Musings

RIK FARROW

Rik is the Editor of *;login:*.
rik@usenix.org

It feels like the '90s all over again. No, I don't mean the IPO of LinkedIn for a ridiculous figure (16 times earnings). Rather, it's the rash of highly publicized security incidents, with Sony being the most recent—and the most frequent. LulzSec has bragged online that their "hack" was embarrassingly easy: a simple SQL injection [1]. The attacks on Sony's networks worldwide have led to a flood of released data, including info on registered users and source code.

The Sony attacks, and those on HBGary Federal and PBS.org, seem to have been done for political reasons. After a decade where computer attacks have been primarily focused on financial gain, this marks a rare turn toward vigilantism.

What these attacks also reveal is the very porousness of network security. When the only way we would learn of attacks was after the California law on disclosure of personally identifying information forced an announcement, it wasn't obvious just how often organizations were being broken into. Now it seems as if we are back in the era of unpatched Linux systems being taken over by automated attacks.

But this time around, things are different.

The Difference

Much has changed between 1999 and 2011. The publication of exploits has largely gone underground, as exploits are now sold on the black market both to governments and to criminals. Attackers are no longer motivated by becoming famous (or infamous), with the recent exceptions of Anonymous and LulzSec. Instead, criminal organizations use exploits to take over Windows PCs for use in botnets, or steal databases loaded with credit card information. The monetization of exploits has long been under way.

But some things have not changed at all.In 1999, only a fool would claim that their Internet-connected system was totally secure, and proof against all attacks. And that is just as true today. Only we behave as if it isn't so.

And instead of attacking Internet-facing *nix servers, today's attackers can rely on a different technique, one that totally trashes any concept of having a "network perimeter." They can gain a foothold inside any network through the use of email and Windows PCs. The Google attack, announced over a year ago (January 2010 [2]), has just been repeated. I believe that similar attacks are behind the exploitations of RockYou (32 million passwords stolen), Gawker (300,000), and certainly of HBGary Federal.

This type of targeted attack requires some research on the attacker's part, to determine the subject line and content for an email message likely to make it pass spam filters and be opened by the intended victim. You can read more about an example of this particular attack in Ned Moran's article in this issue about the Advanced Persistent Threat. But it is not just APT that can take advantage of "spear phishing," sending email containing exploits to targeted individuals. Anyone willing to spend enough time to understand the targeted organization, perhaps by using LinkedIn to uncover links between individuals that could be used to fashion an effective email message, could successfully breach a network today.

The only effective defense against such attacks is either using the mail command-line program on a *nix system without X Window to read email or keeping all critical assets on networks segregated from networks with GUI users. I don't believe for one minute that people are going to go command-line anytime soon, so it looks like we need to consider the alternative approach.

Assume the Worst

Dominique Brezinski explained, in a private email, that he has had a lot of success with the following alternative approach (quoted with permission):

> It is my opinion that in any computing environment with a non-trivial population you must assume some client devices and some user accounts are compromised. There are a couple advantages to making this assumption:
>
> Your defensive strategy no longer makes a delineation between external and internal adversaries. It is only a single, easy hop from outside in. Just assume it happens. Tailor your defensive strategy around the insider problem.
>
> Whether you own the client device or not makes little difference.

Dom's first point echoes what I have already written: assume the worst—your network has already been compromised. Even if it hasn't been (as unlikely as that is), assume it has. Then behave accordingly.

Much has been written about the insider threat (see, e.g., [3]), but Dom is thinking more tactically. Assume your attacker is an insider. How do you go about protecting your critical assets? Isolating them, using firewalls that separate out servers with critical information, is a good start. This is nothing new, as it was considered best practice in the '90s—even as it was usually ignored.

Just isolating servers with critical data is not enough. People still need access to those servers, as do applications. So you must limit that access carefully, both for applications and for users. Dom suggested that users who must have access use "authenticators that are unique per session" and so cannot be stolen or reused. And that users' access does not imply unlimited authorization: in other words, put a system in place that restricts access and logs all activities. It would have been nice if Sony had noticed that someone was downloading megabytes of source code. It would have been better if this had not even been possible. But at the very least, having a system in place that notices unusual activity and notifies someone is not just reasonable, it is a requirement for securing critical data.

