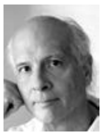
OPINION

Musings

RIK FARROW



Rik is the editor of *;login:*.
rik@usenix.org

While I was at LISA '11, I ran into an old friend, Paul Ebersman. Paul was one of the first employees at UUNET, and every time I ran into Paul, usually at a USENIX conference, he would tell me how the UUNET office had grown onto another floor of a building. Eventually, UUNETs growth was taking over entire buildings. Paul would also tell me how fast the Internet was growing, with traffic doubling every few months. UUNET was the fastest-growing ISP in the 1990s [1] and was founded with help from USENIX.

These days, Paul is working for a company (Infoblox) that makes server appliances, and his own focus is on IPv6. Paul's article about issues with making the transition to IPv6 appears in this issue.

During our conversations at LISA, I asked Paul about security and IPv6, and Paul said, "It's like 1994 all over again." What Paul meant was that back in the mid-'90s, organizations were beginning to use the Internet without having more than the vaguest notion of security. Also, most IPv4 software stacks were largely untested, leading to root compromises and various denial-of-service (DoS) attacks, such as the Ping of Death [2]. In many ways, as the momentum to enable IPv6 on Internet-facing Web sites rolls onward, we will be facing another moment where most people will not be familiar with the new security issues that come with using a new network protocol.

Just the Same

Some things won't be any different: there are still 65535 TCP and 65535 UDP ports in IPv6, just as in IPv4. If your client or server application has a bug that can be exploited using IPv4, it most likely can also be exploited via IPv6 (once you have connectivity). But there may also be bugs, such as the PoD, lurking in relatively untested IPv6 stacks.

IPv6 brings with it an enormous address space, with billions of network addresses and four billion times four billion host addresses. Some people had hopes that this enormous address space would make automatic scanning for hosts unfeasible. Steve Bellovin, Bill Cheswick, and Angelos Keromytis explained in their 2006 *;login:* article [3] that there are a number of strategies worms could use to limit their attacks to existing networks. These same strategies can also be used by attackers when exploring networks they wish to attack.

Paul said that it is an excellent idea to monitor your existing networks to check for IPv6 traffic that might already be there. I tried this, both on my home network,and looking at traffic outside my firewall, and found a fair amount of IPv6 traffic. I used tcpdump with the ip6 filter (Wireshark uses the same input filter) and discovered that Macs and newer Windows systems are both pretty chatty. Both types of systems would occasionally send out Router Solicitation ICMPv6 packets, and both also looked for plug-and-play neighbors. Macs use MDNS (Multicast DNS), Windows boxes are using SSDP (Simple Service Discovery Protocol), and both are sending out IPv6 multicast packets looking for or advertising services. If you've been on a public wireless network and noticed other people's shared disks showing up in a Finder or Explorer window, you have seen these protocols at play. And while these protocols are very helpful in some cases, I consider them harmful when used on non-private networks (hotels, coffee shops, airplanes, conferences, etc.). Nothing like an invitation to be hacked, and no scanning looking for victims required. The victims advertise their presence and potentially vulnerable services.

Most modern operating systems support IPv6 and will automatically bring up a link-local IPv6 interface on all network interfaces. Link-local addresses begin with the prefix fe80::/10 and, as the name suggests, are valid only on the local subnet. Link-local packets cannot be routed and are important to IPv6; they are used in network discovery protocol and for address assignment and DHCPv6 (see Paul's article). The host portion of these IPv6 addresses (EUI-64) is constructed using each interface's MAC (Media Access Control) address, with the 2 bytes FFFE inserted into the middle of these 6 bytes, and the 7th bit set to one [5]. Unless you are using DHCPv6, your systems will use the MAC address when automatically generating IPv6 addresses—unless you have a Windows 7 or newer OS.

Windows 7 systems use a randomly generated host suffix that changes every 15 minutes for privacy, since including your MAC address in a globally accessible IPv6 address makes your system easy to track on the Internet, as well as possibly identifying your system, since the first 3 bytes of the MAC address are assigned to vendors [5]. David Barrera wrote about privacy extensions for IPv6 address for *;login:* in 2011 [6], and perhaps the rest of the mobile system world—Linux, Mac, and BSD laptops, smartphones, and tablets—may catch up to Microsoft's ability to maintain more privacy when using IPv6.

