Data Communications & Networks

Session 4 – Main Theme
Data Link Control

Dr. Jean-Claude Franchitti

New York University
Computer Science Department
Courant Institute of Mathematical Sciences

Agenda

1. **Session Overview**
2. **Data Link Control**
3. **Summary and Conclusion**

What is the class about?

- Course description and syllabus:
  - » http://www.nyu.edu/classes/jcf/g22.2262-001/
  - » http://www.cs.nyu.edu/courses/spring10/G22.2262-001/index.html

- Textbooks:
  - » ***Computer Networking: A Top-Down Approach (5th Edition)***

James F. Kurose, Keith W. Ross
Addison Wesley
ISBN-10: 0136079679, ISBN-13: 978-0136079675, 5th Edition (03/09)

Course Overview

- Computer Networks and the Internet
- Application Layer
- Fundamental Data Structures: queues, ring buffers, finite state machines
- Data Encoding and Transmission
- Local Area Networks and Data Link Control
- Wireless Communications
- Packet Switching
- OSI and Internet Protocol Architecture
- Congestion Control and Flow Control Methods
- Internet Protocols (IP, ARP, UDP, TCP)
- Network (packet) Routing Algorithms (OSPF, Distance Vector)
- IP Multicast
- Sockets

Data Link Control Session in Brief

- Principles Behind Data Link Layer Services:
  - Error Detection and Correction
  - Sharing a Broadcast Channel Multiple Access
  - Link-Layer Addressing
  - Reliable Data Transfer and Flow Control
- Instantiation and implementation of various link layer technologies:
  - Ethernet
  - Link-layer switches
  - PPP
  - Link virtualization: MPLS
  - A day in the life of a web request

Icons / Metaphors

- Information
- Common Realization
- Knowledge/Competency Pattern
- Governance
- Alignment
- Solution Approach

Agenda

1 Session Overview

2 Data Link Control

3 Summary and Conclusion

Data Link Control - Roadmap

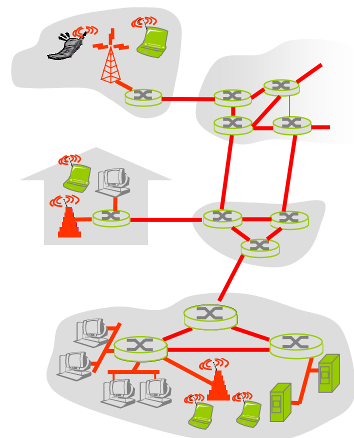
2 Data Link Control

- Introduction and Services
- Error Detection and Correction
- Multiple Access Protocols
- Link Layer Addressing
- Ethernet
- Additional Topics

Link Layer: Introduction

<u>Some terminology:</u>

- hosts and routers are **nodes**
- 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 has responsibility of transferring datagram from one node to adjacent node over a link

Link layer: context

- datagram transferred by different link protocols over different links:
  - e.g., Ethernet on first link, frame relay on intermediate links, 802.11 on last link
- each link protocol provides different services
  - e.g., may or may not provide rdt over link

<u>transportation analogy</u>

- trip from Princeton to Lausanne
  - limo: Princeton to JFK
  - plane: JFK to Geneva
  - train: Geneva to Lausanne
- tourist = datagram
- transport segment = communication link
- transportation mode = link layer protocol
- travel agent = routing algorithm

Link Layer Services (1/2)

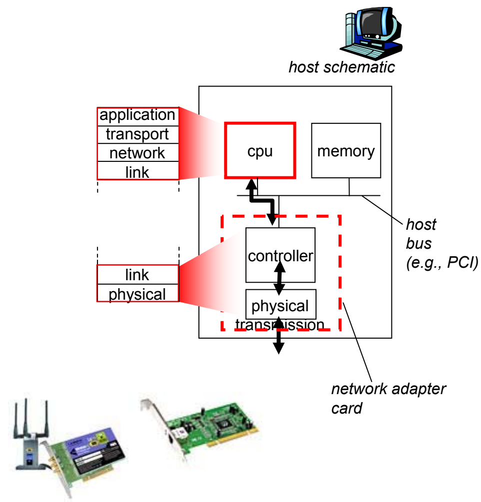
- Framing, link access
  - Encapsulate datagram into frame, adding header, trailer
  - Channel access if shared medium
  - "MAC" addresses used in frame headers to identify source, dest
    - Different from IP address!
- Reliable delivery between adjacent nodes
  - We briefly discussed how to do this already!
  - 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?

Link Layer Services (2/2)

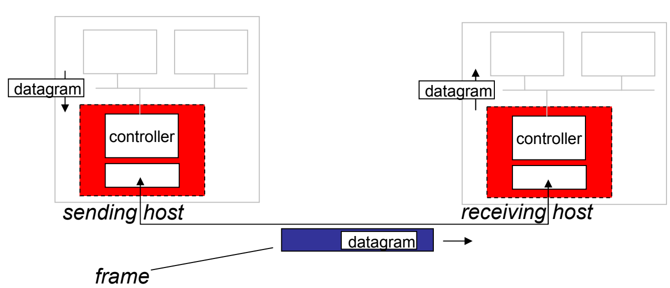
- Flow Control
  - Pacing between adjacent sending and receiving nodes
- Error Detection
  - Errors caused by signal attenuation, noise.
  - Receiver detects presence of errors
    - Signals sender for retransmission or drops frame
- Error Correction
  - Receiver identifies *and corrects* bit error(s)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

Where is the link layer implemented?

- in each and every host
- link layer implemented in "adaptor" (aka *network interface card* NIC)
  - » Ethernet card, PCMCI card, 802.11 card
  - » implements link, physical layer
- attaches into host's system buses
- combination of hardware, software, firmware



Adaptors Communicating



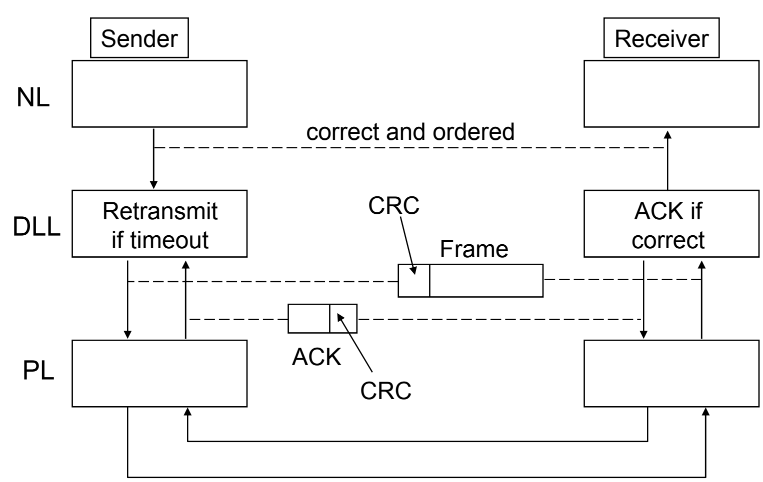
- sending side:
  - » encapsulates datagram in frame
  - » adds error checking bits, rdt, flow control, etc.
- receiving side
  - » looks for errors, rdt, flow control, etc
  - » extracts datagram, passes to upper layer at receiving side

Encoding Information Frames

Typical Fields in a Frame

Start Frame Delimiter	Destination Address	Source Address	Frame Control	Data	Check sum

DLL Operation



Data Link Control - Roadmap

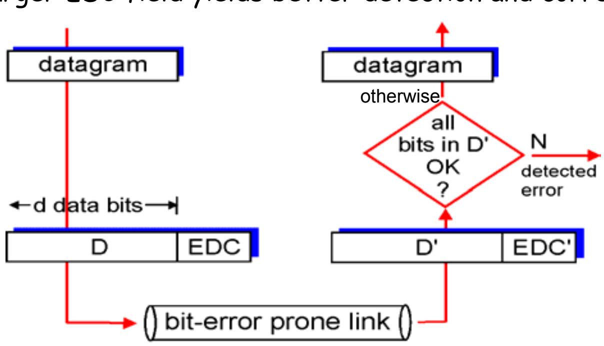
2 Data Link Control

- Introduction and Services
- Error Detection and Correction
- Multiple Access Protocols
- Link Layer Addressing
- Ethernet
- Additional Topics

Error Detection

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, but rarely
  - larger EDC field yields better detection and correction



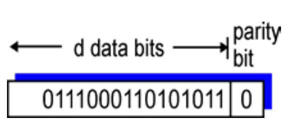
Error Detection

- Additional bits added by transmitter for error detection code
  - Parity
    - Value of parity bit is such that character has even (even parity) or odd (odd parity) number of ones
    - Even number of bit errors goes undetected
    - Checksum can be a simple XOR operation of bits to be checked
- DL protocols use more sophisticated methods, like Cyclic Redundancy Check (CRC)

Parity Checking

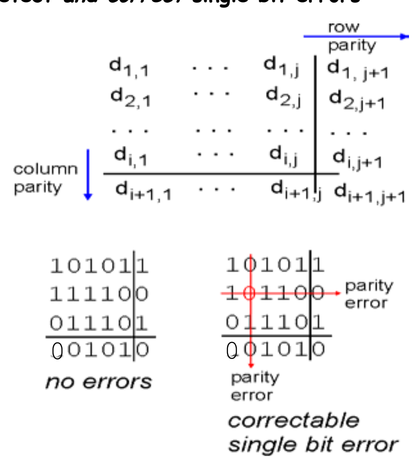
Single Bit Parity:

Detect single bit errors



Two Dimensional Bit Parity:

Detect *and correct* single bit errors

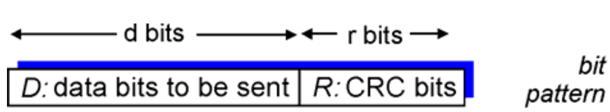


Internet Checksum

- Goal: detect "errors" (e.g., flipped bits) in transmitted segment (note: used at transport layer *only*)
- Sender:
  - Treat segment contents as sequence of 16-bit integers
  - Checksum: addition (1's complement sum) of segment contents
  - Sender puts checksum value into UDP checksum field
- Receiver:
  - Compute checksum of received segment
  - Check if computed checksum equals checksum field value:
    - NO - error detected
    - YES - no error detected. *But maybe errors nonetheless?* More later ….

