Week 7

Intro to Network Layer & IP Addressing

CSCM603154 – **Computer Networks**

Faculty of Computer Science Universitas Indonesia

Network layer: our goals

- understand principles behind network layer services, focusing on data plane:
  - network layer service models
  - forwarding versus routing
  - addressing
  - generalized forwarding

- instantiation, implementation in the Internet
  - IP protocol
  - NAT

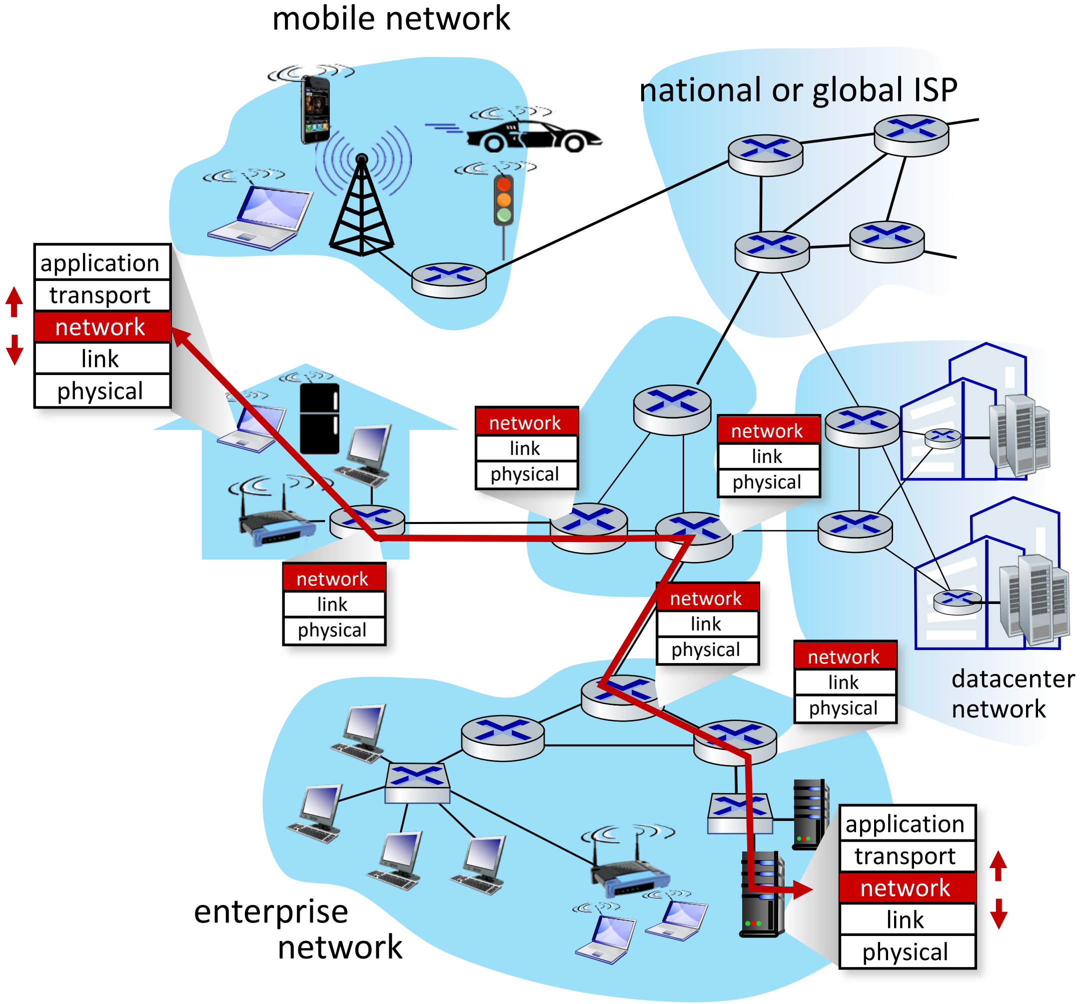
Network layer: "data plane" roadmap

- Network layer: overview
  - data plane
  - control plane
- IP: the Internet Protocol
  - datagram format
  - addressing
  - network address translation
  - IPv6

- Generalized Forwarding, SDN
  - Match+action
  - OpenFlow: match+action in action

Network-layer services and protocols

- transport segment from sending to receiving host
  - sender: encapsulates segments into datagrams, passes to link layer
  - receiver: delivers segments to transport layer protocol
- network layer protocols in *every Internet device*: hosts, routers
- routers:
  - examines header fields in all IP datagrams passing through it
  - moves datagrams from input ports to output ports to transfer datagrams along end-end path



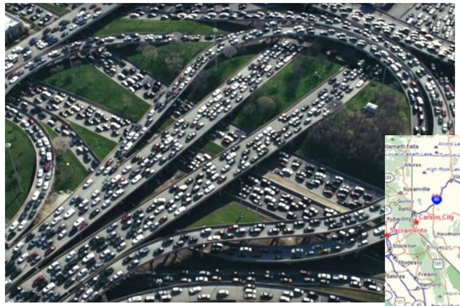
Two key network-layer functions

network-layer functions:

- *forwarding:* move packets from a router's input link to appropriate router output link
- *routing:* determine route taken by packets from source to destination
  - *routing algorithms*

analogy: taking a trip

- *forwarding:* process of getting through single interchange
- *routing:* process of planning trip from source to destination



