

chapter 4

Networking Protocols and Architecture

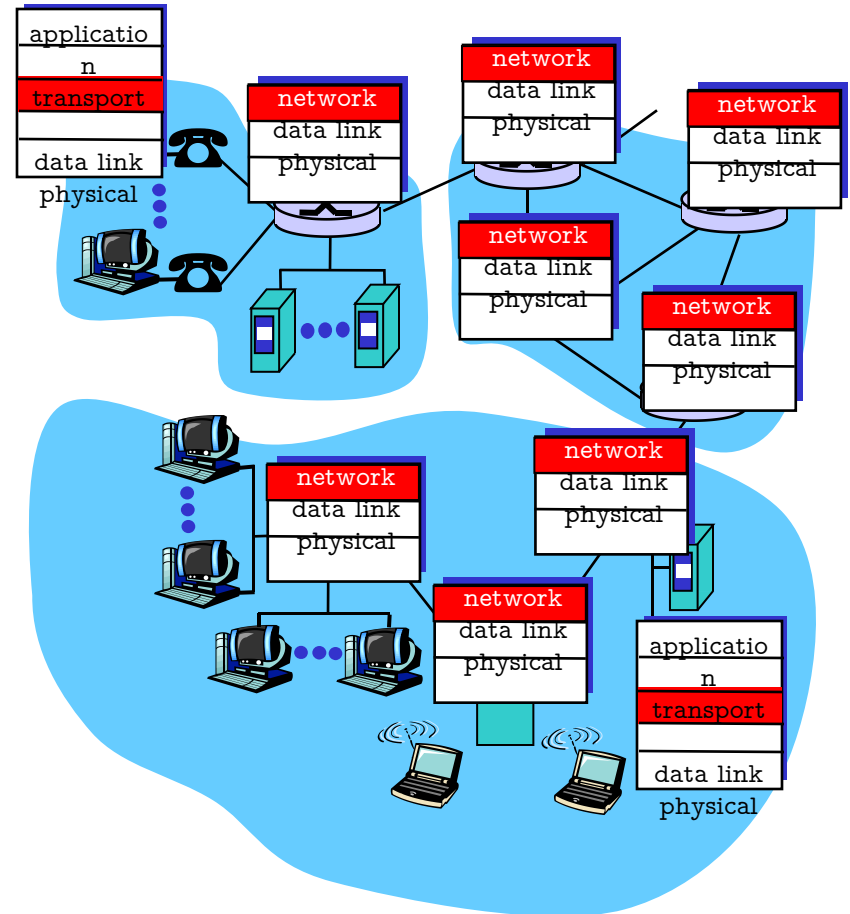
Tameru H.
AASTU, Depart of Software Engineering

Network layer functions

- transport packet from sending to receiving hosts
- network layer protocols in every host, router

Three important functions:

- *path determination*: route taken by packets from source to dest. *Routing algorithms*
- *forwarding*: move packets from router's input to appropriate router output
- *call setup*: some network architectures require router call setup along path before data flows



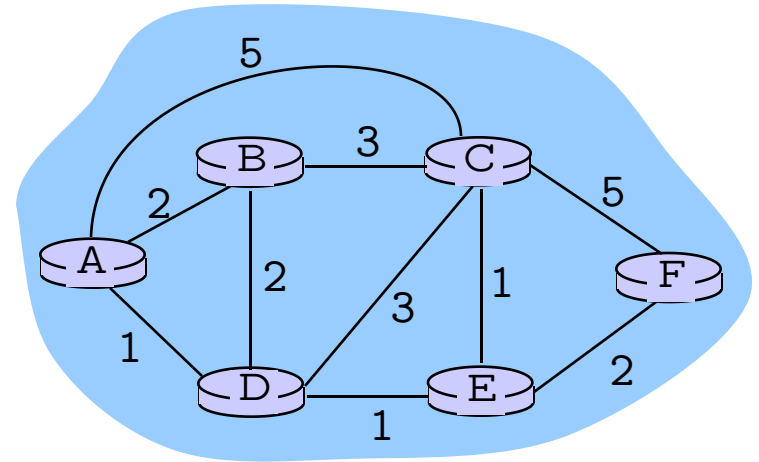
Routing

Routing protocol

Goal: determine “good” path
(sequence of routers) thru
network from source to dest.

Graph abstraction for
routing algorithms:

- graph nodes are routers
- graph edges are physical links
 - link cost: delay, \$ cost, or congestion level



- “good” path:
typically means
minimum 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

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 routing table for that node
- iterative: after k iterations, know least cost path to k dest.'s

Notation:

- $c(i,j)$: link cost from node i to j . cost infinite 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 , that is next v
- N : set of nodes whose least cost path definitively known

Dijkstra's Algorithm

1 **Initialization:**

2 $N = \{A\}$

3 for all nodes v

4 if v adjacent to A

5 then $D(v) = c(A, v)$

6 else $D(v) = \text{infinity}$

7

8 **Loop**

9 find w not in N such that $D(w)$ is a minimum

10 add w to N

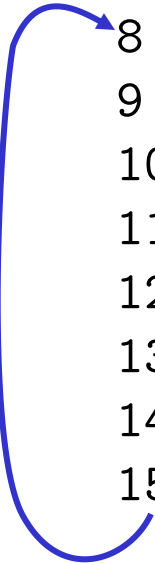
11 update $D(v)$ for all v adjacent to w and not in N :

12 $D(v) = \min(D(v), D(w) + c(w, v))$

13 /* new cost to v is either old cost to v or known

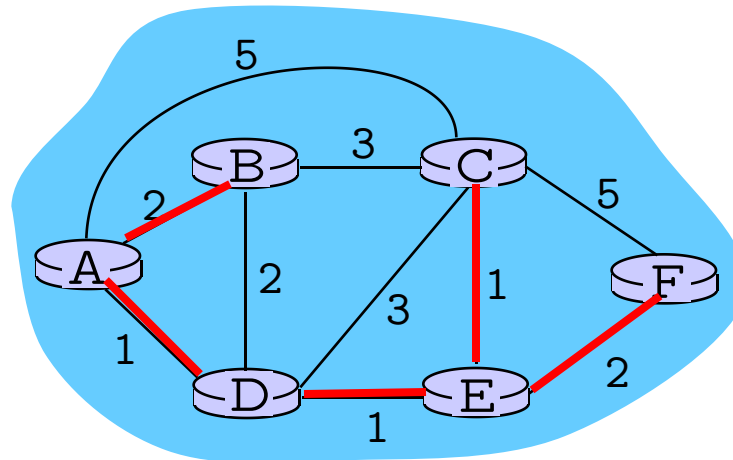
14 shortest path cost to w plus cost from w to v */

15 **until all nodes in N**



Dijkstra's algorithm: example

Step	start N	D(B),p(B)	D(C),p(C)	D(D),p(D)	D(E),p(E)	D(F),p(F)
→ 0	A	2,A	5,A	1,A	infinity	infinity
→ 1	AD	2,A	4,D		2,D	infinity
→ 2	ADE	2,A		3,E		4,E
→ 3	ADEB			3,E		4,E
→ 4	ADEBC					4,E
5	ADEBCF					



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: $O(n \log n)$

Distance Vector Routing Algorithm

RIP

iterative:

- continues until no nodes exchange info.
- *self-terminating*: no “signal” to stop

asynchronous:

- nodes need *not* exchange info/iterate in lock step!

distributed:

- each node communicates *only* with directly-attached neighbors

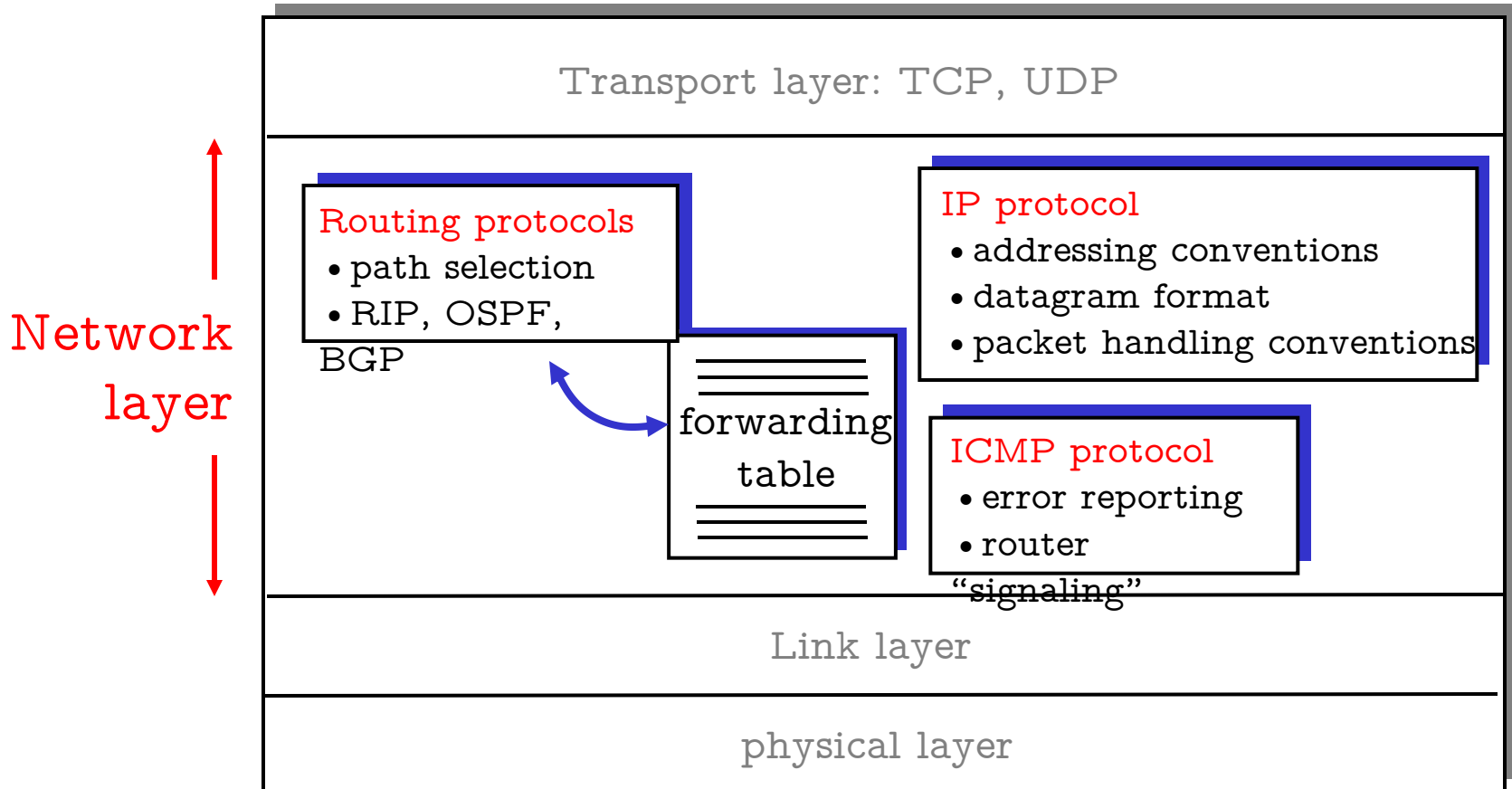
Distance Table data structure

- each node has its own row for each possible destination
- column for each directly-attached neighbor to node
- example: in node X, for dest. Y via neighbor Z:

$$\begin{aligned} D^X(Y, Z) &= \text{distance from X to Y, via Z as next hop} \\ &= c(X, Z) + \min_w \{D^Z(Y, w)\} \end{aligned}$$

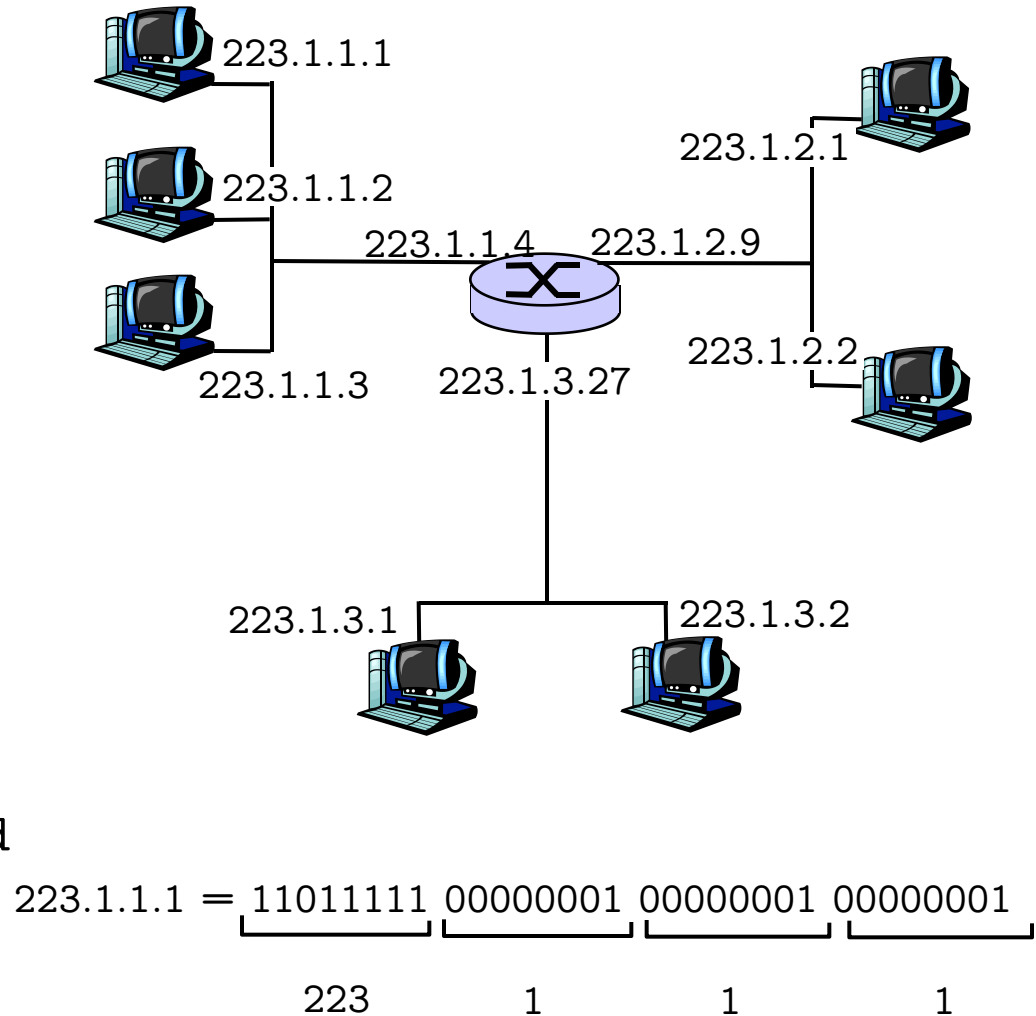
The Internet Network layer

Host, router network layer functions:



IP Addressing: Introduction

- IPv4 address: 32-bit identifier for host, router *interface*
- *interface*: connection between host/router and physical link
 - router's typically have multiple interfaces
 - host may have multiple interfaces
 - IP addresses associated with each interface



