Bluetooth Fingerprinting

Description: This week we quickly cover a bunch of welcome news on the combating ransomware front. We look at the results from last week's Pwn2Own contest in Austin Texas, and at a weird problem that only some users of Windows 11 started experiencing after Halloween. There's a serious problem with GitLab servers and additional supply-chain attacks on JavaScript's package management. Google fixed a bunch of things in Android last Tuesday, and Cisco has issued an emergency CVSS 9.8 alert, and U.S. federal agencies are being ordered to patch hundreds of outstanding vulnerabilities. We have some fun closing-the-loop feedback from our listeners. I'm going to share the details of an interesting IRQ problem I tracked down last week. Then we'll take a look at an aspect of radio frequency fingerprinting that has apparently escaped everyone's notice until seven researchers from UCSD did the math.

High quality (64 kbps) mp3 audio file URL: http://media.GRC.com/sn/SN-844.mp3
Quarter size (16 kbps) mp3 audio file URL: http://media.GRC.com/sn/sn-844-lq.mp3

SHOW TEASE: It's time for Security Now!'s 50th anniversary of a very important technology innovation. Steve and I will chat back and forth, some memories. Also I guess the headline of this show should be "Got 'em." Looks like they have captured some of the principals in one of the worst ransomware gangs of all time. And then Steve talks about an interesting research that shows that your Bluetoothed phone is basically announcing your presence at all times to anyone who asks. It's all coming up next on Security Now!.

Leo Laporte: This is Security Now! with Steve Gibson, Episode 844, recorded Tuesday, November 9th, 2021: Bluetooth Fingerprinting.

It's time for Security Now!, the show where we cover your security and privacy online with Mr. Steven "Tiberius" Gibson of GRC.com. First of all, hi, Steve. I didn't want you to have to sit there with your Vulcan salute for too long. You could get a cramp. I also want to thank Mikah for filling in. Did Mikah do the show?

Steve Gibson: Jason.

Leo: Jason did. All right.

Steve: Yup.

Leo: Jason for filling in last week. Did a wonderful job. I hear there was a bit of a rant.

Steve: Oh.

Leo: I'm sorry I missed that. Always enjoy those.

Steve: No, you're not. No, you're not sorry.

Leo: No? No?

Steve: Well, I mean, I started out by mentioning that I was self-conscious about having been ranting and so hard on Microsoft recently. And then I just completely jumped the shark. I just...

Leo: That's why we love you.

Steve: At least, the only thing I can say in my defense is that it's not a put-on; right? I mean, a lot of these guys who do that are just - it's a put-on. But I just - it's real. So, and actually we have some feedback from our listeners which demonstrates the degree to which they have come to know who their Security Now! host is we'll be getting to. But we have a great Episode 844 for this second week of - where are we? November, with the daylight savings having changed.

Leo: It's a question I ask every day. Where are we?

Steve: Where are we?

Leo: Where the hell are we?

Steve: So, well, and in your case it actually may have a different answer. I never go anywhere. But I'm happy to be here. We're going to talk about Bluetooth fingerprinting, something which is different in an interesting way than we've ever addressed before because it's not fingerprinting the digital information that's all been deeply crypto-ized. It's the analog information. It turns out not all Bluetooth radios are created the same. In fact, no two are created the same. And that leads to some interesting problems. But we're going to talk about, as you have already been, but we have to cover it here, a bunch of welcome news on the combating ransomware front. We're going to look at the results from last week's always entertaining Pwn2Own contest, which took place in Austin, Texas. And at a weird problem that only some users of Windows 11 started experiencing after Halloween. Go figure.

There's a serious problem with GitLab servers, and additional supply chain attacks on JavaScript's package management. Google fixed a bunch of things in Android last Tuesday, and Cisco has issued an emergency CVSS 9.8 which, you know, it only goes to 10. So that's - and I don't think I've ever seen anything other than 9.8. They must be

reserving 9.9 and 10 for, like, end of the Internet scale, you know, a so-called ELE, right, an Extinction Level Event. Also U.S. federal agencies are being ordered to patch hundreds of outstanding vulnerabilities, and you won't believe by when. We have some fun, as I said, closing-the-loop feedback from our listeners.

I'm going to share I think what our listeners will find some interesting details of an IRQ problem I tracked down last week. The old-timers here will remember those horrible days of IRQs on the ISA bus back in the early PC and XT days. Then we're going to look at an aspect of radio frequency fingerprinting that has apparently escaped everyone's notice until seven researchers from UCSD did the math. And we have a really cool Picture of the Week that I just wanted to sort of take a moment to acknowledge.

Leo: Yeah, I agree.

Steve: So I think another great podcast for our listeners. Well, and in fact the inheritor of this anniversary, of course, is a very popular OS that you yourself are in favor of.

Leo: Familiar with, yes, yes. And we should mention, nice new glasses.

Steve: Oh.

Leo: Yeah.

Steve: I'm surprised anyone noticed.

Leo: Oh, no, they're very distinctive.

Steve: Well, I was looking at Alex because I'm sort of conscious of them. They're huge.

Leo: I like them. They're big. They're a little large, yeah.

Steve: Well, and I was actually sitting at a different location in my home where I have a different set of hardware set up to test SpinRite. And I had a keyboard, like two drawers pulled out, and the keyboard was sitting on these drawers. And I thought, okay, let's just go to a different keyboard. So I put that keyboard down, the drawers were out, and I saw all of the previous pairs of glasses that...

Leo: You found your glasses drawer.

Steve: I found my glasses drawer. And I thought, huh, I wonder if any of those cases are not empty. And the first one I opened had these, and they're actually a better prescription now for me than the ones I was wearing.

Leo: Wait, wait. So these aren't new. They're old.

Steve: They're old, yeah.

Leo: That's funny.

Steve: And I just sort of put them on, and I thought, oh, I can see better with these. So I think maybe I'll wear them for a while.

Leo: I think you should wear a different pair every day from now on and throw everybody.

Steve: And what I was noticing was Alex's glasses are like, you know, the size of dimes on him.

Leo: Yeah, that's hipster. That's the hipster thing.

Steve: Okay, well, I am the reverse. We all know that already. I am the reverse of - yeah.

Leo: Do you have aviator, any aviator glasses in that drawer? Because I think you'd look good with aviator. In fact, doesn't our art - I think our artwork has aviator glasses, actually.

Steve: That artwork is not good either.

Leo: I know, you don't like it at all.

Steve: That's Snooty Steve.

Leo: Yeah, your nose is a little bit in the air.

Steve: You're lucky to be here.

Leo: Let me just run over to the TWiT Store and see what glasses Snooty Steve - oh, you're not wearing any glasses in that one. Okay. So that's...

Steve: Oh, thank goodness.

Leo: Yeah, yeah. So it's snooty, but glasses-free. That's, by the way, one of the best...

Steve: Apparently way down at the bottom of the page.

Leo: I don't know why because it's one of our best-sellers. I don't know why that wouldn't be at the very tippity-top.

Steve: Well, you put the ones that people want all the way at the bottom, Leo. That way they're forced to scroll through.

Leo: Oh.

Steve: Is that underwear?

Leo: It looks like a jockstrap; but it's not, it's a face mask.

Steve: Oh, thank goodness. Okay.

Leo: Yeah. Yeah. Why don't we have a Gibson face mask? That's what we really need.

Steve: Oh, that's all right.

Leo: Well, anyway. TWiT.tv/store.

Steve: Don't want to scare the children. Halloween is over.

Leo: Yeah. There it is. There's the Security Now! mug. I was just - I was looking at the wrong side. There it is. And you see no glasses, just a really snooty expression. Whoops. I hit too many...

Steve: Well, you know, I only started - I took my contacts out shortly after I - I don't know, I don't think there was any reason, but Lorrie never knew me with contacts. Everyone else did. And so it was only about four or five years ago that I stopped wearing the contacts. Because I'd worn them since I was in high school. I asked my best friend, we were driving in his parents' car, and I said, "So what do you think about contacts?" He says, "Oh, definitely." He was like, didn't even - not even a hesitation.

Leo: Oh, yes, definitely, Steve.

Steve: Okay, tell me what you really think, Scott.

Leo: You need them. No, no. I like any look you come up with. It's what's inside the noggin that matters.

Steve: We've got a lot of glass here now, so that's good.

Leo: You said there's a big anniversary.

Steve: There is. But first I want to note you have a little SQRL logo behind you.

Leo: Oh, you didn't notice that. Yeah, that's been there for ages.

Steve: Ah, I just saw it for the first time.

Leo: It's carved in wood.

Steve: Very nice.

Leo: I think, I'm trying to remember, was it Don [Name] who did that? I can't remember. He's our woodworker. But, yeah, isn't that nice?

Steve: Huh. Very cool.