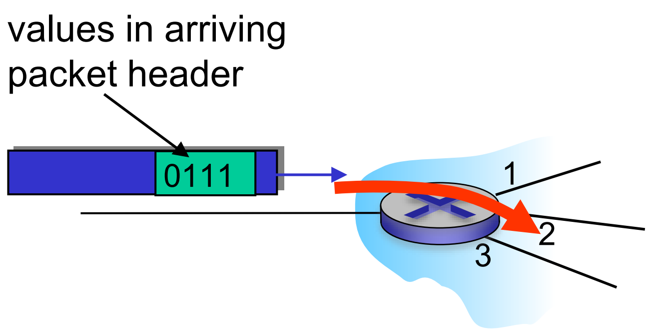
forwarding

routing

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

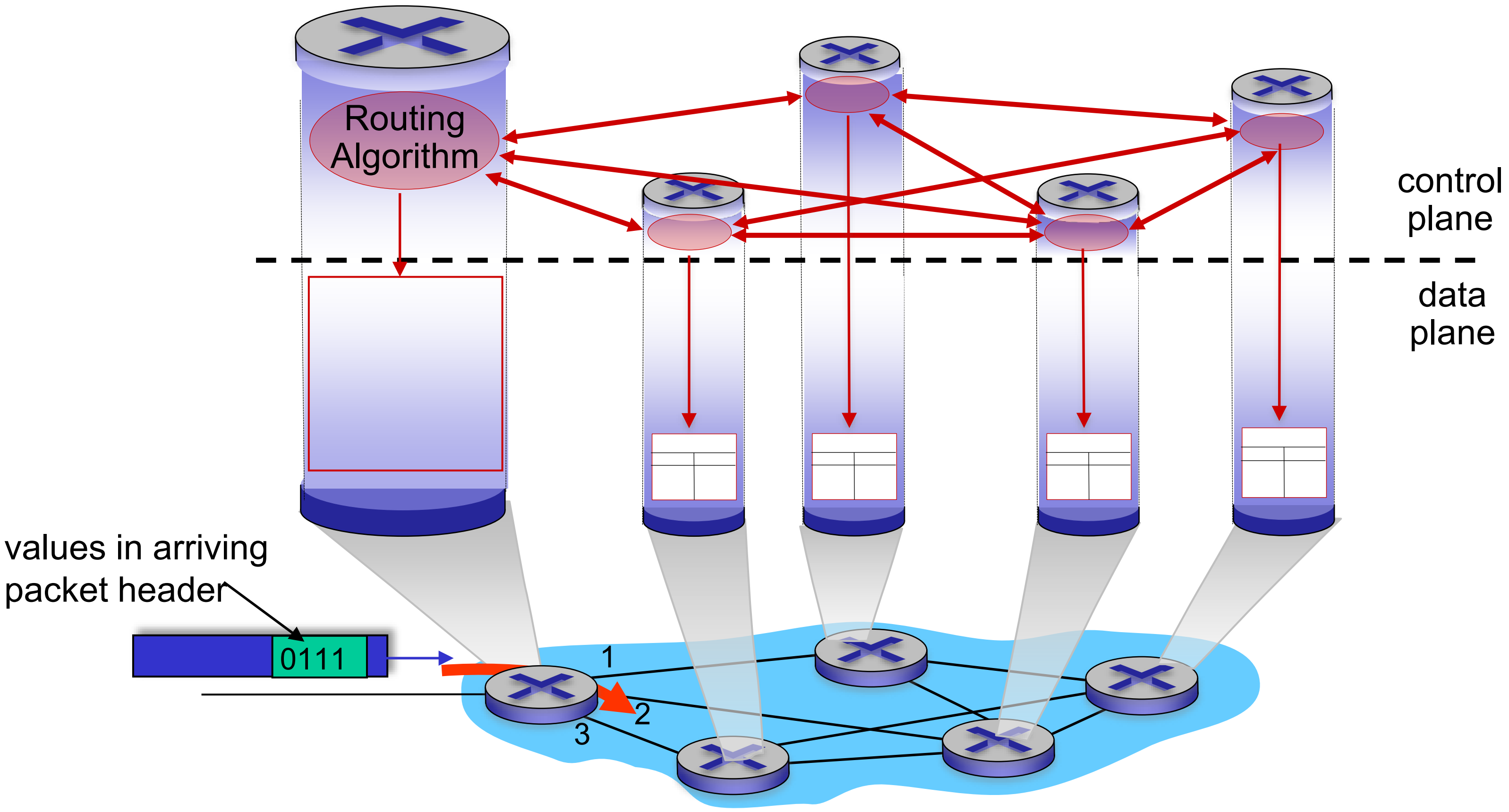


Control plane

- *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

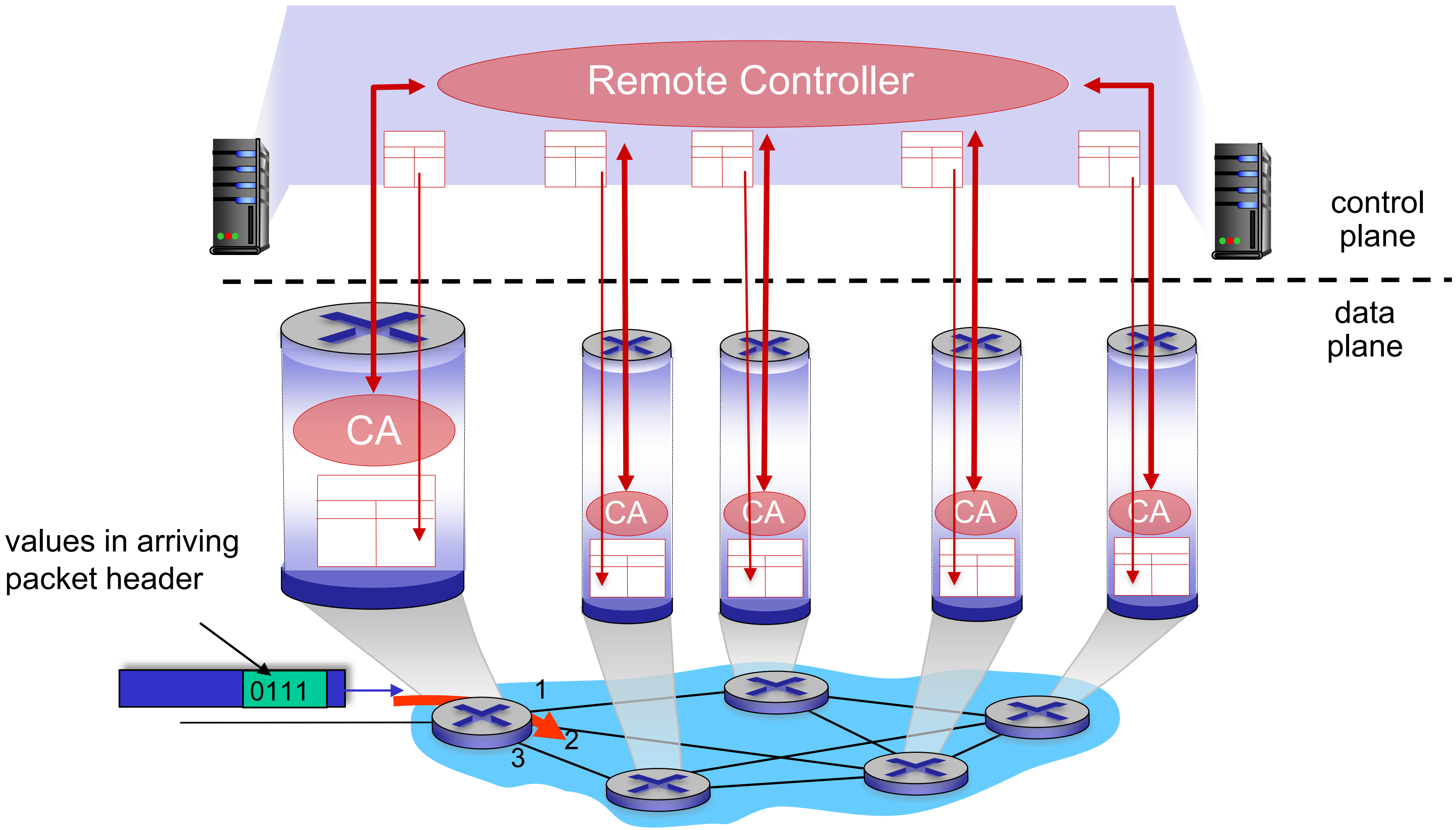
Per-router control plane

Individual routing algorithm components *in each and every router* interact in the control plane



Software-Defined Networking (SDN) control plane

Remote controller computes, installs forwarding tables in routers



Network service model

Q: What *service model* for "channel" transporting datagrams from sender to receiver?

example services for *individual* datagrams:

- guaranteed delivery
- guaranteed delivery with less than 40 msec delay

example services for a *flow* of datagrams:

- in-order datagram delivery
- guaranteed minimum bandwidth to flow
- restrictions on changes in inter-packet spacing

Network-layer service model

Network Architecture	Service Model	Quality of Service (QoS) Guarantees ?			
		Bandwidth	Loss	Order	Timing
Internet	best effort	none	no	no	no

Internet "best effort" service model

No guarantees on*:*

i. successful datagram delivery to destination
ii. timing or order of delivery
iii. bandwidth available to end-end flow

Network-layer service model

Network Architecture	Service Model	Quality of Service (QoS) Guarantees ?			
		Bandwidth	Loss	Order	Timing
Internet	best effort	none	no	no	no
ATM	Constant Bit Rate	Constant rate	yes	yes	yes
ATM	Available Bit Rate	Guaranteed min	no	yes	no
Internet	Intserv Guaranteed (RFC 1633)	yes	yes	yes	yes
Internet	Diffserv (RFC 2475)	possible	possibly	possibly	no

Reflections on best-effort service:

- simplicity of mechanism has allowed Internet to be widely deployed adopted
- sufficient provisioning of bandwidth allows performance of real-time applications (e.g., interactive voice, video) to be “good enough” for “most of the time”
- replicated, application-layer distributed services (datacenters, content distribution networks) connecting close to clients’ networks, allow services to be provided from multiple locations
- congestion control of “elastic” services helps

It’s hard to argue with success of best-effort service model

Sidebar: Network Neutrality

What is network neutrality?

- *technical:* how an ISP should share/allocation its resources
  - packet scheduling, buffer management are the *mechanisms*
- *social, economic* principles
  - protecting free speech
  - encouraging innovation, competition
- enforced *legal* rules and policies

Different countries have different "takes" on network neutrality

Sidebar: Network Neutrality

2015 US FCC *Order on Protecting and Promoting an Open Internet:* three "clear, bright line" rules:

- no blocking … "shall not block lawful content, applications, services, or non-harmful devices, subject to reasonable network management."
- no throttling … "shall not impair or degrade lawful Internet traffic on the basis of Internet content, application, or service, or use of a non-harmful device, subject to reasonable network management."
- no paid prioritization. … "shall not engage in paid prioritization"

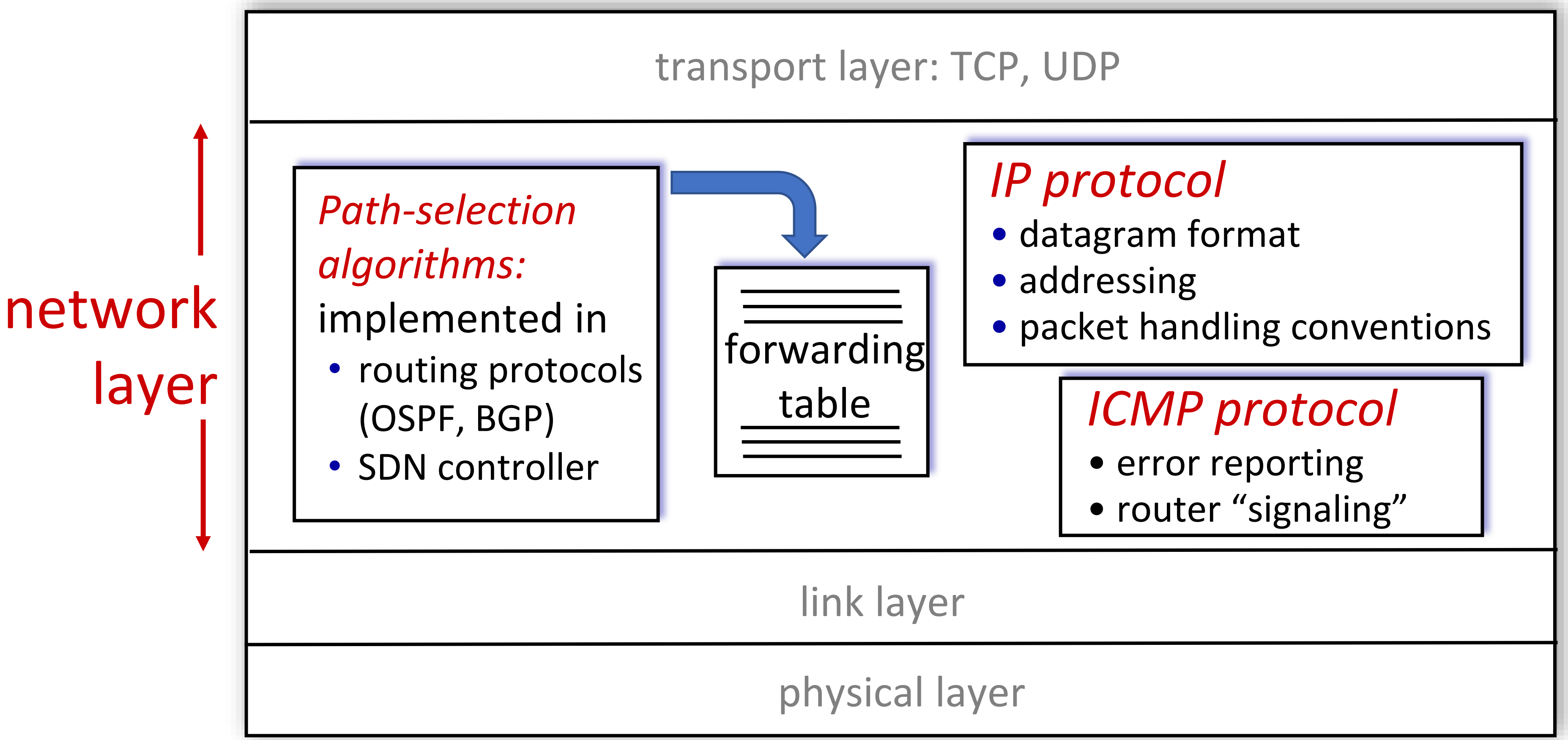
Network layer: "data plane" roadmap

- Network layer: overview
  - data plane
  - control plane
- IP: the Internet Protocol
  - datagram format
  - addressing
  - network address translation
  - IPv6

- Generalized Forwarding, SDN
  - Match+action
  - OpenFlow: match+action in action

Network Layer: Internet

host, router network layer functions:



IP Datagram format

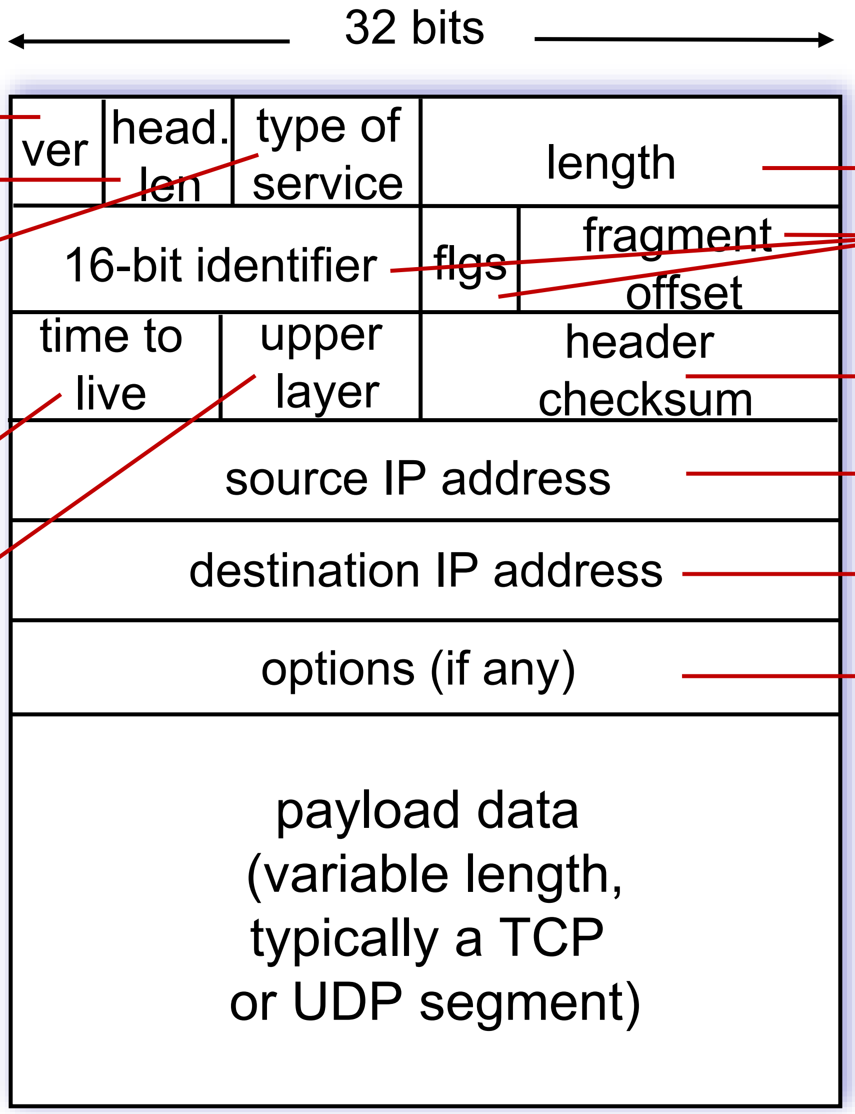
IP protocol version number

header length(bytes)

