SECURITY NOW!

Transcript of Episode #291

Stuxnet

Description: After catching up with a very busy week of software updates and wide-ranging security news, Steve and Leo discuss the revelations documented in Symantec's comprehensive "Stuxnet Dossier."

High quality (64 kbps) mp3 audio file URL: http://media.GRC.com/sn/SN-291.mp3
Quarter size (16 kbps) mp3 audio file URL: http://media.GRC.com/sn/sn-291-lq.mp3

Leo Laporte: This is Security Now! with Steve Gibson, Episode 291, recorded March 9, 2011: Stuxnet.

It's time for Security Now!, the show that covers your security online, your privacy. And here he is, the man of the hour, our guru from the Gibson Research Corporation, the creator of SpinRite - the world's best hard drive utility - and many great free security programs, Steve Gibson. Hi, Steve.

Steve Gibson: Hey, Leo. It's great to be with you again, as always.

Leo: What's our topic of the day today?

Steve: We've got a great one. I referred to it a couple weeks ago, probably when you were off and Tom was holding down the fort here, that Symantec had released a very comprehensive report on their detailed analysis of Stuxnet, which they really took apart. And I thought, well, this would make an interesting topic. And now I'm convinced that, by the end of this podcast, no one listening will be able to doubt that the term "cyberweapon" applies. I mean, that clearly - I don't think there's any way this could have been produced without national-level state sponsorship. And this thing is so sophisticated and was so targeted, the statistics that have been gathered demonstrate it.

And one of the coolest things that Stuxnet did is, as it infected machines, it appended to a log of its - basically of its travels. And by capturing samples, the log was - Symantec was able to obtain more than 3,000 instances of Stuxnet. And by going back through the log that it itself carried, they were able to, basically, follow this thing back in time and determine that there were three specific attack waves, and which five companies were targeted. So it's not just like it's, oh, look, it's everywhere, as though all over the Internet everyone has it. No. I mean, we know exactly how this happened. And it's really fascinating.

Leo: Well, I can't wait to find out more about it. We also have other security news, quite a bit of security news as we approach...

Steve: Oh, we do, yes.

Leo: ...the Pwn2Own Conference, which I think is coming up today or tomorrow. All right, Steve, I have in front of me a list, starting with Patch Tuesday.

Steve: Well, yes, we are just past our standard second Tuesday of the month. So Microsoft has actually a rather lean response this month. They fixed four different vulnerabilities, one which was critical in their media playback which affected all the recent OSes - XP, Vista, and Windows 7 - such that, if you went to a site that had a specially crafted malicious video, it could execute code on your machine. That they fixed.

The bad news is the zero-day exploit, which we have talked about recently, the so-called MHTML exploit - MHTML is sort of a pseudo protocol. In the same way that we have HTTP:, Microsoft defines MHTML: as a way to invoke MIME-encoded HTML. We talked about how that's used for archiving whole web pages, in the same way that MIME stands for Multipart, what is it, Multipart Internet Message Extension or something?

Leo: Yeah, something like that.

Steve: For allowing email to contain nontextual things, like photos and so forth, MIME is how you do that. Similarly, this is how Microsoft has their own proprietary format for storing an entire web page including all of its assets, its other photos and so forth. There's a problem with it such that, if you go to a website that invokes this protocol, similarly they're able to get their own code to run on your machine. Well, that didn't get fixed this Tuesday, and I was hoping it was because it is being actively exploited in the wild.

So I wanted to remind our listeners that there is a one-click easy Fixit button that Microsoft offers. If you go to go.microsoft.com, then ?linkid=9760419, that will take you to this page with the quick fix dealie that just disables that protocol. And probably everyone, I mean, it's one of those things that's on by default. It's got a problem in it that, if you don't know you need it, you probably don't. So, I mean, I immediately went there and just said, I don't need this, I'm turning it off. And had Microsoft fixed it a couple days ago, we'd probably be okay. But like these things, now that it's seen that Microsoft hasn't fixed it, we can expect more exploits to happen. So...

Leo: It's a "sit up and take notice" to hackers.

Steve: It's a problem, yes, exactly. They're saying, hey, we've got another month, probably. So let's jump on this. So more important to do that. So I don't know what you could Google to get there. It's MHTML exploit, but you can go to go.microsoft.com/?linkid=9760419.

Leo: And I get a download, immediate download when I go there. So you're getting a .msi file, an installer.

Steve: Yes, in fact, that is the link that you get when you click the button. And so it'll instantly say, here's your goodie, run this, and we'll - and all it's doing is it's just making some registry tweaks. It's changing, it's basically removing the protocol from the registry where it's defined. So it's not even - it's not installing anything. It's not making any deep changes in your system, just changing some values. And there is, on the related pages about this, you can see do-it-yourself registry stuff, if you don't want to use Microsoft's little quick fixer.

Leo: Yeah, maybe just go to the security advisory 2501696 if you want to read about it. And there's I'm sure a link there to the download, so.

Steve: Yes, that's great advice. Now, everybody else has, in somewhat of a panic, they have pushed out fixes for their browser because of something you referred to that is happening in Vancouver as we record this today. Wednesday the 9th, the 10th, and the 11th of March is the CanSecWest. It's the 12th Annual CanSecWest Security Conference, Canadian Security West Security Conference, happening in Vancouver. One of the, well, first of all, there are a bunch of fun things going on there. In the agenda, one of the talks is "SMS-O-Death," which I get a kick out of. Also "iPhone and iPad Hacking" and "Stale Pointers Are the New Black" are some of the topics for the conference.

Anyway, one of the things that they host there, and we've talked about this for the last several years because fun things come out of it, is the so-called Pwn2Own, basically competition, where the security researchers and white and gray hat hackers who attend attempt to exploit not-before-known, that is to say, zero-day vulnerabilities which they've discovered or know of in the popular browsers. So in the ramp-up to this, Apple recently immediately updated their Safari, both on the Mac OS platform and on the Windows platform. In the case of Windows they moved iTunes, which uses the WebKit engine which is what's common in Safari, they moved iTunes up to 10.2, in the process fixing more than 50 known security vulnerabilities over on the Windows side. iTunes over on the Mac OS was moved to 10.2.1, also just recently. So anyone using Macs, or especially iTunes over on Windows - I don't know, if you're not going into iTunes very often, it's worth to get 50 security vulnerabilities fixed, you want to bring yourself up to 10.2.

Leo: Even if you don't use it often.

Steve: Exactly. Over on...

Leo: It's a little deceptive because it's an iTunes update, and people assume, oh, well. It's like if you were on the Mac, and you didn't use iTunes, you'd go, I could ignore it. But, no, really it's not.

Steve: Yes, because it's WebKit that is being brought along. And that's the rendering engine, the layout engine in Safari. And there's all these problems that they know of in font rendering and HTML layout.

