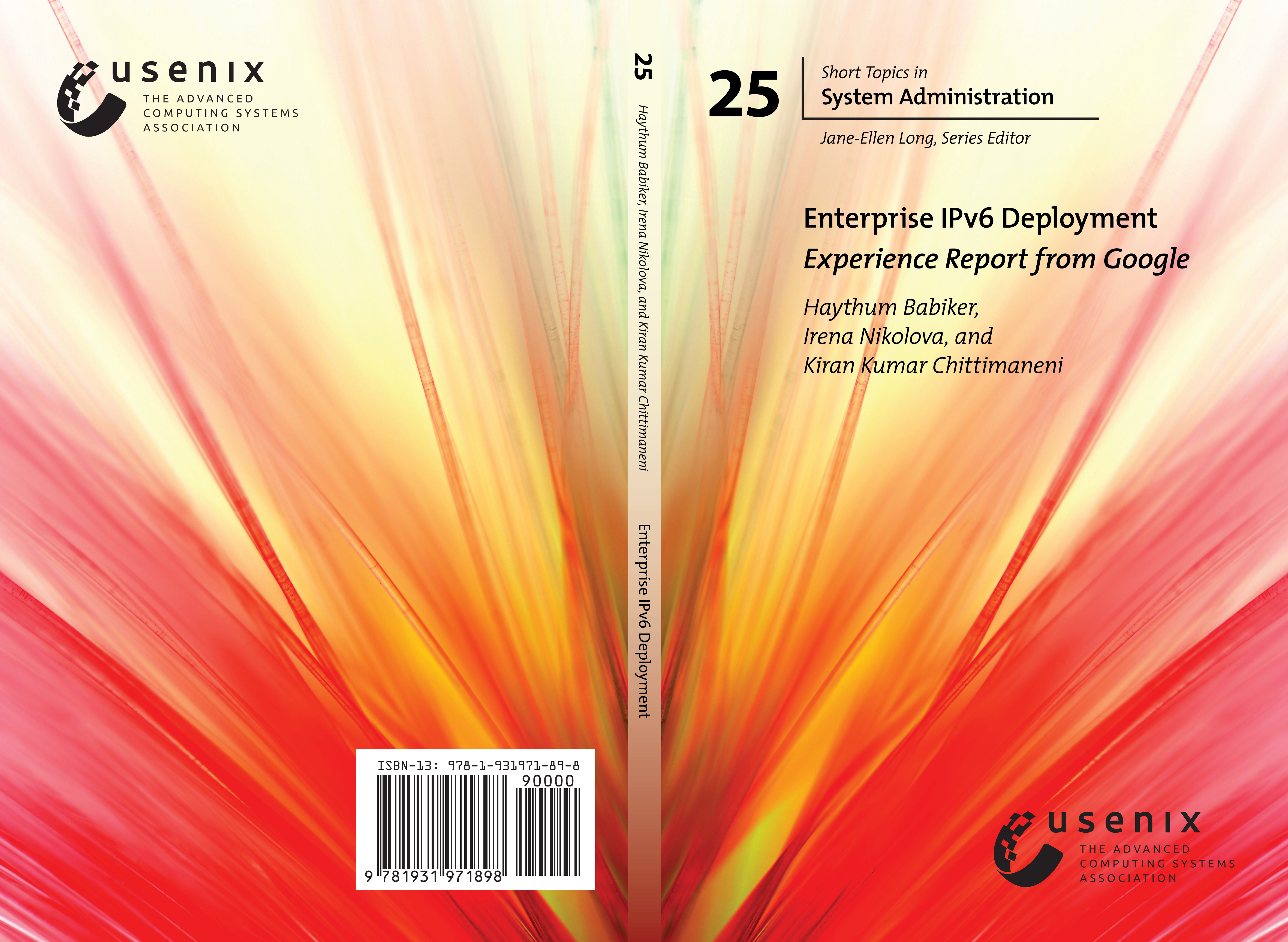

usenix
THE ADVANCED
COMPUTING SYSTEMS
ASSOCIATION

ISBN-13: 978-1-931971-89-8
90000
9 781931 971898

25 *Haythum Babiker, Irena Nikolova, and Kiran Kumar Chittimaneni* Enterprise IPv6 Deployment

25 *Short Topics in* System Administration

Jane-Ellen Long, Series Editor

Enterprise IPv6 Deployment
Experience Report from Google

Haythum Babiker,
Irena Nikolova, and
Kiran Kumar Chittimaneni

usenix
THE ADVANCED
COMPUTING SYSTEMS
ASSOCIATION

25 *Short Topics in*
System Administration

Jane-Ellen Long, Series Editor

Enterprise IPv6 Deployment Experience Report from Google

Haythum Babiker, Irena Nikolova, and

Kiran Kumar Chittimaneni

Published by the USENIX Association
2011

ISBN 978-1-931971-89-8

To purchase additional copies, see http://www.sage.org/pubs/short_topics.html.

The USENIX Association
2560 Ninth Street, Suite 215
Berkeley, CA USA 94710

http://www.usenix.org/

Contents

Figures and Tables

Foreword

IPv6 is an important technology because the Internet cannot continue to function without it. The "killer app" for IPv6 is...we get to keep using the Internet!

The biggest criticism of IPv6 is that the designers ignored certain important operational aspects. That's a fancy way of saying that converting to IPv6 requires a lot of work, and we can't remove IPv4 from our networks until after the entire world is dual-stack IPv4/IPv6. This will take years. Other possible designs might have permitted IPv4 to be decommissioned in parallel or after a small amount of IPv6 had been deployed. Alas, that is not how it turned out. It is backwards-compatible in ways that benefit users at the application level but not in ways that would have made it easy for network engineers and architects.

There are two ways to convert your enterprise network to dual-stack IPv4/IPv6.

The first method, described in this book, requires a lot of planning, negotiating, designing, testing, testing, testing, installation, and testing. It is a long voyage but worth the trip. This book will prepare you for the realities of doing this important project. Benefit from Google's experiences.

The second method is to sit back, relax, and let other people do all the work. This is the technique I used. I had no involvement in the Google IPv6 effort other than cheerleading from the sidelines. I joined an internal IPv6 mailing list. I spent months watching email fly back and forth as people debated, discussed, designed, and planned. Then came a flood of announcement after announcement: NYC will be converted tomorrow. NYC is done! Dublin will be converted tomorrow. Dublin is done! Cambridge will be converted tomorrow. Cambridge is done! This continued for months. Now IPv6 is just part of the standard way networks are deployed at Google. It seems like only yesterday the whole thing was just getting started.

My friends at other companies describe their IPv6 projects in phrases like "not yet" or "next year" or "didn't get funding." It makes me very proud that I work at a company with people like Haythum, Irena, and Kiran. I was delighted to learn that they were sharing their experience with the world so others could benefit.

Don't use the second method. Don't be like me. Be like Haythum, Irena, and Kiran. This book will show you how.

I hope you enjoy reading it as much as I did.

Thomas A. Limoncelli
Google NYC
October 2011

1. Introduction

IPv6, the next generation of the Internet Protocol, is redefining how users and machines communicate with each other across networks. The current version of the Internet Protocol, IPv4, can neither support the growing number of devices connected to the Internet nor encourage the development of new services. These concerns, among other issues, are driving the Internet to upgrade to IPv6. IPv6 provides critical performance and architectural improvements for all networks, and it promotes a freely evolving Internet.

Google's corporate network is a heterogeneous environment consisting of equipment from a host of sources, myriad technologies, and numerous internally developed custom applications and services. The Google network uses a variety of topologies and access mechanisms to provide connectivity to tens of thousands of employees distributed across offices, corporate data centers, and other locations around the world.

This book provides a detailed case study of Google's enterprise IPv6 deployment. We first provide the reader with a basic introduction to the IPv6 protocol characteristics and discuss the issues that IPv6 addresses in an enterprise organization. After showing the evolution of our IPv4-only network to dual-stack (IPv4 and IPv6) over the past several years and the underlying technology trends driving those changes, we explain how to apply these technology drivers to enable new network topologies and services that are not subject to the limitations of the older IPv4 protocol. We cover some advanced topics that provide guidance on scalable and reliable IPv6 deployment. We follow this with a detailed description of the way we implemented IPv6 throughout Google's global network in a relatively short time with a small core group. Finally, we tell about the challenges we faced during the various implementation phases and the network design we used for IPv6 connectivity.

This book is aimed at system administrators, network architects, and enterprise IT managers alike. We assume that the reader is familiar with general enterprise network architecture, requirements, and problems. For those who are not yet familiar with IPv6, we provide some IPv6 concepts and background, followed by a deep dive into IPv6 network design, planning, and deployment methodology.

We hope this book will increase awareness about the new version of the IP protocol and to provide guidance to organizations and administrators who are either planning to deploy IPv6 or are simply seeking to gain a better understanding of it.

2. The Business Case for Change

Inertia, to borrow a term from classical physics, is the resistance of a physical object to a change in its state of motion or rest, or the tendency of an object to resist any change in its motion. This fundamental principle of physics can also be broadly applied both to technology and to how we tend to embrace changing technologies.

Humans in general are not truly resistant to change; we may simply be subconsciously afraid of the side-effects of inertia. Let me illustrate with an example: Imagine a person driving a car down the path of IPv4, on cruise control. As he drives, he approaches a fork in the road. One path continues down IPv4, while the other is a brand-new path down IPv6. He hesitates, unsure of which path to take. IPv4 is familiar territory, whereas IPv6 is uncharted. He questions the benefits of going down an unknown path to get to his destination. Switching to IPv6 would entail stepping on the brakes, turning, and invoking the laws of motion—jolting him from his seat, causing the tires to squeal, perhaps going into a skid. He decides to stick to IPv4 and tells himself he'll go down IPv6 another day.

The Internet community is aware that IPv6 needs to deployed. However, there are several classes of people—CFOs, IT managers, and the like—who continue to question the benefits of doing so. Questions often asked by such people are, "Why bother transitioning to IPv6 when IP addresses are still available and IPv4 suffices?" "Why spend resources on potential hardware and software upgrades, training staff and learning multiple protocols?" "Does lack of IPv6 knowledge and implementation really spell career doom for IT professionals?"

Historically, the most compelling case for making the transition from IPv4 to IPv6 has been that IPv4 addresses are running out. Once considered just another gloomy prophecy, this one came true on February 3, 2011, with the Internet Assigned Numbers Authority's (IANA) announcement that their free pool of IPv4 addresses had dried up [1].

For many small businesses, it may be true that address exhaustion is not a compelling reason to switch to IPv6 today. However, it's simply a matter of time before they will have to migrate to IPv6 if they are to connect to the rest of the world. What is now a matter of choice will evolve into a matter of necessity.

The business case for IPv6 doesn't necessarily lie within the organization but is more likely to be found externally. Enterprises may not need IPv6 in their networks today to send email, store and retrieve data, or access content on the Internet. The need will very likely arise from the organization's Business-to-Consumer (B2C) and Business-to-Business (B2B) applications.

Service or content providers usually have a relatively clear business case: their customers demand IPv6. These providers must either evolve or perish in a field where users are fickle and competition is fierce. Financial growth and well-being are at stake here, and no company would like to miss out on any IPv6-related revenue.

For other organizations, demand is likely to arise from the need to connect to external businesses (partners, vendors, etc.) through custom application interfaces that will compel them to use IPv6. They therefore need to be prepared for that eventuality.

It is also possible that at some point service providers will start charging a premium for the scarce IPv4 addresses. This would directly impact every IT organization's budget. Effectively, then, failing to deploy IPv6 only postpones the inevitable at the risk of increased costs in the future.

The Business Case at Google

There is a much more subtle, yet powerful business case for IPv6 which led us, at Google, to adopt IPv6 early. It can be summed up in one word: innovation. The process of innovation forces us to experiment, take risks, and most importantly, create. It not only allows us to create a new, unique entity but also provides a way for diverse component technologies to come together and form new ecosystems. Let us now take a closer look at some of the motivations for deploying IPv6 in Google's Enterprise Network.

Google strongly believes that IPv6 will enable innovation and allow for the Internet's continued growth. Our company is committed to providing universal access to information for all users, regardless of whether they connect using IPv4 or IPv6. A strong culture of innovation allows us to be at the leading edge of the technology curve. Early adoption of technology also allows us to build for the future. To borrow a quote from ice hockey legend Gretzky, "You want to skate to where the puck is going to be and not where the puck was." That maxim applies quite well in the context of IPv6.

Google engineers are constantly working on new products and features. Before being publicly released, every product goes through a process called "dogfooding," whereby we become consumers of our own products internally. This allows us to experience firsthand how our products work for the end user. So, as various teams marched toward dual-stacking products such as Gmail and YouTube, we had a real need to provide our internal users with a network that was IPv6-ready, so that they could help develop, test, and dogfood these products. Dogfooding allows us to "launch early and iterate often," a principle well known in software engineering and equally applicable to the realm of network engineering. Launching early allows us to shake out bugs in hardware and software; iterating often allows us to mature our designs over time.

Innovation and dogfooding aside, there was a very important technical driver for adopting IPv6 internally: we were running out of private RFC 1918 addresses. The same reasons that caused global IPv4 address space consumption impacted us internally: inefficient address use, explosion of mobile devices, and virtualization, to name a few. The only long-term way to sustain growth was to implement IPv6.

We also wanted to break out of a chicken-or-egg problem. Internet service providers (ISPs) would attribute the slow rate of IPv6 adoption to the lack of content, while content providers would attribute the slow rate of IPv6 adoption to the lack of IPv6 access by end users. To help make a positive contribution to the Internet community, we knew we had to enable IPv6 access to Google engineers to help them launch IPv6-ready products and services.

At the time that we started planning and deploying IPv6 at Google, address pool exhaustion was at least another 3–4 years away, but we saw the writing on the wall. It wasn't a question of *if*, but of *when*, we would run out of IPv4 addresses, both public and private. It seemed to us that IPv6 was the only solution that made sense in the long run. In a certain sense, we also adopted the motto of "If you build it, they will come."

