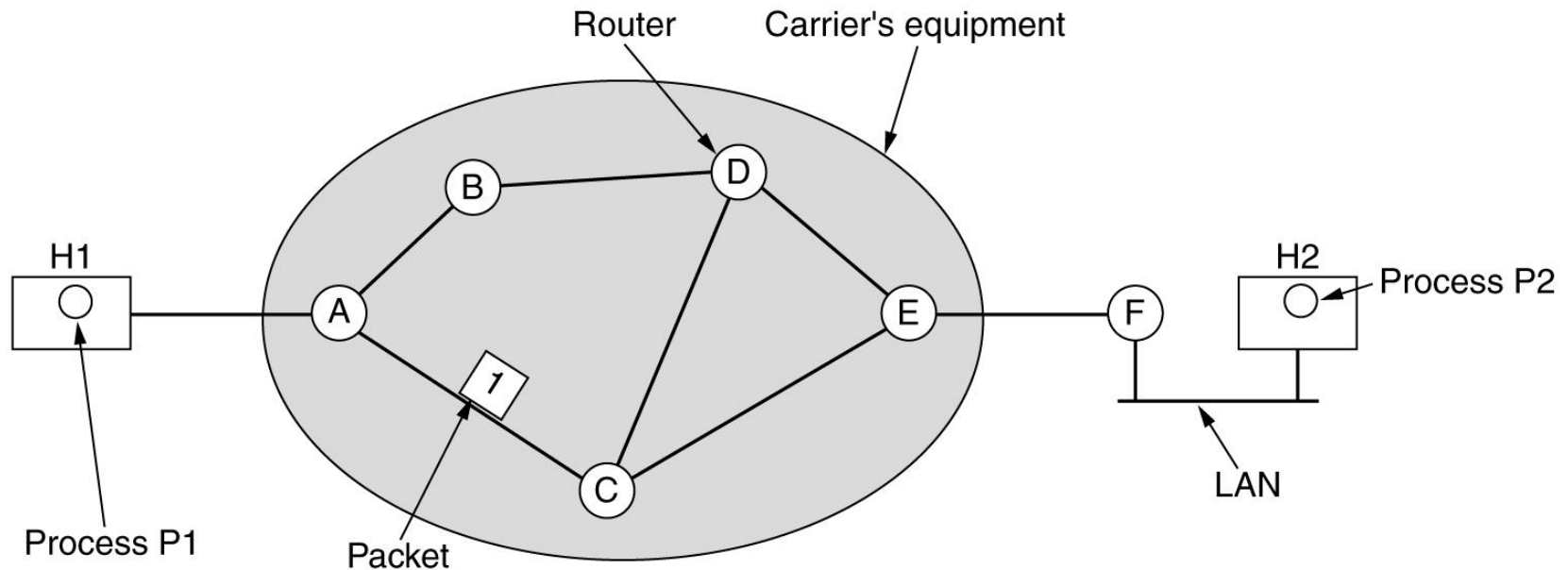


# Network Layer Design Issues

- Store-and-Forward Packet Switching
- Services Provided to the Transport Layer
- Implementation of Connectionless Service
- Implementation of Connection-Oriented Service
- Comparison of Virtual-Circuit and Datagram Subnets

# Store-and-Forward Packet Switching

The environment of the network layer protocols.



# Services provided to Transport layer

The services need to be carefully designed with the following goals in mind:

