
# Formal Verification

**Description:** This week we look at an important security update to Android for Firefox. We bid a fond farewell to Firefox Send and Notes. We look at the promise and growing popularity of the disastrously-named DuckDuckGo Internet search service. We dig into what's behind last Friday's Emergency Directive 20-04 from the DHS/CISA. We'll also take a look at the recent privacy and security improvements incorporated into Android 11 and iOS 14. We have a bit of errata, closing-the-loop feedback, and SpinRite news. Then we're going to take a look at the need for Formal Verification of our complex security protocols going forward in the context of another critical failure of a massively widespread system.

High quality (64 kbps) mp3 audio file URL: http://media.GRC.com/sn/SN-785.mp3
Quarter size (16 kbps) mp3 audio file URL: http://media.GRC.com/sn/sn-785-lq.mp3

---

SHOW TEASE: It's time for Security Now!. Steve Gibson is here. Coming up, a new search alternative that has a very silly name. We'll also talk about Zerologon, an extremely serious, maybe the most serious Windows exploit ever, and what to do to avoid it. And then, finally, the issue of formal verification of protocols. How can we make sure that our protocols are secure? It's all coming up next on Security Now!.

**Leo Laporte:** This is Security Now!, Episode 785, recorded Tuesday, September 22nd, 2020: Formal Verification.

It's time for Security Now! - oh boy, oh boy, oh boy! - the show where we cover your security and privacy online with this man right here, Steven Gibson. Steve is in a new location.

**Steve Gibson:** Coming to our listeners from an alternate location.

**Leo:** It's beautiful.

**Steve:** Well, it's where I am when I'm not in my bear's lair. I was working away on the podcast, watching you guys talking on MacBreak Weekly sort of in the background, just sort of keeping a background sense for where you were along the way. As soon as it's time for everyone's favorite picks of the week I think, okay. And so I was working actually on the main topic, which as a consequence gets a little short-changed this week because suddenly the audio stopped.

And I thought, oh, okay. And I looked over, and I had the little spinning wheel on the player. And then I looked over at my main bandwidth monitor, and it was all red. And so then everything looked okay on the cable modem, but I thought, well. And I waited for a while. It didn't come back. So I rebooted the cable modem. It never reachieved sync at the very first of the five things that flashes. It never even got resynced. So I thought, oh, that's not good. So I dug out a previous cable modem and also had to find its power supply. That took a while. Actually I found a power supply, but it was happy with it. And so fired it up, and it also would not sync to the cable. So I think we just had a local neighborhood outage.

Fortunately, as I mention from time to time, I now have an alternate location where I spend my evenings. This is why I needed sync, some sort of a sync solution is to keep separate work environments synchronized. So I thought, well, okay. And I know you, Leo, when there have been brief interruptions of things, you're just, like, okay, whatever it takes, we'll be patient.

**Leo:** Yup.

**Steve:** Make a podcast.

**Leo:** That's life, yup.

**Steve:** And so anyway, so I grabbed my hook, line, and sinker. I got my microphone.

**Leo:** I can't believe you moved the microphone and everything. That's impressive.

**Steve:** And the Focusrite, the headphones, everything's all set up. And so we're going to have a podcast. This one is titled "Formal Verification," only because I'm really having a problem with the maximum length of what I would like our titles to be. But I thought, okay, let's just cut to the quick: Formal Verification. We have had another, in as many weeks, critical vulnerability discovered in Bluetooth Low Energy connections. And it's a consequence of the fact that we're still not formally verifying our security protocols. We've been talking about this recently, the idea that we now have the technology, and we have the academic maturity, to create representations of complex security protocols and actually simulate them before we ever implement them to make sure they do what we think they do.

Well, once again, that wasn't done. And the problem was found through seven Purdue researchers basically building a formal model of Bluetooth LE and saying to this model, is anything broken? And yeah, sure enough, and it's really bad. So we'll talk about that, which I think is really interesting. And this sort of continues this idea that we've had. We've been talking about this in different contexts for the last few months because it keeps coming up. And it's just no longer okay for these critical security protocols to just be thrown together ad hoc.

Anyway, there was an important security update to Android for Firefox. And Leo, we're going to bid a fond farewell to Firefox Send and Notes.

**Leo:** This is why we can't have nice things on the Internet.

**Steve:** That's exactly right. We're also going to look at the promise and growing popularity of the - I just can't help but think this is disastrously named - DuckDuckGo. I mean, how do you take anything serious that's called DuckDuckGo? Luckily, it was shortened from Duck, Duck, Goose.

**Leo:** Yeah, that's that old kids' game. Did you ever play that?

**Steve:** Never played. And anyway...

**Leo:** Were you ever a kid? I'm just curious.

**Steve:** I tried to skip that. It was painful. And I spent all my time in the library.

**Leo:** No childhood, yeah, yeah.

**Steve:** Yeah, exactly. So anyway, it's turning out to be enjoying at the moment exponential growth. And unfortunately, in this era of COVID, everyone has learned what an exponent is, even if they didn't know before.

**Leo:** Yeah.

**Steve:** So we're going to talk about that briefly. Also we're going to dig into what's behind last Friday's Emergency Directive 20-04 from the DHS/CISA. Then we'll take a look at the recent privacy and security improvements incorporated into Android 11 and iOS 14. They've done some nice things. And there's a bit of sort of some philosophy I want to apply to this. We've got a bit of errata, some closing-the-loop feedback, a little SpinRite update, and then we're going to take a look at the growing need for formal verification of our complex security protocols going forward, in the context of, as I said, another critical failure in a massively widespread system, Bluetooth.

**Leo:** Yeah. That's actually a really good topic for this show. If anybody could call for some sort of standards around these new technologies, it would be you. I mean, we have the IEEE. We have the IETF. We have WWC. We have these standards bodies. I don't know how careful they are. Well, anyway, we'll listen. It'll be very interesting.

**Steve:** So, yeah. They're great at creating standards. The problem is they're just sort of writing it down and saying, "Hey, Joe."

**Leo:** Yeah. It's up to you to make sure it happens, yeah.

**Steve:** "Does this look good to you?" And Joe says, "Yeah, it looks good to me." No, I mean, these are not mistakes in the standard. Okay, there's some weakness in the way the standard was written. But it shouldn't have ever been published the way it is. And these academicians were able to model that and say, oh, look, it doesn't work the way we think it does.

**Leo:** Well, as usual, it's no surprise with Bluetooth, which has never worked the way it should. As usual, a lot of stuff coming up.

**Steve:** So an important and interesting LAN attack bug was recently fixed in Firefox 79 for Android. It only affects the Android version of Firefox, not the desktop version. So it turns out that Firefox on mobile locates other devices on the same LAN to share or receive content. An example might be sharing video streams with a Roku player, for example. To accomplish this location need, Firefox uses the Simple Service Discovery Protocol, SSDP. And if that sounds familiar to our listeners, it's because we've often spoken of it and its quite mixed-blessing capabilities, limitations, and security shortfalls. Although it was normally intended to be a freestanding protocol, its adoption by the infamous Universal Plug and Play (UPnP) system moved it into the UPnP spec.

It's a simple, text-based protocol based on HTTP over UDP. And the system uses LAN-wide multicast addressing. This is part of the problem. On an IPv4 LAN, for example, packets sent to port 1900, which is the UPnP SSDP port, addressed to 239.255.255.250 - that's a reserved IP address on Ethernet. That's one of the multicast addresses, which means that everybody on the same LAN will hear it and pick it up. It'll be received by all the devices. So Firefox uses this SSDP to query all devices on a local network for anything that Firefox might be interested in sharing. When a device is discovered, Firefox's SSDP component retrieves the location of an XML file where that discovered device's configuration is stored.

So Chris Moberly, an Australian security researcher working for GitLab, discovered that Android "intent commands" - and we'll get to that in a second - intent commands could be placed into this returned XML file, and that upon Firefox's retrieval of the file on Firefox for Android, those embedded intent commands would be executed.

