
# Device Bound Session Credentials

**Description:** Android to get "Lockdown Mode." What's in the new editions of Chrome and Firefox? Why did Apple silently re-enable automatic updates? My new iPhone 16, Chinese tariffs and electronics. Dynamic "hotpatching" coming to Win11 Enterprise & Edu. Why is it so difficult for Oracle to 'fess up? Another multiyear breach inside U.S. Treasury. An Apple vs. the UK update. "Thundermail." (Can't someone come up with a better name?) The (in)Security of Programmable Logic Controllers. When LLMs write code and hallucinate nonexistent packages. WordPress core security, and PHP gets an important audit. Device Bound Session Credentials update session cookie technology.

High quality (64 kbps) mp3 audio file URL: http://media.GRC.com/sn/SN-1021.mp3
Quarter size (16 kbps) mp3 audio file URL: http://media.GRC.com/sn/sn-1021-lq.mp3

SHOW TEASE: It's time for Security Now!. Steve Gibson is here. We're going to talk about, well, there's a lot of things. The 100-some fixes in Microsoft's Patch Tuesday last week. Why it's so difficult for Oracle to 'fess up. An Apple vs. the U.K. update, and the arrival of Thundermail. All that and more coming up next on Security Now!.

**Leo Laporte:** This is Security Now! with Steve Gibson, Episode 1021, recorded Tuesday, April 15th, 2025: Device Bound Session Credentials.

It's time for the moment you wait for all week long, Security Now!, the show where we cover your security, privacy, your safety online with the king of all of that stuff, Mr. Steve "Tiberius" Gibson. Hello, Steve.

**Steve Gibson:** Actually, Leo, what they're waiting for all week long is the next protracted event in their life, typically a five-hour commute or a plane flight or something.

**Leo:** Oh, yes, something they can listen to this show and...

**Steve:** Yes, because now it's in their queue, and it's time to spool this into their brain. And, boy...

**Leo:** Think of us...

**Steve:** ...have we got a spool for you today.

**Leo:** Think of us as a pressure driver, spooling up all this information for you to unwind.

**Steve:** Yes, even the title needed to be spooled because it...

**Leo:** It's a little...

**Steve:** It stretched out the screen there down at the bottom.

**Leo:** Yeah. What is Device Bound Session Credentials after all?

**Steve:** Okay. So for Security Now! Episode 1021 for Tax Day - and by the way, I heard you saying before you could be listening to MacBreak Weekly while doing your taxes.

**Leo:** Yes.

**Steve:** And I thought, only if you have a time machine, and once they're finished, you can go back. Unless you're filing estimated, in which case...

**Leo:** You could be listening right now and doing your taxes.

**Steve:** Oh, that's true. And then get in line...

**Leo:** You don't have to mail them till midnight. You've got time.

**Steve:** Yeah, that's true.

**Leo:** And maybe, it's just me, but I always, I mean, I did mine yesterday. I was way ahead of the game; okay?

**Steve:** That's right. Yeah. Well, that's - you had the day off. No podcasts to do.

**Leo:** I think it's - you know what's funny, I actually listened to an old TWiT while I was doing it. I don't know why, I just - Dan Patterson sent me an email saying the first time - he was on TWiT on Sunday - "The first time I was on was back in 2009," this episode. And it had - it was a great episode, had all these great people. And I thought, I'll listen to it. And it was kind of fun to hear about the beginnings of surveillance capitalism.

**Steve:** How was Sunday's big anniversary TWiT? I haven't had a chance to...

**Leo:** Oh, Steve, we had so much fun.

**Steve:** Aww.

**Leo:** Because I had more than 20 videos from listeners and viewers talking about when they first found TWiT. It was very - it was wonderful to celebrate the audience. You know this because you get the emails and the comments.

**Steve:** Yeah. I have a good connection.

**Leo:** We love our audience.

**Steve:** Yeah.

**Leo:** We really do. And so I thought, to celebrate 20 years of TWiT, you're going to be celebrating 20 years of Security Now! in a few months, it'd be fun to, instead of honoring the hosts or the things we've done, to hear from the listeners. And it was really great. I really enjoyed it. We had a fire eater. We had a guy on a boat. We had, not one, but two guys in tractors. I mean, it was a very interesting...

**Steve:** Oh, I thought you were going to say "traction." It's like, whoa.

**Leo:** No, nobody was in traction. There was one person incarcerated, however. One prisoner sent us an email.

**Steve:** He got good bandwidth connection? Can he...

**Leo:** He, and I didn't know...

**Steve:** Because those steel bars, they tend to block WiFi. That's not...

**Leo:** No, no, no. They give him an iPad with podcasts on it. And I don't know if we were ever asked, but apparently Security Now! is on some of them. This one he only was able to get one of the shows, TWiT.

**Steve:** I don't know if you want Security Now! going into the prison, Leo.

**Leo:** Maybe that's - maybe the warden says...

**Steve:** How do I do that hack again? How do I get over to Russia? How do I call a strike?

**Leo:** You know what, good point.

**Steve:** Anyway, so, oh, boy. This is going to frost your snow cone, Device Bound Session Credentials. What we finally have, after 35 years, is a change in the way we manage session cookies, session cookies being the cookies which our browsers receive which continually identify us to websites that we're logged into, the session being our logged-on session. And I'll go back over a little bit of the history of this when we get to it later today, sometime this evening, because we have a lot to cover today.

**Leo:** It's going to be a long show? Is that what you're telling me? I'd better go get lunch. Okay.

**Steve:** But, yeah. You can maybe plan your vacation. So...

**Leo:** I'll do my taxes for 2025.

**Steve:** But don't take it yet because, yes, you do have to lick the stamp on your taxes.

**Leo:** Yes, I do.

**Steve:** Anyway, we're going to talk about the industry finally coming up with what looks like the replacement for and far more security connected to maintaining logged-on state with browsers. And really, we've been asking an awful lot of the lowly cookie which was created, you know, as I said, back in the mid-'90s by some guy named Lou at Netscape. Anyway...

**Leo:** I didn't know his name. That's funny.

**Steve:** Oh, yeah, Lou. We're going to have a lot of fun.

**Leo:** Lou's cookies.

**Steve:** And I'll explain that you do not need to understand it all on this first pass. We're going to be - no doubt we'll be looping back to this a number of times because this is big news. This is a change in the way we, like, the security of logging on in terms of the browser identifying it to the server. And it is very cool. I mean, like we have so much in our toolbox now with all of the crypto that we're able to bring to bear, rather than some little gibberish of ASCII that is like, oh, that looks like him. Anyway, we're also going to, before we get to that, because that's just the coup de grace, we've got Android believed to be getting a lockdown mode next month. What's new in updates to Chrome and Firefox, and there is some cool stuff. Actually, it was the blurb about Chrome that put me onto this. And then I saw that Firefox and Safari were also already working on this.

Why did Apple silently reenable automatic updates? My new iPhone 16, Chinese tariffs, and electronics. Dynamic hotpatching coming to Windows 11 Enterprise and Edu. Cool new tech from Microsoft. Why is it so difficult for Oracle to 'fess up to what is obvious to everybody else that happened? We have another multiyear breach uncovered inside U.S. Treasury, making it the third of three. An Apple vs. the U.K. update. Something I just - I can't get over the name that they've given this, Thundermail. And can't we get a better name?

**Leo:** Thundermail.

**Steve:** You know, it works for a bird to put "thunder" in front of it. It's like, you know, thunder...

**Leo:** It sounds like a male strip show, like Thunder Male Down Under.

**Steve:** It just, every time I see it, I go, oh, god. I'd be embarrassed to be steve@thundermail.com. Anyway, Mozilla's going to do something. We also have the insecurity of programmable logic controllers, why that matters. Oh, and Leo, it turns out that - you probably ran across this because you're amazingly up to date and informed, I find - when LLMs write code and hallucinate non-existent package names...

**Leo:** I know the perfect library for this code, yeah.

**Steve:** ...it's going to be weaponized, yes.

**Leo:** Yeah. I guess so, yes, if you know ahead of time, yeah.

**Steve:** So, and it's got even a worse name than Thundermail, it's slurp something or other. It's like, oh, my god. Well, anyway, we'll be covering it today. We also have WordPress's core security. And PHP had a very important audit funded, and the problems they found are barricaded. No one is talking about them because they're so bad.

**Leo:** Uh-oh. So bad, yeah.

**Steve:** So, but they're being fixed. And I think what we're going to end up seeing, as we'll see, is an important retroactive, you know, everybody who still has supported versions of PHP, uh, now would be a good time to update them. Also, once all that's done, if there's anything left of us, we're going to talk about Device Bound Session Credentials. And I so much want to hyphenate Device Bound. It's not. And it's like, that's wrong. But, you know, we put up with referer being misspelled in HTTP headers all our lives.

**Leo:** That's right, one "R," yeah.

**Steve:** So I suppose we'll leave off the hyphen in Device Bound. Oh, and of course we've got a great picture. So maybe, Leo, we actually have a good podcast this week.

**Leo:** Maybe?

**Steve:** I hope we don't disappoint.

**Leo:** No maybe about it, Steve. I guarantee, I guarantee it. Boy, you're a stickler. I didn't even really think about this. But you're right, device-bound should have a hyphen, shouldn't it.

**Steve:** Bugs me.

**Leo:** Yeah. I never even thought about that. Well, we just have to go with whatever the IETF thinks is right.

**Steve:** Maybe somebody at the World Wide Web Consortium is listening to this podcast and thinks, he's right. That's a typo.

**Leo:** Put in a hyphen. That's a typo.

**Steve:** Let's fix that.

**Leo:** Steve Gibson, ladies and gentlemen. We'll get to our hyphenless discussion in just a moment.

**Steve:** Well, and you know, if you were entitled to watch the game while you were home...

**Leo:** Right.

**Steve:** ...and you were traveling...

**Leo:** Right.

**Steve:** ...then it's not like you're doing anything wrong by still watching it.

**Leo:** No. In fact, I asked Netflix because I thought, well, should we be promoting this. They said, as long as you have a Netflix account, you can be watching Netflix in any other geographic location. That's fine. So that's, you know, that's what you can use a VPN for, too. And it's the only reason, you know, people say, well, you should use Tailscale or something local. But I can't do that, really. That doesn't work as well

if I want to be in London because my house is not in London. So there's an advantage to using ExpressVPN.

**Steve:** More flexibility.

**Leo:** More flexibility.

**Steve:** All right. So...

**Leo:** Let's take the Picture of the Week here.

**Steve:** I think we're going to have to not expose what this picture reveals because it would be a spoiler for those who want to encounter it and solve this puzzle themselves. Because this picture is...

**Leo:** It's a puzzle.

**Steve:** It takes the form of a puzzle, yes.

**Leo:** All right. I'm going to scroll up.

**Steve:** So by all means.

**Leo:** And I see Neil DeGrasse Tyson.

**Steve:** Yup.

**Leo:** And I can read it right away.

**Steve:** Yup. See, I knew you would be able to.

**Leo:** Yeah.

**Steve:** Yes.

**Leo:** But I won't tell you what it says.

**Steve:** You cannot tell us what it says.

**Leo:** But this is a famous example. I've seen other examples where they don't add numbers for letters, but where they take away letters.

**Steve:** Okay, right.

**Leo:** And it shows you how adaptive the human mind is.

**Steve:** Yes.

**Leo:** How able to fill in the gaps we are.

**Steve:** Yes. Yes. So this picture, I gave it the caption "Here's one to think about." And it's a T-shirt that Neil deGrasse Tyson is holding up, credited to a famous physicist in our midst. And I think everyone will enjoy taking a look at the picture.

**Leo:** Do people have a hard time reading this? Do people look at this and go "I don't know what it says?"

**Steve:** Yes.

**Leo:** Really.

**Steve:** I've had some feedback saying that they, you know, had to spend some time thinking about it.

**Leo:** Interesting.

**Steve:** So, and I knew you wouldn't, Leo, because you're just - you're no fun.

**Leo:** It's not LEET, exactly. Well, it's pretty close to LEET. It's pretty close to LEET. So maybe it's - I've spent too many years reading leetspeak.

**Steve:** Ah, that's a very good point. It bears a strong resemblance to that.

**Leo:** Yeah, yeah.

**Steve:** Okay. So nothing's been announced yet, and it's certainly not official. But it would make sense for Android to follow in Apple's footsteps with a higher security mode for their handset, similar to what Apple calls "lockdown mode." And with Google's annual I/O developer conference happening next month, it might be announced then, which might make it available in, you know, kind of the August-September timeframe as part of Android 16.

It's believed that Google has been quietly working on a new, more secure mode for Android that, as I said, was probably inspired by Apple's iPhone Lockdown Mode. According to a placeholder documentation page, which currently 404s, and based on analysis of Android Beta images, the new feature would be named the Android Advanced Protection Mode (AAPM). As with Lockdown Mode, AAPM would not be intended for regular Android users. It would be of use for probable target individuals who are more likely to face threats from oppressive regimes, you know, advanced spyware, network surveillance attacks and so forth.

It's believed to disable older and less secure 2G cellular connections; block users from sideloading apps from unknown sources. Presumably it prevents them from running apps that have already been sideloaded, I would imagine. I don't know. Enable "Memory Tagged Extension," which is a feature to block the exploitation of memory-related exploits; and force a reboot of any devices after more than three days of disuse. That forced reboot feature was spotted by Android Authority as a means of flushing RAM of any resident malware that may have taken up residence in the device during its owner's absence through whatever means, but then wasn't able to obtain persistence so that it, you know, wasn't able to write itself into the file system.

Although Google has offered no official confirmation of any such new Android Advanced Protection Mode, a large amount of code to support it is present in Android 16 betas. We've seen instances where something ended up in a further in the future release, not the most current, next, you know, forthcoming release, so maybe not 16, maybe not till 17. But this doesn't - it's not like this is rocket science, to like turn off things that it already has. So why wait? Anyway, the fact that it's in 16 Beta suggests that it will probably be official soon. Android Authority found the message that informs users that they may not sideload apps.

