;login:

THE MAGAZINE OF USENIX & SAGE
November 2000 • volume 25 • number 7

THEME ISSUE: SECURITY

edited by Rik Farrow

inside:

IN THIS ISSUE

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

in this issue

by Rik Farrow
Theme Issue Editor

<*rik@spirit.com*>

For the ninth time, USENIX and its members created a security symposium. In the past, two years had to pass between conferences. Security was not such a big deal. But no longer. Chairs will always say that this conference is the best ever, and it seems this time that it was true. Wonderful papers, marvelous speakers.

Win Treese put together the Invited Talks track. If you ever want to do something that is nontechnical and really challenging, you should help put together an Invited track. Treese went so far as to add a very human element to the conference by bringing in Suelette Dreyfus to speak on Cryptography and Human Rights. Dave Dittrich scared us all by reminding us that Distributed Denial of Service attacks have not stopped – they just don't make the news these days. Duncan Campbell talked about Echelon for the conspiracy buffs in attendance (and there were quite a few), and Mudge explained how a tool written by the l0pht (now @stake) goes about detecting network interfaces listening promiscuously.

You can find all of the papers online, of course, but I wanted to mention a few. Publius is an interesting effort to permit anonymous and irrevocable publication of sensitive documents. The detecting-backdoors pair of papers (one of which won Best Student Paper) examine a way to detect backdoors and relays using network heuristics and headers only, so that it is not necessary to sniff the data portion of packets, which might be encrypted anyway. You should read the actual papers if you are interested, as well as the summaries in this issue, to get a good idea of the papers, Invited Talks, and a couple of BoFs.

I do not mean to slight any of the paper writers, as I found them excellent reading, especially while sitting in the Denver airport during hours of thunderstorms, waiting for United to allow their planes to approach the passenger tunnels. Isn't it amazing just how fragile our technology base is?

This issue of *;login:* contains feature articles that I solicited from the security community. I especially wanted a better understanding of DNSSec. It is one thing to read the RFCs and quite another to talk to someone, in this case Evi Nemeth, who has played with implementations, written a chapter in a book, and had BIND 9 implementors edit the article you will find here.

I asked David Brumley and Steve Romig if they would contribute again. David already had an idea in mind, the technique that he is using to help secure Linux systems at Stanford University (through the creation of a secured distribution so that people can at least start right). Steve has been working in computer investigations for years and will teach a tutorial at the USENIX conference in San Diego in early December. Steve writes about collating logs and understanding how they are used as evidence.

The Nessus team in France contributed an article about their vulnerability scanner. These guys have put together an open source tool with a large collection of vulnerabilities (no exploits, kiddies), and their tool ranked number one in Fyodor's survey of the best security tools. (ISS ranked seventh, not bad for one of three commercial products mentioned, but pretty poor when you consider that Nessus is free.) Fyodor's list, at *<http://www.insecure.org/tools.html>*, is a nice reference for security tools (50 of them, although the link for VeteScan was broken when I checked it).

Mudge contributed an article about testing supposed secure devices, which dovetails very nicely with Peter Guttman's paper on building a secure open source device for

crypto-functions. I'll have more to say on that later. Sven Dietrich wrote about his own experiences working with Distributed Denial of Service software through his involvement in intrusion detection. And Carole Fennelly interviewed the conference's keynote speaker, Blaine Burnham, helping to explain exactly what Burnham plans to do in the future.

Burnham's speech struck several chords for me. There was the part about goatheads – very nasty, low-growing weeds endemic to the Southwest and the scourge of bicyclists. These weeds produce pretty little flowers, which turn into vicious spiked seeds quite capable of flattening any bike tire, as well as getting stuck in car tires (which is why they are found alongside roadways in many places).

Burnham used the goathead as an analogy. Bicycle riders have learned to take countermeasures (or have flat tires daily during the mid- to late summer), such as extra-thick tires, a plastic internal guard, or (in my case) green goo, anti-freeze mixed with fibers, that fills small holes quite nicely. The problem is, software vendors have yet to figure out about the green goo. When an exploit is discovered for NT or a CGI script, there is nothing that will automatically fix the problem. You, and your system, are history.

Burnham said that this is because no one is writing secure programs. Sure, everyone puts out patches, but that is hardly the proper solution when you need your server running, or when your company has become front-page news. Writing secure software will certainly help. But we have several years' experience with well-known problems with buggy code, for example buffer overflows, and yet buffer overflows are still uncovered at an alarming rate. (One day on BugTraq, a URL was posted for a site named LSD that had exploit code for 20 vulnerabilities alone.) Marcus Ranum has said before that writing secure code is not easy. And he has personal experience of this, not just because he was brave enough to teach a class in writing secure code, but also because his own code fell victim.

