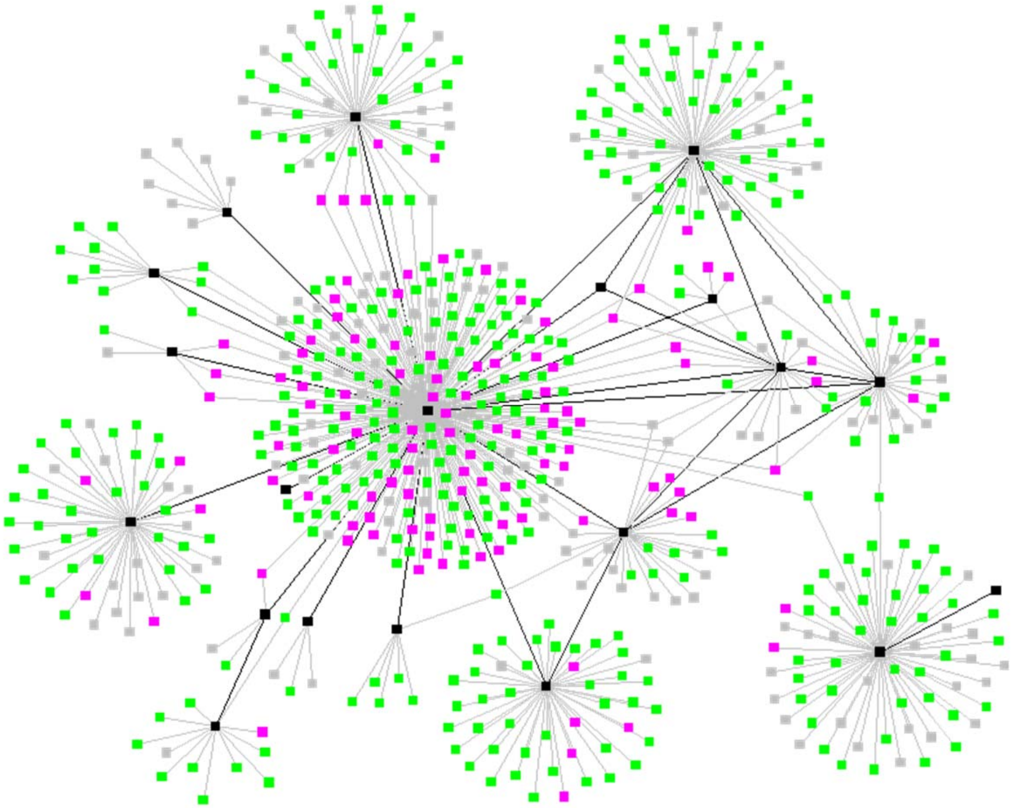
CSCE 560
Introduction to
Computer Networking



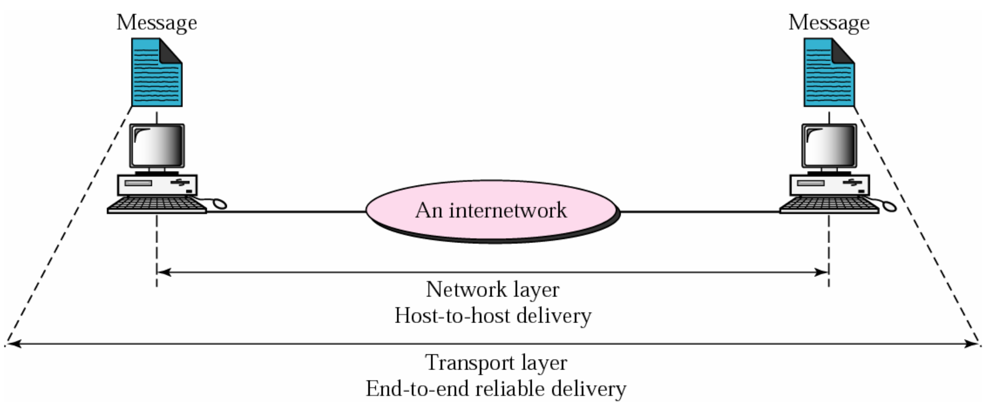
Dr. Barry Mullins
AFIT/ENG
Bldg 642, Room 209
255-3636 x7979

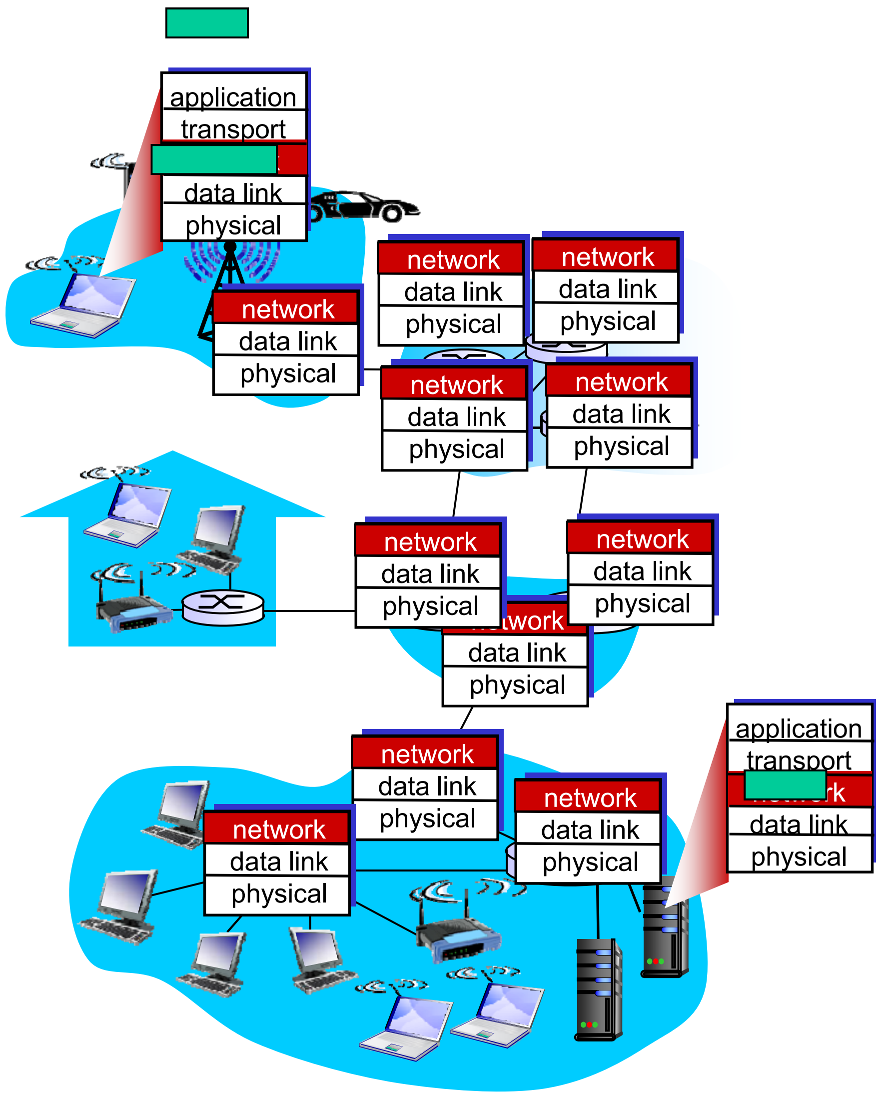
Chapter 4: Network Layer

- 4.1 Overview of Network Layer
  - Data plane
  - Control plane
- 4.2 What's inside a router
- 4.3 IP: Internet Protocol
  - Datagram format
  - Fragmentation
  - IPv4 addressing
  - Network Address Translation
  - IPv6
- 4.4 Generalized Forwarding and SDN
  - Match
  - Action
  - OpenFlow Examples of match-plus-action in action

Network Layer

- Network layer protocols in *every* host and router
- Packets between same source-dest pair can take different paths
- Network: between two hosts
  - IP to IP
- Transport: between two processes
  - Port to port



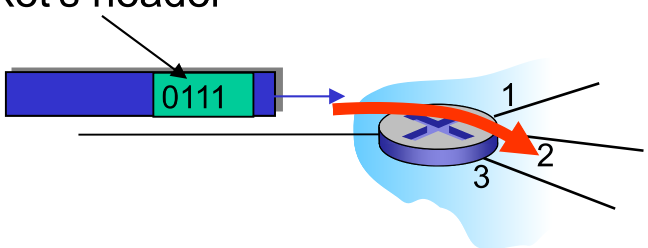


Network Layer: Data Plane, Control Plane

Data plane

- Local, per-router function
- Determines how datagram arriving on router input port is forwarded to router output port
- Forwarding function

Destination value in arriving packet's header



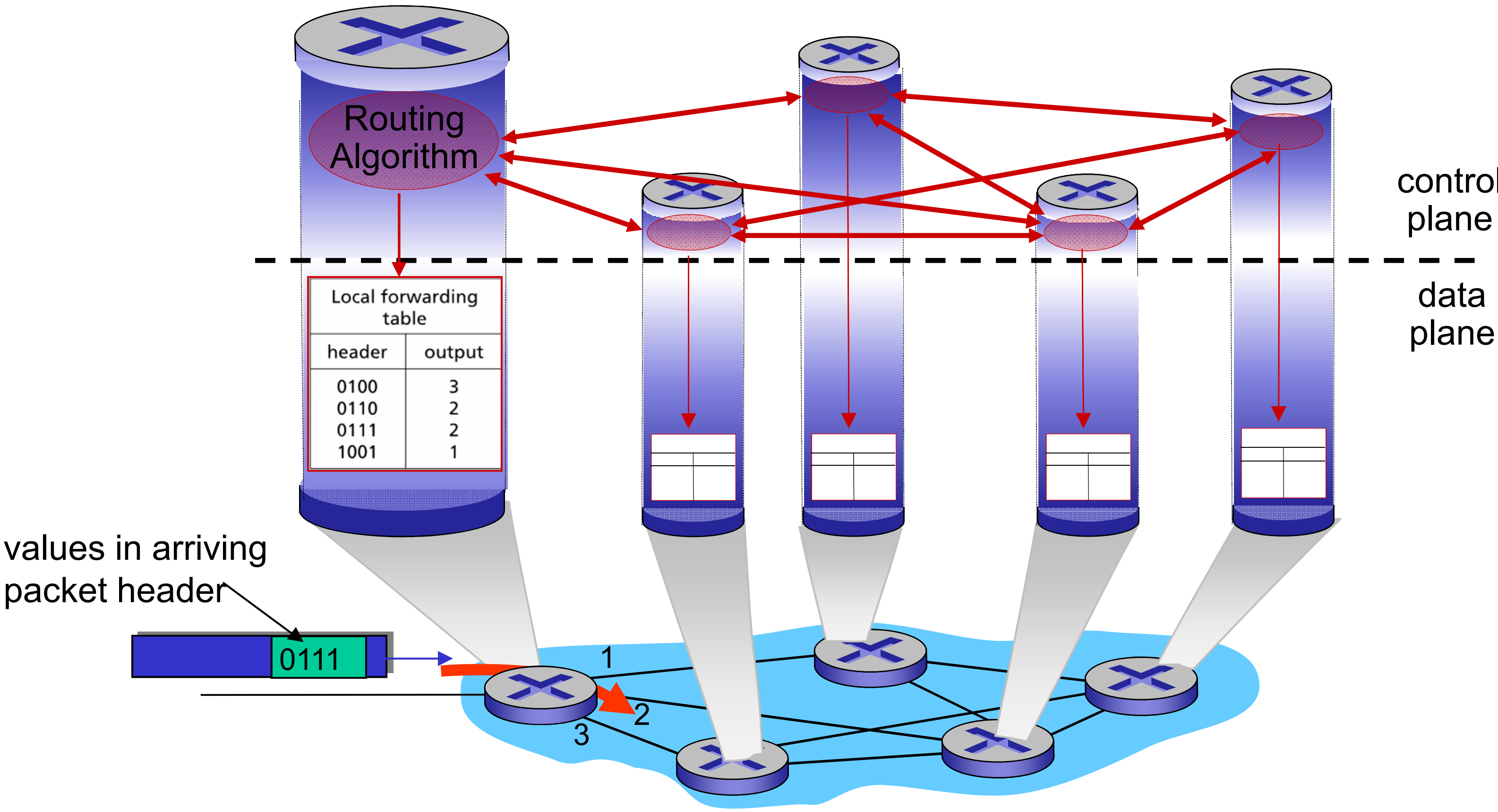
Control plane

- Network-wide logic
- Determines how datagram is routed among routers along end-end path from source host to destination host

- Two control-plane approaches:
  - Traditional routing algorithms
    - Implemented in routers
  - Software-Defined Networking (SDN)
    - Implemented in (remote) servers

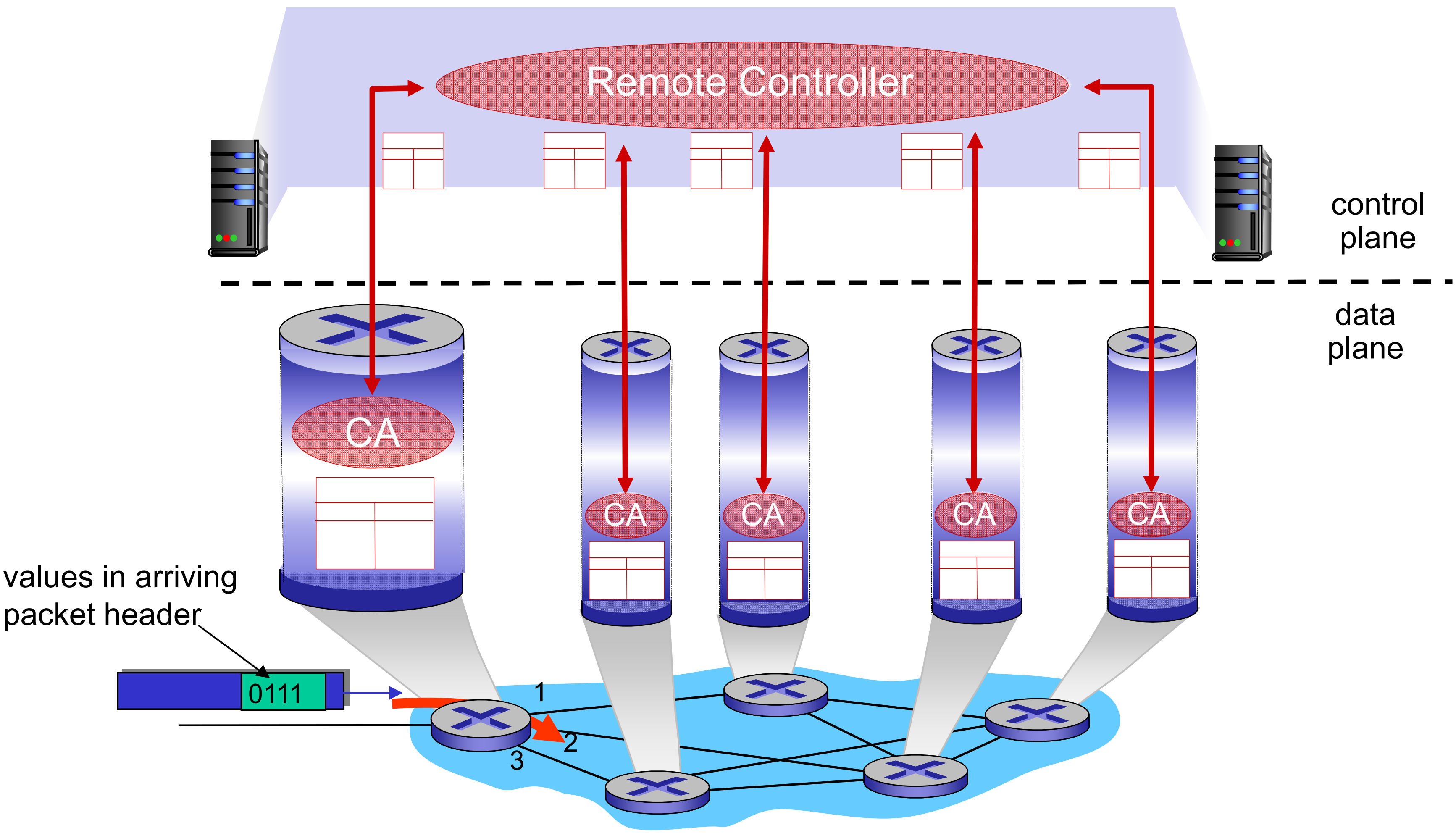
Traditional: Per-Router Control Plane

- Individual routing algorithm components in each and every router interact in the control plane to compute forwarding tables



