
# Exim Under Siege

**Description:** There were several significant stories this week. We have a new DRAM problem called "RAMBleed," news of a Linux server kernel-crashing flaw in TCP, and the occurrence of the expected attacks on Exim email servers - not to mention last week's Patch Tuesday, a Bluetooth surprise, and another useless warning about the BlueKeep vulnerability. Microsoft missed a 90-day Tavis Ormandy deadline. We have a good-news GandCrab wrap-up, Yubico's entropy mistake, a bit of post-announce SQRL news, and a favorite iOS security app. We selected as our title story the attacks on Exim mail servers so that we can talk about the other disasters, which are still pending, next week!

High quality (64 kbps) mp3 audio file URL: http://media.GRC.com/sn/SN-719.mp3
Quarter size (16 kbps) mp3 audio file URL: http://media.GRC.com/sn/sn-719-lq.mp3

SHOW TEASE: It's time for Security Now!. Steve Gibson is here. RAMBleed, the latest Rowhammer attack, is on the docket along with SACK, a new problem that's been in Linux for years. And then we'll look at that Exim server flaw he was talking about last week and how it's starting to spread. It's all coming up next on Security Now!.

**Leo Laporte:** This is Security Now! with Steve Gibson, Episode 719, recorded June 18th, 2019: Exim Under Siege.

It's time for Security Now!, the show where we cover your security online with the guy right here, Mr. Steve Gibson of GRC.com, our amanuensis. No, I'm the amanuensis. You're the man in charge. Hi, Steve.

**Steve Gibson:** Hi, Mom. I mean, Leo. Hi, Leo.

**Leo:** Is Mom watching from high above? From heaven?

**Steve:** I wonder what kind of reception you get up there. I don't know.

**Leo:** Well, it should be crystal clear.

**Steve:** Okay.

**Leo:** It would be the ultimate in disappointments if you're in heaven and the TV reception stinks.

**Steve:** Wouldn't that be? Or you only had like the major network...

**Leo:** Four channels?

**Steve:** Yeah, like in the old days.

**Leo:** God's too cheap to buy cable, yeah, yeah.

**Steve:** And then about 2:00 a.m. this American flag came up, and the jets fly overhead, and then it, like, shuts down until - I don't know why they did that because they had to come back on at, like, 6:00 a.m.; right? So it's like only...

**Leo:** Yeah, what are they doing? Are the dusting the transmitters? Why do they shut down?

**Steve:** Probably some FCC regulation or something where they had to...

**Leo:** I mean, isn't it worse? It's harder on the transmitters to shut them all down and turn them back up.

**Steve:** Yeah. Oh, I'm sure they couldn't turn those things off. Back then you had tubes; and you had to, like, baby the tubes.

**Leo:** You'd have to bring up the plate voltage first.

**Steve:** Do you remember when the TV would go on the fritz? You and your dad would unplug the TV. And then, like, hopefully you'd wait a while because the CRT could function as a big capacitor. And then you'd take the back off, and it had a - what did we used to call it where, if you took the back off, the power cord came with it so that it couldn't be plugged in with the back on. There was a name for that. I don't remember. Anyway, and then there was like all these tubes.

**Leo:** A dead man switch.

**Steve:** And so you'd reach in - it was that kind of thing, but there was a specific name. Anyway, and you'd reach in, and you'd kind of gently, kind of like in a rotating motion, pull each of the tubes out, and you'd carefully put them into a bag, and then you'd go down to the drugstore, where...

**Leo:** Oh, yeah. They had a tube tester.

**Steve:** ...where the tube tester was.

**Leo:** I forgot about all that.

**Steve:** Yup. And you'd stand up on a stool because at that time you were too short...

**Leo:** You were short.

**Steve:** ...to reach. And your dad would hand you the six NVT. And so then you'd turn the dial to look up the six NVT. And then there was a whole array of switches. You just had to slide all the switches into the right positions. And then you'd plug the tube in, and you'd wait, and you'd see it begin to start glowing. Then you could kind of see the red coming up through the plate. And then you'd press the button to test it. And this meter that had, like, red and then yellow and then green, like the needle would go hopefully into the green. But if you were having a problem, you were looking for it. So you'd hope to, like, find it. Or you'd put the tube in, and you'd wait, and you'd wait, and it was like, hmm. And when you waited long enough, you'd look closer, and there would be no red glow. And it's like, the heater burned out, and so...

**Leo:** We found the bad tube.

**Steve:** You'd found the bad tube.

**Leo:** Oh, my god. See, this is a little Father's Day memory. I can see how you got to be a geek. Your dad did this with you, huh?

**Steve:** He encouraged it. Like I was tugging on him. I said, "Come on, Daddy. Let's go, let's go." I mean, so he was fulfilling my urge.

**Leo:** So awesome.

**Steve:** I don't know where it came from, but, you know.

**Leo:** So awesome.

**Steve:** I definitely had some...

**Leo:** Well, now we know how this all happened.

**Steve:** He was an enabler, an early enabler of this podcast.

**Leo:** Yeah. Well, that's awesome.

**Steve:** So we have today an extra, triple-scoop, spiffy good podcast.

**Leo:** Another word from Dad, "spiffy."

**Steve:** Spiffy, yes.

**Leo:** I don't think anybody's said that in this century yet, so you're the first. That's good.

**Steve:** That and "tinkle." Those are...

**Leo:** Another one.

**Steve:** That's another one. So, not surprisingly, this podcast is titled "Exim Under Siege."

**Leo:** Uh-oh.

**Steve:** And I thought the word "siege" is the right one because remember from our discussion of, well, last week's podcast was titled "Update Exim Now!." And we both got a kick out of the fact that it takes a week of holding open a connection and trickling a byte every, what was it, four minutes because the connection's going to time out after five. But you have to just kind of keep it waiting for more. And so it takes a week. So we'll get there. That'll be the - we'll wrap the podcast up by talking about it.

And I had a hard time deciding what I wanted to talk about because there was so much that's going on. We have another Heartbleed-ish - or, well, no, actually it's not Heartbleed, sorry - a Rowhammer-ish problem with DRAM, which has come out from the same team again. And so that was competing with a number of other things that have happened. But I thought, okay, we really ought to follow up on last week's discussion because we now have a worm very aggressively propagating through Exim email servers. And as we would expect, its ultimate goal is to mine cryptocurrency because that's what things do now. It's all about the money. And if you're not going to be able to extort people and get them to pay you to decrypt their files, then let's just mine some coin.

So lots to talk about. We've got, as I mentioned, a new DRAM problem, RAMBleed, which is sort of a cross between Heartbleed and Rowhammer, RAMBleed. We've got a bad Linux server kernel-crashing flaw which, I mean, it's so new, the news just hit yesterday, that it's going to take a little bit of time, a few days, for the bad guys to arrange to craft some packets. This is another reason why raw sockets are a mixed blessing because that's what they'll use. And it's going to be able to bring any Linux server in the last 10 years that has not been updated in the last two days, well, it's going to give it a kernel panic and halt it. So we'll be talking about that next week. And that's why I thought, okay, let's talk about the Exim siege this week because we'll have a new disaster for next week.

So we've also got Patch Tuesday was last Tuesday, and some news there, including a Bluetooth surprise for a number of Windows 8 and 10 users. Another useless warning

about the BlueKeep vulnerability because, if anybody doesn't know by now, DHS chiming in, that's not going to help. Microsoft missed a 90-day Tavis Ormandy deadline - whoops! - by, well, we're not sure if it was one day or more, but it was...

**Leo:** You don't want to miss those. Oy.

**Steve:** They missed it, and there is now a new problem. Doesn't look that bad. Tavis considers it important, but I'm sure we'll see it fixed next month. We've got some good news in the GandCrab wrap-up. We have a mistake in entropy that Yubico found and is in the process of completely flushing out of the install base of FIPS-compatible Yubi devices. We've got, oh, a little bit of post-announce SQRL news, a favorite iOS security app that I've been wanting to talk about, and I was prompted to by a tweet that I saw. And then we're going to talk about what has happened since the Update Exim Now! podcast. It's been a fateful week. And of course I do have a fun Picture of the Week. So lots to talk about.

**Leo:** Not to add on, but Facebook announced that they're going to do a cryptocurrency.

**Steve:** What?

**Leo:** Which is, yeah, this is a big story because that's 2.5 billion potential users.

**Steve:** Wait, wha-wha-wha-wha - okay. Okay. Go ahead, sorry.

**Leo:** It's based on a Swiss foundation and consortium called the Libra Network. It will be the Libra coin. And I at some point would love for you to look at what they're doing to make it better than or different than bitcoin. I thought initially it would be pegged, but apparently it won't be pegged on any fiat currency. They have an interesting proof of stake blockchain that is new. And there are a number of people, including many in the crypto community, who say this could be a watershed for cryptocurrency. I know you have some interest in cryptocurrency.

