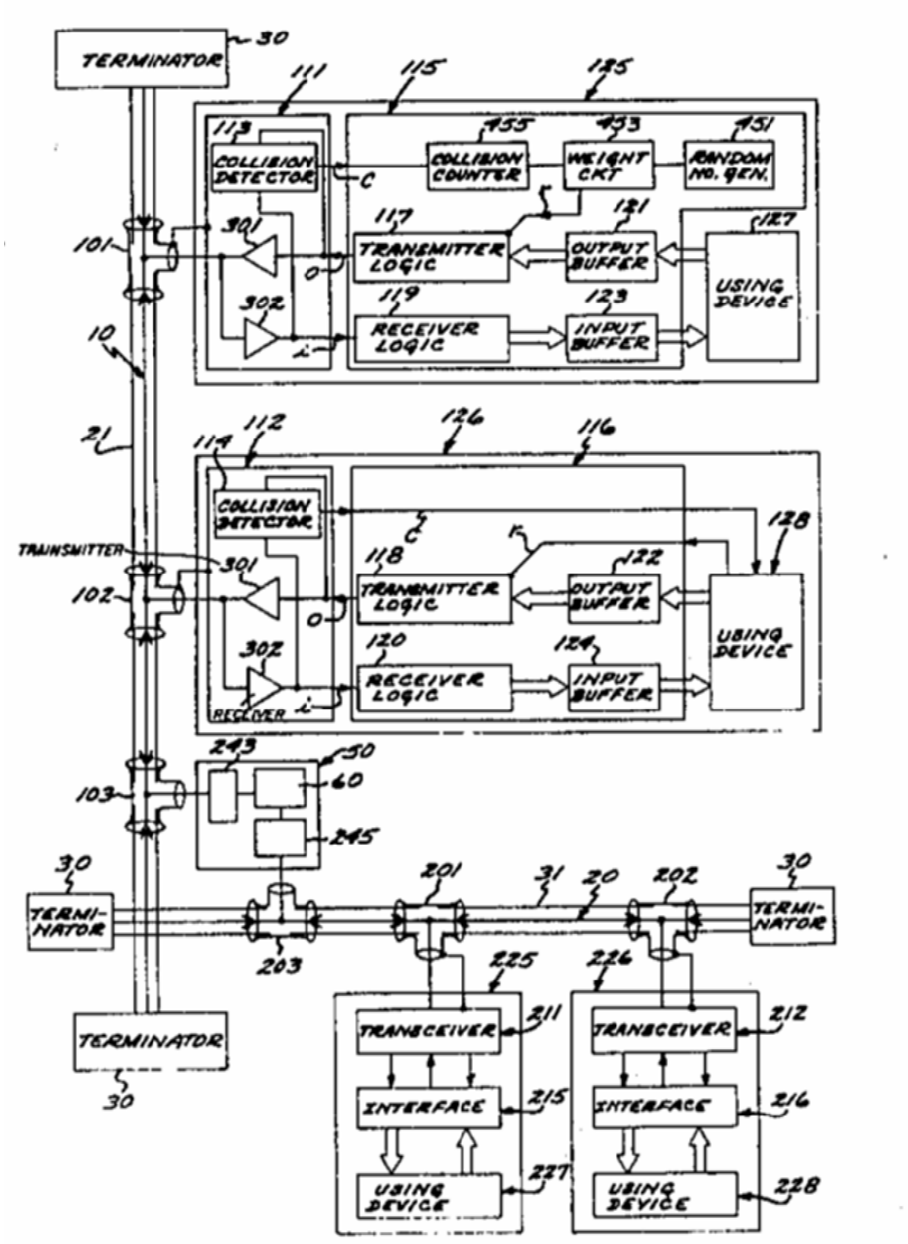
CSCE 560
Introduction to Computer Networking



Ethernet – US Patent #4063220

Dr. Barry Mullins
AFIT/ENG
Bldg 642, Room 209
255-3636 x7979

Link Layer

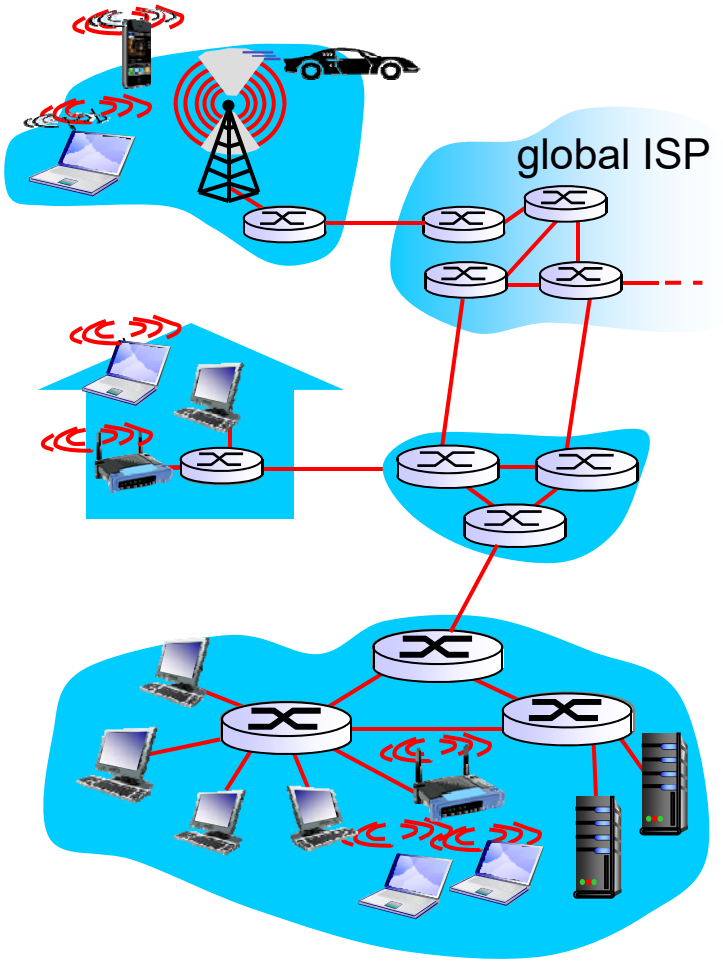
- 6.1 Introduction
- 6.2 Error Detection and Correction Techniques
- 6.3 Multiple Access Links and Protocols
- 6.4 Switched Local Area Networks
- 6.5 Link Virtualization
- 6.6 Data Center Networking
- 6.7 A Day in the Life of a Web Page Request

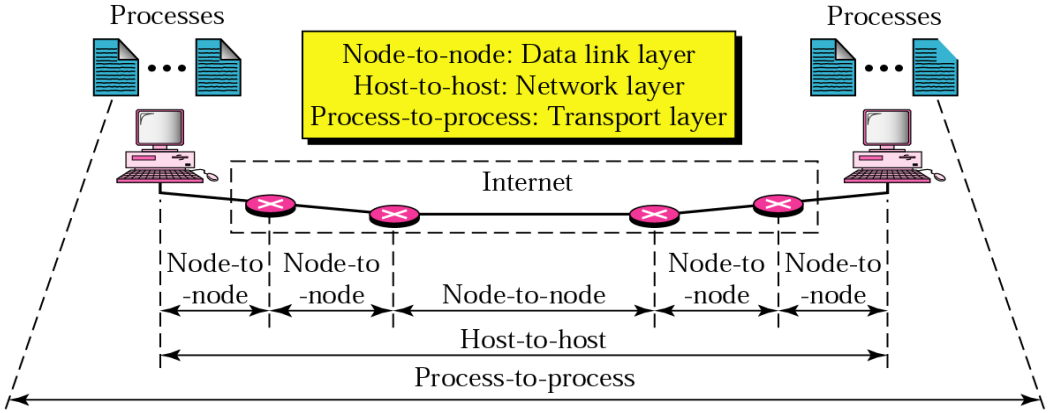
Link Layer: Introduction

Some terminology:

- Communication channels that connect adjacent nodes along communication path are links
  - Wired links
  - Wireless links
  - LANs
- Layer-2 packet is a frame, encapsulates datagram

Data-link layer is responsible for transferring datagram from one node to physically adjacent node over a link

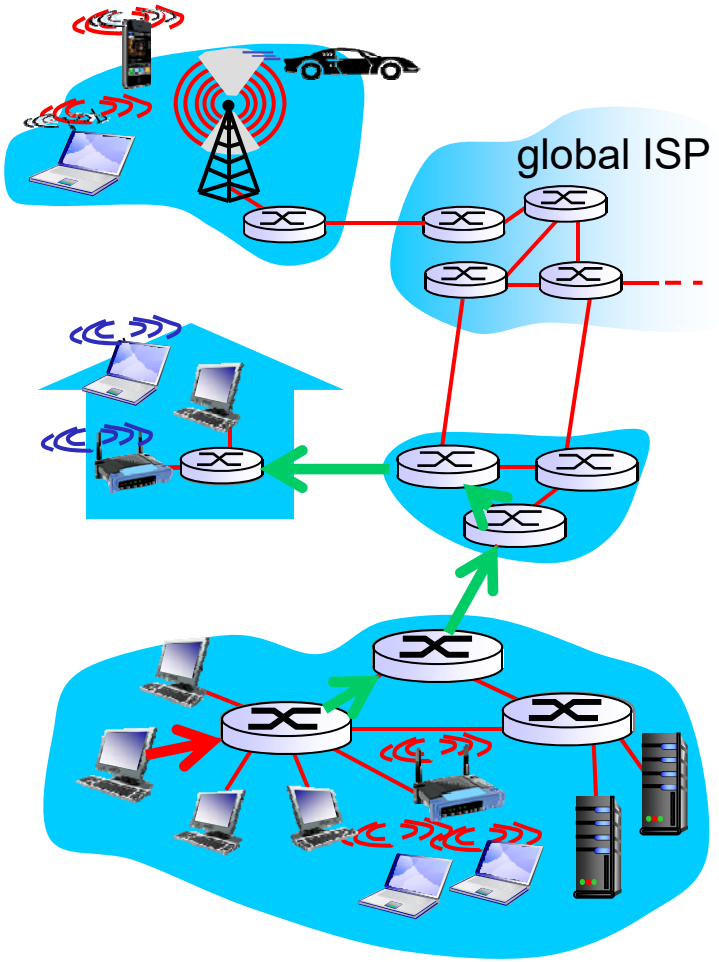




Link Layer: Context

- Datagram transferred by different link protocols over different links:
  - Ethernet on first link
  - Frame relay on intermediate links
  - 802.11 on last link

- Each link protocol provides different services
  - May or may not provide reliable data transfer over link



Link Layer Services

- Framing, link access:
  - Encapsulate datagram into frame, adding header, trailer
  - Channel access if shared medium
  - "MAC" addresses in frame headers to identify source, dest
    - Different from IP address!

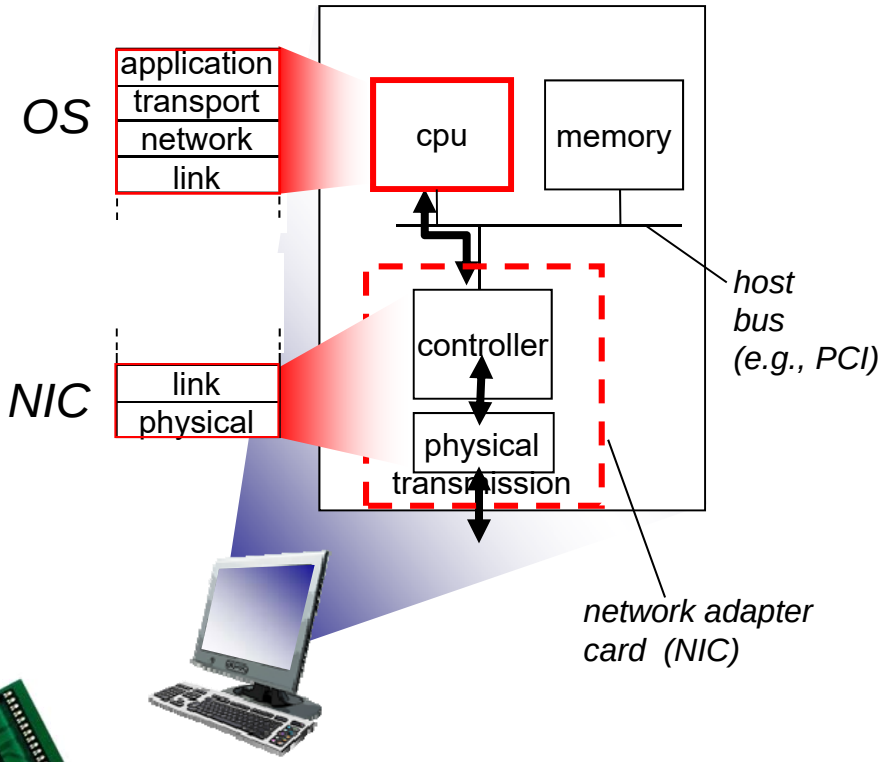
- Reliable delivery between adjacent nodes
  - We learned how to do this already (Chapter 3)!
  - Seldom used on low bit error link (fiber, some twisted pair)
  - Wireless links have higher error rates

Link Layer Services (more)

- Flow Control:
  - Pacing between adjacent sending and receiving nodes
- Error Detection:
  - Errors caused by signal attenuation, noise
  - Receiver detects presence of errors:
    - Signals sender for retransmission or drops frame
- Error Correction:
  - Receiver identifies and corrects bit error(s) without retransmission
- Half-duplex and full-duplex
  - Half duplex: nodes at both ends of link can transmit, but not at same time
  - Full duplex: nodes at both ends of link can transmit at same time

Adapters Communicating

- Link layer implemented in both
  - OS
  - "Adapter" (aka NIC)
- Adapter (Ethernet or 802.11 card) attaches to host's system bus
- Adapter is semi-autonomous
  - Can receive frame and discard without notifying CPU



Link Layer

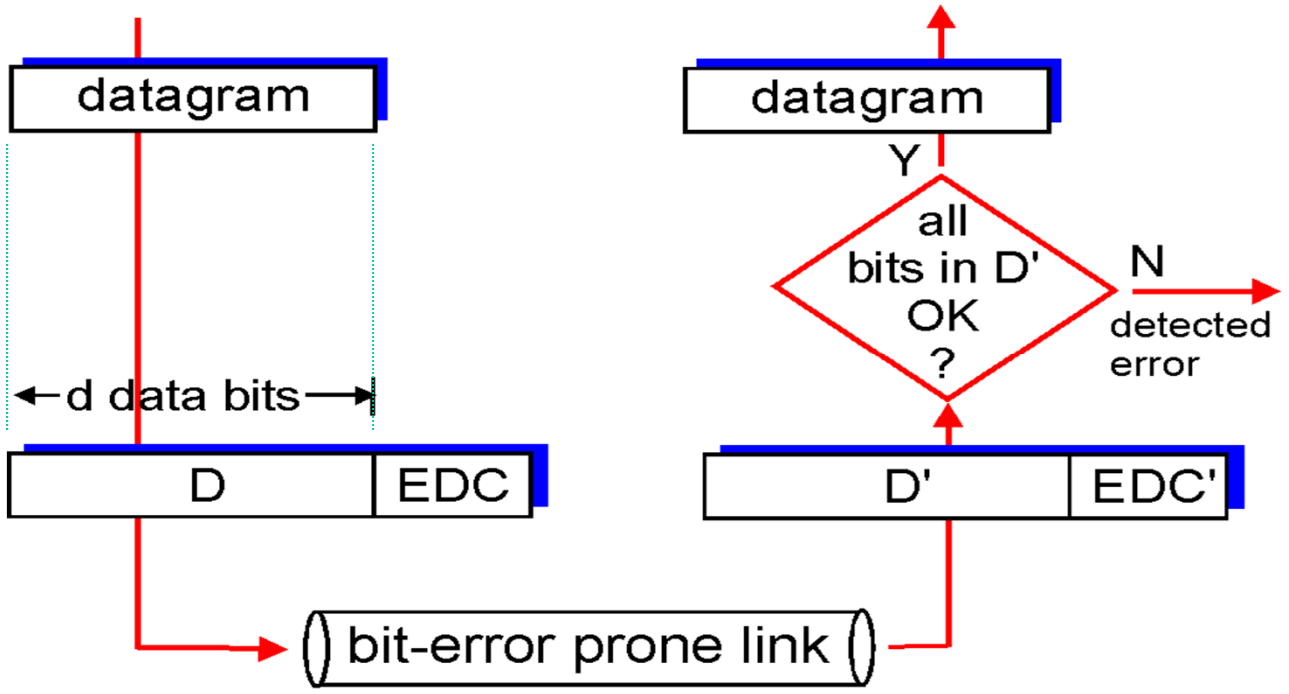
- 6.1 Introduction
- 6.2 Error Detection and Correction Techniques
- 6.3 Multiple Access Links and Protocols
- 6.4 Switched Local Area Networks
- 6.5 Link Virtualization
- 6.6 Data Center Networking
- 6.7 A Day in the Life of a Web Page Request

Error Detection (A Subset of Coding Theory)

EDC = Error Detection and Correction bits (redundancy)

D = Data protected by error checking, may include header fields

- Error detection not 100% reliable!
  - Protocol may miss some errors
  - Larger EDC field yields better detection and correction
    - … but increases # of overhead bits

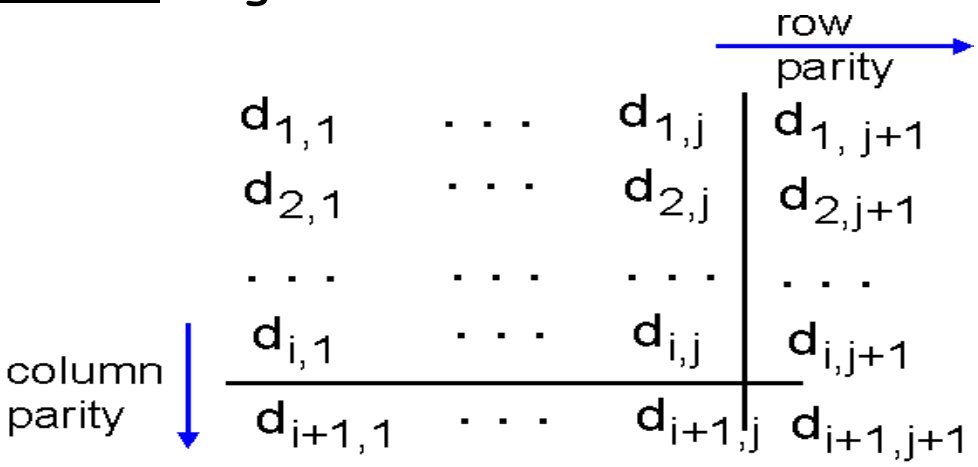


Single Parity Checks

- Single bit is added to a string of bits such that the string plus parity bit always has an even (or odd) number of 1's
  - Transmitted code has an even (odd) number of 1's
  - If odd (even) number of 1's received, then error
  - Does not indicate where error is, so cannot correct
- Can detect one or any odd number of bit errors
- Cannot detect two or any even number of bit errors
  - A burst error is just as likely to cause an even number of errors (undetectable) as an odd number of errors (detectable)
- Example assuming even parity:
  - Data = **0110100** → Even parity = **1**
  - Xmited word = **01101001**
  - Received word = **11101001** → Error b/c odd # of 1's
  - Received word = **11111001** → Even # of 1's so "OK" (but has two bit errors)

Two Dimensional Parity Checking

- Detect and correct single bit errors



1	0	1	0	1	1	
1	1	1	1	0	0	
0	1	1	1	0	1	
0	0	1	0	1	0	

no errors

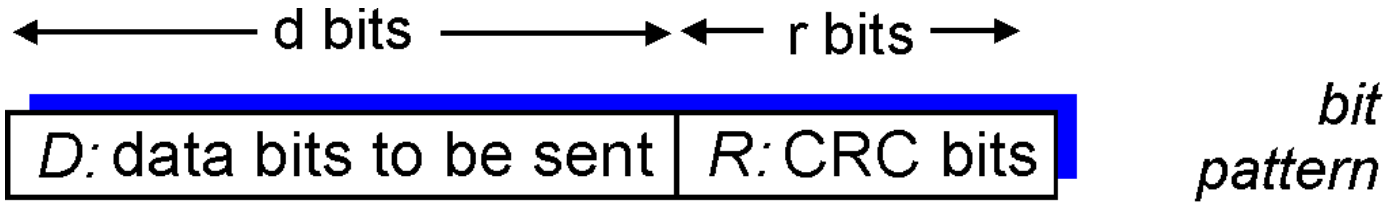
1	0	1	0	1	1	
1	0	1	1	0	0	parity error
0	1	1	1	0	1	
0	0	1	0	1	0	

parity error

correctable single bit error

Cyclic Redundancy Check - Transmit

- View data bits, D, as a binary number
- Choose r+1 bit pattern (generator polynomial), G
  - Always given → Ethernet uses G = 100000100110000010001110110110111
  - $x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1$
  - Ethernet CRC: r = 32 bits → G is r+1 = 33 bits
- Goal: calculate r CRC bits, called R, such that
  - <D,R> is exactly divisible by G (using modulo 2 arithmetic)



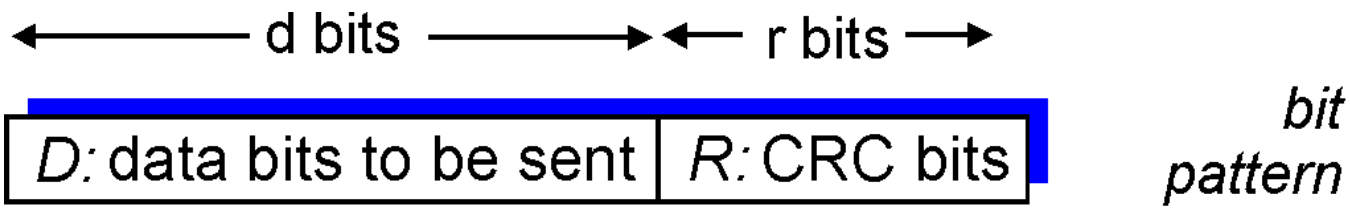
$D * 2^r$ XOR R

mathematical formula

Cyclic Redundancy Check - Receive

- Receiver knows G, divides <D,R> by G (modulo 2 arithmetic)
  - If non-zero remainder: error detected!

- Can detect all burst errors less than r+1 bits and any odd number of bit errors



CRC Example

We want $D \cdot 2^r$ XOR R to be exactly divisible by G, so we start with:

$D \cdot 2^r$ XOR R = nG

Now find R.

