The ISP Column
A monthly column on things Internet

February 2011
Geoff Huston

Transitioning Protocols – Part 1

In a previous article, in November 2010, I looked at some common myths associated with the transition to IPv6. This month I'd like to look behind the various opinions and perspectives about this transition, and look in a little more detail at the nature of the technologies being proposed to support the transition to IPv6.

After some time of hearing dire warnings about the imminent exhaustion of the stocks of available IPv4 address space, we've now achieved the first milestone of address exhaustion, the depletion of the central pool of IANA-managed address space. The last 5 /8s were handed out from IANA to the RIRs on the 3rd February. After some years of industry-wide general inattention and inaction with IPv6 perhaps it's not unexpected to now see a panicked response along the lines of "Maybe we should do something now!"

But what exactly should be done? It's one thing to decide to "support" IPv6 in a network, but quite another to develop a specific plan, complete with specific technologies, timelines, costs, vendors, and a realistic assessment of the incremental risks and opportunities. While working through some of this detail has the normal levels of uncertainty that you would expect to see in any environment that is undergoing constant change and evolution, an additional level of uncertainty here is a by-product of the technology itself.

There's not just one approach to adding support for IPv6 in your network, but many. And it's not just one major objective you need to address, namely incremental deployment of IPv6 as second protocol into your operational network without causing undue disruption to existing services, but two, as the second challenging objective is how to fuel continued growth in your network service platform when the current supply lines of readily available IPv4 addresses effectively dry up.

When?

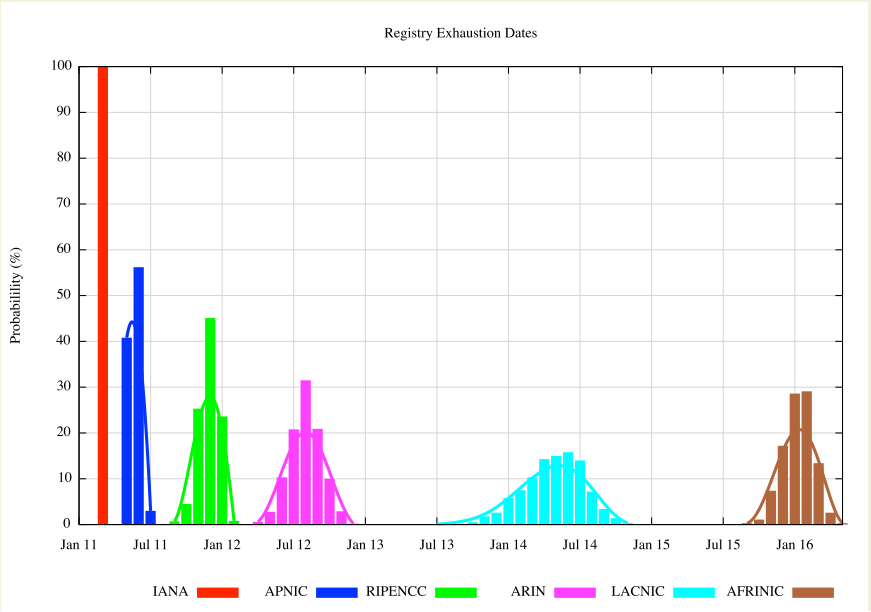
The most common question I've heard recently is: "How long do we have?

The remaining pools of IPv4 address space continue to be drawn down. At the start of February 2011 the IANA pool was fully depleted with the final allocation to the RIRs of IPv4 addresses.

As an update on the September ISP Column the prediction for RIR address exhaustion, using a model based on monthly address demands now looks like the next 18 months or so will see the first three RIRs run dry of IPv4 addresses.

At present, its likely that APNIC be the first RIR to exhaust its available pool of IPv4 addresses, probably in April or May 2011, with the RIPE NCC following in late 2011 and ARIN in mid 2012. The analysis of the probability of the month when this will occur

for each RIR and the associated variance of these so-called "exhaustion times" is shown in the figure below.