Checksuming: Cyclic Redundancy Check

- View data bits, D, as a binary number
- Choose r+1 bit pattern (generator), G
- Goal: choose r CRC bits, R, such that
  - <D,R> exactly divisible by G (modulo 2)
  - Receiver knows G, divides <D,R> by G. If non-zero remainder: error detected!
  - Can detect all burst errors less than r+1 bits
- Widely used in practice (ATM, HDLC)

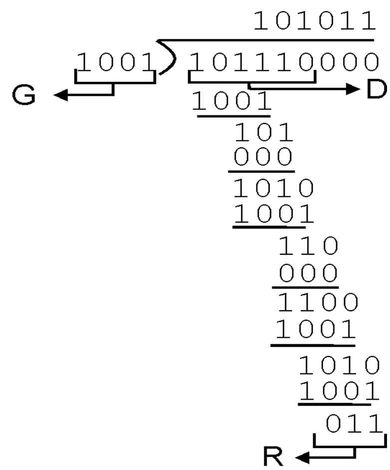


$D * 2^r$ XOR R *mathematical formula*

CRC Example

- Want
  - $D.2^r$ XOR R = nG
- Equivalently
  - $D.2^r$ = nG XOR R
- Equivalently:
  if we divide $D.2^r$ by G,
  want remainder R

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



Example of G(x) Polynomials

- CRC-12
  - $X^{12} + X^{11} + X^3 + X^2 + X + 1$
- CRC-16
  - $X^{16} + X^{15} + X^2 + 1$
- CRC-CCITT
  - $X^{16} + X^{15} + X^5 + 1$
- CRC-32
  - $X^{32} + X^{26} + X^{23} + X^{22} + X^{16} + X^{12} + X^{11} + X^{10}$
    $+ X^8 + X^7 + X^5 + X^4 + X^2 + X + 1$
- CRC's Are Implemented in Shift registers

Data Link Control - Roadmap

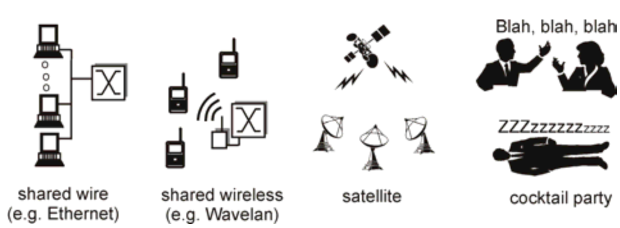
2 Data Link Control

- Introduction and Services
- Error Detection and Correction
- Multiple Access Protocols
- Link Layer Addressing
- Ethernet
- Additional Topics

Multiple Access Links and Protocols (1/2)

Two types of “links”

- Point-to-point
  - PPP for dial-up access
  - Point-to-point link between Ethernet switch and host
- Broadcast (shared wire or medium)
  - Old-fashioned Ethernet
  - Upstream HFC
  - 802.11 wireless LAN



Multiple Access Protocols (2/2)

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

<u>Multiple access protocol</u>

- Distributed algorithm that determines how nodes share channel, i.e., determine when node can transmit
- Communication about channel sharing must use channel itself!

Ideal Multiple Access Protocols

<u>Broadcast channel of rate R bps</u>

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

MAC Protocols: A Taxonomy

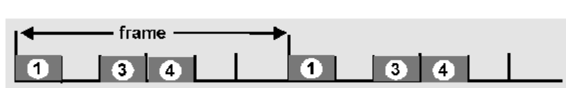
Three broad classes:

- Channel Partitioning
  - Divide channel into smaller “pieces” (time slots, frequency, code)
  - Allocate piece to node for exclusive use
- Random Access
  - Channel not divided, allow collisions “recover” from collisions
- “Taking turns”
  - Nodes take turns, but nodes with more to send can take longer turns

Channel Partitioning MAC Protocols: TDMA

TDMA: time division multiple access

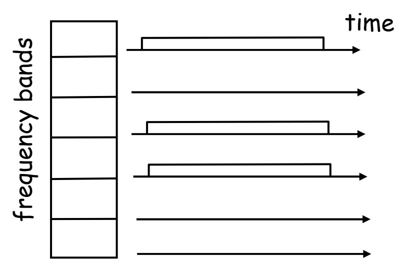
- Access to channel in "rounds“
- Each station gets fixed length slot (length = pkt trans time) in each round
- Unused slots go idle
- Example: 6-station LAN, 1,3,4 have pkt, slots 2,5,6 idle



Channel Partitioning MAC Protocols: FDMA

FDMA: frequency division multiple access

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



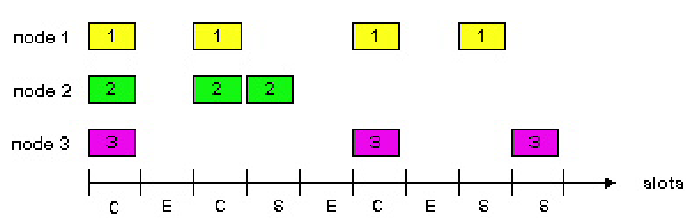
Random Access Protocols

- 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
  - How to recover from collisions (e.g., via delayed retransmissions)
- Examples of random access MAC protocols
  - Slotted ALOHA
  - ALOHA
  - CSMA, CSMA/CD, CSMA/CA

Slotted ALOHA (1/2)

- Assumptions
  - All frames same size
  - Time is divided into equal size slots, time to transmit 1 frame
  - Nodes start to transmit frames only at beginning of slots
  - Nodes are synchronized
  - If 2 or more nodes transmit in slot, all nodes detect collision
- Operation
  - When node obtains fresh frame, it transmits in next slot
  - No collision, node can send new frame in next slot
  - if collision, node retransmits frame in each subsequent slot with prob. p until success

Slotted ALOHA (2/2)



- Pros
  - Single active node can continuously transmit at full rate of channel
  - Highly decentralized: only slots in nodes need to be in sync
  - Simple
- Cons
  - Collisions, wasting slots
  - Idle slots
  - Nodes may be able to detect collision in less than time to transmit packet
  - 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

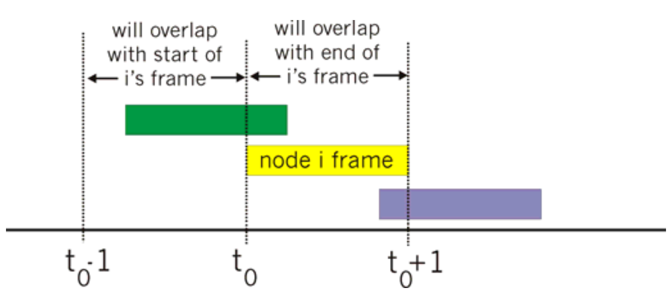
- Suppose N nodes with many frames to send, each transmits in slot with probability *p*
- Prob that node 1 has success in a slot = $p(1-p)^{N-1}$
- Prob that any node has a success = $Np(1-p)^{N-1}$

- 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 1/e = .37

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

Pure (Unslotted) ALOHA

- Unslotted Aloha: simpler, no synchronization
- When frame first arrives
  - Transmit immediately
  - Collision probability increases:
    - Frame sent at t_0 collides with other frames sent in $[t_0-1, t_0+1]$



Pure ALOHA Efficiency

- P(success by given node) = P(node transmits) .

  P(no other node transmits in $[p_0-1,p_0]$.

  P(no other node transmits in $[p_0-1,p_0]$

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

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

  … choosing optimum p and then letting n -> infty ...

  = 1/(2e) = .18

Even worse !

