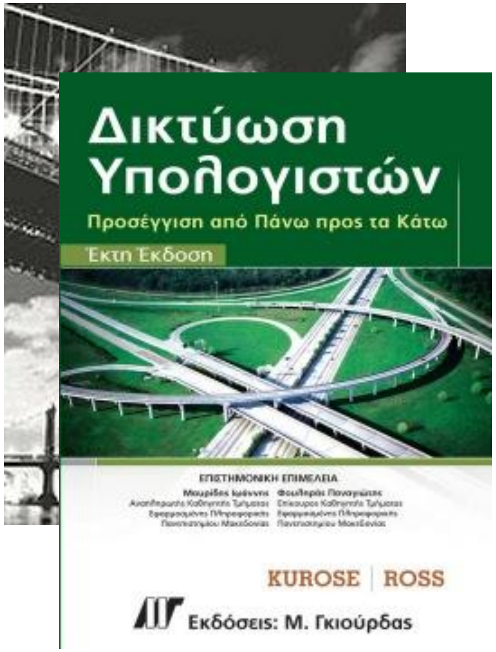
Δίκτυα Υπολογιστών ΙΙ

Κώστας Μαγκούτης
Επίκουρος Καθηγητής
Τμήμα Μηχανικών Η/Υ και Πληροφορικής
Πανεπιστήμιο Ιωαννίνων

Course information

- *introductory* course in computer networking
- course materials:
  - text: *Computer Networking: A Top Down Approach Featuring the Internet*, J. Kurose & K. Ross

    Δικτύωση Υπολογιστών, 6η Έκδοση
    Κωδικός Βιβλίου στον Εύδοξο: 33094885
    Έκδοση: 6η Εκδ./2013
    Συγγραφείς: J.F. Kurose, K.W. Ross
    ISBN: 978-960-512-6575
    Τύπος: Σύγγραμμα
    Διαθέτης (Εκδότης): Χ. ΓΚΙΟΥΡΔΑ & ΣΙΑ ΕΕ



Course information (more)

- class WWW site:

  http://www.cse.uoi.gr/~magoutis/MYY801

  http://piazza.com/uoi.gr/spring2017/myy801

- *everything* is posted on this site
  - announcements
  - syllabus
  - class notes
  - assignments

Course information (more)

- workload:

Coursework	amount	weight %
Homeworks	3	30 %
Final	1	70 %

Course information (more)

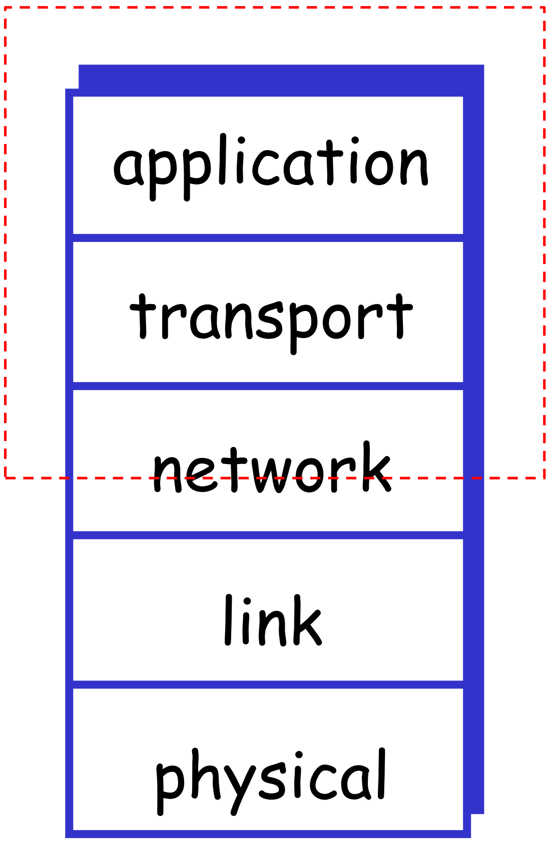
- Lectures
  - Tuesdays 9-12pm I5
  - (labs) Tuesdays 12-2pm (will be announced)

- Staff available to assist you:
  - By appointment
  - During laboratory sessions

Internet protocol stack

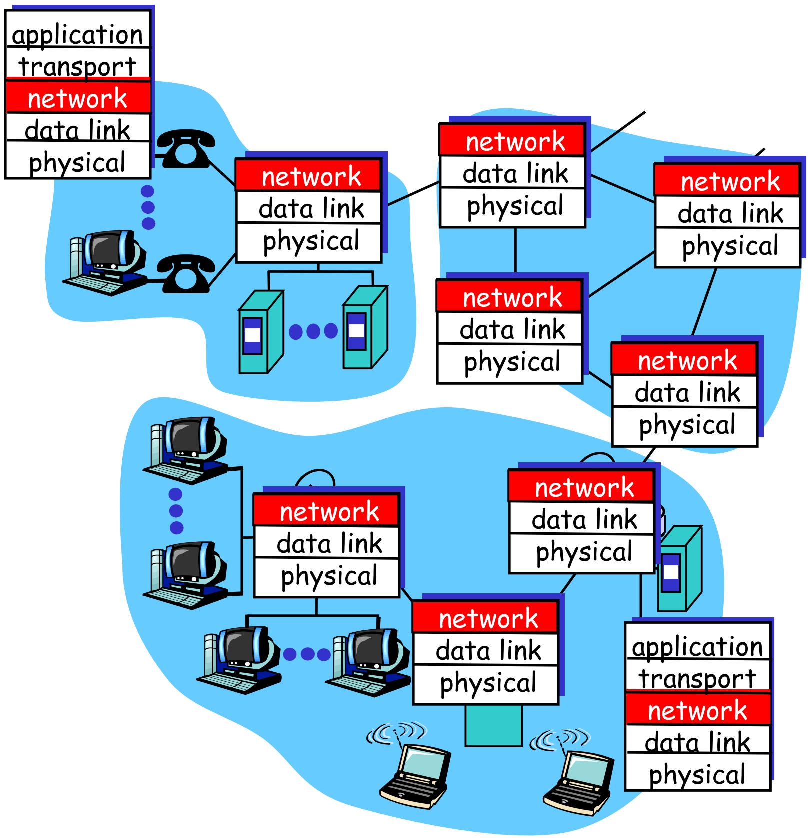
- application: supporting network applications
  - FTP, SMTP, HTTP
- transport: host-host data transfer
  - TCP, UDP
- network: routing of datagrams from source to destination
  - IP, routing protocols
- link: data transfer between neighboring network elements
  - PPP, Ethernet
- physical: bits "on the wire"

Focus of this course



Chapter 4: Network layer

- transport segment from sending to receiving host
- on sending side encapsulates segments into datagrams
- on rcving side, delivers segments to transport layer
- network layer protocols in *every* host, router
- Router examines header fields in all IP datagrams passing through it



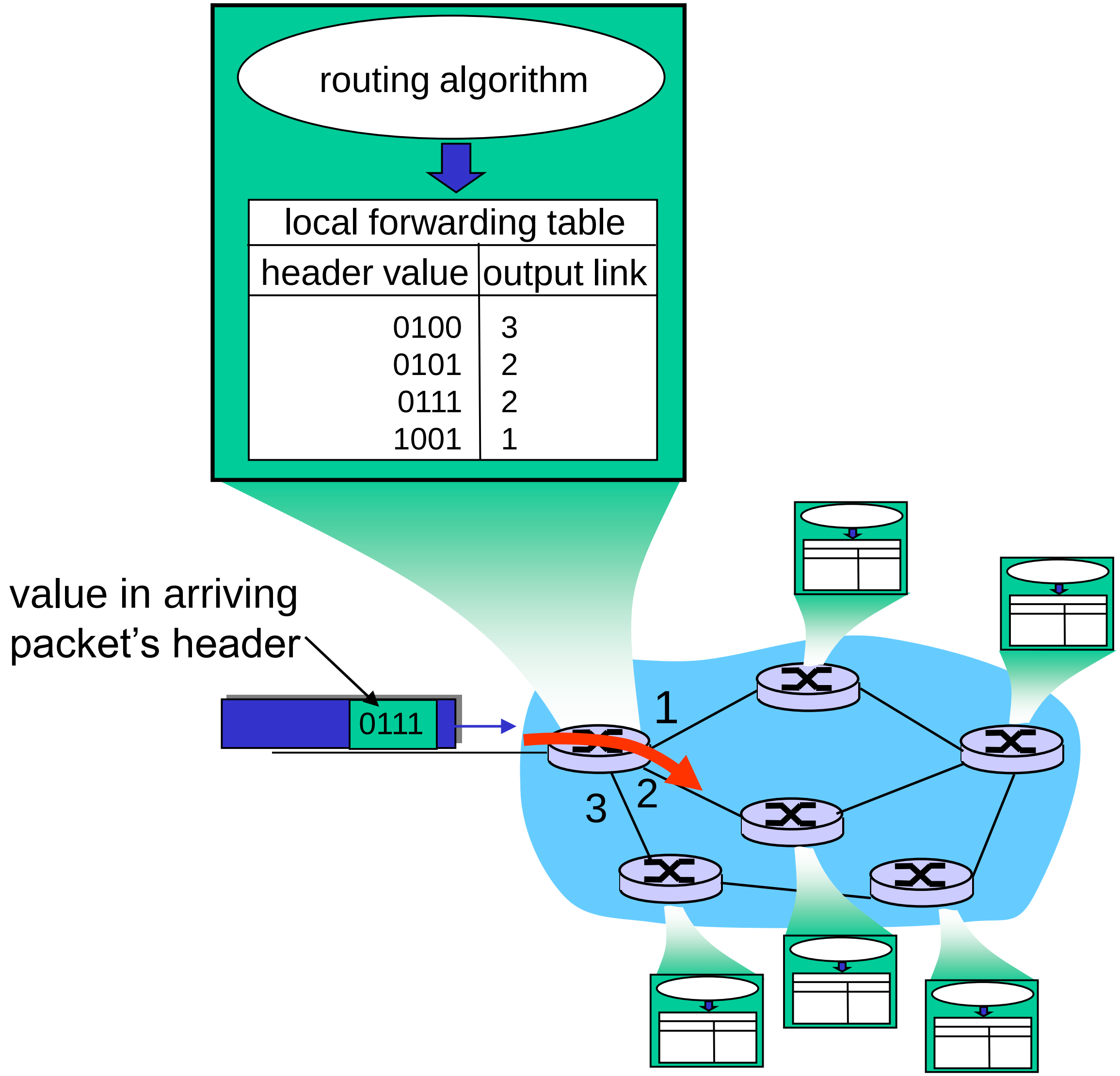
Key Network-Layer Functions

- *forwarding:* move packets from router's input to appropriate router output
- *routing:* determine route taken by packets from source to dest.
  - *Routing algorithms*

analogy:

