
# Careers in Bug Hunting

**Description:** This week we look at a newly available improvement in Spectre mitigation performance being rolled out by Microsoft and who can try it right now, Adobe's ColdFusion emergency and patch, more problems with AV and self-signed certs, a Docker vulnerability being exploited in the wild, the end of Coinhive, a new major Wireshark release, a nifty web browser website screenshot hack, continuing troubles with the over-privileged Thunderbolt interface, bot-based credential stuffing attacks, some SQRL, miscellany, SpinRite, and listener feedback. Then we examine the increasing feasibility of making a sustainable career out of hunting for software bugs.

High quality (64 kbps) mp3 audio file URL: http://media.GRC.com/sn/SN-704.mp3
Quarter size (16 kbps) mp3 audio file URL: http://media.GRC.com/sn/sn-704-lq.mp3

SHOW TEASE: It's time for Security Now!. Steve Gibson is here with the latest on a Docker exploit. Yes, there is a weird one. ColdFusion, why you might want to stop using that. And Spectre, another Spectre mitigation. But do you really need to protect yourself against Spectre? And one 19-year-old Argentine who's become a millionaire finding bugs. It's all coming up next on Security Now!.

**Leo Laporte:** This is Security Now! with Steve Gibson, Episode 704, recorded Tuesday, March 5th, 2019: Careers in Bug Hunting.

It's time for Security Now!, the show where we cover your security and privacy with the guy in charge, Mr. Steve Gibson, GRC.com, Gibson Research Corporation. Hello, Steve.

**Steve Gibson:** Yo, Leo.

**Leo:** How's the research going?

**Steve:** The research is coming along just fine.

**Leo:** I just thought about this. What are you researching at the Gibson Research Corporation?

**Steve:** Well, so the original - okay, had a little too much coffee this morning. The preceding company was Gibson Labs. And of course then I guess it would be like, well, okay, what are you mixing up in your test tubes and beakers in your laboratory?

**Leo:** Well, the Tech Guy website is the Tech Guy Labs. I like the image, anyway, yeah.

**Steve:** Yeah, I do, too. And what happened was there was some concern that the name was encumbered as a consequence of the way I wound down Gibson Labs where Atari kind of bought it because they wanted the Light Pen technology that I developed.

**Leo:** Oh, I didn't know that. Oh.

**Steve:** For the Apple II. And then Atari was collapsing because the home computing market back then and that first attempt at videogames was kind of collapsing. So they reneged on the deal, and I got it back. And then Koala, that had the KoalaPad, I had become good friends with Jeff Heimbuck, who was the VP of Marketing at Atari toward the end. And he'd gone over to become the president of Koala. And so I said, "Hey, Jeff. I got the Light Pen back. Want it?" And he said, "What do you mean, you got it back?" He says, "They're never going to give that up." I said, "Oh, they did. They didn't really have a choice."

So anyway, it was contractual, and attorneys and contracts and everything. So when I wanted to start up again, the best advice from my attorney was, you may want to stay away from Gibson Labs because, if you did something really amazing, someone might come back and say, "Hey, that's ours." And I said, "Oh, okay. I don't want that to happen." So I did, "Let me think, hmm, hmm. Oh, research. Gibson Research."

**Leo:** That's like Labs.

**Steve:** Yeah, it's kind of, you know. And I've been researching stuff.

**Leo:** That's a good name. I never - it's funny because I never really even thought about it. And I just said it, and I thought, hmm.

**Steve:** Yeah. And GRC, it's funny, too, because when we wanted to get a domain name, I remember telling the guy who was going to go do that for me, his name was Millard. Willard or Millard? No, it was Millard. Millard Ellingsworth III.

**Leo:** Oh, my. Oh, my. That's how long ago that was.

**Steve:** I said, "Millard, go get us a domain name." And he came back after, like, in two days or something because nothing was quick back then, I mean it was like ponies and camels and stuff. He said, okay, well, we can't have "Gibson." I said, "Ooh."

**Leo:** You wanted Gibson.com.

**Steve:** No, that was gone already. It was like guitars or refrigerators or something.

**Leo:** Oh, yeah, yeah, yeah, right, yeah.

**Steve:** And he said, "But how about GRC?" And I said, "Oh, I like that."

**Leo:** Three letters is even better.

**Steve:** Oh, my god. Now, I mean, I get offers all the time, not because they want me, they just want three letters.

**Leo:** Three letters, yeah.

**Steve:** And we registered GRC, we were about six months behind another famous domain, Microsoft.com.

**Leo:** Wow.

**Steve:** So, I mean, that was, you know, this was when it was all just beginning to happen back then. And of course Bill was going to do MSN to compete with CompuServe and The Source. And we're all going to get our modems, and we're going to stick the phones in there and dial into a big something.

**Leo:** [Modem sounds]

**Steve:** Yeah, that didn't quite happen the way he thought.

**Leo:** Don't you wish back then you'd thought to register a whole lot of domains? I mean, I wish I'd gotten Leo.com back then. It would have been nice.

**Steve:** Actually, it was very difficult. Back then they were all very picky about, well, Leo, that doesn't sound like a company name. Dot com is for companies. Edu was for schools. And org is for nonprofit. We need to see your certification.

**Leo:** Oh, yeah, I remember that, yeah.

**Steve:** It was very different, you know. And so now it's like, you want dot email? Fine. It's like, okay, whatever. So, yeah. And I feel the same way. I've often thought about, you know, I mean, I've never been a squatter. That annoys me. There have been people who've, like, offered me very tasty domains. And I've said, "You want to give it to me, I'll take it. You want to charge me 25,000, I just won't just because that's wrong."

And there are people who have come to me asking for domains that I've been keeping alive for no reason that I've never used, and I've said, "Yeah, you're right, I'm not going to use it. Here you go." And they're like, "What? You're giving it to me?" I said yeah, you know, because it's wrong to charge. That's just not right. So anyway, this week, speaking of things kind of evolving, there was some news that kind of thought, okay, we need to share this with our listeners because, well...

**Leo:** Of course it is.

**Steve:** It's generally true of this podcast every week, come to think of it. But this was career related. And that's my point. I knew there was one there somewhere. I've really had too much coffee.

**Leo:** That's not possible, Steve. That's not possible. You can't have too much coffee.

**Steve:** So "Careers in Bug Hunting" is the title of this Episode 704 of Security Now! because Santiago, I want to say Lopez, I don't have it in front of me, whoever he is, he made a million dollars.

**Leo:** And he's only 19.

**Steve:** Yes. He is a teenager.

**Leo:** Self-taught.

**Steve:** And he's got a little Mini Cooper, and now he's got a Peugeot. He's buying cars. He apparently likes to swim.

**Leo:** I'm relieved. Because when I saw the title, I thought this was out of storm - what is that troopers movie where they had "Go to space, hunt bugs"? Heinlein, remember that? The Heinlein story.

**Steve:** "Starship Troopers."

**Leo:** "Starship Troopers," thank you, thank you.

**Steve:** Anyway, so that's what we're going to talk about. We're going to wrap up by talking a little bit about and share - HackerOne is now the number one legitimate, unlike Zimperium. And I guess it's going to be a bidding war; right? If you've got a really tasty zero-day, Zimperium may be offering more. If you're having to support yourself, I wouldn't say that it's wrong to sell something to Zimperium. But hopefully HackerOne, who is not turning around and selling them, like to governments, would be an alternative. Anyway, the point is that...

**Leo:** I have lots of questions about this, and we'll talk, of course. But I really - I need to understand this a lot better.

**Steve:** So we're also going to take a look at a newly available improvement in Spectre mitigation performance, which is just today, actually, being rolled out by Microsoft. And there are four people who qualify because it turns out the newer Intel chips can't do this, but the older ones, Broadwell and older, can. There's some registry tweaks. If you're up to 1809 - and I heard you mention after MacBreak Weekly that your system was just now being offered 1809.

**Leo:** Yeah, my Microsoft, yeah, yeah.

**Steve:** That was the October 2018 release. And I've got mine holding off because I can. It's like, okay, I don't really want to take any arrows in the back. But anyway, the point is that the Spectre mitigation slowed things down, especially for older chips. Those older chips can now get a reprieve, maybe. We also have an Adobe ColdFusion emergency and patch.

**Leo:** Uh-oh.

**Steve:** Which, again, I hope anybody who's ColdFusion based, that's the platform that another - Adobe's things refuse to die, and I think Adobe should be worried about why everyone wants them to. But that's another matter.

**Leo:** Does Adobe own ColdFusion now?

**Steve:** Yeah. Adobe has ColdFusion. And again, just die. But it won't. But it should, just like everything else, Shockwave and Flash. I would worry if I were Adobe. And maybe this is a stock tip. I don't own any stock, so I can say this. If you want something to sell short, you know, the company where everyone wishes everything that they published would just die already. Anyway, I really do think I've got to lay off the caffeine.

**Leo:** No, this is going to be a fun show, everybody. Fasten your seatbelts. It's going to be a bumpy ride.

**Steve:** We also have more problems with AV and self-signed certs. A Docker vulnerability being exploited in the wild. The end, believe it or not, at the end of this week, Coinhive is shuttering themselves. We also have a major new Wireshark release, a nifty web browser screenshot website hack courtesy of Bleeping Computer, continuing troubles with the ever-overprivileged Thunderbolt interface, bot-based credential stuffing attacks, some little updates on SQRL, some miscellany, a bit of SpinRite feedback, some listener feedback, and then, breathlessly, we will be examining the increasing feasibility of making a sustainable and ecologically beneficial sustainable career out of hunting for software bugs. So yes, no sign of - I don't think the bugs are going to end, and certainly this podcast never will, either.

