Addressing and Routing

Andrew Scott

a.scott@lancaster.ac.uk

Physical/ Hardware Addresses

- Aka MAC* or link(-layer) address
  - Can only 'talk' to things on same link
- 'Unique' ID given to every network interface 'card' (NIC) on manufacture
  - Generally written in form: f0:1c:2d:f5:a5:41
- Flat addressing scheme
  - IDs allocated sequentially as devices manufactured
  - Cannot tell anything about ID2 by knowing of ID1

* From *Medium Access Control* function in networks

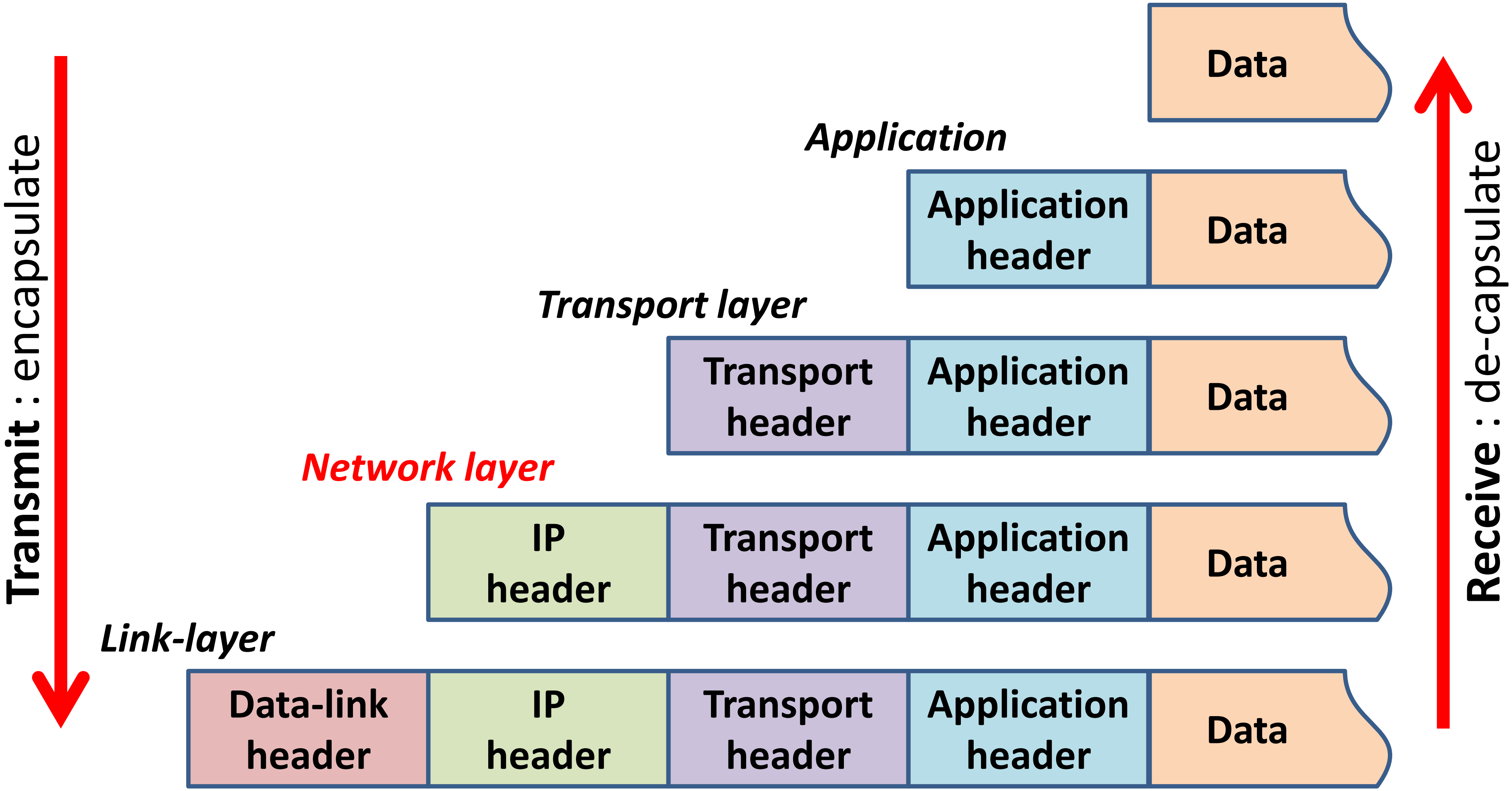
Internet Protocol

- Connectionless **network protocol**
  - No attempt to build path prior to transmission

- Best-effort – *packets may be*:
  - Lost
  - Delivered out of order
  - Duplicated
  - Delayed

Protocol Encapsulation

- IP packets sit within link-layer frames

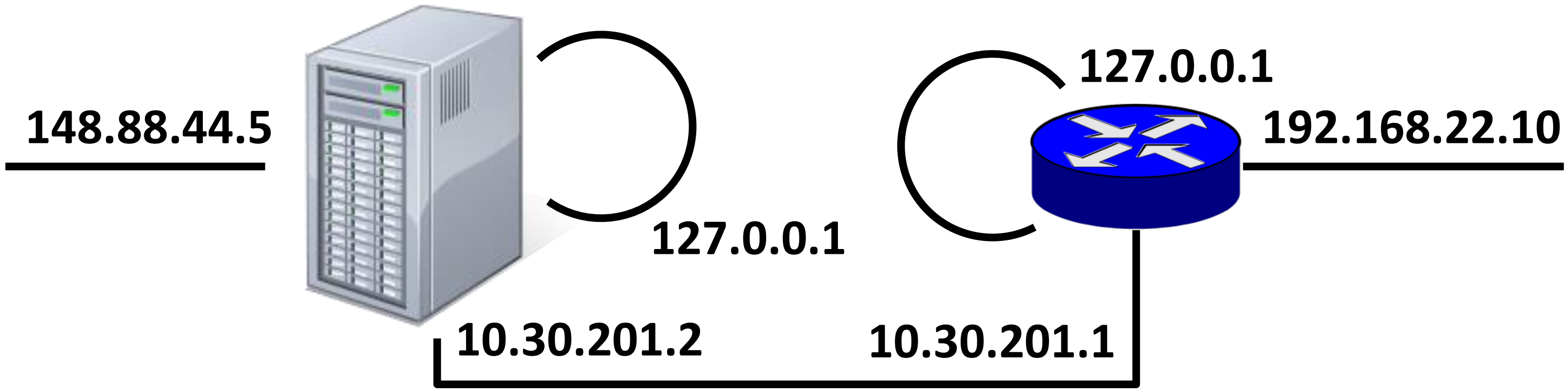


IP Addresses

- Hierarchical
  - Not flat like Ethernet/ MAC address
- Topological
  - Reflects network structure
  - Not geographic
    - Though topology might be constrained by geography

IPv4 Addresses

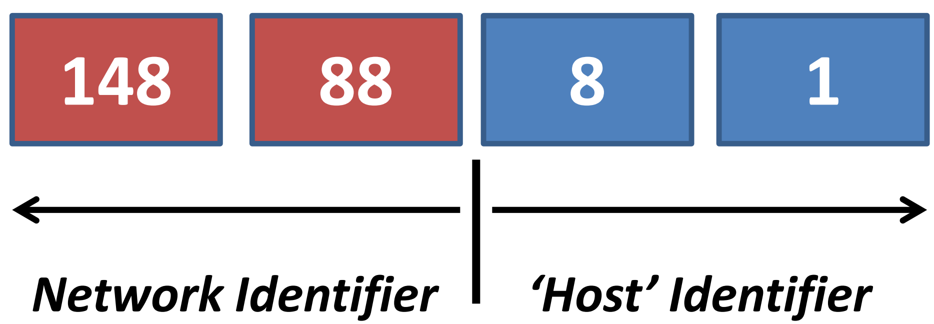
- 32 bit/ 4 byte identifier
  - Identify network interfaces **not** *hosts/ devices* *
  - A device will have multiple addresses
    - At least one per virtual or physical network interface
      - Notice all devices use same loopback address: 127.0.0.1



** Physical interfaces can have multiple IP addresses but not generally useful*

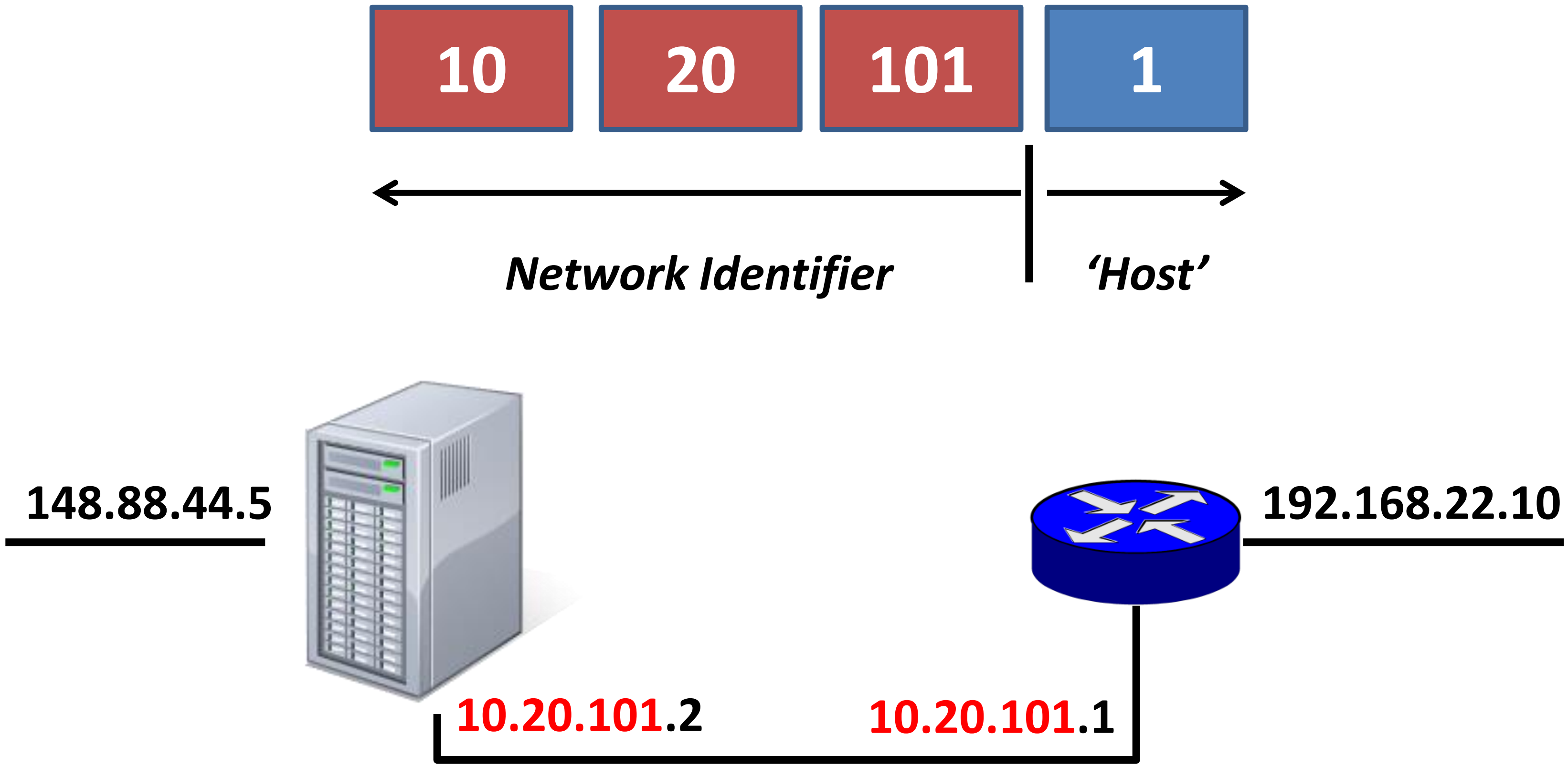
IPv4 Addresses

- Typically written as 4 bytes, e.g., 148.88.8.1
  - Often treated as a 32 bit *long*, i.e., 0x94580801
- Split into network and host parts
  - Split can be made at different positions