“type” of service:
- diffserv (0:5)
- ECN (6:7)

TTL: remaining max hops
(decremented at each router)

upper layer protocol (e.g., TCP or UDP)



total datagram length (bytes)

fragmentation/ reassembly

header checksum

32-bit source IP address

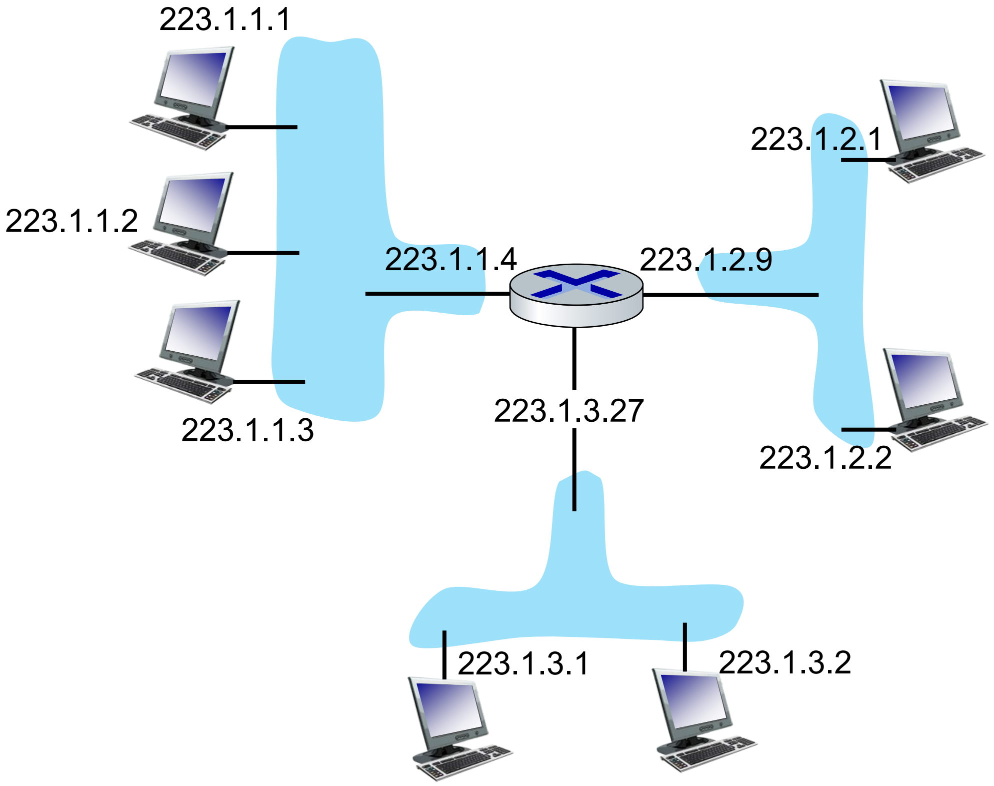
Maximum length: 64K bytes
Typically: 1500 bytes or less

e.g., timestamp, record route taken

overhead
- 20 bytes of TCP
- 20 bytes of IP
- = 40 bytes + app layer overhead for TCP+IP

IP addressing: introduction

- IP address: 32-bit identifier associated with each host or router *interface*
- interface: connection between host/router and physical link
  - router's typically have multiple interfaces
  - host typically has one or two interfaces (e.g., wired Ethernet, wireless 802.11)

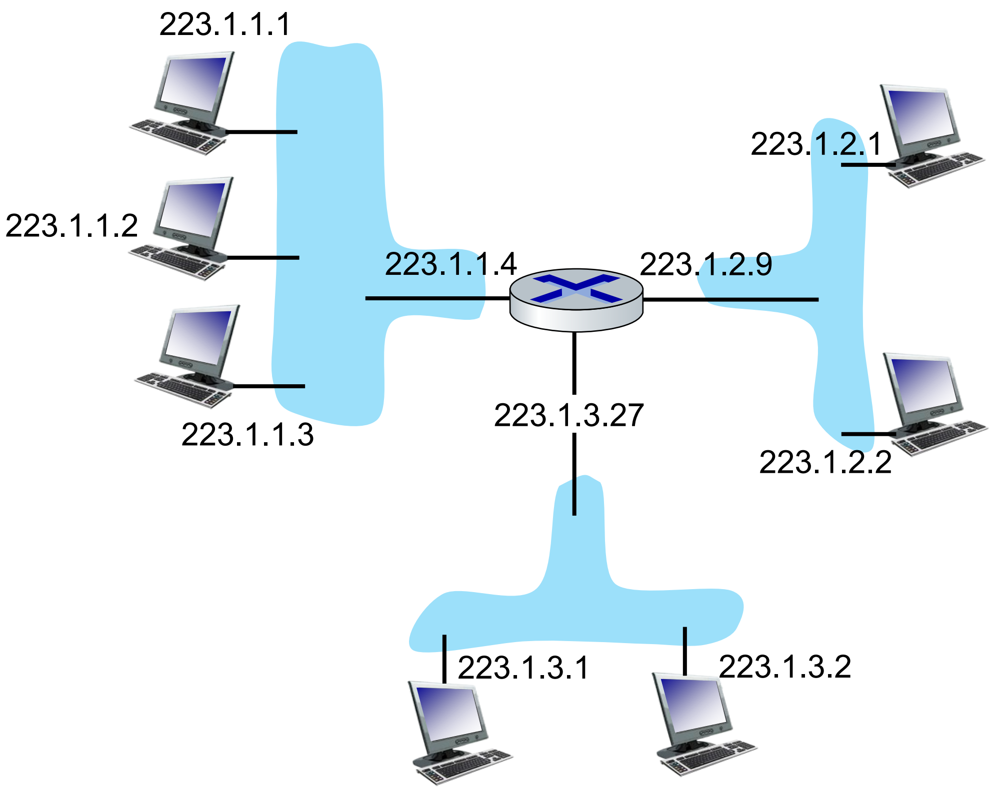


dotted-decimal IP address notation:

$$223.1.1.1 = \underbrace{11011111}_{223}\ \underbrace{00000001}_{1}\ \underbrace{00000001}_{1}\ \underbrace{00000001}_{1}$$

IP addressing: introduction

- IP address: 32-bit identifier associated with each host or router *interface*
- interface: connection between host/router and physical link
  - router's typically have multiple interfaces
  - host typically has one or two interfaces (e.g., wired Ethernet, wireless 802.11)



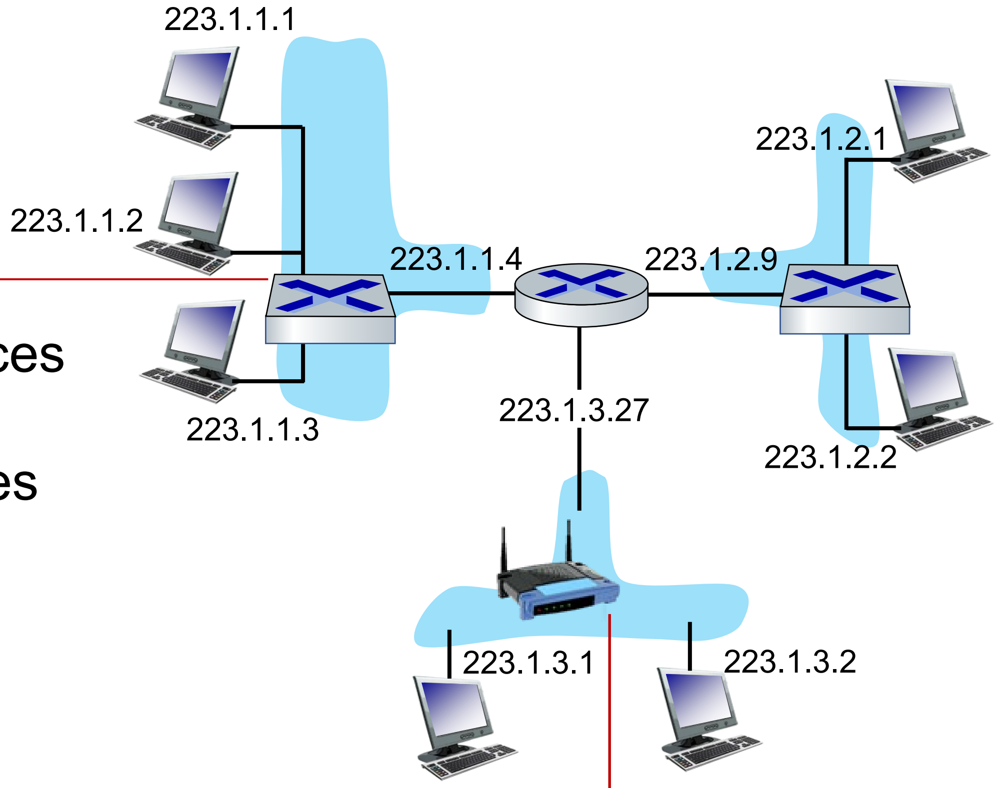
dotted-decimal IP address notation:

$$223.1.1.1 = \underbrace{11011111}_{223}\ \underbrace{00000001}_{1}\ \underbrace{00000001}_{1}\ \underbrace{00000001}_{1}$$

IP addressing: introduction

Q: how are interfaces actually connected?

