Attacking TOTP

Description: What do we learn from January's record-breaking zero-day critical Patch Tuesday? Microsoft to "force-install" a new Outlook into all Windows 10 and 11 desktops? GoDaddy is required to get much more serious about its hosting security. More age verification enforcement is coming, including globally. What another instance of a widely exposed management interface teaches us. DJI drone's official firmware update lifts geofencing for unrestricted flight. CISA's efforts pay off with MUCH improved critical infrastructure security. Listener feedback about TOTP, HOTP and age-verification. And we take a deep dive into cracking authenticator keys.

High quality (64 kbps) mp3 audio file URL: http://media.GRC.com/sn/SN-1009.mp3
Quarter size (16 kbps) mp3 audio file URL: http://media.GRC.com/sn/sn-1009-lq.mp3

SHOW TEASE: It's time for Security Now!. Steve Gibson is here with a rundown of the, what is it, 160 critical patches Microsoft shipped last week on Patch Tuesday? Microsoft's also forcing you to take Outlook. GoDaddy is going to get much more serious about its hosting security. And then, get ready, get your propeller hats on because there will be math. We're going to brute force your one-time password authenticator. Well, at least we'll talk about how hard or easy it would be to do. It's going to be a fun episode, next on Security Now!.

Leo Laporte: This is Security Now! with Steve Gibson, Episode 1009, recorded Tuesday, January 21st, 2025: Attacking TOTP.

It's time for Security Now!, the show where we talk about security, privacy, protecting yourself and your loved ones on the great big vast Internet with this guy right here, our security in chief.

Steve Gibson: You jumped a little bit when you said "We talk about security." I thought, well, you're surprised? No.

Leo: What? Is this the security show? Oh, my.

Steve: We do like to surprise our listeners every week, one way or the other.

Leo: Yes, yes.

Steve: Give them something to think about. And we're going to do that again this week. Today's topic for Security Now! #1009 - and yes, that's four digits - is "Attacking TOTP." We've talked a lot in the past about brute force attacks, and we understand the concept of that. But I thought it would be fun, and this was another one of those outgrowths from a listener feedback question where he mentioned that, well, I don't want to step on my eventual explanation of this, but it led from a listener feedback question that we will get to, that I think produces a really interesting conversation where we look at, not just like, oh, wave our hands over it and say, oh, yeah, you just try a lot of things. No, let's really look at what it means to brute force something like the authenticator that we're all using in our lives every day. Is it secure enough?

Last week we dug deeply into the protocols, the actual algorithms that this thing is using. So now we have that as a basis. And I thought, okay, this is too good an opportunity to pass up. Let's see what it would take to attack an authenticator, what information do we need from it, how much of that information do we need, and what do we need in terms of processing power and capability. So that's our main topic for the day. But we're going to look at, of course, last week's, that is, which is to say January's record-breaking zero-day critical Patch Tuesday, brought to us by none other than Microsoft.

Also there's some interesting news that I thought was, like, what? I had to pursue it. Microsoft will be force-installing - that's the jargon that everyone is using - force-installing a new version, a new and arguably unwanted version of Outlook into every single Windows 10 and Windows 11 desktop, and there is no way to prevent it. Again, we'll dig into that more. GoDaddy is being required to get much more serious about its hosting security. We know they've had some problems there. We've got more age verification enforcement coming, this time internationally. And what another instance of a widely exposed management interface continues to teach us. Also DJI drones' official firmware update lifted its geofencing, now allowing unrestricted flight. Odd timing.

Leo: Isn't that strange? I thought that was odd, yeah.

Steve: Yeah, really. CISA's efforts pay off with much-improved critical infrastructure security. Let's hope everything continues working for them. And also I've got a bunch of listener feedback, a fun piece of errata, something I completely got wrong that several of our listeners said, what? What are you talking about? And then we've going to take a deep dive into cracking authenticator keys. And of course we have a Picture of the Week that will not disappoint. If you haven't seen it yet, Leo...

Leo: I haven't.

Steve: ...be great to share your reaction live...

Leo: Oh, good.

Steve: ...with our audience.

Leo: I like to scroll up live.

Steve: That's a goodie.

Leo: Very good. It's going to be a good show, as always. I loved last week. It was really fascinating to hear how they came up with a TOTP protocol in such a weird way.

Steve: Well, and it's interesting because when we look at the task of accelerating brute forcing of it, you could take the position that that wacky spin...

Leo: Ah, slowed it down.

Steve: ...makes it more difficult to run a brute force.

Leo: Okay. So maybe that's why they did it.

Steve: It was in 2005. I don't think they were thinking clearly about anything back then. But, you know, maybe.

Leo: We can give them the benefit of the doubt. I don't know. All right. Well, we'll talk about it in just a bit when we get to brute forcing TOTP, that is, as the main subject. But as you can just hear there's a lot more in between there and here. All right, Steve. I have not - I have preserved my virginity. I have not looked at - maybe that's not the way to describe it. I have not looked at the Picture of the Week. But I am now about to scroll up.

Steve: I will tell you first that I gave it the caption "So how exactly do you propose we get up there to fix that?"

Leo: Hmm. Okay. There is a scissor-lift involved. Oh. Wow. Is that real? Holy-moly. So there's a scissor-lift. But this is above a swimming pool.

Steve: Yeah. It looks like an Olympic-size, big, big swimming pool.

Leo: Holy cow.

Steve: And apparently there's something that's gone wrong up in the beams, like in the middle, well, not in the middle, but like over the water of the pool. So this scissor-lift is like, it's up like where they'd be standing on the third-story if it were...

Leo: Oh, yeah, it's high, yeah.

Steve: You know, so it's way extended. Then but the problem where they need to be is over the water. So they found some sort of a float which is a large rectangular float. And, you know, again...

Leo: Could that possibly work?

Steve: And you'll see that they've got yellow ties to the four corners of the float.

Leo: So it doesn't float around.

Steve: Well, so that the scissor-lift itself doesn't tip over and it doesn't roll anywhere. So it's anchored itself to the center of the float and then got pushed out. Now, one question I had was like, okay, how do they position themselves? Maybe they like did a hand-over-hand off the top beam in order to, like...

Leo: They float around?

Steve: Like float around, yeah.

Leo: So many questions. So many questions. That's hysterical, Steve.

Steve: Looks legitimate to me. I mean, you know, it's - it looks real.

Leo: Wow. Wow.

Steve: And again, I guess you could do one of those things with a long arm and park it off to the side of the pool and have the long arm reach out with a guy in a basket as your alternative. But otherwise...

Leo: It's crazy.

Steve: Anyway, regardless...

Leo: That's hysterical.

Steve: ...a fun Picture of the Week.

Leo: Absolutely.

Steve: "How exactly do you propose we get up there to fix that?" Okay, Joe, here's what I suggest.

Leo: And of course Phoenix Warp in our YouTube chat says, "I'm not worried about how they got there. How do they get back?" Wow.

Steve: Oh, yeah. Okay. So Patch Tuesday. CrowdStrike's blog was titled "January 2025 Patch Tuesday: 10 Critical Vulnerabilities and Eight Zero-Days Among 159 CVEs." And we touched on this last week, the fact that this was the highest number of patches that we'd seen from Microsoft in years. Not ever, but quite a while. And, well, which goes to show, as we're always saying, things are not getting any better. No.

The article noted, and it said: "This month's leading risk type by exploitation technique is remote code execution (RCEs) with 36% of them being" - okay, so more than a third are like the worst problem you can have, right, remote code execution, followed by elevation of privilege. Well, that's the second worst type you could possibly have because once you get in you need to be able to get the OS's safeguards out of your way in order to do some real damage, which standard users are largely prevented from doing, just to protect them from themselves.

So CrowdStrike gave us a pie chart which shows around the pie 9% of the problems were security feature bypass. So, okay, whatever that is. That's, you know, sort of a generic catchall. 13% denial of service, meaning you crashed something, and so its service was thereby denied. Then we get a big light green chunk, that's the 25% which is elevation of privilege. We drop down to 14% for information disclosure. And then the biggest of all at 36% is remote code execution, followed by a little 3% sliver for spoofing.

So unfortunately, as we've laid out in the past, of all the vulnerability classes, we know that the two most powerful and desired by the bad guys are remote code execution and elevation of privilege, and of course those were the top two, 36% and 25% respectively. And they don't overlap. Those are, you know, summed. So together that's 61% of all 159 problems were of the most serious kind available. Elevation of privilege, as I said, allows someone who arranges to get into a system as a regular and somewhat constrained user to bypass the operating system's privilege strictures. And remote code execution can both create that initial entry into the system, that is, enable the way of getting in; and then, once your privilege has been elevated, allow the bad guys to run the code of their choice to wreak havoc.

Viewed by product, Windows itself received 132 of the patches. And somewhat chillingly, Microsoft's ESU that's the Extended Security Updates for previous Windows operating systems that no longer receive free patches and must have these fixes for Microsoft's own security flaws purchased, those received 95. And in distant third place was Microsoft Office with a relatively sedate 19 patches. It's interesting that current Windows received 132 patches, whereas older Windows, which Microsoft has stopped fussing with, was down at 95. Which, you know, which Windows would you say is objectively safer to use? Uh-huh.

It's so easy to become numb to the idea that these vulnerabilities are being actively exploited. This means that there are serious - somewhere in the world are serious campaigns that are investing heavily - because, you know, these are not easy to find. Other people would have found them, you know, white hat hackers, people getting paid to find problems would have found them. And by the way, these are old. We'll get to that in a second. But so my point is somewhere, I mean, there is, like, serious industry at work investing in discovering these subtle vulnerabilities and then deploying exploits to take advantage of them in the real world because these are zero-days under active attack.

Windows Hyper-V NT Kernel Integration VSP received three patches, all having a severity of Important and a CVSS of 7.8. The three are elevation of privilege vulnerabilities allowing an attacker to gain system privileges. Microsoft has indicated that the weaknesses are due to heap-based buffer overflow, but has not shared details of the vulnerabilities or how they learned of them, what the source of the disclosure was. Microsoft Office Access received patches for another three, all having the same severity

of Important and the same CVSS score of 7.8. But all three of these, that is Microsoft Access, are remote code execution vulnerabilities exploited by opening specially crafted Microsoft Access documents. Microsoft addressed this attack vector by blocking access to certain types of extensions in addition to patching the vulnerabilities.

So here again we have one of those fundamental problems of unneeded features coming back to bite them well into the past. And we'll talk about the past in a second. There were three critical-rated 9.8 problems, which as we know, it's very difficult to get a 10.0. 10.0 is like, we see that very rarely. But 9.8 is regarded as this is really important, you've got to fix it right now because it's going to happen.

The first was a critical remote code execution vulnerability affecting Windows Reliable Multicast Transport Driver (RMCAST), and that has a CVSS, as I had noted, of 9.8. An unauthenticated attacker, meaning anybody out on the public Internet anywhere, can exploit this vulnerability by sending specially crafted packets to a Windows - I love the name of this - Windows Pragmatic General Multicast, that's the PGM, the Pragmatic General Multicast open socket on a server, without any user interaction.

Leo: Wow.

Steve: Uh-huh. However, exploitation is only possible if a program is actively listening on one of these PGM (Pragmatic General Multicast) ports. The vulnerability is not exploitable if PGM is installed or enabled, but no programs are listening as receivers. Since PGM does not authenticate requests, it's crucial to protect access to any open ports at the network level, such as with a firewall. Gee, you think? It's strongly advised to avoid exposing a PGM receiver to the public Internet due to the security risks. So that's a problem.

