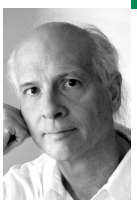
RIK FARROW

musings



Rik is the Editor of *;login:*.

rik@usenix.org

I'VE DECIDED TO USE THIS COLUMN TO defend the ordinary person—certainly a monumental task, one requiring volumes instead of a couple of pages. Yet I believe I can make a dent in the project by focusing on just one group: the part of the human population that does not include most USENIX members or other computer security professionals and CS researchers.

The days when I spent a large part of my life standing in front of MIS and IT folk attempting to explain Internet security are long past, but they have left me with a strong feeling about the people who run both the public and the private computer and network infrastructures in North America. Keep in mind that I was either teaching classes or lecturing at conferences that focus on bringing in business and government IT people, I can say that understanding computer security is a black art for most of these people.

There, I've said it. Not having to stand in front of such an audience again will hopefully protect me from being stoned to death. But the very people in charge of administering our all-important cyber-infrastructure are largely clueless about what really matters. (N.B.: I use the adjective "cyber," even though I loathe it, as it has become popular.) I do not mean this as an attack on anyone's intelligence: if it was easy to get this stuff right, we wouldn't continue to have security problems. After all, the idea behind malware goes back to NSA research in the '70s, and viruses became popular in the late '80s—20 years ago.

Now let's broaden the potential lack of clue a bit. I suggest that most people who use computers and similar networked devices such as cell phones know just as little about computer security as, and likely less than, the managers of our cyber-industry.

All of this should appear blindingly obvious. Instead, I often hear things such as "The best AV product resides in the cerebral cortex" from buddies with a real clue working in security. To those of us who live in the parallel reality where security is easy, we rarely have security problems because we pay attention to the activities that get people into trouble, and we avoid those activities.

Windows

Note that avoiding such activities often includes avoiding the use of Windows. You might wonder

how I could possibly know that, not unreasonably. The answer has to do with email headers, and just what you can see when your preferred mail tool is called Mail. While there are a few Windows users, I see many more "X-Mailer: Apple Mail" and even "User-Agent: Opera Mail/9.64 (Linux)" lines than versions of Outlook in email from my security acquaintances. But modified behavior goes far beyond simply being part of the low-hanging fruit, something you become when you use the world's most attacked software base.

As an example, I'd like to share a recent experience. When a friend visited me, he asked if he could attach his Windows notebook to my network. I said, "Sure, let me set you up outside my network but attached to the Internet." My friend wondered about this, but when I asked him about the status of his AV software, he said his license had expired some time ago. I explained that his Windows systems were surely full of malware by now, and that appeared to end the discussion.

Several days later, I get a call from the friend asking me if I knew about any good, free AV software. I explained that there is no such thing as free AV (ignoring ClamAV for such a user), but explained that he could try MS's Malicious Software Removal tool for free [1]. That tool can remove malware that is currently recognized, unless his system is already being controlled by something like Conficker, malware that prevents access to Microsoft and any AV vendor through its control of the Windows DNS client.

I can only assume that my friend's computer was indeed owned, as he soon resorted to installing some "free" AV software on his notebook. You, my reader, can already guess what happened next. My friend had installed *scareware* on his system, leaving it more infected than ever. As Bill Cheswick once described his dad's computer, my friend's computer was now "spewing blue smoke all over the Internet," to the point that my friend could tell "something was wrong." He asked me if he needed to reinstall Windows, and I told him that it was the next-best thing to do. The best thing for him to do, as he had bought his notebook used and had no install CD, was to install Linux. He could then safely recover his backup files from the USB sticks he was using, as they were likely to be infected as well (another Conficker trait [2]).

My point is not that my friend is stupid. He's actually intelligent and very successful in his field. It is just that his field is not computer security. He wants to use his computer in the same way he uses a car: he gets in, starts it up, and drives off. He probably had about four hours of formal training in driving as well as in the rituals that everyone obeys for the most part, such as driving on a particular side of the road. That's it.

