Subnetting Guide

IP Class Identification

For most questions, you will need to identify the class of the IP and its default subnet mask along with the amount of bits. **Remember:** Each octet in a subnet mask is 8 bits.

Class A: 1-126 | SNM: 255.0.0.0 aka (/8) |8bit Network 24bit Host

Class B: 128-191 | SNM: 255.255.0.0 aka (/16) | 16bit Network 16bit Host

Class C: 192-223 | SNM: 255.255.255.0 aka (/24) | 24bit Network 8bit Host

Prefix Notation: I.E. (/24) | is basically a shorter way to write its subnet mask 255.255.255.0. They are both equal.

Decimal to Binary

Always start by writing down our binary baseline numbers (I just call it that for my own reference)

128 64 32 16 8 4 2 1

The idea is: can you subtract the given decimal from the first binary number (231-128)? Then its remainders (103-64) until you reach zero? If you can, put the number 1 down under the number you are subtracting. If you can't put the number 0 down. Here is an example below.

Convert 231 to Binary

128 | 64 | 32 | 16 | 8 | 4 | 2 | 1

231-**128**=103-**64**=39-**32**=7. (See how the bold numbers match our binary guide?)

When we get to 39-32 you will notice that the remainder is 7. You cannot subtract **16** from 7 or **8** from 7 (i.e. 7-16 or 7-8) as it will result in a negative number (you cannot have negative numbers) so we need to put zeroes under 16 and 8. So to continue we subtract 4 from 7. Since we can do this subtraction 7-4 put the number 1 under 4.

231-**128**=103-**64**=39-**32**=7-**4**=3-**2**=1-**1**=0

The end result is always 0.

The binary number for the decimal 231=11100111

Convert 192 to Binary

192-**128**=64-**64**=0

You may be thinking to yourself, ok now that I have subtracted the remainder I have nothing left! Well, that's ok! Put 0's for the rest of the numbers as there is nothing more to subtract and you're done.

128 | 64 | 32 | 16 | 8 | 4 | 2 | 1

1 1 0 0 0 0 0 0

The binary number for the decimal 192=11000000

Binary to Decimal

Always start by writing down our binary numbers 128 64 32 16 8 4 2 1 for reference.

When given a binary number, we will need to add only the numbers in the string above which have 1's. For example:

Convert The Binary Number 10101010

Start by writing down our base binary numbers. Then put the **given** binary numbers to convery under our baseline like this:

128 | 64 | 32 | 16 | 8 | 4 | 2 | 1

1 0 1 0 1 0 1 0

Then add **only** the numbers that have 1 under them so:

128 + 32 + 8 + 2 = 170

The decimal for 10101010=170

There you have it!

What is the IP address (dotted decimal) for this binary number: 11110000 00001111 10101010 01010101

128 | 64 | 32 | 16 | 8 | 4 | 2 | 1

1 1 1 1 0 0 0 0 = 128+64+32+16=240

0 0 0 0 1 1 1 1 = 8+4+2+1=15

1 0 1 0 1 0 1 0 = 128+32+8+1=170

0 1 0 1 0 1 0 1 = 64+16+4+1=85

The IP is 240.15.170.85

How Many Valid Subnets Do I Have?

To determine this we need to calculate 2 to the power of the number of **subnet** bits.

Here are the values given:

Network = 172.20.0.0 Subnet Mask = 255.255.255.0 (/24)

First we need to determine the **class** the IP address (reference first page). To do this, just look at the first octet.

The ip address given above is a class **B** IP address. Now we need to write the **default** mask for a class **B** IP and put it in binary.

Just know that anytime you see 255 the binary for that decimal is all 1's.

Default Class B Mask:

255.255.0.0 = 11111111 11111111 00000000 00000000

The Given Subnet Mask:

255.255.255.0 = 11111111 11111111 11111111 00000000

Now let's write both of these out because we need to compare them. The process of subnetting is about borrowing host bits. We never borrow **network bits**. The area where you see all 0's (**in red**) are your host bits. The area where they share all 1's (**in purple**) are your network bits. The 0's and 1's in **yellow** are your subnet bits.

11111111 11111111 00000000 00000000

11111111 11111111 11111111 00000000

So from the example above, you see that we have 8 subnet bits which we have borrowed. So to figure out how many valid subnets we have we need to calculate 2^8th power.