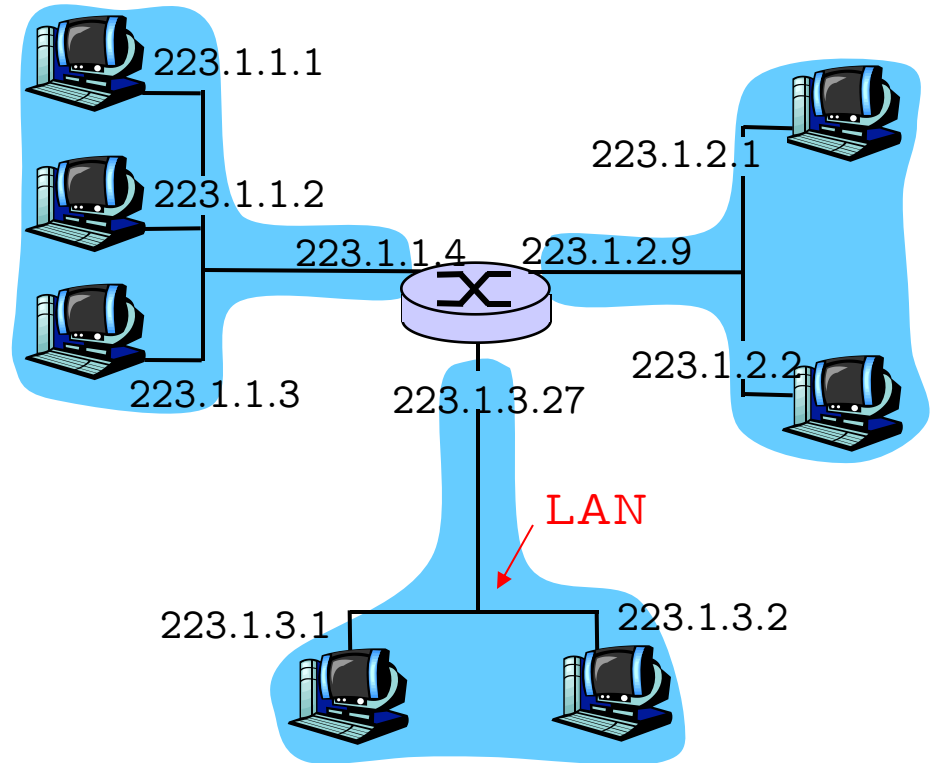
IP Addressing

- IP address:
 - network part (high order bits)
 - host part (low order bits)

- *What's a network ?*

(from IP address perspective)

- device interfaces with same network part of IP address
- can physically reach each other without intervening router



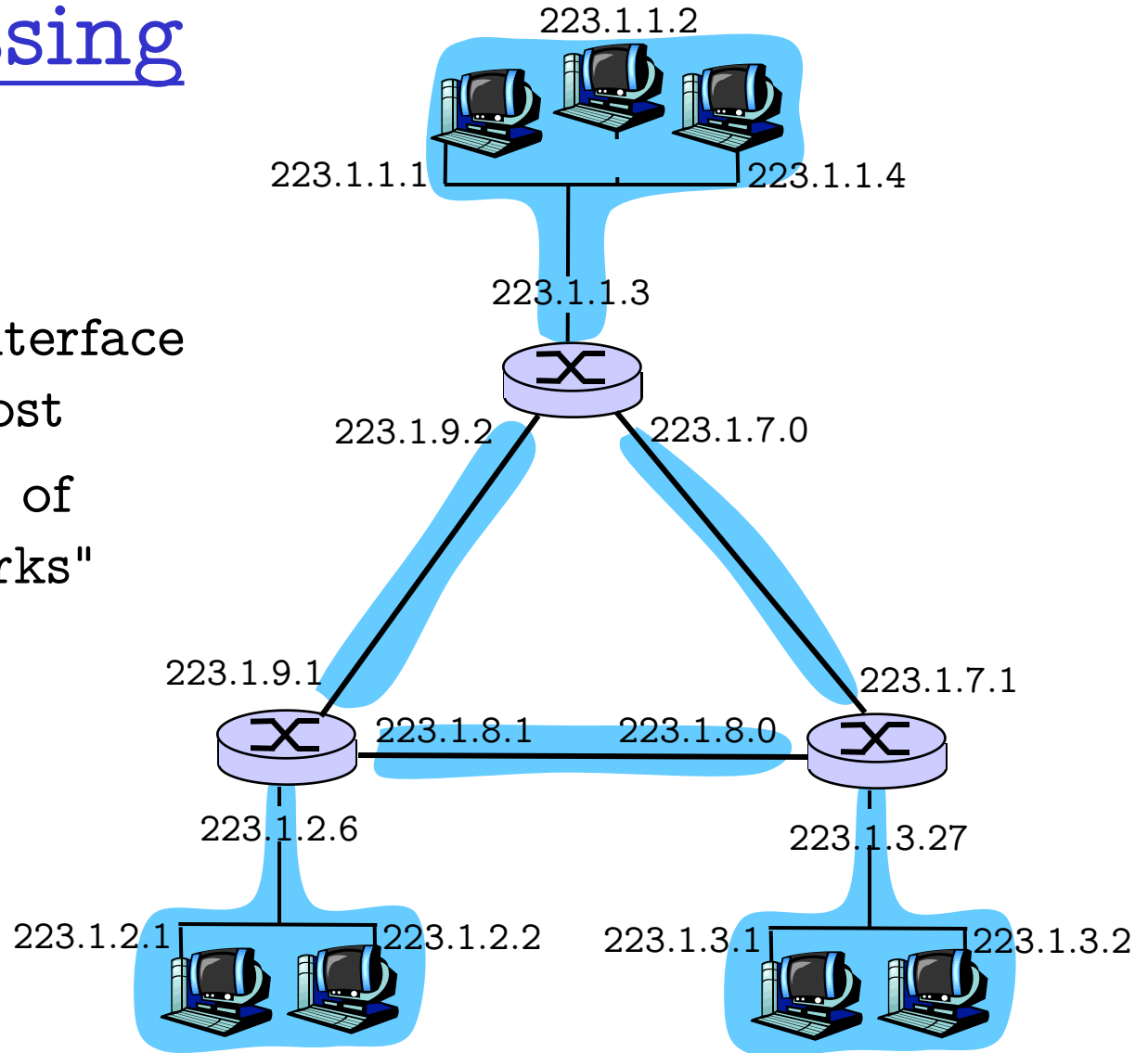
network consisting of 3 IP networks
(for IP addresses starting with 223,
first **24 bits** are network address)

IP Addressing

How to find the
networks?

- Detach each interface from router, host
- create “islands of isolated networks”

Interconnected
system consisting
of six networks



IP Addresses

given notion of “network”, let’s re-examine IP addresses:

“class-full” addressing:

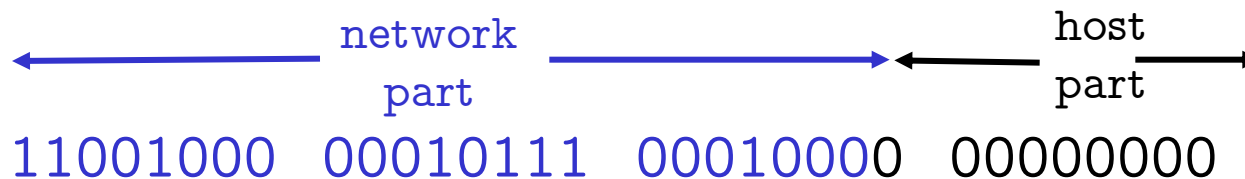
class

A	0	network		host		1.0.0.0 to 127.255.255.255	
B	10		network		host	128.0.0.0 to 191.255.255.255	
C	110		network			host	192.0.0.0 to 223.255.255.255
D	1110			multicast address			224.0.0.0 to 239.255.255.255

← 32 bits →

IP addressing: CIDR

- Classful addressing:
 - inefficient use of address space, address space exhaustion
 - e.g., class B net allocated enough addresses for 65K hosts, even if only 2K hosts in that network
- **CIDR: Classless InterDomain Routing**
 - network portion of address of arbitrary length
 - address format: **a.b.c.d/x**, where x is # bits in network 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 shortly)

IP addresses: how to get one?