Now, I have not dug into this to see how likely it is that a machine might have this port publicly exposed, nor what services might be listening for incoming traffic there. But it's clear from its 9.8 rating, which again, they don't want to give to anything, and that it's a remote code execution exploit, if those conditions were met the result would be, shall we say, not good.

The second of three critical-rated 9.8 RCEs seems much more worrisome, since it affects Windows' old OLE, remember Object Linking and Embedding technology, which allows embedding and linking to other documents and objects from within documents. That was all the rage back in the early days of Windows. In an email attack scenario, which is why this is raising such concern, an attacker could exploit this vulnerability simply by sending a specially crafted email to their victim. Exploitation of this vulnerability might involve either a victim opening the specially crafted email with an affected version of Microsoft Outlook software, but that's not necessary. The Outlook application's displaying of just the preview of the specially crafted email could allow an attacker to remotely execute their own machine on the victim and take it over. So, yikes.

Now, given OLE's age, my guess was that this would have been one of those vulnerabilities that Microsoft would have required payment for fixing on their older, yet still vulnerable machines. And indeed they list Windows Server 2008 and 2012 among the vulnerable systems. Since Server 2008 and 2012 are the equivalent of the desktop Windows 7 and Windows 8, I'd bet that those desktops are vulnerable to this, as well.

Their workaround advice is to - I love this. Okay. So this is bad. What do we do? Their advice, only view your email as plaintext so that Outlook's HTML viewer will not have the chance to invoke OLE for the display of content which, due to this very old bug in Windows OLE - like again, right, we're talking 2008, so this has been a problem since 2008. And it was recently found that there was a way to leverage this which, to my point,

is there's an active industry looking at ways to get into people's Windows networks. And probably not end users; right? They're sending phishing email into enterprises hoping that somebody will just, you know, Outlook just has to sniff it, and it's curtains. But not if you use a plaintext viewer.

And I know this is a hobbyhorse of mine. But this is why it seems wrong to me that Microsoft wants to sell the patch for this bug. How is it okay that they want to charge us for this? What they want to do instead is to force us to move to a newer operating system which has arbitrarily also decided that it may not support the hardware that we have. And as we just saw, these newer operating systems just had significantly more newly introduced vulnerabilities patched, compared to the older operating systems that are being allowed now finally to settle down because Microsoft has stopped "making them better" for us.

Anyway, the third critical 9.8 vulnerability is a trivial-to-exploit elevation of privilege in good old NT LAN Manager. That's the v1 version which refuses to die because there are things out there that still need Windows to connect to them. So it's remotely exploitable across the Internet, and its low attack complexity means that attackers need minimal system knowledge and can consistently can - and this is Microsoft saying this - can consistently succeed with their payload against a vulnerable component in Windows. To eliminate the danger entirely, don't expose any LAN Manager network ports to the Internet. And of course I've been saying for many years that there is no safe way to expose any of Microsoft's networking services, other than two - their web server and their email server.

All of the other services have been found to be vulnerable over and over and over. And if this "simply don't do it" admonition is not useful for you because your application needs you to do this, it leaves you with no other choice, Microsoft says that the danger can be mitigated by setting Windows' "LmCompatibilityLevel" to its maximum value of five on all machines. This forcibly disables both the original LAN Man and NT LAN Man v1, allowing then only the use of NT LAN Man v2. And of course, as I said, we've talked about how this could be a problem in heterogeneous environments where Windows machines have no choice but to communicate with older legacy equipment that, for whatever reason, cannot be updated. So many such situations like that exist today in the real world. That's just the way the real world still looks.

The simplest possible solution to all these I want to highlight again because, boy, do I use it, is to use IP address filtering, simple IP address filtering, where only the IP packets of specific remote machines, filtered by their IP addresses, are allowed to see the older and less secure Windows protocols. You know, yes, this does make the resulting network slightly more brittle, since firewall rules need updating in the event of IP addresses changing. But it is such a simple and bulletproof solution.

And many instances exist where someone casually just like exposed, you know, SMB protocol, Server Message Blocks, the NT LAN Man stuff, to the Internet, relying on username and password authentication, saying, well, you know, it's protected. It's not. And they're having connections coming from other fixed locations. If they're fixed, put a filter in front of that LAN Man port so that only those locations can see it. It's just so simple to do. And it is, I mean, it ends the issue. I mean, it's just such a good solution.

Okay. Before I leave last week's Patch Tuesday topic, I should mention a pair of remaining critical remote code execution vulnerabilities which receive CVSS scores of 8.1. Despite being remotely exploitable across the Internet, they were spared, you know, that same hair-on-fire 9.8 rating because their attack complexity was high. But the bad news is they both exist in Windows Remote Desktop Gateway. Once again, nothing but web and email. And the reason those are secure is they're publicly exposed, meaning they're not supposed to need to authenticate anybody. Anybody can access someone's web

server by design and emails in order to send them email. But Microsoft just doesn't seem to be able to get authentication right, no matter how much time goes by. And boy, are we going to see an example of that in one of our listener feedbacks coming up.

Okay. So Remote Desktop Gateway has these two 8.1 CVSSes. So we've seen problems with this before. And unfortunately, many enterprises believe that they have no choice other than to expose the Remote Desktop Gateway to the public Internet. I would argue that there are always ways around that. But one needs to care enough first to do so. Hopefully our listeners, you know, none of our listeners are any longer affected by this. They've come up with a way of putting something else in front of their enterprise's Windows Remote Desktop Gateway.

To exploit these two vulnerabilities, an attacker needs to win - and we've seen this before also - a race condition by precisely timing their actions. That may be difficult, but most such Remote Desktop Gateways sit unattended and unmonitored, meaning that attackers can try and retry without limit until they succeed. The attack involves connecting to a system running the Remote Desktop Gateway role, then triggering the race condition to create a use-after-free scenario. So memory is being released.

Somewhere a pointer is still not freed and is pointing to that released memory, which then gets reallocated, giving the attacker a pointer to something that might have some juicy content and gives them the hook. So, if successful, Microsoft agrees the attacker can leverage this to execute arbitrary code on the target system. Given the patches available, it appears that this problem was introduced in Server 2012 timeframe since Server 2008 is not affected. So 12 years ago. Or 13 now.

I certainly understand that, once bitten, large enterprises will understandably be very wary of Windows Update, you know, bringing down any of their important applications and infrastructure. It's a devil's bargain. So the best enterprises can do is to give each second Tuesday's updates immediate attention, get the updates deployed as quickly as practical, after verifying that installing them on a few sacrificial systems keeps all the enterprise infrastructure stuff and critical services functioning.

So that said, the smarter thing to do, rather than always being reactive to whatever the latest problem is - and as I said, they're not slowing down, they're arguably speeding up - is to really spend some time arranging to not be vulnerable to most of these problems in the first place by placing some other form of additional access control and authentication in front of anything having the need to offer secured public access and exposure. As I said, web and email servers are meant to receive anonymous connections from the public Internet. Pretty much nothing else is.

What we keep seeing is that the in-built authentication for any other private services is just not trustworthy and cannot be and should not be trusted. Once something other than Windows itself is protecting Windows services, none of this stream of ongoing zero-day actively-being-exploited-in-the-wild vulnerabilities will be a source of concern. That's where you want to be. So it's really worth spending some time thinking about how to get yourself into that position.

Leo: What's your sense - so it seems like, I mean, this is a huge number of flaws to patch. I mean, it's the largest since 2017, I think they said. Which would, just on the surface, people say, oh, well, look how, you know, insecure Windows is. But maybe it's the case that just Windows is in such widespread use that it's more likely that these are discovered and fixed than in a lesser used operation system. Do you think Windows is inherently less secure than any other operating system? Is this a sign of that? Do you understand what I'm saying?

Steve: I am. I do. On Microsoft's side, no other operating system offers the sprawl of features...

Leo: Right.

Steve: ...that Windows does. I mean, the reason enterprise...

Leo: Well, doesn't Linux? I mean...

Steve: No.

Leo: No?

Steve: I mean, Microsoft has, I mean, no enterprise, no sizeable enterprise cannot use Windows.

Leo: Okay.

Steve: You know, there are little artsy ad agencies with Macs.

Leo: Right.

Steve: That's, you know. But there isn't any enterprise or government agency, anything sprawling, because it's the one that they have to use to have the features that they want.

Leo: It has the most features. But along with the most features come the most bugs; right?

Steve: Well, yes. And, I mean, and it is significant that the older purchase the repairs had fewer flaws fixed than the newer operating systems. I mean, and every week on Windows Weekly, you know, you guys are talking, you and Richard and Paul are talking about all, you know, and we got this update, and we got this update, and all this is added now, and this now goes this way. And, I mean, Mary Jo used to be kept busy talking about all of this enterprise crap that they just keep adding. Well, any new code is going to have some percentage of flaws. That's what we see. And that's why I said that, you know, the older operating systems had fewer things to fix because Microsoft stopped screwing with them.

Leo: So it isn't necessarily, I mean, it's more insecure because there's more little edges to attack. But it's not that they're writing worse software, it's just the nature of the beast. And we've said this before, the fact that there were, what is it, 163 patches means there's 163 fewer problems. The longer it gets patched, the more it gets patched, the better...

Steve: The only argument to they're not writing worse software is that - was it 10,000 known bugs at release of, what was it, Windows XP or something?

Leo: Yeah. So a lot of those are cosmetic and, you know.

Steve: Yeah, yeah.

Leo: I mean, what we care about is security flaws. And 10 critical vulnerabilities and eight zero-days and 159 CVEs...

Steve: So somewhere in the world people that aren't listening to this podcast and aren't being sufficiently proactive are having their Windows networks penetrated.

Leo: Right.

Steve: We keep hearing about, I mean, I don't cover it anymore because it's so boring. It's all the ransomware attacks.

Leo: Every day.

Steve: But it's like, yes, it's still going on. And, you know, companies are being victimized. And so...

Leo: But they don't have a choice. You just said they have to use Windows.

Steve: They don't have a choice. Yeah, that's why I also called it a "devil's bargain." It is a devil's bargain. It is a devil's bargain. You have to use Windows because only it will do the things you need. But it is a system dragging legacy code forward. I mean, it's still got OLE in it.

Leo: Right. The fact that OLE's in there is tough, yeah.

Steve: Objects from Windows 3.

Leo: And that's another downside is you can't take anything out. Microsoft can't take anything out.

Steve: It'll break something; right.

Leo: Because somebody's using it.

Steve: Yeah. It's like IE6. It stayed around because people had written, you know, enterprises had written applications that only ran on IE6. And it's like, no, no, no. You can't take it. It'll, well, we'll go out of business. Ugh.

Leo: And when Microsoft has contemplated creating a secure Windows that doesn't have Win32 and is a lot safer, they back off because nobody wants it. That's not - nobody wants that. They don't want the more limited Windows. The whole reason they use Windows is because of all the features.

Steve: Yes. And Intel is a perfect example. Intel learned the lesson a long time ago, backward compatibility as we move forward. You know, you can still run, and I do, 16-bit code on the spiffiest triple-turbo-charged gazillion-core Xeon double-scoop processor. Works great. Boots DOS. You know? You can't even see it.

Leo: You can't [indiscernible] math, but. Okay. Well, it's an interesting question; right? I mean, I think on the face of it you say, well, look at all these flaws, you know, clearly it's a crappy operating system. That's not necessarily the case.

Steve: No. But the takeaway here is don't trust it.

Leo: And pay attention, yeah.

