IP addressing made easy

Md. Akhtaruzzaman Adnan

IP Address

- An IP (Internet Protocol) address is a unique address that different computers or devices on a computer network use to identify and communicate with one another.
- An IP address is used as an identifier to find electronic devices connected to one another on a network.
- Therefore, each device in the network must have its own unique address.
- An IP address is like a mailing address that is used to deliver data (files) to a computer.

Cont..

- Some IP addresses are meant to be unique within the scope of the Internet, whereas others are meant to be unique within the scope of a specific network.
- The Internet Assigned Numbers Authority (IANA) creates and manages IP addresses for the public Internet.
- IANA allocates the superblocks of addresses to Regional Internet Registries, which in turn allocate smaller blocks of addresses to Internet service providers

Brief History

- In May 1974, the Institute of Electrical and Electronics Engineers (IEEE) published a paper entitled "A Protocol for Packet Network Intercommunication".
- The paper's authors, Vint Cerf and Bob Kahn, described an internetworking protocol for sharing resources using packet switching among network nodes.
- A central control component of this model was the "Transmission Control Program" that incorporated both connection-oriented links and datagram services between hosts. The model became known as the Department of Defense (DoD) Internet Model and Internet protocol suite, and informally as TCP/IP.

- IP versions 1 to 3 were experimental versions, designed between 1973 and 1978.
- The dominant internetworking protocol in the Internet Layer in use is IPv4; the number 4 identifies the protocol version, carried in every IP datagram. IPv4 is described in RFC 791 (1981).
- Version number 5 was used by the Internet Stream Protocol, an experimental streaming protocol that was not adopted.

- The successor to IPv4 is IPv6. IPv6 was a result of several years of experimentation and dialog during which various protocol models were proposed, such as TP/IX (RFC 1475), PIP (RFC 1621) and TUBA (TCP and UDP with Bigger Addresses, RFC 1347).
- Its most prominent difference from version 4 is the size of the addresses.
- While IPv4 uses 32 bits for addressing, yielding 2^{32} = approx. 4.3 billion (4.3×10^{9}) addresses, IPv6 uses 128-bit addresses providing $2^{128} = 3.4 \times10^{38}$ addresses.

- In this article, we review some of the basics of IPv4 address layout, and then consider a technique to make working with IPv4 addresses easier.
- Although this is not the “conventional” method you might have been taught to work with in IP address space, you will find it is very easy and fast.
- We conclude with a discussion of applying those techniques to the IPv6 address space.

- The success of TCP/IP as the network protocol of the Internet is largely because of its ability to connect together networks of different sizes and systems of different types.
- These networks are arbitrarily defined into three main classes (along with a few others) that have predefined sizes, each of which can be divided into smaller sub networks by system administrators.
- A subnet mask is used to divide an IP address into two parts. One part identifies the host (computer), the other part identifies the network to which it belongs.
- To better understand how IP addresses and subnet masks work, look at an IP (Internet Protocol) address and see how it is organized.

Basic IPv4 Addressing

- IPv4 addresses are essentially ***32-bit binary*** numbers; computer systems and routers do not see any sorts of divisions within the IPv4 address space.

- To make IPv4 addresses more human-readable, however, we ***break them up into four sections divided by dots, commonly called "octets."***

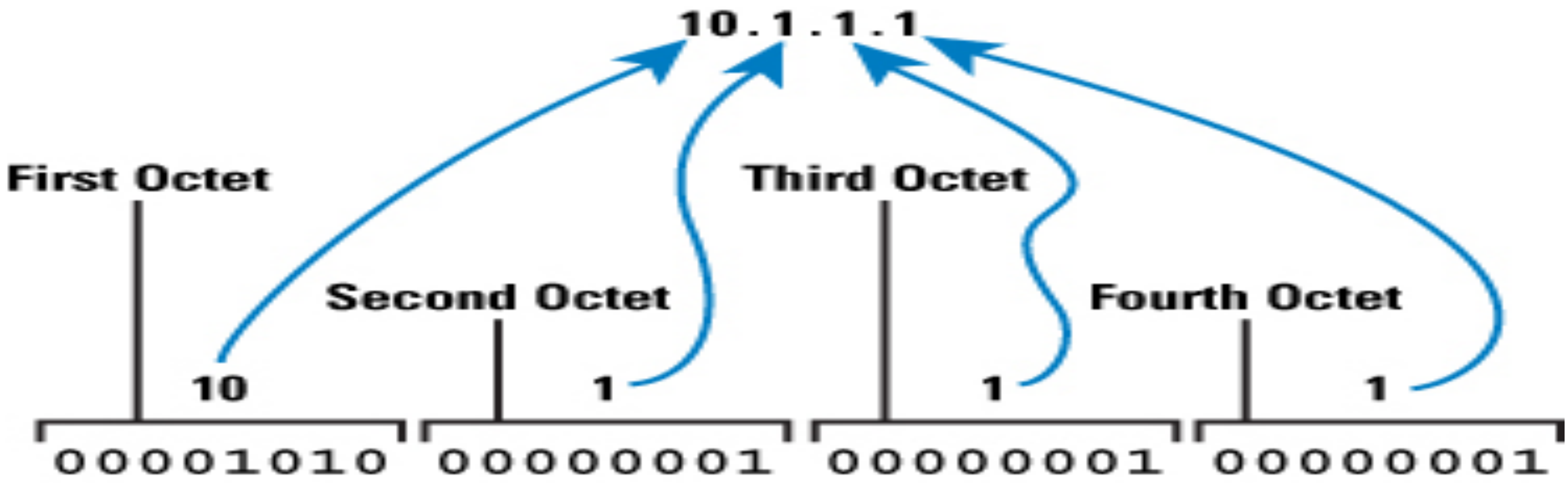
- ***An octet is a set of eight binary digits, sometimes also called a "byte."***

Address Space

A protocol like IPv4 that defines addresses has an **address space.**
An address space is the total number of addresses used by the protocol. If a protocol uses *b* bits to define an address, the address space is 2 to the power *b* because each bit can have two different values (0 or 1).

IPv4 uses 32-bit addresses, which means that the address space is 2
To the power 32 or 4,294,967,296 (more than four billion). Theoretically, if there were no restrictions, more than 4 billion devices could be connected to the Internet.

Figure 1 illustrates the IPv4 address structure



Lowest=00000000 00000000 0000000 0000000= 0. 0. 0. 0

Highest=11111111 11111111 11111111 11111111=255.255.255.255

Range 0-255=total numers In 1 octet=256

DOTTED DECIMAL NOTAION

Switch is on

1	1	1	1	1	1	1	1
2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
128^{1}	64^{1}	32^{0}	16^{0}	8^{1}	4^{0}	2^{0}	1^{0}

11111111 (binary octet) = 128 + 64 + 32 + 16 + 8 + 4 + 2 + 1= 255 (decimal format)

200 (decimal) =128+64+0+0+8+0+0+0= 11001000 (binary)

Switch is on/off

0	1	0	0	0	0	0	1
2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
128	64	32	16	8	4	2	1

01000001 (binary octet) = 0+64+0+0+0+0+0+0+1 = 65 (decimal format)

00000000= 0+0+0+0+0+0+0+0+0=0

At a glance

- The lowest value of any octet is =00000000=0
- The highest value of any octet is=11111111=255
- A single octet has a range of 0-255
- Total numbers in one octet = 256
- Total octets in IPv4=4
- Total numbers in IPv4 = 4 * 256=1024

IP addresses: Networks and hosts

- An IP address is a 32-bit number that uniquely identifies a host (computer or other device, such as a printer or router) on a TCP/IP network.
- IP addresses are normally expressed in dotted-decimal format, with four numbers separated by periods, such as 192.168.123.132.
- To understand how subnet masks are used to distinguish between hosts, networks, and subnetworks, examine an IP address in binary notation.

- For example, the dotted-decimal IP address 192.168.123.132 is (in binary notation) the 32 bit number 110000000101000111101110000100.
- This number may be hard to make sense of, so divide it into four parts of eight binary digits.
- These eight bit sections are known as octets. The example IP address, then, becomes 11000000.10101000.01111011.10000100.

Examples

- Change the following IPv4 addresses from binary notation to dotted-decimal notation.
- **a.**
- **10000001 00001011 00001011 11101111**
- **b.**
- **11000001 10000011 00011011 11111111**
- **c.**
- **11100111 11011011 10001011 01101111**
- **d.**
- **11111001 10011011 11111011 00001111**

- **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
- **c.**
- 231.219.139.111
- **d.**
- 249.155.251.15

Example

- 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

Solution

- **a.**
- There should be no leading zeroes in dotted-decimal notation (045).
- **b.**
- We may not have more than 4 bytes in an IPv4 address.
- **c.**
- Each byte should be less than or equal to 255; 301 is outside this range.
- **d.**
- A mixture of binary notation and dotted-decimal notation is not allowed.

Range of Addresses

- We often need to deal with a range of addresses instead of one single address.
- We sometimes need to find the number of addresses in a range if the first and last address is given. Other times, we need to find the last address if the first address and the number of addresses in the range are given. In this case, we can perform subtraction or addition

Example

- Find the number of addresses in a range if the first address is 146.102.29.0 and the last address is 146.102.32.255.
- **Solution**
- We can subtract the first address from the last address in base 256. The result is 0.0.3.255 in this base. To find the number of addresses in the range (in decimal), we convert this number to base 10 and add 1 to the result.
- **Number of addresses =$(0 \times 256^3 + 0 \times 256^2 + 3 \times 256^1 + 255 \times 256^0) + 1 = 1024$**

Example

- The first address in a range of addresses is 14.11.45.96. If the number of addresses in the range is 32, what is the last address?
- **Solution**
- We convert the number of addresses minus 1 to base 256, which is 0.0.0.31. We then add it to the first address to get the last address. Addition is in base 256.
- **Last address**=**$(14.11.45.96+0.0.0.31)_{256}$=14.11.45.127**

Operations

- We often need to apply some operations on 32-bit numbers in binary or dotted-decimal
  notation. There are three operations that are used : NOT, AND, and OR.

Bitwise NOT Operation

- The bitwise NOT operation is a unary operation; it takes one input. When we apply the NOT operation on a number, it is often said that the number is complemented. The
- NOT operation, when applied to a 32-bit number in binary format, inverts each bit.
- Every 0 bit is changed to a 1 bit; every 1 bit is changed to a 0 bit.

- Although we can directly use the NOT operation on a 32-bit number, when the number is represented as a four-byte dotted-decimal notation, we can use a short cut; we can subtract each byte from 255.

Original number:	00010001	01111001	00001110	00100011
Complement:	11101110	10000110	11110001	11011100

We can use the same operation using the dotted-decimal representation and the short cut.

Original number:	17 .	121 .	14 .	35
Complement:	238 .	134 .	241 .	220

Bitwise AND Operation

- The bitwise AND operation is a binary operation; it takes two inputs. The AND operation compares the two corresponding bits in two inputs and selects the smaller bit
 from the two (or select one of them if the bits are equal).

AND

Input 1	Input 2	Output
0	0	0
0	1	0
1	0	0
1	1	1

Operation for each bit

- Although we can directly use the AND operation on the 32-bit binary representation of two numbers.

First number:	00010001	01111001	00001110	00100011
Second number:	11111111	11111111	10001100	00000000
Result	00010001	01111001	00001100	00000000

- When the numbers are represented in dotted-decimal notation, we can use two short cuts.
- **1. When at least one of the numbers is 0 or 255, the AND operation selects the** smaller byte (or one of them if equal).
- **2. When none of the two bytes is either 0 or 255, we can write each byte as the sum** of eight terms, where each term is a power of 2. We then select the smaller term in each pair (or one of them if equal) and add them to get the result.

First number:	17	.	121	.	14	.	35
Second number:	255	.	255	.	140	.	0
Result:	17	.	121	.	12	.	0

We have applied the first short cut on the first, second, and the fourth byte; we have applied the second short cut on the third byte. We have written 14 and 140 as the sum of terms and selected the smaller term in each pair as shown below.