Leo: So even if you don't use iTunes on Windows, if you use Safari on Windows, you have to do this.

Steve: Yes. Yes.

Leo: Yeah. And I presume there'll be WebKit updates for WebKit itself and other WebKit-based apps. Chrome is WebKit based, so that's probably where the Chrome updates come from.

Steve: Yes. Over on the Mozilla side, similarly, Firefox, they brought themselves completely up, completely current. Anything that they knew of that they hadn't sort of gotten around to pushing out yet, before this conference it's like, oh, we've got to get this out. So in fact I noticed that I'm still at 3.6.13 on my main system, although a laptop that I started up yesterday did bring itself current. Firefox is now on the 3.6 chain, is up to .14. If you're still back at 3.5, that's at 3.5.17. But also Thunderbird needs to be brought up to 3.1.8, and SeaMonkey up to 2.0.12. And this fixes a bunch of stuff: problems in the JavaScript engine, code-handling under HTML, style sheets, scalable vector graphics, objects, and JPEG images. So just across the board. And as is typical now, these are exploitable by enticing a user to visit a malicious website that then gets your browser to do something it was not designed to do.

And lastly, Chrome, always sneaking along as it is, when I fired my Chrome up, it was behind times, and so it quickly updated itself to what is now current, 9.0.597.107, in the process fixing 19 known security issues. And Google never talks much about what these are, but we know for example that several are stale pointer vulnerabilities. In fact, that was the topic of one of the talks was, remember, "Stale Pointers Are the New Black." And also they had an integer overflow problem. And one of the things we are now seeing more of is memory use-after-free vulnerabilities, where memory is released, but then there's still a pointer to it that allows it to be accessed in a way that the designers did not intend. So that's cleaned up in Chrome.

So across the board, our browsers have sort of straightened themselves up in the hopes that they're able to survive the next three days. This Pwn2Own conference, on their website they say, "If you can execute arbitrary code (PWN)" - which of course is the hackronym for maliciously taking ownership of something that was not yours, and really it's interesting because there isn't really a clear history of how that came to be. People are assuming maybe it was a typo. Since "P" on our English keyboard is right next to "O," maybe somebody was typing, meant to type "OWN," O-W-N, but they typed "PWN," they thought, hey, that's kind of cool. Who knows where it came from. But so if hackers are able to execute arbitrary code through a previously undisclosed browser - either Firefox, IE, Safari, or Chrome - exploit, then the site says "you can go home with one (OWN)." So Pwn2Own.

The browser prizes for exploits were increased this year to $15,000. And there are also state-of-the-art phones, laptops, and cash. And then Google went a little further, decided to stoke the fires under Chrome, and tossed in an additional $20K for Chrome-specific exploits which are found. So if anyone's curious, the site is cansecwest.com. And the /agenda.html shows the three days. And I'm sure that we will be talking about, next week, the outcome of these three days. The Pwn2Own competition always produces some fun results.

Leo: It's a great, it's an amazing event. And I think this is the measure of it's a real success is that people are now, I mean, they didn't do this in the past. And credit to Charlie [Miller] for breaking Safari every single time, winning 15 grand and a Macintosh laptop. I think this finally forced Apple, which has been traditionally pretty slow to acknowledge these problems, to do something about it. It's embarrassing for them. So good job.

Steve: Well, and I'll say again that it's worth noting that Microsoft didn't do anything other than just roll out their regular old Patch Tuesday.

Leo: But that patch, that HTML flaw, could well have been the one that people wanted to use; right?

Steve: Yes. And so, and it was disqualified because that's publicly known. So it has to be...

Leo: Oh.

Steve: Yeah. So...

Leo: They can't use something that is known.

Steve: Correct. It's got to be a surprise. So it's going to be something new.

Leo: Charlie Miller. I'm sorry, I said the wrong - so Charlie says, by the way, "I got something." He says, "It's okay." We'll see. It's amazing, I mean, Apple gets Pwned almost immediately, in the last three years that I can remember, almost immediately Safari is broken.

Steve: Yeah, well, I've been studying, for the last several weeks, JavaScript in greater detail than I ever have before because I have something that I want to do that has to be done in real-time, interactively, on the client. And I've gone through all kinds of hoops historically to absolutely not require any scripting on GRC. I have famously the script-free dropdown menuing system, which I created out of pure CSS, nothing but cascading style sheets. The eCommerce system that I wrote for SpinRite commerce does not even need cookies to be enabled. And still the whole shopping card system works, and your state is preserved as you go through that experience, with no state whatsoever being kept in the browser. And but there are times when you need something browser-based interactive, you know, like Gmail, and pretty much Google anything, where you absolutely have to have scripting.

So anyway, so I've been bringing myself up to state of the art where I can actually write a substantial chunk of code in JavaScript, I just shake my head. I mean, I'm just, it's like, oh, my god, I mean, the language begs to have people use it wrong and create bugs. The problem is it tries to be simple, but a language isn't simple. A language is inherently complex. And so by saying, oh, look how simple this is, it does seduce nonprogrammers to try to start using it, and they can get results. But inevitably they start wanting to do more complex things. And the way JavaScript was designed, it just begs you to have problems because it sort of tells nonprogrammers, oh, look, this is just sort of scripting. Anybody can do this. And anyway, so anyway, when I look at these stories of competitions finding problems, and also I look at the complexity in modern browsers, I'm just not surprised that they can be owned. They've gotten incredibly complex.

Leo: Postel's Law is, and this applies to browsers: Be liberal in what you accept and stingy in what you put out.

Steve: Exactly.

Leo: And being liberal in what you accept for a browser means having to support, as you say, many, many, many, many different protocols and ways of interaction.

Steve: And styles, yeah.

Leo: And styles. And it's hard to prevent attacks when you're that open.

Steve: So here's something really interesting, and I don't yet know what it means. But BBC News has reported that, as of May 25, 2011, that European laws dictate that "explicit consent" must be gathered from web users who are being tracked via cookies. The beginning of their report says: "The way websites track visitors and tailor ads to their behavior is about to undergo a big shakeup." The story says that the changes are "demanded by the European e-Privacy directive which comes into force in the U.K. in late May," on May 25. And it says the section of the directive dealing with cookies "was drawn up in an attempt to protect privacy and, in particular, limit how much use could be made of behavioral advertising."

The directive demands that "users be fully informed about the information being stored in cookies and told why they see particular adverts." And then, quoting again, it says, "Specifically excluded by the directive are cookies that log what people have put in online shopping baskets," meaning excluded are first-party cookies, which is what you typically need in order to maintain your local state with a remote server, but it's the third party tracking mechanisms that we've talked about which apparently become, not really outlawed, but again, it's not quite sure what this means.