**Leo:** This very episode may never end, at this rate.

**Steve:** Single episode, just tuck yourself in. So our Picture of the Week ties into a story that we'll be getting to. But I just sort of liked it because nothing else was grabbing me for the week. And this is showing the result of a Shodan search for either of the two ports that Docker uses which are never supposed to be publicly exposed. Docker uses, as do now many systems, an internal network-based API. So the idea is that the Docker daemon binds itself to the localhost IP, 127.0.0.1, on two particular ports which are well-known ports for it, so that processes in the system are able to open a connection within the system.

So it's a nice, I mean, it turns out that this IP stack forms a very convenient Inter Process Communications system, IPC, for one process talking to another. And so the idea is that it's just meant for processes within the system. But because it is networking, if it is misconfigured, and I have to think some misguided people are doing this on purpose, like they think, oh, it'd be really nice to...

**Leo:** To be able to log in.

**Steve:** ...make our Docker API available to people in China. Like, oh...

**Leo:** What could possibly go wrong?

**Steve:** What could possibly go wrong with that? So this shows a map of the world on the left and shows that, of the 3,951 exposures apparently of Docker - and that number should be zero, right, it should not be 3,951 - 929 are in the U.S., followed by 680 in China, 240 in Singapore, 225 in Ireland, 224 in Germany, 214 in France, 212 in Canada, and so on down the line. So for whatever reason, and we'll be covering this in a few minutes, there are lots of opportunities for people to hook onto Docker. And, okay, I won't step on the punch line because we could probably at this point guess what has set up shop in those Docker containers.

Anyway, in the meantime, Microsoft has just rolled out on March 1st, which is last Friday, an update which is available to Windows users. Now, I had intended to go see if it was going to give it to me automatically, but it's only for the October 2018 update 1809 of Windows 10. I'm still back on 1803 because I'm not in a hurry to discover if there's anything that goes wrong when I update myself. But Leo, you could because your system was just updating itself an hour ago to 1809.

**Leo:** Yeah.

**Steve:** So what would be interesting would be to see if you got KB4482887. That was just made available on Friday, March 1st. What it does is it allows - oh, and maybe if it's smart it probably knows you didn't qualify because Skylake and later - wait, no, Skylake and newer - is that earlier? Sooner? I don't know. Anyway...

**Leo:** That's both the same thing. Newer or later, same thing.

**Steve:** More recent.

**Leo:** Yeah.

**Steve:** Okay. Oh, yeah, instead of earlier, later. Okay. More recent.

**Leo:** Have some more coffee. I think you've entered a space-time warp. Alex Gumpel, if you're listening, check to see if we've got 448287 on the 1809 update. I'd be curious. It's not a Skylake, I mean, it's later than Skylake, so it wouldn't - more recent.

**Steve:** Earlier? Newer?

**Leo:** Yeah.

**Steve:** Okay, good.

**Leo:** It wouldn't probably - I think it is. Actually, maybe not.

**Steve:** Yeah, it probably is.

**Leo:** It's a couple years old.

**Steve:** Leo, everything you've got is newer than that.

**Leo:** I would hope.

**Steve:** Anyway, so, I mean, remember I bought my current machine back when it was believed that Windows 7 was going to stop supporting Skylake. And it was like, what? They are going to do that? So I immediately bought, no, I guess it was going to stop supporting after Skylake. Then they thought better of that because many other people with more clout than I were a little annoyed by that fact.

Anyway, remember that Google invented something called Retpoline, which is just difficult to say, Retpoline. It's a contraction of "return trampoline." A trampoline is like the word sounds. A trampoline is something you bounce off of; right? So a return trampoline is a technique of bouncing off a return instruction. Google came up with this as a lighter weight solution to the second variant, the Variant 2 of Spectre. And the problem with it was Microsoft's reaction, or, well, Intel's solution was just to shut down a speculative execution benefit, which is what the Variant 2 of Spectre was leveraging. And just like throwing the switch, it's like, okay, we're turning that off.

Well, it caused a significant performance impact. So what Google realized was, you know, rather than just - there's only like some instances where this is really a problem. And if you modify your code in those particular places, then you can just kind of like make little micropatches all over, but leave it on the rest of the time where it can't actually be leveraged against you. So as Microsoft explains in their update, they said Retpoline - so they're incorporating this invention of Google. Thank you very much, Google.

"Retpoline works by replacing all indirect call or jumps in kernel-mode binaries with an indirect branch sequence that has safe speculation behavior." They wrote: "This proves to be much faster than running all of kernel mode code with branch speculation restricted." And this is their IBRS set to 1, which is their indirect branch restriction on speculation. "However," they write, "this construct is only safe to use on processors where the RETURN instruction does not speculate based on the contents of the indirect branch predictor."

And this is where everyone's head explodes, of course, because it's like, what? But not Google's. Google said what, okay, and then we'll figure out what that means. So Microsoft did, Google did, everybody, you know, said okay. So unfortunately, Intel's newer processors, thus the newness of them, do that. They speculate based on the contents of the indirect branch predictor, which means that Retpoline cannot be used on them.

**Leo:** Is that good or bad? I've kind of lost track at this point.

**Steve:** I know, I know.

**Leo:** So the newer processors are more vulnerable.

**Steve:** Yeah, well, they're more newer.

**Leo:** Okay.

**Steve:** So they're fancy.

**Leo:** I grant you that.

**Steve:** They're fancy. And their RETURN instructions, being fancier, do speculate based on the contents of the indirect branch predictor. Therefore we can't rely on Retpoline to work for them. Processors where we can are all AMD, which don't do that, as well as Intel processors from Broadwell and - okay, I'm going to get this right - earlier, meaning older and before, and up until and including.

**Leo:** No, pre-Broadwell, not Broadwell. Just like it was...

**Steve:** No, no, Broadwell and pre.

**Leo:** Oh, okay. So inclusive. Okay.

**Steve:** Yes. And so that means that Skylake came after Broadwell.

**Leo:** I got it. That's right. Yeah, and by the way, that Surface Studio is Skylake. So that would mean it's...

**Steve:** Yes, that you're not going to get this.

**Leo:** This Retpoline patch.

**Steve:** Retpoline cannot be used on your system because...

**Leo:** I feel like an idiot, but is Retpoline good? Or is Retpoline bad?

**Steve:** Retpoline is a fix for Variant 2 of Spectre.

**Leo:** Okay. But do I not need Retpoline because I have a Skylake processor?

**Steve:** Oh, you wish you could have it, but you can't.

**Leo:** Oh. So in other words, it's bad to have the newer processor because the Retpoline fix doesn't work.

**Steve:** Yes. Although it's kind of good to have it be newer because it's not slowed down as much by Variant 2 of Spectre.

**Leo:** Ah. Because the Retpoline has side effects, bad negative side effects.

**Steve:** Basically we thought we were going to finally stop talking about this. Like we were going to limit this to 2018. But no. Here it is in 2019 still, and it's just as mind...

**Leo:** Bending.

**Steve:** ...boggling as ever.

**Leo:** Yeah.

**Steve:** So here's the takeaway. If any of our listeners have - if their ears are not bleeding. If you happen to have Windows 10 with the 1809 October 2018 update, and they're not going to ever do this, they've said sorry, we're not going to go back in time any further because we're all exhausted. So if you have October 2018 update 1809 of Windows 10, and a Broadwell or older chip, which got slowed down by the 2018 panic over Spectre - and remember, we're, like, no one ever actually used this to attack anybody in the first place; right?

**Leo:** But they could.

**Steve:** Well, yeah, but on an end user's machine, you know, if you've got something in your machine attacking itself, then you've already got problems. So Retpoline and, I mean, willy-nilly, who cares? So the only problem was in cloud environments where you might have a deliberately shared hosting environment where malicious code was running in one VM, and this was being used to try to leak secrets out of an adjoining VM, which is not anything that your typical end user has anyway.

But essentially what'll happen is, if you are that person, Broadwell or older chip, where you would have seen things slow down, if you didn't, for example, use my InSpectre utility to turn this nonsense off because it's just not a problem in the first place, in time Microsoft is going to roll out for those people a performance improvement. Which you could apply today, on March 5th, if you follow some registry changes in the link to the update that I have in the show notes.

**Leo:** Oh.

**Steve:** And because that probably has winnowed down our entire applicable audience to five, I'm not going to go through that now. I'm just going to say that's there.

**Leo:** Is it in InSpectre? Did you put all this in InSpectre?

**Steve:** Yeah. InSpectre has always had this.

**Leo:** It knows all about it, okay.

**Steve:** So, yeah, you could turn this stuff off, and then your machine runs fast and fine. And again, we've always said we will certainly let everyone know if they should ever actually need to turn these mitigations on. As far as anyone knows, this has never actually been used to, I mean, this is probably the biggest example of the sky is falling, security concern without any basis for believing that an individual needs to do anything. But boy, was it a great source of material for 2018.

**Leo:** Oh, man.

**Steve:** Yeah. So today Microsoft updated their posting, saying, "While the phased rollout is in progress," that is, they're going to tippy-toe this out, this Retpoline deployment, sort of cautiously. They said: "While the phased rollout is in progress, customers who would like to manually enable Retpoline on their machines" - those with Windows 10, October 2018, 1809, with a Broadwell or older chip - "can do so," they wrote, "with the following registry configuration updates." And again, link in the show notes for the five of you out of the who knows how many listeners we have who are still at this point, like, oh, yeah, that's me. Go get it.

