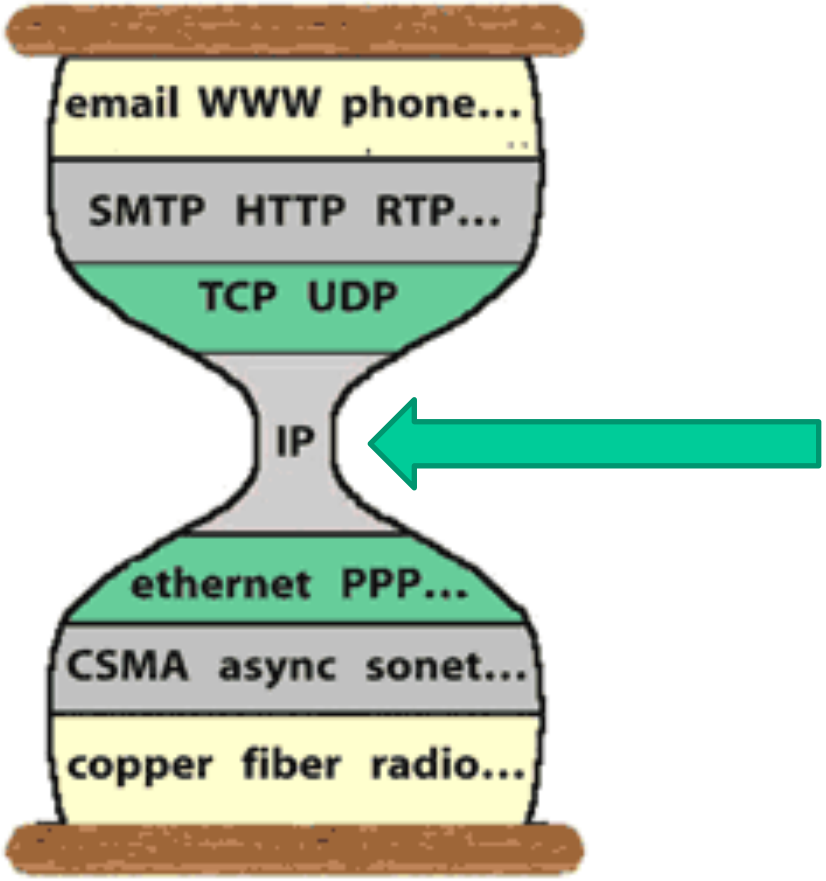
Chapter 4: Network Layer



4. 1 Introduction

4.2 Virtual circuit and datagram networks

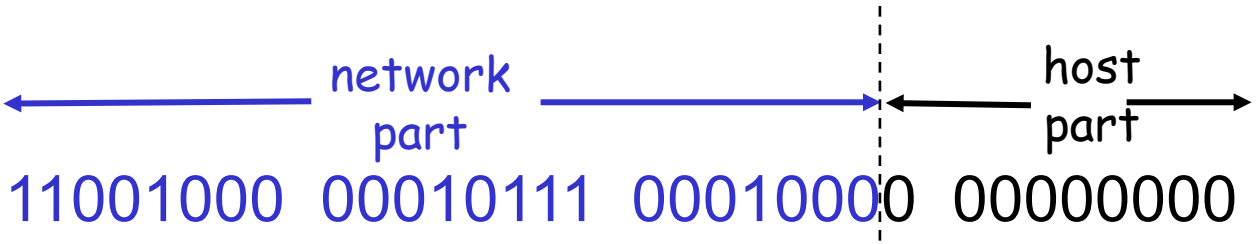
4.3 What's inside a router

4.4 IP: Internet Protocol

- IP datagram format
- IPv4 addressing
- ICMP
- IPv6

This and next week

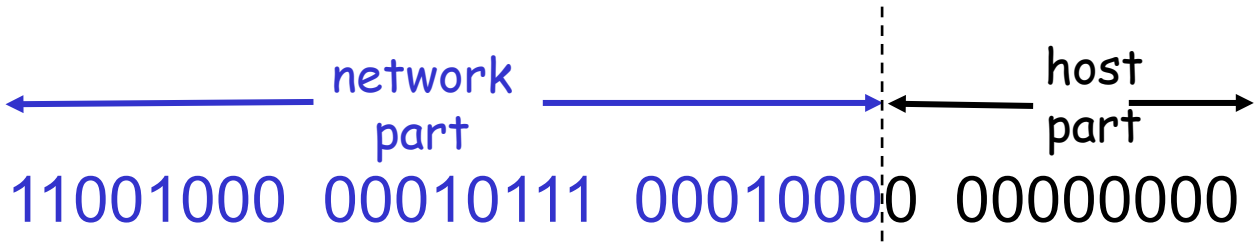
Classless InterDomain Routing



- Internet Service Providers (ISPs), and some large user sites, get blocks of IP addresses from the Regional Internet Registries (RIRs)
- Internet customers get a sub-block from their ISP's address block

ISP's block	11001000 00010111 00010000 00000000
Organization 0	11001000 00010111 0001**000**0 00000000
Organization 1	11001000 00010111 0001**001**0 00000000
Organization 2	11001000 00010111 0001**010**0 00000000
...	....
Organization 7	11001000 00010111 0001**111**0 00000000

CIDR Address Format



- a.b.c.d/x, x = #bits in network ID portion of the address
- address: a.b.c.d, network mask 1111111…10000000

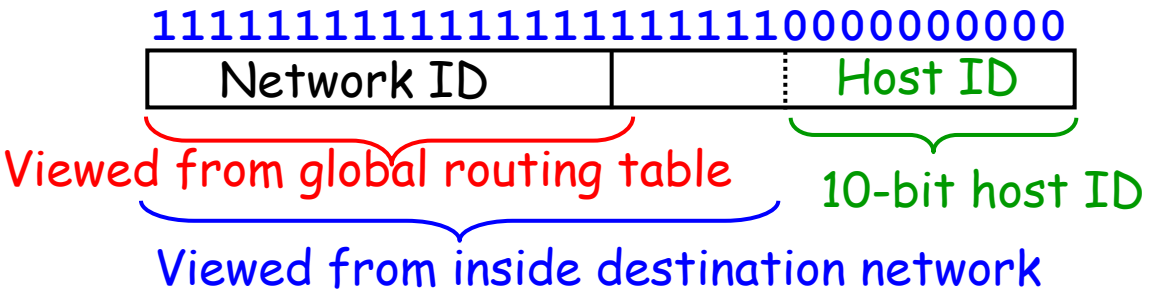
x number of 1s

200.23.16.0/23 **address 200.23.16.0, netmask 255.255.254.0**

IP Subnet

important

- subnet mask: indicates the portion of the address that is considered as "network ID" *by the local site*
  - Does not need to align with byte boundary



- subnets are invisible outside the local site
  - backbone routers only know how to forward packets to the networkID
  - Within the organization:
    - routers store: [subnet, mask, next hop]
    - Each host is configured with an IP address and a subnet mask

Special Addresses

- 0.0.0.0/8
  - “this network”
- 255.255.255.255/32
  - broadcast address of “this network”
- first address of the network (e.g., 192.168.1.0 for 192.168.1.0/24)
  - network address (cannot be used for end-hosts)
- last address of the network (e.g., 192.168.1.255 for 192.168.1.0/24)
  - broadcast address for the network

More Special Stuff

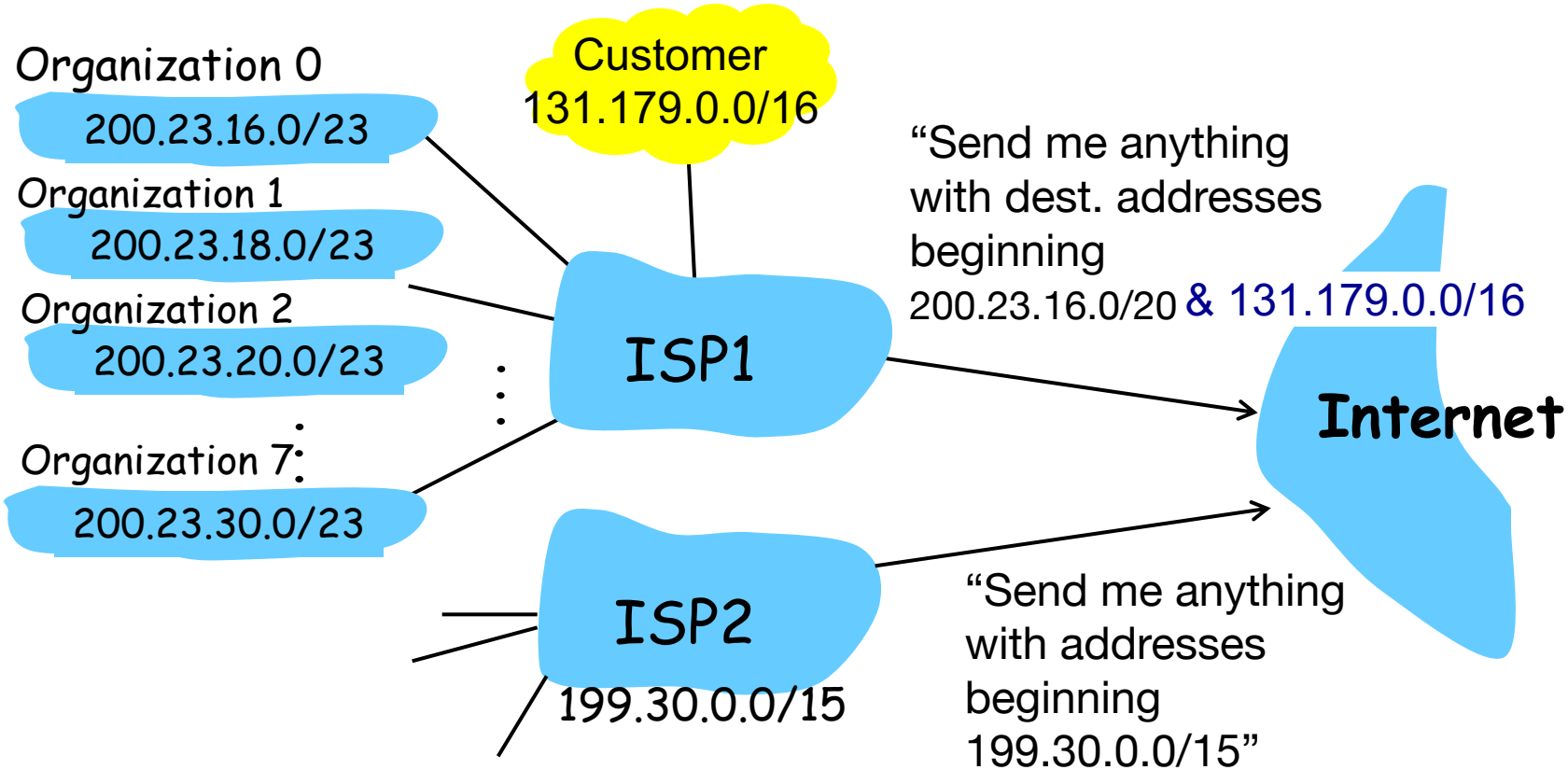
- Loopback
  - 127.0.0.0/8
- Link-local
  - 169.254.0.0/16
- Private-use networks
  - 10.0.0.0/8
  - 172.16.0.0/12
  - 192.168.0.0/16
- and more, refer to RFC 5735

