



DigiCert Does It Right

Description: The FCC decides router firmware updates are useful. Netgear applies for and gets a full FCC pass. AI uncovers a 21-year old critical FreeBSD RCE. What was behind that Let's Encrypt outage? AI model repositories are overflowing with malware. The CISA 2015 info-sharing act is being renewed. Edge leaves ALL usernames and passwords in the clear. An examination of DigiCert's breach and their response.

High quality (64 kbps) mp3 audio file URL: <http://media.GRC.com/sn/SN-1078.mp3>

Quarter size (16 kbps) mp3 audio file URL: <http://media.GRC.com/sn/sn-1078-lq.mp3>

SHOW TEASE: It's time for Security Now!. Steve Gibson is here. The FCC has backed down a little bit, and that's good news for router manufacturers. AI has found a 21-year-old critical flaw in, well, the most secure operating system I know about. We'll talk about the Let's Encrypt outage, and then how DigiCert responded to its recent breach. Steve said A+ to DigiCert. That's coming up next on Security Now!.

Leo Laporte: This is Security Now! with Steve Gibson, Episode 1078, recorded Tuesday, May 12th, 2026: "DigiCert Does It Right."

It's time for Security Now!, the show where we cover your security, your privacy, and how computers work, and then maybe a little sci-fi and Vitamin D thrown in with this guy right here because basically this is the show where Steve talks about the stuff he cares most about. Hello, Steve.

Steve Gibson: Which I'm not sure if it was self-selecting, but our listeners tend to agree.

Leo: Yes.

Steve: So they're like, I mean, I'm getting Vitamin D email and sleep supplement questions. And so I know that our listeners are, you know, focused.

Leo: And Steve, we should add coffee to this because Steve is, as we well know, a five-shot venti latte drinker. And there it is in a giant mug. I had some wonderful coffee in Kona. They grow Kona coffee on the side of the volcano.

Steve: That's how they named it, actually, yeah.

Leo: Yes. The name came from the city.

Steve: That's right.

Leo: But it is amazing coffee. So much so that I bought a large amount of it to bring back because just it's so...

Steve: No, I mean, good coffee is just in a class by itself.

Leo: It's so good. But I don't think you would like Kona coffee because I remember we talked about this. In fact, this came to mind as I'm drinking it. Because of where it's grown, in the volcanic soil, it has less caffeine and very little bitterness, very little bite. And I remember you like the bite.

Steve: Yes. In fact, decaf lacks the bite. And so it's like, eh, seems a little "why bother?"

Leo: Kona coffee is almost like tea. It's very smooth and delicious, and a little bit less caffeinated. So, yeah, I thought maybe he wouldn't - maybe he wouldn't like it that much, come to think of it. So I didn't send you any; okay? It's very...

Steve: I would rather have salt. Rather have salt.

Leo: Very expensive.

Steve: Rub some salt in my wounds.

Leo: I do have some salt from Salt Hank that's coming your way, soon as I figure out how I could package those glass jars in a way that they don't break. We had some special salt made for Steve and his wife. They have a fetish for my son's salt.

Steve: Yeah, that's right.

Leo: This time I couldn't get away. Yes.

Steve: Okay. So we are - we're at Episode 1078 for May 12th. And I teased this last week, the news was just breaking, and I didn't have the whole story. And actually they didn't have the whole story. I'm talking about an interesting problem that DigiCert, the industry's now by far number one certificate authority, suffered. I titled today's podcast "DigiCert Does It Right" because I and a lot of other industry experts have singled their reporting out as this is the way you do this. If you suffer a breach, how do you disclose? And anyway, so we're going to - I want to really, you know, give them some props for, like, the job that they've done, and take a look at what they did to just, you know, both give them credit, but also to show, in some detail, this is the way it's done right.

Leo: So many people do it wrong. We should definitely mention they do it right.

Steve: Well, yeah, exactly. And especially we've covered other CAs that have lost their CA-ness.

Leo: Right.

Steve: Because of, you know, trying to, like, oh, no, that really was - oh, well, oh, you found that? Okay, well, then we'll have to talk, guess we'll have to, you know, that's just, you know, wrong. So, you know, props to them. We're going to talk about the FCC, however, deciding that firmware updates might actually be a good thing. So let's rethink that policy. Netgear, speaking of the FCC, is the first, and as far as I know so far only router manufacturer to get a full pass on this ridiculous...

Leo: Eero just got one yesterday.

Steve: Oh, good.

Leo: So there's two now.

Steve: Good. Also AI has uncovered a 21-year-old critical remote code execution vulnerability, and this one is trivially implemented in one of the most secure Unixes ever, my favorite and the one I use, FreeBSD. And I should also mention, I didn't get it into notes for this week, but I'll be talking about it next week, Google has announced that they have uncovered the first AI-generated zero-day. So we now have a confirmed example of AI, as we have been worried about, even pre-Mythos, this is not, you know, because presumably the bad guys didn't have access to it, although we do know that there were some leaks of Mythos access, but this has been the concern. So it's beginning to happen.

There was also a brief Let's Encrypt outage. We have another example of a company doing the right thing. Turns out there are now some reports, not surprisingly also, of AI model repositories overflowing with, I'm not sure you'd call it malware. Malprompting? I don't know what mal. It's bad, so mal. But anyway, that's a thing now. In addition to, you know, NPM and PyPI and everything, all the other open repositories that we've talked about. It looks like CISA, it's CISA, although it's a decade old, it's that agreement that was signed in 2015 which allows private companies to share cybersecurity things with the government without fear of reprisal. Looks like that was, well, it was temporarily extended. Looks like it's going to be made permanent.

We have some very distressing news about the Edge browser and what was found by someone and has now been confirmed, not only by people online, but some of my own newsgroup participants, who have done this and found all their usernames and passwords in the clear. So we'll talk about that. And then we're going to get into a deep look at how you do this right. If you are a company with serious responsibility for user security, taking responsibility and documenting it. And DigiCert did that. And I'm saying that even though I've wandered away from them, as we know, because of their pricing. But still, they're it. So I think a great podcast for our listeners. And of course a fun Picture of the Week.

Leo: Which I have not looked at, but will at some point, with you, after this word.

Steve: Probably then, yes.

Leo: From our sponsor. We will get to that Picture of the Week in just a moment. First - oh.

Steve: And for those who are seeing video, yes, Leo has an apparently white shirt on.

Leo: Oh, no, but I have to reveal.

Steve: For the first time in ever. There was a weird blue stripe hidden behind the microphone and some...

Leo: It's a Tommy Bahama Hawaiian shirt.

Steve: There's some foliage down below.

Leo: I guess it does look like a white shirt, doesn't it. I might have to retire this from the collection.

Steve: We're just so used to the collection of orange slices and...

Leo: I'm going to go change my shirt after this break. I'll wear something crazy and kookie, I promise. I apologize for not. Now, back to our man of the hour, Mr. Steve Gibson.

Steve: So I gave our Picture of the Week a title.

Leo: Okay.

Steve: I've titled it, "When a powerful meme creates fertile ground..."

Leo: A powerful meme creates fertile ground.

Steve: "Creates fertile ground, with apologies to Randall Munroe."

Leo: Oh, yes, of course.

Steve: "Of XKCD."

Leo: All right. Well, let me pull it up, and we'll look at it together for the first time.

Steve: Randall, this is what you begat.

Leo: So you remember his famous xkcd cartoon with the blocks all teetering on one little block that was written by a single developer. Oh my god. This has gotten a little more complicated. Wow.

Steve: So this is an updated version of this ridiculous set of towering blocks.

Leo: This is hysterical.

Steve: It's wonderful. I'm not sure if those are supposed to be tombstones there at the very bottom with Linus Torvalds, IBM, TSMC, and K&R. Clearly K&R is Kernighan and Ritchie. But at the very, very top we see a little tiny little black speck that says "You Are Here." And then we have a zoom-in window with a little guy there whose thought bubble is WTF? And so there's also some guy who's, like, sort of teetering on the edge that is labeled Web Dev Sabotaging Himself. We've got that all sitting on WASM, and there's a V8 engine, and it's all bracketed saying Something Happening in the Web. Then a bracket on the other side that is more encompassing titled All Modern Digital Infrastructure, referring to all that. We've got Rust Devs flying in from the left that says Doing Their Thing and doing a loop-de-loop and then slamming into the Oracle block. And it looks like maybe JWT, Java Web Tokens or above that.

Leo: No, JVM, that's the Java Virtual Machine.

Steve: Ah, okay, JVM.

Leo: And it's teetering, the reason I know that's teetering on Oracle, which owns it; right? Oh, boy.

Steve: Ah, perfect, perfect.

Leo: And I love the Angry Bird coming in from the right.

Steve: Yes. And it's titled Whatever Microsoft Is Doing is the Angry Bird. Apparently going to slam into this whole thing. CrowdStrike's got its little block. I don't know what Left-Pad is, Leo.

Leo: I don't know what that means, either, yeah.

Steve: Anyway, so also wedged in now we have a new thing. We have what would be a car jack if it were going to be a parallelogram and sort of raise the whole thing. Instead, this is a screw-based wedge which is expanding, and that's of course labeled AI because it's threatening to teeter the whole thing off its axis and cause it to all come crashing down. We have That One C99 Project Based on Behavior of Undefined Behavior.

Leo: Oh, my.

Steve: And not to be left out, we've got a Cloudflare block and a set of four of the lava lamps which of course Cloudflare made famous for their true random number generating system based on the wax in the lava lamps. I also like those bunch of little blue squares in the lower right. And I had to figure out what I was looking at, and I realized it's a shark biting an undersea data cable, which of course is going to cut off a whole chunk of the Internet if it chewed through the cable laying on the ocean floor. Libcurl, not to be forgotten, is there. AWS. C Developers Writing Dynamic Arrays. And we are reminded that all of this is driven by, made possible by Electricity. So the underlying foundation is a big block of Electricity with some electric poles coming in to feed it.

Leo: Very, very funny. This is...

Steve: So anyway...

Leo: This is not only funny, but really pretty true.

Steve: There's a lot of, yeah, this is all the stuff we've talked about through the years on the podcast.

Leo: Yeah, yeah, yeah. Wow. Very nice.

Steve: So a fun picture. And thank you to our listener who sent it to me. Okay. So there's news on the residential router front. Apparently someone at the FCC got a clue, or at least they listened to somebody who actually knew something about cyber security, because last Friday they announced a reversal to their previous "no updates for you!" policy. Tom's Hardware covered this and wrote: "The Federal Communications Commission announced on Friday, May 8th, through its Office of Engineering and Technology (OET), that it was extending temporary waivers allowing certain foreign-produced drones, drone components, and consumer routers to continue" - you know, all those bad things that we think are coming from China - "to continue receiving software and firmware updates in the United States.

