# Chapter 5: DataLink Layer

Course on Computer Communication and Networks, CTH/GU

The slides are adaptation of the slides made available by the authors of the course's main textbook

Computer Networking:
*A Top Down Approach*
5th edition.
Jim Kurose, Keith Ross
Addison-Wesley, 2009



# Chapter 5: The Data Link Layer

**data-link layer** has responsibility of transferring frames from one node to adjacent node over a link

## Our goals:

- understand principles behind data link layer services:
  - error detection, correction
  - sharing a broadcast channel: multiple access
  - link layer addressing
  - reliable data transfer, flow control: *done!*
- instantiation and implementation of various link layer technologies



# Link Layer

- 5.1 Introduction and services
- Framing
- 5.2 Error detection and correction
- 5.3Multiple access protocols

LAN technology
- 5.5 Ethernet
- 5.6 Interconnection
- 5.4 Link-Layer Addressing
- 5.7 PPP
- 5.9 A day in the life of a web request

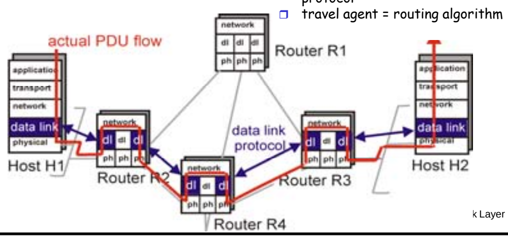
(5.8 Link Virtualization: ATM and MPLS)



# 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

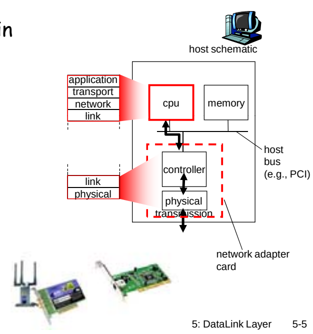
- transportation analogy
- trip from Princeton to Lausanne
  - limo: Princeton to JFK
  - plane: JFK to Geneva
  - train: Geneva to Lausanne
- tourist = datagram
- transport segment = communication link
- transportation mode = link layer protocol
- travel agent = routing algorithm





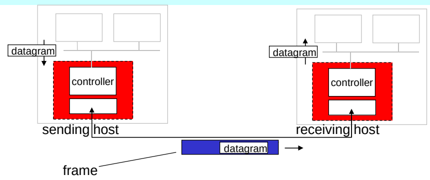

# Where is the link layer implemented?

- in each and every host
- link layer implemented in "adaptor" (aka *network interface card* NIC)
  - Ethernet 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

## Link Layer Services

- 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, flow ctrl
  - Control when errors + pace between adjacent sending and receiving nodes
    - we learned how to do this already (chapter 3)!
  - 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 (more)

- *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



## Link Layer

- 5.1 Introduction and services
- Framing
- 5.2 Error detection and correction
- 5.3Multiple access protocols

LAN technology
- 5.5 Ethernet
- 5.6 Interconnection
- 5.4 Link-Layer Addressing
- 5.7 PPP
- 5.9 A day in the life of a web request

(5.8 Link Virtualization: ATM and MPLS)



## Framing

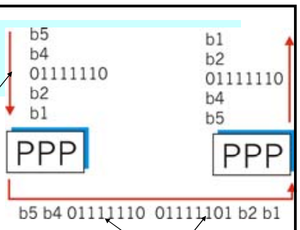
- to detect possible bit stream errors in the physical layer, the data link layer groups bits from the network layer into discrete **frames**
- the receiver must be able to detect the beginning and the end of the frame

Example methods:
- **Clock-based + Character count** : physical-clock synchronization: much dependent on clock drifts + the counter could be garbled up during transmission



## Byte Stuffing

flag byte pattern in data to send




flag byte pattern plus stuffed byte in transmitted data

- <01111110> delimits beginning, end of frame
- "data transparency": data field must be allowed to include <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



## Framing techniques: examples (cont)

...

**Physical layer coding violation**: exploits special encodings at the physical layer, e.g. Manchester encoding (see next ...)

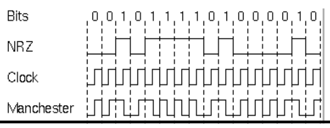
## Encoding

**Problem:** Simple binary encoding (aka Non-Return to Zero, NRZ) introduces problems:

- consecutive 0's or 1's can lead to a situation called *baseline wander* (hard to distinguish signal values)
- hard to recover the clock

**More robust encoding:**

- **Manchester:** XOR NRZ with clock






## Link Layer

- 5.1 Introduction and services
- Framing
- 5.2 Error detection and correction
- 5.3Multiple access protocols

LAN technology

- 5.5 Ethernet
- 5.6 Interconnection
- 5.4 Link-Layer Addressing
- 5.7 PPP
- 5.9 A day in the life of a web request

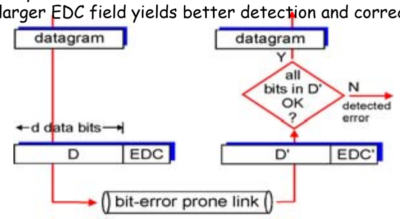
(5.8 Link Virtualization: ATM and MPLS)



