IP Addresses

We now turn our attention to the network layer. Its focus is about how to take a packet from its source and get it to its destination, which may be across the world. This process is called *routing*, where devices called *routers* receive packets and forward them to other routers that are hopefully closer to the destination. The main protocol at this layer is the *internet protocol* (IP). Every machine on the internet has an IP address, which is essentially its "location" on the internet. Routers use various parts of the destination IP address on a packet to know where to forward it to.

Many things in networking have real world analogs. Routing and IP addresses are a lot like postal mail. When you send a letter, you usually put a name, street address, town, state, zip code on the envelope, as well as a return address. That envelope plays a similar role to the IP header, which has a source and destination IP address, in addition to some other things.

When you mail the letter, sorting equipment looks at the zip code and decides where to send it. If you're sending something from Emmitsburg, MD to California, there's no mail truck that goes directly from Emmitsburg to California. Instead things from Emmitsburg might always get sent down to Frederick first. Once there, anything destined for California might get put on a truck to Pittsburgh. From there, maybe it goes to Chicago and then Denver, and so on. Routing a packet is similar in that it will bounce from router to router, hopefully getting closer and closer to the destination. Each router looks at certain bits of the IP address and uses information in its routing table to decide where to send the packet next.

With postal mail, once the letter reaches the destination mailbox, someone at that address will get the mail and deliver it to the right person. That's a little like the role of port numbers at the transport layer.

Below is the output of a useful tool called *traceroute* (called tracert on Windows). It shows the path a packet takes from a source to a destination, `vk.ru` in this case (the first few i addresses were changed so as not to give out possibly sensitive internal network details):

```
 1    2 ms     7 ms     2 ms  10.2.9.1
 2    2 ms     4 ms     2 ms  10.38.11.101
 3    2 ms     1 ms     1 ms  10.255.255.2
 4    2 ms     2 ms     2 ms  50-238-227-245-static.hfc.comcastbusiness.net [50.238.227.245]
 5    4 ms     3 ms     2 ms  96.110.234.37
 6    8 ms     7 ms     5 ms  68.86.205.173
 7    *        *        *     Request timed out.
 8    *        *        *     Request timed out.
 9   94 ms    94 ms    93 ms  195.122.183.218
10  129 ms   127 ms   127 ms  213.59.211.255
11  142 ms   141 ms   142 ms  5.143.251.250
12  188 ms   126 ms   126 ms  ctv-r5.nic.ru [31.177.67.139]
13  169 ms   131 ms   130 ms  dp-r5.nic.ru [31.177.67.76]
14  141 ms   134 ms   132 ms  dp-gw1.nic.ru [31.177.67.73]
15  145 ms   143 ms   143 ms  std-carp5-http.nic.ru [195.208.1.105]
```

We see that the packet took 15 hops to get from the source to the destination. The three times shown are how long it takes to get from the source to that hop, based on three separate timings. The tool actually sends the packet many times, each time getting one hop farther, and timings will vary based on the amount of network traffic, which is why the times are a little variable. Notice the big jump in time from hops 6 to 9. This is likely where the packet crossed the ocean. The right side shows the IP address of the router at each hop, along with a domain name if it has one. For each of these addresses, you could do a geo IP lookup to see roughly where in the world it is located. Just do a search for geo IO lookup tools online.

IP addresses

There are two types: IPv4 and IPv6. We will focus here on IPv4 (IP version 4) addresses. Some examples are 192.168.0.1 and 127.0.0.1. IP addresses are usually written in four groups of decimal numbers, which numbers in each group ranging from 0 to 255. So the smallest possible address is 0.0.0.0 and the largest is 255.255.255.255.

In binary, an IP address is 32 bits long. This means there are $2^{32} \approx 4$ billion possible IP addresses. There are way

more devices on the internet than this. The two solutions to this problem are NAT and IPv6, both of which we will cover later.