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

Interplay between routing and forwarding

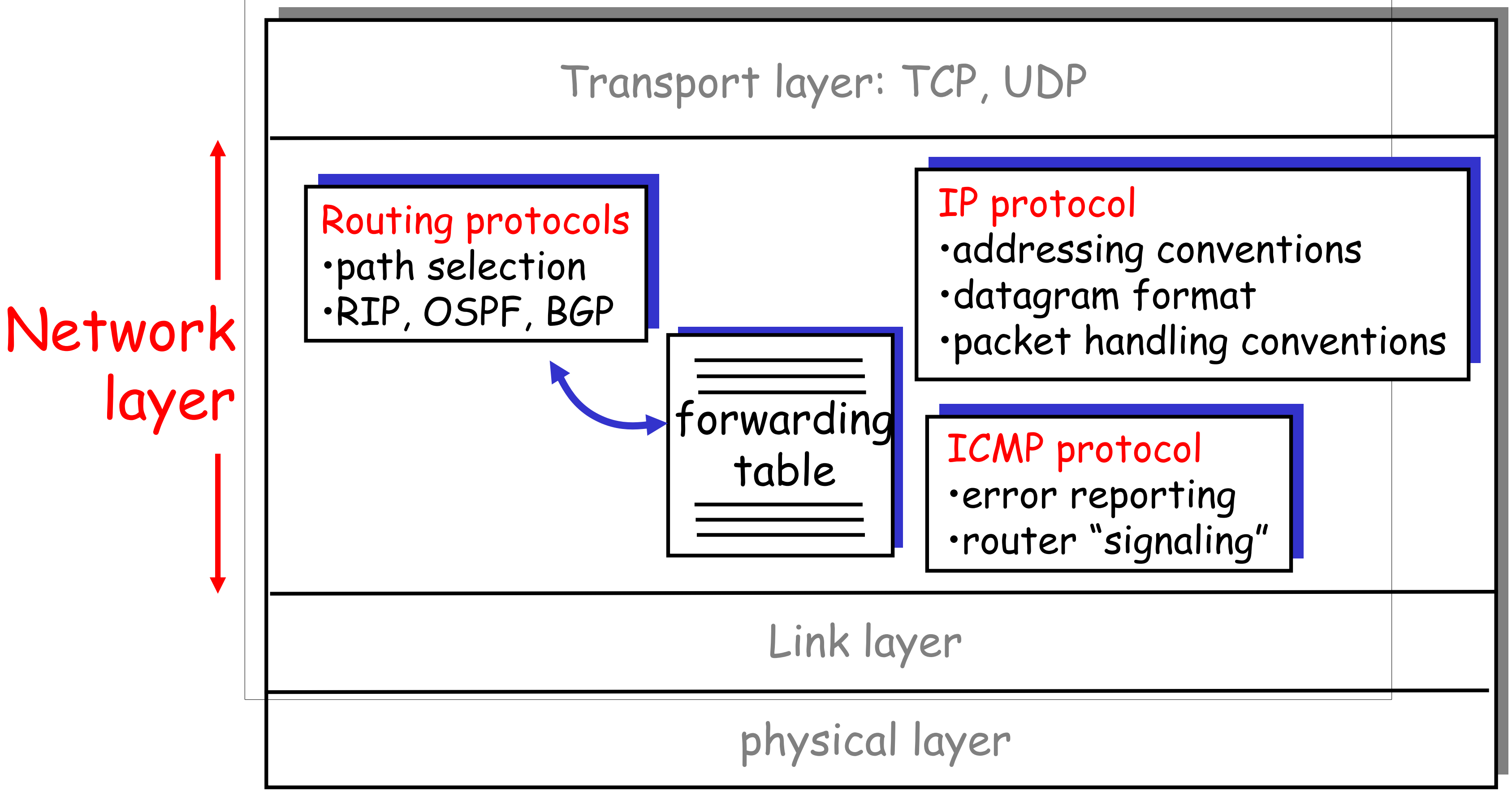


Chapter 4: Network Layer

- IP: Internet Protocol
  - Datagram format
  - IPv4 addressing
  - ICMP
  - IPv6
- Routing in the Internet
  - Hierarchical routing

The Internet Network layer

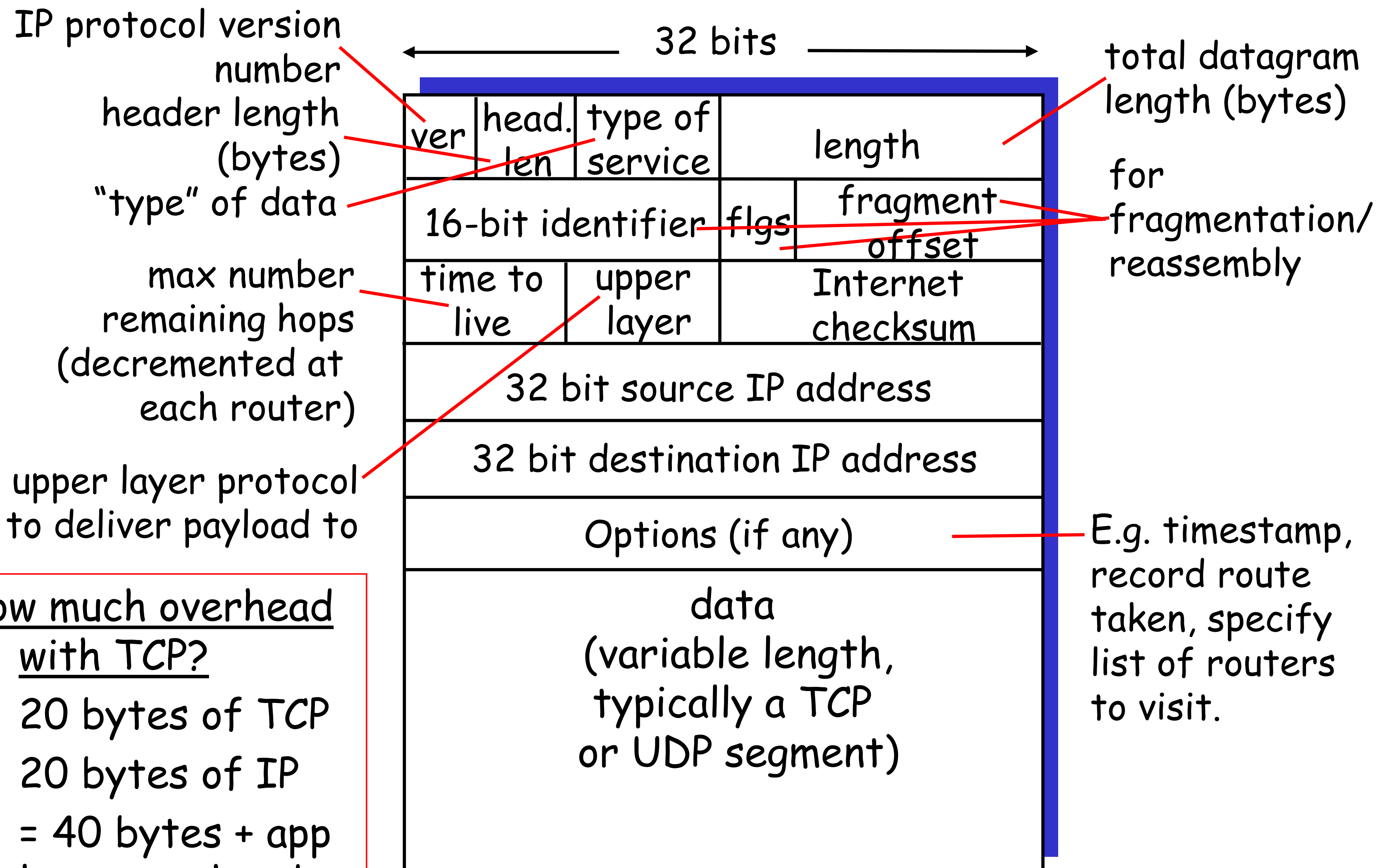
Host, router network layer functions:



Chapter 4: Network Layer

- IP: Internet Protocol
  - Datagram format
  - IPv4 addressing
  - ICMP
  - IPv6
- Routing in the Internet
  - Hierarchical routing

IP datagram format

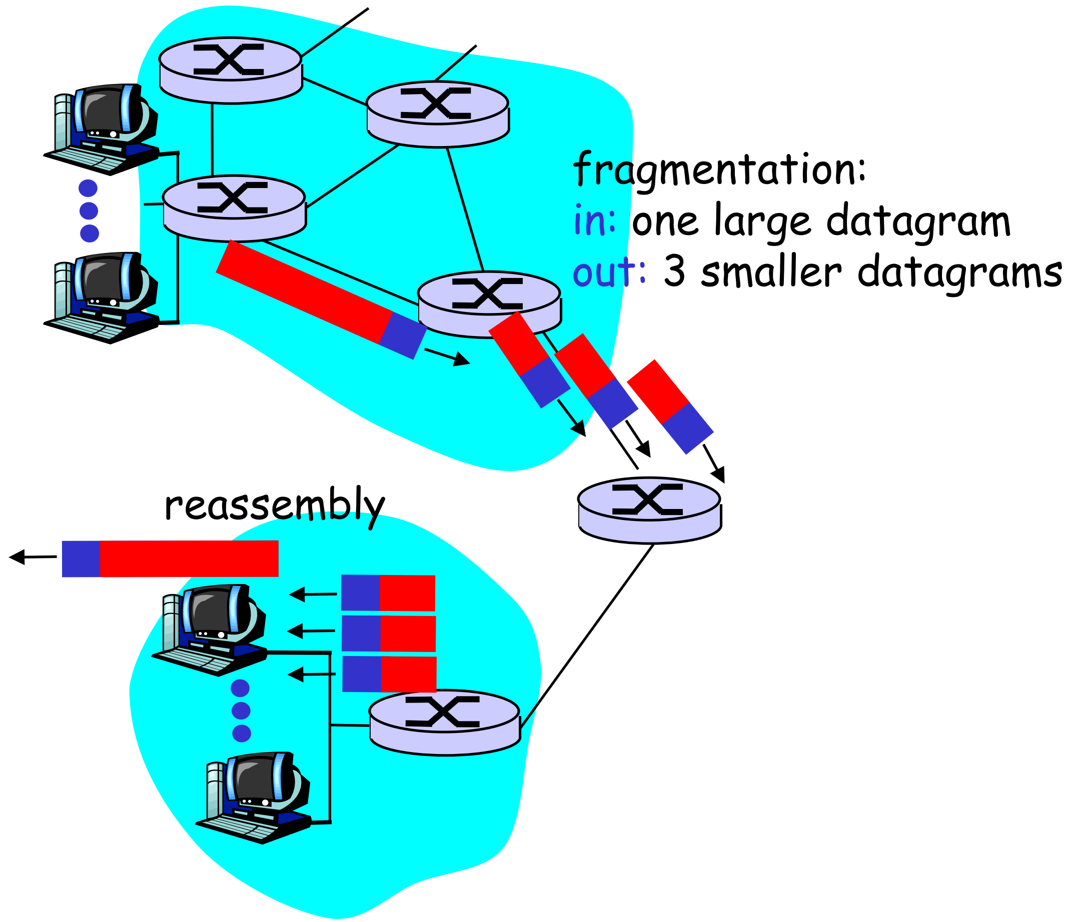


how much overhead with TCP?

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

IP Fragmentation & Reassembly

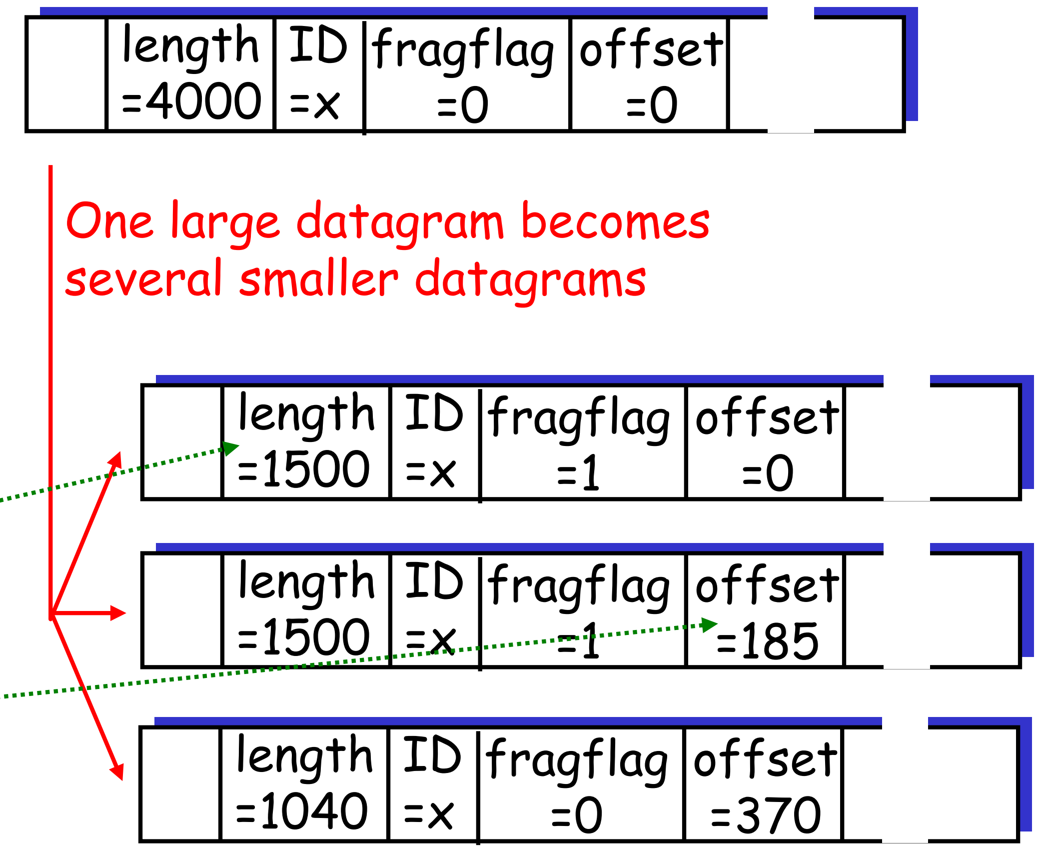
- network links have MTU (max.transfer size) - largest possible link-level frame.
  - different link types, different MTUs
- large IP datagram divided ("fragmented") within net
  - one datagram becomes several datagrams
  - "reassembled" only at final destination
  - IP header bits used to identify, order related fragments



