Network Layer:
Link Layer; Course Summary

Qiao Xiang, Congming Gao, **Qiang Su**

https://sngroup.org.cn/courses/cnns-xmuf25/index.shtml

12/4/2025

This deck of slides are heavily based on CPSC 433/533 at Yale University, by courtesy of Dr. Y. Richard Yang.

Recap: IP Addressing Scheme: Requirements

- Uniqueness: We need an address to uniquely identify each destination
- Aggregability : Routing scalability needs flexibility in aggregation of destination addresses
  - we want to aggregate as a large set of destinations as possible in BGP announcements
- Current: the unit of routing in the Internet is a classless interdomain routing (CIDR) address

Recap: Network Forwarding: Putting it Together

- Forwarding is also called the fast path (upon receiving each packet)

- Slow path: not per packet
  - Get IP address (DHCP, or static)
  - Setup/compute routing table

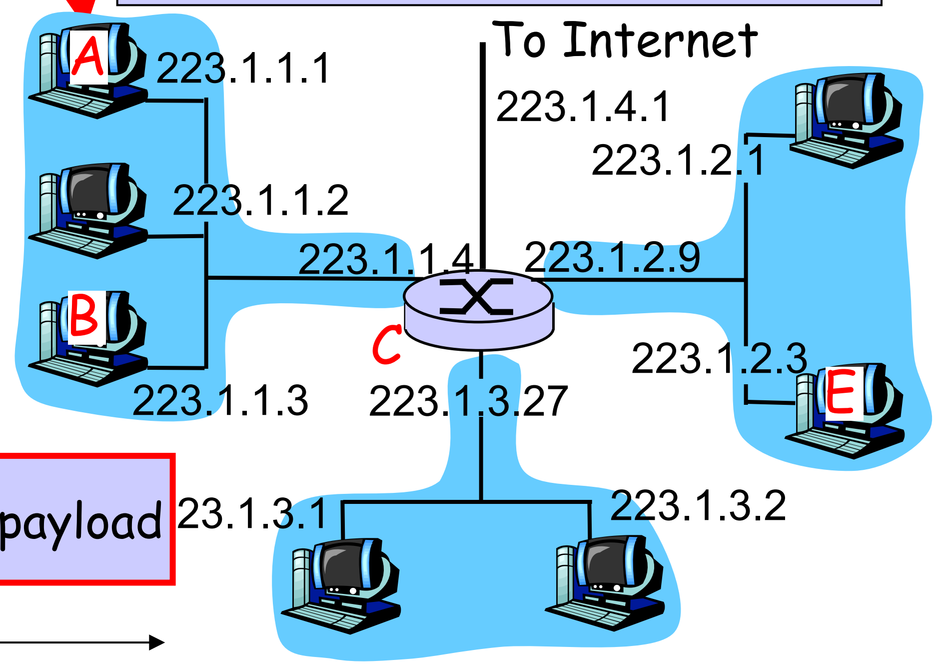
Recap: Putting it Together: Example 2 (Different Networks): A-> E

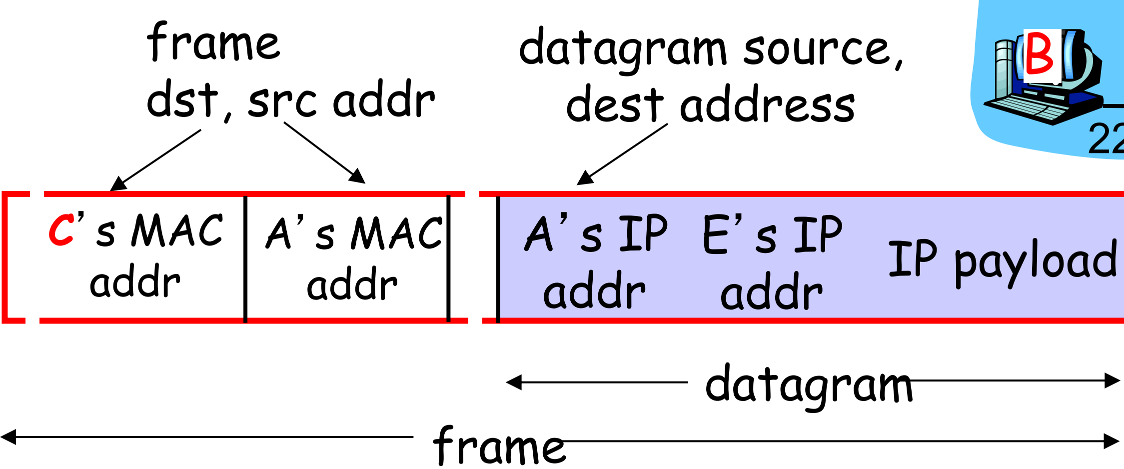
misc fields	223.1.1.1	223.1.2.3	data

- Setting: Host A network layer receives a packet above.
- Action:
  - Host A looks up destination in routing table
    - Find next hop should be 223.1.1.4
  - Hand datagram to link layer to send inside a link-layer frame

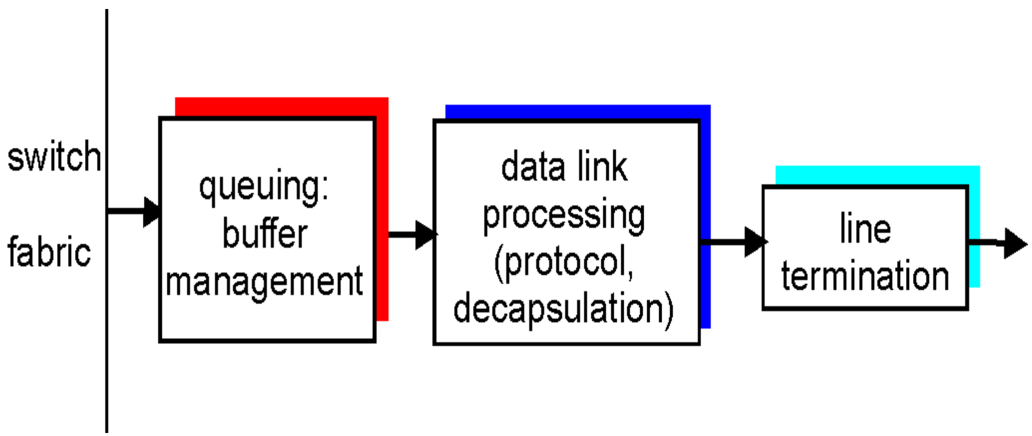
forwarding table in A

Dest. Net.	next router	Nhops
223.1.1/24		1
223.1.2/24	223.1.1.4	2
223.1.3/24	223.1.1.4	2
0.0.0.0/0	223.1.1.4	-



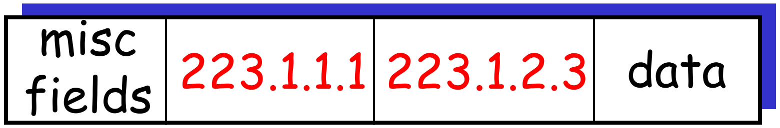


Recap: Look Inside a Router: Output Port



- *Buffering* required when datagrams arrive from fabric faster than the transmission rate
- *Queueing (delay) and loss due to output port buffer overflow !*
- *Scheduling and queue/buffer management* choose among queued datagrams for transmission

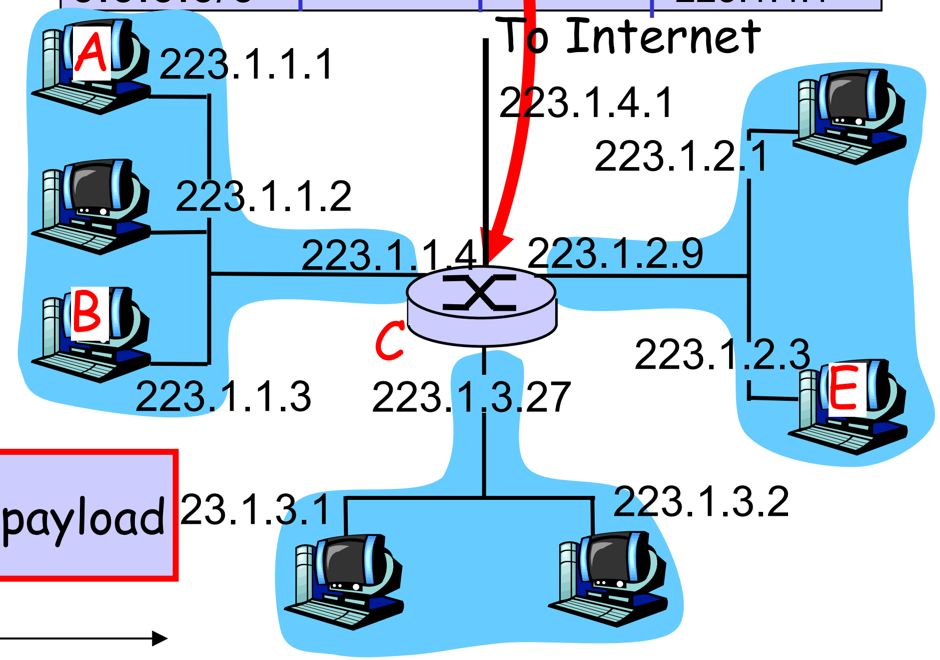
Recap: Putting it Together: Example 2 (Different Networks): A-> E

