8 Internet Scaling

8.1 Goal of this section

The goal of this section is for readers to understand how IP routing scales to a global level. We'll look at problems with addressing the global population and learn how we can use private address spaces and Network Address Translation to scale IPv4, and we'll also dive into more detail on IPv6 and how this helps. We'll also look at Border Gateway Protocol and how this enables us to scale IP routing globally.

8.2 Global level problems: Scaling and Politics

As indicated at the end of the last section, as we scale up to internet scale, we hit two major problems – the sheer technical scaling problems, and human politics, particularly around the issue of trust. This chapter covers some of the main ways that we address these problems.

8.2.1 Scaling

In 2019 there were 360 million different top-level domain names in the world, (O)10m-100m routers, and (O)bns of endpoints – and with internet-enabled mobile devices, that number just keeps growing.

Checking in with what we know so far about IP addresses and IP routing, we're coming up a bit short.

- There are 4bn IPv4 addresses, but these are allocated out in blocks that are distributed and arranged geographically for routing to work. The last /8 subnet block was handed out to be split up in 2011, and the last /22 IPv4 subnet block was allocated near the end of 2019. We're out of addresses.
- Distance Vector and Link State routing protocols that we have scale to (O)100 routers. With tweaks and complex enhancements, some of the largest networks can reach (O)1000 routers. A far cry from the 10m that we need.

8.2.2 Politics

While for the internet to work, all the various companies involved need to work together, most of those companies are also competitors – typically running off very narrow margins. How a company's network is set up and configured to squeeze the maximum performance and redundancy out of a set of hardware is often a competitive advantage, and so while companies do need to advertise what destinations and routes go through their networks, they don't want to reveal the insides of their network to their competitors.

In addition, people and companies want to protect themselves from bad actors accessing and corrupting their kit, and a good way to do that is to prevent those bad actors knowing what exists – again preventing knowledge escaping outside the network.

8.3 Private addressing

IANA has designated the following sets of IPv4 address subnets as private.

- 10.0.0.0/8
- 172.16.0.0/16 – 172.31.0.0/16
- 192.168.0.0/24 – 192.16.255.0/24

These addresses may be used by anyone subject to the explicit restriction that they must not be routable from the rest of the internet (the public internet). Within that restriction, these addresses work exactly as any other IP address subnets – IP addresses are arranged within those subnets, etc.

In addition, users may choose to split those subnets into smaller subnets and use routing between them in their network. A user may make their own network as complicated as they wish, with whatever routing rules they wish, provided there is no routing from any of these addresses reachable from the public internet.

The crucial advantage of these is that because it is not possible to route in or out of your private network space using these IP addresses, these do not need to be globally unique, and there can be many, many independent private networks using the same addresses. If you are at home or in a university computer lab, then if you check your IP address, you will almost certainly see it is in one of the private address spaces.

The above give private networks two excellent properties. Firstly, the non-uniqueness helps us scale with the limited pool of IPv4 addresses that we have available. Secondly, because computers with private addresses must not be routable from outside your network, this hides your computer from bad actors outside your organisation.

8.4 Network Address Translation/Translator (NAT)

Having a private network is all very well, but not being able to route in or out of it to the rest of the world is quite a restriction! We get around this and enable internet connectivity to a private network using a special router called a NAT.

This NAT is a router that has a private IP address on the private network side, so it can act as a default router for all the hosts on the private network. It also has a public IP address on the other side, so it can route and provide routing to the public internet. In addition to normal routing, the NAT also rewrites the IP addresses for the private IP addresses (and ports – which although further up the stack, we touched on in section 6) in all packets that travel through it, so that to the outside all of the IP packets appear to originate from it (and all replies are sent to it) and on the inside the packets travel on the private network to the correct real destination hidden behind the NAT.

Note that NAT is used slightly loosely to both mean Network Address Translation (as a verb – the function of performing swapping of addresses) and Network Address Translator (as a noun – the device that is performing Network Address Translation). The NAT is NATing the traffic in and out of the private network.

A NAT is limited in its function in a few crucial ways.

