The Polyfill.io Attack

Description: What was Entrust's response to Google's decision to refuse trust of any of their TLS certificates signed after October 2024? How have the other CAs responded to this new opportunity? What's a Passkey Redaction Attack, and how worried should you be? And speaking of Passkeys, why not just have each website hold as many as we need? Wouldn't adding port knocking in front of the serious OpenSSH flaw we discussed last week prevent this problem? And if so, what's the larger lesson to be learned? And what about blocking an IP after some number of failed attempts? And finally, once again the Internet dodged a potentially devastating bullet. What happened, and what significant lesson should we take away?

High quality (64 kbps) mp3 audio file URL: http://media.GRC.com/sn/SN-982.mp3
Quarter size (16 kbps) mp3 audio file URL: http://media.GRC.com/sn/sn-982-lq.mp3

SHOW TEASE: It's time for Security Now!. Steve Gibson is here. Coming up, a response from Entrust to getting yanked from Google Chrome and other browsers. We'll also talk about how you can protect yourself using port knocking from the OpenSSH flaw we talked about last week. The biggest story, though, is the big bullet the Internet dodged, it could have been a bad one, the story of Polyfill.io. All that and more coming up next with Steve on Security Now!.

Leo Laporte: This is Security Now! with Steve Gibson, Episode 982, recorded Tuesday, July 9th, 2024: The Polyfill.io Attack. It's time for, oh, time to record a show.

Steve Gibson: Oh, yes, the big red record button, please.

Leo: We now have copies of the show on every streaming service, so the chance of losing a show is very low. It's time for Security Now!, the show where we cover the latest security, privacy news; tell you a little bit about how computers work; sometimes talk about sci-fi with this guy right here, Mr. Steve Gibson. What do you see? What do you know? It's time for the Steve Gibson Show. How do you like it? We're singing that. No?

Steve: Maybe we should wait till year 20 before we deploy the corny jingle.

Leo: We'll never get to 999 if you keep that up, Leo.

Steve: Okay, well, this is the week I've been waiting for because I have a photo that just had to be a Picture of the Week. It had to be given the label Windows Patch Tuesday. And so here we are, second Tuesday of July, Patch Tuesday. Don't know what the patches are going to be, but it doesn't really matter because this photo says it all.

Today's podcast is titled "The Polyfill.io Attack." And oh, baby. The Internet, we the Internet, dodged another potentially devastating bullet. A really interesting story, though, about the back story and history of this. But first we're going to look at, not surprisingly, Entrust responded to Google's decision, which was our topic last week, to refuse the trust of any of their newer TLS certificates. We're going to look at their response, and also at how the other CAs have responded to what they perceive as a new opportunity. And I was thinking, Leo, about how you mentioned last week that somebody in the forum, I guess in Discord or wherever you have the chat dialogue, because I commented that Entrust was not a major, you know, not one of the top-tier certificate providers. I don't remember if it was 1% or .1%.

Leo: It was pretty low, but it was ninth on the list, so it wasn't...

Steve: Well, yes. And I was going to say that, given the total number of certificates, even 1% is...

Leo: Right, right, it's a lot.

Steve: That's a lot of certificates that, you know, you'd like to...

Leo: Yeah, to screw up.

Steve: Yeah, well, or the other CAs would love to take off of Entrust's hands. You don't need all those certificates. Let us handle those for you because you see them unable to carry them all. Also, we're going to answer the question, what's a Passkey Redaction Attack, and how worried should we be? And speaking of Passkeys, why not just have a website, one of our listeners ask, hold as many of them as we need? And also, regarding the OpenSSH attack, wouldn't adding port knocking in front of that serious OpenSSH flaw which we discussed prevent this problem?

Leo: Hmm.

Steve: And, if so, what's the larger lesson there to be learned? And what about blocking an IP after some number of failed attempts? Or nailed, I guess. Number of failed, that would be nailed attempts. What about that? And as I said, we're going to then wrap up with the Internet once again dodging a potentially devastating bullet. But what's really interesting is how we just walked right into this thing. I mean, it's just, like, so obvious. Other people saw it coming. But nah. Anyway, the good news is lessons to be learned, and I think some great takeaways from this week's podcast, "The Polyfill.io Attack."

Leo: I like whatever that is. Polyfill just sounds good. Sounds like something to hold your dentures in place. No, that's another...

Steve: Actually, I think there is a product called...

Leo: Polident.

Steve: Polident, yes.

Leo: And I have written in the past, I have coded flood fills. That's kind of a fun little coding routine. But a polyfill? I don't know what that is. We'll find out. I am prepared to show you the consequences of many Patch Tuesdays.

Steve: One could argue too many Patch Tuesdays.

Leo: One too many.

Steve: Oh, this is just so good.

Leo: Do you want to describe it? Do you need to describe it?

Steve: What we appear to have, given the somewhat blurry background, is like an oil storage tank of some sort. Sort of looks like that, a big barrel painted red, got a big valve on it. But then coming from it in the background, right up to crystal clarity in the foreground, is a pipe coming from the bottom of the output of this barrel, which then makes a right-hand turn, and it heads off the screen. But not before we are treated to what you can only describe as something must have sprung a leak, and the people were desperate to not lose any of the precious fluid that was contained in this tank. This is just - this is a patched pipe from hell. I mean, it's got, like, shims and cardboard and black tape wrapped, and then metal screw-tightenable strangulators, I mean, it's just wonderful.

Leo: It ain't going anywhere, that's for sure.

Steve: The moment I saw it, I thought, wow, this is our Patch Tuesday Picture of the Week.

Leo: Oh, boy. Yup.

Steve: So for those who are listening, if you have signed up to GRC's obviously free email system...

Leo: Well, not obviously. It's good that you do make that for free.

Steve: It is free, of course.

Leo: Yes, yes.

Steve: 5,782 of our listeners, so we're approaching 6,000, 5,782, received the email from me a couple hours ago containing a little thumbnail size of this wonderful picture, which if you click on it, will reveal it in all its full-size glory. So, wow, this is a great Patch Tuesday picture.

Okay. And again, a reminder that, if you would like to get the summary of the show, a link to the show notes, and also the Picture of the Week in your inbox every Tuesday morning, as soon as I have them, the system's working well.

Leo: You have a, you know, that number is actually pretty good. That's almost 10% of your audience is subscribed. So that's pretty good. Good job. But that means there's still nine out of 10 can subscribe.

Steve: Well, and, you know, realistically, a lot of...

Leo: It's what I'd expect.

Steve: Yes, I'm sure that a lot of people are like, hey, I listen to it. Why do I need text? You know, I've already got it buzzing into my ears. So I understand.

Okay. So Entrust responds. We get a letter from el presidente, followed by an interesting FAQ. So Todd Wilkinson, the President and CEO of Entrust, you can imagine this all got his attention. The day after Google's announcement, and of course we were right on top of it the day of, he wrote an open letter to the industry essentially, titled "Thoughts on the Google Chrome Announcement and Our Commitment to the Public TLS Certificate Business." And again, by Todd Wilkinson, President and CEO of Entrust.

He wrote: "Last week Google announced that they would no longer include Entrust root CA certificates in the Chrome Root Program." Not technically accurate, but then, you know, Todd is the president, so we're not expecting complete technical accuracy from a president and CEO. But he's, you know, he's got the gist. As we know, the roots are still staying in Chrome. It's just that the signature date of anything signed by those roots is now being examined. And if that date falls after this coming Halloween, which is to say the end of October, then they will not be trusted. But anyway, Todd's got the idea. He says: "We are disappointed by this decision" - one can imagine - "and want to share how we intend to move forward."

