Listener Feedback #187

Description: Before plowing into 10 questions from our listeners, Leo and I discuss Microsoft's Second Tuesday patches, the CA Security Council's reaction to Chrome's CRLSet revocation revelations, an horrific appeal decision in Oracle v. Google, the forthcoming "Halt and Catch Fire" series, and more.

High quality (64 kbps) mp3 audio file URL: http://media.GRC.com/sn/SN-455.mp3
Quarter size (16 kbps) mp3 audio file URL: http://media.GRC.com/sn/sn-455-lq.mp3

SHOW TEASE: It's time for Security Now!. Steve Gibson's here. It's the Second Tuesday of May. We'll talk about Microsoft Updates. We'll talk about SpinRite. We'll talk about, well, your questions and Steve's answers, certificate revocation, and more. By the way, Steve's going to declare victory a little later on. It's all coming up next on TWiT.

Leo Laporte: This is Security Now! with Steve Gibson, Episode 455, recorded May 13th, 2014: Your questions, Steve's answers, #187.

It's time for Security Now!. Get ready to protect yourself with the Explainer in Chief himself, Mr. Steven "Tiberius" Gibson. Steve's at GRC.com, where he's the creator of SpinRite, the world's best hard drive maintenance utility, and a security guru, kind of perforce. I think it all started for Steve when his own site was DDoSed. And then I think you found spyware on a system; right? Hello, Steve Gibson.

Steve Gibson: Well, it actually - hello, Leo. Great to be with you again, and not for the first time.

Leo: Another one of my discursive introductions.

Steve: Yeah, it was actually when I put my company first with - I think it was our first persistent connection to the Internet. Because we had this weird cc:Mail where we would dial up and get email on a timed basis. And which was really all you needed back in the early days. I mean, the GRC.com domain was registered just after Microsoft.com was registered. So we were playing on the 'Net, but only with email. But anyway, it was a DSL connection. And so now we were persistently on the 'Net. And I was curious about the Internet, and I knew enough about it to poke around a little bit. And in some of my

pokings, I poked a neighbor, in terms of IP address, and there was his C: drive. And I thought...

Leo: Ah, and the NetBIOS hack was invented.

Steve: ...this cannot be good. So then I poked a few more neighbors, and everybody's C: drive was exposed. So after I made sure ours weren't, I thought, okay. This is a problem. And so I just - I wrote ShieldsUP! in order to, I thought - I realized that, when someone comes to my website, I know who they are and what their IP is. So it's possible for me to give them a benign scan back, essentially, a backscatter scan, and check to see whether they had this problem.

And of course then it was Kate who famously found this when you were doing The Screen Savers. And when you guys used it at the studio, it knew the name of your computer and your administrator name and so forth. It was, like, it was a little unnerving to people. I would greet them, "Hi, Joe." And they're like, oh, my god.

Leo: What? What?

Steve: So, and it's all been an interesting journey from there.

Leo: Awesome. Well, I'm so glad we could get together today and don't have to talk about SSL, Certificate Revocation, Heartbleed...

Steve: Actually, there will be a little of that.

Leo: Oh, crap.

Steve: Yeah, well, you know, what I've learned is these big things just don't disappear overnight.

Leo: No, they don't, do they.

Steve: So we have relatively few things to talk about prior to our 10 questions and answers. But I do want to - there are a couple things that happened that we need to spend some time on. One is that it's the Second Tuesday in May today. Also the Certificate Authority Security Council, CASC, has weighed in into the Chrome revocation issue with an interesting statement that I want to share. After which, actually, I'm not sure what the timing was, but around the same time the Chrome developers tripled down. So that's when you go one further than doubling down. They've tripled down. Then there was a really bad decision made by an appellate court, overturning Google's previous victory over Oracle on the Java API issue.

Leo: Yes. Oh, we'd love to talk about that one, yeah.

Steve: Got to talk about that. We have then some other little miscellaneous fun stuff, and 10 questions from our listeners.

Leo: Oh, yay.

Steve: So another great podcast.

Leo: We get a Q&A in here. That'll be fun.

Steve: Yup.

Leo: Well, before we get to the news of the day, I'll tell you what, let me talk a little bit about my friends at Citrix and ShareFile, and you can take a little cup of coffee and - wait, that's a good-looking mug you've got there. Where did you get that thing?

Steve: Are these for sale, Leo?

Leo: I think they are, but I don't know. No, that's a good question. Are our mugs for sale, or are they just our mugs? We don't know. Nice slurp. They are not for sale in any store. You have to be a TWiT host to get one.

Steve: Ah.

Leo: Enjoy that muggery.

Steve: And I'll see if I can get a head strap for it.

Leo: Steve Gibson, Explainer in Chief, what is the news of the day? It's Patch Tuesday.

Steve: It is Patch Tuesday. And everything so far is tracking exactly as predicted. Which is there were a total of eight bulletins. Two were critical; six were important. Of the two critical ones, one is for Internet Explorer, all versions, 6 through 11. And I think it's only on Server 2003 that they're still patching 6, if you have it, because that's sort of the equivalent of XP, which they're no longer patching. And but 6 gets patched there. And when I pulled the notes together, we had the, whatever they call it, their advance notice, which was dated a week ago on the 6th. And after this all went to bed, then I did see that they had updated. So I don't know the details of the IE problem. I don't know that

it's a critical remote code execution exploit, like all the critical ones are. And the other one was on their SharePoint Server and Web Apps Server.

One critical for IE, which is not surprising. IE is Internet-facing. And as we always said, use IE with extreme caution. Maybe don't use it as your mainstream browser. Microsoft makes you use it for things like Windows Update and so forth, so you have no choice there. But that's also not unsafe, to go to Microsoft and do Windows Update. So as long as it's not your main browser, you're okay so far. So nothing really horrible this month. We'll wait and see which month something horrible happens to Windows XP.

As I mentioned at the top of the show, after last podcast, there was - in fact, it was May 8th it was published. There is an organization of certificate authorities. They call themselves the Certificate Authority Security Council. And so this is an industry group of all of the certificate authorities, GlobalSign, VeriSign, DigiCert, the works. GoDaddy, I'm sure, they're all there. Well, they published an official response to the controversy, essentially. Their response starts out: "The recent Heartbleed issue has reawakened interest in SSL certificate revocation." Then they say, parens, see Adam Langley's blog, which is a link to it; Larry Seltzer's articles here and here, two links; and Steve Gibson's web pages.

Leo: Oh. Those are the polar opposites, I guess, and Larry's kind of in the middle. And so that's - that makes sense.

Steve: Exactly. So here they are, saying, from their perspective, they said: "Several years ago, the CA Browser Forum convened a special Revocation Working Group to explore issues and solutions. Leading CAs were actively involved in that group, and many of them invested in moving their OCSP responders to high-performance, high-availability Content Delivery Networks (CDNs) to respond to browser vendors' requests for increased performance and reliability." And as an aside, when I was digging in, I discovered that GlobalSign, for example, and DigiCert, the two that have been on my radar, were all using CDNs. So this notion of their response being too slow, that's apocryphal and really no longer applies today.

