Listener Feedback #94

Description: Steve and Leo discuss the week's major security events and discuss questions and comments from listeners of previous episodes. They tie up loose ends, explore a wide range of topics that are too small to fill their own episode, clarify any confusion from previous installments, and present real world 'application notes' for any of the security technologies and issues we have previously discussed.

High quality (64 kbps) mp3 audio file URL: http://media.GRC.com/sn/SN-253.mp3
Quarter size (16 kbps) mp3 audio file URL: http://media.GRC.com/sn/sn-253-lq.mp3

Leo Laporte: This is Security Now! with Steve Gibson, Episode 253, recorded June 16, 2010: Q&A #94.

It's time for Security Now!, the show that covers all of your security needs. I'm Leo Laporte with our security guru, Mr. Steve Gibson of the Gibson Research Corporation, GRC.com. Hey, Steve.

Steve Gibson: And you have physically, viscerally demonstrated the first news item for us to discuss by...

Leo: Yes, we're starting 45 minutes late.

Steve: ...getting both of your Macs updated with the latest fixes from Apple.

Leo: Okay, just a tip for those of you who are in broadcasting. Probably not a good idea to update your operating system right before you want to start a show.

Steve: Yeah, not in the case of a 300-plus megabyte download that then has to replace a ton of files and...

Leo: Well, that was the interesting thing. The download happened like that. It took very little time to download. It was the reboot.

Steve: Yeah.

Leo: And that makes sense. It's doing a lot of work behind the blue screen there.

Steve: So we have Security Now! Episode 253, Q&A #94.

Leo: Wow. Hard to believe. Ten great questions from you, our audience. Also of course security update news, as we've kind of hinted at.

Steve: Another busy week in the disaster of this security industry, Leo. Just, oh, my goodness. More AT&T hijinks. We've got a zero-day vulnerability from Microsoft. We've got Adobe still squirming around. All kinds of stuff.

Leo: All right. So, tell me about this update, Steve. What did I just do? By the way, for those of you not watching at home - and you can watch video of this now - Steve, I just noticed you're wearing your hacker shirt.

Steve: I am. Yup. I figure after The Portable Dog Killer episode, I'm entitled to be a hacker. And I actually have a little story from a witness, a second-hand witness to that, that I found in the mailbag today, that I thought our listeners would get a kick out of. I wish today we had more security updates than we do.

Leo: We need more.

Steve: We do. We're hanging out here at the moment. What we did get was a relatively major update for the Mac OS, OS X. I'm not sure what cat this is. You and I are running different cats. I have...

Leo: I'm on Leopard. You're on Snow?

Steve: ...snow something or other.

Leo: You're on Snow Leopard?

Steve: Snow Leopard. And so the Snow Leopard folks are up to v10.6.4. You're 10.5.something, I guess.

Leo: I can't remember. Let me look real quick because I didn't upgrade to Snow Leopard because I didn't see any reason to. And it did cause compatibility issues with things like our audio drivers. 10.5.8 if you're on Leopard.

Steve: Okay, 10.5.8. And for both of us a several hundred meg download. Mine was three something.

Leo: Wow. 224 on mine. But I already had Safari. And I presume it's smart enough to look and say, oh, you've got Safari 5, we won't download that.

Steve: In my case I had Safari 5, as well. And so it was 330, or I think it was 313 megs. So it was 23 security fixes, Safari 5 if you didn't already have it, and then just a sort of a handbag, handful, random sampling of various random bug fixes. Nothing really significant there. But everyone who's got Macs ought to update because there were 23 security fixes, which I will not drag everyone through an enumeration of. Just all kinds of good stuff that we want.

The reason I wish we had more update news is that Adobe has now fixed the Flash problem that we have talked about, but declared that they will not be fixing the PDF vector for this until the end of the month. So we have an actively exploited, in the wild, serious, known to the hackers, PDF vulnerability which we're going to get no cure for for two weeks. It is possible to do what we talked about last week, which is to delete or rename this DLL in Windows systems which is actually what - it's the Flash player that Reader brings along. And I'm blanking on the name. Auth something dot dll [authplay.dll]. I blogged about it on my steve.grc.com blog a couple weeks ago.

Leo: I'll go look. I'll go look.

Steve: And so renaming that is probably a good thing to do, knowing now that Adobe has formally declared that they're not going to have a fix for us for another two weeks. So there's that. And Leo will get the name here for you.

Leo: I'm looking right here on your site, steve.grc.com - a-u-t-h-p-l-a-y, authplay.dll.

Steve: Authplay, yes, authplay.dll. I'm recommending that you search your system for that and just change it to authplay.xxx, for example, which will prevent it from being found. If by chance you then opened a PDF that had Flash in it - and I don't know why PDFs would have a Flash in it. But the point is that PDFs are Flash-enabled by default. And disabling the Flash feature, which is available in the UI, doesn't prevent this from being a problem. So go figure that. But renaming this authplay.dll to .xxx will, if you were to open a PDF with this that was trying to invoke Flash, would just cause it not to function. The PDF itself would fail to open. It's like, okay, probably that's a good thing because it was more than likely malicious. So you could wait for two weeks. Be careful about what PDFs you open, or just rename this authplay.dll in order to be safe in the meantime. Then when Adobe's fix comes out, it'll just give you a new copy of authplay.dll with at least this known problem fixed.

Since we last spoke on the podcast, a new vulnerability was revealed. What happened - this is not technically a zero-day vulnerability. I referred to it as such on my blog. And I blogged about it on the 11th, which was when this became known. What happened was - and this is somewhat controversial - a Google security researcher, who claims that he was not doing this under the auspices of Google, named Tavis Ormandy, who's been

known for releasing in a responsible fashion news of other vulnerabilities, informed Microsoft five days before he told the world of a vulnerability that he discovered in Windows XP and 2003 Help System.