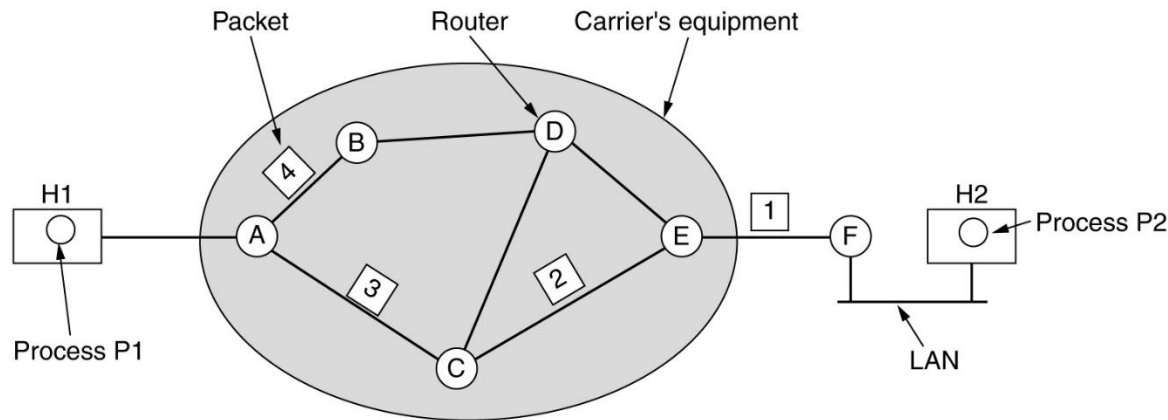
1. The services should be independent of the router technology.
2. The transport layer should be shielded from the number, type, and topology of the routers present.
3. The network addresses made available to the transport layer should use a uniform numbering plan, even across LANs and WANs.

# Packet Switching

- Datagram Approach: Connectionless Service
- Virtual Circuit Approach: Connection Oriented Service

# Implementation of Connectionless Service

Routing within a diagram subnet.



A's table

initially	later
A -	A -
B B	B B
C C	C C
D B	D B
E C	E B
F C	F B

Dest. Line

C's table

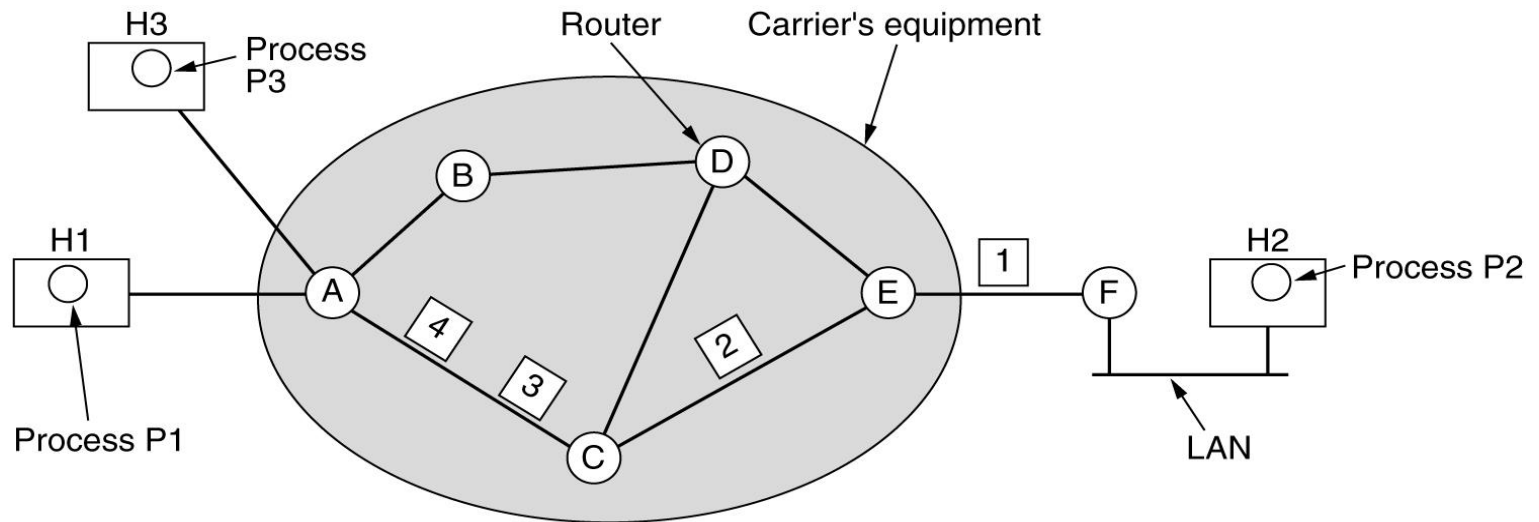
A A
B A
C -
D D
E E
F E

E's table

A C
B D
C C
D D
E -
F F

# Implementation of Connection-Oriented Service

Routing within a virtual-circuit subnet.



