The Subnet Training Guide

A Step By Step Guide on Understanding and Solving Subnetting Problems

by Brendan Choi

easysubnet.com

v2.1

Chapter 1 Understanding IP addressing and Subnetting

1.0 Introduction

An IP address under Internet Protocol Version 4 is a number that is assigned to a device that allows other devices on the network to locate and reach it. Special devices called routers use IP addresses to find the best communication paths between network devices. Examples of devices that need IP addresses to communicate on a network include PC's, servers, printers, disk arrays, thin clients, routers, firewalls, switches, wireless access points, PDA's, mobile phones, IP phones, online gaming systems, and cable TV boxes. More and more devices in the world need IP addresses to communicate on the internet.

IPv4 number values range from **0 to 4,294,967,295** (we will learn why later). Millions of these numbers can be used as "IP addresses". For example, the value **1,789,532,491** is a valid IP number. But we don't write IP values in a flat decimal format like **347,895** or **1,789,532,491**. Instead, we write IP numbers in a special hierarchical "dotted decimal" format. For example, we write **1,789,532,491** as **106.170.25.75**. IP numbers are hierarchical because the IP addressing space (all the IP numbers) is also a ***network addressing space***. This will become clearer as we go along. The IP address tells us something about the network or subnet the IP address is in. To help us find the exact network, we need to know the ***network mask***, also called a ***subnet mask***. A subnet mask is a special IPv4 number, like **255.255.255.0**, that along with the IP address, gives us the exact network address. In other words, the IP address 106.170.25.75 tells us that the address is probably located under the 106.0.0.0 subnet, or close to it, but we still need to know the subnet mask to precisely locate the network. Routers use both the IP address and the subnet mask to precisely locate a device on the network or internet.

Why do addresses need to be hierarchical? Why do we need subnet masks? An address is useful when it not only identifies a device, but also includes location or area information. For example, a postman doesn't just need to know your street address to deliver the mail; he also needs to know your city and postal code. Together, the street address, city name and postal code form a hierarchical addressing system for the postal service. In order to give routers the location or area information, we include the subnet mask along with the IP address. An IP address with a subnet mask is the complete address that a router needs to determine a device's location, and make a path determination through the internet. Without a subnet mask, it's difficult or impossible to precisely locate a device on the internet just based on an IP address. Without a city name and postal code on the envelope, it's pretty hard for the postman to deliver a package. An IP address without a subnet mask is useless to a router. If there were no such things as subnet masks and subnets, then all the routers in the world would have to store the routes to hundreds of millions of ***unicast*** IP addresses, and this would consume an impossible amount of memory and CPU. This would be like if every single post office had to store all the street addresses in the country, instead of just the ones in their own individual city. Devices are assigned ***network address blocks*** (combinations of IP address numbers and subnet masks) by internet service providers, network administrators, and BootP and DHCP servers. Address blocks are given out to companies by internet authorities, such as the IANA (Internet Assigned Numbers Authority) and major telecom and internet service providers.

Computers calculate using mathematical logic and the Base 2 number system, also called the ***binary number*** system. The binary system uses 2 numerical symbols, 0 and 1, which can represent On and Off, or True and False. Each symbol, 0 or 1, is called a ***bit***. Devices read IP number values as binary numbers, and then group the binary numbers in **four groups of eight bits (for a total of 32 bits)**; each of these groups is called a ***byte*** or ***octet***. We then write down each of these four octets as Base 10 decimal numbers in a "dotted decimal" or "dotted quad" format. The decimal number system is the normal number system that we humans use everyday. So the IP number values are in binary form inside the computer, and we convert them to a dotted decimal form. This may all sound very complicated, but it is infact what makes IP addresses easy to read.

Here is an example of four different ways to write down the same IP number value:

IP address value:	**1789532491**
Base 2 binary number:	**1101010101010100001100101001011**
Binary number with octets:	**01101010.10101010.00011001.01001011**
Dotted decimal format:	**106.170.25.75**

Notice we create the octets from RIGHT to LEFT, so we add an extra zero "0" at the LEFT end to make it clear we want to see them as octets. The numbers **1789532491** and **106.170.25.75** are actually the same IP address value, just written in different ways. We use the dotted decimal format because it's easier to read, and because it is hierarchical, it tells us something about the network or subnet. We will learn later how to convert between all these formats.

IPv4 addresses and subnet masks are all **32-bit numbers**. That means that IP addresses and subnet masks can be written with a maximum of up to 32 bits, using the numerical symbols 0 and 1. That is why 4,294,967,295 (which is 2^{32} - 1) is the maximum value for an IP address (actually this maximum value is really a mask, and not a usable IP address). To write 4,294,967,295 in binary format, we write 32 one's.

11111111111111111111111111111111 = 4,294,967,295

In octet and dotted decimal format, it looks like:

11111111.11111111.11111111.11111111 = 255.255.255.255

We will learn more about decimal numbers, binary numbers and and how to convert between them later in this chapter.

Internet Protocol Version 6 uses 128-bit numbers, and will be used more and more as usable IPv4 addresses are exhausted. This guide only covers Internet Protocol Version 4.

1.1 Understanding Numbers, IP addressing and Binary numbers

Our normal decimal, or Base 10, numbering system uses 10 symbols or numbers to represent all values.

0 1 2 3 4 5 6 7 8 9

The Base 16, or hexadecimal system, uses 16 symbols to represent all values.

0 1 2 3 4 5 6 7 8 9 A B C D E F

The Base 2, or binary number system, uses 2 numbers to represent all values.

0 1

For example, the value of "two hundred and thirty one" can be written in these different ways…

Decimal:	231
Hexadecimal:	E7
Binary:	11100111

Internet Protocol Version 4 is a network addressing system composed of 32-bit numbers. An IP address is just a 32-bit number. That means an address value has up to 32 bit values. In binary numbers, they are written like this, from lowest to highest, with values shown…

```
00000000000000000000000000000000 = 0            (lowest value)
00000000000000000000000000000001 = 1
00000000000000000000000000000010 = 2
00000000000000000000000000000011 = 3
00000000000000000000000000000100 = 4
.
.
11111111111111111111111111111101 = 4294967293
11111111111111111111111111111110 = 4294967294
11111111111111111111111111111111 = 4294967295     (maximum value)
```

We will discuss a little later how to convert these binary numbers to decimal values.

Example of IP address: **10111100001100110100111100000111**

Mathematically, in a 32-bit number, there are 2^{32} combinations of values, so there are 2^{32} = 4,294,967,296 unique IP address numbers possible. You can actually see in the example above, the binary values range from 0 to 4294967295, which is exactly 4,294,967,296 unique values. Although there seems to be over 4 billion IP addresses possible, we will learn later that there are many restrictions as to which IP addresses can be used.

Why does a 32-bit number have 2^{32} combinations? Let's look at some examples of 1-bit, 2-bit and 3-bit numbers.

A 1-bit number has 2^1 = 2 combination of values:

0
1

A 2-bit number has 2^2 = 4 combination of values:

00
01
10
11

A 3-bit number has 2^3 = 8 combination of values:

000
001
010
011
100
101
110
111

In general, an N-bit number has 2^N combination of values. So a 32-bit number has 2^{32} = 4,294,967,296 combination of values, ranging from 0 to 4,294,967,295.

You can place an IP address on the network interface of computers and routers to give them an identity on a network. Some of the bits are "turned on" (the 1's), and some of them are "turned off" (the 0's). We normally divide the 32 bits into 4 parts with "." dots. Each part is called an **octet** (means 8 bits). We call this the "dotted quad" format.

Example of IP address in dotted quad format: **10111100.001100110.10011110.00001111**

We do this to make IP addresses easier to read and organize into classes (more on this later). This is why the IP addressing space is actually a hierarchical network addressing space. Binary numbers are a Base-2 number system. Our regular number system is Decimal, also called Base-10. Since we read decimal easier, IP addresses are usually written in a dotted decimal format. The binary values in each octet are converted to decimal.

Example of IP address in dotted decimal format: **173.42.219.117**

Please note that 172.42.219.117 is NOT the same as the number 17342219117! These two numbers have absolutely nothing to do with each other. We will illustrate this a little more later.

The certification exams will ask you to convert binary numbers to decimal numbers, and vice versa. To convert binary numbers to decimal is easy. Since binary is Base-2, each bit is just a power of 2, and we just add up the powers of 2. How is this possible?

As we just discussed, the Binary number system only uses two numerical symbols, 0 and 1, to represent values.

```
0 = zero
1 = one
```

But how do we represent higher values, like two, three, four or eight thousand with just two symbols? We use the same symbols 0 and 1, and as we keep increasing one by one, we write the numbers from RIGHT to LEFT. We add in exactly the same way with decimal numbers.

```
           1
add        1
------------
          10 = two

          10
add       01
-------------
          11 = three

          11
add       01
-------------
         100 = four
```

As you can see, we are just adding each number by one and creating the next new number. Although we add the numbers from RIGHT to LEFT, we could as well write numbers from left to right and write them differently. How we write numbers is arbitrary and has nothing to do with mathematical laws.

```
         100
add      001
--------------
         101 = five
```

```
      101
add   001
--------------
      110 = six

      110
add   001
--------------
      111 = seven

      111
add   001
--------------
     1000 = eight
```

We can keep adding these binary numbers to infinity to write all numbers. Here are some of the numbers, along with the exponential values (powers of 2).

0 = zero
1 = one = 2^0
10 = two = 2^1
11 = three
100 = four = 2^2
101 = five
110 = six
111 = seven
1000 = eight = 2^3
1001 = nine
.
.
1111 = fifteen
10000 = sixteen = 2^4
10001 = seventeen
.
.
11111 = thirty-one
100000 = thirty-two = 2^5
100001 = thirty-three
.
.
.
.
111111111 = five hundred and eleven
1000000000 = five hundred and twelve = 2^9
1000000001 = five hundred and thirteen
1000000010 = five hundred and fourteen
.
.

Notice that in the Binary system, the exponentials (powers of 2) are reached a lot sooner, at 1, 4, 8, 16, 32 and so on. In the Decimal system, we reach the exponentials (powers of 10) at 1, 10, 100, 1000 and so on.

0 = zero

1 = one = 10^0
2 = two
3 = three
.
.
9 = nine
10 = ten = 10^1
11= eleven
12 = twelve
.
.
99 = ninety-nine
100 = one hundred = 10^2
101 = one hundred and one
.
.
.
.
998 = nine hundred ninety-eight
999 = nine hundred ninety-nine
1000 = one thousand = 10^3
1001 = one thousand and one
.
.

When comparing the Decimal and Binary systems, we can see that the Decimal system uses more numerical symbols and the numbers increase their places from right to left a lot slower. For Binary numbers, the numbers increase their places from right to left a lot faster.

For us, the Decimal system is a lot more easier to read and write. To write the value "three thousand five hundred and seventy one" in decimal, we only need to write a few numbers, but we need to write a lot more numbers in binary.

Decimal = 3571
Binary = 110111110011

But computers calculate from electrical signals, that can be either "on" or "off" or positive or negative. So it's more efficient for computers to use binary numbers to calculate.

In IP addressing and subnetting, we are dealing with octets, each with 8 possible bit places. So we need to learn how to convert octets. Let's convert an octet.

Example 1.1:
Example of binary octet: 10111100

$$(1\times2^7) + (0\times2^6) + (1\times2^5) + (1\times2^4) + (1\times2^3) + (1\times2^2) + (0\times2^1) + (0\times2^0)$$

Like in our decimal number system, we compute from RIGHT to LEFT. Notice that the first power of 2, the right most, is 2^0, and the last power of 2, the left most, is 2^7. Going from 0 to 7 is 8 bit places. And 2^0 is just 1, which makes sense since the first place in a number system is the 1's place.

128 + 0 + 32 + 16 + 8 + 4 + 0 + 0 = 188

Therefore, 10111100 = 188. Let's look at some Base-10 decimal numbers, and see how we really do the same thing.

Example 1.2:
Example of decimal number: 74905

This number can be written as: $(7\times10^4) + (4\times10^{3)} + (9\times10^2) + (0\times10^1) + (5\times10^0)$

70000 + 4000 + 900 + 0 + 5 = 74905

Example 1.3:
Example of decimal number: 1842

This number can be written as: $(1\times10^{3)} + (8\times10^2) + (4\times10^1) + (2\times10^0)$

1000 + 800 + 40 + 2 = 1842

As you can see, the numbers written in our normal Base-10 decimal system are just powers of 10. In the Base-2 binary system, numbers are in powers of 2.

Example 1.4:
Example of binary octet: 01110101

$(0\times2^7) + (1\times2^6) + (1\times2^5) + (1\times2^4) + (0\times2^3) + (1\times2^2) + (0\times2^1) + (1\times2^0)$

0 + 64 + 32 + 16 + 0 + 4 + 0 + 1 = 117

Therefore, 01110101 = 117.

Example 1.5:
Example of binary octet: 11100011

$(1\times2^7) + (1\times2^6) + (1\times2^5) + (0\times2^4) + (0\times2^3) + (0\times2^2) + (1\times2^1) + (1\times2^0)$

128 + 64 + 32 + 0 + 0 + 0 + 2 + 1 = 227

Therefore, 11100011= 227.

Example 1.6:
Example of binary octet: 00111100

$(0\times2^7) + (0\times2^6) + (1\times2^5) + (1\times2^4) + (1\times2^3) + (1\times2^2) + (0\times2^1) + (0\times2^0)$

0 + 0 + 32 + 16 + 8 + 4 + 0 + 0 = 60

Therefore, 00111100 = 60.

Notice that the lowest an octet can be is 00000000, which is just 0. And the highest is 11111111, which is just 255.

Example 1.7:
Highest octet: 11111111

$(1\times2^7) + (1\times2^6) + (1\times2^5) + (1\times2^4) + (1\times2^3) + (1\times2^2) + (1\times2^1) + (1\times2^0)$

128 + 64 + 32 + 16 + 8 + 4 + 2 + 1 = 255

Here are examples of larger binary numbers that go beyond 8 bits…

Example 1.8:
Example of binary value: 1011001101

$(1\times2^9) + (0\times2^8) + (1\times2^7) + (1\times2^6) + (0\times2^5) + (0\times2^4) + (1\times2^3) + (1\times2^2) + (0\times2^1) + (1\times2^0)$

512 + 0 + 128 + 64 + 0 + 0 + 8 + 4 + 0 + 1 = 717

Therefore, 1011001101= 717.

Example 1.9:
Example of binary value: 01101000111011

$(0\times2^{13}) + (1\times2^{12}) + (1\times2^{11}) + (0\times2^{10}) + (1\times2^9) + (0\times2^8) + (0\times2^7) + (0\times2^6) + (1\times2^5) + (1\times2^4) +$
$(1\times2^3) + (0\times2^2) + (1\times2^1) + (1\times2^0)$

0 + 4096 + 2048 + 512 + 0 + 0 + 0 + 32 + 16 + 8 + 0 + 2 + 1 = 6715

Therefore, 01101000111011= 6715.

Notice that if the right most binary digit is 1, the decimal number has to be odd. If the right most digit is 0, the decimal number has to be even. Remember this on the certification exams. They will give you a decimal number, and ask you to find the binary form. Just pick the binary numbers that have to be even or odd, and use process of elimination. Often times, you can find the correct binary number without having to convert the entire number. The same reasoning also works to convert binary to decimal numbers. For example, make sure the decimal number is even if the binary number has a 0 at the end.

By itself, an IP address doesn't mean much. It doesn't tell you the exact network where the IP address is located in. We need to associate a network to an IP address, just like a city name and ZIP code has to be associated with a street address to locate a house.

We associate an IP address with a Subnet Mask. We use a Subnet Mask to identify the Subnet or Network Address. A Subnet Mask is just a 32-bit number that starts from the left with contiguous 1's, and ends on the right with contiguous 0's.

Example of Subnet Mask: **11111111.11111111.11111111.00000000**

Some people also call the subnet mask the "network mask" or "netmask"---they mean the same thing. We will also use the terms “subnet address” and “network address” interchangeably. Notice that there is no such thing as a subnet mask with non-contiguous 1's or 0's, or 0's on the left and 1's on the right. You won't see any subnet masks that look like these…

```
11111111.11111111.00000000.11111111
11110000.11111111.11111111.00000000
00000111.11111111.11110000.00001111
00000000.00000000.11111111.11111111
00000000.11111111.11111111.11111111
```

An IP address has to be associated with a Subnet Mask to have a network. To find the Network Address, we just apply what's called a "**Boolean Logical AND operation**" between the IP address and the Subnet Mask.

```
IP address =   11001110.01110110.00110001.11110011
Subnet Mask =  11111111.11111111.11111111.00000000
               -----------------------------------------------------
Network =      11001110.01110110.00110001.00000000
```

Notice that the last octet in the network address are all 0's. The last octet in this example contains the "**host bits**". As we will learn later on in our examples…

Network addresses are all the IP addresses with all host bits set to 0's.

In a Boolean Logical AND operation, both bits have to be 1 for the output to be 1 (TRUE), otherwise the output is 0 (FALSE).

1 + 1 = 1	TRUE + TRUE = TRUE
1 + 0 = 0	TRUE + FALSE = FALSE
0 + 1 = 0	FALSE + TRUE = FALSE
0 + 0 = 0	FALSE + FALSE = FALSE

Example 1.10:

```
Octet =     11001110
Mask =      11111000
            -------------
Output =    11001000

Octet =     01100111
Mask =      11100000
            -------------
Output =    01100000
```

You can think of a Subnet Mask as a device that takes an IP addresses for input, and gives a Network Address as the output.

IP Address → Subnet Mask → Network Address

A Subnet Mask is also like a filter. The contiguous 1's in the subnet mask filter in the bits it needs to create a network address. The 0's in the subnet mask filter out the "host bits". The resulting network address is just a set of "network bits" (more on this later).

Since we don't read binary very well, we have to convert IP addresses and subnet masks into regular decimal numbers. The easiest thing to do is just convert each octet into a decimal number.

Example 1.11:
Suppose we have an IP address 10011110.00011111.01110110.00011110. We convert each octet…

```
10011110 = 158
00011111 = 31
01110110 = 118
00011110 = 30
```

So, the IP address in decimal is 158.31.118.30. We use the Logical AND operation to find the network address, and convert the subnet mask and network address to decimal.

```
IP address = 158.31.118.30 =          10011110.00011111.01110110.00011110
Subnet mask = 255.255.0.0 =           11111111.11111111.00000000.00000000
-----------------------------------------------------------------------------------------------
Network address = 158.31.0.0 =        10011110.00011111.00000000.00000000
```

Example 1.12:

```
IP address = 214.157.88.203 =         11010110.10011101.01011000.11001011
Subnet mask = 255.255.255.0 =         11111111.11111111.11111111.00000000
----------------------------------------------------------------------------------------------
Network address = 214.157.88.0 =      11010110.10011101.01011000.00000000
```

Example 1.13:

```
IP address = 17.143.229.15 =          00010001.10001111.11100101.00001111
Subnet mask = 255.0.0.0 =             11111111.00000000.00000000.00000000
----------------------------------------------------------------------------------------------
Network address = 17.0.0.0 =          00010001.00000000.00000000.00000000
```

Since the subnet masks have octets with all 0's, doing the Boolean Logical AND operation with the zero octets is very easy. In these examples, the Network Address is just the IP address with the last octets (host bits) as all 0's.

By the way, we normally convert the binary IP address to decimal form by octet. We're not converting the entire 32-bit number. In other words, we convert…

10011110.00011111.01110110.00011110 into 158.31.118.30

…and not…

10011110000111110111011000011110 into 2652861982

We use the dotted decimal format (4 octets divided by a dot "**.**") because it's easier to read, and because it is hierarchical, i.e. it allows us to organize IP addresses into network classes (Class A, B, C, D, and E).

For example, if you have a computer or broadband router at home with an IP address of 192.168.1.1, and you were running a web server on it, you could enter http://192.168.1.1 on your web browser to get a web page. But you could also enter http://3232235777 and get the same web page. Why? Because 192.168.1.1 is 11000000.10101000.00000001.00000001 in binary form, but 11000000.10101000.00000001.00000001 is really 11000000101010000000000100000001, which is a very large number, equal to 3,232,235,777. Again, we mainly use the dotted quad format because it's easier to read, and because we can organize the IP addresses into hierarchical classes. It's also easier to convert each octet to decimal than to convert the entire binary number to decimal.

So, is 3232235777 an IP address also? Well, sort of. It's a number in the IP address range, but it doesn't tell us what the network might be (it's not hierarchical). A computer web browser may recognize it, but not humans. However, you can convert 3232235777 to a binary number, which you convert to an IP address.

