;login:

THE MAGAZINE OF USENIX & SAGE
April 2003 • volume 28 • number 2

inside:

USENIX & SAGE
The Advanced Computing Systems Association &
The System Administrators Guild

musings

by Rik Farrow

Rik Farrow provides UNIX and Internet security consulting and training. He is the author of *UNIX System Security* and *System Administrator's Guide to System V*.

rik@spirit.com

Perhaps the mystery of the trojaned open source code has been solved. In my February column, I discussed the trojaned configure scripts that would connect to a fixed IP address, and exec a shell if there was any response. On January 22, CERT published an advisory (*http://www.cert.org/advisories/CA-2003-02.html*) about a double free() bug in CVS servers that allowed an attacker to execute code as root, requiring no more than read-only access to start with.

We have no way of knowing if this will be the only bug ever discovered in CVS or on other services that run on mirrors used to distribute open source software. But it sure makes having signed distributions seem a lot more important. Of course, that precludes having enough signers on your own key ring that you can trust the public key used to sign the code.

Other great events have occurred. The world's fastest spreading and shortest lived worm appeared about 0530 UTC on January 25, late Friday night in the US, where it appeared to have been launched. Sapphire, or Slapper, attacked unpatched versions of Microsoft SQL Server and MSDE. As usual, a patch had been published by Microsoft months before the worm was released. But even Microsoft got the infection, and was still fighting it on Monday morning. Bank of America and Imperial Bank of Commerce ATMs went down on Saturday, and some other well-known organizations (AMEX, A.C. Nielsen, Price Waterhouse, and Citicorp) were hit hard as well.

Perhaps these organizations should not be blamed for not patching their Windows boxes, as Microsoft had put out a later hot fix for MS SQL that would have regressed the DLL involved, ssnetlib.dll, after the initial security patch was posted. Or perhaps the sysadmins involved had no idea they had an open network service listening at a UDP port attached to MS SQL Server. Keep in mind that many applications embedded MSDE, a developer-only version of MS SQL, so that someone running McAfee Centralized Virus Admin, Veritas Backup Exec, or ISS RealSecure 7.0 and Scanner were also vulnerable. ISS, a security company, included a vulnerable version of MSDE at least as late as September 2002, two months after the patch had been announced, and never reported that their own software was vulnerable to the attack.

Sapphire was the fastest spreading worm yet seen. A paper describing Sapphire's spread (*http://www.caida.org/outreach/papers/2003/sapphire/sapphire.html*) explains why. In brief, Sapphire was fast because it used short UDP packets (404 bytes, including headers) to attack, so no TCP handshaking, which caused Code Red to spread much more slowly. The number of Sapphire-infected systems doubled every 8.5 seconds, reaching 90% of all vulnerable hosts in 10 minutes. If you had read the Paxton paper presented at last summer's USENIX Security Conference (*http://www.icir.org/vern/papers/cdc-usenix-sec02/*), Sapphire might not have seemed such a surprise.

The main effect of Sapphire was denial of service. Some networks became unreachable within minutes of the start of the attack. But many ISPs acted quickly to block packets going to port 1434/UDP, which is, after all, not a required or necessary service. The effects on the greater Internet were soon damped down, and almost back to normal within 14 hours: *http://www.matrixnetsystems.com/ea/2003/20030125_packetloss.jsp*.

I found Sapphire interesting because it so easily penetrated "protected" networks. No firewall maintainer in his or her right mind allows arbitrary UDP packets to enter from the Internet. So how did Sapphire get inside financial institutions, Microsoft, and many other sites? The answer includes VPNs, connections to home systems, and various routing leaks.

Books

The notion of various undocumented connections to networks reminds me of one of the books I have been reading lately. The venerable *Firewalls and Internet Security* (Cheswick, Bellovin, and Rubin, Addison-Wesley, 2003, ISBN 0-201-6346-6) has finally come out in a second edition. I had quit bugging Ches about this, feeling it was a futile gesture, but am glad to see the project completed at last.

Firewalls used to be my favorite security book, and still comes close (it now has to compete with Ross Anderson's *Security Engineering*). The first edition (from 1994) didn't even mention HTTP, so the book really needed revision. The new book contains new chapters, including a short one on Web services, IDS, and Network Layout, as well as updates to most other materials. Occasionally, I would find something that seemed very out-of-date, like the mention on page 58 that most sniffers are discovered when a disk fills up or that filtering with routers entails a performance penalty, but that the Internet is "connected by (at best) a DS1 (T1) line (1.544 Mb/sec)."

