Computer Networks
Principles
Network Layer - IP

M. Heusse

Martin.Heusse@imag.fr

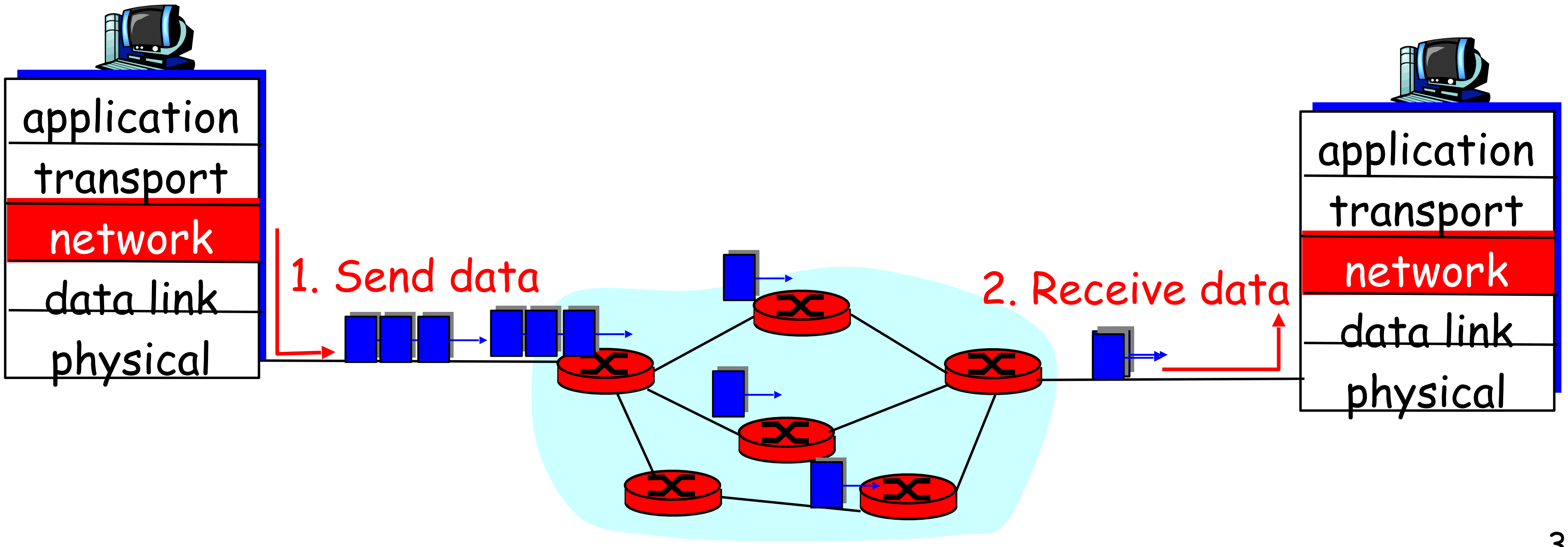
Network Layer

Overview:

- Datagram service
- IP addresses
- Packet forwarding principles
- Details of IP

Datagram networks: the Internet model

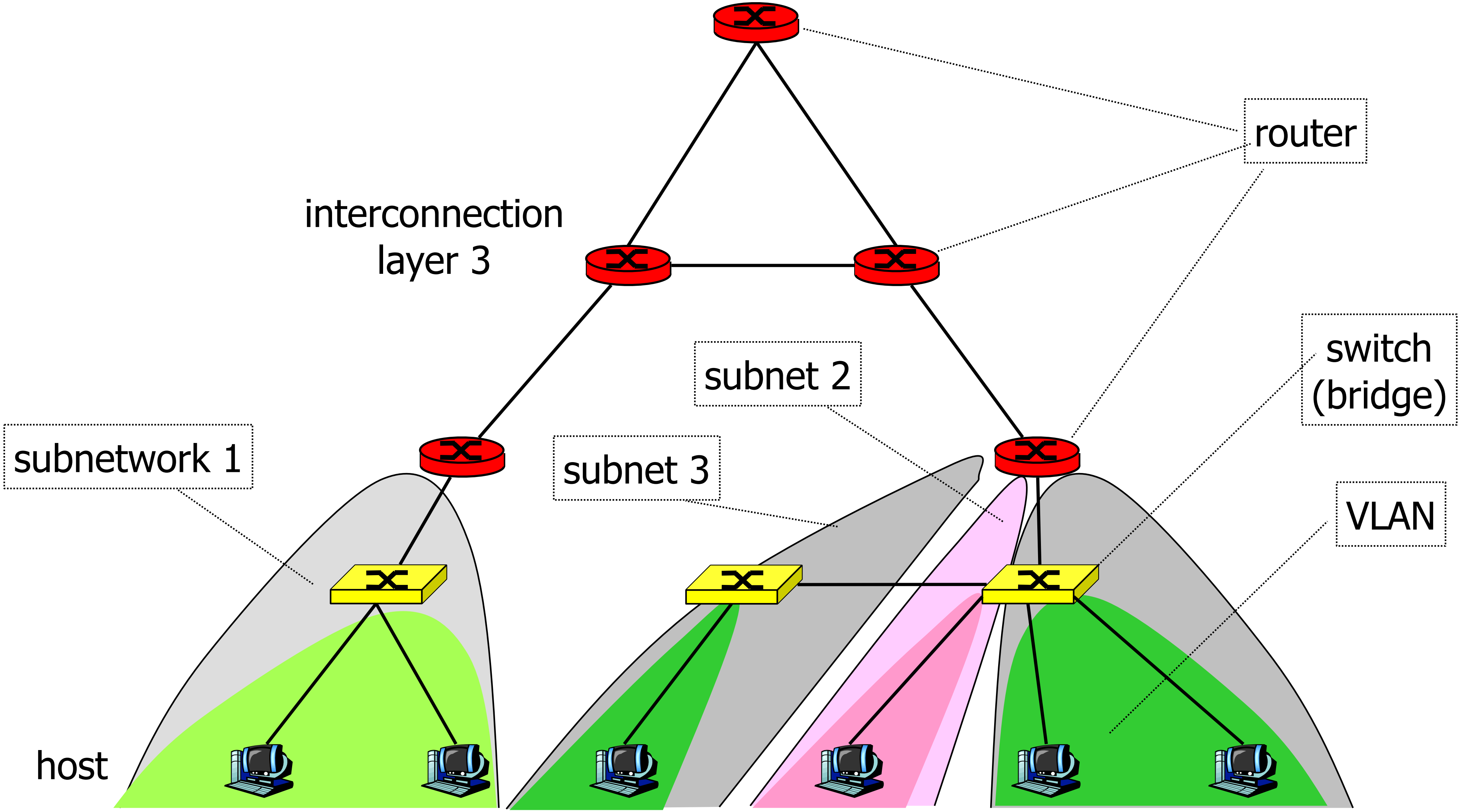
- no call setup at network layer
- routers: no state about end-to-end connections
  - no network-level concept of "connection"
- packets typically routed using destination host ID
  - packets between same source-dest pair may take different paths



IP principles

- Elements
  - **host** = end system; **router** = intermediate system; **subnetwork** = a collection of hosts that can communicate directly without routers
- Routers are between subnetworks only:
  - a subnetwork = a collection of systems with a common prefix
- Packet forwarding
  - **direct**: inside a subnetwork hosts communicate directly without routers, router delivers packets to hosts
  - **indirect**: between subnetworks one or several routers are used
- Host either sends a packet to the destination using its LAN, or it passes it to the router for forwarding

Interconnection structure - layer 3



Interconnection at layer 3

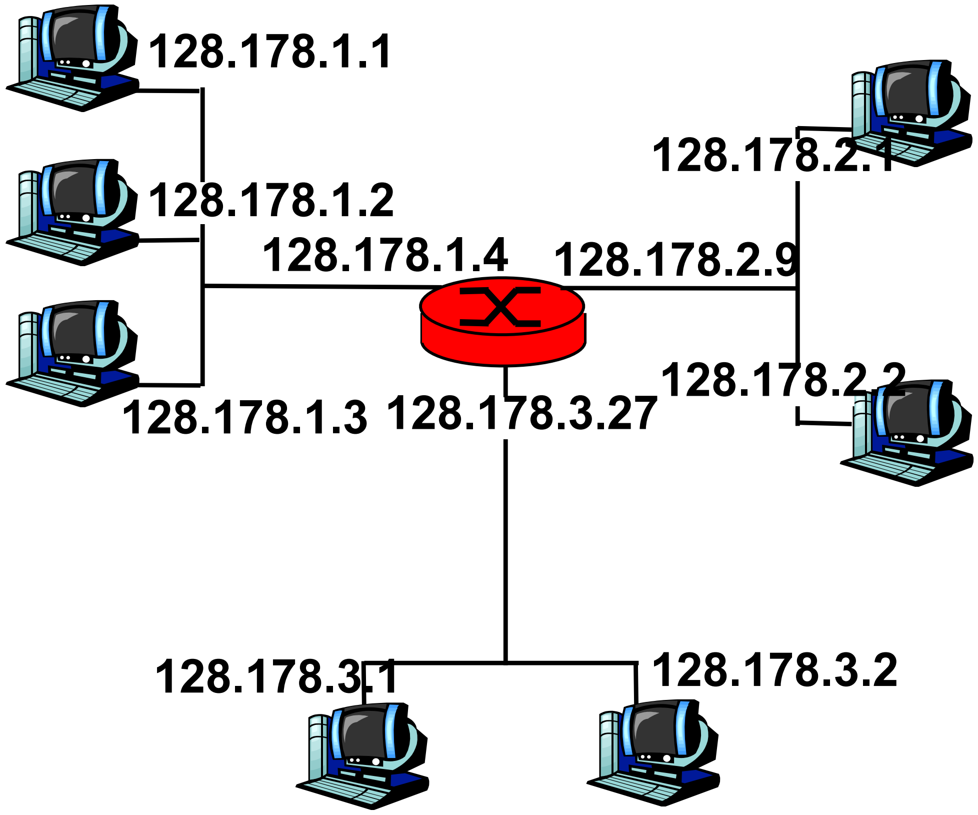
- Routers
  - interconnect subnetworks
  - logically separate groups of hosts
  - managed by one entity
- Forwarding based on IP address
  - structured address space
  - routing tables: aggregation of entries
  - works if no loops - routing protocols
  - scalable inside one administrative domain

IP addresses

- Unique addresses in the world, decentralized allocation
- An IP address is 32 bits, noted in dotted decimal notation: **`192.78.32.2`**
- An IP address has a prefix and a host part:
  - **`prefix:host`**
- Two ways of specifying prefix
  - subnet mask identifies the prefix by bitwise & operation
  - CIDR: bit length of the prefix
- Prefix identifies a subnetwork
  - used for locating a subnetwork - 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 may have multiple interfaces
  - IP addresses associated with interface, not host, router

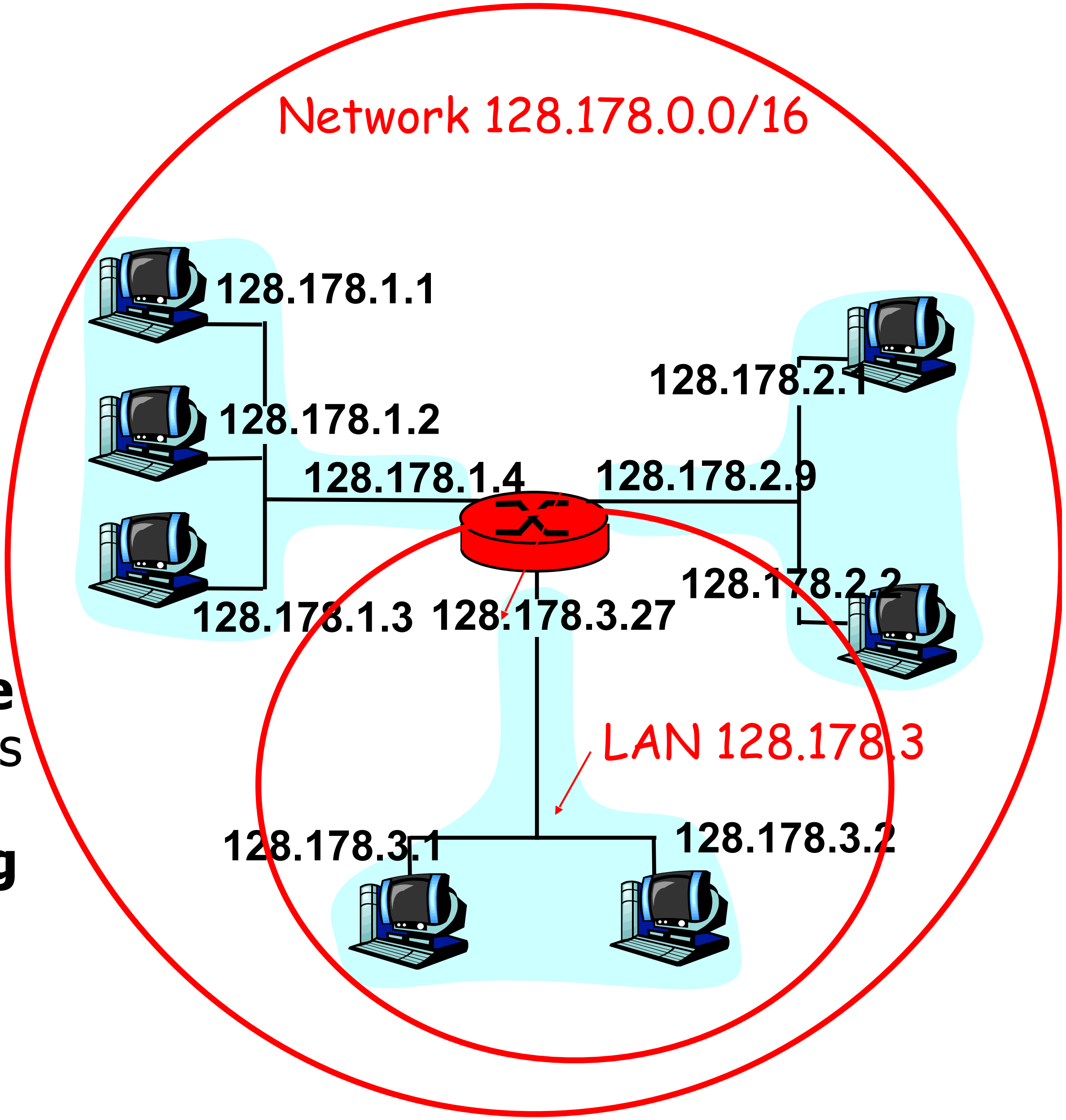


32 bits

$$128.1.1.1 = \underbrace{10000000}_{128}\ \underbrace{00000001}_{1}\ \underbrace{00000001}_{1}\ \underbrace{00000001}_{1}$$