## Error Detection

EDC= Error Detection and Correction bits (redundancy)
D = Data protected by error checking, may include header fields

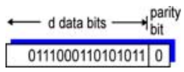
- Error detection not 100% reliable!
  - protocol may miss some errors, but this should happen only rarely
  - larger EDC field yields better detection and correction



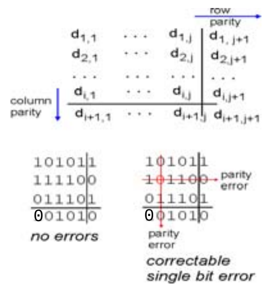



## Parity Checking

**Single Bit Parity:**
Detect single bit errors




**Two Dimensional Bit Parity:**
Detect *and correct* single bit errors





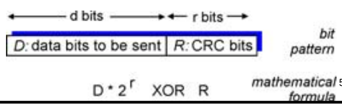

## Internet checksums

**TCP (UDP)'s checksum:**

- segment contents = sequence of 16-bit integers
- checksum: addition (1's complement sum) of segment contents
- sender puts checksum value into UDP (TCP) checksum field

**Cyclic redundancy check (CRC)**

- data bits, D = binary number
- consider r+1 bit pattern (generator), G
- goal: compute 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 errors on less than r+1 bits
- International standards for G (CRC polynomials)






## CRC Example

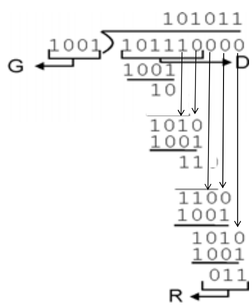
Recall we want:

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

*equivalently:*

if we divide $D \cdot 2^r$ by G, want remainder R

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



# Link Layer

- 5.1 Introduction and services
- Framing
- 5.2 Error detection and correction
- 5.3Multiple access protocols

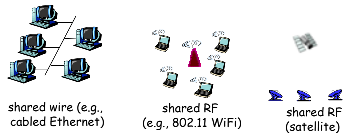
<u>LAN technology</u>
- 5.5 Ethernet
- 5.6 Interconnection
- 5.4 Link-Layer Addressing
- 5.7 PPP
- 5.9 A day in the life of a web request

(5.8 Link Virtualization: ATM and MPLS)



# Multiple Access Links and Protocols

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)
  - Ethernet
  - upstream HFC
  - 802.11 wireless LAN







# 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

*<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!
    - no out-of-band channel for coordination



# Ideal Mulitple Access Protocol

<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
4. Simple



# MAC Protocols: a taxonomy

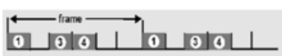
Three broad classes:
- Channel Partitioning
  - divide channel into smaller "pieces" (time slots, frequency); allocate piece to node for exclusive use
- Random Access
  - allow collisions; "recover" from collisions
- "Taking turns"
  - tightly coordinate shared access to avoid collisions

Recall goal: efficient, fair, simple, decentralized

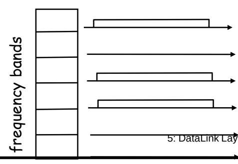


# Channel Partitioning MAC protocols: TDMA, FDMA

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




FDMA: frequency division multiple access
- each station assigned fixed frequency band
- unused transmission time in frequency bands goes idle
  - example: 6-station LAN, 1,3,4 have pkt, frequency bands 2,5,6 idle



## Channel Partitioning CDMA

CDMA: Code Division Multiple Access

- allows each station to transmit over the entire frequency spectrum all the time.
- simultaneous transmissions are separated using coding theory.
- used mostly in wireless broadcast channels (cellular, satellite, etc) - we will study it in the wireless context
- has been "traditionally" used in the military

Observe:

**MUX** = speak person-to-person in designated space

**CDMA** = "shout" using different languages: the ones who know the language will get what you say



## 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

Assumptions:

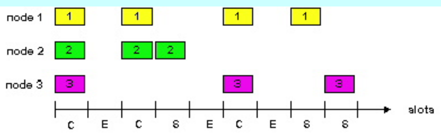
- all frames same size
- time divided into equal size slots (time to transmit 1 frame)
- nodes start to transmit only at slot beginning
- nodes are synchronized
- if 2 or more nodes transmit in slot, all nodes detect collision

Operation:

- when node obtains fresh frame, transmits in next slot
  - *if 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




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** : long-run fraction of successful slots (many nodes, all with many frames to send)

Q: max fraction of successful transmissions?

A: Suppose N stations, each transmits in slot with probability $p$

- prob. successful transmission is:

P[specific node succeeds]= $p\,(1-p)^{(N-1)}$

P[any of N nodes succeeds] $= N\,p\,(1-p)^{(N-1)}$

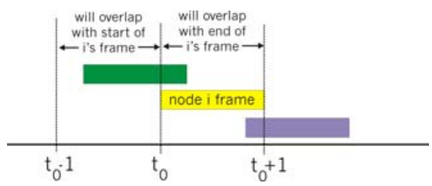
Efficiency $= 1/e = .37$ *LARGE N*

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



## Pure (unslotted) ALOHA

- unslotted Aloha: simpler, no synchronization
- pkt needs transmission:
  - send without awaiting for beginning of slot
- collision probability increases:
  - pkt sent at $t_0$ collide with other pkts sent in $[t_0-1, t_0+1]$



## Pure Aloha (cont.)

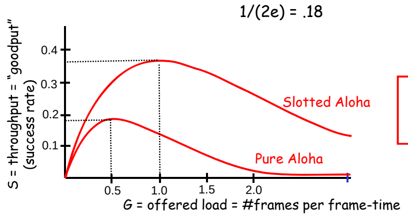
P(success by any of N nodes) = N p · $(1-p)^{2N}$ =

i.e. N p P(no other node transmits in [p0-1,p0] .