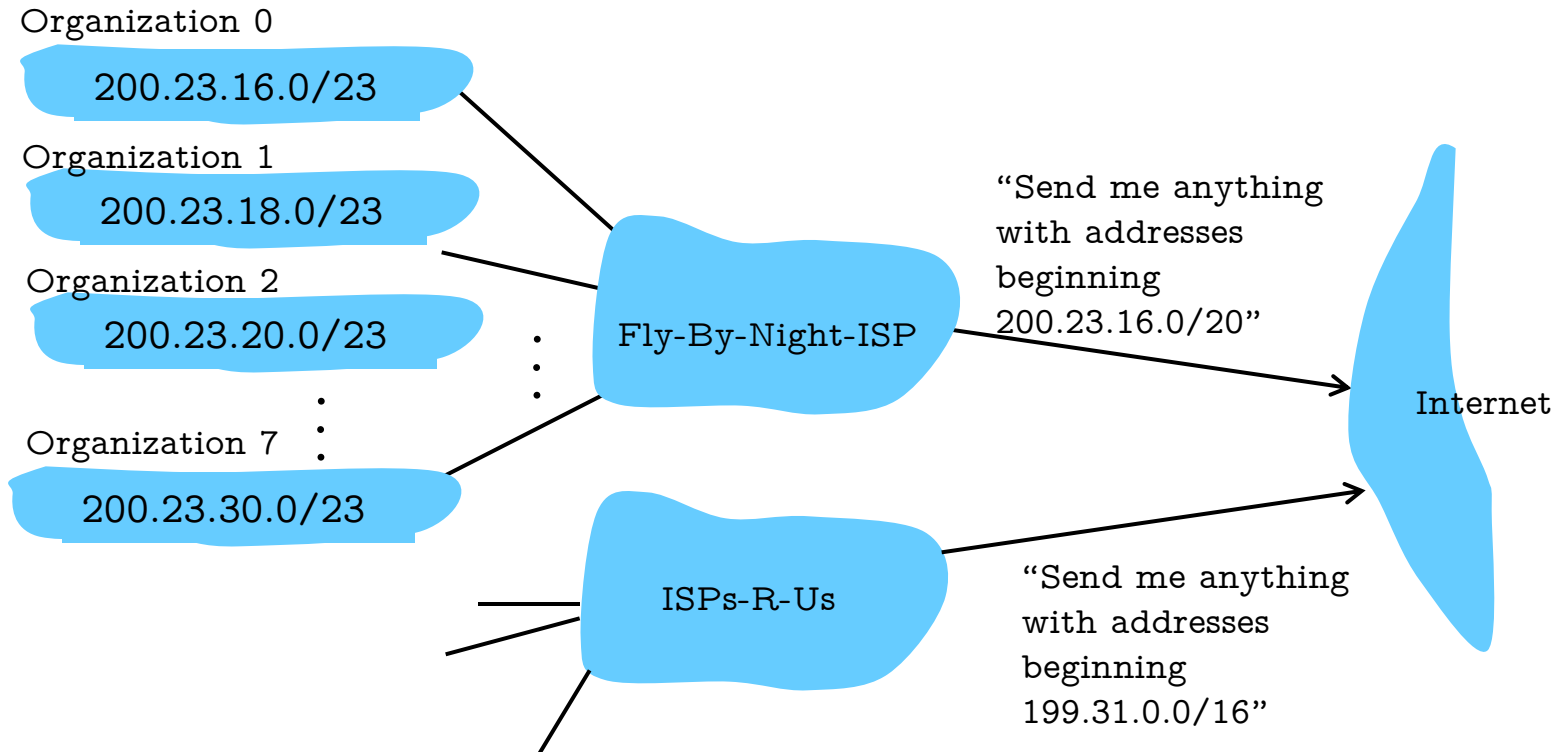
Q: How does *network* get network part of IP addr?

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

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

Hierarchical addressing: route aggregation

Hierarchical addressing allows efficient advertisement of routing information:



IP addressing: the last word...

Q: How does an ISP get block of addresses?

A: **ICANN**: Internet **C**orporation for **A**ssigned
Names and **N**umbers

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

Getting a datagram from source to dest.

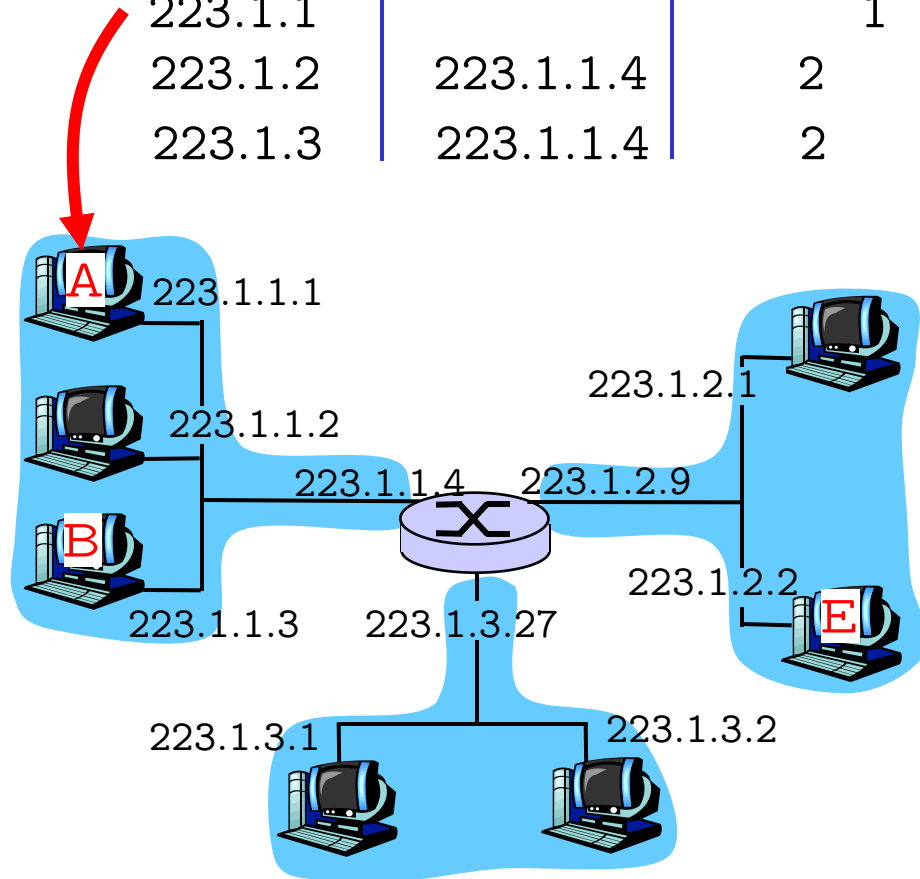
IP datagram:

misc	source	dest	
fields	IP addr	IP addr	data

- datagram remains **unchanged**, as it travels source to destination
- addr fields of interest here

forwarding table in A

Dest. Net.	next router	Nhops
223.1.1		1
223.1.2	223.1.1.4	2
223.1.3	223.1.1.4	2



Getting a datagram from source to dest.

misc fields	223.1.1.1	223.1.1.3	data
-------------	-----------	-----------	------