Powers	2^7		2^6		2^5		2^4		2^3		2^2		2^1		2^0
Byte (14)	0	+	0	+	0	+	0	+	8	+	4	+	2	+	0
Byte (140)	128	+	0	+	0	+	0	+	8	+	4	+	0	+	0
Result (12)	0	+	0	+	0	+	0	+	8	+	4	+	0	+	0

Bitwise OR Operation

- The bitwise OR operation is a binary operation; it takes two inputs. The OR operation compares the corresponding bits in the two numbers and selects the larger bit from the two (or one of them if equal). Figure 5.4 shows the OR operation.

OR

Input 1	Input 2	Output
0	0	0
0	1	1
1	0	1
1	1	1

Operation for each bit

- Although we can directly use the OR operation on the 32-bit binary representation of the two numbers.

The following shows how we can apply the OR operation on two 32-bit numbers in binary.

First number:	00010001	01111001	00001110	00100011
Second number:	11111111	11111111	10001100	00000000
Result	11111111	11111111	10001110	00100011

- when the numbers are represented in dotted-decimal notation, we can use two short cuts.
- **1. When at least one of the two bytes is 0 or 255, the OR operation selects the larger**
- byte (or one of them if equal).
- **2. When none of the two bytes is 0 or 255, we can write each byte as the sum of eight** terms, where each term is a power of 2. We then select the larger term in each pair (or one of them if equal) and add them to get the result of OR operation.

First number:	17	.	121	.	14	.	35
Second number:	255	.	255	.	140	.	0
Result:	255	.	255	.	142	.	35

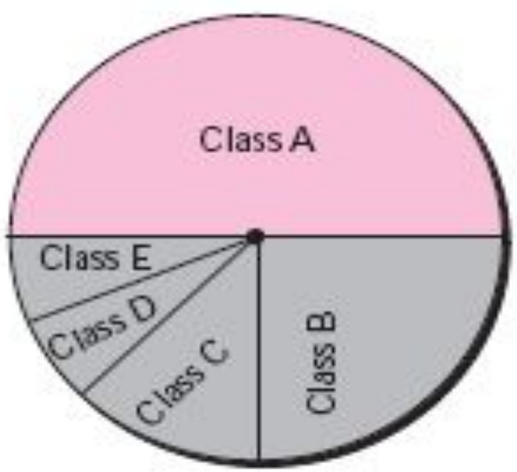
We have used the first short cut for the first and second bytes and the second short cut for the third byte.

CLASSFUL ADDRESSING

- IP addresses, when started a few decades ago, used the concept of *classes. This architecture* is called **classful addressing.**
- **In the mid-1990s, a new architecture, called classless addressing, was introduced that supersedes the original architecture. In this** section, we introduce classful addressing because it paves the way for understanding classless addressing and justifies the rationale for moving to the new architecture.

Classes

- In classful addressing, the IP address space is divided into five **classes: A, B, C, D, and E. Each class occupies some part of the whole address space. Figure shows the** class occupation of the address space.



Class A: 2[illegible] = 2,147,483,648 addresses, 50%

Class B: 2^{30} = 1,073,741,824 addresses, 25%

Class C: 2^{29} = 536,870,912 addresses, 12.5%

Class D: 2^{28} = 268,435,456 addresses, 6.25%

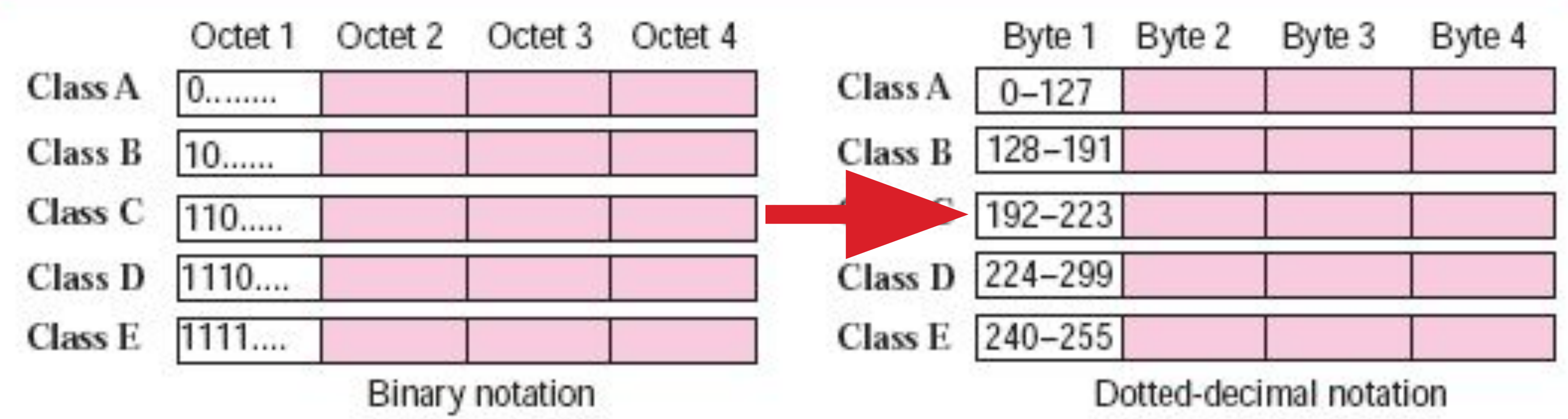
Class E: 2^{28} = 268,435,456 addresses, 6.25%

Class A range=00000000 to 01111111

Recognizing Classes

We can find the class of an address when the address is given either in binary or dotted-decimal notation. In the binary notation, the first few bits can immediately tell us the class of the address; in the dotted-decimal notation, the value of the first byte can give the class of an address (Figure 5.6).

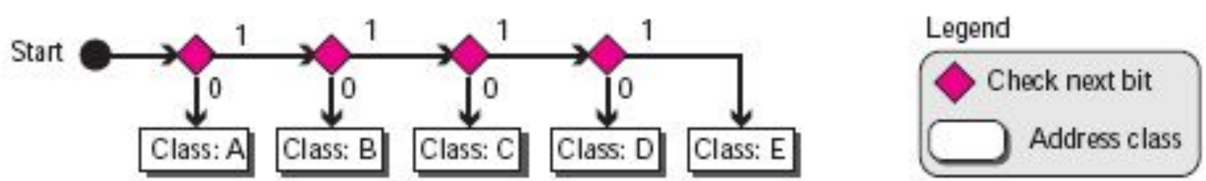
Finding the class of an address



Example
Find the class of each address:
a. 00000001 00001011 00001011 11101111
b. 11000001 10000011 00011011 11111111
c. 10100111 11011011 10001011 01101111
d. 11110011 10011011 11111011 00001111

Computers often store IPv4 addresses in binary notation. In this case, it is very convenient to write an algorithm to use a continuous checking process for finding the address as shown in Figure 5.7.

Finding the address class using continuous checking



- **Solution**
- See the procedure in Figure 5.7.
- **a. The first bit is 0. This is a class A address.**
- **b. The first 2 bits are 1; the third bit is 0. This is a class C address.**
- **c. The first bit is 1; the second bit is 0. This is a class B address.**
- **d. The first 4 bits are 1s. This is a class E address.**

- **Example 5.11**
- Find the class of each address:
- **a. 227.12.14.87**
- **b. 193.14.56.22**
- **c. 14.23.120.8**
- **d. 252.5.15.111**

- **Solution**
- **a. The first byte is 227 (between 224 and 239); the class is D.**
- **b. The first byte is 193 (between 192 and 223); the class is C.**
- **c. The first byte is 14 (between 0 and 127); the class is A.**
- **d. The first byte is 252 (between 240 and 255); the class is E.**

Two-Level Addressing

- The whole purpose of IPv4 addressing is to define a destination for an Internet packet (at the network layer).
- When classful addressing was designed, it was assumed that the whole Internet is divided into many networks and each network connects many hosts.
- In other words, the Internet was seen as a network of networks. A network was normally
- created by an organization that wanted to be connected to the Internet. The Internet
- authorities allocated a block of addresses to the organization (in class A, B, or C).

- Since all addresses in a network belonged to a single block, each address in classful addressing contains two parts: netid and hostid.
- The netid defines the network; the hostid defines a particular host connected to that network. Figure 5.14 shows an IPv4
- address in classful addressing. If *n bits in the class defines the net, then 32 − n bits*
- defines the host. However, the value of *n depends on the class the block belongs to. The*
- value of *n can be 8, 16 or 24 corresponding to classes A, B, and C respectively.*

Figure 5.14 *Two-level addressing in classful addressing*

Two-level addressing can be found in other communication systems. For example, a telephone system inside the United States can be thought of as two parts: area code and local part. The area code defines the area, the local part defines a particular telephone subscriber in that area.

(626) 3581301

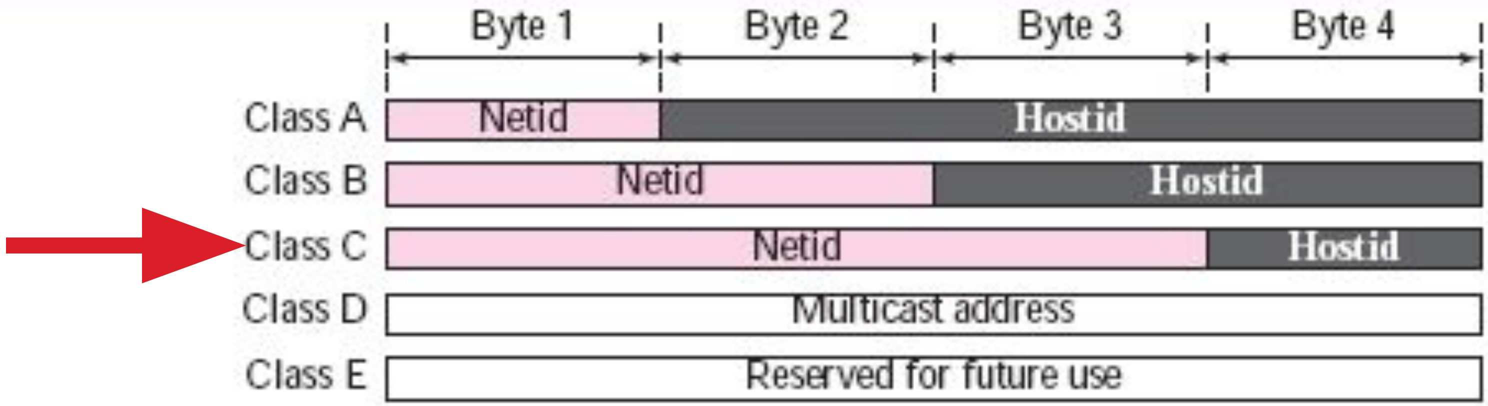
The area code, 626, can be compared with the netid, the local part, 3581301, can be compared to the hostid.

Netid and Hostid

- In classful addressing, an IP address in classes A, B, and C is divided into **netid and hostid.**
- These parts are of varying lengths, depending on the class of the address.
- Note that classes D and E are not divided into netid and hostid.

2 to the power 24=1.67 crores

Figure 5.8 *Netid and hostid*



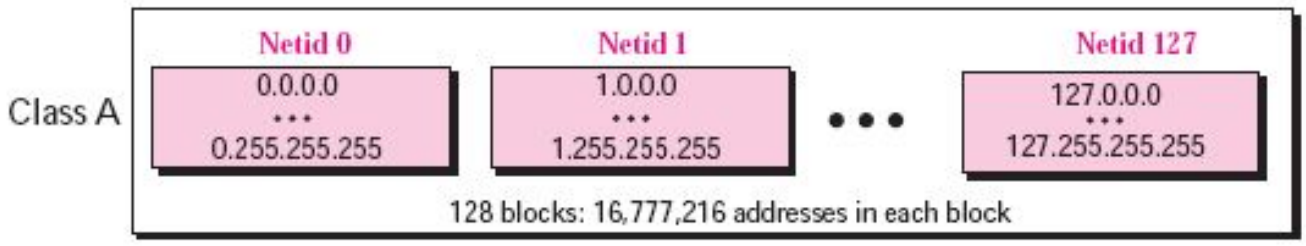
In class A, 1 byte defines the netid and 3 bytes define the hostid. In class B, 2 bytes define the netid and 2 bytes define the hostid. In class C, 3 bytes define the netid and 1 byte defines the hostid.

Classes and Blocks

- One problem with classful addressing is that each class is divided into a fixed number of blocks with each block having a fixed size. Let us look at each class.

