The Internet Protocol Journal

March 2011 Volume 14, Number 1

A Quarterly Technical Publication for Internet and Intranet Professionals

In This Issue

You can download IPJ back issues and find subscription information at:
www.cisco.com/ipj

ISSN 1944-1134

From The Editors

In 2011 we have already seen some important Internet anniversaries and milestones. We have celebrated 25 years of IETF meetings and 40 years of the FTP protocol, but the most significant milestone took place in February when IANA handed out its final blocks of IPv4 addresses to the RIRs (see page 21). It seems like a good time to publish an edition of IPJ devoted entirely to IPv4/IPv6 transition, and to help me with this task I have invited Geoff Huston as co-editor and author for this issue, so let me hand it over to him:

There is a Chinese proverb that states: 寧為太平犬，不做亂世人 "It's better to be a dog in a peaceful time than be a man in a chaotic period." For the Internet, this year is shaping up to be a time that looks more like developing chaos than serenity and peace. The IANA has given out the last /8's, and demand has already depleted the IPv4 address stocks in the Asia Pacific. Meanwhile, the industry has discovered the mass marketing potential of mobile devices, and expects to sell and connect more than 250 million of them in 2011 alone.

The IETF designed IPv6 in the 1990s for this very reason. Its 128-bit address field is easily capable of accommodating the output of a prolific silicon manufacturing industry for many decades to come. But when we look at today's Internet, very little IPv6 can be seen. Estimates of the number of clients with functional IPv6 services hover at around 0.2 to 0.4 percent of the total.

The story about IPv6 transition technologies is complex, and there are many ways to undertake this effort. In this issue we will examine the various approaches and their relative strengths and weaknesses.

In order to send out a broad message about the need to shift online content from exclusively using IPv4 into a dual-stack world of both IPv4 and IPv6, ISOC is supporting *World IPv6 Day* on June 8. Phil Roberts explains this initiative and its role in helping the overall transition effort.

This transition is going to be difficult. It involves all parts of this diverse industry, and means combining some well-understood and widely-deployed technologies in some surprising and challenging ways. There is much to do, and we hope that this issue of IPJ provides an insight into just what the transition to IPv6 will entail.

—*Geoff Huston*, gih@apnic.net
Chief Scientist, APNIC

—*Ole J. Jacobsen*, ole@cisco.com
Editor and Publisher, IPJ

A Rough Guide to Address Exhaustion

by Geoff Huston, APNIC

The level of interest in IPv4 address exhaustion seems to be increasing, so I thought I would share some answers to the most common questions I have been asked on this topic in recent times.

What is the most significant challenge to the Internet today?

What a wonderfully open-ended question! There are so many challenges that I could identify: improving the level of security on the network, eradicating spam and viruses, improving capacity of the network infrastructure, improving the efficiency of high-speed data transfer, improving the accuracy of search engines, building more efficient and high-capacity data centers, and reducing the unit cost of Internet services, to name but a few.

If there is a common factor in many of these challenges, it is *scaling* the network to meet an ever-expanding agenda of more users, more devices, more traffic, more services, and more policies. And with more users and more forms of use come higher levels of diversity of use and greater need to replace implicit mechanisms of trust with explicit forms of trust negotiation and greater levels of demonstrable integrity of operation.

But these topics are all tactical in nature. They reflect the "how" of making the network work tomorrow by studying how to undertake marginal improvements on the network of today. However, it is not clear that the networks not just of tomorrow or next year, but a decade or more hence should reflect the usage patterns and user population of today. Perhaps a more fundamental challenge is to understand what is missing in today's network that we will need in the future.

This discussion leads to a pretty obvious challenge, at least for me. The basic currency of any network is *identifiers*. Identifiers allow the network to distinguish between clients and ensure that conversations occur between those parties who intended to communicate. In the world of packet-switched networking, such as IP, these endpoint identifiers are synonymous with the concept of an *address*. What is missing in today's network is an abundant supply of new addresses that will allow the network to scale up in size by a further factor of at least 1 million, and hopefully more than a billion-fold.

In fact, the supply of addresses is not just inadequate for future needs for a decade hence. The stock of addresses is facing imminent depletion, and the question of availability of addresses is best phrased in terms of months rather than years.

Perhaps the term "address" is somewhat of a misnomer in this context, but it may well be too late to change that now. The primary role of an IP address is not to uniquely identify the location of an endpoint of a network in relation to some positional or topographical coordinate set, but to simply uniquely identify an endpoint to distinguish it from all other endpoints. Its location is not an intrinsic property of this so-called *address*. But common convention is to call these endpoint identifiers "addresses," so I will stick with the same convention here.

So my candidate "most significant challenge for the Internet today" is that we are running out of further supply of IP addresses.

What is an IP address, and why is it so important?

One of the revolutionary changes introduced by the so-called *packet-switched* network architecture of the Internet—as compared to its telephone predecessor that used *circuit switching*—was that a massive amount of "intelligence" was ripped out of the network and placed into the devices that connect at the edge.

IP networks are incredibly simple, and at their most basic level they do very little. They are built of routers and interconnecting conduits. The function of a router is quite simple. As a packet arrives at the router from the connected circuitry (or from a wireless interface), it is divided into a common IP header and a payload. The IP header of the packet contains, among other components, two fixed-length fields: the address of the intended *destination* of the packet, and, like a postal envelope, the address of the packet creator, or the *source*. The router uses the destination address of the packet to make a routing decision as to how to dispose of the packet. For each incoming packet, the router inspects the destination address in the packet and either passes it to a connected computer if there is an address match or otherwise passes it down the *default* path to the next router. And that is a working description of the entirety of today's Internet. The important aspect here is that every connected device must have a unique address. As long as this condition is satisfied, everything else can be made to work.

In the current version of the Internet Protocol, an "address" is a 32-bit field, which can encompass some 4.4 billion unique values.

Why are we running out of addresses?

Blame silicon. Over the past 50 years, the silicon chip industry has graduated from the humble transistor of the 1950s to an astonishing industry in its own right, and the key to this silicon industry is volume.

Individual processor chips may take hundreds of millions of dollars to design, but if fabricated in sufficient volume, each processor chip may take as little as a few dollars to manufacture and distribute. The larger the production run of the silicon die, the lower the unit price of the resultant chip. We currently produce a huge volume of computers every year. In 2008 alone around 10 billion computer processors were manufactured. Although most of these microprocessors are simple 8-bit processors that are used to open doors or run elevators, a sizable proportion are used in devices that support communications, whether it is in laptop computers, smartphones, or even more basic communication applications. Typically we do not invent a new communications protocol for each new application. We recycle. And these days if we want a communications protocol for a particular application, it is easiest to simply embed the IP protocol engine onto the chip. The protocol is cheap, well tested, and it works across almost any scale we can imagine from a couple of bits per second to a couple of billion bits per second.

So it is not just the entire human population of the planet who may well have a desire to access the Internet in the future, but equally important is the emerging world of "things" that communicate. Whether it is the latest fashion in mobile phones or more mundane consumer electronics devices such as televisions or games consoles, all these devices want to communicate, and to communicate they need to have a unique identification code to present to the network, or, an "address."

We are presently turning on more than 200 million new Internet services every year, and today we have used up most of the 4.4 billion addresses that are encompassed by the IP protocol.

When will we run out?

As of September 2010, some 151 million addresses were left in the general-use pool of unallocated addresses that are managed by the central pool administration, the *Internet Assigned Numbers Authority* (IANA). The world's IP address consumption rate peaked earlier this year at a new all-time high of an equivalent rate of 243 million addresses per year.

By early February 2011 IANA handed out its last address blocks to the RIRs.

The five *Regional Internet Registries* (RIRs)[1] still had pools of addresses available for general use at that time, but from that point, as they further run down their local pools, the IANA is now unable to provide any more addresses to replenish them. The Asia Pacific Regional Registry, APNIC, has been experiencing the highest level of demand in the world, accounting for some two-thirds of all addresses consumed in early 2011. APNIC exhausted its general use IPv4 address pool in April 2011.

Although the current models of address consumption show that the other regions will be able to manage available address pools for a few more months, this prediction does not account for the multi-national nature of many of the largest of the service providers, and at this stage it is not known how much address-consumption pressure will shift outward from APNIC to the other RIRs now that APNIC's available address pool is effectively drained. So it may well be that 2011 will see IPv4 addresses cease to be generally available in many parts of the world, and by early 2012 there will be no further generally available IPv4 addresses in Europe, North America and Asia.

What is the plan?

This news of imminent exhaustion of the supply of addresses is not a surprise. Although the exact date of predicted address exhaustion has varied over time, the prospect of address exhaustion was first raised in technical circles in August 1990, and work has been undertaken since that time to understand what might be possible and how that could be achieved.

The 1990s saw an intense burst of engineering activity that was intended to provide a solution for this forthcoming address problem. The most significant outcome of this effort was the specification of a successor IP protocol to that of IPv4, called IP Version 6 or *IPv6*.

Why IPv6 and not IPv5?

It would be reasonable to expect the successor protocol of IP Version 4 to be called IP Version 5, but as it turned out Version 5 of the Internet Protocol Family was already taken. In the late 1980s the Internet Protocol itself was the topic of a considerable level of research, as researchers experimented with different forms of network behavior. Version 5 of the Internet Protocol was reserved for use with an experimental IP protocol, the *Internet Stream Protocol, Version 2* (ST-II), written up as RFC 1190 in 1990. When it came time to assign a protocol number of the "next generation" of IPv4, the next available version number was 6, hence IPv6.

The outcome of this process was a relatively conservative change to the IP protocol. The major shift was to enlarge the address fields from 32 bits to 128 bits in length. Other changes were made that were thought to be minor improvements at the time, although hindsight has managed to raise some doubts about that!

The design intent of IPv6 is a usable lifetime of more than 50 years, as compared with a "mainstream" deployment lifetime of IPv4 of 15 years, assuming that you are prepared to draw a line at around 1995 and claim that at that time the protocol moved from an interesting academic and research project to a mainstream pillar of the global communications industry.

That 50 years of usable life for IPv6 is admittedly very ambitious, because it is intended to encompass a growth of the ubiquity of silicon from the current industry volumes of hundreds of millions of new connected devices every year to a future level of activity that may encompass in the order of hundreds of billions to possibly some trillions of new connected devices every year.

So the technical plan to address the address-exhaustion problem was to perform an upgrade of the Internet and convert the Internet from IP Version 4 to IP Version 6.

Nothing else needs to be changed. This change is not intended to be radical or revolutionary. The change from circuit switching to packet switching was a revolutionary change for both the communications industry itself and for you and me as enthusiastic communicators. The change from IPv4 to IPv6 is intended to be a polar opposite, and at best it is intended to be a transparent and largely invisible transition. E-mail will still be e-mail. The web should still look just as it always did, and anything that works on IPv4 is expected to work on IPv6. IPv6 is not inherently any faster, nor any cheaper, nor is it even all that much better. The major change in IPv6 is that it supports a much larger address field.

How many addresses are in IPv6?

In theory, there are 2 to the power 128 unique addresses in IPv6—a very large number. If each IPv6 address were a single grain of sand, the entire IPv6 address space would construct 300 million planets, each the size of the earth!

But theory and practice align only in theory. In practice the IPv6 address plan creates a usable span of addresses that encompasses between 2 to the power 50 and 2 to the power 60 devices. Although this number is nowhere near 2 to the power 128, it is still a range of numbers that are between 1 million to 1 billion times the size of the IPv4 address space.

How do we transition to IPv6?

Unfortunately IPv6 is not "backward-compatible" with IPv4. Backward compatibility would allow for a piecemeal transition, where IPv6 could be regarded as a fully functional substitute for IPv4, so that the existing network base would keep using IPv4 forever, while the most recent devices would use IPv6 and all devices could communicate with each other. The lack of such backward compatibility implies that this communication is simply not possible. IPv4 and IPv6 are distinct and different communications protocols, in the same way that English and, say, German are distinct and different languages.

Attempts have been made to design various forms of automated protocol translator units that can take an incoming IPv4 packet and emit a corresponding IPv6 packet in the same manner as a language interpreter. However, this approach also has some major limitations, so it is usable only in very carefully constrained contexts.

The implication of this lack of backward compatibility and inability to perform automated translation within the network is that if we want to preserve comprehensive any-to-any connectivity during the transition, we have to equip each device that is performing a transition with both protocol stacks, or, in effect, allow the device to become "bilingual," and conduct a conversation in either IPv4 or IPv6, as required. This transition has been termed a *dual-stack* transition.

When my computer supports IPv6, can I return my IPv4 address?

Each device needs to maintain its capability to converse using IPv4 while there are still other devices out there that remain IPv4-only. So a device that becomes IPv6-capable cannot immediately give up its IPv4 address. It will need to keep this IPv4 capability and operate in dual-stack mode for as long as there are other devices and services out there that are reachable only using IPv4.