IPv6 was designed so that we can use globally unique addresses, and this rightly implies the end of NAT. There are available address ranges (site-local addresses [7]), such as link-local addresses, that will not be routed over the Internet. But having IPv6 allows you to do away with the problems that NAT has caused, such as having to renumber networks when organizations merge or when once separate networks are joined. This also implies that your firewall had better be IPv6-enabled [8] and that you block packets destined for internal services at edges of your network. NAT has provided some protection against clients being visible more by accident than by design, but with IPv6, this limited protection goes away.

Negative NAT

Because you already have systems (Windows and Macs) on your network bursting with eagerness to use IPv6 (just look at those router solicitations!), you are already using IPv6—just inadvertently instead of intentionally. An attacker who manages to gain the ability to run a router advertisement daemon such as radvd [9] can announce whatever IPv6 prefix and routes the attacker desires. This allows the attacker to perform man-in-the-middle (MITM) attacks by routing packets through a system the attacker controls, while forwarding the packets using NAT-

PT [10] to convert them to IPv4 over your existing networks. This attack works just as well in public networks and will work whenever a Windows or Mac system decides to use IPv6 instead IPv4 (for example, as soon as an IPv4 DNS request times out on a Mac).

IPv6 brings with it a large share of annoyances, primarily the very long and awkward 128-bit addresses. It is one thing to memorize an IPv4 numeric address, expressible as four numbers, and another to memorize the IPv6 address for the same network interface. I found a nice little tool called ipv6calc [11] you can install to help you to interpret IPv6 addresses:

```
$ ipv6calc --showinfo fe80::21b:fcff:fefb:742c/64
No input type specified, try autodetection...found type: ipv6addr
No output type specified, try autodetection...found type: ipv6addr
Address type: unicast, link-local
Registry for address: reserved
Interface identifier: 021b:fcff:fefb:742c
EUI-48/MAC address: 00:1b:fc:fb:74:2c
MAC is a global unique one
MAC is an unicast one
$ []
```

In this example, I fed it the autoconfigured IPv6 address for a Linux server, and ipv6calc did a nice job of explaining it.

Firewalls will be another annoyance. In the BSD world, you can have a single set of firewall rules for both IPv4 and IPv6. But in the Linux world you have two sets of rules, one for each protocol, and you must remember that any changes you make to one set of rules you must also make in the other.

If you look at some default rules in a Linux ip6tables firewall file, you will see rules allowing all ICMPv6 packets. IPv6 relies on ICMPv6 for determining MTU, and fragmentation will not work without it. Network discovery, like the router advertisements, also rely on this, but this is something you may want to block at your firewall [12].

Older Linux kernels (pre-2.6.20) do not support stateful filtering of IPv6 packets (CONFIG_NF_CONNTRACK_IPV6) and leave a gaping hole (ports 32,768–61,000) in the ip6tables rules so that client packets can return through the firewall:

```
ACCEPT udp anywhere anywhere udp dpts:filenet-tms:61000

ACCEPT tcp anywhere anywhere tcp dpts:filenet-tms:61000

         flags:!SYN,RST,ACK/SYN
```

This charming behavior reminds me of old Cisco router packet filtering (ACL) rules, back in the days before stateful packet filtering.

Another wonderful feature of IPv6, since deprecated, is the routing header (RFC 5095). Routing header 0 (RH0) was included in IPv6 to support mobile devices so that they can roam between networks and continue to receive response packets. It also allows the construction of what we used to call source routing, a trick that used to work wonders with ancient protocols that trusted the source address for authentication. Today, RH0 headers can be used in amplification DoS attacks.

The Lineup

We start off this issue with an article about dark silicon by Nikos Hardavellas. I've read posts by people who consider dark silicon to be evil, but Nikos makes an excellent case for why we will be seeing dark silicon in future CPU designs.

Paul Ebersman is up next, with the article I've already mentioned about IPv6. Did you know that June 6, 2012, is World IPv6 day [13]? And that someday your IPv4-only servers will not be reachable by IPv6-only devices?

Simson Garfinkel approached me in December, wondering if we could publish an article about Unicode. To be honest, I wasn't excited about character encoding at first, but as I read his article, I discovered lots of things (and many annoyances) that I could finally recognize as Unicode artifices. I learned that my desktop display uses UTF-8, and the FFFE byte string I had been deleting from the beginning of text files was actually an indicator of the type of Unicode character encoding found in the file I was editing (UTF-16, little endian, mysteriously included in IPv6 EUI-64). If you write programs, you need to know about Unicode.

While Simson covers C, Java, and Python, Tobi Oetiker discusses Unicode for Perl hackers. Tobi tells us that there has been Perl support for Unicode for many years. More importantly, he explains where you must actively write Perl code that properly encodes and decodes Unicode, with examples, as well as some surprises you may encounter when dealing with Perl's support for Unicode.