IP Fragmentation and Reassembly

Example

- 4000 byte datagram
- MTU = 1500 bytes

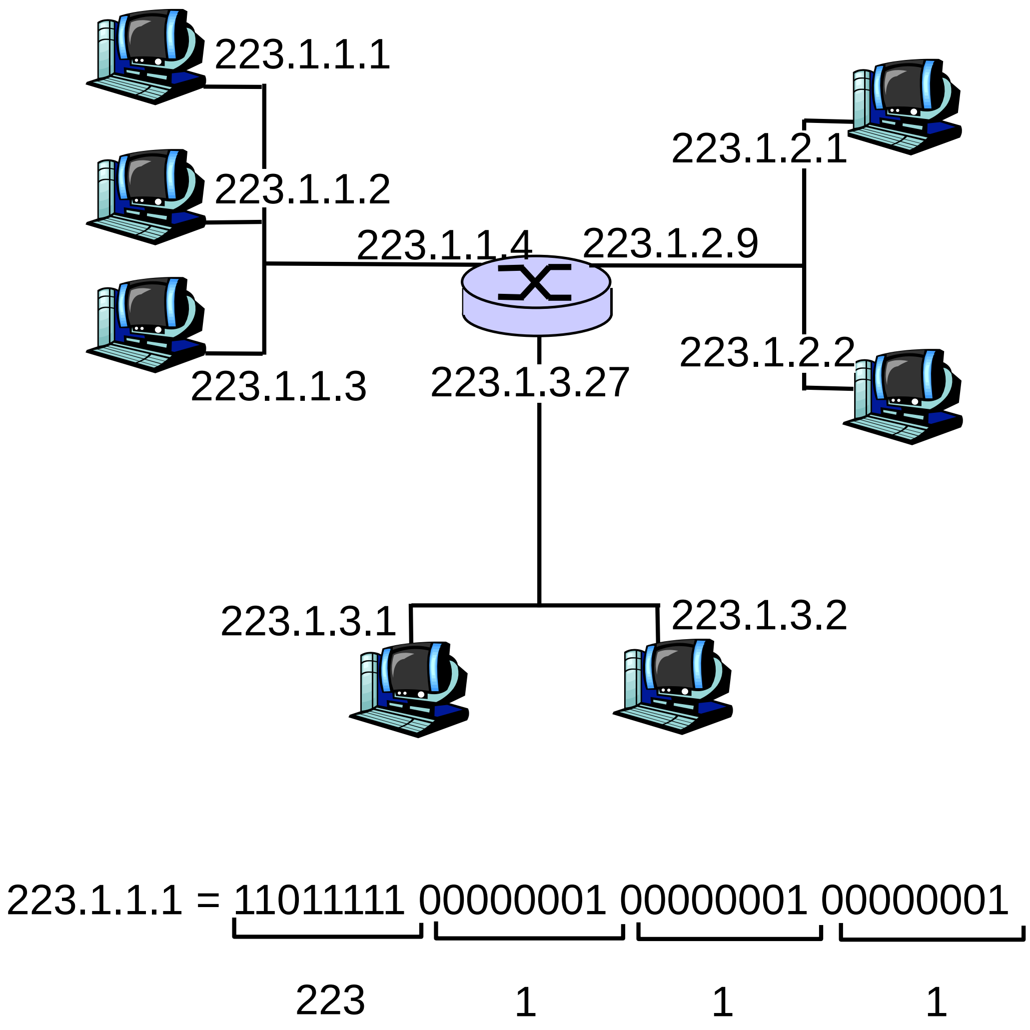


Chapter 4: Network Layer

- IP: Internet Protocol
  - Datagram format
  - IPv4 addressing
  - ICMP
  - IPv6
- Routing in the Internet
  - Hierarchical routing

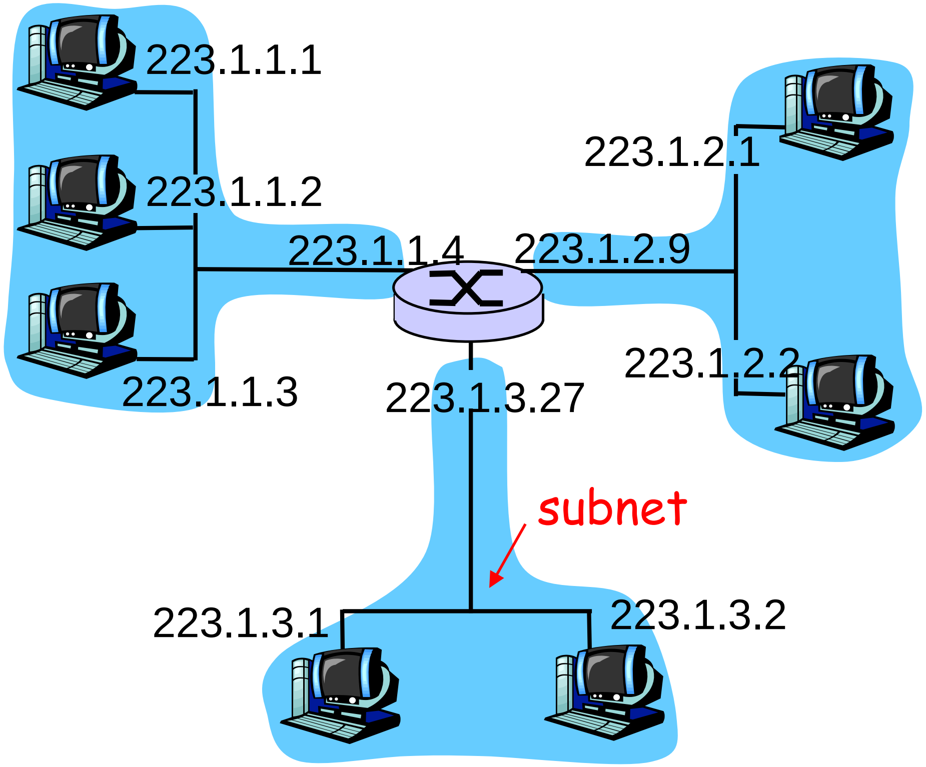
IP Addressing: introduction

- **IP address:** 32-bit identifier for host, router *interface*
- *interface:* connection between host/router and physical link
  - router's typically have multiple interfaces
  - host typically has one interface
  - IP addresses associated with each interface



Subnets

- IP address:
  - subnet part (high order bits)
  - host part (low order bits)
- *What's a subnet ?*
  - device interfaces with same subnet part of IP address
  - can physically reach each other without intervening router

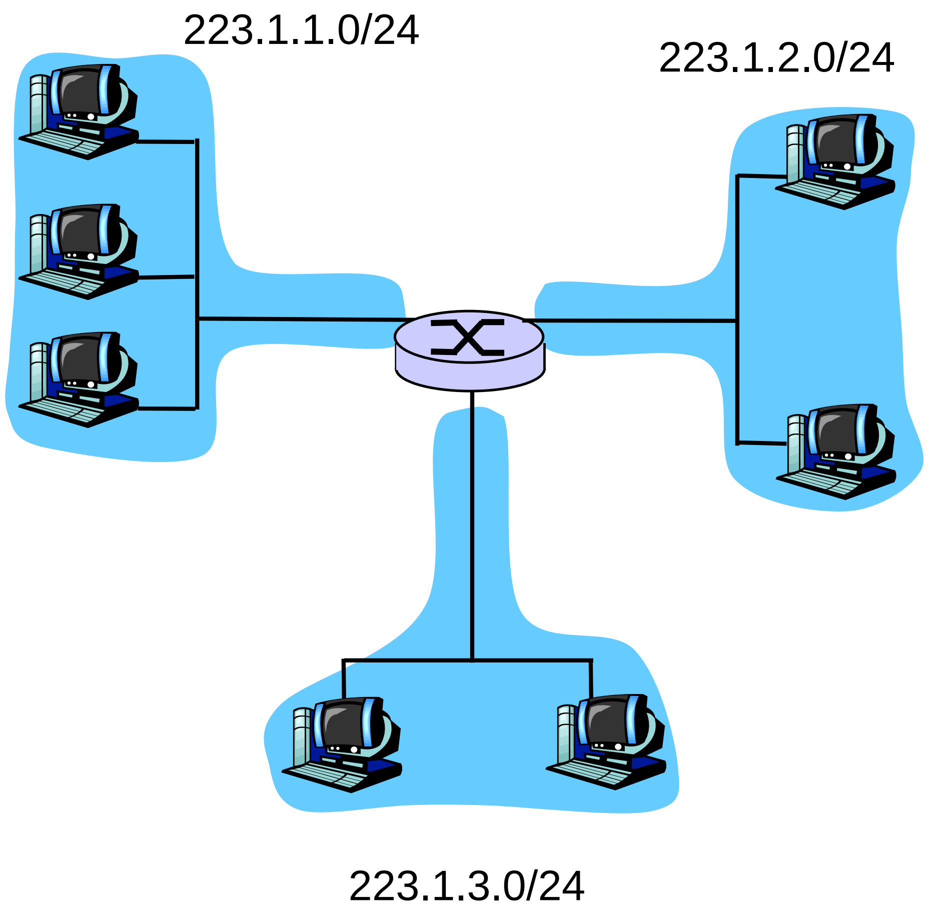


network consisting of 3 subnets

Subnets

Recipe

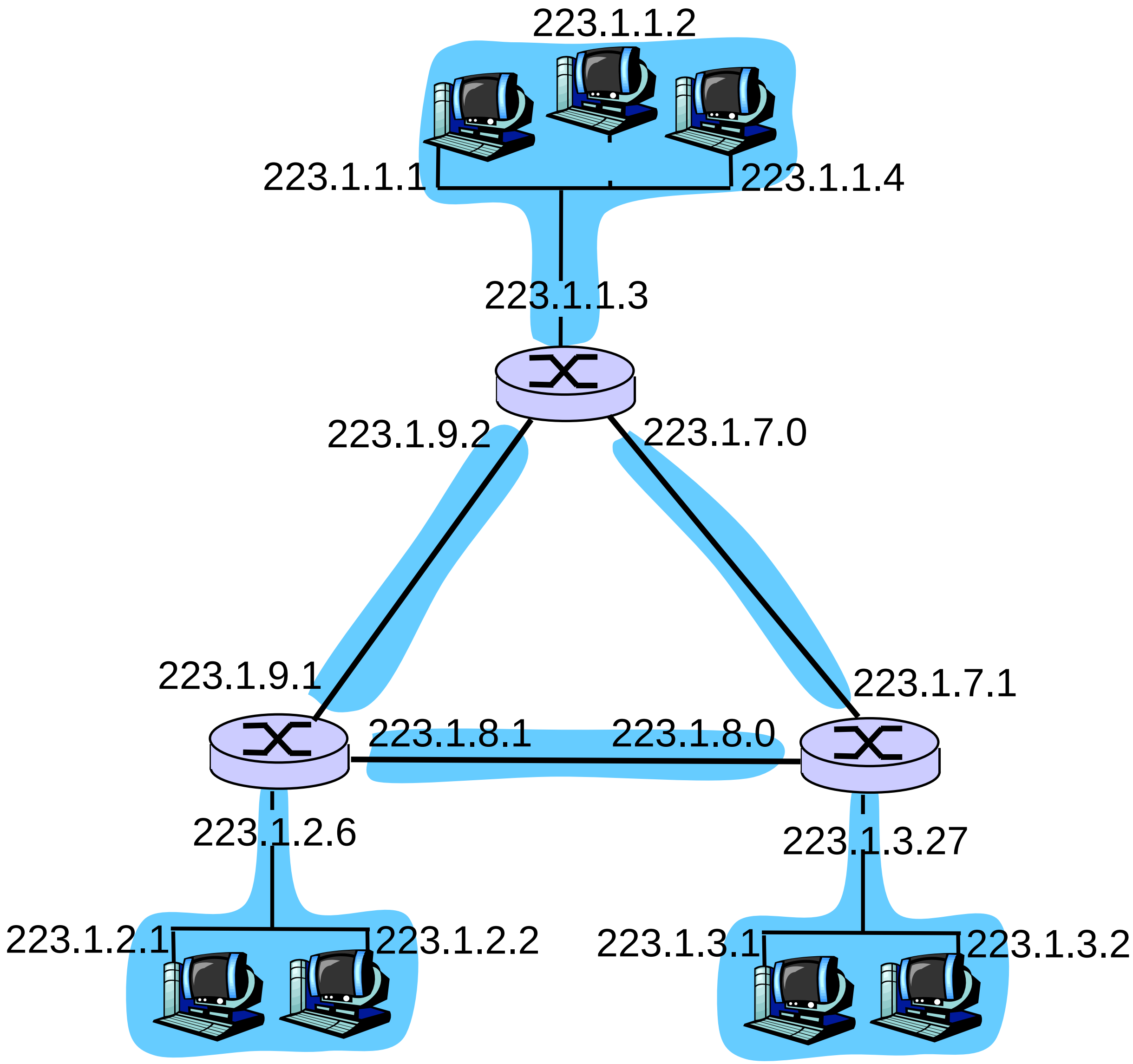
- To determine the subnets, detach each interface from its host or router, creating islands of isolated networks. Each isolated network is called a subnet.



Subnet mask: /24

Subnets

How many?



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

200.23.16.0/23

IP addresses: how to get one?

Q: How does *host* get IP address?