Now, in looking into this further, I determined that the U.K. isn't quite sure themselves yet what it means. They've said, with some embarrassment, that their formal written policy about what exactly this means won't be ready by May 25. And then of course we have the other problem of jurisdiction. I mean, they don't have any jurisdiction on me. I mean, I'm not using any tracking for anything, but the 'Net is global. And so I guess they could, within the - I don't know if this applies. It says "European law." So is this maybe within the EU? I mean, there's a lot more we need to find out about what this means. But it means something. And presumably companies operating within whatever environment this pertains to will have an obligation to disclose their use of tracking.

So again, this is the beginning of what feels like some serious change. We've got the browsers adding do-not-track-me headers. We've got the U.S. grumbling around, trying to figure out what direction it wants to go in. And here we've got now the U.K. saying, okay, there's a date. It doesn't quite mean anything yet, or we're not sure what it means, but there is one. So I just wanted to put that on our listener's radar. That's going to be interesting to see what happens.

Leo: Very, very interesting.

Steve: Meanwhile, Google has famously been having problems, more, with their Android Marketplace apps. 50, or I should say about 50 apps were found recently to all be infected with the same piece of malware known as Droid Dream. It uses a previously known vulnerability in earlier versions of Android, that is, before 2.2.2, which is the version where this hole was closed. But Google became aware that a bunch of apps, on the order of 50, had made it onto the Marketplace, had been downloaded, and were in use by Android users. So they pulled the apps immediately from the Marketplace. They've suspended the accounts of the developers who were believed to be responsible for the infected applications, and they've said they've notified law enforcement.

One thing I guess, and I haven't looked closely at the contracts with Android Marketplace, but you don't put apps up there anonymously. You have to sign an agreement about what your conduct will be. And I did read some editorializing on the 'Net that was saying, well, this is interesting because big companies like Google who have big bunches of attorneys can afford to pursue the people who do this, who are presumably known. So one of the things that Google did was then they took advantage of their so-called "Remote Application Removal Feature" to remotely go into the phones of people who had these applications installed and removed the app, and then installed something called "Android Market Security Tool March 2011," which proactively closed the hole on their versions of Android earlier than 2.2.2. So Google's doing, I would say, everything they can to respond to the fundamental problems that they have with being more open than, for example, Apple is with iTunes and the iPhone and iPad app model.

Leo: And that's the price you pay.

Steve: Yeah. Yeah, so it'll be interesting to see whether, if the Marketplace developed a reputation of we're really going to know who you are in order for you to submit applications, and we're going to stomp on you hard if you maliciously exploit your privilege of putting applications in the Android Marketplace, well, that might really cool things off there.

Leo: Exactly.

Steve: From a malware standpoint.

Leo: I'm interested to see that they have and use the kill switch.

Steve: Yes. Now, they had before, back last summer. There were two instances, I think, where they did it. But this was sweeping. This was much bigger. And it raised eyebrows. There are people who don't like the idea that, without user involvement or agreement, Google is able to go in and change people's phone configuration, go in and remove applications which are bad. I think it's entirely appropriate, frankly, given the level of user, the fact that - and it's very much like Chrome, just sort of always fixing itself and not making a big deal about it. Google is saying, look, there's problems. Stuff's going to happen. We're just going to fix it as best we can when we know about it and move on. So I would, I think - if polled, you're going to have some curmudgeon-y people who dislike it sort of on a conceptual basis. But I would imagine 99.99 percent of the people who have Android are saying, hey, fine, if there was something bad on my phone, I'm glad Google came and took it away.

Leo: Yeah. I think the kill switch is just, I mean, look, this is not your personal computer. It's a computer, but it is a computer on a larger network. And there is a certain, I think, a certain responsibility. I'd hate to see that on PCs, but I think on phones, I think it's understandable. Especially if they use it appropriately.

Steve: Yes, you're inherently connected.

Leo: Yeah.

Steve: So here's a weird thing.

Leo: Okay.

Steve: And I don't quite get this. But Microsoft is actively working to discourage the use of IE6. Now...

Leo: That's good, actually.

Steve: Oh, it's very good. But it's like, okay. First of all, I'm not sure how they're doing, going about that. But they've created a site called TheIE6Countdown.com. And you should go there, Leo, while we're talking about this.

Leo: Okay.

Steve: So it's www.theie6countdown.com. Got to have scripting enabled, otherwise it won't count down for you. It shows at 100 percent in the upper right-hand corner.

Leo: Oh, look at this. Oh, my.

Steve: But so that's - what you're looking at, and what our listeners will look at when they go to the TheIE6Countdown.com, is a map of the world showing the continuing use, against all odds, I mean, we have 7, we have 8, and we almost have 9.

Leo: But look at this, China, it's like...

Steve: Yes.

Leo: Huge. What is that, 59 percent? Or 5.9 percent? I can't understand...

Steve: 34.5 percent of IE use in China is still IE6.

Leo: Wow.

Steve: Which does make you wonder why they're not getting their security updates. Of course we know that...

Leo: It's pirated software is what it is.

Steve: Pirated software doesn't get updated, right. So it's IE6. It's now 10 years old. It's been a decade. And its use is enduring. And it's interesting because in the text down there in the lower left, Leo, that you can see, they're talking about if only people would stop using IE6...

Leo: Hey, you know, good for Microsoft. I think that's great.

Steve: Except it's their fault that it's not standards compliant. I mean, what they're saying is, if you'd stop, if only people, I mean, I agree, it's nice that they're doing it now that they have a much more standards-compliant browser. I've been reading, because I've been studying JavaScript, about the history of Microsoft IE's, for example, their event model still isn't the W3C standard. They've just done it their own way, and they're just thumbing their nose at everybody so that anyone who's programming JavaScript has to special case just for IE because IE's still the dominant browser on the planet.

But so here's Microsoft saying, down there in the lower left, please, if everyone would stop using IE, then web developers would have an easier time because of course IE6 is, like, much worse than 7, 8, or 9, which have been progressively getting better. Not just better in security, and that certainly is the case. I mean, like in the middle there it says "Friends don't let friends use IE6." And so Microsoft is really - and I didn't even know this was their site initially. But I do take my hat off to them.

So in China 34.5 percent are still using IE6. In India the number is 12.3; Saudi Arabia, next lower at 10.7; and Japan, next lower after that at 10.3 percent. So I looked at my own stats because I track browser version, sort of out of curiosity. Interestingly, IE5 still is not completely zero for us. In the last week, out of about 70,000 unique visitors, 0.07 percent of them are using - they came to GRC with IE5. 2.87 percent came with IE6. And

I'm proud to say Firefox users by far outnumber all other users. Firefox 3 is the most used browser at GRC, above even any version of IE. So people who come to GRC know what they're doing.

Leo: Yeah. And I've seen that on my own pages, too. I mean, it's not our audience we have to worry about.

