease

■ Another example



Packets in transit during additive increase, with one packet being added each RTT.

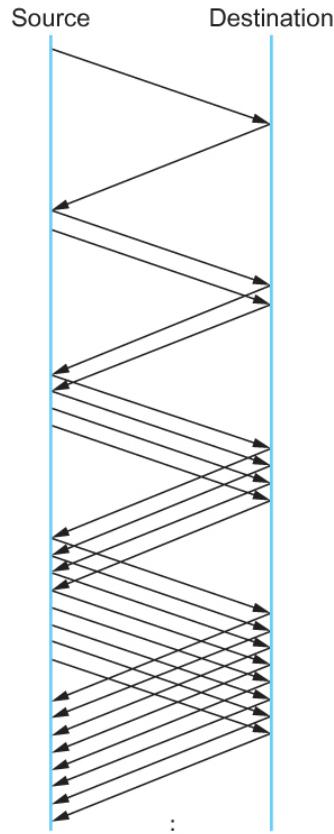
Slow Start

- Drawback of AIMD: **too conservative!** It takes too long to ramp up a connection starting from scratch.
- TCP therefore provides a second mechanism, ironically called *slow start*,
 - Increase the **CongestionWindow** rapidly from a cold start when the **CongestionWindow** size starts at 1;
 - Slow start effectively increases the congestion window exponentially, rather than linearly (additive increase).

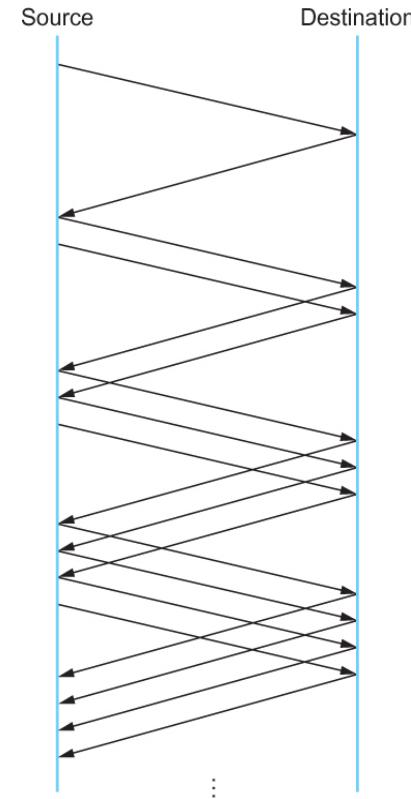
Slow Start

- For example
 - Source starts out by setting **CongestionWindow** to 1 packet.
 - When the ACK for this packet arrives, source adds 1 to **CongestionWindow** and then sends 2 packets.
 - Upon receiving the corresponding 2 ACKs, source increments **CongestionWindow** by 2 and next sends 4 packets.
 - The result is that source doubles the number of packets it has in transit every RTT.

Slow Start



Packets in transit
during slow start.



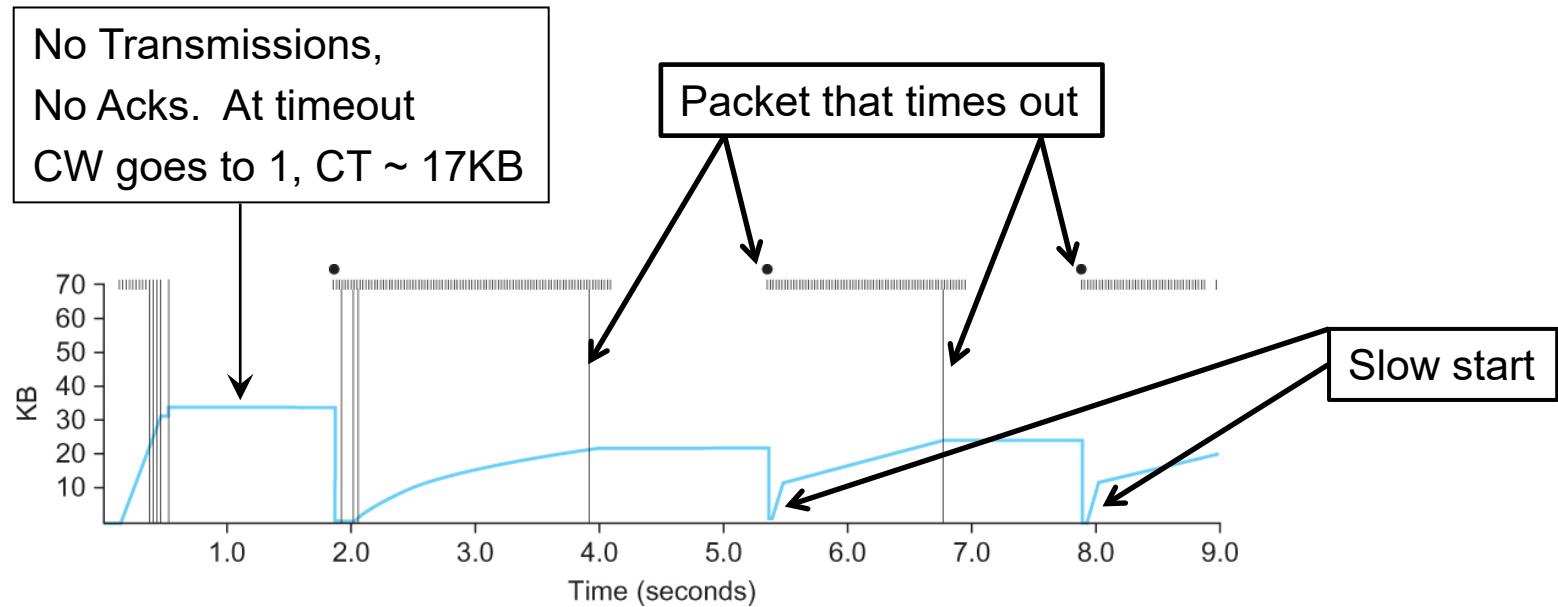
Packets in transit
during additive increase

Slow Start

- There are two situations in which slow start is **favorable**:
 - The first is at the very beginning of a connection:
 - Slow start continues to double **CongestionWindow** each RTT
 - After a packet is lost, a timeout causes multiplicative decrease to divide **CongestionWindow** by 2.
 - The second happens when the connection goes dead while waiting for a timeout to occur:
 - After communications for a while, source has a useful value of **CongestionWindow**, which is called **CongestionThreshold**;
 - Use **CongestionThreshold** as the “target” **CongestionWindow**.
 - Slow start is used to rapidly increase the sending rate up to this value, and then additive increase is used beyond this point. (**Combination! Fine-resolution!**)

Slow Start

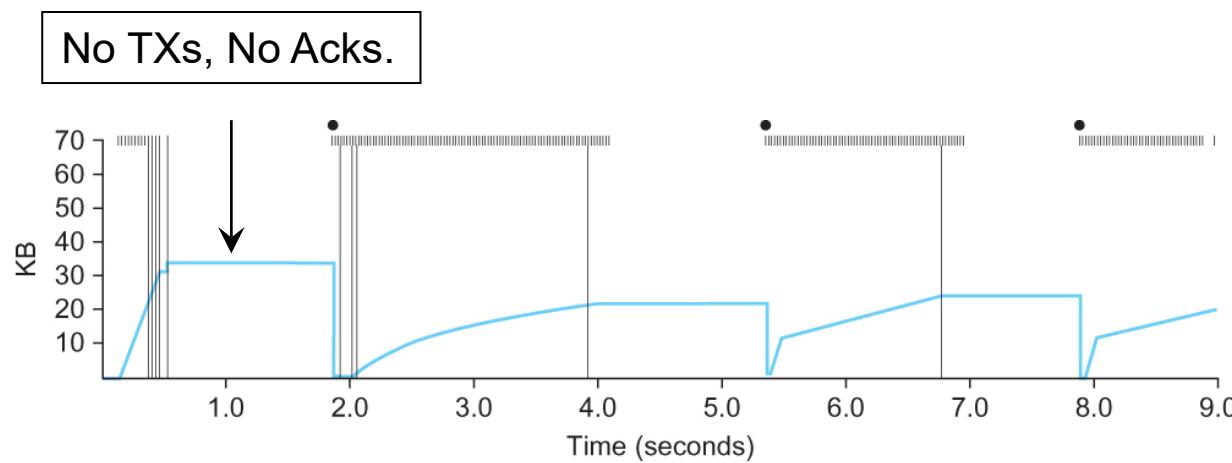
- Cont' – Example for timeout occurs



Behavior of TCP congestion control. Colored line = value of CongestionWindow over time; solid bullets at top of graph = timeouts; hash marks at top of graph = time when each packet is transmitted; vertical bars = time when a packet that was eventually retransmitted was first transmitted.

Fast Retransmit and Fast Recovery

- Issue of slow start + AIMD: TCP timeouts led to long periods of time during which the connection went dead while waiting for a timer to expire.



- A new mechanism called **fast retransmit** was added to TCP.
- It is a heuristic that sometimes triggers the retransmission of a dropped packet sooner than the timeout.

Fast Retransmit and Fast Recovery

- Basic idea
 - Receiver responds with an ACK every time a packet arrives - even if it has already been ACKed
 - When an out of order packet is received, TCP resends the same ACK as last time
 - TCP cannot ACK the data contained in an out of order packet

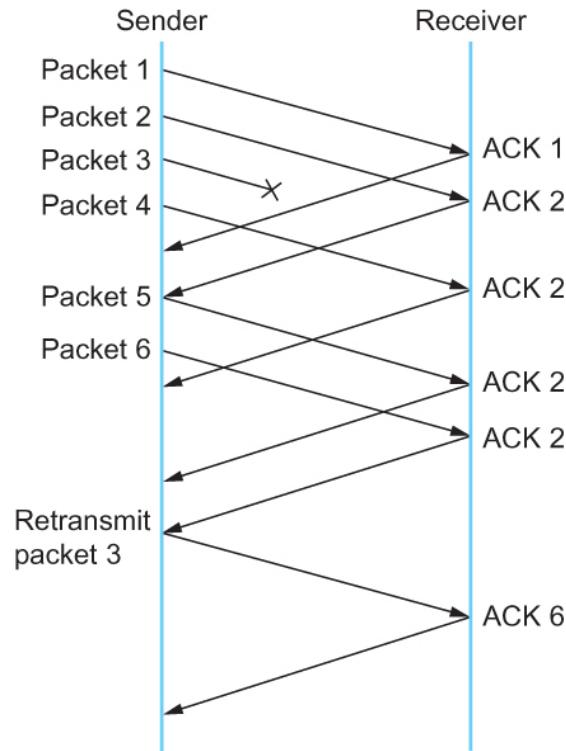
Receiver Assisted Approach

Fast Retransmit and Fast Recovery

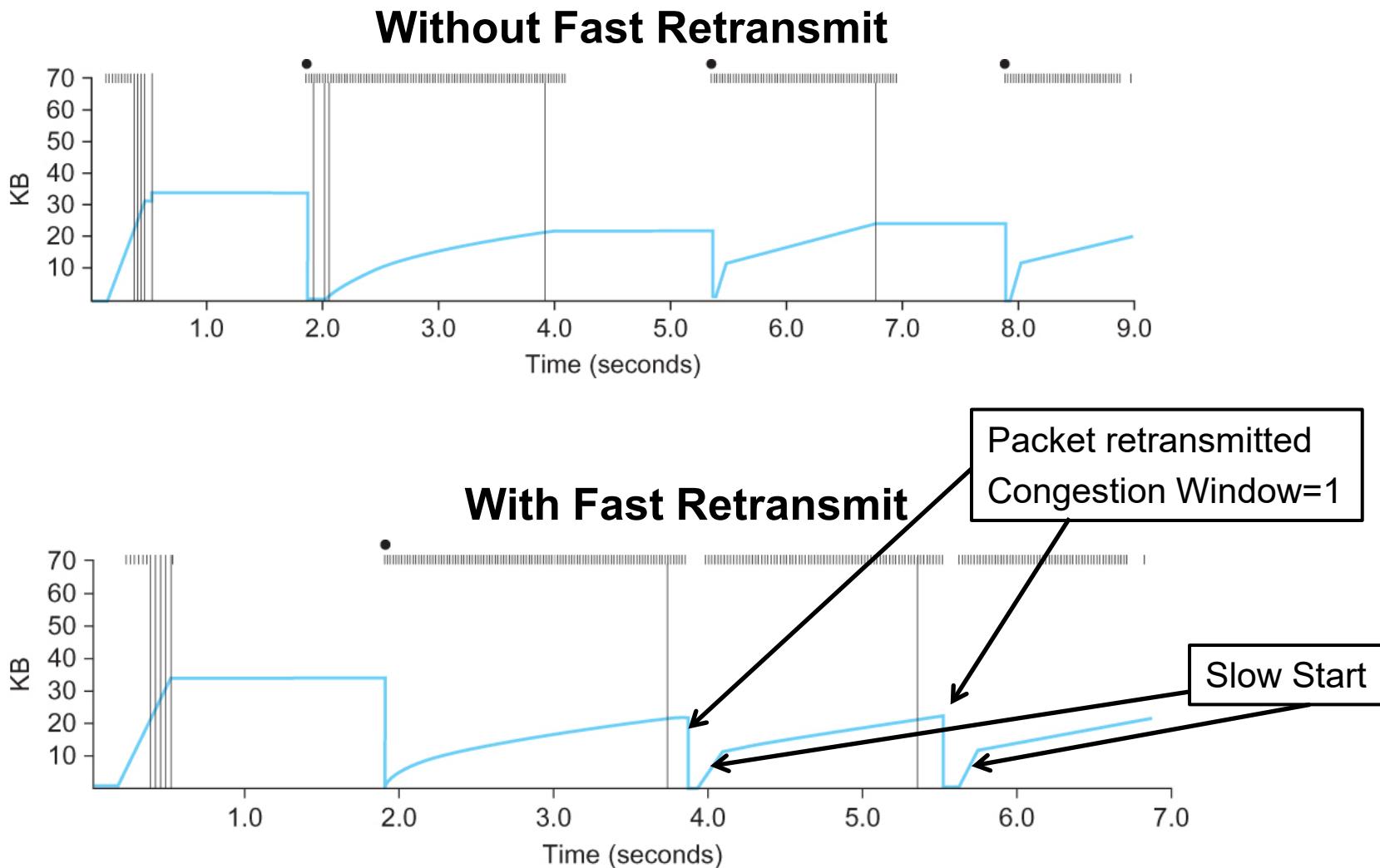
- Basic idea – cont'
 - When the source sees a duplicate ACK,
 - It knows that the other side must have received a packet out of order,
 - Suggests that an earlier packet might have been lost.
 - In practice, TCP waits until it has seen three duplicate ACKs before retransmitting the packet.
 - in case the packet is delayed instead of lost

Fast Retransmit and Fast Recovery

■ Example



Fast Retransmit and Fast Recovery



Fast Retransmit and Fast Recovery

■ Fast Recovery

- When the fast retransmit mechanism signals congestion,
 - NOT drop the CongestionWindow back to 1 packet and run slow start,
 - it is possible to use the ACKs that are still in the pipe to clock the sending of packets.
- This mechanism, which is called **fast recovery**.