IPv4 Addresses

- Interfaces on same link share network part



Private Addresses

- Available for (organisation) internal/ test networks

- **Must not be made externally visible**

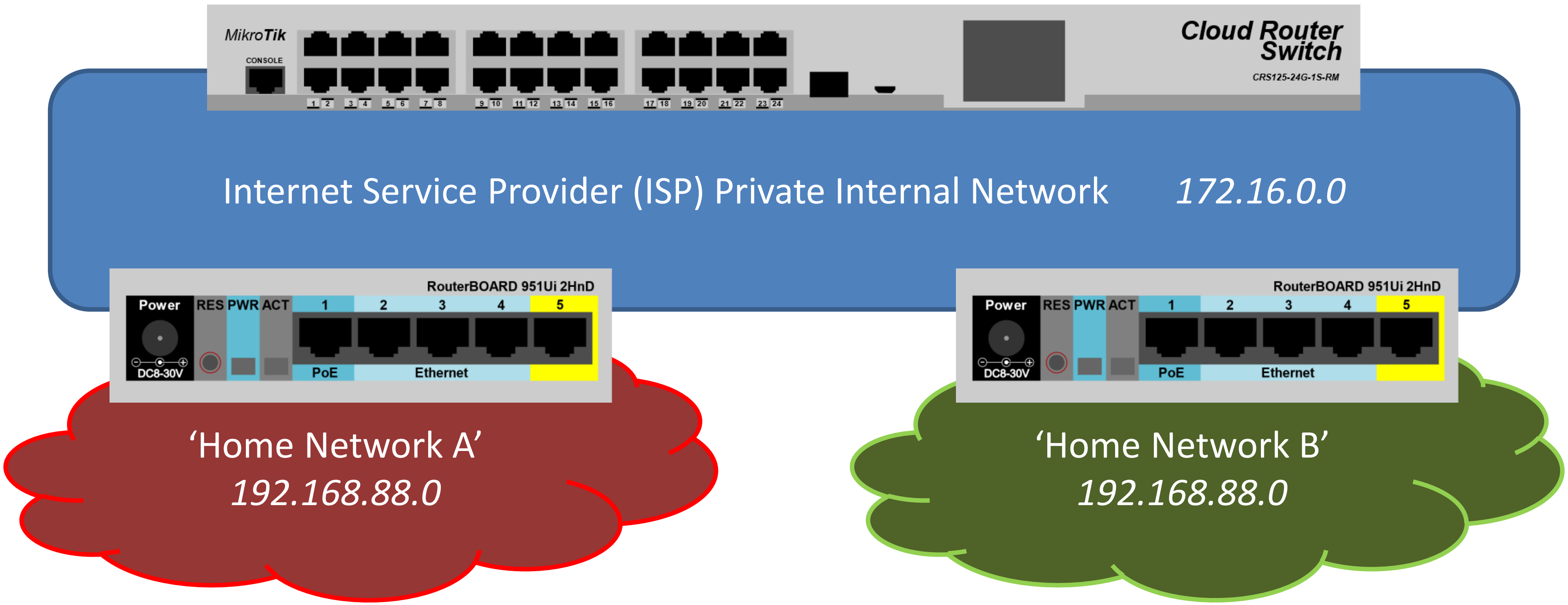
10.0.0.0	--	10.255.255.255
172.16.0.0	--	172.31.255.255
192.168.0.0	--	192.168.255.255

Free for use, but check your organisation isn't already using part of range, for example, University makes extensive use of 10.xxx addresses

RFC1918

Private Addresses

- Allow autonomy from any numbering authority
  - Note that private addresses are not globally unique
    - Therefore ‘meaningless’ outside of organisation



Special Addresses

- Multicast 224.0.0.0 – 239.255.255.255
  - Packets delivered to a group of ‘interested’ devices
  - Hosts subscribe to group in order to receive packets
    - Can be useful for media or data distribution services

RFC5771

- Broadcast 255.255.255.255
  - For delivery to all devices on local network

RFC919

IPv6 Address Representation

- Addresses 128 bits long
  - Prefix/ length notation prefix ***/n***

- Written in hexadecimal, in following format

  XXXX : XXXX : XXXX : XXXX : XXXX : XXXX : XXXX : XXXX/ *n*

- Leading zeros in group can be omitted, for example

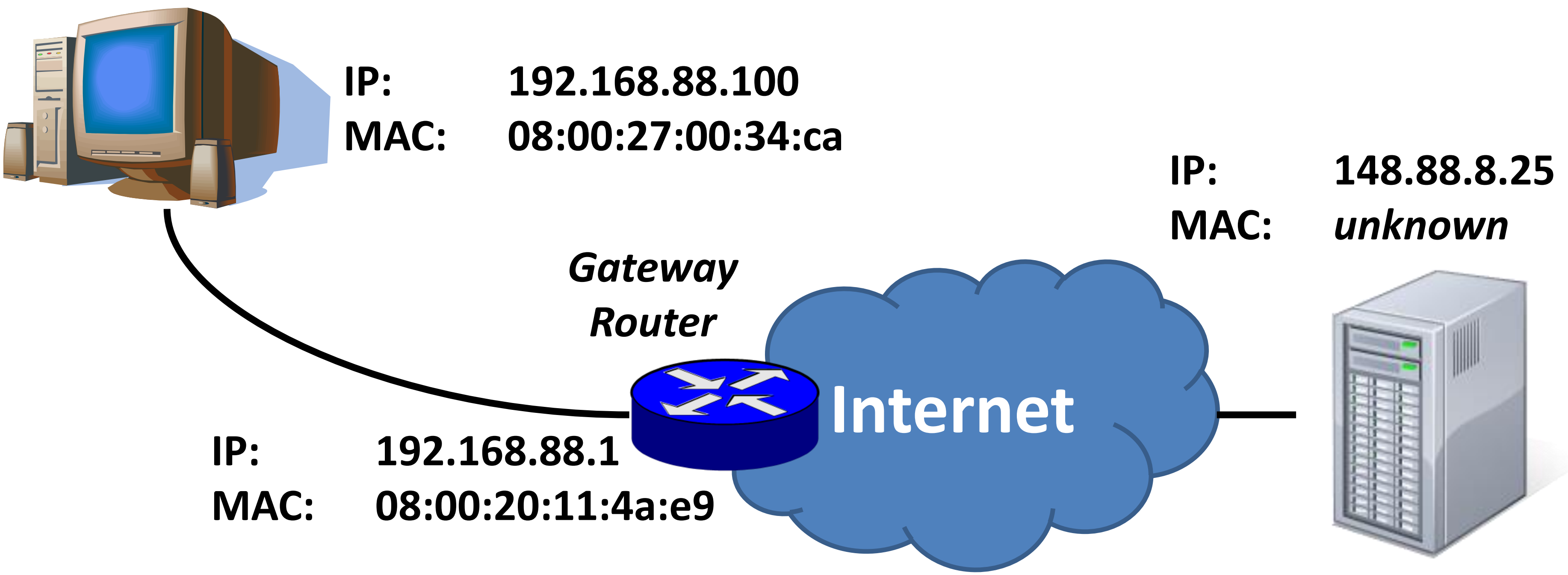
  1080:0:0:0:8:800:200C:417A/64

- At most one set of contiguous zeros can be dropped

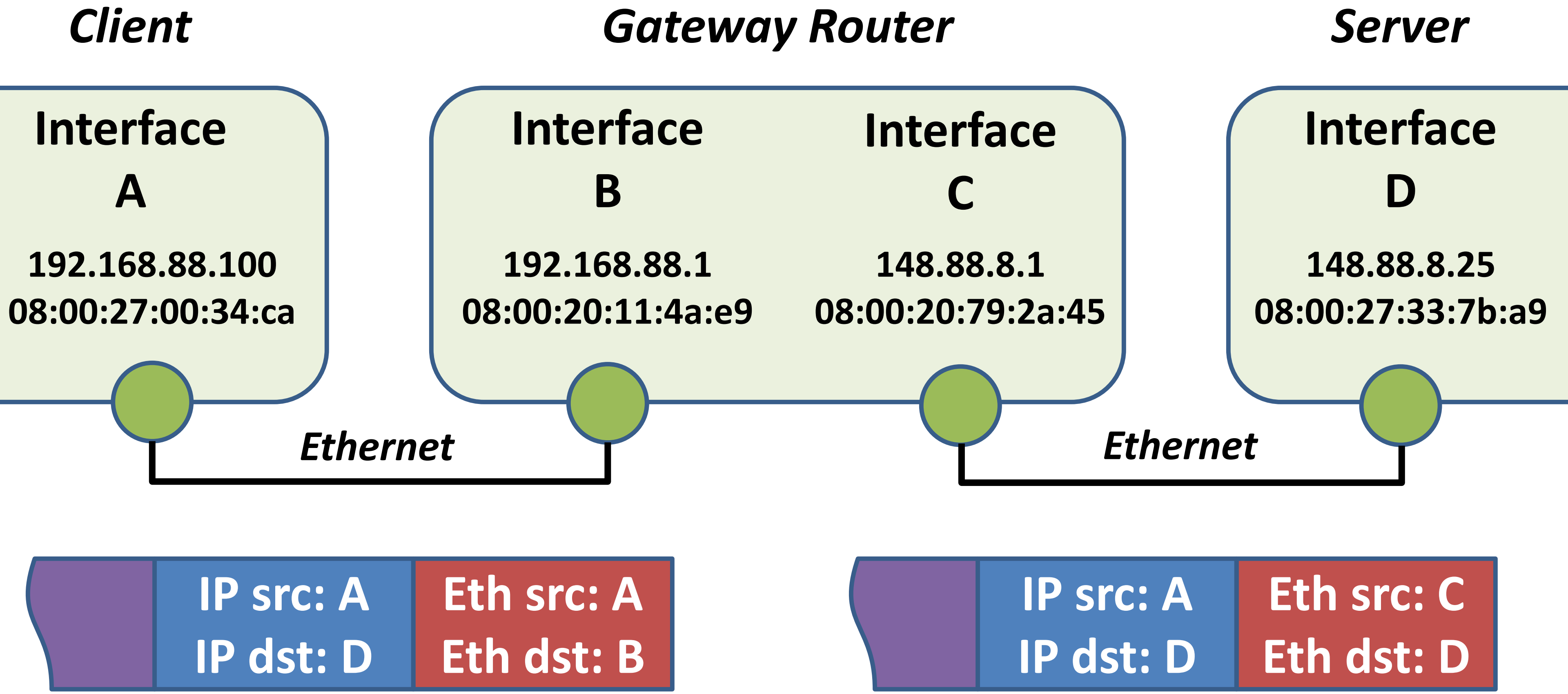
  1080::8:800:200C:417A/64

Sending a Network (IP) Packet

- We must tell our link-layer to send packet to hardware (MAC) address of gateway router
  - Packet is addressed to IP address of server
    …and sent by link-layer to MAC address of gateway router



Looking Hop-by-Hop



Mapping Addresses

- Names ⟺ IP Addresses
  - Domain Name Service (DNS)
- IP Addresses ➡ Hardware/ Link-Layer Addr
  - Address Resolution Protocol (ARP)
- Link-Layer Address ➡ IP Address
  - Reverse Address Resolution Protocol (RARP)

Hardware addresses more commonly known as Medium Access Control (MAC) addresses
See the IEEE LAN/RM in first lecture to see why

Cached mappings : Windows

```
C:\> arp -a

Interface: 10.20.101.72 --- 0xc
  Internet Address      Physical Address      Type
  10.20.101.1           00-0c-42-f8-b2-e2     dynamic
  10.20.101.127         ff-ff-ff-ff-ff-ff     static
  224.0.0.22            01-00-5e-00-00-16     static
  224.0.0.252           01-00-5e-00-00-fc     static
  239.255.255.250       01-00-5e-7f-ff-fa     static
  255.255.255.255       ff-ff-ff-ff-ff-ff     static
```

Linux

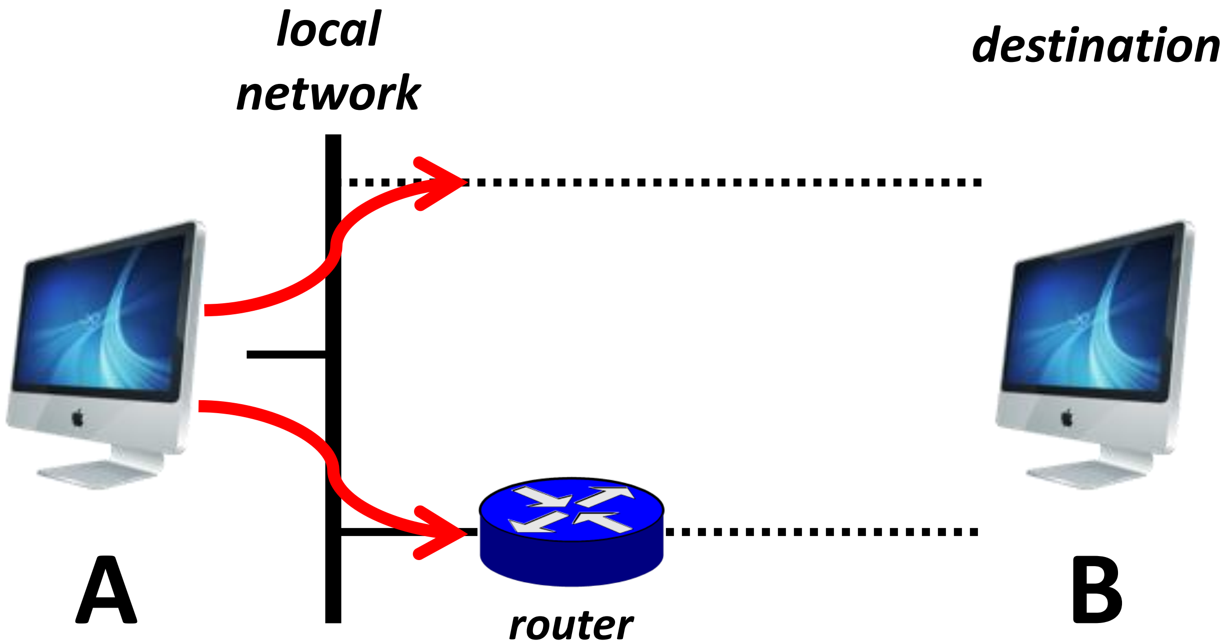
```
acs:~$ arp -a
? (194.80.37.1) at 00:07:b4:00:25:02 [ether] on eth0
```

