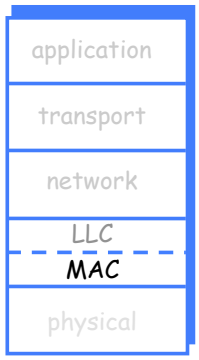
Data Link Layer

The Data Link layer can be further subdivided into:

1. Logical Link Control (LLC): error and flow control
2. Media Access Control (MAC): framing and media access

different link protocols may provide different services, e.g., Ethernet doesn't provide reliable delivery (error recovery)



MAC topics:

- framing and MAC address assignment
- LAN forwarding
- IP to MAC address resolution
  - IP to MAC: Address Resolution Protocol (ARP)
  - MAC to IP: Reverse ARP (RARP), BOOTstrap Protocol (BOOTP), Dynamic Host Configuration Protocol (DHCP)
- media access control

Multiple Access Problem

Broadcast channel of rate R bps, shared medium

- if two users send at the same time, *collision* results in no packet being received (interference)
- if no users send, channel goes idle
- thus, want to have only one user send at a time

Media Access Control:

- determines who gets to send next
- what to do if more than one hosts send at the same time and there's collision

Duplex mode:

- half duplex: only one end can send at a time
- full duplex: both ends can send simultaneously

Ideal Multiple Access Protocol

- when one node wants to transmit, it can send at rate R
- when M nodes want to transmit, each can send at average rate R/M
- fully decentralized:
  - no special node to coordinate transmissions
  - no synchronization of clocks, slots
  - distributed algorithm that determines how nodes share channel, i.e., determine when node can transmit
  - communication about channel sharing must use channel itself!
    - no out-of-band channel for coordination

Categorization of MAC Protocols

1. Random access:
   - Slotted ALOHA
   - ALOHA
   - Carrier Sense Multiple Access with Collision Detection (CSMA/CD)
   - CSMA/CAvoidance
2. Token passing
3. Channel partitioning: TDMA, FDMA, CDMA

Standards:

- 802.3 (CSMA/CD), 802.3a? (GigE)
- 802.4 (token bus)
- 802.5 (token ring)
- 802.11[bagn] (WiFi)

Random Access MAC Protocol

Characteristics:

- sender xmits bits on the wire at full channel rate R bps
  - no prior coordination among nodes
- bits are propagated along the entire network
- destination recognizes that frame is for itself
- destination grabs frame
- while one host is xmitting, all others must wait

Random access means:

- relies on collision to control access
- how to detect collisions
- how to recover from collisions

Ethernet: CSMA/CD

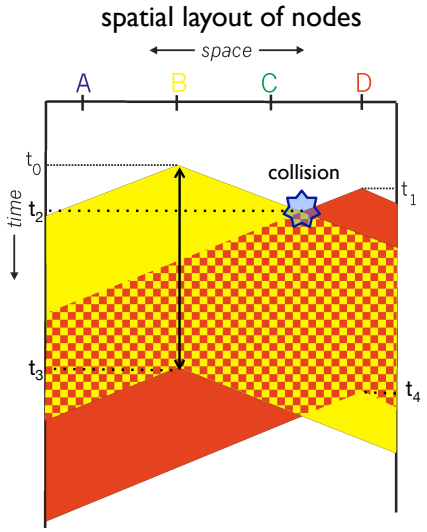
Carrier Sensing:

1. check for presence of electrical signal (carrier) on wire before transmission
2. presence of carrier means someone else is sending, wait
3. start transmission if no carrier detected