Steve: Right. So Adobe released something odd. Adobe Labs released something called "Wallaby" which converts simple Flash games and animations into JavaScript, using HTML5 and scaled vector graphics, so that they can run on, quote, "devices that do not support the Flash runtimes." And of course we know what those are. There aren't many of them. And then on their release notes page they further said that "Complex animations crash the browser, and zooming in and out can cause odd artifacts in the browser." They said: "Wallaby is delivered as a 32-bit application for Windows and Macintosh."

So what this thing is, is you give it your Flash project, written in ActionScript, which is the Flash scripting language, and the various Flash resources, and this thing converts it into a WebKit-compatible HTML5 and script. They said: "Wallaby is designed to emit HTML5 files compatible with WebKit" - and they made that in bold on their release notes - "based browsers. The only" - again in their bold - "supported WebKit browsers at this time are Chrome and Safari on OS X, Windows, and iOS (iPad, iPhone, iPod). Because Wallaby uses WebKit-specific animation primitives, animation will not work and has not been tested on other browsers." So it's a WebKit-specific converter of Flash.

And I've seen a lot of criticism about it out on the 'Net because it has lots of problems. You just can't give it your current Flash project and have it work because - and on their release notes page, if you scroll down, it's labs.adobe.com/wiki/index.php/Wallaby#Release_Notes. But that'll just be the release notes section of that Wallaby page, which is where this can be found. So clearly what this is is a response to Apple and Jobs saying we don't want Flash running on our devices. In fact, we're going to prohibit it from running on our devices. But what's odd is that Apple has also made it very clear that translators cannot be used, that you have to write things natively for their platform in order to use them. And so this is a translator. On the other hand...

Leo: But it runs in the browser, ultimately.

Steve: Exactly.

Leo: Okay. So you can do anything you want on a web page. Apple can't stop you from putting a web page up.

Steve: Exactly. So I guess this is just, you know, I just sort of - this crossed my radar. I thought our listeners would find it interesting.

Leo: I'd like to see what kind of translation, how good the translation is.

Steve: Yeah.

Leo: It uses SVG for graphics.

Steve: Scalable Vector Graphics.

Leo: Yeah. Does it have a video layer? I'm looking through here. Hmm, doesn't seem to have a - yeah, video is unsupported. So the thing that most people use Flash for doesn't work.

Steve: Yeah. And they said games and animations. So it's going to be, I don't know, line drawing stuff and...

Leo: There's a lot of games in Flash that are just animations, I guess.

Steve: And so, if they work, it does allow people a way to make that happen. I guess there's no economic model for it because you can't sell something like this through the iTunes store. All you could do is say, oh, here, click this link and run this game. And we'll hope there aren't any bad security vulnerabilities with what Wallaby does. I don't know. Now...

Leo: Yeah, that's another issue. But that's up to the browser; right? I mean, if it's HTML5...

Steve: Yeah.

Leo: Yeah, the browser is ultimately the security model here.

Steve: Yup, exactly. So two conflicting reports have come out in the last week which, based on the number of people who sent me links and tweeted to me about it, generated a lot of interest. A UCSD study came out which stated that erasing data on solid state disks is difficult to do. They performed some experiments where they took a bunch of various manufacturers' SSDs and USB drives and wrote patterned data on them so that they could sort of track the data by sector, and then erased the data, like tried to do a secure erasure, like an overwrite erasure, and then went into the flash ROM chips themselves and found, as I remember, on the order of 10 percent of the data still survived.

Now, at the same time another report from Australia, from the Murdoch University in Perth, some researchers are complaining that solid-state drives are making forensics, traditional forensics, difficult because the logic in the drives sort of - the firmware in the drives is altering the contents without any external intervention, causing these drives to lose data that forensics people would like to be able to recover. So I've decided, since there was so much interest that has been expressed by our listeners, that I ought to dig into this stuff and what these researchers have done and cover it in detail in a podcast.

So I wanted to let our listeners know that I'm going to, the issue of the secure erasure from SSDs.

And many people also tweeted in response to my comment last week, Leo, you'll remember, about one listener who was annoyed that the podcast was so long. Well, we got...

Leo: We got a response back.

Steve: Oh, not "a." I mean, many people took the time to say, Steve, I love it the way it is. Love you and Leo bantering when you do. Listen for information and entertainment. Go for it, and please don't feel like the guillotine is going to drop if you talk a little longer. So I wanted to acknowledge and thank everyone who thinks we're doing just the right thing here.

Leo: Yeah. I mean, this is the show with the least fluff of any show on the network. You could rightly complain that other shows are padded with BS. But not this show. This show is dense with material.

Steve: Well, I have a little BS.

Leo: Okay.

Steve: Here's just...

Leo: Once in a while is fine.

Steve: I'll ask our listeners to indulge me because this was really neat. A listener of ours named Kent Nelson referred us, meaning GRC, to a posting he found on MediaSmartServer.net. This was posted February 24, 2011, at 3:12 a.m., so somebody who was up in the wee hours of the night. And this is on a board under Windows Home Server troubleshooting and support. And so this person whose name I don't know, but I thank him, said, "I came to this board because I have a particular problem with my Windows Home Server. People were helpful, and I wanted to pay back by contributing a little. Surfing through the topics, I see the same themes recurring over and over again, themes that I recognize from my day job. It would be a game of Whack-a-Mole to answer each of these individually. So here is a piece of advice to all.

"Firstly, what is my day job? I run an IT help desk company focused on the home market. Our workshop processes a vast number of PCs from every manufacturer and with every conceivable configuration. When you deal with thousands upon thousands of machines, patterns start to emerge. The pattern that I am recognizing from your posts is disks with health problems. Not the blindingly obvious, disk-has-stopped-working-altogether type of health problems - the system itself will tell you about those - the much more subtle problems created when a drive starts to have low-level problems.

"Modern disks are miracles of engineering. It is a true wonder that they work at all. In fact" - this guy sounds like me, but it's not me. "In fact, when you look under the covers, you can see just how close to the edge they're actually operating. These things are throwing millions of errors as they try to read data, fail, and have to go back and try again. None of this is surfaced to the operating system. You have to use special software that talks to the SMART subsystem on the drive to get at such data. If a drive has a problem with just a tiny part of its surface, the performance of your computer can fall dramatically as the drive keeps trying to get the data.

"A good number of those posts with folks seeing machines behaving in flaky ways, or working one time and not the next, sound like disk problems to me. This should hardly be a surprise. With your Home Servers, you guys are dealing with many more drives than people with a single drive in a single PC. You are bound to run into more drive problems. The problem is, the operating system gives you no visibility of such problems. It simply waits for the drive to do its thing. The only fault you're ever going to know about with the operating system is complete failure. All the subtleties up to that point, and you can be sure there were many, will be lost.