P(no other node transmits in [p0,p0+1]

=(as n -> infty ...)

1/(2e) = .18




*protocol* constrains effective channel throughput!



## CSMA: Carrier Sense Multiple Access

CSMA: listen before transmit:

- If channel sensed busy, defer transmission
  - back-off, random interval
- If/when channel sensed idle:
  - p-persistent CSMA: transmit immediately with probability p; with probablility 1-p retry after random interval
  - non-persistent CSMA: transmit after random interval

human analogy: don't interrupt others!



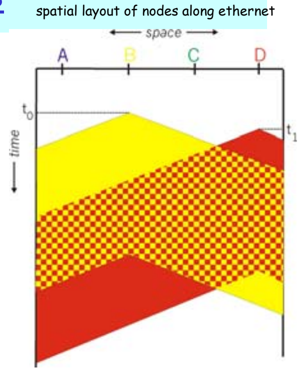
## CSMA collisions

spatial layout of nodes along ethernet

collisions *can* occur:
Due to propagation delay, two nodes may not hear each other's transmission

collision:
entire packet transmission time wasted

note:
role of distance and propagation delay (d)in determining collision
(collision-detection delay = 2d)






## CSMA/CD (Collision Detection)

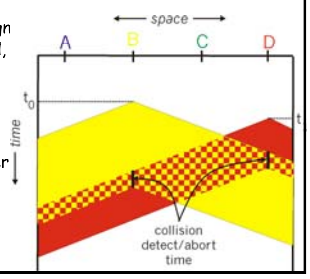
CSMA/CD: carrier sensing, deferral as in CSMA

- colliding transmissions aborted, reducing channel wastage
- persistent or non-persistent retransmission

collision detection:

- easy in wired LANs: measure sign strengths, compare transmitted, received signals
- different in wireless LANs: transmitter/receiver not "on" simultaneously; collision at the receiver matters, not the sender

human analogy: the polite conversationalist






## "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!

Random access MAC protocols

- efficient at low load: single node can fully utilize channel
- high load: collision overhead

"taking turns" protocols

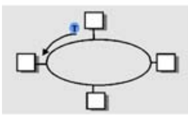
look for best of both worlds!



## "Taking Turns" MAC protocols

Token passing:

- control **token-frame** passed from one node to next sequentially.
- not pure broadcast
- concerns:
  - token overhead
  - latency
  - single point of failure (token)



## IEEE 802.4 Standard (General Motors Token Bus)
(not in must-study material)

**Contention systems limitation**: worst-case delay until successful transmission is unlimited => not suitable for real-time traffic

**Solution**: token-passing, round robin

- *token* = special control frame; only the holding station can transmit; then it passes it to another station, i.e. for token bus, the next in the logical ring
- 4 priority classes of traffic, using timers
- Logical ring-maintenance: distributed strategy
  - Robust, somehow complicated though



## IEEE Standard 802.5 (Token Ring)
(not in must-study material)

**Motivation**: instead of complicated token-bus, have a physical ring

Principle: Each bit arriving at an interface is copied into a 1-bit buffer (inspected and/or modified); then copied out to the ring again.

- copying step introduces a 1-bit delay at each interface.

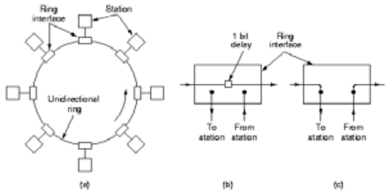



Fig. 4-28. (a) A ring network. (b) Listen mode. (c) Transmit mode.



## Token Ring operation

- to transmit a frame, a station is required to seize the token and remove it from the ring before transmitting.
- bits that have propagated around the ring are removed from the ring by the sender (the receiver in FDDI).
- After a station has finished transmitting the last bit of its frame, it must regenerate the token.