IP Addressing

- IP address:
  - **network** (or prefix) part (high order bits)
  - **host** part (low order bits)
- What's a subnetwork? (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 (sub)networks
(All subnets /24 here)

Special case IP addresses

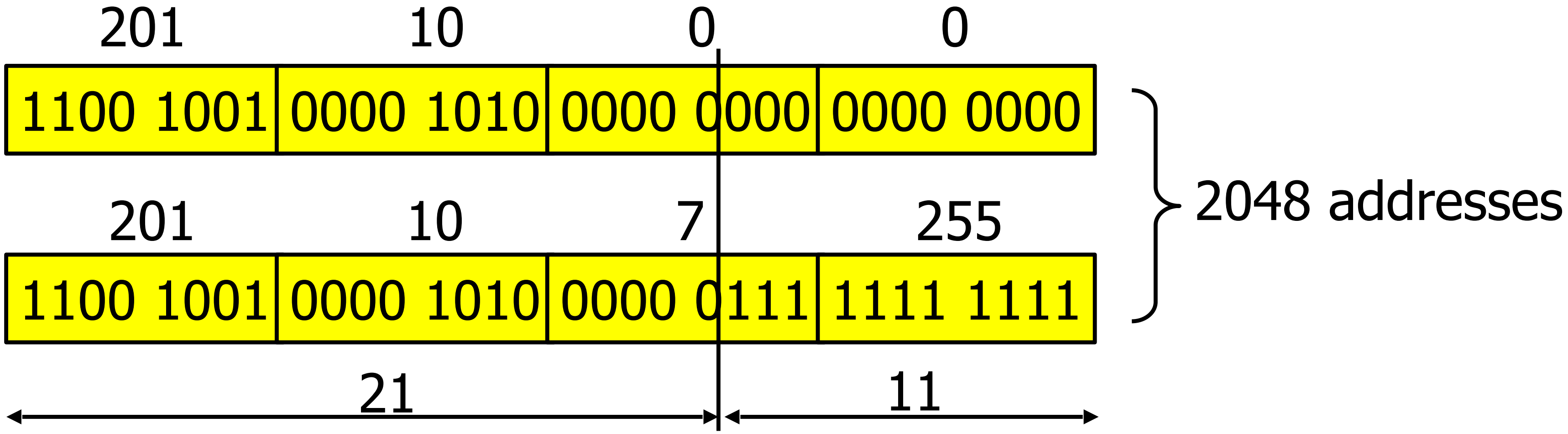
1. 0.0.0.0	this host, on this network
2. 0.hostId	specified host on this net (initialization phase)
3. 255.255.255.255	**limited broadcast (not forwarded by routers)**
4. subnetId.all 1’s	**broadcast on this subnet**
5. subnetId.all 0’s	BSD used it for broadcast on this subnet (obsolete)
6. 127.x.x.x	**loopback**
7. 10/8 **172.16/12** **192.168/16**	**reserved networks for internal use (Intranet)**

- 1,2: source IP@ only; 3,4,5: destination IP@ only

CIDR: IP Address Hierarchies

- The prefix of an IP address is itself structured in order to support aggregation
  - For example: 128.178.x.y represents an EPFL host
    128.178.156 / 24 represents the LRC subnet at EPFL
    128.178/15 represents EPFL
  - Used between routers by routing algorithms
  - This way of doing is called classless and was first introduced in inter domain routing under the name of CIDR (Classless Interdomain Routing)
- Notation: 128.178.0.0/16 means : the prefix made of the 16 **first bits** of the string
- It is equivalent to: 128.178.0.0 with netmask=255.255.0.0
- In the past, the class based addresses, with networks of class A, B or C was used; **now only the distinction between class D** (224.0.0.0 to 239.255.255.255, used for multicast) and non-class D is relevant

CIDR



201 10 0 0

1100 1001 | 0000 1010 | 0000 0000 | 0000 0000

201 10 7 255

1100 1001 | 0000 1010 | 0000 0111 | 1111 1111

2048 addresses

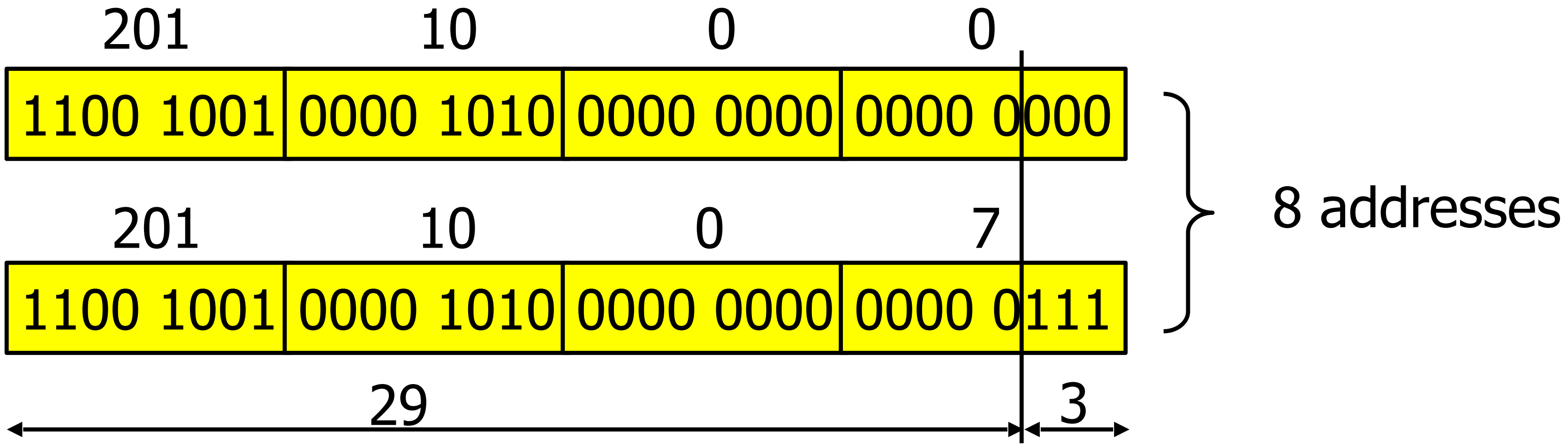
21

11

201.10.0.0/21: 201.10.0.0 - 201.10.0.255
201.10.1.0 - 201.10.1.255
...
201.10.7.0 - 201.10.7.255

1 C class network: 256 addresses
256 x 8 = 2048 addresses

Choosing prefix length



201 10 0 0

1100 1001 0000 1010 0000 0000 0000 0000

201 10 0 7

1100 1001 0000 1010 0000 0000 0000 0111

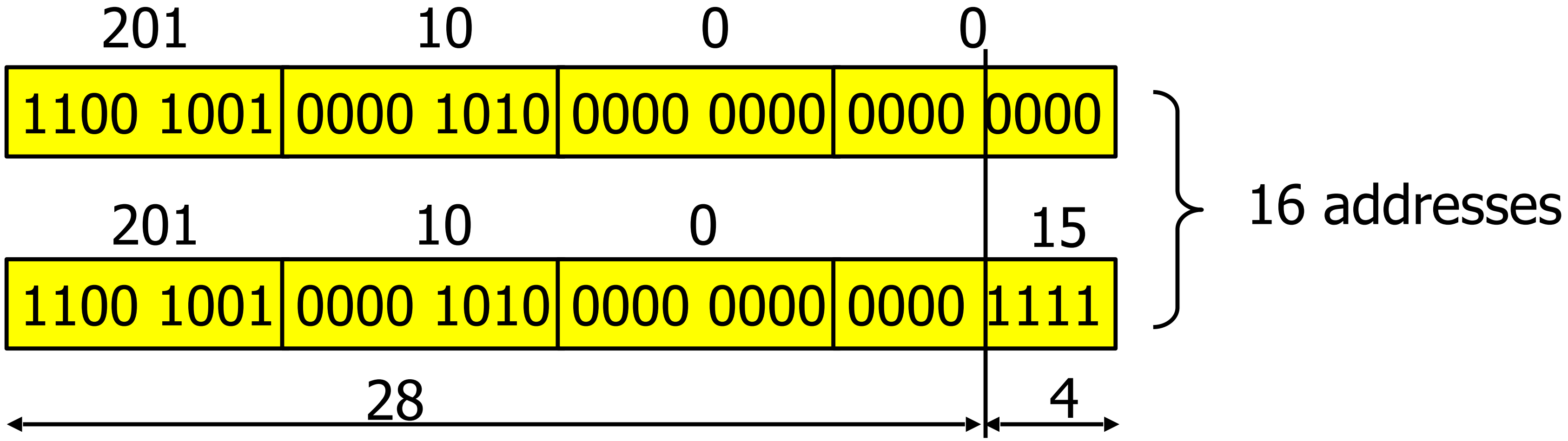
8 addresses

29

3

- prefix = 201.10.0.0/29
  - 8 addresses
  - 2 broadcast addresses: 201.10.0.0, 201.10.0.7
  - only 6 addresses can be used for hosts