- Speed (processing power). The NAT has to rewrite IP headers on all packets (and recalculate checksums) going through the router. This is a significant amount of work compared with the simple match-and-forward rules of a normal router.
- Intelligence. NATs change the IP addresses on the IP headers of packets. But what about IP addresses that appear inside the packet payload? Some NATs have specific intelligence to understand many common protocols and go digging inside the packets to fix up additional IP address information. This in turn is more computationally expensive. Any IP address missed will result in failure when the protocol using that IP address tries to connect directly.
- Outgoing connections only. When a host on the private network wants to communicate out to the public internet, it can do that routing to the public IP address (and the NAT can set up the mappings, so that the public host can respond to it). However, there is no way for a public host to open a connection to the private host. In practice this isn't quite true – it is possible for an administrator to pre-configure the NAT with some standard allowed incoming mappings – for example to a central email server.

8.5 IPv6 (RFC ~~1883~~ ~~2460~~ 8200)

Internet Protocol Version 6 is an updated and improved version of IPv4. It was initially defined in RFC 1883 in 1995, and has been updated and refined significantly since then (the latest version definition is in RFC 8200, in July 2017). Over the years since usage has grown, and most hosts, routers and protocols now support IPv6 – and it is deployed alongside IPv4 in the internet today.

There are lots of differences between the two versions (which we'll cover below), but the key one that solves our scaling problem is that of the address space. IPv4 addresses are 4 bytes long, whereas IPv6 addresses are 16 bytes long – which gives a huge address space for us to use.

8.5.1 IPv6 IP Header

IPv4 Header (reminder)

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |Version|  IHL  |Type of Service|          Total Length         |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |         Identification        |Flags|      Fragment Offset    |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |  Time to Live |    Protocol   |         Header Checksum       |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                       Source Address                          |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                    Destination Address                        |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                    Options                    |    Padding    |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

IPv6 Header

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |Version| Traffic Class |           Flow Label                  |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |         Payload Length        |  Next Header  |   Hop Limit   |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                                                               |
   +                                                               +
   |                                                               |
   +                         Source Address                        +
   |                                                               |
   +                                                               +
   |                                                               |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                                                               |
   +                                                               +
   |                                                               |
   +                      Destination Address                      +
   |                                                               |
   +                                                               +
   |                                                               |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

We'll look at the IPv6 header first, and then compare the differences with the IPv4 header.

IPv6 Header contents

- **Version** (4 bits). This is "6" and indicates that this is an IPv6 header and needs to be parsed as such.
- **Traffic Class** (8 bits). 8 bits used by the network to assign priorities for traffic (similar to the Type of Service) fields from IPv4.
- **Flow Label** (20 bits). This is used by the source to label sequences of traffic that are part of a single flow. It is not required, but allows sources to label sets of packets as conceptually connected.
- **Payload Length** (2 bytes) This is the number of bytes in the packet after the IPv6 Header.
- **Next Header** (1 byte) IPv6 allows additional optional headers after the IPv6 header. This field is an 8-bit selector and identifies the type of the first of these. (Each additional IPv6 header identifies the type of the next following header in the chain, until the last one indicates no further headers)
- **Hop Limit** (1 byte) This is the Time-to-live from IPv4 rebranded. Decremented by 1 by each router.
- **Source IP address** (16 bytes) The source of the IP packet.
- **Destination IP address** (16 bytes) The destination of the IP packet.
- **(Additional optional headers)** These optional headers are a chain of headers that follow on from the header as described in the Next Header field above.
- **(Payload)…**

Key differences from IPv4:

- The IPv4 header includes several fields for services (such as fragmenting of packets). In IPv6 all these are instead chained optional extension headers (linked from the next header field). This gives more flexibility.

- No header checksum. With more modern data-link layers, the integrity is high enough that the lower-layer checksum handling is fine – and this removes unnecessary work from all routers.
- Address size. With 16 bytes of address, IPv6 has lots of space. However, note that this breaks lots of other existing protocols that expect IPv4 addresses. For example, looking back at the ARP protocol (linking IPv4 addresses to MAC addresses), there is no space to handle IPv6 addresses. The IPv6 ARP equivalent is NDP (neighbour discovery protocol).