## IEEE 802.5 Ring: Maintenance
(not in must-study material)

**Centralised**: a "monitor" station oversees the ring:

- generates token when lost
- cleans the ring when garbled/orphan frames appear

**If** the monitor goes away, a convention protocol ensures that another station is *elected* as a monitor (e.g. the one with highest identity)

**If** the monitor gets "mad", though.....



## IEEE 802.5 Ring: Priority Algorithm
(not in must-study material)

Station S

**upon arrival of frame f:**

set prior(f) := max{prior(f), prior(S)}

forward(f)

**upon arrival of T**

if prior(T)>prior(S) then forward(T)

else send own frame f with prior(f):=0

wait until f comes back

prior(T):=prior(f)

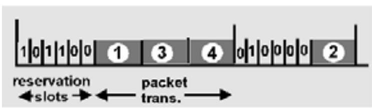
forward(T)



## Reservation-based protocols

Distributed Polling – Bit-map protocol:

- time divided into slots
- begins with N short reservation slots
  - station with message to send posts reservation during its slot
  - reservation seen by all stations
  - reservation slot time equal to channel end-end propagation delay (why?)
- after reservation slots, message transmissions ordered by known priority



## 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, token passing
    - Bluetooth, FDDI, IBM Token Ring



## Link Layer

- 5.1 Introduction and services
- Framing
- 5.2 Error detection and correction
- 5.3Multiple access protocols

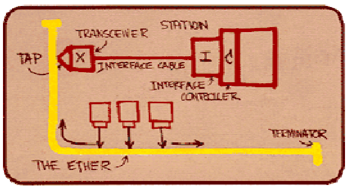
LAN technology
- 5.5 Ethernet
- 5.6 Interconnection
- 5.4 Link-Layer Addressing
- 5.7 PPP
- 5.9 A day in the life of a web request

(5.8 Link Virtualization: ATM and MPLS)



## 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



## Ethernet: uses CSMA/CD

**A**: sense channel, **if** idle
**then** {
transmit and monitor the channel;
**If** detect another transmission
**then** {
abort and send jam signal;
update # collisions;
delay as required by exponential backoff algorithm;
goto A
}
**else** {done with the frame; set collisions to zero}
}
**else** {wait until ongoing transmission is over and goto A}



## Ethernet's CSMA/CD (more)

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

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 x frame-transmission time)
- after second collision: choose K from {0,1,2,3}…
- after ten or more collisions, choose K from {0,1,2,3,4,…,1023}



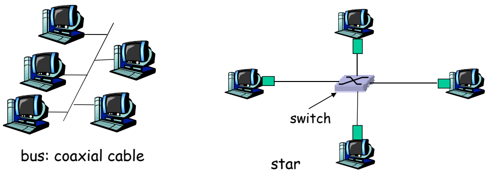
## Ethernet (CSMA/CD) Limitation

- Recall: collision detection interval = 2*Propagation delay along the LAN
- This implies a minimum frame size and/or a maximum wire length

*Critical factor*:
*a = 2 * propagation_delay /frame_transmission_delay*

## Star topology

- bus topology popular through mid 90s
  - all nodes in same collision domain (can collide with each other)
- today: star topology prevails (more bps, shorter distances)
  - Hub or active *switch* in center
  - (more in a while)






## CSMA/CD efficiency

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

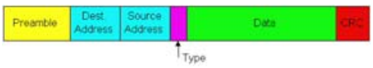
$$\text{efficiency} = \frac{1}{1 + 5t_{prop}/t_{trans}}$$

- Much better than ALOHA, but still decentralized, simple, and cheap



## Ethernet Frame Structure

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
- to synchronize receiver, sender clock rates

Addresses: 6 bytes, frame is received by all adapters on a LAN and dropped if address does not match

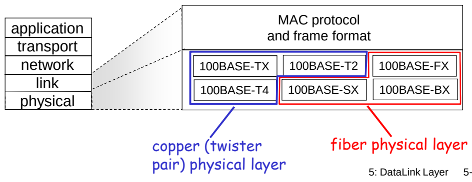
Type: indicates the higher layer protocol, mostly IP but others may be supported (such as Novell IPX and AppleTalk)

CRC: checked at receiver, if error is detected, the frame is simply dropped



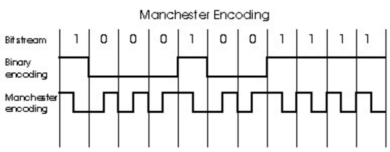
## 802.3 Ethernet Standards: Link & Physical Layers

- *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!



## Ethernet: Unreliable, connectionless

- connectionless: No handshaking between sending and receiving NICs
- unreliable: receiving NIC doesn't send acks or nacks to sending NIC
  - stream of datagrams passed to network layer can have gaps (missing datagrams)
  - gaps will be filled if app is using TCP
  - otherwise, app will see gaps

## Link Layer

- 5.1 Introduction and services
- Framing
- 5.2 Error detection and correction
- 5.3Multiple access protocols

LAN technology

- 5.5 Ethernet
- 5.6 Interconnection
- 5.4 Link-Layer Addressing
- 5.7 PPP
- 5.9 A day in the life of a web request

(5.8 Link Virtualization: ATM and MPLS)

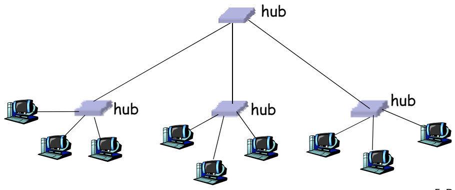


## Interconnecting with hubs

Hubs are essentially physical-layer repeaters:

- bits coming from one link go out all other links
- at the same rate (no frame buffering)

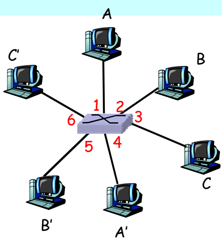
- no CSMA/CD at hub: adapters detect collisions (one large collision domain)
- provides net management functionality (monitoring, statistics)
- Extends distance between nodes
- Can't interconnect e.g. 10BaseT & 100BaseT