TCP Congestion Control Summary

- In this lecture, we will talked about:
 - TCP congestion control protocols;
 - Additive Increase/Multiplicative Decrease
 - Slow Start
 - Fast Retransmit and Fast Recovery
 - The (dis-)advantages of them;
 - You are expected to draw the tx diagram given a specific TCP congestion control protocol;

CPE348: Introduction to Computer Networks

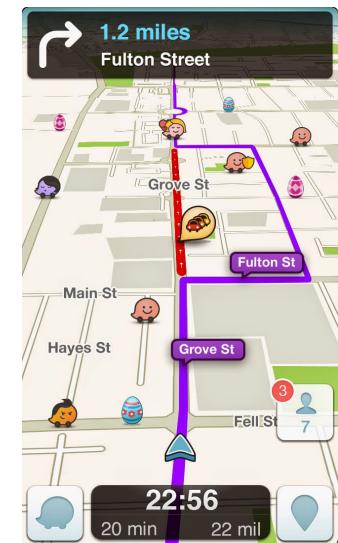
Lecture #18: Chapter 6



Jianqing Liu
Assistant Professor of Electrical and Computer
Engineering, University of Alabama in Huntsville
jianqing.liu@uah.edu
<http://jianqingliu.net>

TCP Congestion Control Overview

- Early on, we talked about:
 - TCP congestion control **protocols**;
- In this lecture, we will talk about:
 - TCP congestion avoidance **protocols**
 - DEC bit
 - Slow Start
 - Fast Retransmit and Fast Recovery



Congestion Avoidance Mechanism

- TCP congestion control is the strategy once congestion happens, as opposed to avoiding congestion in the first place.
- An appealing alternative is **congestion avoidance**
 - Has not yet been widely adopted,
 - Tries to predict when congestion is about to happen
 - Reduces the rate at which source sends data just before packets start getting lost.

Congestion Avoidance Mechanism

■ DEC Bit

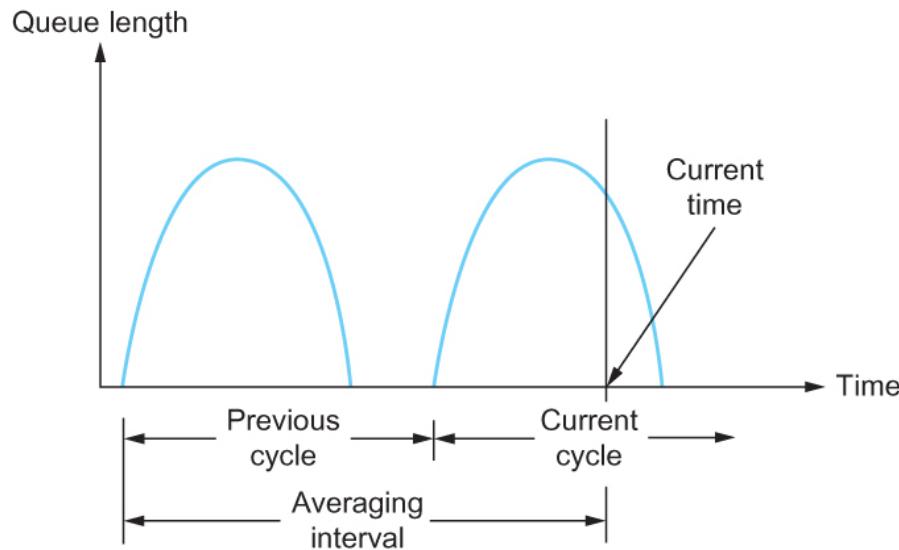
- Initially developed for Digital Network Architecture (DNA), a connectionless network with a connection-oriented transport protocol;
- This mechanism could, also be applied to TCP/IP.
- The idea is to more evenly split the responsibility for congestion control between the **routers** and the **end nodes**.

Congestion Avoidance Mechanism

- DEC Bit
 - Each router monitors the load it is experiencing and explicitly notifies the end nodes when congestion is about to occur.
 - This notification is implemented by setting a binary congestion bit in the packets that flow through the router; hence the name DECbit.
 - The destination host then copies this congestion bit into the ACK it sends back to the source.
 - Finally, the source adjusts its sending rate so as to avoid congestion

Congestion Avoidance Mechanism

- DEC Bit – router participation
 - The router calculates the average queue length over the specified interval.



Computing average queue length at a router

Congestion Avoidance Mechanism

- DEC Bit – source participation
 - The source records how many of its packets resulted in some router setting the congestion bit.
 - In particular, the source maintains a congestion window, just as in TCP
 - The source watches to see what fraction of the last window's worth of packets resulted in the congestion bit being set.

Congestion Avoidance Mechanism

- DEC Bit – source participation
 - If less than 50% of the packets had the bit set, then the source increases its congestion window by one packet.
 - If 50% or more of the last window's worth of packets had the congestion bit set, then the source decreases its congestion window to 0.875 times the previous value.

Congestion Avoidance Mechanism

- DEC Bit – source participation
 - 50% was chosen as the threshold based on analysis that showed it to correspond to the peak of the power curve.
 - The “increase by 1, decrease by 0.875” rule was selected because AIMD makes the mechanism stable.

Congestion Avoidance Mechanism

- Random Early Detection (RED)
 - A second congestion avoidance mechanism is called **random early detection (RED)**
 - When a router detects that congestion is imminent, it notifies the source to adjust its congestion window.

Congestion Avoidance Mechanism

- The difference between RED and DECbit
 - RED does not explicitly send a congestion notification message to the source
 - RED is implemented such that it *implicitly notifies* the source of congestion by dropping one of its packets.
 - The source is then notified by the subsequent timeout or duplicate ACK.

Congestion Avoidance Mechanism

■ Cont'

- The router drops the packet earlier than it has to
- This action notifies the source that it should decrease its congestion window sooner than without RED.
- The router drops a few packets before it has exhausted its buffer space completely
 - Causes the source to slow down its transmission rate
 - The router hopes that this will not cause to drop more packets later on.

Aggressive method!

Congestion Avoidance Mechanism

- Cont' – how to drop packets
 - Consider a simple FIFO queue on a router;
 - With RED, a router could decide to drop each arriving packet with some drop probability whenever the queue length exceeds some drop level.
 - This idea is called **early random drop**.

Congestion Avoidance Mechanism

- Cont' – Early Random Drop
 - RED computes an average queue length
 - **AvgLen** is iteratively computed as
$$\text{AvgLen} = (1 - \text{Weight}) \times \text{AvgLen} + \text{Weight} \times \text{SampleLen}$$
 - where $0 < \text{Weight} < 1$ and **SampleLen** is the length of the queue when a sample measurement is made.
 - **Weight** is a constant that determines how sensitive **AvgLen** is to variations in **SampleLen**
 - In most cases, the queue length is measured every time a new packet arrives at the router.

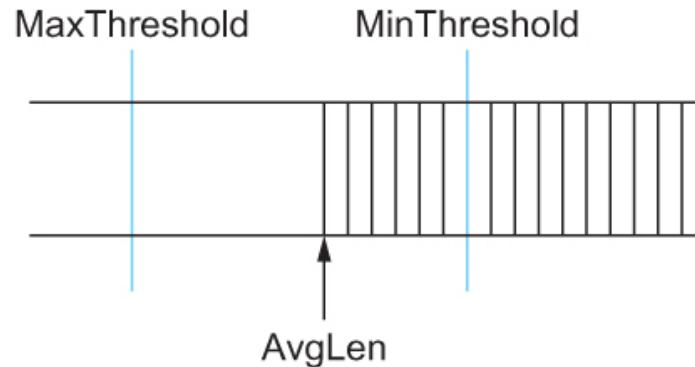
Congestion Avoidance Mechanism

■ Cont' – Early Random Drop

- Next, RED has two queue length thresholds : **MinThreshold** and **MaxThreshold**.
- When a packet arrives at the router, RED compares the current **AvgLen** with them, according to the following rules:
 - if $\text{AvgLen} \leq \text{MinThreshold}$
 → queue the packet
 - if $\text{MinThreshold} < \text{AvgLen} < \text{MaxThreshold}$
 → calculate probability P
 → drop the arriving packet with probability P
 - if $\text{AvgLen} \geq \text{MaxThreshold}$
 → drop the arriving packet

Congestion Avoidance Mechanism

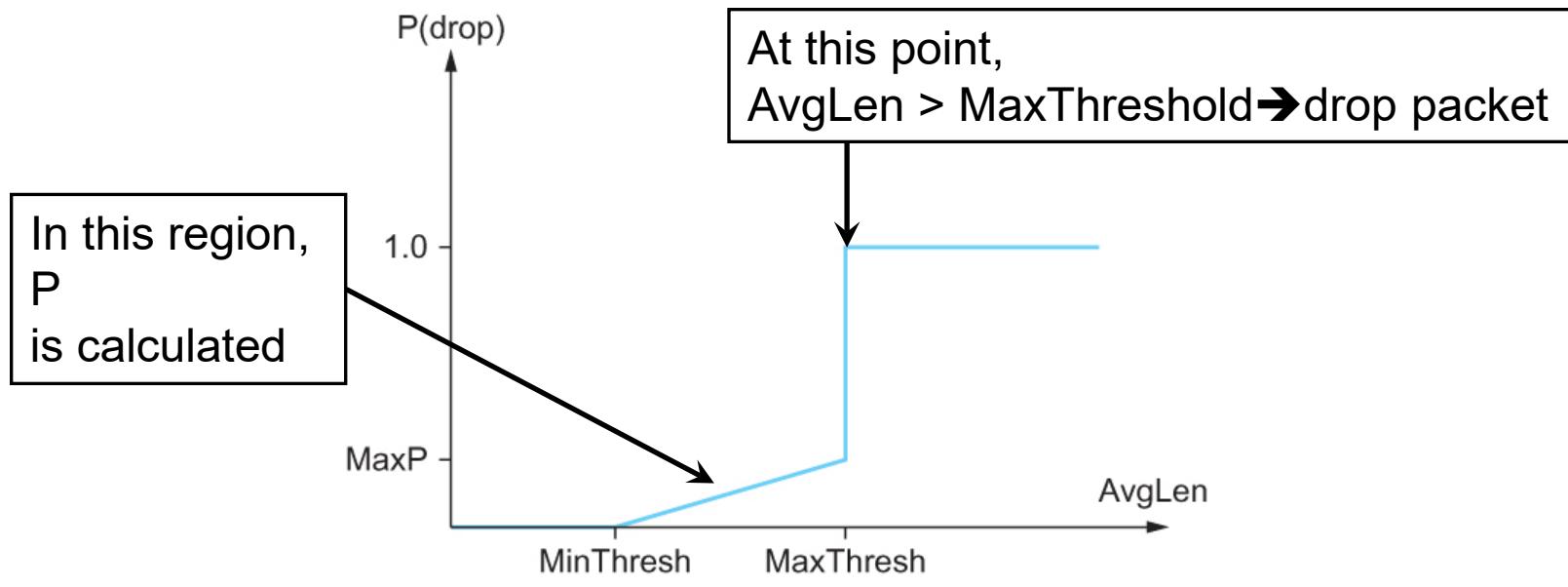
- Cont' – Early Random Drop



RED thresholds on a FIFO queue

Congestion Avoidance Mechanism

- RED – how dropping probability is calculated:



Drop probability function for RED

Congestion Avoidance Mechanism

- Source-based Congestion Avoidance
 - Sources watch for some indication that congestion will happen soon;
 - For example, source may notice an increase in the RTT for each successive packet being sent.

Catch a sign!



Congestion Avoidance Mechanism

- Source-based Congestion Avoidance
 - One algorithm using RTT measurements:
 - The congestion window normally increases as in TCP
 - After every two round-trip delays, the algorithm
 - Averages the minimum and maximum RTT seen so far
 - Checks the current RTT against the average value
 - If the current RTT is greater than the average, source decreases the congestion window by 1/8.

Congestion Avoidance Mechanism

- Source-based Congestion Avoidance
 - Another algorithm decides CongestionWindow based on changes to the RTT and the window size.
 - The congestion window is adjusted once every two RTTs based on the product
 - $(\text{CurrentWindow} - \text{OldWindow}) \times (\text{CurrentRTT} - \text{OldRTT})$
 - For a positive value, decrease the window size by 1/8;
 - For a non-positive value, increase the window by one maximum packet size.

Congestion Avoidance Mechanism

- Source-based Congestion Avoidance
 - A third algorithm the **CongestionWindow** based on the flatness of the sending rate
 - The **CongestionWindow** is adjusted every RTT based on the measured throughput
 - Every RTT increase the congestion window by 1
 - Compare the throughput achieved with the larger window to the throughput achieved with previous congestion window size.
 - If the difference is less than $\frac{1}{2}$ of the throughput when only one packet was in transit (measured at beginning of connection with congestion window at 1) then decrease window by 1.

CPE348: Introduction to Computer Networks

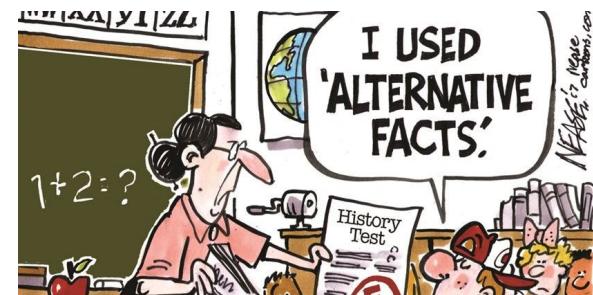
Lecture #19: Chapter 5.4



Jianqing Liu
Assistant Professor of Electrical and Computer
Engineering, University of Alabama in Huntsville
jianqing.liu@uah.edu
<http://jianqingliu.net>

Alternatives to TCP

- Two classes of transport protocols
 - Stream (byte) oriented (TCP)
 - Reliable (TCP)
 - Unreliable(UDP)
 - Request/Reply (RPC) – message oriented

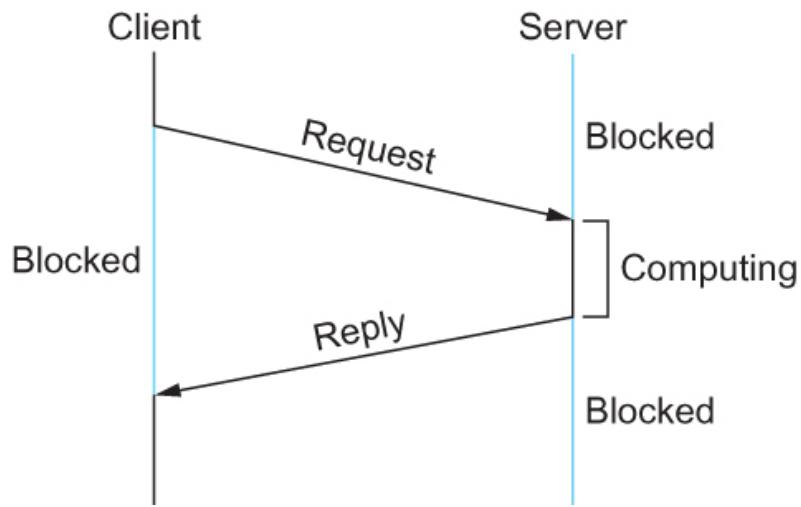


TCP Characteristics Review

- TCP has explicit setup and tear down phases
 - Gives the receiver a chance to deny the connection
 - Tear down allows applications to keep a connection open for long periods of time without the need to send “keep alive” messages
- TCP window based protocol
 - Advertised window can be used with the RTT to determine a rate based design if necessary
- A lot more ...