The leftmost bits of an IP address are more general, identifying what network the address belongs to, and the rightmost bits identify specific hosts (devices) on the network. This is a little like how phone numbers get more specific as you go from left to right. For instance, in the phone number 1-301-447-5214, the first 1 is the country code, 301 is the area code for this part of Maryland, 447 is the Emmitsburg exchange, and 5214 is a specific phone in the MSM Admissions Department.

Class-based system of IP addresses

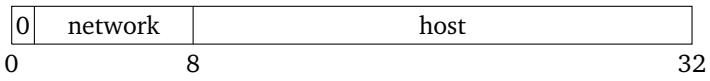
When IP addresses were first created and standardized in the early 1980s, all addresses were divided into five classes: A, B, C, D, and E. The class-based system has long since been abandoned, but it has consequences for what the IP address space looks like today, so it's worth looking at.

To understand the system, it will be helpful to think of an IP address in binary. The address 1.2.3.4 looks like this in binary:

00000001.00000010.00000011.00000100

It's 32 bits long, with the dots coming at bits 8, 16, and 24. Addresses are broken into two parts: the network portion and the host portion. Where exactly that division happens varies based on the address, but the network portion is always to the left of the division and the host portion is on the right. A *host* is networking jargon for a device, such as a computer or phone.

Class A Class A addresses start with a 0 bit. That 0 and the next 7 bits make up the network part of the address. The remaining 24 bits make up the host part of the address. See below.

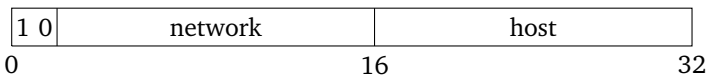


Since there are 24 bits for the host, there are $2^{24} \approx 16$ million hosts that can fit on this network. So Class A networks are huge. However, there are 8 bits for the network, and the first bit is 0, so there are only $2^7 = 128$ possible Class A networks.

Because the division between network and host happens right at bit 8, which is at the first dot, the possible class A networks are 0.0.0.0 – 0.255.255.255, 1.0.0.0 – 1.255.255.255, etc. That is, all addresses starting with a 0 are one network, all addresses starting with a 1 are another, etc. Since Class A addresses start with a 0 bit, the last possible Class A block is the 127 block.

You had to be a pretty important to get a Class A network. These went to a few big schools, US government agencies, and some large companies. Some of the organizations have since given up their blocks, while others still have them. For instance, Apple has the entire 17 block, Ford has the entire 19 block, and the US Department of Defense. has 13 separate Class A networks. See https://www.iana.org/assignments/ipv4-address-space/ipv4-address-space.xhtml for a current list of who has which blocks. Various people have made maps of what the IP space looks like. The most well-known is https://xkcd.com/195/ from 2006. The internet has changed considerably since then, but it's still a really nice picture.

Class B Class A addresses start with 10 in binary. That 10 and the next 14 bits make up the network part of the address. The remaining 16 bits make up the host part of the address. See below.

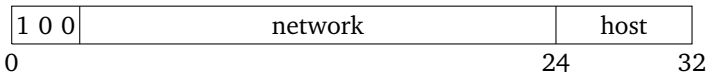


Class B networks have room for $2^{16} = 65536$ hosts. There are 16 bits for network, but the two beginning bits are

set to 10, so there are $2^{14} = 16384$ possible Class B networks.

Since the division happens at bit 16, the second dot, if you had a Class B address, you had all the addresses starting with two specific numbers. For instance, the first Class B block is all addresses from 128.0.0.0 to 128.0.255.255. These are all addresses starting with 128.0. The next Class B block is the 128.1 block, which is 128.1.0.0 to 128.1.255.255. Because Class B addresses start with 10 in binary, the last Class B block is the 191.255 block.

Class C Class C addresses start with 110 in binary. That 110 and the next 21 bits make up the network part of the address. The remaining 8 bits make up the host part of the address. See below.



Since there are 8 bits for the host, there are $2^8 = 256$ hosts possible on each Class C. The network portion is 24 bits, with the first 3 bits being set to 110, so there are $2^{21} \approx 2$ million possible Class C networks. For these, the network/host division is at the third dot. The first Class C network is the 192.0.0 block, which runs from 192.0.0.0 to 192.0.0.255. The next one is the 192.0.1 block, which runs from 192.0.1.0 to 192.0.0.255. The last one is the 223.255.255 block.