Meanwhile, Adobe ColdFusion gets an emergency patch. Last Friday, March 1st, Adobe released an emergency patch for their Java-based ColdFusion website development platform to close a vulnerability that was being actively exploited in the wild to execute

arbitrary code. So yes, emergency. So hopefully, again, if you're using ColdFusion, you are current with your email update notification list, and this is already old news to you, because this was a zero-day that they became aware of. The vulnerability allowed an attacker to bypass restrictions for uploading files. So to take advantage of it, the website had to be configured to accept executable uploads. So, okay. So that immediately, hopefully, disqualifies...

**Leo:** That's a problem right there.

**Steve:** Yeah. Now, on the other hand, there are places where you could imagine you could be allowing executable uploads for some reason where they would be sequestered and then could not be executed. The flaw allows an HTTP request to execute that uploaded file. Whoopsie.

**Leo:** Wow. Wow.

**Steve:** So not good. Really not good. All previous ColdFusion versions on all platforms are vulnerable to this flaw. It's CVE-2019-7816. I've got a link to their security advisory, which you just had on the screen a second ago. Adobe's summary said: "Adobe has released security updates for ColdFusion versions 2018, 2016 and 11 dot anything. These updates resolve a critical vulnerability that could lead to arbitrary code execution in the context of the running ColdFusion service. Adobe is aware of a report that CVE-2019-7816 has been exploited in the wild."

It turns out that an independent consultant named Charlie Arehart discovered this when he found it being used against one of his clients.

**Leo:** Ooh, bad.

**Steve:** Yeah. After figuring out what was going on, Charlie reported the flow to Adobe, along with a proposed solution. To their credit, and doubtless due to the bug's extreme severity affecting all appropriately-configured ColdFusion-based websites ever, Adobe had the fix ready within just a few days. So bravo for them getting on this immediately.

Bleeping Computer interviewed Charlie, who they quoted saying: "Getting folks to implement this fix is of critical importance." Oh, and Charlie did not disclose any additional details of the attacks since he didn't wish to help any attackers. However, he did tell Bleeping Computer that he believes that a skilled attacker will be able to connect the dots from Adobe's security bulletin and find a way to exploit the glitch. So knowing that the key is finding a site that will accept an executable upload, there is now a way then to generate an HTTP query of some sort which will execute that. And so ColdFusion has been around for a long time. It's one of those things that refuses to die. Like everything else that Adobe - except, you know, PDFs. We like those. But everything else, no.

**Leo:** This was also a problem with PHP is if you weren't careful about your directory permissions, you can execute PHP code in a directory, arbitrary code, and boom.

**Steve:** Yup. Yup. So for an interim mitigation, Adobe wrote: "Note: This attack requires the ability to upload executable code to a web-accessible directory, and then execute that code via an HTTP request. Restricting requests to directories where uploaded files are stored will mitigate this attack." Okay. Just update ColdFusion. Again, so if something prevents you from doing that, then oh, my goodness, yes, by all means don't allow directories where files are uploaded to be web accessible by any means. You certainly shouldn't. I mean, that's Web Security 101.

So again, ColdFusion 2018 update 2 and earlier; 2016 update 9 and earlier; and ColdFusion 11 update 17 and earlier. Basically, that is, all of ColdFusion is susceptible. So hopefully anybody, as current security really requires, anybody who is doing things with web-based systems needs to make sure that, you know, we talked about this with Drupal last week. Make sure that your email address that they have on file for you is correct and that alarms go off when they send you a security bulletin because right now the exploit window is - we are seeing how quickly bad guys jump on these things.

We have another instance of a self-signed certificate problem emerging and a collision with another AV. In this case it's Kaspersky. More than a month ago, since early February, or rather for more than a month since early February, Chrome users, okay, so people using the Chrome browser, which we know is most people in the world now, who are also using Kaspersky's AV in its default mode of performing secure connection filtering, which is of course what all of these AV systems that are offering this are doing now because otherwise they can't see anything coming and going from your system, for more than a month users have been getting and complaining about mysterious pop-ups from Kaspersky.

I have a link to the Chromium bug report and a picture of the pop-up. It says Kaspersky, this was both the free and the paid version, says "Cannot guarantee authenticity of the domain to which encrypted connection is established." The application shown is Google Chrome. A URL is bizarre-looking, it's a GUID, one of these hyphenated hex things which is long. The reason is given as a self-signed certificate. So any version of Chrome. In this case it was Windows. And the third-party software involved is Kaspersky. So says the Chromium bug report.

The bug report reads: "There's been a sudden increase in device discovery reports. Reviewing the reports indicated that it's common on all Windows platforms. And reviewing the logs show a commonality of cast channel" - meaning Chromecast channel - "authentication errors, which can often be attributed to antivirus or security software." Then they said: "In a similar timeframe, some discussions appeared on Kaspersky's online forums." And there are two links that are given in the Chromium report, "chrome-self-signed-certificate-cannot-guarantee" and then also "self-signed-certificate-issue-on-google-docs."

The person producing this bug report says: "I was able to reproduce the issue with Kaspersky Free, and confirmed with some external users using Kaspersky Total Security," which is the subscription-based system. To reproduce it: "Have Kaspersky software, either free or Total, installed and running on a Windows machine." Now, here is what's interesting. "Have a Chromecast device connected anywhere to the same network as the computer. Then open Chrome."

So he says: "Immediately when Chrome is opened on a network with a Chromecast device, a pop-up dialog appears from Kaspersky stating 'cannot guarantee authenticity of the domain to which encrypted content is established.'" Okay, you know, remember the user hasn't done anything at this point. They've just fired up Chrome to get ready to do something, and suddenly they're being shown this thing saying we're not happy.

"Even after clicking Continue on the dialog, the Chromecast devices do not appear in the Cast dialog." So Kaspersky is apparently not allowing this to happen or has already dismissed these devices. "Disabling 'Scan Encrypted Communications' in the Kaspersky network settings," which of course is the thing which is allowing Kaspersky to intercept and filter with its own certificate any attempted TLS connections, the bug report says, disabling the scan-encrypted connections, then allows device discovery to work, and the Kaspersky error dialog does not appear.

So what has been figured out, looking at this, is that when the Chrome browser is launched, it sends out a broadcast to the Ethernet broadcast on the local network, querying for any available and listening Chromecast devices. And it turns out that Chromecast devices may be present even when they are unknown to the network's user. For example, many recent smart TVs now include Chromecast built in, so that they're able to receive casts just as part of the service that the smart TV offers. Chromecast has a device discovery service which listens for anybody asking for it and accepts connections on TCP port 8009. And it will establish a TLS connection to a client connecting to it using its self-signed cert.

So when Chrome is started up, it sends out a "are there any Chromecast devices listening out there" on the local broadcast to the LAN. And any powered up and online Chromecast devices will hear the call and reply. Then the Chrome browser attempts to bring up a TLS connection to that responding device's IP at port 8009. In the TLS handshake, the Chromecast device sends a self-signed cert in order to encrypt the communications, and Kaspersky freaks out if it is monitoring all connections, not even to Chrome, but to the PC on which Kaspersky is installed.

So that's what's been happening. Kaspersky's complained that it's not easy to distinguish this event from everything else going on. But they have said that within a week or two they'll have a fix for it shortly. My guess, because there's really no downside to doing so, is that they will simply allow connections to be made from the user's computer to other devices on the same LAN, on the same network subnet, because that's going to be safe, and to make this port 8009 an exception and basically whitelist a self-signed cert warning on port 8009 when it's on the local net. There's no reason not to. You're not opening yourself to any security vulnerability because it's your own LAN. And you're establishing a TLS connection.

The fact that it's a self-signed cert is not a problem. I'm sure that Chromecast is doing the responsible thing, which many vendors, unfortunately, have not. We've talked about this, that is to say that Chromecast is creating a cert on the fly with a unique private key. That way there is no globally known private key that can be leveraged against people who have decided that they want to trust this self-signed cert.

**Leo:** Honestly, we see this crap all the time. This is just why you shouldn't use an AV.

**Steve:** Yes. I know.

**Leo:** It's dumb, overprotective security.

**Steve:** I know. I completely agree.

**Leo:** If you're going to use an AV, I think we all agree you shouldn't be using Kaspersky.

**Steve:** Yeah, that's a little bit of a stretch.

**Leo:** In Soviet Union, antivirus infects you.

**Steve:** None of our listeners at this point would think that I actually need any additional caffeine.

**Leo:** But would you like some anyway? Go for it, Steve.

**Steve:** But my vocal cords have dried out, so...

**Leo:** All right. Time to hydrate.

**Steve:** So Docker containers are having another problem. We've talked about these before, and unfortunately they're back. But we should back up a bit and talk about Dockers a bit, and I don't mean the pants, since we have never discussed them in any detail. And they're becoming increasingly popular and are therefore becoming an increasingly lucrative target for attack on the Internet. And this is again another one of these issues that should not be publicly exposed. We showed the picture of the nearly 4,000 Docker ports that Shodan was indexing as being publicly available.

Okay. So we all understand the concept of a virtual machine since they've been around for a long time. A Docker moves the encapsulation boundary over to the other side of the OS, is a way to think about this. In a virtual machine environment we have the so-called "hypervisor." It takes advantage of the amazingly complex hardware features of processors to create isolated abstractions of the hardware processor itself. Thus the term "virtual machines," you know, virtual CPUs. And once you have an abstraction of a processor, a virtual machine, then you boot an operating system onto that virtual machine to create an instance of a system which can then run that operating system's client software.