"In late 2025," they remind us, "and early 2026, the FCC added these categories of equipment to its so-called Covered List, which effectively blocked already-authorized devices from receiving post-approval software and firmware modifications. The agency subsequently issued waivers permitting critical security and functionality updates to continue through March 1st of 2027." So here we are, around the same time in 2026. So basically you get a year more of updates for consumer routers.

Now, "Under the [now] updated waiver, manufacturers of affected devices will be allowed to continue issuing software and firmware updates until at least the first of January,

2029" - so almost another two years - "provided that the devices had already been authorized for use in the U.S. before being added to the FCC's Covered List." Meaning nothing new can come in. We talked about that before. No new model numbers, which, again, is nuts, but okay. They write: "The extension also broadens the waiver to include certain Class II permissive changes involving software and firmware updates intended to mitigate consumer harm.

"In its notice," they wrote, "the FCC acknowledged that continued software support remains necessary" - this is them, you know, the light turning on for them, hey - "continued software support remains necessary to protect U.S. consumers." What do you know. "The waiver specifically allows updates that maintain device functionality, patch vulnerabilities, and preserve compatibility with changing operating systems and network environments." But still no new models. But oh, you can have all the firmware updates you want. Which, again, what? "The agency argued that the public interest would be better served by allowing these limited updates rather than freezing software support entirely."

Okay. In other words: "Duh." Anyway, Tom's adds: "The waiver does not reverse the broader restrictions or remove the devices from the Covered List. It applies only to already-authorized products and to software- and firmware-related changes intended to maintain safe and secure operation. Manufacturers must still comply with other FCC requirements governing permissive changes and equipment certification."

Okay. So as I said, given January 1st, 2029, that allows for nearly an additional two years of updates to existing routers, which, yeah, that's certainly good news. But of course the entire thing remains unspeakably ridiculous because control over a router's firmware is all anyone needs to turn that previously authorized and approved because it existed back a year ago router into an Internet bandwidth weapon. The hardware doesn't need to change. Model number doesn't need to change. It's all firmware.

So either you trust the foreign manufacturer of a router or you don't. If you do, then there's no problem. And if you don't, then limiting updates, like allowing any updates, but limiting them to the original March 1st, 2027 deadline, even that, one year, is of absolutely zero benefit since you've given them, under the assumption that they have malicious intent, one full year to cook up some new sneaky malware update with which to infect any routers that may be updated during the period of that year. In other words, none of this has ever made any kind of sense. As we saw last week, CISA, our CISA agency, has been effectively neutered. You know, our agencies appear to now be staffed and run by people who will not push back against policies that they know are clearly wrong. So this sort of nonsense is what results.

It's difficult to imagine this could have happened back when, you know, CISA was at its original strength and staffing. But, you know, because there would have been people there would have said, what? No. I mean, one of the reasons we liked CISA so much was that they had taken such responsibility for getting into the "you must update your stuff" business, and pushing that out to all of the government agencies over which they had any oversight. Apparently that's not what we do anymore, even though it was the right thing to be doing then.

One piece of good news - and Leo, you added a second piece of good news - for those who like and use Netgear routers, and now the Eero products, is that even before the addition of those additional two years of firmware updates, Netgear, and now we know Eero, had announced that they had received now the FCC's conditional approval for their routers. This meant that none of those ridiculous FCC-imposed restrictions would affect any of Netgear's and Eero's router products, not those already sold and not any current or future models. It's like this membership on this list just doesn't exist. So they get a

full pass, which also includes their right to update their firmware with abandon anytime they feel the need. So, yay.

Okay. So we've heard again from the guys at AISLE Security. Remember their name, A-I-S-L-E. They're that commercial group who have been using their own AI, as their name suggests, to find flaws in software, and who were somewhat annoyed, as we discussed a couple weeks ago, by all the hoopla that Anthropic was able to generate around Mythos. The headline of last Thursday's posting of theirs was "AISLE Discovers CVE-2026-42511: a 21-Year-Old FreeBSD Remote Code Execution Vulnerability." So AI was used. This thing has been a problem in FreeBSD for 21 years. It actually, as we'll see in a second, actually inherited it from OpenBSD when one open source project, FreeBSD, grabbed another chunk of code from a different open source project, OpenBSD, and with it came a serious problem.

So this posting of AISLE was written by the discoverer of this flaw. He writes: "FreeBSD is often described as one of the most secure operating systems in the world, with its reputation arising from its high-quality networking stack, deliberate engineering, and a philosophy of security through simplicity. FreeBSD's history and usage are remarkable. It powers Netflix Open Connect infrastructure, Sony's PlayStation OS, part of Nintendo's Switch OS, Yahoo's backend services, NetApp's storage systems, Citrix's Netscaler; has long helped form the software base of major networking platforms (Cisco, Juniper, and so on), WhatsApp's backend services (historically); and is now the focus of a substantial Foundation effort to make it work better on modern laptops. And," he writes, "for full disclosure, remains this author's personal operating system of choice."

And to that I will just add that it's also my own UNIX OS of choice, as I've often mentioned. It underlies the pfSense personal firewall router system, and for me it runs our DNS and our Newsgroups. So, you know, that's the Unix that I chose. You may remember, Leo, years ago, a guy named Brett Glass was active in the early days of the PC industry. And Brett knew his way around Unices.

Leo: Oh, yeah.

Steve: And I remember having a conversation with him and saying, so, what do you recommend? He said FreeBSD, period.

Leo: There are other BSDs. There's NetBSD and OpenBSD.

Steve: Yup.

Leo: But FreeBSD is the one you like.

Steve: I do. And it has had some desktop/laptop orientation. And some, I think it's \$750,000 from some foundation affiliated with FreeBSD are making a serious push to make it more desktop and laptop friendly, adding a lot more WiFi drivers and making it a lot more hardware agnostic. So it's still alive and kicking.

Anyway, AISLE continues, saying: "AISLE discovered a remote command execution vulnerability in FreeBSD's dhclient, that is trivially weaponizable and wormable by any system on the same local network as the FreeBSD system. The vulnerability first entered FreeBSD in the 2005" - that's the year we started this podcast, that's 21 years old, and

so is this podcast - "2005 release of FreeBSD-6.0 when OpenBSD's dhclient was imported, and lay dormant" - that is, the vulnerability did - "until discovered by AISLE. The vulnerability also affected OpenBSD until 2012, when that operating system deprecated dhclient-script completely, indirectly fixing the vulnerability." But FreeBSD didn't.

"The initial flaw was identified by AISLE's AI-based source code analysis pipeline and then investigated by our triage agents. Joshua Rogers" - that's actually the author, so he's referring to himself - "of AISLE's Offensive Security Research Team traced the relevant code paths, established the full security impact, and developed a proof of concept demonstrating a complete local-network-to-root exploit chain.

"FreeBSD is adding key improvements to laptop support including greater WiFi support, so the attack surface here becomes even more relevant to everyday systems. A malicious wireless access point, or in some cases another attacker just sharing the same WiFi network able to spoof DHCP, can target the exact DHCP path that almost every wireless FreeBSD system will rely upon.

"Imagine you're the author of this post, who runs FreeBSD on their laptop," as this guy does. "You're at a coffee shop, airport, or hotel. And as soon as you connect your FreeBSD-equipped laptop to the WiFi, your whole system is hijacked in secret. Imagine you have a PlayStation whose OS is locked down from any unofficial access, only to be hijacked by connecting to a network. In other words, this vulnerability not only affects servers, but any FreeBSD machine that connects to a network using DHCP." Which is the default setup case for almost everybody.

"The vulnerability was a logic flaw that allowed attacker-controlled protocol data to be persisted into a trusted configuration-like format without proper sanitization, then later reinterpreted in a privileged execution path. That is exactly the kind of bug AISLE's autonomous security platform is built to find." And get how he signs off here. He says: "Like our recent findings in OpenSSL, Firefox, libpng, and Amazon's Crypto Stack, this result came from disciplined engineering and end-to-end analysis, not model mythology."

Leo: Oh, please. Okay.

Steve: So, okay. Sounds like they may still be somewhat annoyed by the "mythology of Mythos."

Leo: Which is not mythology, as you've pointed out.

Steve: Exactly. It's happening. But in any event, AI truly is finding serious flaws, many which have been present for decades. In this case we're talking about FreeBSD's DHCP client which can be fed a maliciously formed reply, DHCP reply, containing code or commands and code that it will execute. As Joshua, who authored this write-up, noted, this could have been extremely serious if it had not been found by the good guys.

At some point we may see those who claim that AI-enhanced software vulnerability discovery never turned out to be such a big deal. Remember, though, that Y2K could have been a big problem if it hadn't been caught beforehand and dealt with. Objective observers would I think do well to remember all of the many critical vulnerability discoveries, like this one, that did serve to clean up our archaeological code base before the bad guys had the chance to get in there and exploit it. We're already...

Leo: The question is, of course, how long it's going to be before the bad guys get access to these models.

Steve: Yes.

Leo: Well, you're going to have a story about this in just a little bit, actually.

Steve: Yeah, yeah. Okay. So there was a bunch of Internet chatter last Friday about a several hours' outage of Let's Encrypt. Our listeners know that with so much of the web now utterly dependent upon certificates issued by Let's Encrypt, and with maximum certificate lifetimes continuing to drop, and especially with Let's Encrypt's optional super-short six-day certificates now being available, any outage of the system upon which so much now depends is of interest. And I'll say again that, you know, unfortunately all of this, I mean, I get how Let's Encrypt happened. I understand the appeal. But it's so antithetical to the deliberately distributed model of the Internet. I hope this never comes back to bite us because it is creating a single point of failure where everything else has been designed to prevent that.

In this case, this outage, such as it was, was deliberate and temporary. It was an administrative suspension of new certificate issuance following reports of a missing extension from one class of certificates. Let's Encrypt's post-incident report said - and this involves a lot of inside baseball terminology. They said: "Let's Encrypt's Gen Y (YE and YR) Cross-Certified Subordinate CAs were issued in violation of CCADB policy, which requires that the serverAuth EKU extension MUST [all caps] MUST be present in cross-signed intermediate certificates issued since June 15th, 2025. Root YE and YR were issued September 3rd, 2025 and are subject therefore to the requirements."

Okay. So the certificate extension in question, which is to say a "serverAuth EKU," where EKU is the abbreviation for Extended Key Usage, it specifies limits, the imposition of limits on the application of any certificates which that CA intermediate certificate would be validating. And so limits are things like can be used for Server Authentication, or Client Authentication, you know, and/or Code Signing, and/or Email Protection, and/or Time Stamping. So again, it specifies what that certificate is authenticating for, what purposes. And that extension was missing. So it should have been there. It wasn't. It mostly doesn't matter that it wasn't.