He says: "We understand what led us here. We are committed to improvement. And Entrust continues to have operational capabilities to serve our customers' public and private digital certificate needs. These capabilities extend beyond the issuing roots in question. Our recent mis-issuance incidents arose out of a misinterpretation we made of CA/Browser Forum compliance requirements." And let me just stop right there to note that, and when they found out that they had been misissuing certificates, they did not stop doing so. They instead started arguing with the CA/Browser Forum while continuing to issue misissued certificates and refusing to revoke any because they didn't want to upset their customers.

Okay. Anyway. He says: "In our attempt to resolve this issue, our changes created additional non-security related misissuances."

Leo: Oh, boy.

Steve: Wow.

Leo: We started digging a hole, and then we dug it...

Steve: Get a little tangled up there, Todd.

Leo: ...a little deeper.

Steve: Yeah, throw that shovel away. "In our attempt to provide additional flexibility to our customers" - okay, we don't know what that is - "we provided extensions and delays in revocations that were not supported by the CA/Browser Forum Requirements." In other words, we weren't supposed to do that because, you know, there's no latitude here; right? You've got to follow the rules, which you voted on. Anyway, he says: "...which mandate five-day revocation for all certificate mis-issuances." So, you know, he's covering part of this, but he's conveniently failing to mention that we didn't stop misissuing certificates, and that's also "bad dog."

So, he says: "This created an environment in which the community scrutinized past Entrust incidents." Whoops, we looked further back. How did that go? He says: "This identified past Entrust commitments which, if fully implemented, could have helped to prevent these incidents."

Leo: Oh, boy.

Steve: But of course we didn't fix those, either, back then. So, whoops. Our bad. He says: "We agree that there are opportunities for us to improve." In other words, they agree with everybody else. "And we've completed a thorough assessment of our CA operation in the past few months. As a result of this assessment, we made changes in our organization" - I'm tempted to insert here, which no one has seen yet, no - "in our organization, processes, and policies. For example, we've moved the CA product compliance team into our global compliance and operations teams to fully leverage the more robust capabilities of this larger organization."

Leo: Well, thank goodness.

Steve: So apparently the people that were in the sidecar and...

Leo: Geez, Louise.

Steve: Wow.

Leo: We moved them. We moved them over.

Steve: Yeah, they weren't doing it right. So now we've moved the people that weren't doing it right in with the people who hopefully are.

Leo: Yeah, right.

Steve: And maybe they won't contaminate the rest of the group. He said: "We've instituted a cross-functional change control board" - because that's what you want in your change control board, Leo, is some cross-functionality.

Leo: Hereinafter, deck chairs will be rearranged by a committee of people in charge.

Steve: And bolted down with a seatbelt. So they've got a - they've instituted a cross-functional change control board, whatever that is - "and a technical change review board to catch similar issues in the future." Boy, this all sounds great. "We are accelerating R&D for TLS certificate compliance and automation-related work while also improving the tracking of our public commitments" - oh, they're going to be tracking their public commitments, that's good, so they don't wander off course - "and revising our public incident response practices to ensure such issues do not occur again."

Well, talk about showing up after the party's over. He said: "I want to assure you that we are committed to continuing to serve as a public CA" - even though we'll have no customers - "and that we will complete open issues and promised improvements in a timely manner. We're working with Chrome and the other browser root programs to address the raised concerns while also providing continuity for customers" - yeah, continuity, they had that because they were never getting any certificates revoked - "while we execute these changes. We have the expertise to do this, as demonstrated by our ability to deliver our many products and solutions designed to meet demanding global compliance requirements." Wait. It was the demanding global compliance requirements that they were not meeting, which is what they've demonstrated over the years. But, okay.

Finally, he says: "Entrust has been" - and this is true - "a publicly trusted CA for over two decades and has contributed to stronger Web PKI capabilities globally." Again, that's true. "We continue to have operational capabilities to serve customers' certificate needs today and" - well, he says we'll do so in the future. That remains to be seen because they'll have no customers. But, he said: "We respectfully ask for your patience as we work to ensure that you have no disruptions to the service you have come to expect from Entrust."

Okay. Well, so he said what he could. Right? I mean, do you need, at this point, is this where you need the President and CEO to say something? So he did. Of course, he failed to explain that the real concern was their history of like exactly this; right? Now, this is not the first time. This is like the 12th time the industry has heard this, a history of repeatedly broken promises to repair what's broken, and then excuses after the fact that, you know, oh, like why it was that they weren't - these changes weren't made and attempts to justify their actions. So, okay. And of course they keep putting their customers' needs in front of their duties as a CA, which was the beautiful conclusion that one of the techies came to last week.

So anyway, I found also their four-question FAQ regarding this incident to be interesting, especially number three. But here's all four. So this is what they asked themselves and then answered. Question: "What does it mean when Google says you've been 'distrusted'?" Ouch. They say: "This means that public TLS certificates issued from nine

Entrust roots after October 31st, 2024, will no longer be trusted by Google Chrome by default. TLS certificates issued by those Entrust roots through October 31, 2024, will still be trusted by default through their expiration date." Got that one. Nailed it. Okay.

Question number two: "Other CAs have said that after November 1st your TLS certificates won't be valid in the Google Chrome browser. Is that true?" "No, that's not true. All Entrust TLS certificates issued through October 31st, 2024," like we just told you, "will be trusted by default by Google through their expiration date. After October 31st, we will have the operational capabilities to serve customers' certificate needs." But, you know, I don't think they're saying "but not issue any new ones." Oh, no. "With alternative or even partner roots if necessary. Our goal is to ensure that customers have no disruptions and continue to receive global support, certificate lifestyle management" - oh, sorry, "lifecycle management." Be nice to have some lifestyle management - "and expert services to meet their digital certificate needs."

Question three: "Does Google's decision mean Entrust is out of the digital certificate business?" Oh, by no means no. "No. It means the public TLS certificates issued from nine Entrust roots after October 31st, 2024, will no longer be trusted in Chrome's browsers by default. Google's decision has no impact on our private certificate offerings."

Leo: Hmmm? Oh.

Steve: Well, private certificate, not public.

Leo: Private, right, right.

Steve: "Including our PKI, PKI as a Service, and managed PKI - nor our code signing, digital signing, and email, S/MIME and VMC offerings." Now, these are the things that I was referring to last week as the second order casualties of being forced away from the public certificate business. It's just like, well, you know, you don't seem able to make simple web certificates. Should we trust you to do all this other stuff? That's, you know, in question.

They finished answer three, saying: "We have the operational capabilities to serve customers' certificate needs now and in the future." And here they're repeating from question two: "With alternative or partner roots if necessary. Our goal is to ensure that customers have no disruptions and continue to receive global support, certificate lifestyle" - lifecycle - "management, and expert services to meet their digital certificate needs."

And, finally, fourth question: "Does this impact other Entrust solutions?" Answer: "This announcement has no impact on the broad range of Entrust identity-centric solutions." Of course, other than whether anyone is going to want them.

Okay. So as I said, I thought their most interesting response was three, noting that even after Google begins to conditionally mistrust certificates signed by any of their root certs, nothing prevents them - and this is true - from riding on someone else's coattails. Nothing, that is, other than Google releasing another update to Chrome which then also blocks that.

Leo: Right.

Steve: Remember that Entrust originally bootstrapped themselves into the commercial SSL/TLS browser certificate business by having Thawte sign their intermediate certificate, which is how they were then able to sell end certificates. And that allowed them to ride the coattails of the trust that Thawte had earned. So this leaves two questions: Would another CA think it was worth their while to sign an intermediate certificate for Entrust, knowing that their own reputation would then become entangled with Entrust's? And if this were done, what would Google do? Would they move to also block Entrust's intermediate certificates?

And actually, they could probably do that preemptively in Chrome. That is, they could look at intermediate certificates, see if the signer is Entrust, and essentially make the block a little stronger than just blocking certificates signed by the roots directly. You know, you could do a second order also. So we're going to see what Halloween brings this year. And I imagine behind the scenes Entrust is probably begging to be allowed to use these intervening four months - they've got July, August, September, and October - to fix and demonstrate fixes and not have that update to Chrome on November 1st make this happen.

