Listener Feedback #193

Description: Leo and I discuss the week's major security events and discuss questions and comments from listeners of previous episodes. We 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-466.mp3
Quarter size (16 kbps) mp3 audio file URL: http://media.GRC.com/sn/sn-466-lq.mp3

SHOW TEASE: It's time for Security Now!. Steve Gibson is here. We'll talk a little more about iOS security, and then he's going to answer your questions. A lot of conversation about cloud storage and that kind of thing. Stay tuned. A really interesting Security Now! is up next.

Leo Laporte: This is Security Now! with Steve Gibson, Episode 466, recorded July 29th, 2014: Your questions, Steve's answers, #193.

It's time for Security Now!, the show that protects you and your loved ones online. And here he is, the security chief, the Explainer Extraordinaire, Mr. Steven Gibson. It's good to see you, Steve.

Steve Gibson: Great to be with you again, as always. Our last podcast of July, coming into August.

Leo: We referred a little bit to you on MacBreak Weekly again this week because of our conversation about Jonathan Zdziarski's OS 8 or OS 7 security issues. And I think everybody has concluded that you are exactly right in the degree to which you worry about those security issues.

Steve: Yeah, exactly. In fact, I have a little follow-up this week, both on that and canvas fingerprinting. A big errata fix, which is my comment that I'm sure you saw this, too, you must have received tweets and things, that iOS 7 had not been jailbroken.

Leo: Yeah, I should have stopped you on that one.

Steve: Well, and Jonathan did write that; but, as I commented, his paper was about 10 months old. And in fact that was his comment, was that it wasn't until he created the PowerPoint presentation and did an actual presentation that anyone noticed what he had written back in October of 2013. So anyway, so that was old news. And so to everyone who tweeted me, thank you for the correction. And that's corrected.

Leo: Although as Rene Ritchie was pointing out, the jailbreak is the bigger security flaw. I mean, if you're going to really talk about security issues, the fact that you can jailbreak it is a huge security issue; right?

Steve: Yeah, although that, as I understand it, that's still, I mean, it requires the user themselves to jump through all kinds of hoops in order to make that happen. It's not like...

Leo: It's nontrivial, yeah. Although there have been single-button jailbreaks.

Steve: I always think that, I mean, and this is like me, saying it ought to just be something you can do.

Leo: Yeah.

Steve: You know? It ought to not be this cat-and-mouse game between the consumer who owns the device and Apple who is fighting them over control. If the consumer knowingly says, "I want freedom," they ought to be able to just press a button and get it.

Leo: Well, of course that's how it is on Android. There's a checkbox that says, is it okay to buy stuff from third-party stores? You check that box. They warn you there's a risk inherent, and then you just do it.

Steve: Yeah, and there's something you press, like, seven times, too, isn't there?

Leo: Yeah, but you don't need to do that to jailbreak an Android device. It just is a setting. Jailbreaking on the iPhone only really does one thing. It allows you to buy from somebody besides Apple, buy apps or download apps.

Steve: Well, and it also, well, or to install apps other than from the curated, controlled environment.

Leo: That's it, yeah. And so that's a checkbox in Android. There is other stuff, yeah.

You can turn on developer mode by tapping something seven times. And then many Android phones, but not all, can be rooted. Which is really more for people like us who understand computers, that's getting...

Steve: Who know what the word "root" means.

Leo: Yeah, superuser permission, you know.

Steve: And if you don't know what that is, you should not do it.

Leo: You shouldn't do it.

Steve: No.

Leo: But that's, to me, that's the clear and distinct difference between iOS and Android. And if you want that, you should use Android. Right there. Boom.

Steve: Yeah.

Leo: Seems risky, though.

Steve: I heard you mentioning also that Samsung, despite their massively heavy advertising, is not the Android device you should buy. You were liking the HTC?

Leo: I like - I currently use an HTC One, but there are many good choices, including, if it were easier to get, I'd probably recommend this very inexpensive but very nice OnePlus One phone. It's only 300-some bucks. And it's a really nice, state-of-the-art phone.

Steve: And there was just a, was it a law passed? Or, no, I think it was - didn't Obama sign an executive order?

Leo: He hasn't signed it yet. It is a law that both the House and Senate have passed. And the reason - this is that says that you have the right, a legal right to unlock your phone after you've had it for the contract period. It's come and gone. It was, I think it was the Librarian of Congress that said you could do it, and then...

Steve: Yeah, and where did that - how did they get their feet in there? I thought, Library of Congress, what?

Leo: They're responsible for copyright and trademark and IP protection. I know, isn't that weird? It's just weird. The whole world is weird. The U.S. world is weird. Everybody in Europe's going, wait a minute, you don't have - wait a minute. What are you saying? In Europe, it's not only legal, it's required that the carrier unlock the phone.

Steve: Well, and I love the idea of the freedom that that would create. After you've fulfilled your contract, and you're not happy with...

Leo: Well, of course. You bought this hardware. You should be able to run it any way you want.

Steve: And it's not cheap. I mean, it's seriously expensive stuff. I mean, Apple's rolling in cash.

Leo: So AT&T's done this for a while and allowed customers to do this for a while. And now it's required that everybody offer it. Well, soon as President Obama signs it.

Steve: Will be, yeah. So just to follow up on the iOS security thing, after, or maybe it was the afternoon or the morning, anyway, it was just after last week's podcast. And I was saying last week, Apple needs to respond. And as of last podcast, they had not. And of course they, as I was saying it, they were posting their response. Which was just to sort of assert their position, which was that the three main issues that Jonathan had were there for legitimate purposes. Now, he then of course defended his position, as we would expect him to do, just because it's a position, and it's his, saying "I'm still not happy." And it's like, well, okay. We're going to probably not agree to disagree, we're just going to disagree. But so Apple did what I expected.

And Ars Technica continues to report this back-and-forth. And so Dan Goodin, who covers these things, covered the story that Apple responded and Jonathan wasn't happy. And down in the comments there was one that I liked. Someone posting as TheShark wrote: "I'm trying to get upset over this latest 'revelation,' but somehow I just can't. Take the pcapd capability, for example. Why should I be worried that a computer which I've already configured to sync my phone with and which is on the same WiFi network can activate pcapd on my phone? That computer is almost certainly in a position to run pcapd locally and capture the WiFi traffic if it wanted. There's no reason to think that pcapd on the phone is going to see traffic that the computer can't.

"It's the same thing with most of the other data which is accessible. A computer which I've chosen to sync with can actually access my contacts, my photos and other data which I want to sync? This is a concern why? I can imagine some app developers getting worried that authentication tokens which they don't sync and don't want users to be able to directly access might now be available, but it's also easy to imagine how useful it would be in debugging your app to get access to those files as well. Sorry, Jonathan, but I'll be more impressed when you find an actual backdoor. This seems far more like a useful tool than a nefarious one to me." And, I mean, that restates it, I think, pretty well.

So, I mean, but this is the way the security business works. And as I said, I'm not unhappy that Jonathan did this. Apple needs to know that their work is being scrutinized

and that we're all just not sheep following them and accepting everything that they say. For example, they're still arguing that iMessage is secure. And we absolutely know that it's not; that because they are the curator of the certificates, and there's no visibility at all into the certificates that we're receiving from them, which we're using to sign our messages to its recipients, nothing prevents them from slipping one of their own in. And we sign that, and they're able to tap our iMessage. So, but again, this kind of analysis is what we need.

I found, thanks to a listener, a site which demonstrates canvas fingerprinting, Leo, and you should go there: browserleaks.com/canvas. So it's www.browserleaks.com/canvas. And it does nothing without scripting on. But now scroll down, and you will see that it has found your fingerprint. See that green checkbox back up a little ways, right kind of there in the middle, right where that - yeah, there. So what this did was it just fingerprinted your browser using canvas.

Leo: Is this any different from that we talked about, I don't know, two years ago, this kind of...

Steve: No, actually we did talk about this two years ago. It suddenly bubbled back up with these inflammatory headlines.

Leo: Because of Gizmodo. They made a big deal about it in Boy Genius Report.

Steve: The unstoppable tracking technology.

Leo: But we've known about - we've talked about this for years.

Steve: So but the reason I wanted to come back to it again was to correct the record. If you look there, it says 1,847 unique signatures, not 64. So last week in the research report from the guys that found this and developed the technology, in their analysis they found it was less than six bits' worth of identification from that. But they had a relatively small sample size. Browserleaks.com has been there looking at all visitors for a long time. It just - it's looked at me. Now it's looked at you. And it's looked at everybody else who's gone there. And 1,847 is the number. And that's about 10.85 effective binary bits, or a little less than 11 bits.

So that's certainly - which is to say that any of this technology running on anyone's browser that has scripting enabled, and I forgot to highlight that last week, this is all client-side, and it's done by someone injecting some JavaScript onto the page that your browser dutifully renders, and then it sucks that off, makes a hash, and sends it back to the tracking mothership. But I just wanted to say that it turns out it's not - you're not put into one of 64 or something bins. It's 1,847. So that's substantially better. But on the other hand, that's certainly not identifying you on the Internet.

