THE UNIVERSITY OF ADELAIDE AUSTRALIA

COMPUTER NETWORKS AND APPLICATIONS

COMP SCI 3001

Faculty of Engineering, Computer and Mathematical Sciences

LAN Addressing

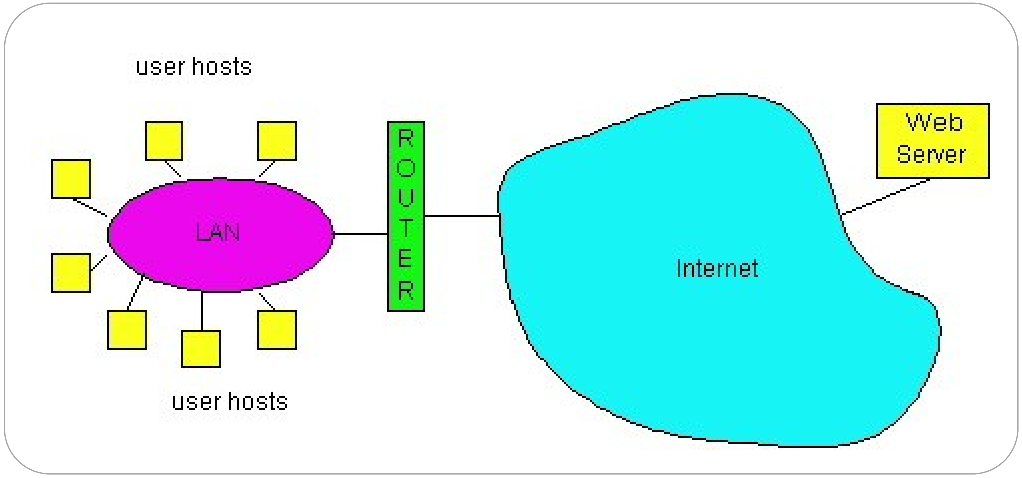
LAN technologies

Data Link Layer so far

- **Services, error detection/correction, multiple access**

Next: LAN technologies

- Addressing
- Ethernet
- Switches
- MPLS



LAN addresses and ARP

We know 32 bit IP addresses

- Network Layer address
  - Used to get datagram **to destination network** (recall IP network definition)

LAN (or MAC (media access control) or physical) address

- *used 'locally" to get frame from one interface to another physically-connected interface (same network, in IP-addressing sense)*
- At LAN level:
  - 48 bit MAC address (for most LANs) burned in the adapter ROM
  - e.g.: 1A-2F-BB-76-09-AD

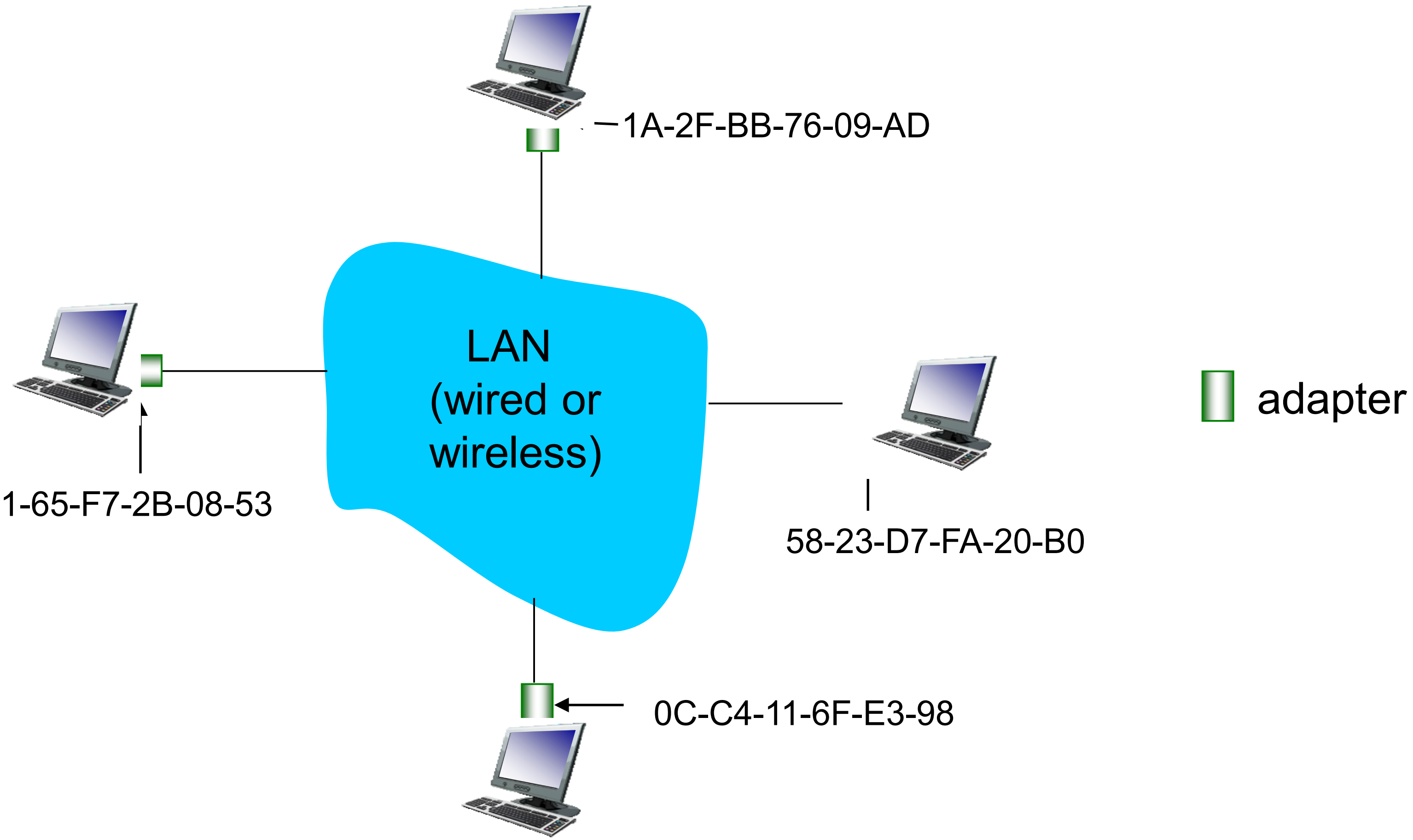
hexadecimal (base 16) notation
(each "number" represents 4 bits)

MAC address allocation

- MAC address allocation administered by IEEE
- Manufacturer buys portion of MAC address space (to assure uniqueness)
- Analogy
  - (a) MAC address - like Tax File 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 network to which one attaches

LAN addresses and ARP

each adapter on LAN has unique *LAN* address

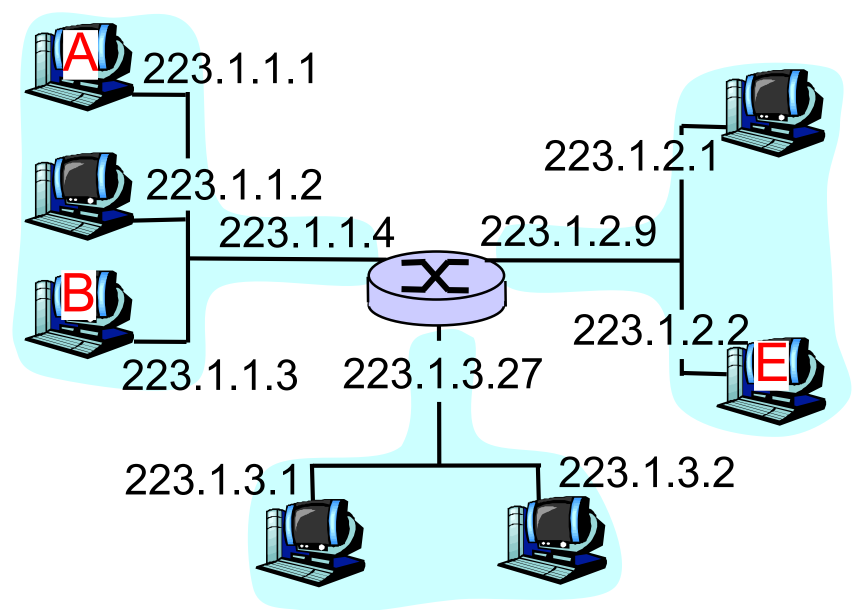


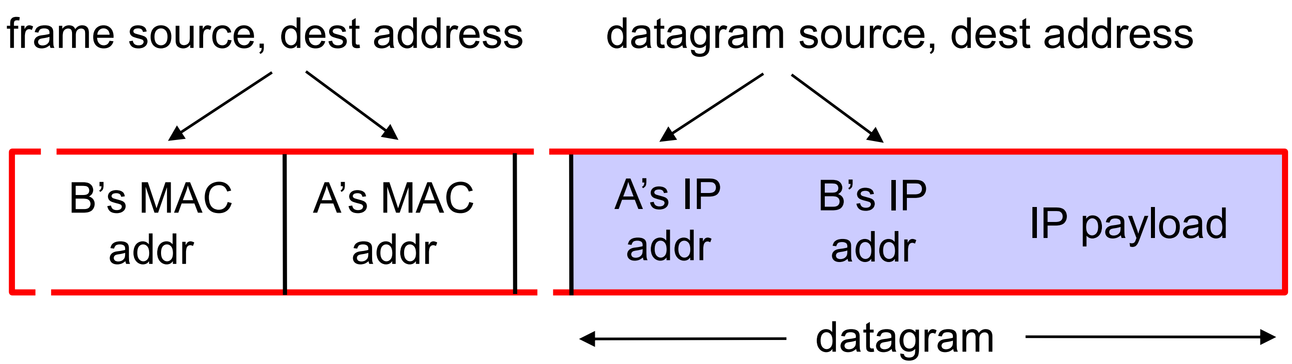
Data Link Layer introduction

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**





How do we work out the MAC address of the destination host?