Leo: Yeah.

Steve: It is nice.

Leo: Yeah.

Steve: So yes, a big anniversary. Well, yeah, it is big for the industry because of what's happened. Last Wednesday, November 3rd, was to whatever degree it's possible to identify a specific date, and in fact this date comes from the cover page of the Unix Programmer's Manual. And, you know, the people behind this are just - they're legends in the industry. This page just says K. Thompson. Well, we know that's Ken; right? Because like I said, legends. And D.M. Ritchie, well, that's Dennis. So it was 50 years ago that someone typed, clearly on a typewriter, "Unix Programmer's Manual, K. Thompson, D.M. Ritchie, November 3, 1971."

And that was the first edition of Unix. It ran on a PDP-11/20, which was not a big PDP-11. They got much bigger after that. This PDP-11, the target platform was a PDP-11/20 with 8K words. It was a 16-bit, the PDP-11s are 16-bit machines. So 8K words, in other words, 16K bytes of RAM, and that was the OS, and still left over plenty of room to actually run programs. So yep, those were the days when operating systems were, you know, the first Unix was written by hand in assembler. And I'm not exactly sure how C came along.

Leo: Yeah, I think they wrote C for it; right?

Steve: Yeah. But it might have been, I'm sure that the very first one was assembler. So maybe they recoded a later version in C. But then of course it absolutely was like the first major project written in C. And just a beautiful piece of work. Someone on Twitter who I may have thanked him for pointing that out to me, and he wrote back, and he said, "Hey, I know you've always been talking about PDP-8s, and you were active back then. You've never talked about Unix in those days."

And of course the problem was it wasn't available to hobbyists. This work was done at Bell Labs. And so it ended up being inherited, the license to Unix, by AT&T. And AT&T kept it as like a proprietary property for a long time. And in fact it was probably, I didn't check the history, but I would imagine it was Unix's proprietary nature that was one of the incentives for Linus Torvalds to say, "Hey, you know, that's a great OS, but no one can get it. So I'm going to write one." Which is very much the same philosophy as Unix, but open source and available to everyone. And as a consequence, we have a very mature Linux operating system.

And then there were some recodings also, like for example one of the licensees of AT&T's Unix was University of California Berkeley. They created their own BSD Unix, which was also licensed to the Regents of the State of California. And then a free recoding of the Berkeley BSD was done to create FreeBSD. So, I mean, there's been lots of forks and branches. In fact, I remember seeing a tree, I'm sure you've see it, too, Leo, back in the day, of sort of the genealogy tree of these operating systems, how they branched off of each other and what was related to what and what came first and second.

Leo: It's a pretty complicated tree, too.

Steve: It's a mess, yeah.

Leo: Yeah, yeah. You can browse the 1996 version of the Unix source code online if you search for Unix source code. And there's a commented version that was put up there by John Lions, and I highly recommend it. It's very interesting reading. That's in C.

Steve: Wow, that would be something to see.

Leo: Yeah. And, you know, it's only 9,000 lines of code. I mean, it's fairly compact. But it's a good way to learn how things have to be done, you know, how they can...

Steve: Yes. And it was sort of the original idea, you know. It was sort of the predecessor of the microkernel, the idea that instead of creating this monstrous operating system that does everything, you create a little supervisor that is able to hand out memory to tasks, that are able to create a task and manage it, and also share the system's time among the various things that are running together. But then of course, and the whole Unix concept was to then create a simple OS and then a large set of simple tools, each which did one particular thing really well. And then you would chain them together in various ways in order to solve problems.

Leo: Here's the original assembly code for the 1971 edition.

Steve: Yup. So it was originally in assembler, as I recalled, not in C.

Leo: Yeah, you're right, yeah, yeah.

Steve: Yup.

Leo: Look at that. All of it's online, which is great now. There's a great learning thing, you know, if you want to understand the evolution.

Steve: Well, so we have lots of welcome progress on the ransomware front, which all sort of came together. Much of the past week's security news were reports, as I said, of various counter-ransomware campaigns. Yesterday the U.S. Department of Justice charged a 22-year-old Ukrainian national with orchestrating the ransomware attacks which were facilitated by the flaws in Kaseya's servers. Following an arrest warrant which was issued by the U.S., that suspect, Yaroslav Vasinskyi, was detained last month by Polish authorities at a border station while he was crossing from Ukraine into Poland. So that probably surprised him. And in court documents which were unsealed yesterday, the DOJ said that Vasinskyi was a long-time collaborator of the REvil group.

Also the U.S. charged a second suspect that helped the REvil gang deploy its ransomware. Identified in court documents as 28-year-old Yevgeniy Polyanin, the DOJ said this Russian national also worked as a REvil affiliate. U.S. believes that Polyanin is the person who breached the network of TSM Consulting, which is a Texas-based managed service provider, from where he deployed the REvil ransomware on the internal networks of at least 20 Texas local government agencies on August 16th, 2019. So this was a few years ago. But again, what we're seeing is a clear ramping up and tightening of the screws. You know, the U.S. has said, okay, we're not going to just ignore this stuff any longer. Although Polyanin is still at large and wanted by the FBI, the DOJ did say that they had managed to successfully seize $6.1 million worth of cryptocurrency assets that he was holding in an FTX account. So they got the money.

The U.S. announcements which were just both recent, I think yesterday, they came hours after Europol had announced similar arrests in Romania, Kuwait, and South Korea. Seven members of affiliate groups who worked with the GandCrab and REvil ransomware programs were detained. So we're rounding them up, which is wonderful.

And yesterday, the U.S. Treasury Department imposed sanctions on the cryptocurrency exchange Chatex for facilitating financial transactions for ransomware actors. An analysis of Chatex's known transactions indicated that over half were directly traced to illicit or high-risk activities such as darknet markets, high-risk exchanges, and ransomware. Officials said that Chatex also had direct ties to Suex, which is a Russian cryptocurrency exchange portal which the Treasury sanctioned in September for the same reasons.

Treasury also sanctioned three Chatex suppliers. Actually they're part of - it's Chatex's infrastructure: IZIBITS OU, Chatextech SIA, and Hightrade Finance Ltd., identifying them as three companies set up as Chatex's infrastructure, which enabled Chatex's operation. Operations for Chatextech and IZIBITS have been suspended by officials in Latvia and Estonia, respectively. Latvian officials are currently working to identify Chatex's board owners, who are all non-Latvian nationals.

And to encourage more of the same, more arrests, the U.S. State Department announced bounties for information that may lead to the identification and/or arrest of members of the REvil/Sodinokibi ransomware group: $10 million for information on REvil's key

leaders and $5 million for information on REvil affiliates. And this follows last Thursday's identical reward announcements for any information which may lead to the identification and/or arrest of members of the DarkSide ransomware group, same terms and conditions.

And the last piece of welcome news from the ransomware universe is that the BlackMatter ransomware-as-a-service group announced last week that it was shutting down its operations as a result of pressure from authorities. So better to slink off into the night rather than be dragged away in handcuffs. And it seems pretty clear that the U.S. and its global partners are seriously turning up the heat on this whole ransomware business. And as a result, I think it's becoming much less clear today that hacking and extorting is an easy and safe way to make a buck on the underworld; you know?

Leo: That's so important. That's really good because I really do think there was this sense for a long time, oh, you know, you can do it with impunity. You know?

Steve: Yeah, exactly, just go in and extort and get paid.

Leo: A couple of months ago Lisa and I were watching a movie about armored car robberies. And I was thinking, why would you do that when it's so dangerous, and you can pretty much, without any trouble at all, steal much more money and walk away?

Steve: And be located anywhere on the globe.

Leo: Any, yeah.

Steve: In any dark corner.

Leo: Yeah. So I'm glad that there's a - there should be a consequence, and I'm glad that they're starting to - it's actually, I'd love to know the story of how they catch these guys because it cannot be easy.

Steve: Well, and certainly they've been under the misapprehension, the bad guys, that they can't be caught, that they are, like, the long arm of the law cannot reach them. And what they're discovering is, wait, where did our cryptocurrency go? That's not fair. It's like, uh-huh. Right. And extorting was? So, yeah.

Leo: Lawrence Abrams, as we know from BleepingComputer, tweeted recently, "REvil ransomware's Tor data leak and payment sites are now showing a page titled 'REvil is bad. They are not the masters they think they are. We have the skills and experience. Do you want to be with the most qualified or losers?'" So maybe somebody else has taken over that page.

Steve: Yeah, they probably let their domain, you know, their...

Leo: It's an onion domain.

Steve: Their onion domain lapse or go public.

Leo: Yeah. Wow.

