krockenmitte, photocase.com

Examining IPv6 on today's Internet

NEXT GENERATION

Is the world ready for the next generation Internet Protocol? We take a look at Linux with IPv6. **BY JÖRG FRITSCH AND PATRICK NEST**

The TCP/IP protocol, which began as an obscure experiment for a handful of academics and U.S. Department of Defense officials, suddenly became popular in the late 1980s and 1990s with the meteoric rise of the Internet. By the early 1990s, the IP address space – which had seemed quite vast in the early days – was beginning to look all too finite, and the experts began to wonder what would happen if the Internet ever ran out of addresses.

Work began on a new version of the Internet Protocol (IP) that would put an end to worries of overcrowding. A plan for the new protocol, which came to be known as "IP Next Generation" (IPng), was adopted by the Internet Engineering Task Force (IETF) in 1994, and the details for the IPv6 protocol were released through a flotilla of documents surrounding the RFC 2460 IPv6 specification.

The huge spaces within the 128-bit IPv6 address promised a nearly limitless supply of unique addresses, and IPv6 also provided other potential benefits with an assortment of new routing, security, and quality of service features. (See "IPv6 Benefits".)

The industry was poised for a transition to the new IP, but for various reasons, this great migration never really happened. New techniques, such as Network Address Translation (NAT) and Classless Internet Domain Routing (CIDR), staved off the end of the old IPv4 address space, and although hardware and software vendors implemented various forms of IPv6 support, Internet service providers were slower to adopt. Because the specifications ensure the compatibility of IPv6 with IPv4 environments, the next generation protocol has functioned more as a rarely used extension of IPv4 than as a separate environment with a full range of new features.

Recently, however, the situation has been changing [1]. The proliferation of Internet-ready mobile phones and other embedded devices raises new concerns about the viability of IPv4 address space. At the same time, the promise of sophisticated IPv6 quality of service capabilities offers potential benefits for future voice and video applications if developers will shift their focus to writing for the IPv6 environment.

In August 2007, the IETF published a draft version of a transition plan for migrating the Internet from "… a predominantly IPv4-based connectivity model to a predominantly IPv6-based connectivity model" [2]. According to the plan – which theoretically expires in February 2008 and may be updated

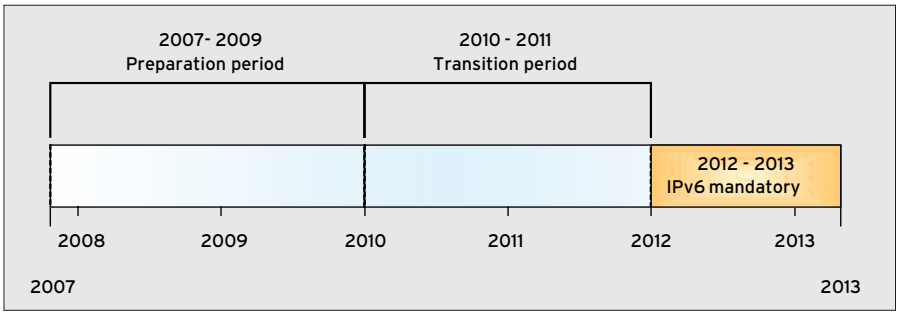

Figure 1: The IPv6 migration plan envisages final IPv6 migration between 2010 and 2011.

by the time you read this article – the transition will occur between 2010 and 2011 (Figure 1). At the end of this period, Internet providers will have to offer their customers IPv6, which will reduce the incentive for software vendors to focus development around the more limited functionality of IPv4.

Governments and political organizations are already starting to pay more attention to IPv6, and the first IPv6-only production networks are projected to come online by July 2008. Given the recent resurgence of interest, we decided the time was right for a look at the current state of IPv6 in Linux environments.

Practical IPv6

Most Linux distributions included IPv6 support. For some applications, you have to enable IPv6 in a configuration file. For example, the Bind name server typically needs an additional option, *listen-on-v6 { any; }*, in its *named.conf*.

For Sendmail, you need to modify the *sendmail.cf* file to tell both the client and the daemon to listen on a defined IPv6 address (enter the *inet6* address family in *ClientPortOptions* and *DaemonPortOptions*).