**Leo:** Oh, please.

**Steve:** I know.

**Leo:** Sanitize your inputs, for crying out loud. This is the oldest trick in the book. Geez.

**Steve:** I know. So I thought, okay, what's an intent? Android developer page explains intents with the title "Sending the User to Another App," which does not sound good from a security standpoint. The developer page says: "One of Android's most important features is an app's ability to send the user to another app based on an action it would like to perform. For example, if your app has the address of a business that you'd like to show on a map, you don't have to build an activity in your app that shows a map. Instead, you can create a request to view the address using an 'intent.' The Android system then starts the appropriate app that's able to show the address on a map."

**Leo:** Apple has a similar scheme. It's a URL-based scheme.

**Steve:** Exactly.

**Leo:** You do need that in a phone because you need inter-app communication.

**Steve:** Yeah, exactly. And so, for example, in iOS it's the scheme of the URL that triggers whatever it is. Well, for example, like mailto://. Anyway, they said: "As previously explained, you must use intents to navigate between activities in your own app." So they're widespread, if that's what you use even internally. And they said: "You generally do so with an explicit intent, which defines the exact class name of the component you want to start. However, when you want to have a separate app perform an action such as view a map, you would use an implicit intent."

Okay. So it's probably not difficult for any of our listeners to figure out that Android intents are powerful, and not something we'd like a remote attacker to be able to cause our Firefox browser to execute at will. But until Firefox for Android 79, that's exactly what was possible. A hacker could, for example, walk into any environment with a large WiFi network, maybe an airport or a large mall. That attacker would connect to the local WiFi, like everyone else has who's there, and begin spamming the network with SSDP packets advertising its own maliciously crafted XML file containing Android intents. Any Android user using a Firefox browser prior to 79 - so again, time to make sure you've got the latest Firefox browser - it would pick up the XML file and issue those maliciously designed intents, that is, Firefox itself would do that, causing whatever the attacker wanted to be executed.

**Leo:** Boy, that's nasty.

**Steve:** It is really nasty.

**Leo:** Boy.

**Steve:** So now that Firefox for Android has been updated - and again, everyone make sure you're using it - the vulnerability's discoverer posted further details along with proof-of-concept code and videos. Chris Moberly wrote on GitLab, on his page on GitLab, I've got the link in the show notes, he said: "The SSDP engine in Firefox for Android 68.11.0 and below" - and I'm not sure why he's saying that here because what we're told is that you have to have 79. So maybe they jumped from 68.11.0 directly to 79. But you want to make sure you've got 79. He says: "...can be tricked into triggering Android intent URIs with zero user interaction. This attack can be leveraged by attackers on the same WiFi network and manifests as applications on the target device suddenly launching, without the user's permission, and conducting activities directed by the intent." As you said, Leo, yikes.

"The target simply has to have the Firefox application running on their phone. They do not need to access any malicious websites or click any malicious links. No attacker in the middle or malicious app installation is required. They can simply be sipping coffee while on a caf's WiFi," he wrote, "and their device will start launching application URIs under the attacker's control."

He says: "I discovered this bug while the newest version of Firefox Mobile 79 was being rolled out globally. Google Play Store was still serving a vulnerable version at this time, but only for a short period. I reported the issue directly to Mozilla, just to be safe. They responded right away and were quite pleasant to work with, providing some good info on where exactly this bug came from. They were able to confirm that the vulnerable

functionality was not included in the newest version and opened some issues to ensure that the offending code would not later be reintroduced by mistake."

He says: "If you find a Firefox bug, I definitely recommend sending it straight to them." He said: "The process is very easy, the team members are smart and friendly, and it's a good way to support a project that has helped shape the way we" - clearly he's a fan of Firefox - "shape the way we use the web." He says: "As long as you have app updates enabled and have recently connected to WiFi, you should have received the new version and are safe from exploitation. You can verify this yourself by opening Firefox on your device, clicking the three dots next to the address bar, and navigating to Settings > About Firefox. If your version is 79 or above, you're safe."

So anyway, really interesting. I mean, an example of one of those things that you can imagine it's so easy to exploit. That's one of the other things we're seeing a lot lately is when a proof of concept is provided, when it's simple to do, it really is the case that there's a stratification of hackers. And things that are really difficult that require an elite hacker, they remain theoretical for a long time. But, boy, if it's a matter of here's a proof of concept that Chris posted on GitLab, any script kiddie can now implement this and see if they get lucky with somebody who for some reason might not have the latest Firefox.

And speaking of Firefox and Mozilla, I was happy to hear Chris talk about how open they were, and accessible and friendly, because we are losing Send and Notes. As our listeners know, the encrypted file sending service was a favorite, Leo, of yours and mine. So ever since that service was suddenly suspended due to rampant abuse, as we know, I've been checking back from time to time, keeping tabs on it. And as I've mentioned several times before, I've wondered out loud whether something more might be going on, since how difficult could it be, especially for them, to add some stronger anti-abuse features, like requiring a verified email address rather than allowing the completely unauthenticated use which was what they were doing before. Well, now the other shoe has dropped, and we know that not only is Send gone for good, but so is whatever Notes was. Actually I had never heard of it.

**Leo:** I hadn't, either.

**Steve:** But it turns out, yeah, it's an inter-Firefox note syncing service. Apparently not that popular, unlike Send, which was launched back in March of 2019. And of course Send became quite popular because it was one of the very few dead simple, easy to use, truly secure end-to-end encrypted and free large file sending solutions. You and I, Leo, both used it because it was easy to send a file to someone.

**Leo:** Yeah.

**Steve:** Unfortunately, it wasn't only legitimate users who found it handy. It turns out that Send was initially taken offline after ZDNet reported that the service was being constantly abused by malware groups who also liked both its end-to-end encryption and the fact that incoming files originated from a Firefox.com domain, which was very legitimate-looking. So actually malware guys were putting encrypted stuff on a remote server and then incorporating the link into their malware. Once the malware got in, the malware would then download the encrypted payload onto the local machine using Firefox Send. So, yikes.

And of course at the time Mozilla said that Send's shutdown would be temporary and promised a way to find and curb the service's abuse, its abuse by malware operators. But

after quite a bit more than a few weeks, things changed. As we recently noted here, Mozilla laid off 250 employees as part of an effort to refocus their business on commercial products. So most of the staff that was supposed to be working to reengineer Send is gone, and the ones who are still around are now working on commercial products, or at least products that have a potential commercial upside - Mozilla VPN, Firefox Monitor, and Firefox Private Network. So we'll keep tabs on that.

But in the meantime, what was it, it was Filemail was the one I was using before, and I dropped it in favor of Send. I guess I'll be going back to Filemail to fulfill that need. There is a Windows desktop version of Filemail. I think they have desktop instances for the various desktop platforms. So you don't need to use it. You can use a browser. But it's a little more robust, they say, if you use their desktop app.

So, okay. We've never talked about DuckDuckGo except maybe in passing. And maybe I'm showing my age, but just naming something DuckDuckGo and then saying, oh, yeah, this is the privacy-protecting anti-tracking search engine, it's like, okay, are you guys serious, really? DuckDuckGo? But what brought it onto my radar is it's breaking records. They had and recently reported that August was 2 billion searches, 4 million of their app extension installs, 65 million - and I really did appreciate this, they said "Guesstimated active users. We really don't know because... privacy."

And I thought, well, that's good that they don't know how many estimated users they have. They are going exponential. And, okay. So let's forgive them this DuckDuckGo. I actually dug around a little bit, like wondering what does that name mean? Where did it come from? We know that Google came from Googolplex, right, a really, really, really, really big number. So I thought, okay. And as you noted, Leo, I skipped over my childhood as much as possible.

**Leo:** Clearly, yes.

**Steve:** So what I learned, what didn't surprise me, DuckDuckGo is known, though I would argue not yet widely enough, which is why I knew it would be of interest.

**Leo:** It's pretty widely known. I think, for instance, if you go to Firefox and look for alternative search providers, along with Google there's DuckDuckGo. And I think that's probably how it's mostly known is as a search provider. But they've now really kind of expanded. In fact, it's ironic because this morning on iOS Today I recommended the DuckDuckGo browser on iOS. So they've really taken off. I don't know if that's a good thing. Usually it's a bad thing, to be honest.

