IP (Internet Protocol)

Lots of slides taken from

From Liebeherr / El-Zarki Lecture Notes

Addressing

Host Addressing

- Why do we need host addressing?
  - If we had no network (no communication) addressing is not needed
  - If we had two machines, the number of addresses needed is 2. The minimum number of bits per address is 1.
  - If we had n machines, we need n addresses and the minimum number of bits required for addressing $log_2(n)$

- There are two types of addresses:
  - Logical addresses: e.g. IP addresses, MSISDN (phone n), etc.
  - Physical addresses: e.g. MAC addresses, IMSI, etc.
  - Why are there two types of addresses? Would not only build networks on top of physical addresses?

Application Identification

- Why do we need application identification
  - Addressing a host is not sufficient. We need to know which application are we addressing: is it Skype, Messenger, a web browser, etc.
  - If we had one application, we would not need application identification

- Network applications are conventionally identified by their port numbers.

IPv4 addressing revisited

- CIDR (Classless Inter Domain Routing)
  - Replaces class-based addressing
  - No need for mask
  - e.g. 192.168.0.1/24 = (192.168.0.1, 255.255.255.0)

- Subnetting

- Reachability at the IP layer
  - e.g. 192.168.0.1/24 and 192.168.1.1/24 are not reachable at the IP layer even if they are physically connected

- Quiz: pinging the link layer?

IP Addresses

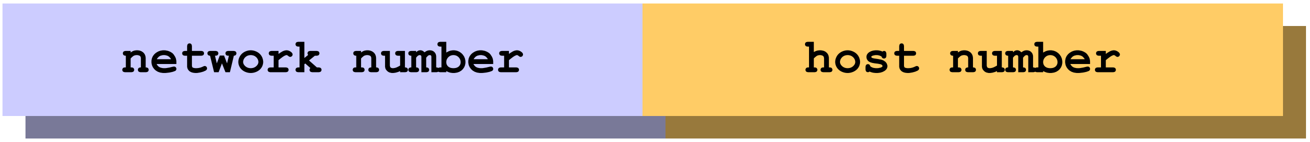
An IP address is a unique global address (at a given time) for a network interface

Exceptions:

IP addresses in private networks (See NAT)

An IP address:

- is a 32 bit long identifier
- encodes a network number and a host number

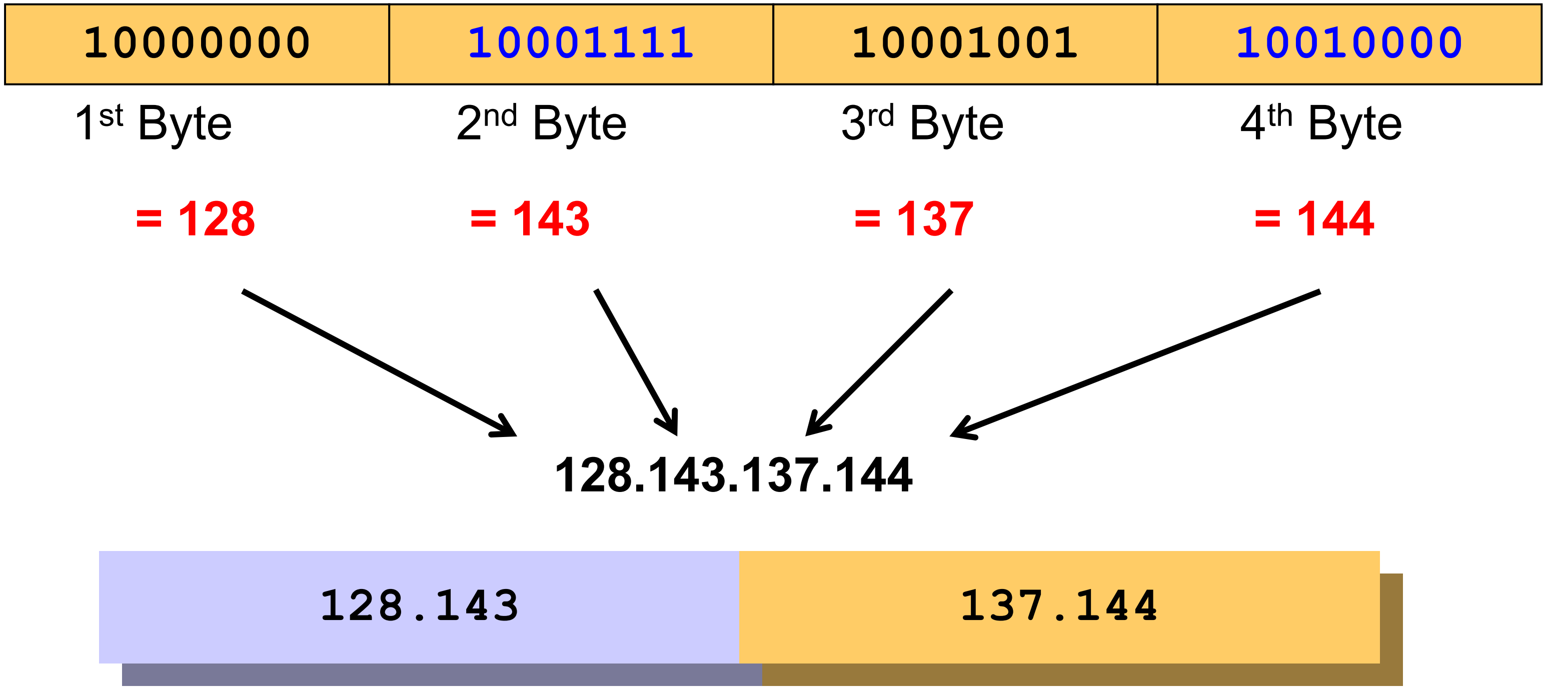


network number

host number

- How long is the network number (prefix) --> Nework Mask

IP Addresses (example)



10000000	10001111	10001001	10010000
1st Byte	2nd Byte	3rd Byte	4th Byte
= 128	= 143	= 137	= 144

128.143.137.144

128.143 137.144

Network address is 128.143.0.0 (or 128.143)

Host address is 137.144

Network Mask is 255.255.0.0

CIDR notation 128.143.137.144/16

Special IP Addresses

Reserved or (by convention) special addresses:

Loopback interfaces

- From127.0.0.1 to 127.255.255.255 are reserved for loopback interfaces
- Most systems use 127.0.0.1 as loopback address
- loopback interface is associated with name “localhost”

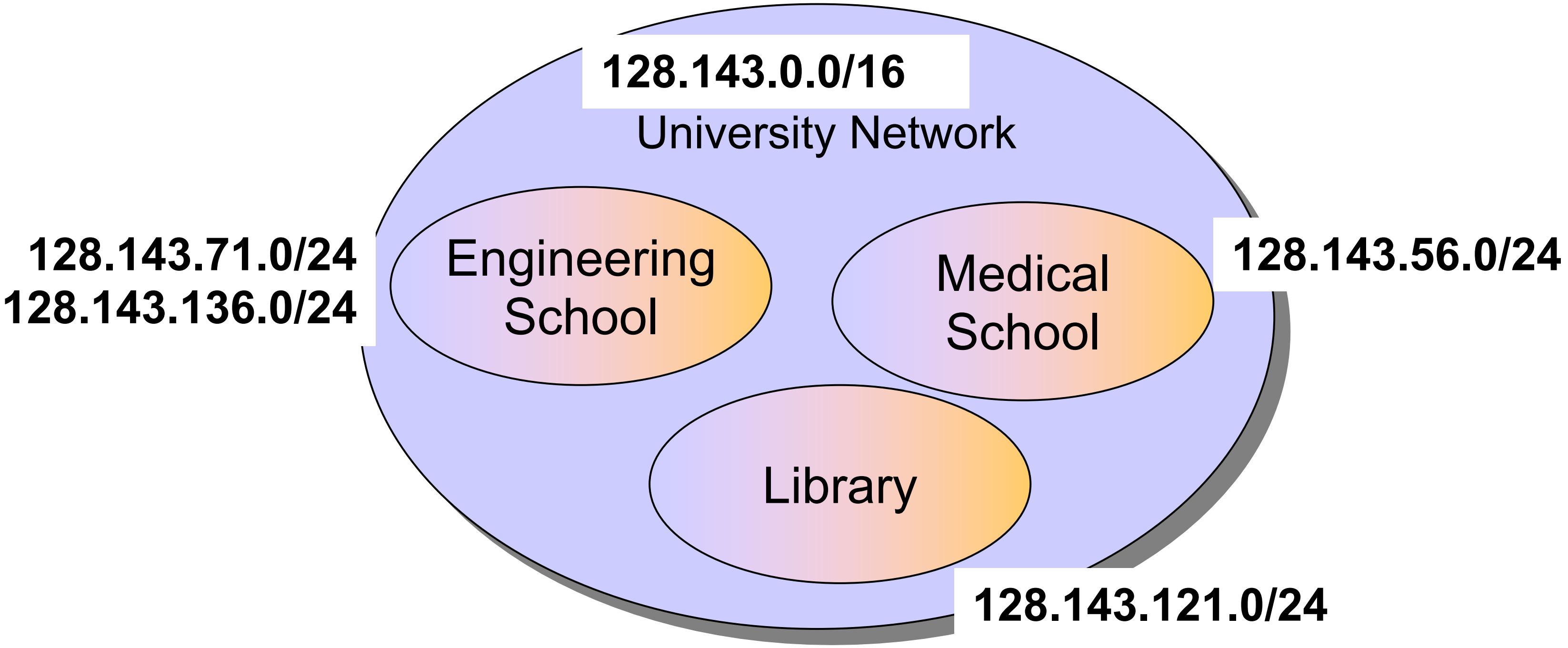
Network address

- Host number is set to all zeros, e.g., 128.143.**0.0/16**

Broadcast address

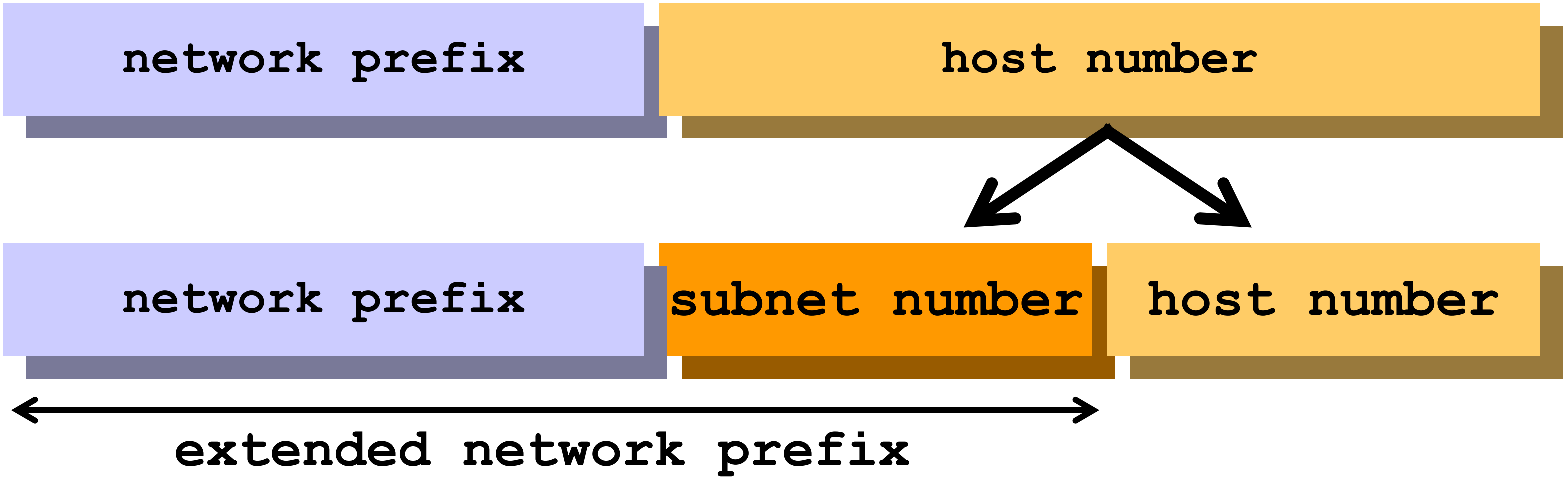
- Host number is all ones, e.g., 128.143.**255.255**
- Broadcast goes to all hosts on the network
- Often ignored due to security concerns

