;login:
THE MAGAZINE OF USENIX & SAGE
February 2002 • Volume 27 • Number 1

inside:

SECURITY MUSINGS
by Rik Farrow

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

Not so very long ago, I took two weeks off. I traveled down the Colorado River, through the Grand Canyon, by raft. No email, no phones, no pagers, and the only news we heard were football scores.

One afternoon, one of the guides asked me what I thought was the most pivotal event in the history of computing. I thought about this briefly, then answered, "Solitaire. If Microsoft hadn't added Solitaire to Windows 3.1, Windows would never have caught on." People learned how to use the mouse and a bit about the "desktop metaphor" by playing Solitaire. Without this game, a crummy windowing interface over an unimpressive operating system would never have survived.

Rather than wander off into yet another Windows-bashing column (how boring!), I'd like to imagine what the world might be like if things had worked out differently. Imagine desktops and servers that could be patched simply by updating a single server. All the patches would include digital signatures, so the clients could check the patches for authenticity before applying them. User accounts would be centrally managed, and a user could log into any desktop system and find his or her home directory and environment waiting to be used.

Of course, people who used the Apollo Domain OS got all this – along with some pretty quirky stuff (see *http://www.citi.umich.edu/apollo-archive/* for a picture of an Apollo workstation). Apollo was not at all open source, and its windowing system had the odd feature that your input always appeared at the bottom of the window where the focus was, even during "full screen editing." Sun killed off Apollo, with its much better security features. Note that you could get X Windows for the Apollo, so you could actually see your input where you thought it should be.

To be honest, I don't really know if the Apollo Domain OS security was as good as it seemed. And the software update/patch system worked because all the hardware came from the same vendor. My only experience working on an Apollo involved getting it to work on a UNIX network with TCP/IP and NFS, something that the Apollo support person was not at all happy about. Why did I want to use NFS when their network file serving was more advanced and much more secure?

In retrospect, Apollo Domain OS looks pretty good. Another thing that still looks interesting is Inferno, a successor to Plan 9 (*http://www.vitanuova.com/inferno/index.html*). I liked several things about Plan 9, especially the notion of having file servers and diskless desktops. Take away those pesky disks from desktop users and do away with viruses, trojans, and other malware. Have some real control over your users (no games at work!), as well as having central file stores that are easy to back up. Plan 9 and Inferno do have improved security, as well as a programming language, Limbo, that, like Java, does not allow buffer overflows and is portable across CPU architectures. The old failing of Inferno was that it was not open source, but that has changed as well.

Inferno uses the Plan 9 concept of a hierarchical namespace to provide security. Basically, if some resource is not part of an application's namespace, the application will not have access to that resource. Public key cryptography is used for authentication as well as for setting up signed and/or encrypted links between networked systems.

Might Inferno work as a desktop OS? Perhaps, but with one very big problem that I'll get to later. It appears to me that one could safely read email without having to worry about the mail tool covertly installing or executing software, or setting up a network

connection to some remote system (Inferno considers the network stack to be a resource).

Linux and BSD have a different control mechanism that might also work. Type enforcement takes traditional ACLs, like those used in Windows 2000, to a finer degree of control. Instead of using an ACL to govern which users can have a particular type of access to an object, type enforcement considers the application used in making the access control decision. For example, user joe may write to /etc/shadow while using the passwd command. Any other attempt to modify /etc/shadow will fail for user joe. A mail tool could be prevented from writing anywhere on the hard drive except for mail folders. And anything stored by the mail tool could also be prevented from having any additional access, so that an executable delivered by email would not be allowed to do any damage.

UNIX and Linux systems do have a capability, called change root, for running a process with limited view of the local file systems. There have been exploits for breaking out of a change root environment, making this technique, while better, still not what I have in mind. FreeBSD (*http://www.freebsd.org*) includes an enhanced change root, called a jail, that provides even more security. Within a change-rooted jail, the superuser cannot use raw sockets, change the IP address given to the jail, or make device files. And the FreeBSD jail call prevents the call from succeeding if any file descriptors pointing to directories outside the new root are open. But instructions for setting up the jail (*http://docs.freebsd.org/44doc/papers/jail/jail-4.html*) call for a complete installation under the change root. Essentially, the jail is a virtual machine with a few features disabled – not what I am looking for either.

There is a really big problem with using Inferno, Plan 9, Linux, or BSD applications. Right now, people are accustomed to using Microsoft Office products as well as Internet Explorer. These applications do not run under the OSes I am interested in. When they run under Windows 2000 or XP, they are apt to misbehave, install trojans, set up back channels, etc. But as long as people insist on using these familiar applications, we won't get anywhere.

Can Microsoft create an operating system where these applications could run securely? Microsoft could graft in type enforcement, for example, but they would still be left with an operating system of ungainly size. They could whittle this down by migrating non-operating system libraries to user space, and then perhaps have a provably secure operating system. At that point, people could use the user-level libraries, properly licensed of course, to run their favorite Microsoft application, under the operating system of their choice (provided it runs on an Intel processor and supplies the same basic set of system calls).

Some people might argue that Windows 2000/XP already has sufficient security. Really? Remember Code Red as an interesting example of a failure to correctly control what IIS, the Web server, could do. A better example appeared recently, when people discovered that if they installed a popular Microsoft game, they could only play it if they were logged in as a user in the administrators group. Ooops.