CIDR Address Calculations

- What is the netmask?
  - 131.179.196.0/24
  - 169.232.34.48/30
  - 196.22.136.0/21
- What is # of bits in network portion?
  - address 93.181.192.0, netmask 255.255.224.0
  - address 10.128.0.0, netmask 255.192.0.0
- How many endpoint addresses are in the network in all above cases?

Hierarchical addressing: route aggregation

Hierarchical addressing enables *route aggregation*, which in turn helps reduce global routing table size



Longest prefix matching

longest prefix matching

when looking for forwarding table entry for given destination address, use *longest* address prefix that matches destination address.

Destination Address Range	Link interface
11001000 00010111 00011000 *********	0
11001000 00010111 00010*** *********	1
11001000 00010111 0001**** *********	2
******** ******* ******* *******	3

examples:

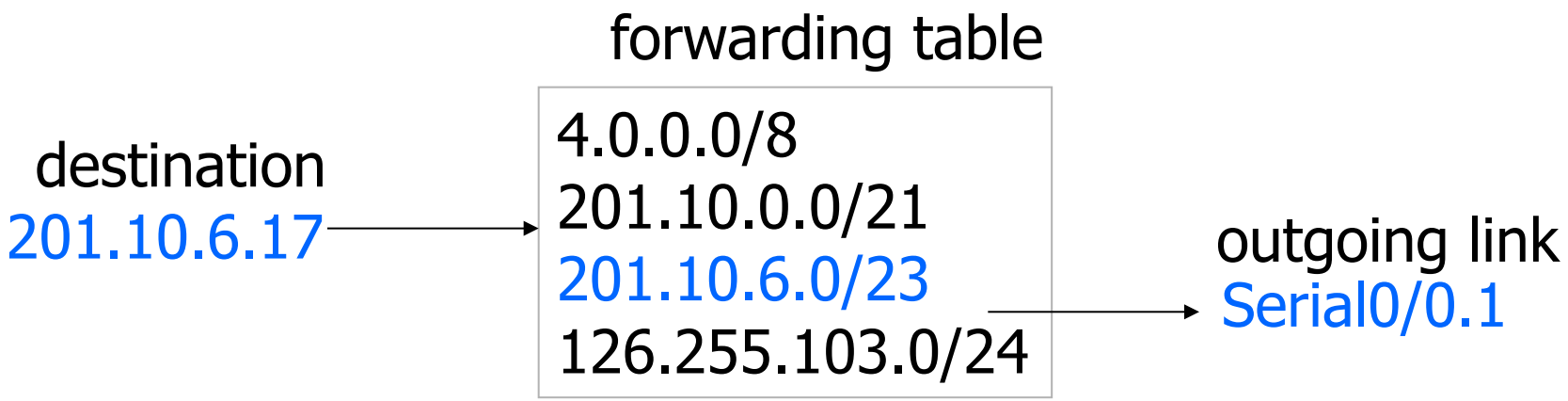
DA: 11001000 00010111 00010110 10100001 which interface?

DA: 11001000 00010111 00011000 10101010 which interface?

(destination address)

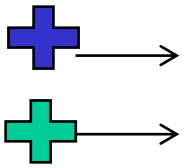
Longest Prefix Match Forwarding

- Forwarding tables in IP routers
  - Maps each IP prefix to next-hop link(s)
- Destination-based forwarding
  - Packet has a destination address
  - Router identifies longest-matching prefix
  - Interesting algorithmic problem: very fast lookups



Simplest Algorithm

- Order items in routing table by the length of the network
- Scan the forwarding table one entry at a time
  - See if the destination matches the entry
  - If so, check the size of the mask for the prefix
  - Keep track of the entry with longest-matching prefix



126.255.103.0/24	126.255.103.0	255.255.255.0
201.10.6.0/23	201.10.6.0	255.255.254.0
201.10.0.0/21	201.10.0.0	255.255.248.0
4.0.0.0/8	4.0.0.0	255.0.0.0

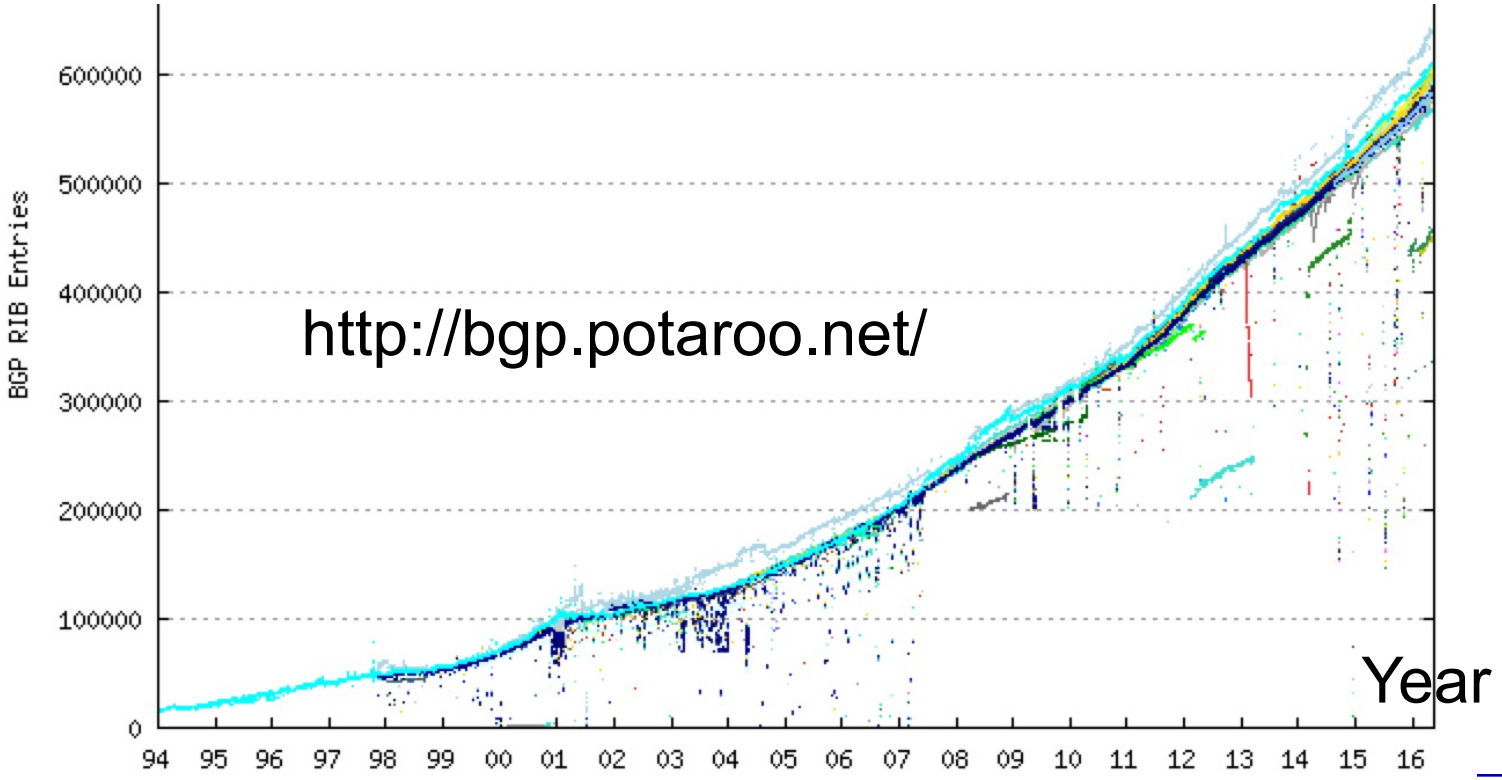
201.10.7.17 **201.10.8.17**

```
126.255.103.0 ?= 201.10.7.17 & 255.255.255.0
201.10.6.0    ?= 201.10.7.17 & 255.255.254.0

201.10.6.0    ?= 201.10.8.17 & 255.255.254.0
201.10.0.0    ?= 201.10.8.17 & 255.255.248.0
```