A: we'll learn about that in chapters 6, 7



A: wired Ethernet interfaces connected by Ethernet switches

A: wireless WiFi interfaces connected by WiFi base station

For now: don't need to worry about how one interface is connected to another (with no intervening router)

IP addressing: CIDR

CIDR: Classless InterDomain Routing (pronounced “cider”)

- subnet portion of address of arbitrary length
- address format: a.b.c.d/x, where x is # bits in subnet portion of address

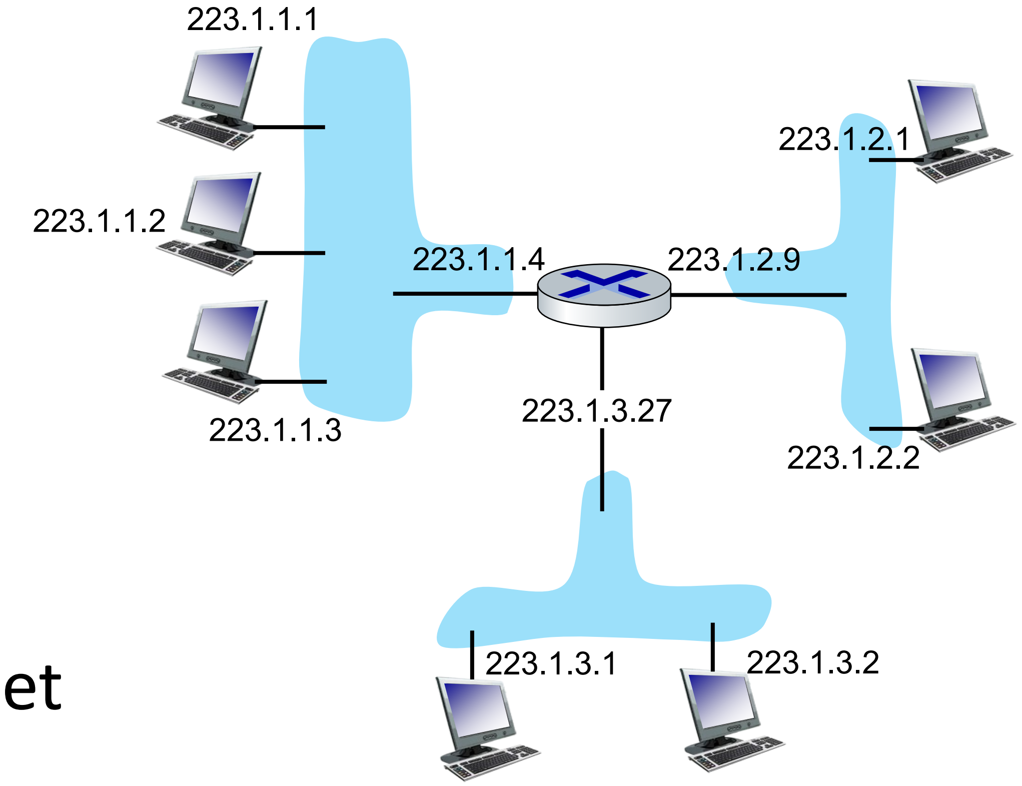
subnet part | host part

11001000 00010111 00010000 00000000

200.23.16.0/23

Subnets

- *What's a subnet ?*
  - device interfaces that can physically reach each other without passing through an intervening router
- IP addresses have structure:
  - subnet part: devices in same subnet have common high order bits
  - host part: remaining low order bits

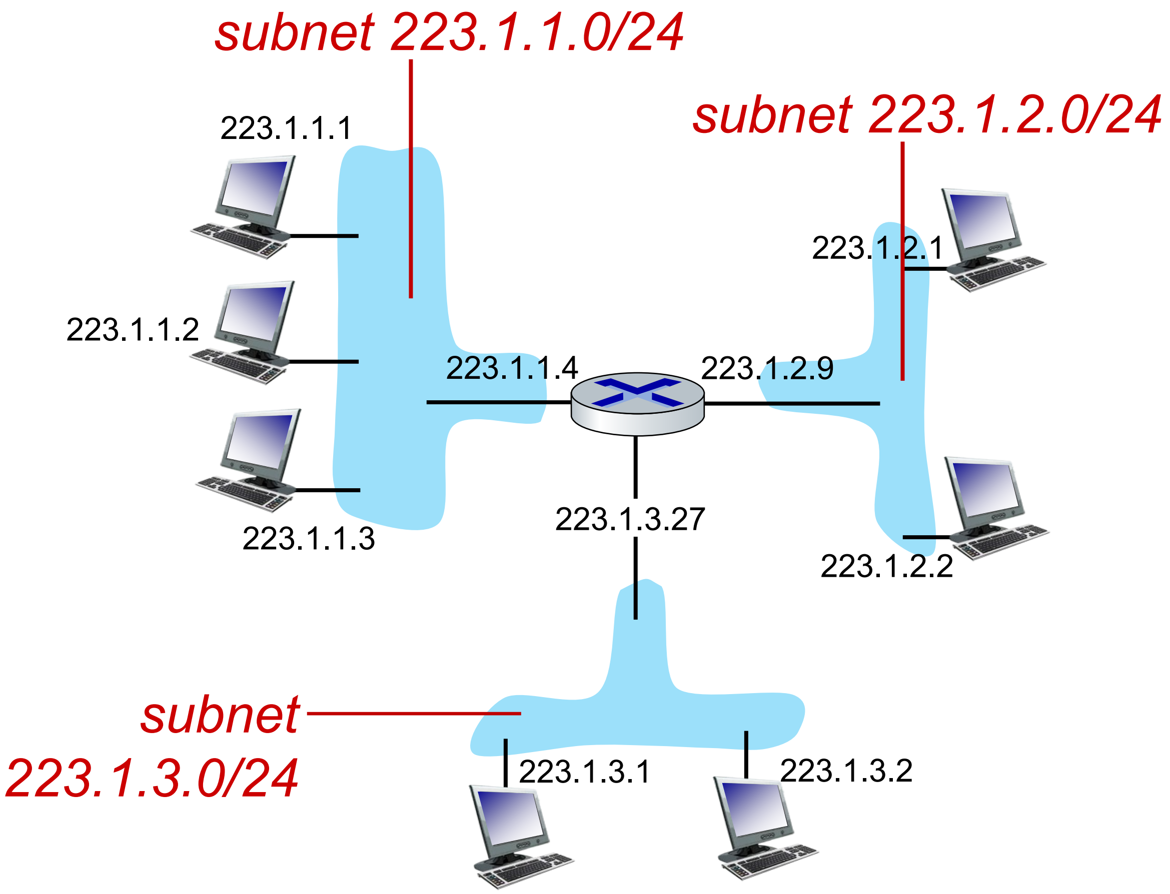


network consisting of 3 subnets

Subnets

Recipe for defining subnets:

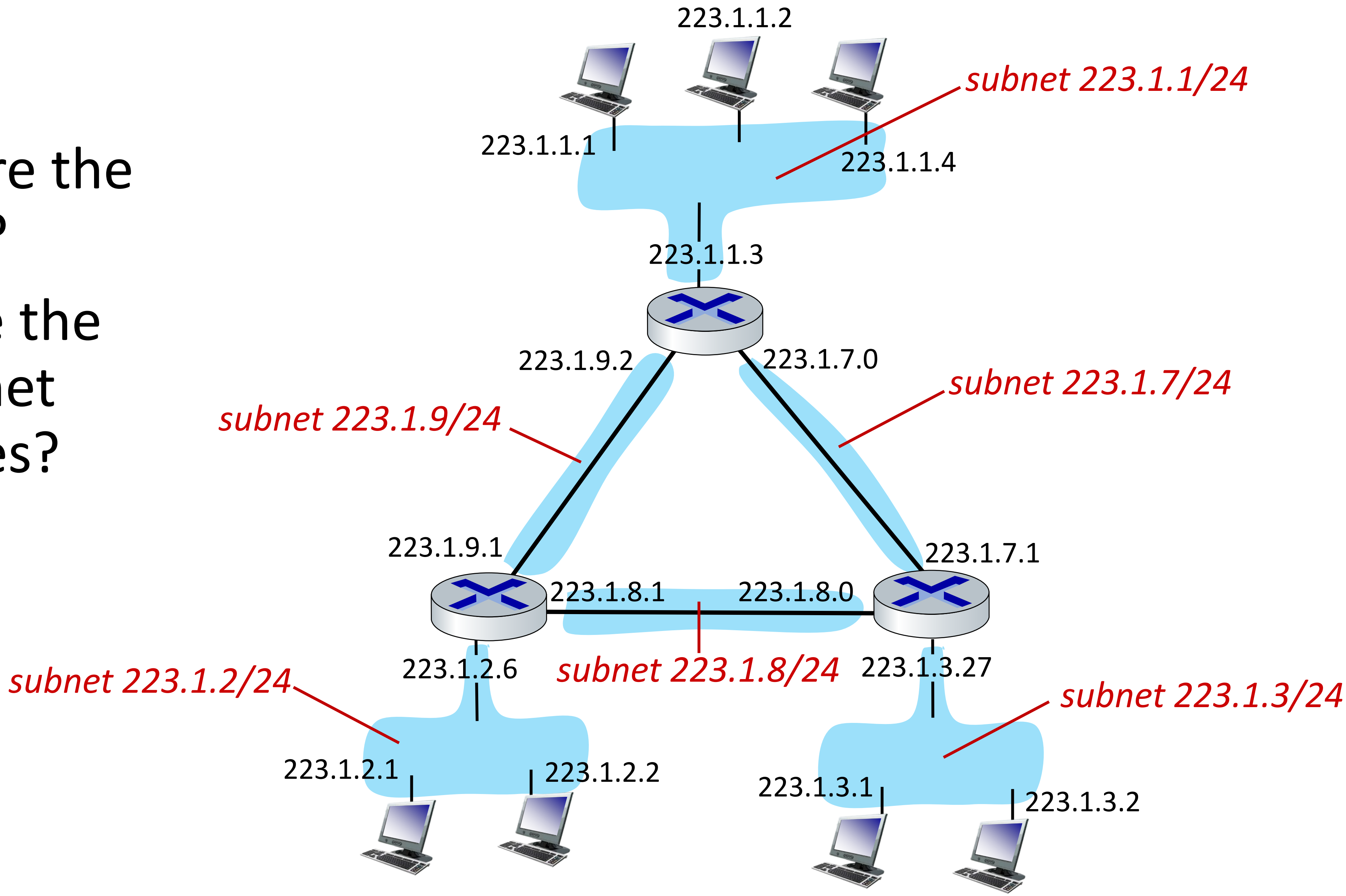
- detach each interface from its host or router, creating "islands" of isolated networks
- each isolated network is called a *subnet*



subnet mask: /24
(high-order 24 bits: subnet part of IP address)

Subnets

- where are the subnets?
- what are the /24 subnet addresses?



Subnet Mask

- A subnet is defined by applying the subnet mask to the IP address
- if a bits is "on" (set to 1) in the subnet mask, then that equivalent bit in the address is interpreted as a network bit
- if a bits is "off" (set to 0) in the subnet mask, then that equivalent bit in the address is interpreted as a host bit
- Standard subnet masks for the 3 classes of addresses
  · for a class A address - 255.0.0.0
  · for a class B address - 255.255.0.0
  · for a class C address - 255.255.255.0

Subnet Mask

- Use the subnet mask and ANDing process to extract the network address from the IP address