8.5.2 IPv6 addresses

As mentioned above IPv6 addresses are 16 bytes long, but in other respects, IPv6 addressing and subnetting works exactly the same as for IPv4. IPv6 addresses are written by humans in "colon-separated-hex" – 8 sets of 4 hex characters, separated by colons. Because IPv6 addresses also often have lots of :0000: blocks in the middle, the largest consecutive set of :0000: blocks is condensed to :: . So, for example the address ABC3:2340:0000:0000:0000:0020:0000:0001 becomes ABC3:2340::0020:0000:0001.

Subnets in IPv6 are handled and written in the same way as for IPv4, but go up to /128 (16 byte address space).

8.5.3 Special IPv6 addresses

As with IPv4, there are address ranges used for specific purposes – for example:

- ::1/128 Localhost. Referring to this computer. (More correctly written ::0001/128)
- 2001::/16 Documentation. None of these addresses may be used in live systems.
- FC00::/7 Private addresses. These are the same as IPv4 private addresses. Note though that the subnet space here is large enough that everyone could easily fit the entire public internet in their own private address space.
- FF00:/8 Global unicast. These are globally routable IPv6 addresses.
- FE80::/10 Link-local addresses. These are IPv6 addresses that are autogenerated on a link based on the MAC addresses of that link. This gives "plug and play", meaning that when you plug an IPv6 computer into a network, it immediately (without further configuration) has IP connectivity to other computers on that link.

8.6 Border Gateway Protocol (BGP) (RFC 4271)

We've covered scaling and some privacy handling for addresses at internet scale. How does routing work? Before we start, a few key pieces of terminology:

All the routing protocols that we covered last chapter (OSPF, RIP, etc.) are used within networks with one controller.

- Such networks are called **Autonomous Systems (ASs)**.
- Such routing protocols are called **Interior Gateway Protocols (IGPs)**.

To route at internet scale, we use additional routing protocols that connect between ASs, to connect everything at internet scale.

- Such routing protocols are called **Exterior Gateway Protocols (EGPs).**

We've already covered various IGPs. There is de-facto only one EGP, called Border Gateway Protocol (BGP). BGP has built into it both the scaling and privacy/abstraction needed to run as an effective EGP. It *also* has built into it the relevant ability for humans to mess with and override the routing logic needed for real world internet routing.

Remember that between IGPs, although companies need to share routes and forward data, in general companies

- don't want to reveal the internals of their network
- want to charge people for using their network
- want to do as little work as possible.

8.6.1 BGP Abstract

BGP routers sit on the edge of IGP networks and are manually configured with adjacencies (a link between two BGP routers) to their neighbours. BGP adjacencies are configured manually, and typically come as part of a contractual agreement between two companies. The contract will cover who will provide what information, what routing, how much data and costs, etc. etc.

BGP adjacencies are either internal to other BGP routers within an AS (called iBGP), or external to other ASs (eBGP). BGP runs like a distance vector protocol, but instead of each "hop" being a router, each "hop" is an entire AS. So for example a route that traverses 5 ASs is lower cost (and therefore better) than a route to the same destination that traverses 6 ASs – even if the 6 ASs are actually really small and the number of actual routers involved is smaller. Remember, the number of routers in an AS is typically secret within that AS.

BGP includes the entire path of ASs traversed, rather than just the route "cost". This allows other routers to spot and prevent loops when a route would go back through an AS – preventing the distance vector counting-to-infinity problem.

BGP is also a hard state protocol – routers explicitly advertise and withdraw routes, and routes do not time out over time. This prevents BGP routers having to regularly re-advertise enormous (internet-wide) lists of routes around the internet every short space of time.

Flexibility

BGP has really become the workhorse of the internet, partly due to some explicit flexibility in the design on top of the above:

- BGP doesn't just work for IPv4 or IPv6. It provides the ability to pass routing information about any kind of destination (with addresses of arbitrary size and structure).
- BGP also allows users to associate almost any additional information with a destination (not just raw path information), making it very flexible and allowing distribution of other information at internet scale.