The good news is that many folk have been busy thinking about these inter-twined objectives of extending the useful lifetime of IPv4 in the Internet and simultaneously undertaking the IPv6 transition, and there are a wealth of possible measures you can take, and a broad collection of technologies you can use. Fortunately, we are indeed spoilt with choices here!

The not so good news is that many folk have been busy thinking about these inter-twined objectives, and there are a wealth of possible measures you can take, and a broad collection of technologies you cans use. These options may, or may not, be optimal for your particular circumstances, and may, or may not, be useful for you in mitigating address depletion and may, or may not, be consistent with your chosen longer term network objectives. Unfortunately, we are spoiled for choices here!

Let's have a look at each of the major transitional technologies that are currently in vogue, and look at their respective strengths and weaknesses and their intended area of applicability. In this column we'll look at this from the perspective of the end user. In the next part of this article we'll look at this from the other side, looking at options for ISPs.

V6 for End Clients

The Dual Stack ISP Client

If your service provider provides a dual stack service with both IPv6 and IPv4 then your task should be relatively straightforward. Configure your modem/router with IPv6 in addition to IPv4 and you are done, assuming of course that your local modem/router unit actually supports IPv6, which may not be the case in many of the older and, unfortunately, all too many of the currently available devices.

The conventional approach in this form of environment is to use IPv6 Prefix Delegation, where the ISP provides the client with an IPv6 prefix, usually a /48 or a /56 IPv6 address prefix,

which is then passed into the client network via an IPv6 Router Advertisement. Local hosts should be configured to configure their IPv6 stack automatically, and your system should be connected as a dual protocol system.

There are, however, some caveats you probably need to be aware of, of which the most important difference is likely to relate to the probable absence of a NAT function in IPv6. These days most commercial IPv4 Internet services assign a single IP address to each client. To allow this address to be shared within the client's network, most IPv4 'edge' devices auto-configure themselves as a NAT device, permitting outgoing connections using TCP, UDP, and allowing some ICMP message types to traverse the NAT, but not much else. For many clients this NAT configuration becomes the default local security framework, as it permits outbound connections via TCP and UDP to be made, but not much else, and permits no sessions to be initiated as incoming sessions. With iPv6 the local network is probably going to be configured with an entire subnet, and instead of a NAT, this subnet will be directly connected to the Internet.

The local network is then in a mixed situation of being behind a NAT in IPv4, but being directly connected to the Internet using IPv6. This asymmetric configuration with respect to IPv4 and IPv6 raises some questions about the impact to the security of your local network. You need to think about adding appropriate filter rules to the gateway's IPv6 configuration that performs the same level of access control to your local site that you have already set up with IPv4 and the NAT. The best advice here is the need to configure some filter rules for IPv6 that limit the extent of exposure of your internal network to the broader Internet to be directly comparable to the configuration you are using with IPv4.

The IPv4 Only ISP Client

Even today, when the IPv4 pools are running dry, its really not very common to have an ISP offering dual stack IPv4 and IPv6 services. Lets look at the more common situation when your ISP is still only offering IPv4. As an end user, can you still set up some form of IPv6 access?

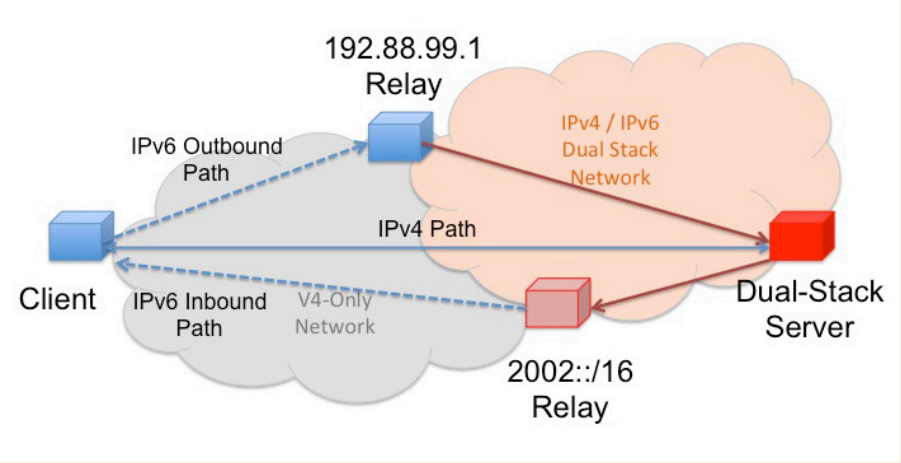
The answer is "Yes", but you are going to have to resort to tunnels, and the story can get somewhat ugly.