Simplest Algorithm is Too Slow

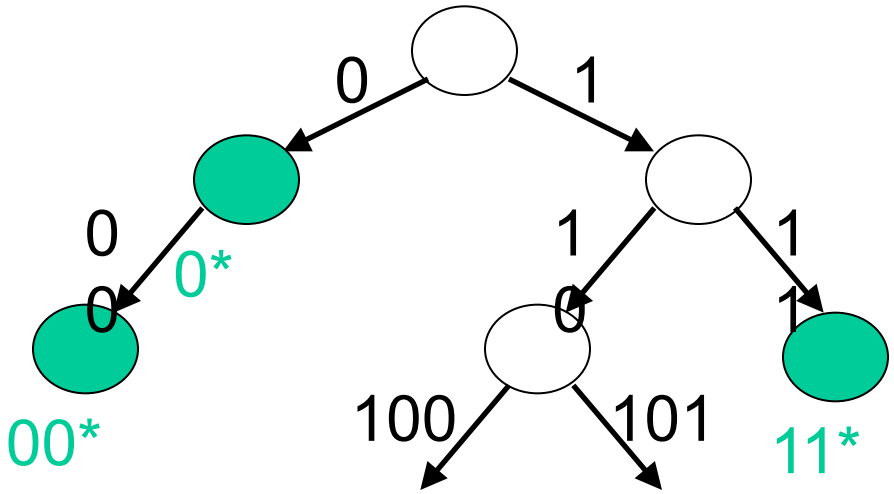
- Overhead is linear in size of the forwarding table
  - Today, that means 150,000-600,000 entries!
  - And, the router may have just a few nanoseconds
  - … before the next packet is arriving



Patricia Tree

FYI

- Store the prefixes as a tree
  - One bit for each level of the tree
  - Some nodes correspond to valid prefixes
  - ... which have next-hop interfaces in a table
- When a packet arrives
  - Traverse the tree based on the destination address
  - Stop upon reaching the longest matching prefix



Even Faster Lookups

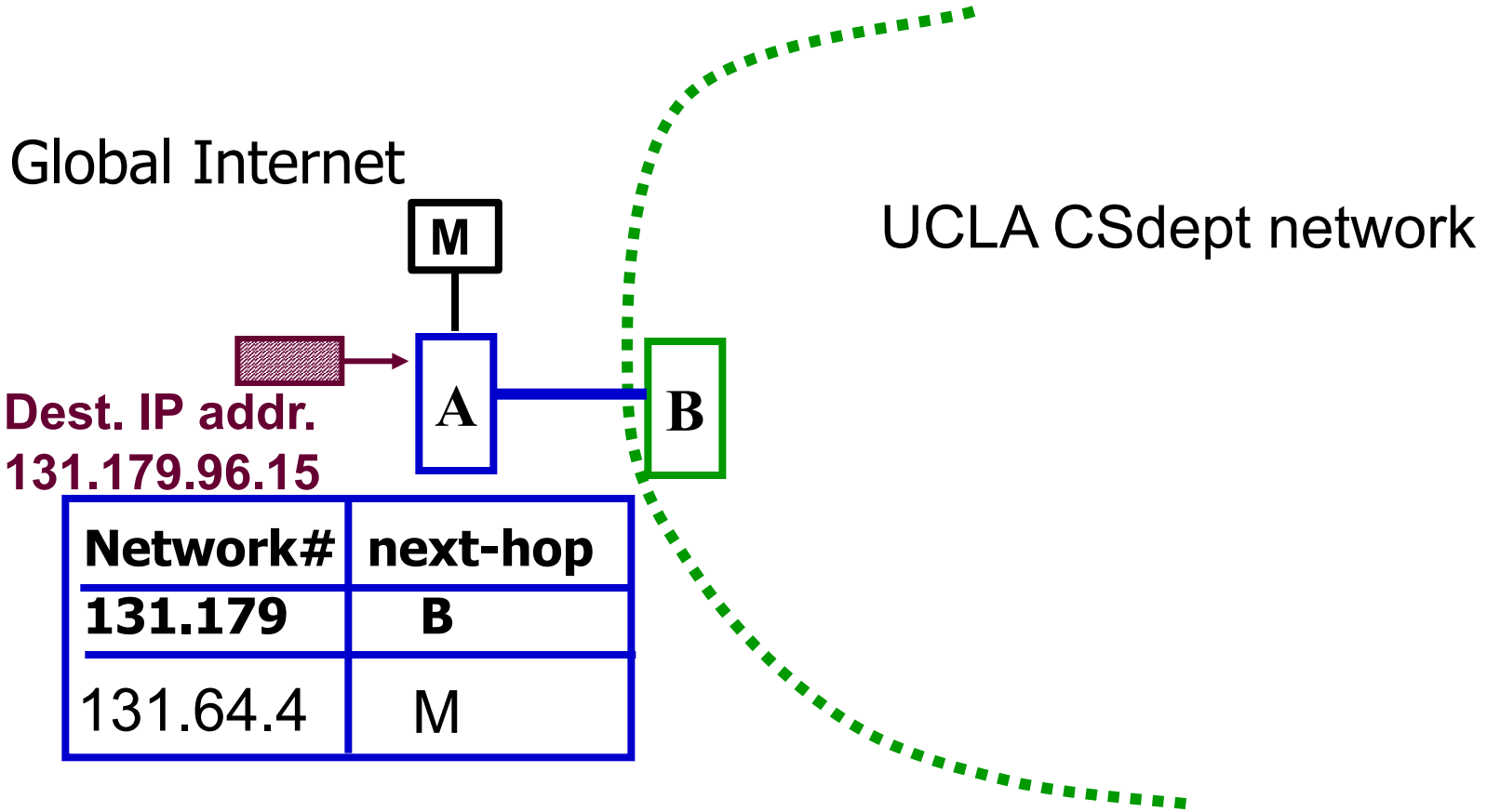
FYI

- Patricia tree is faster than linear scan
  - Proportional to number of bits in the address
- Patricia tree can be made faster
  - Can make a k-ary tree
    - E.g., 4-ary tree with four children (00, 01, 10, and 11)
  - Faster lookup, though requires more space
- Can use special hardware
  - Content Addressable Memories (CAMs)
  - Allows look-ups on a key rather than flat address
- Huge innovations in the mid-to-late 1990s
  - After CIDR was introduced (in 1994)
  - … and longest-prefix match was a major bottleneck

Where do Forwarding Tables Come From?

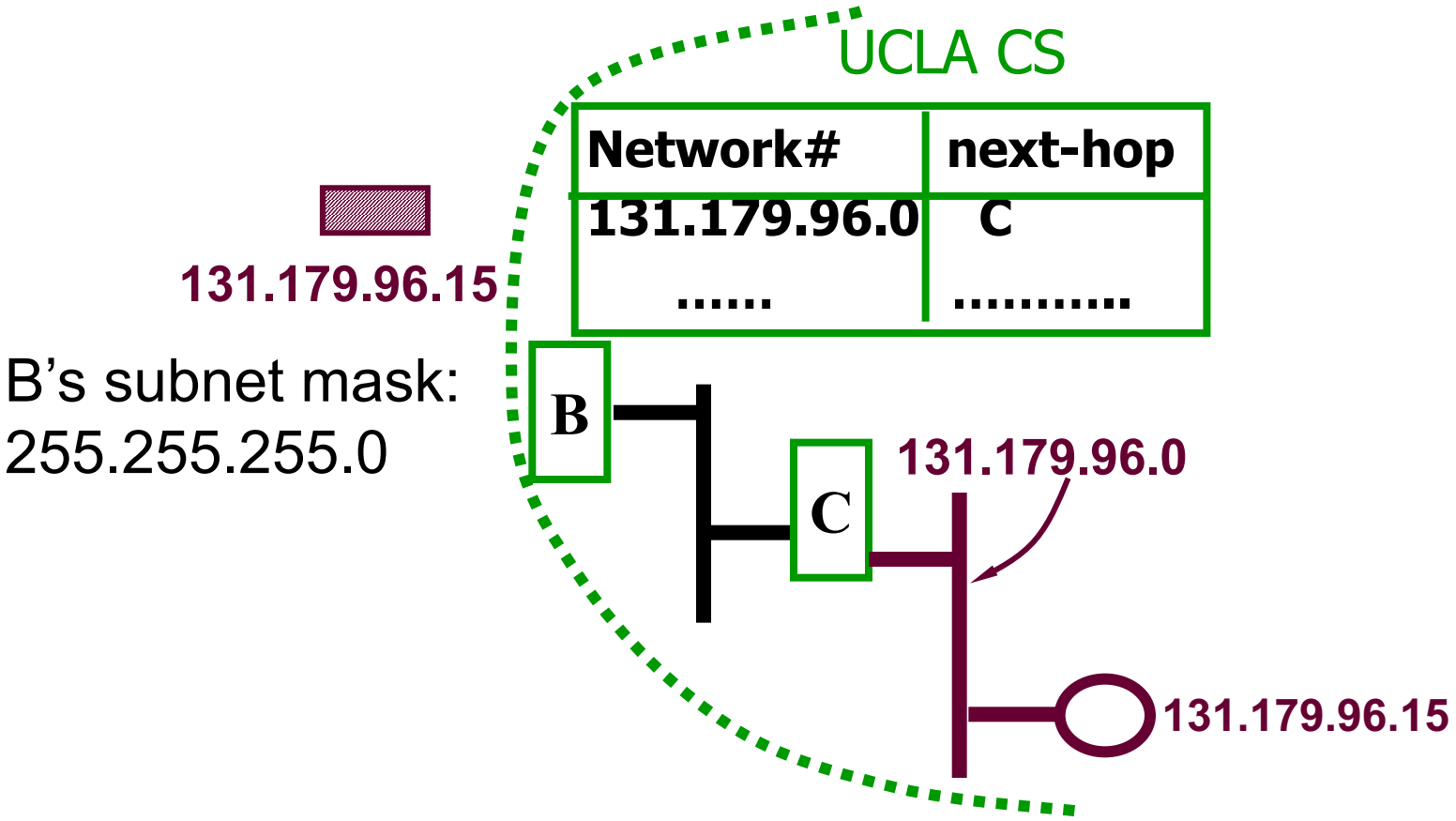
- Routers have forwarding tables
  - Map prefix to outgoing link(s)
- Entries can be statically configured
  - E.g., "map 12.34.158.0/24 to Serial0/0.1"
- But, this doesn't adapt
  - To failures
  - To new equipment
  - To the need to balance load
  - …
- That is where other technologies come in…
  - Routing protocols, DHCP, and ARP (later in course)

An example



Network#	next-hop
131.179	**B**
131.64.4	M

An example



subnet mask(255.255.255.0)	**11111111111111111111111**	**00000000**
subnetted address	**131 . 179 . 96**	**15**

Is routing table exist only on routers?