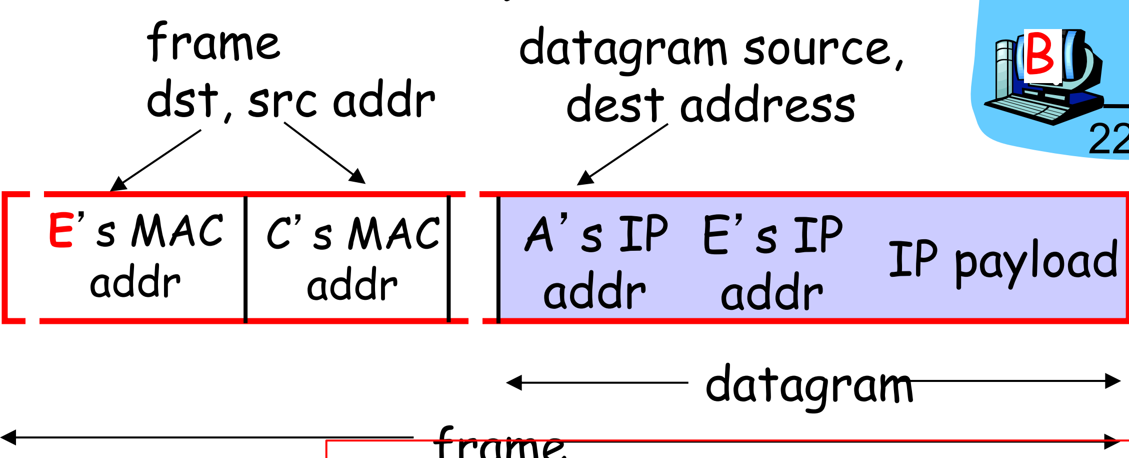


- Setting: Packet above arrives at Router C's network layer.
- Action:
  - Router C conducts standard router actions
    - Assume packet correct, find next hop should be 223.1.2.9
  - Hand datagram to link layer to send inside a link-layer frame

forwarding table in router

Dest. Net	router	Nhops	interface
223.1.1/24	-	1	223.1.1.4
223.1.2/24	-	1	223.1.2.9
223.1.3/24	-	1	223.1.3.27
0.0.0.0/0	-	-	223.1.4.1





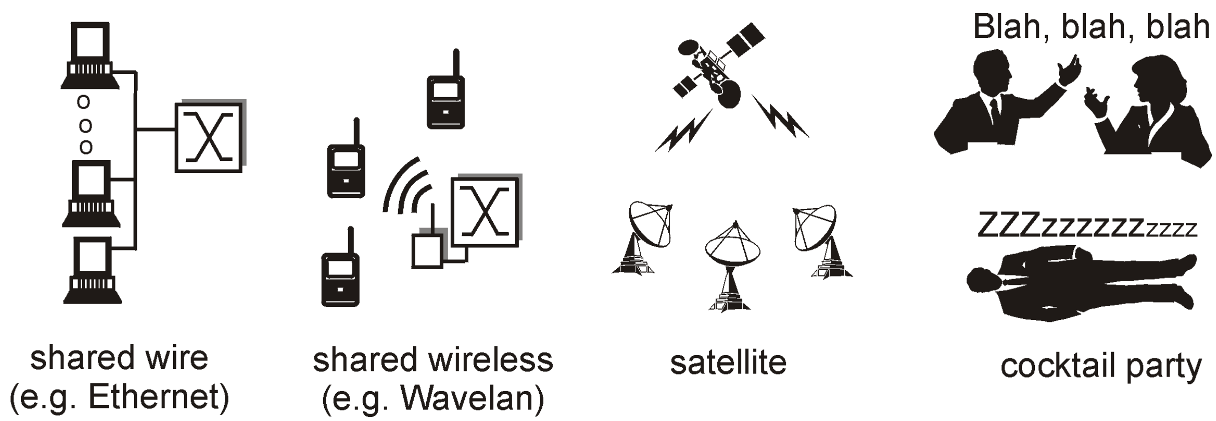
datagram arrives at 223.1.2.3!! (hooray!)

Outline

- Admin and recap
- Network layer
- Link layer
  - Overview
  - Media access
  - Link layer forwarding

Multiple Access Links and Protocols

- Many link layers use broadcast (shared wire or medium)
  - traditional Ethernet; Cable networks
  - 802.11 wireless LAN; cellular networks
  - satellite



- Problem: if two or more simultaneous transmissions, due to interference, only one node can send successfully at a time (see CDMA later for an exception)

Multiple Access Protocol

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

- Discussion: properties of an ideal multiple access protocol.

Ideal Mulitple Access Protocol

Broadcast channel of rate R bps

- Efficiency: when only one node wants to transmit, it can send at full rate R
- Rate allocation:
  - simple fairness: when N nodes want to transmit, each can send at average rate R/N
  - we may need more complex rate control
- Decentralized:
  - no special node to coordinate transmissions
  - no synchronization of clocks
- Simple

MAC Protocols

Goals

- efficient, fair, decentralized, simple

Three broad classes:

- non-partitioning
  - random access
    - allow collisions
  - "taking-turns"
    - a token coordinates shared access to avoid collisions
- channel partitioning
  - divide channel into smaller "pieces" (time slot, frequency, code)

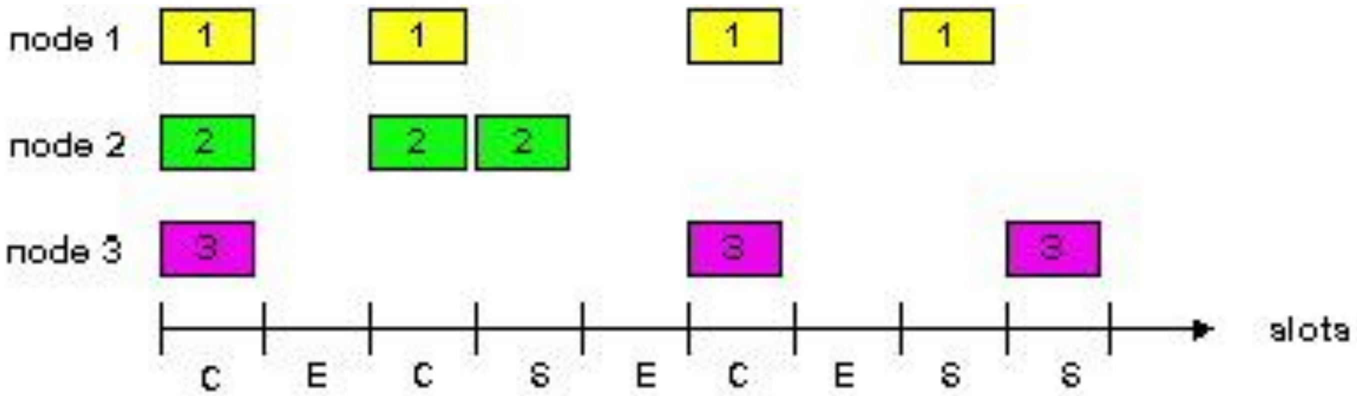
Focus: Random Access Protocols

- Examples of random access MAC protocols:
  - slotted ALOHA and pure ALOHA
  - CSMA and CSMA/CD, CSMA/CA
  - Ethernet, WiFi 802.11

- Key design points:
  - when to access channel?
  - how to detect collisions?
  - how to recover from collisions?

Slotted Aloha [Norm Abramson]

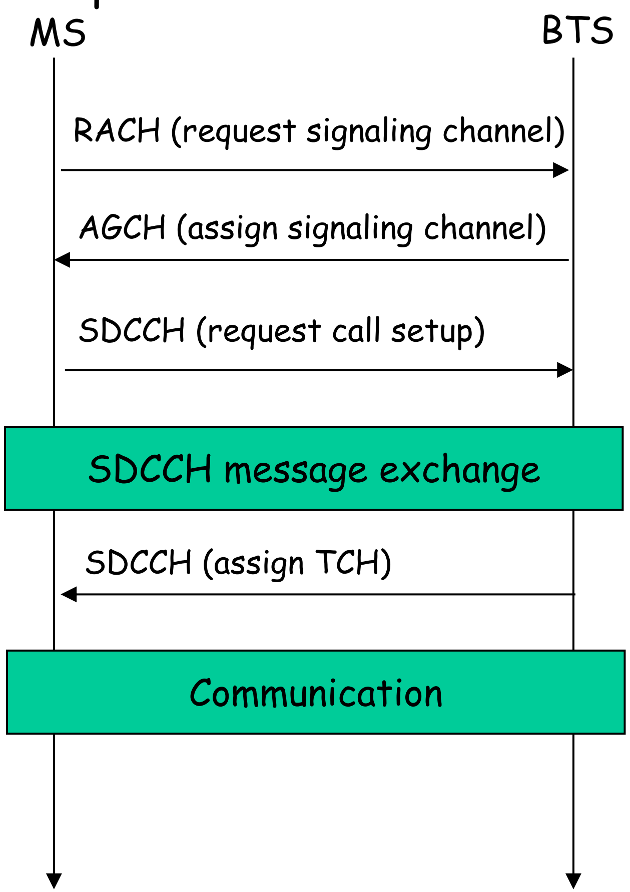
- Time is divided into equal size slots (= pkt trans. time)
- Node with new arriving pkt: transmit at beginning of next slot
- If collision: retransmit pkt in future slots with probability p, until successful.



Success (S), Collision (C), Empty (E) slots

Slotted Aloha in Real Life

- call setup in GSM



- Notations:
  - Broadcast control channel (BCCH): from base station, announces cell identifier, synchronization
  - Random access channel (RACH): MSs for initial access, **slotted Aloha**
  - access grant channel (AGCH): BTS informs an MS its allocation
  - standalone dedicated control channel (SDCCH): signaling and short message between MS and an MS
  - Traffic channels (TCH)

Slotted Aloha Efficiency

Q: What is the fraction of successful slots?