Linux hosts, as well as Mac OS X hosts, Windows Vista systems, and most open source applications, support IPv6 after the first boot. In production environments, many admins just ignore IPv6 and leave it running without any management or control. The result is an unmanaged TCP/IP protocol stack, which is a disaster from a security point of view. Even if you don't plan to roll out an IPv6-ready network implementation, it makes sense to come to terms with IPv6 so you can manage the services that might already be running on your network.

IPv6 means a lot of typing. The sheer length of IPv6 addresses makes them difficult to type and remember, despite the possible shortcuts (see the box titled "IPv6 Shortcuts"). To make IPv6 easier to work with, you should use name and directory services whenever you can. For your first steps in a trial environment, an */etc/hosts* file is probably the right choice, but you will quickly discover the virtues of an IPv6 name server that at least supports normal forward name resolution (name to IPv6 address).

IPv6 for Small Businesses

Just because the administrator can ping one host from another after setting up IPv6 does not mean that the network is ready for production use.

Many Internet HOWTOs recommend the use of ping after completing the IPv6 install to prove that all is well – but with reverse lookup disabled for the IPv6 address of the response package (*ping6 -n*).

In a production environment, it is obviously not enough to set up a working IPv6 topology. Businesses need at least a working name server or directory server, along with a web server, a mail server, and possibly a proxy cache and a Samba server to support heterogeneous environments.

To access the Internet with IPv6, you need the following:

- IPv6 connectivity via your ISP or, for developers and testers, via an IPv6 tunnel broker
- IPv6 Routing
- IPv6 DNS/Directory Services – forward, reverse lookup

Tunnel Brokers

The IPv6 specifications provide a means for encapsulating an IPv6 packet within an IPv4 packet. Several tunnel brokers support delivery of IPv6 packets over the IPv4 Internet through tunneling. Typically, a tunnel broker (RFC 3053) is the only practical solution for operating IPv6 networks on the Internet. Examples of tunnel brokers include Hurricane [3] and SixXS [4].

A tunnel broker sets up an IP-IP tunnel, which is also known as Generic Routing Encapsulation (GRE). The tunnel connects the two endpoints via a normal IPv4 network. This configuration creates virtual tunnel interfaces at both

IPv6 Benefits

The creators of IPv6 weren't just worried about the address space. IPv6 offers a number of additional benefits. Some of the new features are intended to address problems with IPv4, and others are simply an attempt to capitalize on new developments in the evolution of networking.

Other changes include:

- Autoconfiguration – IPv6 can be configured automatically through a system of ICMP-based router discovery messages. According to some reports, this feature could eventually replace DHCP.
- Multicasting – Multicasting, which was added as an afterthought to IPv4, is part of the IPv6 base specification. Multicasting lets you address a packet to a group of recipient addresses.
- Security – IPv6 includes native support for network-layer encryption and authentication, a feature that was eventually added to IPv4 through technologies such as IPsec.
- Payload – The payload of an IPv6 packet can be as large as 4GB – an astronomical increase over the 64KB payload capacity of an IPv4 packet. These "jumbograms" could result in increased efficiency and throughput over networks designed to accommodate them.
- Quality of Service – IPv6 provides a means for specifying the priority of a packet, which could lead to reduced latency for streaming video and other time-sensitive transmissions.

Of course, the IPv6 protocol primarily provides a networking environment; it is up to the applications on either end of the connection to use these new features effectively. Many of the best IPv6 features will not benefit the user until programmers start writing applications that leverage IPv6 enhancements.

endpoints; the interfaces are then configured as if they were two physical interfaces connected directly by wire.

The administrator can configure these interfaces with IPv6 addresses and use them as the default IPv6 route. The endpoint, which can be a single host or a router, appears as if it were wired directly to the rest of the native IPv6 world. This sounds complex, but depending on your choice of operating system, the configuration requires just six commands (see Listings 1 and 2).

Some tunnel brokers simplify IPv6 name resolution and reverse resolution in their Internet portals or offer the option of configuring Border Gateway Protocol (BGP) as the routing protocol.