The implication of this constraint is that we will need to add dual-stack devices to the Internet and consume both IPv4 and IPv6 addresses during this transition.

So, no, you will need to keep your IPv4 address for as long as there are folk out there with whom you want to communicate who have not also migrated to be a dual-stack IPv4- and IPv6-capable entity.

What needs to be done to transition the network to IPv6?

What is encompassed in "transition?" Do all *Internet Service Providers* (ISPs) have to decide when and how to reprogram their systems and reconfigure their routers, switches, and middleware? Will they need to replace all their customers' modems with ones that support IPv6? What is the agenda?

This level of uncertainty about the transition to IPv6 is evidently widespread in today's Internet. Most of the actors in the Internet are unsure about what needs to be done, from the largest of the service providers down to individual end users. Yes, it appears to be a simple matter of reprogramming devices from being just IPv4-capable to being capable of supporting both IPv4 and IPv6, but it is not quite so simple. Dual-stack operation is not easy, nor will it just happen without any form of applied impetus. Imagine that this transition is from everyone on the planet speaking Latin to each other to everyone speaking Esperanto. If this situation were a simple matter of everyone stopping using one language and being rebooted to use the other language one by one, then imagine the plight of the first people to undertake this transition—from being connected and being able to communicate with everyone else using Latin, these first users would find themselves speaking exclusively Esperanto to ... nobody! They would in effect have been disconnected from the network.

So the transition is a little trickier than just turning a big switch from IPv4 to IPv6. Because this transition is a piecemeal and fragmentary one, each device, each router, each firewall, each load server, and all those other components of the network service platform need to be programmed with an additional protocol, and become, in effect, bilingual. And in this case there are no magic interpreters that can "translate" between IPv4 and IPv6. So it is only when the entire network is bilingual in a dual-stack mode that we can turn off IPv4 and consider the transition to be complete.

For an extended period of time the Internet is going to have to operate as two Internets. We have never tried that type of operation before, at least not on a grand scale as this one; in fact, it has often been likened to replacing the jet engines of an airplane while the plane is in flight. Somehow we now have to not only sustain a growth rate of at least some 250 million new connections per year, but at the same time retrofit IPv6 to the existing installed base while continuing to support IPv4. The complexity of this operation is significant, and there is considerable confusion about what to do, when to do it, how much it will all cost, and who will pay. So yes, we are all unsure about what needs to be done.

How long do we expect this dual-stack transition to take?

If only we knew! The Internet today encompasses some 1.7 billion users, and hundreds of millions of devices out there are configured to "talk" only IPv4. Some of these devices will surely die in the coming years, and others may be upgraded or reprogrammed, but others will persist in operation for many years to come while continuing to speak only IPv4. Even looking at what is being sold today, although many general-purpose computers (or at least their operating systems) are now configured to operate in dual-stack mode, when you look at embedded devices such as *Digital Subscriber Line* (DSL) or cable modems, or firewalls, or a myriad of other devices that are integral to the operation of today's Internet, many of these devices are still configured in firmware to operate exclusively using IPv4.

Some modeling of the transition process has projected an 80-year transition process. That projection is heading into the realms of the absurd, given that our expectations for the operational lifespan of IPv6 have a lower bound of just 50 years or so. However, given the sheer scope of the conversion task and the current level of penetration of IPv6 to levels of between 2 and 5 percent of today's Internet, and given that a deadline of 2 years from now implies a conversion rate of in excess of 1 million devices every day in that 2-year span. It seems that an expectation that this transition could be substantially completed in as little as 2 years also strikes an unrealistic note.

So a more realistic assumption is that we will probably take around 5 years to complete this transition, and we will need to operate the Internet in dual-stack mode with both IPv4 and IPv6 across this entire period.

But at the current level of Internet growth, the IPv4 address pool cannot sustain a further 5 years of growth—at least not with the current amount of unallocated addresses remaining in the allocation pools. The current address-consumption rate is some 250 million addresses per year. The depleted IPv4 address pool simply cannot withstand the pressures of a 5-year transition without a radical change to the model of the IPv4 network. And if we need to rework the model of the IPv4 network simply to sustain a transition to IPv6, then can't we simply get going with IPv6 a little more quickly instead?

However, "fully depleted" or even "run out" is perhaps not the most appropriate way to describe what will happen to IPv4 addresses in the coming months. It is probably more accurate to say "unobtainable at the current prices." When the current orderly process of allocation of IPv4 addresses comes to an end, that does not mean that IPv4 addresses will be completely unobtainable. In this world many things that are scarce are still obtainable—for a price. It is quite reasonable to anticipate that for as long as there is still a demand for IPv4 addresses there will be some form of "aftermarket" where addresses are traded for money. However, as with many markets, what is not possible to predict is the price for addresses that will be established by such a market-based address-trading regime.

What about "address sharing" in IPv4? Why do we need IPv6, given that we could simply share addresses in IPv4?

Yes, of course address sharing[2] is an option, and we have been doing it for many years already in IPv4. But is it a viable substitute for IPv6?

As part of the engineering effort to develop a successor protocol to IPv4 in the mid 1990s, the IETF published a novel approach of *address sharing,* which we call today *Network Address Translation,* or NAT.[3] These days almost every DSL modem, and other forms of customer connection equipment, comes equipped with NAT functions. Today most Internet Service Providers give their subscribers a single IPv4 address. At home I have a single IPv4 address, and you probably do too. But in my home I have about 20 connected devices of various sorts (I am counting TiVo units, game consoles, televisions, printers, and such, because they are all in essence Internet-connected devices, and I believe that my situation is not unusual). All these devices "share" the single external IP connection, so all of them "share" this single IPv4 address.

But address sharing has its limitations. When a single household shares a single address, nothing unusual happens. But If I were to try to do the same address-sharing trick of using a single IP address to share across, say 2,000 customers, I would cross over into a world of pain. Many applications today gain speed through parallelism, and they support parallelism through consuming port addresses.

Each IP address can support the parallel operation of 65,535 sessions, using a 16-bit *port identifier* as the distinguishing identifier. But when address sharing is used, these ports are shared across the number of devices that are all sharing this common address. When 2,000 customers are sharing a single address and each customer has some 20 or so devices, then the average number of port addresses per device is 1.5. Common applications that exploit parallel operation include such favorites as *Gmail, Google Maps,* and *iTunes*. With a sufficiently constrained number of available ports to use, these applications would cease to work. Indeed, many network applications would fail, and at a level of a single address shared across 2,000 households, I would guess that up to half of these 2,000 customers would not have a working Internet at any single point in time.

Our experience suggests that address sharing works only up to a point, and then it breaks everything badly. We are already address sharing at the level of sharing a single address per household, and households are these days buying more connected devices of various sorts, not *fewer*. So attempting to share that single address across more than one household is at best a temporary solution, and is not a sustainable option that is an alternative to IPv6.

So we need to transition to IPv6, and we need to do so within an impossibly short time.

This discussion all sounds like a terrible problem. Was this global "experiment" with the Internet all one big mistake? Should we have looked elsewhere for a networking technology back in the 1990s?

The IP address problem is—for me at any rate—a fascinating one. At the time when researchers were working on the specifications for the Internet Protocol in the 1970s, they decided to use fixed-length 32-bit fields of the interface identifier addresses in the protocol. This decision was a radical one at the time. Contemporary network protocols, such as *DECnet Phase III,* used 16-bit address lengths, and 8-bit addresses were also very common at the time. After all, computers were so big and expensive, who could possibly afford more than 256 unique devices in a single network? Eight bits for addresses was surely enough! Using 32 bits in the address field was not an easy decision to make, because there was constant pressure to reduce the packet headers in order to leave more room for the data payload, so to reserve such a massive amount of space in the address fields of the protocol header to allow two 32-bit address fields was a very bold decision.

However, it was a decision that has proved to be very robust. TCP/IP has sustained the Internet from a mere handful of warehouse-sized computers running at mere kilobits per second to today, where probably more than 3 billion devices connect to the Internet in one way or another, at speeds that range from a few hundred bits per second to a massive 100 Gbps—all talking one single protocol that was invented more than 30 years ago.

IP has demonstrated a scale factor 1 billion! In my mind that achievement demonstrates a level of engineering foresight that is truly phenomenal. So in some sense the underlying observation here is not that IPv4 is running out of addresses today, but that it has been able to get to today at all!

Given that IPv4 has been able to scale by a factor of 1 billion, then if we can make IPv6 scale by a further factor of 1 billion from today we will have done well.

Disclaimer

The views expressed in this article do not necessarily represent the views or positions of the *Asia Pacific Network Information Centre* (APNIC).

References

[1] Daniel Karrenberg, Gerard Ross, Paul Wilson, and Leslie Nobile, "Development of the Regional Internet Registry System," *The Internet Protocol Journal,* Volume 4, No. 4, December 2001.

[2] Geoff Huston, "NAT++: Address Sharing in IPv4," *The Internet Protocol Journal,* Volume 13, No. 2, June 2010.

[3] Geoff Huston, "Anatomy: A Look inside Network Address Translators," *The Internet Protocol Journal,* Volume 7, No. 3, September 2004.

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 numerous Internet-related books, and was a member of the Internet Architecture Board from 1999 until 2005; he served on the Board of Trustees of the Internet Society from 1992 until 2001. E-mail: **gih@apnic.net**

World IPv6 Day

by Phil Roberts, ISOC

On June 8, 2011, websites including *Google, Facebook, Yahoo!,* and *Bing* will make their main webpages reachable over IPv6 for a 24-hour period from 00:00 to 23:59 *Coordinated Universal Time* (UTC). This activity, *World IPv6 Day,* a "test flight" of IPv6, is motivating organizations across the Internet industry to prepare their services for IPv6, the next generation of the Internet Protocol. Internet Service Providers, hardware makers, operation system and application vendors, and other websites are indeed working to make this activity of testing IPv6 on an Internet scale successful.

The Internet is a never-ending exercise in collaboration. Making a successful transition to IPv6 is one of the major challenges facing the Internet today. Although IPv6 is used extensively in many large networks today, the World IPv6 Day activity is acting as a focal point to bring together all parts of the Internet industry to accelerate deployment of IPv6 in all parts of the Internet.

For some time the deployment of IPv6 has faced a "chicken-and-egg problem." Website owners have been reluctant to deploy IPv6 because there were not many end users to view their webpages over IPv6. Network operators have been hesitant to deploy IPv6 for many end users because there were few places for those users to view content over IPv6. That the most popular websites in the world according to Alexa rankings are deploying IPv6 on their main webpages is a clear indication that the Internet industry is moving beyond this long-standing impasse. Although June 8 is a 24-hour test, it is clear that this is a move toward regular operation of IPv6, and network operators can confidently roll out IPv6 to end users knowing that the Internet industry is making a concerted effort to make IPv6 an operational reality.

Today, IPv6 connectivity concerns provide another disincentive for a major website to enable IPv6 for regular operation. Badly configured or poorly behaving implementations may prevent end users from reaching a major website that enables IPv6 on its main page. It is currently estimated that this problem will affect only a minor percentage of end users—at the time of the announcement of World IPv6 Day, the estimate was that only 0.05 percent of end users would experience difficulties.

Although this percentage is small, it is potentially a very large number of end users for a website that has visitors numbering in the tens of millions (or more). It is simply impossible from a business point of view for a website of this magnitude to deploy IPv6 alone when this many users could be affected. The users who would not be able to get to that website will simply go to another website in search of similar services.

However, because several such websites have agreed to do this testing at the same time, and for the same duration, individual end users who experience disruption of their connectivity by IPv6 may be able to determine that the problem they are experiencing is indeed not a problem with a set of major websites but may, in fact, be a problem in their own host or network, and will provide an incentive for them to take steps to determine the source of the problem and repair it.

Website owners, network operators, and hardware and software vendors are collaborating to minimize these effects leading up to World IPv6 Day. All of these organizations are working to provide tools to detect these problems and offer suggested fixes in advance of June 8. The test site **http://test-ipv6.com/** allows end users today to test their connectivity and determine whether their connectivity to websites will be affected when those websites enable IPv6.

Some websites have already performed a similar 24-hour test. Last year, the German online news site Heise (**http://www.heise.de**) conducted a similar experiment. The site enabled IPv6 on its main page for 24 hours, turned it off, examined the effects of the experiment, and then permanently enabled IPv6 on its main page. Two major websites in Norway did a similar test, and they also have enabled IPv6 permanently. An activity like this for many websites is clearly a step toward regular and normal IPv6 operations. Website owners will, of course, determine when it makes sense for their business to make IPv6 operations available permanently.

Since the announcement of World IPv6 Day, many other websites from around the world have indicated that they are deploying IPv6, and many of those have decided to join in the global IPv6 test on June 8. The list of websites includes major websites such as *Google, Facebook,* and *Yahoo!* and very small websites with small numbers of visitors. It is exciting that websites from every inhabited continent plan to participate. Major websites from the Czech Republic, Portugal, Brazil, and Japan, for example, are joining this test, with more websites joining every day.

