GoFetch

Description: After I comment on U.S. Department of Justice's antitrust suit against Apple, we'll update on General Motor's violation of its car owners' privacy and answer some questions, including what happy news is Super Sushi Samurai celebrating? Has Apple abandoned its plans for HomeKit-compatible routers? And what appears to be shaping up to take their place? Will our private networks be receiving their own domain names? And if so, what? The UN has spoken out about AI. Does anyone care? And what do I think the prospects are of us controlling AI? What significant European country just blocked Telegram? What did the just-finished 2024 Pwn2Own competition teach? Might the U.S. be hacking back against China as they are against us? And after a bit of interesting SpinRite news and a bit of feedback from our listeners, we're going to spent the rest of our time looking into last week's quite explosive headlines about the apparently horrific unfixable flaws in Apple's M-series silicon. Just how bad is it?

High quality (64 kbps) mp3 audio file URL: http://media.GRC.com/sn/SN-967.mp3
Quarter size (16 kbps) mp3 audio file URL: http://media.GRC.com/sn/sn-967-lq.mp3

SHOW TEASE: It's time for Security Now!. Steve Gibson is here. Lots to talk about. Most importantly, that Apple exploit that everybody said was unpatchable, the end of the world, Steve says not so fast. GoFetch our topic, next on Security Now!.

Leo Laporte: This is Security Now! with Steve Gibson, Episode 967, recorded Tuesday, March 26th, 2024: GoFetch.

It's time for Security Now!, yay, the time I look forward to all week. In this case, for the last three weeks. Thank you to Mikah Sargent for filling in. Steve Gibson, the man about town, is here to talk security.

Steve Gibson: Mikah did a great job.

Leo: Of course he did.

Steve: He held down the fort and was engaging and...

Leo: Good, I'm glad you like him because in about a year he's going to be in charge of the whole damn thing. I notice you're doing the Leonard Nimoy salute. Just want

to tell you it's Live Long and Prosper Day, Leonard Nimoy's birthday. Would be today, March 26th.

Steve: Yeah, and boy, he was born in '31, I think. So, you know? And last time we saw him he was looking it, too. But, you know, he and the old Captain Kirk are still...

Leo: Is Nimoy still alive? I thought he'd passed. He's still alive?

Steve: Oh, that's right, I remember, he did pass.

Leo: He passed, yeah.

Steve: Of course he did. Well, I haven't yet, and he and I have the same birthday.

Leo: Oh. You weren't born in 1931, however.

Steve: No, no, no, no. '55, baby.

Leo: Happy birthday. I didn't know that. Happy birthday.

Steve: Yeah.

Leo: Well, you are, so that means you're four months older than me.

Steve: Yes, I am. Well, and a couple years; right?

Leo: No, wait a minute. I'm November '56. You're March '55. So yes.

Steve: Correct.

Leo: One year and a few months. Okay.

Steve: Yeah.

Leo: Okay. So happy birthday. You going to do anything special to celebrate?

Steve: Thank you. We initially had some plans to go have a fancy dinner. But I said to Lorrie yesterday, I said, you know, I just would rather have a nice steak at home.

Leo: You get to do what you want.

Steve: So she's out picking up some beautiful - that's what she said, actually.

Leo: Yes.

Steve: You'd make a great wife, Leo.

Leo: Well, it should be, you know, you should be. When I was a kid, my mom, there was a birthday dinner that was the same every year my mom would make for us. And I looked forward to that. It was wonderful. So, yeah, happy birthday.

Steve: So we have a tremendous podcast today. Of course it's titled GoFetch, which is the name that's been given by the - I would call it the discoverers, but it's sort of the re-discoverers because they first stumbled onto this two years ago.

Leo: Oh, interesting. Oh.

Steve: And brought it up. And in fact my theory is it's the reason that the M3 chip has a switch, which M1 and M2 doesn't because...

Leo: Interesting.

Steve: Because they kind of scared Apple, but then they weren't really able to make a strong case. Well, boy, has that case been made now. And in fact we're going to start off when we talk about this here in an hour or so about how wound up the tech press has gotten and, you know, mis-wound because, boy, did they get it wrong. But we'll have some fun with that. And again, this is going to be one of our listeners' favorite types of episodes because it's going to be a deep dive. So get out your propeller cap beanies and wind them up because, by the time we're done, everyone is going to understand exactly what happened, why it happened, how it happened, what it means, and like you could go to a cocktail party and really put your friends to sleep.

Leo: Well, I've been saying, because we've been talking about it, obviously, on TWiT and today on MacBreak Weekly, and I've been saying, you know, I'm sure Steve will cover this much more accurately and much more granularly so tune in to Security Now! today. So I'm glad...

Steve: Everybody will know.

Leo: I didn't coordinate with you, I just figured, oh, he's going to jump into this one. So GoFetch.

Steve: I'm also going to jump in briefly, because I'm not a legal scholar or expert, just I have a couple things to say about the U.S. Department of Justice's antitrust suit against Apple. There are some arguments that they'll make that are security related. So it does impinge on us a little bit.

Leo: Yes, it's true.

Steve: But I just sort of have a little sort of an overview of that and, you know, capitalism and monopolies and so forth. We're going to update on General Motors. I don't know if you heard about this, Leo, this astonishing violation of their car owners' privacy.

Leo: Oh, boy.

Steve: Oh, boy. It's unbelievable. Also we're going to look at - we're going to answer the question, what happy news is Super Sushi Samurai celebrating today?

Leo: Okay. I don't even know what that is. Okay, good.

Steve: We're also going to look at whether Apple has abandoned its plans - you were talking about this at the end of MacBreak, actually - for HomeKit-compatible routers, and what appears to be shaping up to take their place. Will our private networks - oh, this is cool - be receiving their own domain names? ICANN has been busy. And if so, what is it? The UN has spoken out about AI. Does anyone care? And what do I think the prospects are of us controlling AI? What significant European country just blocked Telegram?

Also, what did the just-finished 2024 Pwn2Own competition teach us once again? Might the U.S. be hacking back against China as they are against us? I've long been bemoaning the fact that we never hear anything about the other direction. Well, we've heard something. And after a bit of interesting SpinRite update news and a bit of feedback from our listeners, as I said, we're going to spend the rest of our time looking into last week's quite explosive headlines about the apparently horrific unfixable toxic flaws in Apple's M-series silicon. Just how bad is it?

Leo: Okay. Good. And I've been saying don't worry, but we'll find out what the real expert has to say in just a little bit. I look forward to that.

Steve: And of course we do have a fantastic Picture of the Week, courtesy of our marvelous listeners.

Leo: A great life hack, I think, something everybody might want to adopt.

Steve: That's right.

Leo: So Picture of the Week time.

Steve: So for all of my life, Leo, I have found coat hanger wire to be really convenient.

Leo: So useful.

Steve: It is. And, you know, you used to get the coat hangers back from the dry cleaners with your shirts on them, and they'd have a little bit of a paper wrapping on them, but you can take that off. But that gauge of coat hanger is...

Leo: Perfect.

Steve: I mean, you can bend it into all kinds of useful shapes.

Leo: You can bend it. Stick it down the drain to hook a ring that somebody lost, or, yeah.

Steve: Exactly.

Leo: So useful.

Steve: Exactly. You know, it's just super handy. Well, now, here we have an application that I would not recommend.

Leo: Well, I think this is a great life hack. Don't you think everybody should do this?

Steve: Uh....

Leo: Let me show people what we're talking about here.

Steve: So somebody has a USB charging cable which is way too long. And maybe they need it to be long. But this is like a neatnik person. And I think we're seeing sort of a theme here because they've coiled up this way-too-long, I mean, it's like - this looks like 15 feet of USB charging cable. But, you know, you don't want it lying around on the floor; right? So they've coiled it all up. Now, okay. Now what are you going to do? You've got this coil of USB charging cable. You need to hang it somewhere. So, and it doesn't really - you can't really hang it on the charger because it'll fall off. It needs to be more secure than that. So this clever OCD person thought, hey, I've always found coat hanger wire to be really handy for making stuff. So, and happened to have a pair of pliers around. So basically fashioned this beautiful, I mean, by all measures this is a beautiful hook.

Leo: He spent some time with his little pliers there, bending and curving. It's gorgeous.

Steve: Yeah. It is great. And, boy, does it work as a hook.

Leo: Yes.

Steve: To hang this cord on.

Leo: Right around the prongs of that Apple 5-watt charger, just goes right around those beautifully.

Steve: Therein lies the problem.

Leo: Uh-oh. What?

Steve: I've never - I don't recall ever actually touching the leads of my ohmmeter to this wire. It must be that it is coated with some sort of an insulating varnish of some sort.