Think Big, Start Small

Everything is not always about "Business." It is very easy to get bogged down in presenting a business case in an attempt to try and convince other people, management and peers alike, of the virtues and benefits of IPv6. Sometimes the business case for change is no more than a bold vision and people who embrace that vision.

People
"20% Project"
Vision
"IPv6 Everywhere"

Figure 1. Think Big, Start Small

At Google we started out with a simple, yet grand, vision—IPv6 everywhere. The people who would embrace this vision came by virtue of Google's wonderful "20% Time" program. The program at Google gives an engineer the freedom to work on something they're really passionate about. The chosen project need not fit within the scope of the engineer's job description—it can be anything! Such creative and intellectual freedom has led to a profusion of innovations at Google. Gmail, Google Autocomplete,and Google News, for example, all sprang from 20% Time projects. Likewise, IPv6 on the enterprise network started as a 20% Time project, which allowed us to "Think Big, Start Small." We iterated early and often, building IPv6 into a strong component of Google's networking infrastructure.

3. From IPv4 to IPv6

The world has come a long way since the Internet consisted of a few hundred nodes, sparsely dispersed around the world. These nodes communicated with each other using a protocol known as the Internet Protocol. The original creators of the current version of Internet Protocol version 4, also popularly known as IPv4, anticipated a widespread proliferation of computing devices requiring IP addresses. IPv4, with its 32 bits of binary real estate, provides a maximum of 2^{32}, or approximately 4.3 billion, Internet addresses, undoubtedly a very large number. However, many address allocation policies and design choices, coupled with the burgeoning demand, led to a situation in which the pool of these IP addresses would eventually be exhausted, thus necessitating the creation of IPv6.

IPv4 History and Current State

On January 31, 2011, the Internet Assigned Numbers Authority (IANA) allocated two blocks of IPv4 address space to the Asia Pacific Network Information Center (APNIC), the Regional Internet Registry (RIR) for the Asia Pacific region. This in turn triggered a global policy, allocating the remaining IANA pool of five /8 ("slash-8") networks equally among the five RIRs. On February 3, 2011, the world witnessed a watershed moment, as IANA allocated those blocks and announced that the free pool of available IPv4 addresses was fully depleted [1]. This means that there are no more IPv4 addresses available for allocation from the IANA to the five RIRs.

The question is, How did the Internet community get here? Let us take a closer look at the circumstances that shaped the consumption and eventual depletion of IPv4 addresses.

Inefficient Address Allocation and Usage

When IPv4 was first created, only Class A networks were being allocated. The first 8 bits indicated the network part and the remaining 24 bits indicated the local part. In 1978, early Internet architects recognized that this would serve only 256 networks and therefore suggested a hierarchical addressing structure that would allow more networks to be connected [2].

By the early 1980s, the *Classful Addressing* system was created. In this system, addresses were assigned based on classful boundaries, such as Class A, B, or C. Each Class A network would provide over 16 million IP addresses, each Class B would offer over 65 thousand IP addresses, and each Class C over 250 IP addresses. Organizations were to apply for a given class of addresses based on their self-defined needs. This system of address allocation was implemented for another twelve years.

Furthermore, to help conserve IP space without giving up on the goal of aggregation, subnetting was introduced. Subnetting provided an additional level of hierarchy: it inserted a subnet section into the IP address, between the network and local sections. By the early 1990s, however, it became clear that classful addressing coupled with subnetting could no longer meet the demands of the rapidly growing Internet. To help address this, a solution known as *Classless Inter-Domain Routing* (CIDR) was created to extend subnetting beyond the local organization and introduce a new IP addressing scheme that would allow flexibility in defining the network and host bits.

When IPv4 was first created, addresses were assigned based on classful boundaries, such as Class A, B, and C. Each *Class A* network would provide for over 16 million IP addresses, *Class B* over 65 thousand IP addresses and a *Class C* over 250 IP addresses. Organizations applied for a certain class of addresses based on their purported needs. This system of address allocation was highly inefficient by virtue of its lack of flexibility in assigning addresses based on need as opposed to arbitrary sizes. For example, an organization requiring 66 thousand IP addresses (~500 IPs above the Class B limit) would be assigned a Class A network as there was nothing in between. This effectively wasted over 15 million IP addresses, all because of a policy decision to allocate three arbitrarily sized classes.

Additionally, the process of subnetting causes several IP addresses within a given subnet to be unallocatable. For example, the rules of subnetting state that the first IP of a subnet cannot be used by a host since it is the Network Address. Similarly, the last IP of a given subnet cannot be used since it is reserved as the broadcast address. So for every subnet, the network administrators are unable to assign at least two IP addresses to their hosts. For a large enterprise, those numbers add up very quickly.

Smart Phones and Network Aware Devices

Smart phones have revolutionized the world of mobile communications and computing and are becoming ubiquitous. Never before in history has so much computing power been literally in the hands of the mainstream public. In certain societies, cell phones and smart phones are having a revolutionary leapfrogging effect. Entire segments of populations in places such as Africa and India are going straight from paper mail to the world of SMS, instant messaging, and beyond. All of these devices need IP addresses. This recent, rapid adoption of mobile devices has done more than its fair share in hastening the inevitable IPv4 address exhaustion.

In addition to phones, there has been a significant uptick in network-aware devices such as video game consoles, TVs with WiFi connections, and smart refrigerators and cars, all consuming IP addresses at an alarming rate. This trend is growing geometrically.

Virtualization and Cloud Computing

Finally, the advent of virtualization and cloud computing has increased the need for IP addresses by two or three orders of magnitude. For each physical machine there can be many virtual hosts, each requiring a unique IP address. Furthermore, most business applications require that each host's IP address be both static and always available.

Addressing the Problem

The impending exhaustion of IPv4 address space has been a concern since the early 1990s. Even at that time, this issue was taken seriously by the Internet Engineering Task Force (IETF), which started several parallel efforts not only to resolve address limitations but also to work toward providing additional functionality:

1. In early 1993, RFCs 1518 and 1519 attempted to tackle the issue of Classful Addressing by defining and introducing Classless Inter-Domain Routing (CIDR).
2. RFC 1631, in May 1994, also attempted to address the IPv4 address exhaustion issue by creating Network Address Translation (NAT).
3. The IETF formed the IP Next Generation (IPng) Area in late 1993 to investigate the various IPng proposals that were solicited via RFC 1550 and to offer recommendations on how to proceed. After much review, the IETF in 1995 recommended the IP Next Generation Protocol through RFC 1752.

By 1996, a series of RFCs, starting with RFC 1883, were issued detailing the new IPv6 protocol.

Overall, given the growth of Internet usage, explosion of intelligent devices, and emergence of new business applications, there seems to be an insatiable appetite for IP addresses. At this point, with the IPv4 pool exhausted, deploying IPv6 is a requirement, not an option.

4. IPv6: A Primer

IPv6 gives us far more IP addresses than its predecessor. IPv6 addresses have 128 bits, far greater than the 32 bits provided by IPv4. IPv6 can therefore provide a theoretical maximum of 2^{128} or approximately 340 undecillion addresses. This is truly an astonishing number and might conceivably provide enough IP addresses for the foreseeable future.

Address Notation

One of the first points that strikes veterans of IPv4 who are new to IPv6 is the address format. It is long and unfamiliar. Simple decimal notation no longer suffices to deal with IP addresses; hexadecimal numbers are required. Periods have given way to colons. Notation aside, though, addresses are still just binary digits strung together. Under the hood, they look almost the same as IPv4 addresses, just four times longer.

Here is an example of a typical IPv6 address:

2001:0DB8:0000:0000:ABCD:EF12:3456:7890

An IPv6 address is represented in text by converting the 128 bits into eight 16-bit hexadecimal blocks separated by colons. Designers of this IP address schema realized that address formats could be inconveniently long, and so they provided some short cuts that could be interpreted both by humans and by machines. For example, the address above can be written as follows:

2001:DB8::ABCD:EF12:3456:7890

A couple of things should be noted here. First, the leading zero in the second block of hexadecimal digits, "0DB8," is gone, reducing it to "DB8." It is no longer necessary to write the leading zeros in an individual 16-bit hexadecimal block.

Second, all those zeros in the third and fourth hex blocks have shrunk down to two colons (::). The use of the double colon indicates one or more groups of 16 bits of zeros and can be used to compress leading or trailing zeros in an address. The double colon can only appear once in an address. This restriction ensures that computers can programmatically add as many zeros into the IP address as needed to convert this textual representation into the correct binary 128-bit address.

In reality, network administrators are more likely to encounter an IPv6 address that looks something like this:

2001:DB8:0:1000::1395:26AF/96

This representation is similar to the CIDR notation familiar from IPv4, i.e., IP_Address/Prefix_Length

Please note that the prefix length is always denoted in decimal. As in IPv4, the prefix length tells us how many of the leftmost contiguous bits of the address compose the prefix. For example, the node address, from the instance on p. 11, is 2001:DB8:0:1000::1395:26AF and its subnet number is 2001:DB8:0:1000:0:0::/96.

To recap what we have seen so far, that IPv6 address can be written in the following equivalent formats::

2001:DB8:0:1000::1395:26AF/96

2001:DB8:0000:1000:0000:0000:1395:26AF/96

2001:DB8::1000:0000:0000:1395:26AF/96

2001:DB8::1000:0:0:1395:26AF/96

Unlike IPv4, in IPv6 any field can legally contain all zeros or all ones, unless specifically excluded (we give more details in Chapter 9). This flexibility eliminates one of the inefficiencies in IPv4 with respect to address usage.

Address Types

There are only three types of addresses in IPv6, as opposed to the four that could be found in IPv4:

Unicast

Multicast

Anycast

In IPv6, broadcast addresses no longer exist. Instead, IPv6 uses multicast addresses—a welcome change for many who have had to deal with broadcast issues in the past.

The definitions of the three address types mentioned above haven't really changed from IPv4. The type of an IPv6 address is identified by the high-order bits of the address, as follows:

Address Type	Binary Prefix	IPv6 Notation
Unspecified	00...0 (128 bits)	::/128
Loopback	00...1 (128 bits)	::1/128
Multicast	11111111	FF00::/8
Link-Local Unicast	1111111010	FE80::/10
Global Unicast	(everything else)	(everything else)

Table 1. IPv6 Address Type Identifiers (RFC 4291)

Loopback and multicast address definitions should be familiar to the reader from IPv4. Let's take a closer look at some of the lesser known ones here.

Unspecified Address

An address with all zeros is called the "unspecified address." Like 0.0.0.0/32 (RFC 3330) in IPv4 networks, the unspecified address can be used in the source address field of any IPv6 packets sent by an initializing host before it has learned its own address during the boot process. Other rules that apply here are:

> The unspecified address must never be assigned to any node.
>
> It must not be used as the destination address of v6 packets or routing headers.
>
> A packet with a source address unspecified must never be forwarded by an IPv6 router.

Link-Local Unicast

These addresses are used on a single link for purposes such as automatic address configuration, neighbor discovery, or when no routers are present. All interfaces are required to have at least one link-local unicast address. As the name suggests, these addresses are not routable, and routers will drop packets with a link-local address in either the source or the destination IP address field.

Global Unicast

As the name suggests, global unicast addresses are addresses that hosts on a network use to connect to nodes outside their local network.

Enhancements over IPv4

Although one of the primary reasons for creating IPv6 was to respond to IPv4 address exhaustion issues, other important enhancements were built into the protocol to make it more useful than its predecessor.

Stateless Auto Address Configuration (SLAAC)

One of the salient features of IPv6 is its ability to auto-configure an IPv6 address for a given interface. SLAAC is especially important for situations like a home network, full of network-aware devices, which are now able to connect to one another and to the rest of the world without the need for a DHCP server. Of course, this flexibility is beneficial for any network administrator who is interested in the ability of the host to have a unique, routable IPv6 address and is willing to give up control over the exact addresses the hosts use. The auto-configuration process provides an interface with a link-local address and a global address via a combination of locally available information and router advertisements, along with verification steps to ensure uniqueness of the address on a link. An important side benefit of SLAAC is that it makes renumbering a breeze compared to the process in IPv4 networks. Let us take a brief look at how link-local and global addresses are formed in the auto-configuration process.

Link-Local Addresses are formed by appending the link-local prefix FE80, as shown in Table 1, to the interface identifier. Interface IDs are required to be 64 bits long and to be constructed in Modified EUI-64 format. Here is an example of how the interface ID for a link or node could be obtained from an IEEE 802 48-bit MAC address.

Let's take the MAC address for the interface:

58:B0:35:7C:62:24

The next step is to obtain an EUI-64 identifier from this MAC address. In order to create that, two octets are inserted, with hexadecimal values of 0xFF and 0xFE, in the middle of the 48-bit MAC. So the EUI-64 identifier for the MAC address given above will be:

58:B0:35:FF:FE:7C:62:64

The final step is to modify the EUI-64 identifier into an interface identifier by inverting the "u" universal/local bit. The "u" bit is the second low-order bit of 0x58 (the first byte) of the MAC address above—in other words, the seventh bit of an EUI-64 identifier when written in the binary format, going left to right:

0x58 = 010110**0**0 ('u' bit highlighted)

010110**1**0 ('u' bit inverted) = 0x5A

Finally when everything is put together, the interface identifier is:

5A:B0:35:FF:FE:7C:62:64

And the link-local address is:

FE80::5AB0:35FF:FE7C:6264

RFC 4291 describes the process of deriving the interface identifier in much more detail.

Global Addresses are formed in a similar fashion, by appending the interface identifier to the prefix information obtained via router advertisements. A key point to note here is that the sum of the prefix length and the interface identifier length must equal 128 bits; otherwise, the prefix information option in the router advertisement has to be ignored. In other words, SLAAC works only when the advertised prefix length is a /64. Imagine a host interface receiving the following prefix option from the router advertisement:

2001:DB8:0:1000::/64

Using the MAC address from the example in the section above, the SLAAC address for that interface would be:

2001:DB8:0:1000:5AB0:35FF:FE7C:6264

or

2001:DB8::1000:5AB0:35FF:FE7C:6264

RFC 4862 describes the process of stateless auto-configuration in much more detail.