suppose n stations have packets to send

suppose each transmits in a slot with probability *p*

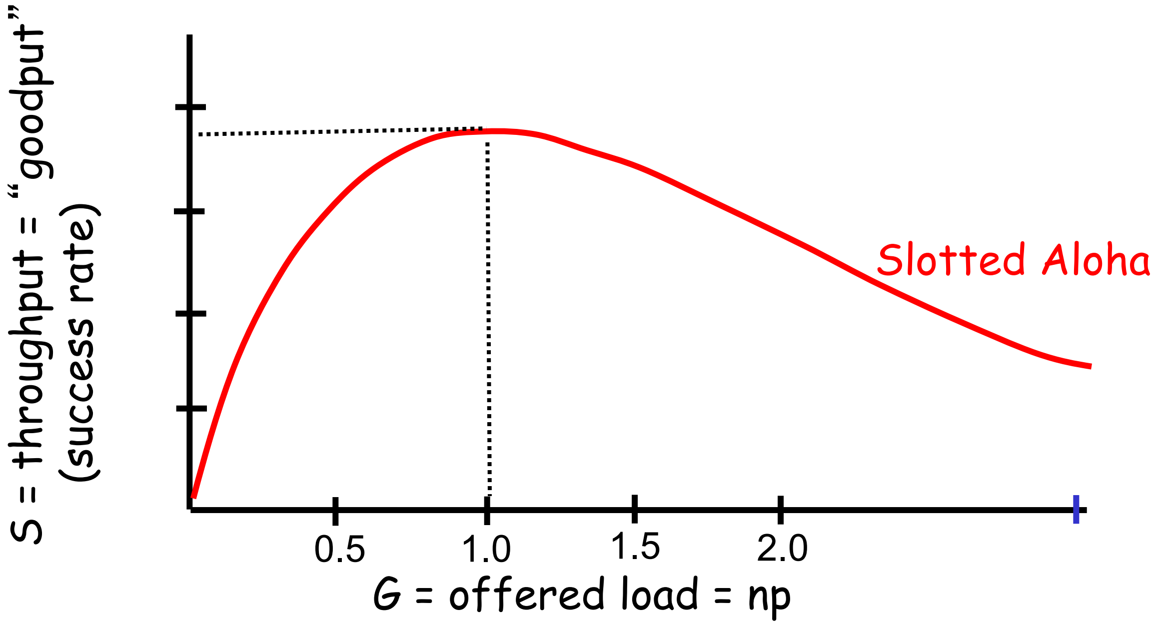
- prob. of succ. by a specific node: $p\,(1-p)^{(n-1)}$

- prob. of succ. by any one of the N nodes

  $S(p) = n * \text{Prob (only one transmits)}$

  $= n\,p\,(1-p)^{(n-1)}$

Goodput vs. Offered Load for Slotted Aloha

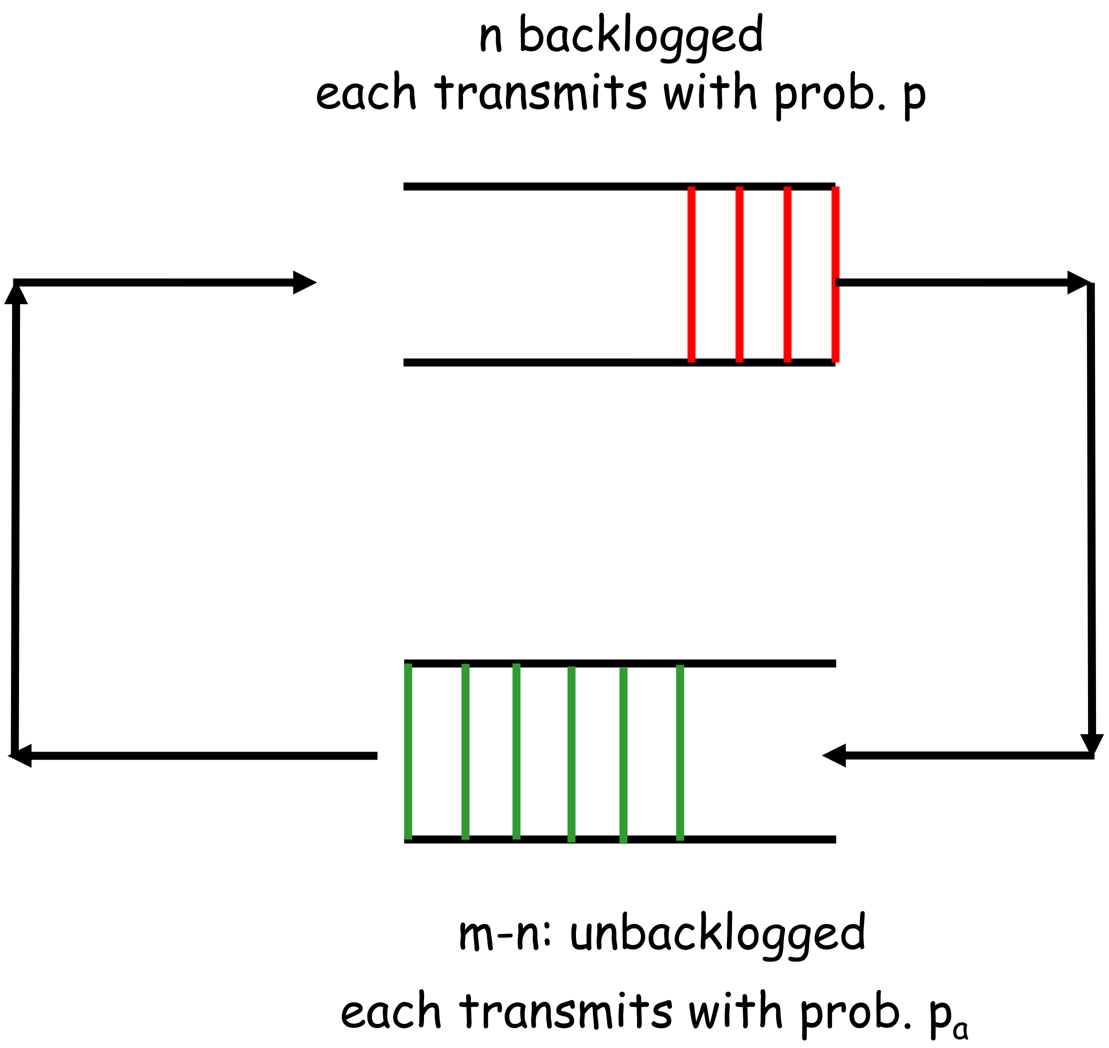


- when p n < 1, as p (or n) increases
  - probability of empty slots reduces
  - probability of collision is still low, thus goodput increases
- when p n > 1, as p (or n) increases,
  - probability of empty slots does not reduce much, but
  - probability of collision increases, thus goodput decreases
- goodput is optimal when p n = 1, n -> infinite, S -> 1/e (~37%)

Dynamics of (Slotted) Aloha

- Slotted Aloha has maximum throughput when $np = 1$
  - Implies we need to adjust p as the number of backlog stations varies.
- Early design question: what is the effect if we do not change p--use a fixed p
  - Assume we have a total of m stations (the machines on a LAN):
    - n of them are currently backlogged, each tries with a (fixed) probability p
    - the remaining m-n stations are not backlogged. They may start to generate packets with a probability p_a, where p_a is much smaller than p

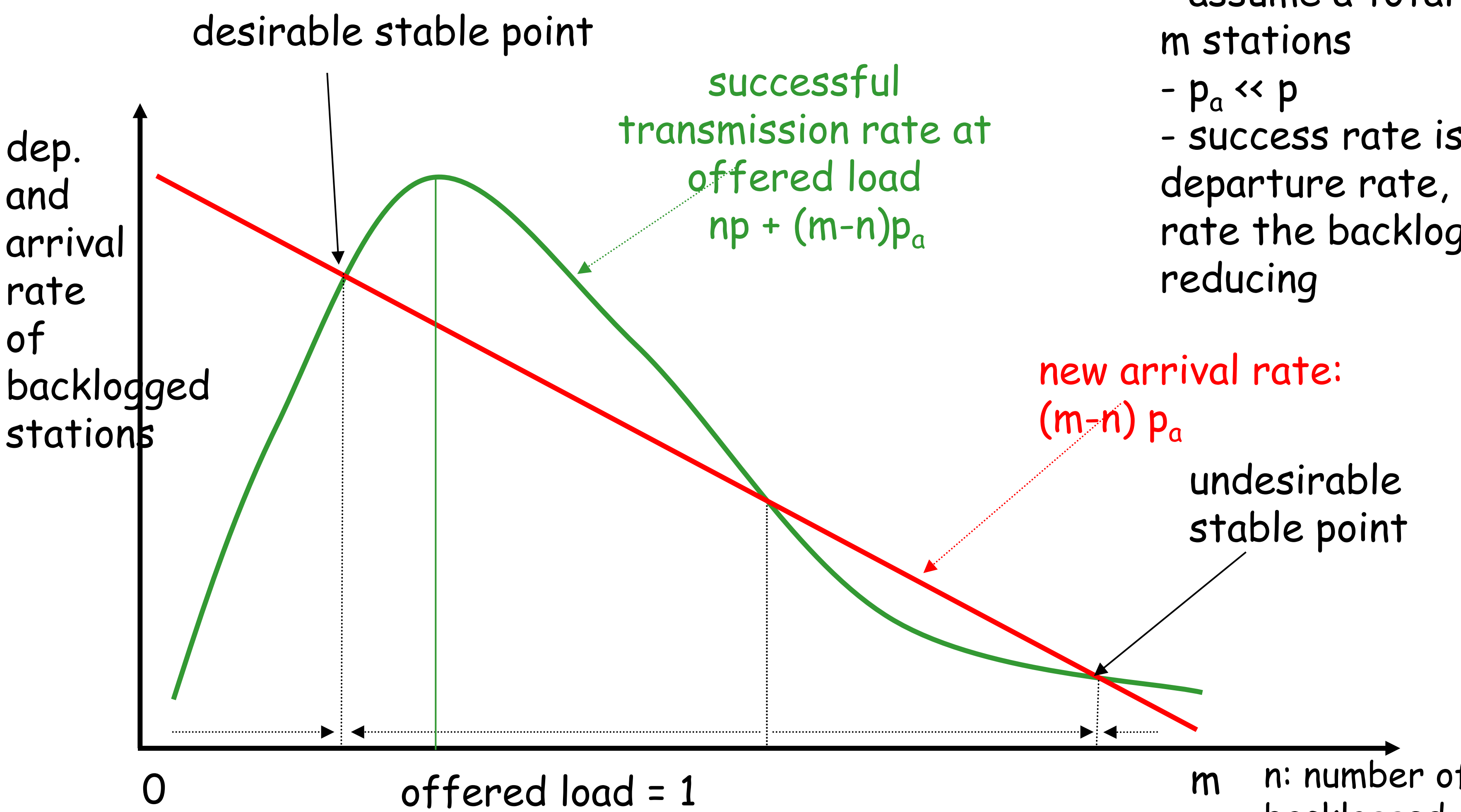
Model



n backlogged
each transmits with prob. p
m-n: unbacklogged
each transmits with prob. p_a

Dynamics of Aloha: Effects of Fixed Probability

- assume a total of m stations
- $p_a \ll p$
- success rate is the departure rate, the rate the backlog is reducing