And will people accept type enforcement as part of their Windows operating system? Unlikely, as it will surely cause slower performance when someone is playing a game.

I'm rambling, and I apologize. But when I hear Richard Clark, the new "cyberterrorism" czar (what cyberterrorism?) say that it is important to our nation's security that

. . . people discovered that if they installed a popular Microsoft game, they could only play it if they were logged in as a user in the administrators group. Ooops.

cable modem and DSL users install personal firewalls, I just want to scream. Why not have them install a simple and secure operating system, one that can play games, run basic business apps, browse the Web, and read email? What good will it do to provide a kludge (the personal firewall) in an attempt to fix something that is badly broken. Why not fix what is broken?

I heard Richard Clark speak at a dinner paid for by Microsoft. I was invited to Microsoft's "Trusted Computing Conference" and attended in the vain hope of participating in a discussion about vulnerability disclosure. Scott Culp had published "It's Time to End Information Anarchy" (*http://www.microsoft.com/technet/treeview/default.asp?url=/technet/columns/security/noarch.asp*) a couple of weeks previously, a treatise that lays the blame for Microsoft's security woes on people who disclose vulnerabilities. Culp did not speak during the conference. Instead, Chris Klaus, of ISS, took the hard-line approach. Essentially, only vendors and select insiders would ever get detailed information about any vulnerability. This should sound familiar, as it was how things worked until about five years ago.

Chris Wysopal (*cwysopal@atstake.com*) took the middle road. Wysopal explained that while working with Steven Christie (*cooley@linus.mitre.org*) on adding new entries to the Common Vulnerabilities and Exposure (CVE) database (*http://www.cve.mitre.org/news/inthenews99.html*), they had both realized that there were no true standards for the handling of vulnerability disclosure. True, there are several example policies for doing this, such as RFPolicy (*http://www.wiretrip.net/rfp/policy.html*), but no actual standards. As vulnerabilities often affect the Internet, they came with the idea of writing an RFC, an Internet standard document, covering vulnerability disclosure.

Actually, what Wysopal suggested was writing two RFCs. One RFC would cover the behavior of both the disclosing organization or individual and the vendor: for example, the timelines for responsibly disclosing to the public information about a vulnerability, for a vendor to respond to information about a bug or problem, or for the vendor to produce a patch. The RFC would also suggest conventions that vendors should follow, such as an email address of "security" as a contact point, as well as a Web page (*http://www.microsoft.com/security* is a good example). The second RFC would discuss standards for the content of a vulnerability disclosure statement and would actually be the more difficult of the two RFCs to write. When I wrote this column, a draft of a single RFC appeared at *http://jis.mit.edu/pipermail/saag/2001q4/000358.html*.

The thing to remember about vulnerability disclosure is that back when there was no public vulnerability disclosure (just the selected insiders, and, of course, all the hackers who exchange this information as well), many vendors pretended not to have security problems. If a vendor did patch a security hole, the patch was rolled into a much larger patch without announcement. And, quite likely, without many people installing the patch either. I don't think we want to go back there.

While I would like to turn back the clock and see something quite unlike Windows take over the desktop market, we have to work with what we have. Someone is sure to remind me of StarOffice, and that there are alternatives to Internet Explorer. Sure there are, but why are my Apache logs full of hits from IE and almost devoid of visits from other Web browsers?

The Microsoft settlement might include sharing some of Microsoft's proprietary information. If Microsoft exposed the Windows libraries, so that other operating systems could compete on an equal basis, this would at least provide a bridge that just might result in people being willing to use other, potentially much more secure, operating systems.

And in that case, I guess they could keep their games on their desktops.

USENIX Needs You

People often ask how they can contribute to the USENIX organization. Here is a list of needs for which USENIX hopes to find volunteers (some contributions reap not only the rewards of fame and the good feeling of having helped but also a slight honorarium). Each issue we hope to have a list of openings and opportunities.

- The ;login: staff seeks good writers (and readers!) who would like to write reviews of books on topics of interest to our membership. Write to peter@matrix.net.
- The *;login:* editors seek interesting individuals for interviews. Please submit your ideas to *login@usenix.org*.
- *;login:* is seeking attendees of non-USENIX conferences who can write lucid conference summaries. Contact Tina Darmohray, <*tmd@usenix.org*> for eligibility and remuneration info. Conferences of interest include (but are not limited to): Interop, Internet World, Comdex, CES, SOSP, Ottawa Linux Symposium, O'Reilly Open Source Conference, Blackhat (multiple venues), SANS, and IEEE networking conferences. Financial assistance to cover expenses may be available. Contact *login@usenix.org*.
- *;login:* always needs conference summarizers for USENIX conferences too! Contact Alain Hénon *ah@usenix.org* if you'd like to help.
- The *;login:* staff seeks columnists for:
  - Large site issues (Giga-LISA),
  - Hardware technology (e.g., the future of rotating storage)
  - General technology (e.g., the new triple-wide plasma screens, quantum computing, printing, portable computing)
  - Paradigms that work for you (PDAs, RCS vs. CVS, using laptops during commutes, how you store voluminous mail, file organization, policies of all sorts)

Contact *login@usenix.org.*