Applying the Subnet Mask

A device with address 192.0.0.1 belongs to network 192.0.0.0

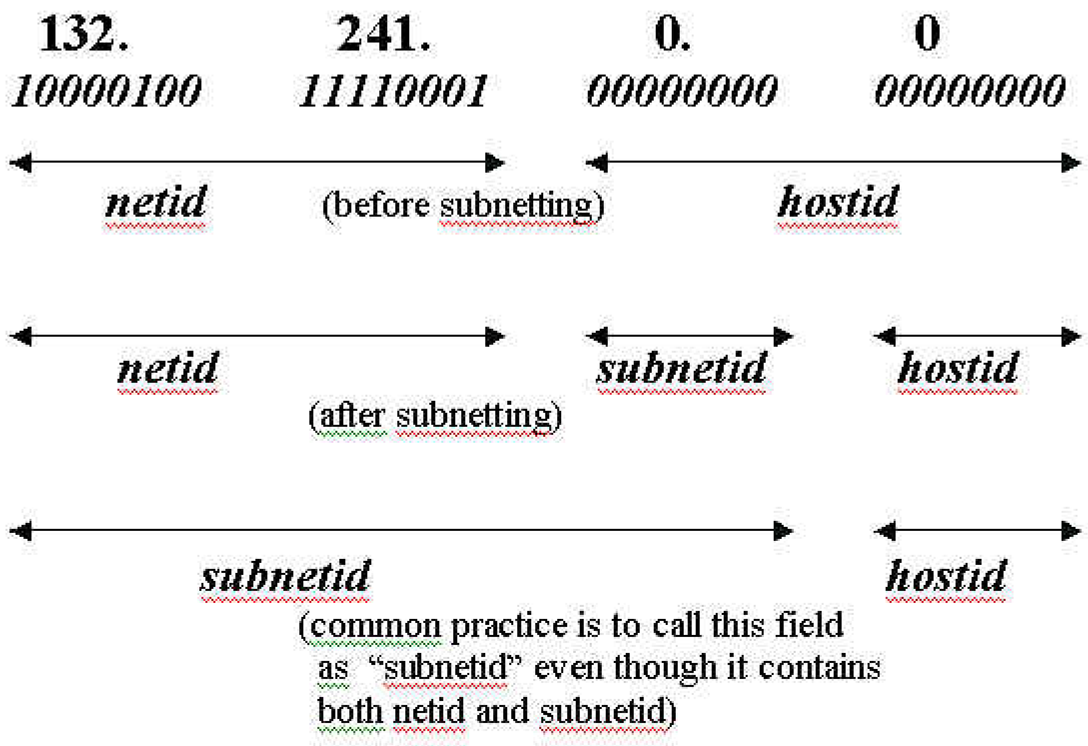
	High order bits Prefix /16		Low order bits	
	192 .	0 .	0 .	1
Host Address	11000000	00000000	00000000	00000001
Subnet Mask	255 11111111	255 11111111	0 00000000	0 00000000
Network Address	11000000	00000000	00000000	00000000
Network	192 .	0 .	0 .	0

Types of Subnetting

- There are two types of subnetting:
  - Static Subnetting
  - Variable Length Subnet Mask (VLSM)
- Variable length is the more flexible of the two.

Static Subnetting

- A portion of host address bits are used as subnetwork address bits
- The "dividing line" between network address and host address parts is shifted variably to the right



Static Subnetting

- Given network of 204.17.5.0/24. Create two network subnet:

Before subnetting:

204.17.5.0	11001100.00010001.00000101.00000000
255.255.255.0	11111111.11111111.11111111.00000000

After subnetting with **two subnet**

204.17.5.0	11001100.00010001.00000101.00000000
255.255.255.128	11111111.11111111.11111111.10000000
204.17.5.128	11001100.00010001.00000101.10000000
255.255.255.128	11111111.11111111.11111111.10000000

Static Subnetting

- Before Subnetting (204.17.5.0/24)

Network Address (NA)	4th Octet of NA (in binary)	Subnet Mask	First Host	Last Host
204.17.5.0	x.x.x.00000000	255.255.255.0	204.17.5.1	204.17.5.254

- After Subnetting (two subnets)

Network Address (NA)	4th Octet of NA (in binary)	Subnet Mask	First Host	Last Host
204.17.5.0	x.x.x.00000000	255.255.255.128	204.17.5.1	204.17.5.126
204.17.5.128	x.x.x.10000000	255.255.255.128	204.17.5.129	204.17.5.254

Static Subnetting

- Before Subnetting (204.17.5.0/24)

Network Address (NA)	Total Host (2^n-2)	First Host	Last Host	Broadcast
204.17.5.0/24	254	204.17.5.1	204.17.5.254	204.17.5.255

- After Subnetting (two subnets)

Network Address (NA)	Total Host (2^n-2)	First Host	Last Host	Broadcast
204.17.5.0	126	204.17.5.1	204.17.5.126	204.17.5.127
204.17.5.128	126	204.17.5.129	204.17.5.254	204.17.5.255

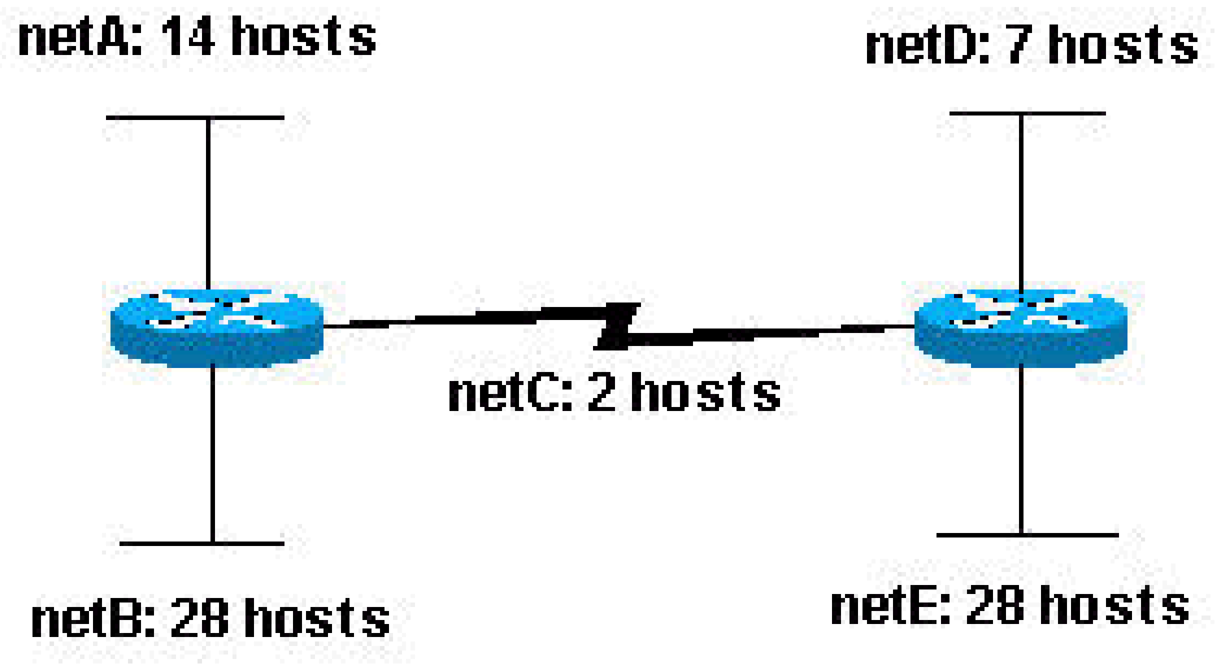
Static Subnetting

- Given network address = 204.17.5.0/24, create 8 subnet!

Network Address (NA)	4^{th} Octet of NA (in binary)	Subnet Mask	First Host	Last Host
204.17.5.0	x.x.x.00000000	255.255.255.224	x.x.x.1	x.x.x.30
204.17.5.32	x.x.x.00100000	255.255.255.224	x.x.x.33	x.x.x.62
204.17.5.64	x.x.x.01000000	255.255.255.224	x.x.x.65	x.x.x.94
204.17.5.96	x.x.x.01100000	255.255.255.224	x.x.x.97	x.x.x.126
204.17.5.128	x.x.x.10000000	255.255.255.224	x.x.x.129	x.x.x.158
204.17.5.160	x.x.x.10100000	255.255.255.224	x.x.x.161	x.x.x.190
204.17.5.192	x.x.x.11000000	255.255.255.224	x.x.x.193	x.x.x.222
204.17.5.224	x.x.x.11100000	255.255.255.224	x.x.x.225	x.x.x.254

Static Subnetting

- Given network address = 204.15.5.0/24. Subnet the network in order to create the network with the host requirements shown.

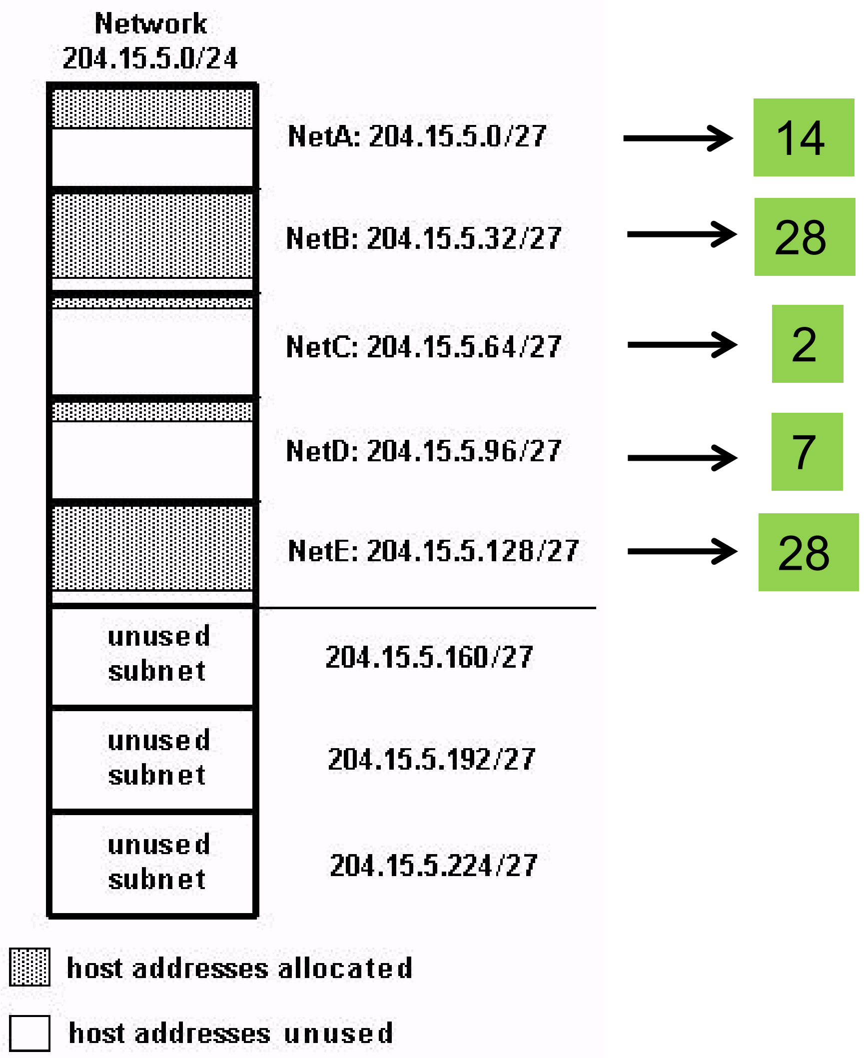


Static Subnetting

- Based on the figure
  - Total subnet based = 5 subnet
  - Maximum number of host = 28 host
- How many bits needed to create 5 subnet?
  - 1 bit ? Only 2 subnets created (2^1 subnet)
  - 2 bits? Only 4 subnets created (2^2 subnet)
  - 3 bits? 8 subnets created (2^3 subnet). 3 subnet will be unused

Static Subnetting

Possible assigned subnets:
netA: 204.15.5.0/27
  host address range 1 to 30
netB: 204.15.5.32/27
  host address range 33 to 62
netC: 204.15.5.64/27
  host address range 65 to 94
netD: 204.15.5.96/27
  host address range 97 to 126
netE: 204.15.5.128/27
  host address range 129 to 158



VLSM

- Based on the previous example, develop a subnetting scheme with the use of VLSM
  - netA: must support 14 hosts
  - netB: must support 28 hosts
  - netC: must support 2 hosts
  - netD: must support 7 hosts
  - netE: must support 28 host
- Determine what mask allows the required number of hosts.