Leo: Is it? Is it, though? Is it?

Steve: Otherwise this would have already exploded because...

Leo: Notice, though, there's a switch here. I think that he has not switched it on yet.

Steve: Oh, boy.

Leo: And I think he's going to get quite a surprise.

Steve: Put your shoes on and use the toe of your shoe to turn this plug on.

Leo: Another point, because he looks like he really is, you know, OCD and careful, he has installed his plugs upside down, which...

Steve: It's true.

Leo: Right?

Steve: You've got the, yeah, not smiley little face.

Leo: Got the ground, they're not smiley.

Steve: They're not looking like a little happy face.

Leo: Yeah.

Steve: Yeah.

Leo: What's going on there?

Steve: It's not good.

Leo: Yeah.

Steve: So anyway, so just so people are thinking that we haven't completely lost our mind, the point is that this hook has two legs that go up behind this USB charger and then bend around, you know, in U shape, to hang over the two prongs of the AC plug.

Leo: Yeah. There you go.

Steve: Wire on wire.

Leo: It's like they made it for this.

Steve: Oh, it's beautiful. I mean, it is a beautiful construction. But no.

Leo: The minute you plug it in, what would happen? Would it heat up? Would it start to glow? Or would it actually short the thing out?

Steve: No, no. You would have an immediate - the good news is, all homes ever made, even when they had screw-them-in fuses in the fuse box in the basement, they had some cutout such that, if any circuit suddenly drew too much power, rather than it exploding in your face...

Leo: Right.

Steve: ...down in the basement something would go boom, and then you'd - now, of course, what you don't want is to run out of fuses because, you know...

Leo: Don't put a penny across it. No no no no no no.

Steve: That's right. Then what people did was like, oh, shoot, you know, I don't know why this fuse blew, but it's inconvenient. So I seem to be fresh out of fuses here.

Leo: Yeah, but here's some good news...

Steve: Stick a penny in the hole, we can stick a penny in the socket and screw the blown-out circuit, you know, the blown-out fuse on top. Anyway, yes, folks, do not do this at home. The only thing I can think, Leo, is that there must be some varnish on this. But...

Leo: Even then.

Steve: ...over time, as it's used, it's going to get moved back and forth, riding on the top of the prongs of this plug, and it's just going to explode at some point.

Leo: Never, ever put metal around the prongs of your plug. I learned that in eighth grade when a...

Steve: Leo, on that note, we do have the picture queued up for next week, and it's another goodie.

Leo: Good. Oh, no.

Steve: It's a variation. It's not the same. We don't want to get repetitious here. But we're going to have just as much fun with it.

Leo: Somebody told me that that is the commercially preferred way of installing a plug socket is upside down like that. And then somebody else in the Discord says that's how you know it's a switched circuit. I've never seen that before. But just to - reason I'm saying that, to preclude all the email that you and I inevitably will get from licensed electricians who will say, absolutely, that's the way to do it correctly.

Steve: Thank you for not overflowing my inbox.

Leo: I also hope that I am not the subject of the Picture of the Week next week because I installed yesterday, we had a little lighting problem, and I and my brother-in-law did a little electric work, electrical work installing a new under-counter lamp. And you see that switch right there, that's just to the right of Joe there? As he was installing the wires, he accidentally backed into it and switched it on and got a little bit of a shock.

Steve: Oh, yes.

Leo: Ow.

Steve: I see you're still wearing the avocado shirt from Sunday.

Leo: I am wearing - this was right after I got home Sunday. They said, "Get in here." And I am also wearing, as sharp eyes will see, the most useful device for a home handyman anywhere.

Steve: Yes. You have a head-mounted lamp.

Leo: A head lamp.

Steve: Yeah.

Leo: Please, do not make that the Picture of the Week next week. I'm just begging of you; okay?

Steve: Yes. Sometimes when Lorrie...

Leo: Oh, you've got one, too. Look at you.

Steve: Well, I've got, no, these are the magnifiers.

Leo: Oh, that's - and what do you wear that for, Steve, besides looking like an alien?

Steve: When I'm building things like this.

Leo: Oh, yes. You've got to get very close.

Steve: That's right.

Leo: Are you soldering with those on?

Steve: Those are little surface-mount components. They're little itty-bitty. So, yeah.

Leo: That's a lot of work. Anyway.

Steve: Anyway.

Leo: Take that off and let's continue.

Steve: Believe it or not, we have news to get to.

Leo: Yes, okay.

Steve: Last Thursday March 21st was - it was by all measures a rough day for Apple. Not only, as I mentioned, did the tech press explode with truly "hair on fire" headlines about critical, unfixable, unpatchable, deeply rooted cryptographic flaws rendering Apple's recent M-series ARM-based silicon incapable of performing secure cryptographic operations...

Leo: Incapable. Incapable.

Steve: Can't be done. Which is the topic we'll be spending the end of today's podcast looking at in some detail, once we get this thing started, because actually it's super interesting. But before that, also last Thursday, the U.S. Department of Justice was joined by 15 other states and the District of Columbia, which wishes it was a state, but isn't, in a lawsuit alleging that Apple has been willfully and deliberately violating Section 2 of the Sherman Antitrust Act.

Now, I'm just going to share five sentences from the DOJ's comments which were delivered last Thursday. They read: "As our complaint alleges, Apple has maintained monopoly power in the smartphone market not simply by staying ahead of the competition on the merits, but by violating federal antitrust law. Consumers should not have to pay higher prices because companies break the law." Okay. "We allege that Apple has employed a strategy that relies on exclusionary, anticompetitive conduct that hurts both consumers and developers. For consumers, that has meant fewer choices; higher prices and fees; lower quality smartphones, apps, and accessories; and less innovation from Apple and its competitors. For developers, that has meant being forced to play by rules that insulate Apple from competition."

Okay. Now, this is not, clearly, a podcast about antitrust law. We all know I'm not an attorney, nor am I trained in the law. So I have no specific legal opinion to render here. However, I've been a successful small business founder/owner/operator throughout my entire life. And I'm certainly a big fan and believer in the free enterprise system and in the principles of capitalism. But I also appreciate that this system of competition is inherently unstable. It has a natural tendency for the big to get bigger through acquisition and the application of economies of scale and leverage. That same system that creates an environment which promotes fair competition can be abused once sufficient power has been acquired.

Those of us of a certain age have watched Apple being born, then fall, only to rise again from the ashes. My own first commercial success was the design, development, production, and sales of a high-speed, high-resolution light pen for the Apple II which allowed its users to interact directly with the Apple II's screen. To my mind, there is no question that as a society we are all richer for the influence that Apple's aggressive pursuit of perfection has had on the world. Things as simple as product packaging will never be the same.

But for some time we've been hearing complaints about Apple's having taken this too far. It's understandable for competitors to complain, and to ask the government to step in and do something. At some point that becomes the government's very necessary role, just as we saw previously when the same thing happened with Microsoft, and some would argue ought to happen again with Microsoft. For many years the U.S. government has done nothing, while Apple has continued to grow and continued to aggressively use its market power to increase its shareholders' wealth. The question is, when does use of market power become abuse of market power? The next few years will be spent in

endless depositions and expert testimony working to decide exactly what sort of cage Apple needs to be constrained within.

One thing we know is that many of the arguments Apple will be making on its own behalf will involve security, the security inherent in its closed messaging system, the inherent security of its closed App Store. You know, things we've touched on many times in this podcast. Apple will allege that by keeping its systems closed, it is protecting its users from unseen nefarious forces. But, for example, the presence of Signal and WhatsApp in the App Store and on Apple devices, which create freely interoperable super-secure cross-platform messaging, suggests that Apple's own messaging technology could work similarly if they wished it to.

During the news coverage of this since Thursday, I've encountered snippets of evidence which suggest that the government has obtained clear proof of Apple's true motives where Apple's technology has been designed to support Apple's interests rather than those of its users. In any event, and, you know, maybe those are aligned. That's really the question; right? Are Apple's interests and its users' interests perfectly aligned?

Nothing is going to happen on this front for a long time. Years will pass, and this podcast will be well into four digits by the time anything is resolved with the DOJ's antitrust lawsuit. The way things have been going, it seems to me much more likely that the laws being written and enacted within the European Union today will be forcing Apple's hand long before the DOJ finishes making its case. All that may eventually be required will be for the U.S. to force Apple to do the same thing that they're already doing over in Europe, here as well. But as for whether Apple-designed silicon cannot perform secure cryptographic operations, that is something this podcast can speak to authoritatively, and we'll be doing so once we've caught up with some more interesting news and feedback.