Routing table is needed on every IP host: it is needed to determine where to send packet

What is routing table on your computer right now?
https://kb.wisc.edu/ns/page.php?id=12364

OSX: netstat -f inet -rn
Windows: netsh interface ipv4 show route
Linux: netstat -A inet -rn

Examples

```
[cawka@benz ~]$ netstat -A inet -rn
Kernel IP routing table
Destination     Gateway         Genmask         Flags   MSS Window  irtt Iface
0.0.0.0         131.179.196.1   0.0.0.0         UG        0 0          0 br0
131.179.196.0   0.0.0.0         255.255.255.0   U         0 0          0 br0
```

```
~ $ netstat -f inet -rn
Routing tables

Internet:
Destination        Gateway            Flags        Refs      Use   Netif Expire
default            131.179.196.1      UGSc          930     3391     en3
default            link#13            UCSI            1        0 bridge1
127                127.0.0.1          UCS             2    17279     lo0
127.0.0.1          127.0.0.1          UH              5 25390421     lo0
127.255.255.255    127.0.0.1          UHWIi           1        1     lo0
131.179.196/24     link#12            UCS             9        0     en3
131.179.196.1/32   link#12            UCS             4        0     en3
131.179.196.1      0:0:c:7:ac:c4      UHLWIir       917        0     en3   1156
131.179.196.47     link#12            UHLWIi          1      200     en3
131.179.196.69     0:8:74:92:b3:8a    UHLWIi          1        5     en3    787
131.179.196.186    0:d:a2:1:b:c8      UHLWIi          1      315     en3   1114
131.179.196.220/32 link#12            UCS             2        0     en3
131.179.196.220    68:5b:35:c0:61:b6  UHLWIi          2        9     lo0
131.179.196.242    link#12            UHLWIi          1        0     en3
131.179.196.255    link#12            UHLWbI          1     1273     en3
169.254            link#12            UCS             2        0     en3
169.254.255.255    link#12            UHLSW           1     1272     en3
224.0.0            link#12            UmCS            2        0     en3
224.0.0.251        1:0:5e:0:0:fb      UHmLWI          1      142     en3
255.255.255.255/32 link#12            UCS             2        0     en3
255.255.255.255    link#12            UHLWbI          1      574     en3
```

Getting an IP packet from source to dest.

Assuming subnet mask = 255.255.255.0

Source host A → destination B:

Host A: [A's addr & subnet mask] = [B's addr & subnet mask] ?

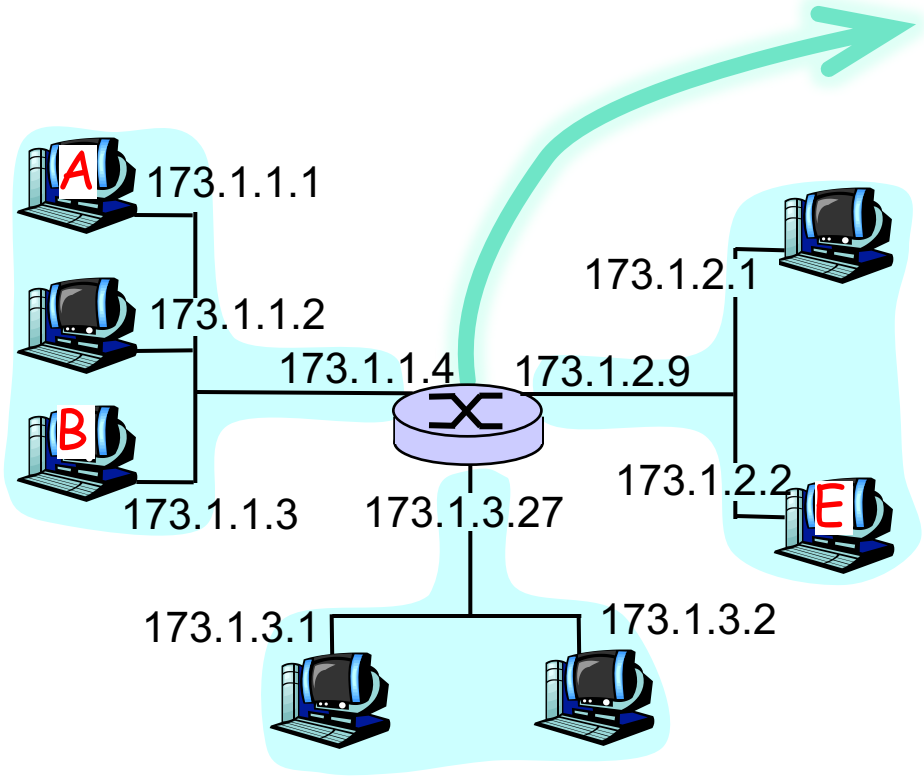
yes: B is on the same net, use link layer protocol to send data to B

Source host A→destination E:

Host A:[A's addr & subnet mask] = [E's addr & subnet mask] ?

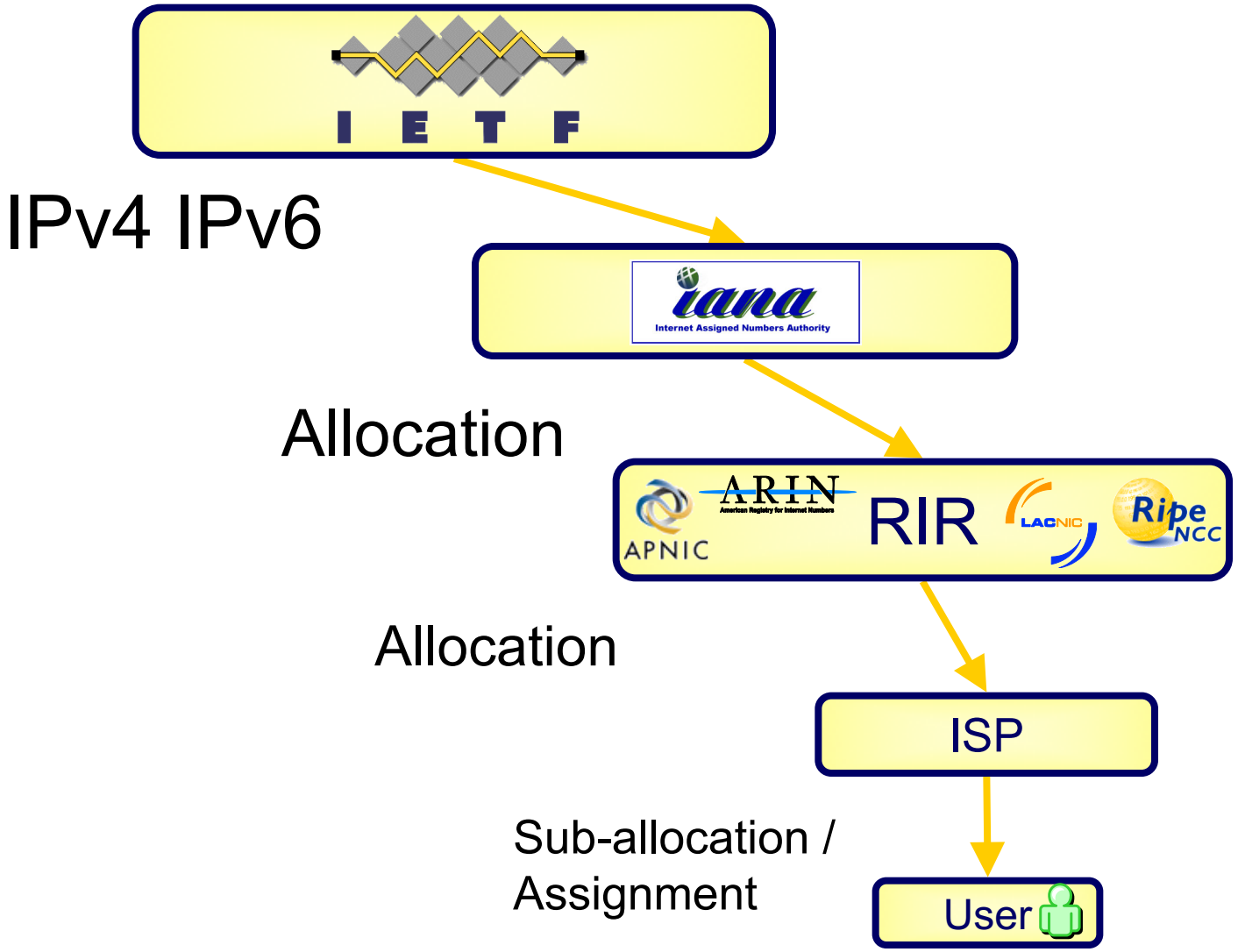
No: Send pkt to default router 173.1.1.4

Router: forward packets to next hop according to its forwarding table



Address management today

FYI



Address management today

FYI

IP addresses Info

FYI

- How many IPv4 addresses we have total?
  - How many can be used by end hosts
    - 224.0.0.0/4 used only for multicast
    - 240.0.0.0/4 reserved for “future” use

- Q: How many IPv4 addresses does ICANN still have?

NEWS

Update: ICANN assigns its last IPv4 addresses

By Stephen Lawson FOLLOW

IDG News Service | Feb 3, 2011 3:01 PM PT

RELATED TOPICS

Internet

Mobile & Wireless

Networking

The Internet Assigned Numbers Authority (IANA) has handed out its last IPv4 addresses, leaving the remaining blocks to regional registries that in some cases may exhaust them within a few months.