**Steve:** Well, yeah. It's academic.

**Leo:** Yeah. Besides that erased hard drive.

**Steve:** Ooh.

**Leo:** Not to rub it in.

**Steve:** Yeah, thank you.

**Leo:** But it will be a stable cryptocurrency. That's their hope, anyway.

**Steve:** And so they would be a gateway for moving U.S. dollars into and out of this coin.

**Leo:** Not just U.S. dollars. They're going to use euros, yen, and pounds, as well as U.S. dollars. It'll be backed by actual assets in those four fiat currencies.

**Steve:** So there'll be an exchange for those.

**Leo:** Yeah. And of course they'll build it into all of their tools, so transactions through WhatsApp, Instagram, and Facebook will probably use - I'm calling them "Zuckbucks," but I think they want to call them "Libra coin."

**Steve:** That's hard to resist, though; you know?

**Leo:** It is, it is.

**Steve:** Once you hear it, you can't forget it.

**Leo:** Yeah. But, you know, you did this great, really in-depth piece some years ago on bitcoin and the...

**Steve:** We explained the blockchain, the whole concept.

**Leo:** Yeah, the mathematics of it. This is kind of an up-to-date, two-token system that I don't fully understand, but I'd love to get your thoughts on. So at some point...

**Steve:** So it exists independently, and there already are Libra coin...

**Leo:** No, no. Well, that's an interesting question. Facebook says not till 2020. But they're making this announcement now. "Libra: Stable cryptocurrency backed by a basket of financial assets which runs on its own blockchain." That's in their press release. So...

**Steve:** I'll know more about it next week.

**Leo:** Yeah, it's backed by a not-for-profit organization out of Switzerland called the Libra Association. I mean, it's fascinating. And of course they're trying to make it governmentally approved. But I can imagine a government might see this as an assault. Certainly the banks will. And they've got MasterCard, Visa, and PayPal as part of the consortium.

**Steve:** No kidding. So this is clearly the largest legitimate entity to ever back a cryptocurrency.

**Leo:** Yeah.

**Steve:** And of course what are the Winkle Twinkles, now in San Diego...

**Leo:** The Winklevi have all - they put their money into bitcoin.

**Steve:** Right, right.

**Leo:** But there's clearly been problems with bitcoin. And so these are of course to try to address those, particularly the volatility of bitcoin, which...

**Steve:** Well, it suffers from not having enough accelerator resistance in its fundamental algorithm, which allowed ASICs and, well, it just distorted the market. For example, Monero worked so well because it is hostile to acceleration. And that tends to just level the playing field more.

**Leo:** Yeah. And of course banking is nervous about it because it could reshape the payment industry. Governments might be nervous about it as the...

**Steve:** Well, it's uncontrollable.

**Leo:** Yeah, the undollarization of the world, introducing a new unit of account for global trade, a shift in monetary economics from public to private sectors. It'll be...

**Steve:** I'm surprised they're not calling it Facecoin. I would think they...

**Leo:** Yeah, I think Zuckbucks, honestly. But I think they're also at great pains not to imply that it's just Facebook behind this; right?

**Steve:** Right, right, right, right. Did you have a chance to see the movie that we talked about last week?

**Leo:** Not yet. I told Lisa, I said we have to see this.

**Steve:** Okay, yeah.

**Leo:** Before Steve spoils it.

**Steve:** I had a number of people, well, a lot of our listeners wrote...

**Leo:** Did they like it?

**Steve:** Loved it.

**Leo:** Oh, I can't wait.

**Steve:** In fact, some had already seen it, and they tweeted that the moment they heard me start to mention a new sci-fi movie on Netflix, they knew exactly what it was I was going to talk about.

**Leo:** What's the name again, so people can...

**Steve:** "I Am Mother," for those who...

**Leo:** "I Am Mother."

**Steve:** Our Picture of the Week, I snapped this off of the web page. It's something I failed to mention last week. I just thought our listeners, some of our listeners certainly would get a kick out of it. One of the early participants over in the web forum took it upon himself - oh, and it came up because of PopSockets. We were talking about those. And he had done a SQRL logo on a PopSocket. And I said, on a what socket?

**Leo:** I have it right here. Right?

**Steve:** Yeah. And anyway, so we also have T-shirts. Not we. I play no part in this.

**Leo:** Who's doing this?

**Steve:** It's someone who - I did work with him. I ordered a couple beta shirts to get the logo the right size and in the right position. But anyway, they are SQRL T-shirts.

**Leo:** Nice.

**Steve:** And they're very nice. I like a sheer T-shirt because I tend to run my engine a little hot. And so if you just go to Amazon and google - there's no links anywhere. I ought to catch up and do that. I've got a whole lot of catch-up to do.

**Leo:** Yeah. I went to your SQRL page. You need to, you know, get that merchandise going there, dude.

**Steve:** So if you just go to Amazon and google "SQRL T-shirt" you'll find them in whatever size, and there's a whole variety of colors, and anyway. So I just thought it would be fun to do that for the Picture of the Week this week, this very busy week.

**Leo:** Actually, you've got to be careful because you could get a T-shirt with a squirrel on it.

**Steve:** Yeah, not that one, no.

**Leo:** It's not the SQRL you're looking for.

**Steve:** Yes. Do look at the shirt before you click "buy now."

**Leo:** Yeah, SQRL, yeah. Here we go. Here we go. Secure Quick Reliable Login.

**Steve:** There we go, yup.

**Leo:** Nice. Very nice. All right.

**Steve:** And there's different styles. There's a hoodie.

**Leo:** Is anybody making money on this? Do you get a little...

**Steve:** Long sleeve. I think he gets like a small little "ving." Not much.

**Leo:** Because they're pretty affordable. That's close to the cost, yeah.

**Steve:** Yeah. Yeah, they're about what you'd expect to pay for a non-SQRL printed T-shirt. And the back is really nice. It's got a big SQRL logo.

**Leo:** Oh, look at that.

**Steve:** And mine says "Simply Secure." And we played around with different slogans.

**Leo:** I like this one. "Passwords? Where we're going we don't need passwords, Marty." I love it. I love it. That's fun. Okay.

**Steve:** Back to the SQRL. I mean...

**Leo:** Yes, back to the SQRL. I like that, too, yeah.

**Steve:** So, okay. Last week, as we know, was Patch Tuesday for this month. Microsoft, well, Microsoft's and Adobe's Patch Tuesday. Adobe had another critical vulnerability, once again fixed, in Flash Player, which Microsoft also promulgated for their own OS. Microsoft fixed 88 vulnerabilities. Once upon a time, Leo, we would have gone, "Oh, my god!" But now it's like, oh, okay. Round number, 88.

**Leo:** It's lucky, two eights. Lucky.

**Steve:** That's right. More than one quarter of those, 21 in total, were critical. Among the remaining 67 noncritical vulnerabilities fixed were the four deemed important which our friend - and I don't really mean that literally - SandboxEscaper found and, as we know, irresponsibly publicly released, as we've previously discussed in some detail. All four of those were various elevation of privilege hacks. And last we heard, she still had one more zero-day up her sleeve, which she was promising or threatening to disclose. So nothing since that disclosure from her.

Of the critical vulnerabilities, 21 in total, eight were in Microsoft's JavaScript Chakra scripting engine, five others in their Edge browser scripting engine that I guess is distinct from Chakra. Chakra is JavaScript, so I'm not sure, you know, in their enumeration they just said "scripting engine." But it was browser scripting engine. All 13 of those scripting problems, both the eight in Chakra and the five that weren't, were memory corruption. And again, those are critical. Microsoft considered those critical because, as we know, memory corruption is where remote code execution vulnerabilities are born. So they need to be taken seriously.

And speaking of remote code execution vulnerabilities, there were three of those fixed in Hyper-V, one in Microsoft's Speech API, another in ActiveX Data Objects, and also that critical Adobe Flash security update. And then among the remaining 67 important vulnerabilities was pretty much everybody was present and accounted for. As I said, the scripting engine, IE, Edge, the app platform and frameworks, Windows input and composition, media, shell, server, authentication, cryptography, datacenter networking, storage and file systems, SQL components, Microsoft's JET database engine, Windows virtualization, the kernel, and IIS.

I didn't mention Office. I guess there's nothing from Office. Maybe that was done separately, or maybe that's next week. Anyway. So another mega Patch Tuesday, which as I said we're now beginning to get kind of used to. And we'll be keeping an eye out for SandboxEscaper, if she has something a little more substantial to drop than that last one that was just, you know, not really very annoying.