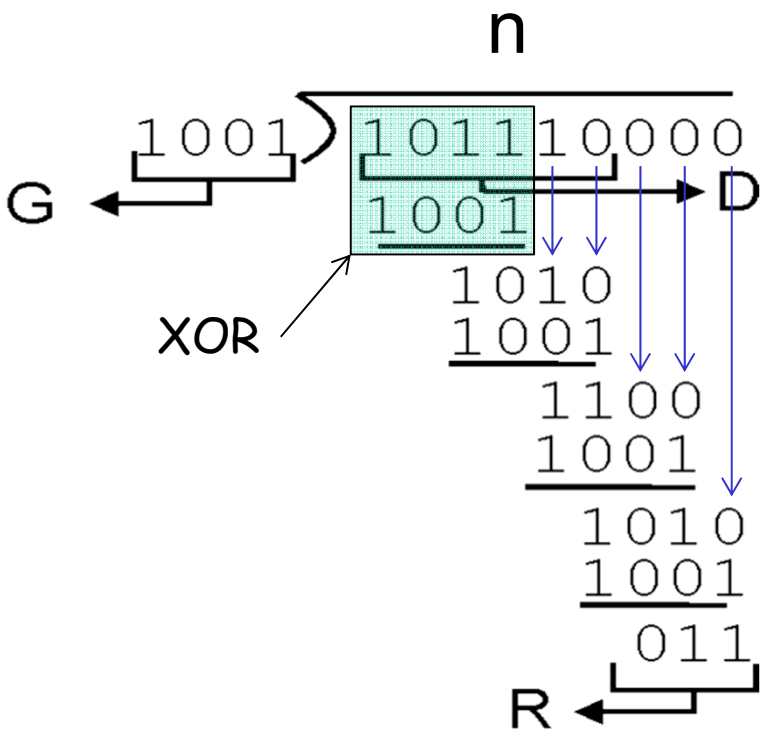
"Add" (mod 2) XOR R to both sides:

$D \cdot 2^r$ = nG XOR R

If we divide $D \cdot 2^r$ by G, we get n with a remainder R

We ignore n; it is not needed

$$R = \text{remainder}\left[\frac{D \cdot 2^r}{G}\right]$$



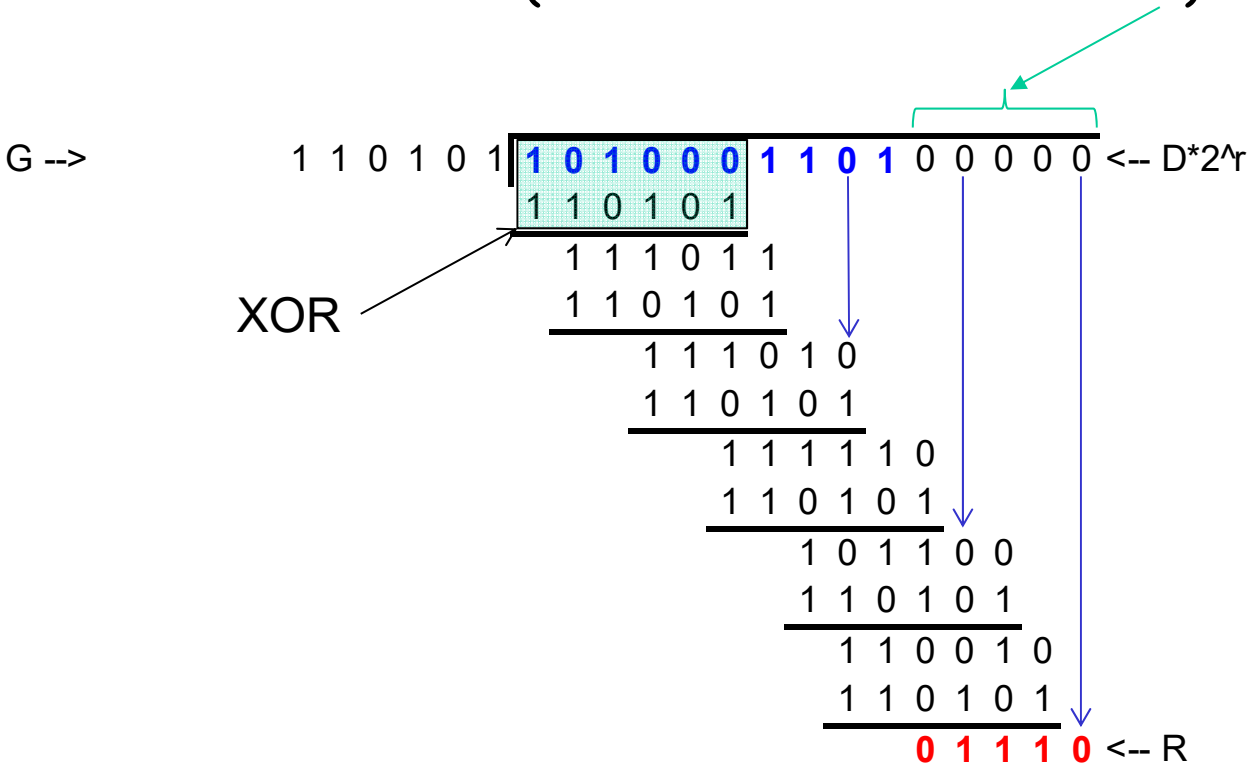
Source sends 101110011

D R

$D \cdot 2^r$ XOR R

CRC Example Transmit

- Data (D) = 1010001101
- Given: Generator = 110101 (G has 6 bits so r = 6-1 = 5)

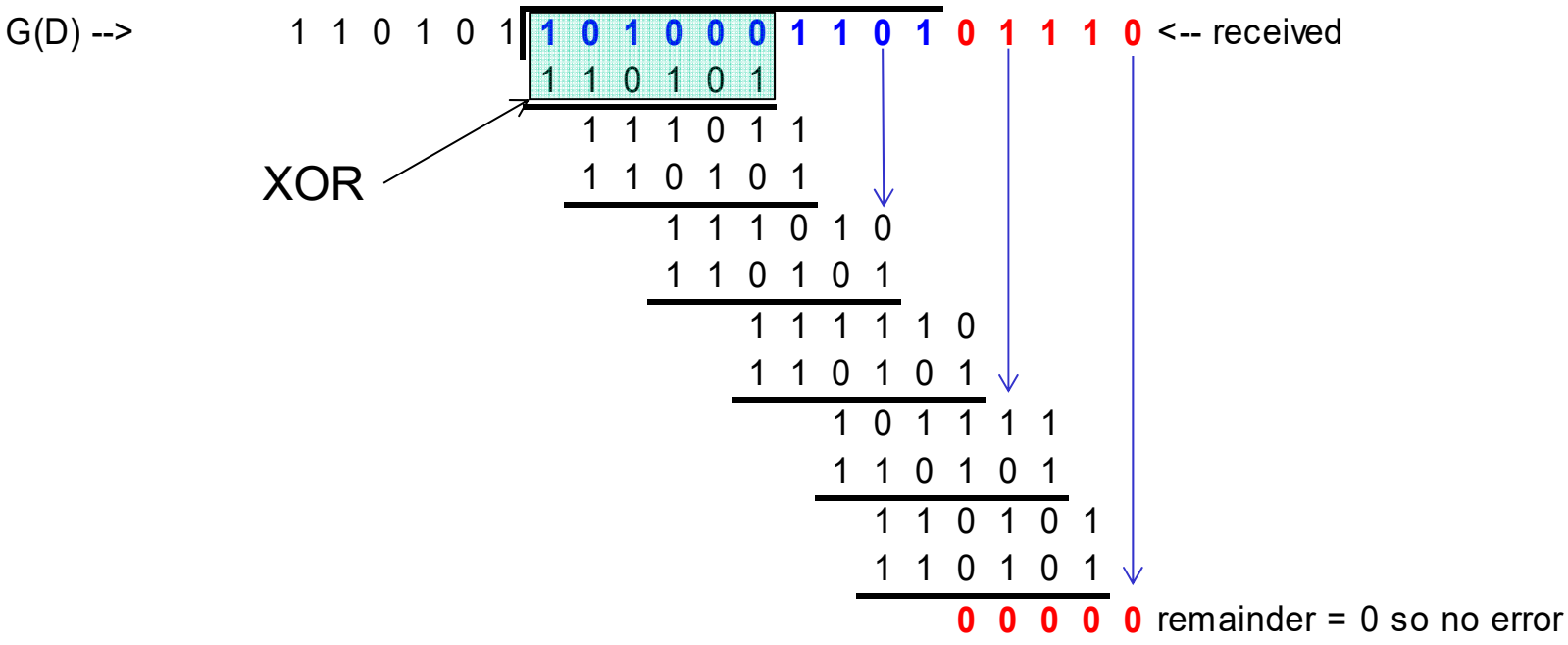


Transmitted message [x(D)] = 1 0 1 0 0 0 1 1 0 1 0 1 1 1 0

CRC Example Receive

- Received data + R = 1010001101**01110**
- Generator = 110101

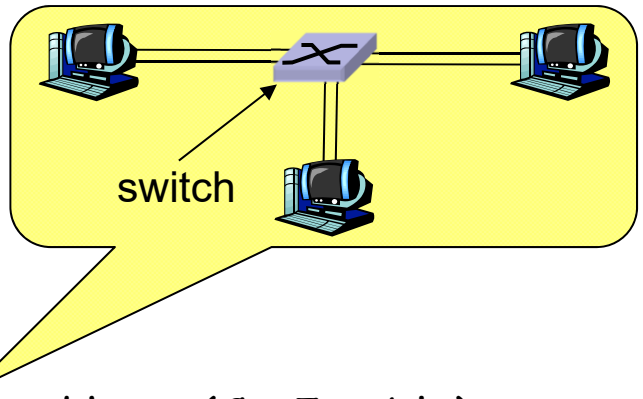
Receive



Link Layer

- 6.1 Introduction
- 6.2 Error Detection and Correction Techniques
- 6.3 Multiple Access Links and Protocols
- 6.4 Switched Local Area Networks
- 6.5 Link Virtualization
- 6.6 Data Center Networking
- 6.7 A Day in the Life of a Web Page Request

Multiple Access Links

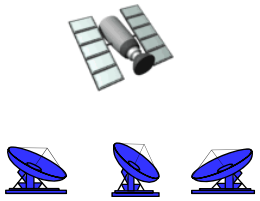


Two types of "links":

- Point-to-point
  - PPP like dial-up access
  - Point-to-point link between Ethernet switch and host (Cat5 cable)
- Broadcast (shared wire or medium)
  - Bus (coax) Ethernet → legacy Ethernet
  - 802.11 wireless LAN

shared wire
(e.g., coax Ethernet)

shared RF
(e.g., 802.11 WiFi)



shared RF
(satellite)

humans at a
cocktail party
(shared air, acoustical)

Multiple Access Protocols

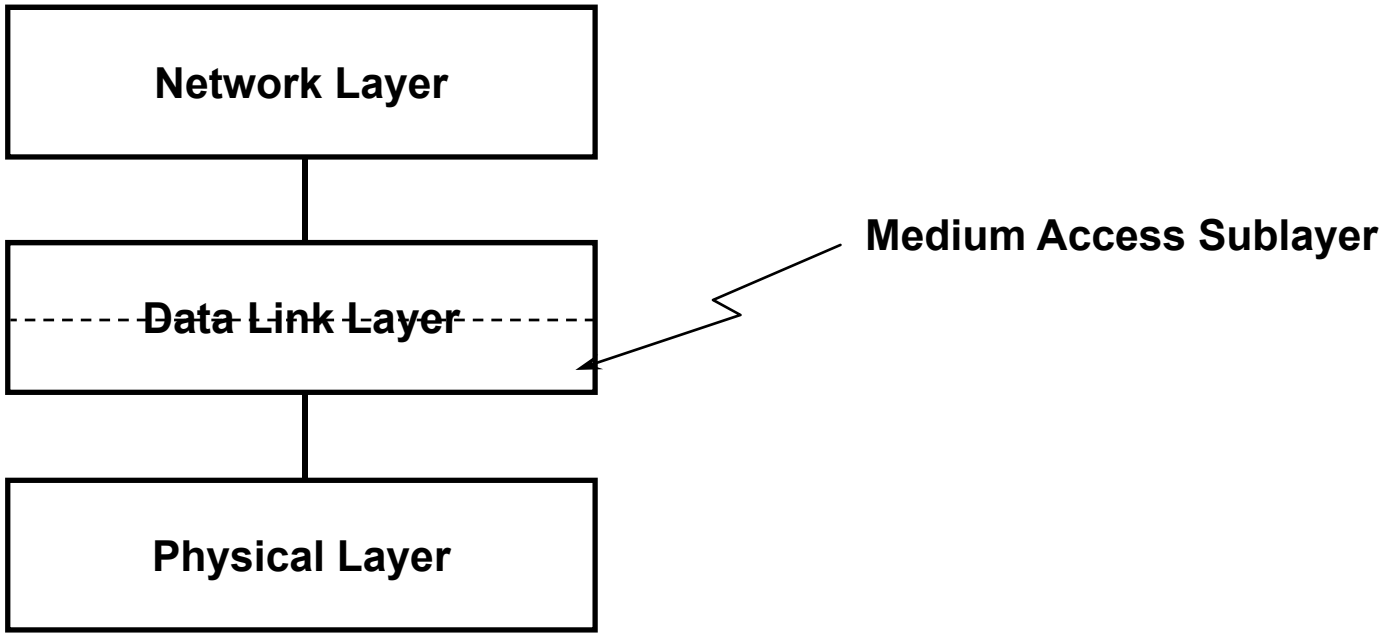
- Single shared broadcast channel
- Two or more simultaneous transmissions by nodes → interference
  - Collision if node receives two or more signals at the same time

Multiple access protocol

- Distributed algorithm that determines how nodes share channel
  - Determines when a node can transmit
- Communication about channel sharing must use the channel itself!
  - No out-of-band channel for coordination

Medium Access Sublayer

- *Medium access control* (MAC) provides coordination among nodes
  - MAC is sublayer between data link layer and physical layer
  - Usually grouped with data link layer



Ideal Multiple Access Protocol

What are the characteristics of an ideal MAC protocol?

Assume a broadcast channel of rate R bps

1. When one node wants to transmit, it can send at rate R
2. When N nodes want to transmit, each can send at average rate R/N
3. Fully decentralized:
   - No special node to coordinate transmissions
   - No synchronization of clocks or slots
4. Simple (cheap)

MAC Protocols: A Taxonomy

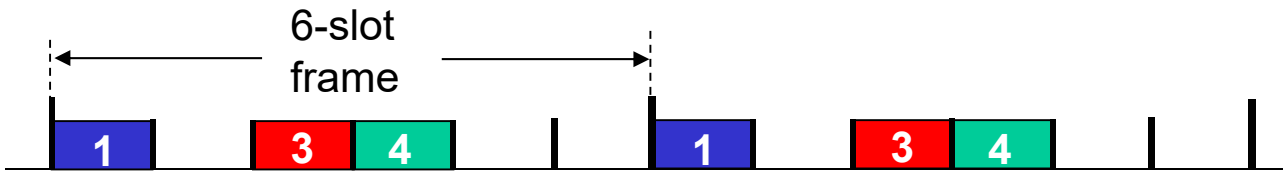
Three broad classes:

1. Channel Partitioning
   - Divide channel into smaller "pieces"
     - time slots
     - frequency band
     - chipping code (discussed in wireless chapter)
   - Allocate a piece to a node for exclusive use
2. "Taking turns"
   - Nodes take turns, but nodes with more to send can take longer turns
3. Random Access ("Free-for-all" approach)
   - Nodes transmit new packets as they are available without concern for other traffic on the media
   - Channel not divided
     - Collisions possible
     - Need a technique to "recover" from collisions

Channel Partitioning MAC Protocols: TDMA

TDMA: Time Division Multiple Access

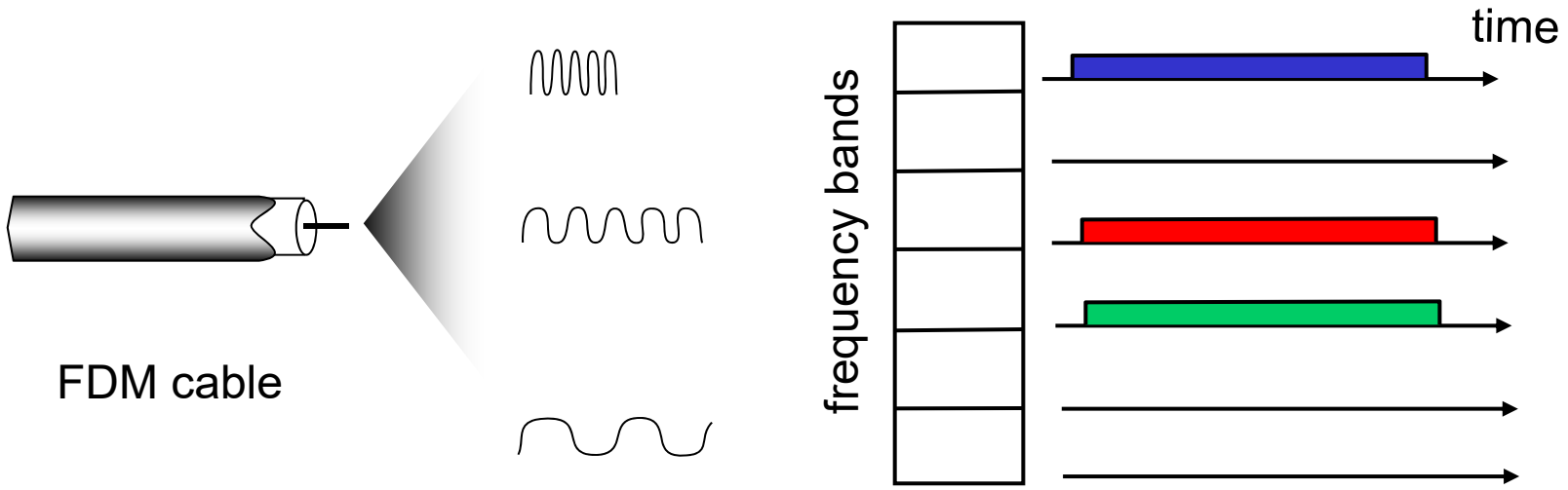
- Access to channel in "rounds"
- Each station gets fixed length slot (length = pkt trans time) in each round
- Unused slots go idle
- Example: 6-station LAN, 1,3,4 have pkt, slots 2,5,6 idle



Channel Partitioning MAC Protocols: FDMA

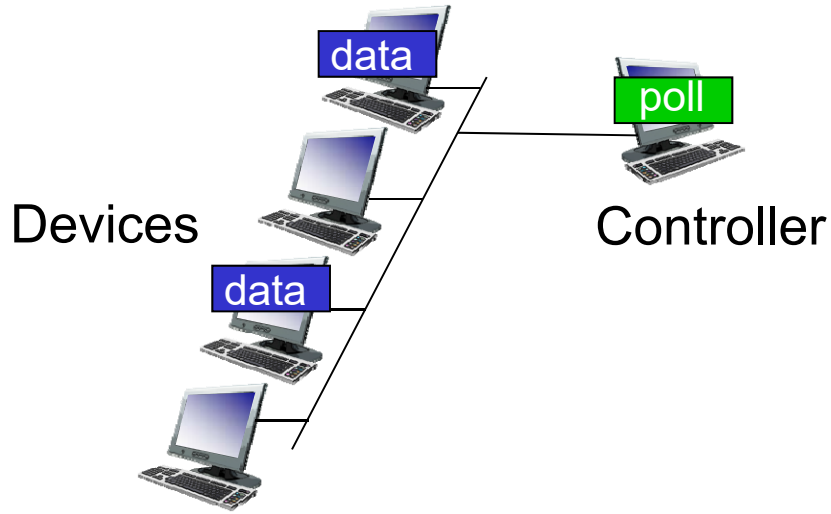
FDMA: Frequency Division Multiple Access

- Channel spectrum divided into frequency bands
- Each station assigned frequency band (bandwidth divided by N)
- Unused transmission time in frequency bands go idle
- Example: 6-station LAN, 1,3,4 have pkt, frequency bands 2,5,6 idle



"Taking Turns" MAC protocols

- Channel partitioning MAC protocols:
  - Share channel efficiently and fairly at high load
  - Inefficient at low load:
    - Delay in channel access
    - 1/N bandwidth allocated even if only 1 active node!
- "Taking turns" protocols
  - Polling
  - Token passing
    - Station only sends when it has the "token"

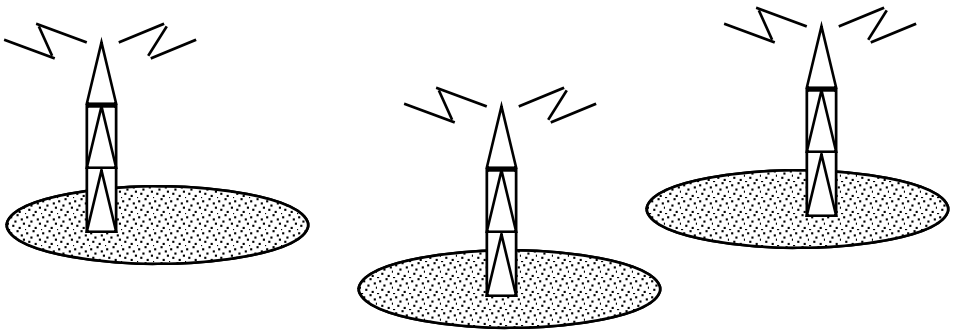


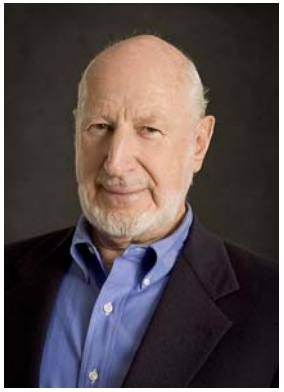
- Random access MAC protocols
  - Efficient at low load: single node can fully utilize channel
  - High load: collision overhead

Random Access Protocols

- When node has packet to send
  - Transmit at full channel data rate R
  - No *a priori* coordination among nodes
- Two or more transmitting nodes → "collision"
- Random access MAC protocol specifies how to:
  - Detect collisions
  - Recover from collisions
    - e.g., via delayed retransmissions
- Examples of random access MAC protocols:
  - Slotted ALOHA
  - ALOHA
  - CSMA, CSMA/CD, CSMA/CA

ALOHAnet





- Developed at the U of Hawaii in early 1970's by Norm Abramson
  - Provided 9600 bps packet-switched communication between central computer and remote terminals
  - Used 2 frequencies in a hub/star configuration
    - All nodes transmit to hub on one channel
    - Hub broadcasts packet to all on other channel
- We'll discuss slotted ALOHA first followed by the original (unslotted) ALOHA

Slotted ALOHA

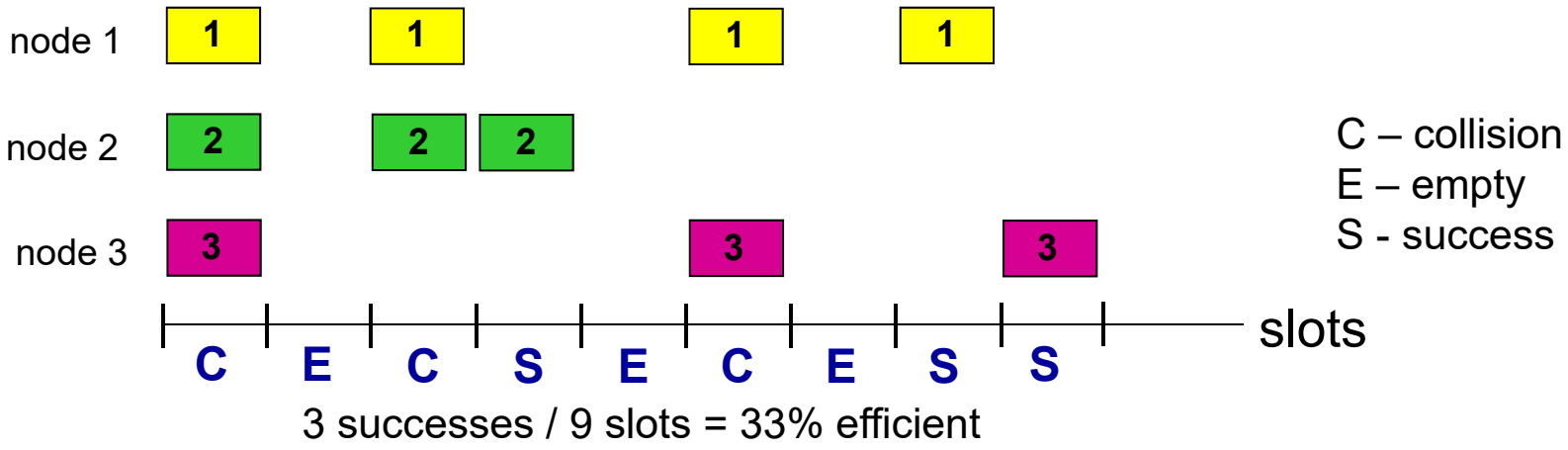
Assumptions

- All frames same size
- Time is divided into equal slots
  - Time to transmit 1 frame
- Nodes start to transmit frames only at beginning of slots
- Nodes are synchronized
- If a node hears its packet being broadcast by the hub, no collision
- Checksum used to detect errors

Operation

- When node obtains new frame from network layer, it transmits in next slot
- If no collision,
  - Node has successfully transmitted frame
  - No retransmission
- If collision,
  - Node retransmits frame in each subsequent slot with probability p until success

Slotted ALOHA



3 successes / 9 slots = 33% efficient

Pros

- Single active node can continuously transmit at full rate of channel
- Highly decentralized: each node detects collisions and independently retransmits
- Simple

Cons

- Collisions, which wastes slots
- Empty / idle slots
- Nodes may be able to detect collision in less time than to transmit packet
  - Would be nice if protocol allowed nodes to stop transmitting before the end of the slot
- Clock synchronization

Slotted ALOHA Efficiency

- Efficiency is the long-run fraction of successful slots when there are many nodes, each with many frames to send
- Suppose N nodes with many frames to send, each transmits in slot with probability *p*

- Prob that one node has success in a slot

  = [prob just one transmits] x [all others (N-1) do not transmit]

  = $p \times (1-p)^{N-1}$

- Prob that any of the N nodes has a success

  = $Np(1-p)^{N-1}$

Slotted ALOHA Efficiency

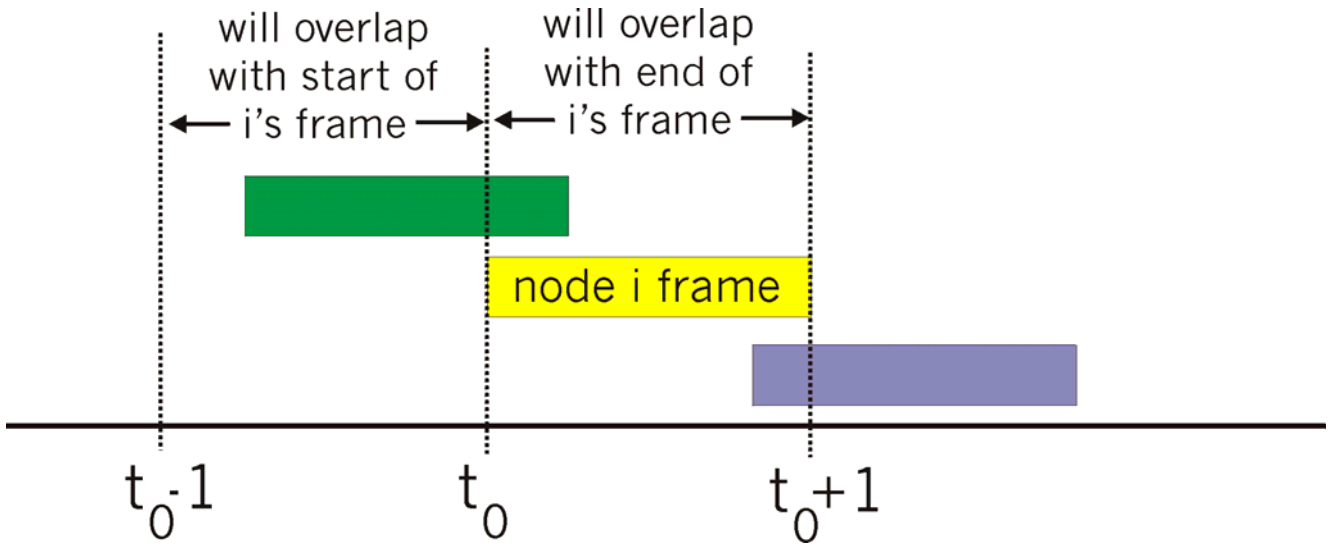
- For max efficiency with N nodes, find p* that maximizes
  $Np(1-p)^{N-1}$

- For many nodes, take limit of $Np^*(1-p^*)^{N-1}$ as N goes to infinity, gives: max efficiency = 1/e = 0.37

- *At best:* channel used for useful transmissions 37% of time!

