A Fowl Incident

Description: This week's answers are many: How has Fosstodon survived a sustained DDoS attack? Or has it? What luck have Europol and the FBI had with taking down DDoS-for-hire services, and have they returned? What's the point of blocking TikTok, and is it even possible? What happens when government-backed surveillance goes rogue? What exactly is "Strategic Objective 3.3," and what, if anything, does it portend for future software? Should you enable GitHub's new secret scanning service and get scanned? What exactly did CISA's secretive red-team accomplish, and against whom?

High quality (64 kbps) mp3 audio file URL: http://media.GRC.com/sn/SN-913.mp3
Quarter size (16 kbps) mp3 audio file URL: http://media.GRC.com/sn/sn-913-lq.mp3

Which messenger apps have been banned by Russia, who's missing from that list, and why? What exactly is old that's new again, what happens when everyone uses the same cryptographic library for their TPM code, what's the latest WordPress plug-in to threaten more than one million sites, and why has Russia fined Wikipedia? And once we've put that collection of need-to-know questions to rest, we're going to examine the surprising revelations that surface as we unearth the fowlest of recent security incidents.

SHOW TEASE: It's time for Security Now!. Steve Gibson is here. Lots to talk about. A DDoS attack against Mastodon. Steve's take on blocking TikTok. Does it accomplish anything? GitHub's new secret scanning service, something we should all probably be running. The latest WordPress plugin that's threatening more than a million sites. Russia fining Wikipedia. And Steve digs deep on the case of the chicken bone points or something. It's all coming up next on Security Now!.

Leo Laporte: This is Security Now! with Steve Gibson, Episode 913, recorded Tuesday, March 7th, 2023: A Fowl Incident.

It's time for Security Now!. You've been waiting all week, so have I, to hear from this guy right here, Steve Gibson, our securitarian.

Steve Gibson: Whoa, securitarian. What?

Leo: Our Secretary of Security.

Steve: That's right, the geritarian or something.

Leo: Every week we get together. Steve explains why the world's going to hell in a hand basket. I suspect, Steve, I may be wrong, but I suspect you will be answering some pretty important questions this week.

Steve: This week's answers are many, as it turns out. How has Fosstodon survived a sustained DDoS attack? Or has it? What luck have Europol and the FBI had with taking down DDoS-for-hire services, and have they returned? What's the point of blocking TikTok, and is it even possible? What happens when government-backed surveillance goes rogue? And exactly what is Strategic Objective 3.3? What, if anything, does it portend for future software? Should you enable GitHub's new secret scanning service and get yourself scanned? What exactly did CISA's secretive red team accomplish, and against whom? Which messenger apps have been banned by Russia, who's missing from that list, and why?

What exactly is old that's new again? What happens when everyone uses the same cryptographic library for their TPM code? What's the latest WordPress plugin to threaten more than one million sites? And why has Russia fined Wikipedia? And once we've put that collection of need-to-know questions to rest, we're going to examine the surprising revelations that surface as we unearth the fowlest of recent security incidents. Thus today's podcast: A Fowl Incident.

Leo: When you say "fowl," F-O-W-L.

Steve: Yes, indeed.

Leo: People who are not watching will not maybe immediately get the pun that you have made.

Steve: A Fowl Incident.

Leo: It's a fowl [clucking]. I can't wait to find out what that's all about.

Steve: And of course we do have another great Picture of the Week.

Leo: Yes.

Steve: Which makes you question humanity once again.

Leo: I saw it, and I went, what? Steve, I think, I bet, I know you've got a Picture of the Week.

Steve: So yes, as I said, this one will make people question humanity once again. I gave this one the caption "Why real world testing is important." And this picture was provided courtesy of Simon Zerafa, a friend of the show. It's a photo of a notice that was posted in some organization somewhere on the wall. And it explains what you should do. It says:

"Please, when using the stairs, stay to the right when going up; stay to the left when going down."

Leo: Oh, dear.

Steve: "This will keep people from running into each other," it says.

Leo: No, no. Quite the opposite. This will guarantee that people...

Steve: Yeah. And you also kind of have to wonder, like, are people having their eyes closed when they're using the stairs at this location?

Leo: You shouldn't need this notice, obviously.

Steve: Yeah, you know, you ought to see people, like, approaching you on the same side, and one of the two of you decides to switch sides. But of course this was just wonderfully written from, like, the perspective of someone standing at the bottom looking up the stairs, not from the perspective of the person actually doing the walking.

Leo: No. Would have been sufficient just to say "stay to the right."

Steve: Please stay to the right, yeah.

Leo: Yeah. That's all you had to say.

Steve: Like when you're driving on a street, like, you know, similar to that.

Leo: That's what everybody does anyway.

Steve: But no, we're going to be more - as I said, why real-world testing is important. You should test this signage after you post it and see what happens, and then consider changing it.

Okay. Last Thursday, Chris Miller posted on Fosstodon: "Hi, all. We're still under a major DDOS attack, and that's why mobile and desktop clients are not currently working. We've had to put the site behind Cloudflare temporarily until the attack stops. We're looking at other long-term solutions, but we need to get through the current moment first. For now, use the web interface. Thanks for your patience." And as of yesterday, when I last looked, that attack was still ongoing.

Fosstodon, as its name suggests, F-O-S-S-T-O-D-O-N, is the largest Mastodon instance inhabited by open-source software denizens. And sadly, this most recent attack marks only the latest in a growing string of DDoS attacks that have hit and easily brought down unprotected Mastodon server instances over the past few months. And Leo, you were saying now that mastodon.social is in trouble.

Leo: Yeah. Gargron, who created Mastodon, runs mastodon.social, his was down. He tooted a couple of days ago. And last week, during this show in fact, I think it was, I was getting a lot of messages from people saying is your site - is twit.social down? We run a Mastodon instance. And I check, you know, we have a thing called Sidekick running that talks about traffic and in particular how many queues are open, how many have been processed, how many haven't, and how many have failed. And it was through - the number of queues was through the roof, which is sometimes a sign of too many people using the site. But nothing had happened to make that the case. So while I don't know, I'm going to guess that was also a DDoS attempt. People were flooding us with posts.

Steve: Or with connection attempts.

Leo: Yeah, or connection attempts.

Steve: So that would have overflowed the inbound queue, yeah.

Leo: That's right. Yeah, Sidekick would have died, yeah.

Steve: So anyway, what's happened is, and here's like some anecdotal examples we just had, the pace of these attacks has increased significantly after Mastodon gained a huge amount of attention thanks to the mass exodus from Twitter following Elon's takeover and his subsequent actions, which struck many as being in the best interests of neither the larger Twitter community nor their own individual best interests. So fearing the approaching end of Twitter, many jumped over to the Mastodon's decentralized model, which inherently prevents a repeat of essentially what Elon is doing at Twitter.

Leo: It also means that DDoS can't bring the whole thing down; right? Because...

Steve: Well, it can't bring...

Leo: Individual instances down.

Steve: Exactly. Being federated, there's other places you could go. There's a guy, feditips, feditips@mstdn.social.