And so this still is far from being unique. It's one more thing that can be used. But it does require scripting. Unlike cookies, for example, that's part of the underlying plumbing of web browsing, this is script-based hack, and so users have a little more control over it. Oh, and I also found out that a lot of people are already blocking that site, the one, I can't remember, it's "all" something, that I mentioned last week, that's

like the king of the injecting of canvas [addthis.com]. Everybody knows about it and has been blocking it for a while. So, yes, this has been around. And as you said, Leo, Gizmodo got headlines and upset everybody.

Leo: What a surprise.

Steve: Yeah. It was funny, too, I saw - it didn't make it into the Q&A.

Leo: Bait from Gawker, what a shock.

Steve: It didn't make it into the Q&A. But somebody, along the same lines, was commenting how his sister was going, I think to China for a few weeks, sort of to be a missionary, and left her laptop home at some inconvenience to her because of that horrible reporting that was done during the Olympics in Russia. Remember where, and we talked about this on the show, the claim was that within minutes of crossing the border, hackers had taken over all of your electronic devices.

And as we know, it was just - it was a horrible story, meaning that it was contrived, and in fact in order to have your Android phone taken over - they may have even jailbroken the phone and installed or turned off things or installed malware or something. I don't remember the details. But the point was it so scared people that it changed their behavior, unnecessarily frightening them from the conveniences that they would otherwise enjoy. On the other hand, she probably did take her smartphone, even if she left her laptop behind, and arguably that's as vulnerable as a laptop, if not more so.

Great news from Open WhisperSystems. WhisperSystems we've talked about for years. This was the company that Moxie Marlinspike founded, which we also reported was acquired by Twitter toward the end of 2011. And shortly after that their first product, which was Android-only, that RedPhone, essentially the RedPhone service was disconnected after Twitter's acquisition. But then it was released as free open source and became available again. What's then happened is that the so-called Open WhisperSystems project, which sort of has continued to live as a free and open source project, has continued to develop this technology, all free, all open source.

And I think it was this morning, I think this is very fresh news, they just announced the release of Signal, which is their free encrypted voice system for the iPhone. So what they said in their blog was: "Secure calls are just the beginning. Signal will be a unified private voice and text communication platform for iPhone, Android, and the browser. Later this summer, Signal for iPhone" - which is now available and free download and also open source and beautifully designed, and based on well-proven robust security protocols - "Signal for iPhone will be expanded to support text communication compatible with TextSecure for Android. Shortly after, both TextSecure and RedPhone for Android will be combined into a unified Signal app on Android, as well. Simultaneously, browser extension development is already under way."

And I forgot to mention that Signal on the iPhone is compatible right now with RedPhone on Android. So we now have the two premier phone platforms supported by essentially a single, cross-platform, truly secure voice communication system. And they'll be adding text to Signal, which will be also compatible with TextSecure. And then they'll essentially be merging TextSecure and RedPhone on the Android platform under the Signal name.

So in a few months there should just be Signal for both iPhone and the Android platform. And the price is right. It's free. So for anybody who has - first of all, RedPhone has been Android-only until now. It's now available for the iPhone in the form of Signal. And later they'll be sort of formally amalgamating them. So that's good news for anybody who wants absolute security. I spent enough time with it to look at it and see that they really did things right. So I'm really pleased that they now have the iPhone platform, as well.

And there was just the announcement of sort of a troubling vulnerability, although it was responsibly disclosed, which means that Google knows about it and has already patched the problem and scanned the Play store to make sure that nobody's taking advantage of it. This was a presentation that will be made at next week's Black Hat conference. And the press has picked it up, and so it's in the headlines today because that's another thing that just happened. And it's being called the "Fake ID" vulnerability. That's the name that was given by Bluebox Security. And Jeff Forristal is the chief technology officer of Bluebox, who will be giving the presentation at next week's Black Hat conference.

And again, Dan Goodin in Ars Technica reported this immediately with a headline that said: "Android Fake ID Vulnerability Lets Malware Impersonate Trusted Applications, Puts All Android Users Since January 2010 at Risk." And then there's just the first couple lines, or the first one line of his report. He wrote: "The majority of devices running Google's Android operating system are susceptible to hacks that allow malicious apps to bypass a key security sandbox so they can steal user credentials, read email, and access payment histories and other sensitive data, researchers have warned."

Okay. So here's what happened. Apparently something broke, and this was with the v2.1 of Android, which was released in January of 2010. And what happened was somewhere along the way certificate chain verification was broken in Android. Now, this is different than revocation, which never existed in Android, still doesn't. But the idea with the chain, and we've talked about this often, I mean, the whole point of a security certificate chain is that you have a trusted root, and it signs another certificate, which may sign another certificate and so forth until you get to sort of the client certificate. And the point is that that certificate, it asserts its signer, and we hope that that assertion is verified. It turns out for the last four and a half years Android has not been checking the signatures on the certificates, and nobody noticed.

So the reason this is important is that there are privileged applications in Android which are trusted to bypass the application sandbox. For example, Adobe's Flash is allowed to act as a plugin for any other application installed on Android devices, presumably to allow it to provide animation and graphics services to them. We know how that works. Or, for example, Google's Wallet has privilege to access the NFC hardware, which normal apps can't because you need to be trusted in order to do that.

So Flash has a certificate, which is signed, which allows Android to trust Flash. And, by the way, that certificate is unique to Flash. And the fact that Flash is carrying it with that certificate, is carrying that signed certificate, is recognized by Android and gives Flash extra privileges that other apps don't have. Similarly, there is a certificate specifically for Google Wallet, which allows it to have access to the NFC hardware.

Well, it turns out no one is checking to see whether those certs are actually validly signed. So anyone can spoof those. Any malware can simply carry those certificates. And, for example, if it carried a Flash certificate, even though the signature was invalid because it couldn't get that certificate signed by an actual authority, if Android doesn't check the signature, then it doesn't matter. So it turns out that this has been true for four and a half years.

So in talking to the press, Jeff Forristal, the CTO of Bluetooth - of Bluebox Security, sorry, said all it takes is for an end-user to choose to install a fake app, "and it's pretty much game over. The Trojan horse payload will immediately escape the sandbox and start doing what evil things it feels like, for instance, stealing personal data." Or, of course, observing everything that the user is doing.

So Google responded and said: "We appreciate Bluebox responsibly reporting this vulnerability to us. Third-party research is one of the ways Android is made stronger for users. After receiving word of this vulnerability, we quickly issued a patch that was distributed to Android partners, as well as to AOSP." And I didn't look up that acronym. You know what that is, Leo?

Leo: Android Open Source Project.

Steve: Ah, perfect.

Leo: So it means more than that. It means the manufacturers of Google-approved Android devices, any of the Android devices that have the Play store on it or AOSP handsets.

Steve: Good. And then, just finishing...

Leo: Actually, wait a minute, nope, take it - might be wrong. I think AOSP is the opposite. It's the Android Open Handset Alliance - oh, it's so confusing. Anyway, it's the other Android folks.

Steve: Well, you've got the acronym right.

Leo: I know the acronym.

Steve: Even though we don't know who they are.

Leo: But the acronym doesn't tell you exactly what it is because Google's obfuscating it.

Steve: So, and then Google said: "Google Play and Verify Apps have also been enhanced to protect users from this issue. At this time, we have scanned all applications submitted to Google Play, as well as those Google has reviewed from outside of Google Play, and we have seen no evidence of attempted exploitation of this vulnerability." And so the good news...

Leo: Yeah. And so if they gave it to OEMs, that's the Handset Alliance people. And then giving it to AOSP means they put it on the open source server so that people

who make nonofficial Google Android devices can also patch it. So everybody, in other words, who's using Android. AOSP is like...

Steve: So there is a Google bug. It's been given the Google bug 13678484. And Bluebox Security has put a scanner up in the Google Play store. I imagine you can find it. I've got the link in the show notes. But it's Bluebox Security Scanner. And it will scan your machine to verify that it has been patched for this problem. And the good news is this is - it's not like one of those things where the application - malware could be hiding some behavior which, for example, yep, there it is on the screen, and it's a free download, so anyone who's interested or curious or worried, a Bluebox Security scanner in the Google Play store.

But the point is this is easy to scan for because it's a security certificate that the application has to have and has to present in order to get these privileges. So it's simply a matter of Google running through, like knowing this is a problem, and running through all the apps to verify affirmatively the signatures on all of the certs that they carry. So...

Leo: Just to be clear, you don't need this to fix your problem. This is just to see if you ever got bit.

Steve: No, no. Actually, I think it's - from the notes it says that it checks to verify that you are no longer vulnerable. That is, that your Android device has been patched through the patching process.

Leo: I think it's also highly likely that, see, Google has its own scanner, which they keep up to date on Android. And it's highly likely they fixed that, as well, at the same time. So that scanner goes through every app you download and checks for known vulnerabilities. So I imagine...

Steve: Is that the Verify Apps that they talk about?

Leo: Yeah, yeah.

Steve: Okay. Yup.

Leo: And they update that easily. So my suspicion is you've got that already. So I don't know...

Steve: Yeah, and Verify Apps they said has been enhanced.

Leo: Yeah. See, the blue people - Bluebox is a business. This is a product. They don't charge you for it. But they would like to get - it's like Lookout. They would like to get it on your system.

Steve: Yeah.