Steve: You can use it and not trust it.

Leo: Right.

Steve: Which means don't put it on the public Internet. Put something in front of it that you have to pre-authenticate to in order to get to it. Use an overlay network. Use...

Leo: Right, zero-trust or something.

Steve: Yeah. Some other system so that you - or use aggressive port filtering so that Russia and China can't just connect to an open port and go, let's see what we can do here. You know?

Leo: Second question. And this is really germane to many of our listeners who are not targets. Do you have to worry about this if you're not a natural target?

Steve: No. No. Nobody has Remote Desktop...

Leo: An individual like me.

Steve: We don't have Remote Desktop Gateway.

Leo: Right. Well, that's [crosstalk] true, yeah.

Steve: On our systems.

Leo: Yeah, I don't have...

Steve: And we probably don't have Remote Desktop exposed. And we're sitting behind a NAT router which is, you know, nature's perfect firewall.

Leo: And I still block IP addresses from Russia and China on my Ubiquiti. And there's also, I mean, I actually run quite a bit of security software. There's times I can't use sites because it's being blocked. For some reason I can't go to Taylor Lorenz's newsletter because...

Steve: And it's annoying that you can't prove a negative.

Leo: It is. I don't like it.

Steve: You'll never know what attacks you thwarted, but you can say, you know, toward the end of your days, well, I never got hacked.

Leo: Didn't get bit.

Steve: Yup.

Leo: I never have, as far as I know. As far as I know. That's a big one.

Steve: Yeah.

Leo: All right. I'm sorry. I didn't mean to interrupt. But these are interesting questions.

Steve: No, it's good to flesh this out. I mean, and I think you make a very good point. I have said I don't want that job at Microsoft. In the same way that I wouldn't want to be in charge of security for Sony Entertainment, I said years and years ago, because it's impossible to secure that.

Leo: As you have said, the hackers - you only have to make one mistake. They can make as many mistakes as they want. You only have to make one to be compromised.

Steve: Right, right. Every single thing that you do has to be secure.

Leo: Perfect.

Steve: Because they only need one route in.

Leo: What a world. It's fascinating.

Steve: Let's take a break, and then we're going to talk about this odd thing Microsoft's decided to do of forcing everyone to get the new version of Outlook.

Leo: This is the new thing. Did you know that Instagram has made every Instagram user follow JD Vance, the new Vice President? You're automatically following him.

Steve: You're not kidding?

Leo: No.

Steve: Oh ho ho.

Leo: There's this new compulsion thing that's happening that worries me a lot because we forget, but really these guys who run all of these apps have a lot of control, and they can do things that maybe you wouldn't want them to do. Anyway, okay. Although I think it's fun to follow JD. He's an interesting fellow. My ex texted me. She said, "I unfollowed him, and it got followed again." It's like, aye aye aye aye aye. All right, Steve. Let's see what Microsoft is imposing on us now.

Steve: Yes. Before we leave the topic of Microsoft I want to give a heads-up to our listeners about the forthcoming so-called New Outlook for Windows. The first I saw of this was a piece of news that said: "Microsoft will force install a new Outlook email client on both Windows 10 and Windows 11 on February 11th and January 28th, respectively." That news blurb then posted a quote which read: "Currently, there is no way to block the new Outlook from being installed. If you prefer not to have new Outlook show up on your organization's devices, you can remove it after it's installed as part of the update."

So I did a bit of poking around, and of course that revealed that the sharp folks over at BleepingComputer were on top of this. Under their similar headline "Microsoft to force install" - which I guess is now a term of art - "new Outlook on Windows 10 PCs in February," they wrote: "Microsoft will force install the new Outlook email client on Windows 10 systems starting with next month's security update. The announcement was made in a new message added to the company's Microsoft 365 Admin Center, tagged MC976059, and it applies to Microsoft 365 apps users.

"As Redmond explains, the new Outlook app will be installed on Windows 10 devices for users who deploy the optional January 28th update and force installed for all who install the February 11th security update," meaning next February's Patch Tuesday. "The new Outlook client will run alongside the classic Outlook app and will not modify

configurations or user defaults. Microsoft added that there's no way to block it from being installed on Windows 10 devices; however, those who don't want it can remove it afterward." Although actually it's a little trickier than that because it'll reinstall it. Well, we'll get there in a second.

So they said: "Microsoft wrote: 'New Outlook exists as an installed app on the device. For instance, it can be found in the Apps section of the Start Menu. It does not replace existing classic Outlook or change any configurations/user defaults. Both classic Outlook and New Outlook for Windows can run side by side. Currently, there is no way to block'" - this is Microsoft. "'Currently there's no way to block the new Outlook from being installed. If you prefer not to have new Outlook show up on your organization's devices, you can remove it after it's installed as part of the update.'" Then they said, BleepingComputer said: "The company added in a support document updated on Thursday." That's last Thursday.

So BleepingComputer said: "To remove the new Outlook app package after it's force installed on your Windows device, you can use the" - and then they show a PowerShell cmdlet Remove-AppxProvisionedPackage cmdlet with the PackageName parameter value Microsoft.OutlookForWindows. They said: "This can be done by running the following command from a Windows PowerShell prompt and adding a new reg value." And I've got this in the show notes for anyone who's interested, although you can easily find it from BleepingComputer.com.

"Next," they said, "add a reg string registry setting named BlockedOobeUpdaters with a value of 'MS_Outlook.'" Then they said: "After removing the Outlook package, Windows Updates will not reinstall the new Outlook client." Otherwise they would, like every month it would be reinstalling it. They said: "The first preview version of the new Outlook for Windows was introduced in May of 2022. The app was generally available for personal accounts in September of 2023 (via the September 26 Windows fall update and the Microsoft Store on Windows 11) and for commercial customers in August of '24."

Okay, so this doesn't seem like, to me, like the end of the world. But, you know, I know our listeners. Some may object to having Microsoft force-installing a new and presumably unwanted Outlook client onto their machines. One would argue whether a Windows 10 or 11 machine could be considered theirs, but we'll leave that for another time.

Leo: Well, yeah, and mail has always been installed automatically; right? I mean...

Steve: Yeah. Yeah. That's a good point.

Leo: Outlook Express and all of that, yeah.

Steve: Yup. You know, so it's sort of there. So this new client is apparently based upon the web version. It's essentially, from what I could gather looking through the Microsoft pages, a port of the web client to a native Windows app. As such, it does not support Outlook's traditional and problematic PST file format, and it also does not support any COM, you know, component object model integration with Outlook. I also noticed that Microsoft says that, unlike traditional Outlook for Windows, the new Outlook offers "limited," they said, limited support for third-party email services such as Gmail, Yahoo!, and so forth. So if you've got your Outlook or an Outlook pulling from multiple other providers, you'll want to, you know, if you were wanting to switch to the new one, you'll want to make sure that it can because Microsoft appears to be moving away from that.

Okay. All that said, complete segue here, I want to take this opportunity to mention that I recently switched away from Mozilla's Thunderbird as my email client, to something that I am...

Leo: Wait a minute. You weren't using Eudora?

Steve: No.

Leo: Okay. I'm just teasing you.

Steve: But that's, you know, thank you, Leo. For years and years...

Leo: You did use Eudora, yeah.

Steve: ...before being driven to Thunderbird, my original true blue email client had always been Qualcomm's Eudora.

Leo: I do still use it, yeah.

Steve: In fact, my tech support guy Greg is still using Eudora.

Leo: Wow.

Steve: Works fine. Life was good. I didn't care when Qualcomm's support for Eudora ended because Eudora worked for me perfectly. But over time, as other email clients' behavior changed, cracks began forming. Email started coming in to me with high-ASCII or Unicode weird like capital "A's" with umlauts in them, added to space characters. And for about a year or so...

Leo: I thought that's how you spelled Viagra.

Steve: Yes, well, it wasn't me spelling it, it was people sending me email. So for a year or so I manually edited them out of every reply that I was quoting. Until, I don't know, a couple years ago I finally decided to switch to Thunderbird. I tried The Bat! for a while, and that never really took hold.

But, you know, I then used Thunderbird for several years. And truth be told, I've never really been happy with it. I'm very finicky about the appearance of my outbound email, you know, the email that I author, and even when I'm quoting somebody. And, you know, pretty much everything that I produce I care about. Our listeners know that well. And Thunderbird's handling of fonts and formatting, the indentation of email threads, and the signatures it appends to email never made sense to me. It was trying to handle formatting details, but it made things mysterious and deliberately uneditable. It's like, don't worry about it, we'll take care of this for you. I wasn't allowed to fix these things when they didn't look the way I wanted them to because Thunderbird's formatting was

not only erroneous, but it was automatic. It apparently believed that it knew better than I did about how things should be. Maybe for some users who just don't care, great, take care of this for me. But it bugged me.

So finally, about two weeks ago, something drove me to seek another email client. As I mentioned, I already had an old copy of The Bat! around, so I tried to resurrect that, but it wasn't - didn't seem to be any kind of an improvement. So I went - oh, and I ought to also mention that Thunderbird really started acting up after I added the whole new GRC email system because incoming email from our listeners has been quite successful. I've never mentioned that I have, I think it's 4,484 pieces of email from our listeners. So that really seemed to, like, Thunderbird kind of got lost somewhere. It would just stop showing me new ones. I'd have to, like, give it a kick and shut it down and restart it or shake it three times. I mean, it just wasn't working. So anyway, so I went, I spent some time two weeks ago cruising around the various Top Ten Best Email Client lineups until I stumbled upon one I had never heard of before named eM Client. And life is good once more.

Leo: Ah, I'll have to try this.

Steve: It's a little difficult - and there's one for the Mac. They have a version for the Mac.

Leo: I've been using Pegasus on Windows, which I like. It's been okay.

Steve: And if you like what you've got, I'm not going to try to convince you otherwise. It's a little difficult for me to explain exactly why...

Leo: It's a personal thing.

Steve: ...it makes a huge difference to me. And yes, it is a personal taste, personal choice thing. But I can say that after setting it up as an IMAP client and allowing it to synchronize with GRC's email server, I almost immediately felt that I had a handle on my email. It found back-and-forth email from long ago and knitted them into threads. It allows me to mark things in various names and colored tags and to then view all of my emails and tags as folders, which are now dynamic. I can also see all my inboxes consolidated into a single view. It doesn't do any mysterious, unwanted, and wrong things with nesting of replies. You know, and since my needs are not necessarily aligned with everyone else's, I'll briefly share a broader view from Wikipedia.

Wikipedia's eM Client page says: "eM Client has a range of features for handling email, including advanced rules management, mass mail, delayed send, or a built-in translator for incoming and outgoing messages. It supports signatures, Quick Text, and tagging and categorization for easy searching. Watch for Replies and Snooze Email functions are available, as well as direct cloud attachments from cloud services like Dropbox, Google Drive, OneDrive, ownCloud or Nextcloud.

"eM Client also provides a lookup service for GnuPG public keys, their eM Keybook in order to more easily send encrypted communications via email, and generally simplify PGP encryption in email communication. eM Client supports all major email platforms including Exchange, Gmail, Google Workspace, Office 365, iCloud, and any POP3, SMTP, IMAP, or CalDAV server. Automatic setup works for Gmail, Exchange, Office 365, Outlook, iCloud, or other major email services. Following the shutdown of IncrediMail, an

auto-import option was added to transfer data from this platform to eM Client. Since v8.2, eM Client supports online meetings via Zoom, Microsoft Teams, and Google Meet. eM Client allows extensive appearance customization. eM Client 10, released in 2024, also provides AI features for composing messages and replies, Inbox categories, and Quick Actions which allow users to create their own macros."