- hard-coded by system admin in a file
  - Wintel: control-panel->network->configuration->tcp/ip->properties
  - UNIX: /etc/rc.config
- DHCP: Dynamic Host Configuration Protocol: dynamically get address from as server
  - "plug-and-play"

(more in next chapter)

IP addresses: how to get one?

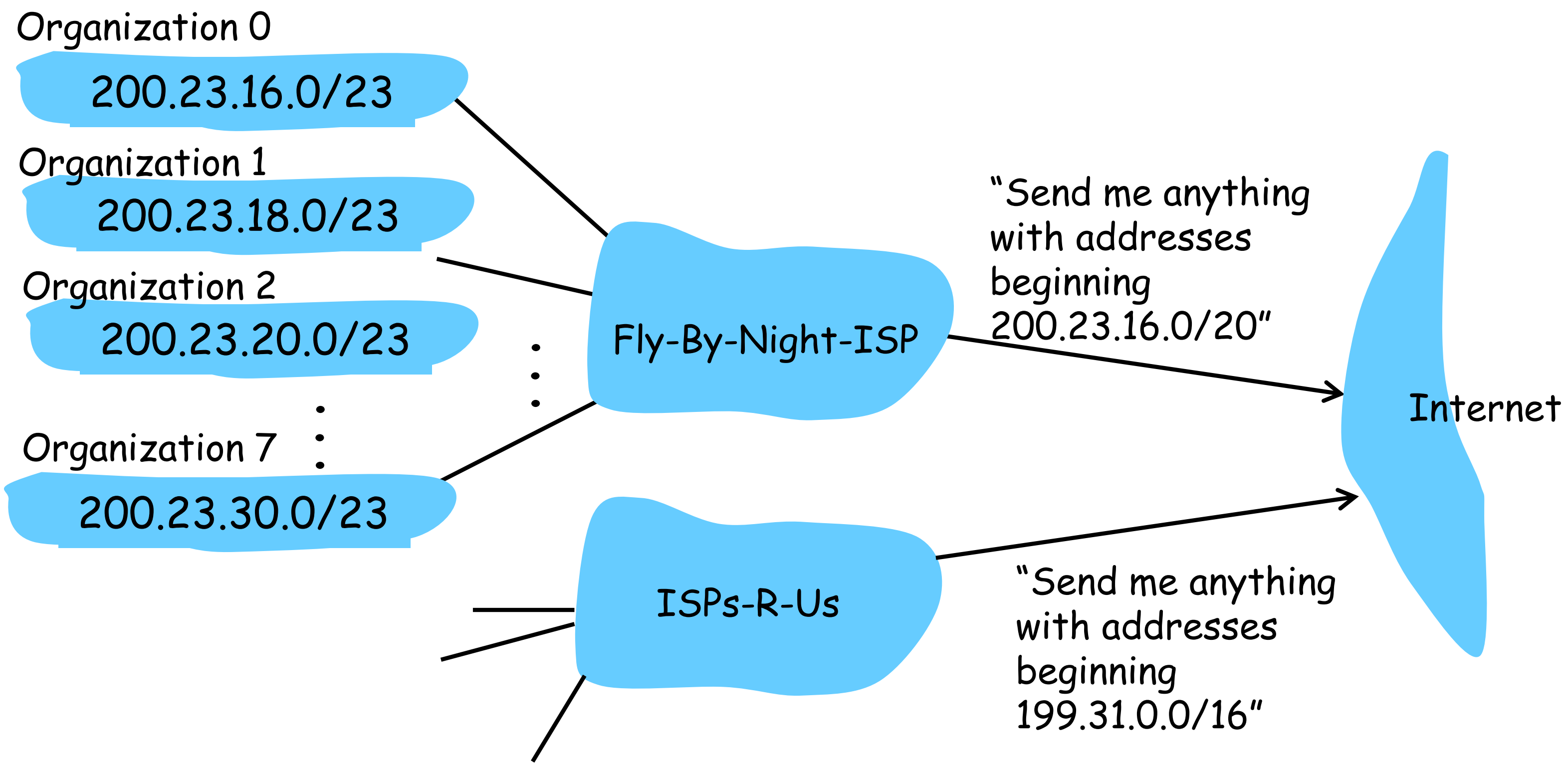
Q: How does *network* get subnet part of IP addr?

A: gets allocated portion of its provider 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 7	11001000 00010111 00011110 00000000	200.23.30.0/23

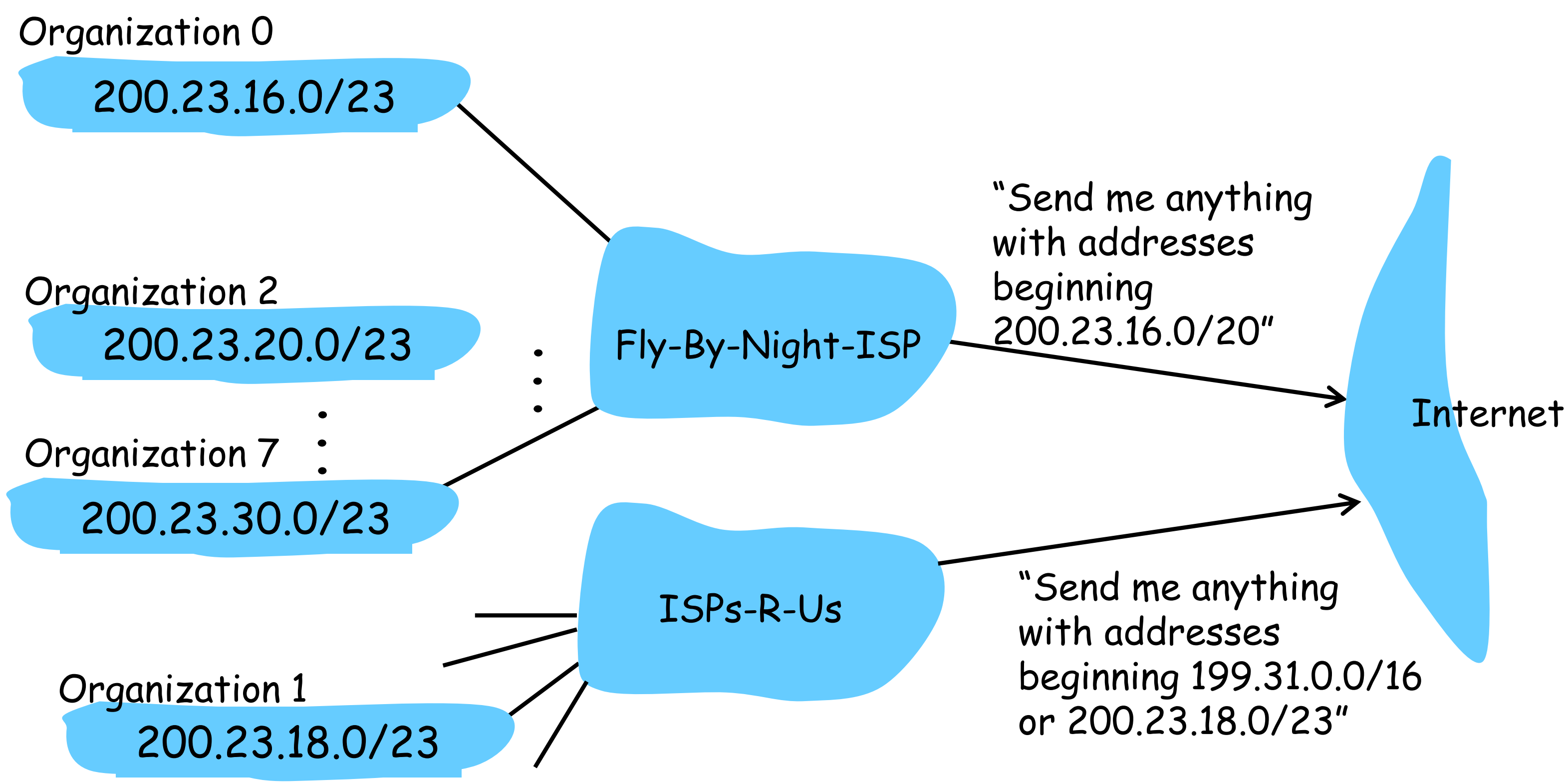
Hierarchical addressing: route aggregation

Hierarchical addressing allows efficient advertisement of routing information:



Hierarchical addressing: more specific routes

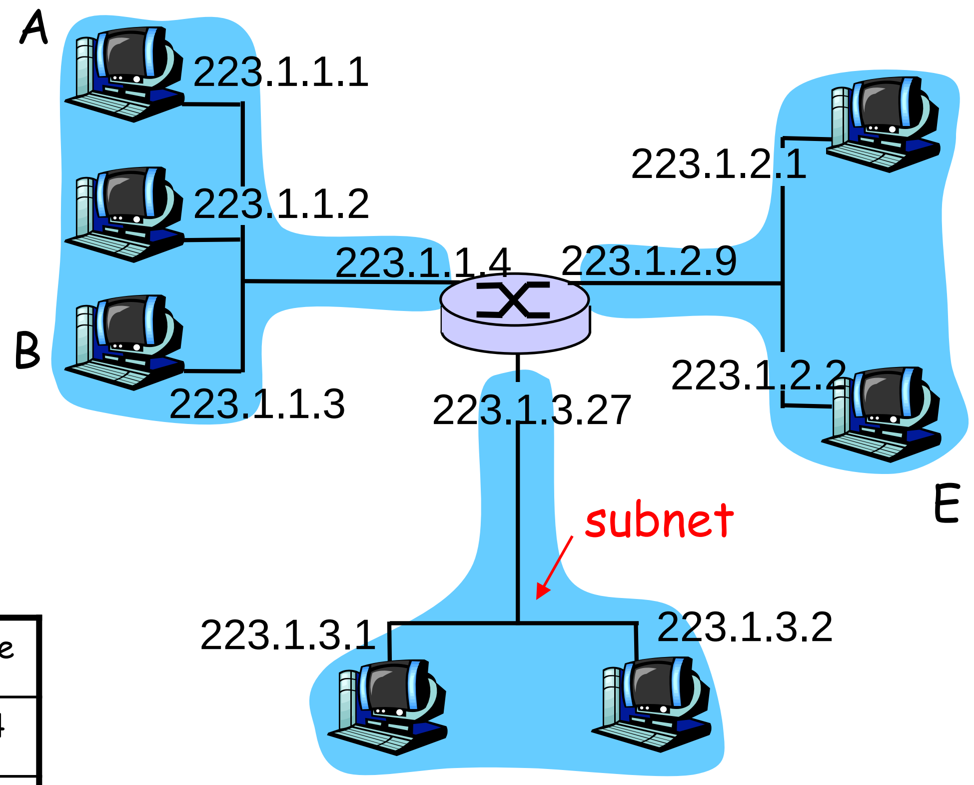
ISPs-R-Us has a more specific route to Organization 1



Moving a datagram from source to destination

Dest.network	Next router	Nhops
223.1.1.0/24		1
223.1.2.0/24	223.1.1.4	2
223.1.3.0/24	223.1.1.4	2

Forwarding table in host A



Dest.network	Next router	Nhops	Interface
223.1.1.0/24	-	1	223.1.1.4
223.1.2.0/24	-	1	223.1.2.9
223.1.3.0/24	-	1	223.1.3.27

Forwarding table in router

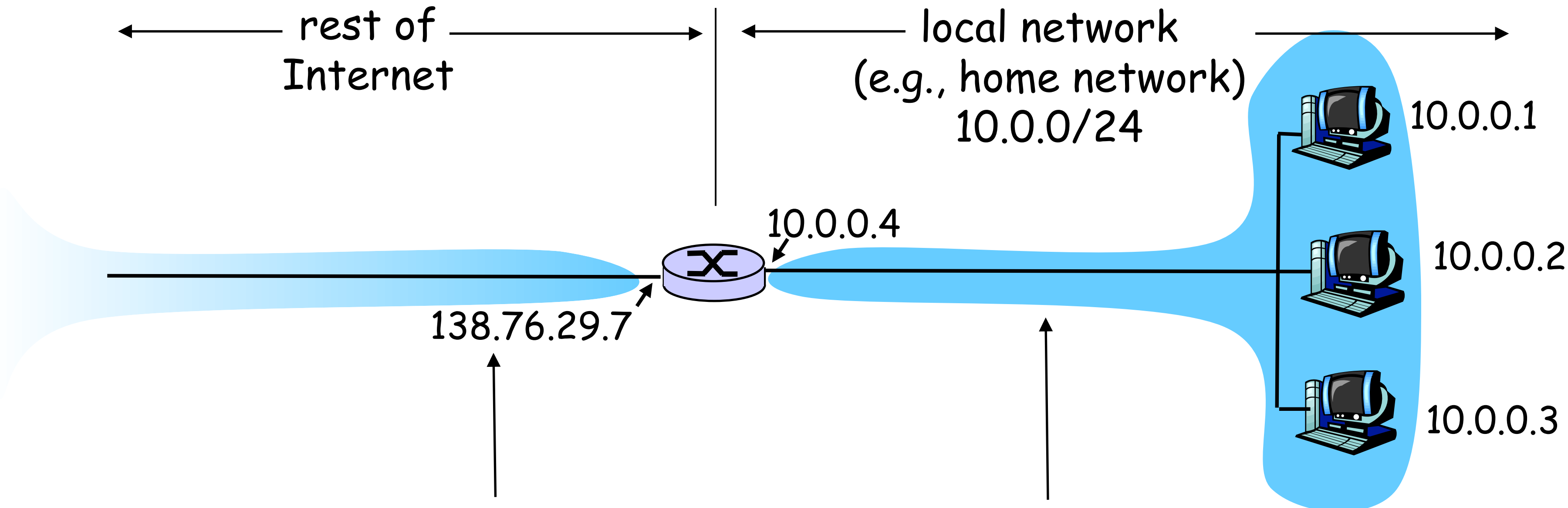
IP addressing: the last word...

