IP Addresses and Subnetting

IP Notation

When the internet was created 32-bit numbers uniquely identified every interface on the network. As a convenience the 32-bit number was broken into 8-bit octets. For example:

$$10001000\ 10000001\ 00001111\ 10101010 = 2290159530$$

Trying to remember either of these is difficult so the **shortcut** notation used looks like:

$$136.129.15.170$$

Practice converting the following 32-bit numbers to the shortcut notation:

1. 10110100 10001101 00110101 00000000
2. 00110100 01110010 11001010 11111111
3. 01000000 10010000 10110100 01110010

You should also be able to convert back to binary notation. Try the following:

1. 192.168.10.1
2. 172.17.34.12
3. 10.128.64.133

Before you look at the answers below do try to do the conversions yourself. Hopefully you got the following answers below.

1. 176.141.53.0
2. 52.114.202.255
3. 64.144.180.114

4. 11000000 10101000 00001010 00000001
5. 10101100 00010001 00100010 00001100
6. 01000000 10010000 10110100 01110010

Subnetting

How do you allow any device (IP address) to talk to any other device? With just a small network, you can just broadcast the information and let the destination device pick it up. However you can't do that with 4+ billion devices. But for small networks you can do that and *early networks actually did broadcast all local traffic to every device*. Each one of these "broadcast" networks is called a **subnet**. You can easily identify a subnet because it contains all network interfaces that packets can reach without being forwarded across a router. Figure 1 shows an example of the three subnets around a router. Notice that each of the interfaces on the router belongs to one of the three subnets.

Since we now need to route a packet, not to a particular device, but a subnet, we must find a way to identify a subnet or range of network address.

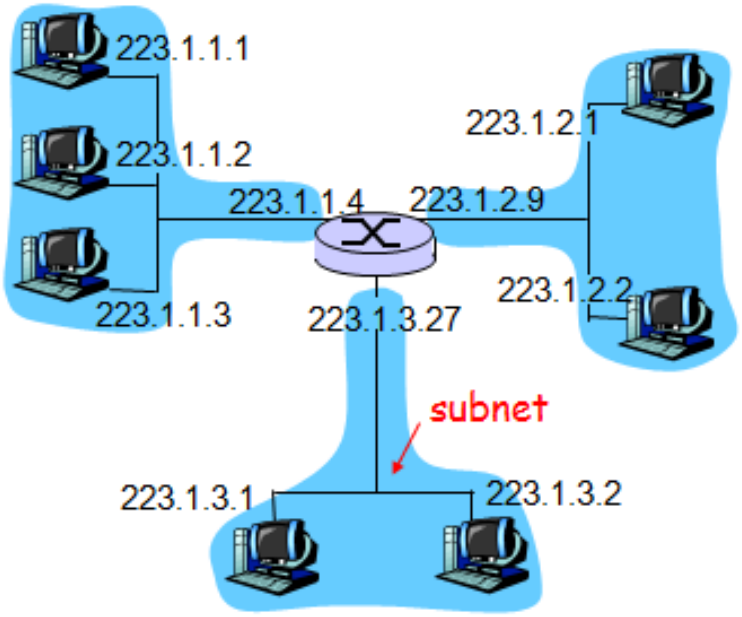

Figure 1

It would be easy to say just take a consecutive range of number somewhere between 0 and 4 billion. But we have to implement routing in hardware and the implementation must be fast. Implementing a bunch of if-statements requires time. We will see how the designers of the internet solved this problem with the following concepts (think binary here):

1. Network Classes (no pretty much obsolete)
2. Network Address / Network Mask / Network broadcast
3. Classless inter-domain routing

Network Classes

Size does matter when you consider the number of hosts you have in an organization. You can't just hand out ranges of IPs with no organization, so the founders created classes which currently have the form shown in Figure 2 and the size is reflected in the octet notation you normally see.

Class A networks have $2^{24} \approx 16$ million IP addresses in them and have the form of **N.H1.H2.H3**. The IP is broken up into two parts. The first part is the network address designated by N. This number stays the same for every IP in the subnet. H1-H3 are numbers that designate a host on the subnet.