Header Format Simplification

One of the main goals of the designers of IPv6 was to optimize the protocol to allow devices to process packets much more efficiently. One optimization was to significantly simplify the header format of IPv6 packets. For starters, the header size has a fixed length of 40 bytes, whereas IPv4 uses a variable length of 20 to 60 bytes. This optimization alone significantly simplifies future router architectures and cuts down on the processing cost of packet handling, as well as limiting the bandwidth costs.

Improved Support for Extensions and Options

The fixed size of the header doesn't keep other information from being encoded into the IP header. In IPv6, optional information is transmitted using "extension headers." These are additional headers that are inserted between the IPv6 header and the upper-layer header in a packet. An IPv6 packet may carry zero, one, or many extension headers, each identified by a unique Next Header value. The first three headers listed below are specified in RFC 2460 and the last two are specified in RFC 2402 and 2406, respectively. (It is important to note that the last two headers are related to IPSEC.)

Hop-by-Hop Options: The hop-by-hop options header carries optional information that must be examined by every node along a packet's delivery path. The hop-by-hop options header may only appear immediately after an IPv6 header.

Fragment: The fragment header is used by an IPv6 source to send a packet too large for the path MTU to its destination. Unlike in IPv4, fragmentation occurs only at the source that is sending the packet; that is, routers do not fragment packets anymore, unless they are the source of those packets. All fragmentation is handled by the source and reassembled by the destination. IPv6 stipulates that every link in the Internet must have an MTU of 1280 bytes or greater. IPv6 hosts perform path MTU discovery (PMTUD) in order to discover path MTU and leverage MTUs greater than the minimum defined.

Destination Options: The destination options header is used to carry optional information that need be examined only by a packet's destination.

Authentication: The authentication header is used to provide connectionless integrity and data origin authentication for IP datagrams.

Encapsulating Security Payload (ESP): The encapsulating security payload header is used to provide confidentiality, data origin authentication, connectionless integrity, an anti-replay service (a form of partial sequence integrity), and limited traffic flow confidentiality.

Flow Labeling Capacity

A new capability was added to enable the labeling of packets belonging to particular traffic flows for which the sender requests special handling, such as non-default quality of service or real-time service. In addition, several other interesting uses for the flow label are actively being discussed in the IETF 6man working group.

Summary

As you can tell, there are aspects of IPv4 that IPv6 builds upon and there are some that are entirely new. IPv6 was designed to allow devices to process packets much more efficiently, in addition to providing new features and functionality such as SLAAC. The power of IPv6 does not simply lie in its extended address space, but also in the flexibility that the new header schema provides.

5. IPv6 Address Policy and Planning

Previous chapters pointed out that inefficient macro-level address allocation policies played a major role in the eventual depletion of IPv4 addresses. Many of these inefficient practices also found their way to the micro (enterprise) level, causing large companies and organizations to run out of private IPv4 addresses internally. If there is anything to be gleaned from the IPv4 address exhaustion issue, it is that having a solid IP address allocation policy is crucial and that such a policy must be carefully planned and executed. Clearly, IP addresses, regardless of version number, are ultimately a finite resource. Like any other resource, it is up to the society that uses it to decide whether it could be sustained for a longer time. Let us take a broad look at how IPv6 address allocation policies work.

The Internet Registry System

The Internet Registry (IR) system was established to ensure conservation, routability, and registration of Internet address space [3]. *Conservation* refers to maximizing the lifetime of the address space allocated. *Routability* refers to the distribution of IPv6 space in a hierarchical manner, permitting the scalability of addresses. *Registration* refers to documentation of address space allocation and assignment in order to ensure uniqueness and to aid in troubleshooting at all levels.

The IR consists of a hierarchical architecture, shown in Figure 2:

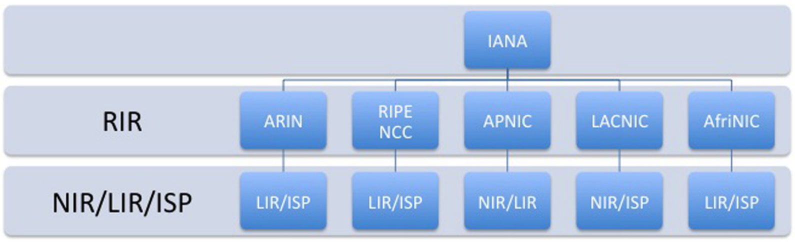

Figure 2. IR Hierarchy

The *Internet Assigned Numbers Authority (IANA)* has authority over all number spaces used in the Internet. IANA allocates parts of the Internet address space to regional IRs according to their established needs.

The *Regional Internet Registry (RIR)* is, as the name suggests, an IR that operates in a large geopolitical region. Currently there are five RIRs, in North America, Europe, Asia, Africa, and Latin America (see Table 2, next page).

RIR	Jurisdiction
AfriNIC	Africa
ARIN	Canada, many Caribbean and North Atlantic Islands, and the United States
LACNIC	South America and the Caribbean
APNIC	Asia Pacific
RIPE NCC	Europe, Central Asia, and the Middle East

Table 2. RIRs with Their Jurisdictions

The *National Internet Registry (NIR)* plays a role similar to that of an RIR but is limited to operation within the borders of a particular country or economy, e.g., the China Network Information Center (CNNIC), or the Japan Network Information Center (JPNIC).

Local Internet Registries (LIRs) are bodies established under the authority of the RIR and IANA. LIRs are typically Internet service providers (ISPs) who assign IPs to end users or possibly to smaller ISPs. Large enterprises can also operate as LIRs in some cases.

Macro IPv6 Address Policy

At the top of the IR hierarchy seen earlier, IANA manages the macro IPv6 space. It allocates this space to regional registries (RIRs), which then allocate space to service providers within their respective jurisdictions. The IPv6 unicast space consists of the entire IPv6 address range except for FF00::/8, which is reserved for multicast addresses. As referenced in RFC 4291, IANA global unicast IPv6 address assignments are currently limited to be made from the range of 2000::/3 [4]. However, no assumptions about 2000::/3 being special should be made. It is likely that, in the future, currently unassigned portions of the IPv6 address space will be assigned to global unicast. All assignments made from this block are registered in the public registry. The IANA allocates sufficient IPv6 address space to the RIRs to support their registration needs for at least eighteen months. Furthermore, it allows each RIR to apply whatever allocation and reservation strategies it feels will ensure the efficiency and efficacy of its work [5].

Typically RIRs allocate a /32 or longer to organizations such as ISPs and LIRs that both assign addresses to other organizations and also provide IPv6 Internet connectivity to end users. Such allocations are subject to the ISPs and LIRs being able to meet any one of several criteria set by the RIRs. For example, ARIN sets the following criteria for assigning an initial allocation to ISPs [6]:

- ❖ Having a previously justified IPv4 ISP allocation from ARIN or one of its predecessor registries
- ❖ Currently being IPv6 multihomed or immediately becoming IPv6 multihomed and using an assigned valid global AS number
- ❖ Providing a reasonable plan detailing assignments to other organizations or customers for one, two, and five year periods, with a minimum of 50 assignments within 5 years

Subsequent allocations are usually provided based on certain utilization thresholds being met by the ISP or LIR.

Each RIR provides guidelines to ISPs and LIRs for making assignments to end-user sites. For instance, ARIN provides the following guidelines to ISPs and LIRs when making assignments to end-user sites:

- /64 when it is known that one and only one subnet is needed
- /56 for small sites, that is, those expected to need only a few subnets over the next five years, such as small businesses or home networks that will need more subnets in the future
- /48 for larger sites, e.g., enterprise sites

RIRs are being particularly attentive about route aggregation, in an attempt to limit the expansion of Internet routing tables. Wherever possible, attempts are being made to distribute address space in a hierarchical manner.

IPv6 Address Planning

If there is any single aspect of IPv6 deployment planning in which an enterprise should invest most of its time, this would be it. The enterprise IPv6 deployment team at Google certainly did so. For one thing, we didn't want to renumber anything later, and we wanted to make sure we had a robust and scalable address plan. IP address assignment at the enterprise level should also endeavor to follow the goals of conservation, routability, and registration, just as IANA and RIRs do at the macro level.

Many factors influence IPv6 address planning, e.g., geographical spread of locations, number of locations, address size requirements at those sites, multihoming requirements, and internal routing architecture. Figure 3 shows Google's geographic spread of office locations.

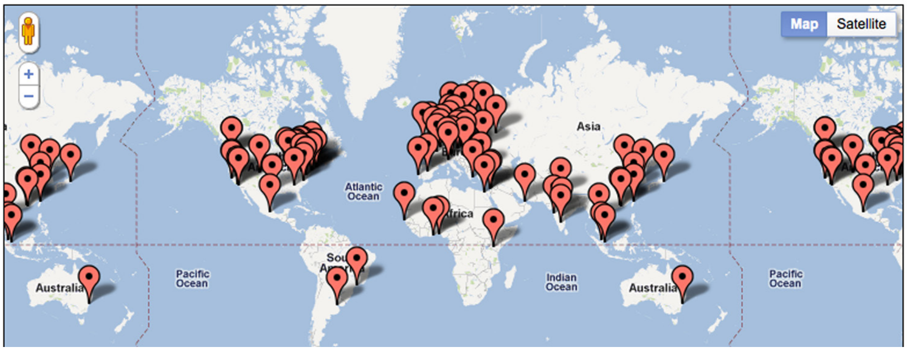

Figure 3. Google Offices

Google's enterprise network is a distributed network that supports 69 Google offices spread across 36 countries around the world. These offices are usually multihomed and vary in size depending on business function (sales, engineering, etc.). Multihoming here refers to the office having two diverse, redundant paths out to either the Internet or our corporate backbone.

Multihoming and PI Space

In general, multihoming considerations greatly affect IPv6 address planning efforts. For example, if an organization has a site that has only one link to the rest of the world, then they will most likely be satisfied with whatever IPv6 address space their ISP provides. However, if the site has more than one connection to the rest of the world, and the organization has the business requirement to be reachable from the outside world through a well-known IP that is not provider-dependent, then they might be interested in obtaining provider-independent (PI) space from an RIR. Google's requirements fell into this latter category, so we, the network engineering team, went ahead and applied for PI space from various RIRs as needed.

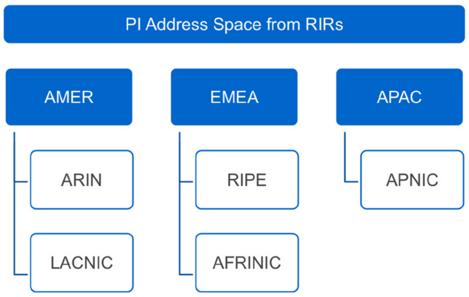

Figure 4. Corporate Network Geographic Regions to RIR Mapping

Figure 4 shows the mapping between high-level business segments and the RIRs that fall under their purview. It is fairly common for multi-national organizations to classify their sites into three major business organizational segments loosely based on geographic regions, namely, AMER (the Americas), EMEA (Europe, the Middle East, and Africa), and APAC (Asia Pacific). This demarcation lends itself quite well to regional aggregation, but it is not based entirely on RIR boundaries.

The process of obtaining PI space from various RIRs is fairly consistent and straightforward. If the enterprise location is already multihomed with IPv4 and has PI space along with a valid global AS number, then the enterprise probably can demonstrate that they qualify for IPv6 PI space, since enterprises typically want to have design and operational parity between IPv4 and IPv6. In other cases, the process involves providing valid justification, along with detailed plans describing how they would use PI space within their network. This process typically involves providing the RIR with the number of enterprise sites, along with network documentation outlining their design.

One of the most important activities an organization should engage in, after obtaining PI space, is coming up with an effective address allocation plan that strikes the best balance between conservation and routability. For instance, within an enterprise there might be a small building in a campus that would need only a few dozen /64s but, in order to scale the routing architecture, the network engineering team might choose to assign a /56 to this building.

It is also important for everyone to understand the difference between *allocation* and *assignment* at this point. *Allocation* is the process whereby a network engineering team in an enterprise carves up their IP space for sub-delegation. For example, within a PI block, space for geographic areas East, West, North, and South could be carved out. Within these broad geographic areas further allocation can happen based on sub-types of sites such as campuses, branch offices, etc. *Assignment* is the process whereby an organization sub-delegates address space for the end user or entity. For example, a /60 was *assigned* to Engineering, a /62 to Sales, etc. from the /56 that was *allocated* to the Dallas branch office.

Sample Address Plan

To better illustrate address planning, let us look at a hypothetical situation involving Company XYZ. XYZ has four business regions: East, West, North, and South. Each region has a combination of large and small sites, some that require multihoming with external ISPs and some that connect exclusively to an internal private network. Under the current IPv4 regime, XYZ aggregates its routes in accordance with its regional business boundaries when possible. Based on their current demand and future anticipated growth, Company XYZ was recently allocated a /40 PI block by an RIR—2001:DB8::/40—thus allowing for a maximum of 256 sites that could be allocated a /48 each.

Theoretically, this means that there are 8 bits remaining (2001:0DB8:00**<u>00</u>**::/40), whose significance can be defined by XYZ, if it chooses to assign /48s exclusively to all of its sites.

One of the goals of address planning should be to maximize summarization in order to minimize advertisements and allow for controlled centralized peering. Therefore, instead of linearly allocating each site a /48, it is almost always better to carve up this space in a way that is consistent with regional and routing boundaries. For instance, the network architect for XYZ may decide to use the first two of the leftmost bits to break the /40 into East, West, North, and South, with each getting a /42:

2001:db8:00::/42 for East

2001:db8:40::/42 for West

2001:db8:80::/42 for North

2001:db8:C0::/42 for South

These /42s can now be announced centrally from each region, thus maximizing summarization.