Remote Procedure Call (RPC)

- Request/Reply Protocol
- Type of protocol not a standard like TCP
- RPC protocols vary in the functions performed

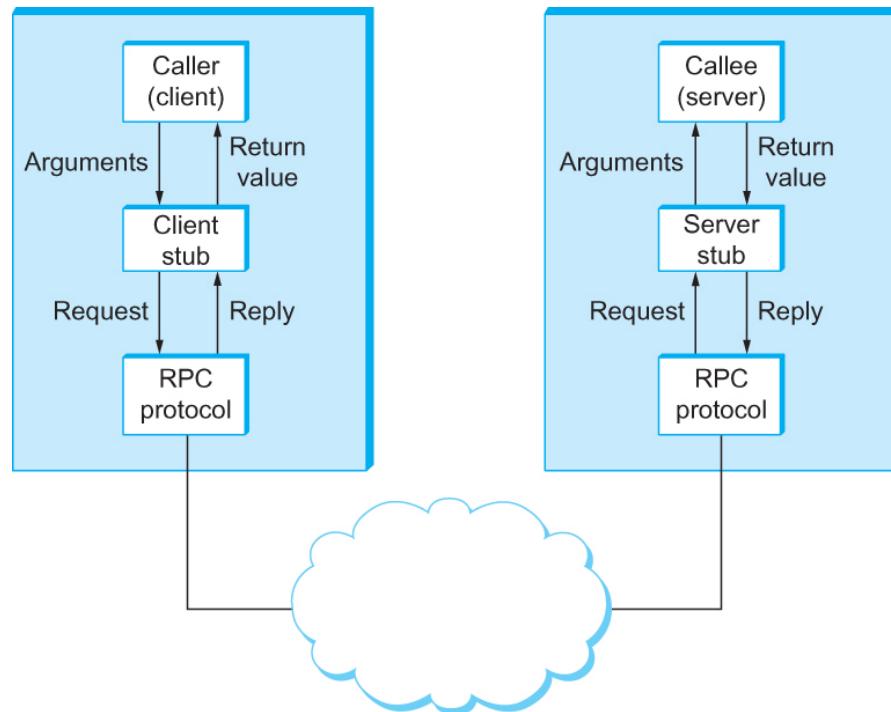


Remote Procedure Call (RPC)

- Applications make a call for a procedure
 - Procedure may be local or
 - Procedure may be remote
- Remote procedure calls are more complicated than local procedure calls
 - Network has more complex properties than a local computer
 - Computers involved may be of different architecture and data representations

Remote Procedure Call (RPC)

- Two major components
 - A protocol for handling messages send/receive – deals with network issues
 - Programming and compiler support

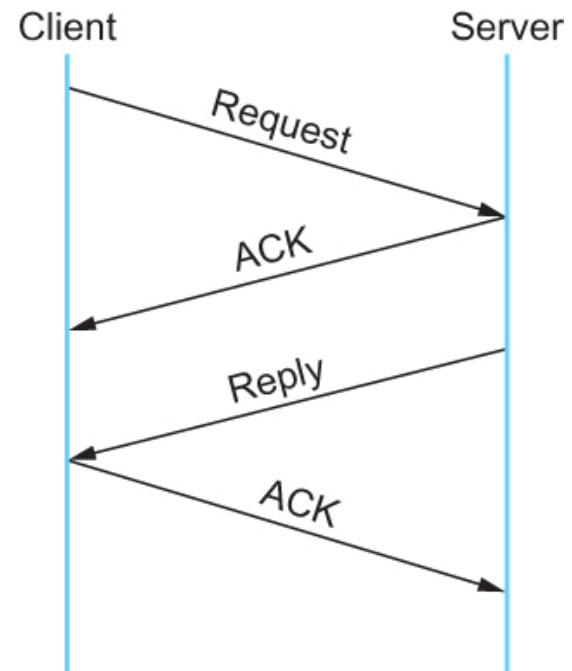


Remote Procedure Call (RPC)

- Two functions performed by any RPC protocol
 - Provide a **name space** to uniquely identify the procedure
 - Hierarchical approach naming structure;
 - Unique name.
 - **Match** each reply message to the request message
 - A message ID: Request and Reply IDs are the same;
 - Calling is blocked until a reply is received.

RPC – Optional Design

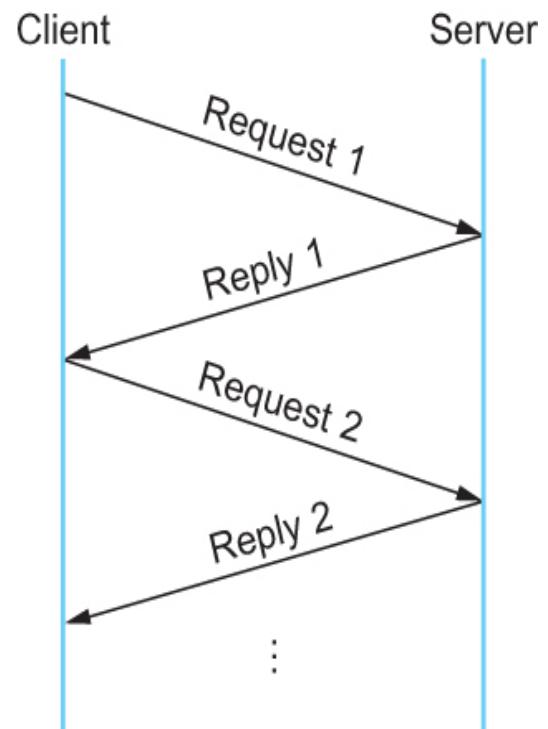
- RPC protocols perform additional functions
 - Provide reliable message delivery
 - Use ACKs and timeouts
 - Server ACKs the request
 - Client ACKs the reply
 - Support large packets through fragmentation and reassembly



RPC – Optional Design

■ Implicit Acknowledgement

- Caller receives a reply – this ACKs that the server received the request
- Server receives the next request indicates that the client received the reply

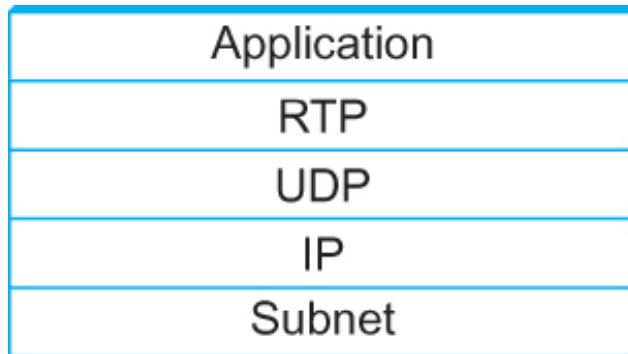


Transport for Real-Time Applications (RTP)

- Real Time applications
 - VoIP – human to human interaction
 - Multimedia – involves video, audio and data
 - Interactive applications – telephony, conferencing
 - Streaming applications – YouTube – lack human to human interaction – not as stringent as interactive applications
- Real time transport protocol concerns
 - Not too much reliability but continuity!

Transport for Real-Time Applications (RTP)

- Real Time Transport Protocol (RTP) can run over many lower-layer protocols
- RTP and UDP are both transport layer protocols
- RTP uses UDP to handle the demultiplexing (ports) of information



Protocol stack for RTP

Transport for Real-Time Applications (RTP)

■ Requirements

- Synchronization of audio and video streams
- Indication of packet loss –
 - Application needs to deal with missing packets
 - Application needs to know that packets are missing
 - Packet loss is indication of congestion – receiver needs to convey this information to the sender
- Frame boundary indication – different for different applications
- A more user-friendly identifier of senders – instead of ip address, use `user@domain.com`

Transport for Real-Time Applications (RTP)

- Requirements, cont'
 - Make reasonably efficient use of bandwidth
 - Don't want a long header
 - Reduce number of extra bits required
 - For voice, packets are short and long headers reduce the capacity for useful data

Transport for Real-Time Applications (RTP)

- RTP Design uses a pair of protocols
 - RTP
 - RTCP – Real-Time Transport Control Protocol
 - Used for passing control information about the data stream
- RTP and RTCP use consecutive port numbers
- RTCP provides three main functions
 - Feedback on the condition of the app. and the network
 - A way to synchronize different media streams
 - A way to provide the ID of the sender for display

Summary

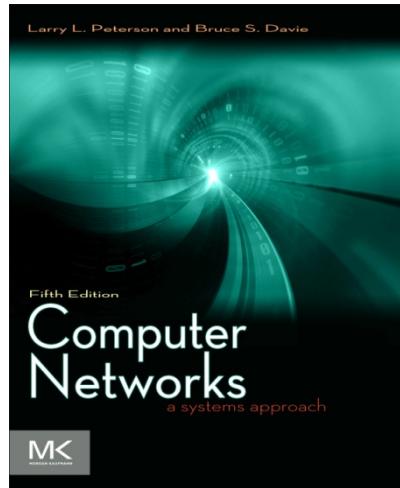
- Discussed how to convert host-to-host packet delivery service to **process-to-process** communication channel.
 - We have discussed UDP
 - We have discussed **TCP**
 - We have discussed **TCP congestion control/avoidance**
- Transport layer protocol alternatives: Remote Procedure Call (RPC) and Real-time Transport Protocol (RTP)

CPE348: Introduction to Computer Networks

Lecture #20: Chapter 6.1



Jianqing Liu
Assistant Professor of Electrical and Computer
Engineering, University of Alabama in Huntsville
jianqing.liu@uah.edu
<http://jianqingliu.net>



Chapter 6

Congestion Control and Resource Allocation

Problem

- We have seen protocols of Layer 1-4 to understand how data can be transferred among processes across networks;
- Problem: How to *effectively* and *fairly* allocate resources among a collection of network devices?

Chapter Outline

- Issues in Resource Allocation
- Queuing Disciplines
- Quality of Service
- TCP Congestion control/avoidance – **We talked it in previous lectures**

This chapter isn't about any protocols of a OSI layer, but an advanced topic in network design!

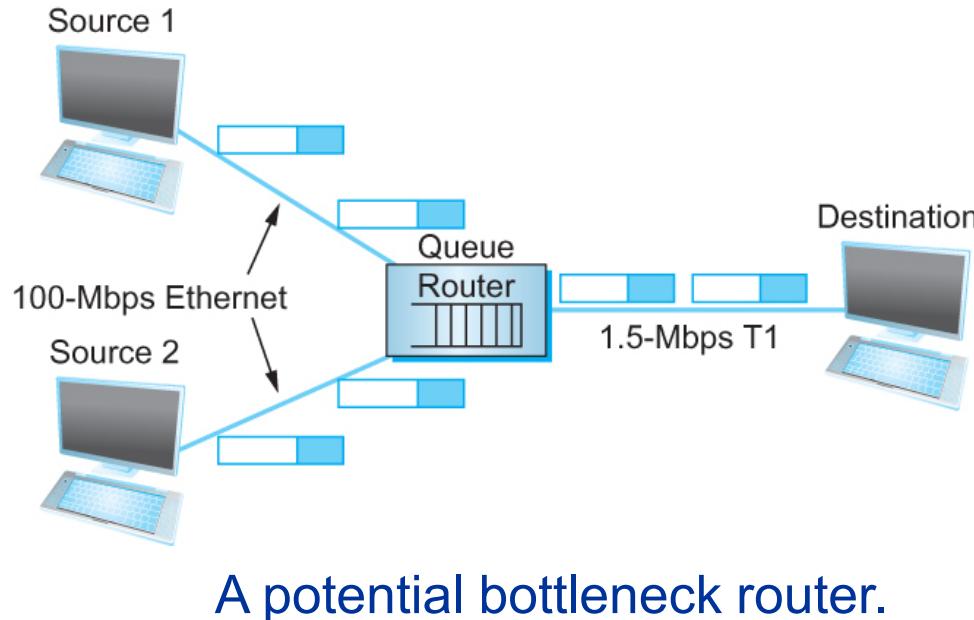
Resource Allocation

- What are network resources
 - Bandwidth of the links
 - Buffers at the routers and switches
 - ...
- Packets that are cached in a router's queue contend for the use of a link



One Example of Resource Allocation

■ Network Model - Packet Switched Network



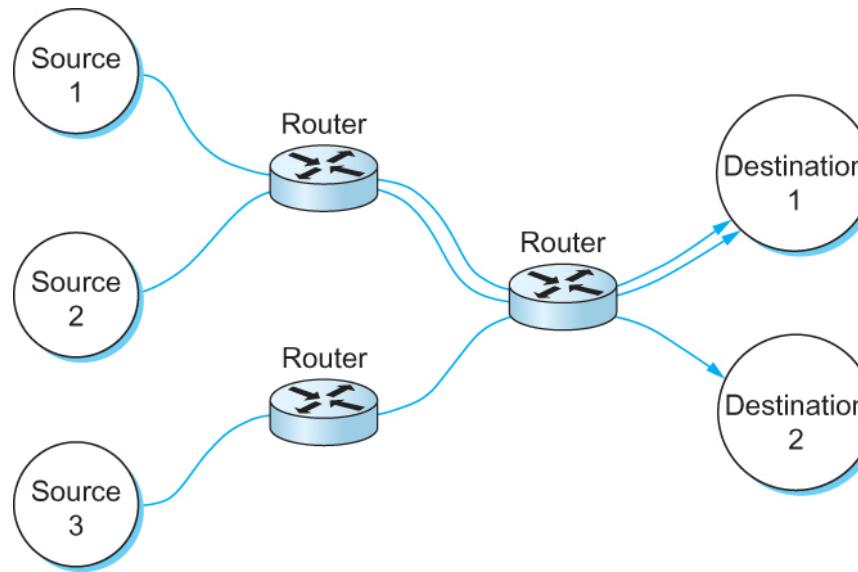
One Example of Resource Allocation

- What is a Flow?
 - A sequence of packets sent between a source / destination pair and following a route through the network
 - Important abstraction in the context of resource allocation



Issues in Resource Allocation

- Network Model
 - Connectionless Flows



Multiple flows passing through a set of routers

Issues in Resource Allocation

- Evaluation Criteria - Fair Resource Allocation
 - The **effective utilization** of network resources **is not the only** criterion for judging a resource allocation scheme.
 - We must also consider the issue of fairness. *However, what exactly constitutes fair resource allocation.*

What is fairness?

- An example
 - A video stream could receive 1 Mbps across some link
 - While a file transfer receives only 10 Kbps over the same link.

Issues in Resource Allocation

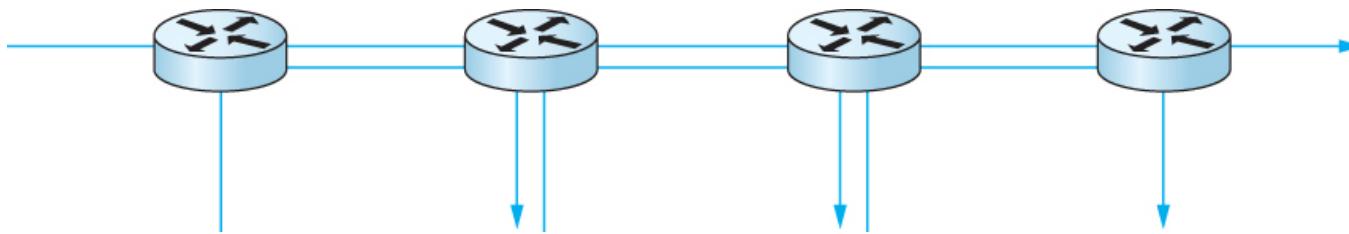
- Evaluation Criteria - Fair Resource Allocation
 - When several flows share a particular link, we would like for each flow to receive **an equal share of the bandwidth**.
 - This definition presumes that a *fair share of bandwidth means an equal share of bandwidth*.