A's table				C's table				E's table			
H1	1	C	1	A	1	E	1	C	1	F	1
H3	1	C	2	A	2	E	2	C	2	F	2
In		Out									

# Comparison of Virtual-Circuit and Datagram Subnets

Issue	Datagram subnet	Virtual-circuit subnet
Circuit setup	Not needed	Required
Addressing	Each packet contains the full source and destination address	Each packet contains a short VC number
State information	Routers do not hold state information about connections	Each VC requires router table space per connection
Routing	Each packet is routed independently	Route chosen when VC is set up; all packets follow it
Effect of router failures	None, except for packets lost during the crash	All VCs that passed through the failed router are terminated
Quality of service	Difficult	Easy if enough resources can be allocated in advance for each VC
Congestion control	Difficult	Easy if enough resources can be allocated in advance for each VC

# Network Layer Services

- Packetizing: encapsulating the payload(data received from upper layer) at source and decapsulating at the destination.
- Routing: To find the best path from source to destination using routing protocols.
- Forwarding: Action applied by each router when packet arrives at one of its interface using routing or forwarding table.
- Routing and Forwarding are related to each other.



# NETWORK-LAYER PERFORMANCE

The performance of a network can be measured in terms of:

- *Delay*
- *Throughput*
- *Packet loss*

# Delay

The delays in a network can be divided into four types:

- Transmission delay
- Propagation delay
- Processing delay
- Queuing delay.

- *Transmission Delay*

$$\text{Delay}_{tr} = (\text{Packet length}) / (\text{Transmission rate}).$$

- *Propagation Delay*

$$\text{Delay}_{pg} = (\text{Distance}) / (\text{Propagation speed}).$$

- *Processing Delay*

$\text{Delay}_{pr}$  = Time required to process a packet in a router or a destination host

- *Queuing Delay*

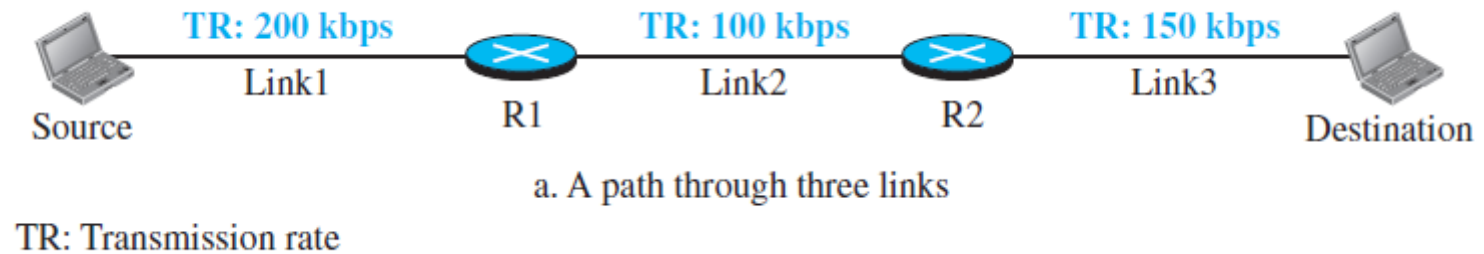
$\text{Delay}_{qu}$  = The time a packet waits in input and output queues in a router

- *Total Delay*

$$\text{Total delay} = (n + 1) (\text{Delay}_{tr} + \text{Delay}_{pg} + \text{Delay}_{pr}) + (n) (\text{Delay}_{qu})$$

# Throughput

- Throughput = minimum {TR1, TR2, . . . TRn}.



# Packet Loss

- When a router receives a packet while processing another packet, the received packet needs to be stored in the input buffer waiting for its turn.
- A router, however, has an input buffer with a limited size. A time may come when the buffer is full and the next packet needs to be dropped.
- The effect of packet loss on the Internet network layer is that the packet needs to be resent, which in turn may create overflow and cause more packet loss.



# IPv4 Address

## *Note*

An IPv4 address is 32 bits long.