Now, within each region, they can choose to further sub-delegate address space based on connectivity type. Given that there is a mix of office types—some small, some large, some multihomed while others connect exclusively to the private internal network—there may be an opportunity to conserve some space for future growth without losing contiguity. A good strategy here would be to assign the larger sites /48s linearly from the top of the pool, while reserving the last /48 to allocate /56s for smaller sites that are connected to the private network. This strategy allows an enterprise to keep the middle of their allocation free for future contiguous /48 assignments and allows aggregation of smaller sites into one /48 at the regional level, which could then be advertised to the Internet. Subnets of /56 blocks are not accepted in the global routing table.

The table below shows the address allocation for Company XYZ from a very high level.

Region	Aggregate	Site-Specific Prefix
East	**2001:DB8:0::/42**	Large Site 1: 2001:DB8:0::/48
		Large Site 2: 2001:DB8:1::/48
		...
		...
		Office 1: 2001:DB8:3F::/56
		Office 2: 2001:DB8:3F:0100::/56
		...
...		
South	**2001:DB8:40::/42**	Large Site 1: 2001:DB8:40::/48
		Large Site 2: 2001:DB8:41::/48
		...
		...
		Office 1: 2001:DB8:7F::/56
		Office 2: 2001:DB8:7F:0100::/56
		...

Table 3. Company XYZ Address Allocation Table

Addressing Plan at Google: The 30K Feet View

The address plan at Google follows many of the principles and strategies described in this chapter. The actual plan itself is so large and complex, it could fill a book by itself. The drawing below shows our addressing plan in a typical campus.

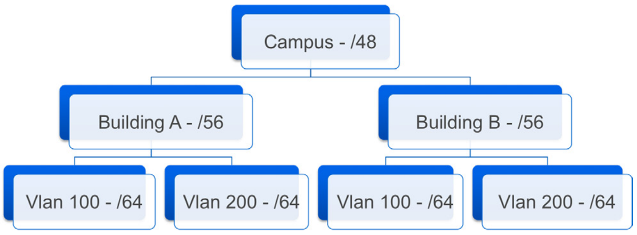

Figure 5. Campus-Level Addressing Plan at Google

At Google, a typical campus consists of several buildings. Each building has the usual access, distribution, and core layers serving various wired and wireless hosts. From a given PI block, we assign /48s to our campus networks, followed by /56s to each building on the campus. Each /56 is then carved into several /64 networks, depending on the requirements of that particular building. Usage of SLAAC, which we have standardized upon, mandates the use of /64 prefixes for each subnet.

Of course, policies and practices are a moving target. No matter how much effort you put into it, an unexpected new application or policy change could demolish your best-laid plans.

6. IPv6 Advanced Topics

IPv6 does not exist as a single entity. Rather, it is an amalgamation of several component technologies that coalesce to build a protocol suite. Let us explore some of these constituent technologies and examine how they provide key features and functionality for IPv6.

ICMPv6

IPv6 uses an enhanced version of the Internet Control Message Protocol (ICMP), ICMPv6. The ICMPv6 protocol is used by IPv6 nodes to report errors encountered upon processing IPv6 packets and to perform diagnostics and other network layer functions such as neighbor discovery [7]. Table 4 lists well-known ICMPv6 messages.

ICMP Message Type	Name	Common Codes
1	Destination Unreachable	0 = no route; 1 = admin prohibit; 2 = not a neighbor; 3 = network unreachable; 4 = port unreachable
2	Packet Too Big	0
3	Time Exceeded	0 = hop limit exceeded; 1 = fragment time exceeded
4	Parameter Problem	0 = erroneous header field; 1 = unrecognized next header type; 2 = unrecognized ipv6 option
128	Echo Request	
129	Echo Reply	

Table 4. ICMP Message Types

Additionally, IPv6 defines multiple ICMPv6 messages used in the IPv6 neighbor discovery protocol, which will be covered in the next section. As with IPv4, ICMP message types Echo Request and Echo Reply are used to perform network diagnostics via tools such as ping. ICMPv6 Packet Too Big is used for path Maximum Transmission Unit discovery [8].

ICMPv6 is an integral part of IPv6 protocol suites and is essential for the operation of the IPv6 protocol, particularly for the neighbor discovery functionality to be covered in the next section.

For more details about ICMPv6 messages format see RFC 4443 [9] and RFC 4620 [10].

IPv6 Neighbor Discovery

A key feature of IPv6 is its ability to provide plug-and-play integration of a node into the local network. Neighbor discovery (ND) is the protocol that was designed to provide local links discovery and announcements. One of the key points to remember is that ND replaces and enhances the function of the Address Resolution Protocol (ARP) in IPv4.

Here is a list of the functions provided by neighbor discovery:

- Router Discovery: Enables a host to discover routers on the local link without using DHCP.
- Prefix Discovery: Enables a host to discover prefix(es) assigned on the local link.
- Parameter Discovery: Enables the discovery of various parameters such as Maximum Transmission Unit (MTU), maximum hop limits for outgoing traffic, DNS nameservers, and search path [11].
- Address Auto-configuration: Enables a host to automatically configure an IPv6 address, i.e., Stateless Address Auto-Configuration (SLAAC).
- Address Resolution: Discovers the link-layer address for a destination on the same link.
- Next-Hop Determination: Detects the next-hop link-layer address for a destination address.
- Neighbor Unreachability Detection: Detects when a host is no longer reachable on the local link.
- Duplicate Address Detection: Detects if a given address is used by another host on the link, before allowing its use.
- Router Redirect: Provides a host with an alternative first hop for a particular destination.

All neighbor discovery functions and applications are limited to the local link, which was forced by ignoring any ND message that has a hop limit not equal to 255. ND messages use link-local (FE80::/10) and multicast addresses for all messages.

Neighbor Discovery Messages

The neighbor discovery protocol defines five ICMPv6 messages that enable communication between nodes on the local link:

- Router Advertisement: Used by routers to advertise themselves on local links and provide other parameters to hosts on the link. These announcements also contain information such as the prefix to be used on the link, maximum transmission unit, and maximum hops count.

- ❖ Router Solicitation: Used by hosts on a link to request a router advertisement.
- ❖ Neighbor Solicitation: Used by hosts to determine the link-local address of a neighbor. Hosts also use this message for duplicate address detection.
- ❖ Neighbor Discovery: Used by hosts to respond to neighbor solicitation messages or to announce an address change to neighbors on the local link.
- ❖ Redirect: Used to inform a host of a better next-hop for a particular destination.

For more details of the format of the neighbor discovery messages, see RFC 4861 [7].

Comparison of IPv4 ARP and IPv6 NDP

Feature	IPv4 ARP	IPv6 NDP
Link Local Address Resolution	Available by default	Available by default
Router Discovery	Not available: requires IRDP [12]	Available and provides link-local address
Prefix Detection	Not available: requires DHCP	Available and supports multiple prefixes on the same link
Communication Method	Broadcast: adds CPU overhead on every device on the link	Multicast
MTU and Maximum Hop Limit	Not available: requires PMTUD	Available
Address Autoconfiguration	Not available: requires DHCP	Available
Neighbor Unreachability	Not available	Available
Unidirectional Failure	Not available: requires higher-level protocol	Available

Table 5. Comparison of IPv4 ARP and IPv6 NDP

Neighbor Address Resolution

As was mentioned earlier, ND replaces and enhances the function of Address Resolution Protocol (ARP) in IPv4. To resolve the link-local address of a neighbor, a node on the link sends a neighbor solicitation message using the solicited node multicast address of the neighbor. In response, the neighbor node responds with a neighbor advertisement message. The original requestor adds the received neighbor's link-local address to its neighbor cache for future use. The next time this needs to happen, the host consults its neighbor cache before repeating the same process.

Here is how the neighbor cache on a node could be displayed:

a. Cisco IOS
```
show ipv6 neighbor
```
b. Juniper JUNOS
```
show ipv6 neighbor
```
c. UNIX/Linux
```
ip -6 neighbor
```
d. Microsoft Windows
```
ipv6 nc
```

Privacy Extension for Auto-Configured Addresses

Use of embedded IEEE identifiers, such as Ethernet MAC addresses, in deriving addresses via SLAAC raised various privacy and security concerns, particularly concerning the potential to track user activity and eavesdropping. Privacy extension provides a solution to this problem by generating a pseudo-random interface ID which enables the host to change its address regularly [7].

To enable privacy extension on Linux, edit the /etc/sysctl.conf file:

```
sudo vi /etc/sysctl.conf
```

and add the following lines:

```
net.ipv6.conf.wlan0.use_tempaddr = 2
net.ipv6.conf.eth0.use_tempaddr = 2
net.ipv6.conf.all.use_tempaddr = 2
net.ipv6.conf.default.use_tempaddr = 2
```

Then restart the network service.

For Mac OS X, follow the same procedure, but just add the line:

```
net.inet6.ip6.use_tempaddr=1
```

For Microsoft Windows, the line to add to sysctl.conf is:

```
netsh interface ipv6 set global randomizeidentifiers=enabled
```

Duplicate Address Detection (DAD)

IPv6 requires each host to perform Duplicate Address Detection (DAD) before it can transmit using a given address, except when using anycast addresses. The node starts DAD by sending a neighbor solicitation message with the tentative address as the target address. The IP source of this message is the unspecified address and the destination is the solicited node multicast address. If a node on the network matches the neighbor solicitation target address, it will respond with a neighbor advertisement message using the tentative address as both source and destination. This indicates that the address is already in use and manual reconfiguration may be required. If no neighbor advertisement is received within the timeout period, it is safe to use this address.

Unreachability Detection

A key function provided by ND is the ability to detect bidirectional neighbor node reachability without completely relying on higher-level protocols. Reachability can be confirmed either by receiving a confirmation from a higher-level protocol, e.g., an ACK in a TCP session, or by receiving a neighbor advertisement in response to a neighbor solicitation request. Once neighbor reachability is confirmed, the neighbor cache is updated to reflect the status.

Neighbor Discovery Configuration

Neighbor discovery is a powerful protocol which provides extensive local-link network control. Below is a sampling of common configuration options for Cisco IOS:

To disable the prefix announcement:

```
ipv6 nd prefix 2001:0db8:1234:1::/64 no-advertise
```

To disable SLAAC:

```
ipv6 nd prefix 2001:0db8:1234:1::/64 no-autoconfig
```

To adjust the reachability timers to reduce the neighbor reachability periods:

```
ipv6 nd reachable-time 750000
```

To suppress the router announcements (RA):

```
ipv6 nd ra supress
```

To adjust the RA interval periods and lifetime:

```
ipv6 nd ra interval
ipv6 nd ra lifetime
```

To adjust the NS periods:

```
ipv6 nd ns-interval
```

To adjust the router priority:

```
ipv6 nd router-preference <High/ Medium / Low>
```

IPv6 Routing

Routing is the process of moving data packets across the network from one host to another. This process is usually handled by routers. Generally speaking, network routers utilize sets of messages to exchange information required to forward packets across networks. These standardized messages and signals form a routing protocol.

Routers maintain a local database, usually referred to as the routing table. This database includes a set of hosts, subnets, or networks, next-hop addresses, preferences, routing protocol metrics, routing protocol names, and interfaces. Routers consult the routing table to determine where to forward the packets. In general, there are two methods used to populate route tables: manually, via static routes, or automatically, via dynamic routing protocols.

Static Routes

As the name implies, static routes are manually configured by the administrator of the router and do not adapt to changes on the network such as link or device failures. Static routes provide the administrator with ultimate control of the device, along with a greater level of security. On the other hand, static routes place a huge administrative burden on network administrators, particularly in large networks.

An example: During the initial phases of the IPv6 pilot at the Google enterprise network, as detailed in the following chapters, static routes were used to provide IPv6 routing between end hosts and the IPv6 gateway. Static routes were chosen since they offered the best protection against any kind of unintended or malicious route injection. On the other hand, as the implementation of IPv6 grew in our network, the task of maintaining static routes became so tedious and time-consuming that we decided to switch to dynamic routing protocols.

IPv6 static route configuration is identical to IPv4 static route configuration on most network platforms. Below are two configuration examples.

Cisco IOS static route:

```
ipv6 route 2001:0db8:2345::/48 2001:0db8:1234:4567::1
```

Juniper JUNOS static route:

```
routing-options {
 rib inet6.0 {
  static {
 route 2001:db8:2345::/48 next-hop 2001:db8:1234::1;
  }
 }
}
```

Routing Protocols

The development of IPv6 introduced the necessity of enhancing and/or modifying existing routing protocols to support IPv6-specific requirements. As a result, newer versions of routing protocols were introduced to handle IPv6 routing. Table 6 shows the various protocol versions and the support for IP protocols.

Protocol Name	Support in IPv4	Support in IPv6
Routing Information Protocol	RIPv1, RIPv2	RIPng
Open Short Path First	OSPFv2	OSPFv3
Integrated Intermediate System to Intermediate System	Integrated IS-IS	Integrated IS-IS
Enhanced Interior Gateway Routing Protocol	EIGRP	EIGRP
Border Gateway Protocol	BGP	MP-BGP

Table 6. Different Routing Protocols' Support for IPv4 and IPv6

The selection of an IPv6 routing protocol usually depends on factors such as:

- ❖ Scalability of the implementation and the size of the network: RIP has a known scalability limitation of a maximum of 15 hops for the diameter of the network [13].
- ❖ Convergence time: Distance vector protocols are synonymous with slow convergence.
- ❖ Routing protocol metrics: RIP relies on hop counts as the only metric, which limits the flexibility of routing decisions [13].
- ❖ Existing implementation of IPv4 routing protocols: Familiarity with a specific routing protocol can lead the implementer to select the next version of the protocol to provide IPv6 routing.
- ❖ Standards-based vs. proprietary routing protocols: For example, EIGRP is a Cisco proprietary protocol, in contrast to all other protocols listed on the table, which happen to be standards-based.
- ❖ Consolidation of the routing protocols for both IPv6 and IPv4: IPv6 offers an organization a unique opportunity to optimize and/or change its internal routing architecture. As an example, many administrators choose to implement IS-IS, since it has a distinct advantage over other standard Interior Gateway Protocols (OSPF and RIP), with its ability to support multiple protocols on a single routing protocol process [14].