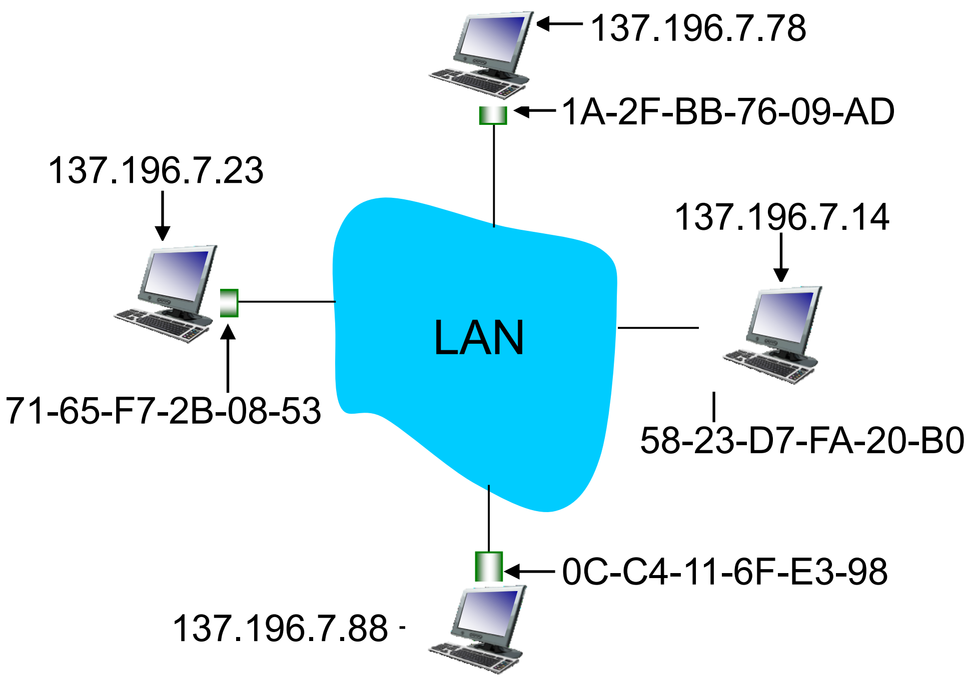
Address Resolution Protocol (ARP)

- Each IP node (host, router) on LAN has **ARP** module and **table**
- ARP Table: IP/MAC address mappings for some LAN nodes

< IP address; MAC address; TTL>

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

- Why?



ARP protocol: same LAN

1. A wants to send datagram to B
   - B's MAC address not in A's ARP table.
2. A broadcasts ARP query packet, containing B's IP address
   - dest MAC address = FF-FF-FF-FF-FF-FF
   - all nodes on LAN receive ARP query
3. B receives ARP packet, replies to A with its (B's) MAC address
   - frame sent to A's MAC address (unicast)

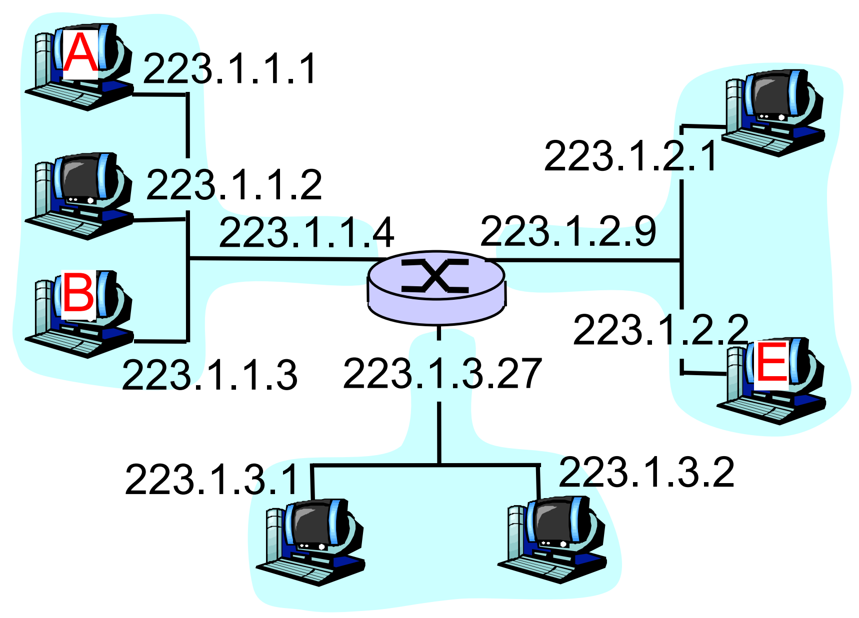
- A caches (saves) IP-to-MAC address pair in its ARP table until information becomes old (times out)
  - soft state: information that times out (goes away) unless refreshed
- ARP is "plug-and-play":
  - nodes create their ARP tables *without intervention from net administrator*

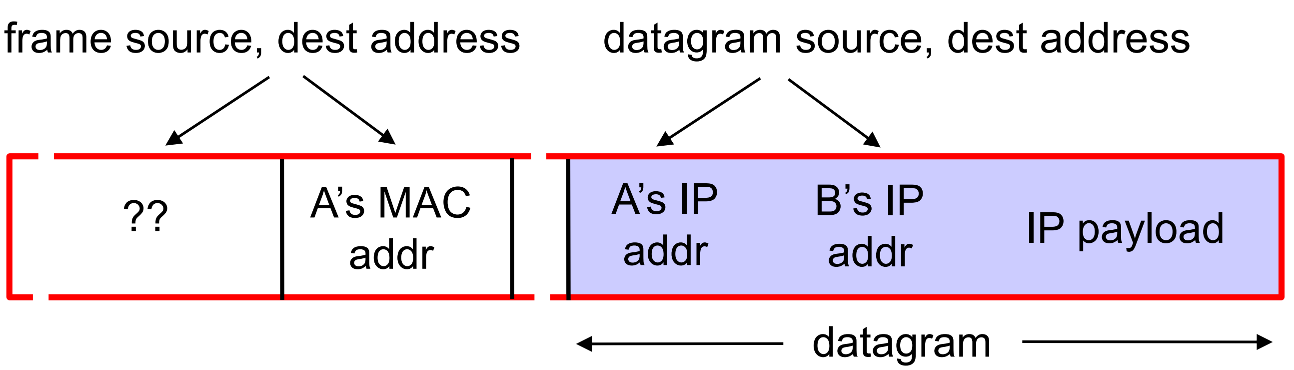
Data Link Layer introduction

Earlier routing discussion?

Starting at A, given IP datagram addressed to E:

- ??



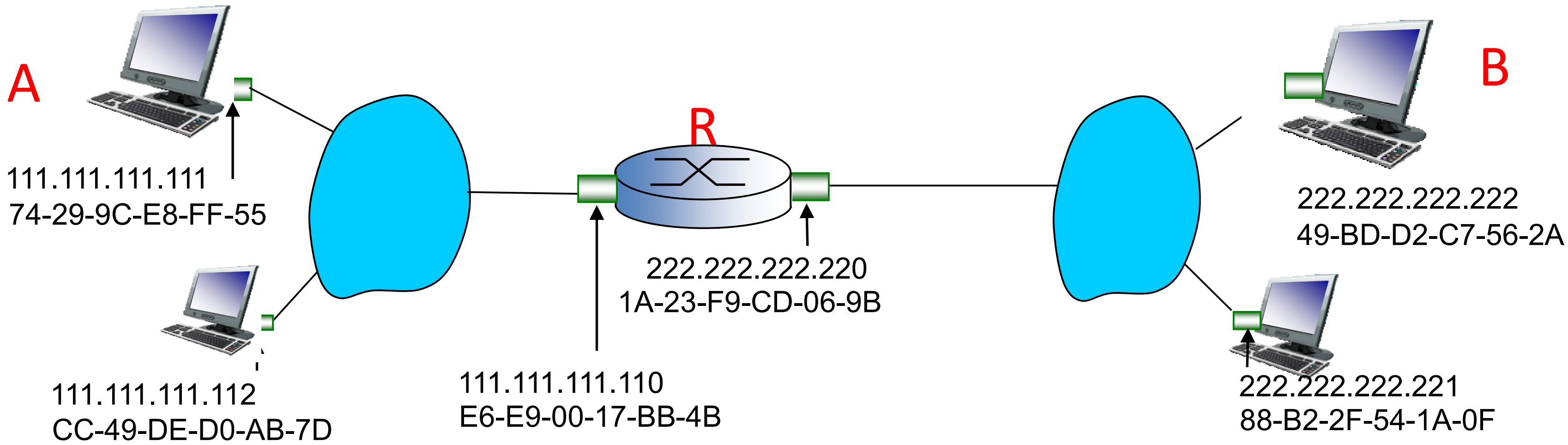


How do we work out the MAC address of the destination host?