VLSM

netA: requires a /28 (255.255.255.240) mask to support 14 hosts

netB: requires a /27 (255.255.255.224) mask to support 28 hosts

netC: requires a /30 (255.255.255.252) mask to support 2 hosts

netD*: requires a /28 (255.255.255.240) mask to support 7 hosts

netE: requires a /27 (255.255.255.224) mask to support 28 hosts

* a /29 (255.255.255.248) would only allow 6 usable host addresses, therefore netD requires a /28 mask.

VLSM

- The easiest way to assign the subnets is to assign the largest first. For example, you can assign in this manner:

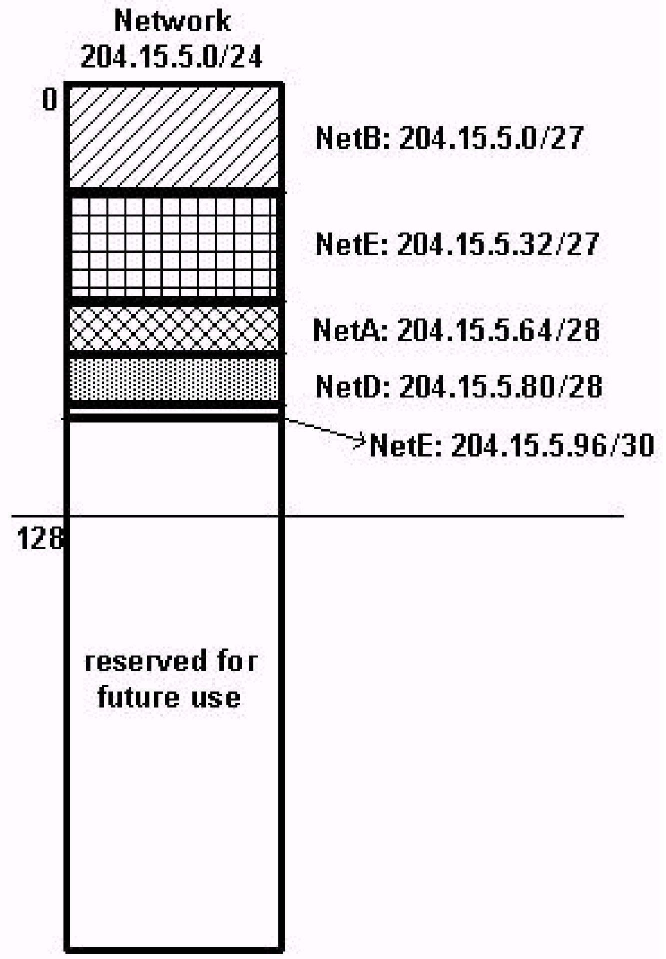
netB: 204.15.5.0/27 host address range 1 to 30

netE: 204.15.5.32/27 host address range 33 to 62

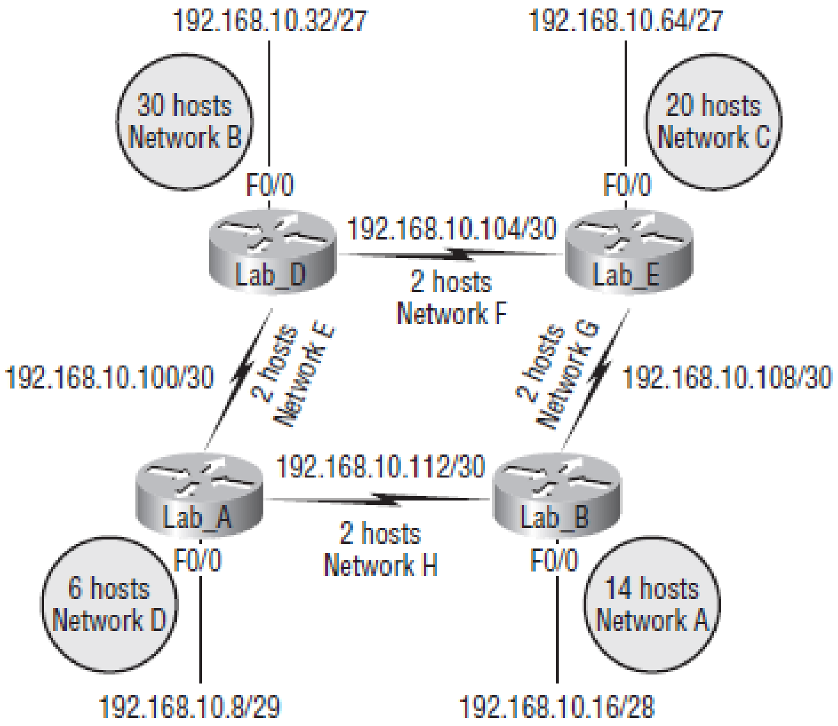
netA: 204.15.5.64/28 host address range 65 to 78

netD: 204.15.5.80/28 host address range 81 to 94

netC: 204.15.5.96/30 host address range 97 to 98



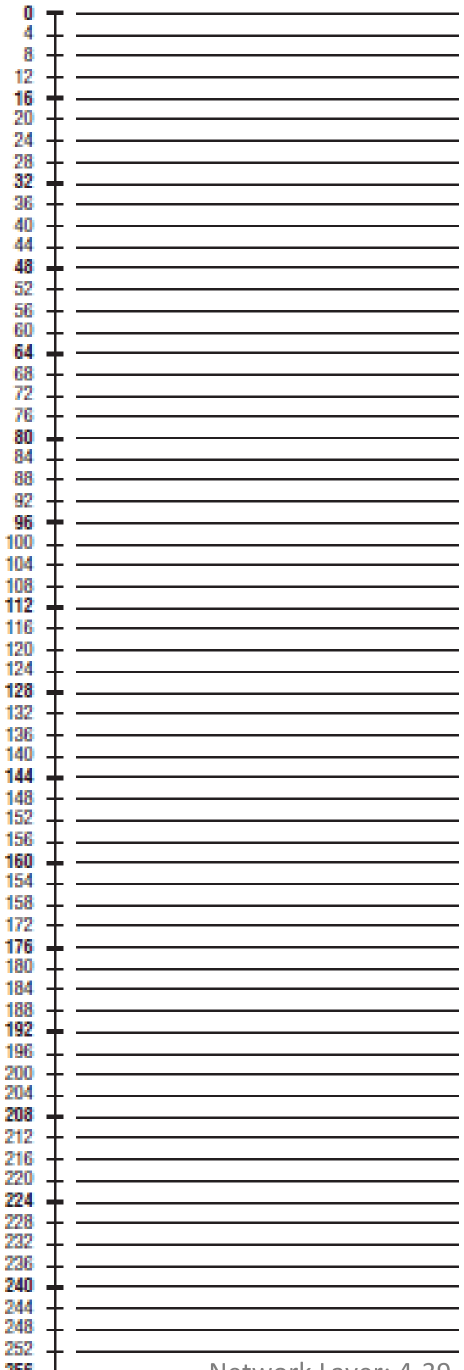
VLSM



Subnet	Mask	Subnets	Hosts	Block
/26	192	4	62	64
/27	224	8	30	32
/28	240	16	14	16
/29	248	32	6	8
/30	252	64	2	4

Class C Network 192.168.10.0

Network	Hosts	Block	Subnet	Mask
A				
B				
C				
D				
E				
F				
G				
H				
I				
J				
K				
L				



IP addresses: how to get one?

That's actually two questions:

1. Q: How does a *host* get IP address within its network (host part of address)?
2. Q: How does a *network* get IP address for itself (network part of address)

How does *host* get IP address?

- hard-coded by sysadmin in config file (e.g., /etc/rc.config in UNIX)
- DHCP: Dynamic Host Configuration Protocol: dynamically get address from as server
  - "plug-and-play"

DHCP: Dynamic Host Configuration Protocol

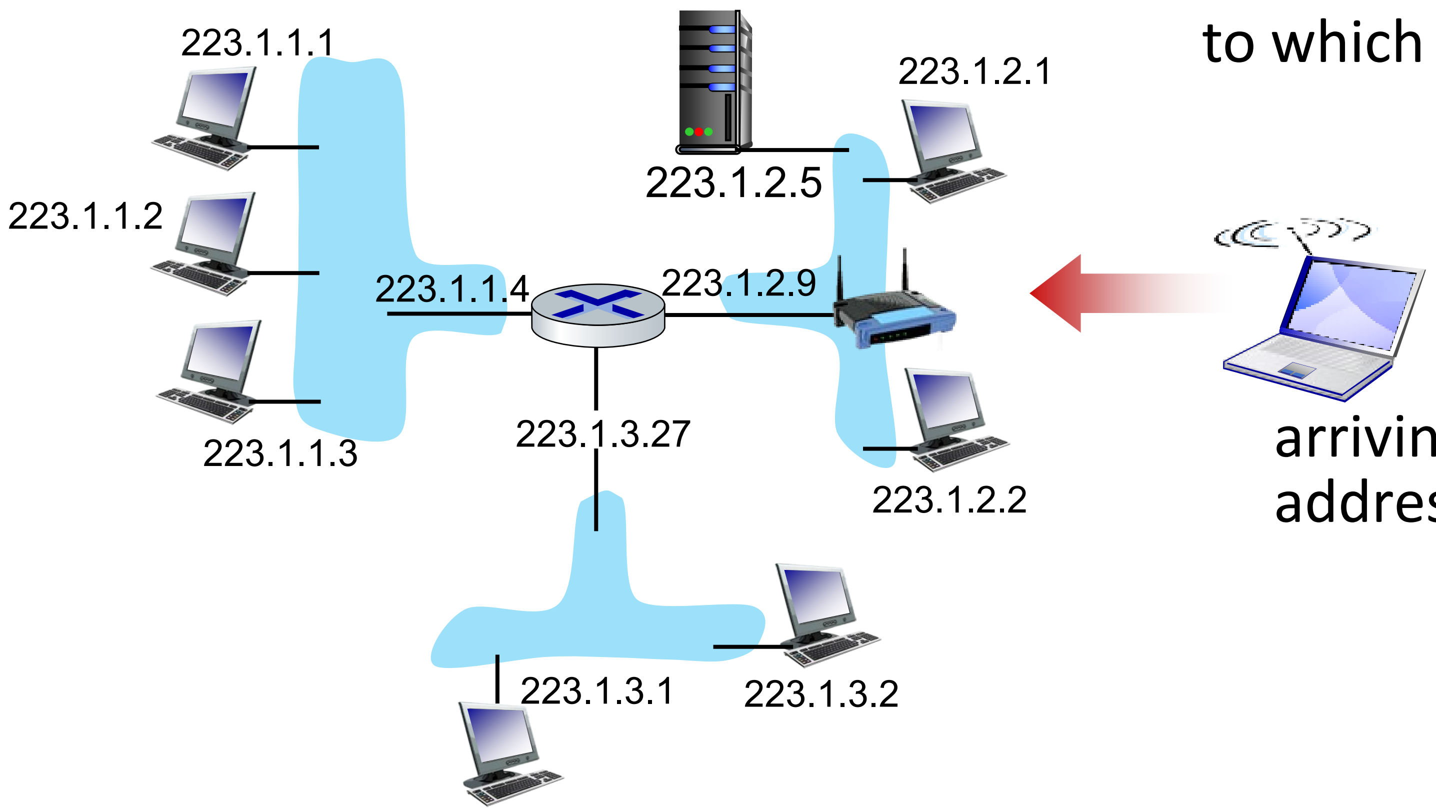
goal: host *dynamically* obtains IP address from network server when it "joins" network

