The Internet Protocol

The Internet Protocol (IP) is the most important protocol at the network layer. Below we will look at the different parts of the IPv4 header and how they work.

Ver.	IHL	TOS/DS	Total Length	
Identification			Flags	Fragment Offset
TTL		Protocol	Checksum	
Source Address				
Destination Address				
Options				

The header consists of 5 rows of 32 bits, followed by some options that are rarely included. So the IP header is typically 20 bytes. Let's look at each part of the header.

Version The first four bits are for the version number. This is always set to 4 since this is IPv4.

IHL IHL is short for internet header length. It takes up four bits of the header. It indicates how many rows long the header is. The purpose of this is to help networking software know where the header ends and the data begins.

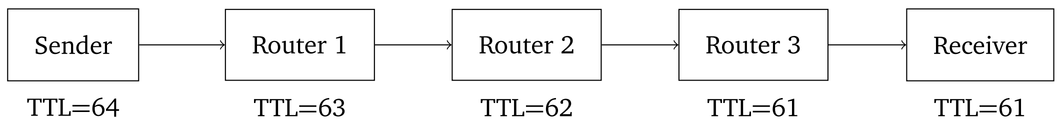
TOS/DS This 8-bit field has changed meaning over the years. Originally, TOS stood for Type of Service, and it has now morphed into Differentiated Services and Explicit Congestion Notification (ECN). The basic idea of Differentiated Services is that it gives a way to prioritize certain types of traffic over others. For instance, streaming video should get higher priority over some OS updates that are being downloaded in the background.

The last two bits of this field have been redefined for ECN, which is used together with an ECN flag in the TCP header. If a router is experiencing congestion, it can set the ECN bits in the IP header to let the let the receiver know about the congestion. The receiver then sets the ECN flag in the TCP header in its ACK back to the sender so that the sender knows to slow down. This is a means of congestion control that avoids packet loss.

Total length The last part of the first row is the total length of the IP header and data combined. It is a 16-bit field, which means the maximum length of an IP packet is $2^{16} - 1 = 65{,}535$ bytes. However, most internet traffic runs at some point over ethernet, which has a 1500-byte limit, so most IP packets will be no larger than 1500 bytes.

Row 2 of the header We will cover this a little later when we talk about fragmentation.

TTL This is short for time to live. Each router decreases the value of the TTL by 1 before forwarding the packet to the destination. This is shown in the example below.



If the TTL ever reaches 0, the router that the packet is at will drop the packet, and it may send back an error message to the sender indicating that the packet's TTL expired. This message is an ICMP "Time Exceeded" message. We will cover ICMP at a later time.

What is TTL for? Sometimes packets get caught in routing loops, where they could bounce around endlessly between the same routers. TTL is a way to kill packets that are on the network for too long. Without it, the network would eventually fill up with these packets.

On the internet, most packets take no more than around 30 to 40 hops to get to their destination, so to be safe, the sender should make sure the TTL is larger than this. The TTL field is 8 bits, which means the maximum TTL is $2^8 - 1 = 255$. Most Windows systems set the TTL to 128 and most Mac and Linux systems set it to 64.

Protocol This is an 8-bit value that indicates what type of traffic is being carried by this packet. Each type has a corresponding numerical code that goes in this field. The most common types are TCP, UDP, and ICMP.

Checksum This is a 16-bit checksum, taken over only the content of the IP header. It is computed in the same way as the TCP and UDP checksums.

Addresses The source and destination addresses are each 32 bits in the header, meaning there are $2^{32} \approx 4$ billion addresses, theoretically.

Options There are a variety of options, but most of them are not used, often for security reasons. One interesting one is source routing, which allows a sender to specify the sequence of routers they want the packet to go through.

Just for fun, here is a capture from Wireshark showing the entire contents of the IP header for a particular packet.

```
Internet Protocol Version 4, Src: 192.168.1.1, Dst: 192.168.1.116
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
    Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
        0000 00.. = Differentiated Services Codepoint: Default (0)
        .... ..00 = Explicit Congestion Notification: Not ECN-Capable Transport (0)
    Total Length: 377
    Identification: 0x0000 (0)
    Flags: 0x4000, Don't fragment
        0... .... .... .... = Reserved bit: Not set
        .1.. .... .... .... = Don't fragment: Set
        ..0. .... .... .... = More fragments: Not set
    Fragment offset: 0
    Time to live: 64
    Protocol: UDP (17)
    Header checksum: 0xb5ae [validation disabled]
    [Header checksum status: Unverified]
    Source: 192.168.1.1
    Destination: 192.168.1.116
```

Fragmentation

Row 2 of the IP header is dedicated to fragmentation. The purpose of fragmentation is if a packet needs to be forwarded on a line which can't handle a packet of that size, then the packet is broken into pieces small enough to fit, called *fragments*. The stuff in row 2 of the header is mostly there to help with reassembling those fragments.