So first off, if you're not running XP or 2003 Server, you don't have a problem. This is an XP/2003-only vulnerability. So Tavis notified Microsoft on the weekend, actually, like on a Saturday, and then gave them five days' notice. And what that unfortunately did was, I mean, even if Microsoft had been able to respond instantly - and we know that they are substantial non-instantaneous responders, sometimes taking as much as a year to fix things that they know about. But the point is that we just had our second Tuesday of the month of June. So we're now, from this point forward, if Microsoft doesn't do anything out of cycle, waiting a full month. I mean, Tavis couldn't have timed this any worse. And only giving them five days' notice, then posting on a well-read security list all the details of the exploit, with demonstration code, in public, caused a lot of controversy.

And the problem of course is that he says he did this on his own time, not under the auspices of Google, despite the fact that he's a security researcher for Google. Now people are saying that this is like Google attacking Microsoft and not giving Microsoft sufficient notice, not doing the whole responsible disclosure dance where the researcher waits until the problem has been patched before going public with it and so forth. So that hasn't happened.

What we have now, since then, okay, so this was - I blogged about this on the 11th and immediately put up a workaround to allow people to protect themselves because I expected that this - this had all the appearance of something that would be jumped on quickly because it was in XP, no patch available. It was also trivial to exploit. And he gave a - Tavis gave a complete explanation in detail, showing code, of what it was he found and how to exploit it, with samples. And sure enough, we're now recording this on the 16th. And yesterday, on the 15th, we began to see this vulnerability being exploited in the wild.

So to all of our listeners, it's my most recent blog posting, so you can go to steve.grc.com. And since then Microsoft has created one of their quick Fixit button deals. You could also just go directly to support.microsoft.com/kb/2219475. So it's, again, support.microsoft.com/kb/2219475. Which will - and I link to that on steve.grc.com currently, which is the top blog on my blog, top posting on my blog, where you can get a link to there. And they'll give you a button that you press to turn this off.

What this does is the same thing that my blog posting recommended back on the 11th, which was there's a protocol handler, something, for example, if you clicked on a link that said ftp:// for File Transfer Protocol, or http://. Well, in this case it's hcp://, which is a URL-style invocation of the Help Center. So it's - HCP stands for Help Center Protocol. And it's a bug in that which is the problem. Well, there are some things in Windows that need that. So disabling this will break some random links in Microsoft's own help system, which they use within Windows to bring up the Help Center. But better that than being exploited with this vulnerability, which Microsoft may very well not get around to fixing for a month because we just had the second Tuesday of June. I don't know if this is going to raise to the level of them doing an out-of-cycle patch.

The problem is that everyone within the sound of this podcast will be able to fix this, but most people are now relying on Windows Update to keep their Windows current. And so this vulnerability is going to be hanging out there being actively exploited for maybe as long as a month. I can't, I mean, given Tavis's expos, it's hard to imagine that Microsoft could say they can't have it fixed in four weeks because he laid the whole thing out; and he laid it out for them, in fact, last weekend. So it's like, okay. My sense is this is worth

doing. Our listeners ought to protect themselves. But again, only if you're not up on Vista and 7 yet, only if you're still back on Windows XP. That's the only place where it's a big problem. And it looks like it is a big problem.

I learned via Twitter from Alejandro, whose twit handle is @microtwit32, that NoScript, the favorite script blocker for Firefox, quietly added support for tabnabbing. We talked about tabnabbing last week or the week before. Remember that that's an interesting exploit where pages that you're not viewing currently, for example in Firefox, can be changed in a way that, if you went back to the page, it could easily fool you to believe that your eBay session had timed out, or Google Mail session had timed out, or something saying, oh, please, reauthenticate. The idea being that the page changes when it's not the tab on top, so you're not viewing the page at the time, don't notice that it changed from something completely different to something that is spoofing one of the services that you are using.

It turns out that scripting is powerful enough now to allow a probing of the services you do use so that a sufficiently sophisticated script could figure out what it is that, like, what banking site you tend to use, and present something convincing on the tab that you're not viewing. So when you switch back to that, it's like, oh, look, my banking site says I need to log in again. So what our NoScript author did at v1.9.9.81 and since - I went back and looked through the update and feature notes. He quietly added a new option which is not - it does not surface to the level of the user interface. So it's not a button you can click on the UI. But if you go, if you put into the Firefox browser's URL field "about:config" and hit Enter, that will take you to a huge page of alphabetically sorted security and UI and every kind of option under the sun that basically governs in great granular detail the way Firefox operates.

The item you're looking for is noscript.forbidBGRefresh, as in background refresh. So again, it's noscript.forbidBGRefresh. Now, that can have a value of 0, 1, 2, or 3. 0 is no change of behavior at all, no blocking of background page refresh changes. 1, which is the default mine had been set to, blocks refreshes on untrusted, unfocused tabs only. Now, trust and untrust is relative to NoScript, that is, have you said that you trust this page, like Amazon.com, for example, or not. The setting of 2 blocks refreshes on trusted, unfocused tabs. I don't know why you would choose that because it doesn't block them on untrusted tabs. But setting 3 blocks them on both trusted and untrusted tabs.