There's also support - I thought this was cool - for a new API which allows apps to detect whether the handset mode has this enabled so that they may apply any of their own security-enhancing behavior. You know, for example, a web browser might disable its internal JIT, its Just-In-Time compilation mode, when it detects that the handset is in this advanced protection mode because we know that the Just-In-Time compilers tend to be where a lot of security flaws have been found to reside in the past. Or, you know, another example may be Instant Messaging apps might disable their automatic display of multimedia content since, again, we've seen security vulnerabilities often discovered and leveraged in the interpreters that are used to display media.

So there are signs that something resembling Apple's Lockdown Mode may be coming soon to Android. And, you know, like Lockdown Mode, it probably reduces the device's convenient functionality too much to be used by most people. But, you know, it would make the smartphone much less fun to use. But the tradeoff is convenience versus security. And in this case, you know, you would be opting for security if you for some reason didn't have an Apple device, and this allows Android to do that.

Also while I was perusing recent news I saw that Chrome had recently moved to their release 135, and Firefox was now at 137. Among the changes, as I mentioned, in Chrome was the title of today's podcast, missing its hyphen, "Device Bound Session Credentials," which we'll be getting to here for a very deep technical dive at the end of today's podcast. But nothing else really stood out about Chrome's 135 beyond that.

The biggest news for Firefox 137 appears to be Tab Grouping, although the ability to use Firefox URL field as an ad hoc calculator for quick math actually excites me more, and I'm sure it's going to get much more use by me. Anyway, somehow I've broken the habit of having a seemingly near infinite number of tabs serving as placeholders for things I plan to get back to eventually. I remember maybe 10 years ago I had over there to my left a Firefox browser, and if it ever crashed, or I lost its tab lineup, it was my knowledge

base. I had so many open tabs. I don't do that anymore. I don't know what happened. But I just kind of got out of that, I guess. But I know from our feedback from our listeners of this podcast that there are many people who do still organize their life around browser tabs. And this is probably going to be a godsend for them.

So the Firefox 137 blog page explains, says: "Tab groups begin rolling out today. Stay productive and organized with less effort by grouping related tabs together. One simple way to create a group is to drag a tab onto another, pause until you see a highlight, then drop to create the group. Groups can be named, color-coded, and are always saved. You can close a group and reopen it later."

Okay, so I thought, great, let's try it. But no matter what I tried, and I was using Firefox 137 when I attempted to drag one tab on top of another. At some point, presumably once some center line somewhere was crossed, the underneath fixed tab that I was in the process of covering up would suddenly scoot over to fill the gap that was left from the tab I was dragging. No matter what I did, I was unable to, in any way, merge two tabs into a single group. I'm just telling everybody in case they have the same experience that I did that it didn't work for me.

Then I noticed that the phrase "Tab groups" was highlighted in the blog posting as a link. Clicking that, I discovered the likely cause of my trouble. That more detailed page, after I drilled down, said: "Starting in Firefox version 137, you can use tab groups to manage open tabs in Firefox by grouping them together and labeling them." Okay, right. Except it's not working. Then it said: "This feature is experimental and is being introduced" - I'm like, how hard is this to do? Why do you have to experiment with it? Anyway, it's being introduced to the Firefox user base through a progressive rollout. It may not yet be available to all users. Number me among them because I just can't get two tabs to merge.

So okay. You know, the Mozilla folks seem pretty excited about this, and they also noted that Firefox's new Tab Grouping system also works for Vertical Tabs. I long ago satisfied my absolute and utter need for vertical tabs using a pair of Firefox add-ons: "Tree Style Tab," which allows a hierarchy of tabs; and also "Tab Session Manager," which allows me to save current sets of tabs as a session and keep them in XML files, load them, save them, restore them, move them around. Love it. Anyway, together those two things do everything I need. But once support for native Tab Groups does finally arrive in my Firefox, which I don't yet have, I may look at switching to Firefox's native vertical tabs and using tab groups. Maybe that'll give me the same stuff that I have now.

**Leo:** I hate it when they do progressive rollouts like that.

**Steve:** Isn't it? It's like...

**Leo:** You just never know what features you have.

**Steve:** Right. And Leo, how hard can this be? You know, it's not like, oh, you know, we're going to upset people, or we're going to break things.

**Leo:** Well, I think that's - I think it's more that than is this going to work. I think it's more like people go, oh, my god, what happened?

**Steve:** I just, yeah, two tabs merged. How do I unmerge them?

**Leo:** Oh, they merged.

**Steve:** Wow.

**Leo:** Oh, well.

**Steve:** Anyway, earlier I said that the feature that appealed to me most was the ability to use Firefox's URL field as a quick ad hoc calculator. And even though that...

**Leo:** That puzzles me. Tell me how you do that?

**Steve:** Works. You just start, like, 35+7.

**Leo:** Really? That's a weird feature. Okay.

**Steve:** I kind of like it. Anyway, that one was enabled for me, and it worked. And it couldn't be any easier. Mozilla writes: "You can now use the Firefox address bar as a calculator. Simply type an arithmetic expression" - and you can use parenthetical, you know, prioritizing and so forth - "and view the result in the address bar drop-down. Clicking on this result will copy it to your clipboard."

**Leo:** I wish I had known this when I was doing my taxes yesterday.

**Steve:** Ah, there you go.

**Leo:** I had to fire up a calculator.

**Steve:** Anyway, yes. And now, you know, I'm often reaching for the calculator that's located next to me at my workspace. In fact, here it is. I've never talked about this. I love this. This is from the SwissMicros guys.

**Leo:** Oh, that's cool. Is that an - it's an HP-51 clone?

**Steve:** It's an HP-35 calculator clone, so it's RPN. It just - it's an extremely nice calculator. Took me a little while to get used to it. You can see that it's got next to the, up there, ABCDEF, so it's got - it's also hex. So it's multi-base calculator. Anyway, just I love this little thing. It took a while to get used to it, but I've got one in each of my locations. Anyway, so my point is I always have a calculator next to me. But, you know, sometimes if I just want to do a quick little bit of math, it's now in the address bar. Also what I noted was that this integrated calculator appears to be part of a larger address bar refresh and update.

**Leo:** Ooh, it's 250 bucks.

**Steve:** Even though they sort of list it on its own. Mozilla explains that we now have a Unified Search Button: "A new, easy-to-access button in the address bar helps you switch between search engines and search modes with ease. This feature brings the simplicity of mobile Firefox to your desktop experience," they said. So I guess mobile Firefox has already had that, and now we're getting it on our desktop. "Search Term Persistence," they said: "Now when you refine a search in the address bar, the original term sticks around, making it easier to adjust your queries and find exactly what you're looking for."

They also have a Contextual Search Mode. And this was freaky. "Firefox detects if you're on a page that has search capability and offers that option to you directly to search from the page's engine from the address bar." What? Anyway, they said: "Use this option at least two times, and Firefox will suggest adding the search engine to your Firefox." Which that was interesting. And then also finally " Intuitive Search Keywords: You can access various address bar search modes with convenient and descriptive keywords. So, for example, you start with @bookmarks or @tabs, or @history, or @actions." And the search will then be aimed at or focused on that specific aspect of Firefox. So anyway, that "Contextual Search Mode" where Firefox is supposedly detecting pages which offer their own searches, to me that's surprising and seems both aggressive and error-prone. So it'll be interesting to see how that all works out.

Anyway, beyond all this, Firefox 137 now identifies all links within the PDFs which its integral PDF viewer displays, turning them into hyperlinks, so it'll do that for you. You don't have to, like, you know, copy and paste them and all that. It's also possible to add your own signature to PDFs without leaving Firefox, and signatures can be saved for reuse later. And also Firefox now provides native support for the HEVC media format codex under Linux. So anyway, it occurs to me that all this further supports my ongoing contention that our web browsers have become incredibly complex, and only continue to become more so.

**Leo:** And this from a guy who uses a Reverse Polish Notation calculator, ladies and gentlemen.

**Steve:** That is true.

**Leo:** By the way, I did the search. I found SwissMicros.

**Steve:** Oh.

**Leo:** They have a whole bunch of different models.

**Steve:** They do. Now, those little credit card size ones have very cheesy keyboards. So, I mean...

**Leo:** Nah, I don't want that. Oh, you already ordered one, it sounds like.

**Steve:** Oh, I own them. I own them all, of course.

**Leo:** I want to get the DM42n. That's...

**Steve:** And it's got - they have nice fonts. They've got, I mean, there's just - they're so much in them. These guys are just...

**Leo:** You can connect them to external storage.

**Steve:** Yes, yes. And you're able to upgrade their firmware. They have a USB port along the top.

**Leo:** This is pretty cool.

**Steve:** They're neat people.

**Leo:** I might have to buy one. And I have no use for it. Zero. I still might have to buy one.

**Steve:** Well, there's always tax time next year, Leo.

**Leo:** That's right. Oh, yeah. That's what I bought it for, honey. It's for taxes. That's it.

**Steve:** No, I am, you know, often doing things. I'm computing, you know, currents and microamps and milliamps.

**Leo:** You've never shown me that before, I don't think.

**Steve:** I never have. I've never mentioned it.

**Leo:** Very cool.

**Steve:** Yeah. They are, they're neat people. I mean, it is a well-made - because you cannot, you can no longer get, which just boggles my mind, any of the good HP Scientific Calculators. The financial calculator was, like, the 15 or the 12. I don't remember. Not the 12. Anyway, there's the financial calculators that HP still makes. But they've given up, all the scientific ones are now algebraic notation instead of RPN.

**Leo:** No.

**Steve:** And they've got big screens that do graphics and all this crap no one really needs. They're just gimmicks. And so these people, they're the real deal. So, I mean, although we have 42 for our iPhone. And so I wonder, I mean, that's a gorgeous calculator, too. So...

**Leo:** Yeah, yeah. I'll put it next to my slide rule. I should have a collection.

**Steve:** It's nice to have clicky buttons. This thing's got beautiful, really nice, you know, keys. It's a great device.

**Leo:** It's pretty cool. I don't know what the price is because it's in Swiss francs, and I hate to see what the conversion's going to be.

**Steve:** It's a couple hundred dollars, and it takes a while to get it to you. But, and I don't know what tariffs...

**Leo:** What tariffs will do, yeah.

**Steve:** ...Trump has aimed at Switzerland. But, you know, I haven't heard them mentioned. So maybe they're just going to get the blanket tariffs. But it's got semiconductors in it. We'll be talking about that a little bit later because I was induced to upgrade my phone. Let's take a break. We're half an hour in, and we're going to talk about Apple and what they just did with their most recent upgrade, which caught some people by surprise.

**Leo:** Yeah. And by the way, there was one other reason that you might want this reboot thing after three days.

**Steve:** Yeah.

**Leo:** If somebody steals your phone, of course it's great to wipe the memory. But if somebody steals your phone, and they can't, you know, and they don't use it or you lose it, and they don't use it, it's nice to have it go into the fully locked mode.

**Steve:** Yes.

**Leo:** After a reboot. Because then it requires a password and all that stuff. I'm sorry, I'm a little busy right now ordering something from Switzerland. Don't - oh, he would want to - he would want to do an ad right now. All right. I got it. I ordered it.

**Steve:** And take a look at the - for our listeners who are interested, it's SwissMicros.

**Leo:** It's pretty cool.

**Steve:** All the documentation is there. I mean, they're engineers. They're Swiss engineers.

**Leo:** Oh, yeah, look at the people who are making your SwissMicro. I mean, this is a serious, serious device here. That's on the...

**Steve:** That's what happens when we give up China, Leo, is we go back to...

**Leo:** We'll make iPhones in the U.S. Sure we will. Sure we will.

**Steve:** Put some wheels on it.

**Leo:** Wow. Let me, yes, let's talk about the advertiser for this segment of Security Now! so you can get right back to our device bound...

**Steve:** And you can get back to ordering your next...

**Leo:** I already bought it. It's too late. I've got it. You've got to work fast on this show. I bought the 42N. I thought, why not just get the grandpa; right? No idea what it's going to cost me.

**Steve:** What do I have? I got the DM32, whatever it is.

**Leo:** Yeah, yeah. They both have RPN. I want to write little software programs.

**Steve:** You've got to, oh, and fully programmable and, oh, and, yeah, it's cool.

**Leo:** We were talking about Forth last week. It's kind of like having a little Forth calculator in your house. So this, the DHL delivery alone is 70 Swiss francs, so I may get it someday in the next six months.

**Steve:** DHL's a good delivery, though.

**Leo:** Oh, yeah, for international you kind of have to use DHL, yeah.

**Steve:** Yeah. Okay. So a posting over in OSXdaily had the headline of a Public Service Announcement. The headline read: "PSA: Automatic Update Enables Itself with MacOS Sequoia 15.4 & iOS 18.4." Now, maybe the guy got up on the wrong side of the bed, as they say. I'm going to share his posting. There's a grain of this that I kind of agree with. But, whoo, not quite to the extent that, I mean, he's really bent.

So he writes: "This is important and relevant to most Mac, iPhone, and iPad users: Installing the latest updates for MacOS Sequoia 15.4 for Mac, iOS 18.4 for iPhone, and

iPadOS 18.4 for iPad will forcibly enable automatic software update for system updates on your device."

**Leo:** Yeah?

**Steve:** Okay, now, given the fact that Updates can again be turned off, his use of the phrase "forcibly enable" seems maybe a little over the top. That implies that it would no longer be possible to again disable automatic updates, which is indeed possible. Anyway, the piece continues: "Some people may already have these auto-update features enabled on their devices and not mind this change."

**Leo:** Who wouldn't?

**Steve:** "Nor would they notice a difference."

**Leo:** No.

**Steve:** "Whereas there may be other people who intentionally disable automatic update and do not wish to have the auto-update feature forced upon their devices." Oh, well. He writes: "With Automatic Updates enabled, this means your Mac, iPhone, or iPad will automatically download and install system software updates onto your devices" - yeah, no kidding - "as they become available, without your approval or prompting."

