Network Layer

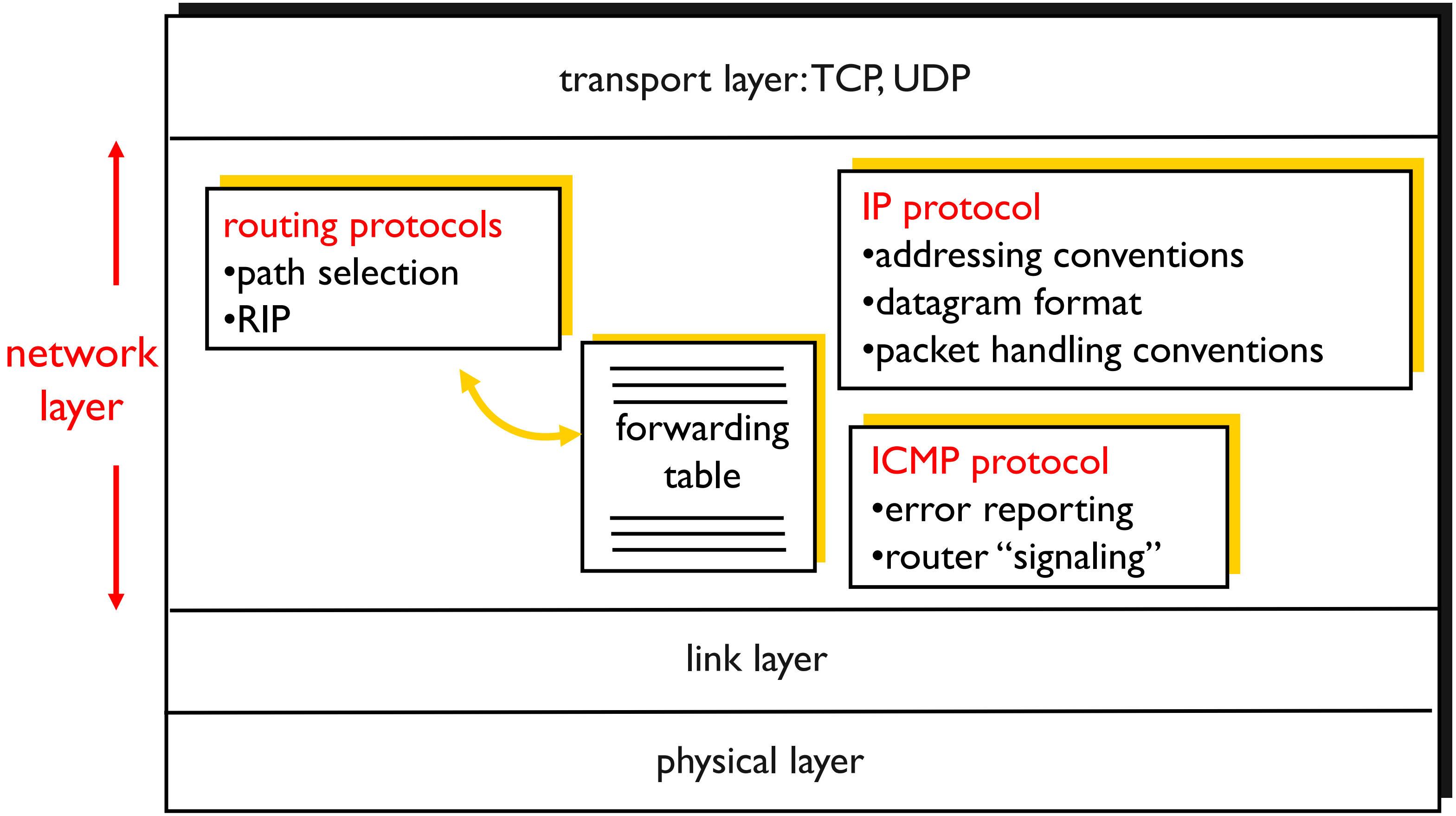
Instructor: C. Pu (Ph.D., Assistant Professor)

Lecture 14

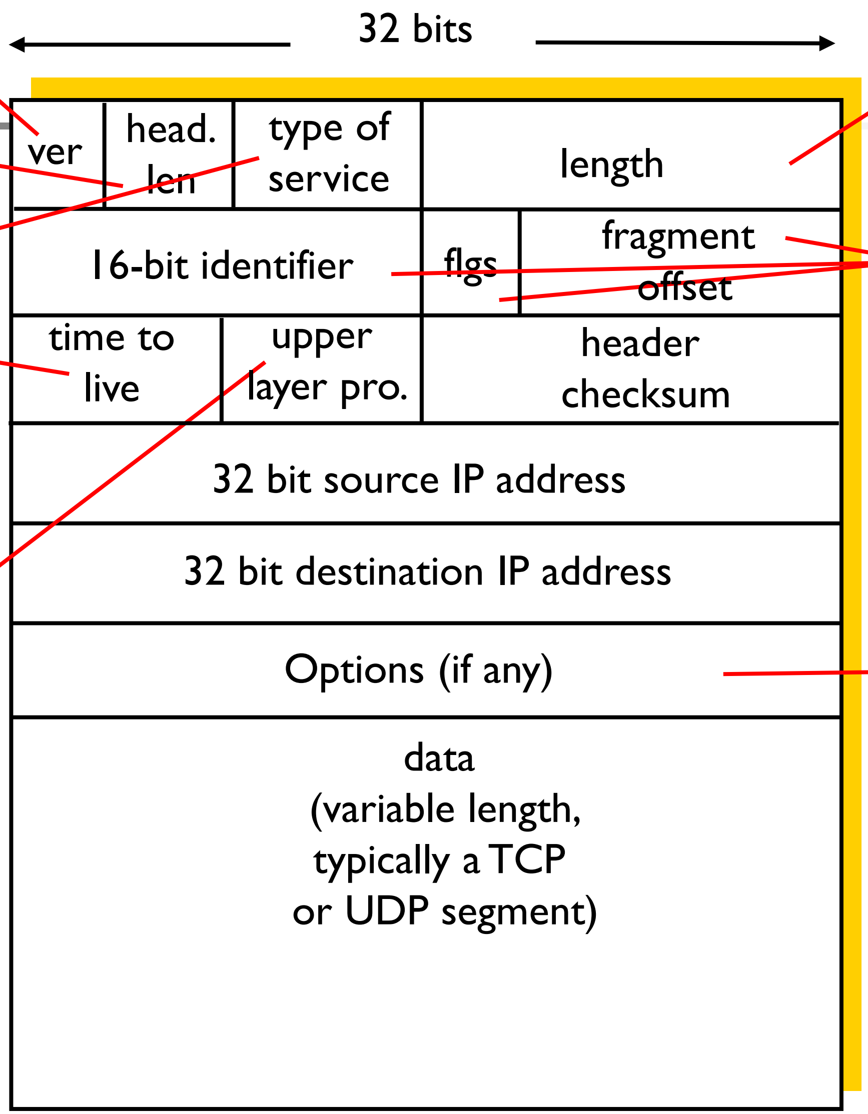
puc@marshall.edu

The Internet Network Layer: Host, Router Network Layer Functions

focus on how addressing and forwarding are done in the Internet!



