CH. 3
IP FORWARDING AND ROUTING

Lecture on IP internetworks

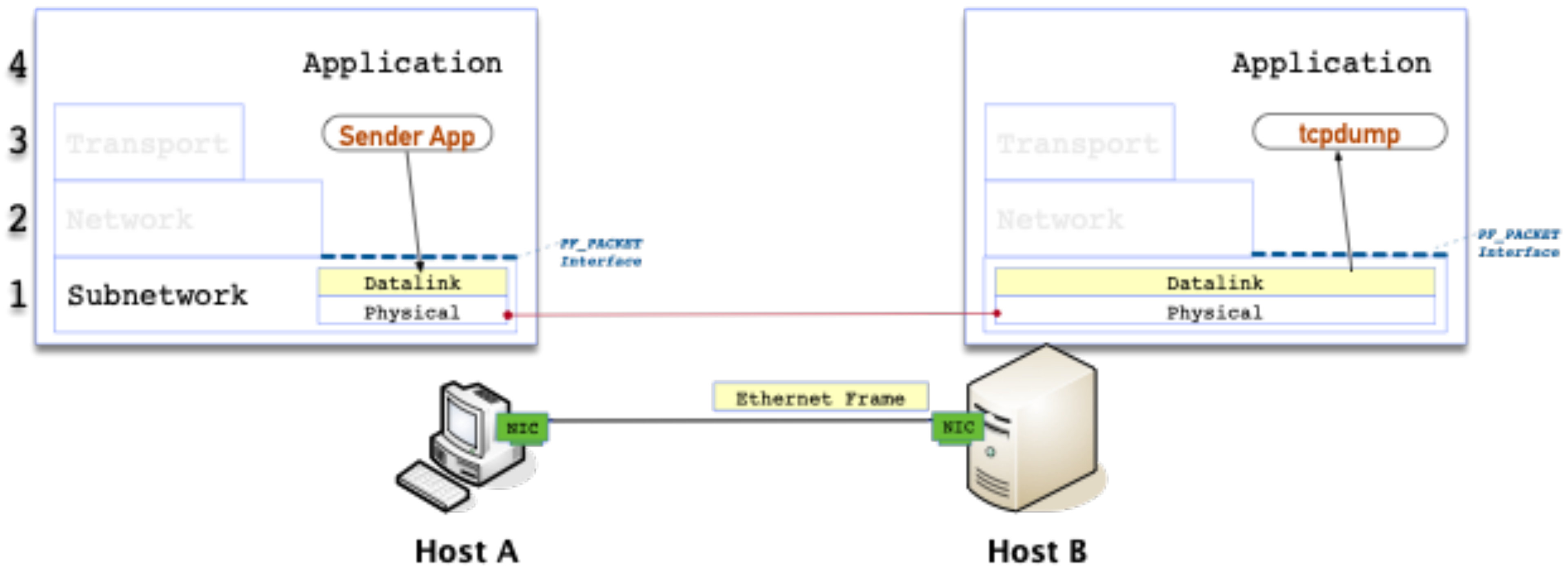
Computer Networks Course, Universidad de León, 2015-2021

Chapter Outline: **IP protocol**

- IP := Internetwork Protocol
  - Packet
  - Addressing
  - Forwarding
    - Longest Prefix Match Algorithm
  - Routing
    - DV Algorithm/RIP protocol
    - Dijkstra Algorithm/OSPF protocol

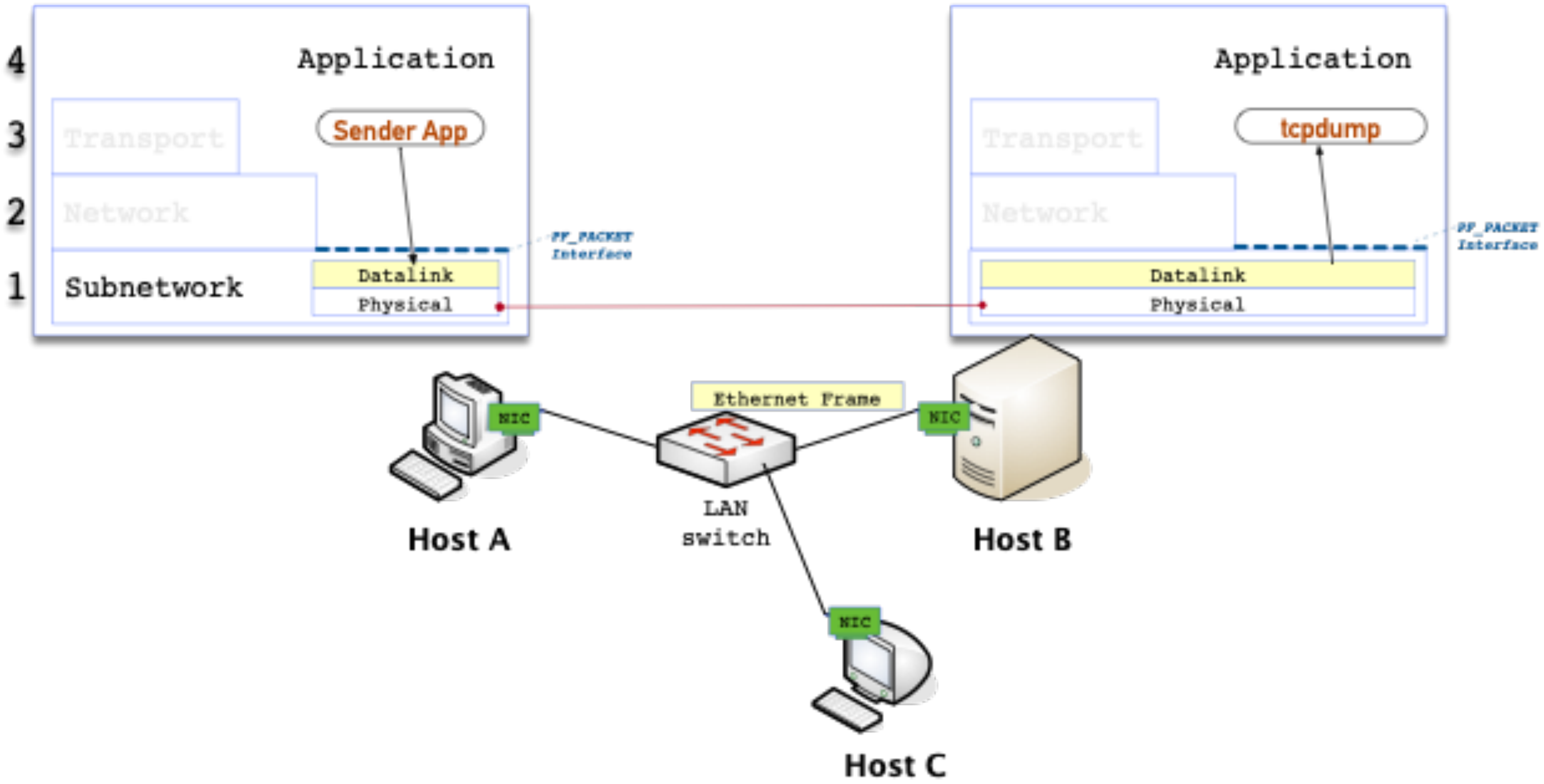
Where are we, now?

- WH5-Practice
  - Send a frame from host A to host B
  - A and B belong to the same network



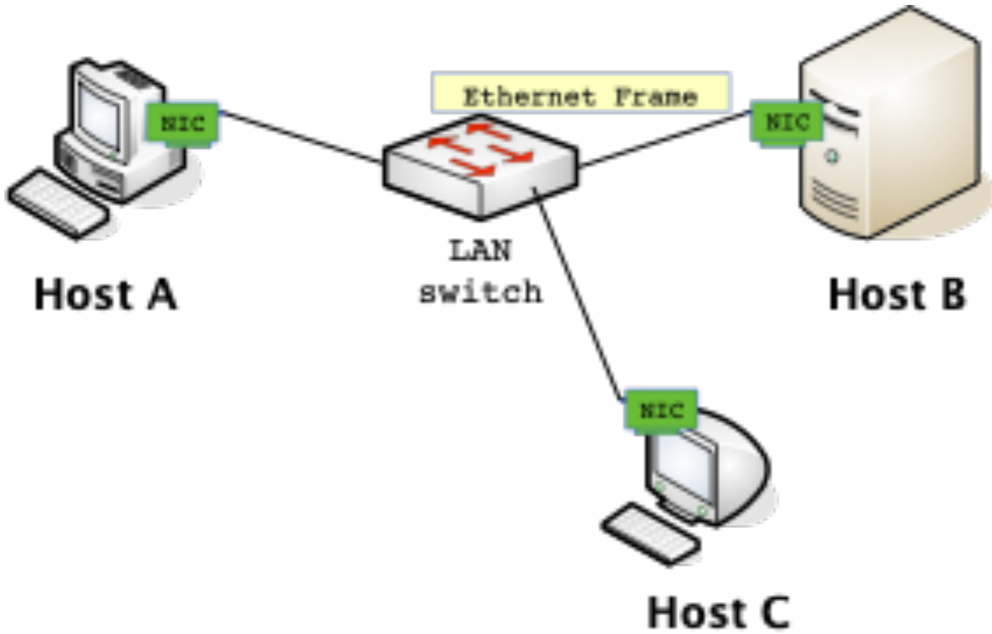
Same context: one network

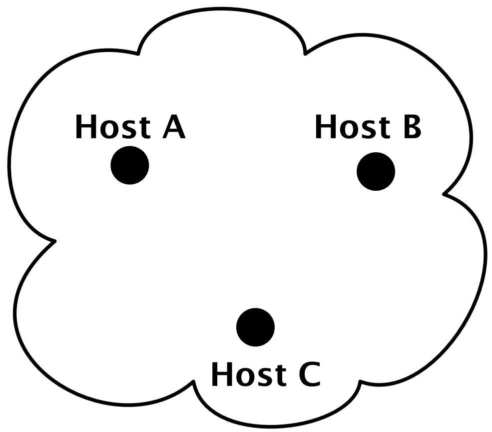
- WH5-Practice
  - Send a frame from host A to host B
  - A and B belong to the same network



Same context: one network

- WH5-Practice
  - Send a frame from host A to host B
  - A and B belong to the same network





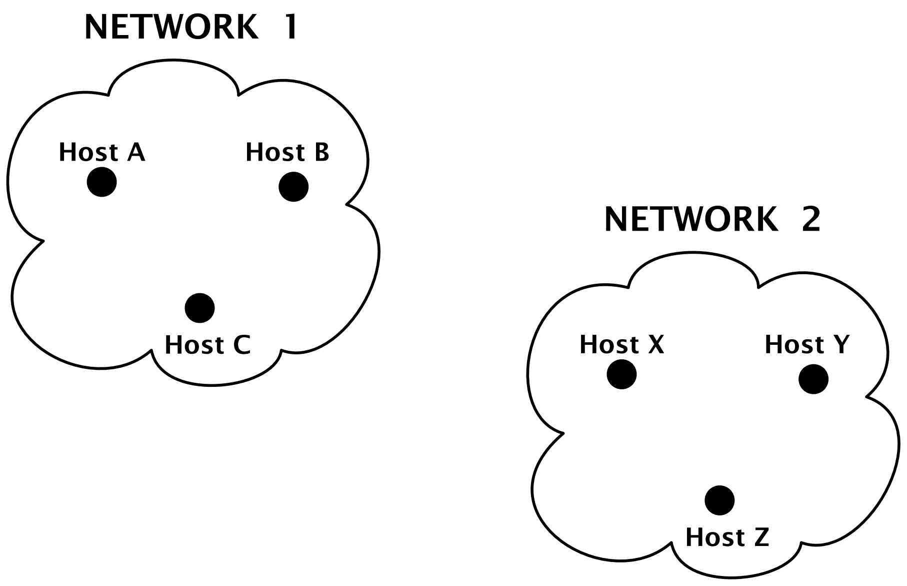
Ethernet frame

Header

Destination Address	Source Address	0x1234	Upper layer payload	CRC32
MAC address of B's NIC	*MAC address **of A's NIC***	*Ethertype*	*Variable-length Payload*	*Error Control*

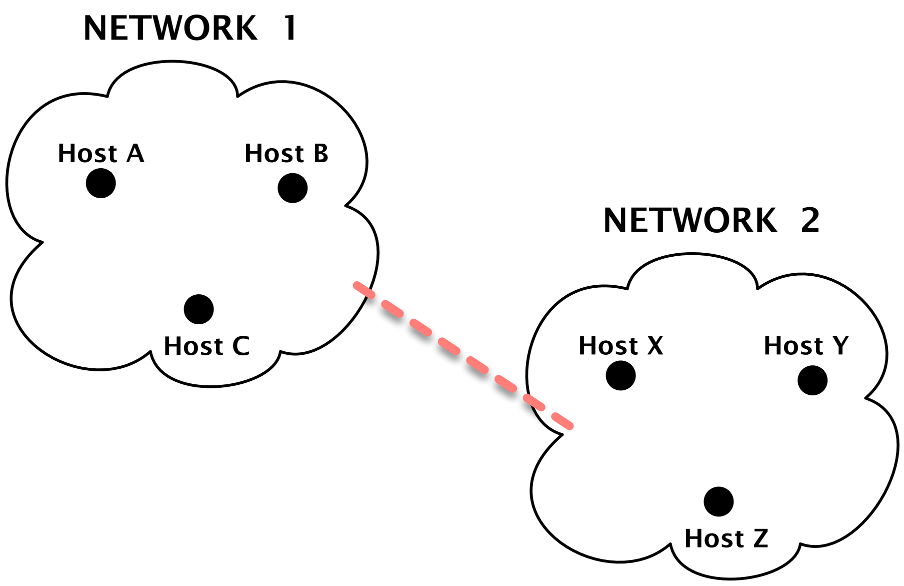
One network scales poorly

- Create many!
  - How to have them connected?



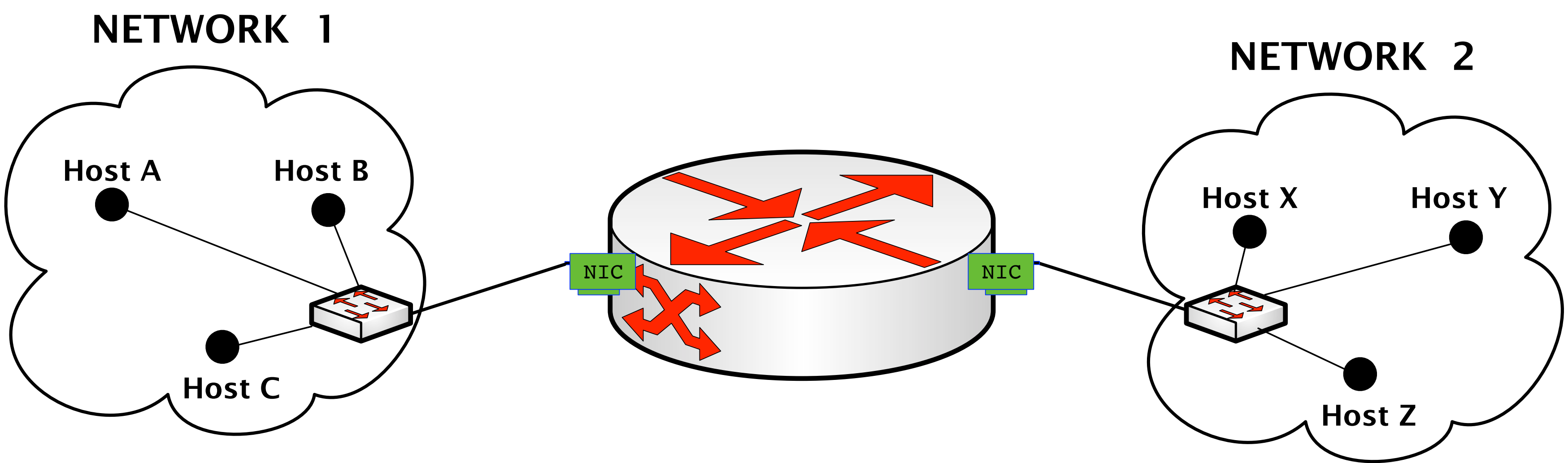
Create two networks

- Connect them directly
  - NO: A single network results!