Leo: Yeah. That's good, yeah, yeah.

Steve: As in federated tips. And he posted, or that person posted: "The mastodon.social server is currently under a heavy DDOS attack and may not work properly. The 12,164 other servers on the network are unaffected. This is part of the reason why federated networks are a good idea: if one server goes down, the others work fine. The more spread out we are on small and medium-size servers, the harder it is for anyone to take down the network because there's no obvious target." He says: "There are also many

other reasons why federation is good." And then he provided a link to fedi.tips/why-is-the-fediverse, hyphenated.

So there's not much more to say here other than to note that fedi.tips is certainly correct. On the one hand, the nearly week-long DDoS attack against Fosstodon has rendered its mobile and desktop clients inoperable. But thanks to their web frontend moving behind Cloudflare's frontend protection, at least their web interface is operable, and they're remaining on the air.

As I noted recently, when we were talking about the most recent DDoS connection rate attack record being broken, no standalone server on the Internet can withstand today's DDoS attacks, not even for an instant.

Leo: Is that because they're amplified? Is that why they're so strong?

Steve: Yes, yes. Today's modern IoT botnet-based attacks are large enough to swamp even medium-size bandwidth providers. That is to say, you know, it might even be that the traffic doesn't actually reach your server because it brings down the routers upstream of your server because these attacks are just so big. So it's not that your server is even down. It's that the traffic from outside on the Internet can't even reach it because the attack just swamps the aggregating routers before they have a chance to even get to your server. So the only recourse - I mean only - if you have to be on the 'Net is to move behind the protective skirts of one of today's major DDoS protection services. The problem is, such service is not free unless the service wishes to provide such service charitably.

And that's one thing that sort of annoyed me. Elsewhere, on Reddit, the Fosstodon admin was grousing a little bit about the need to be rescued by a commercial service such as Cloudflare. In my opinion they should consider themselves incredibly fortunate that such a facility like Cloudflare exists. Otherwise, they would be off the 'Net for as long as the DDoSing cretins wanted to keep them off the 'Net. And I've had experience with that in my past. And it is in fact, I mean, we've got an interesting graph coming up next which gives us a sense for the scale of the problem, not only in attack size, which as we've said is now just, I mean, it's incredible amounts of attack, where to call it overkill is an understatement because it would just - the attacks are so large that they just melt the wires, essentially, between the server being attacked and the outside world.

Okay. So on the subject of DDoS attacks, the network security provider NetScout has noted that the efforts by Europol and the FBI to take down more than 50 DDoS-for-hire services in the middle of last December has indeed led to a measurable decline in DDoS attacks. I've got a chart in the show notes that demonstrates that visually. Those declines have been recorded at broadband providers across both the U.S. and EU. Moreover, NetScout said that DDoS traffic has remained at lower levels for the month after those takedown efforts. This suggests that no new players have immediately appeared on the underground DDoS market to fill the void after last year's takedowns. On the other hand, four or five weeks isn't much time to wait for a new service to get itself up and going.

So to illustrate their data, NetScout provided this nice chart that we've got on the screen right now so that we were able to see for ourselves. That chart leaves little doubt that the takedown effort had its intended effect. Where the takedown occurred, you could see a dramatic decrease in - something. So, wondering exactly what that something is, you then look at the horizontal scale to learn that the chart's horizontal axis is in attacks per day. So you learn that while, yes, the number of attacks per day did indeed fall from their peak of around 3,500 DDoS attacks per day globally, and that peak was reached

twice, and a number of shorter peaks, like more than 3,000 were reached another four times, still, a few weeks ago, a month after this takedown, actually it was the 22nd of January, there were 1,000 attacks in that day. And of course it only takes one attack to ruin a particular website's day.

So it's definitely good that law enforcement is on this, and is taking down these DDoS-for-hire services since they commoditize DDoS attacks. They allow anybody to send a little bitcoin somewhere and say, I want this site removed for X length of time. I think, you know, you pay by the minute how long you want the site to be gone. So that's not the kind of commodity that you want around. And I'm delighted that the FBI and Europol are on that. But we're still seeing individual attacks breaking records, so it's clear that there's still plenty of firepower left, to which Fosstodon and Mastodon.social can both attest. Fosstodon moved itself behind Cloudflare, got their website back up. Once the attack abates, they will presumably move back.

And depending upon the size of the grudge or whatever it is, I mean, like who knows why Fosstodon is being attacked. Part of the problem is these attacks are so trivial to execute now that they don't really have to mean anything. It's just, if you've got an idle botnet that can do DDoS attacks, well, you know, spin the wheel and pick your target. It's, I mean, it's sad, and it's probably the biggest weakness, the fundamental weakness that we have in the way the Internet works with its autonomous routing, is that the brilliance of it is that you can drop a packet on the network anywhere in the world, and the routers will forward that packet to its destination IP. And that's also the bad news because, if you've got a whole bunch of people all dropping packets, all aimed at the same IP, the global Internet will concentrate all of those packets into that single location and create a single point of failure. And that site is no longer able to respond to good packets because the malicious ones have just flooded it.

So that hasn't been fixed, and I don't see any fix for that. It would take, you know, as we've talked about this before, right, some clients that are generating DDoS attacks are able to spoof their source IP. They're able to just make up whatever IP they want as the packets are leaving. Well, that traffic could be blocked at the first time it comes to a network border because that network border knows very well what networks are inside of it.

Leo: You're saying an ISP specifically.

Steve: Correct.

Leo: I mean, we've always - we've said this for years, that ISPs have got to be doing this, have got to block outbound traffic that's not originating from IP addresses inside their range.

Steve: Exactly. It's called "egress filtering." And all the technology is there. It's just...

Leo: I can't believe they're not doing it still.

Steve: I know.

Leo: That's kind of stunning. I thought they were. Maybe, well, you know, maybe Comcast is. Maybe the big ones are. But all it takes is one little Ukrainian ISP or something; right?

Steve: True. And the argument has been, well, then bots would then start making up IPs within the network so you didn't know where in the network it was. And the bots are also disposable; right? They're all compromised routers, you know, that exist in back closets and corners. So in fact it's not like they're people who need to avoid or evade law enforcement. They're just, you know, IoT devices that have long been forgotten, that have not been patched, that have been taken over remotely, and now they're serving somebody else's ends. So we've got a sort of a mess, and it's not being resolved quickly.

So, okay. Last Thursday the headline in Gizmodo read: "We Found 28,000 Apps Sending Their Data to TikTok. Banning the App Won't Help." I'm just going to share the beginning of Gizmodo's very long article since it contains most of the useful information. Gizmodo writes: "President Joe Biden gave federal agencies 30 days to remove TikTok from government devices earlier this week." And that means that was last week. "Until now, most politicians," Gizmodo is saying, "intent on punishing TikTok, have focused solely on banning the app itself. But according to a memo reviewed by Reuters, federal agencies must also 'prohibit Internet traffic from reaching the company.'" And Gizmodo says: "That's a lot more complicated than it sounds." And I'll interject here, a lot less complicated than Gizmodo thinks, but we'll get to that in a minute.