IP Datagram Format



<table>
<tr><td>ver</td><td>head. len</td><td>type of service</td><td colspan="2">length</td></tr>
<tr><td colspan="3">16-bit identifier</td><td>flgs</td><td>fragment offset</td></tr>
<tr><td colspan="2">time to live</td><td>upper layer pro.</td><td colspan="2">header checksum</td></tr>
<tr><td colspan="5">32 bit source IP address</td></tr>
<tr><td colspan="5">32 bit destination IP address</td></tr>
<tr><td colspan="5">Options (if any)</td></tr>
<tr><td colspan="5">data
(variable length,
typically a TCP
or UDP segment)</td></tr>
</table>

IP protocol version number

header length (bytes)

“type” of data

max number remaining hops (decremented at each router)

upper layer protocol to deliver payload to

total datagram length (bytes)

for fragmentation/ reassembly

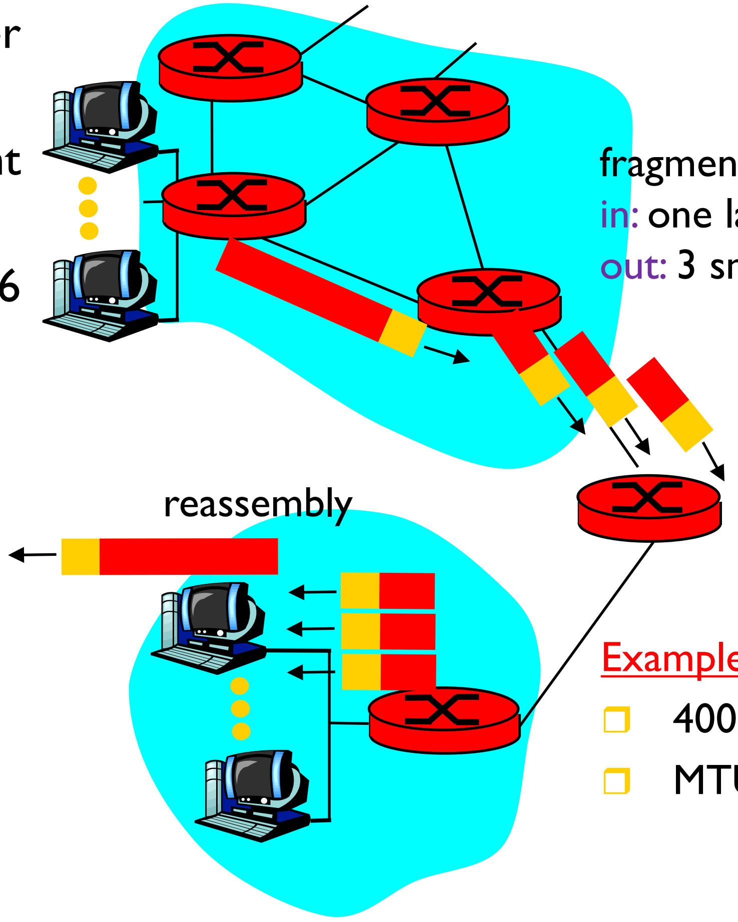
E.g. timestamp, record route taken, specify list of routers to visit.

how much overhead with TCP?

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

IP Fragmentation & Reassembly

- network links have MTU (max. transfer unit) - largest possible link-level frame
  - different link types have different MTUs
  - e.g., some wide-area link – 576 bytes
- large IP datagram divided ("fragmented") within network
  - one datagram becomes several datagrams
  - "reassembled" only at **final destination**
  - IP header bits used to identify order related fragments



fragmentation:
in: one large datagram
out: 3 smaller datagrams

reassembly

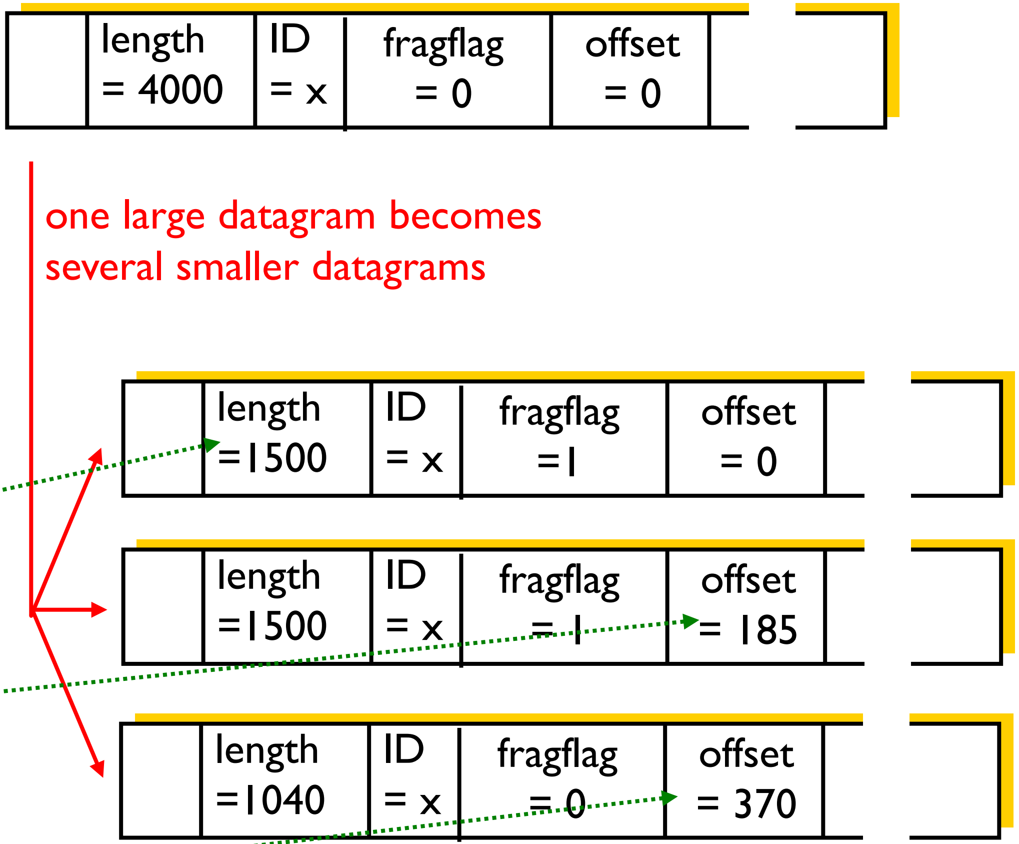
Example

- 4000 byte datagram
- MTU = 1500 bytes

IP Fragmentation & Reassembly (cont.)