But for someone to use a computer securely, they need to be versed in both security and system maintenance, in particular a patching regimen for both the operating system and any installed software. Imagine for a moment that your car would steal your identity if you forgot to update the firmware in the third-party stereo system. That's exactly where we are today, as even Microsoft agrees [3].

Actually, Microsoft is blaming applications for most of their security problems. And for the second half of 2008, this appears to be true. They also state that Vista is more secure than Windows XP, which also appears to be true. Looking at Microsoft Security Bulletins for the first half of 2008, most were for Windows applications, only one was unique to Vista, and four OS patches didn't apply to Vista at all. But of the 30 bulletins I looked at, 12 did.

Microsoft's malware scrubber reports on what it finds, so they can state with certainty that the malware infection rate on Vista is 60.6% less than that of Windows XP SP3 [4]. Somehow that number leaves me unimpressed. Sure, Vista is more secure than XP, but it is still getting infected with recognizable malware at an alarming rate, implied by the 60.6% number. Do you really want to use a computer that is infected with less malware? How about no malware instead?

Parallel Paths

I need to change tracks for a bit, and talk instead about the future of computing. In the June 2009 issue I wrote about some of the differences between SMP and cluster designs. In this issue you will find two articles explicitly about taking advantage of the massive parallelism that's starting to appear in processor design. If you read these two articles, I believe they will help you further understand how working with highly parallel systems requires changes in how we program. Note that Pete Galvin's column also focuses on parallelism, as it applies to using Sun's Niagara-based systems.

Many-core systems are the future of processor architecture, and we can see that systems will require great changes in how they are programmed if we are to realize the potential benefits. What I keep hoping is that while these changes are taking place, the architects of both the hardware and the supporting software will think about security right from the beginning.

I have written and spoken many times about the failure of our current systems when it comes to security. Designing new systems presents a rare opportunity to design in security from the start instead of attempting to add it later, which isthe usual approach. Adding security later works poorly, as I have already mentioned in this column.

Systems such as HiStar [5] actively encourage the use of hardware [6] to build secure systems from the ground up. HiStar and Flume both use information flow control, where data itself is labeled and these labels control which entities can send or receive the data. I really like this concept, as our current security models have a granularity based on users and files, subjects and objects, where the real issues today are for security of individual users, whether that user is running a Web browser or a Web server. In each case we want data from different sources to be isolated, and only merged or shared under controlled circumstances. Ownership of data at the user level is a flawed model, and our current security failures should make this blindingly clear.

The Lineup

This issue starts off with two articles that look at user-level issues. Michael Vrable et al. write about a project that uses the Cloud, in particular Amazon's S3, to store backups. Vrable's software makes intelligent use of minimal Cloud resources to provide full and incremental off-site backups. And he has made this software, Cumulus, available for use.

Switching gears, Leo Meyerovich writes about his experiences with parallelizing browser code. Leo points out that power-limited devices, such as cell phones, will be taking advantage of manycore CPUs and that this can only work when code has been written specifically for parallelism. Leo provides a table of simple experimental results, comparing Safari on a laptop to Safari in the iPhone when using the same WiFi network. His comparison proves that the iPhone's Web page rendering is slow because of its processor, not

the network. Leo goes on to explain how designs for power-limited devices can improve performances through design decisions made across three axes.

Tim Kaldewey has written a thorough explanation of GPU programming. Tim began working with GPUs before the CUDA API made that task easier, and he contrasts programming before and after CUDA. Tim also explains the current downsides—largely bus and memory issues—to using manycore GPUs.

Dave Dittrich provides us with a survey of attack techniques. Dave has had a front row seat, starting with attacks on systems at the University of Washington in the late '90s. He has had ample opportunity to witness how attacks have advanced over the years, including changes that make malware more likely to be installed, yet more difficult to reverse-engineer.