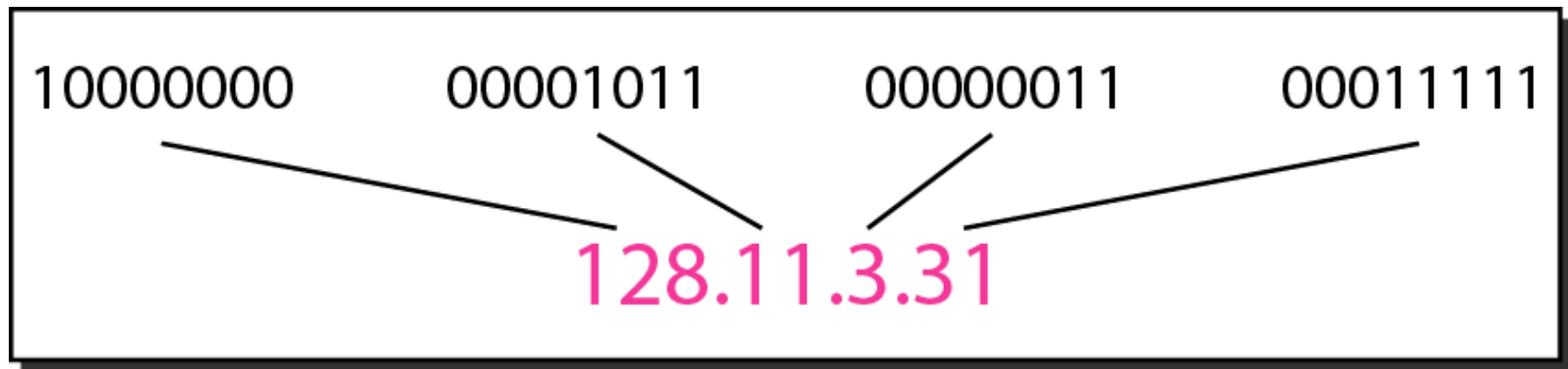
## *Note*

The IPv4 addresses are unique and universal.



The address space of IPv4 is  
 $2^{32}$  or 4,294,967,296.

**Figure** Dotted-decimal notation and binary notation for an IPv4 address



Change the following IPv4 addresses from binary notation to *dotted-decimal notation*.

a. 10000001 00001011 00001011 11101111

b. 11000001 10000011 00011011 11111111

### *Solution*

*We replace each group of 8 bits with its equivalent decimal number (see Appendix B) and add dots for separation.*

a. 129.11.11.239

b. 193.131.27.255

Change the following IPv4 addresses from dotted-decimal  
*notation to binary notation.*

a. 111.56.45.78

b. 221.34.7.82

### *Solution*

*We replace each decimal number with its binary equivalent  
(see Appendix B).*

a. 01101111 00111000 00101101 01001110

b. 11011101 00100010 00000111 01010010

Find the error, if any, in the following IPv4 addresses.

- a. 111.56.045.78
- b. 221.34.7.8.20
- c. 75.45.301.14
- d. 11100010.23.14.67

## *Note*

In classful addressing, the address space is divided into five classes: A, B, C, D, and E.

**Figure** Finding the classes in binary and dotted-decimal notation

	First byte	Second byte	Third byte	Fourth byte
Class A	0			
Class B	10			
Class C	110			
Class D	1110			
Class E	1111			

a. Binary notation

	First byte	Second byte	Third byte	Fourth byte
Class A	0–127			
Class B	128–191			
Class C	192–223			
Class D	224–239			
Class E	240–255			

b. Dotted-decimal notation

Find the class of each address.

*a.* 00000001 00001011 00001011 11101111

*b.* 11000001 10000011 00011011 11111111

*c.* 14.23.120.8

*d.* 252.5.15.111



**Table**      Number of blocks and block size in classful IPv4 addressing

<i>Class</i>	<i>Number of Blocks</i>	<i>Block Size</i>	<i>Application</i>
A	128	16,777,216	Unicast
B	16,384	65,536	Unicast
C	2,097,152	256	Unicast
D	1	268,435,456	Multicast
E	1	268,435,456	Reserved

## *Note*

In classful addressing, a large part of the available addresses were wasted.