Choosing prefix length



201 10 0 0

1100 1001 0000 1010 0000 0000 0000 0000

201 10 0 15

1100 1001 0000 1010 0000 0000 0000 1111

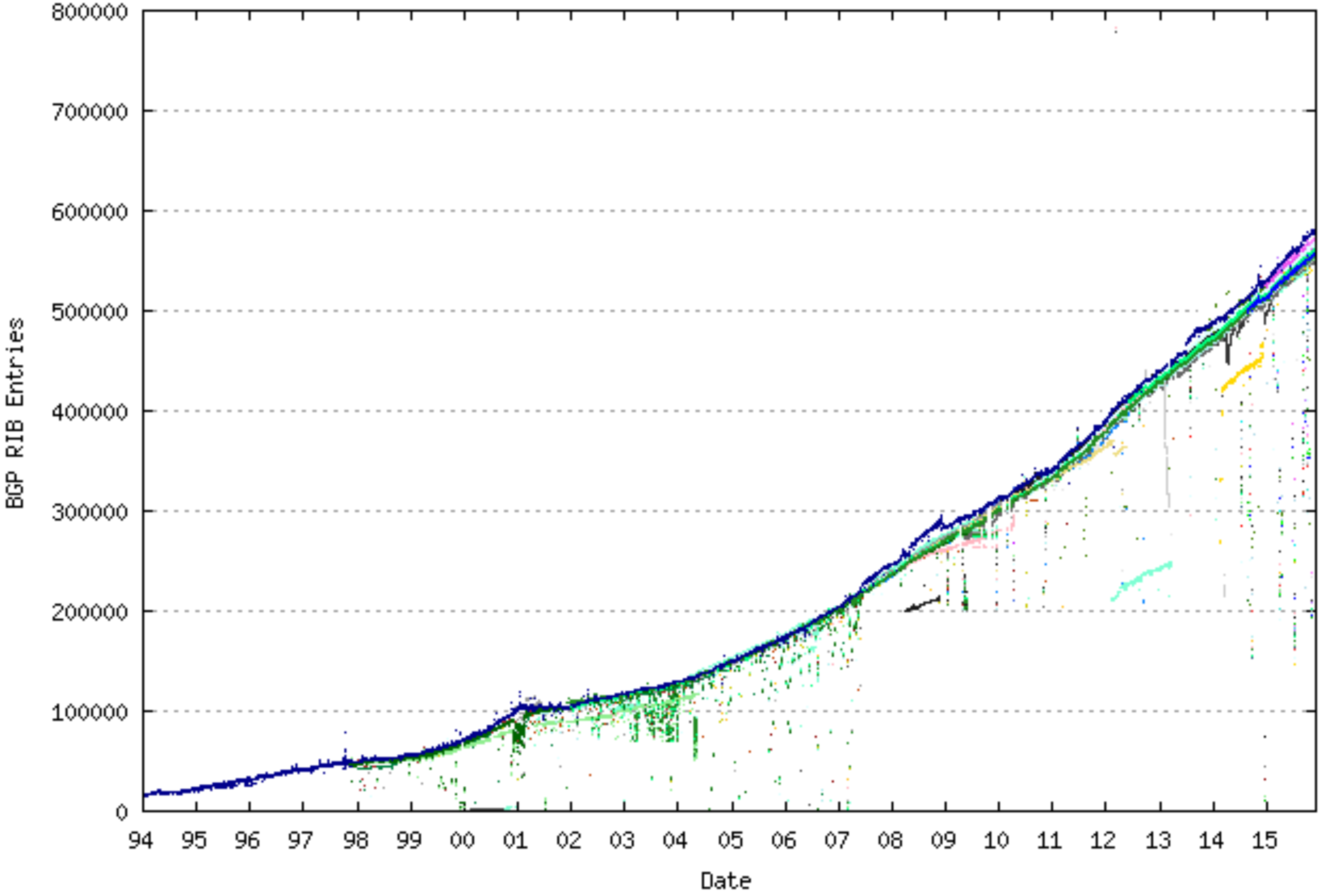
16 addresses

28 4

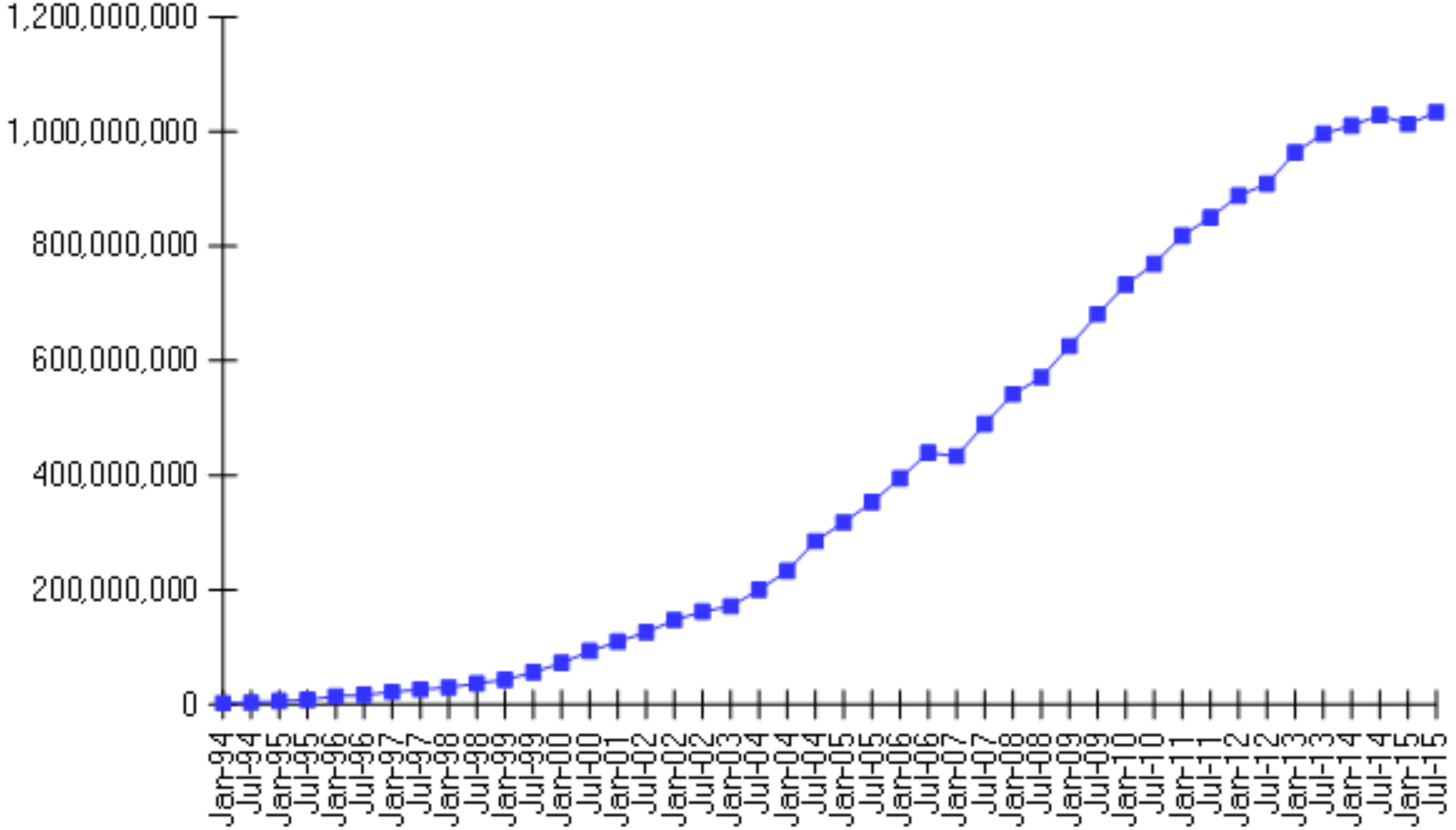
- prefix = 201.10.0.0/28
  - 201.10.0.16/28, 201.10.0.32/28, 201.10.0.48/28…
  - 16 addresses
  - 2 broadcast addresses: 201.10.0.0, 201.10.0.15
  - only 14 addresses can be used for hosts

Address allocation

- World coverage
  - Europe and the Middle East (RIPE NCC)
  - Africa (ARIN & RIPE NCC)
  - North America (ARIN)
  - Latin America including the Caribbean (ARIN)
  - Asia-Pacific (APNIC)
- Current allocations of Class C
  - `193-195/8, 212-213/8, 217/8` for RIPE
  - `199-201/8, 204-209/8, 216/8` for ARIN
  - `202-203/8, 210-211/8, 218/8` for APNIC
- Simplifies routing
  - short prefix aggregates many subnetworks
  - routing decision is taken based on the short prefix


800000
700000
600000
500000
400000
300000
200000
100000
0
BGP RIB Entries
94
95
96
97
98
99
00
01
02
03
04
05
06
07
08
09
10
11
12
13
14
15
Date

Number of hosts



Internet Domain Survey Host Count

Source: Internet Systems Consortium (www.isc.org)

IP Addresses and subnet mask

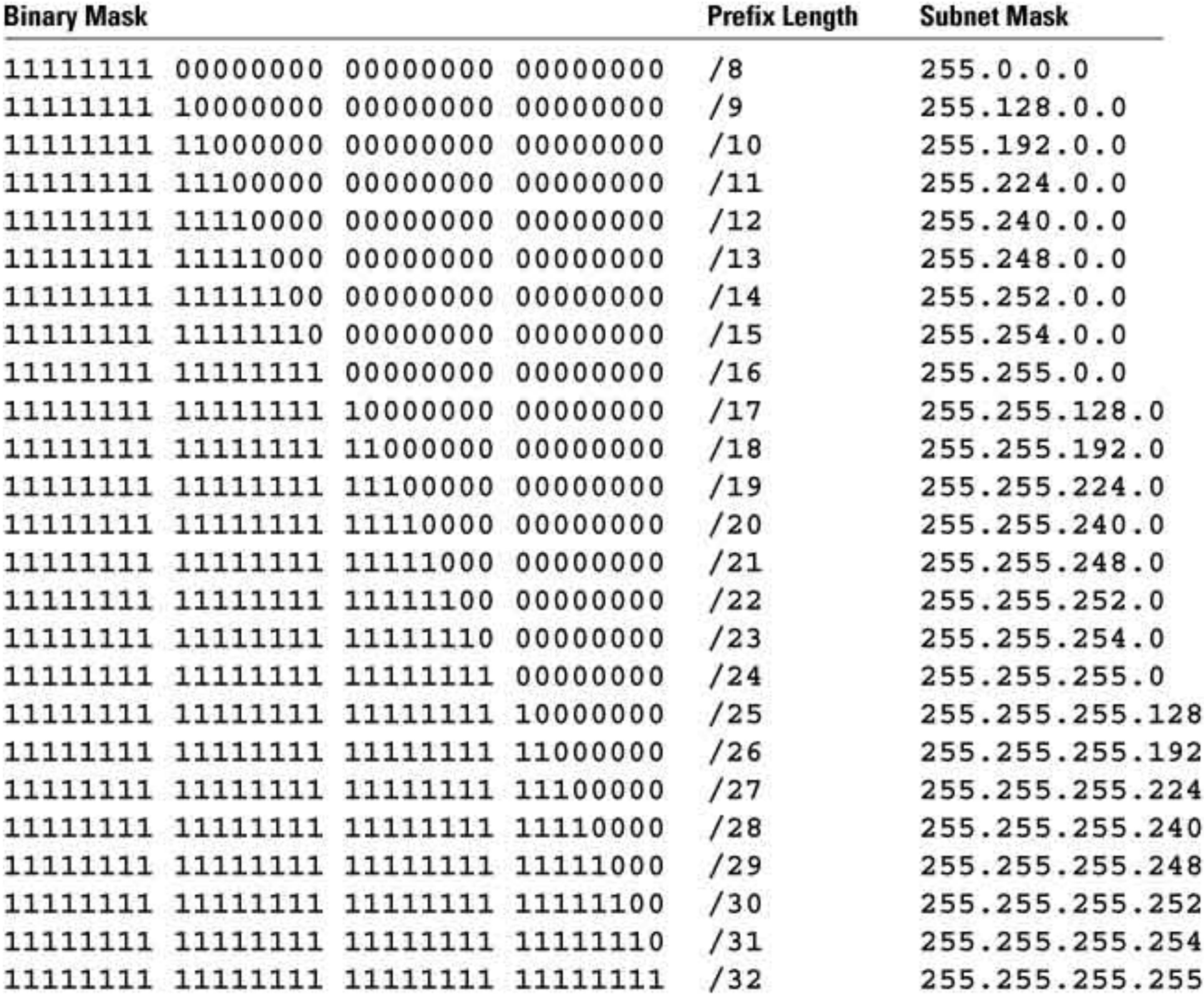
- subnet mask at ETHZ = 255.255.0.0
- CIDR **129.132/16**
- subnet mask at KTK = 255.255.255.192
- CIDR **129.132.119.64/26**
- question: subnet prefix and host parts of spr13.tik.ee.ethz.ch = 129.132.119.77 **?**

```
129.132.119.77 : 10000001.10000100.01110111.01001101
255.255.255.192: 11111111.11111111.11111111.11000000
```

answer:

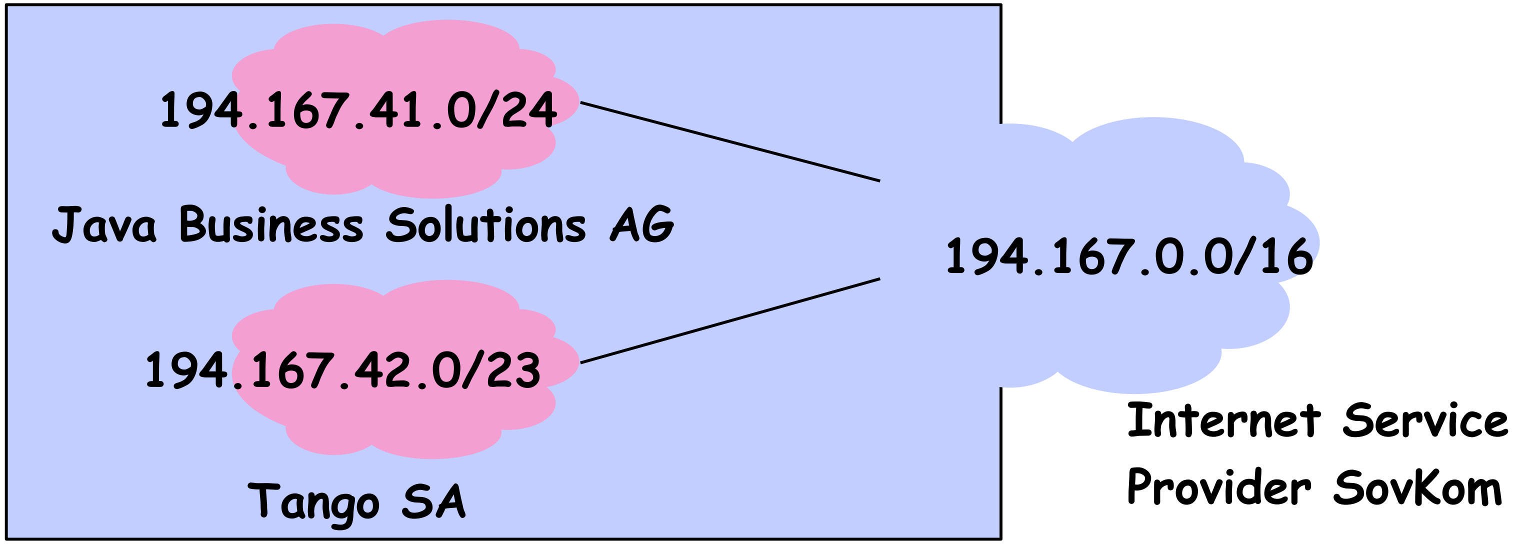
subnet prefix = 129.132.119.64 (64=01000000)

host = 13=001101 (6 bits)



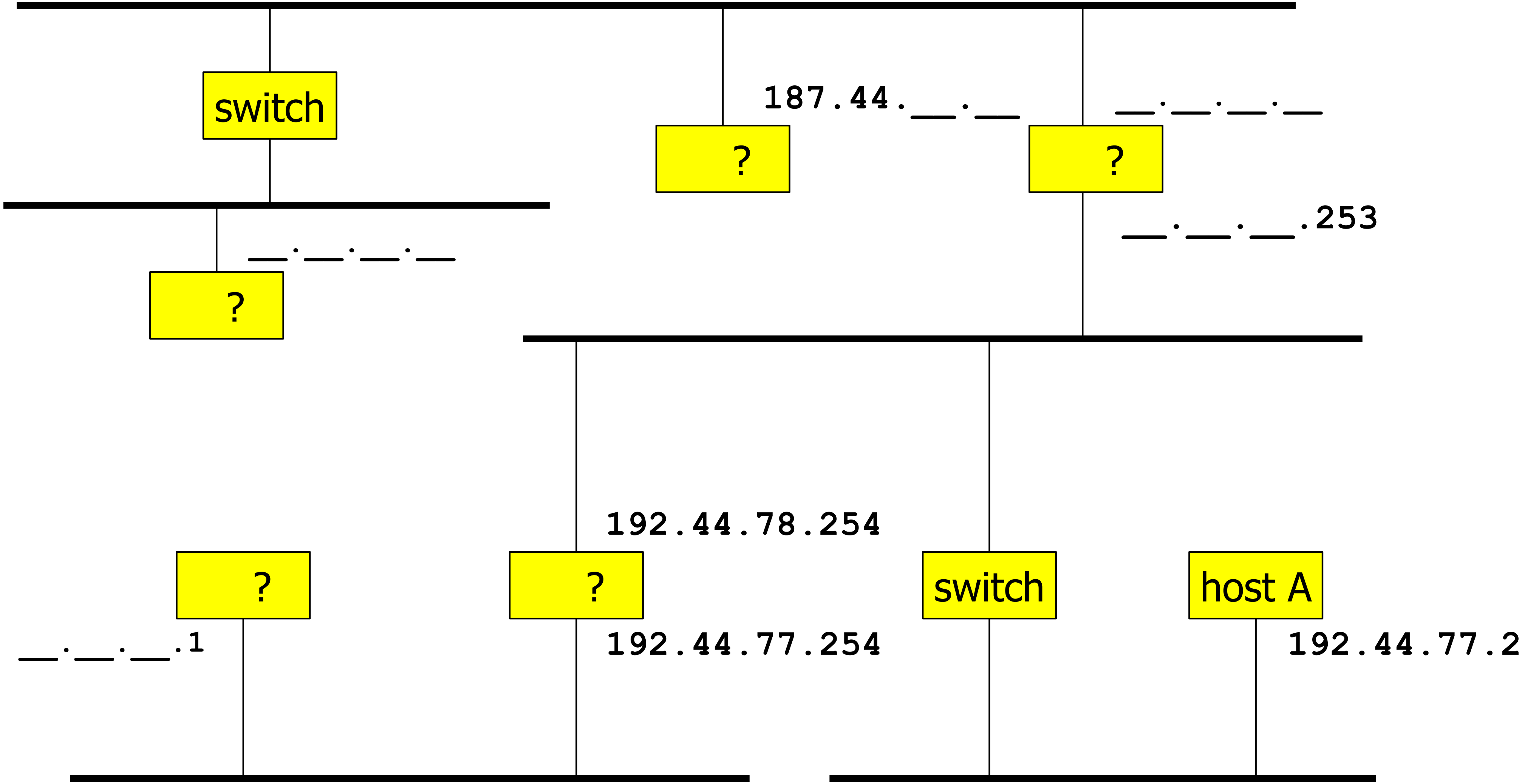
Binary Mask	Prefix Length	Subnet Mask
11111111 00000000 00000000 00000000	/8	255.0.0.0
11111111 10000000 00000000 00000000	/9	255.128.0.0
11111111 11000000 00000000 00000000	/10	255.192.0.0
11111111 11100000 00000000 00000000	/11	255.224.0.0
11111111 11110000 00000000 00000000	/12	255.240.0.0
11111111 11111000 00000000 00000000	/13	255.248.0.0
11111111 11111100 00000000 00000000	/14	255.252.0.0
11111111 11111110 00000000 00000000	/15	255.254.0.0
11111111 11111111 00000000 00000000	/16	255.255.0.0
11111111 11111111 10000000 00000000	/17	255.255.128.0
11111111 11111111 11000000 00000000	/18	255.255.192.0
11111111 11111111 11100000 00000000	/19	255.255.224.0
11111111 11111111 11110000 00000000	/20	255.255.240.0
11111111 11111111 11111000 00000000	/21	255.255.248.0
11111111 11111111 11111100 00000000	/22	255.255.252.0
11111111 11111111 11111110 00000000	/23	255.255.254.0
11111111 11111111 11111111 00000000	/24	255.255.255.0
11111111 11111111 11111111 10000000	/25	255.255.255.128
11111111 11111111 11111111 11000000	/26	255.255.255.192
11111111 11111111 11111111 11100000	/27	255.255.255.224
11111111 11111111 11111111 11110000	/28	255.255.255.240
11111111 11111111 11111111 11111000	/29	255.255.255.248
11111111 11111111 11111111 11111100	/30	255.255.255.252
11111111 11111111 11111111 11111110	/31	255.255.255.254
11111111 11111111 11111111 11111111	/32	255.255.255.255

