1 IPv4 (and IPv6) Networks

For many years, the internet has been running on a standard called IPv4. Even today, in 2016, it is extremely rare that you are assigned anything but an IPv4 address, although the days of the IPv4 standard seem numbered. The simple reason is that IPv4 provides a maximum number of 4 Billion addresses. At the time when this standard was created, and a computer cost hundred thousands of dollars, this seemed like a ridiculously large number. Today, many people constantly use up several IP addresses for their laptop, smartphone, IPad, and so on. The other problem is that the available address space is used extremely inefficiently.

Addresses are virtually always assigned in chunks of a *subnet*. Although it can be smaller, a subnet usually has 256 addresses, and very often more. It is common to find a rather small number of actual addresses actually used. Where I work, we performed an aggressive consolidation of the available address space; still, one often finds subnets where only 20 or 30 addresses are actually used.

1.1 IPv4 Subnets

Most often, your mobile devices get a dynamic IP address assigned when they go online. (Your desktop PC at home can do the same, but it is more likely that you assign it a static IP address so it is easier to log into it.) For example, when you connect wirelessly in the hotel where you are staying, the wireless router assigns your device an IP address though a mechanism called DHCP. The end result is that your computer learns about its IP address, the network mask and broadcast address, the gateway address, the name server address(es), and potentially a number of additional pieces of information, such as a default search domain, a time server, and things like that.

A subnet is defined as the collection of all addresses which are in the same broadcast domain. You can reach any of those addresses directly without traversing a *router*.

The `ifconfig` command (or `ipconfig` on Windows) will tell you the setup of your network interface, such as

```
$ ifconfig eth0
eth0      Link encap:Ethernet  HWaddr 00:01:2e:24:13:45
          inet addr:192.168.12.15  Bcast:192.168.12.255  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:130172103 errors:0 dropped:1 overruns:0 frame:0
          TX packets:102924237 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:15901078273 (14.8 GiB)  TX bytes:33982084132 (31.6 GiB)
          Interrupt:16 Base address:0xc000
```

So you see that the address is 192.168.12.15, the broadcast address is 192.168.12.255, and the network mask is 255.255.255.0. The latter two are related in the following way. If you consider the 4 numbers, each of which can run from 0-255 and be represented with 8 bits, as a 32bit number, then

```
broadcast = address | ( netmask ^ 0xffffffff )
```

where `^` denotes the XOR operator.

So in the example above, the XOR operation on the netmask yields 0.0.0.255, or `0x000000ff`. Bitwise or'ed with the address, this yields the broadcast address 192.168.12.255 that we see.

The span of a subnet can most easily be expressed with the broadcast address. A subnet is the collection of all addresses for which the following is true:

```
address & broadcast_address == address
```

where `&` denotes a bit-wise and.

So in our example, 192.168.12.15 bit-wise "and-ed" with 192.168.12.255 gives 192.168.12.15. As a matter of fact, the condition is true for *all* addresses starting with 192.168.12; so this subnet holds 256 addresses, from 192.168.12.0 through 192.168.12.255.

Let's say that we host a larger collection of machines on that subnet, and 256 addresses are not enough. We could expand the subnet by "shrinking" the network mask, such as 255.255.254.0 (instead of 255.255.255.0). The broadcast address would change to 192.168.13.255, and the subnet would be comprised of all addresses which start with either 192.168.12.x or 192.168.13.x In effect, we have doubled the size of our subnet from 256 to 512 addresses.

Often we see a slightly different notation of the network mask, which is shorter and just gives the number of leading "1"s (in binary notation) of the netmask. Our previous 255.255.255.0 network mask reads 0xffffff00 in hex, and so has 24 leading ones. It is referred to as a "/24" network. Similarly, 255.255.254.0 has 23 leading ones, making it a "/23" network. If I were to specify the above address and network, it would suffice to say "192.168.12.15/24".

Each bit that we chop off the network mask doubles the size of the subnet. For example, my DAQ network at work has a very large broadcast domain; it is a /20 subnet, holding 8192 addresses.

1.2 Routing

Routing is what makes the internet what it is, and ties the different networks and domains together in a rather robust and fault-tolerant way. For example, it happens now and then that we lose a major transatlantic fiber, and still the internet soldiers on, maybe at a slightly slower pace, but largely unnoticed by anyone except the major network operators.

This is achieved by a rather simple-minded way for internet packets to be routed. You may think that a packet "knows" the way to its final destination the same way a driver in the US knows how to drive, say, from New York to Philadelphia – take the Verrezano Bridge, the Interstate 278 and Interstate 95, and so on.

This is not how an internet data packet travels. If that packet would drive by car, it had no idea how to get there, and no overview of the entire trip whatsoever. The car would leave the driveway, and there'd be someone standing saying "to Philadelphia? Take a right". At the next corner, "take a left here". This car would get its directions at *every* turn it needs to make, good only until the next turn.

Although this may be less satisfactory for a human driver, it allows a more efficient trip for our data packets in case that, say, there's a big traffic jam on the Verrezano bridge. The guy giving directions might then direct our car to take the Midtown and Lincoln tunnels in New York instead of taking the bridge, and in this way avoid traffic congestion all together. Two packets arriving a the same destination need not take the same path.

In the same way, if your home PC sends a packet to google, all it really needs to know is how to get out of your house. In virtually all such cases, your wireless router is its first destination, which then takes care of the rest.

Here is the routing info the machine we have already seen:

```
$ netstat -rn
Kernel IP routing table
Destination     Gateway         Genmask         Flags   MSS Window  irtt Iface
192.168.12.0    0.0.0.0         255.255.255.0   U         0 0          0 eth0
127.0.0.0       127.0.0.1       255.0.0.0       UG        0 0          0 lo
0.0.0.0         192.168.12.99   0.0.0.0         UG        0 0          0 eth0
```

This says that all 192.168.12.x addresses are part of this subnet and don't require routing services. All others will be directed to 192.168.12.99, which is the routing interface on this subnet. Whatever comes next is known by the upstream service provider.