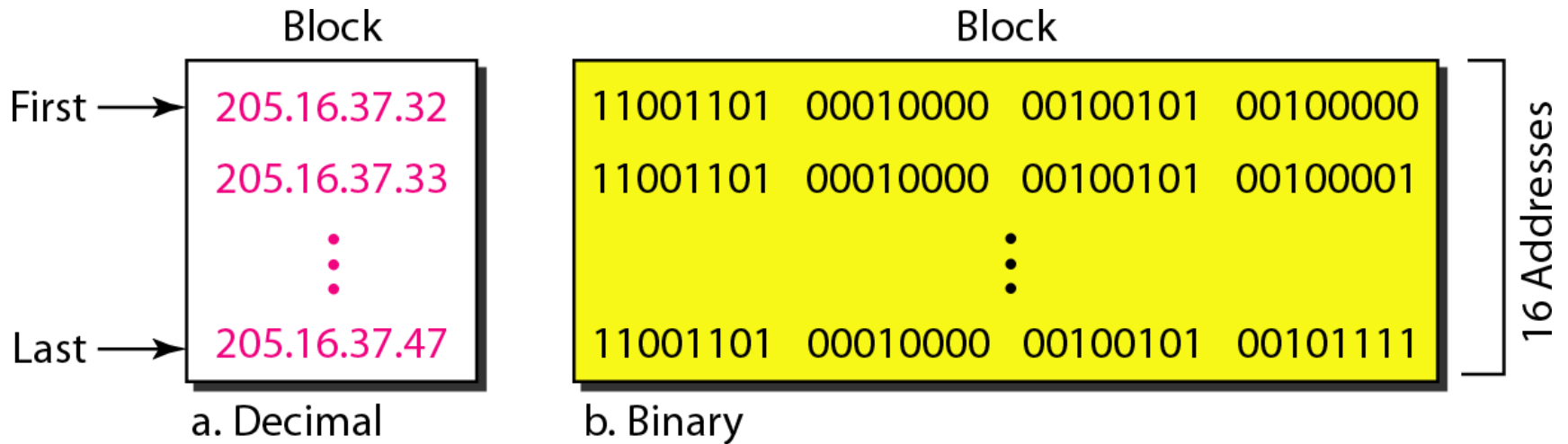
**Table**      Default masks for classful addressing

<i>Class</i>	<i>Binary</i>	<i>Dotted-Decimal</i>	<i>CIDR</i>
A	<b>11111111</b> 00000000 00000000 00000000	<b>255.0.0.0</b>	/8
B	<b>11111111 11111111</b> 00000000 00000000	<b>255.255.0.0</b>	/16
C	<b>11111111 11111111 11111111</b> 00000000	<b>255.255.255.0</b>	/24

## *Note*

Classful addressing, which is almost obsolete, is replaced with classless addressing.

**Figure** A block of 16 addresses granted to a small organization



In IPv4 addressing, a block of addresses can be defined as

$x.y.z.t / n$

in which  $x.y.z.t$  defines one of the addresses and the  $/n$  defines the mask.

## *Note*

The first address in the block can be found by setting the rightmost  $32 - n$  bits to 0s.

*A block of addresses is granted to a small organization. We know that one of the addresses is 205.16.37.39/28. What is the first address in the block?*

### *Solution*

*The binary representation of the given address is 11001101*

*00010000 00100101 00100111*

*If we set 32–28 rightmost bits to 0, we get*

*11001101 00010000 00100101 00100000*

*or 205.16.37.32.*



## *Note*

The last address in the block can be found by setting the rightmost  $32 - n$  bits to 1s.

## *Find the last address for the block in Last Example*

### *Solution*

*The binary representation of the given address is*

*11001101    00010000    00100101    00100111*

*If we set 32 – 28 rightmost bits to 1, we get 11001101*

*00010000 00100101 00101111*

*or 205.16.37.47*

## *Note*

The number of addresses in the block can be found by using the formula  $2^{32-n}$ .