Address Assignement with subnetting



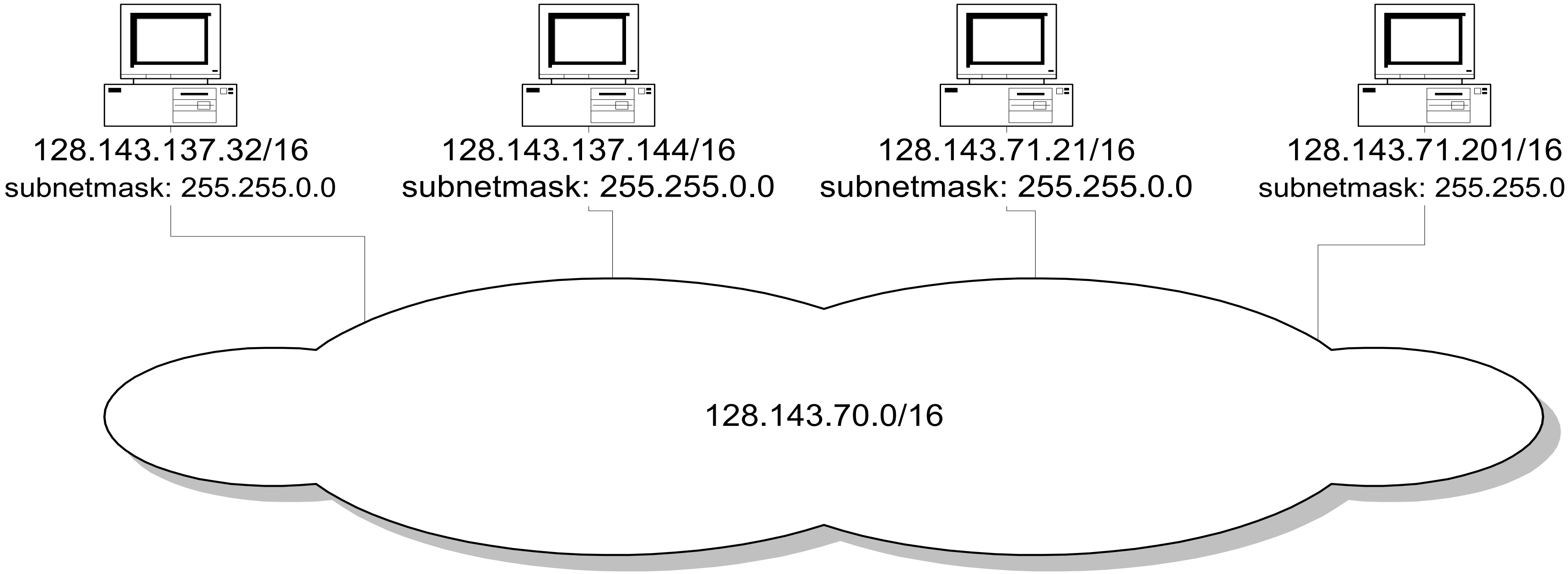
Addresses in each subnet can be administered locally
Facilitates management 128.143.0.0/16 is the entire university

Basic Idea of Subnetting



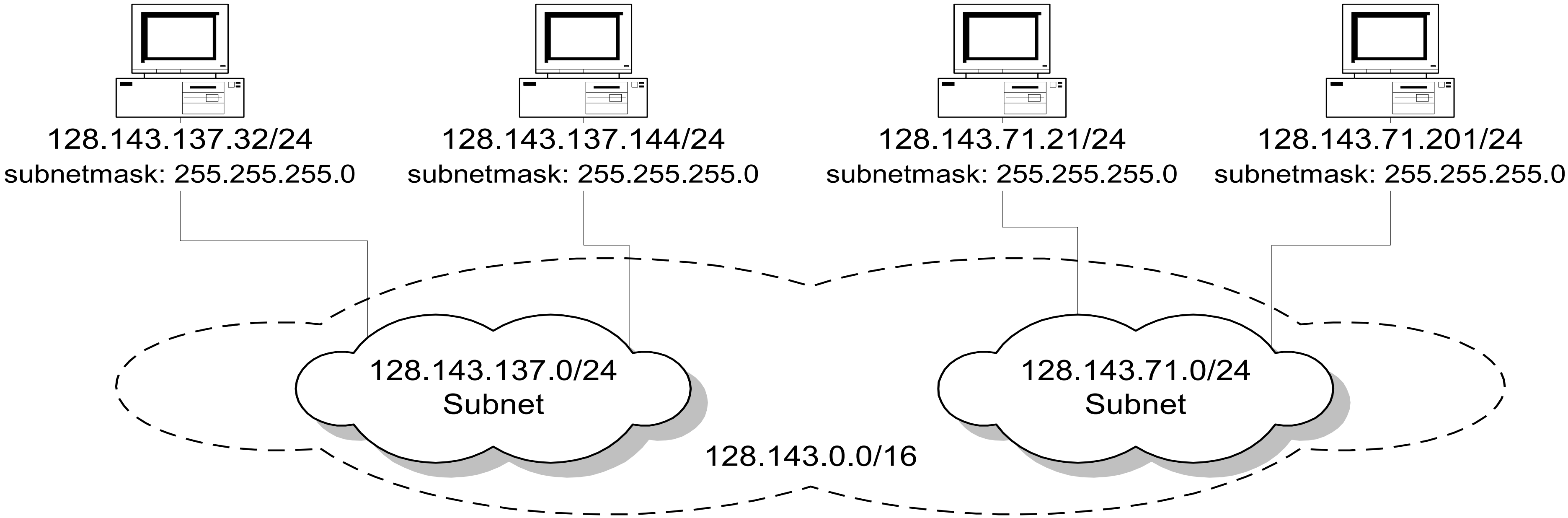
- Subnets can be freely assigned within the organization
- Internally, subnets are treated as separate networks
- Subnet structure is not visible outside the organization
- Length of the subnet needs to identical at all subnets
- Subnets reduce complexity at external routers.

Without Subnetting



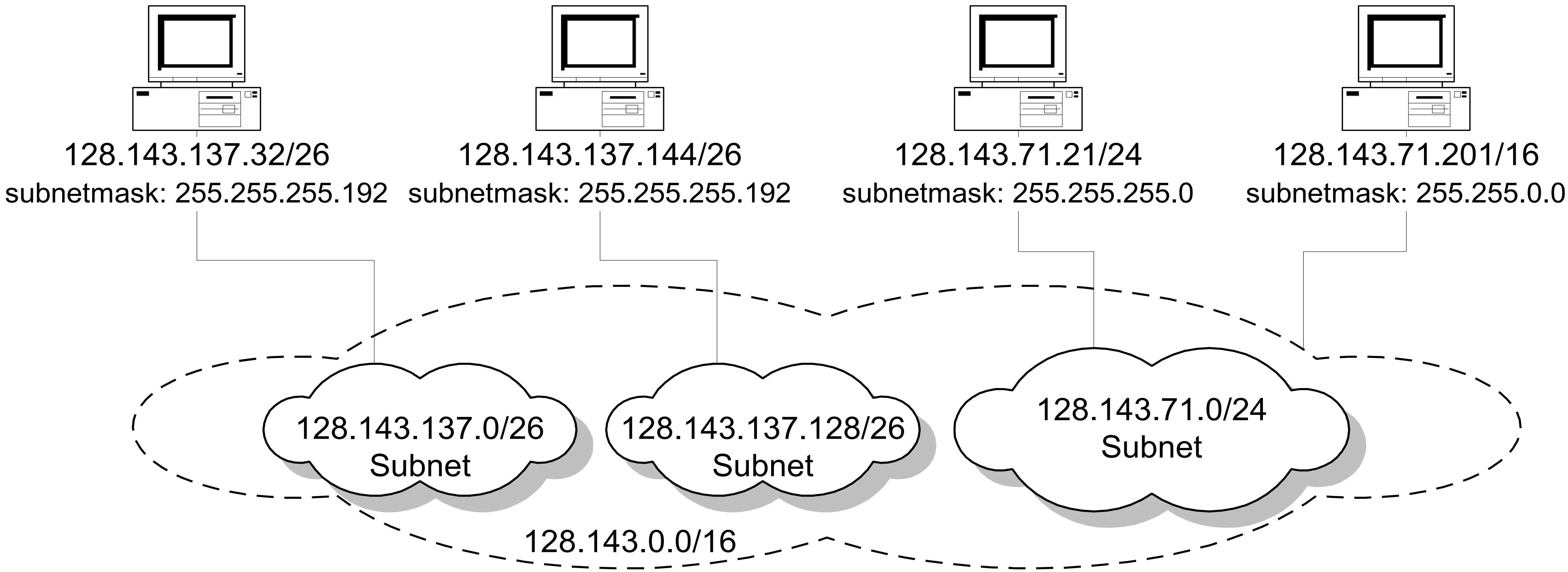
All hosts think that the other hosts are on the same network

With Subnetting



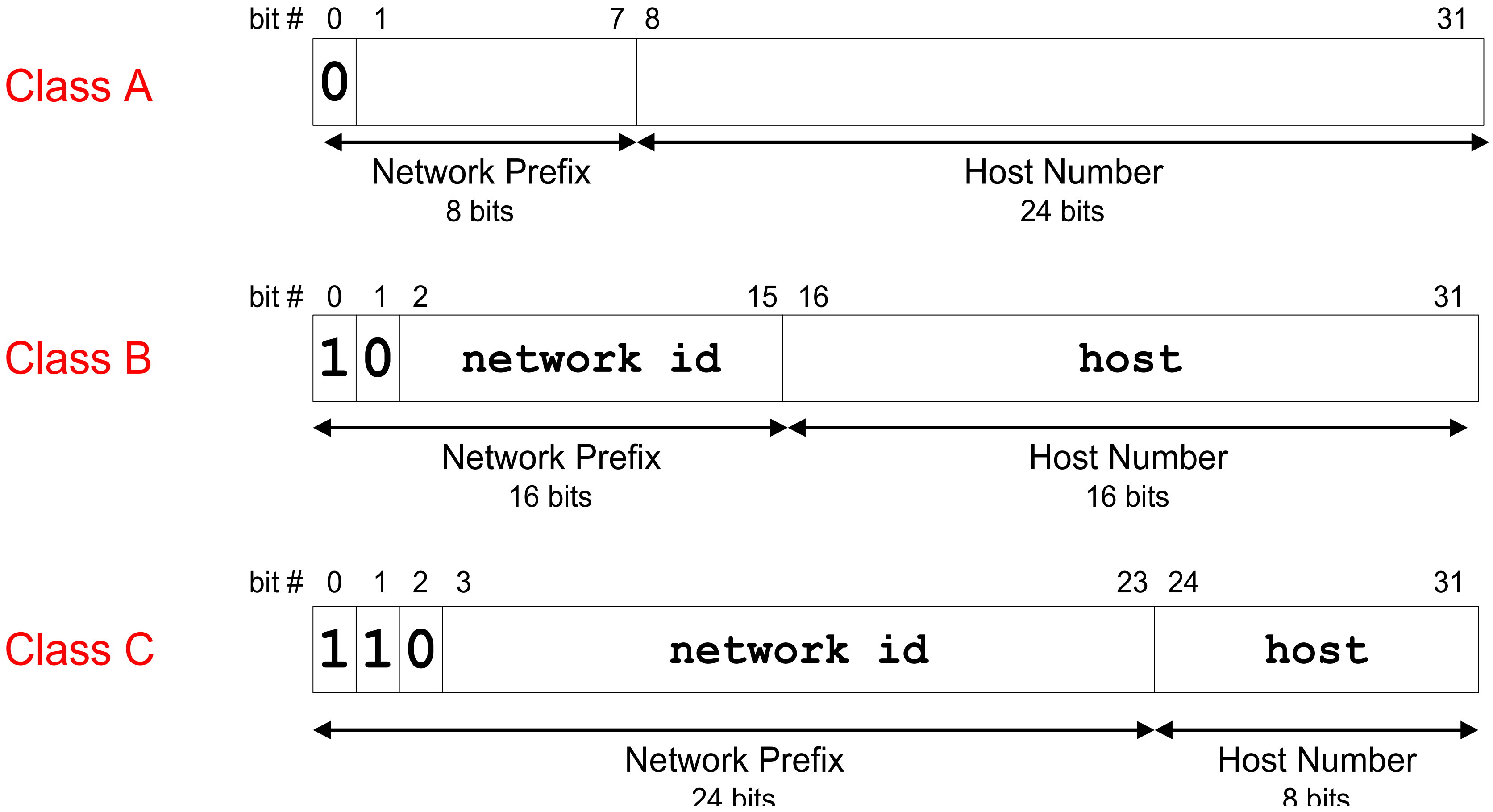
Hosts with same extended network prefix belong to the same network

With Subnetting



Different subnet-masks lead to different views of the size of the scope of the network

Internet Address Classes (Deprecated)



Class A

bit # 0 1 7 8 31

0		

Network Prefix
8 bits

Host Number
24 bits

Class B

bit # 0 1 2 15 16 31

1	0	network id	host

Network Prefix
16 bits

Host Number
16 bits

Class C

bit # 0 1 2 3 23 24 31

1	1	0	network id	host

Network Prefix
24 bits

Host Number
8 bits

Internet Address Classes (Deprecated)

	bit #	0	1	2	3	4	31
Class D		1	1	1	0	multicast group id	

	bit #	0	1	2	3	4	5	31
Class E		1	1	1	1	0	(reserved for future use)	

Problems with Classful Addressing

- Routing tables on the backbone Internet need to have an entry for each network address. When Class C networks were widely used, this created a problem. By the 1993, the size of the routing tables started to outgrow the capacity of routers.

- Too few network addresses for large networks. Class A and Class B addresses were gone.

- Limited flexibility for network addresses:
    - Class A and B addresses are overkill (>64,000 addresses)
    - Class C address is insufficient

CIDR Classless Inter Domain Routing

- IP backbone routers have one routing table entry for each network address

This is acceptable for Class A and Class B networks
2^7 = 128 Class A networks

2^{14} = 16,384 Class B networks

But this is not acceptable for Class C networks (routing table too large)

2^{21} = 2,097,152 Class C networks

- In 1993, the size of the routing tables started to outgrow the capacity of routers

→ The Class-based assignment of IP addresses had to be abandoned

CIDR Classless Inter Domain Routing

CIDR notation of an IP address:

192.0.2.0/18

"18" is the prefix length. It states that the first 18 bits are the network prefix of the address (and 14 bits are available for specific host addresses)

CIDR notation can replace the use of subnetmasks

IP address 128.143.137.144 and subnetmask 255.255.255.0 becomes 128.143.137.144/24

CIDR notation allows to drop trailing zeros of network addresses:

192.0.2.0/18 can be written as 192.0.2/18

CIDR Address Assignment

Backbone ISPs obtain large block of IP addresses space and then reallocate portions of their address blocks to their customers.