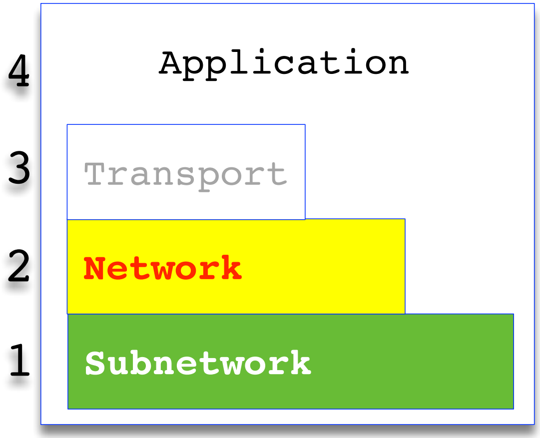
Solution: Create an internetwork

- Communication between networks is accomplished by using an IP Router
  - IP Router: Acts as Gateway between networks

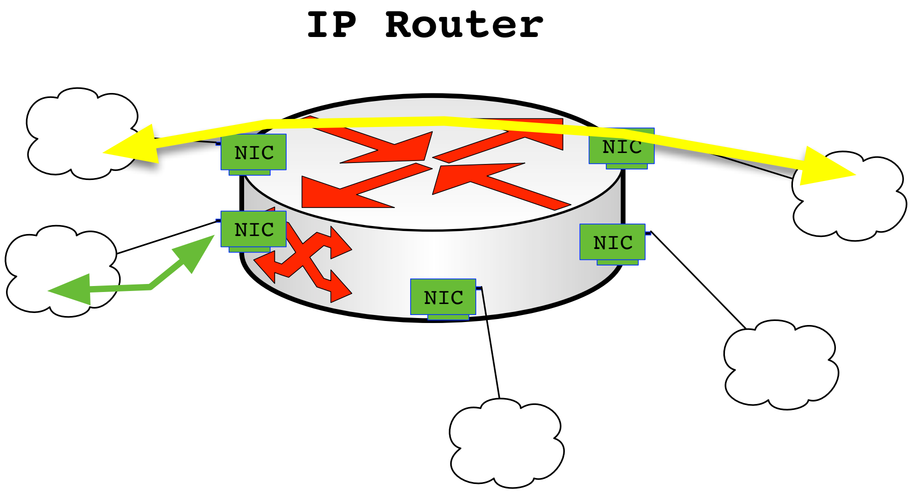


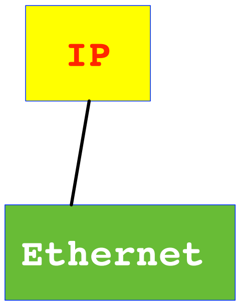
- S/F device
- Layer 2, network layer
- IP Protocol
- IP Packets

IP Router



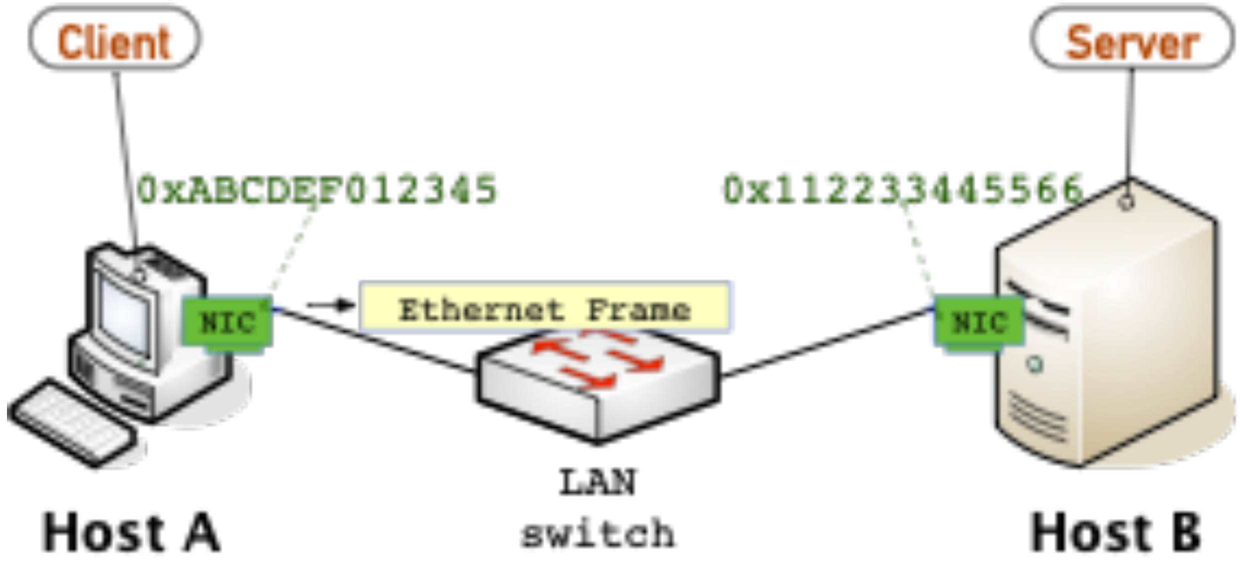
Internet Architecture
At IP Router

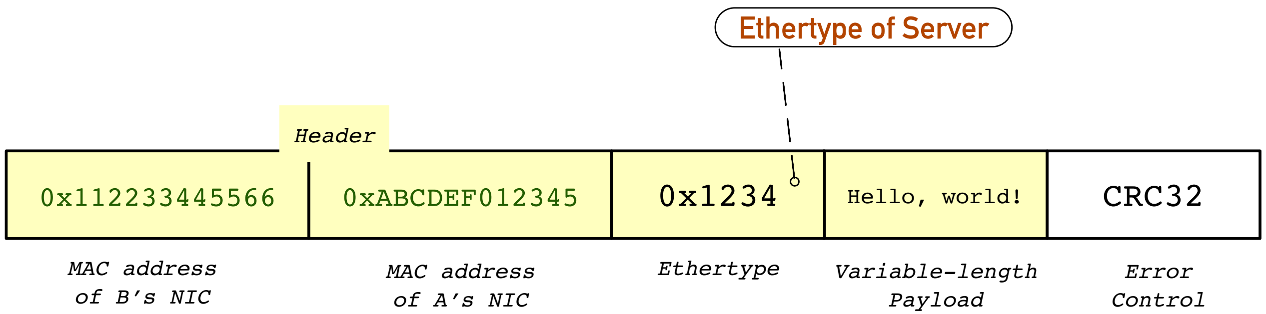




Protocol stack of IP Router

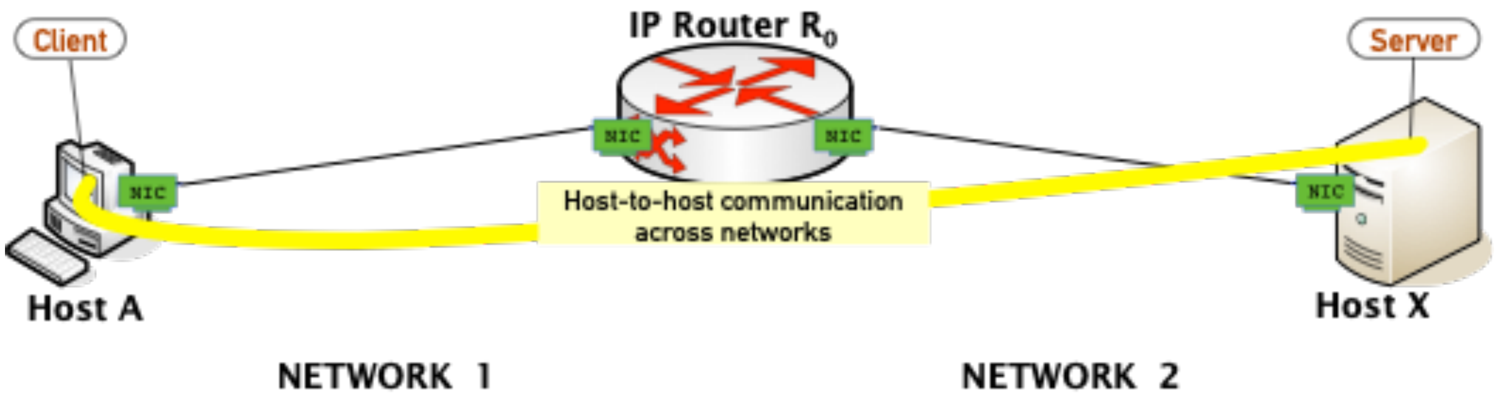
Recall host-to-host communication across one network





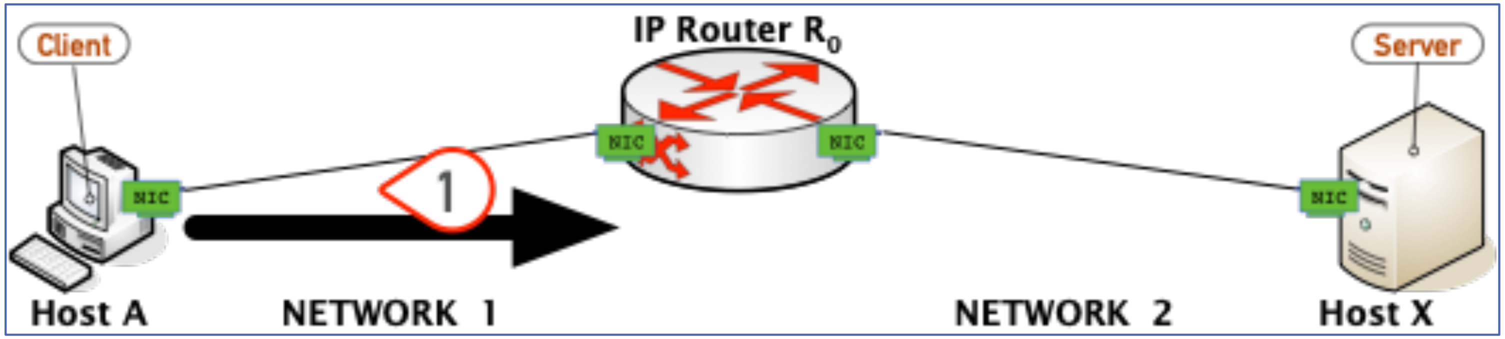
Ethernet frame

Concept: Host-to-host communication across two networks

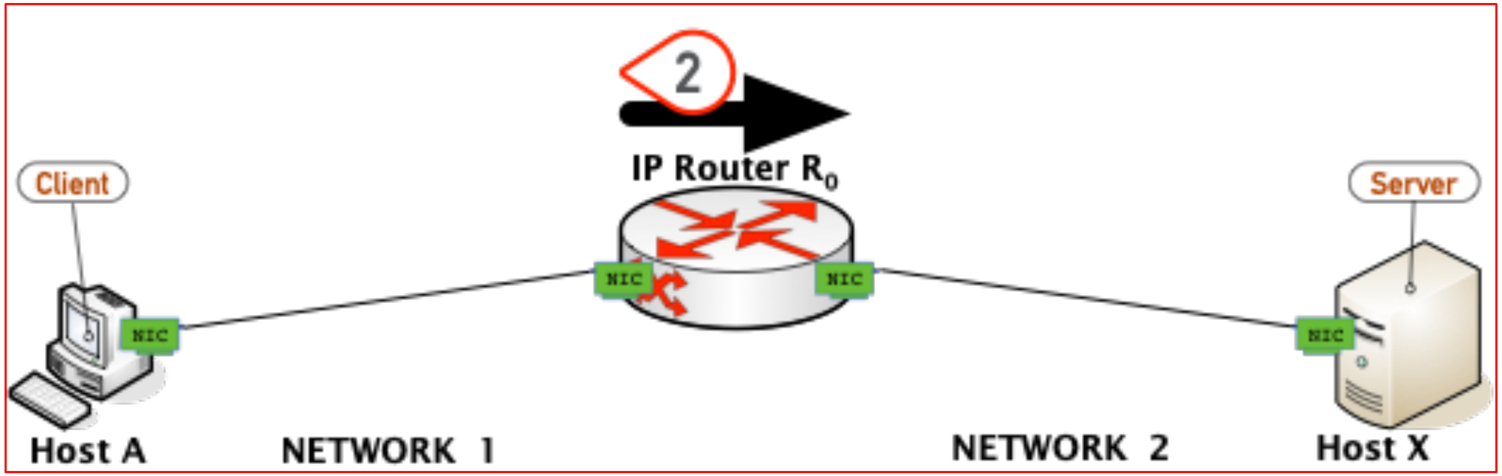


Sending from A to X across an internetwork

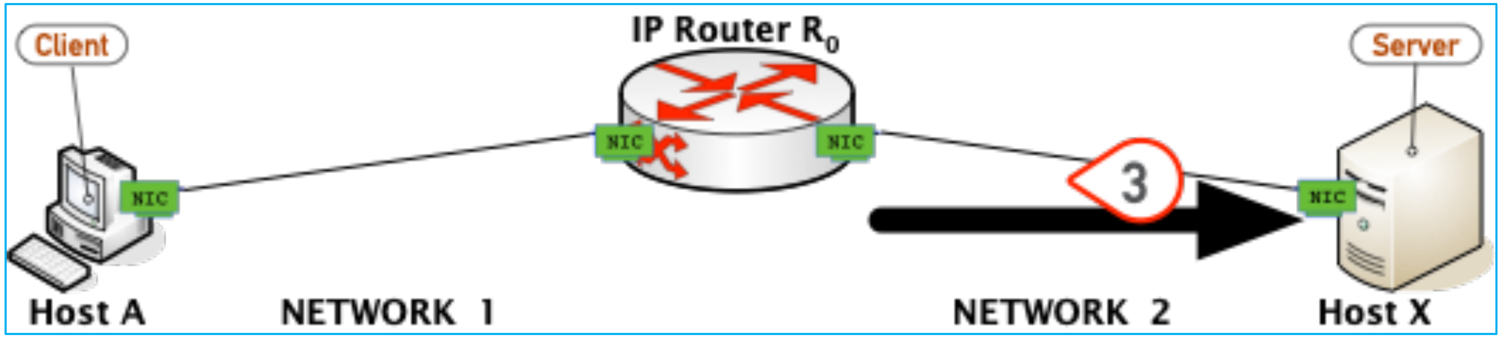
- 1. Host A to R_0



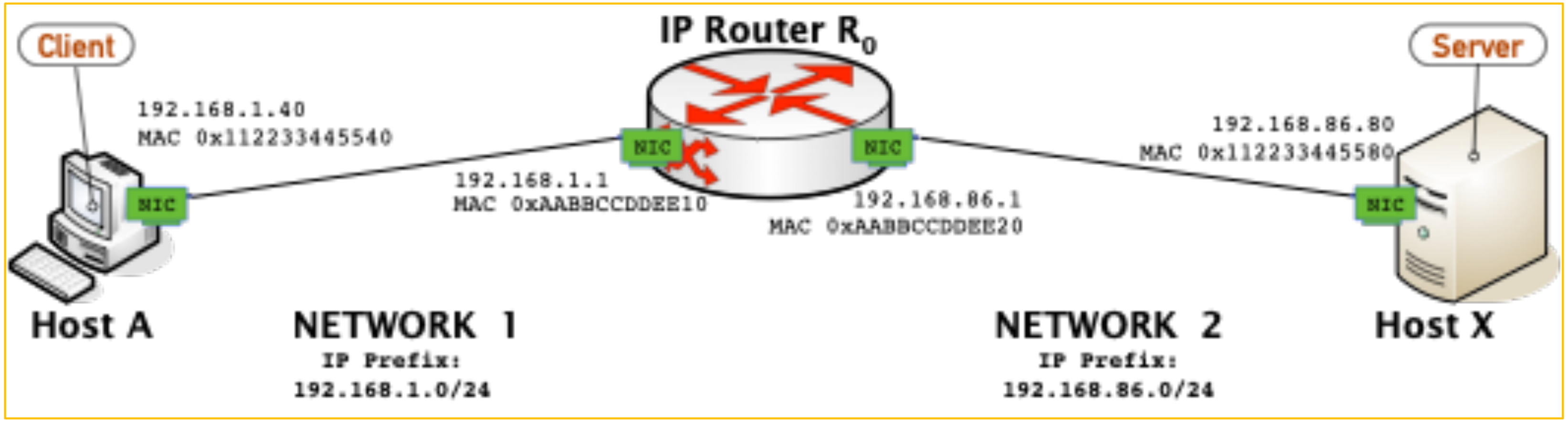
- 2. Forwarding in-router

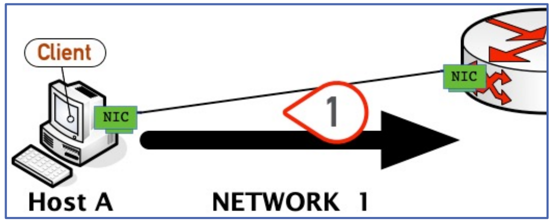


- 3. Router R_0 to Host X

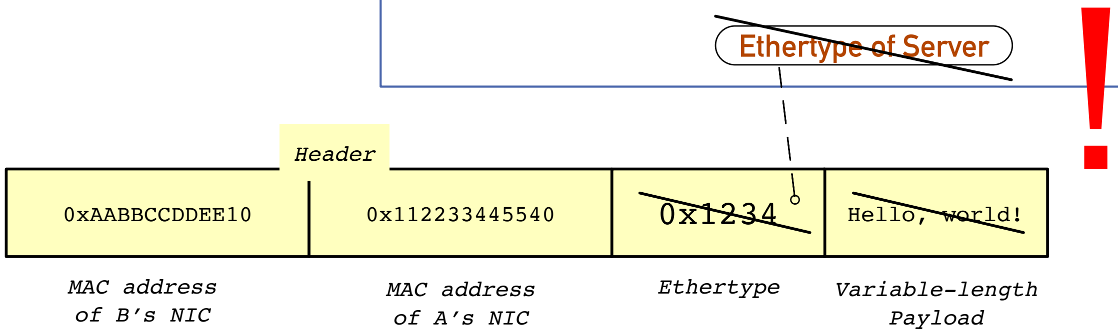


Host A sends to Host X across an internetwork: Phase 1

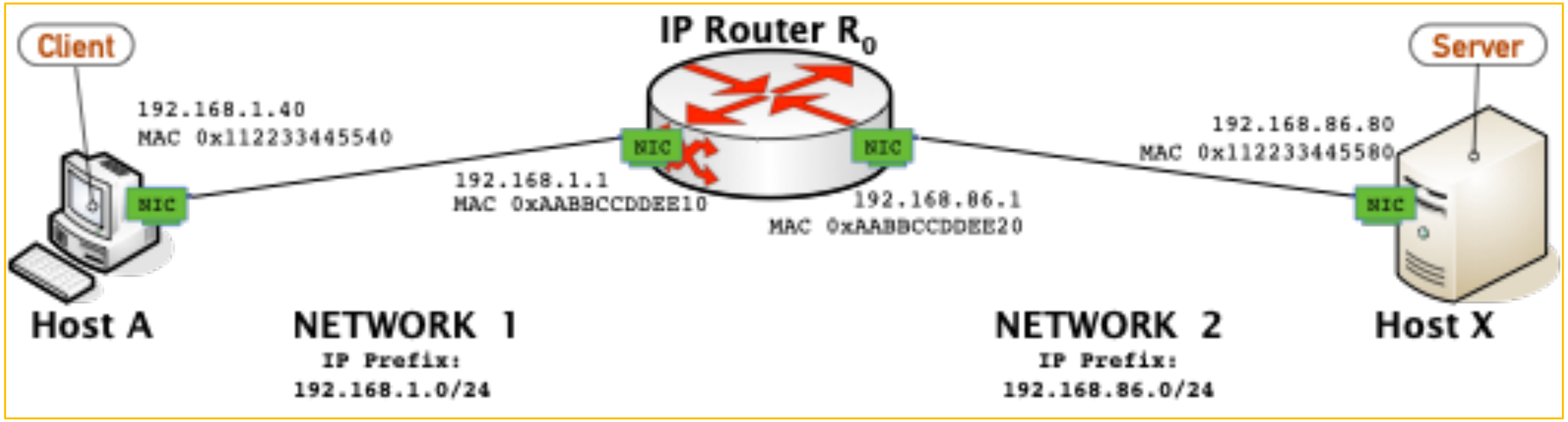


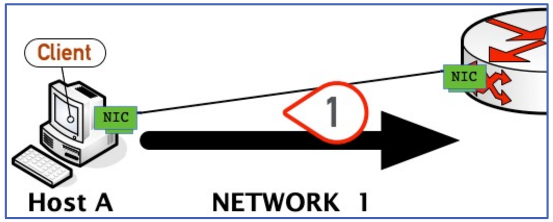


- **Phase 1: Host A sends to R_0**
- Sent frame:
  - DMAC = 0xAABBCCDDEE10
  - SMAC = 0x112233445540
  - Ethertype = ~~ID of receiving App in Host X~~