MORE LIKE TI

The 6 biggest misconception

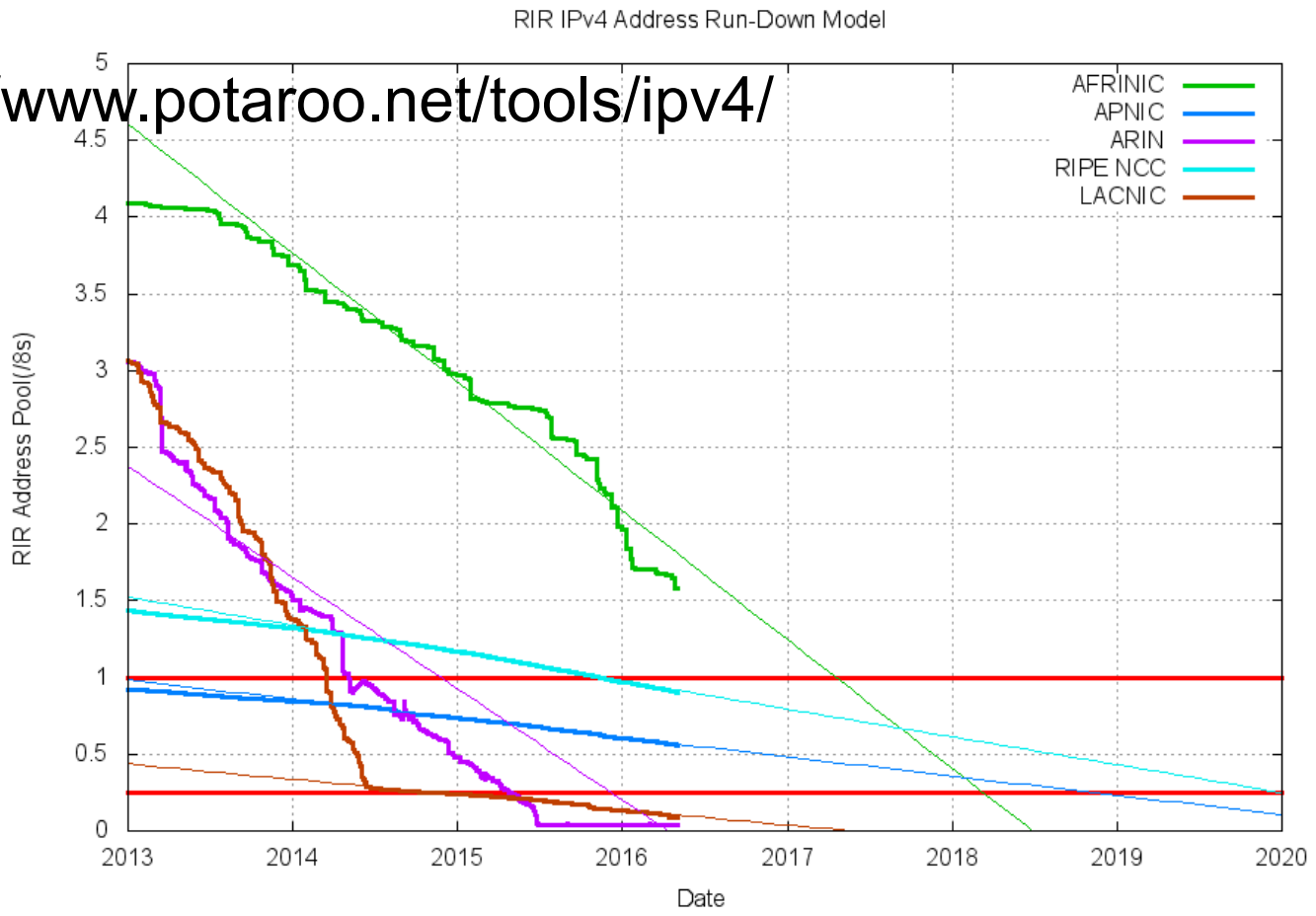
Internet body may use up IPv week

Address allocation kicks off II

Q&A

- Why the world is not exploding, given ICANN don't have IPv4 addresses anymore?

http://www.potaroo.net/tools/ipv4/



Network Address Translation (NAT)

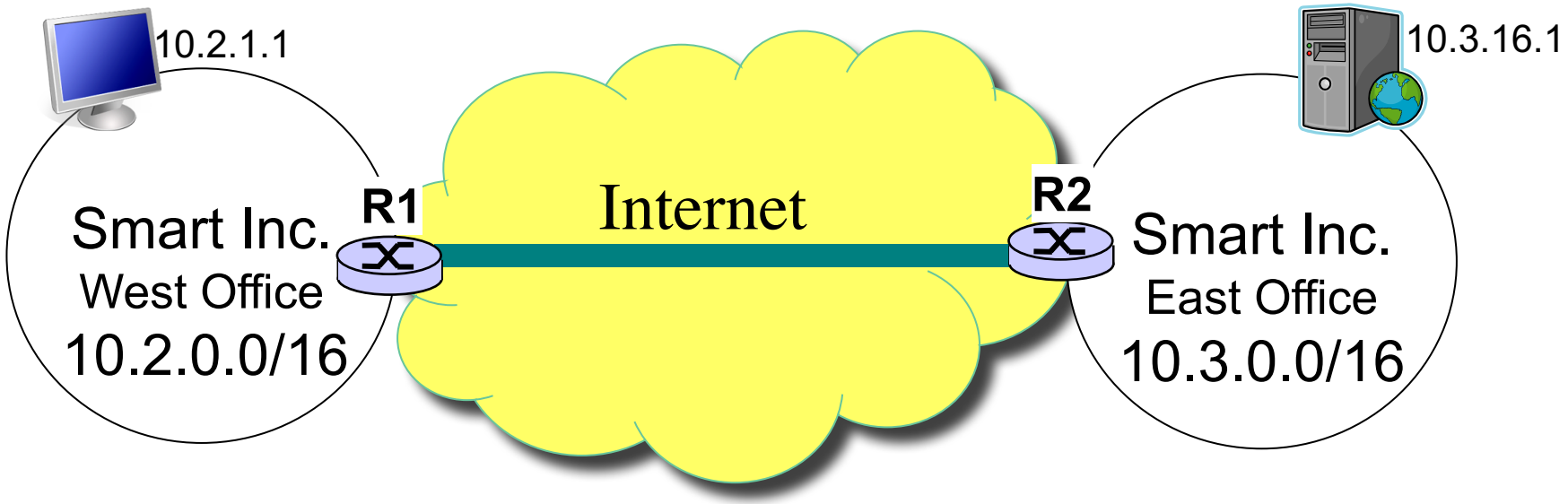
- A “short-term” solution to the problem of the depletion of IP addresses
  - Long-term solution is IPv6

- NAT is a way to conserve IP addresses
  - Can be used to hide a number of hosts behind a single IP address
  - Uses private addresses:
    - 10.0.0.0-10.255.255.255,
    - 172.16.0.0-172.32.255.255 or
    - 192.168.0.0-192.168.255.255

- Today is also used as a “security” measure
  - Private addresses cannot be reached from outside, unless communication initiated from the inside

Private Networks

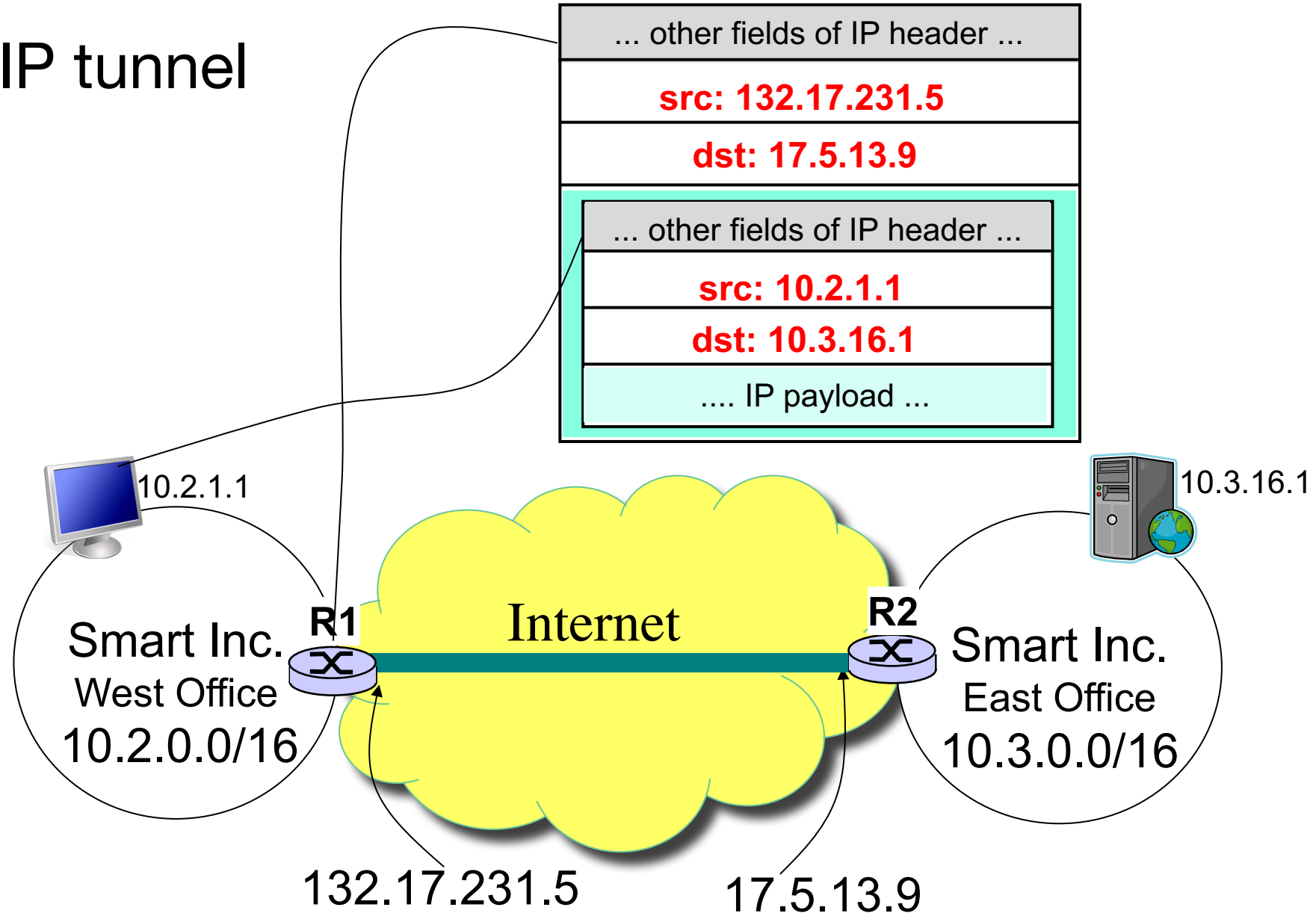
- 10.0.0.0-10.255.255.255,
- 172.16.0.0-172.32.255.255 or
- 192.168.0.0-192.168.255.255