Continuing from their announcement, they said, "Google was part of the Revocation Working Group and announced CRLSets to that group and the wider CA Browser Forum. CAs were disappointed that Chrome wouldn't actively retrieve OCSP responses from them, but we were under the impression that CRLSets would include most revoked certificates. Adam did ask CAs to help CRLSets by telling Google about important revocations, and CAs largely complied, for example, when the CA had to revoke intermediate certificates. But CAs have no reliable way of knowing which end-entity certificate revocations are important" - I mean, it's like there's more than two million of them, which ones would you choose - "since certificate owners don't reliably tell CAs whether or not the revocation is important. Many CAs allow the customer to choose from a list of 'revocation reasons.' But just as companies are hesitant to reveal that they've suffered a security breach, it's assumed that they are hesitant to tell the CA that their private key had been compromised." And then they say, in parens, "(this would constitute an important revocation).

"As a result, end users and browsers have no way to determine whether a certificate was revoked because of the server's loss of control over the key, fraudulent activity by the server administrator, the presence of malware onsite, or simply out of an abundance of caution. Heartbleed is a perfect example of why revocation is important, even without identified key compromise. No one can say for certain that their server's private key was

compromised. Most of the revocations that have occurred are going on CRLs for 'business reasons,' as Adam defines it, and not picked up by CRLSets.

"It's now clear," writes this CA group, "that CRLSets are simply a blacklist of high-profile revoked certificates. Other browsers have similar blacklists, and these can be effective at times, for example, to indicate revocation of an intermediate certificate that may be several years old and does not contain an OCSP pointer. But they're not a substitute for OCSP checking of end-entity certificates.

"Google moved away from supporting OCSP without adequately informing Chrome users of this fact. Although IE and Safari also soft-fail if an OCSP response is not received, those browsers still use OCSP by default." And of course Firefox does, as well, as does Opera, all of them but Google, or Chrome. "The engineers creating those browsers apparently have not concluded that OCSP is broken. Even if revocation checking by OCSP isn't 100% accurate, it can still protect a high percentage of users who navigate to a site with a revoked certificate and receive an OCSP response indicating revocation. Turning off revocation checking for everyone means no one is protected.

"All browsers compete on speed and performance, and OCSP checking can slow page-loading. We think many browser users would trade off a small performance hit for increased confidence in the authenticity of the website." And they finish, saying: "Revocation is a very complex issue, with lots of room for debate. Reasonable people can disagree on the effectiveness of using OCSP. The CASC" - that's this Certificate Authority Security Council - "agrees that OCSP Stapling, and putting OCSP Must-Staple extensions in certificates, is one of the best solutions to address many issues with revocation at this time. But until that happens, we oppose browsers removing non-stapled OCSP checks." So...

Leo: So that's victory.

Steve: ...100% agreement with my position, yes. That is.

Leo: Hmm. Has Adam Langley responded to that?

Steve: No. He's so annoyed by all of this. However, there is a response, of a sort. They have disabled it and removed the checkbox to allow users to turn it on.

Leo: Oh.

Steve: Really. I'm not kidding you. That's what I said when I referred to them "tripling down." It is called "confusing." So it's Issue 361820, and it's check for - now, you'll probably still see it.

Leo: I still have it, yeah.

Steve: My Chrome version is 34. But if you look in the show notes, Leo, there's a link there to code.google.com. It's their Google bug tracking. And 361...

Leo: So it's future versions of Chrome, then.

Steve: Well, it's happening. It's v37. So what happened was...

Leo: I'm at 34, as well.

Steve: Right. So what happened was - because they've got them in the pipeline, and there are, like, nightly releases and so forth.

Leo: You see it says "canary." That's their gamma channel. So there's Chrome, there's Chrome Beta, and there's Chrome Canary, which is the beta beta channel. So that means they're going to slowly migrate it up.

Steve: So what they said was...

Leo: The confusing certificate revocation checkbox.

Steve: Yes. They called it "confusing."

Leo: Doesn't confuse me.

Steve: Well, and understand also that, first of all, it's under Settings, where almost no one goes. Then it's under Advanced, where gurus go, or like people who want advanced settings. And then it's down there, and it's pretty self-evident to me. And so 27 hours ago, when I looked this morning, now it's probably more like 28 or 29, what happened was even this thread was controversial. People were coming in, saying, no, don't remove it. What's confusing about it? And so they shut down the thread. They locked the thread because they just didn't want to discuss it anymore. And then one of the developers posted into the locked thread. The final entry says "Tested the same on Win8 Chrome version 37.0.1987.1." And then that's an official build number. They said, "Canary - fix works as expected. The confusing certificate revocation checkbox is removed under..."

Leo: Works as expected. How hard would that be to remove?

Steve: I know.

Leo: I'm glad they tested it.

Steve: Whoa, that was a tricky patch, baby. Glad they made sure that it - yeah. And if not, you just use some whiteout, you know, on your screen. "Removed under Manage Certificates." And then he actually has a screenshot of the button showing no checkbox

below it.

Leo: I guess if your position is that the certificate checking for revocation doesn't really accomplish anything, as it is their position, then you'd just say we don't want to put something that implies you're being protected when it really doesn't do anything. I guess that's not - that is confusing in the sense that, well...

Steve: This, the reason this whole thing is controversial is you can see both sides. I mean, Adam is never wrong on fact.

Leo: Right.

Steve: He's just set himself a position where he's making perfect be the enemy of good. And...

Leo: And a lot of geeks do that. That's very common.

Steve: Well, and that was my problem with raw sockets. Everyone was like, oh, UNIX has had raw sockets, and Linux has it, and Gibson doesn't know what he's talking about. It's like, folks, there is a gray area that matters.

Leo: Making something better, even if you can't make it perfect, is still better.

Steve: Is worthwhile.

Leo: Yeah.

Steve: Yes. Yeah. Especially when the cost is so minimal. It's not like people are clicking on links and then going, oh, my god, why did I turn that checkbox? I'm so confused. Leo, I'm confused.

Leo: Yeah. Okay.

Steve: So anyway, now I think we can put this to rest.

Leo: I think it's very interesting that the CA authorities themselves agree with you. I mean, that's...

Steve: Yeah, yeah. Well, I mean, as I said, if I had lost control of my cert, I would absolutely want to protect the industry and anyone...

Leo: It's kind of in the - that's what the point is of a cert.

Steve: Yeah.

Leo: Revocation is kind of built into the entire concept.

Steve: If you don't, you can't have trust, if you can't have non-trust.

Leo: Right.

Steve: By definition. And it's one thing to sort of stay, to sort of come at it from a thousand miles and go, oh, well, what's the chance? But I'll tell you, it gets very personal when it's your own certificate. And if I ask myself the question, if I needed to revoke a certificate, do I wish the browsers would honor it? It's like, oh, my - of course. I mean, I absolutely want them to honor that. But Google's decided otherwise. We're going to win, ultimately, just as happened with raw sockets. Microsoft removed them. After they got attacked by them by the MSBlast worm, they thought, oh, this is what Gibson was talking about. So infrastructure will evolve without Google's help. And that was my issue, was we could do this better with Google than without. But it's going to be without. We'll do it anyway. Then they'll switch around because then they're really going to be behind, which is too bad. But so be it.

Okay. Now, in 2010, so four years ago, Oracle sues Google over 37 specific Java APIs which were used in Android.

Leo: Not the code.

Steve: Right.

Leo: The name of the function. The name of the variables. Not the code, the name.