Leo: I wonder how much they care. I mean, they really are in a lot of different - they have fingers in many different pies. I don't know how big a part of their business this is. Todd came to the company from Datacard, which acquired Entrust. And Datacard's business was printing credit cards, and the mag strips on the back. First credit card imprinting, like the raised lettering on it, and then the mag strips on the back. So I don't think he has deep roots in the CA community. Clearly, you know, it's just a management issue at this point. And maybe they don't even care about that business that much. It's kind of amazing.

Steve: We'll see; you know?

Leo: Yeah.

Steve: Somebody wrote a nice letter for him.

Leo: Yeah.

Steve: That he put his name on.

Leo: It's a big company. They have 2,500 employees. They're a giant company.

Steve: Yeah. They've been around since, you know, the...

Leo: Fifty years, yeah.

Steve: Yes. Yeah. And as we noted, they were acquired by private equity.

Leo: Right.

Steve: So don't know that that had any effect, but it happened. Okay. Now, no one will be surprised to learn that competing certificate authorities were not slow to report themselves on the news of Entrust's fall from grace and to offer to provide their services in replacement. Sadly, not all CAs did this in a conscientious fashion. For example, the CA Sectigo misstated the future.

Now, remember that Sectigo is not one of this podcast's favorite CAs. In fact, they would likely be the last CA I would ever choose. Long-time listeners may recall from whose loins Sectigo sprung. That would be none other than Comodo. Essentially, Comodo so badly damaged their own reputation due to a series of famous major certificate authority screw-ups that they decided to rebrand themselves in the apparent hope that no one would notice. Remember that first spyware that I found that was the - I'm trying to remember whether it was - I know that they were Aureate.

Leo: Oh, yeah. Aureate.

Steve: But they renamed themselves Radiate or something. You know, again, they got so, their reputation was so hit by the discovery that they'd snuck into everyone's PCs that they said, oh, I think maybe it's time to change our name.

Leo: Yeah.

Steve: That's right. So, okay. In any event, Comodo/Sectigo gave their rapacious posting the title: "Google to distrust Entrust SSL/TLS certificates: What this means for the industry." And then they tried to tell us. They wrote: "In a significant move to enhance digital certificate security, Google has announced its decision to distrust all public SSL certificates issued by Entrust, effective after October 31st, 2024." Okay, they got that part right.

"This announcement has sent not just ripples, but waves through the industry, particularly among Entrust customers who now face the urgent task of transitioning to new Certificate Authorities." Okay, except there's nothing whatsoever urgent about anything here. You know, as we are all very well aware, all currently issued certificates remain valid through their current lifetime, and everyone has July, August, September, and October to chose another CA when it comes time - I mean, it could be a year from this coming October - when it comes time to replace any Entrust SSL/TLS browser certificate. So not so urgent there, Sectigo.

Then they said, under the heading "The catalyst for distrust,"

"Google's decision is rooted in a series of compliance failures by Entrust." Uh-huh, and look who's talking. "Over the past several months, Entrust has experienced significant issues, including extremely delayed revocations and multiple lapses in meeting established security standards. Google's Security Blog noted: 'Over the past six years, we've observed a pattern of compliance failures, unmet improvement commitments, and the absence of tangible, measurable progress in response to publicly disclosed incident reports.'" Yeah. Rub their face in it. "This lack of progress and ongoing issues justified the revocation of trust in Entrust's public roots.

"To be a trusted browser, a CA must comply with specific requirements defined by the CA/Browser Forum." And I guess they would be an authority on that. "Transparency is crucial, as CAs are expected to work in good faith with browsers to fix and prevent

issues. Recent root program audits indicated a lack of confidence in Entrust's TLS certificate issuance practices, so this news wasn't completely unexpected to the industry, and prompted Google's decision to distrust Entrust certificates in the Chrome browser.

"Implications for business: For businesses using Entrust certificates, this development necessitates immediate action." Okay. No, it deliberately doesn't, but okay. "Any website using an Entrust certificate issued starting November 1st will be treated as an unsecured site on Google Chrome, and likely other major browsers will follow suit. Companies must source a new certificate authority before the deadline" - again, no - "to avoid their websites being flagged as untrusted." Again, just, you know, no companies must source a new certificate authority before their present Entrust certificate dies a natural death by having reached its expiration date.

Then they finally - they finish with "Choosing a reputable Certificate Authority." They said: "Considering Entrust's failings" - and of course our renaming - "businesses must reassess their relationships with CAs. A reputable CA should demonstrate robust compliance with industry standards, transparent operations, and a proven track record of security and reliability. Companies like Sectigo, which offers comprehensive certificate lifecycle management solutions, present viable alternatives."

Now, okay. As I said, these guys would not be my first choice. But we know who my first choice would be. That would be none other than DigiCert. So I went over to see what DigiCert had to say about this. Again, this end of Entrust trust does represent a significant marketing opportunity for every other CA in good standing. Now, rather than going point-by-point through DigiCert's similar marketing piece, I'll just say that they did the honorable thing, as I was hoping they would.

Under their topic "What Does That Mean for Entrust Customers?" they had three bullet points. First: "Public TLS certificates issued off of Entrust roots whose earliest SCT" - that's the signed certificate timestamp - "is after October 31st, 2024 will no longer be valid on Google Chrome." 100% correct. Second: "Those Entrust certificates will be treated as an unsecured site." Right again. And third: "Any TLS certificate with a Signed Certificate Timestamp dated before October 31st, 2024 will be valid for its term."

Okay. Now, with the slight exception that they meant before November 1st, rather than before October 31st, DigiCert got it exactly right, "valid for its term." And then they finished with, of course, "We're Here to Help. We understand this incident poses significant risk of business disruption to a large number of organizations. As the world leader in globally trusted PKI and TLS/SSL solutions, we're committed to making our services and solutions available to help you maintain critical operations and ensure uninterrupted business continuity during the transition from Entrust and beyond."

So, as it happens, GRC's DigiCert certificate expires at the end of this month. And thanks to their excellent service, I already have its replacement standing by, which I'll be switching all of GRC's servers over to shortly. So anyway, I guess we will see in the fullness of time what transpires. And I guess I'm most curious of all, out of all of this, to see whether the decision is reversible, whether, I mean, and I don't know how it would be, really, because all they can do is make more promises.

But their promises are only good when measured against practice. And that takes time. And they've been given time, time and time again, and have not come into compliance. So in order to have no disruption, Google would have to become convinced that this time they really mean it when they sounded like they really meant it all the other times, too, and nothing changed. So, you know, are these one-way decisions that never change? And I guess this is the point. Once they're out of the business, they don't have the capability of proving that they should be in the business because they're out of the business. So, interesting. Leo, we're 30 minutes in. Let's take a break.

Leo: Let's do it.

Steve: And then I'm going to talk about something new: Passkey Redaction Attacks. You don't want your Passkey to be redacted, that's for sure.

Leo: I don't even know what that means. That means cover it over with a black magic marker? I don't...

Steve: An interesting guy coined the term.

Leo: Okay.

Steve: And he likes to refer to himself in the third person, sort of like Trump.

Leo: Oh, hmm.

Steve: So, yeah.

Leo: All right, Steve. Let's hear about this redaction thing.

Steve: Okay, so, yeah. A number of our listeners picked up on a weird story that was featured on the Dark Reading site. Dark Reading is a good site, but maybe they were starved for news last week. The story's headline was "Passkey Redaction Attacks Subvert GitHub and Microsoft's Authentication." Okay, now, naturally that raised my eyebrows, too. The tagline beneath the headline was "Adversary-in-the-middle attacks can strip out the Passkey option from login pages that users see, leaving targets with only authentication choices that force them to give up their credentials."