Anyway, so they said: "Gizmodo has learned that tens of thousands of apps, many which may already be installed on federal employees' work phones, use code that sends data to TikTok. Some 28,251 apps" - and it's unclear which platforms those are on, there's no mention of that here - "use TikTok's software development kits (SDKs)," which they explain are "tools which integrate apps with TikTok's systems, and send TikTok user data, for functions like ads within TikTok, logging in, and sharing videos from the app." That's according to a search conducted by Gizmodo and corroborated by AppFigures, an analytics company.

"But apps aren't TikTok's only source of data. There are TikTok trackers spread across even more websites. The type of data sharing TikTok is doing is just as common on other parts of the Internet. The apps using the TikTok SDK include popular games like Mobile Legends: Bang Bang, Trivia Crack, and Fruit Ninja, photo editors like VSCO and Canva, lesser-known dating apps, weather apps, WiFi utilities, and a wide variety of other apps in nearly every category. The developers for the apps listed above did not immediately respond to a request for comment."

Anyway, Gizmodo article goes on, but we got the gist of this. Okay. So there's two parts to this whole mess. The first is, what are we actually trying to do here; and the second is, is it even possible? Okay. So first, to give the whole idea of banning TikTok some perspective, as well as a bit of an updated reality check about the present nature of consumer tracking on the Internet overall, the non-partisan Brookings Institute titled their commentary from the middle of just last month "TikTok bans won't guarantee consumer safety." And so in this case I've grabbed just the end of that long piece, where they conclude and summarize a bit.

"Are TikTok's data practices different from other companies?" And they say: "Several experts have already argued that TikTok bans won't make Americans safer. One reason is that much of the information collected by TikTok is like that compiled by many companies that host consumer-facing products. The app undoubtedly has information on which videos users have watched, comments they've made about those items, and their geolocation while watching the videos, as well as both users' and their friends' contact

information. But that is true for nearly all digital platforms and ecommerce sites around the world.

"It is also the case that digital firms compile data on users, and many buy and sell consumer data via third-party vehicles. It's been estimated that leading U.S. data brokers have up to 1,500 pieces of information on the typical American, and that both domestic and foreign entities can purchase detailed profiles on nearly anyone with an online presence. Even with aggregated data, it is possible to identify specific individuals through a relatively small number of attributes, with some research estimating that 99.98% of Americans could be de-anonymized from relatively small datasets. Still, what sets TikTok apart are the amount and type of trackers they use.

"According to a 2022 study utilizing Apple's 'Record App Activity' feature, TikTok utilizes over twice the average number of potential trackers for social media platforms. Almost all of these trackers were maintained by third parties, making it harder to know what TikTok is doing with the information they collect. If concerns about TikTok are around the compromising of personal information with government authorities, either in China or elsewhere, there are many firms both within the U.S. and abroad that have been accused of the same. For example, a former Twitter employee was convicted of acting as a foreign agent for Saudi Arabia, providing confidential information from that platform about dissidents to foreign officials. Consumer geolocation data are routinely bought around the world by data brokers and repackaged for sale to advertisers, governments, and businesses.

"Regarding concerns that Chinese companies operating within the U.S. are beholden to Chinese laws, the same can be said of American companies that operate in China. Some observers have expressed worries about Tesla vehicles being made in China for some of the same reasons, and what the company may have to do to maintain good relations with Chinese officials. Furthermore, if the criterion for bans based on national security is access to users' confidential information, there's a long list of American and foreign companies that face security challenges via their Chinese operations. As examples, many digital products sold domestically are made in China. And a wide variety of smart appliances, pharmaceuticals, personal protective equipment, computer chips, and other products are assembled there."

Okay. So what's actually developed over time globally is a rich and deeply interdependent ecosystem. And there are myriad companies collecting and selling data on everyone who's using the Internet. Our illusion of true anonymity is exactly that, an illusion. As third-party cookies once did, most of this operates under the radar. While unseen, it is still utterly ubiquitous. It's everywhere.

Okay. So that leaves the question, assuming that the governments of the world decide that they're going to blacklist TikTok, is that even possible? As we know, IP addresses are readily changed, but the domains used by DNS lookups are generally hardcoded into apps and trackers. That means that DNS lookups are TikTok's Achilles heel. I did a bit of research and identified five domain name roots which often also have subdomains. So some wildcard matching would be necessary. There are also two Akamai CDN domains. But taken together, those would appear to be all of the domains currently in use by TikTok. So they are tiktok.com, tiktok.org, tiktokv.com, tiktokcdn.com, musical.ly, and then the two Akamai CDN networks.

Okay. So if federal agencies were to locally configure their networks' local DNS to blackhole those domains and their subdomains, you know, perhaps returning 0.0.0.0 or 127.0.0.1, or maybe pointing them to a local server so, you know, to have them resolve to something. Once local device caches expire, you know, the DNS caches, and a quick DNS DIG that I did indicates that those domains are running with quite short TTL (time-to-live) expirations, all traffic of any sort bound for TikTok would lose its destination IP

and would be blocked at the border. It would just get dropped at the client device. They would no longer be able to get the IP for any TikTok property.

On the question of whether this would be a good thing to do, I have no opinion. Whereas the technology is interesting, the politics is not. Tensions are clearly on the rise with China, so I suppose that nationalism and protectionism are bound to rise, as well. But I think, more than anything, the technology lesson we take from this is that there is an incredible unseen and largely unappreciated underground of activity that very few Internet users appreciate. Out of curiosity, I went over to MSNBC's website at msnbc.com and uBlock Origin lit up, counting that single website homepage causing my browser to connect to 38 other domains; and foxnews.com connected my browser to 51 other domains. 38 and 51. I'm sure that few are directly affiliated with either property, and how many CDNs do they need? All of that other crap is superfluous. It's, you know, who knows what? And since most people have no idea what's going on, why not load up with revenue-generating trackers?

I doubt that TikTok cares at all about the loss of connectivity to federal government networks. And federal employees who want to continue to use TikTok on their own devices while within those networks can simply switch to their cellular provider for continued unfettered Internet access. If the United States government's actual goal is to protect its citizens from the data collection of a Chinese state-owned and controlled entity, then blocking all TikTok traffic at U.S. borders is going to be necessary. But every time Russia or some other repressive regime does the same, we make fun of them. So, you know, make up your own mind.

Leo: Yeah, I mean, do we want a Great Firewall of the U.S.?

Steve: Exactly.

Leo: I think some people would like that.

Steve: Exactly.

Leo: Yeah.

Steve: It seems crazy, and I think that this just seems like way overblown. I thought it was interesting that Brookings said that in that 2022 analysis, TikTok was running more tracking things.

Leo: Yeah, that is interesting.

Steve: More than twice.

Leo: Yeah.

Steve: Yeah, that's interesting. But on the other hand, it's twice more than a whole lot. So it's a lot lot.

Leo: More than Facebook, really? More than Instagram? I find that hard to believe.

Steve: I agree.

Leo: And I have to say, these same people who are shouting about shutting down TikTok have done very little to protect our privacy with telecommunications companies.

Steve: Right.

