Attacking Bluetooth Pairing

Description: This week we examine still another new Spectre processor speculation attack. We look at the new "Death Botnet," the security of the U.S. DOD websites, lots of Google Chrome news, pushes by the U.S. Senate toward more security, the emergence and threat of clone websites in other TLDs, more cryptocurrency mining bans, and Google's Titan hardware security dongles. We finish by examining the recently discovered flaw in the Bluetooth protocol which has device manufacturers and OS makers scrambling - but do they really need to?

High quality (64 kbps) mp3 audio file URL: http://media.GRC.com/sn/SN-674.mp3
Quarter size (16 kbps) mp3 audio file URL: http://media.GRC.com/sn/sn-674-lq.mp3

SHOW TEASE: It's time for Security Now! with Steve Gibson. Leo Laporte's out this week; and I, Jason Howell, will be filling his shoes, as best as I can, anyway. Steve's going to talk about a whole number of things. But he's going to dive pretty deep on a newly revealed flaw in the Bluetooth wireless protocol. Is it something to be concerned with or not? Steve's going to tell you all about it, next on Security Now!.

JASON HOWELL: This is Security Now! with Steve Gibson, Episode 674, recorded Tuesday, July 31st, 2018: Attacking Bluetooth Pairing.

It's time for Security Now!, the show where we talk about all the latest security news happening throughout the course of each and every week. I'm a new voice on the show because Leo's normally sitting in this seat. But, you know, I'm not the person you're tuning in to listen to and to watch. It's Steve Gibson - the man, the myth, the legend. How you doing, Steve?

Steve Gibson: Hey, Jason. It's great to be working with you this week. Leo, I guess, is off on a little quick three-day trip. But as I noted to you before we began recording, I saw him also prerecording content for his weekend show, and we just learned from John that he's pretty much gone all of September. So you and I will be working together, I guess at least probably for three of the shows in September.

JASON: I think it's probably three episodes during the time that he's gone. I could be wrong. There may be a fourth in there, but I'm pretty positive it's the three.

Steve: I guess Father Robert is back briefly in August, which starts tomorrow, since this is July 31st. And then he's gone forever, so...

JASON: Yeah, I mean, gone forever, that sounds so ominous. Gone forever from TWiT until who knows what happens down the line, I suppose. But, yeah, that's true. I know

he's back sometime in August, and he's going to swing by. I'm not entirely sure how deep he's going to get when he swings by, but...

Steve: Right.

JASON: Oh, he'll be on TWiT. Okay, so he's going to be appearing on TWiT when he swings by.

Steve: Oh, cool.

JASON: And I think I also saw on last week's episode that you've got kind of the whole SQRL thing happening. There was a plan to do something in tandem with Father Robert at some point; right?

Steve: Well, and I'm planning to use you, now, instead.

JASON: Oh, all right-y.

Steve: I am. Because, yeah. Where I am is I've mentioned to Leo, and we'll be talking about this a little bit later because - in the context of Google's announcement of the Titan security keys, which we'll be covering on the podcast. Where I am is that, because there's no way, you know, GRC has sort of off-the-beaten-path newsgroups, old-school, text-only, NNTP, you know, like Usenet-style newsgroups where, like, everything happens on the development front. And it's a really great place for people who are willing to sacrifice the convenience of a web interface and are willing to set up a specific client for that. I mean, there's a bar you have to get over. But that's not practical for something that, you know, like a technology like SQRL is meant to be, that all of our spouses and friends and family and so forth would be using.

So the point is that I have needed standard, easy-to-use web forums. Those exist. But I built them - I chose a really nice forum package, XenForo, which is in its second major version release. And those are the guys who previously did vBulletin, I think it was. So, I mean, they really know their stuff. But it naturally doesn't support SQRL authentication. It doesn't know about SQRL authentication. So I'm right now in the process of teaching it about SQRL so that - because you have to be able to use SQRL to log into the SQRL forums. That would be crazy.

JASON: I would imagine if you did anything different, yeah, people would be holding your feet to the fire on that one.

Steve: Yeah. So anyway, so that's where I am. And but my intention is to sit down with you, Leo, and I was thinking maybe, god, I'm blanking on his name, the guy who's traveling around eating everything in foreign countries.

JASON: Oh, Mike Elgan.

Steve: Mike, yes. Because I want to sort of do it as a presentation, but also sort of contentious, where like you guys say, okay, well, what about this? What about this? What about this? What about this? Because I think that would be most useful to people who are themselves saying and thinking, well, what about this? What about this? Because the point is I think that the SQRL system has an answer to every possible question. And so, I mean, that's what it has to have. And so that makes it unique among all the identity solutions that exist. And I think that's the proper way to, like, vet it to our audience. So anyway, I don't know when that's going to happen because I never know when anything is going to happen. But as soon as we're ready, we will find a time, and we'll make that happen.

JASON: Absolutely. Count me in. That sounds like a lot of fun. I'd be happy to help.

Steve: So in the meantime, however, we've got Episode 674 for this last day of July, the 31st. There were two topics vying for the title. There is the news of a new attack on Bluetooth which generated a lot of press last week, which we couldn't ignore. And then there's also, believe it or not, another Spectre processor speculation attack, which of course has been the gift that just keeps on giving to this podcast all year.

JASON: Yeah, no kidding. It's like a constant feed of information for you to talk about, basically.

Steve: Yeah. And there's a lot to say about both. And so it was like - and then I just thought, okay. We'll make "Attacking Bluetooth Pairing" the title, only because I'm tired of typing Spectre into the title of the podcast. And I'm sure our listeners are like, oh, not another one. But we've got to talk about that. So we'll talk about Spectre first, look at a new trend in botnets which I'm calling "flash botnets" because they're able to now, by leveraging old, well-known vulnerabilities, they're able to flash into existence in a matter of hours. There's a new one called the Death Botnet.

We've also got some interesting stuff about the security of DOD websites. There's a bunch of Google Chrome news. One particular U.S. Senator, Ron Wyden, has been pushing both the DOD, which we'll talk about, and also just the government in general towards more security. We've got an interesting bit about the emergence of and the threat of cloned websites which could easily slip under people's radar unless they're watching for them. And so I want to put that on everyone's radar. Some more cryptocurrency mining bans. The announcement last week of Google's Titan hardware security dongles.

And then we'll finish by talking about, well, the technology in enough detail of this newly discovered Bluetooth protocol flaw to put it into context for our listeners and figure out whether, even though the OS and device manufacturers are actually all scrambling to update their implementations, whether we as end users really have anything to worry about, then and even now, before they get things updated. So I think another typical great Security Now! podcast.

JASON: Awesome. I can't wait. Looking forward to it. And you mentioned Spectre kind of at the top, so that seems like the great place to start.

Steve: Yes, yes.

JASON: It's always evolving. What is NetSpectre, exactly?

Steve: Okay. So you can imagine from the name; that sort of gives it away. A couple weeks ago we talked about something kind of related. It was a Rowhammer attack. "D" hammer or DRAM hammering or row hammering is something we've also been talking about now for a couple years. It turns out that the dynamic memory in our, well, actually all of our devices - laptops, desktops, and even smartphones.

Android has been a subject of hammering attacks. The DRAM is susceptible to noise, like adjacent row noise, because DRAM memory is organized in rows and columns. And if you read frequently from one row, it turns out that there's a probability - it's low, but not zero - that a bit will flip in the adjacent row. And clever attackers, or actually at this point clever academicians, researchers, have figured out how to leverage that into an attack, to arrange to flip the write permission bit on the memory that governs the memory manager to give their process write access to the kernel, for example, where it should only ever have read access.

So anyway, so what happened a couple weeks ago, we talked about this, is the guys who have been doing all this Rowhammer research at VU Amsterdam, Herbert Bos and his team, have come up with a new attack called Throwhammer, which is over the network. Because it turns out that network adapters have gotten so fast that they've had to be able to do DMA, Direct Memory Access, into their buffer memory because they need super high-bandwidth connectivity. Well, that turns out to be row hammerable, and so thus Throwhammer. So first we had network-based Rowhammer attacks. Now we have network-based Spectre attacks.

It turns out that in this case - shoot. I had it in my notes, but I don't see it here. A pair of researchers figured out how to use the network activity on many different systems. They used desktops and laptops and even cloud-based VM systems to induce Spectre speculation vulnerabilities. What they said in their abstract was - and a little bit of this is repetitive, but again they sum it so nicely.

They said: "Speculative execution is a crucial cornerstone to the performance of modern processors. During speculative execution, the processor may perform operations the program usually would not perform. While the architectural effects and results of such operations are discarded if the speculative execution is aborted, microarchitectural side effects may remain." And this is the whole idea of, like, training a branch predictor for a certain prediction and then causing it to deliberately mispredict. Well, nothing at the architectural level sees that because that's a microarchitectural optimization which all processors have incorporated.

Anyway, they continue: "The recently published Spectre attacks exploit these side effects to read memory contents of other programs. However, Spectre attacks require some form of local code execution on the target system." And in fact, in discussing this on the podcast, it's one of the reasons I've said to our listeners, you know, certainly this is of concern in a cloud environment where multiple VMs might be owned by different organizations. One of them could be untrustworthy, and they're running on shared hardware. Because there you've got code from a third party that essentially is sharing the same processor, and that's where the leakage can occur.