Although BGP has a standard way that it calculates the "best" route given a set of routes, it allows configuration options that allow humans to change how the best route is chosen, or to prioritise, drop, or even change and edit any route received from a peer – or indeed any route about to be advertised to a peer. This allows administrators almost complete flexibility to agree how internet routing will work for them and their neighbours.

8.6.2 BGP Protocol Implementation

The BGP protocol itself consists of 5 messages for setting up connections and passing around routing information. On top of this, BGP provides a well-defined method for calculating the "best" route to advertise onwards, and a way for administrators to edit and alter routes pretty much as much as they want, called BGP Policy.

Protocol Messages.

There are 5 protocol messages, of which 4 are very simple.

- **Open** – An initial message exchanged with a newly configured peer. This contains authentication information to prove that this really is the neighbour that you think you're opening a BGP session (connection) with, and other parameters for the session.
- **Keepalive** – A keep alive message swapped frequently (default 30 seconds) to keep the session alive and reassure your neighbour that you are still alive and able to handle routed traffic.
- **Notification** – An error message sent when something goes wrong, to tear down the session. BGP includes a huge list of error codes and sub codes to indicate what has gone wrong.
- **Route Refresh** – A request to your BGP neighbour to send you all their routes – for example if you suspect that you might have got out of sync. As a side note, this is technically a separate piece of function from base BGP (defined in RFC (2918)) but it is mandatory for anyone who wants to run BGP in the internet.
- **Update** – The BGP Update advertises, updates, and withdraws routes to destinations. This message does all the work. We'll unpick in more detail below.

BGP Update

A BGP update message contains two key sets of information:

- **Network Layer Reachability Information** (NLRI)s. These are the destinations being advertised by the update message. They may be new destinations or may be existing destinations. This message overwrites any previous information for those destinations.
- **Attributes**. These are the properties of the routes. Note that there is one set of attributes on an update message and this is applied to all the NLRI advertised.
- **Withdrawn Routes**. The Update message can *also* include a set of explicitly withdrawn NLRIs. The attributes don't apply to these, as these are not being updated – they're being withdrawn.

Key Attributes

There are many, many attributes that can be included in a BGP Update, but there are a few critical ones that are crucial for BGP routing calculations, which we'll cover here.

- Next Hop. The next hop router for the NLRIs. Note that this Next Hop is the BGP next hop, which is not necessarily the IP next hop – even for IP routes, and when forwarding data, the BGP router may have to resolve the actual IP next hop from the BGP next hop using the IGP routing protocol.

  For routes being advertised to iBGP peers, the next hop is left unchanged. For routes being advertised to eBGP peers, the BGP router sets the next hop to itself.

- AS Path. The set of ASs that the path via this next hop traverses. This gives the crude measure of the cost of the route, and also prevents routing loops.
- Local Preference. An administrative hack used for influencing policy decisions. Local preference is only distributed *within* an AS, and is used within an AS to help the BGP routers in the AS decide which routes to prefer.

BGP Routing Decision

The BGP can learn several routes to an NLRI (destination). For each destination it chooses the best route, which it then advertises to its BGP peers and also to its IGP within the AS.

The decision process compares route attributes in a well-defined order. The key points are as follows:

- Prefer the highest local preference. (the within-the-AS admin hack trumps everything).
- Prefer the shortest AS Path. (Lower cost routes are better)
- Prefer routes learned from the IGP over BGP. (If the IGP has a route, it's something within the AS, and the IGP route will be more precise.)
- Prefer eBGP routes over iBGP routes. (Cheaper to send the data to someone else…)

8.6.3 BGP Policy

BGP policy is a flexible framework that allows complete administrative control over your BGP router's behaviour. It can be applied on import (when routes are learned from peers, before they are considered), and also on export (before routes are sent to peers). BGP policy allows the user to create rules that:

- match routes based on any aspect of the route
- permit or deny the route
- edit any of the route attributes (typically either to influence your own or your peer's routing decisions).

For example, you could refuse to accept any routes that traverse AS 1234, you could decide not to advertise any routes to a particular peer for destinations in the region 17.55.0.0/16, or you could change the local preference of routes via a particular next hop to be 100.