IP Addresses



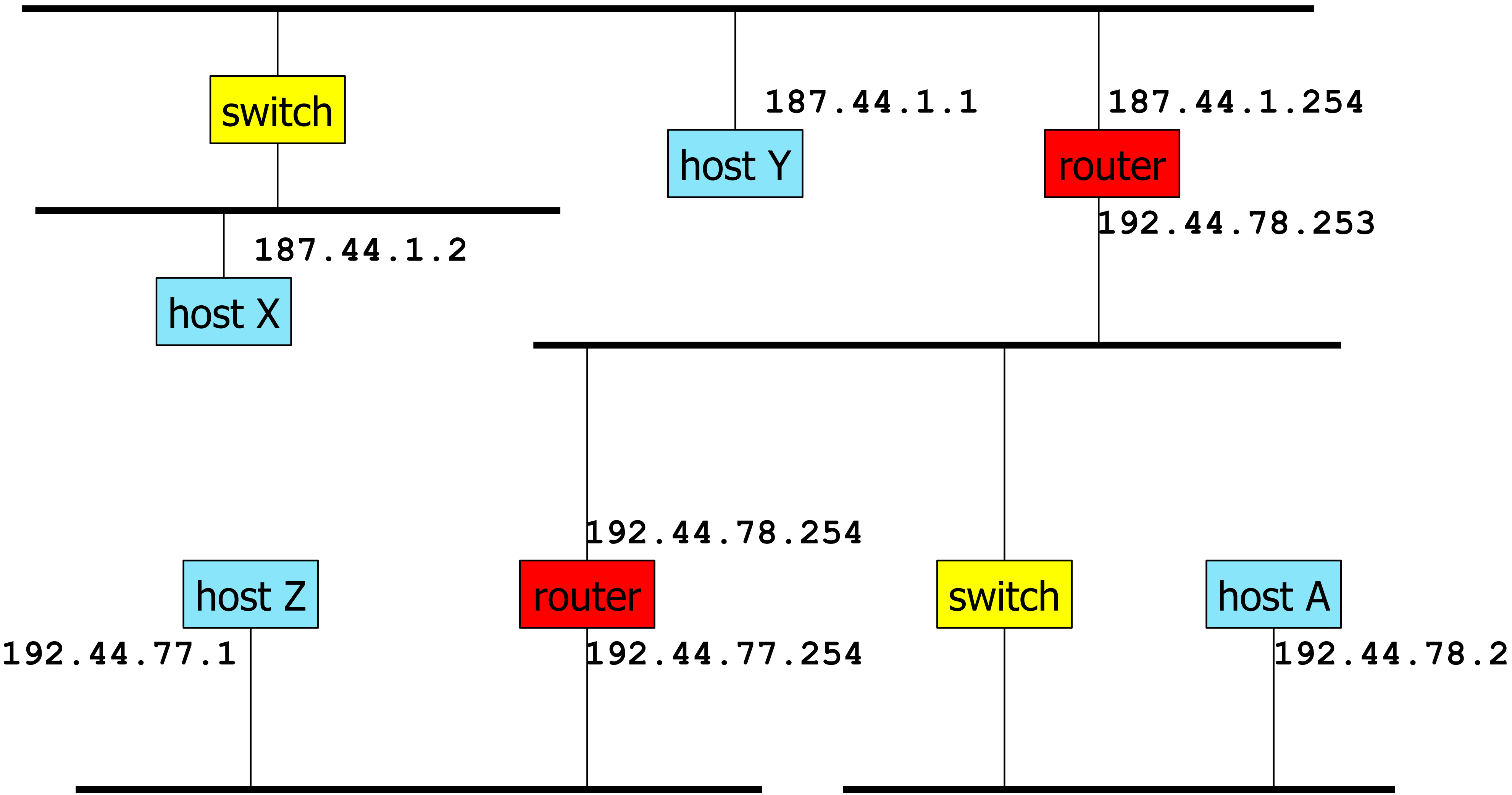
- Sovkom has received IP addresses 194.167.0.0 to 194.167.255.255 total: 2^{16} addr.
- **Java Business Solutions AG** has received IP addresses 194.167.41.0 to 194.167.41.255 total: 2^{8} addresses
- **Tango SA** has received IP addresses 194.167.42.0 to 194.167.43.255 total: 2^{9} addresses

Example



- Can host A have this address?

Example



- Host A is on subnetwork 192.44.78

IP Principles

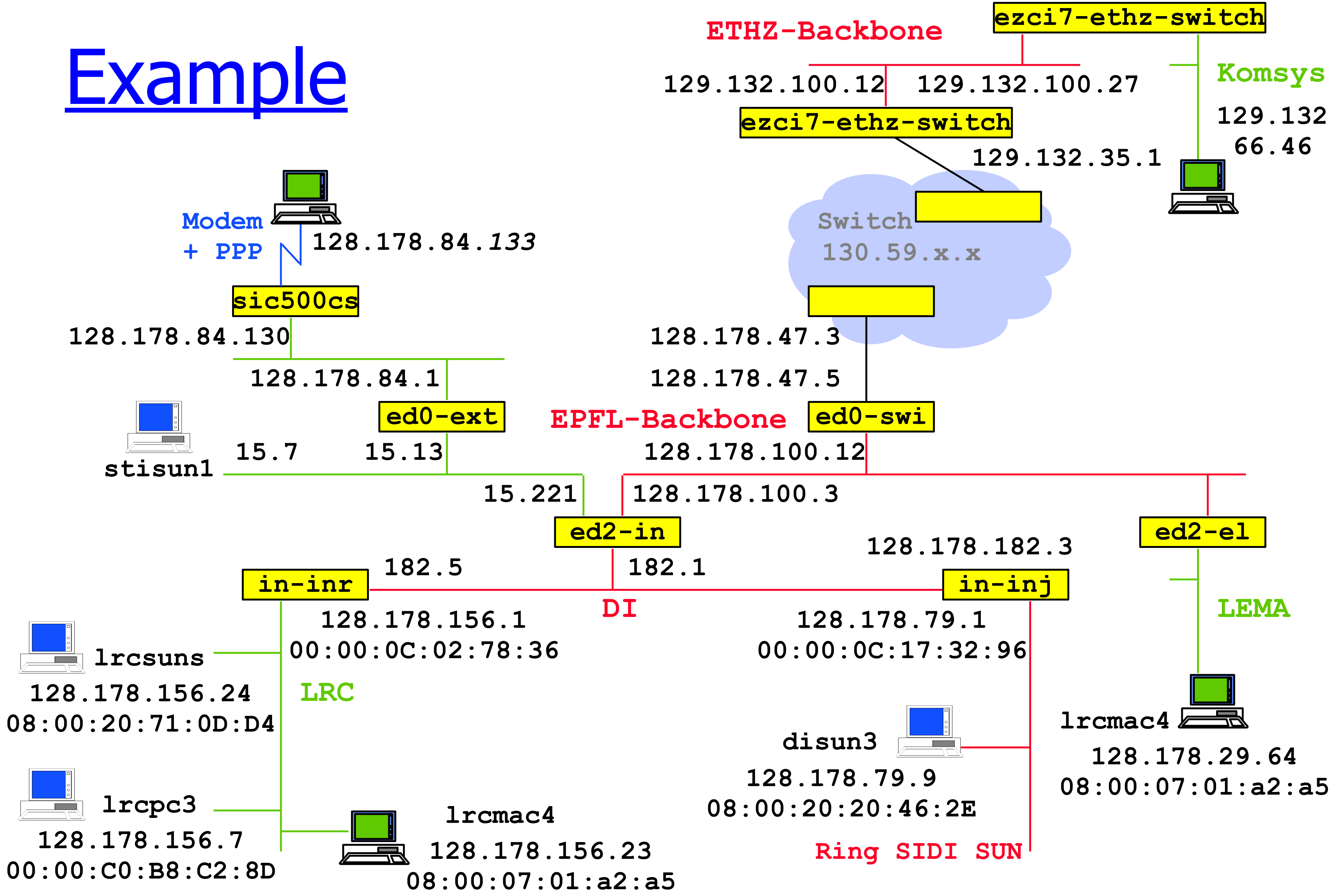
Homogeneous addressing

- an IP address is unique across the whole network (= the world in general)
- IP address is the address of the interface
- communication between IP hosts requires knowledge of IP addresses

Routing:

- inside a subnetwork: hosts communicate directly without routers
- between subnetworks: one or several routers are used
- a subnetwork = a collection of systems with a common prefix

Example


ETHZ-Backbone
ezci7-ethz-switch
Komsys
129.132.100.12
129.132.100.27
ezci7-ethz-switch
129.132
66.46
129.132.35.1
Modem
+ PPP
128.178.84.133
Switch
130.59.x.x
sic500cs
128.178.84.130
128.178.47.3
128.178.84.1
128.178.47.5
ed0-ext
EPFL-Backbone
ed0-swi
15.7
15.13
128.178.100.12
stisun1
15.221
128.178.100.3
ed2-in
ed2-el
128.178.182.3
182.5
182.1
in-inr
in-inj
DI
LEMA
128.178.156.1
00:00:0C:02:78:36
128.178.79.1
00:00:0C:17:32:96
lrcsuns
128.178.156.24
08:00:20:71:0D:D4
LRC
lrcmac4
disun3
128.178.29.64
08:00:07:01:a2:a5
128.178.79.9
08:00:20:20:46:2E
lrcpc3
lrcmac4
128.178.156.7
00:00:C0:B8:C2:8D
128.178.156.23
08:00:07:01:a2:a5
Ring SIDI SUN

IP packet forwarding algorithm

- Rule for sending packets (hosts, routers)
  - if the destination IP address has the same prefix as one of my interfaces, send directly to that interface
  - otherwise send to a router as given by the IP routing table

At `lrcsuns`: Next Hop Table

destination@	subnetMask	nextHop
DEFAULT		128.178.156.1

Physical Interface Tables

IP	subnetMask
128.178.156.24	255.255.255.0

At `in-inj`: Next Hop Table

destination@	subnetMask	nextHop
128.178.156.0 DEFAULT	255.255.255.0 0.0.0.0	128.178.182.5 128.178.182.1

Physical Interface Tables

IP	subnetMask
128.178.79.1 128.178.182.3	255.255.255.0 255.255.255.0

IP packet forwarding algorithm

destAddr = packet dest. address, destinationAddr = address in routing table

Case 1: a **host route** exists for destAddr

```
for every entry in routing table
    if (destinationAddr = destAddr)
    then send to nextHop IPaddr; leave
```

Case 2: destAddr is on a **directly connected network** (= on-link):

```
for every physical interface IP address A and subnet mask SM
    if(A & SM = destAddr & SM)
    then send directly to destAddr; leave
```

Case 3: a **network route** exists for destAddr

```
for every entry in routing table and subnet mask SM
    if (destinationAddr & SM = destAddr & SM)
    then send to nextHop IP addr; leave
```

Case 4: use **default route**

```
for every entry in routing table
    if (destinationAddr=DEFAULT)  then send to nextHop IPaddr; leave
```

Getting a datagram from source to dest.