Steve: Wow. Well, we do have some very skilled good hackers, and Pwn2Own, really, that and the Tianfu Cup in China, brings them out. Last Tuesday through Thursday was the largest ever three-day Fall 2021 Pwn2Own. Since its inception, the Fall Pwn2Own contest has focused on consumer devices while the contest location has wandered around the globe. Nine years ago, the 2012 Amsterdam Pwn2Own, which was the first one, targeted only mobile phones. That's where it began. And we talked about it on this podcast. In the years that followed, the context grew to include smart TVs, wearables, smart speakers, and other appliances.

Last year's contest, which was held in Toronto, was further expanded to include Network Attached Storage (NAS) devices. And this year's 2021 contest, as I said, which occurred in Austin, Texas, added, sort of because of the post-COVID home office environment that has become substantially more busy, expanded the router category and also added printers. So in all, 22 devices were available as targets, with more than $500,000 USD in prize money total. But if there were multiple attacks of different sorts against the same device, it was the device that had the prize. So as we'll see, more than a million dollars was actually won by the contestants.

Now, in the past, we've had fun recreating a running commentary of the event. But as I scanned through it, it was clear that the contest has grown so large that doing that again would take up at least half the podcast.

Leo: Yeah, you can't read the results.

Steve: So it's just not practical. So I'm going to mention the equipment under attack and hit the high achievement points from the three-day event. As I said, Pwn2Own has its root in mobile handsets. So that category remained well represented with the top three: the Google Pixel 5, the Samsung Galaxy S21, and Apple's iPhone 12, each handset running the latest version with all of their operating systems installed, and all available updates patched. So absolutely right current as of the contest day.

And of course as we know in the past that's caught out some of the contestants where an official patch would be released a day before the contest, and it kind of raised some eyebrows. It's like, uh, wait a minute. You know? Did Apple do that on purpose? How foxy are they? Because one of the contestants was all set to blast an iPhone, and when it came to the actual day of the contest, it no longer worked due to a patch that had just happened.

As we know, this was the year of print spooler bugs. But the competition wanted to see about the devices themselves. So three printers from HP, Lexmark, and Canon were on the chopping block. The HP Color LaserJet Pro MFP M283fdw, and wait till you hear what the F-Secure guys did to that poor printer; Lexmark's MC3224i; and Canon's ImageCLASS MF644Cdw.

In the home automation category we had the Facebook Portal, Amazon's Echo Show 10, Google's Nest Hub (2nd Gen), the Sonos One Speaker, and Apple's HomePod mini. There

were two smart TVs: the Sony X80J Series, a 43-incher; and the Samsung Q60A Series, also 43 inches. They were both put to test. With all the problems we've seen from router compromises, five routers were made available: the TP-Link AC1750 Smart WiFi Router turned out not to be so smart; Netgear's Nighthawk Smart WiFi Router, that was their R6700, which was an AC1750; Cisco's RV340, which we've talked about before, actually; MikroTik's RB4011iGS+RM; and Ubiquiti Network's EdgeRouter 4. Representing the network attached storage category was Synology's DiskStation DS920+. Western Digital had two entries, the My Cloud Pro Series PR4100 and also their 3TB My Cloud Home Personal Cloud. And last but not least, the single entry in the external storage category was SanDisk's Professional G-DRIVE ArmorLock SSD 1TB.

Leo: Oh, I use that.

Steve: Okay. I think it stood up. I don't remember any successful attack on that.

Leo: Good.

Steve: Okay. So what happened? The super-abbreviated version, as I said, and I'll expand upon that, make it a little less super-abbreviated, is that the uber-talented participants in this year's event revealed for the first time ever their discoveries of 61 brand new exploitable vulnerabilities across that range of fully patched commercial products, and the participants collectively took home $1,081,250, for the second Pwn2Own in a row to clear the $1 million mark. And, okay. For ranking the participants, recall that the way this works is that awards take the form both of a cash prize and Master of Pwn points, which are totaled to determine each year's Master of Pwn winner.

The French offensive security firm Synacktiv topped the three-day contest's leaderboard by winning 20 Master of Pwn points and earning themselves $197,500. They netted maximum points for a zero-day vulnerability in the Sonos One smart speaker. They also successfully, well, actually in that one they successful demonstrated seizing full control of the Sonos One through a stack-based buffer overflow flaw, and that earned them six points and $60,000 of their total winnings. They also scored another four points and $40,000 from leveraging a configuration flaw which gave them code execution on WD's My Cloud Pro Series PR4100 NAS.

Trailing Synacktiv in second place by only two points were joint winners of the flagship Spring event DEVCORE, who earned 18 points and $180,000 in total. Together with his fellow DEVCORE members, Orange Tsai - and remember it was he who discovered what was described as at the time "a whole new attack surface" on Microsoft Exchange Server last year, which of course brought us all of those problems with Exchange Server. They claimed maximum points for compromising the Sonos One as well, along with four additional points and $40,000 after combining an out-of-bounds read and out-of-bounds write flaws to hack Western Digital's 3TB My Cloud Home Personal Cloud device.

STAR Labs, which finished third overall, chained an out-of-bounds read with a heap-based buffer overflow on the beta version of also the Sonos One, earning five points and $45,000. Fourth on the final standings was Sam Thomas from the U.K. infosec firm Pentest Ltd. Sam earned $40,000 and four points after chaining three bugs to get code execution on WD's PR4100 NAS. So I have a link to the detailed blow-by-blow rundown in the show notes. And it really is quite something. And although he didn't get into the top four for points, to give you a sense for how much action there was, one researcher named Bien Pham from Team Orca of the Sea Security, he really had his...

Leo: [Crosstalk].

Steve: I know, Orca of the Sea Security.

Leo: Oh, man, they've got to be [crosstalk].

Steve: There was something about their domain name, too, I don't remember now, that was kind of cool. But anyway, he really had his way with the routers he attacked. In his first event, which as I recall was at 10:00 a.m. the first morning, he leveraged a logic error to compromise the WAN interface of the Cisco RV340 router to win himself $30,000 and three Master of Pwn points. Two hours later, he used a three-bug chain, including an off bypass and a command injection to take over the LAN interface of that same Cisco RV340 router to earn an additional $15,000 and two more Master of Pwn points. And finally, he finished the first day by using an out-of-bands read bug to take control of that TP-Link AC1750 router through its LAN interface to earn himself an additional $5,000 and one Master of Pwn.

Many of the other individuals and teams demonstrated similar quite impressive, or horrifying depending upon your position, skill. And in the process they successfully demonstrated the exploitation of, as I said, 61, a total of 61 new previously unknown vulnerabilities, which was about twice the previous record for any Pwn2Own. As always, the participants immediately provide full disclosures to each of the affected vendors and will withhold their public disclosure for 120 days, after which, patched or not, they're free to disclose the technical details of their wizardry.

And I mentioned this HP printer. After the competition, Dustin Childs, who's the communication manager for ZDI, the Zero-Day Initiative, who holds the Pwn2Own contests, was asked to name his favorite exploit. He replied: "It's hard to beat an exploit that turns a printer into a jukebox and plays AC/DC."

Leo: Oh, they missed a bet. They really should have played Rick Astley, "Never Gonna Give You Up."

Steve: Yeah, you're right. Dustin was referring to the impressive work of the three-man F-Secure Labs team who targeted the HP Color LaserJet Pro MFP M283fdw to make it play music, or at least make it play a lot of noise depending upon how you feel about AC/DC.

Leo: Did they say which AC/DC song?

Steve: I looked for it, and I didn't...

Leo: "For those about to rock, we salute you."

Steve: Although you can find the whole thing on YouTube. The entire event was streamed. So if you're interested, Leo, while I'm continuing to tell our listeners about...

Leo: I will see if I can find it, yes.

Steve: ...other things, yes. And you may just be able to google or just search YouTube for that particular snippet. And speaking of snippets, Windows 11 snipping tool, its emoji picker, and other parts were failing, or actually maybe are until today. Today being, by the way, Patch Tuesday for November. So we'll see what gets fixed. We'll talk about that next week.

The Verge carried an intriguing report last Thursday that some apparently quite selective parts of Windows 11 were failing after Halloween, October 31st, so exactly after the end of the month. The Verge's headline provides our first clue. It read: "Microsoft warns Windows 11 features are failing due to its expired certificate." So, okay, that's what they said. But it's unclear what "its" refers to here. The Verge wrote: "Microsoft has started warning Windows 11 users that certain features in the operating system are failing to load due to an expired certificate. The certificate expired on October 31st, and Microsoft warns that some Windows 11 users aren't able to open apps like the Snipping Tool, the touch keyboard, or the emoji panel.

"A patch is available to fix some of the issues, but it's currently in preview, meaning you have to install it manually from Windows Update. The patch is KB4006746. It will fix the touch keyboard, voice typing, the emoji panel, and issues with getting started and the tips sections of Windows 11. You'll be able to find this patch by checking for updates in the Windows Update section of the Settings, Windows Update under Windows 11."

