SECURITY NOW!

Transcript of Episode #700

700 & Counting

Description: This week we discuss Chrome getting spell check for URLs; a bunch of Linux news with reasons to be sure you're patched up; some performance enhancements, updates, additions, and deletions from Chrome and Firefox; more Facebook nonsense; a bold move planned by the Japanese government; Ubiquiti routers again in trouble; a hopeful and welcome new initiative for the Chrome browser; a piece of errata; a quick SQRL update; and some follow-up thoughts about VPN connectivity.

High quality (64 kbps) mp3 audio file URL: http://media.GRC.com/sn/SN-700.mp3
Quarter size (16 kbps) mp3 audio file URL: http://media.GRC.com/sn/sn-700-lq.mp3

SHOW TEASE: It's time for Security Now!. Steve Gibson is here. Lots to talk about, including new systemd vulnerabilities. Linux users, listen up. We'll also talk a little bit about Chrome, a new feature giving us URL spell checking, and why TLS 1.0 and 1.1 are soon to hit the highway. It's all coming up next on Security Now!.

Leo Laporte: This is Security Now! with Steve Gibson, Episode 700, recorded Tuesday, February 5th, 2019: 700 & Counting.

It's time for Security Now!, the show where we cover the latest developments in the world of security and privacy, help you understand how computing works, and have a little fun along the way with this guy right here, Steve Gibson. He's the commander in chief of the good ship Security Now!. Aye aye, sir. What you pointing - that is not the logo you want. Maybe this.

Steve Gibson: No, no.

Leo: You want an "L."

Steve: Yeah, but which hand do I use? I can never get that right.

Leo: I don't know. Yeah, it's very important. Sometimes it's an "L." Sometimes it's backwards. Hey, Steve. You're no loser in my book.

Steve: Leo.

Leo: You're number one.

Steve: Well, thank you very much. Great to be with you again for, as I was saying before the podcast to you, I was tempted to title this one "The 700 Club"; but I thought, no, that's been taken. So let's just call this one "700 & Counting."

Leo: The 700th episode.

Steve: Yup.

Leo: Amazing.

Steve: And as is sometimes the case, there was no one particular crazy thing that stood out. So we've got a bunch of stuff to talk about. We've got Chrome getting spell checking for URLs. That's what I call it. That's not what they call it. A bunch of Linux news with reasons to be sure, extra sure you're patched up to date. And some performance enhancements, updates, additions, and deletions from Chrome and Firefox. There's some more Facebook nonsense. And I have a theory about Facebook nonsense we'll talk about. Also a bold move planned by the Japanese government. I've not been listening to the network, so I don't know if you guys have been talking about this on other podcasts, but something big is in the works.

Leo: Yeah. We talked about it on TWiT, yeah. Thought it was very interesting. I'm glad you're going to bring it up, yeah.

Steve: And also we've got Ubiquiti routers in trouble again, big trouble. A hopeful and welcome new initiative for the Chrome browser. A piece of errata, a quick SQRL update, and some follow-up thoughts about VPN connectivity. So I think I can promise another great podcast for our listeners.

Leo: Well, that ought to do her, I'll tell you what. Sounds like fun ahead with Steve. And of course our Picture of the Week is good. Steve?

Steve: So we had two pictures this week. The first one I ran across just when doing some research I encountered a notice and a notification from Firefox that we've talked about in the past, but I had never seen it. I went to LinuxForums.org and got, what do they call it? I think they call it a "wall hanger" or something where it hangs down from - I've seen the term. Anyway, it says: "Have an account on this site? More than 200,000 accounts from LinuxForums were compromised in 2018."

Leo: Wow. That's nice.

Steve: Check Firefox - yeah, isn't that nice? So it's a proactive notification that where you're going has had a security compromise. I think I remember when we talked about this, like within the last year. So it's not like forever. But it's like, while it's relevant. And then you're able to click on "Check Firefox Monitor," provide your email address, and I think it uses Troy Hunt's, I'm sure now that I remember, it uses Troy Hunt's Have I Been Pwned site. Troy provides an API that allows facilities like this to query his backend database. So it checks for you, given the email address, to see whether your address is on the Have I Been Pwned database and, if so, warns you. So just a very nice sort of closing-the-loop, proactive, hey, you know, in case you haven't been listening to Security Now! or you don't have an eidetic memory, we'll help you by saying, hey, remember when we talked about Linux forums being pwned? Well, maybe you should check your email address. So very cool.

Leo: Actually, I just wanted to mention that there is now a Chrome extension to do exactly the same thing. Did you know that this is Safer Internet Day?

Steve: Oh. Well, what a happy coincidence.

Leo: Yeah, Happy Safer Internet Day.

Steve: February 5th.

Leo: Yeah. So Chrome has added a password checkup extension. You have to install it, which I did immediately. And it will say, as mine does right now, none of the recent passwords you've entered were detected in a data breach. So it's a little bit different. It's not precise. It's actually looking at...

Steve: Ah, nice.

Leo: And they make a big point about how they're using technology that they developed with Stanford University with the help of cryptography researchers to keep your privacy safe, you know, that passwords aren't getting sent to Google, that kind of thing. So I don't think they're using Have I Been Pwned, it sounds like. They probably have their own database.

Steve: And is it a Google extension from...

Leo: Yes, Oh, yeah, yeah, it's from Google. Yeah, yeah.

Steve: Oh, interesting, yeah.

Leo: So you can do this in Google, as well.

Steve: Well, and of course what's cool is that Google, if you have your Chrome browser saving your sites' passwords for you, then it's got a local database of the password. And sometimes your username will be your email address and so forth. So it would be able...

Leo: Now, I don't do that, of course, because I use LastPass.

Steve: Right.

Leo: But apparently it's still monitoring as you enter them, or as they get entered.

Steve: Yeah, very nice.

Leo: And they do cross-account protection and stuff. So it's kind of cool, very cool, yeah.

Steve: Well, and so the second picture that we have - I gave the first picture the title "Firefox Warned Me." And so I titled the second one, "And So Did Chrome." Although this was a different one, which is our first story.

Leo: I like this, though.

Steve: Yes. Yes, yes, yes. This shows that paypai.com has been entered into the URL bar of Chrome. And there's a dropdown saying, "Did you mean to go to http://paypal.com?" I'm not sure why it didn't do https://.

Leo: Yeah, that's interesting.

Steve: That's interesting. But so...

Leo: And it's a link, so you could click it, and you could say, oh, whoops, yes.

Steve: Yup. That was a typo. I meant - exactly. So you click, and you go to the right place. Okay. So a bit of terminology first. Typo squatting, which is what this practice is of bad guys registering lookalike or typo, like P-A-Y-P-A-I dot com, and hoping that some percentage of people are going to type "I" rather than "L" and go there. And so what they'll do is they'll set up a fake-looking PayPal site and say, oh, you haven't used this machine, or you haven't registered before on this machine or, you know, one way or the other they will spoof you into getting you to give them your PayPal credentials, believing that that's where you are. And then you're in trouble.

So this so-called "typo squatting" is formerly known as an "IDN homograph attack," IDN being International Domain Name. But typo squatting is a lot catchier. Similarly, although I think that URL spell checking is clear, Google calls their forthcoming technology "navigation suggestions for lookalike URLs." I don't think that has a catchy abbreviation, NSFLU.

Leo: NSFLU.

Steve: NSFLU, yeah. So it's under active experimentation with the Canary release, which is at 70 now, in Chrome. And if all goes as planned, it will be appearing in mainstream release before long. You know, they like to sort of roll these things out incrementally, which is a good thing. Actually, we're going to be talking about Firefox here in a minute where they did something that they followed out that caused a whole bunch of unexpected problems. So it makes sense to have, as Firefox does, a nightly build channel.

And in the case of Chrome they have their Canary release, which precedes the rollout to the public by some length of time, depending. So if you're on the Canary circuit, you can enable this feature if you want to play with it. Under chrome://flags, it's enable-lookalike-url-navigation-suggestions. And of course you, I think - I went there, and it just said "default," which is a little unclear. It'd be nice if it said whether that meant yes or no. But anyway, you could set it to "enable," and then maybe you would see some of that.