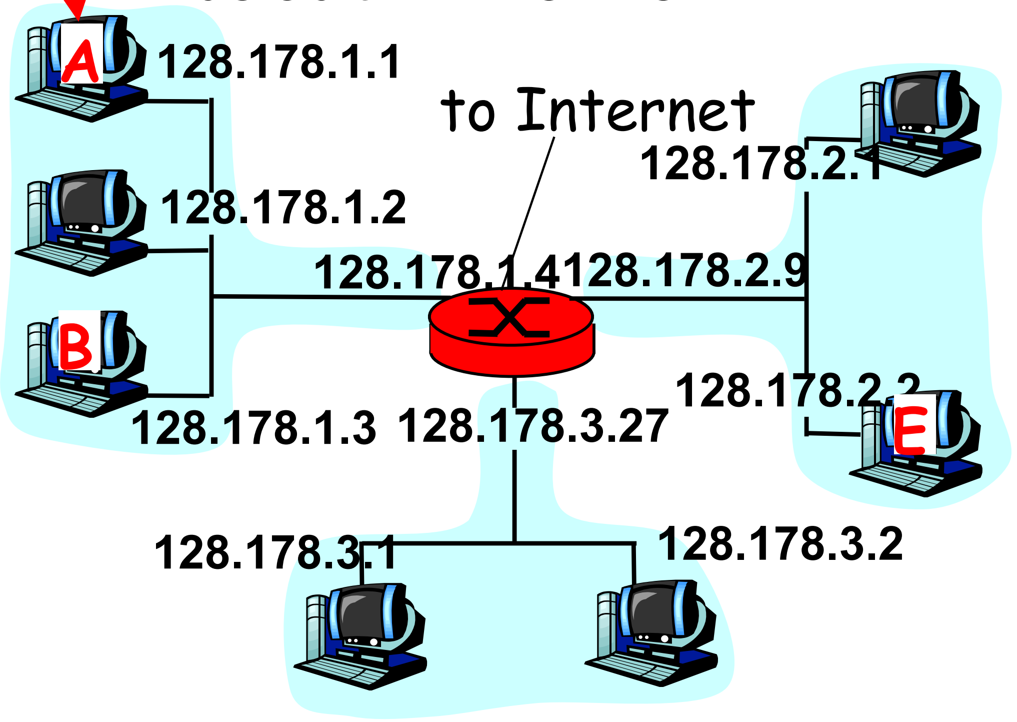
routing table in A

Dest. Net.	next router	Nhops
128.178.1		1
128.178.2	128.178.1.4	2
128.178.3	128.178.1.4	2
default	128.178.1.4	

IP datagram:

misc fields	source IP addr	dest IP addr	data

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

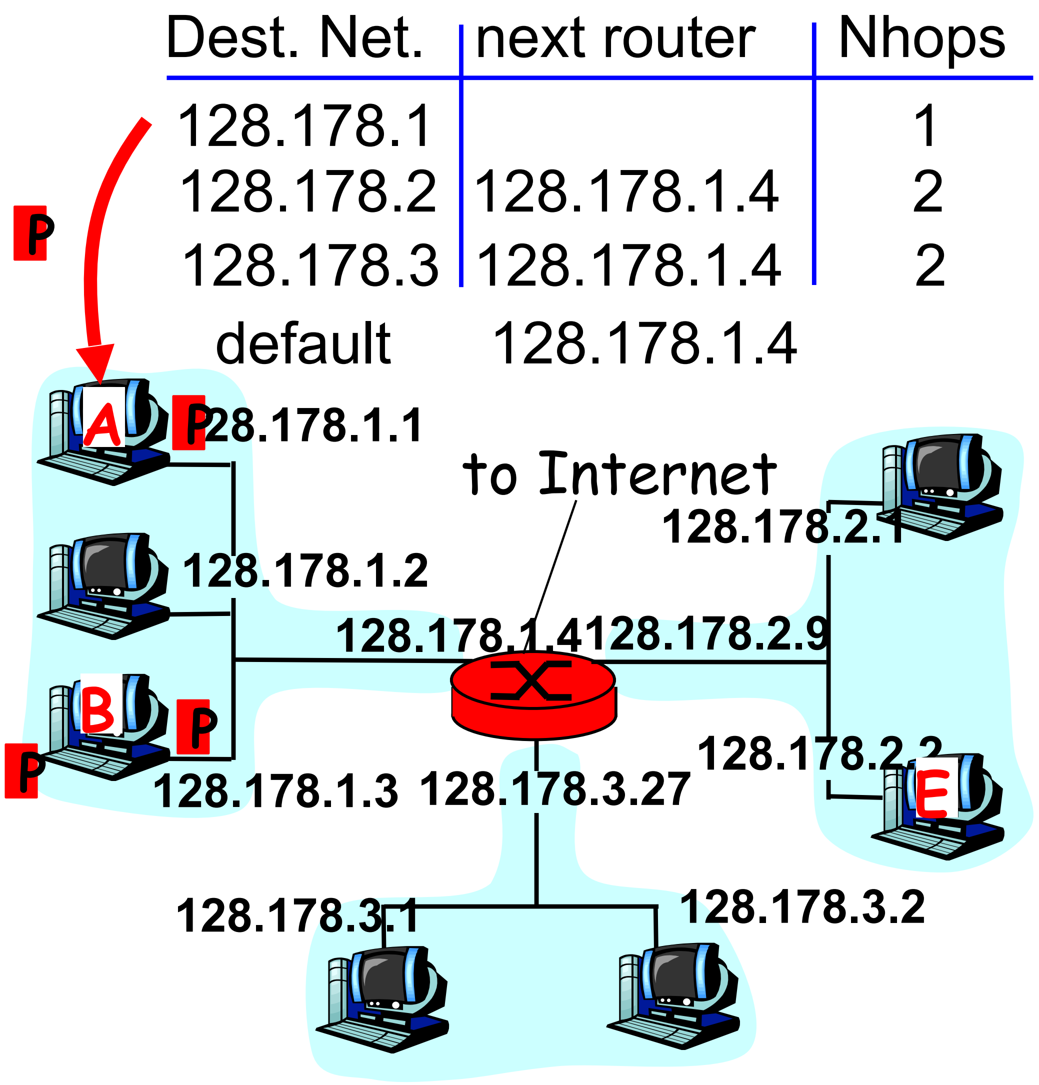


Getting a datagram from source to dest.: same subnetwork

misc fields	128.178.1.1	128.178.1.3	data

Starting at A, given IP datagram addressed to B:

- look up net. address of B
- 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



Dest. Net.	next router	Nhops
128.178.1		1
128.178.2	128.178.1.4	2
128.178.3	128.178.1.4	2
default	128.178.1.4	

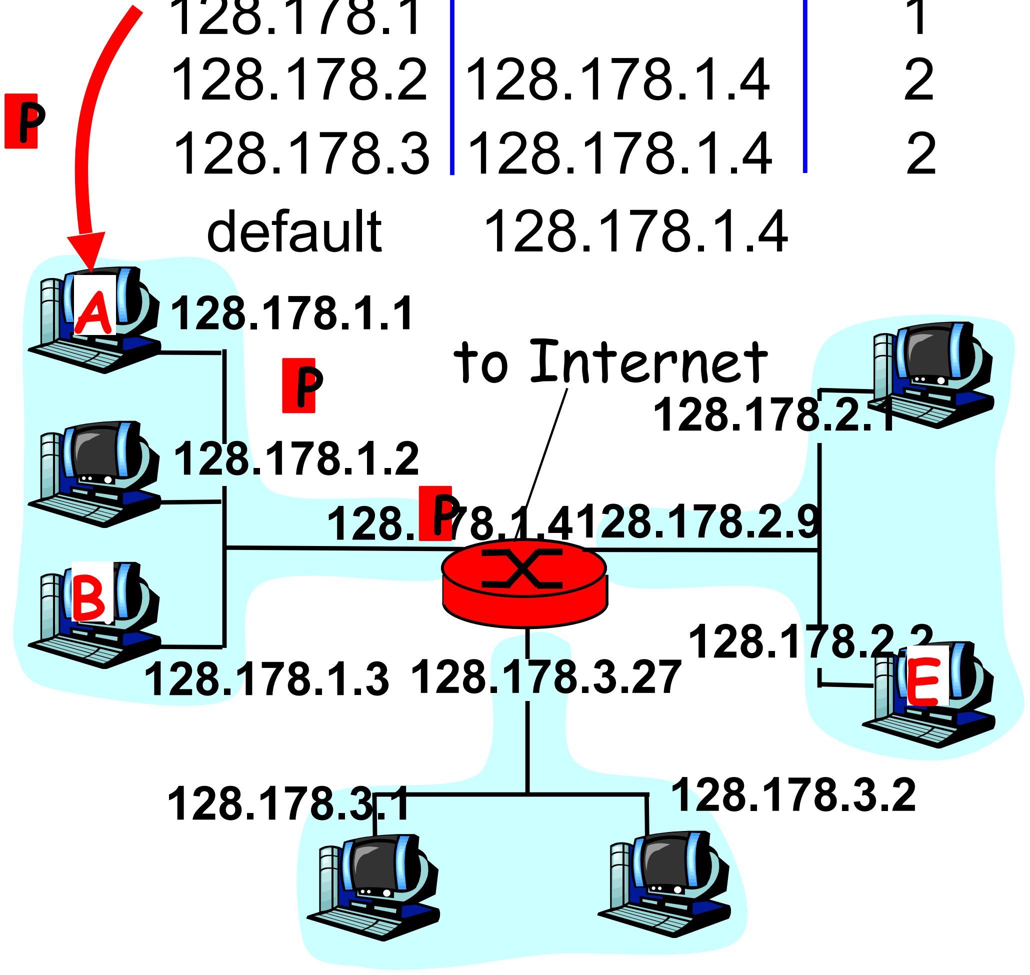
Getting a datagram from source to dest.: different subnetworks

misc fields	128.178.1.1	128.178.2.3	data

Starting at A, dest. E:

- look up network address of E
- E on different network
  - A, E not directly attached
- routing table: next hop router to E is 128.178.1.4
- link layer sends datagram to router 128.178.1.4 inside link-layer frame
- datagram arrives at 128.178.1.4
- continued…..

Dest. Net.	next router	Nhops
128.178.1		1
128.178.2	128.178.1.4	2
128.178.3	128.178.1.4	2
default	128.178.1.4	



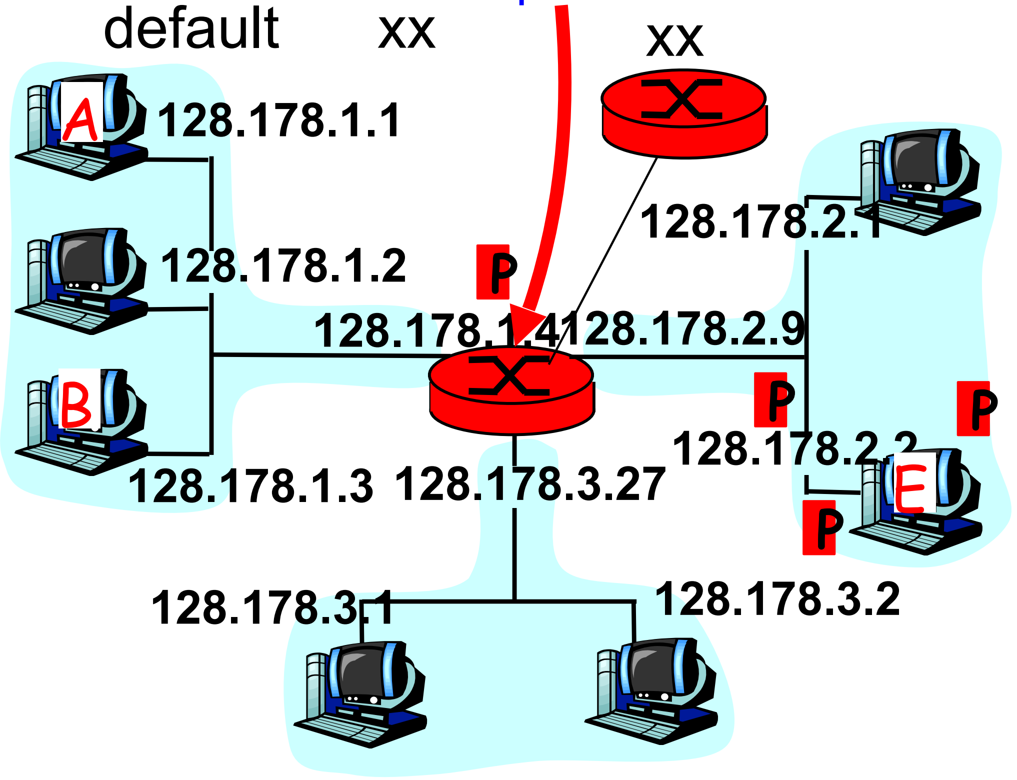
Getting a datagram from source to dest.: different subnetworks

misc fields	128.178.1.1	128.178.2.3	data

Dest. network	next router	Nhops	interface
128.178.1	-	1	128.178.1.4
128.178.2	-	1	128.178.2.9
128.178.3	-	1	128.178.3.27
default	xx	xx	

Arriving at 128.178.1.4, destined for 128.178.2.3

- look up network address of E
- E on same network as router's interface 128.178.2.9
  - router, E directly attached
- link layer sends datagram to 128.178.2.2 inside link-layer frame via interface 128.178.2.9
- datagram arrives at 128.178.2.3!!! (hooray!)



IP packet format

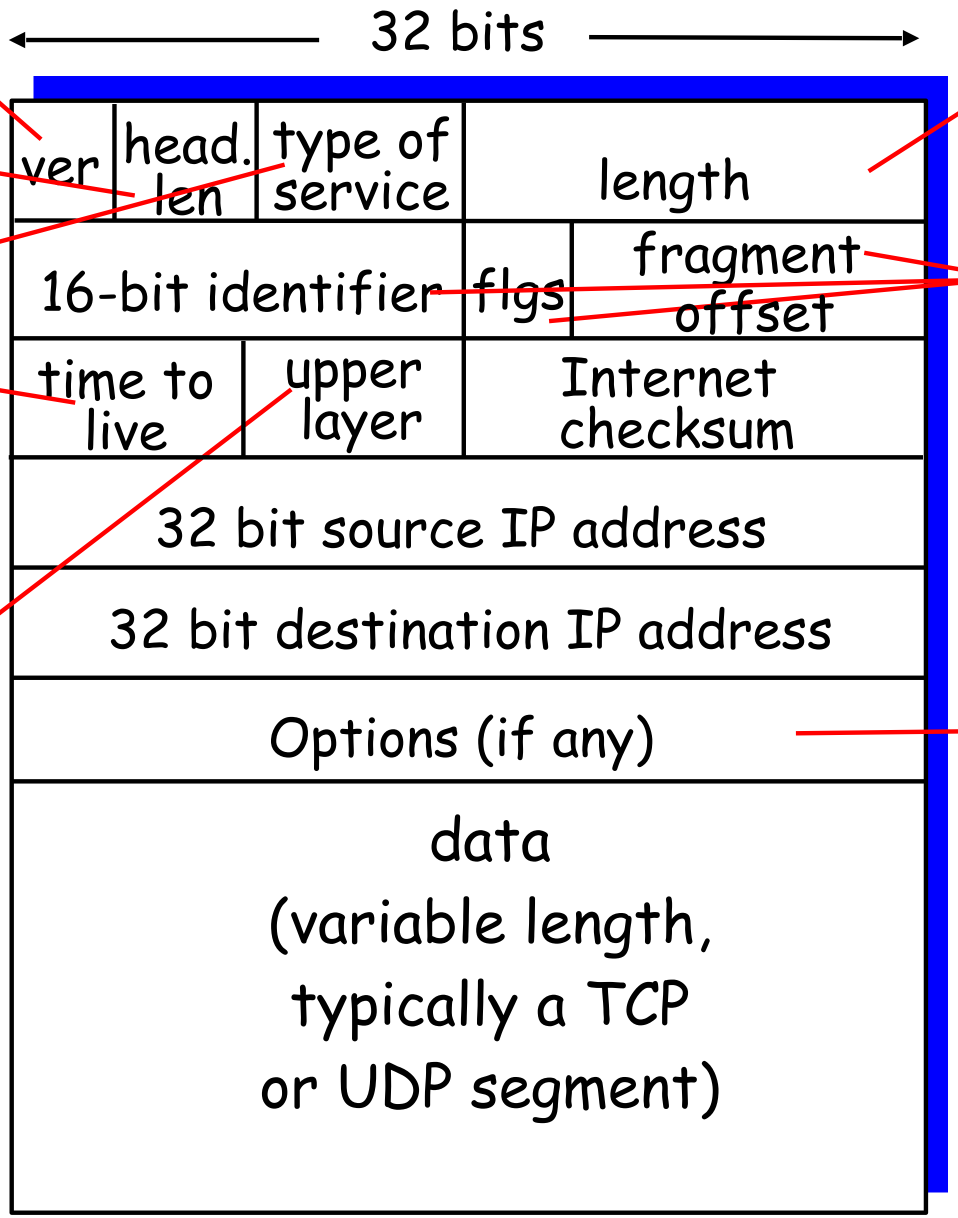
IP protocol version number

header length (bytes)