Q: How does an ISP get block of addresses?

A: ICANN: Internet Corporation for Assigned Names and Numbers

- allocates addresses
- manages DNS
- assigns domain names, resolves disputes

NAT: Network Address Translation



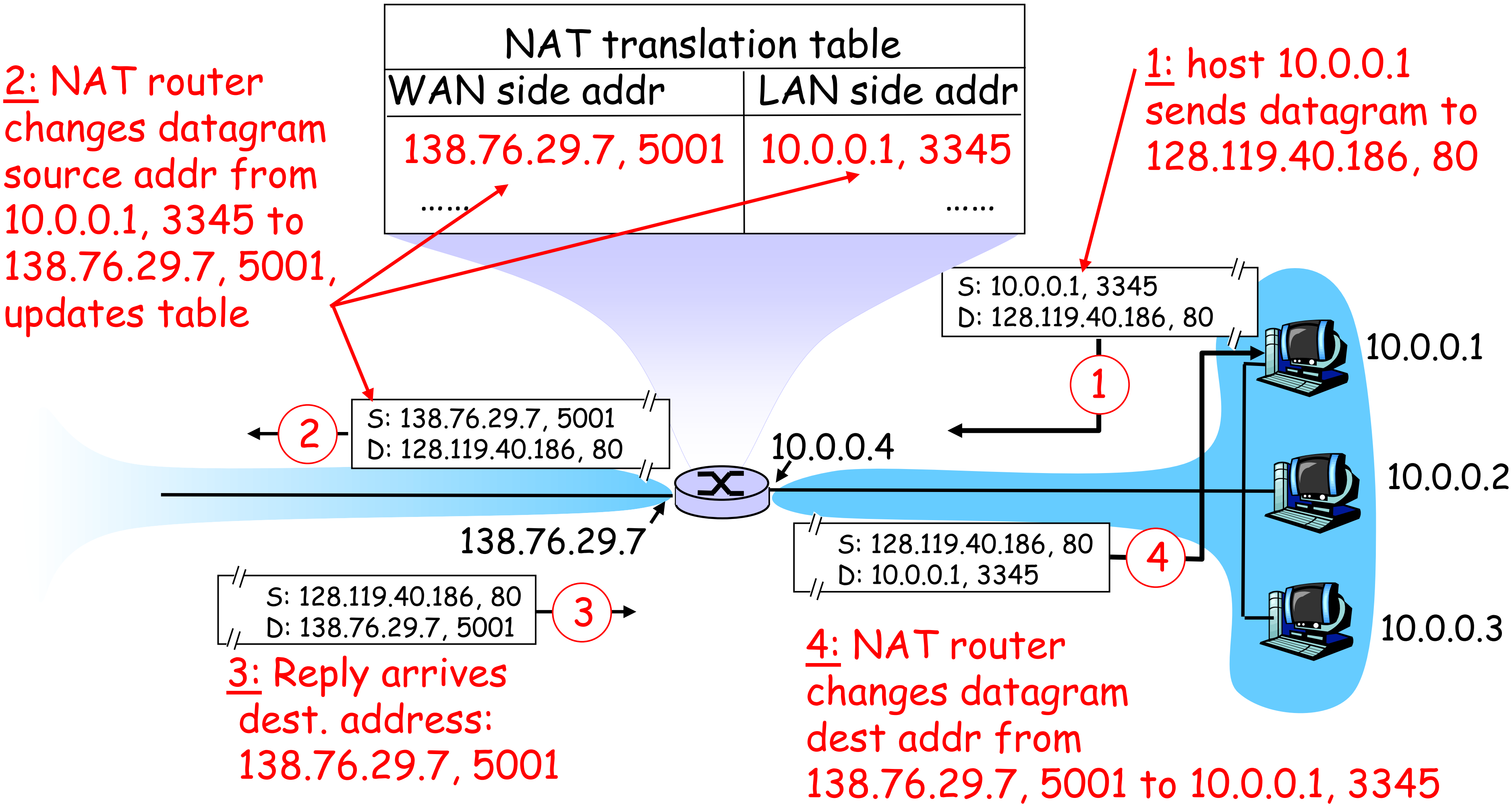
All datagrams *leaving* local network have *same* single source NAT IP address: 138.76.29.7, different source port numbers

Datagrams with source or destination in this network have 10.0.0/24 address for source, destination (as usual)

NAT: Network Address Translation

- Motivation: local network uses just one IP address as far as outside world is concerned:
  - range of addresses not needed from ISP: just one IP address 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, visible by outside world (a security plus).

NAT: Network Address Translation



NAT: Network Address Translation

- 16-bit port-number field:
  - 60,000 simultaneous connections with a single LAN-side address!
- NAT is controversial:
  - routers should only process up to layer 3
  - violates end-to-end argument
  - address shortage should instead be solved by IPv6

Chapter 4: Network Layer

- IP: Internet Protocol
  - Datagram format
  - IPv4 addressing
  - ICMP
  - IPv6
- Routing in the Internet
  - Hierarchical routing

ICMP: Internet Control Message Protocol

- used by hosts & routers to communicate network-level information
  - error reporting: unreachable host, network, port, protocol
  - echo request/reply (used by ping)
- network-layer "above" IP:
  - ICMP msgs carried in IP datagrams
- ICMP message: type, code plus first 8 bytes of IP datagram causing error

Type	Code	description
0	0	echo reply (ping)
3	0	dest. network unreachable
3	1	dest host unreachable
3	2	dest protocol unreachable
3	3	dest port unreachable
3	6	dest network unknown
3	7	dest host unknown
4	0	source quench (congestion control - not used)
8	0	echo request (ping)
9	0	route advertisement
10	0	router discovery
11	0	TTL expired
12	0	bad IP header

Traceroute and ICMP

- Source sends series of UDP segments to dest
  - First has TTL =1
  - Second has TTL=2, etc.
  - Unlikely port number
- When nth datagram arrives to nth router:
  - Router discards datagram
  - And sends to source an ICMP message (type 11, code 0)
  - Message includes name of router& IP address
- When ICMP message arrives, source calculates RTT
- Traceroute does this 3 times

Stopping criterion

- UDP segment eventually arrives at destination host
- Destination returns ICMP "host unreachable" packet (type 3, code 3)
- When source gets this ICMP, stops.

Chapter 4: Network Layer

- IP: Internet Protocol
  - Datagram format
  - IPv4 addressing
  - ICMP
  - IPv6
- Routing in the Internet
  - Hierarchical routing

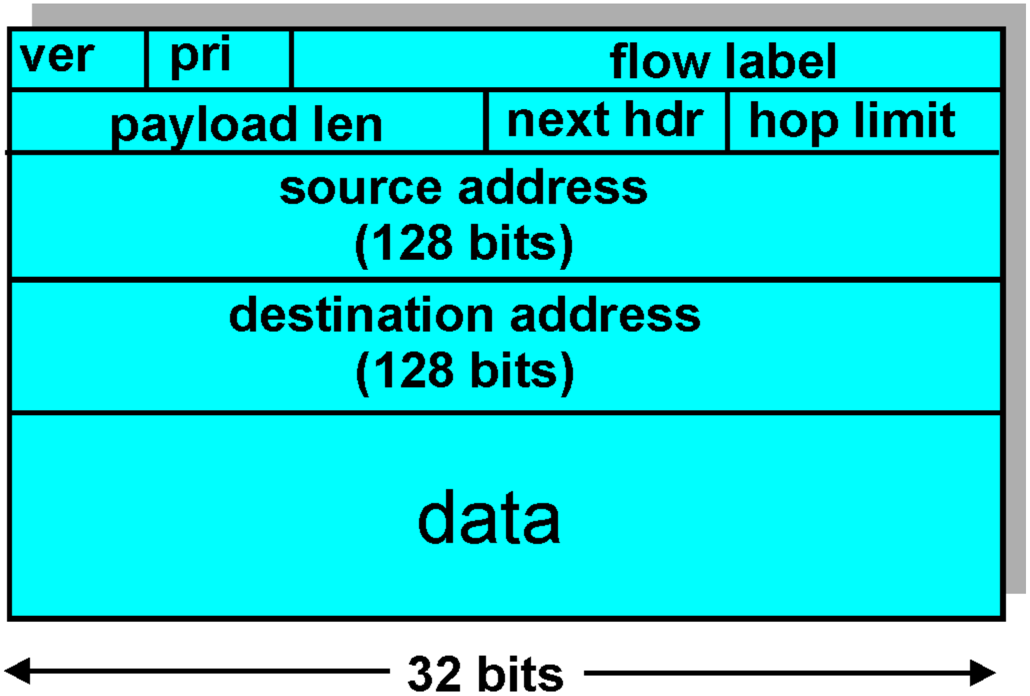
IPv6

- Initial motivation: 32-bit address space soon to be completely allocated.
- Additional motivation:
  - header format helps speed processing/forwarding
  - header changes to facilitate QoS

  IPv6 datagram format:
  - fixed-length 40 byte header
  - no fragmentation allowed

IPv6 Header (Cont)

Priority: identify priority among datagrams in flow
Flow Label: identify datagrams in same "flow."
(concept of "flow" not well defined).
Next header: identify upper layer protocol for data



Other Changes from IPv4

- *Checksum*: removed entirely to reduce processing time at each hop
- *Options:* allowed, but outside of header, indicated by "Next Header" field
- *ICMPv6:* new version of ICMP
  - additional message types, e.g. "Packet Too Big"
  - multicast group management functions

Chapter 4: Network Layer

- IP: Internet Protocol
  - Datagram format
  - IPv4 addressing
  - ICMP
  - IPv6
- Routing in the Internet
  - Hierarchical routing

Hierarchical Routing

Our routing study thus far - idealization

- all routers identical
- network "flat"

... *not* true in practice

scale: with 200 million destinations:

- can't store all dest's in routing tables!
- routing table exchange would swamp links!

administrative autonomy

- internet = network of networks
- each network admin may want to control routing in its own network