But equal shares may not equate to fair shares!

- Should we also consider the length of the paths being compared? ([Example on next slide](#))

Issues in Resource Allocation

- Evaluation Criteria - Fair Resource Allocation



One four-hop flow competing with three one-hop flows

If we are considering path length, what is fair allocation for this situation?

Issues in Resource Allocation

■ Evaluation Criteria - Fair Resource Allocation

- Assuming that fair implies equal and that all paths are of equal length,
- Networking researcher Raj Jain proposed a metric that can be used to *quantify* the fairness of a congestion-control mechanism.
- Jain's fairness index is defined as follows: Given a set of flow throughputs (x_1, x_2, \dots, x_n) (measured in consistent units such as bits/second), the following function assigns a fairness index to the flows:

$$f(x_1, x_2, \dots, x_n) = \frac{(\sum_{i=1}^n x_i)^2}{n \sum_{i=1}^n x_i^2}$$

- The fairness index always results in a number between 0 and 1, with 1 representing greatest fairness.
- Indicates the fairness of the flows

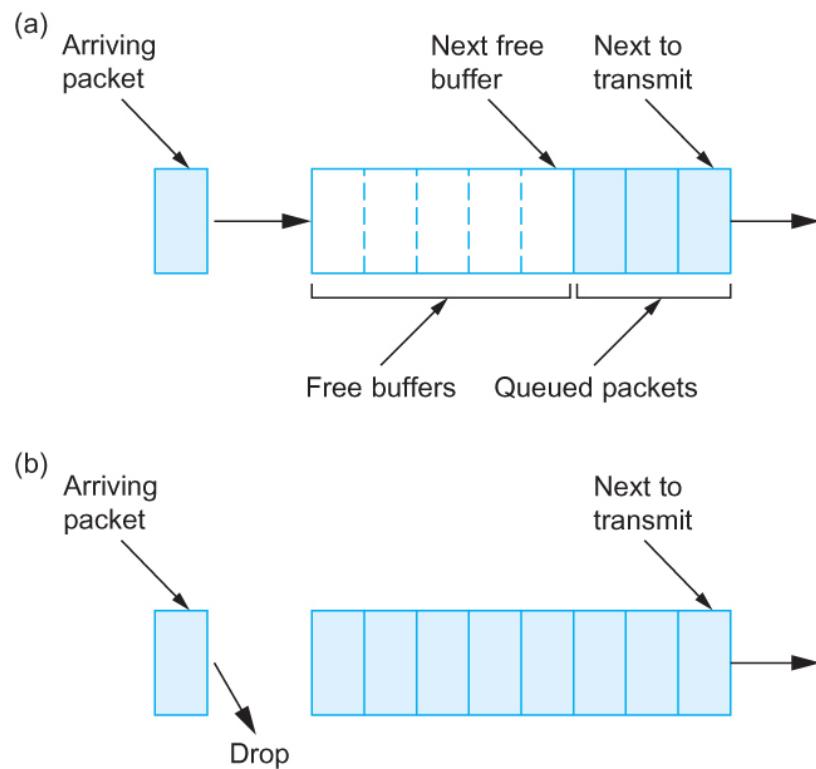
Queuing Disciplines

- **First in First out (FIFO)** queuing, also called first-come-first-served (FCFS) queuing – single queue
- **Priority Queuing** – utilizes FIFO queuing
- **Fair Queuing (FQ)** – each flow has its own queue
- **Weighted Fair Queuing (WFQ)** – variation on FQ
- *Scheduling Discipline* – determines order in which packets are transmitted
- *Drop Policy* – determines which packets are dropped

Queuing Disciplines

- FIFO queuing:
 - Simple to implement - the first packet that arrives is the first packet to be transmitted
 - The amount of buffer space at each router is finite, if a packet arrives and the queue (buffer space) is full, then the router discards that packet
 - Discarding is done without regard to the flow the packet belongs to
 - Discarding is done without regard to the importance of the packet
 - Called *tail drop*, since packets that arrive at the tail end of the FIFO queue are dropped
 - FIFO is a *scheduling discipline*—it determines the order in which packets are transmitted.
 - Tail drop is a *drop policy* — it determines which packets get dropped

Queuing Disciplines



(a) FIFO queuing; (b) tail drop at a FIFO queue.

Queuing Disciplines

- Priority Queuing
- A simple variation on basic FIFO queuing
- Each packet is marked with a priority; the mark could be carried, for example, in the IP header.
- The routers implement multiple FIFO queues, one for each priority class.
- Routers always transmit packets out of the highest-priority queue first. As higher priority queues are emptied, lower priority queue packets are transmitted
- Within each priority, packets are still managed in a FIFO manner.
- High-priority queue can dominate transmission time
- Users have to use self-control on setting the priorities

Queuing Disciplines

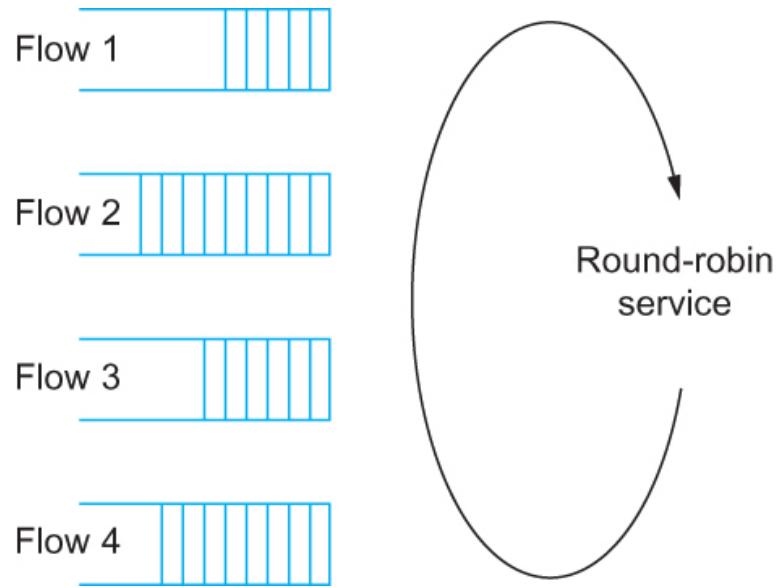
- FIFO queuing: Main problems with FIFO
 - FIFO queuing does not discriminate between different traffic sources
 - FIFO queuing does not separate packets according to the flow to which they belong.

Queuing Disciplines

- Fair Queuing – Addresses FIFO main problems
 - Fair queuing (FQ) maintains a separate queue for each flow currently being handled by the router.
 - Routers service these queues using a round-robin methodology.
 - Tail drop is used on each queue – prevents any given source from increasing its share of network capacity
 - FQ is designed to be used with end-to-end congestion control methods

Queuing Disciplines

■ Fair Queuing



Round-robin service of four flows at a router
Each flow is given an opportunity to have a
packet transmitted

Queuing Disciplines

- Fair Queuing – main complication
 - In Fair Queuing the packets being processed at a router are not necessarily the same length.
 - To truly allocate the bandwidth of the outgoing link in a fair manner packet length must be considered.
- A router is managing two flows using simple round-robin servicing
 - One with 1000-byte packets receives 2/3 of the link BW
 - The other with 500-byte packets receives 1/3 of the link BW

Queuing Disciplines

- Fair Queuing – solution to complication
 - What we really want is bit-by-bit round-robin; that is, the router transmits a bit from flow 1, then a bit from flow 2, and so on.
 - Clearly, it is not feasible to interleave the bits from different packets.
 - The FQ mechanism simulates bit-by-bit round-robin
 - First FQ determines when a given packet would finish transmitting based on bit-by-bit round-robin
 - This finish time is used to sequence the packets for transmission.

Queuing Disciplines

- Fair Queuing – single flow bit-by-bit round-robin
 - To understand the algorithm for approximating bit-by-bit round robin, consider the behavior of a single flow
 - For a single flow, let
 - P_i : denote the length of packet i (*in bits*)
 - S_i : time when the router starts to transmit packet i
 - F_i : time when router finishes transmitting packet i
 - Then, for a single flow $F_i = S_i + P_i$
 - P_i is expressed in how many clock ticks it takes to transmit packet i – one clock tick is the time it takes for a flow to get 1 bits worth of service(transmission time).

Queuing Disciplines

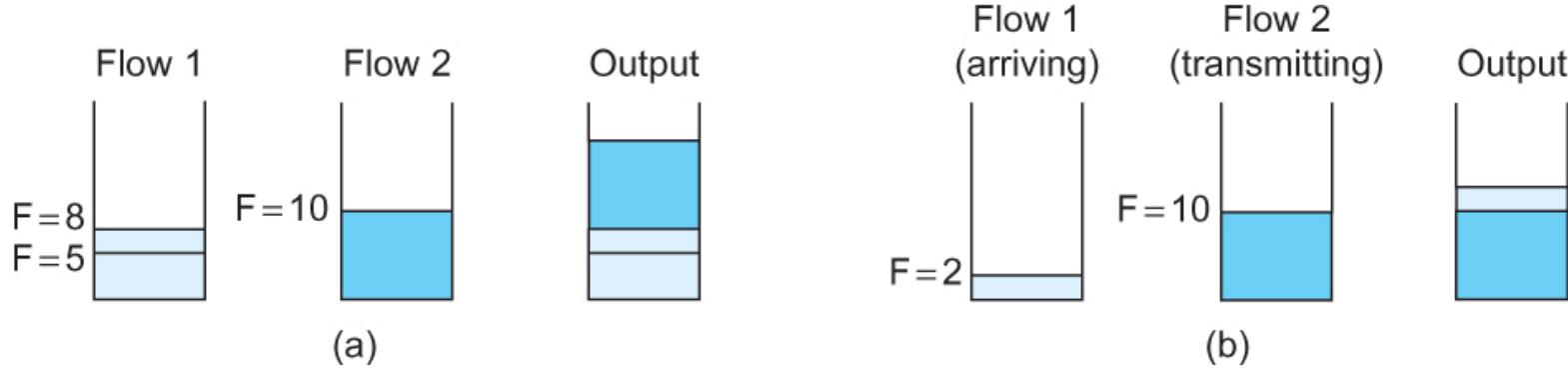
- Fair Queuing - single flow bit-by-bit round-robin
 - When do we start transmitting packet i ?
 - Depends on whether packet i arrived before or after the router finishes transmitting packet $i-1$ for the flow
 - Let A_i denote the time that packet i arrives at the router
 - Then $S_i = \max(F_{i-1}, A_i)$
 - So $F_i = S_i + P_i = \max(F_{i-1}, A_i) + P_i$

Queuing Disciplines

- Fair Queuing - multiple flow bit-by-bit round-robin
 - Now for every flow, calculate F_i for each packet that arrives using the formula $F_i = \max(F_{i-1}, A_i) + P_i$
 - For clocking A_i , advances one clock tick after each active flow transmits a bit
 - Then treat all the F_i as timestamps
 - Next packet to transmit is always the packet that has the lowest timestamp
 - The packet that should finish transmission before all others
 - New arriving packets cannot preempt a packet that is being transmitted
 - Shorter packets that arrive can jump ahead of longer packets waiting to be transmitted

Queuing Disciplines

■ Fair Queuing



Example of fair queuing in action: (a) packets with earlier finishing times are sent first; (b) sending of a packet already in progress is completed

Queuing Disciplines

- Weighted Fair Queuing (WFQ)
 - A weight is assigned to each flow(queue)
 - The weight indicates how many bits are transmitted each time – FQ assigns a value of 1 to all queues
 - Flows could be considered as different classes of traffic
 - WFQ is moving toward a reservation-based resource allocation
 - Now for every flow, calculate F_i for each packet that arrives using the formula $F_i = \max(F_{i-1}, A_i) + P_i/\text{weight}$

CPE348: Introduction to Computer Networks

Lecture #21: Chapter 6.2



Jianqing Liu
Assistant Professor of Electrical and Computer
Engineering, University of Alabama in Huntsville
jianqing.liu@uah.edu
<http://jianqingliu.net>

Quality of Service

- For many years, packet-switched networks have offered the promise of supporting multimedia applications, that is, those that combine audio, video, and data.
 - Once digitized, audio and video information become like any other form of data—a stream of bits to be transmitted.
 - One obstacle to the fulfillment of this promise has been the need for higher-bandwidth links.
- There is more to transmitting audio and video over a network than just providing sufficient bandwidth, however.

Quality of Service

- Participants in a telephone conversation, for example, expect to be able to converse in such a way that one person can respond to something said by the other and be heard almost immediately.
- Thus, the timeliness of delivery can be very important. We refer to applications that are sensitive to the timeliness of data as *real-time applications*.

Quality of Service

- Voice and video applications tend to be the general rule examples, but there are others such as industrial control—you would like a command sent to a robot arm to reach it before the arm crashes into something.
- Even file transfer applications can have timeliness constraints, such as a requirement that a database update complete overnight before the business that needs the data resumes on the next day.

Quality of Service

- The distinguishing characteristic of real-time applications is that they need some sort of assurance *from the network that data is likely to arrive on time (for some definition of “on time”).*
- Whereas a non-real-time application can use an end-to-end retransmission strategy to make sure that data arrives *correctly, such a strategy cannot provide timeliness.*
- This implies that the network will treat some packets differently from others—something that is not done in the best-effort model.
- A network that can provide these different levels of service is often said to support quality of service (QoS).