On an individual's, like an end user's machine, well, if you've got bad stuff running in your system in order to perform Spectre manipulation, well, you've already got problems because your system has got malware on it. And, frankly, there are easier ways to break privacy if malware's running on a system probably than relying on Spectre. So it's less of an issue.

Anyway, so they said: "Spectre attacks require some form of local code execution on the target system. Hence, systems where an attacker cannot run any code at all were, until now, thought to be safe." And that's the key is that, in mitigating these attacks, because mitigation costs performance, that is, you have to - we have speculation in our processors because it is a powerful optimization win. So if you have to turn off speculation, you want to turn it off selectively to minimize the performance impact. Thus it's only been turned off so far where it's been clear that it's necessary to turn it off.

What these guys have done by demonstrating that just network activity is able to leak information, that is, believe it or not, the timing of the return packets, when enough of them are looked at over time, allows them to dribble bits, reliably dribble bits at a low rate out of a system to which they have network access. So that says they're not running any code on the system. So this changes the universe of mitigation for Spectre attacks significantly. These guys did report their discovery to Intel in March, and at this point Intel's probably just, like, thinking, when is this nightmare going to end?

So they said: "In this paper we present NetSpectre, a generic remote Spectre Variant 1 attack. For this purpose, we demonstrate the first access-driven remote cache attack

over network, leaking" - and this doesn't sound like much, but we'll talk about this - "15 bits per hour." Again, slow. That's why I called it a "dribble." But the reason this is significant is that it can be going on as long as it needs to because it's over the network. So it can be going on behind someone's back, and how would they know?

They said: "Beyond retrofitting existing attacks to a network scenario, we also demonstrate the first Spectre attack which does not use a cache covert channel." In other words, normally they're looking for whether data is in cache or not, and sensing the delay required to fetch from main memory as opposed to the much faster access from cache. Anyway, they found some instructions in the Intel processor, the AVX instructions, which give them a cache-free Spectre attack and allow them to get substantially more, well, four times, 60, six zero, bits per hour from the target system. So again, this is just them being a little more academic in seeing what they can do.

They said: "We verified that our NetSpectre attacks work in local area networks, as well as between virtual machines in the Google Cloud." And finally they said: "NetSpectre marks a paradigm shift from local attacks to remote attacks, exposing a much wider range and large number of devices to Spectre attacks. Spectre attacks now must also be considered on devices which do not run any potential attacker-controlled code at all." Like can you say IoT?

They say: "We show that, especially in this remote scenario, attacks based on weaker gadgets" - "gadgets" is the term now in the Spectre literature for some code which they arrange to use to leak the information. So "Attacks based on weaker gadgets which do not leak actual data are still very powerful to break address space layout randomization remotely." So that's the other thing they've done. Address Space Layout Randomization, ASLR, is a mitigation we often talk about because the idea is that one of the more powerful attacks is so-called "Return Oriented Programming," where if you are unable to execute your own code, for example, in a buffer overflow, you can still arrange to execute existing code. That is, you jump into the tail of a subroutine which does a few things you need and then returns to you. So it's called "Return Oriented Programming."

And the point is, to do that - oh, and that's like code in the kernel. So as long as you know where code of the various kernel modules are located, you can jump to little prearranged spots in order to have it do little bits of work for you, come back, jump somewhere else, do a little bit more, and end up building a potent exploit by just using little snippets of code that already exists. But you have to know where it is because it's a blind attack. There's no way you can check beforehand. So the mitigation for that has been to scramble around, that is, address space layout randomization, to randomize where things are loaded. Every time a modern OS boots, the modules that form the OS are all randomized in their locations.

So the point is that these guys have also demonstrated that, even if they're unable to find an information leak where they actually read out memory contents, they are still able to derandomize the address space layout randomization remotely. So that could be like the beginning of an attack on a system that is not rebooted frequently since rebooting would rerandomize the address space layout.

Anyway, so that's the nature of what they've done. The popular press in covering this I think got it wrong because the approach that the press took was, well, 15 bits per hour, that's really not so fast. It's like, okay, wait a minute. The fact that it's a slow dribble of information by no means renders this ineffective. So, for example, imagine the scenario where some malware gets on an employee's machine through a phishing attack inside of an enterprise, and we've seen that before.

Then the question is, what mischief can the malware get up to? Well, normally Intranet servers are going to be hardened against their own internal network, just as they are

hardened against the external network. So if something gets onto an Intranet and is then able to use NetSpectre to, overnight, I think I figured it takes 17 hours on a LAN to obtain a 256-bit private key from a server. So that's not that long. I mean, 15 hours...

JASON: It's not inconvenient at all, yeah.

Steve: Yeah, exactly, running behind the backs of a system. If it's then able to determine what somebody's private key, or the private key on a corporate Intranet LAN, then that's powerful. And they did also verify among Google separate instances of a server running in the Google Cloud there is plenty of bandwidth between cloud servers; that they were able to attack another server running on Google Cloud and obtain the same kind of rate of information. So this is something that definitely needs to be looked at.

They said toward the end of their write-up on leakage, and to give our listeners a sense for how much is necessary, they said: "Desktop and Laptop Computers. In contrast to local Spectre attacks, where a single measurement can be sufficient, NetSpectre attacks require a large number of measurements to distinguish bits with sufficient confidence. Even on a local network, around 100,000 measurements are required to reduce the noise to a level where a difference between bits can be clearly seen. By repeating the attack, the noise is reduced, making it easier to distinguish the bits."

They said: "For our local attack, we had a gigabit connection between the victim and the attacker, a typical scenario in local networks, but also for network connections of dedicated servers and virtual servers. We measured a standard deviation of the network latency of 15.6 microseconds." And they talked about breaking ASLR. They said: "To break ASLR [Address Space Layout Randomization], we require the cache covert channel. On average, this allows breaking the randomization remotely within two hours." So that's not bad.

And on a cloud network they said: "We evaluated the performance in the cloud using two virtual machines instances on the Google Cloud. These virtual machines have a fast network connection. We configured the two machines to each use two virtual CPUs, which enabled a 4Gb per second connection. In this setup, we repeat the measurement 20,000,000 times per bit to get an error-free leakage of bytes. On average, leaking one byte takes eight hours for the cache covert channel, and three hours for the AVX covert channel."

They said: "While this is comparatively slow, it shows that remote Spectre attacks are feasible between independent instances in the public cloud. In particular," they said, "APTs [Advanced Persistent Threats] typically run for several weeks or months. Such an extended timeframe is clearly sufficient to leak sensitive data, such as encryption keys or passwords, using the NetSpectre attack in a cloud environment."

And then here's where I did the math that I quoted before. In the notes I said: "A 256-bit secret key on a local network at the rate of four minutes per bit, or 1,024 minutes, is 17 hours." So entirely feasible. However - okay. So what they're doing is they are sending packets to the server and getting a reply. Their only instrumentation, that is, the only thing they're able to sense is the timing of the reply.

So what that does say is this is completely infeasible over the Internet. It is only within a locally contained cloud environment where you've essentially got the equivalent of an Intranet in the cloud among servers, or on a corporate LAN. I mean, so it's got to be point to point between the network interface which is making the requests, and the network interface that is being probed. There is absolutely, I mean, given how many packets were required to obtain a bit of information with a LAN or in a virtual network environment between VMs, there is absolutely no way this could possibly produce

information over the Internet where you've got multiple jumps with routers all buffering packets and introducing huge amounts of delay. I mean, they're looking at a standard deviation of 15, what was it, 15.6 microseconds. And that's the kind of timing they need. So with a standard deviation of 15.6 microseconds, they're still needing hundreds of thousands of packets per bit.

So as we know, on the Internet we readily see timing of, what, 30, 40, 50 milliseconds, and huge variations, so a much larger, I mean, a huge standard deviation. So there's just absolutely no way this is feasible for a remote attacker, for example, to attack someone's SOHO router which has a public face. But this should be a serious concern both in cloud computing scenarios and also on LANs because, as we said, you could spend a day, I mean, who wouldn't? What attacker wouldn't be happy to pound on a LAN-accessible server for 17 hours if it meant that they could get that server's 256-bit elliptic curve private key?

JASON: Absolutely.

Steve: So yet another mess that speculation brings us.

JASON: Seems to me like this is the perfect kind of example of the slow and steady approach; right? Slow and steady wins the race. It doesn't matter how much you're pulling. What matters is that the plan works, even over a long period of time at low amounts of data. Like there's still something to be gained from that. It's very easy to kind of consider it a theoretical sort of thing and unvaluable to an attacker. But obviously there's something to be gained there. And a day's really not that much time in the grand scheme of things.

Steve: No, not if you've got something, like, well, not when you consider the value you're getting in return.

JASON: Absolutely.

Steve: I mean, if you get a server's private keys, you're able to spoof its identity. So, I mean, the private key is the most protected thing that a server has because that's how it's signing all of its outgoing traffic. So, yeah. And I really think that what we're going to need to see, as we've said, is short-term, kind of hold-our-breath - it's important to remember, too, all of this, all of this Spectre stuff is speculative. That is, in terms of it even being - it's theoretical. There's never been an instance found of this being done in the wild. We know that it's possible, and we know that what's possible ends up happening.

