SECURITY NOW!

Transcript of Episode #876

Microsoft's Patchy Patches

Description: We begin this week by answering last week's double-decryption strength puzzler. I then take a look at what's currently known about FIDO2 support in LastPass and Bitwarden. We look at last week's Mozilla announcement of Total Cookie Protection for Firefox, which doesn't appear to be working for me, and invite everyone to test their browsers. DDoS attacks have broken yet another record, another NTLM relay attack has been uncovered in Windows, Apple messed up Safari five years ago, more than a million WordPress sites were recently force-updated, and another high-severity flaw was fixed in a popular JAVA library. Then after sharing a bit of miscellany and some fun closing-the-loop feedback, we look at the awareness the rest of the security industry is sharing regarding the deteriorating quality of Microsoft's security management.

High quality (64 kbps) mp3 audio file URL: http://media.GRC.com/sn/SN-876.mp3
Quarter size (16 kbps) mp3 audio file URL: http://media.GRC.com/sn/sn-876-lq.mp3

SHOW TEASE: It's time for Security Now!. Steve Gibson is here. He's raring to go. We're going to talk about Passkeys and what the password managers LastPass and Bitwarden have to say about it. We'll also talk about Firefox's new Total Cookie Protection, which seems to be something less than total. And Microsoft, its attitude toward security is, shall we say, a little casual? It's all coming up next on Security Now!.

Leo Laporte: This is Security Now! with Steve Gibson, Episode 876, recorded Tuesday, June 21st, 2022: Microsoft's Patchy Patches.

It's time for Security Now!, the show where we protect you, your loved ones, your privacy online with this man right here, Mr. Steven "Tiberius" Gibson, the host of our show. Hello, Steve.

Steve Gibson: Hey, Leo. Great to be with you again, once again on a Tuesday. I don't think we ever miss our podcast. Someone was saying about holidays, talking about that and days off. Oh, I know what it was, it was our neighbors talking about how if a holiday occurs, then the trash pickup gets skewed by a day.

Leo: Right, yes.

Steve: And what would normally be like on a Saturday when we get our leaves picked up, that becomes Monday. And I said we once were doing the podcast on Mondays, and it used to really bug me when we'd have like a holiday. And now we're protected, we're on protected Tuesday because you might have your day on holiday as we did yesterday, it was a federal holiday, of course. It was Juneteenth. But the podcast goes on.

Leo: I do have to warn you, let's see, no, I guess Independence Day is a Monday this year. So we're all right. Remember last year. Was it last year? There was one year you hated it. We said there's no show, and you hated that. So no, no day off for you in July. The fifth of July we will be here.

Steve: Thank you. That's good. So we're on Episode 876, which I titled "Microsoft's Patchy Patches" for reasons that will be made clear. It turns out that for just sort of the alignment of the planets there were several completely unrelated stories in this past week of other security firms saying, you know, Microsoft's really not doing the job like they used to. And of course that's not news to anybody who has listened to me rant about that over and over and over. But like, okay, it's not just me. So I thought that was interesting. And there wasn't really that much else that happened this week. Lots of other sort of interesting tidbits, but nothing that wanted to grab the title away from that. So that's where we ended.

We're going to begin this week by answering last week's double decryption strength puzzler. Then I take a look at what's currently known about FIDO2's upcoming support in both the LastPass and the Bitwarden password managers, which I know pretty much covers our listener base. We look at last week's Mozilla announcement of Total Cookie Protection in Firefox and wonder about why it doesn't appear to be working, at least for me. And I'll invite everyone to test their own browsers because we have a simple way to do that.

DDoS attacks have broken yet another stunning record. We have another NTLM (NT LAN Manager) relay attack uncovered in Windows. Apple messed up Safari five years ago, which sat that way until it was just found at the beginning of the year by Google's Project Zero. Interesting story there. More than a million WordPress sites were recently force-updated to resolve a very bad problem they had. And we have another high-severity flaw which was fixed in a popular Java library. Of course Log4j was famous earlier this year. Then after sharing a bit of miscellany and a bunch of fun closing-the-loop feedback, we look at the awareness the rest of the security industry is sharing regarding the apparent deteriorating quality of Microsoft's security management. So I think another interesting podcast for our listeners and a very apropos Picture of the Week.

Leo: And of course you probably heard us talk on MacBreak Weekly about the wonderful Microsoft Defender which is being added to iOS, even though all it can do is make recommendations because Apple won't let it do anything else.

Steve: Right.

Leo: I love it.

Steve: Our Picture of the Week.

Leo: I love it.

Steve: Now, you know, in this day and age of virtual reality modeling, you see a picture like this, and you just - you can't tell if it was something someone whipped up algorithmically or if it's like an actual photo of something that exists in the physical world. It looks absolutely authentic. And in fact there's even a reflection of the person who appears to be taking the picture in the polished black headstone. You can sort of see him holding, with his left hand, holding his phone, taking this picture. But regardless, this is a headstone, or a memorial, I guess maybe, for Internet Explorer. We've got the big modernized eGlobe sort of logo, and then a little Japanese something character, and then Internet Explorer. And we get the date of birth and the date of death. 1995.8.17, so August 17, 1995 through June 15th, 2022.

And the best part of this whole thing, this beautiful stone headstone sitting on a granite slab, is sort of the - often on headstones there's a little slogan or something about who this person was or what they meant to people. This one says "He was a good tool to download other browsers." And indeed, many people launched it exactly once.

Leo: That's the only reason, yup.

Steve: To go get a copy of Firefox or go get a copy of Chrome.

Leo: I love it.

Steve: And, yes, so...

Leo: Good epitaph for Internet Explorer.

Steve: Put it to rest.

Leo: Yes.

Steve: Finally. Okay. So last week's key-strength puzzler. The question that we left our listeners with, and boy did it get a lot of response through Twitter and our Security Now! feedback, it reduces to whether a divide-and-conquer attack can succeed. Way back in 2011, the WPS, we talked about this at the time, the WPS, the so-called WiFi Protected Setup protocol, was found to be vulnerable to this style of attack. Remember that the way it was supposed to work was that a user would press a button on their WiFi access point to enable this feature. Then they would enter a preset 8-digit PIN into a device they wished to connect to their router.

And technically, since the eighth digit was a check digit, which was so dumb, like what a dumb way to waste a digit because if it didn't work you could just enter it again, you didn't need a check digit. Anyway, the eighth digit was a check digit. So only really seven digits were important because the eighth could always be calculated from the first seven. Okay. Since seven digits can have 10 million combinations, brute forcing those 10 million combinations was deemed impractical in 2011, within the timeframe that WPS would be enabled and so forth. So it was thought, okay, great. A 10-digit guess is strong enough.

But two researchers, Stefan Viehbock and Craig Heffner, who we talked about at the time, discovered a flaw in that WiFi protocol because all eight digits were not sent to the access point at once. The first four digits were sent before the second four; and, worse, the router's behavior would change if the first four were not correct. Now, how this ever got past the almighty Wi-Fi Alliance will forever be a mystery. But then again, it was only one of many mistakes that made it past the Wi-Fi Alliance through the years.

In any event, the fact that the router's behavior would change if the first four were wrong, meant that it wasn't necessary to guess all seven or eight digits at a time. It was possible to divide and conquer. It was possible to guess just the first four, of which there are only 10,000 combinations; then, having found the first half, separately brute force the final three, since the last digit can be calculated from the preceding seven. Since that's 1,000 maximum for the second three, we have a grand total maximum of only 11,000 possible guesses, reduced from a previously believed 10 million.

Okay. So Erik Osterholm's puzzler from last week amounts to the same question. A ciphertext is encrypted with an effective 512-bit key length by first encrypting the original plaintext with a 256-bit key, the first half of the whole 512-bit key; then by encrypting it again with another 256-bit key, that is, the second half of the 512-bit key. And of course he was asking, why is that not only twice as strong, rather than exponentially stronger?

So if we can brute force decrypt that second encryption in X-time, then brute force decrypt the first encryption also in X-time, what Erik asked is why isn't this strength just 2X rather than X times X? And the correct answer, which all of our many listeners who wrote in answered correctly, amounts to whether it's possible to perform the same sort of divide-and-conquer attack as was possible, which broke the WPS setup protocol. Remember that the weakness that was exploited by the WPS attack was that there was some affirmative feedback after the first half of the guess was made about whether or not that first half guess was correct. And that's what's missing from Erik's double-encryption thought experiment.

Here's how I would phrase it formally: The result of any encryption by a high-quality cipher such as AES's Rijndael is indistinguishable from entropy. Therefore, the result of the first encryption will be indistinguishable from entropy. So when following Erik's suggestion and question, and performing the first decryption, how can the attacker, who is using brute force key guessing, know when their decrypted guess is correct, when both a correct guess and all other incorrect guesses appear equally random? In other words, it is not possible to divide and conquer. The only way to decrypt the double-encrypted plaintext would be to make a first guess at the outer key. Then the attacker would need to try all possible inner keys, that is to say, all 2^256 of them, to see whether any of them worked. Assuming that none did, all of that work would then be discarded; the next outer key would be chosen; and again, all 2^256 possible inner keys would need to be tried.