6to4 Tunnels

If you have public IPv4 addresses on your local network you may elect to configure your local system to use the 6to4 tunnelling protocol.

> 6to4 is an auto-tunnelling protocol, coupled with an addressing structure. The IPv6 address of a 6to4-reachable host begins with the IPv6 prefix 2002::/16 The address architecture embeds a 32 bit IPv4 address of the end host into the next 32 bits. That way the IPv6 address carries the 'equivalent' IPv4 address within the IPv6 address.
>
> To send an IPv6 packet the local host must first tunnel through the local IPv4 network. To do this the local host encapsulates the IPv6 packet in an outer IPv4 packet header. The IP protocol used is neither TCP nor UDP, but protocol 41, an IP protocol number reserved for tunnelling IPv6 packets (RFC2473). The IPv4 packet is addressed to a IPv4-to-IPv6 relay. To avoid manual configuration of each client all these relays all share the same anycast address, 192.88.99.1. These relays strip the outer IPv4 packet header off the packet and forward the IPv6 packet into the IPv6 network. The IPv6 destination treats the packet

normally, and generates a packet in response without any special processing. The reverse path to a 6to4 host uses a IPv6-to-IPv4 relay. The IPv6 address of the 6to4 local host started with the IPv6 address prefix 2002::/16, so the IPv6 packet that is being sent back to this host has a destination address that uses the 2002::/16 6to4 prefix. This prefix is interpreted as an anycast relay address. A route to the IPv6 2002::/16 prefix is advertised by IPv6-to-IPv4 relays. When they receive a packet destined to a 2002::/16 address the relay will lift the IPv4 address from inside the IPv6 address. They will then wrap the IPv6 packet in an IPv4 packet header, using as a destination address this extracted IPv4 address, and using protocol 41 as the IP protocol. The resultant IPv4 packet is then passed to the 6to4 host in the IPv4 network.



If the local network has public IPv4 addresses on the local network, then individual hosts on the local network may use 6to4 directly. Of course, if this is the case then the local gateway needs to be configured to accept incoming IP packets that use protocol 41.

An alternative is for the local network's gateway device to be configured as a 6to4 gateway, and use the IPv4 address on the ISP-side of the gateway as a common 6to4 address for the local network. The gateway then advertises this synthetic 48 bit IPv6 prefix to the interior network via a conventional IPv6 Router Advertisement. The gateway can couple this with a NAT function and provide native IPv6 to interior hosts who are configured on RFC1918 local IPv4 addresses.

In general, 6to4 is a relatively poor approach to provisioning IPv6, and really should be avoided if at all possible. Indeed, the user experience will probably be better overall if you stay running IPv4 and avoid accessing IPv6 via 6to4.

The major issue here is that a successful connection relies on the assistance of both an outbound and an inbound 6to4 third party relay. On the IPv4 side a 6to4 connection relies on the presence of a useable route to a IPv4-to-IPv6 relay, and preferably one that is as close as possible to the IPv4 endpoint. On the IPv6 side a 6to4 connection relies on a useable relay advertising a route to 2002::/16. Again, to avoid extended path overheads, this relay should be as close as possible to the IPv6 endpoint. This path asymmetry can cause connection 'black holes' where one party can deliver packets to the other but not the reverse.

Also, such configurations have problems if the IPv4 host is configured with stateful filters that insist that the IPv4 source address in incoming packets match the destination address of outgoing packets, which is not necessarily the case in a 6to4 connection.