Okay. But think for a moment about how expensive this is in a cloud computing environment. Say that a given hardware system wants to run six separate tasks. Taking the virtual machine approach, the system's RAM is divided up into six partitions. An operating system instance is booted into each one. And then each one is given a task to run. The flexibility this offers is that the hardware could be - could be - simultaneously running six different operating systems, each running their own task.

But the reality in today's computing environment is that more stuff in the cloud is Unix or Linux based. So you're not actually running a totally heterogeneous OS environment. It tends to be homogeneous. So booting six redundant copies of the same Linux VM on a cloud computing hardware instance is very wasteful. And remember back when we were talking about Rowhammer attacks, how we learned that practical VM environments worked very hard to consolidate identical regions of memory.

So if you had six copies of Linux VMs, each running the same OS, a great deal of the memory that they're using is the same because it's the same OS. So, you know, same

drivers, same kernel, same a lot. And so the virtualization hardware allows those duplicate regions of memory to be consolidated so that each VM sees its own memory map and isn't aware that it's actually sharing that physical memory among other VM instances. The point is that there's been a lot of work, difficult work, which in this case exposed Rowhammer vulnerabilities as a consequence, but a lot of work to get back the lack of efficiency of running all of these separate VM instances.

Okay. So under this original VM-based cloud computing model, the encapsulation was the VM. What Docker does is it moves this encapsulation boundary onto the other side of the OS, whereas the VM model the encapsulation boundary was between virtual machine hardware and the VM's OSes. Docker places the boundary above the OS at the OS service level. So whereas a VM contains an operating system, a Docker container does not. The Docker container runs on or above an operating system to which has been added a Docker interface API.

So the Docker container doesn't have the OS in it. It encapsulates all of the various library and service dependencies and requirements of whatever the task or process is that the Docker is intended to perform. So this is like way more efficient in a cloud computing environment. The use of Dockers are becoming increasingly popular since they offer a much more efficient sharing of a hardware instance's resources. And the cloud computer has a single highly tuned instance now of an OS running, which is exposing a Docker API that allows it to host and run many independent instances of the so-called "containers." A container is sort of the equivalent of a VM in Docker land, a Docker container.

So the Docker container resident run-time module, that is, this thing which is running on typically a Unix or a Linux, is known as "runc," short for "run container." Runc is an open source command line utility designed to spawn and run containers and, at the moment, is used as the default runtime for containers with Docker. There's one called "containerd," as in "daemon"; Podman; Kubernetes that we've talked about in the past; and LXC, which is the Linux, sort of the Linux execution container offering.

Okay. So we have a CVE-2019-5736. It reads: "Runc through 1.0 release candidate 6, as used in all Docker before 18.09.2 and other products" - so runc is the problem - "allows attackers to overwrite the host runc binary" - whoops - "and consequently obtain host root access by leveraging the ability to execute a command as root within one of the Docker containers" if a new container with an attacker-controlled image is mounted, or an existing container to which the attacker previously had write access, that can be attached with Docker exec. And they write that this occurs because of file-descriptor mishandling relating to /proc/self/exe.

So what this creates is a Docker breakout security flaw which has been discovered in that runc container runtime, which allows malicious containers with minimal user interaction to overwrite the host "c" runtime binary to gain root level code execution on the host machine. So the maintainer of runc is a senior software engineer at SUSE Linux in Germany. And I won't go through his posting. But he posted details about the flaw and updates with a seven-day window before releasing a proof of concept. They were under pressure to produce a proof of concept because many people who were applying the update felt very strongly about the need to verify that the patch had done what was expected. So a proof of update was produced a week later, creating a relatively small window before the bad guys would have a running proof of concept that they could use against other Docker instances which were exposed.

Which brings us to what the guys at Imperva found, which was taking a look at Docker's public exposures on the Internet. All the big vendors responded immediately. Amazon runs Dockers. Google does. Docker themselves do. But of course it is something that Unix and Linux instances are able to run. So of course there are a gazillion of those out

on the Internet. And as we started off saying at the top of the podcast, nearly 4,000, or actually in some cases a little more than 4,000, were reachable and were believed to represent exposed Docker instances.

As I also mentioned earlier, it is supposed to only be bound to the localhost interface. I can't explain how thousands of these could be exposed publicly except maybe somebody wanted to have a Dockers instance that would be only on the LAN, and then their firewall was misconfigured so it got out. But you could also bind the server only to the local network, rather than just the localhost, but apparently that wasn't done. I guess the problem is it's just a numbers game. If you have enough instances of anything globally, you're going to find some which are misconfigured.

So the guys at Imperva first did a Shodan search to see what was available. Then they dug deeper and connected to the IPs that appeared to be advertising Docker to see what version of Docker was running. And they did find thousands of them still vulnerable. Then they dug even deeper and looked inside. They found that out of the nearly 4,000 IPs that were apparently exposed by the Shodan search engine, about 400 of them were still responding as Docker. The presumption is that other bad things may have crawled inside and closed the door behind them so they were no longer publicly exposed. But 10 percent of them were still exposed. And they said that on these unpatched Docker servers that had remained accessible, they found, not surprisingly, Docker images of cryptominers, as well as legitimate services and production environments.

So where possible, bad guys, given four weeks to - I think it was on February 11th this was first made publicly exposed, publicly disclosed. So it didn't even take a month before immediately cryptominers were set up on these machines, presumably, I mean, they may well be servers, which have some strong hardware. And so there is some hope, no doubt, in the minds of the bad guys that, if you can get a cryptominer mining Monero cryptocurrency on a strong big iron server platform, you stand to make some money. So Imperva summed it up yesterday in their disclosure, so that was just Monday the 4th, saying hundreds of vulnerable Docker hosts exploited by cryptocurrency miners.

Which brings us to sort of an unexpected but interesting story. We've been talking about Coinhive off and on for I think it's been a couple years. Coinhive, of course, is the company deeply in the gray. They pretend to be a well-intended service. The idea was that people who wanted to make money from visitors to their websites, rather than putting ads on the site, would host the Coinhive JavaScript in their web pages so that visitors to their website would download a link to the Coinhive JavaScript, which would then download from Coinhive the mining software, which while the visitor was moving around this cryptocurrency based or cryptocurrency financed website would spend a little time participating in a mining pool in order to generate some money in return for the short-term use of the visitor's processing resources.

Okay. If all of that was the way it worked, that would make sense, especially if a notice is put up on the web page saying, hey, we notice you're blocking ads. Would you mind clicking here, give us your permission to run a benign cryptocurrency miner on your web page while you're visiting our site. You say, okay, yeah, fine. Seems like a reasonable tradeoff. You say yes. That happens. Of course what happened was bad guys said, oh, I'm going to create a Coinhive account, and I'm going to inject this Coinhive mining script everywhere I possibly can.

So as a consequence we discovered, we talked about last year how there were routers, MikroTik routers were injecting Coinhive script into unsecured pages of the browsers behind the router every chance they could get, and all kinds of similar things. There were ads that were carrying Coinhive, so the ad was pinning someone's CPU while they were visiting a page, and that was causing a problem. Then of course there was pushback

from the browsers, who then started trying to detect whether cryptocurrency mining was occurring on the browser, and on and on and on.

Okay. Yesterday's blog post titled "Discontinuation of Coinhive": "Some of you might have anticipated this," reads the blog. "Some of you will be surprised. The decision has been made. We will discontinue our service on March 8, 2019." That's in three days. Today's the 5th. "It has been a blast working on this project over the past 18 months; but to be completely honest, it isn't economically viable anymore. The drop in hash rate - over 50% - after the last Monero hard fork hit us hard." Oh, and also it's worth noting there's another fork slated for the 11th, next Monday, which will further drop the rate. So that no doubt factored into their thinking.

They also said: "So did the 'crash' [in quotes] of the cryptocurrency market, with the value of XMR [Monero currency] depreciating over 85% within a year. This and the announced hard fork and algorithm update of the Monero network" - oh, yeah, they do mention it - "on March 9" - oh, it'll be on Saturday, March 9th - "has lead us to the conclusion that we need to discontinue Coinhive. Thus, mining will not be operable anymore after March 8, 2019. Your dashboards will still be accessible until April 30" - so the rest of March and all of April - "so you will be able to initiate your payouts if your balance is above the minimum payout threshold. Thank you all for the great time we had together."

So out with Coinhive and in, I imagine, with whatever takes its place. This will certainly put a kink in the cryptojacking enterprise, but I doubt this will be the end of it altogether. Recall that Coinhive had monthly revenue themselves, back in the heyday, of approximately a quarter million dollars a month at one point. And in terms of I guess what you would call "market reach," they enjoyed a 62% share of all websites using a JavaScript cryptocurrency miner. Which suggests there are other cryptocurrency miners. And if they can't use Coinhive, the bad guys will just switch to whatever they can use.

So maybe this is a sign of the fact that the cryptocurrency mining phenomenon was supported by that balloon that we had in cryptocurrency valuation, and that the expansion of mining, which allows this continual hard forking and revision of the algorithm of cryptocurrencies, in fact the algorithm change is further fighting back against ASIC mining in the case of Monero.

So it may just be that it's really no longer viable; that we're not going to be suffering this continual concern over mining injection into web browsers because it just no longer pays, that there are other ways for these bad guys to make more money than injecting cryptocurrency mining into the browsers of unwitting users. Despite the fact that at one point 200,000 MikroTik routers had been commandeered and used for cryptojacking campaigns in several different waves. So I imagine we'll see. I will certainly cover what happens moving forward.