So essentially what everyone's been doing is to make this as improbable and as low leverage, that is, generally the Spectre flaws, as possible. Ultimately we're just going to have to learn the lesson and come up with another redesign of chips that recognize that we need to isolate the microarchitecture speculation from the top-level architecture so we can still get the performance that we want from speculation, but that nobody is able to sense that out at the top level. So we get the isolation that we thought we always had, but it turns out that was always an illusion.

JASON: Now, you kind of touched a little bit on IoT in there, and it looks like - it's such an ominous name, Death Botnet. Who's the person that selected this, and why was that not taken already? Because that seems like an obvious one.

Steve: So last week - there's a security researcher with NewSky Security, Ankit, and I always trip over his name. It looks like Anubhav, Ankit Anubhav. And I'll just call him Ankit to...

JASON: Ankit A.

Steve: So that I'm not mangling his last name. So last week he brought us the notion of what I called a "flash botnet" because it had acquired 18,000 Huawei routers within 24 hours, that is, from the moment it started, a day later it had 18,000 routers in its net. And it was using a several-year-old vulnerability that had long since been known, that Huawei had issued a patch for. But these are routers. These are things in people's closets, or they're just sitting there, minding their own business - well, not so much now. But the point is that they had not been updated. And this is the problem that we're really seeing now with routers because it's now become just the play land for hackers because now all hackers know that routers are not being updated, for the most part, with the latest patches. So they just go and look at a two-year-old patch and go, let's find out who's vulnerable.

Well, Ankit found, for this week's update on This Week in Compromised Routers, he found another botnet which its author calls the Death Botnet, who knows why. He fired up a dialogue with the hacker who goes by the handle EliteLands, E-L-I-T-E-L-A-N-D-S. So EliteLands named this new botnet "Death." So far it has not attacked anybody, so it hasn't done anything, although it's going after AVTECH-based devices, which include DVRs, network video recorders, IP cameras and more, once again using a two-year-old and also long-since-patch-available device flaw.

But this one is a little bizarre, believe it or not. It turns out that the AVTECH devices expose their password in the clear. So the password is available in plaintext. So it's easy for someone to scan them, hack into the device to get the password in order to log in. Beyond that, it turns out that, believe it or not, the password process has a flaw such that whatever is used in the password ends up being a command sent to the shell. I mean, it's just hard to believe this. So, for example, if you set the username to "This is my username with a password of reboot," it reboots the device after having performed that. So what Ankit told Bleeping Computer, who had their reporting of this, he says: "If I put 'reboot' as the password, the AVTECH system gets rebooted." And he said: "Of course, the Death Botnet is doing much more than just rebooting."

Ankit says the hacker has been experimenting with different payloads for use in the password field. Meaning that what we're talking about is a shell command, that is, a command, any command executed that you choose in the password field. So Ankit said that he's recently started using these payloads to build a botnet which he refers to as Death. So in the latest version of the payload, Ankit says that EliteLands is adding accounts with a lifespan of only five minutes that execute his payload and then disappear from the infected device.

So Ankit said it's like a burner account because usually people don't set up passwords for access of only five minutes. But in this case that's enough to get the job done. That essentially commandeers the device and gets it to join into the botnet. And then the account which he used is transient, and so it expires off of the device and can no longer be used. So 1,200 AVTECH devices are hijackable in this way, which Ankit determined using a readily available IoT search engine. I don't know if it's Shodan or one of the other ones. But anyway, this was patched more than a year and a half ago. But as we've said, these devices don't actually tend to be patched in the real world. Their users just, you know, it's an IP camera or a DVR or a net video recorder. And so, you know, if it's not broken, don't fix it.

JASON: The person who owns it would have to know, first of all, to jump on there and do some sort of a manual firmware update in order to get that [crosstalk].

Steve: Right, right.

JASON: They're not going to do that. That's the big problem with IoT at this point. No one is going to do that.

Steve: Exactly. And as we've said, what we have to - we have to move to a model where these non-maintained devices are able to update themselves.

JASON: Yes.

Steve: We have to have that. And then it'll just have to be - we'll just have to wait around for the devices which don't know how to update themselves to finally just die off, to age out of the ecosystem and be replaced by solid state network recorders that hopefully will know how to update themselves. But that's what we have to do is have self-updating, self-healing devices. I mean, we already have that with our web browsers. All of our web browsers are now updating themselves. As we know, our OSes are, albeit with a little more user interaction. But Windows 10 has been controversial because it's been so heavy-handed in forcing people, ultimately forcing them to update the OS. So you just don't have a choice at some point, if you're going to be connected to the Internet. So it's going to have to be that our IoT devices take that same responsibility moving forward. And then we just wait for the ones that don't to die off.

JASON: Yeah, because we can't be trusted. That's what we've learned at this point. We cannot be trusted to protect ourselves.

Steve: Yeah, it's really that we're learning that all of these devices, I mean, unfortunately, all of them have vulnerabilities which emerge after they've been shipped, and that users...

JASON: Well after, yeah.

Steve: Yes, and that users just treat them as appliances. It's a plug it in and don't think about it again. Until it breaks, they're happy with it. I mean, that would be the other thing to do would be to build in a deliberate breakage into the device so that it dies after three years, although then you have all kinds of other legal issues.

JASON: Yeah, I think people would be upset about that, too. I mean, but that's just - that's the huge issue with IoT as our homes, you know, I don't think of my home as incredibly smart. I mean, there are smart devices. But if I went into my Eero app and added up all the devices that were connecting, there's tons of them. Even though I don't think that there are. And I'm pretty tuned into technology. If somebody has a decent amount of these devices, and they're just not thinking about updating them, it's so untenable to think I'm going to go over to all my devices every month and make sure that they're updated. So it's got to be on the device side. It's got to happen automatically. I think it's more and more complex.

Steve: Yup. So we have a senator who is often in the news because I have to say he's on top of things. He's Oregon's Senator Ron Wyden. And he's probably a pain in the government's rear end, but it's a pain that we need him to be. Two months ago Ron wrote a letter to the Department of Defense. He addressed it to Dana Deasy, who's the Chief Information Officer, so the CIO, at the U.S. Department of Defense at the Pentagon. The letter reads:

"Dear Mr. Deasy: I write to ask that you take immediate action to require the adoption of" - and get this. I mean, it's hard to believe that this letter had to be written in the first place - "that you take immediate action to require the adoption of cybersecurity best practices on all publicly accessible DOD (Department of Defense) web services.

"In 2015" - right? So three years ago - "the Office of Management and Budget issued memo M-15-13 requiring all federal agencies take steps to secure their websites and other web services, including interfaces for automated programmatic interaction (APIs) from cyberattacks. The OMB memo gave agencies" - okay, so this was in 2015 - "gave agencies until the end of 2016 to enable HTTPS encryption and to enforce its use with HTTP Strict Transport Security (HSTS), which ensures web browsers will not use insecure protocols when connecting to HSTS-enabled websites."

So, okay. In the first place, this is very cool that a senator knows what this is. Maybe his staff; but at least, you know, but we know from Ron in the past that he's savvy to this stuff. He then says: "In 2017," okay, after the OMB letter of 2015 gave federal agencies until the end of 2016 to do this, in 2017 Ron writes: "The Department of Homeland Security issued Binding Operational Directive (BOD) 18-01, reiterating the OMB requirements and requiring civilian agencies to adopt additional forms of basic cyber hygiene."

He says: "A small number of DOD websites, including the Army, Air Force, and National Security Agency" - NSA, right, okay, so we would hope - "currently implement HTTPS by default" - a small number - "and use certificates trusted by major web browsers. Unfortunately, many other sites, including the Navy, Marines, and your own office's website at dodcio.defense.gov, either do not secure connections with encryption" - now, this was two months ago, right, deep into 2018 - "either do not secure connections with encryption or" - get this - "only prove their authenticity using a certificate issued by the DOD Root Certificate Authority" - okay, and then he says, and I won't editorialize, I'll continue reading - which, he says, "many mainstream web browsers do not consider these DOD certificates trustworthy and issue scary security warnings that users are forced to navigate before accessing the website's information. These challenges do not only impact civilians; service members accessing DOD pages from home regularly encounter security warnings and must click through such errors when accessing public DOD resources." So in other words, to the degree that they did implement TLS encryption and authentication through HTTPS, they did so with some random certificate signed by the DOD root that isn't publicly trusted. Unbelievable.

JASON: So then what's the point?

Steve: Yeah.

JASON: I'm very surprised that attention to this - and I'm curious why Ron, like what has motivated Ron Wyden to get so involved with all aspects like this. But surprised that attention hasn't been given to this before. This seems like no-brainer stuff.