Example

- 4000 byte datagram
  - 3980 bytes + 20 bytes IP header
- MTU = 1500 bytes



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

one large datagram becomes several smaller datagrams

length =1500	ID = x	fragflag =1	offset = 0
length =1500	ID = x	fragflag = 1	offset = 185
length =1040	ID = x	fragflag = 0	offset = 370

1480 bytes in data field + 20 bytes of IP header

offset = 185 = 1480 / 8

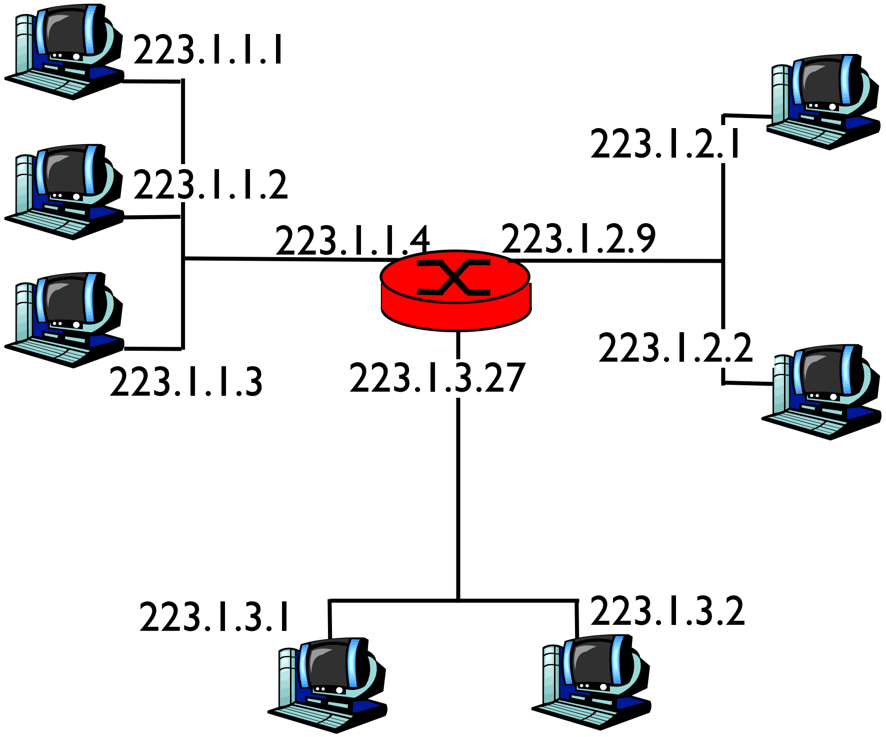
offset = 370 = 2960 / 8

offset is measured in terms of 8 bytes

IP Addressing: Introduction

IP address is technically associated with an interface, rather than with the host of router containing that interface!

- IP address: 32-bit identifier for host and 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***

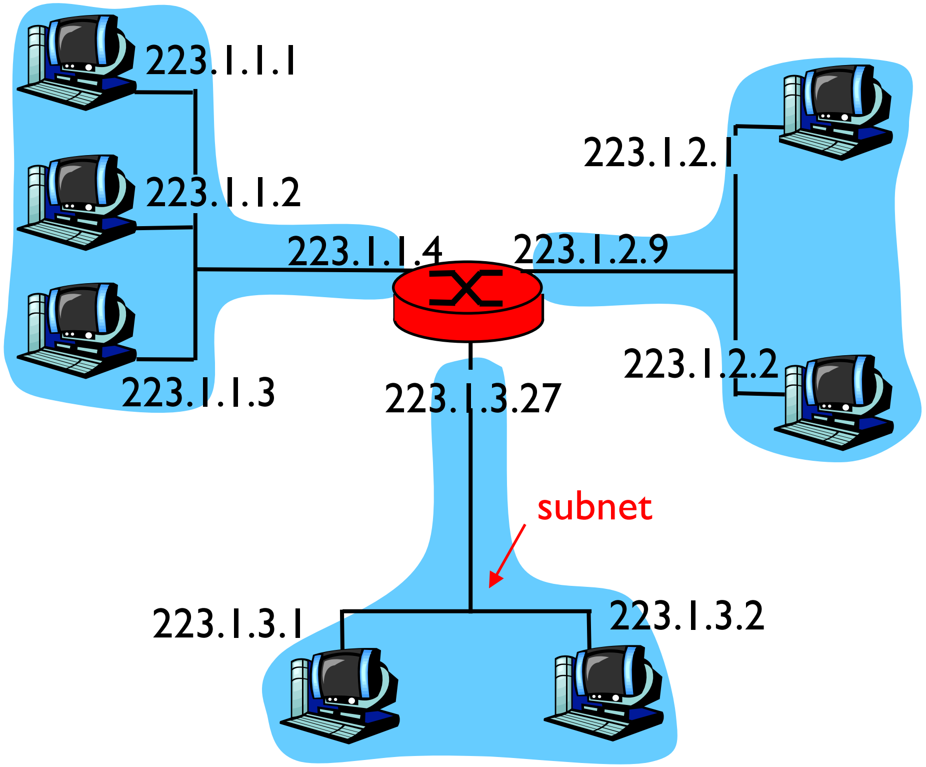


$$223.1.1.1 = \underbrace{11011111}_{223}\ \underbrace{00000001}_{1}\ \underbrace{00000001}_{1}\ \underbrace{00000001}_{1}$$

dotted-decimal notation: each byte is written in decimal form and is separated by a dot from other bytes