Leo: Who lobby very hard and give them a lot of money. And they just - they're mum on that.

Steve: Right. And to your point, Leo, it was a really good one, when you mentioned Facebook and Instagram, the number of trackers is one metric. But the number of tracking events is a different metric. And because of the heavy use of Facebook and Instagram...

Leo: Yes, good point, yeah.

Steve: ...they're getting many more tracking events.

Leo: But we know that because they kill your phone battery. It's the first thing you do if you want to save your phone is take Facebook and Instagram off.

Steve: Yup.

Leo: I don't think TikTok kills my phone battery. But I don't know, maybe it does.

Steve: Let's take a break.

Leo: Okay.

Steve: I'm going to get ready for the next phase of this.

Leo: Get ready for the next phase of your life, Steve Gibson.

Steve: So this piece piqued my interest because of the recent discussions we've been having about the U.K.'s decision to mandate the equivalent of some sort of backdoor in otherwise secure and private communications. A report from the Office of the Inspector General for the Department of Homeland Security was titled "Secret Service and ICE Did

Not Always Adhere to Statute and Policies Governing Use of Cell-Site Simulators." And it was followed "Law Enforcement Sensitive (REDACTED)."

That report found that the U.S. Secret Service and the U.S. Immigration and Customs Enforcement (ICE) have not obtained court orders for multiple operations in 2020 and 2021 where they deployed cell-site simulators - you know, stingrays - to intercept mobile communications. The report found that one Secret Service field office had deployed stingrays on multiple occasions on behalf of a local law enforcement agency without obtaining court warrants. The report also found that ICE "did not believe court authorization was required" for some of its operations. Furthermore, the report also found that neither the USSS (U.S. Secret Service) nor the ICE were documenting operations related to supervisory approval and data deletion procedures.

So, yeah, we need to be careful with the technology that we make generally available. And those who have the technology need to know that their use of it will be monitored, and they will be held accountable. This demonstrates obviously the potential for abuse once loopholes are placed into our supposedly secure and private communications. We'd like to believe that only those holding valid court orders, you know, search warrants, would have access to private communications. But experience suggests otherwise. Everybody's only human; right? Except ChatGPT, and we're not sure what it is.

Okay. Strategic Objective 3.3. This is a biggie. And in fact I'll have something more to say about this next week. I found a recent speech that was given. But dated March 2023, so this month, last week the Biden administration published its 39-page National Cybersecurity Strategy. I haven't had time to go through the entire document; but if it appears worthwhile, I'll likely cover it in additional detail next week. And in fact I did see some stuff about IoT that I want to talk about, too. But one section of the document in particular was brought to my attention by Mark Fishburn, a listener of this podcast who knew I'd find it interesting. And our listeners will know why when I share it.

That section is "Strategic Objective 3.3" labeled "Shift liability for insecure software and services." Get a load of what is now part of the United States official national cybersecurity strategy. It says: "Markets impose inadequate costs on, and often reward, those entities that introduce vulnerable products or services into our digital ecosystem. Too many vendors ignore best practices for software development, ship products with insecure default configurations or known vulnerabilities, and integrate third-party software of unvetted or unknown provenance. Software makers are able to leverage their market position to fully disclaim liability by contract, further reducing their incentive to follow secure-by-design principles or perform pre-release testing. Poor software security greatly increases systemic risk across the digital ecosystem and leaves American citizens bearing the ultimate cost.

"We must begin to shift liability onto those entities that fail to take reasonable precautions to secure their software while recognizing that even the most advanced software security programs cannot prevent all vulnerabilities. Companies that make software must have the freedom to innovate, but they must also be held liable when they fail to live up to the duty of care they owe consumers, businesses, or critical infrastructure providers. Responsibility must be placed on the stakeholders most capable of taking action to prevent bad outcomes, not on the end-users that often bear the consequences of insecure software, nor on the open source developer of a component that is integrated into a commercial product. Doing so will drive the market to produce safer products and services while preserving innovation and the ability of startups and other small and medium-size businesses to compete against market leaders.

"The administration will work with Congress and the private sector to develop legislation establishing liability for software products and services. Any such legislation should prevent manufacturers and software publishers with market power from fully disclaiming

liability by contract, and establish higher standards of care for software in specific high-risk scenarios. To begin to shape standards of care for secure software development, the administration will drive the development of an adaptable safe harbor framework to shield from liability companies that securely develop and maintain their software products and services. This safe harbor will draw from current best practices for secure software development, such as the NIST Secure Software Development Framework. It must also evolve over time, incorporating new tools for secure software development, software transparency, and vulnerability discovery.

"And finally, to further incentivize the adoption of secure software development practices, the administration will encourage coordinated vulnerability disclosure across all technology types and sectors; promote the further development of software bills of material; and develop a process for identifying and mitigating the risk presented by unsupported software that is widely used or supports critical infrastructure. In partnership with the private sector and the open source software community, the federal government will also continue to invest in the development of secure software, including memory-safe languages and software development techniques, frameworks, and testing tools."

Wow. Okay, now, obviously no strategy is law. Right? A strategy is only that. And major software publishers have strong lobbying arms in Washington where legislative votes are available to the highest bidder. So nothing here in this 3.3 section is actionable, and there will be a great deal of pushback against any sort of weakening of today's current blanket contractual protections, which we've often noted is like, well, yeah, you could use the software, but whatever it does, it does, and we don't really know, and if you don't like it, all you can ever do is ask for your money back. That's your maximum recourse.

What caught me off guard here was the precision of understanding about the nature of this problem. You know, that one sentence: "Software makers are able to leverage their market position to fully disclaim liability by contract." In other words, if you don't like it, don't use it. On the other hand, your position doesn't make not using it a practical alternative; right? You have to use Microsoft stuff if you're in the enterprise. So anyway, they said: "Further reducing their incentive to follow secure-by-design principles or perform pre-release testing."

So I would say the writing is not yet even on the wall, but it's obviously in some people's heads. And it just got written down in the official national cybersecurity strategy for the first time ever. So it's obvious that others have noticed the same irresponsible attitudes toward critical software security that we've been discussing here on this podcast. I mean, you know, the fiasco of Microsoft's printer problems which they just ignored for six months, which hurt huge numbers of their customers, was unconscionable.

Leo: So they'd be liable for this.

Steve: Yes.

Leo: Which I think is fantastic.

Steve: Yes.

Leo: I mean, they're not going to think it's very fantastic, I'm sure.

Steve: Oh, no, baby.

Leo: Does it mean liable criminally, or liable civilly?

Steve: I'm sure it's civil liability.

Leo: Yeah. And maybe just opens up the idea that if you were a victim of a printer hack, that you could recover losses from Microsoft. Of course, they're shrink wrap licenses, and all those EULAs and all that say, oh, we're not liable; and, oh, you have to go to arb...

Steve: Well, and that's just it.

Leo: Yeah.

Steve: That is, they're disclaiming liability by contract, which this specifically targets, and says this is not okay.

Leo: Right. Good. We talked about it on TWiT a little bit, and I wanted to get your take on it because that seems a big sea change. And you're right. Will Congress let that go through? Probably not. But it's good to just shake some trees a little bit, I guess.