CSMA (Carrier Sense Multiple Access)

CSMA: listen before transmit

If channel sensed idle: transmit entire frame

- If channel sensed busy, defer transmission

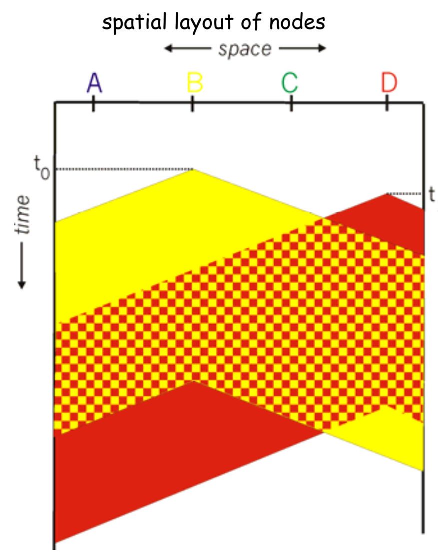
- Human analogy: don’t interrupt others!

CSMA Collisions

Collisions *can* still occur:

- Propagation delay means two nodes may not hear each other's transmission

- Collision
  - Entire packet transmission time wasted
- Note
  - Role of distance & propagation delay in determining collision probability



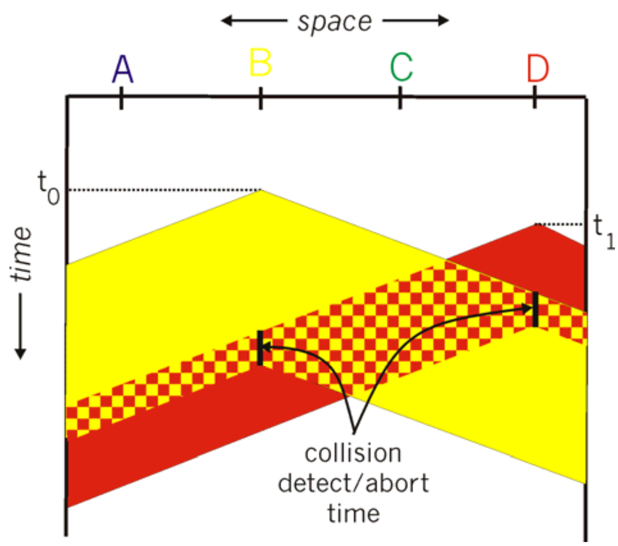
CSMA/CD (Collision Detection)

CSMA/CD: carrier sensing, deferral as in CSMA

- Collisions *detected* within short time
- Colliding transmissions aborted, reducing channel wastage

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

CSMA/CD Collision Detection



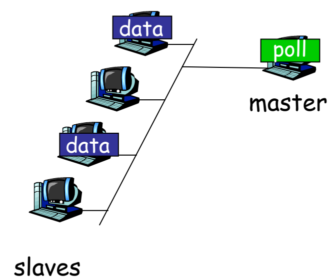
"Taking Turns" MAC Protocols (1/2)

- 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!
- Random access MAC protocols
  - Efficient at low load: single node can fully utilize channel
  - High load: collision overhead
- "Taking turns" protocols
  - Look for best of both worlds!

“Taking Turns” MAC protocols

Polling:

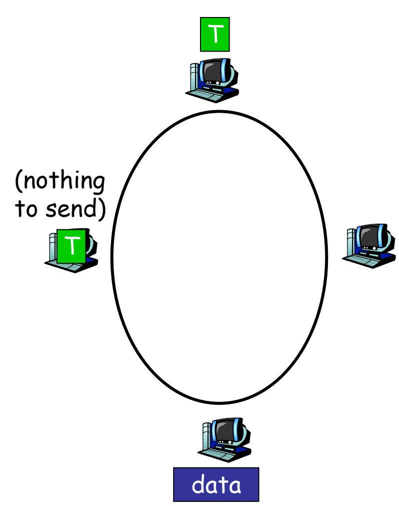
- master node “invites” slave nodes to transmit in turn
- typically used with “dumb” slave devices
- concerns:
  - polling overhead
  - latency
  - single point of failure (master)



“Taking Turns” MAC protocols

Token passing:

- control **token** passed from one node to next sequentially.
- token message
- concerns:
  - token overhead
  - latency
  - single point of failure (token)



Summary of MAC Protocols

- What do you do with a shared media?
  - Channel Partitioning, by time, frequency or code
    - Time Division, Frequency Division
  - Random partitioning (dynamic)
    - ALOHA, S-ALOHA, CSMA, CSMA/CD
    - Carrier sensing: easy in some technologies (wire), hard in others (wireless)
    - CSMA/CD used in Ethernet
    - CSMA/CA used in 802.11
  - Taking Turns
    - Polling from a central site, token passing

LAN Technologies

Data link layer so far:

- Services, error detection/correction, multiple access

Next: LAN technologies

- Addressing
- Ethernet

Data Link Control - Roadmap

2 Data Link Control

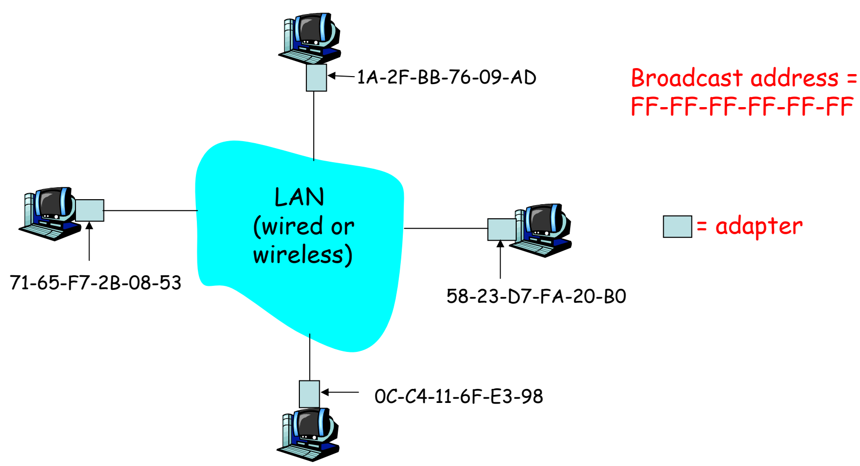
- Introduction and Services
- Error Detection and Correction
- Multiple Access Protocols
- Link Layer Addressing
- Ethernet
- Additional Topics

MAC Addresses and ARP

- 32-bit IP address
  - *network-layer* address
  - Used to get datagram to destination IP subnet
- MAC (or LAN or physical or Ethernet) address
  - Used to get frame from one interface to another physically-connected interface (same network)
  - 48 bit MAC address (for most LANs) burned in the adapter ROM

LAN Addresses and ARP

Each adapter on LAN has unique LAN address

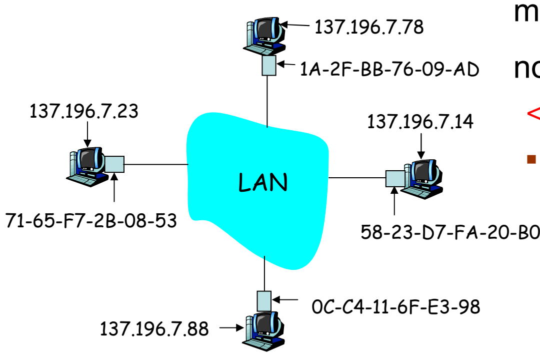


LAN Address

- MAC address allocation administered by IEEE
- Manufacturer buys portion of MAC address space (to assure uniqueness)
- Analogy
  (a) MAC address: like Social Security Number
  (b) IP address: like postal address
- MAC flat address ➜ portability
  - Can move LAN card from one LAN to another
- IP hierarchical address NOT portable
  - Depends on IP subnet to which node is attached

ARP: Address Resolution Protocol

Question: how to determine MAC address of B knowing B's IP address?