Was this the event that forever embittered Ranum? Just kidding, but I am guessing that it came close. Ranum had written large parts of the Firewall Toolkit (fwtk), only to fall prey to a buffer-overflow attack. The attack was not in his code, but in the use of the syslog() subroutine library called by his code for logging. More recently, WU-FTPD fell prey to problems involving logging using sprintf(), and setproctitle() also allegedly had a buffer overflow.

Burnham suggested that instead of just patching programs and trying to write secure code, we actually run secure operating systems. This idea dates back to research in the '60s and '70s, with ideas like the rings in MULTICS, or the concept of a security monitor. Not that we don't use rings in our operating systems today – just not well. For example, the Intel processor has four rings, but only two are used (see John Scott Robin's paper). UNIX, Linux, and NT systems all run OS code in the innermost, privileged ring, and everything else in ring 4 (or an outer ring on other processors). The problem with this is that an operating system, especially one where you can install drivers and loadable modules, is much too large to secure.

Instead, the core of an operating system should be the security monitor. This is very similar to a real microkernel system with a focus on security. (There are research versions of this design out there.) But the designs have so far proven to be too slow, and getting new device drivers for them is a serious problem. Still, I have written about this idea several times in the many years I have contributed to *;login:*, and years before that when I wrote for *UNIXWorld*. Our operating systems must be secure before we can expect our systems to be secure.

;login:

EDITORIAL STAFF

THEME ISSUE EDITOR
Rik Farrow <*rik@spirit.com*>

EDITORS
Tina Darmohray <*tmd@sage.org*>
Rob Kolstad <*kolstad@usenix.org*>

STANDARDS REPORT EDITOR
David Blackwood <*dave@usenix.org*>

MANAGING EDITOR
Jane-Ellen Long <*jel@usenix.org*>

COPY EDITOR
Eileen Cohen

TYPESETTER
Festina Lente

MEMBERSHIP AND PUBLICATIONS
USENIX Association
2560 Ninth Street, Suite 215
Berkeley, CA 94710
Phone: +1 510 528 8649
FAX: +1 510 548 5738
Email: <*office@usenix.org*>
WWW: <*http://www.usenix.org*>

This is not an unsolvable problem – just a very difficult one. One solution may be to build special hardware, perhaps a multiprocessing design where one processor handles device drivers, while another runs the secure OS, perhaps with virtual machines running insecure OSs above it. In this design the driver processor would not have access to system memory, and would rely on the security monitor for transferring data and commands to and from the driver processor and the main processor and memory. It could be done. The question is when.

I want to end this bit of musing by mentioning full disclosure. Full disclosure may disappear. That is, new vulnerabilities will not be announced, only software patches that might relate to security problems. This is exactly where we were six years ago, when some UNIX vendors (as well as a very large non-UNIX vendor) *never* posted information about security problems. They just didn't talk about it.

Today, we are at the opposite extreme. For example, on Labor Day, several different security vendors and teams announced that they had discovered serious problems (read, root compromise) via the locale mechanisms in glibc, right after several OS-distribution vendors announced patches for the problem. But the announcement was not simultaneous, so very large vendors, like Sun and HP, did not have their patches ready yet. PR through bug announcements is the current trend, and if the unruly mob doesn't learn some manners, we may soon find it gagged by law (or lawsuits).

The posting of complete, packaged exploits is another issue. On the one hand, I really appreciate having code to read, as that helps with my understanding of a problem (in UNIX at least – forget about the Win32 API). Unfortunately, it also helps the hoards of script kiddies, who will start trying to hump, er, exploit, every system with the appropriate port open, even if the architecture does not match.

I really do not want to see full disclosure go away. What I would like to see is some moderation, moderation that appears to be forthcoming from groups like SecurityFocus. SecurityFocus evolved out of Scott Chasin's BugTraq mailing list, which began after Brent Chapman got really upset when Chasin posted a sendmail root exploit to the old firewalls mailing list back in 1994. Chasin's posting was in response to a CERT advisory about sendmail that was so vague as to leave everyone wondering what the problem with sendmail might be. Chasin's post turned out to have nothing to do with the CERT advisory. (Read my article at <*http://www.spirit.com/Network/net0800.txt*> to learn more about this.)

Having enough information to determine that your systems are exploitable is good. Having a thousand script kiddies beating down your door is bad (very annoying, especially if you ask the Pentagon).

With that note, I wish you all secure operating systems, and a merry good year.