So I need, like, just give me IMAP, please. I mean, but I need, like, four accounts to help me organize things. Okay. So here's my complaint. My only complaint is that the free version will only handle a single email account. And as I said, I need at least four. And that would be okay if I could purchase a paid version once. But it's "rental ware."

Leo: Yeah, it's a subscription.

Steve: Only available for $40 per year. I rent no other software of any kind, and that's something I actively fight against. So this is the first time I have ever capitulated. But come on. At $3.33 per month...

Leo: It's not expensive, yeah.

Steve: ...allowing installation on three machines, the experience of using this client continues to impress me. And if paying something is what's required to keep this stunning creation alive and maintained, then I'd rather do that than not have any access to it at all. I didn't realize really how unhappy I had been with Thunderbird until I began using eM Client. It's like a continuous happy breeze that washes over me whenever I look at it. Mobile editions are available at no charge, and I can't vouch for anything about it other than their Windows edition, which is all I've used. But as I said, macOS, iOS, and Android are all there. They claim to be in use in over 100,000 businesses and have 2.5 million users.

Leo: Ooh, it has PGP built in.

Steve: Yes, it has PGP built in.

Leo: Ooh.

Steve: And also a GnuPG key management is also built in.

Leo: Oh, now I'm interested, yeah.

Steve: Yeah. So for anyone who might be seeking a similar improvement to a major aspect of their lives, eM Client is available for download. You can get it feature-complete for 30 days in trial mode. I've been tweaking it here and there, like removing displayed columns that I don't need, you know, and I could not be happier. Oh, it's also possible to export all of the tweaks and preference settings you make into an XML file and then import them into another instance of eM Client on a different machine so that you're able to keep cloning all of the improvements that you make as you tune and tweak it along

the way. I've been moving back and forth among machines so I've been able, as I said, to keep the instances looking and operating the same.

Anyway, so I just wanted to pass this along in case any of our listeners might be wishing for something better. This could be it. It's www.emclient.com. And it's not - I can't give you a comprehensive review because I haven't done all these other things with it. But my sense is, you know, as you said at the beginning, Leo, everyone's needs and tastes are so different that no one else's opinion would or should matter to be other than a pointer. So I'm just giving everybody a pointer. As I said, I just need multiple IMAP accounts, and a consolidated inbox is nice to be able to tag things for follow-up and then be able to look at them all as if they were a folder. That's cool. It threads beautifully. Anyway, I just...

Leo: Does it show your GRC Ruby logo?

Steve: It does. But I might be getting it from a favicon because it beautifully pulls favicons from everybody.

Leo: Yeah, I notice that's what it's using, yeah. I just installed it. Very easy. Very straightforward. I will play with it, yeah. It's very interesting, yeah.

Steve: So anyway, I don't know why, but it just - and it could be subtle things, like just the way it sorts or filters or something. But I'm really happy. So I just wanted to share my happiness.

Leo: It has to fit your kind of gestalt. Yeah, yeah.

Steve: Yeah, yeah, it does.

Leo: Interesting. I'll be playing with it.

Steve: Oh, and a listener who is apparently listening, or maybe he just read the show notes, he said: "Hi, Steve. I've been using eM Client for two years now on the Home PC and have been happy with it. Back then I bought a license with only a one-time upfront cost." Oh, had I known.

Leo: I think they, no, I think they still so. Maybe not.

Steve: No.

Leo: Somebody in this - no. They don't offer that anymore.

Steve: He said: "I added lifetime upgrades to that for another one-time fee." So, boy, had I known, I would have done that. He says: "I see that the company charges monthly/yearly now, but they still have a lifetime upgrade purchase option, as well." Whoo.

Leo: Lifetime upgrades, I see it right here for eM Client.

Steve: He says: "I bet you can pay once and have the software from now on. It doesn't make sense for them to charge..."

Leo: $90? What? Interesting.

Steve: Well, so, I mean, that's interesting. And I wonder how many systems you're limited to, if that's all of your personally owned systems.

Leo: Right. Right.

Steve: Because based on what I've seen - again, Leo, I am so - I have just - I have a philosophical problem with...

Leo: I understand.

Steve: ...this whole mode of renting software, you know, paying by the month or by the year. It just annoys me. I just want to own it so that it's mine.

Leo: Yeah, I know what you feel. But I think these days developers are saying, look, if we're going to keep developing it, we're going to keep working on it, that one-time fee is [crosstalk].

Steve: Exactly. And as I said, so first of all, thank you, whoever you are. He signed "AC," so I don't know. But, you know, thanks for that. I'm glad to know that. I will look into that because, I mean, I'm so happy with this thing, I would do that if it would solve my problems.

Leo: Nice. Good. Thank you for the recommendation.

Steve: But to the point of paying, if that's what it takes to create a revenue stream to keep it like compatible with everything and up to date and so forth, then it's like, okay, yeah. I guess, though, I would prefer the old-school option of here's the next version. You bought 10. Here's what 11 does.

Leo: Right.

Steve: Do you want these things?

Leo: Right.

Steve: And so it's up to them to entice me to move forward for an upgrade fee.

Leo: A lot of people do that. I prefer that, as well, offer the early upgrades or whatever, yeah.

Steve: Right. And you know me. I like to offer them every two decades, so - wait, no. Wait, wait. I made it free, didn't I, after 20 years. So I didn't [crosstalk], either.

Leo: Yeah, yeah. Wow. You're crazy. You're a crazy man. On we go with the show, Mr. G.

Steve: So we've previously covered the various security troubles with GoDaddy's web hosting service. The sense I've had is that adding web hosting was an afterthought behind their domain name services, and that that's what got them in trouble because we haven't seen problem with the mainstream domain name services. It's been, well, you know, we've got to add this feature because other registrars are offering hosting.

The news is that the U.S. Federal Trade Commission has decided to require GoDaddy to clean up its act. Last Wednesday the FTC announced that GoDaddy will be required to bolster its cybersecurity program to address years-long deficiencies. The FTC stated that GoDaddy's failure to use industry standard security measures led to what the FTC called "several major security breaches" - and we covered those at the time - between 2019 and 2022. The agency also alleges that GoDaddy deceived its customers about how adequately it safeguards its web hosting product. The agency said that consumers were sent to malicious websites and otherwise harmed after hackers broke into GoDaddy customers' websites and accessed their data.

The extensive information security measures which the FTC is requiring GoDaddy to adopt are similar to the reforms the agency also ordered Marriott to implement after that hotel chain - and we talked about that famously - failed to improve its cybersecurity posture despite being breached three times between 2014 and 2020.

In a statement explaining why the FTC had acted, Samuel Levine, Director of the FTC's Bureau of Consumer Protection, said "Millions of companies, particularly small businesses, rely on web hosting providers like GoDaddy to secure the websites that they and their customers rely on." GoDaddy, which has about five million hosting clients - wow - failed to track and manage software updates, analyze threats to its shared hosting services, properly log and continuously assess cybersecurity incidents, and silo its shared hosting from more insecure platforms.

They said GoDaddy also falsely advertised that it prioritized a strong security program and complied with international frameworks requiring companies take "reasonable" measures to protect personal data. Consequently, the proposed settlement order bars GoDaddy from exaggerating its security practices; orders it to design a "comprehensive," whatever that means, information-security program; and directs it to retain an outside company to assess its enhanced cybersecurity program when it launches and every two years thereafter.

So, okay. It's interesting that the reporting about this referred to the infamous Marriott Hotels - remember the Starwood?

Leo: Oh. Yeah.

Steve: That Starwood Group breach incident. What we recall from that is that Marriott acquired the independent Starwood Group whose network security was a lackluster afterthought, if you can call it that. You know, like way out of date. They didn't bother to update, and there were, like, known, well-known problems. But Marriott, the acquirer, never took the time to thoroughly vet what they were purchasing, and that lack of oversight over their purchase came back to bite them.

Now, GoDaddy's past is similar, inasmuch as it has grown into the behemoth it is today - it's the number one registrar - through a long series of mergers and acquisitions, buying up and consolidating independent Internet registrars. And I recall also that their web hosting business was the result of one or more similar acquisitions. So, much like Marriott, they purchased something that needed work, and was then bitten when their name became tied to that new acquisition's poor security.

I'm sure there's a lesson here for any large organization that purchases any other high-tech entity and just sort of decides they want to bring it under their wing. And you know, probably promises like, oh, don't worry, we're going to allow you to maintain your autonomy. We're not going to get all in there and micromanage you. Okay. But the purchase negotiation should include a very thorough and deep independent third-party review of that soon-to-be-acquired company's security practices. For one thing, the enforcement of true security can be expensive; right? I mean, it's one of the reasons it's not done. Not only is it annoying, but it costs something. That means that an entity's true bottom line profit may be inflated due to a lack of sufficient security. It's making lots of money because it's hoping nothing bad happens.

Since any missing security practices would need to be added afterward, a better purchase price might be negotiated once its lack of security had become apparent. And in any event, the buyer will have a better idea about the potential liability that might come along as part of the package if they don't do something about that beforehand. So again, consider the security, you enterprise people out there, of anything that you might be acquiring and hope, you know, that you can just leave alone. They probably want to be left alone, but you need to decide if you could afford to do that.

I saw a news item that indicated that the U.S. Supreme Court appeared to be poised to support the enforcement of age restriction for adult-content websites. The determination being made was whether more than one third of the site's content contained adult-oriented material. That would be the determination of is this an adult content website. And, if so, any such websites would be forced to affirmatively verify any visitor's age before they would be able to view that site's content. And, you know, how do we get there from here? It's not clear. We don't have a widespread system in place that prioritizes privacy. And what occurs to me is especially for those adults who want privacy in and about the sites they visit, being forced to disclose their identity, that's sort of a - that's going to be a problem for them.

Anyway, since we had just discussed this issue last week, I decided that it was worth mentioning again because I ran across some other news from across the pond about what's to transpire in the United Kingdom. And since the verification of age is I think clearly a sticky wicket here, I decided to share the news from the UK. The publication, the security site The Record reported the following last Thursday.

They said: "The United Kingdom's communications regulator Ofcom, that we've oft spoken of, announced on Thursday that online pornography sites must, by July" - so we've got six months - "verify that all of their users are adults or potentially face being blocked by the country's Internet service providers. James Baker of the Open Rights civil liberties group who's, you know, going to be taking a counter position, expressed concerns that 'the roll-out of age verification is likely to create new cybersecurity risks in

the form of additional scam porn sites that will trick visitors into handing over personal data to verify their age." Which hadn't occurred to me, either.

The Record said: "Ofcom has set out a range of methods that it considers highly effective for checking users' ages, including photo ID matching and checks on credit cards, which you must be 18 to own in Britain. Other age-checking methods could be acceptable," said Ofcom, "but they must 'be technically accurate, robust, reliable, and fair in order to be considered highly effective'" per the definition in the legislation. "Specifically, the regulator has stated that the self-declaration of age and online payments using a debit card which do not require a person to be 18 would not be considered effective, and could leave those sites open to enforcement action. James Baker said: 'Some of the verification methods that Ofcom has defined as highly effective could put people at risk of new cybercrimes,' citing research published with the Electronic Frontier Foundation.

"The age verification measures are part of Britain's controversial Online Safety Act, which passed back in 2023 and aims to enforce technology companies to address a range of online harms. Businesses that fail to comply could face a range of enforcement actions, from being fined up to 18 million pounds, which is currently $22.3 million USD, or 10% of their global revenue, having their websites blocked by British ISPs or even face criminal prosecution.