"type" of data

max number remaining hops (decremented at each router)

upper layer protocol to deliver payload to



32 bits

ver | head. len | type of service | length

16-bit identifier | flgs | fragment offset

time to live | upper layer | Internet checksum

32 bit source IP address

32 bit destination IP address

Options (if any)

data (variable length, typically a TCP or UDP segment)

total datagram length (bytes)

for fragmentation/ reassembly

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

IP header

- Version
  - IPv4, futur IPv6
- Header size
  - options - variable size
  - in 32 bit words
- Type of service
  - priority : 0 - normal, 7 - control packets
  - short delay (telnet), high throughput (ftp), high reliability (SNMP), low cost (NNTP)
- Redefined in DiffServ (Differentiated Services)
  - 1 byte codepoint determining QoS class
    - Expedited Forwarding (EF) - minimize delay and jitter
    - Assured Forwarding (AF) - four classes and three drop-precedences (12 codepoints)

IP header

- Packet size
  - in bytes including header
  - in bytes including header
  - <= 64 Kbytes; limited in practice by link-level MTU (Maximum Transmission Unit)
  - every subnet should forward packets of 576 = 512 + 64 bytes
- Id
  - unique identifier for re-assembling
- Flags
  - M : more ; set in fragments
  - F : prohibits fragmentation

IP header

- Offset
  - position of a fragment in multiples of 8 bytes
- TTL (Time-to-live)
  - in secondes
  - now: number of hops
  - router : --, if 0, drop (send ICMP packet to source)
- Protocol
  - identifier of protocol (1 - ICMP, 6 - TCP, 17 - UDP)
- Checksum
  - only on the header

IP header

- Options
  - strict source routing
    - all routers
  - loose source routing
    - some routers
  - record route
  - timestamp route
  - router alert
    - used by IGMP or RSVP for processing a packet

LAN Addresses and ARP

32-bit IP address:

- network-layer address
- used to get datagram to destination network (recall IP network definition)

LAN (or MAC or physical) address:

- used to get datagram from one interface to another physically-connected interface (same network)
- 48 bit MAC address (for most LANs) burned in the adapter ROM

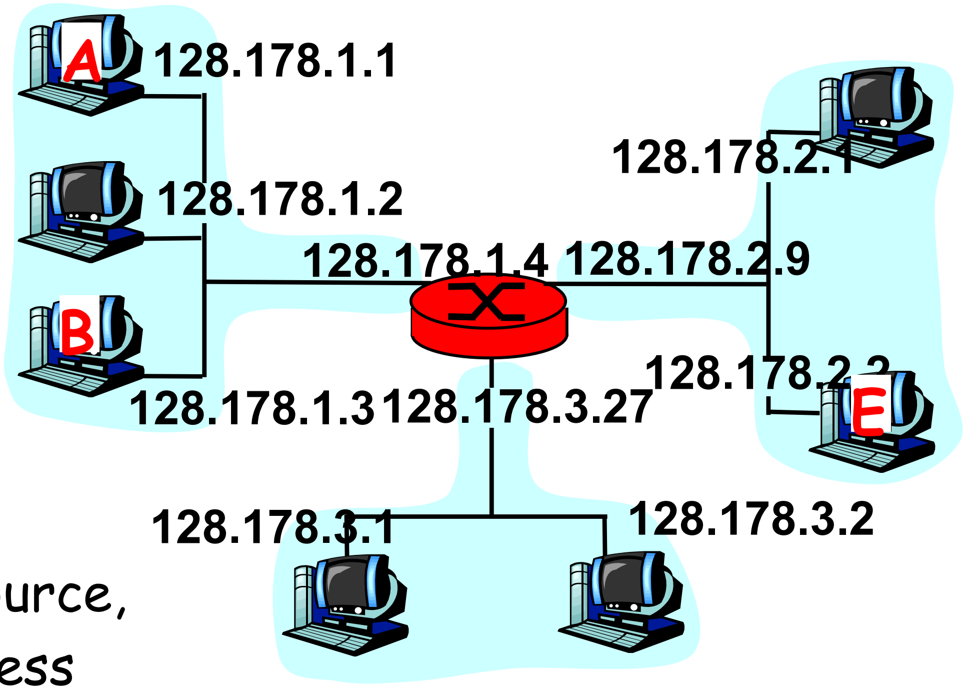
Why different addresses at IP and MAC?

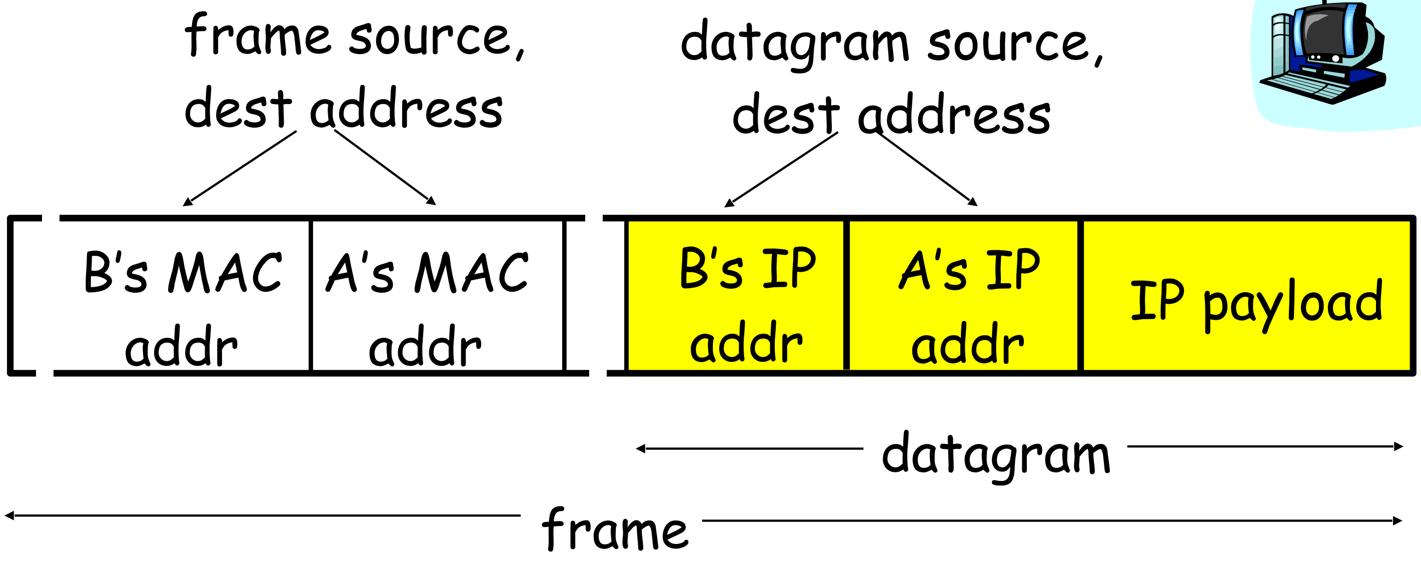
- LANs not only for IP (LAN addresses are neutral)
- if IP addresses used, they should be stored in a RAM and reconfigured when host moves
- independency of layers

MAC Address resolution

Starting at A, given IP datagram addressed to B:

- look up net. address of B, find B on same net. as A
- link layer send datagram to B inside link-layer frame




frame source, dest address

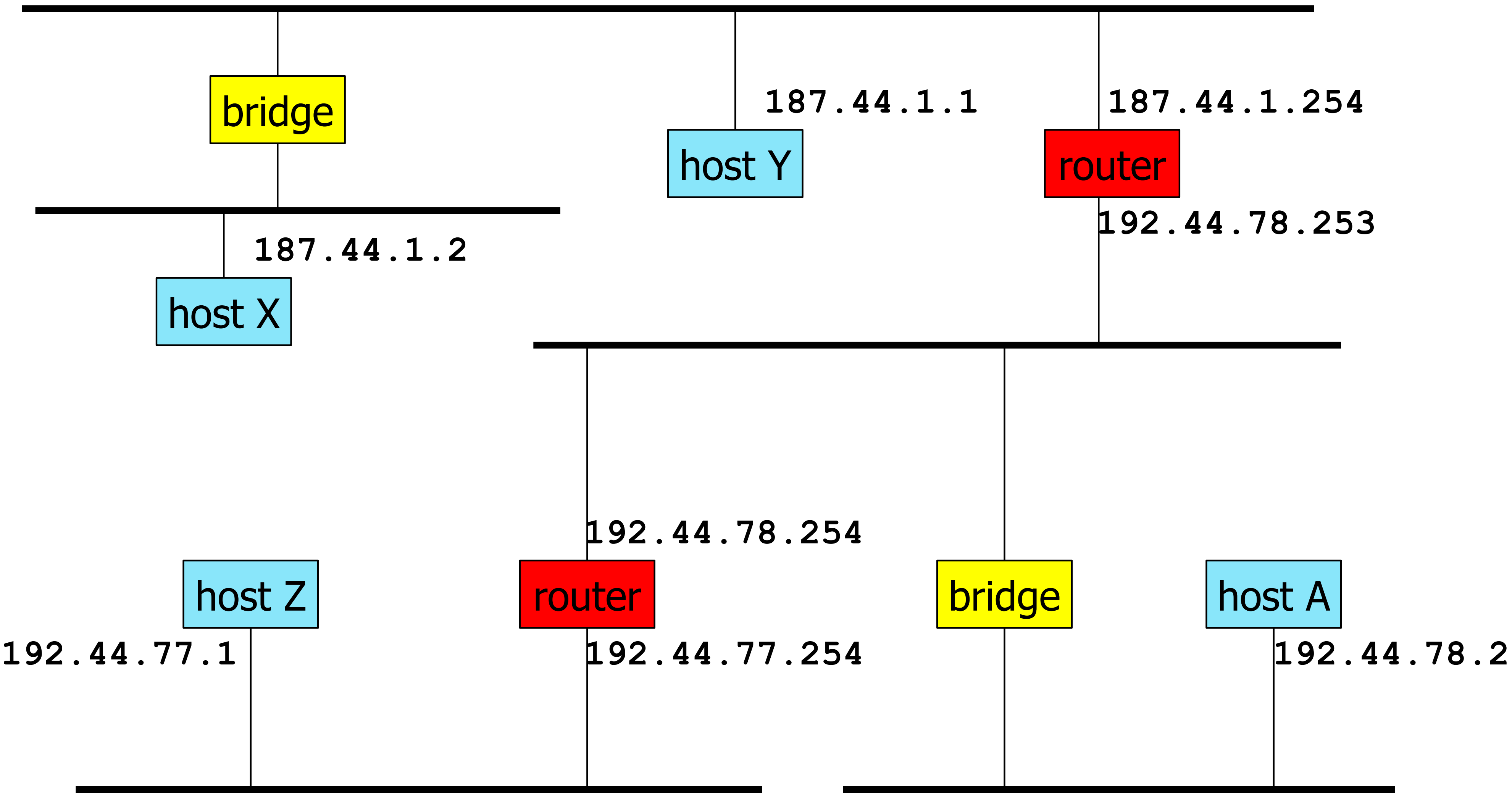
datagram source, dest address

B's MAC addr | A's MAC addr | B's IP addr | A's IP addr | IP payload

datagram

frame

Example (assuming /24)



- Host A is on subnetwork `192.44.78`

Packet delivery

Packet sent by `187.44.1.2 to 187.44.1.1`

MAC-host-Y	MAC-host-X	187.44.1.1	187.44.1.2	payload
Ethernet header		IP header		

X needs to know MAC address of Y (ARP)