**Steve:** Yeah. Well, okay. So it was launched 12 years ago, in 2008, by a guy named Gabriel Weinberg, who is an MIT grad. He had sold his previous operation, something called Opobox, to Classmates.com, for which he got $10 million in 2006. And after that he thought, okay, what am I going to do? He was juggling a few different concepts. He had the idea of structured data, a Quora-style Q&A, and maybe programmatically combating search spam. So in an interview he said: "I started all these projects independently, and none of them really took off. Then I thought, maybe if I put them all together, there might be an interesting experience there."

The result was DuckDuckGo, which he says is a search engine offering direct answers to people's queries. And that's kind of interesting because I find that I often ask Google a question. I don't just put in some search term keywords. I'll just - I might as well just

write it because I figure, maybe it's getting smart enough to actually listen, like all of our other voice assistants are. So I'll just often frame my searches as a query.

Well, it turns out that's exactly what DuckDuckGo expects you to do, and it tries to provide what they call "Instant Answers" at the top of the page that results from you asking it a question, rather than merely delivering a list of links. It does that, too, a little bit lower down. But otherwise, after the Instant Answers, the links it gets, they result from syndicated third parties like Bing, Yelp, and Yahoo. So I guess it's forwarding its queries, or it has a deal with them to, like, access their backend databases, whatever. But it is a privacy shield.

Unlike Google, DuckDuckGo has no filter bubble policy. And of course you and I, and I know you and Jeff and Stacey have talked about this idea that different people get different search engine results because the engine knows, it learns about us over time, which tends to create, you know, the idea is to get more relevant results. Unfortunately, it kind of gets self-reinforcing results at some point. DuckDuckGo can't because it doesn't track our browser, our search history, our device type, our IP address, our usage behavior, or anything. Everyone receives the same results from the same search query. And of course, as we know...

**Leo:** I love that, by the way. I use DuckDuckGo on Firefox. It's a perfect combination, if you ask me. I have for years.

**Steve:** Well, and I like the idea of getting non-colored results because they're not always what you want.

**Leo:** Sorry, I didn't mean to interrupt. I threw you off. I apologize.

**Steve:** No, no, no. It's completely okay. There was something I stumbled upon that I was really amazed at. And I can't see it here in my show notes. I'm just frantically scanning for it. It was really a cool - it was another way that we were being tracked that he mentioned that just startled me. Anyway, Google, of course, not only knows who we are, but according to Alphabet's executive chairman, Eric Schmidt, he was quoted saying, "We know where you've been, and we can more or less know what you're thinking about." Which, I mean, Google's now bragging about, despite their "Do no evil" motto.

Oh, here it is. To which Gabriel Weinberg would reply: "As long as you can tie searches together and you keep any shred of the information, any personal information that can tie things back to you," he says, "then I think it's not truly private." And so I never found out anything satisfying about the name. It's only that, as we mentioned before, it's actually a shortened version of Duck, Duck, Goose, which is some sort of kids' game. And so...

**Leo:** What is this "Hide and Seek" you people are talking about?

**Steve:** Indeed.

**Leo:** You would like it, though. Everybody sits in a circle. And you tap the kids' heads, and you go "Duck, duck, duck, duck." And the minute you say "goose," that kid now has to chase you, and you sit down. It's like tag in a circle. It's fun.

**Steve:** I'm really glad I missed it. You know, there was some comment about, like, oh, people are going to start saying "Duck it," the way people say, "Oh, just go google it." And I thought, no. No one is going to start saying "Go duck it."

**Leo:** No, not happening. Maybe that was his plan. Actually, that's interesting, yeah. Maybe that was it.

**Steve:** Keep working on it, Gabriel. You've got a ways to go. So, okay. Here we are on Tuesday, September 22nd, now two weeks downstream from this month's Patch Tuesday, which puts us six weeks down from August's Patch Tuesday. And after last week's blistering summary of the need to apply this month's patches, I cannot imagine that anyone would not have done so by now, let alone last month's patches. But Leo, difficult as it might be to believe, apparently not everyone listens to this podcast every week.

**Leo:** What?

**Steve:** I know, I know, I know. Some things just cannot be explained. That's okay. For those who are listening, here's what just happened. After giving the world what should have been ample time to apply the August Windows updates, last Monday, a week ago and a day, Secura responsibly disclosed, they published a detailed behind-the-scenes description of just one of the nightmares that Microsoft fixed last month. Not two weeks ago, six weeks ago. Their blog posting begins - I have a link in the show notes to the whole one, for anyone who wants it. They said: "Last month, Microsoft patched a very interesting vulnerability that would allow an attacker with a foothold on your internal network to essentially become domain admin with one click."

**Leo:** Ugh.

**Steve:** "All that is required," they wrote, "is for a connection to the domain controller to be possible from the attacker's viewpoint. Secura's security expert Tom Tervoort previously discovered a less severe Netlogon vulnerability last year that allowed workstations to be taken over, but the attacker required a person-in-the-middle position for that to work. Now he discovered this second, much more severe" - I was musing, I think it was last week because we talked about something horrible that was a 9.9. And I said, "Has there ever been a 10.0?"

**Leo:** Yes.

**Steve:** Uh-huh. And this is it, "...CVSS score 10.0 vulnerability in Microsoft's protocol. By forging an authentication token for specific Netlogon functionality, he was able to call a function to set the computer password of the domain controller to a known value. After that, the attacker can use this new password" - monkey, I'm sure they're going to use monkey. They have to, Leo, they absolutely just, you know, they have to set it to monkey. That's got to be part of the requirement for using this vulnerability. Anyway, "...to take control over the domain controller and steal credentials of a domain admin."

And of course if you're the domain admin, and you can log into the domain controller, it's game over for an enterprise.

They finish their little quickie: "The vulnerability stems from a flaw in a cryptographic authentication scheme used by the Netlogon Remote Protocol, which among other things can be used to update computer passwords. This flaw allows attackers to impersonate any computer, including the domain controller itself, and execute remote procedure calls on its behalf."

Okay. So Secura named this discovery and vulnerability "Zerologon" because it allows an attacker who initially has zero logon privileges on a network to completely take it over. If an attacker can establish not even a foothold, a toehold on any machine within a Windows-based enterprise network, there is no longer any containment. You've talked, Leo, about laterally moving within an enterprise. I mean, this just opens up all access. If malware finds some way, anyway, to run on any workstation, on any large Windows domain-based enterprise, it can obtain free rein over the entire network. It's able to trivially overwrite the credentials of the domain admin, resetting the password and completely taking over the enterprise. So that was Monday, eight days ago.

**Leo:** Oh, boy.

**Steve:** What happened next?

**Leo:** Oh, boy.

**Steve:** It won't surprise anybody. Secura's reported details, coupled with how juicy exploitation of this would be, and of course the opportunity to actually use the password "monkey" on a domain, it motivated the world's hackers, donning hats of all shades to quickly create weaponized proof-of-concept exploits that went public within hours of Secura's report. And the result of that is that the U.S. Department of Homeland Security CISA, last Friday afternoon, issued Emergency Directive 20-04. It's titled "Mitigate Netlogon Elevation of Privilege Vulnerability from August 2020 Patch Tuesday." And it's not a request.

After citing a few relevant sections of the Homeland Security Act of 2002, which gives the DHS authority to compel action on the part of any federal information system administrator, the directive says: "On August 11th, 2020, Microsoft released a software update to mitigate a critical vulnerability in Windows Server operating systems." And then they cite CVE-2020-1472. "The vulnerability in Microsoft Windows Netlogon Remote Protocol (MS-NRPC), a core authentication component of Active Directory, could allow an unauthenticated attacker with network access to a domain controller to completely compromise all Active Directory identity services. Applying the update released on August 11th to domain controllers is currently the only mitigation to this vulnerability," they said, "(aside from removing affected domain controllers from the network)," which of course is impractical.