Pure (Unslotted) ALOHA

- Unslotted ALOHA: simpler, no synchronization
- When frame first arrives from network layer → Transmit immediately
- However, collision probability increases:
  - Frame sent at t_0 collides with other frames sent in the time intervals of [t_0-1 , t_0+1]



Pure ALOHA Efficiency

P(success by a given node) =

P(node transmits) ×

P(no other node transmits in $[t_0-1,t_0]$ ×

P(no other node transmits in $[t_0,t_0+1]$

$= p \cdot (1-p)^{N-1} \cdot (1-p)^{N-1} = p \cdot (1-p)^{2(N-1)}$

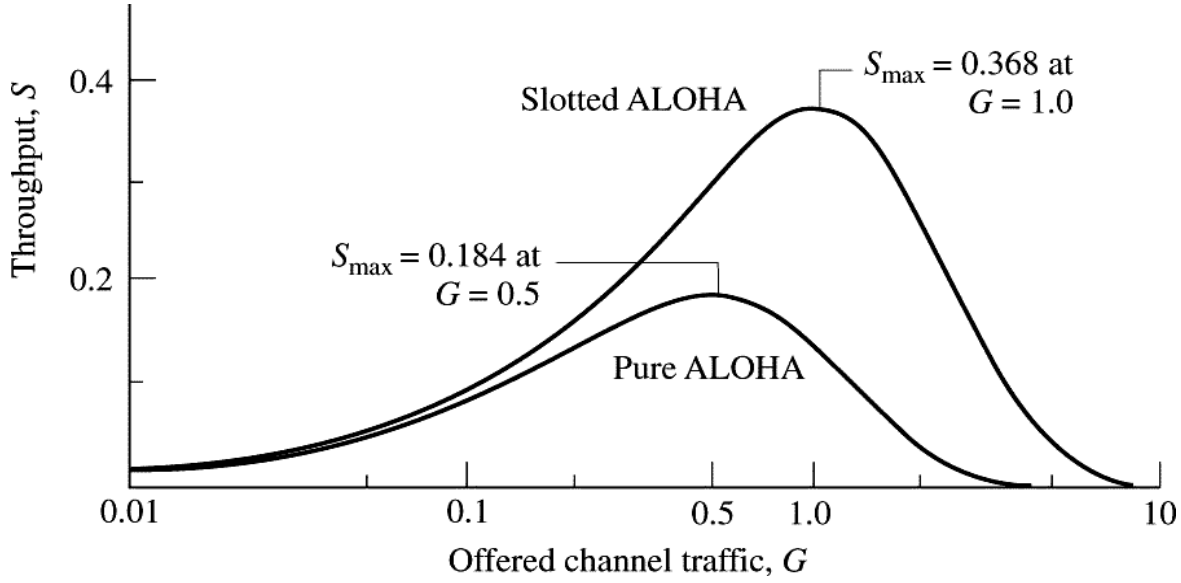
P(success by any given node) = $Np \cdot (1-p)^{2(N-1)}$

… choosing optimum p and then letting N -> infinity …

$= 1/(2e) = 0.18$

Now channel used for useful transmissions only 18% of time!

Even worse !

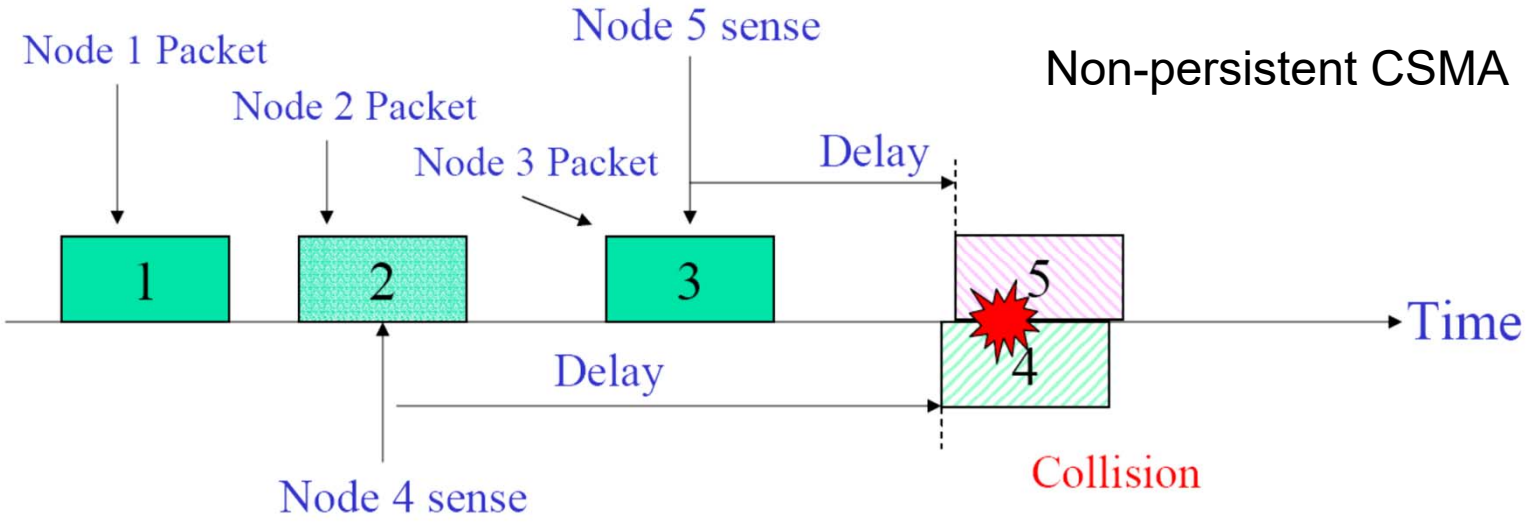


G (transmission attempts per slot)

CSMA (Carrier Sense Multiple Access)

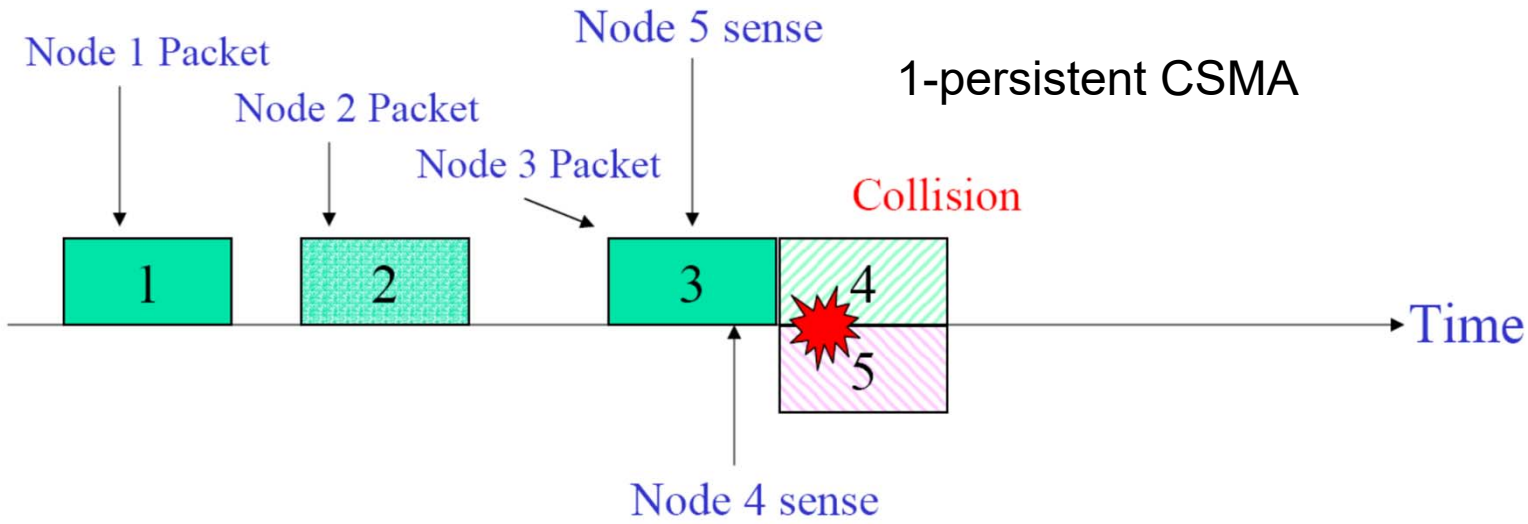
CSMA: listen before transmit

- If channel sensed idle: transmit entire frame immediately
- If channel sensed busy, defer transmission
  - 1-persistent: transmit immediately when channel becomes available (selfish) – shown on next slide
  - Non-persistent CSMA: retry after random amount of time (less greedy)



Tradeoff between 1- & Non-Persistent CSMA

- If 4 and 5 become ready in the middle of 3's transmission,
  - 1-Persistent: 4 and 5 will collide



  - Non-Persistent: Probability is less that 4 and 5 will collide

- If only 4 becomes ready in the middle of 3's transmission,
  - 1-Persistent: 4 succeeds as soon as 3 ends
  - Non-Persistent: 4 may have to wait wasting time

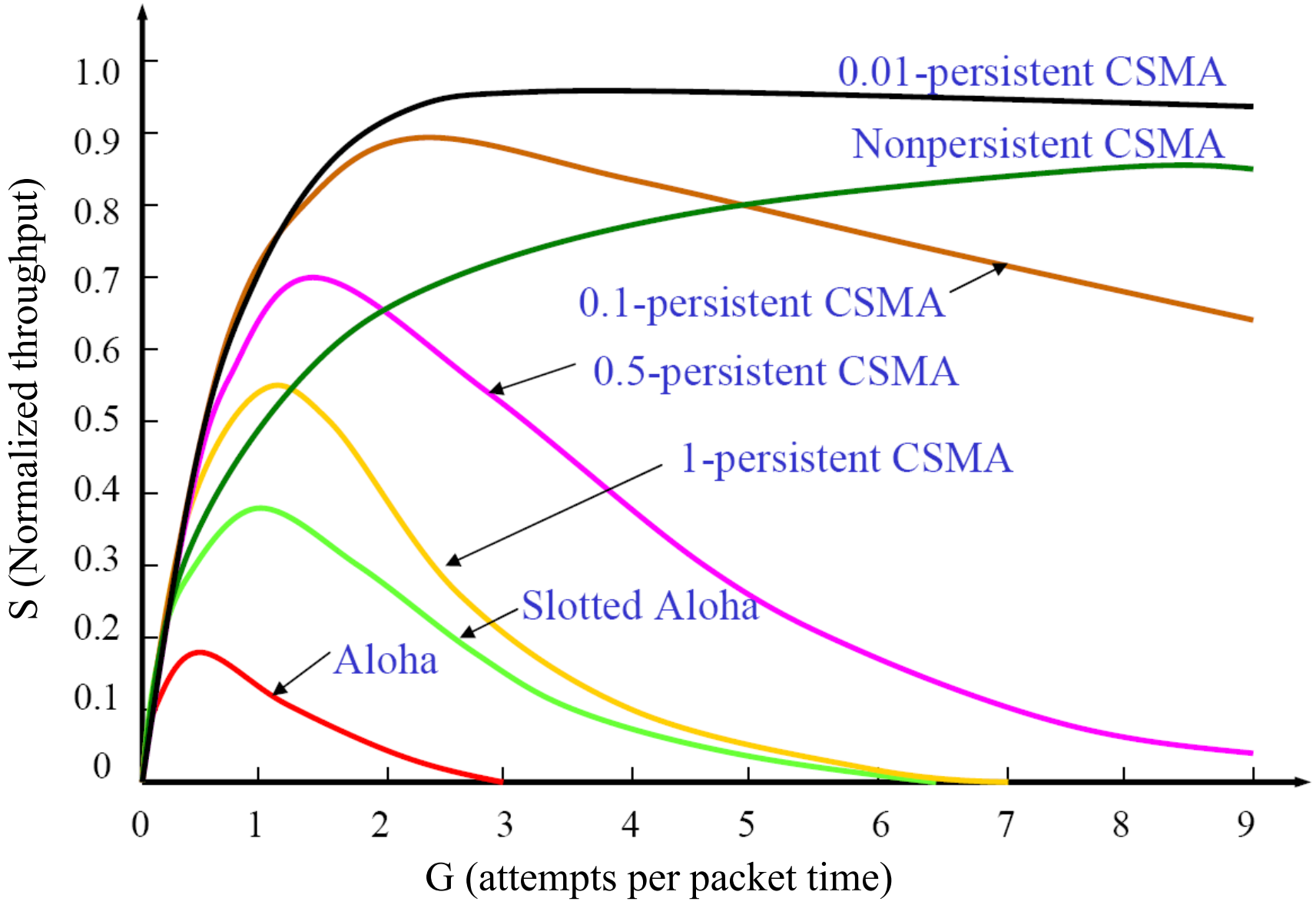
Optimally Greedy: P-persistent CSMA

- Good tradeoff between non-persistent and 1-persistent CSMA
- Channel is slotted with slot size = propagation delay

1. Protocol operation
   - If medium is idle,
     - transmit with probability p or
     - delay for one slot with probability (1-p), then go to Step 1
   - If medium is busy, continue to listen until medium becomes idle, then go to Step 1

- With high 'p', better low-load performance than non-persistent but worse at high load
- With low 'p', best performance (assuming N is large)

Comparison of CSMA & ALOHA Protocols



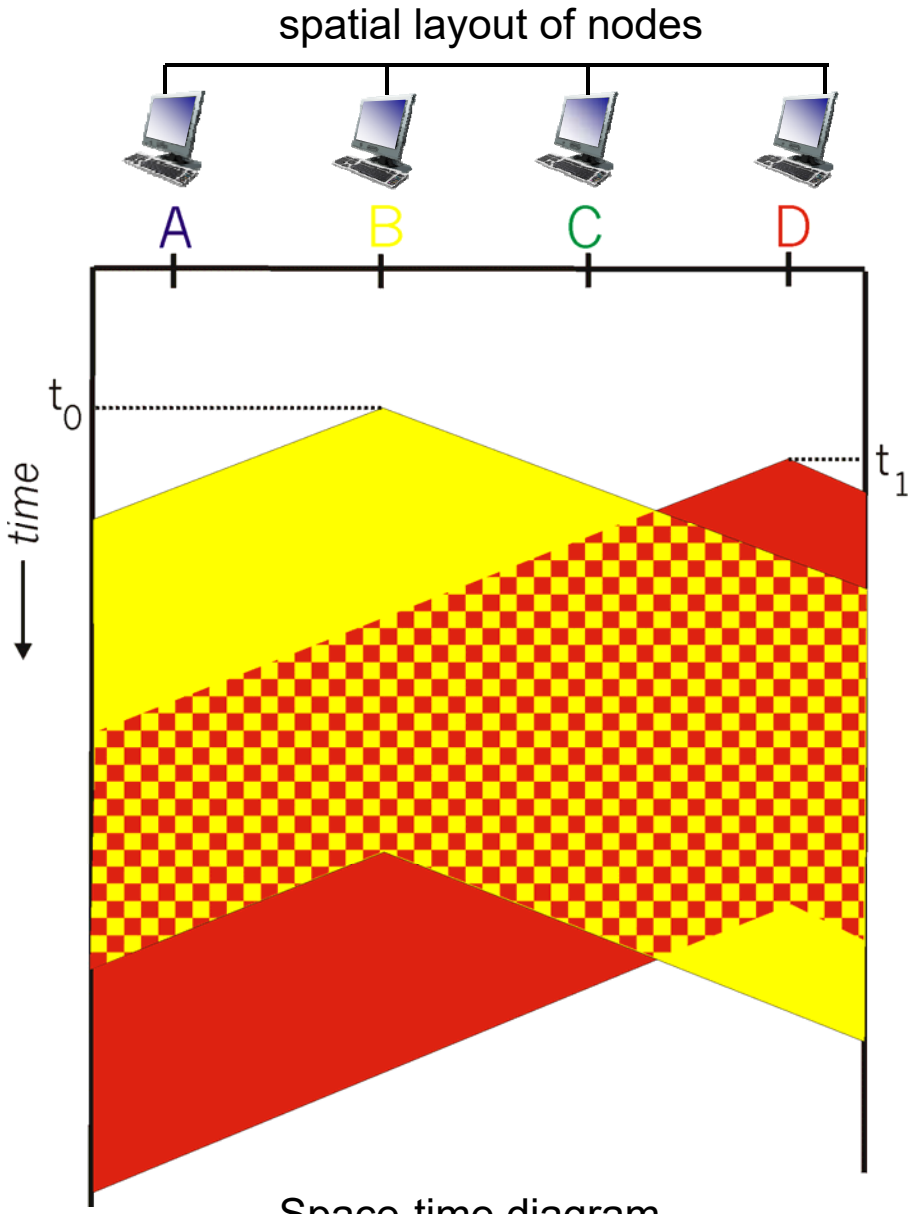
CSMA Collisions

Collisions can still occur:

- Propagation delay means two nodes may not hear each other's transmission in a timely manner

Collision:

- Entire packet transmission time wasted



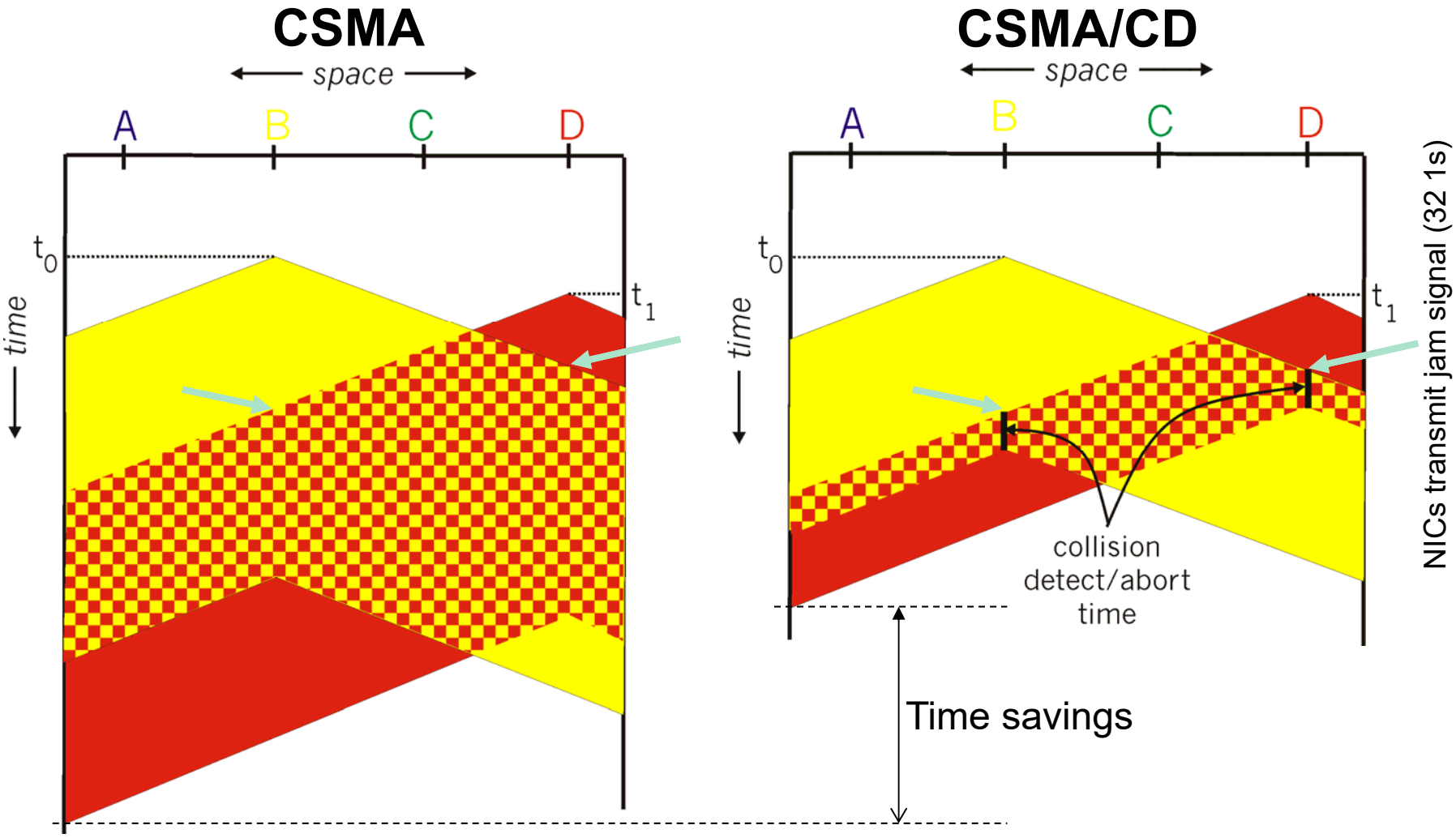
Space-time diagram

CSMA/CD (Collision Detection)

- Maintains carrier sensing and deferral as in CSMA
  - Collisions *detected* within short time
  - Colliding transmissions aborted reducing channel waste