Example:

- Assume that an ISP owns the address block 206.0.64.0/18, which represents 16,384 (2^{14}) IP addresses

- Suppose a client requires 800 host addresses

- With classful addresses: need to assign a class B address (and waste ~64,700 addresses) or four individual Class Cs (and introducing 4 new routes into the global Internet routing tables)

- With CIDR: Assign a /22 block, e.g., 206.0.68.0/22, and allocate a block of 1,024 (2^{10}) IP addresses.

CIDR Address Assignment

Aggregation of routing table entries:

128.142.0.0/16 and 128.143.0.0/16 are represented as 128.142.0.0/15

Longest prefix match: Routing table lookup finds the routing entry that matches the longest prefix

What is the outgoing interface for

128.143.137.0/24 ?

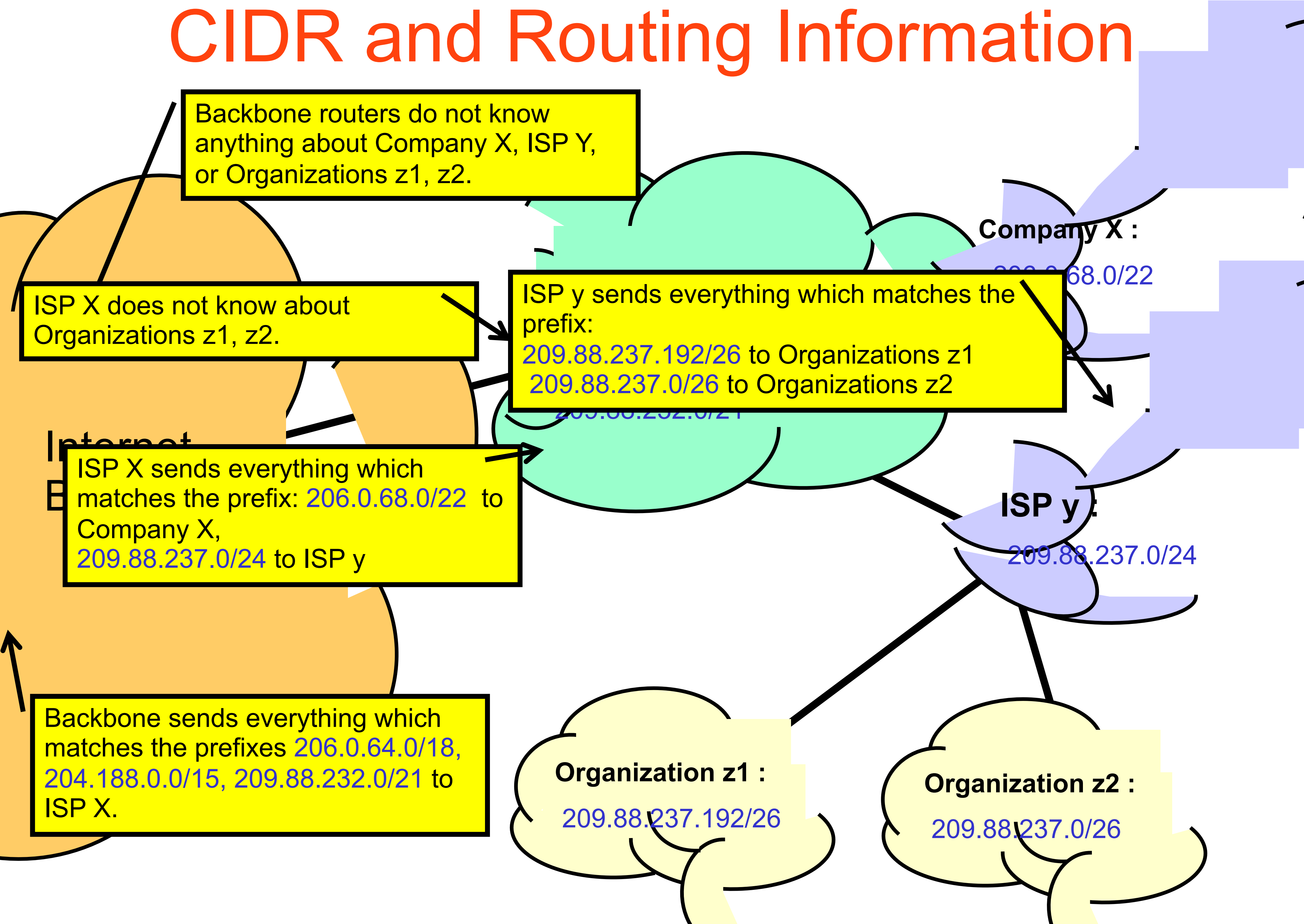
Prefix	Interface
128.0.0.0/4	interface #5
128.128.0.0/9	interface #2
128.143.128.0/17	interface #1

Route aggregation can be exploited

when IP address blocks are assigned

in an hierarchical fashion

CIDR and Routing Information



Backbone routers do not know anything about Company X, ISP Y, or Organizations z1, z2.

ISP X does not know about Organizations z1, z2.

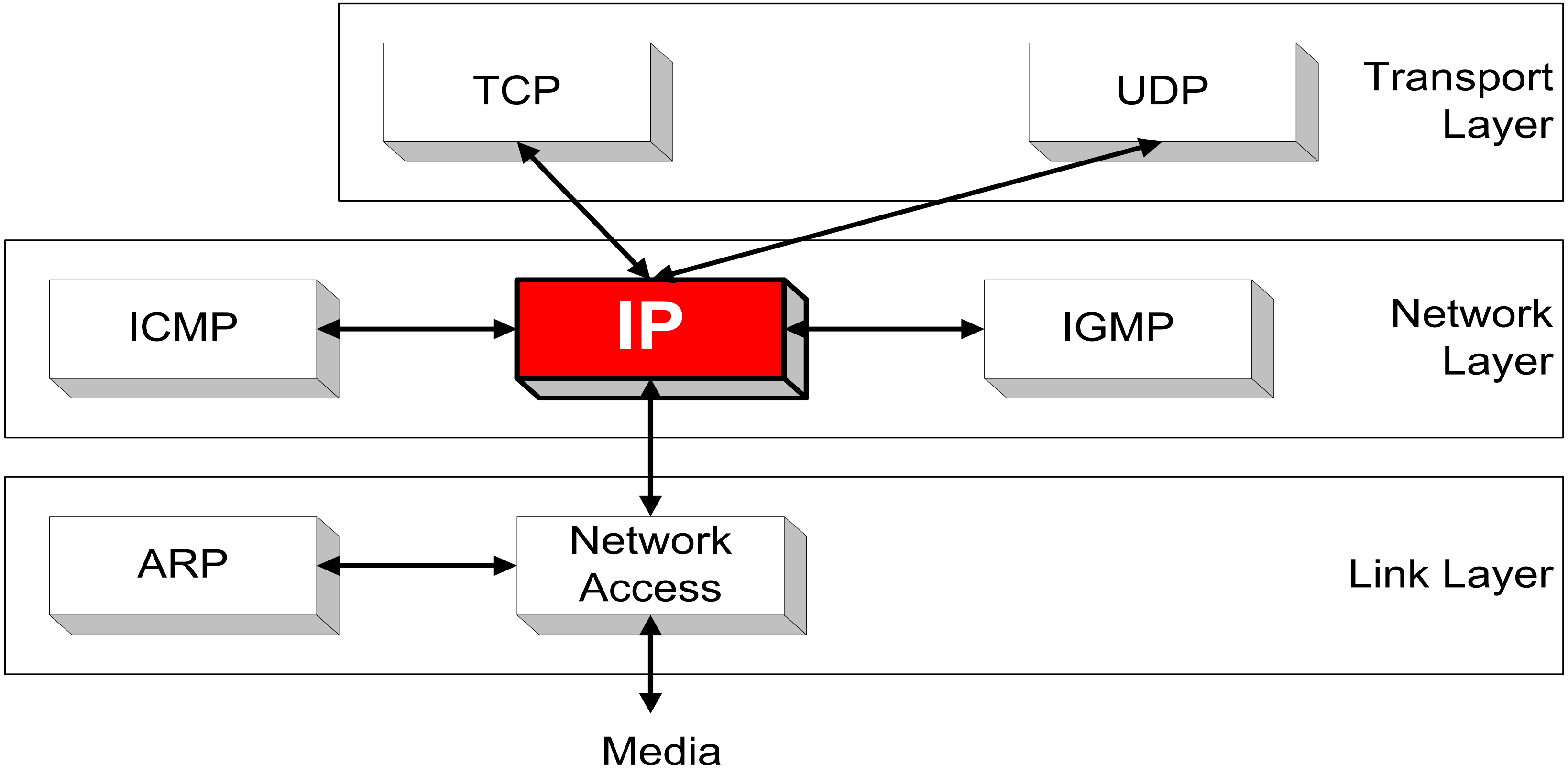
ISP y sends everything which matches the prefix:
209.88.237.192/26 to Organizations z1
209.88.237.0/26 to Organizations z2

ISP X sends everything which matches the prefix: 206.0.68.0/22 to Company X,
209.88.237.0/24 to ISP y

Backbone sends everything which matches the prefixes 206.0.64.0/18, 204.188.0.0/15, 209.88.232.0/21 to ISP X.