But June also brought us - Leo, you're going to love this one - an update to Windows 10 Bluetooth stack. And after applying this month's latest security update, as Microsoft puts it, "...the affected platforms will experience the new behavior." Okay, now, Leo, you'll remember many, many moons ago when Intel published crude conceptual sample source code for what was new at the time Universal Plug and Play functionality. And though they never intended for this to happen, because it was just sample code, many router vendors copied and pasted that engineering source code sample right into their routers and compiled it. And the result was a surprisingly widespread vulnerability across router brands since everyone was using something, the same something, that was never intended to actually be put into production. And I remember we talked about it at the time. It was like, what? No. This was clearly marked "Sample, do not use." But, oh, look, it works. Ship it.

So, okay. Something similar has just happened. The Bluetooth Low Energy specification provides a sample long-term key, the LTK, which was provided in the specification, the BLE specification, only for illustration purposes, and of course was never intended to actually be used in practice. Somehow it found its way into Android's Bluetooth. And as a consequence, Android from 7.0 through 9 built in the long-term key, the same one, the one same long-term key from the Bluetooth Low Energy spec. And as a consequence, it's not secure. It's been identified as a security vulnerability, CVE-2019-2102.

Mitre.org says: "In the Bluetooth Low Energy (BLE) specification, there is a provided example Long Term Key (LTK). If a BLE device were to use this as a hardcoded LTK, it is theoretically possible for a proximate" - meaning someone close by - "a proximate attacker to remotely inject keystrokes on a paired Android host due to improperly used crypto. User interaction is not needed for exploitation." And then it says: "Product: Android. Versions: 7.0, Android 7.1.1, Android 7.1.2, Android 8.0, 8.1, and 9."

Microsoft published their announcement 4507623, saying: "Some Bluetooth devices may fail to pair or connect after applying June." And so they said: "You may experience issues pairing, connecting or using certain Bluetooth devices after installing security updates released June 11, 2019. These security updates address a security vulnerability by intentionally preventing connections from Windows to unsecure Bluetooth devices. Any device using well-known keys to encrypt connections may be affected, including certain security fobs." So in other words...

**Leo:** Oh, this is the Titan story come back.

**Steve:** Yes. Basically Microsoft blacklisted the key which was given as a sample in the spec, which...

**Leo:** As it should be blacklisted; right?

**Steve:** Of course, absolutely.

**Leo:** Yes.

**Steve:** In fact, I was thinking, you know, the industry should take this as a lesson. Anyone who's implementing the spec should preemptively blacklist any sample keys that the spec shows, just so that they never work. The problem is they have worked, and they've worked since Android 7, when they started being used. And if Windows had always blacklisted the key in the spec, if it had occurred to somebody, then they would have never been used because they would have never worked. But instead they have always worked until someone said, you know, this is probably not a good idea.

**Leo:** So I was wondering if this had to do with - remember Google had a problem with its Titan, Bluetooth LE Titan security key. I wonder if this is the same issue.

**Steve:** Oh, no. This is a different issue.

**Leo:** This is a different issue, okay.

**Steve:** Yeah, yeah, yeah. And so this does affect Android devices that will no longer connect to Windows. And frankly, they should really no longer connect to anything. Basically, you're using a known long-term key.

**Leo:** It's like using "password" for the password.

**Steve:** It's, yeah, it's like every web server in the world used the same certificate. It's like, uh, no. It's like, just back away from the terminal. So anyway, hopefully this mitigation will be made more pervasive, not just Windows 10. I mean, here's Microsoft that did the right thing, and people are complaining that they can no longer connect their Android device to Windows after applying the June update. It's like, yes, you should never have been able to connect your Android device to anything with this key from the specification. So anyway, they fixed it, and this will hopefully force some change through the Android ecosystem, slow as molasses as it is. Although, to Google's credit, we know they're working in the direction of speeding that up.

Okay. Now, this will be the topic of next week's podcast.

**Leo:** Oh, boy. There's so much. So much.

**Steve:** Yes. Hasn't happened yet because it just hit yesterday. We were recently talking, Leo, I think it was maybe week before last, about the seemingly nutty idea of China rolling their own Internet-connected desktop OS from scratch.

**Leo:** Right.

**Steve:** Which, you know, they should do. I mean, a homogeneous ecosystem is dangerous because a flaw found in one place affects everybody. You want a heterogeneous ecosystem. So it would be great if China had something different. But I just, you know, again, we were talking about how, well, the example I pulled out of why this is so difficult is how many flaws were found in the early TCP/IP protocol stacks. All kinds of little things you could do to lock up connections and cause problems and get up to all kinds of mischief. And so over decades we've, like, finally got a TCP/IP protocol stack that is solid. Except we don't. We just found a flaw in Linux's TCP/IP stack which dates back to 2.6.29, released 10 years ago.

**Leo:** Ooh.

**Steve:** Every Linux machine now accepting TCP connections on the Internet can be kernel panicked. It can be halted remotely.

**Leo:** Ooh.

**Steve:** Now, tell me that is not going to be next week's topic.

**Leo:** Yeah.

**Steve:** Okay. So there are three CVEs. It's a combination of something known as a SACK, and of course the Register just went wild with their...

**Leo:** Oh, I bet they did.

**Steve:** Having fun with, like, "giving it the sack" and all kinds of things. Anyway. Back in 2011 we recorded a series of three podcasts, carefully describing the low-level operation of TCP. Security Now! 317, which we recorded on September 8th, 2011, was titled "TCP Part 1." Episode 323 was recorded on October 19th. That was "TCP Part 2: Attacking TCP." And then two weeks later Episode 325, recorded November 2nd, 2011, was "TCP Part 3: Necessary Refinements." I mention that because I know that a lot of our listeners have been listening for a year or so, and those were really good. And, I mean, so if you don't feel like you understand what a SYN packet is, what an ACK packet is, if you go back and listen to those - and back then they were probably shorter. So we're not talking two-hour podcasts. They were, like, half an hour.

**Leo:** Well, I don't think they were that short. But okay.

**Steve:** Maybe, yeah. Anyway, you'll get a really good primer on TCP. Okay. So the original ACK packet, as I described it on those podcasts back in 2011, the idea was...

**Leo:** Episode 317.

**Steve:** Yes, Episode 317. The idea was that, when you initiate a TCP...

**Leo:** By the way, it's an hour and 30 minutes, so...

**Steve:** Oh, wow, we had already expanded.

**Leo:** Yeah, we'd already, yeah, expanded. That's a good word.

**Steve:** That's right. We were still doing alternating Q&A episodes. I can't imagine doing that now because there's just so much news every week. And that's why they're a little bit spaced apart. And then some other catastrophe happened between 317 and 323, so we had to cover that. But then we got back to wrapping up the series, or continuing. Anyway, the SYN packet - SYN stands for synchronize. That numbers the first byte that is going to be sent by each endpoint. And as they come in, the receiving side acknowledges the receipt of the last, the number, the sequentially numbered last byte received. So it sends back ACKs. That's the way TCP works.

In RFC 2018, which was dated October 1996, 23 years ago, so not exactly new, this concept of ACKing was improved. The problem was you might have many packets in flight, especially as speed improves on the Internet. And if one packet was dropped, but a bunch of succeeding packets made it, or the way the original spec was written, the

recipient could only acknowledge the latest consecutive sequential byte received. That is, even if they had received other later packets, they couldn't ACK those. And so that was recognized as a problem.

So SACK, standing for Selective Acknowledgement, expanded on the protocol to allow an acknowledgment packet containing a list that was able to enumerate those received and those not received so that, rather than the sending end having to send the packet not received and then resend subsequent packets that had already been received, which would obviously be redundant and would waste bandwidth, this allowed the recipient to specify which packets had not been received, separate from those that had. So that's Selective Acknowledgment, which we didn't talk about at the time because we were trying to keep things simple.

Well, it turns out that, if you combine a mistake in the code which has been in every Linux kernel for the last decade with another feature of TCP which is MSS, the Maximum Segment Size, the idea there is that the sender is able to specify when they're setting up their connection that there's something wrong at their end, they've got no RAM, so only send me no more than this many bytes at a time. These packets are known as "segments" in TCP parlance. And so the idea is you say, okay, I don't have much buffer space. You can't send me 64K blocks. You've got to make it a K or something. Well, it turns out that the smallest size is 48. When you subtract the overhead of 40 bytes for the packet, that leaves 8 bytes.

A security engineer at Netflix, of all places, found this problem and has reported it. If you combine in what is arguably a hack of a very small MSS, Maximum Segment Size, with screwing around with selective acknowledgment - and Red Hat has a beautiful, complete expos on this. I've got a link in the show notes. And here's the problem is that this is not going to be difficult to reproduce. This is not going to be difficult for the bad guys to do on any OS that allows raw sockets. Raw sockets allows you to build your own socket from scratch. It takes control away from the TCP/IP stack. And you can put anything you want to out on the wire.

I mean, and what happens is almost instantly the Linux kernel, which accepted a connection innocently, can be caused to panic and halt. The default behavior of Linux when you get a kernel panic is just it stops. It flashes the keyboard lights, and it assumes something really bad has happened, so it waits for someone to come along and help it. You can configure it to reboot itself, but that's not the default state. There are some workarounds where, if you have a Linux server that you don't want to take down at all, or that you're not ready to update, you are able to disable this SACK support. It's a sysctl command line. I've got it in the show notes also.