In other words, only when both the first 256-bit key and the second 256-bit key were simultaneously correctly applied would the correct plaintext be restored. So this is, indeed, 2^256 times 2^256, which is 2^512 maximum possible brute force guesses needed. And thus, Erik, the answer to your question, and the answer to the teaser that I got a lot of great feedback about. And a lot of people enjoyed the idea of us having that fun.

Okay. I just turned the AC up. It was getting a little warm in here. So third-party authenticators. In the aftermath of the Apple's, Google's, and Microsoft's announcements of their forthcoming support for FIDO2 and passkeys authentication, we've been talking about what all this means. And I believe that we've settled into exactly the right understanding. It's relatively quick and easy for those three major publishers to add this support to their clients, as they've all announced they're going to. And when they've done so, everyone will be just one software update away from having that client-side technology in their hands. But it's a bit like creating the first shortwave radio. There's no one else to talk to yet. So the existence of all of those clients won't be very useful initially. The heavy lift will be getting the millions of individual web servers updated to support the WebAuthn standard at their end, since any use of Apple's, Google's and Microsoft's clients will require that, too.

And I believe that we've also identified that the biggest usability hurdle for the practical use of FIDO2's private passkeys is the need for their dynamic synchronization. And now that the world - it's been interesting to watch. Now that the world has sobered up after the intoxicating passkeys announcement parties, others are realizing what we immediately saw as a problem. A story in Fast Company is titled: "There's a big problem with Apple and Google's plans to nix passwords." And 9to5Mac's headline read: "A world without passwords could further lock users into Apple and Google ecosystems." Yeah. Like we've been saying.

Those stories note that: "FIDO's current proposal has no mechanism for bulk-transferring passkeys between ecosystems. If you want to switch from an Android phone to an iPhone, or vice versa, you won't be able to easily move all your passkeys over." And they didn't mention Windows, but we know the same problem will exist there. "We don't really have a batch export method right now," says FIDO Alliance executive director Andrew Shikiar. He said: "I think that's probably a future iteration."

Wow. So those FIDO guys were really not thinking through the usability angle of all this. You know, saying, "We'd like you all to adopt this half-baked solution today, and we'll worry about exporting your locked-in keys later." The reports that have been published also explain: "The fear is that, if users can easily move all their passkeys between providers, hackers may try to exploit this capability. For now, it's unclear when or how FIDO might address that problem."

And then they quote the president, the president of the FIDO Alliance. Sam Srinivas, Google's product management director for secure authentication, who's also the president of the FIDO Alliance, says: "It's very hard to do it safely from the get-go because, if we give a mechanism without great care for someone to export all these keys, you know who's going to show up first for that."

Leo: That's a good point. That's actually a really good point I hadn't thought about. Because it would have to export that in the clear; right?

Steve: It's got to do it, well, I mean...

Leo: No.

Steve: No.

Leo: Because your input's going to be a secondary FIDO2 server, so...

Steve: Right. I mean, it's clear there are ways this could be done. But again, because it was like the FIDO concept was never meant to scale this way. It was scaled by force because it didn't go as FIDO1. It just didn't - it never got off the ground. So they did one without really thinking it through. Anyway, so in other words, we're going to be quite happy, they're saying, with lock-in. And we're going to tell users that it's too dangerous to allow them to move their keys around themselves.

Leo: That's a huge problem.

Steve: Of course it is. As a cross-vendor user myself, I need Apple and Windows to sync. And I don't see that happening without either a third-party synchronization vendor, which is a thing that could exist, or third-party FIDO2 passkeys being supported by a password manager. Which brings us to two password managers which I've been looking into and want to briefly discuss: LastPass and Bitwarden.

As everyone knows, LastPass was previously a many-years sponsor of this podcast and of the TWiT Network. And Bitwarden is currently a sponsor and offers a compelling array of solutions. So the question I had was where do those two fit within this new and evolving era? As I believe I mentioned last week, I was annoyed with LastPass because their most recent blog posting, from Monday before last, which I was hoping would provide some clarification, left me feeling more confused than I was before. Everything they say feels sort of coy and blurry. Nothing they say just tells us what is going on. Here's an example direct quote from an announcement of an upcoming webinar that they'll be hosting two days from now, this coming Thursday.

They wrote: "It's time to envision a world without passwords, a world that removes the password-related friction that prevents users from securing and managing their passwords easily and automatically. True FIDO2-compliant passwordless access to every device, browser, website, and app will take years to develop" - okay, right - "but LastPass can get you there sooner." What?

Leo: Huh?

Steve: Yeah, right. Okay. How? "Join us to learn how LastPass is enabling an end-to-end passwordless experience for the LastPass vault and all sites stored within." Okay. "What will this enable you to do? Reduce password-related friction for employees, increase usage and adoption, set stronger policies and increase security, fewer lockouts for employees and resets for IT. Hear from LastPass CTO Christofer Hoff as he demonstrates a passwordless login experience, and discusses future plans for FIDO2 authenticators like biometrics and security keys." Okay, well, first of all, FIDO2 authenticators.

Leo: It's not passkeys; right?

Steve: It's like it's just - it's all a big blur. So as I said, this is like their recent blog posting which doesn't actually say anything. And you could - I guess you could sort of forgive them here because they're teasing their webinar in two days. But the blog posting was the same. It was like, it made up new terms and used them in weird ways. Like what does an authenticator have to do with biometrics? Those are two different things. But they used them together. So anyway, for what it's worth, I did want to let our listeners know that there will be a webinar in two days. I've got the link to it in the show notes. I made it this week's shortcut, so grc.sc/876 will bounce you over to a signup page. I'll be watching to see what we learn from Christofer. We know Christofer. You and I have met him, Leo. We were onstage together a couple years ago, just before COVID.

Leo: Oh, yeah. No, that's not the guy we were on with.

Steve: Oh, it isn't.

Leo: No.

Steve: Oh, okay.

Leo: I think this is a new guy. They've gone through some ownership changes.

Steve: Well, yeah. They're owned now by an equity...

Leo: No. No, they were.

Steve: Oh.

Leo: And now they're spun off.

Steve: From the private equity firm?

Leo: Yeah.

Steve: That didn't take long.

Leo: No. So it's unclear, it's really unclear what's going on with them. And the people that we know are gone, pretty much.

Steve: Okay. Well, we know that Joe has long since gone.

Leo: Yeah, Joe Siegrist, the creator, is gone. His niece was there. I think she's gone. And then we knew, we were with the CISO, I think, or the CTO, on that panel. But I think he left, as well.

Steve: In Boston, yeah.

Leo: Yeah, I think he left, as well. So I don't know who these people are. But we should watch.

Steve: So I wanted to watch because I know that just for inertia's sake a lot of our listeners are still there. The good news is Bitwarden is a member of the FIDO Alliance.

Leo: Well, that's a good sign.

Steve: Yes, it is. And so of course Bitwarden is our current password manager sponsor. So I think the problem that any third-party logon system reasonably has had is the chicken-and-egg problem. Which makes it difficult for them to invest in any system which cannot actually be used until it's supported by the world's servers. So the flipside of that is that the clear and obvious need for cross-vendor passkey synchronization which it's more and more clear every week, it's now very clear - FIDO and Google both just throw up their hands saying, "Yeah, that's a problem." And that creates the biggest need and push for third-party passkey managers that there's ever been.

So I wanted to understand where Bitwarden stood. I did some digging and found some dialogue in their community forum under the title "Bitwarden Passkey (How does Bitwarden fit into the new Microsoft/Google/Apple Passkey initiative?)" So the person posting this wrote the question: "Microsoft, Google, and Apple have announced support for the FIDO2 passwordless initiative that media are calling 'Passkeys.' Because Passkeys creates a new key pair for each website login, there is the issue of moving all these key pairs among devices." He says: "I'm sure that Google will do that for Android and Chrome, and Apple will do it for their iPhones and Macs. But what about between Android and Apple or Linux?" And not to mention Windows.

"Would," he asks, "Bitwarden be able to support the new Passkeys cross-platform, like it does with current passwords?" He says: "I want to sync Android to Linux desktop, and I will wait for Bitwarden to support this, if the feature will be added." And that's how I think a lot of us feel. You do not want to get stuck with non-exportable passkeys.

Leo: Could a third party like Bitwarden do passkeys and become, I mean, it's still not exportable. So if you decided you didn't like Bitwarden, you'd be stuck there. But at least it would let you use an Android phone or an iPhone or Windows or Mac.

Steve: Yeah, yeah.

Leo: I mean, that would be a big advantage. So could they do this? They'd keep the database? They'd have to have some sort of biometric, ideally some sort of biometric login; right? Or you could use a YubiKey. I mean, I can use a YubiKey. I do in fact use a YubiKey with my Bitwarden.

Steve: Well, as we know, it's now possible for any apps on those platforms like Android and iOS to leverage the built-in biometrics on the device.