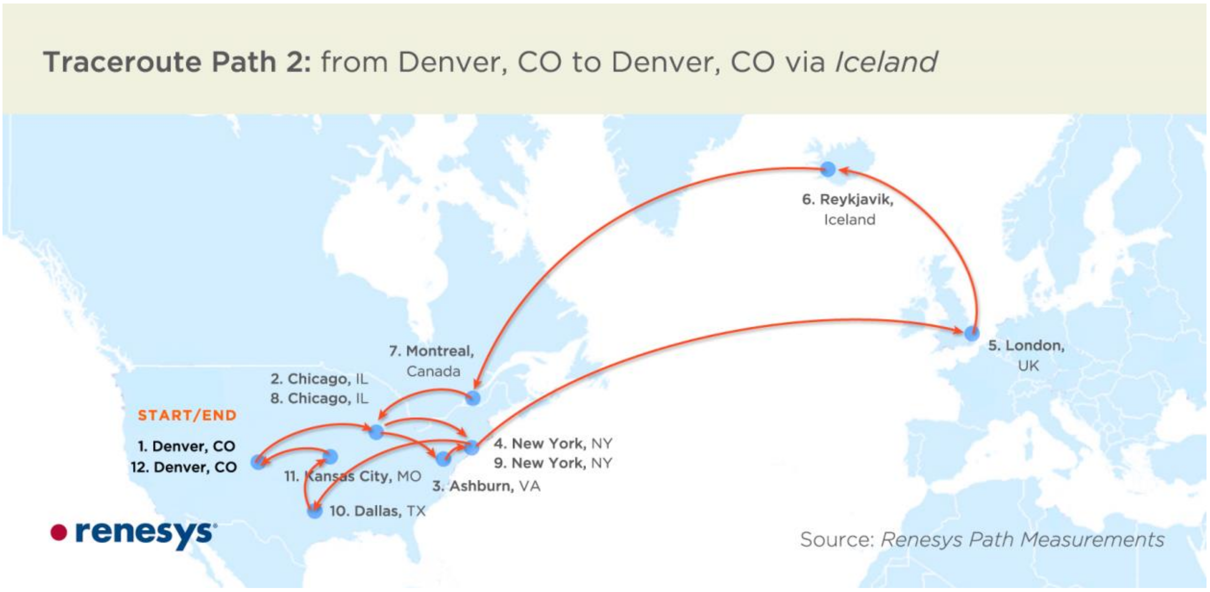
Why would you do this? Some examples:

- The military or intelligence service from country A don't want traffic to go via routers in country B.
- A particular ISP has been a bit flaky recently, so you route around their network until it becomes more stable.
- You have a sharing agreement with another long distance carrier company to split traffic on a particular route between yourselves, using each other as backup.

Some real world examples of things that can be done:

- https://www.zdnet.com/article/russian-telco-hijacks-internet-traffic-for-google-aws-cloudflare-and-others/
- https://arstechnica.com/information-technology/2017/12/suspicious-event-routes-traffic-for-big-name-sites-through-russia/
- https://dyn.com/blog/backconnects-suspicious-bgp-hijacks/

And lest you think this is a "Russia only" problem, here's an example routing flow from 10 years ago, when the CIA was loudly touting that they didn't tap American-American traffic within the USA…



8.6.4 BGP Default-Free core

Most IP routes have a default route, which is shorthand for "Any destination I don't know what to do with, punt traffic to this router that knows more than me." The central core of the internet doesn't have such defaults and genuinely has routes to everywhere. This is called the BGP default-free core, and currently has ~850k routes in: https://bgp.potaroo.net/index-bgp.html

8.6.5 Example flow

08_BGP.pcap shows a BGP connection starting between two BGP neighbours – showing an initial open message, some keepalives and some update messages.

You will notice that the trace also includes a load of TCP messages (BGP runs over TCP), and also that packet 16 contains 3 BGP messages – a KEEPALIVE and two UPDATEs (which is another artefact of BGP running over TCP).

For now, ignore the TCP messages – we'll come on to TCP later in the course.