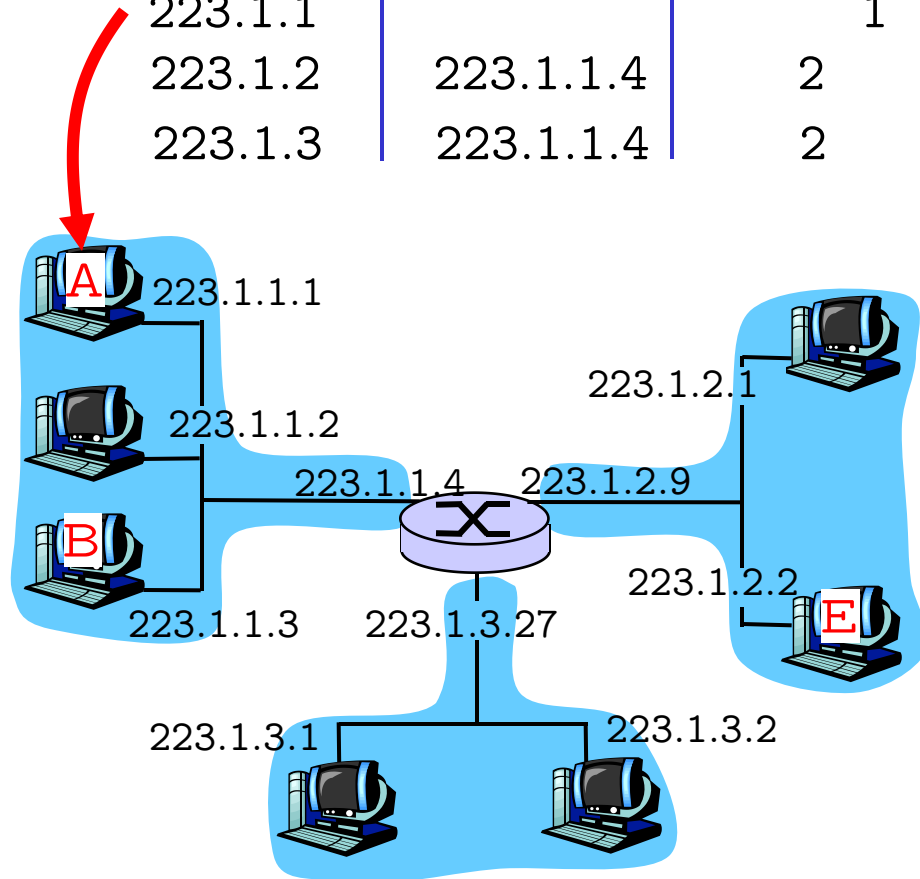
Starting at A, send IP datagram addressed to B:

- look up net. address of B in forwarding table
- find B is on same net. as A
- link layer will send datagram directly to B inside link-layer frame

B and A are directly connected

forwarding table in A

Dest. Net.	next router	Nhops
223.1.1		1
223.1.2	223.1.1.4	2
223.1.3	223.1.1.4	2



Getting a datagram from source to dest.

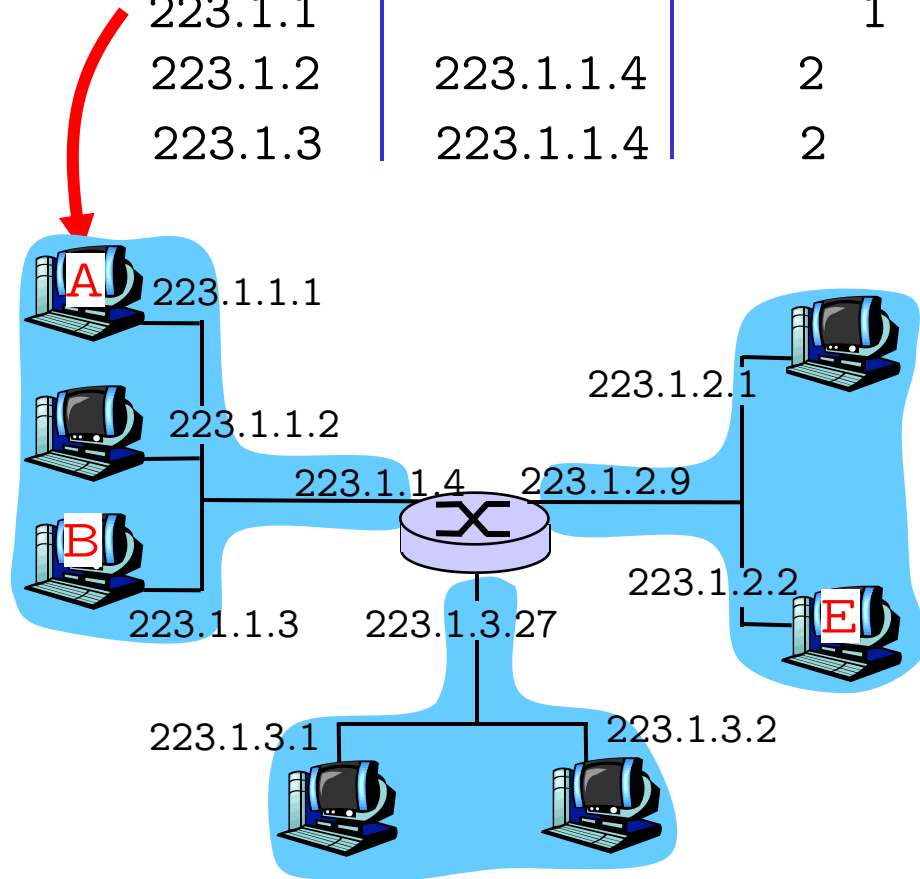
misc fields	223.1.1.1	223.1.2.3	data
-------------	-----------	-----------	------

Starting at A, dest. E:

- look up network address of E in forwarding table
- E on *different* network
A, E not directly attached
- routing table: next hop router to E is 223.1.1.4
- link layer sends datagram to router 223.1.1.4 inside link-layer frame
- datagram arrives at 223.1.1.4
- continued.....

forwarding table in A

Dest. Net.	next router	Nhops
223.1.1		1
223.1.2	223.1.1.4	2
223.1.3	223.1.1.4	2



Getting a datagram from source to dest.

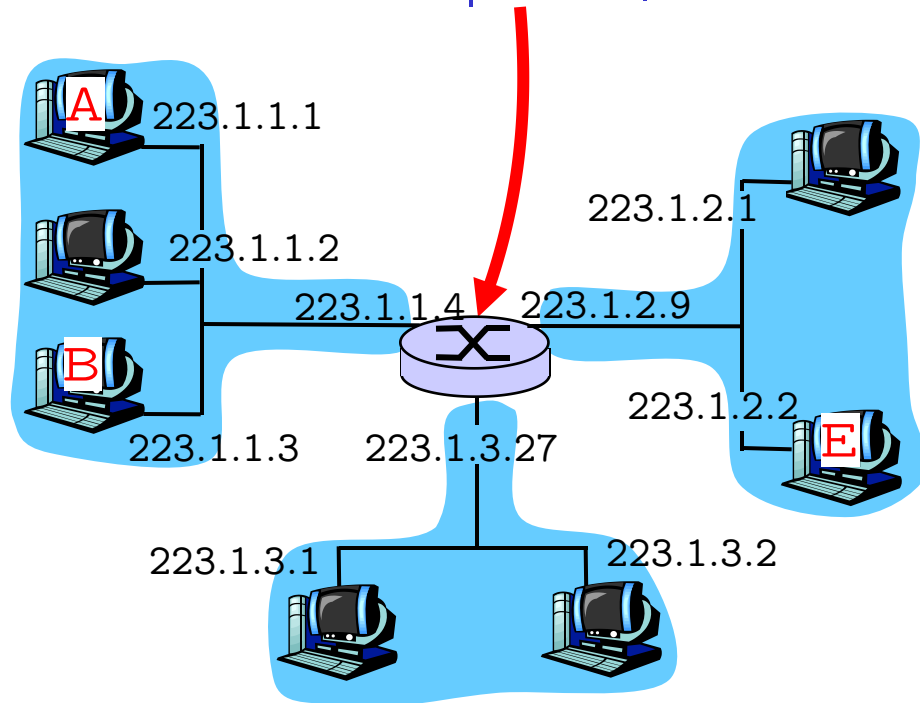
misc fields	223.1.1.1	223.1.2.3	data
-------------	-----------	-----------	------

Arriving at 223.1.4,
destined for 223.1.2.2

- look up network address of E in router's forwarding table
- E on *same* network as router's interface 223.1.2.9 router, E directly attached
- link layer sends datagram to 223.1.2.2 inside link-layer frame via interface 223.1.2.9
- datagram arrives at 223.1.2.2!!!