Leo: Yeah. Now checks for the Fake ID vulnerability, but it's always been checking for other things, as well.

Steve: Well, you know, I was thinking about this. I mean, it's good that they found this. And I think it'll make an interesting presentation. And they did the right thing by disclosing responsibly. The sad thing is that it sort of takes the teeth out of their whole presentation that it's like, well, we found this, and it's been fixed, so nobody has to worry about it.

Leo: Now you know why people hold onto these and don't tell Google or Apple.

Steve: Yeah.

Leo: But they did the right thing. Please do the right thing.

Steve: Yes, they did.

Leo: Even if it takes the teeth out of your presentation.

Steve: Yeah, well, because the problem is this one in particular is so bad that if this were, for example, a zero-day discovery, it would be really bad. If we found it being used rather than them discovering the problem, that would be a whole different deal because it would just take time to push out the change and get everybody to respond. And there would be people hurt in the meantime. This way nobody was hurt. But it's a lot less exciting. Sometimes that's a good thing.

Okay. Speaking of exciting, or maybe not, I'm not sure, we have the final volley in this pretty much ridiculous back-and-forth between Verizon and Level 3. And so I want to discuss it for two reasons. First of all, additional information about the way they feel about this issue, the peering bandwidth issue, comes out in this. And we get a conclusion. So now we're back to - last week we talked about Level 3's response to Verizon's first volley. And so now we have David Young again from Verizon, responding to Level 3's response from last week. And he makes some good points, I think. I'm not taking sides. I'm interested in sort of the technology still, and understanding, like, how they're thinking about this.

So David Young's posting from Verizon called it "Level 3's selective amnesia on peering." And what's interesting about this, and we'll get there in a second, is you and I, Leo, talked about the Level 3/Cogent problem, and it affected me because my T1s are on Cogent bandwidth, and GRC's famously - and there I said the word, only once - in the Level 3 datacenter. So I was cut off from my own servers when they had that peering battle.

Leo: Wow.

Steve: Anyway, so and you may remember, it was a few years ago, I couldn't get...

Leo: Oh, yeah, yeah, I do, yeah.

Steve: I couldn't get to GRC because Cogent and Level 3 were fighting. So David Young writes: "Last week Level 3 decided to call attention to their" - okay, now, again, the wording is of course loaded - "call attention to their congested links." It's not our congested links, it's Level 3's congested links, even though they just interconnect each other's routers. So I'm not sure why it's Level 3's links that are congested. Seems to me it's both...

Leo: Ehhh, it's like when your wife says, "Your son is in trouble again." Mm-hmm, mm-hmm. Yeah.

Steve: Exactly. So "Level 3 decided to call attention" - and actually, no, I would argue that Verizon called attention.

Leo: Yeah, who started this.

Steve: With, as you keep pointing out, the bright red - the only part of the network diagram that was red in Verizon's original posting, as you properly note. So "call attention to their congested links into Verizon's network." Okay, so that's important, as we'll see in a minute. So he's saying: "...Level 3's congested links into Verizon's network. Unlike other content delivery networks, which pay for connections into ISP networks to ensure they have adequate capacity to deliver the content they have been hired to deliver" - and again, remember, this is certainly partisan - "Level 3 insists on only using its existing settlement-free peering links, even though, as Level 3 surprisingly admits in their blog, these links are experiencing significant congestion. Level 3's solution? Rather than buy the capacity they need, Level 3 insists that Verizon should add capacity to the existing peering link for additional downstream traffic, even though the traffic is already wildly out of balance."

So there again we get this, you know, all of this seems as if Verizon's saying these are Level 3's links because they are in this data flowing into Verizon's network. And then we also get this notion that, from Verizon's viewpoint, what a content delivery network pays ISPs to do is to accept their bandwidth to ensure, as David writes, they have adequate capacity to deliver the content they've been hired to deliver. So, and this notion of wildly out of balance, which I've been talking about as we've been looking at this.

So continuing, David says, "Level 3 has been on the other end of these peering disputes in the past," which we know is true. "In 2005, they found that Cogent was in violation of their peering agreement. Explaining the situation in a press release describing the dispute," and he provides the link, "Level 3 said, 'Free peering, also referred to as settlement-free peering, is a contractual relationship under which two companies'" - and this is Level 3 in 2005, referring to their dispute with their Cogent - "'under which two

companies exchange Internet traffic without charging each other. In order for free peering to be fair to both parties, the cost and benefit that parties contribute and receive should be roughly the same. For example, Cogent was sending'" - and this is "was," so that means this was posted after Level 3 broke the links, essentially, cut off the peering relationship.

"'For example, Cogent was sending far more traffic into the Level 3 network than Level 3 was sending into Cogent's network. It is important to keep in mind that traffic received by Level 3 in a peering relationship must be moved across Level 3's network at considerable expense. Simply put, this means that, without paying, Cogent was using far more of Level 3's network, far more of the time, than the reverse. Following our review, we decided that it was unfair for us,' says Level 3, 'to be subsidizing Cogent's business.'"

And then David says: "Level 3 informed Cogent that they would be terminating their peering agreement unless Cogent made alternative arrangements." And then back to Level 3's statement at the time: "'We then contacted Cogent's senior management to offer to discuss alternative commercial terms to allow the continued exchange of traffic. Cogent refused.'"

So then David says: "Level 3 put the onus squarely on Cogent for failing to make alternative paid arrangements for the benefit of customers to handle the unbalanced traffic as other firms had." And then back to Level 3: "'Those firms chose to enter into agreements, either with Level 3 or others, to obtain the appropriate connectivity and keep the interests of their customers paramount.'"

And then David writes: "Summing up their position, Level 3 said" - and this is the last Level 3 statement. "'To be lasting, business relationships should be mutually beneficial. In cases where the benefit we receive is in line with the benefit we deliver, we will exchange traffic on a settlement-free basis. Contrary to Cogent's public statements, reasonable, balanced, and mutually beneficial agreements for the exchange of traffic do not represent a threat to the Internet. They don't represent a threat to anyone other than those trying to get a free ride on someone else's network.'"

And then finishing, David says: "So what has changed for Level 3? Unfortunately, they are now the one 'trying to get a free ride on someone else's network' and failing to 'keep the interest of their customers paramount.'" And finally David at Verizon says: "Fortunately, Verizon and Netflix have found a way to avoid the congestion problems that Level 3 is creating by its refusal to find 'alternative commercial terms.' We're working diligently on directly connecting Netflix content servers into Verizon's network so that we can both keep the interests of our mutual customers paramount."

So anyway, some additional - so there's some intriguing ideas here, that is, that the way these top-tier providers feel is incoming traffic is a burden that they're carrying on behalf of their peering partner. And as I said, I've experienced this myself. When I was setting up my servers in Level 3's datacenter, they wanted to know what my own, just my own little piddling ratio of incoming to outgoing traffic was because, of course, it all adds up. If everyone that they were hosting was only serving and not receiving, then there would be an imbalance created by that datacenter, to some degree, before it has a chance to get diffused.

So I think there's validity to this. I don't think these people have conducted themselves very well, just yapping at each other publicly. But I think, as I hoped we would, we learned something about the way these relationships operate at the high end. And I got a better sense for what I was looking for, which was what is this about balance? Why is that important? And this explains it.

I got an interesting piece of feedback from GRC's HTTPS Fingerprinting page. Everyone will remember I pulled that system together, brought it online, I don't know now, maybe six or more - maybe more like a year ago because it was certainly before I started working on SQRL. And that's been going for a while. Anyway, a G. Evans used the Fingerprinting Feedback page to say, "In your paragraph about machine-resident interception," meaning client-side interception, "you can add Avast antivirus to the list. There is an innocuous settings checkbox that says, 'Scan secure connections,' with no other explanation. Sounds like a good idea, until I read your Fingerprints page in Firefox and noticed the lack of a green label in the address bar for GRC. When I hovered over the lock symbol, it said 'Verified by Avast' as opposed to 'Verified by DigiCert.' Oops. I immediately turned off that option in Avast, and now it's back to normal."

So I'm not sure that's a problem. It requires that you trust Avast. But essentially what that means is that, when this G. Evans, or presumably anybody, installs Avast, it's also putting its certificate in your - I assume this is on a Windows machine - in the OS's root store so that your browser will trust certificates that it signs. And then...

Leo: That's kind of not good behavior.

Steve: I know. And this is what - but the problem is, without doing that, the antivirus system can't scan your traffic. It would have to do it after the traffic were decrypted. And there's probably not a good way to get a shim in there. So what it's doing is...

Leo: You can always do the man in the middle.

Steve: Well, it is. It's a man-in-the-middle attack. I mean, that's what it is. And of course the problem is, if somebody got their certificate, or actually if somebody - let's see how this would work. If somebody reverse-engineered Avast and pulled the certificate...

Leo: Yeah, I mean, you're trusting Avast, basically.

Steve: Yeah. But my point is, I have to think this through, but I think that means anybody could get a hold of - a bad guy could get Avast and extract the certificate from it. And if they knew you were using Avast, then they could move the man in the middle outside of your machine. I don't see anything preventing them from doing that.

Leo: Ooh, that's not good.