…and Cisco IOS

```
wallace# show ip arp
Protocol  Address     Age(min) Hardware Addr   Type   Interface
Internet  10.20.0.1          - 0007.b404.7601  ARPA  GigE0/1.70
Internet  10.20.0.4        123 000c.299f.9e80  ARPA  GigE0/1.70
Internet  10.20.0.5         70 0004.2353.2b46  ARPA  GigE0/1.70
Internet  10.20.0.6        128 0015.17d1.541f  ARPA  GigE0/1.70
Internet  10.20.0.10         2 000c.29a8.396e  ARPA  GigE0/1.70
Internet  10.20.0.12         0 0015.1776.63cf  ARPA  GigE0/1.70
Internet  10.20.0.15         6 000c.29c2.75b9  ARPA  GigE0/1.70
```

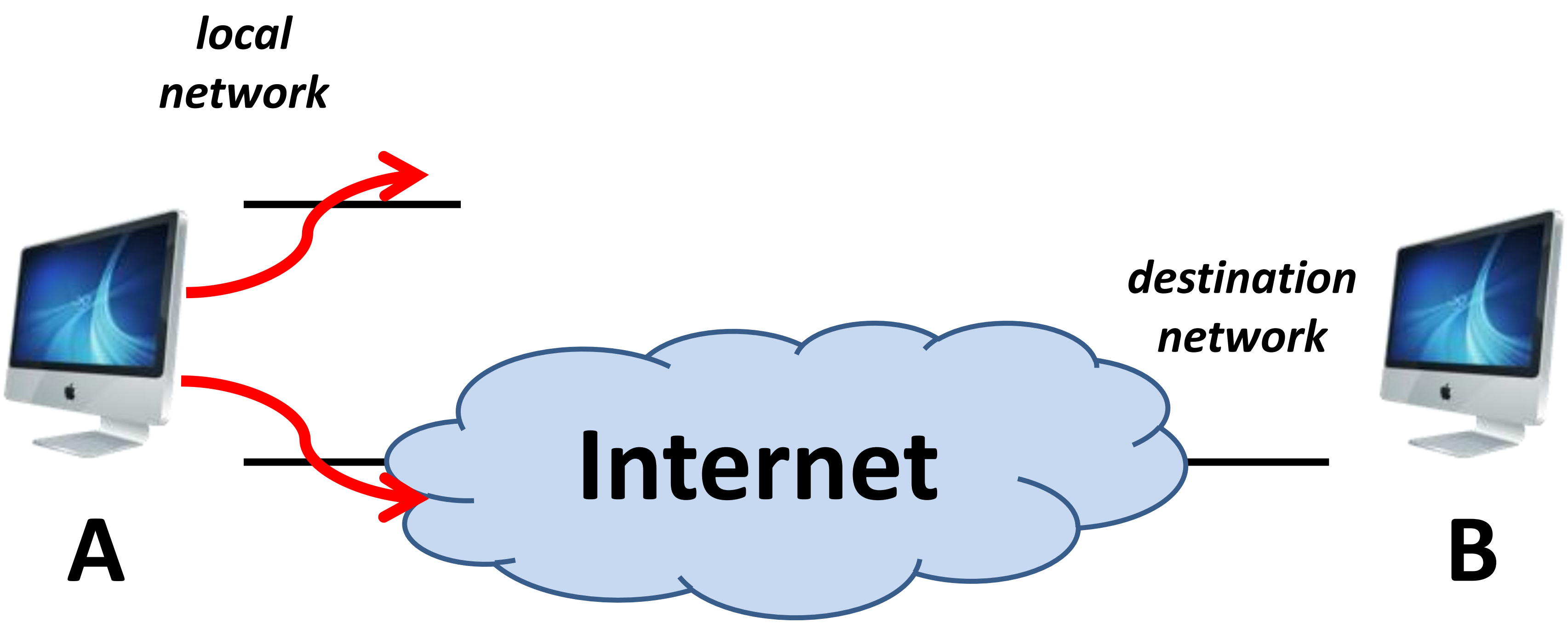
Getting from A to B

- Given a packet destined for host B
  - Is B on same physical link? *...send direct*
  - Is B on a remote network? *...send to router*



More Generally…

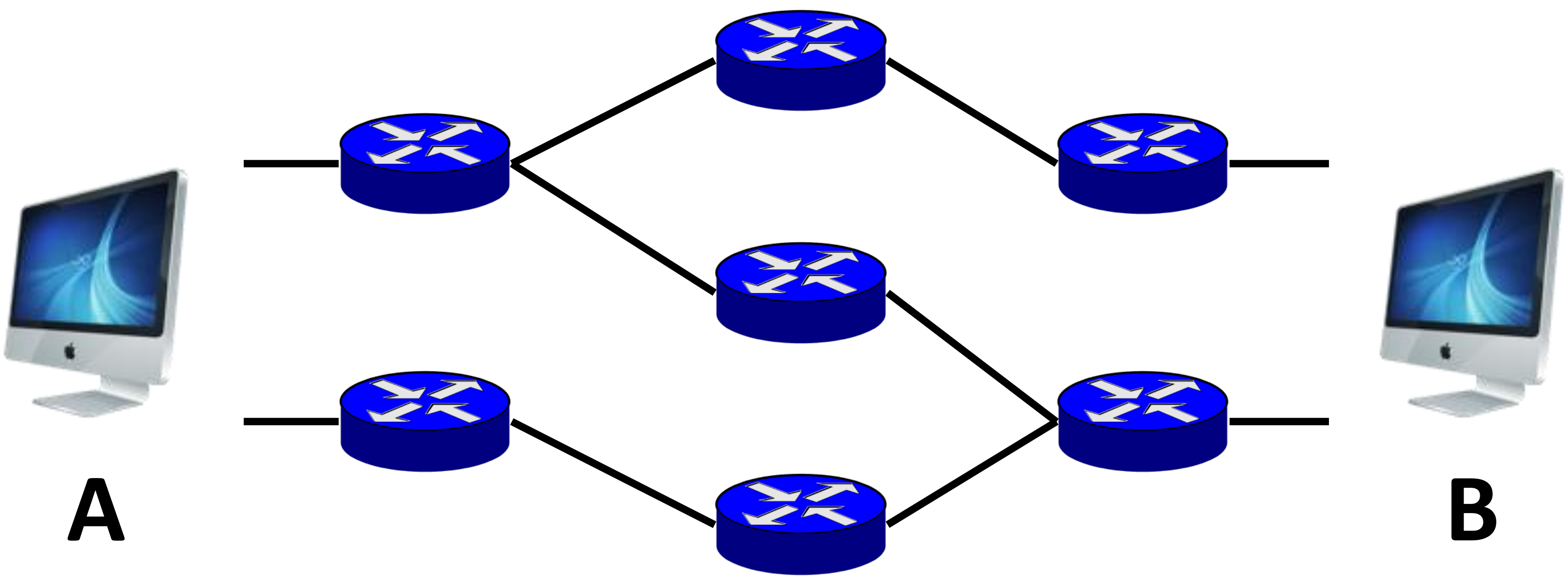
- PC A may have many interfaces, thus options
- If we send to a gateway router… what next?
- How should packet be *forwarded* toward B?



Getting from A to B

- Internet formed from a set of routers
  - No global view (at IP level)

- **Routers conspire to deliver packets to destination**
  - Each pushing packets closer to their destination



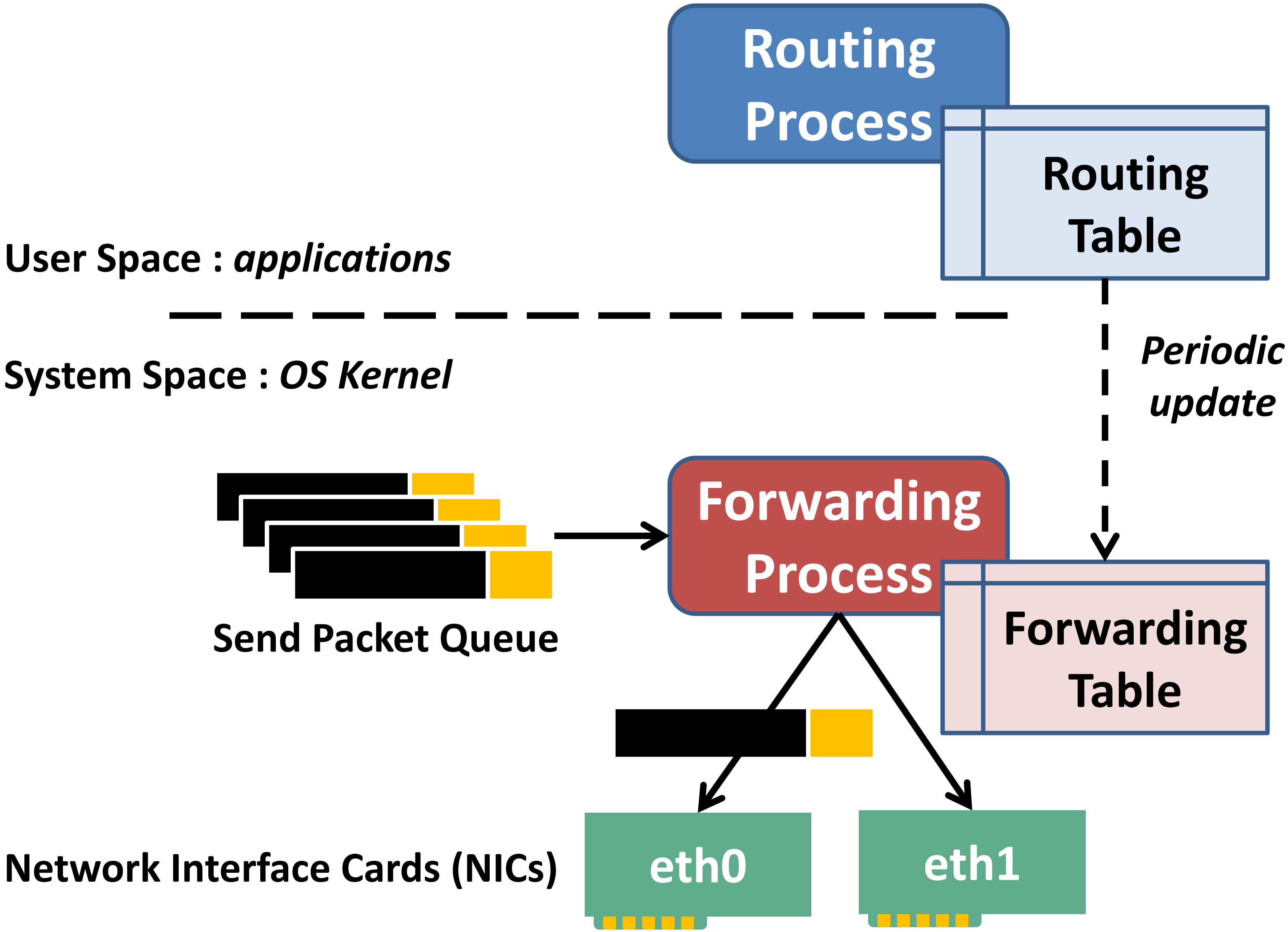
Internet 'Routing'