Problem: collision

CSMA Collisions

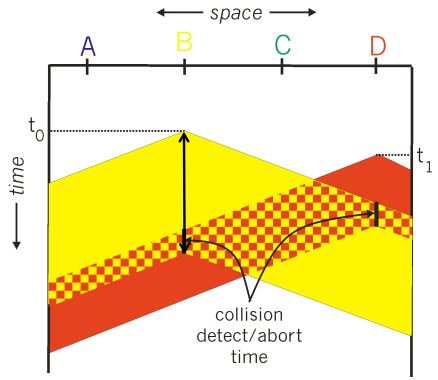
- collisions occurs because propagation delay means two nodes may not hear of each other's transmission when they start transmitting (*A* at t_0, *D* at t_1)
- when collision occurs (at t_2), entire frame transmission time (t_3-t_0 or, equivalently, t_4-t_1) is wasted
- note the role distance & propagation delay play in determining collision probability
- a collision is detected if power received is larger than power transmitted



Collision Detection

- sender must continue to detect collision after transmission
- on collision, frames must be retransmitted
- problem: more collision

4. if adaptor detects collision while transmitting, aborts and sends **jam signal**
5. after aborting, adaptor enters **exponential backoff**



Jam Signal and Exponential Back-off

Jam signal: make sure all other transmitters are aware of collision; 48 bits

Exponential back-off: senders pick a uniformly distributed random delay between $[0,2^0 d]$ before retransmission. Why random?

If collision occurs again, pick another random delay between $[0,2^1 d]$, $[0,2^2 d]$, $[0,2^3 d]$, . . . hence (*binary*) *exponential back-off*

Bit time: .1 μsec on a 10 Mbps Ethernet

➞ for $2^{10} d$, wait time is about 50d msec

CSMA/CD Summary

The algorithm:

1. listen for carrier
2. if no carrier, send frame
3. listen for collision or jamming signal
4. if collision detected, send jamming signal
5. if collision or jamming signal detected, retransmit after exponential back-off

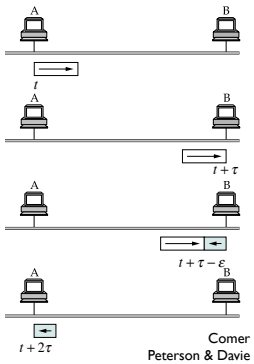
Historical Note:

Collision detection and retransmission with back-off was first used in the ALOHA MAC algorithm from the University of Hawaii (1970) for access to satellite channels

Collision Detection Time

How long must a sender listen for collision?

- let τ be the propagation time from one end of the wire to the other
- within τ time after the transmission of a frame (t), all nodes on the segment would have sensed carrier
- worst case scenario for collision: a node at the other end of the wire starts transmitting at time $t + \tau - \varepsilon$
- the node closest to the collision sends out a jamming signal to ensure collision is detected by the other node



Comer
Peterson & Davie

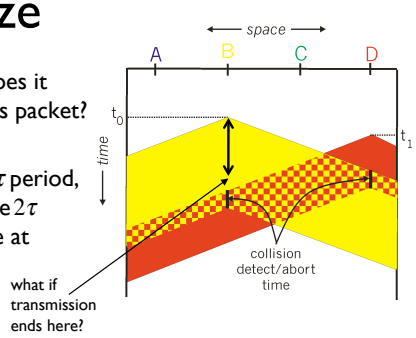
- it takes another τ period for the collision to get back to the original sender

Hence the original sender must listen for 2τ period

Minimum Frame Size

When a sender detects collision how does it know that the collision was caused by its packet?

Answer: sender must hold carrier for 2τ period, i.e., it must be transmitting for the whole 2τ period ➞ each Ethernet frame must be at least 2τ*linkspeed long



Example:

- 10 Mbps Ethernet allows maximum of 5 segments, each 500 m long
- speed of light 3×10^8m/s, but coax propagation 2×10^8m/s
- round-trip propagation delay (2τ) on 2.5 km coax is 25 μsecs
- allowing for 4 repeaters makes end-to-end delay 50 μsecs
- 50 μsecs means 62.5 bytes
- 802.3 standard requires stations to hold carrier for 64 bytes/10 Mbps = 51.2 μsecs