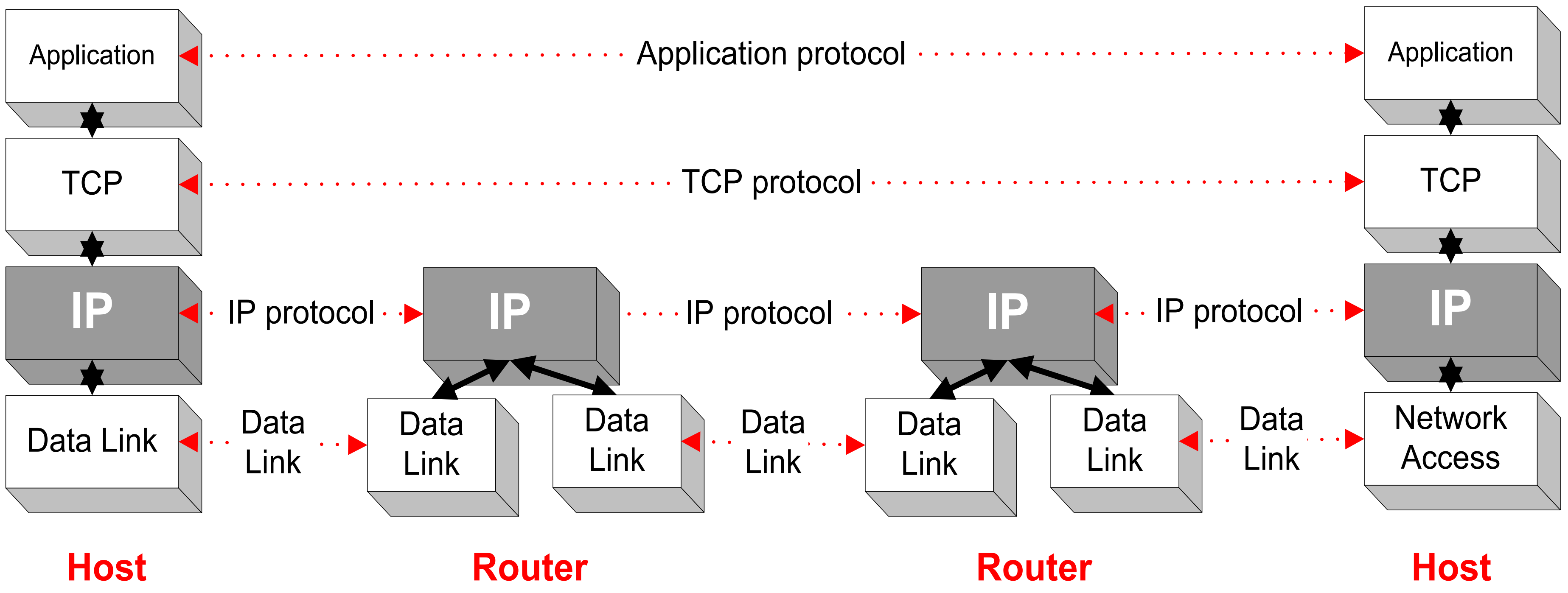
Architecture

IP



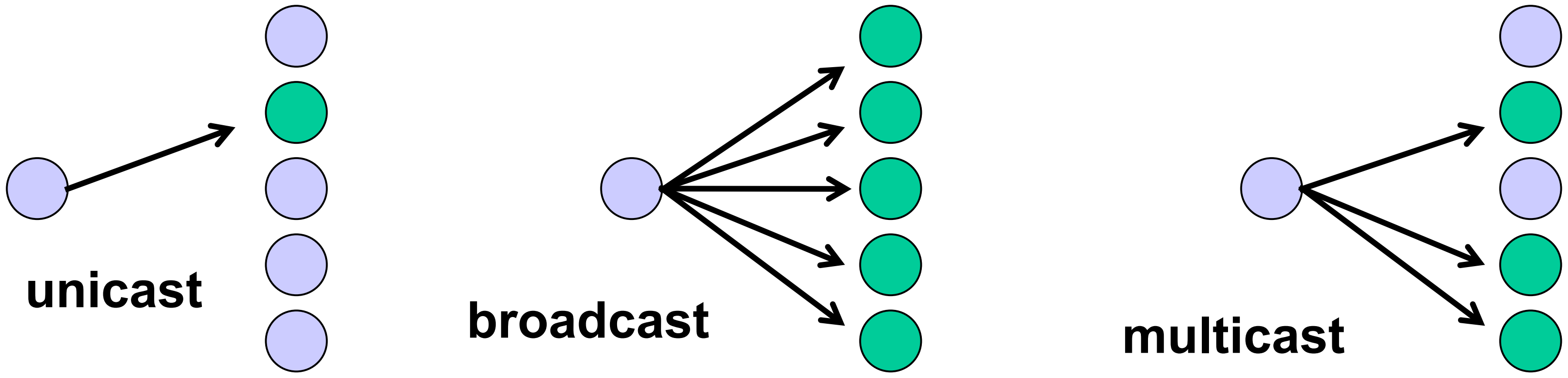
IP is a Network Layer Protocol

IP in TCP/IP Architecture



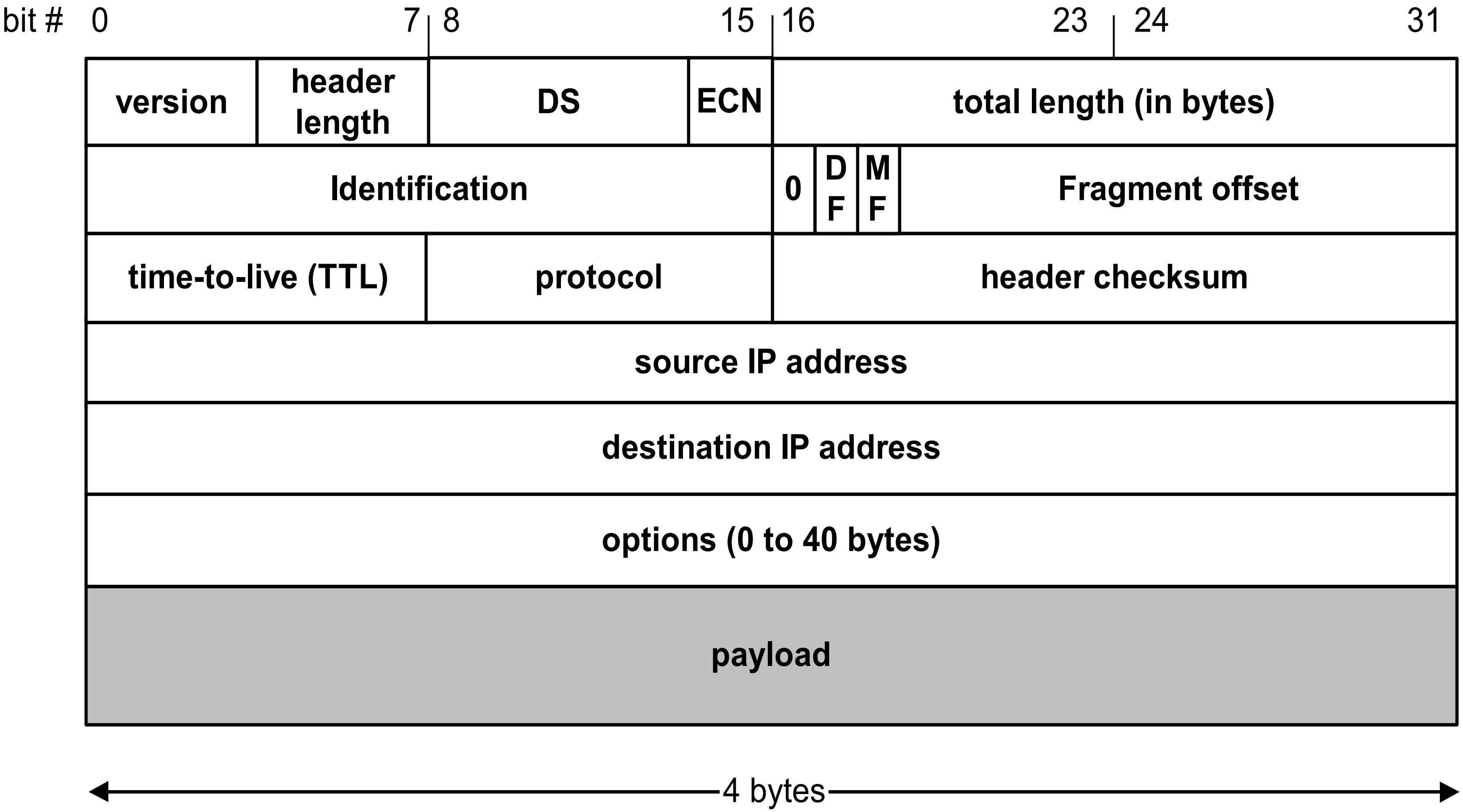
IP is the highest layer protocol which is implemented at both routers and hosts

IP in TCP/IP Architecture



- IP multicast also supports a many-to-many service.
- IP multicast requires support of other protocols (IGMP, multicast routing)

IP Packet Format



bit # 0 7 8 15 16 23 24 31

version | header length | DS | ECN | total length (in bytes)

Identification | 0 | D F | M F | Fragment offset

time-to-live (TTL) | protocol | header checksum

source IP address

destination IP address

options (0 to 40 bytes)

payload

4 bytes

20 bytes ≤ Header Size < 2^4 x 4 bytes = 60 bytes

20 bytes ≤ Total Length < 2^{16} bytes = 65536 bytes

IP Packet Format

Version (4 bits): current version is 4, next version will be 6.

Header length (4 bits): length of IP header, in multiples of 4 bytes

DS/ECN field (1 byte)

This field was previously called as Type-of-Service (TOS) field. The role of this field has been re-defined, but is “backwards compatible” to TOS interpretation

Differentiated Service (DS) (6 bits):

Used to specify service level (currently not supported in the Internet)

Explicit Congestion Notification (ECN) (2 bits):

New feedback mechanism used by TCP

IP Packet Format

Identification (16 bits): Unique identification of a datagram from a host. Incremented whenever a datagram is transmitted

Flags (3 bits):

First bit always set to 0

DF bit (Do not fragment)

MF bit (More fragments)

IP Packet Format

Time To Live (TTL) (1 byte):

Specifies longest paths before datagram is dropped

Role of TTL field: Ensure that packet is eventually dropped when a routing loop occurs

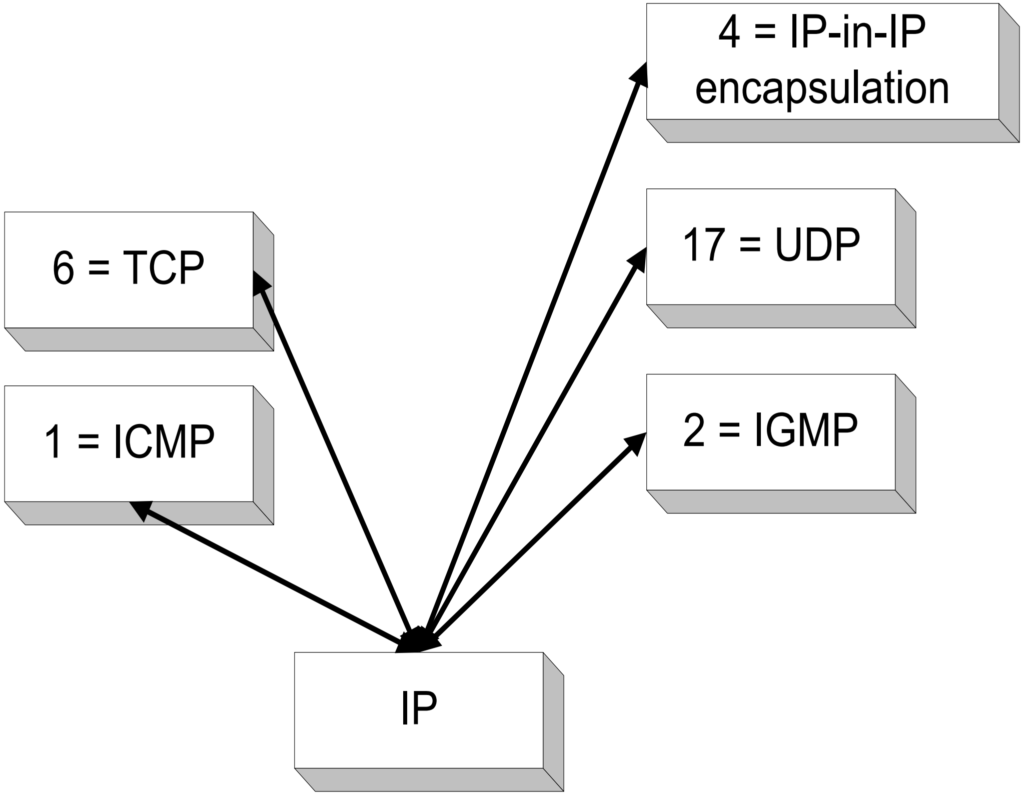
Used as follows:

Sender sets the value (e.g., 64)

Each router decrements the value by 1

When the value reaches 0, the datagram is dropped

Fields of IP Header



Protocol (1 byte):

Specifies the higher-layer protocol.

Used for demultiplexing to higher layers.

Fields of IP Header

Options:
Security restrictions

Record Route: each router that processes the packet adds its IP address to the header.

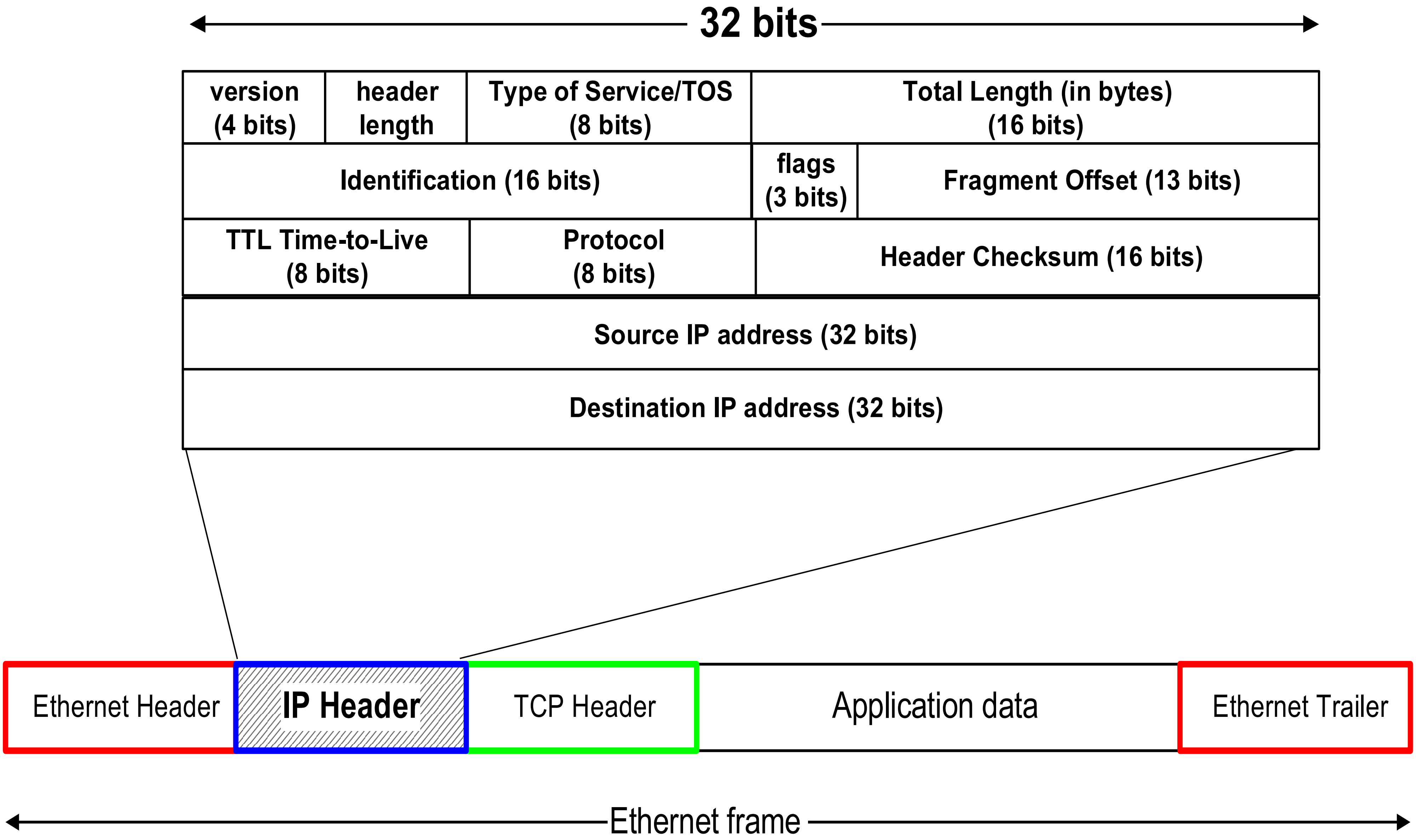
Timestamp: each router that processes the packet adds its IP address and time to the header.

(loose) Source Routing: specifies a list of routers that must be traversed.

(strict) Source Routing: specifies a list of the only routers that can be traversed.