Lesson: if we fix p, but n varies, we may have an undesirable stable point

Summary of Problems of Aloha Protocols

- Problems
  - slotted Aloha has better efficiency than pure Aloha but clock synchronization is hard to achieve
  - Aloha protocols have low efficiency due to collision or empty slots
    - when offered load is optimal ($p = 1/N$), the goodput is only about 37%
    - when the offered load is not optimal, the goodput is even lower
  - undesirable steady state at a fixed transmission rate, when the number of backlogged stations varies

- Ethernet design: address the problems:
  - optimal transmission rate

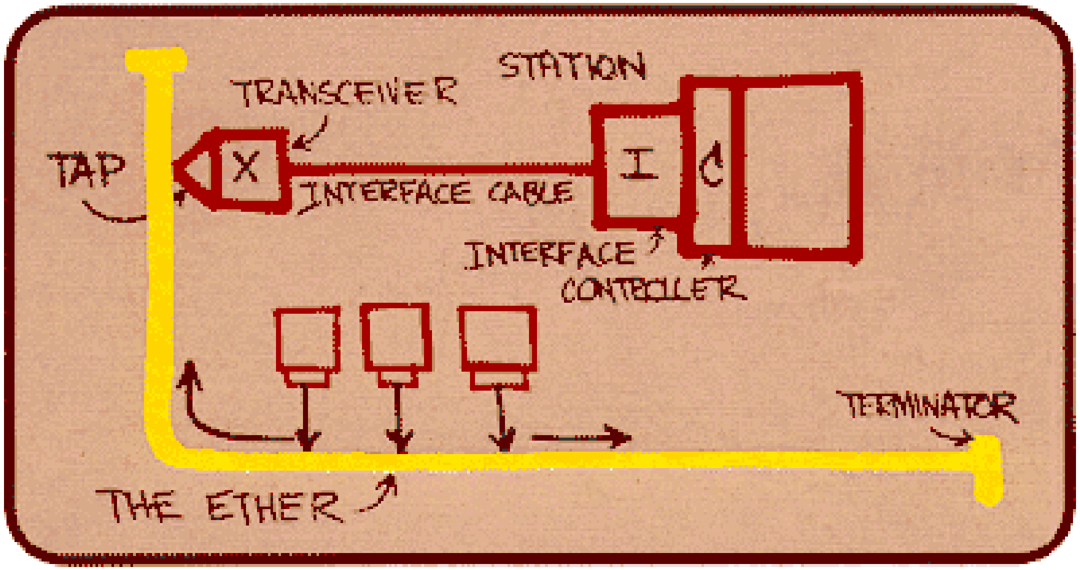
The Basic MAC Mechanisms of Ethernet

```
get a packet from upper layer;
K := 0; n := 0; // K: control wait time; n: no. of collisions
repeat:
  wait for K * 512 bit-time;
  while (network busy) wait;
  wait for 96 bit-time after detecting no signal;
  transmit and detect collision;
  if detect collision
     stop and transmit a 48-bit jam signal;
     n ++;
     m:= min(n, 10), where n is the number of collisions
     choose K randomly from {0, 1, 2, …, 2^m-1}.
     if n < 16 goto repeat
     else give up
```

Ethernet

"Dominant" LAN technology:

- First widely used LAN technology
- Kept up with speed race: 10 Mbps, 100 Mbps, 1 Gbps, 10 Gbps



Metcalfe's Ethernet sketch

Outline

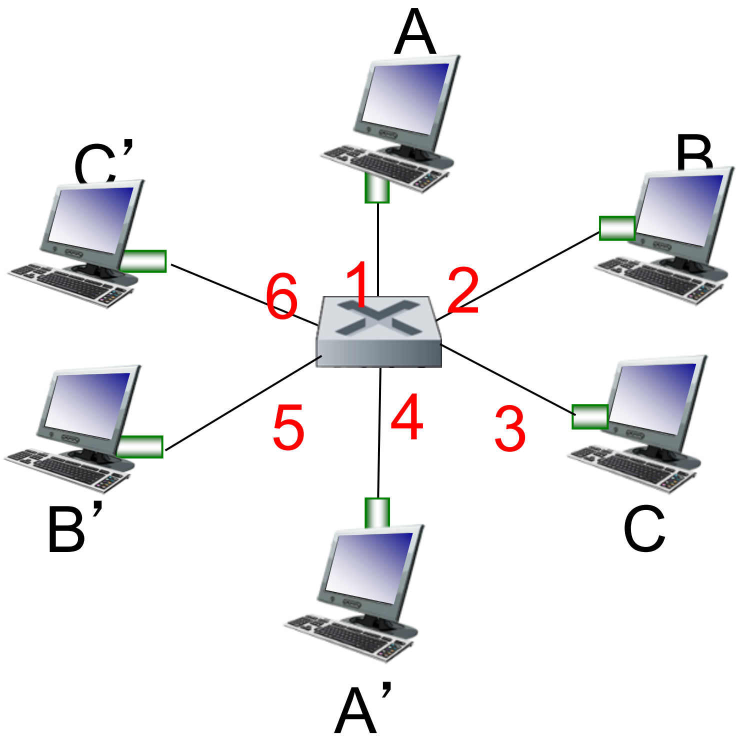
- Admin and recap
- Link layer
  - Ethernet switch

Ethernet Switch

- link-layer device: takes an *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: *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' can transmit simultaneously, without collisions



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

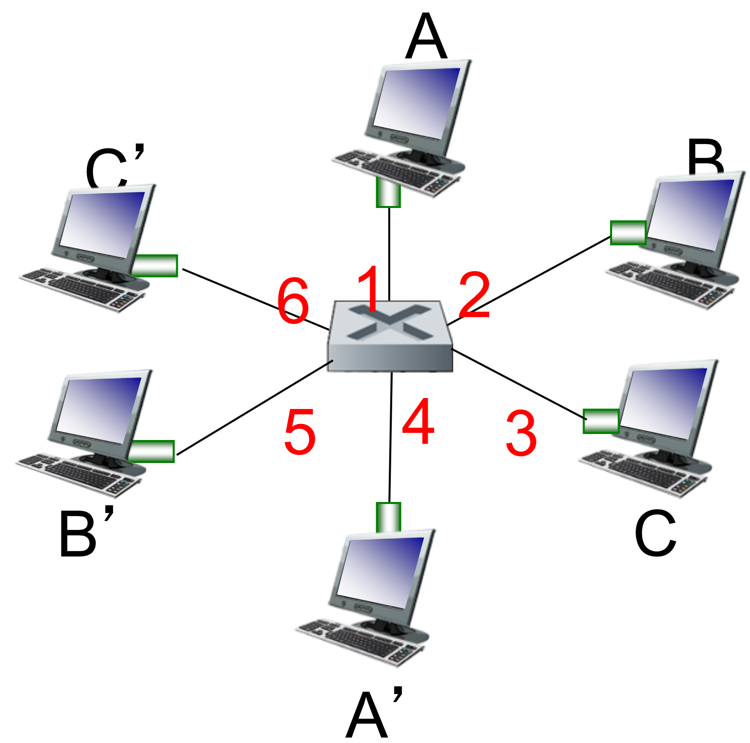
Switch Forwarding Table

Q: how does switch know 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?

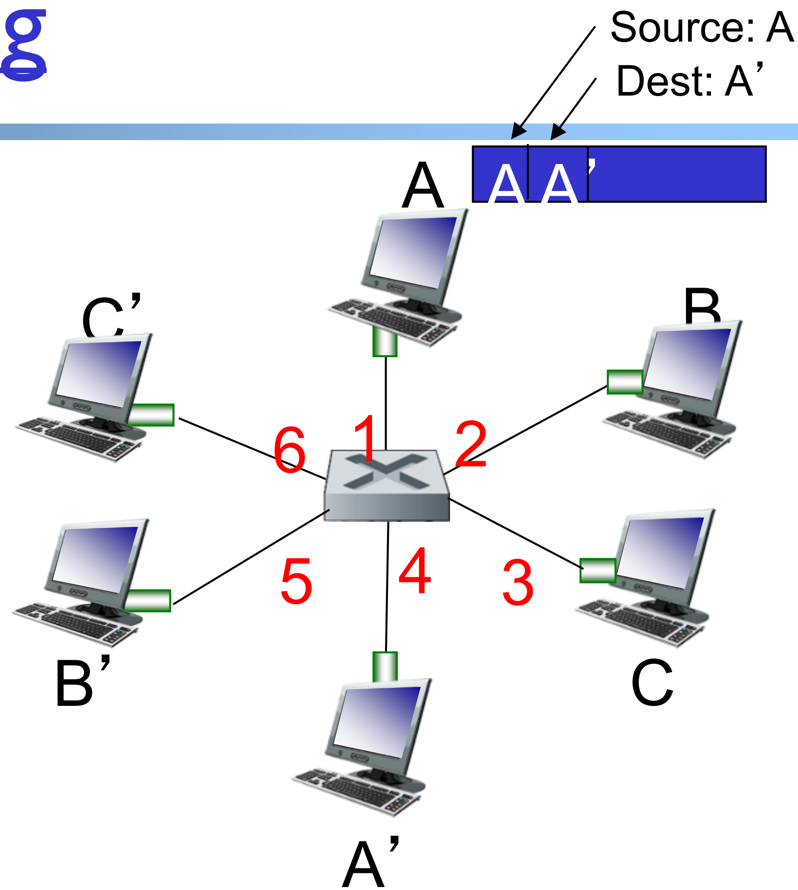
  - something like a routing protocol?