Ubuntu has a knowledge base article titled "SACK Panic." They wrote: "Jonathan Looney discovered several flaws in the way that the Linux kernel's TCP implementation processes Selective Acknowledgement options and handles low Maximum Segment Size (MSS) values. A remote attacker could use these issues to perform" - and technically this is called "denial of service" because, yes, if your kernel crashes and halts, that's a denial of service to everybody - "perform denial of service attacks on a server. CVE-2019-11477 is the highest severity issue because a remote attacker can leverage it to immediately crash a system due to an integer overflow when processing TCP SACKs. It affects all current Ubuntu releases." This is Ubuntu's knowledge base. And of course it's Red Hat. It's AWS. It's all Linuxes.

"You should update your kernel to the versions specified below in the Updates section and reboot. Alternatively, Canonical Livepatch updates will be available to mitigate these two issues without the need to reboot. If neither of those options are possible at this time, you can mitigate the issue by temporarily disabling TCP SACK support." And I've got the command line. But if you just google "sack," S-A-C-K, "panic," or sack, yeah,

probably "sack panic" or "sack TCP," already Google is returning lots of results. Oh, and that sysctl modification is not persistent across reboots.

So there is a quick fix. But as is always the case, there will be lots of servers that are not being well tended, that are in closets. I mean, and think about it. Not only mainstream servers, but Microlinux kernels in cameras and in security systems and in DVRs, I mean, anything that is Linux-based for the last 10 years. In its default configuration, which has this selective acknowledgments enabled because nobody had a reason not to, it was a benefit; right? It's a feature. It makes things more better. You send them a few specially crafted packets, and that thing is dead. It is over. So as I said, that will be - that cheery topic will be next week's podcast, I have a feeling.

**Leo:** No commitments because there's always something else.

**Steve:** Yes. Yes. I can't imagine anything worse, but you never know.

**Leo:** Well, a crash isn't the end of the world. I mean, of course, as we know, crashes are often precursors to hacks.

**Steve:** No, well, this cannot be leveraged in this case.

**Leo:** No. It can't be because it's frozen. It's too late.

**Steve:** It is the kernel recognizing, oh, my god, I don't know what to do. I'm halting.

**Leo:** Right, halt, yeah. [Crosstalk].

**Steve:** So it is a programmed fault.

**Leo:** Yeah.

**Steve:** But, again, having a server which is expected to stay up - oh, and remember that, if it does reboot, you just send the same little packets again, and it's down again.

**Leo:** Right. It's more of a mischief-making thing than anything else.

**Steve:** Yeah. But depending upon - but, you know, if it's in control of your power grid or your nuclear reactor...

**Leo:** Well, that would be bad, yes.

**Steve:** Yeah. So there are many places where Linux servers are assumed not to be vulnerable to immediately being crashed. I mean...

**Leo:** Now, if I'm behind a firewall and not accepting TCP packets, it's not a problem.

**Steve:** Right. The typical home user who is not serving anything, who has no open ports, they're fine.

**Leo:** Yeah.

**Steve:** But any, I mean, it doesn't even have to be a web server. It could be telnet.

**Leo:** Well, I have a Minecraft server that's accepting TCP connections. So it could be...

**Steve:** Yes.

**Leo:** That would be vulnerable.

**Steve:** It could be taken right off the 'Net. Yeah, yeah. So we'll see.

**Leo:** Yeah. Under siege, Steve. Under siege.

**Steve:** I happen to be a big fan of wasabi. I like to mix it with my soy until it kind of forms sort of a dark green mud.

**Leo:** How much wasabi? You use a tiny little ball of wasabi?

**Steve:** Oh, no, no, no, no.

**Leo:** Or do you put the whole clump in there?

**Steve:** I thicken my soy sauce with wasabi. I like to...

**Leo:** I thought you might. I like wasabi, too.

**Steve:** I like to perspire.

**Leo:** It's a great name, actually.

**Steve:** Great. So it was June of 2014, so five years ago, that we got the first whiff of a problem with DRAM memory not being as stable as we all hoped and assumed it was.

And, you know, there'd been the stories about spraying it with Freon, which was surprising to us at the time, and relocating it to a different machine. It was like, wait. You mean the bits hold onto their charge that long? It's like, uh-huh. Anyway, since then, five years ago, this subtle flaw has been developed into a growing family of attacks which we've gleefully covered over the years - Rowhammer, GLitch, RAMpage, Throwhammer, Nethammer, and more recently DRAMmer.

**Leo:** Oh, yeah.

**Steve:** Now to those we add RAMBleed, which is different. And we should probably call that prolific team at the University of Michigan and the Graz University of Technology the "Ram Busters." We have a website, RAMBleed.com, and a logo because, you know, a good exploit needs a good logo. So we got that. The official title is "RAMBleed - Reading Bits in Memory Without Accessing Them." And I'll just read their little description. They said: "RAMBleed is a side-channel attack that enables an attacker to read out physical memory belonging to other processes." Oops. That's not good.

"The implications of violating arbitrary privilege boundaries are numerous [uh-huh] and vary in severity based on the other software running on the target machine. As an example, in our paper" - get this - "we demonstrate an attack against OpenSSH in which we use RAMBleed to leak a 2048-bit RSA key. However, RAMBleed can be used for reading other data, as well.

"RAMBleed is based on a previous side channel called Rowhammer" - which of course we've talked about extensively - "which enables an attacker to flip bits in the memory space of other processes. We show in our paper that an attacker, by observing Rowhammer-induced bit flips in their own memory, can deduce the values in nearby DRAM rows. Thus, RAMBleed shifts Rowhammer from being a threat, not only to integrity, but to confidentiality, as well. Furthermore, unlike Rowhammer, RAMBleed does not require persistent bit flips, and it is thus effective against ECC memory commonly used by server computers." And of course it was ECC that was stated as the cure for the Rowhammer problem because, if you flipped a bit, the ECC would flip it back.

Okay. So what did all that mean? As we've talked about with Rowhammer, remember that the idea was, if you pounded on DRAM, reading DRAM is a destructive process. Remember that so DRAM itself is a row of little itty-bitty capacitors which store electrostatic charge. And if they're charged up, they contain a one. If they're discharged, they contain a zero. So the act of reading a row of memory - oh, and I should also mention that a DRAM is a big grid, a big rectangular grid of rows and columns of these little cells. The act of reading a row transfers the charge out of the row into sense amplifiers on the edge, and thus the row needs to be rewritten. So reading forces a write.

And what was discovered is that, unfortunately, in the push to increase density, the capacitors have been made ever smaller. And the engineers have said, okay, there's enough margin, error margin, tolerance, noise margin, that this is reliable enough. Now, if you want more reliability, because you can get an occasional misread, then you add ECC memory. The idea is there's an extra bunch of bits tacked onto the end of each row to detect and correct a bit if it's read back incorrectly. The point is that DRAM is not perfect. And again, unfortunately, they keep making the storage cells tinier in order to cram more of them onto a smaller chip, in order to keep the cost down and to raise the density. And we end up with a situation where it turns out that, if you make a big ruckus in the neighborhood, then you can cause a misreading, a deliberate misreading in bits nearby, that is, the adjacent row bits.

And then what was discovered, you could even get more bits to flip, or bits to flip more reliably, with what was known as Double Rowhammer, or DRAMmer, where you would be pounding on both rows on either side of your target row, so to kind of get it from both sides and even induce more bit flips. And, for example, we talked about one instance where - this was so clever - where in a VM environment, the private key which was in use by a server - remember that a private key is deliberately the product of two primes, the point being you can't factor it in order to break it apart again.

Well, these guys flipped a bit in the private key on an adjacent virtual machine that they were sharing, which meant that its private key was no longer the product of two primes. It was the product of other things that was easy to factor. And so they factored it and then they were able to spoof their connection, essentially. So it's just super clever.

Okay. What these guys have done - okay. So that was Rowhammer, pounding on adjacent rows on either side of a target to make a bit flip. Well, the observation had been made then, but it wasn't until now that it was weaponized, that the probability of being able to get a bit flip depended weakly but significantly upon the bit states of adjacent rows. In other words, if you could arrange to cause a secret to occupy the physical row above and the physical row below your own memory, then you could use Rowhammer on your own memory and, based on the probability, that is, the success rate of flipping bits in your own memory, you could infer the invisible contents of the rows above and below. Which is just amazing.

They said: "Specifically, '1' bits tend to flip from 1 to 0 when the bits above and below them are 0, but not when the bits above and below them are 1. Similarly, '0' bits tend to flip from 0 to 1 when the bits above and below them are 1, but not when the bits above and below them are 0." In other words, there's a tendency for the adjacent bits to pull the bit in the middle to their same state if you give them the chance. And these guys weaponized this. They nailed it to the ground. And they are able to extract a 2048-bit secret key from an adjacent process that they have no read access to. They can exfiltrate that key over time.