For further information about World IPv6 Day, please visit:
http://www.isoc.org/wp/worldipv6day

There you will find details about the websites that will be turning on IPv6 on June 8, how to join, and information for networks and individuals, including an FAQ.

PHIL ROBERTS joined the Internet Society (ISOC) in 2008. Prior to that he spent several years with Motorola in research and product development, all in the area of mobile broadband systems. He has been active in the IETF for more than a decade. He can be reached at: **roberts@isoc.org**

Transitional Myths

by Geoff Huston, APNIC

Last October, I attended the *Réseaux IP Européens* (RIPE)[1] meeting in Rome, and—not unexpectedly for a group that has some interest in IP addresses—the topic of IPv4 address exhaustion, and the related topic of the transition of the network to IPv6, captured a lot of attention throughout the meeting. One session I found particularly interesting was on the transition to IPv6, where people related their experiences and perspectives on the forthcoming transition to IPv6.

I found the session interesting, because it exposed some commonly held beliefs about the transition to IPv6, so I will share them here, and discuss a little about why I find them somewhat fanciful.

Myth 1: "We have many years for this transition."

No, I don't think we do!

The Internet is currently growing at a rate that consumes some 200 million IPv4 addresses every year, or 5 percent of the entire address IPv4 pool. This growth rate reflects an underlying growth of service deployment by the same order of magnitude of some hundreds of millions of new services activated per year. Throughout a dual-stack transition, all existing services will continue to require IPv4 addresses, and all new services will also require access to IPv4 addresses. The pool of unallocated addresses was exhausted in February 2011, and the *Regional Internet Registries* (RIRs)[2] will exhaust their local pools commencing early 2011 and through 2012. When those pools exhaust, then all new Internet services will need access to IPv4 addresses as part of the IPv4 part of the dual-stack environment, but at that point there will be no more freely available addresses from the registries. Service providers have some local stocks of IPv4 addresses, but even those stocks will not last for long.

As the network continues to grow, the pressure to find the equivalent of a further 200 million or more IPv4 addresses each year will become acute—and at some point will be unsustainable. Even with the widespread use of *Network Address Translators* (NATs)[3] and further incentives to recover all unused public address space, the inexorable pressure of growth will cause unsustainable pressures on the supply of addresses.

It is unlikely that we can sustain 10 more years of network growth using dual stack, so transition will need to happen faster than that. How about 5 years? Even then, at the higher level of growth forecasts, we will still need to flush out the equivalent of 1.5 billion IPv4 addresses from the existing user base to sustain a 5-year transition, and this number seems to be a stretch target. A more realistic estimate of transition time, in terms of accessible IPv4 addresses from recovery operations, is in the 3–4 year timeframe, and no longer.

So no, we do not have many years for this transition. If we are careful—and a bit lucky—we will have about 4 years.

Myth 2: "It is just a change of a protocol code. Users will not see any difference in the transition."

If only that were true!

In an open market environment, scarcity is invariably reflected in price. For as long as this transition lasts, this industry is going to have to equip new networks and new services with IPv4 addresses, and the greater the scarcity pressure on IPv4 addresses, the greater the scarcity price of IPv4 addresses. Such a price escalation of an essential good is never a desirable outcome, and although numerous possible measures can be taken to mitigate the problem, to some extent or other, the scarcity pressure and the attendant price escalation suggest a reasonable expectation of some level of price pressure on IPv4 addresses.

In addition, an *Internet Service Provider* (ISP) may not be able to rely solely on customer-owned and-operated NATs to locally mask out some of the incremental costs of IPv4 address scarcity. It is likely—and increasingly so the longer the transition takes—that the ISP will also have to operate NATs. The attendant capital and operational costs of such additional network functions will ultimately be borne by the service provider's customer base during the transition.

But it is not just price that is affected by this transition—network performance may also be affected. Today a connection across the Internet is typically made by using the *Domain Name System* (DNS) to translate a name to an equivalent IP address, and then launching a connection-establishment packet (or the entire query in the case of the U*ser Datagram Protocol* [UDP]) to the address in question. But such an operation assumes a uniform single protocol. In a transition world you can no longer simply assume that everything is contactable with a single protocol, and it is necessary to extend the DNS query to two queries, one for IPv4 and one for IPv6. The client then needs to select which protocol to use if the DNS returns addresses in both protocols. Then there is the tricky problem of failover. If the initial packet fails to elicit a response within some parameter of retries and timeouts, then the client will attempt to connect using the other protocol with the same set of retries and timeouts. In a dual-stack transitional world, not only does failure take more time to recognize, but even partial failure may take time.

So users may see some changes in the Internet. They may be exposed to higher prices that reflect the higher costs of operating the service, and they may see some instances where the network simply starts to appear "sluggish" in response.

Myth 3: "NAT upon NAT upon NAT will work."

Maybe. But maybe not all the time, and maybe not in ways that match what happens today.

The Internet has been operating for more than a decade now with a very prevalent model of a single level of address translation in the path. Application designers now assume its existence, and also make some other rather critical assumptions, notably that the NAT is close to the client in a client-server world, and that there is a single NAT in the path, and that its particular form of address translation behavior can be determined with numerous probe tests. There is even a client-to-NAT protocol to assist certain applications to communicate port-binding preferences to the local NAT. In a multilevel NAT world, such assumptions do not directly translate, but it is not necessarily the case that the application is aware of the added NATs in the end-to-end path.

However, it is not just the added complexity of the multipart NAT that presents challenges to applications. The NAT layering is intended to create an environment where a single IP address is dynamically shared across multiple clients, rather than being assigned to a single client at a time. Applications that use parallelism extensively by undertaking concurrent sessions require access to a large pool of available port addresses. Modern web browsers are a classic example of this form of behavior. The multiple NAT model effectively shares a single address across multiple clients by using the port address, effectively placing the pool of port addresses under contention. The higher the density of port contention, the greater the risk that this multiple layering of NATs will have a visible effect on the operation of the application.

There is also a considerable investment in the area of logging and accountability, where individual users of the network are recorded in the various log functions through their public-side address. Sharing these public addresses across multiple clients at the same time—as is the intended outcome of a multilayer NAT environment—implies that the log function is now forced to record operations at the level of port usage and individual transactions. Not only does this reality have implications in terms of the load and volume of logged information, there is also a tangible increase in the level of potential back tracing of individual users' online activities if full port usage logging were to be instituted, with the attendant concerns that this back tracing represents an inappropriate balance between accountability and traceability and personal privacy. It is also unclear whether there will be opportunity to have any public debate on such a topic, given that the pressure to deploy multilevel NAT is already visible.

Myth 4: "Changing the Customer Premises Equipment (CPE) is easy."

No, not necessarily.

I think we have all seen many transition plans, including multilevel Version 4 NATs, NATs that perform protocol translation between IPv4 and IPv6, NATs plus tunneling, as in *Dual-Stack Lite,* the *IVI Bi-direction Mapping Gateway, 6to4, 6RD,* and *Teredo,* to call up but a few of the various transitional technologies that have been proposed in recent times. (See the article "Transitioning Protocols" starting on page 22.)

All approaches to dual-stack transition necessarily make changes to some part of the network fabric, whether it is changes to the end systems to include an IPv6 protocol stack in addition to an IPv4 stack, or the addition of more NATs, or gateways into the network infrastructure. Of course, within a particular transitional model there is a selective choice as to what elements of the infrastructure are susceptible to change and what elements are resistant to change. Some models of transition, such as 6RD and Dual-Stack Lite, assume that changing the CPE is easy and straightforward, or at least that such a broad set of upgrades to customer equipment is logistically and economically feasible. 6RD contains an implicit assumption that the network operator has no economic motivation to alter the network elements, and wishes to retain a single protocol infrastructure that uses IPv4.

Where the CPE is owned, operated, and remotely maintained by the service provider, upgrading the image on the CPE might present fewer obstacles than upgrading other elements of the network infrastructure, such as broadband remote-access servers that operate in a single protocol mode, but sweeping generalizations in this industry are unreliable. Service providers tend to operate customized cost models, and appear to be operating with specialized mixes of vendor equipment and operational support systems. For this reason operators tend to have differing perspectives on what component of their network is more malleable, and correspondingly have differing perspectives on which particular transition technology suits their particular environment.

This industry is volume-based, where an underlying homogeneity of the deployed technology—and economies of scale and precision of process—are critical components of reliable and cost-efficient rollouts. It is somewhat unexpected to see this transition expose a relative high degree of customization and diversity in network service environments.

Myth 5: "My ISP has enough IPv4 addresses to last for years, so it does not have a problem."

Well, not necessarily.

The assumption behind this statement is that everyone else is also able to persist with IPv4, and everyone you wish to reach, and every service point you wish to access, will maintain some form of connectivity in IPv4 indefinitely.

But this assumption is not necessarily valid. At the point in time when a significant number of clients or services cannot be adequately supported on IPv4, then irrespective of how many IPv4 addresses ISPs have, they will need to provide their clients with IPv6 in order to reach these IPv6-only services On a network, the actions of others directly affect your own local actions. So if you believe that you need do nothing, and you can use an IPv4 service for years into the future, then this position will be inadequate at the point in time when a significant number of others encounter critical levels of scarcity such that they are incapable of sustaining the IPv4 side of a dual-stack deployment, and are forced to deploy an IPv6-only service. The greater the level of address hoarding, the greater the level of pressure to deploy IPv6-only services on the part of those service providers who are badly placed in terms of access to IPv4 addresses.

Myth 6: "We will always have to run IPv4 protocols."

Probably not.

Or at least not in terms and volumes that are significant to the industry over the forthcoming decades. Protocols do die. DECnet and *Systems Network Architecture* (SNA) no longer exist as widely deployed networking protocols. In particular, networking in the public space is all about any-to-any connectivity, and to support this connectivity we need a common protocol foundation. In terms of the dynamics of transition, this situation is more about tipping points of the mass of the market than it is about sustained coexistence of diverse protocols. When a new technology—or in this case, protocol—achieves a critical level of adoption, the momentum switches from resisting the change to embracing it.

The aftermath of such transitions does not leave a legacy of enduring demand for the superseded technology. As difficult as it is to foresee today, when the industry acknowledges that the new technology achieves this critical mass of adoption, the dynamics of the networking effect propels the industry into a tipping point where the remainder of the transition is likely to be both inevitable and comprehensive. The likely outcome of this situation is that there is no residual significant level of demand for IPv4.

Myth 7: "There is a technology that will translate between IPv4 to IPv6."

Yes, but...

Such a technology effectively maps between IPv4 and IPv6 addresses. One approach, the *IVI Bi-direction Mapping Gateway,* provides a 1:1 mapping by embedding fields of one address in the other. Another approach, originally termed *Network Address Translator – Protocol Translator* (NAT-PT), uses a mapping table in a fashion similar to a conventional NAT unit. The common constraint here is that if there are no IPv4 addresses, then such a bidirectional mapping cannot be sustained in each approach. Ultimately, if every packet that traverses the public Internet requires public address values in the source and destination fields, and the ISP must provide a protocol bridge between IPv4 and IPv6, then public IPv4 addresses are required.

But it is not just the requirement for continued access to addresses that is the critical concern here. A reading of RFC 4966[4], "Reasons to Move the Network Address Translator – Protocol Translator (NAT-PT) to Historic Status" should curb any untoward enthusiasm that this approach is capable of sustaining the entire load of this dual-stack transition without any further implications or problems.

Myth 8: "We do not necessarily have to transition to IPv6. There are substitutes."

Nothing is visible from here!

If we want to continue to operate a network at the price, performance, and functional flexibility that is offered by packet-switched networks, then the search for alternatives to IPv6 is necessarily constrained to a set of technologies that offer approaches that are—at a suitably abstract level—isomorphic to IP. But going from abstract observations to a specific protocol design is never a fast or easy process, and the lessons from the genesis of both IPv4 and IPv6 point to a period of many years of design and progressive refinement to develop a viable approach. In our current context any such redesign is not a viable alternative to IPv6, given the timeframe of IPv4 address exhaustion. It is unlikely that such an effort would elicit a substitute to IPv6, and it is more likely that such an effort may lead toward an inevitable successor to IPv6, if we dare to contemplate networking technologies further into the future.

Other approaches exist, based on application-level gateways and similar forms of mapping of services from one network domain. We have been there before in the chaotic jumble of networks and services that defined much of the 1980s, and it is a past that I for one find easier to forget! Such an outcome is of considerably higher complexity, considerably less secure, harder to use, more expensive to operate, and more resistant to scaling.

Like it or not, the pragmatic observation of today's situation is that we do not have a viable choice here. No viable substitutes exist.

Myth 9: "We know what is happening."