Leo: True. So in fact they do that. When I open Bitwarden on those devices, it does a face recognition, and we're in. On my computers when I set up a new account I have to use the YubiKey the first time on a new system.

Steve: Right. And so for security they might want to enforce the use of some affirmative device in order to protect.

Leo: Right.

Steve: But there is - but they could synchronize passkeys in the cloud exactly as they synchronize usernames and passwords right now. And so I think that's going to be the solution. I don't think Apple is going to address this. I don't think Google are going to address it. They're both saying, I mean, it turns out the president of FIDO is the Google guy. And he's saying, oh, I know.

Leo: I thought and I guess I was wrong, I was told that Apple had said there is a way to get these out. But maybe...

Steve: A key at a time.

Leo: Oh, one key at a time. Oh, well, that's...

Steve: Yes, in order to share a passkey. And so the problem is when you authenticate to a new site, you want all of your ecosystem to be brought up to speed so that you can then go somewhere else, like to a different computer, and log into that site. You don't want to have to manually send that passkey to each, like, cross ecosystem into the other world. Or those two are never going to be synchronized. And that's why I've been using the term "dynamic passkey synchronization." It needs to be done for you on the fly. And that is exactly what Bitwarden supporting this passkeys, FIDO2-style passkeys, would mean. Anyway, the answer...

Leo: I think maybe, and this is a complete conspiracy theory, this will open the idea to maybe there's a better way. And maybe somebody's just going to come across SQRL and say, actually there is a better way, and let's just do this. Because it is better in every respect.

Steve: Yes. It would be good. It would be good if that happened. So the answer in the forum is: "Bitwarden does currently support FIDO2 WebAuthn for multifactor authentication in addition to your master password for vault unlocking." In other words, when you use - and this has been in there for a while. And again, they are already a member of the FIDO Alliance. So they're actually being a WebAuthn server to accept a FIDO2 client's authentication as a very strong factor when you log in to unlock your Bitwarden vault. And I have a picture in the show notes, but there is a Bitwarden blob, says: "Two-step Login via FIDO2 WebAuthn | Bitwarden Help & Support." So that's been in Bitwarden for some time.

And so this guy finishes: "Bitwarden does not support using these passkeys to log in in lieu of the password manager yet, but there is a current similar feature request for this to be supported." And you've got to know that the wizards at Bitwarden understand they've got an opportunity here to get going on this.

Leo: This is one advantage you have as an open source project. Somebody could issue a pull request and implement it. I mean, if the community wants to support it, they can add it. I mean, it sounds like in addition to your master password for vault unlocking, that's just for - basically that's one password. It supports WebAuthn for one password, your master password.

Steve: Exactly. Exactly.

Leo: Yeah, yeah. Which is nice. In fact, you could use a FIDO2, there are FIDO2 YubiKeys you could use for that purpose.

Steve: Right, right. And so what this does mean is that somewhere in Bitwarden there are already people who are fully FIDO-aware. And what we need is for them to reverse roles. Right now they're being a WebAuthn provider for a FIDO2 authenticator. We need them to become a FIDO2 authenticator talking to WebAuthn providers at websites.

Leo: That would be awesome, yeah.

Steve: And that doesn't seem like that big a reach to me. So anyway, the roadmap page has a great deal of discussion of this. So I'm sure it's something that they're aware of now. And again, you really can't fault any password manager for not doing it preemptively. I mean, I did. But I just did it as a proof of concept to demonstrate this is the way we can solve this problem. And I knew that only two sites or three or four in the world were going to be able to use SQRL. But my hope was that by showing how it could be done, that would get the world going.

And it may be that showing that there's a better solution, as you said, Leo, may still get the world to say, hey, you know, why don't we just do this? And it's not that big a reach because the WebAuthn protocol optionally supports the crypto that is key to SQRL's operation, that is, it allows it to use deterministic keys rather than keys that are completely random. So anyway, we'll see. And wow. We're at 36 minutes in. Let's take our first break.

Leo: Holy cow. How does that happen? It's amazing. Thank goodness we have advertisers so we can pause and give you a pause that refreshes.

Steve: Yeah, to rehydrate.

Leo: Yes. You're going to get that thing that Alex Lindsay was talking about?

Steve: I did go do some shopping. Except that...

Leo: A little tempting, eh?

Steve: Except that Lorrie and I both use SodaStreams that carbonate, and it looks like it doesn't like to have a carbonated beverage.

Leo: Oh, no, I'm sure it doesn't. So you're drinking fizzy water when you drink water.

Steve: I am. Yup. In fact...

Leo: You prefer that.

Steve: I just cracked the seal, and it went "pshhhhhhh."

Leo: Yeah. You prefer that.

Steve: Yeah, I like the taste. I like the taste a lot.

Leo: I guess it's slightly acidic; right? So...

Steve: It's slightly acidic.

Leo: Remember I bought the tank, I was going to do that whole SodaStream thing.

Steve: Yeah.

Leo: Bought the tank, and we couldn't find anybody to fill it. So I basically pushed it - I palmed it off on Mikah. The tank is still under his desk. I said, "Here's the tank. Here's the nozzle. You figure it out." So he's going to find - because he said, well, let's share. That was his mistake. He said, "Let's share. We'll leave the tank at work, and then we can just bring in our bottles."

Steve: Yeah.

Leo: So that's a good idea if we can just find somebody to fill it.

Steve: So for me it's a home brewer. We have a...

Leo: Yeah, we have plenty of those, but they don't - they want to use their bottles.

Steve: Oh.

Leo: Which would be fine, I guess.

Steve: So you just...

Leo: It has to be one with a siphon in it.

Steve: It's got to have the siphon tube because it needs to be - you don't want the gas off the top.

Leo: Get the liquid. Yeah, yeah.

Steve: You want the actual liquid off the bottom.

Leo: Yeah, we'll have to figure it out. So anyway...

Steve: So you want a non-cranky home brewer.

Leo: Yes.

Steve: Is what you want.

Leo: One who doesn't mind refilling my bottle.

Steve: Yeah, who's had some of his own brew and is relaxed about it now.

Leo: Yeah, that's it.

Steve: Yeah.

Leo: We're surrounded by breweries. Everybody here makes wine and beer. I mean, that's what you do in this town.

Steve: Yeah, you've got grapes, yeah.

Leo: But we couldn't find anybody. But we'll keep looking. We'll find it. Anyway.

Steve: Okay. So last Tuesday, Mozilla's headline read: "Firefox rolls out Total Cookie Protection by default to all users worldwide."

Leo: Woohoo!

Steve: I know. And the big word that's so easily missed, as we know, is "default." That's what's changed. Until last Tuesday, "sequestered third-party cookies" were optionally available. I had them enabled in my Firefox instances, or I guess I should say disabled, which I had to do manually, as I imagine many of this podcast's Firefox users also did after we talked about the option quite a while ago. But until now it's been an option.

Which means, of course, that the majority of Firefox users would not have had this enabled since it wasn't the browser's default setting. Now it is.

What's most shocking to me is that it took us this long to get here because it is such an easy place to get to. This change does not disable third-party cookies. That's the secret. It merely divides the single massive global cookie jar into individual per-domain or, as web engineers would say, same-origin cookie jars. In that manner, any third party is welcome to set a cookie in anyone's browser. But when that user goes somewhere else, the cookie jar will be switched to a new jar for that new domain.

And again, any third party will be welcome to set their cookie into that jar. But what they will not be able to do is to see the cookie that they had previously set into the same user's same browser when they were visiting that previous domain. And that simple measure kills cookies whose primary purpose had been cross-domain tracking. You just do per-origin cookie jars. Again, such a simple measure. The idea that it took this long for it to happen is to me astonishing, but it finally happened.

Leo: This is the number, this is the problem with cookies; right? I mean, all of these cookie banners and all this missed the point.

Steve: Yes.

Leo: Cookies are fine. They're necessary. It's third-party cookies that are the problem.

Steve: Actually, it's third-party cookies that cross domains.

Leo: That could be read on a first-party site.

Steve: Yes.

Leo: It's the Facebook Like button which gives Facebook a view into that site and who's visiting it.

Steve: Right.

Leo: And even in the original Mozilla or Netscape specification for cookies, they said only the site that created the cookie can read it. But they didn't anticipate this loophole that people would embed little bits of other people's sites on their web.

Steve: Exactly. Exactly. And so this is, I mean, this is such an easy change. So, okay. Now, all that said, it's not working for me under Firefox v101.0.1.

Leo: Oh, no.

Steve: Which appears to be the latest.

Leo: Oh, no.

Steve: Chrome is wonderful as I have it set currently. And I'm sure I went in and tweaked something. But Firefox under its so-called Standard Privacy & Security / Enhanced Tracking Protection is doing nothing. I set it to Strict, and still nothing. I set it to Custom. And then I had to tell it to block - I told it to block cross-site tracking cookies. It still wasn't. It was necessary for me to turn off all third-party cookies in order to get cross-domain third-party cookie blocking to work.