And it turns out that, if there's ECC memory, that is, if they're trying to do this in the presence of ECC memory, ECC, when it detects that a bit flip has occurred, introduces such a dramatic slowdown in performance that that can be detected. So they're able to infer that a bit flip occurred, even if they can't see it, because when they attempt to read it to detect a bit flip, ECC will get there first. It'll see, whoops, we had a problem here. And it'll fix it. But that delays the result of the read so much that even though the data comes back apparently not having been flipped, they go, aha, that took too long. ECC corrected the bit flip that we induced. So even ECC won't fix this problem. So there really is no solution to this.

They wrote: "Users can mitigate their risk by upgrading their memory to DDR4" - because we know that four is better than, you know, more noise-immune than three - with also the feature known as targeted row refresh. The idea is that, as we know, DRAM, because these are little capacitors storing our data in the charge of a capacitor, it tends to bleed over time. Again, they've made them so small that they're, like, counting electrons now. So it's necessary to periodically read through all of DRAM to read every row out before the "1" charges have a chance to get so low that you really can't distinguish them from "0."

So the idea is that the rate at which you refresh affects the integrity of your DRAM. One of the early mitigations for Rowhammer was increased refresh rate. That would improve the noise margins of DRAM in order to minimize the chance of Rowhammer attacks. The alternative is this TRR, which is a new feature, Targeted Row Refresh, the idea being that the memory system looks at the memory access pattern and will over-refresh the areas

that might be subject to either inadvertent or deliberate bit flips by selectively, preferentially, refreshing those rows more than others.

So anyway, so fundamentally there is no good solution here. Just as we have learned that our high-performance CPU architectures are flawed at the fundamental level by being altered by their own execution history, thus Spectre and Meltdown all of last year, we have seen the DRAM is similarly flawed at a fundamental level. That is, I mean, these are not fixable problems. These are fundamental flaws in the systems that we've been using. And we're now - what we've seen for the last, well, really for the last five years, if you consider Rowhammer, but also certainly all of the last year with all of the microarchitectural flaws, is that academic researchers are really looking at things that we've taken for granted closely and successfully poking big holes in what we thought we were able to trust.

So I don't know. In their Q&A that they have, they've mentioned that they are highly suspicious or skeptical that anyone would have already succeeded in weaponizing this and using this. But here is a problem that ups the ante. I mean, it would take - it took a lot of massaging of memory management in order to cause the physical alignment of the secrets they want to exfiltrate with the memory that they have control over. But this is arguably a more powerful attack, that is, this RAMBleed, than all of the previous Rowhammeresque attacks because, given that you have time and the ability to massage memory allocation, which is what's necessary in order to create physical RAM proximity, then you are able - they said three to four bits per second is their exfiltration bit rate once they have everything set up.

So it really does mean that servers need to be looking seriously at using DDR4 DRAM with targeted row refresh. Maybe sacrifice some performance by increasing refresh rate, if possible. And really, stepping back, this doesn't affect typical end users, again, in the same way that Spectre and Meltdown don't, because the real problem is only when you've got reason to believe that something malicious is sharing your hardware with you. As we've often said, if you've got something malicious on your desktop, sharing your desktop hardware, well, you're already in much bigger trouble.

The real problem is in the cloud. And the good news, I guess, is that's where they're able to spend some money in order to implement stronger hardware mitigations against this. But even though this is not good news, it's better to know that we have this problem, I mean, that's why targeted row refresh exists now in DDR4 is thanks to the early work five years ago that DRAM can be exploited successfully with Rowhammer-style attacks. So we got improved DRAM. Similarly, we're getting improved microcode now and improved microarchitectures in the future in order to further mitigate microarchitectural information leakage. So onward.

I just sort of shook my head when I saw another warning about BlueKeep vulnerability. We're now, what, six weeks from the announcement, which would have been the May Patch Tuesday. Still no worm. And as I have said, I doubt that we're really going to see one. When we talk about the Exim worm here at the end of the podcast, I'll reiterate why Exim made sense to be wormed, and BlueKeep just doesn't. So as we know, we've had two warnings from Microsoft. Warnings from lots of other people, too, but two official warnings from Microsoft. Last week we talked about the warning from the NSA.

And now we have the U.S. Department of Homeland Security, the CISA, the Cybersecurity and Infrastructure Security Agency, which yesterday, Monday, published an alert for Windows users to patch the critical severity remote desktop services RCE, the Remote Code Execution security flaw known as BlueKeep. Thank you, DHS. Always keeping an eye on our back. Again, it's not been patched yet. I'm skeptical that anybody getting this Department of Homeland Security advisory is going to be patching it that hasn't already. They did, however, confirm that they had successfully executed a remote

code execution on Windows 2000, which was previously - it was speculated to be vulnerable, but it hadn't been proven to be vulnerable. So they added a little bit of information to the growing pile of information on BlueKeep.

Anyway, as I said, I'm unconvinced that we're going to see a worm, although the world is surely asking for one. That's what we keep hearing is oh, it's wormable, it's wormable, it's wormable. It's like, yeah. Except that it's not difficult to find them. Anyone can find where all the servers are. It takes zero time to exploit because basically you just ignore logon credentials, and you can immediately establish a remote desktop session and set up your cryptocurrency miner and then close the door behind you. So it just seems like as soon as somebody does that, if they haven't already been doing it, they're going to do that. Doesn't make sense to have a worm. But as we mentioned, there is one for Exim that we'll be talking about in a minute.

As I mentioned at the top of the show, when Tavis Ormandy tells you that you have 90 days to fix something, get on it. Of course, we're speaking of Google's illustrious bug finder. Issue 1804 - I have a link to the bug report, Project Zero, Google's Project Zero. Issue 1804 titled "cryptoapi: SymCrypt modular inverse algorithm," reported by taviso@google.com on Tuesday, March 12, 2019. He wrote, and this was not public, so he put this under lock privately on March 12.

"There's a bug in the SymCrypt" - and that's a component of Windows - "multi-precision arithmetic routines that can cause an infinite loop when calculating the modular inverse on specific bit patterns in" - and then it's a particular function call: bcryptprimitives is the module; SymCryptFdefModInvGeneric is the function. So, he said, "I've been able to construct" - and here's the key. "I've been able to construct an X.509 certificate" - that's a standard identification certificate - "that triggers the bug. I've found that embedding the certificate in an S/MIME message, Authenticode signature, secure channel connection" - i.e., TLS, HTTPS - "and so on will effectively DoS any Windows server," and then he says, "e.g. IPSec, IIS" - which of course is their web server - "Exchange, et cetera; and, depending on the context, may require the machine to be rebooted. Obviously, lots of software that processes untrusted content, like AV, call these routines on untrusted data, and this will cause them to deadlock."

He says: "You can verify it like so, and notice the command that never completes." And he gives an example. So he shows C:\, so he's logged into the root directory on a Windows machine. And the command is "certutil.exe," and he gives it "testcase.crt," which is the certificate that he manually constructed to invoke this flaw, this vulnerability that exists in the Windows symmetric - actually it's in the SymCrypt library, which was previously and traditionally the symmetric cryptography, but naturally a certificate is asymmetric crypto. So he says: "I'm filing this as low severity, although you can take down a Windows fleet pretty quickly with it. This bug is subject to a 90-day disclosure deadline. After 90 days elapse or a patch has been made broadly available, whichever is earlier, the bug report will become visible to the public."

Okay. That was on March 12th. Six days later, March 18, Tavis added an update to that with an MSRC, Microsoft Security Center, Label 50858. So it had been acknowledged and given a number. Eight days from then, on Tuesday, March 26th, he added, "Microsoft replied that they would like to issue a bulletin for this issue but need until June 11th." Okay. That was last Tuesday; right? Patch Tuesday. He says: "I count that as 91 days, but within the extension period, so it's acceptable."

Then last week, last Tuesday, June 11th, at 8:16 a.m., six days ago, the label of his update submission was Restrict-View-Commit Deadline Exceeded. "MSRC reached out," he writes, "and noted that the patch won't ship today, and wouldn't be ready until the July release due to issues found in testing." He says: "As today is 91 days, derestricting

the issue." And in there, in this disclosure that I have the link to in the show notes and, Leo, you have on the screen, are two files that implement this vulnerability. Now...

**Leo:** It seems a little dickish to me, but okay.

**Steve:** Well, but 90 days.

**Leo:** Yeah, but that's very anal, though. They said, well, we thought we were going to have it out in time; but we found a problem, so we're going to put it to the next release. Come on.