Here is a way to do it but it's always good to practice a faster way:

2x2=4+4=8+8=16+16=32+32=64+64=**128+128**=256

(power=>) 2nd 3rd 4th 5th 6th 7th 8th

We have **256** Valid Subnets

You might need to subtract 2 from the number of valid subnets depending on the configuration.

<u>If you see:</u>

No ip subnet-zero or the Routing Protocol is **Classful** like **RIPv1** or **IGRP** you will need to subtract 2

How Do I Determine the Number of Valid Host's?

To determine this we need to calculate 2 to the power of the number of **host** bits. Upon figuring out the answer we **always** subtract 2.

Here is the given subnet and mask. How many hosts are on this subnet?

150.50.50.0 /24

First, we need to know what class is this IP? It is **Class B**.

Now we need to write out the default network mask for that class and its binary equivalent. We also need to do the same for the mask of the **given** subnet in our question.

Default Mask: 255.255.0.0 (/16)

Binary Equiv= 11111111 11111111 00000000 00000000

Given Mask: 255.255.255.0

Binary Equiv= 11111111 11111111 11111111 00000000

Let's compare. In the previous example we identified the network, host, and subnet bits. Now we need to look at our host bits (in red).

11111111 11111111 00000000 00000000

11111111 11111111 11111111 00000000

So as you can see we have 8 host bits. Our calculation will be: 2 to the 8th power minus 2 =254

We have 254 **valid hosts**.

The reason we subtract 2 is because there are two ip's which are reserved and should not be used. It is the broadcast and network number.

How To Determine the Subnet Number of A Given IP Address?

When we are given the IP and subnet mask, we need to perform a Boolean AND operation. In order to do this we must make sure:

1. The IP and Subnet Mask is converted to binary.
2. Compare the binary conversions (ip/subnet) bit by bit.

If both bits are 1 the result of the Boolean AND is 1. If 0 is set for that bit on the subnet mask or IP or both, the result of the AND is 0. Here is an example.

IP =178.56.21.9 SN Mask=/24

Now we need to compare these binary numbers bit by bit. In the example below, I have bolded some bits in the first octet to show you how to get the results of the Boolean AND as mentioned previously. From there, do the rest.

1**0**11**00**1**0** 00111000 00010101 00001001 = Binary IP

1**1**11**1**1**1** 11111111 11111111 00000000 = Binary SN Mask

1**0**11**00**1**0** 00111000 00010101 00000000 = AND Result

When you are finished, convert the AND result back to decimal

1**0**11**00**1**0** 00111000 00010101 00000000

178 56 21 0

178.56.21.0 is the subnet that IP-178.56.21.9 belongs to.

Here is another:

What subnet is this IP on: 200.154.150.89 /27

Let's convert the IP to binary = 11001000 10011010 10010110 01011001

Now we need to know what is /27 in dotted decimal? How do we figure that out because we know that 255.255.255.0 is /24. Let's bring back our binary baseline of: 128 64 32 16 8 4 2 1

Since we have 3 additional bits from 24 we need to add the first 3 numbers from our binary baseline: 128 + 64 + 32=224

So the dotted decimal equivalent for /27 is 255.255.255.224

Now we need to convert that subnet mask to binary:

11111111 11111111 11111111 11100000 = SN Mask

Let's Compare:

11001000 10011010 10010110 01011001

11111111 11111111 11111111 11100000

11001000 10011010 10010110 01000000 = AND

Now that Boolean AND needs to be converted back to dotted decimal.

So the subnet that IP 200.154.150.89 (/27) belongs to is: 200.154.150.64 (/27) **Remember** to put the subnet mask!

This answer is also valid: 200.154.150.64 SNM: 255.255.255.224

How To Determine the Range of Valid Host Addresses on a Given Subnet

To start we need to:

1. Find out how many overall host addresses are on the subnet.

2. We also need to know that the **first address** is the network number and the **last address** is the broadcast address for the particular subnet so they are both invalid. Everything in between those two addresses is what's valid.

What is the valid range for this subnet: 200.154.150.64 (/27)

First we need to convert both of the given values above to binary so:

11001000 10011010 10010110 01000000 = Network Number (IP)

11111111 11111111 11111111 11100000 = Subnet Mask

Now we need to figure out how many **host bits** we have. Looking at the two binary numbers above you can see that I have marked that in blue.

We have **5 host bits.** Remember that our given IP is the first address in the range which is set to 0 in our binary conversion above.