- Two <u>distinct</u> parts to process

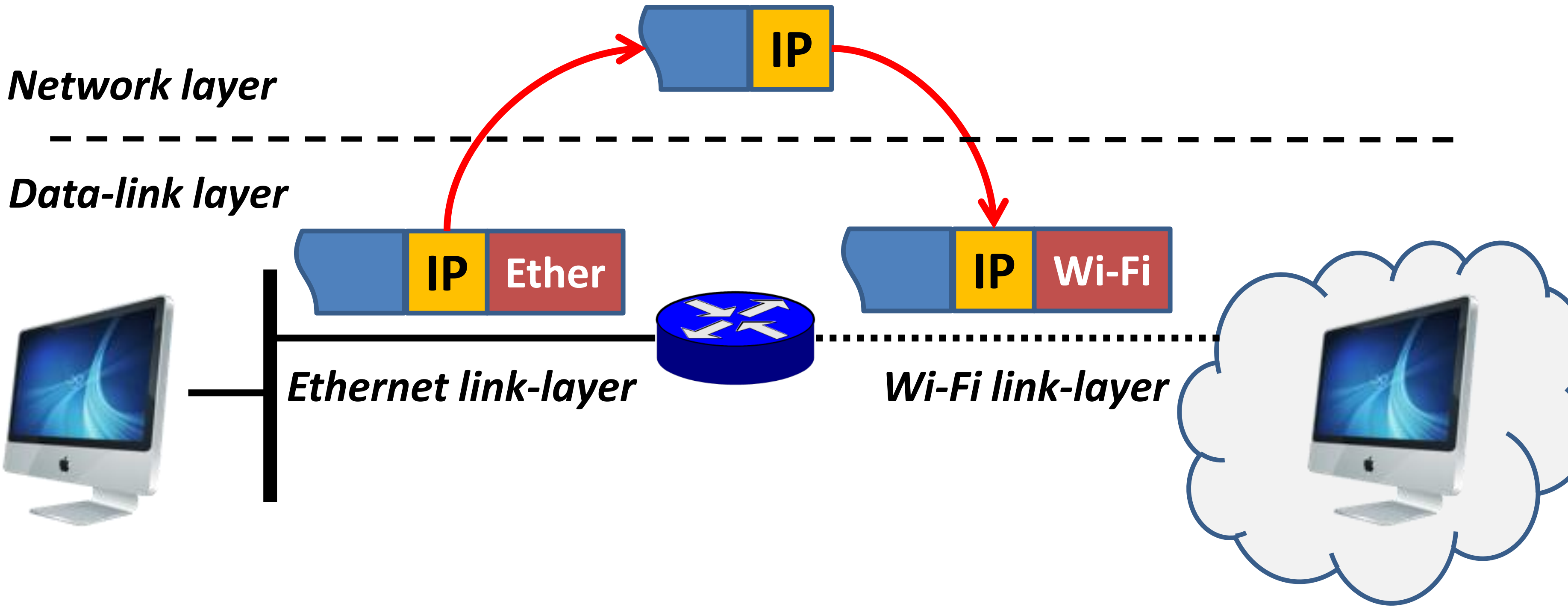
  – Routing
    - **Application level process** to determine routes

  – Forwarding
    - **System level process** directing packets according to learned routes

Routing and Forwarding

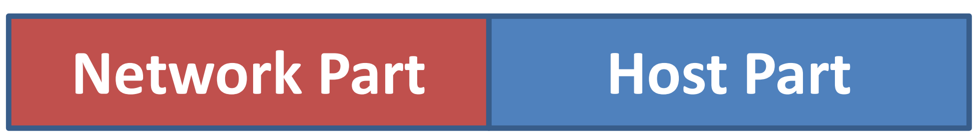


IP Forwarding



IPv4 Address Format

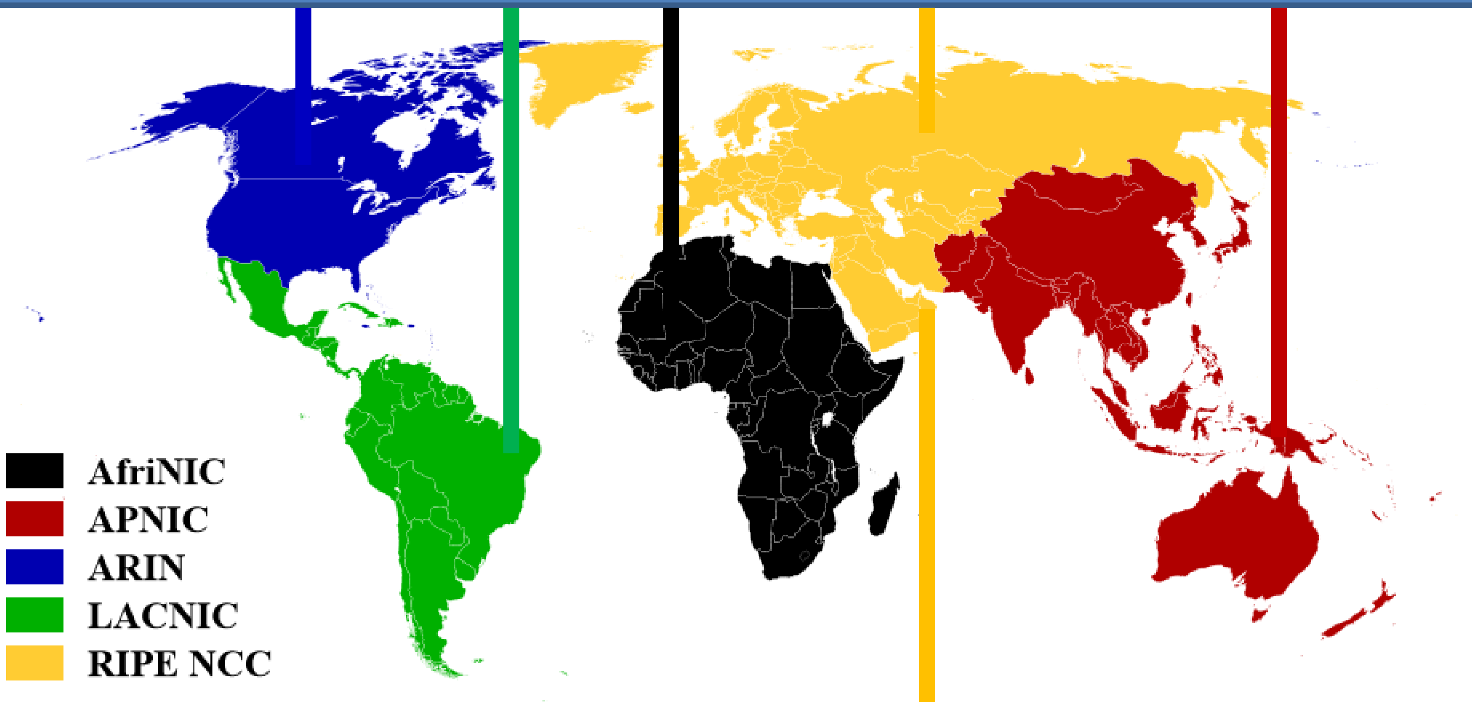
- 32 bit address
- Typically represented in dotted-quad form
  - 194.80.37.5
  - Each part represents 8 bit (byte), thus 0 – 255



Internet Address Allocation

Internet Corporation for Assigned Names and Numbers (ICANN)

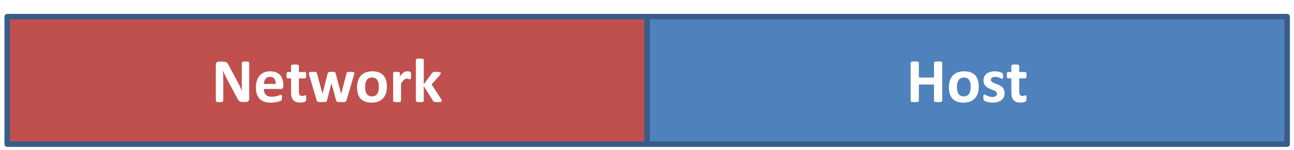
Internet Assigned Numbers Authority (IANA)



European Internet Service Providers (ISPs)

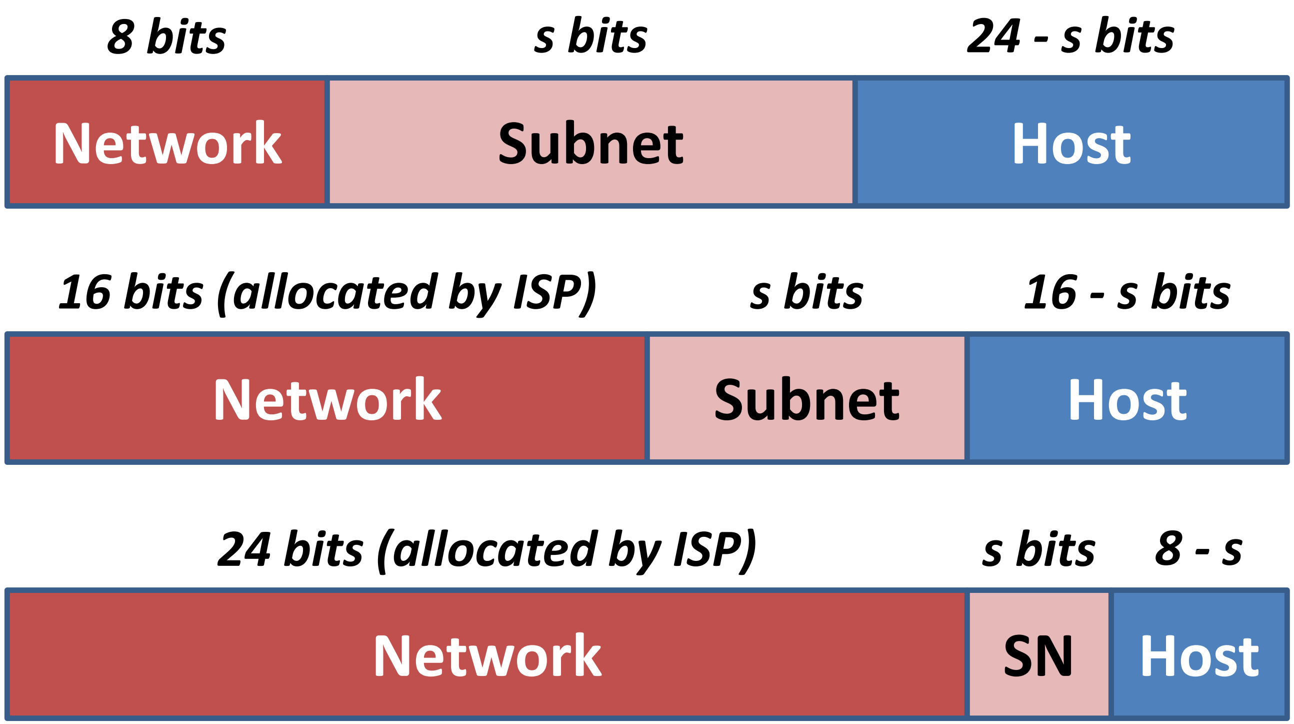
IP Addresses

- Network addresses allocated to organisations by their *Internet Service Provider* (ISP)
  - Used to correctly forward traffic – *how we know destination*
  - Fixed and cannot be changed by organisation
- Host addresses ‘belong’ to organisation
  - Allocate/ change them at will



Internal IP Sub-netting

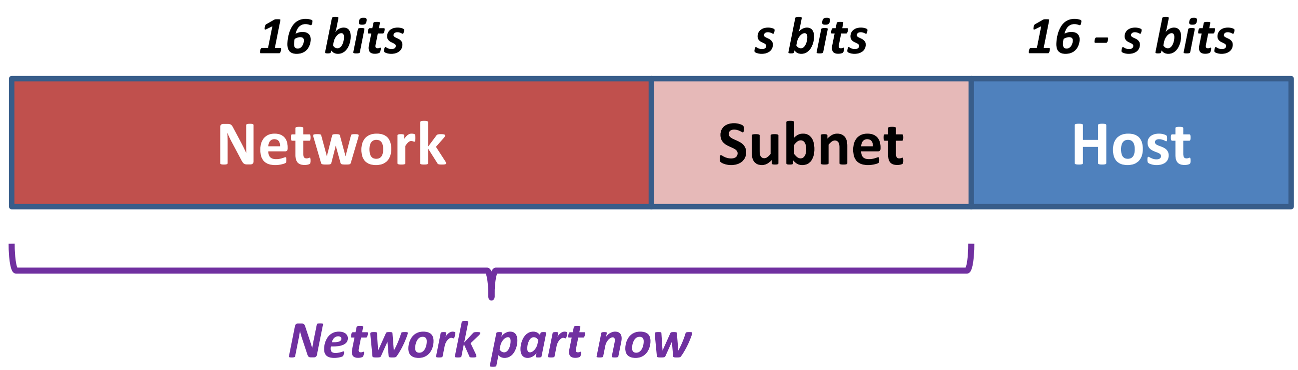
- Within organisation
  - Borrow host bits to form internal (sub-)networks
  - Network part is fixed – *what ISP allocated to org.*



