IPv4 Addresses

1. INTRODUCTION

At the network layer, we need to uniquely identify each device on the Internet to allow global communication between all devices. The identifier used in the IP layer of the TCP/IP protocol suite to identify each device connected to the Internet is called the Internet address or **IP address.** An IPv4 address is a 32-bit address that *uniquely* and *universally* defines the connection of a host or a router to the Internet; an IP address is the address of the interface.

> An IPv4 address is 32 bits long.

IPv4 addresses are *unique*. They are unique in the sense that each address defines one, and only one, connection to the Internet. Two devices on the Internet can never have the same address at the same time. However, if a device has two connections to the Internet, via two networks, it has two IPv4 addresses. The IPv4 addresses are *universal* in the sense that the addressing system must be accepted by any host that wants to be connected to the Internet.

> The IPv4 addresses are unique and universal.

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 $\mathbf{2^b}$ because each bit can have two different values (0 or 1). IPv4 uses 32-bit addresses, which means that the address space is $\mathbf{2^{32}}$ 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.

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

Notation

There are three common notations to show an IPv4 address: binary notation (base 2), dotted-decimal notation (base 256), and hexadecimal notation (base 16). The most prevalent, however, is base 256.

Binary Notation: Base 2

In binary notation, an IPv4 address is displayed as 32 bits. To make the address more readable, one or more spaces is usually inserted between each octet (8 bits). Each octet is often referred to as a byte. So it is common to hear an IPv4 address referred to as a 32-bit address, a 4-octet address, or a 4-byte address. The following is an example of an IPv4 address in binary notation:

```
01110101 10010101 00011101 11101010
```

Dotted-Decimal Notation: Base 256

To make the IPv4 address more compact and easier to read, an IPv4 address is usually written in decimal form with a decimal point (dot) separating the bytes. This format is referred to as dotted-decimal notation. Figure 1 shows an IPv4 address in dotteddecimal notation. Note that because each byte (octet) is only 8 bits, each number in the dotted-decimal notation is between 0 and 255.

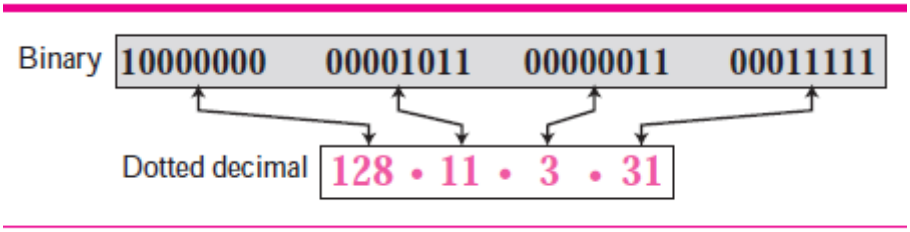


Figure 1 Dotted-decimal notation

Example 1

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 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 2

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

a. 111.56.45.78
b. 221.34.7.82
c. 241.8.56.12
d. 75.45.34.78

Solution

We replace each decimal number with its binary equivalent:

a. 01101111 00111000 00101101 01001110
b. 11011101 00100010 00000111 01010010
c. 11110001 00001000 00111000 00001100
d. 01001011 00101101 00100010 01001110

Example 3

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.

Hexadecimal Notation: Base 16

We sometimes see an IPv4 address in hexadecimal notation. Each hexadecimal digit is equivalent to four bits. This means that a 32-bit address has 8 hexadecimal digits. This notation is often used in network programming.

Example 4

Change the following IPv4 addresses from binary notation to hexadecimal notation.

a. 10000001 00001011 00001011 11101111

b. 11000001 10000011 00011011 11111111

Solution

We replace each group of 4 bits with its hexadecimal equivalent. Note that hexadecimal notation normally has no added spaces or dots; however, 0X (or 0x) is added at the beginning or the subscript 16 at the end to show that the number is in hexadecimal.

a. 0X810B0BEF or $810B0BEF_{16}$

b. 0XC1831BFF or $C1831BFF_{16}$

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 operations in the corresponding base (2, 256, or 16). Alternatively, we can covert the addresses to decimal values (base 10) and perform operations in this base.