"CISA has determined that this vulnerability poses an unacceptable risk to the Federal Civilian Executive Branch and requires an immediate and emergency action. This determination is based on the following," and they have five bullet points: "The availability of the exploit code in the wild increasing likelihood of any unpatched domain controller being exploited; second, the widespread presence of the affected domain controllers across the federal enterprise; the high potential for a compromise of agency information systems; the grave impact of a successful compromise; and the continued

presence of the vulnerability more than 30 days since the update was released." In other words, they have checked, and people have not updated from August, a month later.

**Leo:** It was part of the Windows patches, though; right?

**Steve:** Yes, yes. And it turns out, Leo, people actually aren't patching.

**Leo:** Oh, no.

**Steve:** They're just saying, okay, well, we'll get around to it. Everything seems fine right now. We're busy.

**Leo:** You know, our IT guy, Russell, I know you know who he is, he's really good. He's noticed some clients apparently have applied the patch, but apparently there's more that you have to do besides merely install the patch. There's a switch you need to flip or something. And they're thinking they're fixed when they've merely installed the patch. I don't know anything about that. I haven't tried it. But apparently there's more that you need to do.

**Steve:** So I don't think I put it in the show notes, but Secura has a utility that can be used to verify whether or not systems are vulnerable. So I'll keep proceeding here because I've got three links, for example, to the GitHub stuff. Anyway, so they finish, saying under Required Actions: "This emergency directive requires the following actions: Update all Windows Servers with the domain controller role by 11:59 p.m. Eastern Daylight Time, Monday, September 21, 2020." In other words, last midnight. By last midnight. Here we are on Tuesday. Midnight at the end of the day on Monday, the 21st, this thing mandated, period, no excuses, this is an emergency.

In their whitepaper, Secura wrote: "This attack has a huge impact. It basically allows any attacker on a local network, such as a malicious insider or someone who simply plugged in a device to an on-premise network port, to completely compromise the Windows domain. The attack is completely unauthenticated. The attacker does not need any user credentials." Due to the severity of the problem and trying to do the right thing, even with publishing the whitepaper, Secura said they had developed their own exploit that works reliably; but, given the risk, they weren't releasing it until they were confident Microsoft's patch had been widely installed on vulnerable servers.

So they did the right thing. They waited six weeks. Turns out that's not long enough. They warned, however, that it's not difficult to reverse engineer Microsoft's patch to develop an exploit. And sure enough, GitHub is now hosting at least three exploit scripts. I've got the links in the show notes for anyone who's interested.

Okay. So the technical side of this will be interesting to our listeners. As long as an attacker has the ability to establish a TCP connection to any unpatched domain controller, the attack can be launched. The vulnerability originates from the Windows implementation of the AES-CFB8 encryption. So we know that the use of the robust AES cipher in a cipher feedback mode is this CFB, Cipher Feedback, CFB.

We've talked a lot about cipher modes in the past. It's not enough to have a cipher like AES that can, under the influence of a key, map any combination of 128 bits into another unique combination of 128 bits. That's what it does. That's necessary, but not sufficient

to actually come up with a useful security protocol because, for example, under any given key, the same 128-bit input will always produce the same 128-bit output. This means that the cipher will be leaking a great deal of information when it's used to decrypt a lengthy message because, if any parts of the source message repeat, the encrypted message will repeat, if it's just 128-bit to 128-bit mapping. And, for example, if a message always begins the same way, the encrypted result will always be the same, which you don't want bad guys to know about.

So using a cipher algorithm in this way, in order to sort of put a wrapper around the cipher, is the way to go. If you didn't do that, if you just used something like a direct mapping, we call that ECB, Electronic Code Book mode. And it is for most applications not safe. We've solved these problems by using the cipher as the core of a larger encryption solution. Windows uses this AES-CFB for its mode.

So one of the ways to prevent the problem of the same plaintext message always producing the same encrypted message, is by introducing what's known as an "initialization vector." That's just a fancy term for deliberately introducing some varying material deliberately, like seeding the start of the encryption with something that changes so that something, anything is changing every time the encryption occurs, even if the message being encrypted does not change. And the result will be that the encrypted output always will change.

So what happened here in this case with AES-CFB8, the initialization vectors must be unique for each message. That's generally a requirement. Often, as we've talked about this before, they don't even have to be secret, and oftentimes they're not. But they have to be unique. It turns out that Windows failed to enforce that requirement upon its implementation of AES-CFB8, and the Zerologon vulnerability exploits this failure by sending Netlogon messages that include zeroes in various carefully chosen fields. The Secura write-up provides a very deep dive into the cause of the vulnerability and also offers a complete roadmap for how to exploit it, which is why shortly after it was published last week, proofs of concepts started popping up.

So it turns out that fixing the problem is a bit of a mess, and it's going to break some things because there are places where non-Windows devices are deliberately not using secure Remote Procedure Calls (RPCs). And Microsoft has determined that cannot be allowed to stand in the long term. So Microsoft will be updating Windows in two stages. The first stage is what happened in August, which they released last month. In addition to fixing this initialization vector bypass, this first update also prevents Windows Active Directory domain controllers from accepting any unsecured RPC communications from other Windows endpoints.

And so Leo, I think this is probably how SMB ties in. SMB might by default be using some unsecured RPC communications which are no longer permitted. And so as a consequence of August's change, only secured RPC communications are allowed. So you'd have to turn on, explicitly turn on SMB security in order to still be able to connect to Windows. So again, they know this is breaking some things.

**Leo:** Yeah.

**Steve:** And because there are non-Windows endpoints which will have a problem, Microsoft has introduced logging. It will continue to permit, but log any non-Windows devices that are authenticating themselves over unsecured RPC connections. And so by monitoring this log, network admins will be alerted to any devices they have which will need to be updated before next February because that's when the grace period comes to

an end, on Patch Tuesday, which will be February 9th of 2021, assuming that the Earth still exists. Microsoft will be...

**Leo:** Do you know something we don't?

**Steve:** Microsoft will be releasing a second update that will fully enforce the use of only secure RPC across the domain. Even then, admins will be allowed to make specific exceptions. But really, everybody is encouraged not to. So this is a two-step process. It's interesting that DHS felt that they were forced to elevate this to emergency directive. Nobody in the federal government has a choice. You must do this by last night. Last midnight. Which is six weeks downstream and two Patch Tuesdays from when this happened.

It just, I mean, I'm not on the front lines of IT in an organization like that. But there are lots of federal agencies, and maybe having any downtime is a problem. I mean, I can't guess why it would be that they wouldn't be applying patches. At the same time, how much time are we spending talking about all the problems that patches do create? If we talk about how IT, you know, they have to stage this. So if you've got a server that can't be down, certainly can't be broken by an update, I can see putting off the need. They probably read over these things and think, oh, well, that doesn't sound good. No, that doesn't sound good. Oh, that's not good. But if nothing is critical happening right now, we'll wait a little bit.

And so this is DHS saying, okay, no more. We have a tool, and that is part of what Secura created. I do not have the link in the show notes. So anybody who's interested can find it if you just put in "Secura," S-E-C-U-R-A, and "Zerologon," you'll go to their blog page, and you can find this tool that they have. There are also some links in some of the popular coverage of this on the 'Net. That would allow you to determine if there's a problem. The point is, our takeaway ultimately is that working exploits are on GitHub. All it takes is someone who's forgotten their anti-phishing training.

There was an article that I saw that didn't make it into today's podcast saying that some studies have shown that phishing training, in other words, do not click that link no matter how authentic it looks, fades after about six months. And so they're finding that it's necessary to refresh this training probably sooner than that, every four months, just to remind people that this is still a problem. I mean, I find myself having, when I come across something really bad, sending a note to Sue and Greg and say, I know you know this, but just really, really, really be careful, please, because it's necessary.

So they know this is going to end up getting exploited. It would be interesting to see, like, what effect this emergency order had. Hopefully our federal government is protected. Unfortunately, we know that that's a fraction of the number of domain controller systems in use in enterprises around the world. And if there's any similar sloppiness happening because people are saying, you know, they pause downloads until they're able to get around to it and verify that it's not going to cause problems, well, you may find yourself with a much bigger problem than installing an update that you could always back out of if it had a problem.