Sub-netting

- Notice we can't determine network-host boundary
  - Subnet address only makes sense within organisation
  - *We need some more information…*

- ISP will still treat this as 16bit network address
  - The network address it allocated to organisation



Subnet Masks

	Network			Host
Address	192	168	88	32
Subnet mask	**255**	**255**	**255**	0
Result	**192**	**168**	**88**	0

	Network		Host	
Address	192	168	88	32
Subnet mask	**255**	**255**	0	0
Result	**192**	**168**	0	0

Variable Length Subnet Masking (VLSM)

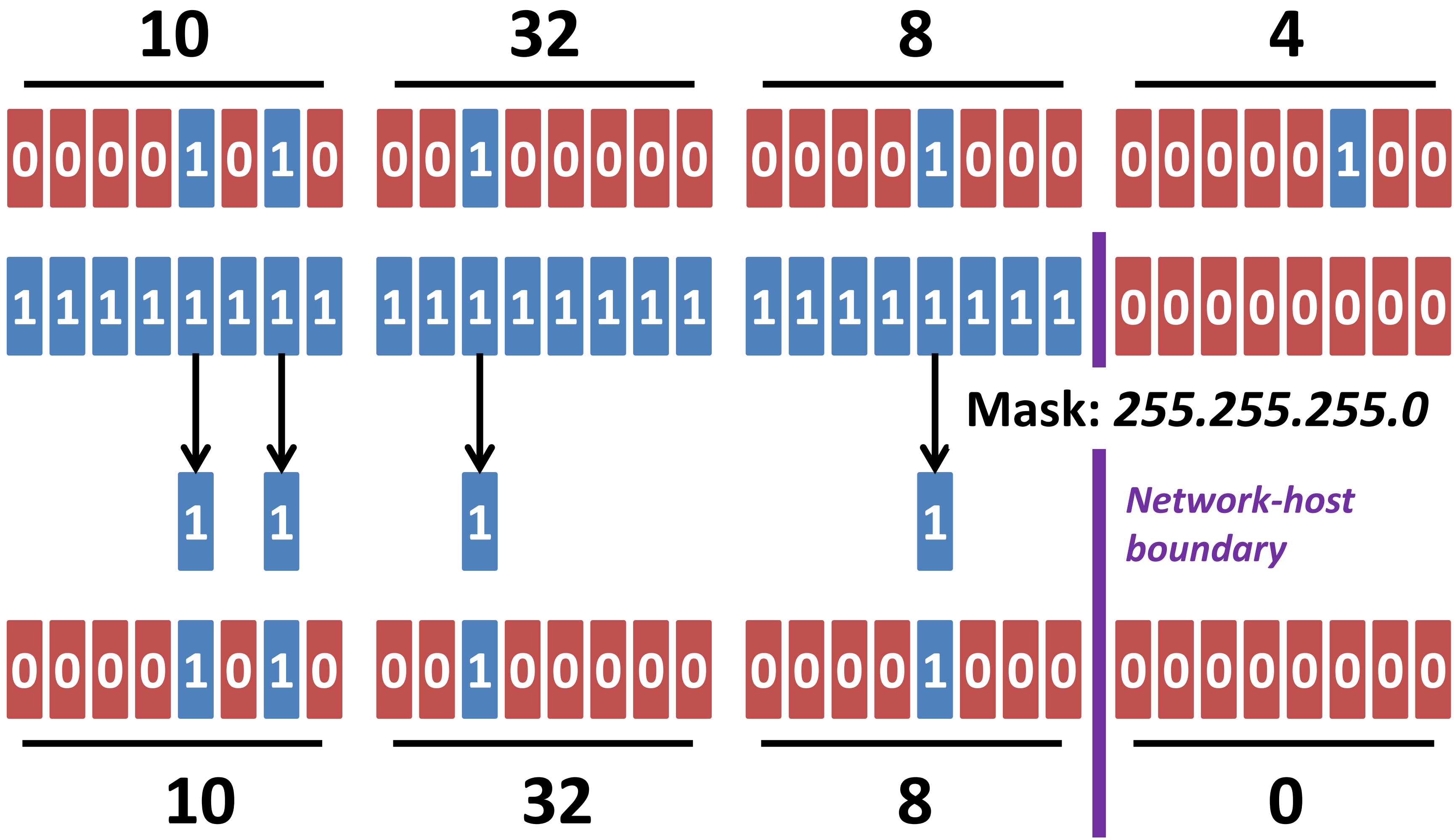
- *Subnet mask* is used to split (sub-)net and host parts

- Mask must have set of ones followed by set of zeros
  - Once a zero bit appears, all remaining bits must be zero

OK	11111111	11111111	11111111	00000000
illegal	11111111	11111111	11111110	10000000

- Any IP address **AND**’d with its subnet mask gives network address on which it resides

Subnet Masks



Masks and Subnets

		128	64	32	16	8	4	2	1
Address	194.80.37.	0	0	0	0	0	0	0	0
Mask	255.255.255.	0							

255.255.255.0	***← Subnet mask***		
194.80.37.**0**	**← Single available subnet**		

Masks and Subnets

		128	64	32	16	8	4	2	1
Address	194.80.37.	X	0	0	0	0	0	0	0
Mask	255.255.255.	128	*Host bits*						

255.255.255.0	**255.255.255.128**	← ***Subnet mask***	
194.80.37.**0**	194.80.37.**0**	← **128 bit unset in address**	
	194.80.37.**128**	← **128 bit set in address**	

Masks and Subnets

		128	64	32	16	8	4	2	1
Address	194.80.37.	X	X	0	0	0	0	0	0
Mask	255.255.255.	192		*Host bits*					

255.255.255.0	**255.255.255.128**	**255.255.255.192**	**← *Subnet mask***
194.80.37.**0**	194.80.37.**0**	194.80.37.**0**	
		194.80.37.**64**	
	194.80.37.**128**	194.80.37.**128**	
		194.80.37.**192**	**(128 + 64)**

Masks and Subnets

		128	64	32	16	8	4	2	1
Address	194.80.37.	X	X	X	0	0	0	0	0
Mask	255.255.255.	224			*Host bits*				

255.255.255.0	255.255.255.128	255.255.255.192	255.255.255.224
194.80.37.**0**	194.80.37.**0**	194.80.37.**0**	194.80.37.**0**
			194.80.37.**32**
		194.80.37.**64**	194.80.37.**64**
			194.80.37.**96**
	194.80.37.**128**	194.80.37.**128**	194.80.37.**128**
			194.80.37.**160**
		194.80.37.**192**	194.80.37.**192**
			194.80.37.**224**

Prefix Notation

		128	64	32	16	8	4	2	1
Address	194.80.37.	1	1	1	0	0	0	0	0
Mask	255.255.255.	224							

3 bytes, 24 bits → ***+3 = 27 bits***

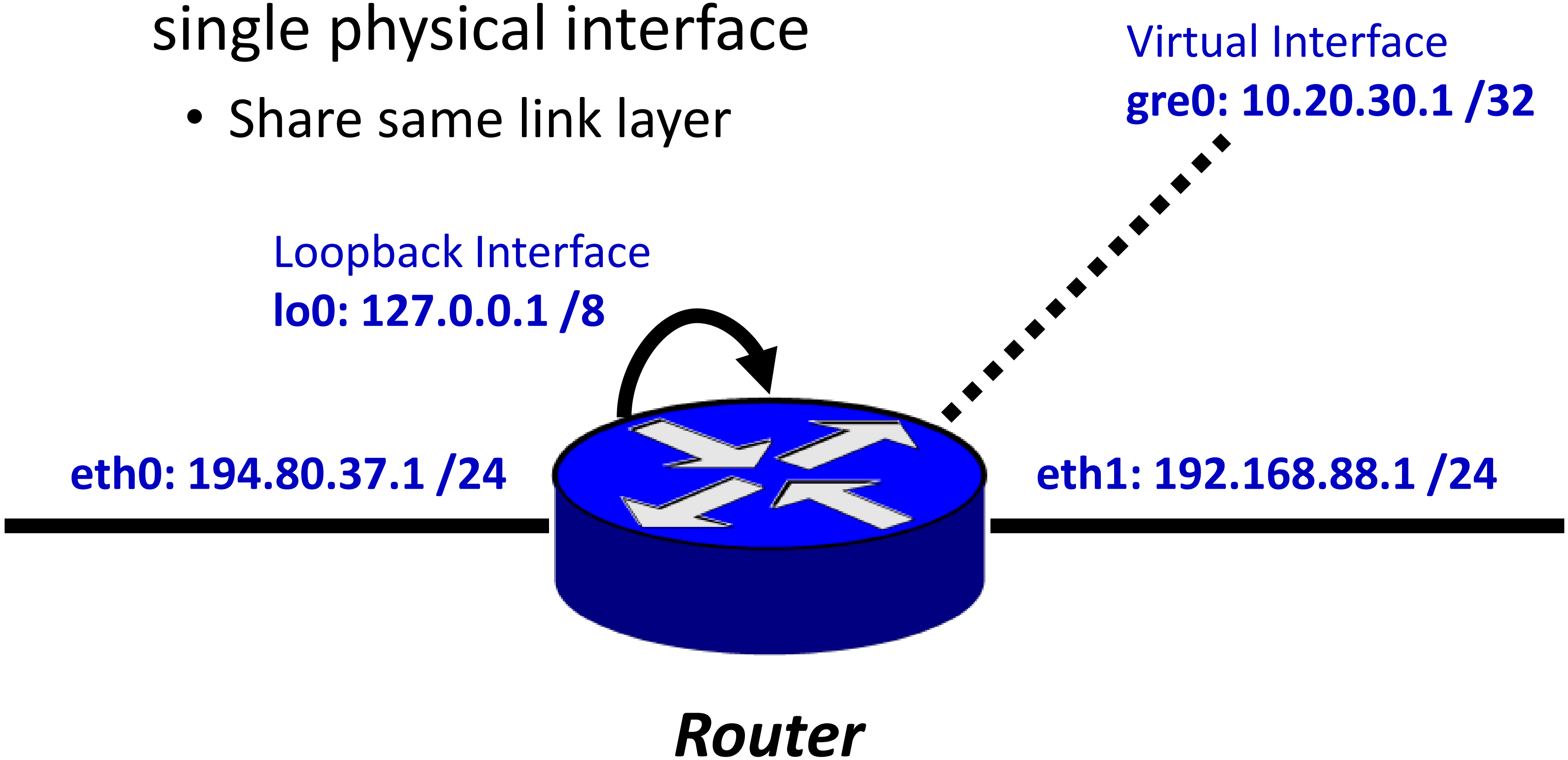
Address: 194.80.37.0
Mask: 255.255.255.224

Prefix: 194.80.37.0 **/ 27**

		128 + 64 = 192	**128 + 64 + 32 = 224**
255.255.255.0	**255.255.255.128**	**255.255.255.192**	**255.255.255.224**
/24	**/25**	**/26**	**/27**

Addresses are bound to Interfaces

- Addresses belong to interfaces not machines
  - Note: Can have multiple *virtual interfaces* on single physical interface
    - Share same link layer



One Hop at a Time

- IP depends on underlying data-link protocol
- Data-link protocol can only address devices on same physical link/ network segment
- IP header holds endpoint addresses
  - Original source and final destination
- Data-link frames ‘holding’ packet sent hop-by-hop
  - Always sent to next-hop router, *until last subnet*

Data sent on wire

Forwarding Table

- Each device maintains a forwarding table

This is one of the device's network interfaces
- can be interface name or address

Net. Address/ Prefix	Subnet Mask	Next Hop Router	Interface	Metric ('cost')
194.80.37.0	**255.255.255.0**	**192.168.88.1**	**eth0**	**0**

192.168.88.1 is router that can get our packets to network 194.80.37.0

This router *MUST* be on same (virtual) link/ subnet as eth0

Forwarding Table

- Many nodes have a ***default route*** prefix: 0.0.0.0 /0
  - – Route of last resort, when no other route available
  - – Default route typically toward our ISP's router

Net. Address/ Prefix	Subnet Mask	Next Hop Router	Interface	Metric ('cost')
0.0.0.0	**0.0.0.0**	**148.88.8.1**	**eth1**	**0**
194.80.37.0	**255.255.255.0**	**192.168.88.1**	**eth0**	**0**