I am not sure that is universally true! The comments I have heard about the current situation lead me to the observation that there are many different perspectives on the situation. Individuals perceive the transition in terms that relate to their own circumstances and their own limitations, and a more encompassing perspective of the entire Internet and this transition is harder to assemble. So, from the perspective of the Internet as a whole, no, we are not really aware of what is happening.

Myth 10: "We know what we are doing."

Individually this statement is, hopefully, true. But at the level of the entirety of the Internet, no, we do not really have a clear perspective of this transition.

Myth 11: "We have a plan!"

See the comment for myth 10.

Myth 12: "The Internet will be fine!"

I am unsure about this one.

The worrying observation is that the Internet has so far thrived on diversity and competition. We have seen constant innovation and evolution on the Internet, and the entrance of new services and new service providers.

But if we rely solely on IPv4 for the future Internet, then this level of competition and diversity will be extremely challenging to sustain. If we lose that impetus of competitive pressure from innovation and creativity, then the Internet will likely stagnate under the oppression of brutal volume economics. The risks of monopoly formation under such conditions are relatively high.

I hope one observation I heard at the RIPE session will be a myth as this transition gets underway:

> *"The incumbents will have all the IPv4 space.*
> *Thanks for playing!"*

If that is *not* a myth, then we are going to be in serious trouble!

Disclaimer

The views expressed in this article do not necessarily represent the views or positions of the *Asia Pacific Network Information Centre* (APNIC).

References

[1] `http://www.ripe.net/ripe/meetings/ripe-61/`

[2] Daniel Karrenberg, Gerard Ross, Paul Wilson, and Leslie Nobile, "Development of the Regional Internet Registry System," *The Internet Protocol Journal,* Volume 4, No. 4, December 2001.

[3] Geoff Huston, "Anatomy: A Look inside Network Address Translators," *The Internet Protocol Journal,* Volume 7, No. 3, September 2004.

[4] Cedric Aoun and Elwyn Davies, "Reasons to Move the Network Address Translator – Protocol Translator (NAT-PT) to Historic Status," RFC 4966, July 2007.

Pool of Unallocated IPv4 Addresses Now Completely Emptied

On February 3, 2011 a critical point in the history of the Internet was reached with the allocation of the last remaining IPv4 Internet addresses from a central pool. It means the future expansion of the Internet is now dependant on the successful global deployment of the next generation of Internet protocol, called IPv6.

The announcement was made by four international non-profit groups, which work collaboratively to coordinate the world's Internet addressing system and its technical standards. At a news conference in Miami, Florida, the *Internet Corporation for Assigned Names and Numbers* (ICANN) joined the *Number Resources Organization* (NRO), the *Internet Architecture Board* (IAB) and the *Internet Society* (ISOC) in announcing that the pool of first generation Internet addresses has now been completely emptied. The final allocation of Internet addresses was administered by the *Internet Assigned Numbers Authority* (IANA), which is a function of ICANN.

"This is a major turning point in the on-going development of the Internet," said Rod Beckstrom, ICANN's President and Chief Executive Officer. "No one was caught off guard by this. The Internet technical community has been planning for IPv4 depletion for some time. But it means the adoption of IPv6 is now of paramount importance, since it will allow the Internet to continue its amazing growth and foster the global innovation we've all come to expect."

Two "blocks" of the dwindling number of IPv4 addresses—about 33 million of them—were allocated in late January to APNIC, the *Regional Internet Registry* (RIR) for the Asia Pacific region. When that happened, it meant the pool of IPv4 addresses had been depleted to a point where a global policy was triggered to immediately allocate the remaining small pool of addresses *equally* among the five global RIRs.

"It's only a matter of time before the RIRs and *Internet Service Providers* (ISPs) must start denying requests for IPv4 address space," said Raúl Echeberría, Chairman of the NRO, the umbrella organization of the five RIRs. "Deploying IPv6 is now a requirement, not an option."

Transitioning Protocols

by Geoff Huston, APNIC

In the previous article, I looked at some common myths associated with the transition to IPv6. In this article I would like to look behind the various opinions and perspectives about this transition, and examine in a little more detail 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 have now achieved the first milestone of address exhaustion, the depletion of the central pool of *Internet Assigned Numbers Authority* (IANA)-managed address space. The last five /8s were handed out from IANA to the *Regional Internet Registries* (RIRs) on February 3, 2011. After some years of industrywide general inattention and inaction with IPv6, perhaps it is 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 is 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. Although 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 is not just *one* approach to adding support for IPv6 in your network, but *many*. And it is not just one major objective you need to address—incremental deployment of IPv6 as a second protocol into your operational network without causing undue disruption to existing services—but two, because 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 are effectively exhausted.

When?

The most common question I have 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[1] of IPv4 addresses.

Using a model based on monthly address demands now predicts that the next 18 months or so will see the first three RIRs depleted of IPv4 addresses.

The *Asia Pacific Network Information Centre* (APNIC) was the first RIR to exhaust its available pool of IPv4 addresses in April 2011, with the *RIPE Network Coordination Centre* (RIPE NCC) predicted to follow in late 2011 and the *American Registry for Internet Numbers* (ARIN) in early 2012. The *Latin American and Caribbean Internet Addresses Registry* (LACNIC) is predicted to follow in 2014, and the *African Network Information Centre* (AFRINIC) in 2016.

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

The not-so-good news is that there is no simple single path to follow. Each individual network needs to carefully consider the transition and select an approach that matches their particular circumstances. For an industry used to playing "follow the leader" for many years, a variety of choice is not always appreciated. And, unfortunately, we are spoiled for choices here.

Let's look at each of the major transitional technologies that are currently in vogue, and examine their respective strengths and weaknesses and their intended area of applicability. We will look at these technologies first from the perspective of the end user and then from the other side, examining options for *Internet Service Providers* (ISPs).

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. If you configure your modem or router with IPv6 in addition to IPv4, you are finished, assuming of course that your local modem or router unit actually supports IPv6—an assumption that may not be valid in many of the older and, unfortunately, 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 through an *IPv6 Router Advertisement*. Local hosts should be constructed to configure their IPv6 stack automatically, and your system should be connected as a dual-protocol system.

You probably do, however, need to be aware of some caveats, of which the most important is likely to relate to the probable absence of a *Network Address Translation* (NAT)[2] function in IPv6. Currently 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 autoconfigure themselves as NAT devices, permitting outgoing connections using the *Transmission Control Protocol* (TCP) or *User Datagram Protocol* (UDP), and allowing some *Internet Control Message Protocol* (ICMP) message types to traverse the NAT, but not much else. For many clients this NAT configuration becomes the default local security framework, because it permits outbound connections through TCP and UDP to be made, but not much else, and permits initiation of no sessions as incoming sessions. With IPv6 the local network is generally configured with an entire subnet, and instead of a NAT, this subnet is directly connected to the Internet.

The local network is then in a mixed situation of being behind a NAT in IPv4, but directly connected to the Internet using IPv6. This asymmetric configuration with respect to IPv4 and IPv6 raises some questions about the effect on the security of your local network. You need to think about adding appropriate filter rules to the gateway 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 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 rapidly depleting, it is really not very common to have an ISP offering dual-stack IPv4 and IPv6 services. Let's look at the more common situation, when your ISP is still offering only IPv4. As an end user, can you still set up some form of IPv6 access?

The answer is "Yes," but you must use 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 Tunneling Protocol*.

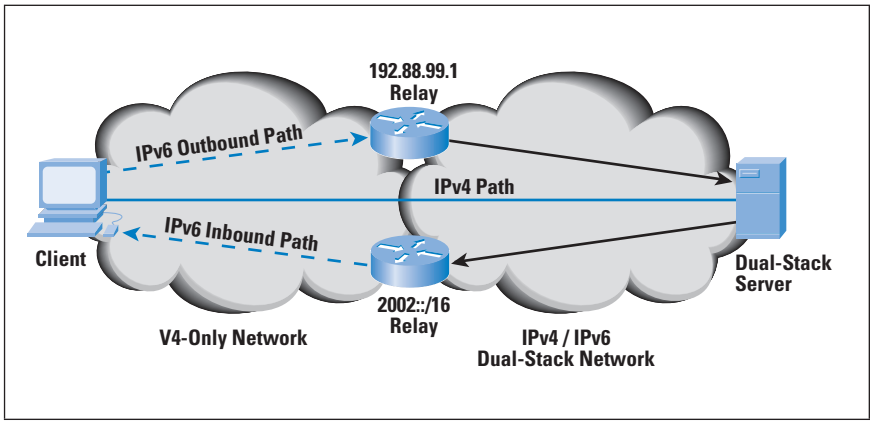
6to4 is an autotunneling protocol coupled with an addressing structure. The IPv6 address of a 6to4-reachable host begins with the IPv6 prefix `2002::/16`. The address architecture embeds a 32-bit IPv4 address of the end host into the next 32 bits. That way the IPv6 address carries the "equivalent" IPv4 address within the IPv6 address.

To send an IPv6 packet, the local host must first tunnel through the local IPv4 network. To perform this tunneling, 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 tunneling IPv6 packets (RFC 2473)[3].

The IPv4 packet is addressed to an IPv4-to-IPv6 relay. To avoid manual configuration of each client, all these relays 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 an 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 a relay receives a packet destined to a `2002::/16` address, it lifts the IPv4 address from inside the IPv6 address. It then wraps 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 (Figure 1).

Figure 1: 6to4 Tunneling Architecture



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 then the local gateway needs to be configured to accept incoming IP packets that use protocol 41.

An alternative is to configure the gateway device of the local network 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 with a conventional IPv6 Router Advertisement. The gateway can couple this advertisement with a NAT function and provide native IPv6 to interior hosts that are configured on RFC 1918[4] local IPv4 addresses.

In general, 6to4 is a relatively poor approach to provisioning IPv6, and you really should avoid it if at all possible. Indeed, your experience will probably be better overall if you continue running IPv4 and avoid accessing IPv6 with 6to4!

The major concern 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 usable 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 usable 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, not necessarily true 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). The 6to4 packets use protocol 41, and there appears to be widespread use of filter rules that block such packets.

Tunneling also adds an additional packet header to a packet, inflating the size of the packet. Such an expansion of the packet on certain path elements of the network may cause path packet size problems, increasing the risk of encountering Path *Maximum Transmission Unit* (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, because another form of tunneling could possibly be an answer. *Teredo* is described in RFC 4380[5].