forwarding table in router

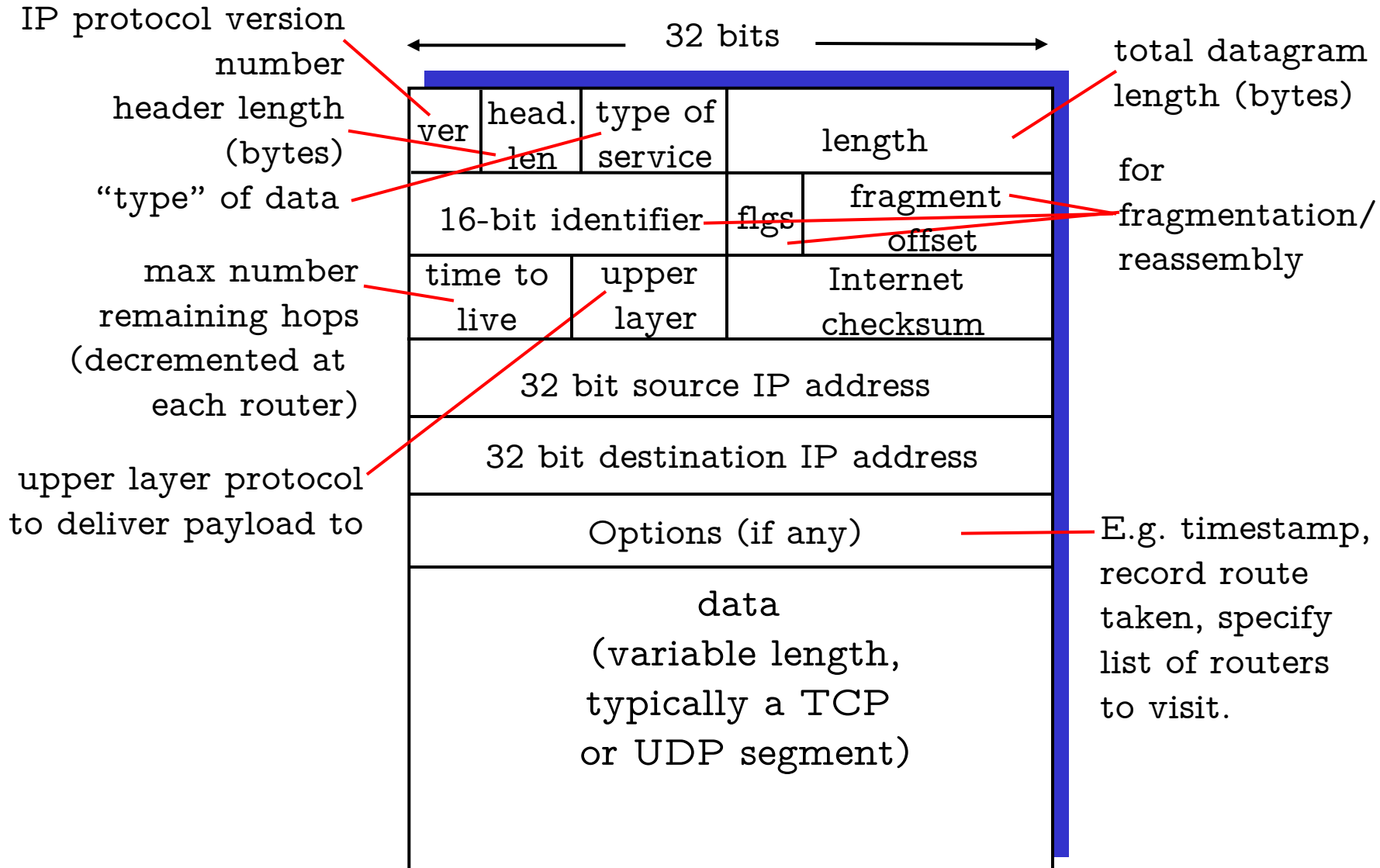
Dest. Net	router	Nhops	interface
223.1.1	-	1	223.1.1.4
223.1.2	-	1	223.1.2.9
223.1.3	-	1	223.1.3.27



(hooray!)

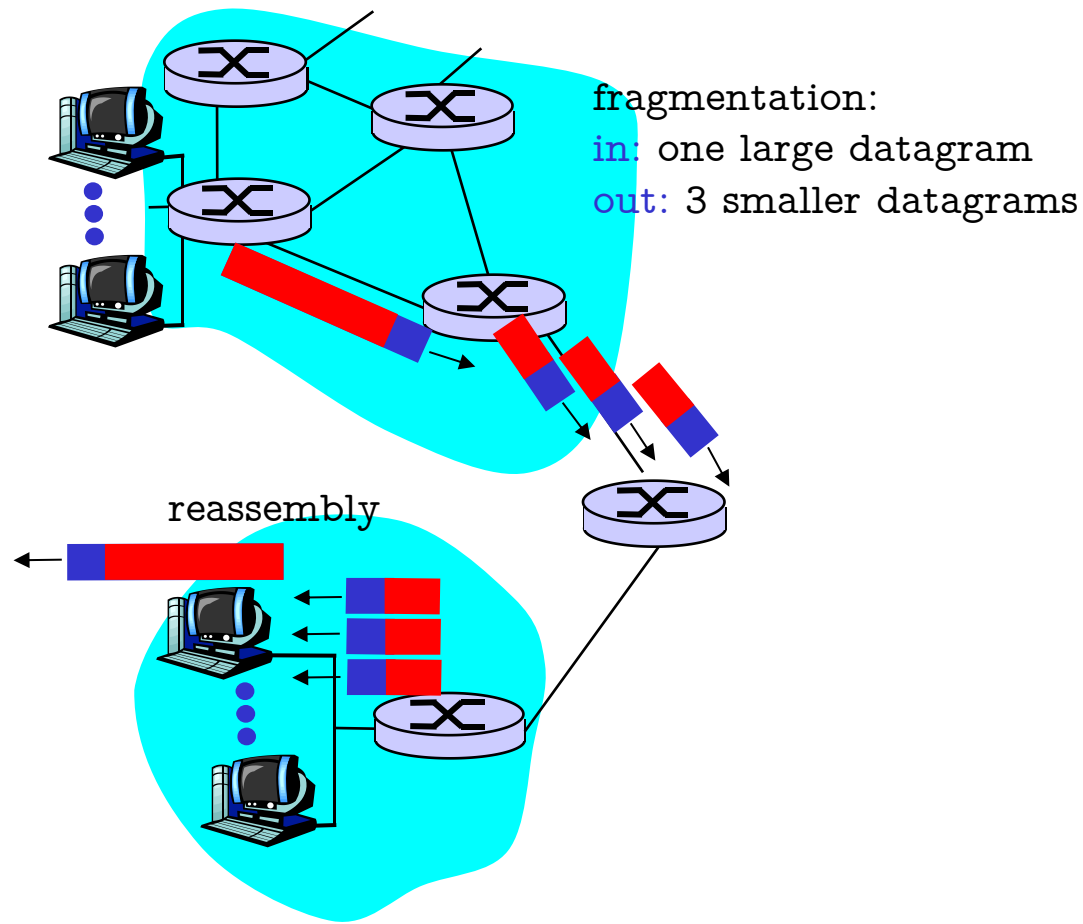
Russian term for yeahhh!!!!