Addressing: routing to another LAN

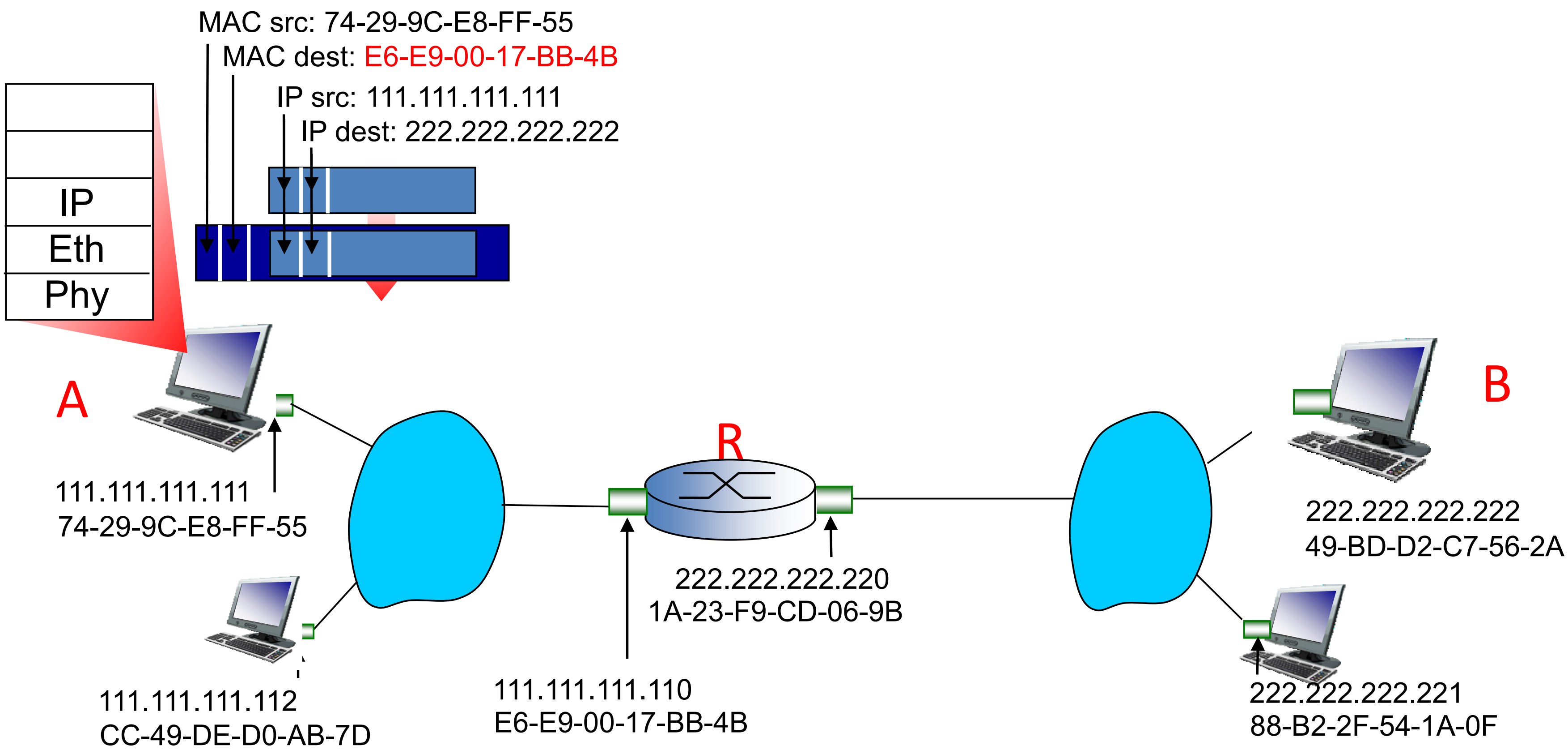
Example: send datagram from A to B via R

- focus on addressing – at IP (datagram) and MAC layer (frame)
- assume A knows B's IP address
- assume A knows IP address of first hop router, R (how?)
- assume A knows R's MAC address (how?)



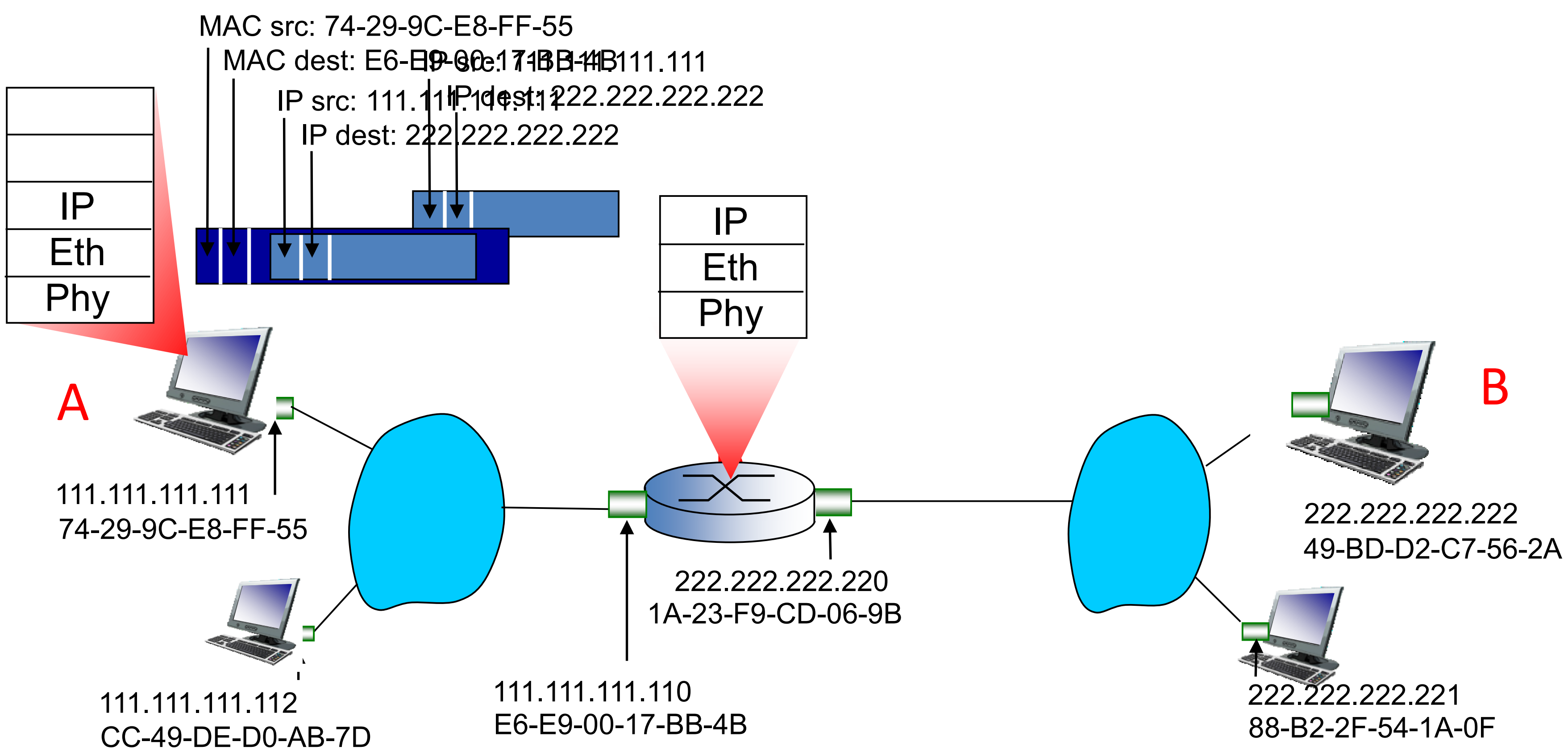
Addressing: routing to another LAN

- A creates IP datagram with IP source A, destination B
- A creates link-layer frame with R's MAC address as dest, frame contains A-to-B IP datagram