Although there were several good reasons for the IPv6 deployment team at Google to consider IS-IS, we decided to select OSPFv3 as the IGP, because of our familiarity with OSPFv2.

OSPFv3 Protocol

OSPFv3 was introduced to support IPv6 (RFC 2740) [15]. As a version modification of its predecessor, OSPFv2, it maintains several key features and mechanisms, such as:

- ❖ Link state flooding
- ❖ Designated router and backup designated router election
- ❖ OSPF domain segmentation into areas
- ❖ Area types and summarization

- Shortest Path algorithms and optimum route selection calculations

Here are the main differences from OSPFv2 that appear in OSPFv3:

- Runs per-link, instead of per-subnet as in OSPFv2. This feature difference was driven by the need to support multiple IP subnets on a single link.
- Uses IPv6 link-local addresses (FE80::/10) to communicate with other OSPF routers except on OSPF virtual links.
- Has three new flooding scopes:
  - Link-LSA: Flooding is limited to local link.
  - Area-LSA: Flooding is limited inside the area.
  - AS-LSA: Domain-wide flooding is permitted.
- Supports multiple instances per link.
- No longer supports OSPF authentication.OSPFv3 relies on IP (AH) authentication header and ESP encapsulation security payload for authentication, confidentiality, and integrity.
- Packets and LSA formats have changed in OSPFv3.
- Explicitly handles unknown LSA types.

In summary, OSPFv3 protocol changes were introduced to handle specific IPv6 requirements. The main concepts in OSPF such as areas, summarization, links types, and OSPF adjacencies were preserved, which drastically reduces the learning curve of OSPFv3 for anyone familiar with OSPFv2.

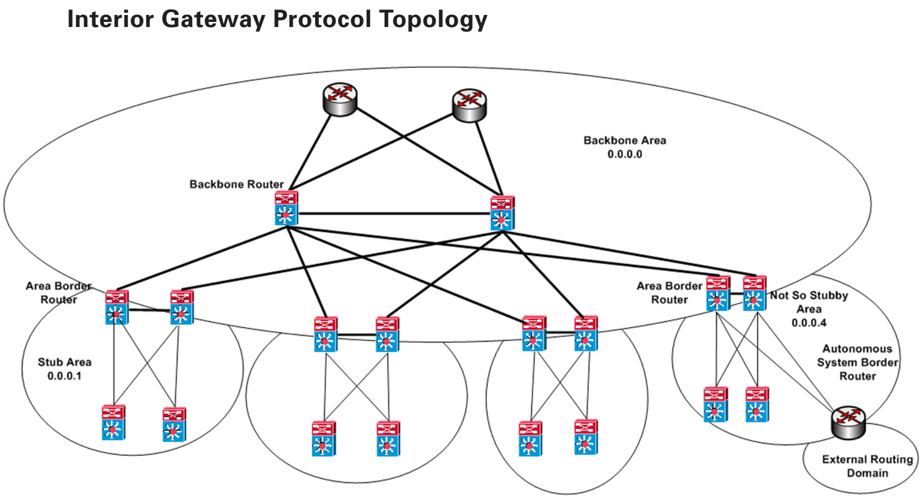

Figure 6. OSPF Topology for a Large Campus

Generally, in a large-scale network deployment, the optimal design choice is to segment the OSPF domain into smaller OSPF areas, to either reduce LSA flooding or control the announcements between the areas. The rationale for containing LSAs in smaller logical entities drives multi-area deployments.

Also, OSPF outbound route filtering inside the area can be challenging, due to the way OSPF floods the LSAs inside the area, which may influence the implementer to divide the domain's logical boundaries in order to be able to control the OSPF routing announcements [16].

Here is a sample OSPFv3 configuration on Cisco IOS:

```
interface gigabitethernet 2/6
ipv6 address 2001:0db8:1234:1000:3401::1/127
ipv6 ospf 100 area 1.2.3.4

interface gigabitethernet 2/6
ipv6 address 2001:0db8:1234:1000:3501::1/127
ipv6 ospf 100 area 0.0.0.0

ipv6 router ospf 100
  router-id 1.2.3.4
  log-adjacency-changes
  auto-cost reference-bandwidth 10000
  area 1.2.3.4 nssa no-summary
  area 1.2.3.4 range 2001:0db8:1234:1000::/52
  passive-interface default
  no passive-interface GigabitEthernet2/6
  no passive-interface GigabitEthernet2/7
```

And here is a sample OSPFv3 configuration on Juniper JUNOS:

```
protocols
     ospf3 {
      reference-bandwidth 10g;
       area 0.0.0.0 {
            interface lo0.0;
       interface ge-0/0/0.0;
       interface ge-1/0/0.0;
            }
     }
```

Multiprotocol Border Gateway Protocol (MP-BGP)

Border Gateway Protocol (BGP) is an Exterior Gateway Protocol (EGP) used to interconnect independent routing domains known as Autonomous Systems (AS). BGP is well-known as the routing protocol used widely to interconnect different Internet service providers. Multiprotocol BGP is an extension for BGPv4 that enables it to carry routing information for multiple Layer 3 protocols (IPv6, IPX, etc.) [17].

MP-BGP uses a capability field within the Address Family Identifier (AFI) to distinguish among the different protocols routed inside the advertisements. MP-BGP reserves AFI 2 to announce IPv6 routes. The MP-BGP protocol was designed to be backward-compatible with BGPv4. Configuring MP-BGP is similar to the standard BGP configuration process.

Here is a sample Cisco IOS MP-BGP IPv6 unicast configuration:

```
router bgp 64899
 no bgp default ipv4-unicast
  address-family ipv6 unicast
   neighbor 2001:0db8:2345:1234::1 remote-as 64895
   neighbor 2001:0db8:2345:1234::1 activate
   network 2001:0db8:5678::/48
   neighbor 2001:0db8:2345:1234::1 route-map OUTBOUND-POLICY
     out
   neighbor 2001:0db8:2345:1234::1 route-map INBOUND-POLICY
     in
```

And here is a sample Juniper JUNOS MP-BGP configuration:

```
protocols {
  bgp {
   family inet6 {
      unicast;
      }
      neighbor 2001:0db8:2345:1234::1 {
      peer-as 64895;
      export OUTBOUND-POLICY;
      import INBOUND-POLICY;
  }
}
```

Address Summarization

IPv6 provides a large address space for each organization. As we described in Chapter 5, it is crucial to allocate addresses in a hierarchical manner, to enable optimum address summarization and maintain smaller routing table size internally and in the global routing table.

OSPFv3 supports the same address summarization scheme for inter-area routes and for external routes. By dividing the network into smaller areas and allocating a single aggregate prefix for each area, the number of inter-area LSAs can be reduced dramatically and the convergence of the OSPF domain can be improved.

Here is a sample Cisco IOS OSPFv2 summarization:

```
ipv6 router ospf 100
  area 1.2.3.4 range 2001:0db8:1234:0100::/52
  summary-prefix 2001:0db8:1234::/48
```

And here is a sample Juniper JUNOS OSPFv3 summarization:

```
ospf3 {
  area 1.2.3.4 {
    area-range 2001:0db8:1234:0100::/52
  }
```

MP-BGP IPv6 summarization is identical to IPv4 BGP summarization implementation.

Here is a sample Cisco IOS MP-BGP summarization:

```
router bgp 64895
  address-family ipv6 unicast
    aggregate-address 2001:0db8:1234::/48 summary-only
```

And here is a sample Juniper JUNOS MP-BGP/OSPFv3 external summarization:

```
routing-options {
  rib inet6.0 {
    aggregate {
    route 2001:0db8:1234:/48 {
          discard;
    }
    }
  }
}
```

First-Hop Redundancy for IPv6

IPv6 neighbor discovery's periodic router announcements and neighbor unreachability feature provide a mechanism to address the redundancy of the first hop on the local link.

Using neighbor discovery default parameters, the hosts on the links will be able to detect a failure of the router within 38 seconds, which is considered a long time in today's networking world. Although this failover delay can be improved by adjusting the neighbor discovery timers on the link, that is not a recommended practice, due to the risk of increasing the neighbor discovery traffic overhead on the link. The best alternative is to implement first-hop redundancy via IETF VRRP (standard) or Cisco HSRP (proprietary) protocols [18].

Cisco Hot Standby Router Protocol (HSRP) is designed to ensure high network availability by providing first-hop redundancy for IP hosts on the LAN. HSRP is implemented by forming a group of routers in the LAN. One router in the group will be elected to take the active role of forwarding the network traffic, while the other routers will assume the standby role. The standby router will monitor the status of the HSRP using hello keepalive packets and will take over the role of the active router if it detects that the current active router has any reachability failure [18].

The HSRPv2 protocol introduces support for IPv6 first-hop redundancy. It creates a virtual MAC address, using the HSRP group number, and a virtual IPv6 link-local address. The active router sends the periodic RA using the virtual link-local address. HSRP uses priority to influence the selection of the active router in the group.

The Virtual Router Redundancy Protocol was introduced by the IETF to handle first-hop redundancy on the LAN. VRRP elects a master router within a group of routers to handle traffic forwarding, while other routers take the role of backups. Virtual Router ID (VRID) is associated with the master router MAC address. The backup router will monitor the VRRP announcements of the master and take over as master if the current master fails or experiences reachability issues [19].

VRRPv3 adds support for IPv6 first-hop redundancy: The VRID gets associated with the master router after election. The master router sends router announcements using the virtual router link-local address and responds to any neighbor solicitation for the virtual router address. For more details of VRRPv3 operation, see RFC 5798 [20].

Here is a sample Cisco HSRPv2 configuration:

Primary:

```
interface vlan 402
  ipv6 address 2001:0db8:1234:1:2::1/64
  standby version 2
  standby 402 ipv6 autoconfig
  standby 402 priority 200
  standby 402 preempt delay minimum 600
```

Secondary:

```
interface vlan 402
  ipv6 address 2001:0db8:1234:1:2::2/64
  standby version 2
  standby 402 ipv6 autoconfig
  standby 402 priority 200
  standby 402 preempt delay minimum 600
```

And here is a sample Juniper JUNOS VRRPv3 for IPv6 configuration [21]:

```
ge-0/1/0 {
  unit 0 {
  family inet6 {
    address 2001:0db8:1234:4::1/64 {
    vrrp-inet6-group 1 {
      virtual-inet6-address 2001:0db8:1234:4::100;
      priority 200;
      preempt;
      accept-data;
    }
    }
  }
  }
}
```

Summary

This chapter provided details about the requirements, capabilities, and impacts of more advanced IPv6 technologies such as ICMPv6, neighbor discovery, IPv6 routing, and first-hop redundancy. Multiple other solutions and interesting topics exist and could be used in other deployments—for example, IPv6 renumbering, multicast, quality of service—however, those are not covered here. The topics we discussed were selected specifically for their relevance to the Google Enterprise IPv6 deployment.

7. IPv6 Planning

Enterprise Network Evolution

Since the beginning of the Internet, enterprise networks have evolved from a collection of loosely connected workstations, manually configured printing services, and supercomputers into a dynamic environment with constantly evolving systems and applications. Such networks vary at multiple levels, from large campuses with multiple buildings, to regional offices, cloud computing, and multiple distributed services and resources. The basic requirement is no longer just to connect end devices, such as user workstations, wireless access points, printers, or Voice over IP phones, to a network, but also to be scalable, secure, with decreased cost and lowered overhead. For example, security used to be performed mainly at the perimeter of the network, by routing traffic through a firewall. Today, the security model is much more network- or even application-centric; it is embedded within the network infrastructure. Another example is the way enterprise networks are moving toward high-speed wireless access anywhere and convergence at the service level.

The modern corporate network also strives to ensure simplicity, through end-to-end connectivity and innovation, to provide a basis for developing new network services. Although the fundamental drivers of increased network traffic—cost-effective processing power, new applications, pervasive information-sharing—have not been displaced, changing business requirements have accelerated the pace of this growth.

Introduction to the Typical Enterprise

Today's corporate deployments differ greatly in size, distribution, networking hardware, connectivity type, and services provided. However, each of these networks follows the well-known three-layered design, with core, aggregation (distribution), and access layers deployed [22].

Figure 7 (next page) shows a single-site three-layer enterprise campus network with various types of end devices connected to the network via both wired and wireless access. The edge network services are provided by core devices.

Regardless of differences in architecture among various enterprise network types, each of them generally provides at least the following three types of functionality:

❖ Internet presence: Access to services and content offered by the enterprise to the customer, partners, and/or the general public over the Internet. These services and content are usually located in one or multiple data centers of the enterprise.

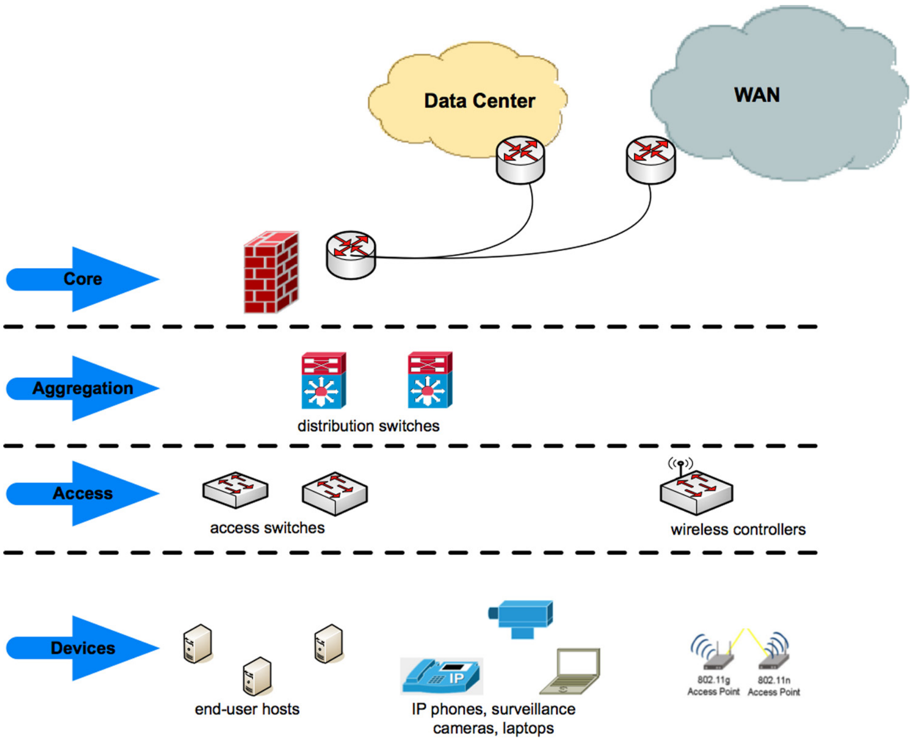

Figure 7. Typical Enterprise Network Architecture Today

- Intranet end-user Internet connectivity: Access to services and content on the Internet for enterprise employees.
- Intranet applications: Access to services and content located inside the enterprise and accessed only by enterprise users and applications. These services and content are usually located in the enterprise data centers, but can also exist in other locations such as offices.