- Each IP node (Host, Router) on LAN has ARP table
- ARP Table: IP/MAC address mappings for some LAN nodes

  < IP address; MAC address; TTL>

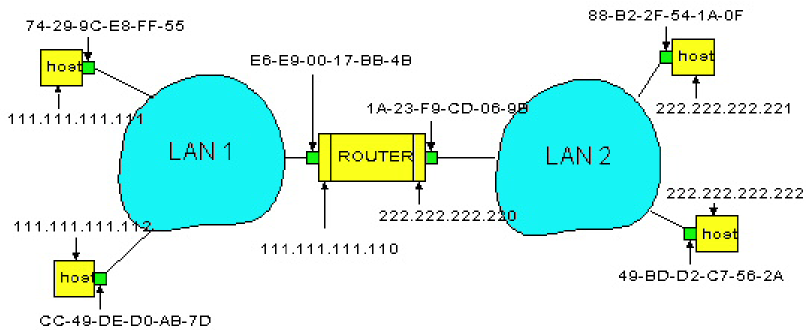
  - TTL (Time To Live): time after which address mapping will be forgotten (typically 20 min)

ARP 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 machines on LAN receive ARP query
- B receives ARP packet, replies to A with its (B's) MAC address
  - Frame sent to A's MAC address (unicast)
- A caches (saves) IP-to-MAC address pair in its ARP table until information becomes old (times out)
  - Soft state: information that times out (goes away) unless refreshed
- ARP is "plug-and-play"
  - Nodes create their ARP tables without intervention from net administrator

Routing to Another LAN (1/2)

walkthrough: send datagram from A to B via R
assume A know’s B IP address

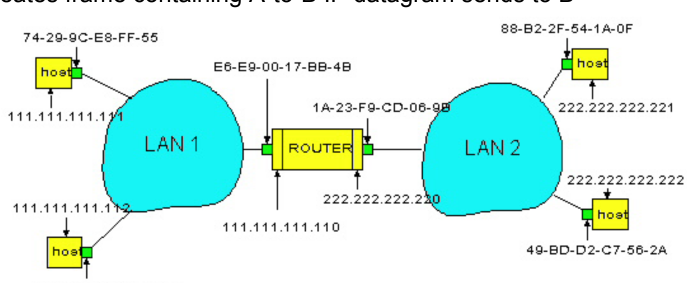


- Two ARP tables in router R, one for each IP network (LAN)
- In routing table at source Host, find router 111.111.111.110
- In ARP table at source, find MAC address E6-E9-00-17-BB-4B, etc

Routing to Another LAN (2/2)

- A creates datagram with source A, destination B
- A uses ARP to get R’s 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
- R’s adapter receives frame
- R removes IP datagram from Ethernet frame, sees its destined to B
- R uses ARP to get B’s MAC address
- R creates frame containing A-to-B IP datagram sends to B

This is a really important example – make sure you understand!



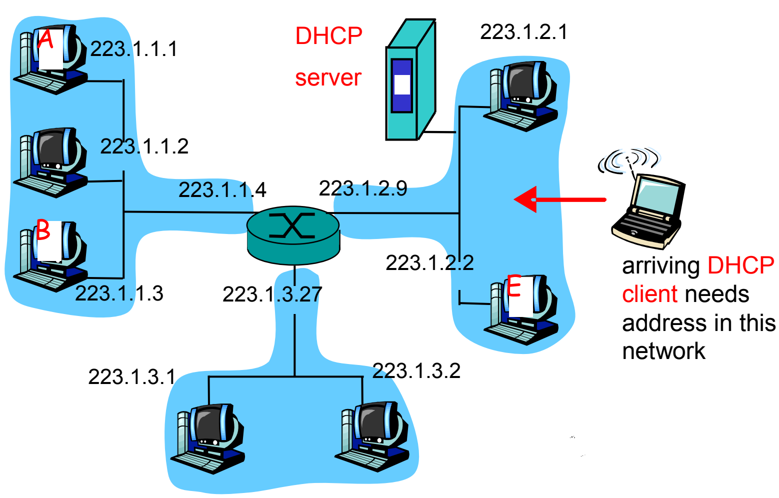
DHCP: Dynamic Host Configuration Protocol

Goal: allow host to *dynamically* obtain its IP address from network server when it joins network

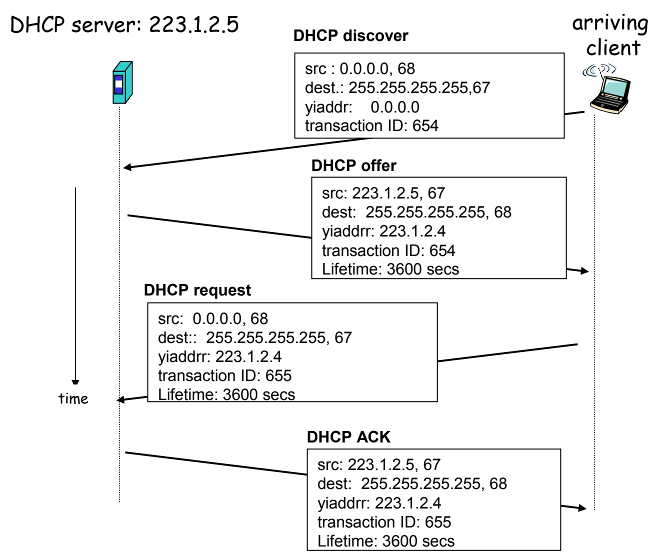
- Can renew its lease on address in use
- Allows reuse of addresses (only hold address while connected an "on")
- Support for mobile users who want to join network (more shortly)

- DHCP overview
  - Host broadcasts "DHCP discover" msg
  - DHCP server responds with "DHCP offer" msg
  - Host requests IP address: "DHCP request" msg
  - DHCP server sends address: "DHCP ack" msg

DHCP Client-Server Scenario (1/2)



DHCP Client-Server Scenario (2/2)



Data Control Link - Roadmap

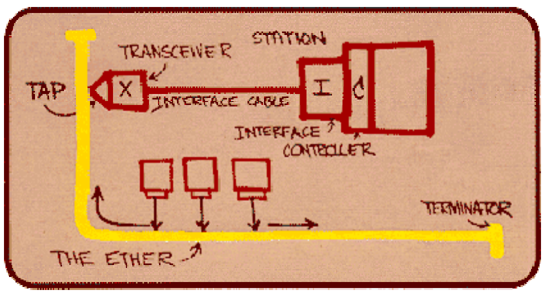
2 Data Control Link

- Introduction and Services
- Error Detection and Correction
- Multiple Access Protocols
- Link Layer Addressing
- Ethernet
- Additional Topics

Ethernet

“Dominant” wired LAN technology

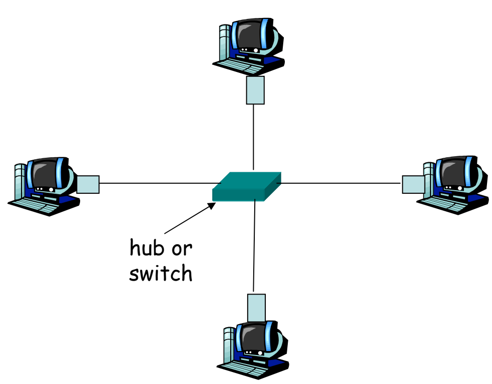
- Cheap $20 for 100Mbs!
- First widely used LAN technology
- Simpler, cheaper than token LANs and ATM
- Kept up with speed race: 10 Mbps – 10 Gbps



Metcalfe’s Ethernet sketch

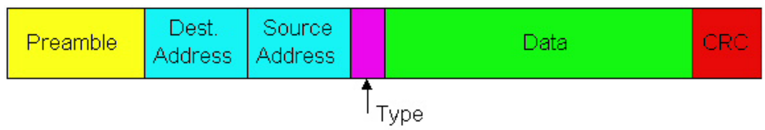
Star Topology

- Bus topology popular through mid 90s
- Now star topology prevails
- Connection choices: hub or switch (will not cover this)



Ethernet Frame Structure (1/2)

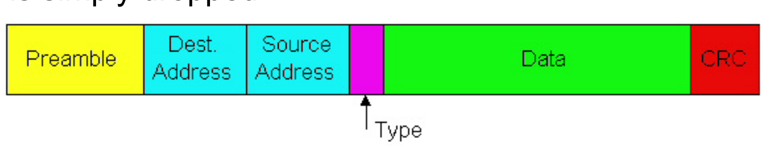
- Sending adapter encapsulates IP datagram (or other network layer protocol packet) in Ethernet frame



- Preamble
  - 7 bytes with pattern 10101010 followed by one byte with pattern 10101011
  - Used to synchronize receiver, sender clock rates

Ethernet Frame Structure (2/2)

- Addresses: 6 bytes
  - If adapter receives frame with matching destination address, or with broadcast address (eg ARP packet), it passes data in frame to net-layer protocol
  - Otherwise, adapter discards frame
- Type: indicates the higher layer protocol (mostly IP but others may be supported such as Novell IPX and AppleTalk)
- CRC: checked at receiver, if error is detected, the frame is simply dropped



Unreliable, Connectionless Service

- Connectionless: No handshaking between sending and receiving adapter
- Unreliable: receiving adapter doesn't send acks or nacks to sending adapter
  - Stream of datagrams passed to network layer can have gaps
  - Gaps will be filled if app is using TCP
  - Otherwise, app will see the gaps

Ethernet Uses CSMA/CD

- No slots
- Adapter doesn't transmit if it senses that some other adapter is transmitting, that is, carrier sense
- Transmitting adapter aborts when it senses that another adapter is transmitting, that is, collision detection
- Before attempting a retransmission, adapter waits a random time, that is, random access

Ethernet CSMA/CD Algorithm (1/2)

1. Adaptor receives datagram from net layer & creates frame
2. If adapter senses channel idle, it starts to transmit frame. If it senses channel busy, waits until channel idle and then transmits
3. If adapter transmits entire frame without detecting another transmission, the adapter is done with frame !
4. If adapter detects another transmission while transmitting, aborts and sends jam signal
5. After aborting, adapter enters **exponential backoff**: after the mth collision, adapter chooses a K at random from $\{0,1,2,\ldots,2^m-1\}$. Adapter waits K·512 bit times and returns to Step 2

Ethernet CSMA/CD Algorithm (2/2)

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

Bit time: .1 microsec for 10 Mbps Ethernet ;
for K=1023, wait time is about 50 msec

See/interact with Java applet on textbook companion Web site:
highly recommended !