**Leo:** Well, that's not true.

**Steve:** I know. "Automatic Updates may be problematic for many reasons. For one," he writes, "not everyone has the bandwidth available" - in their brain, apparently - "to automatically download huge software updates. Additionally, not everyone wants to install the latest software updates when they become available. Many users prefer to wait a little while to see if there are any critical bugs or issues discovered before putting the latest system software on their device." He says: "And this is a reasonable caution. Though it's not common, Apple has dumped out some bad software updates in the past..."

**Leo:** That's true.

**Steve:** "...that had to be pulled due to various issues."

**Leo:** That's true.

**Steve:** Yeah. And, of course, many Mac, iPhone, and iPad users just simply prefer to manually update and manage their devices on their own, without the computer or device doing it for them.

**Leo:** I've always has automatic turned on, and it always says, "There's an update. Would like to proceed?"

**Steve:** Yeah.

**Leo:** It's just downloading it ahead of time.

**Steve:** Right. He says: "But your personal computing behaviors and your" - get this, Leo. "But your personal computing behaviors and your opinion is irrelevant, as Big Cupertino knows what is best for you, your iPhone, your Mac, and your iPad." Right. "As we know, for the vast majority of their users they probably do know what's best." And he says, he finishes: "Apple has decided that you will have automatic updates enabled on your devices, and your installation of iOS 18.4, MacOS Sequoia 15.4, or iPadOS 18.4 was apparently used as an agreement to that setting change. If you don't like that, you can change it back and disable automatic system software updates." Well, and the rant continues, believe it or not.

**Leo:** And by the way, you still can turn it off. I'm just checking right now.

**Steve:** Of course.

**Leo:** You can turn it off.

**Steve:** I did, too. I did, too. I went over and looked, and like, okay, there's a big switch. Yeah. He says, anyway, we don't learn anything more from him beyond the fact that this author really, really...

**Leo:** Is upset.

**Steve:** ...dislikes the idea that Apple might feel that having automatic updates enabled for the masses is sufficiently important that it should be done. I can certainly agree that it would have been polite for Apple to ask before re-enabling disabled automatic updates, since if Apple were to find them disabled on a device, it would have had to be deliberate on the part of the device's owner to turn them off. But perhaps there are instances where that could have been malicious. I don't know. Maybe malware gets in and flips that off.

**Leo:** Oh, good point. Good point, yeah.

**Steve:** In order to keep itself around.

**Leo:** That's exactly why they do it, yes.

**Steve:** Yes.

**Leo:** But we've seen that kind of behavior in the past, yes.

**Steve:** Yes. And there might be something that they have done with this update that they might actually need to emergency roll back. But if automatic updates were off, they wouldn't be able to. So maybe they're saying, look, look, you know, we need to just kick this on again so, you know, for safety's sake. In any event, since I know there are many listeners of this podcast who do strongly prefer taking and having manual and deliberate control over the updating of anything, I wanted to make sure that everyone knew that the move to these latest macOS, iPhoneOS and iPadOS releases will have re-enabled, if we believe this guy, I don't know because I leave mine on, my phones are set to automatically update, so it was on after the most recent update, and I don't know if it turned it on.

**Leo:** If it turned it off, yeah.

**Steve:** Yeah. [Crosstalk]

**Leo:** Benito's saying he turns all updates off, always, on all of his devices.

**Steve:** Yes.

**Leo:** I don't know, did you notice 18.4 turning it back on again? You know, there is something I do get upset about. They turn on Apple Intelligence every single update. That's like a five or six gigabyte download. That you should get upset about because there's no security reason for that.

**Steve:** Yeah. Let's let this guy know because that would be good for another big rant.

**Leo:** Yeah, he's got a whole 'nother link baby blog post [crosstalk], yeah.

**Steve:** That's exactly right. Okay. Now, I also want to take a moment to note that I'm now the proud owner of a shiny new iPhone 16 Pro.

**Leo:** Ooh, fancy.

**Steve:** Now, as I've mentioned before, I had been happily using an older iPhone 12 Pro without any problems. But I became concerned last week over the threat of Chinese import tariffs significantly inflating the prices of iPhones. The threat appeared to be real, with Apple in a panic, you know, flying iPhones in from India. And, like, all kinds of, you know, kerfuffle about this. But after poking around Apple's site for a while, like looking at the 16 and, okay, and my 12 is still working good, I decided that my older iPhone, which was, as I said, still working just fine, would almost certainly last me through whatever tariff turbulence we were going to be experiencing even for the next few years. I later mentioned this to my wife, Lorrie, whose response was, "My god, buy yourself a new phone. Yours is old and small!"

**Leo:** She was talking about the phone.

**Steve:** Yes.

**Leo:** Now, see, this is why we get married. She's absolutely right. I would have said the same thing. You deserve a modern phone, Steve.

**Steve:** Right. I was driving a 20-year-old BMW when we met. And she was a little - it's a little sketchy. Why are you driving, you know, do you have any resources? Do you, you know, am I going to be picking up the tab?

**Leo:** Honey, I broke down on the 405. Can you come get me?

**Steve:** So last Thursday, I returned to the Apple Store, and I did that. Now, as we know, I'm not somebody who always needs to have the latest and greatest. My stash of Palm Pilots in the refrigerator is testament to that.

**Leo:** Oh, I hope Lorrie doesn't find those.

**Steve:** I'm also a testament to the "If it's not broke, don't fix it" school of thought. So I usually use electronics until they're worn right down to the nub. But I have to say that the 16 is a lot more responsive than the 12 was. And since I no longer wear a watch, every time I saw Lorrie's phone displaying the time of day on its dim OLED screen, I thought that was a terrific feature.

**Leo:** Yes. Mm-hmm.

**Steve:** You know, we purchased hers for her birthday, and she lives on that thing, way more than I do on mine. She'll - I don't get this. She'll be sitting right next to a booted-up desktop computer with a full-size screen and a keyboard that actually invites typing, rather than actively fighting against your data entry. And she'll be squinting at websites on her phone.

**Leo:** That's because she's a modern woman, Steve. She's modern.

**Steve:** I don't get it. I don't get it. In any event, last Friday, the day after I purchased the 16, the news broke that imports from China of smartphones and electronics were being exempted from the 154% import tariffs that had formed part of my purchase motivation. But then over this past weekend the U.S. Commerce Secretary, Howard Lutnick, explained during an interview on ABC's "This Week" Sunday morning show that, in another month or so, a new set of tariffs specifically targeting all semiconductor imports would be taking effect, and that smartphones would be caught up in that.

**Leo:** Sigh.

**Steve:** Now, a few months ago I purchased a new set of servers for GRC that I have not gotten around to deploying yet, but they're here. When the second one of an earlier set of five died a few months ago I decided that I needed to be ready in case I lost another one.

**Leo:** Wait a minute. You buy five at a time?

**Steve:** I had five running.

**Leo:** Oh, you have five servers all at once?

**Steve:** Yes.

**Leo:** Are they load balancing?

**Steve:** No. A couple are running - I think three are running Windows, two are running Unix, and they're in various state - you know me, security. So they're physically isolated. I'm not sharing function between a secure server and a server that's running PHP.

**Leo:** So each does something else.

**Steve:** Yes.

**Leo:** Okay.

**Steve:** Yes. So they have very specific...

**Leo:** Like an image server and a - right, okay.

**Steve:** Yeah. But also security boundaries. And as I said, the server that I have that has PHP on it, it's all by its lonesome. And it's got its own physical firewall. There are only a few things that it's able to do because PHP. And we're going to get to the audit which demonstrates the wisdom of that.

**Leo:** Yes.

**Steve:** So, and the fact that I myself just had to update PHP because of that CGI vulnerability that my version of PHP had at the time that this was happening. So anyway, so I've got three new, brand new servers. And I'm now somewhat more glad that I already have those in hand in case their cost might soon be increasing. You know, they were not inexpensive, and it appears that a few months from now they might become more expensive.

**Leo:** Can I ask you which company you buy from? I'm just curious.

**Steve:** Yeah, the servers that have been dying were Intel motherboard, you know, Intel SIRIUS, the best I could get servers at the time. And now I've switched to Supermicro because I do have a Supermicro machine that has been going for about 40 years. And it just will not die. So I thought, okay, I'm going to go back to the ones that seem more solid than Intel.

**Leo:** I think a lot of people will be very interested in your choice, so thank you.

**Steve:** And I've looked at - I've looked at the Intel motherboards, and I'm no longer impressed with their build quality. You know, they've got stuff like...

**Leo:** They used to be the king of the hill; didn't they.

**Steve:** I know. And that's why I thought I'm going to go with the best because, you know. And they end up paying for themselves in the long term. But two out of the five of these identical Intel servers just stopped working.

**Leo:** Do you buy towers or blades? What form factor do you have?

**Steve:** They're all - originally I had three 2U Intel servers, and these are all 1U. The other five are now 1U servers. And they're all 3.5" across drives in the front, all running RAID 6 with physical RAID 6 controllers because I'm still old-school.

**Leo:** And this is in your living room? Where are you...

**Steve:** Oh, no. These are all in a - over at Level 3.

**Leo:** Oh, they're over at Level 3. Oh, okay.

**Steve:** In their datacenter.

**Leo:** They're colo. Okay.

**Steve:** Yup. Yeah.

**Leo:** Cool. Thank you for sharing that.

**Steve:** Anyway, so - yeah.

**Leo:** You should have - I made on my website an "I use this" page. I don't use anything nearly as interesting as you do. But for people who are interested in what microphones we use and stuff, you should make a little "I use this" page. I think that'd be interesting.

**Steve:** Well, what's really interesting is that I also found myself purchasing some - oh, shoot. Now I'm forgetting. I ended up using a router that we've talked about in the past.

**Leo:** MikroTik?

**Steve:** No, it wasn't.

**Leo:** Not Ubiquiti.

**Steve:** Yeah, yes, it was Ubiquiti.

**Leo:** It was Ubiquiti. Yeah, I love my Ubiquitis.

**Steve:** Only the - there was one particular family of Ubiquiti routers that allowed me to do the static port address translation that I really need to do. I have some other big iron equipment that I was using back in the day, and one of them is still alive. Several of them have died. And I thought, okay, I need, you know, I need to have this functionality. And Ubiquiti is the router that, you know, it's - and boy, am I impressed with their technology.

**Leo:** Me, too, yeah. I'm really happy with it.

**Steve:** You know, you'll remember this. One of the things, speaking of, you know, what hardware I'm using, the most famous thing I did back in my TechTalk column days on InfoWorld was Steve's Dream Machine. Where, you know...

**Leo:** Yes, I remember.

**Steve:** ...I chose this motherboard, these drives, this controller, you know, this keyboard, you know. And I basically kitted out, like if you were going to build the ultimate machine that was also tricky because it was like, okay, these drives say they're only this big, but they actually have these extra cylinders on them, and you can format them to this size and get the maximum size partition, blah blah blah. I really spent a lot of time, you know, finding, like, the best value, not the most expensive, but the best value in each different category. And anyway, that was popular then.

Anyway, what I wanted to say is I have no crystal ball. And any rational actor, looking at the past month of tariff actions, would be foolish to place any large bet because who knows what's going to be true in the next hour. I'm quite certain that no one really knows what the future holds. But I very clearly heard the U.S. Commerce Secretary state that the administration's intention is to use higher import tariffs on all products

containing semiconductors to force a shift in semiconductor manufacturing from offshore to the U.S. So independent of the practicality, feasibility, and sanity of any of that, we may indeed see the cost of devices containing semiconductors rising. What I would be willing to bet on is that prices are certainly not going to be dropping anytime soon. I don't see any way that happens.

So I just wanted to take a moment to talk about this since I'm now more glad than I was that I had purchased those new servers a few months back. I would likely be doing that now for strategic savings if I had not already. I certainly don't know anymore about what's going to happen than anyone else. And this could all change tomorrow. That's the nature of where we are today. But if any of our listeners were waiting on the purchase of any big-ticket items containing semiconductors, it might be worth considering that prices may indeed be higher six months from now than they are today. I would certainly not place any bets on them being lower.

So, you know, as for my iPhone 16 Pro, if Apple ever does get around to deploying some useful AI, I'll be glad to have a device that allows me to experiment with it. My 12 wouldn't have. And in the meantime, it's nice to have a dim clock on the lock screen, and to be able to edit text messages that I've already sent. So happy that I made that jump.

**Leo:** Good.

**Steve:** And yes, Leo, we both have wives that said, oh, my god, come on.

**Leo:** Steve, you deserve it. Get it.

**Steve:** Your phone is old.

**Leo:** I mean, I understand the desire to run something into the ground. You still, I mean, you don't want to use the latest Windows either. So I understand that. That's commendable. But you deserve a nice phone.

**Steve:** Well, yeah. I just discovered yesterday that my iPhone 6, it used to be all pooched out because the battery expanded. But it turns out that goes down over time. So maybe I can bring that back to life.

**Leo:** No, no, no. No. and don't bring it on an airplane, either. Oh, my god.

**Steve:** Okay. So we were just talking about Apple silently enabling updates. Microsoft also recently made some news for Windows 11 Enterprise and Education users. And I'll bet you guys are going to be talking about it tomorrow on Windows Weekly.

**Leo:** Oh, yeah.

**Steve:** Windows 11 Enterprise and Education users will be getting updates on steroids in the form of the much-anticipated no-reboot-required hotpatching.

**Leo:** Hallelujah.

**Steve:** Yeah. Microsoft will then only require a once-per-quarter full cold reboot with all of the other interim updates able to be applied directly to Windows running in memory. So in other words, reboots dropped from 12 a year to four per year.

**Leo:** Wow.

**Steve:** So not over; but, you know, only one third as often. Microsoft's announcement blog posting about this is titled "Hotpatch for Windows client now available," where David Callaghan, writing for the Windows IT Pro Blog, says: "Hotpatch updates for Windows 11 Enterprise, version 24H2 for x64, both AMD and Intel CPU devices are now available. With hotpatch updates, you can quickly take measures to help protect your organization from cyberattacks, while minimizing user disruptions.