3232235777 → 11000000101010000000000100000001 → 11000000.10101000.00000001.00000001 → 192.168.1.1

That is why, in the Introduction, we said 1789532491 is the same thing as 106.170.25.75.

1789532491 → 1101010101010100001100101001011 → 01101010.10101010.00011001.01001011 → 106.170.25.75

One last thing. No one writes, and no computer or router will recognize, an IP address in dotted quad numbers without the dots ".". **In other words, the IP address 192.168.1.1 has absolutely nothing to do with the number "19216811"!** If you type http://19216811 into a web browser, you will probably get nothing. That number would translate to an IP address of 1.37.57.171. Why?

19216811 → 1001001010011100110101011 → 00000001.00100101.00111001.10101011 → 1.37.57.171

On certification exams, you also need to know how to convert decimal numbers to binary. To do this, remember that a binary number is a sum of bits representing powers of 2, from 2^0 to 2^7. If the bit is 1, that place is added. If the bit is 0, that place is not added. In other words, we can write a binary number as some combination of the following powers of 2…

$2^7 \mid 2^6 \mid 2^5 \mid 2^4 \mid 2^3 \mid 2^2 \mid 2^1 \mid 2^0$

Since we are working with IP addresses, the exams will usually ask you to convert a decimal number between 0 and 255. We know that for this range of numbers, you only need to look at 8 bit places in the binary form, from 2^0 to 2^7. We might as well get familiar with the first 8 bit places in decimal form…

$2^7 \mid 2^6 \mid 2^5 \mid 2^4 \mid 2^3 \mid 2^2 \mid 2^1 \mid 2^0 \rightarrow 128 \mid 64 \mid 32 \mid 16 \mid 8 \mid 4 \mid 2 \mid 1$

When converting decimal binary, we want to know which bit is turned on, and which is turned off. As we will see, when you convert a decimal number to binary, you're really doing all the math in decimal form.

Example 1.14: Convert 235 to binary

We start with 235, and divide by each bit place and get a remainder. If the remainder is less than the next bit place, we just add a 0 at that bit place. We continue until the remainder is 0.

235 → 1×128 → remainder 107
107 → 1×64 → remainder 43
43 → 1×32 → remainder 11
11 → 0×16
11 → 1×8 → remainder 3
3 → 0×4
3 → 1×2 → remainder 1
1 → 1×1 → remainder 0

The binary number for 235 is 11101011.

Example 1.15: Convert 109 to binary

We start with 109, and find we can't divide by 128, so we skip it and continue to 64.

109 → 0×128
109 → 1×64 → remainder 45
45 → 1×32 → remainder 13
13 → 0×16
13 → 1×8 → remainder 5
5 → 1×4 → remainder 1
1 → 0×2
1 → 1×1 → remainder 0

The binary number for 109 is 01101101. Since the left most bit is 0, you might see this number written as 1101101 also. Remember we really count from RIGHT to LEFT.

Let's convert a number larger than 255.

Example 1.16: Convert 457 to binary

For this number, we have to go to 9 bit places, so the bit places go from 2^0 to 2^8 .

$2^8 | 2^7 | 2^6 | 2^5 | 2^4 | 2^3 | 2^2 | 2^1 | 2^0$ → 256 | 128 | 64 | 32 | 16 | 8 | 4 | 2 | 1

457 → 1 × 256 → remainder 201
201 → 1× 128 → remainder 73
73 → 1×64 → remainder 9
9 → 0×32
9 → 0×16
9 → 1×8 → remainder 1
1 → 0×4
1 → 0×2
1 → 1×1 → remainder 0

So the binary number for 457 is 111001001.

Example 1.17: Convert 248 to binary

248 → 1×128 → remainder 120
120 → 1×64 → remainder 56
56 → 1×32 → remainder 24
24 → 1×16 → remainder 8
8 → 1×8 → remainder 0
0×4
0×2
0×1

The binary number for 248 is 11111000.

Example 1.18: Convert 98 to binary

98 → 0×128
98 → 1×64 → remainder 34
34 → 1×32 → remainder 2
2 → 0×16
2 → 0×8
2 → 0×4
2 → 1×2 → remainder 0
0×1

The binary number for 98 is 01100010. Since the left most bit is 0, you might see this number written as 1100010 also. Remember we really count from RIGHT to LEFT.

At the end of this chapter are exercises to convert binary numbers to decimal and decimal to binary.

1.2 IP Address Classes and Subnet Masks

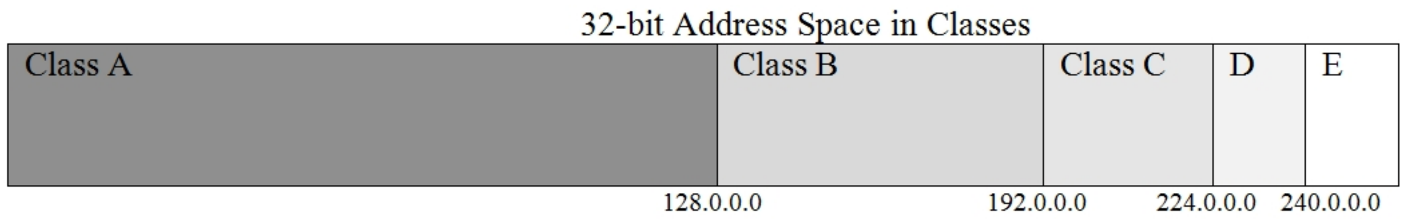
The Internet Authorities originally organized the IP address space into 5 classes. The classes are defined by how many 1's in the left most bits.

Class A	00000000.00000000.00000000.00000000 to 01111111.11111111.11111111.11111111
Class B	10000000.00000000.00000000.00000000 to 10111111.11111111.11111111.11111111
Class C	11000000.00000000.00000000.00000000 to 11011111.11111111.11111111.11111111
Class D	11100000.00000000.00000000.00000000 to 11101111.11111111.11111111.11111111
Class E	11110000.00000000.00000000.00000000 to 11111111.11111111.11111111.11111111

In decimal form…

Class A	0.0.0.0 to 127.255.255.255
Class B	128.0.0.0 to 191.255.255.255
Class C	192.0.0.0 to 223.255.255.255
Class D	224.0.0.0 to 239.255.255.255
Class E	240.0.0.0 to 255.255.255.255

If we were to visualize the allocation of the addresses into the classes, it would look like this:



The Internet Authorities originally thought that most of the addresses would be used in a few very large subnets. Classes are not used today on the real internet. Classless Interdomain Routing, or CIDR, will be discussed in Chapter 3. But an understanding of IP address classes is still very useful when learning about subnetting.

Some things to remember…
1. The 127.0.0.0 network is the loopback network and not used on the internet.
2. IP's that have the last octet as 255 are not assigned to hosts; 0.0.0.0 and 255.255.255.255 are not assigned to anything.
3. Class D is used for multicast groups. For example, routing protocols can send an update to a group of routers instead of a unicast update to each router. Class E is used for research and is not used on the internet.

There are also a range of IP addresses you can only used privately, not on the internet. These are the **RFC 1918** private IP's:

Class A	10.0.0.0 to 10.255.255.255
Class B	172.16.0.0 to 172.16.255.255
Class C	192.168.0.0 to 192.168.255.255

IP addresses that are not in these ranges are public IP's, owned by individuals, companies or internet authorities. You will often see private IP's used in books since they can be used internally by any organization. Hosts with private IP addresses can still get on the internet by connecting to a firewall or router that uses Network Address Translation (NAT). For example, a firewall can use NAT to allow a group of internal hosts using private IP's to share a pool of public IP's to get on the internet.

The Class A, B and C addresses each have their own Default Subnet Masks.

Class A default subnet mask = 11111111.00000000.00000000.00000000 = 255.0.0.0
Class B default subnet mask = 11111111.11111111.00000000.00000000 = 255.255.0.0
Class C default subnet mask = 11111111.11111111.11111111.00000000 = 255.255.255.0

Class A has a 8 bit subnet mask, Class B has a 16 bit mask, and Class C has a 24 bit mask. Another way to write subnet masks, is to use a shorthand called CIDR or slash "/" notation. In CIDR notation, we just indicate the number of bits turned on (the 1's) in the subnet mask.

Class A = /8 = 11111111.00000000.00000000.00000000 = 255.0.0.0
Class B = /16 = 11111111.11111111.00000000.00000000 = 255.255.0.0
Class C = /24 = 11111111.11111111.11111111.00000000 = 255.255.255.0

When we learn subnetting later in this chapter, we will see other kinds of subnet masks…

/28 = 11111111.11111111.11111111.11110000 = 255.255.255.240
/19 = 11111111.11111111.11100000.00000000 = 255.255.224.0
/13 = 11111111.11111000.00000000.00000000 = 255.248.0.0

As stated above, Class A, B and C have these IP address ranges…

Class A	00000000.00000000.00000000.00000000 to 01111111.11111111.11111111.11111111
Class B	10000000.00000000.00000000.00000000 to 10111111.11111111.11111111.11111111
Class C	11000000.00000000.00000000.00000000 to 11011111.11111111.11111111.11111111

Using the Boolean Logical AND operation, the IP address and the subnet mask determines the network address.

Example 1.19: 10.45.211.7/8

The /8 means the subnet mask is 255.0.0.0. Doing the Logical AND operation, we can see that the network address is 10.0.0.0.

Example 1.20: 164.25.234.2/16

The /16 means the subnet mask is 255.255.0.0. Doing the Logical AND operation, we can see that the network address is 164.25.0.0.

Example 1.21: 209.142.17.55/24

The /24 means the subnet mask is 255.255.255.0. Doing the Logical AND operation, we can see that the network address is 209.142.17.0.

Each subnet mask octet has 8 bit places, there are 2^8 = 256 combination of numbers in an octet. The number range is 0 to 255. Since Class A IP addresses requires the first bit to be 0, and the subnet mask is 8 bits long, Class A has 2^7 unique combinations of network addresses. Class B requires the first 2 bits to be 10, and since Class B's subnet mask is 16 bits long, it has 2^{14} combinations of networks. Class C requires the first 3 bits to be 110, and since Class C's subnet mask is 24 bits long, it has 2^{21} combination of networks.

Class A	2^7 = 128 networks
Class B	2^{14} = 16384 networks
Class C	2^{21} = 2097152 networks

The number of hosts per subnet is equal to the number of bits in the subnet masks that are 0's.

Class A	24 bits are 0's → 2^{24} = 16777216 addresses/subnet
Class B	16 bits are 0's → 2^{16} = 65536 addresses/subnet
Class C	8 bits are 0's → 2^8 = 256 addresses/subnet

As you can see, as the subnet mask increases, you gain more subnets, but get less hosts per subnet. The smaller the subnet mask, the less subnets you get, but the more hosts per subnet you get. Another way to say this is that Class A networks tend to be large, and Class C networks tend to be small. Of course, not all the IP's can be used for hosts, like 127.0.0.0 network, 0.0.0.0, 255.255.255.255, or any IP that ends with the 255 octet. Also, you can't use the first and last IP address in each subnet. The first IP address in a subnet is the Network Address of the subnet, and the last IP address in each subnet is the Broadcast Address. The figures we compute here present a baseline value of addresses/subnet.

The Network address identifies the network or subnet. The Broadcast Address identifies all hosts on the subnet. When a packet goes to the broadcast address, it goes to all hosts. You can't use network or broadcast addresses on the interfaces for hosts or routers.

Network = 192.168.1.0/24
Network Address = 192.168.1.0
Broadcast Address = 192.168.1.255

Network = 172.16.0.0/16
Network Address = 172.16.0.0
Broadcast Address = 192.168.255.255

Network = 10.0.0.0/8

Network Address = 10.0.0.0
Broadcast Address = 10.255.255.255

So, for each subnet, we subtract 2 IP addresses to get the usable number of IP addresses per subnet.

Class A	24 bits are 0's → 2^{24} - 2 = 16777214 hosts/network
Class B	16 bits are 0's → 2^{16} -2 = 65534 hosts/network
Class C	8 bits are 0's → 2^{8} -2 = 254 hosts/network

Earlier in this chapter, it was stated that the IP 32-bit addressing space has 2^{32} = 4,294,967,296 unique values. We've computed a baseline of values for each class of addresses:

Class A	128 networks, 16777216 addresses/network
Class B	16384 networks, 65536 addresses/network
Class C	2097152 networks, 256 addresses/network

If we add these 3 classes up, we find the number of IP addresses is 2147483648 + 1073741824 + 536870912 = 3,758,096,384. That means there are at least 4,294,967,296 - 3,758,096,384 = 536,870,912 addresses **not usable**. Considering that many addresses are used for subnet network addresses, broadcast addresses, private **RFC 1918** addresses, and are unusable due to various rules, the number of actual usable addresses for hosts on the Internet is much less than 3,758,096,384 . By using IP address Classes A, B and C, we are in effect, wasting IP addresses. In Chapter 3, we will learn how supernetting and classless addressing can reclaim these "lost" IP addresses.

Here are lists of how IP address blocks are allocated on the real internet: http://www.iana.org/assignments/ipv4-address-space , http://en.wikipedia.org/wiki/List_of_assigned_/8_IP_address_blocks .

1.3 Understanding Subnetting

When we subnet, we basically chop up a default Class A, B or C network (also called "major" network) into smaller networks (subnets). We get more subnets, but each subnet has smaller number of hosts per subnet. We do this by applying a new subnet mask that divides the network into smaller subnets.

For example, a Class C network, by default, has 254 hosts/subnet. But suppose if you have several LAN's, each with only about 30 hosts. How can you use that single Class C network for several of these LAN's? The answer is that we subnet that Class C network with a new subnet mask.

Example 1.22:
Let's say we have Class C major network 192.168.1.0/24, so our subnet mask is 255.255.255.0. And we need 5 networks that can hold 30 hosts each. The 192.168.1.0/24 network can hold 254 hosts. What subnet mask do we need to create several more smaller subnets? Let's try a mask with 3 extra bits…

11000000.10101000.00000001.00000000 = Network
11111111.11111111.11111111.00000000 = Default Subnet Mask = /24
11111111.11111111.11111111.11100000 = New Subnet Mask = /27

Notice that the new default mask we're trying has 3 extra bits, so it's a /27 subnet mask. That means there is an additional 2^{3} = 8 combinations of networks. These 3 extra bits are called the "**Subnet Bits" or "Network Bits"**. There are now only 5 zero bits left. That means there are now 2^{5} -2 = 30 hosts per subnet. So, with the new subnet mask, we get 8 new subnets, each with 30 hosts per subnet. The 5 zero bits are called the **Host Bits**.

Network = 192.168.1.0/27 8 new subnets, 30 hosts/subnet
New subnet mask = 255.255.255.224

We can use 5 of these 8 subnets, and leave 3 of them for future use.

An easy way to count the number of subnets gained is to calculate 2^m, where "**m**" is the number of extra 1's in the new subnet mask (the subnet bits). The number of hosts/subnet is $2^n - 2$, where "**n**" is the number of 0's left in the new subnet mask (the host bits).

2^m = number of subnets
$2^n - 2$ = number of usable hosts per subnet

As you can see, increasing the subnet mask bit by bit increases the number of subnets by a factor of 2, and halves the number of addresses per subnet.

What do the new 8 subnets look like? The new subnet mask octet is 11100000, which has 3 contiguous 1's. If you run a Logical AND operation between every possible IP address in the 192.168.1.0/24 block and the new subnet mask, you will get all the possible subnet network addresses. **The network addresses are related to all possible values in the extra bits (shaded portion).**

```
11000000.10101000.00000001.xxxxxxxx   ← all possible IP addresses in 192.168.1.0/24
11111111.11111111.11111111.11100000   ← new subnet mask /27
-----------------------------------------------
11000000.10101000.00000001.xxx00000   ← all possible network addresses
```

```
00000000 = 0
00100000 = 32
01000000 = 64
01100000 = 96
10000000 = 128
10100000 = 160
11000000 = 192
11100000 = 224
```

So, the 8 new /27 subnets look like these…

192.168.1.0	(zero subnet)
192.168.1.32	
192.168.1.64	
192.168.1.96	
192.168.1.128	
192.168.1.160	
192.168.1.192	
192.168.1.224	(broadcast subnet)

The new subnet mask octet, which is 224 in this example, is also called the **"interesting octet"**. You can also call the corresponding octet in the subnet network address the "interesting octet". The interesting octet is where subnetting "takes place", and where the subnet bits meet the host bits. It is the last non-zero octet in a subnet mask. As for our original problem, we only needed 5 new subnets. We can pick any of these 8 subnets, and leave 3 of them for future use. The best practice is to choose a block of contiguous subnets.

The highlighted octets here are examples of interesting octets.

Network Address = 131.173.144.0
Subnet Mask = 255.255.254.0

Network Address = 95.96.0.0
Subnet Mask = 255.224.0.0

Network Address = 201.188.167.48
Subnet Mask = 255.255.255.240

Notice that the network addresses have all the binary host bits set to 0's. As we said earlier in this chapter…

1. Network addresses are all the IP addresses with all host bits set to 0's.
2. Network addresses are related to all possible values in the extra bits.

Notice also that the network addresses relate to the size of each subnet. In the above example, each subnet contains 30 hosts, but it's really 32 addresses per subnet. **The interesting octet of the network address increments by 32**. This **"network increment"** tells us what the new subnet network addresses are. That then gives us other pieces of information, like the broadcast address and subnet address range. We will talk about the increment again a little later.

The first subnet, where the 3 extra subnet bits are all 0's (000), is called the **Zero Subnet**. The last subnet, where the 3 extra subnet bits are all 1's (111), is called the **Broadcast Subnet**. On some older routers and computers, the zero subnet and broadcast subnet are not used. So the above formulas should be listed as…

2^m = number of subnets
$2^m - 2$ = number of subnets if zero and broadcast subnets are not used
$2^n - 2$ = number of usable hosts per subnet

As explained above, the "m" value refers to the new subnet bits, and the "n" value refers to the remaining host bits.

In each subnet, the first address is the Network Address, and the last address is the Broadcast Address.

Example: 192.168.1.32/27
Network Address = 192.168.1.32
Broadcast Address = 192.168.1.63

Example: 192.168.1.160/27
Network Address = 192.168.1.160
Broadcast Address = 192.168.1.191

Notice that the Broadcast Address is an IP where, in the last octet, the remaining 5 host bits are all 1's.

63 = 00111111
191 = 10111111

This is true in general. In all subnets, an IP address with all host bits set to 1 is a Broadcast Address.

You will notice that all network addresses end in an even number octet or zero. That's just because, in binary numbers, the subnet masks all end in 0's, so after the Logical AND operation with any IP address, you get a network address that ends with 0's also. You will never see a network address like 192.168.1.33 or 145.19.0.67. **This does not mean that all IP addresses that end in an even number are network addresses!** Just because an IP address ends with an even number, does not mean it is a network address. It all depends on the subnet mask. For example, here are some IP addresses that end with even numbers, but they are not network addresses.

192.168.1.44/27
10.16.0.0/10
153.14.76.0/19
219.153.179.64/25

However, if the subnet mask was changed, these IP addresses could be network addresses. Here, we change them.

192.168.1.44/30
10.16.0.0/12
153.14.76.0/23
219.153.179.64/26

Likewise, all broadcast addresses end in an odd number octet. That's because broadcast addresses have host bits of all 1's. A broadcast address is just 1 bit less than the *next network's* network address. You will never see a broadcast address like 192.168.1.24 or 176.28.34.0. **But this does not mean that all IP addresses that end in an odd number octet are broadcast addresses.** Here are some IP addresses that end with odd numbers, but they are not broadcast addresses.

192.168.1.43/27
10.16.0.1/10
153.14.76.27/19
219.153.179.65/25

The subnet mask is what determines the network addresses, and hence, the broadcast addresses.

Let's look at a Class B network.

Example 1.23: 172.16.0.0/16
Let's subnet this major network to get 100 subnets, each holding 500 hosts. Assume we can use the Zero Subnet and the Broadcast Subnet. What is the new subnet mask that we need? How many extra subnet bits do we need?

If we want a subnet with 500 hosts, n = 9, since $2^n - 2 = 2^9 - 2$ = 512 - 2 = 510 hosts. To get 100 subnets, we can set m = 7, since 2^7 = 128 subnets. A new subnet mask where m = 7 extra subnet bits, and n = 9 host bits, is 11111111.11111111.11111110.00000000, which is a /23 subnet mask.

