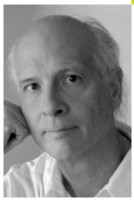
RIK FARROW

musings



rik@usenix.org

I HAVE BEEN TO THE BRAIN GYM RE-cently and really gotten a workout. Perhaps you've heard all the buzz about how you need to "exercise" your brain if you want to stay sharp as you get older. Although many products out there purport to do this, all I need to do is leap into learning something new. Stretching my brain is tiring, but stimulating.

While I was attending LISA '07 in Dallas, I wandered into the BoF run by two gents from ARIN. Mark Kosters, CTO of ARIN, was talking to an almost empty room about the need to start the migration to IPv6. Granted, this was late in the evening after a great reception (mechanical bull riding, armadillo racing, and free drinks), but I found myself feeling sorry for these earnest folks who were largely being ignored. I decided to invite Mark to write an article about the depletion of IPv4 address blocks and to dig deeper into the issue myself.

Immediately I found other things to do. Some were brain gym–like forays into weird, alien landscapes, such as setting up iptables within Xen 3.0, with bridges and unreal and virtual network interfaces. Others didn't stretch my brain at all, because they were familiar tasks.

I procrastinated until I came up against the wall of a firm deadline: this issue of *;login:* was going to be published without my column. Faced with a final deadline, I finally cracked the books and the Web, and got serious about IPv6.

The Next Generation

IPv6 goes way back, almost to the dawn of the Internet. Well, not the real dawn, but to 1994, the beginning of public awareness of the Internet. There had already been rumblings about the fast depletion of IPv4 addresses, and the number of Internet hosts was growing at double digit rates *every month*. IPv6 was designed not simply to create a humongous address space, which it does, but also to provide a more flexible framework which will support new services such as mobility, autoconfiguration, and improved security.

IPsec has already been integrated into IPv4, and thus security is a less interesting reason for migrating to IPv6. Autoconfiguration is much more interesting, as is the possibility of getting huge amounts of routable address space in IPv6. No more fighting with RIRs (Regional Internet Registries) to get a scrawny /24; register now, and get more host addresses than even Google currently needs.

I won't attempt to repeat the arguments that Kosters makes in his article. IPv4 address blocks are vanishing rapidly, and that will make it more difficult for you to get the IPv4 address space you or your organization needs. I also believe it will lead to, at the least, a gray market in IPv4 addresses, as hoarded address space gets auctioned off. Seems silly to me to get involved with another sordid affair, with domain squatters replaced with v4 address brokers, when an almost unlimited number of IPv6 addresses are available.

Instead, I'd like to point you in the direction of some resources, as well as to amuse you with my own attempts to use IPv6.

Transition

If you travel back in time far enough or are really an Internet pioneer, you will know of the original flag day. On that day (January 1, 1983), the Network Control Protocol (NCP) could no longer be used within ARPANET and TCP/IP was the only acceptable protocol. Now, imagine for a moment making a similar transition from IPv4 to IPv6.

Okay, that's long enough. We really don't want to go there, and neither did the designers of IPv6. They provided a number of transition mechanisms, including dual-stack hosts and routers and various forms of tunneling. I found a couple of books [1, 2], chose the smaller one, started reading, and quickly learned how I could start using IPv6.

Of the three methods that looked relatively easy, 6to4 tunneling interested me the most. Teredo, a method of tunneling IPv6 packets within UDP packets, has the disadvantage that its main purpose is to make IPv6 accessible to people using NAT or behind stateful firewalls. Teredo uses relay servers, one type for encapsulation and another for both registration and getting clients to set up state to the relay servers. Teredo sets up globally routable IPv6 addresses for systems behind NAT or firewalls. Microsoft has added this capability to Vista, and it can be added to XP. If you are a control freak, like firewalls, or are merely paranoid, you may wish to block this behavior [3].