"Hotpatching represents a significant advancement in our journey to help you, and everyone who uses Windows, stay secure and productive. So let's talk about the benefits," he writes, "how it works, and how you and your organization can take advantage of this advancement as part of your Windows servicing journey. Hotpatching offers numerous enhancements when it comes to keeping Windows client devices up to date. Immediate protection: Hotpatch updates take effect immediately upon installation, providing rapid protection against vulnerabilities.

"Consistent security: Devices receive the same level of security patching as the monthly standard security updates released on the second Tuesday of every month. And minimized disruptions: Users can continue their work without interruptions while hotpatch updates are being installed. Hotpatch updates don't require the PC to restart for the remainder of the quarter." He says: "Note: OS features, firmware, and/or application updates may still cause a restart in the quarter."

He says: "You'll first create a hotpatch-enabled quality update policy in Windows Autopatch through the Microsoft Intune console. All eligible Windows 11 Enterprise, version 24H2 devices managed by this policy will be offered hotpatch updates in a quarterly cycle. The hotpatch" - and one also thinks, Leo, that maybe at some point in the future, once hotpatches have been proven and seen not to cause any trouble, Microsoft could certainly be pushing them out more frequently than quarterly.

**Leo:** That's a good point. Maybe monthly, yeah.

**Steve:** I mean, yes, more frequently than monthly if something bad happens and they want to immediately fix it. It's like, why not? It doesn't require, you know, any big change. So they said: "The hotpatch updates follow the same ring deployment schedule as standard updates. Devices receiving the hotpatch update will see a different Knowledge Base number tracking the hotpatch release and a different OS version than devices receiving the standard update that requires a restart."

Hotpatch updates operate on a quarterly cycle. So cumulative baseline month. So they said: "In January, April, July, and October, so four times per year, devices install the monthly fixed security update and restart. This update includes the latest security fixes, cumulative new features, and enhancements since the last cumulative baseline. Then subsequent two months: Devices receive hotpatch updates, which only include security

updates and do not require a restart. These devices will catch up on features and enhancements with the next cumulative baseline month, which is to say quarterly.

"This cycle," he wrote, "reduces the number of required restarts for Windows updates from 12 to just four per year, thanks to eight planned hotpatch updates annually. To enable hotpatching for Windows client devices, you'll need: A Microsoft subscription that includes Windows 11 Enterprise E3, E5, or F3; Windows 11 Education A3 or A5; or a Windows 365 Enterprise subscription. Devices running Windows 11 Enterprise, version 24H2 (Build 26100.2033 or later) and with the current baseline update installed. An x64 CPU including AMD64 and Intel," and he said: "ARM64 devices are still in public preview," but coming, so not available yet, but that'll happen. And, finally, "Microsoft Intune to manage deployment of hotpatch updates with a hotpatch-enabled Windows quality update policy."

Okay. So we've known for some time that patching Windows on the fly without rebooting is both possible and practical, since this has been an aftermarket feature that the gang over at 0patch have been offering for some time. So, you know, they do in RAM patching of DLLs that are loaded on the fly. So in instances where Microsoft has strategically decided to abandon Windows security, the ongoing availability of those zero-patches may be a godsend. But, bringing this to Windows enterprise and education client machines means that millions more systems will be able to receive the benefits of on-the-fly hot patching. Microsoft is not yet suggesting that this boot-avoidance technology might be available for their latest server platforms; but, boy, avoiding unnecessary server reboots would appear to be a nice feature for the future, you know, not having server downtime.

I don't have any problem with a brief once-a-month reboot of any of my workstation machines. You know, it's just not a problem for me. And, you know, Microsoft has already invested heavily in minimizing the time required to install updates. As we know, they no longer require the huge amounts of time they once did. I remember, like, sitting around, like for hours, while something spun around on the screen, or we watched dots chasing each other. You know, it's gotten a lot better. So for me, the monthly updates aren't causing much trouble.

Okay, now, a little bit just checking back briefly on where we are with Oracle before we take another break. The TL;DR on this is "They're still lying and denying." Which is just, it's like to everyone's amazement. Security researcher Kevin Beaumont, who we've followed often because he's very involved in the industry, published on Medium from his DoublePulsar.com site, under the headline "Oracle attempt" - Oracle, he uses the term "Oracle" as a...

**Leo:** That's British.

**Steve:** ...a plural, yeah.

**Leo:** Yeah, that's how the British do it. Companies are singular; in other countries it's often plural.

**Steve:** It keeps, it's like, you know, data technically is plural, but I never did it right.

**Leo:** Yeah, exactly.

**Steve:** Anyway, so he says: "Oracle attempt to hide serious cybersecurity incident from customers in Oracle SaaS service." Kevin wrote: "Being a provider of cloud SaaS (Software-as-a-Service) solutions requires certain cybersecurity responsibilities, including being transparent and open. The moment where this is tested at Oracle has arrived, as they have a serious cybersecurity incident playing out in a service they manage for customers. Back on March 31st, BleepingComputer ran a story around a threat actor named rose87168 claiming to have breached some Oracle services inside *.oraclecloud.com.'"

And of course our listeners may recall that the fact-digging Lawrence Abrams did for BleepingComputer, which we talked about, was so thorough as in my appraisal to cross the line from evidence to proof of Oracle's apparently deliberate obfuscation and misdirection about the incident. So Kevin continues: "Oracle told BleepingComputer and customers: 'There has been no breach of Oracle Cloud. The published credentials are not for the Oracle Cloud. No Oracle Cloud customers experienced a breach or lost any data.'"

He says: "The threat actor then posted an Archive.org URL and provided it to BleepingComputer, strongly suggesting they had write access to login.us2.oraclecloud.com, a service using Oracle Access Manager. This server is entirely managed by Oracle. Oracle have since requested Archive.org take down the proof, and the Wayback machine no longer shows the page. The threat actor then provided a several hour-long recording of an internal Oracle meeting, complete with Oracle employees talking for two hours. The two-hour video includes things like accessing internal Oracle password vaults and customer-facing systems. Both Hudson Rock and BleepingComputer were then able to confirm with Oracle customers that their data including staff email addresses was in data released by the threat actor.

"The threat actor, rose87168, is still active online and releasing more data and threatening to release more. They've also released data to cybersecurity threat intelligence providers. In data released to a journalist for validation, it has now become 100% clear to me that there has been a cybersecurity incident at Oracle, involving systems which processed customer data. For example, the threat actor has publicly provided complete Oracle configuration files current, also. As one example, they've provided Oracle web server configuration files. All the systems impacted are directly managed by Oracle. Some of the data provided to journalists is also current. This is a serious cybersecurity incident which impacts customers, in a platform managed by Oracle.

"Oracle are attempting to wordsmith statements around Oracle Cloud and use very specific words to avoid responsibility. This is not okay. Oracle need to clearly, openly, and publicly communicate what happened, how it impacts customers, and what they're doing about it. This is a matter of trust and responsibility. Step up, Oracle, or customers should start stepping off."

Kevin then provides three updates. In Update 1 he said: "Oracle rebadged old Oracle Cloud services to be Oracle Classic. Oracle Classic has the security incident. Oracle are denying it on Oracle Cloud by using this scope, but it's still Oracle Cloud services that Oracle manage. That's part of the wordplay." Second Update: "Although Oracle used the Archive.org exclusion process to remove evidence of writing to one of the Oraclecloud.com web servers, they forgot to remove a second URL that clearly shows the threat actor rose87168 having posted their email address on an Oracle Cloud page." And by the way, I went to that URL, and it is still there, and I saw rose87168@protonmail.com posted there.

**Leo:** On an Oracle hosted page.

**Steve:** Yes.

**Leo:** So that's pretty conclusive.

**Steve:** Yes. And then the third and final update: "Multiple Oracle Cloud customers have reached out to me to say Oracle have now confirmed" - get this, Leo - "Oracle have now confirmed a breach of their services. However, Oracle are only doing so verbally. They will not put anything in writing. So they're setting up meetings with large customers who query." He writes: "This is similar behavior to the breach of medical PII (Personally Identifiable Information) in the ongoing breach at Oracle Health, where they will only provide details verbally and not in writing."

Over on Mastodon, Kevin posted: "And now, a class action lawsuit has been filed against Oracle over a data breach at Oracle Health, which Oracle has not acknowledged in public." I have a link, if anyone's interested, to the class action breach court document PDF. He said: "This Oracle thing keeps getting more and more wild. I've never seen a response so bad from a large organization. They're throwing their own security staff under the bus by having them face customers, rather than the corporation actually take responsibility."

And, you know, Oracle's handling of all this could be taught, and should be taught, as a short course in how not to ever handle a data breach. This whole business of only having verbal conversations and refusing to put anything into writing feels like attorneys being asked how to run a company. I'm not sure that's a formula for success. Through my years as a small businessman I've had occasion to receive the advice of attorneys. I always thank them, and pay them, and carefully consider the value of their advice.

**Leo:** And then move on.

**Steve:** Yes. But what they would advise often seems to follow reactions to worst-case scenarios.

**Leo:** Right. They're there to protect you from the worst, yes.

**Steve:** Yes. Whereas I've found that being more open and trusting and optimistic has always worked better for me.

**Leo:** Me, too.

**Steve:** One of our listeners, whose name is Keith, wrote from Canada. He said: "Hi, Steve. Thank you for covering the Oracle Cloud breach in the latest episode highlighting the significance of the breach and the SEC violations. Given the 'OCI classic' breach, as they're dubbing it now, and the separate Oracle Health breach, I'm thoroughly confused on how they haven't had to disclose to the SEC. As a Canadian Oracle Health customer, it's very frustrating to me that they seem to be above SEC regulations and still refuse to disclose breaches to us so that we can be proactive in protecting our organizations. I'm a huge fan of you, Leo, and the show."

**Leo:** Aww.

**Steve:** "Thanks for everything you guys do."

**Leo:** Thank you.

**Steve:** And I wouldn't know what to tell Keith. You know, regulations only have teeth if they're backed up by the certainty of enforcement. And to say that things are somewhat confused in the U.S. at this particular moment could safely be considered an understatement. Both our DOJ and SEC are currently preoccupied with trying to figure out which end is up and what their priorities should be. So it may be that Oracle lucks out on this one, and that it slips by on the government side. But as I noted, U.S. citizens have already filed lawsuits that may force depositions to be taken and place additional facts on the record, which ultimately makes enforcement a given. So we'll see.

Okay. So the United States Treasury has something known as the Office of the Comptroller of the Currency, OCC for short. A couple of months ago, in January of this year, CISA discovered that the emails for nearly 100 of the OCC's staff had been intercepted since the breach originally occurred - get this, Leo - back in June of 2023.

**Leo:** Oh, my god.

**Steve:** Nearly two years.

**Leo:** Two years.

**Steve:** Encompassing more than 150,000 pieces of email. Someone has been rummaging around in there. None of the nearly 100 staffers at the U.S. Treasury's Office of the Comptroller of the Currency have enjoyed any actual email privacy. It's all just been an illusion. And Treasury does appear to be either a high-priority target or to have less than adequate security...

**Leo:** Or both.

**Steve:** ...since this OCC breach - yeah, both - is the third Treasury office to recently disclose a breach. Before this, we had the Office of Foreign Assets Control (OFAC) and the Committee on Foreign Investment in the U.S. (CFIUS). For both of those two previous intrusions the U.S. government has now credited the Chinese-backed hacking group Silk Typhoon.

Now, this news connected with something I heard over the weekend. An Asian analyst was interviewed by Fareed Zakaria during his Sunday morning show on CNN. She made the comment about how at some point, as tensions between the U.S. and China escalated, China might decide to weaponize all of the data they'd been collecting through their pervasive cyber intrusions into the U.S. That gave me a bit of a chill because, unfortunately, it really made sense. We've seen a great deal of evidence of Chinese, apparently state-sponsored, actors rummaging around inside U.S. government and industry networks. But nothing overt and obvious has come of it. It might be that an

"attack" as such, and I have that in quotes, would take the form of using all of the information that's been gleaned against U.S. interests. In other words, "weaponizing all of that data."

We don't know, you know, that this recent and long-running U.S. Treasury Office of the Comptroller of the Currency email breach was the same as who previously was found to have breached those other two U.S. Treasury offices. So far there's been no attribution. But at this point it would almost be surprising if it wasn't the Silk Typhoon group backed by China. So it would be so much better if we could all just get along. That doesn't seem to be happening, though, sadly.

There's some news on the Apple vs. the UK and what Apple will do about the UK's demands to be able to obtain the stored iCloud data for anyone in the world they request. Apple Insider's headline was: "UK iCloud backdoor mandate hearing must be made public eventually."

**Leo:** Oh.

**Steve:** They wrote: "After a legal challenge by Apple, the hearing about blowing open Apple's iCloud encryption in the UK for the sake of national security will not be kept secret, but it's not clear when the details will be made public. After the hearing about a mandated backdoor happened behind closed doors, Apple very nearly immediately filed an appeal, with the backing of most of the world's governments, privacy advocates, and journalism organizations. That appeal has been heard, and at some point the results of the hearing will be made clear.

"The Investigatory Powers Tribunal rejected claims from the UK government that national security would be hurt by revealing the results of the hearing, or exposing who attended the hearing. In short, the appeal found that there was no reason to restrict what it calls "open justice," so the results of the hearing must be made clear in due time. It's not clear when that will happen, as case management orders will be made only after Apple and the UK government have time to consider the ruling and propose drafts."

So at least we're going to find out what that, you know, is about. Basically we've got bureaucracy. Whatever is going to happen will apparently grind away slowly. But the fact that the UK government now knows that it will not also be able to conduct everything in secret may, hopefully, dampen their zeal somewhat and reign them in. What's interesting about this is that there's no middle ground here. There's no gray area. UK users either will or will not have the ability to enable Apple's Advanced Data Protection for their stored iCloud data.