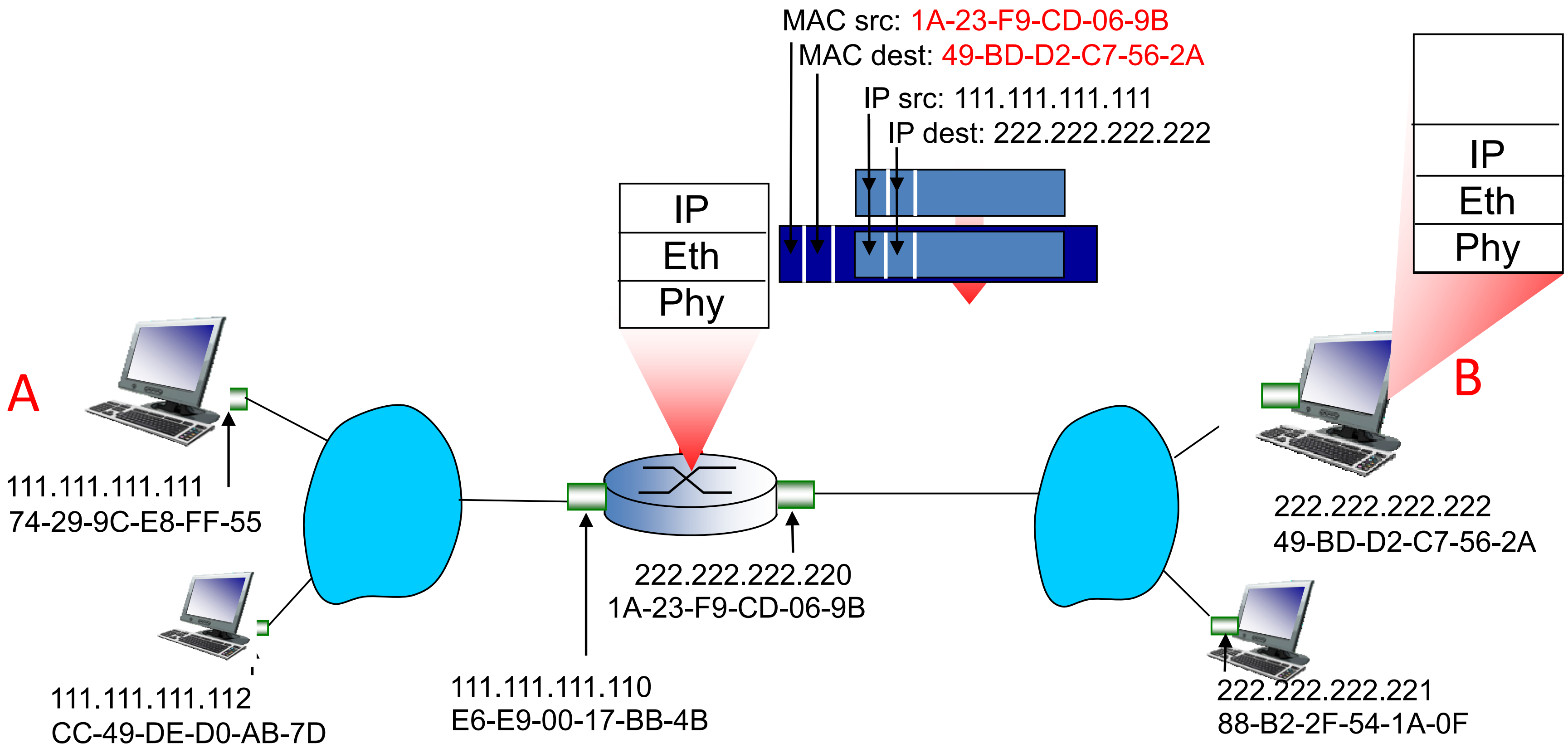
Addressing: routing to another LAN

- frame sent from A to R
- frame received at R, datagram removed, passed up to IP



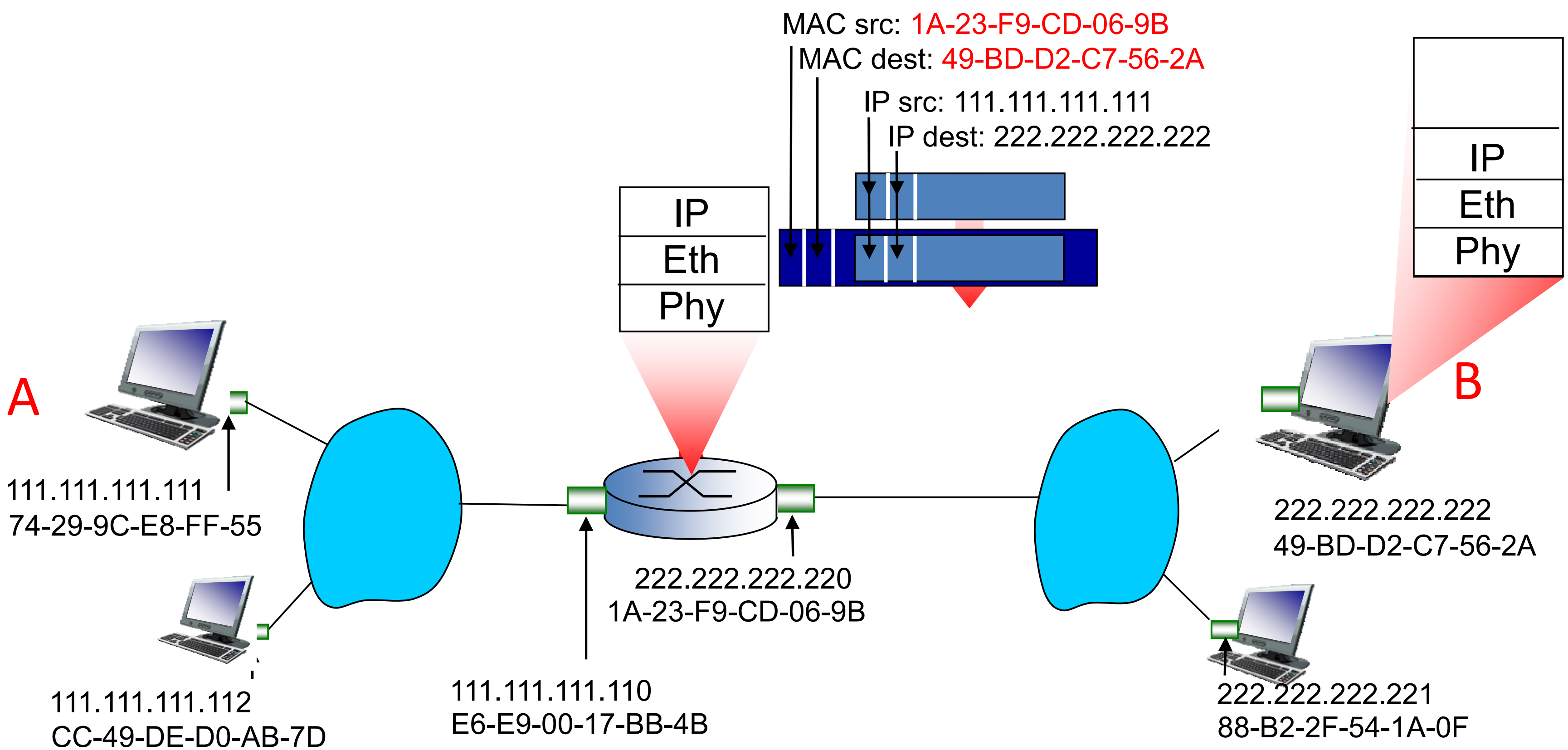
Addressing: routing to another LAN

- R forwards datagram with IP source A, destination B
- R creates link-layer frame with B's MAC address as dest, frame contains A-to-B IP datagram



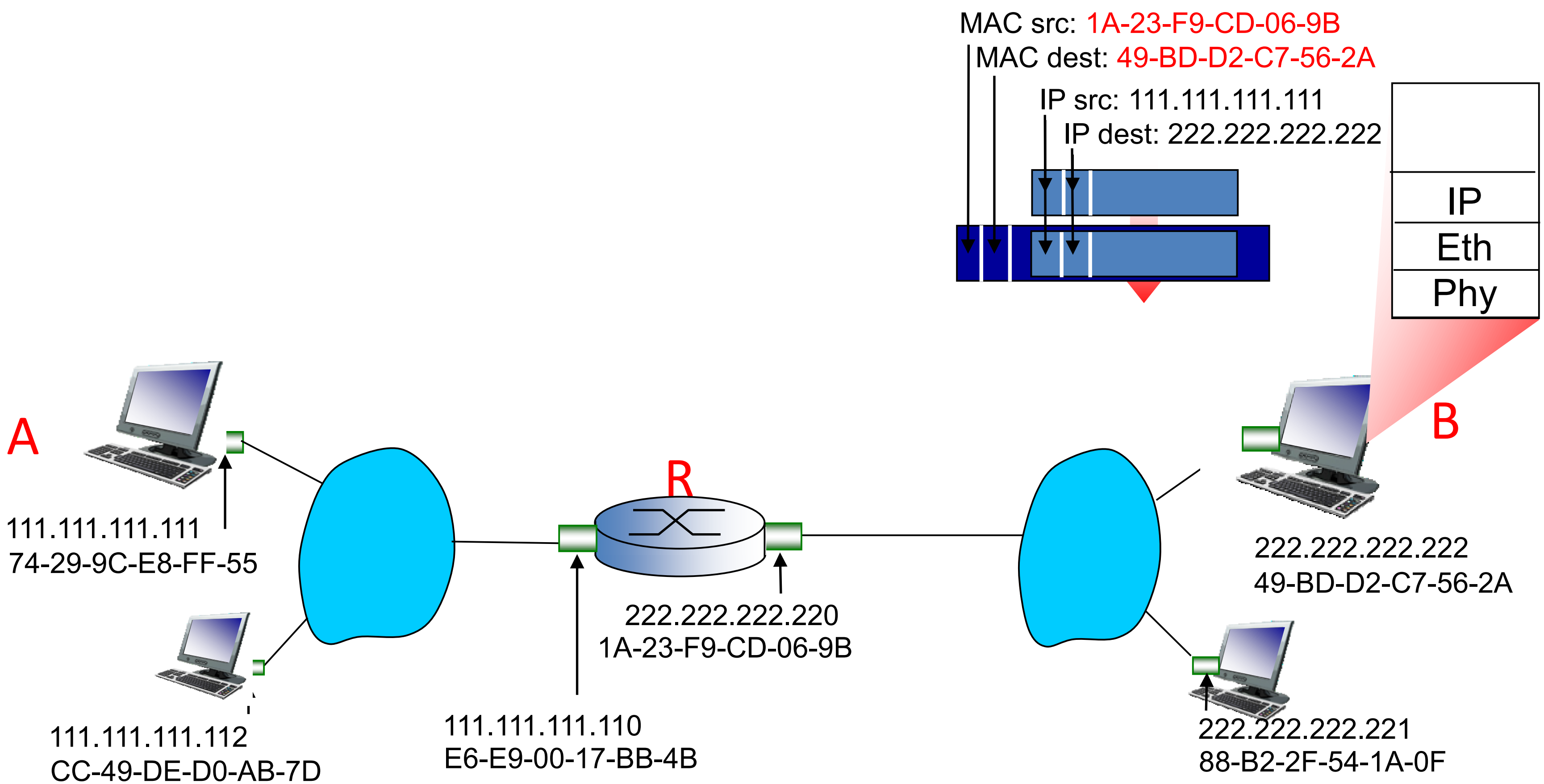
Addressing: routing to another LAN

- R forwards datagram with IP source A, destination B
- R creates link-layer frame with B's MAC address as dest, frame contains A-to-B IP datagram