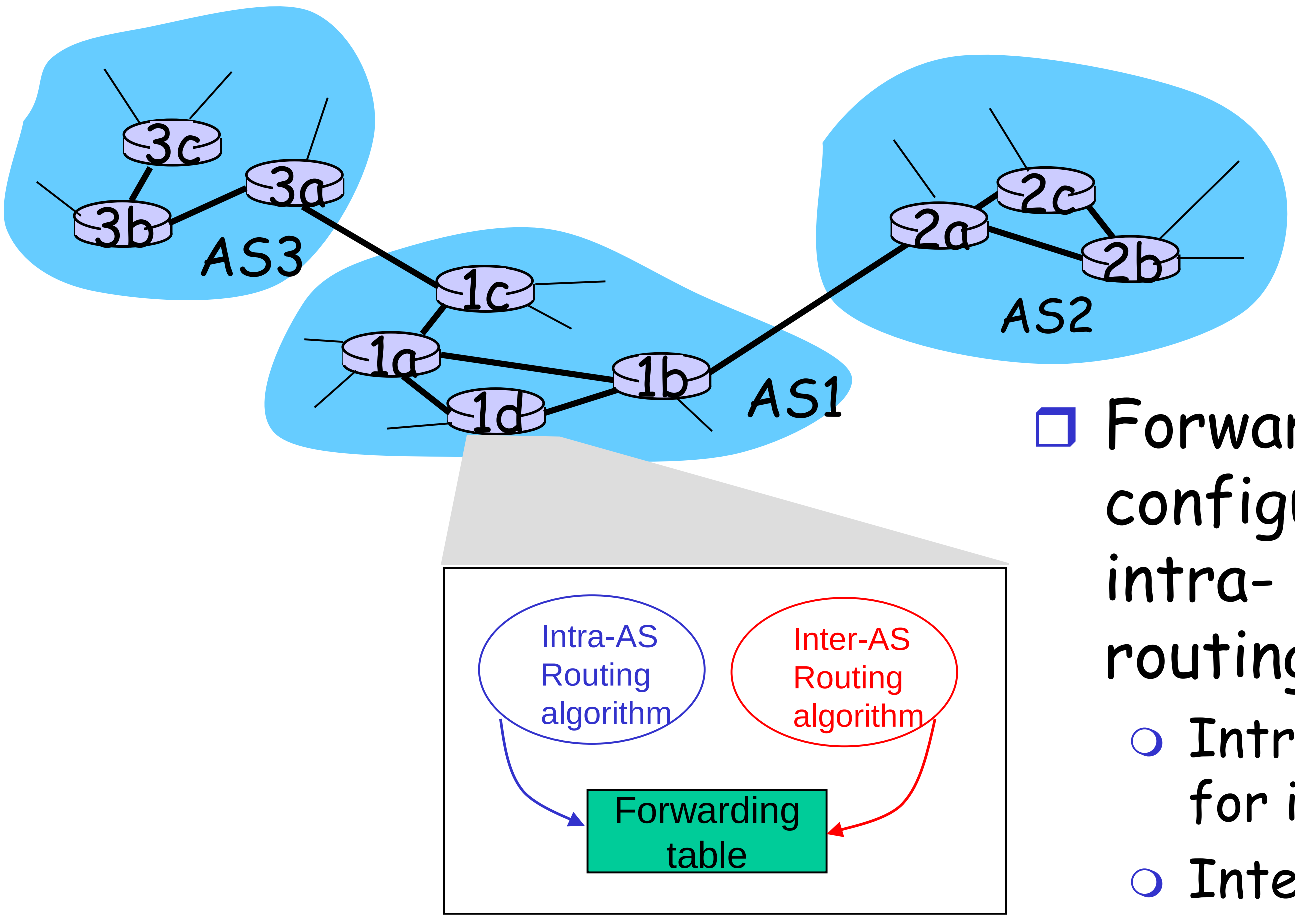
Hierarchical Routing

- aggregate routers into regions, "autonomous systems" (AS)
- routers in same AS run same routing protocol
  - "intra-AS" routing protocol
  - routers in different AS can run different intra-AS routing protocol

Gateway router

- Direct link to router in another AS

Interconnected ASes



- Forwarding table is configured by both intra- and inter-AS routing algorithm
  - Intra-AS sets entries for internal dests
  - Inter-AS & Intra-As sets entries for external dests

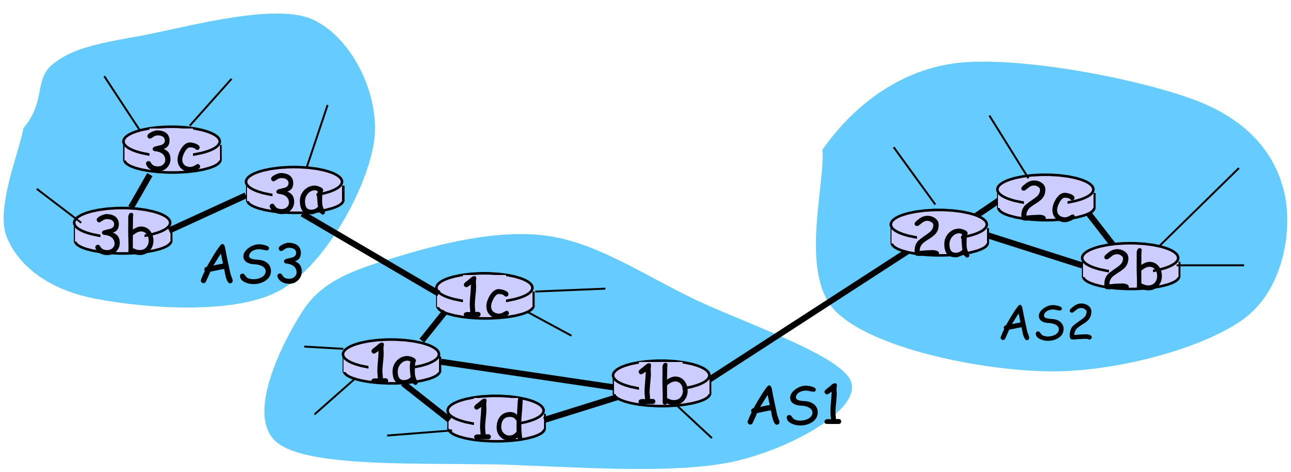
Inter-AS tasks

- Suppose router in AS1 receives datagram for which dest is outside of AS1
  - Router should forward packet towards one of the gateway routers, but which one?

AS1 needs:

1. to learn which dests are reachable through AS2 and which through AS3
2. to propagate this reachability info to all routers in AS1

Job of inter-AS routing!

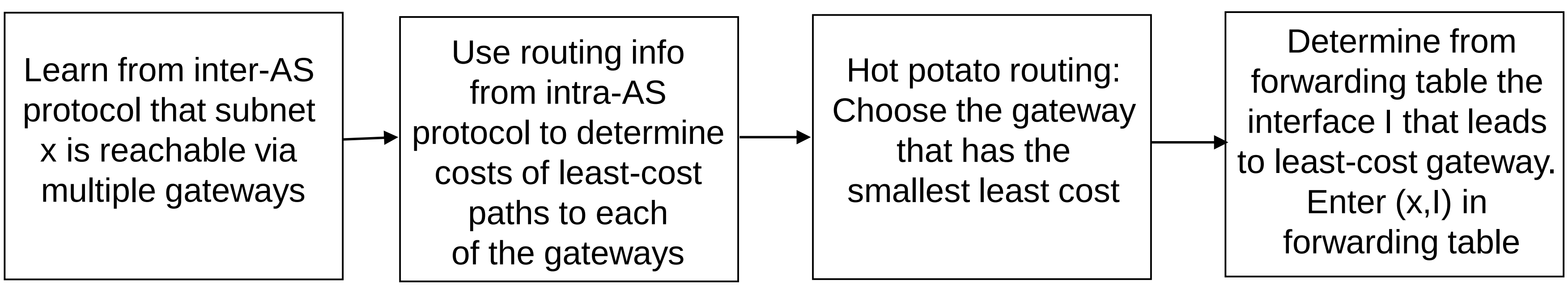


Example: Setting forwarding table in router 1d

- Suppose AS1 learns from the inter-AS protocol that subnet *x* is reachable from AS3 (gateway 1c) but not from AS2.
- Inter-AS protocol propagates reachability info to all internal routers.
- Router 1d determines from intra-AS routing info that its interface *I* is on the least cost path to 1c.
- Puts in forwarding table entry *(x,I)*.

Example: Choosing among multiple ASes

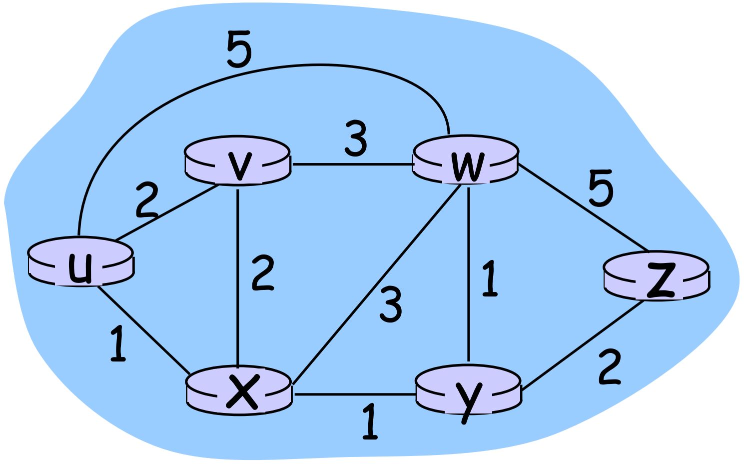
- Now suppose AS1 learns from the inter-AS protocol that subnet *x* is reachable from AS3 *and* from AS2.
- To configure forwarding table, router 1d must determine towards which gateway it should forward packets for dest x.
- This is also the job on inter-AS routing protocol!
- Hot potato routing: send packet towards closest of two routers.



Chapter 4: Network Layer

- Routing algorithms
  - Link state
  - Distance Vector

Graph abstraction

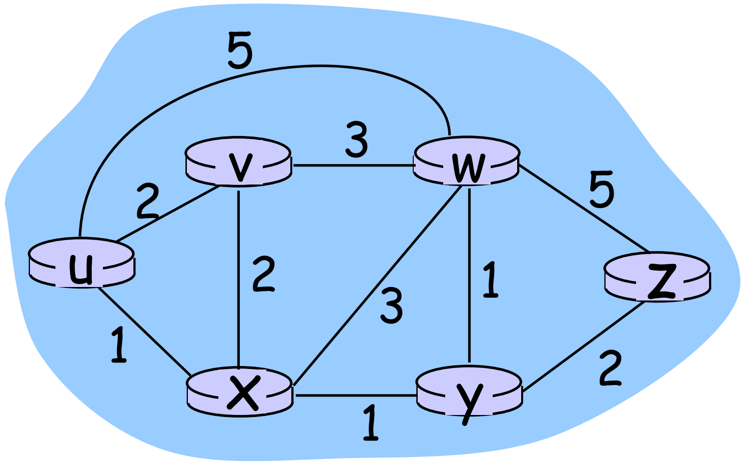


Graph: G = (N,E)

N = set of routers = { u, v, w, x, y, z }

E = set of links ={ (u,v), (u,x), (v,x), (v,w), (x,w), (x,y), (w,y), (w,z), (y,z) }

Graph abstraction: costs



- $c(x,x')$ = cost of link (x,x')

  - e.g., $c(w,z) = 5$

- cost could always be 1, or inversely related to bandwidth, or inversely related to congestion

Cost of path $(x_1, x_2, x_3,\ldots, x_p) = c(x_1,x_2) + c(x_2,x_3) + \ldots + c(x_{p-1},x_p)$

Question: What's the least-cost path between u and z ?

Routing algorithm: algorithm that finds least-cost path

Routing Algorithm classification

Global or decentralized information?

Global:
- all routers have complete topology, link cost info
- "link state" algorithms

Decentralized:
- router knows physically-connected neighbors, link costs to neighbors
- iterative process of computation, exchange of info with neighbors
- "distance vector" algorithms

Static or dynamic?

Static:
- routes change slowly over time

Dynamic:
- routes change more quickly
  - periodic update
  - in response to link cost changes

Chapter 4: Network Layer

- Routing algorithms
  - Link state
  - Distance Vector

A Link-State Routing Algorithm

Dijkstra's algorithm

- net topology, link costs known to all nodes
  - accomplished via "link state broadcast"
  - all nodes have same info
- computes least cost paths from one node ('source") to all other nodes
  - gives forwarding table for that node
- iterative: after k iterations, know least cost path to k dest.'s

Notation:

- c(x,y): link cost from node x to y; = ∞ if not direct neighbors
- D(v): current value of cost of path from source to dest. v
- p(v): predecessor node along path from source to v
- N': set of nodes whose least cost path definitively known

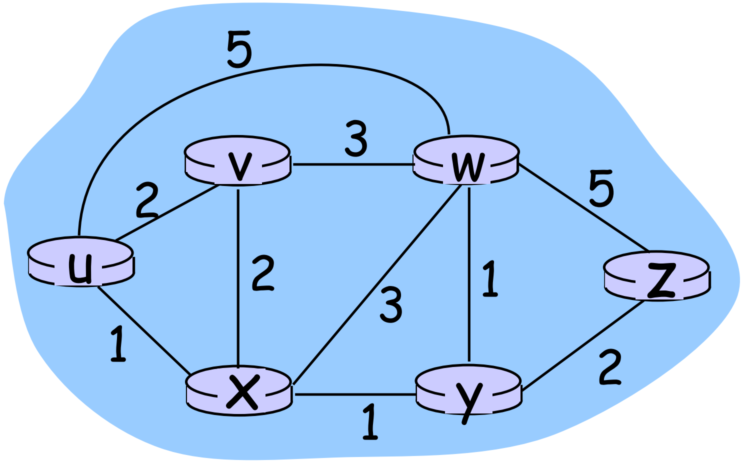
Dijkstra's Algorithm

```
Initialization:
  N' = {u}
  for all nodes v
    if v adjacent to u
        then D(v) = c(u,v)
    else D(v) = ∞

 Loop
   find w not in N' such that D(w) is a minimum
    add w to N'
    update D(v) for all v adjacent to w and not in N' :
        D(v) = min( D(v), D(w) + c(w,v) )
    /* new cost to v is either old cost to v or known
     shortest path cost to w plus cost from w to v */
 until all nodes in N'
```