"For their part, Ofcom's chief executive, Melanie Dawes, said: 'For too long, many online services which allow porn and other harmful material have ignored the fact that children are accessing their services. Either they don't ask; or, when they do, the checks are minimal and easy to avoid.'" Yeah, like I talked about last week, the Yes I'm 18 button. She said: "'That means companies have effectively been treating all users as if they're adults, leaving children potentially exposed to pornography and other types of harmful content.'

"She said: 'As age checks start to roll out in the coming months, adults will start to notice a difference in how they access certain online services. Services which host their own pornography must start to introduce age checks immediately, while other user-to-user services - including social media - which allow pornography and certain other types of content harmful to children will have to follow suit by July at the latest.'

"Baker, again of the Open Rights Group, said: 'There needs to be a specific and enforceable guarantee that age verification systems will be private, safe, and secure. The new plans miss this vital step, so place people at risk of data leaks and having their sexual interests exposed to blackmailers and scammers.'"

Wow. So I would say it's very safe to conclude that the handwriting is on the wall here. You know, like it or not, both the U.S. and the UK are going to be seeing some sort of true age verification, more than just pressing the button that claims your age, which I guess has just been there to technically let the sites off the hook, saying, well, this visitor said they were 18, so it's on them, not on us. And it's worth noting that whereas it's very difficult for any regulator to ascertain the effective network security of any given organization, it could hardly be any easier for regulators to determine for themselves whether a given website is effectively verifying the ages of its visitors. Just go there from any anonymous IP and see what happens.

So I don't know, Leo. Will it be a third-party entity that produces an age verification service? Will Apple and Google get in? I, you know, it's just not clear.

Leo: Yeah. There are AI-based kind of face recognition technologies. Paris wrote a story on information about Yoti, Y-O-T-I. But what you really don't want is for me to have to offer my driver's license to the porn site or go into a - this is something

Britain proposed a few years ago - go into a pub to verify my age by showing my driver's license and getting a certificate from the pub. I don't - it's a huge privacy concern. I think probably the best way to do it would be a third party, if you could trust the third party. Maybe a pub isn't such a bad idea, or a government office, where they see it, they look at it, they sign a paper that says, yes, you're over 16, you're over 18, and leave it at that. All, by the way, unaddressed by any of these regulations.

Steve: Right. All they're saying is we want this.

Leo: Figure it out.

Steve: You must do this. And, yeah. I saw something that was interesting, and the idea would be that a phone or a computer would have a verified age and identity with photos of you, and you would be required in real-time to do essentially a selfie for that app, so it would be seeing your animated real-time photo, be able to compare it to the photos it has on record of you internally, and say, yes, that's you, and then itself have an API that a site could verify in order to say, you know, I mean, and that's the thing, the kind of thing that Apple could offer if they were willing to get into this game.

Leo: This is what both Meta and Google and everybody have said is that, you know, Meta says we don't want to do this. X says we don't want to do this. The phone should do it. Because the phone has enough information. You can, I mean, in many states, I can do it in California, put your driver's license into your phone and use that for age identity without really revealing any other information.

Steve: Right.

Leo: So they're saying Apple should be responsible for this. Apple, on the other hand, does not want to be responsible. And I don't blame them. This isn't their problem. I don't know what the answer...

Steve: No, and of course it does, then, it means that anybody who doesn't have the requisite phone...

Leo: Right. That's a problem. Right.

Steve: ...is then disadvantaged, even though they may otherwise qualify. I mean, this is a real mess.

Leo: Yeah.

Steve: You know, I started out talking about how the cyber world is fundamentally different from the real world. If you were 10 and tried to walk into a strip club, you know, your age is...

Leo: Yeah, the real world, the bouncer's going to say get out of here.

Steve: Exactly. But on the Internet, no one knows how old you are. I mean, it's a fundamental difference, and we've been ignoring it up until now. We have been completely just saying, oh, well, you know, [crosstalk] problem.

Leo: Also I think you could make the case that the people who are proposing this really don't want it to work. They want porn to be banned. That's their real goal. And so in that case, you know, it's kind of disingenuous of them to say...

Steve: And we have real First Amendment problems in the United States.

Leo: Well, that's - they can't do that. So they have to do this kind of backdoor system. I don't, you know, it's going to be an interesting few years. But again, as I said...

Steve: Where have we heard that?

Leo: As I said, I think that hackers are going to be the freedom fighters, and that the people who know how to get around these things, how to use the Internet without giving up your privacy, are going to be the ones who come out on top. So start studying now.

Steve: If I were in high school, Leo, I could make some money on the side, I tell you. It's like that first scene in "The Matrix" where Neo is selling some contraband digital thing; you know.

Leo: Right, right, right. Or "Mr. Robot." Those people are - those are the ones. And you could be that one. If you listen to this show, you have the knowledge to become that person. Start thinking about your OPSEC and start considering these companies and the federal government as perhaps an adversary, and think of ways you can keep them out of your cheese. That's kind of what I think. But, you know, I'm old. I don't need to worry about it. So I'm going to leave that for you young folks. I got nothing to hide.

Steve: Yeah. Any AI that takes a look at us, Leo, is going to go, whoa, is there a heartbeat?

Leo: Every word in the house, every - this show, everything, is to an unknown AI. I don't even know what it is or where the server is or anything.

Steve: We know you gave up a long time ago.

Leo: I give up. And there's benefits, by the way, to that, as well. Until they come knocking on your door.

Steve: [Crosstalk] blood pressure goes down. It's like, yeah.

Leo: And say, "Mr. Laporte, come with us."

Steve: Oh.

Leo: And then my blood pressure might go back up.

Steve: Okay. So reinforcing the point I made about never relying upon any single manufacturer's public-facing remote access authentication, the security of the Fortinet security appliance, a major mainstream device, has once again been found wanting. In a posting on the Arctic Wolf security firm's website, titled "Console Chaos: A Campaign Targeting Publicly Exposed Management Interfaces on Fortinet FortiGate Firewalls," they listed four key takeaways.

First, Arctic Wolf observed a recent campaign affecting Fortinet FortiGate firewall devices with management interfaces exposed on the public Internet. Everyone heard that, right, "with management interfaces exposed to the public Internet." What could possibly go wrong?

Number two, the campaign involved unauthorized administrative logons - imagine that - on management interfaces of firewalls, creation of new accounts, SSL VPN authentication through those accounts, and various other configuration changes.

Third, while the initial access vector is not definitively confirmed, a zero-day vulnerability is highly probable. And I should note since they posted this it has been confirmed.

And fourth, organizations should urgently disable firewall management access on public interfaces as soon as possible. Once again, that final point, organizations should urgently disable firewall management access on public interfaces as soon as possible. Organizations should never have had it turned on in the first place. Again, you cannot count on any single vendor's authentication. Layer your security. Put a layer in front of anything that requires authentication. Always.

I forgot to mention that this is so serious that CISA and multiple cybersecurity firms warned of a zero-day vulnerability in FortiGate firewalls that hackers are actively exploiting. CISA ordered all federal civilian agencies to patch the vulnerability by today, January 21st, making it one of the shortest deadlines CISA had ever issued. And Fortinet said in an advisory that the bug is being exploited in the wild, but did not say how many customers had been impacted. The company said threat actors attacking organizations with the vulnerability are creating administrative privileged accounts on targeted devices and changing settings related to firewall policies. In other words, reading between the lines, we know that they're creating accounts and enabling SSL VPN so that they can then march right back in and get onto the internal firewall, or the internal network behind the firewall.

So patching as soon as possible is the responsibility of the owner of the device. But again, this was being exploited before any problem was known and before any patches were available. Secure remote access to a device such as this is entirely possible, but it should never rely solely upon the manufacturer's account logon protections. Always add your own independent layer of authentication. And that seems to be the unintended

theme of today's podcast because we're seeing so many instances where people are being hurt by not doing that. So do it.

Okay. So what's up with DJI lifting firmware-enforced drone geofencing? I posed the introduction of this next surprising bit of news as a question, so I'll follow up with, "And is it really?" But, like, it is. So why? I was put onto this by a short one-liner in the Risky Business newsletter, which said simply: "DJI gives the middle finger to U.S.: Facing an impending ban in the U.S., Chinese drone maker DJI has removed firmware restrictions preventing its drones from entering no-fly zones." So I thought, "Whoa! If true, I didn't see that coming, and that's no way to smoke the peace pipe with authorities in the U.S."

The Risky Business news then provided a screenshot of a posting by Matthew Stoller on Bluesky Social, which read: "Chinese drone maker DJI, the world's biggest drone producer, is disabling geofencing in the U.S. You can now fly your drone over airports, military bases, prisons, infrastructure, wildfires, and the White House, if you want. This is a gloves-off move by China," he finished, and then provided a link to the Viewpoints blog at DJI.

Okay. So Viewpoints bills itself as the official DJI blog, and it's at dji.com. I've got a link in the show notes for anyone who's interested. So last week's DJI blog, this was early in the week, is titled: "DJI Updates GEO" - that's all caps G-E-O - "System in U.S. Consumer & Enterprise Drones." And the posting says: "The update follows changes in Europe in 2024 and aligns with FAA Remote ID objectives. DJI has announced updates to its geofencing system (GEO) which applies to most of its consumer and enterprise drone products in the United States. These changes will take effect starting from January 13 on both the DJI Fly and DJI Pilot flight apps. This update follows similar changes implemented in the European Union last year.

"With this update, DJI's Fly and Pilot flight app operators will see prior DJI geofencing datasets replaced to display official FAA data. Areas previously defined as Restricted Zones, also known as No-Fly Zones, will be displayed as Enhanced Warning Zones, aligning with the FAA's designated areas. In these zones, in-app alerts will notify operators flying near FAA designated controlled airspace, placing control in the hands of the drone operators, in line with regulatory principles of the operator bearing final responsibility." Okay. So, you know, they're saying the same thing, but kind of in a gentler way. They said: "To update, operators need to connect their flight app to the Internet and click 'Update' on the FlySafe pop-up notification."

When DJI, and this is them, they're saying: "When DJI first introduced the GEO system in 2013" - so 12 years so - "consumer drones were still a relatively novel technology, and formal drone flight rules and regulations were sparse. The geofencing system was created as a voluntary built-in safety feature to help foster responsible flight practices and prevent DJI drone operators from unintentionally flying into restricted airspace, such as around government buildings, airports, or prisons.

"For many years, DJI has led the drone industry in safety, making several unprecedented commitments" - which apparently they're backing off - "to integrating advanced safety systems into its drones, including: First to install altitude limits and GPS-based geofencing to guide drone pilots away from unsafe locations. First to deploy autonomous return-to-home technology if drones lose connection to their controllers or have critical low batteries. First to integrate sensors for nearby obstacles and approaching aircraft. First to operate Remote Identification technology to help authorities identify and monitor airborne drones.

"Since then," they wrote, "global regulations and user awareness have evolved significantly, with a greater focus on geo-awareness and Remote ID solutions which makes detection and enforcement much easier. National aviation authorities, including

the European Aviation Safety Authority in the EU, the UK Civil Aviation Authority, and the FAA in the U.S., have established comprehensive geographical zones for unmanned aircraft systems and enforce drone regulations.

"This GEO update has been active in the UK and several EU countries since January 2024" - okay, so for the past year - "starting with European countries that have implemented geographical maps compliant with existing technical standards, such as Belgium, Germany, and France. In June, it expanded to Estonia, Finland, and Luxembourg. The remaining EU countries under EASA jurisdiction will also receive the update this month.