Checking Host Table

U – route 'up'
G – gateway route
H – host route

```
root@acs:~# route
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
default         194.80.37.1     0.0.0.0         UG    0      0        0 eth0
link-local      *               255.255.0.0     U     1000   0        0 eth0
194.80.37.0     *               255.255.255.128 U     1      0        0 eth0
root@acs:~#
```

```
C:\> route print
Network Destination        Netmask          Gateway       Interface  Metric
          0.0.0.0          0.0.0.0      10.20.30.1     10.20.30.72     25
       10.20.30.0  255.255.255.128         On-link     10.20.30.72    281
      10.20.30.72  255.255.255.255         On-link     10.20.30.72    281
     10.20.30.127  255.255.255.255         On-link     10.20.30.72    281
        127.0.0.0        255.0.0.0         On-link       127.0.0.1    306
        127.0.0.1  255.255.255.255         On-link       127.0.0.1    306
  127.255.255.255  255.255.255.255         On-link       127.0.0.1    306
        224.0.0.0        240.0.0.0         On-link       127.0.0.1    306
        224.0.0.0        240.0.0.0         On-link     10.20.30.72    281
```

Background

Updating Routing Table: Linux

```
root@as:~# route
Kernel IP routing table
Destination    Gateway        Genmask         Flags Metric Ref    Use Iface
default        194.80.37.1    0.0.0.0         UG    0      0        0 eth0
link-local     *              255.255.0.0     U     1000   0        0 eth0
194.80.37.0    *              255.255.255.128 U     1      0        0 eth0
root@as:~#
root@as:~# route add -net 11.22.33.0 netmask 255.255.255.0 gw 194.80.37.2
root@as:~#
root@as:~# route
Kernel IP routing table
Destination    Gateway        Genmask         Flags Metric Ref    Use Iface
default        194.80.37.1    0.0.0.0         UG    0      0        0 eth0
11.22.33.0     194.80.37.2    255.255.255.0   U     0      0        0 eth0
link-local     *              255.255.0.0     U     1000   0        0 eth0
194.80.37.0    *              255.255.255.128 U     1      0        0 eth0
root@as:~#
root@as:~# route del -net 11.22.33.0 netmask 255.255.255.0
```

Multiple Next Hop Matches

- Likely to be more than one matching entry
  - If only due to default route
    - Which always matches
- **Always select longest matching prefix**
  - Most specific entry
  - Should be best path to destination

Longest Prefix Match

- Apply mask to address and compare with prefix

Destination Address = 10 | 32 | 5 | 9

Prefix				Mask				Masked Address				
10	**32**	0	0	**255**	**255**	0	0	**10**	**32**	0	0	✔
10	**32**	**5**	0	**255**	**255**	**255**	0	**10**	**32**	**5**	0	✔
192	**168**	**88**	6	**255**	**255**	**255**	0	**10**	**32**	**5**	0	✘

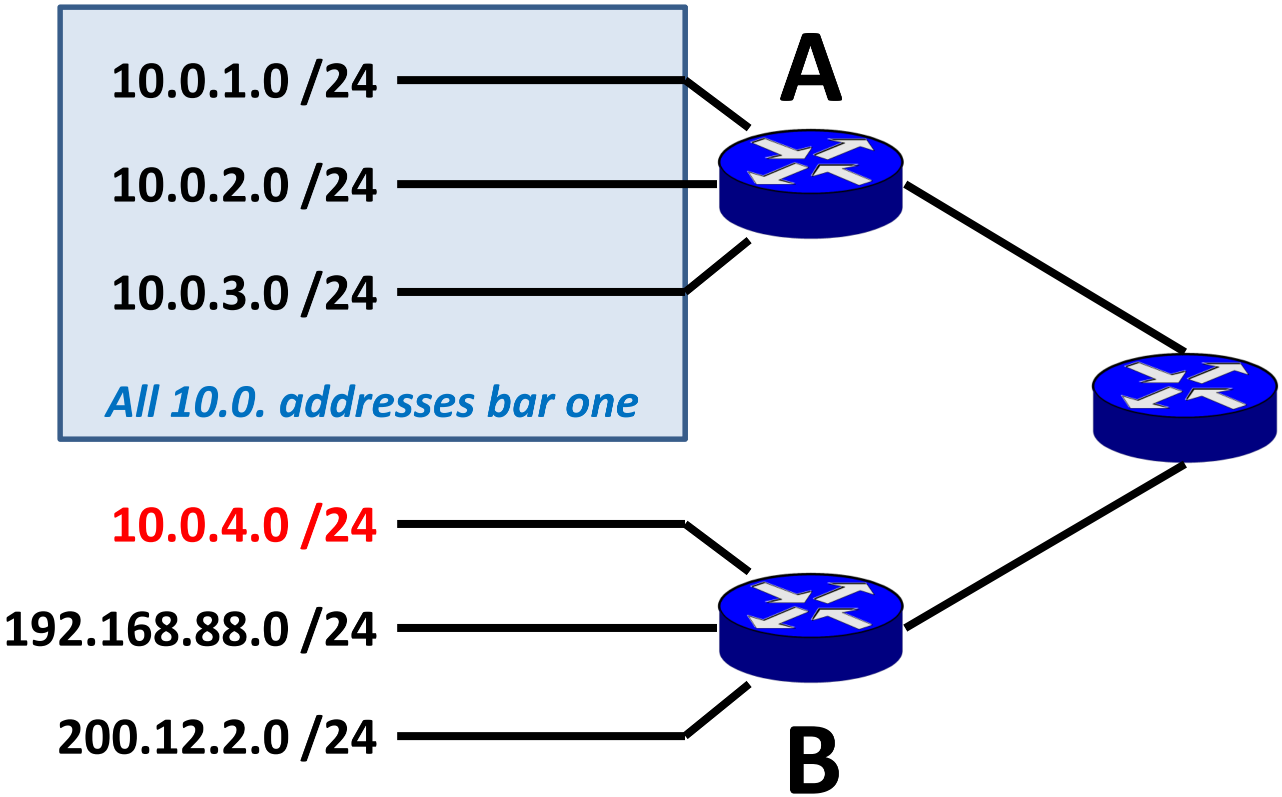
= ?

= Prefix ?

- Options are:
  - 10.32. shortest of two
  - 10.32.5. longest of two ← *we select this*

Why Multiple Matches?

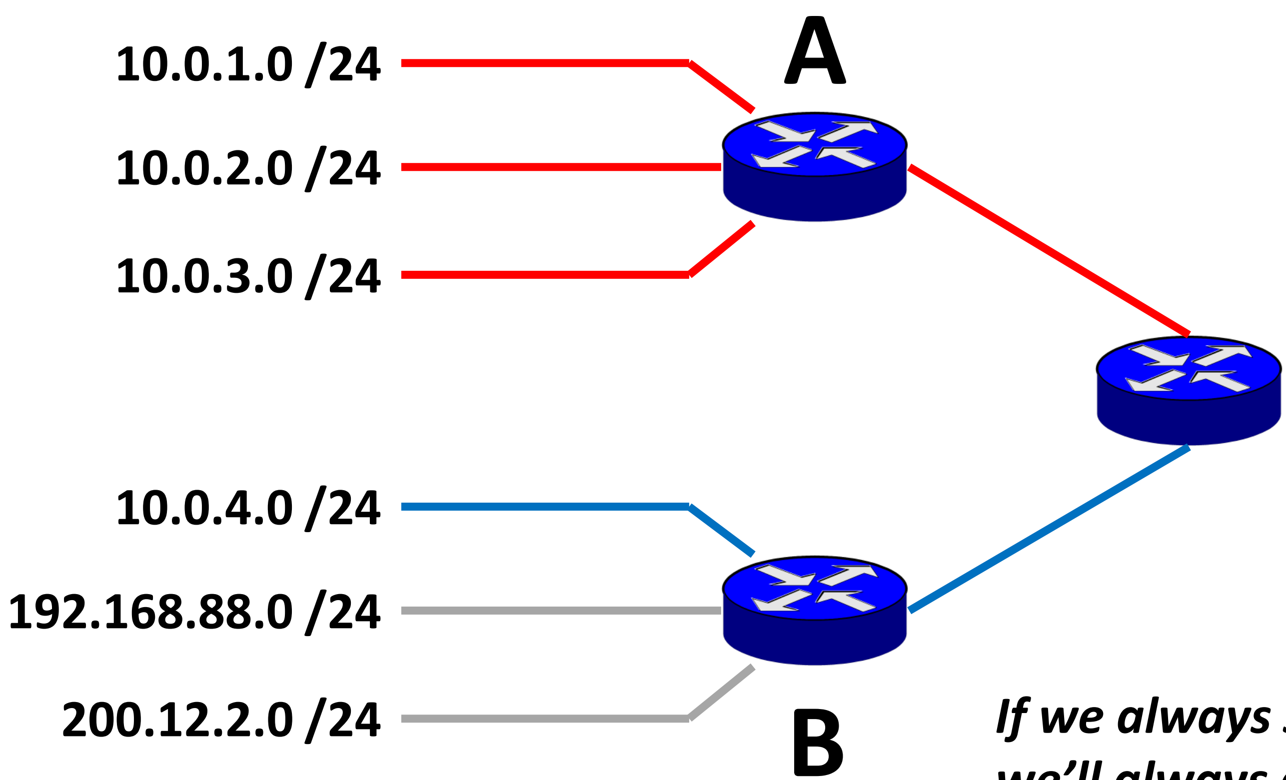
- Two equivalent scenarios :
  1. Fully enumerated table



Network Address	N.H.
10.0.1.0 /24	A
10.0.2.0 /24	A
10.0.3.0 /24	A
10.0.4.0 /24	**B**
192.168.88.0 /24	B
200.12.2.0 /24	B

Why Multiple Matches?

- Two equivalent scenarios :
  2. 10.0.0.0/16 with **exception** for 10.0.4.0/24



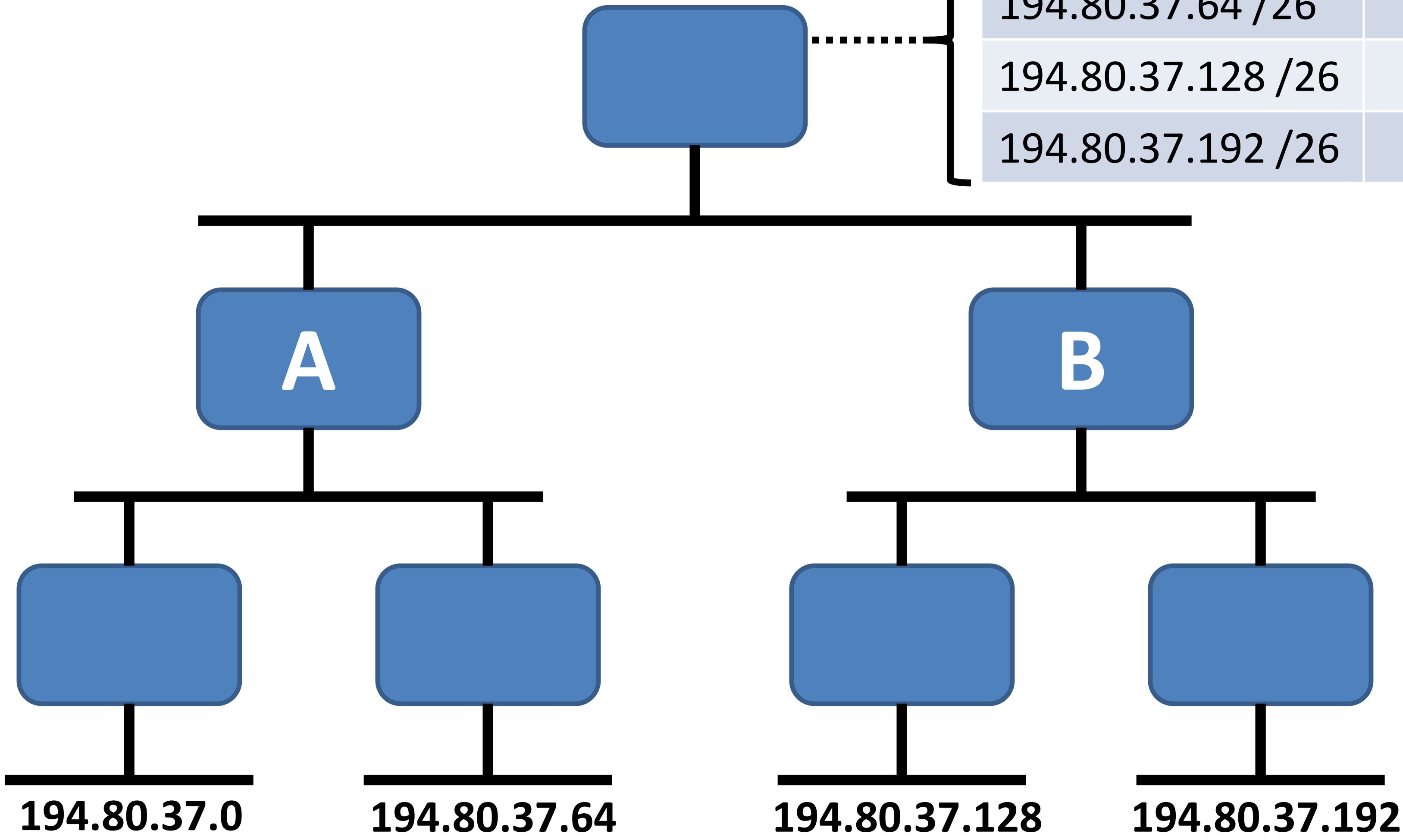
Simpler routing table
(fewer to check = faster)