New subnet mask = 255.255.254.0

In this new subnet mask, the interesting octet is 254. We can use 100 of the 128 subnets, and leave the remaining 28 for future use. Since we only needed 500 hosts per subnet, each subnet has 10 extra IP addresses we can use for future growth. Here's a list of some of the 128 new /23 subnets…

Subnet 0 = 172.16.0.0 (zero subnet)

```
Subnet 1 = 172.16.2.0
Subnet 2 = 172.16.4.0
Subnet 3 = 172.16.6.0
.
.
Subnet 125 = 172.16.250.0
Subnet 126 = 172.16.252.0
Subnet 127 = 172.16.254.0    (broadcast subnet)
```

The network addresses come from the extra bits in the new subnet mask octet 11111110. The network addresses are derived from doing a Logical AND operation between every possible IP address in the original 172.16.0.0/16 block and the new subnet mask. The network addresses are related to all possible values in the extra bits (shaded portion).

```
10101100.00010000.xxxxxxxx.xxxxxxxx   ← all possible IP addresses in 172.16.0.0/16
11111111.11111111.11111110.00000000   ← new subnet mask /23
---------------------------------------------------
10101100.00010000.xxxxxxx0.00000000   ← all possible network addresses
```

Here are some of the first few and last few subnets:

```
00000000.00000000 = 0.0
00000010.00000000 = 2.0
00000100.00000000 = 4.0
00000110.00000000 = 6.0
.
.
11111010.00000000 = 250.0
11111100.00000000 = 252.0
11111110.00000000 = 254.0
```

Here are the first 4 broadcast addresses, where all the host bits are set to 1:

```
00000001.11111111 = 1.255 → 172.16.1.255
00000011.11111111 = 3.255 → 172.16.3.255
00000101.11111111 = 5.255 → 172.16.5.255
00000111.11111111 = 7.255 → 172.16.7.255
```

We see here that the interesting octet of the network addresses increment by 2. An easy way to find the network increment is to notice that all the possible Logical AND outputs are just **multiples of the first bit, which is the first value greater than 0**.

00000000 = 0

00000010 = 2 → 2^1 ← **(first bit greater than 0)**

00000100 = 4 → $2^2 = 2\times2^1$

00000110 = 6 → $2^2+2^1 = (2^1+1)\times2^1$

.

.

11111010 = 250 → $2^7+2^6+2^5+2^4+2^3+0+2^1+0 = (2^6+2^5+2^4+2^3+2^2+0+1+0)\times2^1$

11111100 = 252 → $2^7+2^6+2^5+2^4+2^3+2^2+0+0 = (2^6+2^5+2^4+2^3+2^1+0+0)\times2^1$

11111110 = 254 → $2^7+2^6+2^5+2^4+2^3+2^2+2^1+0 = (2^6+2^5+2^4+2^3+2^2+2^1+1+0)\times2^1$

Is there an easier way to find the first bit? Yes, the first bit is just 100000000 minus the interesting octet 11111110.

```
             100000000 = 256
subtract     011111110 = 254     ← interesting octet
             ----------------------
             000000010 = 2       ← increment
```

In **Example 1.22** above, the interesting octet of the subnet mask was 224, so the increment of the interesting octet in the network addresses was 256 - 224 = 32.

```
             100000000 = 256
subtract     011100000 = 224     ← interesting octet
             ----------------------
             000100000 = 32      ← increment
```

As you can see, 100000000 is just 2^8 = 256!! Notice that 100000000 is a 9-bit number, not 8 bits. As we saw earlier, the highest value an octet can have is 255. So you need an extra bit to get to 256.

We see a similar behavior in our regular Base-10 decimal numbering system…

```
             10,000,000
subtract      9,999,000
             --------------
                  1,000 = increment between all values "xxxx000", like from 4,762,000 to 4,763,000.


                1,000,000
subtract          999,900
             ---------------
                      100 = increment of all values "xxxx00", like from 823,700 to 823,800.

               100,000
subtract        99,900
             ------------
                   100 = increment of all values "xxx00", like from 60,500 to 60,600.

             10,000,000
subtract      9,900,000
             --------------
                100,000 = increment all values "xx00000", like from 1,400,000 to 1,500,000.
```

When doing subnetting problems on certification exams, the most important formula you have to remember is this:

network address increment = 256 - (interesting octet of subnet mask)

Another way to say this is that the increment is simply the first bit in the extra (subnet) bits!

The network address increment is the most important number to find. Once you have the increment, you can find the subnet network addresses, the broadcast addresses, and address ranges of the subnets. We will use the increment to solve practically every problem in Chapter 2.

Let's now subnet a Class A network.

Example 1.24: 10.0.0.0/8
Suppose we want to subnet this network to get 4000 subnets, each with 4000 hosts. Assume we can use zero subnet and broadcast subnet.
We want m = 12 and n = 12, since $2^m = 2^{12}$ = 4096 subnets, and $2^n - 2 = 2^{12} - 2$ = 4096 - 2 = 4094. A new subnet mask where m = 12 is 11111111.11111111.11110000.00000000, which is a /20 subnet mask.

New subnet mask = 255.255.240.0

Notice that the extra bits span the 2nd octet and half of the 3rd octet. There's nothing unusual about this. If this mask was /25, the bits would go all the way into the 4th octet. All it means is that the mask has more contiguous 1's. The interesting octet of the subnet mask in this example is 240. **That means the network address increment in the interesting octet is 256 - 240 = 16**. Here's a list of some of the /20 subnets…

```
Subnet 0 = 10.0.0.0              (zero subnet)
Subnet 1 = 10.0.16.0
Subnet 2 = 10.0.32.0
Subnet 3 = 10.0.48.0
.
.
Subnet 15 = 10.0.240.0
Subnet 16 = 10.1.0.0
Subnet 17 = 10.1.16.0
Subnet 18 = 10.1.32.0
.
.
Subnet 4093 = 10.255.208.0
Subnet 4092 = 10.255.224.0
Subnet 4091 = 10.255.240.0       (broadcast subnet)
```

The network addresses are derived from doing a Logical AND operation between every possible IP address in the original 10.0.0.0/8 block and the new subnet mask. The network addresses are related to all possible values in the extra bits (shaded portion).

```
00001010.xxxxxxxx.xxxxxxxx.xxxxxxxx   ← all possible IP addresses in 10.0.0.0/8
11111111.11111111.11110000.00000000   ← new subnet mask /20
-------------------------------------------------
00001010.xxxxxxxx.xxxx0000.00000000   ← all possible network addresses
```

Here are some of the first few, middle and last few subnets:

```
00000000.00000000.00000000 = 0.0.0
00000000.00010000.00000000  = 0.16.0
00000000.00100000.00000000  = 0.32.0
00000000.00110000.00000000  = 0.48.0
.
```

```
.
00000000.11110000.00000000  = 0.240.0
00000001.00000000.00000000  = 1.0.0
00000001.00010000.00000000  = 1.16.0
00000001.00100000.00000000  = 1.32.0
.
.
10110000.10010000.00000000  = 176.144.0
10110000.10100000.00000000  = 176.160.0
10110000.10110000.00000000  = 176.176.0
10110000.11000000.00000000  = 176.192.0
.
.
11001111.11110000.00000000  = 207.240.0
11010000.00000000.00000000  = 208.0.0
11010000.00010000.00000000  = 208.16.0
11010000.00100000.00000000  = 208.32.0
.
.
11111111.11010000.00000000  = 255.208.0
11111111.11100000.00000000  = 255.224.0
11111111.11110000.00000000  = 255.240.0
```

When looking at the shaded area, think of it as a tumbler with numbers on it, like in a gas station meter or a car odometer. The bits are tumbling and increasing from the RIGHT to the LEFT one by one. Notice that the bits in the interesting octet wraps back to 0000 when it reaches 1111, and the octet to the LEFT increases by 1.

Let's verify again that the increment in the interesting octet is the first bit in the extra bits…

00000000 = 0
00010000 = 16 $\rightarrow 2^4$ ← **(first bit greater than 0)**
00100000 = 32 $\rightarrow 2^5 = 2^1\times2^4$
00110000 = 48 $\rightarrow 2^5+2^4 = (2^1+1)\times2^4$
.
.
11010000 = 208 $\rightarrow 2^7+2^6+0+2^4+0+0+0+0 = (2^3+2^2+0+1+0+0+0+0)\times2^4$
11100000 = 224 $\rightarrow 2^7+2^6+2^5+0+0+0+0+0 = (2^3+2^2+2^1+0+0+0+0+0)\times2^4$
11110000 = 240 $\rightarrow 2^7+2^6+2^5+2^4+0+0+0+0 = (2^3+2^2+2^1+1+0+0+0+0)\times2^4$

The increment is 2^4 = 16, and all the "interesting octets" of the network addresses are just multiples of the first bit!

```
             100000000 = 256
subtract     011110000 = 240      ← interesting octet
             -------------
             000010000 = 16       ← increment
```

As we see again, the **network increment = 256 - the interesting octet of the subnet mask.**

Here are the first 4 broadcast addresses, where all the host bits are set to 1:

```
00000000.00001111.11111111 = 0.15.255 → 10.0.15.255
```

00000000.00011111.11111111 = 0.31.255 → 10.0.31.255
00000000.00101111.11111111 = 0.47.255 → 10.0.47.255
00000000.00111111.11111111 = 0.63.255 → 10.0.63.255

By now, you've probably noticed that the interesting octet in all subnet masks will have one of these values:

10000000 = 128
11000000 = 192
11100000 = 224
11110000 = 240
11111000 = 248
11111100 = 252
11111110 = 254

As you do more subnetting examples in this chapter, and in Chapter 2, you will remember these masks and their decimal values more and more.

Notice how important the subnet mask is when it comes to subnetting. Without knowing the subnet mask, we don't know anything about the network. In other words, "172.16.1.1" doesn't tell you if it's 172.16.1.1/16, 172.16.1.1/19 or 172.16.1.1/26. We need to know the subnet mask to find out what network an IP is on.

Can you use the same IP address on different subnets? No, you cannot use the same IP address on different subnets on the internet. You can, however, use the same private **RFC 1918** address on different subnets within your organization. For example, you can put 192.168.1.1/24 on an internal lab machine, and put 192.168.1.1/28 on an internal workstation somewhere else. You will, of course, need some internal Network Address Translation if you needed the two devices to exchange data.

Example 1.25: **183.201.0.0/16**

Suppose we want to subnet this network to get 500 subnets, each with 100 hosts. Assume we can use the zero subnet and broadcast subnet.
We want m = 9 and n = 7, since $2^m = 2^9$ = 512 subnets, and $2^n - 2 = 2^7 - 2$ = 128 - 2 = 126. A new subnet mask where m = 9 is 11111111.11111111.11111111.10000000, which is a /25 subnet mask.

New subnet mask = 255.255.255.128

Notice that the extra bits go into the 4th octet. The interesting octet of the subnet mask in this example is 128. That means the increment is 256 - 128 = 128. Here's a list of some of the /25 subnets…

Subnet 0 = 183.201.0.0 (zero subnet)
Subnet 1 = 183.201.0.128
Subnet 2 = 183.201.1.0
Subnet 3 = 183.201.1.128
.
.
Subnet 509 = 183.201.254.128
Subnet 510 = 183.201.255.0
Subnet 511 = 183.201.255.128 (broadcast subnet)

The network addresses are derived from doing a Logical AND operation between every possible IP address in the original 183.201.0.0/16 block and the new subnet mask. The network addresses are related to all possible values in the extra bits (shaded portion).

```
10110111.11001001.xxxxxxxx.xxxxxxxx    ← all possible IP addresses in 183.201.0.0/16
11111111.11111111.11111111.10000000    ← new subnet mask /25
-----------------------------------------------
10110111.11001001.xxxxxxxx.x0000000    ← all possible network addresses
```

Here are some of the first few and last few subnets:

```
00000000.00000000 = 0.0
00000000.10000000 = 0.128
00000001.00000000 = 1.0
00000001.10000000 = 1.128
.
.
11111110.10000000 = 254.128
11111111.00000000 = 255.0
11111111.10000000 = 255.128
```

It’s easy to see here the increment of the interesting octet in the network addresses is 128. Since 128+128 = 256, the subnet bit in the interesting octet just wraps back to 0 and the octet to the left increases by 1.

Here are the first 4 broadcast addresses, where all the host bits are set to 1:

```
00000000.01111111 = 0.127 → 183.201.0.127
00000000.11111111 = 0.255 → 183.201.0.255
00000001.01111111 = 1.127 → 183.201.1.127
00000001.11111111 = 1.255 → 183.201.1.255
```

Chapter 2 includes exercises on finding network addresses, number of subnets, and number of hosts per subnet.

1.4 Understanding VLSM

VLSM stands for Variable Length Subnet Masks, and its basically concerned with subnetting networks that have already been subnetted. In other words, we are subnetting subnets. Before, we were subnetting from the classful major networks, like from /8, /16 or /24. Now we are subnetting from the subnets themselves.

Why do we want to use VLSM? Because VLSM allows us to use IP addresses more efficiently. Suppose you have a subnet address that can hold 4000 IP addresses, but your largest LAN only holds 200 hosts. And suppose you have several more LAN's that hold even fewer hosts. How do you use that single 4000 IP's subnet? If there was no such thing as subnetting or VLSM, you would have to use that 4000 IP's subnet on the LAN with 200 hosts, and then order some smaller networks. You would be wasting 3800 IP's on the large network. By subnetting, and using VLSM, you get to divide up more and more of the address space, and allocate that space to more and more networks. You would be more efficient, and less wasteful, with your IP addresses.

We've actually been doing a simple form of VLSM already. In the above examples, we've been subnetting major networks with default subnet masks. All we did was extend the number of bits in the subnet mask. We chopped up the major network into a number of subnets, each one with their own subnet network address and broadcast address.

So what's the difference with the VLSM we will learn here? We're just going to subnet an existing subnet. We will also choose specific subnets to subnet, and leave others for future use. VLSM questions are basically questions about subnet ranges, and selecting which ones to use, which ones to subnet, and which ones to leave alone for future use. VLSM requires some planning, and doing complex VLSM problems requires paper and pencil. VLSM questions are also somewhat open ended, because a lot of times there will be several different ways to solve a VLSM problem, and there may be different correct answers. On the certification exams, you will need to eliminate the wrong answers and select the best answers for the correct response. We will do some simple VLSM examples here, and Chapter 2 contains more examples and exercises. More challenging VLSM problems will be done in Chapter 3.

Example 1.26: 192.168.1.128/26

Assume here that the Class C network 192.168.1.0/24 has been subnetted to /26. The subnet mask of a /26 is 255.255.255.192. The interesting octet is 192, so the network increment is 256 - 192 = 64. This also means there are 64 addresses in the subnet (62 usable host addresses). Another way to tell is that the interesting octet is 11000000, so $2^n - 2 = 2^6 - 2 = 62$ hosts.

Here is the address range inside 192.168.1.128/26...

```
192.168.1.128 (network address)
192.168.1.129 (first usable host address)
192.168.1.130
.
.
192.168.1.190 (last usable host address)
192.168.1.191 (broadcast address)
---------------------------------------------
192.168.1.192 (next network address)
```

With VLSM, we can subnet this subnet further. For example, suppose we need IP's for networks that only contain 12 hosts. We can apply a higher subnet mask to this subnet. We will apply a /28, which is subnet mask 255.255.255.240. The new interesting octet is 240, or 11110000. This means there are 2 extra subnet bits, and 4 host bits remain. That means m = 2, and n = 4 for the new subnet mask.

```
network address =    11000000.10101000.00000001.10000000 = 192.168.1.128
old subnet mask =    11111111.11111111.11111111.11000000 = 255.255.255.192
new subnet mask =    11111111.11111111.11111111.11110000 = 255.255.255.240
```

Because the interesting octet is 240, the network increment is 16. Since m = 2 and n = 4, there are $2^m = 2^2 = 4$ new subnets created, and $2^n - 2 = 2^4 - 2 = 16 - 2 = 14$ hosts per subnet. In VLSM problems, we are normally allowed to use the zero and broadcast subnets.

Since the increment is 16, the 4 new subnets look like this…

```
Subnet 0 = 192.168.1.128     (zero subnet)
Subnet 1 = 192.168.1.144
Subnet 2 = 192.168.1.160
Subnet 3 = 192.168.1.176     (broadcast subnet)
-----------------------------------
next subnet = 192.168.1.192
```

We can also find these subnets using the binary forms like we did earlier in this chapter. The network addresses are derived from doing a Logical AND operation between every possible IP address in the original

192.168.1.128/26 block and the new subnet mask. The network addresses are related to all possible values in the extra bits (shaded portion).

```
11000000.10101000.00000001.1xxxxxxx    ← all possible IP addresses in 192.168.1.128/26
11111111.11111111.11111111.11110000    ← new subnet mask /28
---------------------------------------------------
11000000.10101000.00000001.1xxx0000    ← all possible network addresses

                  10000000 = 128 (zero subnet)
                  10010000 = 144
                  10100000 = 160
                  10110000 = 176 (broadcast subnet)
                  ---------------------
                  11000000 = 192 ← next subnet
```

As mentioned earlier in this chapter, older routers may not allow the use of subnet network addresses where the subnet bits are all 0's and all 1's. These are the zero subnet and broadcast subnets, respectively. If zero and broadcast subnets are not allowed, then we get 2 new subnets, instead of 4 new subnets. Modern routers and routing protocols that can use VLSM usually allow the use of zero and broadcast subnets.

In any case, let's use 192.168.1.144/28 for the subnet we need to hold 12 hosts.

In real life networks, the links between routers need their own networks. These networks only contain router links. Point-to-point links between 2 routers obviously only need 2 IP addresses. The subnets on these point-to-point links are /30 subnets, since n = 2 and $2^n - 2 = 2^2 - 2 = 2$ usable IP's. The /30 subnet mask is 255.255.255.252. Suppose we need some of these small /30 subents for some routers.

We can take 192.168.1.160/28 and subnet it even further to /30.

```
network address =    11000000.10101000.00000001.10100000 =  192.168.1.160
old subnet mask =    11111111.11111111.11111111.11110000 =  255.255.255.240
new subnet mask =    11111111.11111111.11111111.11111100 =  255.255.255.252
```

Since we are going from /28 to /30, m = 2 and n = 2, and there are $2^m = 2^2 = 4$ new subnets created, and $2^n - 2 = 2^2 - 2 = 4 - 2 = 2$ hosts per subnet. The new interesting octet is 252, so the network increment is 4.

The 4 new subnets look like this…

```
Subnet 0 = 192.168.1.160     (zero subnet)
Subnet 1 = 192.168.1.164
Subnet 2 = 192.168.1.168
Subnet 3 = 192.168.1.172     (broadcast subnet)
----------------------------------
next subnet = 192.168.1.176
```

The network addresses are derived from doing a Logical AND operation between every possible IP address in the original 192.168.1.160/28 block and the new subnet mask.

```
11000000.10101000.00000001.101xxxxx    ← all possible IP addresses in 192.168.1.160/28
```

```
11111111.11111111.11111111.11111100    ← new subnet mask /30
--------------------------------------------------
11000000.10101000.00000001.101xxx00    ← all possible network addresses

                10100000 = 160 (zero subnet)
                10100100 = 164
                10101000 = 168
                10101100 = 172 (broadcast subnet)
                --------------------
                10110000 = 176 ← next subnet
```

Let's use 192.168.1.164/30 and 192.168.1.168/30 for our point-to-point link subnets.

To summarize, we started with this large subnet that could hold 62 hosts.

```
                192.168.1.128/26
```

We then subnetted that to /28, and got these subnets…

```
                192.168.1.128/28
                192.168.1.144/28
                192.168.1.160/28
                192.168.1.176/28
```

We then used 192.168.1.144/28 to hold a network with 12 hosts. Then we subnetted 192.168.1.160 further, and got 4 more subnets. At the end we got these subnets…

```
                192.168.1.128/28
                192.168.1.144/28 (use on 12 host subnet)
                192.168.1.160/30
                192.168.1.164/30 (use on point-to-point link)
                192.168.1.168/30 (use on point-to-point link)
                192.168.1.172/30
                192.168.1.176/28
                ----------------------
                192.168.1.192 ← next subnet
```

The unused subnets can be used in the future, or further subnetted. As you can see, although subnetting and VLSM does allow us to use IP addresses more efficiently, we do lose some IP's every time we subnet. Remember that on each subnet, we cannot use the Network Address and the Broadcast Address.