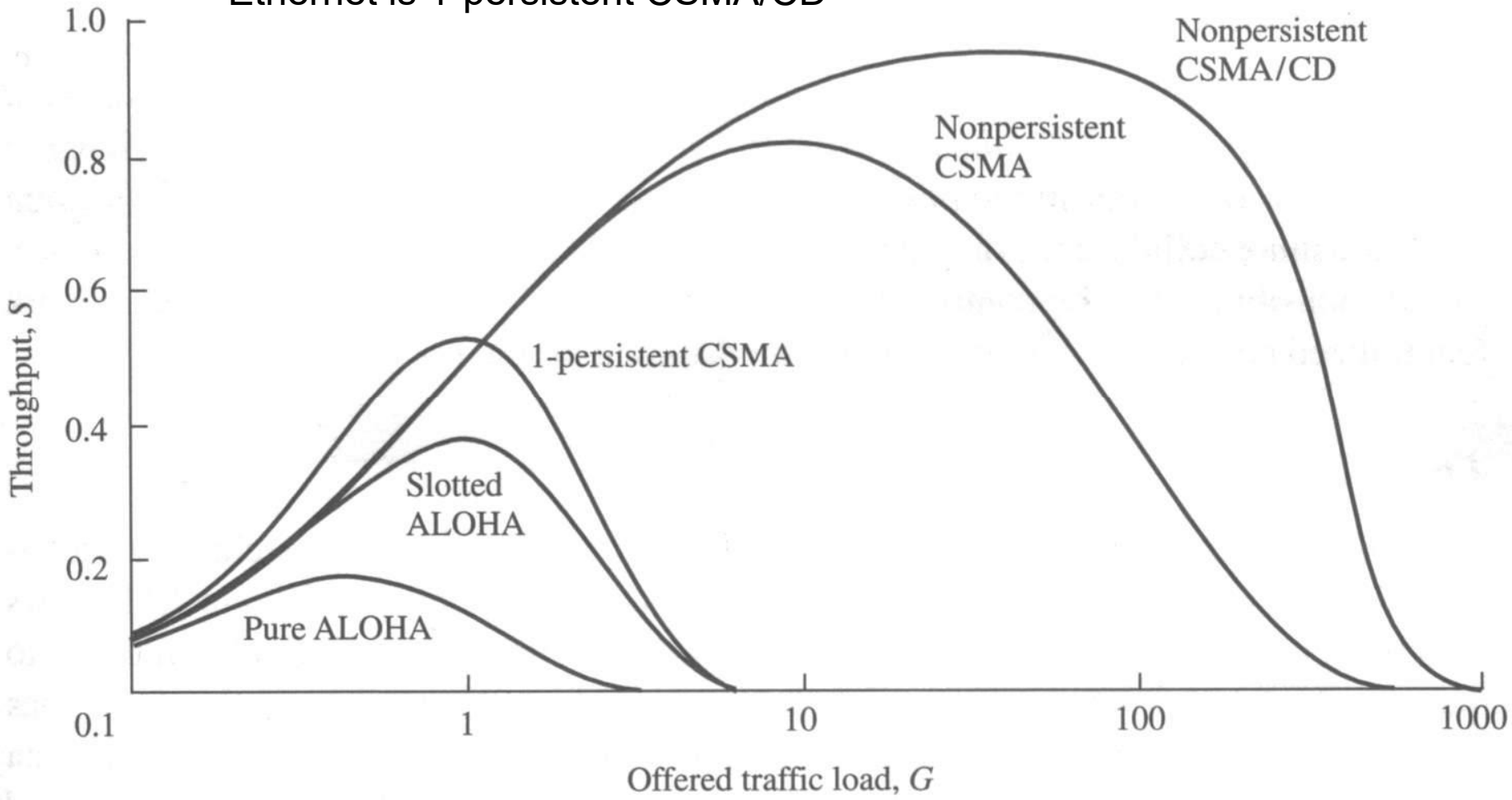
- Collision detection:
  - Easy in wired LANs: measure signal strengths, compare transmitted, received signals
  - Difficult in wireless LANs: receiver shut off while transmitting

CSMA/CD (Collision Detection)



CSMA/CD Versus Other Protocols

Ethernet is 1-persistent CSMA/CD



Local Area Networks (LANs)

- Multiple access protocols used extensively in LANs
- LAN is a network that resides in a geographically-restricted area
  - Usually span a building or a campus
  - Short propagation delays
  - Small number of users
  - Inexpensive
- LANs governed by the IEEE 802 standards
  - Also responsible for metropolitan area networks
  - IEEE 802 standards are restricted to networks carrying variable-size packets
  - Why 802?
    - It was the next number available although…
      the first meeting was held on the 2nd month in 1980

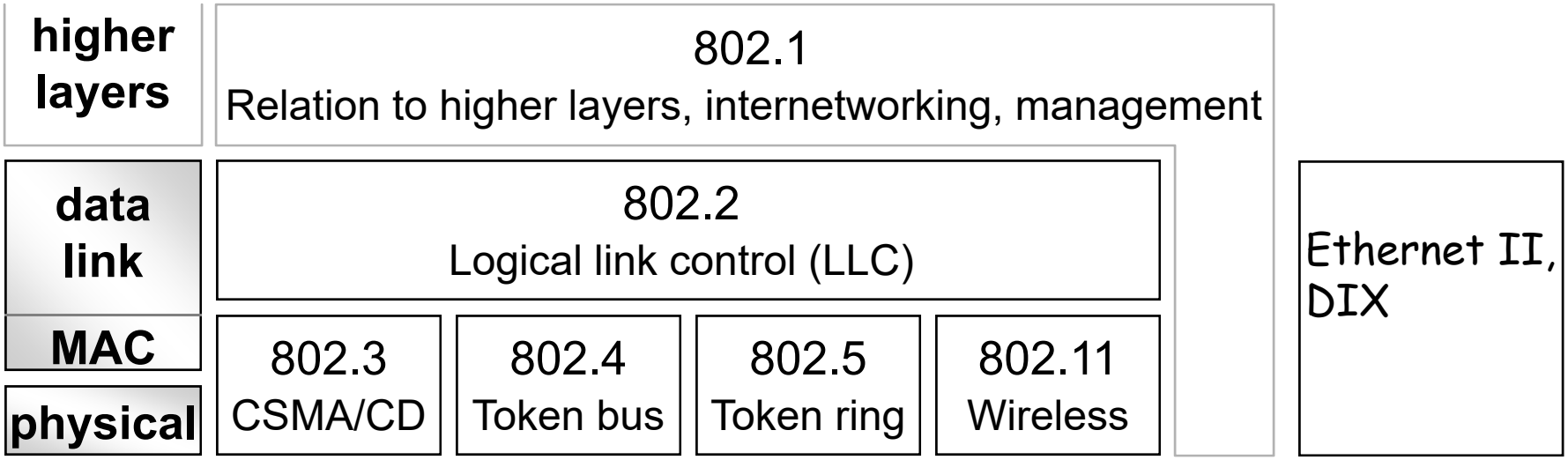
IEEE 802 Standards Working Groups

Number	Topic
802.1	Overview and architecture of LANs
802.2 ↓	Logical link control
802.3 *	Ethernet CSMA/CD bus
802.4 ↓	Token bus (was briefly used in manufacturing plants)
802.5	Token ring (IBM's entry into the LAN world)
802.6 ↓	Dual queue dual bus (early metropolitan area network)
802.7 ↓	Technical advisory group on broadband technologies
802.8 †	Technical advisory group on fiber optic technologies
802.9 ↓	Isochronous LANs (for real-time applications)
802.10 ↓	Virtual LANs and security
802.11 *	Wireless LANs
802.12 ↓	Demand priority (Hewlett-Packard's AnyLAN)
802.13	Unlucky number. Nobody wanted it
802.14 ↓	Cable modems (defunct: an industry consortium got there first)
802.15 *	Personal area networks (Bluetooth) 802.15.4 ZigBee
802.16 *	Broadband wireless Wireless metropolitan area networks
802.17	Resilient packet ring

The important ones are highlighted
The ones marked with ↓ are hibernating
The one marked with † gave up

IEEE 802 Standards (cont'd)

- 802 standards define:
  - Physical layer protocol
  - Data link layer protocol
    - Logical Link Control (LLC) Sublayer
    - Medium Access (MAC) Sublayer



Link Layer

- 6.1 Introduction
- 6.2 Error Detection and Correction Techniques
- 6.3 Multiple Access Links and Protocols
- 6.4 Switched Local Area Networks
- 6.5 Link Virtualization
- 6.6 Data Center Networking
- 6.7 A Day in the Life of a Web Page Request

IP versus MAC Addresses

- 32-bit IP address:
  - Network layer address
    - Used to get datagram to destination IP subnet
  - IP hierarchical address NOT portable
    - IP subnet dictates the IP address of a node
- MAC (AKA LAN or physical or Ethernet) address:
  - Used locally to get frame from one interface to another *physically-connected* interface on the same network (typically a LAN)
  - MAC addresses are completely flat ➜ Portability
    - Can move LAN card from one LAN to another
- Analogy:
  - MAC address: like Social Security Number
  - IP address: like postal address

MAC Addresses

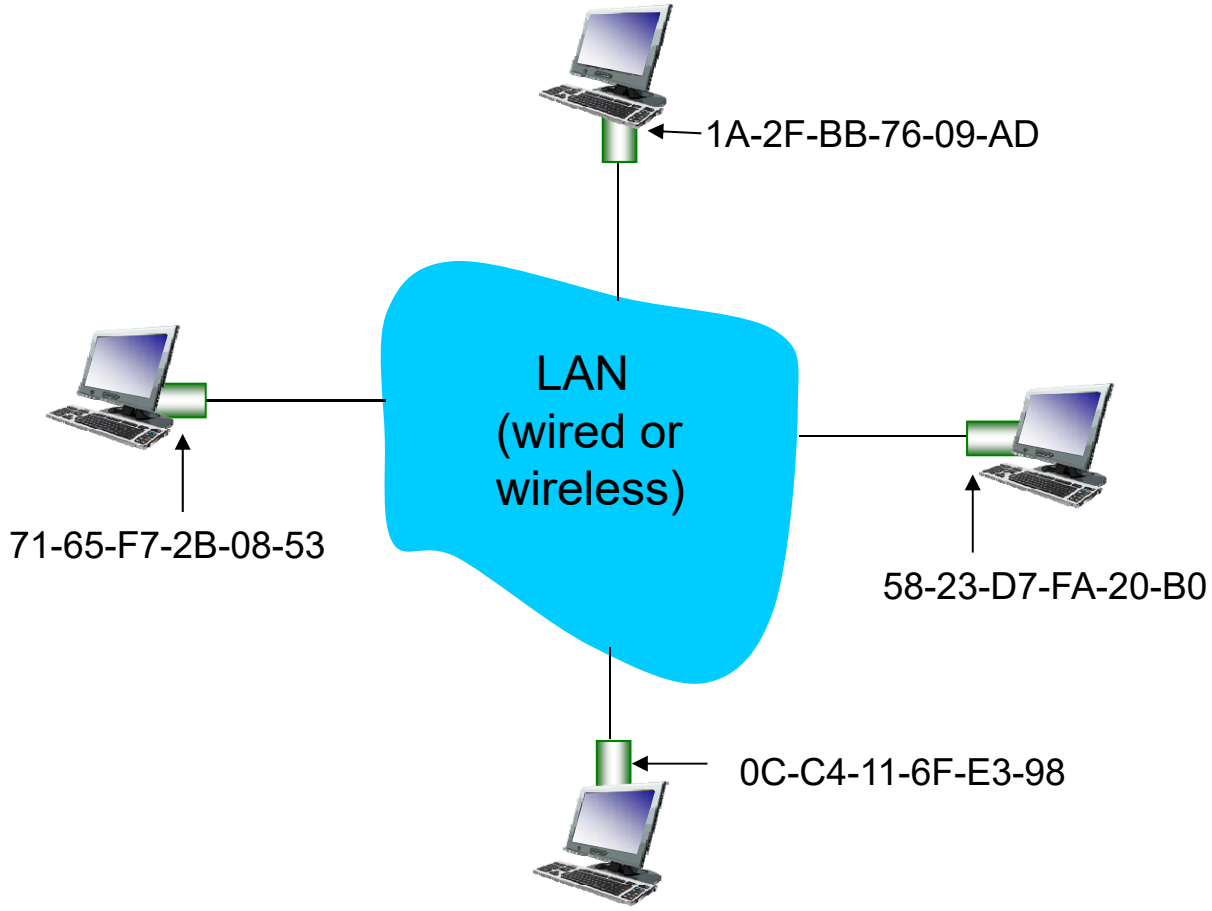
- MAC address allocation administered by IEEE
  - 48-bit MAC address burned in the adapter ROM
    - 00-13-72-A4-96-BE
  - Manufacturer buys portion (2^{24} addresses) of MAC address space (to assure uniqueness)
    - First 24 bits identify the manufacturer
      - » Organizationally Unique Identifier (OUI)
      - » http://standards.ieee.org/regauth/oui/oui.txt
      - » 00-13-72 (hex)
      - » Dell Inc. One Dell Way, Round Rock Texas 78682
    - Last 24 bits is device ID assigned by manufacturer
- Why have a separate MAC address?
  - Can use adapter on non-IP networks
  - Adapter can filter out frames without passing data up stack

Addresses


What's your address?
173.168.15.10
No, your local address.
127.0.0.1
I mean your physical address.
29:01:38:62:31:58

MAC Addresses

- Each adapter (NIC) on LAN has a unique MAC address

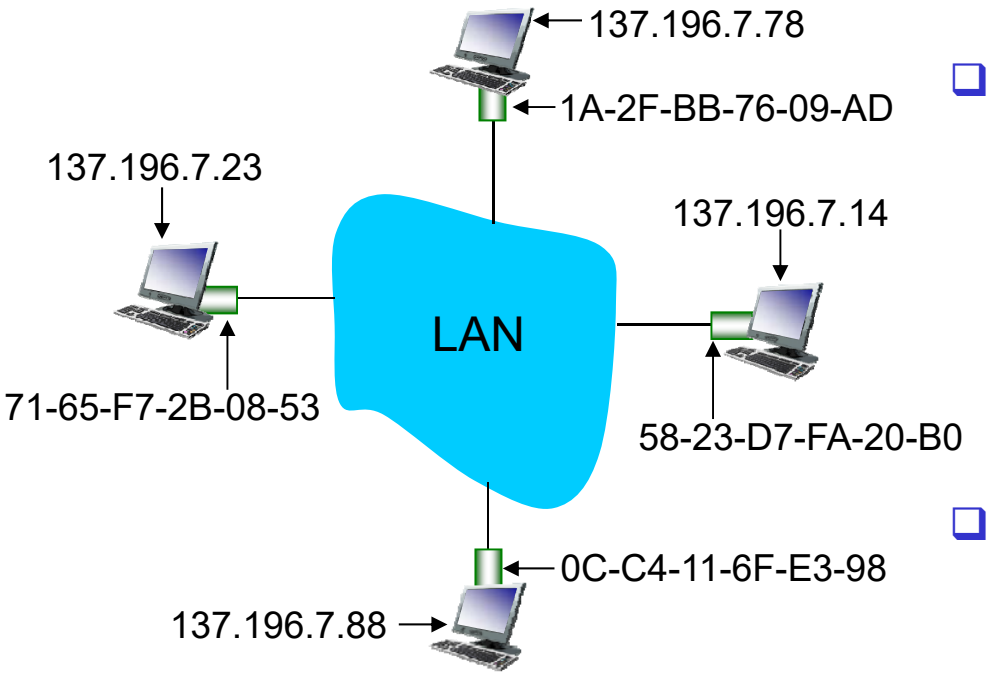


Broadcast address =
FF-FF-FF-FF-FF-FF

adapter

ARP: Address Resolution Protocol

Question: How to determine MAC address of B if we know B's IP address?



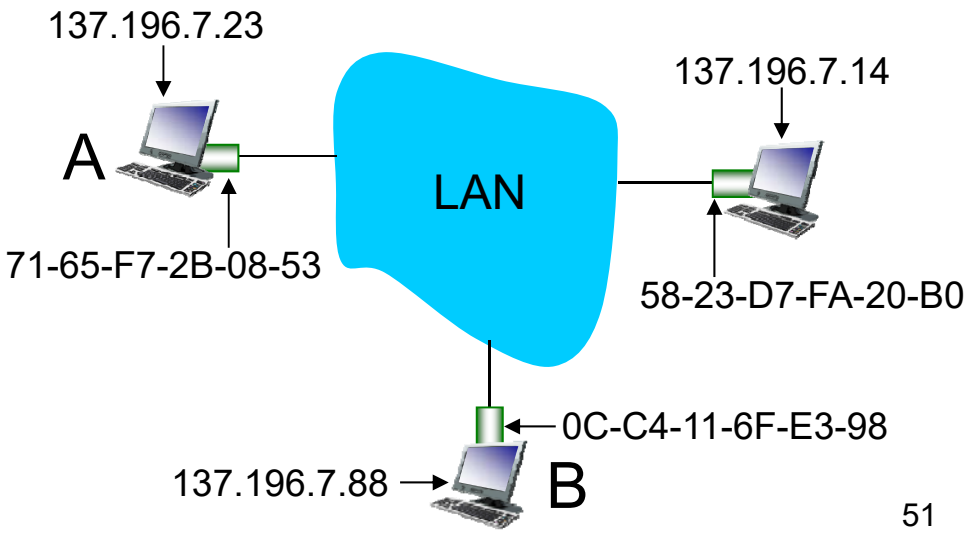
- ARP resolves IP addresses to MAC addresses within a subnet
- Each IP node (Host, Router) on LAN has an ARP table
- ARP Table: IP/MAC address mappings for some LAN nodes

  < IP address; MAC address; TTL>

  - TTL (Time To Live): time after which address mapping will be forgotten (typically 20 min)
- `arp -a` displays current arp table in Windows

ARP Protocol: Same LAN (Network)

- A wants to send datagram to B, and B's MAC address not in A's ARP table.
- A broadcasts ARP query packet containing B's IP address
  - Dest MAC address = FF-FF-FF-FF-FF-FF
  - All nodes on LAN receive ARP query
- B receives ARP query packet & replies to A with its (B's) MAC address
  - Frame sent directly to A's MAC address (unicast)
- A caches (saves) IP-to-MAC address pair in its ARP table until information becomes old (times out)
- ARP is "plug-and-play":
  - Nodes create their ARP tables without intervention from an administrator

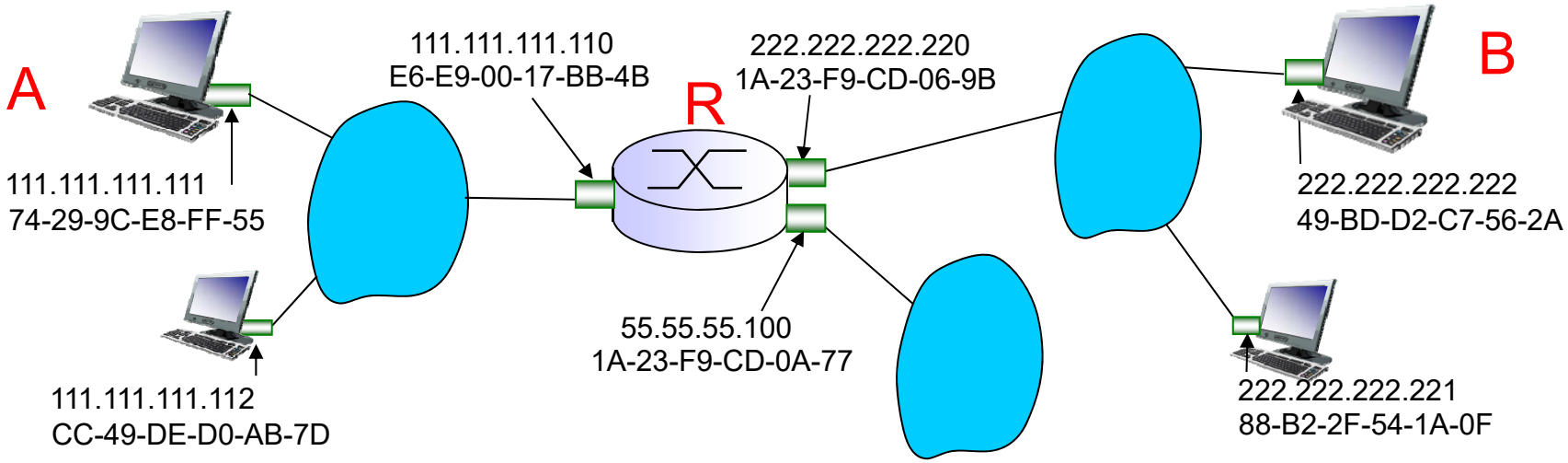


Routing to Another LAN

Walkthrough: Send datagram from A to B via R

assume A knows B IP address via DNS

- Three ARP tables in router R, one for each IP subnet (LAN)

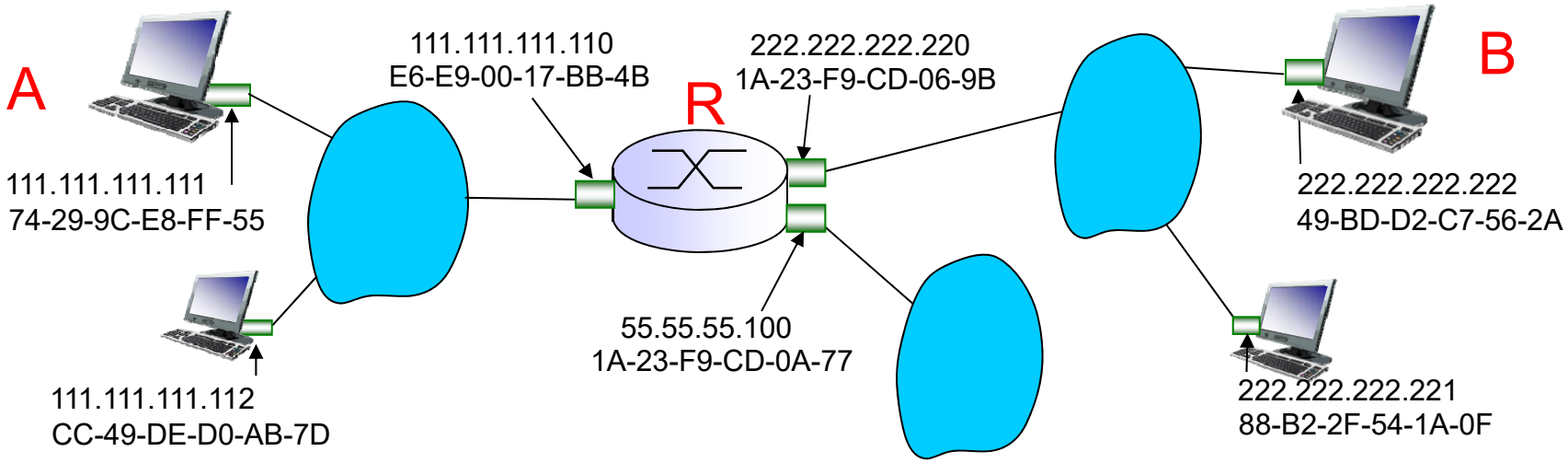


Subnet mask: 255.255.255.0

Routing to Another LAN

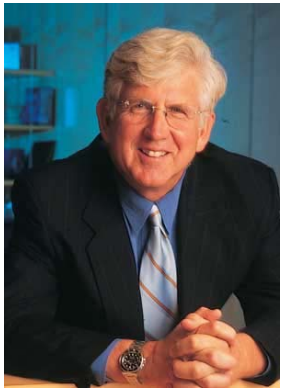
How does A know to send the frame to R?