Now the question is, how do I know? A piece of technology I spent a great deal of time developing many years ago, in fact I think it was in '08. The pages have dates on them. So, I mean, this is - I've been focused on this third-party cookie problem for a long time. Any, it's GRC's Cookie Forensics. If you google "GRC cookie forensics," it's the first link that comes up because it's been there since the dawn of the Internet. And if you click it, it does an instant test of your browser's current cookie handling. And we'll show you green for good and red for bad, and like blank if there's no cookie transaction going on. My Firefox, unless I turn off, as I said, all third-party cookies, it's not blocking third-party cookies.

I actually maintain a separate domain, GRCtech.com, just for this purpose. I created a third-party so that I could experiment and then automate that testing in order to show people what their browser was doing. So anyway, I just - I wanted to say that, great news, that Firefox says they're doing this. But it doesn't seem to be working. And so anyway, GRC Cookie Forensics, for anyone who is interested. And hopefully they'll get it working at some point. And they did say they're rolling it out. So maybe that's what's going on is that I've not been rolled on. It hasn't been rolled out to me yet, or maybe they're tiptoeing. I don't know.

But boy, Chrome looks great. The way I've got it set up it just comes back completely happy. And in fact I have a different page, it's GRC.com/cookies/stats.htm. And that shows a series of bar graphs. Again, you can see it's been a long time since I've been there since I have IEv5 and IEv6 and IEv7 that I'm tracking, and a bunch of others. But I also have Chrome. And boy, it used to be - the bars used to be all the way at the top. And these are GRC's visitors. GRC's visitors all have Chrome, their Chrome, blocking third-party cookies. Maybe that's the default now in Chrome, and I once talked about that, and I've forgotten it. I don't know.

Yup, so now you're showing the cookie forensics on the site. And see all those red, if you scroll down into that second group of red, that's all bad. Those are third-party session and persistent cookies that were just - you can see it says oldest cookie was one second old. So they were just sent.

Leo: Is this my browser?

Steve: Yes.

Leo: Oh, funny. I'm on Firefox.

Steve: Yeah, I know. And it's not good.

Leo: Oh. Hmm.

Steve: Uh-huh. And if you were to go under - if you go in the hamburger menu to Settings and, let's see, what is it, Settings, Privacy & Security, and then Enhanced - then you get there, and then, yeah, Privacy & Security.

Leo: Oh, see, I'm on Standard. Usually I'll run under Custom.

Steve: Okay. Now do that. Now, and normally it tells you you need to refresh your page. It didn't tell you that you have to do that.

Leo: So all third-party cookies, they don't want to block those. And I know why. Sometimes third-party cookies are from image servers that are really first-party, but they're on a different URL or different IP address. So which would I want on this?

Steve: Well, try just instead of doing Custom, try doing - so first of all...

Leo: Strict should do it; right? Yeah, let's see.

Steve: Strict should do it. Oh, and it does say refresh your tabs, yeah.

Leo: Reload tabs, yeah.

Steve: Okay. Now go back to GRC.

Leo: So social media trackers, cross-site cookies in all windows. Okay, good. All right.

Steve: Yeah. Exactly. Should work.

Leo: Now let's go back and refresh.

Steve: Now if you scroll down there is a button that I have, or you can do that, yup.

Leo: Uh-oh.

Steve: Still bad.

Leo: Worse.

Steve: Did not - I know. It did not...

Leo: There's one more red dot than there was last time.

Steve: Yeah, the icons, exactly.

Leo: Yeah.

Steve: And so now go back over, and if you go to Strict...

Leo: I'll go back to - I'll make it even stronger. Custom is more than Strict; right? If I do Custom...

Steve: Yes.

Leo: Then I can say let's block a lot of crap here. So what should I block?

Steve: I had to turn them all off. And see...

Leo: All cookies? No.

Steve: I had to.

Leo: Well...

Steve: No, no, no, third-party.

Leo: Third-party cookies. All right.

Steve: But that's extreme. And the point is that should no longer be necessary.

Leo: Yeah. Let me refresh. Icons are going to always be a problem, I guess. Icon, icon. And now these are empty. What does that mean? No third-party cookies.

Steve: Yes, that means no third-party session cookies were received from your browser.

Leo: Okay. So it's not green, it's empty. That means there was no cookie at all.

Steve: Correct.

Leo: And green would be okay; right? First-party.

Steve: And orange, the orange meant that an older cookie was received, but that's not that big a problem.

Leo: Right.

Steve: But it was necessary for you to go all the way to locking all third-party, not the default that they say they're going to be supporting, and not even only blocking tracking cookies. But as you said, Mozilla was maintaining a list of sites that they were blocking which I thought was unfortunate. But they're still not doing the right thing.

Leo: Generally at home, you know, I think I turned this to Standard only because of the show because generally at home I run Custom. But I don't turn - I just do cross-site. I do, like, this setting. But okay. So now cross-site tracking cookies blocked, cross-site tracking cookies and isolate other cross-site cookies.

Steve: Yeah. You would think that would be enough.

Leo: That should be enough.

Steve: Yes

Leo: But it's not.

Steve: No. So refresh the page.

Leo: Okay. All right. Let's see what we've got here. [Buzzer sound].

Steve: Yeah.

Leo: [Buzzer sounds]. Okay. I guess I'm going to - because this is the problem is that I think everybody's scared off when they do this because it says this is going to break stuff. This may cause websites to break. And as I my experience is, it does sometimes.

Steve: Well, yeah. Yes. And the other problem is that when Mozilla announces this, we just assume, oh, good, it's all good now. Well, you know, as they say...

Leo: No, it's not by default.

Steve: Trust but verify.

Leo: So, and GRC.com/cookies. That's very valuable.

Steve: Yeah.

Leo: Will that get me there, or do I have to type in "forensics," as well?

Steve: No, I think GRC.com/cookies probably takes you - I think the first page has a cookie monster saying "Delete cookies?"

Leo: Delete cookies, grrr. And then this is good because it's an explainer, which is a good thing.

Steve: Yes. You know me. Back in those days I was doing a lot of explaining.

Leo: Yeah, yeah. Now we keep you busy with the show, you don't get to.

Steve: That's right. That and SpinRite.

Leo: And then this is it, the bottom link is the web cookie operations.

Steve: Actually it's the third link there in that block of links.

Leo: Cookie forensics, there it is.

Steve: Cookie forensics, yeah.

Leo: Yeah, nice. This is a great tool. Thank you for - I knew about it, but I forgot. So thanks for reminding me, yeah. Very nice.

Steve: So we will see if they're going to roll it out and eventually get it right. And the cookie forensics page lets us find out real quickly.

Okay. So DDoS in the news. We keep breaking DDoS attack records. We're pretty much at the point where our eyes just glaze over now at the size of these attacks, gigawatt bits per second and all that. It's like, whoa, okay. My wires melted. I guess I should have cut the wire where it says "Engage firewall."

So once again, Cloudflare has reported that last week it stopped and mitigated the largest HTTPS DDoS attack on record. That attack weighed in at 26 million requests per second and was aimed at a website of a protected Cloudflare customer. So attacks of this size no longer originate from individual compromised hardware devices because they just don't have the speed or connectivity. Instead, most of the attacking IPs, it turns out, were owned - and this was in Cloudflare's analysis - by other cloud service providers whose virtual machines and their powerful servers had been hijacked to generate the attack. Since HTTPS query attacks cannot be spoofed, Cloudflare was able to trace the attack back to a powerful botnet of 5,067 - that's not approximately, that's exactly - 5,067 IPs, each of which generated approximately 5,200 requests per second at peak.

A Cloudflare spokesperson said that to put the size of this attack in perspective, they had been tracking another much larger in agent count, but less powerful in query rate botnet consisting of over 730,000 devices. Now, rather than letting that number just wash over us, let's stop and think about that for a second. A single identified botnet consisting of more than 730,000 compromised devices. Nearly three quarters of a million "somethings," all participating in coordinated attacks.

Now, however, the devices are apparently things like light switches and $5 outlet plugs, since that botnet, while large in number, generated fewer than one million requests per second overall, thus roughly 1.3 requests per second on average per device. 1.3 requests per second per device pretty much classifies the device as a $5 light switch or plug, but individually they're still able to generate some Internet traffic. The ones I have at home all phone home to China. Since they live behind a SoHo NAT router actually two series-connected NAT routers it's exceedingly unlikely that anyone got into them from the outside. It's far more likely that, if they have been up to some mischief lately, they've been sold into slavery by their original producer.

I'll say again that no one should have IoT gadgets attached to their primary home network. It takes deliberate work to set up and maintain a secondary IoT network, but I cannot think of anything more important for residential security. The entity they are phoning home to may not be friendly. And how would you ever know? Now, the good news is that most recent IoT routers, WiFi routers, support one or more isolated guest networks. Thank goodness. That's what you want to use for those unknowable IoT widgets. This makes the establishment of a secure perimeter far more easy and stable.