Okay, now, so first I need to pause here for an old fart "get off my land, get off my lawn" comment.

Leo: Land is better. Get off my land, I like it.

Steve: Get off my land. I've got my shotgun ready. It's what they're calling an Adversary-in-the-Middle attack. Okay. So it appears that the politically correct "PC" police have decided that the use of the traditional and time-honored term "Man in the Middle" is no longer proper, as if "Man in the Middle" was actually a reference to some man somewhere, and that as a consequence all men should now be taking offense at this slur to our gender. Well, I am a man, and I'm not the least bit offended because I, and I'm sure everyone else, have always understood that the use of the term "man" in man-in-the-middle was always meant as an abstraction. So if any of our listeners were also rolling their eyes at the use of this new replacement term "Adversary in the Middle," anyway, at least know that you are not alone.

Leo: Well, you're not going to like this, but I understand the Evil Maid Attack has been rebranded the Evil Sanitation Worker Attack. So I think that really there's just - the world has changed, Steve. The world has changed.

Steve: There's just no hope. I don't know, Leo. Maybe 999 should be it because - no, I'm just kidding, everybody. No, 999 won't be it.

Leo: Old man, don't give up yet. It's not too late.

Steve: I can adjust.

Leo: Okay.

Steve: I can adjust. Although I'm sticking with man in the middle; you know?

Leo: Gotcha.

Steve: Okay. So that's out of my system. And I just completely distracted us from the article.

Leo: Yes. You can just call it MITM. That's what you call it.

Steve: MITM. Dark Reading refers to a piece from a site I've never encountered before called eSentire. I don't know why it's called that. I think all the good names were taken. So everything about this is a bit odd, since the article is written by a guy named Joe Stewart. And in the introduction, which he wrote, he refers to himself, as I mentioned before, in the third person.

Okay. I'm only going to share the first bit in his introduction because I'll take it from there. But Joe writes of himself, he said: "In the past year, the uptake of Passkeys has surged, with industry giants such as Apple, Microsoft, and Google championing their adoption. Joe Stewart, Principal Security Researcher with eSentire's Threat Response Unit (TRU), has been reviewing many of the leading software providers' implementations of Passkey technology and their current 'authentication process.'

"Regrettably, Stewart found that cybercriminals can still break into accounts protected by Passkey technology on many online platforms, such as banking, ecommerce, social media, website domain name administration, software development platforms, and cloud accounts, by using Adversary-in-the-Middle (AitM) phishing attacks. Stewart explains in this" - and I guess he should know because he wrote it - "blog how Passkeys are designed to work, how threat actors can currently circumvent Passkey security, and the steps that online software providers and websites that use or intend to use Passkey technology must take to ensure that their Passkey implementation is secure from threat actors."

Okay. Now, as I said, I read through Joe's article looking for this new problem that he had discovered. And there isn't one that I could find. All of this boils down to, if someone goes to a phishing site that's pretending to be the authentic site, the miracle that is

Passkeys won't protect you because the fake site won't offer Passkey authentication. Instead, it will offer any of the traditional authentication fallbacks that are still completely prone to interception by phishing.

Leo: See, this shows you how good Passkeys are. Right? In a way, this is proof, I mean, Passkeys work.

Steve: Yes.

Leo: Otherwise they wouldn't turn them off.

Steve: And it's unclear who would have ever believed otherwise, but Joe wants to be sure that everyone is on the same page with this. So, you know, for what it's worth, we have a new term, the Passkey Redaction Attack, which when you think about it, that's what you'd call this. It's a subset of phishing. Since the FIDO2 WebAuthn Passkeys technology is immune from phishing, the phishing site simply removes the Passkeys login option and gets you the old-fashioned way.

Leo: They probably don't want to use a YubiKey to log in, either.

Steve: Oh, no, it can't do that. You've just got to say...

Leo: Give me your password, buddy.

Steve: What's my pet's first name, or what's my porn name, or whatever it is.

Leo: Forget the passwords. Here's - what are your secret questions? That's what you need.

Steve: That's right. Wow. Okay. So those of our listeners who are worried, you know, basically, if you go to a phishing site, folks, you're in deep doo-doo; right? I mean, you clicked on a leak in email. You're at a bogus site, and nothing is going to help you. So don't do that.

Ahmad Khayyat from Riyadh, Saudi Arabia, said: "Hello, Steve. Instead of synchronizing Passkeys, isn't it more secure to have a Passkey per device, locked into that device's TPM or equivalent facility? Instead of backing up Passkeys, have backup Passkeys on additional devices. Moreover, it's probably more feasible to convince sites to support multiple Passkeys per user than to convince Google, Apple, and Microsoft to support Passkey portability."

I completely agree with that latter. The big problem here is that there's no way to know which sites support multiple Passkeys and which don't. You know, you're just going to try to associate another Passkey with a site from a different device, and the device says, sorry, you already got one. Use that one. But you can't because it's on the device you're not using right now.

The Passkeys spec states that Passkey supporting sites should provide many-to-one Passkeys to account mappings. But as we know today, not all do. And it only takes one to ruin your lunch. And running into a site that doesn't means that the user cannot add another device to that site, which is a breakdown of the Passkeys promise. Hopefully this will eventually change. But it's also true that having Apple, Google, and Microsoft performing their own cross-device synchronization of Passkeys, just as they've always done for passwords, takes the pressure off of sites to improve their Passkeys implementations. Because, right, works for everybody using Apple, Google, and Microsoft. What's wrong with you?

So for the time being, the only practical solution is to either have that be your complete and total solution, remain within one of the closed ecosystems which is provided by the big three, or use a third-party password manager such as 1Password or Bitwarden, both sponsors of the TWiT network, which will provide the kind of cross-platform compatibility that Passkeys was intended to provide, but doesn't yet universally.

And Passkeys was intended to provide it the right way, by having sites provide a many-to-one Passkeys to account mapping, then giving the user a user interface where they could see all of the Passkeys which are currently registered on their account, and administrate them. You know, say yes and no to various Passkeys. Like remove Passkeys for devices they're no longer using or don't want or have given a device to a family member, but they shouldn't have access to the family's banking site because, you know, they're not old enough yet and so forth. Anyway, we don't have that unfortunately. We can hope that we get that moving forward.

Max Feinleib said: "I'm guessing" - I love this because I was pronouncing it "noo-vo" mailer. He says: "I'm guessing that nuevoMailer..."

Leo: Nuevo. Nuevo.

Steve: Well, he was saying "nuevo."

Leo: Right. That's Spanish for new, yeah.

Steve: As in Spanish for new, yes, nuevoMailers. So now that he says it, and now that you've said it, Leo, and now that I've heard it, I'm sure that's the case, nuevoMailers.

Leo: Nuevo.

Steve: Thank you, Max. And I did get a nice email back from Panos, nuevoMailers author...

Leo: Oh, nice.

Steve: ...saying that he'd heard from some of our listeners. So believe me, as I said, I gave it an absolutely zero restraint endorsement because this thing, I'm just - you know, when you send out email to thousands of people in 30 minutes, it's a puckering sort of experience, and I keep expecting something to go wrong. But it just works. It's great. So nuevoMailer. Now I know how to even pronounce it. So a bonus.

Leo: Nuevo.

Steve: Yeah. Okay. So John, whose Twitter handle is @PsyVeteran, and I did get actually this from Twitter this week, he said: "Hey, Steve. Listening to you two discussing the OpenSSH bug in Security Now! 981 last week," he said, "I note that a port knocking setup would completely mitigate this front door camping. Love listening to you guys and excited for 999 and beyond. On a side note, while I am all signed up for email, I cannot see what address to send SN feedback on." Oh, so he fell back to Twitter because he didn't know how to email me. I'll get to that in a second.

