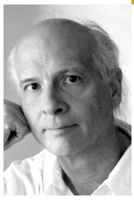
RIK FARROW

# musings



Rik is the Editor of *;login:*.

*rik@usenix.org*

**IMAGINE WATCHING THE BEGINNING** of a stock car race. As the drivers climb into their cars, they ignore the webbing for covering the drivers' windows and do not attach their five-point restraints or the head-and-neck supports. Although this goes against safety recommendations (and NASCAR rules), the drivers have decided that these safety measures are "inconvenient" and "interfere with the experience."

Certainly my imagined event sounds unbelievable today. Yet at the end of Jeremiah Grossman's invited talk at the 2009 USENIX Security Symposium, I asked the audience, over 300 of the people who had chosen to attend the premier security research conference, for a show of hands on users of NoScript. Only a few hands went up, and I sat down, astonished.

Later that evening, a usability researcher approached me. He said that NoScript was considered a bad example of usability. I certainly understand that, yet when the consequences of not using NoScript are considered, it is like not choosing to wear a seatbelt while racing because of the inconvenience. That security researchers would choose a "better experience" and "convenience" over Web browsing safety still amazes me.

## Through the Browser Window

For many years now, Web browsers have been the pen testers' choice for getting past firewalls. I know some very good penetration testers, and as firewalls became common as well as including better configurations, the initial penetration method was to attack via Web browsers. The pen testers may abuse vulnerabilities in Web browsers, but just as often they simply use features of Web browsers combined with normal human weaknesses, such as trust, that are easily exploited. While home users' Windows systems are the most infected with malware installed via driveby attacks [1], businesses are not immune. Businesses can be targeted using techniques similar to those of my pen testing friends, but the goals in these cases are different. Businesses may be targeted for intellectual property or secrets, but these days the target is often bank account information [2, 3].

I did ask Jeremiah Grossman if he used NoScript. Grossman said he did, which is not surprising considering that he had just presented the Top 10 Web

Hacking Techniques (see article on p. 16) and NoScript blocks several of these attacks. I particularly shuddered when contemplating clickjacking, a technique that allows an attacker to trick a browser user into clicking on the button of the attacker's choice. Clickjacking is a feature of modern browsers, allowing an attacker to move (hover) over an iframe so that the button to be clicked is always under the user's mouse. And this happens invisibly to the user, as the iframe is hidden beneath other content.

Many years ago, I mused about having a button on my browser that would give me the option of allowing scripting to work for a particular site. NoScript has provided that button for many years now, and Giorgio Maone has continued to add security features to NoScript over the years as well. I asked Maone if he wanted to write about NoScript for this issue, and you can find his article on p. 21.

Maone considers the same usability issues that some people complain about to be a feature. He wants NoScript to work without cluttering up your browser with popups. I will say that I had to learn that when a Web site doesn't work as expected, I need to see what NoScript is blocking. For the most part, I have already whitelisted the sites I trust, which also happen to be those I visit often. When I visit a new site, I have to decide whether to allow scripting to work. I usually enable scripting temporarily unless I know I will be visiting a site frequently. And I don't enable scripting for all sites that request the ability, as some of these sites just use scripting to present advertising or to collect information about your browsing habits. I'd rather maintain my privacy. And when I want to buy something, I will research products rather than buy the product with the spiffy and/or annoying ad.

Advertising sites themselves can be sources of malware. In [1], the researchers mention that a source of drive-by downloads comes from reselling advertising slots on Web pages. If there are no current buyers for advertising on a particular Web page, these potential slots can be resold to other advertising networks. You could wind up being the victim of an attack even when visiting a trusted site, if you don't use NoScript. Because NoScript blocks scripting based on the site the script comes from, you can still view your favorite site while preventing other sites from executing scripts.

## Banks and Credit Cards

Credit card companies provide you with some insulation against loss of credentials. If someone steals your credit card info, you can report it to the credit card company and pay a limited amount (at most $50 in the US). Banks, however, look at credential loss completely differently. Banks have traditionally focused on using SSL to protect transaction data while it traverses the Internet. At the same time, banks assume that the endpoints of the communication, including browsers, are secure. Yet that is unlikely to be the case today for most PC users.

Even the use of one-time passwords and password generation tokens does not provide protection for users of malware-infected PCs. Malware has been designed to wait until the user has provided authentication and then to initiate a fund transfer request that appears to the bank to be authenticated. If you can't trust your own computer, SSL really doesn't help you at all. Adding insult to injury, banks in some countries hold users responsible for losses if the computers they use are not secure.

In the US, recent looting of the bank accounts of small businesses and even a county have garnered some news. And in these cases, the owners of the accounts, whose credentials have been stolen using malware and botnets, were held responsible for the losses as well.

## Secure Operating Systems

It would be helpful if we could use secure operating systems. Just recently, an Australian research project undertaken to prove the correctness of an operating system, the seL4 microkernel, was completed. You can read what one of the researchers, Gerwin Klein, has to say about this starting on p 28 of this issue. The operating system executes with the highest level of privilege and has exclusive access to all hardware devices, including disk and network devices, as well as arranging for access to pages in memory. Having an operating system with proven security guarantees is a great leap forward.