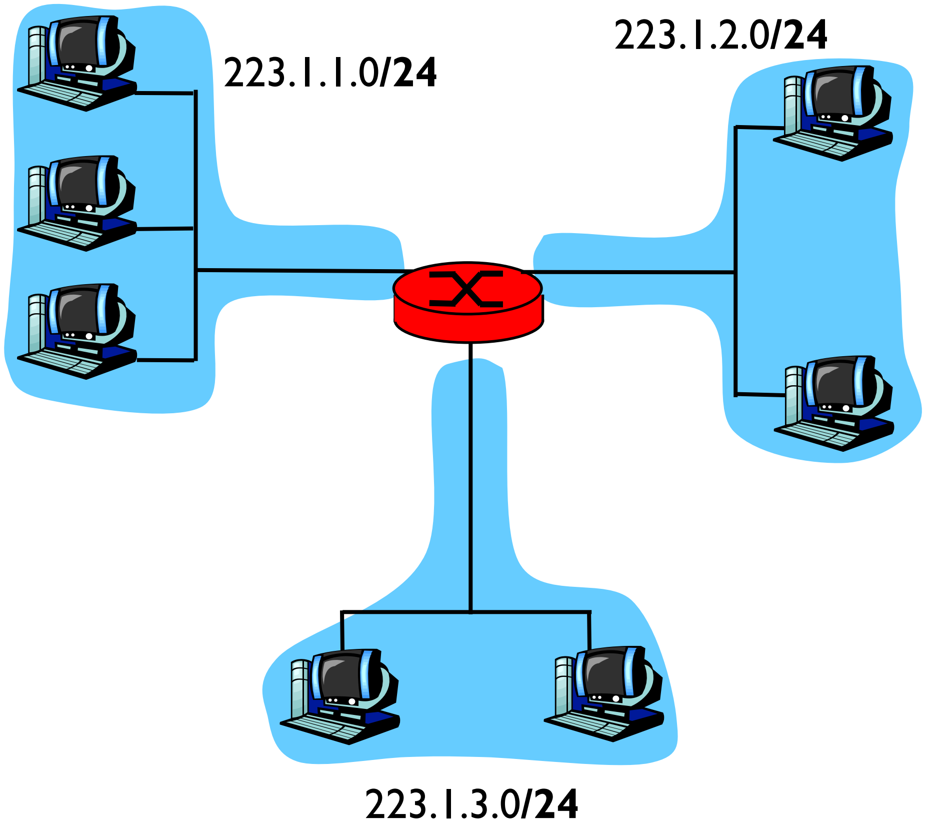
Subnets

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



network consisting of 3 subnets

Subnets (cont.)



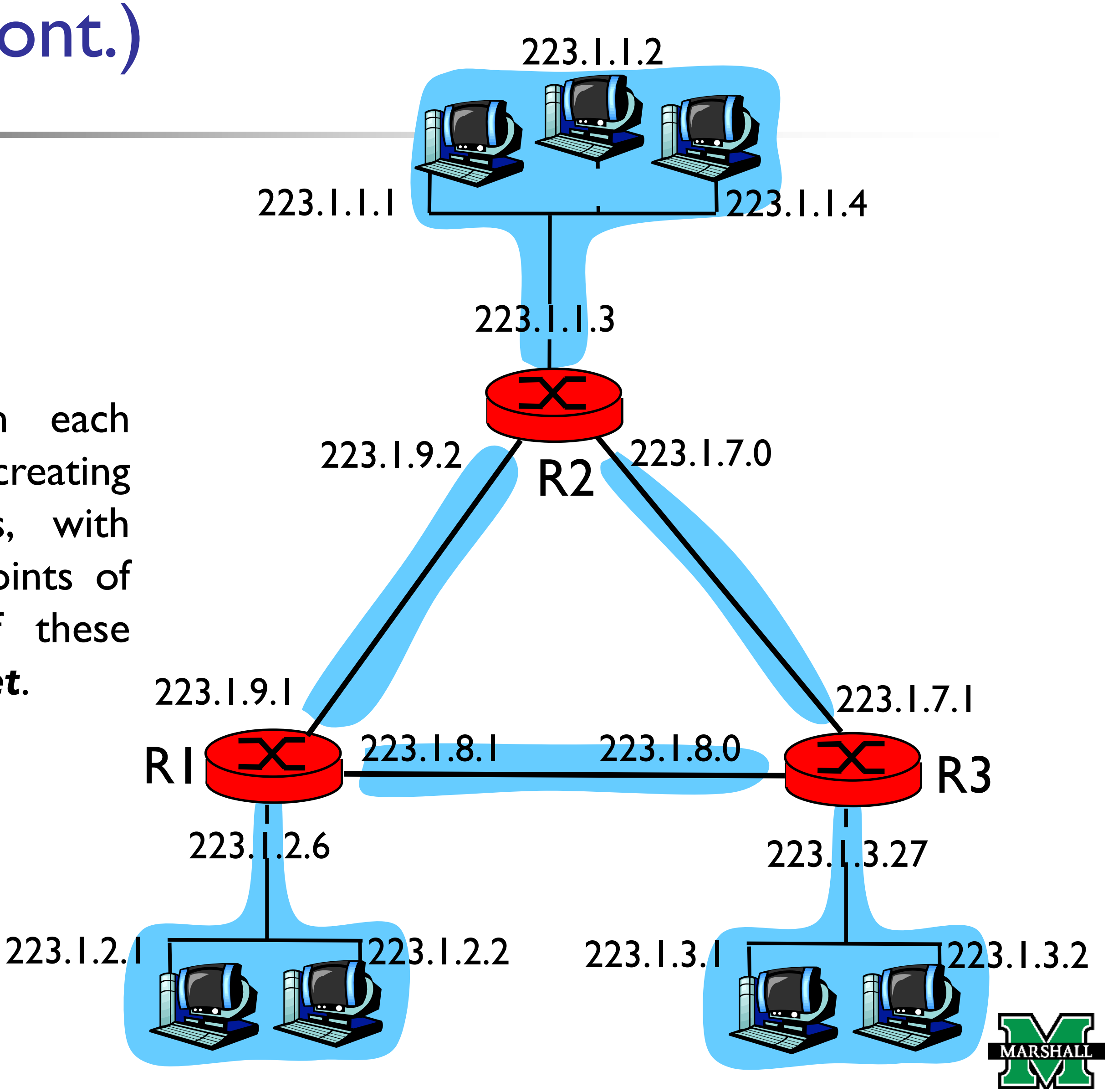
Subnet mask: /24, indicating the leftmost 24 bits of the 32-bit quantity define the subnet address.

Subnets (cont.)

Q: How many subnets?