Steve: I know. And so what happened was this was prompted by Wyden's awareness of what was going to be happening with Chrome. The letter continues: "The DOD cannot continue these insecure practices. Starting in July" - and we'll be talking about this in a minute because we've been speaking of it before, of course - "the Google Chrome browser" - and I'll editorialize here, noting that it is the majority browser on the Internet - "will begin warning visitors of non-HTTPS sites that the requested site is not secure. These warnings will erode the public's trust in the department and its ability to defend against sophisticated cyber threats." Or your mother. Okay, he didn't say that. "Moreover, the DOD's refusal to" - I love it, "refusal" - "to implement cybersecurity best practices actively degrades the public's security by teaching users to treat critical security warnings as irrelevant."

JASON: It's a really good point.

Steve: "Normalizing these warnings increases the risk of cybercrime and foreign government hacking as users, both military and civilian, incorporate these dangerous practices reinforced by the DOD into their daily habits."

He finishes: "DOD has prided itself on cybersecurity leadership, and now is the time to again demonstrate that leadership. I urge you to direct all DOD agencies and offices to take" - I mean, I was going to interrupt myself to say, you know, everybody else has - "to take the following three concrete steps to improve cybersecurity of their publicly accessible web services. First, adhere to all guidelines specified in that OMB memo M-15-13 and the subsequent DHS Binding Operational Directive 18-01, including enable HTTPS with HSTS on all public web services. Facilitate the adoption of HSTS by delivering a list of all public DOD domains, including .mil addresses, to DHS as required by the DHS BOD. Obtain and deploy certificates" - imagine this - "trusted by major web browsers for all web services accessible to the general public. And evaluate the use of shorter-lived machine-generated certificates such as those available at no cost from organizations like Let's Encrypt."

There I would argue that, if you're going to do this, get EV certs from DigiCert and go with the best. If you're going to bite the bullet, then inspire as much confidence as possible because why not? And he says: "Please provide me an action plan" - so this was two months ago. This was in May he sent this - "by July 20, 2018, describing your progress implementing these steps and detailing an estimated date by which all publicly accessible DOD web services will implement these cybersecurity best practices. If you have any questions regarding this request, please contact Chris Soghoian..."

JASON: Soghoian, I think.

Steve: Soghoian, that's it, "...Chris Soghoian in my office." So it's significant that Chris is there in Ron's office or accessible through Ron, at least. So the good news is they did reply. The DOD did reply last week, so close to if not by the deadline, saying: "We've been working on it for two years, and we plan to have this done by the end of the year." So, okay.

JASON: Celebrate.

Steve: In the meantime, Chrome 68 and subsequent did execute on its intention of flagging non-HTTPS sites as "Not Secure." That is happening as of last week, when 68 emerged from beta and became the version that everyone is using. And for what it's worth, the DOD is not alone. According to Cloudflare's telemetry, more than half of the top one million sites, 542,605 of the top one million sites do not use or do not redirect users to, an HTTPS version of the HTTP URL. In other words, while they may have support for HTTPS, if you enter the site with HTTP, you stay there.

And of course that's a problem. That's not promoting the user to the secure version. It means you go in either way, and you get to remain. So what this means is that, starting with the release of Chrome 68, a large number of users will see, if they happen to HTTP, a "not secure" indicator next to most of the sites they visit after they update to Chrome 68. And I've said before, and I still don't know why it's not happening, that our browsers really should start attempting an HTTPS connection when not otherwise forced to HTTP. So, for example, traditionally, if you just put in www.something.com, the browser defaults to HTTP. They really should start flipping that and defaulting to HTTPS.

JASON: Is there a reason that they don't? I use a Chrome plugin called HTTPS Everywhere, and that kind of forces that behavior. But why don't they do that? Because that seems like - it seems like the right direction to go, more secure down to less secure if more secure doesn't exist.

Steve: Yes, it is. And it's just inertia. It's just that there's a concern that it might break some things. But that's becoming decreasingly probable. And I did some googling just because I was curious what effect the appearance of this not-secure indicator would have. I heard Leo on the weekend, I thought I heard him grumbling about it, like he was annoyed that Chrome had done this because I guess it was upsetting people. Maybe he had some calls on his Tech Guy show over the weekend. I'm not sure what the back story is, and I'll ask him to elaborate next week. But in googling I ran across an instance of somebody posting in the Google forum their upset that foxnews.com was now showing not secure. And that did not make them happy.

So I thought, wow, really? So I went over, and I put in www.fox - or maybe just foxnews.com. And sure enough, it took me to http://www.foxnews.com. And in Chrome, not secure. And then I was curious. So I added an "S," HTTPS, and that worked fine, too. And now Chrome was not saying that it was not secure. It wasn't super happy, I mean, it wasn't an EV cert, so it didn't light up green. But I checked the certificate at https://www.foxnews.com, and it was a perfectly valid certificate issued by DigiCert.

And then what was interesting was they had a little "i" to the left of the URL for information. So I clicked on that. And even though it was HTTPS, that is, in the HTTPS version, everything was okay, I got a red warning saying "Your connection to this site is not fully secure." And then in black underneath it said, "You should not enter any sensitive information on this site, for example, passwords or credit cards, because it could be stolen by attackers." So I thought, oh, what? So again, dug in some more, and I found some dialogue about this from people that might have been in the same thread because there was some back-and-forth grumbliness about Fox News showing as not secure.

And Google finally stepped in and said, "Hi, all. The reason why https://www.foxnews.com is getting marked 'not fully secure' is because it includes a search form that sends data to http://www.foxnews.com." And he says: "Note the insecure HTTP. We currently count that as not fully secure and downgrade the security indicator icon to the 'i' symbol rather than the lock. A secure HTTPS site should load all resources over HTTPS and have all forms submit to HTTPS. If you have any questions about this policy or our current implementation of it, feel free to follow up in this thread."

And what's interesting is that GRC has been historically accessible both ways. I did quite a while ago, years ago, switched over to HSTS and preloaded GRC.com and www.GRC.com in Chrome as a known HTTPS-only site so that that would be the only way that Chrome users could connect. And that list has been propagated since. But you can, in a site's URLs, you can just leave the http: off and just do //www.foxnews.com. And then any URLs on that page inherit the security of the containing page. So that if the page is HTTPS, then all of the nonspecific links get upgraded to that.

JASON: Right, that makes sense.

Steve: Yeah. Although what you really want to do is just - it's time for, like, to everybody to switch over. So anyway, I just wanted to give some more detailed coverage because a lot of people are going to be seeing either "not secure" or this "not fully secure." And, you know, people are squawking. But this transition cannot have caught anyone by surprise. Google has deliberately signaled the plans to do this for more than a couple of years, well in advance, to website admins, all of whom had ample time to migrate their sites to fully secure connections. I mean, maybe they just want to be stubborn, and they don't want to.

But clearly that's not the case for Fox News. They've got a good DigiCert certificate. They just haven't gone like the final step of moving HTTP, you know, if they've got the cert, HTTP accesses should simply redirect. I mean, that's like one line in the Apache server

config in order to redirect the user to the same URL, but with "S" at the end of HTTP. So that's trivial. Just do it. And then it's clear that they need to go in and remove some older HTTPs within the content of the page, like that search form, and switch it over to HTTPS.

So anyway, Google tends to lead by having people kicking and screaming, but leading by example and getting people to make these changes. Nobody's ever happy with it, but this is what it takes. And so I'm sure there will be people who will, you know, this puts pressure on sites to decide what they want to do. Maybe somebody wants to be stubborn and just stay HTTP. It's certainly their right to do so. But it is true that the site is not secured. It doesn't really mean it's not secure, but I understand what Google is doing.

JASON: I mean, not only is the site not secured - and sure, that's up to the site owner to make that determination. But no matter what going forward, anyone using Google Chrome to navigate to that site is going to be presented with this big scary red screen. I mean, at some point, I mean, this is what Google's move in this regard was all about; right? Like yes, they signaled it far in advance. They gave people ample warning.

But even Google knew that there was going to be an insane number of sites that weren't going to do the work to update to this. And they know that people are going to squawk and be upset when they try and visit their favorite website and they're presented with this, and that the website owner isn't going to like it. But they did it anyways because they know that, in the long run, the end game is that more of these sites are going to switch over, even kicking and screaming, but eventually they will because they aren't going to sacrifice the potential of their site losing traffic or whatever because people are scared of it now.

Steve: Well, and they would arguably not switch unless there was some pressure on them.

JASON: Absolutely.

Steve: And as you said, I mean, again, more than half of the top one million sites right now, 542,605, are not pushing people to HTTPS when the site supports both. They're allowing people to come in with HTTP. So if they've got bookmarks with HTTP, they just keep using that entry because that's the default, because the site doesn't bounce them over to the secure version. And we also know that Google's not done. This is the Chrome 68 step. Downstream this gets more scary. It gets red with a slash through it if sites don't take the hint at this point. So it's not over yet.

JASON: Yeah, absolutely.

Steve: So there were some other cool features, I won't go into them in detail, that also were dropped with Chrome 68. Most of them were for developer side enhancements, some CSS enhancements, and WebRTC got some upgrades. But they did also add the Web Payments API, which has been evolving over time. The intention of the Web Payments API is to provide - to essentially standardize through the W3C an API for purchasing stuff so that users who purchase through their web browsers, as so many of us do now, will be able to use, where sites support it, a unified experience for purchasing. So that's one thing that I expect users to see.