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)

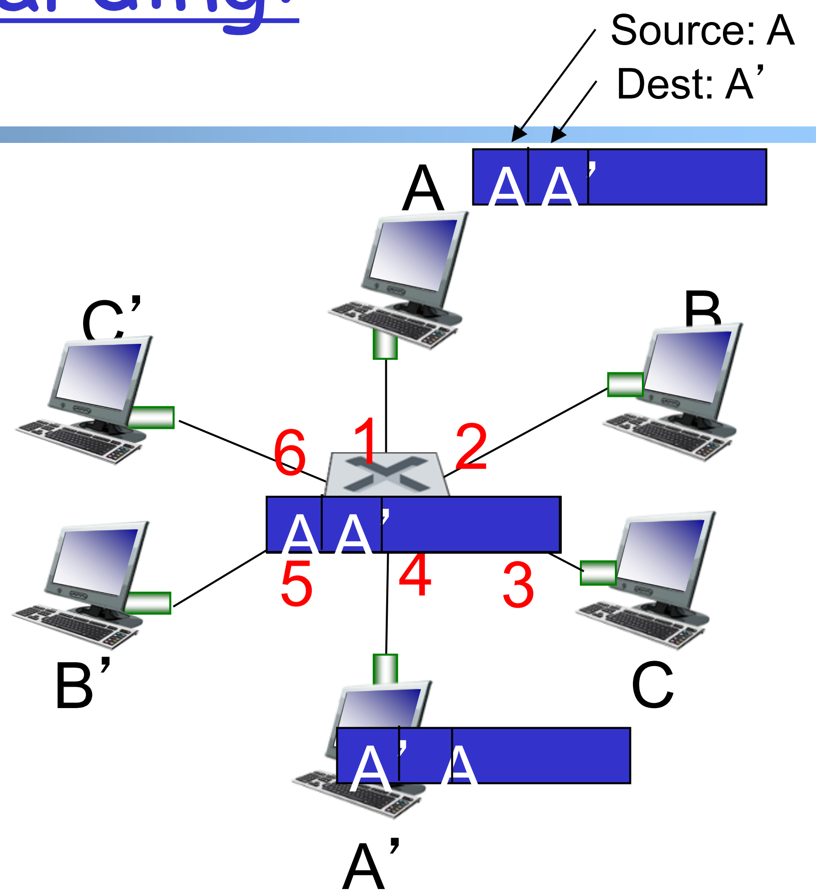
Switch: Frame Filtering /Forwarding

when frame received at switch:

1. record incoming link, MAC address of sending host
2. index switch table using MAC destination address
3. if entry found for destination
   then {
   if destination on segment from which frame arrived
   then drop frame
   else forward frame on interface indicated by entry
   }
   else flood /* forward on all interfaces except arriving interface */

Self-Learning, Forwarding: Example

- frame destination, A', location unknown: *flood*
- destination A location known: selectively send on just one link

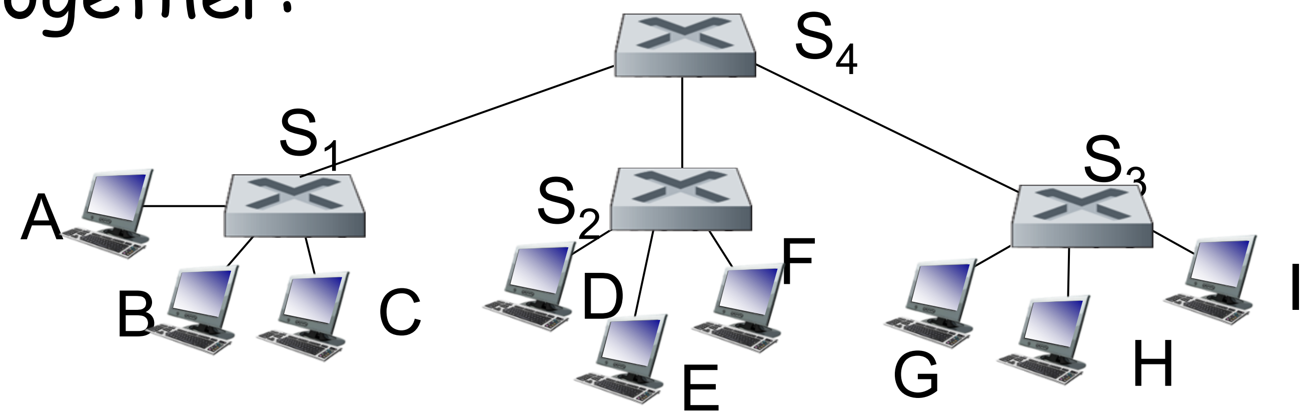


MAC addr	interface	TTL
A *A'*	*1* *4*	*60* *60*

switch table (initially empty)

Interconnecting Switches

self-learning switches can be connected together:

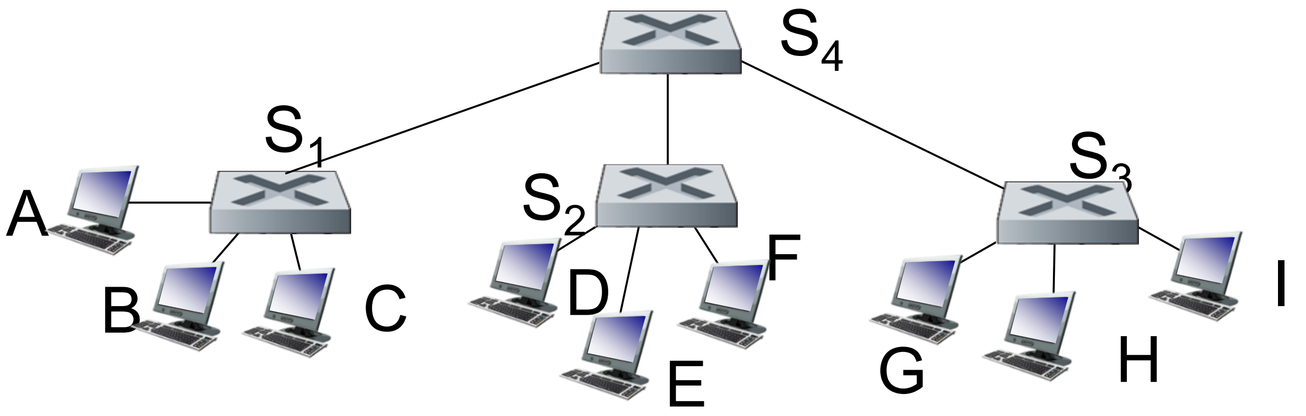


Q: sending from A to G - how does S_1 know to forward frame destined to G via S_4 and S_3?

- *A:* self learning! (works exactly the same as in single-switch case!)

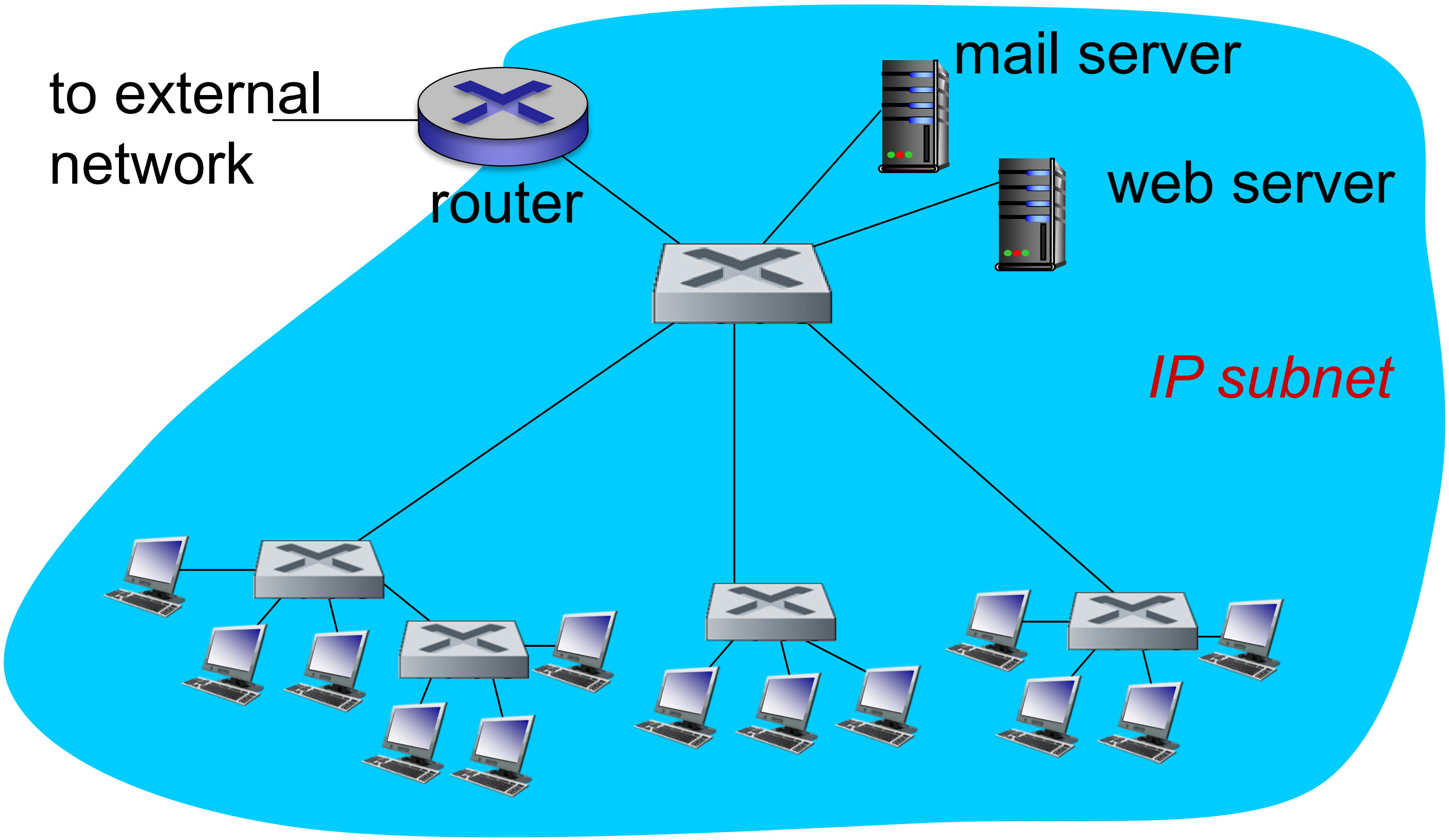
Self-Learning Multi-switch Example

Suppose C sends frame to I, I responds to C



- Offline Exercise: show switch tables and packet forwarding in S_1, S_2, S_3, S_4