**Leo:** Who do you think created Coinhive? You think that was just some graduate student in his dorm room?

**Steve:** Yeah.

**Leo:** That's the impression I get now.

**Steve:** Yeah. I really do think that it was - I think it was well intended. It was probably always misdirected because there was never a way to prevent its abuse. And he was

saying, set up an account with me, and you can have your visitors to your site mine Coinhive to generate some revenue for you. That's kind of a cool model. It was like, oh, okay. The problem is bad guys. There was no way to police it. There was no way to keep bad guys from injecting that same script, creating an account at Coinhive and then just spraying this script everywhere they could in order to get nonpermitted cryptocurrency mining happening. So I think it was kind of a good idea.

It would be interesting to know how much money the guy is, like, pocketing as he shuts things down and steps away. But it was just probably always a bad idea, only because there's just no way to do it, or at least he didn't have a way to do it in a way that did not allow it to be abused. I still think it's interesting. I mean, the idea of generating revenue for a site, I mean, I feel badly for Wikipedia. I donate every year when Jimmy Wales comes knocking, and he does. But then I want to support Wikipedia, but would it be a bad thing if my processor was used to generate some revenue for them while I'm looking at a Wikipedia page? That's a tradeoff I could make, rather than have ads on. I would really not like to have ads on Wikipedia. So, but the Wikipedia pitch every year is this is expensive for us to keep this stood up and maintained.

So it still strikes me as an interesting model, the idea that, because of the design of the cryptocurrency, it is still feasible for a CPU to mine, and it hasn't been, like as is the case in bitcoin, that's just gone. That's just all custom hardware now. They designed an algorithm with Monero which is hostile to that kind of scaling, ASIC scaling. And the idea of generating revenue while you're visiting a site with your permission, that really does seem interesting.

**Leo:** Yeah, yeah.

**Steve:** I think the way to do it maybe would be to install something on your computer so that you can mine efficiently, and then have a web standard where the browser is able to get your permission and then engage the miner while you are there, and basically the CPU work that you're doing goes to benefit the mining pool that has been assigned to the website that you're visiting, that sort of thing. So anyway, I can imagine some ways of it being done in a user-supported way that makes sense. This really was kind of a hack. And unfortunately it didn't really pay off in the long term. And boy, was it abused.

**Leo:** Yeah. That's the real tragedy.

**Steve:** Yeah. So we have a new version, I wanted to notify our listeners, of Wireshark. Wireshark has been around since the late 1990s. Back then it was named Ethereal. And boy, have I gotten my mileage out of this thing. It is, for those who don't know, Wireshark is the go-to utility for capturing and analyzing packetized network traffic. I've been using it, as I mentioned, for years. And things like ShieldsUP! and, boy, especially the very complex DNS spoofability test service would have been far more difficult for me to get built were it not for Wireshark. I absolutely used it to look at the way things were happening "on the wire," as they say.

It was originally named Ethereal, and then it got renamed in 2006 to sidestep some trademark issues because Ethereal was not available. It always relied upon an old-timer, a venerable packet capture driver known as WinPcap, which has also been around forever. And ShieldsUP!, GRC's service that we were talking about before, originally used the same driver. I used WinPcap in the beginning, until I later wrote my own kernel driver so that I could do more custom work down in Ring-0, back when I really needed per-packet efficiency.

So the big change, which was announced last Thursday with the release of Wireshark v3.0.0, is that the WinPcap driver has finally been abandoned in favor of a new driver called Npcap. Npcap is an NDIS v6 filter shim driver, meaning that it inserts itself neatly into the network stack in such a way that it's able to watch and inject network traffic without needing to worry about any adapter-specific details since those are handled down in the lower layers of the stack.

This Npcap driver is EV-signed so that the latest Win10 systems will allow it to slip in between their network layers. Version 3 also for the first time can capture localhost loopback traffic. As I was talking about anything that binds to 127.0.0.1, normally Wireshark has bound to network interfaces, and so you could only see traffic coming and going in and out of your machine. For me, that's been an inconvenience. So it's very cool that the new Wireshark will allow you to monitor the things going on that I was talking about before in the case of Docker, internal communications using the network protocol, but which never leave your machine. Wireshark 3 can now do that. So that's going to be very handy.

And because we have a lot of communications which is now encrypted, one of the other cool features of Wireshark, it's had this for a while, is if you give it the server's private key, it can decrypt TLS communications from its packet capture passively. And we've sort of talked about how that's possible. Remember that we talked about how the NSA is sucking up all of this network traffic which they cannot read. But if in the future the private key, even after it's expired, becomes available to them, they can go back and use the server's private key to decrypt past captured network traffic. This uses that same approach in order to allow probably a custom rolled server private key. You have to be very, very careful with your private key that it doesn't get loose.

But being able to bring up a service with a self-signed certificate that you've told the client service to trust, and then give the private key of that self-signed cert to Wireshark so that you can then see into the TLS traffic that is being transacted, that's just super handy. So anyway, I just - oh, and also in WiFi. In the past you had to have typically an AirPcap hardware WiFi dongle in order to sniff radio traffic, especially promiscuously sniff all the traffic that it was able to see. Not so anymore. This Npcap that the position of the Npcap driver in the stack means that it's able to capture 802.11 WiFi traffic out of the air without needing any special hardware. So I just wanted to make sure to put it on everyone's map that Wireshark is now at v3.0 and with lots of feature improvements, including it's able to decode many more network protocols than it was before. So definitely very cool.

And this was a neat trick. I'm not sure that I have a use for it, but it's another thing that I wanted to sort of add to our listeners' bag of tricks. This is from Bleeping Computer's founder Lawrence Abrams. He discovered that it was possible to use either Chrome or Firefox in a headless fashion, that is, no UI shown, to use the browser to render the image of a remote website's page by URL and save the page rendering to an image file, all with never launching the browser itself, which is kind of interesting.

In the show notes I've got a link to Bleeping Computer's post about this, or you could just go to BleepingComputer.com and read down through the chronologically posted items. Chrome and Firefox can take screenshots of sites from the command line. Anyway, basically the idea is you open up a command window, and you need to give it the path to Chrome, wherever Chrome.exe is located, then --headless --screenshot=, and then the path to the image.png, whatever you want to call it, and then the URL.

And when you launch this command, nothing appears to happen. It takes a minute or two or however long, I mean, hopefully not that long, and the cursor drops down and gives you a command prompt again. If you go look, and everything worked right, you'll find an image file which, if you then open it, is a picture of that URL as it would have

been rendered by Chrome. There are some other additions to the command line possible. You can specify the window size in width and height. You can tell it you don't want scroll bars to be shown and a few other things. And the same thing can be done with Firefox.

So anyway, I don't know specifically how this might be useful, but I just thought it was a real cool hack. Larry reported that it was quicker and easier to do this with Firefox than with Chrome, but both could do it. And it does provide you with a means - you can imagine maybe a periodic script to take a picture of a web page and then check to, like, check it for changes or who knows what. So anyway, just a very cool little hack that I thought was worth sharing.

Oh, boy. We've talked about the danger of DMA, Direct Memory Access, enabled interfaces. The first one we encountered was the venerable Firewire. And it was with some surprise that it was like, we learned that Firewire, cute and small as the form factor was, it's a nice little plug, it's a high-speed serial interface. And there is a sort of a meta command language that runs over that serial interface that allowed Firewire to directly transfer blocks of memory into and out of the machine. The idea was, whoever designed this was like, oh, wouldn't this be nifty if the Firewire peripheral, whatever it was that we plugged in, typically a streaming video device, a camera, who knows what it was back in the day. Probably an optical disk writer or something. If it could autonomously suck data out of the OS and/or send data back in, how fun.

Well, yes, except that we have talked about for years on this podcast that, if not protected, it opens up a system to, not a remote, but a very potent local attack. So it turns out that Firewire is known to have this problem. We have talked about how a very powerful successor to that, known as Thunderbolt, has the same capabilities. And it turns out that, even though there has been work done on mitigating these problems, somehow no one ever got around to actually turning them on, believe it or not, with the single exception of macOS. So bravo for macOS.

I have a link to a PDF in the show notes, and I will just share briefly this abstract of the research. I won't go into it any more deeply than that. But they wrote: "Direct Memory Access attacks have been known for many years." And indeed they have been, which makes it kind of a quandary how we're still so vulnerable to them. They wrote: "DMA-enabled I/O peripherals have complete access to the state of a computer and can fully compromise it, including reading and writing all of system memory. With the popularity of Thunderbolt 3 over USB Type-C and smart internal devices, opportunities for these attacks to be performed casually with only seconds of physical access to a computer have greatly broadened.

"In response, commodity hardware and operating system (OS) vendors have incorporated support for Input-Output Memory Management Units (IOMMUs), which impose memory protection on DMA and are widely believed to protect against DMA attacks. In this research, they say, "we investigate the state of the art in IOMMU protection across OSes using a novel I/O-security research platform, and find that current protections fall far short when placed within a functional network peripheral that uses its complex interactions with the OS for ill intent.

"We describe vulnerabilities in macOS, FreeBSD, and Linux, which notionally utilize IOMMUs to protect against DMA attacks. Windows uses the IOMMU only in limited cases, and it remains vulnerable. Using Thunderclap" - which is what they call their research and their device - "an open source FPGA research platform we built, we explore new classes of OS vulnerability arising from inadequate use of the IOMMU." In other words, all of our platforms now have it; and it is disabled, believe it or not, almost all the time.