So John's correct that port knocking would prevent exploitation of the OpenSSH flaw. And I suppose if one were really a belt-and-suspenders type, then adding that additional layer of pre-authentication authentication would make sense. But the problem is OpenSSH is supposed to be super secure, and the only protection anyone needs all by itself. Leo, you and I both authenticate our OpenSSH instances with passwords and certificates. A certificate is like a 2,048-bit secret key which cannot possibly be brute forced. Then a password is added to that, just to keep anyone who might obtain local access to the certificate from using it. So my point is, OpenSSH already provides so much security that anyone could be forgiven for not deploying an additional outer layer of protection.

But having said that, there is a larger, more general lesson to be learned here. And I'm glad John asked. And that's the inherent problem with the security of any monoculture. Relying entirely upon security from any single source is never going to be as safe as using multiple different sources of security. Or stated another way, using a "layered security solution" is able to provide the strongest security by far. And if you're going to go to the trouble of layering, it's even better when the various layers have nothing in common with one another. So John's observation that adding a "layer" of port knocking security is exactly that. And the advantage of port knocking is that it does not require a fixed IP for the "knocker." The whole point of knocking is that the secret knock identifies the IP of the knocker dynamically, whose traffic is then allowed to progress into the next layer of security.

I've often noted that I have all of the links between my various sites protected, not only with their own native security, whatever that may be and about which each of their vendors will boast at great length, but also with point-to-point IP address filtering so that none of those links are even visible to anyone else out on the Internet. Their own protection may be strong, but why take an unnecessary risk?

Of course, IP filtering is only feasible when the IP addresses of the endpoints are relatively static. But there really is no stronger security than adding a layer of simple IP address filtering to Internet traffic whenever and wherever possible. If anyone is listening to this who has publicly accessible endpoints which have fixed IPs, where IP-based filtering is not currently in place but could be, allow me to urge you to consider preventing anyone else from even being able to see those ports. I mean, it's almost magic. It's really worth it.

Now, John also mentioned that he signed up for our email system but didn't see our inbound email address listed anywhere. That's mostly by design, to tell the truth, because I would prefer to only give Security Now! listeners direct access to this mailbox. The address is shown on GRC's old feedback.htm page. And since many others have mentioned this, I've just changed the "reply to" address for the mailings to the Security Now! list to that correct address, which is "securitynow@grc.com."

So everyone who has already received today's podcast email, all 5,782 of you, containing this show's summary, the Picture of the Week, the full show notes link, will see that it was sent from "securitynow@grc.com." So simply hitting Reply to any of that Security Now! email will properly address a reply directly to me.

And speaking of heterogeneous layers of security, Dr. Brian of London wrote: "Fail2Ban. I can't believe any serious sysadmin is not running this." And he provides a link to its GitHub page, github.com/fail2ban/fail2ban. He said: "That completely negates this ssh vulnerability as far as I can tell." And as far as he can tell is exactly correct. After being installed into any Linux system, "Fail2Ban" creates a daemon running in the background.

The project describes itself by writing: "Fail2Ban scans log files like /var/log/auth.log and bans IP addresses conducting too many failed login attempts. It does this by updating system firewall rules to reject new connections from those IP addresses for a configurable amount of time. Fail2Ban comes out of the box ready to read many standard log files, such as those for sshd and Apache, and is easily configured to read any log file of your choosing, for any error you wish."

So, cool solution. It provides another layer of security based upon dynamic IP address filtering, and it's another layer that any mature security solution ought to have. Why would any service allow any single remote IP address to be pounding away with repeated authentication failures? It's just nuts. But sadly, that's still not built in. So this sort of protection, you know, it wouldn't block a widely distributed attack, where attempts are coming from a huge botnet, for example, with each bot having a different IP. But in this instance the exact timing required to exploit this recent SSH flaw means that it would be very difficult to launch such an attack through a widely distributed network of attackers.

So again, Fail2Ban. If you've got publicly accessible important services that can be intolerant and should be utterly intolerant of many failed login attempts, and you're running Linux or a compatible OS for Fail2Ban, it's free, it's open source. I can't imagine what the downside would be.

Leo: What a good idea. I mean, a lot of services have that turned on. My Synology I have that turned on. So after 10 failed attempts it's like, yeah, no, you're not getting in here ever again. I also block China and Russia, which is probably a good idea since I don't think I'll be logging in from there any time soon.

Steve: No. And in fact I have the same thing on my email server.

Leo: Right.

Steve: Because they're just, you know, they're not spamming valid addresses. They're just going - they're doing a dictionary attack.

Leo: Right.

Steve: First names A through Z.

Leo: Yeah.

Steve: Yes. And so after a server tries, I think it's five times with bogus accounts, I just put it on a blacklist for, I don't know, some length of time, and just say, look, just - I'm just not even going to entertain your guesses anymore because they could guess right at some point.

Leo: Right.

Steve: And I'd rather not have that crack with them.

Leo: It's a simple and effective defense.

Steve: Let's take another break, and then we're going to talk about "The Polyfill.io Attack." And, oh, baby.

Leo: It sounds, now that I think about it, I think it's a Pokmon is what it is. You don't know what Pokmon are, probably.

Steve: Of course I know Pokmon. I have one.

Leo: You know Pokmon?

Steve: I have several in the refrigerator, Leo.

Leo: Oh. Are they tasty? There is a Pokmon with that name, actually.

Steve: Actually, I was always the uncle who was able to get for his nieces and nephews the unavailable Christmas presents.

Leo: Ah.

Steve: Yeah. So...

Leo: So you do know a little bit about all this.

Steve: Oh, yeah.

Leo: That's a good uncle. All right, Steve. Continue on. Soldier on, my friend.

Steve: This is big.

Leo: Uh-oh.

Steve: We originally examined the consequences of the removal of a cloud-hosted service whose DNS CNAME record continued to point to the missing service. This then allowed ne'er-do-wells to establish their own replacement cloud service under the abandoned domain name as a means of hosting their malicious services within someone else's parent domain. That's not good.

Leo: Unh-unh.

Steve: But bad as that is, it's not the worst thing that can happen.

Leo: Oh.

Steve: Arguably, the worst thing that could happen would be for a domain which is hosting a highly popular, often downloaded code library to fall into nefarious hands.

Leo: Ah. That would be awful.

Steve: It would be awful. And I would not be talking about this this week if it had not just happened.

Leo: Oh, boy.

Steve: So first I'll back up a bit and explain what's going on here. An unhealthy "design pattern" that's developed is websites hosting pages which dynamically pull significant chunks of code from third-party sites. Back in the good old days, a website provided all of the pieces of the web page that it would present to its visitors. Naturally, advertising ended that with ads being filled in by third-party advertising servers on the fly. And though we've never had the occasion to talk about it directly, the same thing has become commonplace with code libraries. A perfect example came to mind with GRC's own XenForo web forums.

XenForo mixes a bunch of different technologies, with PHP being its primary implementation language on the server side. But the forums' browser-side polish, things like fading button highlights, some fancy animations, pop-up floating dialogs and so forth, are provided courtesy of JavaScript running in the user's browser. And rather than reinventing the wheel, XenForo, like most websites, takes advantage of the wildly popular jQuery library. Wikipedia explains.

They said: "jQuery is a JavaScript library designed to simplify HTML DOM (Document Object Model) tree traversal and manipulation, as well as event handling, CSS animations, and Ajax." Ajax being the browser's web page code itself making queries to other resources, typically back to the server as the user does things on the site. Wikipedia said of jQuery: "It is free, open-source software using the permissive MIT License. As of August 2022" - get this - "jQuery is used by 77% of the 10 million most popular websites." So just shy of four out of every five of the top 10 million websites are using jQuery. "Web analysis indicates that it is the most widely deployed JavaScript

library by a large margin, having at least three to four times more usage than any other JavaScript library."