*Find the number of addresses in  
last example*

*Solution*

*The value of  $n$  is 28, which mean the  
number of addresses is  $2^{32-28}$  or 16.*

## *Another way to find the first address, the last address, and*

*the number of addresses is to represent the mask as a 32-bit binary (or 8-digit hexadecimal) number. This is particularly useful when we are writing a program to find these pieces of information. In Example the /28 can be represented as*

*11111111 11111111 11111111 11110000*

*(twenty-eight 1s and four 0s).*

*Find*

*a. The first address*

*b. The last address*

*c. The number of addresses.*

## Solution

- a. The first address can be found by ANDing the given addresses with the mask. ANDing here is done bit by bit. The result of ANDing 2 bits is 1 if both bits are 1s; the result is 0 otherwise.*

Address:	11001101	00010000	00100101	00100111
Mask:	11111111	11111111	11111111	11110000
First address:	11001101	00010000	00100101	00100000

- b. The last address can be found by ORing the given addresses with the complement of the mask. ORing here is done bit by bit. The result of ORing 2 bits is 0 if both bits are 0s; the result is 1 otherwise. The complement of a number is found by changing each 1 to 0 and each 0 to 1.*

Address:	11001101	00010000	00100101	00100111
Mask complement:	00000000	00000000	00000000	00001111
Last address:	11001101	00010000	00100101	00101111

*The number of addresses can be found by complementing the mask, interpreting it as a decimal number, and adding 1 to it.*

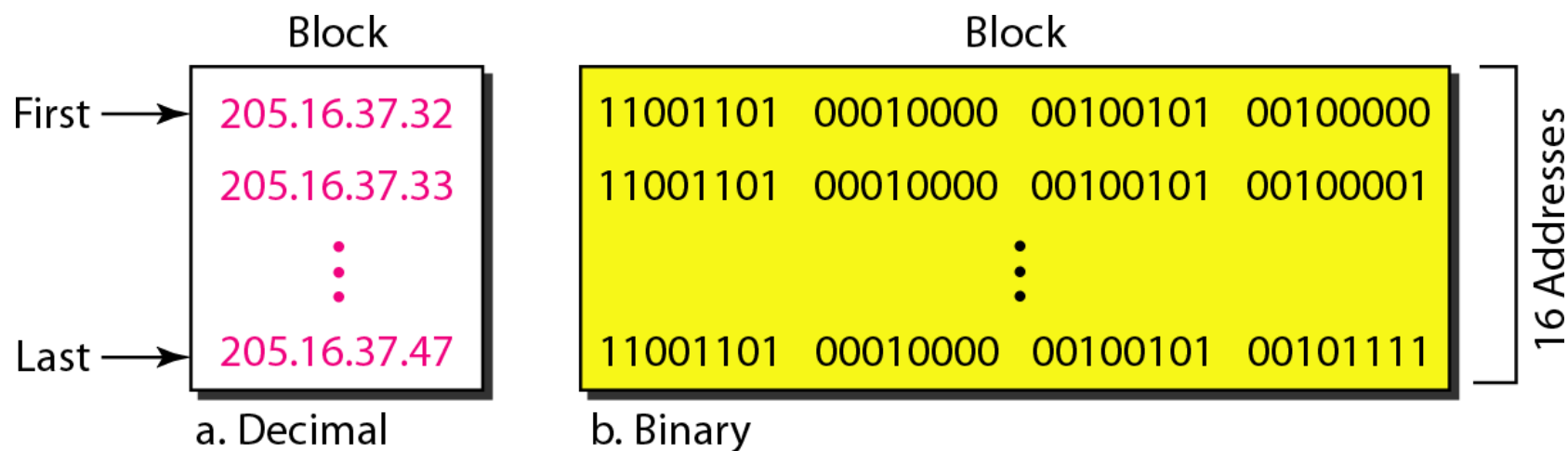
Mask complement:      **00000000 00000000 00000000 00001111**

Number of addresses:     $15 + 1 = 16$



## Figure

*A network configuration for the block  
205.16.37.32/28*



The first address in a block is normally not assigned to any device;  
it is used as the network address that represents the organization to the rest of the world.