SDN: Logically Centralized Control Plane

- A distinct (typically remote) controller interacts with local control agents (CAs)

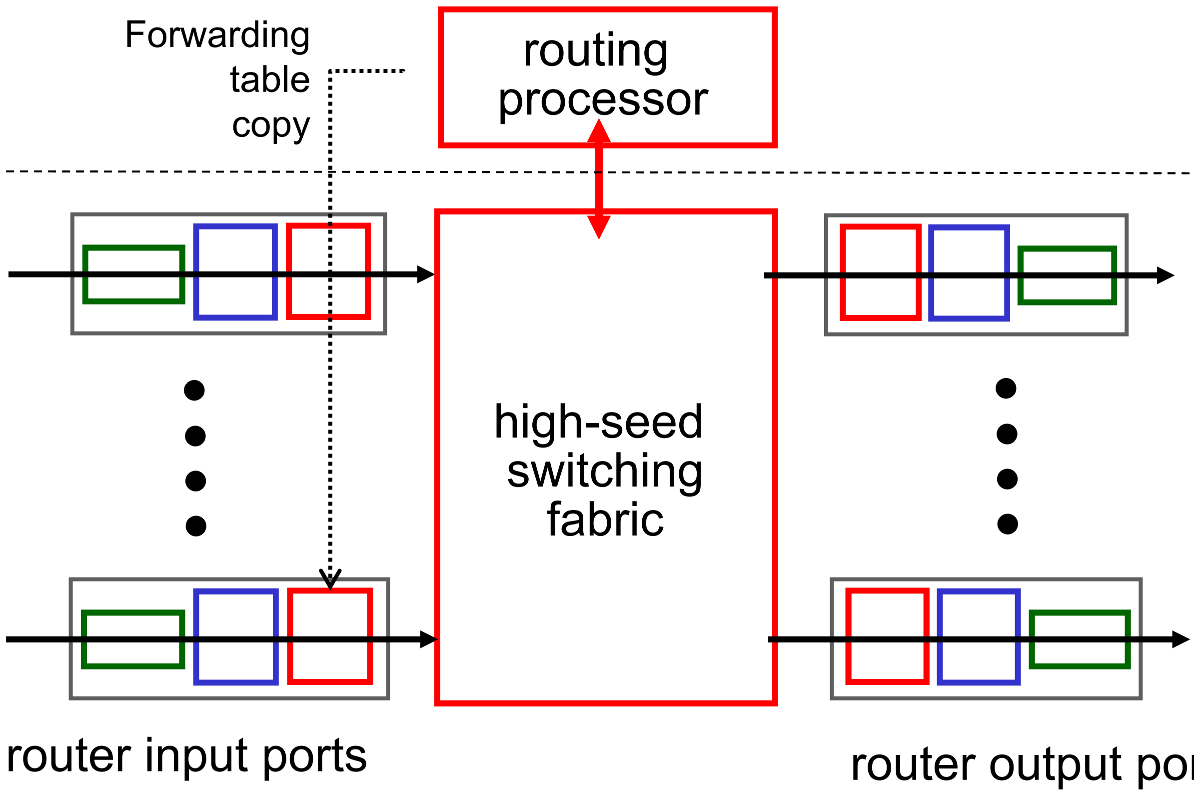


Chapter 4: Network Layer

- 4.1 Overview of Network Layer
  - Data plane
  - Control plane
- 4.2 What's inside a router
- 4.3 IP: Internet Protocol
  - Datagram format
  - Fragmentation
  - IPv4 addressing
  - Network Address Translation
  - IPv6
- 4.4 Generalized Forwarding and SDN
  - Match
  - Action
  - OpenFlow Examples of match-plus-action in action

Router Architecture Overview

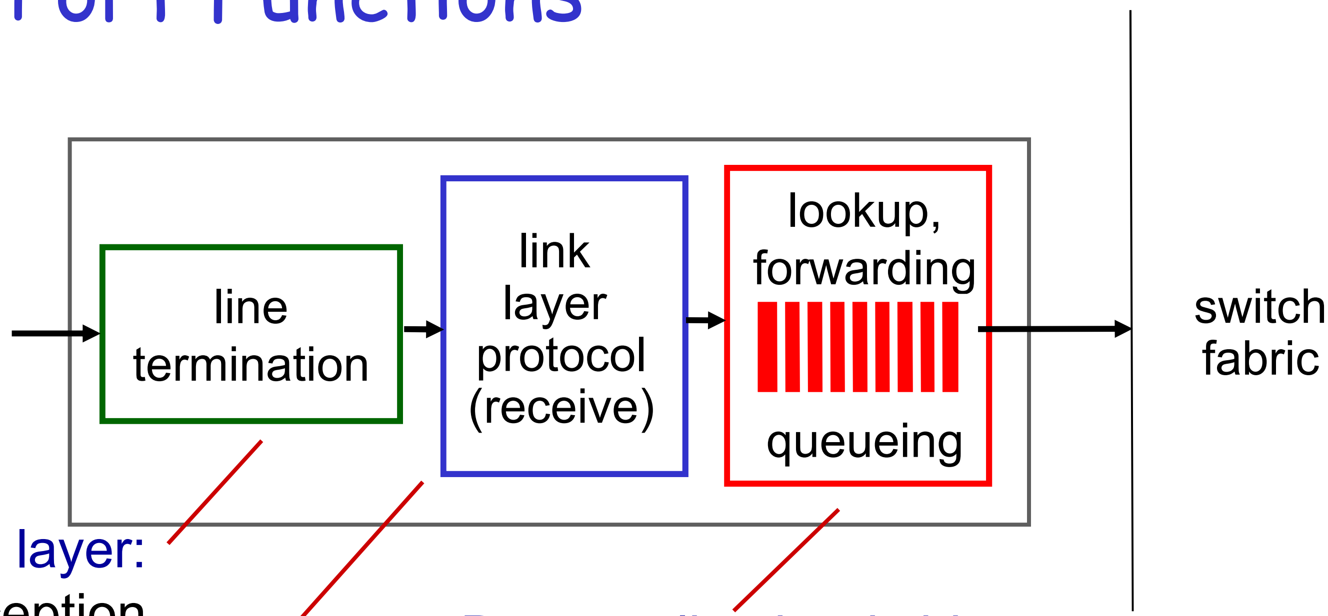
- High-level view of generic router architecture



Routing, management
control plane
(software)
operates in
millisecond / seconds

Forwarding
data plane
(hardware)
operates in
nanoseconds

Input Port Functions



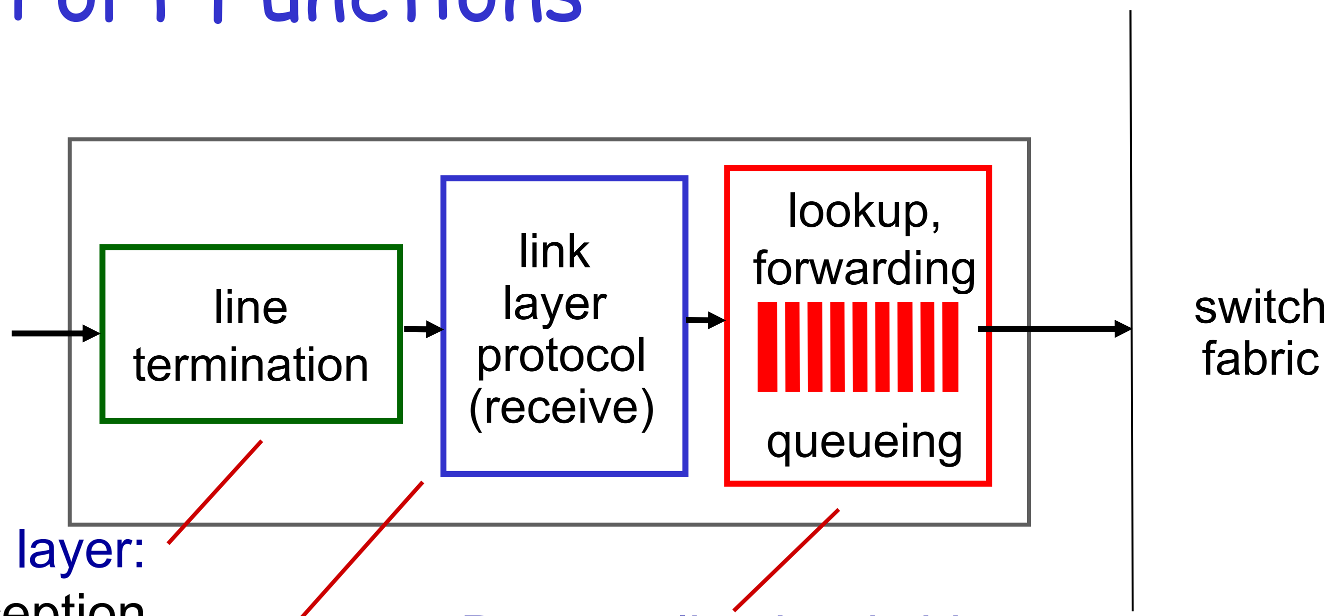
Physical layer:
bit-level reception

Data link layer:
e.g., Ethernet
(Chapter 6)

Decentralized switching:

- Using header field values, lookup output port using forwarding table in input port memory *("match plus action")*
- Goal: complete input port processing at 'line speed'
- Queuing: if datagrams arrive faster than forwarding rate into switch fabric

Input Port Functions



Physical layer:
bit-level reception

Data link layer:
e.g., Ethernet
(Chapter 6)

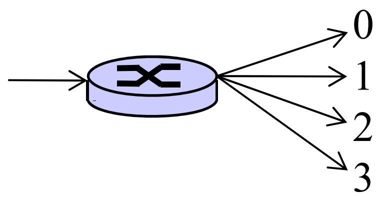
Decentralized switching:

- Destination-based forwarding (traditional)
  - Forward based only on destination IP address
- Generalized forwarding (SDN)
  - Forward based on any set of header field values

Datagram Forwarding Table

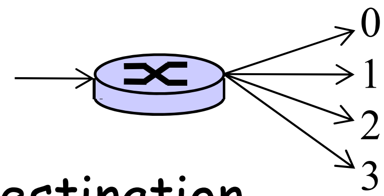
4 billion IP addresses

Rather than list individual destination addresses, list *range* of addresses (aggregate table entries)



	Destination Address Range	Link Interface
200.23.16.0 through 200.23.23.255	11001000 00010111 00010000 00000000 through 11001000 00010111 00010111 11111111	0
200.23.24.0 through 200.23.24.255	11001000 00010111 00011000 00000000 through 11001000 00010111 00011000 11111111	1
200.23.25.0 through 200.23.31.255	11001000 00010111 00011001 00000000 through 11001000 00010111 00011111 11111111	2
	Otherwise	3

Longest Prefix Matching



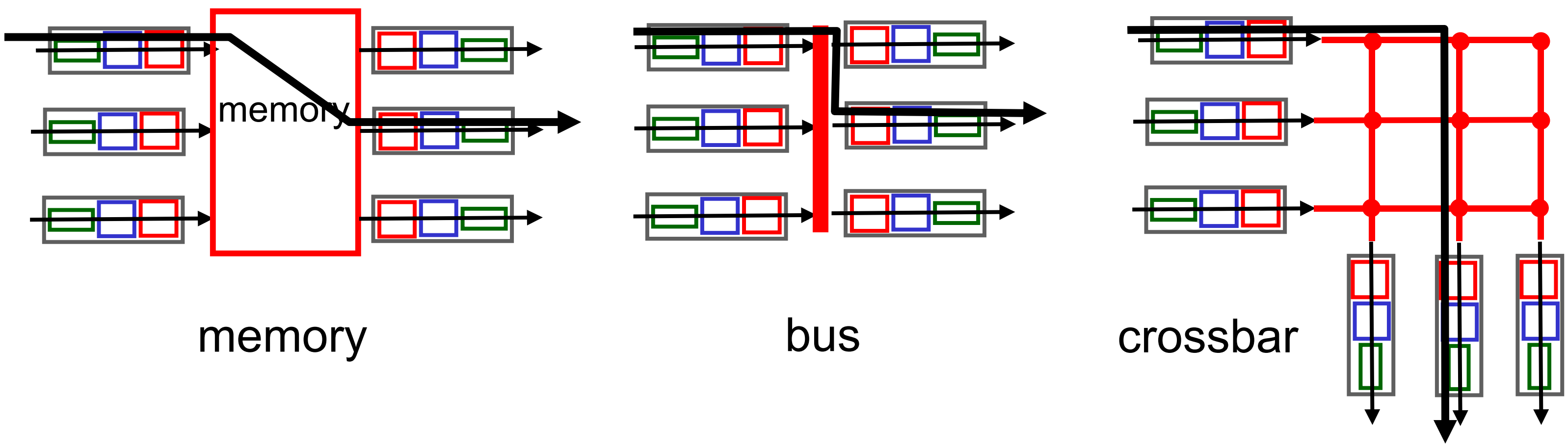
- When looking for forwarding table entry for given destination address, use longest address prefix that matches destination address

Prefix Match	Link Interface
11001000 00010111 00010	0
11001000 00010111 00011000	1
11001000 00010111 00011	2
Otherwise	3

- Examples: Which interface will this router send the datagram with the destination address listed?
  - 11001000 00010111 00010110 10100001
  - 11001000 00010111 00011000 10101010
  - 11001000 10010111 00011000 10101010
  - 11001000 00010111 00011010 10101010

Switching Fabrics

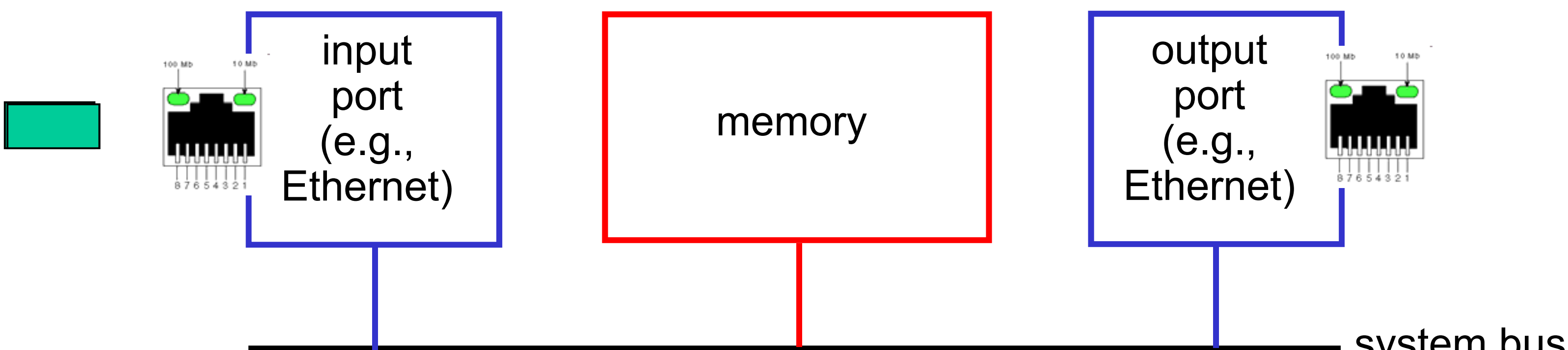
- Transfer packet from input buffer to appropriate output buffer
- Three types of switching fabrics



Switching Via Memory (Simplest)

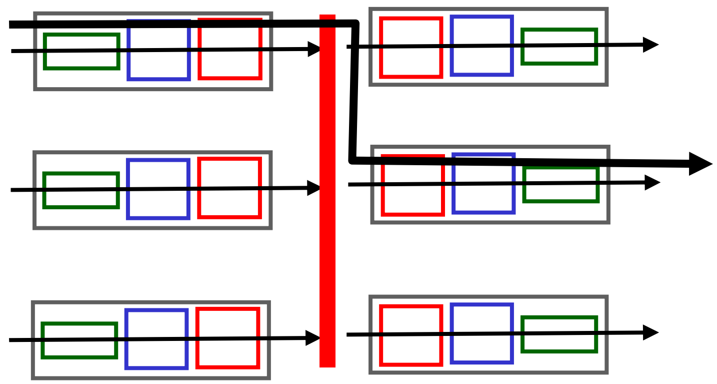
First generation routers:

- Traditional computers with switching under direct control of CPU
- Packet copied to system's memory
- Speed limited by memory bandwidth (B)
    - 2 bus crossings per datagram → forwarding thruput = B/2



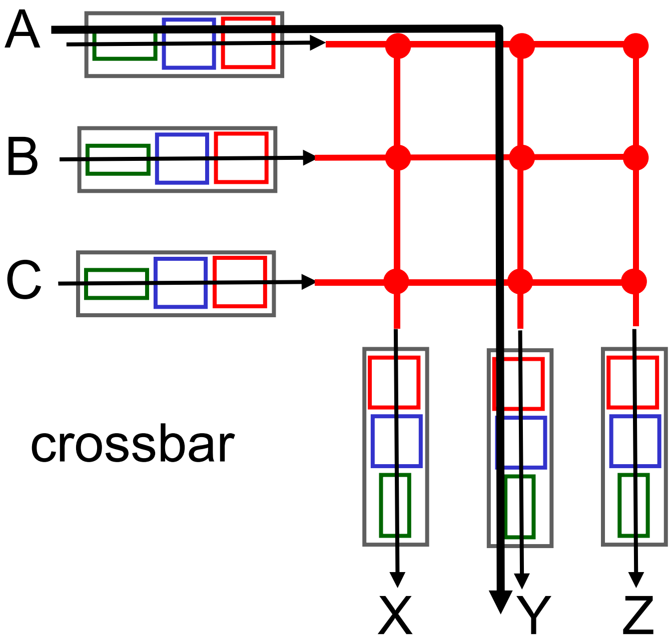
Switching Via a Bus

- Datagram transferred input port to output port via a shared bus
  - Switch adds switch-internal header
  - All output ports receive packet but only correct port keeps it
- Only one packet crosses the bus at a time
- 32 Gbps bus, Cisco 5600: sufficient speed for access and enterprise routers (not regional or backbone)



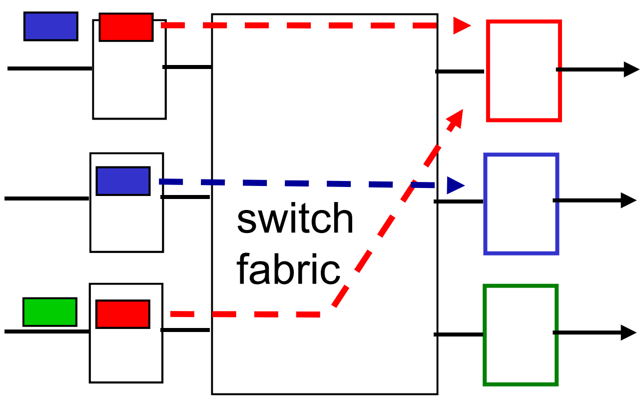
Switching Via a Crossbar Network

- Offers more than one bus simultaneously
- Advanced design: fragmenting datagram into fixed length cells, switch cells through the fabric
- Cisco 12000 series: switches 60 Gbps through the interconnection network

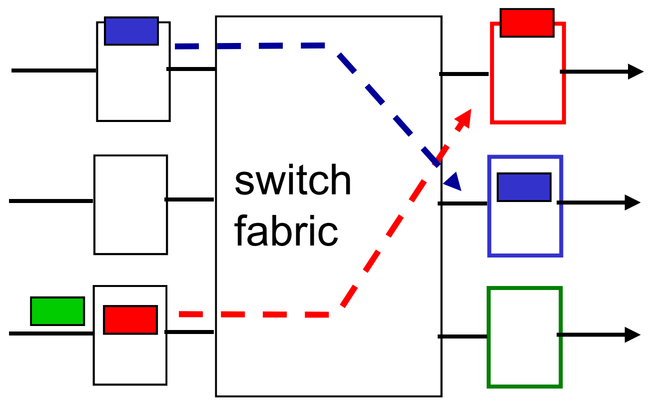


Input Port Queuing

- If fabric slower than input ports combined → queuing may occur at input queues
  - Queuing delay and loss due to input buffer overflow!
- Head-of-the-Line (HOL) blocking: queued datagram at front of queue prevents others in queue from moving forward

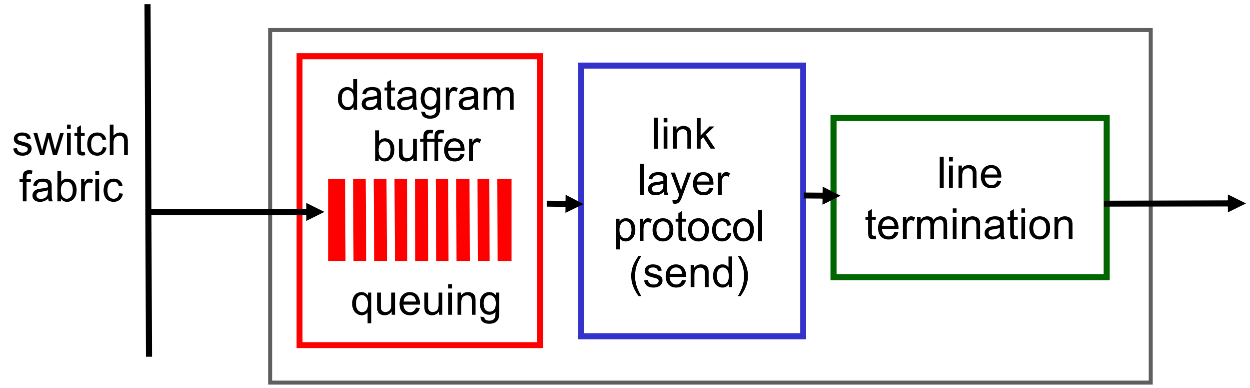


Output port contention:
only one red datagram can be transferred.
lower red packet is blocked

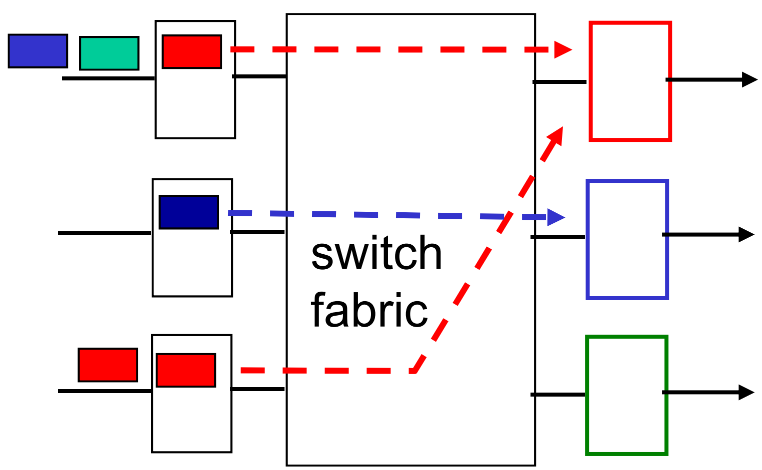


One packet time later: green packet experiences HOL blocking

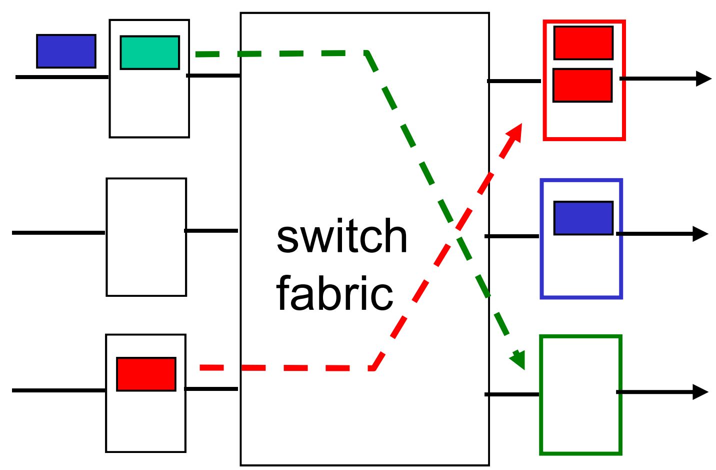
Output Ports



- Buffering required when datagrams arrive from switching fabric faster than the transmission rate
- Scheduling discipline chooses among queued datagrams for transmission
- Queuing (delay) and loss due to output port buffer overflow!



Output port contention at time *t*



One packet time later –
Assumes switch operates at 2X line speed

Chapter 4: Network Layer

- 4.1 Overview of Network Layer
  - Data plane
  - Control plane
- 4.2 What's inside a router
- 4.3 IP: Internet Protocol
  - Datagram format
  - Fragmentation
  - IPv4 addressing
  - Network Address Translation
  - IPv6
- 4.4 Generalized Forwarding and SDN
  - Match
  - Action
  - OpenFlow Examples of match-plus-action in action

IP Datagram Format

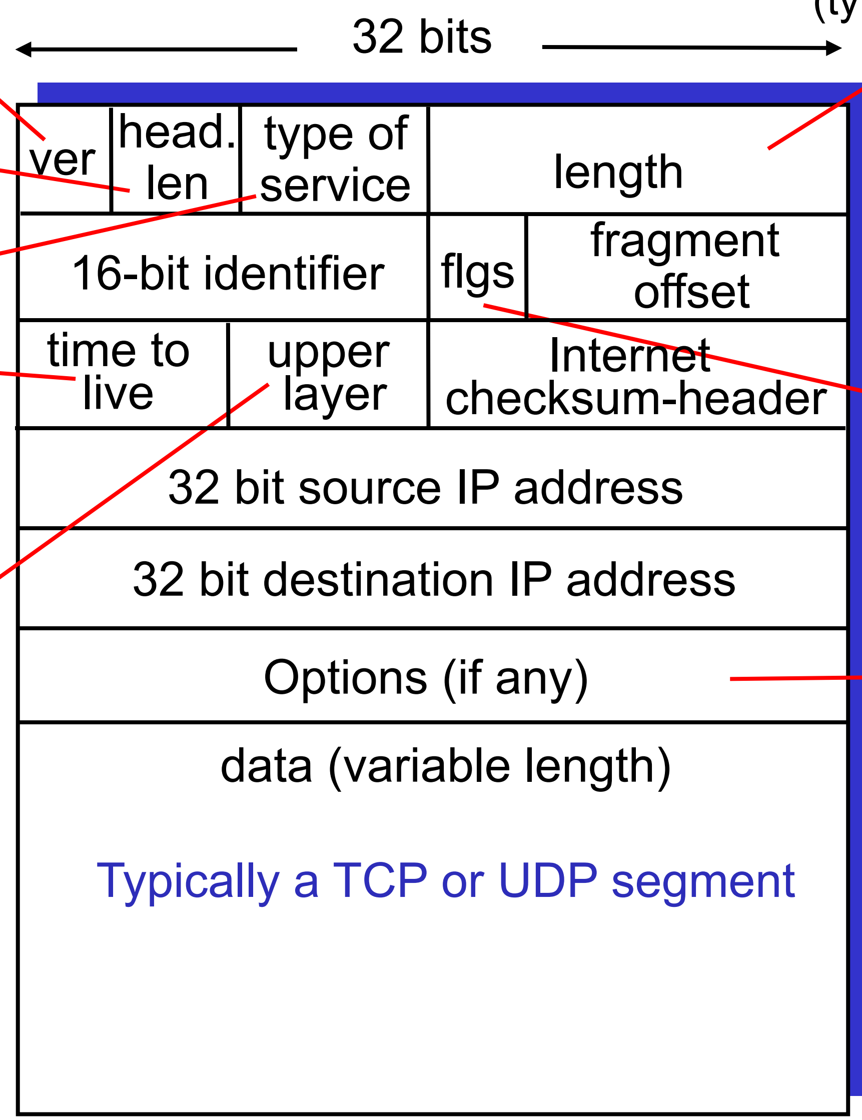
IP protocol version number

Header length (in 32-bit words)

"Type" of data –
could be used to provide QOS

Max number remaining hops (decremented at each router)

Upper layer protocol to deliver payload to (demuxing)
TCP = 6, ICMP = 1,
UDP = 17… (in decimal)



Total datagram length header + data in bytes (typically 1500)

This row for fragmentation/ reassembly

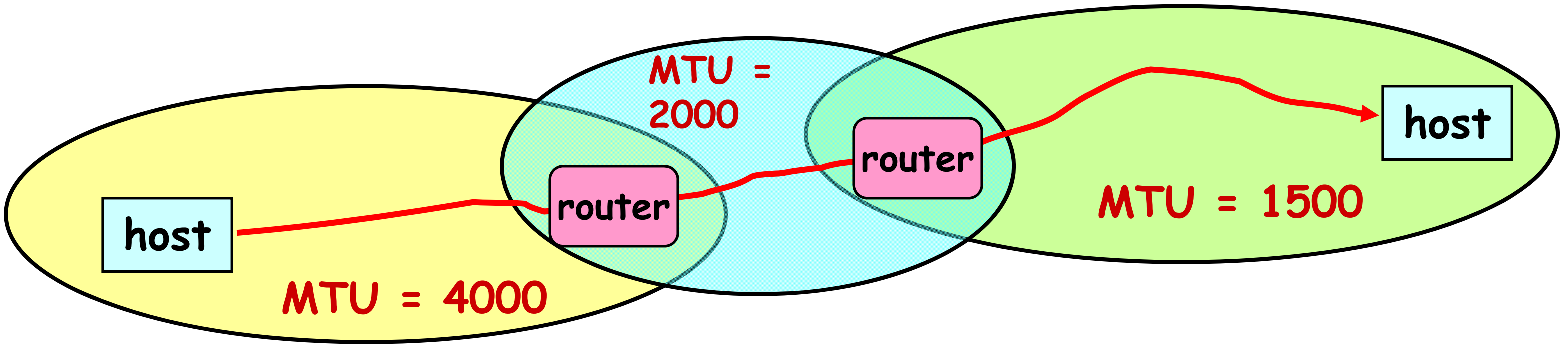
3 bits:
- 0 (always 0)
- Don't fragment
- More fragments

e.g., timestamp, record route taken, specify list of routers to visit

How much overhead so far?