And I changed mine to 3 because I can't really see a valid reason why, whether I trust a site or not, if I'm not looking at the page, I don't think it needs to change what I'm not seeing. And in fact I've noticed that I'm sometimes distracted when I notice a page that I'm not looking at is changing, is, like, refreshing. Some script timer timed out, and it's changing the ads on the page, or it's refreshing the whole page in order to get new content or something. Well, I'd just rather not have it do that behind the scenes. So I like the fact that NoScript now lets us prevent any nonfocused page from changing itself. Seems like a useful thing to do.

So again, in Firefox, "about:config" in the URL field. Then just, I think, in fact I'm sure that there's a search feature in that about:config page. I just scrolled way down manually because it was alphabetic. So noscript.forbidBGRefresh. And it normally is 1, so it blocks untrusted, unfocused tabs from changing. I changed mine to 3 to block both trusted and untrusted. I can't see any reason, I can't see any negative side effect from doing that.

There is one other option that you'll notice on the immediate succeeding line, which is noscript.forbidBGRefresh.exceptions. And for whatever reason he has Mozilla.org listed there, probably just as an example. So what that allows you to do is, if it turned out there was some site that was having a problem with being unable to refresh itself, or if

you just wanted specifically to allow specific domains the ability to override that, this gives you exceptions to the blocking rule, allowing them to behave as if you didn't have any prevention at all. So that's a cool feature in NoScript that we wouldn't know about if I hadn't received this nice twit note from Alejandro. And so I want to thank him for that. And I think it's useful for our listeners.

Leo: NoScript is such an amazing tool. This guy's just constantly updating it.

Steve: Yeah, he's doing a great job.

Leo: Yeah, yeah.

Steve: Then, in AT&T dog house, we talked last week about the mistake that they made by allowing their web service to return the email address given the so-called ICC-ID of SIM cards, which are in, in this case, the Apple 3G Tablet. Well, it turns out that that was sort of the first problem. When people who know GSM took a closer look at this, they realized there was another consequence that had not yet received any attention. There's another number, very much like the ICC-ID. This one's called the IMSI. The IMSI is supposed to be secret, whereas the ICC-ID is printed on the outside of the SIM card itself. It's on your receipt when you register a phone or buy a phone. The ICC-ID is not intended to be secret. The original concept for the IMSI is that there would be a database somewhere such that the ICC-ID could be used to securely query a database which would then return the secret IMSI number when given an ICC-ID.

It turns out that a number of the cell phone vendors, I know it's AT&T and T-Mobile and a couple of others, decided that that was kind of a pain to have to do that. So they decided to use a stunningly simple transformation, merely a matter of swapping digits around, essentially, that allows you to calculate the IMSI from the ICC-ID. Meaning that what was supposed to be, in the spec, a secure, non-obvious relationship for the sake of security, now becomes a matter of getting out a pencil and paper. And from an ICC-ID you can compute the IMSI.

So that becomes, I mean, and this has been known for a long time. Wasn't a big deal. Except that now we have the exposure of this 114,000 ICC-IDs, which were really just obtained by guessing what they probably were, since they're generally sequential. And so this hacker group that we talked about last week, Goatse, just wrote a script in PHP to guess all these ICC-IDs, using the AT&T server to confirm them and to return the associated email address. Okay, now we know that these - so we have some piece of information about the email address. Generally from the email address you can guess who it belongs to - rahmemanuel[@]whitehouse.gov, we know who he is, and so forth.

Leo: And why he was using that address is beyond me. Was he? No. I think it was a Gmail address.

Steve: Don't remember. But so we have their email addresses. Oftentimes you can tell who they are. Well, now we know that it's very possible to get the IMSI. So what does that give you? The IMSI is this information that is supposed to be secret. And through a formal API that's public because it's universal, you're able to query the GSM cellular network to determine the full account name of the owner, their phone number. This is

the information we talked about some time ago where you now have the ability to track them as they roam anywhere in the world. That is, you can determine which cell tower their phone is currently associated with. You can retrieve their voicemail. And if you are physically near them, which is now not difficult because you're able to determine which cell tower that they're at, it turns out it's possible to intercept their speech and SMS messages. Now, in the case of an iPad, which is not a speech device, it's a data-only device, there is no voicemail account. You're not going to have speech or SMS, probably, associated with it. So these don't represent such a big problem.

So, again, this is - to me it feels like, yes, a privacy concern, maybe a little bit of a tempest in a teapot because days ago when this news surfaced, again there was another whole flurry of oh, my goodness, everyone's pulling their hair out. I'm thinking, okay, well, it's unfortunate that the cell companies have associated the ICC-ID with the IMSI. They shouldn't have done that. They did it for simplicity's sake. It should have relied on a secure access to a back-end database so that you couldn't get the IMSI, even knowing the ICC-ID, because the ICC-ID is intended to be not super secure. You'd like to have the IMSI kept secret for all of those reasons I just enumerated. Basically it's a key into someone's current cell phone behavior at this point. So anyway, I wanted to cover it. A number of people wrote to say, "Hey, Steve, did you see this? What do you think of it?" My feeling is, yes, that's not good. It's not the end of the world. But that's what's going on.

Leo: It's good to get that kind of straight because it's sometimes reported as the end of the world.

Steve: It too often is. I think by, I mean, and in some cases I think people like it to be the end of the world. They're wanting to bash on AT&T.

Leo: People hate AT&T so much.

Steve: Yeah, I mean, I do, too. But, you know, still. So my feeling is, those are the facts. People can decide for themselves how they feel about it.

There was an interesting story that TheRegister.co.uk picked up that I wanted to share with our listeners because it's sort of - it's an example of what can happen, and it reinforces something that I've talked about before that I just wanted to refresh. So the Register story is about crooks, as they put it, siphoning a rather sobering amount of money, $644,000, from a New York City School District bank account.