Anyway, as for Cloudflare's latest finding, that recent attack was, on average, 4,000 times stronger due to its use of virtual machines and powerful infrastructure servers. Cloudflare also notes that HTTPS DDoS attacks are more costly to produce than others because these days they require more computational resources, needing as they do to bring up a secure TLS encrypted connection for every attacking query. The bottom line is, DDoS attacks are no longer survivable unless the target is isolated behind an attack-mitigating bandwidth provider. If you're not, and your organization is being attacked, just declare a holiday and send everyone home until the attack has passed. That's another reality of today's Internet.

Okay. As for another reality of today's Internet, we have MS-DFSNM. As we've all learned, complexity is the sworn enemy of security. In any complex environment consisting of complex interacting components, the addition of another component requires an understanding of that new component's potential interaction with all other existing components. In the same way that adding each bit to a key doubles that key's total complexity, adding components to a system creates exponential complexity growth. And this is one of the biggest dilemmas which we keep seeing that Microsoft has stumbled into. Now we have a newly uncovered type of Windows NTLM (NT LAN Manager) relay attack which has been named DFSCoerce. DFSCoerce leverages the Distributed File System (DFS) Namespace Management Protocol (MS-DFSNMP), and that allows the attacker to seize control of a domain.

When we hear the phrase "a new form of NTLM relay attack" at this point our eyes roll because you need to wonder just how many different forms of NTLM relay attacks there can be if new forms are still being discovered and uncovered in the year 2022. NTLM relay attacks are a well-known method that exploits Microsoft's original half-baked and never sufficiently robust challenge-response mechanism. We've talked about this so many times before. Rather than simply discarding it decades ago as being too broken to fix, they have kept this sickly patient on life support by continually attempting to patch it and wrap it in additional layers of gauze. As a result, today's NTLM relay attacks work the same way today as they did back then. A malicious party sits between clients and servers, intercepts and relays validated authentication requests to gain unauthorized access to network resources, effectively allowing it to gain an initial foothold in Active Directory environments.

Filip Dragovic, who has been a prolific discoverer of problems, we've mentioned him before, also discovered this latest wrinkle in the rich NTLM attack surface. He tweeted: "Spooler service disabled, RPC filters installed to prevent PetitPotam, and File Server VSS Agent Service not installed, but you still want to relay." And he says "Domain Controller authentication to Active Directory Certificate Services? Don't worry. MS-DFSNM has your back." Meaning yes, there's a way you can still do that.

What is MS-DFSNM? Well, it appears to be another of those things that some random Microsoft engineer added in one of those "Oh, here's what we need to add to get that done. Won't it be neat?" One of those fits of protocol design 15 years ago back in 2007. That's when it first appeared. It went into Windows, and now it can never be removed. For anyone who's curious, it provides yet another remote procedure call interface for administering distributed file system configurations.

To give everyone a taste for this, here's what Microsoft's first paragraph of their overview of this protocol explains. Microsoft says: "The DFS Namespace Management Protocol (DFSNMP) is one of a collection of protocols that" - now listen to this. It's one of a collection of protocols. Not the only one. This is a part of a collection - "that group shares that are located on different servers by combining various storage media into a single logical namespace. The DFS namespace is a virtual view of the share. When a user views the namespace, the directories and files in it appear to reside on a single share. Users can navigate the namespace without needing to know the server names or shares hosting the data. DFS also provides redundancy of namespace service."

Yeah. And DFSNMP is one of a collection of protocols for doing that. So okay, sure. That sounds neat. And I guess 15 years ago, when there was nothing better to do, then okay, let's add that. So this is a perfect example of what appears to be complexity for its own sake, and nothing could be more antithetical to security. Fifteen years ago the danger of this should have been appreciated, but apparently it wasn't. In any event, we're stuck with it now.

The discovery of this particular DFSCoerce attack follows the related PetitPotam attack which is an abuse of Microsoft's Encrypting File System Remote Protocol (MS-EFSRPC) to coerce Windows servers, including domain controllers, into authenticating with a relay under an attacker's control, letting threat actors potentially take over an entire domain. Sound familiar? Yeah. Same exact attack using an entirely different protocol. But don't worry, there's lots more of those protocols where those came from.

The CERT Coordination Center noted in detailing this attack: "By relaying an NTLM authentication request from a domain controller to a Certificate Authority Web Enrollment or Certificate Enrollment Web Service on an Active Directory Certificate Server System, an attacker can obtain a certificate that can be used to obtain a Ticket Granting Ticket (TGT) from the domain controller." And yes, if that makes your head spin, it probably should.

So here's my advice: If any of our listeners are offered a job, given responsibility for managing and securing any significant Windows enterprise installation, first, you should start off being single because you will wind up being single. And you should be sure to get a lot of money because you're going to be trading your life and your sanity for that thankless job.

To mitigate NTLM relay attacks, Microsoft recommends enabling protections like Extended Protection for Authentication (EPA), SMB signing, and turning off HTTP on Active Directory servers. Again, those are all mitigations, not cures or solutions. Just wrap it up in some more gauze. Wow.

Although I didn't set out with this goal, this did wind up being a pile-on-Microsoft episode. It's not that it's not deserved, but it does feel somewhat redundant. So I'm happy to be able to report that Apple screwed something up, too.

Leo: Oh, what a relief.

Steve: In a somewhat predictable way.

Leo: And WordPress, I'm sure, at some point. But okay, keep listening, yeah.

Steve: Yes. We're about to get to them, as a matter of fact.

Leo: Okay, good.

Steve: Yes. Google's Project Zero discovered that a security flaw in Apple's Safari was found being exploited in the wild earlier this year. What was interesting about this particular flaw was that it was originally fixed back in 2013, then inadvertently reintroduced in December of 2016, and only just fixed last month. The issue, tracked now today as CVE-2022-22620 and bearing a hefty CVSS of 8.8, is another use-after-free vulnerability in WebKit that was being exploited by a piece of specially crafted web content to give its exploiter arbitrary code execution on the machine. That's never good.

Early in February of 2022, Apple shipped patches for the bug across Safari, iOS, iPadOS, and macOS, while acknowledging that "it may have been actively exploited." Uh huh. And the sun may rise in the morning. Google's Project Zero's Maddie Stone wrote: "In this case, the variant was completely patched when the vulnerability was initially reported in 2013. However, the variant was reintroduced three years later during large refactoring efforts. The vulnerability then continued to exist for five years until it was finally fixed when it was discovered as an in-the-wild" - meaning yes, actually being exploited - "zero-day in January of this year," 2022.

Maddie explained that both the October 2016 and the December 2016 commits, one of which reintroduced the original bug from 2013, were very large. The commit in October changed 40, four zero, files with 900 additions and 1,225 deletions. The commit in December changed 95 files with 1,336 additions and 1,325 deletions. It seems untenable for any developers or reviewers, that is, code reviewers, to understand the security implications of each change in those commits in sufficient detail, especially since they're related to long-lived semantics.

So whatever it was that was happening at the end of 2016, it was apparently a major revamp of some core Safari system. And it appears that you get bit either way. On the one hand, if you leave crappy old code alone under the theory of "if it's not broken don't fix it," you wind up eventually with even older crappy old code. Right?

Leo: Yup.

Steve: But if you bite the bullet to make huge sweeping revamping code-modernizing changes, you get bit by the introduction of brand new bugs which might be the same as some very much older bugs that were previously found and fixed. Now, given a choice, I think I would do what Apple apparently did. Code, as we know, really doesn't evolve very well. If enough time passes, the assumptions that were once baked into the original code base no longer hold true, and they really can begin to chafe, and things can begin to crumble. If the problem code was rewritten to solve the problems of a strong new future, it can be best to scrap a large previous investment, no matter how solid it now is, and just start over with all the benefit of the knowledge acquired through the intervening years. That seems to be what Apple did. Okay, so not perfectly. But this got fixed. So, yeah, they did mess up, but I would say probably for the right reasons.

And Leo, speaking of WordPress, one million WordPress...

Leo: I confess I looked ahead. I had a feeling. Go ahead.

Steve: One million WordPress sites were just force-updated. The WordPress security guys at WordFence identified an exploited in-the-wild zero-day bug in a widely used, more than one million installations, WordPress plug-in called Ninja Forms, which is a customizable contact form builder. The severity of the bug earned it a CVSS of, yes, 9.8; and it affected many versions of Ninja Forms starting with v3.0. WordFence explained that the bug made it possible for unauthenticated attackers to call a limited number of methods in various Ninja Forms classes, including a method that deserialized user-supplied content. And that resulted in an Object Injection which could allow attackers to execute arbitrary code or delete arbitrary files. That's never good.

Although the update was supposed to be automatic and forced, any Ninja Forms users listening to this are advised to definitely check their WordPress installations to verify that they are now running the latest release of Ninja Forms. So in other words, if you know you're a Ninja Forms user, and one million plus installations are, hopefully they all got fixed. But if you know you're such a user, make sure that you're running an updated version.

And in sort of a related deserialization problem, there was another vulnerability and attacks on a very popular JSON library known as FastJson. It has a CVSS of 8.1 and was recently found and fixed. It was patched at the end of May and affected all Java applications that rely on the FastJson versions 1.2.80 or earlier. And I'm not going to go into detail. It was an issue where a user-provided content could be deserialized into a Java class. Now, that's obviously bad. A user, if a user can provide a blob, then deserializing it turns it back into Java code, which then gets instantiated and run. So obviously that's a remote code execution vulnerability.