The writing in *Firewalls* is terse. You will find information about the most popular network services and the issues involved in passing them through firewalls, as well as the related risks. An "evening with Berferd" survives, and there is also a new section about a different attack, complete with crude forensics used to examine Clark, the ULTRIX system in question. While I wouldn't recommend this book to firewall "newbies" (I *don't* want them attempting to rip out the IP forwarding routines in their kernel), I do recommend it to people who must run firewalls and who already understand the difference between an application gateway and a circuit relay.

Matt Bishop also has a new book out: *Computer Security, Art and Science* (Addison-Wesley, 2003, ISBN 0-201-44099-7). Matt wrote his book as a textbook for advanced undergraduates and graduate students of computer science. In other words, you can learn a lot about security using this book in addition to using it to teach security, but it is not a book for people wanting to improve the way they maintain system or network security. Perhaps getting Matt to teach a couple of semesters to the programmers who write our network servers and authentication code might be a good idea, though. And if your goal is to really learn about computer security, Matt's book is clear, easy to read, thorough, and includes exercises as well as telling you which chapters to skip if your math skills are rusty (or perhaps never at the level of following formal proofs).

I was fortunate enough to get to tech edit Mick Bauer's *Building Secure Servers with Linux* (O'Reilly, 2002, ISBN 0-596-00217-3). That forced me to read his book in excruciating detail, which he later claimed he appreciated. *Building* explains securing a Linux server at a very practical level. There is a section that talks about risk analysis at the beginning, but then the book gets very nuts-and-bolts, with examples of firewalling, OS hardening, using SSH, OpenSSL, securing DNS, SMTP, Web servers, FTP servers, handling logs, and some simple IDS techniques.

I didn't just read Mick's book, I used his instructions to do things that I had put off, like putting BIND in jail. Note that I did this on a BSDi box, not a Linux box. I believe that most of the configuration examples will work just as well on any OS where a server package is supported (except Windows), so this book can have a broader audience than just Linux users. If you find you need something that goes beyond what comes as documentation for some open source security package, check out Mick's book. The writing is clear, it's well organized, and provides enough detail for you to install and configure the services it covers.

And, finally, you can read a book about *BGP* (van Beijnum, O'Reilly, 2002, ISBN 0-596-00254-8). You might wonder why I would read a book about BGP besides curiosity, but there are important intersections between BGP and security. Van Beijnum does a good job explaining issues like peering and how that affects a site that wants or needs multiple connections to the Internet for reliability. Filtering is also vitally important to the proper functioning of BGP (and in being a good neighbor) and is covered, but not in its own chapter.

The Finale

The Internet survived yet another attack. While Sapphire disrupted local connectivity, and access to many sites became difficult, the Internet as a whole continued running. And the quick efforts to dampen the packet flooding caused by Sapphire, estimated at 55 million packets per second at its peak (CAIDA paper), were accomplished by the night-shift personnel at ISPs, not by government organizations.

Sapphire could have been much worse, though not in the sense that it could have killed or maimed people. In a bit of cheerful news in an otherwise depressing time, Marcus Sachs (director of communications infrastructure protection in the Office of Cyberspace Security in Washington, DC) said in a response to Sapphire, "There was no lasting damage done to the infrastructure. We'd like to see the term cyber-terror dropped." Yes, someone in the US government who understands that terrorism involves "something that physically kills people" (his words).

But all Sapphire did was invade and attack. The SQL module involved, ssnetlib.dll, runs with Local System privileges. Had the author been a bit less frugal with his or her code, he or she could have included a timer that stopped the worm from scanning about 15 minutes after it was released. A second timer could then have counted down the seconds until some other activity began, whether it was a different DoS (like Code Red flooding the IP address for whitehouse.gov) or perhaps something a bit nastier, not as subtle as disk formatting, but some other form of data destruction. Or Sapphire could have downloaded and installed DDoS agents. A relatively unheard-of trojan named Leaves had been installed on over 20,000 systems last summer and communicated via IRC. The author of Leaves, a British man, was caught before he used his network of agents for anything.

The Internet has proven to be remarkably resilient. The security of Internet-connected systems, on the other hand, has been proven to be a rare beast, although it probably does exist in a minority of locations. Perhaps more people need to follow the advice of Cheswick, Bellovin, and Rubin (install bulkhead firewalls internally) and of Mick Bauer (harden those servers!).