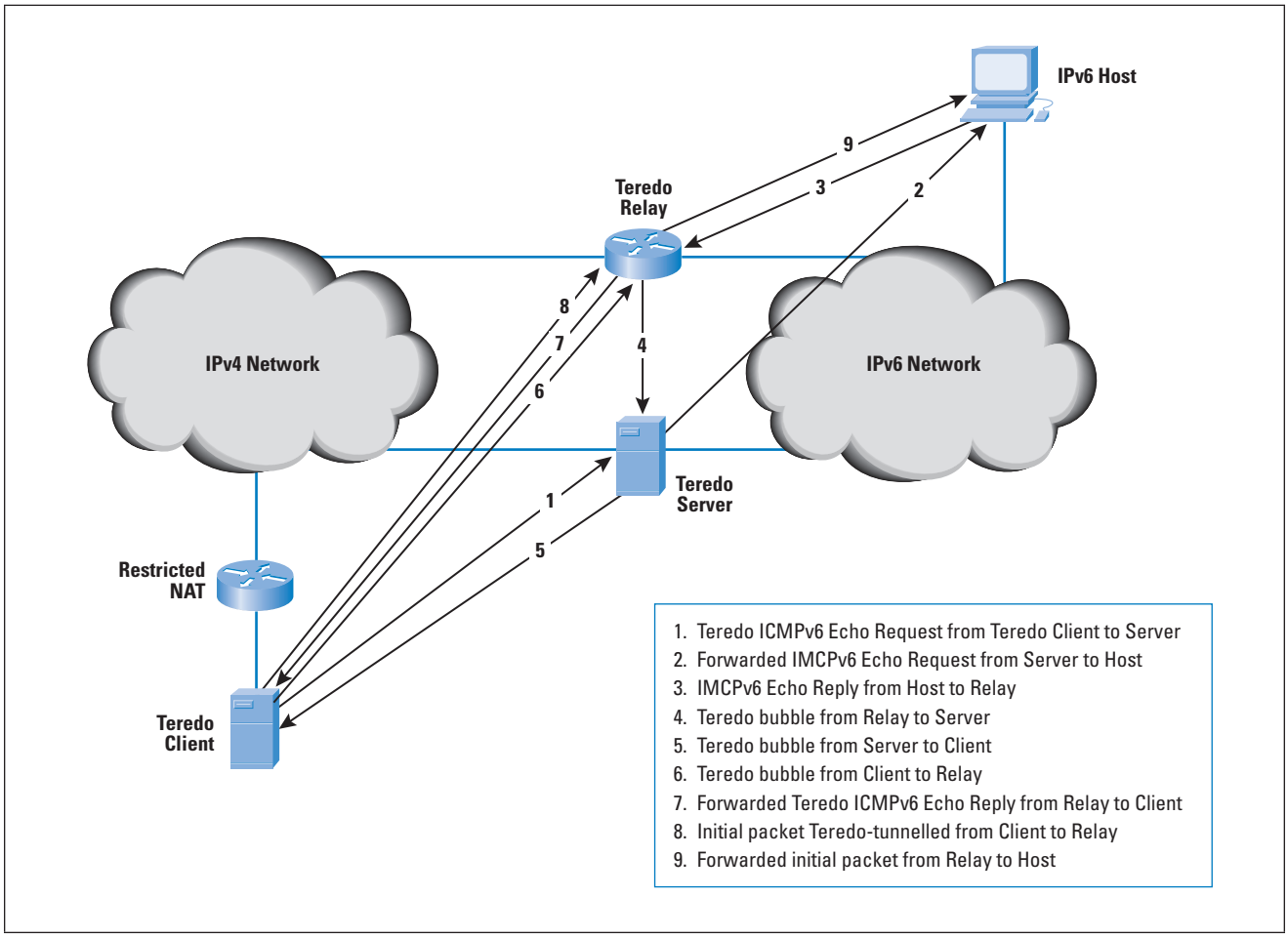
Teredo, like 6to4, is an autotunneling protocol coupled with an addressing structure. Like 6to4, Teredo uses its own address prefix, and all Teredo addresses share a common IPv6 /32 address prefix, namely `2001:0000::/32`. The next 32 bits are the IPv4 address of the Teredo server. The IPv6 interface identifier field is used to support NAT traversal, and it is encoded with the triplet of a field describing the NAT type, the view of the relay 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 view of the relay 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 *Session Traversal Utilities for NAT* (STUN)[6] active probing approach to determine the type of NAT; it 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 tunneling protocol used by Teredo is not the simple IPv6-in-IPv4 protocol 41 used by 6to4. NAT devices are sensitive to the transport protocol and generally pass only TCP and UDP transport protocols. In the Teredo case the tunneling is UDP, so all IPv6 Teredo packets are composed of an IPv4 packet header and a UDP transport header, followed by the IPv6 packet as the UDP payload. Teredo uses a combination of ICMPv6[7] message exchanges to set up a connection and tunneled packets encapsulated using an outer IPv4 header and a UDP header, and it contains the IPv6 packet as a UDP payload.

It should be noted that this reliance on ICMPv6 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.

Figure 2: Teredo Tunneling



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. Figure 2 shows an example packet exchange that Teredo uses when the client is behind a Restricted NAT.

Teredo represents a different set of design trade-offs as compared to 6to4. In its desire to be useful in an environment that includes NAT functions in the IPv4 path, Teredo is a per-host connectivity approach, as compared to the 6to4 approach, which can support both individual hosts and entire end sites within the same technology. Also, Teredo is a host-centric multiparty 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 turn on Teredo in your dual-stack host; if everything goes according to theory, you should be able to establish IPv6 connectivity. It is highly likely that the IPv6 Teredo connection will fail in strange ways, and, like 6to4, this is a technology best avoided!

Tunnel Brokers

In contrast to these autotunnel approaches, the simplest form of tunneling 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 the 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 *Virtual Private Network* (VPN) tunnels, the tunnel can be configured using UDP or TCP, and with some care, the tunnel can be configured through NAT functions in the same way as VPN tunnels can be configured through NAT functions.

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 will be 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 approach a viable one for a larger mass market of consumer devices and services.

Client Conclusions

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

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 its service offering, and offer clients a dual-stack service in both IPv4 and IPv6.

IPv6 for Internet Service Providers

Although the "self-help" autotunneling approaches for clients outlined earlier in this article are a possible answer, their utility is appropriately restricted to a very small number of end clients who have the necessary technical expertise and who are willing to debug some rather strange resultant potential problems relating to asymmetric paths, third-party relays, potential MTU mismatches, and interactions with filters. This approach is not a reasonable one for the larger Internet.

From the perspective of the mass market for Internet Services, we cannot assume that clients have the motivation, expertise, and means to bypass their ISP and set up IPv6 access on their own, either through autotunneling or manually configured tunnels. The inference from this observation is that for as long as the mass-market ISPs do not commit to IPv6 services, and for as long as they continue to stall in deploying services supporting dual access for their clients, the entire IPv6 transition story remains effectively stalled.

How can ISPs support IPv6 access for their clients?

The Dual-Stack Service Network

Perhaps it is obvious, but the most direct response here is for the ISP to operate a *Dual-Stack Network*.

And the most direct way to achieve this operation is for the ISP's infrastructure to also support IPv6 wherever there is IPv4, so that the delivery of services to the ISP's clients in IPv6 faithfully replicates the service offered in IPv4.

This solution implies that the network needs to support IPv6 in the ISP's routing infrastructure, in the network data plane, in the load-management systems, in the operational support infrastructure, in access and accounting, and in peering and in transit. In short, wherever there is IPv4 there needs to be IPv6.

The infrastructure elements that require dual-stack service at the next level include the routing and switching elements, including the internal and external routing protocols. The task includes negotiating peering and transit services in IPv6 to complement those in IPv4. Network infrastructure also includes VPN support and other forms of tunnels, as well as data center front-end units, including load balancers, filters and firewalls, and various virtualized forms of service provision. The task also includes integration of IPv6 in the network management subsystem and the related network measurement and reporting system. Even a comprehensive audit of the supported *Management Information Bases* (MIBs) in the active elements of the network to ensure that the relevant IPv6 MIBs are supported is an essential task. A similar task is associated with equipping the server infrastructure with IPv6 support, and at the higher levels of the protocol stack are the various applications, including web services, mail, *Domain Name System* (DNS), authentication and accounting, *Voice over IP* (VoIP) servers, Load Balancers, Cloud Servers, and similar applications.

And those are just the common elements of most ISPs' infrastructures. Every ISP also has more specialized elements in its service portfolio, and each one of these elements also requires a comprehensive audit to ensure that there is an IPv6 solution for each of these elements that leads to a comprehensive dual-stack outcome.

As obvious as this approach might appear, it has two significant problems. First, it requires a comprehensive overhaul of every element in the ISP's service network. Even for small-scale ISPs this overhaul is not trivial, and for larger service provider platforms it is an exercise that may take months if not years and make considerable inroads into the operating budgets of the ISPs. Secondly, it still does not account for the inevitable fact that in the coming months the current supply lines of IPv4 addresses will end and any continued expansion of the service platform will require some different approaches to the way in which IPv4 addresses are deployed in the service platform.

Although the approach of simply provisioning IPv6 alongside IPv4 in a simple dual-protocol service infrastructure may appear to be the most obvious response to the need to transition to IPv6, it may not necessarily be the most appropriate response for many ISPs to the dual factors of IPv6 transition and IPv4 address exhaustion.

Are there alternative approaches for ISPs? Of course.

Hybrid Approaches

Saying that an ISP must deploy IPv6 across all of its infrastructure and actually doing it are often quite different. The cost of converting all parts of an ISP's operation to run in dual-stack mode can be quite high, and the benefit of running every aspect of an ISP's service offering in dual-stack mode is dubious at best.

Are there middle positions here? Is it possible for an ISP to deliver robust IPv6 services to clients while still operating an IPv4-only internal network? One way to look at an ISP's network is as a transit conduit (Figure 3).

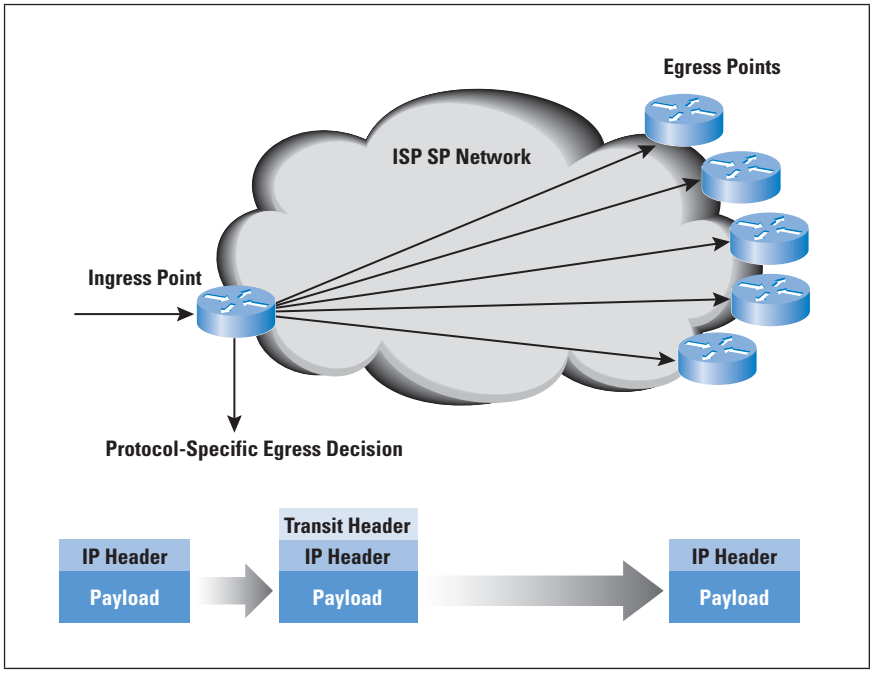


Figure 3: Generic ISP Packet Transit Architecture

The ISP needs to be able to accept packets from an external interface, determine the appropriate egress point for the packet within the context of the local network, and then ensure that the packet is passed out this egress interface. The internal network need not operate in the same protocol context as the protocol of the packets the network is handling. Viewed at a level of the minimal essentials, the network needs to be able to have some protocol-specific capability at its ingress points in order to determine the appropriate egress point of each incoming packet, and thereafter during the transit of the service provider's network, the minimum necessary association to maintain the identity of this preselected egress point with the packet. Now if the network uniformly supports the same protocol as the packet, then the same egress decision can be made at each forwarding point within the network.

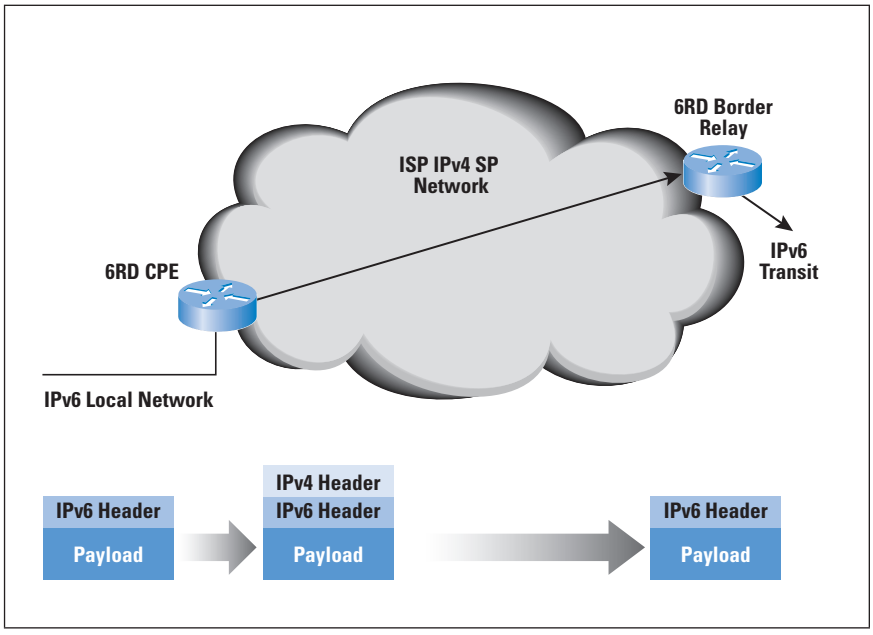
Alternatively, the packet can be encapsulated with an outer wrapper that identifies the egress point using the same protocol context as that used by the service provider's internal switching elements, and the packet can be passed through the service provider's transit network using only this temporary wrapper to determine the sequence of forwarding decisions. *Multiprotocol Label Switching* (MPLS) networks are an excellent example of this form of approach, as are other forms of IP-in-IP encapsulation. The advantage of this approach is that the internal infrastructure of the service provider network need not be altered to support additional carriage protocols: the changes to specifically support IPv6 are required only at the network ingress elements, and a basic encapsulation stripping function is used at all egress points.

With this information in mind, let's look at some of these hybrid approaches to supporting IPv6 in a service provider network.