Addressing: routing to another LAN

- R forwards datagram with IP source A, destination B
- R creates link-layer frame with B's MAC address as dest, frame contains A-to-B IP datagram



COMPUTER NETWORKS AND APPLICATIONS

COMP SCI 3001

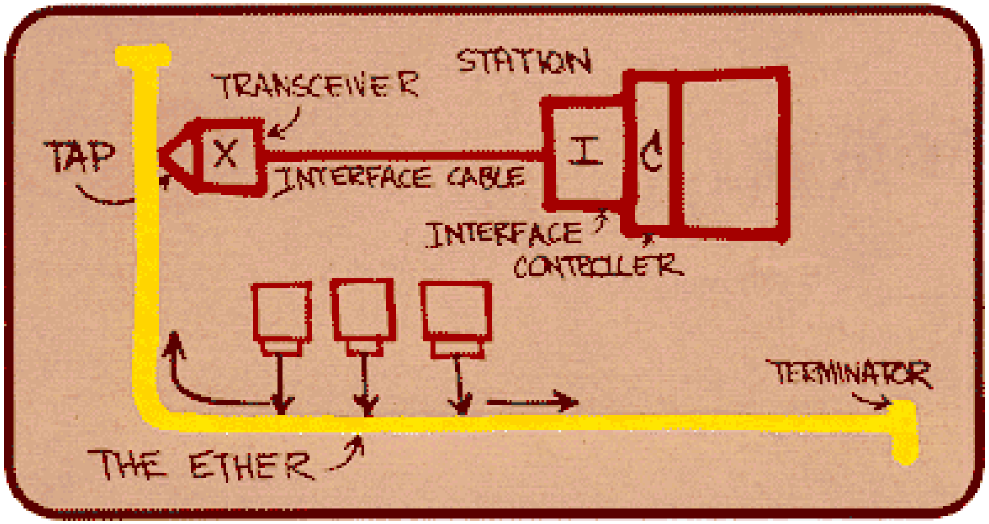
Faculty of Engineering, Computer and Mathematical Sciences

Ethernet

Ethernet

"dominant" wired LAN technology:

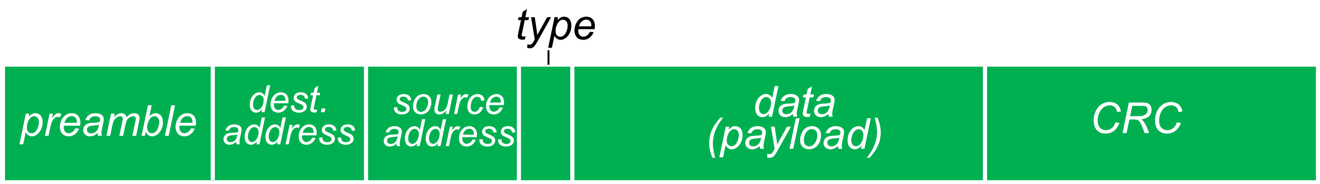
- cheap $20 for NIC
- first widely used LAN technology
- simpler, cheaper than token LANs and ATM
- kept up with speed race: 10 Mbps – 10 Gbps



Metcalfe's Ethernet sketch

Ethernet frame structure

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

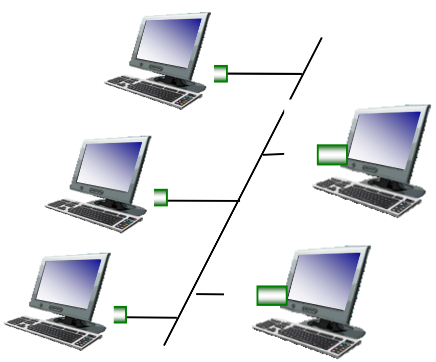


preamble:

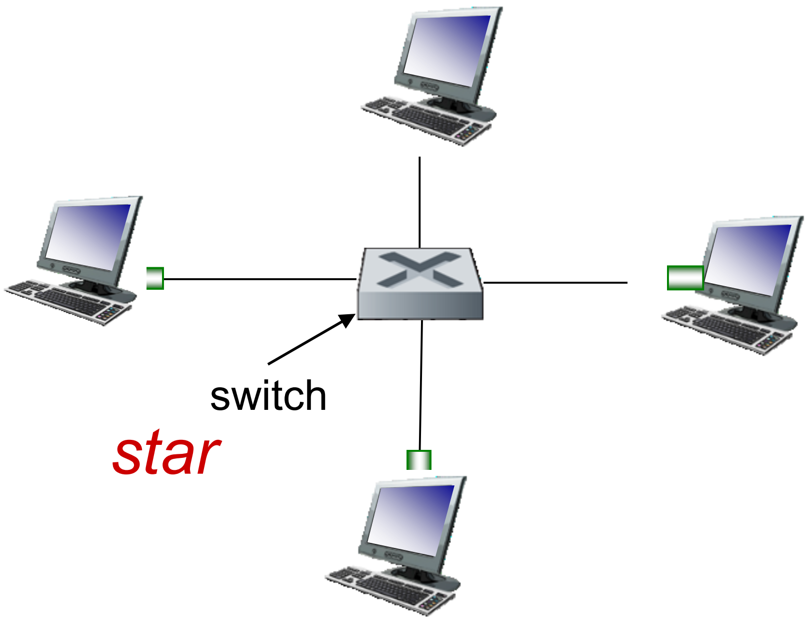
- 7 bytes with pattern 10101010 followed by one byte with pattern 10101011
- used to synchronize receiver, sender clock rates

Ethernet: physical topology

- *bus:* popular through mid 90s
  - all nodes in same collision domain (can collide with each other)
- *star:* prevails today
  - active *switch* in center
  - each "spoke" runs a (separate) Ethernet protocol (nodes do not collide with each other)



bus: coaxial cable



star

Data Link Layer introduction

- *bus:*
  - *Every one was able to share the wire because it used a media access control protocol called what? CSMA/CD*

- *star:* prevails today (**also called switched networks**)
  - active *switch* in center
  - each "spoke" runs a (separate) Ethernet protocol (nodes do not collide with each other)
  - Full duplex communication, separate wires for sending and receiving
  - So collisions domains are isolated and the CD part of CSMA/CD is not needed any more.
  - The switch is transparent => as if there is a direct wire from A to B

- NOTE original Ethernet specification are half-duplex

THE UNIVERSITY OF ADELAIDE AUSTRALIA

COMPUTER NETWORKS AND APPLICATIONS

COMP SCI 3001

Faculty of Engineering, Computer and Mathematical Sciences

Switches

Ethernet switch

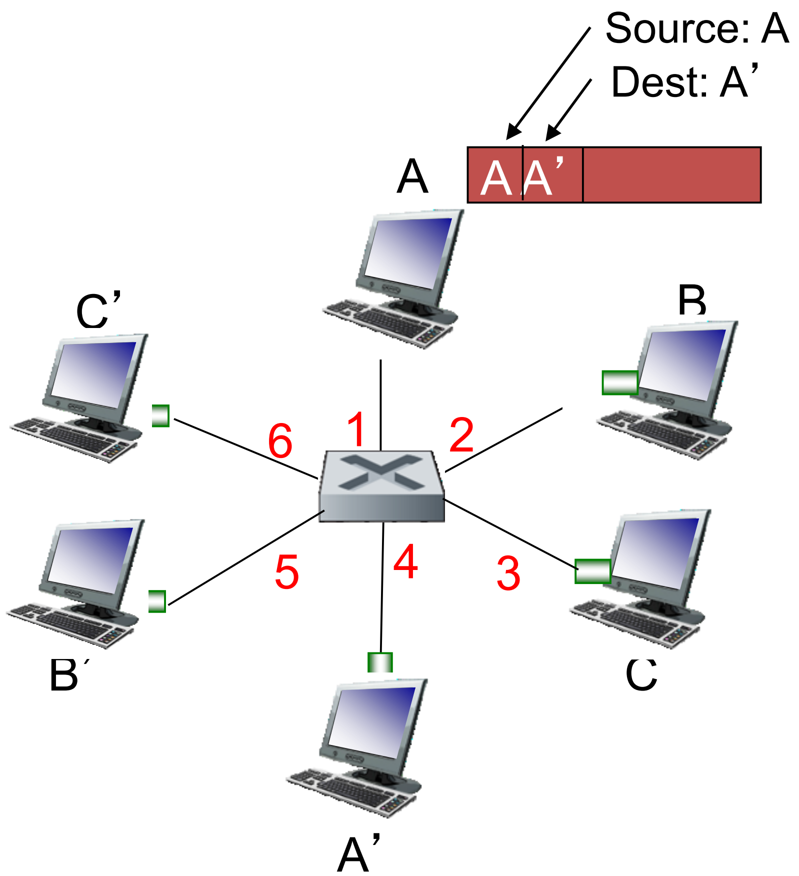
- link-layer device: takes an *active* role
  - Filtering, Storing, Forwarding Ethernet frames
  - examine incoming frame's MAC address, selectively forward frame to one-or-more outgoing links when frame is to be forwarded on segment
- *transparent*
  - hosts are unaware of presence of switches
- *plug-and-play, self-learning*
  - switches do not need to be configured

How do switches know where to send information? Switch filtering

- Switches **learn** which hosts can be reached through which interfaces: maintains ***switch table***
  - when frame received, switch **learns** location of **sender**: incoming LAN segment
  - records sender location in switch table
- Switch table entry
  - [node LAN address, switch interface (incoming), time stamp, TTL]
  - stale entries in filtering table dropped (TTL can be 60 minutes)

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)