Our smartphone mobile OSes are acquiring new privacy and security features, both behind the scenes, which is always a good thing, and also more visibly. And they are, as a result, gradually becoming more secure. As I thought about the things we're going to talk about here, looking at useful user-facing features that Android and iOS have just added, Android 11 and iOS 14, it's easy to wonder why these features were not previously present. And I think the answer is that they do further complicate the user's

experience by requiring more attention and engagement. We'll talk about iOS new features next.

But for example, iOS 14, I mean, yeah, we'll talk about iOS in a second. But one thing I deliberately didn't mention then is an iOS feature which places a tiny orange dot at the top of the screen to indicate that the application has previously been given and currently has permission to use the smartphone's microphone and camera. It's supposed to be sort of a - look like a recording indicator. It's not red; it's orange. And given the original deliberately imposed simplicity of the Apple iPhone user interface, there was certainly a time when notifying the user of this would have been considered noisy and unnecessary. The point is, that's clearly no longer true. The goal and desire to keep it simple has not diminished. But what has changed, against the best interests of smartphone users, is the nature of today's threat landscape. It is far more active than ever. So smartphone users need to get all the help that they can get from their OS.

Okay. So to this end, we now have Android 11, which brings billions of its users, those who can upgrade and all those who purchase Android 11 devices in the future, more control over their security and privacy than they've had before. And yes, it's unfortunate that such control is necessary; but it's better to have it, I would say, than not at this point. And it's not like super intrusive anyway.

Okay. So what's new in Android 11? One-time permissions. Android 11 users get a useful feature that iOS has had for some time. This allows users to grant apps single-use access to their devices' more sensitive permissions, such as location, microphone, and camera. We're seeing variations of this on different platforms. For example, when a permission might be semi-permanent, users will be reminded after a while that such and such an app still has such and such a permission, and asked whether the app should retain it or not. Which is sort of a nice compromise between immediately removing it and sort of being kinder and gentler.

So Android 11 will now autonomously revoke permissions. In addition to having the single use, Android 11 has the ability to autonomously revoke permissions for long-unused apps. It could be a bit of annoyance, though those permissions can always be regranted. But this is sort of part of working to keep the casual user more safe while attempting to minimize the impact of enforcing that safety. And in general, we'll be talking here about this notion of minimum required permissions. That's always a good idea. We've talked about how firewalls flipped from once blocking known problem ports to blocking everything, and only opening the ports which are known to be needed to be open.

In what I think is a huge win for Android 11, the OS will also be getting incremental updates. Google has increased their plug-and-play store's integration on Android 11, allowing those devices to directly download and install critical OS security patches as modules, instantly, just like an app from Google Servers. We talked about the plan for this a long time ago, the idea that the OS would get modularized, and that Google would not have to force, you know, these would only be made available through the long OEM loop to a third party and hopefully to the end user, but rather be delivered as part of the app store, much more like the Apple model has always been. And that's now here.

So, boy, I think that's going to be a huge welcome improvement. And it would sure be nice for everybody to be able to get 11. I'm not sure how far back 11 will be made available to earlier devices. But once you get there, you have a chance of being kept current. Which is fabulous.

Also it tightens up an app's right to obtain location information when it's in the foreground, versus moving in the background. We'll talk about iOS, which has done some interesting things there, as well. When an app requests permission to access the user's

location, Android 11 initially only grants foreground location permission, that is, while the app is in use actively. If the app additionally desires access to location information while it's in the background, such apps will now be required to produce a separate and independent permission request. It's not all just lumped together into location services. And granting the permission is not as simple as quickly clicking yeah, yeah, okay, whatever. No.

Now, to enable background location access, users must go to a settings page and set an "allow all the time" option for the app's location permissions. And even to obtain such a permission setting, Google will now be requiring their app developers to explain why their app needs background location access at all. Just wanting it just because, that's no longer going to be enough in order to get it. So a bunch of nice improvements in Android 11. It's clear they're all going to help improve user security, especially the from-the-store OS updates, which I just think that's going to make a world of difference for so many and for the security of so many Android users. Yes, there's more for the user to do. It requires more involvement from the user. But I think that's necessary as we go forward.

And as for iOS 14, the list of iOS updates and improvements, I scanned through it. It is overwhelming. And being an iOS user myself, I'm beginning to feel more and more as though I barely know how to use the device in my pocket. Oh, Leo. That's just crazy. I saw you jumping around with...

**Leo:** Oh, those widgets, yeah.

**Steve:** Oh, my. Customizing things and locking things in place. And it's like, okay. I mean, really it's not the phone that Steve Jobs gave us all those years ago.

**Leo:** No. The nice thing is you don't actually have to do any of that stuff. And the most important improvements are what you already talked about, the same with Android, which is it now pops up warnings and lets you know what's going on. And you'll get that no matter what. So I think they're kind of on the right track.

**Steve:** Yeah, yeah. And my hope is that these things are discoverable so that it's kind of like, well, you know, will it do this? Oh, look, it does.

**Leo:** It does, yeah, exactly.

**Steve:** And then you are able to do it. Yeah, I immediately pinned a few of my most used iMessage contacts.

**Leo:** Yes, in the messages is nice. Yes, I do that.

**Steve:** It's a little jarring. I'm not used to it yet. It kind of, you know, every time I bring up iMessage it's like, whoa, okay.

**Leo:** Yeah, got these heads; right? Yeah.

**Steve:** Yup, exactly. Exactly.

**Leo:** And when somebody talks to you on those heads, you'll have a little pop-up there, which is kind of fun, yeah. I have to add you to it, though. I'm going to put you in there. You've been texting me a lot lately.

**Steve:** There's been more going on, yeah. That's true. Okay.

**Leo:** We're commiserating together.

**Steve:** Yes. With iOS 14 they did something that I just think is brilliant. They have restricted access to the LAN. Apple noticed that while we're at home, very few of our apps have any need to access any devices on our home network. All they need is to see our broader router and the wider Internet. I'm sorry, our border router and the wider Internet. And of course in this era of IoT local network access, some apps certainly do have a need for local LAN access. But it is typically limited to one or two or a small number of explicit IoT-interfacing apps. I think this is just brilliant that they noticed it.

**Leo:** It's interesting. Because Apple has a WiFi backup which they prefer you use to iCloud for your phone. But it's not using - it's not backing up to, I mean, even on your computer you can have WiFi backup, but I bet you it's going through the cloud and down again.

**Steve:** It might be. Or since the OS is in control, they're able to give [crosstalk] themselves.

**Leo:** Right, right.

**Steve:** Yeah, if they wanted to. So here's what they realized that I think is so cool. The vast majority of our phones' apps are entirely ignorant of what's going on on our home or our corporate LAN. And the wisdom of always and only granting minimal rights teaches that by default no apps should have local LAN access. They don't need it. And only those apps that actually require it should have it granted to them.

So what that means is, if an app is malicious, if it gets onto the App Store, if it gets by Apple scrutiny, if it gets updated and then has some behavior that would otherwise have allowed it to get onto your LAN and be a far more devastating consequence, iOS just says no. That app won't have access unless there's a reason for it to. So yeah, a few will. But I just think that this idea of by default restricting apps to only the Internet is a tiny brilliant perfect solution for increasing the security of our systems.

Also, back in 2014, when iOS 8 added, I guess you'd call it unassociated MAC address randomization, we talked about this six years ago. The idea was that it was an anti-tracking measure. Normally a WiFi radio has an Ethernet MAC address. As we know, it's a 48-bit address. It's the pair of hex digits separated by colons which uniquely identify that piece of hardware, the intention is globally, 48 bits. It used to be fixed. Then people started worrying that, well, wait a minute. My phone is broadcasting my hardware radio Ethernet MAC address all the time, wherever I go, making it trivial to track me by my MAC address. So Apple said, okay, we're going to compromise here. As long as your

phone has not associated with an access point, we're going to broadcast a random MAC because, after all, who cares? It doesn't matter what it is. But when you associate with an access point, then we will revert to the actual MAC for the device.

iOS 14 has taken that a step further. If you look under WiFi, Leo, if you haven't already, in your settings under the WiFi your phone, your iOS device is currently associated with, there is a new switch turned on by default, private address. What that does is it synthesizes a unique MAC per access point. Because the one thing that was still possible before was that when you associated with a MAC, then you would be giving that access point your actual MAC.