Doug Hughes approached me during LISA with an article idea. Doug, like many of us today, works in an organization that manages huge amounts of data that, naturally, must not be corrupted. But Doug had encountered silent data corruption—that is, errors not reported by disk drives. Doug explains how he discovered this data and provides suggestions for how you can do this as well.

David Blank-Edelman takes us on a journey beyond Perl's core functions for handling error messages. If you have ever used Perl, you will be familiar with warn() and die(), but David shows us much more helpful error functions—that is, ones that provide more useful debugging information or allow you to catch errors the way other programming languages often do.

Dave Beazley also steps outside the bounds of Python, exploring add-on Python libraries. Python comes well-equipped with libraries, but Dave suggests two (currently) non-standard ones that extend regular expression (regex) and Web interaction (requests), the first with new features, the second with both more features and easier programming. I do want to note that I was unable to install regex on my older CentOS (5.7) system.

Dave Josephsen is still waxing enthusiastic about the features of Graphite, part of a suite of monitoring software. In this article Dave shows us some neat tricks for creating graphs using the URL interface to Graphite.

Robert Ferrell, having heard about Nikos's article, decided to create his own spin on dark silicon and other things considered "dark" today.

Many books are reviewed in this issue, including two by a new book reviewer, Mark Lamourine, one by Trey Darley, one by Brandon Ching (back after a long break), and five by Elizabeth Zwicky. Elizabeth covers intro to the command line and Python, while Mark covers two programming-related topics. Trey volunteered to read and review the massive second edition of Richard Stevens's *TCP/IP Illustrated,* and Brandon Ching takes a look at an intro to HTML 5.

This issue includes reports from LISA '11. We did not manage to cover all 34 sessions of LISA, but we covered most sessions. We also have a report from the Advanced Topics Workshop at LISA and one from CHIMIT, an ACM workshop on computer human interfaces as they apply to system administrators.

Now that you know that your network is already (hopefully) supporting IPv6 to a limited extent, perhaps it is time to learn, and do, more about IPv6. One of the best papers at LISA covered how Google approached adding IPv6 support, first internally, then externally [14]. Learning about and using IPv6 is the best way not to be surprised by attacks against your networks that utilize this protocol.

At the very least, you should be monitoring IPv6 traffic on your networks. As far as switches are concerned, IPv6 is just another Layer 3 protocol, and routers and firewalls are the only real barriers that prevent you from having a dark network, one that you are unaware of. Don't forget: it's 1994 all over again.

References

[1] UUNET: https://en.wikipedia.org/wiki/Uunet.

[2] The Ping of Death (PoD): https://en.wikipedia.org/wiki/Ping_of_death.

[3] S. Bellovin et al., "Worm Propagation Strategies in an IPv6 Internet," *;login:*, vol. 31, no. 1, USENIX: https://www.usenix.org/publications/login/february-2006-volume-31-number-1/worm-propagation-strategies-ipv6-internet.

[4] Link-local addresses: https://en.wikipedia.org/wiki/Link-local_address#IPv6.

[5] IPv6 Extended Unique Identifier, 64 bit: http://wiki.nil.com/IPv6_EUI-64_interface_addressing.

[6] D. Barrera et al., "Back to the Future: Revisiting IPv6 Privacy Extensions," *;login:*, vol. 36, no. 1, USENIX:https://www.usenix.org/publications/login/february-2011-volume-36-number-1/back-future-revisiting-ipv6-privacy-extensions.

[7] Unique local address: https://en.wikipedia.org/wiki/Site-local_address.

[8] See the April 2008 issue of ;login: for two articles that discuss IPv6 firewalls: https://www.usenix.org/publications/login/april-2008-volume-33-number-2.

[9] Router advertisement daemon (radvd): http://www.litech.org/radvd/.

[10] IPv6 MITM attack using NAT-PT: http://resources.infosecinstitute.com/slaac-attack/.

[11] ipv6calc homepage: http://www.deepspace6.net/projects/ipv6calc.html.

[12] IETF, "Recommendations for Filtering ICMPv6 Messages in Firewalls," May 2007: http://www.ietf.org/rfc/rfc4890.txt.

[13] Internet Society, World IPv6 Day: http://www.worldipv6day.org/.

[14] Haythum Babiker et al., "Deploying IPv6 in the Google Enterprise Network: Lessons Learned" (Practice & Experience Report): https://www.usenix.org/conference/lisa11/deploying-ipv6-google-enterprise-network-lessons-learned-practice-experience.