Steve: Right. And we waited two years for - and we remember talking about this, I mean, because this is like, wait a minute, you can't, no, you can't protect an API. I mean, Linux copied the UNIX API. And multiple people do languages. And so the assumption has always been that the language itself is, especially when it was expressly made public, I mean, no one would have used Java if Sun hadn't said, of course everyone can use it. This is, you know, we've got one. Other people have them, everybody. And we just want to make sure they're compatible. That would really be their only issue. So we waited two years. And one of the most impressive judgments in the history of technology came down from a judge who was so determined to rule correctly, he learned Java.

Leo: Isn't that awesome? And wrote a range check. He said, "I did it over the

weekend."

Steve: Yes. And when you read his ruling, he's using all of the terms correctly: "instantiate" and "instance" and "prototype." I mean, he's using these terms that are programmer terms, and he's a judge.

Leo: Judge William Alsup. And pat on the back to Judge Alsup.

Steve: Yes. So at the time, two years ago, Wired summed it up nicely. They said: "Oracle said the Java APIs were like a beautiful painting. Google said they were more like a file cabinet. And in the end, Judge William Alsup came closest to agreeing with Google, comparing an API to a library that organizes the Java programming language.

"In the much-anticipated 2012 ruling," which we waited for for two years, "in the epic legal battle between Google and Oracle, Alsup wrote: 'Each package is like a bookshelf in the library. Each class is like a book on the shelf. Each method is like a how-to-do-it chapter in a book. Go to the right shelf, select the right book, and open it to the chapter that covers the work you need.' His ultimate point was that the organization of a library is not subject to copyright. Yes, he said, the books [themselves] are copyrightable" - that would be the actual implementation of the code - "but not the way the books are organized.

"In other words, Google did not infringe on Oracle's copyright." And, by the way, this was both a patent and a copyright suit, and the patent got completely thrown out. There was just, like, no chance that this stuff was - you couldn't patent the APIs. So they said, okay, well, then they're copyrighted. So it wasn't subject to patent. So, "In other words, Google did not infringe on Oracle's copyright when it cloned 37" - and understand, cloning is what you have to do. It's not like you can - the API is the language. It's the function calls that you use in order to invoke aspects of the language. So you can't change the arguments around. They have to be the same. And, for example there, are Windows, there are people who made Windows work-alike OSes, and even Wine running on Linux.

Leo: Well, let's go back - we wouldn't have a PC industry if Compaq hadn't been able to reverse-engineer the BIOS.

Steve: And, by the way, that is "Halt and Catch Fire."

Leo: It is going to be Compaq. Okay, good.

Steve: Yes, that's why...

Leo: It must have been. I figured it.

Steve: That's why it's in Texas. And I've been watching the trailers for it.

Leo: Me, too. I've been trying to figure it out, if it's Compaq or if it's eEye or what.

Steve: Yes. And at one point, and you can tell, I mean, the trailers are cut very quickly. They roll very fast. But at one point you see someone, they're talking about, like, stripping it down. You see them sliding the cover off of an original XT or PC. And you also see them lifting that classic IBM logoed blue and sort of tan striped box out of the trunk. And so these guys are going to reverse-engineer the IBM PC. And so, I mean, this couldn't be more apropos to this discussion because, remember, I mean, this was controversial. And what they did was they did a cleanroom implementation of the BIOS. They had people who never, who expressly never had contact with an IBM PC, only the specification of the BIOS calls, that is, the BIOS API. And it stood up in court. So here we are again with Google and Oracle.

And so they said: "In other words, Google did not infringe on Oracle's copyright when it cloned 37 Java APIs in building its Android mobile operating system. Though Google copied the organization of the APIs, it built the code behind them on its own or at least mostly on its own."

"The Java and Android libraries are organized in the same basic way." This is the judge again in his ruling. "The Java and Android libraries are organized in the same basic way." Again, his analogy to a public library. "But all of the chapters in Android have been written with implementations different from Java, but solving the same problems and providing the same functions."

And then Google wrote: "This reaffirms our longstanding understanding of the law, that these APIs were free for anyone to use as we did, taking just the declarations and doing our own independent implementations. That's the way developers use Java. You can't say a language is free for everyone to use and then hold back the nouns and the verbs."

So now we move forward to last week, 2014, and the bad news that shook the industry, those of us that were watching. I tweeted it and got a lot of responses because I just, you know, this is one of your head-shakers, is Oracle appealed that decision. And it's worth mentioning that Alsup wrote this so carefully, and so clearly, specifically to withstand appeal because he knew there was a lot at stake. Everybody has rights to appeal, and he wanted his ruling and his investment in, like, taking the time to understand this, to become a Java programmer in order to judge this, he wanted it to withstand appeal. So it didn't.

In updating their article, again in Wired, Wired wrote: "Oracle won a big legal victory over Google on Friday" - that was last Friday - "when a federal appeals [court] overturned a ruling in their epic battle over the Java programming language. Larry Ellison and company are calling it a win for the entire software industry" - oh, my god - "but others see it differently." Count me among them. "They believe it could harm the industry in enormous ways. Some even think it could come back to bite Oracle. The dispute comes down to arcane code used in Google's Android operating system; and, if the courts ultimately find in favor of Oracle, the decision could reverberate across the tech industry. The situation is complicated," writes Wired, "but it can be summed up pretty simply. Oracle owns Java. Google cloned Java in building Android. Oracle sued. And now the courts are trying to decide when it's okay to clone someone else's software." Now, we have to be careful...

Leo: It's not cloning. And that's the thing that bugs me.

Steve: Yes, I was going to say we have to be careful with this summary because this is not accurate. They didn't - they cloned the interface.

Leo: They cloned the way you call, the name by which you call routines. That's all.

Steve: Right, like the definition of the language, not the plumbing, not the implementation. Yup.

Leo: Well, now, it will be appealed again, of course, to the Supreme Court, in all likelihood.

Steve: Yeah.

Leo: The good news is, while I have no high opinion of the Supreme Court's technical knowledge, they have been in recent decisions very pro intellectual property. Actually, I don't know if that's going to be a good thing at all. Come to think of it. We'll see.

Steve: Well, with any luck they will look at Alsup's decision and his logic, and they will say this holds, that the appellate court made a mistake.

Leo: Yeah, a huge mistake, a huge mistake.

Steve: Yes. Oh, and Leo, do you know what would happen? It would be the end of life as we know it. I mean, I can't even, I mean, wow. I mean...

Leo: Well, I mean, Microsoft could say to Wine, no, can't do that. But that's not the end of the world. I guess AT&T Bell Labs that own the UNIX trademarks, copyrights, could say to Linus, no more Linux. That would be pretty disastrous.

Steve: Yes, yes. I mean, throughout. Anybody who is using a standard interface could have the originator say - in fact, it would end up changing us to the point where anything that, moving forward, any interface that someone wanted to promote would have to be expressly and explicitly released into the public domain before anybody else would consider coding to it because otherwise you're locked into their terms. I mean...

Leo: Maybe that's not a bad thing. That might not be a bad thing at all.

Steve: Remember that it was Microsoft's copying Turbo, Philippe at Borland, and driving

- no, no, I'm sorry, it was Philippe copying Microsoft. Microsoft was selling languages for hundreds, four, five, $600, when Philippe comes out with Turbo Pascal for 49 bucks and just, I mean, absolutely, if you could have had a blood pressure cuff on Bill Gates at that point, it would have shot off the scale.

Leo: Well, and there's a good example. I mean, Pascal or any programming language, if you write a compiler, you're performing the same function, but you're not with the same code. You're just emulating the API.

Steve: It's the language, yes.