Finally, it seems that many sites operate with firewall filters that disallow incoming packets other than TCP and UDP (and possibly some forms of ICMP). 6to4 packets using protocol 41, and there appears to be widespread use of filter rules that block such packets.

Tunnelling also adds an additional packet header to a packet, which inflates the size of the packet. Such an expansion of the packet on certain path elements of the network may cause path packet size issues, increasing the risk of encountering path MTU 'black holes' due to the increase of the packet size by 20 bytes when the IPv4 packet header is attached to the packet.

Teredo Tunnels

If the local network is behind an IPv4 NAT, and the NAT gateway does not support 6to4, then all is not lost, as there is another form of tunnelling that could be a possible answer. Teredo is described in RFC 4380.

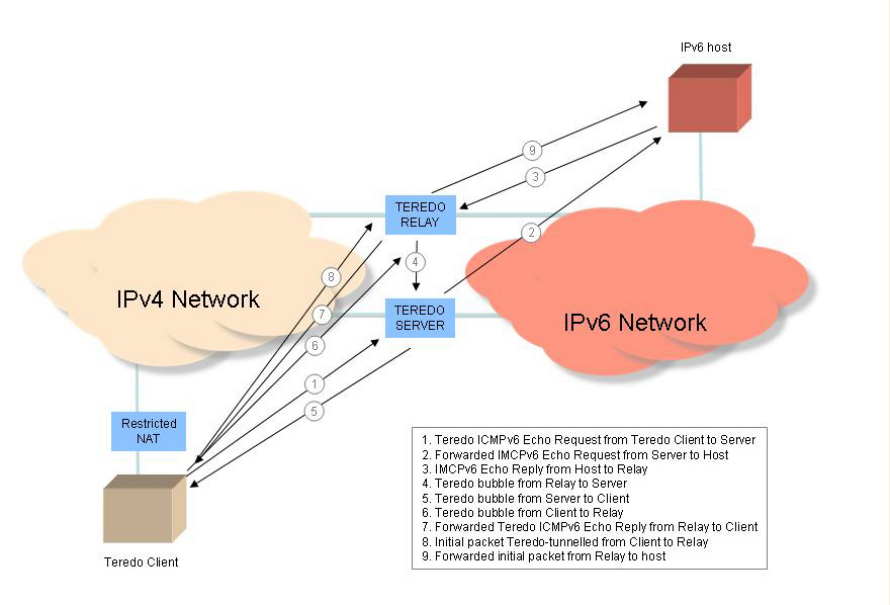
Teredo, like 6to4, is an auto-tunnelling protocol, coupled with an addressing structure. Like 6to4, Teredo uses its own address prefix, and all Teredo addresses share a common IPv6 /32 address prefix, namely 2001:0000::/32. The next 32 bits are the IPv4 address of the Teredo server. The IPv6 interface identifier field is used to support NAT traversal and it is encoded with the triplet of a field describing the NAT type, the relay's view of the UDP port number used to reach the client (the external UDP port number used by the NAT binding for the client) and the relay's view of the IPv4 address used to reach the client (the external IPv4 address used by the NAT binding for the client).

Teredo uses what has become a relatively conventional approach to NAT traversal, using a simplified version of the STUN active probing approach to determine the type of NAT, and uses concepts of "clients", "servers" and "relays". A Teredo client is a dual stack host that is located in the IPv4 world, assumed to be located behind a NAT. A Teredo server is an address and reachability broker that is located in the public IPv4 Internet, and a Teredo relay is a Teredo tunnel endpoint that connects Teredo clients to the IPv6 network. The tunnelling protocol used by Teredo is not the simple IPv6-in-IPv4 protocol 41 used by 6to4. NATs are sensitive to the transport protocol and generally pass only TCP and UDP transport protocols. In Teredo's case the tunnelling is UDP, so all IPv6 Teredo packets are composed of an IPv4 packet header, a UDP transport header, followed by the IPv6 packet as the UDP payload. Teredo uses a combination of ICMPv6 message exchanges to set up a connection and tunnelled packets encapsulated using an outer IPv4 header and a UDP header, and contain the IPv6 packet as a UDP payload.