Connecting private nets via IP tunneling

important

- IP tunnel



Explanation of the previous slide

- Router R1 is configured to put all packets with destination address under 10.3.0.0/16 into the tunnel to R2, i.e.
  - Putting an outer IP header with source address=132.17.231.5, dest. addr=17.5.13.9
- Similarly, R2 is configured to put all packets with destination address under 10.2.0.0/16 into the tunnel to R1
- When each encapsulated packet reaches R2, R2 takes off the outer IP header, then forwards the packet to its real destination
  - R1 does the same
- IP tunneling is a general solution to build a virtual wire across the global Internet

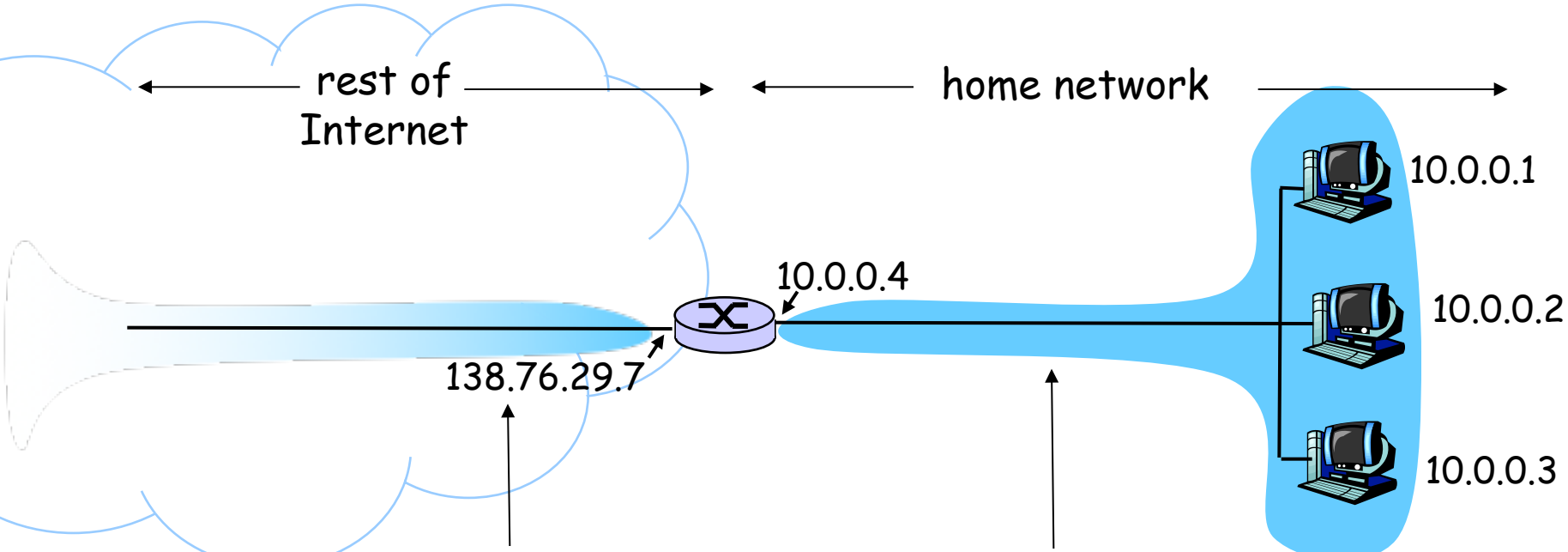
Network Address Translation (NAT)

- A “short-term” solution to the problem of the depletion of IP addresses
  - Long-term solution is IPv6

- NAT is a way to conserve IP addresses
  - Can be used to hide a number of hosts behind a single IP address
  - Uses private addresses:
    - 10.0.0.0-10.255.255.255,
    - 172.16.0.0-172.32.255.255 or
    - 192.168.0.0-192.168.255.255

- Today is also used as a “security” measure
  - Private addresses cannot be reached from outside, unless communication initiated from the inside

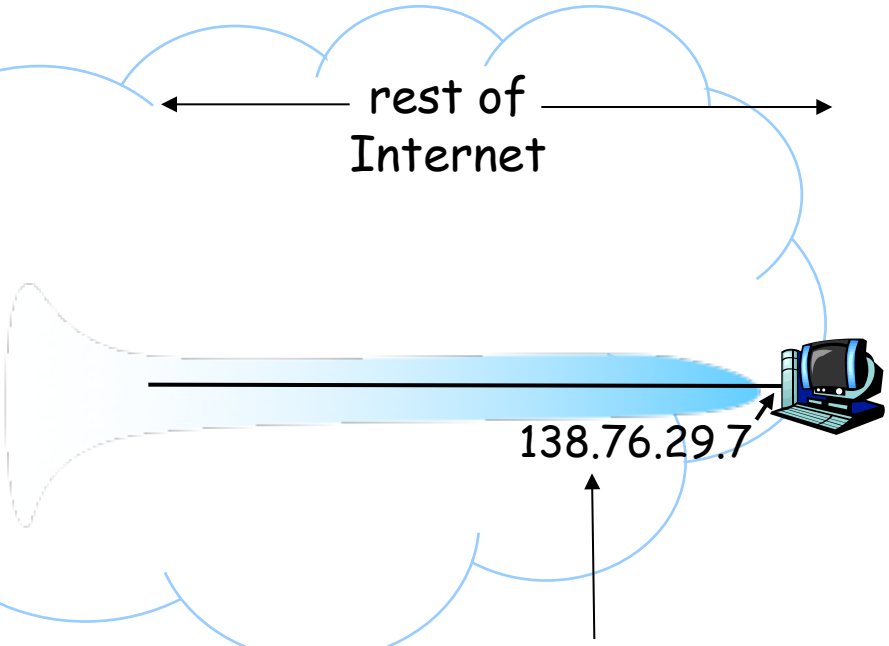
Using Private Addresses at Home



All packets *leaving* this box must have the same source IP address 138.76.29.7

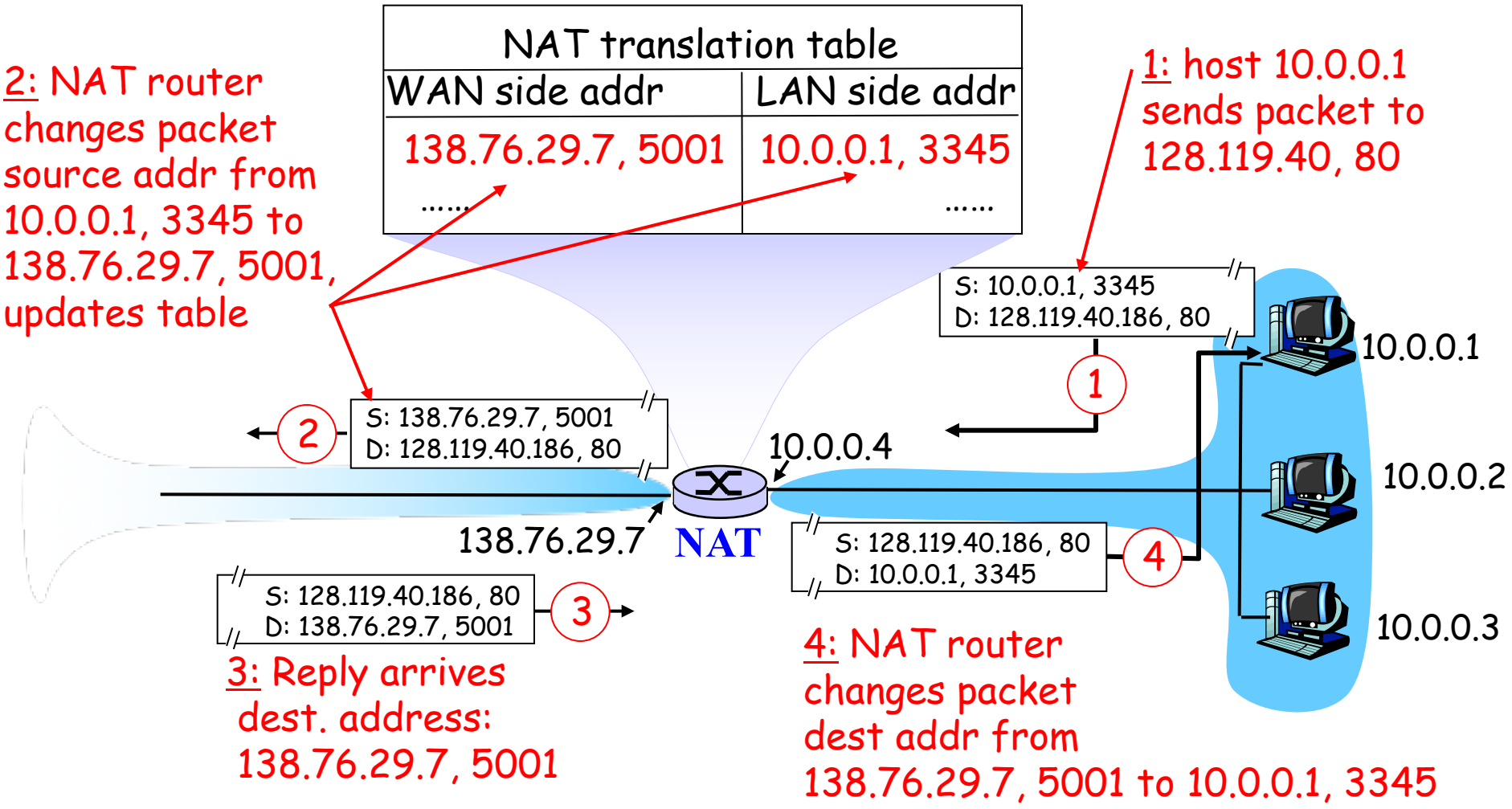
packets with source or destination in this network have 10.0.0.0/8 address for source, destination (as usual)

NAT: Network Address Translation



All packets *leaving* this box must have the same source IP address 138.76.29.7

NAT: Network Address Translation



Would the NAT table overflow after running for a long time?