Leo: It's the language. Oh, boy. You know, it's just such a stupid decision. It's very discouraging.

Steve: Well, I have to say I have not looked at the reasoning for the appeal except that they must have made a mistake. They must have looked at this and said, "Oh, look. Copyright applies." And so clearly Oracle's attorneys learned from Alsup's carefully written judgment how to strengthen their position that copyright applies. And they were copyrighting argument, like function names and argument definitions, and saying this is ours. And again, the annoying thing is, if anyone believed this would ever happen, back when what's his name at Sun invented Java...

Leo: Oh, James Gosling. I wonder what he thinks of all this.

Steve: Was it Gosling? Or was it - I'm trying to think. There was the other guy, I thought, that did Java, when it was going to be a set-top box interpretive language.

Leo: Yeah. What was the name of it? It was Oak.

Steve: Yeah. And I can't remember, you know, the little wunderkind who was there at Sun.

Leo: I thought it was Gosling. But I don't know who else.

Steve: I'm not pinging on his name.

Leo: Gosling wrote Oak.

Steve: Scott somebody?

Leo: Oh, you're thinking of Scott McNealy. You mean the guy who founded Sun.

Steve: Maybe I was thinking of Scott.

Leo: Yeah.

Steve: Gosling did Java?

Leo: Gosling wrote Java, yeah.

Steve: Okay, yeah. Well, again, yeah. So my point is that, if anyone believed that their proprietary argument...

Leo: Bill Joy, are you thinking of? No.

Steve: Bill Joy. That's who I was thinking of.

Leo: Bill Joy. I knew you'd be thinking of him.

Steve: Yeah.

Leo: Love Bill.

Steve: Yeah. Anyway, so if anyone imagined that Java would ever be held to this standard, nobody would have adopted it. I mean, obviously, Google wouldn't have. I mean, Oracle's annoyed that Google has done such a big thing with Java, whereas they never managed to. So this is your typical...

Leo: I think this is why Oracle bought Sun, bought Java, was to sue the pants off people. That's what I'm thinking.

Steve: Wow.

Leo: Who knows? I mean, this is just - damn you, Larry Ellison.

Steve: I know.

Leo: Geez, Louise.

Steve: Yeah, bad. So we talked about "Halt and Catch Fire," which I'm sure now, based on the trailers, it's going to be a fun, fun story of the reverse-engineering of the IBM PC.

I think that's even going to be, I mean, it's a different story, obviously, than the skunkworks project down in Boca to create the PC.

Leo: Right. Which would have been, frankly, maybe a more interesting story. I don't know. I mean...

Steve: Yeah, yeah.

Leo: I thought that was quite an interesting story. And of course the guy who did it died in a plane crash.

Steve: Yes. Fabulous story. And of course they could have brought in all of the fun stories that we have in our industry about the visit to DRI and the visit to Microsoft and so forth, yeah.

Leo: Gates would have been involved, yeah, yeah.

Steve: I did want to mention, they're not an advertiser on this podcast, but they are lurking around TWiT somewhere. I got sent this package, it's called Harry's Shaving Products.

Leo: Yes, we sent you a Harry's Kit, yeah.

Steve: Yes, H-a-r-r-y-'-s. And in anticipation, well, because I had it in the first place, but also because I thought they were going to be an advertiser, I shaved with them. And, wow, Leo, I'm...

Leo: You like it, huh?

Steve: I really - I don't know if it's the cream or the blades, but something gave me the best shave I think I've ever had. And I normally use the Edge gel and the Gillette whatever it is, oh, the Fusion is the one I use. And I hate shaving. And I don't think this will make me love it any more, but I'll certainly hate it less. Anyway, I've already ordered refills of these things because this, I mean, I switched. I'm now using this.

Leo: Wow, that's awesome. Yeah, they're not - they advertise on some of our shows. We'll get them on your show. I'm sure that's why we sent you a kit.

Steve: For our listeners, Harrys.com. And apparently it's less expensive than the ridiculously overpriced razorblades. I'm not a big fan of the silver, I got the fancy pack with the silver...

Leo: The Truman, you got the Truman Kit, which we talked about.

Steve: ...executive handle. I like the rubber, the plastic one that looks like it's more square. It's on its way. I ordered one immediately because it's like, okay, let's just - we'll improve on this a little bit.

Leo: You did miss a spot over your lip there a little bit, but that's - no, just kidding.

Steve: Oh, and also it came in underneath, because I like to kind of trim the upper edge of my upper lip underneath the - oh, anyway, I'm completely happy. So, I mean, no endorsement for any reason except I wanted to let our listeners know, if you're a blade shaver, not like a buzzing machine shaving person, check it out because you may feel the same way I do. I'm just - I was really pleased.

Leo: I think there's an offer code TWIT for four free blades, I think it is, with purchase. So if you buy - the next time you buy a kit, use the offer code TWIT, and I think you get four free blades.

Steve: And they ship pretty quickly.

Leo: Oh, yeah, they're great.

Steve: I got a notice, like, maybe two days after I ordered. And they didn't know who I was, so it wasn't any special deal. It was just it's on its way to me, so I'm pleased.

Leo: Well, I'll make sure they know this, and that they should buy ads. This was not a paid ad. This was...

Steve: No.

Leo: ...an unsolicited user testimonial.

Steve: I just wanted to say, it's like, hey, I discovered something. We're rolling forward again with SQRL. On Sunday afternoon the gang in the newsgroup got a hold of the growing version that I have; and, surprisingly, it worked everywhere. That is, on everybody's versions of Windows. And we're tracking down some two functions that were not written in Wine in order so it doesn't sort of silently log in the background. It's logging "Fix me, fix me, fix me" because I'm calling some things that aren't implemented, but they're not necessary, actually. And this is all a preamble for next week's podcast because it is the entropy harvester, which is now written and running and producing high-quality true random numbers, not pseudorandom numbers.

Leo: Love that name.

Steve: And in a way which - oh, what, "harvester"? Yeah.

Leo: Entropy harvester, yeah.

Steve: Entropy harvester. And I got also a nice note from a Nick Bowen in Walnut Creek. We've talked about, recently, obviously, about how SpinRite recovers people's data. Then we were talking recently about speeds up people's drives. And we've touched a couple times on how it can even help you destroy your data. And when I entertained the idea of deliberately building that into SpinRite 6.1 or some version of SpinRite, all the guys over in the SpinRite newsgroup said no, no, no, no, no, don't put in a secure wipe in SpinRite. That would be crazy. Keep that separate.

And, I mean, the reason I was interested is that all of the work that I was doing and will soon be doing again on 6.1, as soon as I'm done with SQRL and get back to 6.1, all of it was completely applicable to writing the absolute best secure wiping utility that the industry has seen in, I mean in terms of thoroughness and speed. I'll be really positioned to do that once I have all of this low-level, large-buffer technology running in SpinRite. Until then - oh, and so I am going to do a separate utility. I've already got the name, and a trademark on it, in fact, and domains and so forth. It's called Beyond Recall, which will be GRC's absolute secure wiper for spinning and solid-state media. But that's tomorrow. Today, what generally people use is DBAN, which is Darik's Boot and Nuke, DBAN, and it's spelled D-a-r-i-k.

Leo: If I ever meet Darik, I'm going to shake his hand. DBAN is awesome. I love it.