Network Address	N.H.
10.0.0.0 /16	**A**
10.0.4.0 /24	**B**
192.168.88.0 /24	B
200.12.2.0 /24	B

If we always select Longest Prefix Match we'll always get to correct destination

Building Tables

Forwarding Table	
Subnet	Nxt Hop
194.80.37.0 /26	A
194.80.37.64 /26	A
194.80.37.128 /26	B
194.80.37.192 /26	B



Route Summarisation

- Looking at that last example:

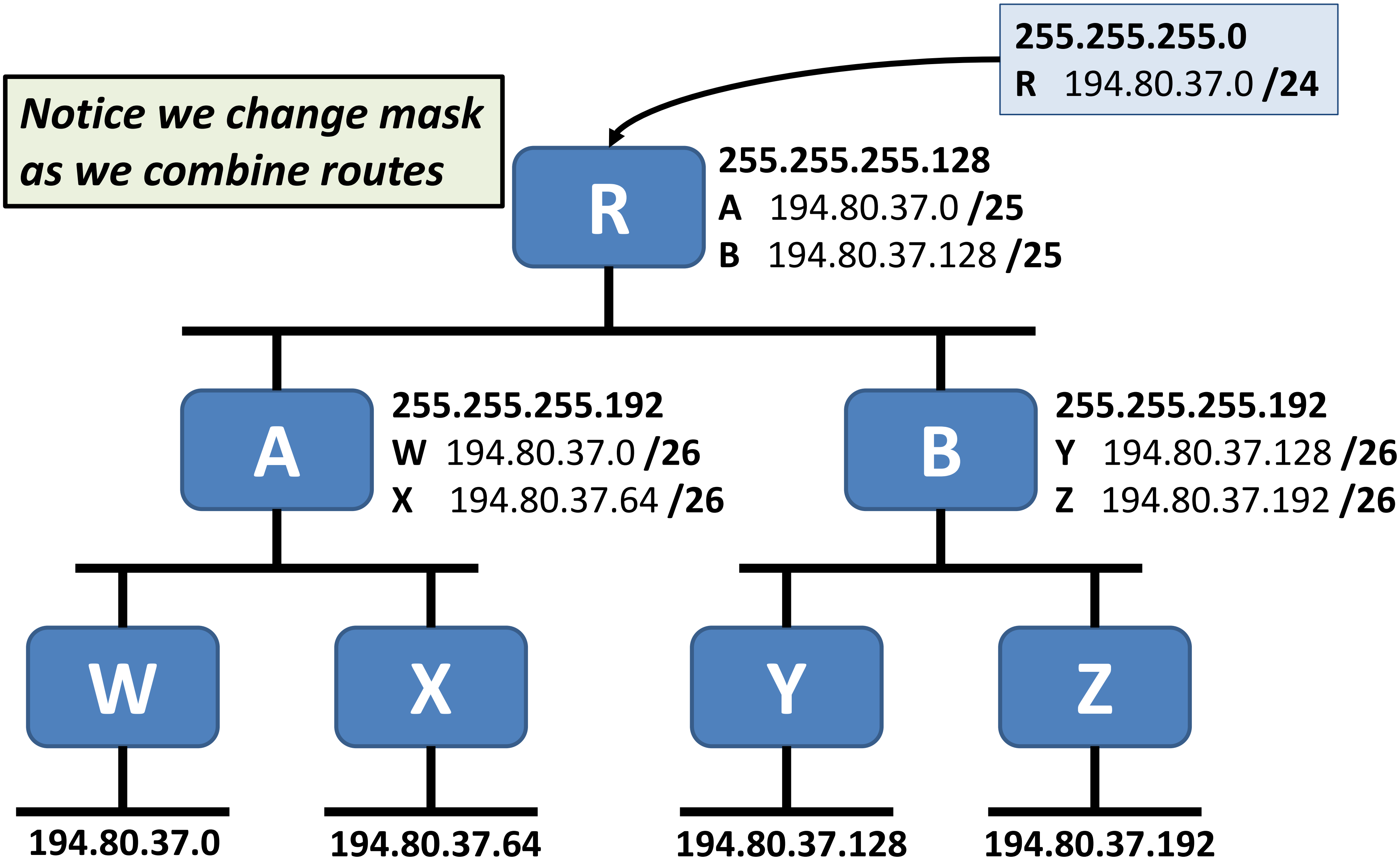
- We can combine these routes
  - Two routes differ only in last bit
    - 194.80.37.0 + 194.80.37.64
    - 194.80.38.128 + 194.80.37.192
  - Each pair has the same next hop

Initial Forwarding Table	
Subnet	**Nxt Hop**
194.80.37.0	**A**
194.80.37.64	**A**
194.80.37.128	**B**
194.80.37.192	**B**

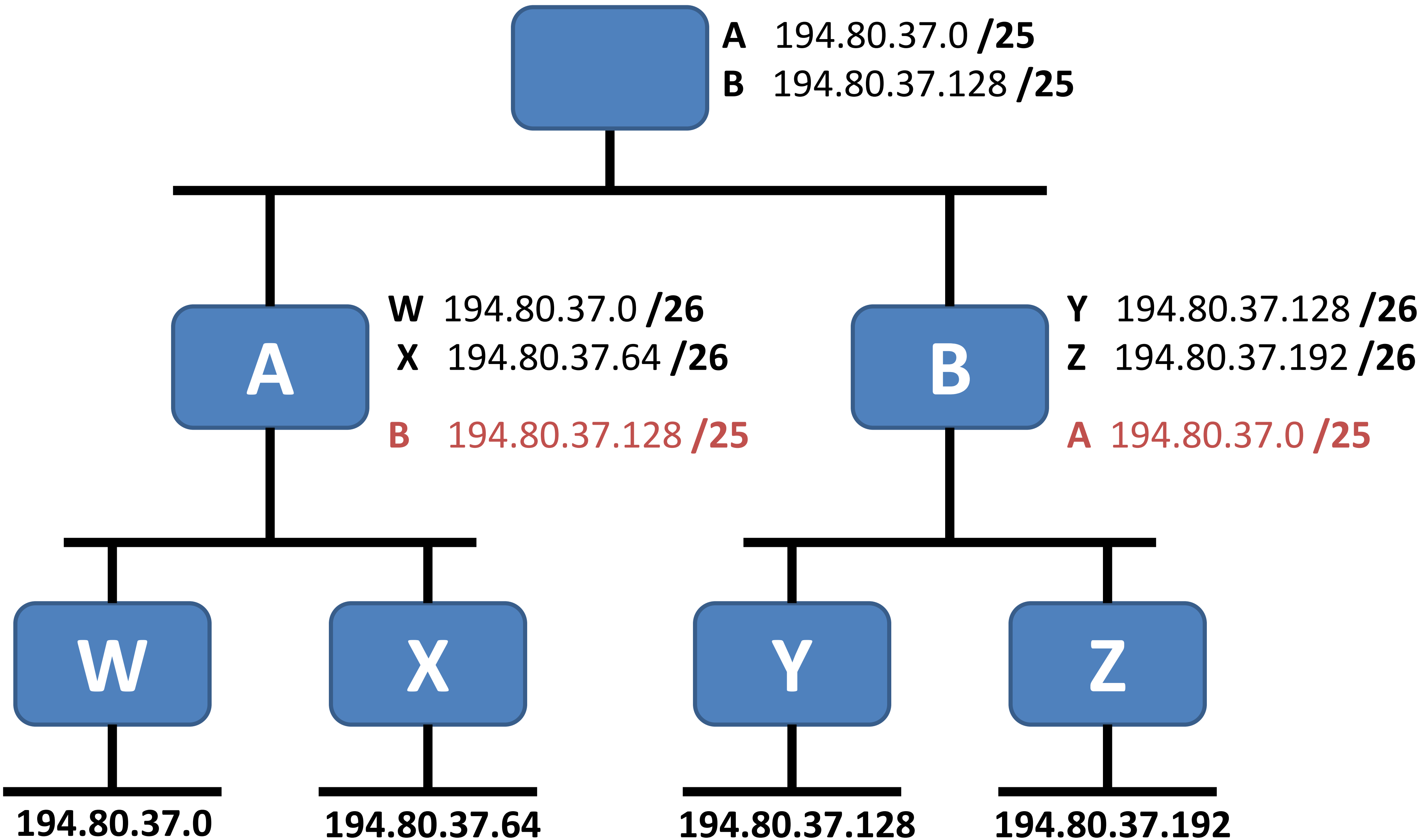
New Forwarding Table	
Subnet	**Nxt Hop**
194.80.37.0	A
194.80.37.128	B

**** aka. Route Aggregation***
– though this is really a specific wide area mechanism

Route Summarisation



Completing the tables



Question

A router supporting variable length subnet masks and classless inter-domain routing (CIDR) has the following forwarding entries:

Address/ Prefix Length	Next Hop
0.0.0.0/0	A
10.20.0.0/17	B
10.20.64.0/18	C
192.168.80.0/21	D
192.168.88.0/22	E

Showing full details of your working identify:

- Which next hop entries would match each of the following destination IP addresses, and
- In each case, which next hop would the router select?

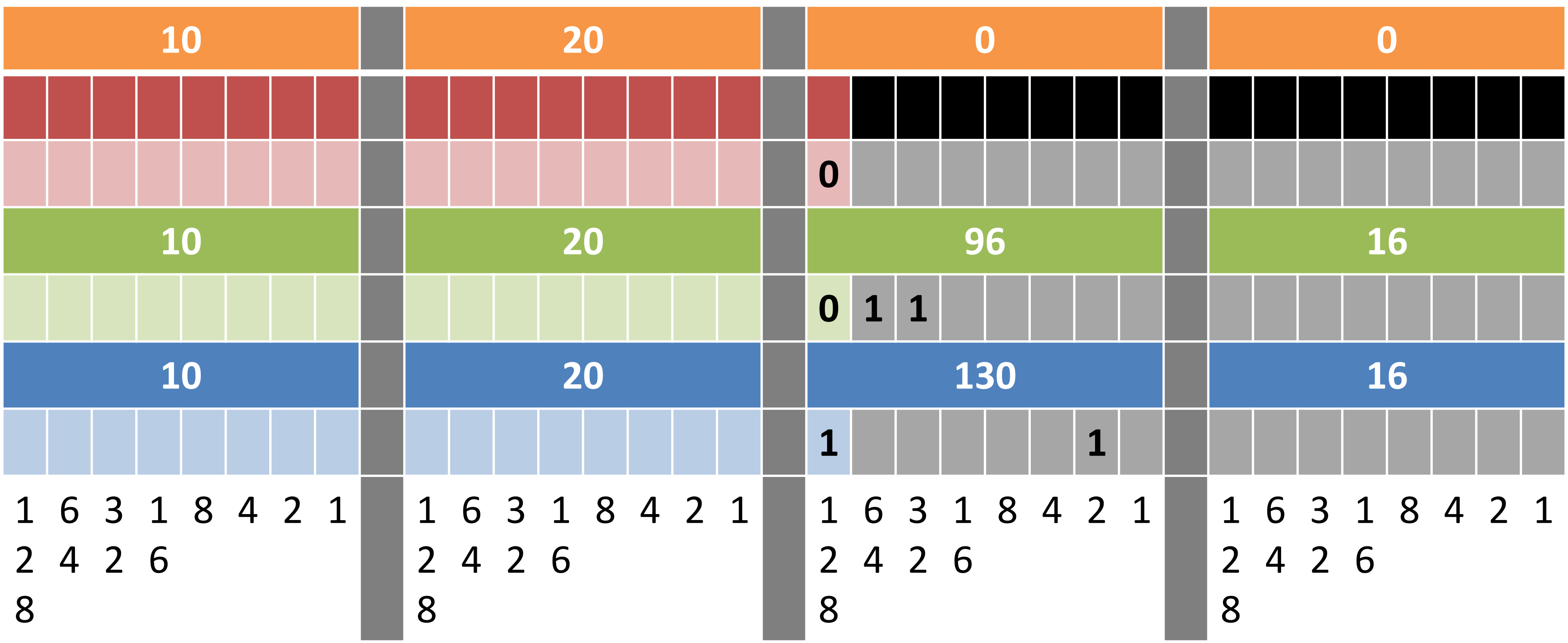
    i. 10.20.96.16
    ii. 10.20.130.16
    iii. 192.168.87.4
    iv. 192.168.89.2

First…

WHICH ADDRESSES MATCH?