Padding: Padding bytes are added to ensure that header ends on a 4-byte boundary

IP header (e.g. no options)



32 bits

version (4 bits)	header length	Type of Service/TOS (8 bits)	Total Length (in bytes) (16 bits)	
Identification (16 bits)			flags (3 bits)	Fragment Offset (13 bits)
TTL Time-to-Live (8 bits)		Protocol (8 bits)	Header Checksum (16 bits)	
Source IP address (32 bits)				
Destination IP address (32 bits)				

Ethernet Header

IP Header

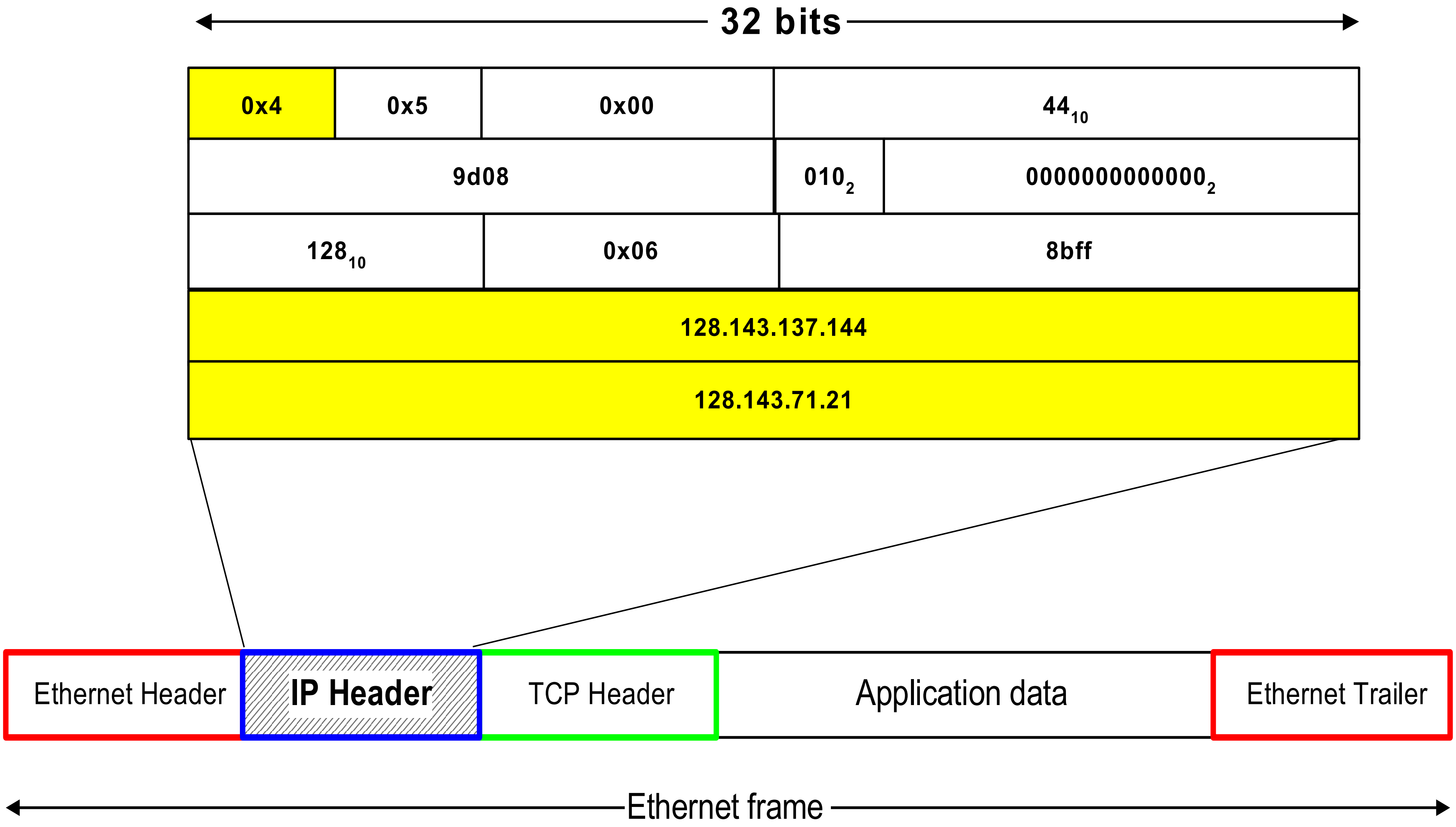
TCP Header

Application data

Ethernet Trailer

Ethernet frame

IP Header (IPv4 example)



32 bits

0x4 | 0x5 | 0x00 | 44_{10}

9d08 | 010_2 | 0000000000000_2

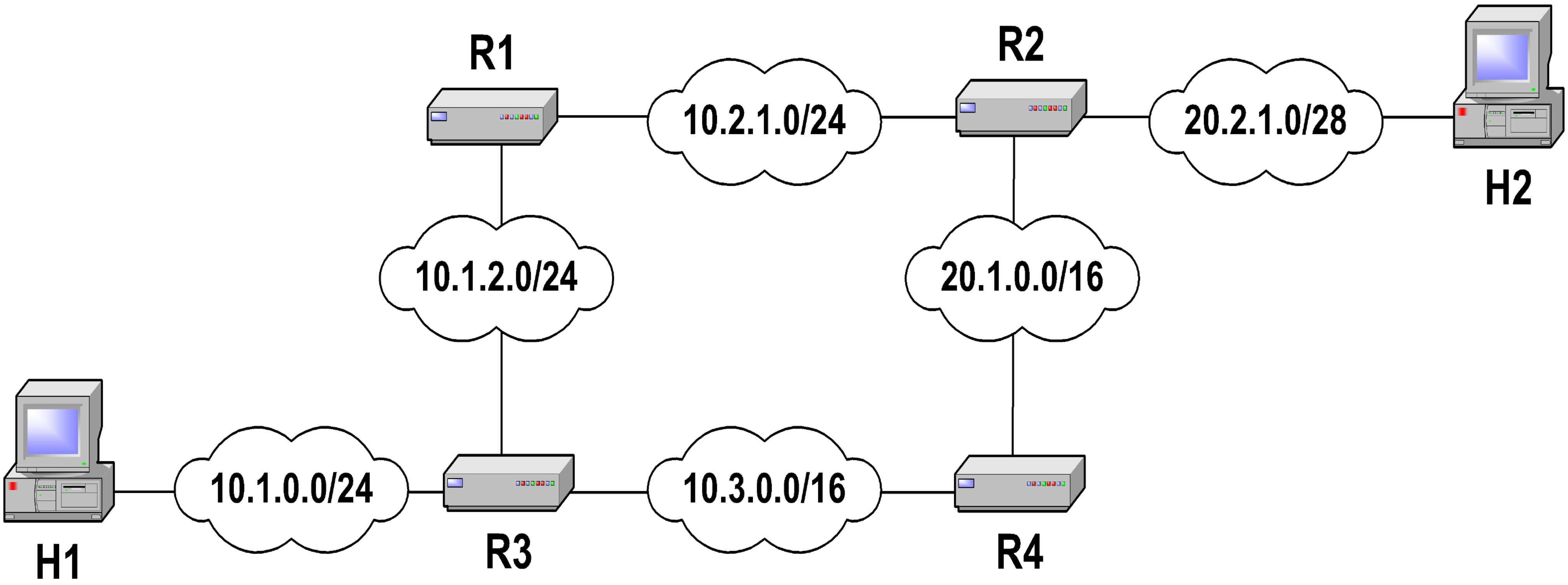
128_{10} | 0x06 | 8bff

128.143.137.144

128.143.71.21

Fowarding

Delivery of an IP Packet



R1 ... R4 are routers

H1, H2 are hosts

How to find a route from H1 to H2?

Routing

Routing table of a host or router

IP datagrams can be directly delivered (“direct”) or is sent to a router (“R4”)

Destination	Next Hop	interface
10.1.0.0/24	direct	eth0
10.1.2.0/24	direct	eth2
10.2.1.0/24	R4	serial0
10.3.1.0/24	direct	eth1
20.1.0.0/16	R4	serial0
20.2.1.0/28	R4	serial0

Each router and each host keeps a routing table which tells the router how to process an outgoing packet

Main columns:

Destination address: where is the IP datagram going to?

Next hop: how to send the IP datagram?

Interface: what is the output port?

Next hop and interface column can often be summarized as one column

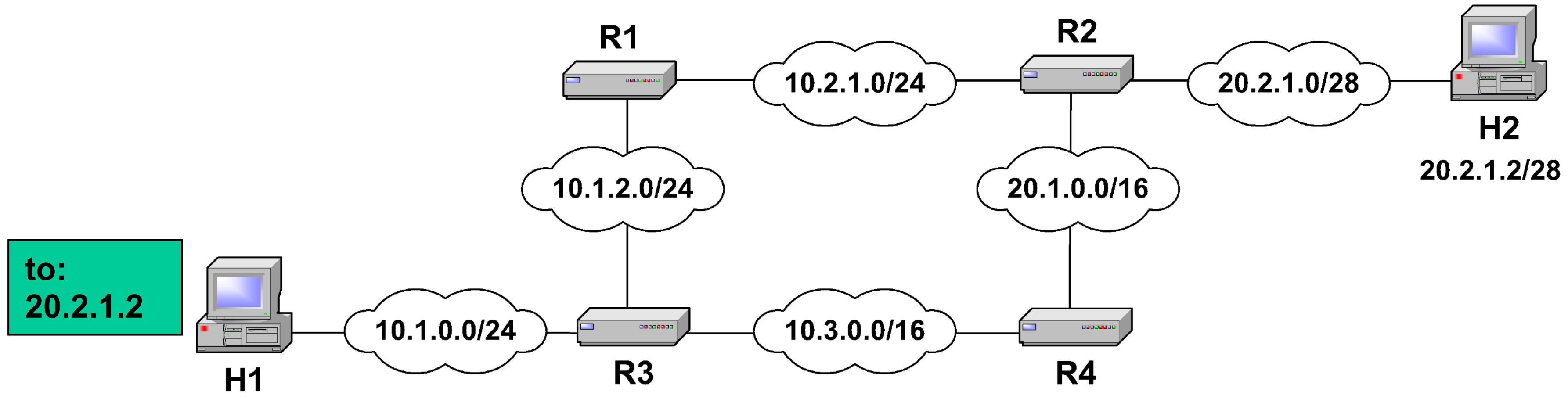
Routing tables are set to help each datagram get closer to the its destination

Delivery with Routing Tables

Destination	Next Hop
10.1.0.0/24	R3
10.1.2.0/24	direct
10.2.1.0/24	direct
10.3.1.0/24	R3
20.2.0.0/16	R2
30.1.1.0/28	R2

Destination	Next Hop
10.1.0.0/24	R1
10.1.2.0/24	R1
10.2.1.0/24	direct
10.3.1.0/24	R4
20.1.0.0/16	direct
20.2.1.0/28	direct

Destination	Next Hop
10.1.0.0/24	R2
10.1.2.0/24	R2
10.2.1.0/24	R2
10.3.1.0/24	R2
20.1.0.0/16	R2
20.2.1.0/28	direct



Destination	Next Hop
10.1.0.0/24	direct
10.1.2.0/24	R3
10.2.1.0/24	R3
10.3.1.0/24	R3
20.1.0.0/16	R3
20.2.1.0/28	R3

Destination	Next Hop
10.1.0.0/24	direct
10.1.2.0/24	direct
10.2.1.0/24	R4
10.3.1.0/24	direct
20.1.0.0/16	R4
20.2.1.0/28	R4

Destination	Next Hop
10.1.0.0/24	R3
10.1.2.0/24	R3
10.2.1.0/24	R2
10.3.1.0/24	direct
20.1.0.0/16	direct
20.2.1.0/28	R2

Delivery of IP Packets

There are two distinct processes to delivering IP datagrams:

1. Forwarding: How to pass a packet from an input interface to the output interface?
2. Routing: How to find and setup the routing tables?

Forwarding must be done as fast as possible:

- on routers, is often done with support of hardware
- on PCs, is done in kernel of the operating system

Routing is less time-critical

- on a PC, routing is done as a background process

Processing of an IP Packet

Processing of IP datagrams is very similar on an IP router and a host

Main difference:
“IP forwarding” is enabled on router and disabled on host

IP forwarding enabled
⍰ if a datagram is received, but it is not for the local system, the datagram will be sent to a different system

IP forwarding disabled
⍰ if a datagram is received, but it is not for the local system, the datagram will be dropped

Processing of an IP Packet

1) IP header validation

2) Process options in IP header

3) Parsing the destination IP address

4) Routing table lookup

5) Decrement TTL

6) Perform fragmentation (if necessary)

7) Calculate checksum

8) Transmit to next hop

9) Send ICMP packet (if necessary)

Types of routing table entries

Network route

Destination addresses is a network address (e.g., 10.0.2.0/24)
Most entries are network routes

Host route

Destination address is an interface address (e.g., 10.0.1.2/32)
Used to specify a separate route for certain hosts

Default route

Used when no network or host route matches
The router that is listed as the next hop of the default route is the default gateway (for Cisco: gateway of last resort)

Loopback address

Routing table for the loopback address (127.0.0.1)
The next hop lists the loopback (lo0) interface as outgoing interface

Types of routing table entries

Longest Prefix Match: Search for the routing table entry that has the longest match with the prefix of the destination IP address

128.143.71.21

Destination address	Next hop
10.0.0.0/8	R1
128.143.0.0/16	R2
128.143.64.0/20	R3
128.143.192.0/20	R3
128.143.71.0/24	R4
128.143.71.55/32	R3
default	R5

The longest prefix match for 128.143.71.21 is for 24 bits with entry 128.143.71.0/24
Datagram will be sent to R4

1. Search for a match on all 32 bits
2. Search for a match for 31 bits

   …..