The one thing that is still a puzzle for me is that we were supposed to get this strict site isolation feature in Chrome 68. And I've got Chrome 68 now, and it's still off. So I don't know why. I talked about it a couple weeks ago. Leo brought up Chrome that was supposed to have it. And if you put in the URL chrome://flags, then you search for "isol" as the first four letters of isolation. That brings up a selection of instances of settings.

And they're all, like the main site isolation for me yesterday when I double-checked again, was still disabled.

JASON: Yeah, mine's disabled right now, as well.

Steve: Yeah. And we keep hearing that it's enabled for almost all users. And I'm thinking, almost all who? I don't know where that is. But it's very cool because it deeply sandboxes the rendering, cross-domain rendering. So it breaks even one page, it's not just a page per process. Now it's a domain per process to further insulate domains from each other. Potentially expensive in terms of resources on the client system, on the client machine, but so much stronger that Google is looking at backing out of their Spectre mitigations because they believe that this per-domain isolation, when it's available, if it's ever available, will be really useful. So we'll see.

JASON: I just activated it myself and then restarted my - because I'm on Chrome OS, restarted the entire computer and then realized that's a horrible idea during a show. When I do these things, it usually means I mess something up along the way. But thankfully everything's all back up, so we're good to go.

Steve: Yay.

JASON: So thank you for that tip. All right. So what else - he's been busy. What else is Ron Wyden up to, and why are we still talking about Flash? It amazes me that this is still a topic in 2018, but it is. Here we are.

Steve: Unbelievable. It really is. So we've talked about how glad we are that Adobe is finally going to shoot it and just put it out of its misery.

JASON: Yes.

Steve: By the end of - but not, like, tomorrow. They're dragging it out to the end of 2020. So Ron Wyden has written another one of his letters, this time to - get this - the NIST (our U.S. National Institute of Standards and Technology), the NSA (the National Security Agency), and the DHS, (the U.S. Department of Homeland Security). The letter asks those government officials in those departments to find solutions and procedures to mandate the removal of Adobe Flash content from all U.S. government websites by August 1st of 2019. Okay, so one year from today. So that makes it about a year and a half before its official end of life. And of course we know that, as I said, Adobe is due to formally end-of-life Flash in 2020, after which they have said they would cease to provide any technical support for the software. Okay. Lord help us. It just needs to go away.

JASON: Yeah.

Steve: In his letter, well, and, I mean, it can because HTML5 - basically Flash now is only an old-school media player. It used to be that games and puzzles and all kinds of stuff were run on Flash because it was - we didn't really have JavaScript as pervasive as we do now. The web just didn't have the media handling and especially the video playing capabilities. Well, all that's now available in every browser through HTML5. So it's only inertia. Again, it's only because Flash used to be what people were using that it's still there.

JASON: Well, and what my question in that regard of it being inertia, okay, it was there. Are they still actively using it and updating it? Or is it just like legacy Flash pieces? Because this isn't just about security. It's also about broken sites because as users we're not using Flash anymore. So we visit a site that has Flash, we're missing something. Or even worse, it looks like garbage.

Steve: Well, and of course iOS devices have never supported Flash.

JASON: Right, that's true.

Steve: That was one of Jobs' big issues was he said I refuse to support that on the iPhone or on the iPad. So it's always had a problem there. So anyway, so Ron said in his letter, he said: "The federal government has too often failed to transition away from software that has been decommissioned." Actually, he referred to Windows XP, which I guess the federal government is still using. And frankly, I have to say, I would be, too, if my XP machine hadn't died last month. But now I'm on Windows 7. I'm happy there.

And he says, he points out Flash's serious, and I love this, "largely unfixable cybersecurity issues" as one of the reasons U.S. government sites should take proactive steps to permanently remove this technology from all of its sites well before the cut-off date, after which, as he says, Adobe won't provide any security fixes at all. So he asks that government agencies not deploy new Flash-based content on any federal website, effective within 60 days; that federal agencies remove all Flash-based content from their websites by a year from tomorrow, August 1st, 2019; and require agencies to remove Flash from desktop computers by that same date, by August 1st, 2019.

So anyway, it is in decline. We know that. This last February Google's director of engineering said that the daily use by their Chrome users had dropped from 80 percent, that is, I should say Chrome users who encountered at least one page containing Flash content per day had dropped from 80 percent in 2014. So four years ago, four out of five Chrome users did encounter at least one page containing Flash content. Now that's 8 percent. So from 80 percent to 8 percent by the time he made this statement in February of this year. So way down. And there's a web technology survey site, W3Techs, said that only 4.9 percent of today's websites utilized Flash, which is down from 28.5 percent at the beginning of 2011.

So that doesn't represent as much of a precipitous decline as Chrome's users have encountered, that is, a factor of 10 drop. Here it's a factor of, what, like five. But still, it's clear that it's time for it to go. So I think what'll end up happening, I mean, it's been so deprecated that it must just be sort of abandoned sites that have not figured out by now that, boy, you know, if we want our videos to play, we've just got to move over to HTML5.

JASON: Yeah. I guess it must be painful to move your Flash content over because, if a government site is still serving it up - well, either that, or there's a theme here.

Steve: Well, and if it's not - yeah, exactly. There is the "we don't care" theme.

JASON: Yes, exactly.

Steve: But if it is active Flash content, then it may be they no longer have the source. They lost the programmer. They've got this SWF thing that actually does some business logic in it. And someone's going to have to recode that in JavaScript. And that will take some resources. So, I mean, it's certainly the case that there were other things than video that Flash was able to do. And so if there are sites that have proprietary business logic written in Flash, well, sorry, folks, you're going to be SOL. I mean, there's no question that browsers will refuse and OSes will refuse to run Flash in a couple years. So minimize the pain by getting it done sooner rather than later.

JASON: I guess it's also a cautionary tale for anyone building a website or building a service around certain tools like that; right? You hinge so much of what you're offering to people on a certain piece of technology. And if and when it's deemed insecure or decides

to go away, yeah, you're left footing the bill. But I guess that's just the world that we live in.

Steve: Well, 10 years ago it probably made sense.

JASON: Yeah, good amount of time. Yeah, exactly, it was a good amount of time.

Steve: Yeah, there was no useful alternative back then. Now there is.

JASON: Absolutely is.

Steve: I did have in the notes, and I got off on the strict site isolation, I did want to mention that Firefox is also moving toward strict site isolation. They have a project that they call Fission, which I don't know, seems like the reverse of strict site isolation. But anyway, that's the name of the project. And it's intending to provide much stricter site isolation sandboxing, very much like what Chrome is doing, and bringing that to the Mozilla Firefox platform. So everyone's going to experiment with that. We'll hope that it doesn't become too burdensome on our systems. But it does look like giving each domain its own process would be a useful increase in sandboxing security.

JASON: Yeah, and reading through that what kind of was a little alarming was their example of showing just how much memory overhead they were getting right now, and the number's far too high. When I think of Firefox as a browser, and it might be an outdated belief because I don't use Firefox on a daily basis necessarily, definitely not on mobile, a little bit on desktop, but I don't think of it as the most memory capable, let's say. It always seems to kind of slow itself down in my experience with it. So that seems like a big hurdle for them to overcome here.

Steve: Yeah, yeah. There was an interesting piece in Bleeping Computer that sort of caught me up short, I guess. And I wanted to make sure I put it on our listeners' radar. And it's a problem. And that is that there is a trend now for the registration by third parties of legitimate domain names in other top-level domains. And Bleeping Computer's article covered specifically French and Spanish language versions of legitimate sites: keepass.fr, 7zip.fr, inkscape.fr, gparted.fr, clonezilla.fr, audacity.es, clonezilla.es, handbrake.es, gimp.es, thunderbird.es, audacity.fr, I mean, and on and on.

And what's happening is that these are not sites that are put up by the owners of the official dotcom domains. They normally do contain French and Spanish language translations of the legitimate sites, but they are offering adware-laden downloads of those products, of 7-Zip or KeePass or GParted or Clonezilla. So of course the elevated danger beyond that is that they could also be offering fraudulent malware. At this point they're not. What they're doing is they are wrapping them in adware installers because the adware pays per installation.

So these are legitimate-looking, language-specific websites using the second-level domain of the legitimate site that's normally a dotcom domain, offering altered downloads which install components of software that the original providers don't have anything to do with, and certainly don't want to be supporting. So just be aware. I don't know, I mean, the problem is, in this instance, all of these various sites were registered using the same email address. So it's one entity that has decided this is the way they're going to make money is they're going to adware-enhance legitimate software, create foreign language translations of the original English-language dotcom sites under .fr and .es, which of course are French and Spanish, respectively. So under those TLDs.

And of course, I mean, some companies will take the time and effort to register the equivalent domain in every other TLD, which, I mean, it's one of the problems, one of the

fundamental problems with the way the Internet is structured is what do you do about that? I mean, if you invest in a major trademark like GRC or like TWiT, well, you don't want some creep registering the same second-level domain in a different TLD and drawing some traffic to their site under the belief that it's related in some way to GRC.com or TWiT.tv or Google.anything else.