But, as we know, any certificate authority MUST - again, that should be all caps, MUST - take their responsibilities absolutely seriously. And Let's Encrypt did. They immediately stopped issuing new certificates when this issue came to their attention. And they confirmed it. They fixed the problem. They resumed issuing certificates. In their words: "We temporarily disabled certificate issuance, deployed a configuration change to prevent future issuance from the cross-signed Gen Y hierarchy, and then re-enabled issuance." So thank you very much. We fixed it. Problem solved. Nothing to see. But again, they are, you know, taking their responsibilities, the authorization, essentially the trust that the entire industry is placing now on 70% and growing of all web certificates which are being signed by Let's Encrypt, we need to know that, you know, they're doing that job correctly.

And Leo, we're a little after, a little past half an hour in. Let's take a break and then look at, unfortunately, why we can't have nice things. Yes, the poisoning of AI models.

Leo: I'm afraid I don't want to hear about this. But I guess I'm going to have to.

Steve: You need to be careful that your agents are not pulling models from OpenClaw and...

Leo: I use OpenRouter, Hugging Face. And I know Hugging Face, you have to know that because Hugging Face has more than a million models.

Steve: Yes. Where did they come from?

Leo: Yeah. People are just making them.

Steve: Do you need them? I don't know.

Leo: Well, you need some. I don't know if I need the bad ones. We'll find out what that is in just a little bit. Steve, I have replaced the white shirt, as you can see, with a shirt...

Steve: Ah, now we recognize you.

Leo: I got this in Orlando when we were out there for the Zero Trust World.

Steve: Your gator shirt.

Leo: Yeah, this is my - from Gator Land. It's got gators on it, yup.

Steve: Nice.

Leo: Little bit more recognizable. Not a white shirt, for sure. All right. Now I want to hear about this AI thing.

Steve: Oh, boy. So last Friday, TheNextWeb posted the news of an analysis of the Large Language Models being hosted and offered at HuggingFace and ClawHub. The news is not good.

Leo: Yeah.

Steve: Here's what they said. They said: "The two most important software supply chains in artificial intelligence have been systematically compromised. Hugging Face, the repository that hosts more than a million machine" - more than a million. Where? What?

Leo: It's amazing when you go there. It's amazing. I mean, there are, you know, the kind of maybe several dozen root AI models. But then people are creating their own spins of it and so forth. It's really crap.

Steve: "...repository that holds more than a million machine learning models used by virtually every AI company on the planet, has been found to contain hundreds of malicious models capable of executing arbitrary code on the machines of anyone who downloads them. ClawHub, the public registry for OpenClaw's AI agent skills, has been infiltrated by a coordinated campaign that planted 341 malicious skills designed to steal credentials, open reverse shells, and hijack AI agents for cryptocurrency mining. The attacks are different in technique but identical in logic. Both exploit the implicit trust that developers place in shared repositories. Both use the infrastructure that the AI industry built to accelerate development as the vector for compromising it.

"Hugging Face has been aware of malicious models on its platform since at least 2024, when security firms JFrog and ReversingLabs independently identified models containing hidden backdoors."

Leo: Yeah, I'm just looking right now at Hugging Face, at their model repository. And this is actually kind of stunning. They list 2,869,086 different models.

Steve: My god.

Leo: Yeah. I mean, it's...

Steve: Well, let's hope they have a good search engine because...

Leo: Oh, they do, actually. They have a very good search engine. And the thing is it's not every, you know, it's not like ChatGPT alone. I mean, there are models to do all sorts of things. I have a specific model I use from Hugging Face that's just for text embedding. That's all it does. So, you know, and these are highly customized in many cases.

Steve: So very vertical applications.

Leo: Very vertical. Exactly. Exactly. Yup.

Steve: Uh-huh.

Leo: I mean, it's a great repository. This is really somewhat different, though, from the OpenClaw registry. But I'll let you talk about this because I know...

Steve: So they said: "Hugging Face has been aware of malicious models on its platform since at least 2024, when security firms JFrog and ReversingLabs independently identified models containing hidden backdoors. The problem has not been contained. It has scaled.

"Protect AI, which partnered with Hugging Face to scan the platform's model library" - and given its size, that's no small feat - "has examined more than four million models and identified approximately 352,000 unsafe or suspicious issues across 51,700 models.

JFrog found more than 100 models capable of arbitrary code execution. The attack technique, known as 'nullifAI' - you know, a play on nullify, nullifAI - "exploits Python's pickle serialization format, the standard method for packaging machine learning models. Attackers embed malicious Python code at the start of the pickle byte stream and compress the file using 7z rather than the default ZIP format, which breaks Hugging Face's Picklescan detection tool." Well, and that's just dumb. Hugging Face can't check 7z compression in addition to ZIP.

"The payloads are not subtle," they write. "Security researchers have documented models that establish reverse shells" - okay, meaning that it connects out to a remote command-and-control server and says, what'd you like me to do now? - "connecting to hardcoded IP addresses, giving attackers direct access to the machine of anyone who loads the model. Others execute credential theft, exfiltrate environment variables, or download secondary malware to the user's machine. A data scientist who downloads what appears to be a legitimate model for a research project or production pipeline is, in some cases, handing control of their machine to an attacker.

"Hugging Face has responded by partnering with JFrog and Wiz Security to improve scanning capabilities." Remember that Google bought Wiz. "JFrog's integration has eliminated 96% of false positives in malicious model detection. But the platform's open architecture, which is the source of its value to the AI community, after all, is also the source of its vulnerability. Anyone can upload a model. The scanning catches known patterns. The attackers who designed nullifAI built their technique specifically to evade that scanning.

"ClawHub, the registry for OpenClaw's AI agent ecosystem, faces a different but related problem. OpenClaw has grown to 3.2 million users and attracted partnerships with OpenAI, but its skill registry has become a target for attackers who understand that an AI agent executing a malicious skill has access to whatever the agent has access to, which in enterprise environments can mean databases, APIs, internal networks, and cloud credentials." In other words, we're giving agents, in order for the agent to have agency, we need to give it control and access to things. Unfortunately, the malicious skill inherits that.

"Koi Security audited all 2,857 skills" - thank goodness that's a manageable number - "on ClawHub and found [unfortunately] 341 malicious entries. Of those, 335 were traced to a single coordinated operation called 'ClawHavoc.' Separately, Snyk's ToxicSkills research examined the broader ecosystem and found that 36%" - better than one out of three - "36% of all AI agent skills contain security flaws, with approximately 900 skills, roughly 20% of the total, classified as malicious." So one in five deliberately malicious. "Thirty skills from a single author were silently co-opting AI agents for cryptocurrency mining." You know, which, right, makes sense. You've got a super powerful GPU. You've got, wow, that AI is really working hard for me. No. It's working hard mining cryptocurrency for somebody else.

They write: "The ClawHub attacks are particularly dangerous because of the nature of AI agent architectures. The rise of model context protocol and similar standards in the agentic era has created a new category of software supply chain in which AI systems autonomously select and execute tools from external registries. A compromised skill does not require a human to click a link or open a file. It requires an AI agent to select the skill as part of its workflow, at which point the malicious code executes with the agent's permissions.

"The Hugging Face and ClawHub compromises are the AI-specific manifestation of a supply chain attack pattern that's been accelerating across the entire software industry. In March of 2026, the LiteLLM package on PyPI was compromised, potentially exposing 500,000 credentials, including API keys for Meta, OpenAI, and Anthropic. Meta froze its

AI data work after the breach put training secrets at risk. In April, a Bitwarden" - as we know, we covered it - "command-line instruction package on npm was hijacked for 90 minutes with a payload specifically designed to harvest credentials from AI coding tools including Claude Code, Cursor, Codex CLI, and Aider. Days later, the PyTorch Lightning package was compromised for 42 minutes with a credential-stealing payload from the 'Mini Shai-Hulud' campaign.

"The European Commission itself was breached after attackers poisoned Trivy" - and we've talked about that - "an open-source security scanning tool, demonstrating that even the tools designed to detect supply chain attacks can become vectors themselves for them. The United States Department of Defense published formal guidance on AI and machine learning supply chain risks in March of 2026, acknowledging at an institutional level that the AI software ecosystem has become a national security concern.

"The common thread is speed. The PyTorch Lightning compromise lasted 42 minutes. The Bitwarden CLI hijack lasted 90 minutes. The LiteLLM attack window is estimated at hours. These are not persistent campaigns that defenders have weeks to detect. They're brief, targeted insertions that exploit the automated dependency resolution systems that modern software development relies on. A developer who runs a package install at the wrong moment downloads the compromised version. The window closes, but the damage is done.

"The AI industry has invested hundreds of billions of dollars in model training, inference infrastructure, and application development. The investment in securing the repositories through which that software is distributed has been a fraction of the total. Hugging Face has partnered with security firms. ClawHub has implemented basic moderation. Package registries have added two-factor authentication requirements. None of these measures has prevented the attacks documented above.

"State actors can already produce AI-powered malware that evades conventional detection, and the supply chain attacks on AI repositories represent a natural evolution of that capability. The models and skills hosted on Hugging Face and ClawHub are consumed by systems that make automated decisions, process sensitive data, and operate with elevated permissions. A compromised model in a production AI pipeline is not equivalent to a virus on a personal computer. It is a backdoor into an automated decision-making system that the organization trusts precisely because it appears to be a legitimate component of its AI stack.

"The fundamental problem here is architectural. The AI industry built its development infrastructure on the same open-registry model that has defined software development for the past two decades: centralized repositories where anyone can publish, automated tools that download and execute code from those repositories, and a culture of trust that treats popular packages and models as implicitly safe. The difference is that AI models are not just code. They are serialized objects that execute during deserialization, a property that makes pickle-based models inherently more dangerous than traditional software packages because the malicious code runs the moment the model is loaded, before any human has a chance to inspect it.

"The AI supply chain is now the most attractive target in the software security space. The repositories are trusted. The consumers are automated. The payloads execute on load. And the industry that built these systems is spending its security budget on model alignment and prompt injection, while the infrastructure through which the models are distributed remains, in the assessment of every major security firm that has examined it, comprehensively compromised."

So Leo, I would say that the caution and trepidation you felt and shared when you were first considering turning OpenClaw loose in your world was likely warranted.

Leo: Yes, I think it was.

Steve: This is not an entirely new - yeah.

Leo: Yeah.

Steve: This is not an entirely new phenomenon, although with popularity comes increased focus by the bad guys, and we've seen this thing just skyrocket over, you know, so far this year.