Leo: Good. I always said back in the day, in fact it was during - it's funny how you began this with the good old days of Apple because back in the day when the Department of Justice was suing Microsoft, I always said, if Apple were as big and powerful as Microsoft, they'd be just as bad. But they aren't. In fact, they almost went out of business in '97. And now that they are even a little bit bigger than Microsoft, yeah, they're just as - it's what happens.

Steve: It is. It is exactly what happens. And it's not that anybody is a bad person. You know, I mean, they argue, the executives argue that it's their job to maximize shareholder wealth.

Leo: Right. That's capitalism.

Steve: Yes, exactly. Exactly. And so it's a fundamental property that there need to be constraints. And of course in the U.S. we have those. Boy, is it painful to get them. But, you know, it's interesting. And I think I heard Rene saying that he thought he was going to have to spin up another podcast in order to keep track of this.

Leo: Oh, I'm sure.

Steve: I'm not going to bother with that.

Leo: No, we're not going to do it, yeah.

Steve: No.

Leo: No.

Steve: It'll go on and on and on, and we'll mention it once in a while, and that'll be it.

Leo: Right. This thing will go for years, exactly as it happened with Microsoft

Steve: So last week we shared the difficult, I mean, truly difficult to believe, but true story that General Motors had actually been sharing - and by sharing I'm pretty sure the proper term would be selling - the detailed driving record data of its car owners, down to how rapidly the owner's car accelerated, how hard it braked, and its average speed from Point A to Point B. Leo, they literally have instrumentation in there that is monitoring everything the car does. And these cars are all interconnected now. It was all being beamed back to GM, who it turns out was selling it to LexisNexis, a major data broker.

Anyway, so what happened was - and this was a New York Times or Washington Post, I think it was The New York Times piece last week that just blew the lid off this. Some guy, I think he was in Canada, or maybe he was just up north, he saw his insurance go up 21% in one year, although he'd never been in an accident and didn't have tickets. And so when he asked his insurance company why, they sort of hemmed and hawed. He also tried to obtain alternate insurance, and all the quotes that he got back from competing companies were the same. Finally one of them said, well, you should check your LexisNexis report because it's a little worried about your driving habits.

Leo: Oh, so now there's like a credit report, there's now a car driving report.

Steve: Yes.

Leo: But you know what, in some ways, A, I'm not surprised. Insurance companies have for years offered good driver discounts. In the past you used to have an app and stuff. I'm not surprised to hear this. And honestly...

Steve: Optionally installed app.

Leo: Right.

Steve: For, like, low-mileage drivers where it would monitor - I'm sure we talked about it on the podcast.

Leo: Yeah, but this is good for you and me because insurers, instead of this guy who really is not a safe driver paying the same as you and me who drive like little old

men, because we are, we should get reduced; right? And he should pay more. It's fair, I think.

Steve: And should it be done without consent.

Leo: Well, in a way I bet you he did consent. I bet you there is somewhere a document that he signed when he bought that car that said data's being collected. You saw the Mozilla report last year. We talked about it, about how cars are a privacy nightmare.

Steve: Yeah, well, we were all wondering recently how your sexual habits were being recorded by a car. It's like, what? Like, is it monitoring the suspension?

Leo: Should this car be rockin', as you well know, don't you be knockin'.

Steve: Okay. So the good news is this produced an outcry which caused GM to immediately terminate this conduct. And no doubt threats of lawsuits were involved, too. They said GM is immediately stopping the sharing of this data with these brokers. The report said: "After public outcry, General Motors has decided to stop sharing driving data from its connected cars with data brokers. Last week, news broke that customers enrolled in GM's OnStar Smart Driver app have had their data shared with LexisNexis and Verisk. Those data brokers in turn shared the information with insurance companies, resulting in some drivers finding it much harder or more expensive" - exactly as you said, Leo - "to obtain insurance. To make matters much worse, customers allege they never signed up for OnStar Smart Driver in the first place, claiming the choice was made for them by salespeople during the car-buying process."

Leo: Yeah. And, you know what, it comes with the car. And, you know, it's good, it's all for your safety. That's why we put it in, so that if you get a wreck, you can press the OnStar button.

Steve: That's right.

Leo: That's why they did it.

Steve: And people will come.

Leo: Yes.

Steve: We're not Big Brother watching over you.

Leo: No, of course not.

Steve: Okay. So I saw this bit of happy cryptocurrency news that just made me smile. It seems that last week the blockchain game - I didn't know there was a blockchain game. But yes, someone has made a game out of blockchain, and it's called Super Sushi Samurai. Super Sushi Samurai had $4.6 million worth of its tokens stolen. However, it just reported that they had all been recovered. So what happened? They explained that the hack was actually the work of a security researcher who exploited a bug in their code to move the funds out of harm's way to prevent future theft.

Leo: Yeah, that was all. Yeah.

Steve: That's right.

Leo: Just want to move them out of harm's way.

Steve: Super Sushi Samurai described the incident as a "white hat rescue" and has ended up hiring the white hat to be a technical advisor. So that's what I call a G-rated happy ending.

Leo: Okay.

Steve: Yeah.

Leo: I believe it. Why not?

Steve: And also you guys touched on this on MacBreak. Apple Insider has some interesting coverage about Apple's apparently failed initiative to move their HomeKit technology up into home routers. I was a fan of this, since it promised to provide router-enforced inter-device traffic isolation, and the only place that can really be accomplished is at the router. Our listeners know that I've been advocating for the creation of isolated networks so that IoT devices would be kept separate from the household's PCs. But what Apple proposed five years ago, back in 2019, would have additionally isolated each IoT device, like with that level of granularity, from all the others. So here's what Apple Insider explained.

They said: "Apple's HomeKit Secure Routers were announced in 2019, but were never really taken up by manufacturers. And now some vendors are claiming Apple is no longer pursuing the technology." And we'll get to why in a minute. "HomeKit Secure Routers," they wrote, "were introduced by Craig..."

Leo: Federighi.

Steve: Federighi. I know his name. The problem is, you know, I'm a big Star Trek person, and I want to say Ferengi. Which, you know...

Leo: Craig Ferengi over there at Apple headquarters.

Steve: Had to stop from saying it. I was, it's not Ferengi. Come on, Steve.

Leo: That's pretty funny. I would never have guessed that. Wow.

Steve: So "...Craig Federighi at World Wide Developer Conference 2019, and in the same breath and at the same time they introduced HomeKit Secure Video. The latter, that is, HomeKit Secure Video, took time to reach the market, but it was used, and manufacturers adopted it, even if others would not."

Okay. Now, "During [this year's just happened] CES 2024, two router vendors separately told AppleInsider that Apple is no longer accepting new routers into its program. If that claim is correct and it probably is, since it came from the same rejected manufacturers given the lack of HomeKit Secure Routers on the market" - that is, in five years not much happened - "it appears that Apple has abandoned the idea, even though Apple still has active support pages on the matter. However, Apple Insider noted that it also has support pages on AirPort routers too, and those are," as they put it, "dead as a doornail."

Leo: Those really are dead, yeah, yeah. I was so excited that Apple would offer this security standard, that we could have some confidence in the security and, frankly, firmware updatability of our routers. It's a little disappointing to me.

Steve: Yeah. Anyway, it's not going to happen. They backed out. Apple Insider, to make a long story short, polled the various routers that Apple listed. There is one Linksys Velop AX4200 and an AmpliFi Alien router are apparently the only two that are currently listed by Apple as being supported. Eero has a notice saying that its Eero Pro 6E and 6+ do not support Apple HomeKit, and they have no plans to offer Apple HomeKit router functionality.

Anyway, so not everything that gets announced happens. And asking router manufacturers to modify their firmware to incorporate the required HomeKit functionality, and it appears that it may have taken some significant customization, it was just never going to get off the ground.

And this is probably for the better since it appears that we have already - and oh, thank god, blessedly quickly - moved beyond disparate proprietary closed IoT ecosystems. Which, you know, is where it looked like we were headed with Amazon Alexa and Apple's HomeKit and Google's Home and Samsung's SmartThings, all creating their, oh, let's do our own thing. All the buzz appears to now be surrounding the interoperability technology known as "Matter." This was formerly known as CHIP, which stood for Connected Home over IP. It's now been rebranded as Matter, and everyone appears to be seeing the light. Nobody wants to be left out.