"There are tools for monitoring the SMART subsystems on your drive, but we have found them to be of very limited use in the workshop. I only know of one tool that really goes to the depth of the drive and corrects these faults. It's called SpinRite." And he says (www.grc.com). "I should say this post is not an advertisement for the product. I have no connection with the vendor. If I knew of alternatives, I would list them here, but I do not. To the best of my knowledge, this is the only tool that does this, and it is extensively used by professional workshops throughout the world. Certainly, it gets run on every spinning drive that comes through our workshop. He says: (It is not suitable for SSDs.)"

He says: "I have personally seen it bring back to life nonbooting PCs, recover data that was thought to be beyond recovery, and speed up systems to no end. And I have seen this numerous times. There are techies who say that such a thing is impossible and could never work. But they are also the people who have never tried it. Do not take my word for it. Ask around. The only downside is it is not free ($89). However, to folks like you, with terabytes of data to protect and manage, it is a reasonable investment. On my own personal kit I run SpinRite on any new drive before deploying it, then again after each six months of use. I get longer life and much lower failure rates.

"I suggest you consider running this product on any PC or server that displays any odd behavior. You need to extract the drives from your Home Server and connect them directly to the motherboard of a spare PC because you need to boot from a CD to run it, which is not an option on a headless server. It will work via a USB adapter, but this is very much second-best, so plug the drive into the motherboard. It takes an age to run on large drives, but there is no way around that when you read the documentation on what it is doing. Do not be put off by their website, which is very amateurish."

Leo: No, it is not.

Steve: Well, that's what he wrote.

Leo: Okay.

Steve: "This is a geeky product for geeky people that's been around for more than 20 years now. I apologize that this post is unbalanced in that it is focused on a single product, which makes it feel commercial. If foreign members know of anything else which can perform these tasks and has a good reputation, then please comment, and we can add some more balance. I suggest the moderators consider making this a sticky post, as with an audience like this, managing lots and lots of big drives, disk health will be a crucial topic. You will also save yourself a lot of time trying to whack each of those moles."

Leo: Yeah. Boy, that's - you must be thrilled about that. I mean, that is great. And by the way, I don't know what they responded in the forum, but I don't know of anything that's anything like SpinRite. I mean, it's...

Steve: No, there was nothing. There were some people who said, yeah, we agree. And they were some Security Now! listeners, too, so that was neat.

Leo: It's the one and only. I mean, I suppose, as SSDs become more prevalent, it may be the last of its kind; right? I mean...

Steve: Yeah. Yeah.

Leo: I can't imagine at this point, I mean, I don't know how much longer spinning drives will last. I guess we've got another decade or so. But...

Steve: Oh, Western Digital just bought Hitachi. Did you see that in the news?

Leo: No, I didn't see that. Interesting. Hitachi bought those IBM drives.

Steve: Yup. Hitachi bought the technology from IBM. Now WD has acquired from Hitachi. So it's WD and Seagate, pretty much.

Leo: Wow. Wow. That's too bad. I liked those Hitachi drives.

Steve: Oh, that was absolutely my favorite. When I could choose, that's what I - yeah, I'm with you, Leo. They were great drives.

Leo: And now, let's talk about Stuxnet.

Steve: So what we have is, without argument, a true cyberweapon which was, over the course of about nine months from the time it was first seen to the last version that was seen, was under development. Symantec called it the most complex threat they had ever analyzed because of the number of different functions that it contained and also the fact that it was very cross-platform. It was, or is, because it's still out there a little bit, but it is a Windows-based worm, but it's designed to infect non-Windows-based systems. Many things are absolutely no longer in doubt. It cannot be doubted that this was directly targeted at the Iranian nuclear enrichment project. And I'll explain exactly why we know and how we know what we know. But it contained multiple zero-day exploits bundled in a Windows rootkit to hide itself from anyone. The first ever PLC, or Programmable Logic Controller rootkit, that had never been done before. It incorporated antivirus invasion techniques that I'll detail in a minute, where it literally looked to see what AV tools were in the system and knew how to get around them by version number.

Leo: Wow. Oh, wow. Talk about targeted.

Steve: Oh. It had, well, and what that means is, think about it, it means the people who developed it ran it in these different AV environments and watched the AV tools capture it. See, because one of the things it needed to do was it was trying to remain hidden. So, for example, after it replicates itself three times from a USB stick, it removes itself from the USB stick.

Leo: Oh, wow. Oh.

Steve: To minimize the chance of discovery, it figures, okay, I have spread onto three new systems, me, the USB stick. So I'm going to now - Stuxnet sees that because it's logging and recording what it's doing, and then it deletes it from the USB stick so that someone later wouldn't see it and wonder, whoa, wait a minute, what's this? So, I mean, it's all of this stuff. It's got process injection and hooking code that allows itself to insert itself in other processes in the machine; an array of network infection techniques, including a peer-to-peer technology that allows it to spread within local area networks; and a command-and-control interface. It connects to a couple of domains that I'll describe in detail in a minute, in order to report on its existence and to give those domains the opportunity to update the code. So essentially a binary package comes back which is actually encrypted. It's decrypted and then executed in order for Stuxnet to evolve over time.

So it's, functionally, it's able to self-replicate through removable drives, as I was saying. And that exploits a vulnerability which Microsoft knew about. And we've talked about it, it was that .LNK vulnerability that where just - then you'd have to open the link, a shortcut. Just viewing the shortcut in Windows Explorer could cause that file to execute by malforming the way the link file was made. And the rootkit which is hiding this knows exactly how many bytes long the file is; and when Windows Explorer attempts to retrieve that from the directory, the rootkit says there's no file here. So you just don't see it, even though it's sitting there on the drive.

So it's also able to spread through the LAN using a vulnerability that was also known for some time in the Windows print spooler in order to - so everyone has this service running in Windows by default. The LAN is a trusted environment. So unless those Windows machines were patched current, they would have this problem. Oh, which is a perfect example for one of the questions we were asked last week. Remember the guy whose company had 15 machines behind a "Windows Server," and they were back on SP2, and no one was patching them. And he said, you know, is this a problem? Well, here's a perfect example of where machines on a LAN have visibility to each other, and the Windows firewall protects you from WAN-based things, but because Microsoft wants to make things easy, like filesharing, does not protect you from LAN-based threats to the same degree. So if you're not patched, if you've got this Windows print spooler service listening, then Stuxnet would have been able to infect all the machines on that network.

And there's an SMB exploit, another well-known problem in the server messages block, the so-called file and printer sharing service, which Stuxnet also knows. So machines that were kept really current would have been safe because these were known and patched vulnerabilities in several cases. But there were, and still are, even today, Stuxnet is using some privilege escalation exploits which have never been made public, which it uses in order to get around these AV devices. So it copies and executes itself on remote computers through network shares.