Example 1.27: **172.16.8.0/21**

Suppose we are given this Class B subnet, which can hold $2^{11} - 2$ = 2046 hosts, since n = 11. We need IP's allocated to LAN's that will hold 200 hosts, 50 hosts and 2 hosts (for router point-to-point links). What subnet masks will we need, and which subnets can we use? Assume we can use zero subnet and broadcast subnets.

The IP address range for the current subnet network goes up to 172.16.15.255:

```
                172.16.8.0/21  (current subnet)
                172.16.16.0/21          (next subnet)
```

To get 200 hosts, n = 8, since $2^8 - 2$ = 254. To get 50 hosts, n = 6, since $2^6 - 2$ = 62. Starting with a /21 subnet, we can increase the subnet mask to /24 to get n = 8. After that, we can use one of the new subnets to go to a /26 subnet to get n = 6. Once we get the /26 subnets, we can subnet further to /30 to get the 2 host subnets.

For /24, m = 3, so we get 8 extra subnets. The new subnet mask is 255.255.255.0, and the 3rd octet is the interesting octet. The network increment is 256 - 255 = 1. Here are the 8 new /24 subnets:

172.16.8.0
172.16.9.0
172.16.10.0
172.16.11.0
172.16.12.0
172.16.13.0
172.16.14.0
172.16.15.0
172.16.16.0 (next /21 subnet)

We can verify this by doing Logical AND operations between every possible IP address in the original 172.16.8.0/21 block and the new subnet mask.

```
10101100.00010000.00001xxx.xxxxxxxx    ← all possible IP addresses in 172.16.8.0/21
11111111.11111111.11111111.00000000    ← new subnet mask /24
---------------------------------------------------
10101100.00010000.00001xxx.00000000    ← all possible network addresses

          00001000.00000000 = 8.0
          00001001.00000000 = 9.0
          00001010.00000000 = 10.0
          00001011.00000000 = 11.0
          00001100.00000000 = 12.0
          00001101.00000000 = 13.0
          00001110.00000000 = 14.0
          00001111.00000000 = 15.0
```

Let's choose 172.16.8.0/24 for the 200 host LAN. Let's use 172.16.15.0 for the next subnet. We increase the subnet mask from /24 to /26, and m = 2. The new subnet mask is 255.255.255.192 and the increment is 64. We get 4 new subnets…

172.16.15.0
172.16.15.64
172.16.15.128
172.16.15.192

We can verify this by doing Logical AND operations between every possible IP address in the original 172.16.15.0/24 block and the new subnet mask.

```
10101100.00010000.00001000.xxxxxxxx    ← all possible IP addresses in 172.16.15.0/24
11111111.11111111.11111111.11000000    ← new subnet mask /26
---------------------------------------------------
10101100.00010000.00001000.xx000000    ← all possible network addresses

          00000000 = 0
```

```
01000000 = 64
10000000 = 128
11000000 = 192
```

Let's choose 172.16.15.0/26 for our 60 host LAN.

We can subnet the 172.16.15.192/26 subnet into some /30 subnets. We will get 2^4 = 16 new subnets, each with 2 hosts. The increment will be 4. Here are some of the /30 subnets…

```
172.16.15.192
172.16.15.196
172.16.15.200
172.16.15.204
.
.
172.16.15.248
172.16.15.252
```

We can verify this by doing Logical AND operations between every possible IP address in the original 172.16.15.192/26 block and the new subnet mask.

```
10101100.00010000.00001000.11xxxxxx    ← all possible IP addresses in 172.16.15.192/26
11111111.11111111.11111111.11111100    ← new subnet mask /30
------------------------------------------------
10101100.00010000.00001000.11xxxx00    ← all possible network addresses
```

```
11000000 = 192
11000100 = 196
11001000 = 200
11001100 = 204
.
.
11111000 = 248
11111100 = 252
```

We will use these /30 subnets for the router point-to-point links.

So we took our 172.16.8.0/21 subnet, and subnetted it into these subnets:

```
172.16.8.0/24          (use for 200 host LAN)
172.16.9.0/24
172.16.10.0/24
172.16.11.0/24
172.16.12.0/24
172.16.13.0/24
172.16.14.0/24
172.16.15.0/26                  (use for 60 host LAN)
172.16.15.64/26
172.16.15.128/26
172.16.15.192/30
172.16.15.196/30
172.16.15.200/30
```

172.16.15.204/30
.
.
172.16.15.248/30
172.16.15.252/30

As you can see, we can use any of these subnets for future networks of various sizes. We can subnet some of them further, and leave some as they are. We will no longer be wasting IP addresses in one single large network. This is what VLSM is all about.

Notice that with VLSM, a network and subnet mask tells us the next network address, but it doesn't tell us the next network's subnet mask, or the previous network address. In other words, in a VLSM environment, given 192.168.2.96/27, we have no way of knowing what the previous network address is. The previous network could be 192.168.2.64/27, but it could also be 192.168.2.92/30 or 192.168.2.32/26. We have no way of knowing without additional information.

????????????	← what's the previous network address?
192.168.2.96/27	
192.168.2.128/???	← what's the next network's subnet mask?

We know that the next network address has to be 192.168.2.128, simply because 192.168.2.96/27 ends at 192.168.2.127 (the broadcast address). But we have no way of knowing 192.168.2.128's subnet mask, it could be /27, /26, /28 or anything else. In a VLSM environment, you need to know the IP addressing plan of your networks. Without VLSM, you can only assume the major class network has been subnetted.

Of course, when doing VLSM, the network addresses and subnet masks have to make sense. You need to make sure the subnets' address ranges don't overlap. In other words, you can't have these sets of subnet network addresses:

192.168.1.32/27 ← should be /28 or higher	172.16.128.0/17 ← should be /20 or higher
192.168.1.48/28	172.16.144.0/20
192.168.1.64/27	172.16.160.0/20

Always double check your subnet masks make sure the addresses don't overlap. If you tried to enter an overlapping subnet mask into a router, you will probably get an error. Chapter 2 will go into resolving VLSM questions as quickly as possible and provide exercises for practice.

Just for reference, here's a list of all the possible subnet masks. You don't need to memorize this. As you do more subnetting, these masks will just become obvious.

11111111.11111111.11111111.11111111 = 255.255.255.255 = /32
11111111.11111111.11111111.11111110 = 255.255.255.254 = /31
11111111.11111111.11111111.11111100 = 255.255.255.252 = /30
11111111.11111111.11111111.11111000 = 255.255.255.248 = /29
11111111.11111111.11111111.11110000 = 255.255.255.240 = /28
11111111.11111111.11111111.11100000 = 255.255.255.224 = /27
11111111.11111111.11111111.11000000 = 255.255.255.192 = /26
11111111.11111111.11111111.10000000 = 255.255.255.128 = /25
11111111.11111111.11111111.00000000 = 255.255.255.0 = /24
11111111.11111111.11111110.00000000= 255.255.254.0 = /23
11111111.11111111.11111100.00000000= 255.255.252.0 = /22
11111111.11111111.11111000.00000000= 255.255.248.0 = /21

11111111.11111111.11110000.00000000= 255.255.240.0 = /20
11111111.11111111.11100000.00000000= 255.255.224.0 = /19
11111111.11111111.11000000.00000000= 255.255.192.0 = /18
11111111.11111111.10000000.00000000= 255.255.128.0 = /17
11111111.11111111.00000000.00000000= 255.255.0.0 = /16
11111111.11111110.00000000.00000000= 255.254.0.0 = /15
11111111.11111100.00000000.00000000= 255.252.0.0 = /14
11111111.11111000.00000000.00000000= 255.248.0.0 = /13
11111111.11110000.00000000.00000000= 255.240.0.0 = /12
11111111.11100000.00000000.00000000= 255.224.0.0 = /11
11111111.11000000.00000000.00000000= 255.192.0.0 = /10
11111111.10000000.00000000.00000000= 255.128.0.0 = /9
11111111.00000000.00000000.00000000= 255.0.0.0 = /8
11111110.00000000.00000000.00000000= 254.0.0.0 = /7
11111100.00000000.00000000.00000000= 252.0.0.0 = /6
11111000.00000000.00000000.00000000= 248.0.0.0 = /5
11110000.00000000.00000000.00000000= 240.0.0.0 = /4
11100000.00000000.00000000.00000000= 224.0.0.0 = /3
11000000.00000000.00000000.00000000= 192.0.0.0 = /2
10000000.00000000.00000000.00000000= 128.0.0.0 = /1
00000000.00000000.00000000.00000000 = 0.0.0.0 = /0

Notice that /31 is not used, as a subnet with 0 hosts ($2^0 - 2$) is practically useless.

Does /32 or /0 mean anything? Yes, /32 refers to the IP address or host itself. This is like a singularity, it only points to the IP and host. It's not a network. In other words, when we write "192.168.1.1/32", we are referring to that specific IP address (also know as unicast address). It is a way to make clear that we are referring to a host, and not a network or subnet.

What does /0 or 0.0.0.0 mean? It refers to all possible IP addresses or networks, like in a router's default route. If you have a default route to destination network 0.0.0.0, you can send packets to some unknown destination to that 0.0.0.0 route.

If you think about it, a host is its own /32 "subnet" (it's not really a subnet), all real subnets (like /27, /15, /9, etc) are just summarizations of a set or block /32 addresses. The largest possible summary network is /0, which encompasses all IP addresses. We will talk more about classless addressing, supernetting and summarization in Chapter 3.

Another thing you don't need to memorize are the powers of 2. We will list the first 17 (from 0 to 16) as a reference. You will automatically remember these as you do more subnetting. You can also just remember that you just multiply by 2 to get the next one.

$2^0 = 1$
$2^1 = 2$
$2^2 = 4$
$2^3 = 8$
$2^4 = 16$
$2^5 = 32$
$2^6 = 64$
$2^7 = 128$
$2^8 = 256$
$2^9 = 512$

$2^{10} = 1024$
$2^{11} = 2048$
$2^{12} = 4096$
$2^{13} = 8192$
$2^{14} = 16384$
$2^{15} = 32768$
$2^{16} = 65536$

1.5 Chapter 1 Summary

1. An IP address is a 32-bit number.

2. Using the Boolean Logical AND operation, an IP address and a Subnet Mask gives the Network Address.

3. Subnetting an existing network with a new subnet mask provides new subnets. The new subnet addresses are found by doing a Logical AND between all possible IP's in the network and the new subnet mask. The network addresses are related to all possible values in the extra bits. **The interesting octet of the network addresses are all multiples of the first bit greater than zero in the extra (subnet) bits. This first bit is 256 - "the interesting octet" of the subnet mask.** The interesting octet is the last non-zero octet in the subnet mask and of the network address. We call the **256 – ("the interesting octet" of the subnet mask)** value the **network address increment**. The network increment tells us the subnet network addresses. Once we know the network addresses, we can find the address ranges and broadcast addresses.

1. Network addresses are all the IP addresses with all host bits set to 0's.
2. Network addresses are related to all possible values in the extra bits.
3. The interesting octet of the Network Addresses increments by 256 – (interesting octet of the Subnet Mask)

4. When we subnet a network, we can find the extra number of subnets gained, and the number of hosts in each subnet. The number of extra subnets is $\mathbf{2^m}$, where "**m**" is the number of extra bits (subnet bits) in the new subnet mask; it is $\mathbf{2^m - 2}$ if the Zero and Broadcast subnets have to be excluded. The number of hosts/subnet is $\mathbf{2^n - 2}$, where "**n**" is the number of host bits remaining in the new subnet mask. Some old routers and computers require the number of extra subnets to be $\mathbf{2^m - 2}$.

5. A subnet's Network Address is the IP address with all the host bits set to 0. The Network Address is the first address in a subnet's address range. The second IP address is the first usable IP for a host.

6. A subnet's Broadcast Address is the IP address with all the host bits set to 1. The Broadcast Address is the last address in a subnet's address range. The IP address before the Broadcast Address is the last usable IP for a host.

7. The Network Address with all the subnet bits set to 0 is the "Zero Subnet" or "Subnet Zero". The Network Address with all the subnet bits set to 1 is the Broadcast Subnet/Network. On some older routers or computers, you cannot use the zero and broadcast networks.

8. Classless Interdomain Routing (CIDR) notation, or slash "/" notation, is used to simplify the writing of subnet masks. Instead of 255.255.255.224, we can write /27.

9. Variable Length Subnet Masking (VLSM) is simply the subnetting of existing subnets. VLSM allows large networks to be divided into smaller more usable subnets so IP addresses will not be wasted. In VLSM questions, zero and broadcast subnets are usually allowed.

Below are exercises for practice. For more exercises, go to this free website: http://www.subnettingquestions.com/custom/bren/ .

1.6 Exercises

Convert these decimal numbers to binary:

1. 0
2. 6
3. 14
4. 25
5. 49
6. 63
7. 98
8. 107
9. 122
10. 136
11. 163
12. 200
13. 219
14. 243
15. 253
16. 256
17. 285
18. 302
19. 483

Convert these binary numbers to decimal:

20. 00110011
21. 00000000
22. 11111111
23. 11100011
24. 00001111
25. 11111100
26. 01010111
27. 00000110
28. 11110000
29. 01100111
30. 11001111
31. 00111101
32. 11100001
33. 10011110

34. 100000000

In the following questions, assume you can use zero and broadcast subnets.

35. You have increased a subnet mask by 3 bits. How many new subnets have you created?
36. You have increased a subnet mask by 9 bits. How many new subnets have you created?
37. You had a /9 subnet, and have increased it to /13. How many new subnets have you created?
38. You had a /17 subnet, and have increased it to /22. How many new subnets have you created?
39. You had a /24 subnet, and have increased it to /29. How many new subnets have you created?
40. You have increased a subnet mask, and now there are 3 host bits remaining. How many hosts can each new subnet hold?
41. You have increased a subnet mask, and now there are 7 host bits remaining. How many hosts can each new subnet hold?
42. You have increased a subnet mask, and now there are 11 host bits remaining. How many hosts can each new subnet hold?
43. What are the Broadcast addresses in each of these /24 subnets?

    145.126.203.0/24
    145.126.204.0/24
    145.126.205.0/24

44. What are the Broadcast addresses in each of these /16 subnets?

    188.74.0.0/16
    188.75.0.0/16
    188.76.0.0/16

45. You've applied a new /27 subnet mask to the following /24 network block.

```
11001110.00010011.0001000.xxxxxxxx = 206.19.16.0/24
11111111.11111111.1111111.11100000 = new /27 subnet mask
------------------------------------------------
11001110.00010011.0001000.xxx00000
```

    What are the network addresses of the first 4 new subnets?
    What is the network increment? Assume you can use zero and broadcast subnets.

46. You've applied a new /20 subnet mask to the following /16 network block.

```
00111101.11100111.xxxxxxxx.xxxxxxxx = 61.231.0.0/16
11111111.11111111.11110000.00000000 = new /20 subnet mask
--------------------------------------------------
```

00111101.11100111.xxxx0000.00000000

What are the network addresses of the first 5 new subnets?
What is the network increment? Assume you can use zero and broadcast subnets.

47. You've applied a new /19 subnet mask to the following /15 network block.

01111100.110111xx.xxxxxxxx.xxxxxxxx = 124.220.0.0/15
11111111.11111111.11100000.00000000 = new /19 subnet mask

01111100.110111xx.xxx00000.00000000

What are the network addresses of the first 5 new subnets?
What is the network increment? Assume you can use zero and broadcast subnets.

Please see Chapter 2 for more exercises on subnetting and VLSM.

Chapter 2 Subnetting Step By Step, including VLSM

Chapter 2 requires some prior exposure to IP addressing. If you are totally new to IP addressing, you should read Chapter 1 first, or read here and use Chapter 1 as a reference to answer any questions.

I recommend using a free IP subnet calculator as a learning aid to help you visualize the IP addressing space when subnetting. Many can be found online, and the one I use is at http://www.wildpackets.com/products/free_utilities/ipsubnetcalc/overview .

2.1 Short Review of IP Addresses and Subnetting

An IP address is just a 32-bit number, and we divide it into 4 parts to make it easier to read:

Example of an IP address:

11010111.10010011.00111001.11100001

We call each part of 8 bits an **octet**. Some bits are "turned on" (the 1's), and some bits are "turned off" (the 0's).

A Subnet Mask is used to tell us what Network Address the IP address is located in. A Subnet Mask is just a 32-bit number with contiguous 1's.

Example of a Subnet Mask:

11111111.11111111.11111111.00000000

We get the Network Address by doing a "Logical AND operation" between the IP and the Subnet Mask:

```
11010111.10010011.00111001.11100001
11111111.11111111.11111111.00000000
-----------------------------------
11010111.10010011.00111001.00000000
```

The Network Address is **11010111.10010011.00111001.00000000** .

But doing binary math takes a lot of time, and you have to find the Network Address in a few seconds on the certification exams. We will learn a quicker way to get the Network Address.

We also like to read numbers in decimal, and not binary, so in the above example:

IP address = **215.147.57.225**
Subnet mask = **255.255.255.0**
Network address = **215.147.57.0**

We say that 215.147.57.225, with a subnet mask of 255.255.255.0, is located in network 215.147.57.0.

Another way to write the Subnet Mask is to use CIDR or slash "/" notation to indicate how many bits are "turned on" (seen as "1").

Examples:

```
255.255.255.0 = 11111111.11111111.11111111.00000000 = /24
  255.255.0.0 = 11111111.11111111.00000000.00000000 = /16
```

255.0.0.0 = 11111111.00000000.00000000.00000000 = /8

So, as a shorthand, we write the above IP and Subnet Mask together as **215.147.57.225/24**. You will find IP's and subnet masks written both ways on the certification exams.

While routers and computers may not use IP classes anymore, it helps us to understand subnetting by dividing the IP addresses into 3 classes:

Class A address range = 0.0.0.0 to 126.255.255.255
Class B address range = 128.0.0.0 to 191.255.255.255
Class C address range = 192.0.0.0 to 223.255.255.255

Each class has a Default Subnet Mask:

Class A default mask = 255.0.0.0 = /8
Class B default mask = 255.255.0.0 = /16
Class C default mask = 255.255.255.0 = /24

In real life, we subnet, and the subnet mask have more bits going to the RIGHT. So the subnet masks aren't always going to be just /24, /16 or /8. Here are examples of subnet masks in real life:

11111111.11111111.11111111.11100000 = 255.255.255.224 = /27
11111111.11111111.11111111.11110000 = 255.255.255.248 = /28
11111111.11111111.11111111.11111100 = 255.255.255.252 = /30
11111111.11111111.11111000.00000000 = 255.255.248.0 = /21
11111111.11111111.11100000.00000000 = 255.255.224.0 = /19
11111111.11100000.00000000.00000000 = 255.224.0.0 = /11

Notice in the above examples, Class C addresses can only use the first three subnet masks, while Class B can use the first five, and Class A can use all six subnet masks. So, what happens if subnet mask bits decreased and "shrink" to the LEFT, like this Class C address with a subnet mask of 21 bits?

192.168.1.1/21

Is this possible? Does this make sense? Yes, that's called **Supernetting**, and we'll discuss that in Chapter 3. For now, let's stick with Subnetting.

Notice that Class C addresses only have 1 octet to subnet. But the Class B addresses have up to 2 octets, and the Class A addresses have up to 3 octets to subnet. That's why Class C subnetting problems tend to look "easier". But after reading this chapter, you won't really care anymore. Here are all the Class C subnet masks with their corresponding CIDR notation:

/24 = 255.255.255.0	(default subnet mask)
/25 = 255.255.255.128	
/26 = 255.255.255.192	
/27 = 255.255.255.224	
/28 = 255.255.255.240	
/29 = 255.255.255.248	
/30 = 255.255.255.252	

If you have trouble memorizing all that, you may want to just memorize how many bits the interesting octet has to the RIGHT of the Class C /24 default subnet mask.

/24 → 00000000 → 0 bits → 0 (default subnet mask)
/25 → 10000000 → 1 bits → 128
/26 → 11000000 → 2 bits → 192
/27 → 11100000 → 3 bits → 224
/28 → 11110000 → 4 bits → 240
/29 → 11111000 → 5 bits → 248
/30 → 11111100 → 6 bits → 252

This is also a good way to memorize later in Class B and Class A addresses. Notice that /31 = 255.255.255.254 is unusable for Class C. However, for Class B addresses, it's okay to have a subnet mask like 255.255.254.0. Class B addressing will be discussed later in this chapter. Chapter 1 includes a list of all possible subnet masks and their CIDR notation.

By the way, you probably notice that subnet masks, regardless of Class, always include one of these octets:

0
128
192
224
240
248
252
255

This makes thing easier to remember. You will never see any other numbers, like 216 or 144, as a subnet mask octet. This is discussed some more in Chapter 1.