Steve: Yes. And so in this case, however, Nick wrote, he said: "A friend recently brought me his computer for me to run DBAN on, prior to him getting rid of it." Which is of course why it's - because it's freaky, in fact. If you buy, like, drives from Fry's, they're often not sealed. And I've purchased some supposedly new drives and found other people's software installed on them. So...

Leo: More a comment on Fry's than anything else, but go ahead.

Steve: Yeah, exactly. Anyway, so he says: "But it was now an older machine, and DBAN would not run because the hard drive would just grind. I had purchased SpinRite a couple of years ago and hadn't used it yet." So, Nick, thank you for purchasing it for supporting us. "But I thought this would be a great opportunity. So I ran a quick Level 2 repair scan, and it fixed the issue. This allowed me to securely wipe the drive before it was given away. Thanks for the podcast and great product. Nick."

So here the moral is today you can run SpinRite to fix the drive enough that then you can nuke the drive using DBAN to absolutely scuttle any data that it's got. And of course all current owners of SpinRite will be able to upgrade to 6.1 for no charge, as soon as that's ready. And then I think probably once the SpinRite series is finished, 6.1 will probably not have native USB because I don't want to delay it for USB because it's going

to do so much for the majority of users, for Mac and high performance on PCs directly operating at the hardware level and not using the BIOS, that I want to get that out. But I don't want to stop. Then I'm going to look into USB and adding that support natively. That finishes 6. I think then that I will write Beyond Recall and create a new product, the first new product we've had in 25 years. And then...

Leo: What? Oh, that's good.

Steve: ...go on to SpinRite 7.

Leo: You are a busy boy.

Steve: So that's currently the architecture. But first SQRL, which is where I'm working now and making great progress. And I think things are going to happen very quickly now. So I'll keep everyone up to date. But it won't be long.

Leo: Steve's roadmap, product roadmap you've just heard right there. Steve Gibson, Leo Laporte. Are you ready, my friend?

Steve: You betcha.

Leo: Questions, answers, the people want to know. We'll start with Question 1 from Paul Byford, Tamworth, the United Kingdom. He wanted to follow up and elaborate on the future of revocation and DNS. I guess you're right. We're not done with it yet.

Steve: Eh, I think this one will be - by the end of this podcast.

Leo: We'll have covered everything you never want to know.

Steve: But Paul brought up a really good point that I hadn't focused on enough, so I thought I would share his thought.

Leo: He writes: Great show. Listener since Episode 1, et cetera, et cetera. In last week's podcast you touched on DNSSEC and DANE. Although I know you've talked about DNSSEC in the past, I think it's worth looking at it in terms of the chain of trust and revocation debate. As I understand it, each link in the certificate chain for DNSSEC is tied to the domains within a URL from the top-level domain on down, and so there should be a clear path for checking the chain of trust. Also all the certificates are pushed out though the DNS system and so should be cached on servers close to the user, normally at the ISP, and refreshed often, typically once a day.

So it seems to me that this solves both the speed of access issue and the revocation

problem, as well as providing a quick way to validate CA-issued certificates from the current chain of trust through the TLSA records and DANE. I know that doesn't completely solve the nightmare that is the X.509 certificate validation, but it's a good step forward and does provide a second chain of trust for confirming that users are connecting to the server they think they are. What do you think?

Steve: Yeah. First of all, Paul lays out the architecture, I think, cleanly. And he's absolutely right. As I did mention last week, once we get DNSSEC - and it's a slow process because there's so much inertia. We need servers to get updated. We need clients to get updated. We need to make sure that they're, like, that the Internet infrastructure itself allows us to work. And it's one of those situations where all DNS servers right now understand DNS. But DNSSEC requires an extension to that understanding, and it's broken unless everybody agrees to understand it. So it just - it's tough to get. But oh, my goodness, is it going to be powerful because it will give us an Internet-scale addressable directory that is secure.

So as I did mention yesterday, this DANE is a means for using domain name association of data, where, for example GRC could publish the hash of our certificate; and, thanks to DNSSEC, it is unspoofable. None of Dan Kaminsky's concerns with spoofing or any DNS hijacking or rewriting or anything because you end up with, essentially, in the same way that we have a signed certificate, which is the reason we trust the certificate, we have a signed DNS record. And this is where this chain comes in because, in fact, people have probably heard about like the root servers are now signed, meaning that their records are cryptographically signed. So we just need to extend this out to the end machine.

And so, for example, as I was saying, GRC could publish through DNS the hash of our certificate. And if browsers used DNS to look up the hash, they would have an absolutely secure means of knowing that, as recent as the cache is, GRC is asserting that this is our certificate, not the certificate authority, but we, because GRC controls our own DNS records. So what's cool about this is, as Paul says, it creates a different chain. It also creates one where the entity whose credentials are being relied on has control over the trust, which is really neat because it means, if something happened, we could change our certificate and immediately change our DNS record and get it resigned so that it's verifiably from us. And then, as that DNS update propagated through the Internet, that new certificate would then be understood to be current.

So we just need DNSSEC. I can see so many different, valid, really useful things. I mean, and we know, for example, that once we have it, the antispam technologies which are relying on DNS will also get more leverage. We're just going to have, it's just incredibly useful to have a secured directory for the Internet that scales at Internet size. And DNS has already proven itself able to do that. The hierarchicalness and the caching and the very low, the lightweight use of UDP packets, just it's made DNS a real success. Now we need to lock it down and secure it. And when we have that, wow. I think we're going to end up finding lots of uses for it.

Leo: Where do we stand right now? I mean, I know...

Steve: It's a good question because I was looking into that in this context. And I know that the late-model client operating systems do support DNS at some level. It's something I ought to really focus on a little bit more because...

Leo: Yeah, I'm curious. And I know that OpenDNS has decided not to support DNSSEC.

Steve: Right. They're using their own protocol, and we've talked about it. They're using DNS Curve as their solution, where they install a proprietary client in the users of the OpenDNS system, and then that establishes a secure link to them to prevent spoofing and forgery and interception and so forth.

Leo: Question 2 comes from Chris Fowlkes. This is a big fat softball right over the plate for Mr. Gibson. Steve. It's actually a tweet. He's @MyGhostWorld. Do you think it's a good thing that there are so many CAs, certificate authorities? Is too many a bad thing? In Firefox there's a huge list.

Steve: Well, there's a little more to say, I think, than just punting the softball. One of the things that was mentioned in the last couple weeks, which is why this sort of caught my attention, is that it's worth noting that the vast majority of actively used certificates come from a tiny minority of certificate authorities. So that, if you had GoDaddy and VeriSign and DigiCert and GlobalSign and just a few more, you end up with covering the large bulk of the sites you want to go to.

Yes, you could go to an obscure site somewhere that would say, wait a minute, and your browser would say I don't trust this certificate, it's signed by somebody I've never heard of. But some people that are really security-conscious have experimented with dramatically winnowing down the number of certs in their local root store of certificate authorities whose signatures on received certificates from web servers they trust, and they do really well with only a handful of them. And of course the advantage then is, if you went to a major site that you'd not been having trouble with, and suddenly you had trouble, that would be a big red flag. That would be, wait a minute, why do I have a certificate from a site that was fine yesterday, but it's not now? Probably because that site is being spoofed by somebody.

And the other thing, Leo, you may have seen this in the news, and it's not in my notes here, but I wanted to remind myself not to forget mentioning it. Many people tweeted it. And it was, unfortunately, it was one of these stories that came out, and everyone picked up on it, with a bad headline, talking about I think it was the non-insignificant number of fraudulent certificates that are in use on the Internet. Did you see that in the last week?