All those guys I just mentioned - Amazon with Alexa, Apple with their with HomeKit, Google with Home, and Samsung's SmartThings - all have announced and are supporting Matter. It's now at v1.2, open, open source, license free. Anyone can create Matter-compatible devices. If they follow the spec, they will interoperate. And more than 550 companies have announced their commitment to Matter. So this is done; right? I mean, all of the biggies are going to be supporting Matter. They really have no choice. And at this point I wanted to make sure I brought it up because I wouldn't purchase something, you know, that random AC plug that I got for a shockingly, I don't know, it was $4 or something. It's amazing. How can this be an Internet-connected device? Just like, what, the plastic and the prongs would cost $4.

Leo: It does report back on your driving habits, however.

Steve: Oh, it's got a little eyeball in it that follows you around the room.

Leo: Yeah.

Steve: It's kind of freaky.

Leo: Does Matter have, though, I mean, the thing about Apple's HomeKit router standard was it had security requirements built in. Does Matter have something like that?

Steve: That's what they were, you're right, that's what they were going to produce.

Leo: Yeah.

Steve: I think Matter is about interconnectivity.

Leo: Right. Right.

Steve: Which is not to say it couldn't be made more secure, but that's not their focus.

Leo: Right.

Steve: Thank you, my friend. Okay. In a cool bit of news, ICANN, the Internet Corporation for Assigned Names and Numbers, is going to make an assignment. It's in the process of designating and reserving, get this, a top-level domain specifically for use on private internal networks. In other words, our 10-dot and 192.168-dot networks, and there's a 17.16 thing in there, too, will be obtaining an official TLD of their own. So localhost may soon be less lonely. Here's the Executive Summary which explains and lays out the rationale behind ICANN's plans.

They wrote: "In this document, the SSAC" - that's the Security and Stability Advisory Committee because, you know, that's what you want in your Internet is some security and stability advising. They recommend "the reservation of a DNS label that does not and cannot correspond to any current or future delegation from the root zone of the global DNS," which is the very long-winded way of saying we're going to get our own dot-something TLD.

They said: "This label can then serve as the top-level domain name of a privately resolvable namespace that will not collide with the resolution of names delegated from the root zone." That is, you know, the public DNS root zone. "In order for this to work properly, this reserved private-use TLD must never be delegated in the global DNS root. Currently, many enterprises and device vendors make ad hoc use of TLDs that are not present in the root zone when they intend the name for private use only. This usage is uncoordinated and can cause harm to Internet users." Oh, my.

"The DNS has no explicit provision for internally-scoped names, and current advice is for the vendors or service providers to use a sub-domain of a public domain name for internal, or private use. Using sub-domains of registered public domain names is still the best practice to name internal resources. The SSAC concurs with this best practice and encourages enterprises, device vendors, and others who require internally-scoped names to use sub-domains of registered public domain names wherever possible. However, this is not always feasible, and there are legitimate use cases for private-use TLDs."

And I'll just note that, you know, for example, an individual could register a domain with Hover, who I don't know if they're still a sponsor of the TWiT Network. They are still my domain name provider. I moved everything away from Network Solutions...

Leo: I agree.

Steve: ...once it became clear...

Leo: I don't think they're a sponsor anymore, but we still love them.

Steve: Yup. They're the right guys. Anyway, so, you know, Johnny Appleseed, you could get that. Of course you can't get dot Johnny Appleseed, so that wouldn't work. But you could get, you know, a dotcom or some inexpensive subdomain of some established top-level domain and just use that for your own purpose. Because you have that subdomain, nobody else is going to be able to use it publicly. So you're safe. So that's what these guys were saying.

So they continue: "The need for private-use identifiers is not unique for domain names, and a useful analogy can be drawn between the uses of private IP address space and those of a private-use TLD. Network operators use private IP address space to number resources not intended to be externally accessible, and private-use TLDs are used by network operators in a similar fashion. This document proposes reserving a string in a manner similar to the current use of private IP address space. A similar rationale can be used to reserve more strings in case the need arises." Okay. So they go on and on.

Anyway, finally, after all the bureaucrat boilerplate has settled down, ICANN wrote: "The Internet Assigned Numbers Authority (IANA) has made a provisional determination that '.internal' should be reserved for private-use and internal network applications. Prior to review and approval of this reservation by the ICANN Board, we are seeking feedback on whether the selection complies with the specified procedure from SAC113" - more bureaucracy - "and any other observations that this string would be - to verify that it would be an appropriate selection for this purpose." So it's all but certain that .internal will be reserved and will never be used for any public purpose, and therefore it would be safe for anyone to start using it for any internal purpose.

Leo: That's nice to have, thank you.

Steve: Very cool, .internal. I saw some commentary saying, well, it only took 30 years.

Leo: That's true, yeah.

Steve: Yeah, that is true.

Leo: Took them a little while.

Steve: Okay. So last Thursday, as I said earlier, was a very busy day. Not only did the DOJ announce their pursuit of Apple, and Apple's M-series silicon was discovered to be useless for crypto, but the United Nations General Assembly adopted a resolution on artificial intelligence, not that anyone cares or that anyone can do anything about AI in any event. But for the record, UN officials formally called on tech companies to develop safe and reliable AI systems that comply with international human rights. And I loved this. They said: "Systems that don't comply should be taken offline." So, you know, you have a mean AI, just unplug it, folks. Officials said the same rights that apply offline should also be protected online, including against AI systems.

I've never said much about AI here. Just as I'm not trained as an attorney, I do not have any expertise in AI systems. What I do have, however, is stunned amazement. As they would say over in the UK, I am gobsmacked by what I've seen.

Leo: It is impressive; isn't it. What do you...

Steve: Oh, my god.

Leo: You know, I haven't ever asked you, and we talk about it all the time on the other shows, but what do you think the future holds?

Steve: Well, here I come. So what I may lack in expertise appears to have been made up for by my intuition, which has been screaming at me ever since I spent some time chatting with ChatGPT 4. My take on the whole AI mess and controversy can be summed up in just four words, and they are: "Good luck restraining anything."

Leo: Yeah, that's my attitude. Exactly.

Steve: Yes. I doubt that any part of this is restrainable. At some point in the recent past we crossed over a tipping point, and we're seeing something that no one would have believed possible even five years ago. Everyone knows there's no going back. Only people who have not been paying attention imagine that there's any hope of controlling what happens going forward. I don't know, and I can't predict what the future holds. But whatever is going to happen is going to happen, and I'm pretty sure that it's bigger than us.

Leo: Yes.

Steve: We're not a sufficiently organized species to be able to control or contain this.

Leo: Mm-hmm. Look how well we've done with nuclear proliferation.

Steve: Yeah.

Leo: And that's - it's still incredibly hard to purify enough plutonium to make a bomb. It's trivially easy, and the process is well, well known, to make an LLM.

Steve: It's out.

Leo: It's out. It's done.

Steve: You know, it would be like government saying, whoops, stop exporting crypto. Like, what?

Leo: Yeah, exactly.

Steve: You know. And so Leo, you and I are on the same page. I mean, it is - and if we don't, like, do it, we know North Korea's not sitting around doing nothing. They apparently have quite smart people.

Leo: Yes.

Steve: It annoys me that they're so good at hacking. But, boy, they are serious hackers. And so, you know...

Leo: It's going to happen.

Steve: It is. I would argue it already has, and we just haven't - it hasn't dawned on us yet.

Leo: Yeah.

Steve: Right? Like there's some inertia of recognition.

Leo: I for one, I'm excited. I mean, this is sci-fi. We are going to live in, I think I might even live to see it, a very weird and different future. It's coming.

Steve: Yeah.

Leo: It's going to be fun. Buckle up.

Steve: That's exactly right. I think that's exactly right. Okay. So a few more points to get to. In a somewhat disturbing turn, Spain has joined the likes of China, Thailand,

Pakistan, Iran, and Cuba to be blocking all use of and access to Telegram across its territory. This came after Spain's four largest media companies successfully complained to the high court in Spain that Telegram was being used to propagate their copyrighted content without permission.

A judge with Spain's high court had asked Telegram to provide certain information relating to the case, which apparently Telegram just blew off and ignored. They chose not to respond to the judge's request. So he ordered all telecommunications carriers to block all access to Telegram throughout the country. That began yesterday. So, you know, it's a problem.

Leo: I'd be very interested to see how this holds up because about - I heard that about a third of Spain uses Telegram.

Steve: Yes. It has already created a huge...

Leo: Yeah.

Steve: Yes. There's a huge consumer backlash against this, as one would expect.

Leo: Yeah, remember Brazil tried to do this. And they ended up having to back down. I think it was for WhatsApp. But they ended up having to back down because you can't - we can't communicate. What are you doing?

Steve: Well, have you seen the movie "Brazil," Leo? That explains the whole problem.

Leo: You know it was a great movie.