Oh, and I also noted that the flag - because I'm not on the Canary circuit, but mine had that. So I'm in the just-released stable Chrome 72. It had the flag, but it doesn't appear to be wired up yet. So if you are on the stable release, and you do go there, and you turn it on, and then you experimentally type in "paypai," you know, P-A-I dot com, and nothing happens, it's like, well, yeah. Apparently the flag is there, but it's a do nothing. It hasn't been connected yet.

So we'll get there. And I think that's great. I mean, we're beginning to see a number of these reactions to longstanding problems where the browser developers are saying, you know, we just can't bury our heads in the sand here. Users are being tricked and hurt by clever bad guys leveraging common behaviors, which technically is not our, the browser's, responsibility. If someone says I want P-A-Y-P-A-I dot com, darn it, well, we should take them there. It's like, well, yes, but let's just make sure they didn't mean the much more common PayPal. And the beauty of the position that of course Chrome is in is they know exactly how many people go to PayPal.com versus PayPai.com. And they can say, oh, you know, the chances are very good that this person actually meant PayPal. So I think this is a fabulous enhancement.

Leo: I feel like I've seen that before, but maybe I haven't.

Steve: Well, and the other thing, too, is that I guess people still type in URLs manually, but that's sort of fallen by the wayside, too. I mean, aren't we just mostly clicking on stuff?

Leo: Yeah, yeah, yeah.

Steve: You know, it's like, when was the last time you had to, like, I don't even trust myself to enter a URL. Besides, they're not friendly anymore. They've got, like, GUIDs build into them, where it's 326957.3-4729 - it's like, okay, I can't enter that. Just give me a link to click. In fact, we may see ultimately the URL become sort of more obscured. And it's like, well, do you really want to enter this by hand? Because, you know, that's only for sophisticated users. It's like, okay.

So speaking of sophisticated users, it is definitely time to, as I said, catch up on your patch up for Linux. We talked previously about a not yet really made clear vulnerability in a module of a number of desktop Linuxes called "systemd." It's not in the core base Linux, but it's added to a number of the desktop environments in order to help sort of create the desktop, get all kinds of different widgets and things loaded and get the system started up. A number of the more popular Linux desktop distros have adopted this.

Well, just last week a company called Capsule8, they're a U.S. cybersecurity company, published a working proof of concept which unfortunately weaponized two of the three vulnerabilities which were reported late last year by Qualys. And we talked about it at the time. There wasn't much information, and that was on purpose because we didn't want this to happen, what just has. On the other hand, it's been a couple months, and hopefully people are keeping their Linux distros current. If not, as I said, now would be a good time to catch up your patch up.

On the 9th, Qualys posted the following detailed report. Their report was titled "System Down: A systemd-journald exploit." And they said: "We discovered three vulnerabilities in systemd-journald." So this is a logging component of the system daemon module. There are three CVEs that have been assigned for 2018 ending in 16864, 16865, and 16866. The first two are memory corruptions where the attacker can control allocs, and the third one is an information leak which results from an out-of-bounds read. So in researching it, thanks to the audit trail that we have on Linux modules, Qualys was able to see that the first of these, the 16864, was originally introduced in April of 2013 in the systemd v203. And it became exploitable in February of 2016 in systemd v230, or build, I guess, v230.

They said: "We developed a proof of concept for 16864 that gains EIP" - that's the Extended IP, or the Extended Instruction Pointer - "control on a 386 machine," on an x86 architecture. The second one, 685, "was introduced in December of 2011 with systemd v38 and became exploitable in April of 2013 with v201." Then the final, the third one, which is 16866, "was introduced in June of 2015 with systemd v221 and was inadvertently fixed in August of 2018."

So Qualys says: "We developed an exploit for 16865 and 16866 that obtains a local root shell in 10 minutes on an x86 architecture and 70 minutes on a 64-bit machine on average." They said: "We will publish our exploit in the near future." So they said: "To the best of our knowledge, all systemd-based Linux distributions are vulnerable, but SUSE Linux Enterprise 15, openSUSE Leap 15.0, and Fedora 28 and 29 are not exploitable because their user space is compiled with GCC's -fstack-clash-protection." Okay. So systemd is used by Arch Linux, CentOS, CoreOS, Debian. Is it Mageia? Yeah, looks like Mageia.

Leo: Mangia mangia. No, I don't know the word, no.

Steve: Mangia, yeah, no. Okay. Good. Yeah, it's spelled M-A-G-E-I-A. So maybe it's a not...

Leo: Mageia.

Steve: Mageia, yeah. Mint, Red Hat Enterprise, Solus, and Ubuntu. So of course the big ones there are Debian and Ubuntu. And I did have Fedora in my list, but at least 28 and 29 are not exploitable. So the point is this is a module which, if it has not been recently patched, is subject to a local vulnerability. It's worth noting that this is almost certainly only of local concern.

The 18685 is vulnerability triggered. I'm going to go into this a little bit because it's interesting how we fit these together. This is another example of individual vulnerabilities not being a great cause of concern. But then you mix them, you combine them in order to get what you're - the hacker does in order to get what they're looking for. So 16865 is a vulnerability triggered by code in the systemd's logging software - that's the journald - that allocates temporary memory to contain a log entry without first checking that the request is of sensible size. This means that you're able to allocate, basically ask it to log something that's megabytes in size.

And in fact that's how they found it. They were actually - Qualys was actually doing some research in something else and had some reason to actually produce a log entry of that size, like a dump of some sort, and it crashed the system. And they said, "Whoops, what crashed?" and that's of course where these things all begin. So what this allows for is ready code execution. But that's thwarted by the presence of ASLR, Address Space Layout Randomization, which of course as we know prevents you or dramatically improves or increases the difficulty of exploiting because it's randomizing where things are located in memory.

Fortunately or not, the second bug, 16866, allows specially formatted text sent to the system log to cause the same systemd to write out a message containing data from parts of memory that the user is not supposed to see. So this is a classic information disclosure vulnerability which again would, okay, if it's something running in your own machine, is that really a big deal? Well, that allows the attacker the information required to then bypass Address Space Layout Randomization and exploit the previous vulnerability to run code of their choosing to accomplish whatever malignant goal they may have. So it's not clear that a remote attacker would be able to cause the log to be written or essentially to exploit this remotely. But it's certainly the case that something running on your own machine could.

So the idea would be, if this weren't patched, and now what's happened is, last week, full weaponized proof-of-concept code exists. There is also, I'm seeing just sort of in the air this year a growing focus on Linux attacks. Linux has so far and sort of compared to Windows enjoyed relative obscurity in the hacker community. That really does seem to be changing now, maybe because Linux is being adopted on an international scale.

As we were talking about last week, Leo, it's like, well, okay, why is it that Russia and China are using Windows? That just seems nuts. And of course, as we know, they're moving away from that to their own typically Linux-based platforms. So maybe that's why we're beginning to see more of this. But in any event...

Leo: Some would say that one of the flaws is it's become more monolithic. More and more applications or distros are using systemd, which is not, you know, that's a little controversial.

Steve: Yeah.

Leo: I'll just look at my systemd, and it's v232. These are fairly old versions they're talking about. I think. Oh, no, 230 is vulnerable. Okay.

Steve: Yeah, yeah.

Leo: And I'm on, unfortunately, the Debian stable version, and Debian doesn't quickly update these. And so it doesn't look like there's an update available, so that's interesting.

Steve: Well, good, I'm glad you checked. And for our listeners, again, I can't say for sure that somebody outside your system couldn't force logging. I don't think they could obtain the results of the logging, which is what is necessary for the second, for the ASLR bypass. But just if in general you're lazy about, I mean, deliberately lazy, as many of us are now on Windows, like I don't think I want that feature update yet because it hasn't been going so well for people. You know, just if in general you're not in a hurry to change things, under the "If it's not broke, don't fix it" approach, which I certainly understand, being somebody who just recently left XP, then now would be a good time to say, well, let me just check to see if there's something new. Or maybe give it a month. But don't give it forever because this thing could end up biting you.