IP datagram format



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



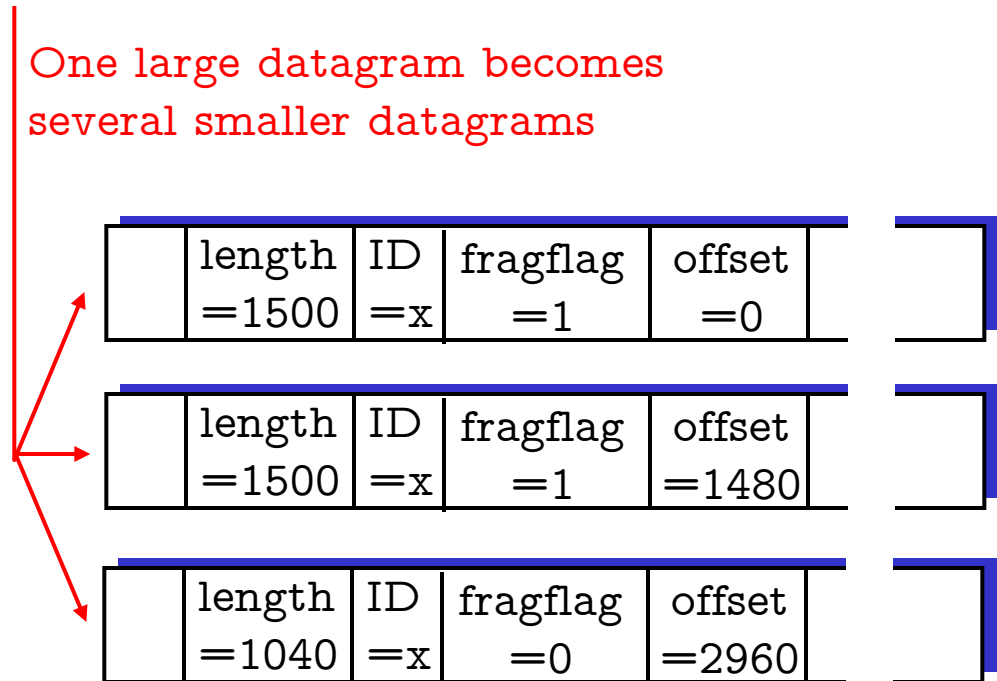
IP Fragmentation and Reassembly

Example

- 4000 byte datagram
- MTU = 1500 bytes

	length	ID	fragflag	offset	
	=4000	=x	=0	=0	

One large datagram becomes
several smaller datagrams



ICMP: Internet Control Message Protocol

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

<u>Type</u>	<u>Code</u>	<u>description</u>
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

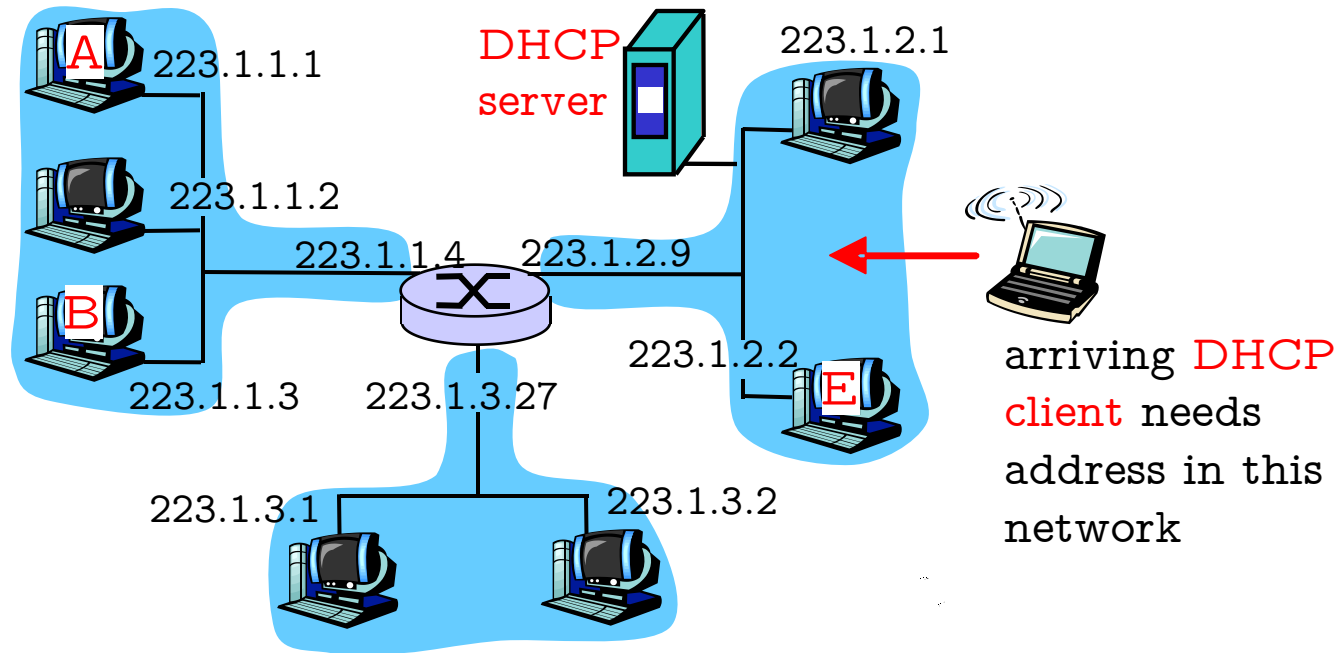
DHCP: Dynamic Host Configuration Protocol

Goal: allow host to *dynamically* obtain its 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 an “on”

Support for mobile users who want to join network
(more shortly)