Steve: That's right, yes. Wonderful movie. So last week Vancouver held its 2024 Pwn2Own hacking competition. One security researcher by the name of Manfred Paul distinguished himself by successfully exploiting, get this, all four of the major web browser platforms.

Leo: Wow.

Steve: He found exploits in Chrome, Edge, Firefox, and Safari. He became this year's "Master of Pwn" and took home $202,500 in prize money. Overall, and here's really the lesson, the competing security researchers-turned-hackers successfully demonstrated 29 previously unknown zero-days during the contest and took home a total of $1.1 million in prize money.

Leo: That money comes from the companies that they're pwning, pretty much; right?

Steve: Yes, yes. Twenty-nine, okay, 29 previously unknown zero-days were found and demonstrated. To me this serves to demonstrate why I continue to believe that the best working model that's been presented for security - and okay, yes, I'm the one who presented it - is "porosity." Porosity.

Leo: Porosity.

Steve: You know, we don't want it to be, but security is porous. How else can we explain that one lone research hacker is able to take down all four of the industry's fully patched browsers whenever someone offers him some cash to do so? And that overall, 29 new previously unknown zero-days were revealed, when others were similarly offered some cash prize incentive. You know, you push hard, and you can get in. That's the definition of porous. And that's the security we have.

Leo: Yeah.

Steve: I should also take a moment to give a shout-out to Mozilla's Firefox team, who had patched and updated Firefox in fewer than 24 hours following the vulnerability disclosure. Frederik Braun posted on Mastodon: "Last night, about 21 hours ago, Manfred Paul demonstrated a security exploit targeting Firefox 124 at Pwn2Own. In response, we have just published Firefox 124.0.1 (and Firefox ESR 115.9.1) containing the security fix." He says: "Please update your foxes. Kudos to all the countless people postponing their sleep and working toward resolving this so quickly. Really impressive teamwork again. Also, kudos to Manfred for pwning Firefox again."

So, you know, this is the way security is supposed to work at the best of times. White hat hackers are given some reason to look, and compensated for their discoveries, which makes the products safer for everyone. And then the publishers of those products promptly respond to provide all of that product's users the benefits of that discovery. Yay.

And in this welcome bit of news, perhaps we and others are giving as good as we get. I've often noted that all we ever hear about, you know, about attacks on our infrastructure are Chinese state-sponsored attacks that are successfully getting in. You know, and I note that naturally we never hear about our similar successes against China. It's not like the NSA is going to brag. So I've wanted to believe that, while we would not destructive if we were to get in, that we'd only seek to have a presence inside Chinese networks so that they understand that we're just not sitting here defenseless over on this side of the Pacific.

Well, it turns out that last week China's state security agency themselves urged their local companies to improve their cybersecurity defenses. The Ministry of State Security said that foreign spy agencies have infiltrated hundreds of local businesses and government units. So that does sound like we may be at parity in this weird cyber cold war that we're in. I hate it; but, you know, it's what we've got.

Oh, and just a reminder. There has been an observed significant increase in tax season-related phishing. So I just wanted to remind everyone that, as happens at every time this year, you know, phishing scams suddenly jumped with all kinds of like, oh, you just - we received your electronically submitted return, but it had a problem. Please click here. But that's not from the IRS. So everybody put up your skepticism shields and resist clicking.

I have two quick notes of news that I think everyone'll find interesting on the SpinRite front. One of the things that quickly became apparent as our listeners were wishing to obtain and use 6.1 was that the world had changed in another way since SpinRite 6's release back in 2004. Back then, Linux was still largely a curiosity, you know, with a relatively small fan base and no real adoption. Not so today, at least not among our listeners.

Back in 2004 it was acceptable to require a SpinRite user, I mean, just assumed that a SpinRite user would have Windows, which they would use to set up the boot media since Windows and Mac was pretty much all there was, and SpinRite was never really targeted at the Mac market. Today, we've encountered many would-be users who do not have ready access to a Windows machine. And they've been having a problem. So I needed to create a non-Windows setup facility that I have long envisioned but never needed until now. Today it exists.

Over at GRC's prerelease.htm page is, as before, the downloadable Windows/DOS hybrid executable, and now also a downloadable zip file. The zip file, which is smaller than 400K, contains the image of the front of a 4GB FAT32 DOS partition. So any SpinRite owner without access to Windows, because using Windows is still easier, may choose to instead download this zip file. And it's personalized. I've added on-the-fly partition creation, and SpinRite is added to the file system. It's then truncated, and I've got on-the-fly zipping. I've been busy. It contains about an - the zip file, which is as I said less than 400K, contains an 8.3MB file which is named SR61.img. Any Linux user can, you know, "dd" copy that file onto any USB thumb drive to create an up to 4GB FAT32 partition that will immediately boot and run SpinRite.

But the tricky bit that I worked out last week is that when this drive is booted for the first time, if the media onto which this image file was copied is smaller than the partition described by the image, which is a 4GB partition - for example, SpinRite's owner copies the image to an old but trusted 256MB thumb drive - a little built-in utility named "downsize" kicks in, examines the size of the partition's underlying physical drive, and dynamically on the fly "downsizes" the partition to fit onto its host drive. It's all transparent and automatic. And since this same technology was also going to be needed for SpinRite 7, it made sense to get it done, so it's there now.

Second point. A new wrinkle to surface last week is bad RAM. Over in GRC's web forums, a SpinRite 6.1 user reported data verification errors being produced by SpinRite when running on his cute little ZimaBoard. SpinRite always identified and logged the location of the apparent problem. But from one run to the next there was no correlation in where the problems appeared to be occurring. And when he ran the same drive under SpinRite on a different PC, it passed SpinRite's most thorough Level 5 testing without a single complaint. And he was able to go back and forth to easily recreate the trouble multiple times on one system, but never on the other.

The inhabitants of the forums jumped on this and suggested a bad or undersized power supply for his ZimaBoard, flaky cabling, and anything else they could think of, all great suggestions. Finally, I asked this user to try running the venerable MemTest86 on his brand new ZimaBoard. And guess what? Yep. Memory errors. There should never be any. But the first time he ran MemTest86 it found six, and the second time it found 101.

Seeing that, we ran MemTest86 on all of our ZimaBoards, that is, all of the developers, and they all passed with zero errors, as they always should. So this user had a ZimaBoard with a marginal DRAM memory subsystem. There was no correlation in the locations of the errors that SpinRite was reporting from one memory - oh, that his MemTest was reporting from one pass to the next. But there were always two specific bits out of the 32 that MemTest86 always identified as being the culprits. They were soft.

And SpinRite was getting tripped up by this machine's bad RAM when it was performing data verification that's available from SpinRite's levels 4 and 5.

The problem was not the drive. It was the machine hosting SpinRite and the drive. So by this point our longtime listeners who've grown to know me, listening to this podcast, know what I'm going to say next. Yep, SpinRite 6.1 now tests the memory of any machine it's running on.

Leo: Oh, clever. Wow. Who needs MemTest? I've got SpinRite.

Steve: That's right. It works great. It, like, immediately found the errors this guy was having. What's interesting is that SpinRite 1.0, back in 1988, also built-in a memory test. Back then it made sense to verify the RAM memory that would be used to temporarily hold a track's data while SpinRite was pattern testing the physical surface and giving it a fresh new low-level format. But I don't know when it happened, somewhere along the way I removed that feature from SpinRite. We never heard of it ever being useful, so my initially over-cautious approach seemed to have been proven unnecessary - until last week.

So late last week I implemented a very nice little DRAM memory tester right into SpinRite and then had the guy with the bad ZimaBoard give it a try. It successfully determined that his machine's memory was not reliable, and SpinRite will then refuse to run on any such machine after making that determination. You know, it's just not safe to run it. And of course no such machine should be trusted for actually doing anything else. You know, it's like send it back to the manufacturer; or, if you can, change the RAM, or diagnose it.

So anyway, this new built-in RAM testing feature - which is not yet present. Don't go download an updated copy of SpinRite. It's not there yet, not yet present in any SpinRite that's available for download. But it'll appear, along with a few other minor improvements that I've made, shortly. So I'm sure I'll be announcing it next week.

And I just have two little pieces of feedback from our listeners because we have lots to still talk about here. I got a note from someone whose handle is Jazman. He said: "Hi, Steve. Great show, as always. I work in a cell phone-free environment. Not only no service, but we're not allowed to bring them. We have Internet computers, but we're not trusted to install anything on them." Sounds like he is with the NSA. "The problem is I like to have two-factor authentication to protect my email and other stuff. My understanding is, if I were to use Passkeys, I need my phone. I use Bitwarden with two-factor authentication. My question: Are there any good solutions for a cell-free environment? Kind regards, Bjorn."