But here's what's weird. Microsoft's patch doesn't address the problems with the Snipping Tool app. Microsoft says: "To mitigate the issue with Snipping Tool, use the Print Screen key on your keyboard..."

Leo: Yeah?

Steve: Uh-huh, "...and paste the screenshot into your document."

Leo: Yeah?

Steve: "You can also paste it into Paint to select and copy the section you want." Okay. So in other words, don't use the Snipping Tool.

Leo: Don't use it. Do what you used to do before the Snipping Tool.

Steve: It broke.

Leo: It broke.

Steve: Yeah. So, okay. So I thought about this. It's not clear how many Windows 11 users are affected by these issues, and we haven't been able - oh. I'm sorry. The Verge said: "It's not clear how many Windows 11 users are affected by these issues, and we [The Verge] haven't been able to replicate the Snipping Tool problems on multiple patched systems."

Leo: Oh.

Steve: I know. So again, Leo, like I said, I'm not going to let myself get wound up about this again. But what? Anyway, if you're having issues, some Verge readers have reported being able to change the system date back to October 30th, then launch the Snipping Tool to get it working again. You can then, with it launched, change the system date back once the app has loaded successfully.

Leo: This just in. I believe I have found from Pwn2Own...

Steve: Oh, joy. Oh, joy.

Leo: I believe I might have found - I'm getting some music from something else. So let me figure out what that is. Oh, there we go. This is a tweet, and I believe this is the actual - we'll see what kind of music it plays. That's a little - it is, it's AC/DC, coming out of a printer. That's a microphone picking it up, coming out of the - okay.

Steve: Okay. So the printer doesn't have a speaker.

Leo: No.

Steve: They actually made the hardware of the printer do that.

Leo: Yeah, yeah, yeah.

Steve: Wow.

Leo: That's impressive as hell.

Steve: Wow.

Leo: I can't quite make out the song, but it does sound like AC/DC.

Steve: Wow. Well, and we were talking about old computer systems. And of course that was an old hack in the old days.

Leo: Yes.

Steve: You would disengage the clutch on an IBM 1403 printer, and then you could get it to play. Like by timing the hammer strikes on the printer, you could get it to do a very good job. But that's something.

Leo: We think this is "You Shook Me All Night Long," but we're not - oh, thunder. It is. Thunder. All right. I recognize it, yeah. Okay.

Steve: Wow.

Leo: It's "Thunderstruck."

Steve: Okay. So back on this weird Windows 11 problem, the expired certificate is also causing...

Leo: I hope we don't get taken down on YouTube now because we played it, a printer version of "Thunderstruck."

Steve: I don't think anything could even recognize...

Leo: I don't think so. I think we're safe. All right. I'm sorry. Back to the certificates. Sorry about that.

Steve: "The expired certificate is also causing issues with the accounts page in the settings section of Windows 11 with S mode enabled and the input method editor UI. It's not clear when the Snipping Tool and S mode issues will be addressed," said the Verge. Microsoft says what they always say: "We're working on a resolution for Snipping Tool and the S mode-only issues and will provide an update when more information is available."

Okay. So if we on this podcast reverse engineer the trouble, it sounds as though some Windows 11 apps have themselves been signed in such a way that their own digital signatures will not validate as of November 1st, 2021. But the way Authenticode signatures are designed, if the signature was valid at the time of signing, the signatures themselves never expire. That is, unlike other certificates where the certificate needs to be valid at the time it's used, because Microsoft recognized that code might well be used after the certificate that signed it had expired, then we had to change the rules.

So it cannot be that the signature itself has expired because that's all right. But like any digital signature, it's based on a chain of trust which typically has an intermediate certificate and a root certificate. So if any certificate in the chain being used by any of those apps which are all signed had expired, even though the app's signing certificate itself was still technically valid, Windows would not be able to validate its signature and would refuse to run the app.

Now, as to why only some Windows 11 users are experiencing this, and why among those who are, only some of their apps are seeing this, and why a fix was quickly deployed for most of the apps but not the Snipping Tool and apparently S mode stuff, this is, as I said, additional evidence that for whatever reason Windows is growing, let's just say, ever more complex. I use and love Windows. I dearly hope they're able to hold it together. Time will tell. And next week we'll see what adventures today's Patch Tuesday brought. Oh, and by the way, Leo, last week you missed the news that we had - what we talked about the week before, hoping that the printer issues had finally been resolved, as Microsoft said, but no.

Leo: No, of course not. That was a safe bet.

Steve: So maybe that's happening today, and we'll find out next week. So here's a sales pitch for GitLab servers. A page on TechRepublic introduces the reason for wanting one, as follows: They say: "If you're a Git user, you know that having local repositories that can be accessed via a local LAN or external WAN is a crucial element of the development process. You can certainly opt to go with GitHub, but that negates the ability to host locally. So when you want to host your own repositories, where do you turn? In a word, GitLab. GitLab allows you to host an on-premise Git repository that can be accessed from either your local LAN or, if you have an available public IP address, from outside your company. GitLab is fairly easy to install and incredibly simple to use."

Leo: Yeah, I like GitLab.

Steve: Right.

Leo: Yeah, I use it. Am I in trouble?

Steve: And unfortunately...

Leo: Uh-oh.

Steve: Depends upon when you last updated. Unfortunately...

Leo: It's on my Synology. It's the built-in kind of git for Synology. That's why I have it.

Steve: Unfortunately, unpatched GitLab servers also contain a now widely exploited and widely known flaw that's been used and is being used to create multi-thousand member botnets which are generating in excess of, yes, one terabit per second of DDoS attack traffic.

Leo: Oh, my god.

Steve: And why? Because they're not only easy to deploy, that is those GitLab servers, but "Wouldn't it be just great to put this on the WAN?" Right. And this is where we all say, "What could possibly go wrong?" And oh, my goodness.

So here's the situation: Bad guys are exploiting a security flaw in these GitLab self-hosted servers to assemble botnets and launch ginormous DDoS attacks, some in excess of, and they've been clocked, at a terabit per second. Damian Menscher, a Security Reliability Engineer at Google Cloud, who's responsible for Google's DDoS defenses, disclosed last Thursday that attackers are exploiting, and this is a CVE-2021, it's 22205, a vulnerability that GitLab patched back in April of this year. But of the 60,000 GitLab

servers publicly exposed to the Internet, and all our listeners know where this is going, only about half have since been updated with the patch.

The flaw exists in GitLab's ExifTool, which is a library commonly used to remove the metadata from images uploaded to web servers. It was discovered by William Bowling and reported to GitLab via GitLab's bug bounty program which they have over at HackerOne. In a report filed via HackerOne, Bowling said he discovered a way to abuse how the ExifTool handles uploads of, and I'd never run across this, a DjVu file format. Anyway, it's used for scanned documents. And because of a flaw in this ExifTool, he's able to gain control over the entire underlying GitLab web server. Which is to say, if you had a GitLab web server that was exposed to the 'Net, and 60,000 are, 30,000 have not been fixed, you can take it over.

Public proof-of-concept code for the vulnerability appeared in June, around the same time that the Italian security firm HN Security first spotted attacks. At the time, an HN Security researcher said the company began an investigation after spotting randomly named user accounts being added to compromised GitLab servers. Those were accounts that were most likely created by the attackers to allow remote control of the hacked systems. While the purpose of these intrusions at the time were unclear to HN Security, yesterday Google's Menscher said the hacked servers were part of a botnet comprising "thousands of compromised GitLab instances" that was launching large-scale DDoS attacks.

So as we've so often observed, botnet operators are exploiting the tardiness of individuals and enterprises around the world when it comes to patching their software. And we'll be talking about the order from the CISA about that in a minute. But in this case, this is in-house GitLab servers. According to Rapid7's analysis last Monday, and this is where we got the number, 60,000 GitLab servers are connected to the Internet, half unpatched. And it's worth noting that GitLab is not the only user of this ExifTool. So the exploit of it might very well impact other types of web applications where that tool might be part of the image upload processing path. So other exploitation could be forthcoming; and other things might need patching, as well.

It was noted that one way to prevent attacks might be to prevent uploading of DjVu - yeah, I don't need those uploaded - at the server if companies don't need to explicitly handle that file type. On the other hand, if you're going to do that, why not just update GitLab to all of its current patches and be done with it. So anyway, if anybody listening to this, and I wouldn't be surprised if some of our listeners have, certainly all of our listeners, Leo, would have patched this back in April.

Leo: Oh, yeah, yeah.

Steve: When it first happened. No doubt.

Leo: I was just checking. Mine's up to date, yeah.

Steve: But just to be sure.