32. Search for a mach on 0 bits

Host route, loopback entry
→ 32-bit prefix match
Default route is represented as 0.0.0.0/0
→ 0-bit prefix match

Route Aggragation

Destination	Next Hop
10.1.0.0/24	R3
10.1.2.0/24	direct
10.2.1.0/24	direct
10.3.1.0/24	R3
20.2.0.0/16	R2
30.1.1.0/28	R2

Destination	Next Hop
10.1.0.0/24	R3
10.1.2.0/24	direct
10.2.1.0/24	direct
10.3.1.0/24	R3
16.0.0.0/4	R2

Longest prefix match algorithm permits to aggregate prefixes with identical next hop address to a single entry

This contributes significantly to reducing the size of routing tables of Internet routers

Updating Routing Tables

Adding a route:
Route all traffic to 10.0.2/24 through `eth2`

Destination	Next Hop/ interface
10.0.2.0/24	eth2

Adding a default gateway:
Route all traffic that does not match any entry through `eth0`

Destination	Next Hop/ interface
0.0.0.0/0	eth0

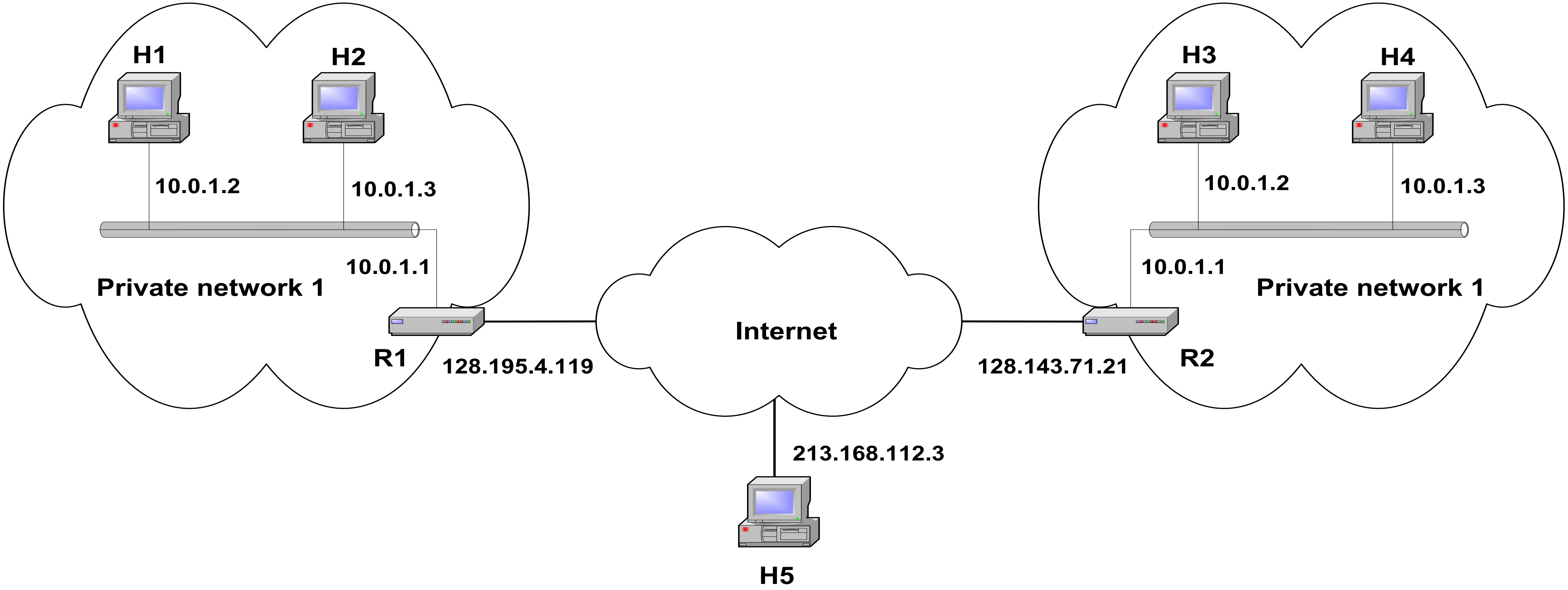
Adding and Updating Routes:

- Manual: static configuration of network routes or host routes
- Automatic: dynamic configuration with routing protocols

ICMP messages help management of routing tables

NAT (Network Address Translation)

Private Networks



Private IP network is an IP network that is not directly connected to the Internet

IP addresses in a private network can be assigned arbitrarily.
Not registered and not guaranteed to be globally unique

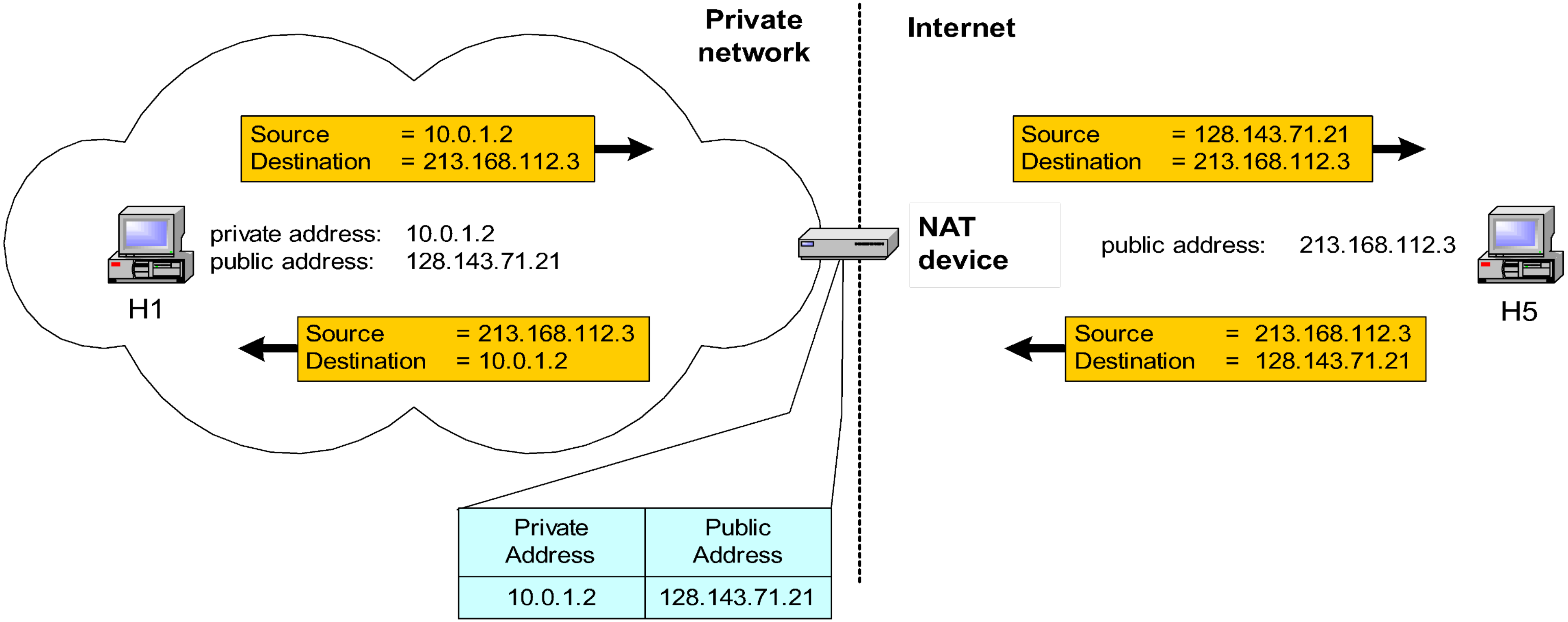
Network Address Translation

NAT is a router function where IP addresses (and possibly port numbers) of IP datagrams are replaced at the boundary of a private network

NAT is a method that enables hosts on private networks to communicate with hosts on the Internet

NAT is run on routers that connect private networks to the public Internet, to replace the (IP address, port) pair of an IP packet with another (IP address, port) pair.

Network Address Translation



Private network

Internet

Source = 10.0.1.2
Destination = 213.168.112.3

private address: 10.0.1.2
public address: 128.143.71.21

Source = 213.168.112.3
Destination = 10.0.1.2

Source = 128.143.71.21
Destination = 213.168.112.3

public address: 213.168.112.3

Source = 213.168.112.3
Destination = 128.143.71.21

Private Address	Public Address
10.0.1.2	128.143.71.21

NAT device has address translation table

Main uses of NAT

Pooling of IP addresses

Supporting migration between network service providers

IP masquerading

Load balancing of servers

Pooling of IP addresses

Scenario: Corporate network has many hosts but only a small number of public IP addresses

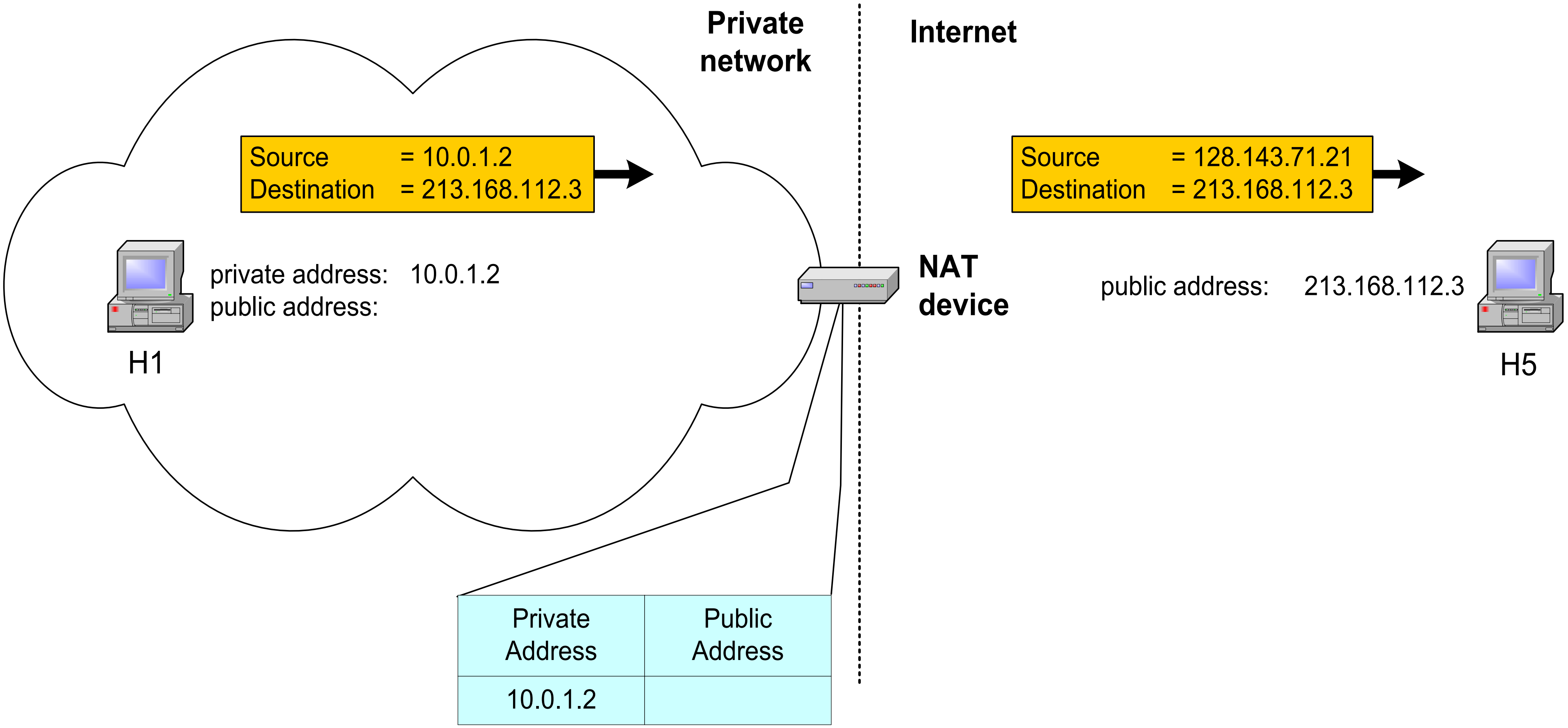
NAT solution:

- Corporate network is managed with a private address space

- NAT device, located at the boundary between the corporate network and the public Internet, manages a pool of public IP addresses

- When a host from the corporate network sends an IP datagram to a host in the public Internet, the NAT device picks a public IP address from the address pool, and binds this address to the private address of the host

Pooling of IP addresses



Source = 10.0.1.2
Destination = 213.168.112.3

private address: 10.0.1.2
public address:

Source = 128.143.71.21
Destination = 213.168.112.3

public address: 213.168.112.3

Private Address	Public Address
10.0.1.2	

Pool of addresses: 128.143.71.0-128.143.71.30

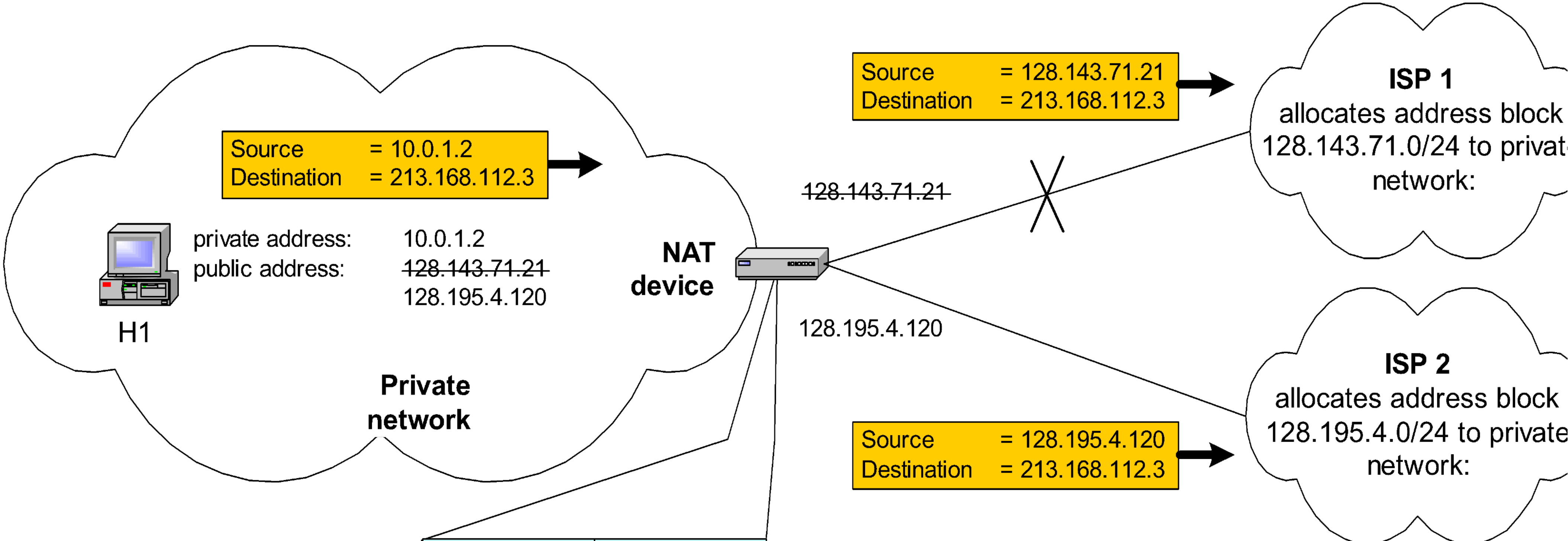
Supporting migration between network service providers

Scenario: In CIDR, the IP addresses in a corporate network are obtained from the service provider.
Changing the service provider requires changing all IP addresses in the network.