And Siemens has a version of Windows called WinCC, which runs something called Step 7, which is their - it's all Windows hosted. And this is sort of the programming and code-writing and debugging tool to which you connect Siemens-based programmable logic controller devices in order to sort of download the code that you write. PLCs are programmed in sort of a - they have, like, an assembly language and also sort of a simple, step-based, basic language in order to tell them what they want to do. They're pretty simple-minded. But so you do all your authoring of this stuff on a Windows-based machine, then hook up the device and download it into the PLC. Stuxnet is able to update itself through this peer-to-peer mechanism.

So through using remote procedure calls, RPCs, Stuxnet sets up a server when it installs itself in a - when it infects a machine, and then sends out a broadcast for any other machines to see if they are of a later version. And, if so, they share their updates with older versions of Stuxnet. So it's constantly keeping itself up to speed. And it exploits a total of four unpatched Microsoft vulnerabilities, two of which have never been disclosed publicly, as I mentioned before.

Okay. So what's significant about this, when you look at how comprehensive it is, is that it could never have been designed blind. That is, this is just - this is not something that script kiddies, no matter how much they want to, could create. In order to pull this off, you need, first of all, essentially schematics of the target. Somehow, someone got, through information leakage, very detailed description of what it was that was going on in Iran's nuclear enrichment program. And of course that's not information they were letting go of. We know that because the targeting side of Stuxnet only fires when it sees a specific configuration of frequency converters tied onto this programmable logic controller which matches the fingerprint of what was going on in Iran.

The problem with Stuxnet is that it's a little bit blunt in that it is a propagating virus. A hundred thousand copies of it are, like, infected Windows machines all over the place. So although it was dispersed in a targeted fashion that I'll talk about in a second, because of these abilities it has to propagate, it got loose from the targeted companies, the five companies with connections to Iran that were infected with this. And it got out into the wild.

Well, we wouldn't want this thing infecting our own nuclear power plants or opening the floodgates on the Hoover Dam or anything else. I mean, programmable logic controllers are used for all of these things. This is like the way process control systems are run. And so you don't want to let something loose that is this powerful that is going to misfire.

Leo: That's like weaponized anthrax. You've got to have some sort of protocol.

Steve: Yes. And so what that meant was that the designers of this thing had to know exactly what it was going to find if it could get into the enrichment plant. They had to know exactly what was there because, I mean, there were versions of it all over. I mean, it was found in thousands of other Siemens systems. So this thing, I mean, this was - this upset a lot of people who...

Leo: Oh, yeah.

Steve: Because, I mean, this got into many of these Siemens PLC systems. But because the equipment that it found connected to the programmable logic controller didn't exactly match what it was designed to find, they didn't do anything malicious in those cases, thank goodness. But so you have to know exactly what your target is. And then, as I had mentioned in a prior podcast as information began to come out, I remember saying to you, Leo, a few months ago, somebody had to actually have this equipment. I mean...

Leo: You called it. You called it totally.

Steve: You had to set it up. You had to, I mean, you don't just write code and say, well, hope this works. I mean, all of this had to be prototyped. So you had to have frequency converters and basically mock up what is in Iran in a lab somewhere in order to write the code to make this go. So basically, as Symantec put it, a mirrored environment had to be created in the lab. Also remember that this thing, in order to work, it needed to get into the kernel in order to set up a rootkit to protect itself. It needed to have digitally signed drivers. And we know where they came from, remember? They came from Realtek and JMicron, two companies in the same industrial park, same physical location. So it is believed that some agent broke into and physically compromised those facilities to steal their private keys for their credentials.

Leo: Wow. There's a novel here.

Steve: Oh, I know.

Leo: I mean, what a book.

Steve: It really is. I mean, this is real. You couldn't, I mean, this is - it's incredible. So some agent, you know, covert, undercover, in the middle of the night, went into RealTek Semiconductor and JMicron and did whatever they had to do to get their private digital signing keys and made off with them so that the drivers could be signed for this to all work. So, I mean, there are so many facets to this.

Now, the problem with these PLC-based machines, these programmable logic controller authoring machines, is it is understood that security is a concern. So they are never directly connected to the Internet. So the designers of Stuxnet understood that they were not going to infect the machine. But think about it. As a consequence of not being connected to the Internet, you have to get data in and out of them. So it's thumb drives. Which is the infection vector. If you're going to have a standalone machine because you're worried about security, well, you're going to use thumb drives.

And so a lot of attention in Stuxnet is paid to infecting removable drives, protecting their contents, keeping the contents invisible. And it is believed that essentially that strategy is what worked, that machines that were connected to the Internet got infected with Stuxnet and then, in the normal course of transferring data, updating files, here you've got this machine running your nuclear enrichment facility, and you're all proud of yourself that it's not on the Internet, so nothing can get to it. Yet there's a new version of the PLC software. So you download it over on this machine...

Leo: Oh, boy. Oh, boy.

Steve: ...load it onto your thumb drive, and then bring it over to the not-on-the-Internet Iranian enrichment plant controlling computer, and bang. That gets it infected. So when Stuxnet arrives in a new machine - and I have it written down here somewhere the domains that it queried.

Leo: I should look. I have your notes.

Steve: Here it is. It's mypremierfutbol.com, so www.mypremierfutbol.com and www.todaysfutbol.com are two servers which originally pointed into Malaysia and Denmark. When the worm was able to get itself installed, it would look up the IPs of those DNS domains and send a package of sort of status, including its log of its entire history of infection. It had a timestamp, information about the OS version, and additional information, and that log. So over the time that Stuxnet was known about, Symantec was able to collect over 3,280 unique samples, individual instances of Stuxnet, each with a different log because each log tracked basically the lineage, all the ancestral versions. As it had infected one machine after another, it kept appending to this log.

What they know as a consequence of being able to mine these logs, this 3,280 different instances of Stuxnet, is that there were three events targeting exactly five organizations, each having a presence within Iran. From those three events, targeting five organizations, 12,000 infections can be traced back to exactly those five organizations. So basically - and we don't know how Stuxnet was planted in those organizations. Could have been a conspirator. Could have been emailed in. Somehow they got within those organizations.

The first organization, and they remained anonymous in this report, was targeted twice, in June 2009 and then again in April 2010. The second organization was targeted three times, in that June '09 attack, the second one in March of 2010, and then in May of 2010. The third one was targeted once, the third organization targeted once in July of 2009, as was the fourth organization. And the fifth one was targeted once in May of 2009, but had three initial infections because the same initially infected USB drive was inserted into three different PCs. Oh, yeah. So they were like, it was targeted once, but it was, like, salted in three different locations within that organization. And so Symantec was able to track back all the way back to those very original three instances within that fifth organization.