- 20 bytes of TCP
- 20 bytes of IP
- = 40 bytes + app layer overhead (if any)

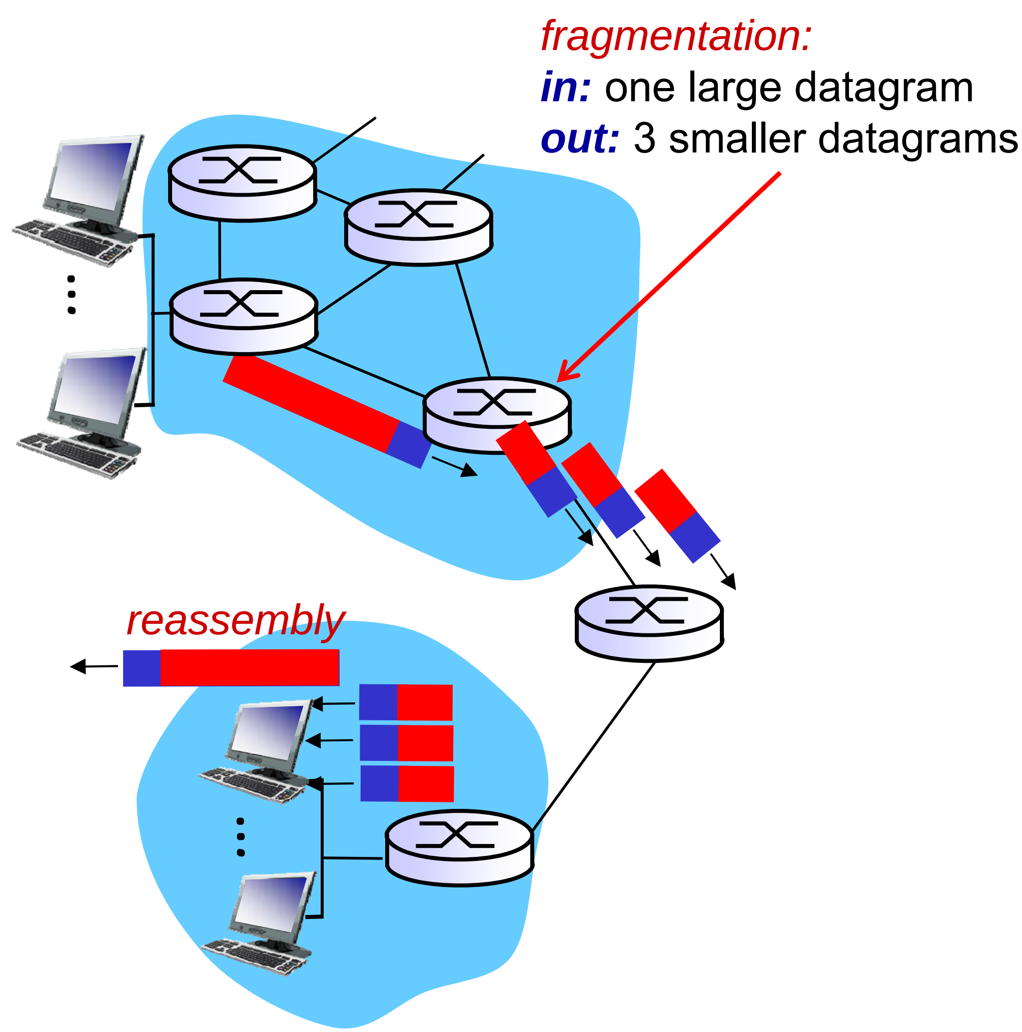
IP Fragmentation



- Every network has own Maximum Transmission Unit (MTU)
  - Largest IP datagram it can carry within its own packet frame
    - Ethernet is 1500 bytes
  - We typically don't know MTUs of all intermediate networks in advance
- IP Solution
  - When a large datagram hits a network with a smaller MTU
    - Fragment the datagram
    - Fragmented datagram can be further fragmented as it proceeds farther

IP Fragmentation and Reassembly

- Large IP datagram divided ("fragmented") within net
- One datagram becomes several datagrams
- "Reassembled" only at final destination
- IP header bits used to identify, order related fragments

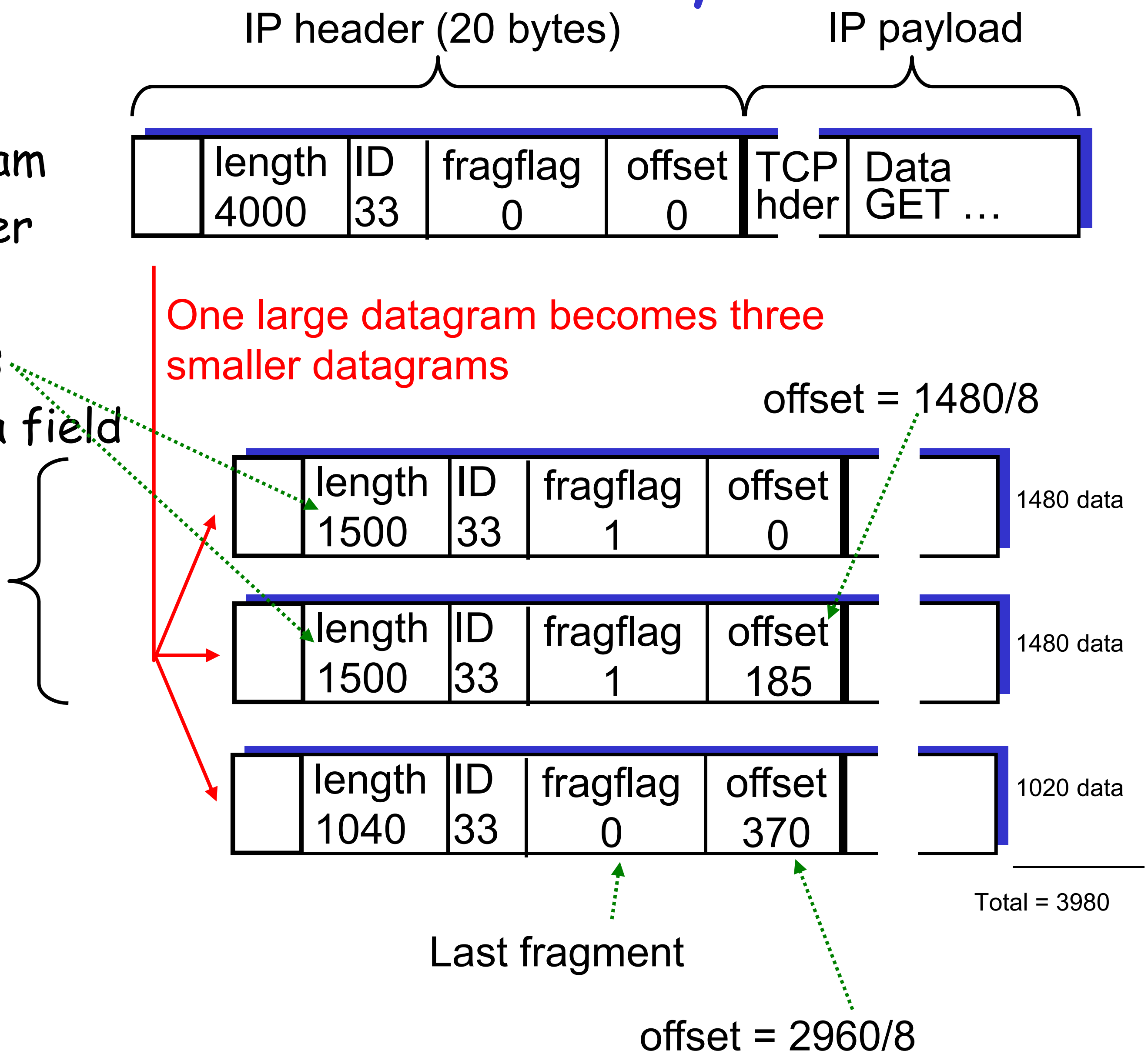


IP Fragmentation and Reassembly

Example

- 4000 byte datagram
  - 20 bytes header
  - 3980 payload
- MTU = 1500 bytes
- 1480 bytes in data field

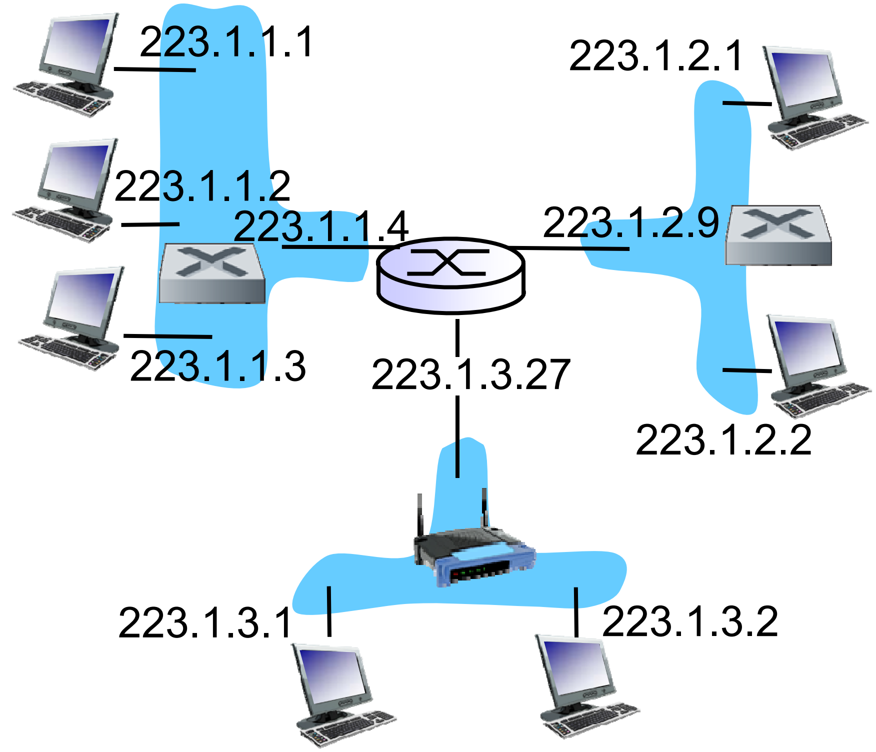
Two additional IP headers added



Total bytes sent = 4040

IP Addressing: Introduction

- IP address: 32-bit identifier for host, router interface
- Interface: connection between host/router and physical link
  - Routers typically have multiple interfaces
  - Host typically has one or two interfaces
- IP addresses associated with each interface
- Uses dotted-decimal notation



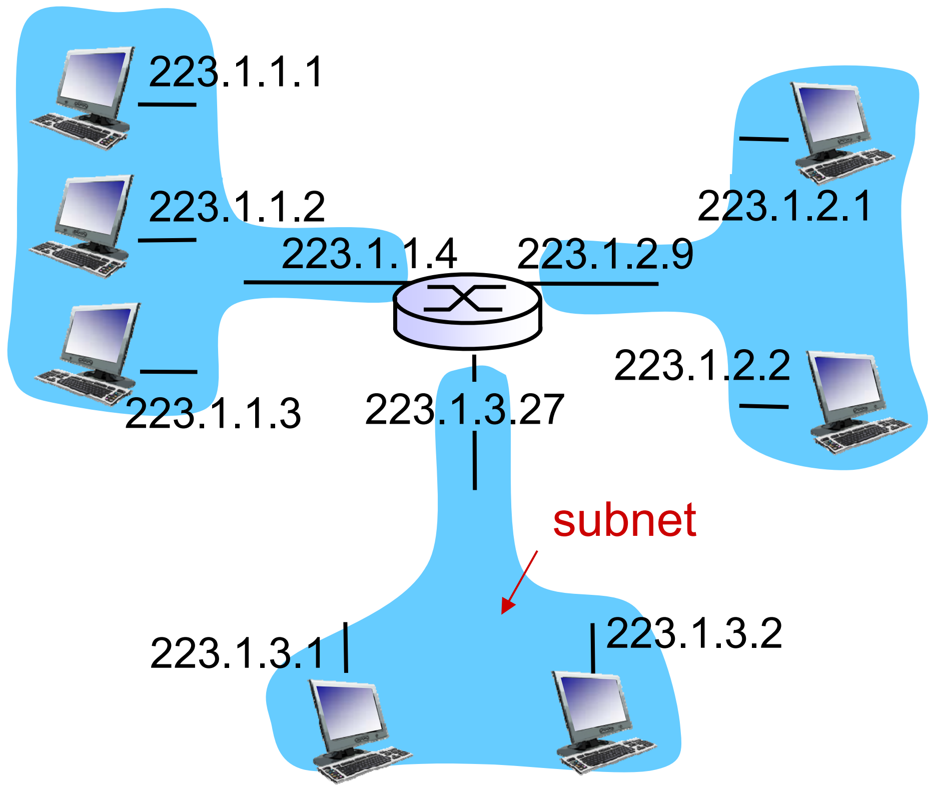
$$223.1.3.2 = \underbrace{11011111}_{223}\ \underbrace{00000001}_{1}\ \underbrace{00000011}_{3}\ \underbrace{00000010}_{2}$$

Subnets

- IP address:
  - Subnet part (high order bits)
  - Host part (low order bits)

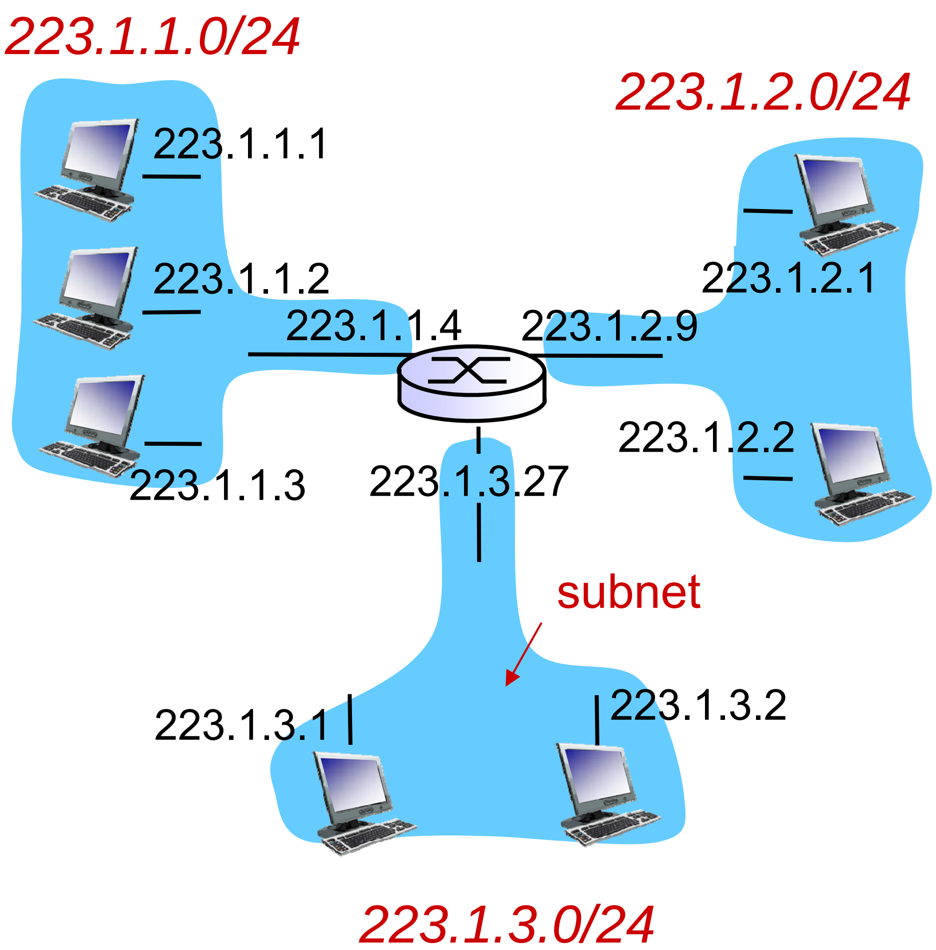
$$\underbrace{223.1.3}_{\text{Subnet}}.\underbrace{2}_{\text{Host}}$$

- *What's a subnet ?*
  - Device interfaces with same subnet part of IP address
  - Can physically reach each other without intervening router



Network consisting of 3 subnets

Subnets



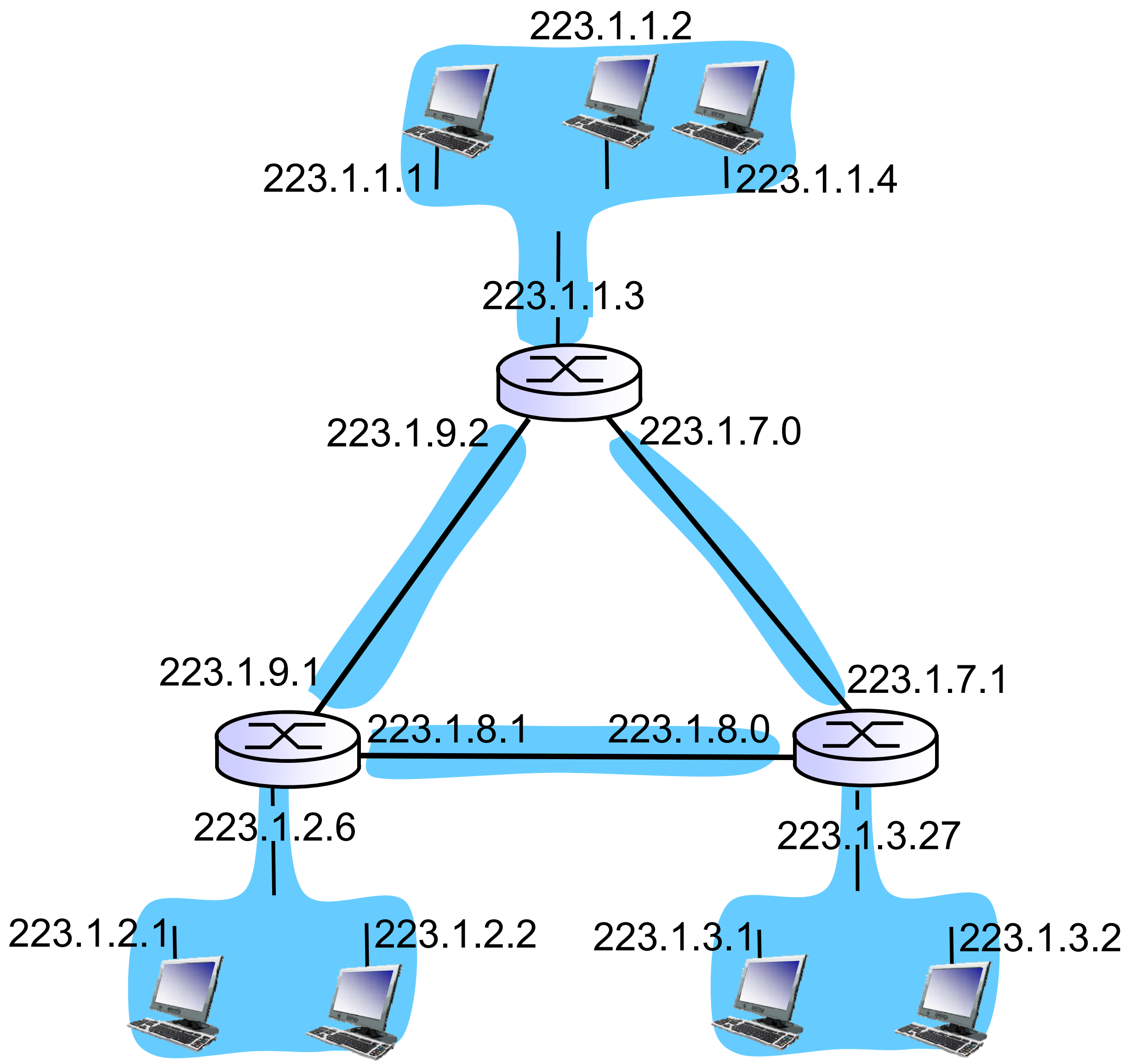
Recipe

- To determine the subnets, detach each interface from its host or router, creating islands of isolated networks
- Each isolated network is called a subnet

Subnet mask:
/24
255.255.255.0

Subnets

How many?



