AISE 4430

Intro to Networking, Security, and IoT Systems

Arnav Goyal

2025-2026

Contents

1 Sample Problems & Solutions **3**

1.1 Sample Problem Set 5 3

1.1.1 Problem I 3

1.1.2 Problem II 3

1.1.3 Problem III 4

1.1.4 Problem IV 5

1.1.5 Problem V 6

1.1.6 Problem VI 7

1.1.7 Problem VII 8

1 Sample Problems & Solutions

1.1 Sample Problem Set 5

This sample problem set deals with the IP architecture, subnetting, etc.

1.1.1 Problem I

Question:

Do Routers have IP addresses? If so, how many?

Answer:

A Router is a device that joins different **subnets** together, To communicate on each of these networks it needs an **interface** on each one. So yes, a Router has one IP address for every interface it connects to.

1.1.2 Problem II

Question:

What is the 32 bit binary equivalent of the IP Address 223.1.3.27?

Answer:

IP Addresses follow the **four-dotted-decimal** convention, in decimal this consists of four numbers from 0-255 placed together with dots in between them. To convert these into binary, we must write out the corresponding 8 bits of binary for each number. An easy way to do this is to first convert the number into **hexadecimal** format, and then write the binary out for it.

To convert a decimal into hex, we perform the following recursive operation. Suppose our 8-bit hex number is labeled H, where H_0 and H_1 correspond to the LSh and MSh respectively. Lets also suppose our decimal number is labeled D, Note here that the % sign corresponds to the mod operator

$$H_n = \left\lfloor \frac{D}{16^n} \right\rfloor \% \, 16$$

This means our corresponding LSh and MSh become F and D respectively

$$H_0 = \left\lfloor \frac{223}{1} \right\rfloor \% \, 16 = 16 = F$$

$$H_1 = \left\lfloor \frac{223}{16} \right\rfloor \% \, 16 = 13 = D$$

Thus 223 decimal is equivalent to DF in hex. Now we just write it out in binary. Repeating this process for each number in the IP address provides the following result for the binary IP address.

$$1101\_1111 \; 0000\_0001 \; 0000\_0011 \; 0001\_1011$$

1.1.3 Problem III

***Question*:**

Consider a router that interconnects three subnets: Subnet 1, Subnet 2, and Subnet 3. Suppose all of the interfaces in each of these three subnets are required to have the prefix 223.1.17/24. Also suppose that Subnet 1 is required to support up to 125 interfaces, and Subnets 2 and 3 are each required to support up to 60 interfaces. Provide three network addresses (of the form a.b.c.d./x) that satisfy these constraints.

***Answer*:**

To do this problem we need to figure out what this notation means first. Here we have an IP address followed by a slash. The main part of this address is 223.1.17.0. Note that this number has an implicit zero, because there is no number specificed before the slash. We also have the /24 part, which means that we have 24 bits in the *entire* IP address reserved for the host (main part of the IP). This means we have 32 - 24 = 8 bits leftover to partition subnets into, Note that since IP addresses are four numbers within 0-255 it will always fit into 32 bits.

So with the 8 bits leftover we need to partition it properly into three subnets of sizes 125, 60, and 60. When we partition a subnet, we need to understand that two addresses for *each* subnet are unusable by other interfaces, these are the **network address** and the **broadcast address**. These reserved addresses take up the lowest and highest slots of the subnet partition respectively.

Thus the number of usable bits per partition becomes

$$\text{slots} = 2^b - 2$$

Our challenge is finding out the number of bits we need in each partition, we always start with the largest one (125).

so we will need a slot that fits 125 interfaces + 2 reserved addresses = 127 slots in total. This fits super will into a 7 bit value. We have all 8 bits left over since the end part of the main IP is 0, thus we have one bit left in the IP address, so our notation becomes

$$\text{Subnet 1} = 223.1.17.0/25$$

This means we have IP addresses for this subnet ranging from 223.1.17.0 to 223.1.17.127, where the boundaries are reserved, and anything in between is usable by interfaces.

For the other two subnets, we do something similar. We will need 60 + 2 slots, which fits into a 6 bit value, this means we have the following 223.1.17.x/26

this 'x' value will need to be found though, its the start of the subnet partition, so here since our larger net ends at 127, we put 128

so our first subnet of 60 becomes

$$\text{Subnet 2} = 223.1.17.128/26$$

This second subnet ranges from 223.1.17.128 to 223.1.17.191, where the boundaries are reserved, and anything in between is usable by interfaces. This process repeats for subnet three, where it becomes:

$$\text{Subnet 3} = 223.1.17.192/26$$

1.1.4 Problem IV

***Question*:**

Consider a subnet with prefix 101.101.101.64/26. Give an example of one IP address (of form xxx. xxx. xxx. xxx) that can be assigned to this network

***Answer*:**

We know the subnet's boundaries are reserved for network/broadcast address, so anything within these boundaries works! Thus we can just write anything withing the range 101.101.101.65 to 101.101.101.126, .64 and .127 are reserved.