As Figure 2 shows there are 0-127 or 128 Class A networks (but really, who needs 16 million IP address?) Out of those networks, 127.N1.N2.N3 reserved for localhost IP addresses (interestingly they felt that more IPs were needed than ports). Also they reserved 10.N1.N2.N3 as a non-routable private class A network.

As you can see Class A networks take up half of all available IPs. Half of the remaining IPs (128-191) go to Class B and 192-223 become the class C networks.

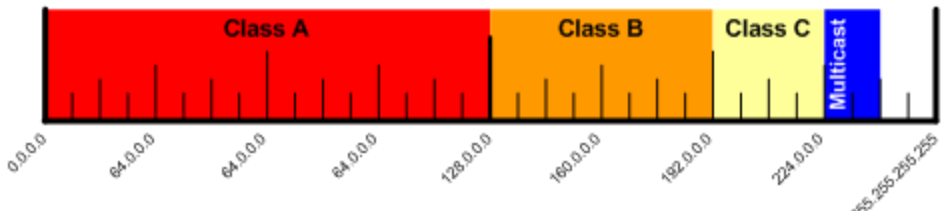

Figure 2

Each Class B network takes the form of N1.N2.H1.H2 and have the same Network/Host parts as Class A networks except that each IP address is broken after bit 16th bit (after the second octet). So for example, 161.223.55.32 is on the 161.223.H1.H2 network. Numerically which host is it? There are several ways to calculate it. If you think of the different numbers in the octet as being a digit in a base 256 number system then this host is the 256*55+32 = 14,112th host in this Class B network. If you just convert it to a binary number, its 0011011100100000 which also converts to 14,112.

Class C networks have IPs of the form N1.N2.N3.H. So the first three numbers/octets form the Network address and the last octet designates the different hosts. For example 200.0.0.23 breaks up as follows. Network part is 200.0 and Host part is 0.23.

Class	Range	Reserved Range	Reservation Purpose
A	1.0.0.0 – 126.255.255.255	10.0.0.0-10.255.255.255	Private
B	128.0.0.0-191.255.255.255	169.254.0.0-169.254.255.255 172.0.0.0-172.255.255.255	Self-Assigned (NAT) Private
C	192.0.0.0-223.255.255.255	192.168.0.0-192.168.255.255	Private
Multicast	224.0.0.0-239.255.255.255		
Reserved	240.0.0.0-255.255.255.255	255.255.255.255	Broadcast

Network Address, Mask and Broadcast Address

Although we introduced the "network part" of an address, we haven't talked about the network address. First, how do we identify a Class A, B or C address? Well what does it start with:

Class	Starts with
A	0
B	10
C	110

This allows easy identification of Class networks. Fortunately the internet has gone classless.

To talk about the network address its helpful to understand the concept of a mask. In general terms a mask hides those things that need to be hidden while allowing other parts to be seen (or seen through). This is exactly what a network mask does.

A network mask is a binary string of 1's followed by 0's (strictly segregated, no mixing). For example in the Class C network: 192.168.3.0, the mask is 11111111 11111111 11111111 00000000 or 255.255.255.0. When you "AND" any IP address in the network with the mask, you obtain the **Network Address**. *This will always be the first address in the subnet*.

There is also a *shortcut notation* for a mask. Since you can define a mask by the number of 1s that start the mask (e.g. a Class C subnet always has 24 1s followed by 8 0s in the mask), we write the preceding range of IP addresses as **192.168.3.0/24**. This is equivalent to

Network Address=192.168.3.0
Subnet mask = 255.255.255.0

We now have defined enough Information to clearly define a subnet. We do this with a

W.X.Y.Z/mask

How many IPs will be in a network? Simple:

$$Size_{subnet} = 2^{32-mask}$$

Really we don't to define the broadcast address in different terms than the network address and mask. Simply put, the last address in the range is the broadcast address and every host in the subnet should listen on that address.

Given any IP address and a mask, you can find the network and broadcast addresses.

$$Address_{network} = IP\ AND\ Mask$$

$$Address_{broadcast} = IP\ OR \sim Mask$$

Where $\sim Mask$ is the one's complement of the Mask.