Leo: That's terrible.

Steve: "The New York City Department of Education was" - and I'm reading from the Register - "defrauded out of more than $644,000 by hackers who targeted an electronic

bank account used to manage 'petty cash' expenditures, investigators said. The DOE's small item payment process account at JPMorgan Chase was supposed to be limited to purchases of less than $500, but an oversight by officials

allowed electronic transfers of any amount, according to investigators who probed the

theft. The crooks were able to perpetrate the scam for more than three years because education officials didn't bother to reconcile account statements on a regular basis."

Leo: You know, I reconcile my account statements. Why wouldn't - if they're - ugh. That's too bad.

Steve: "'It is difficult to understand how the DOE accumulated years of account statements, reflecting hundreds of thousands of public dollars spent to pay bills, but did not review them,' the report, which was written by Special

Commissioner of Investigation for the New York City School District, stated. 'A cursory examination would have shown that the charges were not normal school expenses.'

"The individual who headed the theft was Albert Attoh, who in April was sentenced to 364 days in federal prison after pleading guilty to bank larceny. He was also ordered to pay more than $275,000 in restitution and be on probation for two years following his release. According to the report, Attoh provided the account and routing information

to others so they could use it to pay student loans and invoices for purchases at Home Depot and other retail outlets. In return, Attoh demanded cash payments [from them]. Because DOE officials failed to block the use of electronic transfers, the account was wide open. All that was required was the account number and the bank routing [information]."

So I had mentioned quite a while ago that we were seeing - the security industry was seeing an increase in the level of this kind of electronic transfer fraud. It's some of the vulnerabilities that are opened by malware that gets on people's machines and is involved in their banking transactions. When I first saw this, I then made sure that, for my own company, that things were still in place that I had set up years before, which was to explicitly lock down our accounts against electronic transfer. It turns out that it's a little inconvenient for my operations manager, Sue, who has to physically write a check from one account to the other. But we don't do it that often. And I just wanted to share this example with our listeners and really encourage them to change the defaults, which is what these probably are, on accounts that they have that are relatively static, where they're not actively moving money around.

You know, our banking industry in general is wanting to automate itself. It's wanting us not to come into the bank. They'd rather use ATMs. They'd much rather that we did things online. Well, all of that is convenient for them, and it minimizes the level of service that they have to provide. But it comes at a substantial expense to security. So as a consequence, in general, accounts have these defaults to allowing this kind of fund transfer. Well, this is a perfect instance of real-world security where, if you do not actively need that feature, turn it off. And one of the problems is that, unlike, for example, fraudulent credit card purchases, where the credit card company stands behind your use of the card, and you have to sign an affidavit saying, yes, I never purchased all of this stuff that was not sent to me anyway, it went somewhere else, this is not the case in these kinds of cash transactions. When this cash is transferred off to somewhere else, it's gone. There's no one for you to appeal to. There's no one for you to get angry with. Your bank says, well, we're sorry, but we were just doing what we were instructed to do.

So I just want to make an appeal to our listeners to think about the way their accounts are structured. If they've got more than one, if they've got places where they park money or they park investments or that kind of thing, just make sure that your bank is instructed to turn off any of these automation features that you don't actually need, that

you're not using. It's increasingly risky, unfortunately, for these defaults to be on. And so it's, I think, worth taking a moment just to say, make sure you are in agreement with your bank about what they're allowed to do and what not, what requires physical presence in the bank in order to perform.

Leo: It really is true that there is convenience versus security. It's a balance beam.

Steve: Yes.

Leo: More convenient, less secure. Often.

Steve: Yes. It absolutely is. I did receive, shortly after last week's podcast, a tweet from a Dan Bowser that I got a chuckle out of. He's probably a Mac user, or maybe a Linux user. He's certainly not a fan of Windows. And so we, of course, talked as we always do about security patches and so forth. And so I looked up and saw this come in. So Dan wrote: "Every Windows machine has an unpatchable

critical vulnerability."

Leo: Oh, no, what's that?

Steve: "The power on switch."

Leo: Ooh, burn.

Steve: Okay. Okay. And I did run across a fun note in my mailbag today, while pulling questions for the Q&A, from Brad, who says, "Dear Steve, I work for a sizeable organization and am charged with using a popular disk-wiping utility, Kill Disk" - which is pretty well named, I think - "to erase hard drives in our machines before they are either redeployed or recycled." Glad to know that large companies have such a policy. And he says, "These old machines, and the hard drives in them, can be up to eight or more years old. On approximately one out of every 15 or so drives" - I thought that was an interesting statistic, too. "On approximately one out of every 15 or so drives, the wiping utility will hang at a certain point, unable to complete the 10 passes of the drive that we require to satisfactorily dispose of the data. When this happens, we have to spend the time and effort to physically destroy the hard drives.

"Recently I decided to try my copy of SpinRite on a drive where the wiping utility had gotten stuck. SpinRite ran at Level 2. DynaStat kicked in and resolved the hard drive's issues to the point that, when SpinRite had finished, the disk wiping utility was now able to fully run its 10 passes on the drive, saving me the time and trouble of physical destruction, and of course making the drive usable again. As a result, a purchase of four SpinRite licenses to give us a site license is now planned for when our budget comes up later this year. I first heard of SpinRite in the 'Rootkits for Dummies' book."

Leo: There's a book called "Rootkits for Dummies"?

Steve: "Rootkits for Dummies" - "...as a way to restore sectors where the rootkit NTFS hider lives."

Leo: Wow.