Okay. So, and we've been talking about this for the last couple weeks, whether to have two-factor and now optionally Passkeys managed by your password manager, or to keep it separate. In a phone-free environment, I agree that relying upon Bitwarden for all authentication services is likely the best bet. I think it's probably your only bet; right? You know, we would usually prefer to have Passkeys or our authenticator on a separate device like a phone. But where that's not possible, merging those functions into a single password manager like Bitwarden makes sense. And I should just note that YubiKeys are also Passkeys-capable, and they're able to store up to 25 Passkeys in a YubiKey. So a YubiKey is another possibility, if that somewhat limited Passkeys capacity doesn't pose a problem.

And finally, William Ruckman. He said: "Hi, Steve. Are Passkeys quantum safe? I thought public key crypto was vulnerable." And we'd also been speaking this recently about how the big difference between username and password and Passkeys is the essentially

symmetric crypto secret keeping, whereas Passkeys uses public key crypto, which is why William's asking. So it's a terrific question because, as we know, it's the public key crypto that Passkeys offers, which is why it's so valuable. The good news is the FIDO2 specification which underlies WebAuthn, which underlies Passkeys, already provides for plug-in future-proof crypto. So Passkeys and WebAuthn/FIDO2 will all be able to move to quantum-safe algorithms whenever that's appropriate, and as soon as we've settled on them, and they've been standardized. So yes.

Leo: Good. That's good news. And it would be backward to all the Passkeys you already are using and all that. Right? Maybe not.

Steve: No. Could not be.

Leo: You'd have to regenerate them all. Okay.

Steve: Yes. If you change the crypto, you would have to regenerate the Passkeys because you're holding private keys with a specific algorithm.

Leo: Right.

Steve: And there is actually no way for the website even to help you. I mean, it might say, you might go through a "use your old Passkey, now use your new Passkey."

Leo: Yeah.

Steve: And if you did that, you know, sequentially, then it would get - actually SQRL had a similar facility. So it would use the first authentication to assert your identity and thus honor the second authentication, which would be from the newfangled crypto. And now it would have the public key under the new algorithm.

Leo: Cool. We don't have to worry about it yet. There's only about four sites that use this.

Steve: I know. I saw, in doing some research just yesterday, I saw someone who had something to sell, they were trying to sell some equivalent of a YubiKey, I think. And it said, "Since the majority of the Internet's websites are now using Passkeys." And I thought, are you in a time machine? What are you talking about?

Leo: Oh, you're talking 2030. Oh, yeah. Maybe. Maybe.

Steve: Yeah, yeah.

Leo: There's literally...

Steve: The majority as long as you only log into PayPal.

Leo: Yeah, right. There's literally just a handful of sites that use it.

Steve: I know. I know.

Leo: It's too bad because it's so easy when it works. I heard you talked a lot about that last week with Mikah. And I agree with you, I think it's going to be a big improvement someday. Someday our prince will come.

Steve: Okay. After our final announcement, Leo, oh, boy, we're going to have some fun.

Leo: Oh, boy.

Steve: Get the beanies lubed.

Leo: I don't know if you should say that. Shouldn't say that out loud, anyway. And now, let's talk about Fetch. GoFetch.

Steve: So, GoFetch. Last Thursday the world learned that Apple had some problems with their cryptography. Unfortunately, it would be impossible to determine from most of the tech press's coverage of this whether this was an apocalyptic event or just another bump in the road. Ars Technica was apparently unable to resist becoming click-bait central with their headline "Unpatchable vulnerability in Apple chip leaks secret encryption keys." Wow. That would be bad if it was true.

Fortunately, it's not the least bit true. It's not unpatchable, and it's not a vulnerability in an Apple chip. Kim Zetter's Zero Day goes with: "Apple Chip Flaw Lets Hackers Steal Encryption Keys." This "chip flaw" [in air quotes] theme seems to have become pretty popular, even though nowhere did any of the actual researchers ever say anything about any chip flaw. Even Apple Insider's headline read "Apple Silicon vulnerability leaks encryption keys and can't be patched easily." What?

Apple was told 107 days before the disclosure, back on December 5th of last year. Apple is certainly quite aware of the issue, and I'm sure they're taking it seriously. And for their newer M3 chips, all that's needed is for a single bit to be flipped. Tom's Hardware went with: "New chip flaw hits Apple Silicon and steals cryptographic keys from system cache. 'GoFetch' vulnerability attacks Apple M1, M2, M3 processors, can't be fixed in hardware." Oh dear.

Except for a few details: It's not new. It's not a flaw. Nothing ever "hit" Apple Silicon. And as for it not being fixable in Apple M1, M2 or M3 processors, if you have an M3 chip, just flip the bit "on" during crypto operations and the unfixable problem is solved. And finally, as we'll see by the end of this topic today, there are equally simple workarounds for the earlier M-series processors.

Okay. So I could keep going because the material in this instance was endless. Not a single one of the headlines of the supposedly tech press stories that covered this characterized this even close to accurately. It's not a flaw. Nothing is flawed. Everything

is working just as it's supposed to. It's not a vulnerability in Apple Silicon. Apple Silicon is just fine, and nothing needs to change. And it's certainly not unfixable or unpatchable. CyberNews headline was "M-series Macs can leak secrets due to inherent vulnerability." The only thing that's inherently vulnerable here is the credibility of the tech press's coverage of this.

Leo: Holy cow.

Steve: It really has been quite over the top. After sitting back and thinking about it, the only explanation I can come up with is that, because what's actually going on with this wonderfully and subtly complex problem, no one writing for the press really understood what the researchers have very carefully and reasonably explained. So they just went with variations on Ars Technica's initial "unpatchable vulnerability in Apple's chip" nonsense, under the assumption that Ars must have actually understood what was going on, so everyone just copied them.

Leo: I just assumed Dan Gooden knows. If he doesn't know - right, yeah.

Steve: You know, Dan's on the ball, typically. And we do know, in fairness to Dan, he doesn't provide the headlines. Back when I was writing the Tech Talk column for InfoWorld, I was often really annoyed by what my columns were headlined because that's not what I said in the text. But, you know, some copy editor, I guess that's what they're called, gave it the headline that would get people to turn to the page. So, okay. Not Dan's fault.

Okay. The TL;DR of this whole fiasco is that a handful of researchers built upon an earlier two-year-old discovery, which three of them had been participants in back then, that was dismissed at the time by Apple, you know, as being of only academic interest. It's yet another form of side-channel attack on otherwise very carefully designed to be side-channel attack-free constant-time cryptographic algorithms. The attack surrounds an ARM-based performance optimization feature known as DMP. And I was thinking, boy, if the acronym had been EMP, that would have really blown the tech press right off the top. Anyway, not EMP, DMP. And a variation of the same type of optimization is also present in the newest Intel chips, their Raptor something or other. Anyway, I'll get to that.

Okay. And so, true to Bruce Schneier's observation that attacks never get worse, they only ever get better, about a year and a half after that initial discovery two years ago which never amounted to much, it turned out that the presence of this DMP, which I will be explaining in detail, optimization feature actually did and does create an exploitable vulnerability that can be very cleverly leveraged to reveal a system's otherwise well-protected cryptographic secrets. After verifying that this was true, the researchers did the responsible thing by informing Apple, and we have to assume Apple decided what they wanted to do next. Okay. Unfortunately, that true story doesn't make for nearly as exciting a headline. So none of the hyperventilating press explained it this way.

One important thing that sets this apart from the similar and related Spectre and Meltdown vulnerabilities from yesteryear is that this new exploitation of the DMP optimizer is not purely theoretical. All we had back in those early days of speculative execution vulnerabilities was a profound fear over what could be done, over what this meant. It was clear that Intel had never intended for their chip's internal operation to be probed in that fashion, and not much imagination was required to envision how this might be abused. But we lacked any concrete real-world proof of concept. Not so today.

And not even post-quantum crypto is safe from this attack since we're not attacking the strength of the crypto, but rather the underlying keys are being revealed.

The GoFetch proof-of-concept app running on an Apple Mac connects to the targeted app, also on the same machine, which contains the secrets. It feeds the app a series of inputs that the app signs or decrypts or does something using its secret keys, basically inducing it to perform cryptographic operations that require it to use the secrets it's intending to keep. As it's doing this, the app monitors aspects of the processor's caches which it shares with the targeted app in order to obtain hints about the app's secret key. Okay. So how bad is it?