- A creates datagram with source IP-A, destination IP-B
- A broadcast ARP request; R unicasts reply of MAC address for 111.111.111.110
- A creates link-layer frame with R's MAC address as dest, frame contains A-to-B IP datagram
- A's adapter sends frame, and R's adapter receives frame
- R removes IP datagram from Ethernet frame, sees dest IP address is B
- R determines the correct subnet interface based on the dest IP
- R uses ARP to get B's MAC address on appropriate interface
- R creates frame containing A-to-B IP datagram and sends to B



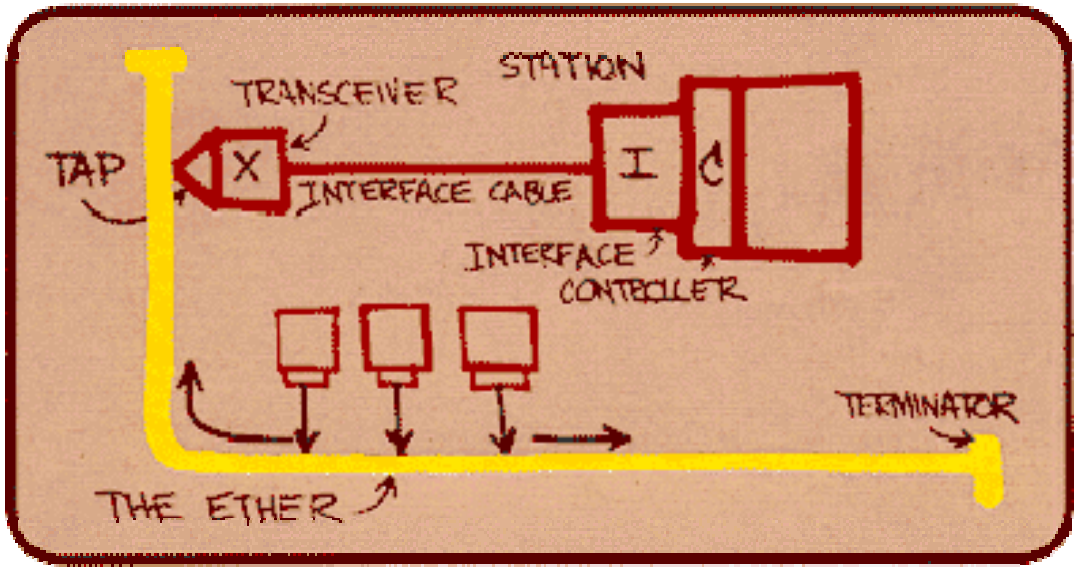
Subnet mask: 255.255.255.0

Ethernet



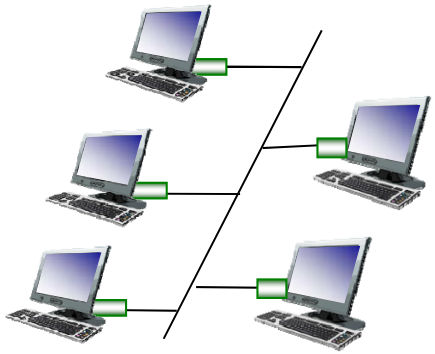
- Ethernet refers to the family of LAN protocols covered by the IEEE 802.3 standard
- Developed by Bob Metcalfe in 1972 based on ALOHAnet
- First widely used and still dominant LAN technology
- Cheap $5.13 for 1 Gbps card!
- Kept up with speed race: 10 Mbps – 100 Gbps

Metcalfe's Ethernet sketch

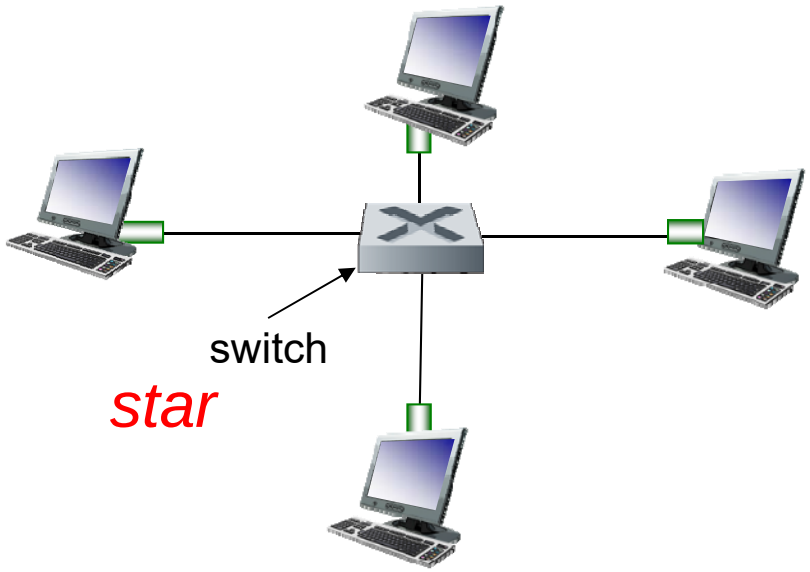


Ethernet Topologies

- Bus topology popular through mid 90s
  - All nodes in same collision domain (can collide with each other)
  - Channel is half-duplex → a node can either transmit or receive but not both at the same time
- Star topology now prevails
  - Active *switch* in center
  - Each "spoke" runs a (separate) Ethernet protocol (nodes do not collide with each other)
  - Each node can transmit and receive at the same time on a pair of dedicated twisted wires – full duplex and no collisions!



bus: coaxial cable



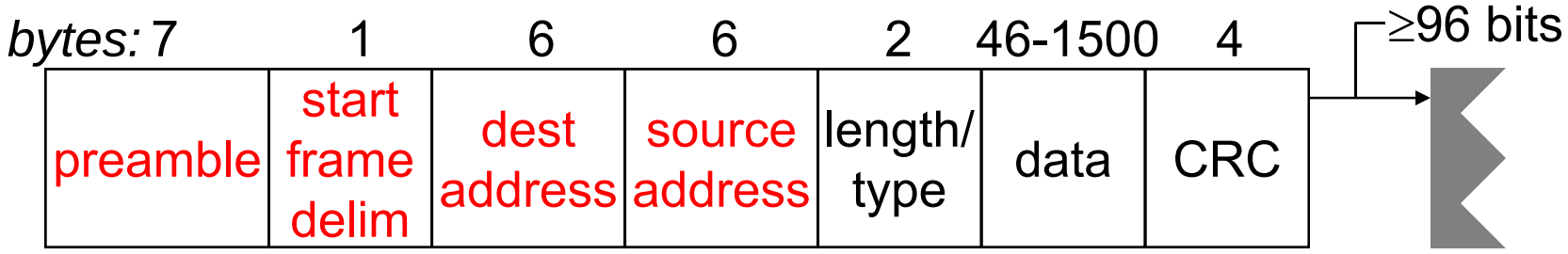
star

Ethernet: MAC Layer

- Data encapsulation
  - Frame Format
  - Addressing
  - Error Detection

- Link Management
  - CSMA/CD
  - Backoff Algorithm

Ethernet Frame Structure



Sending adapter encapsulates IP datagram in Ethernet frame

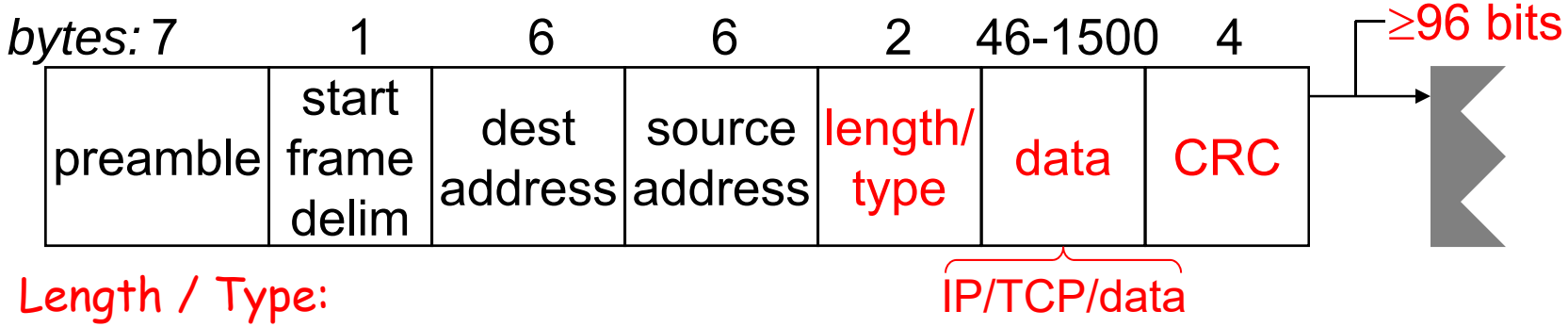
Preamble: Alternating 1's and 0's for synchronization

Start frame delimiter (SFD): 10101011 to start frame

Destination address: Unique 48-bit MAC address of destination

Source address: Unique 48-bit MAC address of source station

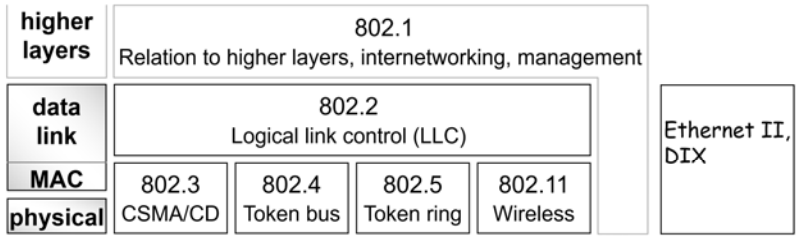
Ethernet Frame Structure



- Length / Type:
  - 0 : 1500 (0x0000 – 0x05DC) → length of data (802.3 frame)
  - > 1535 (0x0600) → type of higher layer protocol (Ethernet II)
    - Typically 0x0800 for IP; Length determined using framing
- Data: Min = 46 (pad with 0's); Max = 1500 data bytes (MTU = 1500 B)
  - To ensure that no node can completely receive a frame before the transmitting node has finished sending it, Ethernet defines a minimum frame size
- CRC: CRC over address, length, and data fields
- Minimum frame length = 72 bytes → 64 bytes (512 bits) for header/data + 8 bytes for preamble/SFD
- Interframe gap (IFG) of at least 96 bits

IEEE 802.3 CSMA/CD LAN

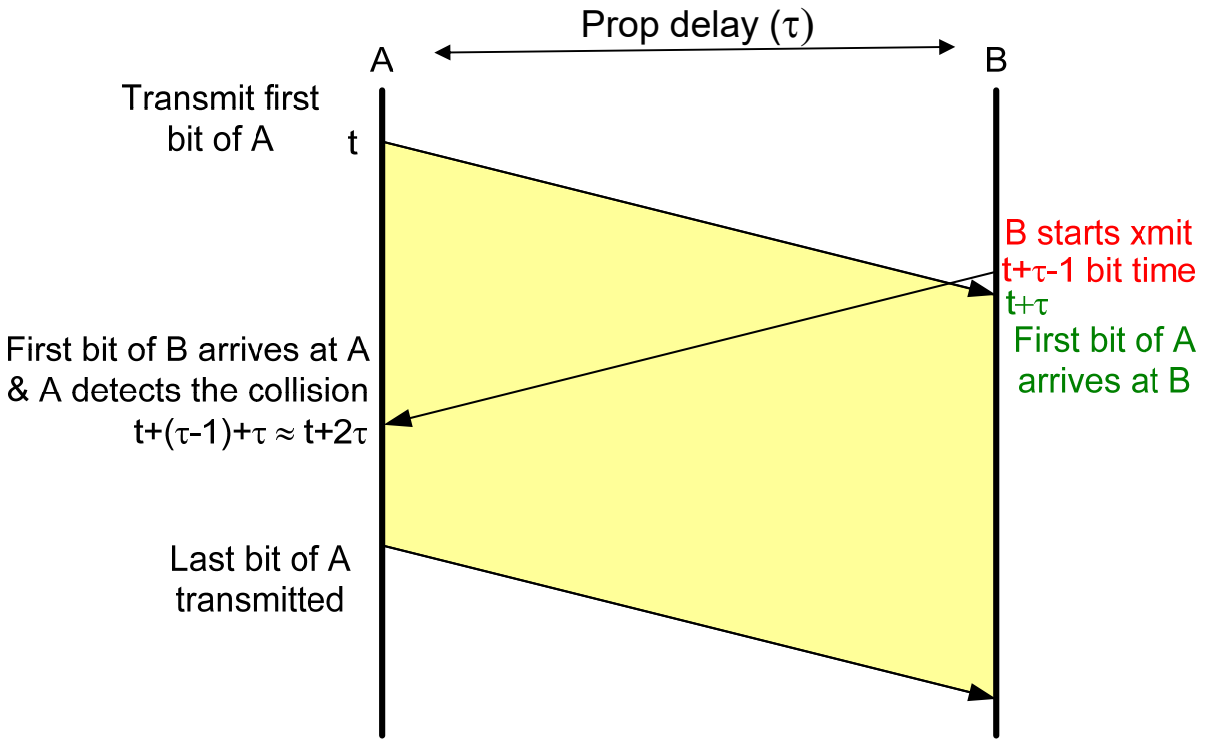
- IEEE 802.3 and Ethernet are nearly identical
- "Ethernet" (Ethernet II, DIX)
  - Ethernet developed during the mid-1970's at Xerox Palo Alto Research Center with the objective to share resources such as printers
  - Later refined by Digital Equipment Corporation, Intel, and Xerox (DIX standard)
  - An industry standard from 1982 that is based on the first implementation of CSMA/CD by Xerox
  - Predominant version of CSMA/CD in the US
- 802.3:
  - IEEE's version of CSMA/CD from 1983
  - Interoperates with 802.2 (LLC) as higher layer



- Difference for our purposes: Ethernet and 802.3 use different methods to encapsulate an IP datagram

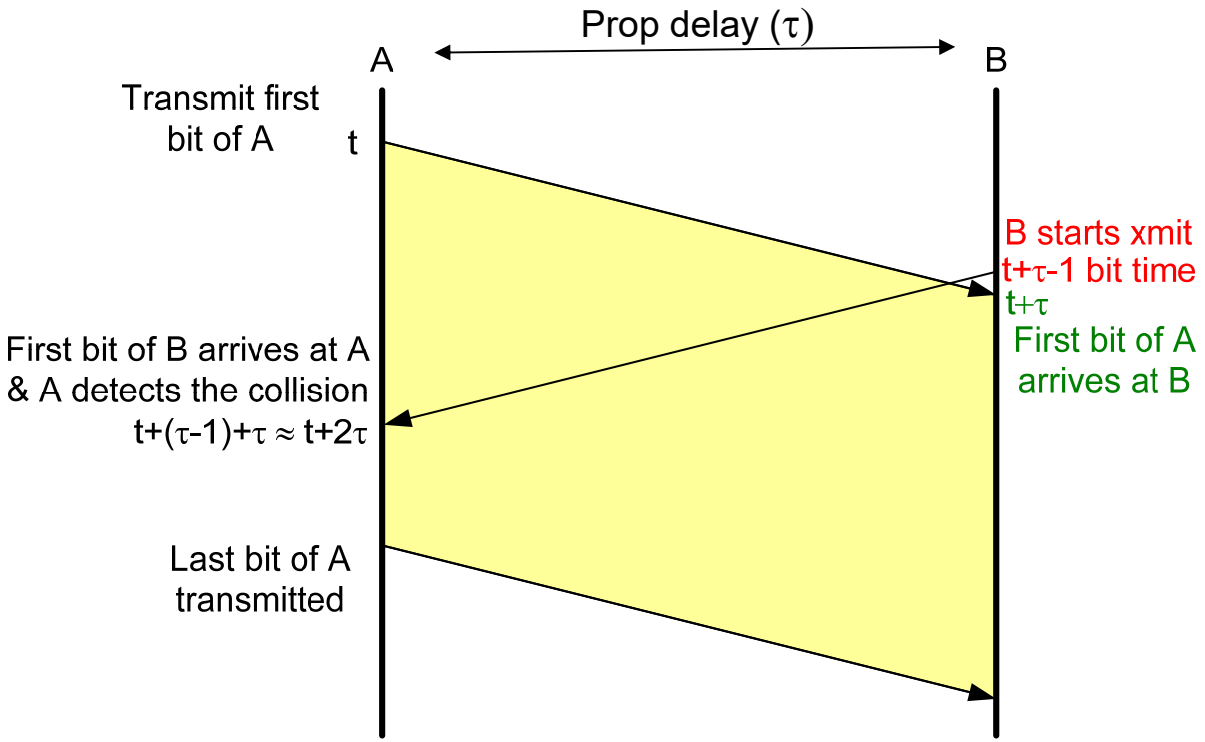
Limitations on Ethernet Length

- Prop delay plays a large role in determining the physical length of link
  - Suppose A sends a packet at time t
  - and B sees an idle line at a time just before t + τ (i.e., t + τ - 1 bit time)
  - … so B happily starts transmitting a packet at t + τ - 1 bit time
- B detects a collision and sends jamming signal at t + τ
  - but A doesn't see collision until t + 2τ - 1 bit time



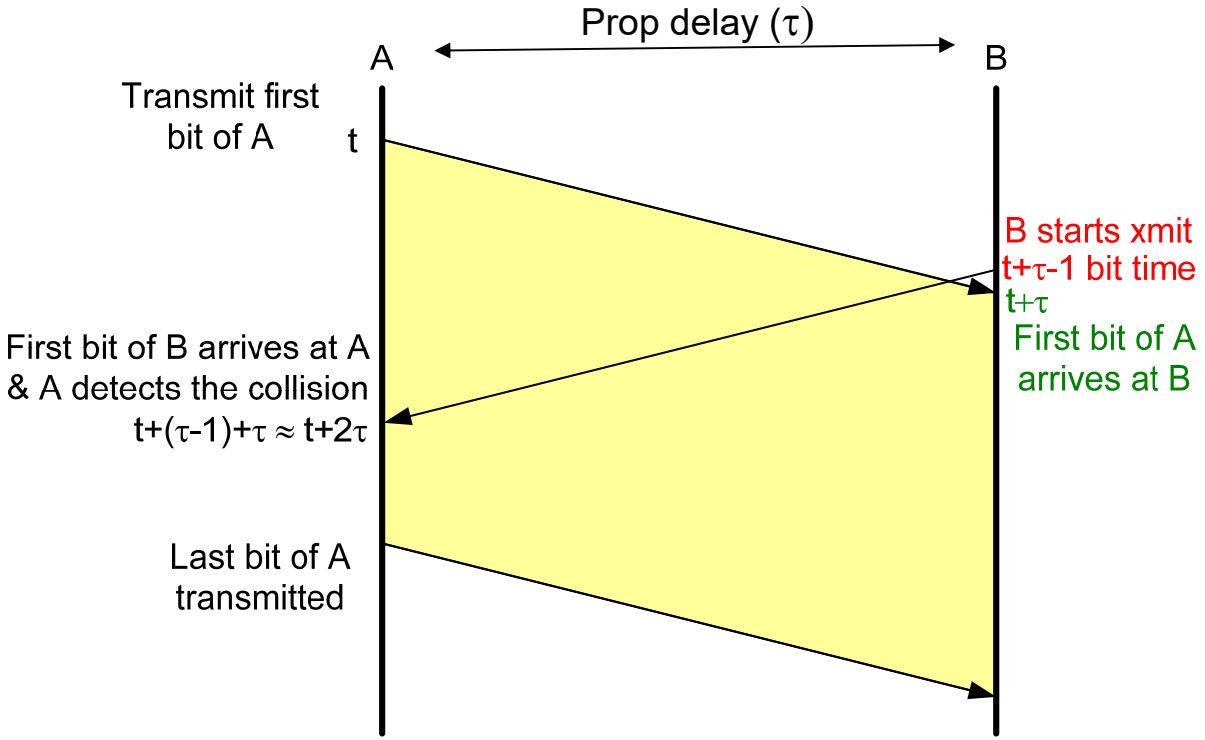
Limitations on Ethernet Length

- A must transmit at least 2τ in order to guarantee to see a collision

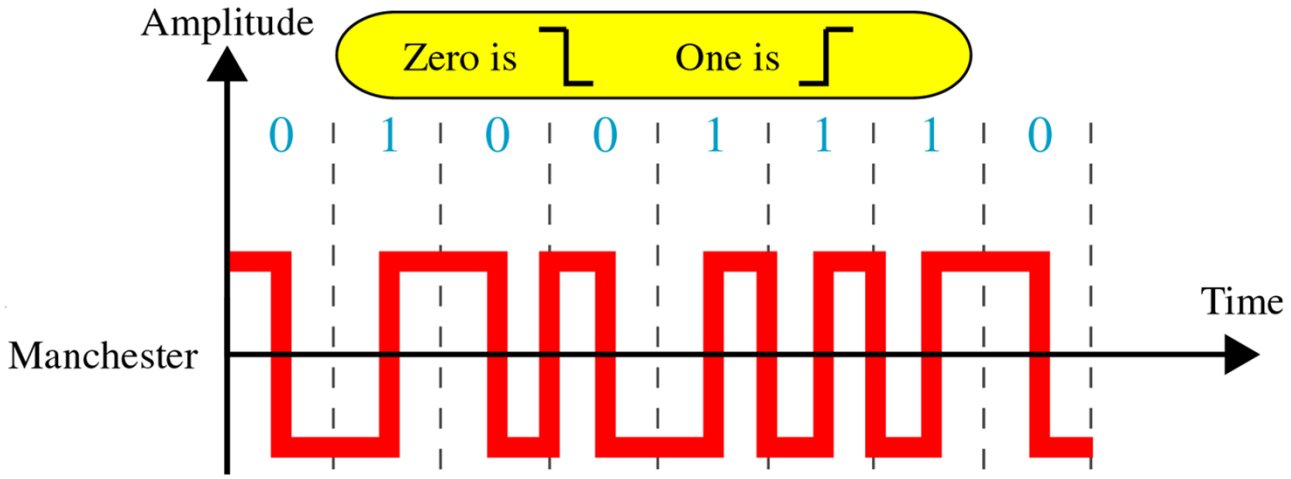


Limitations on Ethernet Length

- Imposes restrictions on Ethernet
  - Maximum length between nodes: 2500 meters
  - Minimum length of the frame: 512 bits (64 bytes) for header/data
    - 512 bits is also known as the slot time
    - See "B.1.3 Minimum frame length determination" in IEEE 802.3-2008 for description and calculation



Manchester Encoding



- Used in 10BaseT
- Each bit has a transition
- Allows clocks in sending and receiving nodes to synchronize to each other
  - No need for a centralized, global clock among nodes!

Ethernet - Unreliable, Connectionless Service

- Unreliable: receiving adapter doesn't send acks or nacks to sending adapter
  - If NIC transmits the entire frame without hearing a collision, it assumes success and moves on to the next frame
  - Frame is discarded if CRC fails
  - Stream of datagrams passed to network layer can have gaps due to CRC failure
  - Gaps will be filled if application at receiver is using TCP
    - Otherwise, application will see the gaps

- Connectionless: No handshaking between sending and receiving adapter

Ethernet Uses CSMA/CD

- Transmit at any time
  - No "slots"
- Carrier sense
  - Adapter doesn't transmit if it senses that some other adapter is transmitting
- Collision detection
  - Transmitting adapter aborts when it senses that another adapter is transmitting
- Before attempting a retransmission, adapter waits a random time
  - Random access

Ethernet CSMA/CD Algorithm

1. NIC receives datagram from net layer & creates frame

2. a. If NIC senses channel idle for 96 bit times (called the interframe gap), it starts to transmit frame (1-persistent)

   b. If it senses channel busy, waits until channel idle and then transmits

3. If NIC transmits entire frame without detecting another transmission, the NIC is done with frame !

4. If NIC detects another transmission while transmitting, aborts and transmits a 32-bit jam signal → typically all 1's
   - Ensures all other xmits are aware of collision
   - Destroys CRC → stations receiving frame drop it

5. After aborting, NIC enters exponential backoff:
   After the n^{th} collision, NIC chooses a K at random from $\{0,1,2,3,4,\ldots,2^m-1\}$ where $m = \min(n,10)$
   - NIC waits K·512 bit times and returns to Step 2

Ethernet's CSMA/CD (more)

Bit time:

transmission time for 1 bit

1 bit/10 Mbps = 0.1 microsec for 10 Mbps Ethernet

If K = 1023, wait time is
= (1023 * 512) / 10 x 10^6
= 52 msec

Exponential Backoff:

- *Goal*: adapt retransmission attempts to current load
  - Heavy load: random wait will be longer
- 1st collision: choose K from {0,1}; delay is K· 512 bit times
- After 2nd collision: choose K from {0,1,2,3}
- After 3rd collision: choose K from {0,1,2,3,4,5,6,7}
- After 10–15 collisions, choose K from {0,1,2,3,4,…,1023}

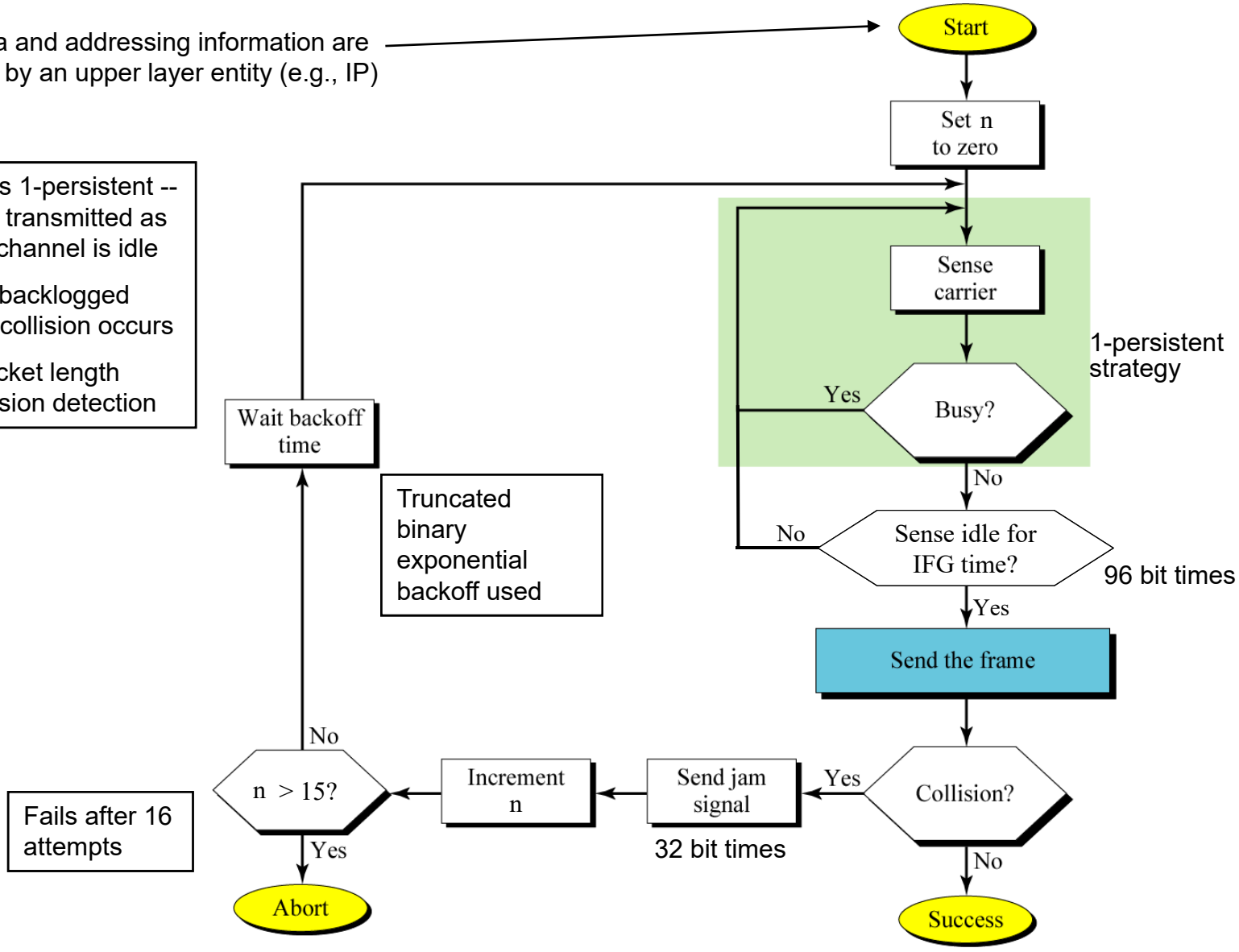
IEEE 802.3 Operation

New data and addressing information are provided by an upper layer entity (e.g., IP)