**Steve:** Ninety days, Leo.

**Leo:** I would say, okay, I'll give you to the next release.

**Steve:** They could have fixed it.

**Leo:** Well, okay.

**Steve:** Yeah, I mean, again...

**Leo:** What's the point of a 90 day? Give me the rationale for the 90 days.

**Steve:** So the rationale is that these sort of things could be found by other people. We don't know they haven't been. And we don't know that they're not being exploited.

**Leo:** And now we know they have.

**Steve:** Yes, now we know they have.

**Leo:** So what is the point of this? To punish them?

**Steve:** To motivate companies to take these things seriously.

**Leo:** Well, yeah. But I think they did, and they had a fix, and then it turned out there were unforeseen issues. Wouldn't you want to give them...

**Steve:** We don't know, we don't really know. I mean, 90 days is a long time to fix what is probably something simple in the asymmetric crypto library. I mean, as far as we know, they dicked around, to coin a term, for 60 days and then thought, oops, whoops, maybe we need to do this. And so they said, well...

**Leo:** Well, what if they didn't? What if they worked really hard, they got a fix, they didn't - and these things happen; right? Sometimes the fix causes another bug. In their testing, right before they released it, they found a bug. They said, well, okay, we're going to have to wait till next Patch Tuesday because you don't want us to release this with another bug. They're working hard on it. What if that's the case? What if they worked every day between the discovery of this and the release of it, and they just want one more, give us one more Patch Tuesday? I don't know. It feels a little dickish to me. It's not the first time TO has been a little prickly.

**Steve:** You know, we have seen instances where companies have gone six months blowing off reports.

**Leo:** And that I understand, absolutely.

**Steve:** And not bothering.

**Leo:** But I also don't think it's appropriate to be 100% anal about 90 days. Come on. That's a little anal. What you're looking for is the intent.

**Steve:** He was going to give them an extra day, Leo.

**Leo:** A whole 91.

**Steve:** He was going to give them 91. But I certainly take your point. The good news is, as we were discussing, GandCrab, which was this weird Ransomware as a Service - remember I had fun reading the, well, I didn't read it, I read my own - I wrote a pitch for people who are unable to create crypto malware of their own to use GandCrab's in return for giving the GandCrab folks a relatively small, apparently, piece of the action. They announced that they had "earned" all the money - "extorted" is more appropriate - all the money that they needed to. They had laundered it by investing in legitimate Internet and other businesses, and so they were shutting down. They urged everyone, all of their subcontractors, to wrap things up, that they were no longer going to be supplying keys.

Well, it turns out that Bitdefender managed to hack their servers and obtain the entire database of keys. So if anybody was ever infected by version 1, version 4, version 5.0 to 5.2, those keys are available. Bitdefender has a link to the GandCrab removal tool for versions 1, 4, and 5 of GandCrab. BleepingComputer covered this news in their coverage and explained that, after being installed, this GandCrab removal tool, it will need to connect to the Internet in order to check in with Bitdefender's servers and obtain keys for that specific machine.

BleepingComputer suggested that you test decrypt sort of a subdirectory that's not crucial to make sure that it's working. If it proves itself out, then go ahead, turn it loose on your entire machine; and, had you been encrypted, you can get your data back. So this wasn't quite as cool as GandCrab deliberately turning the keys over. But the result is the same, and that is that anybody who had been encrypted and who had not yet managed to decrypt themselves, thanks to Bitdefender is able to do so.

Yubico hit an interesting bump and is replacing all of its FIPS-certified keys, that is, versions 4.4.2 or 4.4.4. There was no 4.4.3. So anybody listening to this who hasn't already received notification, who has 4.4.2 or 4.4.4, should receive and can receive at no charge a replacement to 4.4.5. In the middle of March, so just about exactly three months ago, Yubico internally discovered that some of the data left behind by the FIPS firmware power-up self-test was lingering in the device's random bits buffer. And for the first cryptographic operations which were made after power-up, those bits which were not high entropy were mistakenly being delivered as if they were high-quality entropy. Once all of those bits were consumed, then the random bits buffer would be replenished with the intended high-quality entropy.

But this meant that the initial post-power-up state of the device was not generating the intended entropy, which they have fixed in 4.4.5. In their advisory they said: "An issue exists in YubiKey FIPS Series devices, versions 4.4.2" - actually, I think I've already pretty much covered everything they said. They found the issue mid-March. They did a full investigation.

"To safeguard the security of our customers, Yubico has been conducting an active key replacement program for affected FIPS devices since the issue was discovered and recertification was achieved." They had to get FIPS-recertified for 4.4.5. "At the time of this advisory, we estimate that the majority of affected YubiKey FIPS Series devices have been replaced, or are in process of replacement with updated, fixed versions of the devices. However, if you have purchased a YubiKey FIPS Series device or received one from another entity and have not been contacted by a Yubico representative, we ask that you review this advisory to determine if you may be affected and use the replacement portal to receive updated keys."

So, you know, they immediately fixed the problem and have proactively replaced the hardware that was found to have defective firmware. So props to Yubico. And if anybody has a 4.4.2 or 4.4.4 FIPS device, it's either that or the C versions. Anyway, you probably know [crosstalk].

**Leo:** The standard YubiKeys are not FIPS devices.

**Steve:** Are not FIPS, yes.

**Leo:** In fact, if you have a FIPS key, it should say on the key, if you look at the little dongle that sticks out from it where the QR code is, it'll say "FIPS" on it.

**Steve:** Exactly.

**Leo:** When this came out I was all worried, but I realized I just have the regular FIDO U2F key.

**Steve:** Yeah. And they did note - their explanation goes on at some length. In trying to assess the severity, they noted that the biggest problem was with ECDSA, Elliptic Curve Signing, because the elliptic curve keys are much shorter, that is, typically 256 bits. So the percentage of bad entropy, about 80 bits of bad entropy, forms a much larger percentage of the shorter key. But, for example, a 2048-bit RSA key, because it's so much longer, the percentage of bad entropy 80 bits is a much smaller fraction. And you could, you know, you would like it to be 100% entropy. But the actual threat is much less

in terms of practical use if you're using a 2048-bit key. So anyway, they've got it fixed, and they've pulled them all out of the field, and they are replacing them. So that's all you can ask.

I mentioned last week Sysmon, which was being updated by our friend Mark Russinovich at the Sysinternals branch of Microsoft, or subsidiary, or whatever it is now. Microsoft bought Sysinternals and then kept it alive. We were all worried when it happened. Everybody downloaded all the Sysinternal tools that day because we weren't sure if they were being purchased to destroy them or what was going on. But no.

We are now - I mentioned it last week because Sysmon was announced to be forthcoming with an update that would allow it to also log DNS queries made by systems to the event log. And I directed our listeners, typically end users, many of us, to the NirSoft's DNS Query Sniffer. And I got a lot of positive feedback via Twitter about people thinking it was really cool to be able to watch their machine reach out and look up DNS addresses. Also, but we have a lot of enterprise listeners, too, who said, "Hey, Steve, just so you know, Sysmon rocks. Our enterprise uses it like crazy. And we love the fact that it will be adding DNS logging because this allows us then to pull all of the logs from all of the endpoints in our enterprise and see if anything is doing anything that seems suspicious."

So definitely a useful feature. And I didn't mean to say that it wasn't, just that for an end user, digging down into the event log, I mean, I have to do it for my own development purposes from time to time, and it's not right there in your face, and not real-time scrolling and all the other good stuff that you'd like to have. So both purposes have a point.

I have updated the SQRL Explainer. I forgot to mention in the first, what was it, 17 pages. It's now at 21 pages because by popular demand I added a three-page glossary at the front of all of the various terms. One page of standard crypto terminology, and I think about a page and a half of SQRL-specific terminology, terms that had to be invented in order to discuss new features that SQRL brings to the world. But then I also - someone else mentioned that I had forgotten to mention the "Ask" feature in the first release.

During this protracted five and a half years, almost six, of development of SQRL, we kind of wandered off course a few times. There were, like, there was discussion. It's like, well, shouldn't it also fill in forms? Why not? While we're doing things, we can do anything we want to. Form-filling is annoying. Let's put that in. And it was like, oh, I mean, and that's maybe an egregious example of a suggestion that's, like, way off of authentication.

But there was some initial design where it was more involved in the sign-in process and in the account management than strictly authentication. And so a couple times I had to slap myself and say, "Bad Steve. Let's stick to authentication. That's what this is. Let's not make it overly complicated. Keep it simple." And so forth. So it is maybe significant that one non-authentication feature survived. And it did because it was just too unique. There wasn't any way to replace it. There wasn't, I mean, unlike form filling and things, and that problem has been solved already. And this is the - we call it "Ask" because it allows a limited out-of-band, meaning out-of-browser communication between the web server and the SQRL user. To allow, for example, a website to confirm something that it really, really, really wants to make sure the user intends to do, like are you sure you want me to transfer $100,000 to this numbered account in the Cayman Islands?