A: 6

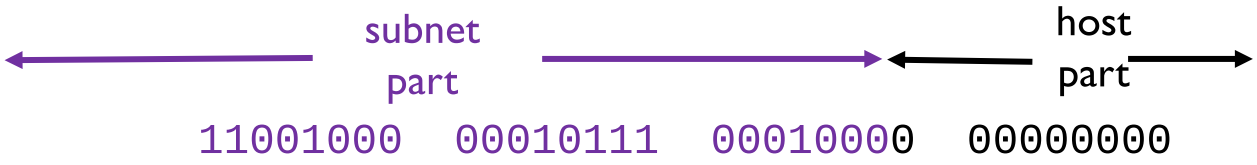
To determine the subnets, detach each interface from its host or router, creating islands of isolated networks, with interfaces terminating the end points of the isolated networks. Each of these isolated networks is called a ***subnet***.



IP Addressing: CIDR

- Internet's address assignment strategy

- CIDR: Classless InterDomain Routing
  - generalizes the notion of subnet addressing
  - 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

IP Addresses: How to get one?

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

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

IP Addresses: How to get one? (cont.)

- once an organization has obtained a block of address,
  - assign individual IP addresses to the host and router interfaces

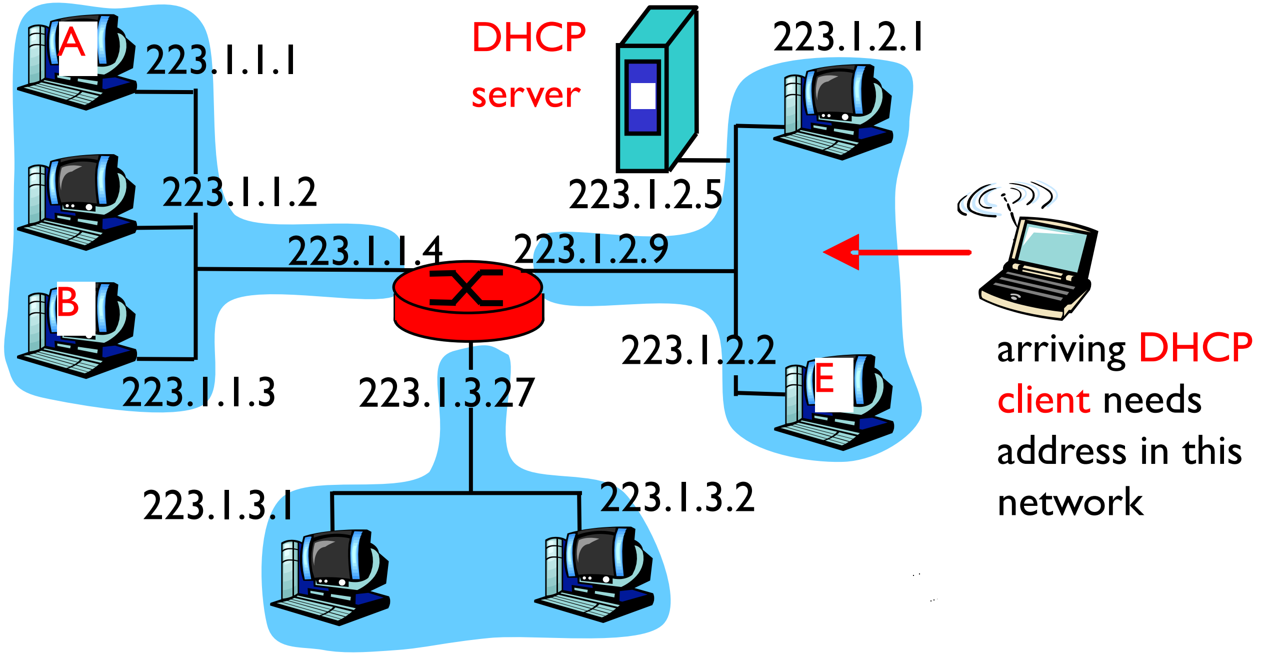
- Q: How does a ***host*** get IP address?
  - hard-coded by system admin in a file
  - DHCP: Dynamic Host Configuration Protocol: dynamically get address from a server
    - "plug-and-play"

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

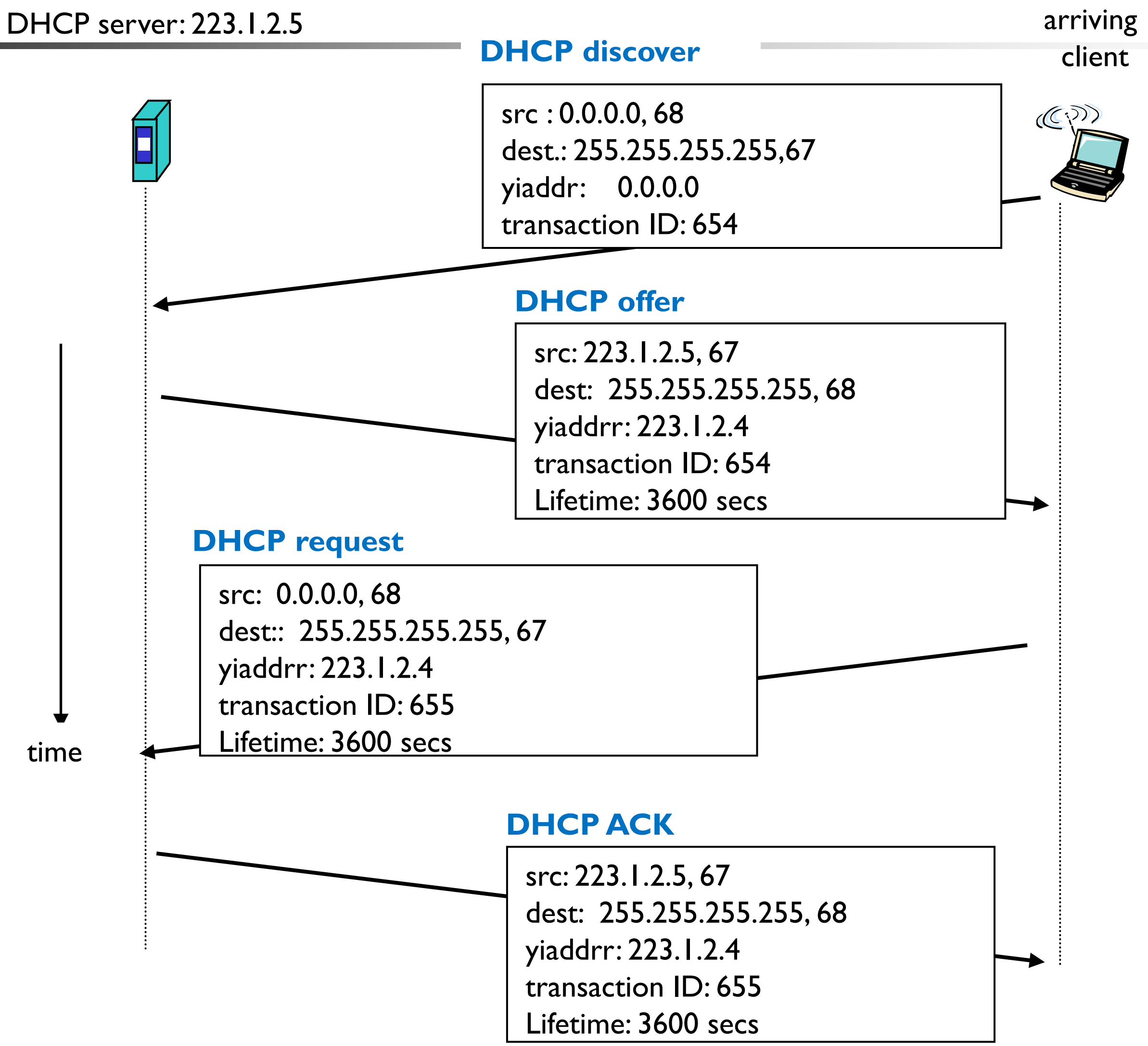
- DHCP overview:
  - host broadcasts "DHCP server discover" msg
  - DHCP server responds with "DHCP server offer" msg
  - host requests IP address: "DHCP request" msg
  - DHCP server sends address: "DHCP ack" msg