- ***Class A*** ***(***2147483648 addresses***)***
- Since only 1 byte in class A defines the netid and the leftmost bit should be 0, the next 7 bits can be changed to find the number of blocks in this class. Therefore, class A is divided into $2^7 = 128$ blocks that can be assigned to 128 organizations.
- However, each block in this class contains (2147483648/128) = 16,777,216 (2^{24}) IP addresses, which means *the organization should be a really large one* to use all these addresses. Many addresses are wasted in this class.

Figure 5.9 *Blocks in class A*

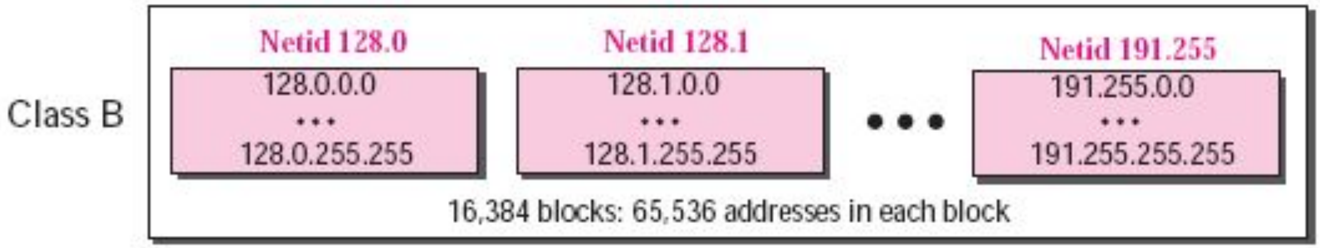


Millions of class A addresses are wasted.

Class B (1073741824 addresses)

- Since 2 bytes in class B define the class and the two leftmost bit should be 10 (fixed), the next 14 bits can be changed to find the number of blocks in this class.
- Therefore, class B is divided into 2^{14} = 16,384 blocks that can be assigned to 16,384 organizations.
- However, each block in this class contains 2^{16} 65,536 addresses. (1073741824/16384=65536)
- Not so many organizations can use so many addresses. Many addresses are wasted in this class.

Figure 5.10 *Blocks in class B*

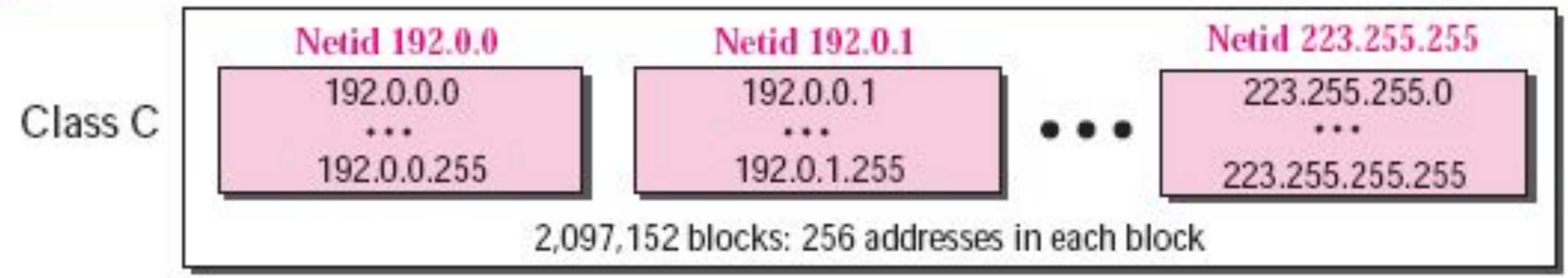


Many class B addresses are wasted.

Class C (536870912 addresses)

- Since 3 bytes in class C define the class and the three leftmost bits should be 110 (fixed), the
- next 21 bits can be changed to find the number of blocks in this class.
- Therefore, class C is divided into 2^{21} = 2,097,152 blocks, in which each block contains 256 addresses, that can be assigned to 2,097,152 organizations.
- However, not so many organizations were so small as to be satisfied with a class C block.

Figure 5.11 *Blocks in class C*



Not so many organizations are so small to have a class C block.

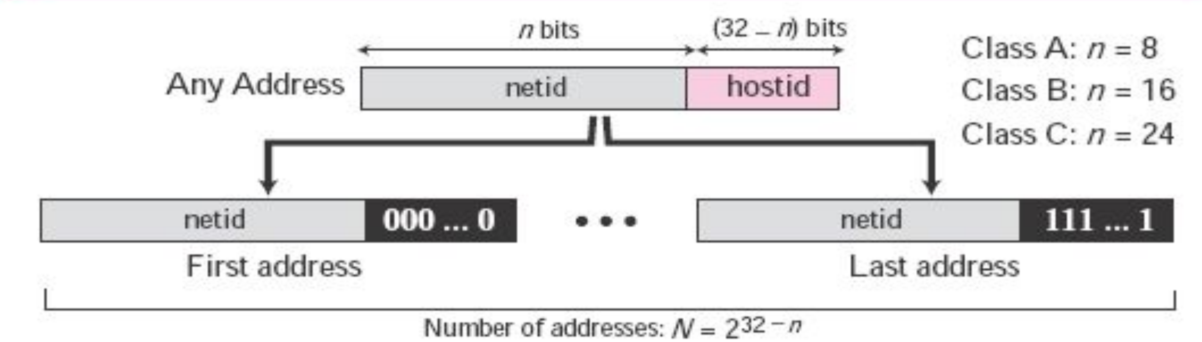
- Private IP Address and Public IP Address are used to uniquely identify a machine on the internet.
- Private IP address is used with a local network and public IP address is used outside the network.
- Public IP address is provided by ISP, Internet Service Provider.

Sr. No.	Key	Private IP Address	Public IP Address
1	Scope	Private IP address scope is local to present network.	Public IP address scope is global.
2	Communication	Private IP Address is used to communicate within the network.	Public IP Address is used to communicate outside the network.
3	Format	Private IP Addresses differ in a uniform manner.	Public IP Addresses differ in varying range.
4	Provider	Local Network Operator creates private IP addresses using network operating system.	ISP, Internet Service Provider controls the public IP address.
5	Cost	Private IP Addresses are free of cost.	Public IP Address comes with a cost.
6	Locate	Private IP Address can be located using ipconfig command.	Public IP Address needs to be searched on search engine like google.
7	Range	Private IP Address range:10.0.0.0 – 10.255.255.255, 172.16.0.0 – 172.31.255.255, 192.168.0.0 – 192.168.255.255	Except private IP Addresses, rest IP addresses are public.
8	Example	Private IP Address is like 192.168.11.50.	Public IP Address is like 17.5.7.8.

Extracting Information in a Block

- **A block is a range of addresses.**
- Given any address in the block, we normally like to know three pieces of information about the block: the number of addresses, the first address, and the last address.
- Before we can extract these pieces of information, we need to know the class of the address, which we showed how to find in the previous section.
- After the class of the block is found, we know the value of *n, the length* of netid in bits. We can now find these three pieces of information.
- 1. The number of addresses in the block, *N, can be found using* $N = 2^{32-n}$.
- **2.** To find the first address, we keep the *n* leftmost bits and set the $(32-n)$ rightmost bits all to 0s.
- **3.** To find the last address, we keep the *n* leftmost bits and set the $(32-n)$ rightmost bits all to 1s.

Figure 5.15 *Information extraction in classful addressing*

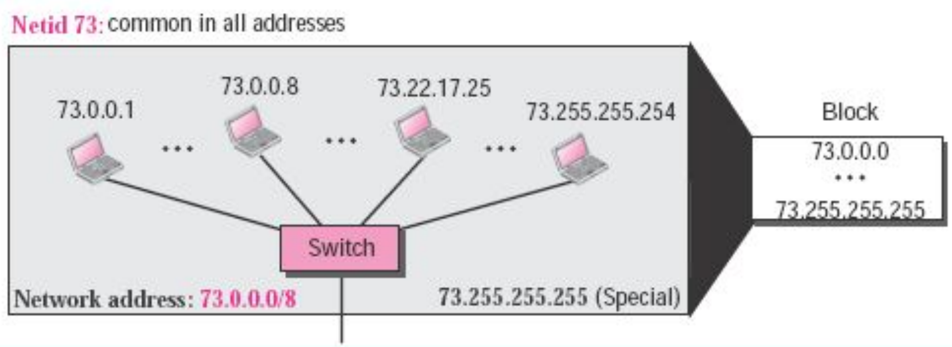


An address in a block is given as 73.22.17.25. Find the number of addresses in the block, the first address, and the last address.

Solution

Since 73 is between 0 and 127, the class of the address is A. The value of n for class A is 8. Figure 5.16 shows a possible configuration of the network that uses this block. Note that we show the value of n in the network address after a slash. Although this was not a common practice in classful addressing, it helps to make it a habit in classless addressing in the next section.

Figure 5.16 *Solution to Example 5.13*



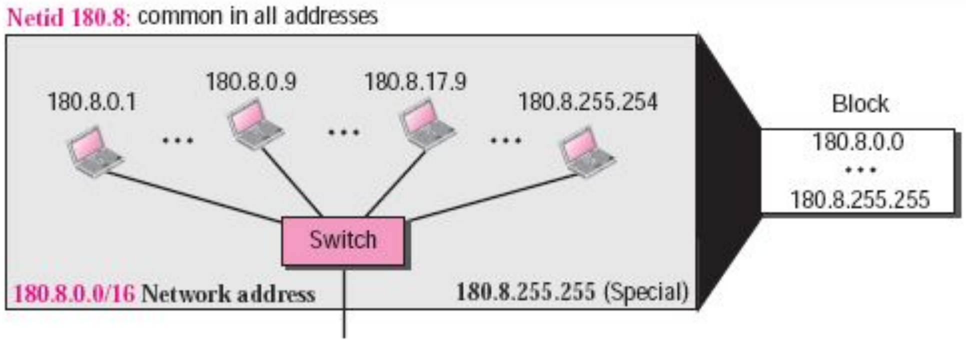
1. The number of addresses in this block is $N = 2^{32-n} = 2^{24} = 16{,}777{,}216$.
2. To find the first address, we keep the leftmost 8 bits and set the rightmost 24 bits all to 0s. The first address is 73.0.0.0/8 in which 8 is the value of n. The first address is called the *network address* and is not assigned to any host. It is used to define the network.
3. To find the last address, we keep the leftmost 8 bits and set the rightmost 24 bits all to 1s. The last address is 73.255.255.255. The last address is normally used for a special purpose, as discussed later in the chapter.

- TEST1
  An address in a block is given as 180.8.17.9. Find the number of addresses in the block, the first address, and the last address.

Solution

Since 180 is between 128 and 191, the class of the address is B. The value of n for class B is 16. Figure 5.17 shows a possible configuration of the network that uses this block.

Figure 5.17 *Solution to Example 5.14*



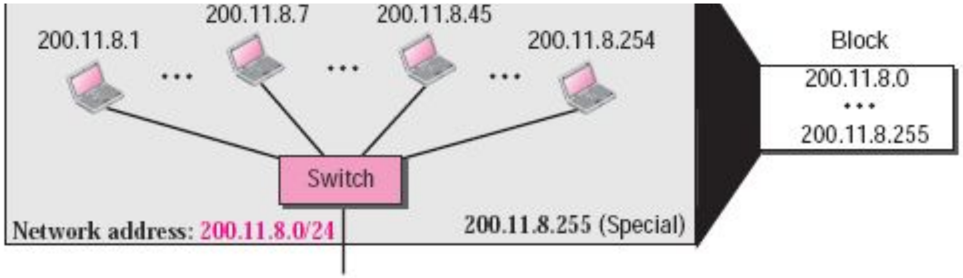
1. The number of addresses in this block is $N = 2^{32-n} = 2^{16} = 65{,}536$.
2. To find the first address, we keep the leftmost 16 bits and set the rightmost 16 bits all to 0s. The first address (network address) is 18.8.0.0/16, in which 16 is the value of n.
3. To find the last address, we keep the leftmost 16 bits and set the rightmost 16 bits all to 1s. The last address is 18.8.255.255.