6RD

6RD, described in RFC 5969[8], is an interesting refinement of the 6to4 approach. It shares the same basic encapsulation protocol and the same address structure of embedding of the IPv4 tunnel endpoint into the IPv6 address. However, it has removed the concept of third-party relays and the use of the common `2002::/16` IPv6 prefix, and instead uses the provider's IPv6 prefix. The effect of these changes is to limit the scope of the tunneling mechanism to that of tunneling across the network infrastructure of a single provider, and the intended function is to tunnel from the *Customer Premises Equipment* (CPE) to IPv6 *Border Relays* operated by the customer's ISP (Figure 4).

Figure 4: 6RD Tunneling



If 6to4 is not recommended for use because of high failure rates of connections and suboptimal performance, then why would 6RD be any better?

The most compelling reason to believe that 6RD will perform more reliably than 6to4 is that 6RD removes the wild-card third-party relay element from the picture. For outbound traffic the CPE provides the tunnel encapsulation, which is, hopefully, under the ISP's operational control. The IPv6-in-IPv4 tunnel is directed to the ISP's own 6RD Border Relay rather than the 6to4 relay anycast address. Because this process is also under the ISP's direct operational control, it eliminates the outbound third-party relay function. For the reverse path, the use of the provider's own IPv6 prefix in 6RD, instead of the generic `2002::/16` prefix, ensures that the inbound packets are sent through IPv6 directly to the ISP, and the IPv6-in-IPv4 tunnel is again limited to a hop across the ISP's own internal infrastructure.

As long as the ISP effectively manages all CPE devices, and as long as the CPE itself is capable of supporting the configuration of additional functional modules that can deliver unicast IPv6 to the client and 6RD tunnels inward to the ISP, then 6RD is a viable option for the ISP. At the cost of upgrading the CPE set to include 6RD support, and the cost of deployment of 6RD Border Relays that terminate these CPE tunnels, together with IPv6 transit from these Border Relays, the ISP is in a position to provide dual-stack support to its client base from an internal network platform that remains an IPv4 service platform, thereby deferring the process of conversion of its entire network infrastructure base to support IPv6.

For ISPs seeking to defray the internal infrastructure IPv6 conversion costs over a number of years, or for ISPs seeking an incremental path to IPv6 support that allows the existing infrastructure to remain in place temporarily, 6RD can be an interesting and cost-effective alternative to a comprehensive dual-stack deployment, as long as the ISP has some mechanism to load the CPE with IPv6 support and 6RD relay functions.

MPLS and 6PE

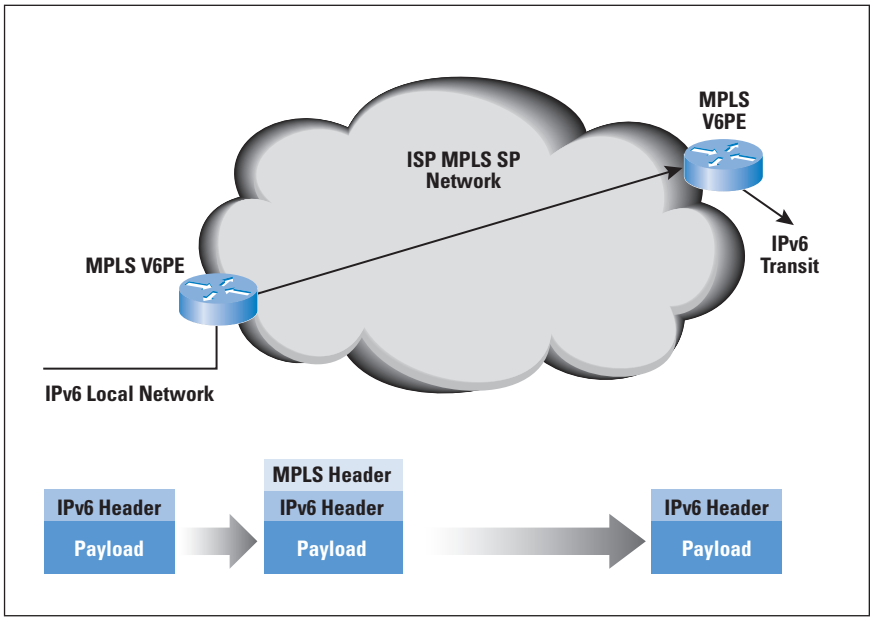
The 6RD approach has many similarities to MPLS, in that an additional header is added to incoming packets at the network boundary, and the encapsulation effectively directs the packet to the appropriate network egress point (as identified by ingress), where the encapsulation is stripped and the original packet is passed out.

Rather than using an IPv4 header to direct a packet from ingress to egress, if the network is already using MPLS, why not simply support IPv6 on an existing MPLS network as a PE-to-PE MPLS path set and bypass the IPv4 step?

Why not, indeed, and RFC 4659[9] describes how this bypass can be achieved.

If you are running an MPLS network, then the role of the interior routing protocol and label distribution function is to maintain viable paths between all network ingress and egress points. The protocol-specific function in such networks is not the interior network topology management function, but the maintenance of the mapping of egress to protocol-specific destination addresses (Figure 5).

Figure 5: MPLS and 6PE



As with 6RD, if the local problem is some form of prohibitive barrier to the immediate deployment of IPv6 in a dual-stack configuration across the network infrastructure, then this approach allows an IPv4 MPLS network to set up paths across the network IPv4 MPLS infrastructure from provider edge to provider edge. These paths may be used to tunnel IPv6 packets across the network by associating the IPv6 destination address of the incoming packet with the IPv4 address of the egress router, using the *interior Border Gateway Protocol* (iBGP) *Next-Hop* address, for example.

The incremental changes to support IPv6 are constrained to adding IPv6 to the service provider's iBGP routing infrastructure, and to the provider-edge devices in the MPLS network, while all other parts of the service provider's service platform can continue to operate as an MPLS IPv4 network for now.

IPv4 Address Compression

It is not just the challenge of adding a new protocol to the existing IPv4 network infrastructure that confronts ISPs. The entire reason for this activity is the prospect of exhaustion of supply of IPv4 addresses. When this prospect was first aired, in 1990, it was assumed that the Internet would be supported by industry players that acted rationally in terms of common interests.

One of the more critical assumptions made in the development of transitional tools was that transition activity would be undertaken well in advance of IPv4 address exhaustion. Competitive interest would see each actor making the necessary investments in new technologies to mitigate the risks of attempting to operate a network in an environment of acute general scarcity of addresses. As much fun as the debate as to whom the "last" IPv4 address should be given might be, it was assumed that this event was, in fact, never going to happen. The assumption was that industry actors would anticipate this situation and take the necessary steps to avoid it. The transition to IPv6 would be effectively complete well before the stocks of IPv4 addresses had been exhausted, and IPv4 addresses would be an historical artefact well before we needed to use the last one!

Obviously, this scenario has not happened.

This industry is going to exhaust the available supplies of IPv4 addresses well before the transition to IPv6 is complete—and in some cases well before the transition process has even commenced! This situation creates an additional challenge for ISPs and the Internet, and raises a further question as well. The challenge is to fold into this dual-stack transition the additional factor of having to work with fewer and fewer IPv4 addresses as the transition process continues. This situation implies that the necessary steps that the ISP must take include ones that increase the intensity of use of each IPv4 address, and wherever possible substitute a private-use IPv4 address for public IPv4 addresses.

The question that this scenario raises is one of guessing how long this hybrid model of an Internet where a significant proportion of network services and network clients remains entrenched in an IPv4-only world will persist. For as long as such IPv4-only network domains persist, and for as long as these IPv4-only network domains encompass significant service and customer populations, all the other parts of the Internet are forced to maintain residual IPv4 capability and cannot transition their customers and services to an IPv6-only environment. Students of economic game theory may see some rich areas of study in this developing situation.

More practically, for an ISP the question becomes one of attempting to understand how long this hybrid period of attempting to operate a dual-stack network with continuing postexhaustion demand for further IPv4 addresses will last. Will an after-market for the redistribution of addresses emerge? How will the increasing scarcity pressure affect pricing in such a market? How long will demand persist for IPv4 addresses in the face of escalating prices? Will the industry turn to IPv6 in a rapid surge in response to cost escalation for additional IPv4 addresses, or will a dual-stack transition lumber on for many years? In such a large, diverse, heterogeneous environment of today's Internet, the one constant factor is that the immediate future of the Internet is clouded with extremely high levels of uncertainty.

The cumulative effect of the individual decisions made by service providers, enterprises, carriers, vendors, policy makers, and consumers has created a somewhat chaotic environment that adds a significant level of uncertainty and associated investment risk into the current planning process for ISPs.

Carrier-Grade NATs

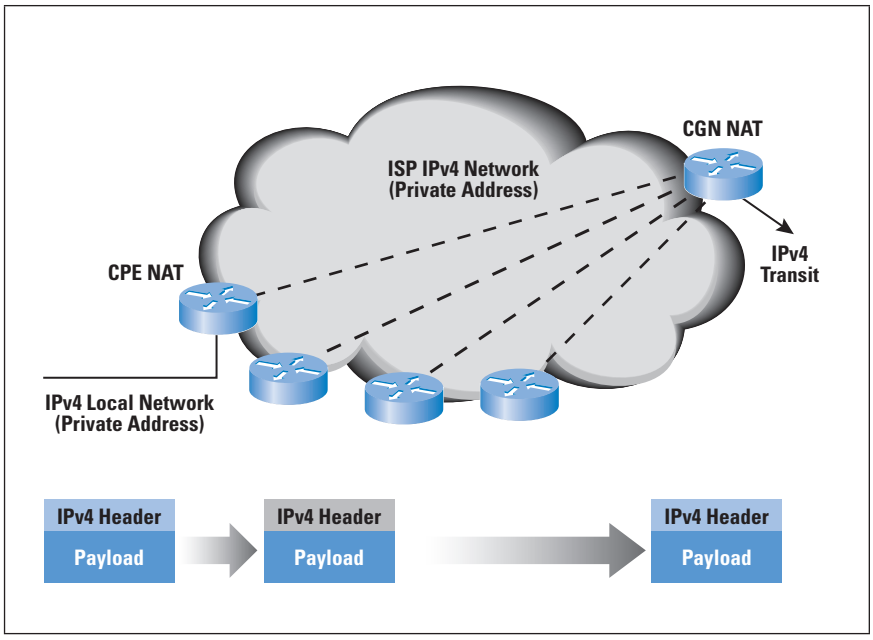
I have often heard it said that address scarcity in IPv4 is nothing new, and it first occurred when the first NAT device that supported port mapping was deployed. At this point the concept of *address sharing* was introduced to the Internet, and, from the perspective of the NAT industry, we have not looked back since.

In today's world NATs are extremely commonplace. Most clients are provisioned with a single address from their ISP, which they then share across their local network using a NAT. Whether it is well advised or not, NATs typically form part of a client's network security framework, and they often are an integral part of a customer's multihoming configuration if the client uses multiple providers.

But in this model of NATs as the CPE, the ISP uses one IPv4 address for each client. If the ISP wants to achieve greater levels of address compression, then it is necessary to share a single IPv4 address across multiple customers.

The most direct way to achieve this scenario is for ISPs to operate their own NAT, variously termed a *Carrier-Grade NAT* (CGN) or a *Large-Scale NAT* (LSN), or *NAT444*. This approach is the simplest, and, in essence, is a case of "more of the same" (Figure 6).

Figure 6: Carrier-Grade NATs



The Carrier-Grade NAT allows a single public address to be shared across multiple clients, who, in turn, further share this address across the end systems in their local networks.

From behind the CPE in the client edge network not much has changed with the addition of the CGN in terms of application behavior. It still requires an outbound packet to trigger a binding that would allow a return packet through to the internal destination, so nothing has changed there. Other aspects of NAT behavior, notably the NAT binding lifetime and the form of NAT "cone behavior" for UDP, take on the more restrictive of the two NAT functions in sequence. The binding times are potentially problematic in that the two NATs are not synchronized in terms of binding behavior. If the CGN has a shorter binding time, it is possible for the CGN to misdirect packets and cause application-level problems. However, this situation is not overly different from a single-level NAT environment where aggressively short NAT binding times also run the risk of causing application-level problems when the NAT drops the binding for an active session that has been quiet for an extended period of time.

However, one major assumption is broken in this structure, namely that an IP address is associated with a single customer. In the CGN model a single public IP address may be used simultaneously by many customers at once, albeit on different port numbers. This scenario has obvious implications in terms of some current practices in filters, firewalls, "black" and "white" lists, and some forms of application-level security and credentials where the application makes an inference about the identity and associated level of trust in the remote party based on the remote party's IP address.

This approach is not without its potential operational problems as well. For the service provider, service resiliency becomes a critical concern in so far as moving traffic from one NAT-connected external service to another will cause all the current sessions to be dropped. Another concern is one of resource management in the face of potentially hostile applications. For example, an end host infected with a virus may generate a large amount of probe packets to a large range of addresses. In the case of a single edge NAT, the large volumes of bindings generated by this behavior become a local resource-management problem because the customer's network is the only affected site. In the case where a CGN is deployed, the same behavior will consume port-binding space on the CGN and, potentially, can starve the CGN of external address port bindings. If this problem is seen to be significant, the CGN would need to have some form of external address rationing per internal client in order to ensure that the entire external address pool is not consumed by a single errant customer application.