"DJI reminds pilots to always ensure flights are conducted safely and in accordance with all local laws and regulations. For flights conducted in Enhanced Warning Zones" - the new term - "drone operators must obtain airspace authorization directly from the FAA and consult the FAA's No Drone Zone resource for further information."

Okay, now, while this posting from early last week is far less inflammatory than the "middle finger" reference I first encountered, it does say exactly the same thing, which is it's going to be the responsibility of the drone operators, not the firmware and the technology, to enforce this so-called "enhanced warning zones." So in other words, operators will be notified, but the updated firmware will no longer prevent a DJI drone from flying right into and across what was previously designated as a no-fly zone.

Okay. Apparently, variations of this "middle finger" reference were widely picked up and circulated. And this prompted DJI to release a second blog posting later last week, on Thursday. The second blog posting was titled "DJI's GEO System Is an Education - Not Enforcement - Tool." It attempted to clarify DJI's position and I guess mollify the critics. It said: "Earlier this week, we announced an update to the DJI geofencing system (GEO) in which prior DJI geofencing datasets in most of our consumer and enterprise drone products in the United States will be replaced with official FAA data.

"We first introduced the GEO system in 2013, at a time when consumer drones were still" - and they repeat that paragraph from the first posting. They said: "However, some concerning reactions circulating online are either categorically false or seek to politicize this update given the current geopolitical climate. In the first Get the Facts article of the year, we want to take this opportunity to dispute the information and set the record straight."

Okay. "FACT 1," they say: "Politics does not drive safety decisions at DJI. For over a decade, DJI has led the drone industry in safety, making several unprecedented commitments and investments to integrate advanced safety systems into our drones, often ahead of regulatory requirements and without being prompted by competitors. To suggest that this update is linked to the current political environment in the U.S. is not only false, but also dangerous. Politicizing safety serves no one. We encourage discussions and comments to remain focused on technological facts and evidence. To understand the true reasons behind this update, read on.

"FACT 2: Aviation regulators around the world, including the FAA, have advanced the principle of operator responsibility. This GEO update aligns with and respects this principle. Similar updates to the GEO system began in the EU last year, with no evidence of increased risk. We had planned to roll this update in the U.S. months ago, but delayed the implementation to ensure the update worked properly. To add, over a decade has passed since DJI introduced the GEO system, and regulators have not chosen to mandate geofencing, instead opting for solutions like Remote ID (which requires drones to broadcast the equivalent of a license plate), LAANC (automated drone flight approvals in controlled airspace near airports) and community-based training.

"FACT 3: The GEO system has always been an educational - not an enforcement - tool. The GEO system has also not been removed." Okay, well. "Warning zones and in-app alerts remain in place so continue educating pilots on safe flight operations." In other words, it's making them aware, but it's their choice. "This change gives back control," they write, "to operators and provides them the information they need to fly safely. DJI remains committed to promoting safe and responsible flight practices and will continue its community education efforts, reminding pilots to always ensure their flights are conducted safely and in accordance with all local laws and regulations."

And finally, "FACT 4: In addition to aligning with the FAA's operator responsibility-led principles, the update to 'Enhanced Warning Zones' provides two operator benefits. First, reduced operational delays for pilots. The previous 'No Fly Zones' often placed an unnecessary burden on operators. While a user could receive instantaneous approval through LAANC to fly, they were still required to submit an application to DJI and wait for manual review and an unlocking license." In other words, it was enforced. "This process could result in missed opportunities, delayed operations, or unnecessary wait times. This was especially challenging for commercial operators, drone businesses, and most critically, public safety agencies performing lifesaving work, where delays are simply unacceptable.

"And second, improved consistency with official FAA data. Previously, the global geofencing system relied on ICAO Annex 14 configurations for airspace around airports, which did not always align with official FAA data. This mismatch caused confusion among operators unsure about where it was safe to fly. By displaying official FAA data, this update ensures operators can view airspace as FAA intends, clearly understanding where they can and cannot fly." Or I should say should or should not fly.

And they finished: "We hope this explanation clarifies the real reasons behind the updates to the GEO system: an opportunity to align with regulatory principles, empower customers with greater control, and provide them with accurate, official information to confidently operate their drones within safe and permitted airspace." And I guess to me an interesting aspect is that they've deliberately taken themselves out of the loop and removed responsibility for creating exceptions to their policies, which is interesting, especially given who knows what's going to happen with them and the U.S. and legislation.

So, but, you know, when all is said and done, it's clear that their firmware will no longer be taking responsibility for flatly refusing to allow someone to fly somewhere that it believes they shouldn't. And given the concerns and accusations that have been levied at DJI over the possible use of their high-quality camera-equipped drones for unwanted surveillance, it's not a stretch to imagine the conspiracy theories that this would have triggered.

And given the United State's current political climate with China, which is certainly a thing, I have no idea what's really going on here. If nothing else, it would appear to be an inopportune time for DJI to remove its historically firmware-enforced No Fly system, which would seem like a good thing for them to have if they're saying, you know, we have no intention of allowing our drones to be misused for eavesdropping. Anyway, but I thought it was interesting, and I wanted our listeners to know that this had happened.

Leo: Yeah. It's very strange. It's like, if you want to get banned faster, do that.

Steve: Exactly. Allow your drones to fly over prisons and military bases and...

Leo: Well, Super Bowl is coming up. And remember, I mean, in the fires in L.A. that a drone punched a hole in one of the...

Steve: Yes. There were only two, they called them "super scoopers," which scoop up water. One was grounded because a drone punched a 3x6 hole in the leading edge of its wing.

Leo: And dollars to doughnuts it was a DJI, I mean, that's what everybody uses.

Steve: Actually, I saw the FBI photo of the debris, and it says DJI on a chunk of grey plastic.

Leo: Seems irresponsible to turn off the geofencing. You know, I have a DJI. I love my DJI.

Steve: It's the best drone. That's what everybody uses that is, you know, is a professional photographer.

Leo: I mean, I guess we should trust everybody that they're not going to do bad things.

Steve: And Leo, have you noticed how movies now have like all these...

Leo: Oh, yeah, there's drone shots all the time.

Steve: All the time. It's really nice to...

Leo: It is.

Steve: ...be able to offer that.

Leo: Much smoother than a helicopter shot. They've replaced, they've basically replaced the helicopters.

Steve: And much lower cost for movie producers.

Leo: Yeah, yeah. Getting all sorts of interesting shots everywhere now, yeah. And I immediately go - Lisa and I watch, I go, "Drone. Drone."

Steve: Yup. I say the same thing to Lorrie while we're watching a movie. It's like, oh, we wouldn't have that were it not for inexpensive drones.

Leo: Yeah. Not just movies. TV shows, everywhere.

Steve: Okay. We're at an hour 40.

Leo: Okay.

Steve: So a break time, then we're going to look at CISA's huge improvement in vulnerability, the huge improvement that CISA has driven in vulnerability remediation.

Leo: Nice.

Steve: It's an astonishing graph we have here.

Leo: Love it.

Steve: In the show notes.

Leo: All right. I will queue it up. Okay, Steve. On we go.

Steve: So in its recently published "Cybersecurity Performance Goals Adoption Report" - and I'm sure that's got an abbreviation - CISA said that the number of critical infrastructure organizations enrolled in its vulnerability scanning service - remember we talked about that they were going to be doing proactive vulnerability scanning from the Internet to detect problems early - doubled over a two-year period, reaching now 7,791 organizations at the end of August of 2024. CISA added 1,200 vulnerabilities to its known exploited vulnerabilities catalog through the same period. And during the two-year period of analysis, critical infrastructure organizations enrolled in CISA's vulnerability scanning service reduced their average remediation times from 60 days to 30 days. So cut it in half and cut a month off of what it had been.

I have a chart in the show notes showing the average remediation time over the past two years, from 2022, the middle of 2022, to the middle of 2024. And it's very clear. It shows federal, international, private, and SLTT, showing a clear downward trend in remediation times. And of course all...

Leo: That's good; right?

Steve: Oh, yeah, yeah, yeah.

Leo: It is, okay.

Steve: Yes, so that's - yeah.

Leo: Faster remediation, yeah.

Steve: It looks like it's, you know, almost like a third of what it was before overall. So followers of this podcast know firsthand that this is not a simple feat to pull off. It's especially true for any sort of large and lumbering bureaucratic organization, that is, you know, bringing your remediation time down like that. But this is truly looking like a significant change in the security posture and active vulnerability reduction which we know that we need.

You know, we talk about the work that CISA is doing more and more frequently because they're doing so many things surprisingly right. They really are having a huge effect by raising the awareness of cybersecurity as a crucial consideration for any and every organization. I would say, Leo, over the past, I don't know, five years or so, we've really seen, like, the notion of cybersecurity get on the map. Ransomware certainly helped. Seeing the true effect that being a victim created, nobody wants that for their organization. But it really - it's clearly happened now. So anyway, we've come a long way, certainly during the 20 years of this podcast.

Leo: Yeah. You deserve some credit. I think you've been fighting the good fight every week.

Steve: Well, you know, just taking a clear, sober look at the news, you know, we end up coming up with a bunch of conclusions that history keeps affirming for us.

A bit of Closing the Loop. Listener Earl Rodd, he said: "Other stats on six-digit numbers that I feel feed our psychological tendency to see patterns where there are none." He said: "Remembering that only 151,200 of the million have all six digits unique." Okay? So, you know, we've got a million potential, obviously, you know, 000000 to 999999. So a million potential six-digit numbers. Of those, only 151,000 and a few more have all six-digits unique. 157,600 have at least three of the same digit. That's more than have six unique digits, meaning that it is more common to have three of the same digit occurring out of only six. There's only six. So there are more instances of a digit repeated three times than all of them being unique. So that's significant. 395,200 out of the million have four or fewer unique digits. And 409,510 have at least two consecutive digits the same.

So, you know, so .4, right, 40%, actually 41% have at least two consecutive digits the same. So I think really there just aren't that many possibilities in a six-digit number. You know, and also in thinking about this again, we've talked about that famous Birthday Paradox a lot; right? Given randomly distributed birthdays occurring throughout the year of 365 days, we are surprised by how small a group of people is needed to get a better than 50% chance of there being any two people having the same birthday, a birthday collision.

When you think about it, the same thing is happening with our six-digit authenticator codes. Here we have six digits and only 10 possibilities for each one of those six-digit places. I think that the same sort of counterintuitive experience occurs where the likelihood of inter-digit collisions is actually much higher than our intuition would predict. You know, as with the surprising Birthday Paradox, every digit has a collision possibility with every other one. And there aren't that many possibilities for each digit.

I received a great piece of feedback from someone who's in the field trying to do the right thing. This is important because Microsoft, as I had said earlier, for all practical purposes owns the enterprise world. This listener's feedback contains a bunch of Microsoft jargon that will mean something to our enterprise listeners. For everyone else

these details are not important because everyone will be able to understand the fundamental dilemma that our enterprises face.

So he said: "Hi, Steve. I would like to remain anonymous. I'm 24 years old and have been a listener since around Episode 900. I work as an IT systems admin for a local government in North Carolina. One of my responsibilities is managing security for our city's police department. We are required to comply with the FBI's CJIS, that's Criminal Justice Information Services, security policy, which is updated regularly. I've included a link to the policy below. It's 451 pages long, and all law enforcement agencies must adhere to it and pass periodic audits."