Steve: Okay. So get this. There's a problem with being unable to install the rootkit because it insists on going on a specific physical sector. And if that physical sector happened to be bad, oh, darn, you wouldn't be able to install your rootkit there.

Leo: Right.

Steve: So they said, oh, run SpinRite to fix the sector; then you'll be able to install your rootkit.

Leo: I love it.

Steve: Not quite how I intended SpinRite to be used when I was designing it. But there you go.

Leo: You have users in many areas.

Steve: He says, "Out of the stories for SpinRite on Security Now!, this was one application of the software I hadn't yet heard of."

Leo: No kidding.

Steve: "Thank you both for an outstanding product and podcast."

Leo: Now, you can, in fact, if it has to be on a specific physical sector, you wouldn't be able to move it. I mean, SpinRite moves things; right?

Steve: SpinRite works with the drive to relocate sectors underneath the file system. So if the drive - if SpinRite couldn't recover the data, it would do the best job it could and then tell the drive, swap this out for a good sector. So one thing I did want to mention to Brad, and a tip for people who might want to use SpinRite like this, or who don't care about the data in the sector - remember he talked about how DynaStat kicked in.

Leo: Right.

Steve: DynaStat is very patient, and some might say very stubborn. It normally reads 2,000 copies of the sector while it's doing its - DynaStat stands for "dynamic statistics," where it's analyzing the data that it is able to read. Even if the drive won't read the sector, SpinRite's able to read what's there. And so it uses that in order to perform its data recovery. Well, in a case like this where you really don't care what's in the sector, you're not trying to recover the data, you're trying to repair the sector without recovering the data, there is a command line option for SpinRite that allows you to dial down or, frankly, up, the strength of DynaStat recovery. It defaults to a hundred, as in a hundred percent. So you can say SpinRite space slash DynaStat space 0 [SpinRite /DynaStat 0], for example, or 1, to bring it down to 1 percent of normal strength, which would be 20 reads rather than 2,000 reads, or to 0, which says, eh, don't bother recovering this data, just replace it.

So in a case like a drive-wiping scenario, where you're unable to wipe because of a bad sector, you could use SpinRite to fix the drive without recovering the data by running it with DynaStat 0 setting, in which case it would just perform the - it would just - it would repair the sector without recovering the sector's data.

Leo: Very interesting.

Steve: So that's cool. Yeah.

Leo: Always nice - somebody's asking in the chatroom, you should do a show on SpinRite and how it works at some point. Might be...

Steve: Well, that's a little self...

Leo: Self-serving?

Steve: Self-serving, yes.

Leo: All right, Steve. I have some questions, if you are in the mood to answer some.

Steve: Sure, absolutely. And also some just good comments from our listeners, some feedback.

Leo: Yeah, by the way, you can always submit feedback to Steve at any time by going to GRC.com/securitynow or GRC.com/feedback, the direct link.

Steve: Yup.

Leo: This is Question 1 from an automotive engineering listener requesting anonymity. We were talking about that OBD port on the car and how it can be used to reprogram a car. In podcast 251 of Security Now!, you read a letter from someone who spoke as if on behalf of an entire industry. I say he does not. I've been in the industry he mentions for 15-plus years on the technical side. I have a Masters in Computer Engineering, 21-plus years of professional experience. He said no one ever considers security. He may speak for after-market devices. He doesn't speak for car company original devices.

On OEM, that is, car company-designed programs, we do study security. Money is spent on independent consultants to analyze security, and vehicle and customer safety are highly appreciated. This is a quick note - I'm at work - but I couldn't let one person's flippant comments destroy an industry. The vehicle hacking that has had press lately was tied to a car with an after-market device connected to the OBD-II, as Leo mentioned. The takeaway from this is be careful what you add to your vehicle. Know what you've installed, just as you're careful on what you install into your house or your PC. You agree?

Steve: Yeah, and I thought that was an important point. This doesn't let me off the hook, I mean, in terms of, like, oh, good, now I'm not going to worry about this, because we know from five years of this podcast that security has been a concern during the five-year life of the podcast. Certainly we've seen it ramp up recently. Yet the problems don't go away. The problems persist because our systems, our computer systems are phenomenally complex. And of course cars, automobiles are getting phenomenally complex. So I'm really glad and heartened to hear that the automotive industry understands the problem, is paying attention to it, has analysts, independent consultants looking at all this. That's all good. That's all necessary. The problem is nothing is sufficient. It just - that's the nature of this stuff. It's too hard to do. So I'm glad for it. But I will predict that we will see problems in the future. It's just - it's inevitable.

Leo: He echoes what I was saying, though, that this hack at least requires physical access to the car. So there are a lot of hacks - when you talk about security, if somebody has physical access, they have a lot more they can do than just over the Internet. And as of yet, this stuff requires physical access.

Steve: Correct.

Leo: So just a point.

Steve: Correct.

Leo: John Hughan with Question 2 in Austin, Texas, wonders why microcode reduces complexity. This is from the last episode where we talked about "RISCy Business." Hey, Steve. Great show, as always. I'm hoping in the upcoming Q&A that you might be able to explain in a bit more detail how having microcode made engineers' jobs easier in terms of the number of AND and OR gates required to implement complex instructions. Why is it not the case that having a "computer within a computer" just

meant that those AND and OR gates had to be implemented in the microcode area in order to run those instructions and manage the "main" area? Or, if microcode allows those types of instructions to be executed in a fundamentally different way that doesn't require those AND and OR gates, why can't the rest of the instruction set be implemented that way? Keep up

the great work.

So he's saying really, when you were saying that one of the things that came up was that they were building into the silicon these fancy instructions, like linked lists, so they came up with microcode as a way to implement it within the silicon, almost in software. But was it software? Or is it - does it require actual AND and OR gates?