They said: "The complex vulnerability space for IOMMU-exposed shared memory available to DMA-enabled peripherals allows attackers to extract private data (sniffing

cleartext VPN traffic) and hijack kernel control flow (launching a root shell) in seconds using devices such as USB-C projectors and power adapters." In other words, the mythical evil power adapter, 100% feasible. "We have now worked closely with OS vendors to remedy these vulnerability classes, and they have now shipped substantial feature improvements and mitigations as a result of our work."

And I have in the show notes a table from their research showing Windows 7, 8.1, Windows 10 Home and Pro, 10 Enterprise, two versions of Enterprise, macOS 10.10 and so forth, Linux, Ubuntu, Fedora, Red Hat, Linux, FreeBSD, PC-BSD. The sobering thing is there is a "can use" IOMMU. It is not available for Windows 7, not available for Windows 8, not available for Windows 10 Home Pro. Only available for the others. So Windows 10 Enterprise has it, macOS, the Linuxes Ubuntu/Fedora/Red Hat, FreeBSD, PC-BSD. But it is only enabled by default on one OS, and that's Mac. In other words, even where it is available for use, it is not enabled. And then the table goes into the details of their research further.

But what this says is that right now today, if you have typically a laptop was where the vulnerability would be, with Thunderbolt 3 available to an I/O connector, probably a laptop, as I said, with USB Type C connector, someone could plug something into that machine while it's running and steal its secrets in a few seconds. And what this research demonstrates convincingly. So their mitigation recommendation is mine. If you do not know that you need Thunderbolt 3, for example, probably Alex Lindsay, who's doing crazy video stuff...

**Leo:** He needs it, yeah.

**Steve:** Needs it and knows he needs it and is using it. The rest of us, eh. I have no need for it. There's nothing I'm doing that is, you know, that I'm attaching a Thunderbolt 3 peripheral to. Yes, USB 3, for sure. That's different than Thunderbolt 3. The point is you can almost always disable Thunderbolt in the BIOS.

**Leo:** I'm confused because I guess - oh, but Windows 10 does, well, see, I use Thunderbolt 3 on Windows 10 Home, I thought, but maybe not. Or Pro. I'm sure I do.

**Steve:** Well, yes. And so what they're talking about is that it is vulnerable. So the IOMMU is the gatekeeper, essentially, for Thunderbolt. But it allows...

**Leo:** But it says Windows 10 does not support IOMMU. Windows 10 Home or Pro.

**Steve:** Correct. So Home or Pro...

**Leo:** Can't support Thunderbolt 3?

**Steve:** No, has Thunderbolt 3 and is vulnerable.

**Leo:** Oh.

**Steve:** The IOMMU is what provides...

**Leo:** Oh, it's like Retpoline. It's that thing all over again. It's a double-negative product, okay.

**Steve:** Yes, exactly. So the IOMMU is the gatekeeper that allows Thunderbolt to be used safely.

**Leo:** So on macOS when I use Thunderbolt it is safe, then.

**Steve:** Yes. And only macOS. Only on macOS. Yup, exactly.

**Leo:** Okay. So don't use Thunderbolt 3 on Windows 10.

**Steve:** Well, yes, exactly. If you don't know you need it, and the only one we know who does is Alex Lindsay...

**Leo:** And he doesn't use, I promise you, doesn't use Windows 10.

**Steve:** Right. Oh, that's a good point. So he's already clear because he's on macOS. So I would say just reboot into your BIOS. Or maybe you're on a mountaintop somewhere, or in a cave.

**Leo:** Somebody would need physical access; right? DMA requires physical access.

**Steve:** Exactly. It is a physical access vulnerability. So let's also remind ourselves that nobody has ever been attacked ever in the history of man by Spectre and Meltdown, as far as we know, despite all the ink that it has been given. Similarly, if there's nobody that's going to come along and stick something in your port that might be malicious, you don't have anything to worry about, either. But if you might be subject to a targeted attack, or even just a brush-by attack in an airport, where you're using a laptop, or if you're a high-value target, hopefully your enterprise management has already disabled Thunderbolt 3. It should not be enabled on your device unless you know you need it.

**Leo:** Wow. I had no idea that it was that risky.

**Steve:** Yeah, it is. It is a port into your operating machine's memory that would allow something to suck it out and take your keys, your BitLocker keys, your encryption keys, anything that is statically available in RAM at that instant. Or plant things. It's able to inject something into your system instantly. Yeah, so remove it. Disable it unless you know you need it. It's one of those things where it's like, oh, look, Thunderbolt 3, isn't that wonderful. Yeah. And when you buy something that needs it, then turn it on. Until then, no. Leave it off. Turn it off.

**Leo:** Wow. Good advice. Okay. All right. Ready to wrap things up with Steve Gibson.

**Steve:** Yeah. So we are seeing a lot of bot-based credential stuffing, "credential stuffing" being the new jargon that we've talked about now a couple times. Once upon a time we had what is now the quaint image of a hacker in his basement, repetitively trying to log into some target victim's account by guessing their password; right? Typing in candidate after candidate attempt, one at a time, over and over, until, hah, what do you know, I'm in. That evolved into an automated brute-force attack against someone, first running through dictionaries of commonly used passwords and keyboard keystroke walks and eventually getting down to trying every possible password.

Then we had website-based database breaches where hundreds of thousands of usernames and hopefully hashed passwords were disclosed. The bad guys would then use high-speed ASIC-based hashing rigs to reverse the hashes in bulk back to their original textual input for use in impersonating the people whose passwords were unlucky enough to be reversible. And now today we have the latest evolution of the so-called credential stuffing attacks, where fleets of bots, increasingly composed of code loaded into compromised consumer routers, are fanning out across the Internet, not only to replicate themselves, but to launch patient and widely distributed username and password guessing attempts against Internet-facing websites.

Akamai issued a report. I won't go into it in great detail, but there were some summaries of their stats were interesting. I do have a link to the PDF in the show notes. And they covered three different issues. This story is the first of those three. They said: "All three of our stories in this issue of the State of the Internet Security Report are about things most organizations are not examining." They said: "Whether the cause is that organizations don't perceive some issues as important to their environment, if they don't have tooling to monitor these issues, or if the resources to monitor this traffic are not available," they say, "this traffic is often being overlooked.

"Although organizations examine the traffic generated by botnets, without specialized tools that traffic is often treated the same as any other type of network activity. There are very few places where this is more dangerous than in the retail sector," they say, "where botnet creators and retail defenders are playing a multidimensional game, with real money on the line." Akamai wrote: "Our team looked at All-In-One (AIO) bots and considered them in the context of the billions of credential abuse attempts," they say, "that we see on a monthly basis."

They said, okay, so here's some numbers. Between May 1st and December 31st there were 10 million - no, sorry, whoa, I got my zeroes off - 10 billion with a "b," 10,000,588,772 credential stuffing attempts in the retail industry detected on Akamai's network. In other words, they're looking at their traffic. Their network is widely distributed and ubiquitous. So they detected more than 10 billion username and password guessing attempts against retail industry partners over their network.

They said when that's expanded to all other customer industries, so that is just 10 billion on the retail industry, when they expand it to all other customer industries, Akamai detected, okay, just shy of - I'll keep the numbers short because no one cares about all these digits - barely shy of 28 billion. It was 27 billion, 985 million blah blah blah credential abuse attempts over that eight-month period. So that works out to more than 115 million attempts to compromise or log into user accounts every day, 115 million a day.

They say: "The reason for these attempts is not complex. The malicious actors responsible for them are looking for data such as personal information, account balances,

and assets; or they're looking for opportunities to cash in on the online retail market that's expected to hit 4.88 trillion, online retail market at 4.88 trillion by 2021."

The credential stuffing attempts, as Akamai refers to them, logged by Akamai are automated, thanks to bots. Bots can represent up to 60% - six zero, we've talked about this number before - 60% of overall web traffic are not people clicking links and looking at web pages, but are now automated thingies. But less than half of them are actually declared as bots, which is often the case. A bot only, for example, Googlebot and Bingbot I often see cruising around my servers. Well, they're declaring themselves as bots. But many times a bot wants to look like a user clicking a link on a Chrome or Firefox or IE browser, so they pretend to be users clicking links on browsers.

**Leo:** Is this for click fraud, a lot of it?

**Steve:** Yeah, yeah. And just because you wouldn't expect Googlebot to be logging in as a user. So you could very easily prevent a self-declared bot from successfully logging in with a username and password. But whereas you'd want to present a non-bot appearance if you're trying to use this credential stuffing attack. So anyway, and they said that not only are less than half of them declare themselves as bots, many bots are not malicious. They're for good purposes. They're like checking to see if a web page has been updated. Maybe they're looking at prices on other websites. Who knows what they're doing?

Anyway, Akamai said: "For criminals, credential stuffing attacks are a numbers game. They're counting on the fact that people recycle their passwords across different accounts. When this happens, a compromised set of credentials from one website quickly translates into dozens of others." The point being that essentially they've become sophisticated enough to take advantage of everything that they see on a successful attack immediately gets relayed across the industry, looking for other places the same credential can be used to log in. So again, it's one of those, they're taking advantage of every hint and clue that is available.

Oh, and Akamai said: "Consider the 116 million accounts compromised during the LinkedIn data breach. Using this list of email address and password combinations, criminals targeted dozens of other websites in hopes that people were using their LinkedIn credentials elsewhere. These credential stuffing attempts led to several secondary account takeovers. This is why" - and this of course directly speaks to what you were saying about LastPass. "This is why security professionals stress the use of password managers," writes Akamai, "as well as the use of long and unique password strings for each website." And of course that has also created a dependence on password managers because I can't log in anywhere. I don't know any of my passwords anymore.