Okay. So jQuery has become so widely and universally used and standardized that you could almost think of it as a browser extension itself since so much code takes advantage of it. If you're a developer of web pages that depend upon jQuery, you'd like to obtain the benefits of any bugs that are fixed, speed improvements, or other optimizations as soon as they become available. So rather than hosting the jQuery library yourself, you instead ask the browser to download the latest version of jQuery from the Internet. The question is, where should it be obtained? I've included a screenshot of one of the configuration dialogs for the XenForo forum system.

The option is labeled "jQuery source," and it has four radio buttons, so we get to pick one from among the four. The choices are: locally hosted, meaning that the server you're visiting provides the jQuery library itself; Microsoft CDN at aspnetcdn.com; jQuery CDN at jquery.com; or Google Ajax API CDN at googleapis.com. And the default from the XenForo folks was the final choice, to obtain the library on the fly, every time it's needed, which is all the time, like every page, directly from Google.

Now, we might wonder why Microsoft and Google are interested in being benefactors by providing access to code libraries on their content delivery networks. But remember, 77% of the top 10 million websites are all using jQuery. In a world where third-party cookies are able to track their users, every time any web browser anywhere fetches a copy of the jQuery code library from Microsoft or Google, that browser's cookie for that CDN is sent along with the site the user's browser is visiting and the user's current IP address because that's where it's connecting from. So Microsoft or Google will obtain some information to add to the pile they already have about every user's whereabouts and "whenabouts." In return, our browsers probably don't need to wait very long to receive that library since both of those CDNs are extremely high performance.

Okay. So the main point here is that a model has gradually evolved over time where websites we visit are no longer providing all of the content that runs in their pages, and that in addition to ads and Gravatars and web fonts and a bunch of other stuff, today's web pages also contain significant code libraries from third-party servers.

At the beginning of our discussion of this I noted that "Arguably, the worst thing that could happen would be for a domain which is hosting a highly popular and often downloaded code library to fall into nefarious hands. And I would not be talking about this if it hadn't happened." Armed now with this bit of background, we can easily understand the consequences of this and of how serious it could be.

So what happened? First we need to answer the question, what's a polyfill? There are times when a device or a library may not be the latest and greatest, like when someone is still using an older version of Internet Explorer which may not support the latest HTML standards. What can be done in such cases is that a so-called "polyfill" library can be used. The fill is that it's filling in for missing APIs.

So a website that wishes to be able to run on the widest range of browsers, while still being able to take advantage of the latest features of the most recent browsers, can choose to have its web pages download and invoke a polyfill library. Then, when the page's code runs, the polyfill library will check to see whether the underlying web browser supports the various features that the site wishes to use. And if the answer is "no," the polyfill library itself will make up the difference. It'll fill in the gap. It'll fill in for the down-version browser by emulating the functionality that's natively missing from that browser. So in short, polyfilling is another popular practice and library. It's mostly used to fill in for missing JavaScript features, but both PHP and Python have available polyfills, as well.

So just as with the wildly popular jQuery library, polyfill libraries, while not as wildly popular as jQuery, are still in wide use. That makes it, let's just say, very bad when a Chinese company purchases the Polyfill.io domain and its associated GitHub account...

Leo: Oh. Oh.

Steve: Uh-huh, and then proceeds to do bad things with them. Essentially, this company purchased a position of extreme power and responsibility and then chose to misbehave. Exactly two weeks ago, on June 25th, the forensics team at the security site "Sansec" broke the news with the headline "Polyfill supply chain attack hits more than 100,000 sites." And before I go on, I'll just note that, as we'll see, their initial estimation of more than 100,000 sites fell short by around 280,000.

Leo: Oh.

Steve: The real number has turned out to be more than 380,000 web sites. They wrote: "Polyfill.js is a popular open source library to support older browsers." And they said 100,000-plus. We now know it's 380,000-plus sites. They said: "Embed it using the cdn.polyfill.io domain. Notable users are JSTOR, Intuit, and World Economic Forum. However, in February this year, a Chinese company bought the domain and the GitHub account. Since their purchase, this domain was caught injecting malware into mobile devices via any site that embeds cdn.polyfill.io. Any complaints were quickly removed from the GitHub repository.

"The polyfill code is dynamically provided based on the HTTP query headers, so multiple attack vectors are likely. Sansec decoded one particular malware which redirects mobile users to a sports betting site using a fake Google analytics domain." And I love this. Get this, Leo, www.googie-anaiytics.com. So get your googie anaiytics here. "The code has specific protection against reverse engineering, and only activates on specific mobile devices at specific hours. It also does not activate when it detects an admin user. It also delays execution when a web analytics service is found, presumably to not end up in the stats.

"The original Polyfill author recommends that Polyfill should no longer be used at all, as it is no longer needed by modern browsers. Meanwhile, both Fastly" - and we'll see where Fastly figures in here because it's significant - "and Cloudflare have put up trustworthy alternatives, if you should still need it. This incident is a typical example of a supply chain attack."

Now, they followed up this report with a series of four updates, one per day, starting the same day. So on the 25th of June, 26th, 27th, and 28th. First they updated: "Google has already started blocking Google Ads for eCommerce sites that use Polyfill.io." The next day: "Someone launched DDoS attacks against our infrastructure and BleepingComputer, who was the first to cover our research." The day after that: "Cloudflare has implemented real-time rewrites of cdn.polyfill.io to their own version." A little later: "Namecheap has put the domain on hold altogether, which eliminates the risk for now. However, you're still recommended to remove any Polyfill.io references from your code."

And finally, on June 28th: "We are flagging more domains that have been used by the same actor to spread malware since at least June" - so a year ago - "of 2023: bootcdn.net, bootcss.com, staticfile.net, staticfile.org, unionadjs.com, xhsbpza.com, union.macoms.la, and newcrbpc.com." So a big problem with this code.

And then Censys weighed in with some terrific reporting based upon their extensive visibility into the Internet. I should note that my server logs are full of port probes from Censys. They want to know what's going on and who's running what, where. They're another site like Shodan. But the nice thing, for me at least, is that their reverse DNS resolves to their domain name. So at least, if I'm curious, I can see who's knocking at the door. I still don't let them in, but it's marginally less creepy to know that it's not an attacker, you know, trying to actually get in. It's just somebody, as their name suggests, doing a census of the Internet.

Anyway, last Friday, under the headline "Polyfill.io Supply Chain Attack - Digging into the Web of Compromised Domains," Censys wrote: "On June 25, 2024, the Sansec forensics team published a report revealing a supply chain attack targeting the widely-used Polyfill.io JavaScript library. The attack originated in February 2024 when Funnull, a Chinese company, acquired the previously legitimate Polyfill.io domain and GitHub account. Shortly thereafter, the service began redirecting users to malicious sites and deploying sophisticated malware with advanced evasion techniques.

"On the 27th of June, Namecheap suspended the malicious Polyfill.io domain, mitigating the immediate threat for now. However, Censys still detects 384,773 hosts embedding a polyfill JS script linking to the malicious domain as of July 2,2024. These hosts include websites associated with major platforms like Hulu, Mercedes-Benz, and Warner Bros. Security experts strongly advise website owners to remove any references to Polyfill.io and its associated domains from their codebase as a precautionary measure. Cloudflare and Fastly have offered alternative, secure endpoints for polyfill services as a workaround.

"Further investigation has uncovered a more extensive network of potentially compromised domains. Researchers identified four additional active domains linked to the same account that owns the Polyfill.io domain. Censys detected 1,637 million hosts referencing one or more of these endpoints. At least one of these domains has been observed engaging in malicious activities dating back to June of 2023, but the nature of the other associated domains is currently unknown." And you've got to ask yourself, why would 1,637,000-plus hosts be referencing these domains? I mean, they've got to be thinking that they're pulling something from them. It's just mindboggling. "This incident," they write, "highlights the growing threat of supply chain attacks on open-source projects."