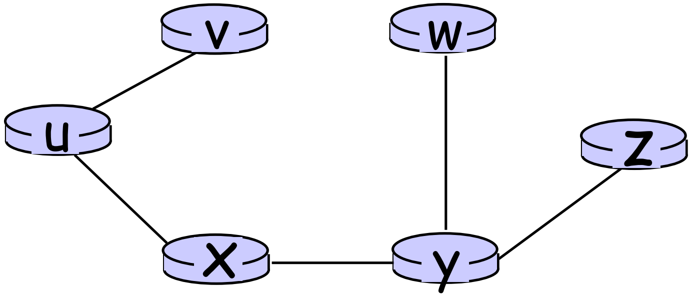
Dijkstra's algorithm: example

Step	N'	D(v),p(v)	D(w),p(w)	D(x),p(x)	D(y),p(y)	D(z),p(z)
0	u	2,u	5,u	1,u	∞	∞
1	ux	2,u	4,x		2,x	∞
2	uxy	2,u	3,y			4,y
3	uxyv		3,y			4,y
4	uxyvw					4,y
5	uxyvwz					



Dijkstra's algorithm: example (2)

Resulting shortest-path tree from u:



Resulting forwarding table in u:

destination	link
v	(u,v)
x	(u,x)
y	(u,x)
w	(u,x)
z	(u,x)

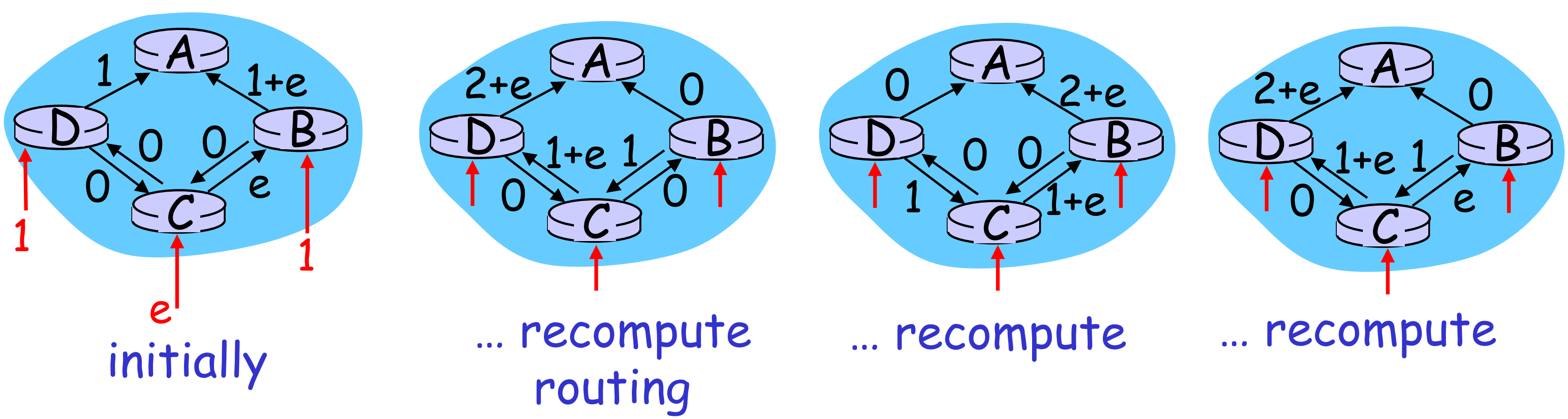
Dijkstra's algorithm, discussion

Algorithm complexity: n nodes

- each iteration: need to check all nodes, w, not in N
- n(n+1)/2 comparisons: $O(n^2)$
- more efficient implementations possible: O(nlogn)

Oscillations possible:

- e.g., link cost = amount of carried traffic



Chapter 4: Network Layer

- Routing algorithms
  - Link state
  - Distance Vector

Distance Vector Algorithm

Bellman-Ford Equation (dynamic programming)

Define

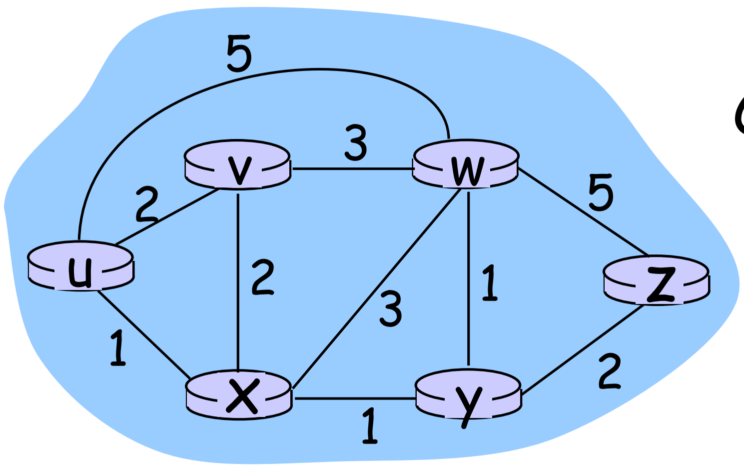
$d_x(y)$:= cost of least-cost path from x to y

Then

$$d_x(y) = \min_v \{c(x,v) + d_v(y)\}$$

where min is taken over all neighbors v of x

Bellman-Ford example



Clearly, $d_v(z) = 5$, $d_x(z) = 3$, $d_w(z) = 3$

B-F equation says:

$$d_u(z) = \min \{ c(u,v) + d_v(z),\ c(u,x) + d_x(z),\ c(u,w) + d_w(z) \}$$
$$= \min \{2 + 5,\ 1 + 3,\ 5 + 3\} = 4$$

Node that achieves minimum is next hop in shortest path ➜ forwarding table

Distance Vector Algorithm

- $D_x(y)$ = estimate of least cost from x to y
- Distance vector: $\mathbf{D}_x = [D_x(y): y \in N]$
- Node x knows cost to each neighbor v: $c(x,v)$
- Node x maintains $\mathbf{D}_x = [D_x(y): y \in N]$
- Node x also maintains its neighbors' distance vectors
  - For each neighbor v, x maintains $\mathbf{D}_v = [D_v(y): y \in N]$

Distance vector algorithm (4)

Basic idea:

- Each node periodically sends its own distance vector estimate to neighbors
- When a node x receives new DV estimate from neighbor, it updates its own DV using B-F equation:

$$D_x(y) \leftarrow \min_v\{c(x,v) + D_v(y)\} \quad \text{for each node } y \in N$$

- Under minor, natural conditions, the estimate $D_x(y)$ *converge to the actual least cost* $d_x(y)$

Distance Vector Algorithm (5)

Iterative, asynchronous:
each local iteration caused by:
- local link cost change
- DV update message from neighbor

Distributed:
- each node notifies neighbors *only* when its DV changes
  - neighbors then notify their neighbors if necessary

Each node:

wait for (change in local link cost or msg from neighbor)

↓

recompute estimates

↓

if DV to any dest has changed, *notify* neighbors

node x table | **node y table** | **node z table**

time ↓

i = 0

Dx	via y	via z
y	(2)	∞
z	∞	(7)

Dy	via x	via z
x	(2)	∞
z	∞	(1)

Dz	via x	via y
x	(7)	∞
y	∞	(1)

i = 1

Dx	via y	via z
y	(2)	8
z	(3)	7

Dy	via x	via z
x	(2)	8
z	9	(1)

Dz	via x	via y
x	7	(3)
y	9	(1)

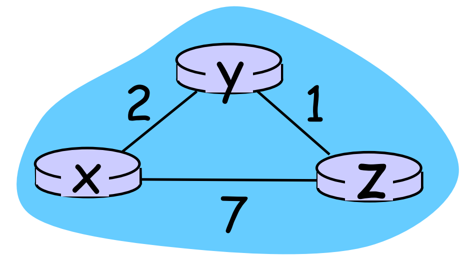
i = 2

Dx	via y	via z
y	(2)	8
z	(3)	7

Dy	via x	via z
x	(2)	4
z	5	(1)

Dz	via x	via y
x	7	(3)
y	9	(1)

$D_z(x)_{via\ y} = c(z,y) + D_y(x) = 1 + 2 = 3$



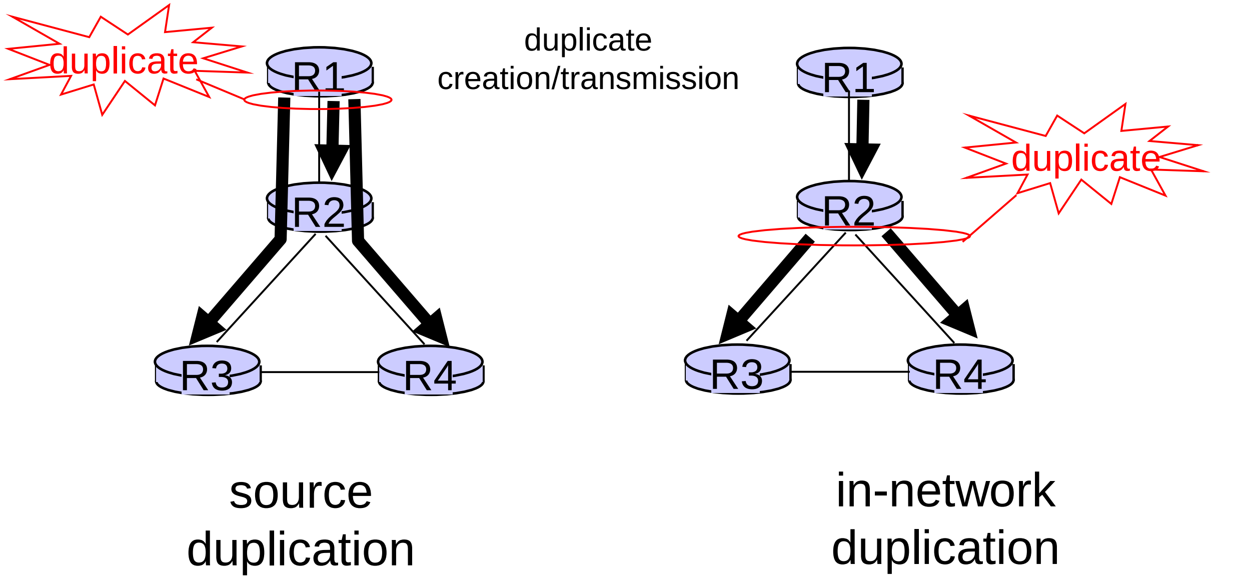
$D_y(z)_{via\ x} = c(y,x) + D_x(z) = 2 + 3 = 5$

Chapter 4: Network Layer

- 4.7 Multicast routing

Broadcast routing

- deliver packets from source to all other nodes
- source duplication is inefficient:



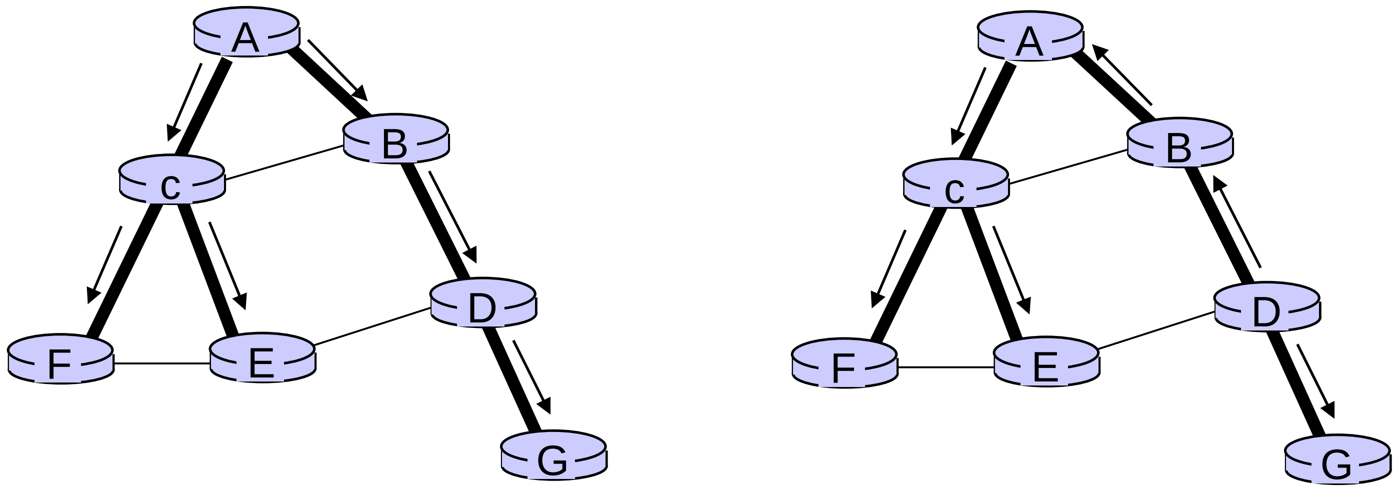
- source duplication: how does source determine recipient addresses?