**Leo:** He's back, yeah, yeah.

**Steve:** Yes. So you were still trackable to the degree that you associated with access points. Now, every different access point you associate with, by default, iOS 14 will synthesize a unique MAC for that access point. So it's able to identify you uniquely, but you are nontrackable. Actually, it's exactly the way SQRL works with SQRL identities; right? Every time you go back to that same website, SQRL identifies you as a unique identity. Yet every different site you go to gets a different identity.

**Leo:** Right.

**Steve:** So iOS has done that with our MAC addresses on our iPhones.

**Leo:** Does this confuse MAC address filtering?

**Steve:** Well, that's probably why you can turn it off is that it might be, for example, in an enterprise setting they're doing MAC address filtering. Now, as we know, MAC address filtering is weak because the MAC address is in public; right?

**Leo:** It's broadcast, yeah.

**Steve:** Yeah. That's why we've never...

**Leo:** It's easy to spoof.

**Steve:** Yes, we've never promoted it as anything really useful because a bad guy can see what MAC addresses are on the network being accepted and just adopt one of them on the fly. So not much security there. But so my guess is maybe in an enterprise you might need to turn it off if, for example, privileges were granted to you as you roamed among access points in an enterprise. There you might want to have it turned off. But it is switchable on a per-network mode. So again, they did the right thing. It's defaulted on all the time. So everybody automatically gets this protection. Only if it causes a problem might you need to turn it off. And I imagine that the provisioning system that enterprises have with iOS devices would allow them to do that just automatically, too. So very, very cool. Again, a nice piece of new technology from Apple.

As is what I call "fuzzy location services." With GPS in our phones, of course, the phone knows precisely where it is, even which direction it's facing at all times. So it's possible for apps to be granted that same awareness, as well. But as with LAN access, not all apps have the actual need for that level of location precision. Exact location can compromise privacy. And by the way, Leo, this is what I was trying to remember. It turns out there are ad networks which obtain the location of the user as part of their tracking. So talk about another breach of privacy aspect.

So what Apple realized is that, very much as with LAN access, not all apps have the actual need for that level of location precision. Exact location can compromise privacy. But there are some that do have a need. For example, whereas a driving navigation app does need to know precisely where you are, a weather app only needs to know your approximate location, within several miles of where you are, to do its job. So again, they follow this concept of not disclosing anything more than is actually necessary. And fuzzing your location for apps that don't need pinpoint location is now part of iOS 14. And again, bravo.

Couple more things. Of course the invention of a shared system-wide copy-and-paste clipboard was one of the many brilliant innovations to arise from the user interface work that was done at Xerox PARC for their Alto system. But as we know, with great power comes great responsibility. And the irresponsible or malicious capture of our instantaneous clipboard content has been a longstanding security problem. We've talked about it often. And believe it or not, in that continuing quest to track us, there are also ad networks used in apps that continually spy on our clipboard contents, as yet another means of locking onto and following us wherever we go. Just, again, they've been very clever, to their credit, but really at a cost of real intrusion of privacy.

iOS 14 adds a notification every time an app accesses the clipboard. We talked about this as a coming feature some months ago, and in fact remember that it had outed a number of apps that were doing this constantly and caused their developers some embarrassment. And so at the time I said I think this is just the greatest thing to ever happen.

**Leo:** Including TikTok, by the way.

**Steve:** Yes. Yeah. It's like, okay, why are you looking so much? Yeah. And because some apps do not have a legitimate need, at the developer level Apple has included new APIs to allow apps to quietly ask iOS what sort of contents the clipboard contains without needing to obtain it and thus trigger a notification. In other words, so there are apps that do have some reason for wanting to look at the clipboard. But if they simply do a copy, it will pop up this notification. And as often as not, the information there is not for them.

So what Apple has added in order to quiet this otherwise rather noisy, at least at first, noisy new but really cool feature, they've added the ability for an app to ask for the type of information on the clipboard. If it's not a URL, for example, for an app that only needs URLs, it's able to ask, but not pop something up, and vice versa, and all kinds of other stuff that you might be putting on your clipboard. So again, Apple was clever in trying to keep the visual noise down. But boy, a tip of the hat for the fact that they are going to be popping up a notification whenever that happens. Again, a perfect example of the kind of thing that on the iPhone's launch so many years ago would have just made no sense.

**Leo:** They didn't even have cut and paste in the original iPhone.

**Steve:** Is that true? There was no inter-app?

**Leo:** Yeah, there was no clipboard. No.

**Steve:** Whoa. Wow.

**Leo:** That was actually a huge missing feature that people complained about for months. They eventually added it.

**Steve:** And in fact there were no apps in the beginning.

**Leo:** There were no apps.

**Steve:** It was just a half-filled home screen that looked like, well, wait a minute; you know?

**Leo:** And Steve said at the time, and actually it was prescient, you could use a web page for most apps. And now you could with progressive web apps. But Apple doesn't really support it very well. So everywhere but Apple. It's come around full circle.

**Steve:** So two last little bits of Safari improvement. As is vogue in these days, Safari now has also gained the ability to check whether any of the passwords it has saved for its users may have appeared in any known password breach data dumps. And of course it does this safely, without revealing the password collisions to anyone but its user, who might then wish to change their publicly disclosed password, now knowing that it is present in some data breach. One of the things that we know the brute forcers are doing is they take those lists as their seed material. Before they start doing algorithmic brute forcing, they try all of the known passwords, knowing that people tend to reuse them. And even if it wasn't reused, maybe the breach was in a site where you're trying to log on.

So anyway, Safari now has password breach notification built in. And there's a website privacy report. I tried it, but mine hadn't been used enough yet in order to have acquired any information. It told me, yeah, we don't know enough yet. But you're able to tap a privacy report button to view all the cross-site trackers that are being blocked by Safari's intelligent tracking prevention system to get some sense for the work it's doing to help you and also, on the flipside, which sites you're going to are exhausting themselves trying to support tracking.

So overall, both Android 11 and iOS 14 have given us a bunch of welcome improvements, both behind the scenes and also some things that are going to be coming to the attention of their users as necessary. And, boy, I just tip my hat to the cleverness that iOS has offered in terms of just restricting the presence of unneeded information of all kinds. They've done a great job.

**Leo:** Yeah, increasingly I've moved towards the iPhone platform. I love Android for its customizability, but it's nice to know that you're relatively secure on your iPhone.

**Steve:** And it sure does seem like iOS is catching up on that customizability side.

**Leo:** Oh, yeah, absolutely. With widgets, that's a big improvement, yeah. Yeah, I think they're getting closer and closer together, both of them. I mean, Android's done, as you pointed out, a lot of great privacy stuff, too. And now, let's talk errata.

**Steve:** So as the sun has set...

**Leo:** You're getting darker and darker, yeah.

**Steve:** Beginning to look like a disembodied head.

**Leo:** I love it.

**Steve:** Okay.

**Leo:** That's fine. That's good. I like it.

**Steve:** Put up with it for once. Okay. We have a bit of errata. Thanks to some feedback from some knowledgeable folks hanging out in GRC's Security Now! newsgroup, I learned that although it's clearly wrong for Windows to be issuing TRIM commands to hard disk drives that do not accept them, and thus are logging errors, there are drives, strangely enough, which employ a storage density increasing technology known as Shingled Magnetic Recording (SMR).

**Leo:** Oh, yeah, SMR. Oh, yeah. Oh, yeah.

**Steve:** Yup. This Shingled Magnetic Recording can increase the storage density of a shingled drive by around 25%. But the technology is tricky because, just as with an SSD, where a partial update is time-consuming and expensive because like a whole region needs to be rewritten at once, similarly these SMR drives can also benefit from being affirmatively notified by the operating system when regions of their storage no longer contain active file system data, and therefore do not need to be rewritten as part of a nearby update. In other words, there's a use for TRIM commands on hard disk drives, not just on solid-state SSDs.