So Chrome 72, which we were talking about a second ago, that's the just-released stable channel version, is evolving. And I thought that some of the changes they've made are interesting because they involve deprecations and removals of things. So, for example, pages as of Chrome 72, the current Chrome, for me I had to go into Help About, and then that kind of gave it a little kick in the butt, and it updated me from 71 to 72, and then you do a relaunch, and there you are. Pages, and this was a good thing, may no longer use the window.open to open a new page during a page unload, thank goodness. And of course we know that there are misbehaving sites where you leave a page, and it springs up another page to say, wait a minute. Like bad sites do that.

So they've just decided, okay. The Chrome pop-up blocker already prohibited this, but it's now prohibited whether or not the pop-up blocker is enabled. So yay for a usability improvement because, you know, if you're leaving somewhere, you're closing a page, just let me go. I don't want one last "please don't go" or whatever nonsense. Also in this 72, HTTP-based public key pinning - which is different than HSTS. That's header-based public key pinning. This is, well, okay, that's - I'm getting myself confused here, and I don't want to confuse people.

HTTP-based public key pinning is HPKP, which was intended to allow websites to send an HTTP header that pins one or more of the public keys present in the site's certificate chain. But nobody ever felt comfortable with that because what that would mean would be that, if you failed to anticipate an upcoming certificate change, you would have been proactively saying "Only trust this certificate that we're sending you." And we did talk about this at the time, I remember. So what using that meant was you would have to be absolutely sure that you were replacing your certificate with a new one. And then the problem would be, what if someone hadn't visited your site for a long time and only had a certificate that had then expired? You would be providing them with a newer updated certificate which your earlier certificate and page delivery had said not to trust.

So the point of this is, because this was never really very well thought through, it never achieved much adoption. So, I mean, this is one of the benefits of the kind of telemetry that Chrome is obtaining from its users, is it's able, you know, they're able to look at this and go, you know, this only ever got like not quite 2%, which actually is the number that I remember seeing about this. So we're going to just get rid of it because no one's using it, no one is going to use it, and it just, you know, this was something we can get rid of. So it's gone in 72. And notice that this was something where the removal of it wasn't anything that anyone depended upon. It was a belt-and-suspenders sort of thing. So, okay, we're going to just trust the suspenders, and we'll go beltless. And there are better solutions for doing this, too.

Also, we've talked about the fact that, over the long term, FTP is finally going to disappear from our web browsers. I mean, back in the day with Netscape Navigator you did sometimes, it was sort of convenient that you could use your web browser to show you the contents of an FTP server and click on links and navigate around. It turns out that, up until now, a web page could use an FTP link to render an image. You could have an image tag that wasn't https://, it was ftp. And it was like, what? Who would use an FTP link to render images on a page? But apparently, well, maybe nobody, but the point was the feature has always been in our browsers until now.

So Chrome is beginning the process we discussed recently of deprecating FTP. You'll still be able to use it to browse a directory and click on links to download things manually. But the browser page will no longer use FTP to pull its own content, like images and so forth. If that breaks anybody's page, well, okay. It's time to move your content. Well, first of all, it's nonsecured. It's nonsecurable. There is SFTP and FTPS which are secure versions, but that's not what we're talking about here. So just in general it's going away; and, you know, fine. I'm all for something like this that is this old and is not being really used in our current ecosystem to be removed.

And speaking of deprecation, TLS 1.0 and 1.1 are not long for the world. 1.0 is even older than this podcast, believe it or not, Leo, nearly 20 years old.

Leo: Nothing's older than this podcast.

Steve: And because 1.0 and 1.1 can make use of MD5, okay, and SHA-1, both which are no longer considered sufficiently strong, it's really time to retire these. They also are both able to use RC4 and CBC ciphers. Well, CBC being an encryption mode. As we know, RC4, I loved it for its simplicity and that it was as strong as it was. But it suffered from implementation weaknesses because its pseudorandom stream generator needed more warm-up time than it was being given in practice, so people said, okay, we don't trust this just because. And the CBC, cipher block chaining, its construction is flawed, which made it vulnerable to some attack.

So anyway, the point is there were modes of 1.0 and 1.1 TLS that were just getting old and considered vulnerable and time to move away. So they are further deprecated in Chrome 72. And I looked, trying to figure out what exactly that meant in this case. I mean, we know in the case of, as I was just saying, FTP, pages would no longer render FTP-provided assets. Maybe it shows something on the URL bar that's like, stop doing this. I don't know. Because it's not actually being killed completely until Chrome 81, which is due about a year from now, in early 2020. So they're doing the right thing. They're giving people time to move away.

So presumably, if you go to a 1.0 or 1.1 site under Chrome 72, something will happen. I don't know what. Apparently it still works. Maybe it slaps you or something. I don't know. Maybe one of our listeners will find a server. Even GRC is at TLS 1.2 now. So you can't test it with me. It'd be interesting to know what happens with this deprecation of 1.0 and 1.1. Presumably it's something that the user sees that causes them to be worried, that then puts pressure on the site to update their servers in the next year sometime. So, good.

And part of this was Chrome's deprecation policy that I thought was interesting. They said: "To keep the platform healthy, we sometimes remove APIs from the web platform which have run their course." And they said: "There can be many reasons why we would remove an API, such as: They are superseded by newer APIs. They are updated to reflect changes to specifications to bring alignment and consistency with other browsers. They are early experiments that never came to fruition in other browsers and thus can increase the burden of support for web developers."

They said: "Some of these changes will have an effect on a very small number of sites. To mitigate issues ahead of time, we try to give developers advance notice so that they can make the required changes to keep their sites running. Chrome currently has a process for deprecations and removals of APIs, essentially announcing on the blink-dev mailing list. Set warnings and give time scales in the Chrome DevTools Console" - ooh, I'll bet that's where the TLS 1.0 and 1.1 deprecation stuff shows, it probably is a dev console warning - "when usage is detected on the page." And then: "Wait, monitor, and then remove the feature as usage drops."

So, bravo. I'm glad that the number one browser in the industry is making these moves and essentially creating some coverage for other browsers that want to follow along and also keep things clean.

Facebook, Leo.

Leo: Say no more. Say no more.

Steve: Oh, boy. They got in trouble again. In fact, maybe you guys talked about it because it was relative to iOS things in this case.

Leo: Oh, yeah. We talked about it. We talked about it on TWiT. We talked about it on MacBreak Weekly. We talked about it.

Steve: So I'm not of the camp that believes this is particularly sinister. I know a lot of people run around and lawsuits are generated and so forth. I just think it's ungoverned and irresponsible.

Leo: That's all. That's all.

Steve: I suspect - yeah.

Leo: They're like teenagers.

Steve: Exactly. I suspect it's what happens under conditions of explosive growth, which they have had historically, and lack of adult supervision. When you tell a horde of recently degreed 20-something coders to just do stuff and we'll keep whatever works, this is what you get is a lot of experiments that are sometimes not wise in retrospect. So in today's installment of "What has Facebook wrought now..."

Leo: Or just, "Now what?"

Steve: Now what? We have TechCrunch's report. Of course it was covered by a lot of people, and TechCrunch headlined theirs: "Facebook Pays Teens to Install VPN That Spies on Them." And so the short version of this is it has been discovered that Facebook was paying as much as - and I don't know what set the price, maybe how active they were or how busy they were or where they were going; but, I mean, 20 bucks a month is not nothing - to install this app, which as part of its installation requires you to install Facebook's root certificate.

So as we know, what this allows is a full man-in-the-middle interception of all the traffic that your device, typically your smartphone, is transacting, allowing full visibility into everything that you're doing for monitoring purposes, in return for which you're apparently being paid somehow. And maybe you - I'm sure probably you have more details from the consumer end.

Leo: Yeah, gift cards. They were sending them gift cards, yeah.

Steve: Ah, okay.

Leo: Twenty bucks a month.

Steve: So ages 13 to 35, apparently, in this coverage, since 2013. And again, the concern was, okay, wait a minute. This is not okay. It is a privacy breach. It's a root certificate installed on the phone. Oh, and also, and this was the thing that upset Apple so much, is that Facebook had this enterprise, was participating in this enterprise developer program, which is a means for allowing a company to develop iOS apps for its own internal use that allows them to be signed, and thus honored by the enterprise's employees, but not the public at large. Public at large is supposed to go through the iTunes store in order to download these things.