In-network duplication

- *flooding:* when node receives broadcast packet, sends copy to all neighbors
  - problems: cycles & broadcast storm
- *controlled flooding:* node only broadcasts pkt if it hasn't broadcast same packet before
  - node keeps track of packet ids already broadacsted
  - or reverse path forwarding (RPF): only forward packet if it arrived on shortest path between node and source
- *spanning tree:*
  - no redundant packets received by any node

Spanning tree

- first construct a spanning tree
- nodes then forward/make copies only along spanning tree

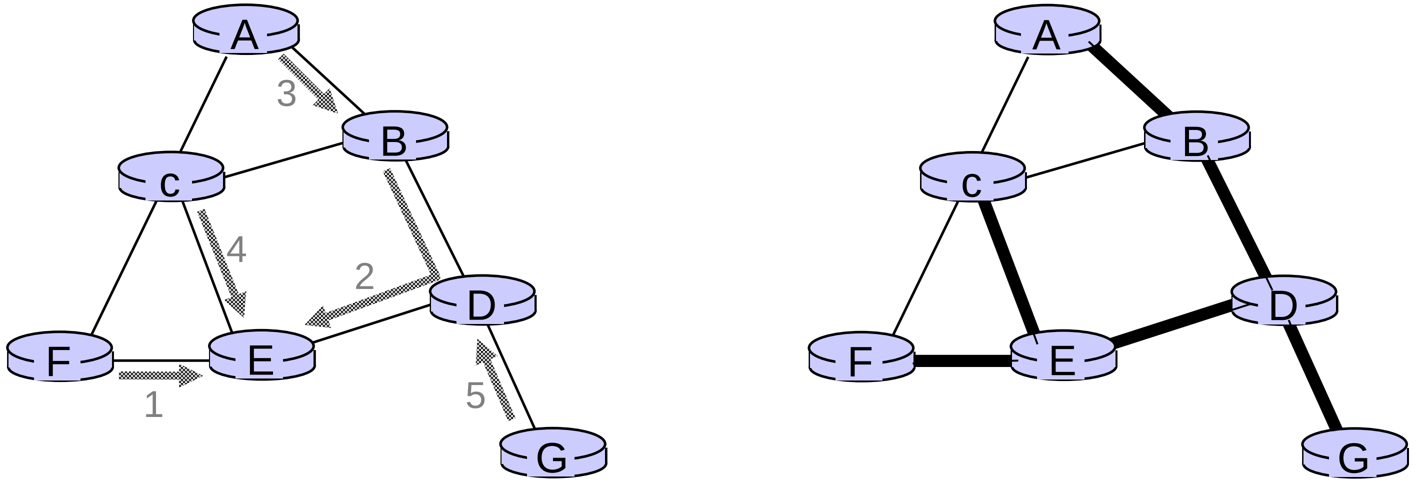


(a) broadcast initiated at A

(b) broadcast initiated at D

Spanning tree: creation

- center node
- each node sends unicast join message to center node
  - message forwarded until it arrives at a node already belonging to spanning tree



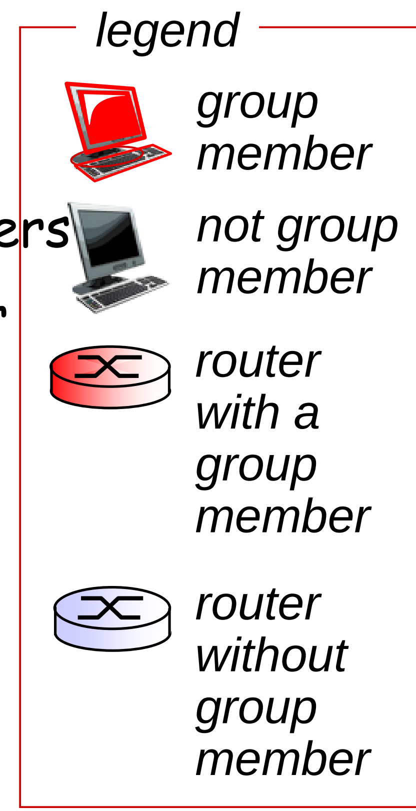
(a) stepwise construction of spanning tree (center: E)

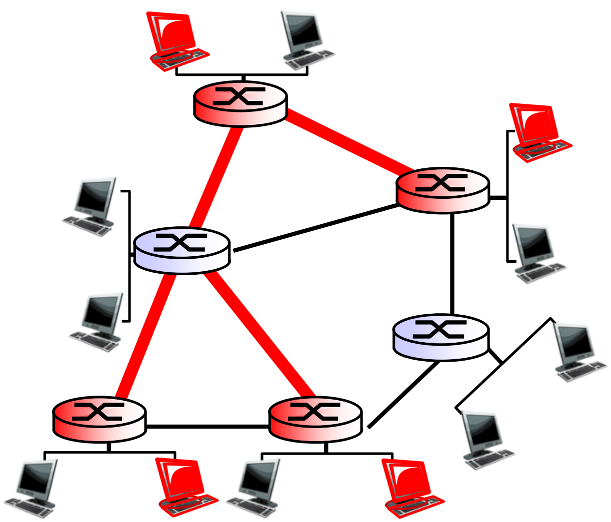
(b) constructed spanning tree

Multicast routing: problem statement

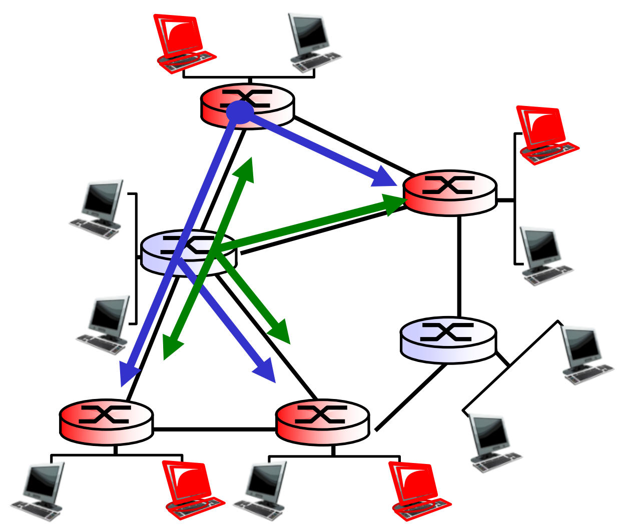
goal: find a tree (or trees) connecting routers having local mcast group members

- *tree:* not all paths between routers used
- *shared-tree:* same tree used by all group members
- *source-based:* different tree from each sender



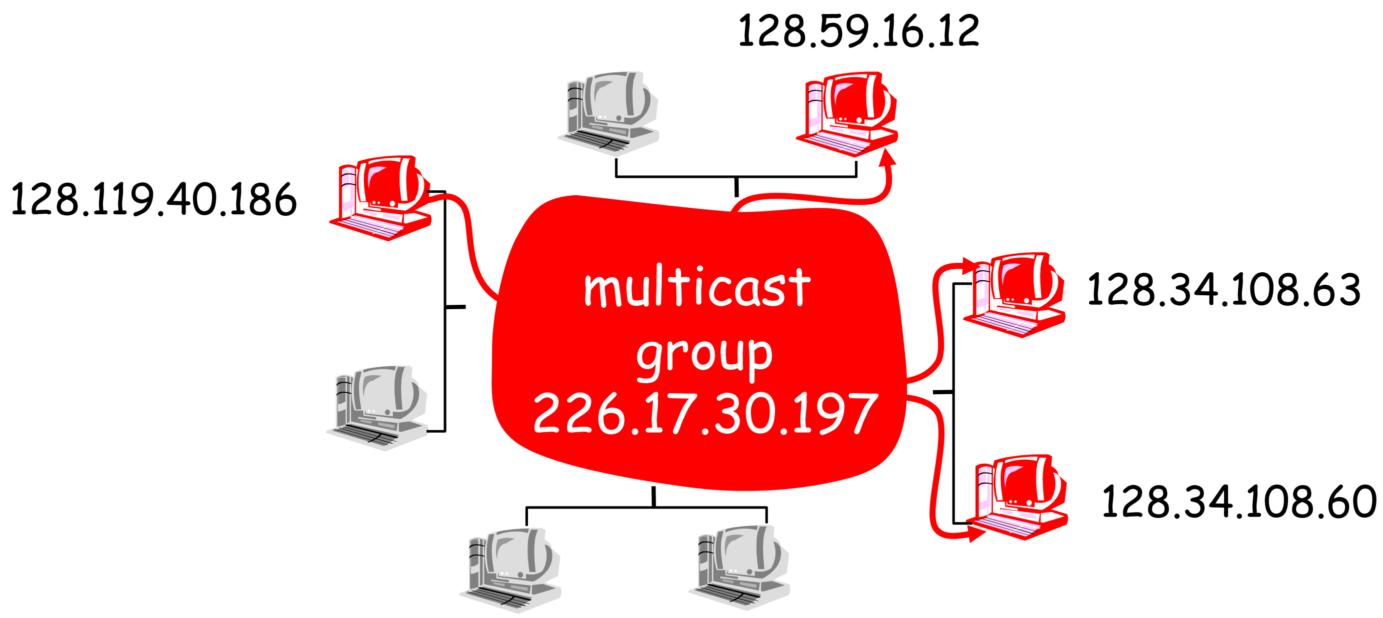


shared tree



source-based trees

Internet Multicast Service Model



multicast group concept: use of indirection

- hosts addresses IP datagram to multicast group
- routers forward multicast datagrams to hosts that have "joined" that multicast group

Multicast groups

- class D Internet addresses reserved for multicast:

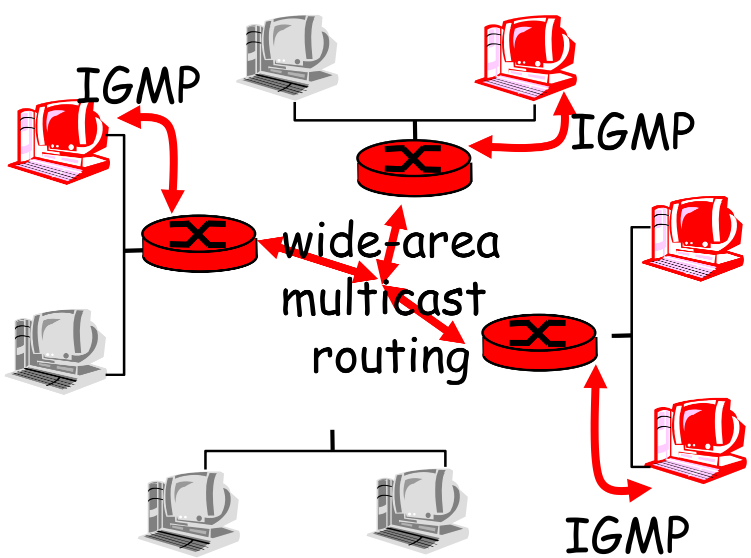
1110	Multicast Group ID

← 28 bits →

- host group semantics:
  - anyone can "join" (receive) multicast group
  - anyone can send to multicast group
  - no network-layer identification to hosts of members
- *needed:* infrastructure to deliver mcast-addressed datagrams to all hosts that have joined that multicast group

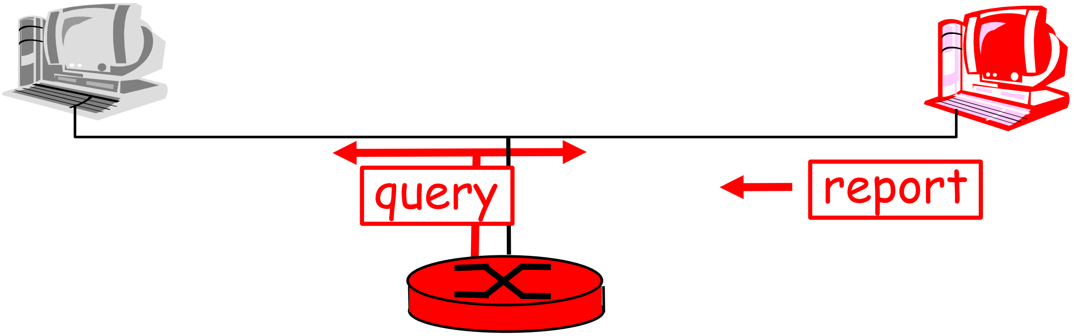
Joining a mcast group: two-step process

- *local:* host informs local mcast router of desire to join group: IGMP (Internet Group Management Protocol)
- *wide area:* local router interacts with other routers to receive mcast datagram flow
  - many protocols (e.g., DVMRP, MOSPF, PIM)



IGMP: Internet Group Management Protocol

- *host:* sends IGMP report when application joins mcast group
  - IP_ADD_MEMBERSHIP socket option
  - host need not explicitly "unjoin" group when leaving
- *router:* sends IGMP query at regular intervals
  - host belonging to a mcast group must reply to query



IGMP

IGMP version 1

- *router:* Host Membership Query msg broadcast on LAN to all hosts
- *host:* Host Membership Report msg to indicate group membership
  - randomized delay before responding
  - implicit leave via no reply to Query
- RFC 1112

IGMP v2: additions include

- group-specific Query
- Leave Group msg
  - last host replying to Query can send explicit Leave Group msg
  - router performs group-specific query to see if any hosts left in group
  - RFC 2236

IGMP v3: under development as Internet draft