Steve: No. That's scary, actually, because what Avast is doing is they're - when he went to GRC, GRC's certificate was sent to Avast, and Avast minted their own fake GRC.com certificate and signed it, and then sent it on to the browser. So the browser thought it was connecting to GRC with a secure connection, when in fact you had a man-in-the-middle attack, well, a man-in-the-middle presence, not an attack in this case. And you weren't actually getting my certificate from DigiCert. You were getting Avast's certificate that was like a fake GRC.com certificate they had just...

Leo: That should always be a red flag. That's terrible.

Steve: Just, yeah. And so...

Leo: Do other antiviruses do this? Or security programs do this?

Steve: Yeah. I've heard of others doing it. We know that appliances do it. And now here's an instance of Avast doing it. I'm trying to think of the other one.

Leo: Does McAfee do it?

Steve: I don't think so. I'm wondering if Kaspersky does it. I think maybe it's the one that does it. But, you know, we trust the Russians, so...

Leo: Yeah, of course. Why not?

Steve: That's right. Okay, now, Leo. The other day - we're now in Miscellany. I've got three things to talk about. I saw you guys running the whole "Building the Brick House" at high speed.

Leo: Yeah.

Steve: And I thought it was so neat.

Leo: Yeah. Three years ago John made a great time lapse of that, had the foresight.

Steve: He did. Now, okay. But at one point toward the end, everybody is moving around like their hair's on fire. And...

Leo: The tables change shape.

Steve: Oh, it was wonderful.

Leo: It's a party. We had a party.

Steve: And the control booth appeared, then it disappeared, then it came back, then disappeared. Then it spun around, and then it disappeared and came back. And it was, you know. Anyway, it's hard to stop watching it. It's mesmerizing. But it was just - it looked like a science fiction movie because, I mean, bzzzzzzzzz...

Leo: It's so cool.

Steve: Everyone buzzing around. But at one point the camera position was slowly moving. I don't mean it was rotating.

Leo: No, I know what you're talking about.

Steve: It was moving. How did you do that?

Leo: It's a special proprietary trick. No, it's called a "slider." So they have like an I-beam with a special, very, very, very slow platform that moves.

Steve: With a clockwork motion.

Leo: Clockwork, exactly. And you put your camera on that. It's designed for time lapses because obviously it has to move extremely slowly. So it moves...

Steve: Right, and that's what puzzled me is it was smooth. But I thought, wait a minute. Someone, like, I didn't know if you were going up, stop-frame animation, and somebody was moving it. But it didn't look like that. It was really smooth. And but for it to be done in time lapse - okay, there it is. Yup.

Leo: In fact, if you watch "House of Cards" - do you watch "House"...

Steve: You went to all the trouble of - oh, of course.

Leo: Okay. Watch the beginning of "House of Cards" next time you watch the show because they have a wonderful title sequence which is all time lapses of Washington, D.C.

Steve: Yes, yes, yes.

Leo: And many of them are done with a slider. So they slowly move as the time lapse is going. It's a wonderful effect. And, yeah, if you're not paying attention, it just looks really cool. And obviously most people don't because you've seen that "House of Cards" opening many times.

Steve: But them, it's like, well, of course, okay. But you guys...

Leo: Hey, we, uh, we, uh, we're cool. No, credit to John Slanina, JammerB, because he did a great job. He had the foresight to know that we would want that. What you'll also notice is the camera moves a little bit because we boarded it up at one point. Did you notice that?

Steve: I thought I saw, yeah, like some construction guy put a big steel I-beam right in front of it. And I was like, whoops.

Leo: Well, where we put it, eventually a wall was going to go. So you see it, and you see the wall come in front of it, and then you have to - we had to move the camera to get around that.

Steve: Is this available online for our listeners who haven't...

Leo: John, did you put - did you ever put that online, John? Here, let me get John's - did you ever put that online? We probably have it on inside.twit.tv somewhere. Huh? Yeah, give me a link. All right. So he's getting - so, yeah, it's probably on our blog, inside.twit.tv. But I just was there looking for it, and I didn't see it. It's maybe very deep. It's been three years, after all.

Steve: For what it's worth, it's absolutely fun. This thing starts with an empty room that doesn't look anything like, well, it doesn't look like anything. And you see them zooming around at warp 10, building this.

Leo: Is it on YouTube? Oh, it's on YouTube.

Steve: It's got great music. It's got a nice soundtrack. I mean, somebody - it was really well put together. And then there was also - it switched to a time lapse in your office, and we see your office being built with the beautiful wooden cabinetry and everything.

Leo: Yeah, I can't believe three years later, and a million and a half dollars later. If you go to YouTube and search for, I'm told, "TWiT time lapse," you will find it. Yeah, there it is, Studio Upgrade Time-Lapse. Only 21 - which one? This one? No, no, that's the old one. So there's a studio upgrade. That's when we put in the new lighting rig in The Cottage. The one you want is the TWiT Brick House Time-Lapse. And for some reason I'm in a Santa Claus outfit at the beginning. Is that the one, John? Am I in the wrong - oh, this is me introducing it, for some reason, again, in a Santa Claus outfit with a fake fireplace where you're sitting right now, Steve. Anyway, yeah, you can see the - I talk a long time, don't I. There we go.

Steve: At 1:00 p.m. Oh, there it is, yep.

Leo: Yeah. It was, you know, when we came in here, this was an empty - it was an

old furniture factory. It was a drugstore for 60 years. A software company was in here for a little bit. We tore out the walls of the cubicles, put glass in because they were not glassed in. My office is glassed in. And you can see the whole process, yeah. It's pretty fun. You can also see the day pass because...

Steve: Yes, as the sun is setting.

Leo: The sunset, yeah.

Steve: You can see it coming through the window. Yeah, oh, there it goes.

Leo: Yeah. He cuts out the night stuff because nothing happens all night long.

Steve: Very nice.

Leo: But you can see the sunset. You're looking from the west, so that's exactly what you're seeing as the sun comes down. It's pretty cool.

Steve: That's some - yeah, it's very cool.

Leo: I'm very proud of this. This was an amazing project. And there you go.

Steve: It's working. I heard you mention "Lucy" on MacBreak Weekly.

Leo: Yeah, did you see it?

Steve: Absolutely.

Leo: Did you hate it?

Steve: Eh. I was - I didn't love it as much as I hoped I would. But I was wondering about three quarters of the way through what they could do because it was just so ambitious, that is, in terms of where this was headed. It was like, what is the trajectory going to take them on? I like action, and I like kick-ass attractive girls; you know? And it's, yeah, I did enjoy it. I think for anyone who thinks they would like it, they would probably love it. And if you're not sure, then maybe wait till "Guardians of the Galaxy," which opens this Friday.

Leo: I was surprised how poorly reviewed it was. I'm a fan of the director, though.

He did...

Steve: Yes, Luc...

Leo: ..."La Femme Nikita," Luc Besson, and "The Fifth Element," which is one of my favorite movies. And I've interviewed him, many years go when he was first starting out. He's a French director. And he's, I think, super talented.

Steve: Yeah, no. I really - I loved the Asian super bad guys. And I like that genre of movie. So I thought it was fun.

Leo: It's kind of B.S. because I don't think that whole thing about we only use 10% of our brain is really true. But still, it's a good one.

Steve: Yeah, yeah. Okay, now, final bit of miscellanea. I wanted to put on people's radar a forthcoming and very exciting next-generation memory technology. We of course have hard drives that we've talked about. We've got static RAM, which is very fast, but has a density limitation because each bit has at least two transistors. You can think of them as inverters that are connected to each other. If you think about an inverter that is something where a one comes in and a zero goes out, or a zero goes in and a one comes out, if you connect that to another inverter and then connect the output of the second inverter back to the input of the first, it's stable. That is, the first one puts out a zero, which makes the second one put out a one, which goes around to the input of the first one that makes it put out a zero. So, and if you did something to force that to change, like you forced the input that was a one going into the first one down to zero, then that first inverter puts out a zero, causing the second inverter to put out a one, and keeps it in that mode. So that's called a "flip-flop." Two inverters back to back, connected to each other, is a flip-flop.

So static RAM is just that. It's a huge array of those. The problem is each cell takes up a lot of space because it requires that. So Dynamic RAM is simpler. It's just essentially a capacitor. And the problem with it is that the charge on the capacitor bleeds off, which is why Dynamic RAM needs to be refreshed. The refreshing is a scanning through the entire contents of the RAM to read the cells before they have fully lost their charge, to recharge the ones that were draining. And the advantage of Dynamic RAM is the cell is, although it requires refreshing, it's much smaller than a static RAM. And that means the Dynamic RAM can be much denser.

But both the flip-flops connected to each other and the leaky capacitor's Dynamic RAM, they're volatile. You turn the computer off, and they lose their charge. Now, we've talked about the surprising non-volatility of Dynamic RAM. Remember all of the freezing the DRAM with Freon and then quickly taking them out and putting them into a different machine. And it turns out that, if you make them really cold, you slow down the decay rate enough that they will hold their charge long enough to get moved into a different machine and so forth. So that.

Now we have non-volatile. And of course that's an often-mentioned topic because it turns out that the density has been increasing. And I've been talking about the technology of non-volatile RAM, essentially how it uses a transistor with a floating gate where the gate

is separated by an insulator, and charge is driven through the insulator and stranded out on that gate, but that allows that transistor's state to be read. So that's what all of our current non-volatile solid-state memory is.