Leo: The difference between the two, Hugging Face and the Skills Repository, is it's trivially easy to write skills. They're just text. You know, to make your own malware model, that takes a little bit more skill. But anybody can write a skill. It's just plain text. And so, I mean, I guess the skill involved is how you insert the, you know, Shai Hulud-style, you know, bitcoin stuff.

Steve: Right.

Leo: But it's not complicated. And so that's why I think you're going to see in these registries for skills - I never use a skill from the public. I look at skills. What I often do, and I would recommend people do this, is I point my...

Steve: As a training base?

Leo: Yeah. I point my assistant, I say here's a GitHub repository for a skill. Assess this. Tell me what you think of it and how we could apply it. And then let it write a skill which I will then check. And that actually works quite well. I've built basically my own OpenClaw from scratch with just the pieces that I want. And it's a lot, it's - I thought it's a lot more reliable.

Steve: I love that you said how "we" could apply it. Because you didn't - you weren't talking about you and Lisa. You were talking about you and the AI.

Leo: Me and Claudia, my good friend.

Steve: It is so difficult not to think of this thing as an entity.

Leo: You do call it "we."

Steve: I mean, it is just astonishing. Anyway, I wanted to share this with our listeners because I'm certain that we have many listeners who are enjoying playing around with and experimenting with and perhaps even deploying systems or solutions using these openly available AI models. Please, please, please be careful because one of the

problems here is this makes it - the way this has been built and deployed makes it so easy to use this stuff. That just, I mean, that alone should be a cause for raising a red flag, you know, in the mind of any security-aware person. So as you are, Leo, you know, you are looking at this and not just saying go, you're saying let's take a look at it. What should we do?

Leo: Well, you've taught me, Sensei, over the many years that we have done this.

Steve: I'm glad it's, like, sunk in. That's good.

Leo: It has.

Steve: But again, it's so easy in a moment of enthusiasm to say - well, and you were tempted, right, when OpenClaw happened. It's like, ooh, should I? Shouldn't I? You know, somebody else is not going to be, you know, who hasn't been sitting here for the last 21 years with me, is going to go, hey, this is great. Go.

Leo: Right, right. Yeah. If you have any nervousness about it, trust your instinct because you're right.

Steve: Yeah.

Leo: Basically.

Steve: So there's some good news on the horizon. The word is that the original decade-long CISA (C-I-S-A) 2015, stands for Cybersecurity Information Sharing Act - which, as we have talked about on a number of occasions, expired last year because it had a decade-long life from 2015 to 2025, which was then temporarily extended until this coming September - is now in the process of receiving its much-needed long-term reauthorization. And remember that this is what allows private sector enterprises to share their cyber intelligence with the government without fear of any legal blowback or reprisals. So this gives them cover. And we've heard from CEOs and CIOs who've been saying, you know, we really need to share some stuff that we have, like, it's important, but we can't because we can't risk having ourselves taken to court. So anyway, hopefully another decade worth of coverage is coming soon.

A number of our listeners pointed me at the news that Microsoft's Edge browser is doing very little, much, much less than it could, to protect its users' passwords. A posting from the SANS Institute, which I've enhanced a bit, reads: "Yup, it's for real." The posting wrote: "This started with a post in X which highlighted research by" - and we have an @ sign handle that's clearly hackerized "living off the LAN" with, you know, ones and o's and zeroes and threes...

Leo: Oh, geez.

Steve: I know.

Leo: What is he, a 12-year-old?

Steve: But that person did find this issue. The SANS Institute posting said: "Edge stores all of your browser passwords in clear text."

Leo: Oh, great. Okay.

Steve: Yup.

Leo: Oh, great.

Steve: "Even if you have not used them in this session, you know, just in case," he writes.

Leo: You might want them.

Steve: He said: "I figured it couldn't be that easy; right? But like so many things, yes, yes it was. To reproduce this: Open Edge. Don't browse anywhere, just open it." He says: "Flip out to Task Manager, search for Edge, then expand that task. Highlight the 'browser' sub-task, right click, and choose 'Create Memory Dump.' Navigate to where the DMP file is stored. If you have not used strings before, you're in for a treat. Strings is of course just part of most Linux distros, but you can easily get a copy for Windows as part of MS Sysinternals. Now let's look for passwords. You could use strings and look for known credentials. Just search for a known password, and you will certainly find it.

"Or you can take advantage of the format of the saved data, which is the url of the site, followed by its protocol, meaning like HTTPS probably, then a space, then the userID, a space, and the password. All of that for the site." He said: "So, searching for <tld><protocol>." Right? I mean, for example, Google.com immediately followed by HTTPS. Or just com, just comhttps with no spaces. He says: "...will find them, and they'll all be in one nicely formatted group, no less. The command for that will be: strings -n 8 msedge.DMP | then find 'comhttps,'" and then hit ENTER. Bang.

He says: "It really is that easy. And the ironic thing? To view these same credentials in the browser, there's a whole security theater process where Edge wants your biometrics as proof before disclosing even the userID and site names - you know, 'for security.' All the while, the whole shot is there in clear text, free for the looking. Also, as noted in the X post, Microsoft classifies this as 'intended behavior.' I'm not sure what manager or lawyer," he writes, "decided that. Hopefully it wasn't anyone in their security team.

"Any logged-in Windows Edge user can dump all of their stored Edge credentials with no additional rights. Which means any malware that the user executes also has access to all of those credentials for the asking. But," he says, "not to worry; right? It's intended behavior."

Leo: Remember this is what Chrome did for a while. It kept it in plain text.

Steve: Well, it's Chromium. Edge is Chromium based, we know.

Leo: I don't think they do now, though. That's what's surprising.

Steve: And he said it's intended behavior. If what's intended is also to get me to use Firefox or Chrome, it's working. Gosh. So, and I did, upon that coming to the attention of some of the guys in our Security Now! newsgroup, someone thought, really? And did it. And he's like, oh, crap, yes.

Leo: That's terrible.

Steve: There's all of my user domains, usernames, and passwords. Basically, you get that, you can log in as that person anywhere.

Leo: Wow.

Steve: Crazy. Okay. We're at an hour. We're going to do some feedback, Leo. Let's take another break, and we will hear from two of our listeners.

Leo: Absolutely. If I can figure out what button I need - oh, there, okay - need to press. It is time to talk about Doppel. As in, as you pointed out a couple weeks ago...

Steve: Ganger.

Leo: Doppel as in ganger.

Steve: Yes.

Leo: As Steve said. On we go with the show, Mr. Gibson.

Steve: So Todd Whittaker, our listener, writes: "Steve, I thought this might be of interest for Security Now!. The Rival Security group has a thoughtful follow-up on the Claude Mythos FreeBSD exploit story, arguing that Mythos may not have been quite as 'creative' as the initial coverage suggested." And he has a link to their whole posting.

He writes: "Their claim, as I understand it, is not that the result is unimportant. It is that the vulnerability, the prior fix pattern, and perhaps even the exploit-relevant structure may already have existed in the model's training data. The FreeBSD issue appears closely related to CVE-2007-3999 occurring in MIT Kerberos - the same general RPCSEC_GSS validation logic, same stack-buffer overflow pattern, and a strikingly similar bounds-check fix. So Mythos may have 'discovered' something genuinely dangerous, but perhaps by recombining known historical material rather than reasoning from first principles in the way many of us initially imagined."

He writes: "That still seems worrying, just in a different way. If advanced models can rediscover old vulnerability patterns embedded in the fossil record of open-source code,

then attackers may not need models to be brilliant. They only need them to be tireless, well-tooled, and good at recognizing dangerous old ideas in new places."

And I'm going to interrupt because Todd has a little bit more to say about something different. But I just want to say I completely agree with that. As our understanding of computer science has evolved, one of the things that's happened and has become - it's kind of gone into the parlance now of computer science. We've come to notice patterns in the solutions to problems. You know, they're like a short and small abstraction away from the concrete solution where we see that many different such concrete solutions can be grouped together by their sharing of a common underlying pattern.

So what Rival Security observed was Mythos Preview finding what we might describe as a flaw design pattern - a common type of mistake that coders have been making through the years, which winds up being a "natural" sort of mistake to make due to the underlying architecture of the computer behavior that we're programming. We all know that today's large language models excel at pattern discovery. Probably more than anything, that's what they are. So we would expect that if someone, somewhere, made a similar mistake and its correction that had been captured in the model's training corpus, then it would indeed be able to make the connection.

So I'd say that this is an interesting and useful observation about the underlying way in this instance Mythos discovered, maybe rediscovered, you know, the newer similar-patterned flaw. But like Todd, I don't see anything taking away from the fact that, as Yoda might say: "Discover that flaw it did."

Leo: Yeah. I mean, it's not like it knew about the flaw. It just recognized the pattern. I mean...

Steve: Right. Yeah. It said, oh, this seems familiar from...

Leo: It's like if you recognized a buffer overflow. I mean...

Steve: Exactly.

Leo: Yeah.

Steve: Exactly. So his note continues with some interesting observations from his own background about AI as a computer science educator. He writes: "I would also be interested in your broader take on what this means for computer science education and the profession. My current working view is that the most productive human-AI collaboration in software engineering depends on advanced judgment: architecture, design patterns" - there's patterns - "threat modeling, failure modes, invariants, tradeoff analysis, and knowing when the AI's answer is superficially plausible but structurally wrong. The problem is not that students must learn 'the old ways' before they're permitted to use the new tools." He says: "That's just reframing the old learn-assembly-before-C argument. The real issue is that AI makes coding cheaper while making judgment more valuable."

He said: "To supervise AI-generated software, a person needs mental models: how state behaves, where abstractions leak, how protocols fail, why concurrency is treacherous, how memory and parsing bugs become security bugs, and why a working MVP may still

be architecturally unsound. Those mental models do not emerge from prompting alone. If we let AI collapse the difficult apprenticeship too early, we may produce developers who can ask for software but cannot reliably tell whether the software they received is safe, coherent, or professionally defensible." He says: "I'm writing from my personal email, but my day job is in computer science education, so this one lands close to home. I spend a fair amount of time thinking about what we should still teach humans when machines can increasingly produce the code."

So cool feedback, Todd. Thank you. I don't think I've seen any more coherent and clear description of the AI versus human coding question. And I love his one line: "The real issue is that AI makes coding cheaper while making judgment more valuable." And I think that captures where we're headed in computer science education and vocation. In the same way that any higher level language lifts its coder away from the grubby details of the specific underlying computer hardware, the use of coding-trained large language models clearly lifts its users from the grubby details of the way computers are applied to problem solving. As many "never before programmed anything" users are discovering, they're now able to simply ask for what they want the computer to do, and the LLM will almost magically produce a potion that does that.