IEEE 802.3 is 1-persistent -- new packets transmitted as soon as the channel is idle

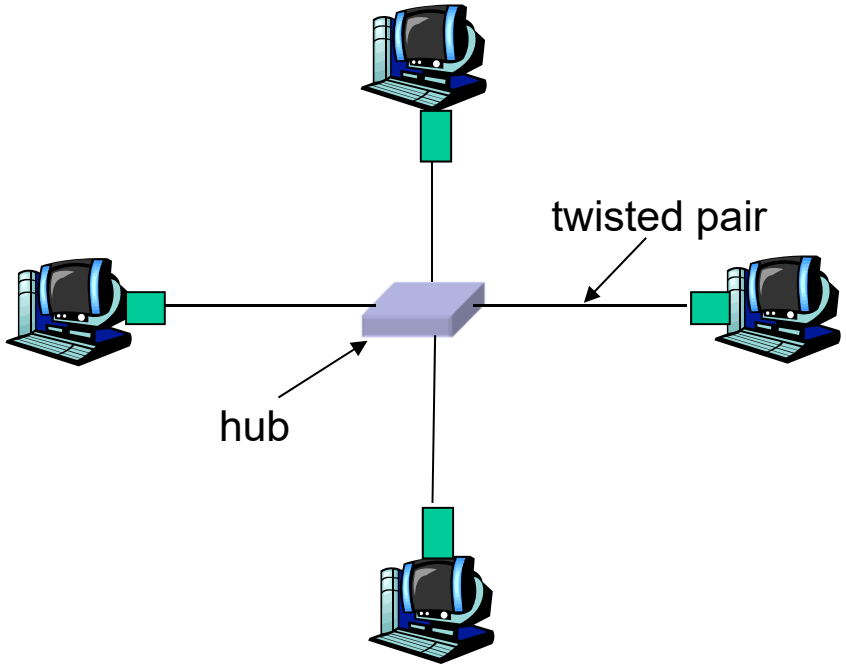
Packets are backlogged only when a collision occurs

Minimum packet length ensures collision detection



10BaseT / 100BaseT / 1000BaseT

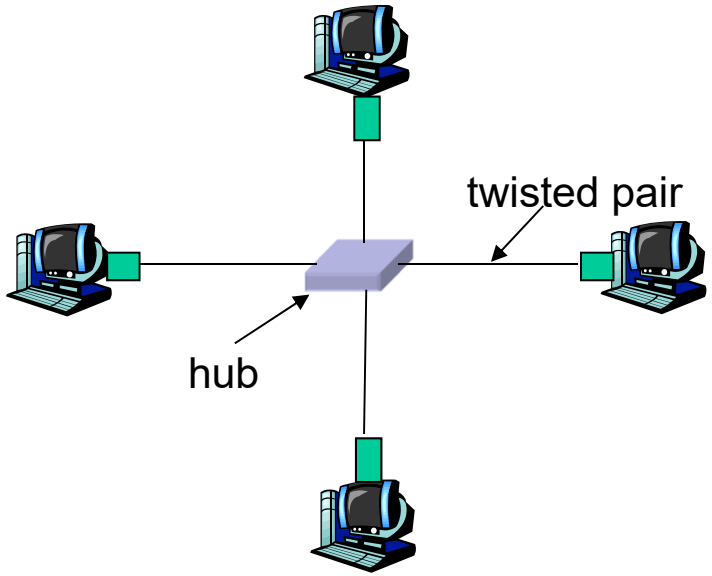
- 100BaseT → 100 Mbps, "Fast Ethernet", 2 pairs of wires in Cat5
- 1000BaseT → 1Gbps, uses all four pairs of wires in Cat 5/6
- Nodes connect to a hub: "star topology"; 100 m max distance between nodes and hub



Woot! High Speed!
…wait what?

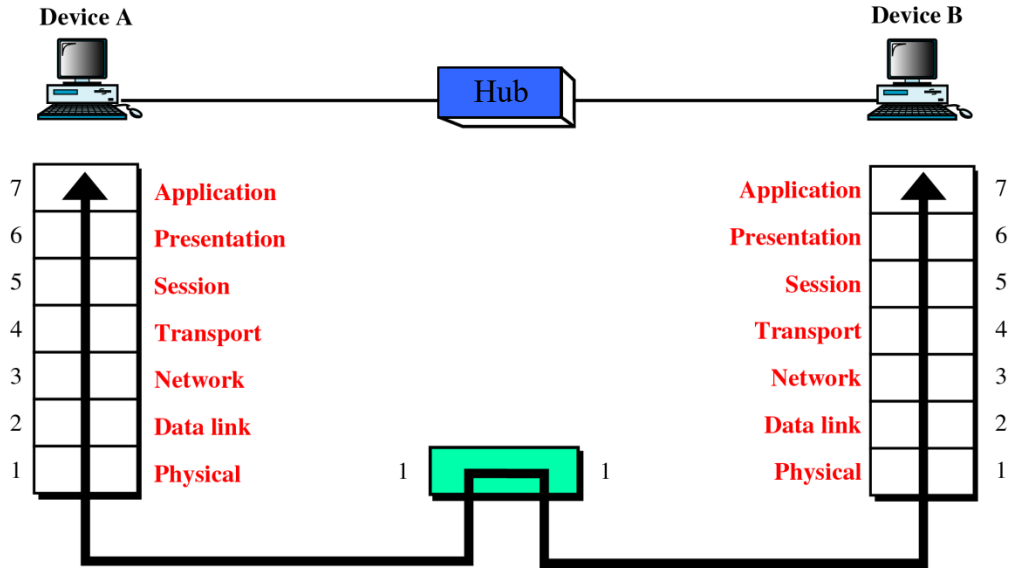
Hubs / Repeaters

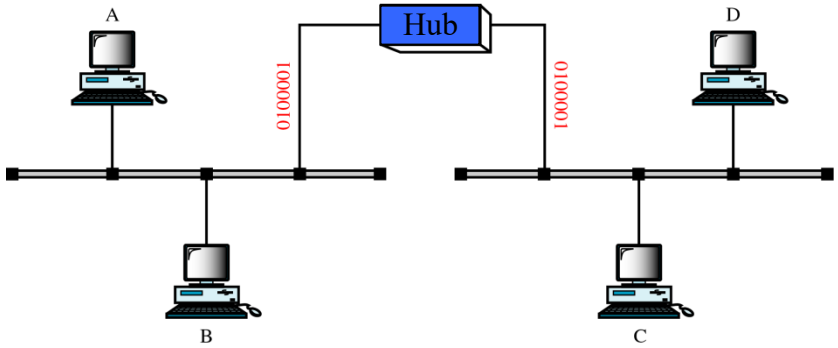
- Hubs are essentially physical-layer repeaters
  - Provides a limited "extension cord" for Ethernet
  - Hardware device that copies electrical signals from one link to all other links at the same rate
    - ONE collision domain
  - No frame buffering
  - No CSMA/CD at hub → host NICs detect collisions



Hubs / Repeaters

- Does not change the functionality of the network
- Operates only at the physical layer
  - Regenerates the signal to abate attenuation

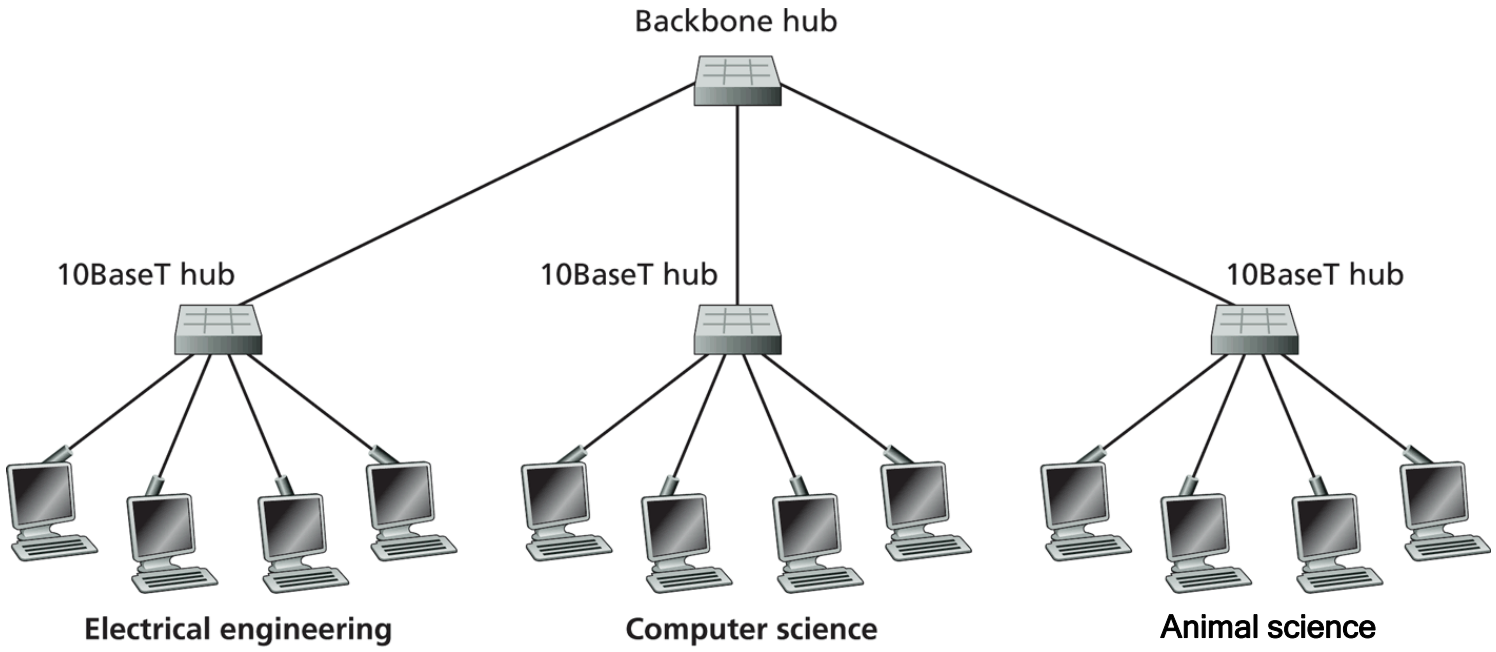




Frame sent from A to B will also be heard by C and D

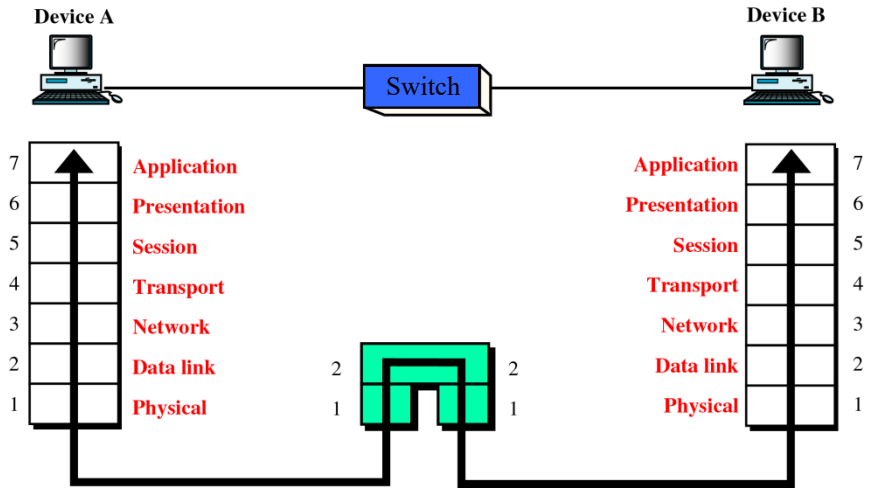
Interconnecting With Hubs

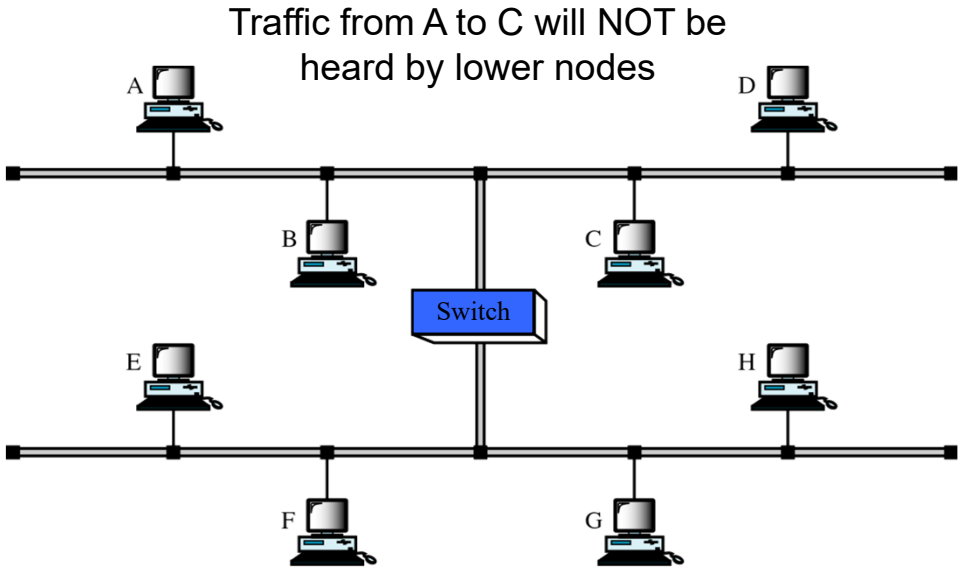
- Individual segment collision domains become 1 large collision domain
  - Without Backbone hub → max aggregate throughput = 30 Mbps
  - With Backbone hub → max aggregate throughput = 10 Mbps
- Can't interconnect 10BaseT and 100BaseT with a hub



Switch / Bridge

- Link layer device
- Divides a large network into smaller segments
- Keeps traffic for each segment separate
  - Multiple collision domains – reduces collisions
- Store and forward complete packets – discards packets with errors
- Examines frame header and selectively forwards frame based on dest MAC address





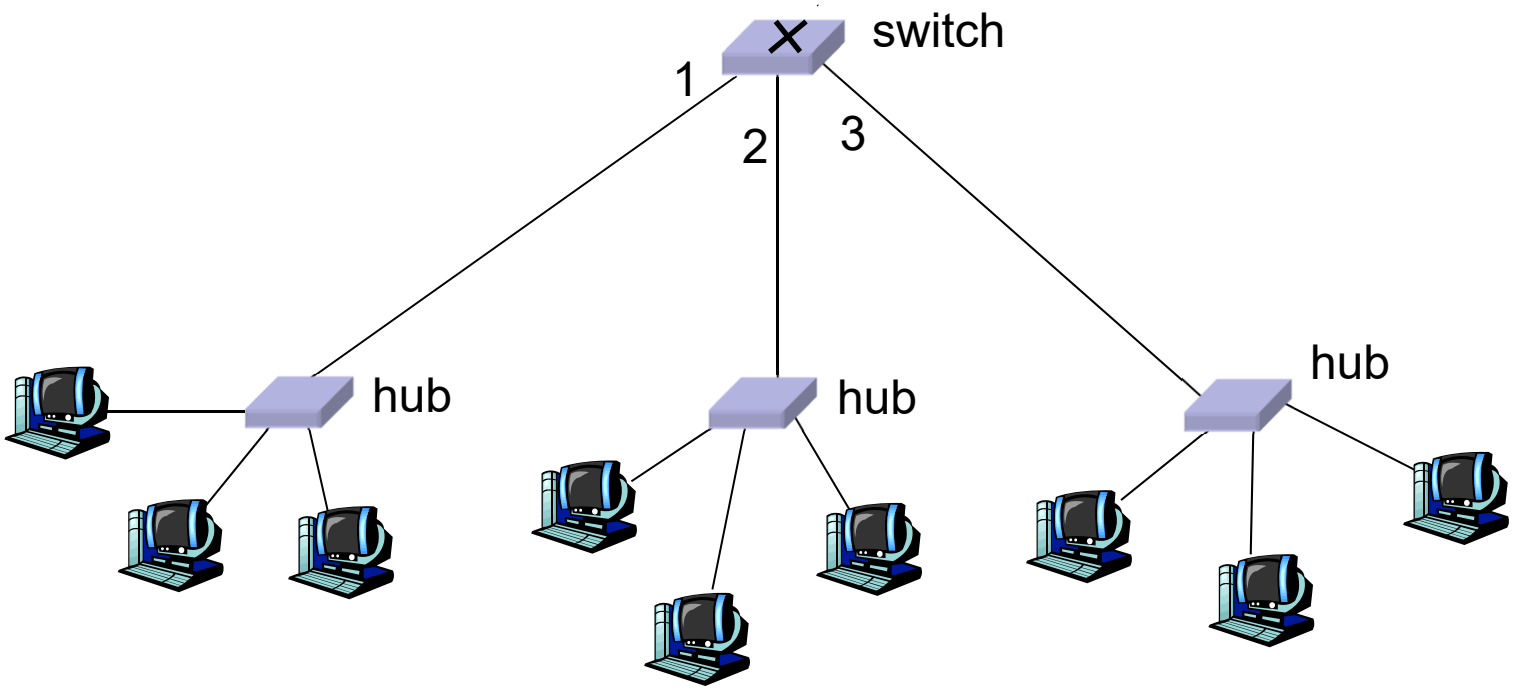
Why Ethernet Switching?

- LANs may grow very large
  - Switch has a very fast backplane
  - It can forward frames very quickly to the appropriate collision domain
- Cheaper than upgrading all host interfaces to create a faster network
  - Combinations of shared/dedicated, 10/100/1000 Mbps interfaces possible
- No limit on the number of switches between hosts

Switch Characteristics

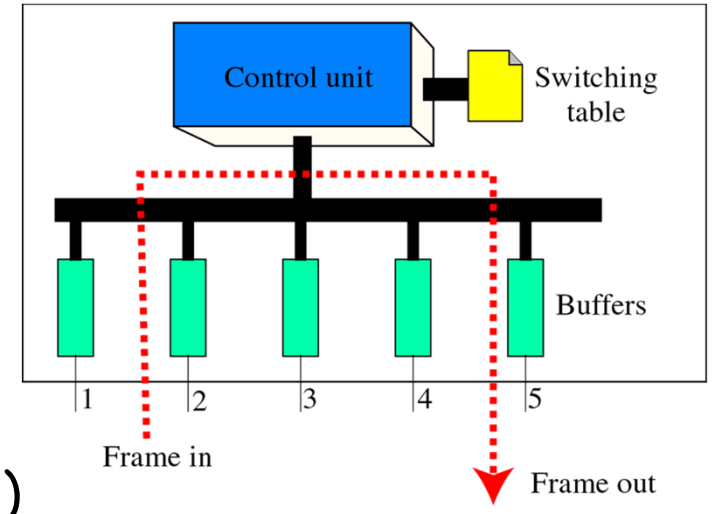
- When frame is to be forwarded on segment, uses CSMA/CD to access segment
- Uses exponential backoff if collision occurs
- Transparent
  - Hosts are unaware of presence of switches
- Plug-and-play, self-learning
  - Switches do not need to be configured
- Not considered an interface / adapter
  - They do not have MAC addresses
- Cut-through switching possible: frame forwarded from input to output port without first collecting entire frame
  - Error detection not possible since entire frame not in buffer
  - Slight reduction in latency

Forwarding



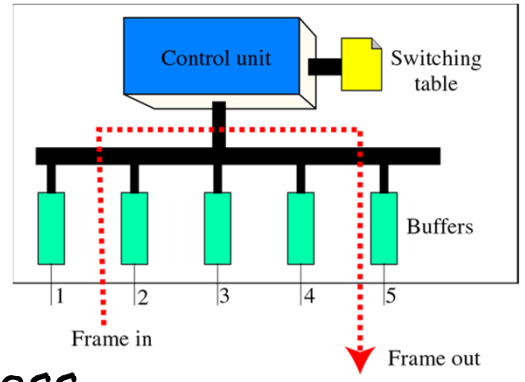
- How does the switch determine onto which LAN segment to forward frame?
- Looks like a routing problem...