**Leo:** Yeah, yeah.

**Steve:** So they said: "The battle against credential stuffing isn't an easy one to fight. When asked, 71% of the respondents to an Akamai survey conducted by Ponemon Institute said that" - so 71% of respondents said - "that preventing credential stuffing attacks is difficult because fixes that prevent such action might diminish the web experience for legitimate users." In other words...

**Leo:** Well, we know they do. Look at those silly, god, it drives me nuts now, the new ones, because you're basically - you know what you're doing. You're teaching the Google autonomous vehicle unit how to recognize cars and storefronts and crosswalks.

**Steve:** That's right.

**Leo:** And it pisses me off. They're using, I mean, my cycles for their benefit. But I guess they have to with that many bots out there.

**Steve:** They said: "On average, organizations report experiencing 12.7 credential stuffing attempts each month."

**Leo:** Wow. So now I understand why, I mean, I couldn't figure out why do these sites care if I'm a bot or not? But now I know, yeah.

**Steve:** Yeah, yeah. With each attempt, each of those 12.7 on average credential stuffing attempts targeting 1,272 accounts. So they said: "The reflexive action to just block the bots responsible for these attempts outright makes sense at first," Leo, as you said. "But such a move," writes Akamai, "might cause serious harm to the business if legitimate customers are impacted."

They said: "The same survey revealed 32% of respondents lacked visibility into credential stuffing attacks," meaning they don't even know it's happening. "And 30% said they were unable to detect and mitigate them. When asked if their organization had sufficient solutions and technologies for containing or preventing credential stuffing attacks, 70% of those responding said their organization was lacking when it came to such defenses." So this is something that they're just kind of ignoring and hoping for the best and hoping that their customers secure themselves against these sorts of attacks, sort of saying, well, you know, we're not really looking at that too much.

Okay. Posted over in the SQRL Forum, CosmaP said: "Hi. My phone committed suicide yesterday." And he has a frowny face. He said: "Fortunately, my provider, EE in the UK, was on the ball, and I received a replacement today." He said: "Great customer services, a win from EE." He said: "Long story short, reinstalled," he said, "well, I am still reinstalling all the apps and came to the SQRL app. Installed the app." And he has an Android phone. He said: "No problem. Imported my identity." He means his SQRL identity. "No problem. Entered the rescue code, no problem. Signed into the forums using my SQRL password, no problem. All smooth as silk, and I am back operational." He says: "It helped that I have all the required info in one place." He says: "Job's a good'un. Regards, Cosma."

**Leo:** What info would he need to keep to reinstall his SQRL account? Is there a QR code or...

**Steve:** Yes. When you create your identity, you print a page, hopefully. You can store it as a file, but it's better if you just print it out because we know that paper is one medium with great longevity. And that's all you need. That is your one, hopefully for your entire life, identity for SQRL. And so when you want to bring up another device, or in his case his phone croaked, so he got a new phone. He just let the Android client see that piece of

paper. And then you can export your identity with or without your password. And it's slightly safer to export it without your password because then you need the so-called rescue code, which is much higher - because it's 100% entropy, it's not something any user creates. Users, as we know, cannot come up with anything random.

So he apparently just chose for maximum security to export his identity to paper without his password, which is best for like archival storage. But that meant in order to use it he had to use the so-called rescue code in order to decrypt it for his phone. And what we're going to do, Leo, when we get together, we're going to have a SQRL party in your studio, and we're just going to turn a bunch of cameras on and let them run and do all of this. You, Lisa, Jason, Mike Elgin if he's around, I hope he will be, and I will just go through all this and do it all candid and cover all the what-ifs and everything else. I think it's going to be perfect.

Oh, and I got an interesting observation from someone, I guess he's a listener. Oh, yeah. His name is Jeff Root. And he wrote: "SQRL's friction is its lack of friction." He said: "I've had the SQRL client on my Android phone for a while, but nothing to use it with. After watching SN-703, I decided to try out the Forum login. And it was anti-climactic. You've been teaching us for years," he says, "that convenience and security are opposites. The entire security community has been in agreement on that. But SQRL, by being incredibly convenient and easy, appears insecure because of that. And so the biggest impediment to SQRL adoption may be that SQRL has zero friction."

Well, I thought that was kind of a fun observation. Actually, I think this is why it's going to succeed. I mean, when you experience it, you really do have a thought of, like, and this is secure? I mean, because it's so easy. I mean, and, see, I think that's the key is that so far the way we've responded to the problem with usernames and passwords is by just putting people through more hoops. I mean, I'm now constantly having to go to my one-time authenticator and look up the six digits in order to log into this or that or the other. And if it's near the end of its expiration, then I kind of have to wait for a new code to be emitted so that I'll know that I'm able to copy those digits over into the form that's waiting. And sometimes I'm having to switch pages. It's a pain on an iPad because of having to switch back and forth between apps and go through all this.

So far, our solution has been adding factors and making this a much bigger problem. So Jeff's point is we've taught everybody that, if you really want security, you've got to do more things. Well, what we're going to see with SQRL is, because security was designed into it from the beginning, the security is a given. But its ease of use, I think that's what is going to sell it to users because users don't care about security. I mean, they're, like, grumbling about having to have different passwords now for all their different websites and having to have, sorry LastPass, a password manager.

LastPass is never going to go away. Usernames and passwords are never going to go away. But I wouldn't be surprised if LastPass thought, hmm, maybe we ought to build SQRL into LastPass because that could certainly be done, too, and then we'd have the best of all worlds.

Anyway, we will be talking about SQRL both on a TWiT Live Special and certainly more moving forward. But I do think, when you see it, Leo, when you have that experience, it's like, wait. This could be the way we log in everywhere someday? Uh-huh.

**Leo:** You know, it's not two-factor. You don't need a password at all.

**Steve:** No.

**Leo:** It's your password.

**Steve:** Well, it's not even - you don't even need a username. See, that's the thing. All these other things you need to identify yourself. SQRL identifies you and confirms your identity. It's a zero-factor.

**Leo:** That's nice, yeah.

**Steve:** Yeah, I mean, it is. Oh, it's freaky. Also I did want to make a note. The developer of the Android client asked for some help with his user experience, the so-called UX. So if we have any user experience experts who would like to volunteer some time and, for example, would like the Android client which Daniel is working on to work the way they think it should, he would be happy to have the client work the way an expert thinks it should. But we need the expert to tell him. I have a SQRL feedback form, GRC.com/sqrl/feedback.htm. Drop me a note. I will get back to you with a way to contact him.

The way things are going, we have about, I think I looked, it was like 657 people now members in the SQRL Forum, which is just right. I mean, I'm not inviting everyone there yet because a flood of new users would not help anybody. The clients are moving forward. I'm working now, I've got the introduction Q&A, the user Q&A finished. I'm now working on the what-if. What if this? What if that? But what if this, and what if that, and what if that? So that's all getting fleshed out.

So I'm still putting the user-facing content together, the goal being that, once everybody does get invited there and come and play, it won't be a disaster. But for that to be the case, most of this has to be self-serve. So I'm working on all the self-serve stuff. But the clients are still - Jeff's iOS client and Daniel's Android client, lots of people are using them, but they're still not feature complete. And I'm delighted that Daniel is saying, hey, I would welcome assistance with the user experience side. So if we have any people who have experience developing UI stuff, GRC.com/sqrl/feedback.htm. Drop me a note, and I'll put you in touch with Daniel. And everybody would be very thankful for having a better client experience.

I love this little bit of miscellany. Remember I used the term, I said, whoa, that was a doozy. And I didn't know if D-O-O-S-E-Y was like, was that really a word? Matt in London knew where the word came from. He sent me a note saying "Duesy as in Duesenberg." He said: "Hey, Steve. I heard that a Duesy [D-U-E-S-Y] is named after the Duesenberg car that was so expensive that no one could afford one. Hence slang for a magnificent failure." And I thought that was a kick. Where is that thing? Wow, that's a Duesy. Well, turns out it's like, yeah, from the Duesenberg.

And I got an interesting note, Ralph in New York City. The subject was "SpinRite still working after all these years." And he didn't mean on one drive. He said: "I have a LAN with two WiFi high-def security cameras on the 2.4GHz band. They are recording to a USB 3 120GB SSD plugged into the router. Both cameras frequently stream together, but many times only one or the other would record a file, and there were random freezes on many of the files during playback. I was suspecting a bandwidth issue until I ran a Level 4" - that's the deep pass - "pass of SpinRite on the SSD." And this was really interesting. "Watching the real-time screen, I could see random pauses, retries on reads and writes. After SpinRite completed, both cameras happily record at the same time. This won't be a surprise, but SpinRite REALLY" - he has in all caps - "works."

And so there's something to think about. The real-time screen on SpinRite actually shows you the data. It just flickers on the screen. But you can easily - and you actually can recognize, like, oh, look, there's my name. It's like showing you the actual raw data in that drive that SpinRite is working on. But on a drive which is not responding very well, there will be visible pauses while the drive and SpinRite negotiate this data that it seems to be having a problem with. And so he was seeing that on an SSD. And again, running SpinRite on it fixed the problem. This is why I've talked about SpinRite beyond the 6.x series. It's clear to me that it has plenty of life left in it because, even if drives stop spinning, although the name will be a problem, there's still going to be a use for it.