Exponential Backoff

- *Goal*: adapt retransmission attempts to estimated current load
  - heavy load: random wait will be longer
- First collision: choose K from {0,1}; delay is K· 512 bit transmission times
- After second collision: choose K from {0,1,2,3}…
- After ten collisions, choose K from {0,1,2,3,4,…,1023}

CSMA/CD Efficiency

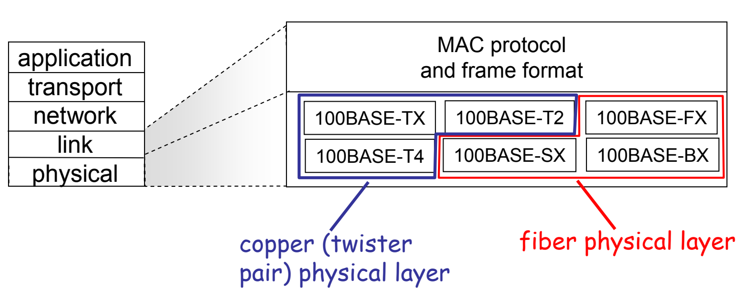
- T_{prop} = max prop between 2 nodes in LAN
- t_{trans} = time to transmit max-size frame

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

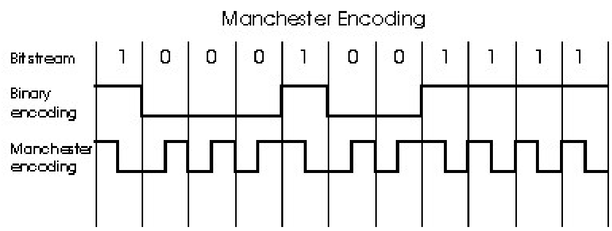
- Efficiency goes to 1 as t_{prop} goes to 0
- Goes to 1 as t_{trans} goes to infinity
- Much better than ALOHA, but still decentralized, simple, and cheap

802.3 Ethernet Standards: Link & Physical Layers

- *many* different Ethernet standards
  - » common MAC protocol and frame format
  - » different speeds: 2 Mbps, 10 Mbps, 100 Mbps, 1Gbps, 10G bps
  - » different physical layer media: fiber, cable



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!
- Hey, this is physical-layer stuff!

Data Control Link - Roadmap

2 Data Control Link

- Introduction and Services
- Error Detection and Correction
- Multiple Access Protocols
- Link Layer Addressing
- Ethernet
- Additional Topics

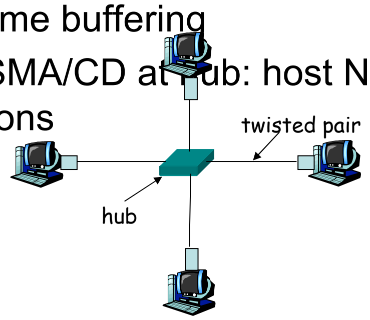
Link Layer

- Introduction and services
- Error detection and correction
- Multiple access protocols
- Link-layer Addressing
- Ethernet
- Link-layer switches, LANs, VLANs
- PPP
- Link virtualization: MPLS
- A day in the life of a web request

Hubs

… physical-layer ("dumb") repeaters:

- bits coming in one link go out to *all* other links at same rate
- all nodes connected to hub can collide with one another
- no frame buffering
- no CSMA/CD at hub: host NICs detect collisions

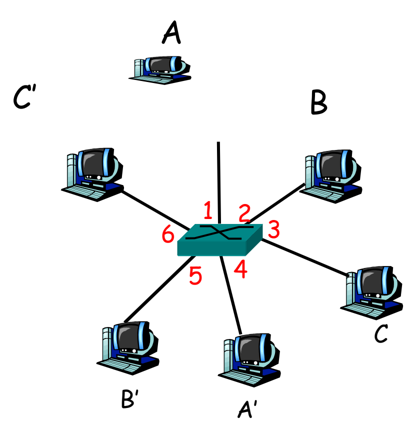


Switch

- link-layer device: smarter than hubs, take *active* role
  - » store, forward Ethernet frames
  - » examine incoming frame's MAC address, selectively forward frame to one-or-more outgoing links when frame is to be forwarded on 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

Switch: allows *multiple* simultaneous transmissions

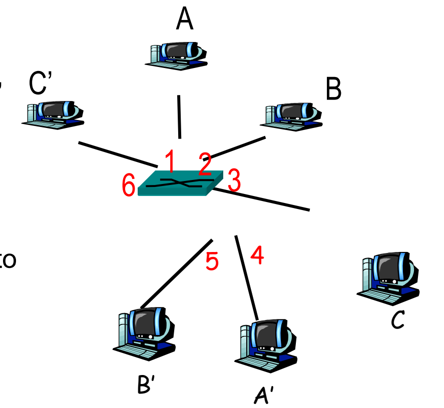
- hosts have dedicated, direct connection to switch
- switches buffer packets
- Ethernet protocol used on *each* incoming link, but no collisions; full duplex
  - » each link is its own collision domain
- *switching:* A-to-A' and B-to-B' simultaneously, without collisions
  - » not possible with dumb hub



switch with six interfaces (1,2,3,4,5,6)

Switch Table

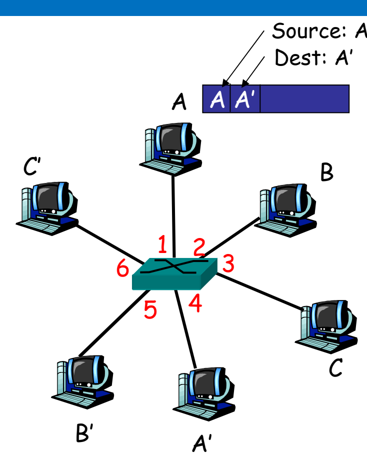
- *Q:* how does switch know that A' reachable via interface 4, B' reachable via interface 5?
- *A:* each switch has a switch table, each entry:
  - (MAC address of host, interface to reach host, time stamp)
- looks like a routing table!
- *Q:* how are entries created, maintained in switch table?
  - something like a routing protocol?



switch with six interfaces (1,2,3,4,5,6)

Switch: self-learning

- switch *learns* which hosts can be reached through which interfaces
  - when frame received, switch "learns" location of sender: incoming LAN segment
  - records sender/location pair in switch table



MAC addr	interface	TTL
A	1	60

Switch table (initially empty)

Switch: frame filtering/forwarding

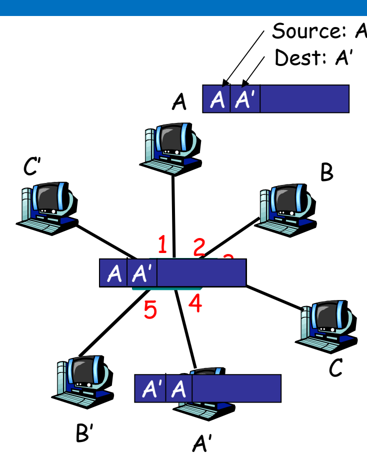
When frame received:

1. record link associated with sending host
2. index switch table using MAC dest address
3. **if** entry found for destination
   then {
   if dest on segment from which frame arrived
   then drop the frame
   else forward the frame on interface indicated
   }
   else flood

forward on all but the interface on which the frame arrived

Self-learning, forwarding: example

- frame destination unknown: *flood*
- destination A location known: *selective send*

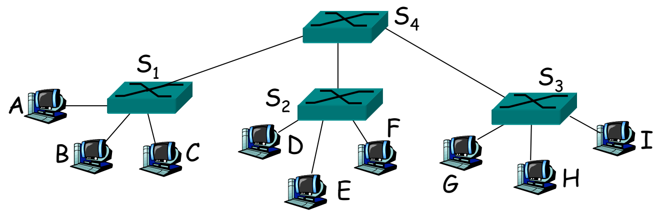


MAC addr	interface	TTL
A	1	60
A'	4	60

Switch table (initially empty)

Interconnecting switches

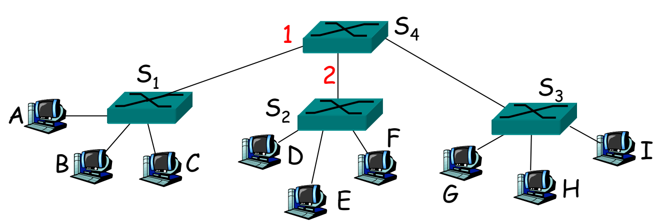
- switches can be connected together



- *Q:* sending from A to G - how does S_1 know to forward frame destined to F via S_4 and S_3?
- *A:* self learning! (works exactly the same as in single-switch case!)