We also need secure applications that we can use. I've written about the OP browser before [4], a browser which uses process-based isolation for each site that goes well beyond what Google Chrome and IE8 do today. Just the week before I completed this column, the source code to the OP browser went online [5]. There are still issues with the OP browser, mainly having to do with running isolated windows on top of window managers that do not support the concept of sharing a display among different security principles—that is, each site acting as a separate user, isolated from what other sites can do. And sites that rely on overlapping views, such as mashups that use overlays on top of maps, are very difficult to deal with. But the OP browser, because of its design, already shows higher performance on multicore systems than IE8 or Chrome for certain tasks.

### SECURE TCP

Not even TCP itself can be considered safe. Security issues with TCP connection state have been known since 1985 [6], but little has been done beyond quick fixes, such as initial sequence number randomization. As DNSSEC begins to see wider adoption, starting in December 2009, root server operators will really be feeling the effects of having to support TCP connections, as TCP connection state can easily be abused. Attackers began using SYN floods against TCP in 1996, and only non-standard kludges defend against these and similar denial-of-service attacks today.

Metzger, Simpson, and Vixie have written about a change to TCP that eliminates these issues. This change, TCP Cookie Transactions (TCPCT), has been discussed for many years in some form and appears close to being implemented in at least two OS stacks soon. TCPCT can easily be integrated into the Internet, as the new option will be ignored by systems, including firewalls, that don't recognize it. You can read about TCP Cookie Transactions beginning on p. 35 of this issue.

## Lineup

I have actually touched on many of the articles in this issue of *;login:* already. Dave Dittrich has written an article that both recalls the history of distributed denial of service (DDoS) attacks and ethics. Both researchers and investigators need to be bound by a code of ethics, perhaps legally bound. Dittrich carefully covers this concept with a story about how he collected the source code to early DDoS tools.

Peter Galvin covers an emerging feature in OpenSolaris: Immutable Service Containers (ISCs). ISCs are a containment mechanism designed to be used for networked services. Initially they work with Solaris Zones, but may eventually work under Solaris VM environments as well. ISCs promise to be another useful tool for securing services.

David Blank-Edelman provides more general advice for Perl programming, or, as he has put it, he “likes to get meta.” Blank-Edelman describes simple techniques, as well as how to endure them, for improving the robustness of your Perl code.

Dave Josephsen provides 7 tips for successful Nagios implementations. Actually, you would do well to pay attention to his list no matter what type of monitoring and reporting you are doing.

Robert Ferrell regales us with his own definitions of terms used in the security industry. Robert has his own way of looking at things, as you will have noticed. I find that I can strongly agree with Robert on his definitions, as a large dose of cynicism is in order when it comes to computer security.

We have lots of book reviews this time around, and we close with reports from the 2009 USENIX Security Symposium and two associated workshops: HotSec and CSET.

I became paranoid about UNIX security, and later Internet security, starting in 1984. That was the year someone shared a much-copied list of security exploits that had occurred at UCSC over a few years. The list provided a reminder of what clever students could do with a little knowledge and a dose of misguided motivation.

Today, exploiting browsers is big business. Exploits are sold on the black market, converted into easy-to-use toolkits for exploiting browsers and Web servers, then sold. These tools are designed to steal login credentials or to proxy authenticated connections to banks and financial institutions. Nothing magical is involved here, as our current Web browser technologies actually support the installation and use of tools that have browser-wide impact. In fact, without this support, NoScript itself would not work.

Strap into your Web browsers! I encourage you to endure the inconvenience of having to decide, perhaps after some research, whether you consider a site safe or not. While NoScript’s user interface could be easier to use, I find a bit of inconvenience a lot more palatable than the consequences.

## REFERENCES

[1] N. Provos, P. Mavrommatis, M.A. Rajab, and F. Monrose, “All Your iFRAMEs Point to Us,” *Proceedings of the 17th USENIX Security Symposium*, July 2008, pp. 1–15.

[2] Kelly Jackson Higgins, “Attack of the Mini-Botnets,” DarkReading: http://www.darkreading.com/security/attacks/showArticle.jhtml?articleID=216402026.

[3] “Clampi Targets Banking Info”: http://www.usatoday.com/tech/news/computersecurity/2009-07-30-clampi-computer-virus_N.htm.

[4] Chris Grier, Shuo Tang, and Samuel T. King, “Building a More Secure Web Browser”: http://www.usenix.org/publications/login/2008-08/pdfs/grier.pdf; Rik Farrow, “Musings,” http://www.usenix.org/publications/login/2008-08/openpdfs/musings.pdf.

[5] OP-Browser source: http://code.google.com/p/op-web-browser/source/checkout.

[6] R. Morris, “A Weakness in the 4.2 BSD UNIX TCP/IP Software”: pdos.csail.mit.edu/~rtm/papers/117.pdf.