It seems unlikely in the extreme that the UK's demand to be able to obtain the data belonging to anyone they choose anywhere in the world has any chance of ever happening. But they might well force Apple to disable ADP for citizens in the UK. We'll see. But again, the only good thing about this is that it's black and white. That is, either you have it or you don't. So hopefully the fact that there's a sharp point on this will help, you know, a clean decision, a clean and clear decision to come out of all this.

Now, I missed this news, this next news, when it happened 10 days ago. But I felt the need to come back to put it on everyone's radar because what Mozilla is doing with a suite of new cloud service offerings which they're calling, unfortunately, Thundermail...

**Leo:** Thundermail. That's what you need. You've got to say it right.

**Steve:** Oh, thank you. I will need you again when we're talking about Roskomnadzor.

**Leo:** Roskomnadzor. I'm sorry.

**Steve:** No, it's good. We have Thundermail and Thunderbird Pro. I'm sure this will be of interest to many of our listeners for much the same reason we choose to use Mozilla's Firefox.

**Leo:** Mm-hmm.

**Steve:** So Mozilla wrote: "Today we're pleased to announce what many in our open source contributor community already know. The Thunderbird team is working on an email service called Thundermail."

**Leo:** Good. Another way to make money. That's good, yes.

**Steve:** Yes, exactly. Exactly, Leo. "As well as file sharing, calendar scheduling, and other helpful cloud-based services that as a bundle we have been calling Thunderbird Pro. First, a point of clarification: Thunderbird, the email app, is and always will be free. We will never place features that can be delivered through the Thunderbird app behind a paywall. If something can be done directly on your device, it should be. However, there are things that cannot be done on your computer or phone that many people have come to expect from their email suites. This is what we are setting out to solve with our cloud-based services.

"All these new services are, or soon will be, open source software under true open source licenses. That's how Thunderbird does things, and we believe it is our super power. It's also a major reason we exist, to create open source communication and productivity software that respects our users. Because you can see how it works, you can know what it's doing, and that it's doing the right thing.

"The Why for offering these services is simple." Okay, now, the truth is they want to survive. But, okay. They wrote: "Thunderbird loses users each day to rich ecosystems that are both products and services, such as Gmail and Office 365. These ecosystems have both hard vendor lock-ins through interoperability issues with third-party clients, and soft lock-ins through convenience and integration between their clients and services. It's our goal to eventually have a similar offering so that a 100% open source, freedom-respecting alternative ecosystem is available for those who want it. We don't even care if you use our services with Thunderbird apps. Go use them with any mail client. No lock-in, no restrictions, all open standards. That is freedom. So what are the services?"

They have Thunderbird Appointment. "Appointment," they write, "is a scheduling tool that allows you to send a link to someone, allowing them to pick a time on your calendar to meet. The repository for Appointment has been public for a while and has seen pretty remarkable development so far. It's currently in a closed Beta, and we're letting more users in every day. Appointment has been developed to make meeting with others easier. We weren't happy with the existing tools as they were either proprietary or too bloated, so we started building Appointment."

Then there's Send. "Send is an end-to-end encrypted file sharing service that allows you to upload large files to the service and share links to download those files with others.

Many Thunderbird users have expressed interest in the ability to share large files in a privacy-respecting way, and it was a problem we were eager to solve. Thunderbird Send is the rebirth of Firefox Send - well, kind of. We've rebuilt much of the project to allow for a more direct method of sharing files, from user-to-user without the need to share a link. We opened up the repo to the public earlier this week. So we encourage everyone interested to go and check it out. Thunderbird Send is currently in Alpha testing, and will move to a closed Beta very soon."

Thunderbird Assist. "Assist is an experiment, developed in partnership with Flower AI, a flexible, open-source framework for scalable, privacy-preserving federated learning that will enable users to take advantage of AI features. The hope is that processing can be done on devices that can support the models. And for devices that are not powerful enough to run the language models locally, we are making use of Flower Confidential Remote Compute in order to ensure private remote processing, very similar to Apple's Private Cloud Compute. Given some users' sensitivity to this, these types of features will always be optional and something that users will have to opt into. As a reminder, Thunderbird will never train AI with your data. The repo for Assist is not public yet, but it will be soon."

And then Thundermail. Thundermail is an email service in search of a better name. No, okay, that's not what it actually says. I just think that "Thundermail" sounds dumb.

**Leo:** It's because it's Thunderbird. I guess I understand.

**Steve:** Right. You know, you just can't put "thunder" in front of everything...

**Leo:** In front of anything. It's Thundernow.

**Steve:** It's Thunderdome.

**Leo:** It's Steven "Thunder" Gibson.

**Steve:** Oh, god. Anyway, it also supports calendars and contacts as well as mail. They wrote...

**Leo:** I'm interested. I mean, I'm a Fastmail customer, which does all the same things. But I'm very interested. I'd like to find out more.

**Steve:** They said: "We want to provide email accounts to those who love Thunderbird, and we believe that we are capable of providing a better service than the other providers out there, email that aligns with our values of privacy, freedom, and respect for our users. No ads, no selling, no training AI on your data, just your email, and it is your email. With Thundermail, it is our goal" - I can't, my god, please, something else.

**Leo:** You can't resist.

**Steve:** I can't. "It is our goal to create a next-generation email experience that is completely, 100% open source and built by all of us, our contributors and users. Unlike the other services, there will not be a single repository where this work is done. But we will try and share relevant places to contribute in future posts like this. The email domain for Thundermail will be Thundermail.com" - thank god - "or tb.pro. Additionally, you will be able" - here it is - "to bring your own domain on day one of the service."

**Leo:** Nice. Good. That's critical for me.

**Steve:** Now, that starts being interesting, having Mozilla behind a 100% open source privacy-respecting email service where we're also able to bring our own domain, presumably by pointing our own domain's MX records at Mozilla. That would be cool.

So everyone listening can head to Thundermail.com. You will get - the only thing there at Thundermail.com is a simple signup page demonstrating their inherently techie nature. You'll see what I mean when you see the page.

**Leo:** Yes. It's command line based.

**Steve:** Yes. And that allows you to sign up for their Beta waitlist, which will give you notification as soon as this thing is, you know, as soon as you're able to actually sign up for the service. And I did that immediately.

**Leo:** Oh, yeah. Me, too. Yeah, I'm very curious, yeah.

**Steve:** So they said under Final Thoughts: "Don't services cost money to run?" And they said: "You may be thinking this all sounds expensive. How will Thunderbird be able to pay for it?" And they say: "And that's a great question." Right, answering it, or asking it of themselves. And they said: "Services such as Send are actually quite expensive. Storage is costly. So here's the plan. At the beginning, there will be paid subscription plans at a few different tiers. Once we have a sufficiently strong base of paying users to sustainably support our services, we plan to introduce a limited free tier to the public. You see this with other providers. Limitations are standard as free email and file sharing are prone to abuse."

**Leo:** Yes.

**Steve:** Yeah. "It's also important to highlight again that Thunderbird Pro will be a completely separate offering from the Thunderbird you already use." Or in my case "once used," since I still am happily switched away from Thunderbird to eM Client. They said: "While Thunderbird and the additional services may work together and complement each other for those who opt in, they will never replace, compromise, or interfere with the core features and free availability of Thunderbird. Nothing about your current Thunderbird experience will change unless you choose to opt in and sign up with Thunderbird Pro. None of these features will be automatically integrated into Thunderbird desktop or mobile, or activated without your knowledge. This has been a long time coming."

And the person who posted this wrote in the first person: "It is my conviction that all of this should have been part of the Thunderbird universe a decade ago. But it's better late than never. Just like our Android client has expanded what Thunderbird is, as will our iOS client, so too will these services. Thunderbird is unique in the world. Our focus on open source, open standards, privacy, and respect for our users is something that should be expressed in multiple forms. The absence of Thunderbird web services means that our users must make compromises that are often uncomfortable ones. This is how we correct that." In other words, they're going to be providing a complete suite of web services like the other guys do.

And he finished, writing: "I hope that all of you will check out this work and share your thoughts and test these things out. What's exciting is that you can run Send or Appointment today, on your own server." I thought that was interesting. You can run Thunderbird Send or Thunderbird Appointment today on your own server. He said: "Everything that we do will be out in the open, and you can come and help us build it. Together we can create amazing experiences that enhance how we manage our email, calendars, contacts, and beyond. Thank you for being on the journey with us." And so we all want Mozilla to stay alive. If not for Thunder whatever, then for the sake of Firefox.

**Leo:** Yes.

**Steve:** So if their addition of cloud-based services appeals to people as a reasonable alternative to Office 365 and Gmail, and that creates a revenue stream to support all of Mozilla, then I'm all for it. So again, Thundermail.com to sign up for the news. And yay.

A quick note is that over in the category of age restrictions, Meta has extended teen account protections. The existing "teen accounts" security protections which exist on Instagram are also being extended to Facebook and Facebook Messenger accounts. Now, the feature prevents children under the age of 16 from modifying a series of privacy settings on their accounts without a parent's approval. This includes settings related to who can contact the account, and what content they see on the sites. Meta is also expanding these restrictions so that, for example, teens won't be able to live stream on their sites without a parent's approval. So that's good.

Okay. So with our podcast two weeks ago falling on April Fool's Day, that made last week's podcast fall on the earliest possible Patch Tuesday day, April 8th. Looking back at the news of last week, Microsoft patched 126 vulnerabilities.

**Leo:** Wow.

**Steve:** Because, you know.

**Leo:** Every month.

**Steve:** Every month.

**Leo:** It's always.

**Steve:** That's right.

**Leo:** I mean, I guess it's good they're patching them.

**Steve:** It's better than not.

**Leo:** Yeah.

**Steve:** You know me, I wish they'd just leave it the heck alone and stop messing with it. But no. One of those was an actively exploited zero-day. It was an Elevation of Privilege in the Windows Common Log File System driver, which tends to be a vulnerability magnet for some reason. They've had a lot of problems with that driver over the years. Microsoft's security team, I mean, okay. So it's a log file system driver. Probably some summer intern. They said, hey, just go do the, you know, write the logging driver while you're here for the summer. We saw that happen with the color mapping that NT did once, and it was a disaster. So anyway, you want to put your good guys on the things that are going to run in the kernel.

Microsoft Security Team indicated that the now-patched zero-day was being exploited by the RansomExx ransomware group. And that makes sense since, once you somehow arrange to get your code running on a well-locked-down Windows machine, that code will likely be running under the account of the user who somehow made a mistake that allowed it to come in and run with deliberately restricted privileges. So even though you may be in as a bad guy, it's still generally necessary to arrange to obtain admin privileges if you're, you know, in the case with a ransomware intrusion, your goal is to do a lot of damage. You need to get root on the machine to do that.

Google also patched a pair of zero-days last week with Android. One of the fixes is a patch for a Cellebrite exploit used by Serbian authorities to unlock the phones of journalists and anti-government protesters. The exploit and the hacks were first detailed in an Amnesty International report in February. There are no details on the second zero-day other than that it leverages an undisclosed flaw in the Android kernel USB audio driver. But being in the Android kernel suggests that it was likely a powerful root-level exploit. This also makes it the third month in a row that Google has fixed zero-days in the Android OS. And as we know, these things are complicated, and it's very difficult to get every little detail right. But that's what security requires.

If I wasn't so excited about talking about Device Bound Session Credentials today, as we will be shortly, I would be spending our time digging into a 25-page, recently published piece of security research which was just so juicy. It examined the status of the security of PLCs, you know, the critical Programmable Logic Controllers that generally contain just enough computational ability to figure out when to turn off the toilet paper rolling machine, to then cut the paper and start on another role after first painting a little bit of glue onto the cardboard tube so that the new end of the paper sticks to it. You know, that's what these things do.

In a very real sense, PLCs are what actually run the world. We've talked about them extensively in the past on this podcast specifically because they're silent workers that essentially make all of today's infrastructure go. In a very real sense, they are today's infrastructure. And as a consequence, their security is crucial.

In the Abstract of their 25-page paper, the team of researchers wrote: "Billions of people rely on essential utility and manufacturing infrastructures such as water treatment plants, energy management, and food production." You know, not to mention nuclear reactors. "Our dependence on reliable infrastructures makes them valuable targets for

cyberattacks. One of the prime targets for adversaries attacking physical infrastructures are Programmable Logic Controllers because they connect the cyber and the physical worlds.

"In this study, we conduct the first comprehensive systematization of knowledge that explores the security of PLCs. We present an in-depth analysis of PLC attacks and defenses, and discover trends in the security of PLCs from the last 17 years of research. We introduce a novel threat taxonomy for PLCs and Industrial Control Systems. Finally, we identify and point out research gaps that, if left ignored, could lead to new catastrophic attacks against critical infrastructures."

Now, as I promised, and as I said, I'm not digging into this. I mean, I would love to. But we don't have time. But here's a brief summary of that research written by a security reporter who did dig into it. He wrote: "A team of academics has conducted a review of 133 papers, 119 attack methods, and 70 defense methods that target PLCs to assess the actual impact of a possible cyberattack targeting these devices. The research found that, even if most PLCs have built-in access control features, most of them have been shown to be ineffective. Where encryption has been used, the algorithms are often ineffective. Disabling unused protocols and monitoring is the best way to prevent and detect attacks." So if anyone is interested in more detail, I have a link to their 25-page research analysis in the show notes.

Okay. I've got one that's pretty much guaranteed to make you just shake your head. And Leo, I know you already know about this. Six researchers, four from the University of Texas at San Antonio, one from Virginia Tech, and the last one from the University of Oklahoma, just published a paper titled "We Have a Package for You! A Comprehensive Analysis of Package Hallucinations by Code Generating LLMs." You know, large language models. In their usage, just to be clear, by "package" they mean a reference to some open source code library that would be handy to have and to add to a project in order to provide some missing functionality. So here's what this team of six wrote for their paper's Abstract. I have a link to their entire paper in the show notes.