Classful Addressing (A Bit☺of History)

- Hierarchical 32-bit addresses identify a connection to a network:
  - Network (site)
  - Host
- Five forms of IP addresses:

1st Octet		0 1 2 3 4 8 16 24 31	
0 - 127	*Class A*	0 | netid | hostid	16M hosts
128 - 191	*Class B*	1 | 0 | netid | hostid	65K hosts
192 - 223	*Class C*	1 | 1 | 0 | netid | hostid	256 hosts
224 - 239	*Class D*	1 | 1 | 1 | 0 | multicast address	
240 - 247	*Class E*	1 | 1 | 1 | 1 | 0 | reserved for future use	

IP Addressing: CIDR

CIDR: Classless InterDomain Routing

- Subnet portion of address of arbitrary length
- Address format: a.b.c.d/x
  - where x is # bits in subnet portion of address
  - x is often called the prefix

subnet part | host part

11001000 00010111 00010000 00000000

200.23.16.0/23

Special IP Addresses

- In general, all 1's means "any" and all 0's mean "this"

- Directed broadcast
  - Network (or subnet) + all 1's hostid
  - 129.92.102.255

- Limited broadcast: all 1's (network and hostid) = 255.255.255.255
  - Used when an IP node must perform a one-to-everyone delivery on the local network but the network ID is unknown
  - Router will not forward

Special IP Addresses

- 127.0.0.1 is local loopback – technically defined as 127.0.0.0/8
- 0.0.0.0 is default route – network ID and host ID are both 0
  - "This" host on "this" network – typically same as loopback
- All host bits = 0 (e.g., 129.92.102.0), typically the network address
  - Rarely see this address in use
- Router interfaces typically end in x.x.x.1 or x.x.x.254

IP Addresses: How to Get One?

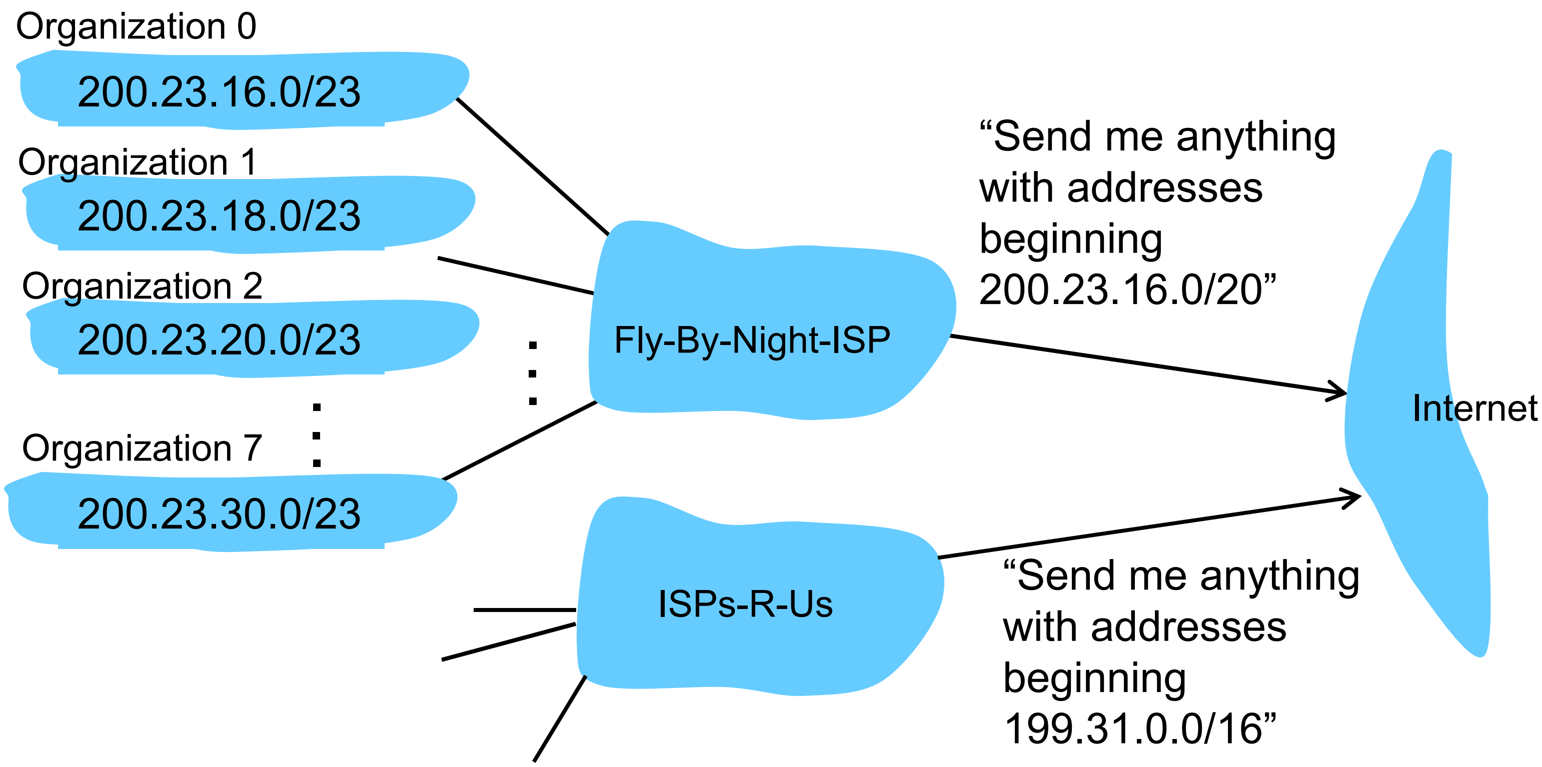
Q: How does the ***network*** get subnet part of IP addr?

A: Gets allocated a portion of its ISP's address space

ISP's block	11001000	00010111	00010000	00000000	200.23.16.0/20
Organization 0	11001000	00010111	00010000	00000000	200.23.16.0/23
Organization 1	11001000	00010111	00010010	00000000	200.23.18.0/23
Organization 2	11001000	00010111	00010100	00000000	200.23.20.0/23
Organization 3	11001000	00010111	00010110	00000000	200.23.22.0/23
...	.....			....	....
Organization 7	11001000	00010111	00011110	00000000	200.23.30.0/23

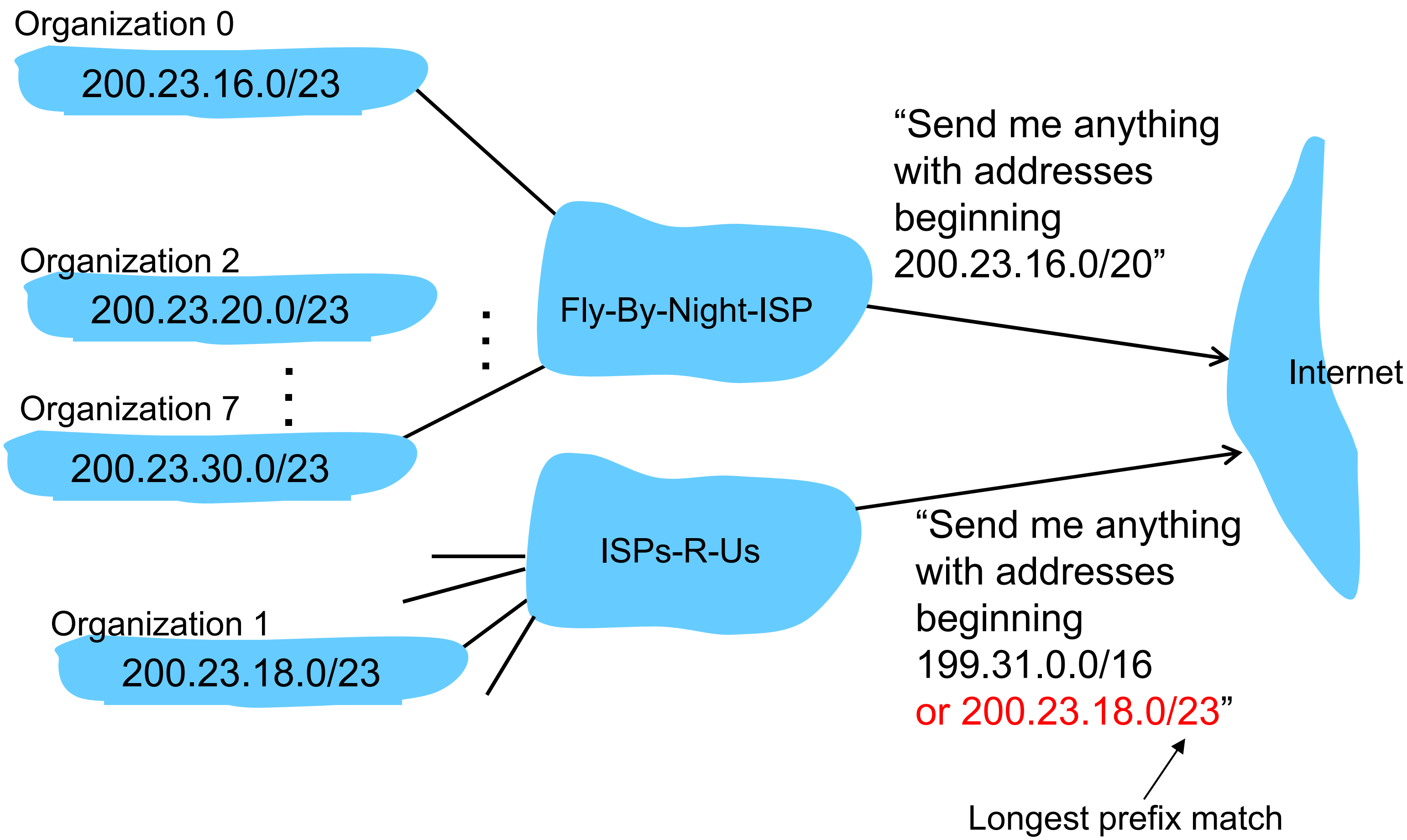
Hierarchical Addressing: Route Aggregation

Hierarchical addressing allows efficient advertisement of routing information:



Hierarchical Addressing: More Specific Routes

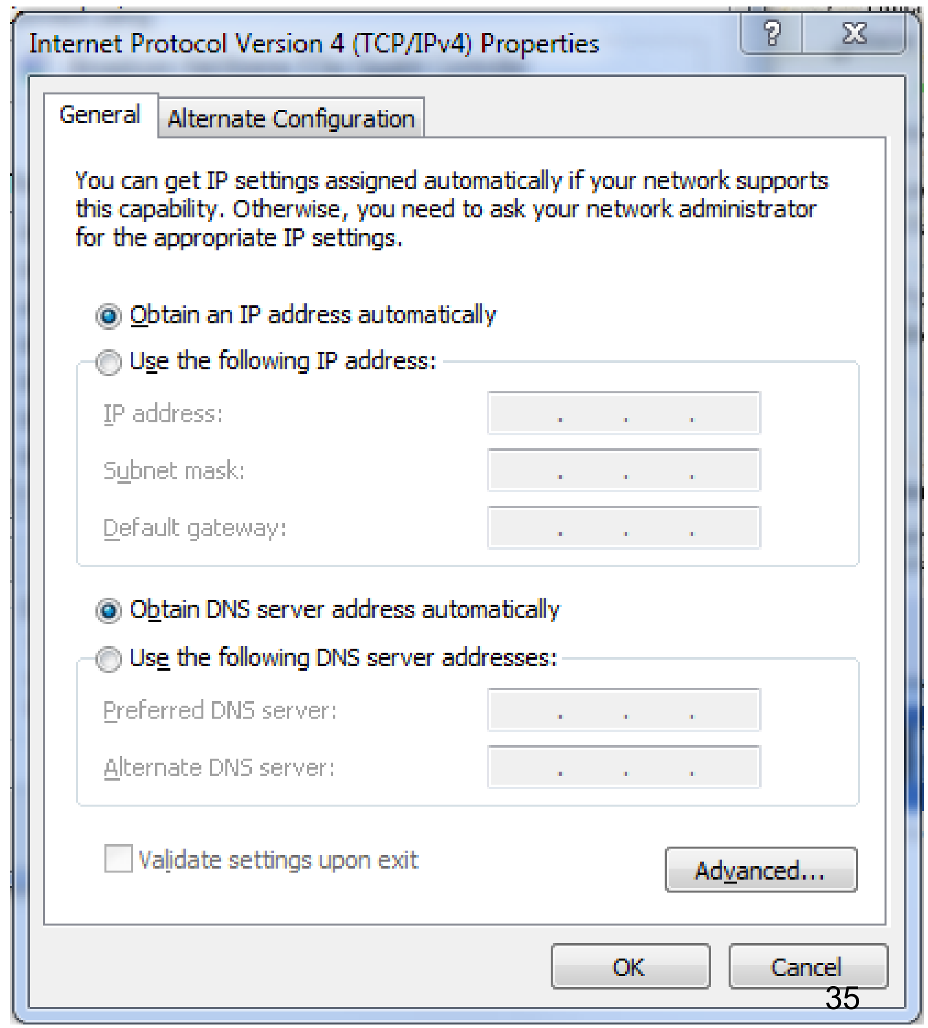
What if Fly-By-Night acquires ISPs-R-Us and moves Org 1 there?
ISPs-R-Us has a more specific route to Organization 1



IP Addresses: How to Get One?