Steve: Yes, exactly. And I liked John's question. I thought it was a really good one because he's saying, well, okay, all you've really done is move the complexity from one place to somewhere else. Why is it any less complex? There's two things that microcode does. The first is that, as I described it, the microcode which is used to implement instructions is generally a long word, that is, it's many, many bits wide. And the bits are turned on and off in order to open and close paths through the system in order to implement the instruction. So you route some bits of the instruction word to the adder. And then you route some, like the memory fetch results to the adder, and those get added. And then they go into a buffer.

And so one of the real powers of using a ROM, a Read-Only Memory, is as a lookup table. If you imagine a matrix, a two-dimensional grid, where you have a bunch of inputs on one side, that is, like on the horizontal, and a bunch of outputs on the vertical. And this grid is filled with a collection of ones and zeroes at the intersections such that when you select one of these addresses, some number of bits change on the output. What a ROM does, it allows you to have an arbitrary association between the inputs and the outputs. And if you were to implement that same arbitrary association in discrete logic, you'd just pull your hair out trying to, with standard ANDs and ORs and NANDs and NORs, inverters and all that, trying to wire up what you can do so easily with a simple table.

So the first part of this is that a table lookup, as it's called, can beautifully, with almost no components, just like a little ROM, can allow you to map an arbitrary combination of inputs into a different combination of outputs. So that's a huge simplifying thing, which is one of the things that microcode is, is a table.

And the second part is that, by doing a big job in steps, you don't have to do it all at once. So microcode implies multiple steps to achieve some end. So without this notion of multiple steps, all the instructions you had, no matter how complex they were, were just going to have to happen, bang, just in a single cycle. I mean, it would be like an amazing amount of work somehow almost magically being done, bang, all at once. Instead, if you've got microcycles, then you're able to break up a complex job into many smaller steps, each of which is more simple. So that's the second way that you get simplicity is by sort of factoring all the kinds of things the computer might do into simpler, smaller steps, and then allowing yourself multiple steps to achieve a bigger result. And as a consequence, the savings are dramatic, such that virtually all systems today have used microcode in order to get the job done.

Leo: All right, are you ready for another question, my friend?

Steve: Yeah. Or an observation, in this case.

Leo: In this case, from Simon in Canada, with a security data point from a hospital operating room: Hi, Steve. This week my five-year-old daughter underwent a relatively minor surgical procedure, but still one that required full anesthetic. Oh, that's always scary. Standard operating procedure - literally in this case - dictates that when possible one of the parents attend until the point the child is unconscious, which is why I found myself standing in an OR of a well-known children's hospital, clad in a surgical gown, mask, and paper booties.

After the anesthesiologist had done his stuff, and my little angel was peacefully sleeping, I had time to take in a little more of my surroundings. It was then I noticed - oh, dear - that the rest of the nurses and doctors, who currently had nothing to do, were watching a World Cup game streaming live on one of the operating room computers via a Flash player. Now, I obviously have no idea whether this PC was segmented from the more critical systems in the OR, but I do know that the screen immediately next to it was displaying the medical imaging. I also know that, A, there is at least one computer with a Internet connection in that operating room; and, B, it's got Flash installed - one of the most fertile attack vectors for recent malware. Just an interesting observation I thought you might be interested in sharing with your listeners and viewers. Thank you, Steve and Leo, for your great work. Wow.

Steve: Yeah. You know. Not surprising, unfortunately. I don't know what it will take for the word to get out that this kind of thing is a problem. I mean, we had, remember, UK hospitals that were almost shut down by Conficker getting into their networks, into their operating room computers...

Leo: It's amazing, just amazing.

Steve: ...and causing problems. So you've just got to shake your head. I mean, there's nothing we can do about it. But it's worth just sort of being aware of it.

Leo: Question 4, James Truesdale in St. Louis, Missouri had a RISCy question: Listening to the podcast, heard your explanation of how instruction sets grew due to programmer requests for more complex instructions to make their life easier. I had this thought: Instead of adding instructions, why not just use macros - you mean, do more work? - for commonly used operations?

Steve: Well, you just answered the question, Leo.

Leo: You mean work harder? No.

Steve: Yeah. The idea was that back then, memory was very expensive. And so it really wasn't the program, I mean, the programmers wanted more powerful instructions, rather than using more, less powerful, instructions.

Leo: Right.

Steve: So, for example, in the case of, for example, the VAX that has a linked list instruction, which is like doing all this amazing pointer moving around, programmers were able already to manually manage linked lists, and they could have certainly hidden all of the instructions that they were using underneath a macro. Of course, that would have been dangerous because it's very easy to forget how much work a macro is doing, specifically because it's hiding all the work it's doing from you. It's convenient from a programming standpoint. But the problem is it would be expensive in terms of time, and also the memory that it would take up.

So back then, different from now, where you might say, hey, wait a minute, you know, RISC approaches are much more many small, simple instructions than CISC machines were back then. Back then memory was expensive and slow. So what the programmers were saying was, hey, we're expending all these instructions to manipulate pointers in a way that it'd be really convenient if we just had an instruction that could do it for us. Then we'd save all of this expensive memory and all the time it takes to fetch from this expensive memory. So macro doesn't do the job that implementing complex instructions in microcode does.

Leo: Yeah, I think we kind of touched on that last week, but just it's worth reiterating. It isn't laziness, it's a response really to scant resources, as a lot of this stuff is. And as resources change, you change what you do. It's why we don't need RISC so much anymore.