They wrote: "The reliance of popular programming languages such as Python and JavaScript on centralized package repositories and open source software, combined with the emergence of code-generating Large Language Models (LLMs), has created a new type of threat to the software supply chain: package hallucinations. These hallucinations, which arise from fact-conflicting errors when generating code using LLMs, enable a novel form of package confusion attack that poses a critical threat to the integrity of the software supply chain. This paper conducts a rigorous and comprehensive evaluation of package hallucinations across different programming languages, settings, and parameters, exploring how a diverse set of models and configurations affect the likelihood of generating erroneous package recommendations and identifying the root causes of this phenomenon.

"Using 16 popular LLMs for code generation and two unique prompt datasets, we generate [get this] 576,000" - over half a million - "576,000 code samples in two programming languages that we analyze for package hallucinations. Our findings reveal that the average percentage of hallucinated packages is at least 5.2% for commercial large language models and 21.7% for open source large language models, including a staggering 205,474 unique examples of hallucinated package names, further underscoring the severity and pervasiveness of this threat.

"To overcome this problem, we implement several hallucination mitigation strategies and show that they're able to significantly reduce the number of package hallucinations while maintaining code quality. Our experiments and findings highlight package hallucinations as a persistent and systemic phenomenon while using state-of-the-art large language

models for code generation, and a significant challenge which deserves the research community's urgent attention."

Okay. So that's part one. LLMs are still just making stuff up, including the names of add-on packages that it would be nice to have. And just as "typosquatting" has developed over time into a serious threat, researchers are warning that something which unfortunately is being called AI "slopsquatting" is on the horizon.

**Leo:** Let me see if it sounds better when I say it this way: "AI Slopsquatting."

**Steve:** Uh, no.

**Leo:** No, no better.

**Steve:** Still bad. Here's what the Risky Business security newsletter wrote. They said: "Security firms, open source experts, and academics are warning about a new supply chain vector they're calling "slopsquatting." The technique's name is a combination of terms like AI slop and typosquatting. It revolves around the increasing use of AI coding tools to generate blocks of source code that may sometimes make their way into production systems.

"A recent academic paper" - and that's the one whose Abstract I just shared - "analyzed 16 AI coding models and found that these tools generate shoddy code that often includes and loads packages and libraries that don't exist. DevSecOps company Socket Security says that such behavior opens the door to slopsquatting, where threat actors study the LLMs and then register package names hallucinated or likely to be hallucinated in the future." It turns out that's actually feasible.

"The attack looks farcical and impractical, but so did typosquatting," they write, "when it was first described years ago. Yet, years later, it is one of the most pervasive and common sources of supply chain issues in the software development industry. It may sound ridiculous that developers would not spot a typo in the names of packages they install, but reality has shown that they don't. Does it actually sound that far off," he poses, "that developers would not spot nonexistent packages in huge blocks of code they're using when cutting corners?"

**Leo:** Yeah, see, that's the problem; right?

**Steve:** Yes. "The use of AI coding tools is increasing, and the chances that developers may use code blocks generated through these tools is also growing exponentially, along with the chances of a successful slopsquatting attack." So that's Risky Business wrote. This raised my curiosity, so I looked further.

The Socket Security folks further summarized some of the paper's findings. They wrote: "The researchers tested 16 leading code-generation models, both commercial (like GPT-4 and GPT-3.5) and open source (like Code Llama, DeepSeek, WizardCoder, and Mistral), generating a total of 576,000 Python and JavaScript code samples. Their key findings were 19.7% of all recommended packages did not exist. Open source models hallucinated far more frequently, 21.7% on average, compared to

commercial models at 5.2%. The worst offenders (Code Llama 7B and Code Llama 34B) hallucinated in over a third of its outputs. GPT-4 Turbo had the best performance with a hallucination rate of just 3.59%. Across all models, the researchers observed over 205,000 unique hallucinated package names. These findings point to a systemic and repeatable pattern, not just isolated errors."

And here's the key: These hallucinations are not just one-offs. If they were, they could not be weaponized; right? They are persistent and recurrent. The Socket Security guys explained. They said: "In follow-up experiments, the researchers reran 500 prompts that had previously triggered hallucinations, 10 times each. They found an interesting split when analyzing how often hallucinated packages reappeared in repeated code generations.

"When re-running the same hallucination-triggering prompt 10 times, 43% of hallucinated packages were repeated every time, while 39% never repeated at all. This stark contrast suggests a bimodal pattern in model behavior: hallucinations are either highly stable or entirely unpredictable.

"Overall, 58% of hallucinated packages were repeated more than once across 10 runs, indicating that a majority of hallucinations are not just random noise, but repeatable artifacts of how the models respond to certain prompts. That repeatability increases their value to attackers, making it easier to identify viable slopsquatting targets by observing just a small number of model outputs. The consistency makes slopsquatting more viable than one might expect. Attackers don't need to scrape massive prompt logs or brute force potential names. They can simply observe LLM behavior, identify commonly hallucinated names, and register them."

So just a cautionary tale here about the potential for the weaponization of large language model outputs. We know that bad guys would like nothing more than to get their code included into high-profile product offerings. If future coders become too comfortable with directly using LLM-created code without scrutinizing it carefully, I would argue line by line, just copying and pasting and testing what the LLM produces, it's no longer far-fetched to imagine that the LLM's mistaken output itself might have been weaponized for the purpose of causing the download and inclusion of a malicious library.

If we were to take this a step further, imagine arranging to seduce LLMs to train on tasty valid libraries, which they would tend to then invoke into their solutions, only to have any retrieval by a non-LLM return a malicious version of that package. There's no such thing as a free lunch, coders.

**Leo:** And how do you test it? Because you can't just say, well, does this exist because it does exist now because of...

**Steve:** Right.

**Leo:** ...slopsquatting. And, you know, so now you have to validate all the libraries to make sure it's doing - not doing anything malicious. Ay ay ay. What a mess.

**Steve:** Yup. A real supply chain mess.

**Leo:** Yeah.

**Steve:** Basically the LLM has a knowledge that the coder lacks of available packages and is pulling stuff in from all over.

**Leo:** Right.

**Steve:** So the coder either needs to truly educate themselves about the nature of the libraries that the LLM knows about and has invoked, or just hope for the best. And hoping for the best could really bite you in the - what is not the best place.

**Leo:** Yes. Wow.

**Steve:** We wind up talking about WordPress because such a large portion of the Internet's websites are running WordPress CMS (Content Management System) code. The core WordPress offering has become extremely solid over time. But its very large plug-in ecosystem is another matter entirely. That plug-in ecosystem is WordPress's primary attraction, but also its primary weakness as a secure platform.

WordFence is an independent WordPress-focused security firm. During the previous year, security researchers at WordFence discovered and disclosed more than 8,000 WordPress site vulnerabilities. 8,000 WordPress site vulnerabilities. But fully one quarter of those have remained unpatched. 2,000 unpatched today. Many of the affected plug-ins are obscure, but many are popular and unmaintained. But as I noted, the WordPress core has grown increasingly solid, with only five of those 8,000 known issues disclosed last year impacting the WordPress core, and all of them were immediately fixed.

So the takeaway here is this. As I've said every time we've previously considered the important WordPress landscape, be very, very careful about what you add to the base WordPress core offering. Only add those features you really need and will really use, and check to see the history of any add-on's maintenance to verify that someone is still around to maintain that code, or that it really looks like it is sufficiently solid because add-ons are the WordPress security Achilles heel, not the core offering.

**Leo:** It's the plugins, yeah.

**Steve:** Yup.

**Leo:** But you really can generalize this advice to everything. Don't install apps you don't need.

**Steve:** Same with your iPhone.

**Leo:** Yeah. Don't use libraries you don't know.

**Steve:** That really is true.

**Leo:** That, you know, the browser you use is probably secure.

**Steve:** And the add-ons to the browser. The more crap you add to these things, yup, the greater the probability that one you add will be bad. Yup.

**Leo:** Especially nowadays. Holy cow.

**Steve:** And there are of course degrees of badness. And one could argue that WordPress add-ons - the problem is, you know, they're just written by, you know, Johnny in the closet. I mean, they're just random.

**Leo:** And what are they written in, Steve? They're written in PHP.

**Steve:** Uh-huh. They are.

**Leo:** Johnny in the closet using his personal home page software.

**Steve:** That's right. And that's why the only server I have that is running any PHP has its own port on an isolated router, and it doesn't get to talk to any of my other stuff at GRC.

**Leo:** That's smart.

**Steve:** Because I just do not trust it. It can melt down internally, but it can't touch, you know, GRC.com where, you know, ecommerce and other things live because, you know, I take my own advice.

So speaking of PHP's language interpreter, it just got a much welcomed security audit, which it turns out was also much needed. WordPress, like a great many other web-facing systems such as I was just talking about, GRC's web forums, our email system, our link shortener, all written in PHP. I love them, but they're on an isolated server.

So also in the news was that PHP's language interpreter recently received a security audit. Quarkslab received a commission to really examine the core component of PHP. Last Thursday they posted their results. They wrote: "The Open Source Technology Improvement Fund, Inc., thanks to funding provided by the Sovereign Tech Fund, engaged with Quarkslab to perform a security audit of PHP-SRC, the interpreter of the PHP language.

"The audit aimed to assist PHP's core developers and the community in strengthening the project's security ahead of the upcoming PHP 8.4 release. The codebase was analyzed with a defined scope, which was established and agreed upon by both PHP's core developers and the OSTIF (Open Source Technology Improvement Fund) teams. Based on this scope and the allocated timeframe for the audit, an attack model was developed and approved by the PHP team. The assessment was conducted within a set timeframe, with the primary focus on identifying vulnerabilities and security issues in the code according to the defined attack model.

"The following scope of work was defined by PHP Foundation and the OSTIF. The key tasks included base tooling evaluation;

improve SAST tooling to enhance the existing GitHub CI without extra cost and with low maintenance; build fuzzers compatible with oss-fuzz for potential critical functions that are not

currently covered; cryptographic and manual code review. High-priority tasks were the php-fpm master node and php-fpm worker glue code. Those are the modules that invoke PHP for handling web queries.

"Also FPM pool separation; the MySQL Native Driver;

RFC 1867 PHP header parsing and MIME handling; PDO emulated prepares; JSON parsing with a focus on json_decode; OpenSSL external functions and its stream layer external openssl; libsodium integration with ext/sodium; functionalities related to passwords ext/standard/password.c; functionalities related to hashing ext/hash; and functionalities related to CSPRNG, the Cryptographically Secure Pseudorandom Number Generator, ext/random/csprng.c."

So that was their mission and scope. How did they proceed? They wrote: "To assess the security of PHP-SRC, Quarkslab's team first needed to familiarize themselves with the structure of the project and understand the key tasks outlined in the audit's scope. To achieve this, Quarkslab experts gathered and reviewed the available documentation and project resources. With a clear understanding of the features to be evaluated, Quarkslab developed an attack model that incorporated all the requested key tasks. This model was then presented to PHP's core developers; and, once approved, the assessment began.

"The evaluation employed a combination of dynamic and static analysis. The static analysis focused on scrutinizing the source code to visually identify vulnerabilities related to the implementation and logic of the specified assessment targets. Dynamic analysis was used to complement the static review by speeding up the process through fuzzing and validating or refuting the hypotheses generated during the static analysis."

So, you know, and they're taking this formal approach because they've been contracted, essentially, to perform this audit. And it would be easy to say, oh, yeah, we did. But, you know, they're getting paid. So they need to say, what do you want us to do? Okay. Here's how we're going to do it. Okay? Okay. Now we're going to do it.

So what did they find? They wrote: "During the timeframe of the security audit, Quarkslab has discovered several security issues and vulnerabilities, among which were two security issues considered as high severity; six security issues considered medium severity; nine security issues considered low severity; and 10 issues considered informative. Most vulnerabilities have been shared," they wrote, "via security advisories on the PHP-SRC GitHub repository. Other bugs and issues are provided only in this report. Four CVEs were issued, one for each of the two high severity vulnerabilities, and two others for two of the nine low severity vulnerabilities."

Okay. So they produced a detailed - oh, boy - a very detailed 106-page full audit report, and I have a link to it in the show notes for anyone who wants to dig in. However, they also wrote: "This audit report contains two security issues currently redacted, while PHP maintainers are actively working on the fixes. Details will be provided after fixes are applied by PHP maintainers. Fixes are complex and in progress."

In other words, two of the 17 security-related problems they discovered were too severe to publicly report until they have been fixed. Although it's speculation at this point, this strongly suggests that many earlier releases of PHP are also very likely to be in identical trouble; and that, depending upon what bad guys could do with it if they knew about it, we may be facing a critically important security update across all still supported release

versions of PHP. So we will certainly be, you know, standing by and staying tuned and see whether PHP needs an update. They're not talking about what they found.

But it is very, very cool that a truly worthwhile audit was done of PHP. And really, you end up feeling a lot better about PHP 8.4, knowing that it has had this kind of audit. It's like back in the days of VeraCrypt, or TrueCrypt, that got audited. And it's like, okay, people really did take a look at it, and it came out the other end with no big problems found. So couple things need to get fixed; but once they are, yay. And Leo?

**Leo:** Yes. Okay.

**Steve:** Let's take our last break and then, finally, we are going to get to...

**Leo:** I've been waiting all day for this.

**Steve:** The unhyphenated Device Bound Session Credentials.

**Leo:** Well, it's about time.

**Steve:** And people may be a little glad that what we've done so far has been a little fluffy by comparison because you're going to need to have conserved your strength for what's coming.

**Leo:** I think all my session credentials are device bound, but what do I know? Let's find out.

**Steve:** None of them are.

**Leo:** None. All right. What are device bound session credentials, with or without their hyphen?

**Steve:** So as I said at the top, while I was scanning through recent events, I noted that Chrome had recently moved to 135 and Firefox went to 137. So I scanned through Chrome's mind-numbing list of things that had been fixed and added and changed. There were several truly new features added by the W3C, the World Wide Web Consortium, which Firefox and Safari are also echoing. The most interesting of them was something called "Device Bound Session Credentials," which is the soon-to-be-available feature that named today's podcast, obviously.