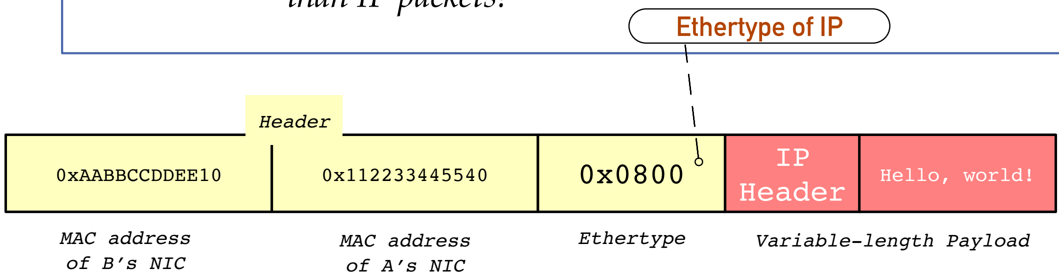


Host A sends to Host X across an internetwork: Phase 1

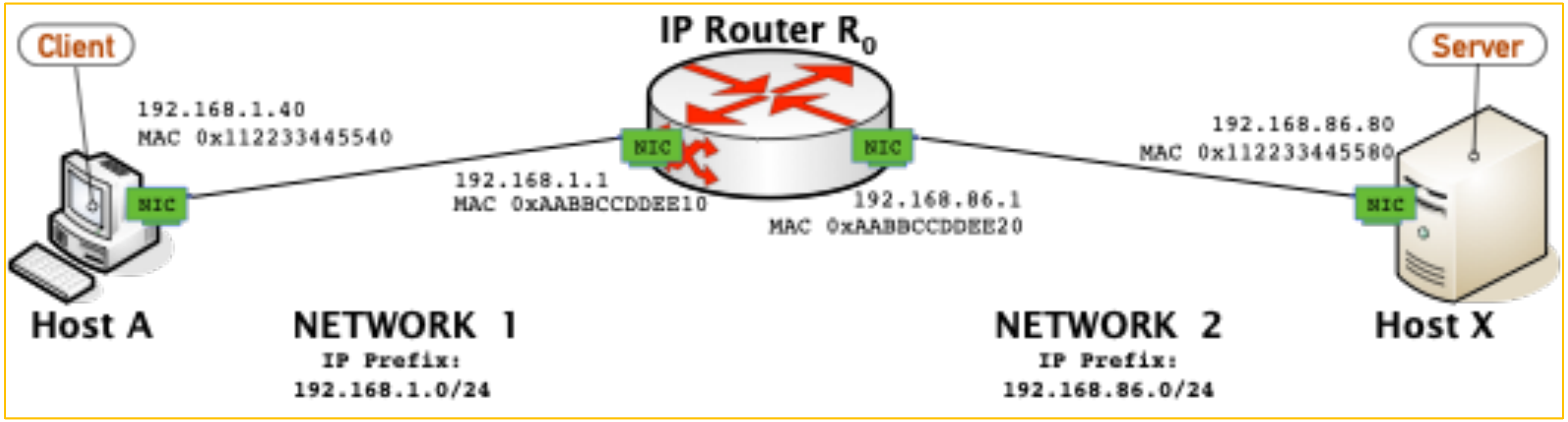


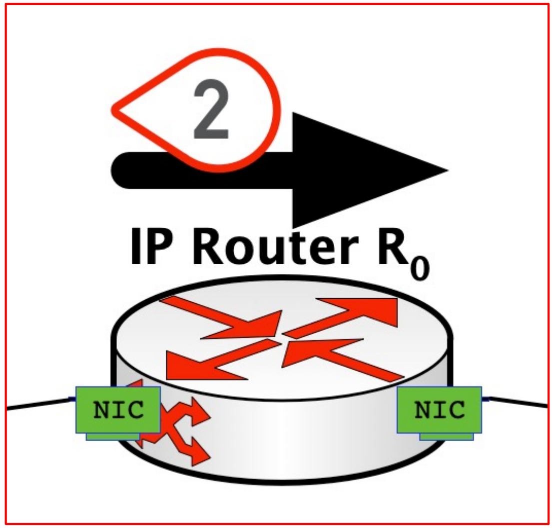


- **Phase 1: Host A sends to R_0**
- Sent frame:
  - DMAC = 0xAABBCCDDEE10
  - SMAC = 0x112233445540
  - Ethertype = ID of <u>IP Protocol</u>
  - Payload = An IP Packet
    - ***R_0 will not be able** to forward subnet payloads other than IP packets!*

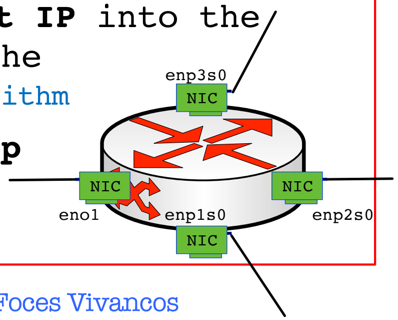


Host A sends to Host X across an internetwork: Phase 2

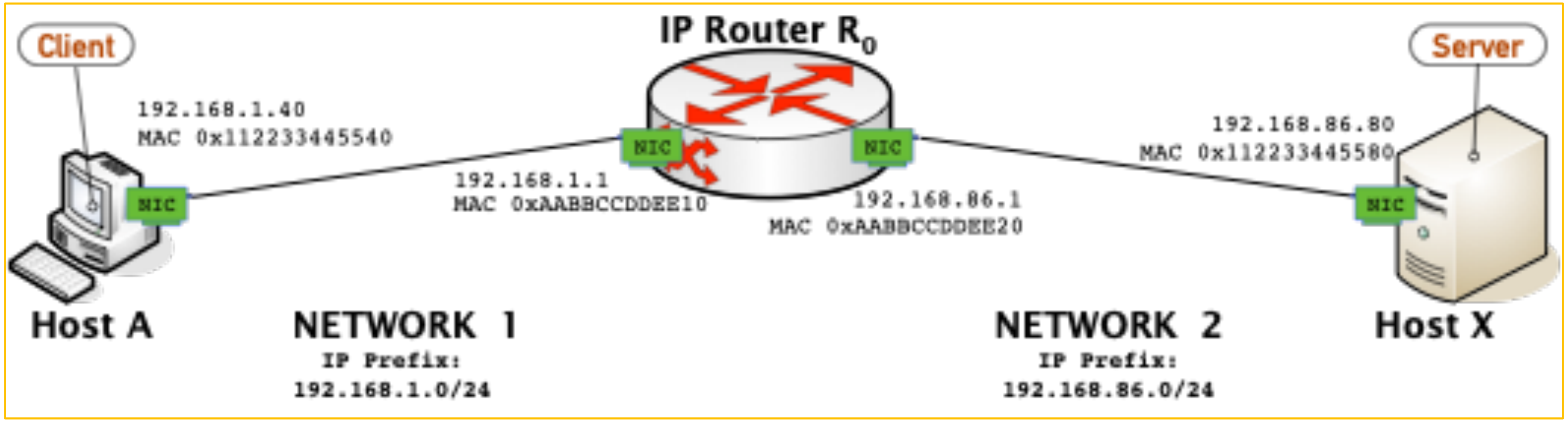


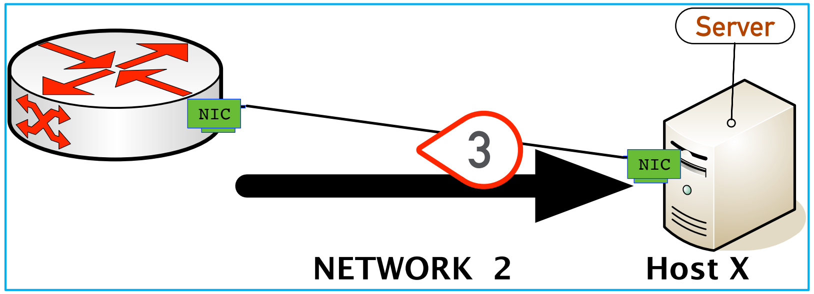


- **Phase 2: Router R_0 forwards <u>IP packet</u>:**
  - Received frame's payload must be an IP Packet
  - Ethertype = 0x0800
  - The IP Packet is deencapsulated from ether frame
  - R_0 looks up Packet's **Dest IP** into the Forwarding Table using the Longest Prefix Matching algorithm
  - Result will tell **next hop**



Host A sends to Host X across an internetwork: Phase 3





- Phase 3: Router R_0 transmits frame to Host X
- Frame sent:
  - `DMAC = 0x112233445580`
  - `SMAC = 0xAABBCCDDEE20`
  - `Ethertype = 0x0800`
  - `Payload = IP packet sent by A to` R_0 `which was deencapsulated by` R_0.

Addresses, MAC and IP

- **IP address** is used for:
  - Locating a host's network in Internet
  - And for locating and identifying the host within its network

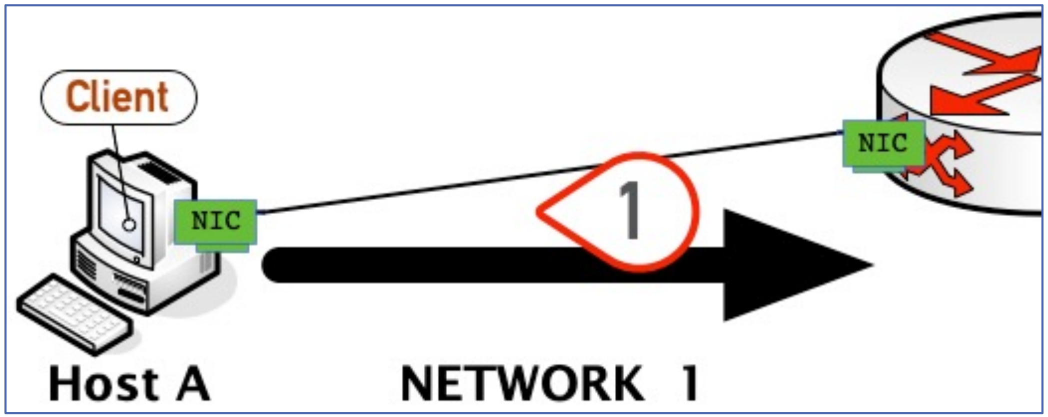
- Finally, communication with the host within its network entails:
  - The host's NIC **MAC address**

- Every time a defective NIC has to be replaced, the MAC changes

Review of communication of Host A to Host X across an internetwork

- Host A must know the IP address of R_0
  - Known as the Default Router of host A!
- However, host A does not know the MAC address of R_0
  - It might even have changed from last communication!

Can Host A find the MAC of R_0 left interface, that is, automatically?

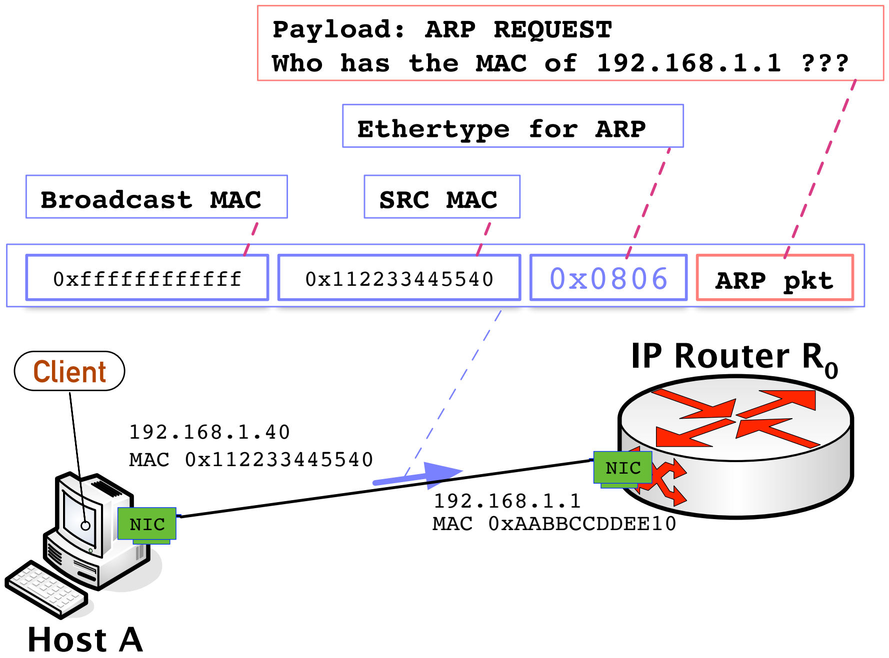


Address Resoultion Protocol

ARP, a protocol ancillary to IP for resolving an IP address into its corresponding MAC address within a single network

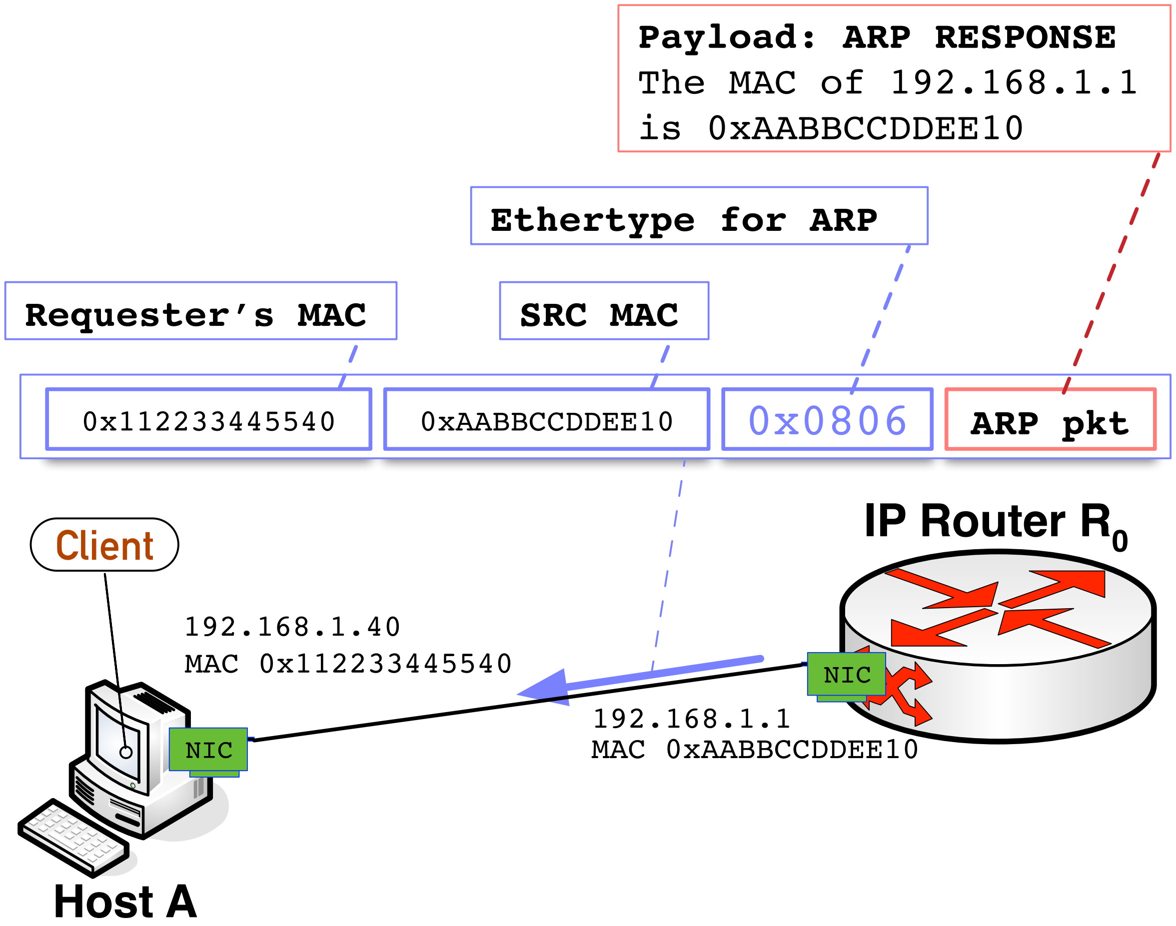
ARP Request

- Host A finds the MAC address of its *default router* R_0 by using ARP
  - 1. A sends ARP request for the IP address of R_0
  - 2. R_0 responds with its MAC address