Steve: Well, and ask if the public would not be for this. So, you know, it's like anybody who hears this is like, well, yeah.

Leo: Of course.

Steve: Why wouldn't they be responsible? Of course. And as I've been saying, it's insane, what we have now, where there's zero accountability.

Leo: And we've all seen those, you know, disclaimers on boxes of software and so forth that say, you know, not representing that this is fit for any purpose whatsoever.

Steve: Merchantability or fitness of use. Yeah, we don't know...

Leo: We don't guarantee nothin'.

Steve: You know, you may drive the car off the lot and the wheels all fall off. Well, you know, we thought we screwed them on tight, but I guess Henry did not do that.

Leo: But actually, that's the difference because car manufacturers are liable.

Steve: That's my point.

Leo: Software manufacturers are not.

Steve: This is an anomaly for something that has become as important as software has.

Leo: I agree.

Steve: And it's only because of history; right? Back when it was just like, well, you could store your wife's recipes on your Apple II. And well, if we lost them, we're sorry, you should have held onto the paper copy. It just - and it went from there; right? It never changed.

Leo: Good. I love it. I hope they get this happening. This is good.

Steve: This is big news. And again, just the fact that it's been written down, and now people are going to see it and go, huh. Yeah. Why is that that way? You know? It's like, okay, good.

So back in December, GitHub announced the public beta of their free secret scanning - the name of this is weird. Secret scanning alerts. I had to, like, what? Anyway, public beta in December of their free secret scanning alerts across public repositories. Now, by "secret scanning" they mean that GitHub will proactively scan all code submitted for the inadvertent inclusion of any secrets like, whoops, we left the admin password in the code by mistake. It's a very cool idea. And since its initial release in beta, more than 70,000 public repositories have turned on secret scanning alerts and have uncovered thousands of leaking secrets, stuff that the authors did not intend to leave in their code.

So as of one week ago, last Tuesday, a week ago today, last Tuesday, GitHub's secret scanning alert experience is generally available and free for all public repositories. GitHub users can enable secret scanning alerts across all of the repositories they own to notify them of any leaked secrets across their entire repository history including code, issues, descriptions, and comments. GitHub secret scanning works with more than 100 service providers in the GitHub Partner Program. In addition to alerting users, they will notify their partners when one of their secrets is leaked. With secret scanning alerts enabled, regular users will now also receive alerts for secrets where it's not possible to notify a partner, for example, if self-hosted keys are exposed, along with a full audit log of actions taken on the alert.

So one example of this in practice is a DevOps consultant and trainer whose handle is @rajbos, I guess his name is Rob. Anyway, he enabled secret scanning on approximately 14,000 repositories and discovered over 1,000 secrets. Rob remarked: "My research proves the point of why everyone should have secret scanning enabled. I have researched 14,000 public GitHub Action repositories and found over 1,000 secrets in them."

Leo: Oh, my god.

Steve: Yeah. "Even though I train a lot of folks on using GitHub Advanced Security, I found secrets in my own repositories through this."

Leo: It's easy. It's easy to make that mistake. It really is.

Steve: Yeah.

Leo: I have to be very careful because I have - and I bet you a lot of these are this. I have, well, some of it's going to be code with like your Amazon secret key, API key in it; right?

Steve: Right, right.

Leo: But I also, I back up my dotfiles, my settings files. My PGP stuff's in there. I back up my Emacs setup. And it's very easy to, you know, often people hardcode secrets into those files. And then you're thinking of the GitHub, it's easy to forget and miss it.

Steve: Yup. And, for example, you might have put that in just for, like, during testing; right?

Leo: Yeah.

Steve: So you didn't have to keep entering the password every time. You just put a little shortcut in. And then when you go to production you forget.

Leo: Forget.

Steve: And it's like, whoops.

Leo: There's a way to do it, you know, they're really easy - not that easy. But there is a little thing you could do to have PGP encrypted keys in all those secrets. And then they're kind of safe. They're still in the code, but they're safe. And I've always thought, oh, I should probably do that. And I have never done it. So, you know, I'm counting on my own brilliance to remember not to upload those files. Oh, boy. Good luck.

Steve: So also last Tuesday, CISA revealed the somewhat bracing results of a secret red team exercise that they carried out against the network of a "large unnamed U.S. critical infrastructure organization with a mature cyber posture." During that exercise they "obtained persistent access to the organization's network, moved laterally across multiple geographically separate sites, and gained access to systems adjacent to the organization's sensitive business systems." CISA says that on at least 13 separate occasions its red team triggered "measurable events." It wasn't clear whether this was

deliberate or unavoidable, but that should have gotten it caught, and didn't. In every case the organization failed to detect these actionable events.

As I noted, CISA officials declined to name the organization. But they did say that, although they did manage to get in, they found the organization had good cybersecurity policies in other parts of its network, such as up-to-date and hardened perimeter infrastructure and good password policies. Okay. So how did CISA's red team gain its initial entry? Phishing, today's most difficult to corral cyber weakness. People on the inside are on the inside, and it's just too easy for someone to inadvertently allow a bad guy in. And that's what happened in this instance. And once they got in, they stayed in, moved laterally, set up shop, moved around, explored the network, all unseen.

Okay. So I titled this little short bit of news "What's Left?" after reading that Russia had formally legally banned the use of all foreign messaging applications inside Russian financial institutions and state-owned companies. Again, this is one of those, what, only now? Like why would you not do it a long - it's like, are you still using Windows in Russia? What?

Anyway, the law, which entered into its effect this month, outlaws the use of apps such as Discord, Microsoft Teams, Skype for Business, Snapchat, Telegram, Threema, Viber, WhatsApp, and WeChat. Now, the page announcing this was in Russian, and it lists those. And if you're interested I have the link here in the show notes, like we have any Russian speakers. Notably missing is Apple's iMessage and Android's Messages, which are both end-to-end encrypted. You can see the name of the things that I named. There they are. Nine specifically enumerated messaging apps.

So this legislation, which specifically enumerated those nine third-party messenger apps doesn't mention the two mobile platforms' native messengers. So perhaps that's the answer to my rhetorical question, "What's left?" Or perhaps those are just implied. The news was that all foreign messaging applications were banned. So perhaps Russia has some state-owned or trusted messaging apps that no one is going to want to use or trust? I don't know. Anyway, just sort of interesting. That's like, okay, can't use these. But we don't know about iMessage, or Messages on the Android platform.

As we've noted before, CISA is maintaining a list of what they call the KEV list, the Known Exploited Vulnerabilities. And KEV has become a common abbreviation within the security industry for that growing list. In other words, these are vulnerabilities which CISA has had reports of actually being used in the field. So that's like the patch first list; right? And once upon a time it wasn't very big. But it's been growing. During the last year, the size of this catalog of known exploited vulnerabilities very nearly tripled in size. It jumped from 331 entries at the start of 2022, but finished out last year with 868 individual vulnerabilities known to have been seen being actively exploited.