Now, the problem is, if that's in your browser, there's just so many ways that can go wrong. I mean, so many - our browser has unfortunately become a battleground. But with this "Ask" feature, the server returns a question and is able to optionally provide the label of neither or up to one or two buttons. And when the SQRL client sees it, it presents

a special dialog to the user with the question as the contents of the dialogue, and prompts them for a response.

So anyway, that's now documented in the Explainer. And it's the one thing beyond authentication that survived all attempts to restrict this to strict authentication. We succeeded except here. And here I think, I mean, who knows? It may never get used. But I can see, if this were part - it's the kind of thing where, if it weren't in the spec from the beginning, then it would be difficult for anyone to ever rely on it. But it's in there. It's a simple addition. It was easy to add. And it potentially could be a real benefit because, again, it allows a web server that knows it's talking to a SQRL user because that's the way they authenticated, to send them a question completely separate from the browser channel that - oh, and I don't know if I - I guess I didn't really explain it, is that there is no potential, there's no possibility of a man-in-the-middle intercept of this, due to the way the SQRL protocol works. It cannot be intercepted. So anyway, it made it.

We already talked about T-shirts on Amazon. We're up now north of 2,000 members in the SQRL web forums as a consequence of big - we got a big bump, not surprisingly, and lots more new SQRL users. So thank you to all of our listeners who are playing with it and learning it. A lot of interest has been generated. People have been having trouble with SQRL on Mac and Linux under Wine. And so I just wanted to say that I went to lengths to make sure that GRC's Windows client would work under Wine. But it's not my intention that it really be - that it receive heavy use.

What we need is a native Mac implementation of a SQRL client. There is some work underway on a native client under Linux. Far as I know, no one's working on a macOS version. But we also have browser extensions for Firefox and Chrome and Edge. And so those work beautifully on a platform independent fashion. So the problem's been solved. There were a lot of problems with getting Wine to work, just because it's amazing to me that a Windows app even tries to run on a non-Windows environment, and I'm surprised that it works as well as it does. But it does.

And also, Leo, you and I talked about what we will do, which will be sort of the official SQRL Explainer professional video production that I'm very excited about doing. I want to solicit, explicitly solicit, user videos for how to use SQRL. I think users are better placed, and it makes more sense for end users who are interested in creating videos for how to do things, the different parts of using SQRL, to create videos. And if they do them, I will be happy to host the best of those on the GRC SQRL web forums and to host the bandwidth. I'm annoyed by commercials in YouTube, and so I will host the bandwidth.

But if people who listen to the podcast are camera equipped and like the idea of producing some how to use SQRL videos, all different aspects of the experience, I'd love to have them and love to host them. So I just wanted to make that explicit. I'm not going to do that. I will do the technical how it works presentation. But be really fun to have user-sourced videos.

And just a quick note about SpinRite. I realized through some other work that I'd been doing that I was underestimating the performance that I think we're going to get under 6.1. I'd long been talking about half a terabyte per hour. And it looks like SpinRite will be able to do, on a 7,500 rpm 2TB drive, a Level 2 data recovery scan in closer to three hours, just a few minutes more than three hours. So that's not bad for 2TB. And of course what I had failed to take into account was that the data rate on these drives increases as their aerial bit density goes up. And their aerial bit density has gone nuts in the last five years. And as a consequence, their raw maximum data rate has been increasing. And the point is SpinRite will be able to run at whatever the maximum throughput of the drive is. It will not miss a revolution as it's operating. So it's going to be screaming. And as soon as I get the rest of SQRL put to bed, as we know, I will be back to it.

I mentioned an iOS security app. Andy Pastuszak tweeted about TOTP. He said: "I went and enabled TOTP." That's of course Time-based One-Time Passwords, the increasingly ubiquitous authenticators. He said: "I went and enabled TOTP on every site I have an account on that supports it. And now my Google Authenticator is a complete mess. Having one long list of TOTP codes can be quite problematic, especially since I use a Firefox extension that wipes all my cookies when I close the browser," meaning that he has to constantly re-log on as his sessions are not sticky across browser sessions.

He says: "I'd love it if there was a TOTP app that does not sync and allows me to organize my folders, my codes by folder or even tags, or even just had a search feature." God, I don't know how many he has, but congratulations, Andy. He says: "Until SQRL gets here, we're all stuck with two-factor authentication. Could you maybe recommend the two-factor authentication app that has some organization behind it and has the Steve Gibson seal of approval? I can't be the only listener with this problem."

Well, so it just so happens, and he didn't mention whether he was an iOS users, turns out he was, and he loved my suggestion. I heard back from him after I gave him my suggestion. I have now been for quite a while using an iOS, and it's also available for macOS, app for two-factor authentication which I love. It's called OTP Auth. It's got a super high rating, 4.8 on the iTunes store. And if you look at the little graph of ratings, it's just like the five stars is almost all the way to the end. I don't know if - because I own it, iTunes won't show me what the price is, so I don't know if I bought it or if it was a free download. But it is ad free.

**Leo:** It's free with in-app purchases. But I don't see what the in-apps are.

**Steve:** Okay. So maybe I bought it after I used it or something. I've never seen an ad. And the author says it's ad...

**Leo:** Well, it's ad-free, yeah.

**Steve:** Yeah. He says it...

**Leo:** It's 3.99 if you want to maybe just pay, like give him some money. Maybe that's it.

**Steve:** Ah, good. Then, okay. So encrypted iCloud sync. Siri support. Apple Watch support. It'll support a notification center widget, and I use one. So you can very quickly swipe left from your root page, and it'll come up, and you can choose. It is folder enhanced, and there is a notification folder so you can choose which of your sites you want to show in the notification center. He says secure application using Face ID, Touch ID, or password. Create encrypted backups of all accounts. Import/export encrypted accounts using AirDrop, iCloud, Dropbox, Mail, and so forth. Works offline.

Anyway, it's done right. And so I do recommend - oh, he has a Twitter handle, @otpauth, O-T-P-A-U-T-H. So that's, you know, I recommended it to Andy. I just wanted to commend it to all of our listeners because I think it was really well done. And really quickly, My Krol, M-Y space K-R-O-L, tweeting from that, said: "Hi, Steve. A lot of podcast episodes I hear you talking about the special tabs in Firefox. Can you explain me what you are talking about?" And so very shortly, or very quickly, Tree Style Tab. It's an add-on for Firefox. Just search for Tree Style Tab, three words. It went through a facelift

a while ago. It is now skinnable using CSS. So you can make a series of customizations to it to make the tabs just the way you want them. And I love it. So Tree Style Tab.

**Leo:** Okay, Steve. On we go with the show.

**Steve:** So the question is, can you hear me?

**Leo:** Yes.

**Steve:** Okay. Your audio is doing some bizarre roboto thing.

**Leo:** Oh. All right.

**Steve:** But that's...

**Leo:** I'll just shut up.

**Steve:** As long as it's coming in your direction, that's the only thing that matters.

**Leo:** You sound great.

**Steve:** Okay, great. So, okay. So not a surprise to anybody who's been listening to this podcast so far. The Exim mail server, which we urged everyone to update, like immediately last week, has in fact come under siege. The first picture here in the show notes is a little difficult to parse unless you stare at it for a while. What it shows is, from the announcement day of the vulnerability, the industry's response in the first six days. And I have to say it is very impressive. The big green area is non-vulnerable Exim servers. And so if you just look at the very far left edge, for those of our listeners who are audio only, almost all of the servers on the 'Net, virtually all of them were vulnerable.

One day later - again, this is just very impressive. In one day the percentage of vulnerable servers drops from essentially 100% down to about 25 percent. So far fewer vulnerable servers in one day. The problem is, that was a good day. Then it sort of wanders, and we're still left with an Internet full of vulnerable servers. In fact, the second picture is one of our famous Shodan outputs. And it turns out - remember I asked last week, sort of hypothetically I wondered whether it was possible for us to - whether servers would announce their versions. Turns out they apparently do. Or at least it's possible to query them. I haven't looked at it any further because this is a Shodan search.

The title is "Search for Product Exim Minus 4.92." Anyway, the idea is it's a search term for versions lower than the one 4.92, which is the lowest one which is vulnerable. So less than that. And so we have, a week later, just shy of two million vulnerable Exim servers in the United States - 1,996,569. In Russia, 192,737. Canada, almost 143,000. The Netherlands, 137,000. Germany, nearly 130,000. The U.K. 123. Anyway, you get the idea. More than two million looks like maybe - in fact there is a number somewhere. I think it was 3.5 million was the total that I saw.

So CyberReason, an Amit Serper at CyberReason, two days after last Tuesday's podcast, so last Tuesday, June 11th, our podcast was titled "Update Exim Now!" Two days later, Thursday, June 13th, Amit Serper of CyberReason, his posting was "New pervasive worm exploiting Linux Exim server vulnerability." For the summary he said: "There's an active, ongoing campaign exploiting a widespread vulnerability in Linux email servers. This attack leverages a week-old vulnerability to gain remote command execution on the target machine, search the Internet for other machines to infect, and initiates a cryptominer."