Once I understood what this was about, that it was right, and given that this new technology is intended to be an extremely secure replacement for an aspect of session cookies, not entirely, as we'll see, but the way you get them essentially, I knew we needed to update the record because session cookies would not, as they have been forever, not be long for this world, and that's a big deal that will change everything.

As we've had the discussion many times in the past, the entire model of the web is for a user client, typically an interactive web browser, to request some resource from the

Internet using a URL which contains the unique address of the requested object, unique Internet-wide. Somewhere there's something, the browser says "I want that." As a result of the browser's connection to it and then supplying the address of the requested object, a web server returns whatever it is that the browser requested, and then they may, and often do, disconnect. So when you think about it, it's to me incredible to consider how far we have stretched that simple basic query and reply model. We've created the modern Internet world with it. Browser, ask for something, a server somewhere sends it back, says here you go, disconnects.

This original model, the thing that Sir Timothy John Berners-Lee first conceived of as the World Wide Web, never had any notion of a "session." That is, there was no way originally for anyone to logon to anything, since doing so would require that this "logged on" state would be saved somewhere. And Tim's original idea was entirely stateless.

**Leo:** Interesting. I didn't realize that.

**Steve:** Yes. The "web" was just a mass of pages containing links to other pages. And that was it.

**Leo:** But that's very limited.

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

**Leo:** Because you can't identify yourself.

**Steve:** All it was, was like a big knowledge base.

**Leo:** Just a blob; right.

**Steve:** A big directory, just like - and remember back then, Leo, like the original websites were like a list of links.

**Leo:** That's right.

**Steve:** They were just like link lists.

**Leo:** It was hypertext. That's hypertext.

**Steve:** That would take you, yeah, to somewhere else. So that...

**Leo:** Yeah. No memory, nothing. No state, nothing.

**Steve:** Right, right. All of that changed in June of 1994 when MCI asked Netscape to come up with some way for the user's browser to retain transaction data so that MCI would not need to retain it at their end.

**Leo:** Otherwise you have to log in every time you go to MCI Mail.

**Steve:** Actually, it's worse than that. Every query, you actually...

**Leo:** Oh, yeah. That's right.

**Steve:** There isn't - you can't actually log on.

**Leo:** You can't remember them. Yeah, I don't know who this is.

**Steve:** The server does not remember you.

**Leo:** Wow.

**Steve:** Ever. There's no memory of a previous query. And that's the way the 'Net originally was. So a Netscape engineer by the name of Lou Montulli came up with the idea of a web browser cookie that a web server would give to a visiting web browser. And every time thereafter, if the web browser contained a "cookie" that matched the domain that the browser was querying, the browser would voluntarily return that cookie token in all of its queries to the server.

**Leo:** So you save state locally on your machine. So the server doesn't have to do it. You re-identify yourself. By the way, the original name for this was Persistent Client-Side State Information. And it to this day irks me they didn't call them "pixies" instead of "cookies." It should have been a pixie.

**Steve:** Oh, that'd be much better.

**Leo:** Much better.

**Steve:** Yeah.

**Leo:** Although maybe it sounds a little scary that you have some pixies in your machine.

**Steve:** Well, and you really can't do that, you can't do pixie in that monster voice of yours.

**Leo:** Pixie. No, you do it - you do it in this voice.

**Steve:** Oh, that's good. Okay, so believe it or not, Leo, even back then, when this was first introduced, it was somewhat controversial.

**Leo:** Oh, really. Wow.

**Steve:** It suddenly meant that not every query from a browser was independently and entirely anonymous as they originally were. But by the same token, if you'll pardon my pun, the web server would usually have the browsing user's IP address. Still, people were aware of this back in the mid '90s, that a cookie, suddenly you lost a little bit of the anonymity that you had previously enjoyed.

Now, through the years, the cookie specification was formalized, and many new features were added. You know, expiration of cookies and other various flags. Many years ago we talked about the Firesheep hack, where HTTPS was only briefly used during login to a website, like Facebook, after which the connections would drop back to less compute-intensive plaintext HTTP. The trouble was that this exposed the user's "session cookie," which is how the user was logged in, how the user's interaction with the remote Facebook server kept being reidentified as being them. That was the only way remote servers had to recognize a user's repeated activities because all web queries stand alone otherwise.

So if a bad guy were to sniff a cookie, they could instantly impersonate that logged-in user. And they could because the traffic was just plaintext. Anybody looking at plaintext - and, you know, I remember doing it in my local Starbucks. I didn't log in as a person, but I saw a whole column down the right-hand side of the other people at Starbucks whose authentication tokens my browser had just sniffed.

So this obvious flaw was fixed, for example, by switching to always keeping all traffic encrypted using HTTPS, as we do now. As we know, virtually the entire Internet has switched to always-on HTTPS. But if a browser ever even once made the mistake of issuing an HTTP query to a remote server, whatever cookies it might be carrying for that server's domain would still be sent in the clear. So the formal cookie specification was again tweaked so that the server who's setting the cookie could set a "secure" flag with a cookie. This would instruct the browser to never send the cookie over any unencrypted HTTPS query. So today, all responsible cookie setting now also uses the "secure" flag to prevent any cookie leakage.

But if you stand back for a moment and consider how much work we're asking these poor old original cookies to do for us, and how much more technology we have readily available to us today than we did 31 years ago back in 1994 - especially our lovely crypto technology today - the need to replace these trusty and crusty old cookies, which are just dumb pseudorandom bits of gibberish, with something far more powerful, resilient, and resistant to abuse, it's hard to resist. And today it's something we can do easily. That session cookie replacement is now on the horizon, it's everything it could be, and it's called "Device Bound Session Credentials," or DBSC for short. And it actually does a lot more than cookies ever could.

Okay. So what are Device Bound Session Cookies? The World Wide Web Consortium's (W3C's) public GitHub page, part of which I'm going to share, is quite dense and quite matter of fact. But don't worry if some of this is initially confusing and flies over your head. It'll be flying over most of our heads. This is enough of a change from the way things have always been done for the past 31 years that it will likely take another

podcast or two for all of what this means to sink in. We'll all get there together. I'm sure we'll be going back to this multiple times in the future.

**Leo:** So this is going to be a cookie replacement. Is this going to be implemented for sure?

**Steve:** Yes. It is already on, you know, Safari, Firefox, and Chrome are all working on it right now. And it is in, well, Firefox, Safari and Firefox have it. And Chrome got it with 135, with the update that just happened.

**Leo:** That's hysterical. Because what are we going to do about all the cookie banners that we have to click through? Are we going to have DBSC banners?

**Steve:** Yeah, it's going to be a mess.

**Leo:** Yeah.

**Steve:** Okay. So here's what the W3C considers to be their "explainer." And I'll take a break here because at one point what they're saying becomes more clear. So I'll end up explaining what's going on. So they write: "Device Bound Session Credentials aims to reduce account hijacking caused by cookie theft. It does so by introducing a protocol and browser infrastructure to maintain and prove possession of a cryptographic key.

"The main challenge with cookies as an authentication mechanism is that they only lend themselves to bearer-token schemes." Okay, that meaning where the browser is the bearer of and holder of a token, which is useful, but there's a lot it can't do. So this says "they only lend themselves to bearer token schemes. On desktop operating systems, application isolation is lacking, and local malware can generally access anything that the browser itself can, and the browser must be able to access cookies. On the other hand, authentication with a private key allows the use of system-level protection against key exfiltration."

In other words, if we think about TPM, and we think about having a private key and proving that we have it by signing a challenge, and someone verifies our signature with our public key, that is, if we take this to a whole 'nother level, all of these other mechanisms exist today, and we've not been using them for the past 31 years.

So they said: "DBSC offers an API for websites to control the lifetime of such keys, behind the abstraction of a session, and a protocol for periodically and automatically proving possession of those keys to the website's servers." Now, I should explain that, as I'm reading this now, because I understand what it is doing, this all makes sense to me. The first time I read it, I was like, what? Okay. So this is the first time everyone's hearing it, so I understand. You're having my reaction. It's like, what? Anyway, this is going to get clear.

So they said: "There is a separate key for each session, and it should not be possible to detect if two different session keys are from one device." That's for privacy's sake. "One of the key goals is to enable drop-in integration with common types of current auth infrastructure. Meaning the rest of the world doesn't have to change to incorporate this. By device-binding the private key, and with appropriate intervals of the proofs, the browser can limit malware's ability to offload its abuse off the user's device, significantly

increasing the chance that either the browser or server can detect and mitigate cookie theft." In other words, cookies are going to still exist, but they're going to be short-lived, and the key is not in the browser. The key is in the device.

**Leo:** Uh-huh. So this eliminates that whole Firesheep thing of I got into your thing, I stole your Facebook cookie, and now I can logon as you on my machine because it's device bound.

**Steve:** Correct.

**Leo:** That makes sense. Although haven't we fixed that with HTTPS?

**Steve:** No. All that is, is the communication.

**Leo:** Right.

**Steve:** It isn't the authentication.

**Leo:** So it prevents somebody from getting in and stealing the cookie. But if they could still get the cookie, it would still be good.

**Steve:** Right.

**Leo:** Got it.

**Steve:** But this periodically re-authenticates, requires that cookies be re-authenticated.

**Leo:** To the device, yeah.

**Steve:** To the device. So if someone takes them elsewhere, they can't use them for long. And if there's any question about them, then reauthentication can be required. Anyway, so this says: "DBSC is bound to a device with cryptographic keys that cannot be exported from the user's device under normal circumstances. This is called 'device binding'" - unfortunately it's not hyphenated - "in the rest of this document.

"DBSC provides an API that servers can use to create a session bound to a device, and this session can periodically be refreshed with an optional cryptographic proof the session is still bound to the original device." Which I didn't understand the first time I read it, but it'll get clear in a minute. "At sign-in, the API informs the browser that a session starts, which triggers the key creation. It then instructs the browser that any time a request is made while the session is active, the browser should ensure the presence of certain cookies. If these cookies are not present, DBSC will hold network requests while querying the configured endpoint for updated cookies." Now, okay, let me stop because I didn't understand what the heck they were talking about the first time I read that. Now I get it.

So we're going to log into a service. So with DBSC present, after the user authenticates themselves with a browser on a device, there is now a new API that causes the device's DBSC public key to be sent to the remote server, to the website server. So as part of the user authentication on the device, the DBSC public key is sent to the remote server. That's what it uses then to reauthenticate the user whenever necessary. And we also now need to think of, not just a web server, but an authentication side of the server. That is, there's sort of an asynchronous separate authenticator on the website that is running adjacent to the regular website.

So what happens then is the website tells the browser, you need to have session cookies, and you're not sending me any session cookies. So the browser then queries this new authenticating portion of the site through an API and says I need updated session cookies. Please challenge me. So that authenticating side sends a random blob to the browser. The browser uses like the system's TPM, the Trusted Platform Module, that maintains a private key that never leaves, that cannot leave, to sign that challenge. The blob is a challenge that has never existed before, never exist again. And it'll just be an always increasing random number. Doesn't matter. Just has to be unique. And that's a good way to get it unique. It signs it and sends it back signed. So that proves to the authenticating portion, this DBSC authenticating portion, that it's still in communication.

This browser is on the device that originally logged in because that's the only way that it could sign a challenge using the private key that exists only on that device. And having performed that, successfully performed that cryptographic challenge, that authenticating portion, the new authenticating portion of the website then sends new, fresh, but short-lived session cookies, old-school cookies, to the browser. Which the browser then returns to the regular website, saying hey, look, it's me. And I've just re-proven who I am. And so the website says, oh, good, okay. Now we can proceed.

So, and that's where in what I just read it said: "If these cookies are not present, DBSC will hold network requests, meaning keep them pending, like not answer them, while querying the configured endpoint for updated cookies." So it goes through all that to get the updated cookies. Then it's able to provide them, and we proceed.

So they wrote: "DBSC's goal is to reduce session theft by offering an alternative to long-lived cookie bearer tokens" - that's what we've always had up until now - "that allows session authentication that is bound to the user's device. This makes the Internet safer for users in that it is less likely their identity is abused, since malware is forced to act locally and thus becomes easier to detect and mitigate. At the same time, the goal is to disrupt the cookie theft ecosystem and force it to adapt to new protections.

"DBSC's primary threat model is that of an attacker who can read and tamper with the user agent, such as with a malware-compromised browser" - or like, for example, bad extensions in your browser - "in which the malware can read and modify browser memory and secrets stored on disk. In many operating systems, malware may be able to obtain privileged (root, kernel, et cetera) access. DBSC aims to address this threat by establishing a cryptographic protocol in which secrets can be stored in dedicated systems (such as secure enclaves), though DBSC does not specify how implementers should store, backup, or sync keys as long as such storage is robust against the described threat.

"As a secondary consideration, DBSC also mitigates against certain types of network and server compromise, such as network Attackers-in-the-Middle (where an attacker can read or modify network traffic) or HTTP server log leaks (where a server mistakenly logs full HTTP request and response headers to logs which can be read by unprivileged insiders)." And of course if they had full headers they would be seeing the cookies that are being transacted.

"In all of these scenarios, DBSC aims to enforce the specific constraint that temporary read/write access to a user agent or network traffic does not enable long-lived access to any established DBSC sessions. For example, if an attacker has malware running within a victim browser process, they should be unable to continue to authenticate as the victim browser once that malware has been removed. (Note, however, that the definition of 'long-lived' depends upon the configuration refresh period. Within that period, attackers may continue to have short-lived access to any established sessions.)"