So it's not entirely, well, it is entirely nuts for Windows 10 to be issuing TRIM commands to drives that say I don't know what you're talking about and to continue to do so. But there are some spinning drives where it does make sense. So anyway, I wanted to say I stand corrected on the idea that no spinning drive can make use of TRIM. It turns out there is an application for it.

Someone whose Twitter handle is unfortunately, well, it looks like it's @mehtechie, M-E-H-T-E-C-H-I-E, he had a good experience that I just wanted to share and reinforce it for anyone who might have forgotten this. He tweeted: "I just wanted to say thank you so much for InitDisk. Just rescued a previously thought completely dead 2TB Crucial SSD."

And so again, that's one of the things that we discovered while we were developing InitDisk. Remember that InitDisk was the work I did to create a new thumb drive formatting and prep tool which will be part of the next SpinRite. Actually, it's going to be part of the forthcoming benchmark test also because it'll init a thumb drive to boot DOS, and that will also have the benchmark on it.

But what we found was that there were a number of instances where people had thumb drives that just no longer worked. They just - nothing worked on them. InitDisk brought them back to life. And in this case a 2TB Crucial SSD, that's a nice piece of hardware, so it's cool that it did that. So I just wanted to share that.

And GRC's new forum system is up and running, and SQRL is working perfectly there. All of those final edge cases that I referred to last week appear to have been resolved and eliminated. So I'm pulling the final benchmark together. I've named it ReadSpeed, since that's what it is. We'll be doing a pre-release round of testing in the newsgroup first. Then I'll have something soon for the wider world to play with, as well as a readily accessible place for ReadSpeed's users to hang out and interact. So some fun stuff coming soon.

Okay. Formal verification. It's clearly where we're headed. At this point I would say that the technology is still emerging and maturing, that is, the technology required to create a language or a description process which - and it's kind of AI-ish - which allows you to describe to a computer-based system at a detail level how a security protocol is designed, tell it what characteristics you want that protocol to have, and then ask it did we achieve our goal with this design.

And I say that it's clearly still emerging and must be maturing because we're still seeing too many complex protocols being developed just in an ad hoc manner. Yeah, a bunch of smart guys get around, sketch something out, hopefully not on the back of a napkin, maybe on a whiteboard, and say yeah, that looks about right. What do you think? And they all agree, and somebody writes it up, and now we have WiFi or Bluetooth or whatever. And what we also unfortunately keep having are problems with those protocols.

There's two types of problems. There's implementation problems, which would be somebody read the spec and didn't code it correctly. So for example, any buffer overrun would be an implementation problem. Nothing wrong with the specification. The spec doesn't talk about where buffers are. It talks about how large they have to be. But implementing it in an insecure way, that's the implementer's fault, not the spec writer's fault. It's the spec that says this is how the system should work. So there's implementation separate from specification.

Okay. So let's first step back and take a look at another new vulnerability. We keep talking about these. And what's interesting is it was found. It was previously unknown to the world. It was found when the seven researchers at Purdue were wanting to check out Bluetooth Low Energy protocol and succeeded in using an automated verifier to do that.

Okay. So once again we have alarming headlines: "Billions of Devices Vulnerable to New BLESA (B-L-E-S-A) Bluetooth Security Flaw." Another headline read: "Bluetooth Spoofing Bug Affects Billions of IoT Devices." And so yes, here we are again, week after week. The research into this was conducted by a bunch of guys, as I said, at Purdue; and their research is titled "BLESA" - and that's Bluetooth Low Energy Spoofing Attacks, B-L-E-S-A. Anyway, the title is "Spoofing Attacks Against Reconnections in Bluetooth Low Energy." And so I'll share the abstract from their paper.

They said: "The Bluetooth Low Energy protocol ubiquitously enables energy-efficient wireless communication among resource-constrained devices. To ease its adoption, BLE

requires limited or no user interaction to establish a connection between two devices. Unfortunately, this simplicity is the root cause of several security issues. In this paper, we analyze the security of the BLE link layer, focusing on the scenario in which two previously connected devices reconnect. Based on a formal analysis" - that's the key - "a formal analysis of the reconnection procedure defined by the BLE specification, we highlight two critical security weaknesses in the spec. As a result, even a device implementing the BLE protocol correctly may be vulnerable to spoofing attacks.

"To demonstrate these design weaknesses and further study their security implications, we develop BLE spoofing attacks. These attacks enable an attacker to impersonate a Bluetooth Low Energy device and to provide spoofed data to another previously paired device. BLESA can be easily carried out against some implementations of the BLE protocol, such as the one used in Linux. Additionally, for the BLE stack implementations used by Android and iOS, we found a logic bug enabling BLESA. We reported these security issues to the affected parties, Google and Apple, and they acknowledged our findings."

Okay. Then these guys share some important and useful background. They explain Bluetooth Low Energy is the most widely used low-energy communications protocol; and, by 2023, the number of BLE-capable devices is expected to reach five billion. The BLE protocol - so anyway, the good news is it's not too late to get to the ones that haven't been created yet. The BLE protocol enables wireless short-range communication which allows two devices to connect and exchange data.

A typical usage scenario consists of a smartphone and a gadget device such as a fitness tracker, for example, that communicate over BLE. Every BLE connection involves a device acting as a client, in this example the smartphone, and another device acting as a server, in this example the fitness tracker. The first time two devices connect, allowing them to exchange data, they perform a specific pairing procedure which varies depending on the type of the connected devices and their user-interfaces' capabilities. And of course we've covered all manner of these in the past.

They said the major factors that have helped the rapid growth of the adoption of BLE-enabled devices are its low cost and the minimal setup effort required for end users. Unfortunately, previous research has shown that these user-friendly features have a negative impact on the security of this protocol. Right? The famous tradeoff of convenience versus security. This concern is particularly worrisome since Bluetooth Low Energy communication is often used in security-sensitive devices such as physical safety devices like locks, or health monitoring devices like medical implants.

Researchers have pointed out and studied many implementation weaknesses in the BLE protocol by manually analyzing its specification. Manually analyzing. Additionally, some previous work has performed a formal automated analysis of the BLE specification. However, these formal approaches only focused on limited aspects of the entire protocol, such as the BLE pairing mechanism. This limitation is due to the fact that it is challenging to fully formalize and automatically analyze the Bluetooth Low Energy specification. The challenges stem from the complexity of the Bluetooth Low Energy protocol, difficulties in modeling its multilayer design; right? I mean, you don't know where the problem's going to be, so you've got to model the whole thing - and its multiple variants and optional features which allow this protocol to support a wide range of devices with significant different capabilities.

"In this paper we study a previously unexplored mechanism of the Bluetooth Low Energy protocol. Specifically, we focus on the mechanism of dealing with reconnecting two previously connected devices. This mechanism comes into play when, for instance, a server device, the fitness tracker, moves out of the BLE wireless communication range of

a connected client, the smartphone. And then at a later time the two devices get close enough to reestablish a connection."

Okay. So what we have now is a new awareness of a critical exploitable flaw in the reconnection of all the Bluetooth Low Energy devices we're using right now. I mean, this exists today. It's not theoretical. As we know, reconnections occur when the Bluetooth devices move out of range and then back into range. So normally when reconnecting, the two Bluetooth Low Energy devices should check each other's cryptographic keys which were negotiated during the original pairing process and reconnect and continue the exchange of their data.

But what the Purdue team found was that the official Bluetooth Low Energy spec did not contain sufficiently precise and strong enough language to describe the reconnection process. So there's a little bit of sloppiness. As a result, two systemic issues have made their way into Bluetooth Low Energy software implementations and down the hardware supply chain to us. First, the authentication during the device reconnect is optional instead of mandatory, if you can believe that. It's like, what? The authentication during the device reconnection is optional instead of mandatory.