Okay, now, here's the interesting part. Although this near tripling of the list in one year would suggest that new bugs are being exploited, a look at the issuance dates of the catalog's CVEs for those shows that the vast majority of the new CISA KEV entries are for older vulnerabilities that companies failed to patch for years, all which came under attack last year. The oldest of those was the exploitation of a bug that originally had a patch available for it 21 years ago, back in 2002, which was it came back to life and was being used for vulnerability exploitation. So, wow.

As I said, you know, we've often talked about how amazingly difficult it is to get the old stuff patched. And that's another advantage of this directive 3.3; right? If the software were more secure out of the gate, rather than relying, I mean, like relying by policy, which is what Microsoft has become; right? They are relying by policy on updates. There's not even a - it's not a shock. It's hundreds of security vulnerabilities per month are being fixed. So, and we hear stories, right, of how what is it, Windows 7 or 8 shipped

with over 10,000 known problems? And, yeah, some of them are that the color of this gray shading changes if you shake the computer sideways during a full moon. They're just not big problems. But obviously a lot of them are big problems.

And so what's happened is it's just become, oh, yeah, what day is it? Okay, ship it. We have weekly updates. We'll fix it later. Well, not all systems have weekly updates, or monthly updates I should have said. So it would be nice if there was a little more pressure to get it right the first time, since we seem as an industry, as a world, we're just not keeping things patched, even when fixes are available.

Speaking of patches, remember when everyone used Intel's sample Universal Plug and Play implementation code as their actual production code, apparently without ever actually looking at the code, and even in the face of that comment header block at the top of the code which loudly stated that this was sample code only and should never be used for production? And that as a result of that, the entire industry suffered as a whole from a widespread vulnerability in that sample code which everything had in their routers.

Okay, well, history is - it's not really repeating, but it's certainly reminiscent. This time it's a pair of widespread vulnerabilities which have befallen multiple vendors' Trusted Platform Module, the TPM code. It's not, as I said, not quite the same as the UPnP debacle, since these vendors were at least using the TPM Reference implementation library, which should have been okay. But this is another example, at least, of the danger of monocultures; and why we're more healthy if, as another example, we keep browsers other than Chromium alive.

In this case, researchers at Quarkslab discovered a pair of buffer overflow vulnerabilities in libraries implementing the TPM 2.0 security specification. The vulnerabilities would allow an attacker who could gain access to the TPM's command-line interface to leverage the vulnerabilities to corrupt the TPM's memory and access sensitive information handled by the TPM, such as encryption keys - which of course is exactly what all of the TPM's fancy hardware technology is supposed to prevent. Patches for this were released at the end of last month, end of February. And since many vendors all directly implemented the same reference library, the TPM implementations from IBM, the Trusted Computing Group themselves, Red Hat, SUSE Linux, and many others are all affected. So we can look for some updates to that coming soon.

Okay. Two last pieces. WordPress. Just a quick note to all of our listeners who manage WordPress sites. More than three million WordPress sites which are currently running a plugin called All in One SEO will need to be updated to resolve a set of vulnerabilities that could be used to hijack sites. As is often the case, the troubles were found and reported by the researchers at WordFence. And again, three million-plus WordPress sites currently vulnerable.

As I've noted before, when we were talking about the inherent danger of third-party-developed WordPress add-ins, which appear to be having constant security problems, and again, not because anybody's malicious typically, just because there are unprofessional developers who create a widget and think, hey, I'm going to make this available to other people. Anyway, adding protection from WordFence, if it fits your budget, would seem like a large and worthwhile ounce of prevention. So just a reminder that those guys are there, and they really are on the ball.

And finally, the Russian government has fined the Wikimedia Foundation, the organization behind the Wikipedia portal, two million, okay, well, rubles - that's about $27,000 - for failing to delete "misinformation," Leo, about the Russian military and its invasion, oops, its special operation in Ukraine. According to Reuters, this is the third time Wikipedia has been fined by Russia since the country's invasion of Ukraine.

Wikipedia said the recent fine was related to articles on its Russian language portal related to Russian Invasions of Ukraine, the Battle for Kyiv, War Crimes during the Russian Invasion of Ukraine, the Shelling of Mariupol Hospital, the Bombing of the Mariupol Theater, and the Massacre in Bucha.

I noted that last November, when the second of the three fines was levied, the same fine of two million rubles was set. But back then those two million rubles were worth $33,000 dollars; today, $27,000. So those rubles appear to be slipping against the dollar a little bit.

The Wikimedia Foundation has stated that they refuse to back down and remove what it said was clearly fact-based, multiply-sourced, verified truth. And they've been appealing these fines in Russian court. So far they've only had one successful ruling. But, you know, we can hope. Leo?

Leo: I mean, I wouldn't expect much from the courts, to be honest with you.

Steve: No. I'm surprised they even, like bother.

Leo: Bother, yeah. I mean, who runs those courts? Let's think about that; right?

Steve: Yeah. And, you know, I would imagine if they, you know, they should not pull the content down. If Russia wants to block all of Wikipedia, fine.

Leo: Let 'em.

Steve: Just another loss for Russia's citizens.

Leo: Right.

Steve: Who are part of this mess.

Leo: Sad. It's very sad.

Steve: Okay. Our last break, and then, oh, baby. We're going to discuss a fowl incident.

Leo: I can't wait. I was trying to decide where to go to dinner tonight, and I'm thinking now Popeye's. But we'll find out. We'll find out. And now I can't wait to hear all about this fowl story.

Steve: We are going to encounter a mystery and solve it.

Leo: Ah.

Steve: When I read from a Chick-fil-A data breach report submitted to the U.S. State of Maine's Attorney General, which disclosed that 71,473 Chick-fil-A account-holding customers had had their accounts breached through a credential stuffing attack, I was skeptical. And I'm at least still a bit confused. That number just seems far too huge to be the result of what amounts to opportunistic, previous breach-driven guessing of account usernames and passwords. And really, okay, think about it. Why would some random hacker be going out of their way to compromise the accounts - and more than just a few - of Chick-fil-A customers? Why not Chase, Bank of America, or TD Ameritrade? I mean, Chick-fil-A, really? Those are the customer accounts that you choose to penetrate? You must really have a thing for chicken.

So I don't know. It doesn't make any sense to me. That number, first of all, seems too big 71,473 individual Chick-fil-A customer accounts, each which would have taken effort to compromise. And what do you get for all that effort? I don't know. Apparently some Chick-fil-A redeemable loyalty reward points.

Okay. We've been talking about this form of attack recently. But just to reiterate, since this is where details matter, credential stuffing attacks are the reuse of username, email, and passwords, leaked from previous online site breaches, which are then being used to blindly guess login credentials at other unaffiliated websites. The point is, as I've said before, all rational logic suggests that this should be an extremely low-yield attack, meaning that in order to correctly guess the logins for 71,473 individual Chick-fil-A customers, it would be necessary to wrongly guess a gazillion other times.

When I initially saw that large number, my first thought was that it couldn't actually be a true credential stuffing attack. And for the record, I've never been a fan of the term "credential stuffing." The industry, I think, could have come up with a better name, like "credential reuse attack." But credential stuffing it is. And as long as we all know that it's a credential reuse attack, that's fine.