To be able to expand, these networks need IP addresses so that they can communicate with each other and with the rest of the world. Since the IPv4 address space is almost exhausted, deploying the IPv6 protocol is the only option that can assure continuous growth of the enterprise networks themselves along with the global Internet, improving openness, simplicity, and innovation.

Key Areas to Be Identified Before Planning Begins

Once the IT team in an enterprise decides to deploy IPv6 inside the organization, they must start by answering several questions:

1. Do they need to deploy IPv6 everywhere?
2. Should they fully transition at once or integrate gradually?
3. What's the status of hardware and software on the network?
4. Does the service provider support all required services over IPv6?

5. Where can they get IPv6 addresses?
6. What are the operational costs of obtaining addresses?

The answers to these questions affect the ease of IPv6 integration and will generally indicate how early an organization can start working on the design and implementation of an IPv6-capable network.

The remainder of this chapter examines the steps involved in planning IPv6 integration using the experience we, the IPv6 deployment team at Google, gained in planning for IPv6 support in the Google enterprise network.

Design Philosophy and Key Design Decisions

As mentioned earlier, our project started as a grass-roots activity undertaken by enthusiastic volunteers who followed the Google practice of contributing 20% of their time to internal projects that fascinate them. The first volunteers had to learn about the new protocol from books and then plan labs to start gaining practical experience. Our ultimate goal was to enable IPv6 across the entire corporate network, so that internal services and applications could follow.

Our methodology was driven by seven principles:

1. Think globally and try to enable IPv6 everywhere—in every office, on every host and every service and application we run or use inside our corporate network.
2. Work iteratively: Plan, implement, and iterate launching small pieces rather than trying to complete everything at once. This way our team members could get to know the IPv6 protocol, gain confidence using it, and no longer refer to IPv6 as the "new," but instead start referring to IPv4 as the "old" protocol.
3. Implement reliably: Every IPv6 implementation had to be as reliable and capable as the IPv4 ones, or else no one would use and rely on the new protocol connectivity. Also, an unreliable IPv6 implementation would create or contribute to the perception that IPv6 is unreliable and should not be used, when actually the protocol would not be unreliable; it would be the design and deployment that were at fault.
4. Don't add downtime: Fold the IPv6 deployments into our normal upgrade cycles, to avoid additional network outages.
5. Follow the KISS ("Keep It Simple, Stupid") design principle: Avoid unnecessary complexity by keeping the IPv6 network design as close to the IPv4 one as possible. Any deviations should be due to inherent differences in the underlying protocols/technologies, operational practices, and device capability constraints.
6. Avoid SPOFs (single points of failure) as much as possible.
7. Take risks: Do not be afraid to learn lessons along the way in order to end by making the world a better place!

Our planning began with extensive internal communications. We wanted to ensure we had management and senior engineering support, and we needed the initial group of volunteers to focus on education and get to know the new protocol.

We started auditing the existing standard hardware and, together with our network equipment vendors, evaluated what needed to be upgraded or replaced in order to support the required IPv6 features. Then we identified software and services and developed an upgrade plan for them as well. Of course, not everything could be upgraded at once, especially sensitive applications (e.g., critical financial applications), but listing the items we needed to work on was a good start. After that we reviewed the current vendor support for IPv6 and evaluated appropriate technical options for deployment.

Last but not least, a crucial decision we took was to include IPv6 in all existing procurement, deployment, and support procedures globally. This decision means, for example, that if a proposed new piece of hardware equipment, new service, or new technology does not support IPv6, we will not be implementing it inside our corporate network. It also means that if there is an operational issue with the IPv6 network, the problem will be resolved with the same SLA (Service Level Agreement) as for IPv4.

Chapter 5 explained in detail what considerations need to be taken into account for allocating and assigning address space within an organization. Following these recommendations and best practices, we created a comprehensive addressing plan for offices, campus buildings, and data centers. Our initial IPv6 addressing scheme also followed the guidelines specified in RFC 5375 (IPv6 unicast address assignment) [23]. Our plan became:

- Assign /64 for each VLAN.
- Assign /56 for each building.
- Assign /48 for each campus or office.
- Assign /128 for loopback addresses.

We decided to use Stateless Address Auto-Configuration capability [24] for end-host IP address assignments. As we saw in previous chapters, SLAAC enables a host to generate its own addresses using a combination of locally available information and information advertised by routers, thus eliminating manual address assignment. SLAAC also allowed us to work around the limited DHCPv6 client support that was available in various operating systems at the time and helped speed the rollout of IPv6 [25]. In addition, SLAAC provides a seamless method for renumbering and providing address privacy via the privacy extension feature (RFC 4941) [26].

Meanwhile, we decided not to use ULA (Unique-Local Addressing, RFC 4193), but to take advantage of global IPv6 addresses. We requested provider-independent (PI) space from various Regional Internet Registries [27]. We needed PI IPv6 space to solve any potential multihoming issues with our many service providers. In 2008 we got our first ARIN-assigned /40 IPv6 space for GOOGLE IT, so we were ready to proceed!

Next, we had to choose which transition mechanism to use:

- Dual-stack [28]
- Building various types of tunnels (as a 6-to-4 transitioning mechanism) [29] on top of the existing IPv4 infrastructure
- Creating a separate IPv6 infrastructure

The third option was our least preferred choice, as it would have required additional time and resources to order data circuits and to build a separate parallel infrastructure for IPv6 connectivity. That's why we adopted a simple mantra: *Dual-stack when you can and* *tunnel when you must.* Dual-stack was also the best choice for us, since IPv4 and IPv6 will most probably need to co-exist in our enterprise for several more years.

Our corporate network infrastructure follows industry recommendations and does not differ much from the standard enterprise architecture described earlier. Our distributed locations and corporate data centers have access, aggregation, and core layers, built from different networking vendors' platforms and device models, including Cisco, Juniper, Aruba, and Silverpeak. Because of the hardware and software diversity, we had a choice of multiple approaches:

- ❖ Enable IPv6 at the core first and then move toward the access layer.
- ❖ Enable IPv6 at the access layer and then move toward the core.
- ❖ Create isolated IPv6 islands across the network.

Here we took the hybrid approach of enabling IPv6 at our core devices first and then having isolated IPv6 islands (offices) across the corporate network. This allowed us to provide IPv6 access to those end hosts that were already IPv6-capable.

We also designed a scalable IPv6 backbone to accommodate all existing WAN clouds—MPLS, Internet transit, and the Google production network, which we use as a service provider. Along with the decision to dual-stack the network, we tried to keep the IPv6 network design as close to the IPv4 network, in terms of routing and traffic flows, as possible. We applied our principle of changing the minimum amount necessary.

By keeping the IPv6 design simple, we wanted to ensure scalability and manageability: a simple design requires the least effort from the network operations team to support it.

Planning is essential before the new protocol can be deployed in an enterprise network. No single model can apply to all networks, due to the wide diversity in architectures and in the services enterprises offer around the world. Each enterprise must make its own decisions about transition mechanisms, deployment strategy, and implementation. We have given a summary of the experience we gained while planning for IPv6 deployment in Google Corporate as a guide for others in working through these choices. In the next chapter we continue to discuss the details of our corporate deployment.

8. Google's Corporate Deployment

At the time that we, the deployment team at Google, started planning for IPv6 there were not many public tunnel or broker services available to provide IPv6 access for home networks. That is why everyone involved in the initial deployments was very eager to get their hands dirty deploying IPv6 in the corporate network—which, in turn, let us move quickly with our deployments.

Deployment Evolution

As mentioned already, we decided to follow the guideline: *Dual-stack when you can, tunnel when you must.* Dual-stack was an obvious choice mainly because of its capability to support a gradual evolution to an IPv6-dominant network. It is true that tunnels are generally considered unhealthy. They are hard to troubleshoot and introduce a lot of latency. On the other hand, tunnels offer a great way for network engineers to experiment with IPv6, and deployment can be fairly rapid.

While we were conducting our initial tests and were getting to know the protocol, several volunteers expressed a desire to experiment with IPv6. In order to quickly set up connectivity for their machines, we created static GRE tunnels between them and a single dual-stacked router we happened to have that was connected to our upstream transit provider. These tunnels would, in some cases, add an additional 200 ms of latency.

After our initial tests and design docs were completed, we dual-stacked several labs and small locations where there was a high demand for IPv6—engineering offices with folks working on IPv6 support for Google products. The dual-stacked labs had a GRE tunnel between the lab egress routers and this same single dual-stacked router, which provided IPv6 connectivity at that time (see Figure 8, next page).

These tunnels allowed us to test and validate IPv6 routing and reachability, in addition to allowing engineers to develop and test IPv6 applications without having to wait for an infrastructure upgrade. Another benefit of providing WAN connectivity by directly encapsulating IPv6 in GRE packets was that we avoided having to use any protocol-translation technologies.

After thoroughly testing the dual-stack transition mechanism in the labs, we started looking into implementing it on our WAN links. We were approaching a stage where we needed to significantly increase the footprint of our deployment, and GRE tunnels do not scale to large deployments. The management overhead for maintaining GRE tunnels would simply have been too big to sustain operations.

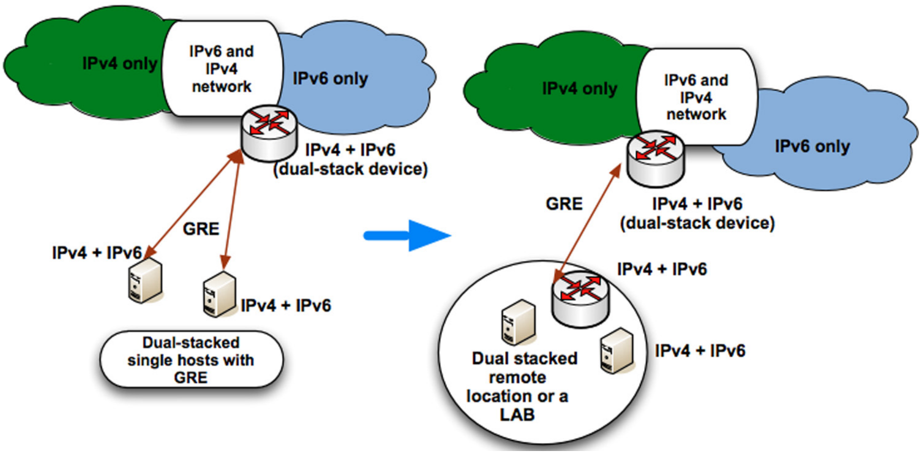
IPv4 only
IPv6 and IPv4 network
IPv6 only
IPv4 + IPv6 (dual-stack device)
GRE
IPv4 + IPv6
IPv4 + IPv6
Dual-stacked single hosts with GRE
IPv4 only
IPv6 and IPv4 network
IPv6 only
GRE
IPv4 + IPv6 (dual-stack device)
IPv4 + IPv6
IPv4 + IPv6
Dual stacked remote location or a LAB

Figure 8. Phase I: Dual-Stack Separate Hosts and Labs Using GRE

In parallel, we expanded our WAN infrastructure from a single dual-stacked router, to IPv6+IPv4-connected and -capable devices in several major points of presence (PoP) around the world, where the newly created tunnels were terminated. At this point the IPv6 WAN consisted of production grade devices and links. In addition, we expanded our testing and planning to explore integration of more advanced services over IPv6, such as multicast and QoS.

In the next phase we started dual-stacking entire offices and campus buildings and then building GRE tunnels from the WAN border routers to the egress IPv6 peering routers (Figure 9).

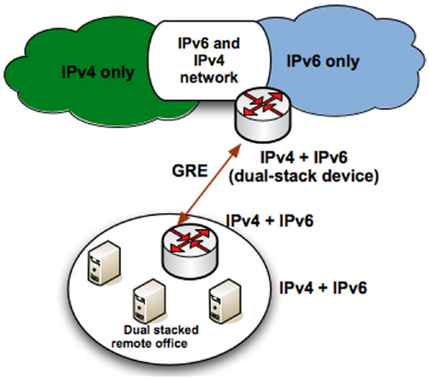

Figure 9. Phase II: Dual-Stack Offices and Campus Buildings, Still Using GRE Tunnels

During these implementation phases we constantly changed the initial designs, defining more granular IPv6 routing policies and creating multiple operational documents about how to support or troubleshoot IPv6 issues.

Also, over time, we obtained more IPv6 blocks from various Regional Internet Registries to support growth in our offices. We invested a lot of time in coming up with a flexible and scalable addressing plan that allowed us to easily incorporate addresses from various

RIRs as needed. We also concluded that the technologies we chose worked well in our environment. Although SLAAC, in particular, had some disadvantages, such as not providing DNS server addresses to the hosts, it did meet our needs.

In the third phase we started dual-stacking entire offices, including WAN upstream connections, while still trying to prioritize deployment in those offices with immediate need for IPv6. Unfortunately, dual-stacking WAN connections proved to be more challenging than we expected. Not all access or transit providers are ready with IPv6 offerings. The ones that are ready may or may not have IPv6 Service Level Agreements (SLAs) that are comparable to the ones offered with IPv4 products.