So the tendency for a commercial company is to immediately also grab their main dotcom site name in all other TLDs and essentially just squat on them or grab them and then aim them all, redirect them all to their primary authentic domain. But if you do that, then that kind of kills all the whole point of having individual country-based TLDs. So the whole thing just sort of didn't evolve right. And I guess the way it should work is that an English-language dotcom site could create an authentic French-language .fr and Spanish language .es version of their own site, and then do it legitimately.

But that's also a huge amount of work. And a lot of these are just open source sites, you know, they're .orgs or .nets. I mean, they're just - they don't have the resources. And, by the way, as we know, you have to pay annually for domain names. So if you go and grab your name in every TLD there is, suddenly your annual maintenance cost just jumps up because you've got to maintain the registration of those everywhere, and some of them are not as inexpensive as the dotcom variants. And there's a bizarre, I mean, there's, like, they're constantly proliferating, with there being more of them all the time.

So it's a mess, and I don't see a good solution. The problem is, since they're legitimate registrations, and they legitimately own the domain, they can get, you know, they can be using Let's Encrypt and have certificates and HTTPS and have Google Chrome say, yeah, this is a secure site. And if it looks legitimate, how would a user know that, if it's a French translation of a dotcom with a valid certificate that looks right, how would they know not to download the software and in fact get this extra adware installed on their computer that they don't want to have? So anyway, I just wanted to sort of put it on everyone's radar because it's a problem. And I don't see a solution.

JASON: Yeah, that sounds nasty. And it's not just the fact that you end up at the site, and you see the TLD, and everything checks out in your head, looks legit or whatever.

Steve: It is.

JASON: But also the fact that, like, you installed the app that you expected to install, and it's not like it's some wacky other thing. It's exactly what you wanted. It just did this other stuff underneath. So from that point on you still kind of go on living your life, not thinking anything happened because every single step of the way checked out.

Steve: Yup.

JASON: Seemingly, anyways. The article on Bleeping Computer mentioned something called VirusTotal as a service to upload files that you download from these things. Are you aware of that service? Is it like, if you don't have a virus scanner on your computer, that's a way to run the file through to make sure it's legit?

Steve: Yeah, it's very cool. We have talked about it on the podcast. It's a service owned by Google. And you're able to, if you have - the way it's mostly used is, if you have a file that you just - for some reason you feel a little sketchy about it, like, I'm not sure where this came from. I found this on a thumb drive, or a friend gave it to me or something. It's free. And what it does is, I don't remember how many there are, it was like 70 or 80. What it does is it passes the file through every AV solution in the industry and shows you how many of them think that there's something sketchy about it. And so it's very cool. You can also do it with URLs, so like is this a sketchy site URL? It'll evaluate those.

For a while I was, with SQRL, Windows Defender was taking a while and scrutinizing my SQRL client deeply. And so I kept putting versions of it up on VirusTotal, and it turned out, I mean, I'd just built it fresh. I knew that there was nothing weird in it. It turns out it was just the reputation. It hadn't earned a reputation because it was so new. And that behavior all went away, and now there's no delay. But, yeah, VirusTotal is a very cool solution, and available for free.

JASON: Yeah, there it is. Nice to know about that. I didn't realize that existed. Awesome stuff. This next one I've been very curious about. This is about Google's kind of Play Store rules, and that involves cryptocurrency mining now.

Steve: Well, yeah. And for a while, I mean, cryptocurrency mining is on the outs, obviously.

JASON: Yeah.

Steve: The whole idea of mal-mining or cryptojacking, stealing someone's processor time, has been problematical. Actually it's a bigger problem on smartphones because, although smartphones are balanced to allow the processor to burst when necessary to get some work done, they assume most of the time the processor is going to drop to a more or less idle, low power state. Of course, that's antithetical to cryptocurrency mining that wants to just go balls to the wall and squeeze as much processing out of any processor and GPU that it can find. So there have been instances of devices truly being destroyed by malicious mining on them. Batteries overheat; cases expand; gases are produced; the cases break, are deformed; the batteries leak; all kinds of stuff.

So over time we've been seeing Google sort of incrementally putting the screws to cryptocurrency mining. For a while, the intermediate step was that, as long as the mining was not deceptive, that is, as long as an app said, "Hi, I'm John's Ethereum miner. Download me and plug your phone in because otherwise I'm going to drain your battery. Let's mine some Ethereum." I mean, when an app was right upfront, Google said, well, okay. If the user wants to do that, then fine. Whereas the mining that did it surreptitiously in the background, those had been officially banned and were removed whenever they were found.

Well, what happened just now is Google has changed their policy. They have now, in their Developer Policy Center, which is a long, comprehensive document about all the things, and it's a growing list of things, that apps are not allowed to do, all kinds of deceptive stuff and pretending to be one thing and being another, you know, basically the list is growing because Google wants to have a basis for rejecting things from the Play Store and say, look, you're in violation of our policy.

So what just appeared under the Developer Policy Center, under a new section that was added called "Financial Instruments," they said: "We don't allow apps that expose users to deceptive or harmful financial instruments." And then they said: "We do not allow apps that provide users with the ability to trade binary options." And specifically, under "Cryptocurrencies," they said: "We don't allow apps that mine cryptocurrency on devices, period." They said: "We permit apps that remotely manage the mining of cryptocurrency."

So this is a change. So even a Play Store app that is, as I was mentioning before, is right upfront and telling it like it is, sorry, that's no longer the case. They did not notify anyone of these changes. They did not publicly preannounce this. So there were developers who had apps removed who were complaining on Reddit about this. But Google said, sorry, we've made a policy change. We're no longer going to allow any cryptocurrency mining at all to be sold through or offered through the Google Play Store. So I suppose you

could still sideload if you wanted to in order to get things onto your device. But it's probably a bad idea.

JASON: And if you're mining cryptocurrency, there's probably some sort of Venn diagram between, like, cryptocurrency on mobile miners and people who are comfortable sideloading apps. There's probably a big overlap there; you know what I mean?

Steve: Yes.

JASON: I'm sure it'll be all right. I'm sure they'll be okay. But this is good news, I think, for Google to do this. They're not alone. Apple did this very recently, as well.

Steve: Last month.

JASON: Google's been blocking this from Chrome extensions in the Web Store. So, yeah, this makes sense.

Steve: Yeah. It was time. And I think it was clearly something, I mean, the handwriting was on the wall that this was not going to be allowed to continue.

JASON: Yeah.

Steve: We've talked also about in the past about supply chain attacks. And this is sort of worrisome. One that occurred recently was an app that Leo's not a big fan of, but actually I am, CCleaner, got a version of itself infected, like the actual website for CCleaner was attacked, and the download for CCleaner was altered. I don't remember now what malicious was put into it. It might have been file encrypting malware, or it might have been a cryptocurrency miner. But it was found and fixed quickly.

But the point is that there people were legitimately downloading from the CCleaner site an app that was not what it appeared to me in a so-called supply chain attack. Microsoft in their cloud blogs talked about their discovery of a new, a recent supply chain attack which Bleeping Computer covered. And I'm just going to share what Bleeping Computer wrote because it's a nice summary. Microsoft's write-up, if anyone is interested, I have a link in the show notes, and it is much longer. Bleeping Computer did a nice summary.

They said: "Microsoft said today that hackers compromised a font package installed by a PDF editor app and used it to deploy a cryptocurrency miner on users' computers. The OS maker" - meaning Microsoft - "discovered the incident after its staff received alerts via the Windows Defender ATP, the commercial version of the Windows Defender antivirus. Microsoft employees say they investigated the alerts and determined that attackers breached the cloud server infrastructure of a software company providing font packages as MSI files." That's the Microsoft installer format.

"These MSI files were offered to other software companies. One of these downstream companies was using these font packages for its PDF editor app, which would download the MSI files from the original company's cloud servers during the editor's installation routine. But hackers created a copy of the company's cloud servers. Microsoft security researchers said that attackers created the first company's infrastructure, duplicating it on a replica server that the attackers owned and controlled. They copied and hosted all MSI files, including font packages, all clean and digitally signed, in the replica server. Then the attackers decompiled and modified one MSI file, which was an Asian fonts pack, to add the malicious payload with the bit mining code.

Using an unspecified weakness, which they said did not appear to be a man in the middle or a DNS hijack, the attackers were able to influence the downstream parameters used by the PDF editor app. That's the one which was then following up and downloading the

font packages. The parameters included a new download link that pointed to the attackers' server. Users who downloaded and ran the PDF editor app would unknowingly install the font packages, including the malicious one, from the hackers' cloned server. Because the PDF editor app was installed under system privileges, the malicious coin miner code hidden inside would receive full access to a user's system. So actually it was sort of fortuitous and lucky that all it wanted to do was to mine cryptocurrency and not get up to more mischief.

"The malicious miner would create its own process named xbox-service.exe under which it would mine for cryptocurrencies using victims' computers." And I was thinking about that. I mean, if you looked into why your system suddenly was running slower, and then checked with process monitor and saw that xbox-service.exe was using up a lot of processor, well, maybe you'd think, oh, well, okay. Maybe it's supposed to. You know, I mean, a lot of people aren't going to know.

Microsoft said Windows Defender ATP detected mining-specific behavior from this process. Investigators then tracked down the origin of the process to the PDF editor app installer and the MSI font packages. Security researchers said it was easy to identify which MSI font package was the malicious one because all the other MSI files were digitally signed by the original software company except that one file, which lost its authenticity when the crooks injected their coin miner code into it.