Instead of just beating up on Sony, let's look at another example: WikiLeaks and the 251,287 diplomatic cables [4]. When I first learned of this immense treasure trove

of secret or sensitive information, allegedly leaked by a young soldier, I couldn't believe it. Why on earth would an intelligence officer stationed in Iraq have access to diplomatic cables? This sharing of data that doesn't seem as if it belongs on a soldier's laptop came about as a reaction to the lack of sharing of intelligence that may have contributed to the success of the attacks of 9/11. But suppose that a system was in place that detected the collection of this vast amount of data, including cables going back more than 45 years? If even a reasonable amount of monitoring of access was being performed, would this soldier have been able to collect so much information without being detected? Keep in mind that Bradley Manning is now in prison not because of a system that managed and logged access, but because he chatted online with Adrian Lamo.

To ground these ideas, let's imagine you have a database that serves information to customers using a Web front-end. And to make things interesting, let's also imagine that this database contains a customer's name, address, and credit card information. You want your customers to have a pleasant experience while using your system. You don't want them to be able to download the entire database of sensitive information, just their own information. This is what is meant by limiting authorization: the customer has access to her own information only. Additional controls would include isolating the database on its own network, and using firewalls to limit access to that database to only the Web server's application and to the people who must manage the database. You must also include controls that prevent the dbadmins (or an attacker who has taken over their desktops) from abusing their privileges. And that is the really difficult part.

When you consider that the database of information used to restore the cryptographic info for lost SecureID tokens was stolen from RSA, a security company, you can see that the concept of isolating critical assets, even when their compromise will lead to terrible results [5], is often ignored.

On the other hand, I hope it is useful to view your network of email and Web connected desktops as already compromised. If it isn't already, it soon will be. Keep your valuables someplace else. Please.

Spam Kings

I don't spend all of my time lamenting the lack of real security or attending cool workshops like HotOS (see the reports in this issue). I sometimes get to read really interesting stories from researchers who have been doing tremendous work.

I interviewed Stefan Savage (UCSD) for this issue, and I can't say that it was hard work. Stefan is a great storyteller, and his story is a compelling one. Starting in 2006, Stefan, along with several other professors at UCSD, UCB, and ISI, began investigating spam. They started by looking at how spam is delivered, moved on to the botnets that deliver most spam, and, finally, studied the fulfillment side of spam. As we all know, if no one clicked on the links in spam and then actually bought something, spam would have vanished years ago. But spam is still a successful marketing tool.

Stefan tells us about how they got to the point where they were actually buying pharmaceuticals, fake Rolexes, and software by following spam links. This was the latest step in a long process, and the results were published in a paper [6] at the IEEE Symposium on Security and Privacy (Oakland) this summer. I enjoyed listening to Kirill Levchenko, in the crowded ballroom at the Claremont Resort,

explain how hard it was to get permission to actually buy spam-advertised goods, as this involved both using research funds (you want to do WHAT!?!) and tricky negotiations that made it possible to track the transactions via credit card companies. And this is just one of the stories Stefan has to tell. I also liked the one about how the FBI was about to arrest them at a USENIX workshop, but you should read this for yourself.

Stefan's interview also provides a clear window into successful research. Stefan and his associates followed paths that were not always successful. Sometimes he worried that his students (and other advisors' students) were wasting their time, only to be surprised by the results. And the results so far have been over 14 papers published, including three at USENIX Security '11 and one at CSET '11.

Ned Moran has written about APT, using a recent example of a spear phishing attack against US government employees. APT differs from attacks by Anonymous in that it requires teams of people ready to react to a successful penetration. The actual technology does not appear that exciting, although I believe the details of the remote access tool that Ned dissects will prove interesting to *;login:* readers.

Raphael Mudge has been working with the Metasploit penetration testing software to create his own front-end, Armitage, that makes it easier for a team to work together. Raphael's tool is designed for helping red teams practice attacks, and if you are interested in the attacker's perspective, I suggest you read his article.

Peter Gutmann has provided us with a short article about the problem with SSH key fingerprints. It is not that the fingerprints are useless, it is that they are both not used properly and too easily abused.