1.1.5 Problem V

Question:

Suppose an ISP owns the block of addresses of the form 101.101.128/17. suppose it wants to create four subnets from this block, with each block having the same number of IP addresses. What are the prefixes (of form a.b.c.d/x) for the four subnets?

Answer:

To answer this lets look at how many bits we have left to partition this ID into.

$$32 - 17 = 15$$

Thus we have 15 bits to work with! Note that this is over 8 bits, which means we will be using the last two numbers in the IP address, instead of only the last one. Lets obscure the first two numbers as x = 101.101. for shorthand

Our entire range of values will be

$$2^{15} = 32,768$$

to divide this by four we can essentially shift two bits right, resulting in each subnet having 13 bits to work with

$$2^{13} = 8192$$

So each subnet will have 8192 slots, where only 8190 are usable, the other two are reserved.

We can divide 8192 by 256 to see how many incrments in the third number we get per slot, 8192/256 = 32 increment per 8192 slots

We start at x.128, so lets add 31 each time, four times. This gives the following subnets

Subnet 1 => x.128.y to x.159.z

Subnet 2 => x.160.y to x.191.z

Subnet 3 => x.192.y to x.223.z

Subnet 4 => x.224.y to x.255.z

To find out the y and z values from above we can see whats left over from our increment, here our increment cleanly divides into 32 slots, thus the start and end will be 0 and 255 respectively as we fully fill up the chunk, however if we have any left over addresses, we would have to calculate things manually by hand tediously. Thus our 4 subnets become

Subnet 1 => 101.101.128.0 to 101.101.159.255 (101.101.128/19)

Subnet 2 => 101.101.160.0 to 101.101.191.255 (101.101.160/19)

Subnet 3 => 101.101.192.0 to 101.101.223.255 (101.101.192/19)

Subnet 4 => 101.101.224.0 to 101.101.255.255 (101.101.224/19)

1.1.6 Problem VI

Question :

Suppose an ISP owns the block of addresses of the form 192.168.15.0/24. Suppose it wants to create four subnets from this block. What are the prefixes (of form a.b.c.d/x) for the four subnets? Find the number of interfaces that each subnet can support?

Answer :

This one is very similar to the previous one, here we have 8 bits left over to work with, four subnets as well. This means we have

$$2^8 = 256 \text{ total slots}$$

$$2^6 = 64 \text{ slots per subnet}$$

So each subnet will have an increment of 63 in the smallest number, This means our subnets become

Subnet 1 => 192.168.15.0 to 192.168.15.63 (192.168.15.000/26) Subnet 2 => 192.168.15.64 to 192.168.15.127 (192.168.15.064/26) Subnet 3 => 192.168.15.128 to 192.168.15.191 (192.168.15.128/26) Subnet 4 => 192.168.15.192 to 192.168.15.255 (192.168.15.192/26)

Each subnet has 64 slots, and 60 of them can be used for interfaces, thus we have 240 total slots for interfaces available here throughout all 4 subnets.

1.1.7 Problem VII

***Question*:**

Suppose an ISP owns the block of addresses of the form 132.59.0.0/16. Suppose it wants to create two hundred subnets from this block. What are the prefixes (of form a.b.c.d/x) for the four subnets? Find the number of interfaces that each subnet can support?

***Answer*:**

Here we need 200 subnets, so we need to see how many are used to MUX between the subnets. Here clog2 refers to the ceiling of log base 2 operation, since we are working with binary values it is abbreviated to the common shorthand.

$$b = \text{clog2}\,(200) = 8$$

Thus we have 16 bits reserved for the ID, another 8 bits to select each subnet, and the remaining bits determine the slots per subnet, so we have 24 fixed bits, and thus 8 bits left per subnet. This means each subnet will support 253 interfaces and 2 more reserved slots.

The first four subnets have the following, since each subnet has 256 slots, it increments the second lowest decimal number by 1. Thus our subnets become

Subnet 1 => 132.59.0.0/24 Subnet 2 => 132.59.1.0/24 Subnet 3 => 132.59.2.0/24 Subnet 4 => 132.59.3.0/24

Subnet n => 132.59.$(n-1)$.0/24, for any n in [0, 200], the other slots [201, 255] will be effectively unused.