- Q: How does the host get an IP address?
- Hardcoded by system admin in a file or via a GUI
  - Windows 7:
  - Control Panel
  - Network
  - Change adapter settings
  - Select interface
  - Properties
  - TCP/IPv4
  - Properties

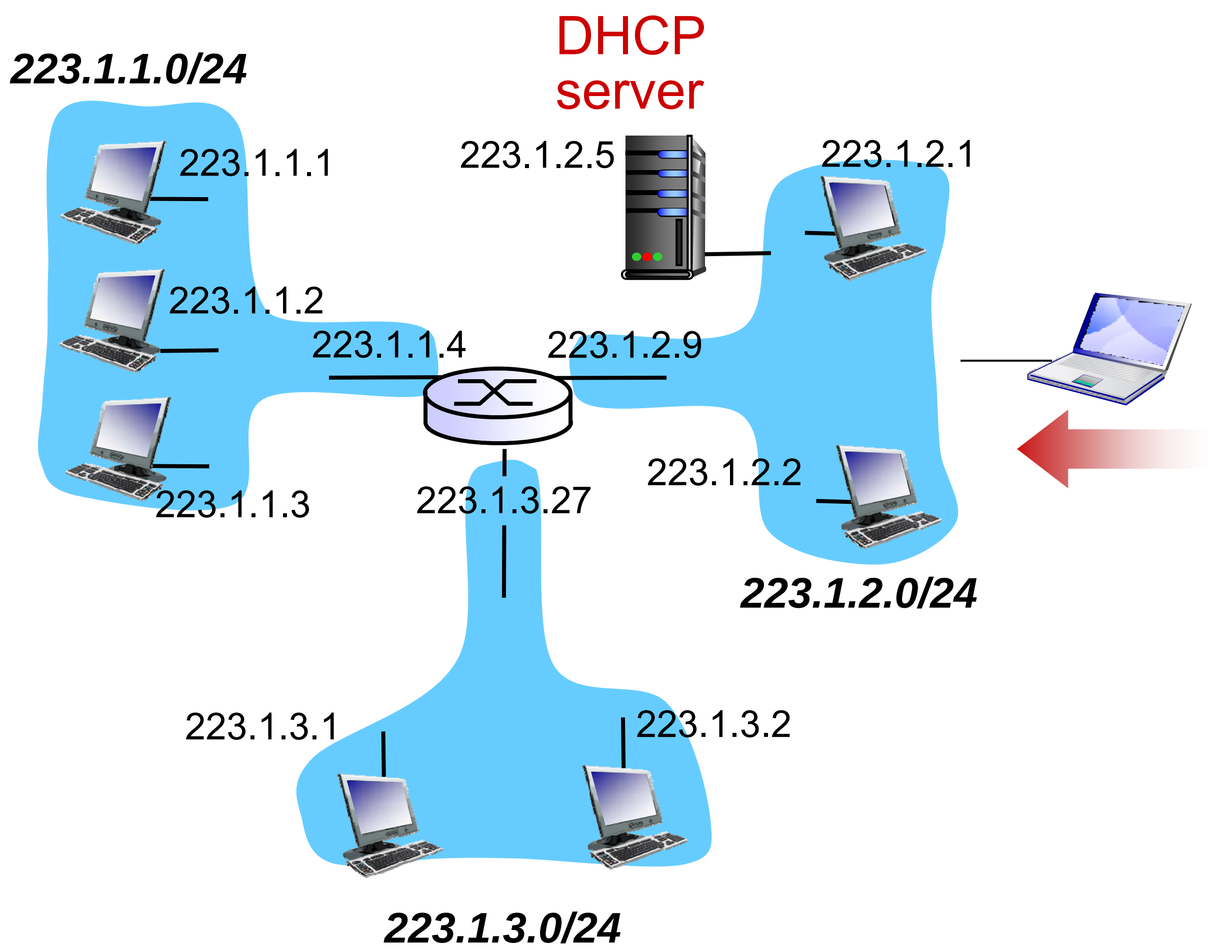
  - UNIX: /etc/rc.config
- OR...



DHCP: Dynamic Host Configuration Protocol

- Allows host to dynamically obtain its IP address from DHCP server when it joins the network
- DHCP server maintains a pool of IP addresses that are assigned on a temporary basis
  - Allows reuse of addresses
    - Only hold address while connected and “on”
  - Issues a lease on an IP address for a period of time
    - Host can renew its lease on address in use
- DHCP also provides
  - Subnet mask
  - Default gateway address
  - DNS server name and address
- “plug-and-play”
- Check out your own computer → **`ipconfig /all`**
- Support for mobile users who want to join network

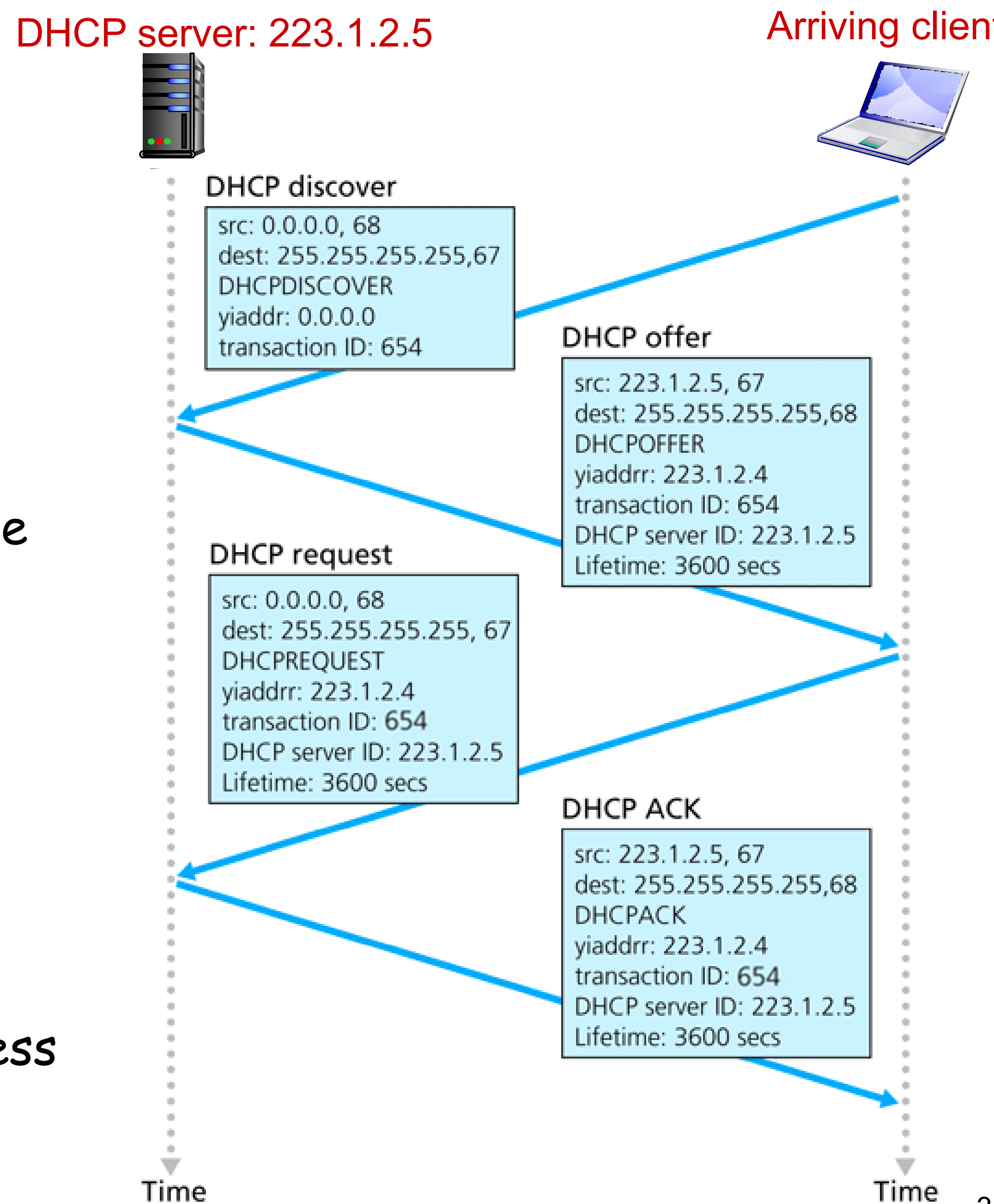
DHCP Client-server Scenario



Arriving client using DHCP needs address in this network

DHCP Scenario

- Client-server protocol
- DHCP server
  - listens on UDP 67
  - sends out to UDP 68
- DORA
- When half the lease time expires, the client broadcasts another DHCPREQUEST asking for a renewal



yiaddr: your internet address

IP Addressing: The Last Word...

Q: How does an ISP get block of addresses?

A: ICANN: Internet Corporation for Assigned Names and Numbers

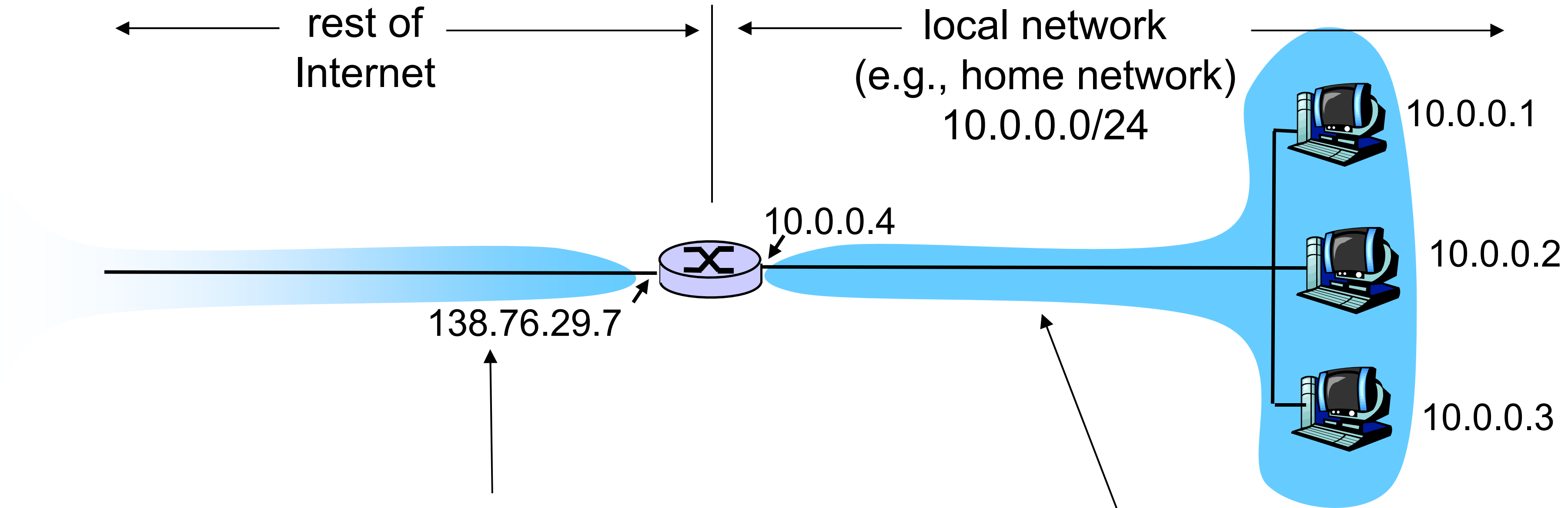
- Non-profit private corp performing IANA functions under contract with Department of Commerce
- Coordinates IANA functions which include
  - Allocates addresses to regional Internet registries
    - Internet registries handle allocation within their region
  - Manages DNS root servers
  - Assigns domain names, resolves disputes
  - Uses RFC 2050 - Internet Registry IP Allocation Guidelines

- Fun fact
  - The founder of Blackhat and DEFCON (Jeff Moss aka "Dark Tangent") is the chief security officer of ICANN

IANA - Internet Assigned Numbers Authority

NAT: Network Address Translation

Private IP Addresses
10.0.0.0 - 10.255.255.255
172.16.0.0 - 172.31.255.255
192.168.0.0 - 192.168.255.255



- *All* datagrams *leaving* local network have same source NAT IP address: 138.76.29.7 different source port numbers
- NAT-enabled router looks like a single device with one IP address

Datagrams in this network have 10.0.0.0/24 address for source, destination (as usual)

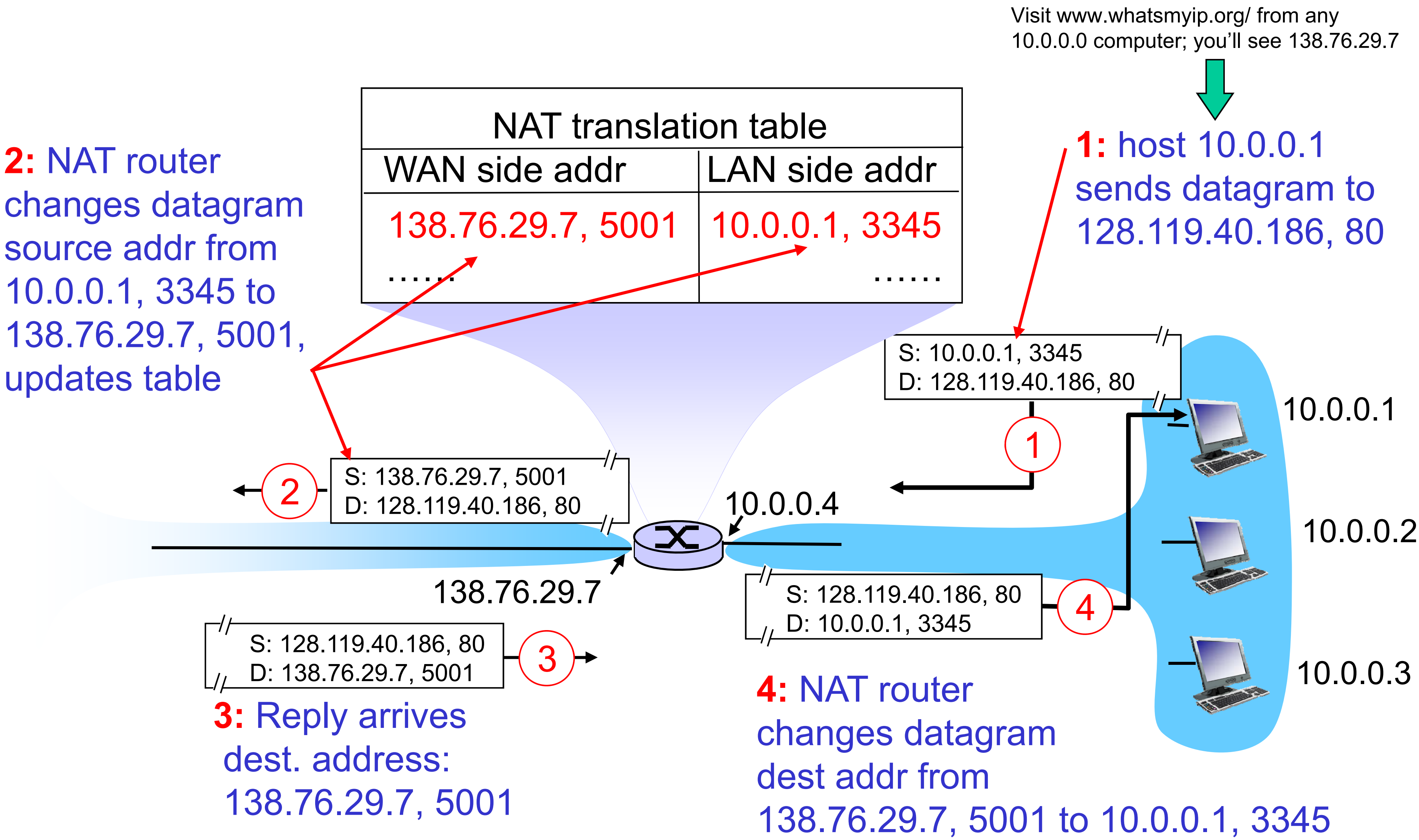
NAT Motivation

- Local network uses just one IP address as far as outside world is concerned:
  - No need to allocate range of addresses from ISP
    - One IP address is used for all devices
- Can change addresses of devices in local network without notifying outside world
- Can change ISP without changing addresses of devices in local network
- Devices inside local net not explicitly addressable
  - Not visible by outside world (a security plus!)