CSMA/CD Efficiency (η)

t_{prop} = max propagation time between 2 nodes in the LAN
t_{trans} = time to transmit maximum-size frame

$$\eta = \frac{t_{trans}}{t_{trans} + 5t_{prop}} = \frac{1}{1 + 5t_{prop} / t_{trans}}$$

$$\eta \to 1 \text{ as } t_{prop} \to 0 \text{ or as } t_{trans} \to \infty$$

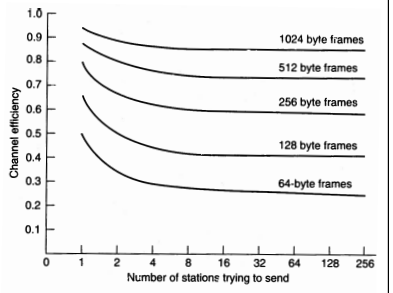


Fig. 4-23. Efficiency of 802.3 at 10 Mbps with 512-bit min frame size

Tanenbaum

Repeaters and Bridges

Each Ethernet segment is limited to 500 m long by signal attenuation

Repeaters: repeat and strengthen signal (physical layer)

Ethernet only allows 4 repeaters: max 2.5 km. Why?

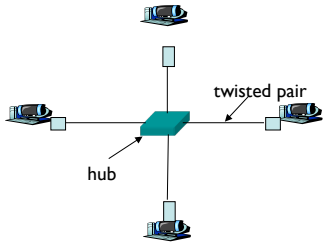
Bridges: equivalence of routers at the data link layer

- forward frames between segments
- unlike routers, only know whether a node is in a segment
- does not propagate interference and collisions (must buffer)
- increase effective/aggregate bandwidth of a LAN by taking advantage of spatial locality
- can connect segments with different MAC protocols

Hubs

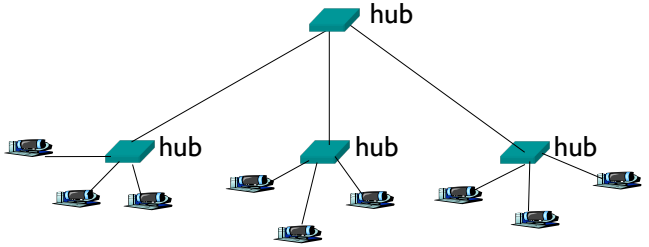
Hubs are essentially physical-layer repeaters:

- bits coming from one link go out all other links
- at the same rate
- no frame buffering
- no CSMA/CD at hub: collision detection left to host adaptors



Interconnecting with Hubs

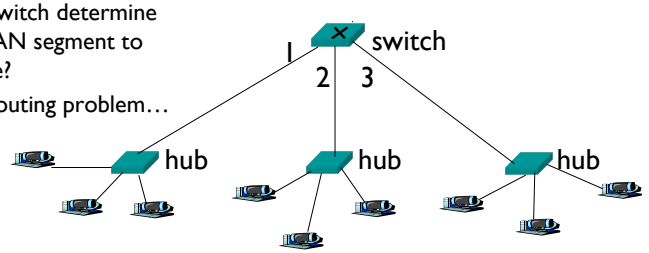
- backbone hub interconnects LAN segments
- extends max distance between nodes
- but individual segment collision domains become one large collision domain
- can't interconnect 10BaseT & 100BaseT



Switches

Link layer router-equivalent:

- stores and forwards Ethernet frames
- examines frame header and selectively forwards frame based on MAC destination address
- when frame is to be forwarded on a segment, uses CSMA/CD to access segment
- transparent: hosts are unaware of presence of switches
- plug-and-play: self-learning, switches do not need to be configured
- How does a switch determine onto which LAN segment to forward frame?
- Looks like a routing problem…



Transparent Bridges/Switches and Backward Learning

How does a bridge know which segment a node is located at?