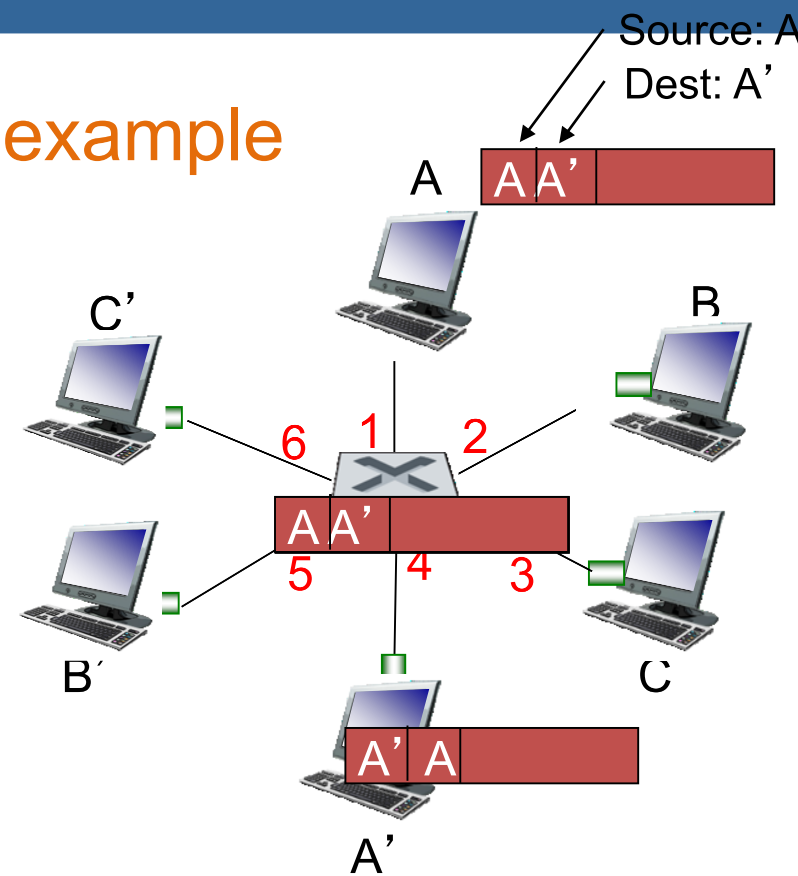
Filtering and Forwarding

```
if destination is on LAN on which frame was received
    then drop the frame
    else {
        lookup switch table
        if entry found for destination
            then 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, A', locaton unknown: *flood*

- destination A location known: selectively send on just one link

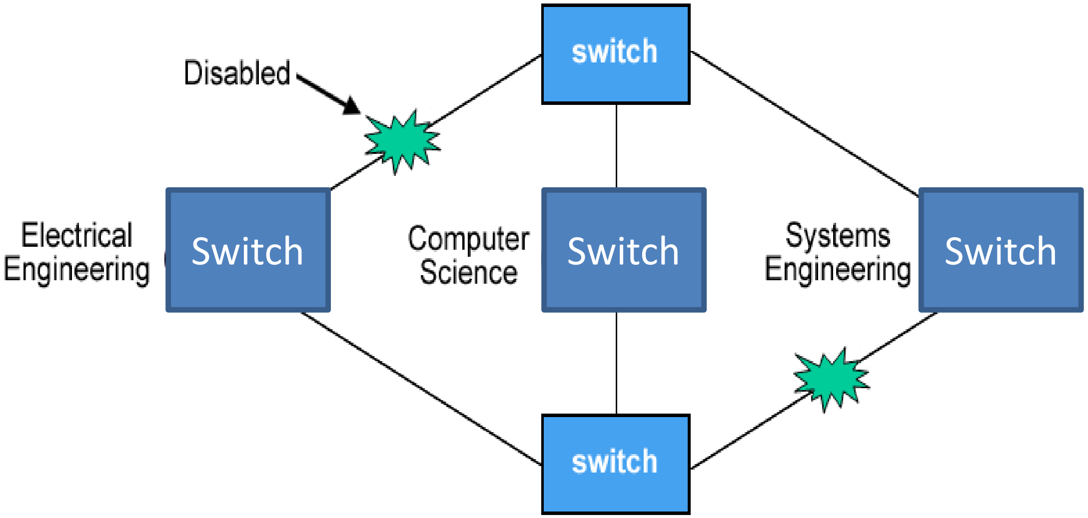


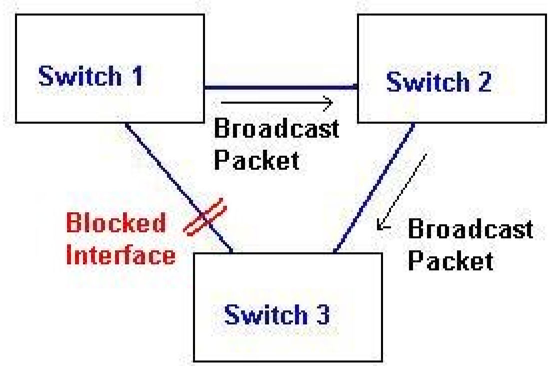
MAC addr	interface	TTL
A	*1*	*60*
A'	*4*	*60*

switch table (initially empty)

Switches - spanning trees

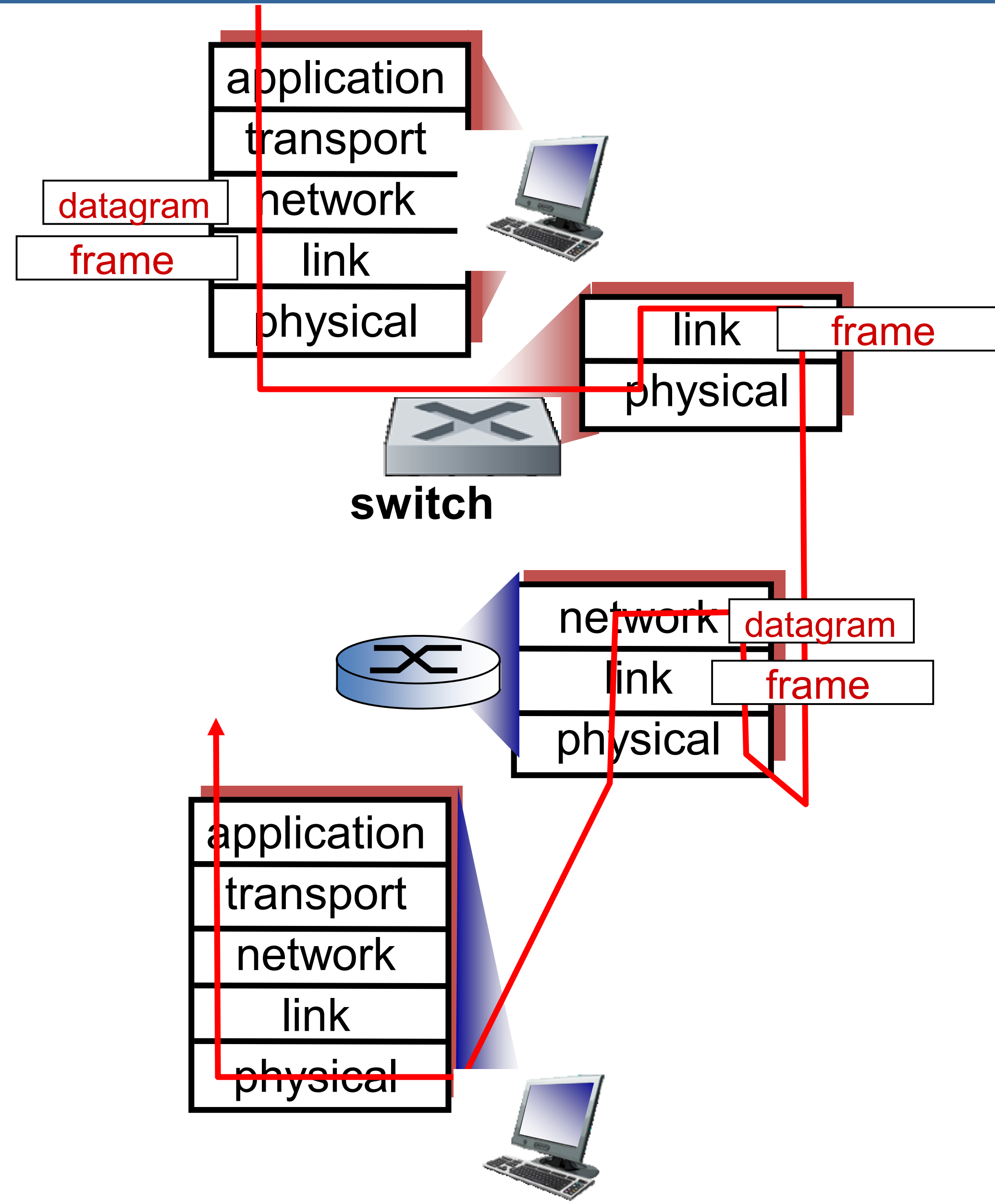
- Increased reliability
  - desirable to have redundant, alternate paths from source to destination
- With multiple simultaneous paths, cycles result
  - switches may multiply and forward frame forever
- Solution: organize switches in a spanning tree by disabling subset of interfaces





http://www.youtube.com/watch?v=ihF_78oIaDI

Switches vs. routers



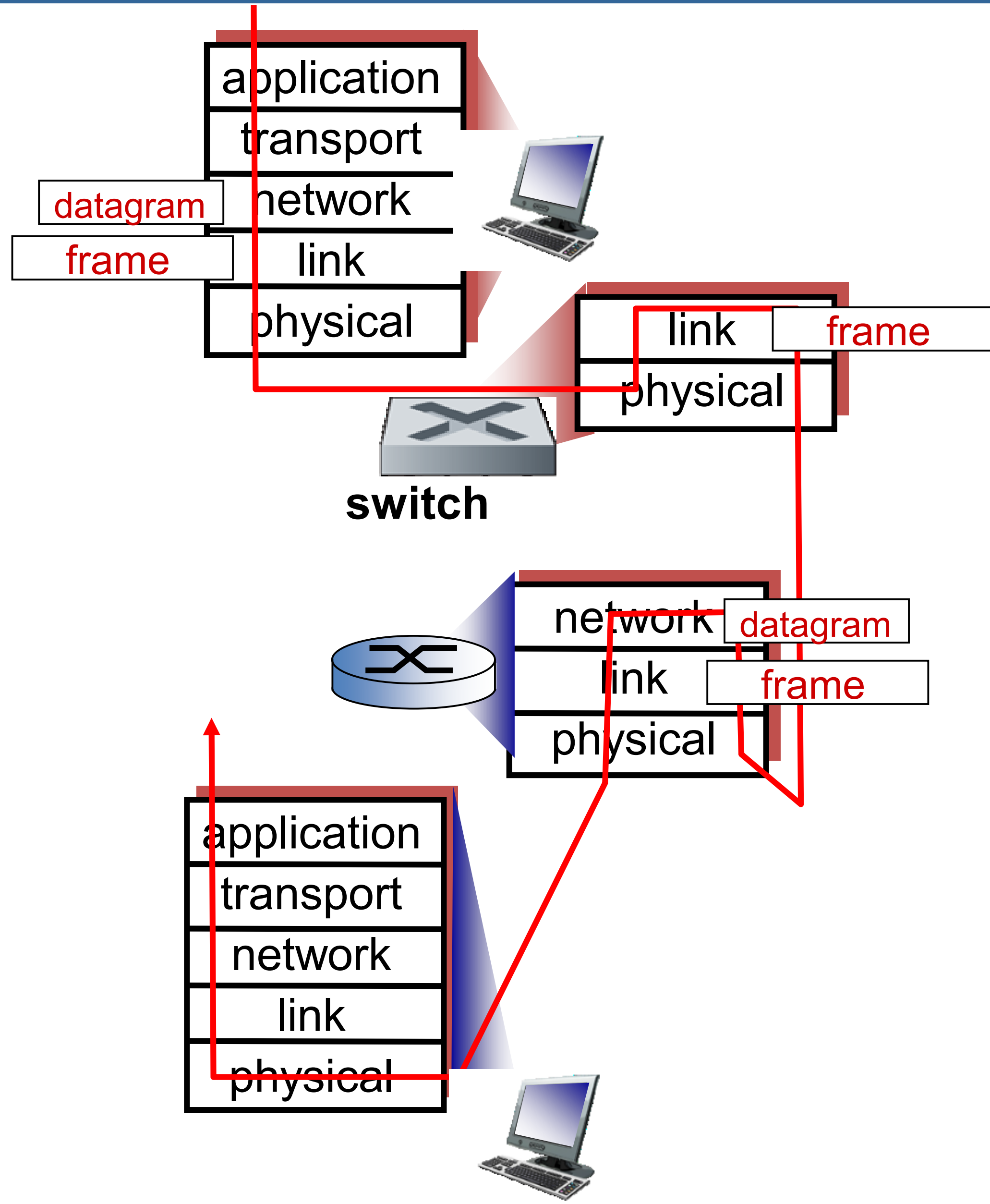
Switches vs. routers

both are store-and-forward:

- *routers:* network-layer devices (examine network-layer headers)
- *switches:* link-layer devices (examine link-layer headers)

both have forwarding tables:

- *routers:* compute tables using routing algorithms, IP addresses
- *switches:* learn forwarding table using flooding, learning, MAC addresses



Routers versus switches

Switches + and –

+ Switch operation is simpler requiring less processing bandwidth (only layer 2)

+ Plug and Play devices

– Topologies are restricted with switches (a spanning tree must be built to avoid cycles)

– Switches ***do not offer protection from broadcast storms*** (endless broadcasting by a host will be forwarded by a switch)

– Large networks will lead to nodes with large ARP tables and a lot of ARP traffic

Routers versus switches (cont.)

Routers + and –

- + **Arbitrary** topologies can be supported, cycling is limited by TTL counters (and good routing protocols)
- + Provide **firewall protection** against broadcast storms
- – Require IP address configuration (not plug and play)
- – Require higher processing bandwidth (e.g. routing algorithms)

Switches do well in ***small*** (few hundred hosts) while routers used in ***large*** networks (thousands of hosts)

COMPUTER NETWORKS AND APPLICATIONS

COMP SCI 3001

Faculty of Engineering, Computer and Mathematical Sciences