Leo: Yeah. Sometimes people put servers up and never pay any attention to them ever again.

Steve: Yeah, yeah. I'm sure it happens. Okay. More supply chain attacks. Successful attacks on the software supply chain continue to be discovered. Obviously, this is a trend that has basically evolved during 2021, and I'm afraid it's going to be with us as all the bad trends tend to be. So the uncomfortable thing about this is when you discover an attack that's been going on for some time. It begs a question, what other attacks haven't you yet discovered? Right?

So in this most recent instance of discovery, a pair of extremely popular JavaScript npm packages named "coa," C-O-A, and "rc," which boast weekly combined downloads of 23 million, so 23 million downloads per week, were discovered to have been infected with password-stealing malware. The security team responsible for the npm JavaScript package manager are the ones who discovered it, and they've warned users that two of its most popular packages had been hijacked by some unknown threat actor. As I said, the two affected packages are "coa" and "rc." Coa, C-O-A, is a command-line argument parser with around 8.8 million weekly downloads. And rc is a configuration loader with about 14.2 million weekly downloads.

The list of versions basically are all those that are in current use: 2.0.3, 2.0.4, 2.1.1, 2.1.3, 3.0.1, and 3.1.3. And as I'll note in a second, they've not been changed for a while. Those are the coa versions. There are three versions of rc: 1.2.9, 1.3.9, and 2.3.9. Both packages appear to have been compromised at the same time and were the result of attackers gaining access to a package developer's account. So unfortunately, that's currently the Achilles heel. You might have vulnerabilities in the packaging system itself. Or you get in and impersonate the owner of the account and then mess with the things they have access to.

Anyway, once inside, the threat actor added a post-installation script to the original codebase. The post-installation ran an obfuscated TypeScript which would check for the operating system details and download, depending upon what it found, either a Windows batch or a Linux bash script. The deobfuscated version of the Windows batch script revealed that in the case of Windows, the compromised packages would download and run a DLL file containing a version of the Qakbot (Q-A-K-B-O-T) trojan. What a mess.

Okay. The compromise to coa was spotted first after its new installation routine started crashing build pipelines for React-based applications. Last Thursday the npm team tweeted shortly after detecting the coa compromise, which was triggered by a wave of reports about failed builds. They tweeted: "The compromised developer account has been temporarily disabled, and we are actively investigating the incident and monitoring for similar activity. We will share additional information as appropriate based on our investigation." That was from npm. They tweeted @npmjs on the 4th. The matching compromise to the rc package was discovered a few hours later.

The npm security team removed all the compromised coa and rc versions to prevent developers from accidentally infecting themselves. But it appears unlikely that either of the compromises had much actual chance of slipping through. First of all, builds using it were crashing. Also, both libraries are widely used. The malicious code was not well hidden, and both libraries had not seen any new releases since December 2018 and 2015, respectively. So all those factors alone would have raised sufficient suspicion to trigger a security audit within most professional developer teams. You know, if they saw something that hadn't been changed in four years suddenly got a new version, it's like, uh, what? Let's look into this a little more closely.

And one last interesting point. The malicious code present in these incidents is nearly identical to the code used in the compromise of the User-Agent parser. Remember UAParser library? We talked about it recently. That occurred late last month. So it appears that someone, and apparently someone either not highly skilled or someone maybe in a hurry, has set their sights upon JavaScript and the npm supply chain for

exploitation. And if that's the case, and anybody is like pulling packages from npm, I would check for things that have not been modified in a while that suddenly get an update, and see what's behind the update.

As I mentioned, it's Tuesday. It's Patch Tuesday for Microsoft, as well as for many others in the industry. So next week we'll check back to see what new adventures in computing have been created by today's updates. But last Tuesday Google rolled out its monthly security patches for Android, in the process fixing 39 flaws, including a zero-day that it said was being actively exploited in the wild in limited targeted attacks. That one was being tracked as CVE-2021-1048. It's a use-after-free vulnerability in the kernel that can be exploited to obtain local privilege escalation. And since we depend upon code containment to a great degree in today's computing, although privilege escalation is not as scary sounding as remote code execution, we know it's important. And the very fact that this flaw was being used in the wild demonstrates that it was doubtless quite useful for some nefarious purpose.

Aside from that zero-day, last Tuesday's patches also foreclosed on a couple of critical remote code execution (RCE) vulnerabilities in the System component that, as I said, could allow remote adversaries to execute malicious code within the context of a privileged process by sending a specially crafted transmission to targeted devices. So, cool that that was found. And that was not known to be exploited.

Two other critical flaws in Qualcomm's closed-source components of Android were also fixed, along with a fifth critical vulnerability in Android TV which could have permitted an attacker in close proximity to silently pair with the TV and get arbitrary code execution privileges not requiring any further authentication.

And interestingly, in terms of actively exploited, in-the-wild, total zero-days found and fixed so far this year, when compared with Windows, Chrome, and even iOS, Android has been faring surprisingly well. After counting last Tuesdays latest addition, Android has only needed to address a total of six zero-days this year. And it's not as if there isn't tremendous pressure to pry into Android. We know there is. So way to go, Google. I don't remember the count, Leo, last week, but I think it was 18 we're up to. Oh, no, 14 and 15. So 15 total zero-days in Chrome this year.

Leo: Jesus.

Steve: I know.

Leo: Oh, my god.

Steve: It's been bad. But only six zero-days in Android for the year. So again, props to them.

We had another problem with a default key in a Cisco device. Last Thursday, Cisco released an update to fix one of those very rare CVSS 9.8 vulnerabilities in their Policy Suite products. The announcement of this is a bit misleading. They said, and I'm quoting: "A vulnerability in the key-based SSH authentication mechanism of Cisco Policy Suite could allow an unauthenticated remote attacker to log into an affected system as root. This vulnerability is due to a weakness in the SSH subsystem of an affected system. An attacker could exploit this vulnerability by connecting to an affected device through SSH, naturally. A successful exploit could allow the attacker to log into an affected system as

the root user. Cisco has released software updates that address this vulnerability. There are no workarounds that address it."

Okay, now, you don't get to say that "A vulnerability in the key-based SSH authentication mechanism could allow an unauthenticated remote attacker to log into an affected system as the root user" when the vulnerability, air quotes, in question turns out to be a hard-coded preset factory default SSH key which is readily discoverable in the device's firmware. So is this vulnerability as they say due to a weakness in the SSH subsystem of an affected system? Maybe. It would be a weakness in the SSH subsystem that it contained a default key.

Leo: Private key?

Steve: Yes. Yes.

Leo: Jesus.

Steve: The SSH key was like, not generated at runtime or install time. There was just one, and all those devices shared the same one. I know. The good news is this was not discovered being used in the wild. To their credit, Cisco discovered this problem on their own. So I want to give them props for looking at their own code, presumably, hopefully, being horrified by what they found and addressing the problem. After updating their devices, the new installation process creates unique - imagine that - SSH keys on the fly, rather than shipping every device with the same default starter key.

But the concern is this keeps happening to Cisco. We've talked about many similar default login credential problems in the past. Again, they audited them; right? But the bigger and more important question is how this horribly weak design pattern ever became policy in the first place. It had to have been because so many devices have had this problem. We know that the awareness of security has been growing over time. But was there ever a time when shipping enterprise-class networking equipment with factory preset SSH credentials would have been reasonable?

And the other worry is that the previous instances of Cisco's default credentials were quite a while ago. How is it that the discovery and remediation of those several years ago would not have triggered a full cross-product-offering audit? And that only now, a couple years later, another similar problem is being found? You know, I've said this in the past. It's good that they're checking their stuff. But it would be good to know that they found them all. So one fewer problem out in the field, for those who update.

U.S. federal agencies have been ordered to patch hundreds of actively exploited flaws. To which I say, yeah, okay, good luck with that order. One way, as we know, to fight the threat of foreign cyberattacks such as ransomware is to go after the individuals behind the attacks. And that's how we started the podcast today. But the other way is to turn inward and examine how those attacks are being so successful in the first place. To that end, CISA, our awkwardly named U.S. Cybersecurity and Infrastructure Security Agency, last week published a catalog of known, exploited, and patched vulnerabilities, including those from Apple, Cisco, Microsoft, and Google. And in addition to this catalog, CISA has issued an order requiring all federal agencies to prioritize applying patches for those security flaws within "aggressive" timeframes. I'll say it's aggressive.

The Binding Operational Directive issued last Wednesday said: "These vulnerabilities pose significant risk to agencies and the federal enterprise. It is essential to aggressively

remediate known exploited vulnerabilities to protect federal information systems and reduce cyber incidents."

The catalog lists around 176 vulnerabilities identified between 2017 and 2020, and 100 flaws from 2021 this year alone. And the catalog will be updated with additional actively exploited vulnerabilities as and when they become known, provided they've been assigned Common Vulnerabilities and Exposures (CVE) identifiers and have clear remediation action.