Example 5

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.

$$\text{Number of addresses} = (0 \times 256^3 + 0 \times 256^2 + 3 \times 256^1 + 255 \times 256^0) + 1 = 1024$$

Example 6

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.

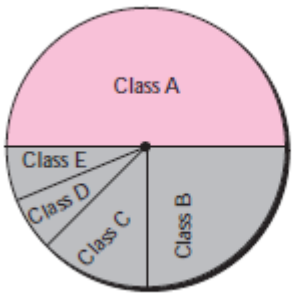
$$\text{Last address} = (14.11.45.96 + 0.0.0.31)_{256} = 14.11.45.127$$

2 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. Classless addressing is discussed in the next section.

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 2 shows the class occupation of the address space.



Class A: 2^{31} = 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%

Figure 2 Occupation of the address space

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

Recognizing Classes

We can find the class of an address when the address is given either in binary or dotteddecimal 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 3).

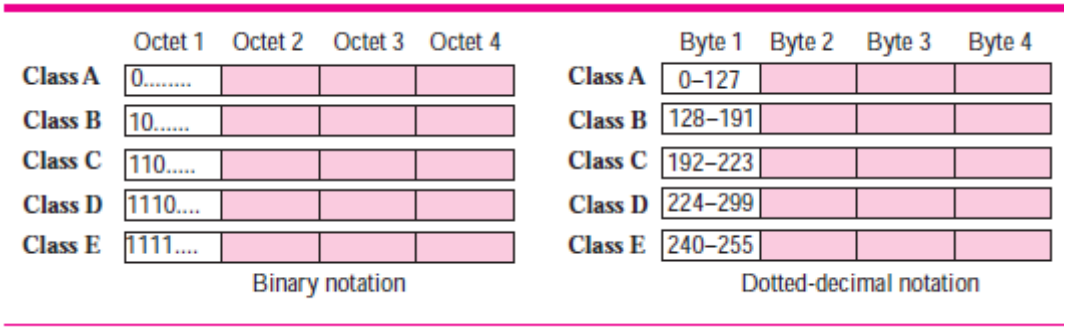


Figure 3 Finding the class of an address

Note that some special addresses fall in class A or E. We emphasize that these special addresses are exceptions to the classification; they are discussed later in the lecture. 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 4.

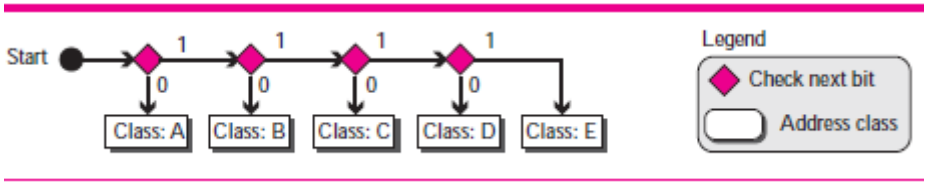


Figure 4 Finding the address class using continuous checking

Example 10

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

Solution

See the procedure in Figure 4.

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 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.

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. Figure 5 shows the netid and hostid bytes. Note that classes D and E are not divided into netid and hostid, for reasons that we will discuss later.

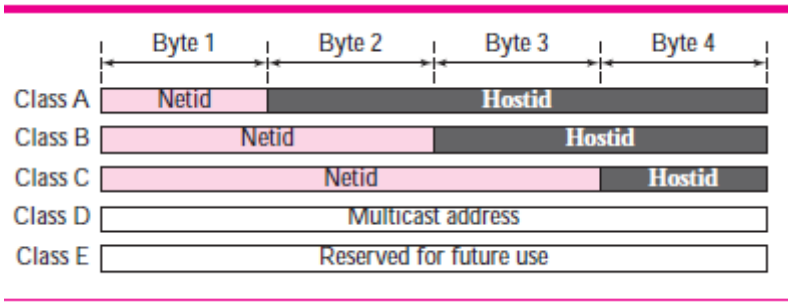


Figure 5 Netid and hosted

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

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 (the number is less because some blocks were reserved as special blocks). However, each block in this class contains 16,777,216 addresses, which means the organization should be a really large one to use all these addresses. Many addresses are wasted in this class. Figure 6 shows the block in class A.