- can renew its lease on address in use
- allows reuse of addresses (only hold address while connected/on)
- support for mobile users who join/leave network

DHCP overview:

- host broadcasts DHCP discover msg [optional]
- DHCP server responds with DHCP offer msg [optional]
- host requests IP address: DHCP request msg
- DHCP server sends address: DHCP ack msg

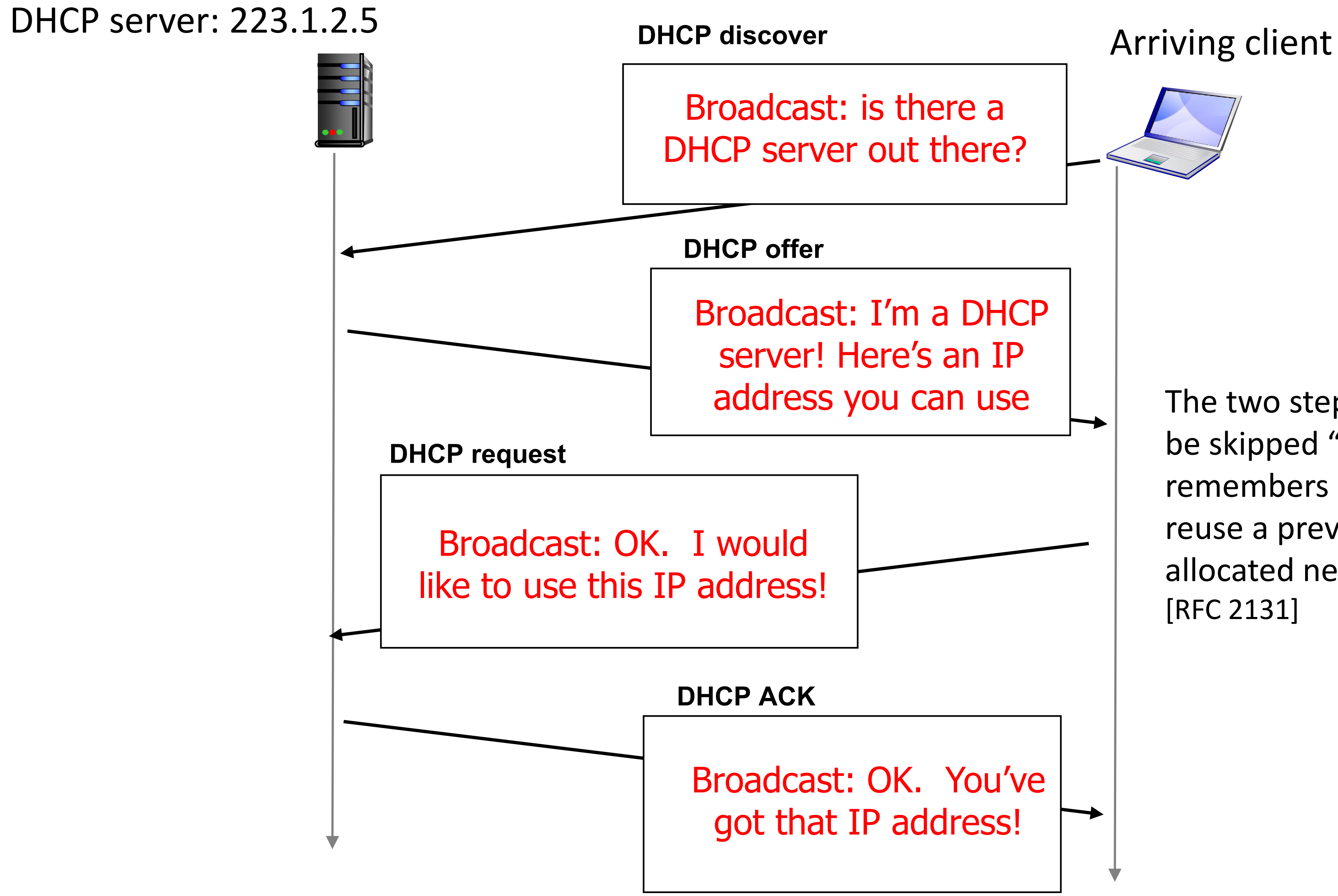
DHCP client-server scenario



Typically, DHCP server will be co-located in router, serving all subnets to which router is attached

arriving DHCP client needs address in this network

DHCP client-server scenario



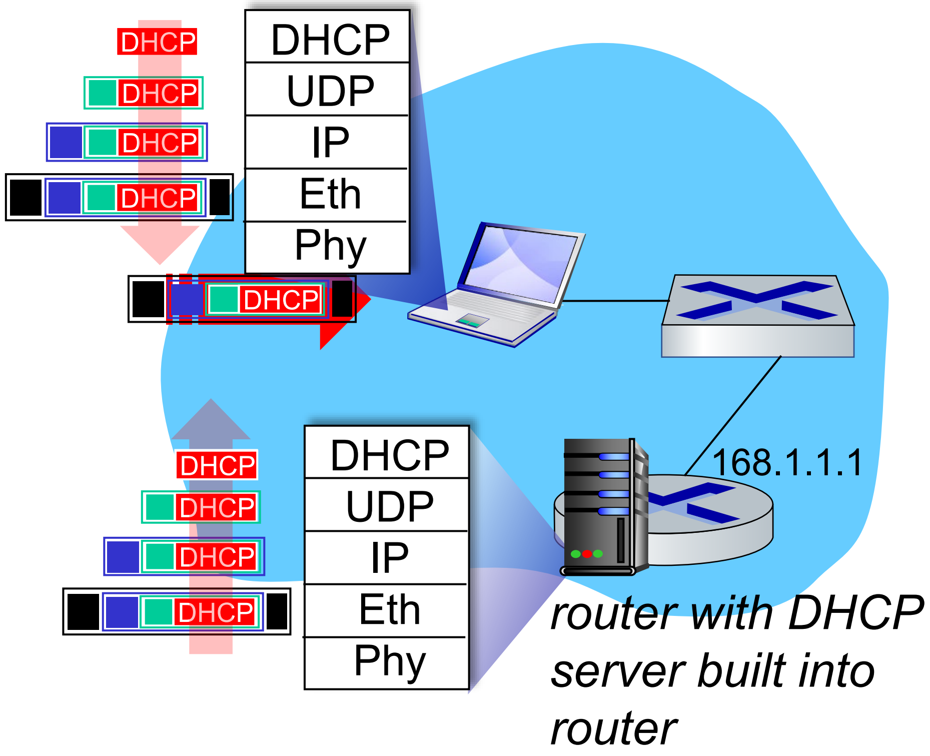
The two steps above can be skipped "if a client remembers and wishes to reuse a previously allocated network address" [RFC 2131]

DHCP: more than IP addresses

DHCP can return more than just allocated IP address on subnet:

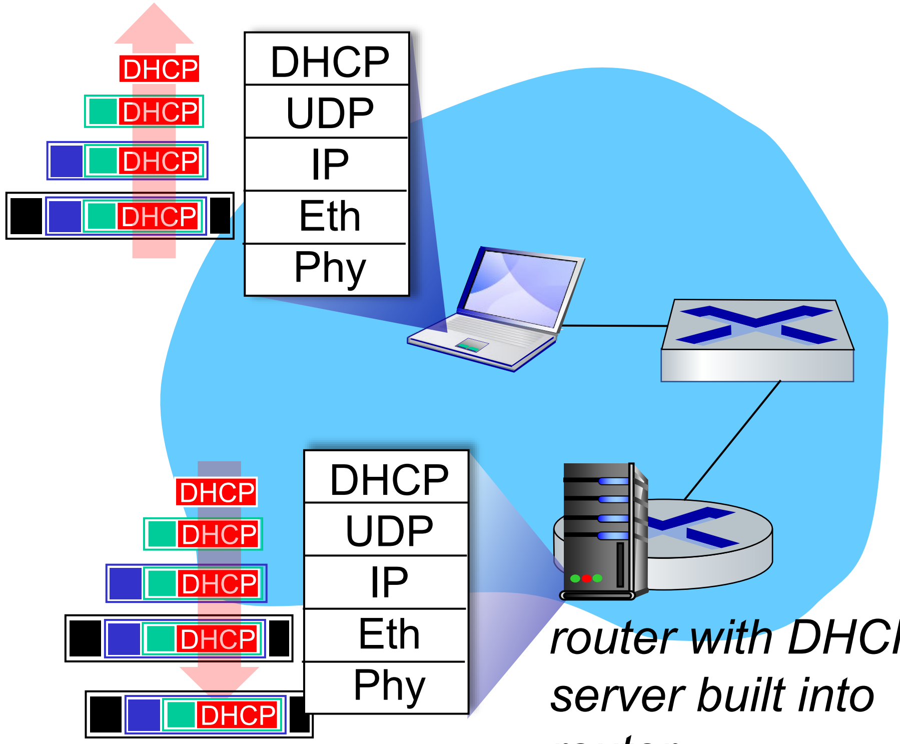
- address of first-hop router for client
- name and IP address of DNS sever
- network mask (indicating network versus host portion of address)

DHCP: example



- Connecting laptop will use DHCP to get IP address, address of first-hop router, address of DNS server.
- DHCP REQUEST message encapsulated in UDP, encapsulated in IP, encapsulated in 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

DHCP: example



router with DHCP server built into router

- DCP server formulates DHCP ACK containing client's IP address, IP address of first-hop router for client, name & IP address of DNS server
- encapsulated DHCP server reply forwarded to client, demuxing up to DHCP at client
- client now knows its IP address, name and IP address of DNS server, IP address of its first-hop router

IP addresses: how to get one?

Q: how does *network* get subnet part of IP address?