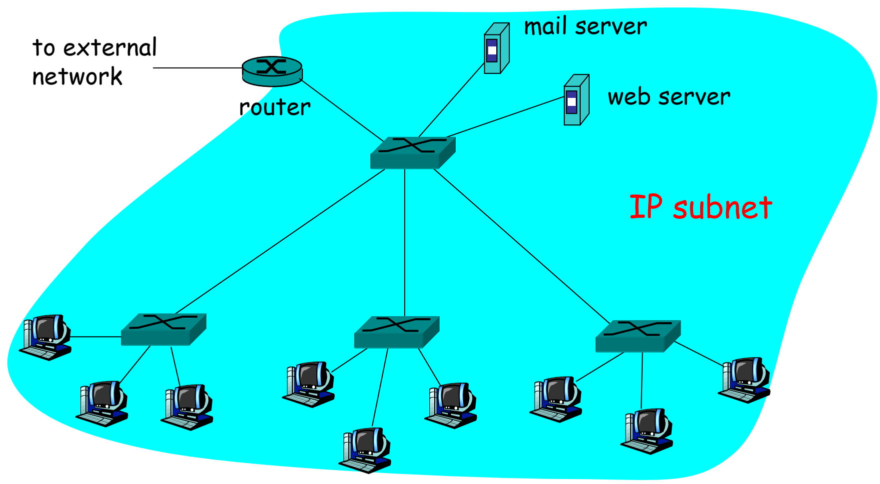
Self-learning multi-switch example

Suppose C sends frame to I, I responds to C



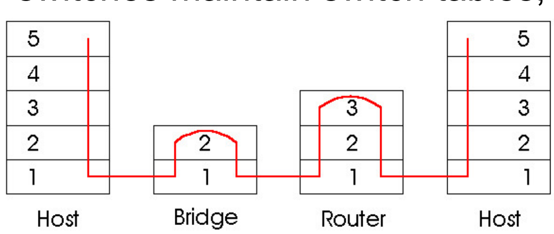
- *Q:* show switch tables and packet forwarding in S_1, S_2, S_3, S_4

Institutional network



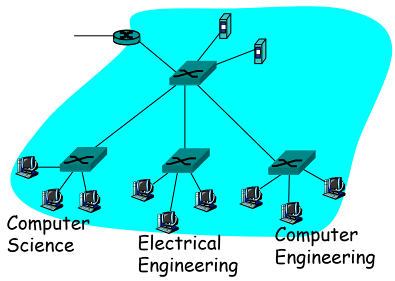
Switches vs. Routers

- both store-and-forward devices
  - » routers: network layer devices (examine network layer headers)
  - » switches are link layer devices
- routers maintain routing tables, implement routing algorithms
- switches maintain switch tables, implement



VLANs: motivation

What's wrong with this picture?



What happens if:

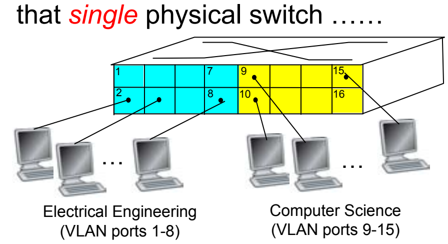
- CS user moves office to EE, but wants connect to CS switch?
- single broadcast domain:
  - all layer-2 broadcast traffic (ARP, DHCP) crosses entire LAN (security/privacy, efficiency issues)
- each lowest level switch has only few ports in use

VLANs

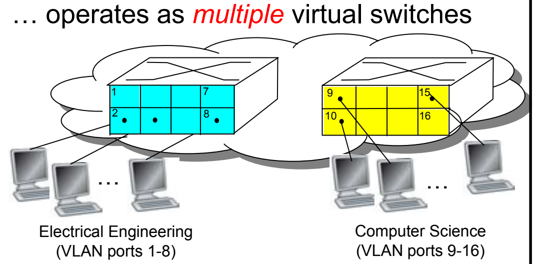
Virtual Local Area Network

Switch(es) supporting VLAN capabilities can be configured to define multiple ***virtual*** LANS over single physical LAN infrastructure.

Port-based VLAN: switch ports grouped (by switch management software) so that *single* physical switch ……

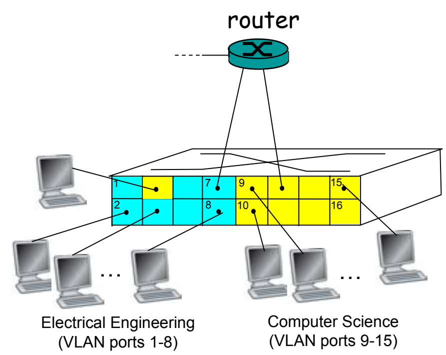


… operates as *multiple* virtual switches

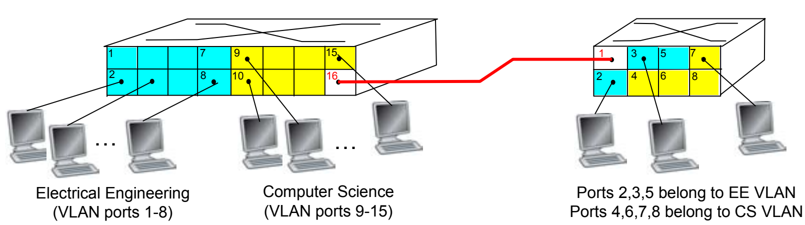


Port-based VLAN

- *traffic isolation:* frames to/from ports 1-8 can *only* reach ports 1-8
  - » can also define VLAN based on MAC addresses of endpoints, rather than switch port
- *dynamic membership:* ports can be dynamically assigned among VLANs
- *forwarding between VLANS:* done via routing (just as with separate switches)
  - » in practice vendors sell combined switches plus routers

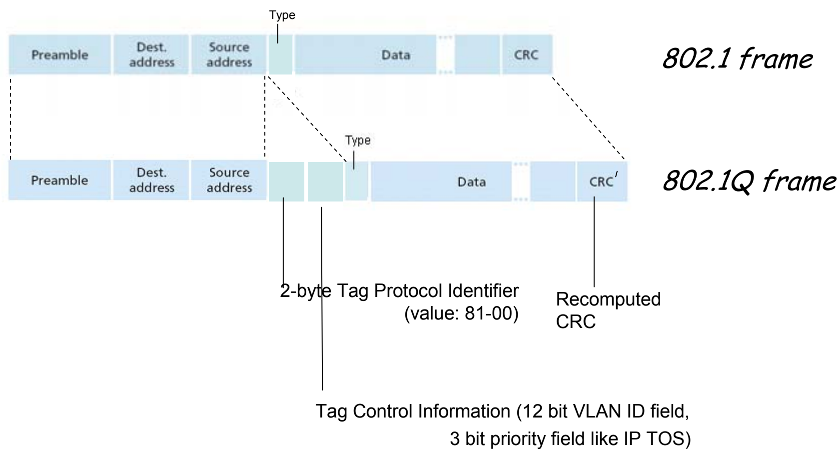


VLANS spanning multiple switches



- *trunk port:* carries frames between VLANS defined over multiple physical switches
  - » frames forwarded within VLAN between switches can't be vanilla 802.1 frames (must carry VLAN ID info)
  - » 802.1q protocol adds/removed additional header fields for frames forwarded between trunk ports

802.1Q VLAN frame format



Link Layer

- Introduction and services
- Error detection and correction
- Multiple access protocols
- Link-Layer Addressing
- Ethernet
- Link-layer switches
- PPP
- Link virtualization: MPLS
- A day in the life of a web request

Point to Point Data Link Control

- 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
- popular point-to-point DLC protocols:
  - » PPP (point-to-point protocol)
  - » HDLC: High level data link control (Data link used to be considered "high layer" in protocol stack!