So for that many accounts to be successfully attacked, I'm suspicious of whether Chick-fil-A might have earlier lost control of their own customer account data, and that the leaked information itself was now being used to login and attack their customers. This would convert the attack from "surprisingly successful low yield against a bizarre target" to "unsurprisingly high yield against the only available target," if it was Chick-fil-A's data that had been breached, you know, leaked. If you've somehow acquired Chick-fil-A's customer logon data, then that's going to be your only target.

On the other hand, if we assume that this was actually a true credential stuffing attack, and putting aside for the moment the question, out of a universe of equally suitable targets, why would an attacker choose Chick-fil-A, the Chick-fil-A's disclosure to various states' attorneys general...

> **Leo:** I wouldn't even choose them for a chicken sandwich, let alone a credential stuffing attack.

Steve: No, no, no. And you'll get a kick at what I put down at the very end of the show notes. We'll get there here in a minute. The Chick-fil-A's disclosure to various states' attorneys general did state that the attack took place over a two-month span. So this is Chick-fil-A's disclosure; right? So Chick-fil-A is saying the attack took place over a two-month span from last December 18th through February 12th, last month. Okay. So that's 56 days during which 71,473 Chick-fil-A customer accounts were breached, at an average rate of 1,276 successful account breaches per day. So the logistics of such a credential stuffing attack would be that an attacker has a massive database of

prospective login credentials which they, for some reason, choose to aim at Chick-fil-A's website.

Leo: I mean, first of all, who has an account with Chick-fil-A?

Steve: Well, yeah.

Leo: You'd have to be quite a fan.

Steve: I agree. And in blind account credential guessing, they then pour this massive database through the website's clearly unrestricted authentication frontend at presumably some massive rate. Perhaps the attack was distributed with the massive database spread among many attacking clients in order to increase the overall rate of credential guessing because of course modern websites are able to simultaneously entertain many incoming connections, but in a situation where there's no apparent oversight over failed authentication, which must have been happening with like millions of instances, monitoring or throttling a failed authentication of any kind of missing.

So there's no real need to distribute the attack. Nothing prevents a single attacking machine, or only a handful, from each establishing their own hundreds or thousands of simultaneous login sessions with a single Chick-fil-A server. That works, too. If we accept Chick-fil-A's claim that this was truly blind credential guessing from a database of previous completely unaffiliated websites, then the per-guess yield had to be quite low. Again, how many people are there in this leaked database that also happen to be Chick-fil-A customers?

Leo: Oh, I'm looking at Chick-fil-A's website now because I'm really curious why would you have a login. And they have something called Chick-fil-A One, which is a point, you know, a customer loyalty thing. You earn points, and then you get a free waffle fry or something.

Steve: Yeah, you know, every X number of coffees I used to buy at Starbucks, I would earn stars.

Leo: But again, why would you want to hack it? What are you going to get, somebody's Chick-fil-A points?

Steve: Exactly. Exactly.

Leo: Maybe their credit card. I mean...

Steve: It had to be, well, and that wasn't available. It was blinded so it could not be seen.

Leo: Okay, okay.

Steve: So the yield per guess had to be quite low. A very low per-guess yield meant that the total number of guesses had to be massive. So this in turn means that the Chick-fil-A website servers raised no alarm of any kind, while starting last December the 18th their incoming connection rate had to have skyrocketed as millions of attempts to log in there were now failing. The authentication failure rate had to be astronomical, yet nothing at their end took any notice.

Leo: So this is a crime of opportunity because most sites you wouldn't even be able to do that. Maybe rate limited, or an alarm would be raised. Maybe Chick-fil-A just didn't have any protection; right?

Steve: Clearly they didn't, or it wouldn't have...

Leo: They weren't paying attention.

Steve: ...gone 56 days without them noticing.

Leo: Yeah.

Steve: Once Chick-fil-A somehow became aware of the attack, presumably when a sufficient number of their own customers complained of some sort of account tampering, like where did my chicken bone points go or whatever the hell it is, you know, they were able to identify exactly, you know, because that's what you want is they want the chicken bone points.

Leo: I want those chicken bone points, mm, mm, mm.

Steve: That's right. They were able to identify exactly which of their customers had their accounts breached in this manner. Remember, we have an exact count, right, 71,463. We're never going to forget that.

Leo: In a way, this is Chick-fil-A's way of saying we just never thought anybody would want to get in.

Steve: That badly, yeah.

Leo: We didn't secure it because we thought, well, who would care if you were - oh, my gosh.

Steve: I know. Okay. Okay. So the reports now that they've repaired any damage done to those customers, they've restored their previous chicken bone points, made them whole again, and instructed them to change their Chick-fil-A passwords, and also anywhere else that they were reusing the same password.

Okay. Now we know what happened. We have two takeaways from this fowl incident. First, if we accept on its face that this was an unaffiliated, if somewhat bizarre attack, then it could only have succeeded as it has due to the continuing presence of a widespread and stubborn reuse of user passwords across sites.

Leo: Although we know what happens. We really do.

Steve: And we do, unfortunately. And of course that - and remember, that's not yesterday, that's today, right now. This was just happening. This means that, more than ever, it is rapidly becoming truly imperative for the uniqueness of passwords to be enforced across all of a user's online accounts. We've learned from our own podcast audience's reports, in the wake of the LastPass vault debacle, that updating passwords can be a slow, tedious, and laborious process. But now, more than ever before, if your password manager offers a global password reuse audit, as many do, maybe also a password strength audit, it's important that you allocate some time to begin replacing any duplicated passwords, and also in strengthening any existing passwords that do not contain sufficient state-of-the-art entropy.

I'm sure that this message is largely redundant and is unnecessary for this podcast's audience. But the success of this Chick-fil-A attack informs us that everyone listening needs to share their understanding of this and why this is important with everyone they know.

Leo: Yes, adjacent, podcast-adjacent listeners, yes, yes.

Steve: Yes. Because if we're to believe Chick-fil-A, it's clear that the reuse of passwords remains widespread across the Internet, and we're starting to see an evolving epidemic of this new form of surprisingly successful attack.

This brings us to the second and more interesting technology question: "What can websites do in the face of what appears to be an escalating and approaching epidemic of opportunistic low-yield credential reuse attacks?" When a user logs into a website, their browser requests the login page, which presents a form to be filled out. Okay, now, I got to this point in preparing today's show notes, when I thought that I ought to go over to Chick-fil-A's website to see whether they presented the login as a single form or multiple staged forms. And what did I find?

I have a picture of what I found in the show notes. You can go to Chick-fil-A and click "Login," and you'll see it, too. There are a pair of standard login credentials prompting for email address and password. But there's also a "Sign in with Google" and a "Sign in with Apple." And in the spot where there was once a "Sign in with Facebook," the page now reads "Looking for Facebook Login?" Isn't that interesting.

Leo: Hmm.