Each switch has a switch table, entry in switch table:

- <MAC Address, interface, timestamp>
- stale entries in table dropped (TTL can be 60 min)

switch *learns* which hosts can be reached through which interfaces

- when a frame is received, switch "learns" location of sender: incoming interface connects to the LAN segment through which a sender may be reached
- records sender/interface pair in switch table
- called "backward learning"

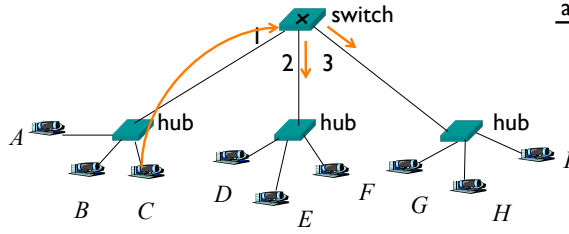
Frame Filtering/Forwarding

When a switch receives a frame:

look for MAC destination address in switch table
if entry found for destination {
  if destination on segment from which frame arrived {
    drop the frame
  } else {
    forward the frame on interface indicated
  }
} else {
  flood // forward on all but the interface on which the frame arrived
}

Switch Example

Suppose C sends a frame to D

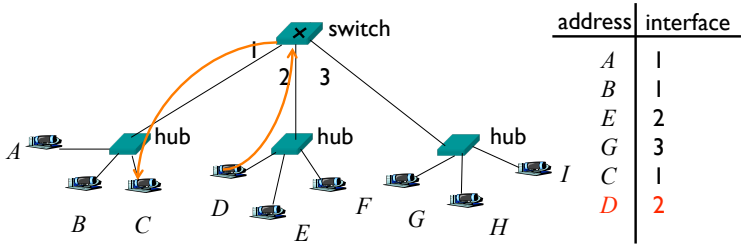


address	interface
A	1
B	1
E	2
G	3
C	1

Switch receives frame from C
  records 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 now sends a frame to C



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

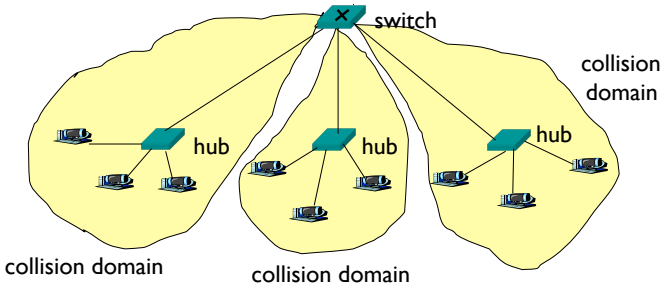
Switch receives frame from D
- records in switch table that D is on interface 2
- because C is in table, switch forwards frame only to interface 1

frame received by C

Switch: Traffic Isolation

switch installation breaks subnet into LAN segments

switch filters packets:
- same-LAN-segment frames are not usually forwarded onto other LAN segments
- segments become separate collision domains

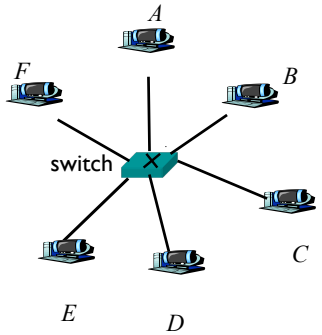


Switches: Dedicated Access

Hosts have direct connection to switch

No collisions; full duplex

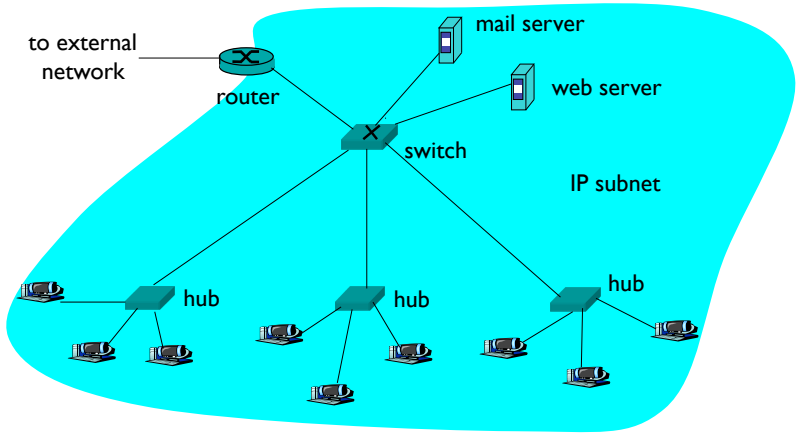
Switching: A-to-D and B-to-E simultaneously, no collisions