Quality of Service

- Taxonomy(classification) of Real-Time Applications
 - The first characteristic by which we can categorize applications is their tolerance of loss of data,
 - One lost audio sample can be interpolated from the surrounding samples with relatively little effect on the perceived audio quality.
 - A robot control program is likely to be an example of a real-time application that cannot tolerate loss—losing the packet that contains the command instructing the robot arm to stop is unacceptable.
 - real-time applications are categorized as *tolerant* or *intolerant* depending on whether they can tolerate occasional loss

Quality of Service

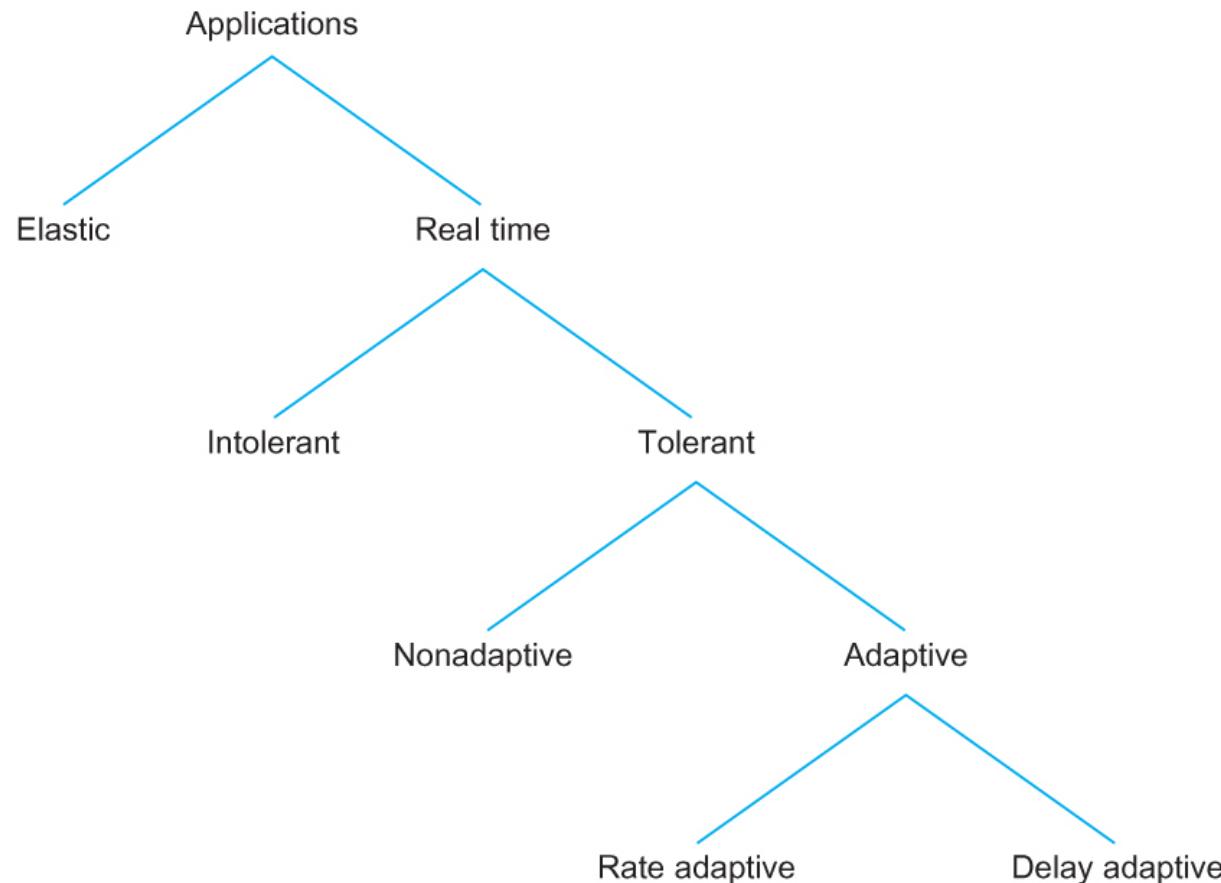
- Taxonomy of Real-Time Applications
 - A second way to characterize real-time applications is by their adaptability.
 - For example, an audio application might be able to adapt to the amount of delay that packets experience as they traverse the network.
 - We call applications that can adjust their playback point *delay-adaptive applications*.

Quality of Service

- Taxonomy of Real-Time Applications
 - Another class of adaptive applications are *rate adaptive*.
 - *For example, many* video coding algorithms can trade off bit rate versus quality.
 - Thus, if we find that the network can support a certain bandwidth, we can set our coding parameters accordingly.
 - If more bandwidth becomes available later, we can change parameters to increase the quality.

Quality of Service

■ Taxonomy of Real-Time Applications



Quality of Service

- Approaches to QoS Support
 - *fine-grained approaches, which provide QoS to individual applications or flows*
 - In this category we find “Integrated Services,” a QoS architecture developed in the IETF and often associated with RSVP (Resource Reservation Protocol).
 - *coarse-grained approaches, which provide QoS to large classes of data or aggregated traffic*
 - In this category lies “Differentiated Services,” which is probably the most widely deployed QoS mechanism.

Quality of Service

- Integrated Services (RSVP)
 - Service Classes
 - Guaranteed Service
 - The network should guarantee that the maximum delay that any packet will experience has some specified value
 - Controlled Load Service
 - The aim of the controlled load service is to emulate a lightly loaded network for those applications that request the service, even though the network as a whole may in fact be heavily loaded

Quality of Service

- Integrated Services (RSVP)
 - Overview of Mechanisms
 - Flowspec
 - With a best-effort service we can just tell the network where we want our packets to go and leave it at that, a real-time service involves telling the network something more about the type of service we require
 - The set of information that we provide to the network is referred to as a *flowspec*.
 - Admission Control
 - When we ask the network to provide us with a particular service, the network needs to decide if it can in fact provide that service. The process of deciding when to say no is called *admission control*.

Quality of Service

■ Integrated Services (RSVP)

- Overview of Mechanisms
 - Resource Reservation
 - Mechanism by which the users of the network and the components of the network itself exchange information such as requests for service, flowspecs, and admission control decisions. We refer to this process as *resource reservation*
 - Packet Scheduling
 - Finally, when flows and their requirements have been described, the network switches and routers need to meet the requirements of the flows.
 - A key part of meeting these requirements is managing the way packets are queued and scheduled for transmission in the switches and routers.
 - This last mechanism is *packet scheduling*.

Quality of Service

■ Differentiated Services

- Whereas the Integrated Services architecture allocates resources to individual flows,
- The Differentiated Services model (DiffServ for short) allocates resources to a small number of classes of traffic.
- In fact, some proposed approaches to DiffServ simply divide traffic into two classes.

Quality of Service

- Differentiated Services - Approach
 - Suppose that we have decided to enhance the best-effort service model by adding just one new class, which we'll call "premium."
 - Clearly we will need some way to figure out which packets are premium and which are regular old best effort.
 - Use the packets to identify themselves to the router when they arrive. This could obviously be done by using a bit in the packet header—if that bit is a 1, the packet is a premium packet; if it's a 0, the packet is best effort

Quality of Service

- Differentiated Services - Approach
 - If an identifying bit is used, there are two questions we need to address:
 - Who sets the premium bit, and under what circumstances?
 - What does a router do differently when it sees a packet with the bit set?

Quality of Service

- Differentiated Services
 - There are many possible answers to the first question, but a common approach is to set the bit at an administrative boundary.
 - For example, the router at the edge of an Internet service provider's network might set the bit for packets arriving on an interface that connects to a particular company's network.
 - The Internet service provider might do this because that company has paid for a higher level of service than best effort.

Quality of Service

- Differentiated Services
 - Assuming that packets have been marked in some way, what do the routers that encounter marked packets do with them?
 - Here again there are many answers. In fact, the IETF standardized a set of router behaviors to be applied to marked packets. These are called “per-hop behaviors” (PHBs), a term that indicates that they define the behavior of individual routers rather than end-to-end services

Quality of Service

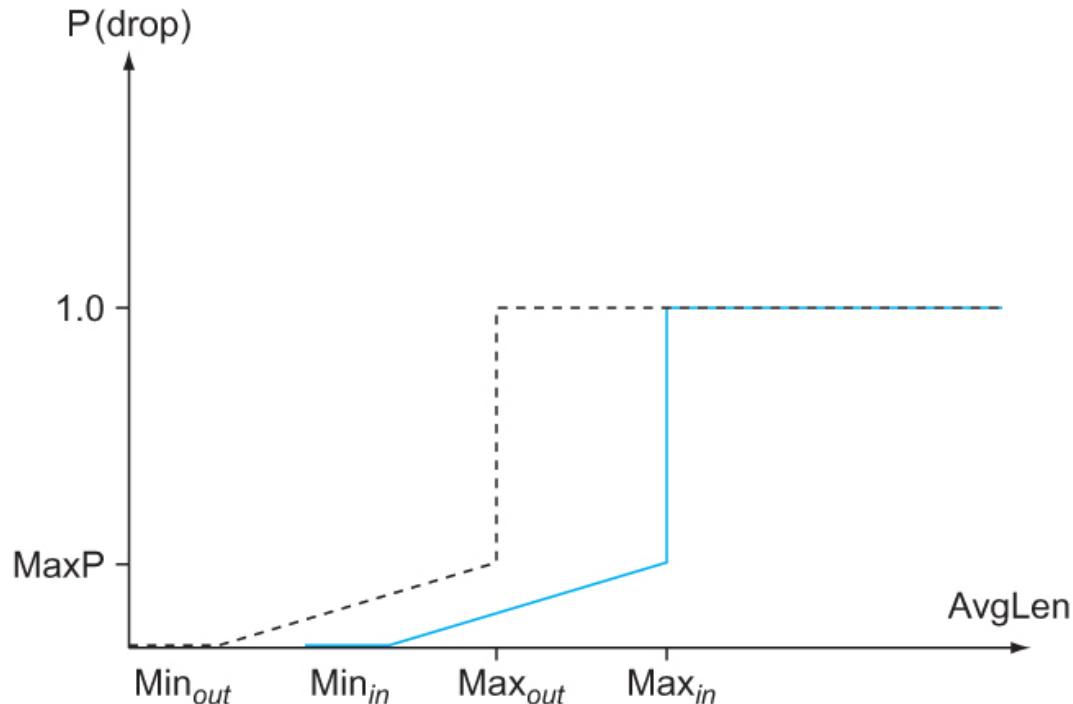
- Differentiated Services
 - The Expedited Forwarding (EF) PHB
 - One of the simplest PHBs to explain is known as “expedited forwarding” (EF). Packets marked for EF treatment should be forwarded by the router with minimal delay and loss.
 - The only way that a router can guarantee this to all EF packets is if the arrival rate of EF packets at the router is strictly limited to be less than the rate at which the router can forward EF packets.

Quality of Service

- Differentiated Services
 - The Assured Forwarding (AF) PHB
 - The “assured forwarding” (AF) PHB has its roots in an approach known as “RED with In and Out” (RIO) or “Weighted RED,” both of which are enhancements to the basic RED algorithm.
 - For our two classes of traffic, we have two separate drop probability curves. RIO calls the two classes “in” and “out”
 - The “out” curve has a lower MinThreshold than the “in” curve
 - Under low levels of congestion, only packets marked “out” will be discarded by the RED algorithm.
 - If the congestion becomes more serious, a higher percentage of “out” packets are dropped
 - If the average queue length exceeds Min_{in} , RED starts to drop “in” packets as well.

Quality of Service

- Differentiated Services
 - The Assured Forwarding (AF) PHB



RED with In and Out drop probabilities

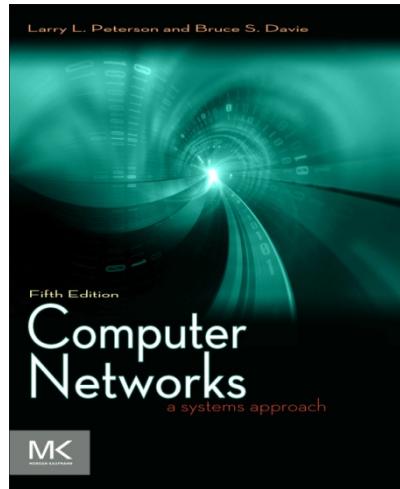
CPE348: Introduction to Computer Networks

Lecture #21: Chapter 8



Jianqing Liu
Assistant Professor of Electrical and Computer
Engineering, University of Alabama in Huntsville
jianqing.liu@uah.edu
<http://jianqingliu.net>

*Some slides are borrowed from Dr. Kevin Butler at UF



Chapter 8

Network Security

Cryptosystem

A cryptosystem is a 5-tuple consisting of

$$(E, D, M, K, C)$$

Where,

E is an *encryption* algorithm

D is an *decryption* algorithm

M is the set of *plaintexts*

K is the set of *keys*

C is the set of *ciphertexts*

Cryptosystem – key

- A key is an input to a cryptographic algorithm used to obtain confidentiality, integrity, authenticity or other property over some data.
 - ▶ The security of the cryptosystem often depends on keeping the key secret to some set of parties.
 - ▶ The *keyspace* is the set of all possible keys
 - ▶ *Entropy* is a measure of the variance in keys
 - typically measured in bits
- Keys are often stored in some secure place:
 - ▶ passwords, on disk keyrings, ...
 - ▶ TPM, secure co-processor, smartcards, ...
- ... and sometimes not, e.g., certificates

Cryptosystem – algorithm

- Algorithm used to make content unreadable by all but the intended receivers

$E(key, plaintext) = ciphertext$

$D(key, ciphertext) = plaintext$

- Algorithm is public, key is private*
- Block vs. Stream Ciphers
 - Block: input is fixed blocks of same length
 - Stream: stream of input

Cryptosystem – hardness

- Functions
 - ▶ Plaintext P
 - ▶ Ciphertext C
 - ▶ Encryption (E) key k_e
 - ▶ Decryption (D) key k_d

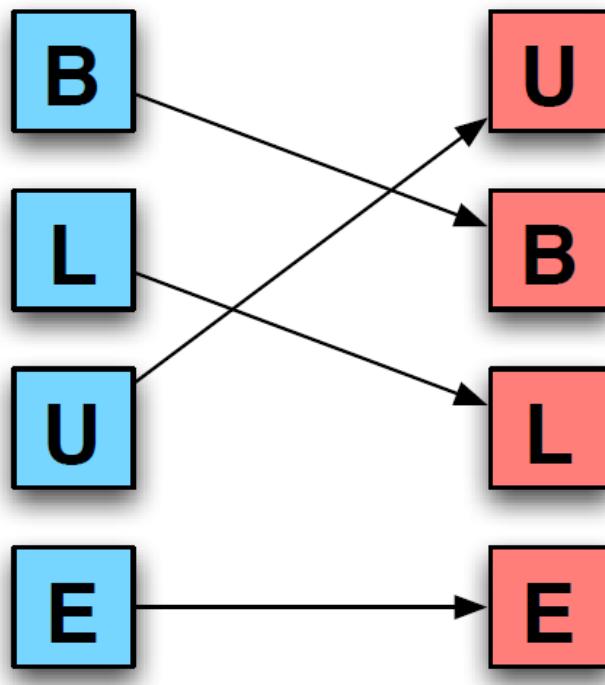


$$D(E(P, k_e), k_d) = P$$

- Computing P from C is hard, computing P from C with k_d
 - ▶ Is easy for all P s (operation true for all inputs) ...
 - ▶ ... except in some vanishingly small number of cases

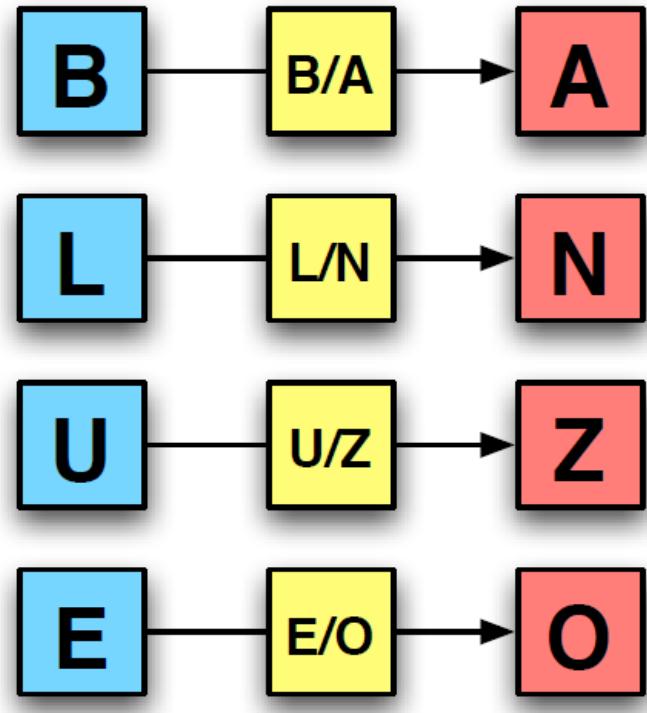
History of Cryptography – Crypto 0.0

- Transposition Ciphers



History of Cryptography – Crypto 0.0

- Substitution Ciphers



History of Cryptography – Crypto 1.0

- **Crypto 1.0 concerns**
 - encryption and authentication of data,
 - during communication and storage/retrieval
- **Crypto 1.0 primitives:**
 - Symmetric (secret key):
 - Stream/block ciphers
 - Message authentication codes
 - Asymmetric (public key):
 - Public-key encryption
 - Digital signatures
 - Key-exchange protocols
 - Keyless:
 - Cryptographic hash functions