Classless Inter-domain Routing

This is all well and good, but what if you want a smaller than 255 IPs in a subnet? We don't have IPs to waste in IPv4 and hence IPv6 has an address size of 128 bits (4 times the size of an IPv4 address and 2^{96} times as many IP addresses available in the internet). The trick is that we allow the mask to have any number of 1s not just 8, 16 or 24 a.k.a Class A, B, and C sizes. And as we'll see we can then route hierarchically.

Let's take the Class C network we have been using. 192.168.3.0/24. Now suppose we have a main office with 70 computers in it and two branch offices each with 24 computers. We have to route between them, because if they are on the same subnet, you will have collisions (see test question on last test about what delay causes collisions). Therefore we need three subnets. Here is how you do it.

1. Determine the needs of each network and then the size of mask required to get that many IPs

Main Office	70 computers → $2^7 > 70$, 7 bits of host addressing or a /25 mask	128 addrs
Branch 1	24 computers → $2^5 > 24$, 5 bits of host addressing or a /27 mask	32 addrs
Branch 2	24 computers → $2^5 > 24$, 5 bits of host addressing or a /27 mask	32 addrs

At this point, we should ask ourselves if it's possible. The answer of course is yes. We are using much less than 255 addresses given in the Class C size space.

2. The next step is to break the host address space down into separate subnets.

This can only be done by halving the host address space (base 2 property). Forget about the first 3 octets for a minute since our host address space is fully contained in the last octet.

00000000 /25 → 0-127
10000000 /25 → 128-255

This is because the first bit of the last octet shown above is NOW part of the network address. How do we know? The MASK tells us so. Here is what the whole address looks like.

11000000 10101000 00000011 00000000 → 11000000 10101000 00000011 00000000
11000000 10101000 00000011 10000000

Now we have two subnets: 192.168.3.0/25 and 192.168.3.128/25. One of these will work for the main office (We'll use the first one). That means we need to break up the second one.

11000000 10101000 00000011 10000000 → 11000000 10101000 00000011 100000000
11000000 10101000 00000011 110000000

This gives us the three we need:

192.168.3.0/25
192.168.3.128/26
192.168.3.192/26

Let's do one more example: Suppose we have the same Class C and we want to break it up into 120, 15 and 70 addresses. Now we can do this, because the number IPs needed doesn't exceed what we have.

120 → /25
15 → /27
70 → /25 (but we can't waste that many IPs 128-70 = 58 wasted IPs)

What else can we do?

/24 → 2x /25 One of those is used

The rest are broken up as follows:

/25 → 2x /26→4x /27 (each of which has 32 addresses) That's enough actually 32 and 96 will work for our scenario! We use one /25 the other is broken into two /26 subnets (we use one of these) and break the remaining one into two /27 subnets. Remember mask implies size (you should be able to do these in your sleep before long). So we get the following configuration:

Network Size	Subnet(s) Assigned
120	192.168.3.0/25 (128)
70	192.168.3.128/26 (64 addresses) 192.168.3.192/27 (32 addresses)
15	192.168.3.224/27 (32 addresses)

Routing

The above example is perfect for understanding Classless inter-domain routing. Who is responsible for the 192.168.3.0/24 address range? If I'm the organization that got the IP address range, then I MUST KNOW HOW TO GET A PACKET TO ANY DEVICE IN THE SUBNET. Assuming my ISP gave me that address range and packets from the internet come only from my ISP (a reasonable assumption for now), then the connection from my router to their router connects the whole class C in some way and their router needs only one entry. That entry is the prefix/interface pair

11000000 10101000 00000011, eth2

Assuming eth2 is the interface name that they have going to my network.

Review

Given an IP address and a mask (in any form) you should be able to give me the Network and Broadcast addresses.

Given a subnet (network address and mask) you should be able to break it up into two equally sized subnets.

Can you break a network down into three equally sized subnets? (absolutely not! You can break it up into exactly two subnets. You can then break one or both of these …)

How big will your subnets be? (Exactly half the size of the network you created them from)

Do you lose any addresses as you subnet? Technically no, but in practice yes – namely the network and broadcast addresses.