1st Address In The Range: 200.154.150.64 = Network Number and Not Valid for hosts. You cannot assign this number to a host device.

How do we figure out the last address??? Let's look at our binary conversion above. Looking at the network number we need to change the 0's **that match** to 1's.

Note: I have marked one of the 1's with red because that bit was already set so we don't have to worry about changing it.

So now we have:

11001000 10011010 10010110 01011111 = Network Number (IP)

11111111 11111111 11111111 11100000 = Subnet Mask

Now that we have this new network number we need to know what to do with it. Let's go back to our binary baseline. Let's put that last octet in our network number under our baseline because for all of the numbers that have a 1 under them, we need to add them up.

128 | 64 | 32 | 16 | 8 | 4 | 2 | 1

0 1 0 1 1 1 1 1

So we are adding – 64+16+8+4+2+1=95

Last Address In The Range: 200.154.150.95 – This is the broadcast number for this particular subnet and is also invalid for use on a host.

The valid range of the addresses is everything in between so: 200.154.150.65 - 200.154.150.94 is the valid range of hosts.

Meeting Stated Design Requirements

This is one of the more difficult subnetting questions because it's one that needs to be well thought out. Really read questions like this. With the help of this example and enough practice questions, things should become a little easier.

You have this question:

You have a network of 165.10.0.0. You need **at least** 150 subnets that have no more than 200 hosts apiece. What subnet mask should you use?

1. We need to identify what Class the network is in. It's a **Class B** network. Then we need to write out the default mask for it.

255.255.0.0 (/16) – 16 Network Bits 16 Host Bits

Remember: Subnetting is the process of borrowing host bits. Going back to the original question, it's saying that we need at least 150 subnets. We need to figure out how many bits it's going to take us to get to the amount of subnets that we need.

So we use **2** to the power of the number of bits until we get a number over 150. The answers that we get will determine how many subnets we have. I have built a table below for easier viewing.

Bits	Power	Subnets
1	1x2	2
2	2x2	4
3	4x2	8
4	8x2	16
5	16x2	32
6	32x2	64
7	64x2	128
8	128x2	256

So we are saying that we need to borrow 8 host bits to get 256 valid subnets.

The next question would be: If we have 8 subnet bits to borrow and with a default mask of 255.255.0.0 what is our mask in dotted decimal (this is not the final answer as you will see when you keep reading)?

Well it would be: 255.255.255.0 or 165.10.0.0 (/24)

Explanation: Our default mask for a Class B IP **already has** 16 network bits set to 1:

255.255.0.0 = **11111111 11111111** **<=network|host=>** 00000000 00000000

Since we are **borrowing 8 host bits** we need to set the next octet to 1's so:

11111111 11111111 11111111 00000000 and what does that equal - **255.255.255.0**

The first part of this question is only partially done. We need to do the **second part** of this question to really validate the first part. We need to make sure that the values we have allow no more than 200 hosts per subnet.

If we started out with a default subnet mask of:

255.255.0.0 which is 16 network bits and 16 host bits

Then **borrowed 8 bits** leaving us with the mask of:

255.255.255.0

How many host bits does that leave us with? It leaves us with **8 host bits.**

So 2 to the 8th power as we know is 256.

256 – 2=254 per subnet (We are subtracting 2 because the network address and broadcast address are invalid)

Unfortunately, we have way too many hosts now which makes the answer invalid.

However we can correct that. We need to borrow a host bit because the original question says that we need **at least** 150 subnets yet we have 256.

We had:

11111111 11111111 11111111 00000000 and what does that equal - **255.255.255.0**

Now we have:

11111111 11111111 11111111 10000000 (notice the 1 in red. This is to represent the borrowed bit). What does that equal in dotted decimal? - **255.255.255.128.** Remember that last bit was borrowed and set to 1 so the dotted decimal for the binary number 10000000 = 128

So we now have:

2^9 = 512 valid Subnets

Leaving us with:

7 host bits = 128-2=126 valid host addresses per subnet

In our table we have added another row.

Bits	Power	Subnets
1	1x2	2
2	2x2	4
3	4x2	8
4	8x2	16
5	16x2	32
6	32x2	64
7	64x2	128
8	128x2	256
9	**256x2**	**512**

So the correct answer is 165.10.0.0 with as mask of 255.255.255.128 or 162.10.0.0/25 (how did we get that mask of 25 anyway? Count the 1's in our final dotted decimal with the red 1 in it on this page. Boom)