## Switch: allows *multiple* simultaneous transmissions

- hosts may 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)

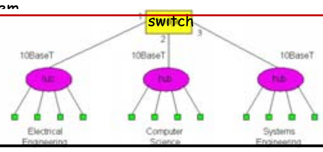


## Switches (bridges): cont.

- Link Layer devices: operate on frames, examining header and selectively forwarding frame based on its destination
  - filtering: same-LAN-segment frames not forwarded to other seg's
- Advantages:
  - Isolates collision domains:
    - higher total max throughput
    - no limit on number of nodes nor distances
  - Can connect different net-types (translational, …)
  - Transparent: no need for any change to hosts LAN adapters

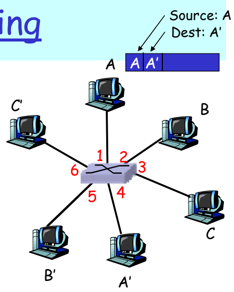
forwarding: how to know LAN segment on which to forward frame?

- looks like a routing probl...




## 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

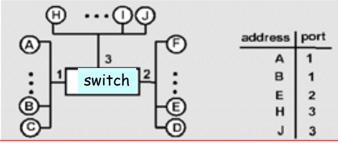
When frame received:

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

## Switch Learning: example

Suppose C sends a frame to D and D replies with a frame to C

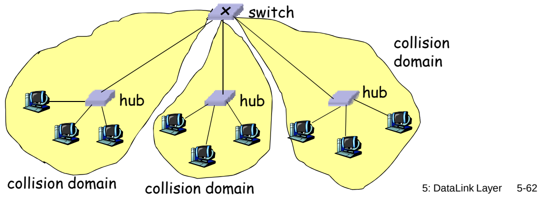



- C sends frame, switch has no info about D, so floods
  - switch notes that C is on port 1
  - frame ignored on upper LAN
  - frame received by D
- D generates reply to C, sends
  - switch sees frame from D
  - switch notes that D is on interface 2
  - switch knows C on interface 1, so *selectively* forwards frame out via interface 1



## Switch: traffic isolation

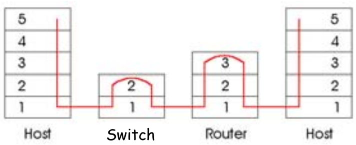
- switch installation breaks subnet into LAN segments
- switch filters packets:
  - same-LAN-segment frames not usually forwarded onto other LAN segments
  - segments become separate collision domains






## Switches vs. Routers

- both store-and-forward devices
  - routers: network layer devices (examine network layer headers)
  - Switches (bridges) are Link Layer devices
- routers maintain routing tables, implement routing algorithms
- switches maintain filtering tables, implement filtering, learning (and spanning tree) algorithms






## Routers vs. Bridges/Switches

Bridges/Switches

+ Bridge operation is simpler requiring less processing bandwidth
- Topologies are restricted with bridges (a spanning tree must be built to avoid cycle)
- Bridges do not offer protection from broadcast storms (endless broadcasting by a host will be forwarded by a bridge)

Routers

+ arbitrary topologies can be supported, cycling is limited by good routing protocols
+ provide firewall protection against broadcast storms
- require detailed configuration (not plug and play) and higher processing capacity

Bridges/switches do well in small (few hundred hosts) while routers used in large networks (thousands of hosts)

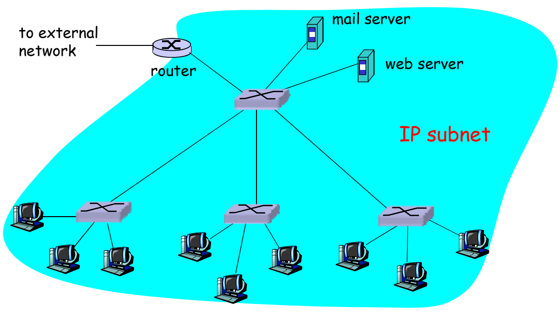


## Summary comparison

| | hubs | routers | switches |
|---|---|---|---|
| traffic isolation | no | yes | yes |
| plug & play | yes | no | yes |
| optimal routing | no | yes | no |

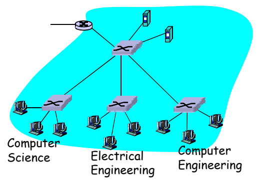


## Institutional network



## VLANs: motivation

What's "wrong" with this picture?