Two bits of closing-the-loop feedback. Fresher in the U.K. sent me a note: "Where the 'f,' he said, is Jeff's iOS SQRL client? Mentioned in the last Security Now!, where is it? It's not in the app store, so what are we supposed to do? You didn't make it U.S. only, I hope." Well, first of all, no, because Jeff Arthur is in the U.K. also, Fresher. But this is to my point. People who want it enough are able to find it. And people have wanted it enough. But at this point we have enough people, I mean, a flood would not be - it wouldn't be good for anybody. We have enough people working with these things and participating and helping to flesh out content. So Fresher, if you want it, and you're a clever person, you could find it. But it is not available in the app store. So enough said.

Neil Taneja in Chandler, Arizona, asked about drive mounting and unmounting. He said: "You mentioned a few Security Nows ago about how you mount and unmount drives automatically for your backups to protect them. How are you doing that?"

So in Windows, fire up a command prompt and give the command "mountvol," M-O-U-N-T-V-O-L. What you will see is a list of mounted volumes by drive letter that we're all used to seeing, C: and D: and Z:, whatever, and then a series of \\?\Volume, then open curly brace, and then one of those GUIDs, and then closed curly brace. That's the volume ID by which Windows identifies the volume. And the volume has been mounted when it's been associated with a C:\, a drive letter. Mountvol also, if you do mountvol/? you get a list of things, and you are able to create an association, that is to say, mount one of those and delete the mounting, which is to say unmount one with a /d command. So, Neil, that should give you everything you need to build that into a little script and allow drives to appear and disappear from your system in order to keep the drive offline when it's not in use.

And lastly, to our listeners who are - or maybe, you know, well, yeah, we know we have listeners of all ages. We have people who say, hey, how do I get started in security? I want to consider a career. What do I do? Or maybe you've got some free time. Maybe you're living with your folks, or you're in high school, and you're thinking it'd be fun to see if you could earn some extra cash on the side. It is truly possible to have a career, if you're good, as a bug hunter, and getting bug bounties.

It was a picture that was covered in HackerOne's posting. We have a picture of 19-year-old Santiago Lopez who has just crossed the $1 million mark from purely - $1 million U.S. He is the first bug bounty hunter millionaire, just from finding and reporting security vulnerabilities through HackerOne's bug bounty program. I noted at the top of the show that we don't know if maybe he's made actually more money than that because he's sold some really tasty ones to Zimperium. Who knows? He's got an interesting Twitter feed I would commend people to go poke at. I've got a link in the show notes also.

And last Friday, March 1st, the BBC ran an interview with Santiago titled "How One Teenager Is Making Millions by Hacking Legally." Their little summary said: "This is 19-year-old Santiago Lopez from Argentina." I have a problem pronouncing that.

**Leo:** Argentina.

**Steve:** Argentina. Gee, yeah. I've got one more syllable in it. Argentina, thank you, Leo. "He's the first millionaire bug bounty hacker, which means he gets paid" - this is the BBC talking. Of course we all know what they are - "gets paid to find glitches in the software of some of the world's biggest companies. Mr. Lopez made his money on the world's biggest ethical hacking platform, HackerOne. BBC News's Joe Tidy has been to see how he spends the money." Thus the BBC story that I think our listeners may find interesting. And I have a link, as I said, to that in the show notes.

And also HackerOne did a report published on February 1st, so a little over a month ago. They said: "Today the HackerOne community hit $45 million in bounty payouts. Join us as we celebrate the hackers who are making the Internet a safer place every single day. The party is going to last the whole way to a history-making $50 million in bounty payouts." And in this posting that I have, or their PDF that I have a link to, they started off by defining "hacker." So they show "hacker" and how it's pronounced phonetically, declaring it to be a noun. And for their definition, which I really like, they said: "One who enjoys the intellectual challenge of creatively overcoming limitations." And I think that's a great definition of a hacker, "one who enjoys the intellectual challenge of creatively overcoming limitations."

And again, I would commend this report to our listeners who might be interested. They have a lot of bios and details about the hackers. I'll share just the top of it. They said: "Welcome to the age of the hacker. Hackers are heroes. They are in it for the good, and there is more opportunity than ever before. We share some of their stories and celebrate their impact in this, the third annual Hacker Report. The Hacker Report details the more than 300,000 individuals that represent our hacker community today. It highlights where hackers live, what motivates them, what their favorite hacking targets and tools are, where they learn, why they collaborate, and much more.

"In 2018 alone, hackers earned more than $19 million in bounties, almost the entire amount awarded in the years prior combined." So it's on a ramp. "And while the most successful find it very lucrative, it's about so much more than money. Many are finding career-building opportunities through bug bounties, with companies hiring from within the hacker community at a faster clip than ever before." And that's a point I've made. There are some very beautiful pieces of work where I've thought to myself, boy, you know, if I were in the hiring business still, I'd ask this person for a job. They are good. They say: "Companies are utilizing bug bounty reports and hacker engagement as an enhanced resume of proven skills that will impact company goals and security efforts from day one.

"The generosity and camaraderie of hackers continues to impress, with more emphasis than ever before on education, collaboration, and giving back. As hacking grows in popularity, training continues to be a focus. With more than 600 hackers registering to join the ranks on any given day, in-depth training modules such as Hacker 101 Capture the Flag challenges are in demand." They say: "This past year we saw incredible individual performances such as hackers earning $100K for one vulnerability, and the first hacker [Santiago] passing the $1 million milestone. We also saw unmatched collaboration, like hackers acting as teams to report over 250 valid customer vulnerabilities."

They say: "Hackers represent a global force for good, coming together to help address the growing security needs of our increasingly interconnected society. The community welcomes all who enjoy the intellectual challenge to creatively overcome limitations. Their reasons for hacking may vary, but the results are consistently impressing the growing ranks of organizations embracing hackers through hacker-powered security" - I like that - "hacker-powered security, leaving us all a lot safer than before."

And they did note that top earners, top hacker earners can make up to 40 times the median annual wage of a software engineer in their home country respectively. So I think what we're seeing is we're seeing a sea change here, hacker-powered security. And, you know, if you've got "skillz" with a "z," I think you ought to consider it.

**Leo:** Where does HackerOne gets its money from? Do private enterprises hire them?

**Steve:** Yes. So bounties are posted by, like, GE and Tesla and others to say we are formally inviting people to try to find problems in our products. And, if found, we will pay.

**Leo:** Yeah, but why go through HackerOne? Why not just go directly to GE? Does HackerOne keep a cut?

**Steve:** No, well, they do keep a small cut. But they're a clearinghouse. And so they've got the hackers registered with them, and they're able to then post opportunities of, you know, here are things that GE would be interested in having you attempt to find vulnerabilities in, and here are the payouts from GE on various classes of vulnerabilities that you find.

**Leo:** Got it. And they keep 5% or something for that service.

**Steve:** Exactly. Exactly. And so they're like the clearinghouse, and they put the hackers together with those offering bounties to have, basically, hacker-powered security, have their security tested and improved. I just think it's a win.

**Leo:** It's a brilliant business idea, actually, on their part.

**Steve:** Totally see, yes, totally see that for the right kind of guy, who's eclectic and doesn't want to work for the man and has faith and confidence in their own skills, it's something you could start part-time while you're in high school or going through college, see if you've got what it takes. And just like this podcast that is not running out of material, lord knows, we're at the two-hour and nine-minute mark at this point, so it is very clear the world is not going to run out of bugs ever.

**Leo:** Very nice. And that concludes the thrilling, gripping edition of this week's Security Now!.

**Steve:** Indeed it does.

**Leo:** I would like you to spend some time on a future show on how to learn how to do this kind of thing. We've talked about one of the techniques people use, fuzzing, but there must be other kind of standardized techniques. I think it would be interesting, I mean, obviously you can't teach us in a two-hour show how to become a white hat hacker. But maybe some resources and places to look to learn that skill.

**Steve:** Will do.

**Leo:** Yeah. I think it'd be very interesting. You'll find Steve at GRC.com. Lots of great stuff there, including SpinRite, the world's finest hard drive recovery and maintenance utility. Get yourself a copy. If you've got a hard drive, you need SpinRite. You can also get this show...

**Steve:** Or even an SSD.

**Leo:** I think I'm going to call them all hard drives.

**Steve:** Okay.

**Leo:** Right?

**Steve:** Mass storage. Mass storage.

**Leo:** Mass storage. I mean, a hard drive just means not a soft drive. I don't know what that means. It means a hard place you store your stuff.

**Steve:** Yeah. Okay, good, yeah.

**Leo:** Yeah. It doesn't imply spinning, or does it? Hard drive. I think probably it's a good idea to distinguish that from an SSD.

**Steve:** Well, we are seeing that SpinRite is valuable on SSDs. We keep getting reports, so yeah.

**Leo:** You'll also find great free stuff. He gives away a lot of stuff at GRC.com. All you need to know about SQRL.

**Steve:** Everything else is given away.

**Leo:** Clues to the secret hunt for the iOS app. All of that. I don't know if there's clues there or not. He also has this show, and he has audio of it, and he has a very nicely written transcription of every word so you can read along as you listen. It's useful for searching, as well. GRC.com.

Our website, TWiT.tv, has the show as well, audio and video, TWiT.tv/sn. You'll find it on YouTube, too. It's everywhere. In fact, you could subscribe with your favorite podcast client, and that way you'd be guaranteed to get the episode, every episode, the minute it's available. We will come to you. We will deliver, hand-deliver, or CacheFly will, hand-deliver a copy to your door.

Steve, thanks so much. Have a great week, and we will see you next time on Security Now!.

**Steve:** Thanks, buddy.