History of Cryptography – Crypto 1.0

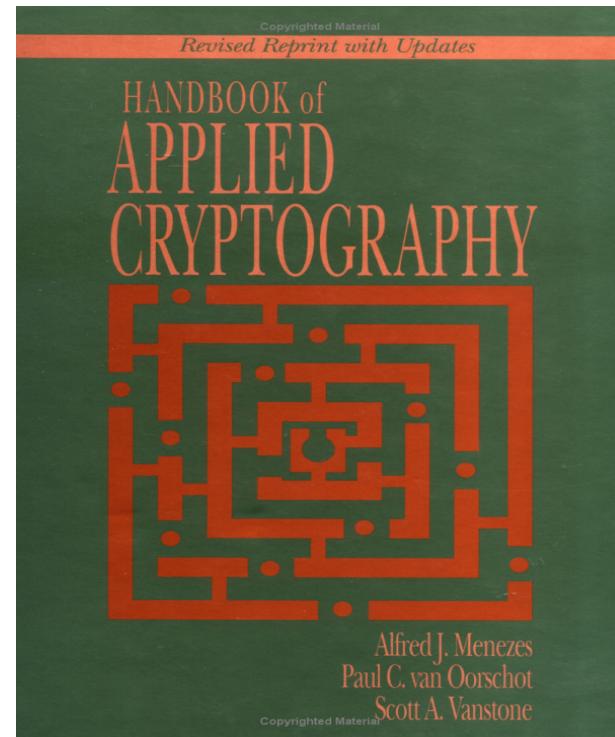
- <http://cacr.uwaterloo.ca/hac/>
- **Handbook of Applied Cryptography**
- **By Alfred J. Menezes, Paul C. van Oorschot and Scott A. Vanstone**

CRC Press

ISBN: 0-8493-8523-7

October 1996, 816 pages

Fifth Printing (August 2001)



History of Cryptography – Crypto 1.0

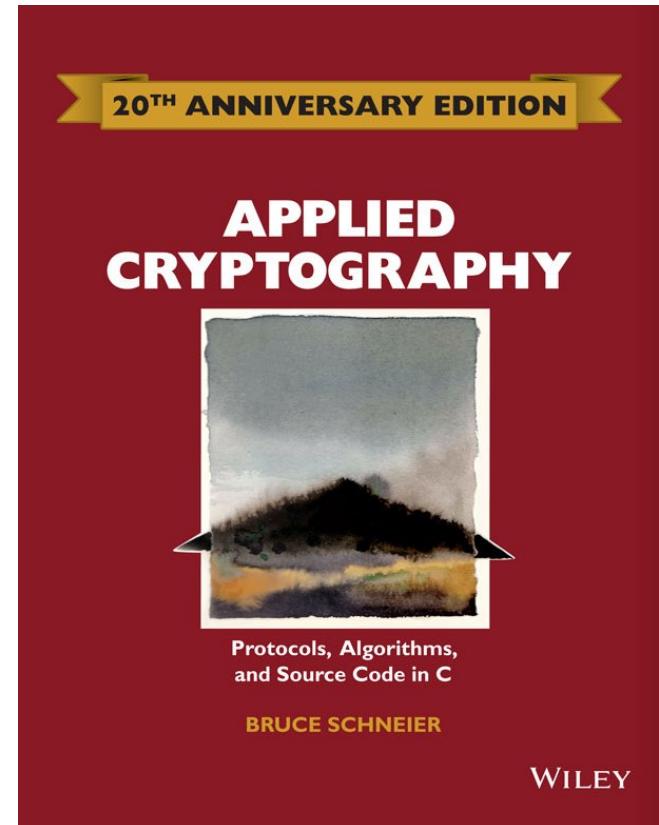
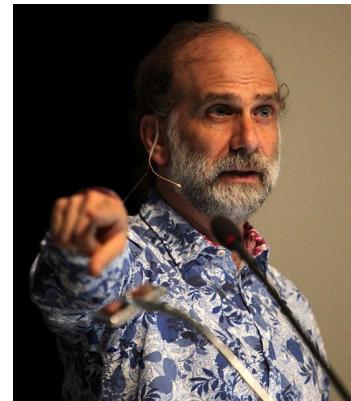
- Ch. 1 - Overview of Cryptography
- Ch. 2 - Mathematics Background
- Ch. 3 - Number-Theoretic Reference Problems
- Ch. 4 - Public-Key Parameters
- Ch. 5 - Pseudorandom Bits and Sequences
- Ch. 6 - Stream Ciphers
- Ch. 7 - Block Ciphers
- Ch. 8 - Public-Key Encryption
- Ch. 9 - Hash Functions and Data Integrity
- Ch. 10 - Identification and Entity Authentication
- Ch. 11 - Digital Signatures
- Ch. 12 - Key Establishment Protocols
- Ch. 13 - Key Management Techniques
- Ch. 14 - Efficient Implementation
- Ch. 15 - Patents and Standards



History of Cryptography – Crypto 1.0

- Applied Cryptography: Protocols, Algorithms, and Source Code in C
- By *Bruce Schneier*

John Wiley & Sons
ISBN 978-1-119-09672-6
1996, 784 Pages

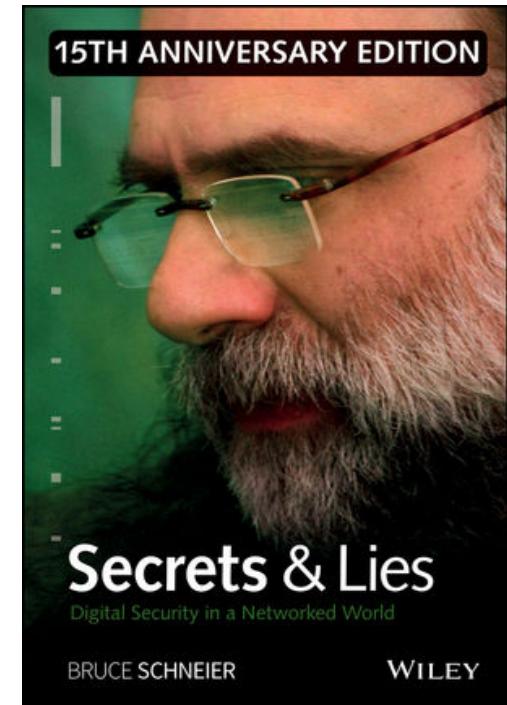


History of Cryptography – Crypto 1.0

- “A colleague once told me that the world was full of bad security systems designed by people who read *Applied Cryptography*.”

--- *Bruce Schneier*

- So, please be extra careful!



History of Cryptography – Crypto 2.0

- **Crypto 2.0 additionally concerns**
 - computing with encrypted data,
 - partial information release of data,
 - hiding identity of data owners or any link with them.
- **Crypto 2.0 primitives:**
 - homomorphic encryption
 - secret sharing
 - blind signatures
 - oblivious transfer
 - zero-knowledge proofs
 - secure two/multi-party computation
 - functional encryption
 - indistinguishable obfuscation

History of Cryptography – Crypto 2.0

- No books or graduate courses cover Crypto 2.0
- Only research papers, papers, papers
- A huge gap...



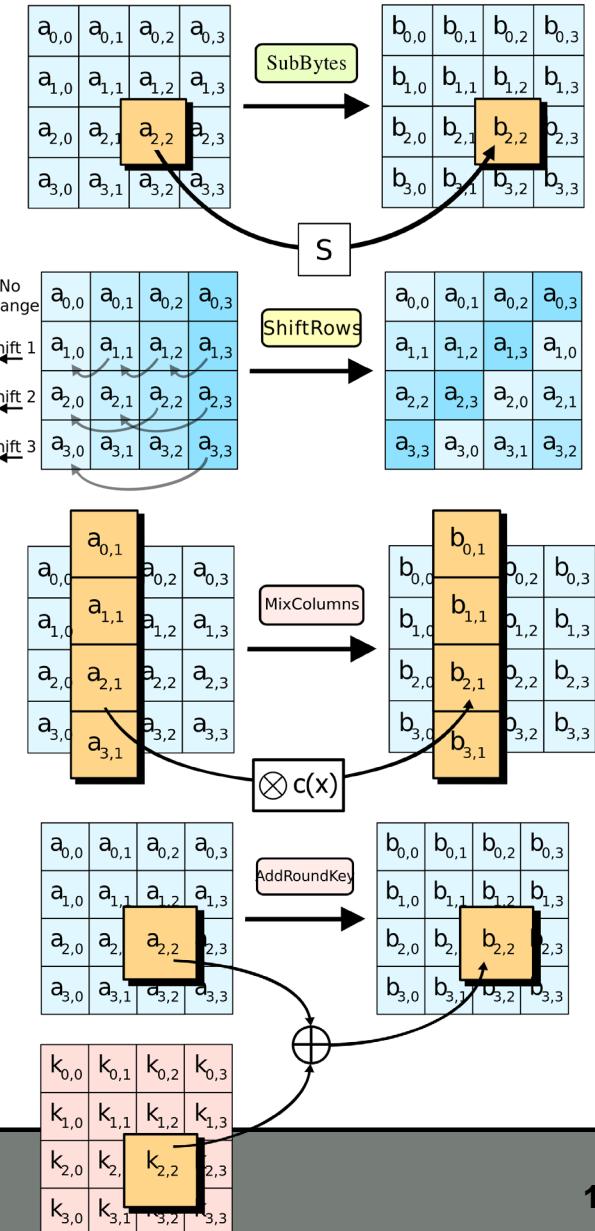
This lecture

We will only talk about several Crypto 1.0 techniques...

...and some applied crypto in computer networks!

Advanced Encryption Standard (AES)

- ▶ Rijndael (pronounced “Rhine-dall”)
- ▶ Currently implemented in many devices and software, but there are still DES holdouts
- ▶ AES takes 128, 192 or 256 bit keys;
- ▶ AES repeats many rounds (10,12,14) of transformation (a.k.a., substitution-permutation network) to encrypt the plaintext.



Hash Function

- Hash algorithm
 - Compression of data into a hash value
 - E.g., $h(d) = \text{parity}(d)$
 - Such algorithms are generally useful in systems (speed/space optimization)
- ... as used in cryptosystems
 - ▶ *One-way* - (computationally) hard to **invert** $h()$, i.e., compute $h^{-1}(y)$, where $y=h(d)$
 - ▶ *Collision resistant* hard to find two data x_1 and x_2 such that $h(x_1) == h(x_2)$



Hash Function

- How do you design a “strong cryptographic hash function?”
- No formal basis
 - ▶ Concern is backdoors
- MD2, MD4, MD5 (128bit):
 - ▶ Broken, Broken, Broken
 - ▶ MD4, MD5: Similar, but complex functions in multiple passes
- SHA-1 (160 bit)
 - ▶ “Complicated function”
 - ▶ Theoretical weaknesses
- SHA-2 (224, 256, 384 or 512-bit)
- SHA-3 (224, 256, 384 or 512-bit)

Basic truths of cryptography...

- Cryptography is not frequently the source of security problems
 - ▶ Algorithms are well known and widely studied
 - Use of crypto commonly is ... (e.g., WEP)
 - ▶ Vetted through crypto community
 - ▶ Avoid any “proprietary” encryption
 - ▶ Claims of “new technology” or “perfect security” are almost assuredly *snake oil*



Common issues that lead to pitfalls

- Generating randomness
- Storage of secret keys
- Virtual memory (pages secrets onto disk)
- Protocol interactions
- Poor user interface
- Poor choice of key length, prime length, using parameters from one algorithm in another

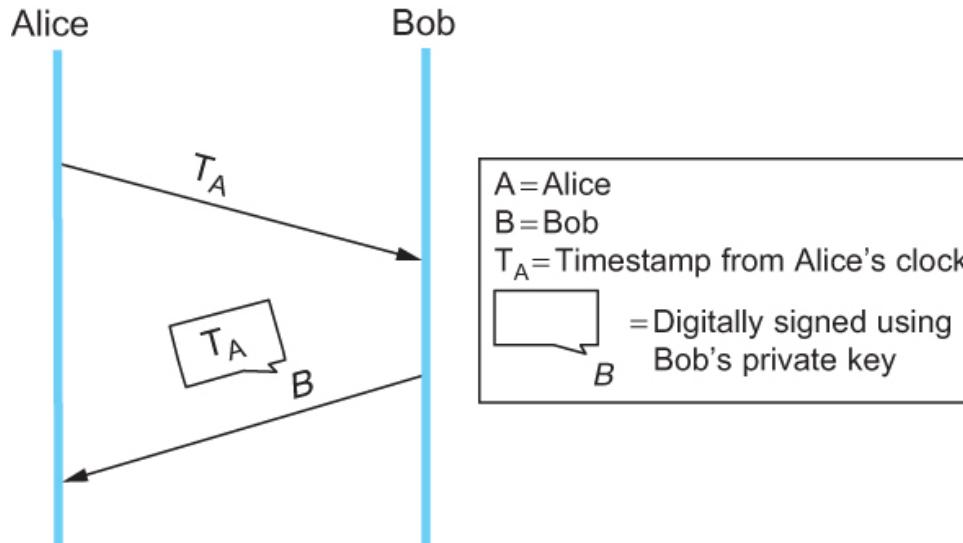


Important Principles

- Don't design your own crypto algorithm
 - ▶ Use standards whenever possible
- Make sure you understand parameter choices
- Make sure you understand algorithm interactions
 - ▶ E.g. the order of encryption and authentication
 - Turns out that authenticate then encrypt is risky
- Be open with your design
 - ▶ Solicit feedback
 - ▶ Use open algorithms and protocols
 - ▶ Open code? (jury is still out)