Institutional Network



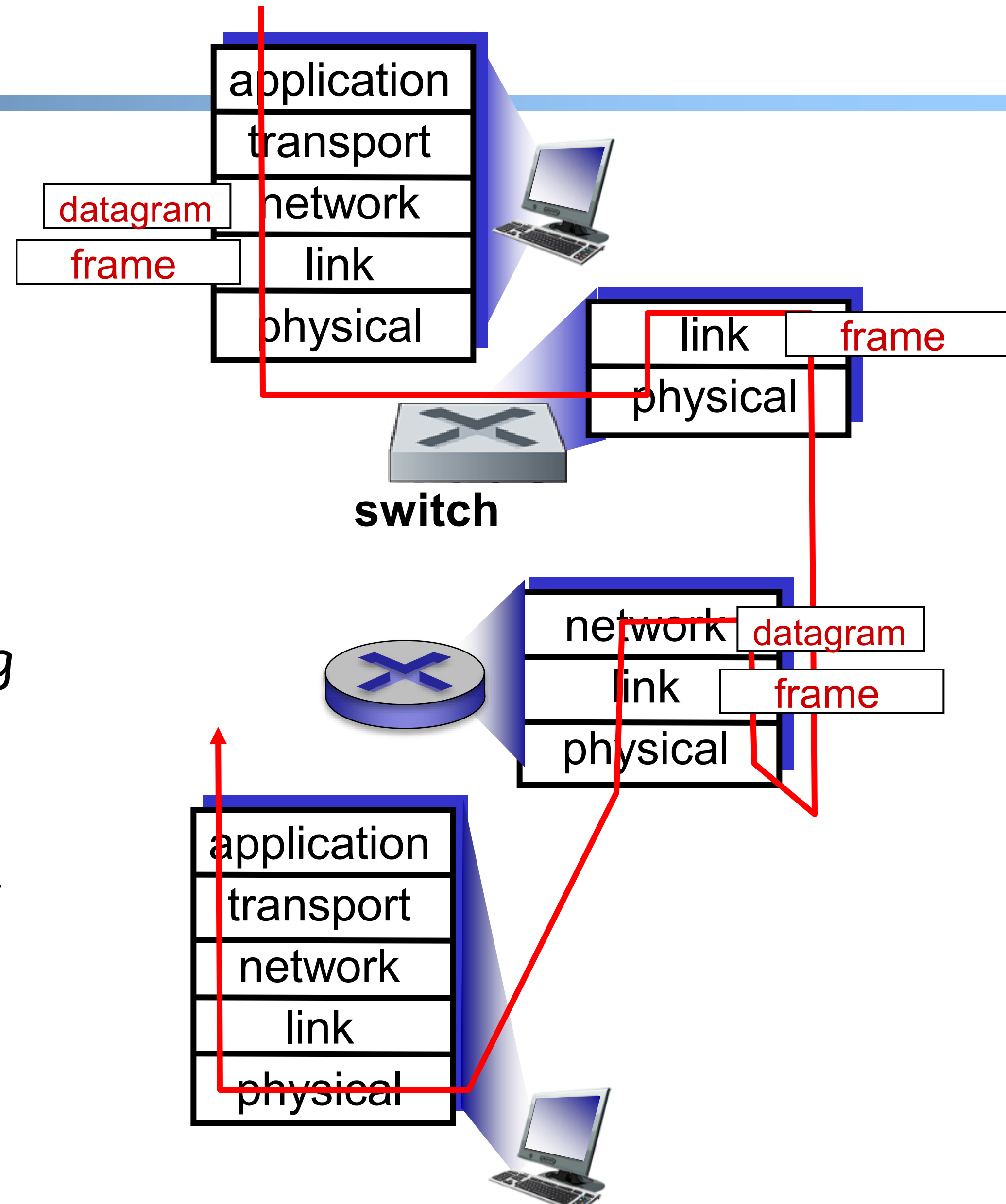
Switches vs. Routers

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



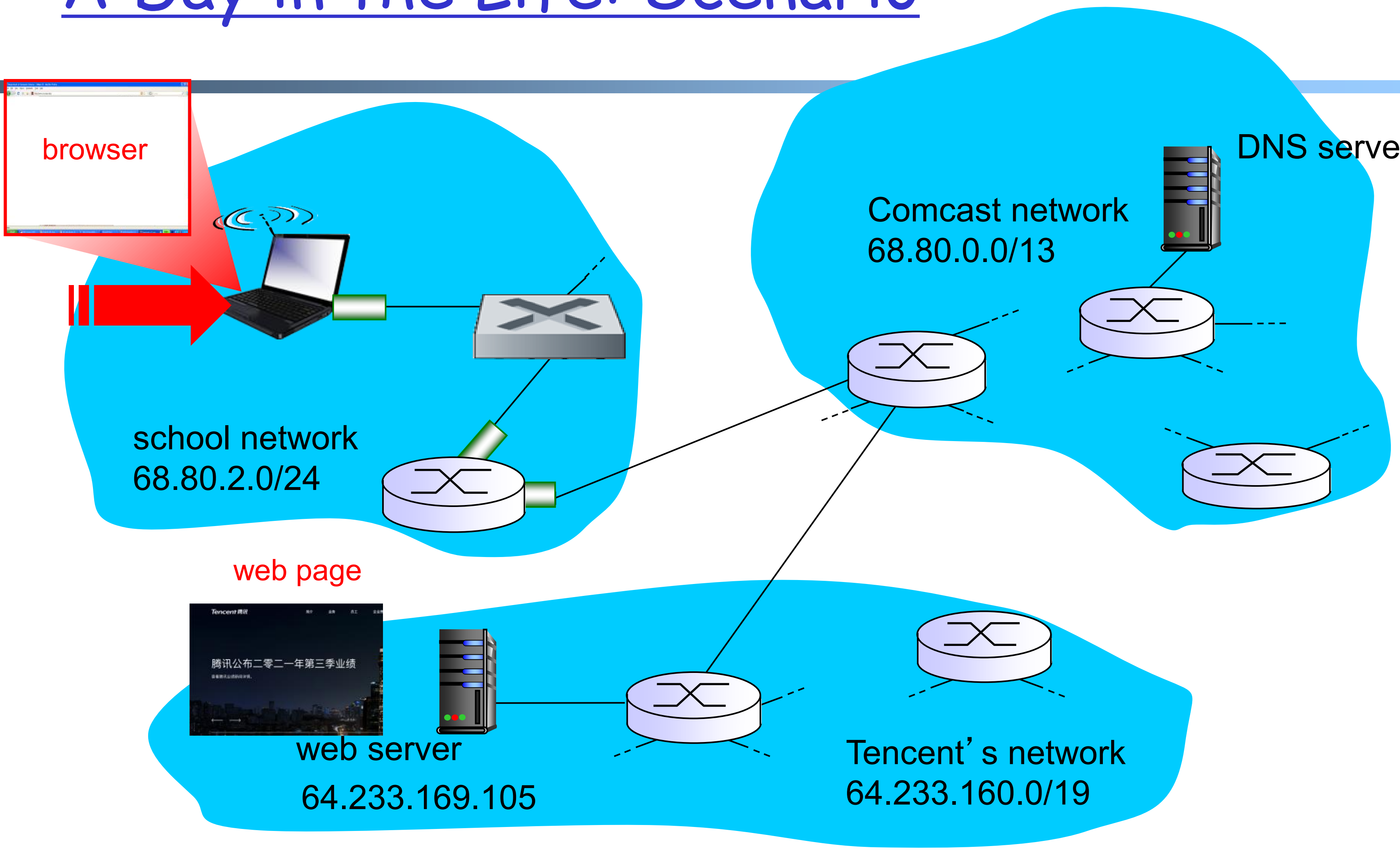
Outline

- Course Summary

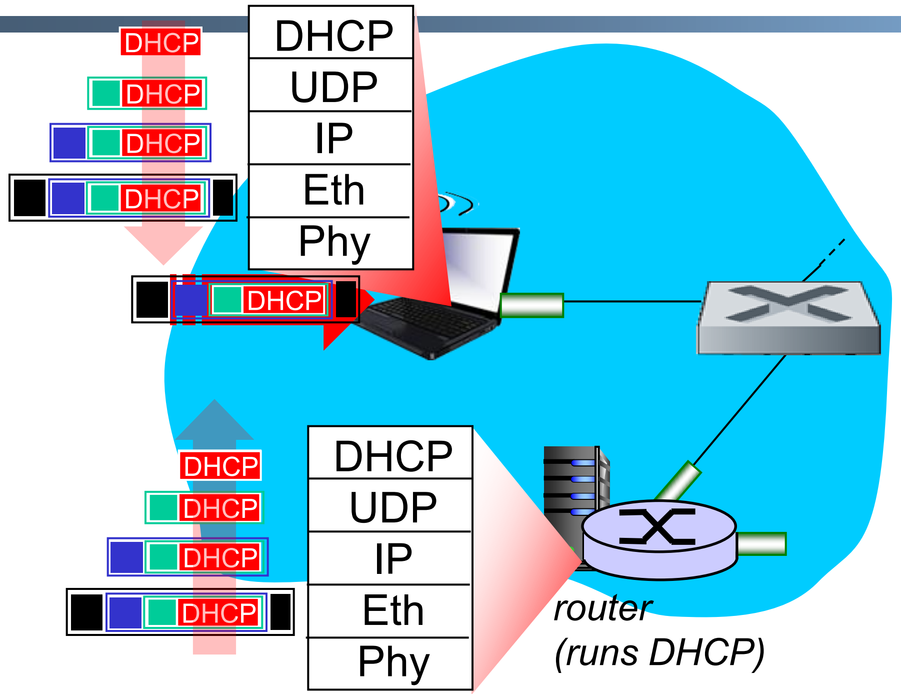
Synthesis: A Day in the Life of a Web Request