The three key parts of the header for this purpose are Identification, Fragment Offset, and the MF (more fragments) flag. The Identification field is used to indicate what packet the current fragment belongs to. It's possible that fragments from multiple different packets could arrive, and this field is used to help know which fragments go with which packets.

The Fragment Offset field is used to determine where the current fragment fits in the overall set of fragments for a packet. It is a byte number, sort of like the TCP sequence number.

The MF flag is used to indicate if there are more fragments coming or not. If there are, it's set to 1, and if there aren't then it's set to 0. This is how the receiver will know what the last fragment is; the MF flag will be 0 for it.

There are two other bits in row 2 of the IP header. One of those bits is a reserved bit that senders must set to 0. The other bit is the DF (don't fragment) flag. It tells a router not to fragment a packet. If the packet is too large, and the DF flag is set, then it is dropped and an ICMP error message may be sent back to the sender indicating this. It is used for something called *path MTU discovery*, covered a little later, which is a way for a sender to figure out the largest packet they can send without it having to be fragmented.

Fragmentation was an important part of IP when it was first created, but it's not much used in the modern internet. Fragmentation has been used to evade firewalls. Some firewalls determine whether or not to accept a packet by looking at things like port numbers and TCP flags. A packet can be cleverly fragmented so that the header info is broken across fragments, making the fragments able to sneak by the firewall.

IPv6

Back in the 1990s it started to become apparent that we would eventually run out of IPv4 addresses. Those addresses are 32 bits, meaning there are around 4 billion addresses theoretically possible, though because of the way the address space was divided up, there are considerably fewer possible addresses. In the late 1990s, IP version 6 was developed to deal with the lack of addresses. IPv5 was the designation given to an experimental protocol that never went anywhere.

IPv6 simplified some facets of IP and replaced 32-bit addresses with 128 bit addresses. That means that whereas there are 2^{32} possible IPv4 addresses, there are 2^{128} possible IPv6 addresses. To give a sense for how large this number is, below we show what these values equal.

$2^{32} = 4294967296$
$2^{128} = 340282366920938463463374607431768211456$

Also, just for effect, here is what an IPv6 address would look like if written in the same notation as IPv4 addresses:

10.47.112.90
10.47.112.90.254.11.67.6.103.14.97.212.144.192.87.206

Dotted decimal notation is too unwieldy to use for IPv6 addresses, so a different scheme is used. They are broken into 8 groups of 4 hex digits. Here is an IPv6 address of Google, shown in two different ways:

2607:f8b0:4004:0802:0000:0000:0000:200e
2607:f8b0:4004:802::200e

The first way shows all 32 hex digits written out. IPv6 addresses tend to have long runs of zeros, and the address can be displayed in a shortened form called *zero-compressed form*, where a run of all zeros is replaced with a double colon (::). This can only be done once in the address, as otherwise it would be ambiguous. To convert a zero-compressed address into its full form, add enough zeros until you get 8 full blocks. The address abcd::1234 would require 6 blocks of zeros to get to 8 blocks, becoming abcd:0000:0000:0000:0000:0000:0000:1234.

IPv6 address structure CIDR notation is used for IPv6 similarly to how it is used for IPv4 but with one important difference. In IPv4, a /24 network has 24 bits for the network and $32 - 24 = 8$ bits for the host, meaning $2^8 = 256$ addresses on that network. In IPv6, a /24 would mean 24 for the network and $128 - 24 = 104$ bits for the host, meaning 2^{104} addresses on that network, a mindbogglingly huge number.

IPv6 addresses are generally broken up into 3 parts: 48 bits for the network ID, 16 bits for the subnet ID, and 64 bits for the host ID. Unlike in IPv6, this is not variable. The network portion of the address is never allowed past the 64th bit.