The new technology - and this is something that popped up on my radar a couple years ago when HP announced they were seeing breakthroughs in it. HP calls theirs a "memristor." And the other term is "RRAM," as opposed to, for example, DRAM is Dynamic RAM. RRAM is Resistive RAM." And there have just been some new announcements of breakthroughs in that which are very exciting. The idea is that this uses the migration of, I think I read silver ions, through essentially a crossbar.

Imagine vertical conductive strips on one side, horizontal conductive strips on the other to create a grid, and the intersections are the bit cells. And essentially you can cause the interconnecting resistance to change, and it stays changed. So it is nonvolatile because you have to do something to it, basically drive a current through it in order to force this migration. And once you do, you permanently change the resistance at the intersection.

The reason this is exciting is it is a two-terminal solution. Unlike any of the transistorized solutions, which are large, this makes this very dense. And it turns out it is very high performance and has extreme endurance. So, for example, in terms of what we're actually, what they're actually making in the lab now, current flash technology that we have allows about 16GB to fit on a 200mm-square chip. So 200 square millimeters can hold 16GB of current flash technology. Using Resistive RAM, which is working in the lab, they can put a terabyte in the same space as 16GB. So this is shockingly more dense.

One company is Crossbar-Inc.com or just Crossbar is the company name. But if anyone wants to look, www.crossbar-inc.com, so named because that's the architecture of this. Oh, the other thing is not only is this technology super dense, but it lends itself to 3D. That is, a stacking of layers. So you can just keep building these crossbars back and forth, back and forth, back and forth, and stack them. And in fact you can build this on top of existing integrated circuits. So, for example, you could take a complex chip like a processor and have all of its real estate there doing its stuff, and then on top of it lay another layer of non-volatile RAM that then interconnects to it in order to create sort of a sandwich.

So quoting from Crossbar's page, they said: "With 20X higher performance and 20X lower power than NAND," which is the technology of flash, "and 10X the endurance at half the die size, Crossbar has shattered traditional technology barriers for [the] NOR [and] NAND [style] embedded memory applications and will enable a new wave of electronics innovation for" and blah blah blah, you know, PR stuff. But the technology looks real. HP expected to have it last year and to be commercializing it. But they haven't been heard from for a while. So maybe they're having problems with yield and so forth.

Moving this from the lab into commercial production is always challenging. We saw that, for example, when we were talking about the supercapacitors that we were hoping we would have by now, and somehow that seems to have gone on the back burner. But we may be looking at some serious increase in solid-state RAM performance.

Leo: Awesome.

Steve: And the good news is SpinRite will still be useful, which is why I'm so encouraged. All I will say this week about SpinRite, although I think maybe someone mentions it in the Q&A - I'm not sure, we do talk about SQRL a little bit there - is

yesterday at 5:21 in the afternoon, tweeting from Tweetbot for iOS, which is my favorite client also, someone named Ron Tyska just tweeted. And he said: "@SGgrc SpinRite revived a completely dead SSD, saved me $400. Thanks."

So it is really the fact that people, our own customers, began repairing and reviving SSDs with SpinRite that really got me motivated and reinterested in giving it a future because I was little depressed here as the world seemed to be going solid-state. And it continues to seem to be doing that, despite the fact that hard drives of course also continue to amaze us with how inexpensive they're able to create high-end mass storage. But it's clear that the nature of the economics is such, it's always going to be that these devices will be operating on the edge of reliability. They will be reliable enough that they do their job for a few years and then begin to die in some way. And the good news is SpinRite will be there and be able to pull them back.

And the other thing I noted was in reading through other people's comments that I don't bother sharing, something I've never said, but I realize the truth of it, is anything SpinRite can fix, it would have prevented. And I think that's absolutely true. Anything it can fix, it would have prevented. Which is a way of thinking about it from a preventative maintenance standpoint. Lots of people use it after it's pulled them back from the grave, and they understand what it does. But if it had ever been used before the drive got into that shape, that would have never gotten into that shape. So that's an interesting way of phrasing it.

Leo: That's kind of cool. And that's why it's the world's best disk recovery and maintenance utility. Questions...

Steve: Right, and preventative.

Leo: And preventative, yes. Questions are ready for you, Steve. Are you ready for questions?

Steve: Let's do it.

Leo: Let's do it, starting with Adam P. He says: How do they do it? You praised SpiderOak - we should refer people back to that episode of Security Now! where you talk about all the different, or not all, but many different cloud storage solutions [SN-349]. And you did say, in fact, that your favorite was SpiderOak because it's Trust No One. They never have access to a user's private keys. Well, I'm wondering where you got that information. According to this SpiderOak Q&A page, FAQ page, if your hard drive crashes you only need your password to get your data back. Doesn't that mean they have to be storing your private key on their server? And, by the way, this makes their "zero knowledge" claim complete bunk.

Steve: Okay. So this was interesting for a couple reasons. First of all, recently Edward Snowden disparaged Dropbox, which doesn't offer TNO security. And he specifically mentioned SpiderOak as what he would use because of their TNO operation.

Leo: As did you, which makes me think Edward Snowden listens to Security Now!. So hi, Eddie.

Steve: Well, I've been impressed by his technical knowledge.

Leo: Yeah, he seems to know this stuff.

Steve: He really does know this stuff. So I wanted to explain this to Adam and our listeners. It is absolutely the case that you must use a strong password. But your password can be used as the sole decrypter of a private key. You don't want to use your password directly as the key because then you could never change your password. So instead, your password is used to generate the key which then encrypts the actual key. But all of that can be done on your computer. And the encrypted data that whatever cloud provider has can be brought to you. Then you use your password to generate a key which then decrypts the encrypted key, which is used in turn to decrypt the contents.

However, as I said, the quality of the password is what matters. And this is the same technology, for example, that TrueCrypt or other properly designed hard drive encryption tools use, where, again, absolutely, you need a good source of entropy to create the original key. Then you need the user to create a password that will not succumb to brute-force attack. And brute-force attack needs to be thwarted by, for example, running it through hopefully a memory-hard and, in order to get time-hard, a password-based key derivation function, PBKDF, which then turns the password into the key which decrypts and encrypts the actual encryption key for the data.

So that technology - the good news is all of this is readily available. I'm using it in SQRL. Anyone who's doing TNO by definition is doing some version of this. Maybe they're using - there are variations, for example, that use public key technology. Everything I've just described is only symmetric key and maybe some hashing in there in order to...

Leo: But PGP is public key and uses a similar technique; does it not?

Steve: Yeah. Well, there are many different configurations. The one player that - oh. And I should also mention, Leo, that when you were taking a break, I announced that I was going to make time to go back and revisit the whole cloud storage deal because of the incredible drop in cost, the availability of higher bandwidth. We did the cloud storage podcast years ago. And it's time for an update. So I mentioned that. Then in the second week some of our listeners had found - I knew that there was an encryption, like a drive encryption Wikipedia page. It turns out there is a cloud provider Wikipedia page.

And so what I had intended to do originally was create a public spreadsheet where I would put everything that I found about all these providers. Turns out it's largely there. But what I want to do is to focus on the idea of using a TNO client on the user's end and sort of any provider that you want to use to store your data. So I'm still going to do that.

But I want to mention Boxcryptor. They're also TNO. And I've been just doing some preliminary poking around, and they really look good. I need to get some experience with them. And they have a Boxcryptor Classic which does not use a licensing model. I'm not a big fan of rented software. I want to just buy it and own it and not have to pay them

annually for the privilege. Their newer product has additional features and is only available in an annual payment basis. But they still offer, with the same technology, the Classic version, which I like because it's give them money once and then you own it. And it is really worth looking at.

So anyway, we will be coming back to this as I had promised, with a little different take, though, because now I found out that other people have done a phenomenally comprehensive job of putting on a grid all of the cloud storage providers and breaking them down in a very useful fashion.

Leo: I've been really happy with File Transporter, which is not really cloud. You make your own cloud. You have a local hard drive which you sync with. And I have one here and one at the office. And it's using strong encryption on the hard drive and SSL for transport to the hard drive. And then the two, home and office, sync. So it works, I think, really nicely for me. And at no point does the company have access to my data. I mean, as long as they haven't put a backdoor in there. So I get the benefit of a cloud storage solution with nothing stored on the Internet. It's all stored here.

Steve: And it's a little appliance?

Leo: Yeah, it's the cutest little thing.

Steve: I think I've seen a picture of it, yeah.

Leo: Yeah. Well, I have one that just looks like a little - it's hard to describe.

Steve: Sort of a little pagoda kind of thing.

Leo: Yeah, exactly, it's a little pagoda thing. And then I have another one. Most recently they've started making a $99 dongle that you attach your own USB drive to.

Steve: Nice.

Leo: And that works great. And so there's no monthly fee. There's no actual cloud storage. You're your own cloud. This to me has worked really well. And because I can put 3TB on it, there's a lot of storage. And I feel like there's nothing on there that I wouldn't want the NSA to see, but it's just nice to know you control it.