DHCP Client-Server Scenario



arriving DHCP client needs address in this network

DHCP Client-Server Scenario



DHCP server: 223.1.2.5

arriving client

DHCP discover

src : 0.0.0.0, 68
dest.: 255.255.255.255,67
yiaddr: 0.0.0.0
transaction ID: 654

DHCP offer

src: 223.1.2.5, 67
dest: 255.255.255.255, 68
yiaddrr: 223.1.2.4
transaction ID: 654
Lifetime: 3600 secs

DHCP request

src: 0.0.0.0, 68
dest:: 255.255.255.255, 67
yiaddrr: 223.1.2.4
transaction ID: 655
Lifetime: 3600 secs

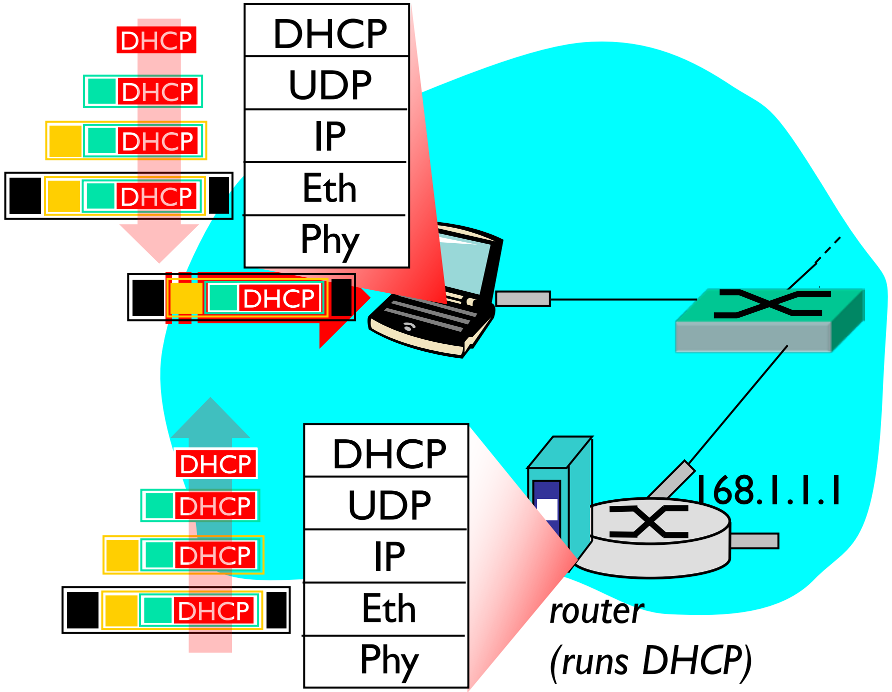
DHCP ACK

src: 223.1.2.5, 67
dest: 255.255.255.255, 68
yiaddrr: 223.1.2.4
transaction ID: 655
Lifetime: 3600 secs

DHCP: More Than IP Address

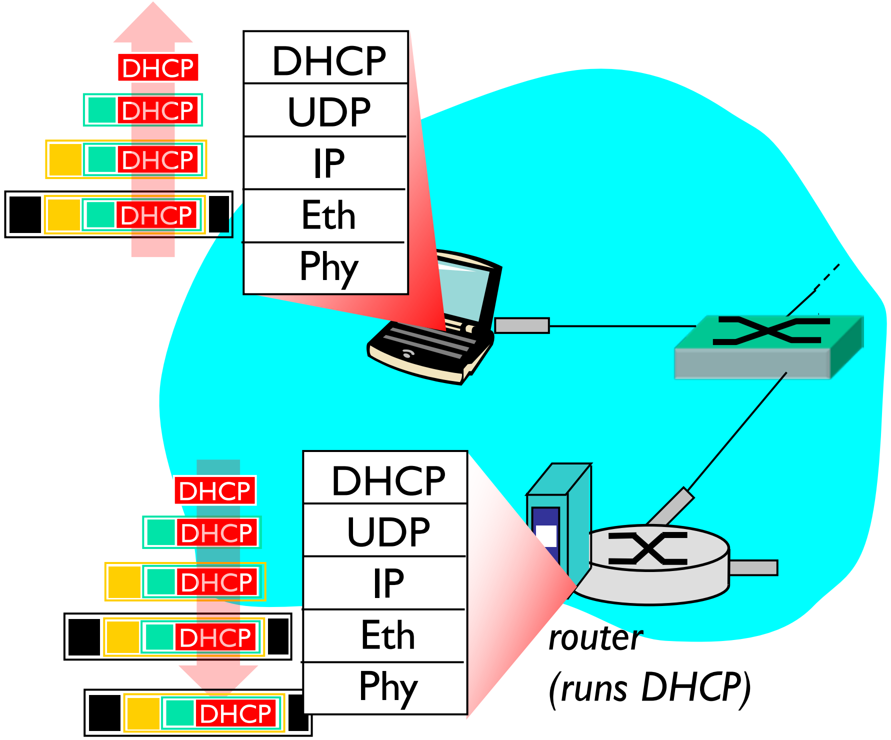
- DHCP can return more than just allocated IP address on subnet:
  - address of first-hop router for client
  - name and IP address of DNS sever
  - network mask (indicating network versus host portion of address)

DHCP: Example



- connecting laptop needs its IP address, addr of first-hop router, addr of DNS server: use DHCP
  - DHCP request encapsulated in UDP, encapsulated in IP, encapsulated in 802.1 Ethernet
  - Ethernet frame broadcast (dest: FFFFFFFFFFFF) on LAN, received at router running DHCP server

DHCP: Example (cont.)