A: gets allocated portion of its provider ISP's address space

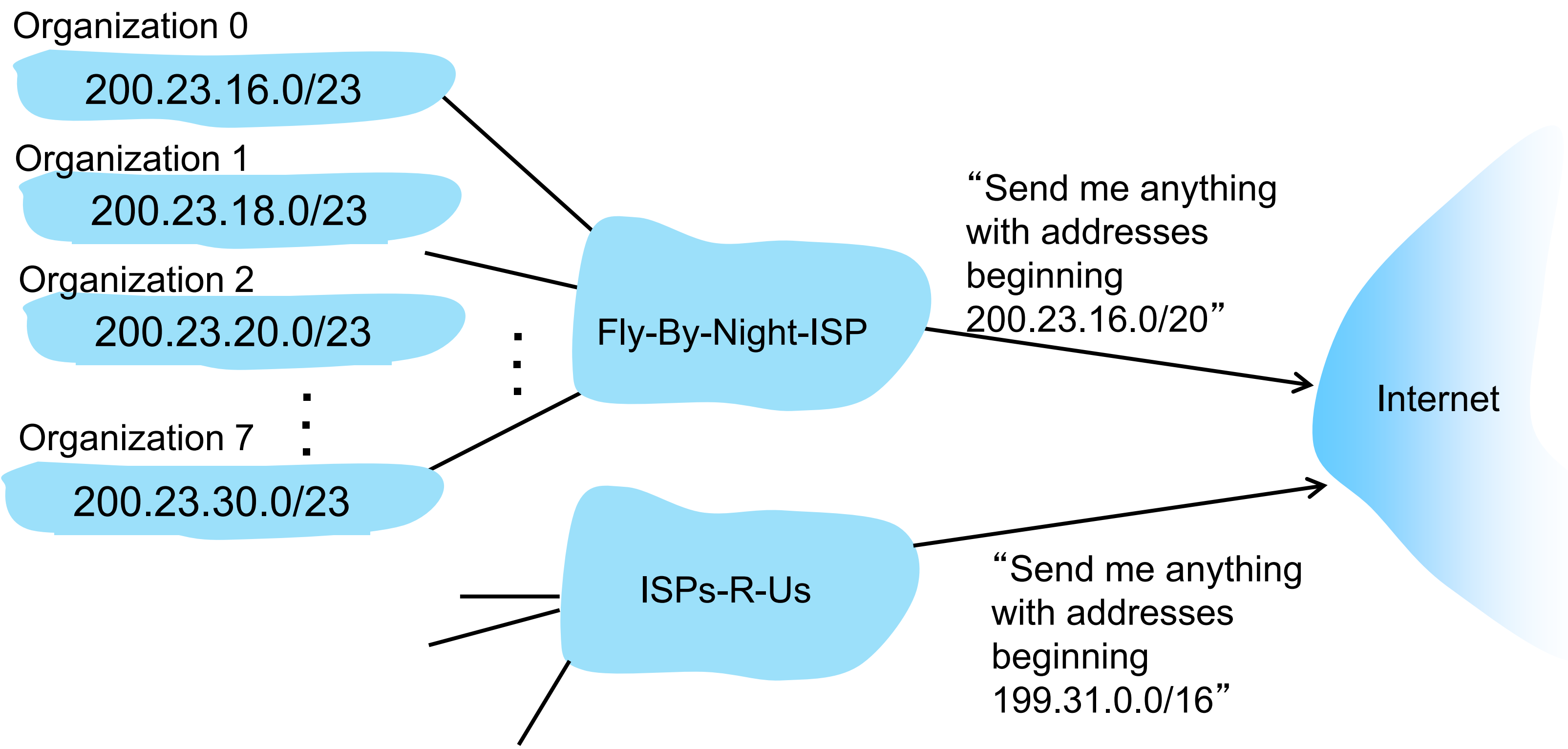
ISP's block	11001000	00010111	00010000	00000000	200.23.16.0/20

ISP can then allocate out its address space in 8 blocks:

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 7	11001000	00010111	00011110	00000000	200.23.30.0/23

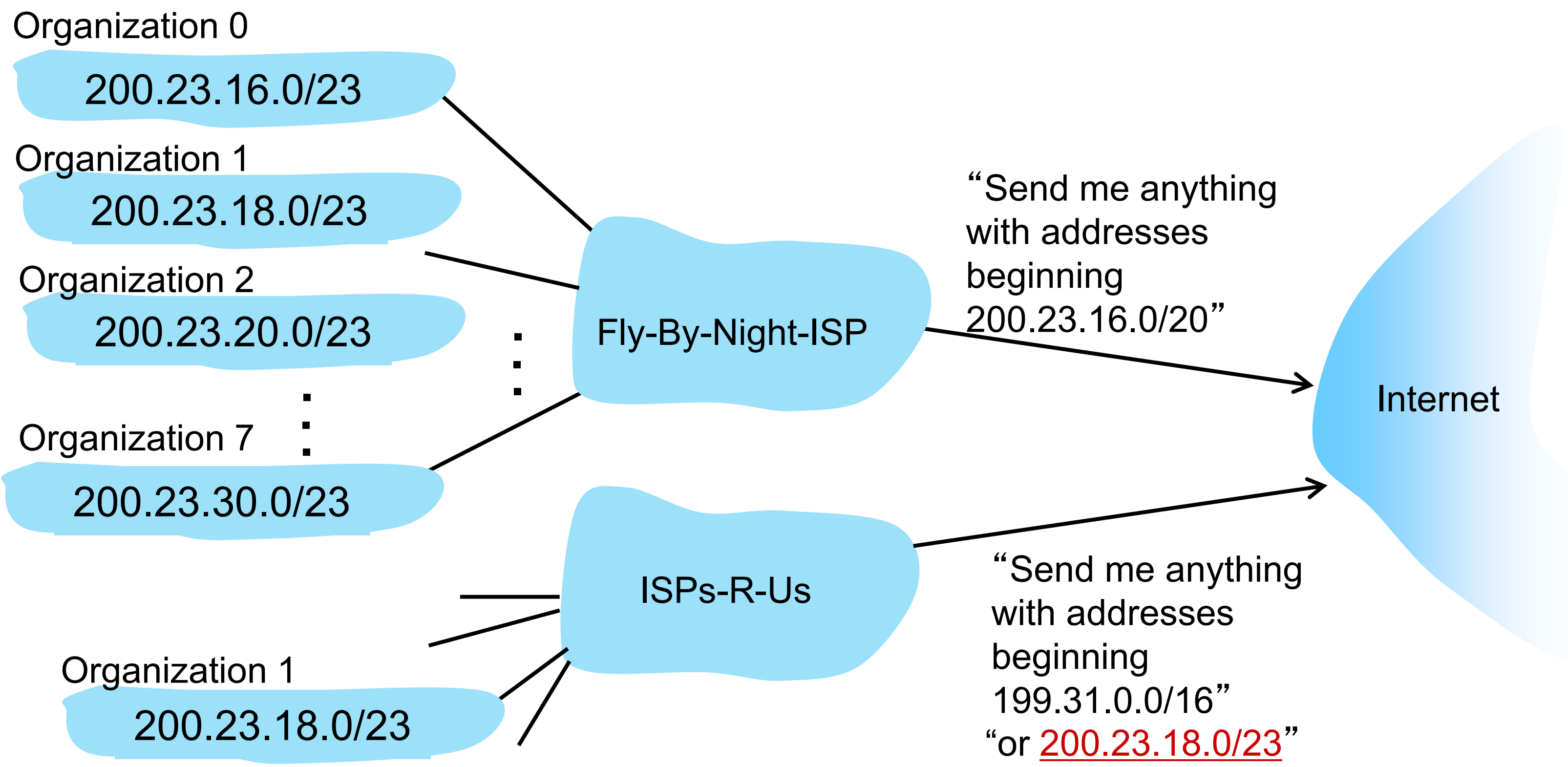
Hierarchical addressing: route aggregation

hierarchical addressing allows efficient advertisement of routing information:



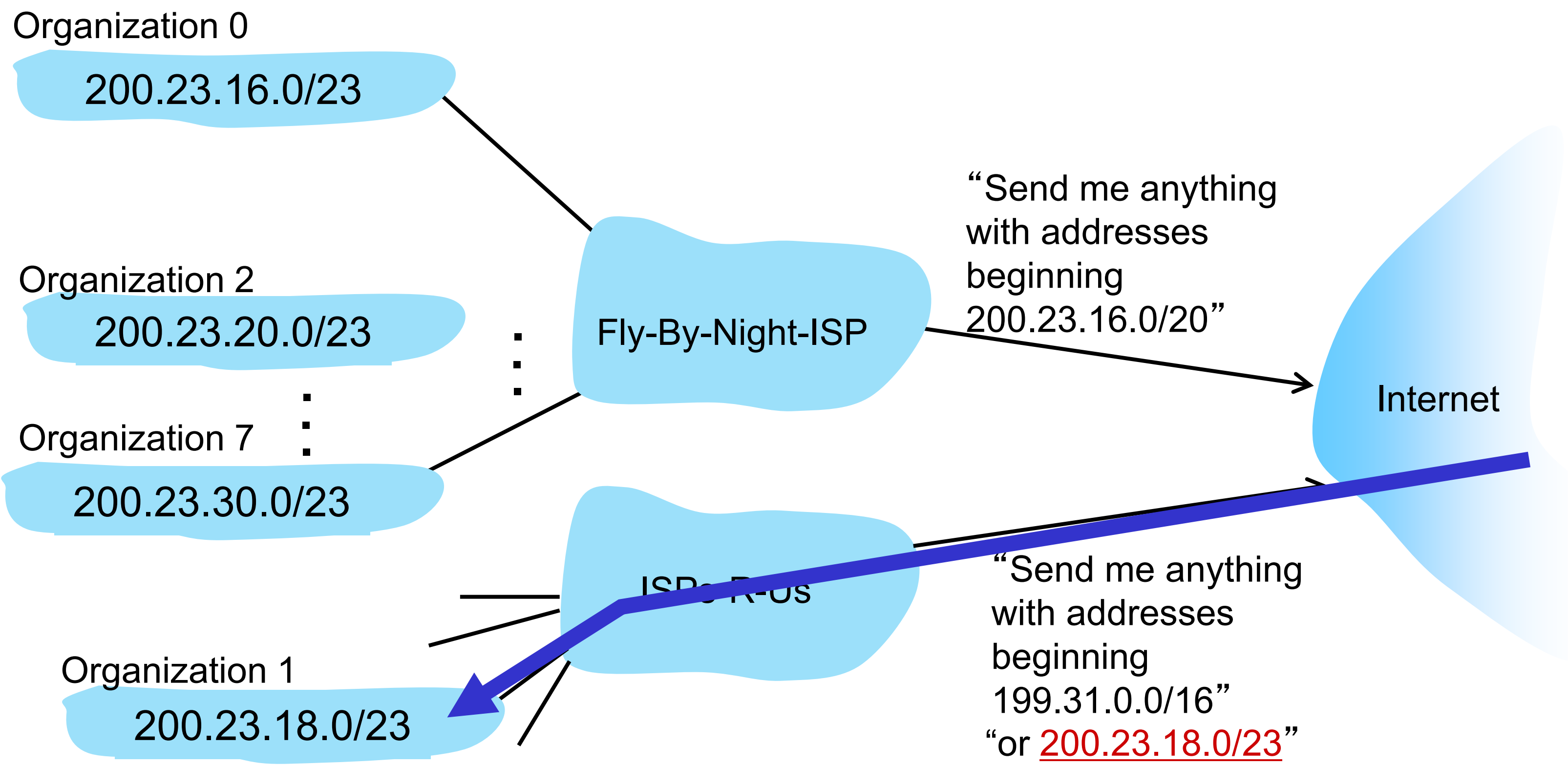
Hierarchical addressing: more specific routes

- Organization 1 moves from Fly-By-Night-ISP to ISPs-R-Us
- ISPs-R-Us now advertises a more specific route to Organization 1



Hierarchical addressing: more specific routes

- Organization 1 moves from Fly-By-Night-ISP to ISPs-R-Us
- ISPs-R-Us now advertises a more specific route to Organization 1



IP addressing: last words ...

Q: how does an ISP get block of addresses?

A: ICANN: Internet Corporation for Assigned Names and Numbers http://www.icann.org/

- allocates IP addresses, through 5 regional registries (RRs) (who may then allocate to local registries)
- manages DNS root zone, including delegation of individual TLD (.com, .edu , …) management

Q: are there enough 32-bit IP addresses?

- ICANN allocated last chunk of IPv4 addresses to RRs in 2011
- NAT (next) helps IPv4 address space exhaustion
- IPv6 has 128-bit address space

"Who the hell knew how much address space we needed?" Vint Cerf (reflecting on decision to make IPv4 address 32 bits long)