And here's the kicker: The binding directive mandates that security vulnerabilities discovered this year, in 2021 - those are the ones being given the highest priority - must be addressed by next Wednesday, November 17, 2021. In other words, at the time of the issue, which was last Wednesday, the federal agencies were given two weeks to patch all of these, all 100, while setting a patching deadline of May 3, 2022 - so way out - for the remaining much older vulnerabilities. Although the Binding Operational Directive is primarily aimed at federal civilian agencies, CISA is recommending that private businesses and state entities review the catalog and remediate the vulnerabilities to strengthen their security and resilience posture. To which I say, yeah, okay.

The new strategy also sees the agency moving away from a strict severity-based vulnerability remediation prioritization to those that pose significant risk and are being abused in the real-world intrusions in light of the fact that adversaries do not only use critical weaknesses to achieve their goals. Some of the most widespread and devastating attacks have chained multiple vulnerabilities rated individually only high, medium, or low.

Tim Erlin, the VP of Strategy for Tripwire, said: "This directive does two things. First, it establishes an agreed-upon list of vulnerabilities that are being actively exploited. Secondly, it provides due dates for remediating those vulnerabilities. By providing a common list of vulnerabilities to target for remediation, CISA is effectively leveling the playing field for agencies in terms of prioritization. It's no longer up to each individual agency to decide which vulnerabilities are the highest priority to patch." And to that I'll also add that it will presumably give federal IT departments some much-needed leverage over their own management, who might otherwise not be providing them with the time, talent, and budget that they require to meet a much more nebulous, well, is everything patched.

So now there is a mandate with a deadline. So the IT department can say, "You want us to meet this order's deadline or not? Here's what it's going to take." So probably a good thing, to light a fire. And depending upon how much clout CISA ends with, or ends up with, you can imagine that the government can begin to apply some pressure at some point after this order deadline has passed.

Twitter was the conduit throughout all of the past week for showing me how much this podcast's listeners have grown to know me, Leo. A barrage of welcome tweets all came in saying the same thing. I lost count of the number of people who wrote very close variations of "I can just hear you saying, 'What could possibly go wrong?'" And then they all attached links to various tech news coverage that during last week's Ignite conference Microsoft announced that Excel would now be supporting JavaScript.

Leo: Oh, boy. Oh. What could possibly go wrong?

Steve: So indeed, yes, what could possibly go wrong? It seems like a great idea, Microsoft, for keeping this podcast going for another 17 years.

And then we even had the meta-tweet from Joel Pomales, who wrote: "Pretty sure @SGgrc is getting bombarded with 'What could possibly go wrong?' tweets." So that was a tweet about the fact that there were doubtless going to be tweets, as there were.

And in another example of knowing me so well, Kevin Jones tweeted. He said: "I thought @SGgrc would appreciate this." And then he attached a tweet from an account called "I Am Developer," and it's @iamdevloper. And it's great. So it shows: "1969: What are you doing with that 2KB of RAM? Answer: Sending people to the moon." Uh-huh. "2017: What are you doing with that 1.5GB of RAM? The answer: Running Slack." Right.

Leo: Well, Slack might be more complicated than sending people to the moon, I guess.

Steve: Ugh. Anyway, it's progress. And following up on last week's Trojan Source topic, Joseph Lee tweeted: "@SGgrc GitHub now has a warning about hidden Unicode directional control characters for source code." And in the Halloween October 31st GitHub Changelog they said: "Warning about bidirectional Unicode text. A warning is now displayed when a file's contents include bidirectional Unicode text. Such text can be interpreted or compiled differently than it appears in a user interface. For example, hidden bidirectional Unicode characters can be used to swap segments of text in a file. This can cause code to appear one way and be interpreted or compiled another way." And of course that was last week's topic for the podcast, Trojan Source.

Some enterprising guys, two guys in the U.K. figured out that you could use the right-to-left reading and left-to-right reading escape characters to swap pieces of text in source code. It would look perfectly fine to anyone reviewing the code, but it would act very differently when it was compiled. So a very cool topic, and nice that GitHub - we're seeing indications that the industry is taking this as seriously as I think they should.

Okay. I'm still dealing with the fallout from SpinRite's fifth technology development release. Actually I may now just barely be past dealing with the fallout. But we learned something of phenomenal importance last week. There are motherboard and add-in adapter BIOSes which do not correctly report the hardware interrupt their devices use. I received so many notes from our listeners saying that they really enjoyed my discussion of tracking down that problem with the ThinkPad whose NVMe controller had kind of died in a really weird way, and then coming up with a way to recover its BitLocker-encrypted drive contents when I didn't have any recovery key, that I'm not going to shy away from a bit of technical talk here.

Several testers of releases 4 and 5 reported hangs or timeouts on their systems. On release number 5 we had many fewer problems than on release 4 since I had found and fixed the trouble I was having with Intel chipsets when configured for ATA, but not AHCI, operation. But the developer pre-release number 5 was still having some hangs. I believe that any problem, as a developer, any problem I can reproduce, I can fix. And I was unable to make any such hang happen.

So I purchased one of the same adapters that one of our testers was using that was having trouble. It was a little IO Crest adapter - I got it from Amazon for I think it was 20 bucks - using a Marvell chip and offering both serial SATA and parallel old-school IDE parallel cable connections. If drives were connected to its SATA ports, no problem. SpinRite cruised right through. But attach a drive to its IDE parallel cable and, blammo, a hard full-system hang that only the reset button or a full power cycle would end.

As I stepped through the code, instruction by instruction, the moment I tried to execute an instruction that had the controller send a command to its drive to actually do

something, the debugger never returned from stepping into that instruction, and the entire system was locked up hard. The only thing I could do was reset. And the only thing that could do what I saw was a hardware interrupt.

Hardware interrupts are such an incredibly powerful innovation in computing that it's difficult to imagine life without them. The UNIVAC 1103, from 1953, is generally credited with the first use of hardware interrupts. And since that was also so near the birth of computing, it's clear that the early pioneers of computing themselves quickly realized the limitations of purely sequential instruction-by-instruction program flow. Normally, one CPU instruction follows the next, with programmed jumps and loops, both conditional and unconditional, to facilitate complex logic flows.

A hardware interrupt does exactly what its name says. Some hardware event interrupts that pre-programmed flow of instructions. When the interrupt occurs, the CPU's program counter is saved, and a new program location is loaded into it. Thus the hardware event causes the computer to instantly jump to some other location of its code. An example might be to count the ticks of a hardware clock in order to maintain the time of day for the system. Once that clock tick has been counted, the hardware interrupt would be ended we say that the interrupt has been "serviced" by restoring the program counter to its previously saved value, which causes the CPU to resume executing code from the point of its interruption as if nothing had happened at all. It's completely transparent from the standpoint of the code.

And of course we old-timers will recall the early days of the PC with its ISA bus, when interrupts were a real problem. There were only 15 hardware interrupts available back then, with the system's clock and its keyboard always occupying the first two. But pretty much every peripheral added com ports, printer ports, the screen, the floppy drive, hard disk drives, everything each needed to have their own dedicated hardware interrupt request signal, or IRQ, as it's called. So as you added more stuff to your system, juggling those interrupts often posed a real challenge.

Okay. Now it's 2021, and executing an instruction that would generate an interrupt was locking up the system. So exactly as with the early UNIVAC 1103 back in 1953, the CPU, this Intel CPU I was testing, was being yanked away from executing my code. Only in this case, the CPU was never returning. So I tested the theory that an interrupt was causing the trouble by disabling the drive's hardware interrupts. It's possible to tell a drive not to generate an interrupt request when it needs attention. And sure enough, no hang. So to be sure, I reenabled the drive's interrupts, but disabled the entire system's interrupt servicing system. It turns out that's possible to do because there are times when one's code absolutely positively must not be interrupted for any reason. And again, no hang. So that told me where the trouble was.

So I dug in and discovered that the controller's BIOS was reporting that its controller interrupted on IRQ 5 when it was actually interrupting on IRQ 11. That mattered. I was waiting for the drive to signal its completion on IRQ 5, so I had placed code there ahead of time to receive and handle that interruption. But instead, by signaling its completion on IRQ 11, when I single-stepped into that instruction, control of the processor was yanked over to whatever code might have been handling IRQ 11, if any, and disaster struck.

This sort of problem never occurred in any previous versions of SpinRite because all previous SpinRites have used whatever BIOS was present, and the BIOS always knew the truth itself, even if it wouldn't tell the truth to anyone else when asked. So having obtained absolute proof that there are motherboard firmware and add-in adapter BIOSes out in the world that do not accurately report the IRQ that's being signaled by their hardware, I needed to switch to Plan B.