- 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.tencent.com

A Day in the Life: Scenario

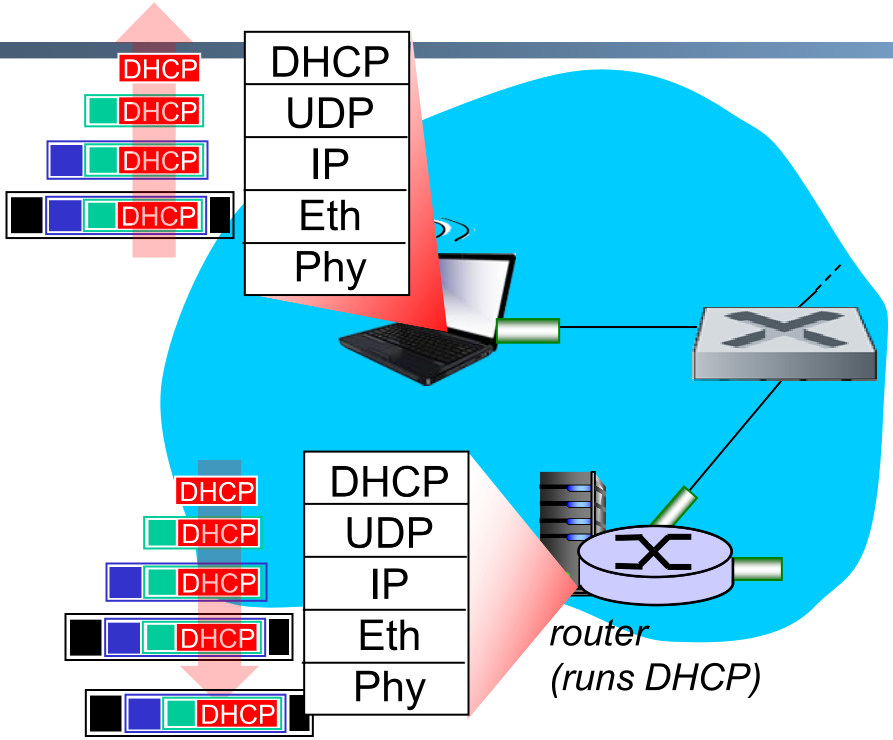


A Day in the Life... Connecting to the Internet



- 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 802.3 Ethernet
- Ethernet frame broadcast (dest: FFFFFFFFFFFF) on LAN, received at router running DHCP server
- Ethernet demuxed to IP demuxed, UDP demuxed to DHCP

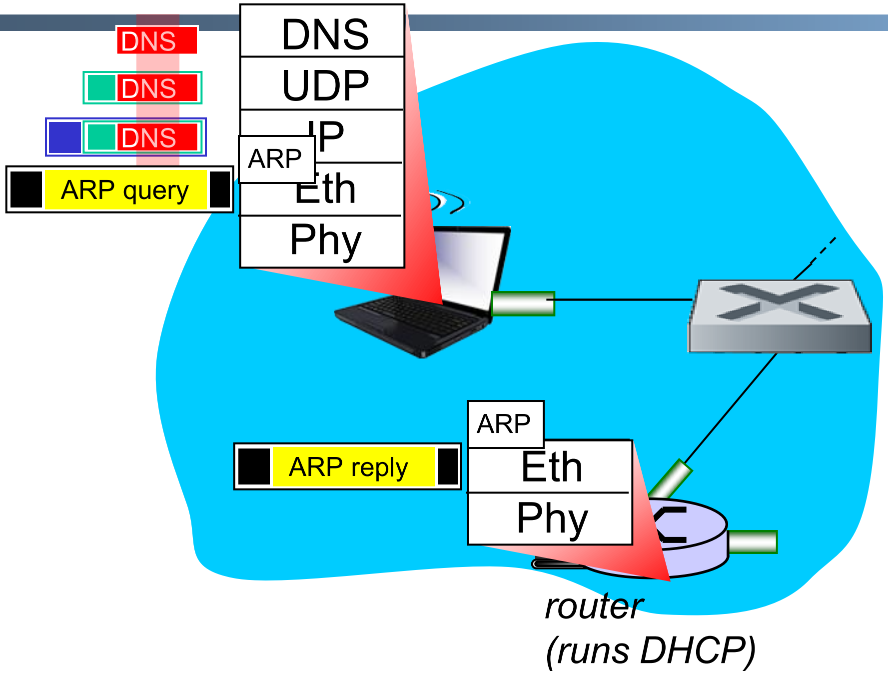
A Day in the Life… Connecting to the Internet



- 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

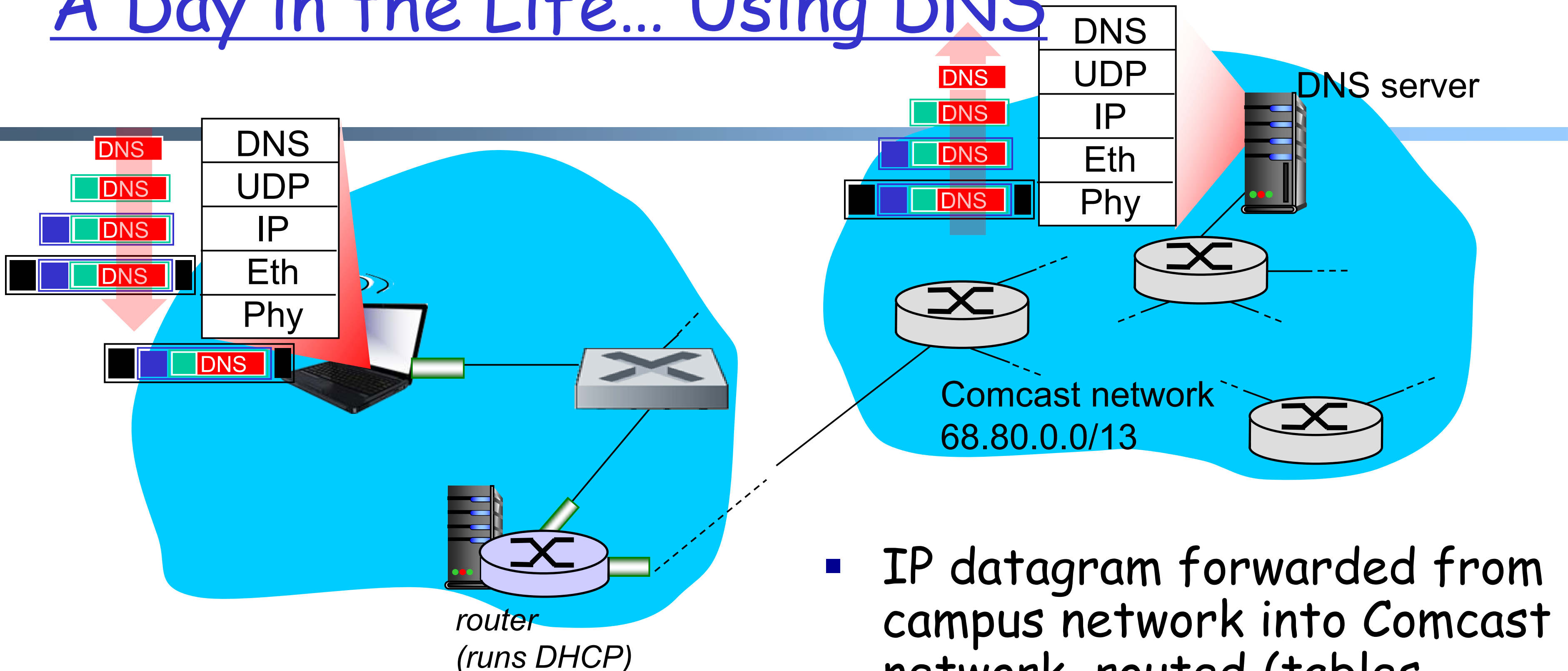
Client now has IP address, knows name & addr of DNS server, IP address of its first-hop router

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



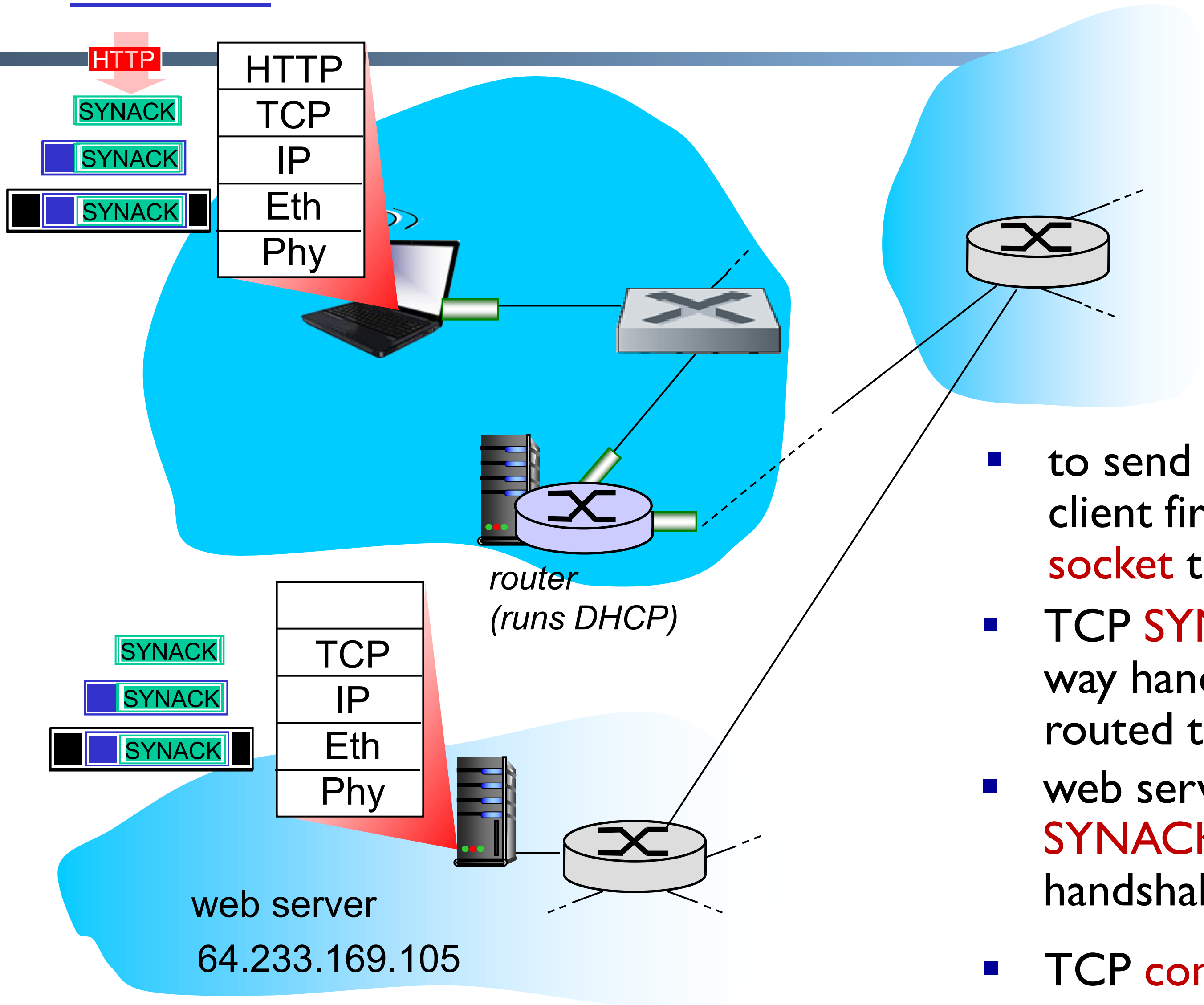
- before sending *HTTP* request, need IP address of www.tencent.com: *DNS*
- DNS query created, encapsulated in UDP, encapsulated in IP, encapsulated in Eth. To send frame to router, need MAC address of router interface: *ARP*
- *ARP query* broadcast, received by router, which replies with *ARP reply* 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
- demuxed to DNS server
- DNS server replies to client with IP address of www.tencent.com