Well, Facebook was doing an end run around the iTunes store and misusing its enterprise developer program in order to make these apps available to end users without going through iTunes and the iTunes app store, and thus subject to Apple's scrutiny and verification. So Apple's not happy. Facebook said that they had removed this from iTunes availability. Then it turns out that Apple had actually booted them from the store. It's a mess. And if you have anything more to add, Leo, I'm all ears.

Leo: No. Just they're still offering, both Google and Facebook still offer their snooper apps on Android, in case you want to run them.

Steve: Oh, that's right, I forgot. Still there. Still there, available for use on Android.

Leo: Well, it's not against the rules on Android.

Steve: Yeah. TechCrunch reported they asked a German researcher, Will Strafach, and he was quoted saying: "The fairly technical-sounding 'Install Our Root Certificate' step is appalling," he said. "This hands Facebook continuous access to the most sensitive data about you, and most users are going to be unable to reasonably consent to this, regardless of any agreement they sign..."

Leo: Yeah, that's the issue is do you know what you're getting? Yeah. And teenagers - so it was aimed, the Facebook product was aimed at 18 to 35 year olds, 5% of whom were teenagers. And Facebook says, "But we got a signed consent form from their parents." But that's the point is do the teenagers or even their parents really understand what they're agreeing to? Probably not.

Steve: Right, right.

Leo: They're just, you know, Facebook says something like, you know, we'd like to watch what you do online so we can make our product better.

Steve: Yeah, exactly.

Leo: How bad could that be? And I don't think Facebook's nefarious. I mean, that's what Strafach was saying, you know, is that, well, you shouldn't give anybody root access. But, I mean, honestly, I don't think they're using it for a man-in-the-middle attack. Well, they are, literally. But...

Steve: Well, in fact, that's why I was careful to say "man-in-the-middle interception" because...

Leo: Yeah, there you go, not "attack." Yes, yes, yes.

Steve: Right.

Leo: That's right, yeah. And they say, "We told everybody. We told them we were going to do that."

Steve: You know? Well, and in fact back when I discovered that Ad-Aware stuff in my machine and created OptOut, arguably the industry's first spyware removal tool, the uproar was unbelievable. And the guys at Ad-Aware said, wait a minute, you know? It's in the fine print. That's like, that doesn't, you know, sorry, but you know nobody reads that. You can't.

Leo: Right. Yeah. Yeah, it's a little disingenuous to say, oh everybody knows what we were doing. We said that we were doing it. It's okay. You chose it.

Steve: In a few weeks the Japanese government is going to try its hand at the practice that's come to be known as "credential stuffing" across its own country's massive install base of IoT devices.

Leo: Wow.

Steve: And you've got to know that in Japan they're IoT happy. Anything it can find publicly from enterprise network level down to end-user routers whose owners never change their default passwords. NHK is Japan's national public broadcasting organization, which reported that the government had approved this first-of-its-kind venture several weeks ago. So mid-February, staffers from the National Institute of Information and Communications Technology, NICT, will take many previously successful usernames and passwords and use them to attempt, like official use them to attempt to break into as many as 200 million randomly selected, publicly accessible IoT devices located across Japan - routers, webcams, DVRs, IoT...

Leo: That's, like, twice the population of Japan, by the way. It's two per man, woman, and child.

Steve: Well, when you consider that electric toothbrushes are online in Japan, everything. So then the owners of the breached devices will be told by the Japanese government to please fix their devices.

Leo: Please.

Steve: Bolster their cybersecurity. Okay. We'll see how well that stage goes. So the intent behind this interesting white hat move is to reduce the viable attack surface that's currently available to attackers prior to next summer's approaching Tokyo Olympics and Paralympics, which occurs in 2020, so summer after next. Sophos noted that some systems did go down around the time of the opening ceremony for the Winter Olympics in Pyeongchang, South Korea last year. So sort of worth being maybe a little preemptive. So although the immediate goal is to tighten up Japanese citizens' Internet-facing security before the Olympics, the result will almost certainly be improved security overall, I mean, given everything we know about what happens when you scan the public Internet. You find stuff.

So the NICT has reported that IoT devices were at the heart of a large number, more than half, 54%, of the cyberattacks that it detected in 2017. And of course that number's only growing. So I've talked about several times, way back in the early days of Code Red and Nimda, that I participated in some discussions with the U.S. federal government. I was on a conference call that was particularly memorable with the head of the DOJ. And a bunch of us security types were begging to be allowed to write a sanitizing worm that would find the vulnerable devices and fix them for their owners. It would have been easy for us to do. But we were told in no uncertain terms at the time that doing so would be illegal and would open us to the full prosecution and wrath of the U.S. federal government. So it was like, uh, okay. We'd like to help but we don't want to be put in prison. So you know, Leo, since then a lot has changed, and things seem a lot less black-and-white than they were then.

Leo: Mm-hmm.

Steve: We now have, you know, we're post-Snowden and post-WikiLeaks. We've got leaks about the CIA and apparently about the NSA, which suggests that no one here is a Boy Scout. And it's increasingly seeming that, because of the size of this threat, that tying the hands of white hat hackers while the black hats are allowed to run free and to cripple our cyberinfrastructure is seeming less and less correct in practice.

I don't think we've talked about it, but in the news recently has been the observation that apparently foreign actors are able to cripple our electric grid. And there was something else. Some other major aspect of our cyberinfrastructure has been reported in the news. And it's just like, okay, are we actually completely unable to do anything with the fact that we are hosting this kind of potential trouble? And it sort of seems counterintuitive. So if this Japanese experiment bears fruit, and you can bet that all governments are watching closely to see how it goes, it seems to me it could go a long way toward opening some doors and maybe changing some minds about the feasibility of being proactive.

And it could be done incrementally, also. For example, individual ISPs could be chartered with the ability to inspect the publicly available ports of their own customers, whom they know because they have a customer relationship with them, and then to deal with them one on one to resolve any vulnerabilities that are detected. And that of course would create, if that were allowed, that would create a market for a big hardware security vendor to produce carrier-grade network vulnerability scanning, which would scan into an ISP's address space, looking for and logging and then allowing vulnerabilities to be handled. It just sort of seems like we're going to end up there sooner or later. And sooner's better than later.

Leo: Yeah, yeah. Wow.

Steve: Yeah. Firefox 65 is where we are now, recently. And I've not had any problems because I'm not using Avast, AVG, Bitdefender, ESET, or Kaspersky. But it turns out people who were, were having problems.

Leo: Oh, really.

Steve: Yeah.

Leo: That's pretty much everybody who uses antiviruses.

Steve: Yeah. Well, in order to maintain their relevance, all of those AV tools had begun the practice of installing a root certificate of their own in the user's machine. Because, after all, the web has gone HTTPS. And if they're not able to scan into the files that are being downloaded, the content that is being brought into browsers, how do they maintain relevance?

So what happened was Firefox 65 began to roll out, and people began getting notices that their connections were not reliable or the sites they were visiting could not be trusted. It turns out that Mozilla was implementing a man-in-the-middle attack detector. And the Mozilla people also - and apparently none of their early testers were using Avast or AVG or Bitdefender or ESET or Kaspersky, so they didn't have any problems. But as soon as it began going out to a wider audience, this began to happen.

And what's interesting is that we've long known, we've talked about this on the podcast, that Mozilla, due to its sort of odd origination with Netscape and their own security suite, Firefox, for example, on Windows, does not use the Windows built-in security API. They bring their own Netscape security package with them, and it contains their own root store. So it turned out that the certificates that were being stuck into the Windows root store were not also being trusted by Firefox that has its own root store. And so there was this disconnect present which Firefox 65 then began to detect. The moment these alarm bells began to sound, as 65 was rolling out, Mozilla halted the release of 65. Also, AVG and Avast both issued patches in order to back off of their scanning of Firefox so that they would no longer be filtering that and triggering these events.