Then there are tunnel brokers, organizations such as www.sixxs.net, which will match you up with tunnel providers if you are an ISP, and even set up your own tunnel right from your PC. I found myself a bit wary of this approach as well, but would have gone this route if my ISP still was set up to use this.

6to4 tunneling, on the other hand, is something you can do yourself as long as you have a public IPv4 address that you can use, and a Linux or BSD system handy. I plugged a laptop loaded with Ubuntu into the hub outside my firewall, gave it a static IP address, started hacking away . . .

And ran into problems immediately. There really isn't a lot of info about configuring Linux systems for 6to4 on the Net, and even less about debugging your setup. 6to4 tunnels IPv6 packets within IPv4 packets using protocol 41. Like Teredo, this system relies on public relays, but they work quite differently. One set of relays advertises a route to 2002::/16, and these routers convert IPv6 packets destined for your 6to4 tunnel to IPv4 packets destined for the IPv4 address of your tunnel. The other set of relays consists of systems advertising 192.88.99.1/32, an anycast route (see the February 2008 *;login:* article about anycasting). These systems convert the IPv4 packets you send into IPv6 packets and forward them onto the IPv6-capable Internet.

You do need to learn something about IPv6 addresses to work with 6to4, but not a lot. Your 6to4 IPv6 address consists of the 2002::/16 prefix and your IPv4 address converted into base 16, something you can easily do with a few lines of Perl (the Net::IP module does the work) or even bash shell scripting:

```
IPV4=your.address.here PARTS=`echo $IPV4 | tr . ' '`
printf "%02x%02x:%02x%02x\n" $PARTS
```

Then you follow the instructions for setting up the 6to4 tunnel for any recent Linux or BSD variant [4]. So I followed the instructions, tried ping6 www.kame.net (KAME is the group responsible for the BSD implementation of IPv6), and waited for the results—and waited, and waited.

Perhaps the anycast route to 192.88.99.1 doesn't work, I thought. I tried traceroute 192.88.99.1. This stopped before reaching the relay server—blocked by an ACL, I suppose. I asked people on other networks to try this as well. I found a couple that worked (Qwest and Sprint networks) and several that didn't (including AT&T, my upstream provider). I also noticed that some routes terminated in Europe.

But perhaps these ISPs are just blocking traceroute. Maybe I had other problems. I did a lookup on www.kame.net, and it turned out that my own ISP doesn't return the AAAA records used for IPv6 addresses. My internal DNS server does, so I just typed in one of the addresses for KAME: 2001:200:0:8000::42. Still not working, and watching tcpdump output showed me that even though ping6 claimed to be sending out packets, I sure couldn't see them.

I was convinced that I had done something wrong with the tunnel or the interface it was using. Linux kernels, like most IPv6 implementations, will automatically assign link-local addresses, beginning with fe80::, to interfaces, and I thought this might be the problem. But IPv6 allows you to assign multiple addresses to each interface, so eth0 with more than one address is not the problem.

Finally, I noticed that I had misentered the command that creates the tunnel. I had carefully converted my IPv4 address into hex, then mistyped that hex when using the ip command, sigh. Once I got that working, ping6 to KAME worked, and I could ping6 a 6to4 tunnel router, 2002:c058:6301::1, as well. Success!

The Future

Obviously, my exercise would have been a lot simpler if my own ISP offered IPv6, but it doesn't. It doesn't even support AAAA records in its DNS server.

I asked Vint Cerf, a big supporter of IPv6, when Google would start advertising AAAA records for its servers. Cerf said that hosts trying to reach Google using IPv6 might not get access because they live in a disconnected IPv6 island, but that Google is working with people on this issue.

There are loads of other issues as well. Dave Piscitello wrote a report for ICANN about support for IPv6 in commercial firewalls, as well as writing an article about it for this issue of *;login:*. The answer at this point is that open source software currently has better support for IPv6 firewalling. Your Linux systems have ip6tables, Mac OS X has ip6fw, and so on. But if your organization relies on a commercial firewall product, support is sketchy.