Steve: Clicking that link takes existing Chick-fil-A customers to a page that says, it screams in big red boldface: "Facebook login is no longer available." Huh. It says: "To continue using your account, you will need to set up an email and password login method." In other words, old school. "Please enter the email address associated with your Facebook account to get started and find your Chick-fil-A One account." And then they prompt for your Facebook email. But they're not letting you log in with your

Facebook account any longer. Though we cannot know for sure when Chick-fil-A's probably most popular "Login with Facebook" OAuth2 option was removed, anyone doing some forensic post-incident digging would be skeptical of the coincidence given the recent attack.

Unfortunately, since their login page is algorithmically generated and thus cannot be brought up with a specific, with a static URL, the Internet archive's Wayback Machine cannot be queried to see when "Login with Facebook" was removed as an option. But given that "Login with Google" and "Login with Apple" are both still present, and that Chick-fil-A's replacement for "Login with Facebook" is migrating users from Facebook login to their native login, I would bet a month's pay that now we know what actually happened.

It was not a generic credential stuffing attack. It was specifically a Facebook credential reuse attack. And now we understand why Chick-fil-A was the target. It was because they offered the popular "Login with Facebook" option. Somebody, somewhere, has a boatload of in-the-clear Facebook login username and password credentials. And over the course of several months they explored the intersection of that stolen Facebook credential set with the Chick-fil-A customer account database.

Leo: Which is pretty close to 1:1, I'm betting. I don't know, but I'm just thinking. But you'd have to log into - so you'd have to have a Facebook cookie on your browser; right?

Steve: No. No, no, no. No.

Leo: Because you'd log in with Facebook, and that'd do it.

Steve: Exactly. You could log in with Facebook. Then you provide your Facebook credentials, and you're logged into Chick-fil-A. There were 71,473 Chick-fil-A customers whose Facebook credentials are part of the stolen Facebook dataset, and as a consequence their Chick-fil-A accounts were breached.

Now, when you think about it, if you had a trove of valid Facebook login credentials, what's the most valuable thing you could do with them? Who wants to log into those random users' Facebook accounts? There's no money in that. No. What you want to do is leverage the increasingly pervasive use of OAuth2 account login to compromise the myriad other accounts belonging to those hapless Facebook users who have chosen to identify themselves to other website properties only through their Facebook credentials.

Leo: So you're testing it. You're not trying to get chicken points.

Steve: No, no, no. In fact, the stolen accounts were for sale on the dark web. People wanted chicken bone points.

Leo: Oh, okay. All right, okay.

Steve: So they were actually compromising those accounts...

Leo: So that was intentional, just - boy.

Steve: ...[crosstalk] that were worth something.

Leo: Aim a little higher, kids. I just...

Steve: But think about it. So again, if you have valid Facebook login credentials, you don't care about logging into people's Facebook accounts.

Leo: No. Who cares about that?

Steve: You want to log into the other accounts those people have elsewhere on the Internet where they've used Facebook to identify themselves.

Leo: Now, probably a lot of these people don't use two-factor, either. Right.

Steve: Correct.

Leo: So all you need is the credentials. You don't...

Steve: Correct. So now, as the attacker, you have a valuable and potentially widespread attack. If this is true, as seems extremely likely given all the evidence, we have a perfect example of why the use of OAuth2 for logging in with a common credential poses a significant threat. What are we loudly telling everyone who will listen about their passwords? We're saying, "Do not use the same username and password to login to multiple websites." Right? Everybody knows that.

But the use of OAuth2, which is now being actively promoted due to its extreme ease of use, is a direct contravention of that advice. It is the explicit reuse of a single set of credentials, Facebook credentials, across a great many website properties. And while there may not be much value to an attacker to use a stolen Facebook credential to log into that Facebook user's account, there might well be significant value in their ability to log in everywhere else that Facebook user used their Facebook credentials to create non-Facebook accounts.

So this begs the question, how are in-the-clear Facebook account credentials harvested? And we quickly find an example. Bloomberg News, Technology and Cybersecurity article from October 7th, 2022, okay, just last October, just two months before we're told the attack on Chick-fil-A began, has the headline "Facebook Is Warning 1 Million Users About Stolen Usernames & Passwords." The article says the company found more than 400 problematic Android and iOS apps. Games and photo editors tricked users into providing credentials.

Okay, now, just grabbing one paragraph from Bloomberg's coverage, they wrote: "A typical scam would unfold, for example, after a user downloaded one of the malicious apps. The app would require a Facebook login to work beyond basic functionality, thus tricking the user into providing their username and password. Users could then, for

example, upload an edited photo directly to their Facebook account. But in the process, they unknowingly compromised their account by giving the author of the app access."

So what the Chick-fil-A attack probably reveals is the new use to which stolen Facebook credentials are now being put. Again, obtaining access to a Facebook user's account is far less profitable than being able to log into any and all of that user's non-Facebook accounts where they may have something of value. Most users, as we know, have not the faintest clue how all of this technology we've given them to use works, no idea whatsoever. And Leo, as The Tech Guy, you know that better than anyone.

Leo: Yeah.

Steve: So when they're asked by a spiffy-neat app they've just downloaded to provide their Facebook login credentials so that the app can link to and synchronize with their Facebook account, they don't know any better.

Leo: Right.

Steve: Why would they?

Leo: Right.

Steve: And most users would not understand that, in the process, thanks to the fact that they have also been using "Login with Facebook" everywhere they possibly can because it's so much easier to do that, that they are also giving away the access to their accounts at all of those other websites, as well.

We've often noted that the use of OAuth2 is an inherent privacy compromise since that third-party - Facebook, Google, Apple, or whomever - knows everywhere you're using them to log in, and where you are at the time. But this Chick-fil-A attack, with its subsequent removal of its most popular "Login with Facebook" option, reveals a much darker side of the widespread use of this form of single sign-on solution. All of the common wisdom urges users to avoid credential reuse, but credential reuse is exactly what single sign-on promotes. And thus this Fowl Incident has highlighted a worrisome truth behind a growing trend.

And I leave you with the end of the show notes Picture of the Week, which I found on the Chick-fil-A site, which is also a little cute. It's got three cows standing up, each holding a sign. The first one says "Eat," the second one says "Mor," and the third one says "Chikin." So, yes.

Leo: Yeah, this is the famous...

Steve: The cows are encouraging you to eat more chicken.

Leo: And less burgers, and more...

Steve: And less beef, yes.

Leo: Yes, more chicken fingers. Which is amazing because chickens don't even have fingers. Well. Well, sir, you have done it again. You have gotten to the bottom of one of the nation's most critical security issues, the incredible Chick-fil-A breach.

Steve: Be sure to dump your chicken bone points out before the next hack.

Leo: You've now explained it to all in a way no one else - this is a scoop, sir. This is a scoop. You've figured it out. You know, just turn on two-factor everywhere. Facebook has it. I don't know why you wouldn't use it. And then no one will steal your chicken bone points. Okay? Okay.

Steve: Or any other bone points anywhere else.

Leo: Keep your bone points to yourself. Don't keep, however, don't keep your podcast points to yourself. Spend them right here. Join us every Tuesday as we do Security Now!. I know you want every copy of every show.