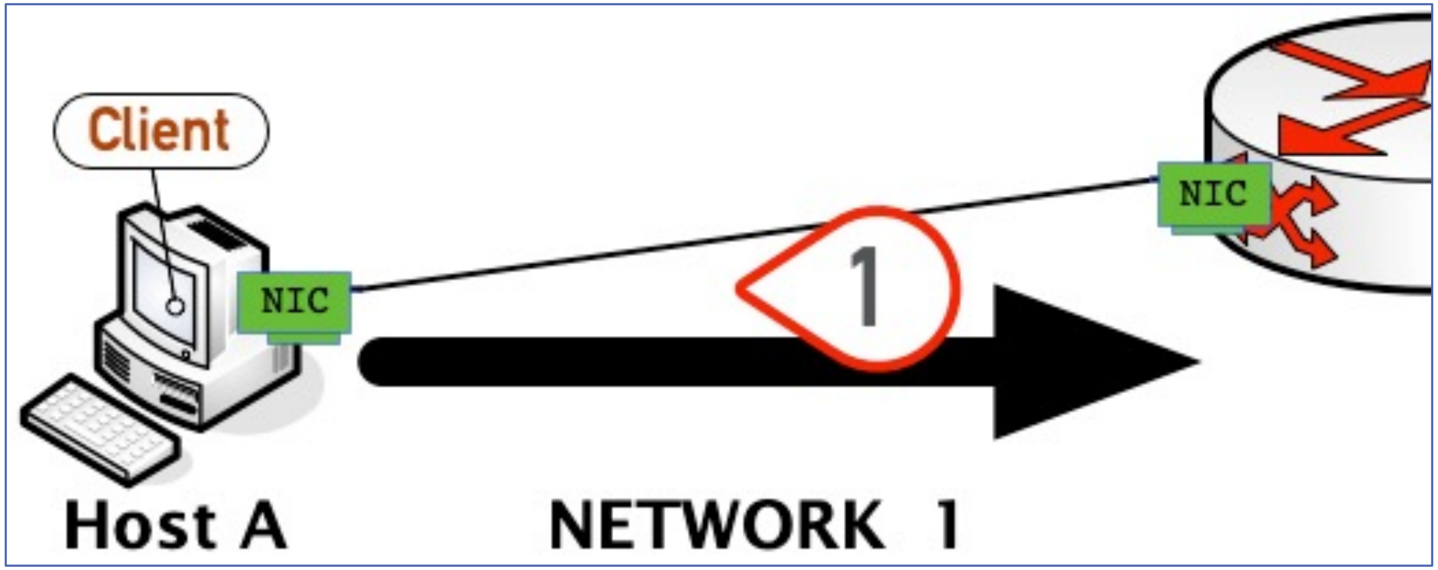
ARP Response

- R0 responds with its MAC address



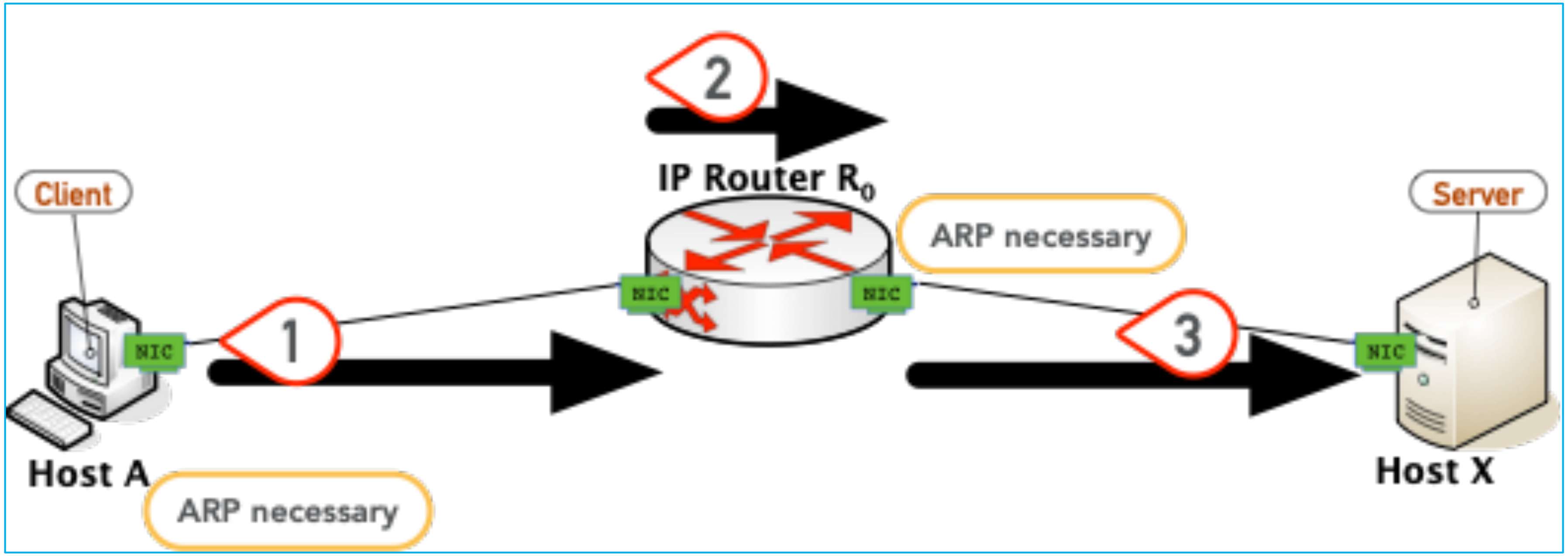
Now, Phase 1 of communication of A with X can continue

- Host A knows R_0 IP address (Default router !)
- After ARP Response, Host A knows the MAC of Ro (Left interface)



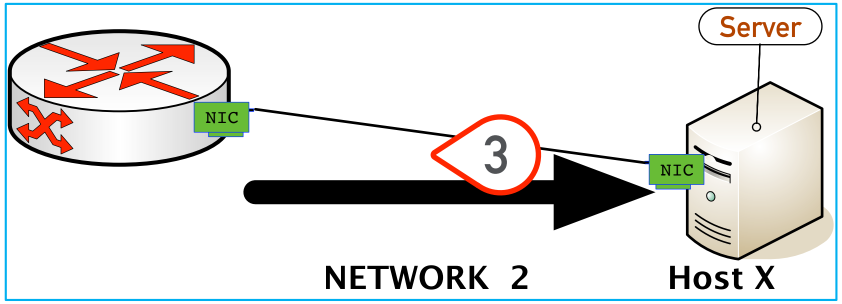
Done.

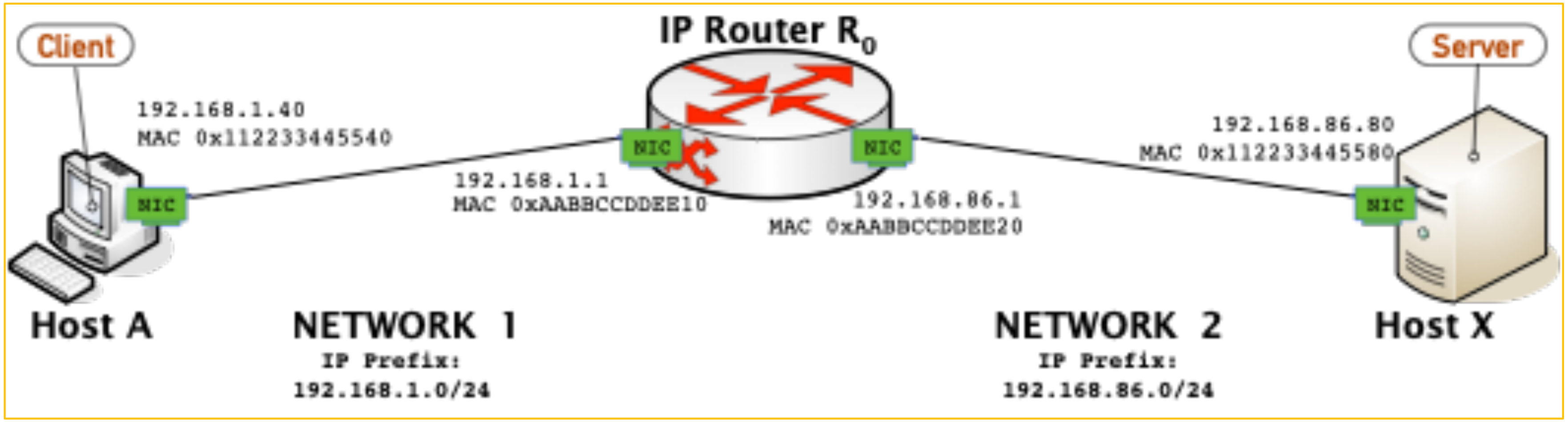
- Phases 1 and 3 entail ARP request/response, as well !
- Host A successfully handed a message to Host X over separate networks



Exercise. Explain arp resolution in Phase 3

- R_0 needs resolving the IP of Host X into its MAC address
- Explain the ARP process as we did earlier
- Provide detail about the full ARP transaction:
  - ARP Request
  - ARP Response



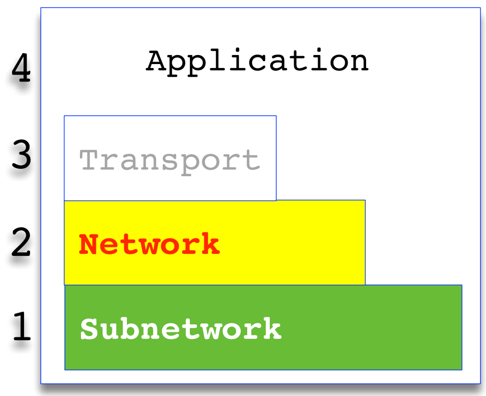


Internetworking with IP

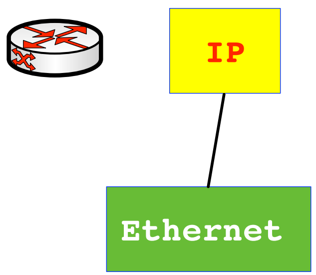
IP must run at every Internet host which includes hosts themselves and IP routers

Internetworking with IP

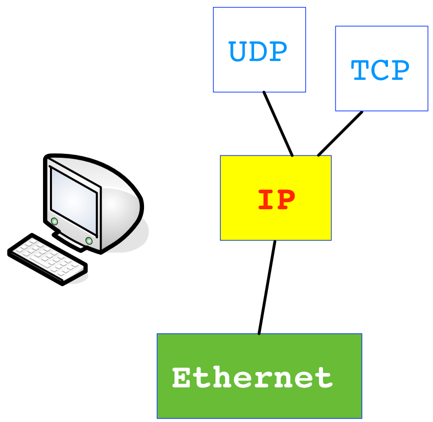
- IP = Internet Protocol
- Key for scalable, heterogeneous internetworks
- It runs on all the hosts and routers
  - Single logical internetwork
- Established by IETF



Internet Architecture



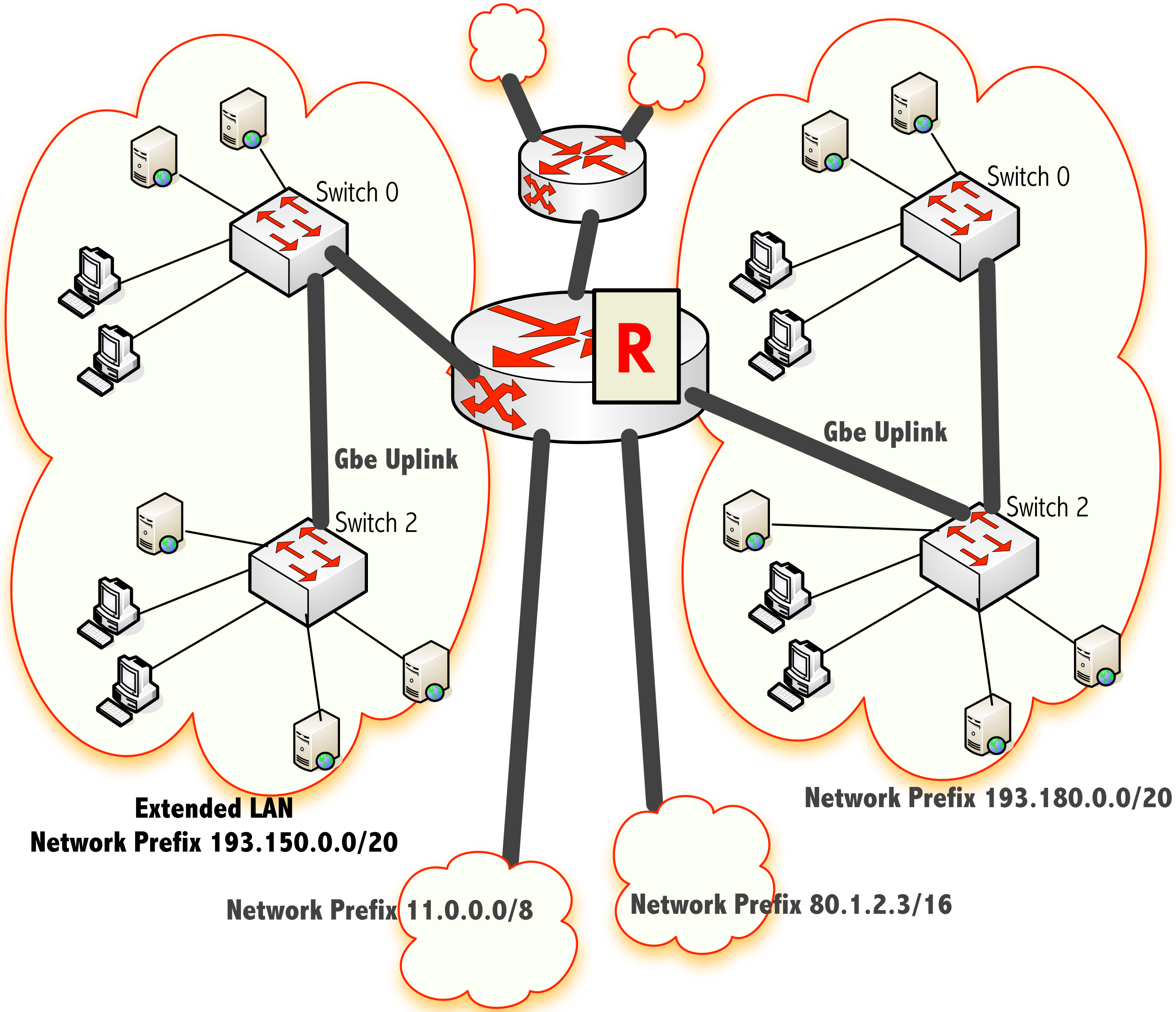
Protocol stack
of IP Router



Protocol stack
of IP Host

Internetworking with IP

- Each IP network must have a unique number known as Network Prefix
  - Example: Prefix 193.146.96.0/20 maps Unileon!
- **Router R** has these prefixes in Forwarding Table
  - 193.150.0.0/20
  - 11.0.0.0/8
  - 80.1.2.3/16
  - 193.180.0.0/20
  - The prefix of its Default Router
- Not *final* hosts, but prefixes
- Recall switches
  - Individual MAC in Fwd Table!



IP Service Model

- Connectionless and unreliable

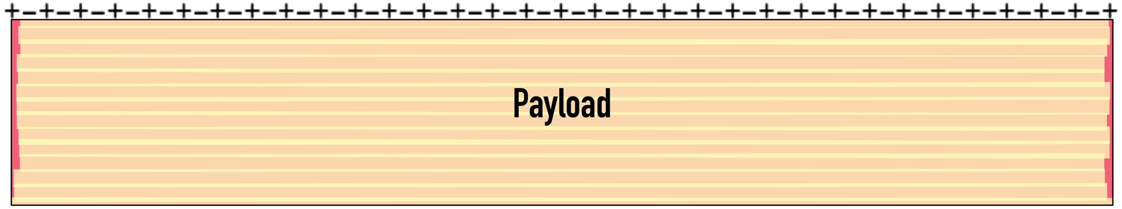
- Best-effort
  - Routers can drop packets
    - Packet loss
  - Routers can reorder packets
  - Routers can erroneously duplicate packets
  - Routers can delay packets
    - Queuing delays

- Global **Addressing** scheme
  - IP addresses
  - For locating and identifying hosts
    - Decimal Dot Notation (DDN): 193.146.101.46

IPv4 packet format

- Version
  - IPv4
  - IPv6 (For future)
- IHL: number of 32-bit words in header
- TOS: Type of Service (For QoS)
- Total Length: number of bytes in this packet
- FRAGMENTATION
  - Ident (16)
  - Flags (3)
  - Offset (13)
- TTL: Max. number of hops this datagram is permitted to cross
- Protocol: Multiplexing Key
  - Examples: TCP = 6, UDP = 17, ICMP = 1
- Checksum (16): of header only
- Destination IP Address (32 bits)
- Source IP address (32 bits)

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|Version|  IHL  |Type of Service|          Total Length         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|         Identification        |Flags|      Fragment Offset    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Time to Live |    Protocol   |         Header Checksum       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                       Source Address                          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    Destination Address                        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    Options                    |    Padding    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```