Although a tunnel broker is easy to configure, the solution comes with the same costs all IP tunnels have, such as overhead because of the smaller MTU and related data transfer issues if path discovery does not work on all the nodes. The configuration also relies on the tunnel working correctly and being available. For operational networks, the tunnel broker option is probably not a good idea.

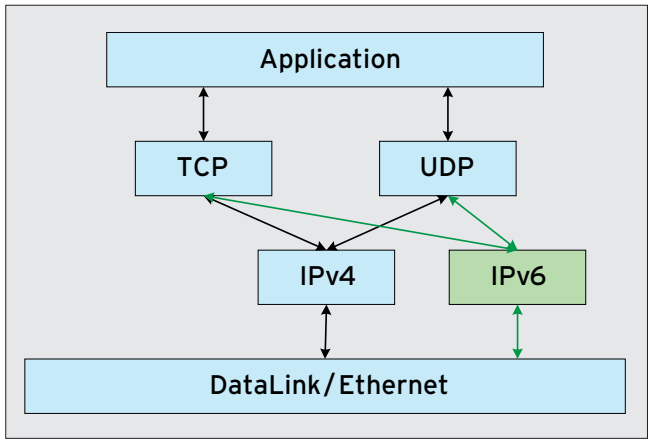

Figure 2: A dual-stack environment supports both IPv4 and IPv6.

The First IPv6 Request

After you set up the connection, it's time for an initial test. You can start by accessing an IPv6-capable website in your browser. The results of an ordinary browser session are typically quite sobering: When DNS name resolution returns an IPv4 A Record and an IPv6 AAAA record, all browsers use the IPv4 variant and request the IPv4 version of the HTML page. This is true of all applications that run in mixed IPv4/IPv6 environments.

Another problem you could face when browsing with IPv6 is that is that many of the IPv6-enabled sites have long disappeared. IPv6 link lists are often fairly ancient, and at least half of the links might not even exist.

The most reliable website we could find with respect to IPv6 support is the KAME project [5] (Figure 3). Some other sites offer IPv6 support, but IPv6 content is typically the same as the accompanying IPv4 content, so the benefits of browsing with IPv6 on today's Internet are limited.

Big websites such CNN or Google are not reachable via IPv6, although some sites are preparing for the transition. Google has reserved an IPv6 /20 subnet, and there is some speculation as to whether Google might be planning to offer ISP services in the future. Ebay was assigned a /41 subnet a couple of months ago.

DNS Obstacles

If you want to offer or access network services, name resolution is imperative. For your first experiments, you can start by using an */etc/hosts* file and relying on *nsswitch.conf* with the *files* option, which tells the system to search the hosts file first for name resolution. This

Automatic

Theoretically, an administrator can simply configure the router on a network with IPv6 interface identifiers (i.e., with EUI 64 addresses). Any clients that establish a connection to the network are automatically configured with an IPv6 address and router address (via Neighbor Discovery, ND, and Router Advertisements, RA). You don't need a DHCP server. This method is referred to as stateless autoconfiguration. Of course, an autoconfigured network without name service isn't much use because it doesn't support any kind of name resolution. The IPv6 address for the name server does not autoconfigure.

Various draft proposals have attempted to improve this, for example, by Router Advertisements or Anycast addresses (RFC 4339) to configure the DNS server. Thus far, none of these proposals has been implemented.

Even though the name server cannot be located if you rely on stateless autoconfiguration, it does not actually cause any problems in today's dual-stack environments (Figure 2) because each host has an IPv4 name server that can respond with IPv6 address records, if necessary. However, if this problem is not solved, it will eventually detract from the elegance of the IPv6 network. Generally, you can expect all servers to have static IP addresses and all clients to self-configure using stateless autoconfiguration.

IPv6 Shortcuts

IPv6 admins use two approaches for shortening the extremely long IPv6 addresses.

The first approach is to collate multiple leading zeros and just leave them out. Each IPv6 address comprises eight hexadecimal integers separated by colons. Assuming you have the number *:0090:*, you can abbreviate it to *:90:*, and if you just have zeros between two colons, *:0000:*, you can leave them out completely. This means that *2001:0000:0000:0090:00AD:0000:1234:abcd* becomes *2001::90:AD:0000:1234:abcd*.