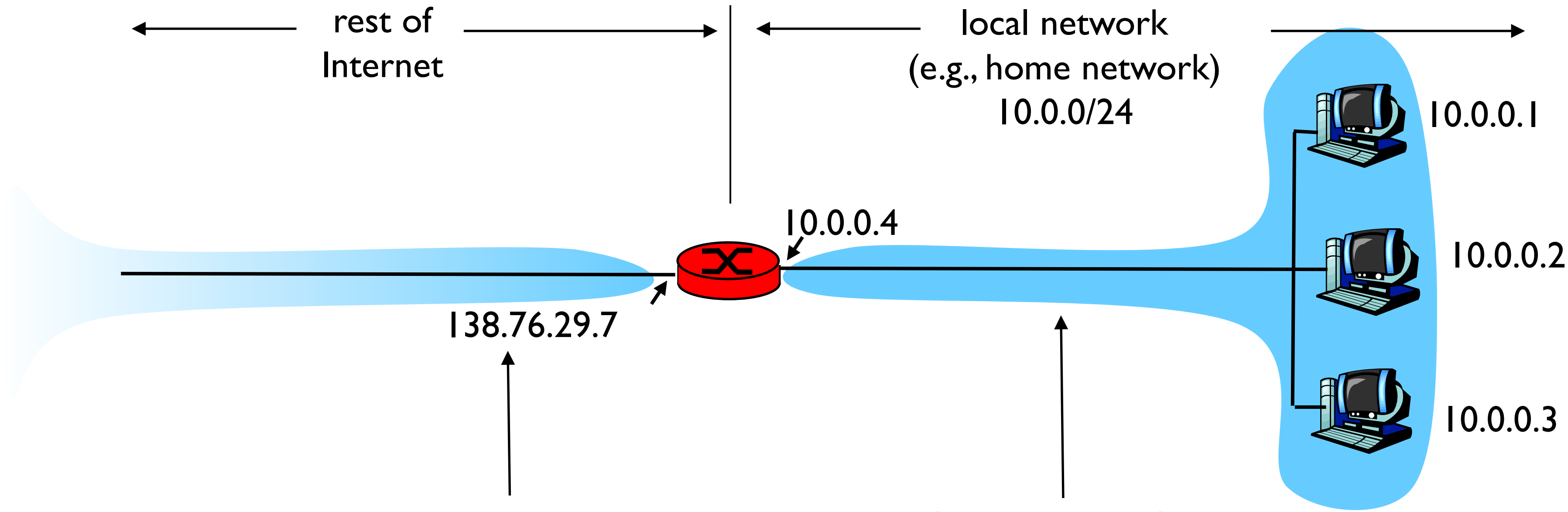


- DHCP server formulates DHCP ACK containing client's IP address, IP address of first-hop router for client, name & IP address of DNS server

- encapsulation of DHCP server, frame forwarded to client, demux'ing up to DHCP at client

- client now knows its IP address, name and IP address of DSN server, IP address of its first-hop router

NAT: Network Address Translation



rest of Internet

local network (e.g., home network) 10.0.0/24

10.0.0.1

10.0.0.2

10.0.0.3

10.0.0.4

138.76.29.7

all datagrams *leaving* local network have same single source NAT IP address: 138.76.29.7. *all* traffic *entering* local network have same destination address: 138.76.29.7.

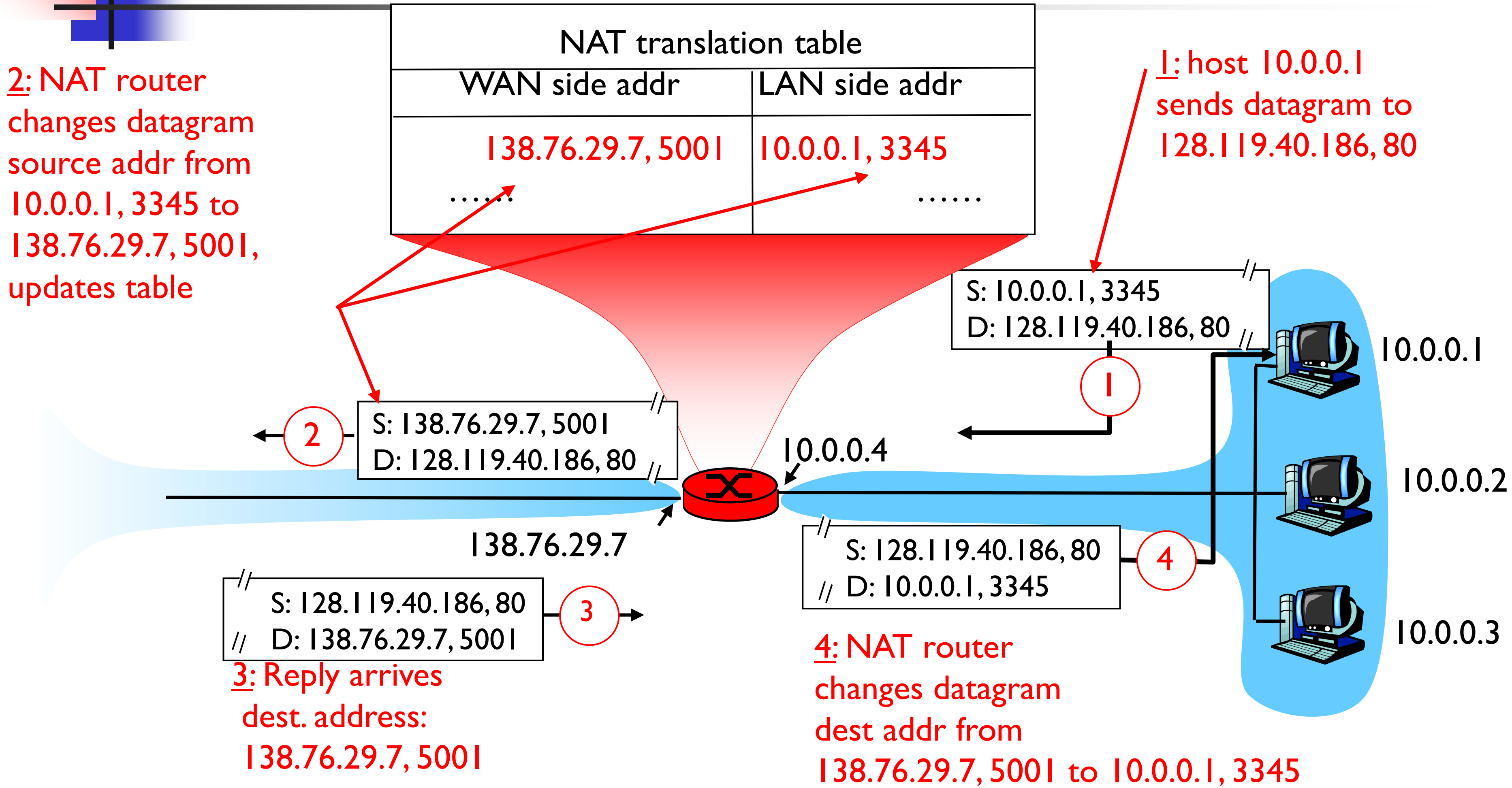
datagrams with source or destination in this network have 10.0.0/24 address for source and destination (as usual)