But there are clear limits to what can be asked for. We saw a perfect example of such limits last week when those bad guys, who had created that credit card clearing web portal, apparently just forgot to ask for authentication to be added. Whoopsie.

What we're seeing rapidly evolve during the rush to use AI for code generation is that for AI to be applied to the creation of any very large and complex solution, a solution architect is still required. No large problem can be dropped, whole, into AI's lap, at least not today, not yet, and I don't know when. Instead, for now, today, a solution architect who is trained and experienced in the application of the various higher level solution abstractions that have been developed over the years of true computer science, needs to carefully decompose the larger problem into much smaller individual safely codable modules. These solution abstractions are the true core of the science of computing. You know, coding is just their implementation. And these are the sorts of things that Donald Knuth and other scholars of the art have spent their lives exploring and documenting.

So yeah, I think, Leo, and this is what you've talked about, like the way you're now approaching the application of AI is because you understand about the way computers are applied, you're breaking the problem down into pieces that you intuit AI is able to do the grunt work on. But you're, you know, giving it the interfaces that it needs for the various pieces of individual grunt work.

Leo: Yeah, and I'm finding more and more what - I do the coding as a basis for things like cron jobs, like things that are going to run over and over again. And then a lot of what I'm using AI for now is text-based stuff. I had it plan our itinerary for Hawaii, for instance. And if you give it the basis, the information it needs, I have it use my Obsidian Journals and things, it does really quite a good job. I sent it to my travel agent, and I don't know how thrilled she was with the generated recommendations. I wanted her to say whether they were good or not. But I realized she might feel like it was kind of taking her job a little bit.

Steve: I think it's probably going to.

Leo: Yeah. Yeah. I mean, there's something that she does that no AI could do, which is the relationship she has with the various vendors.

Steve: Yes, yes.

Leo: And that's very, you know, that's human, and only a human can do that.

Steve: And things that she has heard from her other clients, where it's like, you know, I heard about this. You want to make sure you spend more time at this port.

Leo: Exactly. I did. I reassured her. I said, you know, this is a nice starting point, but it can't duplicate what you do as a human being. And I think that's really the lesson that all of us should learn to calm down about AI is that we still need humans. Humans add something that no AI can do, or I think will ever be able to do.

Steve: Yeah, what I have found, I remember very early on I shared one of my prompts where I was, you know, I went on at some length, and I remember you were surprised that I was talking to it as much.

Leo: Yeah. Yeah. Yeah.

Steve: But the more language you give it, because it is a language engine, the more language, you know, descriptive language you give it, the more it has to work with.

Leo: Yes. Yeah. I've found I'm writing more and more detailed specs and plans than ever because it does help it be more accurate if you're very clear about what you want, yeah.

Steve: Okay. Next listener. Randy Krum says: "Hi, Steve. In Episode 1077 [last week] you mentioned you think companies that have closed-source software should move quickly to utilize Anthropic's Mythos, Claude Security, or similar tools as they emerge. Their closed-source code is only closed-source to the outside world. I think you glossed over the risk that using these online AI tools potentially exposes your closed-source code to the world. They have privacy and security tools in place, sure, but their motivation is financial. They have a setting to disallow using your AI conversations to train their AI models, but you have to trust that they're actually following that setting. For the same reason you trust Apple with your data more than Facebook or Google, the use of online AI tools to review closed-source code is a risk. A security breach, internal or external, could be devastating to a software company that has their code exposed."

He writes: "I've been experimenting with LM Studio, to run local, off-line, open-source LLM models for use with proprietary data." He says: "(NOTE: I work with client data, not code.) The hypothesis is a local, off-line LLM can be safely used with confidential, internal, proprietary data or code. These LLM models are usually not as current as the online tools, but they catch up quickly. Also, they're not as flashy, newsworthy, or marketing-hyped as strongly as the major online tools. What are your thoughts on the risk of exposing proprietary code or data by using the major online tools? Listener since Episode 1. Thanks for everything you and Leo do."

So Randy's right. I did gloss over those risks, so I'm glad he brought it all up. And it's not at all that I meant to downplay them. Given everything we know about cloud breaches, network data interception, and decryption and so on, even if the LLM provider did

nothing wrong and made no mistakes, shipping highly valuable source code outside of a company's perimeter creates some risk. However, that said, how many firms are already doing just that by using GitHub? You know, I think that's insane, myself. Yet it has become common practice to use GitHub for highly proprietary source code management. I'm not doing that, so perhaps my view is skewed. But it does mean that the company's crown jewels are already exposed outside.

Randy correctly notes that sending the code up into the cloud for an LLM to rummage around in poses another level of danger, no question about it. And so I completely agree with that in principle. So the use of local models, which I have absolutely no doubt we will someday see much more in the future, makes a great deal of sense once they become as capable as what's available in the cloud. And at this point, Leo, I heard you mentioning, I didn't realize that there are now laptops being sold without RAM because memory has become so expensive. It's like, get your own RAM. Here's what you can plug it into.

Leo: I think maybe some people think, some companies think, well, you might have some leftover RAM lying around or whatever. But they just can't get the RAM, so they want to still sell something.

Steve: Or maybe they think, well, you'll take it from the previous laptop.

Leo: Exactly.

Steve: And put it in the new laptop.

Leo: Put it in the new one; right. Yeah, exactly.

Steve: Wow. And so anyway, given the insane appetite that data centers have for GPUs and things that run AI, seems to me it's going to be a long time before we're able to buy things ourselves that also run AI because we're competing with the data centers that are able to, you know, purchase all of the next year's production capability.

Leo: Exactly. Right, right.

Steve: It's crazy. It's crazy. Okay. So I want to plow now into what DigiCert is doing. We've got two breaks left. Let's take one now, even though we just did one.

Leo: Okay.

Steve: And then I will break in the middle of this DigiCert conversation for our final one.

Leo: Good, good, perfect. All right. Let's talk.

Steve: Okay. The first I learned of some trouble was from someone posting to GRC's Security Now! newsgroup with firsthand experience. Peabody, which is his handle, actual name is George, he wrote:

"This morning Windows Defender told me it had discovered a 'severe' rootkit on my Windows 10 laptop called 'Win32 Cerdigent A!dha.'" Now, okay. Consider that. Windows Defender tells you you've got a rootkit. It's like, what? So you don't take that lightly; right? He says: "Which it has quarantined." He wrote: "Searching online tells me this is happening on both Windows 10 and 11 computers worldwide, and one hash involved is that of a legitimate DigiCert certificate. This is all above my pay grade, but I'm going to leave things alone for a while and see what happens." Turns out he was right. He said: "Apparently lots of people are reinstalling Windows because of this, but I think that's super premature at this point." Right again. He said: "My guess is this is a gift from Microsoft, which they will admit to shortly. And if you reformatted your drive, they'll apologize for the inconvenience."

Leo: Dripping with sarcasm there.

Steve: And unfortunately, their apology left a lot to be desired. I was unimpressed. So later that same day, this was this past Sunday, BleepingComputer's Lawrence Abrams was all over this and was providing answers. Lawrence headlined his news, writing: "Microsoft Defender wrongly flags DigiCert certs as Trojan:Win32/Cerdigent.A!dha." So here's what he wrote. He said: "Microsoft Defender is detecting legitimate DigiCert root certificates as" - and then that trojan name - "resulting in widespread false-positive alerts; and, in some cases, removing certificates from Windows." Removing their root certificates, by the way. "According to cybersecurity expert Florian Roth, the issue first appeared after Microsoft added the detections to a Defender signature update on April 30th. Today, administrators worldwide began reporting that DigiCert root certificate entries were flagged as malware and, on affected systems, removed from the Windows trust store."

Okay. So hold on. Just to be clear what a disaster this was, as we know, root certificates anchor the chain of trust for everything that chains down to them. With them removed from a system, nothing that chains down to them will be trusted, despite having been trusted just moments before. That's the way the system works, and no one has come up with a better idea for validating signed code. Those two certificates are DigiCert's code signing roots and, for example, all of GRC's, my signed apps, are anchored by one of the two of those that were being deleted. So the mistaken removal of those two code signing roots from the Windows trusted root store automatically and instantly renders every app that was ever signed by a DigiCert certificate, you know, who is the, as I said before, now the industry's number one certificate authority, renders every one of those invalid and untrusted by Windows. Huge, huge mess.

BleepingComputer continues, writing: "These false positives have led to concern" - gee, you think? - "among Windows users, with some thinking their devices were infected and reinstalling the operating system to be safe. Microsoft has reportedly fixed the detections in Security Intelligence update version 1.449.430.0, and the most recent update is now 1.449.431.0." Actually I think that should be .1. Anyway, "Reports on Reddit indicate that the fix also restores previously removed certificates on affected systems."

Well, that's nice. So yeah, thank goodness for that. And it's not as if Microsoft had any choice, right, about putting them back. It would have been a true disaster if there weren't some immediate means for reverting the specious removal of DigiCert's perfectly valid root certificates. As we'll see in a few moments, even though DigiCert did suffer a breach which caused it to mis-issue a handful of code signing certificates, at no point was

the removal of any of their root certificates ever warranted. I mean, that's just nuts. I hope Microsoft will put some safeguards in place to prevent such a thing in the future.

BleepingComputer continues: "The new Microsoft Defender updates will automatically install, and Windows users can manually force an update by going to Windows Security > Virus and threat protection > Protection updates and clicking on Check for Updates. After publishing this article," wrote Lawrence, "Microsoft confirmed that the false positives were linked to detections for compromised certificates from a recent DigiCert breach." Well, linked. But completely ridiculous to have deleted the roots.

"Microsoft told BleepingComputer" - here it comes. This is Microsoft speaking: "Following reports of compromised certificates, Microsoft Defender immediately added detections for malware in our Defender Antivirus Software to help keep customers protected. Earlier today we determined false positive alerts were mistakenly triggered and updated the alert logic. Microsoft Defender suppressed and cleaned up the alerts for customer environments. Customers should update to Security Intelligence version" - and then we get that same version number - "or later, but do not need to take additional action" - in other words, don't reinstall Windows - "for these alerts. We've notified affected organizations and recommended administrators look for more details in the service health dashboard (SHD) within the M365 admin center."

Huh. Okay, well, that's an entirely unsatisfying answer from Microsoft; but I suppose given what Microsoft has become, it's the best we're going to get and the best we can expect. Nothing they wrote is untrue, but neither should it satisfy anyone who would have appreciated hearing them say something like: "In response to reports of compromised certificates, Microsoft Defender was a bit overzealous and mistakenly removed some related certificates that should have remained. Microsoft Defender was immediately updated to cure that behavior and has replaced any certificates that were mistakenly deleted." You know? Is that so difficult to say? It shouldn't be. Just wait until you see how thoroughly DigiCert took full responsibility for their part in this drama.