Cut-through switching: frame forwarded from input to output port without storing
- slight reduction in latency

switches can support combinations of shared/dedicated and 10/100/1000 Mbps interfaces

Example Enterprise Network Switch/Hub Installment

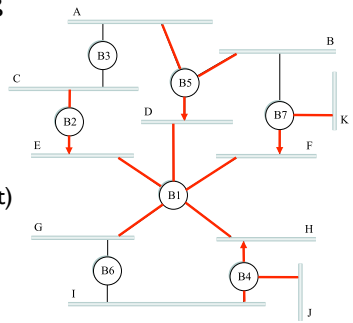


Switches and Spanning Tree

LANs may form cycles, causing broadcast storm

Bridges/switches detect cycles by doing distributed spanning tree computation:

- all bridges broadcast serial #, root ID, cost to root
- bridge with lowest serial # becomes root of tree
- all bridges determine root port (port to root)
- the spanning tree consists of bridges (nodes) and root port (links)



Peterson & Davie

Forwarding on the tree:

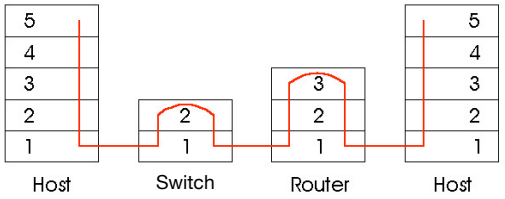
- each LAN determines a designated bridge by lowest cost to root, break tie by serial #
- forward frames only on links that are part of the tree

Switches vs. Routers

Both store-and-forward devices

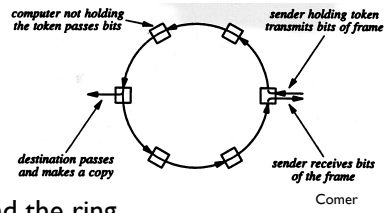
Given bridges, why do we still need routers?

- routers are network layer devices (what does this mean?)
  - routers maintain routing tables, implement routing algorithms
- switches are link layer devices
  - switches maintain switch tables, implement filtering, backward learning algorithms



Token Ring MAC Protocol

- a **token** goes around a ring network
- to send data, a node must first grab the token
- a frame sent from a source is passed from node to node around the ring



Comer

- destination recognizes own address and makes a copy of frame
- sender removes frame from ring
- each node can only transmit one frame at a time; must return token to the ring after each frame transmission

Why let the sender, instead of the receiver, remove frame from the ring?

Token Ring MAC Protocol

Token:

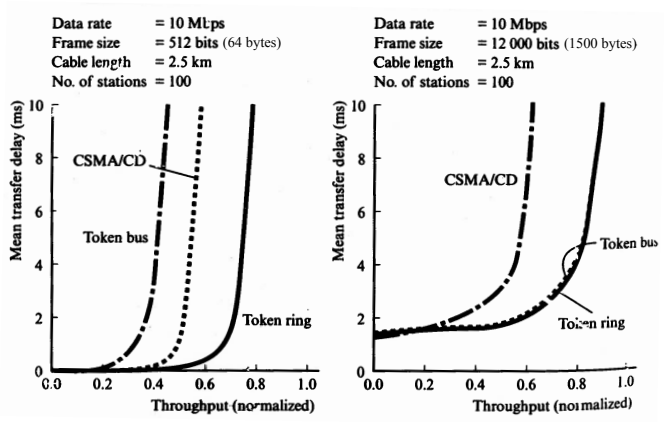
- a special bit pattern
- use bit-stuffing if data resembles token
- only one token on ring at a time (managed by a monitor)