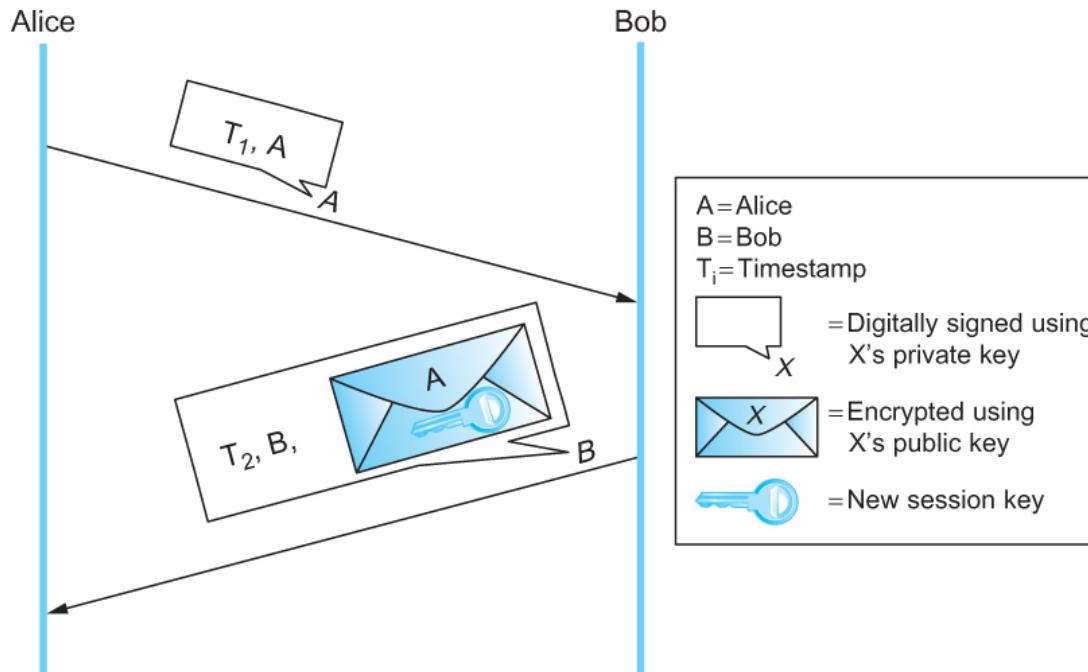
Network Authentication Protocols

- Originality and Timeliness
 - Challenge-response Protocol



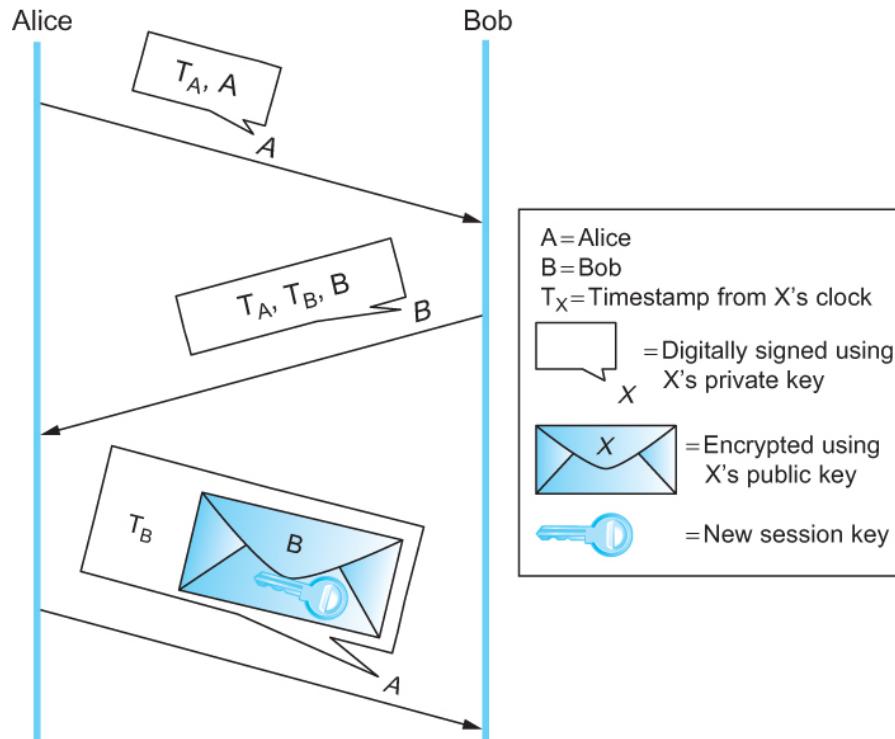
Network Authentication Protocols

■ Public Key Authentication Protocols



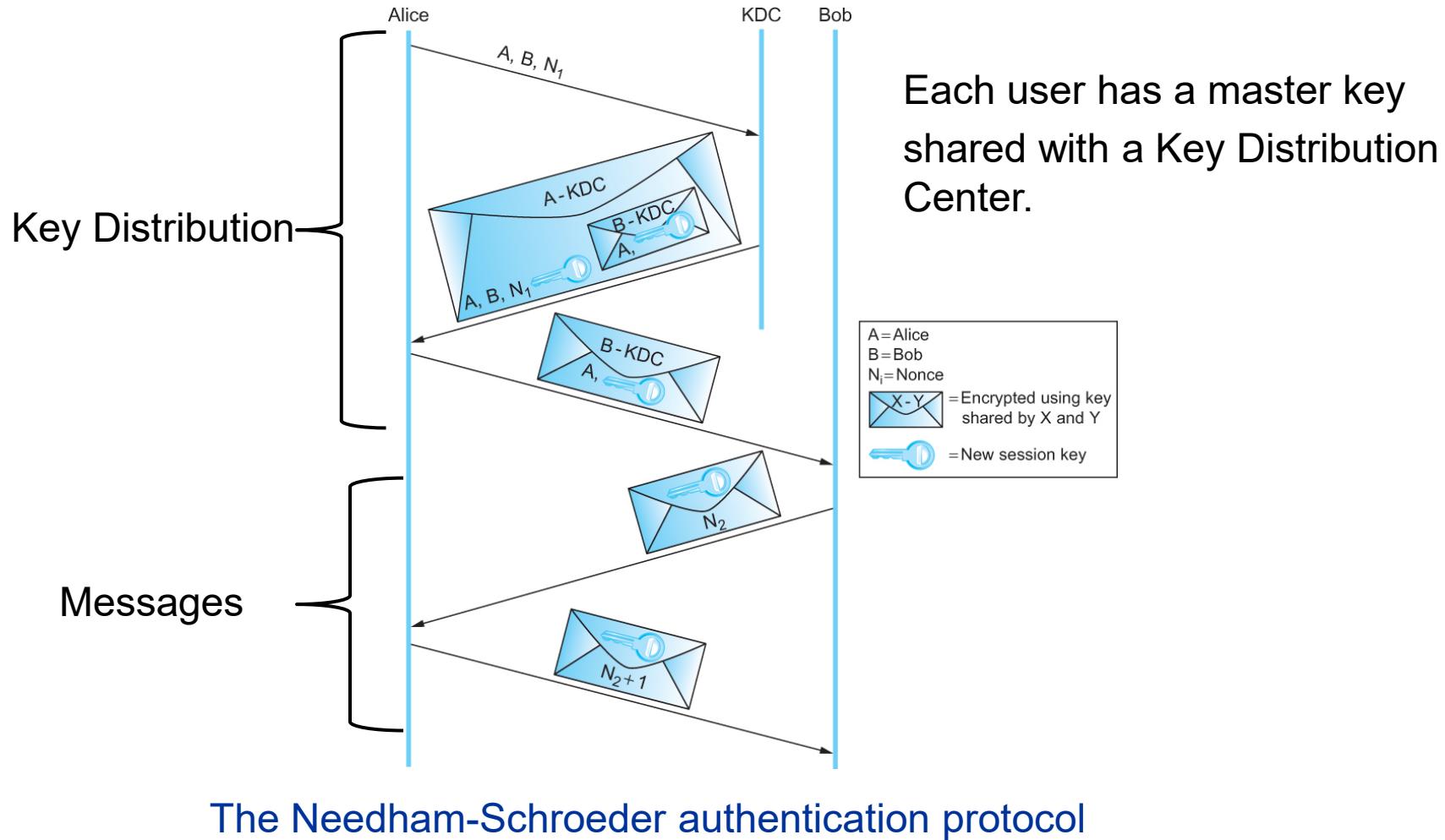
Network Authentication Protocols

■ Public Key Authentication Protocols



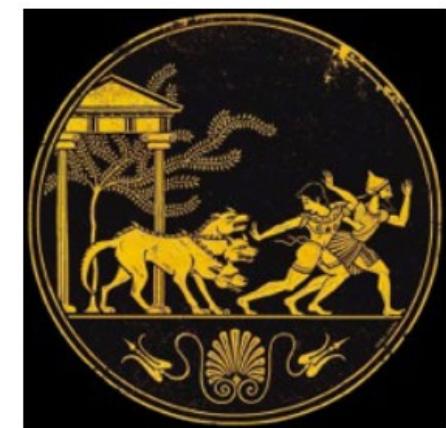
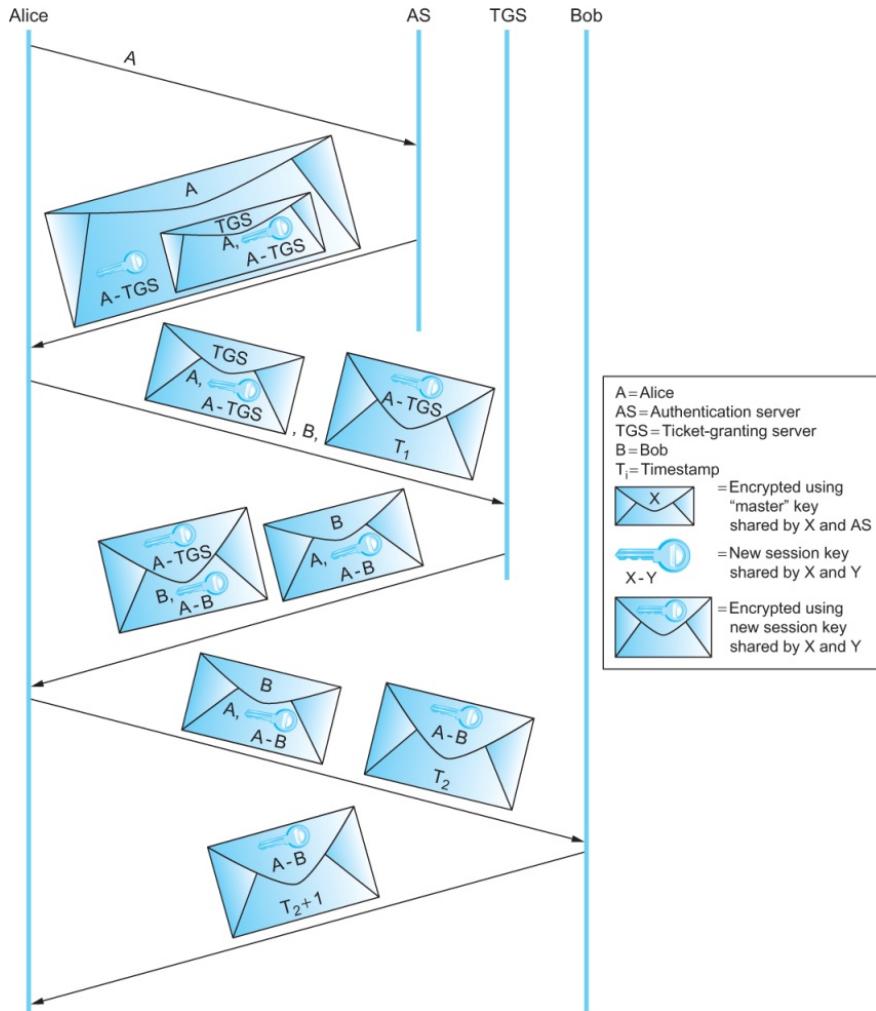
Network Authentication Protocols

■ Symmetric Key Authentication Protocols



Authentication Protocols

■ Symmetric Key Authentication Protocols - Kerberos



Summary

- General idea of cryptosystem
 - History
 - Building blocks
- Overview of a couple of crypto techniques
 - AES
 - Hash function
- One network authentication protocol
 - Kerberos

CPE348: Introduction to Computer Networks

Lecture #1: Introduction



Jianqing Liu
Assistant Professor of Electrical and Computer
Engineering, University of Alabama in Huntsville
jianqing.liu@uah.edu
<http://jianqingliu.net>

What is networking?

Scenario 1

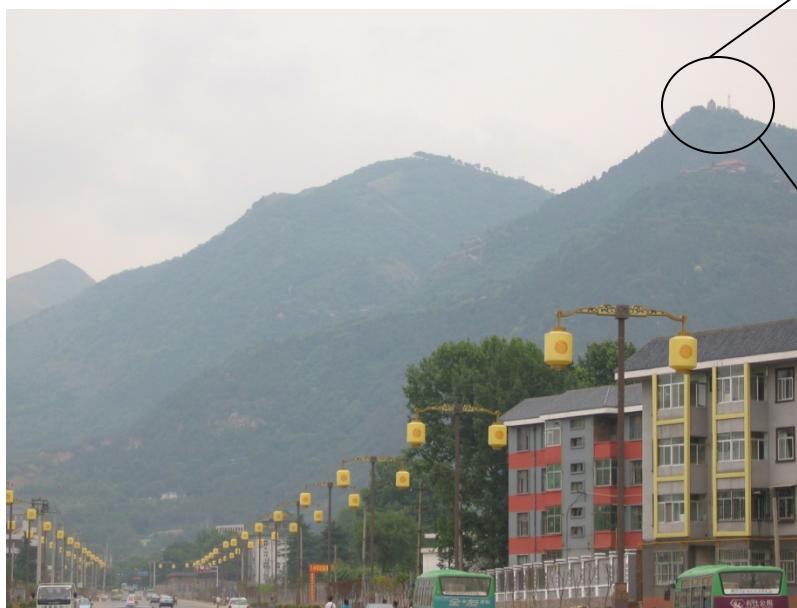
- Xi'an, China, the ancient capital for a few dynasties in China



Terracotta Army Pit

Scenario 1

- An old Chinese Story in Zhou Dynasty (“烽烟戏诸侯，一笑失天下”)
- Zhou ([1122 BC](#) to [256 BC](#))



Scenario 1



Protocol: smoke-on indicates enemy attack, so allies would come to rescue

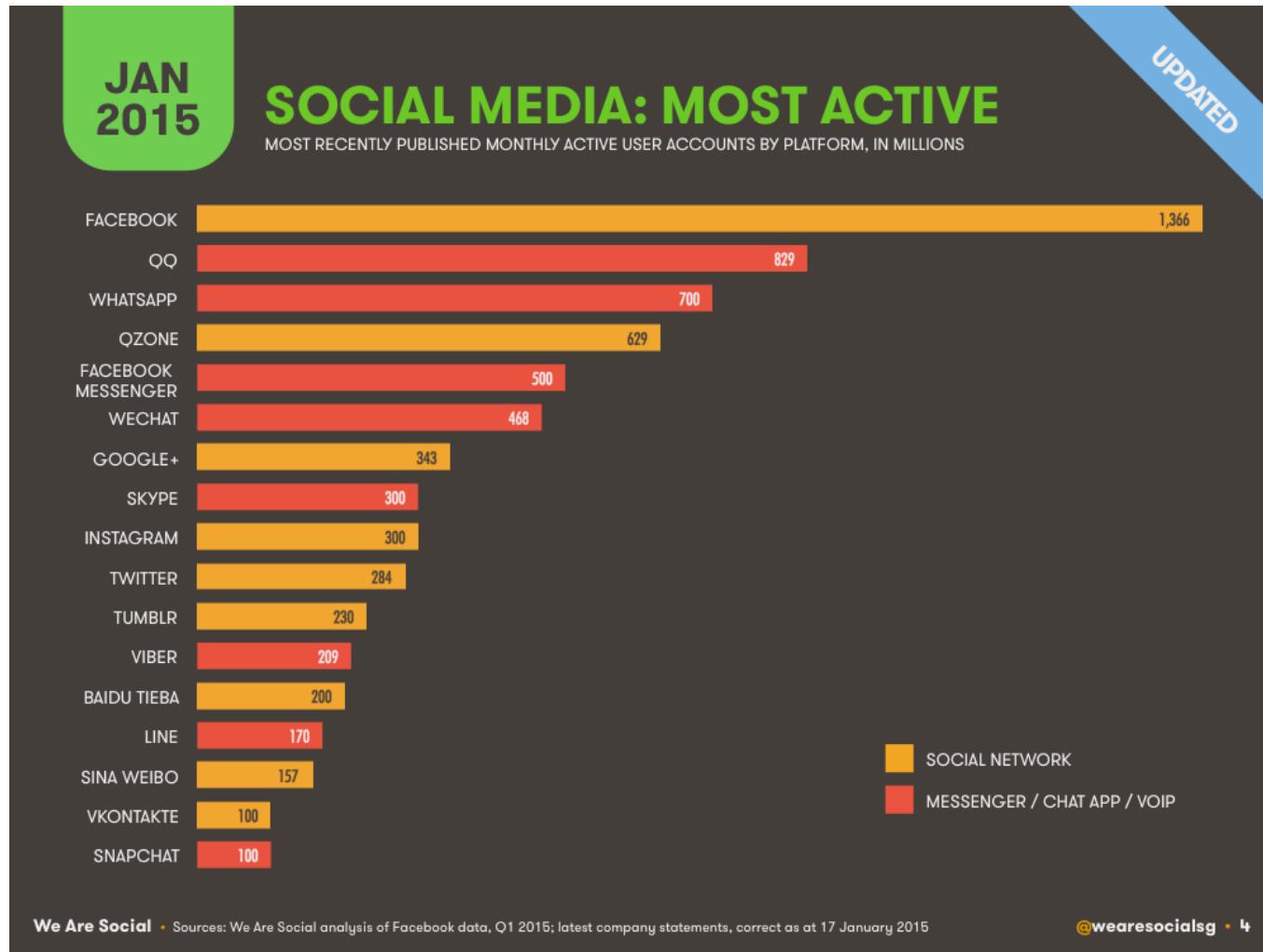
Emperor used this to please his concubine by lighting the smoke even if there is no attack

Protocol failure!