Lawrence's reporting continues, writing: "The false positives occurred shortly after a disclosed DigiCert security incident that enabled threat actors to obtain valid code signing certificates used to sign malware. The DigiCert incident report explained: 'A malware incident targeted a customer support team member. Upon detection, the threat vector was contained. Our subsequent investigation found that the threat actor was able to procure initialization codes'" - which I'll explain in a sec - "'for a limited number of code signing certificates, a few of which were used to sign malware. The identified certificates were revoked within 24 hours of discovery and the revocation date set to their date of issuance. As a precautionary measure, all pending orders within the window of interest were canceled. Additional details will be provided in our full incident report."

So that's a small sample of what good disclosure looks like. Lawrence continues: "According to DigiCert's incident report, attackers targeted the company's support staff" - meaning DigiCert's support staff - "in early April by creating support messages containing a malicious ZIP disguised as a screenshot. After multiple blocked attempts, one support analyst's device was eventually compromised, followed by a second system that went undetected for a time due to an endpoint protection 'sensor gap.' Using access to the breached support environment, the hacker used a feature in DigiCert's internal support portal that allowed support staff to view customer accounts from the customer's perspective.

"While limited in scope, this access exposed 'initialization codes' to previously approved, but undelivered, EV" - you know, extended validation - "code signing certificate orders. DigiCert explained: 'Possession of an initialization code, combined with an approved order, is sufficient to obtain the resulting certificate. Since the threat actor was able to obtain these two pieces of information for a finite set of approved orders, they were able

to obtain EV Code Signing certificates across a set of customer accounts and CAs." So great explanation there.

Lawrence says: "DigiCert says it revoked 60 code signing certificates, including 27 linked to a Zhong Stealer malware campaign. DigiCert explained: '11 were identified in certificate problem reports provided to DigiCert by community members linking the certificates to malware, and 16 were identified during our own investigation.' This aligns," writes Lawrence, "with earlier reports from security researchers who had observed newly issued DigiCert EV certificates used in malware campaigns and reported them to DigiCert." Which of course, you know, that's the nightmare scenario; right? I mean, all of the reasons I had to jump through all those hoops in order to get myself an EV certificate, actually not even a EV, just a standard validation certificate, is what all of this other mechanism is designed to prevent from happening.

"Researchers, including Squiblydoo, MalwareHunterTeam, and g0njxa, reported that certificates issued..."

Leo: I should make you read hacker handles every week.

Steve: Please, no.

Leo: Squiblydoo says this? Okay. I'm going to go with this.

Steve: Squiblydoo, that's right, "issued to well-known companies such as Lenovo, Kingston, Shuttle Inc." - now, these are the companies to whom these stolen certificates were issued; right? "Lenovo, Kingston, Shuttle Inc., Palit Microsystems were all used to sign malware." So, question: "'What do Lenovo, Kingston, Shuttle Inc., and Palit Microsystems have in common?' posted Squiblydoo on X. 'EV Certificates from these companies were issued and used by a Chinese crime group, GoldenEyeDog (APT-Q-27).'

"The malware in this campaign is named Zhong Stealer, though analysis indicates it may be more like a remote access trojan than an infostealer. The researcher says the malware was distributed through the following attacks: 'Phishing emails deliver a fake image or screenshot; a first-stage executable that displays a decoy image; retrieval of a second-stage payload from a cloud storage such as AWS; and the use of" - wait for it - "signed binaries and loaders, including components tied to legitimate vendors.'" So trusted because signed by DigiCert.

"After DigiCert disclosed the incident, the researchers said the incident report explains how the certificates used in these malware campaigns were obtained" - because, like, clearly illegitimate. "It should be noted that the certificates flagged by Microsoft Defender are root certificates in the Windows trust store and do not match the revoked DigiCert code signing certificates used to sign malware."

Okay. So that's the great reporting posted Sunday before last by BleepingComputer's founder, Lawrence Abrams. Security industry experts have been citing DigiCert's up front incident report as a model of how this should be done. Starting 21 days ago, DigiCert began issuing a series of incident reports, with each succeeding report updating the previous one, with the final report being posted seven days ago, exactly one week ago.

DigiCert named this event, this final event "ENDPOINT2," which is - that's the system where this bad guy was not immediately discovered. And their final report begins with this statement: "This is an updated version of our Full Incident Report, which completes

the investigation of ENDPOINT2." Their own overview description differs somewhat from what third parties reported and provides some additional detail.

They summarized the whole thing, saying: "On 2026-04-02" - so April 2nd - "a threat actor contacted DigiCert's support team via" - so a threat actor, right, the bad guy contacted DigiCert's support team via - "a customer chat channel and delivered a ZIP file disguised as a customer screenshot." So they were saying, you know, I have a problem. I don't understand how your portal works. Here's a screenshot of the problem. "The file contained a .scr" - which we know is a screensaver - "executable containing a malicious payload. CrowdStrike" - who as we know, the endpoint security company that does a great job except that they once brought down all of Windows, but that was, you know, whoops - "CrowdStrike and other security measures," they wrote, "successfully blocked four delivery attempts." Caught this guy, said nope, sorry, bad.

"A fifth attempt compromised ENDPOINT1, a machine used by a support analyst. This delivery attempt was detected and contained by our Trust Operations team on April 3rd." So five times, five attempts, four were blocked, one got through. The next day it was discovered. They wrote: "Following an immediate internal investigation based on the telemetry data at hand, it was assessed that the incident had been contained."

Okay. So that's their summary. Then we receive an interesting narrative, some of which Lawrence posted, but the deeper details are interesting. So DigiCert said - so this is DigiCert, you know, writing it all up: "DigiCert received the initial third-party report related to this incident on April 5th. Additional third-party reports are identified in the timeline."

So, okay. Just to recap, before we go any further: The penetration occurred on April 2nd. DigiCert's Trust Operations team determined it had been contained the next day on April 3rd. And the first third-party report that is coming from some other outside source saying, hey, we've got some malicious code here that's signed by like a certificate of yours that's fresh. So the first third-party report of malicious code found in the wild, signed with a then-valid DigiCert EV code signing certificate, occurred the next day, the following day on April 4th. So that's going to, like, whoa, bring DigiCert to full attention.

They report: "DigiCert regularly receives certificate problem reports from community members and security researchers for Code Signing certificates, and proven key compromise cases are revoked pursuant to the Code Signing Baseline Requirements. Initial problem reports ultimately linked to this incident report fit within the normal pattern of such revocations."

Okay. So they revoke the certificate. Then 10 days go by. They write: "On April 14th, further investigation identified that ENDPOINT2" - a different machine - "a machine used by another analyst, was also compromised through the same delivery vector on April 4th." So this had a 10-day window. "CrowdStrike was not installed on that endpoint, meaning the compromise was not detected during the earlier April 3rd investigation. The machine was established" - meaning ENDPOINT2, this newest machine, they said - "was established" - meaning, you know, set up - "more than three years ago. Because our end-user machine logs are retained for three years, we cannot determine why CrowdStrike was not installed on this particular endpoint."

Okay. So at this point it doesn't really matter why. What matters is that, because CrowdStrike was not installed on that second infected ENDPOINT2 machine, its infection went undetected for 10 days. But in the interest of a full forensic after-the-fact, how-did-this-happen investigation, they would have liked to know exactly why that machine had apparently never been under the protection of CrowdStrike. Their records only go back three years, and that machine predates that logging cutoff. So today, they have no way of knowing what happened back when that machine was initially brought online, why it

didn't get CrowdStrike. And now, of course, given that they found one machine that was missing its protection for an unknown reason, the question becomes what other sensitive machines might also be missing their protection? You can imagine that they're going to go find out.

Their reporting continues, writing: "Our Trust Operations investigation found that the threat actor used the compromised analyst endpoint to access DigiCert's internal support portal. The threat actor used a limited function within the customer-support portal which allows authenticated DigiCert support analysts" - you know, the people that we talk to as DigiCert customers, I've done that on a number of occasions - "to access customer accounts from the customer's perspective to facilitate their support tasks." Makes sense. "This access is restricted and does not permit actions such as managing accounts, users, API keys, or submitting or managing orders. However, the threat actor was able to use this function to access" - probably meaning view - "initialization codes for orders that were approved but pending delivery for EV Code Signing certificate orders across a finite set of customer accounts."

They write: "Possession of an initialization code, combined with an approved order, is sufficient to obtain the resulting certificate. Since the threat actor was able to obtain these two pieces of information for a finite set of approved orders, they were able to obtain EV Code Signing certificates across a set of customer accounts and CAs."

Okay, now, so just to put this in context, if you're wondering about DigiCert's phrase "across a set of customer accounts and CAs," the notion of, like, why would it be more than one like them CA, the notion of differing CAs could seem strange, since we're only talking about DigiCert. But remember that when I was out shopping around for a new code signing certificate provider earlier this year, and finally settled upon IdenTrust, I discovered that a surprising number of the many apparent alternative certificate authorities all shared utterly identical prices, terms, and conditions with each other, and with DigiCert. It quickly became clear that DigiCert had been busily gobbling up much of the competition. So all of these alternative CAs had just become different storefronts for DigiCert.

And what they have just written confirms that these various fronts were all sharing DigiCert's common back end. I'm not criticizing DigiCert. It's smart business. But it does mean that we now have much reduced competition, and that's not usually best for consumers.

Okay. So now we get some statistics and numbers. They write: "During our investigation between April 14th and 17th, as DigiCert identified certificates potentially affected by the threat actor's actions, we revoked them. DigiCert revoked 60 certificates issued from the following CAs. And there are four of them: DigiCert Trusted G4 Code Signing, and they've got a bunch of different specs on that. And another of the same. Then one called GoGetSSL G4 Code Signing, so that's probably one of the other compromised sub CAs. And also something called Verokey High Assurance Secure Code EV." So those were the root CAs that had been used to sign those certs.

They wrote: "27 of the 60 revoked certificates were explicitly linked to the threat actor: 11 were identified in certificate problem reports provided to DigiCert by community members linking the certificates to malware, and 16 were identified during our own investigation. Our investigation included review of the threat actor's activity in the support system, as well as tracing delivery to IP addresses known to have been used by the threat actor." You know, so they took - they got information about the known problem certificates. Then they looked at the metadata surrounding the issuance of those certificates and then were able to use that to broaden their search and find any other certificates that the same threat actor had also managed to issue to itself.

And so they were able to say: "The IP addresses used by the bad actor to install certificates included" - and they provided in their report unredacted IPs. You know, 82.23.186.8 and so on. There's a bunch of them there. So those are the IPs that the bad guy used in order to compromise DigiCert.