Using this phased deployment approach allowed us to gradually gain skills and confidence during the process. This incremental approach also enabled us to manage our risk effectively.

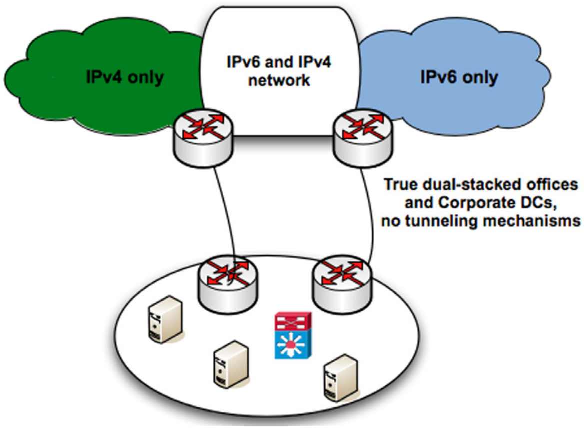
IPv4 only
IPv6 and IPv4 network
IPv6 only
True dual-stacked offices and Corporate DCs, no tunneling mechanisms

Figure 10. Phase III: Dual-Stack the Upstream WAN Connections to the Transit and Mpls VPN Providers

After most offices had been dual-stacked and hosts had IPv6 connectivity, either natively with dual-stacked upstream connections or via GRE over IPsec tunnels, we started looking for a way to run IPv6-only networks. Our goal was to have our access and aggregation network layers be IPv6 only. After evaluating various options, we decided to test and evaluate DS-Lite technology [30]. With DS-Lite, all packets going to an IPv4 destination travel to the office egress device encapsulated in IPv6 packets. The IPv6 headers are then decapsulated at the network exit point, which is a router that has both IPv4 and IPv6 addresses. There the IPv4 packets are extracted. After that, they follow normal routing across the IPv4 network. IPv6 packets, on the other side, traverse from the source to the destination unmodified (see Figure 11, next page).

This technology seems very promising, and new drafts/RFCs improving it are constantly being published. We made several successful tests and are in the process of implementing this in our production environment, but there is obviously still a long way to go before we can fully support an IPv6-only network.

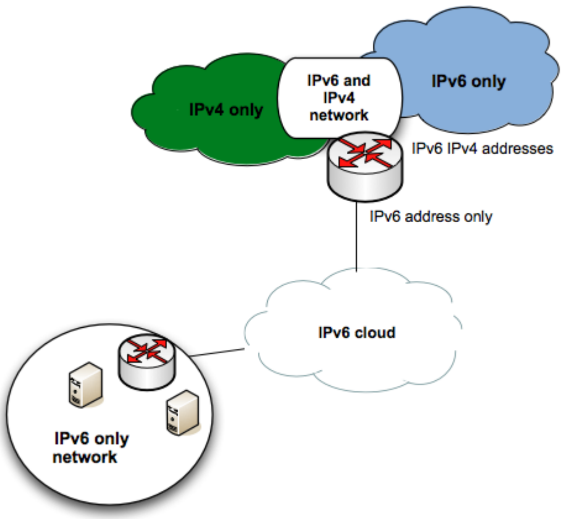

Figure 11. DS-Lite Trial Implementation

Configuration Samples

As described in detail earlier, we decided to use the following routing protocols:

- HSRPv2: First hop redundancy
- OSPFv3: Interior gateway protocol
- MP-BGP: Exterior gateway protocol

In this section, some configuration examples for core layer (border routers) and POP backbone routers are shown.

Interface and protocols configuration of the Juniper routers:

```
groups {
  GRE-INTERFACE-MTU1416 {
    interfaces {
      <gr*> {
      unit <*> {
        family inet6 {
          mtu 1416;

interfaces {
  gr-0/0/0 {
  unit 1 {
    apply-groups GRE-INTERFACE-MTU1416;
    description "IPv6 tunnel";
    tunnel {
    source 192.2.0.1;
```

```
      destination 198.51.100.1;
      }
      family inet6 {
      address 2001:DB8:0:1000::2/64;
      }
    }
    }
  }
  protocols {
    ospf3 {
    reference-bandwidth 10g;
    area 0.0.0.0 {
      interface lo0.0;
      interface ge-0/0/1.0;
      interface ge-0/0/2.0;
      }
    }
  }
    bgp {
    group GRE-v6 {
      type external;
      import POLICY-IN-v6;
      family inet6 {
      unicast;
      }
      export POLICY-OUT-v6;
      local-as 64xxx;
      multipath multiple-as;
      neighbor 2001:DB8:0:1000::1 {
      description "IPv6 tunnel";
      local-address 2001:DB8:0:1000::2;
      peer-as 65xxx;
      }
    }
```

In the example above, note that all interfaces are included in OSPFv3 and also that the design aims to be as simple as possible.

On Cisco:

```
interface Tunnel100
description IPv6 GRE tunnel
no ip address
ipv6 address 2001:DB8:0:1000::1/64
tunnel source 198.51.100.2
tunnel destination 203.0.113.2
ipv6 ospf 100 area 0.0.0.0
!

!
ipv6 router ospf 100
log-adjacency-changes
passive-interface default
no passive-interface Vlan301
no passive-interface GigabitEthernet3/1
no passive-interface GigabitEthernet3/2
!

router bgp 65xxx
address-family ipv6
  no synchronization
  network 2001:DB8:0:1111::1/128
  aggregate-address 2001:DB8:1000::/32 summary-only
attribute-map SET-COMMUNITY-V6
  neighbor GRE-v6 soft-reconfiguration inbound
  neighbor GRE-v6 route-map POLICY-IN-v6 in
  neighbor GRE-v6 route-map POLICY-OUT-v6 out
  neighbor 2001:DB8:0:1111::2 activate
  neighbor 2001:DB8:0:1111::2 soft-reconfiguration inbound
    exit
```

The following routing policies are applied to BGP peers across the GRE tunnel:

Juniper and Cisco configurations:

```
policy-options {
policy-statement POLICY-IN-v6
term default-in {
  from {
  route-filter ::0/0 exact;
  }
  then {
  local-preference 350;
  accept;
  }
}
term accept-in {
  from {
  prefix-list V6-Only
  }
  then {
  local-preference 350;
  community add LEARNED-v6;
  accept;
  }
}
term no-leaks {
  then reject;
}

policy-statement POLICY-OUT-v6 {
term aggregate-out {
  from community OFFICE-6INT;
  then accept;
}
term no-leak {
  then reject;
}

route-map POLICY-IN-v6 deny 10
description deny-default
match ipv6 address prefix-list DEFAULT-ONLY-IPv6
!
route-map POLICY-IN-v6 permit 15
description regional-subnet
match community IPv6-SUB
set local-preference 350
set community 65xxx:64000 additive
!
```

```
route-map POLICY-IN-v6 permit 20
description rest-all
match ipv6 address prefix-list V6-ONLY
set local-preference 300
set community 65xxx:64000 additive
!

route-map POLICY-OUT-v6 permit 10
description default-out
match ip address prefix-list DEFAULT-ONLY-IPv6
!
route-map POLICY-OUT-v6 permit 20
description regional-subnet
match community POP-V6
match ipv6 address prefix-list V6-ONLY
!
```

DS-lite AFTR node configuration on Juniper:

```
services {
service-set sset2 {
  syslog {
  host local {
    services any;
    }
  }
  tcp-mss 1024;
  softwire-rules r1;
  nat-rules r1;
  interface-service {
    service-interface sp-0/3/0.0;
  }
}
softwire {
  softwire-concentrator {
    ds-lite ds1 {
      softwire-address 2001:DB8:0:1000::1;
      mtu-v6 1400;
    }
  }
  rule r1 {
    match-direction input;
    term t1 {
      then {
        ds-lite ds1;
      }
    }
  }
}
nat {
  pool p1 {
    address 192.0.2.192/32;
    port {
      automatic;
    }
  }
  rule r1 {
    match-direction input;
    term t1 {
      from {
        source-address {
          192.0.0.0/29;
        }
      }
      then {
        translated {
          source-pool p1
          translation-type {
            source dynamic;
          }
        }
        syslog;
      }
    }
  }
}
chassis
fpc 0 {
    pic 3 {
      adaptive-services {
        service-package layer-3;
      }
    }
  }
interfaces {
  ge-0/0/0 {
    description "AFTR-Internet";
    unit 0 {
      family inet {
        address 192.0.2.1/24;
      }
    }
  }
    sp-0/3/0 {
      unit 0 {
        family inet;
        family inet6;
      }
    }
    ge-1/3/0 {
      description "B4 device";
      unit 0 {
        family inet {
          address 192.0.0.1/29;
        }
        family inet6 {
          service {
            input {
              service-set sset2;
            }
          output {
            service-set sset2;
          }
        }
        address 2001:DB8:0:1000::1/64;
     }
    }
  }
```

Corresponding configuration for the B4 element (in our case, that is a Linux machine running 2.6.32 kernel):

```
#sudo ip -6 tunnel add dsltun mode ipip6 remote
2001:DB8:0:1000::1 local 2001:DB8:0:2222::1ff:fe00:284c dev
eth0

#sudo ip link set dev dsltun up

#sudo ip route add 203.0.113.113/32 dev dsltun

#sudo ip route add 192.168.0.1/32 dev dsltun

--> Redirected one external and one internal routes to the
tunnels
```

Machine interfaces and routing table state:

```
linux:~$ ifconfig dsltun

dsltun     Link encap:UNSPEC HWaddr 26-20-00-00-10-5F-
00-1B-00-00-00-00-00-00-00-00

  inet6 addr: fe80::a800:1ff:fe00:284c/64 Scope:Link

  UP POINTOPOINT RUNNING NOARP MTU:1460 Metric:1

  RX packets:6 errors:0 dropped:0 overruns:0 frame:0

  TX packets:314404 errors:227 dropped:227 overruns:0 car-
rier:224

  collisions:0 txqueuelen:0

  RX bytes:504 (504.0 B) TX bytes:31445748 (31.4 MB)
```

```
linux:~$ netstat -rn

Kernel IP routing table

Destination   Gateway    Genmask         Flags MSS Window irtt
Iface

203.0.113.113 0.0.0.0    255.255.255.255 UH     0  0        0
dsltun

10.0.0.254    0.0.0.0    255.255.255.255 UH     0  0        0
eth0

192.168.0.1   0.0.0.0    255.255.255.255 UH     0  0        0
dsltun

0.0.0.0       10.0.0.254 0.0.0.0         UG     0  0        0
eth0
```

AFTR router NAT table:

```
Interface: sp-0/3/0, Service set: sset2

Flow                         State              Dir           Frm
count

IPIP 2001:DB8:0:2222::1ff:fe00:284c ->2001:DB8:0:1000::1:0
Forward I                    0

     Softwire 2620:0:105f:1b:a800:1ff:fe00:284c ->
2001:DB8:0:1000::1

TCP  10.0.0.1:55673 -> 203.0.113.113:80     Forward I     6

     NAT source          10.0.0.1:55673 -> 192.0.2.192:1209

     Softwire 2620:0:105f:1b:a800:1ff:fe00:284c ->
2001:DB8:0:1000::1

ICMP                         203.0.113.113      -> 192.0.2.192
Watch                        O                  8

     NAT dest            192.0.2.192        ->
10.0.0.1

     Softwire 2620:0:105f:1b:a800:1ff:fe00:284c ->
2001:DB8:0:1000::1

ICMP                         10.0.0.1           -> 203.0.113.113
Watch                        I                  8

     NAT source          10.0.0.1           -> 192.0.2.192

     Softwire 2620:0:105f:1b:a800:1ff:fe00:284c ->
2001:DB8:0:1000::1

TCP  203.0.113.113:80     -> 192.0.2.192:1209 Forward O   6

     NAT dest            192.0.2.192:1209   ->
10.0.0.1:55673

     Softwire 2620:0:105f:1b:a800:1ff:fe00:284c ->
2001:DB8:0:1000::1
```

Dual-Stack Network Operation and Management

Having a dual-stacked or IPv6-only network is not just a deployment challenge, but also an operational one. A critical aspect of transitioning to IPv6 technology is ensuring that the right network monitoring software systems are in place to guarantee business continuity. The biggest issue here is that many monitoring applications either do not support IPv6 at all or include only basic IPv6 functionality. With the increased address space and auto-configuration capabilities, enterprises need tools for effective hierarchical IP address management and for monitoring various protocol adjacency states.

An important first step in transitioning to enhanced IPv6 tools support was implementing NetFlow v9 on some of our aggregation and core devices. The NetFlow v9 protocol allows export of both IPv4 and IPv6 traffic samples, thus facilitating IPv6 application and bandwidth analysis.

Security Considerations

Proper security planning and implementation are vital for the success of any IPv6 deployment. For an enterprise network, the main difference in IPv6 protocol security is that with IPv6 we do not need and thus do not deploy NAT (network address translation) anymore. Some network engineers still believe that NAT is a security feature, since it hides the internal enterprise network. While it is true that LAN hosts are not accessible from outside the netwoek, NAT alone cannot protect the enterprise network. Even under IPv4, enterprises need to have a solid way to secure the network: the appropriate access control lists (ACLs) to filter inbound and outbound traffic should be in place.

When creating IPv6 ACLs and control plane policies, we tried to follow industry best practices, which continue to evolve. The following basic rules were adhered to:

1. Filter internal-use IPv6 addresses at the enterprise border routers.
2. Discard packets from and to bogon and reserved space.
3. Accept only certain ICMPv6 error messages (simply blocking all ICMP is no longer an option with IPv6, due to the way neighbor discovery and MTU discovery work) [31].
4. Permit ICMPv6 ping.
5. Filter specific extension headers.
6. Filter unneeded services at the network edge.
7. Use anti-spoof filtering.

We also followed a couple of other industry best practices:

1. From a privacy point of view, it is important to enable privacy extensions for SLAAC.
2. Use standard MD5 authentication for IPv6 BGP peering.