NAT solution:
- Assign private addresses to the hosts of the corporate network
- NAT device has static address translation entries which bind the private address of a host to the public address.
- Migration to a new network service provider merely requires an update of the NAT device.
The migration is not noticeable to the hosts on the network.

Note:
- The difference to the use of NAT with IP address pooling is that the mapping of public and private IP addresses is static.

Supporting migration between network service providers



Source = 10.0.1.2
Destination = 213.168.112.3

private address: 10.0.1.2
public address: ~~128.143.71.21~~ 128.195.4.120

H1

NAT device

Private network

Source = 128.143.71.21
Destination = 213.168.112.3

~~128.143.71.21~~

128.195.4.120

ISP 1
allocates address block
128.143.71.0/24 to privat
network:

ISP 2
allocates address block
128.195.4.0/24 to private
network:

Source = 128.195.4.120
Destination = 213.168.112.3

Private Address	Public Address
10.0.1.2	~~128.143.71.21~~ 128.195.4.120

IP Masquerading

Also called: Network address and port translation (NAPT), port address translation (PAT).

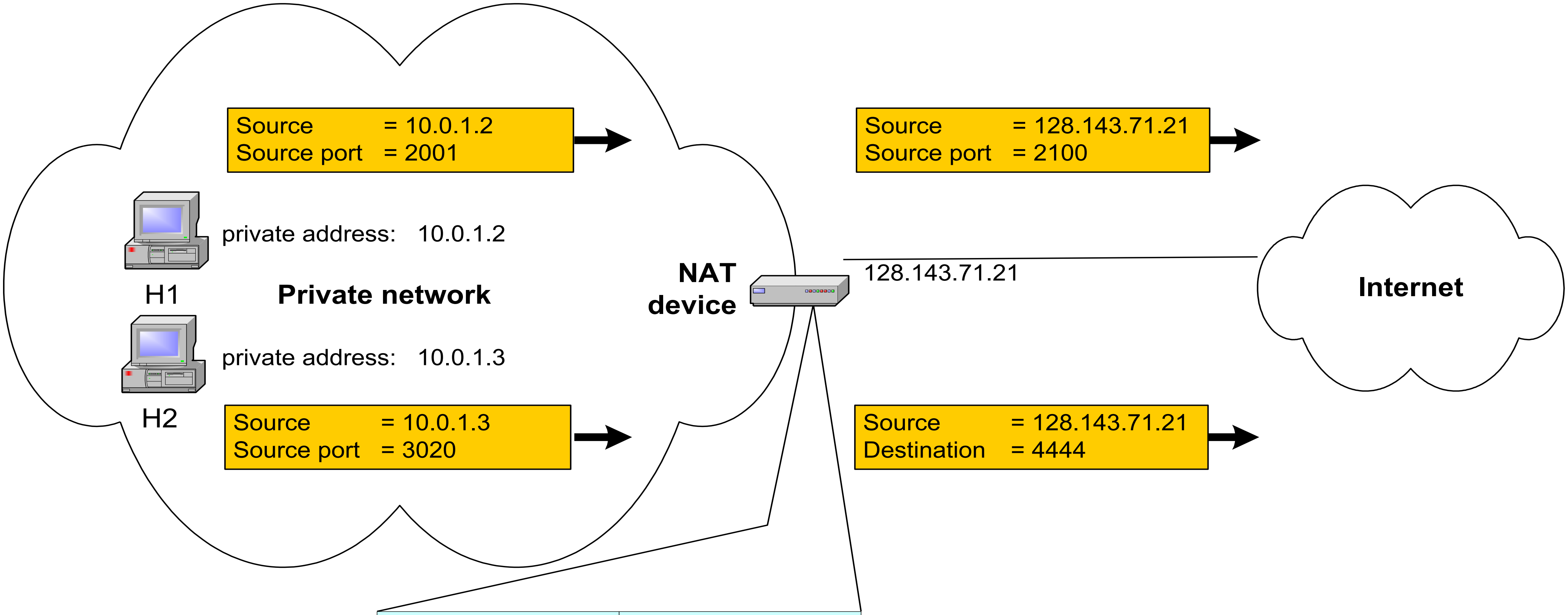
Scenario: Single public IP address is mapped to multiple hosts in a private network.

NAT solution:

- Assign private addresses to the hosts of the corporate network

- NAT device modifies the port numbers for outgoing traffic

IP Masquerading



Source = 10.0.1.2
Source port = 2001

private address: 10.0.1.2

H1

Private network

private address: 10.0.1.3

H2

Source = 10.0.1.3
Source port = 3020

NAT
device

Source = 128.143.71.21
Source port = 2100

128.143.71.21

Internet

Source = 128.143.71.21
Destination = 4444

Private Address	Public Address
10.0.1.2/2001	128.143.71.21/2100
10.0.1.3/3020	128.143.71.21/4444

Load balancing of servers

Scenario: Balance the load on a set of identical servers, which are accessible from a single IP address

NAT solution:

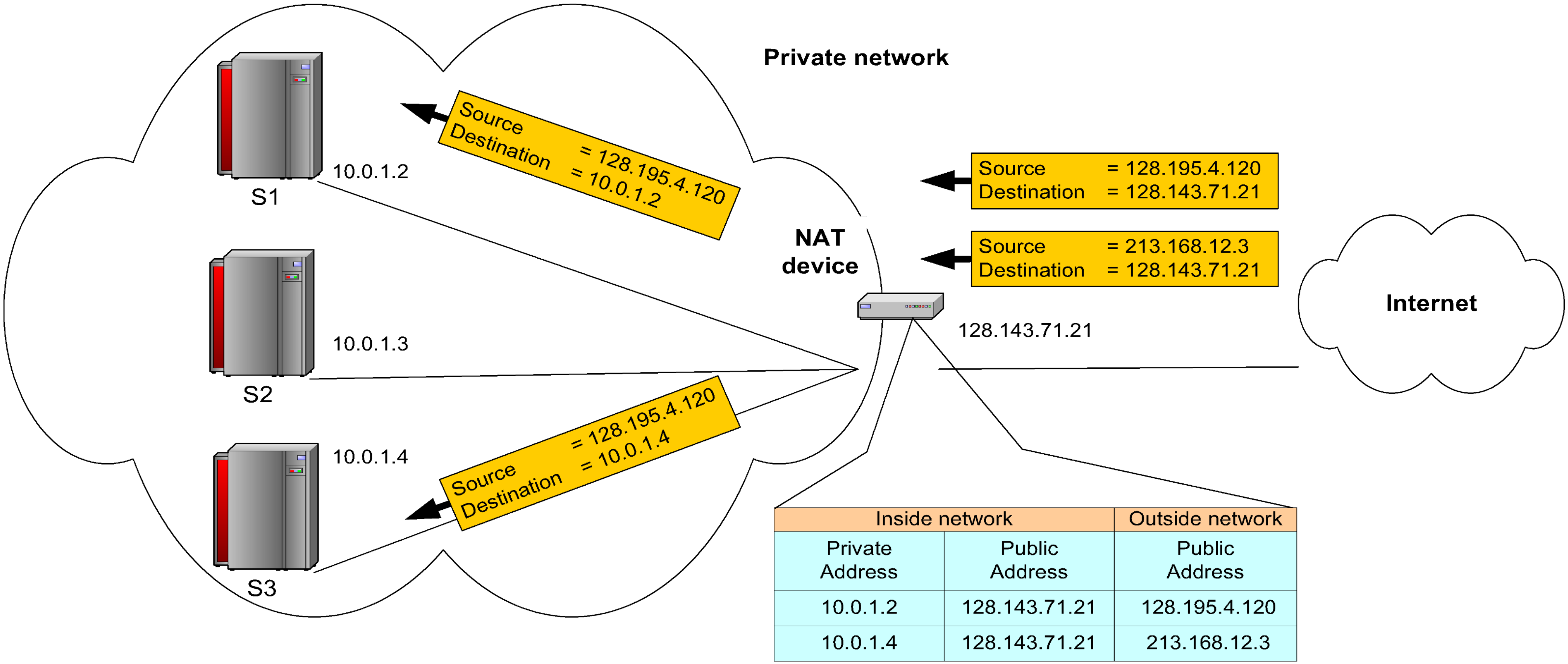
- Here, the servers are assigned private addresses

- NAT device acts as a proxy for requests to the server from the public network

- The NAT device changes the destination IP address of arriving packets to one of the private addresses for a server

- A sensible strategy for balancing the load of the servers is to assign the addresses of the servers in a round-robin fashion.

Load balancing of servers



Source = 128.195.4.120
Destination = 10.0.1.2

Source = 128.195.4.120
Destination = 10.0.1.4

Source = 128.195.4.120
Destination = 128.143.71.21

Source = 213.168.12.3
Destination = 128.143.71.21

Inside network		Outside network
Private Address	Public Address	Public Address
10.0.1.2	128.143.71.21	128.195.4.120
10.0.1.4	128.143.71.21	213.168.12.3

Concerns about NAT

Performance:
- Modifying the IP header by changing the IP address requires that NAT boxes recalculate the IP header checksum

- Modifying port number requires that NAT boxes recalculate TCP checksum

Fragmentation:
- Care must be taken that a datagram that is fragmented before it reaches the NAT device, is not assigned a different IP address or different port numbers for each of the fragments.

Concerns about NAT

End-to-end connectivity:

- NAT destroys universal end-to-end reachability of hosts on the Internet.

- A host in the public Internet often cannot initiate communication to a host in a private network.

- The problem is worse, when two hosts, that are in different private network, need to communicate with each other.

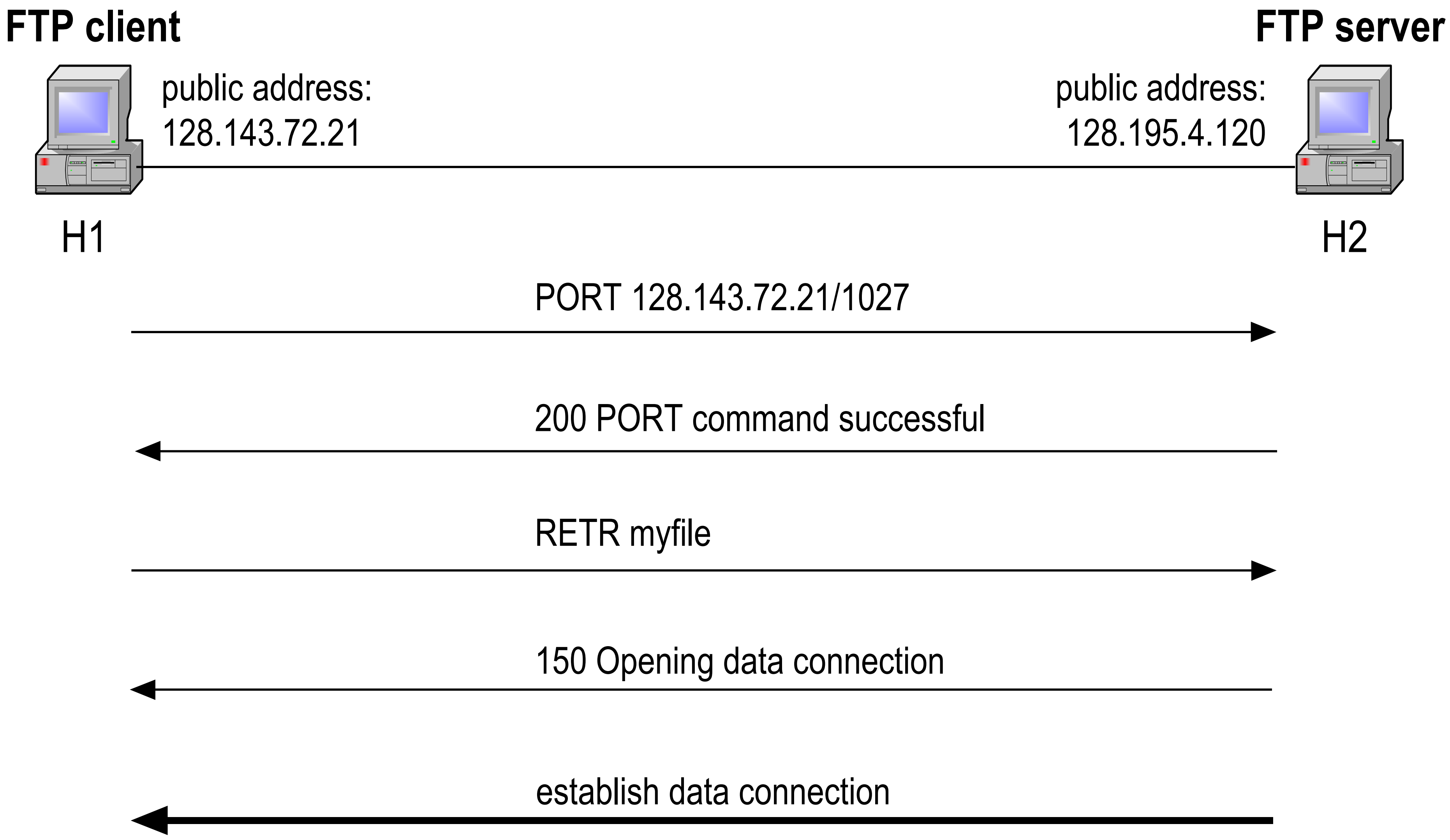
Concerns about NAT

IP address in application data:

- Applications that carry IP addresses in the payload of the application data generally do not work across a private-public network boundary.