- **Test 2**
- An address in a block is given as 200.11.8.45. Find the number of addresses in the block, the first address, and the last address

Solution

Since 200 is between 192 and 223, the class of the address is C. The value of n for class C is 24. Figure 5.18 shows a possible configuration of the network that uses this block.

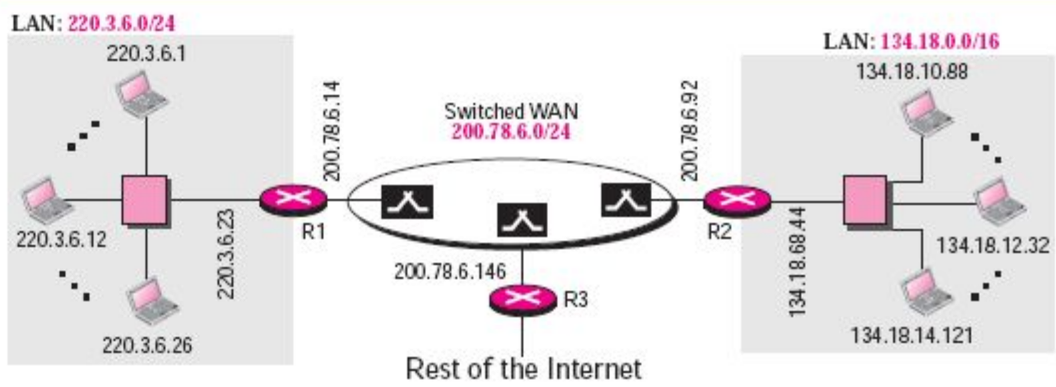
1. The number of addresses in this block is $N = 2^{32-n} = 2^8 = 256$.
2. To find the first address, we keep the leftmost 24 bits and set the rightmost 8 bits all to 0s. The first address is 200.11.8.0/24. The first address is called the network address.
3. To find the last address, we keep the leftmost 24 bits and set the rightmost 8 bits all to 1s. The last address is 200.11.8.255.



An Example

Figure 5.19 shows a hypothetical part of an internet with three networks.

Figure 5.19 *Sample internet*



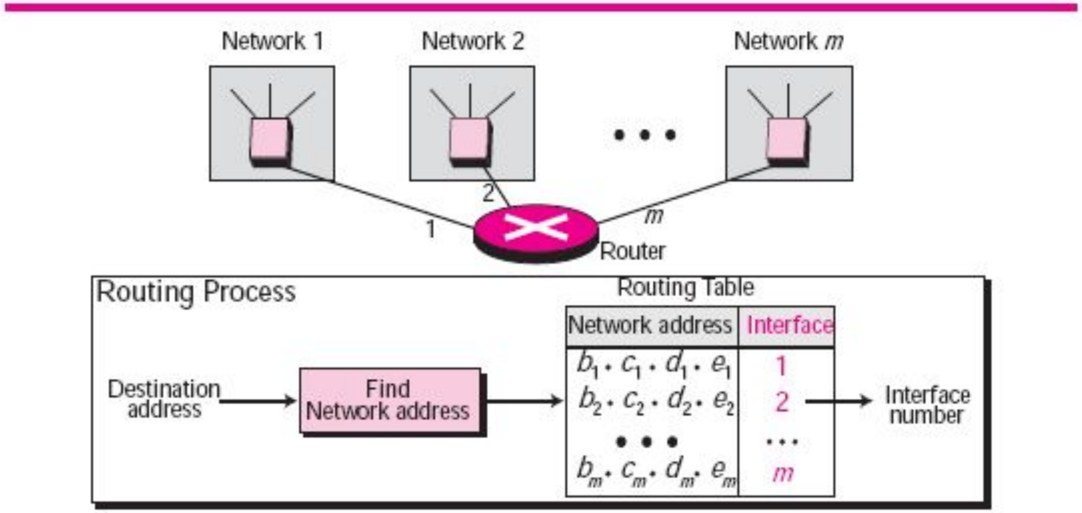
We have

1. A LAN with the network address 220.3.6.0 (class C).
2. A LAN with the network address 134.18.0.0 (class B).
3. A switched WAN (class C), such as Frame Relay or ATM, that can be connected to many routers. We have shown three. One router connects the WAN to the left LAN, one connects the WAN to the right LAN, and one connects the WAN to the rest of the internet.

Network Address

- The first address, **network address,** is particularly important because it is used in routing a packet to its destination network. For the moment, let us assume that an internet is made of *m* networks and a router with *m* interfaces. When a packet arrives at the router from any source host, the router needs to know to which network the packet should be sent; the router needs to know from which interface the packet should be sent out.
- When the packet arrives at the network, it reaches its destination host using another strategy that we discuss in later chapters. Figure 5.20 shows the idea. After the network address has been found, the router consults its routing table to find the corresponding interface from which the packet should be sent out. The network address is actually the identifier of the network; each network is identified by its network address.

Figure 5.20 *Network address*

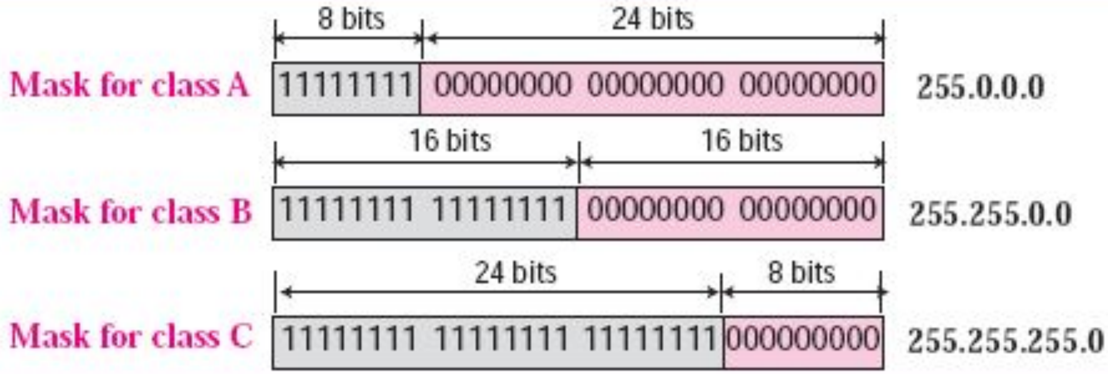


Network Mask

Network Mask

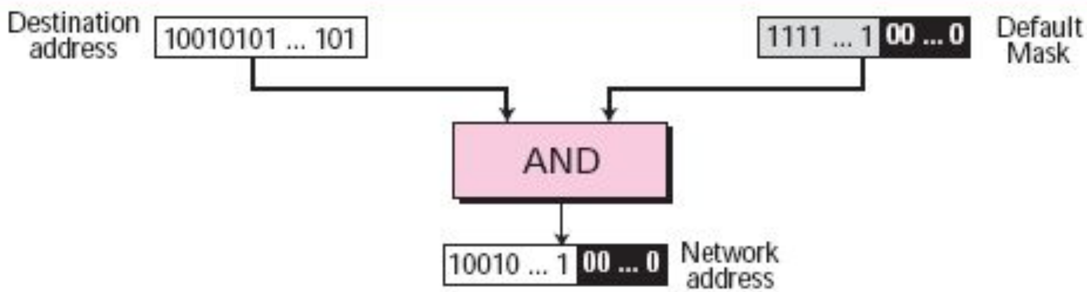
The methods we described previously for extracting the network address are mostly used to show the concept. The routers in the Internet normally use an algorithm to extract the network address from the destination address of a packet. To do this, we need a network mask. A **network mask** or a **default mask** in classful addressing is a 32-bit number with *n* leftmost bits all set to 1s and (32 – *n*) rightmost bits all set to 0s. Since *n* is different for each class in classful addressing, we have three default masks in classful addressing as shown in Figure 5.21.

Figure 5.21 *Network mask*



To extract the network address from the destination address of a packet, a router uses the AND operation described in the previous section. When the destination address (or any address in the block) is ANDed with the default mask, the result is the network address (Figure 5.22). The router applies the AND operation on the binary (or hexadecimal representation) of the address and the mask, but when we show an example, we use the short cut discussed before and apply the mask on the dotted-decimal notation. The default mask can also be used to find the number of addresses in the block and the last address in the block, but we discuss these applications in classless addressing.

Figure 5.22 *Finding a network address using the default mask*



Do yourself

- A router receives a packet with the destination address 201.24.67.32. Show how the router finds the network address of the packet.

Solution

We assume that the router first finds the class of the address and then uses the corresponding default mask on the destination address, but we need to know that a router uses another strategy as we will discuss in the next chapter. Since the class of the address is B, we assume that the router applies the default mask for class B, 255.255.0.0 to find the network address.

Destination address	→	201	.	24	.	67	.	32
Default mask	→	255	.	255	.	0 255	.	0
Network address	→	201	.	24	.	0 67	.	0

We have used the first short cut as described in the previous section. The network address is 201.24.0.0 as expected.

Three-Level Addressing: Subnetting

- As we discussed before, the IP addresses were originally designed with two levels of addressing. To reach a host on the Internet, we must first reach the network and then the host.
- But It soon became clear that we need more than two hierarchical levels, for two reasons.

  First, an organization that was granted a block in class A or B needed to divide its large network into several subnetworks for better security and management.
- Second, since the blocks in class A and B were almost depleted (low) and the blocks in class C were smaller than the needs of most organizations, an organization that has been granted a block in class A or B could divide the block into smaller subblocks and share them with other organizations.

IMPORTANT!!!

- The idea of splitting a block to smaller blocks is referred to as subnetting.
- In **subnetting, a network is divided into several smaller subnetworks (subnets)** with each subnetwork having its own subnetwork address.

Example 5.17

Three-level addressing can be found in the telephone system if we think about the local part of a telephone number as an exchange and a subscriber connection:

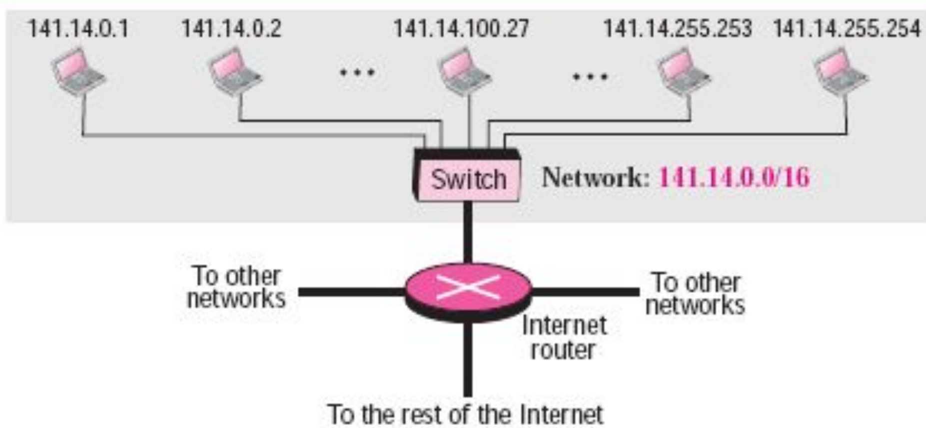
(626) 358 - 1301

in which 626 is the area code, 358 is the exchange, and 1301 is the subscriber connection.

Example 5.18

Figure 5.23 shows a network using class B addresses before subnetting. We have just one network with almost 2^{16} hosts. The whole network is connected, through one single connection, to one of the routers in the Internet. Note that we have shown /16 to show the length of the netid (class B).

Figure 5.23 *Example 5.18*



- Figure 5.24 shows the same network in Figure 5.23 after subnetting. The whole network is still connected to the Internet through the same router. However, the network has used a private router to divide the network into four subnetworks. The rest of the Internet still sees only one network; internally the network is made of four subnetworks.
- Each subnetwork can now have almost 2^{14} hosts. The network can belong to a university campus with four different schools (buildings).
- After subnetting, each school has its own subnetworks, but still the whole campus is one network for the rest of the Internet. Note that /16 and /18 show the length of the netid and subnetids.