IBM's token ring link speed is 16 Mbps

Token ring:

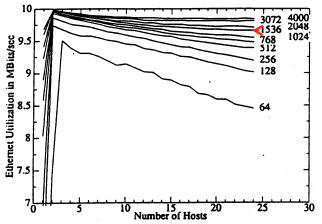
- advantage: no collision
- disadvantage: failure of a node or link disables the whole network

Token Ring Performance



Halsall

CSMA/CD Efficiency (η)



Measured Capacity of an Ethernet: Myths and Reality

David R. Boggs
Jeffrey C. Mogul
Christopher A. Kent

Figure 3-3: Total bit rate

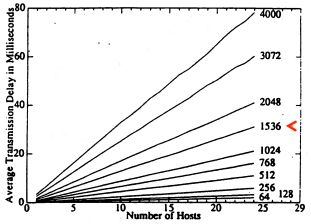
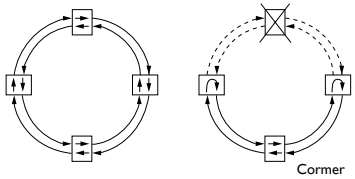


Figure 3-7: Average transmission delay

Other MAC Protocols



Cormer

FDDI:

- operates at 100 Mbps
- uses the token ring MAC protocol
- for robustness, uses two counter-rotating rings
- if a link/node goes down, the dual-ring can be reconfigured to a single ring network (hence called **self-healing** network)

SLIP/PPP: serial line, point-to-point protocol, no need for media access control, just framing

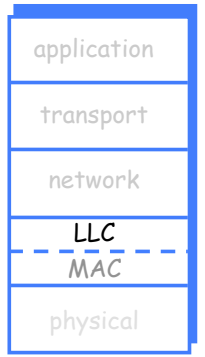
ATM/Frame Relay/SONET: for backbone links . . . .

Data Link Layer

The Data Link layer can be further subdivided into:

1. Logical Link Control (LLC): error and flow control
2. Media Access Control (MAC): framing and media access

different link protocols may provide different services, e.g., Ethernet doesn't provide reliable delivery (error recovery)



MAC topics:

- framing and MAC address assignment
- LAN forwarding
- IP to MAC address resolution
  - IP to MAC: Address Resolution Protocol (ARP)
  - MAC to IP: Reverse ARP (RARP), BOOTstrap Protocol (BOOTP), Dynamic Host Configuration Protocol (DHCP)
- media access control

Ethernet: Connectionless Service

No handshaking between sending and receiving adaptor

Receiving adaptor doesn't send ACKs or NACKs to sending adaptor

- stream of datagrams passed up to network layer can have gaps
- gaps will be filled if application uses reliable transport layer
- otherwise, application will see the gaps

Other data link protocols may provide error correction and flow control

Transmission Errors

Three kinds of transmission errors:

1. sent signal changed (received wrong data)
2. sent signal destroyed (doesn't receive data)
3. spurious signal created (received random data)

Caused by noise on the channel:
interference, cosmic rays

Error Control

Ways to detect errors, general idea:

- sender computes some info from data
- sender sends this info along with data
- receiver does the same computation and compares it with the sent info

Not often used for largely reliable links,
but useful for unreliable links such as wireless

Used at the transport layer also
(the Internet is an unreliable "link")

Field: Information Theory

Error Control

Two types of error control:

1. error detecting code
2. error correcting code (ECC),
   a.k.a. forward error correction/control (FEC)

Examples error detecting code:

- parity check
- checksum
- cyclic redundancy check (CRC)

Error Control

Trade-offs between alternate methods:

- complexity of info computation,
- bandwidth transmission overhead, and
- degree of protection (# of bit errors that can be detected)