I love this one, Question 5 from Haystacks Calhoun in New York City. He wonders about Google Search's SSL beta. Is it true that the new secure search - we talked about this on TWiG, and I think we talked about it on Security Now!, how they allow HTTPS when you do a search - is not immediately secure if used at some places? For example, at work, because they have "a web cache doing a man-in-the-middle attack on those searches." Apparently an examination of the certificate shows it's from "the web proxy and not from Google." I'm told this is less secure as it will "show in the web proxy history." Can you confirm or explain the reality of this? Thanks. We did kind of talk about this, too.

Steve: Well, I thought this was worth mentioning, though, because I could see how people could assume that simply using HTTPS to search Google would immediately protect them from anyone knowing what they're searching for. That is, would protect them from someone, for example, otherwise being able to look at their search queries. And so we certainly - we've talked about this issue of SSL interception using a proxy whose certificate has been installed on your browser, much as is sometimes now and increasingly being done in the workplace so that corporations are able to apply the behavior filtering that they want to, to prevent people, for example, from going to social networking sites during the day while they're at work. Wait till you get home to do that. Or in order for the antiviral software to be able to perform its antiviral checking of even content that comes in over SSL. As a consequence, proxies are able now, increasingly, to peek into those connections.

Well, that does mean, as Haystacks has apparently heard people saying, that it is not automatically the case that your Google searches cannot be eavesdropped on by corporate management just because you're using secure searches; that, if the technology

is there, as would be revealed, as he says, by looking at the SSL certificate from a connection to Google, is it Google's certificate or the proxy's certificate that you see? If it's not Google's, then you've actually connected to something other than Google between here and Google, which has done so for the specific purpose of getting into your connection and seeing what's going on.

Leo: Question 6 from Jeff Dunn in Riley Township, Michigan. He's worried about recovering the keys to the kingdom. He says: I have a "what if" question, Steve, on TPM, the Trusted Platform Module, and whole drive encryption. Assuming the keys are stored in the TPM, how do you recover the data - not SpinRite style, which is of course below the file system, but at the file system - if the TPM or the motherboard fails? Is there any way to get the data back?

Steve: That is a great question, something we've never talked about before. So the Trusted Platform Module we have covered in the past is a secure means of storing cryptographic keys which is mounted physically on the motherboard, so it's not easily removed. The question being, what would you do if, for example, something like TrueCrypt was relying on the Trusted Platform Module to obtain its keys, which is a good way for something like TrueCrypt to operate because you would have to authenticate to the Trusted Platform Module before it would release that information. The problem is, what happens if it dies? Motherboards die, random chips die, lightning strikes machines, blows them out. The answer is that, in every case that I've seen, there is a means for backing up the data that's contained in the TPM. And you absolute...

Leo: Ah.

Steve: You absolutely want to do that. So you could argue, wait a minute, if I'm backing up the data in the TPM, then that's not secure. And that's true. Basically you're saying, give me a copy of what's in the Trusted Platform Module so that I can have that offline. And that's the key. Again, it's one of these security balancing things. You need the data in the TPM to be online and to sort of be on the front line, where it's protecting the data from any other activity, viruses, malware, or just misuse, whether deliberate or not. But it also makes sense to have a secure backup copy where you stick it on a thumb drive, for example, and you put it in a safety deposit box. You're responsible, and it's important, that that backup be secured. But it's offline, and you're talking then about physical security rather than protocol data security, like the TPM is. So for sure make a backup copy of the data in the TPM onto a thumb drive, onto a CD, wherever. And then physically secure that somewhere so that, if the worst happened, you would be able to reload this data and still get access to your protected content.

Leo: Very important, yeah. That's good to know. And same thing with the security certificates used by BitLocker on Windows. You can back up that certificate. But if you don't, you're done. So back it up.

Steve: Yes. And frankly, any webmaster who's used SSL has gone through the same thing. My server's private keys are necessarily stored on the server. That's what it uses to negotiate its side of the SSL handshake and connection. Yet it's crucial that I protect those from bad guys because I don't want them getting my server's private key, or that would allow them to spoof SSL connections. And that's true: Any site which is offering

secure connections, there is confidential data, there is very private data in the form of the server's private keys which the webmaster is responsible for safeguarding. And so I've got those written carefully and locked up physically in an offsite location so that I always have them if I need them, but so that they can't get loose inadvertently.

Leo: Michael in Denmark with a question. He wanted a sanity check on soliciting malicious traffic: Steve and Leo, just a quick one here regarding stealth ports. I just changed Internet service providers, got a new router. My new router is kind of locked to the ISP's configuration, only has very limited capabilities - no firewall, but nonetheless basic port-forwarding capabilities. I use port-forwarding for a couple of services. I want the rest of my ports to be stealthed. I found I could achieve this by setting the DMZ forwarding IP to an IP in my range that is not used. Oh, that's interesting. My question is, however, is there any risk connected with this? The router will now allow traffic to flow to my internal net. But there's nobody at that IP address, so there shouldn't be any danger; right? I mean, can some sort of malicious traffic enter my network and do mischief? I don't see how, but I thought I'd better ask you just to be on the safe side. That's a clever hack.

Steve: Well, now, okay. Some listeners who are familiar with port-forwarding are rolling their eyes at this point, saying wait a minute, this is dumb. Everyone knows you can use a DMZ to forward to a nonexistent IP. Okay. It has been done before. The reason I chose this question was that the data doesn't, does not, appear on your network. And it's not even aimed at a nonexistent IP. And I thought that was significant because remember that the way Ethernet works, when your router receives a packet from the outside, bound for any IP, it looks in its ARP table, the Address Resolution Protocol table, to find out which MAC address on the network has been associated with that IP.