PPP Design Requirements [RFC 1557]

- packet framing: encapsulation of network-layer datagram in data link frame
  - » carry network layer data of any network layer protocol (not just IP) *at same time*
  - » ability to demultiplex upwards
- bit transparency: must carry any bit pattern in the data field
- error detection (no correction)
- connection liveness: detect, signal link failure to network layer
- network layer address negotiation: endpoint can learn/configure each other's network address

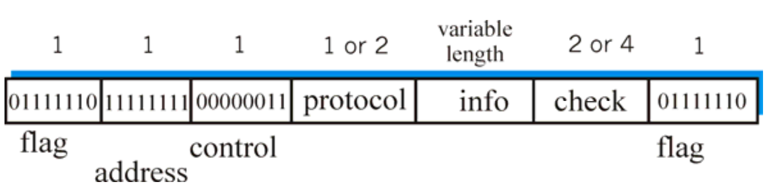
PPP non-requirements

- no error correction/recovery
- no flow control
- out of order delivery OK
- no need to support multipoint links (e.g., polling)

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

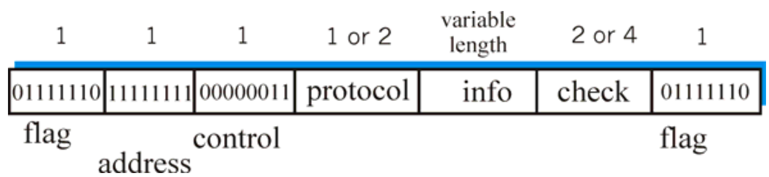
PPP Data Frame

- Flag: delimiter (framing)
- Address: does nothing (only one option)
- Control: does nothing; in the future possible multiple control fields
- Protocol: upper layer protocol to which frame delivered (eg, PPP-LCP, IP, IPCP, etc)



PPP Data Frame

- info: upper layer data being carried
- check: cyclic redundancy check for error detection

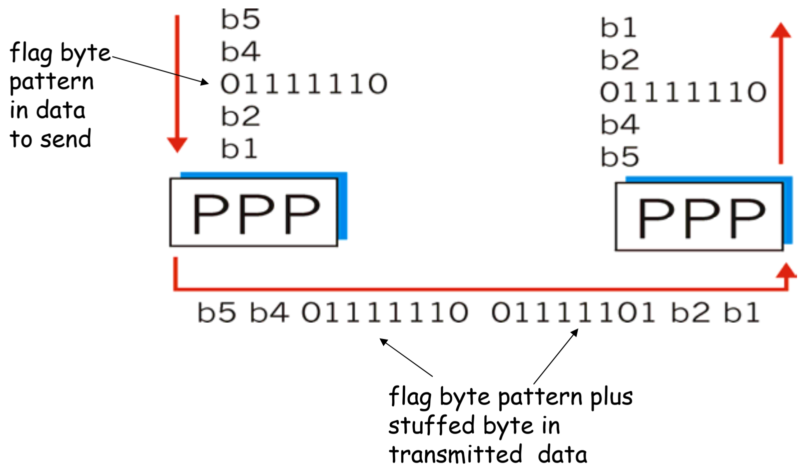


Byte Stuffing

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

- Sender: adds ("stuffs") extra < 01111110> byte after each < 01111110> *data* byte
- Receiver:
  - » two 01111110 bytes in a row: discard first byte, continue data reception
  - » single 01111110: flag byte

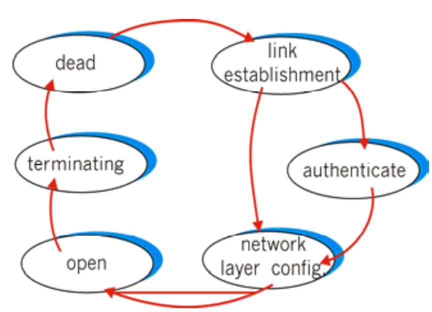
Byte Stuffing



PPP Data Control Protocol

Before exchanging network-layer data, data link peers must

- configure PPP link (max. frame length, authentication)
- learn/configure network layer information
  - for IP: carry IP Control Protocol (IPCP) msgs (protocol field: 8021) to configure/learn IP address



Link Layer

- Introduction and services
- Error detection and correction
- Multiple access protocols
- Link-Layer Addressing
- Ethernet
- Link-layer switches
- PPP
- Link virtualization: MPLS
- A day in the life of a web request

Virtualization of networks

Virtualization of resources: powerful abstraction in systems engineering:

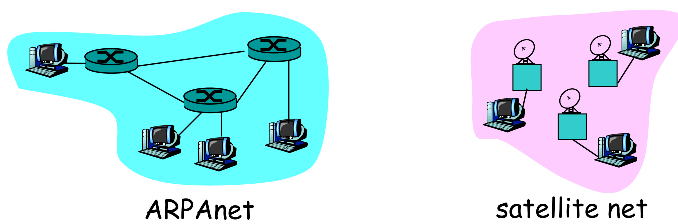
- computing examples: virtual memory, virtual devices
  - Virtual machines: e.g., java
  - IBM VM os from 1960's/70's
- layering of abstractions: don't sweat the details of the lower layer, only deal with lower layers abstractly

The Internet: virtualizing networks

1974: multiple unconnected nets … differing in:

- ARPAnet
- data-over-cable networks
- packet satellite network (Aloha)
- packet radio network

- addressing conventions
- packet formats
- error recovery
- routing



"A Protocol for Packet Network Intercommunication",
V. Cerf, R. Kahn, IEEE Transactions on Communications,
May, 1974, pp. 637-648.

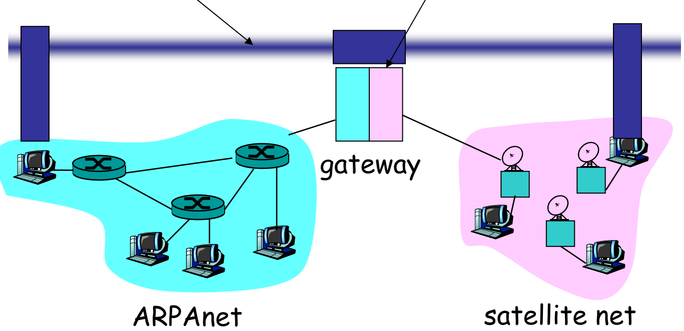
The Internet: virtualizing networks

Internetwork layer (IP):

- addressing: internetwork appears as single, uniform entity, despite underlying local network heterogeneity
- network of networks

Gateway:

- "embed internetwork packets in local packet format or extract them"
- route (at internetwork level) to next gateway



Cerf & Kahn's Internetwork Architecture

What is virtualized?

- two layers of addressing: internetwork and local network
- new layer (IP) makes everything homogeneous at internetwork layer
- underlying local network technology
  - » cable
  - » satellite
  - » 56K telephone modem
  - » today: ATM, MPLS

… "invisible" at internetwork layer. Looks like a link layer technology to IP!

ATM and MPLS

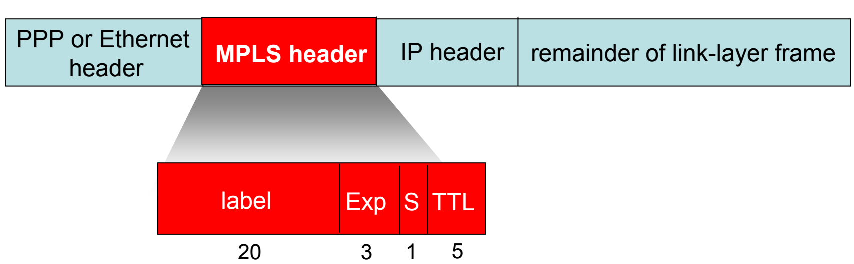
- ATM, MPLS separate networks in their own right
  - » different service models, addressing, routing from Internet
- viewed by Internet as logical link connecting IP routers
  - » just like dialup link is really part of separate network (telephone network)
- ATM, MPLS: of technical interest in their own right

Asynchronous Transfer Mode: ATM

- **1990's/00 standard for high-speed** (155Mbps to 622 Mbps and higher) *Broadband Integrated Service Digital Network* architecture
- Goal: *integrated, end-end transport of carry voice, video, data*
  - » meeting timing/QoS requirements of voice, video (versus Internet best-effort model)
  - » "next generation" telephony: technical roots in telephone world
  - » packet-switching (fixed length packets, called "cells") using virtual circuits

Multiprotocol label switching (MPLS)

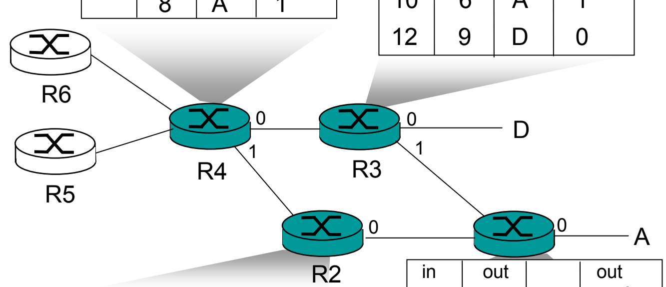
- initial goal: speed up IP forwarding by using fixed length label (instead of IP address) to do forwarding
  - » borrowing ideas from Virtual Circuit (VC) approach
  - » but IP datagram still keeps IP address!