As of the end of this past weekend, SpinRite no longer cares at all which hardware IRQ is signaled for completion. Since IRQ 0 and 1 are permanently connected to the clock and the keyboard, SpinRite now monitors all of the remaining 13 interrupt lines at once, in parallel. And as a result, SpinRite is now working perfectly on that controller, where before it had a hard hang. And once I publish the next dev release, we'll see how many of the other similar-appearing problems have also been resolved with this change. But it does feel like we're getting close.

So anyway, a cool little bit of field problem-solving. And I had to take sort of an out-of-box solution. You could normally, on a normal OS, you could never get away with hooking all of the interrupts. First of all, in today's operating systems there's a gazillion of them. It's not like the ISA bus days. PCI has completely changed the complexion of interrupts. But more importantly, I mean, there's so much going on all at the same time that it's just not feasible to squat on all the hardware interrupts on a system.

SpinRite has the advantage of owning the entire system. Nothing else is going on. You've got timer ticks on Interrupt 0. You've got the user using the UI on Interrupt 1. And then nothing else is happening. So basically I just grab all of the remaining 13 interrupts and look for any activity on any of them, and take that to mean that the disk finished its work. And that worked. So we didn't have many problems remaining. We'll see how many are extinguished by this when I release number 6.

Leo: Nice, nice.

Steve: Yeah, very cool.

Leo: And now let's talk about Bluetooth with Steve.

Steve: Yup. So seven researchers from the University of California at San Diego have completed some very important privacy-related research which will be formally presented during the upcoming 2022 IEEE Symposium on Security and Privacy. The title of their paper provides the first hint into what they've found. It's titled "Evaluating Physical-Layer BLE Location Tracking Attacks on Mobile Devices." BLE, of course, Bluetooth Low Energy. The key is their use of the term "physical" layer, rather than "logical" or "data" or "application" layer. In other words, something about the way the Bluetooth radio is transmitting, not what the Bluetooth radio is transmitting.

So here's how they introduce their work in their paper's abstract. They said: "Mobile devices increasingly function as wireless tracking beacons. Using the Bluetooth Low Energy protocol, mobile devices such as smartphones and smartwatches continuously transmit beacons to inform passive listeners about device locations for applications such as digital contact tracing for COVID-19, and even finding lost devices. These applications use cryptographic anonymity that limit an adversary's ability to use these beacons to stalk a user. However, attackers can bypass these defenses by fingerprinting the unique physical-layer imperfections in the transmissions of specific devices.

"We empirically demonstrate that there are several key challenges that can limit an attacker's ability to find a stable physical layer identifier to uniquely identify mobile devices using Bluetooth Low Energy, including variations in the hardware design of BLE chipsets, transmission power levels, differences in thermal conditions, and limitations of inexpensive radios that can be widely deployed to capture raw physical-layer signals. We evaluated how much each of these factors limits accurate fingerprinting in a large-scale field study of hundreds of uncontrolled Bluetooth Low Energy devices, revealing that

physical-layer identification is a viable, although sometimes unreliable, way for an attacker to track mobile devices."

Okay. So what we're talking about here is exploring the feasibility of fingerprinting individual Bluetooth radios located in our consumer pockets, using subtle differences in individual device RF emissions. It turns out that this is far from the first time that some researchers have considered and looked into the possibility of doing this.

Okay. So for example, here are some titles of previous papers submitted through the years and presented during various security, communications, and engineering conferences. Back in 2006, "Detecting Rogue Devices in Bluetooth Networks Using Radio Frequency Fingerprinting." In '07, "Implications of Radio Fingerprinting on the Security of Sensor Networks." In '08 there were three papers: "Wireless Device Identification with Radiometric Signatures," "Using Spectral Fingerprints to Improve Wireless Network Security," and "Passive Steady State RF Fingerprinting: A Cognitive Technique for Scalable Deployment of Co-Channel Femtocell Underlays." In '09, "Physical-Layer Identification of RFID Devices." In 2011, "Identifying Wireless Users via Transmitter Imperfections." In 2020, last year, "Aircraft Fingerprinting Using Deep Learning."

Okay. So clearly the idea of fingerprinting a radio frequency transmitter by closely and carefully characterizing the details of its transmission not the data it's transmitting but the precise way it's transmitting is a well-established question, and a potential problem for our privacy. Since the time that researchers began looking into all of this, as we know, we've all taken to carrying individual radios in our pockets, each of which is deliberately and constantly broadcasting RF beacon signals. The designers of these technologies have gone out of their way with all sorts of fancy state-of-the-art crypto to cleverly and deeply anonymize the data that's being transmitted. But they've completely missed and skipped over the truth that inter-device differences may be sufficiently distinctive to render those devices individually identifiable. That's the question that this UC San Diego team set out to answer.

So here's at the top of their paper how they explained how they see this. They said: "The mobile devices we carry every day, such as smartphones and smartwatches, increasingly function" - and as I look at this, it looks like it's exactly what I already read in the abstract. They have a little more detail. They talk about Apple and Google smartphones, as well as Apple's intrinsic Continuity protocol, used for automated device hand-off and other proximity beacons.

"However," they say, "by their nature, BLE wireless tracking beacons have the potential to introduce significant privacy risks. For example, an adversary might stalk a user by placing BLE receivers near locations they might visit and then record the presence of the user's beacons. To address these issues, common BLE proximity applications cryptographically anonymize and periodically rotate the identity of a mobile device in their beacons." And they say: "For instance, BLE devices periodically reencrypt their MAC address, while still allowing trusted devices to determine if these addresses match the device's true MAC address. Similarly, COVID-19 contact tracing applications regularly rotate identifiers to ensure that receivers cannot link beacons from the same device over time." And of course we covered that extensively when we talked about the joint Apple/Google COVID-19 contact tracing protocol.

So they said: "While these mechanisms can foreclose the use of beacon content as a stable identifier, attackers can bypass these countermeasures by fingerprinting the device at a lower layer. Specifically, prior work has demonstrated that wireless transmitters have imperfections introduced in manufacturing that produce a unique physical-layer fingerprint for that device, for example, Carrier Frequency Offset and various carrier modulation offsets. Physical-layer fingerprints can reliably differentiate many kinds of wireless chipsets, including a recent attempt to distinguish among 10,000

WiFi chipsets." That last bit of research they are pointing to was published last year titled "Deep Learning for RF Fingerprinting: A Massive Experimental Study." And that was published in the IEEE Internet of Things Magazine.

So they continue: "To the best of our knowledge, no prior work has evaluated the practicality of such physical-layer identification attacks in a real-world environment. Indeed, prior to Bluetooth Low Energy tracking beacons, no mobile device wireless protocol transmitted frequently enough, especially when idle, to make such an attack feasible." So in other words, we've been advancing this technology. Now we've got radios which are constantly spitting things out. And in their paper, I didn't include it in the notes, I think it was, in the case of Apple, 853 beacon transmissions per minute. So a lot, you know, frequent. Thus you get a big sample of what a given radio is putting out.

They said: "Additionally, there is no existing BLE fingerprinting tool that can measure the physical-layer imperfections in BLE transmissions accurately." In other words, they had to invent something. "Prior techniques for fingerprinting either provide low-precision fingerprints because they use short-duration transient signal features, or provide high-precision fingerprints, but require long-duration signal features which exist only in protocols like WiFi, but not in BLE."

They said: "Our first contribution is a tool that uses a novel method to recover these imperfections by iteratively adding imperfections to a re-encoded clean copy of a received packet, until they match the imperfections of the received packet over the air." So in other words, in order to determine the feasibility of actually doing this, that is, fingerprinting, they first needed to develop the best technology possible to do so. And they did. And very cleverly, they realized that Bluetooth Low Energy transmissions are inherently short bursts during which not enough information is gleaned. But if you listen to a lot of those short bursts, they came up with a way of producing a high-resolution model from many short samples, short temporal samples.

They said: "Our next contribution is an evaluation of how practical it is for an attacker to track BLE-beaconing devices using their RF fingerprint. Namely, using lab-bench experiments, we identify four primary challenges to identifying BLE devices in the field. First, BLE devices have a variety of chipsets that have different hardware implementations. Second, applications can configure the BLE transmit power level, resulting in some devices having lower signal-to-noise ratio BLE transmissions. Third, the temperature range that mobile devices encounter in the field can introduce significant changes to physical-layer impairments. And fourth, the low-cost receivers that an attacker can use in the wild for RF fingerprinting are not significantly less accurate than the tools used in prior studies." In other words, they just used $150 software-defined radios. It turns out that provides all the resolution and accuracy needed.