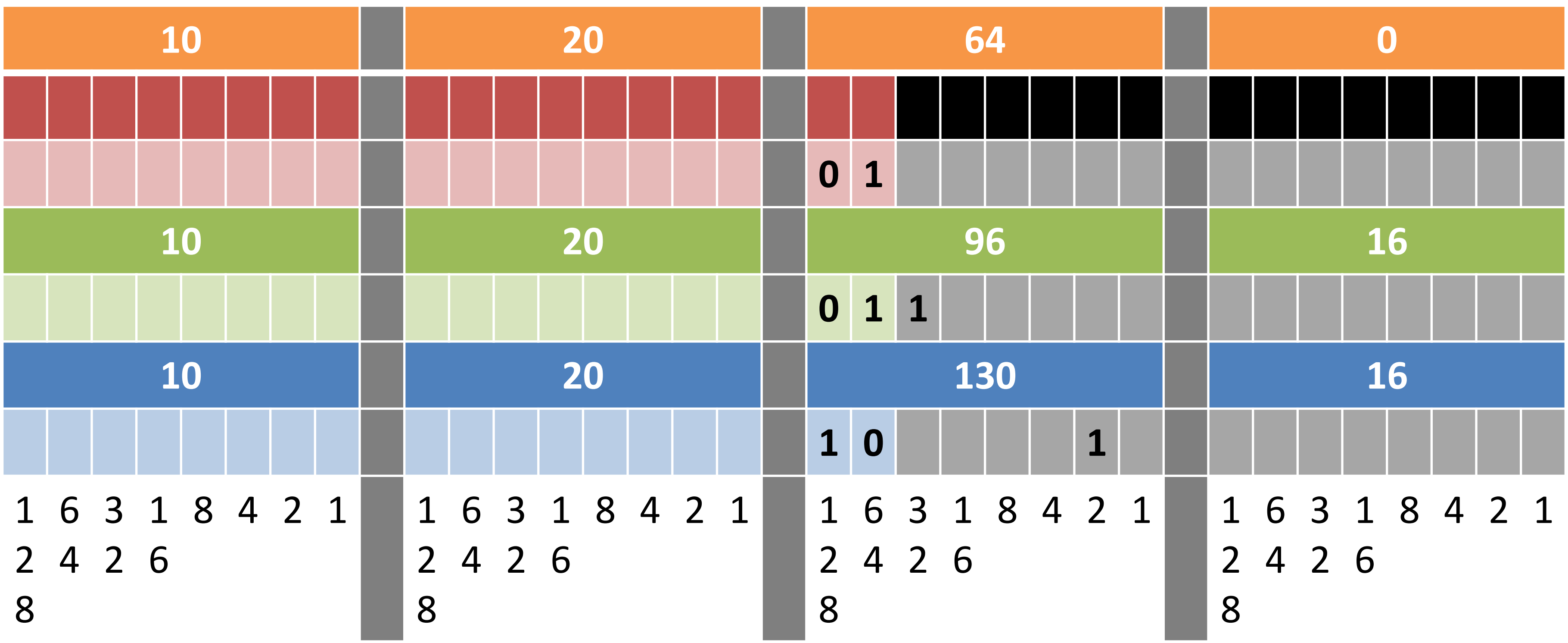
10.20.0.0/17

- 10.20.96.16 matches to 17 bits
- 10.20.130.16 doesn’t match to 17 bits



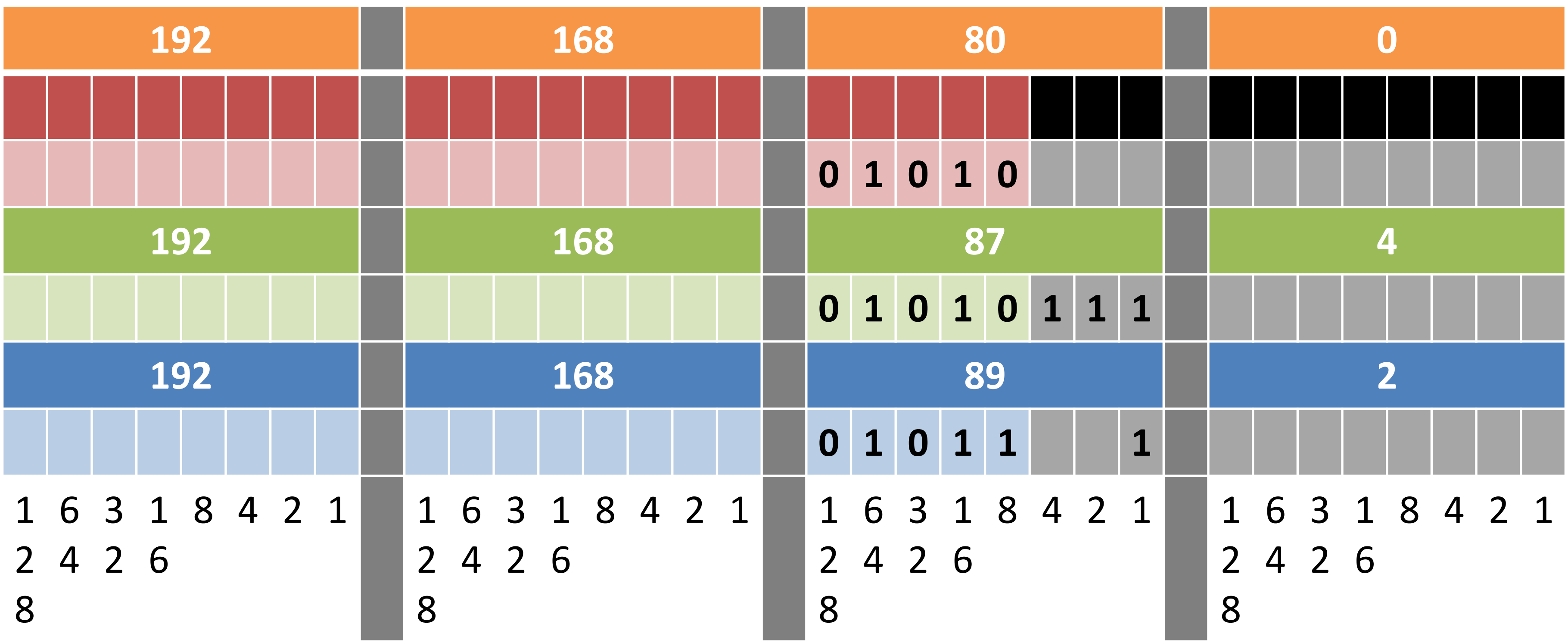
10.20.64.0/18

- 10.20.96.16 matches to 18 bits
- 10.20.130.16 doesn't match to 18 bits



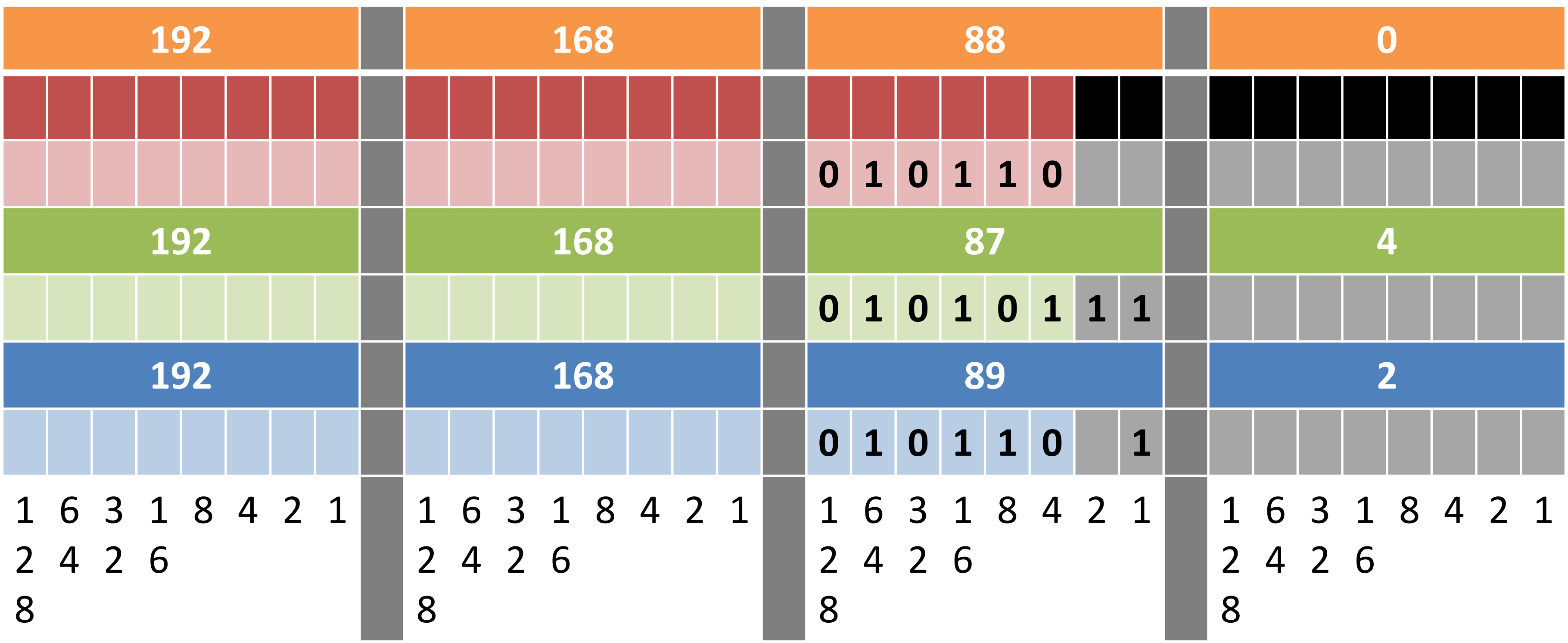
192.168.80.0/21

- 192.168.87.4 matches to 21 bits
- 192.168.89.2 doesn't match to 21 bits



192.168.88.0/22

- 192.168.87.4 doesn’t match to 22 bits
- 192.168.89.2 does match to 22 bits



So…

WHAT CHOICES DO WE HAVE?

10.20.96.16

- Choice is
  - 0.0.0.0 at 0 bits
  - 10.20.0.0 at 17 bits
  - 10.20.64.0 at 18 bits
- Select longest prefix (18 bits): 10.20.64.0 → C

Address	Prefix Length	Next Hop	Matches
0.0.0.0	0	A	Yes, always!
10.20.0.0	17	B	Yes
10.20.64.0	18	C	Yes
192.168.80.0	21	D	Obviously not
192.168.88.0	22	E	Obviously not

10.20.130.16

- Choice is
  - 0.0.0.0 at 0 bits
- Select longest prefix (0 bits): 0.0.0.0 → A
  - We use the default route

Address	Prefix Length	Next Hop	Matches
0.0.0.0	0	A	Yes, always!
10.20.0.0	17	B	No
10.20.64.0	18	C	No
192.168.80.0	21	D	Obviously not
192.168.88.0	22	E	Obviously not

192.168.87.4

- Choice is
  - 0.0.0.0 at 0 bits
  - 192.168.80.0 at 21 bits
- Longest prefix is (21 bits): 192.168.80.0 → D

Address	Prefix Length	Next Hop	Matches
0.0.0.0	0	A	Yes, always!
10.20.0.0	17	B	Obviously not
10.20.64.0	18	C	Obviously not
192.168.80.0	21	D	Yes
192.168.88.0	22	E	No

192.168.89.2

- Choice is
  - 0.0.0.0 at 0 bits
  - 192.168.88.0 at 22 bits
- Longest prefix is (22 bits): 192.168.88.0 → E

Address	Prefix Length	Next Hop	Matches
0.0.0.0	0	A	Yes, always!
10.20.0.0	17	B	Obviously not
10.20.64.0	18	C	Obviously not
192.168.80.0	21	D	No
192.168.88.0	22	E	Yes

Summary

Destination	Valid next hops	Next hop
10.20.96.16	A, B, C	C
10.20.130.16	A	A
192.168.87.4	A, D	D
192.168.89.2	A, E	E