NAT Implementation

- NAT router must:
  - Outgoing datagrams: replace (source IP address, port #) of every outgoing datagram with (NAT IP address, new port #)
    - Remote clients/servers will respond using (NAT IP address, new port #) as destination address
  - Remember (in NAT translation table) every (source IP address, port #) to (NAT IP address, new port #) translation pair
  - Incoming datagrams: replace (NAT IP address, new port #) in dest fields of every incoming datagram with corresponding (source IP address, port #) stored in NAT table

NAT: Network Address Translation



CDN NAT addr – 209.194.136.34

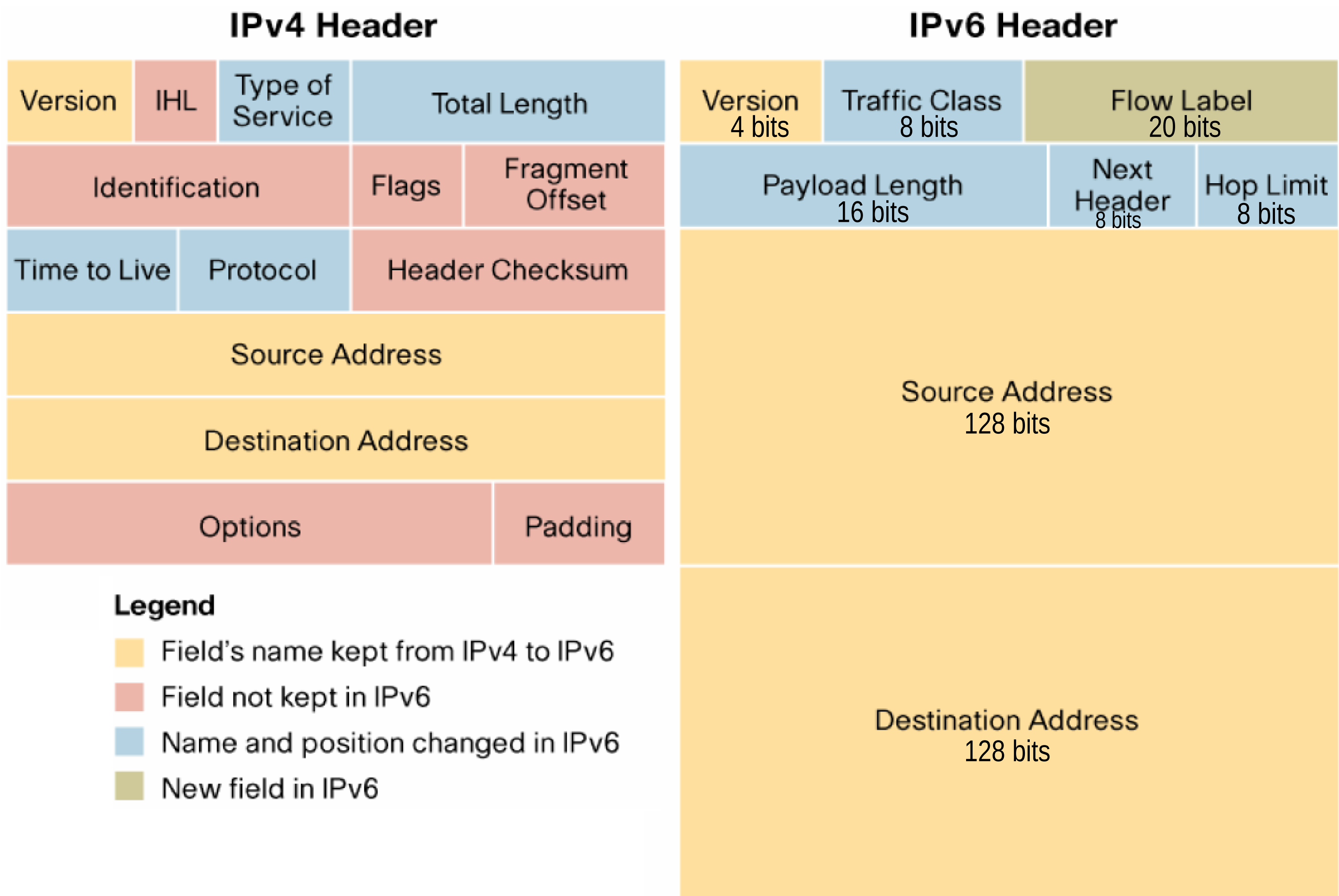
NAT: Network Address Translation

- 16-bit port-number field:
  - Over 64,000 simultaneous connections with a single LAN-side address!
- NAT is controversial:
  - Servers may be looking for well-known port numbers which could be changed during NAT translation
  - Routers should only process up to layer 3
    - Changing port #s is processing layer 4 information
  - NAT possibility must be taken into account by app designers, e.g., P2P applications
  - Address shortage should instead be solved by IPv6 …

IPv6

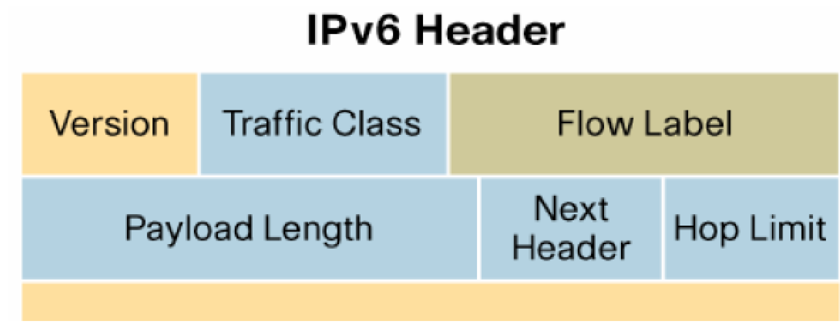
- Initial motivation: 32-bit address space depleted as of 3 Feb 11
  - https://www.nro.net/news/ipv4-free-pool-depleted
- IPv6 is designed to take an evolutionary step from IPv4
  - Downward compatible
  - Functions which worked in IPv4 were kept
  - Functions that didn't work were eliminated
- Additional motivation:
  - Header format helps speed processing/forwarding
  - Header changes to facilitate QoS

IPv6 Fixed-length 40-byte Header



http://www.cisco.com/en/US/technologies/tk648/tk872/technologies_white_paper0900aecd8054d37d.html

IPv6 Header



- Version: 6
- Traffic Class (Priority): identify priority of datagrams in flow (concept of "flow" not well defined)
- Flow Label: identify datagrams in same "flow"
- Payload Length: size of the payload in octets, including any extension headers
- Next header: identify upper layer protocol for data or the next options header
  - Options are available but outside of header
    - 8 Extension Headers
- Hop Limit: replaces TTL

Other Changes From IPv4

- Checksum removed entirely to reduce processing time at each hop
- Fragmentation not allowed within routers
  - Must fragment at source
  - Packets are dropped if too big
    - ICMPv6: new version of ICMP
      - Additional message types, e.g., "Packet Too Big"

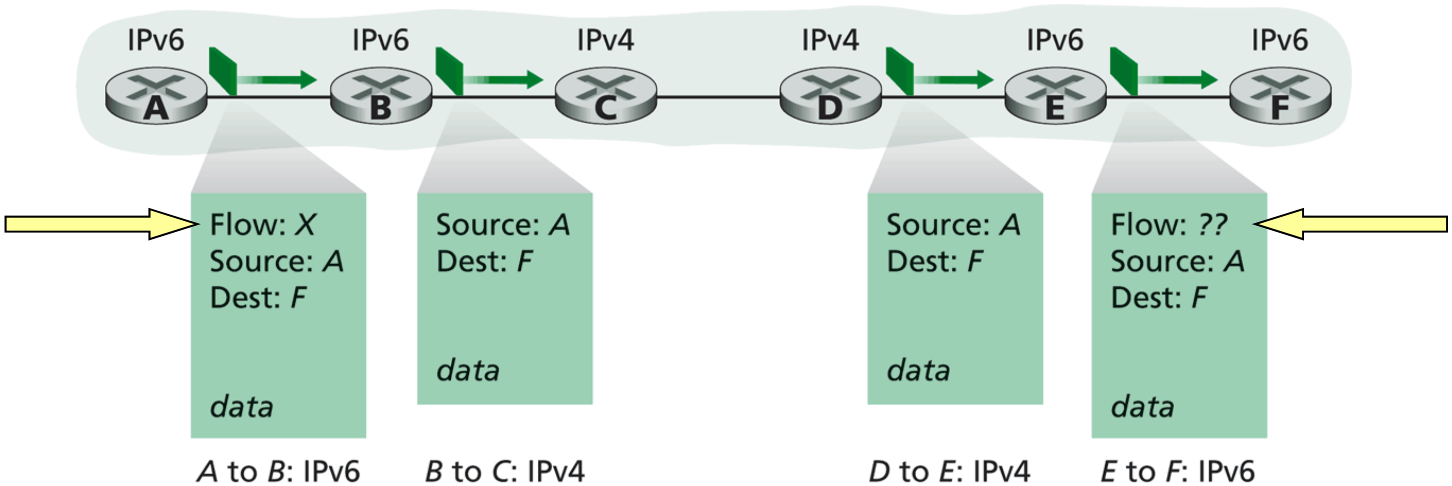
- IPv6 address → 128 bits
  - 8 16-bit integers (each integer represented by 4 hex digits)
  - Dotted decimal unwieldy: **105.220.136.100.255.255.255.255.0.0.18.128.140.10.255.255**
  - Colon hex → **69DC:8864:FFFF:FFFF:0:1280:8C0A:FFFF**

Other Changes From IPv4

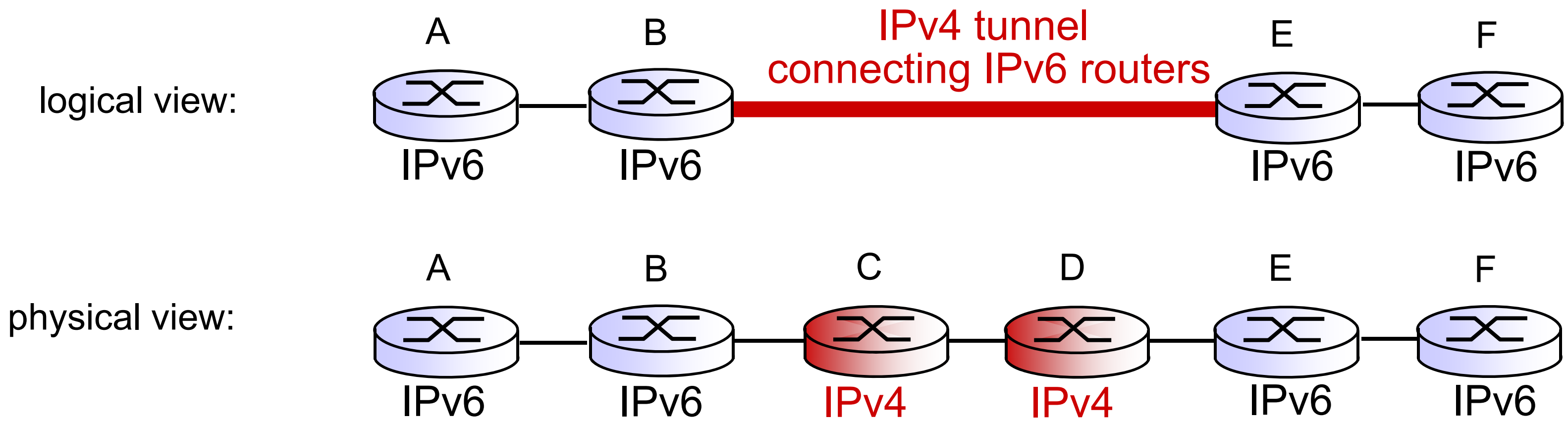
- What does 128-bit addressing imply?
  - Never run out of addresses?
  - IPv4 allows for only 4,294,967,296 unique addresses worldwide
    - Less than one address per person alive
    - Nov 2018 → I'm using 26 for my family alone
      - 5 cell phones, 3 smart TVs, 5 PCs
      - 4 DVD players, 1 Wii, 1 Xbox, 7 security cameras
  - IPv6 allows for around 2^{128} or approx 3.4×10^{38} addresses
    - That's about 4.4×10^{27} addresses per person (7.6 billion)
    - That equates to 667×10^{21} addresses per square meter of the surface of the Earth!!!! (510.1 trillion m^2)

Transition From IPv4 To IPv6

- Not all routers can be upgraded simultaneous
  - How will the network operate with mixed IPv4 / IPv6 routers?
- Could run a dual-stack on each host
  - Some IPv6 header information could be lost

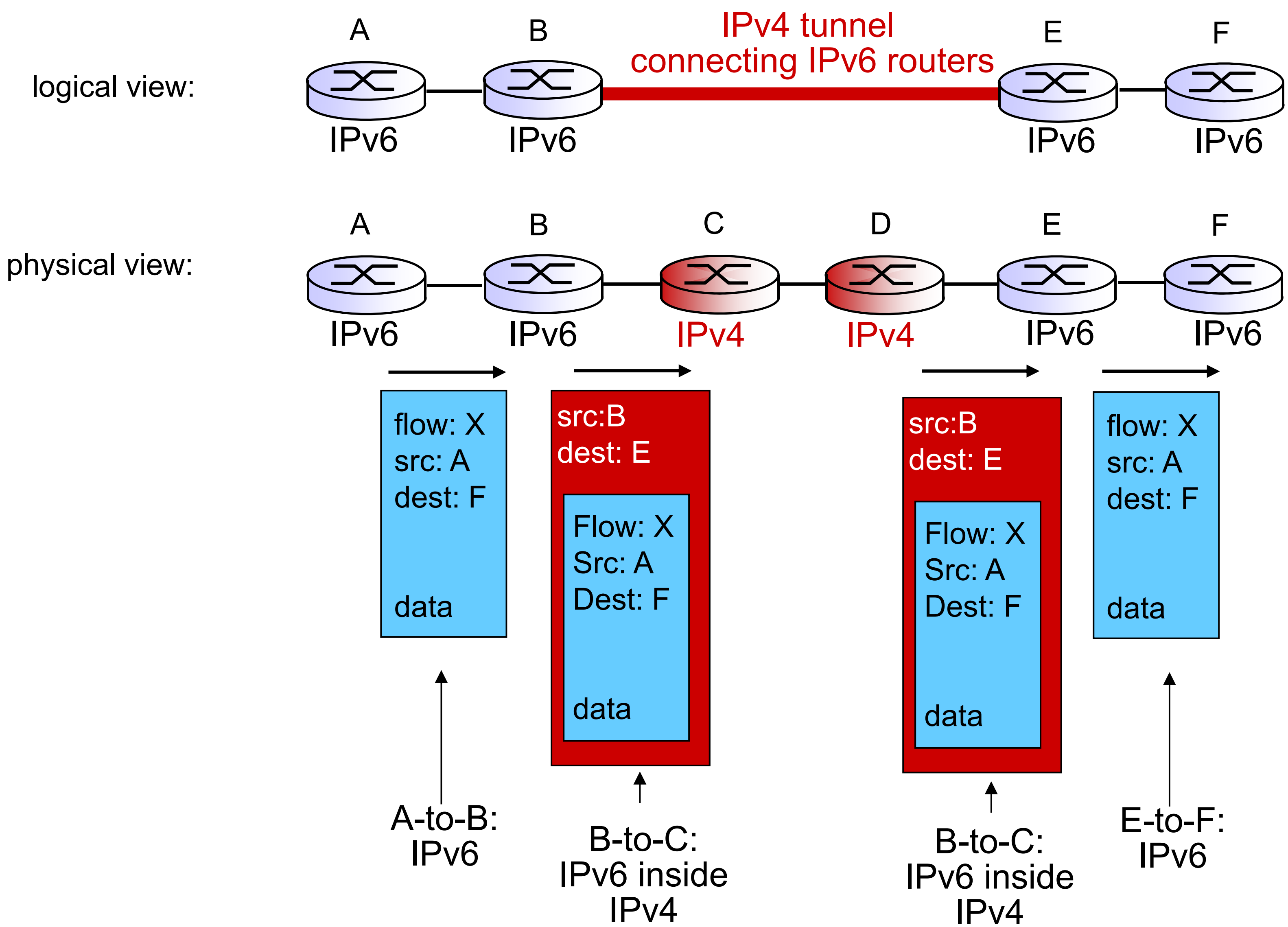


Tunneling



logical view:
A
B
IPv4 tunnel
connecting IPv6 routers
E
F
IPv6
IPv6
IPv6
IPv6
physical view:
A
B
C
D
E
F
IPv6
IPv6
IPv4
IPv4
IPv6
IPv6