Self Learning



- A switch has a switching table
- Entry in switching table:
  - (MAC Address, Interface, Time Stamp)
  - Stale entries in table dropped (TTL can be 60 min)
- Switch *learns* which hosts can be reached by which interfaces
  - When frame received, switch "learns" location of sender's MAC address → incoming LAN segment
  - Records sender MAC / location pair in switch table

Filtering/Forwarding



When switch receives a frame:

Record link associated with sending host's MAC address

Index switch table using dest MAC address

if entry found for destination

then

{

if dest is on segment from which frame arrived

then drop the frame

else forward the frame on interface listed in table

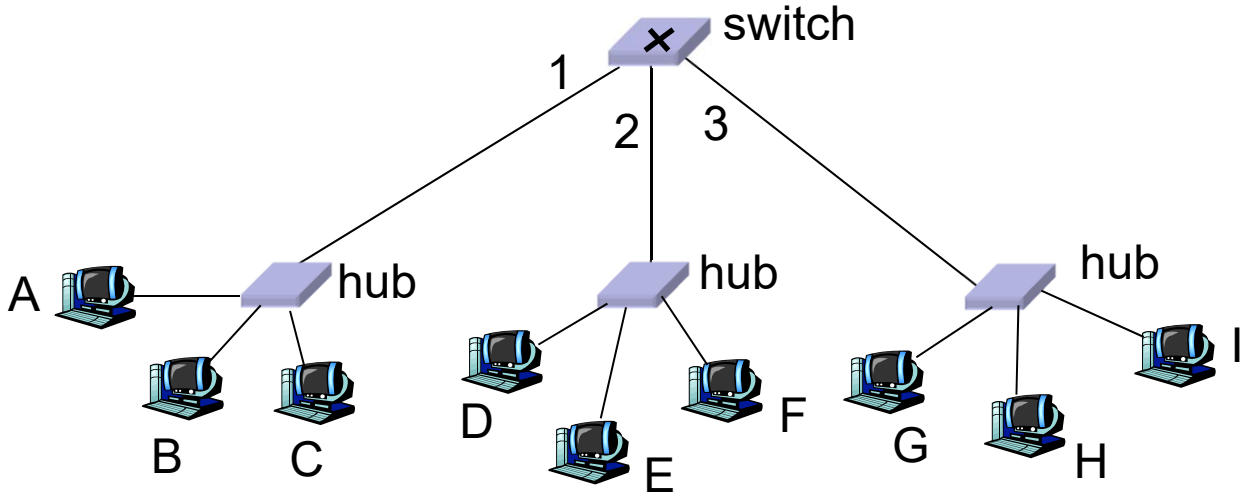
}

else flood

Forward on all but the interface on which the frame arrived

Switch Example

- Suppose C sends frame to D



Before

address	interface
A	1
B	1
E	2
G	3

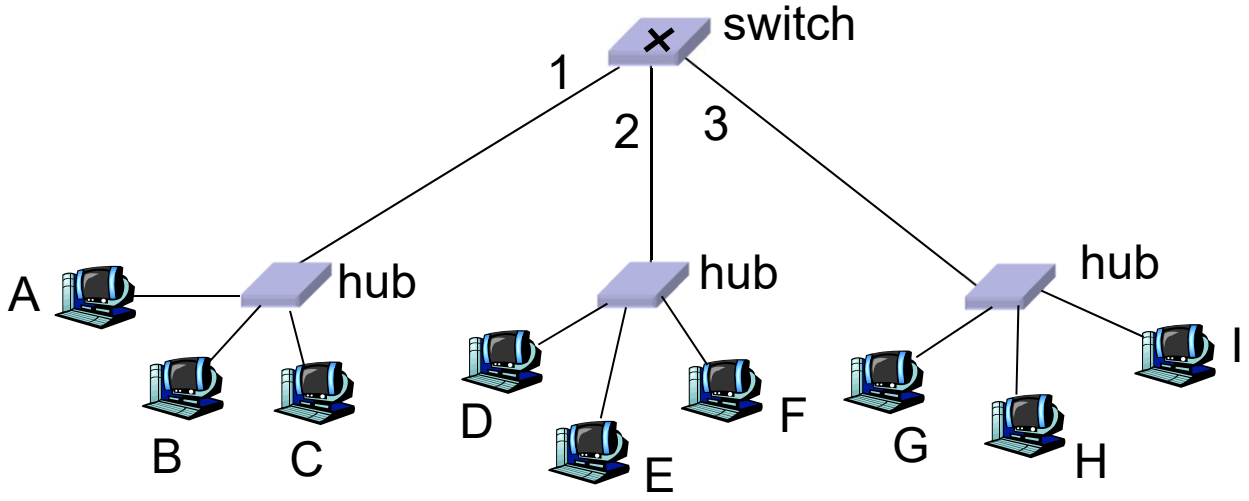
After

address	interface
A	1
B	1
E	2
G	3
C	1

- Switch receives frame from C
  - Notes in switch table that C is on interface 1
  - Because D is not in table, switch forwards frame to interfaces 2 and 3
- Frame received by D

Switch Example

- Suppose D replies back with frame to C



Before

address	interface
A	1
B	1
E	2
G	3
C	1

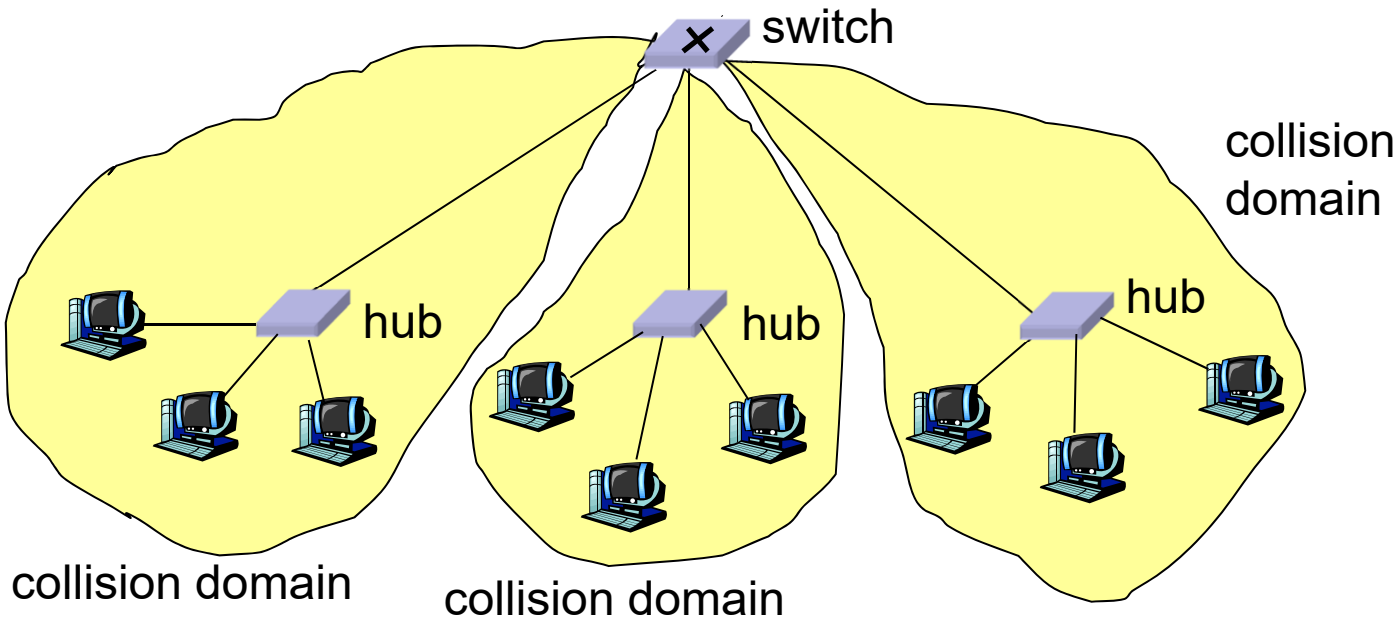
After

address	interface
A	1
B	1
E	2
G	3
C	1
D	2

- Switch receives frame from D
  - Notes in switch table that D is on interface 2
  - Because C is in table, switch forwards frame only to interface 1
- Frame received by C

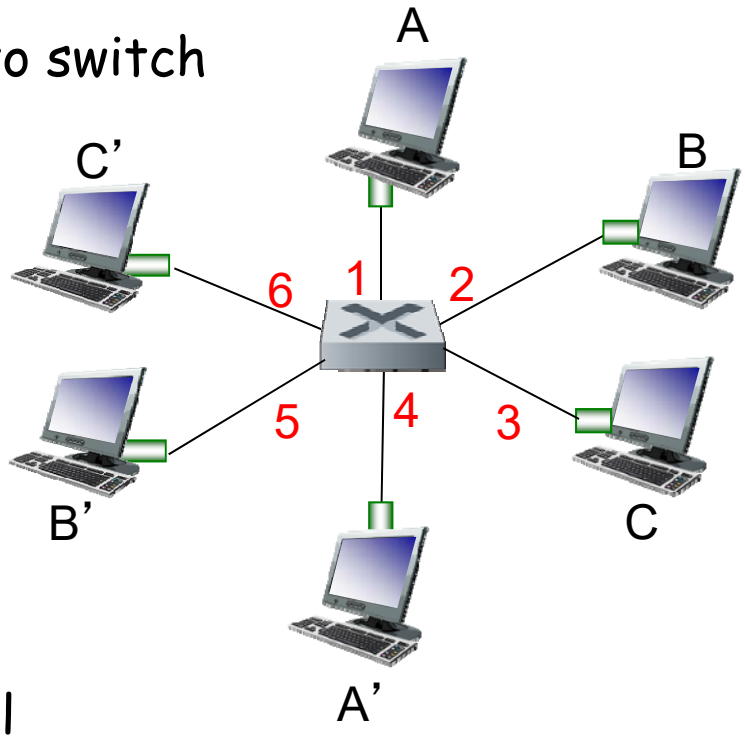
Switch: Traffic Isolation

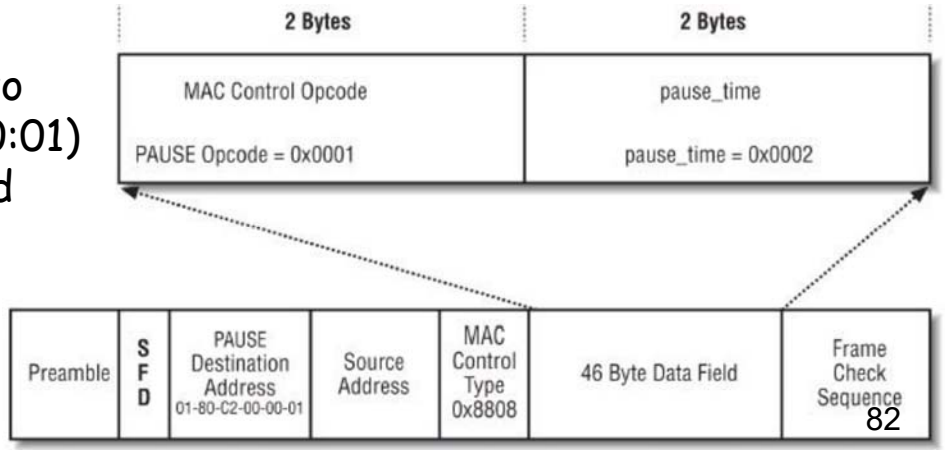
- Switch breaks subnet into LAN segments
- Switch filters packets:
  - SAME-LAN-SEGMENT frames not usually forwarded onto other LAN segments
  - Segments become separate collision domains



Switches: Dedicated Access

- Hosts have dedicated direct connection to switch
- Switch can buffer frames
- Each host is its own collision domain
- No collisions; full duplex
- Effectively have a point-to-point connection between all hosts
- Switching: A-to-A' and B-to-B' simultaneously, no collisions
  - Aggregate throughput is 20 Mbps
- Use a MAC control frame for flow control
  - PAUSE frames
  - Overwhelmed receiver sends frame to special MAC address (01:80:C2:00:00:01) specifying how long the sender should pause transmission



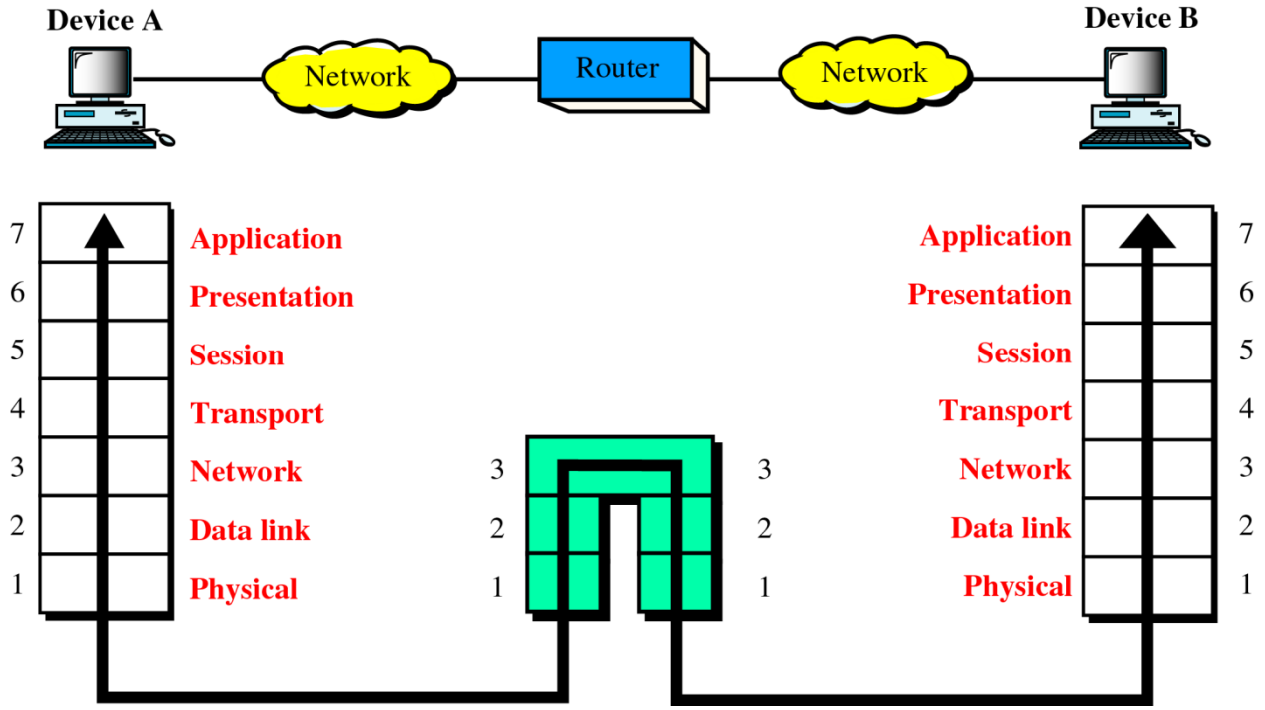


The Linksys SD205 Switch

- This newly-redesigned Linksys 5-Port 10/100 Switch can significantly increase your network traffic's speed.
- A switch serves the same function as a hub in a network design -- tying your network equipment together. But unlike a simple-minded hub which divides the network's bandwidth among all the attached devices, a switch delivers full network speeds at each port.
- Installing this cost-effective 5-Port 10/100 Switch can potentially increase your network speed by five times!
- It's the perfect way of integrating 10Mbps Ethernet and 100Mbps Fast Ethernet devices, too. All five ports are auto speed negotiating, and have automatic MDI/MDI-X crossover detection, so you don't have to worry about the cable type.
- Each port independently negotiates for best speed and half- or full-duplex mode, for up to 200Mbps of bandwidth per port.
- Fast store-and-forward switching prevents damaged packets from being passed on into the network.

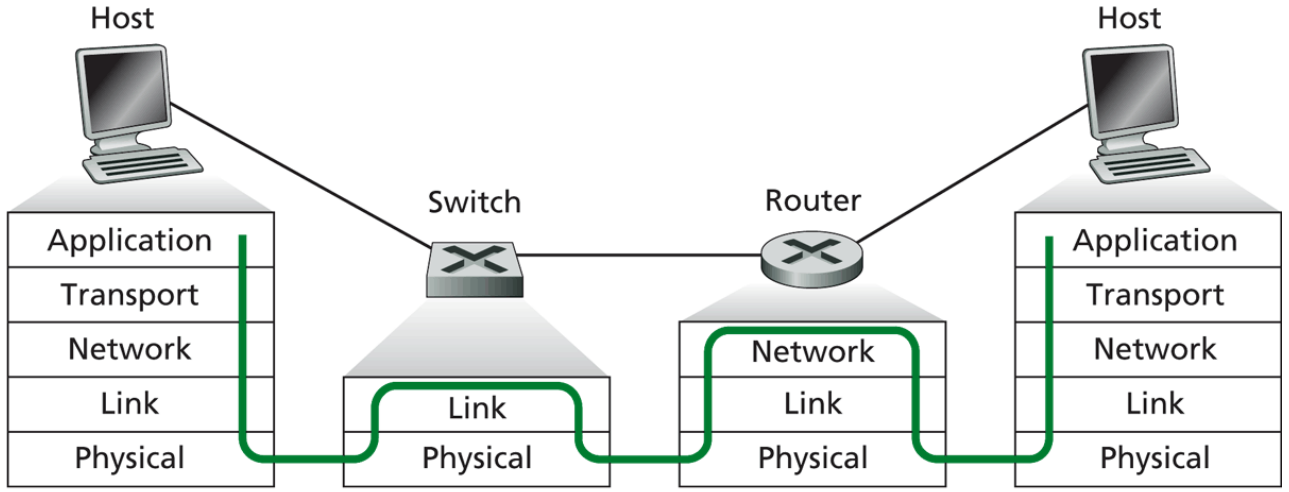
Network Hardware Terminology

- Router – a layer-3 packet device
  - Special purpose, dedicated computer that attaches to two or more networks and routes packets from one to another
  - Uses network layer (IP) addresses

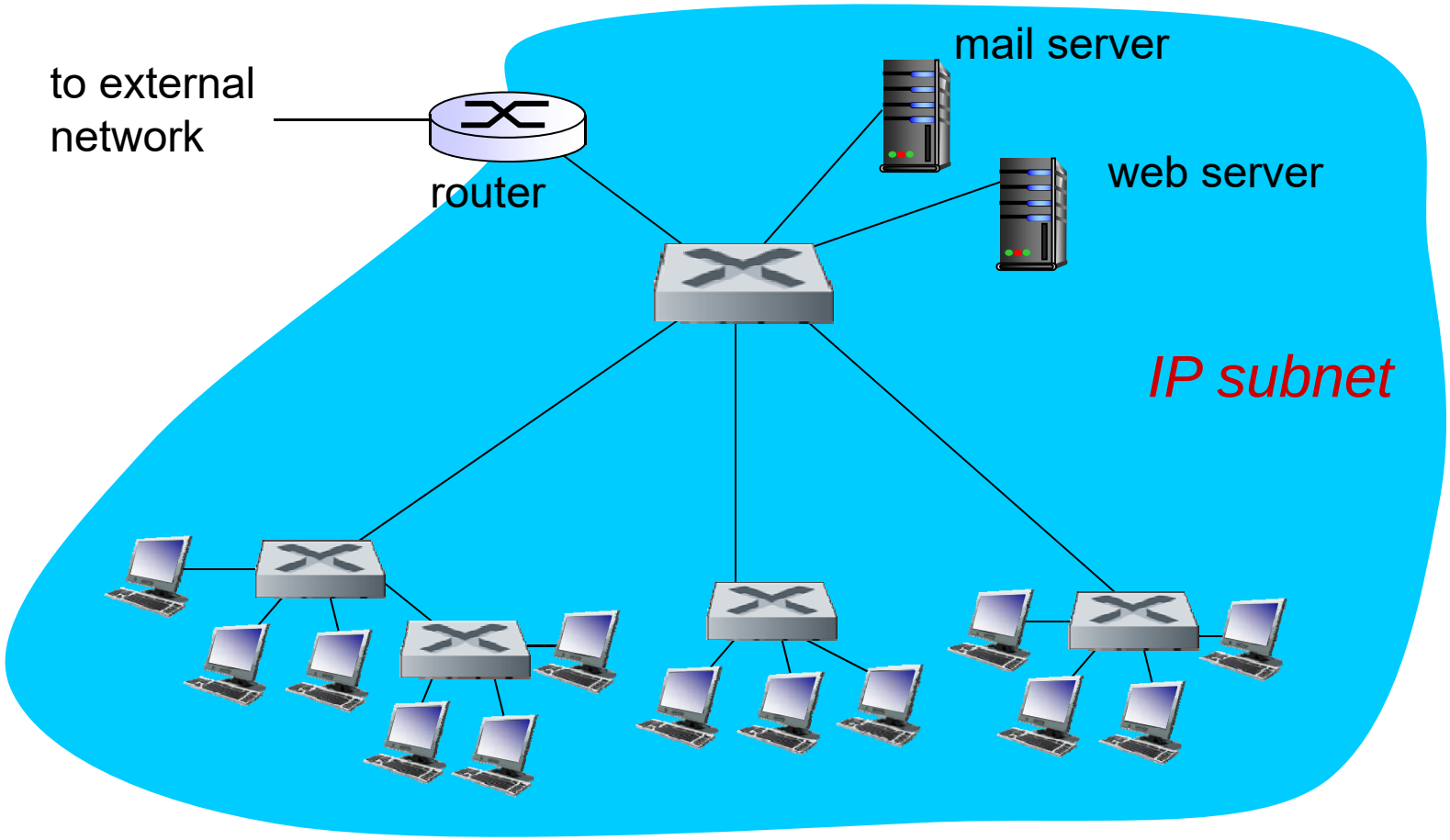


Switches vs. Routers

- Both store-and-forward devices
  - Routers: network layer devices (examine IP address)
  - Switches: link layer devices (examine MAC address)
- Routers maintain forwarding tables
  - Implement routing algorithms
- Switches maintain switching tables
  - Implement filtering, learning algorithms



Institutional Network



Point to Point Data Link Control (DLC)

- Commonly used in establishing a direct connection between 2 nodes
- One sender, one receiver, one link → easier than broadcast link
  - No Media Access Control
  - No need for explicit MAC addressing
  - e.g., dialup link, ISDN line, fiber optic lines (SONET)

- Popular point-to-point DLC protocols:
  - PPP – Point-to-Point Protocol
  - HDLC – High-level Data Link Control

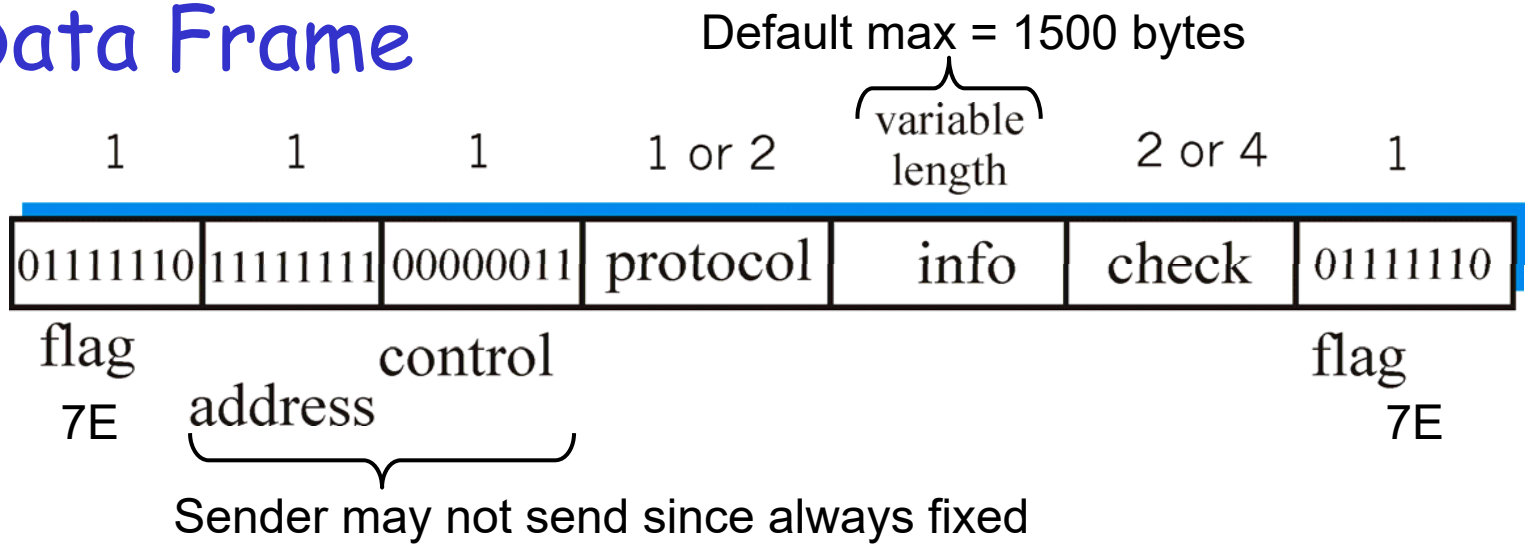
PPP Design Requirements [RFC 1547]

- Packet framing:
  - Encapsulation of network-layer datagram in data link frame
  - Receiver must be able to demultiplex upwards
- Bit transparency: must carry any bit pattern in the data field
- Error detection (no correction)

- No error correction/recovery
- No flow control
- Out of order delivery OK

Error recovery, flow control, data re-ordering
all relegated to higher layers!

PPP Data Frame



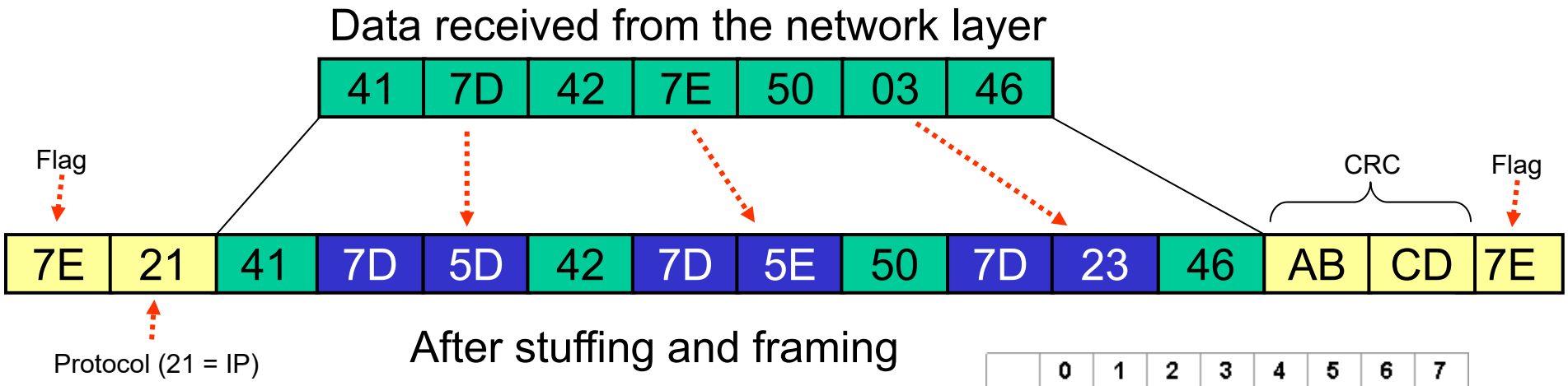
- Flag: delimiter (used for framing)
- Address: does nothing (only one option → 11111111)
- Control: does nothing; in the future possible multiple control fields
- Protocol: upper layer protocol to which frame delivered
  - IP, DECnet, AppleTalk, PPP-Link Control Protocol (PPP-LCP)
- Info: upper layer data being carried (e.g., IP datagram)
- Check: cyclic redundancy check for error detection

PPP Byte Stuffing

- "Data transparency" requirement: data field must be allowed to include the flag pattern <01111110>
  - Q: Is received <01111110> data or flag?