Figure 5.23 *Example 5.18*

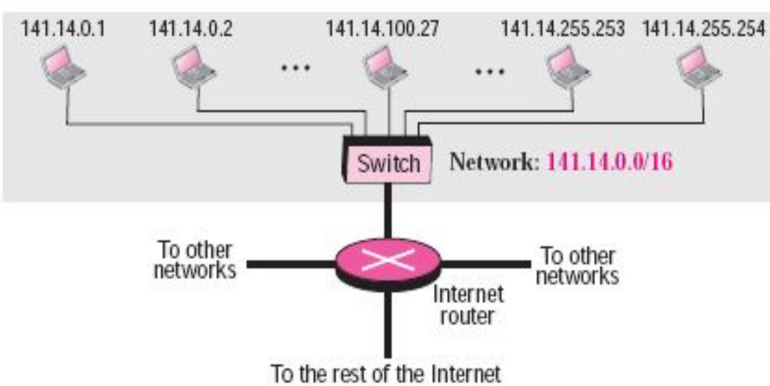
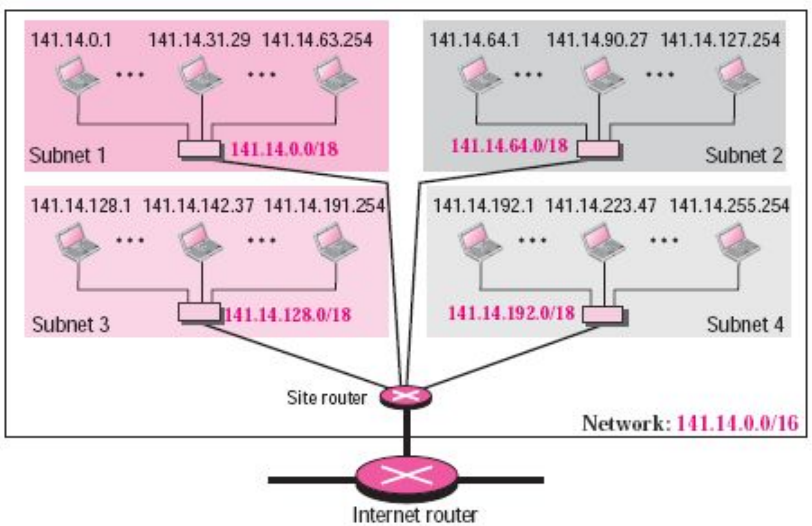


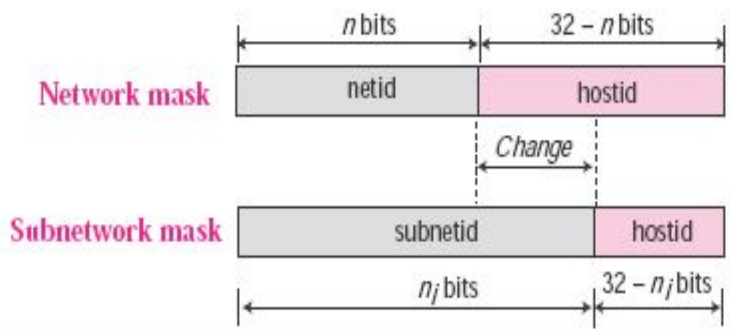
Figure 5.24 *Example 5.19*



Subnet Mask

- We discussed the network mask (default mask) before. The network mask is used when a network is not subnetted. When we divide a network to several subnetworks, we need to create a subnetwork mask (or subnet mask) for each subnetwork. A subnetwork has subnetid and hostid as shown in Figure 5.25.

Figure 5.25 *Network mask and subnetwork mask*



Subnetting increases the length of the netid and decreases the length of hostid. When we divide a network to s number of subnetworks, each of equal numbers of hosts, we can calculate the subnetid for each subnetwork as

$$n_{sub} = n + \log_2 s$$

in which n is the length of netid, n_{sub} is the length of each subnetid, and s is the number of subnets which must be a power of 2.

Subnet Mask

Example 5.20

In Example 5.19, we divided a class B network into four subnetworks. The value of $n = 16$ and the value of $n_1 = n_2 = n_3 = n_4 = 16 + \log_2 4 = 18$. This means that the subnet mask has eighteen 1s and fourteen 0s. In other words, the subnet mask is 255.255.192.0 which is different from the network mask for class B (255.255.0.0).

Figure 5.23 *Example 5.18*

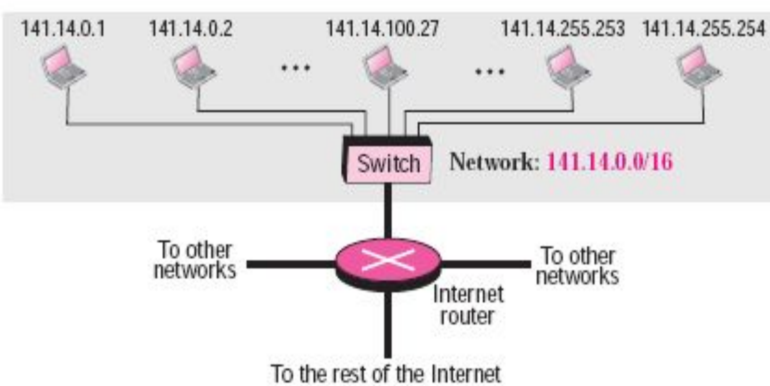
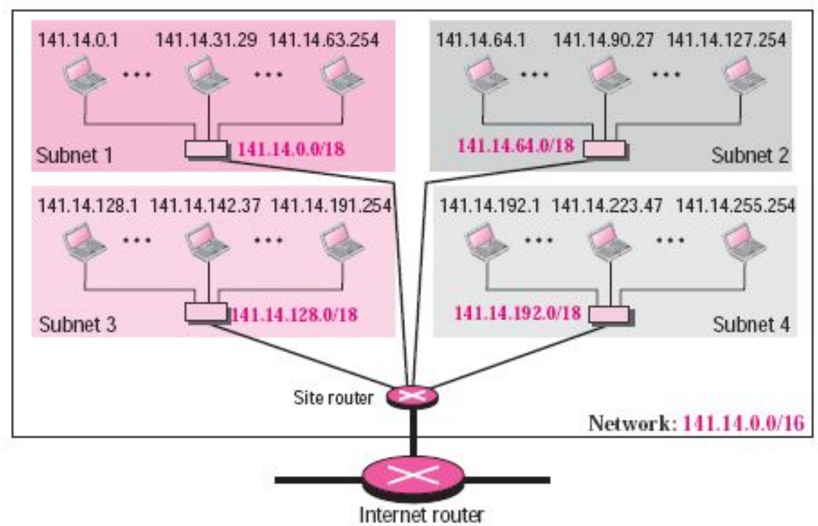


Figure 5.24 *Example 5.19*



Subnet Address

- When a network is subnetted, the first address in the subnet is the identifier of the subnet
- and is used by the router to route the packets destined for that subnetwork. Given
- any address in the subnet, the router can find the subnet mask using the same procedure
- we discussed to find the network mask: ANDing the given address with the subnet
- mask. The short cuts we discussed in the previous section can be used to find the subnet
- address.

When a network is subnetted, the first address in the subnet is the identifier of the subnet and is used by the router to route the packets destined for that subnetwork. Given any address in the subnet, the router can find the subnet mask using the same procedure we discussed to find the network mask: ANDing the given address with the subnet mask. The short cuts we discussed in the previous section can be used to find the subnet address.

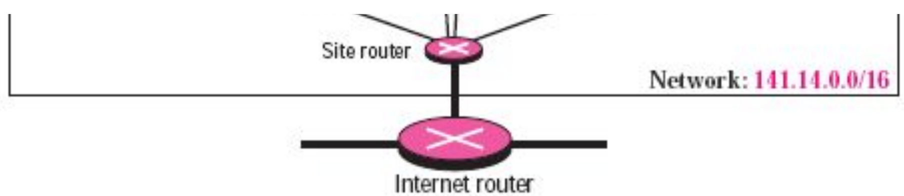
Example 5.21

In Example 5.19, we show that a network is divided into four subnets. Since one of the addresses in subnet 2 is 141.14.120.77, we can find the subnet address as:

Address	→	141	.	14	.	120	.	77
Mask	→	255	.	255	.	192	.	0
Subnet Address	→	141	.	14	.	64	.	0

The values of the first, second, and fourth bytes are calculated using the first short cut for AND operation. The value of the third byte is calculated using the second short cut for the AND operation.

Address (120)	0	+	64	+	32	+	16	+	8	+	0	+	0	+	0
Mask (192)	128	+	64	+	0	+	0	+	0	+	0	+	0	+	0
Result (64)	0	+	64	+	0	+	0	+	0	+	0	+	0	+	0



Classless addressing

- Subnetting in classful addressing did not really solve the address depletion problem and made the distribution of addresses and the routing process more difficult.
- With the growth of the Internet, it was clear that a larger address space was needed as a long-term solution. The larger address space, however, requires that the length of IP addresses to be increased, which means the format of the IP packets needs to be changed.
- Although the long-range solution has already been devised and is called IPv6, a short-term solution was also devised to use the same address space but to change the distribution of addresses to provide a fair share to each organization.
- The short-term solution still uses IPv4 addresses, but it is called *classless* addressing.
- In other words, the class privilege was removed from the distribution to compensate for the address depletion.

Motivation

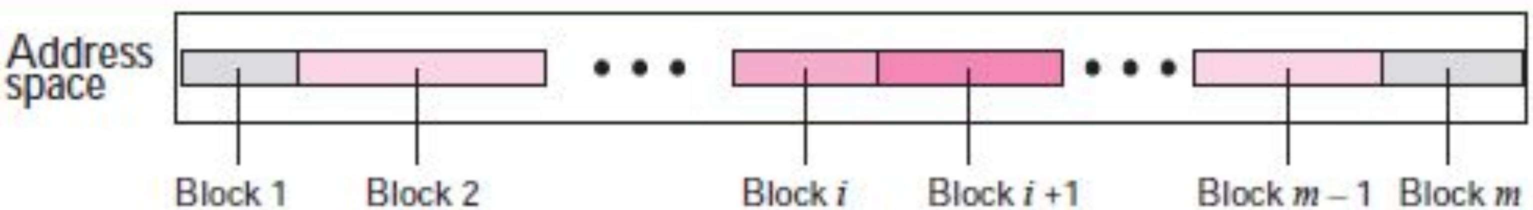
- There was another motivation for classless addressing. During the 1990s, Internet service providers (ISPs) came into prominence. An ISP is an organization that provides Internet access for individuals, small businesses, and midsize organizations that do not want to create an Internet site and become involved in providing Internet services (such as e-mail services) for their employees.
- An ISP can provide these services. An ISP is granted a large range of addresses and then subdivides the addresses (in groups of 1, 2, 4, 8, 16, and so on), giving a range of addresses to a household or a small business.
- The customers are connected via a dial-up modem, DSL, or cable modem to the ISP.
- However, each customer needs some IPv4 addresses.
- In 1996, the Internet authorities announced a new architecture called classless addressing.
- In classless addressing, variable-length blocks are used that belong to no classes. We can have a block of 1 address, 2 addresses, 4 addresses, 128 addresses, and so on.

Variable-Length Blocks

- In classful addressing the whole address space was divided into five classes. Although each organization was granted one block in class A, B, or C, the size of the blocks was predefined; the organization needed to choose one of the three block sizes.
- In classless addressing, the whole address space is divided into variable length blocks.
- Theoretically, we can have a block of $2^0, 2^1, 2^2, \ldots, 2^{32}$ addresses.
- The only restriction, as we discuss later, is that the number of addresses in a block needs to be a power of 2.
- An organization can be granted one block of addresses.

Figure 5.27 shows the division of the whole address space into nonoverlapping blocks.

Figure 5.27 *Variable-length blocks in classless addressing*



Two-Level Addressing

- In classful addressing, two-level addressing was provided by dividing an address into *netid and hostid. The netid defined the network; the hostid defined the host in the network.*
- The same idea can be applied in classless addressing. When an organization is granted a block of addresses, the block is actually divided into two parts, the **prefix and** the **suffix.**
- **The prefix plays the same role as the netid; the suffix plays the same role as** the hostid. All addresses in the block have the same prefix; each address has a different suffix.

Figure 5.28 shows the prefix and suffix in a classless block.