So anyway, this is another problem that we're seeing in various forms, the supply chain attack where users are being as responsible and cautious as possible, where they are visiting a site whose security looks good. It's got its certificates intact. Everything looks copacetic. They download an installer which looks like it's doing a legitimate job. It's signed. It's working correctly. But once the PDF, in this case, PDF viewer or the app that was installed by the installer runs, it then reaches out and flushes itself out. At that point, it's going to a different, you know, to this cloned network. And we don't know how it was pointed to a different service, but it was, and downloaded this altered font package, among all the font packages, which installed the cryptocurrency miner.

So the problem is everything was done right by the end user, yet they still got a cryptocurrency miner in their system. So again, running as system, where the malware could have done anything it wanted to. So I don't know how someone would protect themselves from this. It's certainly beyond the expectation of an end user to, well, actually the app you've installed is just going out and doing stuff on the network, and you trust it because it was secured by the installer, and the installer was secured by the download, and the download was secured by the certificate of the sites you went to. So an intact chain of trust. Yet you've still got a cryptocurrency miner in you. So this is a fruitful attack, and I have no doubt we'll be seeing more of them in the future.

JASON: And when a computer has been co-opted like that, there's a cryptocurrency miner happening active in the background, like what does the user of the computer actually notice? Do they notice any sort of slowdown? I mean, that's not light work that's happening. Or I don't know; is it?

Steve: Right. It is, yeah, well, no, no, it is normally very heavy.

JASON: Yeah.

Steve: Because the cryptocurrency miner wants to use as much of the system's resources as possible. So one of the things - but the cryptocurrency miner also wants to remain undiscovered to whatever degree is possible. It doesn't want to have itself removed by drawing attention to itself. So what normally happens is, and we've talked about this in the past, they are beginning to be a little foxier about when they use the system. Like they'll notice if the system is idle and, for example, use all of the CPU, but

use it at the lowest priority possible so that other things that need to run, running at normal priority, will be able to preempt the cryptocurrency miner, yet it'll still suck up all the slack time.

Or it may wait for a period of inactivity so that there actually is an API call that is, like, notify on system inactivity, where some period of time of no mouse and keyboard use has occurred. So then it figures, oh, I'm still running, but the user has walked away, so now's a good time to do some mining. And then the second the user touches the mouse or types something, the miner will suspend itself, again not wanting to give away the fact that it's there.

JASON: Interesting. Yeah, I definitely see more of that, for sure.

Steve: Yeah. So Titan security keys. This was big news last week. Google launched their own USB and Bluetooth-based FIDO (that's the Fast IDentity Online, F-I-D-O) FIDO U2F (which is the Universal 2nd Factor protocol) hardware tokens. Of course we're all familiar with the equivalent from YubiKey, which is actually the company that I'm probably responsible for discovering. Stina tells everybody that it was this podcast, which occurred a couple weeks after a significant RSA conference. I happened to be attending the RSA conference for the podcast with press credentials, and Stina was there standing at the top of an escalator looking around for some press people. And she asked me if I was interested in authentication, and I said yeah. And I didn't know how much I was interested; but, I mean, it had been a constant topic of ours for the podcast. We talked about the little eBay and PayPal football that generated the six-character changing one-time password token.

Anyway, so she explained that she had a little thing that emulated a USB keyboard, and it produced a one-time password. And every time you touched it, it spit out a different string that was cryptographically secure. And I gasped because it was such a clever and cool idea to use this little tiny fob as a USB keyboard. It needed no drivers. You could just put the cursor in a field and touch it, and it would type this crypto string into a browser form and then off you'd go. The first chunk of digits was not changing. That was its ID. And the second longer string was a one-time password which changed every time.

So, I mean, it was just - it was beautiful. And three weeks I think or so later I did the entire podcast on the YubiKey, explaining how it worked and what it did. And Stina, I guess she has - there was some presentation someone pointed me to years ago where she said, yeah, we got put on the map thanks to Security Now!, which was very cool.

JASON: That's awesome.

Steve: So they've been, I mean, they have been the hardware dongle provider of choice. And I don't know what their relationship to Google is. I know that they've worked with Google, that Google has had lots of YubiKeys in the past. And so I don't know if this has been done in conjunction with Google, or if Google just decided they wanted something different or go in a different direction than Yubico. But so this is something like that.

I did give Yubico a full presentation on my solution, SQRL, last November and sat down with Stina and three other techies, I mean, like THE crypto techies, ran through the whole thing, answered all their questions, and they were very interested. They're basically just waiting for me to get this thing done. And if it gets traction, they're very interested in doing a SQRL-based hardware token of the same ilk as what they've done before because SQRL you can absolutely have the user's super secret one private key put into a piece of hardware to protect it from any possibility of theft. And I've gotten a lot of questions, not surprisingly, as time has gone on with AuthN and FIDO and so forth, about how SQRL and FIDO are different. And I will certainly address that in a document to put them side by side.

One of the things I will say is that FIDO is by its nature, and even by its name, you know, U2F, Universal 2nd Factor. What it is not is a first factor. And that's probably one of the biggest differences is that what SQRL is, is a secure one factor, that is, it identifies you and authenticates that identity. None of these 2nd factor things are able to identify you. They have to know who you are claiming to be first because they store your credentials on the server. So the server has to send the credentials back to you for you to then authenticate against those credentials. So it's a small thing, but it makes a big difference in the login flow because with SQRL you don't have to identify yourself. You just click the "Log in with SQRL" button, and you're done.

And there's a lot more, too. SQRL has what we call "complete lifecycle management," or "identity lifecycle management," where if a bad guy gets your SQRL key, malware gets into your phone where you're storing it, or something happens, or you were to lose it, there is a provision for locking down your identity and securely recovering your identity and managing it over the long term, none of which is built into FIDO. That's considered all, like, per site. Each site will handle that individually.

So anyway, when it's finally done, when I get SQRL's authentication working on the SQRL public web forums, then it'll be time to officially announce and launch this. And there is an Android client, an iOS client, web extension. There's a native Linux client, a PHP support server side, Java server side code. So there's a lot of other stuff going on that I'll be talking about in the meantime. And right now the Google Cloud users are able to get these Titan security keys. They are said to be available widely through the Google Store at some point in the future. And I look forward to learning a little more about what's going on behind the scenes with these. Are they actually being manufactured by Yubico and privately branded, or what? I don't know.

JASON: Or completely removed from Yubico. I guess Google's had its employees, 85,000 of them, using these types of security keys since 2017. They've said that there have been no account takeovers since they put that requirement in place. So it'd be interesting if they were using the Yubico keys all that time and then, like, all right, we got what we needed. Now we have our own. See you later. Hopefully there's some sort of a relationship there.

Steve: Yeah. Yeah.

JASON: Well, that's cool. I want to get one myself and check it out.

Steve: Sounds cool. So I did get a nice note on the topic of SpinRite, which I try to share with our users every week when I can, from Joshua Montgomery, who's in Springfield, Illinois. The subject was "SpinRite Saved One of Our ATMs." He sent this on the 26th of July. He said: "Steve, I've been an avid listener since the beginning of Security Now!. Thanks to you and Leo for all of your hard work." He says: "I wanted to thank you for making this splendid product. It saved my neck. We had a very high-traffic ATM hard drive fail. The tech said they could not make it for 48 hours." He says: "I pulled the hard drive from the ATM and ran SpinRite on it. It fixed enough of the drive that I could clone it to an SSD within three hours. Now we have one the fastest ATM machines in the North. Keep up the good work. Joshua."

JASON: That's got to feel good, to get a message like that.

Steve: I get them all the time, and I really do love it, to be able to, just like here, SpinRite just fixes the drive and then off people go again.

JASON: Yeah, from 48 hours to three hours, no big deal.

Steve: Especially on a high-traffic ATM. Apparently it was, like, really important.

JASON: Yeah, you don't want any downtime on something like that. That's people's money. Including your own.

Steve: Yup.

JASON: So tell me a little bit about Bluetooth because I'm, like, late to the game on Bluetooth, at least when it comes to headphones, anyway. I mean, I have Bluetooth devices that I use on a regular basis, but I've been reluctant to use Bluetooth over the years just because I find it not as dependable, dropouts. There's always the kind of potential for security risk depending on what's going over. And this definitely ties into that.

Steve: Well, and you've got to keep them charged. And so there's that, also. And actually I'm a little concerned about the health risks. I mean, you're sticking a radio in your ear canal.

JASON: That's true.

Steve: And it's like, let other people do that. I just think I'd rather not do that myself. But Bluetooth of course has lots of other applications. It's become, because it is high speed and low power, and it's sort of got a nice range compromise, you know, NFC is deliberately like contact near-field radio. And of course WiFi is deliberately long-range radio. Bluetooth sort of fits right there in the middle, at 30 feet or 10 meters as the official distance, and then there are some extended distances. But it has always been - there's always been a vulnerability.