The guys who had this realized that this was dangerous. So unfortunately they created a block list of objects that they did not want to be allowed to be deserialized. Well, that's obviously a bad idea. It's like having a firewall where you close the ports for known attacks. Okay, we tried that in the beginning. That didn't work. So they should have done a deny-all and then an allow whitelist. Anyway, it's been fixed. The function that was enabled by default called AutoType is no longer enabled by default. Things have been cleaned up. If you have any connection to anything known as FastJson, then you'll want to make sure that you're running the latest version.

Okay. Finally, some bits of Miscellany. A few weeks ago I mentioned that I had appeared as a guest of one of our own listeners on his own Trek Profiles podcast. Last Wednesday John tweeted. He said: "Now comes Episode 66 of Trek Profiles! In this one, we speak with Steve Gibson about his..."

Leo: I know him. He's famous.

Steve: "...about his Star Trek fandom, his history with the show, and we endeavor to discover why we all love Trek so much." He says: "Plus, you'll get to meet the mysterious Bunn-Ons. Get it wherever you get your audio." And so it's called Trek Profiles, for anyone who's interested. And one thing that John did, and I did know he was going to, he took the recording he had made of me saying "We are the Bunnons! Surrender your ship or be destroyed!"

Leo: Yeah.

Steve: And our listeners will remember that that is, backwards, that's yo-sha ba-di-dro, pa-shor-yor-nar-ros, sna-na ba-na-ni. I said that on the podcast. Well, he reversed the audio.

Leo: Oh, good for him.

Steve: And it worked. Now, the timing on the "We are the Bunnons" part I got a little bit off. But after all it was 50 years ago, and I still remember those words. But the rest of it was intelligible as "Surrender your ship or be destroyed."

Leo: Oh, my god. I really have to listen to this. It's on, for some reason, it's on Amazon Music. Maybe that's - it can't be the only place it is.

Steve: I think he said get it wherever you listen to podcasts.

Leo: Oh, good. Okay. Okay. So the link you put in your show notes goes to Amazon Music.

Steve: Yeah. That's the link that he had provided, and so I just put it in the show notes for anybody who has it.

Leo: Nice.

Steve: Okay. Closing the Loop. First was from jvstech. He said: "Hey Steve. I'm sure you've had several others already mention this, but USB type A connectors" - remember, which we talked about last week as how they demonstrate that we're in a simulation because they're always wrong the first time you try to plug them in. "USB type A does not break probability. It is, in fact" - and this does explain it, actually - "in a state of quantum superposition. The connector exists in both the correct and incorrect orientations simultaneously until observed, at which point the wave-function collapses and a single orientation manifests. This is why it often takes three tries to plug it in correctly." And he says: "Thanks for the amazing podcast, by the way."

And there was an unrelated tweet that I actually showed, saying "It's a well-known fact that you must spin a USB three times before it will fit. From this we can gather that a USB has three states: up position, down position, and superposition." So apparently a lot of people have had fun with this sort of thing in the past.

Brian Tillman tweeted: "Here's a question I have. Suppose I have an account with some service, and they decide to support SQRL. How do I get my SQRL identity associated with an existing account?" And so I wanted to share Brian's question since I'm sure that FIDO's passkeys will be working the same way. So it actually has a future practical application. In all cases, an additional method of authenticating is simply added to an account. So you would logon first through your traditional means, presumably a username and password. Then under your account settings you would choose to add a SQRL, or more likely a FIDO passkey. That would trigger an authentication transaction from the website. You would authenticate with your client, which would synthesize a pair of FIDO keys, and the client would provide the website with your specific public key for it to keep on record.

In the case of FIDO's passkeys, that's as far as it goes. But with SQRL, since we thought through the entire paradigm, we realized that since a system's security is limited by the security of the weakest link, adding a super-secure means of authenticating doesn't actually increase security unless and until you also eliminate the possibility of using all weaker links. In other words, if the site still allows you to log on with username and password, it can still get hacked, and you can still lose control of your password just like you always have been able to.

That's why SQRL goes a step further and is able to request that a website disable all non-SQRL authentication. Once a user has become familiar with SQRL and is confident in its use, they're able to set a switch in their SQRL client, which when they then visit any websites, causes their SQRL client to request that sites no longer honor their traditional username and password, or any other non-SQRL means of authentication. Which at that point actually does elevate your security to the level of a true WebAuthn-style SQRL or FIDO-style authentication. Again, another example of what SQRL does that the FIDO people just didn't bother with.

Jose C. Gomez tweeted: "Hi Steve. I am the owner/host of SQRL OAuth that Leo uses on the forum. I had wandered off onto other things. I got alerted by Jeff Arthur" - Jeff of course is the author of the iOS SQRL client. He says: "Thank you, Jeff. And everything is back up and running. I have added some monitoring tools to keep a better eye on it. I honestly thought nobody was using it." Which may have been the case. He says: "I wish it would get more adoption," he says, "SQRL in general. But alas, with all this new FIDO, et cetera, I suspect it isn't going to happen. Nevertheless, I have fixed the issues, brought the site back up, and even did some bug fix so the TWiT community should be back up and running for those that want to use it. Thanks for the heads up, and thanks again to Jeff for reaching out. Cheers."

And David Lemire said: "Hi Steve. Listen every week and always enjoy the show. But I'd like to suggest you use as careful a definition for IoT as you do for zero-day. You often point out how Microsoft's use of the term" - meaning zero-day - "isn't really appropriate." Indeed.

He said: "Well, NIST has a definition for an IoT device in NIST IR 8259, which has also been adopted into U.S. public law in the Cybersecurity Improvement Act of 2020. It says such devices" - that is, IoT devices - "have a network interface for interacting with the digital world and a sensor or actuator for interacting with the physical world. I bring this up because you often use routers as examples when you talk about IoT; but by the NIST definition, which is pretty well accepted, a router isn't IoT because it lacks an interface to the physical world. I know you like to be precise about things so I thought you'd want to know about the NIST definition."

So David, thank you, yes. And I think that's interesting. And I think that we do have a problem here with a weak definition. IoT is weakly defined. My original feeling about the use and definition of IoT devices was actually more along the lines of what David cites as NIST's formal definition, things like light switches and plugs and Internet-connected thermostats, all of which qualify under that definition. But my original definition has been broadened, I think usefully and appropriately, to now include what I would call "unattended devices" as the primary distinctive feature which determines their IoT-ness for the purposes of concerns about security and their abuse.

Is a router an IoT device? Increasingly, the Internet security community is adopting that definition. So I don't know what to do. If something like a router is not an IoT device, then it would be nice to have some broadly agreed upon term for the class of Internet-connected and often quite powerful devices that can have firmware flaws allowing them to be remotely subverted. I think we're stuck with IoT, and we need to call those things something. So I think it's probably IoT.

And finally, chuck3000 said: "Re SN-875. Here is a reason for an electronic pet door." He said: "Apparently this is common in Florida." And I've got a photo in the show notes that shows an alligator entering someone's home through the swinging flap which was insufficient at repelling the alligator. I imagine he's looking for the pet cat.

Leo: I admire the Floridian, the Florida man who had the intestinal fortitude to stand there and take that picture. Because I wouldn't have.

Steve: Good point. Someone took that picture.

Leo: I would not have stood there. The alligator's mouth is wide open, and he's a foot or two away. And those things move fast.

Steve: Apparently they don't turn rapidly.

Leo: Oh, okay.

Steve: Lorrie spent a lot of time in Florida, and she told me that what you learn is to zigzag.

Leo: Ah.

Steve: Because they can run in a straight line, but they're not good at turning.

Leo: See, you've learned something on Security Now! today, how to run away from an alligator. That's wild. What a picture. Oy.

Steve: Yeah.

Leo: Okay. That's a good reason to have a chipped cat door.

Steve: Yeah. And here comes my water.

Leo: By the way, here's a weird one. So I went to your Cookie Forensics site in Microsoft Edge with default settings.

Steve: Yeah?

Leo: Nothing.

Steve: Whoa.

Leo: It's actually better than Firefox. No icon issues. I find that hard to believe.

Steve: It's Chromium based, and Chrome did the same thing for me. It behaved itself, although I had assumed it's because I'd turned on a bunch of stuff.

Leo: This is default, I think. I didn't mess with this.

Steve: Yeah.

Leo: So, wow.

Steve: That's cool.

Leo: Yeah. Well, yeah, it's on basic tracking protection.

Steve: Wow. And if you turn off tracking protection?

Leo: I don't even have it turned on. I don't even have it turned on. Which of course it's Microsoft, so it's off by default. Send do not track request? No. Yeah, wow. It seems to me maybe they've got sneaky ways of getting cookies out of there.

Steve: That's interesting. Well, I don't doubt that our listeners are going to dig into that, given this tool, and they'll let me know what's going on.