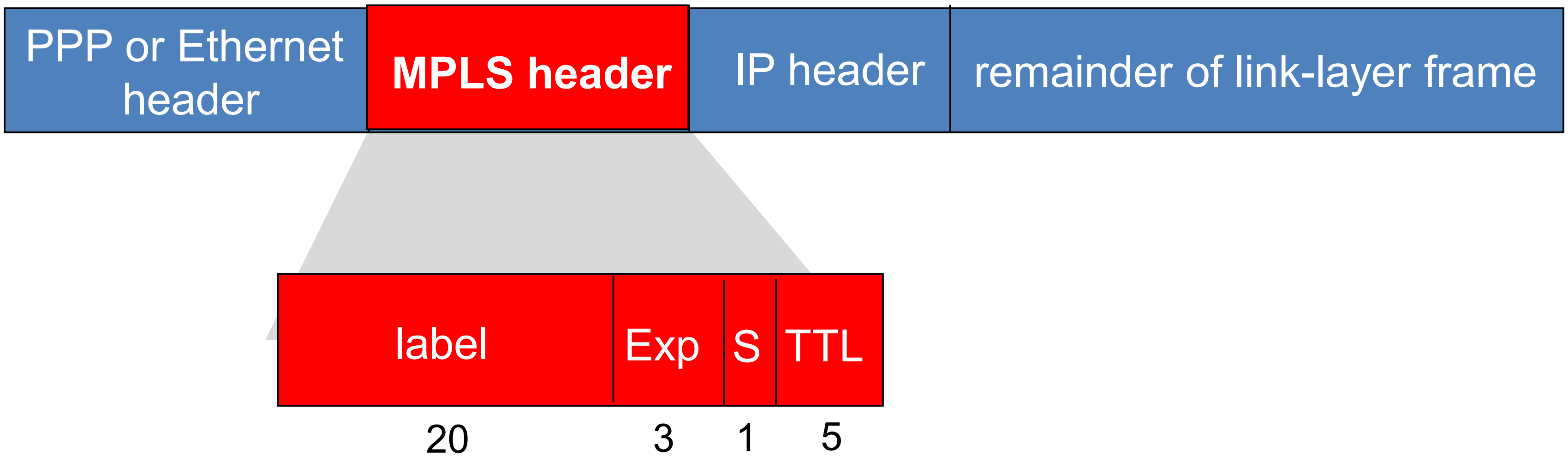
MPLS

MPLS - where did it come from?

- IP over ATM was a mechanism to get a good layer 3 protocol working over fast and efficient lower level hardware
  - but there were scalability problems
- Toshiba produced a **Cell Switching Router** (CSR) that had an ATM switching fabric controlled by IP protocols
- A little later Ipsilon produced their **IP Switching** solution. Their rationale:
  - want a device as fast as an ATM switch that routes
  - the Internet needs fast routers, not switches, because IP is dominant
  - ATM signalling and mapping to IP is complex; ditch ATM control protocols!
- Cisco followed this up with **Tag Switching**
  - didn't need data traffic to flow to populate its tables
  - worked for link layer technologies other than ATM
  - after revision and consultation this became MPLS

Multiprotocol label switching (MPLS)

- ***Initial goal***: high-speed IP forwarding using fixed length label (instead of IP address)
  - fast lookup using fixed length identifier (rather than longest prefix matching)
  - borrowing ideas from Virtual Circuit (VC) approach
  - but IP datagram still keeps IP address!

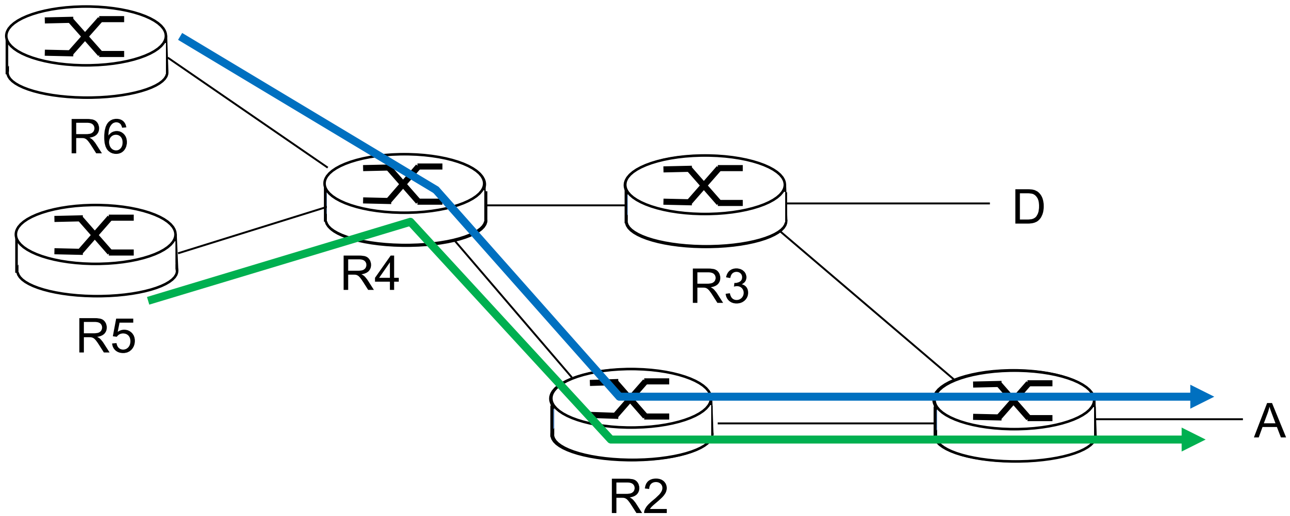


Need MPLS capable routers: Label-Switched Router (LSR)

MPLS capable routers

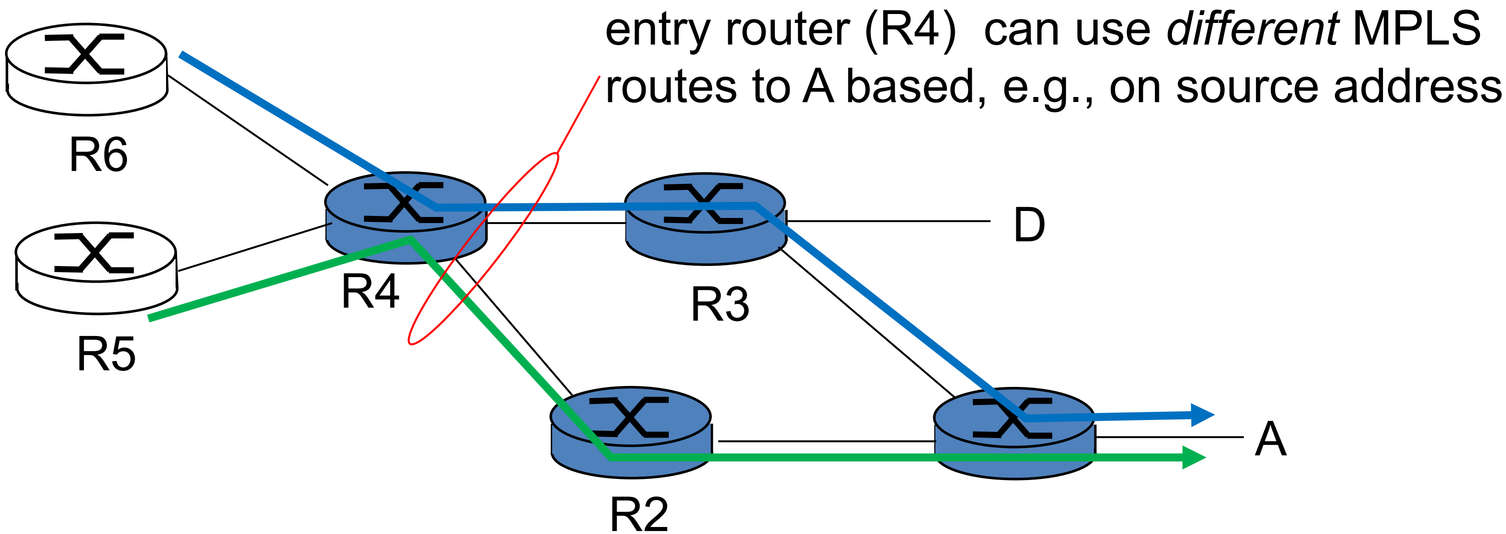
- a.k.a. **label-switched router**
- forward packets to outgoing interface based **only** on label value (*don't inspect IP address*)
  - MPLS forwarding table distinct from IP forwarding tables

MPLS versus IP paths

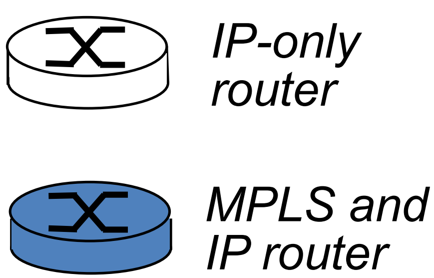


- *IP routing:* *path to destination determined by destination address alone*

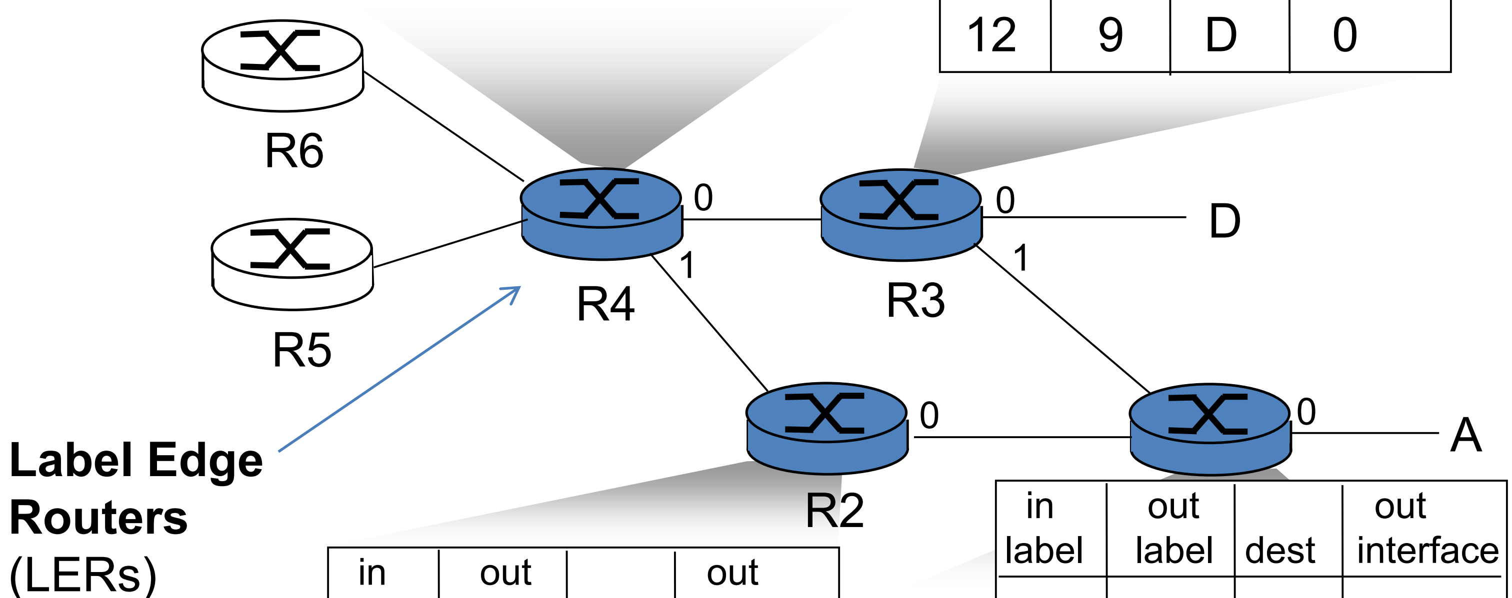
MPLS versus IP paths (*main interest now*: traffic engineering)



- *IP routing:* *path to destination determined by destination address alone*
- *MPLS routing:* path to destination can be based on source *and* dest. address
  - *fast reroute:* precompute backup routes in case of link failure



MPLS Example



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

MPLS - how it works

- Routers work out routing tables and forward packets through their interfaces
  - for unicast packets we use a longest prefix match on the destination IP address
- This partitions all possible packets that a router can forward into a finite number of disjoint subsets. Why disjoint?
  - from a forwarding point of view anything sent to the same next hop is the same