Class D Class D networks start with 1110 in binary. There is no network/host division here. Instead, this block of addresses is for something called *multicast*. Communications can be broken into three classes: *unicast*, *broadcast*, and multicast. Most traffic is unicast, which is where a sender sends a packet to a single receiver. In broadcast, a sender sends something to an entire network. Multicast is somewhere in between, where a message goes to multiple recipients.

This range is from 224.0.0.0 to 239.255.255.255, encompassing $2^{28} \approx 268$ million addresses, or 6.25% of all addresses. While multicast is used for a variety of purposes in networking, setting aside this much space for it turned out to be much more than was necessary.

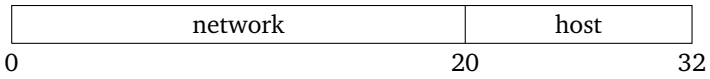
Class E Class E networks start with 1111 in binary. These are all the addresses from 240.0.0.0 to 255.255.255.255, which is roughly 268 million addresses. This address range is designated as reserved for future use. However, it likely will never be used. Most routers and operating systems are programmed to drop any packets that have an IP address in this range. If it was ever decided to use this address space, it would require reprogramming so many devices that it's apparently not worth the effort. There were a few proposals back in the late 2000s and 2010s to use this space, but they didn't go anywhere.

Classless Inter-Domain Routing

The class-based system had three sizes of networks: huge Class A networks of over 16 million hosts, 65536-host Class B networks, and 256-host Class C networks. Only a few groups got Class A's. Class C networks were too small for many organizations, and trying to patch together several Class C's was a pain, so many organizations would opt for a Class B. But if an organization only had 1000 hosts and they got a Class B, then they would end up wasting most of the 65536 addresses they got in their Class B network.

As the internet started to grow through the 1980s into the 1990s, it became apparent that the class-based system assigned addresses too inefficiently. So the it was dropped in favor of something called *Classless Inter-Domain Routing* or CIDR. In the class-based system (often called *classful*), the network/host division always happens at the dots, at bits 8, 16, or 24. CIDR allows it to happen at just about any bit.

To indicate where the division happens, CIDR notation is used. If the division happens at bit 20, then the network is called a /20 network. In this notation, a Class A is a /8 network, a Class B is a /16, and a Class C is a /24. Below is a picture of how things are divided in a /20 network.



The 20 in the notation indicates that there are 20 bits for the network part of the address. Since IPv4 addresses are 32 bits, the host portion is $32 - 20 = 12$ bits. This means we have room for $2^{12} = 4096$ hosts. Technically, it is 4094 for reasons we will talk about in a bit.

Special IP addresses

There are various parts of the IP address space that are used for specific purposes. Though there are many ranges, we will only look at a few important ones here.

Private address ranges These are probably the most important to know. The three private address ranges are shown below in CIDR notation. The first two are the most common.

10.0.0.0/8	10.0.0.0–10.255.255.255
192.168.0.0/16	192.168.0.0–192.168.255.255
172.16/0.0/12	172.16.0.0–172.31.255.255

These ranges are used by local area networks (LANs). If you have a wifi router at home, it's likely using one of the first two ranges. The router may be at address 10.0.0.1 and the various devices on the network would be at addresses like 10.0.0.2, 10.0.0.3, etc. A wifi router at your neighbor's house could be using the same range of addresses. These are all local, private networks, each one using the same range of IP addresses, but separate from the others. If you try to send a message to an IP address in this range, if there is a machine on the local network with that address, then it will go to that machine. But that message won't ever leave your network. Routers are programmed not to forward packets with these addresses to outside the network.

The idea is a little like phone system extensions. If you are at a certain business and dial extension 4444, you'll go to whatever phone at the business has that extension, maybe the HR office. If you're at another business and dial 4444, you'll go to somewhere on that network, maybe the finance office.