So basically everyone scurried around and stepped away from what was essentially a collision of sort of best attempts and good intentions in order to solve this problem. So what I learned, which I never knew before and was an interesting tip, was it's possible to ask Firefox to use the Windows trust store. If you go to, in Firefox, the standard "about:config," as we know, when you put in "about:config" you get this massive list of an incredible number of settings. But so then you need to use the search in order to whittle this down. If you just put in the phrase "enterprise," like the starship, what you will find is one item: security.enterprise_roots.enabled. That will normally be set to its default of false. If you double-click on it, that flips it over to true, and that causes Firefox to then use the built-in Windows security root, rather than the one that it comes prepackaged with. And then AV tools, which are installing root certs in the Windows store, will also be trusted by Firefox and solve this problem.

So on the 'Net the coverage of this has had people disabling, web scanning, uninstalling their AVs and so forth. But I know that people who are using a third-party AV are doing so because they want to. And people who are using Firefox are doing so because they want to. Switching over to the Windows root store allows you to keep everything on and running and not have to back off of the AV that you want to use, still be able to filter HTTPS and not have any spurious warnings. And Firefox has said that they're going to be continuing to move in this direction. I don't know yet what they're planning for 66. This is 65 that I've just been talking about. 66 says that there will be some sort of MITM, some sort of man-in-the-middle warning. They are wanting to alert people when there are non-authentic certificates being used.

And here's the problem is that malware does this. There is malware which, as part of its installation, is installing its own root into the Windows store in order to intercept and there do bad things with your traffic. And the problem is that AV wants to do exactly the same thing for a benign purpose. So I don't know how you solve this problem. The other thing is you can't pre-warn Firefox, for example, with the formal certificate for the AV tool because in order to prevent exploitation, the installation of the antivirus has to generate a unique certificate just for that installation.

We've talked about this before. For example, Lenovo has made the mistake of having one certificate which they use globally. Well, that's a problem because the Lenovo system had the private key corresponding to the public key, which they stuck in the root store. So that allowed anybody that got a hold of the private key, and it was easy because it was right there in the Lenovo app that was running on your system, was able to produce certificates that all Lenovo installations would trust. So that's why the only way to do this safely for an AV tool is for them to mint their own certificate pair just for that installation so that only that particular installation of the AV tool has the right to sign certificates for that computer, and it's not globally trusted.

So for that reason, it's not possible to tell Firefox that, oh, don't worry, trust anything that, for example, Avast or AVG does. So this is sort of a collision of - a collision that it's not clear how we're going to solve because I guess the anti - I just can't see how that could work. Maybe the AV tool could install a root, and then it signs a certificate per installation, and Firefox trusts a certificate on the chain. Anyway, I haven't thought it all through. But it does, for the time being, switching to the Windows root store solves the problem. Firefox will be happy. You can use Avast and AVG and the others. And it's not clear how moving forward we solve the problem of trusting some behavior, but not the other, without causing false positives that would scare users. Certainly everybody's heart's in the right place, but there's just sort of a fundamental collision of technology.

Ubiquiti routers are again in trouble. Jim Troutman, who's the co-founder of an Internet exchange point NNENIX, which is the Northern New England Neutral Internet Exchange, he tweeted last Tuesday morning, when we were doing the previous podcast, he said: "Heads up. Ubiquiti network devices are being remotely exploited via port 10001 discovery service." So it's 10001 discovery service. "It results in loss of device management, also being used as a weak UDP DDoS amplification attack."

He wrote: "56 bytes in, 206 bytes out." And he followed that tweet up saying that "Attackers are sending small packets of 56 bytes to port 10001 on Ubiquiti devices, which are reflecting and relaying the packets to a target's IP address amplified to a size of 206 bytes," which gives them an amplification factor of 3.67. So not a huge amplification factor. But still, if nothing else, it allows an attacker to obscure their IP address by spoofing the source IP and bouncing packets off Ubiquiti devices.

And it turns out it's all Ubiquiti devices. I saw the list of the devices that were impacted, and it just looks like this is a standard feature of all Ubiquiti firmware is to be deliberately discoverable with an unauthenticated UDP packet on port 10001. And of course we know that this is a - we've talked about UDP reflection attacks in the past. This discovery service is deliberately designed to be lightweight, so it uses UDP.

And as we know, UDP is lightweight specifically because it does no connection setup. There's no handshaking back and forth with ACK and SYN, SYN/ACK, and ACK packets. It's just a simple UDP query that generates a reply. Well, that means that there's no way to verify the sender's source IP. The destination packet is sent blindly back to the source IP, which allows an attacker who's able to spoof their source IP and send traffic out of their network to bounce traffic off of those devices.

So the guys at Rapid7 picked up on this tweet and did some quick sleuthing to figure out what was going on. So what they found was a little bit daunting. They said that they - so they used a 4-byte packet and suggested that this meant an amplification attack of between 30 to 35 times. But I think they may not have been factoring in the per-packet UDP overhead. So of course you're not actually able to send 4 bytes. You've got to wrap that in a UDP header and then the IP header, which all expands the size so that the effect of amplification is much lower than that because you need to wrap it in those things in order for it to go anywhere.

Anyway, what they found was that their scan of the Internet discovered - get this - 498,624 unique IPs answering with port 10001 on UDP open. Of those, almost all of them, 487,021 unique IPs were confirmed to be answering this discovery protocol. And of those, again, almost all of them, 486,388 unique physical devices based on the MAC address tuples which were returned in this discovery protocol, were found in the responses. So we have just shy of half a million confirmed Ubiquiti devices wide open and ready to reflect and amplify, albeit weakly, like almost by a factor of four, a bandwidth flooding attack.

And also, as we have seen before with Ubiquiti devices, most of them are located in Brazil. There's the NanoStation, which had 172,563; the AirGrid was next largest with 131,575. Then it dropped to the LiteBeam with 43,673, the PowerBeam with 40,092, the NanoBeam with 21,360, and the NanoBridge with 20,440. And then it continued dropping, diminishing across the entire Ubiquiti family. And what was of a little somewhat more concern was that this discovery protocol was actually returning a wealth of information. What Rapid7 found was that a number of devices had, in addition to just being probed, they'd been hacked: 9,146 of them returned the device name HACKED-ROUTER-HELP-SOS-HAD-DUPE-PASSWORD.

Leo: [Laughing]

Steve: Uh-huh. Almost 4,000 said HACKED-ROUTER-HELP-SOS-WAS-MFWORM-INFECTED.

Leo: So who's putting these titles in? The hackers?

Steve: Yeah, yeah.

Leo: Just to be, I mean, doesn't that give them away?

Steve: Or maybe good guys.

Leo: Yeah, maybe like the Japanese government. I don't know.

Steve: Yeah, that one said HELP-SOS-WAS-MFWORM-INFECTED, meaning we disinfected you, 4,000 routers. And now we just thought we'd leave a little breadcrumb behind in case you check your device name and realize, oh, wait. 1628 say HACKED-ROUTER-HELP-SOS-DEFAULT PASSWORD. Whoops. 1,168 HACKED-ROUTER-HELP-SOS-VULN-EDB-39701. So apparently a vulnerability number.

Leo: This sounds like an automated worm doing this; right?

Steve: Yeah, it sounds like maybe somebody did turn something loose and fixed these things. And a little over a thousand said HACKED-ROUTER-HELP-SOS-HAD-DEFAULT-PASSWORD. Slightly different than DEFAULT-PASSWORD. So but all of them start off saying HACKED-ROUTER-HELP-SOS. So probably one person who was going around finding and fixing these things and changing the device name in the process to say, you might want to give this a little attention. So anyway, this led Rapid7 to conclude that the attackers had also identified additional vulnerabilities, although actually we're now thinking. And I think you're right, Leo, these are not hackers. This is a white hat who was fixing these things and leaving some evidence behind.

So if anyone is interested, this would be a good time. Ubiquiti has said that they are looking into updating their firmware with a fix for this.

Leo: Yeah, I'd look into it, too, if I were you.

Steve: Yeah.

Leo: Yeah, let's look into that.

Steve: There is a command line, I've got it in the show notes here; and a link to it, also in the show notes. You can, if you get into your Ubiquiti management console, you use the command "configure" that puts it into configuration mode. So then the prompt changes to having a "#." Then you say "set service ubnt-discover disable." And, like, yes, do that.