MPLS capable routers

- a.k.a. label-switched router
- forwards packets to outgoing interface based only on label value (don't inspect IP address)
  - » MPLS forwarding table distinct from IP forwarding tables
- signaling protocol needed to set up forwarding
  - » RSVP-TE
  - » forwarding possible along paths that IP alone would not allow (e.g., source-specific routing) !!
  - » use MPLS for traffic engineering
- must co-exist with IP-only routers

MPLS forwarding tables



R4:

in label	out label	dest	out interface
	10	A	0
	12	D	0
	8	A	1

R3:

in label	out label	dest	out interface
10	6	A	1
12	9	D	0

R2:

in label	out label	dest	out interface
8	6	A	0

R1:

in label	out label	dest	out interface
6	-	A	0

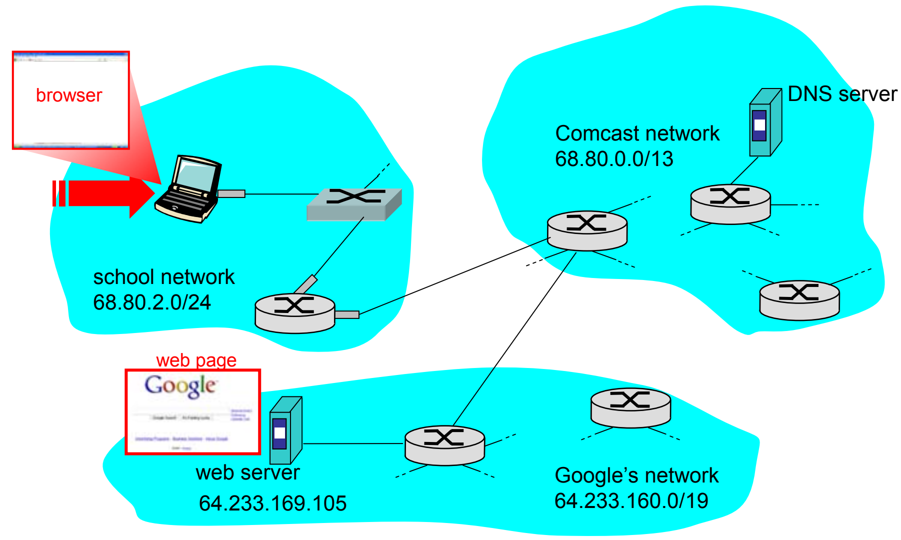
Link Layer

- Introduction and services
- Error detection and correction
- Multiple access protocols
- Link-Layer Addressing
- Ethernet
- Link-layer switches
- PPP
- Link virtualization: MPLS
- A day in the life of a web request

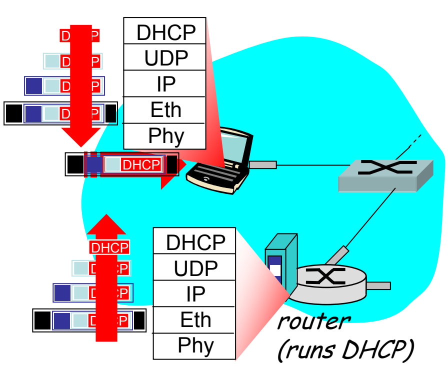
Synthesis: a day in the life of a web request

- 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

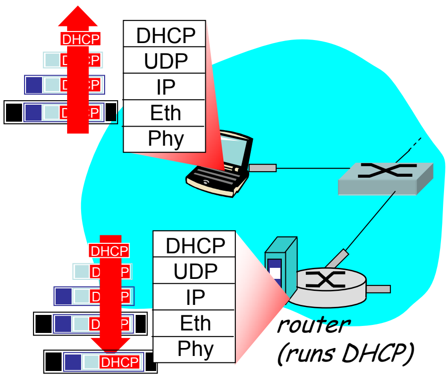


A day in the life… connecting to the Internet



- connecting laptop needs to get its own IP address, addr of first-hop router, addr of DNS server: use ***DHCP***
- DHCP request ***encapsulated*** in ***UDP***, encapsulated in ***IP***, encapsulated in ***802.1*** Ethernet
- Ethernet frame ***broadcast*** (dest: FFFFFFFFFFFF) on LAN, received at router running ***DHCP*** server
- Ethernet ***demux'ed*** to IP demux'ed, UDP demux'ed to DHCP

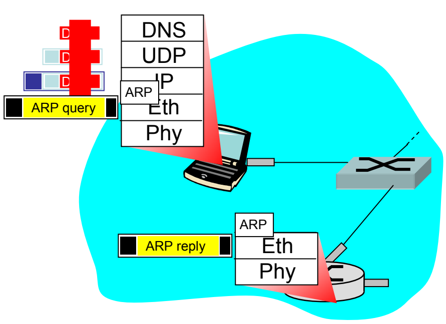
A day in the life… connecting to the Internet



- DHCP server formulates ***DHCP ACK*** containing client's IP address, IP address of first-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

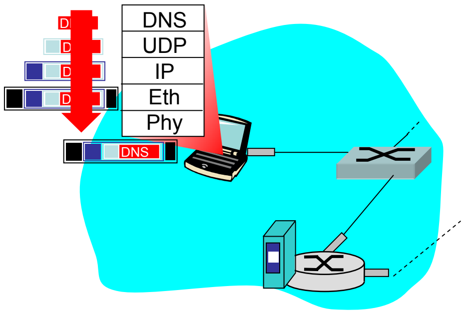
Client now has IP address, knows name & addr of DNS server, IP address of its first-hop router

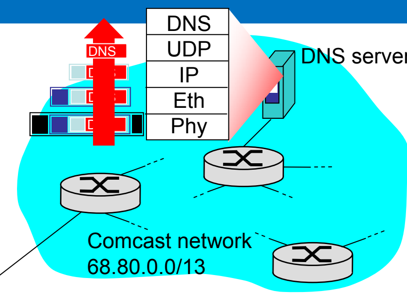
A day in the life… ARP (before DNS, before HTTP)



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

A day in the life… using DNS

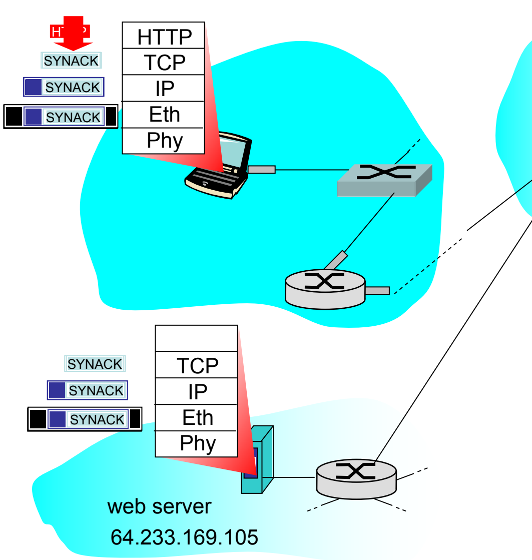


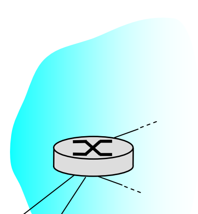


- IP datagram containing DNS query forwarded via LAN switch from client to 1st hop router

- IP datagram forwarded from campus network into comcast network, routed (tables created by ***RIP, OSPF, IS-IS*** and/or ***BGP*** routing protocols) to DNS server
- demux'ed to DNS server
- DNS server replies to client with IP address of www.google.com

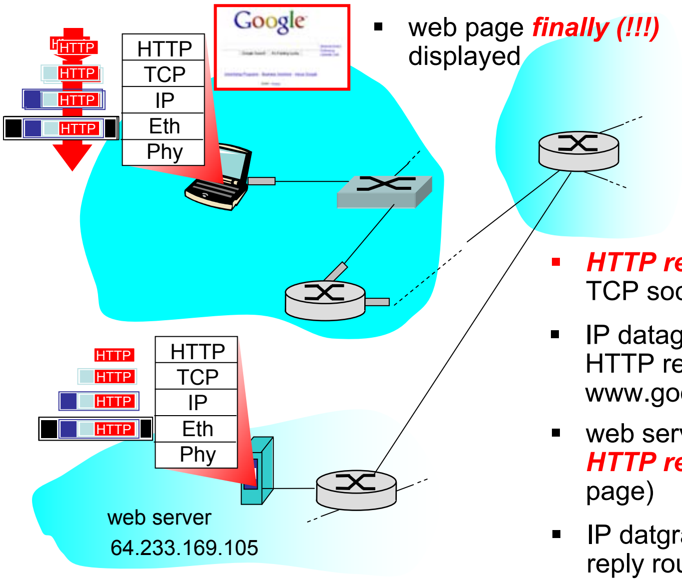
A day in the life… TCP connection carrying HTTP





- 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 SYNACK*** (step 2 in 3-way handshake)
- TCP ***connection established!***

A day in the life… HTTP request/reply



- web page ***finally (!!!)*** displayed

- ***HTTP request*** sent into TCP socket
- 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

DLC - Summary

- principles behind data link layer services:
  - » error detection, correction
  - » sharing a broadcast channel: multiple access
  - » link layer addressing
- instantiation and implementation of various link layer technologies
  - » Ethernet
  - » switched LANS, VLANs
  - » PPP
  - » virtualized networks as a link layer: MPLS
- synthesis: a day in the life of a web request

DLC: let's take a breath

- journey down protocol stack *complete* (except PHY)
- solid understanding of networking principles, practice
- ….. could stop here …. but *lots* of interesting topics!
  - » wireless
  - » multimedia
  - » security
  - » network management

Agenda

1 Session Overview

2 Data Link Control

3 Summary and Conclusion

Assignments & Readings

- Readings

  - Chapter 5
- Assignment #3

Next Session: Data Link Control (Part II)