Loopback addresses The entire 127 block, specifically 127.0.0.0/8 is used for loopback addresses. They are called this because they "loop back" to your own device. They are often used for testing things on your own device or for running internet services (like a web server) on your device for your own non-public purposes. You will often see loopback addresses associated with the name *localhost*. The most common loopback address is 127.0.0.1. It's unfortunate that the whole block is used for loopback, since there really isn't a need for that many loopback addresses.

Two more special addresses On any network, the lowest and highest addresses are not available. The lowest address is called the *network address*. It is used to refer to the network itself, and there can't be a host with that address. The highest address is the *broadcast address*. It is used for a host to send a message to all other hosts on the network. For instance, on the network 192.168.0.0/16, the address 192.168.0.0 is the network address and 192.168.255.255 is the broadcast address.

Subnetting

Finding the range of addresses on a network One nice thing about the class-based system is that it's very easy to tell what the range of addresses is on a network. In the classless system, it's a bit trickier. To do this, it helps to write the address in binary. Let's look at the address 10.45.184.218 on a /22 network. It is shown below in binary, with the network/host division indicated by a vertical bar and the host portion highlighted.

00001010.00101101.101110|**01.11011010**

Python is useful for quickly doing binary conversions. To convert 45 into binary, use `bin(45)`. You may need to add zeroes on the left to get it to a full 8 bits. To convert a binary number, like 10010011, into decimal, put a 0b in front of it and print out the value, like 0b1001001. Python will automatically convert it to decimal. The Python script below converts a decimal IP address into binary.

```
'.'.join('{:08b}'.format(x) for x in [10,45,185,218])
```

To get the lowest possible address on a network, set all of the hosts bits to 0 and to get the highest possible address, set all of the host bits to 1. Then convert back to decimal. These are shown below for the example address just given:

original	00001010.00101101.101110	**01.11011010**	10.45.185.218
lowest	00001010.00101101.101110	**00.00000000**	10.45.184.0
highest	00001010.00101101.101110	**11.11111111**	10.45.186.255

So the network runs from 10.45.184.0 (network address) to 10.45.186.255 (broadcast address).

Number of hosts on a network In a $/n$ network, the value of n indicates how many bits of the address are for the network part. The host part then has $32-n$ bits. The number of hosts possible on that network is then $2^{32-n}-2$. The minus comes from removing the network and broadcast addresses. For instance, a /28 network has 28 bits for the network and $32-28=4$ for the host. Therefore, there are $2^4-2=14$ possible hosts on it.

Suppose we need a network large enough for 1000 hosts. What size network would we need? To do this, we do things in reverse of what we did above. We look at powers of 2 until we get one that is large enough. We have $2^1=2$, $2^2=4$, $2^3=8$, etc., and eventually we get to $2^9=512$ and $2^{10}=1024$. This tells us that we would need 10 bits for the host part, meaning $32-10=22$ for the network part, which means it's a /22 network.

Subnet masks CIDR notation is one way to specify where the network/host division is. Another way is the so-called *subnet mask*. Here is the subnet mask associated with a /20 network:

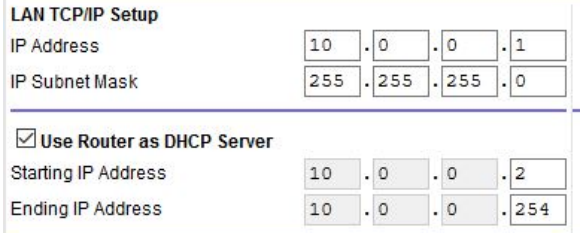
11111111.11111111.11110000.00000000

It is 20 ones followed by $32-20=12$ zeroes. It's usually written in decimal notation, which is 255.255.240.0 in this example. In general, the subnet mask associated with a $/n$ network has n ones followed by $32-n$ zeroes.