As I mentioned, the attack works against both pre- and post-quantum encryption. The demo GoFetch app requires less than an hour to extract a 2048-bit RSA key, and a little over two hours to extract a 2048-bit Diffie-Hellman key. The attack takes 54 minutes to extract the material required to later assemble a Kyber 512-bit key and about 10 hours for a Dilithium 2 key, though some time is also required afterwards for offline processing of the raw data that is collected. In other words, it is an attack that is practical to employ in the real world.

Okay. So what exactly is DMP, what did the researchers discover, and how did they arrange to make their Macs give up the closely held secrets being hidden inside? The research paper is titled "GoFetch: Breaking Constant-Time Cryptographic Implementations Using Data Memory-Dependent Prefetchers." Okay, now, that sounds more complex than it is. We have "Breaking Constant-Time Cryptographic Implementations." We already know that a classic side-channel vulnerability which is often present in poorly written crypto implementations is for an algorithm to in any way change its behavior depending upon the secret key it's using. If that happens, the key-dependent behavior change can be used to infer some properties of the key. So the first portion of the title tells us that this attack is effective against properly written constant-time cryptographic implementations that do not change their behavior in any way. That's not where things got screwed up.

The second part of the paper's title is "Using Data Memory-Dependent Prefetchers," and that's what's new here. If you guessed that the performance optimization technique known as DMP stands for "Data Memory-Dependent Prefetchers," you'd be correct. Three of the seven co-authors of today's paper co-authored the earlier groundbreaking research two years ago which described their reverse-engineered discovery of this DMP facility residing inside Apple's M-series ARM-derived chips. Back then they raised and waved a flag around, noting that what this thing was doing seemed worrisome; but they stopped short of coming up with any way to actually extract information. And the information that they had was made public.

Now, we don't know for sure that sophisticated intelligence agencies somewhere might not have picked up on this and turned it into a working exploit, as has now happened. But we do know for sure that Apple apparently didn't give this much thought or concern two years ago, since every one of their Mac M-series chips was vulnerable to exploitation several years later.

Okay. I'm going to share today's research abstract, today's, the updated current research abstract and introduction, since it's packed with information and some valuable perspective. And then I'll break it down. So they wrote: "Microarchitectural side-channel attacks have shaken the foundations of modern processor design. The cornerstone defense against these attacks has been to ensure that security-critical programs do not use secret-dependent data addresses. Put simply, do not pass secrets as addresses, for example, data memory instructions. Yet the discovery of data memory-dependent prefetchers (DMPs), which turn program data into addresses directly from within the memory system, calls into question whether this approach will continue to remain secure.

"This paper shows that the security threat from DMPs is significantly worse than was previously thought and demonstrates the first end-to-end attacks on security-critical software using the Apple M-series DMPs. Undergirding our attacks is a new understanding of how DMPs behave which shows, among other things, that the Apple DMP will activate on behalf of any victim program and attempt to 'leak' any cached data that resembles a pointer. From this understanding, we design a new type of chosen-input attack that uses the DMP to perform end-to-end key extraction on popular constant-time implementations of classical and post-quantum cryptography."

And by way of introduction, they said: "For over a decade, modern processors have faced a myriad of microarchitectural side-channel attacks, for example, through caches, TLBs" - Translation Lookaside Buffers - "branch predictors, on-chip interconnects, memory management units, speculative execution, voltage/frequency scaling, and more." You know, as we know, even like the sound of the power supply changing can leak information.

They said: "The most prominent class of these attacks occurs when the program's memory access pattern becomes dependent on secret data. For example, cache and TLB side-channel attacks arise when the program's data memory access pattern becomes secret dependent. Other attacks, for example, those monitoring on-chip interconnects, can be viewed similarly with respect to the program's instruction memory access pattern. This has led to the development of a wide range of defenses - including the ubiquitous constant-time programming model, information flow-based tracking, and more all of which seek to prevent secret data from being used as an address to memory/control-flow instructions.

"Recently, however, Augury" - that's what they called their first research two years ago, A-U-G-U-R-Y, and it related to an auger being used - "demonstrated that Apple M-series CPUs undermine this programming model by introducing a Data Memory-Dependent Prefetcher that will attempt to prefetch addresses found in the contents of program memory. Thus, in theory, Apple's DMP leaks memory contents via cache side channels, even if that memory is never passed as an address to a memory/control-flow instruction." Okay. And again, I will explain exactly what all that means. I've got a couple paragraphs left.

They said: "Despite the Apple DMP's novel leakage capabilities, its restrictive behavior has prevented it from being used in attacks. In particular, Augury reported that the DMP only activates in the presence of a rather idiosyncratic program memory access pattern, where the program streams through an array of pointers and architecturally dereferences these pointers. This access pattern is not typically found in security critical software such as side-channel hardened constant-time code, hence making that code impervious to leakage through the DMP.

"With the DMP's full security implications unclear, in this paper we address the following two questions: Do DMPs create a critical security threat to high-value software? And can attacks use DMPs to bypass side-channel countermeasures such as constant-time programming? This paper answers the above questions in the affirmative, showing how Apple's DMP implementation poses severe risks to the constant-time coding paradigm. In particular, we demonstrate end-to-end key extraction attacks against four state-of-the-art cryptographic implementations, all deploying constant-time programming." And just to be clear, when they say "end-to-end attacks," they mean they run something, and they get the key. Meaning all the work is done, nothing left for the reader to finish. You know, this thing works.

Okay. As we've had the occasion to discuss through the years on this podcast, the performance of DRAM, the dynamic RAM memory that forms the bulk of our system's memory, has lagged far behind the memory bandwidth demands of our processors.

Through the years we've been able to significantly increase the density of DRAM, but not its performance. And as we know, even the increase in density has met with challenges in the form of susceptibility to adjacent row interference which led to the various DRAM hammering attacks.

But on the performance side, the saving grace has been that processor memory access patterns are not linear and non-repetitive. They are typically highly repetitive. The programs almost always "loop," meaning that they are executing the same code again and again, over and over. And that, in turn, means that if a much smaller but much faster "cache" of memory is inserted between the main DRAM and the processor, the processor's repetition of the same instructions, and often the data for those instructions, can be fulfilled much more quickly from the local cache than from main memory.

During our discussions of speculative execution we saw that another way to speed up our processors was to allow the processor to run well ahead of where execution was; and, if the code encountered a fork in the road, in the code's flow, it would fetch ahead down both paths of the fork so that once the path to be taken became known, whichever way that went, the system would already have read the coded instructions for that path and have them ready to execute. In practice, this is accomplished by breaking our processors into several specialized pieces, one being the prefetch engine whose job it is to keep the execution engines fed with data from main memory.

Many instructions do not make any main memory accesses. They might be working only within the processor's internal registers or within what's already present in the processor's local cache. So this gives the prefetching engine time to anticipate where the processor might go next and to guess at what it might need. In a modern system, there's never any reason to allow main memory to sit idly by, not even for a single cycle. A good prefetching system will always be working to anticipate its processor's needs and to have already loaded the contents of slower DRAM into the high-speed cache when the processor gets to needing it.

Okay. Now let's add one additional layer of complexity. One of the features of all modern processor architectures is the concept of a pointer. A location in memory or the contents of a register could contain an object's value itself, or instead it could contain the memory address of the object. In that second case we would say that the value in the memory or register contains, instead of the value of the object itself, a pointer to the object. As a coder, I cannot imagine my life without pointers. They are absolutely everywhere in code because they are so useful.

We need one bit of new vocabulary to talk about pointers. Since a pointer is used to point to or refer to something else, the pointer contains a reference to the object. So we call the act of following a pointer to the object "dereferencing" the pointer. We'll see the researchers using that jargon in a minute. But first let's think about that cache-filling prefetch engine. Its entire reason for existence is to anticipate the future needs of its processor so that whatever the processor wants will already be waiting for it and instantly available from its cache. The processor will think that its prefetch engine is magic.

So one evening, probably about seven years ago, some Apple engineers are sitting around a whiteboard with a bunch of half-eaten pizzas. They're brainstorming ways to further speed up Apple's proprietary silicon. Given the timeframe, this would first be able to appear in their A14 Bionic processor. So one of them says: "You know, we're already doing a great job of fetching the data that the processor is going to ask for. But when we fetch data that contains what looks like pointers, we're not fetching the data that those pointers are pointing to. If the data really are pointers, then there's a good chance that once the processor gets its hands on them, it's going to be asking for that data next. We could anticipate that and have it ready, too, just in case it might be useful. I mean,

what's the whole point of being a prefetching engine; right? That's the whole point. That's what we're here for."