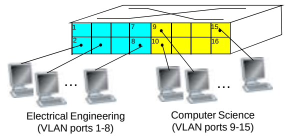

What happens if:

- 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

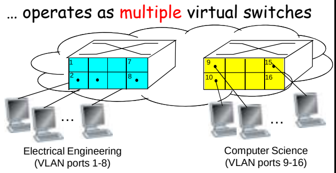


## VLANs

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




… operates as multiple virtual switches



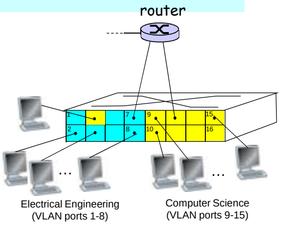

**Virtual Local Area Network**

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



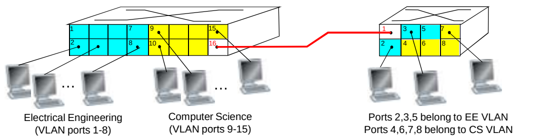
## Port-based VLAN:

- *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





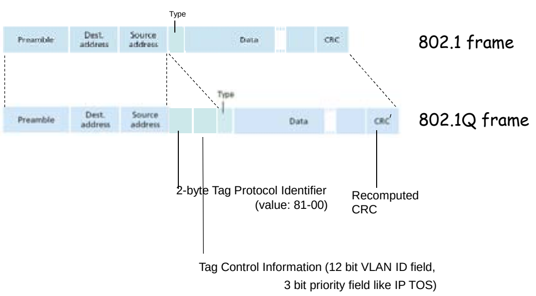

## VLANS spanning multiple switches




- *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

- 5.1 Introduction and services
- Framing
- 5.2 Error detection and correction
- 5.3Multiple access protocols

LAN technology

- 5.5 Ethernet
- 5.6 Interconnection
- 5.4 Link-Layer Addressing
- 5.7 PPP
- 5.9 A day in the life of a web request

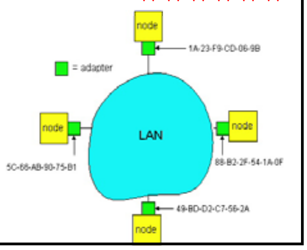
(5.8 Link Virtualization: ATM and MPLS)

# LAN Addresses

32-bit IP address:
- *network-layer* address
- used to get datagram to destination network (recall IP network definition)

LAN (or MAC or physical) address:
- to get datagram from one interface to another physically-connected interface (same network)
- 48 bit MAC address (for most LANs) burned in NIC's ROM (sometimes resettable)

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




# LAN Address (more)

- MAC address allocation administered by IEEE
- manufacturer buys portion of MAC address space (to assure uniqueness)

Analogy:

(a) MAC address: like People's Names or PersonalNum's

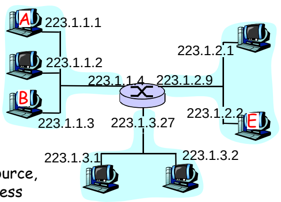
(b) IP address: like postal address

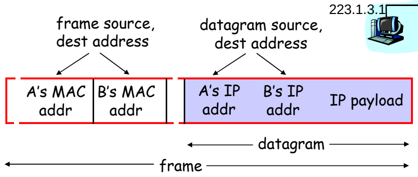
- MAC flat address => portability
  - can move LAN card from one LAN to another
- IP hierarchical address NOT portable
  - depends on network to which one attaches



# Recall earlier routing discussion

Starting at A, given IP datagram addressed to B:
- look up net. address of B, find B on same net. as A
- link layer send datagram to B inside link-layer frame



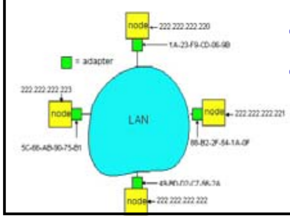






# ARP: Address Resolution Protocol

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

- Each IP node (Host, Router) on LAN has ARP module, table
  - ARP Table: IP/MAC address mappings
    - < IP address; MAC address; TTL>
    - < ................................ >
    - TTL (Time To Live): time to cache (typically 20 min); afterwards:
  - A broadcasts ARP query pkt, containing B's IP address
  - B receives ARP packet, replies to A with its (B's) physical layer address
  - A caches (saves) IP-to-physical address pairs until they times out
    - soft state: information that times out (goes away) unless refreshed

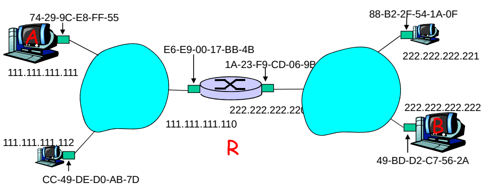





# Addressing: routing to another LAN

walkthrough: send datagram from A to B via R

assume A knows B's IP address

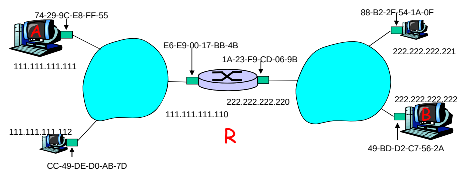



- two ARP tables in router R, one for each IP network (LAN)



- A creates IP 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 NIC sends frame
- R's NIC 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!