  - we refer to these subsets as **Forwarding Equivalence Classes** (FECs)

MPLS - how it works (cont.)

- If we know that a set of IP prefixes are all going out the same interface we can throw all of them into the same **Forwarding Equivalence Class**
- Our forwarding table is now a set of FECs and a next hop for each FEC
- We can set the granularity of these as we wish
  - a FEC can model many entries or just one
  - **this is where the scalability comes from**
- We need to map into the FEC consistently or we can get some odd results

MPLS - label swapping

- When a packet arrives, the label is extracted and compared to a forwarding table
- The table will have subentries matching each incoming label
  - these contain the outgoing label, the outbound interface and the designated next hop
- You can have multiple subentries for each incoming label to implement (among other things) traffic management capabilities
- The forwarding table may also specify resource usage, such as which outgoing queue to use
  - this allows the implementation of Quality of Service requirements
- The incoming label is replaced by the outgoing label in the subentry and sent out over the designated interface to the specified next hop
  - this is called **label swapping**

MPLS Features

- Single forwarding algorithm
  - multicast is the same as unicast, just with more subentries
  - Just make sure swapping works correctly
- Multiprotocol support: up and down
  - the forwarding component is ***not network layer specific***
  - we can stick a label onto the front of any packet
  - similarly, we can also use an underlying link layer protocol through the *use of existing link layer fields or ‘shim’ headers*

MPLS Features

- *flexibility:* MPLS forwarding decisions can *differ* from those of IP
  - use ***destination and source addresses*** to route flows to same destination differently (***traffic engineering***)
  - re-route flows quickly if link fails: pre-computed backup paths (useful for VoIP)
- MPLS can support multiple levels of connection tunnelling through label stacking
  - VPN support

MPLS signaling (distributing labels)

- modify link-state flooding protocols to carry info used by MPLS routing,
  - e.g., link bandwidth, amount of "reserved" link bandwidth

- *IETF effort: entry MPLS router uses RSVP-TE signaling protocol to set up MPLS forwarding at downstream routers*