The shortest span of time between the compilation of Stuxnet, where it was, literally, its source code was compiled, which inherently binds some date information into the code, to an initial infection was 12 hours. So this thing was built and, in at least one case, within 12 hours an infection was planted. The longest span between compilation time and infection was 28 days, and the average was 19. So this whole thing took place in the latter half of 2009 and the beginning of 2010. And so they know from, again, looking at these logs, that there were three attack waves: essentially June 22 in 2009, March 1 in

2010, and April 14 in 2010. And Stuxnet was getting better. The March 1st attack was a much more capable worm than the June one.

So if you can sort of put yourself in the mindset of the people who were doing this, who designed this, they had a goal. And they had a system which was providing them feedback. And so that was a mixed blessing because obviously Symantec is able to determine everything they have because of the feedback which the worm provided to its command and control servers every time it propagated. But you could see also that, while it was unknown, before it became known, this was vital information for the designers because it allowed them to profile the performance of this weapon they had written in the wild, and these were spear attacks. I mean, they were somehow sending agents into Iran or into affiliated companies and planting Stuxnet there. We know that because of the dispersion of the virus. Of all infections of Stuxnet globally, 58.31 were in Iran. 58.31 percent, sorry. 58.31 percent.

Leo: That's pretty effective.

Steve: So, yeah. Like...

Leo: Good job.

Steve: Nearly 60 percent were in Iran. But that's just machines infected. So that means it wasn't released in Santa Clara and all went there because all the machines between here and Santa Clara would be infected. I mean, so the point is that it started there somehow. Somehow it was planted in that location, like near to its goal, and then spread locally. And of course due to the fact that it was a worm, and used unpatched but known vulnerabilities of Windows, it did get loose. Yet as I said, the weaponized end, thank goodness, was so tightly targeted that it didn't do damage to all the other Siemens Systems that it sought out and did infect, 18 percent in Indonesia and 10 percent in India. And then it fell off.

And also the Siemens Step 7 system that I mentioned, of the infections, 67.6 percent of the Iranian infections had Step 7 software installed. So it was, again, it was seeking out and looking for these process control-based systems. 8.1 percent in South Korea of the infections had Step 7 installed, 5 percent in the USA, and 2.18 in the U.K. So it did, for example, in the U.S., 5 percent of the infections of Stuxnet were Siemens-based systems. So it was infecting U.S.-based process control systems. And the good news is the flood gates of Hoover Dam didn't get opened as a consequence. So...

Leo: Well, I think it's really clear that, well, we know - in fact, I think we know who did this now because there have been some revelations. But I think it's pretty clear that they were heavily targeting.

Steve: Yeah, well, and this evidence, I mean...

Leo: And effectively.

Steve: I guess what I find so interesting is that, if you really take advantage of the information coming back to you, you can, as you said, Leo, this is a plot. I mean, we can work out what had to happen in order for this result. In order for the drivers to be signed with good driver certificates from two innocent companies, somebody had to go and break into them and get their private keys in order to sign the drivers. Stuxnet the virus is aware of Kaspersky KAV versions 6 through 9, the current McAfee products, AntiVir, BitDefender, eTrust, F-Secure, Symantec and Symantec Common Client, ESET's NOD32, and Trend's PC-cillin. It has code in it to specifically see that those products are in the system.

And remember, one of its priorities is stealth. It very much wanted to get its work done before it was being found. So what it did was it would look in the system to see if these things were present. And, if so, it would look at the EXEs to determine the versions, and had version-specific behavior, so it was designed to go underneath the detection. And in several cases it did that by using either one of two because it ran on all versions of Windows, XP through Win7, not earlier than XP. It used vulnerabilities that had never been published for getting admin privileges, if it was not running with admin privilege, and it used those in order to place some kernel-level hook games in order to install itself into processes in a way that specifically would not be detected by these intrusion detection systems that were designed to detect exactly this behavior.

The rootkit that it installed, even with these tools, with these AV systems in place, it was able to install a rootkit, robustly and reliably, and filter the API calls that Windows was making to the kernel such that, if a file with a .LNK extension was going to be enumerated in a directory search, and the file was 4,171 bytes long, the rootkit would just remove that from the listing because the malicious link files that Stuxnet used were obviously 4,171 bytes long. And if a file was named "~WTR[FOUR DIGITS].TMP," whose file was between 4Kb and 8Mb, but the sum of those four digits, modulo 10 was zero, then that file would also not appear.

Leo: And why would that - I don't understand what the point of that is.

Steve: And so, well, so this was - Stuxnet needed some flexibility in its payload. So the link files wouldn't be seen, but it needed other files from time to time that it might need to hide. And so what it would do is it would design - it designed the rootkit filter such that - sort of with a pattern match. So that if the pattern was "~WTR[FOUR DIGITS].TMP, and if the sum of those digits added up to zero modulus 10, then, that is, added up to 10, 20, 30, 40, for example, or zero, I guess, then that triggered the rootkit not to show that file.

Leo: So they could hide in plain sight.

Steve: Yes.

Leo: It would be an obvious rootkit file.

Steve: Precisely. And it had to coexist. So this was on the thumb drive or on the system where Stuxnet was installed. It had to coexist with other things. And it would look, when it was going to jump onto the thumb drive, it would verify that the drive had not just been infected by comparing the files with the current time. It would verify that the infection source was less than 21 days old. Meaning that after three weeks...

Leo: Wow, it expires.

Steve: Yes. It would stop trying. It would just...

Leo: So cool.

Steve: It was. It was just brilliantly designed. So the point is, again, it was trying to not get discovered. So it gave itself three weeks, on a given system, it gave itself three weeks to infect all the drives it could. And after that point it would go silent and just not do it anymore because, again, it figured, hey, if I haven't done it within three weeks, then - and who knows what the developers knew about the protocol being used in Iran's nuclear enrichment facility. They might have known, for example, that something happened every two weeks or every week or something. So if they were able to get onto the machine that was one step away from the machine doing the development and controlling the programmable logic controller process control stuff, if they could get to that machine, and they knew that, like, there would be some thumb drive-based communication between those two within three weeks, and if not, then they're just not on a machine where that's going to happen.

Leo: Sounds like they really knew what they - not only what they were doing, but where they were going to be. I mean, this was so clearly targeted.

Steve: Yes. And the drive, the thumb drive, had to have had at least three files and five meg of free space because you wouldn't want to run into, I'm sorry, you don't have enough room on your drive to hold our rootkit and our Stuxnet virus. So one of the files, WTR4141.tmp, and if you think about it, 4141, that adds up to 10, which is zero modulus 10, that would - it was sort of like the advance guard that was a small bit of code that hid its companion file, ~WTR4132. And again, 4132, that sums to 10. So that's zero modulus 10. And that contained the entire Stuxnet payload that jumped over onto the thumb drive.