## Link Layer

- 5.1 Introduction and services
- Framing
- 5.2 Error detection and correction
- 5.3Multiple access protocols

LAN technology
- 5.5 Ethernet
- 5.6 Interconnection
- 5.4 Link-Layer Addressing
- 5.7 PPP
- 5.9 A day in the life of a web request

(5.8 Link Virtualization: ATM and MPLS)



## 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



## 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)
- bit transparency: no constraints on 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

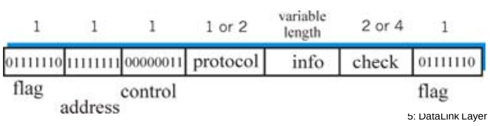
- no error correction/recovery
- no flow control
- "out of order" delivery OK

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



## PPP Data Frame

- Flag: delimiter (framing; hence "stuffing" in payload)
- 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)
- info: upper layer data being carried
- check: cyclic redundancy check for error detection

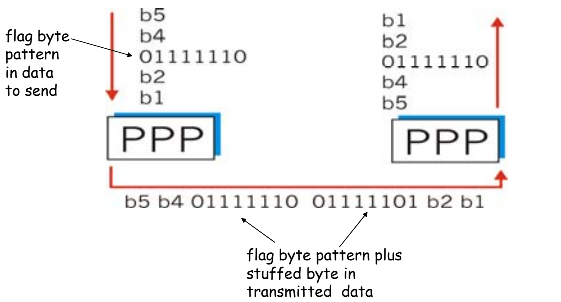





## Framing method: Byte Stuffing

- "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

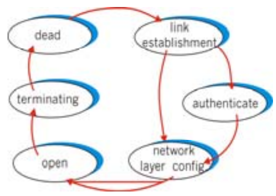





## PPP Data Control Protocol

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

- 5.1 Introduction and services
- Framing
- 5.2 Error detection and correction
- 5.3Multiple access protocols

LAN technology

- 5.5 Ethernet
- 5.6 Interconnection
- 5.4 Link-Layer Addressing
- 5.7 PPP
- 5.9 A day in the life of a web request

(5.8 Link Virtualization: ATM and MPLS)

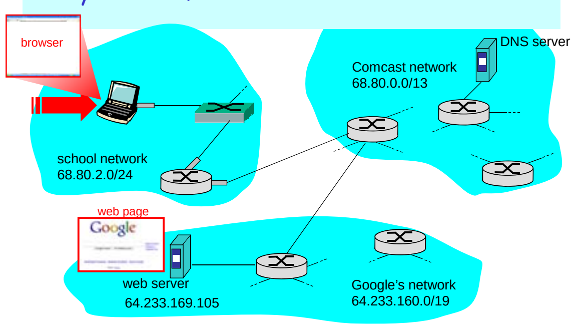


## Synthesis: a day in the life of a web request

- 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

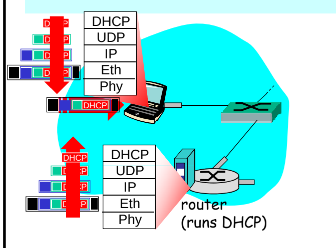


## A day in the life: scenario



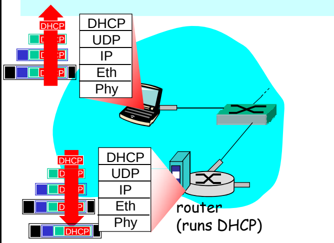



## A day in the life… connecting to the Internet




- 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

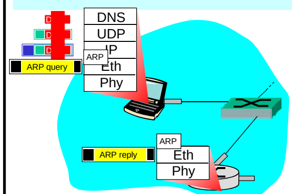



- DHCP server formulates *DHCP ACK* containing client's IP address, IP address of first-hop router for client, name & IP address of DNS server
- encapsulation at DHCP server, frame forwarded (**switch learning**) through LAN, demultiplexing at client
- DHCP client receives DHCP ACK reply

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



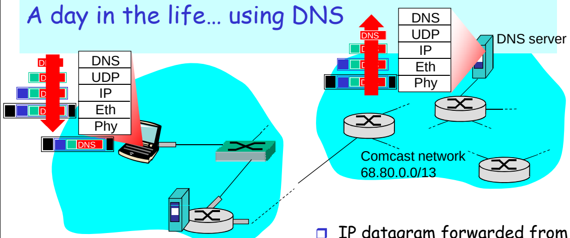
## A day in the life… ARP (before DNS, before HTTP)




- 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



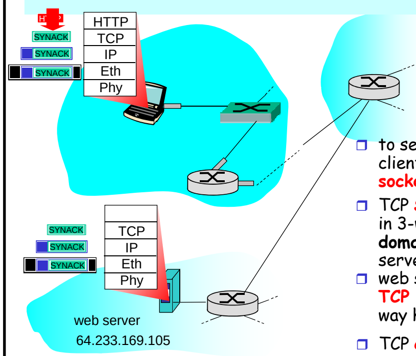
## A day in the life… using DNS




- 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** and **BGP** routing protocols) to DNS server
- demux'ed to DNS server
- DNS server replies to client with IP address of www.google.com



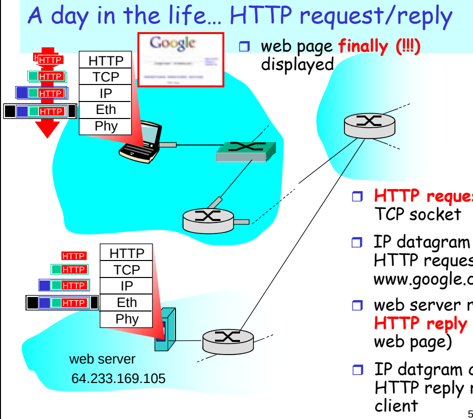
## A day in the life… TCP connection carrying HTTP