Pretty much all subnetting questions fall into one of these 3 categories:

1. given an IP address and subnet mask, calculate the Network Address, Broadcast Address, and usable address range.
2. given a major network address and subnet mask, how many subnets and how many hosts can we get?
3. given a major network address and subnet mask, what is the network address of Subnet N?

There are lots of different ways to ask the same question. Pretty much all questions will be one of the above. We will tackle each one, and in each IP address class. We will discuss the #3 category for Class C addresses in this chapter, but for Class A and B, we will discuss it in Chapter 4.

2.2 Solving Class C Subnetting Questions

Given an IP address and Subnet Mask, you actually have everything you need to find the Network Address, Broadcast Address and usable address range. You do not need to do binary math.

Example 2.1:
Suppose we have this…

IP address = 195.114.23.98
Subnet Mask = 255.255.255.240

The subnet mask octet that we need to care about is 240. This is the "**interesting octet**". We don't care about the 255 octets because they are just 11111111.11111111.11111111, and we know the Logical AND operation for those just give back 195.114.23. But what do we do with 98 AND 240? Do we have to do binary math? In Chapter 1, we first looked at binary math, and then found a way to compute the network increment.

All we have to do is 256 - 240 = 16. The 16 is what we call the **network address increment**. The **multiple** of 16 that is closest to 98 is 96, since 16 × 6 = 96. So we now have the Network address…

IP address = 195.114.23.98
Subnet Mask = 255.255.255.240
Network Address = 195.114.23.96

Since the increment is 16, the ***next*** Network Address is simply 195.114.23.112, since 96 + 16 = 112. That also means the Broadcast Address is 195.114.23.111 . The Broadcast Address is simply the address that comes right before the ***next*** Network Address.

IP address = 195.114.23.98
Subnet Mask = 255.255.255.240
Network Address = 195.114.23.96
Broadcast Address = 195.114.23.111
Next Network Address = 195.114.23.112

The Network Address is also called the "Subnet Address" or "Subnet Number"; these terms mean the same thing. We also know the ***previous*** Network Address should be 195.114.23.80, since 96 -16 = 80.

IP address = 195.114.23.98
Subnet Mask = 255.255.255.240
Previous Network Address = 195.114.23.80
Network Address = 195.114.23.96
Broadcast Address = 195.114.23.111
Next Network Address = 195.114.23.112

Actually, as we learned in Chapter 1, the previous Network Address could be different from 195.114.23.80 if VLSM was used on the original 195.114.23.0/24 network. We will work more on VLSM later. For now, we will ignore VLSM.

So, for network 195.114.23.96, the usable IP address range (the IP's you can use for hosts) is simply the addresses between the Network Address and the Broadcast Address…

Usable range = 195.114.23.97 to 195.114.23.110

The IP addresses 195.114.23.97 and 195.114.23.110 are the first and last IP's, respectively. The address 195.114.23.98 is simply one address in this range of IP's.

To find all the values like Network Address, Broadcast Address, and usable IP range, we just need to start with the formula we learned in Chapter 1:

network increment = 256 - (interesting octet of the subnet mask)

Once you get the increment, you just need to do some simple arithmetic to find everything else. There are only a few possible increments, and when you do some subnetting, they'll stick in your head:

256 - 254 = 2
256 - 252 = 4
256 - 248 = 8
256 - 240 = 16
256 - 224 = 32
256 - 192 = 64

256 - 128 = 128

Example 2.2:
Suppose we have this…

IP address = 217.34.189.204
Subnet Mask = 255.255.255.192

The interesting octet is 192. All we do is 256 - 192 = 64. The increment is 64, and the multiple of 64 closest to 204 is 192, since 64 × 3 = 192. The increment of 64 tells us everything we need…

IP address = 217.34.189.204
Subnet Mask = 255.255.255.192
Previous Network Address = 217.34.189.128
Network Address = 217.34.189.192
Broadcast Address = 217.34.189.255
Next Network Address = 217.34.190.0

Notice that the next Network Address, 217.34.190.0, is not really part of the 217.34.189.xxx block, but we put it here to show there's an upper boundary. Notice that an octet can't be higher than 255, since 11111111 is the highest binary number in the octet. When we add 1 more bit, then we go to the next network block, which is 217.34.190.0 . We will keep on subnetting pass boundaries like this when we learn VLSM later in this chapter.

The usable address range is just the IP's between the Network address and the Broadcast address. We list all the values here…

IP address = 217.34.189.204
Subnet Mask = 255.255.255.192
Previous Network Address = 217.34.189.128
Network Address = 217.34.189.192
First usable address = 217.34.189.193
Last usable address = 217.34.189.254
Broadcast Address = 217.34.189.255
Next Network Address = 217.34.190.0

When you're calculating these values, notice that all you really need to care about is that last octet. The other octets, 217.34.189, stay the same. Just think about that last octet.

Example 2.3:
Suppose we have this Class C address…

IP address = 197.14.253.125
Subnet Mask = 255.255.255.248

The interesting octet is 248. The increment is 8, since 256 - 248 = 8. The multiple of 8 closest to 125 is 120, since 8 × 15 = 120. The increment and multiple tells us everything we need…

IP address = 197.14.253.125
Subnet Mask = 255.255.255.248
Previous Network Address = 197.14.253.112
Network Address = 197.14.253.120
First usable address = 197.14.253.121

Last usable address = 197.14.253.126
Broadcast Address = 197.14.253.127
Next Network Address = 197.14.253.128

The **256 - (interesting octet)** equation is easy to remember. It always works, and comes from binary math, as you learned in Chapter 1. What may be difficult is finding the closest multiples to get the Network Address. You need to do some quick multiplications in your head, maybe even some guessing, before you corner in on the exact multiple you need.

Example 2.4:
I often have trouble remembering that 16 × 7 = 112. I don't do this everyday. So what I do is guess a bit until I narrow down the number we need. Suppose we have this…

IP address and subnet mask = 192.168.1.115/28

Remember that /28 is just 255.255.255.240 . The interesting octet is 240. So 256 - 240 = 16, and we need to find the multiple of 16 closest to 115. We know 16 × 6 = 96. But 96 looks too far from 115. I think we can add 96 + 16 = 112, which is even closer to 115. We can't add another 16, or we'll get past 115. So now we have the closest multiple we need, which is 112 (and now we know 16 × 7 = 112).

IP address = 192.168.1.115
Subnet Mask = 255.255.255.240
Previous Network Address = 192.168.1.96
Network Address = 192.168.1.112
First usable address = 192.168.1.113
Last usable address = 192.168.1.126
Broadcast Address = 192.168.1.127
Next Network Address = 192.168.1.128

On the certification exams, you will sometimes find IP's and subnet masks written with the CIDR slash "/" notation, like 211.188.45.77/26. Sometimes the subnet mask is written in longer form, like 255.255.255.192. Either way, it is the same thing. At the end of the chapter are exercises that allow you to practice using both the shorter and longer forms of subnet masks.

Example 2.5:
Here's an example where getting the Network Address may seem a little confusing.

IP address = 200.111.33.12
Subnet Mask = 255.255.255.224

The interesting octet is 224. The increment is 32, since 256 - 224 = 32. But there is no multiple of 32 that is closest to 12, except for 0 (12 × 0 = 0). Well, that does work. This IP address is located in the very first subnet, sometimes called Subnet 0, or "zero subnet".

IP address = 200.111.33.12
Subnet Mask = 255.255.255.224
Network Address = 200.111.33.0
First usable address = 200.111.33.1
Last usable address = 200.111.33.30
Broadcast Address = 200.111.33.31
Next Network Address = 200.111.33.32

Notice that in this case, we do not state the previous Network Address. We would need more information to figure out what the previous network address is, especially in a VLSM environment (which is everywhere in real life).

Example 2.6:
Here's another example…

IP address = 194.34.72.12
Subnet Mask = 255.255.255.240

The interesting octet is 240, and so the increment is 16. The multiple of 16 closest to 12 is 0, since 16 × 0 = 0. So here again is an IP address that is located in the zero subnet.

IP address = 194.34.72.12
Subnet Mask = 255.255.255.240
Network Address = 194.34.72.0
First usable address = 194.34.72.1
Last usable address = 194.34.72.14
Broadcast Address = 194.34.72.15
Next Network Address = 194.34.72.16

Again, notice there is no previous Network Address. We would need more information to figure out what network addresses exist before this one.

2.3 Class C Major Networks: Calculating Number of Subnets and Number of Hosts per Subnet

When working with networks, you often need to subnet a larger network to create more subnets. Each subnet is smaller, but you will waste less addresses. A single major Class C network can hold 254 addresses, but you probably need multiple networks, and some of your LAN's will hold 10 hosts, some hold 50 hosts, some 80 hosts. To use that 1 single major Class C network, you need to subnet that network. You need to find the appropriate subnet mask. The bits in the interesting octet will tell you how many subnets you get, and how many hosts per subnets you get.

Example:

Class C network = 194.177.19.0
Subnet Mask = 255.255.255.248
Interesting octet = 248 = 11111000
number of bits turned on = 5
number of bits turned off = 3
number of usable subnets = $2^5 - 2 = 30$
number of usable hosts/subnet = $2^3 - 2 = 6$

Given a major Class C network, it is easy to find out how many subnets you can get and how many "usable hosts per subnet" you can get when you apply a Subnet Mask. This was discussed in Chapter 1, and here we will show the calculations. Refer to Chapter 1for a list of powers of 2.

One of the most common questions people ask before taking certification exams is whether they can use "subnet zero", and whether they have to "subtract 2" to get the number of usable subnets or usable hosts. The "subnet zero" is also an option on routers to enable the use of the zero subnet and the broadcast subnet. My answer is this…

1. *The certification question should tell you if you can or cannot use subnet zero (and, hence, the broadcast subnet). Most of the time when doing VLSM, you should be able to use subnet zero and the broadcast subnet. If the question asks for "usable subnets", then subtract 2 to get the usable number of subnets. If the question doesn't say anything, or says you can use subnet zero and the broadcast subnet, then you don't have to subtract 2.*
2. *You ALWAYS have to subtract 2 to get the usable number of hosts per subnet. That never changes. You can't use the network address and the broadcast address for hosts.*

Example 2.7:
Example of subnetting a Class C major network…

Major Network address = 210.84.197.0
Subnet Mask = 255.255.255.224

The interesting octet is 224, and in binary form, that is **11100000**. Therefore, 224 has 3 bits turned on, and 5 bits turned off. You can review these concepts in Chapter 1. We simply use the following formulas to find the number of valid subnets and valid hosts per subnet…

2^m = number of subnets, if subnet zero allowed
2^m - 2 = number of subnets, if subnet zero <u>NOT</u> allowed
2^n - 2 = number of usable hosts per subnet

The "m" is the number of 1's (number of bits turned on) in the interesting octet of the subnet mask (technically, it's the number of 1's after the Class C default subnet mask, but for Class C, it means the same thing). The "n" is the number of 0's left in the subnet mask (number of bits turned off). For 224, we have m = 3, and n = 5.

We now have everything we need…

$2^m - 2 = 2^3 - 2 = 8 - 2 = 6$ subnets
$2^n - 2 = 2^5 - 2 = 32 - 2 = 30$ hosts/subnet

If the exam question says you can use the subnet zero (and, hence, the broadcast subnet), then the number of subnets is simply **$2^3 = 8$ subnets**. But "hosts per subnet" will always have a subtraction of 2! Routers and computers do not allow 32 hosts per subnet in the above example.

Finding the Network Addresses to each subnet is easy, too, since we know 256 - 224 = 32, and the increment is 32. We just keep adding 32 until we reach the broadcast subnet.

Subnet 0 = 210.84.197.0 (zero subnet)
Subnet 1 = 210.84.197.32
Subnet 2 = 210.84.197.64
Subnet 3 = 210.84.197.96
Subnet 4 = 210.84.197.128
Subnet 5 = 210.84.197.160
Subnet 6 = 210.84.197.192
Subnet 7 = 210.84.197.224 (broadcast subnet)

As you can see, the number of usable subnets is either 6 or 8, depending on what the certification exam allows you to use the zero subnet or not. On real life routers, "subnet zero" is usually enabled by default, so 8 subnets would be the answer. You really have to read the exam question carefully to find out what they allow.

Also, notice that each subnet above allows for 30 hosts. In each subnet, you just subtract the Network Address itself and the subnet's Broadcast Address. The above **$2^n - 2$ = hosts/subnet** formula also told us the same thing. Remember that each individual subnet network has its own Network Address and Broadcast Address.

Finding the subnets and address ranges is important on the certification exams, especially on the VLSM questions.

On the certification exams, it is common to be asked to find a subnet mask, given a major Class C network and requirements for number of subnets, and number of hosts per subnet.

Example 2.8:
If we have a Class C network of **196.103.45.0** , and we need 3 subnets that can hold 50 hosts each, what subnet mask should we use? Assume we can use subnet zero.

$$2^m = 2^2 = \text{number of subnets} = 4$$
$$2^n - 2 = 2^6 - 2 = \text{number of hosts per subnet} = 64 - 2 = 62$$

So, we need a subnet mask with 2 bits turned on, and 6 bits turned off, which is 11000000 = 192 . Our answer is…

Subnet Mask = 255.255.255.192

we get 4 subnets, each can hold a maximum of 62 hosts, which is great since we only needed 3 subnets, each holding 50 hosts. We can use 3 of the 4 subnets, and keep the remaining subnet for future use. Each subnet will give us 62 hosts, and since we will only use 50 hosts/subnet, each subnet will have 12 free addresses left for future use.

We can also list all 4 subnets…

Subnet 0 = 196.103.45.0 (zero subnet)
Subnet 1 = 196.103.45.64
Subnet 2 = 196.103.45.128
Subnet 3 = 196.103.45.192 (broadcast subnet)

Which 3 subnets should we use? Which one should we keep available for future use? You can use any three of them. The best practice is to use up contiguous blocks, such as Subnets 0, 1 and 2, or Subnet 1, 2 and 3.. That way you can use VLSM. This will be discussed later in this chapter.

Example 2.9:
If we have a Class C network of **204.119.3.0** , and we need 50 subnets with 2 hosts each, what subnet mask should we use? Assume we cannot use subnet zero.

$$2^m - 2 = 2^6 - 2 = 64 - 2 = 62 \text{ usable subnets}$$
$$2^n - 2 = 2^2 - 2 = 4 - 2 = 2 \text{ usable hosts per subnet}$$

So we need a subnet mask with 6 bits turned on, and 2 bits turned off, which is 11111100 = 252. So our answer is…

Subnet Mask = 255.255.255.252

This subnet mask gives us 62 usable subnets, so we can use 50 of them, and keep the remaining 12 subnets for future use. We will list some of the 64 subnets (62 usable), including the last few subnets…

Subnet 0 = 204.119.3.0 (zero subnet) unusable
Subnet 1 = 204.119.3.4
Subnet 2 = 204.119.3.8
Subnet 3 = 204.119.3.12
.
.
Subnet 61 = 204.119.3.244
Subnet 62 = 204.119.3.248
Subnet 63 = 204.119.3.252 (broadcast subnet) unusable

Which of these subnets should we use? we can use any of them, but the best practice is to use contiguous subnets, like Subnet 1 to Subnet 50, or Subnet 13 to Subnet 62. This will allow us to use VLSM on the unused subnets in the future.

If you want to see all 64 subnets, we recommend you use a free IP subnet calculator that lists all the subnets when given a major network and a subnet mask. Notice that in IP address ranges, the last subnet (the broadcast subnet) has a last octet (252 in this example) that is the same with the subnet mask.

Example 2.10:
If we have a Class C major network of **218.42.227.0** and a **Subnet Mask of 255.255.255.128**, how many subnets do we get, and how many hosts per subnet? Assume we can use subnet zero.

Class C network = 218.42.227.0
Subnet mask = 255.255.255.128

The interesting octet = 128, which is 10000000, so m = 1, and n = 7. So the answers are…

$$\mathbf{2^m = 2^1 = 2 \text{ subnets}}$$
$$\mathbf{2^n - 2 = 2^7 - 2 = 128 - 2 = 126 \text{ usable hosts/subnet}}$$

we can also list the subnets…

Subnet 0 = 218.42.227.0 (subnet zero)
Subnet 1 = 218.42.227.128 (broadcast subnet)

As you can see in this example, this subnet mask gives only 2 subnets and 126 hosts per subnet. If subnet zero was not allowed, then we could not use this subnet mask for this Class C major network.

2.4 Class C Major Networks: Calculating Subnet Network Address for Subnet N

Sometimes you are asked to find the Network Address for Subnet N, given a Class C major subnet and a subnet mask.

Example 2.11:
Suppose we need to find the Network Address for **Subnet 37** for this network…

Class C network = 199.16.150.0
Subnet Mask = 255.255.255.252

The interesting octet is 252, and the increment is 256 - 252 = 4. We just need to calculate 37 × 4 = 148. This will be the last octet in the Network Address. So the Network Address for Subnet #37 is…

Network Address = 199.16.150.148

In general, all you need to calculate is…

Last octet = N × (increment)

We actually did something like this in the example above, when we listed all the network addresses. We can list some of the networks here also…

Subnet 0 = 199.16.150.0 (subnet zero)
Subnet 1 = 199.16.150.4
Subnet 2 = 199.16.150.8
Subnet 3 = 199.16.150.12
.
.
Subnet 37 = 199.16.150.148
.
.
Subnet 62 = 199.16.150.248
Subnet 63 = 199.16.150.252 (broadcast subnet)

Example 2.12:
In another example, calculate **Subnet 12** for this network:

Class C network = 211.9.137.0/29

The Subnet mask is 255.255.255.248, so 256 - 248 = 8, and 12 × 8 = 96. So the network address for Subnet 12 **= 211.9.137.96**.

Chapter 4 includes a procedure to calculate Subnet N for Class B and Class A networks. The procedure requires converting the value N to a binary number.

The exercises at the end of the chapter will help you review all Class C network questions. We will now move forward to Class B networks.

2.5 Solving Class B Subnetting Questions

The only difference between Class B and Class C networks, is that Class B networks have 2 octets to worry about. The interesting octet can be in the 3rd octet or the 4th octet.

Here are some subnet masks, with CIDR "/" notation, where the interesting octet is in the 3rd octet:

255.255.192.0 → /18
255.255.240.0 → /20
255.255.254.0 → /23

And here are some subnet masks where the interesting octet is in the 4th octet (just like Class C networks):

255.255.255.128 → /25

255.255.255.240 → /28
255.255.255.248 → /29

Either way, solving for the Network Address, Broadcast Address, and usable address ranges is the same thing. You still use **256 - (interesting octet)** just like in Class C networks.

Example 2.13:
Example with Class B network:

IP address = 129.88.5.34.77
Subnet Mask = 255.255.240.0

The interesting octet is 240. This is the 3rd octet, and notice that the IP address has a 34 in the 3rd octet. The increment is 16, since 256 - 240 = 16. The multiple of 16 closest to 34 is 32, since 16 × 2 = 32. The Network Address is 129.88.5.32.0.

IP address = 129.88.5.34.77
Subnet Mask = 255.255.255.240
Network Address = 129.88.5.32.0

Notice that the Subnet Mask has a 0 in the 4th octet, which is 00000000 in binary. So the Logical AND operation with the 77 in the 4th octet of the IP address will just give 00000000. That's why the Network Address also has a 0 in the 4th octet. Just like in Class C networks, the increment and multiple tell us everything we need…

IP address = 129.88.5.34.77
Subnet Mask = 255.255.255.240
Previous Network Address = 129.88.5.16.0
Network Address = 129.88.5.32.0
First usable address = 129.88.5.32.1
Last usable address = 129.88.5.47.254
Broadcast Address = 129.88.5.47.255
Next Network Address = 129.88.5.48.0

Example 2.14:
Example with Class B network:

IP address and subnet mask = 137.44.66.232/23

The subnet mask for /23 is 255.255.254.0, and the interesting octet is 254. This is the 3rd octet, and notice that the IP address has a 66 in the 3rd octet. The increment is 2, since 256 - 254 = 2. The multiple of 2 closest to 66 is also 66, since 2 × 33 = 66.

IP address = 137.44.66.232
Subnet Mask = 255.255.254.0
Previous Network Address = 137.44.64.0
Network Address = 137.44.66.0
First usable address = 137.44.66.1
Last usable address = 137.44.67.254
Broadcast Address = 137.44.67.255
Next Network Address = 137.44.68.0

Example 2.15:
Example of Class B network with Subnet Mask in 4th octet:

IP address = 190.51.89.137
Subnet Mask = 255.255.255.224

The interesting octet is 224, and the increment is 256 - 224 = 32. The multiple of 32 closest to 137 is 128, since 32 × 4 = 128.