Leo: Yeah.

Steve: Then you say "commit," C-O-M-M-I-T, which then writes it into the configuration. And then you want to back out of configure mode. There's no reason for this to be on. You are painting a flag on yourself. You've saying, "Hey," you know, "I got a Ubiquiti router." And a probe of that 10001 port will tell the attacker all about your device. So that's not something you want to be advertising. That's just not good.

So to all listeners, I mean, I have recommended the Ubiquiti EdgeRouter X, which was a stunning little powerhouse for $49 back in the day. I still think it's an amazing device because what was unique about it was that the ports on it, I think it was a five-port device, every one of them was a separate interface, rather than being a switch, which allowed you go configure it and set it up for really good interport isolation to create very well secured domains using a $49 device. I mean, it's an amazing little tool.

Unfortunately, they left this discover service on, on all of them. So the takeaway is, by all means, update your firmware now, and again if they don't have a fix for this yet. But in the meantime, disable this Ubiquiti discover service. You don't want to be sending - oh, in fact, no, shoot, it's not. I was going to say you could use ShieldsUP! to check you, except that that's UDP, and ShieldsUP! is a TCP scanner with the exception of the UPnP service, that port 1900 that I deliberately created for that purpose. And I just don't have time right now to add a UDP test for 10001. So anyway, you know who you are. If you have a Ubiquiti router, I would take some time to turn that service off because you don't want to be easily discoverable.

Leo: And just to reiterate, updating the firmware will not fix at this point because they haven't updated the firmware. So you've got to turn it off.

Steve: Yes, yes, you want to turn it off. Just in general. I mean, you know, we've talked about being stealth. It's just a good thing. And here's something that's not just responding to a ping, it's saying here's all this information about me.

Leo: Here I am. What would you like to know?

Steve: Please, please discover me. It's like, uh, no.

Leo: We discovered you.

Steve: Bad idea. So, okay. Before our final break I want to talk about a very encouraging development on the Chrome side. Google is working on something that they're calling Never-Slow Mode for faster browsing. Bleeping Computer put me onto this interesting experiment one of the Chrome developers is working on. He calls it Never-Slow Mode. And the idea is that if, while a web page is loading, something is taking too long, or something is too large, it will be abandoned and ignored in favor of getting the page loaded. I've got a link to his description in the - it's under Review for the Chromium project. And so I'm going to share what he posted.

He said: "Never-Slow Mode." He says: "Prototype - Do Not Commit," meaning I'm not suggesting that we push this out in any channel right now. But he said it adds the enable-features=NeverSlowMode flag to enforce per-interaction budgets designed to keep the main thread clean. Now, the main thread is that thing, it's the main thread which downloads the content. And he says: "Currently blocks large scripts, sets budgets for certain resource types (scripts, fonts, CSS [style sheets], images), turns off document.write, clobbers sync XHR" - that's the AJAX queries - "enables client-hints pervasively, and buffers resources without Content-Length set."

Now, that means if the resource - normally a resource that you're requesting returns a Content-Length tag, or a Content-Length header so the requestor can immediately see the size of this. So he says it buffers resources without the Content-Length set, meaning it will get them, but it's going to kind of hold them off to the side until it can see how large they are, rather than holding up the page.

He says: "Budgets are reset on interaction," meaning so this is a non-interaction, click to load the page. As soon as you click or tap or scroll or do anything interactive, then the budget's reset. So he says: "Long script tasks greater than 200ms," so that's a fifth of a second, "pause all page execution until the next interaction." So, like, for example, that would instantly stall anything trying to mine cryptocurrency on your page. It would just shut down.

So he says these caps, as in caps, budget caps, do not apply to workers, meaning worker threads; and size caps are lifted for resources loaded from Service Worker Cache Storage, meaning things that are happening asynchronously off on the side, they're allowed to continue. This is just like the main "show me the page," like the experiential thread. So he says current caps, all values are wire, for example, transfer/compressed size. So this is the actual over-the-wire compressed size, not the expanded size. Which is to say the size that you feel because it's taking time to get into your machine. So he has the per-image maximum size set to a megabyte. Or is that a megabit? I don't know. M-I-B, lowercase I? Anyway, good question. I didn't think to wonder before. Could be a megabit or a megabyte.

Leo: It's capital B. I've never seen it written with the M-I like that.

Steve: Yeah, it's kind of odd.

Leo: That must be some sort of European thing.

Steve: Could be. So but what's interesting is the total image size, well, we'll call it a byte, a megabyte. Total image budget, 2 megabytes. So there's both a per and a total. Per-style sheet max size, 100K. Total style sheet budget, 200K. Per-script maximum size 50K. Total script budget 500K. Per-font max size, 100K. Total font budget is 100K. Total connection limit is 10. Long-task limit, as I mentioned, is 200ms.

So then he said, under TODO he said: "<iframe> depth is not yet limited." Oh, that's interesting, so nested iframes. He said: "And font-loading is not yet greedy. Feature-policy header to trigger on per-page basis is not implemented. No UI is implemented to inform users that a page is slow." So, okay. So I'm not suggesting that this is going to be like an immediate win. And I would agree with anybody who thought that it seemed fraught with peril.

But I really do like the underlying incentive behind this because today there is virtually zero pushback against sloppy coding, lazy image sizing, massive code libraries being downloaded just so that a single function can be accessed. I've seen that so many times, and it's just insane. Some sloppy web coder wants a particular function in some library. The library is massive, and it's all downloaded so one function can be called because the guy's too lazy to code something simple themselves. It's just, ugh. Anyway, that's what's happened to the web now. And these guys don't care because they're busy, and no one is making them care. So they just let the page's consumer pay in bandwidth and time.

So the idea that the number one web browser in the world might start actively pushing back and is experimenting with this against needlessly slow page loads, I mean, again, this could be fixed. Images could be given more care, and compressed. Code could be looked at to say, wait a minute, why is it so big? Anyway, I just think - I say bravo to this. I hope that the experiment is - we know that Google cares. Google has done things like this about being a little bit police-y about caring about the page load time and the experience that their users have. So I just wanted to sort of put this on everyone's radar. We'll see where this goes. It would be nice, I think, if it had some future.

Leo: Steven "Tiberius" Gibson.

Steve: If we say that often enough, my Wikipedia page is going to get changed.

Leo: Well, careful what you say there. Go ahead, Steve.

Steve: So Firefox is finally catching up with Chrome, Opera, and Microsoft Edge in a nice way. It turns out that web browser extensions often track their users. They employ tracking scripts like Google Analytics, or redirects through servers that are used to track a user's browsing and search behavior. Since this is not the behavior that users typically want when they are switching into incognito mode, Chrome, Opera, and Edge have already stopped running browser extensions by default when the user switches into their whatever the particular browser calls it, their please don't remember anything I do while I'm in here mode.

The good news is that Firefox will similarly begin blocking extensions. Users will be notified that extensions are not running when they switch into incognito. And they'll have the ability to selectively reenable those, perhaps that they need, like LastPass or SQRL, that they trust and wish to be able to continue using while they're in that mode. This new feature has landed in Firefox's nightly builds. So those on the nightly, if anyone's doing Firefox Nightly, you can see it. And it was expected to move into production builds once any bugs had been worked out. But this one is noncontroversial. It's like, yeah. We should be doing that, too. So Firefox is going to get that.

Speaking of getting that, Linux is getting, the Linux kernel is getting a few more options. It turns out the Spectre and Meltdown mitigations have been more expensive than Intel said. Who would have ever thought that, Leo?

Leo: What did they say, just out of curiosity? Yeah, it cost us a couple of bucks.

Steve: No, no. Remember, no, expensive in terms of performance.

Leo: Oh, oh, yeah.

Steve: Remember, it's like, oh, well, you know, you're going to have to turn off - half of the chip is spent in predicting stuff. But don't worry, turn that off, and it won't hurt you that much.

Leo: Not at all. Not at all.