To ensure uniqueness, the last group of zeros can't be abbreviated; otherwise it would be unclear how many zeros went in each *::* abbreviation space.

The second method is to define a constant prefix for your own network. In the previous example, the prefix could be *2001:0000:0000:0090::/60*. If your ISP gives you a subnet of /60, the prefix on your own network will never change, so you can define it in your applications and leave it out after doing so. The IPv6 name server administrator would just define this once, and you can then just work with the remaining four hexadecimal integers. The prefix need not be mentioned in internal network plans, in documentation, or in correspondence.

Right now, all native IPv6 addresses start with *2001:*. Previous IPv4 addresses converted to IPv6 start with *2002:*.

(The specifications define a way of calculating unique IPv6 addresses from IPv4 addresses.)

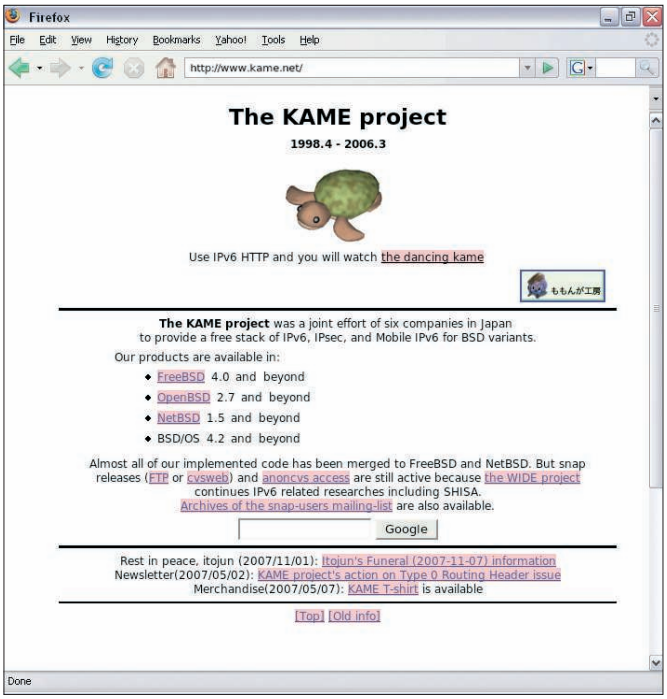

Figure 3: The KAME project offers reliable IPv6 support.

configuration works for IPv6, too. As with IPv4, you can add IPv6 host addresses and names to */etc/resolv.conf*. This solution will not scale, but it does save some typing and trouble.

At a minimum, you'll need to make three major changes to the name server configuration:

- *named.conf* (must bind to the IPv6 address of the network interface)
- zone file AAAA (must exist in the zone files for IPv6 hosts)
- reverse lookup file

AAAA records are the IPv6 counterpart to the A records used by IPv4. All other attributes, such as MX RR, CNAME, and so on, remain unchanged.

Expect some complications with configuring name resolution because, again, applications will look for IPv4 first. The "IPv6 Commands" box shows an overview of the tool options. Vendors are inconsistent with regard to the syntax of the IPv6 DNS information in the *nsswitch.conf* file. Red Hat lets the administrator keep the *dns* keyword for IPv6, whereas SUSE insists on *dns6*, and Solaris requires *ipnodes*.

IPAM

Currently, IPv6-capable ISPs in Europe typically assign /52 subnets to their customers. We would advise larger companies to apply for more addresses. In the United States, IPv6-capable ISPs are not as mean and have been known to assign /48 subnets to bigger customers. Converting these subnet numbers to absolute decimal numbers does not help much; the number of IPv6 addresses in a /64 subnet – for an ADSL router or home user – is greater than you can imagine.

Managing this many IPv6 addresses with a spreadsheet would be difficult. Instead, you should opt for an IP address management tool. Currently, we are only aware of commercial tools – for example, BT INS IPControl [6], BlueCat Networks [7], and Infoblox [8].

Network Equipment Vendors

The equipment supplied by most major network equipment vendors (e.g., Juniper and Cisco) has had IPv6 support for a couple of years, and you do not need to worry about switches or routers. However, there are some differences between major network equipment suppliers with respect to firewalls.