IP address = 190.51.89.137
Subnet Mask = 255.255.255.224
Previous Network Address = 190.51.89.96
Network Address = 190.51.89.128
First usable address = 190.51.89.129
Last usable address = 190.51.89.158
Broadcast Address = 190.51.89.159
Next Network Address = 190.51.89.160

Example 2.16:
Another example of Class B network with Subnet Mask in 4th octet:

IP address = 131.224.233.254
Subnet Mask = 255.255.255.252

The interesting octet is 252, and the increment is 256 - 252 = 4. The multiple of 4 closest to 254 is 252, since 4 × 63 = 252.

IP address = 131.224.233.254
Subnet Mask = 255.255.255.252
Previous Network Address = 131.224.233.248
Network Address = 131.224.233.252
First usable address = 131.224.233.253
Last usable address = 131.224.233.254
Broadcast Address = 131.224.233.255
Next Network Address = 131.224.234.0

Notice that, unlike in Class C networks, in this situation, there is a next Network Address. That's because in this Class B network, the actual block we're subnetting is 131.224.0.0, which goes all the way to 131.224.255.255, which is the Broadcast Address in the very last Broadcast Subnet. Of course, with VLSM, we really need more information about 131.224.234.0, because it might have a different subnet mask. We will discuss VLSM a little later. To get a better visual perspective for these larger address spaces, I recommend using a free IP subnet calculator.

2.6 Class B Major Networks: Calculating Number of Subnets and Number of Hosts per Subnet

Since the default subnet mask of Class B addresses is /16, or 255.255.0.0, there's a lot more room for subnetting than for Class C addresses. As you can see, there are 2 octets for subnetting in Class B. We will still use the same formulas to find the number of subnets and number of hosts per subnet:

2^m - 2 = number of subnets, if subnet zero NOT allowed
2^m = number of subnets, if subnet zero allowed

2^n - 2 = number of usable hosts per subnet

But now in Class B, the "m" and "n" can have a wider range of values. The "m" value is the number of 1's, the bits "turned on", after (to the RIGHT of) the Class B default subnet mask. The "n" value is the number of 0's, the bits "turned off", in the subnet mask. Example…

255.255.240.0 = 11111111.11111111.11110000.00000000

As you can see, there are 4 bits turned on after the Class B default subnet mask. And there are 12 0's. So m = 4 and n = 12. More examples…

255.255.252.0 = 11111111.11111111.11111100.00000000 → m = 6, n = 10
255.255.192.0 = 11111111.11111111.11000000.00000000 → m = 2, n = 14
255.255.255.128 = 11111111.11111111.11111111.10000000 → m = 9, n = 7
255.255.255.248 = 11111111.11111111.11111111.11111000 → m = 13, n = 3

Example 2.17:
Suppose we have **Class B Major Network 129.0.0.0**, and **Subnet Mask 255.255.254.0**. Assume we can use Subnet Zero. How many subnets and hosts per subnets do we get?

Class B network = 129.0.0.0
Subnet mask = 255.255.254.0

The number of 1's and 0's past the Class B default subnet mask is 11111110.00000000, so m = 7, and n = 9. So the answers are…

2^m = 2^7 = 128 subnets
2^n -2 = 2^9 -2 = 512 - 2 = 510 usable hosts/subnet

We also know the increment is 256 - 254 = 2, so we can also list some of the subnet networks…

Subent 0 = 129.0.0.0
Subnet 1 = 129.0.2.0
Subnet 2 = 129.0.4.0
.
.
Subnet 125 = 129.0.250.0
Subnet 126 = 129.0.252.0
Subnet 127 = 129.0.254.0

Notice how with 510 hosts per subnet, the hosts fill up two octets. In other words, notice that there are no subnet networks like 129.0.253.0 or 129.0.23.0. The largest number an octet can have is 255. So 510 hosts must fill up 2 octets. That's why the subnet networks "jump", like from 129.0.2.0 to 129.0.4.0.

Chapter 1 has a list of all the powers of 2. Here are some that will look familiar to you as do more Class B and Class A subnetting.

$2^0 = 1$
$2^1 = 2$
$2^2 = 4$
$2^3 = 8$

$2^4 = 16$
$2^5 = 32$
$2^6 = 64$
$2^7 = 128$
$2^8 = 256$
$2^9 = 512$
$2^{10} = 1024$
$2^{11} = 2048$
$2^{12} = 4096$
$2^{13} = 8192$

Example 2.18:
Suppose we have Class B Major Network **187.211.0.0/27**. How many subnets and hosts per subnets do we get? Assume we cannot use subnet zero.

Class B network = 187.211.0.0
Subnet mask = 255.255.255.224

The bits past the Class B default subnet mask are 11111111.11100000, so m = 11, and n = 5. So the answers are…

$$\mathbf{2^m = 2^{11} - 2 = 2048 - 2 = 2046 \text{ subnets}}$$
$$\mathbf{2^n - 2 = 2^5 - 2 = 32 - 2 = 30 \text{ usable hosts/subnet}}$$

We also know the increment is 256 - 224 = 32, so we can also list some of the subnets…

Subnet 0 = 187.211.0.0 (subnet zero) unusable
Subnet 1 = 187.211.0.32
Subnet 2 = 187.211.0.64
Subnet 3 = 187.211.0.96
.
.
Subnet 2045 = 187.211.255.160
Subnet 2046 = 187.211.255.192
Subnet 2047 = 187.211.255.224 (broadcast subnet) unusable

This Class B major network has been divided up into 2048 subnet networks, 2046 of them usable! That's a lot of subnets! Notice how the 3rd octet just keeps increasing everytime the 4th octet goes past 255.

Now let's find some subnet masks given specific requirements on how many subnets and hosts per subnet we need.

Example 2.19:
Suppose we have a Class B network of **142.155.0.0** , and we need 1000 subnets with 50 hosts each, what subnet mask should we use? Assume we can use subnet zero.

Let's fine the "m" and "n" that gives us the answers we need…

$$\mathbf{2^m = 2^{10} = 1024 = 1024 \text{ subnets}}$$
$$\mathbf{2^n - 2 = 2^6 - 2 = 64 - 2 = 62 \text{ usable hosts per subnet}}$$

So we need a subnet mask with 10 bits turned on and 6 bits turned off (past the Class B default subnet mask), which is 11111111.11000000 = 255.192. So our answer is…

Subnet Mask = 255.255.255.192

This subnet mask gives us 1024 subnets, so we can use 1000 of them, and keep the remaining 24 subnets for future use. We will list some of the 1024 subnets, including the last few subnets…

Subnet 0 = 142.155.0.0 (zero subnet)
Subnet 1 = 142.155.0.64
Subnet 2 = 142.155.0.128
Subnet 3 = 142.155.0.192
.
.
Subnet 1021 = 142.155.255.64
Subnet 1022 = 142.155.255.128
Subnet 1023 = 142.155.255.192 (broadcast subnet)

With Class B addresses, you definitely have to practice calculating powers of 2 at higher values, or at least memorize some of them. Refer to Chapter 1 for a list of powers of 2.

Example 2.20:
Suppose our Class B Major Network is **133.245.0.0**, and we need 60 subnets with 250 hosts each. Suppose that we're told that we will double the number of users on each subnet in the next year. What subnet mask should we use if we want to plan for expansion? Assume we cannot use subnet zero.

If all you want are subnets that hold 250 hosts, you can choose n = 8, let m = 8. You will have $2^8 - 2 = 254$ usable subnets, and $2^8 - 2 =$ 254 hosts per subnet. That will work right now. But, if you want to plan for future expansion, when the number of users might be 500, then you want to increase the "n" value. If we set n = 9, and m = 7, then we have…

$$2^m - 2 = 2^7 - 2 = 126 \textbf{ subnets}$$
$$2^n - 2 = 2^9 - 2 = 510 \textbf{ hosts per subnet}$$

Now we have subnets that can hold more than 500 hosts, which is want we since we plan on having double the number of users. We can use 60 of the 126 subnets, and save the rest for future needs (or subnet those further with VLSM). The subnet mask we want has m = 7 and n = 9, so we have…

Subnet Mask = 11111111.11111111.11111110.00000000 = 255.255.254.0

In the real world, you will often need to account for future growth when subnetting. This brings forth an issue regarding certification exams.

The exam questions will state whether they want you to subnet according to "efficiency" or "plan for future growth". If the question states you should subnet for efficiency, and not waste IP addresses, then make your subnetting as close as possible. For example, in the above example, if there was no need to plan for future growth, we would've chosen a subnet mask where n = 8 and m = 8, which would be 255.255.255.0. That subnet would've allowed for 254 hosts per subnet, and we have 250 hosts right now, so that works fine. We would only be "wasting" 4 IP addresses per subnet, which is very little. If we chose 255.255.254.0, we would be wasting a lot more IP's. But if the question states something about planning for future growth, like you will double the number of users, then you have to look for a better subnet mask, and we chose 255.255.254.0. Bottom line…read the question carefully.

Example 2.21:
Let's have one more example before we go into Class A addresses. Suppose we have a Class B network of **128.142.0.0** , and we need 14 subnets with 4000 hosts each, what subnet mask should we use? Assume we can use subnet zero.

$$\mathbf{2^m = 2^4 = 16 \text{ subnets}}$$
$$\mathbf{2^n - 2 = 2^{12} - 2 = 4096 - 2 = 4094 \text{ hosts per subnet}}$$

So we need a subnet mask with 4 bits turned on and 12 bits turned off (past the Class B default subnet mask), which is 11110000.00000000 = 240.0. So our answer is…

Subnet Mask = 255.255.240.0

We can list some of the subnets…

Subnet 0 = 128.142.0.0 (zero subnet)
Subnet 1 = 128.142.16.0
Subnet 2 = 128.142.32.0
.
.
Subnet 13 = 128.142.208.0
Subnet 14 = 128.142.224.0
Subnet 15 = 128.142.240.0 (broadcast subnet)

Notice that these are very big subnets, each one holding over 4000 hosts!

2.7 Solving Class A Subnetting Questions

Solving Class A networks is the same as Class B networks, only now there's one more octet to think about. Class A networks will have subnet masks that go into the 2nd, 3rd or 4th octets. Here are some examples…

2nd octet → 255.128.0.0
2nd octet → 255.248.0.0
3rd octet → 255.255.192.0
3rd octet → 255.255.254.0
4th octet → 255.255.255.224
4th octet → 255.255.255.248

Example 2.22:
Let's do some problems. Example with Class A network:

IP address = 44.218.77.251
Subnet Mask = 255.248.0.0

The interesting octet is 248. This is the 2nd octet, and notice that the IP address has a 218 in the 2nd octet. The increment is 8, since 256 - 248 = 8. The multiple of 8 closest to 218 is 216, since 8 × 27 = 216. From this we get the Network Address…

IP address = 44.218.77.251
Subnet Mask = 255.248.0.0
Network Address = 44.216.0.0

With the increment, we can calculate the previous Network Address, and the next Network Address, and everything else…

IP address = 44.218.77.251
Subnet Mask = 255.248.0.0
Previous Network Address = 44.208.0.0
Network Address = 44.216.0.0
First usable address = 44.216.0.1
Last usable address = 44.223.255.254
Broadcast Address = 44.223.255.255
Next Network Address = 44.224.0.0

Example 2.23:
Example with Class A network:

IP address = 5.12.227.4
Subnet Mask = 255.255.192.0

The interesting octet is 192. This is the 3rd octet, and notice that the IP address has a 227 in the 3rd octet. The increment is 64, since 256 - 192 = 64. The multiple of 64 closest to 227 is 192, since 64 × 3 = 192. From this we get the Network Address…

IP address = 5.12.227.4
Subnet Mask = 255.255.192.0
Network Address = 5.12.192.0

With the increment, we can calculate the previous Network Address, and the next Network Address, and everything else…

IP address = 5.12.227.4
Subnet Mask = 255.255.192.0
Previous Network Address = 5.12.128.0
Network Address = 5.12.192.0
First usable address = 5.12.192.1
Last usable address = 5.12.255.254
Broadcast Address = 5.12.255.255
Next Network Address = 5.13.0.0

Example 2.24:
Example with Class A network:

IP address and subnet mask = 124.0.203.85/25

The Subnet Mask for /25 is 255.255.255.128, and the interesting octet is 128. This is the 4th octet, and notice that the IP address has a 85 in the 4th octet. The increment is 128, since 256 - 128 = 128. The multiple of 128 closest to 85 is 0, since 128 × 0 = 0. From this we get the Network Address…

IP address = 124.0.203.85
Subnet Mask = 255.255.255.128
Previous Network Address = 124.0.202.128

Network Address = 124.0.203.0
First usable address = 124.0.203.1
Last usable address = 124.0.203.126
Broadcast Address = 124.0.203.127
Next Network Address = 124.0.203.128

2.8 Class A Major Networks: Calculating Number of Subnets and Number of Hosts per Subnet

Class A networks have a lot more subnets and hosts per subnets to calculate. The number of bits in the subnet masks turned on and turned off (the "m" and "n" values) can be quite large. Examples…

255.192.0.0 = 11111111.11000000.00000000.00000000 → m = 2, n = 22
255.252.0.0 = 11111111.11111100.00000000.00000000 → m = 6, n = 18
255.255.254.0 = 11111111.11111111.11111110.00000000 → m = 15, n = 9
255.255.255.248 = 11111111.11111111.11111111.11111000 → m = 21, n = 3

Remember that the default Class A subnet mask is 255.0.0.0, which is /8. The same formulas used for Class C and B are used for Class A networks. Let's do some subnet calculations…

Example 2.25:
Example of Class A major network with a /12 subnet mask…

Class A network = 21.0.0.0
Subnet mask = 255.240.0.0

How many subnets and hosts per subnet do we get? Assume we cannot use subnet zero.
We can see here that m = 4 and n = 20.

$$\mathbf{2^m - 2 = 2^4 - 2 = 14 \text{ usable subnets}}$$
$$\mathbf{2^n - 2 = 2^{20} - 2 = 1048576 - 2 = 1048574 \text{ hosts per subnet}}$$

You probably won't ever need to do 2^{20} in your head on an exam, or need to remember this value. But if you ever do need to do it at a job interview, just remember that $2^{20} = 2^{10} \times 2^{10} = 1024 \times 1024$. You can tell that a $1000 \times 1000 = 1$ million, so its not surprising that 2^{20} is close to 1 million. At a job interview, just give 1 million as a reasonable estimate.

A handy math formula that you learned in high school is this:

$$2^a = 2^b \times 2^c \text{ , if } a = b + c$$

Examples:
$$2^{16} = 2^{10} \times 2^6 = 1024 \times 64 = 65536$$
$$2^9 = 2^7 \times 2^2 = 128 \times 4 = 512$$

This makes multiplying large numbers a little easier. Notice that 1024 × 64 is really close to 1000 × 64 = 64000. At a job interview, just give an estimate. Most people will accept a reasonable estimate for an answer. If you had to do this on a multiple choice exam, just pick the answer closest to 64000, which is 65536.

Part of the strategy in certification exams is choosing the "most right", "closest" or "best answer", or eliminating the "most wrong" or "can't be right" answers. You don't always need to know the exact answer to pick the right one!

In the above example, the increment is 16, since 256 - 240 = 16. Here are some of the first and last subnets…

Subnet 0 = 21.0.0.0 (zero subnet)
Subnet 1 = 21.16.0.0
Subnet 2 = 21.32.0.0
.
.
Subnet 13 = 21.208.0.0
Subnet 14 = 21.224.0.0
Subnet 15 = 21.240.0.0 (broadcast subnet)

These are obviously very large subnets! In the real world, VLSM will be used to subdivide these subnets even further so more networks can use these IP addresses.

Example 2.26:
Example of Class A major network with a /22 subnet mask…

Class A network = 118.0.0.0
Subnet mask = 255.255.252.0

How many subnets and hosts per subnet do we get? Assume we can use subnet zero.
We can see here that m = 14 and n = 10.

$2^m = 2^{14}$ = 16384 subnets
$2^n - 2 = 2^{10} - 2$ = 1024 - 2 = 1022 hosts per subnet

The increment is 4, since 256 - 252 = 4. Here are some of the first and last subnets…

Subnet 0 = 118.0.0.0 (zero subnet)
Subnet 1 = 118.0.4.0
Subnet 2 = 118.0.8.0
.
.
Subnet 16381 = 118.255.244.0
Subnet 16382 = 118.255.248.0
Subnet 16383 = 118.255.252.0 (broadcast subnet)

Example 2.27:
Example of Class A major network with a /26 subnet mask…

Class A network = 126.0.0.0
Subnet mask = 255.255.255.192

How many subnets and hosts per subnet do we get? Assume we can use subnet zero.
We can see here that m = 18 and n = 6.

$2^m = 2^{18}$ = 262144 subnets
$2^n - 2 = 2^6 - 2$ = 64 - 2 = 62 hosts per subnet

The increment is 64, since 256 - 192 = 64. Here are some of the first and last subnets…

Subnet 0 = 126.0.0.0 (zero subnet)
Subnet 1 = 126.0.0.64
Subnet 2 = 126.0.0.128
.
.
Subnet 262141 = 126.255.255.64
Subnet 262142 = 126.255.255.128
Subnet 262143 = 126.255.255.192 (broadcast subnet)

Example 2.28:
Suppose we have Class A major network **115.0.0.0**, and we need 1500 subnets, with 8000 hosts per subnet. What subnet mask do we need? Assume we can use subnet zero.

we can choose m = 11, with 2^{11} = 2048, and n = 13, with $2^{13} - 2$ = 8192 - 2 = 8190. That satisfies our conditions of 1500 subnets and 8000 hosts/subnet. With m = 11, and n = 13, the subnet mask will be…

Subnet mask = 255.255.224.0

Example 2.29:
Suppose we have Class A major network **62.0.0.0**, and we need 64000 subnets, with 240 hosts per subnet. What subnet mask do we need? Assume we can use subnet zero.

We can choose m = 16, with 2^{16} = 65536, and n = 8, with $2^8 - 2$ = 256 - 2 = 254. That satisfies our conditions of 64000 subnets and 240 hosts/subnet. With m = 16, and n = 8, the subnet mask will be…

Subnet mask = 255.255.255.0

In the real world, you won't see too many subnets with millions or thousands of unused addresses. That's because VLSM is used to subnet networks even further so IP addresses can be used on more networks and not wasted.

2.9 Solving Basic VLSM Questions

VLSM questions are basically questions about subnet address ranges. You're given a network address and asked to satisfy some requirements, like you need a subnet that can hold 500 hosts, another subnet to hold 200 hosts, and another to hold 100 hosts. You basically solve the larger subnet first, then subnet further and further "to the RIGHT", and solve the smaller subnets. You'll end up with several different subnets, each with a different subnet mask. The subnet masks get larger as you move to the RIGHT, like you'll start with a /25, then /27 and /29 and so on.

In the real world, VLSM questions tend to be "open ended", because you can decide on which subnet networks to use, and which to save for future use. On the exams, VLSM questions will tend to be multiple choice with multiple answers; they will usually list possible subnets as the answers, and include impossible subnets as the wrong answers.

Some points to remember when doing VLSM questions.

1. The subnet zero and broadcast subnet are usually allowed. So you use 2^m instead of $2^m - 2$.
2. The "m" is no longer just the number of 1's from the Default Subnet Mask, but from the previous Subnet Mask. We'll see this in a moment.
3. To speed up your thought process, it's okay to calculate 2^n instead of $2^n -2$, since the "subtract 2" doesn't make a whole lot of difference. For example, when you want to find a subnet that holds 100 hosts, whether you remember $2^7 -2 = 126$ or $2^7 = 128$ doesn't make much difference. Just calculate or estimate as quickly as possible.
4. VLSM questions will usually state that you should subnet for "efficiency". This means don't waste IP address space, subnet as close as possible. The only exception is when a question asks you to "plan for the future", and in that case, they'll let you know how much the number of hosts will increase.
5. Make sure your answers (if its multiple answers) make sense together. Don't choose subnets that don't make sense together.
6. Find the larger subnets first, then work your way to the smaller and smaller subnets.

You can review VLSM in Chapter 1. Right now, we'll learn by example. We'll do 2 Class C examples, and then finish with Class B and Class A examples.