So just to be clear, 1,637,160 website hosting servers are currently emitting web pages that are causing their visitors' web browsers to download resources from these malicious domains. Speaking of the hosts that they have found attempting to pull from Polyfill.io, Censys added: "The presence of domains like 'www.feedthefuture.gov' in these top results also highlights the use of Polyfill.io across various sectors, including government websites. In total, Censys observed 182 affected hosts displaying a '.gov' domain." In other words, 182 .gov websites were pulling code from Polyfill.io owned by a malicious Chinese company. Which is where we interject: What could possibly go wrong?

"While estimates of the scale of affected websites vary widely between sources," they said, "Sansec reported 100,000, while Cloudflare suggested 'tens of millions,' it's clear that this supply chain attack has had a widespread impact. Further investigation has uncovered an extensive network of potentially related domains. A Twitter user discovered that the maintainers of the polyfill GitHub repo had leaked their Cloudflare API secrets within the repo." Whoops.

"It was thereafter discovered that the leaked Cloudflare API key is still active and shows four additional active domains linked to the same account: bootcdn[.]net, bootcss[.]com, staticfile[.]net, and staticfile[.]org." And I should just mention that we're all able to put Polyfill.io and those other four domains into our hosts file, or into whatever local resolver

we have, and blackhole them so that for no reason would anyone in our own networks or our browsers successfully download code from those sites, where they never would be wanting to.

They wrote: "Bootcss[.]com, has been observed engaging in malicious activities that are very similar to the Polyfill.io attack, with evidence dating back to June of 2023. The two main malicious domain names involved were both registered at the end of May. The evil jQuery was introduced by highlight.js hosted on cdn.bootcss.com. When the request for highlight.js has a specific Referer and mobile User-Agent, the server will return highlight.js with malicious code; otherwise, it returns normal code, which is highly disguised. Moreover, there are specific conditions for whether malicious jQuery is introduced when this code is executed."

Okay. In other words, matching on a specific referrer enables the code to only target users visiting specific websites, and matching on the mobile device's User-Agent causes it to only target users using specific device types. So a highly targeted attack which, because it's delivering benign code most of the time, would easily go unseen.

So this is all really quite bad. The one person whose reactions we've not yet heard from is Andrew Betts, the originator of polyfill. Not Polyfill.io, just polyfill. Way back on February 25th, after the change of ownership of Polyfill.io had occurred and before all this came to a head, Andrew tweeted. This is the originator of the polyfill process, the polyfill library for web browsers. He tweeted: "If your website uses http://polyfill.io, remove it IMMEDIATELY," all caps. He says: "I created the polyfill service project, but I have never owned the domain name, and I have had no influence over its sale." So he wants to fully disavow himself of any association or responsibility for what might happen.

He said: "No website today requires any of the polyfills in the Polyfill.io library. Most features added to the web platform" - meaning the polyfill web platform - "are quickly adopted by all major browsers, with some exceptions that generally cannot be polyfilled anyway, like Web Serial and Web Bluetooth. Domains that serve popular third-party scripts" - so listen to this. This is Andrew. "Domains that serve popular third-party scripts are a huge security concern. The team at google-analytics.com, for example, could read or modify almost any website in the world, like gov.uk or wellsfargo.com. If you own a website, loading a script implies an incredible relationship of trust with that third party. Do you actually trust them?"

And I should explain that, as I said, Andrew never endorsed the distribution of his polyfill library via a third party. He intended to have the hosting site that needed to use the library host it themselves, just like in the good old days. He's clearly aware of the havoc that would ensue if any major third party - like Google with their ubiquitous analytics code being injected into every website there is - were to go rogue or be compromised by a sophisticated attack. But somewhere along the way, someone - not Andrew - had the bright idea of hosting polyfill at Polyfill.io, and over time everyone started to use it rather than host it themselves. And this was the crucial mistake that so many in the industry made.

Think about it: You've purchased TLS certificates to protect your site; right? You have electronic and physical security, maybe even hardware security modules to protect your secrets. You've implemented Passkeys to offer the latest state-of-the-art public key login authentication. You've gone through the popular checklists of all the things you need to do to be secure, and everything is checked off on them. You've done everything you can think of. Then you have every one of your precious and secure site's web pages downloading and running active JavaScript code that runs with full permissions in a first-party context from an unknown entity in China that turns out to be malicious. What's wrong with this picture? What's right with this picture? It's nuts.

So what happened in this case? And Leo, after sharing our final sponsor, we're going to look closely at who was behind Polyfill.io. And if it was the Financial Times, would you be surprised?

Leo: Yes.

Steve: Because it was.

Leo: It was? This is a cliffhanger. Okay. Now Steve's going to explain what the Financial Times has to do with this mess.

Steve: So what happened? Without some deeper investigation, it's impossible to say exactly. I took a look back in time, thanks to the Internet Archive's Wayback machine. The Wayback machine began taking snapshots of the Polyfill.io site back in 2013, so 11 years ago. And it certainly all looked aboveboard and solid for years. It was a service being brought to the world by the Financial Times with bandwidth and CDN services provided by Fastly, and the public domain polyfill code present on GitHub.

An early version of the site's homepage said: "Just the polyfills you need for your site, tailored to each browser. Copy the code to unleash the magic." And then there's a simple line. It's got an open angle bracket, and it says <script src="https://cdn.polyfill.io/v2/polyfill.min.js">. And then all that's closed, and then you close the script. That will cause the browser that reads that line to go to that domain through the CDN, pull polyfill.min.js, deminify it, and execute it. And what it then chooses to do is entirely up to it. You've loaded it because of the privileges it needs as a first party in your own domain's code. It can do anything to your page it wants. It can take the esses off of the HTTPs. It can log the username and password that are entered into your page's forms. It can do whatever it wants.

They said: "Polyfill.io reads the User-Agent header of each request and returns polyfills that are suitable for the requesting browser. Tailor the response based on the features you're using in your app, and see our live examples to get started quickly.

"The polyfill service is developed and maintained by a community of contributors led by a team at the Financial Times. It evolved from a previous service developed by Jonathan Neal, and our cdn.polyfill.io domain routes traffic through Fastly, which makes it available with global high availability and superb performance, no matter where your users are."

So at this point the Financial Times is referring to cdn.polyfill.io as "their domain." But things changed, and it's interesting to trace the site's evolution through the years. The Wayback machine makes that easy. What I discovered was that something happened toward the end of the year last year, on November 1st of 2023. The day before, on October 31st, Halloween of last year, the site was offering polyfills while boasting that in just the last 30 days it had fulfilled 61.2 billion requests while delivering 292.98 trillion bytes worth of polyfills, and that was just in the past 30 days. 61.2 billion requests offering nearly 300, just shy of 300 trillion bytes in polyfills. And Fastly's name was still proudly highlighted in red against a black background.

But then, the next day on November 1st, the site's pages changed. Gone was the activity tracking, the reference to GitHub, or any mention of Fastly. So as I said, it would take interviewing the parties involved to learn more. One guy who would likely know the whole story is Jake Champion, whose Twitter handle is @JakeChampion. His personal bio at JakeChampion.name mentions his involvement with the polyfill project, he's at

Financial Times, and his copyright can be found at the bottom of the earlier pages. But he has his Twitter account locked for viewing only by approved followers, so I wasn't able to say whether he may have been tweeting something about what had been going on.

What's clear is that maintaining the 100% free polyfill service was doubtless becoming expensive, and it was the definition of thankless. Eleven years earlier it would have been much easier than it was today. And with nearly 300TB of polyfills being served every 30 days, that's 10TB per day, day in and day out, in return for nothing.