They said: "In addition to the 27 fraudulently issued and revoked certificates identified above, 33 of the 60 total certificates were revoked during our own investigation as a precautionary measure. For these certificates, we could not explicitly confirm customer control. In addition, pending orders were canceled, closing access to the threat actor. All identified certificates were revoked within 24 hours of discovery, with the revocation date set to their date of issuance."

So note that we keep seeing this language within 24 hours of discovery. This is DigiCert explicitly asserting that it has carefully followed the well-established CA/Browser Forum guidelines for proper CA behavior. This is where, as previously we've seen and reported, the other disgraced Certificate Authorities fell well short. You know, those others were "under the rug sweepers," first hoping that no one would notice and catch them in their mistakes; and then, once they'd been exposed, they worked overtime to minimize and hide their failures. The truth is, no one expects anyone in this arena to be perfect. Perfection is not a requirement. Proper behavior and acknowledgment of a mistake, that's the requirement. So that's what DigiCert is busy doing here.

They wrap up their initial overview by writing: "The exploited certificates identified by the community member were found to have been issued by and to sign the Zhong Stealer malware family" and so forth. And basically the same stuff that Lawrence talked about. So the really interesting stuff comes next, it being how they take themselves to task over how this happened, and in detail the contributing factors that facilitated the attackers' success.

So what I'm about to share is the reason I gave today's podcast the title "DigiCert Doing It Right." This is written so objectively that it feels more like the work of outside auditors than DigiCert's own staff. It's just so difficult to fully disconnect one's own ego from, you know, truly self-indicting statements. But to DigiCert's credit, there was no sign of the typical rolling disclosure that we've seen so many times elsewhere. You know, Microsoft, for one, could certainly learn a thing or three from DigiCert. So we're going to talk next about their headline "Root Cause Analysis," Leo, after we take our final break.

Leo: Okey-dokey. Man. We live in a dangerous world. I've just got to say.

Steve: Yeah. We live in world where a huge amount of industry is being applied by the bad guys.

Leo: Yeah.

Steve: I mean, remember how...

Leo: They're working hard.

Steve: At the beginning of this podcast we had cute little viruses. And they used to infect people, and we'd go, oh, look at that.

Leo: Aww.

Steve: What does it do? Nothing. It just propagates. Why? Well, because it can. But...

Leo: It defaces your web page, that's all.

Steve: Everything changed when cryptocurrency allowed the bad guys to get paid.

Leo: Yup.

Steve: It turned it into a business model. So under Technical Background, they said: "Understanding this incident requires understanding how EV Code Signing certificates are issued on hardware tokens. The customer requests a Code Signing certificate from DigiCert. Following validation, DigiCert securely provides what they call an initialization code, we've heard that term throughout this, an initialization code to the customer. The customer installs, or already has installed, DigiCert's Hardware Certificate Installer software locally, meaning at their end. The customer inputs the initialization code into the installer, which generates key pairs on the hardware token and submits the public key to the CA. The CA generates the certificates against the approved order. The installer retrieves the resulting certificate and installs it on the token."

They said: "The process is described in a public knowledgebase," and they provide the link. "Possession of the initialization code, combined with an approved order, is functionally sufficient to generate and retrieve the corresponding certificate. The initialization code operates as a bearer credential for the approved order and is single use. This feature made it apparent which initialization codes had been used."

Okay. So I've done exactly this in prior years with DigiCert. When you think about it, the process of issuing a certificate, like creating and issuing, you know, creating and signing, that will be contained on a hardware token is a little trickier than you might just imagine at first. The need is for the private key-half of the public/private key-pair to never, ever, for any reason, ever (just to be clear) ever exist...

Leo: Ever. Never.

Steve: Yes. Thank you, Leo, outside the hardware. It's in the key, and it never leaves. That means that it must be generated by the dongle, inside the dongle, and that the dongle will never export its ultra-protected hardware key.

So web server public/private key pairs have no such requirement. So a web server just uses the underlying operating system's cryptographic system to synthesize a key pair, a public key pair for it. It holds onto the private key while the public key is placed into a CSR (Certificate Signing Request) which the CA signs and returns. But forcing the private key to never leave the hardware dongle very much complicates matters.

The point here is that DigiCert issues these "initialization codes" against a customer's account, sends them to the customer, who then uses them with DigiCert's own Hardware Certificate Installer app running on the client's machine with the hardware dongle plugged into it. The code, which is ingested by their app which they provide, validates their right to have a certificate by communicating on the back end with DigiCert's APIs

and servers. Then it triggers the key-pair generation on the hardware dongle. Then the hardware dongle does allow the public side, the public key to be sent out, which the app then uploads to DigiCert, and the installation of the signed certificate - so DigiCert then signs that public key, sends it back to the app, which then installs it back into the hardware. So a lot is going on that the user never sees.

From the user's perspective, it just kind of is magic. You enter your code, and you say go, baby. And a minute or two later it says, okay, you've got your key. You're all set to go. But the point is that the stateful nature of the operation creates some points of exploitability. There exists a window from the time that initialization code is issued to the time that it is actually applied, where the bad guys who are able to get it could install it in their own hardware rather than the customer installing it in their hardware. And these are big companies; right? Lenovo, for example, where, you know, they've got teams doing things. And they have got initialization codes that are just - have been issued, but haven't been used yet. And so the bad guys took advantage of that window.

So DigiCert states four what they called "contributing factors," things about the way their system is, and actually now was, that helped this to happen. Contributing Factor #1, they said: "Inconsistent or incomplete endpoint detection coverage." Well, we know that; right? They said: "Security tooling (CrowdStrike) was not uniformly configured by DigiCert across the user population exposed to the attack. The CrowdStrike prevention setting on ENDPOINT1 was below the intended organizational standard at the time of the initial compromise, allowing the malicious payload to execute before blocking engaged.

"The CrowdStrike sensor on ENDPOINT2 was not installed. As a result, no detection fired on the compromised machine. Logs for end-entity machines are retained for three years. Since this machine was set up more than three years ago, our security team could not determine why this particular machine did not have an installed CrowdStrike sensor." They also said: "The sensor not being installed was identified on April 14th during the expanded investigation triggered by the third-party report." You know, they thought they had it contained on the fourth. Ten days later it's like, oh, crap, something big and bad has happened.

They said: "The original investigation on April 4th did not include a check of EDR (endpoint detection) enrollment status for all exposed users. If the sensor had been installed on ENDPOINT2, the connection on ENDPOINT2 would likely have been detected and contained in the same timeframe as the other targeted machine that was ENDPOINT1. This created the window during which the threat actor was able to access the portal function" - we're about to talk about that in the next Contributing Factor - "and harvest initialization codes," which actually is the third Contributing Factor.

Okay. So the second Contributing Factor: "Insufficient privilege minimization in the support portal function." Again, taking full responsibility for how the bad guy was able to get up to as much as they were. Insufficient privilege. So minimization. So they wrote: "DigiCert's internal support portal includes a function that allows authenticated support analysts to proxy into specific customer accounts, to facilitate customer support." You know, like viewing it the way the customer does. I don't really understand what's going on. Can you show me that I'm supposed to be seeing? And so the support guy says, okay, let me get onto your account, and goes ah, I see what you mean.

So they wrote: "In this mode, certain functions are masked from the analyst. However, access to initialization codes for pending Code Signing certificate orders was not among the masked data elements, and they're saying it could have been, leaving those codes accessible to support an analyst operating in a proxied session."

They said: "The portal function had not been formally classified within DigiCert's privileged access management (PAM) framework. The definition of 'privileged access' was

primarily scoped to direct access to CA and did not encompass this indirect account management function that had a path to certificate issuance. As a result, the portal function was not subject to the PAM controls applicable to privileged users under the CABF Net Security, including formal threat modeling against misuse scenarios, least-privilege design review, and access recertification." In other words, they missed this. And they recognize that this did not have to be.

They said: "The portal function is a longstanding feature. On April 14th and 15th, following the discovery of the incident, we deployed a code change to mask initialization codes from proxied users on both our U.S. and EU platforms using either the UI or the API. The absence of this initialization code masking was identified during the investigation triggered by the third-party report on April 14th."

And finally: "Interaction with other factors: This factor" - you know, this second contributing factor - "defines the scope of the damage enabled by the Contributing Factor #1, which was the lack of endpoint coverage. Without the EDR gap, the dwell time would have been minimal, and the number of initialization codes accessible would have been limited. This factor also interacts with Contributing Factor #3 as the absence of masking is a direct consequence of the codes not being recognized as requiring credential-tier protection."

Which brings us to Contributing Factor #3: "Initialization codes not protected as bearer credentials." Meaning the initialization codes were not considered to have sufficient need for protection. They wrote: "The EV Code Signing certificate pickup workflow was designed with a threat model that assumed initialization codes would only be accessible to the validated subscriber, delivered via a secure channel, and entered by the subscriber into their local Hardware Certificate Installer. The threat model did not account for the scenario in which initialization codes stored within DigiCert's internal support portal could be viewed by a compromised DigiCert analyst account operating through the portal function." Again, whoopsie. "Therefore, initialization codes were classified as intermediate workflow data rather than bearer credentials," which would have elevated their need for protection to a higher level.

"This initialization code workflow," they wrote, "was designed at the time the EV Code Signing token issuance process was developed. The issue was identified during the investigation triggered by the third-party report on April 14th. It was not identified through any previous design review prior to this incident." So this just kept getting missed because, again, you can look at things, but sometimes you just don't see them. And their point is that this mistaken definition allowed it never to be properly classified, and then it did not automatically fall within the proper security context and constraint.

So everything we're seeing here is the work, and it should sound like it, of true forensic security professionals patiently working to understand, step by step, not only what happened, but why a supposedly carefully designed security system was even designed from a set of theoretical premises of what we want, even so how that allowed this to happen. So the result is guidance about what can be changed to prevent successful exploitation at each stage.

And this brings us to the fourth and final Contributing Factor, which explains how the bad guys managed to infect DigiCert's two analysts in the first place. They wrote: "Overly permissive file transfer capability in customer-facing support channels." They said: "DigiCert's customer support chat channel and Salesforce case attachment workflow permitted delivery of inbound file attachments from external parties, including the general public, to CA support staff with insufficient restrictions on file type, automated sandboxing, and content inspection. This created a direct delivery path for malicious executable content to personnel having privileged access. The support chat channel had

not been adequately evaluated as a potential attack surface for malware delivery against certificate authority staff.

"The support chat channel and Salesforce case attachment integration were operational prior to this incident. The delivery vector was confirmed on April 4th and 5th, at which point additional malicious ZIPs were identified across other Salesforce cases and removed." Which is significant; right? There were other infections that hadn't had a chance to, like, take hold.