Carl in Philly has a short comment about your future: I'm a happy user and a listener for many years. I eagerly await the completion of SQRL and then, once you've finished that, SpinRite 6.1, and then - boy, this guy's really looking ahead - v7. Please consider any improvements you could make to SpinRite specifically for solid-state drives because that's going to become more important in the future. I think you probably are aware of that, Mr. Gibson.

Steve: Yeah, and even SpinRite 6.1 is going to pay homage to SSDs. I will specifically call out ways that SpinRite 6.1 should be used with SSDs. For example, and we've talked about it already, but I'm just going to - I'm going to sort of make it more clear, for example, that Level 2 is what you want to do normally on an SSD. Although Level 4 is not going to hurt it, it is writing. And so we know that writing takes a tiny toll on an SSD. But reading doesn't. Reading is a nondestructive process and doesn't wear the SSD. So running Level 2 over Level 4 would make sense.

But absolutely, for anyone who's interested, I mean, it's really because, as I said before, because SpinRite has a future with solid-state because it's always going to be the case that manufacturers are going to be relying on error correction. And they're going to be putting as many bits as they can get away with, and then they're going to push it a little further than they probably should for the realities of commercial competition, competitiveness. And so SpinRite's definitely going to stay in the game.

Leo: Well, I like that. Ben in Australia shares his thoughts about website hosting providers: Steve, I won't bore you with too much praise here, but please consider the usual sentiments. After the question you had on the last show, I thought I would pop in a quick two cents. I've been a happy DreamHost customer for years now. I'm on shared hosting, since it provides ample space and bandwidth for my needs. They also offer VPS solutions, which I've used in the past to run proprietary services, et cetera. Currently hosting four websites under my account, have set up customers on other accounts with them, all without any issues. My hosting, DNS, databases have all been migrated over the years as they've upgraded servers, and I have not noticed any downtime or issues related to this.

The features and specs DreamHost offers is superb IMHO, and I have not come across such a flexible hosting provider in my 15-year career as a developer. I have used many other providers including 1&1 for clients' websites and email, as the good Padre suggested. Have not experienced problems with 1&1, but found their - is this just an ad? But found their feature set to be very limited.

Steve: No.

Leo: It's a plug. Okay. I'm not going to go on. It's just a plug for DreamHost. That's nice. Do you have - is there a question there?

Steve: No, but we discussed this with the Padre.

Leo: There are about 3,000 hosting companies.

Steve: Well, and the original question was someone wanted to use a hosting provider, and I knew of DreamHost. But someone said to me that they had been purchased by someone, or that they'd gone out of business or something. Padre was saying that 1&1 was the one he was suggesting. So I just wanted to let people know that this guy...

Leo: Yeah, I don't want to get in the business of plugging hosting providers because

there's so many of them, and there is nothing to distinguish DreamHost and 1&1 from others.

Steve: Okay, well, that's good to know.

Leo: And unless you've tried them all, I wouldn't make a plug for any of them. You use your - you do your own hosting. We use SoftLayer. We know SoftLayer's good because that's what we've been using for eight years. We're actually moving many of our machines to a hosted Drupal solution soon, I think. But I don't want to get in the business of plugging these guys.

Steve: Okay.

Leo: Thank you, though. Bill in Miami writes: I am overjoyed, for once, that Steve is dead wrong. Well, shall I go on? Steve, at the same time I was listening to your podcast containing the grim report that XP registry hacks, or that THE XP reg hack appeared to work no more, my one XP - oh, the one that we were talking about that makes XP look like the Embedded XP.

Steve: Yes, XP Embedded.

Leo: Yeah. My one XP machine was cheerfully updating. Since then, all three of my other XP machines have done the same. I'm not sure why you had problems, but it doesn't seem universal at this point.

Steve: Okay. So I'm glad for the feedback. I've heard from other people, and so I just wanted to correct the record. And you'll remember, Leo, that I said I was going to apply the hack on an XP SP3 machine, my little mailing station, where I have a weighing scale and a label printer and so forth. I did that, and it brought in a bunch of old updates but hasn't continued to do any additional updating. Yet other people have continued to cruise along. So I can't explain it. I don't know why this one machine of mine isn't doing it. So that's anecdotal. It is definitely the case that XP machines that have that link added to their registry are continuing to update and receive the security patches for probably till 2019, as I remember, five more years.

Leo: Incidentally, a number of people in the chatroom saying that they've been very unhappy with DreamHost. So I really don't like anecdotal recommendations. When you go out and you review, as you often do, these guys, and really do the job, that's a different thing. But anecdotal recommendations are as good as the person who's telling them. And I really don't want to start getting people writing you, trying to get a recommendation for any, you know, hey, let me just recommend my product.

Matthew Inderwiesen in Orlando, Florida has two quick questions: Steve, thanks as always for all you do. Two questions: For SQRL you mention on non-Touch ID devices you'd like the user to just to enter the first - or last, can't remember - part

of their password to reconfirm they are who they say they are. This sounds like a great convenience because I'm going to be using a password I can't remember. Ha ha. But the trade-off in security, wouldn't that require you to store more than just a full hashed password and compromise it a bit? Curious what ingenious ideas you have to pull off this nifty trick. And, two, I've been looking at the online comparisons of backup storage services - wiki link - and you're right. What does "wiki link" mean? That there's a link there?

Steve: Oh, no, he was just saying on Wikipedia.

Leo: Oh, that one that we were talking about. Okay. And you're right, it's pretty darn comprehensive. I just scanned through it on the show, and boy, yeah, it is long. One thing is I was really hoping to get your thoughts or list of services that encrypt files locally that are TNO and then back up to whoever. I'm still going over the list, and some of the answers may in fact be there. But ultimately my question or request was if you would still not mind maintaining a page with your current recommendations for software that could do this, just like any preferences you had - "Steve's Pick" for hard disk encryption, coffee, razor blades, anything else that you use because it lives up to your standards. So two parts to that one.

Steve: Okay. So really great question about the way SQRL's going to operate. As we know, SQRL is able to represent the user anonymously and in a sticky fashion so that it generates an identity per site. And every time you go back, you're able to reassert the same identity. But the problem is we still need to prove that we are who we are to SQRL, since we've empowered it with the ability to represent us to the entire Internet. And still we don't have a good way to do that.

Now, Touch ID begins to get there, except that we know that even that can be spoofed. Your fingerprint can be lifted off, and people have demonstrated this, lifted a fingerprint off of a glass, turned it into something that spoofs the capacitive sensor in the iPhone Touch ID and unlocks the phone. So I really think still a password is the best solution. Also, your fingerprint is not robust in terms of constitutional protection in the U.S., whereas something you know is. You cannot be compelled to relinquish a password. That's considered a violation of your Fifth Amendment rights against self-incrimination.

So the idea with SQRL is you only have to remember one. But if that's going to be the case, it's got to be a really good one because, as with everything, if somebody captured, if somebody grabbed your phone and ran off with it, for example, and you had SQRL installed there, the weakness would be brute-force attack, as is always the case with a password. I've gone to extreme measures to make that incredibly resistant. In fact, it's the most resistant system ever. I took the Scrypt PBKDF and created something called EnScrypt, which iteratively uses Scrypt, which itself is memory hard, in order to also make it time hard, to the point where, when you put your full password in, it takes five seconds of full saturation processing time to turn the password into a symmetric key. And there is no way to speed that up. So that's burdensome because five seconds is too long to wait every time you want to reauthenticate.

So what I've done is I've created the notion of a hint. And the idea is that, the first time you use SQRL, when you bring the system out of hibernation, or it boots up, or you unlock your phone, that is, when you're starting what we could consider a session, for the best security you should be required to enter your really long password. And the

idea, of course, is this is the only one you ever need to remember, sort of like a LastPass password, where then SQRL does all the rest for you. So you enter that once. And then you are asked to reauthenticate as you use SQRL, moving around the Internet, logging into websites.

Well, I don't want to require that you go through the pain of typing in that long password every time, certainly not waiting five seconds. That's a one-time process for maximum protection against anyone ever being able to brute-force your password. It just becomes impossible if every single guess takes five seconds. And it actually takes 16MB of RAM that's actively in use so it cannot be put on an FPGA or ASIC because there's just no way to give individual instances 16MB of RAM. And that can be scaled easily in the future as systems evolve, and it starts becoming feasible to give GPUs and ASICs and things more memory.

The idea then is, when you enter the password, that five-second process synthesizes the symmetric key. At that point the first "n" characters - and the user can set that. It defaults to four. And actually I spent 12 hours yesterday working on exactly this code, and that part of SQRL is finished. I'm getting very close to having the whole user experience portion of SQRL finished. Then it's just the protocol stuff, and we're done.

So at the moment that the full password is decrypted from this five-second process, and there's no way to short-circuit that and no way to get to the end without going through every single iteration to arrive at that, at that moment the first "n" characters of the user's password are reencrypted for one second. And that reencryption is saved. So as long as that exists, the user is able to reauthenticate using only those first "n" characters. And while it's not as safe as requiring a whole password, in terms of the logic of use, really what we're wanting to do is we're wanting to prevent anyone else from picking up your phone after you have authenticated with your long password and then be able to impersonate you.