Okay. So to interrupt here for a second, all that sounds like the right thing so far. This clearly sets a high bar that's onerous to meet. But we know from everything we've seen that unless this level of specification and its enforcement by audit are applied, you know, the everything appears to be working so let's not break it rule will be taken by default. You know, everyone has too much work to do, and no one wants to go looking for trouble. And while first achieving compliance might well be a heavy lift, once things have been tightened up to meet the audited requirements, remaining compliant should only require a much more modest effort going forward.

Okay. Anyway, our listener continues. He says: "One requirement in the policy found on page 97, requirement number 20, is especially challenging." Surprisingly, that is all secrets must be hashed and salted.

Leo: Huh. That's nice to hear. That's good.

Steve: But Leo, that it's challenging?

Leo: Yeah, well.

Steve: Okay. He says, you know: "We might wonder why that would be challenging; right? After all, hashing" - and this is to your point, Leo. "Hashing and salting stored secrets such as passwords has been standard operating procedure for a very long time."

Leo: Yeah.

Steve: I didn't find the earliest reference to salting hashes in our transcripts, that is, there are many of them. That's the problem. I have more than 10 pages of search results. But, well, of salt. So I am assuming we're not talking about recipes. I found a reference from 2012 where you and I were talking about it as if it was something that everyone knew. Right? So 12 years ago, yeah, of course, salt. And I imagine we were talking about it from the start. But I was curious for the sake of this discussion, how old the idea of salting a hash for storing secrets was. So I asked the o1 Mini Model of ChatGPT the following question.

Leo: You're finding a lot of use in these AIs, aren't you.

Steve: Oh, I love this thing, yes. There are some things it's very good at. I asked it: "What's the earliest appearance of the recommendation that stored passwords should

both be hashed and salted for secure storage?" And I received the following reply: "The recommendation to store passwords, both hashing and salting, has its roots in the late 1970s, primarily driven by the practical implementations in early operating systems and evolving security best practices." This thing's amazing.

It wrote: "Unix v7, 1979. One of the earliest and most influential implementations of salted password hashing was introduced with Unix v7 in 1979. This version of Unix featured the crypt function, which incorporated a 12-bit salt alongside the hashing process."

Leo: Before you go too much farther, do you want to quickly tell us what salting and hashing is?

Steve: Oh. Okay.

Leo: Can you do it quickly?

Steve: Yeah. Yeah. Okay. The idea is that we would always use a standard hash function like SHA-1 that we were talking about with the time-based one-time passwords. And so the idea is, rather than just saving a password, a service would hash the password so that, if their database was breached, the passwords themselves in the clear, like the thing that the user provided, would not be stolen. All that any bad guy could get would be the hash. The problem is that you could then - a bad guy could run through a bunch of common passwords, hash them in order to determine their hashes, and then look for any matches of the hashes with the stored password.

So the idea was to add what was technically termed "salt." That's, you know, like sprinkling some salt on it. The idea is you would just - you would take another value. And it doesn't even matter, and actually it would be non-encrypted, I was going to say it doesn't matter if it's not a secret.

But the idea is you would add the salt to the user's password so that the hash would no longer directly represent what the user password was, in order to break simple hash-matching problems. And that's why even here in Unix v7, 12 bits, which is 4096 possible combinations, 12 bits is enough. It doesn't need to be cryptographically strong salt. It just needs to - it's something thrown in to further scramble the hash so that - because you're always using the same hash function, you know, a well-known hash function. So that's the idea. And in fact in ChatGPT's response, it gave me a purpose for salting which I skipped here in the show notes. I just wrote down "skipping over o1's completely correct explanation of the purpose of salting."

It then added, under "Evolution in Security Practices," it said: "Following the implementation in Unix, the practice of salting hashed passwords became a cornerstone in password security. Early 1980s, security literature and guidelines began to formally recommend the use of salts in conjunction with hashing to protect stored passwords. And in subsequent decades" - again, decades - "as computing power increased and new attack vectors emerged, the methods for hashing, e.g., transitioning from DES-based hashing to more secure algorithms, like bcrypt, scrypt, and Argon2, salting became more sophisticated, further strengthening password storage mechanisms."

And then it ended with "Key takeaway: While the precise first recommendation in academic or security policy literature might be harder to pinpoint, the practical implementation of hashing with salting in Unix v7 in 1979 marks the earliest prominent

appearance of this security practice. This implementation set a standard that has been built upon and refined in subsequent years to enhance the security of stored passwords." Okay. I could not have phrased any of that any better.

Leo: Thank you.

Steve: And now we have a marker.

Leo: Yeah.

Steve: This brings us back to our listener who quoted page 97 of the security requirements his IT systems were required to offer. "All secrets must be hashed and salted."

Leo: Yeah.

Steve: Which he said was especially challenging. He continued - this is our listener. "Like many small-to-medium-size cities, we operate on a tight budget and are often behind on adopting the latest technologies. We still rely on Active Directory, which syncs with Microsoft Entra, formerly Azure AD, via Microsoft Entra Connect, for managing Office 365 products and Exchange Online. However," he wrote, "Active Directory does not salt user password hashes."

Leo: Of course not. Jesus.

Steve: And, he says...

Leo: By the way, this not computationally difficult. It is well known. There's no reason not to do that.

Steve: There is none, Leo. It's just obscene...

Leo: Ridiculous.

Steve: ...at this point. He says: "However, Active Directory does not salt user password hashes, and it seems Microsoft has no plans to implement this feature." And he's correct.

Leo: Wow.

Steve: Active Directory is still using older LAN Manager or NT LAN Manager user passwords which have never incorporated salt. Even though Unix had it in 1979. As we know, both of these technologies, NT LAN Manager and LAN Manager, are horrifically old and insecure. Yet they are still in use. So what are people supposed to do?

Our listener continues, writing: "From my research, Microsoft's suggested solution is to migrate entirely to the cloud" - no kidding - "with Entra ID, Azure AD, eliminating the need for on-premise domain controllers and moving all authentication to the cloud. Here's where we run into two major issues," he writes. Limited features in GCC, which is - GCC is the abbreviation for Government Community Compliance, which is one of the packages that Microsoft offers to governments.

He says: "We're on the GCC tenant of Microsoft 365, which lacks many features available to regular enterprise customers. I recall you mentioning the federal government's frustration with Microsoft. Local governments face similar challenges. Information about feature differences between enterprise, GCC, and GCC High is not easily accessible, especially from Microsoft. We tested a full migration to Entra ID with Intune for device management, but Intune in GCC is noticeably less functional than in the enterprise environment. Many settings and options are grayed out, often with messages indicating that our tenant didn't contain the correct license. And there are the high costs," he says. "Fully migrating to the cloud is expensive, with steep annual fees."

Leo: Yeah, of course. That's why Microsoft is not updating SMB. They want you to go to the Azure. Yeah.

Steve: Uh-huh. He says: "It would require us to upgrade every user's license from Office 365 to Microsoft 365. Given the lack of features in GCC, it's hard to justify the additional cost. So my question is, for IT environments that still rely on on-premise Active Directory, what solutions are available to salt password hashes in Active Directory? Thanks for your insight, and I appreciate all the work you do."

Leo: Great question.

Steve: Unfortunately, this is where the expression "caught between a rock and a hard place" comes in. I'm not an expert on Microsoft's enterprise offerings, for which I will be eternally grateful. But I poked around, and nowhere could I find any solution for specifically adding salt to Active Directory passwords. There are all manner of enhanced security and authentication features such as Kerberos. But even there, Kerberos authentication uses the unsalted password stored by Active Directory.

So on principled grounds, I so strongly dislike the idea of these blanket security requirements driving organizations into Microsoft's cloud services where they will even be more at Microsoft's mercy than they are today, and then have even less recourse when Microsoft raises their rental rates. The only thing I can suggest is that an appeal be made proactively to the auditor that they're beholden to, to explain the situation and ask what solutions other government organizations may have found. You know, has this single requirement driven everyone else into the cloud? Or is there a wink and a nod that allows this one requirement to be quietly ignored? Because I see no way around it.

Leo: Wow.

Steve: There is no way to add this to Active Directory. You know, Microsoft has moved on. They've moved to the cloud. And if you're holding onto actually owning your own hardware and keeping your costs low and leaving things as they are, well, you're going to need an exception because your passwords, believe it or not, have never been salted.

Leo: I will ask Richard tomorrow because he knows a lot about this stuff. He might have an idea. But I think you're probably right, that this is just Microsoft's way of pushing you into the cloud.

Steve: Wow. Dean Wheaton said: "Hi, Steve. I have a suggestion for the podcast. I'm a longtime listener, not quite back to the beginning, but something like 16 years. I am a member of Club TWiT, and I do enjoy the respite from advertising. However, I would like to know which advertisers support the show and maybe take advantage of special offers, for instance, for a VPN provider. Would Leo consider inserting a short, this podcast is supported by blank, which offers 15% off using promo code blank? Or whatever short announcement is appropriate, pointing the listener to the show notes which might have full details in place of each advertisement, instead of cutting out the advertisement audio. Best regards, Dean in Maryland."

Now, to Dean I say, I sometimes found myself in a similar situation. So I discovered some time ago that TWiT maintains an easy-to-find sponsors page at TWiT.tv/sponsors.

Leo: And this is up to date. If somebody doesn't buy ads, we take them right off of it. So if they're on here, they are currently supporters.

Steve: Yup. You can also just go to TWiT.tv, and it's in the menu at the top toward the right end of the page. And the entries there include the special discount sponsor codes...

Leo: That's right.

Steve: And their URLs. So anyone can at any time check that out. And that way you'll also get information about TWiT sponsors other than those that may only be a sponsor on this podcast.

Leo: Yeah. All these companies probably show up on Security Now! once in a while. The only reason they wouldn't be on is because we're sold out.

Steve: There's no room for them.

Leo: There's no room for them. Everybody wants to be on your show, I have to tell you. So they all deserve your patronage because they all support Security Now!. If they could get on, they would be on.

Steve: Yup. And as you scroll through that list on the screen, Leo, I recognize them all from your reads here during the podcast.

Leo: Sure, yeah. 1Password, Bitwarden, CacheFly.

Steve: Yup.

Leo: 1Password and Bitwarden were on today. Coda, DeleteMe, ExpressVPN. That's the VPN we recommend. NetSuite I think was on.

Steve: ThreatLocker was also on...

Leo: ThreatLocker was just on. Vanta was just on. I think Veeam was just on.

Steve: Yup. Thinkst Canary off and on.

Leo: Yeah, yeah.

Steve: And Veeam was also on, yup.

Leo: So, yeah. I think that the people who pay for no ads might not want to have those little short announcements. So we're just going to - go there.

Steve: Yeah. Anyway, it's easy to find for anybody who wants them, you know, just TWiT.tv, and it says "sponsors" up in the upper right.

Leo: If you click those links, that takes you to the offer, the best offer, the current offer.

Steve: So I have a piece of errata to share because my mistake was picked up by several of our listeners, who essentially asked variations of, "What do you mean, Syncthing hardly ever updates?" This feedback is from our listener Brendan Coop, who offered some interesting additional information. Brendan wrote: "I'm catching up on last week's show, and I was surprised to hear you say that Syncthing is rarely updated. I rarely use Windows, and love Notepad++, but agree that at times it seems to update just to increase the version number. I think the developer sends political messages with some updates, which is their right. I've been a Syncthing user from way back when BitTorrent Sync went from being a useful free application to a mess with lots of restrictions."

Leo: And they sold to Resilio. That's when I moved to Syncthing, as well. Yup.