So it's not difficult to imagine someone coming along and offering to purchase that massive burden. Or maybe just take it off the Financial Times' hands. We don't know. I've not found anything about that yet. Maybe some of our listeners will know or will find out. I'd love to know. But however it happened, in the blink of an eye, overnight, suddenly 384,773 websites, or tens of millions if you take Cloudflare's number, were suddenly inviting an unknown stranger's JavaScript code into and onto their previously super-secure and secured pages, thus bypassing and making a joke of everything else they had done to create secure websites.

While digging around for all of this interesting information and backstory, I ran across one other chilling bit. This was posted over on Medium by Amy Blankenship, who describes herself as a full stack developer at a financial technology company. She primarily writes about React, Javascript, Typescript, and testing. And under the headline "Those Scary Stories About Polyfill.io? They're Just the Beginning," she posted the following to Medium.

She said: "The Internet exploded this week about how Polyfill.io was injecting malicious code into websites that were linking to it from the CDN at the URL where it has been served for years. The thing is, ownership of the GitHub repo and the download domain were transferred in February. They didn't wait until this week to start making changes. Here's how I know.

"In February, we started to get mysterious errors when some of our users tried to log in. The stack trace the Sentry error boundary was giving us didn't make any sense. It was deep within the okta-react library code we were using. But we had not recently upgraded our okta-react, okta-js, or Sentry code or the code that called it.

"Long story short, in the course of stepping through the code in the debugger, I discovered that some code in Okta was expecting to receive an object that had iterable properties. Sometimes when it received the object, those properties could not be iterated. Digging further, I found that the object was returned from code deep within the polyfill library."

In other words, Okta, the major identity, cloud security, and access management company, currently worth $6 billion, had some deeply embedded dependency upon the Polyfill.io library, which it was pulling from Polyfill.io.

Leo: Doh.

Steve: Thus from China.

Leo: No. Good on Amy Blankenship for finding that. That's amazing.

Steve: Yup. Wikipedia reminds us: "Okta, Inc. is an American identity and access management company based in San Francisco. It provides cloud software that helps companies manage and secure user authentication into applications, and for developers to build identity controls into applications, website, web services, and devices. It was founded in 2009 and had its IPO offering in 2017, reaching a valuation of over $6 billion."

That's right. And its core, Okta-react and Okta-JavaScript library code was pulling from a library that is now being supplied by a clearly hostile and malicious company named Funnull, based in China.

Leo: Wow.

Steve: We know that because shortly after Funnull acquired the Polyfill.io domain and GitHub account, the Okta library itself began mysteriously acting up. And this, of course, brings us to, at the top of page 17 of the show notes, Randal Munroe's wonderful xkcd cartoon showing a large and ungainly collection of stacked blocks reaching up to the heavens. We would call it a "house of cards" if it were composed from playing cards rather than blocks. That detailed and stacked assortment is then labeled "All modern digital infrastructure."

And then down near the very bottom, off to one side, is a crucial twig whose presence is holding up the entire edifice, the whole assembly, such that without it everything would come tumbling down. And that little twig is labeled "A project some random person in Nebraska has been thanklessly maintaining since 2003." Unfortunately, in this case, as the result of a number of unfortunate events, it would be labeled "Control acquired by a hostile and malicious Chinese company."

Now, Leo and I live in California. This state has a major, seismically active, geological fault running through it known as the San Andreas Fault. The San Andreas Fault runs North-South about 750 miles through California, forming part of the tectonic boundary between the Pacific Plate and the North American Plate. This is the same fault that Lex Luthor was determined to trigger...

Leo: With a nuclear bomb, yeah.

Steve: ...and use to turn California's eastern neighboring state of Nevada into beachfront real estate by sinking all of California into its Pacific ocean. Fortunately, well, we have Superman, so that hasn't happened yet. The relevance of this is that the somewhat precarious state of our Internet's security infrastructure puts me in mind of the nature of seismic faults, and the earthquakes that accompany them. As tectonic plates slowly shift, pressure builds up, and that's not a good thing. So here in California what we're looking for - and we're hoping for - is a more or less continuous series of small earthquake tremors where no one is hurt, and the china plates remain on their shelves. People text each other asking, "Hey, did you just feel that one?" That's much better than a long period of quiet, which allows massive pressures to build up, only to be released in a single massive event. So what we want here in California is lots of small, barely noticeable events.

How many times on this podcast, especially over the last five years, where it seems to be happening more frequently, have I noted that we all just dodged a bullet? That the Internet just experienced another tremor? We discovered a serious problem that did not take everything down with it. So here again, the industry just received another wakeup

call with no known actual, let alone horrific, damage being done, and an important lesson was hopefully learned that was not expensive. Let's all hope that things remain like this, with a continuing series of small and harmless little earthquakes, rather than, as it's referred to in California, "the big one" hitting.

Leo: This could have been the big one. Big time.

Steve: This could have been...

Leo: Holy cow.

Steve: ...really bad.

Leo: We don't know how many sites chose Polyfill.io to load that library, though. It could have been...

Steve: Yes. Yes. 1.67 million.

Leo: Oh, we do.

Steve: 1.67 million sites. We have a count.

Leo: And got it from Polyfill.io.

Steve: They got it from the CDNs looking at the pages and seeing how many.

Leo: Right.

Steve: And also, right, the various sites that are pulling from Polyfill.io right now after taking control of that domain.

Leo: Wow.

Steve: Millions, millions of different websites.

Leo: So somebody does, somebody benign now controls it. So we're okay.

Steve: Yes. Namecheap took the site down. They took the domain name.

Leo: Do they offer the library or no?

Steve: Both Cloudflare and Fastly are hosting the final benign version for anybody who needs it. But as its author said, nobody does anymore.

Leo: Right, right.

Steve: So a lot of this, again, this is also stupid inertia hurting us.

Leo: Running on automatic, yup, yup.

Steve: All that crap was left in the pages because no one's sure if we need it or not, so we'd better leave it there. Meanwhile, China owns the domain, and it's pulling JavaScript as a first party into every one of those pages' code, that could do anything.

Leo: It's the big one. It's the big one. But it isn't. We dodged a bullet. Amazing story.

Steve: Yup.

Leo: What a great story. And a cautionary tale, as you point out. That's why you listen to this show, so you know, you're prepared, and you're keeping an eye out. And again, credit to Amy Blankenship, who noticed a weird anomaly with her okta logins.

Steve: Yup.

Leo: Said, you know, something's not right here. And that's a note to everybody. When little weird anomalies happen, they're not necessarily the big ones, just little ones. That's how Clifford Stoll noticed the bad guy in his network, because there were one or two cent discrepancies in the balance sheet.

Steve: And actually I'll bet our listeners are as twitchy as I am now.

Leo: Yeah.

Steve: I see, like, some spike in bandwidth, I go, wait, whoa, what, what...

Leo: It's like, whoa, what's going on?

Steve: What's going on?

Leo: If it's out of the ordinary, it needs to be investigated. Wow. What a story. You'll hear more stories like that every week on Security Now!. Steve Gibson is the man in charge. You can get on his mailing list or send him email by going to GRC.com/email. Then you can verify, validate your address, sign up if you wish for the free newsletters, including, by the way, the show notes, so you get a picture, a peek at the Picture of the Week ahead of time. GRC.com/email.

Now, while you're there, you can also buy a copy of Steve's bread and better, didn't mention it this week, but SpinRite pays the bills for Steve. And it is a very useful, must-have, frankly, tool for anybody with mass storage for both performance, reliability, and maintenance, and recovery, if necessary. GRC.com. Look for SpinRite. Lots of free stuff there, including ShieldsUP!. ValiDrive, which is really a great thing.

And of course this show. Steve has the usual 64Kb audio, you know, the kind of traditional version. He also has a nontraditional 16Kb version for people with very little bandwidth and no real hearing. If you could put up with it, you know, you'll save a lot of bits. Many bits will not have to die for your listening. He made it for Elaine Farris who does those wonderful transcripts of every episode. All that's at GRC.com, along with the show notes.