The address space **10.0.0.0/8** is reserved for a private networks or a realm with private address

NAT: Network Address Translation (cont.)

- 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, e.g., NAT-enabled router
  - 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 network not explicitly addressable and visible by outside world

NAT: Network Address Translation (cont.)



NAT translation table

WAN side addr	LAN side addr
138.76.29.7, 5001	10.0.0.1, 3345
……	……

1: host 10.0.0.1 sends datagram to 128.119.40.186, 80

S: 10.0.0.1, 3345
D: 128.119.40.186, 80

2: NAT router changes datagram source addr from 10.0.0.1, 3345 to 138.76.29.7, 5001, updates table

S: 138.76.29.7, 5001
D: 128.119.40.186, 80

S: 128.119.40.186, 80
D: 138.76.29.7, 5001

3: Reply arrives dest. address: 138.76.29.7, 5001

S: 128.119.40.186, 80
D: 10.0.0.1, 3345

4: NAT router changes datagram dest addr from 138.76.29.7, 5001 to 10.0.0.1, 3345

NAT: Network Address Translation (cont.)

- Implementation: NAT router must:
  - *outgoing datagrams: replace* (source IP address, port #) of every outgoing datagram to (NAT IP address, new port #)
    - remote clients/servers will respond using (NAT IP address, new port #) as destination addr.
  - *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

IPv6

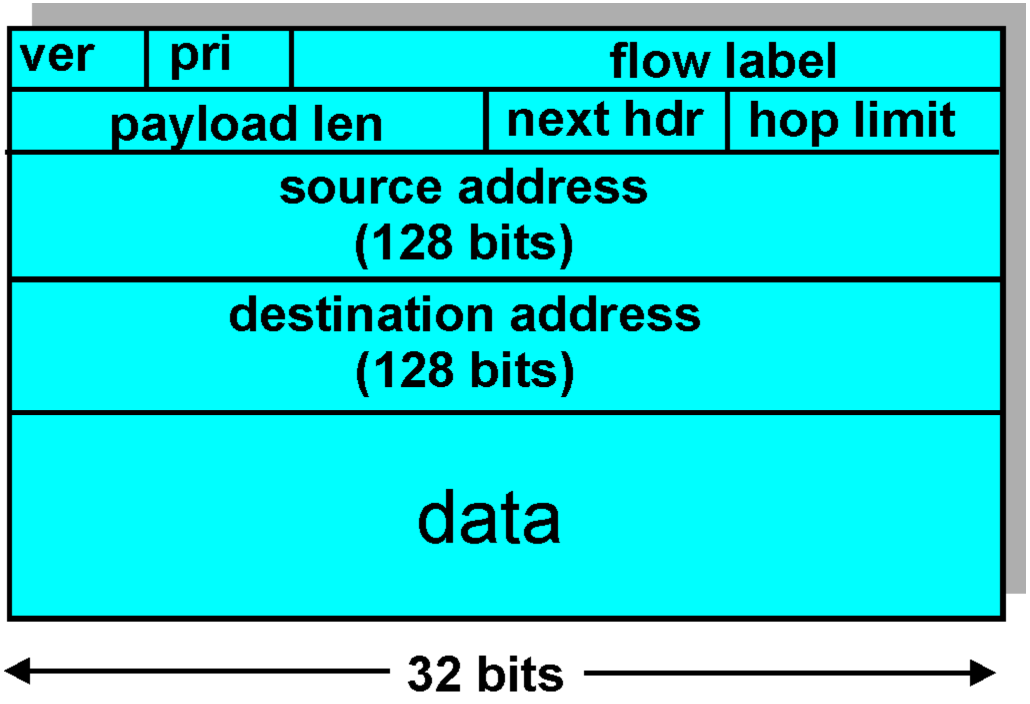
- Initial motivation:
  - 32-bit address space soon to be completely allocated
- additional motivation:
  - header format helps speed processing/forwarding
  - header changes to facilitate QoS
- IPv6 datagram format:
  - fixed-length 40 byte header
  - no fragmentation and reassembly allowed at intermediate routers
    - these operations can be performed only by the **source** and **destination**

IPv6 Header

Priority: identify priority among datagrams in flow

Flow Label: identify a flow of datagrams

Next header: identify upper layer protocol for data (e.g., TCP or UDP)



ver | pri | flow label

payload len | next hdr | hop limit

source address (128 bits)

destination address (128 bits)

data

32 bits

Other Changes from IPv4

- *Checksum:* removed entirely to reduce processing time at each hop
  - the transport & link layers performs checksum
- *ICMPv6:* new version of ICMP
  - additional message types, e.g., “Packet Too Big” due to no fragmentation / reassembly
  - If an IPv6 datagram is too big to forward?
    - simply drop & send a “Packet Too Big” ICMP error message back
    - the sender will send a smaller IP datagram
    - → speed up IP forwarding within the network