Figure

*Two levels of hierarchy in an IPv4 address*

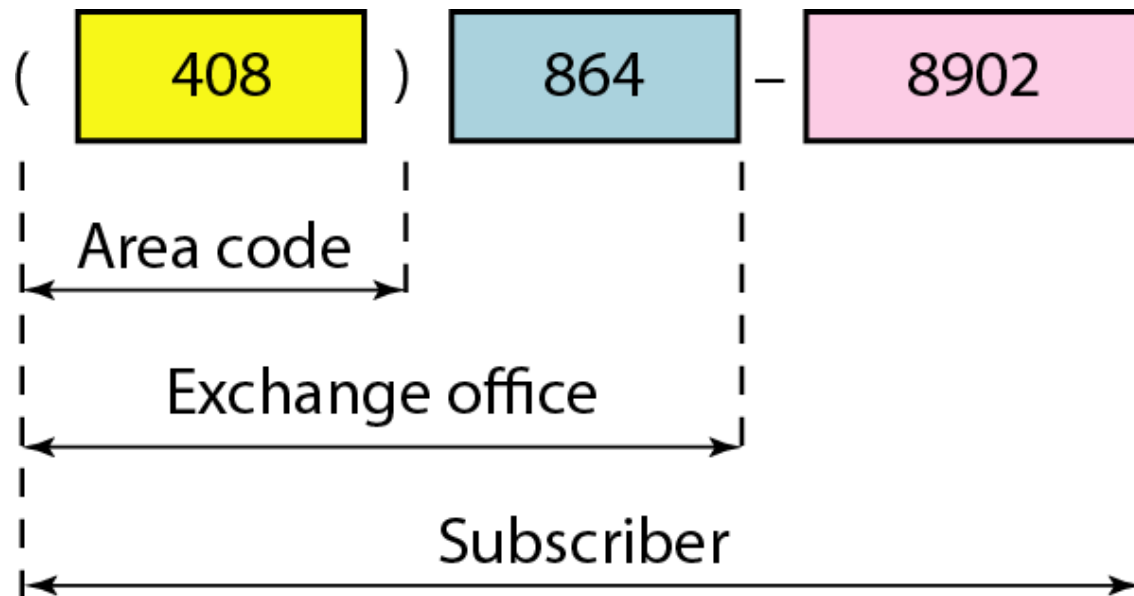
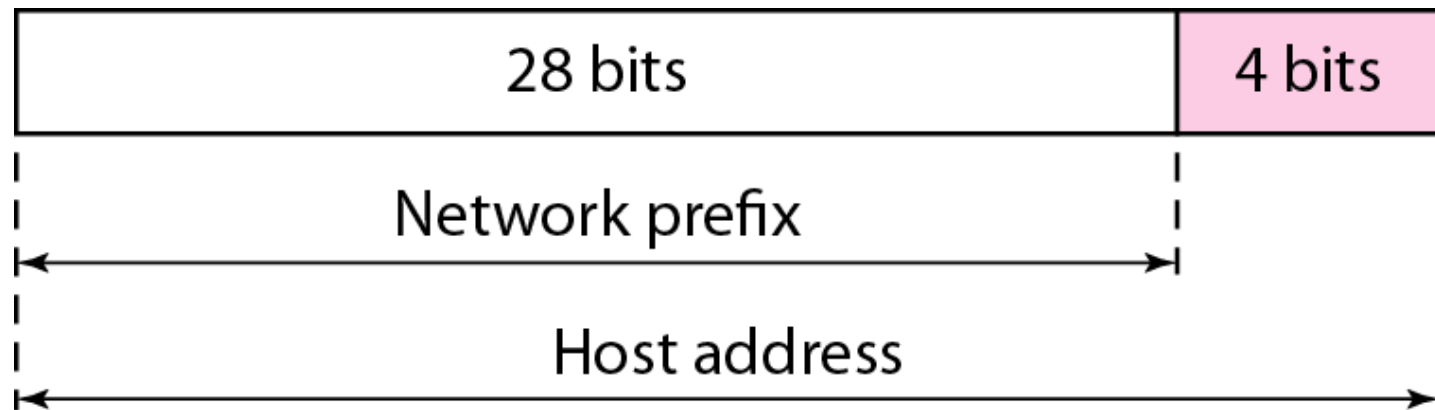
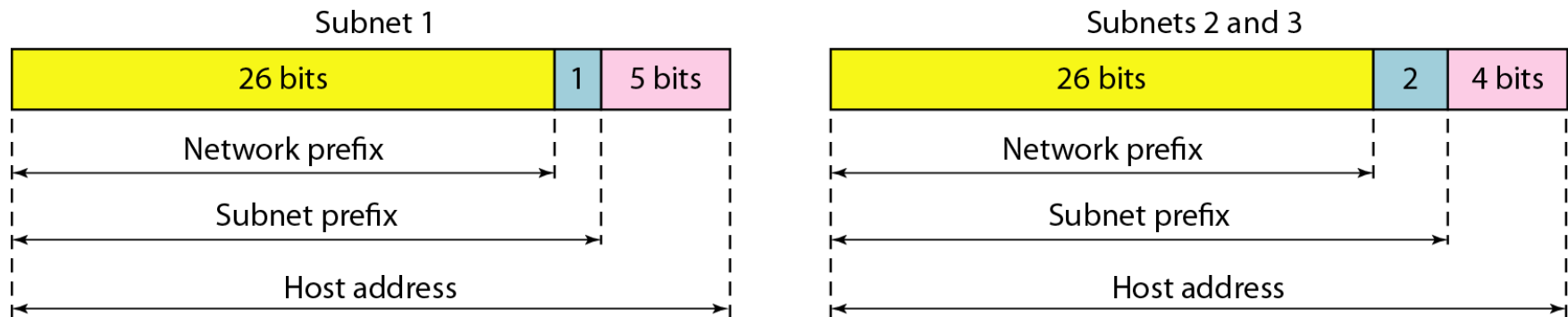