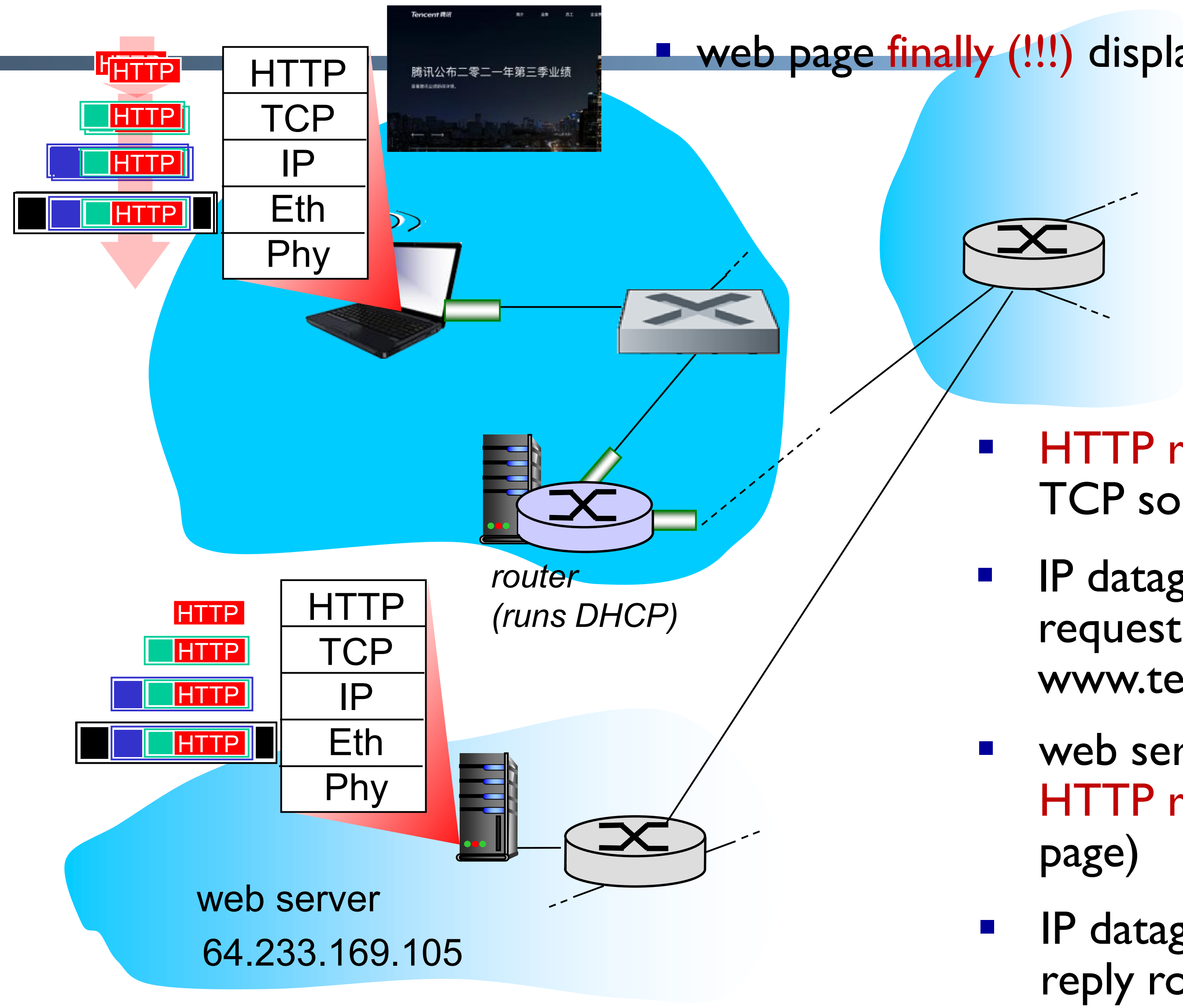
A Day in the Life...TCP Connection Carrying HTTP



- 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.tencent.com
- web server responds with HTTP reply (containing web page)
- IP datagram containing HTTP reply routed back to client

Course Topics Summary

- The Internet is a general-purpose, large-scale, distributed computer network
- Major design features/principles
  - packet switching/statistical multiplexing
    - time-reversibility, queueing theory and performance analysis
  - layered architecture, hour-glass architecture
    - end-to-end principle
  - decentralized (social-technocal) architecture
    - e.g., DNS (hierarchy delegation), interdomain routing (peer-to-peer)
  - resource allocation framework
    - axiom-based design (NBS); optimization decomposition through duality
  - adaptive control
    - e.g., sliding window self clocking, AIMD adaptation, Ethernet exp backoff
  - tradeoff between theoretical impossibility and practice

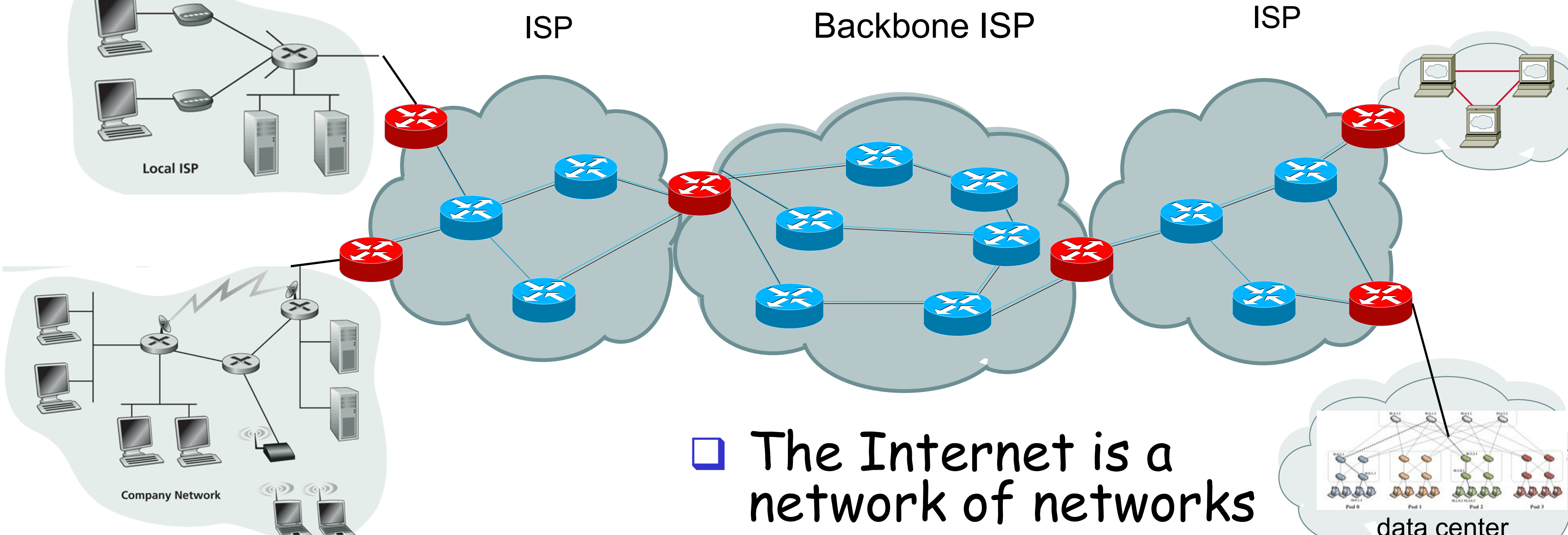
First-Day Class: What is a Network Protocol?

- A **network protocol** defines the format and the order of messages exchanged between two or more communicating entities, as well as the actions taken on the transmission and/or receipt of a message or other events.

Protocols that we have touched on?

First-Day Class: Internet Physical Infrastructure

Residential access
- Cable, Fiber, DSL, Wireless



Campus access, e.g.,
- Ethernet
- Wireless

- The Internet is a network of networks
- Each individually administrated network is called an Autonomous System (AS)

https://www.google.com/loon/how/

First-Day Class: General Complexity

- Complexity in highly organized systems arises primarily from design strategies intended to create robustness to uncertainty in their environments and component parts.
  - Scalability is robustness to changes to the size and complexity of a system as a whole.
  - Evolvability is robustness of lineages to large changes on various (usually long) time scales.
  - Reliability is robustness to component failures.
  - Efficiency is robustness to resource scarcity.
  - Modularity is robustness to component rearrangements.

First-Day Class: Evolution

- Driven by Technology, Infrastructure, Policy, Applications (usage), and Understanding:
  - technology
    - e.g., wireless/optical communication technologies and device miniaturization (sensors)
  - infrastructure
    - e.g., cloud computing vs local computing
  - applications (usage)
    - e.g., mobile computing, content distribution, game, tele presence, sensing
  - understanding
    - e.g., resource sharing principle, routing principles, mechanism design, optimal stochastic control (randomized access)
- Complexity comes from evolution.
- Don't be afraid to challenge the foundation and redesign!