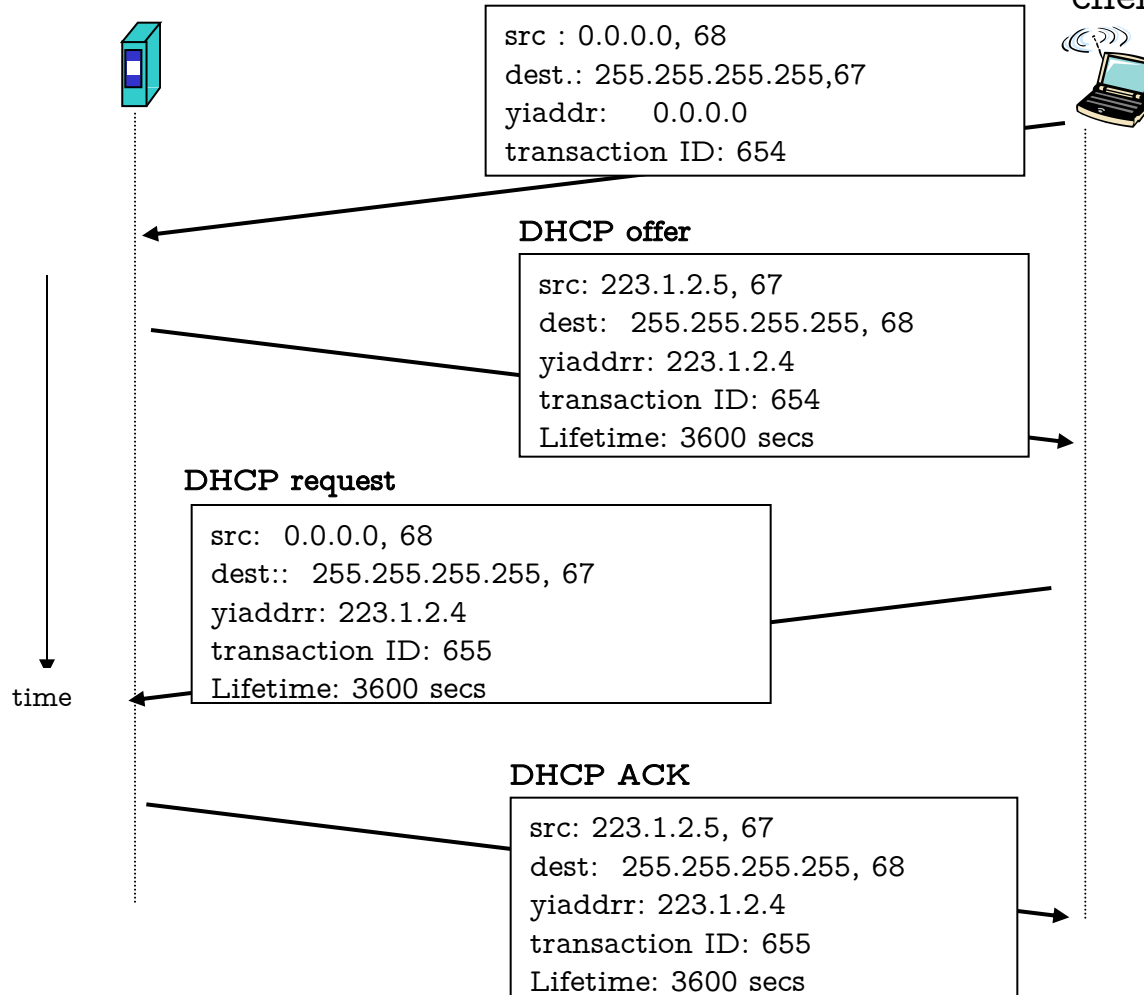
DHCP client-server scenario



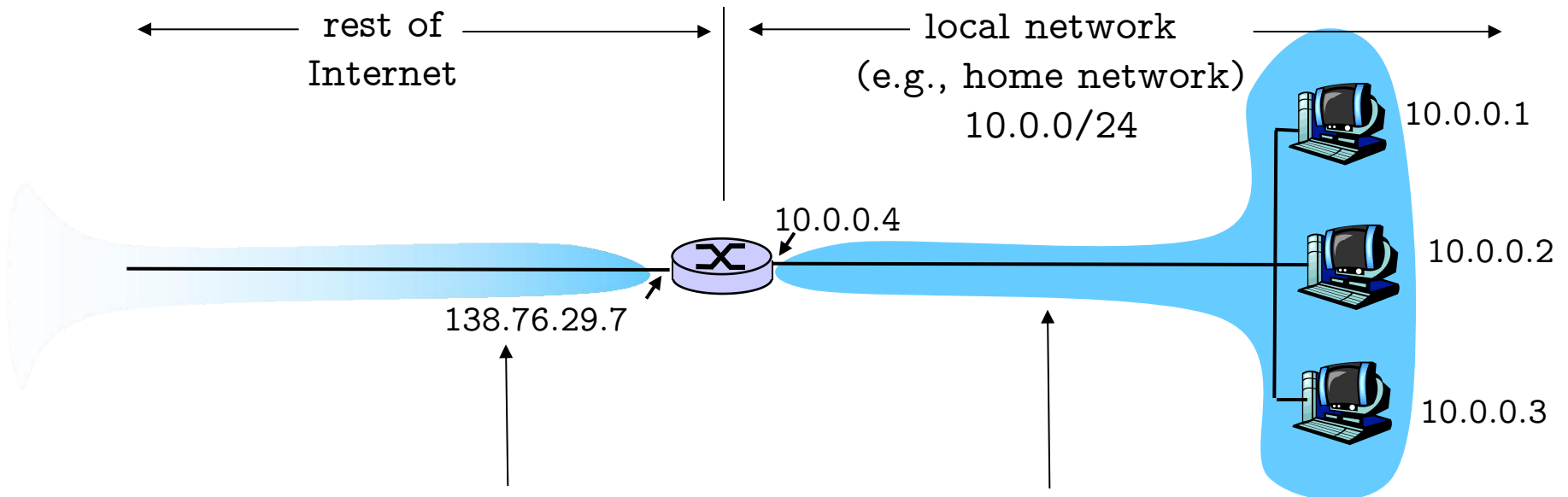
DHCP client-server scenario

DHCP server: 223.1.2.5

arriving
client



NAT: Network Address Translation



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

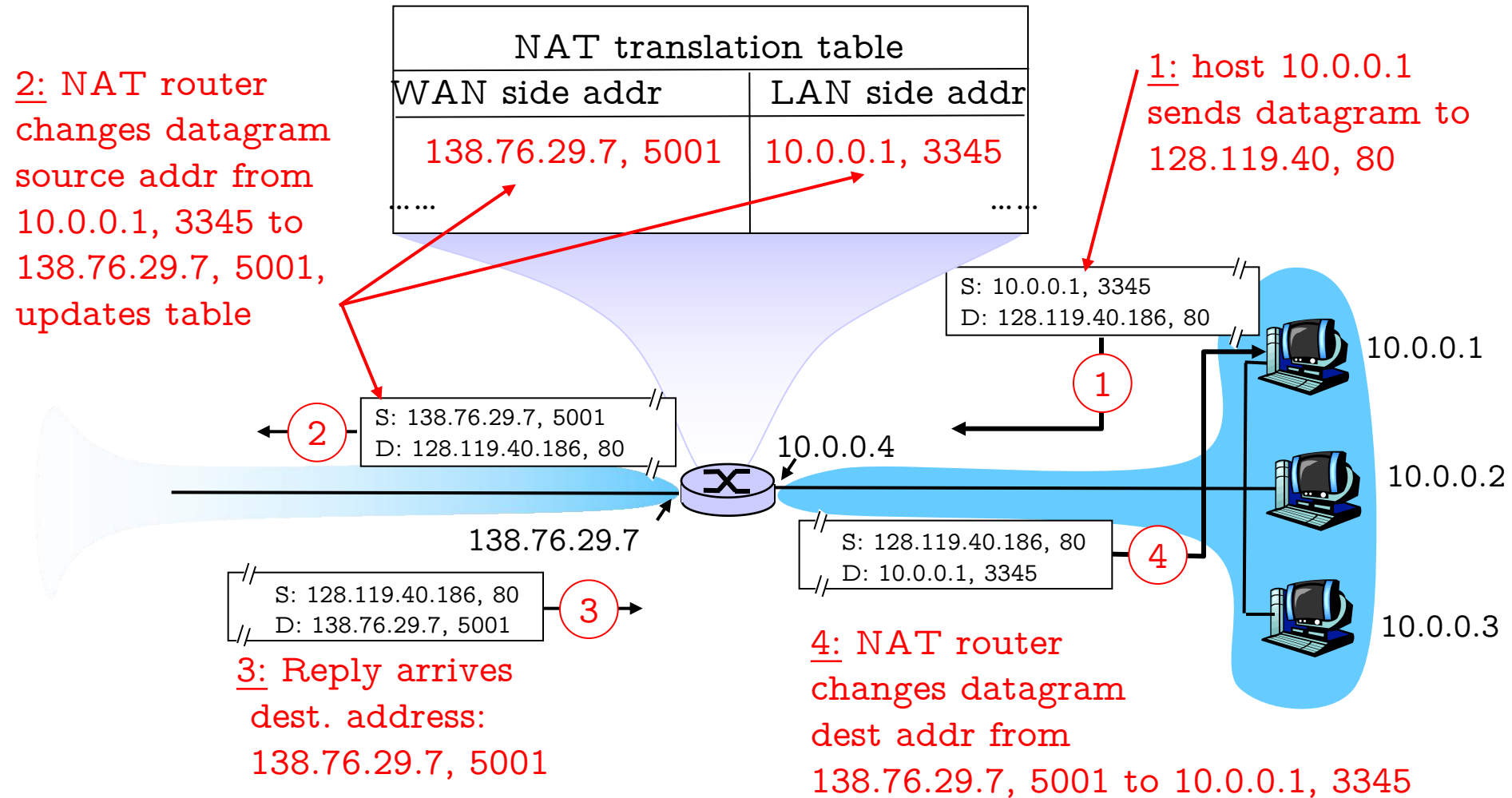
Datagrams with source or destination in this network have 10.0.0/24 address for source, destination (as usual)

NAT: Network Address Translation

Motivation: local network uses just one IP address as far as outside world is concerned:

- no need to be allocated range of addresses from **ISP**: - just 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, visible by outside world (**a security plus**).

NAT: Network Address Translation



Question?

Quiz

Define at least 2 functions of OSI layers and compare it with TCP/IP layer.