Scenario 2



Social media popularity



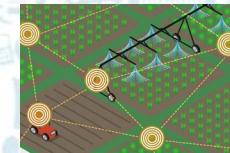
A more connected world



Smart Health



Smart Transportation



Smart Industry



Smart Energy

Impact of Networking

- Good
 - Improve working efficiency
 - Improve people's quality of life (QoL)
 - Improve society's connectivity
 - ...We cannot live without networks (Internet)
- Bad
 - Waste too much time (addiction to cyberspace)
 - Invade people's privacy (The Emperor's New Cloths)

The good (the innocent)...

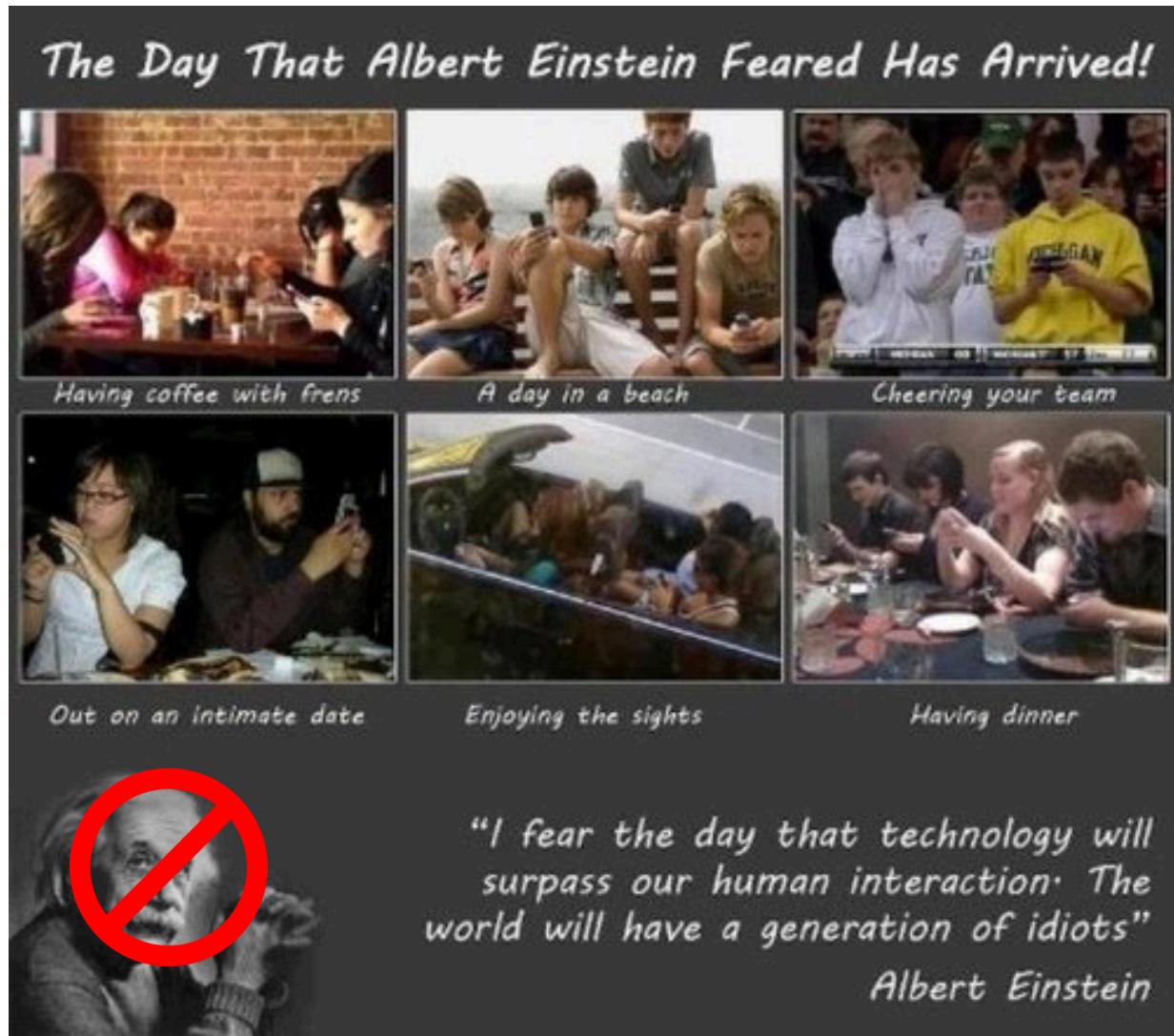
St. Peter's Square



“How I met your mother”



More fun in cyberspace?



We are in the cyberspace age...

- We create it and we have to live with it...
- How did we get here?

Something we love to have, something we hate to have, something we cannot live without...

...we can only tell part of the story in this course, here it is...

What is about this course?

Overview

- **What?** computer network vs. communication network: --- an **interconnected** collection of **autonomous** computers to accomplish the task of **information exchange** among these computers
- Computers → communications devices
- Internet → NGI (Next Generation Internet), Internet2, vBNS, all-optical, wireless Internet, NSF GENI: clean slate design, OpenFlow, SDN...
- Globalization: Optical & Wireless

The essence of computer communications...

The fundamental problem of communication is that of reproducing at one point either exactly or approximately a message selected at another point.

([Claude Shannon](#), *The Mathematical Theory of Communication*)



Overview (cont.)

- Why? An integral part of life
- the idea of teamwork: resource sharing (grid computing, cloud computing...)
- information exchange: be informed
- fast communications: be updated
- convenience: mobile (anywhere anytime connected...)
- money making: E-commerce/m-commerce
- health care: elderly care/smart aging, health monitoring, health social networks...
- infotainment (life killing or life enriching): entertainment (Internet gaming, blogging, social networking...)
- more...

Overview (cont.)

- [History](#)
 - smoke signals --- story of Zhou dynasty (scenario 1)
 - telephony: Alexander Graham Bell (the “decibel”)
 - telegraphy: Morse code
 - 1950s: connecting central computer to terminals
 - 1970s: Aloha systems for packet radios
 - DARPA projects: robust communication network design
 - ARPANET and Tymnet

Overview (cont.)

- History (cont.)
 - Internet (1980s)
 - World Wide Web (1990s)
 - Internet telephony
 - digital age: paradigm shifts
 - wireless Internet / Ubinet: Ubiquitous Networking
 - any one, anywhere, anytime, any form, with flexible data rate--future generation telecommunication networks!
 - xG networks (2000s)...
- Telecomm bust! (2000s)...Telecomm will come back! ...
It starts to come back! ... It's coming back!...It is
booming again...

Nutshell “Network” Design

- “Wire them together” to form a network
 - Need to hook TWO devices first (direct link): transmission media between two (physical layer)
 - Two should understand each other (protocols)
 - Two should communicate efficiently (efficiency)
 - Multiple users sharing the same “wire” should coordinate (MAC)
 - Two “far away” should be able to communicate (routing and transport)
 - Access any information we desire (applications)
 - Protect it and what is on it (security)

General design tasks

- Need to address the point-to-point (P2P) data transfer (PHY layer)
- Guarantee the reliable transfer P2P (data link layer)
- Coordinate the efficient use of a link if shared (MAC)
- Design more efficient routing schemes (efficiency in terms of overheads, power saving or other resource) if two points are not directly connected (routing)
- Need to assure the reliability end-to-end (e2e) (transport)
- Guarantee the efficient data transfer across a network (flow & congestion control)
- Protect the network infrastructure and information content

Network scalability

- Hierarchical network topology
 - Clustering ideas: a group of nodes form a network, multiple groups form another high layer network, and so on, lab-dept-university-city-state-national-international-universe
 - Backbone: high-speed links connect multiple networks
 - optical fiber
 - cable
 - satellite
 - Last mile: reaching the customers (first mile...)

Network classification

- Network classification
 - Based on geography
 - LAN: Local Area Networks
 - MAN: Metropolitan Area Networks
 - WAN: Wide Area Networks
 - Based on transmission media
 - wired networks
 - wireless networks

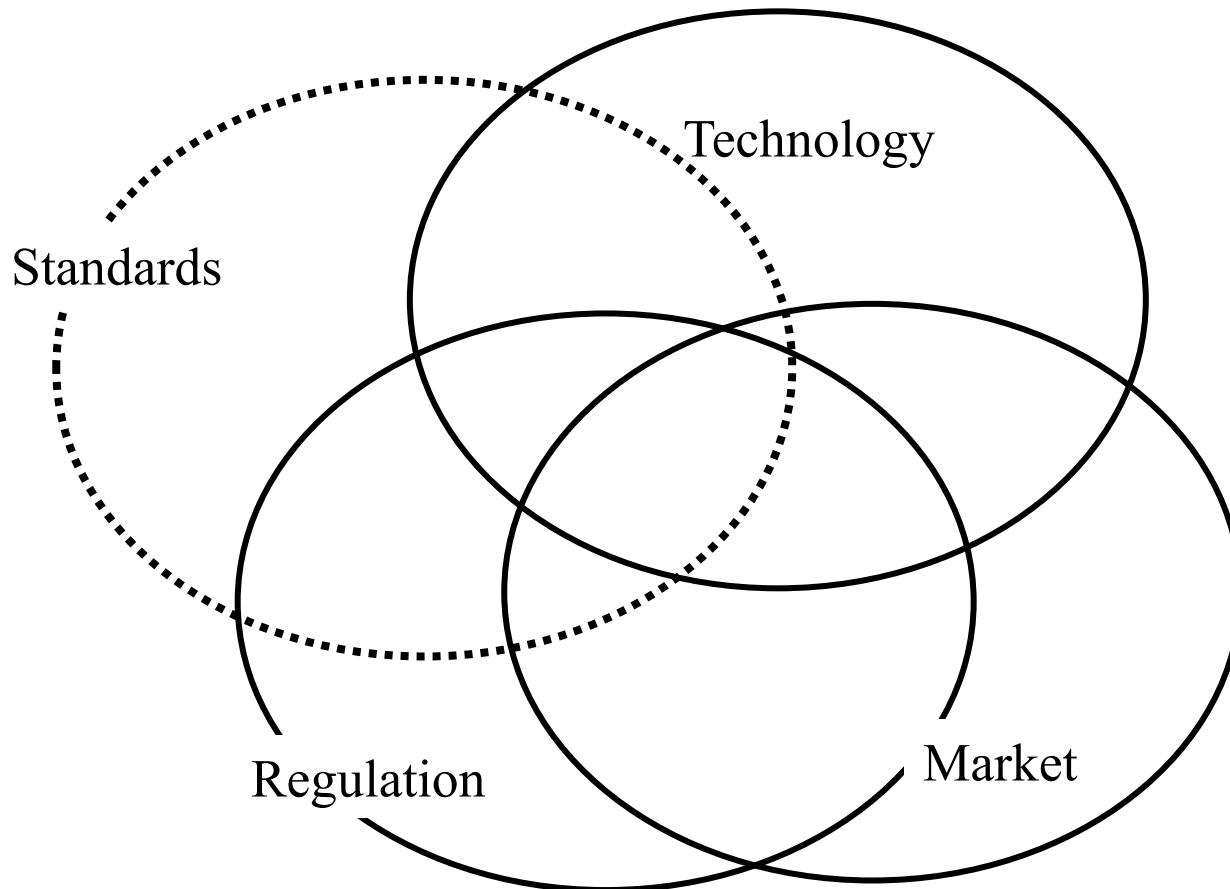
Network classification

- Network classification (cont.)
 - Based on information content
 - telephony
 - data networks
 - multimedia networks
 - Based on communication technology
 - analog networks
 - digital networks

Network design

- Design methodology
 - Top-down: breaking down design task into different design subtasks
 - Bottom-up: finishing all subtasks and assembling all into one

Key factors to network success...



Further reading...

- Exercises: web search
 - Find who invents WWW
 - Find who wrote the TCP/IP
 - Determine who is the real founding father of Internet
 - Find what mobile social networks are
 - What are mobile Healthcare or wireless healthcare systems?
 - Check out this: <http://gma.yahoo.com/video/news-alan-dershowitz-young-people-080000867.html>

More about this course...

Goals

Introduction to the basic computer network concepts and underlying technologies including

1. Local Area Networks (LANs), Ethernet, Internet;
2. MAC, TCP/IP and Application Layer Protocols;
3. Socket Programming and Network Security.

It's a big field, so we have
to focus on just a few topics.

Basics

- Students are expected to attend all lectures;
- Course videos will be recorded on Panapto;
- No late homework, Canvas submission, hardcopy;
- No makeup exams without a written excuse, and no makeup exams for the Final;

So.... don't miss exams or homework

Grade

Homework 10%

- 5 homework assignments: 2% each

Project 20%

- PA #1: 8% backoff protocol simulation
- PA #2: 4% CRC & IP Checksum calculation
- PA #3: 8% Wireshark TCP packet analysis

Midterm exam 40%

- 2 midterm exam: 20% each (online, close books/notes)

Final exam 30%

- Comprehensive, close books/notes, April 28th

Grade

Letter grade scale

- >90%: **A**
- 80-90%: **B**
- 70-80%: **C**
- 60-70%: **D**
- <60%: **F**

Academic Honesty

Students enrolled in this course are expected to conform to the UAH policies concerning academic misconduct as outlined in the UAH Student Handbook at

<http://www.uah.edu/student-support/student-conduct/handbook>

- Collaboration on exams or laboratory assignments will not be permitted and will be considered cheating. All work submitted for a grade must be entirely your own, including the laboratory programming assignments.
- **Students who cheat will be reported to the University Judicial Officer and will receive no credit (0) for that exam or assignment.**

Logistics

TA

- Siddharth Sankar Das
- Email: sd0064@uah.edu

Instructor's Office hour

- MW 4:30-6:00PM
- ZOOM: 596-759-0587

Thanks
Be safe and have a great semester!!!