IPv6 integration is not "just a network upgrade" but a complex process, involving many elements and capabilities, and these evolve over time, rather than changing all at once.

Multiple solutions and transition mechanisms are available. Which choices are best depends on the starting network architecture and its complexity. This chapter provided details on how we at Google approached the challenge of deploying IPv6 in our enterprise network and the key design and implementation principles we followed. We hope that network engineers and IT managers will find the material useful for making design and implementation decisions and that they will turn their networks into modern IPv6-capable enterprise infrastructures forming tomorrow's Internet.

9. Challenges and Issues Encountered

We faced numerous challenges during the planning and deployment phases, mainly technical, but also administrative and organizational ones, such as resource assignment, project prioritization, and training. Let's take a walk down Memory Lane and look at some of the most interesting issues we encountered over the past few years as we deployed IPv6 in the Google corporate network.

IPv6? Show Me the Money!

There used to be a time when IPv6 was considered an "Advanced Feature" by major network equipment vendors and came with a price tag. This cost factor undoubtedly contributed greatly to the delay in adoption of IPv6 around the world. Most IT managers found it hard to justify to the CFO why they needed dollars to deploy a technology that brought no perceivable benefits to the company. To get IPv6 under consideration took a great deal of energy invested by IPv6 evangelists all around the world and in enterprise boardrooms in arguing for IPv6 and presenting appropriate business cases to companies worldwide.

QA Department Outsourced to Customer

One of the biggest realizations that dawned upon us when we started deploying IPv6 was that we had become de facto QA engineers for the vendors, who were selling us IPv6 products. It turned out that most vendors run IPv6 only in developmental labs, not in their own corporate networks. So, effectively, they were not eating their own dogfood. Nobody can recreate real-world scenarios in the lab—there are just too many variables, permutations, and combinations to be tested. Instead of using their own products in their own networks, vendors often simply sold them to customers and expected them to do their QA testing, which was quite frustrating for us. The good news is that things are definitely moving in the right direction now: the vendors seem to be testing their solutions and features before they sell them.

O DHCPv6, Where Art Thou?

One of our basic design goals was to have design and operational parity between IPv4 and IPv6. To align ourselves with that goal, we needed DHCPv6 for address assignment, just as we have in the v4 world. Unfortunately, not all major operating systems had out-of-the-box DHCPv6 client functionality enabled. Various add-ons were available that one could integrate into OSes in order to support this feature, but that was not acceptable according to our internal IT policy. The lack of DHCPv6 support played a key role in our decision to go with the SLAAC technology. Fortunately, as of Q2 2011, all major OSes support DHCPv6 client functionality—a welcome change from 3–4 years ago when we started.

Where's My Dancing Turtle?

Picture a group of engineers huddled around a terminal wanting to see the dancing turtle for the first time. The person at the keyboard opens up his favorite open source browser and types in the legendary www.kame.net and hits Enter. To their dismay, they get to the Web page only to find a still turtle. Everything checks out: routing looks good, DNS is working, he can even ping over IPv6. What could the problem be? It turned out that browsers could independently enable or disable their IPv6 usage. For example, in Firefox, typing "about:config" in the address bar brings users to a configuration options page. If "network.dns.disableIPv6" is set to True, then they won't be able to get to the dancing turtle.

Reserved Anycast Addresses

We were excitedly planning to use all zeros or all Fs under a given prefix to address our router interfaces, because we thought the rules forbidding that no longer applied in IPv6. However, we were doomed to disappointment. All zeros is a reserved anycast address, the subnet router anycast address. The last 128 addresses of any given network are reserved for anycast purposes as well [32]. So, for example, in a given /64, as long as each interface has an address between 0001:0000:0000:0000 and FFFF:FFFF:FFFF:FF00, the routers are happy.

VLAN Pooling and IPv6 Don't Mix

Several WiFi solutions today have a VLAN pooling feature that allows the assignment of wireless clients IPs from different VLANs in a sort of round-robin fashion. This eliminates the need to assign a very large subnet for wireless clients and allows network administrators more flexibility in managing the WiFi network. When we dual-stacked our WiFi VLANs, we noticed that IPv6 failed to work for clients whenever VLAN pooling was enabled. Upon investigation, we realized that each client was receiving multiple router advertisements (RAs), one for each VLAN that was configured in the VLAN pool. It seemed as though the wireless controller was not respecting VLAN boundaries in IPv6 the way it did in IPv4.

As we dug deeper, we found out that the problem lay in the way WiFi treats multicast traffic in general. Since RAs are sent out as multicasts, the AP would essentially broadcast that multicast packet out into the air. Every client associated with that AP, regardless of which VLAN it belonged to, could listen to multicast traffic. As a result, each client would listen to every RA that was sent out across the VLANs on that particular AP. For example, if there were three clients associated with that AP, even though they were spread out across three different VLANs they would still be able to hear multicast/broadcast traffic destined to any of those three VLANs and hence would receive three RAs.

The interesting fact here is that the same problem exists in IPv4, but it has a somewhat less catastrophic effect. There is no real way around this problem other than for each vendor to implement a proprietary work-around or simply to turn off VLAN pooling and go back to a big, flat subnet in the IPv4 world. The solution provided by our vendor in a later software release was to implement IPv6 firewalling to restrict the neighbor discovery and routers advertisement multicast traffic leaking across VLANs.

We Need Updated Protocol Standards

None of the vendor WAN acceleration devices (or, also, the one we use in our corporate network) can optimize or encrypt IPv6 traffic using Web cache control protocol (WCCP) for traffic redirection, since the current protocol standard, WCCPv2 [33], does not support IPv6. Currently we are evaluating alternative redirection mechanisms such as policy based routing (PBR) [34] to overcome this limitation.

Fun with Hardware and Software Limitations

One of the most vexing issues we have discovered is that either major networking vendors lack hardware support for enterprise IPv6 features or the software implementation of such features leaves a lot to be desired. This is especially valid for some of the midrange devices and platforms. Here are some observations we found interesting:

- Certain hardware platforms support IPv6 in software only. This creates conditions where even a simple IPv6 ACL can cause high CPU usage.
- Not all vendors implement multiple tunneling mechanisms. For example, one of our routing platforms only offers GRE.
- Not all platforms/vendors support NetFlow v9.
- Memory resource allocation on some lower-end switches is limited, particularly when IPv6 routing is enabled. Optimizing the routing table using summarization to reduce the routing table size seemed our best option for addressing this problem.
- The majority of enterprise-class wireless vendors still lack basic IPv6 functionality. For example, at least one of our wireless equipment vendors did not have support for IPv6 ACLs for a long time and still currently lacks support for IPv6 routing.
- Neighbor discovery unreachability detection can have substantial CPU overhead on campus core switches, especially when the number of network devices connected to the switch grows. In cases like this, we had to lower the neighbor discovery reachability timer in order to reduce CPU utilization on the switches with higher numbers of network devices.

It's Not Always (Just) the Network!

Enabling IPv6 on the network sometimes caused unexpected results. Here are some of the common issues we saw:

- Standard enterprise OSes would automatically create ISATAP tunneling as their default IPv6 connectivity method. This could not only lead to broken IPv6 connectivity but also have security implications.
- Older enterprise OSes and applications generally either do not support IPv6 at all or have IPv6 disabled by default.
- Applications not supporting IPv6 include proxy support for IPv6, VoIP call manager, and other key applications.

- Not all printers really support IPv6, although some do offer limited functionality.

Organizational Challenges

Resource allocation tends to be very IPv4-centric. This is primarily due to the fact that IPv4 is still the bread and butter of most organizations, so the operational stability of IPv4 systems is of prime importance. Competing priorities will not go away anytime soon. IT managers will have to work hard to justify resource assignments for IPv6-related projects.

Personnel training is another challenging issue that anyone planning to deploy IPv6 should invest some time considering. It is especially important to have operational folks trained in IPv6 so that IPv6 can be supported as a *production* service.

10. Closing Remarks

In IPv6, changes, new features, and optimizations abound. But there is something else that sets it apart. There is a sense of community in the world of IPv6 that many in the world of IPv4 have not experienced. Everyone deploying IPv6 feels like part of a movement and part of history in the making. IPv6 is raw yet elegant, powerful yet nimble. The fact that it is still being developed means that it can be shaped into whatever the Internet community wants it to be.

IPv6 will clearly enable further growth of the Internet. The restoration of end-to-end connectivity will foster openness, simplicity, and innovation. Migration to IPv6 is not a Layer 3 or network problem. It's more of a Layer 7–9 problem: resource planning, vendor relationship/management, organizational buy-in, and the like. The networking vendors' implementations mostly work, but they do have bugs; we should not expect something to work just because it's declared to be supported.

Going forward, more and more enterprise and service provider organizations will continue to migrate parts or entire networks to the new protocol. Through the efforts of many, IPv6 network design will constantly evolve and improve over time.

At Google, around 99% of the engineers accessing our corporate network now have IPv6 to their desks and can access Google public services (search, Gmail, YouTube, etc.) over IPv6. This way, they can work on creating, testing, and improving IPv6-aware applications and Google products. At the same time, we are still doing a lot of work toward enabling IPv6 support across all the tools and applications used in the corporate network.

It will take time and effort before IPv4 can be turned off everywhere, but we are working hard toward that goal. Ultimately, we expect to successfully support employees working on an IPv6-only network in our enterprise.

References

[1] Free Pool of IPv4 Address Space Depleted: http://www.nro.net/news/ipv4-free-pool-depleted.

[2] Development of the Regional Internet Registry System: http://www.cisco.com/web/about/ac123/ac147/archived_issues/ipj_4-4/regional_internet_registries.html.

[3] Internet Registries IP Allocation Guidelines: http://tools.ietf.org/html/rfc2050.

[4] Internet Protocol Version 6 Address Space: http://www.iana.org/assignments/ipv6-address-space/ipv6-address-space.xml.

[5] IANA Policy for Allocation of IPv6 Blocks to Regional Internet Registries: http://www.icann.org/en/general/allocation-IPv6-rirs.htm.

[6] Criteria for initial allocation to ISPs: https://www.arin.net/policy/nrpm.html#six512.

[7] IPv6 Neighbor Discovery RFC 4861: http://tools.ietf.org/html/rfc4861.

[8] PMTUD for IPv6 RFC 1981: http://tools.ietf.org/html/rfc1981.

[9] IPv6 ICMPv6 RFC 4443: http://tools.ietf.org/html/rfc4443.

[10] IPv6 Node Information Queries: http://tools.ietf.org/html/rfc4620.

[11] IPv6 router advertisement options for DNS: http://tools.ietf.org/html/rfc6106.

[12] ICMP Router Discovery Messages: http://tools.ietf.org/html/rfc1256.

[13] RIPng for IPv6 IETF RFC: http://tools.ietf.org/html/rfc2080.

[14] Routing IPv6 with IS-IS IETF RFC: http://tools.ietf.org/html/rfc5308.

[15] OSPF for IPv6 IETF RFC: http://tools.ietf.org/html/rfc2740.

[16] OSPF version 2 IETF RFC: http://tools.ietf.org/rfc/rfc2328.

[17] MP-BGP IETF RFC: http://tools.ietf.org/rfc/rfc2328.txt.

[18] Cisco Configuring First-Hop Redundancy: http://www.cisco.com/en/US/docs/ios/ipv6/configuration/guide/ip6-fhrp.html#wp1055254.

[19] VRRP IETF RFC 3768: http://tools.ietf.org/html/rfc3768.

[20] VRRPv3 for IPv4 and IPv6, IETF RFC 5798: http://tools.ietf.org/html/rfc5798.

[21] Juniper JUNOS IPv6 Configuration Guide: http://www.juniper.net/techpubs/software/junos/junos53/swconfig53-ipv6/frameset.htm.

[22] IBM Data Center Networking: http://www.redbooks.ibm.com/redbooks/pdfs/sg247786.pdf.

[23] IPv6 unicast address assignment: http://tools.ietf.org/html/rfc5375.

[24] Stateless Address Auto-Configuration RFC: http://tools.ietf.org/html/rfc4862.

[25] DHCPv6 client RFC: http://www.ietf.org/rfc/rfc3315.txt.

[26] SLAAC Privacy Extensions: http://tools.ietf.org/html/rfc4941.

[27] Regional Internet Registry: http://www.iana.org/numbers/.

[28] Dual Stack technology: http://en.wikipedia.org/wiki/IPv6#Dual_IP_stack_implementation.

[29] 6-to-4 tunneling mechanisms: http://en.wikipedia.org/wiki/IPv6#Tunneling.

[30] Dual-Stack Lite technology: http://tools.ietf.org/html/draft-ietf-softwire-dual-stack-lite-05.

[31] Recommendations for Filtering ICMPv6 Messages in Firewalls: http://www.ietf.org/rfc/rfc4890.txt.

[32] See http://www.iana.org/assignments/ipv6-anycast-addresses/ipv6-anycast-addresses.xml.

[33] WCCP protocol RFC: http://www.ietf.org/rfc/rfc3040.txt.

[34] Policy Based Routing: http://tools.ietf.org/html/rfc1104.

About the Authors

Haythum Babiker is a principal architect in the Network Engineering team at Google, where he is involved in the network architecture and design of next-generation enterprise networks. His interests include, not just IPv6 network design and deployment, but also multicast, scalable and fault-tolerant backbone architecture, and WAN technologies. Prior to joining Google, Haythum was a Senior Engineer for Entertainment UK, where he was principally involved with the architecture and design of a large-scale enterprise network.

Irena Nikolova is a Network Engineer at Google. Her interests range from network management tools to development of advanced networking services like anycast and designing and deploying IPv6-only networks. Irena is currently earning a PhD degree in Distributed Systems and Networks at Sofia University, Bulgaria.

Kiran (KK) Chittimaneni is a Senior Network Engineer at Google and the Tech Lead for distributed office networking. He has been with Google for the past six years and has played a key role in building out the enterprise network. KK has played an active role in the world of IPv6, presenting at multiple leading network conferences evangelizing enterprise IPv6 adoption. KK holds an MS degree in Telecommunications and a BE in Electronics.