Verbatim copy from © IETF RFC 791

IP addressing principles

- IP addresses must be unique across the entire Internet

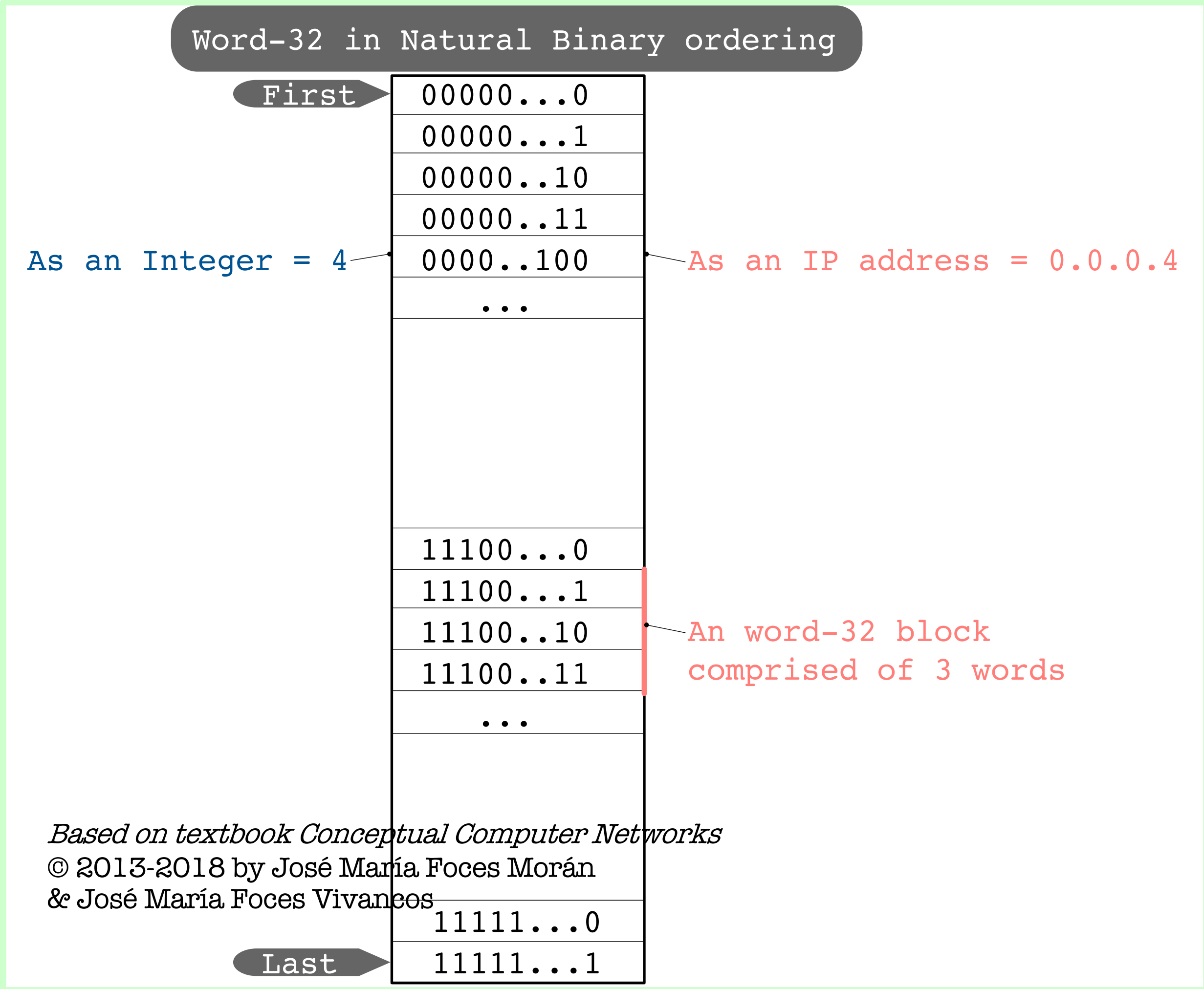
- IPv4
  - 32 bits wide, 2^{32} possible IP addresses
  - Not all may be used for numbering hosts
  - IP address assignment presents some inefficiencies

- Hierarchical. Every IP address contains two parts:
  - Network number in the Most Siginifcant bits
  - Host number in the the Least Significant Bits

- Usually, IP addresses are denoted by using DDN (Decimal Dot Notation):
  - 10.3.2.4
  - 128.96.33.81
  - 192.12.69.77

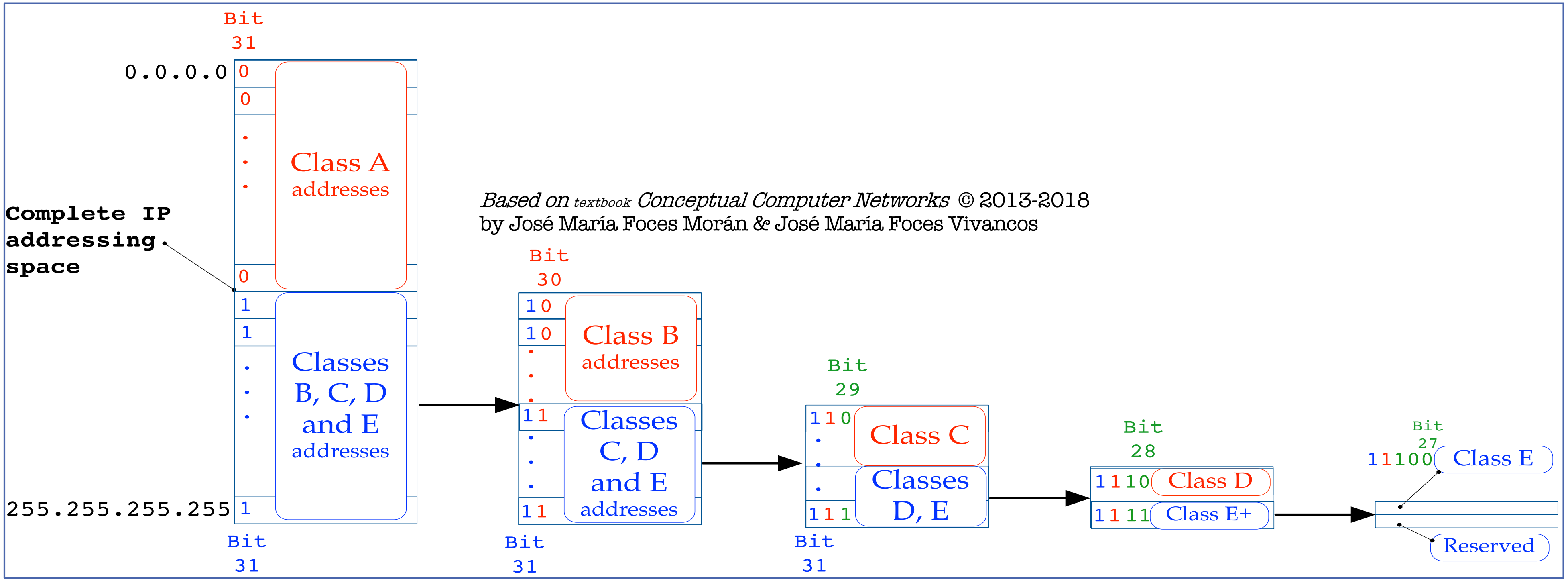
The full IP addressing space

- IPv4 address
  - 32 bits
  - 2^{32} max IP addresses available
- Binary representation:
  - Non-negative integers



Evolution of IP addressing

- Classful Addressing
  - Original technique
    - Divide addressing space into successive halves
  - Inefficient
  - Obsolete



Classful addressing, inefficient

- Class A
  - Resulting IP blocks: 128
  - Size of each network block: $2^{32-8} = 2^{24} = 16777216$ addresses
  - Giant size
    - Very inefficient

```
Class A

Network bits                         Host bits
   0000 0000.0000 0000.0000 0000.0000 0000


Class B

Network bits                         Host bits
   1000 0000.0000 0000.0000 0000.0000 0000


Class C

Network bits                         Host bits
   1100 0000.0000 0000.0000 0000.0000 0000
```

Classful addressing, inefficient

- Class B
  - Resulting IP blocks is: $2^{16-2} = 2^{14} = 16384$
  - Size of each network block: $2^{32-16} = 2^{16} = 65536$ addresses

```
Class A

Network bits                          Host bits
   0000 0000.0000 0000.0000 0000.0000 0000

Class B

Network bits                          Host bits
   1000 0000.0000 0000.0000 0000.0000 0000

Class C

Network bits                          Host bits
   1100 0000.0000 0000.0000 0000.0000 0000
```

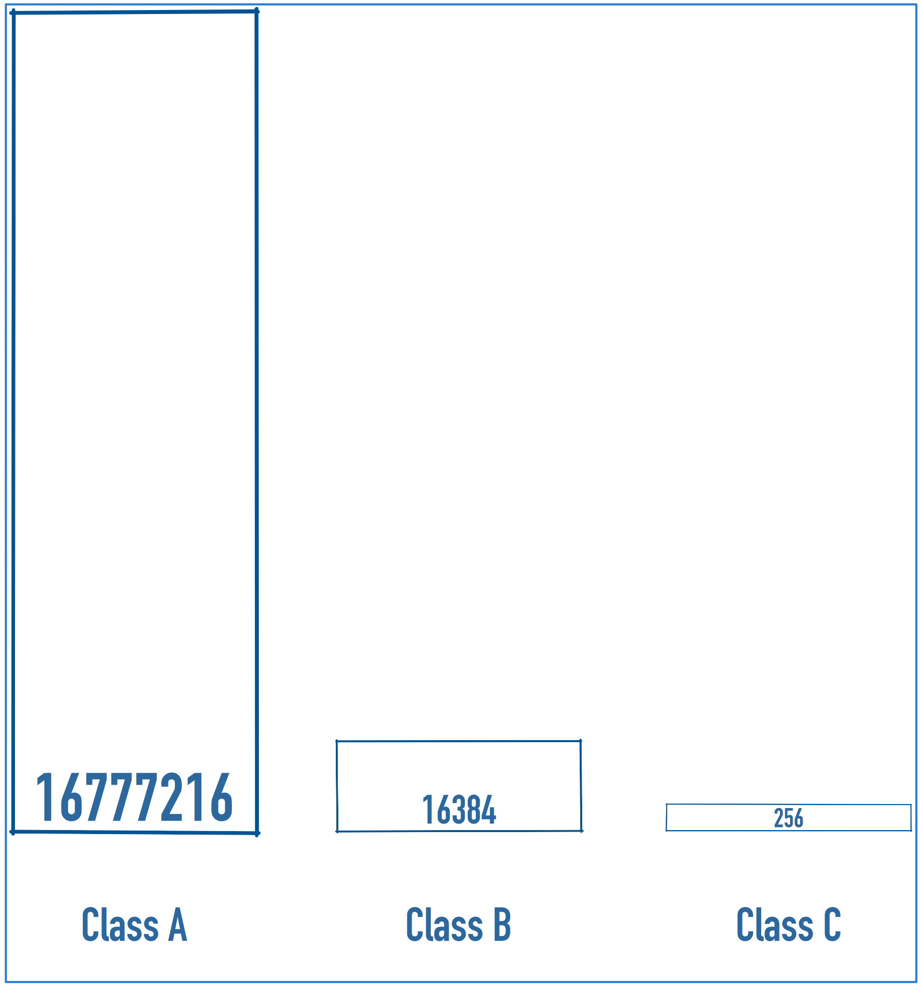
Class C: exercise

- Class C
  - Number of networks?
  - Number of addresses in each network

```
Class A

Network bits                                  Host bits
   0000 0000.0000 0000.0000 0000.0000 0000


Class B

Network bits                                  Host bits
   1000 0000.0000 0000.0000 0000.0000 0000


Class C

Network bits                                  Host bits
   1100 0000.0000 0000.0000 0000.0000 0000
```

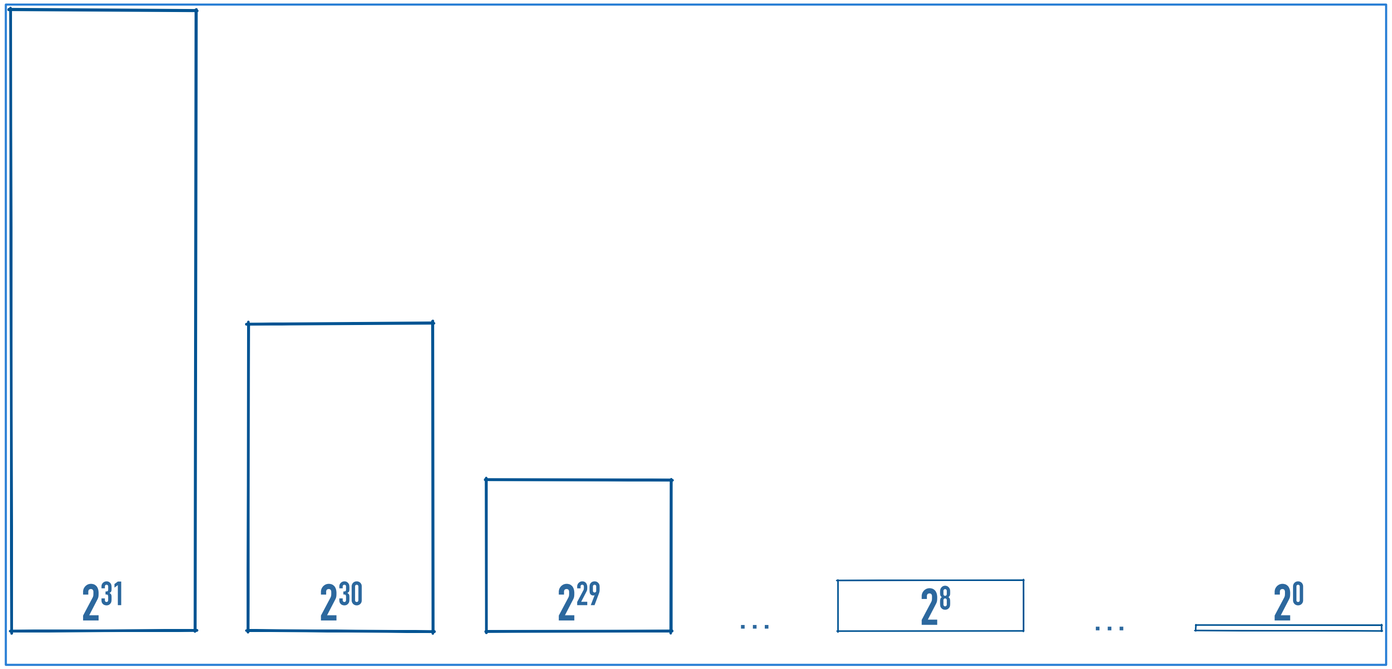
Classful addressing, summary

- Only three network sizes available
  - Class A
    $2^{24} = 16777216$
  - Class B
    $2^{16} = 65536$
  - Class C
    $2^{8} = 256$
- Little flexibility



Classless Inter Domain Routing = CIDR

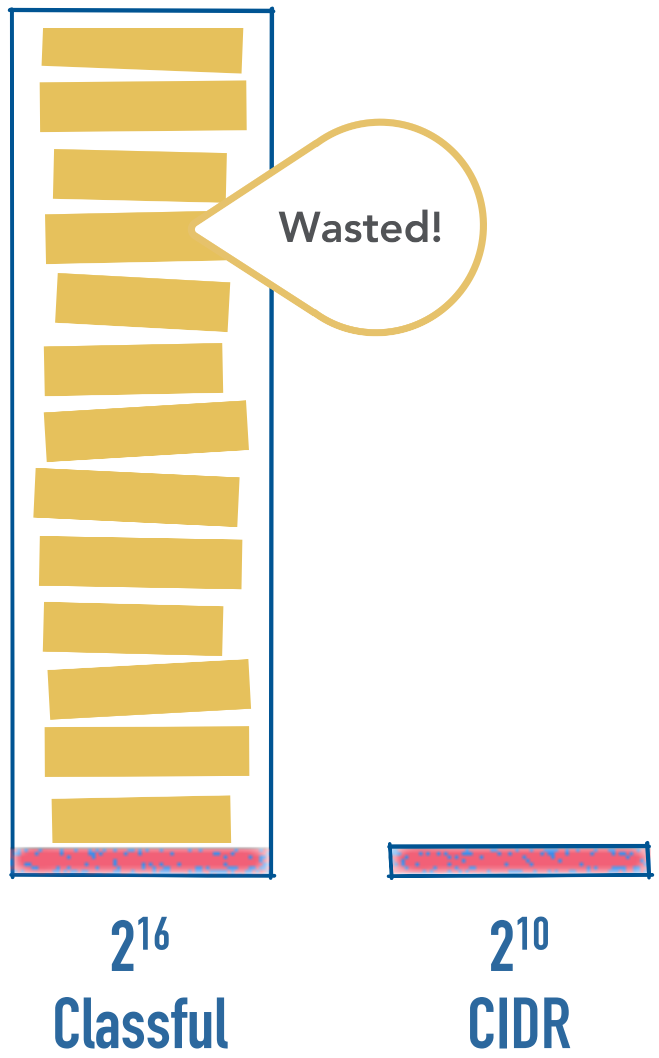
- The solution to the lack of efficiency of Classful Addressing
  - Specified in RFC 4632
  - CIDR is pronounced like *Cider*
- An IP address block can have any 2^n size (n integer)
  - Not only 2^{24}, 2^{16} and 2^8



CIDR is efficient

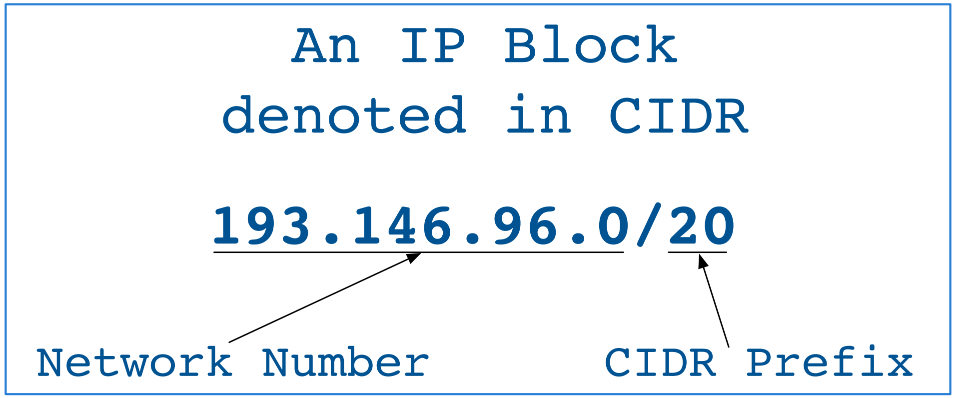
- Unileon network public IP addressing uses CIDR
  - $2^{10} = 4096$ addresses

- With Classful addressing Unileon would have had to purchase a full B-class IP block:
  - $2^{16} = 65536$ addresses