In fact, the research paper which was published and will be given is titled "Breaking the Bluetooth Pairing Fixed Coordinate Invalid Curve Attack." And our listeners have heard me say many times in the past that the only way I think it is secure to pair Bluetooth devices is to walk out into the middle of a large parking lot, like on a Sunday morning when it's empty, or maybe go into the middle of a football stadium. That is, physically isolate yourself from the environment and do the pairing because Bluetooth pairing is the moment of vulnerability. That's when there is stuff happening that is the devices finding each other and exchanging cryptographic material and deciding, like agreeing upon a secret key which they will then use for everything else. That's the weakness. That's the moment of maximum vulnerability.

Now, the protocol works to minimize that. There's all this, you know, each side displays a six-digit code to the other, or one side is a screen and the other side has a key. And so you enter the number that one side is showing. The idea is they're trying to use an out-of-band communication, that is, a visual real-world communication outside of the radio band. But there are many instances where you don't have that luxury. You've got like an IoT device. It's a door lock. Or, well, a door lock might have a keypad, so that's a bad example. But like a sliding window lock where it's just a little magnet that is going to use a Bluetooth connection to the home hub, and it doesn't have anything.

So pairing has always been problematical. And so the spec over time has evolved through Bluetooth, well, up to 4.0 and 4.2, continually working to improve the pairing technology. So the crypto used in the latest Bluetooth is Elliptic Curve Diffie Hellman, ECDH, which is an evolution of the original RSA-style Diffie Hellman, which is really a cool technology. We've talked about this before. This is a key agreement protocol which, even though it's hard to imagine, two parties can exchange data in the clear, that is, can publicly exchange some information with each other. And even though a third party can

capture all of their interaction, those first two parties are able to securely agree upon a secret which the third party, despite seeing all of their conversation, cannot obtain.

So it's just - it's very cool. And it's one of the fundamental architectures for cryptography. And elliptic curve is being used today because it allows much shorter keys. And that ends up being important. The packet length, for example, for Bluetooth Low Energy is - I think it's 16 bits, or it's very, very small. So maybe it's 16 bytes. Anyway, it's very small. So it's impractical to exchange a large key in a Bluetooth Low Energy environment. You really want the key to be shorter. And the computational power of these little devices is really reduced. So that's why elliptic curve is where everybody is going moving forward.

Okay. So what's this attack? The short version is it is nothing to worry about. What these guys discovered is an active attack where there should not be one. But it does require that an attacker be involved in the pairing, be able to intercept and modify some of the traffic that is being exchanged. What the current spec fails to do is to authenticate one of the parameters that is being exchanged. In Elliptic Curve Diffie Hellman there is the private side of the public and private key pair. The private side is a large scaler number which is kept secret. The public side is a coordinate on the elliptic curve, which is exchanged as an X and a Y. It turns out that the protocol only requires, well, yeah, the protocol only requires that the X coordinate of the X and Y coordinate pair, which form the public side of the public/private key, is validated.

And what these attackers discovered is, if they can arrange to get themselves in a position to somehow zero the Y coordinate of the X/Y coordinate point on the curve with a 50% probability in some cases, or a 25% probability in most cases, they're able to decrypt the private key that these guys - that the two valid endpoints end up negotiating. Thus the reason that I don't think this is a big deal. So it's like only in the event of the pairing, only if an attacker can actually intercept and alter by zeroing some of the data being exchanged, can this even be done, and even then with a 25 or maybe 50% chance of determining what the private key is that is negotiated between the parties. And in the other case their having zeroed the Y coordinate causes the agreement to fail.

So they're either able to get the key that the parties negotiate or the Bluetooth pairing fails, which would cause it to be redone until it succeeds, which would suggest they're ultimately able to obtain the private key. But I don't know how, I mean, in practical terms, an attacker gets themselves in a position where they're able to prevent the recipient at each end, because it's an exchange, to prevent the recipient from receiving the proper coordinate. So you'd have to have directional antennas aimed at each of the devices being paired.

Also Bluetooth, remember, is also a frequency-hopping, spread-spectrum technology. So you have to lock onto the frequency hopping and know which frequency each side is going to be transmitting to the other on. That's now doable, but still it's more involved. You then presumably have to flood the recipient with a saturation, a radio saturation attack to prevent it from receiving the valid data from the other side; then stop flooding it and then send it the altered data which you would have received from the other side while it was trying to send it to the side that you were flooding; then alter that by zeroing the Y coordinate; and then forward it on to the side that you had just flooded; and then do the entire same thing immediately to the other side. So it's like, Holy Toledo.

JASON: So Steve, you're saying then that there's a chance that it could happen.

Steve: Yes. This is like, okay, uh, yeah.

JASON: Yeah, it sounds like it was a long way to go.

Steve: Yeah. And so it is a vulnerability. It is there. It is theoretical. Yes, it could happen if, like, all the planets aligned correctly and somehow a bad guy is desperate to do this and has the technology and is present during the event of pairing. I mean, maybe there is a - and I can't think of what it would be, like a high-value system which is going to be pairing over Bluetooth at a known time and place, and a bad guy could set up in order to, like, intercept it, that would be high enough value to be worth doing. But I really can't imagine the scenario.

So the attackers have looked at different systems. They did find that the Android Bluetooth stack Bluedroid is vulnerable, whereas Windows is not because it does not support the Low Energy SC pairing that is vulnerable. There is a protocol, SSP, Secure Simple Pairing. The vulnerability in SSP depends upon the Bluetooth chip firmware implementation, since the handshake is performed in the chip rather than by the host. They said that during their research they found that the devices of most chip vendors are affected. Qualcomm's, Broadcom's, and Intel's implementations are vulnerable, which they said together constitute most of the Bluetooth chip market.

So they said: "We stress that every device - mobile phone, laptop, or car - that uses such a chip is vulnerable." On the other hand, I would stress that it really doesn't matter because, I mean, think about this. Our listeners should know that there's this chance. It is also the case that everybody is scrambling to fix this. So Android will get an update.

JASON: I think they already did. I believe it may have been in the June security patch. But then again, it's part of the patch. Who has the patch? You know, that's the conundrum that Android users are always in.

Steve: Yes.

JASON: Like Google did something about it, and they included it in their security update. But did you ever get that update? Probably not.

Steve: Yup. So there's a lot of furor in the press about, oh my god, Bluetooth is vulnerable to hacking during pairing. It's like, yeah.

JASON: Technically so.

Steve: Theoretically. And anyway, the fix it turns out is simple. If the Y coordinate is zeroed, then the public side of the public key pair, which is supposed to be a point on the curve, will be invalid. So all anybody has to do is what they maybe should have always done, which is verify that the public point they receive from the other side is a valid point on the curve. They're not doing that. So that's all they have to do. So fixing it is not a big problem. It'll generate a next rev of the spec, which will require it because now we know, oops. Maybe there's some chance of it being compromised, and we don't want that to be the case because we all want Bluetooth to be secure. So it'll be even more secure.

I think it's fine just the way it is with this bizarre caveat that, okay, there's a theoretical way that the handshake could be intercepted that would, with some probability of 25 or 50%, allow the bad guy to negotiate in order to watch, to recover the private key that both guys end up with. But wow.

JASON: Far cry from BlueBorne, which I believe was the Bluetooth virus that we heard about last year.

Steve: Exactly. Exactly.

JASON: Steve. Got any other things that aren't included in this list that you want to kind of finish things off with? I know obviously GRC.com, but I don't want to start the close of the show yet until I know for sure.

Steve: Close it. Close it. We're done.

JASON: You've got it all in. It's all in there.

Steve: My work here is finished.

JASON: Your memory dump is done. It's over. And I really appreciate it. GRC.com. I know everybody listening and watching appreciate it, too. Anybody who wants to catch all the stuff that Steve is up to, go to GRC.com. Obviously you talked about SpinRite, best hard drive recovery and maintenance tool. You can get your copy there. Information about SQRL. Audio and video of the show, you're hosting that on that site, as well, as well as transcripts; right?

Steve: Yup. Yup, exactly.

JASON: Right on. People can find all that there. TWiT.tv/sn, if you want to go to our site. And you can of course find all the audio and video there, subscribe to the podcast, Security Now!. Do a search for Security Now! in any of the podcatchers that you're using, you're going to find it. Search for TWiT or Security Now! and you'll definitely come across that. And then of course we have a live chatroom, got a live stream. We do this show every Tuesday at 1:30 p.m. Pacific, 4:30 p.m. Eastern, 20:30 UTC. If you go to TWiT.tv/live you can tune in and watch as the show is recorded and participate in chat. Everybody's always talking about a number of the security topics. Everybody kind of expands on the topics of the day, and it's a lot of fun to participate in that way, as well.

Steve, I always appreciate having the chance to do a podcast with you, so thank you for allowing me to be your cohort today. I really appreciate it.

Steve: Well, likewise. And I won't see you next week or for a few weeks, but then you're going to be back for a run of three.

JASON: That's right.

Steve: So that'll be great. We will have more of the Jason/Steve team in September.

JASON: Right on. I'm looking forward to it. That's right, that's in September. So Leo will be out then, and I will see you then. But next week Leo and Steve will be back for another episode, so make sure you don't miss it, of Security Now!. Take care, everybody.

Steve: Thanks, Jason. Bye.