Leo: Yeah. Weird. All right. We are going to pile on Microsoft in just a second. All right. On we go with the show.

Steve: Okay. So today's podcast title, "Microsoft's Patchy Patches," was chosen after encountering a number of separate and independent pieces in the tech press, all decrying Microsoft's recent vulnerability and patch handling. Since this has been something that we know I've been observing and repeatedly noting here, it was somewhat comforting to get a bit of a reality check that my perceptions are not coming out of left field. Dan Goodin, writing for Ars Technica, published a piece headlined: "Botched and silent patches from Microsoft put customers at risk, critics say. Case in point: It took five months and three patches to fix a critical Azure threat."

And Jonathan Greig, writing for The Record, separately published: "Debate rages over Microsoft vulnerability practices after Follina and Azure issues." Since Jonathan's reporting contained some new information from interviews he conducted across the industry, some from veteran Microsofties, I'll start by sharing some of what Jonathan found.

Microsoft finally released a patch for the much-discussed Follina vulnerability - we've talked about it a number of times, that was CVE-2022-30190 - amid fixes for 55 other issues last Tuesday. But Microsoft's initial response to the issue, as we know, and several others, has stirred debate among security experts who question Microsoft's recent handling of vulnerabilities. Microsoft initially claimed Follina "wasn't a security issue" after being sent evidence by the head of advanced persistent threat hunting organization Shadow Chaser Group. They eventually acknowledged the issue, but several security experts have aired concerns about Microsoft's responses to a number of vulnerability reports.

Last Monday, Amit Yoran, the CEO of the cybersecurity firm Tenable, published a lengthy blog post criticizing Microsoft for its recent response to two disclosed vulnerabilities affecting the Azure Synapse service. In his blog posting, Amit wrote: "After evaluating the situation, Microsoft decided to silently patch one of the problems, downplaying the risk. It was only after being told that we (Tenable) were going to go public, that their story changed, 89 days after we initially notified them of the vulnerability, when they privately acknowledged the severity of the security issue. To date, Microsoft customers have not been notified. This is a repeated pattern of behavior. Several security companies have written about their vulnerability notification interactions with Microsoft, and Microsoft's dismissive attitude about the risk that vulnerabilities present to their customers."

Yoran went on to say that Microsoft's frequent reticence to notify customers of issues was "a grossly irresponsible policy." In response to questions about Yoran's comments, Microsoft told Jonathan Grieg, reporting for The Record, that it only assigns CVEs to issues that require customers to take action. What? So now they're not vulnerabilities if Microsoft handles them secretly?

The Microsoft spokesman said: "We addressed the issue that Tenable reported to us, and no customer action is required." This apparently is Microsoft's new "sweep it under the rug" policy. And really, when you think about it, isn't this exactly what a behemoth that's unanswerable would do if it's unable to act responsibly? "Nothing to see here. What CVE?"

Aaron Turner, CTO at the security company Vectra, said he understood both sides of the debate as a longtime former Microsoft security team member. Microsoft wants to have the freedom to manage their cloud services the way they see fit, Turner said. He said: "I was at Microsoft in the worst of times, from 1999 through 2006, when the company had to go from some of the worst security management policies to eventually leading the industry in predictability, transparency, and one of the best supporters of responsible disclosure."

Turner explained that he knows and respects Yoran personally, but did not think that Tenable's blog post was constructive. The rules around responsible disclosure do indeed need to be updated, according to Turner. But he noted that both sides have room for improvement. Well, I'd like to hear more, but he didn't offer it, at least for this article.

And I'll note that, if Turner left Microsoft in 2006, when things were going great, according to him, then he will have missed the events of the past 16 years, when things have definitely taken a turn for the worse. In a few minutes I'll be sharing some thoughts from someone who once tested Windows, before Microsoft decided that actual testing was no longer needed.

Anyway, Turner said there needs to be clearer rules around research into core Platform as a Service and Infrastructure as a Service technologies - in other words, how to deal with Cloud stuff - as well as easier ways for cloud platform operators to provide testing capabilities to researchers and clear responses to responsibly disclosed vulnerability information. And that's what Tenable wants.

Several other researchers were less forgiving of Microsoft, pointing out that more than 33% of the vulnerabilities added to the CISA Cybersecurity and Infrastructure Security Agency's list of known exploitable bugs, 33%, more than, came solely from Microsoft. One third of all. Microsoft had the most vulnerabilities added to the list in every month this year. And those are known exploited bugs.

Andrew Grotto, former White House Director for Cybersecurity Policy, who is now a cybersecurity professor at Stanford University, argued that Microsoft's market dominance was part of the problem. Yeah, no kidding. As we know, one of my theories has been that they just don't need to do anything since there are no consequences when they do not.

Anyway, Grotto explained: "The data speaks to an outsized representation of Microsoft products having the most critical vulnerabilities. On some level it may reflect the sheer prevalence of Microsoft products, but it's not like there aren't other vendors whose products are constantly being poked and prodded and tested. No other vendor appears with the same frequency and level of severity in terms of vulnerabilities that Microsoft's products seem to," says Grotto. "Does the market force Microsoft to remedy this problem or not? What worries me," he says, "is right now there is not a ton of competition, so I'm a bit pessimistic about this trend changing."

Steven Weber, professor of the Graduate School of Information at UC Berkeley, said procurement is the best way to drive positive changes in security practices. Government procurement practices right now are making the government less secure, but also hurting the private markets as well, Weber explained, because it is not creating greater demand for better security. He said: "It's important to keep in context that the widespread market penetration of a company's products is no explanation for why its products are also the most vulnerable." Amen to that. "What we ought to be asking is, given that we know and are shown again and again that Microsoft's products are highly vulnerable, why do they remain so prevalent in the market?" Well, we know.

And Dan Goodin's reporting added some additional depth to this topic. Reporting in Ars Technica, Dan observed: "Blame is mounting on Microsoft for what critics say is a lack of transparency and adequate speed when responding to reports of vulnerabilities threatening its customers. Microsoft's latest failing came to light on Tuesday in a post" - that's last Tuesday - "in a post that showed Microsoft taking five months and three patches before successfully fixing a critical vulnerability in Azure." And I'll just say that part of the problem here is that everyone wants, those people finding these problems want to behave responsibly. They want Microsoft to fix this. Yet Microsoft is under no pressure to fix it because these companies will wait until they do. Maybe that's the mistake that they're making.

Anyway, Orca Security first informed Microsoft in early January of this year of this very critical flaw, which resided in the Synapse Analytics component of the cloud service Azure and also affected the Azure Data Factory. It gave anyone with an Azure account the ability to access the resources of other customers. "From there," Orca explained, "an attacker could gain authorization inside other customer accounts while acting as their Synapse workspace." They wrote: "We could have accessed even more resources inside a customer's account depending on the configuration. It would leak customers' credentials stored in their Synapse workspace; communicate with other customers' integration runtimes saying we could leverage this to run remote code execution on any customer's integration runtimes; and take control of the Azure batch tool managing all of the shared integration runtimes. We could run code on every instance."

In other words, this was a horrible, critical vulnerability. Yet Orca said that despite the urgency of the vulnerability, Microsoft responders were slow to grasp its severity. Microsoft botched the first two patches. And it wasn't until just this past Tuesday in June, June's Patch Tuesday, that Microsoft issued an update that entirely fixed the flaw, and Orca finally felt it was safe to talk about this. A timeline provided by Orca shows just how much time and work it took them to shepherd and coax Microsoft through the remediation process.

On January 4th the Orca Security research team disclosed the vulnerability to the Microsoft Security Response Center (MSRC), along with keys and certificates they were able to extract. So that was part of the vulnerability was the ability to extract keys and certificates which then empowered to do this. A month goes by, February 19th. Another month, March 4th, MSRC requested additional details to aid its investigation. Each of those two times, on February 19th and March 4th, Orca responded the next day.

Now we're in late March. MSRC deployed the initial patch. March 30, Orca was able to bypass the patch. Synapse remained vulnerable. March 31st, Azure awards Orca $60,000 for their discovery. Oh, joy. But Azure remains vulnerable. April 4th, 90 days after disclosure, Orca Security notifies Microsoft that keys and certificates are still valid. Orca still had Synapse management server access. Three more days, April 7th. Orca met with MSRC to clarify the implications of the vulnerability and the required steps to fix it in its entirety. Three more days, April 10th, MSRC patches the first bypass, and finally revokes the Synapse management server certificate. Orca was able to bypass the patch yet again. Synapse remained vulnerable.

Five more days. Now we're at April 15th. MSRC deploys the third patch, fixing the RCE and reported attack vectors. May 9th, both Orca Security and MSRC publish blogs outlining the vulnerability, mitigations, and recommendations for customers. End of May, Microsoft deploys more comprehensive tenant isolation including ephemeral instances and scoped tokens for the shared Azure Integration Runtimes. In the end the fix was silent, with no notification ever provided to Microsoft customers that they had ever been in this danger and had been for five months.