NAT implementation

NAT router must do the following:

- *outgoing* packets: replace (*source* IP address, source port #) of every outgoing packet to (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* packets: replace (destination NAT IP address, destination port #) of every incoming packet with corresponding (source IP address, port #) stored in NAT table

- Delete NAT table entries that have not been used for some time

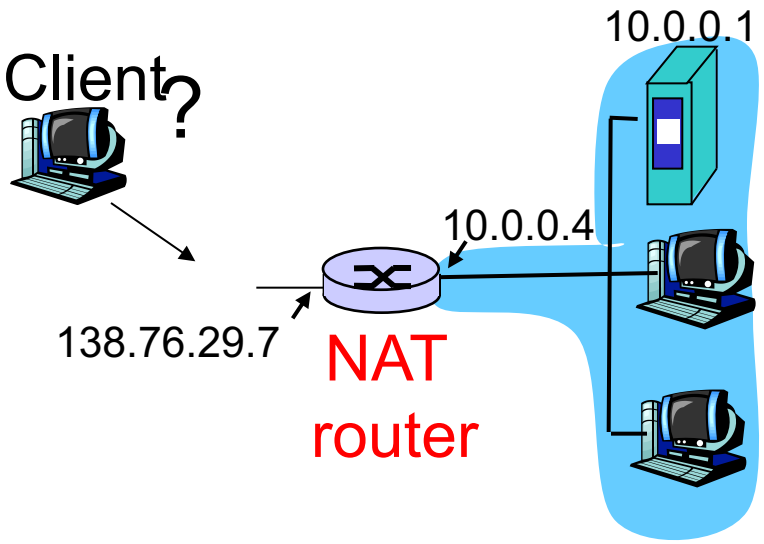
NAT Limitations and Problems

- Are there any?

Problems due to NAT

- Increased complexity (e.g. router has to keep the NAT table)
- Single point of failure
- Cannot run services inside a NAT box
  - All application designs have to worry about NAT traversal problem

- client wants to connect to the server with address 10.0.0.1
- server address 10.0.0.1 is only visible within the LAN
- The only externally visible address for all internal hosts: 138.76.29.7



NAT Traversal: from outside to inside?

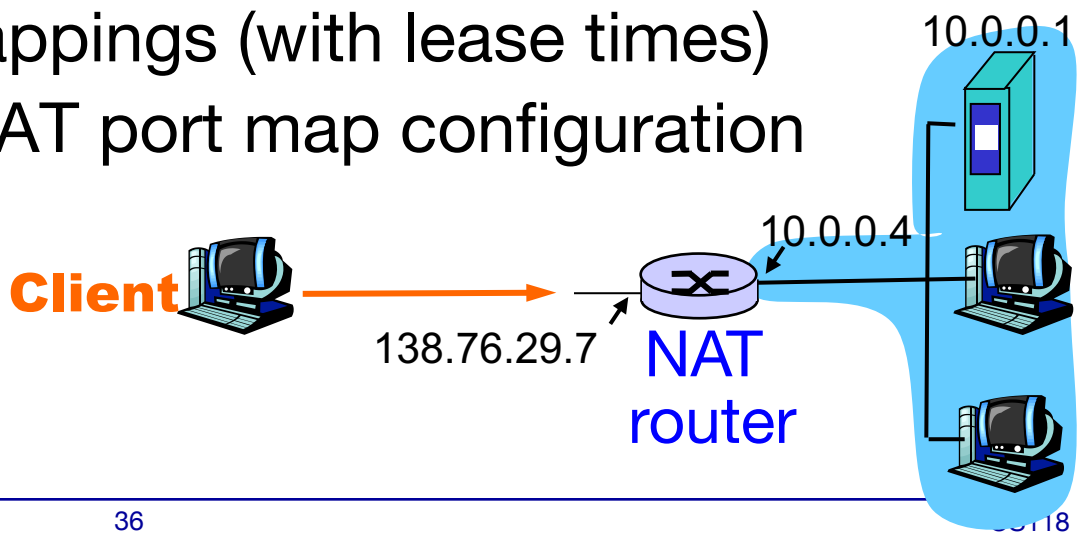
Solution 1: statically configure NAT to forward incoming connection requests at a given port to server, e.g.

- (138.76.29.7, port 2500) always forwarded to 10.0.0.1 port 2500

Solution 2: Universal Plug and Play (UPnP) Protocol: Allows host behind a NAT to:

- learn the public IP address (138.76.29.7)
- add/remove port mappings (with lease times)

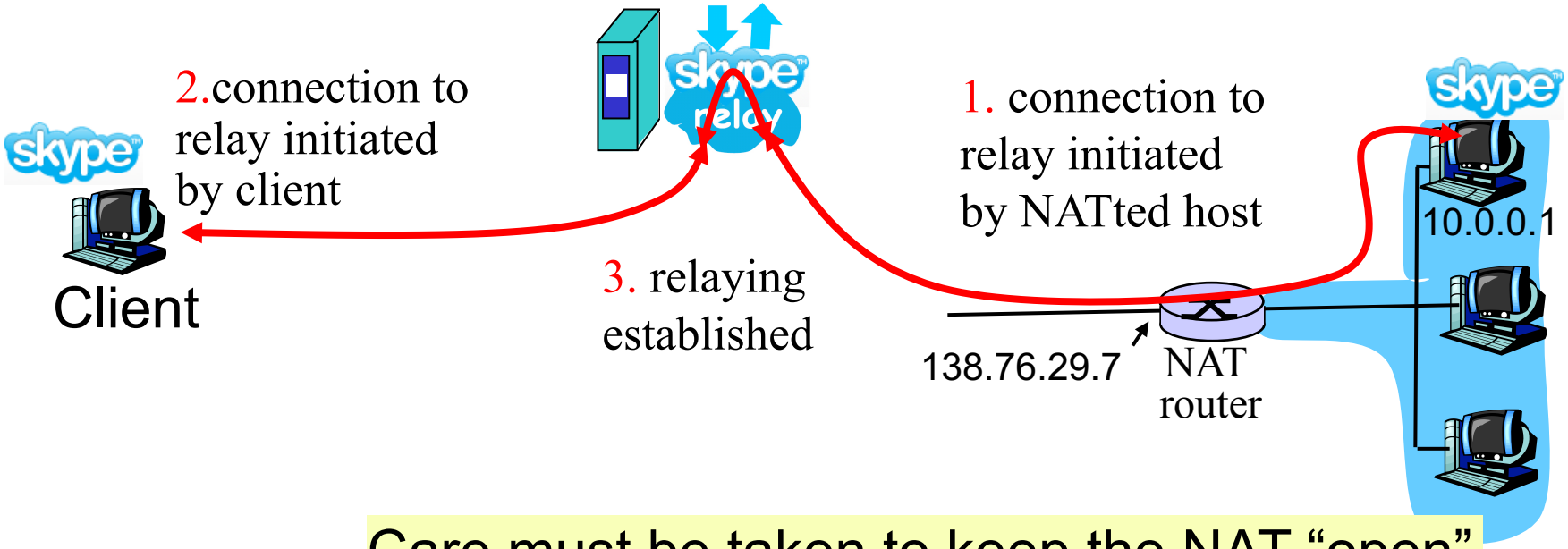
i.e., automate static NAT port map configuration