Figure 5.28 *Prefix and suffix*

In classful addressing, the length of the netid, n, depends on the class of the address; it can be only 8, 16, or 24. In classless addressing, the length of the prefix, n, depends on the size of the block; it can be 0, 1, 2, 3, . . . , 32. In classless addressing, the value of n is referred to as **prefix length**; the value of $32 - n$ is referred to as **suffix length**.

Subnet Masks (cont)

0	0.0.0.0		
1	128.0.0.0	17	255.255.128.0
2	192.0.0.0	18	255.255.192.0
3	224.0.0.0	19	255.255.224.0
4	240.0.0.0	20	255.255.240.0
5	248.0.0.0	21	255.255.248.0
6	252.0.0.0	22	255.255.252.0
7	254.0.0.0	23	255.255.254.0
8	255.0.0.0	24	255.255.255.0
9	255.128.0.0	25	255.255.255.128
10	255.192.0.0	26	255.255.255.192
11	255.224.0.0	27	255.255.255.224
12	255.240.0.0	28	255.255.255.240
13	255.248.0.0	29	255.255.255.248
14	255.252.0.0	30	255.255.255.252
15	255.254.0.0	31	255.255.255.254
16	255.255.0.0	32	255.255.255.255

NOTE!!

- The number of addresses in a block is inversely related to the value of the prefix length, *n*.
- *A* small *n means a larger block; a large n means a small block.*

Slash Notation

- The netid length in classful addressing or the prefix length in classless addressing play a very important role when we need to extract the information about the block from a given address in the block.
- However, there is a difference here in classful and classless addressing.

❑ In classful addressing, the netid length is inherent in the address. Given an address, we know the class of the address that allows us to find the netid length (8, 16, or 24).

- ❑ In classless addressing, the prefix length cannot be found if we are given only an address in the block. The given address can belong to a block with any prefix length.

- In classless addressing, we need to include the prefix length to each address if we need to find the block of the address. In this case, the prefix length, *n, is added to the address* separated by a slash. The notation is informally referred to as **slash notation.**

An address in classless addressing can then be represented as shown in Figure 5.29.

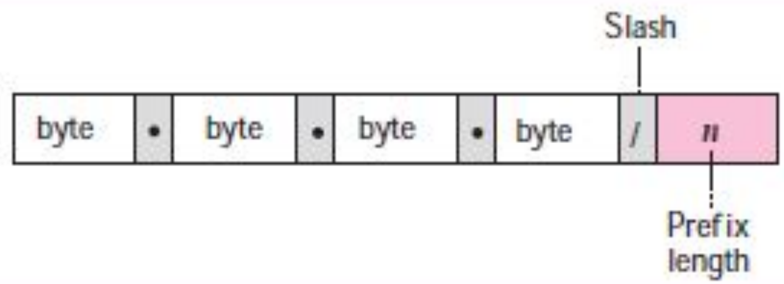


Figure 5.29 *Slash notation*

The slash notation is formally referred to as **classless inter domain routing or CIDR (pronounced cider) notation.**

In classless addressing, we need to know one of the addresses in the block and the prefix length to define the block.

Example 5.25

In classless addressing, an address cannot per se define the block the address belongs to. For example, the address 230.8.24.56 can belong to many blocks some of them are shown below with the value of the prefix associated with that block:

Prefix length:16	→	Block:	230.8.0.0	to	230.8.255.255
Prefix length:20	→	Block:	230.8.16.0	to	230.8.31.255
Prefix length:26	→	Block:	230.8.24.0	to	230.8.24.63
Prefix length:27	→	Block:	230.8.24.32	to	230.8.24.63
Prefix length:29	→	Block:	230.8.24.56	to	230.8.24.63
Prefix length:31	→	Block:	230.8.24.56	to	230.8.24.57

Network Mask

- The idea of network mask in classless addressing is the same as the one in classful addressing. A network mask is a 32-bit number with the *n leftmost bits all set to 0s and* the rest of the bits all set to 1s.

- **Example**
- The following addresses are defined using slash notations.
- **a. In the address 12.23.24.78/8, the network mask is 255.0.0.0. The mask has eight** 1s and twenty-four 0s. The prefix length is 8; the suffix length is 24.
- **b. In the address 130.11.232.156/16, the network mask is 255.255.0.0. The mask** has sixteen 1s and sixteen 0s.The prefix length is 16; the suffix length is 16.
- **c. In the address 167.199.170.82/27, the network mask is 255.255.255.224. The** mask has twenty-seven 1s and five 0s. The prefix length is 27; the suffix length is 5.

Extracting Block Information

An address in slash notation (CIDR) contains all information we need about the block: the first address (network address), the number of addresses, and the last address. These three pieces of information can be found as follows:

- The number of addresses in the block can be found as:

$$N = 2^{32-n}$$

  in which n is the prefix length and N is the number of addresses in the block.

- The first address (network address) in the block can be found by ANDing the address with the network mask:

First address = (any address) AND (network mask)

  Alternatively, we can keep the n leftmost bits of any address in the block and set the 32− n bits to 0s to find the first address.

- The last address in the block can be found by either adding the first address with the number of addresses or, directly, by ORing the address with the complement (NOTing) of the network mask:

Last address = (any address) OR [NOT (network mask)]

  Alternatively, we can keep the n leftmost bits of any address in the block and set the 32 − n bits to 1s to find the last address.

Test 5

- One of the addresses in a block is 167.199.170.82/27. Find the number of addresses in the network, the first address, and the last address.

Solution

The value of n is 27. The network mask has twenty-seven 1s and five 0s. It is 255.255.255.240.

a. The number of addresses in the network is $2^{32-n} = 2^{32-n} = 2^5 = 32$.

b. We use the AND operation to find the first address (network address).The first address is 167.199.170.64/27.

Address in binary:	10100111 11000111 10101010 01010010
Network mask:	11111111 11111111 11111111 11100000
First address:	10100111 11000111 10101010 01000000

c. To find the last address, we first find the complement of the network mask and then OR it with the given address: The last address is 167.199.170.95/27.

Address in binary:	10100111 11000111 10101010 01010010
Complement of network mask:	00000000 00000000 00000000 00011111
Last address:	10100111 11000111 10101010 01011111

Test 6

- One of the addresses in a block is 17.63.110.114/24. Find the number of addresses, the first address, and the last address in the block.

One of the addresses in a block is 17.63.110.114/24. Find the number of addresses, the first address, and the last address in the block.

Solution

The network mask is 255.255.255.0.

a. The number of addresses in the network is $2^{32-24} = 256$.

b. To find the first address, we use the short cut methods discussed early in the chapter.

Address:	17	.	63	.	110	.	114
Network mask:	255	.	255	.	255	.	0
First address (AND):	17	.	63	.	110	.	0

The first address is 17.63.110.0/24.

c. To find the last address, we use the complement of the network mask and the first short cut method we discussed before. The last address is 17.63.110.255/24.

Address:	17	.	63	.	110	.	114
Complement of the mask (NOT):	0	.	0	.	0	.	255
Last address (OR):	17	.	63	.	110	.	255

Test 7

- One of the addresses in a block is 110.23.120.14/20. Find the number of addresses, the first address, and the last address in the block.

Solution

The network mask is 255.255.240.0.

a. The number of addresses in the network is $2^{32-20} = 4096$.

b. To find the first address, we apply the first short cut to bytes 1, 2, and 4 and the second short cut to byte 3. The first address is 110.23.112.0/20.

Address:	110	.	23	.	120	.	14
Network mask:	255	.	255	.	240	.	0
First address (AND):	110	.	23	.	112	.	0

c. To find the last address, we apply the first short cut to bytes 1, 2, and 4 and the second short cut to byte 3. The OR operation is applied to the complement of the mask. The last address is 110.23.127.255/20.

Address:	110	.	23	.	120	.	14
Network mask:	0	.	0	.	15	.	255
Last address (OR):	110	.	23	.	127	.	255

Block Allocation

The next issue in classless addressing is block allocation. How are the blocks allocated? The ultimate responsibility of block allocation is given to a global authority called the Internet Corporation for Assigned Names and Addresses (ICANN). However, ICANN does not normally allocate addresses to individual Internet users. It assigns a large block of addresses to an ISP (or a larger organization that is considered an ISP in this case). For the proper operation of the CIDR, three restrictions need to be applied to the allocated block.

1. The number of requested addresses, N, needs to be a power of 2. This is needed to provide an integer value for the prefix length, n (see the second restriction). The number of addresses can be 1, 2, 4, 8, 16, and so on.
2. The value of prefix length can be found from the number of addresses in the block. Since $N = 2^{32-n}$, then $n = \log_2 (2^{32}/N) = 32 - \log_2 N$. That is the reason why N needs to be a power of 2.
3. The requested block needs to be allocated where there are a contiguous number of unallocated addresses in the address space. However, there is a restriction on choosing the beginning addresses of the block. The beginning address needs to be divisible by the number of addresses in the block. To see this restriction, we can show that the beginning address can be calculated as $X \times 2^{n-32}$ in which X is the decimal value of the prefix. In other words, the beginning address is $X \times N$.

Example 5.30

An ISP has requested a block of 1000 addresses. The following block is granted.

a. Since 1000 is not a power of 2, 1024 addresses are granted ($1024 = 2^{10}$).

b. The prefix length for the block is calculated as $n = 32 - \log_2 1024 = 22$.

c. The beginning address is chosen as 18.14.12.0 (which is divisible by 1024).

The granted block is 18.14.12.0/22. The first address is 18.14.12.0/22 and the last address is 18.14.15.255/22.

Relation to Classful Addressing

All issues discussed for classless addressing can be applied to classful addressing. As a matter of fact, classful addressing is a special case of the classless addressing in which the blocks in class A, B, and C have the prefix length $n_A = 8$, $n_B = 16$, and $n_C = 24$. A block in classful addressing can be easily changed to a block in class addressing if we use the prefix length defined in Table 5.1.

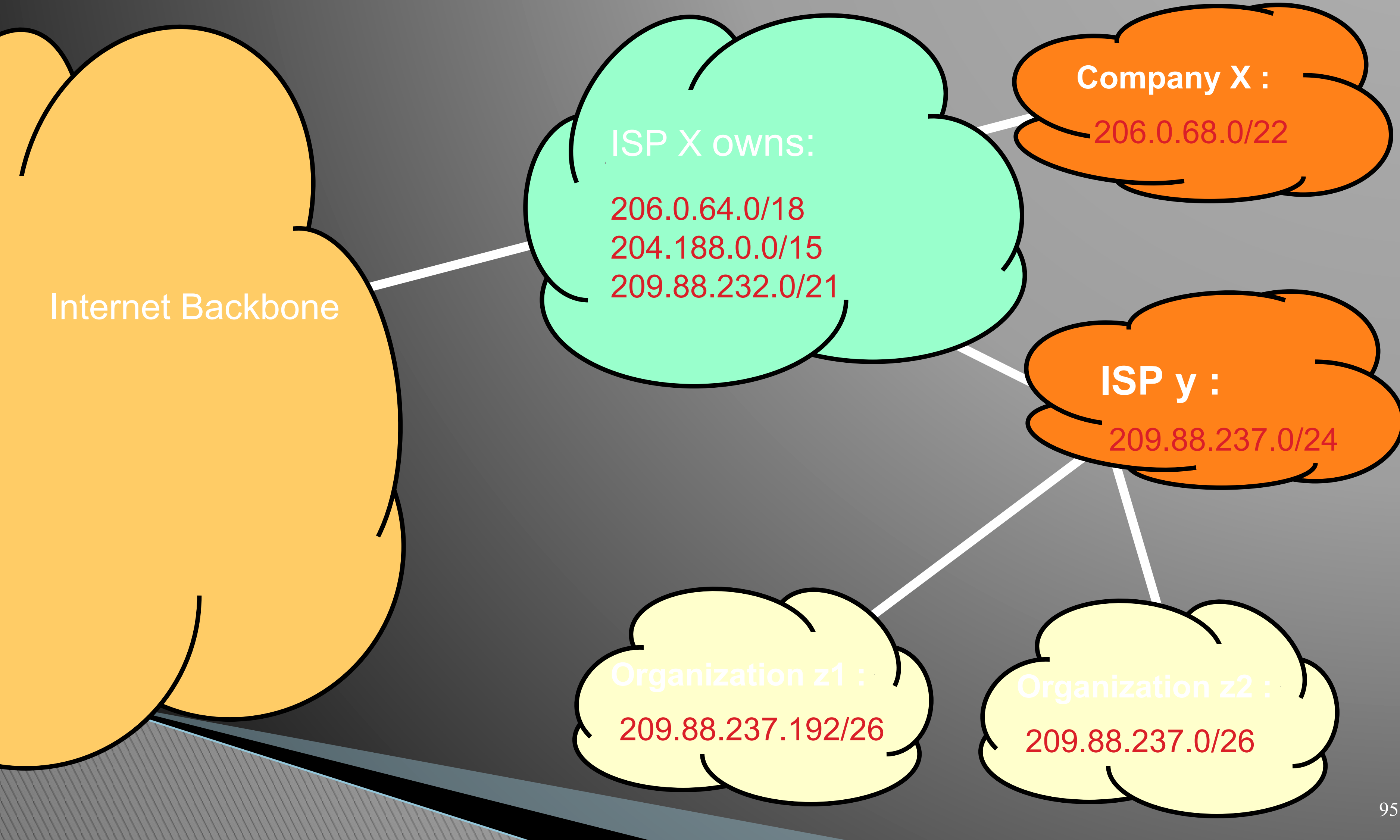
Table 5.1 *Prefix length for classful addressing*