Now, at this point, the pizza is forgotten, and several in the group lean forward. They're thinking about the kinds of cars they're going to be able to get with the raises this idea will earn them. Then they realize they need to make it work first. Although they're immediately hooked by the idea because they know there's something there, one of them plays devil's advocate, saying: "But the cache is context-free." What he means by that is that the prefetch engine sees everything as data. It's all the same to it. The prefetcher doesn't know what the data means. It has no meaning in DRAM. It's all just mixed bytes of instructions and data. It's a hodgepodge. It's not until that data is fetched from the cache and is actually consumed by the processor that the data acquires context and meaning.

The answer to the "but the cache is context-free" guy is, yeah, and so what? If some data that's being added to the cache looks like a pointer, and if it's pointing into valid DRAM memory, what's the harm in treating it as a pointer and going out and also grabbing the thing that it might be pointing to? If we have time, and we're right, it's a win for the processor. The processor won't believe its luck in already having the thing it was just about to ask for already magically waiting there in its local cache.

So finally, after their last dry-erase marker stops working from the hastily scribbled diagrams on their whiteboards, they're satisfied that they're really onto a useful next-generation optimization. So one of them asks: "Okay. This is good. But it needs a name. What are we going to call it?" One of them says, "Well, how about Data Memory-Dependent Prefetching, or DMP for short?"

So here we've just seen a perfect example of where and how these next-generation features are invented, over pizza and dry-erase markers. And it's also easy to see that the security implications of this don't even make it onto the radar. All they're doing, after all, is anticipating a possible future use of what might be a pointer, and prefetching the thing it's pointing to in case they're right, and it is a pointer, and in case the processor might eventually ask for it. It's disconnected from whatever the processor is doing; right? It's a Data Memory-Dependent Prefetcher. What this amounts to is a somewhat smarter prefetcher. It cannot be certain whether it's fetching a pointer. But in case it might be, it'll just jump ahead even further to also prefetch the thing that what might be a pointer may be pointing to.

Okay. So now let's hear from the geniuses who likely also consumed their share of pizza while they scratched the itch that had apparently been lingering with at least three of them for a couple of years, ever since that first bit of work, when they discovered that Apple had dropped this Data Memory-Dependent Prefetcher into their silicon. Here's how they explain what they came up with.

They said: "We start by reexamining the findings in Augury. Here we find that Augury's analysis of the DMP activation model was overly restrictive and missed several DMP activation scenarios. Through new reverse engineering, we find that the DMP activates on behalf of potentially any program, and attempts to dereference any data brought into cache that resembles a pointer. This behavior places a significant amount of program data at risk, and eliminates the restrictions reported by prior work. Finally, going beyond Apple, we confirm the existence of a similar DMP on Intel's latest 13th-generation Raptor Lake architecture with more restrictive activation criteria. Next, we show how to exploit the DMP to break security-critical software. We demonstrate the widespread presence of code vulnerable to DMP-aided attacks in state-of-the-art constant-time cryptographic software, spanning classical to post-quantum key exchange and signing algorithms."

Okay. And then finally, this last bit is the key to everything. I'll read it first, then take it apart. They said: "Our key insight is that while the DMP only dereferences pointers, an attacker can craft program inputs so that when those inputs mix with cryptographic secrets, the resulting intermediate state can be engineered to look like a pointer, if and only if the secret satisfies an attacker-chosen predicate. For example," they said, "imagine that a program has secret s, takes x as input, and computes and then stores y = s?x (s XORed with x) to its program memory. The attacker can craft different x's and infer partial or even complete information about s by observing whether the DMP is able to dereference y. We first use this observation to break the guarantees of a standard constant-time swap primitive recommended for use in cryptographic implementations. We then show how to break complete cryptographic implementations designed to be secure against chosen-input attacks."

Okay. So they realized that Apple's DMP technology is far more aggressive than they initially appreciated. It is busily examining all of the data that's being put into the cache for all of the processes running in the system. It's looking for anything that looks "pointer-like"; and, when found, it's going out and prefetch that because it may be pointing to something that the process was going to ask for in the future.

Their next step was to realize that, since this "pointer-like" behavior is highly prone to producing false positive hits which would prefetch miscellaneous bogus data, and since it operates indiscriminately on any and all data in the system, they can deliberately trick Apple's DMP system to misfire. When it does, it will prefetch data that wasn't really being pointed to, and they can use standard, well-understood cache probing to determine whether or not the DMP did in fact misfire and prefetch. Since the cause of that mixes secrets with what they provide, it reveals information about the secret.

They induce the isolated process containing the secrets to perform a large number of cryptographic operations on their deliberately crafted data while using the now well-understood behavior of the DMP to create an inadvertent side channel that leaks the secret key, even though the cryptographic code itself is being super careful not to behave differently in any way based upon the value of the secret key. In other words, it's being betrayed by this advanced operation of their prefetching cache. The code's care doesn't matter because the cryptographic code, as I said, is being betrayed. What I've just explained is a version of what these very clever researchers revealed to Apple back 107 days ago from last Thursday in early December last year.

So what does Apple do about this? This does seem like the sort of thing Apple ought to be able to turn off. One of the things we've learned is that these initial nifty-seeming slick performance optimizations, like Spectre and Meltdown and all the others, always seem to come back to bite us sooner or later. So the lesson we absolutely as an industry have to take away, and it's surprising we haven't yet, is that anything like this should have an off switch.

And what do you know? It may have been, and likely was, in reaction to these researchers' initial Augury DMP paper back in 2022 that Apple added that off switch to their M3 chip. Apple announced it on October 30th last year, the day before Halloween. And that M3 can have DMP turned off. I've heard, but I haven't confirmed, that Apple's own crypto code is flipping DMP off during any and all of their own cryptographic operations. So it may only be non-Apple crypto code running on Macs that are endangered on M3-based machines. The researchers cite their compromise of the Diffie-Hellman Key Exchange in OpenSSL, you know, not an Apple library, and the RSA key operations in the Go language library. So again, not Apple's.

So what about the non-M3 chips, the Apple A14 Bionic, the M1, and M2? Well, it turns out that these so-called SoC, you know, Systems on a Chip, all have multiple cores, and the cores are not all the same type. Only half of the cores are vulnerable because only

half of them incorporate the DMP. Apple's M-series have two types of cores: the bigger Firestorm cores, also known as the performance cores; and the smaller Icestorm cores, also known as the efficiency cores. On the M1 and M2 chips, only the Firestorm performance cores offer the problematic DMP prefetching system. So all Apple needs to do is to move their crypto over to the smaller efficiency cores. Crypto operations will run more slowly there, but they will be completely secure from this trouble.

So is Apple going to do any of these things? Have they already? The press thinks that nothing has been done yet. I find that curious given that the concerns are real and that solutions are available. But so far all the press has reported - now, again, Apple knew about this in early December. All the press has reported that Apple has been curiously mute on the subject. Apple just says "no comment." This is doubly confounding given that Thursday's research disclosure came as no surprise to them, right, and also that the firestorm of truly over-the-top apoplectic and apocalyptic headlines that have ensued as a result really does need a response.

I imagine that something will be forthcoming from Apple soon. Until then, for what it's worth, the attack, if it were to happen, would be local, and would be targeted, and would require someone arranging to install malware onto the victim's machine. It's not the end of the world. And as I'm always saying around here, anyone can make a mistake. But Apple's customers would seem to need and deserve more than silence from Apple. So we ought to hear something.

Leo: Yeah.

Steve: But at least now we understand exactly what's going on.

Leo: And by the way, if somebody can install that on your system, they could also just put a keystroke logger on there. There's all sorts of ways they can get full access. That's, you know, in fact that's probably a lot easier, to do it some other way than a side-channel attack. Does it take a lot of monitoring and trial and error to have the side-channel...

Steve: No.

Leo: It doesn't.

Steve: It takes an hour, and you get the key.

Leo: Okay.

Steve: And so you actually do get a secret that was trying to be protected.

Leo: So I can see a nation-state saying, oh, good, all right. What we'll do is we'll get this on there through some other malware exploit. We'll run it, and then we'll erase all traces. Guy will never know he was hacked. But we've got the key, and we've got the key forever.

Steve: Right.

Leo: Until he changes it. I can see that.

Steve: Right. And, yeah, and the point I made was that, you know, when this became public two years ago, these guys apparently stopped their research. We don't know that the NSA did. The NSA might have gone, hey, that's interesting. Let's take a look at that.

Leo: Oh. The NSA could - NSA's probably been working on the same thing forever; right? I mean, they know about these side-channel attacks. They know about speculative execution. They know what Spectre and Meltdown produced on the x86 platforms. I'm sure they were looking for it, too. Just whose professors are better, I guess.

Steve: Yeah. Hopefully we have good profs.

Leo: I think we have good profs in the NSA.