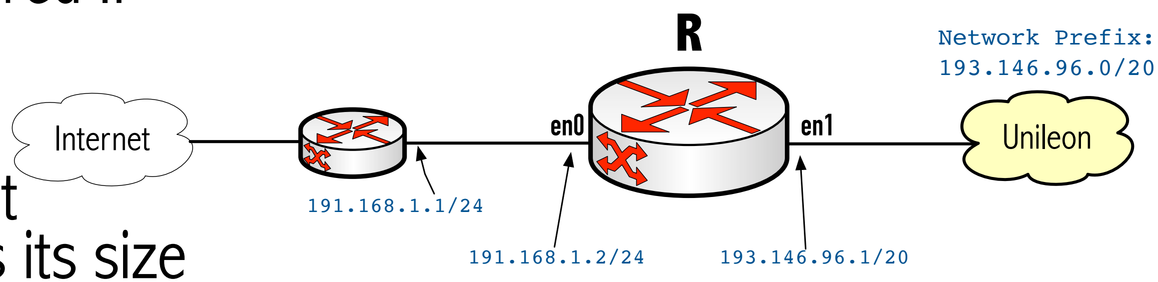
An IP block is represented by a Prefix Number

- Permits specifying the desired IP block size from among 2^n
- The Router R which has a direct connection to the considered network sets its size
- Example:
  - Network number = 193.146.96.0
  - Desired IP Block size = 4096
    - $\log_2 4096 = 12$; $4096 = 2^{12}$
      - 12 Host bits
      - 32 – 12 = 20 Network bits
      - CIDR Prefix = 20



Each network receives a CIDR Prefix

- Permits specifying the desired IP block size from among 2^n
- Router R which has a direct connection to network sets its size



- Example:
  - Network number = 193.146.96.0
  - Desired IP Block size = 4096
    - $\log_2 4096 = 12; 4096 = 2^{12}$
      - 12 Host bits
      - 32 – 12 = 20 Network bits
      - CIDR Prefix = 20

R forwarding table

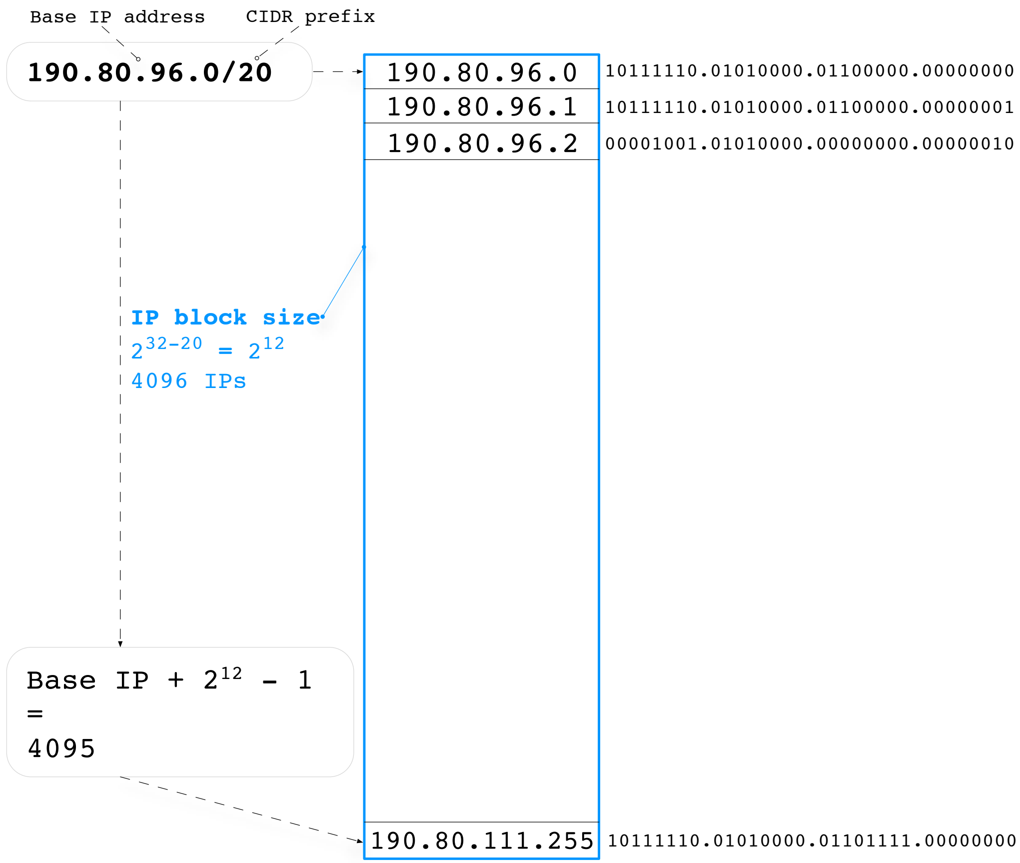
Destination Network Prefix	Next hop	Interface
193.146.96.0/20	Direct	en1
192.168.1.0/24	Direct	en0
Default	192.168.1.1	en0

Partitioning the IP space: The concept of IP block

- IP block

  A block of 2^n consecutive IP addresses which base IP address **r** (The first) is divisible by 2^n

  - 1. **Size** = 2^n
  - 2. **First** integer r of block: r mod $2^n = 0$
    - Otherwise: *First integer must be aligned on a 2^n boundary*



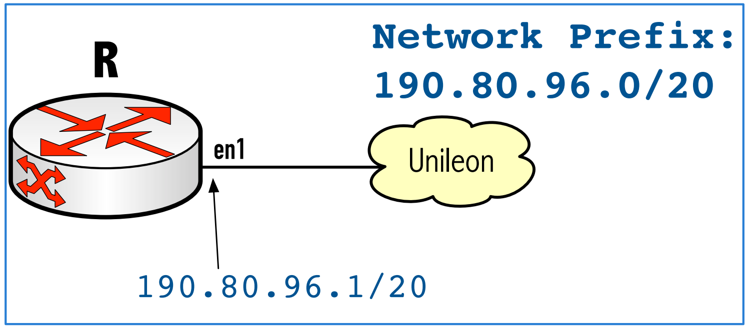
Example of an IP Block housing 4096 addresses

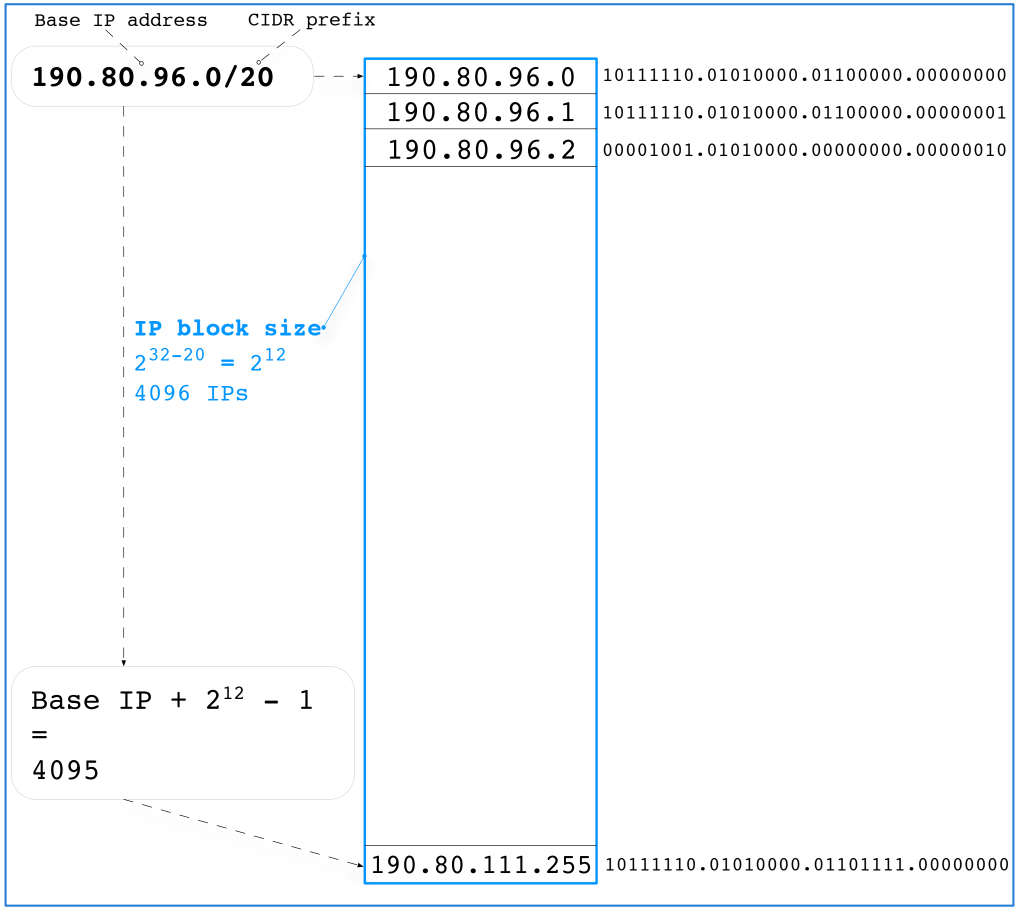
IPv4 address expressed in DDN (Dot Decimal Notation): 190.80.96.0/20

IPv4 address uses 32 bits

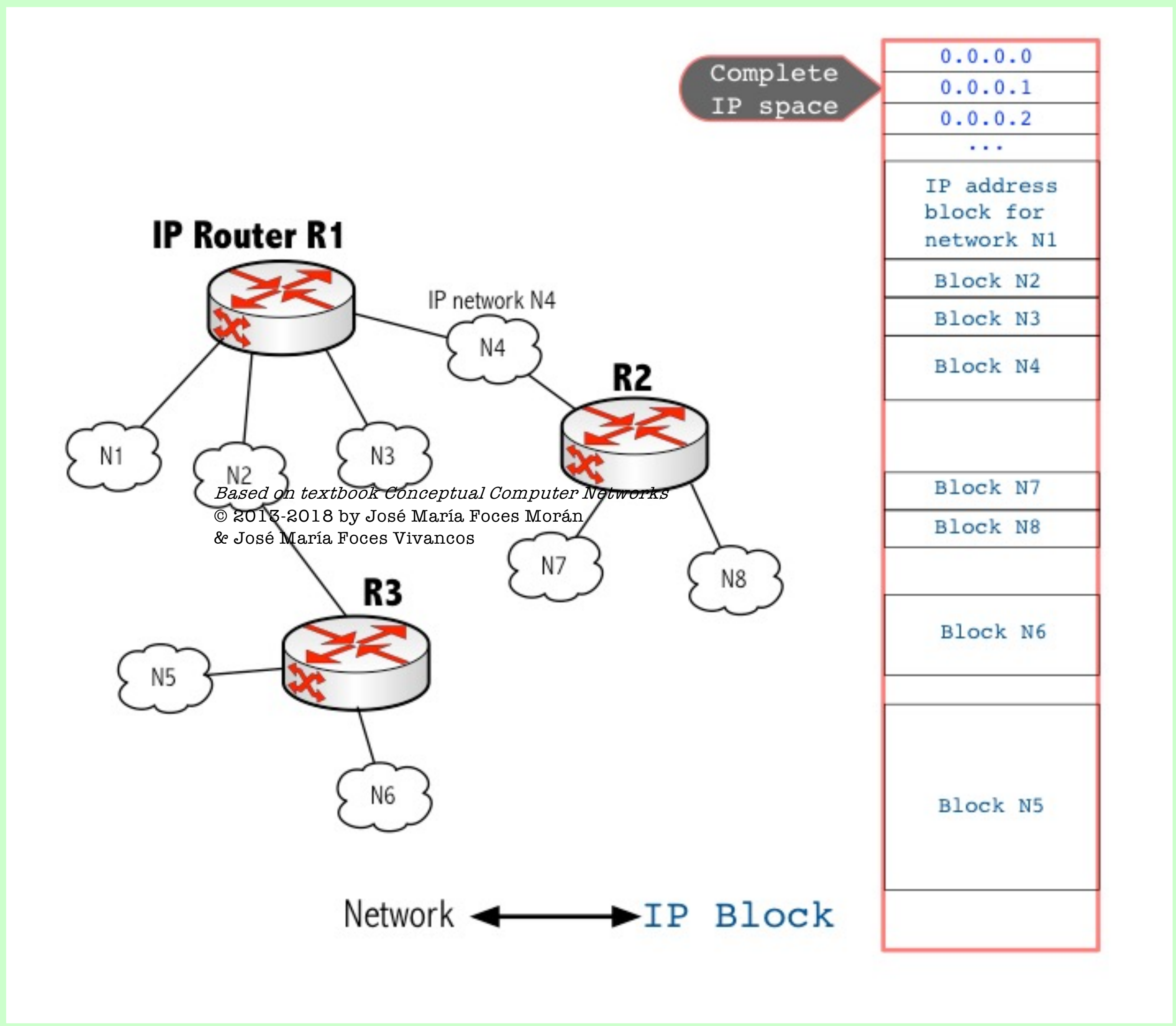
00001001.01010000.0000	0000.00000000
CIDR prefix is /20: 20 high bits are used for representing the IP block number (Base address or Network Number)	These 32-20 bits represent the IP block size: $2^{(32-20)}$ = 4096 Ip addresses

Network numbering = CIDR Network Prefix = IP Block

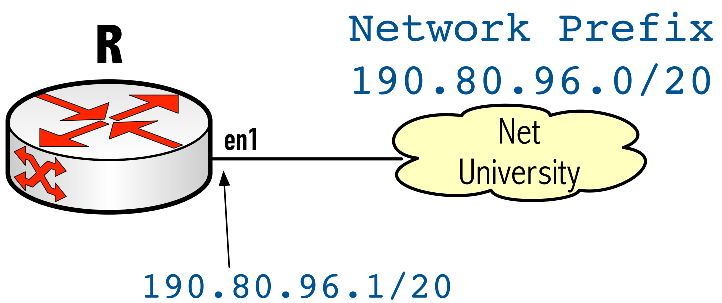


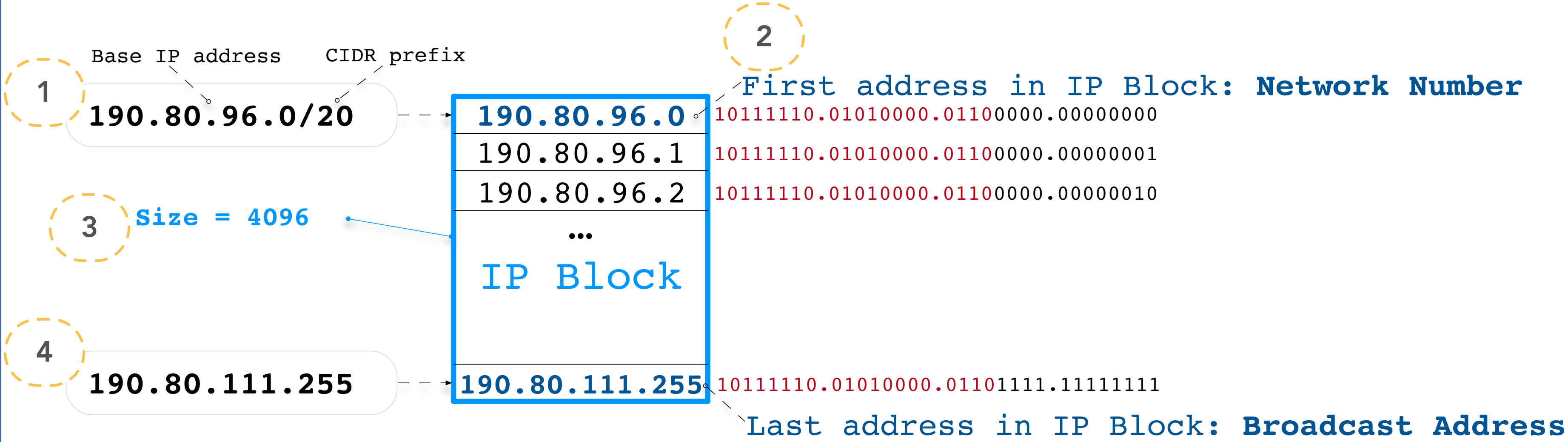


Each network must be mapped to one IP block



General IPv4 conventions for IP Blocks





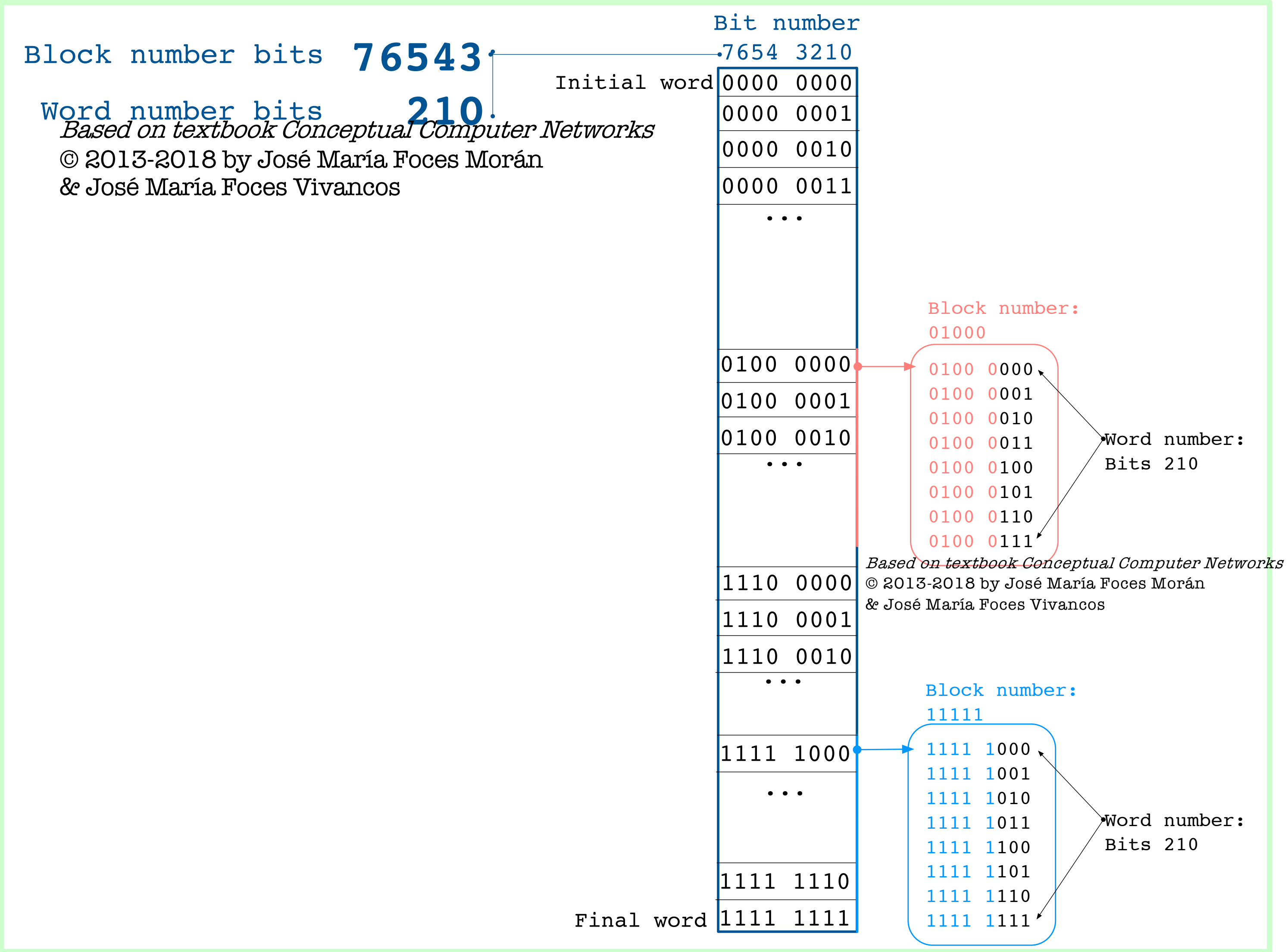
1 Giventhe *Network Prefix 190.80.96.0/20*, calculate Network Number and Broadcast Address