Cisco's ASA firewall only supports IPv6 at the command line, but the Juniper ISG Firewall can handle IPv6 addresses at the command line and in its browser-based GUI. Cisco supports the dual IPv4/IPv6 protocol stack in wireless networks. Other more specialized products, like load balancers (such as F5 and Nortel Alteon), also support IPv6, and often they have useful features for migrating from IPv4 to IPv6. We did not investigate the extent of IPv6 support for low-priced, consumer-grade hubs.

Listing 1: IP Tunnel with linux-route2

```
01 modprobe ipv6
02 ip tunnel add he-ipv6 mode sit remote 209.51.161.14 local
   83.84.117.191 ttl 255
03 ip link set he-ipv6 up
04 ip addr add 2001:470:1f06:12f::2/64 dev he-ipv6
05 ip route add ::/0 dev he-ipv6
06 ip -f inet6 addr
```

Listing 2: IP Tunnel with *BSD and OS X

```
01 ifconfig gif0 tunnel 83.84.117.191 209.51.161.14
02 ifconfig gif0 inet6 alias 2001:470:1f06:12f::2 2001:470:1f06:12f::1
   prefixlen 128
03 route -n add -inet6 default 2001:470:1f06:12f::1
```

Native IPv6

Right now, the typical approach to using IPv6 natively is a dual-stack implementation. You rent an IPv4 DSL connection or a leased line, and the ISP gives you a static /48 IPv6 subnet (equivalent to 65536 /64 subnets) via the same connection with dual stacking. Typically, the assigned IPv6 addresses are static, even if your IPv4 addresses are assigned by means of DHCP. IPv6 was advertised as a "service feature" with DSL four or five years ago, but some ISPs have stopped actively promoting IPv6.

In the past three years, it has become increasingly difficult to use IPv6 on your own network.

Native IPv6 without IPv4 connectivity does exist; however, it is very rarely offered, typically only in the U.S. and Asia. After connecting your operational network to the IPv6-only Internet, you can't even exchange email with the rest of the world. Again, if you try to establish a monoculture (either IPv4 or IPv6), you will need to invest a fair amount of time permanently disabling the other protocol stack throughout your network.

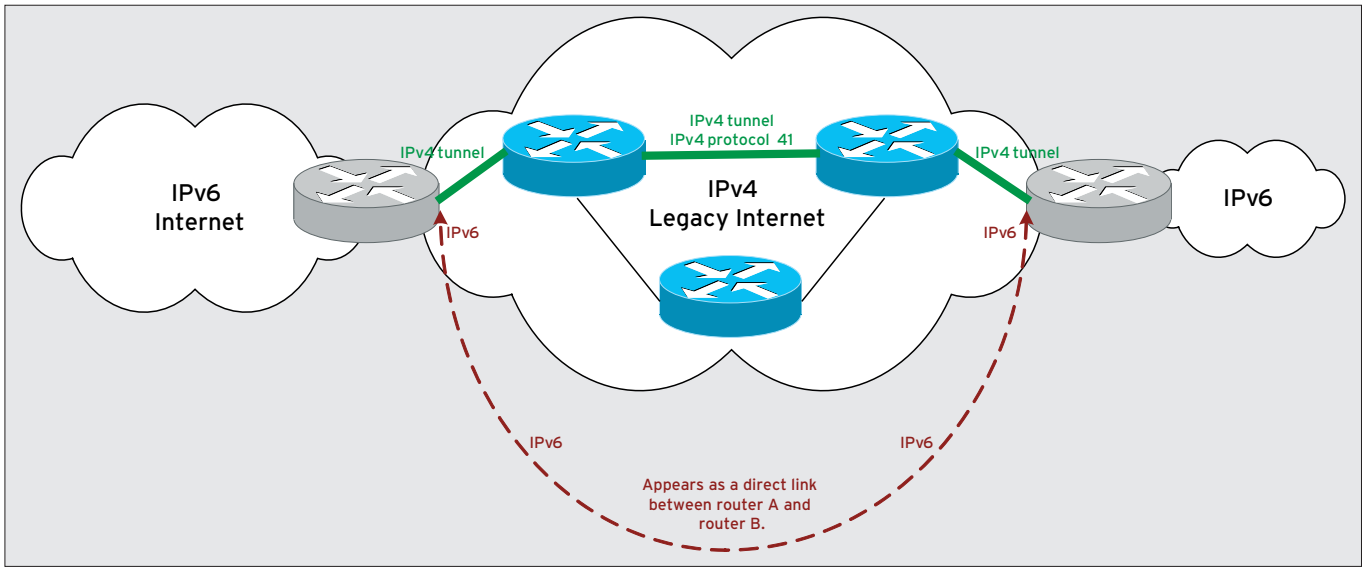

Figure 4: Tunnel brokers provide tunneled access to an IPv6 network via IPv4.