- Some NAT devices inspect the payload of widely used application layer protocols and, if an IP address is detected in the application-layer header or the application payload, translate the address according to the address translation table.

Normal FTP operation



FTP client

public address:
128.143.72.21

H1

FTP server

public address:
128.195.4.120

H2

PORT 128.143.72.21/1027

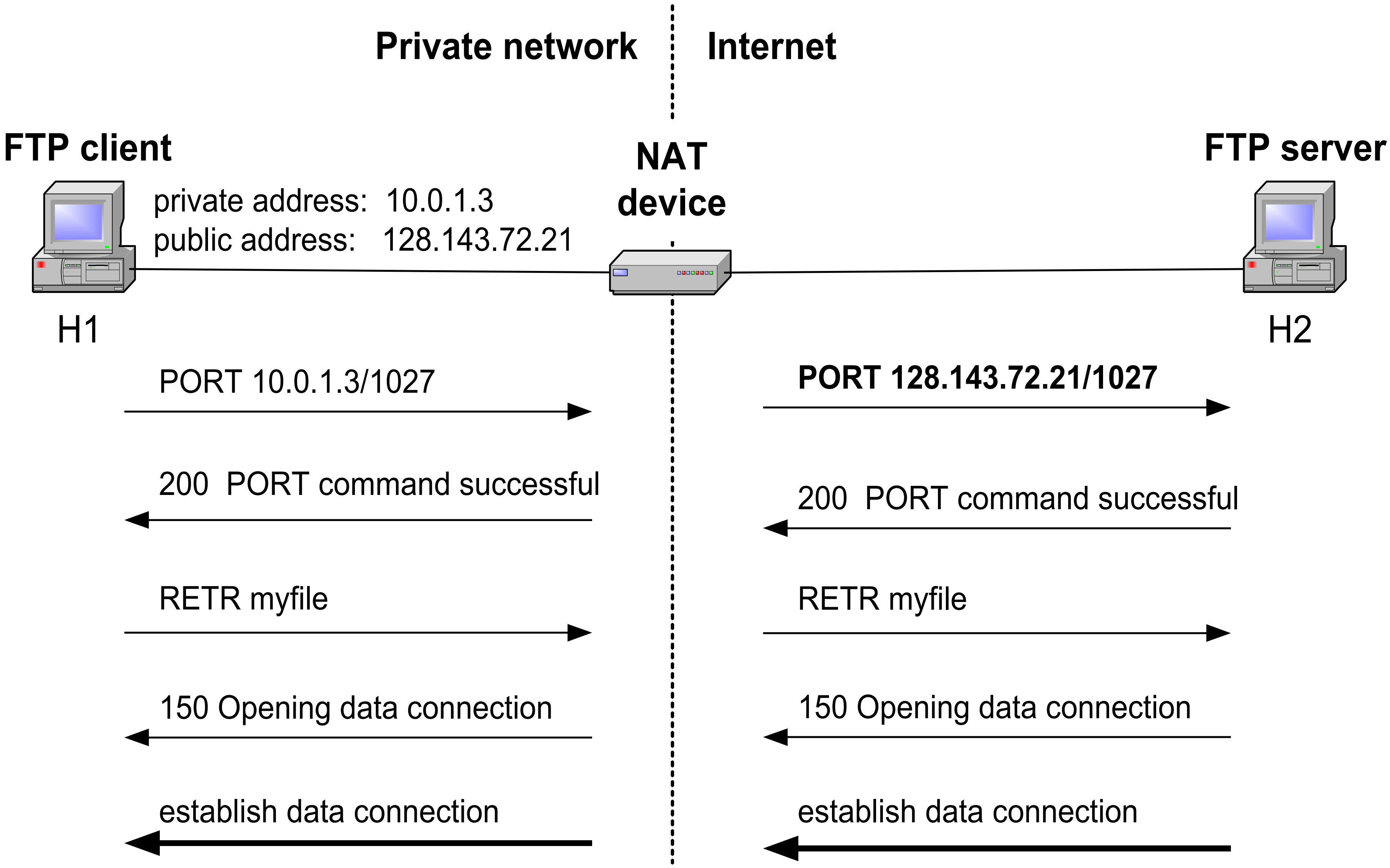
200 PORT command successful

RETR myfile

150 Opening data connection

establish data connection

NAT device with FTP support



Private network

Internet

FTP client

private address: 10.0.1.3
public address: 128.143.72.21

H1

NAT device

FTP server

H2

PORT 10.0.1.3/1027

PORT 128.143.72.21/1027

200 PORT command successful

200 PORT command successful

RETR myfile

RETR myfile

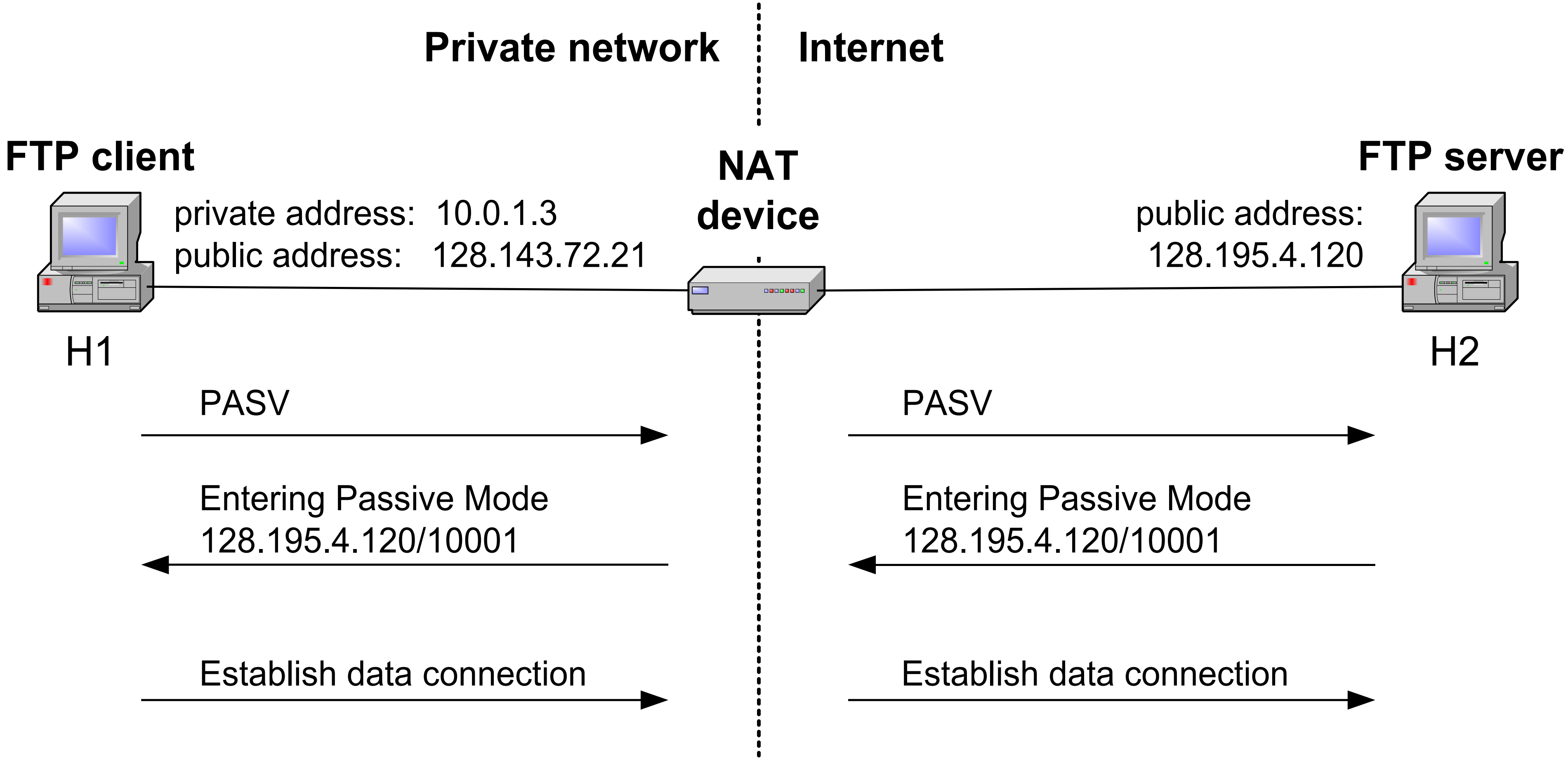
150 Opening data connection

150 Opening data connection

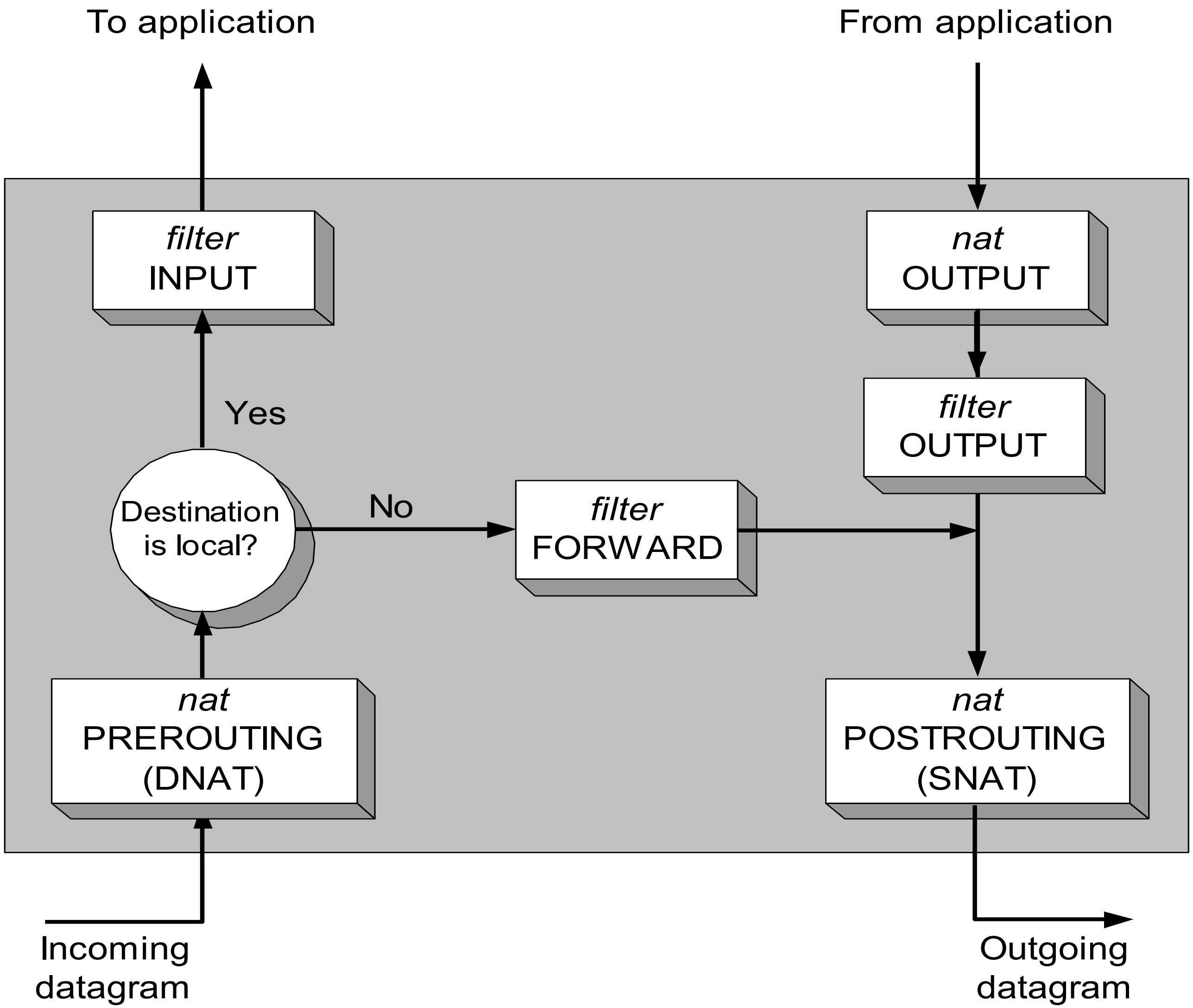
establish data connection

establish data connection

FTP in passive mode and NAT



Configuring NAT in Linux



Linux uses the Netfilter/iptable package to add filtering rules to the IP module

Configuring NAT with iptable

First example:

```
iptables -t nat -A POSTROUTING -s 10.0.1.2
         -j SNAT --to-source 128.143.71.21
```

Pooling of IP addresses:

```
iptables -t nat -A POSTROUTING -s 10.0.1.0/24
         -j SNAT --to-source
          128.128.71.0-128.143.71.30
```

ISP migration:

```
iptables -t nat -R POSTROUTING -s 10.0.1.0/24
         -j SNAT --to-source
          128.195.4.0-128.195.4.254
```

IP masquerading:

```
iptables -t nat -A POSTROUTING -s 10.0.1.0/24
         -o eth1 -j MASQUERADE
```

Load balancing:

```
iptables -t nat -A PREROUTING -i eth1 -j DNAT
--to-destination 10.0.1.2-10.0.1.4
```