Packet sent by `187.44.1.2 to 192.44.78.2`

MAC-router	MAC-host-X	192.44.78.2	187.44.1.2	payload
Ethernet header		IP header		

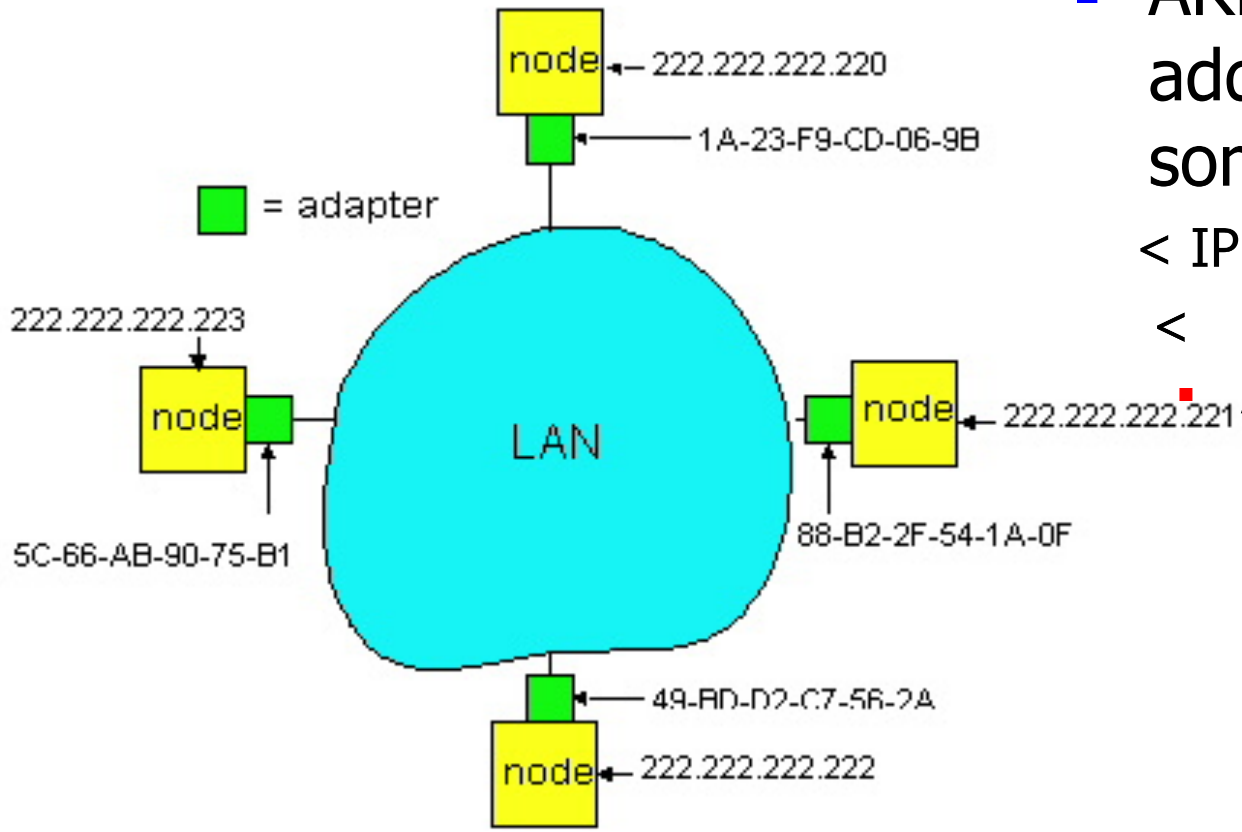
MAC-host-A	MAC-router	192.44.78.2	187.44.1.2	payload
Ethernet header		IP header		

X needs to know MAC address of router (X knows the IP address of router - configuration)

Router needs to know MAC address of A

ARP: Address Resolution Protocol

ARP is used to determine the MAC address of B given B's IP address



- Each IP node (Host, Router) on LAN implements ARP protocol and has ARP table
- ARP Table: IP/MAC address mappings for some LAN nodes

  < IP address; MAC address>

  < ……………………….. >

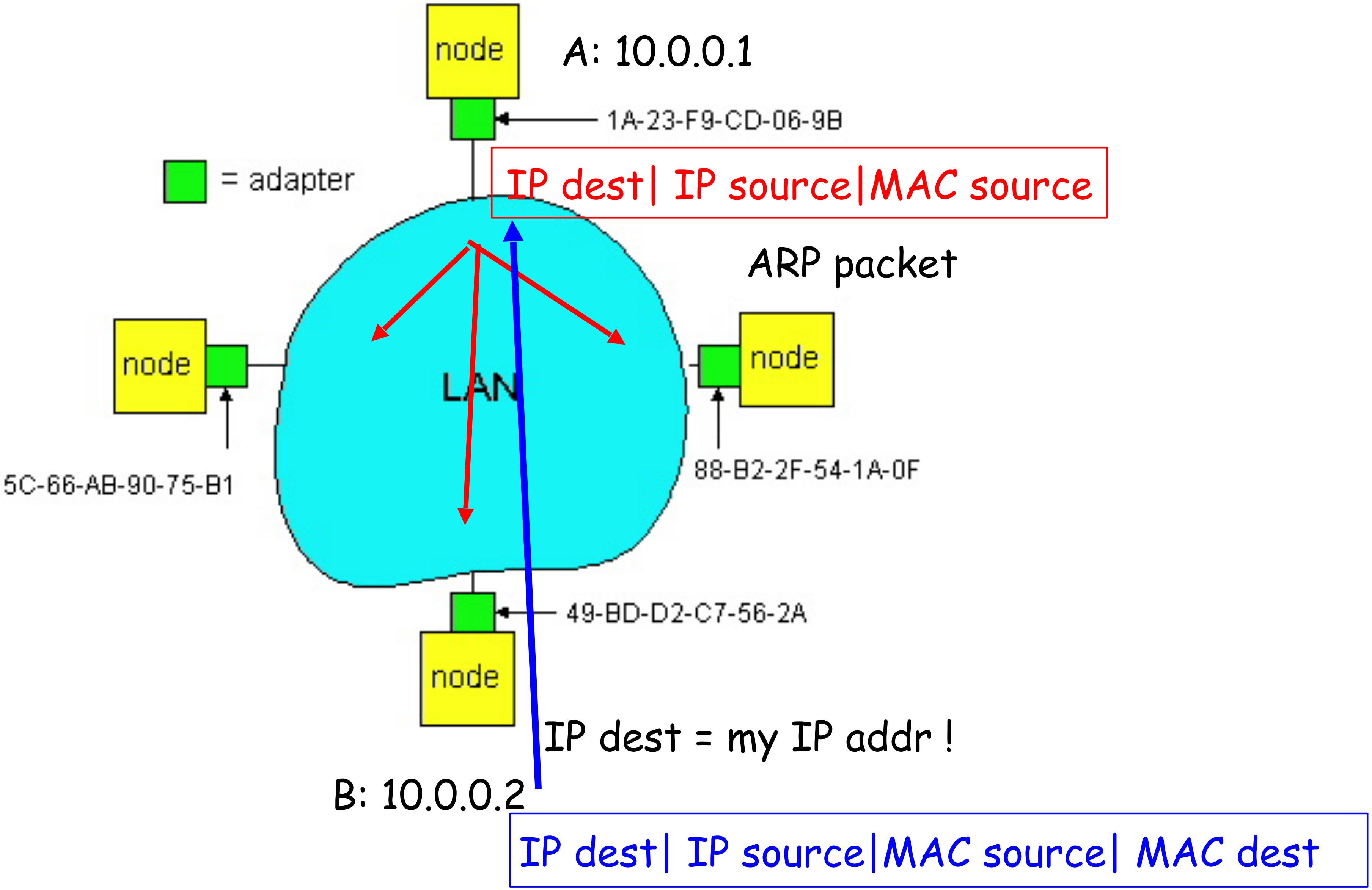
  - ARP table is a cache: after an interval (typically **20 min**) the address mapping will be forgotten

ARP protocol

- A knows B's IP address, wants to learn physical address of B
- A broadcasts ARP query pkt, containing B's IP address
    - all machines on LAN receive ARP query
- B receives ARP packet, replies to A with its (B's) physical layer address
- A caches (saves) IP-to-physical address pairs until information becomes old (times out)
    - soft state: information that times out (goes away) unless refreshed

ARP protocol

Cache on A:	IP address	MAC address	TTL
	10.0.0.2	49:BD:D2:07:56:2A	00:12:45



IP dest| IP source|MAC source

ARP packet

IP dest = my IP addr !

IP dest| IP source|MAC source| MAC dest

ARP frame

- Request (broadcast)

sender Ethernet address
sender IP address
target Ethernet address ???
target IP address

- Reply (unicast)

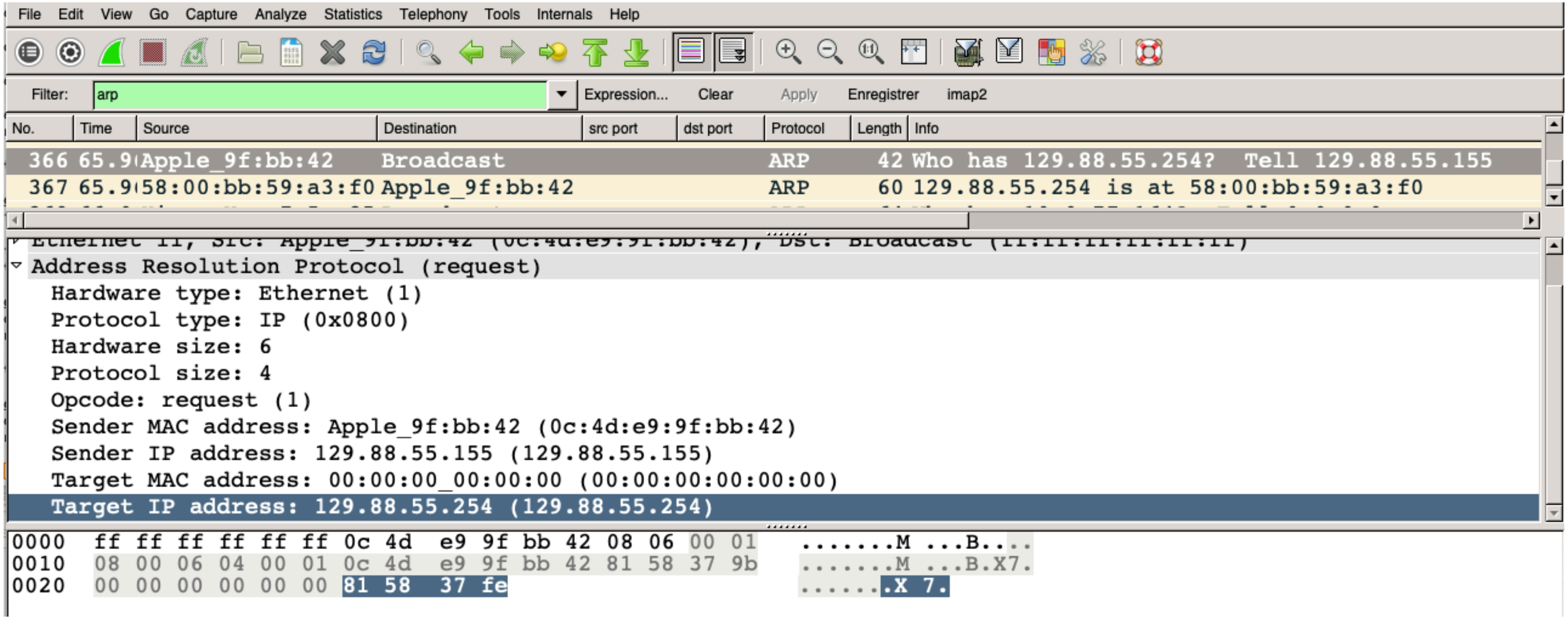
sender Ethernet address
sender IP address
target Ethernet address
target IP address

- Cache: net.link.ether.inet.max_age: 1200 (sysctl on mac OS)

ARP frames

- Request (**broadcast**)
  sender Ethernet address
  sender IP address
  target Ethernet address ???
  target IP address

- Reply (**unicast**)
  sender Ethernet address
  sender IP address
  target Ethernet address
  target IP address



Filter: arp

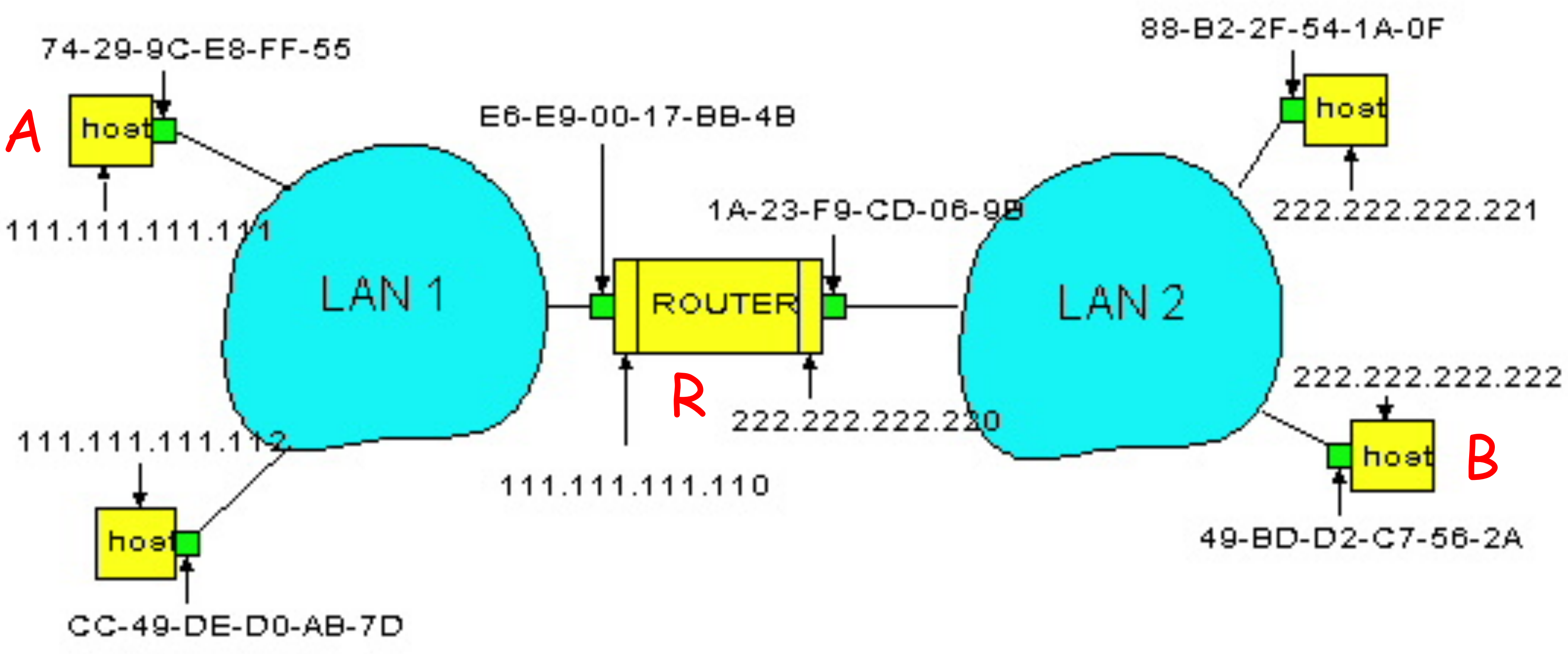
```
No.  Time   Source             Destination      src port  dst port  Protocol  Length  Info
366  65.9   Apple_9f:bb:42     Broadcast                            ARP       42      Who has 129.88.55.254?  Tell 129.88.55.155
367  65.9   58:00:bb:59:a3:f0  Apple_9f:bb:42                       ARP       60      129.88.55.254 is at 58:00:bb:59:a3:f0
```

```
▷ Ethernet II, Src: Apple_9f:bb:42 (0c:4d:e9:9f:bb:42), Dst: Broadcast (ff:ff:ff:ff:ff:ff)
▽ Address Resolution Protocol (request)
    Hardware type: Ethernet (1)
    Protocol type: IP (0x0800)
    Hardware size: 6
    Protocol size: 4
    Opcode: request (1)
    Sender MAC address: Apple_9f:bb:42 (0c:4d:e9:9f:bb:42)
    Sender IP address: 129.88.55.155 (129.88.55.155)
    Target MAC address: 00:00:00_00:00:00 (00:00:00:00:00:00)
    Target IP address: 129.88.55.254 (129.88.55.254)
```