Example 2.30:
We have a Class C network of **192.168.2.0/24**. We need subnets to hold **100 hosts**, **50 hosts**, and **10 hosts**. What networks can we use?

We'll start with satisfying the 100 hosts requirement. We obviously need a subnet that can hold over 100 hosts, which is a /25 or 255.255.255.128, since the interesting octet is 10000000, n = 7, and $2^7 = 128$ hosts (again, we'll just leave out the "subtract 2" part for expediency). The m = 1, so we have 2 subnets.

Subnet 0 = 192.168.2.0/25
Subnet 1 = 192.168.2.128/25

We'll use 192.168.2.0/25 for our 100 host subnet. We'll subnet 192.168.2.128 further. We need a subnet to hold 50 hosts, in other words, a /26 or 255.255.255.192, since the interesting octet is 11000000, n = 6, m = 1. Notice that m = 1, and not 2. Why? Because the previous Subnet Mask had an octet of 10000000. Since we are subnetting a subnet (and not just a major Class C network), we have to move to the right of the previous Subnet Mask. We get 2 subnets, each with 64 hosts.

Subnet 0 = 192.168.2.0/25 (used on 100 host subnet)
Subnet 1 = 192.168.2.128/26
Subnet 2 = 192.168.2.192/26

Let's pick the 192.168.2.128/26 for the 50 hosts subnet. Notice that we changed the subnet mask for 192.168.2.128 from /25 to /26. The 192.168.2.128 is just a network address, and we are free to change it from /25 to /26. In the real world, we can do this, as long as this is part of a plan that makes sense. VLSM requires planning, and your subnets need to be calculated carefully. Lack of planning can cause problems, like address ranges overlapping.

We'll subnet 192.168.2.192 further. Now we just need a subnet to hold 10 hosts. Why don't we just pick the 192.168.2.192/26? That will work, but we want to subnet as efficiently as possible, not wasting too many IP's. So we subnet one more time. A subnet that can hold 10 hosts is a /28 subnet, or 255.255.255.240, since 240 is 11110000, and $2^n = 2^4 = 16$ hosts. The m = 2, since the previous subnet mask was /26 or 11000000. Since m = 2, we get $2^m = 2^2 = 4$ subnets.

Subnet 0 = 192.168.2.0/25 (used on 100 host subnet)
Subnet 1 = 192.168.2.128/26 (used on 50 host subnet)
Subnet 2 = 192.168.2.192/28
Subnet 3 = 192.168.2.208/28
Subnet 4 = 192.168.2.224/28
Subnet 5 = 192.168.2.240/28

We'll take the 192.168.2.192/28 for our subnet to hold 10 hosts. On the exam, the other 2 subnets could be listed as possible answers also. The answers we choose are:

192.168.2.0/25
192.168.2.128/26
192.168.2.192/28

This doesn't mean these answers will be listed on the exam. They might list 192.168.2.128/25 as the right answer, which means they wanted you to subnet the 192.168.2.0 subnet (the reverse of what we did here). In other words, VLSM questions usually have more than one way to solve. That's what we mean when we said that VLSM questions tend to be "open ended". There are always more than one right answer, and more than one way to arrive at the right answers. **On the exams, you have to look carefully at all the multiple choice answers!** You need to know all the possible answers, and see all the wrong answers. Also, all the answers you choose have got to make sense. Here are examples of some wrong answers, and combination of answers that don't make sense.

192.168.2.0/29
192.168.2.225/25
192.168.2.140/24
192.168.2.128/25
192.168.2.192/27

For example, why would you choose 192.168.2.128/25 **and** 192.168.2.192/27 together? The first subnet has 128 hosts, and the second subnet is only 64 hosts away (128 + 64 = 192)! You can't subnet like that. That combination of answers won't make sense.

Example 2.31:
We have a Class network of 200.113.14.0/25, and we want a subnet to hold **50 hosts**, another to **30 hosts**, and another to hold **2 hosts**.

We can see that 200.113.14.0/25 already has 2 subnets:

200.113.14.0/25
200.113.14.128/25

Let's work with 200.113.14.0/25, and save 200.113.14.128/25 for future use. With 200.113.14.0/25, we can subnet to /26. This makes the m = 1 and n = 6, so we get 2 subnets, each with 64 hosts.

200.113.14.0/26
200.113.14.64/26

We choose 200.113.14.0/26 for our 50 host subnet. For the 30 host subnet, we subnet to /27, so m = 1 and n = 5, so we get 2 subnets with 32 hosts each (actually 30 hosts).

200.113.14.0/26 (used on 50 host subnet)

200.113.14.64/27
200.113.14.96/27

We choose 200.113.14.64/27 for the 30 host subnet. For the 2 host subnet, we subnet to /30, so m = 3 and n = 2, so we get 8 subnets with 4 hosts each (actually 2 hosts).

200.113.14.0/26	(used on 50 host subnet)
200.113.14.64/27	(used on 30 host subnet)
200.113.14.96/30	
200.113.14.100/30	
200.113.14.104/30	
200.113.14.108/30	
200.113.14.112/30	
200.113.14.116/30	
200.113.14.120/30	
200.113.14.124/30	

We'll choose the 200.113.14.96/30 subnet, and use the others in the future. Why do we need these small /30 subnets? Because they are used on point-to-point links between routers. You use VLSM so you don't have to waste a large subnet, like /26, just for a point-to-point link which only needs 2 IP addresses. That is why on network diagrams, you usually see routers connected together on /30 subnets. Notice that 200.113.14.128 is the next network address, and it's the 200.113.14.128/25 we wanted to save for future use. So our answers are:

200.113.14.0/26
200.113.14.64/27
200.113.14.96/30

Again, on the exams, they might list other correct answers. You have to visualize the address ranges in your head. At the end of this chapter are some address range questions to practice on. The faster you are at quickly finding address ranges, the faster you'll be at solving VLSM questions.

We end now with a Class B and Class A example.

Example 2.32:
We have Class B network 172.16.0.0/20. We need subnets to hold **400 hosts**, **100 hosts**, **45 hosts**, and **10 subnets** to hold **2 hosts**.

To hold 400 hosts, we need a subnet that can hold 512 hosts, or a /23, since n = 9 means $2^n = 2^9 = 512$ hosts. Since we went from /20 to /23, we get 3 more bits turned on, so m = 3 and $2^m = 2^3 = 8$ subnets. The /23 is a 255.255.254.0 subnet mask, to the increment is 256 - 254 = 2. So we choose this as our first subnet 172.16.0.0/23 to hold the 400 hosts.

172.16.0.0/23	(used on 400 host subnet)
172.16.2.0/23	
172.16.4.0/23	
172.16.6.0/23	
172.16.8.0/23	
172.16.10.0/23	
172.16.12.0/23	
172.16.14.0/23	

Notice that the next subnet (not shown above) is 172.16.16.0/20. Remember, that we originally have a subnetted Class B network with 172.16.0.0/20. This makes sense since /20 is a 255.255.240.0 subnet mask, and the increment is 16.

Let's choose 172.16.2.0 and make it a /25 to hold 100 hosts.
Let's choose 172.16.8.0 and make it a /26 to hold 45 hosts. Going from /23 to /26 means m = 3, 2^3 = 8 subnets.

172.16.8.0/26 (used on 45 host subnet)
172.16.8.64/26
172.16.8.128/26
172.16.8.192/26
172.16.9.0/26
172.16.9.64/26
172.16.9.128/26
192.16.9.192/26

Let's choose the 192.16.9.192 and make it /30 to hold 2 hosts. Going from /26 to /30 means m = 4, 2^4 = 16 subnets. So we will use 10 out of the 16 subnets.

192.16.9.192/30
192.16.9.196/30
192.16.9.200/30
192.16.9.204/30
192.16.9.208/30
192.16.9.212/30
192.16.9.216/30
192.16.9.220/30
192.16.9.224/30
192.16.9.228/30

So the subnets we chose were:

172.16.0.0/23	(to hold 400 hosts)
172.16.2.0/25	(to hold 100 hosts)
172.16.8.0/26	(to hold 45 hosts)
192.16.9.192/30	(to hold 2 hosts)
192.16.9.196/30	"
192.16.9.200/30	"
192.16.9.204/30	"
192.16.9.208/30	"
192.16.9.212/30	"
192.16.9.216/30	"
192.16.9.220/30	"
192.16.9.224/30	"
192.16.9.228/30	"

As you can see, there are many possibilities. On job interviews, you need to show them you understand that there are many subnets to choose from.

Example 2.33:
we have a Class A network 10.16.0.0/14 network. We want **2 subnets** to hold **4000 hosts**, and **2 subnets** to hold **2000 hosts**.

Experience tells me that, $2^{12} = 4096$ ($2^{10} = 1024$, and $2^{14} = 2^2 \times 2^{10} = 4 \times 1024$), so a /20 network will hold 4000 hosts. Going from /14 to /20 means m = 6, so there will be $2^6 = 64$ subnets. A /20 is 255.255.240.0 subnet mask, so the increment is 16. We will list some of the subnets here…

10.16.0.0/20 (used on 4000 host subnet)
10.16.16.0/20
10.16.32.0/20
10.16.48.0/20
10.16.64.0/20
10.16.80.0/20
10.16.96.0/20

We'll use 10.16.0.0/20 and 10.16.16.0/20 for the 2 subnets holding 4000 hosts. Let's choose 10.16.32.0 and change it to /21. From /20 to /21, we get n = 11, or $2^{11} = 2048$ hosts. The m = 1, so we get 2 subnets. That's all we needed. A /21 subnet mask is 255.255.248.0, and the increment is 8. So we get these 2 subnets:

10.16.32.0/21
10.16.40.0/21

So we choose these subnets to satisfy our requirements:

10.16.0.0/20	(to hold 4000 hosts)
10.16.16.0/20	(to hold 4000 hosts)
10.16.32.0/21	(to hold 2000 hosts)
10.16.40.0/21	(to hold 2000 hosts)

Below are exercises for practice. For more exercises, go to this free website: http://www.subnettingquestions.com/custom/bren/ .

2.10 Exercises

Find the subnet network address given the following IP addresses and subnet masks:

1. 158.12.33.200/27
2. 135.124.61.77 , 255.255.254.0
3. 149.50.85.193/28
4. 216.83.170.244 , 255.255.255.224
5. 168.23.103.205/27

Find the network address for Subnet N:

6. Class C network 192.168.4.0 has been subnetted by 255.255.255.224. What is the network address for Subnet 0? 192.168.4.0

7. Class C network 201.154.19.0 has been subnetted by 255.255.255.248. What is the network address for Subnet 7? 201.154.19.56

8. Network 214.98.31.0/24 has been subnetted into /26 networks. What is the network address for Subnet 3? 214.98.31.192

9. Network 198.14.23.0/24 has been subnetted into /30 networks. What is the network address for Subnet 21? 198.14.23.84

Find the first usable IP address in the subnet that these host IP's belong to:

10. 146.19.129.143/21

11. 155.126.31.225/25

12. 185.29.104.201/23

13. 159.255.220.70 , 255.255.255.192

Find the last usable IP address in the following subnets:

14. 134.141.202.0 255.255.254.0

15. 165.164.58.128/28

16. 7.33.64.0/20

17. 167.53.128.0/20

18. 135.134.116.0/22

19. 144.142.99.192/26

20. 195.218.43.232/29

21. 220.229.5.32 255.255.255.224

Find the broadcast IP address of the following subnets:

22. 169.13.229.16/28

23. 158.253.115.0 , 255.255.255.0

24. 47.188.176.0/20

25. 161.15.100.0/26

26. 190.202.158.0/24

Find the range of usable IP addresses in the subnet these IP's belong to:

27. 152.4.162.245/28

28. 150.158.3.117/25

29. 184.11.71.123 , 255.255.255.0

Find the number of subnets and usable hosts per subnet in these subnetted major networks:

30. 181.2.0.0/26

31. 182.10.0.0/21

32. 129.122.0.0/22

33. 191.115.0.0/28

34. 205.86.174.0/26

35. 191.115.0.0/28

36. 179.88.0.0 , 255.255.254.0

37. 172.64.0.0/20

Given a Class A, B or C major network, what subnet mask do you need to subnet the network and satisfy the requirements?

38. 3.0.0.0	You need 2500 subnets and 3000 hosts per subnet.
39. 185.136.0.0	You need 600 subnets and 60 hosts per subnet.
40. 139.125.0.0	You need 29 subnets and 1400 hosts per subnet.
41. 144.97.0.0	You need 100 subnets and 300 hosts per subnet.
42. 206.252.54.0	You need 2 subnets and 90 hosts per subnet.
43. 172.29.0.0	You need 400 subnets and 90 hosts per subnet.
44. 182.126.0.0	You need 110 subnets and 400 hosts per subnet.

VLSM questions

In each question, find the best 3 combinations of subnets. Subnet for efficiency. Zero and broadcast subnets are allowed.

45. You have network 192.168.1.0/24, and you want subnets to contain 100 hosts, 25 hosts and 6 hosts. Which combination of 3 subnets can you use?

a. 192.168.1.0/28 b. 192.168.1.0/27 c. 192.168.2.64/26

d. 192.168.1.128/25 e. 192.168.1.96/29 f. 192.168.1.40/25

46. You have network 150.210.0.0/16, and you want subnets to contain 2000 hosts, 500 hosts and 100 hosts. Which combination of 3 subnets can you use?

a. 150.210.128.0/18 b. 150.210.0.0/17 c. 150.210.16.0/21

d. 150.210.32.0/23 e. 150.210.64.0/28 f. 150.210.34.0/25

47. You have network 125.0.0.0/8, and you want subnets to contain 1000 hosts, 200 hosts and 100 hosts. Which combination of 3 subnets can you use?

a. 125.0.4.0/22b. 125.110.5.0/11 c. 125.0.32.0/24

d. 125.0.31.0/24 e. 125.0.125.128/26 f. 125.0.33.128/25

48. You have network 217.84.19.0/24, and you want subnets to contain 30 hosts, 10 hosts and 2 hosts. Which combination of 3 subnets can you use?

a. 217.84.19.24/25 b. 217.84.19.64/27 c. 217.84.19.128/28

d. 217.84.19.0/30 e. 217.84.19.160/30 f. 217.84.19.16/25

49. You have network 180.32.64.0/23, and you want subnets to contain 120 hosts, 50 hosts and 10 hosts. Which combination of 3 subnets can you use?

a. 180.32.64.128/25 b. 180.32.65.0/26 c. 180.32.64.0/24

d. 180.32.64.64/25 e. 180.32.64.192/27 f. 180.32.65.64/28

50. You have network 152.227.160.0/21, and you want subnets to contain 500 hosts, 80 hosts and 5 hosts. Which combination of 3 subnets can you use?

a. 152.227.160.224/24b. 152.227.160.128/25 c. 152.227.160.32/30

d. 152.227.162.0/23 e. 152.227.166.0/25 f. 152.227.167.128/29

Chapter 3 CIDR, Supernetting and Classless Addressing

3.1 CIDR

The explosive growth of the internet in the mid-to-late 1990's increased the size of the world's routing tables. Large routing tables consumed more memory, CPU and bandwidth. New routers and routing protocols had to support CIDR and supernetting to keep the tables at an acceptable size. CIDR stands for Classless Interdomain Routing, and basically means that routers can group blocks of classful networks into a larger network (with smaller subnet mask). For example, instead of knowing 32 Class C routes, a router only needs to know about a single /19 route.

Example 3.1: 32 Class C routes grouped into a single /19 route…

```
201.166.32.0/24
201.166.33.0/24
201.166.34.0/24
201.166.35.0/24        >>>>>>>>>>>>>>>>>>>>>   201.166.32.0/19
.
.
201.166.63.0/24
```

Example 3.2: 16 Class B routes grouped into a single /12 route…

```
180.16.0.0/16
180.17.0.0/16
180.18.0.0/16
180.19.0.0/16          >>>>>>>>>>>>>>>>>>>>>   180.16.0.0/12
.
.
180.31.0.0/16
```

As you can see, CIDR makes the routing tables smaller, and saves router CPU, memory and bandwidth.

CIDR is a specific form of supernetting applied to blocks of classful networks. Internet authorities use CIDR to allocate the last remaining blocks of IP Version 4 addresses to internet service providers. Some of the rules for CIDR are (1) the number of classful networks to be supernetted have to be a power of 2 (e.g., 8, 16, 32, 64) and (2) the last octet of the first network has to be an even number.

3.2 Supernetting and Classless addressing

Supernetting basically means the grouping (summarizing) of subnets into a larger network (with smaller subnet mask). For example, instead of knowing 4 routes to Class C /24 networks, the router can just send packets to a single /22 route:

Example 3.3:

```
192.168.4.0/24 \
192.168.5.0/24  | ----------------------------------> 192.168.4.0/22
192.168.6.0/24  |
192.168.7.0/24 /
```

In this example, any packets going to the 192.168.4.0/24, 192.168.5.0/24, 192.168.6.0/24 or 192.168.7.0/24 networks, will simply go to 192.168.4.0/22. When more and more routes are supernetted, the routing table shrinks to a smaller size. We also say that the 4 routes are "summarized" into 1 route. Terms like CIDR, supernetting, and route summarization relate to similar ideas.

Example 3.4: This group of /18 networks…

184.64.0.0/18
184.64.64.0/18
184.64.128.0/18
184.64.192.0/18

…can be grouped to this larger network: 184.64.0.0/16

As you can see, supernetting is basically the reverse of subnetting! You can use the network increment, that we learned in Chapters 1 and 2, and apply it to a group of networks to find a supernet.

Are there multiple supernets possible? In the previous example, besides 184.64.0.0/16, isn't 184.0.0.0/8 a possible supernet? Yes, 184.0.0.0/8 is definitely a supernet. There are also other possibilities, such as 184.0.0.0/5 or 182.0.0.0/6. Is there a right answer? There is no single right answer, but on the certification exams, you will be probably be asked to find the closest supernet, the one which summarizes only the given specific networks and nothing else. The supernet 184.64.0.0/16 is a better answer than 184.0.0.0/8 because 184.0.0.0/8 is a supernet for many more networks besides the /18 networks in the previous example. In any case, read the question carefully before choosing the right answer(s).

Supernetting implies addressing has to be "**classless**", meaning that the subnet masks do not have to follow Class A, B, or C default masks, or increase to the right of the default mask. With classless addressing, a Class C address 204.114.17.8 can have a /18 subnet mask.

Before classless addressing, only addresses and subnet masks like these were allowed…

201.44.87.39/24
192.168.1.0/29
176.91.85/14/16
180.33.98.14/18
10.10.10.10/8
95.216.222.19/14

With classless addressing, any IP address can have any subnet mask, so these are also allowed…

205.174.16.2/17
192.168.2.0/21
14.92.74.223/6
139.72.99.228/14
172.32.65.10/10
199.14.25.89/13

All modern routers now support classless addressing. Classless addressing means we do not have care about whether an IP address is in a Class A, B or C network. We are free to use whatever subnet mask we want. Classless addressing and supernetting is using VLSM without classful boundaries. The IP addresses themselves can still be categorized into classes, since a Class A is just any IP with a 0 as the left most bit, Class B is 10 in the left most bits, and Class C is 110 in the left most bits (see Chapter 1). But the subnet masks are no longer /8, /16, /24, and they no longer have to be "to the RIGHT" of their default subnet mask.

As we learned in Chapter 1, there are over 536 million unusable IP addresses in the 32-bit address space because of the Class A, B and C organization. By getting rid of classes, we can reclaim many of those previously unusable IP addresses. In other words, 201.15.21.0/24 is no longer restricted to 254 usable host addresses. Supernetting allows us to grab more host IP's, by allowing subnets like 201.0.0.0/8. This helps to alleviate the shortage of IP Version 4 addresses. Sooner or later, we will run out of 32-bit IP addresses, and will have to use 128-bit IP Version 6 addressing.

Example 3.5: We can reclaim more "lost" IP addresses for hosts by supernetting and classless addressing:

201.15.21.0/24 → 256 addresses

201.0.0.0/8 → 16777216 addresses

<u>3.3 Route Summarization</u>

Route summarization refers to the grouping of a set of networks or routes into a single network or route. One obvious summary route is the default route 0.0.0.0/0. In a routing table, you may have a few static routes, some routes learned by routing protocols like OSPF or EIGRP, and you may have a default route that is used for all other unknown destinations. Route summarization really is a form of supernetting specifically done by routing protocols.