The other concern here is one of *scalability*. Whereas the most effective use of the CGN in terms of efficiency of usage of external addresses occurs when the greatest numbers of internal edge NATed clients are connected, there are some real limitations in terms of NAT performance and address availability when a service provider wants to apply this approach to networks where the customer population is in the millions or larger. In this case the service provider must use an IPv4 private address pool to number every client. But if network 10 is already used by each customer as its "internal" network, then what address pool can be used for the service provider's private address space? One of the few answers that come to mind is to deliberately partition the network into numerous discrete networks, each of which can be privately numbered from `172.16.0.0/12`, allowing for some 600,000 or so customers per network partition, and then use a transit network to "glue" together the partitioned elements.

The advantage of the CGN approach is that nothing changes for the customer. There is no need for any customers to upgrade their NAT equipment or change it in any way, and for many service providers this motivation is probably sufficient to choose this path. The disadvantages of this approach lie in the scaling properties when looking at very large deployments, and the concerns of application-level translation, where the NAT attempts to be "helpful" by performing *Deep Packet Inspection* and rewriting what it thinks are IP addresses found in packet payloads. Having one NAT do this process is bad enough, but loading them up in sequence is a recipe for trouble.

Are there alternatives?

The Address-plus-Port Approach

One NAT in the path is certainly worse than none from the perspective of application agility and functions. And two NAT functions do not make it any better! Inevitably, that second NAT device adds some additional levels of complexity and fragility into the process.

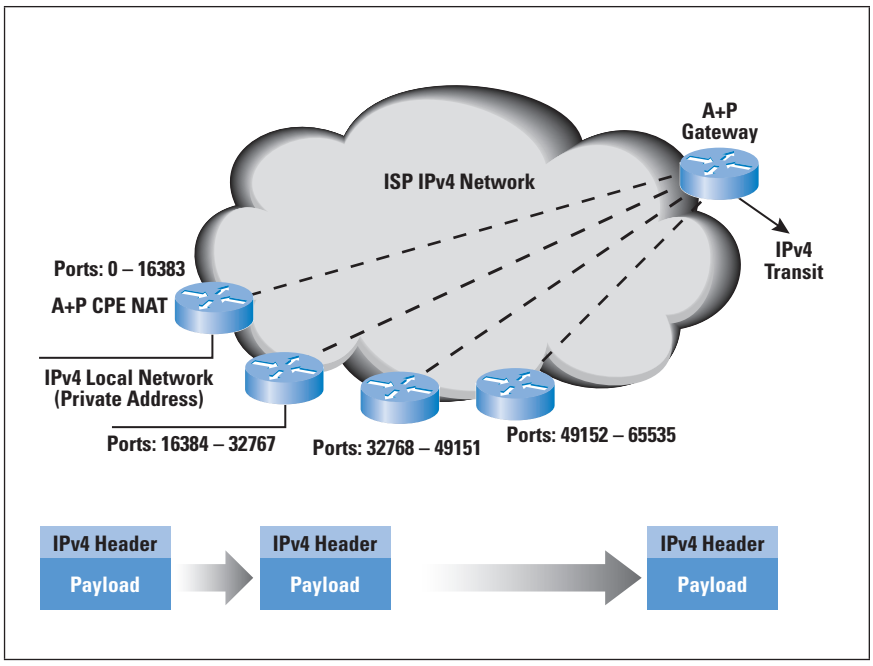
The question is, can these two NAT functions be collapsed back into a single NAT, yet still allow sharing of public IPv4 addresses across multiple end clients? CPE NAT devices currently map connections into the 16-bit *port* field of the single external address. If the CPE NAT could be coerced into performing this mapping into, say, 15 bits of the port field, then the external address could be shared between two edge CPEs, with the leading bit of the port field denoting which CPE. Obviously, moving the bit marker further across the port field will allow more CPE devices to share the one address, but it will reduce the number of available ports for each CPE in the process.

The theory is again quite simple. The CPE NAT is dynamically configured with an external address, as happens today, and a port range, which is the additional constraint. The CPE NAT performs the same function as before, but it is now limited in terms of the range of external port values it can use in its NAT bindings to those that lie within the provided port range. Other CPE devices are concurrently using the same external IP address, but with a different port range.

For outgoing packets this scenario implies only a minor change to the network architecture, in that the RADIUS exchange to configure the CPE now must also provide a port range to the CPE device. The CPE is then constrained such that as it maps private addresses and TCP or UDP port values to the external address and port values, the mapped port value must fall within the configured range.

The handling of incoming packets is more challenging. Here the service provider must forward the packet based not only on the destination IP address, but also on the port value in the TCP or UDP header, because there are now multiple CPE egress points that share the same IP address. A convenient way to perform forwarding is to take the Dual-Stack Lite approach and use an IPv4-in-IPv6 tunnel between the CPE and the external address-plus-port (A+P) gateway. This address-plus-port gateway needs to be able to associate each address and port range with the IPv6 address of a CPE (which it can learn dynamically as it decapsulates outgoing packets that are similarly tunneled from the CPE to the address-plus-port gateway). Incoming packets are encapsulated in IPv6 using the IPv6 destination address that it has learned previously. In this manner the NAT function is performed just once, at the edge, much as it is today, and the interior device is a more conventional form of tunnel server (Figure 7).

Figure 7: Address-plus-Port-Approach



This approach relies on every CPE device being able to operate using a restricted port range, to perform IPv4-in-IPv6 tunnel ingress and egress functions, and act as an IPv6 provisioned endpoint for the service provider network. This set of constraints is perhaps unrealistic for many service provider networks. Further modifications to this model propose the use of an accompanying CGN operated by the service provider to handle those CPE devices that cannot support this address-plus-port function.

This approach has some positive aspects. Pushing the NAT function back to the network edge has some considerable advantage over the approach of moving the NAT to the interior of the network. The packet rates are lower at the edge, allowing for commodity computing to process the NAT functions across the offered packet load without undue stress. The ability to control the NAT behavior with the *Internet Gateway Device* protocol as part of the *Universal Plug and Play* (uPnP) framework will still function in an environment of restricted port ranges. Aside from the initial provisioning process to equip the CPE NAT with a port range, the CPE and the edge environment are largely the same as that of today's CPE NAT model.

That is not to say that this approach is without its negative aspects, and it is unclear as to whether the perceived benefits of a "local" NAT function outweigh the problems in this particular model of address sharing. The concept of port "rationing" is a very suboptimal means of address sharing, given that when a CPE is assigned a port range, those port addresses are unusable by any other CPE. The prudent service provider would assign to each CPE a port address pool equal to some estimate of peak demand, so that, for example, each CPE would be assigned some 1024 ports, allowing a single external IP address to be shared across only some 60 such CPE clients. The Carrier-Grade NAT and Dual-Stack Lite approaches do not attempt this form of rationed allocation, allowing the port address pool to be treated as a common resource, with far higher levels of usage efficiency. The leverage obtained in terms of efficiently using these additional 16 bits of address space is reduced by the imposition of a fixed boundary between customer and service provider use. The central NAT model effectively pools the port address range and would result in more efficient sharing of this common pool across a larger client base.

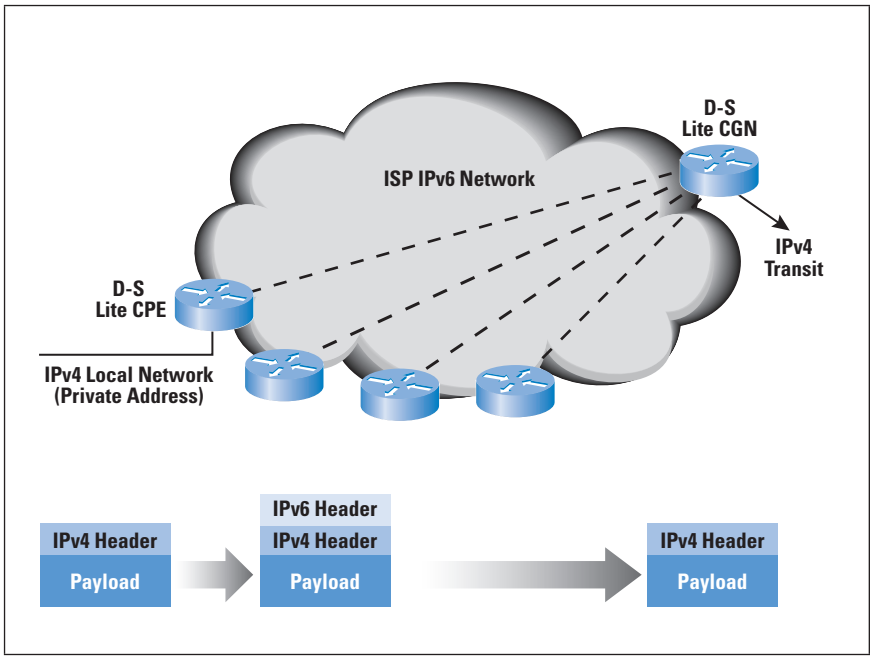
The other consideration here is that this approach means a higher overhead for the service provider, in that the service provider would have to support both "conventional" CPE equipment and address-plus-port equipment. In other words, the service provider will have to deploy a CGN and support customer CPE using a two-level NAT environment in addition to operating the address-plus-port infrastructure. Unless customers would be willing to pay a significant price premium for such address-plus-port service, it is unlikely that this option would be attractive for the service provider as an additional cost above the CGN cost.

Dual-Stack Lite

The concept behind the *Dual-Stack Lite* approach is that the service provider's network infrastructure will need to support IPv6 running in native mode in any case, so is there a way in which the service provider can continue to support IPv4 customers without running IPv4 internally?

Here the customer NAT is effectively replaced by a tunnel ingress-egress function in the Dual-Stack Lite home gateway. Outgoing IPv4 packets are not translated, but are encapsulated in an IPv6 packet header, which contains a source address of the carrier side of the home gateway unit, and a destination address of the ISP's gateway unit. From the service provider's perspective, each customer is no longer uniquely addressed with an IPv4 address, but instead is addressed with a unique IPv6 address, and provided with the IPv6 address of the provider's combined IPv6 tunnel egress point and IPv4 NAT unit (Figure 8).

Figure 8: Dual-Stack Lite



The service provider's Dual-Stack Lite gateway unit will perform the IPv6 tunnel termination and a NAT translation using an extended local binding table. The NAT "interior" address is now a 4-tuple of the IPv4 source address, protocol ID, and port, plus the IPv6 address of the home gateway unit, while the external address remains the triplet of the public IPv4 address, protocol ID, and port. In this way the NAT binding table contains a mapping between interior "addresses" that consist of IPv4 address and port plus a tunnel identifier, and public IPv4 exterior addresses. This way the NAT can handle a multitude of net 10 addresses, because they can be distinguished by different tunnel identifiers.

The resultant output packet following the stripping of the IPv6 encapsulation and the application of the NAT function is an IPv4 packet with public source and destination addresses. Incoming IPv4 packets are similarly transformed, where the IPv4 packet header is used to perform a lookup in the Dual-Stack Lite gateway unit, and the resultant 4-tuple is used to create the NAT-translated IPv4 packet header plus the destination address of the IPv6 encapsulation header.

The advantage of this approach is that there now needs to be only a single NAT in the end-to-end path, because the functions of the customer NAT are now subsumed by the carrier NAT. This scenario has some advantages in terms of those messy "value-added" NAT functions that attempt to perform deep packet inspection and rewrite IP addresses found in data payloads. There is also no need to provide each customer with a unique IPv4 address, public or private, so the scaling limitations of the dual-NAT approach are also eliminated. The disadvantages of this approach lie in the need to use a different CPE device—or at least one that is reprogrammed. The device now requires an external IPv6 interface and at the minimum an IPv4/IPv6 tunnel gateway function. The device can also include a NAT if so desired, but it is not required in terms of the basic Dual-Stack Lite architecture.

This approach pushes the translation into the interior of the network, where the greatest benefit can be derived from port multiplexing, but it also creates a critical hotspot for the service itself. If the Dual-Stack Lite NAT fails in any way, the entire customer base is disrupted. It seems somewhat counterintuitive to create a resilient end-to-end network with stateless switching environments and then place a critical stateful unit right in the middle!

Protocol Translation

So far we have looked at two general forms of approach to hybrid networks that are intended to support both IPv6 transition and greater levels of address usage in IPv4, namely address mapping and tunneling. A third approach lies in the area of protocol translation.