2 Calculate Network number: **Network Prefix & Mask** where
Mask = [CIDR prefix bits 1][32-(CIDR prefix) bits 0] = [**20** bits 1][32-20 bits 0] = $255.255.240.0_{10}$

3 Calculate IP block size: $2^{32-20} = 2^{12}$; Size = 4096 IPs

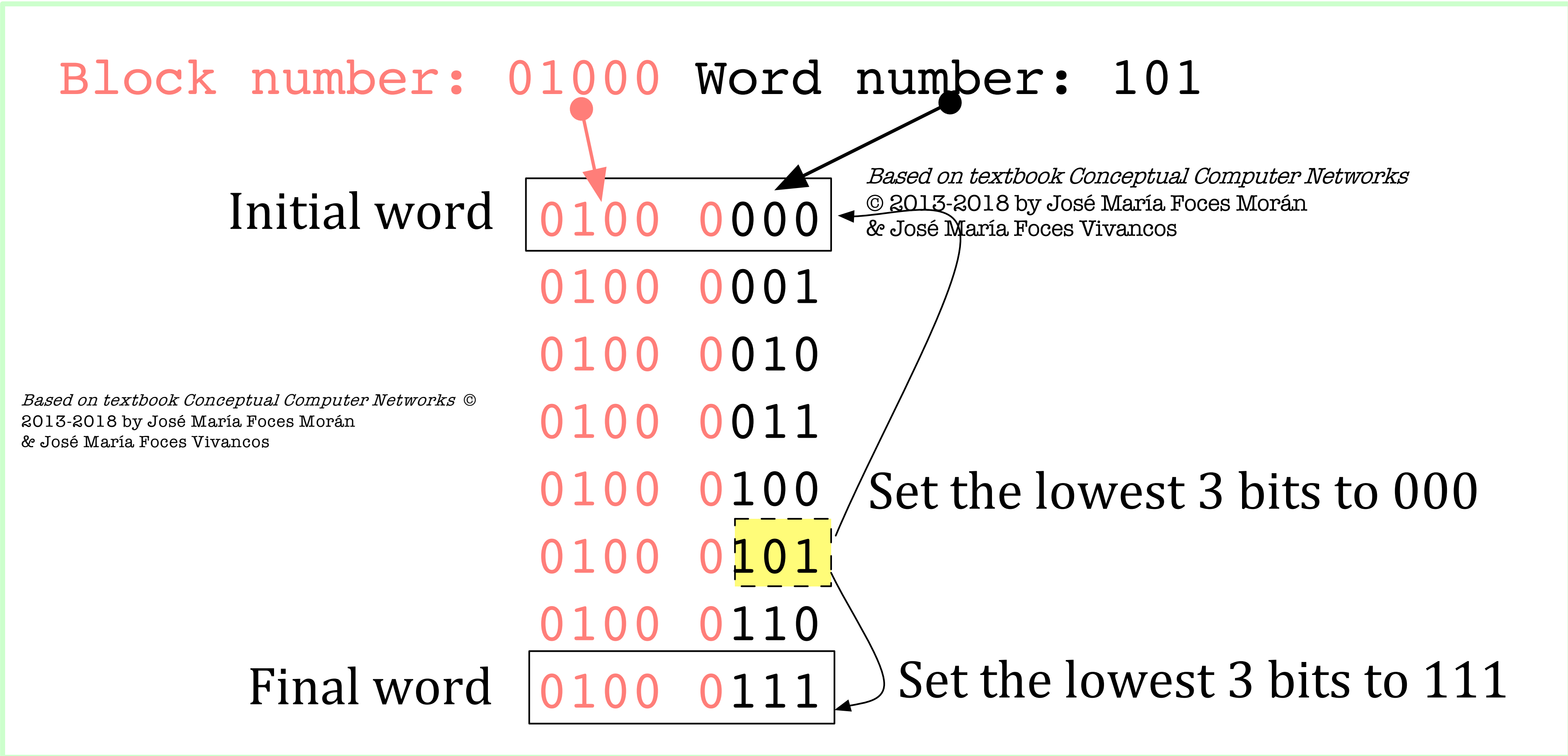
4 Broadcast address = Last Address = Network Number + (Size - 1)
190.80.96.0 + (4096 - 1) = 190.80.111.255

Block of 2^8 words broken down into 2^5 blocks of 2^3 words each



Given an aligned 2^n-sized block, compute first and last 8-bit words

- An IP block is conceptually the same: an aligned 2^n-sized block of IP addresses (32-bit words)



Computing first and last with single operation


```
First word  0100 0000
            0100 0001
            0100 0010
            0100 0011
            0100 0100
            0100 0101
            0100 0110
Last word   0100 0111
```

1. Block size = 2^3 = 8
   Integer power of 2
   Ok!

2. First address aligned
   01000000 mod 8 = 0
   Ok!

Given a word, compute the first address in a single operation

- *Set the lowest 3 bits to 0*
- *Leaving the other 5 bits untouched*
- *Which bit-wise logical operation? Bit-wise AND*

M	B	M&B
0	0	0
0	1	0
1	0	0
1	1	1

If M = 0, then result is always 0

If M = 1, then result is = B

M is known as 1-bit MASK

Mask for computing the first word


```
First word  0100 0000
            0100 0001
            0100 0010
            0100 0011
            0100 0100
            0100 0101
            0100 0110
Last word   0100 0111
```

1. Block size = 2^3 = 8
   Integer power of 2
   Ok!

2. First address aligned
   01000000 mod 8 = 0
   Ok!

\- -

Given word 0100 0101**, compute the first address in a single operation**

- *Set the lowest 3 bits to 0*
  MASK low bits = 000

- *Leaving the other 5 bits untouched*
  MASK high bits = 11111


```
  WORD  = 0100 0101
& MASK  = 1111 1000
- - - - - - - - - -
  first = 0100 0000
```

Mask for computing the last word


```
First word  0100 0000
            0100 0001
            0100 0010
            0100 0011
            0100 0100
            0100 0101
            0100 0110
Last word   0100 0111
```

1. Block size = 2^3 = 8
   Integer power of 2
   Ok!

2. First address aligned
   01000000 mod 8 = 0
   Ok!

- -

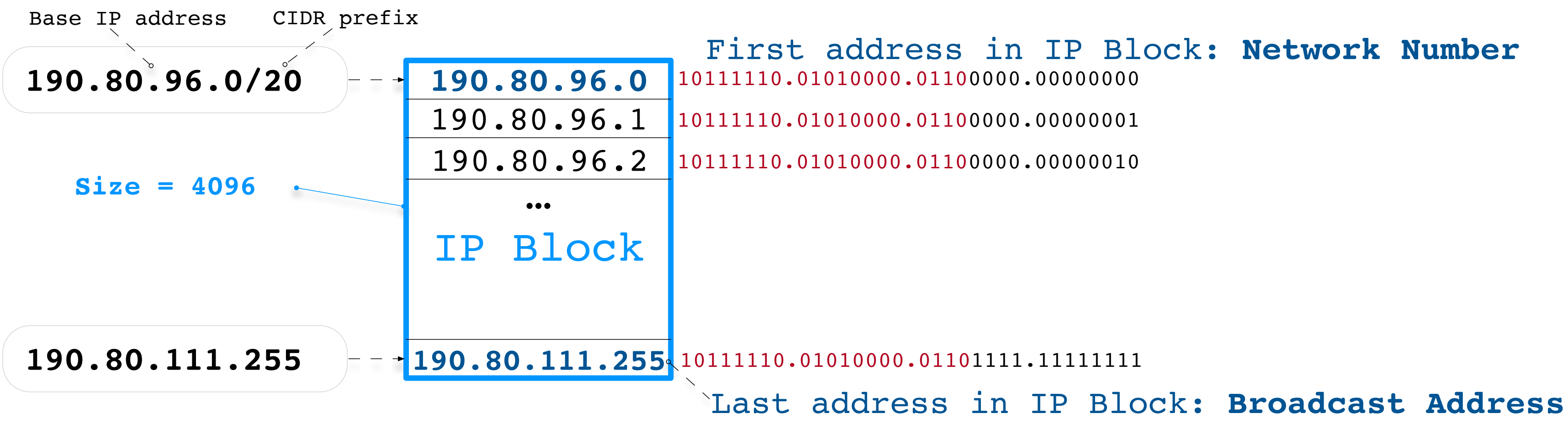
Given word 0100 0101**, compute the last address in a single operation**

- *Set the lowest 3 bits to 1*
  MASK low bits = 111

- *Leaving the other 5 bits untouched*
  MASK high bits = 00000


```
  WORD  = 0100 0101
| MASK  = 0000 0111
- - - - - - - - - -
  first = 0100 0111
```

Same for IP Blocks



1 Given *IP address 190.80.96.2 calculate the Network Prefix*

CIDR prefix is 20, therefore the 20 Most Significant bits represent the IP block number

- *MASK 20 high bits = 1111 1111 1111 1111 1111*
- *MASK (30-20) low bits = 0000 0000 0000*
- *MASK 32 bits = 1111 1111 1111 1111 1111 0000 0000 0000*
- MASK in DDN = 255 .255 .240 .0

```
  Given IP address = 190.080.096.002
& MASK             = 255.255.240.000
- - - - - - - - - - - - - - - - - - -
  First            = 190.080.096.000 The network number or Base address
```


2 Given *IP address 190.80.96.2 calculate the Broadcast Address*

CIDR prefix is 20, therefore the 20 Most Significant bits represent the IP block number

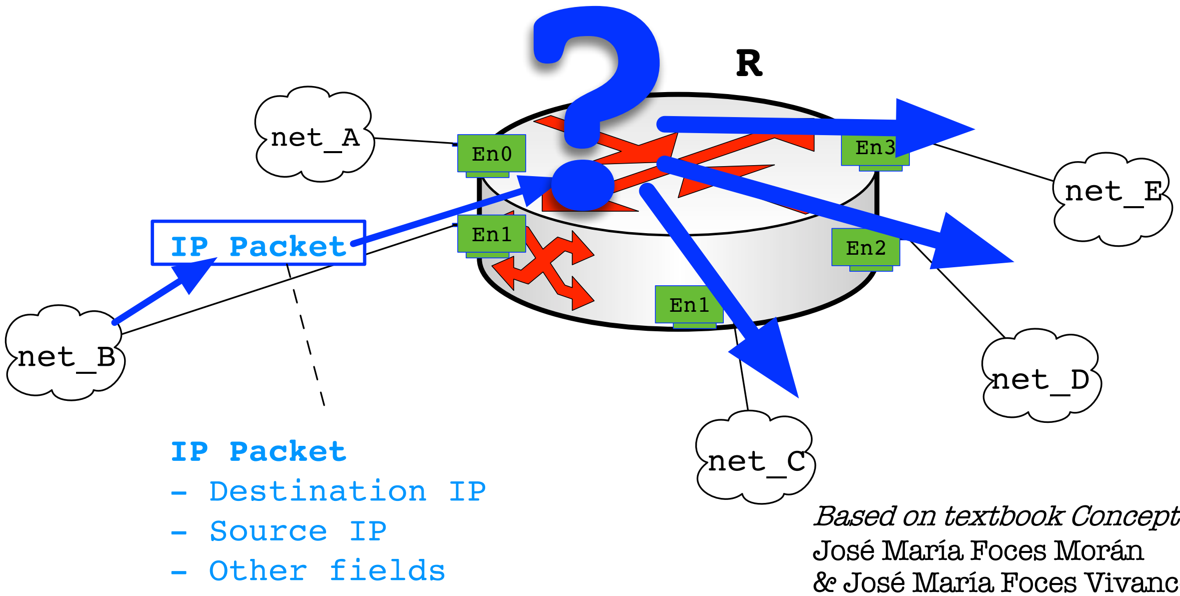
- *MASK 20 high bits = 0000 0000 0000 0000 0000*
- *MASK (30-20) low bits = 1111 1111 1111*
- *MASK 32 bits = 0000 0000 0000 0000 0000 1111 1111 1111*
- MASK in DDN = 000 .000 .000 .255

```
  Given IP address = 190.080.096.002
| NOT MASK         = 000.000.000.255
- - - - - - - - - - - - - - - - - - -
  First            = 190.080.096.255 The broadcast address
```

Does this IP belong to this Prefix?

- This is the core about the IP Forwarding Algorithm

Does Destination IP **belong** to some of my known Network Prefixes?

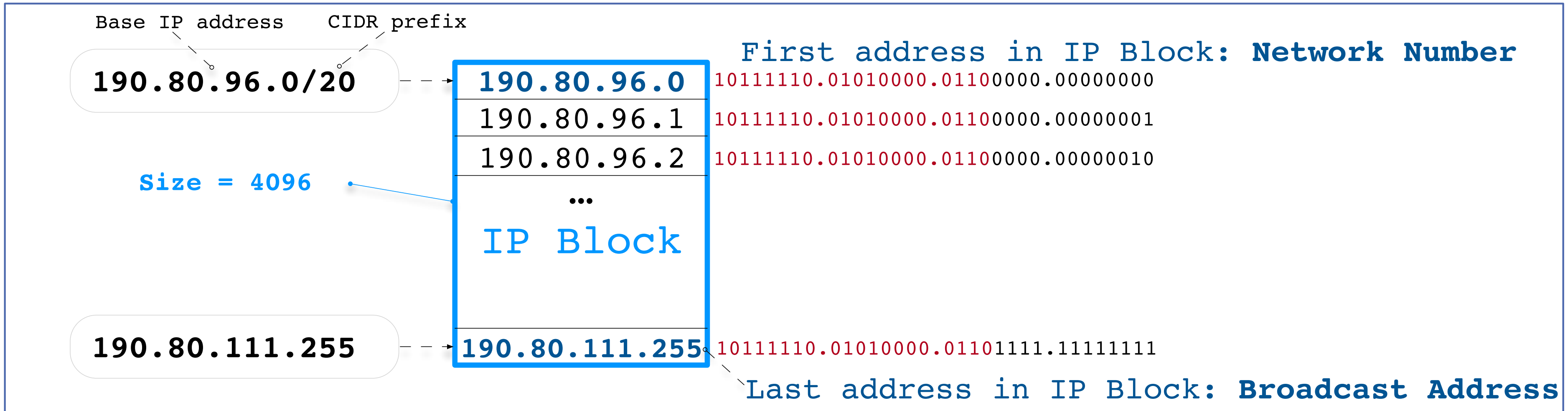


R forwarding table

Destination Network Prefix	Next hop	Interface
net_A	Direct	En0
net_B	Direct	En1
net_C	Direct	En2
net_D	Direct	En3
net_E	Direct	En4

Does this IP belong to this Prefix?

- This is not a match since the IP does not belong to IP block



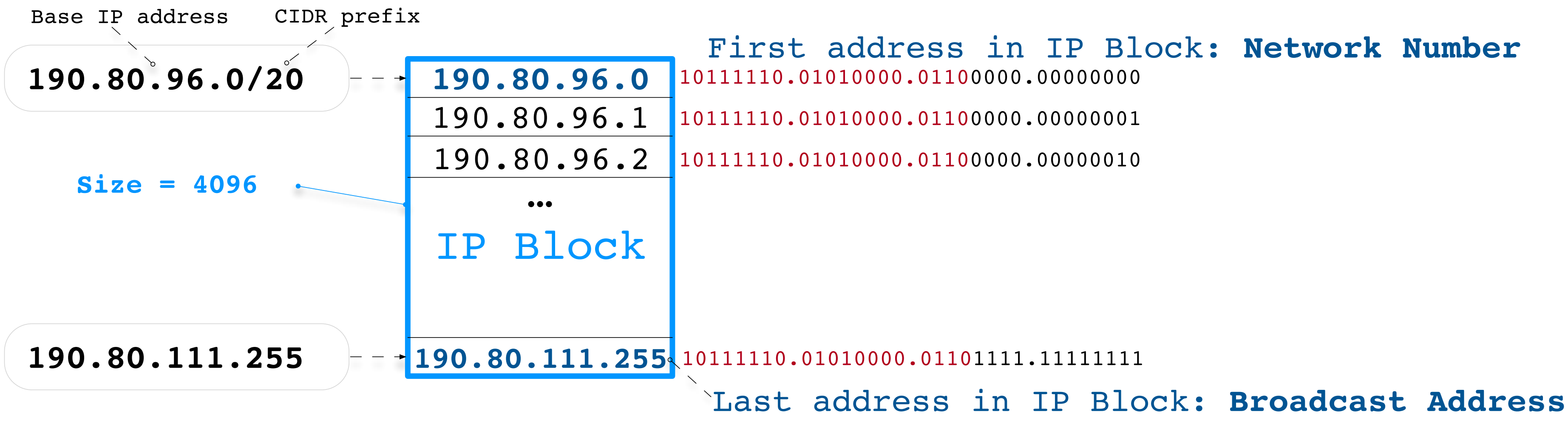
1 Does IP address belong to this block?

```
  Given IP address = 190.080.095.002
& MASK             = 255.255.240.000
- - - - - - - - - - - - - - - - - - -
  Result           = 190.080.095.000

  Prefix number    = 190.080.096.000
```


2 Since Prefix Number ≠ Result -> IP address does not match the Prefix

Does this IP belong to this Prefix?

