the network police blotter

by Marcus J. Ranum

Marcus J. Ranum is CEO of Network Flight Recorder, Inc. He likes cats: they are complex yet manageable. When he's not working 10-hour days he plays console games and pursues too many hobbies for his own good.

<mjr@nfr.net>

Greetings

. . . and welcome to the first installment of what will be a regular series of columns in *;login:*. I'm a security nerd, so I will primarily be sticking to that topic, with occasional forays into other issues. Over the course of the last 14 years or so, I have evolved through the various stages of the techie life cycle: system administrator, programmer, network manager, security manager, project leader, product manager, consultant, chief technology officer, and chief executive officer. Why is that important? Because it's taught me that perspective is everything, and your ability to put things in perspective is somehow (I can't quantify) related to the variety and depth of experiences in your personal and professional life. So I find myself sometimes in the unique position of trashing positions I once held or waving off complex technology issues as "mere details." All I can ask is that you bear with me; I promise I won't write anything that hasn't got some underlying point that I've thought out.

Security Today

Let's talk about the state of enterprise security today. That's a huge topic, of course, but it's going to be important to us all for a long time to come. Obviously, the Internet is a big chunk of that problem, but security concerns will eventually push their way into virtually anything that's doing computing. Assuming that hasn't already happened. From the "30,000-foot view" there really isn't a huge difference between a company's intranet and the Internet. About the only difference I usually see is that security is ignored on the intranet and paid attention to at the Internet connection. This results in a sense of security, since there's a separation between the things we control and the things we don't control. That's very important to management since it breaks things neatly into *things that are our problem* and *things that are not our problem*. When you see people trying to break big problems into smaller, more tractable problems, that's a sure sign that they are trying to *manage complexity*. Managing complexity is a difficult problem, so let's examine a couple of ways in which it applies to security.

I sometimes have the privilege of addressing rooms full of technical people who are interested in security. This is a great chance to conduct quick unscientific polls. One of my favorites has to do with browsers. You ask a roomful of technical people to raise their hands (and keep them up) if they have had the following happen:

1. A browser crash in the last hour (one or two hands go up).
2. A browser crash in the last 24 hours (5% of the hands go up).
3. A browser crash in the last week (60% of the hands go up).
4. A browser crash in the last month (the rest of the hands go up).
5. A browser crash in the last year (general laughter).

The next question is:

Given that you've proven to your satisfaction that you're using unreliable technology, how many of you engage in e-commerce or online stock trading using a browser?

This question is usually followed by nervous laughter, and a brave few admit it and raise their hands. I don't know about you, but I buy lots of stuff online. The reason is

Perhaps the browser in Dreamcast will never have a security flaw. If so, it'll be the first secure browser ever.

simple: *it's the only game in town*. Well, not the "only," but someplace in the back of my mind I made a quick assessment of the options, and convenience beat security hands down, considering that the risks in telephone or storefront sales are on a par with e-commerce. I'm not sure that, as a technological society, we should be comfortable with using something because it's the only game in town. But is there an alternative?

Regardless of whether or not there is an alternative, the scope of the problem is only going to get larger. I recently bought a Sega Dreamcast gaming console. It has a 56k modem, an IP stack someplace inside of it, and a little logo on the front that says "Compatible with Microsoft Windows CE." And a lot of trademark symbols. One of the other things my Dreamcast came with was a CD that contains a browser. Now, browsers demand open Internet access and mean e-commerce. Perhaps the browser in Dreamcast will never have a security flaw. If so, it'll be the first secure browser ever. One thing for sure, the average age of Internet users will continue to go down, along with their level of technical sophistication. Console gaming appliances and other Internet-access appliances are tools that are being deployed to manage the complexity of getting onto the Internet. I'm not saying that getting on the Internet is exactly hard now, but it's going to get even easier. I can't count the number of browser-bearing telephones, PDAs, and toilet seats we'll be presented with in the coming years. But it'll be a lot. Consumer appliances are all designed to manage complexity, on the assumption that average users don't want to understand what they're doing.

Managing Complexity

Indeed, the browser itself is a tool for managing the complexity of the Internet, circa 1992. Goodness, those ftp commands had such a wretched interface! Let's give the average user a way of getting on the Internet by just pointing and clicking, and all the details of HTTP, ftp, telnet, etc., shall be hidden from them by an overlay of graphics that "do what I mean" at the click of a mouse button. Newer-generation browsers are so complex they take on some of the properties of operating systems. They dynamically load programs, search paths for plug-ins as UNIX shells do, call other executables, maintain their own file systems and caches, and so on. Indeed, there is one browser maker that claims its browser is an operating system or is such an important part of the operating system that the two can no longer be separated. Yeah, whatever. But what are the implications for security? I'll tell you: they're bad. Any time a system tries to do things for the user and hides the details, there's a good opportunity for a bad guy to dupe the system. Putting a browser-style user interface on something is a great way of reducing the apparent complexity of a system – but it replaces that with the complexity of the browser.