When they finally got there, one file, which was a DLL on this Step 7 PLC programming computer, the DLL was s70tbxdx.dll, that got renamed to, instead of the last characters being xdx.dll, it got renamed to sxs.dll. And a replacement s70tbxdx.dll, which was the PLC rootkit, it was installed. So essentially this DLL that was - it's very comprehensive. It has, like, 140 different, what Microsoft calls "exports." Those are, like, functions that the DLL can offer. The replacement file didn't duplicate all of those. For almost all of them, it simply forwarded those calls to the fake DLL to the real one because it knew what it had renamed the real one.

So when 135 of those different functions were called, it handed them off to the original DLL to work correctly. But the few that it needed to alter allowed it to intercept those functions on their way to the Siemens programmable logic controller and essentially add its own code to the code that was being downloaded and arrange for that code never to be visible, never to be seen. And so everything we talked about was for just the sake of getting a bit of code, custom-written code, appended to the front of the code controlling the PLC. And it also had to be that code that looked around at what it was connected to and knew whether to do anything or to stay inert. And so that's the history of the world's first, I mean, truly weaponized Internet worm.

Leo: Do you think the people who wrote this were security researchers? Virus authors? Do you think they took existing code and modified it? I mean, it sounds fairly sophisticated.

Steve: Anyone who...

Leo: Maybe, like, they contracted out, I mean, look. Israel did this; right? We know that.

Steve: Yes.

Leo: We've heard that in fact they had exactly the same setup intentionally. It was pretty clear.

Steve: Yes.

Leo: And they had, of course, they had means. They had the motive because they didn't want Iran to have a nuclear bomb.

Steve: Yeah, I would say they probably did it with help. I mean, I believe that there are resources in the U.S. And, I mean, we certainly would not be hostile to the intention of keeping Iran from getting a nuclear bomb. And the argument was that that's what they were using this nuclear enrichment for, despite their denials, saying that they just want it for electric power generation. So you have to think that, within the NSA, within our own government, and sort of shady organizations, I mean, we were talking about, what is it, HSGary is the company? Or HSB? I can't remember the name of the company [HBGary]. But it's a major sort of hacking contractor that...

Leo: Right. This is what they do.

Steve: ...organizations in Washington use for their own purposes. And it was those guys who created a device, a Firewire device that allowed the guts to be sucked out of a computer just plugging it into the Firewire port to DMA and copy the contents out. And that came from that company. So there are even commercial organizations - it's where our tax dollars are going - that have this kind of competence and are able to participate in projects like this. And you know, Leo, how many times in the early days of this podcast, like Episode 2 and 3 and 4, when we were looking at sort of interesting Windows viruses and worms and things and commenting, isn't it nice, I mean, and aren't we really lucky that they're not malicious?

Leo: Right, right.

Steve: So many of these things - and I scratch my head. It's like, why - okay. I'm glad that they're not doing bad things, but...

Leo: Why not?

Steve: ...who's going to all the trouble of creating them, just to sort of have them float around out there?

Leo: They're practicing.

Steve: That's what they did. They just floated around out there.

Leo: I think I remember the very first virus was written by that guy Morris...

Steve: The Morris Worm.

Leo: ...the Morris Worm, just to see what would happen. He wasn't malicious.

Steve: No.

Leo: It escaped.

Steve: And it really hurt his reputation a lot.

Leo: Yeah, his father was a very famous security guy.

Steve: Exactly.

Leo: And so I think that some of this in the early days was just people, you know, hackers are curious. And I could see how tempting it would be to say, you know, I could create something that would spread itself. Wonder what would happen? And just do it. I can understand that.

Steve: And within the white hat community we still hear echoes of, well, boy, you know, why can't we write a disinfector worm?

Leo: Yeah. Yeah. Remember that? Yeah.

Steve: It's like, I know you want, yeah, I know you want to. But sorry, even if you're altering someone's machine, and you think that's a good thing, you're doing it without their permission.

Leo: Well, and I think ultimately, while it sounds like Stuxnet was pretty carefully crafted not to do harm and was very specifically targeted...

Steve: Oh, danger.

Leo: ...it's still a bad, bad, bad, bad, bad idea.

Steve: Yeah, and Leo, imagine if it had misfired.

Leo: Right.

Steve: Imagine if there had been, literally, collateral damage from the thousands of Siemens computer systems. I mean, it went in, and it replaced a DLL. I mean, we depend upon these process control systems to run big plants. And it was in there replacing a DLL in order to get access to the programmable logic controller, to then go add code to it, and hope that it didn't do the wrong thing. I mean, it was gutsy.

Leo: Look, we know there is no such thing as perfect code. And we also know that programmers have a little bit of hubris. And there's probably not a programmer alive who thinks he can't write perfect code.

Steve: Yeah, and not one alive that ever has.

Leo: Yeah. If it were that easy, everybody, anybody would do it. What a great subject. Show notes, as always, are on Steve's site, GRC.com. And I put the show notes in our TWiT wiki every week. It's the one show I actually do that because you have such good notes. I always make sure they're in the wiki at wiki.twit.tv. You can get 16KB versions of the show at GRC.com. Steve has transcripts, too, which is really great. And this is the kind of show that I could imagine a college class or somebody who's teaching security might very well want to get people to listen to or read because it's so interesting. John is asking, was there an Easter egg in Stuxnet?

Steve: You know, actually I skipped over that, but...

Leo: There was?

Steve: Yeah, well, there were some odd things. Like there were some codes which, if you took it to represent a date, was the birthday of somebody famous in Iran. I mean, it was those sorts of things, really subtle. And in Symantec's report they made a point of saying, look, this is what Wikipedia says about this, I mean, about this particular collection of characters. But remember, the people doing this would have strong reasons to be pointing fingers to someone else. So we absolutely couldn't take that as ego out of control, but rather just additional subterfuge.

Leo: Red herring. It could be a red herring.

Steve: Precisely.

Leo: Wow. Oh, I want somebody to do the - some intrepid journalist to do the research on this and write a book. What a fascinating story that must have been. I don't think we'll ever know because...

Steve: No. Because, I mean, it really, oh, thank goodness it didn't misfire, Leo. As I really came to understand what this thing was, I was thinking, oh, goodness. I mean, this was really - this was potent.

Leo: Well, I can guarantee you in future Security Nows we'll be talking about worse. Absolutely. Sad to say.

Steve: Well, we'll be here.

Leo: Yeah. 291 episodes in, and no sign of stopping. Steve Gibson, he's a machine. Visit GRC.com for your copy of SpinRite. You've got to have it. If you've got a hard drive, you've got to have SpinRite. And of course if you've got questions, we do a feedback episode every other episode. And now is the time to go to GRC.com/feedback, ask those questions. Maybe you'll get included in next week's episode. And tune in every Wednesday at 11:00 a.m. Pacific, Apple permitting. Thank you for moving last week. 11:00 a.m. Pacific, 2:00 p.m. Eastern time at live.twit.tv to watch. Steve, thanks so much.

Steve: Thanks, Leo.

Leo: We'll see you next time on Security Now!.