Example 3.6: This group of /29 networks…

192.168.6.192/29
192.168.6.200/29
192.168.6.208/29
192.168.6.216/29

…can be summarized into 1 network: 192.168.6.192/27

Example 3.7: This group of /18 networks…

184.64.0.0/18
184.64.64.0/18
184.64.128.0/18
184.64.192.0/18

…can be summarized into 1 network: 184.64.0.0/16

Example 3.8: This group of /14 networks…

56.32.0.0/14
56.36.0.0/14
56.40.0.0/14
56.44.0.0/14
56.48.0.0/14
56.52.0.0/14
56.56.0.0/14

56.60.0.0/14

…can be summarized into 1 network: 56.32.0.0/11

Route summarization is necessary on routers because it makes the routing tables smaller, thus saving CPU, memory and bandwidth resources. Also, if there's a problem like routes flapping (going up and down at a high frequency), summarization prevents unnecessary routing updates to flood the network. One thing to realize, however, is that a specific route summary is for the benefit of other routers. The router doing the summarization must still have the group of specific routes in its own routing table. The summarization is in the routing updates sent to other routers. It is other routers who will benefit from the summary route and smaller routing table. Of course, if all routers are doing summarization, and updating each other with summary routes, then all routers will have smaller routing tables and benefit from each other's summarizations.

Summarization and supernetting is not really that much different than what we've learned in Chapters 1 and 2. They are just the reverse of subnetting. Subnettting and supernetting/summarization are opposites of each other.

Basically, all subnets or networks are summarizations of /32 host addresses. For example, a default Class C /24 network is just a summarization of 256 possible /32 addresses (of which 254 are usable by hosts). A Class B /16 network is a summarization of 65536 /32 addresses, and a Class A /8 network is a summarization of 16,777,216 /32 addresses. The largest possible summary network is 0.0.0.0/0, which is the default route used by routers. Spatially, you can think of the IP address space as like this…

/0 network --------> subnets/supernets like /3, /8, /16, /19, /24, /27 --------> individual /32 addresses

On routers, a /32 address is also called a "host address" or unicast address, for good reason. Routers need subnets, supernets and route summarizations to minimize CPU, memory and bandwidth resources. If these network strategies did not exist, the world's routers would have hundreds of millions of /32 addresses in flat non-hierarchical routing tables, which would consume an impossible amount of CPU, memory and bandwidth. This would be like if the post offices did not use postal codes and each one had to lookup every street address in the country to deliver the mail!

On the certification exams, you will usually be asked to find the closest supernet among multiple answer choices. In other words, the supernet should only summarize the given block of specific routes, not anything else. For example, in **Example 3.8**, you would choose 56.32.0.0/11 as the answer, instead of 56.0.0.0/8.

3.4 Exercises

For each group of networks, choose the closest supernet.

1. Block of 4 subnets

194.75.96.32/29
194.75.96.40/29
194.75.96.48/29
194.75.96.56/29

a. 194.75.96.0/30 b. 194.75.96.32/30 c. 194.75.96.32/27

d. 194.75.96.0/23 e. 194.75.96.64/25 f. 194.75.128.0/27

2. Block of 32 subnets

201.63.64.0/24
201.63.65.0/24

201.63.66.0/24
201.63.67.0/24
.
.
201.63.94.0/24
201.63.95.0/24

a. 201.63.64.128/18 b. 201.63.32.0/19 c. 201.63.64.128/29

d. 201.63.64.0/17 e. 201.63.64.0/19 f. 201.63.64.32/24

3. Block of 4 subnets

64.16.0.0/16
64.17.0.0/16
64.18.0.0/16
64.19.0.0/16

a. 64.16.0.0/8 b. 64.0.0.0/20 c. 64.16.0.0/12

d. 64.16.2.0/18 e. 64.16.32.0/16 f. 64.16.0.0/14

4. Block of 8 subnets

20.0.0.0/9
20.128.0.0/9
21.0.0.0/9
21.128.0.0/9
22.0.0.0/9
22.128.0.0/9
23.0.0.0/9
23.128.0.0/9

a. 20.0.0.0/4 b. 20.0.0.0/6 c. 20.0.0.0/12

d. 20.128.0.0/6 e. 22.16.0.0/8 f. 21.0.0.0/7

5. Block of 16 subnets

137.0.0.0/16
137.16.0.0/16
137.32.0.0/16
137.48.0.0/16
.
.
137.192.0.0/16
137.208.0.0/16
137.224.0.0/16
137.240.0.0/16

a. 137.32.0.0/14 b. 137.16.0.0/18 c. 137.0.0.0/15

d. 137.0.0.0/8 e. 132.0.0.0/10 f. 137.16.0.0/6

6. Block of 8 host addresses

194.225.176.80/32
194.225.176.81/32
194.225.176.82/32
194.225.176.83/32
194.225.176.84/32
194.225.176.85/32
194.225.176.86/32
194.225.176.87/32

a. 194.225.176.80/27 b. 194.225.176.80/24 c. 194.225.176.88/26

d. 194.225.176.80/16 e. 194.225.176.0/28 f. 194.225.176.80/28

Chapter 4 Advance VLSM and Subnetting Problems

4.1 Class A and B Major Networks: Calculating Subnet Network Address for Subnet N

Given a Class A or B major network that has been subnetted, how do we find Subnet N? For example, if we had 121.0.0.0/27, how do we find the network address for Subnet 438491? As you can see, for large networks such as Class A and B, the value N can be a very big number. Questions like this for Class A and B networks are probably not on certification exams, but they might be asked in job interviews, so we practice them here.

As we learned in Chapter 1, the network addresses relate to the extra bits gained by the subnet mask.

Example 4.1: 143.217.0.0/29 What is the network address for Subnet 5803?

This is a Class B network, so the default subnet mask is /16. With a subnet mask /29, there are 2^{13} = 8192 subnets created. N = 5803.

```
10001111.11011001.xxxxxxxx.xxxxxxxx ← all possible IP addresses in 143.217.0.0/16
11111111.11111111.11111111.11111000 ← new subnet mask /29
---------------------------------------------------
10001111.11011001.xxxxxxxx.xxxxx000 ← all possible network addresses
```

The network addresses are derived from doing a Logical AND operation between every possible IP address in the original 143.217.0.0/16 block and the new subnet mask. The value of Subnet N is related directly to the value of N.
All subnet network addresses:

```
00000000.00000000   N = 0   0.0
00000000.00001000   N = 1   0.8
00000000.00010000   N = 2   0.16
00000000.00011000   N = 3   0.24
00000000.00100000   N = 4   0.32
.
.
.
11111111.11101000   N = 8189       255.232
11111111.11110000   N = 8190       255.240
11111111.11111000   N = 8191       255.248
```

Notice that the extra subnet bits in the shaded area form a binary number that is equal to the value N. After converting the decimal value N to binary, all we need to do is add the 3 zeroes on the right, add the dots "." to create the octets, and convert the binary octets to decimal.

Convert N = 5803 to binary = 1011010101011

Add 3 zeroes on the right: 1011010101011000

Add the dots and octets from RIGHT to LEFT: .10110101.01011000

Convert octets to decimal: .181.88

So the network address for Subnet 5803 is 143.217.181.88.

Answer: 143.217.181.88/29

Example 4.2: 73.0.0.0/30 What is the network address for Subnet 4194299?

This is a Class A network, so the default subnet mask is /8. With a subnet mask /30, there are 2^{22} = 4194304 subnets created. N = 4194299.

```
01001001.xxxxxxxx.xxxxxxxx.xxxxxxxx ← all possible IP addresses in 73.0.0.0/8
11111111.11111111.11111111.11111100 ← new subnet mask /30
-----------------------------------------------
01001001.xxxxxxxx.xxxxxxxx.xxxxxx00 ← all possible network addresses
```

The network addresses are derived from doing a Logical AND operation between every possible IP address in the original 143.217.0.0/8 block and the new subnet mask. The value of Subnet N is related directly to the value of N.
All subnet network addresses:

```
00000000.00000000.00000000     N = 0   0.0.0
00000000.00000000.00000100     N = 1   0.0.4
00000000.00000000.00001000     N = 2   0.0.8
00000000.00000000.00001100     N = 3   0.0.12
.
.
11111111.11111111.11110000     N = 4194300   255.255.240
11111111.11111111.11110100     N = 4194301   255.255.244
11111111.11111111.11111000     N = 4194302   255.255.248
11111111.11111111.11111100     N = 4194303   255.255.252
```

Convert N = 4194299 to binary = 1111111111111111111011

Add 2 zeroes on the right: 111111111111111111101100

Add the dots and octets from RIGHT to LEFT: .11111111.11111111.11101100

Convert octets to decimal: .255.255.236

So the network address for Subnet 4194299 is 73.255.255.236

Answer: 73.255.255.236/30

This procedure of converting the N value to binary, then forming octets, and converting the octets to decimal numbers also works for Class C networks. But for Class C networks, it is faster to use the **Last octet = N × (increment)** formula.

4.2 Advance VLSM Problems

In Chapters 1 and 2, we studied VLSM questions which asked for a subnet network address for each group of hosts. More advance VLSM questions deal with finding multiple subnets for each requirement. This is basically a task of organizing and planning subnet address ranges. Use the network increment to get the address ranges.

These types of questions are unlikely to be on certification exams. They are included here for you to gain more experience.

Example 4.3: Suppose we start with a major Class C network 192.168.1.0/24, and we need the following…

1 subnets with 100 hosts/subnet
2 subnets with 30 hosts/subnet
6 subnets with 6 hosts/subnet

We know we have this stituation…

192.168.1.0/24	Current Subnet
192.168.2.0/24	Next Subnet

We start with the largest 100 hosts network requirement, and subnet to /25. Going from /24 to /25, we gain 2 subnets, and use one and subnet the other to /27.

192.168.1.0/25	(use for 100 host subnet)
192.168.1.128/25	(subnet to /27)

The /27 takes care of the 30 host subnets. Going from /25 to /27, we gain 4 subnets. We use 2 of them, and subnet 2 others to /29.

192.168.1.128/27	(use for 30 host subnet)
192.168.1.160/27	(use for 30 host subnet)
192.168.1.192/27	(subnet to /29)
192.168.1.224/27	(subnet to /29)

The 2 /29 subnets takes care of the 6 host subnets. Going from /27 to /29 means we gain 4 subnets each, for a total of 8 subnets. We use 6 of them for the 6 hosts subnet, and leave 2 subnets for future use.

192.168.1.192/29	(use for 6 host subnet)
192.168.1.200/29	(use for 6 host subnet)
192.168.1.208/29	(use for 6 host subnet)
192.168.1.216/29	(use for 6 host subnet)
192.168.1.224/29	(use for 6 host subnet)
192.168.1.232/29	(use for 6 host subnet)
192.168.1.240/29	← leave free for future use
192.168.1.248/29	← leave free for future use

Here's a complete list of the subnets we chose…

192.168.1.0/25	(use for 100 host subnet)
192.168.1.128/27	(use for 30 host subnet)
192.168.1.160/27	(use for 30 host subnet)
192.168.1.192/29	(use for 6 host subnet)
192.168.1.200/29	(use for 6 host subnet)
192.168.1.208/29	(use for 6 host subnet)
192.168.1.216/29	(use for 6 host subnet)
192.168.1.224/29	(use for 6 host subnet)
192.168.1.232/29	(use for 6 host subnet)

Other choices could've been made. There's no single right answer. What networks you choose depends on your environment, your goals, etc.

Example 4.4: Start with a major Class B network that has been subnetted to 180.14.0.0/20, and we need the following…

2 subnets with 500 hosts/subnet
11 subnets with 100 hosts/subnet
14 subnets with 50 hosts/subnet
12 subnets with 2 hosts/subnet

We know we have this situation…

180.14.0.0/20	Current Subnet
180.14.16.0/20	Next Subnet

We will get the largest 500 host subnets first. We know we need some /23 networks, and we can get 2 of them for the 500 host subnets, and subnet the rest.

180.14.0.0/23	(use for 500 host subnet)
180.14.2.0/23	(use for 500 host subnet)
180.14.4.0/23	(subnet to /25)
180.14.6.0/23	(subnet to /25)
180.14.8.0/23	(subnet to /25)
180.14.10.0/23	(subnet to /26)
180.14.12.0/23	(subnet to /26)
180.14.14.0/23	← leave free for future use

The 100 host subnets require a /25. We take 3 /23 subnets and subnet to /25. Each /25 gets 4 subnets.

180.14.4.0/25	(use for 100 host subnet)
180.14.4.128/25	(use for 100 host subnet)
180.14.5.0/25	(use for 100 host subnet)
180.14.5.128/25	(use for 100 host subnet)

180.14.6.0/25	(use for 100 host subnet)
180.14.6.128/25	(use for 100 host subnet)
180.14.7.0/25	(use for 100 host subnet)
180.14.7.128/25	(use for 100 host subnet)

180.14.8.0/25	(use for 100 host subnet)
180.14.8.128/25	(use for 100 host subnet)
180.14.9.0/25	(use for 100 host subnet)
180.14.9.128/25	← leave free for future use

We use 11 of the 12 /25 subnets. The 50 host subnets require a /26, and we use 2 of the /23 subnets listed above. We get 8 subnets each, for a total of 16 /26 subnets.

180.14.10.0/26	(use for 50 host subnet)
180.14.10.64/26	(use for 50 host subnet)

180.14.10.128/26	(use for 50 host subnet)
180.14.10.192/26	(use for 50 host subnet)
180.14.11.0/26	(use for 50 host subnet)
180.14.11.64/26	(use for 50 host subnet)
180.14.11.128/26	(use for 50 host subnet)
180.14.11.192/26	(use for 50 host subnet)

180.14.12.0/26	(use for 50 host subnet)
180.14.12.64/26	(use for 50 host subnet)
180.14.12.128/26	(use for 50 host subnet)
180.14.12.192/26	(use for 50 host subnet)
180.14.13.0/26	(use for 50 host subnet)
180.14.13.64/26	(use for 50 host subnet)
180.14.13.128/26	(subnet to /30)
180.14.13.192/26	← leave free for future use

We use 14 of the /26 subnets, and subnet one of them to /30. The /30 gives us 16 subnets, and we use 12 of them, and leave the rest to future use. We list some of the 16 subnets here…

180.14.13.128/30	(use for 2 host subnet)
180.14.13.132/30	(use for 2 host subnet)
180.14.13.136/30	(use for 2 host subnet)
180.14.13.140/30	(use for 2 host subnet)
.	
.	
180.14.13.168/30	(use for 2 host subnet)
180.14.13.172/30	(use for 2 host subnet)
180.14.13.176/30	← leave free for future use
.	
180.14.13.184/30	← leave free for future use
180.14.13.188/30	← leave free for future use

Here's a list of all the subnets we chose from above…

180.14.0.0/23	(use for 500 host subnet)
180.14.2.0/23	(use for 500 host subnet)
180.14.4.0/25	(use for 100 host subnet)
180.14.4.128/25	(use for 100 host subnet)
180.14.5.0/25	(use for 100 host subnet)
180.14.5.128/25	(use for 100 host subnet)
180.14.6.0/25	(use for 100 host subnet)
180.14.6.128/25	(use for 100 host subnet)
180.14.7.0/25	(use for 100 host subnet)
180.14.7.128/25	(use for 100 host subnet)
180.14.8.0/25	(use for 100 host subnet)
180.14.8.128/25	(use for 100 host subnet)
180.14.9.0/25	(use for 100 host subnet)
180.14.10.0/26	(use for 50 host subnet)
180.14.10.64/26	(use for 50 host subnet)
180.14.10.128/26	(use for 50 host subnet)
180.14.10.192/26	(use for 50 host subnet)

180.14.11.0/26	(use for 50 host subnet)
180.14.11.64/26	(use for 50 host subnet)
180.14.11.128/26	(use for 50 host subnet)
180.14.11.192/26	(use for 50 host subnet)
180.14.12.0/26	(use for 50 host subnet)
180.14.12.64/26	(use for 50 host subnet)
180.14.12.128/26	(use for 50 host subnet)
180.14.12.192/26	(use for 50 host subnet)
180.14.13.0/26	(use for 50 host subnet)
180.14.13.64/26	(use for 50 host subnet)
180.14.13.128/30	(use for 2 host subnet)
180.14.13.132/30	(use for 2 host subnet)
180.14.13.136/30	(use for 2 host subnet)
180.14.13.140/30	(use for 2 host subnet)
180.14.13.144/30	(use for 2 host subnet)
180.14.13.148/30	(use for 2 host subnet)
180.14.13.152/30	(use for 2 host subnet)
180.14.13.156/30	(use for 2 host subnet)
180.14.13.160/30	(use for 2 host subnet)
180.14.13.164/30	(use for 2 host subnet)
180.14.13.168/30	(use for 2 host subnet)
180.14.13.172/30	(use for 2 host subnet)

As you can see, we could've used other combination of subnets. Most of the time, you should choose contiguous subnets, and allow room for growth.

4.3 Exercises

A major network has been subnetted. Find the network address for Subnet N.

1. 132.56.0.0/29	find subnet 6977
2. 163.203.0.0/26	find subnet 1015
3. 116.0.0.0/20	find subnet 3977
4. 15.0.0.0/27	find subnet 379457
5. 1.0.0.0/30	find subnet 4194302

Answers to Chapter Exercises

Chapter 1

1. 00000000
2. 00000110
3. 00001110
4. 00011001
5. 00110001
6. 00111111
7. 01100010
8. 01101011
9. 01111010
10. 10001000
11. 10100011
12. 11001000
13. 11011011
14. 11110011
15. 11111101
16. 100000000
17. 100011101
18. 100101110
19. 111100011

20. 51
21. 0
22. 255
23. 227
24. 16
25. 252
26. 87
27. 6
28. 240
29. 103
30. 207
31. 61
32. 225
33. 158
34. 256
35. 8
36. 512
37. 16
38. 32
39. 32
40. 6
41. 126
42. 2046
43. 145.126.203.255, 145.126.204.255, 145.126.205.255
44. 188.74.255.255, 188.75.255.255, 188.76.255.255

45. 206.19.16.0
206.19.16.32

206.19.16.64
206.19.16.96
increment = 32

46. 61.231.0.0
61.231.16.0
61.231.32.0
61.231.48.0
61.231.64.0
increment = 16

47. 124.220.0.0
124.220.32.0
124.220.64.0
124.220.96.0
124.220.128.0
increment = 32

Chapter 2

1. 158.12.33.192
2. 135.124.60.0
3. 149.50.85.192
4. 216.83.170.224
5. 168.23.103.192
6. 192.168.4.0
7. 201.154.19.56
8. 214.98.31.192
9. 198.14.23.84
10. 146.19.128.1
11. 155.126.31.129
12. 185.29.104.1
13. 159.255.220.65
14. 134.141.203.254
15. 165.164.58.142
16. 7.33.79.254

17. 167.53.143.254
18. 135.134.119.254
19. 144.142.99.254
20. 195.218.43.238
21. 220.229.5.62
22. 169.13.229.31
23. 158.253.115.255
24. 47.188.191.255
25. 161.15.100.63
26. 190.202.158.255
27. 152.4.162.241 to 152.4.162.254
28. 150.158.3.1 to 150.158.3.126
29. 184.11.71.1 to 184.11.71.254
30. 1024 subnets, 62 hosts
31. 32 subnets, 2046 hosts
32. 64 subnets, 1022 hosts
33. 4096 subnets, 14 hosts
34. 4 subnets, 62 hosts
35. 4096 subnets, 14 hosts
36. 128 subnets, 510 hosts
37. 16 subnets, 4094 hosts
38. 255.255.240.0
39. 255.255.255.192
40. 255.255.248.0
41. 255.255.254.0
42. 255.255.255.128

43. 255.255.255.128

44. 255.255.254.0

45. d, b, e, 192.168.1.128/25, 192.168.1.0/27, 192.168.1.96/29

46. c, d, f 150.210.16.0/21, 150.210.32.0/23, 150.210.34.0/25

47. a, c, f 125.0.4.0/22, 125.0.32.0/24, 125.0.33.128/25

48. b, c, e 217.84.19.64/27, 217.84.19.128/28, 217.84.19.160/30

49. a, b, f 180.32.64.128/25, 180.32.65.0/26, 180.32.65.64/28

50. d, e, f 152.227.162.0/23, 152.227.166.0/25, 152.227.167.128/29

Chapter 3

1. c. 194.75.96.32/27

2. e. 201.63.64.0/19

3. f. 64.16.0.0/14

4. b. 20.0.0.0/6

5. d. 137.0.0.0/8

6. f. 194.225.176.80/28

Chapter 4

1. 132.56.218.8
2. 163.203.253.192
3. 116.248.144.0
4. 15.185.72.32
5. 1.255.255.248