And the reason for that is we're still using cookies. And the reason we're still using cookies is that it's still too expensive to use this crypto all the time. I mean, it's important to understand what an insane number of queries our browsers are generating. I mean, it's just a flood of queries coming out of our browsers. They cannot be each individually cryptographically authenticated every time. It's still too expensive. So the idea is we're going to compromise. We're going to be able to periodically reauthenticate short-life cookies. And, importantly, before something critical is done, like acknowledging a funding transfer, or confirming a purchase or something. It's absolutely practical to ask for an updated reconfirmation of the device's authentication. So on an interactive level, we certainly have the speed to do that.

And so a compromise has been necessary. The previous approaches to replace cookies for binding sessions have failed because they were unwilling to make a compromise, and it's just too expensive. So this is a nice solution. And the other important aspect of this is that most of the website doesn't need to change. Most of the website, all of the website that is not about DBSC, it just sees session cookies, so it's got everything it's always had. We're only adding a new authentication slice to the overall site.

So they said: "What are the non-goals? DBSC will not prevent temporary access to any browser sessions while the attacker has ongoing access to a compromised user-agent." Right, because we're still, you know, we're still using cookies, but not long. "An attacker with ongoing access to a compromised user-agent (or decrypting middlebox, et cetera) will be able to continuously access fresh DBSC-controlled bearer tokens [cookies]; and an attacker with malware running on a compromised device will, on many modern operating systems, be able to treat even secure elements as a signing oracle" - meaning able to get it to sign on their behalf - "in order to provide proof-of-possession of the DBSC secret keys." So again, as do all modern security protocols, they clearly outline, these are the things we do. These are the things we know we don't do. And we're not claiming to be able to do everything.

So they said: "So what makes Device Bound Session Credentials different?" And they wrote: "DBSC is not the first proposal towards these goals, with a notable one being Token Binding. This proposal offers two important features that we believe makes it easier to deploy than previous proposals. DBSC provides application-level binding and browser-initiated refreshes that can make sure devices are still bound to the original device.

"For websites, device binding is most useful for securing authenticated sessions for users. DBSC allows websites to closely couple the setup of bound sessions with user sign-in mechanisms, makes session and key lifetimes explicit and controllable, and allows servers to design infrastructure that places verification of session credentials close to where user credentials (cookies) are processed in their infrastructure.

"Other proposals have explored lower-level APIs for websites to create and use protected private keys, for example via Web Crypto or APIs similar to WebAuthn. While this works in theory, it puts a very large burden on the website to integrate with. In particular, since the cost of using protected keys is high, websites must design some infrastructure for collecting signatures only as often as needed.

"This means either high-touch integrations where the keys are only used to protect sensitive operations like making a purchase, or a general ability to divert arbitrary requests to some endpoint that collects and verifies a signature, then retries the original request. The former doesn't protect the whole session and violates the principle of secure by default, while the latter can be prohibitively expensive for large websites built from multiple components by multiple teams and may require non-trivial rewrites of web and RPC frameworks."

Finally they said: "DBSC instead allows a website to consolidate the session binding to a few points: At sign-in, it informs the browser that a session starts, which triggers the key creation. It then instructs the browser that any time a request is made while that session is active, the browser should ensure the presence of certain cookies. The browser does this by calling a dedicated refresh endpoint specified by the website whenever such cookies are needed, presenting that endpoint with a proof of possession of the private key. That endpoint in turn, using existing standard Set-Cookie headers, provides the browser with short-term cookies needed to make other requests."

Okay. So again, there we finally get some sense for what's going on. Many previous efforts, as I said, to replace cookies have been proposed. None have taken hold. This one demonstrates a carefully crafted compromise.

Rather than constantly and continually using expensive public key crypto to prove its identity, DBSC sets up a secondary, essentially a "cookie supplier" for a website. The website tells the browser which cookies it needs to be providing. If the browser doesn't have those, or if they're near expiring, then, and only then, it separately connects to the "cookie supplier," where it uses rigorous state-of-the-art crypto to authenticate its device - not its browser, not its user, its device to the hardware, I mean, the device's hardware to the website's cookie supplier. Having done so, the cookie supplier returns regular old-fashioned cookies, which the browser will then use when subsequently transacting with the main website's pages.

The explainer continues, saying: "This provides two important benefits. First, session binding logic is consolidated in the sign-in mechanism, and the new dedicated refresh endpoint. All other parts of the website continue to see cookies as their only authentication credentials. The only difference is that those cookies are now short-lived. This allows deployment on complex existing setups, often with no changes to non-auth-related endpoints. And second, if a browser is about to make a request where it has been instructed to include such a cookie, but doesn't have one, it defers making that request until the refresh is done.

"While this may add latency to such cases, it also means non-auth endpoints do not need to tolerate unauthenticated requests or respond with any kind of retry logic or redirects. This again allows deployment with minimal changes to existing endpoints." They said: "Note that the latency introduced by deferring of requests can be mitigated by the browser in other ways, which will be discussed later."

And, interestingly, under "TPM Considerations," you know, Trusted Platform Module, they wrote: "DBSC depends on user devices having a way of signing challenges while protecting private keys from exfiltration by malware. This usually means the browser needs to have access to a Trusted Platform Module on the device, which is not always available. TPMs also have a reputation for having high latency" - meaning they're not fast - "and not being dependable. Having a TPM is a requirement for installing Windows 11, and can be available on previous versions. All our studies are for public key cryptography using Elliptic Curve DSA_P256 algorithm.

"Chrome has done studies to understand TPM availability to understand the feasibility of secure sessions. Current data shows about 60%, and currently growing, of Windows

users would be offered protections. Studies have also been done on the current populations of TPMs, both for latency and predictability. Currently the latency for signing operations averages 200ms" - so one-fifth of a second - "with only 5% of signing operations exceeding 600ms, and the error rate is very low, currently around 0.001%." And if you got an error you just retry.

"Based on this research, TPMs are widely available, with a latency and consistency that is acceptable for the proposed usage." And as we know, TPMs of the future having some crypto engine as part of every device is absolutely the future. So the spec is here. We already have 60% coverage. And that's only going to be going up over time.

So they ask: "What about privacy considerations?" They said: "An important high-level goal of this protocol is to introduce no additional surface for user tracking. Implementing this API for a browser or enabling it for a website should not entail any significant user privacy tradeoffs. There are a few obvious considerations to ensure we achieve that goal.

"Lifetime of a session and key material: This should provide no additional client data storage, for example, a pseudo-cookie. As such, we require that browsers MUST clear sessions and keys when clearing other site data like cookies." So, like, no DBSC residual will outlive cookie life. "Cross-site/cross-origin data leakage: It should be impossible for a site to use this API to circumvent the same origin policy and similar cookie policies. Implementing this API should not meaningfully increase the entropy of heuristic device fingerprinting signals." Right? So you're not, I mean, they're designing this very much with the state of the art of privacy in mind.

"This API, which allows background 'pings' to the refresh endpoint when the user is not directly active, must not enable long-term tracking of a user when they've navigated away from the connected site." That's a very good point because there is a new communications protocol set up between the browser and the refresh endpoint to obtain updated cookies. But that only needs to be happening while the user is actively looking at that tab on that site.

"Each session has a separate new key created, and it should not be possible to detect that different sessions are from the same device." So the keys are all isolated. "Registration and refresh will only be performed over a secure connection, or with localhost for testing." They said: "To achieve these goals, we add the following constraints to DBSC requests: Registration and refresh are made in the context of the request that triggered them. For registration, this is the request serving the Sec-Session-Registration header. For refresh, this is the request deferred due to missing cookies." They said: "Cookie refresh only occurs if the cookie is accessible. DBSC will not attempt to refresh a third-party cookie if the third-party cookies are blocked. And proactive refreshes must only occur if any tab has a page from the site currently loaded."

And then lastly: "While DBSC addresses a general problem of session hijacking and can be applicable to any browser consumer, it is possible to expand this protocol to better support enterprise use cases. By adding specifics to key generation, we can provide a more secure environment for enterprise users. This is the goal of DBSC(E), which is an extension to DBSC. The high-level design of DBSC(E) is described in the DBSC(E) Overview. DBSC(E) removes the vulnerability DBSC has, where a malware, if already present in the device during the key generation, can potentially take over a session. DBSC(E) proposes to mitigate this vulnerability by introducing device key chaining."

Okay. So I am fully aware that what we've just done was a lot to digest. And we're at the end of a lengthy podcast with no time to dig further into this. But at least the essence of this new system is probably now clear: Cookies still exist, but they are short lived, rather than persisting, as they often do these days, essentially forever. I mean, I can't

remember the last time I logged into many services that I use every day or two. They are staying current.

As cookies near what will now be their shorter end of life, the browser will be able to ping a website, a newly defined website endpoint, meaning, you know, something that is part of the specification, where it'll be, you know, some dot name directory off of the root, where a specific service, newly defined service will always be available if DBSC is supported. The browser will be able to ping that at any time separately in order to obtain a refresh of the cookies that are about to be expiring, and at that time reauthenticate its device to that remote site.

So to do this, that authenticating endpoint will send a cryptographic challenge that the browser must sign and return, and the browser can only do so using an unexportable private key that's buried in the hardware of the device that the browser is running on top of. The only thing that can be done with that key is signing cryptographic challenges to prove that the device has the key. Once the browser returns the challenge properly signed, the cookie provider will refresh the cookies for the domain, and the browser will then continue to be able to use the original website without trouble.

The cleverness of this solution is that it minimizes the changes that are required for the rest of the website by concentrating the new authentication scheme in one location. And by using shorter lifetime old-school cookies, it achieves compatibility with existing systems while also using the cookies as a form of short-term identity cache so that the system's far, far slower crypto hardware is not overwhelmed and is only needed to occasionally refresh the cookies.

Chrome, Firefox, and Safari have all added support for Device Bound Session Credentials to their web browser offerings. So now people, websites, researchers can begin experimenting with this and start bringing this onboard. And I'm sure we'll be talking about this more in the future.

**Leo:** Is it a done deal? I mean, is this for sure what's going to happen?

**Steve:** It requires adoption, like anything else.

**Leo:** Yeah.

**Steve:** I mean, you were saying on MacBreak Weekly that you wished Passkeys - or maybe it was on our podcast - that, you know, you wished Passkeys had more adoption than they do. But recent surveys show less than half of people are using anything other than username and password.

**Leo:** Yeah. Yeah. Yeah.

**Steve:** So, you know. So it has to be in the browser. It has to support a TPM. That's the first step. Then it's up to the web server...

**Leo:** The sites; right.

**Steve:** ...to decide that it wants to adopt it.

**Leo:** Right.

**Steve:** And so it'll be like, you know, you know all the extra hoops you have to jump through if your financial advisor sends you email, and you've got to authenticate, or your bank is making you do extra stuff.

**Leo:** Right.

**Steve:** It'll be places where they really, really care about knowing that you're using a particular device.

**Leo:** Right.

**Steve:** But what's cool is, once you create a binding, as they call them, a binding between the private key in a device and a remote entity like a bank or your domain name supplier, like I would like to have much stronger authentication between the computer I'm sitting at and Hover.com.

**Leo:** Right. Right.

**Steve:** And so we have never had a mechanism to offer that. This offers that.

**Leo:** Yeah.

**Steve:** When I am sitting up my account at Hover, they could query this, get the public key for the private key in my device, and that would be part of my Hover account. And then anytime in the future they could require me to be sitting at this computer in order to authenticate to Hover.com.

**Leo:** Or they could say, well, you're at that computer, so you don't have to go through the extra multifactor authentication or something; right?

**Steve:** Correct. Correct.

**Leo:** Because right now with Hover I have to do multifactor every single time I log on.

**Steve:** Exactly.

**Leo:** And so it kind of makes sense that sites that do have this higher need for security might adopt it first. I'd love it if my bank adopted this. That'd be fantastic; right?

**Steve:** Yes, yes. And essentially it would be extremely good for short-term reauthentication of a device. You are at this device. Because we just gave you something, and your device signed it for us, and only that one device in the galaxy could do so.

**Leo:** Very - I think this sounds like a good idea.

**Steve:** We need it. And so...

**Leo:** And this is no effort on the user's part. The user might not even be aware of it.

**Steve:** You would never see it.

**Leo:** Yeah.

**Steve:** It would be completely transparent.

**Leo:** Love it.

**Steve:** It might say, you know, we just authenticated your device. You're done.

**Leo:** You wouldn't need more CAPTCHAs. You can get rid of those CAPTCHAs. You could reduce the number of MFA logins. You know, Hover could say it once, put that special cookie on my hard drive, and then I wouldn't need to do it again on that device. I think that's...

**Steve:** Right. Actually, Hover would receive the public key for this feature on your device.

**Leo:** Right.

**Steve:** And that's all they would ever need. It would be part of your account.

**Leo:** You still would want to log in. I think they would still want a password and a login.

**Steve:** Yes. Yes. In order to authenticate that it was you at that device.

**Leo:** On your device; right, exactly.

**Steve:** Yes. But this allows cryptographic binding of device to remote account.

**Leo:** I think this is good.

**Steve:** It's a good thing.

**Leo:** I'm glad they're implementing it, yeah.

**Steve:** Yeah.

**Leo:** Did this come from the IETF? W3C? Was this...

**Steve:** W3C, and it's in all three browsers. It's in Safari, Firefox, and Chrome. And now all of our listeners know about it.

**Leo:** I mean, and that presumably means it's in all the Chromium derivatives like Edge, Brave...

**Steve:** Yeah. And it's because it was just added to Chrome 135 that we're talking about it today.

**Leo:** Yeah. Great. You know what? This wasn't so bad. This was great. As always, Steve makes it clear. And I tell you what, that's why you listen to this show because it keeps you up to date on these kinds of things. I really appreciate that, Steve. I don't think - I doubt there's any other podcast in the world that has spent any time on Device Bound Session Credentials at all. We're the first, and we'll probably remain that way. This is why we listen every Tuesday, right about 1:30 p.m. Pacific, 4:30 Eastern. And we'll see you next week on Security Now!.

**Steve:** Thanks, buddy. Bye.