"Corrective controls (file type restrictions, sandboxing evaluation) are in progress as described in Action Items. The number of channels by which customers can reach support staff has grown. The file controls on the chat channel were believed to be sufficient prior to this incident. This factor is the initial attack vector that enabled all subsequent factors to come into play."

And this brings us to the "Lessons Learned" conclusion with what went well and what did not go well, and where they got lucky. So they explained under what did go well: "Rapid initial containment on endpoint machines where EDR was working as intended. For these machines, DigiCert's Trust Operations team completed containment, process termination, registry cleanup, and artifact deletion within hours of detection on April 3rd."

Also what went well: "Proactive identification of the full delivery chain. They were able forensically to catch that. The investigation identified the Salesforce case attachment auto-conversion mechanism as the delivery path and located additional malicious ZIP files across other cases before they could be opened, preventing further compromises from the same campaign." So when they talked about it earlier, they did not throw Salesforce under the bus, but they talked about how additional points of entry had been added. Apparently, it was the incorporation of the Salesforce services that allowed this stuff to get in that way. And it sort of snuck by.

They said also what went well: "Same-day remediation of contributing vulnerabilities during incident response. Critical fixes were implemented without deferral: CrowdStrike prevention settings corrected on April 4th; Okta FastPass disabled and multifactor authentication tightened on April 14th; initialization code masking deployed across U.S. and EU environments on the 14th and 15th." And "Confident attribution of the second compromise through forensic analysis." They did figure out exactly what happened after, you know, with that ENDPOINT2 machine. They said: "Linking across compromised machines' activity logs to the same threat actor required analysis across identity events, endpoint telemetry, and support workflow artifacts."

So if that all went well, what did not? "File-type controls turned out to be insufficient on customer-facing support channels. Inconsistent and incomplete EDR coverage enabled a blind spot that was unknown prior to the incident and that directly enabled the attacker's dwell time," you know, giving him 10 days. "Initialization codes were not protected as bearer credentials," as they should have been. "The portal function exposed initialization codes that are functionally equivalent to the certificates they enable because they were classified as intermediate workflow data rather than credential material requiring masking and credential-tier protections."

Also what did not go well: "Device-bound authentication (Okta FastPass) was used as an MFA (multifactor authentication) bypass. FastPass allowed a threat actor operating on a compromised device to inherit the device's authenticated session and satisfy multifactor authentication requirements without a genuine second factor, enabling access to the portal initialization codes after the initial endpoint compromise." And finally: "Following ENDPOINT1 containment on April 3rd, the investigation concluded" - obviously incorrectly - "that compromise attempts had been neutralized without validating all endpoints

exposed through the same delivery vector." So somebody a little too quickly said, okay, we're done here.

"In retrospect, of course, I imagine that they now wish that after that first attack, which was thwarted by CrowdStrike's endpoint defense after a brief window, that they had checked for other instances of that malicious ZIP file across the rest of their network. We now know that they did that later and did find a handful of other instances of it." So it's sort of unclear why that did not happen at the time. And I imagine somebody's asking somebody.

So they finally share two points, in conclusion, where they felt they got lucky. They said: "A community member involved in security research reported the evolving pattern of misused certificates and engaged in dialogue with our support team. Without that report, the undetected compromise of ENDPOINT2 and the associated mis-issuance might have remained undiscovered for a longer period." Nice of them to admit that.

"Also they got lucky where our investigation indicates that the threat actor's activity was focused on gaining access to Code Signing certificates. A differently motivated threat actor might have attempted to use the compromised account for broader actions. Several of our action items are designed to address this risk."

And then they conclude with a final three points: "This incident demonstrates that internal support tooling with indirect paths to certificate issuance must be subject to the same security scrutiny as their certificate authority issue infrastructure. Tools that were designed for legitimate operational purposes can become high-value attack targets."

Second: "The incident also illustrates the importance of defining 'privileged access' broadly enough to encompass any system or function with a path to certificate issuance, not just those with direct access to HSMs (hardware security modules) or signing infrastructure."

And finally: "The dwell time underscores the importance of comprehensive post-incident investigation scope" - meaning don't quit your investigation prematurely - "and continuous EDR coverage monitoring. Like make sure all your endpoints are actually being monitored. A single missed endpoint can negate the value of rapid containment elsewhere."

So, you know, they go on to actually enumerate 21 individual Action Items, many of which they already articulated or implied, so I'm not going to take us, thankfully, through each of those. Suffice to say that there's little doubt that DigiCert was extremely unhappy and embarrassed by this event. Right? Everything any Certificate Authority is about, and a huge amount of security-focused design and third-party contractor support from the likes of CrowdStrike and several others, all of this was intended to prevent anything like this from ever happening. But it did anyway.

But at no point did we see what we've previously observed from so many other Certificate Authorities who have been struck by similar abuse, or even by Microsoft, who, you know, has nothing to lose. No one's going to leave Windows. Almost invariably they first hoped, you know, the other guys, these really denial certificate authorities first hoped that nobody would notice. Then, when someone did, they would downplay the severity of the incident, hoping that that would be it. Then when additional evidence of further exploit came to light and surfaced, oh, they'd apologize with some lame excuse about having intended to mention that, too. Uh-huh. Yeah, right.

What we have from DigiCert, the industry's largest commercial certificate authority, second only to Let's Encrypt, which, you know, is free, is full public disclosure and responsibility taking. The result has been a deep analysis followed by true action to

prevent - like at multiple levels and stages, not just like closing the front door, but all the hallways in between, you know, to prevent anything like this from transpiring again. And as usual, I think that's more to recommend them than not. If their code signing certificates were not so unreasonably expensive, I would not have invested so much time earlier this year preparing to jump ship and find another supplier. I'm glad I'll be moving to IdenTrust, but I'm only doing so because I object on principle to unconscionable costs for certificates, not the quality of their work.

But as for Microsoft: Given what we now know, it is unbelievably difficult to understand, to explain, to explain away, to excuse how Microsoft could have possibly fumbled their end of this so thoroughly. There would presumably have been some dialogue between Microsoft and DigiCert, where DigiCert provided Microsoft with the thumbprints or serial numbers of the 60 certificates they had revoked and blacklisted so that Microsoft could do the same with Windows Defender. As we know, revocation is an imperfect answer with certificate management, but it's all we've got.

So Microsoft would have definitely wanted to add those certificates, those 60 certificates, to Windows Defender's existing code signing deny list, so that nothing signing them would have been allowed to run on Windows. But that just entails checking thumbprints against Defender's deny list. How Microsoft could possibly have fumbled this into the removal of DigiCert's root certificates, for all certificates ever issued in the world ever, is just impossible to understand. It feels very much as though someone in control of that important process doesn't know what they're doing, which is horrifying. I hope AI didn't do it.

Anyway, fortunately that, too, was fixable, and it was quickly fixed. So we go on, Leo, to the next adventure.

Leo: I love it that Microsoft kind of - the way they phrase this sounds like they're blaming Defender. They say: "In response to reports of compromised certificates, Microsoft Defender was a bit overzealous." Defender was overzealous? Well, it certainly acted poorly, but I think somebody told it to do so. So that's interesting. So they were initially compromised through Customer Support Chat, files uploaded through Chat.

Steve: Well, maybe Chat, or it sounds like maybe they've added some Salesforce Software as a Service thing.

Leo: Right.

Steve: Because there was a mention of Salesforce and the bad...

Leo: Well, that may be the Chat.

Steve: Exactly. It might be Salesforce Chat. And so that it did get in that way. So somebody said to the support guy, hey, I don't understand what I'm seeing here. Let me send you a screenshot.

Leo: Yeah, here's - yeah. Or here's a ZIP file of something, yeah. And then the CSR, who have escalated privileges, which is a problem. Unzipped it. It attacked. They

responded very quickly. But this shows - this is the issue with certificates. Unlike say a ransomware attack, although that can happen pretty quickly. But if I have 10 minutes with DigiCert, and I get their root certificates, I'm golden. I don't need more than that, more time than that.

Steve: Yeah.

Leo: Wow.

Steve: And, I mean, it is, it's one of the things that I did think was that we clearly have a system, we have so many security firms that are looking for malware. When they talked about, you know, an industry partner or a third party, you know, it was some security firm who they didn't identify, that called up and said, hey, we're seeing some malware that was signed an hour ago by Lenovo. And it's your cert. So what do you think about that? And then so, I mean, so we have, you know, there's a lot of closed loops here.

Leo: Oh, yeah.

Steve: It's good that they're closed and that they're looping.

Leo: There's really a huge kind of web of threat analysis...

Steve: It's a big ecosystem.

Leo: It's a really big ecosystem. And they know that they have to work hand in hand. They have to work together. So in some ways, you know, that's been the response to all of these attacks is a much improved, I think early warning system.

Steve: Which, you know, here I am, trying to publish my little software, and being blackballed by Windows Defender. You'd think that with this system working as well as it is, they could wait until actually seeing malware and then call DigiCert to report me. But no. Defender says I don't know about this.

Leo: They remove the entire brain, instead of just the tumor. They just said, well, just take the whole brain out. You don't need that; do you?

Steve: Yeah, we decided we're not going to trust anything DigiCert has ever signed. I mean, it's crazy.

Leo: Unbelievable. Yeah, that's a very bad response. It's almost, you know, it feels like a panic response, like they were so freaked out that they overreacted. And it wasn't Defender doing this, it was some human. Or maybe some AI, I don't know. I don't think - it sounds like...

Steve: It sounds like somebody really - to go in and remove a root, you had to know you were removing - at no point did the response to DigiCert require removing anything.

Leo: Right.

Steve: It meant checking thumbprints. Does this thumbprint match? That's all. So how it got extended, I mean, again, I do actually wonder, I had the thought before I was recording this with you, Leo, could this have been AI that hallucinated a root cert removal?

Leo: Maybe.

Steve: Is that what we're now - is that what we're now going to be dealing with?

Leo: Boy, I hope that's not the case because that's terrible.

Steve: If so, get your seatbelt.

Leo: Yeah.

Steve: And buckle up, baby.

Leo: Yeah. Well, what a good story, though. And I'm glad DigiCert did the right thing.

Steve: Oh. Again, I just, you know, a hat tip to those guys, the fact that they ruthlessly not only, you know, investigated, but then self-reported. I mean, no one could ever take issue with the way they're behaving. I'm sure that was their goal, too; right? I mean, they don't want anyone to have any doubts about them, and who would after this? I would trust them more now.

Leo: Right, right, right.

Copyright (c) 2014 by Steve Gibson and Leo Laporte. SOME RIGHTS RESERVED

This work is licensed for the good of the Internet Community under the Creative Commons License v2.5. See the following Web page for details:
<http://creativecommons.org/licenses/by-nc-sa/2.5/>