Class	*Prefix length*	*Class*	*Prefix length*
A	/8	D	/4
B	/16	E	/4
C	/24		

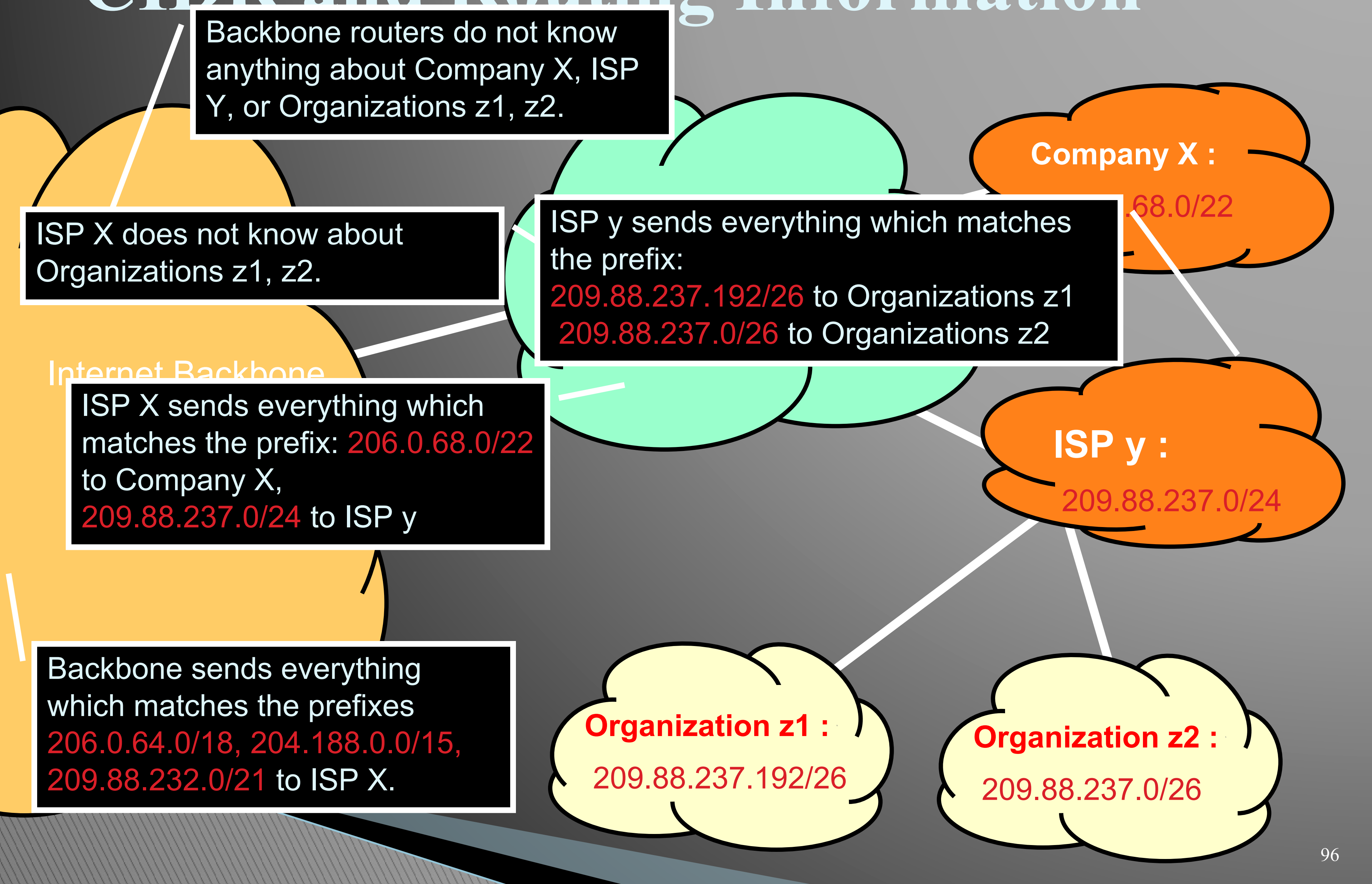
Subnetting

- Three levels of hierarchy can be created using subnetting. An organization (or an ISP) that is granted a range of addresses may divide the range into several subranges and assign each subrange to a **subnetwork (or subnet). The concept is the same as** we discussed for classful addressing. Note that nothing stops the organization from creating more levels. A subnetwork can be divided into several sub-subnetworks. A sub-subnetwork can be divided into several sub-sub-subnetworks. And so on.

CIDR and Routing Information



CIDR and Routing Information



Subnetting

- Three levels of hierarchy can be created using subnetting. An organization (or an ISP) that is granted a range of addresses may divide the range into several subranges and assign each subrange to a **subnetwork (or subnet). The concept is the same as** we discussed for classful addressing. Note that nothing stops the organization from creating more levels. A subnetwork can be divided into several sub-subnetworks. A sub-subnetwork can be divided into several sub-sub-subnetworks. And so on.

Designing Subnets

The subnetworks in a network should be carefully designed to enable the routing of packets. We assume the total number of addresses granted to the organization is N, the prefix length is n, the assigned number of addresses to each subnetwork is N_{sub}, the prefix length for each subnetwork is n_{sub}, and the total number of subnetworks is s. Then, the following steps need to be carefully followed to guarantee the proper operation of the subnetworks.

1. The number of addresses in each subnetwork should be a power of 2.
2. The prefix length for each subnetwork should be found using the following formula:

$$n_{sub} = n + \log_2 (N/N_{sub})$$

3. The starting address in each subnetwork should be divisible by the number of addresses in that subnetwork. This can be achieved if we first assign addresses to larger networks.

The restrictions applied in allocating addresses for a subnetwork are parallel to the ones used to allocate addresses for a network.

Test 8

- An organization is granted the block 130.34.12.64/26.
- The organization needs four subnetworks, each with an equal number of hosts.
- Design the subnetworks and find the information about each network.

Solution

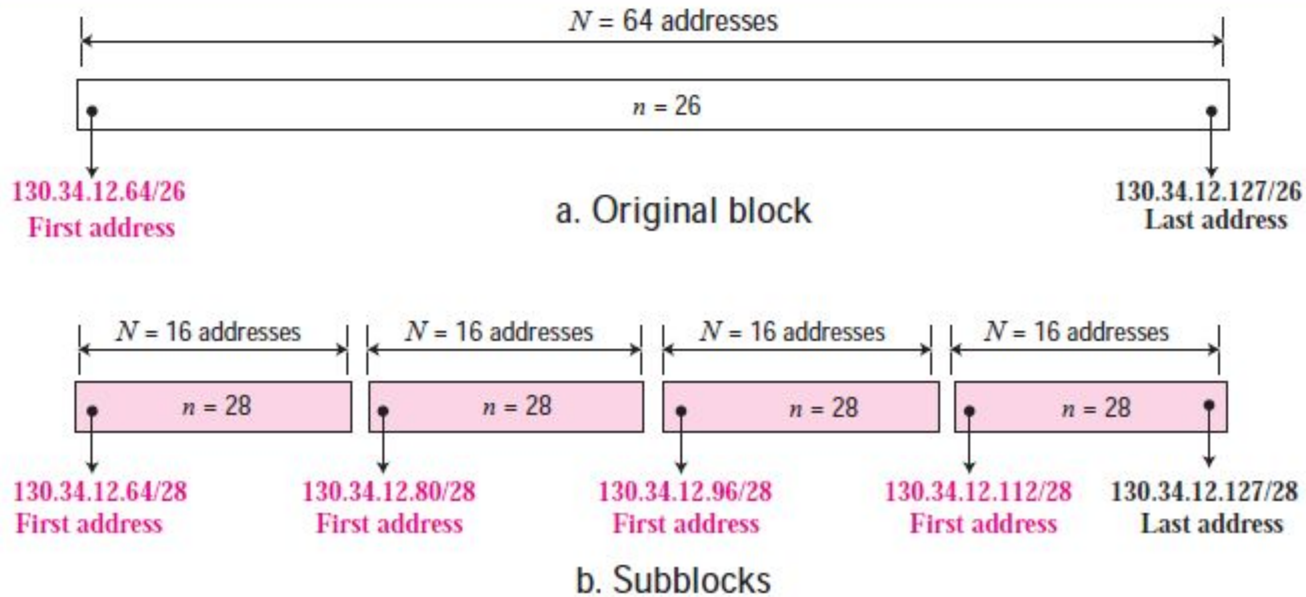
The number of addresses for the whole network can be found as $N = 2^{32-26} = 64$. Using the process described in the previous section, the first address in the network is 130.34.12.64/26 and the last address is 130.34.12.127/26. We now design the subnetworks:

1. We grant 16 addresses for each subnetwork to meet the first requirement (64/16 is a power of 2).
2. The **subnetwork** mask for each subnetwork is:

$$n_1 = n_2 = n_3 = n_4 = n + \log_2 (N/N_i) = 26 + \log_2 4 = 28$$

3. We grant 16 addresses to each subnet starting from the first available address. Figure 5.30 shows the subblock for each subnet. Note that the starting address in each subnetwork is divisible by the number of addresses in that subnetwork.

Figure 5.30 *Solution to Example 5.32*



- An organization is granted a block of addresses with the beginning address 14.24.74.0/24. The organization needs to have 3 subblocks of addresses to use in its three subnets as shown below:

- ❑ One subblock of 120 addresses.
- ❑ One subblock of 60 addresses.
- ❑ One subblock of 10 addresses.

Solution

There are $2^{32-24} = 256$ addresses in this block. The first address is 14.24.74.0/24; the last address is 14.24.74.255/24.

a. The number of addresses in the first subblock is not a power of 2. We allocate 128 addresses. The first can be used as network address and the last as the special address. There are still 126 addresses available. The subnet mask for this subnet can be found as $n_1 = 24 + \log_2 (256/128) = 25$. The first address in this block is 14.24.74.0/25; the last address is 14.24.74.127/25.