Tunneling



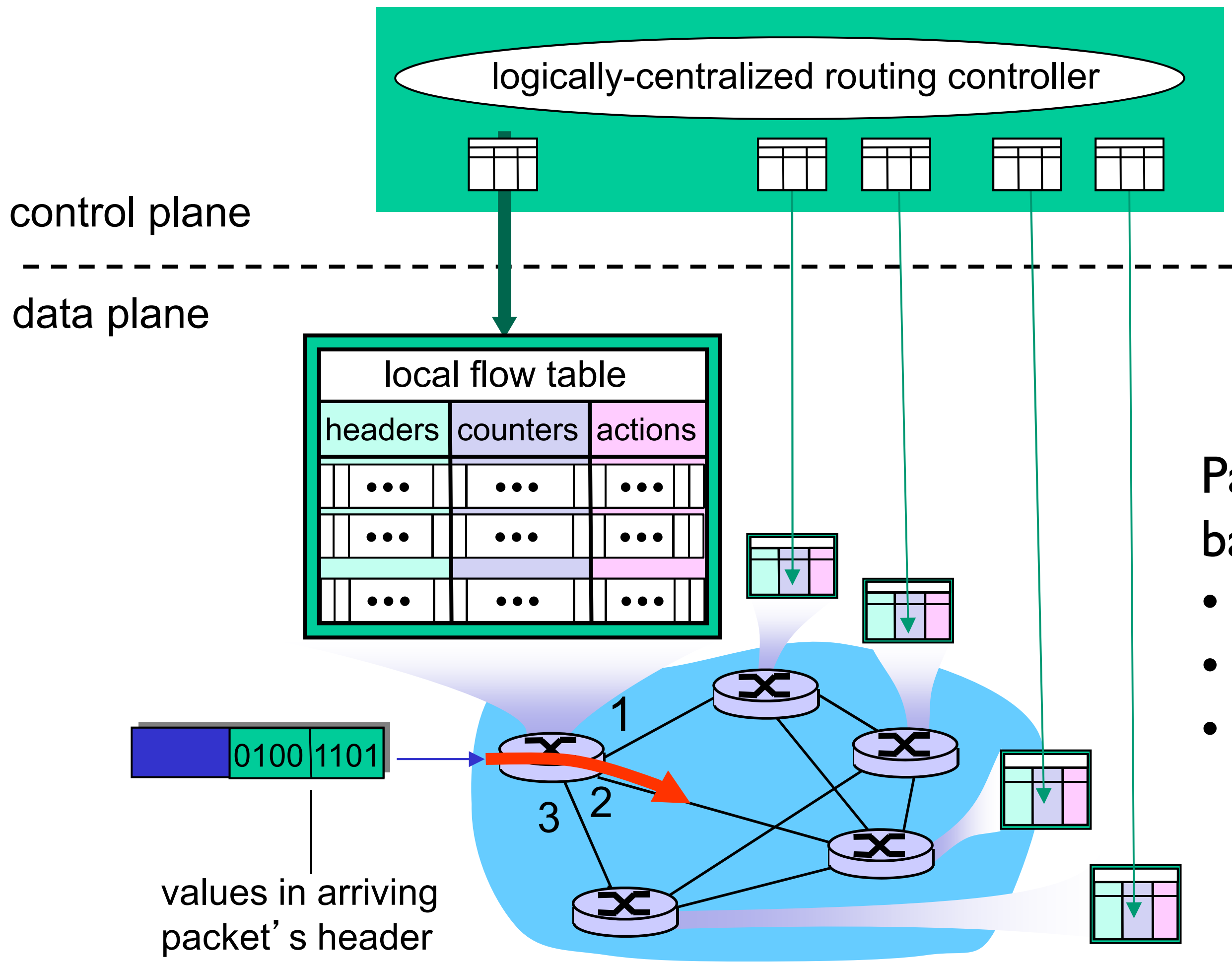
logical view:
A
B
IPv4 tunnel
connecting IPv6 routers
E
F
IPv6
IPv6
IPv6
IPv6
physical view:
A
B
C
D
E
F
IPv6
IPv6
IPv4
IPv4
IPv6
IPv6
flow: X
src: A
dest: F
data
src:B
dest: E
Flow: X
Src: A
Dest: F
data
src:B
dest: E
Flow: X
Src: A
Dest: F
data
flow: X
src: A
dest: F
data
A-to-B:
IPv6
B-to-C:
IPv6 inside
IPv4
B-to-C:
IPv6 inside
IPv4
E-to-F:
IPv6

Chapter 4: Network Layer

- 4.1 Overview of Network Layer
  - Data plane
  - Control plane
- 4.2 What's inside a router
- 4.3 IP: Internet Protocol
  - Datagram format
  - Fragmentation
  - IPv4 addressing
  - Network Address Translation
  - IPv6
- 4.4 Generalized Forwarding and SDN
  - Match
  - Action
  - OpenFlow Examples of match-plus-action in action

Generalized Forwarding and SDN

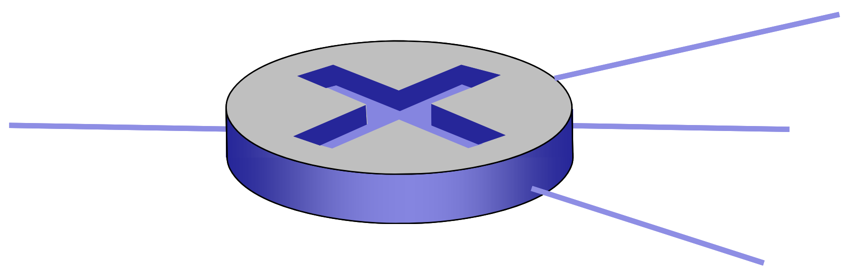
- Each router contains at least one flow table that is computed and distributed by a logically-centralized routing controller



Packet decisions are based on
- Transport headers
- Network headers
- Link headers

OpenFlow Data Plane Abstraction

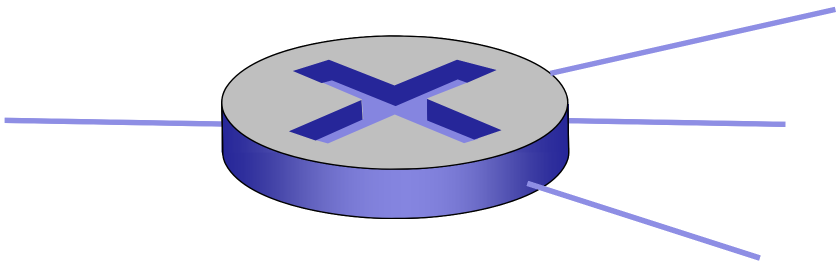
- Flow: defined by header fields
- Generalized forwarding: simple packet-handling rules
  - Pattern: match values in packet header fields
  - Counters: #bytes and #packets
  - Actions for matched packet
    - Drop, forward, modify, or send packet to controller
  - Priority: disambiguate overlapping patterns



Flow table in a router (computed and distributed by controller) defines router's match+action rules

OpenFlow Data Plane Abstraction

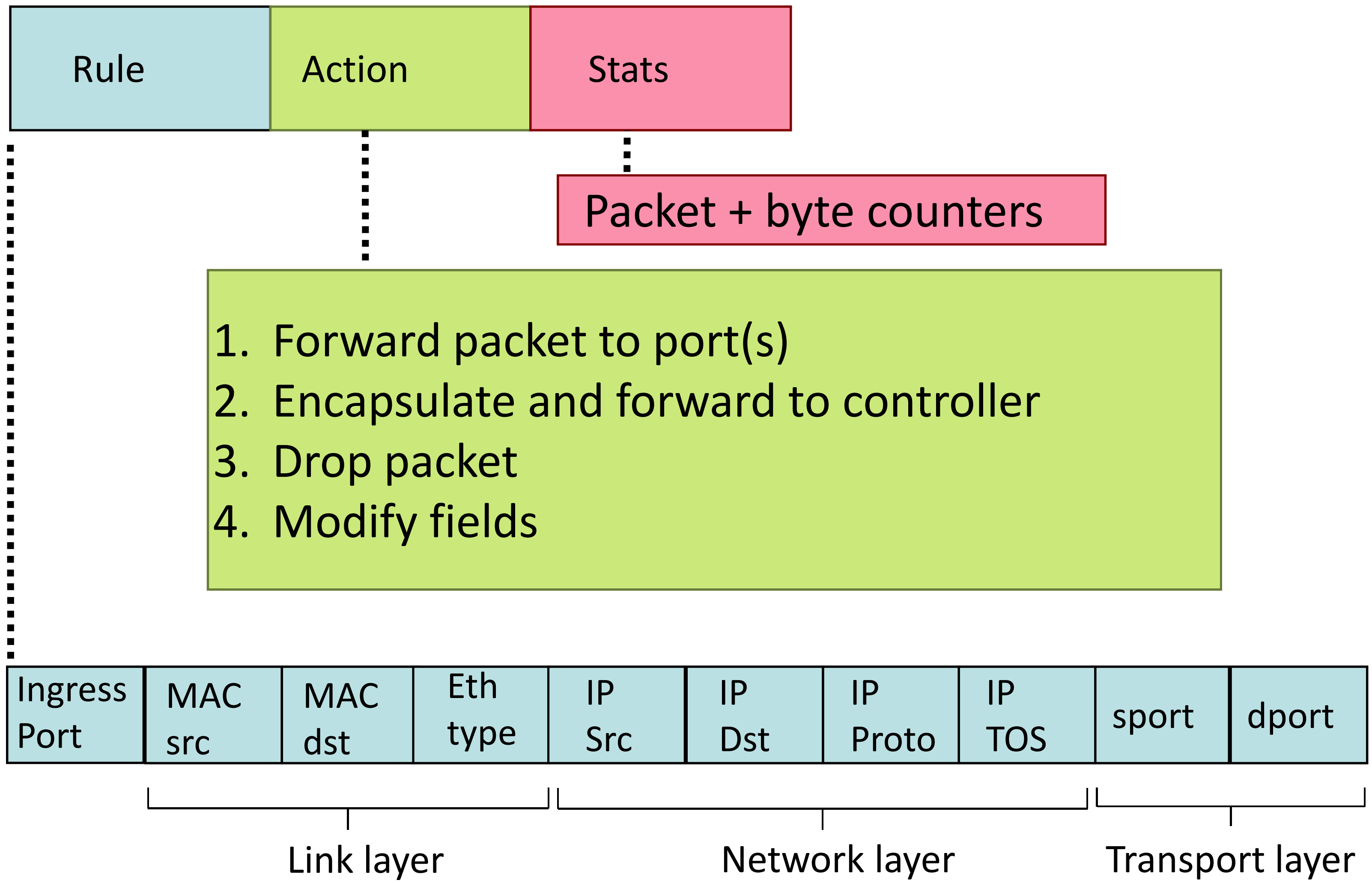
- Flow: defined by header fields
- Generalized forwarding: simple packet-handling rules
  - Pattern: match values in packet header fields
  - Counters: #bytes and #packets
  - Actions for matched packet
    - Drop, forward, modify, or send packet to controller
  - Priority: disambiguate overlapping patterns



* : wildcard

1. src = 1.2.*.*, dest=3.4.5.* → drop
2. src = *.*.*.*, dest=3.4.*.* → forward(2)
3. src = 10.1.2.3, dest=*.*.*.* → send to controller

OpenFlow 1.0: Flow Table Entries



IP Proto – upper layer protocol IP TOS – type of service

OpenFlow Abstraction

Match + Action: unifies different kinds of devices

- Router
  - match: longest destination IP prefix
  - action: forward out a link
- Switch
  - match: destination MAC address
  - action: forward or flood
- Firewall
  - match: IP addresses and TCP/UDP port numbers
  - action: permit or deny
- NAT
  - match: IP address and port
  - action: rewrite address and port

Examples

Router: Destination-based forwarding

Ingress Port	MAC src	MAC dst	Eth type	IP Src	IP Dst	IP Proto	IP TOS	sport	dport	Action: Forward
*	*	*	*	*	51.6.0.8	*	*	*	*	port6

IP datagrams destined to IP address 51.6.0.8 should be forwarded to router output port 6

Switch: Destination-based layer 2 forwarding

Ingress Port	MAC src	MAC dst	Eth type	IP Src	IP Dst	IP Proto	IP TOS	sport	dport	Forward
*	22:A7:23: 11:E1:02	*	*	*	*	*	*	*	*	port3

Layer 2 frames from MAC address 22:A7:23:11:E1:02 should be forwarded to output port 3

Examples

Firewall:

Ingress Port	MAC src	MAC dst	Eth type	IP Src	IP Dst	IP Proto	IP TOS	sport	dport	Forward
*	*	*	*	*	*	*	*	*	22	drop

Do not forward (block) all datagrams destined to TCP port 22

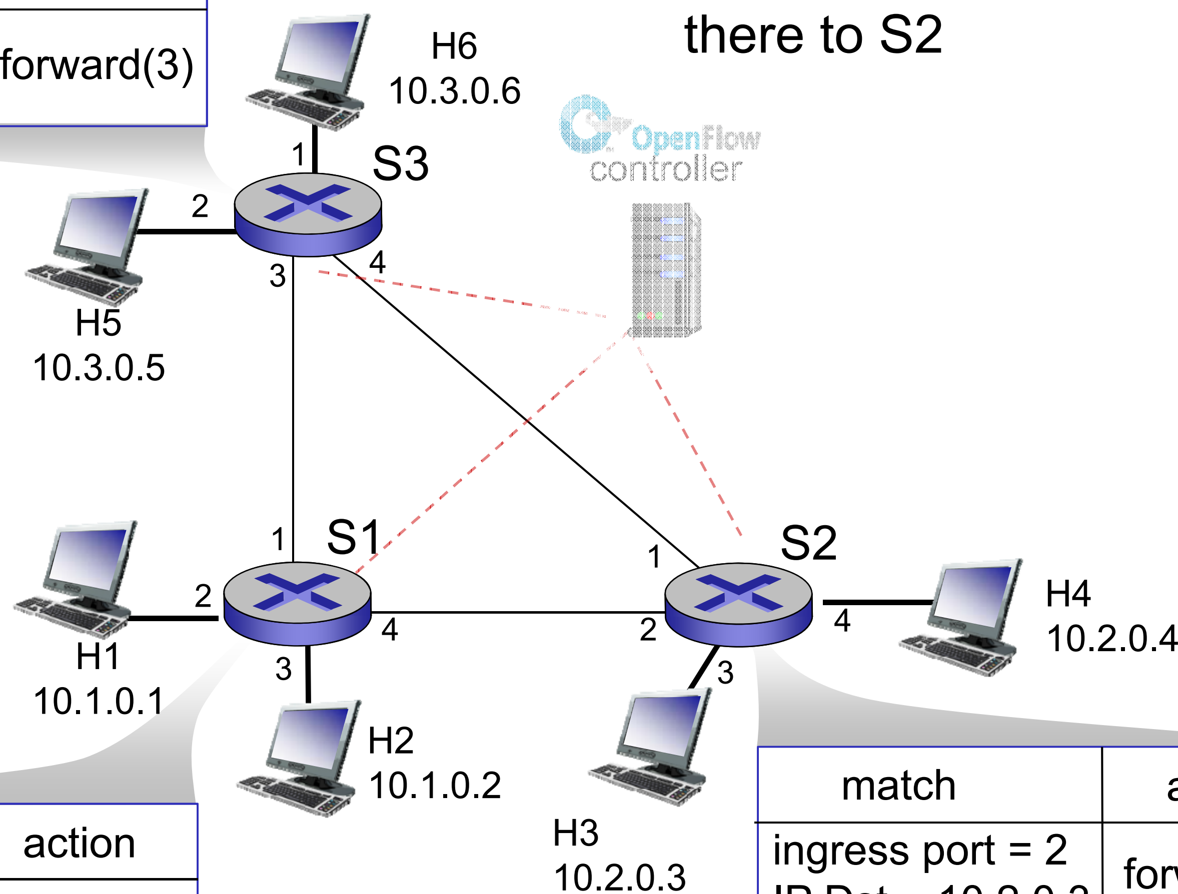
Ingress Port	MAC src	MAC dst	Eth type	IP Src	IP Dst	IP Proto	IP TOS	sport	dport	Forward
*	*	*	*	128.1.2.3	*	*	*	*	*	drop

Do not forward (block) all datagrams sent by host 128.1.2.3

OpenFlow Example

Example:
Datagrams from hosts H5 and H6 should be sent to H3 or H4, via S1 and from there to S2

match	action
IP Src = 10.3.*.* IP Dst = 10.2.*.*	forward(3)



match	action
ingress port = 1 IP Src = 10.3.*.* IP Dst = 10.2.*.*	forward(4)

match	action
ingress port = 2 IP Dst = 10.2.0.3	forward(3)
ingress port = 2 IP Dst = 10.2.0.4	forward(4)