Ben Hindman and a long list of co-authors explain Mesos, a system that works with Hadoop and other frameworks, such as MPI, used for performing work in parallel on many systems. Mesos is itself a framework that improves the performance of parallel tasks by using dynamic partitioning of systems, instead of the static partitioning supported by Hadoop and MPI.

Tom Limoncelli and Doug Hughes, co-chairs of LISA '11, explain why they believe that DevOps, the chosen theme for LISA, is important. DevOps implies close collaboration between sysadmins and developers, which is how systems are being developed today.

David Blank-Edelman continues on the theme of Web frameworks with Mojolicious. Mojolicious is similar to Dancer, but only when you first encounter it. Mojolicious provides a stand-alone (no other modules required) and complete Perl-based Web framework, designed to make difficult things, such as managing sessions, simple.

Peter Galvin has renamed his column "Galvin's All Things Enterprise." Peter's first installment covers that buzziest of buzzwords, the cloud, but without getting lost in the clouds. Peter defines cloud computing from an IT perspective and explains why it is important.

Dave Josephsen strays away from monitoring to tell us about a project he has been working on for months. Dave shares with us descriptions of the shell-based batch processing system that replaced the hundreds of scripts he inherited when he started working as a sysadmin.

Robert Ferrell takes us on a "circuitous ramble" through the different types of job interviews, before sliding into talking about entitlement and security. His own recent encounter with a surreal interview provides fodder for his column.

Elizabeth Zwicky reviews four books, including a couple she really likes, and one on interviewing for security positions. She did not communicate with Robert Ferrell, so this is a serendipitous occurrence. I contributed two book reviews myself this issue, including one on a novel written by a Google employee about the potential for abuse of data collected about users of a company that sounds vaguely like Google—not quite Google, as this fictional company has access to lots more data than Google does.

Finally, we have the HotOS summaries. HotOS is one of my favorite workshops, even if it only happens once every two years. This is the place for OS researchers to expose their sometimes very far-reaching ideas in front of an audience of critical thinkers.

Before leaving you, dear reader, I want to remind you that your network has been compromised. How do I know? I don't have to know, I can guess. Unless you are running a single stripped-down *BSD system on a firewalled network with no GUI, the odds that your network has working bots on it, along with remote access tools, is high. Even if your network hasn't been compromised, how would you know?

I like to put a sniffer outside my network and analyze the traffic I find there. This is possible for my network because there are only two users and a couple of lightly used Web servers. If you reconfigure your networks so that your critical assets live behind severely restricted firewalls, you could do this as well for the traffic going between the protected network and the rest of your networks. But in the "let's keep things as wide open as possible so we can make more money, uh, get work done" mode, real security is just not possible.

References

[1] "Hackers Claim to Have Hit Sony Again": http://www.reuters.com/article/2011/06/06/us-toni-cybersecurity-sony-idUSTRE75563L20110606.

[2] Rik Farrow, "Google Attacked via Email": http://blogs.usenix.org/2010/01/14/google-attacked-via-email/.

[3] Dawn Cappelli, Andrew Moore, and Timothy Shimeall, "Protecting against Insider Threat": http://www.sei.cmu.edu/library/abstracts/news-at-sei/security-matters
200702.cfm.

[4] United States diplomatic cables leak: https://secure.wikimedia.org/wikipedia/en/wiki/United_States_diplomatic_cables_leak.

[5] Dan Goodin, "Stolen RSA Data Used to Hack Defense Contractor": http://www.theregister.co.uk/2011/06/06/lockheed_martin_securid_hack/.

[6] Kirill Levchenko, Andreas Pitsillidis, Neha Chachra, Brandon Enright, Mark Felegyhazi, Chris Grier, Tristan Halvorson, Chris Kanich, Christian Kreibich, He Liu, Damon McCoy, Nicholas Weaver, Vern Paxson, Geoffrey M. Voelker, and Stefan Savage, "Click Trajectories: End-to-End Analysis of the Spam Value Chain," *Proceedings of the IEEE Symposium on Security and Privacy*, May 2011: http://www.imchris.org/research/levchenko_sp11.pdf.