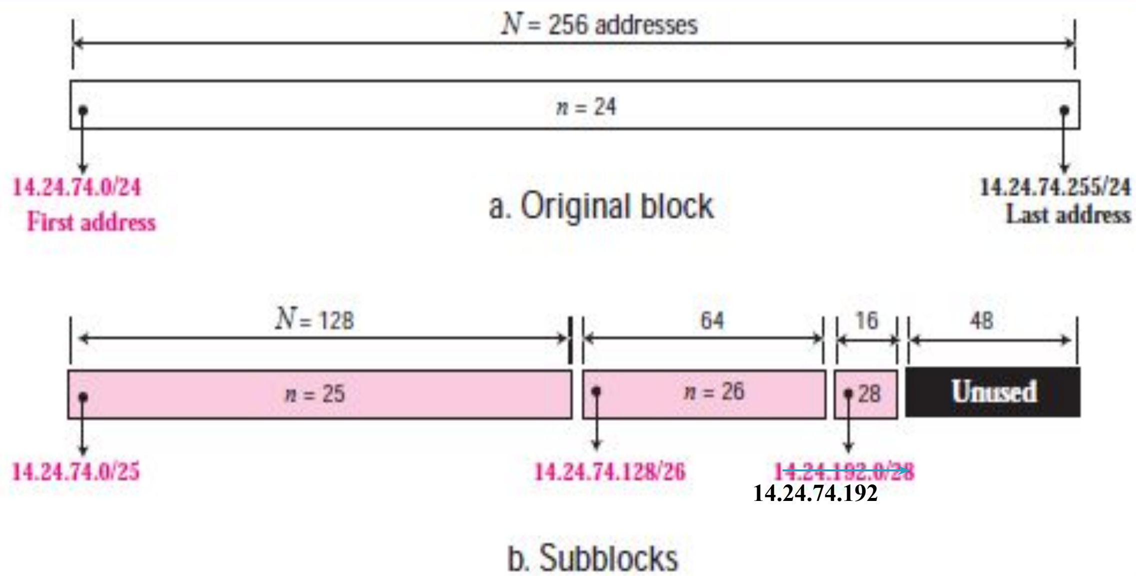
b. The number of addresses in the second subblock is not a power of 2 either. We allocate 64 addresses. The first can be used as network address and the last as the special address. There are still 62 addresses available. The subnet mask for this subnet can be found as $n_1 = 24 + \log_2 (256/64) = 26$. The first address in this block is 14.24.74.128/26; the last address is 14.24.74.191/26.

c. The number of addresses in the third subblock is not a power of 2 either. We allocate 16 addresses. The first can be used as network address and the last as the special address. There are still 14 addresses available. The subnet mask for this subnet can be found as $n_1 = 24 + \log_2 (256/16) = 28$. The first address in this block is 14.24.74.192/28; the last address is 14.24.74.207/28.

d. If we add all addresses in the previous subblocks, the result is 208 addresses, which means 48 addresses are left in reserve. The first address in this range is 14.24.74.209. The last address is 14.24.74.255. We don't know about the prefix length yet.

e. Figure 5.31 shows the configuration of blocks. We have shown the first address in each block.

Figure 5.31 *Solution to Example 5.33*



Test 8

- Assume a company has three offices: Central, East, and West. The Central office is connected to the East and West offices via private, point-to-point WAN lines.
- The company is granted a block of 64 addresses with the beginning address 70.12.100.128/26.
- The management has decided to allocate 32 addresses for the Central office and divides the rest of addresses between the two other offices.

1. The number of addresses are assigned as follows:

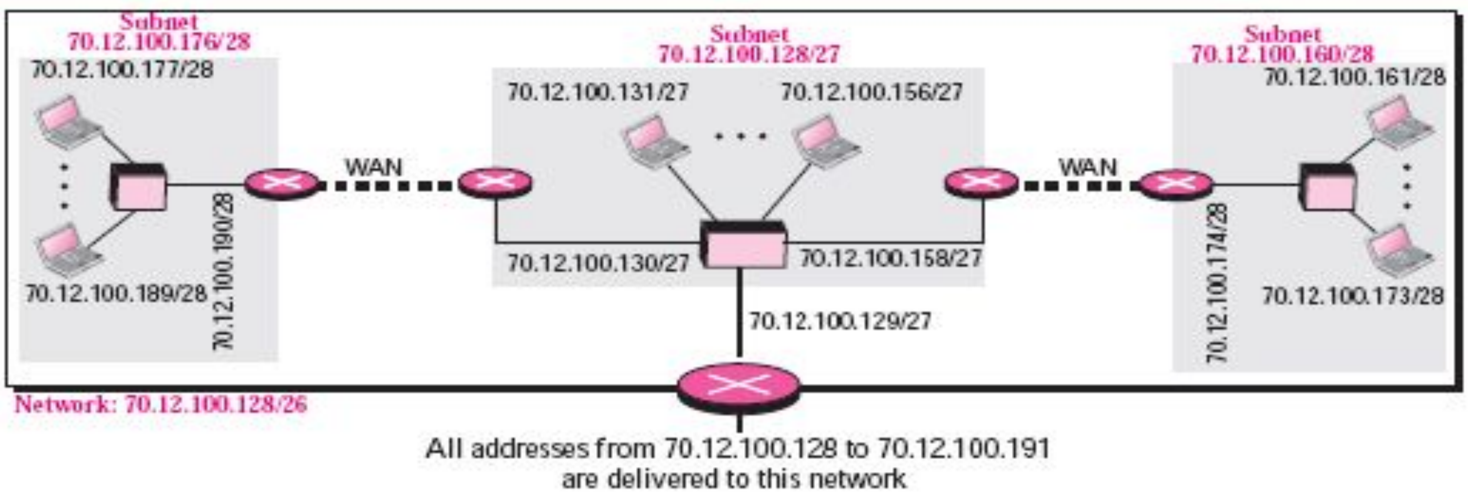
Central office $N_c = 32$	East office $N_e = 16$	West office $N_w = 16$

2. We can find the prefix length for each subnetwork:

$n_c = n + \log_2(64/32) = 27$	$n_e = n + \log_2(64/16) = 28$	$n_w = n + \log_2(64/16) = 28$

3. Figure 5.32 shows the configuration designed by the management. The Central office uses addresses 70.12.100.128/27 to 70.12.100.159/27. The company has used three of these addresses for the routers and has reserved the last address in the subblock. The East office uses the addresses 70.12.100.160/28 to 70.12.100.175/28. One of these addresses is used for the router and the company has reserved the last address in the subblock. The West office uses the addresses 70.12.100.160/28 to 70.12.100.175/28. One of these addresses is used for the router and the company has reserved the last address in the subblock. The company uses no address for the point-to-point connections in WANs.

Figure 5.32 *Example 14*



Address Aggregation

One of the advantages of CIDR architecture is **address aggregation.** ICANN assigns a large **block of addresses** to an ISP. Each ISP in turn divides its assigned block into smaller subblocks and grants the subblocks to its customers; many blocks of addresses are aggregated in one block and granted to one ISP.

Example 5.35

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:

- The first group has 64 customers; each needs approximately 256 addresses.
- The second group has 128 customers; each needs approximately 128 addresses.
- The third group has 128 customers; each needs approximately 64 addresses.

We design the subblocks and find out how many addresses are still available after these allocations.

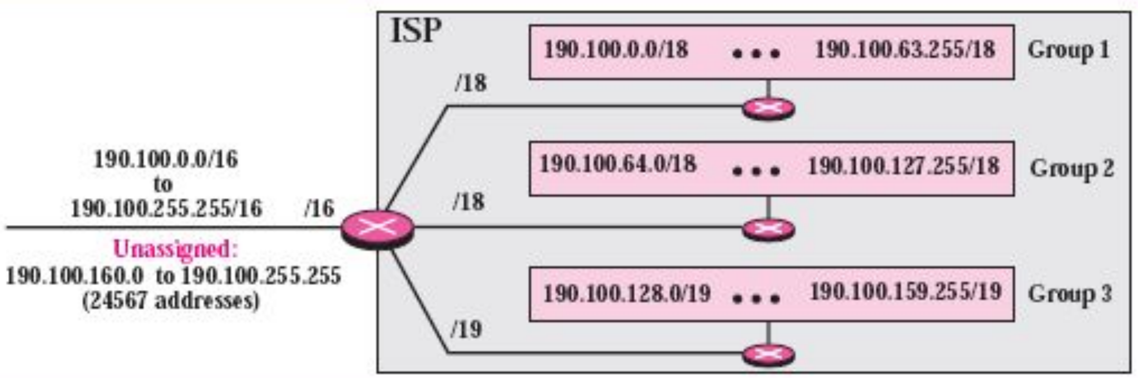
Solution

- Let us solve the problem in two steps. In the first step, we allocate a subblock of addresses to each group. The total number of addresses allocated to each group and the prefix length for each subblock can found as:

Group 1: $64 \times 256 = 16{,}384$	$n_1 = 16 + \log_2 (65536/16384) = 18$
Group 2: $128 \times 128 = 16{,}384$	$n_2 = 16 + \log_2 (65536/16384) = 18$
Group 3: $128 \times 64 = 8192$	$n_3 = 16 + \log_2 (65536/8192) = 19$

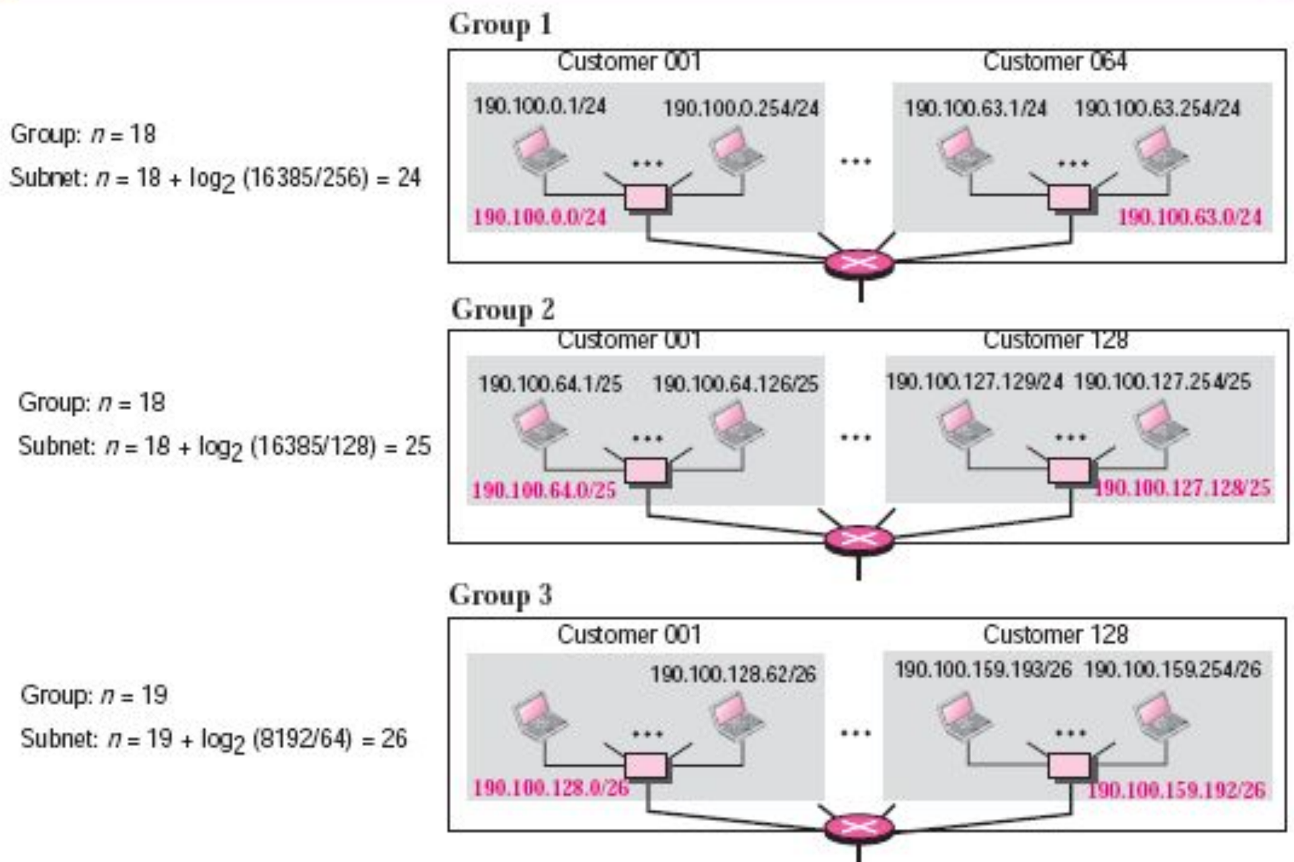
Figure 5.33 shows the design for the first hierarchical level.

Figure 5.33 *Solution to Example 5.35: first step*



Now we can think about each group. The prefix length changes for the networks in each group depending on the number of addresses used in each network. Figure 5.34 shows the second level of the hierarchy. Note that we have used the first address for each customer as the subnet address and have reserved the last address as a special address.

Figure 5.34 *Solution to Example 5.35: second step*



References

- TCP-IP Protocol Suit by Behrouz A. Forouzan
- CISCO online
- Microsoft windows LAN technology