For bullet points he said: "Currently, more than 3.5 million servers are at risk worldwide. The attack scours the Internet for a vulnerability discovered last week, CVE-2019-10149, using already infected servers to spread to as many as possible, i.e., worm. The target of this attack, Exim servers, run almost 57% of the Internet's email servers. The attack culminates in the downloading of a coin miner payload, which as we have seen previously with WannaMine can have a negative impact on any organization." Finally: "These kinds of attacks have big implications for organizations. The recovery process from this type of attack is costly and time consuming."

So what do we know? From his posting, he says: "CVE-2019-10149, which was first discovered on June 5th" - so let's see. June 5th, and this is on the 13th, so eight days it took - "is now being used as" - well, and also remember that the vulnerability takes a week to do, which says immediately, within a day or two, after the announcement happened, this thing was weaponized and turned loose so that other Exim servers could be commandeered, and this thing could propagate. He says - so first discovered on June 5th. Eight days later is now being used "as the vulnerability for a widespread campaign to attack Exim servers and propagate across the Internet."

He says: "We are aware of an initial wave of attacks as described by Freddie Leeman on June 9th. The first hacker group began pushing exploits from a C2 server located on the clear web. A second round of attacks by a different attacker are being analyzed by the Nocturnus team." That's these guys. "The campaign uses a private authentication key that is installed on the target machine for root authentication." That is an SSH key. I'll get there in a second. "Once remote command execution is established, it deploys a port scanner to search for additional vulnerable servers to infect." So classic worm behavior.

"It subsequently removes any existing coin miners on the target, along with any defenses against coin miners, before installing its own." And he writes: "Note: This is a very long script that downloads additional scripts and changes or adds many configurations on Linux servers." That's where he meant that remediation was tough because it's like really hosed your server. "This blog has the highlights of what the script is doing to provide a fast reference guide to the attack. Some of the things that the script is doing are not documented in this blog post. The hash of the script is available at the end of this article. It has also been uploaded to VirusTotal."

So the highlights he mentions - I wanted to close a window. So for highlights he mentions: "This is a highly pervasive attack that installs cron jobs for persistence and downloads several payloads for different stages of the attack. In one of those stages, one of the payloads is a port scanner written in Python. It looks for additional vulnerable servers on the Internet, connects to them, and infects them with the initial script. In the attack, the attackers add an RSA authentication key for the SSH server, which allows them to connect to the server as root and own it completely."

Okay. So again, remember that what this thing does is this is not remote code execution. This is remote command execution. So what they did was they came up with a script of commands that would cause the server to download an SSH key which would install as an authenticated SSH-RSA key that would then allow somebody to SSH into the server to do whatever else they wanted to. So he writes: "If you are running an updated version of

the Exim server, or you think that your server is compromised, please look for the following entry in your SSH configurations." And of course if you're a Linux server user, you know to look in /root/.ssh and in other users' .ssh directories. And then he provides a textual version of the SSH-RSA key text, which anyone could compare against. If you find that, it means your server has been taken.

He says: "In the final stage of the infection, the script downloads what appears to be a Windows icon file (.ico)." He says: "Its headers were modified to appear as a .ico file. However, the icon file is actually a password-protected zip archive under the password 'no-password.' It's a 64-bit, statically linked, stripped, and UPX-packed ELF file which is then extracted from the archive. When unpacked, there is another ELF executable that is the coin miner." He says: "All in all, there were four scripts downloaded. We're currently working on identifying more information about the campaign.

"At this point we're still conducting research to dig up more details on the attack, the breadth of the campaign, the payloads being used, et cetera. Since this campaign has such a broad scope, we felt it would be wise to share as soon as we became aware. This document will continue to be updated as we dig up more information. It's clear," he writes, "that the attackers went to great lengths to try to hide the intentions of their newly-created worm. They used hidden services on the Tor network to host their payloads and created deceiving Windows icon files in an attempt to throw off researchers and even system admins who are looking at logs. The prevalence of vulnerable Exim servers, 3,683,029 across the globe according to Shodan, allows attackers to compromise many servers in a relatively short time, as well as generate a nice stream of cryptocurrency revenue."

So that was last Thursday. On Friday, June 14th, the next day, Microsoft's TechNet blog posted: "Prevent the impact" - this is Microsoft. "Prevent the impact of a Linux worm by updating Exim." So Microsoft jumped onboard, saying: "This week MSRC confirmed the presence of an active Linux worm leveraging a critical Remote Code Execution vulnerability, CVE-2019-10149, in Linux Exim email servers running Exim version from 4.87 to 4.91. Azure customers running VMs with Exim 4.92 are not affected," okay, meaning not running the vulnerable version of Exim. However, those who are, are susceptible.

They wrote: "Azure has controls in place to help limit the spread of this worm from work we've already done to combat spam, but customers using the vulnerable software would still be susceptible to infection." Meaning if you're on Azure with a Linux instance and Exim 4.87 to 4.91, you want to fix that. They said: "There is a partial mitigation for affected systems that can filter or block network traffic via Network Security Groups (NSGs). The affected systems can mitigate Internet-based wormable malware or advanced malware threats that could exploit the vulnerability. However, affected systems are still vulnerable to Remote Code Execution exploitation" - and actually it's Remote Command Execution, as we know - "if the attacker's IP address is permitted through Network Security Groups."

So anyway, even Microsoft and Azure are seeing this. And on Saturday MSRC also confirmed that they have detected this worm targeting Azure customers. So anyway, what we were expecting did happen. The reason this is a worm is it takes a week to obtain access. That's what you want to use a worm for. Doesn't make any sense for a low number of servers to be trying to infect a large number of servers. You want exponentiation. You want servers that are infected to start looking for other servers.

We know that this thing takes seven days, right, because it uses a funky set of edge conditions on email delivery timeouts where you need to keep a connection alive, and then after seven days you then accept a fail. You then close it, accept a failure message, and then the remote server executes the command which you put in as the account

name on an email server that you control. When all that happens, you are in, but it takes time. So this is a perfect setup, a perfect scenario for a worm. And unfortunately it is loose. It is out there. It is scanning, and it is installing itself as a cryptominer on, wow, way more than 3.5 million currently vulnerable Exim servers. It's going to find them all. We've seen what Internet-scale worms do. They find them all. And this thing will.

So a week before this, at the announce time, there was a startling change in version number. There is a process, there's clearly a system in place for updating these Exim email servers because the percentage of them that are vulnerable dropped from 100% to about 25% overnight, in a day. But that still leaves plenty that are vulnerable. And whoever it is who sets up shop in these and is arranging to mint cryptocurrency and then to close the door behind them, they're probably going to be minting for quite some time because a lot of these systems are clearly unattended, and they've been forgotten.

**Leo:** Wow.

**Steve:** Yeah. It happened.

**Leo:** There you go.

**Steve:** An Internet-scale worm. We haven't had one for quite a while.

**Leo:** Seems like a really hard thing to implement, though. Have we seen any exploits? I mean, you've got to sit there for six days banging on this server.

**Steve:** Well, that's why you want other people's email servers to be doing it for you.

**Leo:** Yeah. Okay, that makes sense.

**Steve:** Yeah, no, I mean, it's spreading. It's in active use right now.

**Leo:** Oh, it is. Okay, okay.

**Steve:** Yeah, yeah, yeah.

**Leo:** Not hypothetical.

**Steve:** Nope.

**Leo:** All right. There you go. Steve Gibson, GRC.com. If you want to get transcripts of this show, 16Kb versions, or the full 64K audio, he's got it there, along with SpinRite, the world's best hard drive maintenance and recovery utility. Also lots of other stuff, including SQRL information, and soon links to SQRL PopSockets and SQRL T-shirts, I'm sure.

**Steve:** Yes. I just need to catch up.

**Leo:** Just get those up there, yes.

**Steve:** I've been racing behind the announcement ever since.

**Leo:** You can also find audio and video from the show at our website, TWiT.tv/sn. We do the show every Tuesday, about 1:30 Pacific, 4:30 Eastern, 20:30 UTC. You can watch us do it live or listen live at TWiT.tv/live. And if you're live, go in the chatroom. Join the folks in there who are watching live because there's always a conversation behind the scenes in our chatroom, irc.twit.tv. We will be back next week. Maybe you'll talk about Zuckbucks. I don't know. Maybe we'll talk about - what's that other thing that's going to happen, bad thing?

**Steve:** Oh, it's going to be the complete collapse of Linux.

**Leo:** Oh, yeah, that. Oh, yeah, Linux just ending. It's over. It's all over.

**Steve:** Linux SACK. All Linux servers will be kernel faulted, yes. Panicked.

**Leo:** Next week. I'll see you then, Steve.

**Steve:** Maybe.