- 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



## Link Layer

- 5.1 Introduction and services
- 5.2 Error detection and correction
- 5.3Multiple access protocols
- 5.4 Link-Layer Addressing
- 5.5 Ethernet
- 5.6 Hubs and switches
- 5.7 PPP
- A day in the lifetime of a web-request
- 5.8 Link Virtualization

## Link Layer

- 5.1 Introduction and services
- Framing
- 5.2 Error detection and correction
- 5.3Multiple access protocols

LAN technology

- 5.5 Ethernet
- 5.6 Interconnection
- 5.4 Link-Layer Addressing
- 5.7 PPP
- 5.9 A day in the life of a web request

(5.8 Link Virtualization: ATM and MPLS)



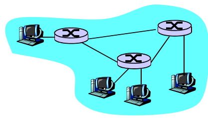
## The Internet: virtualizing networks

1974: multiple unconnected nets

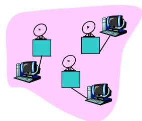
- ARPAnet
- data-over-cable networks
- packet satellite network (Aloha)
- packet radio network

… differing in:

- addressing conventions
- packet formats
- error recovery
- routing



ARPAnet



satellite net

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



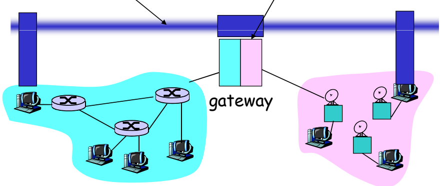
## The Internet: virtualizing networks

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



## On ATM: Asynchronous Transfer Mode nets

- 1980's telco's proposal for future networking
- Smart core, simple terminals

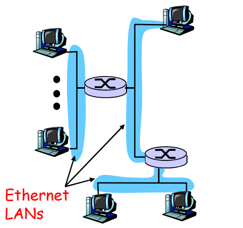
- small (48 byte payload, 5 byte header) fixed length *cells* (like packets)
  - fast switching (pipelined/cut-through)
  - small size good for voice
- virtual-circuit network: switches maintain state for each "call"

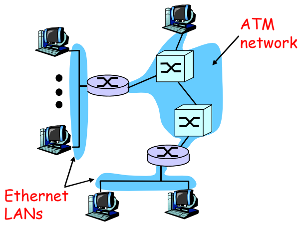
- well-defined interface between "network" and "user" (think of telephone company):
  - several transport (Adaptation)-layer protocols, one per expected type of traffic

## IP-Over-ATM

**Classic IP only**
- 3 "networks" (e.g., LAN segments)
- MAC (802.3) and IP addresses

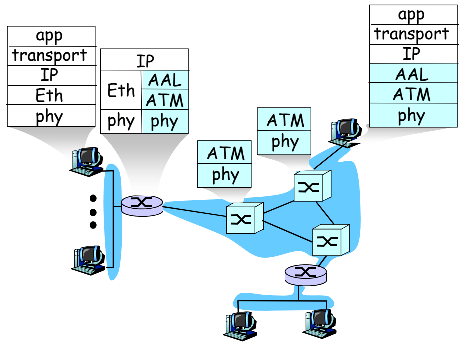
**IP over ATM**
- replace "network" (e.g., LAN segment) with ATM network
- ATM addresses, IP addresses






## IP-Over-ATM





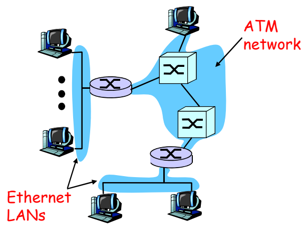

## Datagram Journey in IP-over-ATM Network

- at Source Host:
  - IP layer maps between IP, ATM dest address (using ARP)
  - passes datagram to AAL5
  - AAL5 encapsulates data, segments cells, passes to ATM layer
- ATM network: moves cell along VC to destination
- at Destination Host:
  - AAL5 reassembles cells into original datagram
  - if CRC OK, datagram is passed to IP



## IP-Over-ATM

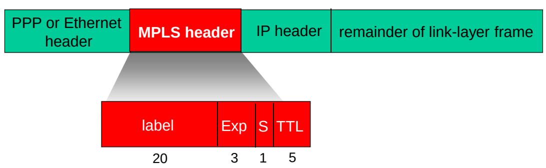
**Issues:**
- IP datagrams into ATM AAL5 PDUs
- from IP addresses to ATM addresses
  - just like IP addresses to 802.3 MAC addresses!






## 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!





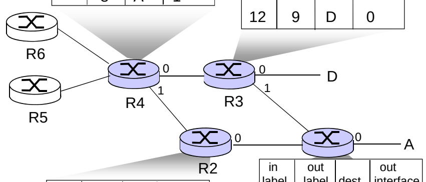

## MPLS capable routers

- 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

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

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




| in label | out label | dest | out interface |
|---|---|---|---|
| 8 | 6 | A | 0 |

| in label | out label | dest | out interface |
|---|---|---|---|
| 6 | - | A | 0 |



# Chapter 5: 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
  - PPP
  - Link Virtualization: ATM and MPLS