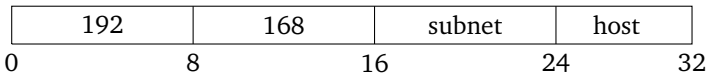
The reason for this notation is that routers can take the bitwise AND of the subnet mask and the IP address to quickly pull out the network portion of the address. The bitwise AND operation is very quick for computers to do. Personally, I find CIDR notation easier to deal with than subnet mask notation, but you will find both in common use. For instance, below is the output I got from running the `ipconfig` command in Windows on my home network (use `ifconfig` if you're on Mac or Linux).

```
IPv4 Address. . . . . . . . . . . : 192.168.1.108
Subnet Mask . . . . . . . . . . . : 255.255.255.0
Default Gateway . . . . . . . . . : 192.168.1.1
```

If we convert the subnet mask to binary, we would see 24 ones followed by 8 zeroes. So this is a /24 network. In particular, it is 192.168.1.0/24, which has addresses from 192.168.1.0 to 192.168.1.255. Below is a screenshot from the router setup page of a different router I own. We can see from the subnet mask, as well as the list of addresses below, that it is using 10.0.0.0/24, and we could configure it to use a bigger network by changing around the subnet mask.



Creating subnets Suppose we have a large address space, like 192.168.0.0/16 to work with. We could use it as one large network with 65,534 devices on it, but that can be a pain to manage. Sometimes we want to break it up into smaller *subnetworks* or *subnets*. To do this, we borrow some bits from the host portion of the address to create subnetworks. For instance, we could use the entire third block (bits 16-23) for the subnet and the rest (bits 24-31) for the host.

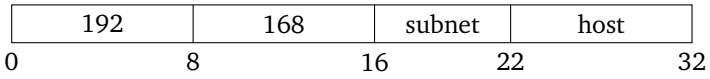


Since we have 8 bits for the subnet and 8 bits for the host, this will give us $2^8 = 256$ subnets and $2^8 - 2 = 254$ hosts on each subnet. The first few subnets are listed below.

192.168.0.0 – 192.168.0.255
192.168.1.0 – 192.168.1.255
192.168.2.0 – 192.168.2.255
192.168.3.0 – 192.168.3.255

Putting the division at the 24th bit, makes the scheme particularly simple. To get to a new subnet, we just increase the third block by 1.

Suppose we need subnets larger than 256 hosts, maybe a subnet large enough for 1000 hosts. Then we would have to move the subnet host division left somewhat. As we saw above, if we have 10 bits for the host, then we can fit $2^{10} - 2 = 1022$ hosts. This will leave 6 bits for the subnet, so there will be $2^6 = 64$ subnets, each with 1022 hosts. Here is what the breakdown will look like.



To figure out the range of addresses in each subnet, it helps to write things in binary. There's no point in converting the 192 and 168 to binary, so we won't. But we will convert the last two blocks. The first few subnets are shown below with the host part (the last 10 bits) highlighted.

192.168.000000**00.00000000** – 192.168.000000**11.11111111** (192.168.0.0 – 192.168.3.255)
192.168.000001**00.00000000** – 192.168.000001**11.11111111** (192.168.4.0 – 192.168.7.255)
192.168.000100**00.00000000** – 192.168.000101**11.11111111** (192.168.8.0 – 192.168.11.255)
192.168.000110**00.00000000** – 192.168.000111**11.11111111** (192.168.12.0 – 192.168.15.255)
192.168.001000**00.00000000** – 192.168.001001**11.11111111** (192.168.16.0 – 192.168.19.255)

Notice that the subnet portion is going up by 1 in binary at each step, namely 000000, 000001, 000010, 000011, 000100. Notice also that the decimal addresses are going up by 4 in the third column.

Getting an IP address

IP addresses are managed by a group called the Internet Assigned Numbers Authority (IANA). They give out IP address blocks to regional internet registries (RIRs). The one in charge of addresses for the U.S. is the American Registry for Internet Numbers (ARIN). ARIN is who to contact if you want your own block of IP addresses. IPv6 addresses are given out pretty readily, but since IPv4 addresses are all but used up, you have to jump through some hoops to get a block of them. There is also a marketplace for IP address blocks that has developed.

If you just need a single IP address, you might be able to get one from your ISP. Or if you rent a virtual private server, that virtual machine will often have its own IP address.