And they said: "Our final contribution is a set of field experiments to evaluate how significantly these challenges diminish an attacker's ability to identify mobile devices in the field. We leverage the fact that BLE tracking beacons are already used on many mobile devices. We perform an uncontrolled field study where we evaluate the feasibility of tracking Bluetooth Low Energy devices when they're operating in public spaces where there are hundreds of other nearby devices.

"To the best of our knowledge, our work is the first to evaluate the feasibility of an RF fingerprinting attack in real-world scenarios. We show that even when there are hundreds of devices we encountered in the field, it is still feasible to track a specific mobile device by its physical-layer fingerprint. However, we also observe that certain devices have similar fingerprints to others, and temperature variations can change a device's metrics. Both of these issues can lead to significant misidentification rates. In summary, we find that physical-layer tracking of BLE devices is indeed feasible, but it is

only reliable under limited conditions, and for specific devices with extremely unique fingerprints, and when the target device has a relatively stable temperature."

Okay. So believe it or not, that's just the tip of iceberg. Their wonderfully detailed paper goes on for 15 pages, and for anyone interested it provides all the detail anyone could want. Even Spencer Webb, who, by the way, tweeted this link to me and put me onto the story. Thank you, Spencer. We've spoken of Spencer before. He's a radio frequency antenna designer guru. So of course this roused his interest. All of the work that they did, and the datasets, they've posted publicly on GitHub.

And I'm tempted to conclude that there's nothing for us to worry about overall. And for the most part that's the most reasonable conclusion. But one part of their paper stuck out and caught my attention. They conducted a case study of tracking a specific person. And I'll read what they wrote and then clarify the tricky bits. It's not very long. It was in their paper, "Case Study 2: Tracking a Person."

They said: "We conducted an end-to-end tracking attack executed on a controlled target, a volunteer who uses an iPhone. The attacker first carries their SDR (Software Defined Radio) sniffer close to the target device to obtain the device's physical-layer fingerprint. Simultaneously, the attacker scans for nearby BLE devices using a commonly available BLE scanner phone app, and they record the MAC address of the BLE device with the highest observed signal strength, which will be the nearest device, the target's phone. Then later, post-processing all of the data and signals collected, they use the target's MAC address to select out the target's device packets from the raw sniffer capture. They feed those packets into the BLE tracking toolkit to train its classifier with the target device's fingerprint."

Okay. "After creating the fingerprint, the attacker tracks their target by placing an SDR and laptop close to their target's home. The attacker can determine when the target is home by observing when the classifier running on the laptop indicates the packets received by the SDR match the target device's fingerprint. The attacker tracks their target for one hour in their study, during which the target walks inside and outside the house two times." And in their paper Figure 18, which I've captured in the show notes below, "shows the number of unique MAC addresses observed every 10 seconds during this hour. There are approximately 30 other devices nearby that could be confused with the target." In other words, it is a Bluetooth Low Energy packed environment.

Then in the second chart on the right, the blue bar shown in Figure 19 shows what they called the ground truth of when the person was inside the house during this hour. "The attacker's identification toolkit runs once every 10 seconds, and the red bar shows the time durations during which the tracking toolkit thinks the person was present. The bars perfectly match except for immediately prior to minute 10, where the toolkit falsely detects the presence of the target for 50 seconds, even though it had not yet actually returned."

So again, our listeners can't see, but the first chart showing the number of Bluetooth Low Energy devices seen every 10 seconds for an hour, I mean, it's a ragged-looking chart that peaks, it looks like maybe peaks at about 37. It hits a low point maybe at about 20. But in general it's like, you know, easily 20 to 30 devices that are all generating beacons at the same time. Despite that, and having never been trained on any of those, this device located outside the house, the right chart shows it's virtually perfectly aligned. That is, when the person's in the house, this thing detects the radio defects of the transmission separately from all the other devices, never having been trained on any of them.

So to me, this was significant because it suggests that, while the natural variations in RF signal generation are not generally sufficient to identify arbitrary individuals within a

large population, it may well be feasible to train up an inexpensive RF classifier to recognize a specific Bluetooth radio. And remember the protocols that are being transmitted are anonymized. The MAC address is being randomly changed. We've talked about all of that being done in the past, specifically to thwart tracking. Yet the layer under that, the layer essentially that carries the data, the analog layer, has enough variation in it that it can be seen uniquely, with sufficiently high accuracy that the same radio will later be recognized with a high probability of success.

And so you can imagine that law enforcement and counterterrorism agencies might find ample application for such passive device recognition. You don't need to install malware on it. You don't need to plant a bug on somebody. You just briefly get near enough to capture their beacon blabbing, as all Bluetooth beacons now blab. That gives you a sufficient number of samples to uniquely identify that device when it is again within radio range.

So I think this research has very usefully highlighted a weak, but potentially very significant flaw in the assumption that the only thing we need to be concerned about for protecting our anonymity and privacy is the digital data our radios transmit. It's now clear that subtle differences in the analog component of the way they transmit can be significant. And it feels as though this is the sort of thing that Apple and Qualcomm might find to be of sufficient concern that they might consider adding some deliberate noise into their radio modulation channels to thoroughly thwart this form of deanonymization attack.

Beautiful research by these guys, and something no one really, I mean, obviously people have been talking about it since '06. But we just deployed all this technology, and it's like, okay, fine. Let's make sure we scramble the data that we're sending. Well, it turns out that there are enough differences in the radio that is being used to send the data that it could be identified from a crowd. Very cool.

Leo: This kind of beaconing happens on Android, too. I'm sure, and I know they didn't try it with Android, but I bet you anything you could do it with Android.

Steve: Yeah. And in fact let me - where is the research? I've got it right here.

Leo: Basically, beaconing, as soon as they announced beaconing, I said, "This is a bad idea."

Steve: Yeah, I agree.

Leo: It doesn't benefit us as users. It's totally for advertising, marketing, maybe museums. But it's not really a user-facing technology.

Steve: Yeah. And we have no control over it; right?

Leo: Yeah, yeah. It's just you just have it, yeah.

Steve: Okay. So iPhone 10 running iOS sends out 872 advertisements per minute. They had a ThinkPad X1 Carbon running Windows. That sent out, that was number two, 864

beacons per minute. MacBook Pro 2016 running OS X, 576. Apple Watch 4 iOS, 598. Google Pixel 5 running Android, 510.

Leo: There you go, yeah.

Steve: And the Bose QuietComfort 35 headphones, 77.

Leo: You know, I'll never forget turning on, what was it, I think I was trying to pair my headphones. As I was bicycling I was trying to pair - or I have a helmet, a Bluetooth helmet - to my phone, and then bicycling down the street and watching all the Bluetooth show up, all the different Bluetooth devices show up. Bluetooth has never in any way attempted to hide itself. Bluetooth devices announce themselves. That's part of the technology. And this beaconing is even worse. The problem with the beaconing is you don't know it's doing it. Yeah, I'm not at all surprised, yeah.

Steve, another brilliant job, and not a single rant. Amazing. Thank you, sir. Actually, the listeners are very disappointed. I hope you'll have something for us next week. Steve is of course at GRC.com. That is not just the home of SpinRite, the world's finest mass storage maintenance and recovery utility, currently version 6.0. 6.1 is, as you can tell, well on its way. And you'll get a free copy if you buy 6.0 now. You also get to participate in the development of the new SpinRite.

Many free things at his website including this show, 16Kb audio, handwritten transcriptions, 64Kb audio as well. All you have to do is go to GRC.com, a great place to get the show and participate in a conversation about the show. He's got forums there. And you can leave even feedback at GRC.com/feedback. He's on Twitter, @SGgrc. So if you want to DM him, you can slide into his DMs there. Or just tweet, "What could possibly go wrong?" and see what happens, @SGgrc.

We have the show at our website, as well. Of course TWiT.tv/sn. You can get copies of the audio and the video there. If you want to watch us make the show live, we do it every Tuesday, right after MacBreak Weekly. That's sometime between 1:30 and 2:00 p.m. Pacific, which would be 4:30 to 5:00 p.m. Eastern time, 20:30 UTC. And if you want to watch or listen live, go there. You can also go to live.twit.tv, resolves to the same thing. If you're watching live, chat live in our chatroom, irc.twit.tv. Club TWiT members also have access to our Discord. You can chat there, too. And you can also download shows after the fact, if you want, not only at the website, but subscribe in your favorite podcast player, and you'll get it automatically. There's also a YouTube channel.

If you are subscribing and listening after the fact, there are also asynchronous ways to communicate besides Steve's forums at GRC.com. We have our own TWiT Discourse forum at twit.community, and we have a Mastodon instance in the Fediverse at TWiT.social. Both are free to join. We'd love to have you on any of those platforms. We love our community, and it's great to stay in touch. Steve, have a wonderful week, and I'll see you next week on Security Now!.

Steve: Will do. Bye.