Steve: Yeah, yeah. Wait, wait, wait. You're saying that you engineered half of the chip just to make this faster, but it doesn't really make it any faster, so you don't mind if we turn it off? Anyway, yeah. ZDNet wrote, they started their coverage saying: "Believe it or not, the mitigations for the Spectre class of CPU vulnerabilities are now some of the biggest enemies of sysadmins." And of course this follows what we've been saying for some time, that not only did Intel understandably significantly minimize the performance hit that disabling these powerful enhancements would cause, but there's never been an attack found in the wild. So I'm going to paraphrase what ZD covered for brevity.

They said: "Despite being security-focused patches, these mitigations are known to introduce huge performance hits to Linux systems. A recent benchmark showed that one of the many Spectre mitigations, the one named Single Thread Indirect Branch Predictors (STIBP), introduced" - get this, Leo - "a 30% performance dip for PHP servers, which caused system administrators to reconsider applying some of these patches. Despite being more than a year old, the Meltdown and Spectre vulnerabilities have remained" - as we know - "a purely theoretical threat. No malware strain or threat actor has ever used" - as far as we know - "any in a real-world attack." It's all theoretical for the last year plus. "Consequently, during the past year, system and network admins have called on the Linux project for options to disable these protections.

"Many argued that the threat is theoretical and could easily be mitigated with proper perimeter defenses in some scenarios. Even Linus Torvalds has called for a slowdown in the deployment of some performance-hitting Spectre mitigations. The Linux kernel team has reacted favorably toward these requests and has been slowly adding controls to disable some of the more problematic mitigations.

"For example, since Linux Kernel 4.15, administrators can disable the kernel's built-in mitigations for the Spectre v2 vulnerability (CVE-2017-5715) with the 'nospectre_v2' kernel command line parameter. Since Linux 4.17, admins have been empowered to disable all mitigations for Spectre v4 (CVE-2018-3639) with the 'nospec_store_bypass_disable' command line parameter. Similarly, a way to disable mitigations for Spectre v1 has been added in the Linux Kernel 4.19, with the addition of the 'nospectre_v1' parameter. These three parameters were added even though the kernel already had existing 'spectre_v2' and 'spec_store_bypass_disable' options for months, which allowed system admins to control the complexity level of the Spectre-class mitigations, which also included an 'off' mode."

But sysadmins wanted some way, of course, those are kernel build things; right? So it's like, wait a minute, are these really off? So sysadmins wanted some way to absolutely, positively guarantee that Spectre mitigations would not somehow kick in at all, ever, no matter what. So now the most recent kernel releases have these three newer parameters which can be used.

"The latest effort to have these mitigations turned off and stay off is the addition of the PR_SPEC_DISABLE_NOEXEC control bit to the Linux kernel. This bit prevents child processes from starting in a state where the protections for Spectre v4" - that's the one that's really expensive in performance - "are still activated, despite being deactivated in the parent process." So experts argue that some processes, and I'm among those who argue - that some processes don't need Spectre protections, and the performance impact far outweighs the security impact, especially in closed systems where malicious code cannot be introduced, such as graphics rendering farms, off-the-grid supercomputers, or other strictly confined systems where no third-party code is ever run.

And I think that's exactly right. As we've noted before here, the only real even theoretically practical threat is shared hosting systems where potentially malicious code might be running in an adjacent virtual machine. Things like Linux-based routers and many other server scenarios don't have that kind of exposure to possibly hostile code. And we've argued, and I do, that even an end user system, if you've got something in your machine which is using Spectre to try to create a breach, then it's dumb because it's already in your machine. So, I mean, there are much easier ways to gain a foothold than something as really unlikely to be feasibly exploited as Spectre. So I think this makes a lot of sense, and I was glad to see this.

I have a piece of errata that I wanted to share relating to last week's Cisco Small Business, the RV320 and 325 Dual Gigabit WAN VPN Routers, where I told everybody, hit pause right now if you have one of these because Cisco, remember, left default logins in this router. A listener, Bryan, wrote: "Just finished SN-699, where Steve tells everyone to pause the podcast, unplug your RV320, update it, then come back when that's done."

Anyway, he says: "The TLDR is, according to Cisco, Vulnerable Products says this vulnerability affects Cisco Small Business RV320 and RV325 Routers running firmware releases 1.4.2.15 through 1.4.2.19. Products confirmed not vulnerable: 'Only products listed in the Vulnerable Products section of this advisory are known to be affected by this vulnerability.'" So he says that's from the security advisory for the command injection vulnerability. The information disclosure vulnerability only affects 1.4.2.15 through 1.4.2.17, so a subset.

He said: "I have family ties to a local business that's running an RV320, so as soon as I got home this evening I contacted them to begin planning for an ASAP update. Once I got logged into their router, saw their firmware version, and read Cisco's security advisory notes, I relaxed a bit. They are running an older firmware version, not one that's affected."

Leo: Oh, that's interesting. It's one of the ones in the middle.

Steve: Yes, exactly. He said: "For what it's worth, firmware version 1.4.2.15 appears to be the first 1.4.x release, which dropped 15-Sept-2017." Okay. So that means that since then things have been vulnerable. And he said: "And version 1.4.2.19 was released 29-Apr-2018. So this has been hanging out there for about 16 months. Not great, but much better than five-plus years."

He says: "That makes sense to me. Steve mentioned on the podcast that these were very popular enterprise and SMB firewalls, but also mentioned that the number of exploitable devices found via scans was somewhere in the neighborhood of 9,000. That number seemed low to me," he wrote, "for such an ostensibly popular firewall. But if, apparently, only the last three firmware releases before the newest, just-released 1.4.2.20 are subject to these attacks, that winnows down the field considerably. For what it's worth, just in case anyone else starting running around with their hair on fire about this one."

So Bryan, thank you very much for the clarification. And I imagine by now any of our listeners who may have been affected probably updated to the latest, and that's probably good, too. So nice to know that if you had not been keeping yourself current for a couple years...

Leo: And really not been keeping - you're good, yeah.

Steve: Yeah, it's probably a little bit better, yeah. But I wanted to share a couple heartwarming, not about SpinRite this time, about SQRL. Oh, first of all, I've been avoiding using the name of our illustrious XenForo assistant coder since I didn't know whether or not he wanted to remain anonymous. But I've been so impressed with his work that I asked Rasmus if he would mind that I let the world know that his name is Rasmus Vind. He has a site, HiveWorkshop.com, which is a XenForo-driven forum. And he's as in love with XenForo as forum software as I am. So it was nice to see because he is a serious web developer who clearly knows his way around PHP, JavaScript, HTML, CSS, and all the contemporary web tools of the day.

And I'm so glad that he happened upon a question that I had posted over in the XenForo developer site back in May when I was just, you know, I had the forums up, and I thought, okay, I cannot have SQRL forums where you can't use SQRL to log in. That's just, you know, I can't have that. So he of course raised his hand and said, "Hey, I know XenForo development. Can I help?" And of course what I did then was, so that he didn't have to implement SQRL on the server side in PHP, I instead created the API which makes adding SQRL support to anything pretty easy. But you still have to know the environment where you are. And since he's a PHP XenForo guy, I said by all mean, be my guest. So Rasmus, an official shout-out and thank you at HiveWorkshop.com. I found a posting - oh, and there it is, yeah.

Leo: And so it has SQRL on it now?

Steve: Not yet. His doesn't.

Leo: Yours does.

Steve: But mine does. And so, for example, Archimedes, someone named Archimedes posted on Thursday at 6:48 p.m., he said: "I've been lurking around the newsgroup for some time now, watching the development of SQRL with great anticipation. Seeing the news today that the forums were SQRL-ready, I figured I'd go ahead and take the plunge. What an incredibly seamless experience." He wrote: "Downloaded Steve's reference client and created an identity in less than five minutes," he said, "and that's only because I read everything slowly."

He said: "I purposefully didn't even install SQRL," meaning you don't have to install it, you can just run it. He said: "And yet I was able to easily get going with the forums in Chrome. Logout and login is a snap. Incredible work, fantastic result. As a professional developer in an American corporate world, I really wish I had more time to help actively. But I'll definitely cheer on from the sidelines. Archimedes."