So the idea is, every time you use SQRL during a so-called "session," it'll pop up when you're wanting to log into a site and say, what's your hint? And so you give it the first "n" characters. You go bing bing bing bing and say okay. And a one-second process then, which is not going to be burdensome for people, ensues, which decrypts the in-RAM encrypted under the hint key, and that allows SQRL to function. And then anything that interrupts the session, your screen blanking - and these are all options that can be tuned in the UI - your screen blanking, the system hibernating, if you walk away from the system for "n" number of minutes, and you're able to set that, then SQRL automatically flushes that in-RAM key. And the next time you're asked to authenticate, you have to use the full authentication. But that again creates the temporary authentication which you can use for the duration of a session. So, oh, and if you ever mistype just that hint, if you don't get the first four correct the first time, SQRL wipes the hint data from RAM and then reprompts you for the full password.

So it's a nice tradeoff. If you want more than four characters, you can. But again, it takes one second per guess. And if you've chosen a good password, and it's long enough, in terms of cracking, anyone who picked up, who sat down at your computer or got your phone while you were authenticated, one mistake, and then they have to use the full password. So I think I've covered - it's a tradeoff. Oh, and if you don't like that, you can turn that off. You can turn the whole hinting system off, requiring you to enter your long password every single time. Oh, and you can also control that five seconds is the default. You can control that. So if you'd rather use a long password, but only have it take two seconds or one second, then you can do that, too. Lots of flexibility. And it's all working, by the way.

Leo: And how about a Steve's Picks page?

Steve: A lot of people ask for that. Maybe someday. I just - I can't do anything more than I'm doing right now.

Leo: We have a TWiT Picks page. We'll start putting your picks on the TWiT Picks page. How about that?

Steve: Okay. I like it.

Leo: That's right at the front on TWiT.tv. Matthew Urch, writing from - I like that, Urch, U-R-C-H, writing from Toronto Ontario, wonders about cloud storage encryption. Seems to be the topic of the day. Love Security Now!, been listening since I discovered it a few weeks ago? Welcome, Matthew.

Steve: Yeah.

Leo: My question has to do with encrypting the contents of cloud storage. I know I can simply create a TrueCrypt volume I store on the cloud, but then every time I make a change it's a massive file that has to be re-synced. Is there a solution to ensure my data is protected on the cloud that is a little friendlier to the syncing nature of those cloud solutions? I would think file by file would be better; right? I found CryptSync, which is a Google code page, C-R-Y-P-T-S-Y-N-C, which seems like a good solution, but I lack the knowledge to vet it myself. Have you heard of it? Or is another solution out there?

Steve: So I think I sort of stepped on this one already because I remember when I chose this, this was the reason I wanted to mention Boxcryptor, which I have, and I have been looking at. And everything about it I'm liking. They've got full documentation of their crypto. They lay out what they do. You can buy it. It's completely cloud provider agnostic, so you can use it on your own systems. You can use it on your own remote storage or on cloud providers. You could encrypt folders remotely and then have other ones that are not encrypted.

So anyway, I would say to Matthew, take a look at Boxcryptor. I need to look - essentially what I'm going to do is, rather than try to talk about every remote cloud provider under the sun, and as we've seen there are just too many of them, I just can't, I mean, people, ever since I mentioned I wanted to do this, it's like, people are writing about ones I've never heard of. I'm less interested in the monolithic, oh, don't worry, we'll take care of you. I understand there's a market for that. Jenny is using the one that is a frequent sponsor of the show, and I'm blanking on it.

Leo: Carbonite.

Steve: Carbonite, yes. And that's perfect for her. It provides the set-it-and-forget-it

backup that she needs. But for our more techie users, who want to roll their own, what I really want to find is the right client-side tool. Oh, and the other thing about Boxcryptor is cross-platform - Windows, Mac, iOS, Android. So they've got all that covered. So I would say take a look at that. And I've not looked at CryptSync, so I can't speak about it, but I definitely will. And we'll end up doing a roundup of all that.

Leo: You remember - I should tell you what I use, just to throw it in. You remember Phil Zimmermann, who, I'm sorry, not Phil Zimmermann. That's the PGP guy. Who did PKZIP?

Steve: Phil Katz.

Leo: Kaplan, Kaplan, Katz, right.

Steve: Katz.

Leo: His company, PKWARE, does a program called Viivo, which the idea is you continue to use Dropbox or whatever, and it's public key file encryption, so it's what you talked about with Pre-Internet Encryption, PIE. Right?

Steve: PIE, right, right, PIE.

Leo: Yeah. So that would work; right? If you did it file by file before you sent it up to Dropbox?

Steve: Yes. And they are one on my list of it done right.

Leo: I've used them. I'm not sure - yeah. So okay. Good. There you go.

Steve: Yup.

Leo: I don't know how much that costs. I can't remember. It's not free, though. It's a commercial product.

Brian Mooney, Springdale, Arkansas wonders about the best many-router configuration for providing network isolation: In a recent podcast you were discussing WiFi-networked light bulbs and the Internet of Things. You recommended keeping these sort of devices on a separate WiFi network, for instance, the router's built in guest network. Or, if you can't get that, a second router. Two questions: First, how robust and secure do you think a typical consumer router's guest network function is - actually I've been wondering that myself - at isolating traffic from the rest of my home network?

> Second, if I were to add a second router and provide a guest WiFi network, tell me about the proper physical configuration. Should it be plugged into my existing router, or placed between my existing router and my cable modem? Should I have three routers in a Y-configuration with one end router providing guest WiFi, the other providing home WiFi, both plugged into a third router connected to my modem. What do you say?

Steve: Okay. So, yes. This came up because we were talking about, just sort of in general, the security concerns about having all of these random appliances on someone's WiFi network, where we're seeing already the not-surprising security vulnerabilities because people are rolling their own solutions because we are not yet in a position where the standards have been adopted. We've talked about the three different standards which are on their way, but they're months old at this point, so they've not yet begun to appear in products.

So I think the only way to be comfortable is to have two WiFi networks, one that your critical infrastructure stuff runs on, that is, your personal computers and so forth, and another one that is untrusted, the so-called "guest network," where you can also have your light bulbs and your refrigerator. And didn't we have a pasta machine on that? Or was that, no, that was SpinRite that was going to make pasta.

So, okay. So here's the issue. If you had two routers in series, so you have your cable modem Router A, and then Router B is plugged into Router A. And these are both WiFi routers. Then the nice thing is - remember that we sort of can think of routers like one-way valves. They allow stuff out, but not in. That's the nature of NAT. So what that means is that your highest protected data, if that was behind the innermost router, Router B, then there is no visibility from Router A into Router B.

Now, the problem with this is sort of what level of hacking might go on because, if your untrusted router is A, and you've got untrusted devices there, that's still using Ethernet as its under-layer technology. The physical protocol is Ethernet. And there are still problems with ARP and ARP spoofing. That is, it might be possible for a device on the untrusted network to convince Router B that it is the gateway, using ARP technology. And that would cause Router B's traffic to go to the device on Router A. So that's a problem.

If we reverse the roles, and we make the untrusted network as the inner one, then the problem is its traffic is going to be going through the trusted network, and that just doesn't seem right, either. So what we've demonstrated is neither series connected routers are robustly secure. If you had to choose one or the other, I think I'd rather - an attack down at the MAC address and ARP level would be pretty sophisticated to pull off. So I think I'd rather have the inner network be the trusted one and the outer one not be trusted.

But routers are now cheap. Especially non-WiFi routers. I mean, everyone probably has them in the attack from before WiFi. So the absolute strongest solution is as Brian suggests, the Y configuration, where the cable modem is connected to a non-WiFi router to which both of the WiFi routers are connected. Now essentially both networks are at parity, but they are completely cut off from each other because the routers route IP packets and do not route Ethernet packets, that is, they are not Ethernet bridges, they are IP bridges. What that means is that the routers are moving IP packets across, but they are creating completely separate Ethernet networks.

So now we technically have four Ethernet networks. There's the little tiny network between the cable modem and the first router. Then there's the two networks between each of the, well, actually we've got five Ethernet networks. The one from the cable modem to the first router, then the two Ethernet networks to each of the WiFi routers, and then the two more ethernet networks behind each of the two WiFi routers. And every single one of those is disjoint and cannot be attacked.

So that's the way to set things up if you want maximum security. Use any standard non-WiFi router - it doesn't have to be non-WiFi, but you don't need it to be WiFi - as that first one connected to the outside world, to the cable modem, and then a pair of WiFi routers that are essentially peers of each other, but there's no way for them to have any traffic hacks between them because there's just no way they have any control to get across to the other guy's network. As for how secure the current guest network functions are in routers, I have no idea. We're seeing a lot of attacks against these routers. So I wouldn't bet that, while the technology is trying to be secure, I just don't know if they've pulled it off.

Leo: Question 8, Mark Harrold in Sydney with a question about SQRL: I've been very interested in SQRL since you first mentioned the idea. So today I was eager to try out Ralf Wondratschek's Android Client. After testing for a while, I made a suggestion to Ralf regarding the inclusion of an auto logout feature, something I think banks would be keen to see.

This got me thinking about the potential for widespread acceptance of SQRL. If, for example, a developer decided to allow very short authentication passwords, let's say four characters, and the client were released, this fact alone might limit SQRL's acceptance by financial institutions. Could this occur? And if so, do you agree this could weaken what is otherwise a rock-solid solution to replace usernames and passwords?