Leo: No, I didn't.

Steve: Well, it turns out it was a bogus story.

Leo: I'm glad I didn't see it, then.

Steve: What they were talking about, for those who did, was certificates being minted by border appliances that we have often talked about because I am so opposed to them. So that, for example, antivirus is able to decrypt your encrypted communications in order to scan it and then reencrypt it. So these are the high-end corporate firewalls and corporate

AV appliances which are, I mean, even some software does this, some end-user software, installs its certificate on your machine so you trust certificates it signs. And then, when you think you're communicating to a site, you're actually communicating to it. It's signing the site certificate that it synthesizes and giving it to your browser to make it transparent.

And so that's what the story referred to, is their instrumentation on the client side was picking up a high percentage of fraudulent certificates. Well, yes. But they were locally fraudulent. They're not out on the Internet being fraudulent because nobody would trust them. They're trusted because your computer has been, some would say, compromised by adding to its root store a certificate for the AV company or whoever it is who did this appliance. So that's what that all was.

So in answer to Chris, and asking do I think it's - is it a problem that there are so many certificate authorities, I talked about this years ago when I happened to look and saw that there were, like, 400 of them that Windows was trusting. And I remember on this podcast saying, oh, my lord. And that's where, of course, unfortunately, our much-made-fun-of Hong Kong Post Office came from because they were among those hundreds. And so the danger is that any one of those can sign a certificate for any domain. There's an any-to-any mapping right now. And that's one of the things that DNS can help solve because in DNS we could publish what authority is our chosen authority for our domain. And so the browser could check and say, oh, look, Gibson likes DigiCert. I'm not going to trust a certificate from anybody else, even if it's otherwise trustworthy, because that's not what GRC has in their DNS record. So we don't have that yet, although that record is defined.

The problem is everybody wants to be a certificate authority. I mean, you're printing money. You are. You set up a system which verifies people's identity to varying degrees, depending upon what kind of certificate you want. Do you just want a DV, a Domain Validation certificate? Do you want an Organization Valid certificate? Or do you want an EV, an Extended Validation certificate? So once you establish that, and you create a web presence and a web system that allows people to submit their certificates for signing and send them back, you then charge them a lot of money, I mean, a lot of money for these things, and they all expire every two or three years. So if you have kept them as a loyal customer, they're going to come back and pay you a lot of money again. For nothing. For bits. So who wouldn't want to be in that business? And then who's to say that a legitimate new company shouldn't be in that business?

And then of course they say, hey, we're in the business now. So they get their certificates signed by somebody who's already trusted. That bootstraps them in until they get their certificate pushed out into all the browsers and they can make a valid claim to the browser, saying, hey, we're a legitimate company. Check us out. We're good guys. We should be able to do certs, too. And the browsers say, yeah, okay, you meet all of our policy requirements. And so the world gets one more CA. And that's happened. So it's like, who wouldn't want to do it, and that's why there's so many of them. And, yes, it is, from a pure security standpoint, the more we have, the more opportunity for problems because, unless you really know who they are, all of them, your browser's trusting them all. Yikes.

Leo: Yikes.

Steve: But it's understandable how it happened.

Leo: Yeah. From Cairns, "Can" as they say it in Australia, Abhi Beckert with a little more on the EFT question. Now, he's in Australia, so banking rules are different: Disabling electronic access does not protect you from fraud. I know someone who lost $10,000 when somebody took money from his account by simply faking his signature. He never got the money back because he was never able to prove to the bank that it was a fraudulent transaction. The bank maintained all their security policies were followed and that it was him, not some criminal, who withdrew $10,000 out of his account. Still, I agree it's a good idea to disable electronic access. My retirement fund account only accepts electronic deposits. You may not withdraw or transfer out of the account electronically. Now, it's different in Australia than it is in the U.S. The rules vary.

Steve: Yes. I think the lesson, though, I liked when he summed up, saying, for example, in his retirement account you can only do deposits. The takeaway from this whole discussion is I would like our security-conscious listeners to think of this like a firewall. The original firewall was all packets could come in unless we denied them. And we quickly learned that was a bad idea. So we flipped the rules around so it's a deny by default, and then selectively permit. Similarly, in the way people - because I would just hate to have people lose money for no reason. And so in the same way, think about the way your accounts are interconnected and what privileges they have, and remove the ones you don't need. By default, they probably all allow everything because unfortunately that's, you know, the banking industry hasn't caught up with the lessons we've learned with firewalls.

And so I know in my case I've had to selectively disable features in my various accounts where I looked at them, I said, oh, wait, I never want this to happen. I don't need this. I don't need this. I don't, I mean, if I could turn off those ridiculous checks that they keep sending me from my credit card company, I would do that in a heartbeat, but that doesn't seem to be an option. So think of it that way. Think in terms of features that are there that you could live without and whether there's any way you could imagine they could be abused because, if you can turn them off, then do. So I appreciate this little reminder.

Leo: Rick Andrews, Mountain View, California writes: Steve, listening to your podcast 453 on Certificate Revocation, I heard you say the website owner sends the private key to the CA. I know you know it's the public key, but it is in the transcript. That's what you said. Can you correct the transcript? I guess the record.

Steve: Yeah. I've actually caught myself making that mistake a couple times, and it's just because I'm running at a thousand miles an hour, and I say the wrong thing. So I wanted to, if anyone was confused by that, I'm not going to go back and fix the transcript because it's still in the audio, and it's still in the video. So it's more important to let everyone know, who probably knows anyway because I have said it correctly way more than I have said it incorrectly, and that is, what's so cool about this is the server owner mints a pair, a public key and a private key. The private key never leaves.

And that's what's so neat about this is in no way does this public key infrastructure ever require that they disclose their private key. And SQRL, the SQRL protocol works the same way. It never leaves. Then in the case of the PKI system, the public key is what goes off and gets signed. And that's the public key which the server is going to be sending off to everybody who connects with it. So, like, sending it to the certificate authority for signing

loses nothing. I mean, basically they're adding their signature. And then that's what's being sent down to every browser that connects.

So the browser gets the public key, verifies the signature, which causes them to trust the identity assertion that's being made. And then they use the public key to send things to the server that only the server with the matching private key can decrypt, and vice versa. So it's a slick, simple system, and I will try to be careful when I say "public" and "private" in the future. But, yes, the private key, you're right, Rick, I know the difference, never leaves, never should. Only the public one is floating around in the public.

Leo: Frank in Mnchen points out why certificate handling should be in the browser, not the OS: In the last Security Now! [SN-454] you made a point that the certificate handling should be done in the OS instead of the browser. Well, I disagree, especially in light of the recent XP expiry. He says, "XPiry"? If Firefox used Windows to handle the certificates, we wouldn't get updates anymore. It'd eventually not be safe to run Firefox in XP anymore. That's a good point.

Steve: That is a good point. And now, for example, we have no more updates to XP. There are known malicious certificates which are - some of them are root certificates that have gone bad, and some of them are intermediate certificates. And there is, in Windows Certificate Manager, an untrusted certificates category, and it's non-empty. There are certs in there. And if any certificate came that was signed by one of those intermediates or chained back to that bad root, Windows would know. But Frank's absolutely right that XP no longer gets the benefit of that.