So if you ask for a block of addresses and get a /48, you will get a network large enough to accommodate 2^{16}=65,536 subnets of 2^{64} =18,446,744,073,709,551,616 hosts each. This is of course more than almost anyone would reasonably need, and it's led to some people complaining about the IPv6 address space being wasted in a similar way to how the IPv4 address space was initially wasted. IANA hands out networks to organizations that ask in sizes of /48 or possibly larger (like /32). You can't get something small, like a /104. The address space is really huge, so this doesn't seem to be a problem right now to be handing out such large networks.

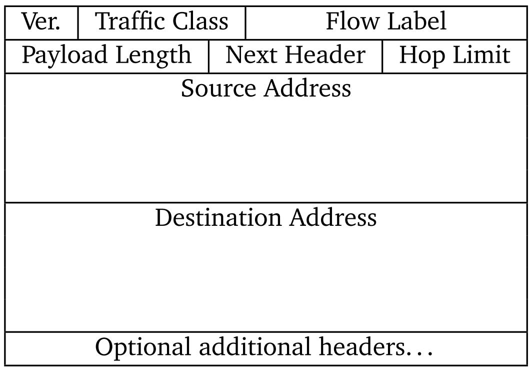
The reason that such big networks are handed out is that it makes life a lot easier for routers. Routers use the network part of an address along with a routing table to know where to forward packets. The more complicated the routing scheme is, the larger and slower the routing table will be. The growth of routing tables is already a problem for IPv4, and people want to keep it from being a problem in IPv6.

Using IPv6 All major operating systems support IPv6. If your home router is fairly new, then it supports IPv6. Your ISP may support it or it may not.

Dual Stack is where a system can use IPv4 or IPv6 and can use whichever one it needs in a given situation. *Tunnelling* is used if you are on an IPv4 network and need to contact someone using IPv6. Since you're on an IPv4 network, you can't send an IPv6 packet directly, so instead you put your IPv6 packet into an IPv4 packet as its data. That IPv4 packet can then travel through your network, and when it gets to the transition point between IPv4 and IPv6, the packet is "unwrapped" and the IPv6 packet is pulled out of the IPv4 packet. Tunnelling is a broader concept than just this. It applies to any situation in which one type of protocol is hidden inside another. The concept is important in networking and security.

Though it's been around for over 20 years, IPv6 has still not taken over for IPv4. As of 2022, Google reports that around 40% of users accessing their sites are using IPv6, with the rest using IPv4. Part of the reason for this is that it takes some effort for network administrators to convert their systems from IPv4 to IPv6. As long as IPv4 still works, unless IPv6 provides them with some significant benefits, there's not much point in taking something that works and replacing it with a new system and all the glitches that come with doing something new. At least for now, though it's not perfect, Network Address Translation (NAT) seems to be solving the problem of the lack of IP addresses. NAT allows multiple users to share the same IP address. We will have more on it later. Another part of it is psychological. People are comfortable with IPv4 and IPv6, with its huge addresses, looks really strange.

The IPv6 header Below is what the IPv6 header looks like.



The version is set to 6. The Traffic Class and Flow Label fields are used to prioritize certain types of traffic over others. The Payload Length field is the length of the packet's data along with the length of any optional headers. It is 16 bits, which limits IPv6 packets to 65535 bytes. However, there is a jumbogram optional header that allows for larger packets. The Next Header field is used to help software process the headers and know where the packet data starts. The Hop Limit field is the TTL, renamed to more accurately describe its function.

The IPv6 header is considerably simpler than the IPv4 header. One of the goals of IPv6 was to simplify things for routers. Gone is fragmentation, which is moved to an optional additional header. In IPv4, fragmentation could be done by either the original sender or any of the intermediate routers, but in IPv6, it's only done by the original sender. The receiver is the one that puts the fragments back together.

Also missing from the IPv6 header is the checksum. The problem with the checksum in IPv4 is that the checksum is done on the header content, and the TTL changes at each router, so the checksum has to be recomputed by each router. This adds additional work for the router, which adds up when it's processing millions of packets. Both TCP and UDP, as well as layer 2, have checksums, which makes the IPv4 checksum is somewhat redundant.

Types of traffic IPv6 support unicast, multicast, and *anycast*. Anycast is similar to multicast in that the packet can go to multiple different destinations; however, instead of going to all of them, it goes to just one, whichever is the most convenient. It's a little like how a phone call to a help center could theoretically go to any of the lines in the help center, but it will end up going to the first available line. Note also that IPv6 does not specifically have broadcast addresses. Broadcast is achieved via multicast.