So if something comes in to an IP address that doesn't exist, the router will make an ARP broadcast saying, hey, I've got a packet here for IP address 192.168.0.111. Who has that? There'll be no response because no machine on your LAN will have that IP. So the router cannot put that potentially malicious traffic on your LAN. It has nowhere to send it to. So it'll make that broadcast when something bogus comes in to your DMZ port, which you've deliberately set to a nonexistent IP. The router makes the ARP broadcast, says who's got this. Nobody answers, and the router throws it away. So it's a great thing to do.

Leo: Clever.

Steve: Because it not only means that your router won't respond to any of that traffic, but your LAN is completely safe. None of that traffic can enter the LAN because there's nowhere for it to go. The router's trying to send it to someone. Nobody's saying, hey, me, I've got that IP. So the router has no choice but to discard it.

Leo: Clever.

Steve: Very cool.

Leo: We're going to do one more because we're running out of time. But I think this is an appropriate one because Starbucks is adding free WiFi on July 1st. William D. Elliott in Dallas, Texas wants a WiFi Best Practices reminder: Long-time listener, Steve. With this new free WiFi in all Starbucks stores, I mean, that's going to be the largest rollout of free WiFi in the world, I mean, it's thousands of stores. Could you briefly review the basics, or best practices, for those of us who want to use our laptops in Starbucks? We can't bring a router into Starbucks to protect us, so what do you recommend people do?

Steve: Well, actually you can bring a router into Starbucks.

Leo: Do you?

Steve: No, but Mark Thompson does.

Leo: Of course he does.

Steve: There are some travel routers which are little WiFi access points that you're able to plug yourself into. So, but that doesn't solve the problem because - and we need to discuss what the problem is very quickly.

Leo: That gives you firewall, but it doesn't encrypt your traffic.

Steve: Exactly that. Exactly, Leo. So the idea would be that it would give you a firewall, as any router does, that would prevent people on the LAN from having access to your computer. But all computers now have a firewall running by default. All Macs, all Linux machines, all Windows machines have a firewall as part of their operation. And it's a firewall blocking unsolicited incoming traffic. So while I still think it's nice in a home scenario to have a firewall - certainly belts are still useful, even if you've got suspenders - in a wireless setting you'd still have unencrypted traffic between that little router and the location's hotspot. So while it is possible to have a router, it doesn't help you.

So again, the thing to remember is that all of the traffic which you transact with your machine can be seen by anyone. We know there are people increasingly that are sniffing wireless traffic. And unfortunately, as things like this happen, as free WiFi becomes more prevalent, and as there is generally greater value in the data which is going to be sniffed, it tends to encourage this behavior. Also there are wider spread sniffing tools which make it easier to capture this kind of traffic and even parse it for you, so that it says, oh, look, here's a web session, would you like to see the web page? I mean, there are tools out there that will reconstruct from the streams of packets everything that's going on in these locations. And they're unfortunately easy to use and becoming more widespread.

So you just have to remember, in all of these situations, that the fact that it's free and open also means that anyone has access to it. Only SSL protects you. Only secure connections protect you. So it's very often the case that email to, for example, POP and IMAP servers, may not be encrypted. So not only is your logon credentials available, but very often the actual data that you're sending and receiving in email is available. If

you're using web-based mail, make sure that it's secure. And if you're able to do things with websites that accept an HTTPS, it's worth trying to put that in there. In general, I think it's best to just be afraid. I mean...

Leo: That's true.

Steve: It really is.

Leo: Be afraid, be very afraid.

Steve: Just be afraid. That's the best advice I have for you, is use it if you have to, but try not to. And if you are using it, be afraid.

Leo: Or if you use something like GoToMyPC to get to a secure computer, that's essentially - that uses SSL. That would essentially be secure. I suppose, I don't know, I don't have experience with other ones, but something like LogMeIn probably uses SSL. If it doesn't...

Steve: Yes. And it's a very good point. Any VPN solution - we know that I'm working on one, CryptoLink. If you have access to OpenVPN or HotSpotVPN, any kind of a VPN solution is also great protection because it will wrap your computer and all of its traffic in that tunnel and get it out of the danger area before it unwraps it and decrypts it. So that also makes a lot of sense.

Leo: Steve Gibson, as always, a wonderful show. We have two questions we didn't get to, but will you save those for next time?

Steve: I'm going to, yes.

Leo: Good. If you wish to send Steve a question or a comment or a suggestion, GRC.com/feedback. While you're there take a look at Security Now!, of course, the podcast - 16KB versions available of every show, 253 episodes now. Transcripts, as well, thanks to Steve, who foots the bill for that, and we really appreciate it. And, you know, tip him. Buy a copy of SpinRite. It's there also, the world's best - that's a good tip because you get to keep it - the world's best hard drive maintenance and recovery utility. He also has a lot of freebies there at GRC.com.

You can watch us do this show. We do it, if I don't install an operating system update, at 11:00 a.m. Pacific, 2:00 p.m. Eastern Time, that's 1800 UTC, at live.twit.tv every Wednesday afternoon. Please stop by and watch live, or subscribe at TWiT.tv/sn. We have subscription links to audio and video now. Thanks, Steve. We'll see you next week...

Steve: See you next week, Leo. Thanks.

Leo: ...on Security Now!.

Copyright (c) 2006 by Steve Gibson and Leo Laporte. SOME RIGHTS RESERVED

This work is licensed for the good of the Internet Community under the Creative Commons License v2.5. See the following Web page for details: http://creativecommons.org/licenses/by-nc-sa/2.5/