It should be noted that this reliance on ICMP6 to complete an initial protocol exchange and confirm that the appropriate NAT bindings have been set up is not a conventional feature of IPv4 or even IPv6, and IPv6 firewalls that routinely discard ICMP messages will disrupt communications with Teredo clients.

The exact nature of the packet exchange in setting up a Teredo connection depends on the nature of the NAT device that sits in

front of the Teredo client. An example packet exchange that Teredo uses when the client is behind a Restricted NAT is shown below.



Teredo represent a different set of design trade-offs as compared to 6to4. In its desire to be useful in an environment that includes NATs in the IPv4 path Teredo is a per-host connectivity approach, as compared to 6to4's approach which can support both individual hosts and entire end sites within the same technology. Also, Teredo is a host-centric multi-party rendezvous application, and Teredo clients require the existence of dual stack Teredo servers and relays that exist in both the public IPv4 and IPv6 networks. Teredo is more of a connectivity tool than a service solution, and one that is prone to many forms of operational failure.

On the other hand, if you are an isolated IPv6 host behind an IPv4 NAT, and you want to access the IPv6 network then 6to4 is not an option, and you either have to set up static tunnels across the NAT to make it all work, or you can turn on Teredo in your dual stack host and you if everything goes according to theory, you should be able to establish IPv6 connectivity.

Tunnel brokers

In contrast to these auto-tunnel approaches, the simplest form of tunnelling IPv6 packets over an IPv4 network is the manually configured IPv6-in-IPv4 tunnel.

Here an IPv6 packet is simply prefixed by a 20 octet IPv4 packet header. In the outer IPv4 packet header the source address is the IPv4 address of the tunnel ingress, the destination address is the IPv4 address of the tunnel egress and IP protocol field uses value 41, indicating that the payload is an IPv6 packet. The packet is passed across the IPv4 network from tunnel ingress to egress using conventional IPv4 packet forwarding, and at the egress point the IPv4 IP packet header is removed and the inner IPv6 packet is routed in an IPv6 network as before. From the IPv6 perspective the transit across the IPv4 network is a single logical hop.

Alternatively, like VPN tunnels, the tunnel can be configured using UDP or TCP, and with some care, the tunnel can be configured through NATs in the same way as VPN tunnels can be configured through NATs.

The advantage of this approach is that the need to manually configure the tunnel endpoints ensures that the tunnel relay function is not provided, intentionally or unintentionally by third parties through some well-intentioned, but ultimately random, act of goodwill. The need to perform a manual configuration also reduces the chances that the tunnel is broken through local firewall filters. Of course the need to perform a manual configuration does not lend itself to a plug-and-play environment, nor is this a viable approach for a larger mass market of consumer devices and services.

Summary

None of these approaches to offer IPv6 connectivity to end hosts behind an IPv4-only service provider offer the same level of robustness and performance as compared to native IPv4 services. All of these approaches require a significant degree of local expertise to set up and maintain, and often require a solid understanding of other aspects of the local environment, such as firewall and filter conditions and path MTU behaviour to maintain. With the exception of the tunnel broker approach, they also require third party assistance to support the connection, which further adds to the set of potential performance and reliability issues.

It appears that the most robust and reliable way to provision IPv6 to end hosts is for the service provider to provision IPv6 as an integral part of their service offering, and offer clients a dual stack service in both IPv4 and IPv6.

In the second part of this article we'll look at the options for service providers to provision dual stack services to their clients.

Disclaimer

The above views do not necessarily represent the views or positions of the Asia Pacific Network Information Centre.

Author

Geoff Huston B.Sc., M.Sc., is the Chief Scientist at APNIC, the Regional Internet Registry serving the Asia Pacific region. He has been closely involved with the development of the Internet for many years, particularly within Australia, where he was responsible for the initial build of the Internet within the Australian academic and research sector. He is author of a number of Internet-related books, and was a member of the Internet Architecture Board from 1999 until 2005, and served on the Board of Trustees of the Internet Society from 1992 until 2001.

www.potaroo.net