No error detection method is fool-proof

Parity Check

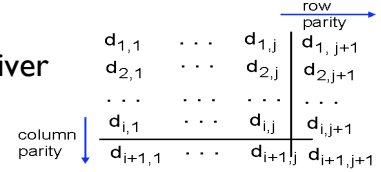
- uses an extra bit (**parity bit**) for error checking
- even parity: total # of 1 bits (incl. the parity bit) is an even number
- odd parity: total # of 1 bits is odd
- single-bit parity examples:

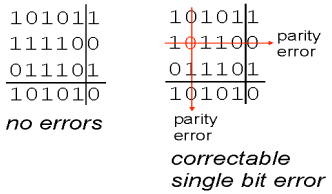
  0100101, even-parity bit =

  0101101, even-parity bit =
- what happens when an error is detected?
  - discard data and if reliability is required, have sender retransmit
- problem: can not detect even # of flipped bits

2D Parity Check as ECC

- generates both a horizontal/row parity and a vertical/column parity
- both parity info sent to receiver
- receiver can detect *and correct* single-bit errors
- problem: can not detect even # of flipped bits





no errors

correctable single bit error

Error Correction vs. Detection

ECC generally requires more redundant bits than just detection

It is generally cheaper to retransmit data only when error has been detected than to transmit redundant data *all the time*

FEC is most useful when:

1. link is very noisy, e.g., wireless link
2. retransmission will take too long, e.g.,
   - satellite communication
   - deep space probe transmission
   - real-time audio/video streaming

Checksum

Used also by TCP and UDP

Sender treats data as a sequence of integers and computes their (1's complement) sum

Example: 16-bit checksum
- the string "Hello world." has an ASCII representation of [48 65 6C 6C 6F 20 77 6F 72 6C 64 2E]
- checksum: 4865 + 6C6C + 6f20 + 776F + 726C + 642E + carry = 71FC

Advantages:
- ease of computation (only requires addition)
- small amount of additional info to carry: one additional 16-bit or 32-bit integer

Checksum

Disadvantage:
- with 16-bit checksum, 1 in 64K corrupted packet will not be detected (probability of a random 16-bit number matching the checksum of a corrupted packet is $1/2^{16}$)

Data Item In Binary	Checksum Value	Data Item In Binary	Checksum Value
00001	1	00011	3
00010	2	00000	0
00011	3	00001	1
00001	1	00011	3
totals	7		7

⇒ under current Internet conditions (error rate etc.), 1 in every 300M packet accepted corrupted!

Mogul (1992) measured on a busy NFS server that has been up 40 days:

Layer	# checksum errors caught	~#pkts
ethernet	(CRC) 446	1.7×10^8
IP	14	1.7×10^8
UDP	5	1.4×10^8
TCP	350	3×10^7

Cyclic Redundancy Check

Goal of any error detection/correction code: maximize probability of detecting error with minimal redundant info

32-bit CRC protects against most bit errors in messages thousands of bytes long, also used in storage systems (CD, DVD)

CRC is based on finite fields math

Consider a binary message as a representation of an n-degree polynomial, with the coefficient of each term being 1 or 0 depending on the bit in the message, with the most significant (leftmost) bit representing the highest degree term

- For example: 1011 represents $1x^3 + 0x^2 + 1x^1 + 1x^0 = x^3 + x + 1$

An m-bit message represents a polynomial of $m-1$ degree

Polynomial Arithmetic

The math says you can divide one such polynomial by another such polynomial of lower or equal degree by dividing the binary representation of the polynomials, e.g., to divide $x^5+x^3+x^2+x$ by x^3+1, divide 101110 by 1001

Polynomial arithmetic is done using modulo-2 arithmetic, with no carry and borrow: 1+1 = 0+0 = 0 and 1+0 = 0+1 = 1, e.g.,

```
10011011        11110000        01010101
11001010 +      10100110 -      10101111 -
--------        --------        --------
01010001        01010110        11111010
```