BrianOfLondon wrote. He posted: "All done. There's probably an orphan account somewhere called brianoflondon_s that you can delete if you want. I've now associated this account with my SQRL ID which I'm using on my MacBook via Wine and on iPhone via Jeff's client." That's Jeff Arthur in the U.K. He said: "Everything is working." Then he said: "WordPress... As soon as that plug-in is ready, boy do I want to try it out on various WordPress sites."

And then Jason L. said: "After I got locked out of my previous account (Jason), due to losing SQRL association and stupidly not creating a password or entering an email address, I decided it would probably be easier just to create a new account. This time I used a password and email address. I then associated this account with my SQRL ID. It works. I have logged in several times using SQRL. I have even been able to use Jeff

Arthur's iOS client to have my computer log into to the site." Meaning using his phone. He said: "I think that is the coolest thing. I can't wait to demonstrate this to my friends and family on their computers. I won't even touch their computers. I will just have them type the URL of these forums. They will think it's some sort of magic trick, or I hacked them, LOL."

He said: "I am assuming once SQRL is stable here and there's no danger of losing associations again, there will be a way to remove password and/or email address since the whole point of SQRL is that websites will not have secrets to keep safe. I realize that sites will probably have email addresses to store, but those aren't really secret, and odds are they have already been stolen anyway." He said: "I know there's a feature in the client to only use SQRL to login, but figured that feature is not functional yet, and I am not ready to do that until I know it is safe to do so."

So anyway, just a note to our patient listeners that we are just about there. What Jason's referring to is that Rasmus and I had a miscommunication. He was using the API in a form where the identity associations were temporary because he wasn't informing the API that he needed to make it permanent. And every night at midnight I have a sweeper that goes through and cleans up any usage of SQRL that didn't result in a permanent use of it on the website where it's being used. And as a consequence, people had initially created their SQRL associations to accounts they already had, and SQRL no longer worked the next day. So anyway, that was just, whoops. It was lack of my clear documentation, actually. So that's now clearly documented, and that's been fixed.

And then of course also people were having fun with the fact that you don't need to tell the forum anything about yourself. You'd have to have a name so that, if you post something, there's a handle there. But not even an email address if you would choose not to have notifications, not to receive notifications from the forum. So anyway, we're getting very close. I have some documentation to catch up on. And then we will get this thing officially released. And the various add-on guys are also working on stuff. Jeff is working on iOS. The Android client is running. And we have, I mentioned, an extension for Chrome. Actually he's developing it under Firefox, and it also runs under Chrome since they have a common extension API. So I am very excited to be actually getting back to SpinRite soon.

Which brings me to a SpinRite question from Patrick McAuley in - and I don't know how to pronounce this, Leo. G-U-E-L-P-H?

Leo: Guelph; right?

Steve: Guelph?

Leo: In Ontario. Guelph, yeah. We have an actual Ontarian in the studio, and he's saying it's Guelph.

Steve: Yes, Guelph, Ontario. So he just said...

Leo: Clark says yup.

Steve: Guelph, Guelph, not a typo. Anyway, simple question. He asks: "Will SpinRite work on an external USB-connected drive?" He said: "Any PC setup drive changes needed?" And the answer is it will work, and no changes are needed. In the early days of USB on motherboards, it was a little questionable because not all BIOSes supported USB drives. Now USB support is in the BIOS. And so the only trick is you want to have the drive connected when you start SpinRite up, meaning when you boot the system, so the BIOS can see the drive and then include it in its list of drives for SpinRite to run on. That's the only requirement. And so long as you do that you are good to go.

And I did see actually a question I referred to at the top of the show about some follow-up on OpenVPN, which actually ties into one of the sponsors of the TWiT network and my comment about why would anybody not run the OpenVPN client for Android, that we were talking about a couple weeks ago. It was two weeks ago. Matt in Providence, and I assume that's Rhode Island, he was looking at what he called a stealth VPN to defeat VPN-blocking WiFi. And so I assume he's talking about WiFi that blocks IPSEC, which is not uncommon; whereas SSL-based VPNs are able to get through that typical blocking. So, and as a matter of fact our sponsor, a sponsor on the TWiT network, NordVPN, offers that, but I'll get there in a second.

So he said: "Please add a show topic for this." He says: "I have a need for VPN obfuscation/stealth VPN when using certain public WiFi networks. In the name of security," he says, "forget the terms of service." Presumably the network says you shouldn't use a VPN on here. He says: "I found a service named TorGuard for around $5 per month." He said: "I'm also interested to know whether you can host your own VPN server on AWS that can act like a stealth VPN, or if running OpenWRT VPN on my home Linksys WRT router can act as a stealth VPN. By the way, I'm a proud licensee of SpinRite, and it fixed a boot issue for me on a computer where I wasn't running it preemptively to prevent problems. Great product. Thanks."

Okay. So I already gave the disclaimer that NordVPN is a sponsor.

Leo: No, they're not, but okay. ExpressVPN is our sponsor.

Steve: You told me two weeks ago they were. Were they?

Leo: ExpressVPN. They might have been, yeah, but not now. ExpressVPN is.

Steve: Okay.

Leo: Don't get confused. It's important.

Steve: Sorry. I'm glad to be straightened out. So we were talking about OpenVPN. Typically these VPN services are using OpenVPN or offer it as an option. So you are able to configure your use of an OpenVPN client for use with whatever service you like. And I got completely thrown off by this confusion of providers. I'm sorry, Leo.

Leo: I believe Express does stealth, as well, just so you know.

Steve: Yes, yeah, exactly. But I did want to mention that the OpenWRT is only an OpenVPN client. As far as I could tell, it will not do OpenVPN serving. So you probably still do need - you need something that can be an OpenVPN server. For example, like pfSense, you could run that, and that would do it. Or just use a service which, as we know, are not very expensive and will allow you to connect wherever you hope to be. So I wanted to provide a little bit of clarification to Matt in Providence for that. And that's our show.

Leo: Well, thank you very much. And for further information, TWiT.tv/sponsors is a list of all the people who pay good money to be on this show. No, and there's nothing wrong with NordVPN. I think at one point they were briefly a sponsor.

Steve: I will check that next time.

Leo: But our current VPN sponsor is the number one VPN, ExpressVPN. And, yeah, that's exactly the point. I mean, if you don't mind appearing to be coming from your house, perfectly fine, which I don't think you do. But if you'd like to appear somewhere else...

Steve: Like in the cloud, baby.

Leo: In the cloud, or somewhere not on prem, or not even in your country, then you might need some help from a friend. Steve Gibson is a good friend. Thank goodness we've got him for 299 more shows, and that's all.

Steve: I think that'll be enough, but okay.

Leo: You think. We'll both be a little older and a little grayer by then. Maybe, what's that, five years?

Steve: But no wiser. I think we've topped out on wisdom, Leo.

Leo: I think wisdom's done. I think it's down - I don't know about you. I can't speak for you. You seem to be getting smarter. I'm definitely going downhill. If you go to GRC.com, that's Steve's site, the Gibson Research Corporation. He doesn't sell giant coffee mugs, although I'm telling you, you'd find a market. You'd find a market for head-sized coffee cups, screen-sized. Geez, Louise. Now I want coffee, inexplicably.

You will find lots of great stuff: SQRL; SpinRite, which is his bread and butter, the world's finest hard drive recovery and maintenance utility; and this show, fueled by caffeine, heavily fueled by caffeine. He has an audio version, and he has really nicely done transcripts. Elaine Farris, our transcriber, writes it all down so you can read along as you listen, which for a lot of people is very helpful. GRC.com. We have audio and video, oddly enough, at our site, TWiT.tv/sn.

Steve: Leo. Yes, sorry.

Leo: No one knows. No one knows. But if you want to see Steve's giant mug, there's no better place to do it.

Steve: The mug or the mug.

Leo: Be a good product, Steve's Giant Mug. All you have to do is go to your favorite client and subscribe, and that way you'll get every copy automatically, the minute it's available, audio or video. Or go to TWiT.tv/sn for Security Now! Steve, have a great week. See you next time on Security Now!.

Steve: Thank you, my friend, for 701.

Leo: Wow.

Steve: Bye.