And this account from Orca followed 24 hours after the previously mentioned security firm, Tenable, related a similar tale of woe of Microsoft's failing to transparently fix vulnerabilities that also involved Azure Synapse. And this is wonderful. In a statement, Microsoft officials wrote: "We are deeply committed to protecting our customers, and we believe security is a team sport." What? What are you guys smoking up there in Redmond?

Leo: The Tigers bought uniforms.

Steve: Yeah. The security of your proprietary software is a team sport? They said: "We appreciate our partnerships with the security community" - although of course we don't act as if we do - "which enables our work to protect customers." Right, because we're not doing security in-house anymore. Apparently we're relying on outsiders. Whoa. "The release," they said, "the release of a security update is a balance between quality and timeliness." Yeah, right, we wouldn't want to rush it out and get it wrong. Oops, we did. Three times. Oh, well, never mind. "And we consider the need to minimize customer disruptions while improving protection." Oh, so we don't want to notify them. That might be disruptive. Right. Wow.

So I recalled that a few years ago Microsoft laid off their large and expensive testing teams. So I went looking for, and found, a description of exactly what changed and how, and why today's system has become so badly broken. At least in part why. Martin Brinkman, writing for Ghacks.net, had a piece a few years ago titled "Former Microsoft Employee explains why bugs in Windows updates increased."

Have the number of bugs in Windows updates increased in the past couple of years? If so, what's the reason for the increase in bugs? What might it be? That's the question that former Senior SDET, that stands for Senior Software Development Engineer in Test, Jerry Berg, recently answered. Berg worked for 15 years at Microsoft, where one of his roles was to design and develop tools and processes to automate testing for the Windows OS. He left the company after Windows 8.1 shipped to the public. Microsoft changed testing processes significantly in the past couple of years. Berg described how testing was done in the late 2014 and early 2015 period, and how Microsoft's testing processes changed since.

Back in 2014 and '15, Microsoft employed many teams that were dedicated to testing the operating system, builds, updates, drivers, and other code. The teams consisted of multiple groups that would run tests and discuss bugs and issues in large daily meetings. Tests were conducted manually by the teams and through automated testing and, if tests passed, would give the okay to integrate the code into Windows. The teams ran the tests on "real" hardware in labs through automated testing. The machines had different hardware components - processors, hard drives, video and sound cards and so forth - and other components to cover a wide range of system configurations.

Then Microsoft laid off almost the entire Windows Test teams, and it moved the focus from three different systems - Windows, Windows Mobile, and Xbox - to a single system. The company moved most of the testing to virtual machines which meant, according to Berg, that tests were no longer conducted on real and diverse hardware configurations for the most part. They were tested on generic VM's.

Microsoft employees could self-host test releases of Windows, which would mean that their machines would also be used for testing purposes. The main idea behind that was to get feedback from in-house Microsoft employees when they encountered issues during their work days. Berg notes that self-hosting is not as widely used anymore as it was before. Right, because who wants to be a tester when that means that one's main machine will be crashing during their day.

So now today the primary source of testing data, apart from the automated test systems that are in place, running on VMs, comes from outside Microsoft, from Microsoft's users, through Telemetry and Windows Insiders. Windows Insider builds are installed on millions of devices, and Microsoft collects Telemetry from all of these devices. If something crashes, Microsoft gets information about it. Unfortunately, one of the problems associated with the collecting of Telemetry is that most bugs are not caught by it. If something does not work right, Microsoft may not be able to discern the relevant bits from Telemetry data. While it is in theory possible that users report issues, many don't. Additionally, while Insiders may report bugs, it is often the case that necessary information is not supplied to Microsoft, which poses huge issues for the engineers tasked with resolving these problems.

Back in 2014 and '15, Microsoft's Testing team would be tasked with analyzing bugs and issues, and supplying engineers with the data they required, patches, to resolve these. Nowadays, Berg notes, all of that is gone, and it's Telemetry that the engineers look at to figure out how to fix these issues, and fixes are then pushed to customer devices running Insider Builds again to see if the issue got fixed or if it created new bugs.

One of the main reasons why Microsoft stopped pushing out new feature updates to everyone at once was that issues that were not detected by that previous process could potentially negatively affect a large number of customers. Yeah, no kidding. To avoid total disasters like the Windows 10 version 1809 launch, gradual rollouts were introduced that would prevent feature updates from being delivered via a Windows update to the majority of machines in the early days of the release.

So Microsoft exchanged the dedicated in-house testing team for remote Telemetry data that it gathers from Insider Builds that it pushes to consumer and business devices, and replaced much of the PCs that it once used for testing with virtual environments. All of that has led to an increased number of issues and bugs that customers face on production machines when installing Windows updates or feature updates. And it appears to have created disconnection throughout various major arteries of Microsoft.

I have no doubt that individual Microsoft employees are doing the best they can with what they have. But management appears to have royally screwed the pooch when they decided to disband serious prerelease testing in favor of collecting outside telemetry from Windows Insiders. And we've already talked about what Microsoft plans to do next by further automating the Windows Update system to dynamically back out when things go wrong and to learn from its mistakes. This essentially broadens its Telemetry collection from Windows Insiders to include all other commercial Windows users.

Stepping back to look at the big picture, it becomes clear that what Microsoft has essentially done by disbanding serious internal testing on real hardware, switching to limited testing on virtual machines and outsourcing its software testing first to Windows enthusiasts and then to all Windows users, is to turn us all into their unpaid beta testers. And as we've seen in example after example, the external security community becomes their thankless security research arm.

So someone hearing this says to me: "Okay, Gibson. What OS are you using?" And we all know the answer to that. I'm sitting in front of Windows. I've always been sitting in front of Windows. And before that I was sitting in front of DOS. I also often sit in front of FreeBSD UNIX, which runs several of my most important servers. And I spent a lot of time in front of Debian Linux when I was working to recover that Lenovo laptop's weird-acting BitLocker-encrypted NVMe drive that had died in a mysterious way and would no longer boot. That's the one that I managed to get finally working by hooking it to an external PCIe Thunderbolt port. I used Debian then thanks to the powerful command-line tools that Linux offers.

But the fact remains I am most comfortable sitting in front of Windows. The tools I prefer using are hosted only there. And the tools I'll be using for SpinRite 7 when I move away from DOS are only hosted on Windows. My failure to preemptively show Microsoft the folly of shipping a consumer OS with userland access to raw sockets taught me that Microsoft had become a blind behemoth. So I have no illusions that we can change Microsoft. But understanding the path and the trajectory that Microsoft's policies have put it on remains valuable. And so I think we can predict the future.

Leo: Terrible.

Steve: It's not good.

Leo: Yeah. It's why I do not use Windows. I know you need to, but...

Steve: I hear Paul and Mary Jo on Wednesdays just saying, "We don't understand what's going on." I mean, they don't. They're as inside as you can be, and they're just like, yeah, we don't know what happened.

Leo: Yeah. I have a Windows machine sitting in front of me. This is the only one I have. Just because I guess I have to help people figure out what to do.

Steve: Yes, you do. Yup.

Leo: That's no fun.

Steve: It's a love/hate relationship.

Leo: You know, I mean, most operating systems are kind of the same and do the same thing.

Steve: There certainly has been sort of an aggregation around...

Leo: Yeah, they're very similar.

Steve: ...the right way to do things.

Leo: Yeah. And I think anybody who uses Windows, if they picked up, I wouldn't say Debian, but maybe Ubuntu, which is based on Debian, or Pop!_OS, or Manjaro Linux, which is what I use, would feel pretty much right at home almost right away using GNOME or KDE. It still baffles me. But, you know, nobody ever got fired for buying a Microsoft Windows machine.

Steve: That's exactly right. Microsoft has become the IBM of yesteryear.

Leo: Yeah, yeah. Thank you, my friend, as always. You'll find Steve Gibson at GRC.com. That's the Gibson Research Corporation. But it's of course the home of SpinRite, the world's best hard drive or mass storage maintenance and recovery utility. You'll also find a lot of other good things there, including this show. He has a unique, couple of unique versions: a 16Kb audio version, which sounds a little scratchy but is small, smaller than that bottle. And, wow. That's a lot of water. And he also has transcripts, which are great for reading along. And I guess you said the transcripts are not as popular as the show notes. But I think for searching, they may not be downloading them, but for searching they're very, very valuable.

Steve: Oh, no comparison, yes, super valuable.

Leo: Show notes are the kind of thing you almost would want to subscribe to, like it's a newsletter every week with the contents and the images and the links from the show, which I think is fantastic. That's also at GRC.com. We have video, oddly enough, of the show. If you want to see Steve's giant bottle, that's at TWiT.tv/sn. There's also of course a YouTube channel dedicated to it. We also have audio versions at TWiT.tv/sn. Or you can subscribe in your favorite podcast player and get it automatically.

You can watch us record this show Tuesdays at about 1:30 to 2:00 p.m. Pacific, that's 4:30 Eastern, 20:30 UTC. The stream is at live.twit.tv. The chatroom is going, it's hot and heavy, irc.twit.tv. That's open to all. And of course the Discord is also chatting away.