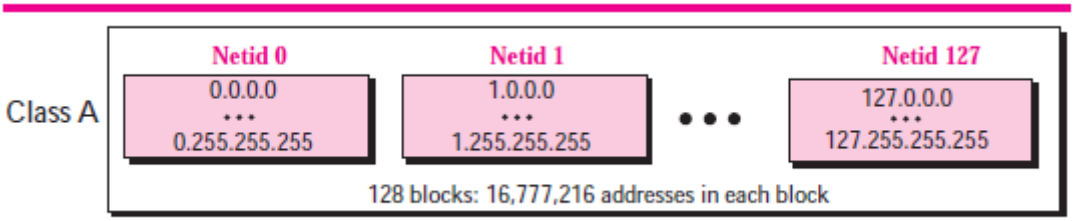


Figure 6 Blocks in class A

Millions of class A addresses are wasted.

Class B

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 (the number is less because some blocks were reserved as special blocks). However, each block in this class contains 65,536 addresses. Not so many organizations can use so many addresses. Many addresses are wasted in this class. Figure 7 shows the blocks in class B.

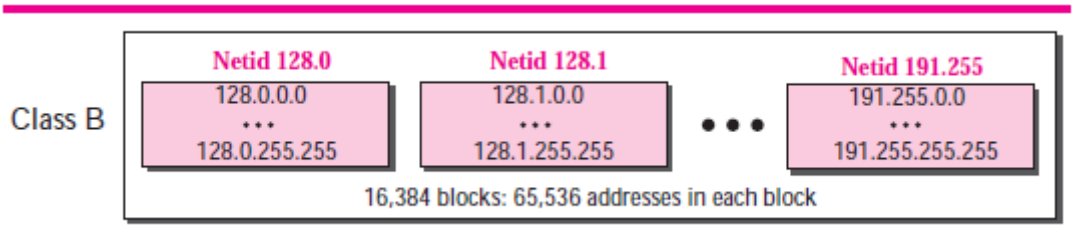


Figure 7 Blocks in class B

Many class B addresses are wasted.

Class C

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 (the number is less because some blocks were reserved as special blocks). Each block contains 256 addresses. However, not so many organizations were so small as to be satisfied with a class C block. Figure 8 shows the blocks in class C.

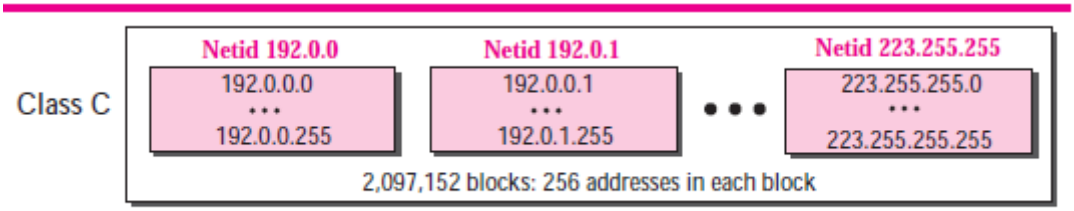


Figure 8 Blocks in class C

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

Class D

There is just one block of class D addresses. It is designed for multicasting, as we will see in a later section. Each address in this class is used to define one group of hosts on the Internet. When a group is assigned an address in this class, every host that is a member of this group will have a multicast address in addition to its normal (unicast) address. Figure 9 shows the block.

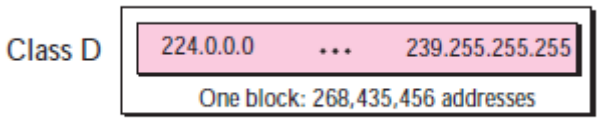


Figure 9 The single block in Class D

Class D addresses are made of one block, used for multicasting.

Class E

There is just one block of class E addresses. It was designed for use as reserved addresses, as shown in Figure 10.

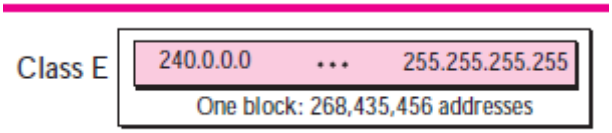


Figure 10 The single block in Class E

The only block of class E addresses was reserved for future purposes.