The second problem, the authentication can potentially be circumvented if the user's device fails to enforce the IoT device to authenticate the communicated data. Again, the authentication can potentially be circumvented if the user's device fails to enforce the IoT device's authentication of communicated data. These two issues leave the door open to a BLESA attack, during which a nearby attacker is able to bypass reconnection verifications to send spoofed data to a Bluetooth Low Energy device.

So it turns out that, despite the somewhat vague language that the Purdue guys found in the spec, the problem is not present in all BLE real-world implementations. The researchers analyzed multiple software in the real world used to support Bluetooth Low Energy on various OSes. They found that BlueZ, which is used in Linux-based IoT devices, and Fluoride, which is used for Android, and the iOS Bluetooth Low Energy stack were all vulnerable to BLESA attacks, while the Low Energy stack in Windows devices was not. It was immune to these.

In their published paper last month, they wrote: "As of June 2020, while Apple has assigned CVE-2020-9770 to the vulnerability and fixed it, the Android Bluetooth Low Energy implementation in our tested device," they said, "a Google Pixel XL running Android 10" - so we don't know about 11, which as we know just is out. In Android 10 it was still vulnerable. "And as for Linux-based IoT devices, the BlueZ development team said they would deprecate the part of their code that opens devices to BLESA attacks and instead use code that implements proper BLE reconnection procedures which are immune to BLESA."

So we have yet another Bluetooth Low Energy attack. In this instance it's necessary to be present during the attempted reconnect. So not something that is going to be a common problem for us. You can imagine some NSA little box stuck near a lock somewhere, and it's observing and interfering with a reconnection. When somebody brings their smartphone near the lock, they see each other. At that moment there's a BLESA vulnerability, and they can take advantage of it, that sort of thing.

But we shouldn't be allowing that in Bluetooth Low Energy. It should absolutely be the case that authentication is reinforced and always enforced after that initial pairing. And what these guys found, thanks to them having formally described the protocol to a verifier, this thing managed to say, hey, you know, look over here in the spec because, if this isn't done right, you're going to have a problem. They looked, and that's what they found.

**Leo:** That's really interesting. So these verifiers could be very useful tools. That's really interesting.

**Steve:** Yes. I imagine that in the future we'll be looking back on the day when we were just doing sort of seat-of-the-pants protocols as, like, what were they thinking?

**Leo:** Right.

**Steve:** Because the protocols, today's protocols are complex. And it's just too easy to miss some little interaction that makes it into the spec and then, as these guys said, all the way down the supply chain to us. It's just it's getting too complicated for us. We need computers to help design computers.

**Leo:** Really interesting, yeah.

**Steve:** And then the network in the sky took over, and Skynet said thank you very much for turning over control of the protocol.

**Leo:** We'll take it from here.

**Steve:** Yeah, we got you, we got you.

**Leo:** You won't mind if we leave a few little holes in there for us. Yeah. So I'm actually fascinated by the whole notion of correctness in software, verifiable correctness in software. And this is related to that, which is basically correctness in protocols. And, now, do you have to - you have to state the protocol in some sort of highly defined language.

**Steve:** Descriptive language, yeah.

**Leo:** Descriptive language.

**Steve:** So what researchers have found is that it is easy to do this at a toy level, you know, very simple protocols, very simple language. But as soon as the protocol starts getting complex, the complexity of automated, fully automated proofs becomes incredibly difficult.

**Leo:** Right.

**Steve:** It goes exponential.

**Leo:** Right. You need some rigor.

**Steve:** It's like adding bits to a binary number and seeing how the number of possibilities goes up with every bit.

**Leo:** Right.

**Steve:** Well, you're adding, like, philosophies instead of bits.

**Leo:** That's why it's hard to do, yeah, yeah, yeah.

**Steve:** Oh, yes.

**Leo:** Interesting. Well, my friend, we have come to the end of another emergency in Security Now!.

**Steve:** First offsite emergency podcast.

**Leo:** It's the first time you've done that.

**Steve:** As the lights dim. Yeah, in 15 years I've not had to go somewhere else. Fortunately, I have a somewhere else now.

**Leo:** Yay. Yeah, in past years you might have been doing this from Starbucks.

**Steve:** And you can't do that today because...

**Leo:** No, nobody there. Nobody there. 784 episodes, and finally we had to go offsite. That's pretty good. I think you can keep that record up. Thank you, Steve Gibson. You'll find him at his site, GRC.com, the Gibson Research Corporation. That's where SpinRite lives. That's the world's best hard drive recovery and maintenance utility. It's getting better all the time. Getting close now to v6.1, but you'll be picking up 6.0. But don't worry about that, you'll get a free upgrade. Plus you'll be able to participate in the development of 6.1, so that's nice.

**Steve:** Pre-release, yes.

**Leo:** Yeah, yeah. You can also get the podcast there. He's got 16Kb audio for the bandwidth impaired, 64Kb for those with ears. And for those with eyes, very nicely written, human-transcribed transcripts so that you can read along as you listen. A lot of people find that a better way to understand. And of course it has the real benefit of it being searchable, so you can quickly find what you're looking for. All the transcripts are online and searchable, which means if there's something Steve talked about in the last 15 years, you should be able to find it with no problem at his site: GRC.com. Lots of other great free stuff there. You can leave feedback for him, questions or suggestions, at GRC.com/feedback. He's also on Twitter, @SGgrc, and

he accepts - his DMs are open, as the kids say. So you can always leave questions, thoughts, comments: @SGgrc.

We are at TWiT.tv. That's the website to go to if you want to watch us do this show live of a Tuesday afternoon. TWiT.tv/live is the feed. That's running 24/7, audio and video. When we're doing live shows you can watch behind the scenes. When we're not, you can watch recordings of behind the scenes, which for many is just as good as the original behind the scenes. On-demand versions of the show make it easy to listen at your convenience. Just go to TWiT.tv/sn for audio and video, or watch on YouTube, or ask your Amazon Echo for Security Now!.

I have a sad note. A number of people have written to me saying we can't listen to Security Now! on TuneIn in the U.K. anymore. We checked into that. And apparently due to a lawsuit by Sony and Warner Media, TuneIn has decided to pull all non-U.K. content from TuneIn in the U.K., and that included all of our shows. So if you were a listener via TuneIn Live, which was a great thing, you could say "Play Security Now!" on TuneIn and you'd get the latest version, I apologize. Nothing we can do about that. But there are other providers, so there's other places to go. Basically we try to be everywhere we possibly can so that you can get us no matter what.

Steve, we're doing - what are we doing for the holidays? Are we going to do a best-of? Do you want to do a best-of this year?

**Steve:** Yeah, it would be like last year, a best-of.

**Leo:** Okay. Yeah, we're collecting them.

**Steve:** You and I are going to have a meeting on Friday, and then...

**Leo:** Yes. We've got a panel event next week. Is it next week, or two weeks? Two weeks from now. It's going to be good.

**Steve:** Two weeks, but we have a pre-meeting on Friday.

**Leo:** Yeah, we'll get together. That we will not broadcast. But that's going to be a fun one because we're doing an ITProTV event.

**Steve:** Right.

**Leo:** It's "Cyber Hangover: What the IT World's Going to Look Like Post-COVID." I think that's a great topic. Amy Webb is going to join us. I don't know if you've met her, but you'll love her, super smart futurist. Don Pezet from ITProTV. I'll be there. That's going to be a lot of fun. So that's coming up October 1st. Actually October is security month, so we have a couple events planned for October. LastPass is doing another event, as well.

**Steve:** Very cool.

**Leo:** So we're going to have a lot of fun over the next few weeks. So I was going to mention best-ofs. If you want to submit a best-of, TWiT.tv/bestof. And you can leave your thoughts about a great moment that happened in the last year.

**Steve:** Yay.

**Leo:** Steve, have a great week, and we'll see you next time on Security Now!.

**Steve:** Okay, buddy. Hopefully back in my regular lair, I'm sure by then.

**Leo:** We miss the blinking lights.

**Steve:** Bye.

**Leo:** Bye-bye.