RFC 2765[10] contains the details of a relatively simple protocol-translation mechanism. The approach relies on the basic observation that IPv6 did not make any radical changes to the basic IP architecture of IPv4, and that it was therefore possible to define a stateless mapping algorithm that could translate between certain IPv4 and IPv6 packets. Of course the one major problem here is that there are far more addresses in IPv6 than in IPv4, so the approach used was to map IPv4 addresses into the trailing 32 bits of the IPv6 address prefix `::FFFF:0:0/96`. The approach assumed that to the IPv6-only end host the entire IPv4 network was visible in this mapped IPv6 prefix, and that when the IPv6-only end host wished to communicate with a remote host who was addressed using this IPv4-mapped prefix it would use a source address also drawn from the same IPv4-mapped prefix. In other words, it assumed that all IPv6-only hosts were also assigned a unique IPv4 address.

The *NAT-Protocol Translation* (NAT-PT) approach attempted to relax this constraint, allowing IPv6-only hosts to use a dynamic mapping to a public IPv4 address through the NAT-PT function, in the same way as NAT functions work in an all-IPv4 domain (Figure 9). The proposed approach assumed that the local host was located behind a modified DNS environment where the IPv4 "A" record of an IPv4-only remote service is translated by the DNS gateway into a local IPv6 address where the initial 96 bits of the IPv6 address identify the internal address of the NAT-PT gateway and the trailing 32 bits are the IPv4 address of the remote service. When the local host then uses this address as an IPv6 destination address, the packet is directed by the local routing environment to the NAT-PT device. This device can construct an "equivalent" IPv4 packet by using the local IPv4 address as the source address and the last 32 bits of the IPv6 address as the destination address, and bind the IPv6 source port to a free local port value. These sets of transforms can be locally stored as an active NAT binding. Return IPv4 packets can be mapped back into their "equivalent" IPv6 form by using the values in the binding to perform a reverse set of transforms on the IP address and port fields of the packet.

This approach was published as RFC 2766[11] in February 2000. Some 7 years later in July 2007, the IETF published RFC 4966[12], deprecating NAT-PT to "historic," with an associated list of applications that would not operate correctly through such a device. This negative judgement of NAT-PT seems rather curious to me, given that conventional CPE NAT functions in IPv4 appear to share most, if not all, of the same shortfalls that are listed in RFC 4966. Given the extensive set of compromises that are required in the environment that is partially crippled by IPv4 address exhaustion, it seems rather contradictory to insist upon extremely high levels of functions and robustness from these hybrid translation approaches.

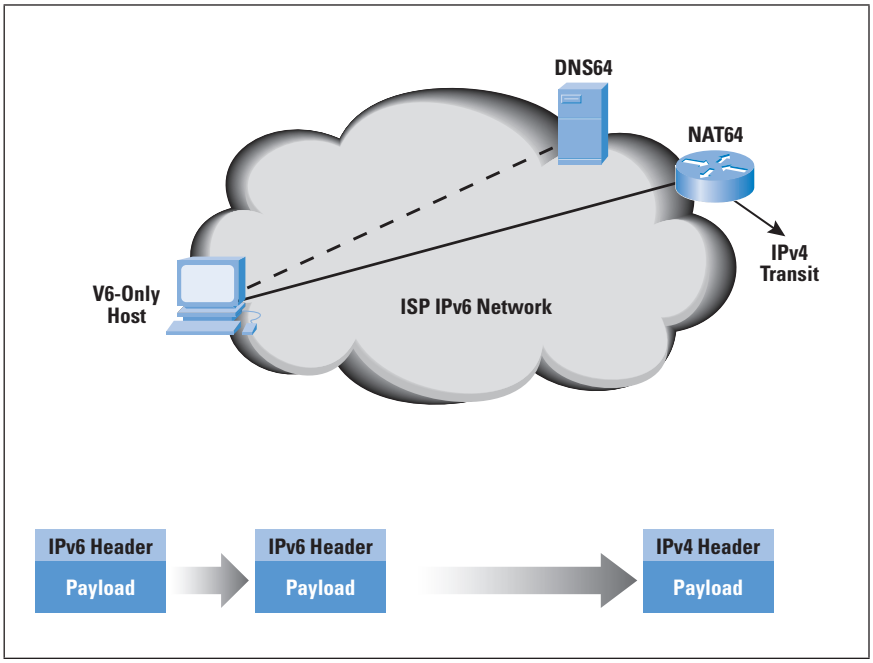


Figure 9: NAT Protocol Translation – NAT64

Not unsurprisingly, NAT-PT is undergoing a revival, this time under the name "NAT64." Not much has changed from the basic approach outlined in NAT-PT. The IPv6-only client performs a DNS lookup through a modified DNS server that is configured with DNS64. If the queried name contains only an IPv4 address, the DNS64 server synthesises an IPv6 response by merging the prefix address of the NAT64 gateway with the IPv4 address. When the client uses this address, the IPv6 packet is directed to the NAT64 gateway, and the same transform as described previously for NAT-PT takes place.

This setup is similar to the CGN model, in so far as the service provider operates a common NAT that shares an IPv4 address pool across a set of end clients.

ISP Conclusions

There really is no single clear path forward from this point. Different ISPs will see some advantages in pursuing different approaches to this dual problem of introducing IPv6 into their service portfolio and at the same time introducing additional measures that allow more efficient use of IPv4 addresses.

However, one common theme is becoming clear. So far ISPs have been able to "externalize" many of these problems by pushing much of the complexity and fragility of NAT functions out to the customer and loading up the CPE with these functions. This approach of externalizing much of the complexity of address compression in NAT functions over to the customer's network cannot be sustained with the IPv6 transition, and no matter which approach is used, whether it is a CGN, NAT64, Dual-Stack Lite, 6RD, or MPLS with 6PE, the ISP now has to actively participate in the delivery of IPv6 and in increasing the efficiency of the use of IPv4.

So for the ISP it is time to start making some technical choices as to how to address the combination of these two rather unique challenges of transition and exhaustion.

References

[1] Daniel Karrenberg, Gerard Ross, Paul Wilson, and Leslie Nobile, "Development of the Regional Internet Registry System," *The Internet Protocol Journal*, Volume 4, No. 4, December 2001.

[2] Geoff Huston, "Anatomy: A Look inside Network Address Translators," *The Internet Protocol Journal*, Volume 7, No. 3, September 2004.

[3] Alex Conta and Stephen Deering, "Generic Packet Tunneling in IPv6 Specification," RFC 2473, December 1998.

[4] Yakov Rekhter, Bob Moskowitz, Daniel Karrenberg, Geert Jan de Groot, and Eliot Lear, "Address Allocation for Private Internets," RFC 1918, February 1996.

[5] Christian Huitema, "Teredo: Tunneling IPv6 over UDP through Network Address Translations (NATs)," RFC 4380, February 2006.

[6] Jonathan Rosenberg, Rohan Mahy, Philip Matthews, and Dan Wing, "Session Traversal Utilities for NAT (STUN)," RFC 5389, October 2008.

[7] Alex Conta, Stephen Deering, and Mukesh Gupta, "Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6 (IPv6) Specification," RFC 4443, March 2006.

[8] Mark Townsley and Ole Troan, "IPv6 Rapid Deployment on IPv4 Infrastructures (6rd) – Protocol Specification," RFC 5969, August 2010.

[9] Jeremy De Clercq, Dirk Ooms, Marco Carugi, and Francois Le Faucheur, "BGP-MPLS IP Virtual Private Network (VPN) Extension for IPv6 VPN," RFC 4659, September 2006.

[10] Erik Nordmark, "Stateless IP/ICMP Translation Algorithm (SIIT)," RFC 2765, February 2000.

[11] George Tsirtsis and Pyda Srisuresh, "Network Address Translation – Protocol Translation (NAT-PT)," RFC 2766, February 2000.

[12] Cedric Aoun and Elwyn Davies, "Reasons to Move the Network Address Translator - Protocol Translator (NAT-PT) to Historic Status," RFC 4966, July 2007.

Further Reading

The IETF has been working on the issues related to the transition to IPv6 for the past 18 years, and in the intervening period has generated many hundreds of documents. In selecting the following documents as a helpful reading list, I have tried to select only from the more recent documents and those that are overviews of transition technologies rather than reference specifications for individual technologies.

[1] Jari Arkko and Fred Baker, "Guidelines for Using IPv6 Transition Mechanisms during IPv6 Deployment," Internet Draft, Work in Progress, December 2010.

The document discusses the IPv6 deployment models and migration tools, and considers what appears to be effective in networks to date. This Internet Draft, **`draft-arkko-ipv6-transition-guidelines-14.txt`***, is about to be published as an Informational RFC.*

[2] Brian Carpenter and Sheng Jian, "Emerging Service Provider Scenarios for IPv6 Deployment," RFC 6036, October 2010.

This document describes practices and plans that are emerging among Internet Service Providers for the deployment of IPv6 services, using data collected in a survey of numerous ISPs carried out in early 2010.

[3] Reinaldo Penno, Tarun Saxena, Mohamed Boucadair, and Senthil Sivakumar, "Analysis of 64 Translation," Internet Draft, Work in Progress, **draft-ietf-behave-64-analysis-01**, January 2011.

This paper is a working document of the IETF's BEHAVE Working Group. The document notes that because of specific problems, NAT-PT was deprecated by the IETF as a mechanism to perform IPv6-IPv4 translation. Since then, new efforts have been undertaken within IETF to standardize alternative mechanisms to perform IPv6-IPv4 translation. This document evaluates how the new translation mechanisms avoid the problems that caused the IETF to deprecate NAT-PT.

[4] Fred Baker, Xing Li, and Kevin Yin, "Framework for IPv4/IPv6 Translation," Internet Draft, Work in Progress, August 2010.

It is common in the IETF these days to generate a "framework" document as part of the process of developing technical specifications. This draft is a framework document for the general IPv4/IPv6 translation technology. This Internet Draft, **draft-ietf-behave-v6v4-framework-10.txt***, will soon be published as an Informational RFC.*

[5] Elwyn Davies, Suresh Krishnan, and Pekka Savola, "IPv6 Transition/Coexistence Security Considerations," RFC 4942, September 2007.

The transition into a dual-stack environment, while attempting to preserve the integrity of a single service regime, presents numerous security concerns. This document is a good overview of such concerns.

[6] Dan Wing and Andrew Yourtchenko, "Improving User Experience with IPv6 and SCTP," *The Internet Protocol Journal,* Volume 13, No. 3, September 2010.

Building efficient applications in a dual-stack world can be very challenging. It is often the case that poor management of a dual-stack system can make the user experience far slower than just continuing in the IPv4 world. One way to redress this problem is to exchange sequential testing of IPv6 and IPv4 connectivity into a parallel operation—both protocols at once. This article explains the concept.

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 numerous Internet-related books, and was a member of the Internet Architecture Board from 1999 until 2005; he served on the Board of Trustees of the Internet Society from 1992 until 2001. E-mail: **gih@apnic.net**

Call for Papers

The Internet Protocol Journal (IPJ) is published quarterly by Cisco Systems. The journal is not intended to promote any specific products or services, but rather is intended to serve as an informational and educational resource for engineering professionals involved in the design, development, and operation of public and private internets and intranets. The journal carries tutorial articles ("What is...?"), as well as implementation/operation articles ("How to..."). It provides readers with technology and standardization updates for all levels of the protocol stack and serves as a forum for discussion of all aspects of internetworking.

Topics include, but are not limited to:

- Access and infrastructure technologies such as: ISDN, Gigabit Ethernet, SONET, ATM, xDSL, cable, fiber optics, satellite, wireless, and dial systems
- Transport and interconnection functions such as: switching, routing, tunneling, protocol transition, multicast, and performance
- Network management, administration, and security issues, including: authentication, privacy, encryption, monitoring, firewalls, troubleshooting, and mapping
- Value-added systems and services such as: Virtual Private Networks, resource location, caching, client/server systems, distributed systems, network computing, and Quality of Service
- Application and end-user issues such as: e-mail, Web authoring, server technologies and systems, electronic commerce, and application management
- Legal, policy, and regulatory topics such as: copyright, content control, content liability, settlement charges, "modem tax," and trademark disputes in the context of internetworking

In addition to feature-length articles, IPJ contains standardization updates, overviews of leading and bleeding-edge technologies, book reviews, announcements, opinion columns, and letters to the Editor.

Cisco will pay a stipend of US$1000 for published, feature-length articles. Author guidelines are available from Ole Jacobsen, the Editor and Publisher of IPJ, reachable via e-mail at `ole@cisco.com`

The Internet Protocol Journal, Cisco Systems
170 West Tasman Drive
San Jose, CA 95134-1706
USA

ADDRESS SERVICE REQUESTED

PRSRT STD
U.S. Postage
PAID
PERMIT No. 5187
SAN JOSE, CA

The Internet Protocol Journal is published quarterly by the Chief Technology Office, Cisco Systems, Inc.
www.cisco.com
Tel: +1 408 526-4000
E-mail: ipj@cisco.com