Rudi van Drunen continues his series on hardware by showing how to build your own Stratum 1 time server using inexpensive hardware. Rudi demonstrates a bit of hardware hacking on a Soekris single-board computer that can increase the accuracy of a GPS-timesource by a factor of 1000, then explains how to build and install a FreeBSD firmware package that completes the project.

David Blank-Edelman begins a two-part series on using CGI::Application to build a simple Web application. David chose this Perl module because it is simple to learn and use, yet provides the state required for his example application.

Peter Galvin explains how to tell if an application will run well on Sun's Niagara-based systems. Niagara systems have multiple threads per core, and many cores as well, and these work very well to hide memory latency and provide great throughput. But if the target application does not use a lot of parallelism in its design and implementation, all of this hardware remains underused and performance suffers. Pete provides both tips and pointers to tools to determine if your applications will do well on Niagara.

Dave Josephsen takes a careful look at what happens when open source projects go commercial. I believe his cautionary tale will be familiar to many readers, as he writes about a Zimbra installation.

Robert Ferrell has written a parable about security that speaks for itself (or perhaps for Robert).

We have many great book reviews in this issue, as well as reports for NSDI, IPTPS, HotPar, and HotOS. Both the HotOS and HotPar reports include a lot of the discussion among participants, bringing these workshop reports alive.

I honestly try not to write about the failure of security too often, as I don't want to sound like a broken record, that is, a pre-Internet storage device where audio was recorded on spiral tracks on cheap vinyl media. After all, there are some bright sides to the current state of security. Enterprising criminals have succeeded in using the enormous amount of wasted desktop cycles to make money. Read Brian Kreb's article [7] about all the ways that people's Windows desktops are abused in moneymaking mayhem.

While I wish I could say that Linux is the answer, I will say that running operating systems other than Windows would certainly help many people. For the real programmers, there are the BSDs, so low in adoption rate that just about no one will exploit them. Then there are various Linux versions, a much simpler approach for the average person, and one that I have successfully convinced several friends to use (if only for their online banking and purchases). Apple's Mac OS has close to 10% of the desktop market but does not approach the exploit rate of Windows systems. This will not always be

the case unless Apple does a lot more to secure their applications, something I think they are interested in doing.

You, too, should consider doing what you can to reduce cyber-crime. Encourage your friends and relatives to use other operating systems. Dan Geer famously wrote about the dangers of software monocultures [8], and we are living with the results of ignoring that today.

REFERENCES

[1] Malicious Software Removal Tool: http://www.microsoft.com/downloads/details.aspx?FamilyId=AD724AE0-E72D-4F54-9AB3-75B8EB148356&displaylang=en.

[2] Wikipedia, Conficker Initial Infection: http://en.wikipedia.org/wiki/Conficker#Initial_infection.

[3] MS Security Intelligence Report Volume 6: http://www.microsoft.com/security/portal/sir.aspx.

[4] "MS Blames Non-Redmond Apps for Security Woes," The Register: http://www.theregister.co.uk/2009/04/08/microsoft_security_report/.

[5] HiStar: http://www.scs.stanford.edu/histar/.

[6] Nickolai Zeldovich, Hari Kannan, Michael Dalton, and Christos Kozyrakis, "Hardware Enforcement of Application Security Policies Using Tagged Memory," *Proceedings of the 8th USENIX Symposium on Operating Systems Design and Implementation (OSDI '08)* (USENIX Association, 2008): http://www.usenix.org/events/osdi08/tech/full_papers/zeldovich.

[7] Brian Krebs, "The Scrap Value of a Hacked PC," WashingtonPost.com: http://voices.washingtonpost.com/securityfix/2009/05/the_scrap_value_of_a_hacked_pc.html.

[8] Dan Geer, "Monoculture on the Back of the Envelope," *;login:*, December 2005: http://www.usenix.org/publications/login/2005-12/openpdfs/geer.pdf.