NAT Traversal

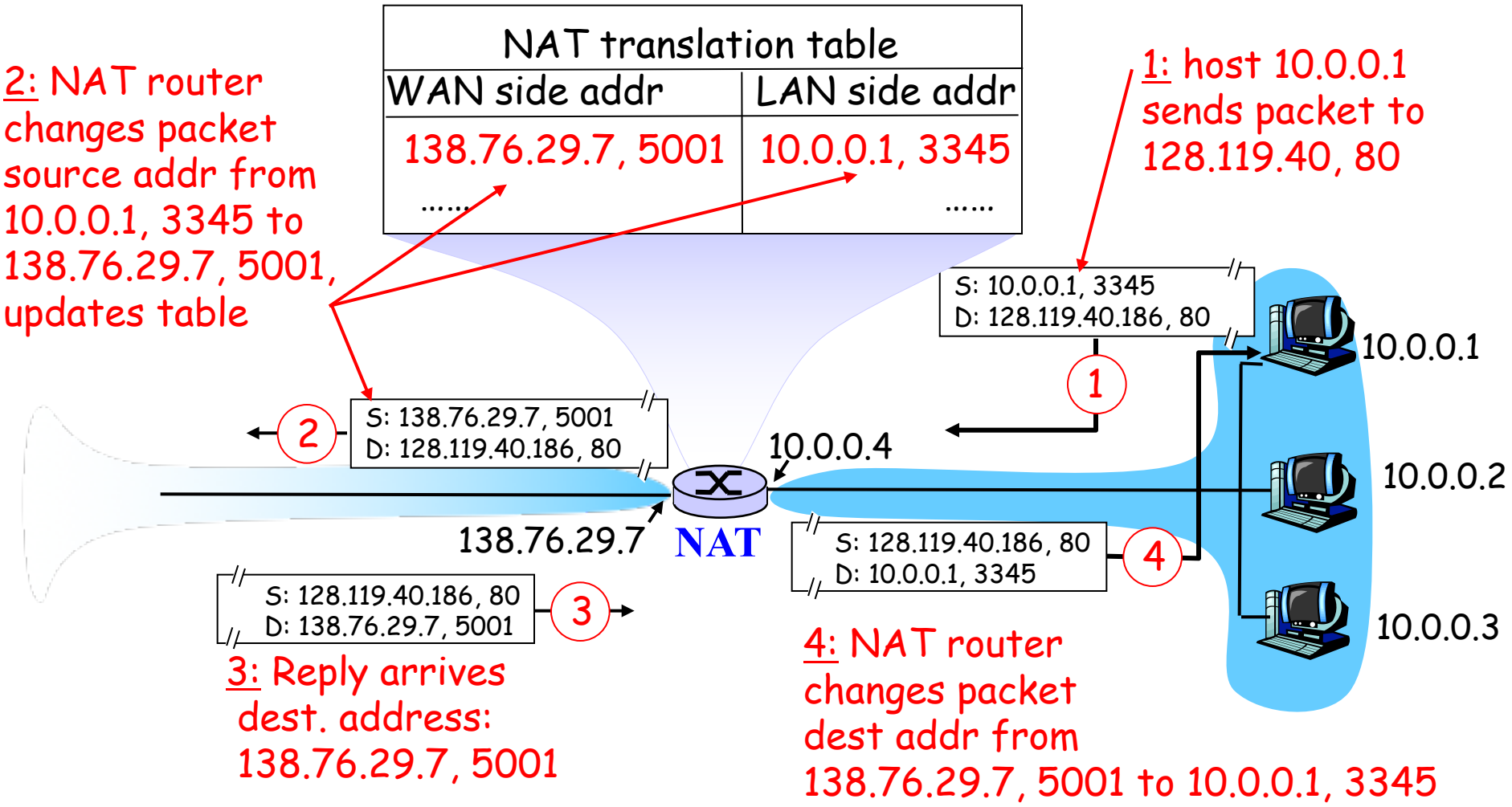
Solution 3: relaying (used in Skype)

- NATed client establishes connection to relay
- External client connects to relay
- relay bridges packets between to connections



Care must be taken to keep the NAT "open"

NAT: Network Address Translation

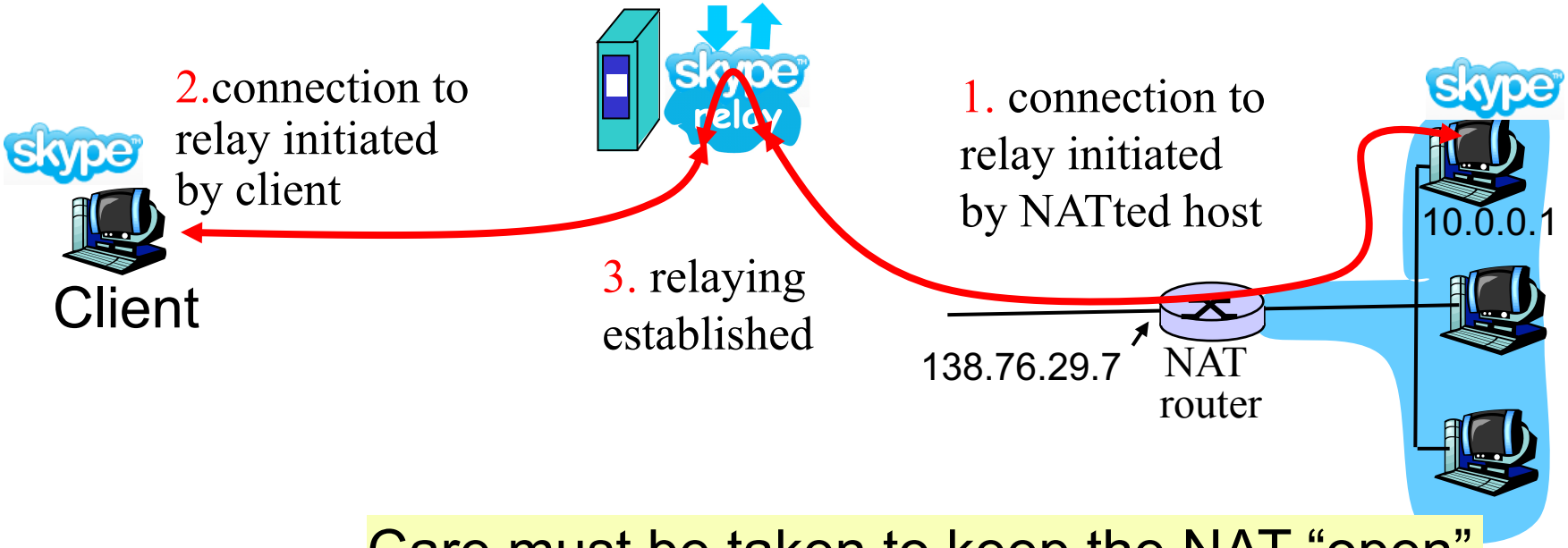


Would the NAT table overflow after running for a long time?

NAT Traversal

Solution 3: relaying (used in Skype)

- NATed client establishes connection to relay
- External client connects to relay
- relay bridges packets between to connections



Care must be taken to keep the NAT "open"

How your computer gets an IP address

important

- Sometimes: Hard-coded by system admin in a file
  - Windows: control-panel->network->configuration->tcp/ip->properties
  - UNIX: /etc/rc.config
- Most of the time: DHCP, Dynamic Host Configuration Protocol
  - dynamically alocates address (and other necessary info) to a host
    - IP address for the host
    - IP address for default router
    - Subnet mask
    - IP address for DNS caching resolver
  - Allows address reuse

IP addresses: how to get one?

FYI

Q: How does the dept DHCP server get addresses?

A: configured by the dept network staff

Q: How does an ISP/site get block of IP addresses?

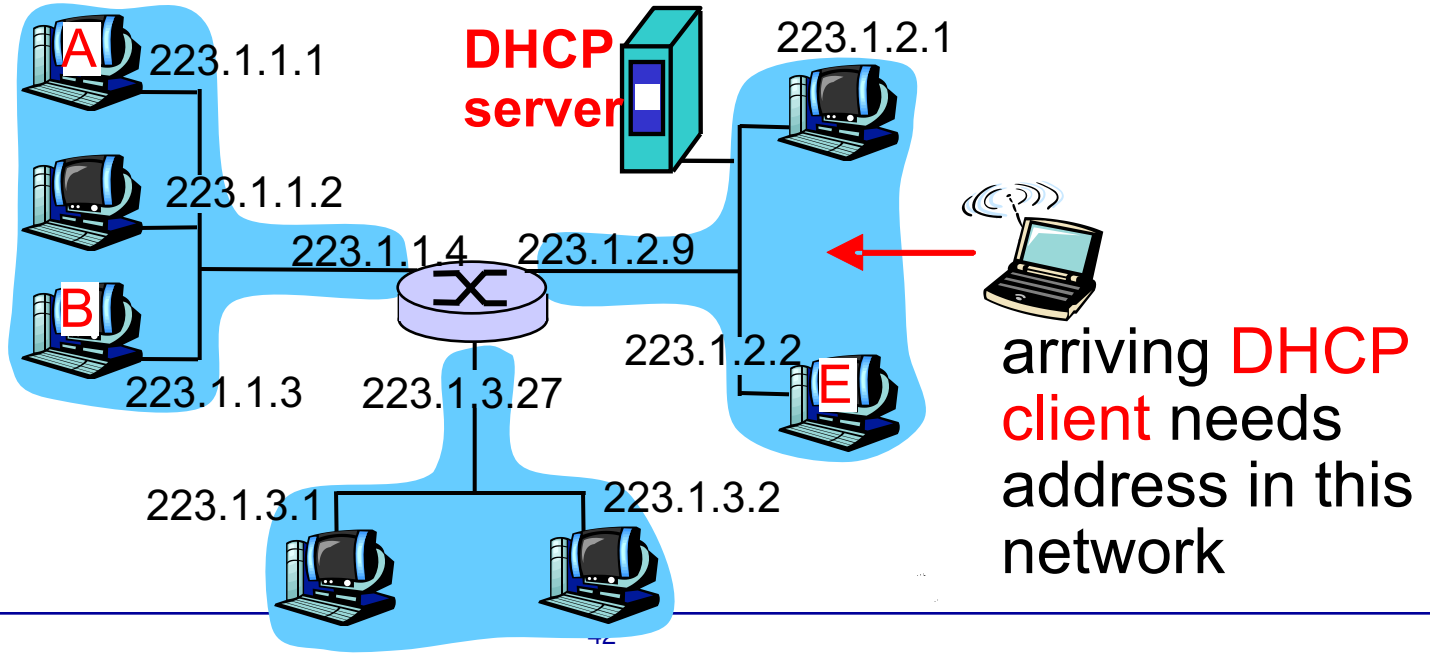
A: request from ICANN: Internet Corporation for Assigned Names and Numbers

- allocates addresses to RIRs (Regional Internet Registries)
- manages DNS
- assigns domain names, resolves disputes

FYI

DHCP Overveiw

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



DHCP client-server scenario

FYI