Besides, if tunnels are available, will you ever have to move to IPv6? I believe that you will, and the sooner you start learning about IPv6, the better it will be for you. Not just avoiding the panic of a personal flag day, when you hear that management has decided you will transition next week, but also the advantage you can personally gain by becoming familiar with technology that is going to be getting a lot more important in the near future.

Just imagine a future where most people carry around computers that are constantly in contact with the Internet. Oh, yeah, that's right, a good percentage of cell phone users already are carrying around Internet connected computers. In the US, most of these cell-phone users essentially use provider-controlled tunnels. But in China and other parts of the world, cell phones get fully routable IPv6 addresses (there is nothing like RFC1918 private address space in IPv6). There are already more cell phones in the world than IPv4 addresses. Will cell phone users want to tunnel IPv6 over IPv4 to reach your Web site?

Other than the growth of new IPv6 users, there has yet to be an IPv6 killer app. But given the issues with tunneling, as IPv6 users increase a new Internet divide, between the old and the new Internet, might arise.

I suggest taking advantage of the access you already have to computers and network devices that are IPv6-enabled, and learn now, while you are still ahead of the game.

The Lineup

I've already mentioned two articles, the first by Mark Kosters and Megan Kruse of ARIN. NAT (private network addresses: RFC 1918 [5]) and CIDR (Class InterDomain Routing, RFC 1519) have allowed us to cruise along using IPv4 without tremendous pain. And while early projections of IPv4 address depletion had us running out of addresses in 1996, today's projections are a lot more convincing. Kosters and Kruse not only discuss the dire danger, but also tell us more about getting IPv6 addresses.

Dave Piscitello, himself a networking pioneer, had mentioned to me that he had done some research on support for IPv6 in commercial firewalls. I tried some polite armtwisting, with the result that Dave has written a complete description of his research project. The news could be better, but it will get better only when customers start asking for more IPv6 support from firewall vendors.

Next up, Octave Orgeron finishes his series on working with Solaris 10 LDoms. LDoms are interesting even if you don't run Solaris and have the right hardware, because they point the way to future systems with hardware hypervisor support.

Matthew Sacks shares his experiences with working with Linux and VMware. Sacks had encountered problems with VMs crashing because they ran out of memory. He and his co-workers report on their solution here.

Aditya Sood next explains WiFi security. Sood explains its weaknesses and offers suggestions for better WiFi security.

Michael McCool then writes about the issues in achieving high performance on hardware that supports parallel execution. McCool begins by describing the various CPU features that support parallelism, starting with the obvious ones such as multicore processors. But he goes much deeper than that, in the first of several articles we hope to publish about parallel programming.

Filling out the magazine, we have our regular columnists, but no summaries. Strangely enough, no one seems interested in attending conferences or workshops over Christmas vacation, and even shortly thereafter, so this issue of *;login:* is a bit shorter than usual. Have no fear: the next issue will include conference summaries from FAST '08 and the 2008 Linux Storage & Filesystem Workshop.

REFERENCES

[1] Niall Murphy and David Malone, *IPv6 Network Administration* (O'Reilly, 2005).

[2] Marc Blanchet, *Migrating to IPv6: A Practical Guide for Mobile and Fixed Networks* (Wiley and Sons, 2005).

[3] Teredo security considerations: http://en.wikipedia.org/wiki/Teredo_tunneling#Security_considerations; http://www.microsoft.com/technet/network/ipv6/ipv6_teredo.mspx.

[4] Setup of 6to4: http://www.getipv6.info/index.php/Linux_or_BSD_6to4_Relays; http://tldp.org/HOWTO/Linux+IPv6-HOWTO/configuring-ipv6to4-tunnels.html.

[5] RFC 1918, address allocations for private networks: http://www.faqs.org/rfcs/rfc1918.html.