Now, Firefox does, and Chrome does. And, well, actually Chrome doesn't on XP because Chrome's using XP's security system. So Chrome on XP would not be getting the updates. Firefox would. But Frank has got a good point. If new evil certificates occurred - and in honesty those rarely happen. They can happen. But we've talked about when they have. It hasn't happened for, like, four or five years. But, yes, I certainly do take Frank's point. It's a good one.

Leo: Chris in Colorado wonders about self-signed certificates: As a web programmer I routinely create self-signed certificates on web servers so that during development I can connect to secured resources, http and ftp. So if I create a self-signed cert at the server and then install it on my computer, am I putting myself at risk to man-in-the-middle or some other attack? Thanks.

Steve: That's a great question. I do the same thing myself. I actually have a cert that is www.steve. And there's definitely no top-level domain named Steve. There's no risk at all. Actually, there is some reason to argue that it's even more secure than using a chain because there's less moving parts and less to go wrong. Remember that the root certificates that certificate authorities have are self-signed. The way the chain gets anchored is with a certificate that the certificate authority signed itself. So it's a self-signed certificate that then signs an intermediate that signs the end cert.

So Chris and anybody else, self-signed certs are a tremendous solution. You install one in order to connect and then put the public key on your end so that you trust it, in exactly the same way as the PKI model works, but arguably even a little more simply because you're not trusting a chain. You're just verifying that the certificate at the other end is correct. And, yeah, it's a great solution. And no, no man-in-the-middle attack risk or

anything else.

Leo: Greg, writing from Unseen.is - wow, Unseen.is - wonders about outbound firewall filtering. Do you set up your software firewalls - do you - hello. Do you set your software firewalls to block outbound traffic? We all agree that inbound traffic should be blocked by default and allowed for only specific reasons. But what about outbound traffic? Do you recommend "block all unless whitelisted," or "allowing all unless blocked"?

Steve: Well, you know, it's an interesting question because what's different about outbound is we know where it came from. And this was what created that whole early software firewall industry. Leo, you and I spent countless hours on The Screen Savers talking about ZoneAlarm and similar firewalls where, thanks to the fact that outbound traffic is created in the system, the system knows which application originated it. So that's sort of gone out of favor these days. It's sort of become automatic so that, when you run applications, the applications are able to interact with the software firewall API when they need to in order to open ports automagically, so that returning traffic can come in.

For example, someone runs Skype on a Windows machine, Skype no longer has any trouble at all doing what it wants because there's the Universal Plug and Play out on the router that's probably on unless the user has turned it off, and then there's Windows' own firewall, which says, oh, yeah, Skype, and Skype works with it in order to create whatever holes are necessary through the firewall. For advanced users, there are still controls. You can go into Windows firewall, for example, and mess around with what applications are doing what. But by default, the way things have evolved is incoming traffic that is not expected is just blocked. It's dropped at the firewall boundary. Yet outbound traffic generated by applications is just generally allowed out.

So we've sort of gone back to controlling the applications that run rather than also separately controlling what they're able to do. If you had a firewall that you were willing to manage like we did back in the ZoneAlarm days, where you'd install something, and then it would start giving you pop-ups and requiring permission, and you'd have to deal with it, it's like, well, you certainly can do that. But we've just sort of switched, sort of fallen back to just kind of keeping an eye on what applications are running in our machine.

Leo: All right. I wonder what Unseen.is is. What is ".is"?

Steve: Where is ".is"? I don't know.

Leo: It's not Ireland. That's ".ir"; right? Mathew Taylor - wait a minute. Wasn't there a story about ".is"? Well, anyway. Or was it ".si"? Iceland? Maybe. I don't know. Israel? Israel, that's what it is. It's Israel.

Steve: Ah, Israel.

Leo: Mathew Taylor, Brisbane, Australia. Another great Aussie listener, or reader, or writer, wonders how he could have an open DNS resolver when ShieldsUP! says it's closed: My ISP told me my network's sending out way too many DNS requests, and I have a problem. They talked about ensuring port 53 was not open to the Internet. The network in question only has one computer on it, Windows 7, with no fancy DNS server technology running. I rang them. They said I could test my public IP with an open DNS resolver test, which I did, and it came back saying I was running one. What? But ShieldsUP! reports that port 53 is closed. How can both these tests be true? Eventually we figured out that the router, a Billion 7700N R2, had been compromised. Oh. Oh, we're seeing more and more of this. And we replaced the firmware to fix it. What's happening there?

Steve: So what's happening is DNS uses UDP. And ShieldsUP! tests TCP.

Leo: Ah.

Steve: So what ShieldsUP! has always done is to look for the port status of TCP because TCP connection setup always returns a packet if it's open for business. So we send a SYN, a so-called "synchronize" packet, and we get back a SYN/ACK if that port says, yeah, I'm open, come connect. And at that point my test drops the connection, or maybe I send a reset back. I don't remember now. But I don't proceed to do a "full open," as it's called. We only do a half-open and then drop.

But one of the beauties of DNS is that it is even - it is lighter weight than TCP, so that you send, rather than establishing a connection through the so-called three-packet handshake, DNS just sends a packet out with a question, and it gets a packet back with the answer. And if it doesn't get the answer back, it could have been lost, so it sends it again. And it does it a few times. And then, if it doesn't get any answers, then it asks all the DNS servers that it has registered to see if anybody will get an answer. And the first one that answers gets promoted then to the top of the list of servers that it asks next time.

So it's a nice, worked-out system. But that's why 53 wasn't shown as open, is that his router wasn't responding to TCP port 53, even though that is valid. You can establish a TCP connection to a DNS server and do the same kind of stuff over TCP. And in fact, something called "zone transfers," for example, when you're moving an entire DNS zone to a different machine, when a DNS server wants to get a whole zone, it'll do that only over TCP. It's not allowed over UDP. UDP is just for making your typical DNS queries. But in the case of his compromised router, it set up a server, a DNS server, only answering actual DNS queries. And that meant UDP and not TCP. So ShieldsUP! didn't show it, even though there was a UDP server there.

And I've thought about enhancing the service over the years. It would be an undertaking. So maybe when I'm closer to the undertaker I will do that because every UDP port is going to have a different protocol. I would need to actually create a UDP, I mean, a DNS server to meaningfully test people's port 53, and I'd have to create servers, different UDP service servers, or clients actually, to check the servers at the other side. And that's way different than just checking for open ports. So it's on my to-do list, but it's not even on the roadmap at this point. And that explains why.

Leo: .IS is Iceland.

Steve: Iceland.

Leo: Israel, as I should have known, is .il. Benjamin, Austin, Texas wonders, as we often do, about pseudorandom mixing: In all the talk of using random numbers there's something I'm unclear about. You talked before about how you can have devices that produce true random values, albeit at a speed that precludes relying entirely upon them for all your random number needs. My question is you've discussed mixing functions before. Once you combine pseudorandom values with a truly random seed, isn't the larger result random, as well? Unless I'm missing something, mixing in a true random number generator to the pseudorandom output would nullify any sort of guessing you could do on the factors that aren't truly random, would it not?

Steve: Okay. Well, there are a couple ideas kind of jumbled together here. Probably the best example, the best classic example that we've talked about is the so-called "one-time pad" in crypto, where you take true randomly arrived-at values, for example, from dice rolls, and you write them down, then you mix your plaintext with that. And because of the mixing, which can be as simple as an XOR, where the random values simply randomly flip bits in your plaintext, even though that doesn't seem like super-amazing crypto, the fact that you randomly flip bits is the best cipher that exists, strange as that is, because since there's no pattern to the bits you flipped, there's no way for someone to try to crack it using anything.