Figure *A frame in a character-oriented protocol*



Each address in the block can be considered as a two-level hierarchical structure: the leftmost  $n$  bits (prefix) define the network; the rightmost  $32 - n$  bits define the host.

**Figure** *Three-level hierarchy in an IPv4 address*



- *An ISP is granted a block of addresses starting with 190.100.0.0/16 (65,536 addresses). The ISP needs to distribute these addresses to three groups of customers as follows:*
  - a. The first group has 64 customers; each needs 256 addresses.*
  - b. The second group has 128 customers; each needs 128 addresses.*
  - c. The third group has 128 customers; each needs 64 addresses.*

*Design the subblocks and find out how many addresses are still available after these allocations.*

## Group 1

*For this group, each customer needs 256 addresses. This means that 8 ( $\log_2 256$ ) bits are needed to define each host. The prefix length is then  $32 - 8 = 24$ . The addresses are*

1st Customer:	190.100.0.0/24	190.100.0.255/24
2nd Customer:	190.100.1.0/24	190.100.1.255/24
...		
64th Customer:	190.100.63.0/24	190.100.63.255/24
Total = $64 \times 256 = 16,384$		



## Group2

For this group, each customer needs 128 addresses. This means that 7 ( $\log_2 128$ ) bits are needed to define each host. The prefix length is then  $32 - 7 = 25$ . The addresses are

1st Customer:	190.100.64.0/25	190.100.64.127/25
2nd Customer:	190.100.64.128/25	190.100.64.255/25
...		
128th Customer:	190.100.127.128/25	190.100.127.255/25
Total =	$128 \times 128 = 16,384$	

*For this group, each customer needs 64 addresses. This means that 6 ( $\log_2 64$ ) bits are needed to each host. The prefix length is then  $32 - 6 = 26$ . The addresses are*

1st Customer:	190.100.128.0/26	190.100.128.63/26
2nd Customer:	190.100.128.64/26	190.100.128.127/26
...		
128th Customer:	190.100.159.192/26	190.100.159.255/26
Total =	$128 \times 64 = 8192$	

*Number of granted addresses to the ISP: 65,536 Number of allocated addresses by the ISP: 40,960 Number of available addresses: 24,576*

## Figure

*An example of address allocation and distribution by an ISP*