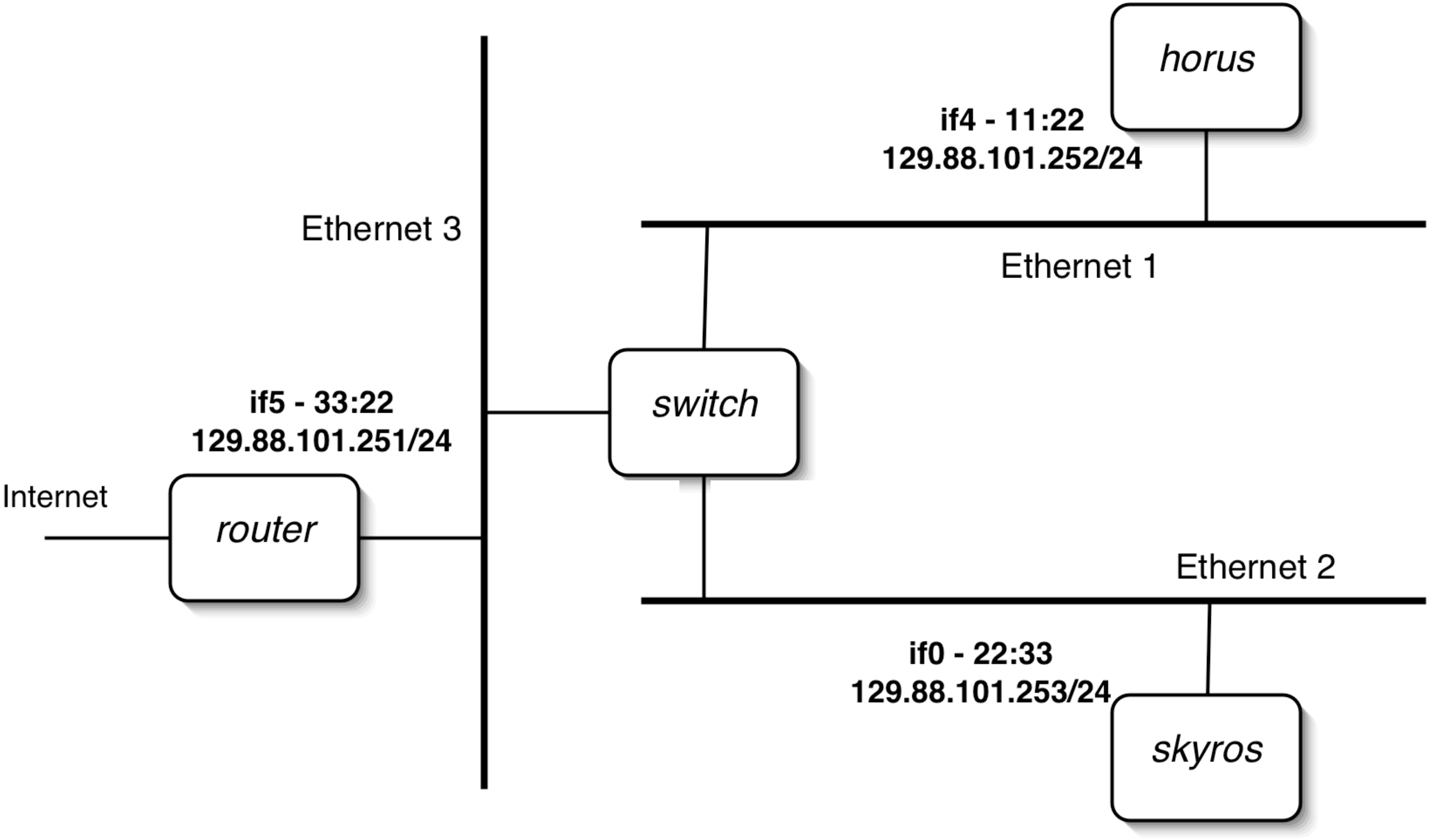
```
0000  ff ff ff ff ff ff 0c 4d  e9 9f bb 42 08 06 00 01   .......M ...B....
0010  08 00 06 04 00 01 0c 4d  e9 9f bb 42 81 58 37 9b   .......M ...B.X7.
0020  00 00 00 00 00 00 81 58  37 fe                     .......X 7.
```

Routing to another LAN

walkthrough: routing from A to B via R



- In routing table at source Host, find router 111.111.111.110
- In ARP table at source, find MAC address E6-E9-00-17-BB-4B, etc



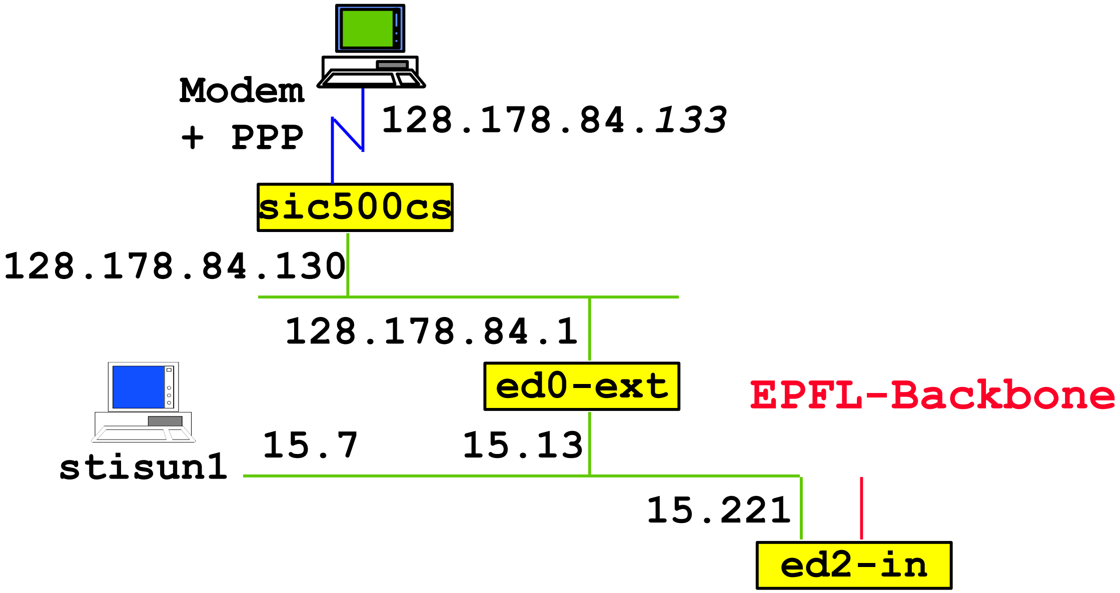
- I'm executing the following commands on **`skyros`**:

```
ping 129.88.101.252
ping 195.221.19.1
```

- What will you observe on the LANs?

Proxy ARP

- Proxy ARP: a host answers ARP requests on behalf of others
  - example: **`sic500cs`** for PPP connected computers
  - manual configuration of **`sic500cs`**



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:
  - 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	router advertisement
10	0	router discovery
11	0	TTL expired
12	0	bad IP header

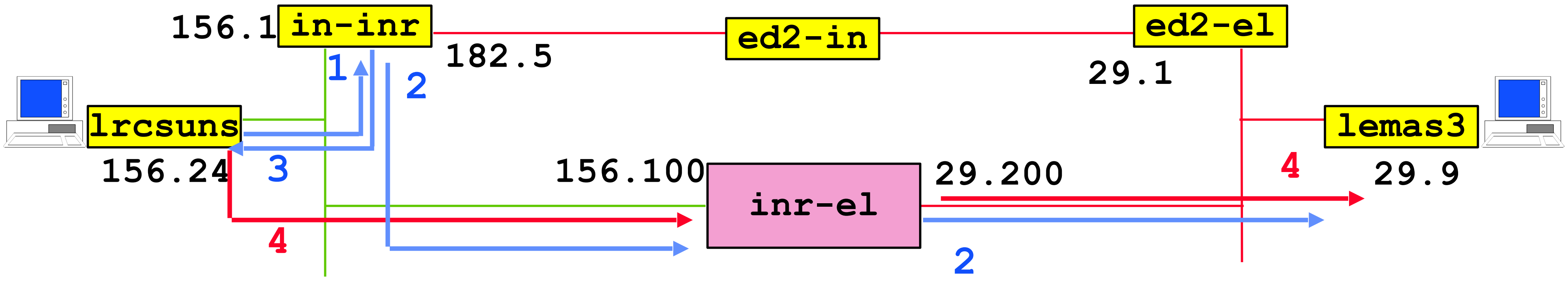
ICMP Redirect

- Sent by router to source host to inform source that destination is directly connected
  - host updates the routing table
  - ICMP redirect can be used to update the router table (eg. **`in-inj`** route to LRC?)

ICMP Redirect Format

```
/                                                               /
|                  IP datagram header  (prot = ICMP)            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|   Type=5        |     code        |           checksum        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|              Router IP address that should be preferred       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|        IP header plus 8 bytes of original datagram data       |
/                                                               /
```

ICMP Redirect example



	dest IP addr	srce IP addr	prot	data part
1:	128.178.29.9	128.178.156.24	udp	xxxxxxx
2:	128.178.29.9	128.178.156.24	udp	xxxxxxx
3:	128.178.156.24	128.178.156.1	icmp	type=redir code=host cksum 128.178.156.100 xxxxxxx (28 bytes of 1)
4:	128.178.29.9	128.178.156.24	udp	....

ICMP Redirect example (cont’d)

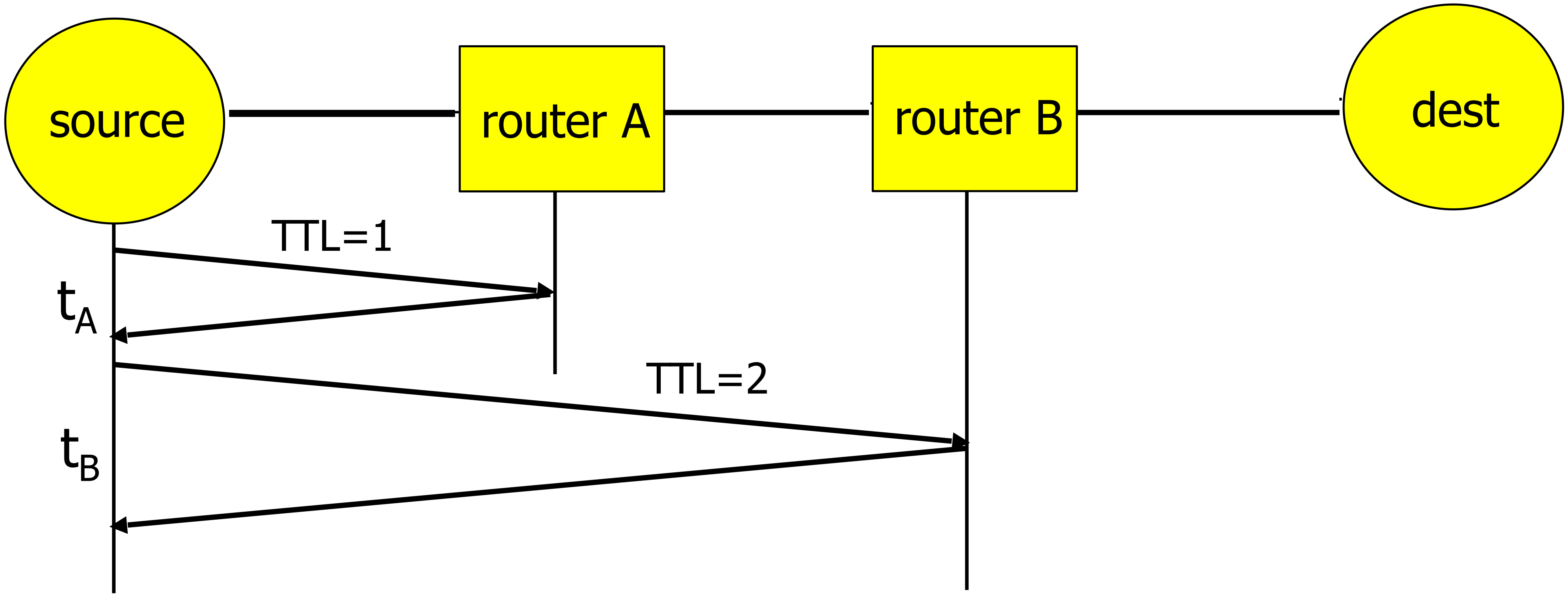
After 4

```
lrcsuns$ netstat -nr
Routing Table:
  Destination           Gateway           Flags  Ref   Use   Interface
-------------------- -------------------- ----- ----- ------ ---------
127.0.0.1            127.0.0.1             UH        0  11239  lo0
128.178.29.9         128.178.156.100       UGHD      0     19
128.178.156.0        128.178.156.24        U         3  38896  le0
224.0.0.0            128.178.156.24        U         3      0  le0
default              128.178.156.1         UG        0  85883
```

Tools that use ICMP

- ping
  - ICMP Echo request
  - wait for Echo reply
  - measure RTT
- traceroute
  - IP packet with TTL = 1
  - wait for ICMP *TTL expired*
  - IP packet with TTL = 2
  - wait for ICMP *TTL expired*
  - ...

Traceroute



Summary

- The network layer transports packets from a sending host to the receiver host.
- Internet network layer
  - connectionless
  - best-effort
- Main components:
  - addressing
  - packet forwarding
  - routing protocols and routers (or how a router works)
- Routing protocols will be seen later