Professional administrators should check to see whether new hardware they intend to purchase (e.g., proxy caches, mail servers, and so on) supports IPv6. Although IPv6 should not be a required criterion for new acquisitions, it makes sense to know what difficulties you'll face when migrating your own infrastructure to IPv6.

Security: Fear of the Unknown

Native IPv6 environments are hard to implement. Because of the affinity of all operating systems and applications to IPv4, it is difficult to imagine permanently disabling IPv4. Living with a dual stack for the next couple of years makes far more sense. IPv4 is unlikely to disappear soon.

Basically, you can say that all IPv4 security settings are meaningless in IPv6 and vice versa. Administrators have to manage firewall rules for both worlds separately – as if the neighboring world does not even exist. This parallel configuration is easier to handle than you might think. Because IPv6 is not very widespread, a firewall with just a couple of rules and an IPv6 clean-up rule as a catch-all will do the trick, and the same thing applies to access lists on routers. But you have to be bold enough to trust the IPv6 implementation on your firewall device just as much as you trust the IPv4 implementation.

Gaps in the Puzzle

Currently, the future of IPv6 is full of gaps. The biggest problem is the lack of commitment on the part of Internet service providers, which makes the use of IP tunnels (tunnel brokers) inevitable (Figure 4).

Draft proposals for DNS support in auto configuration have been around for years, but they remain drafts. And IPv6 support from major network device suppliers is still uneven. In our lab, devices from two major vendors had trouble with the ICMPv6 Neighbor Discovery Protocol (counterpart to ARP).

Although these issues have been fixed, customers looking for support often hear things like, "You're the first one to ever ask me that."

All told, these trivial issues won't keep IPv6 down for long. You can find a detailed list of IPv6-capable applications on the Internet [9].

Conclusions

Although most network equipment and open source applications implement IPv6, the "next-generation Internet" is nothing more than a neat experiment right now because of unresolved issues and ISP inertia. As of this writing, it is still impossible to build mission-critical services on IPv6.

Still, IPv6 is a neat toy if you want to demonstrate your skills and carve a swath at the cutting edge of the network universe. ■

IPv6 Commands

For troubleshooting purposes, you'll need a tool that reveals routes, open connections, and ports. *netstat* does not understand IPv6 unless you insist on it with the *-A inet6* option. For example, *netstat -A inet6 -rn* displays your routes. The *-n* switch suppresses reverse lookup so that you will not wait forever for a response if IPv6 reverse lookup is broken anywhere on the path.

The IPv6 equivalent to *ping* is called *ping6*; again, it is a good idea to disable reverse lookup by specifying *-n*.

The *arp* command does not exist for IPv6. The Neighbor Discovery Protocol replaces ARP. Some distributions and *nixes use the *ndp* command instead, whereas others give you *ip* command options to display the same information, such as *ip -6 neigh*.

INFO

[1] Government Computer News: *http://www.gcn.com/IPv6*

[2] IPv6 migration plan: *http://www.ietf.org/internet-drafts/draft-jcurran-v6transitionplan-01.txt*

[3] Hurricane Electric Tunnel Broker: *http://tunnelbroker.net*

[4] SixXS Tunnel Broker: *http://www.sixxs.net*

[5] KAME.net: *http://www.kame.net*

[6] BT Diamond IP: *http://bt.ins.com/software*

[7] BlueCat Networks: *http://www.bluecatnetworks.com*

[8] Infoblox: *http://www.infoblox.com*

[9] IPv6-capable open source applications: *http://www.deepspace6.net/docs/ipv6_status_page_apps.html*