Firewalls are devices for managing complexity at a network level. When I built my first firewall, it had to service only ftp, telnet, SMTP, NNTP, and DNS. Today's firewalls are expected to operate in a service environment that is hugely more complicated. The number of services deployed across the typical firewall has gone up dramatically, while our comprehension of them and their implications has gone down. How many people know intimately all the features and hooks of some of the cool new Web apps? Probably the people who coded them and nobody else. In many cases, the security of the protocol is that the protocol is unpublished and changes constantly from version to version. The firewall, in its highest-level view, is a thin layer of incompatibility that is deliberately placed between two areas that are incomprehensibly complex. The firewall breaks all the stuff we don't understand and trust and, we hope, protects us by doing so. What's scary to me is that more and more applications "understand" firewalls – by which the designers mean they tunnel holes through them so that the mysterious

undocumented protocols will still work. Firewalls have now become just another piece of complexity to kludge around.

Intrusion-detection systems are devices for detecting deviation from expected complexity. If I expect my network to contain a mix of applications of a certain type and it starts seeing traffic of another type, it means my network has gotten more complex without my permission, and that usually spells trouble. Vulnerability scanners are tools for assessing whether our complex systems are in expected configurations. System managers no longer have time to understand the jillions of things that could be wrong with their systems and have to rely on a piece of software to put a nice interface on it all by summarizing what needs to be fixed and why. At every level where we simplify the complex, we lose some information – and we lose our ability to understand what is going on behind the scenes. I suspect that if you asked some kids if they knew there was an IP stack in their Dreamcast, many of them would wonder if it plugged into the expansion slot or the memory-card port. I believe we need to be building the next generation of systems so that they are accessible to the nontechnical, but the more layers of paint and duct tape we put around the underpinnings, the harder it is to see and understand the implications of all the cracks underneath.

At every level where we simplify the complex, we lose some information – and we lose our ability to understand what is going on behind the scenes.

Appliances are merely the next trend in managing complexity. But will we eventually have too many appliances? I was looking at an ad the other day for a plug-in fileserver. Put the 10BaseT connector in, power it on, and it's an instant fileserver. How is it secured? Well, presumably, since it's an appliance, it's remotely controllable. I'm not saying it's a bad product, but it's designed to appeal to the nontechnical, and as a direct result of being nontechnical they probably won't even think about the security questions. We're seeing the same thing with the new generation of home high-speed Internet connections. Instead of intermittently dialing up to the Internet, a whole new population leave their machines connected 24 hours a day, where they can be quickly scanned and pillaged at leisure. It's not a new security problem – plain old dialup has the same issues – but the customer base is increasingly less sophisticated, and the service providers are reluctant even to breathe a word about security, because then they'd have to educate their customers. Worse, they'd have to educate their customers about a problem with the service they propose to sell. That's bad for business.

Today's Toy Is Tomorrow's Business Tool

Complex and poorly understood technologies are being rushed into customers' hands at an ever-increasing rate. One of the things that fascinates me about the Internet is how quickly an application can gain a massive installed base. For example, one of the online messaging systems in widespread use was signing up 100,000+ users a day. That was before it became really popular. It was a toy application and had essentially zero security built into it. It had no good authentication or encryption, and eventually it came to support file transfers and remote URL sharing. Now it's probably still a toy application, but I bet that within two years people will be using it to negotiate mergers and acquisitions, hold product strategy meetings, or issue stock buy/sell orders. It'll still have essentially zero security built into it. How can we, as security people, get application designers to build security into version 1.0 of their software? Would the world be a better place if we could put security gurus in a time machine and send them back to whisper in Tim Berners-Lee's ear, "Put security in it, this is gonna be *big*!"? I don't know how Tim would answer, but the usual application developer's answer would have to be, "That's too complex, I haven't got time, I'm on a tight release schedule." In other words, the way the application designers manage the complexity of security is by leaving it for later.

Okay, I've rambled enough. Next column, we'll talk about less nebulous high-level stuff and try to pick on something more technical.

I'd also like to run a contest, in which the winner will get a cool T-shirt. The winner's entry will be in the next column. Your mission, should you choose to accept it, is to write a computer-security haiku. Email entries to <mjr@nfr.net> with a subject line reading "haiku." I'll notify the winner before the next issue.