Like a statistical analysis where, if you'd had a simple substitution cipher, the so-called "Caesar cipher," where you're just substituting one character or symbol in the language for another, a frequency analysis will immediately reveal the frequency characteristics of the original language and allow you to start determining what the substitution was. But if you flip bits randomly, that destroys any frequency bias. Again, it's like it doesn't seem like it should, like that would be strong enough, but that's all it takes is randomly flipping bits. And only the identical random reflipping or unflipping puts it back, puts the message back to what you had. So that's where you're talking about mixing in randomness.

Now, the other thing you mention is pseudorandom values and a seed. I was recently studying the Intel chip random number generator because it's one of many sources of entropy which SQRL harvests. And I have this concept of harvesting broadly from a number of different streams of entropy with all having different characteristics in terms of amount of entropy, speed, attackability, knowability. And that's the topic for next week's podcast, which I think people are going to really enjoy.

But what Intel does is they use a true random number generator to seed a pseudorandom number generator. And so what's pseudo about a pseudorandom number generator is that the numbers look random, even though they're algorithmically produced. And if you knew what the - in fact, in the Intel case, they use an AES counter DRBG, Deterministic Random Bit Generator. And that means they have the AES cipher, which has a counter fed into its data inputs. And we don't know what the counter is, like where the counter is in its cycle. And then there's also a key. The AES cipher key determines what the mapping will be between the counter's value and the output. And that's really all you need. You just spin the counter, and out comes values. The advantage of that is it can go at incredibly high speeds.

And so, as Benjamin says, it may be the case that the quantum source for generating true random numbers just doesn't produce noise at the speed that we want to consume randomness. So architectures like the Intel chip, they do a compromise. They use the low-bitrate true random generator to generate the seed for the algorithmic pseudorandom number generator. Now, what they also do is they are constantly reseeding because the concern is, if you had a large enough sample of pseudorandom numbers, maybe you could work backwards and determine what the count and the key were. And if that happened, you could determine what the past and the future numbers were because the danger with one of these algorithms is that its internal state could ever get known, in which case, because it's just an algorithm, you could predict the future and the past.

And but the beauty is, since this is on the silicon, and very much like Apple's Secure Enclave, there just isn't any way in there from the outside, from the pins, from the firmware, from the software. You just can't get there. So they've hidden it, and that makes it safe enough, in the same way sort of like that the Trusted Platform Module that never really got off the ground was also a piece of hardware that the software had limited access to. So these are some of the concepts we'll be fleshing out in greater detail next week.

Leo: I thought, now that almost all [Windows sound] - excuse me. I just rebooted. Had that burrito for lunch. I thought that most all new hardware had TPM in it.

Steve: There may be TPM, but we're just not seeing the kind of use of it that we would expect.

Leo: Really? I thought the fingerprint readers and a lot of these devices were relying - for some reason I thought TPM had actually kind of sneakily become a success. But I could be wrong on that. We want to, we still want to do, on Know How, a random number generator involving, what was it, a diode? What was it that you suggested?

Steve: Yeah. Turns out it's very simple, just a diode biasing a base emitter junction on a transistor.

Leo: I love that.

Steve: It just generates noise. It just [white noise simulation], you know, it just comes out. And then you count it or filter it or do whatever you want to do to it.

Leo: Our final question is a tweet from Scott Martin, @ScottMartin: What happens if someone creates a new SQRL ID on a client that doesn't have good entropy? Couldn't you end up with collisions, then?

Steve: Well, a great question. And we will be - this is our lead-in to next week's podcast because I have arranged for no clients ever not to have sufficient entropy.

Leo: Entropy harvesting.

Steve: At least none that I have anything to deal with. One of the things that some of the guys in the newsgroup were saying, you know, Steve, deal with this random stuff later. Other people who are trying to write clients are waiting for the final spec to get blessed by you writing the code to implement it.

Leo: Yeah, but this is much more fun. This is more fun.

Steve: Well, what's kind of funny about me is I just - I can't jump ahead somehow. I mean, I have to have this done in order to move forward. And the nice thing about this is I published all the source code and the algorithm and everything in enough detail that anybody else who wanted to implement the solution I came up with has everything they need to do so, even in a different language, because I've completely explained how it works. I'll give some links to that stuff next week.

But the danger is that entropy becomes an afterthought. That is, and this is what has happened in the past. People have gotten the algorithms all worked up. And then it's like, oh, wait. We need random stuff. Oh, let's just call the RAND function or something. I mean, it just doesn't get the attention it needs. And nothing is more important than us having sufficient entropy. So I wanted to just know that I'd solved that problem so that I could say, "No platform that my code ever runs on will ever have insufficient entropy." And I'll prove that next week.

Leo: By the way, I checked out - TPM is on almost everything now, including Chromebooks.

Steve: Oh, it is.

Leo: It is. Acers, ASUS, Dell, HP, Lenovo, Fujitsu, Panasonic...

Steve: So even non-laptops. Because laptops had them early and for a long time, but I wasn't aware that it was on desktop models, too.

Leo: Yeah, well, and I think the cheap stuff doesn't probably. But because it's built into ethernet chipsets from Broadcom and stuff, it's all over the place. It's supported on all the operating systems except Linux. I imagine Linux could support it, too, if it really wanted to.

Steve: Meaning that there's an OS API, apparently, that allows you then to talk to it.

Leo: Yeah. Windows 7, 8, Vista all support it. Windows 2008, Server 2008. It works with BitLocker. So if you're using BitLocker with your Windows machine, the built-in

OS encryption will use TPM.

Steve: The keys, that's good, good.

Leo: It's funny because I, like you, remember this whole controversy over TPM. And kind of one just figures, aw, they probably gave up because it was an Intel spec. But no. Not only did they not give up, they won. They just did it sneakily.

Steve: Good. Good.

Leo: Without telling anyone. Steve Gibson is at GRC.com. That's where you'll find 16Kb audio of this show, as well as full transcriptions written by a human being, the lovely and talented Elaine Farris, out in her ranch in the middle of nowhere.

Steve: I don't even want to know how hot it is out there right now.

Leo: Oh, can you imagine? We also have full-quality audio and high-def video available at our site, TWiT.tv/sn for Security Now!, and wherever finer netcasts are aggregated and distributed, including iTunes, Stitcher, and of course all you need really is an app. Almost all the mobile platforms now have TWiT apps from our many third-party developers. And we tip our hats to you guys and encourage you all to download and install it. Makes it easy to listen and watch whenever you want. Steve is also the man behind SpinRite, the world's finest hard drive maintenance utility. You'll find that at GRC.com, and lots of freebies. Next week we go entropy harvesting.

Steve: Yep.

Leo: That should be fun. I like that.

Steve: It's going to be really interesting. Another little bit of a propeller-head episode, but I think it's going to give everyone lots to think about.

Leo: I don't mind that. And what was I going to say? Oh, questions for the following week can be posted on Steve's site. He doesn't do email. Just go to GRC.com/feedback and leave your questions there. Or tweet him. He is @SGgrc on the Twitter.

Steve: I am.

Leo: More and more of the questions come via Twitter. That's cool. And he talks to

people and converses. Doesn't follow anyone. He follows you if you @ him, @SGgrc. Thank you, Mr. G.

Steve: Thanks, Leo.

Leo: See you next time on Security Now!.