- If so, this is a **match** (of length /20)



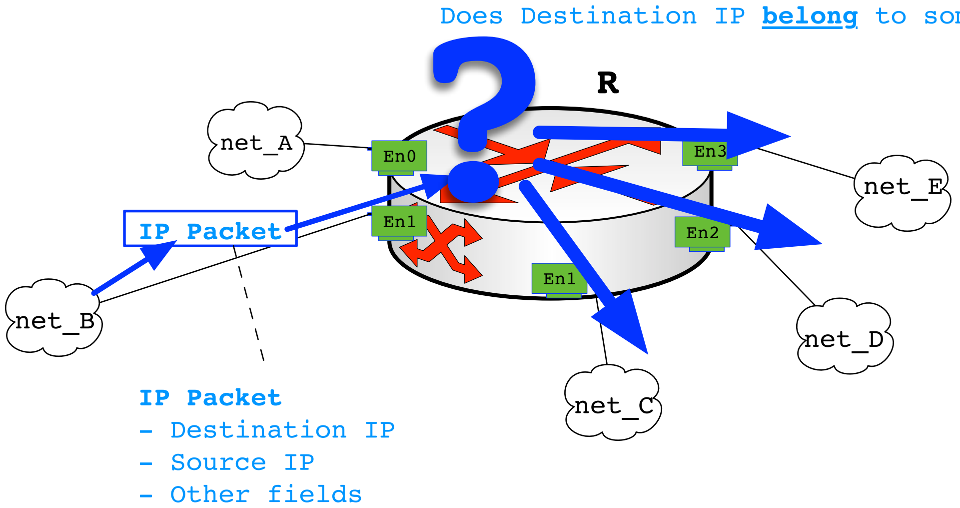
1 Does IP address belong to this block?

```
  Given IP address = 190.080.096.002
& MASK             = 255.255.240.000
- - - - - - - - - - - - - - - - - - -
  Result           = 190.080.096.000

  Prefix number    = 190.080.096.000
```


2 Since Prefix Number = Result -> IP address does **match** the Prefix

CONCEPT: An IP address belongs to *multiple* IP blocks

- If an IP matches **various Prefixes**, which one is the chosen one?
  - The Longest. The longest will tell us the next hop!!!
  - Longest Prefix Matching is the name of the IP forwarding algorithm



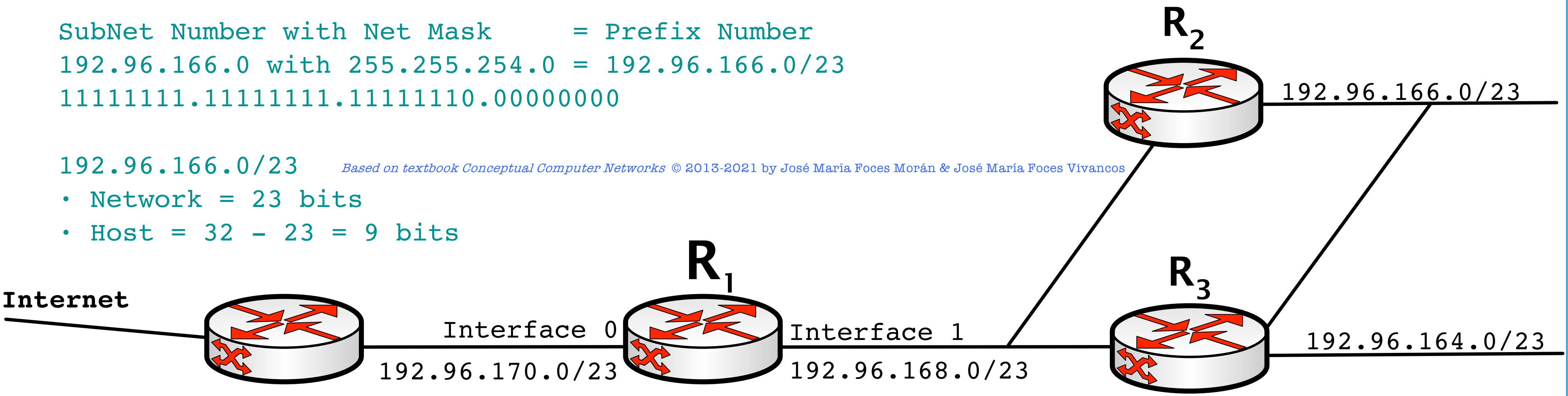
R forwarding table

Destination Network Prefix	Next hop	Interface
net_A	Direct	En0
net_B	Direct	En1
net_C	Direct	En2
net_D	Direct	En3
net_D	Direct	En4

Example: Mask from CIDR prefix


```
SubNet Number with Net Mask      = Prefix Number
192.96.166.0 with 255.255.254.0 = 192.96.166.0/23
11111111.11111111.11111110.00000000

192.96.166.0/23
• Network = 23 bits
• Host = 32 - 23 = 9 bits
```

Based on textbook Conceptual Computer Networks © 2013-2021 by José María Foces Morán & José María Foces Vivancos



Based on textbook Conceptual Computer Networks © 2013-2021 by José María Foces Morán & José María Foces Vivancos

Exercise from Ed. 5 of P&D (Solved)

Check other exercises at
paloalto.unileon.es/cn
- Exams
- Notes, etc

http://paloalto.unileon.es/cn/notes/CN-NotesOnVLSMandCIDR.pdf

✓ **56.** Suppose a router has built up the routing table shown in Table 3.19. The router can deliver packets directly over interfaces 0 and 1, or it can forward packets to routers R2, R3, or R4. Assume the router does the longest prefix match. Describe what the router does with a packet addressed to each of the following destinations:

(a) 128.96.171.92
(b) 128.96.167.151
(c) 128.96.163.151
(d) 128.96.169.192
(e) 128.96.165.121

Table 3.19 Routing Table for Exercise 56

SubnetNumber	SubnetMask	NextHop
128.96.170.0	255.255.254.0	Interface 0
128.96.168.0	255.255.254.0	Interface 1
128.96.166.0	255.255.254.0	R2
128.96.164.0	255.255.252.0	R3
⟨default⟩		R4

Subnet Masks:

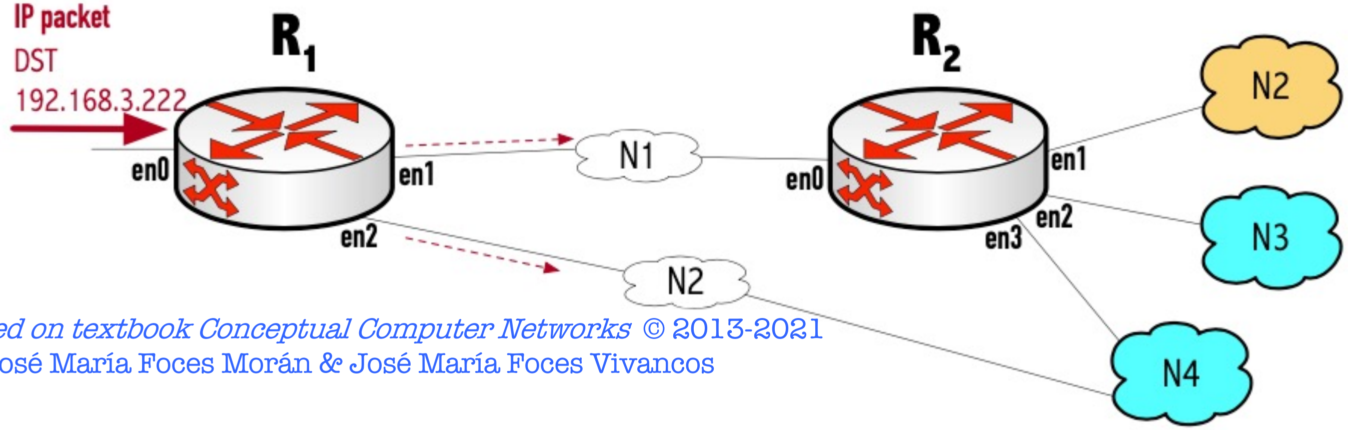
CIDR /23 = 255.255.254.0

CIDR /22 = 255.255.252.0

Exercise about LPM/VLSM/CIDR

Router R1

Network Prefix Number	Next-Hop	Interface
192.168.4.0/24	Direct	en1
192.168.3.0/24	Direct	en2
192.168.8.0/24	192.168.4.2	en1
192.168.2.0/24	192.168.4.2	en1
192.168.3.0/24	Direct	en2
192.168.2.0/23	192.168.3.2	en1
192.168.8.0/24	192.168.3.2	en2
192.168.2.0/24	192.168.3.2	en2
192.168.2.0/23	192.168.3.2	en2



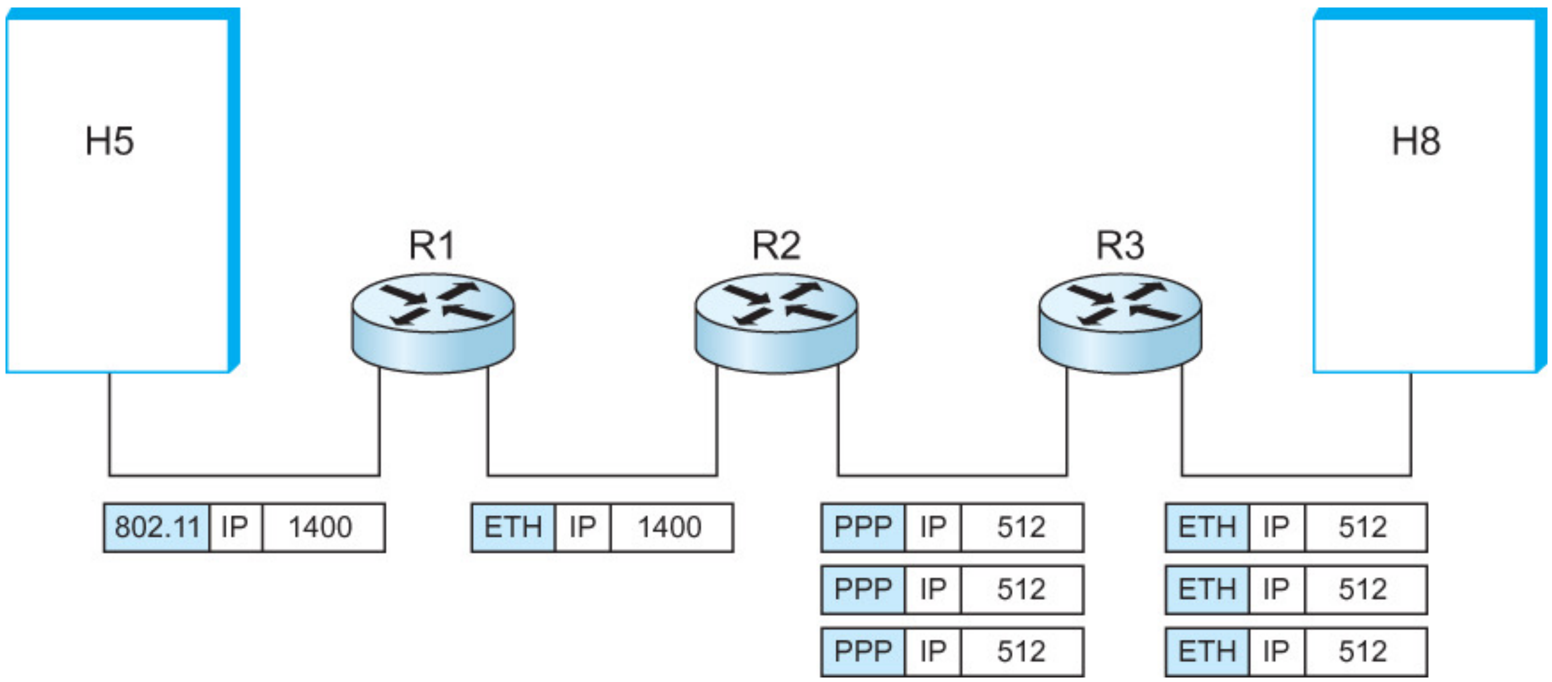
LPM for forwarding IP packet: Which of the prefixes matching 192.168.3.222 is the best? Otherwise, which is the Longest Prefix that matches this IP?

IP Fragmentation and Reassembly

- Each network has some MTU (Maximum Transmission Unit)
  - Ethernet (1500 bytes), FDDI (4500 bytes)

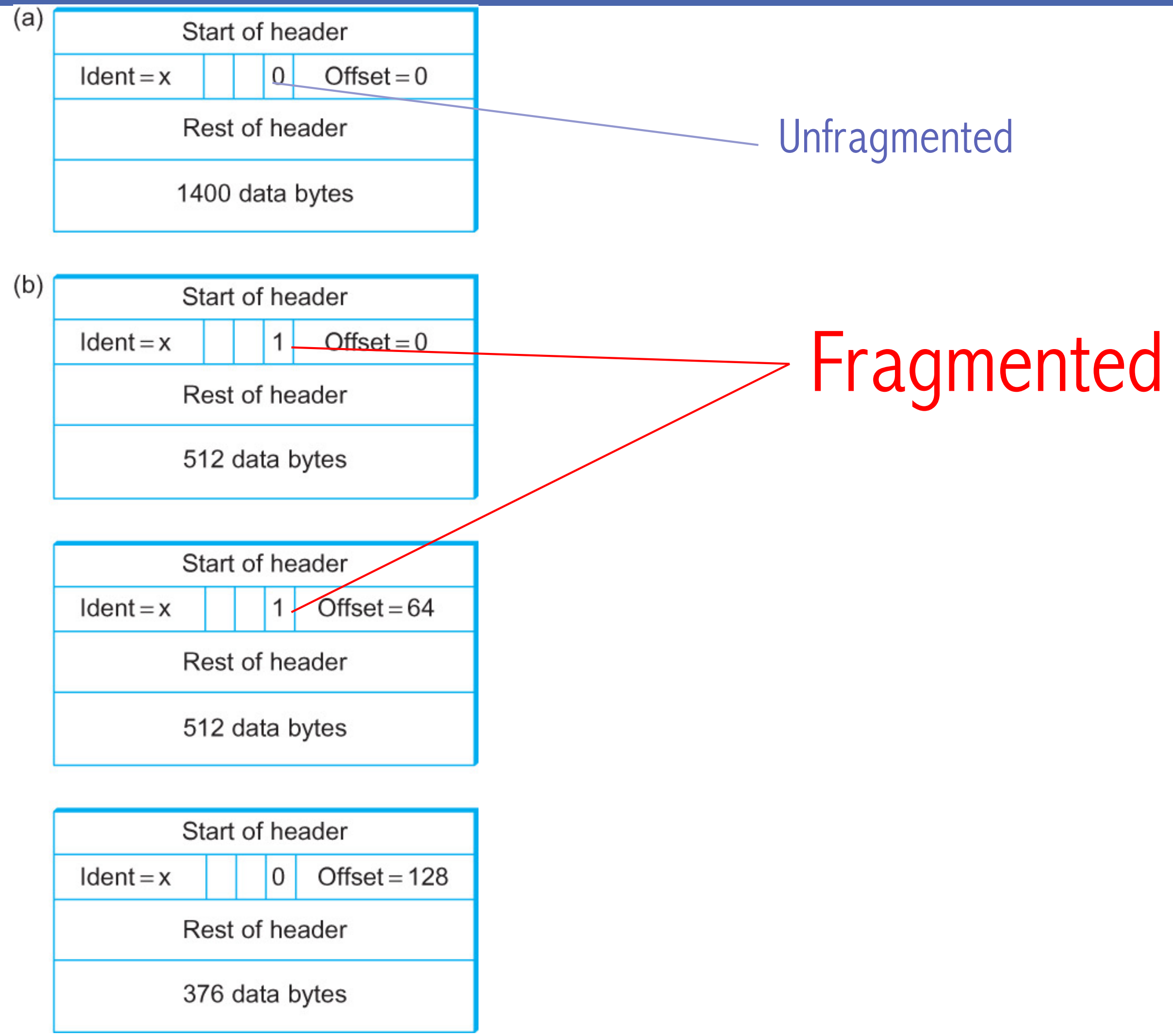
- Strategy
  - Fragmentation occurs in a **router** when it receives a datagram that it wants to forward over a network which has (MTU < datagram)
  - Reassembly is done at the receiving host
  - All the fragments carry the same identifier in the *Ident* field
  - Fragments are self-contained datagrams
  - IP does not recover from missing fragments

IP Fragmentation and Reassembly



IP datagrams traversing the sequence of physical networks

IP Fragmentation and Reassembly



Header fields used in IP fragmentation. (a) Unfragmented packet; (b) fragmented packets.

Happy birthday

- **Ethernet became 40 *a few years ago!***