Note that both addition and subtraction are identical to XOR

Constructing CRC

Let's call the polynomial to be divided T and the divisor/ generator polynomial G

Let t be the number of bits in T and $r+1$ be the number of bits in G, $t \geq r+1$

Let's call the remainder of T/G, R; R is of r bits

Want: the polynomial represented by the message to be exactly divisible by G, such that if the receiver divides the message by G and the remainder is not 0, it will know the message has been corrupted

$M = (T\text{-}R)$ is exactly divisible by G

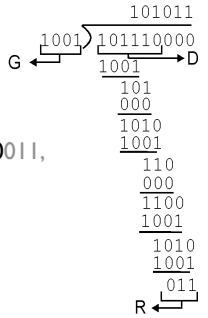
Constructing CRC

Recall: multiplying a number by 2 is the same as shifting it left by 1 bit

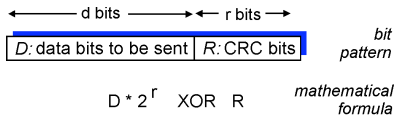
Let D be the message to be sent, e.g., D =101110

Construct T as $D \cdot 2^r$, D shifted left by r bits, e.g., r =3, T =101110000

Let G = 1001, compute R, the remainder of T/G, by doing the long-division, with modulo-2 arithmetic, e.g., R = 011



Now $M = (T\text{-}R) = (D \cdot 2^r - R) = (D \cdot 2^r \text{ XOR } R)$, e.g., 101110011, is exactly divisible by G



How to Choose G?

Let the string of bit errors introduced be represented as polynomial E

Error will not be detected only if $T+E$ is exactly divisible by G

Want G that makes this unlikely. What's known:

- if x^r and x^0 terms have non-zero coefficients, G can detect all single-bit errors
- as long as G has a factor with at least 3 terms, it can detect all double-bit errors
- as long as G contains the factor $(x+1)$, it can detect any odd number of errors
- G can detect any burst (sequence of consecutive) errors of length < r bits

Usually, you just look up a commonly used G, e.g., Ethernet uses CRC-32

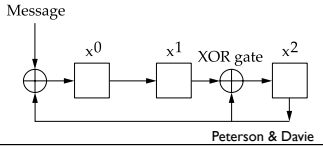
CRC-32: 10000010011000001000111101101110111
CRC-CCITT: 1000100000010000 1

CRC Hardware Implementation

CRC can be cheaply implemented in hardware by implementing the long-division to compute R as a combination of linear feedback shift register (LFSR) and XOR gates

The shift registers and XOR gates represents the G:

- the 0-th term of G occupies the leftmost bit of the shift registers
- each XOR gate represents a modulo-2 addition in G
- the message is fed into the circuit most significant (leftmost) bit first
- each bit of the message causes the current content of the shift registers to be shifted right by one bit
- when the message is exhausted, the shift registers contain R
- for example, computing CRC with $G = x^2+1$ can be implemented as:



Link Layer Services

Half-duplex and full-duplex

- with half duplex, nodes at both ends of link can transmit, but not at same time

Framing, link access:

- encapsulate datagram into frame, adding header, trailer
- channel access if shared medium
- "MAC" addresses used in frame headers to identify source, dest
  - different from IP address!

Error Detection:

- errors caused by signal attenuation, noise.
- receiver detects presence of errors:
  - signals sender for retransmission or drops frame

Link Layer Services

Error Correction:

- receiver identifies and corrects bit error(s) without resorting to retransmission

Flow Control:

- pacing between adjacent sending and receiving nodes

Reliable delivery between adjacent nodes

- seldom used on low bit error link (fiber, some twisted pair)
- wireless links: high error rates
- Q: why both link-level and end-end reliability?

Midterm Review: Mon, 2/16 in class

Midterm Exam: Wed, 2/18, 6:40-8:40 pm in 1670 CSE (next door)

Spring Break: week of 2/23