I guess what I'm suggesting here is a certification process where SQRL clients must meet minimum specs. Banks and the like might be then able to reject clients that are not certified. Thanks for the great podcast. Wish you every success with SQRL. Mark Harrold, Sydney.

Steve: That's really interesting. I hadn't thought about that before, I mean, in nine months, because one of the things this does is, as Mark rightly points out, SQRL isolates, moves, essentially, the authentication up to the client, where now the user is authenticating themselves to the SQRL client. And then we've got absolutely world-class robust authentication between the SQRL client and the website. But what that means is we're blinding the website from any information about how well the user is authenticating with the client. That is to say, with current technology, we are all hitting the problem of, especially back in the old days, of websites saying, oh, your password must be at least eight characters because, when we're giving the site our password, the site is able to make some evaluation of how secure the password is. In this situation, there's deliberate isolation so that sites don't have any way of knowing how the user's authenticating to their SQRL client.

Now, I've built in a password complexity acceptor in my SQRL client, which is kind of cool. The gang in the newsgroup will be pounding on it, and we'll see what they think about it. It separates out the password. It classifies each character as an alphabetic uppercase or lowercase, special character or numeral, or something else because it has to be multilingual also because we've now 60 languages that people are waiting to

translate the user interface into. So I had to accommodate multilingual passwords, as well.

And then it looks at - it examines the password for transitions between those classes, total number of unique characters, so that it rates like a chain of ones or chain of dots or something, it gives you credit for them, but not as much. And so number of total length, number of repetitions, total number of unique characters, and transitions. And then it puts that into a formula to come up with a complexity value. Oh, and it gives you a meter as you're entering your password, sort of training you and helping you choose a good password. But I've implemented that. Nothing forces all SQRL clients to do that. I hope they will. But Mark's right.

I don't know, I haven't looked at Ralf's Android client. He's got one running, by the way. But if it allows you to put in a short password, that's a concern. So I don't know what - I'll talk with the gang over in the newsgroup and see what they think. SQRL could declare the password length that the user is using in the protocol. I don't like that. And of course the client could be lying. So that seems kind of flaky. But Mark certainly raises a good point. It's really up to the user to strongly authenticate themselves so that their use of their client is not hacked by somebody else.

Leo: Fair enough. Question 9, Jeff Cours was wondering about SQRL's PRNG, Pseudorandom Number Generator, and the LibreSSL forking problem: Steve, thanks for the show and your detailed explanations of security issues. In Episode 464, you covered the issue Andrew Ayer flagged in LibreSSL's PRNG, in which a grandchild of a forked process with the same Process ID as its grandparent could generate the same sequence of random numbers as its grandparent. In that episode, you mentioned that SQRL's PRNG - do you pronounce it purng? - would be...

Steve: I don't think anyone's ever tried to pronounce this.

Leo: Well, if you're going to say SQRL, you ought to say PRNG - pseudorandom number generator would be immune to this problem. Could you please elaborate? Is it because SQRL's random seed includes the time of the process's creation, and the PRNG automatically reseeds itself periodically? When you described SQRL's PRNG in 456, you also said SQRL doesn't need a lot of entropy to run. Would SQRL's approach work for an SSL library, or does SSL need too much entropy?

Steve: This question hit me because I've had the same thing on my mind ever since we got some sense for how poor the pseudorandom number generator in LibreSSL, which is OpenSSL, apparently is. And actually what Andrew found was that in moving from OpenSSL to LibreSSL, they neutered the ability to request a reseeding of the package's pseudorandom number generator such that children of children would inherit the same entropy pool and then start generating identical pseudorandom numbers, which is not good.

So I've been wondering, okay, I solved this problem easily. And, like, why doesn't something that's in such heavy use as OpenSSL, I mean, even having to request a reseeding is kludgey. And I think it's because they're just living in a land of "C," and they're thinking only as software people and wanting complete platform independence because doing this right requires some hardware. It requires dipping down into the hardware and using this, I mean, in the hardware is a huge, as I described when I talked

about SQRL's PRNG technology, incredible entropy. I mean, there was so much going on that is beyond a programmer's control, beyond anyone outside the chip's ability to know. The chip is maintaining all kinds of counters, branch predictors, cache misses and matches, and all this richness in order to get the performance that it has, that nobody outside the chip has any control over.

And so just asking the chip for some of that, now, I have the advantage that mine's running on an Intel - I'm writing in assembly language in the first place, and running on an Intel chip only. So I can do this. But, boy, to me, this seems like so mission-critical that there ought to be platform-specific, hardware-specific code in these packages that just reaches down and gets this because this is available on all the platforms, just in different shapes and forms.

Leo: And now, ladies and gentlemen, without further ado, our last question. And it comes to us from Robert Elliott of North Battleford - where is SK?

Steve: I don't know. I thought maybe you would know.

Leo: Saskatchewan.

Steve: Okay, yeah.

Leo: Does that sound like Saskatchewan? Yeah, I think so.

Steve: I'll go with that.

Leo: North Battleford, SK.

Steve: I don't think it's South Korea.

Leo: Could be.

Steve: They wouldn't have North Battleford.

Leo: North Battleford. That sounds very Canadian, North Battleford. He brings us a Nifty Idea of the Week: Odometer Incrementation. Hello, Steve and Leo. Blah blah blah. Everything. Included SpinRite. Went back and listened to all the podcasts, et cetera. At work my boss was talking to our delivery driver about a new fleet card he was issued and how, with this new card, the fuel purchaser needed to input the current odometer reading. Well, that's clever. I was not involved in the conversation. But when I heard that I was like Archimedes in the bath. Eureka! Would it be possible for the fleet card company to utilize the odometer reading as a OTP authenticating the purchase? Because the odometer is a forward-running counter, it

would logically never repeat itself and could be used to verify the authenticity of the purchaser.

In further discussion with my boss, he put forward the position it might simply be accounting to ensure that a person is not filling up a personal vehicle as well as the company vehicle. What do you think? And if it's a worthy idea, please include it in a Q&A. I told them I was going to send this in to you, and they lovingly called me a geek. It is Saskatchewan, by the way.

Steve: Ah. I think - I don't know. I think it's probably an accounting measure. But I love that Robert, as a listener of the podcast, thought of it as a One Time Password.

Leo: It's really great.

Steve: Because it has - it really is clever. I mean, it's a way of demonstrating, if you're always driving the same car, and in a fleet mode you're always - the delivery driver is using the fleet vehicle. Then, as he notes, that counter is, as we would say, monotonically increasing. And so an absolute security check, in addition to verifying that the odometer and the gas consumption are sort of staying consistent, so as his boss mentions, it could just be to catch the use of that gas card for non-fleet business.

But similarly, if anybody back there was checking that every single use of the card demonstrated a reasonable increase in the reading, that's not something that any thief could ever know. And so here we've got a six-digit counter, which could be anything between zero and 150,000, probably. And it's not going to go very far between events where it needs to be refilled. So it has absolutely many of the qualifications for a one-time password. I thought that was just...

Leo: So cool, yeah.

Steve: ...a really great observation.

Leo: Nice observation. By the way, what a surprise, here we are in beautiful North Battleford, Saskatchewan, and of course the world-famous North Battleford Travelodge sign. And with that, we conclude this...

Steve: Another uplifting week of security.

Leo: Steve's at GRC.com. That's where he hangs his hat. And of course that's where you'll find SpinRite, the world's best hard drive maintenance and recovery utility.

Steve: Yes.

Leo: You'll also find 16Kb audio versions of this show. You'll find full transcriptions written by an actual human person named Elaine. You'll find many of the things he talks about, including in many cases Steve's Picks because he sneaks some in there. There's no one page; but believe me, his opinions are well known, and they are all over there. Now, if you have a question for Steve, go there at GRC.com. And the feedback form is GRC.com/feedback. That's the one and only place you can leave questions for Steve. Don't try the chatroom, don't try to send him email. It's just not worth it.

Steve: Not because we don't love...

Leo: Didn't we used to get weird emails in return? I seem to feel like that was - for a while you would send out a strange email in response. Was that my imagination? Like don't email me.

Steve: I probably had a cranky phase.

Leo: You can go to our page, and you'll get full-quality audio and video, as well, of the shows, TWiT.tv/sn, or wherever you get your favorite podcasts. We do the show every Tuesday right after MacBreak Weekly, right about 1:00 p.m. Pacific, 4:00 p.m. Eastern time. That'd be 2000 UTC on the TWiT network. Hey, really great to talk to you.

Steve: Likewise, Leo.

Leo: Have a wonderful weekend. Or week.

Steve: I don't know what's in store for next week's episode. Probably I will get to the analysis of the web-based password managers. That's on my short list.

Leo: Very good.

Steve: Because remember there were those five password managers that there was some question about their integrity. So I want to give - that's going to require some study. And I think that's lined up for next week unless some new disaster befalls us, as does seem to happen with some regularity.

Leo: Every week. What fresh hell is this? Thank you, Steve.

Steve: Thanks, everybody.

Leo: See you next time.

Steve: Thanks, Leo.

Copyright (c) 2014 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/