- Sender: adds ("stuffs") extra control escape byte 0x7D (01111101) before each 0x7E or 0x7D data or CRC byte
  - The next (data/CRC) byte has its sixth bit complemented:
    - Data/CRC byte 0x7E (01111110) becomes 0x5E (01011110) and transmitted as two bytes 0x7D, 0x5E
    - Data/CRC byte 0x7D (01111101) becomes 0x5D (01011101) and transmitted as two bytes 0x7D, 0x5D
- Receiver: removes control escape bytes (0x7D) and complements the sixth bit in the next byte
  - 0x7D, 0x5E becomes 0x7E
  - 0x7D, 0x5D becomes 0x7D

PPP Byte Stuffing



- By default, ASCII control codes (0x00 – 0x1F) are also escaped
  - For example, byte 0x03 is transmitted as two bytes 0x7D, 0x23
  - 00000011 → 00100011

	0	1	2	3	4	5	6	7
0	NUL	DLE	space	0	@	P	`	p
1	SOH	DC1 XON	!	1	A	Q	a	q
2	STX	DC2	"	2	B	R	b	r
3	ETX	DC3 XOFF	#	3	C	S	c	s
4	EOT	DC4	$	4	D	T	d	t
5	ENQ	NAK	%	5	E	U	e	u
6	ACK	SYN	&	6	F	V	f	v
7	BEL	ETB	'	7	G	W	g	w
8	BS	CAN	(	8	H	X	h	x
9	HT	EM	)	9	I	Y	i	y
A	LF	SUB	*	:	J	Z	j	z
B	VT	ESC	+	;	K	[	k	{
C	FF	FS	,	<	L	\	l	|
D	CR	GS	-	=	M	]	m	}
E	SO	RS	.	>	N	^	n	~
F	SI	US	/	?	O	_	o	del

PPP Data Control Protocol

Before exchanging network-layer data, data link peers must

- Configure PPP link using PPP-Link Control Protocol (LCP)
  - Bringing up, testing, bringing down lines; negotiating options
    - Max. frame length
    - Skip use of address and control fields
    - *Authentication*: key capability in ISP access
  - Behaves very much like TCP connection establishment
- Learn/Configure network layer information via a family of *Network Control Protocols* specific to different network layer protocols
  - For IP: carry IP Control Protocol (IPCP) msgs (protocol field: 8021) to configure/learn IP address

Link Layer

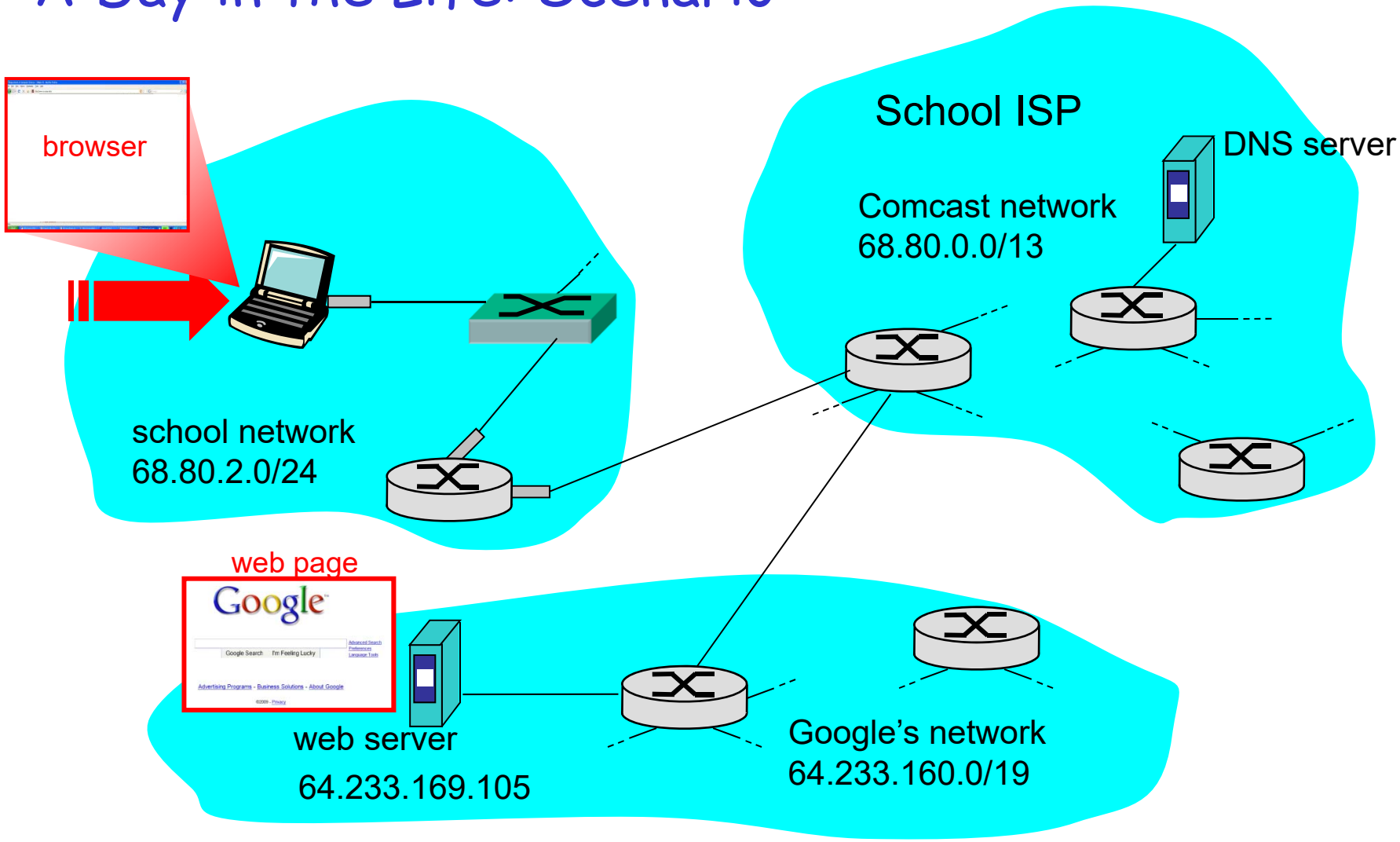
- 6.1 Introduction
- 6.2 Error Detection and Correction Techniques
- 6.3 Multiple Access Links and Protocols
- 6.4 Switched Local Area Networks
- 6.5 Link Virtualization
- 6.6 Data Center Networking
- 6.7 A Day in the Life of a Web Page Request

Synthesis: A Day in the Life of a Web Request

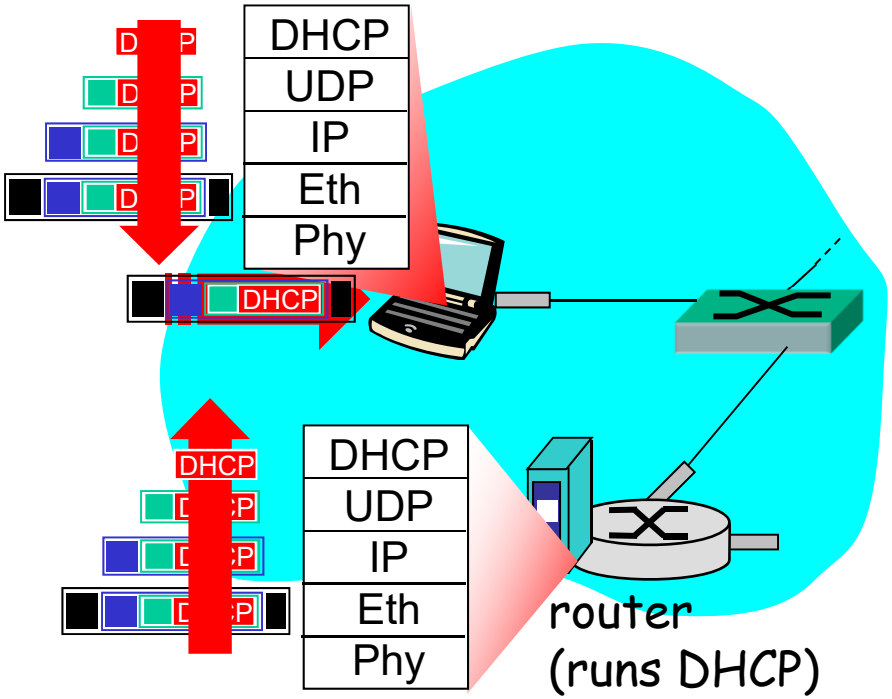
- Journey down protocol stack complete!
  - application, transport, network, link

- Putting-it-all-together: Synthesis!
  - *Goal:* identify, review, understand protocols (at all layers) involved in seemingly simple scenario:
    - Requesting www page
  - *Scenario:* student attaches laptop to campus network, requests/receives www.google.com

A Day in the Life: Scenario



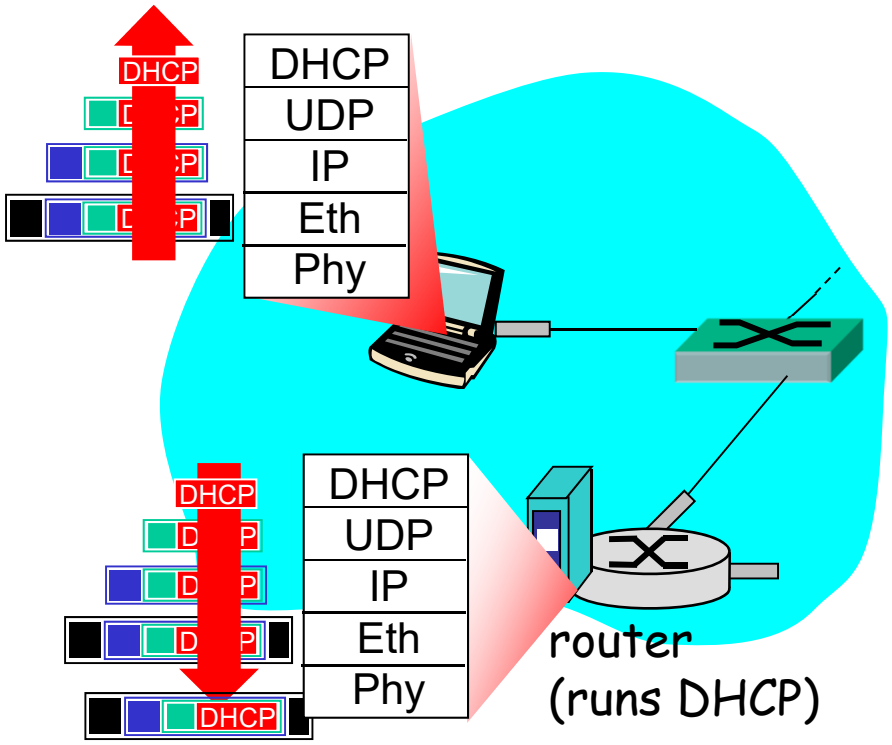
A Day in the Life… Connecting to the Internet



- Connecting laptop needs to get its own IP address, addr of 1st-hop router, addr of DNS server: use *DHCP*

- DHCP packets encapsulated in UDP, encapsulated in IP with broadcast address 255.255.255.255, encapsulated in Ethernet
- Ethernet frame broadcast (dest: FF:FF:FF:FF:FF:FF) on LAN, received at router running DHCP server
- Ethernet demux'ed to IP demux'ed to UDP demux'ed to DHCP

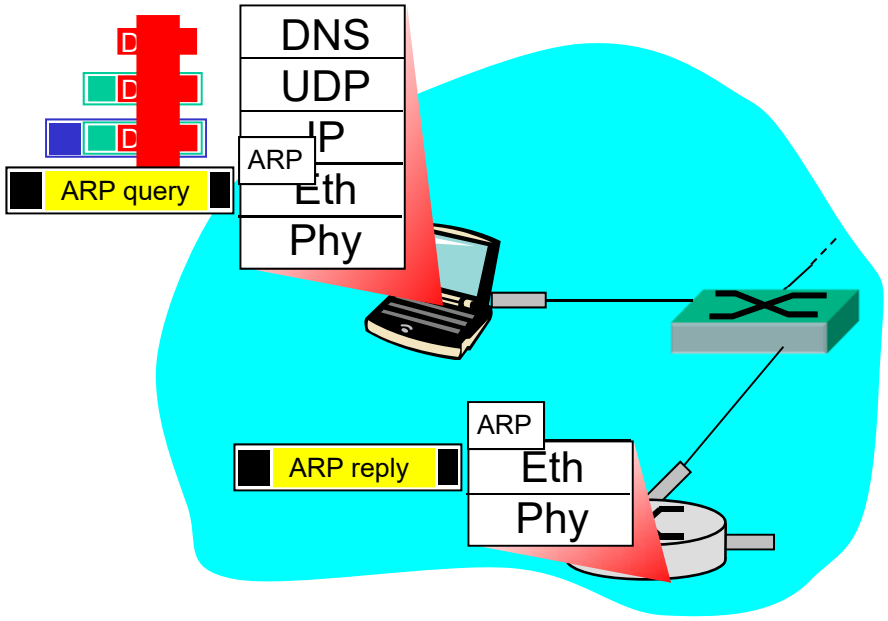
A Day in the Life… Connecting to the Internet



- After DHCP Offer and Request, DHCP server formulates *DHCP ACK* containing client's IP address, IP address of 1st-hop router for client, name & IP address of DNS server
- Encapsulation at DHCP server, frame forwarded (switch learning) through LAN, demultiplexing at client
- DHCP client receives DHCP ACK reply

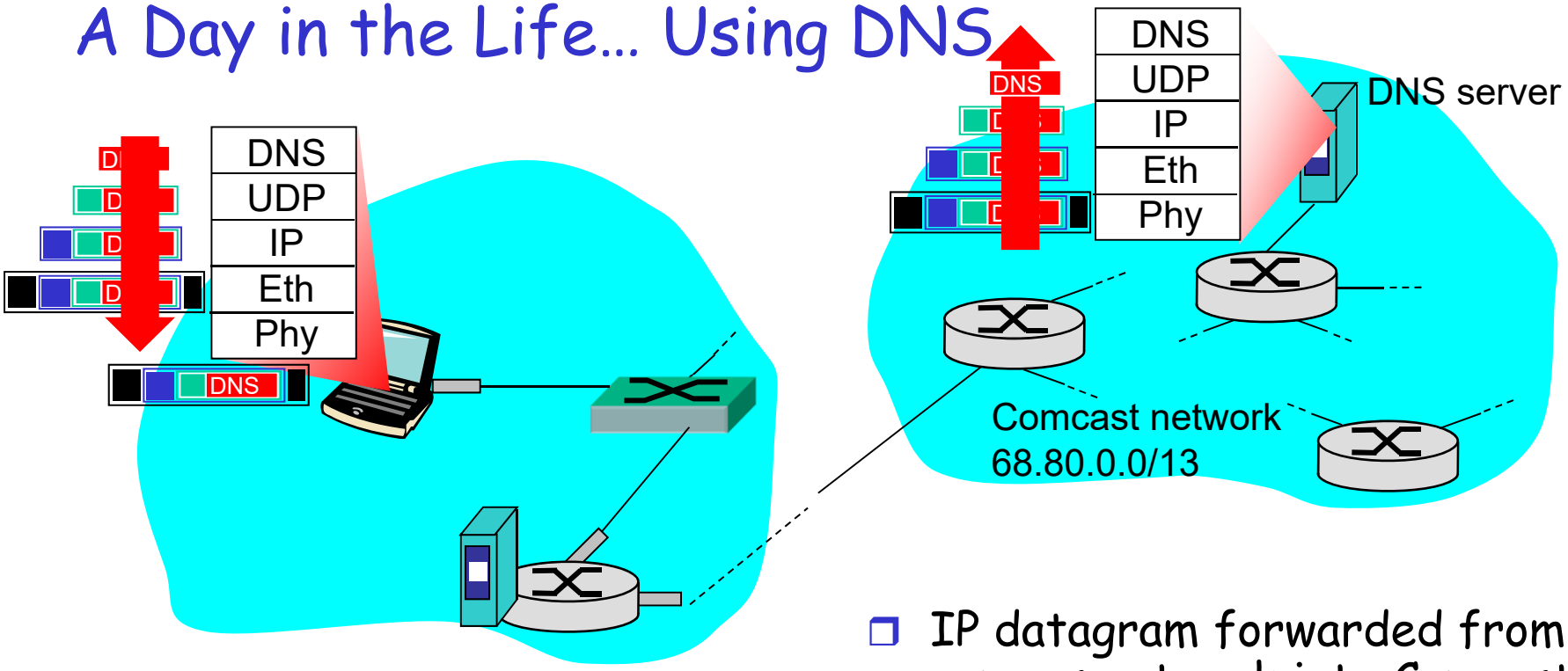
Client now has IP address, knows name & addr of DNS server, IP address of its 1st-hop router

A Day in the Life… ARP (Before DNS, Before HTTP)



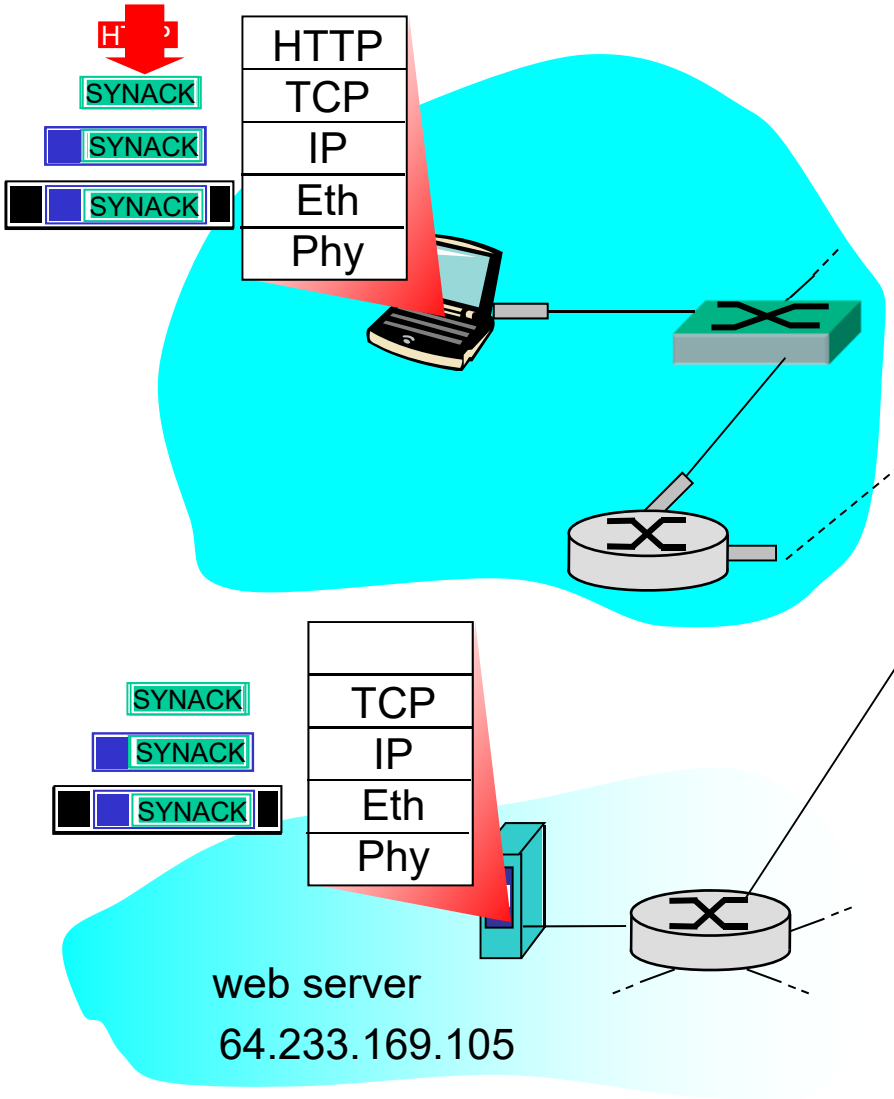
- Before sending *HTTP* request, need IP address of www.google.com: *DNS*
- DNS query created, encapsulated in UDP / IP / Eth. In order to send frame to router, need MAC address of router interface: *ARP*
- *ARP query* broadcast, received by router, which replies unicast with *ARP reply* giving MAC address of router interface
- Client now knows MAC address of 1st-hop router, so can now send frame containing DNS query

A Day in the Life… Using DNS



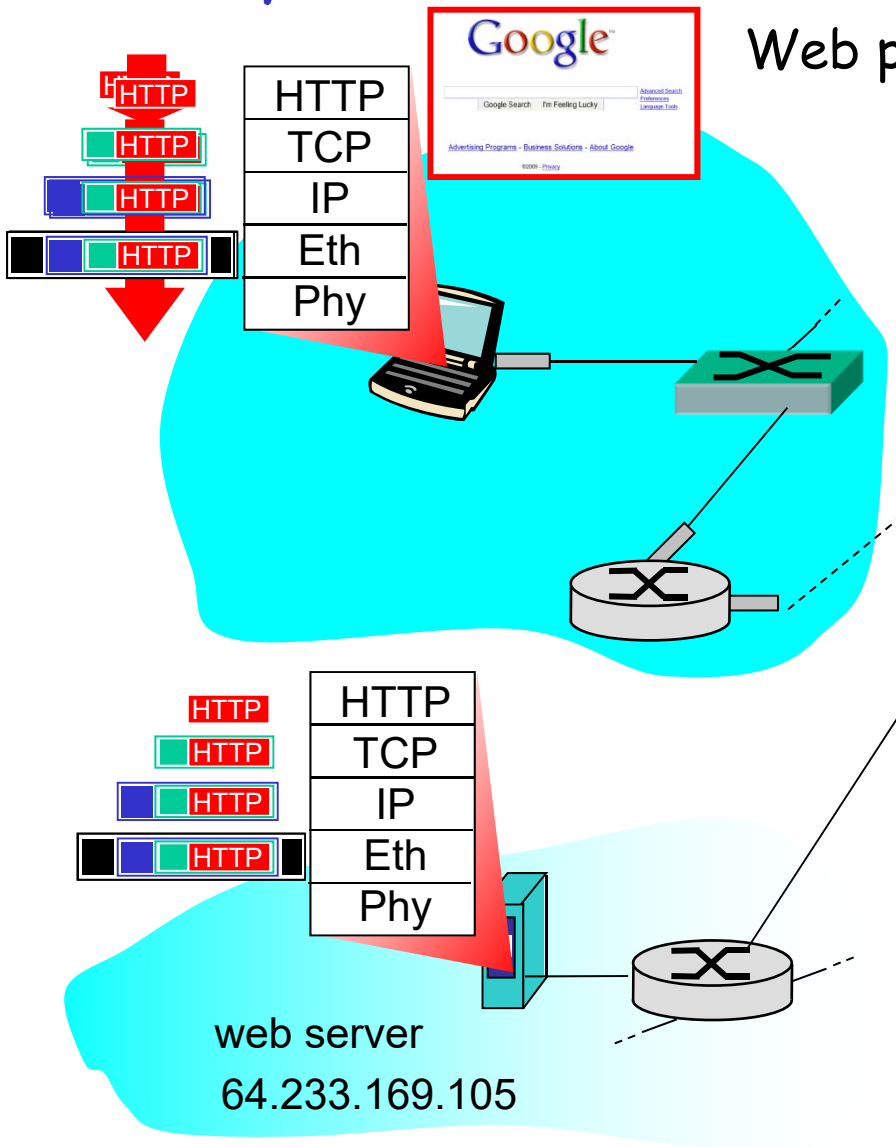
- IP datagram containing DNS query forwarded via LAN switch from client to 1st-hop router
- *Note* This does not show the root and TLD DNS servers
- IP datagram forwarded from campus network into Comcast network, routed (tables created by RIP, OSPF and/or BGP routing protocols) to DNS server
- Demux'ed at DNS server
- DNS server replies to client with IP address of www.google.com

A Day in the Life… TCP Connection Carrying HTTP



- To send HTTP request, client first opens TCP socket to web server
- TCP SYN segment (step 1 in 3-way handshake) inter-domain routed to web server
- Web server responds with TCP SYN-ACK (step 2 in 3-way handshake)
- TCP connection established!

A Day in the Life… HTTP Request/Reply



Web page finally (!!!) displayed

- HTTP request (GET) sent into TCP socket piggyback on the final ACK in 3-way handshake
- IP datagram containing HTTP request routed to www.google.com
- Web server responds with HTTP reply (containing web page)
- IP datgram containing HTTP reply routed back to client