Steve: Yup. He said: "I stumbled onto Syncthing and have never looked back. I have Syncthing running on more than 25 devices, including various Android phones and tablets. I have half a dozen backup servers running on ODROID HC2 and HC4 devices running Linux at various locations."

Leo: Wow.

Steve: "It functions as a live backup system that syncs as files are changing. Most of the time there's a local server that should sync quickly while the offsite servers can catch up,

even if I shut down the source device before the remote servers are synced up. I can also turn on my laptop when I use it. And before long, it matches my desktop computers."

Leo: Yup. Yup.

Steve: "Not sure what I would do without Syncthing."

Leo: It's become my backup strategy entirely. It's just incredible, yeah.

Steve: Yeah. He said: "One thing I've not heard you talk about is self-hosting the relay and discovery service."

Leo: Oh, interesting.

Steve: He said: "I've been doing that since day one and have it running at five or six locations. I never rely on the public servers that Syncthing provides." And he says: "TNO."

Leo: TNO.

Steve: He said: "When I first started using Syncthing, it was very early in the development, and it was a little rough around the edges. As I recall, it used to update more than monthly and possibly more than weekly at times. A while back they switched to a monthly update cycle. And it seems to update at the beginning of the month, most months. What made your comment about how rarely they updated it stand out, especially this month, is that they issued two updates shortly after the initial monthly update, which is unusual." In other words, I got it exactly wrong. He said: "You picked the worst month in the past couple of years to say they rarely update the software, since this is the first time in more than two years they've done it more than twice in one month."

He said: "I've attached the update log I have on one of my backup servers. Luckily, it updates automatically, and all of my Linux devices send me an email with my update log when they update." He said: "This month's updates included updates to the relay and discovery servers, which doesn't happen often. I had to update them three times this month instead of the normal zero times." And so, yes, we have a, I won't even try to read it or go through it, but yeah, many, many, many updates. Which somehow I've missed. So I certainly stand corrected. I'm obviously not seeing those update notices for whatever reason. And perhaps I did happen to see one specifically because there were so many of them last month, and so that caught my attention. In any event, I'm happy to have that corrected. And it's interesting to hear about Brendan's success running his own relay and discovery servers.

Leo: Yeah. I want to do that. That's cool.

Steve: I've considered doing that. But my particular application, because I've got fixed IPs, allows me to create direct point-to-point links between remote Syncthing instances. I took the trouble to do that, which I've been very happy with, after noticing that the use of the communal relaying was dramatically slowing down the resyncing process. In other words, Syncthing has become super popular.

As you'd expect, there are, although you can often knit between NAT routers and get a direct point-to-point connection, as we talked about in the early days of the podcast, using a rendezvous server in order to help two Syncthing instances both behind NAT still establish a point-to-point link nevertheless. Still, there are plenty of cases where that won't happen. So a relay server is needed where both instances go out to the relay server in order to have their traffic relayed. As that becomes more popular, and of course this is just a, I don't know who is nice enough to host these relay servers, but they're getting bogged down.

Leo: Yeah.

Steve: So that was slowing down my syncing to a point where it became intolerable. So I went to the effort of establishing point-to-point links. But I could see the feasibility of running a rendezvous server, you know, a relay and a rendezvous server myself for Syncthing because, like, Brendan, it really is a terrific service.

Leo: Yeah. And it would just be for you; right?

Steve: Yeah. I would just use it for myself.

Leo: Internal in the network, which means it would be faster.

Steve: Right. Brendan is in TNO mode. So he has pointed his Syncthing instances to the IP of his own relay server.

Leo: Right. So you can run public ones. That's interesting. But I presume you can also run the private ones.

Steve: Right. Right.

Leo: So that's what's going on is that there are people all over the world running public relays.

Steve: And thank you, all you people.

Leo: Thank you, yeah.

Steve: Yeah.

Leo: I had no idea. Wow. I'm sure it's fragmented so it doesn't - nobody gets the whole file or anything.

Steve: Yes. Yes. Oh, well, no, it's all - oh, Leo, it's all super encrypted. It is absolutely end-to-end encrypted. So all their relaying is opaque data that they have absolutely no access to.

Leo: Yeah, perfect.

Steve: Yeah, I mean, we wouldn't be - you wouldn't have me looking at you, telling you how much I use it.

Leo: And it's on GitHub, the relay server. So you could easily install it. I bet you there's a - I would hope there's a Synology package because that would make it very much easier for me just to have it running on Synology.

Steve: Yeah.

Leo: Oh, very interesting.

Steve: Okay. We are at our final break before we attack TOTP.

Leo: Let's go after - let's see, I mean, we talk about brute forcing a lot. I think this is going to be a very interesting education in the technique of brute forcing.

Steve: Yes. We established such a foundation last week for exactly what is going on here, that when the question of is it strong enough came up, I thought, ooh, let's answer that question.

Leo: Yeah. Now, Steverino, let us talk about brute-forcing TOTP. That's exciting.

Steve: So this week we have another example of an instance where a piece of listener feedback I started replying to kept expanding until it had acquired a life of its own...

Leo: I love it.

Steve: ...and I realized that our listeners would probably enjoy another journey and thought experiment in a direction this podcast has never taken us, bizarrely, I mean, except in broad strokes. Following from last week's podcast topic of HOTP and TOTP, this week we're going to take a detailed look at the task of attacking and cracking a key for the authenticators we all use. We're going to answer the question of whether the 80, eight zero, 80-bit keys that most sites give authenticators to use are long enough to contain sufficient entropy. And if by any chance you tend to skip podcasts from time to time so that you missed last week's main HOTP and TOTP topic, I would strongly suggest

that you pause here to first listen to that one, since I need to assume that everyone here is now aware of what happened last week.

So this all started with an interesting piece of feedback from our listener, Lachlan Hunt. Lachlan wrote: "Hi, Steve. I enjoyed your review of HOTP and TOTP algorithms in Episode 1008, and wanted to share some of my own observations. I agree that the algorithms are designed to be very easy. I had previously implemented it as a hobby project, and the whole HOTP algorithm can be done in around 10 lines of code. It's a fun coding challenge, and I used it to brute force the next year's worth of codes and see when interesting numbers will appear. See the screenshot showing my 1Password two-factor authentication token equaling 000000." And sure enough, he took a picture of his phone. He had presumably set the calendar and clock forward, knowing when it was going to happen, having done this reverse-engineering of his own code, and then watched it happen and took a picture. So very cool.

He said: "The widespread use of QR codes for setting up TOTP is not actually defined by either RFC, and instead seems to have originated with Google Authenticator and copied by all other implementers. The QR code encodes the secrets as base 32 strings." Now, okay. So base 32 means an alphabet of 32, so he says: "where each character represents five bits." Which could be this just 2^5 is 32. He says: "I had a look at the secrets for some of my own accounts to see how long the secrets were. Many sites had secrets with 16 characters, which is only 80 bits." Right? 16x5. Sixteen characters, 32 combinations per character, five bits per character, so 80 bits. He says: "On the other hand, the longest secret I saw was a full 256 bits, which seems extreme."

He said: "However, the HOTP RFC actually requires that the secret key be a minimum of 128 bits, with a recommendation to use 160 bits. The ones below 128 bits are technically not compliant."

Leo: Interesting.

Steve: And that's Google, by the way. So he said: "Finally, I thought it was a nice coincidence that there are a million possible six-digit codes, and there are a little bit over a million 30-second intervals in a year."

Leo: Oh, so it won't repeat for a year. Well, it will. I mean, it repeats; right? But you could have [crosstalk].

Steve: Yeah, actually it does not repeat.

Leo: Oh.

Steve: But in a year - because it just keeps on going. So you'll get a different set in the second year. But you will probably see them in a different order the next year.

Leo: That's fine.

Steve: And not necessarily because you could see the same one five times in one year.

Leo: Right.

Steve: And not see any for 10 years.

Leo: Right.

Steve: I mean, that's the nature of true pseudorandom.

Leo: Right, that's called "pseudorandom," yes.

Steve: Yes. Okay. So the HOTP recommendation of a 160-bit secret key input to the SHA-1 HMAC makes some sense since as we saw last week, SHA-1 produces a 160-bit hash, so that's also the output size of HOTP's HMAC. So there's some symmetry there. But the way the HMAC works, and obviously from what we've just said, and I did talk about last it week, the key length can be anything you want because you're just mixing it in, much like you are salting, very much like you're salting a password hash. You just throw in the secret into the HMAC and SHA hashing it all together. So it can be whatever length that you want.

But Lachlan observed that many sites were using secrets having 16 characters, which expanded to "only" 80 bits, and Google chief among them. How should we feel about that? Using a key having only 80 bits for this application provides - okay, and I'm going to read the number - 1,208,925,819,614,629,174,706,176 unique keys. That's roughly 1.2 million million million million possible keys. So we've got four sets of six zeroes following the 1.2. Okay. Which brings us to the question of whether this is a sufficient number. To address that question we need to remember that when judging relative security, everything is about the application in which the various security components will be used.

So what's the security model of an HOTP-based TOTP authenticator? The purpose of time-based authentication is the generation of a completely unpredictable code generated within any 30-second window. Using an authenticator whose specific key is hidden among more than 1.2 million million million million possible wrong keys would appear to meet that requirement. But one of the key concepts in security is that of a security margin. So how much security margin do 80-bit time-based authentication keys provide? To answer that question, we need to examine the system and design an optimal attack to determine a key.

Given the proven high quality of SHA-1 for pseudorandom bit generation, which is then wrapped by the HMAC algorithm, the only known attack on authentication would be brute force guessing of different input keys which would then be used to generate a specific six-digit authentication code output at a specific time. So let's say that we knew our targeted authenticator's output at a given time. So we know the time and the six-digit code produced at that time.

Given the solid design of the authentication algorithm, which is essentially an extremely well-designed cryptographically strong hash function with some ad hoc post-hash processing, the only strategy available to us is simple brute force guessing. That is, we can only go forward through that function. We cannot go backward. There's no way to go back, especially from a six-digit code to go back and somehow miraculously get an 80-bit key. The information is obviously not available in a six-digit code to somehow magically get an 80-bit key. So we can only go forward over and over and over.

Okay. So let's say that we knew our targeted authenticator's output. We start testing all 1.2 million million million million possible keys one at a time, starting at zero.

Leo: That's going to take a while.

Steve: It's going to take a while. Each key we feed into the algorithm is combined with a timestamp for the one-time authenticator output we know. That's processed by the HOTP's HMAC SHA-1 algorithm, each use of which requires two uses of SHA-1 with some XORing and bit manipulation. That's what the HMAC is. Then as we saw last week, we performed the extraction of the four bytes from the 20, followed by the modulus one million division to extract the remainder and to arrive at our first candidate six-digit code. Whew.

Leo: Whew.

Steve: Being a high-quality pseudorandom six-digit code, this first candidate will have one chance in a million of matching the six-digit code we're seeking. The probability of things happening is something that often trips people up. If the probability of something random happening is one in a million, we might tend to assume that giving that one in a million thing one million opportunities to occur...

Leo: Yeah, that'd fix it.

Steve: ...or in our case one million key guesses, that we would probably get a collision of six-digit values. And that's true. But it's not guaranteed. Probability theory tells us that even given one million guesses of a one in a million event, there's a 36.79% chance of never hitting upon the value we're seeking. 36.79%. So we're probably going to, but it's not guaranteed. 36.79%, we're not going to hit it. That does mean that given one million guesses, that the reverse, a 63.21% chance that we will hit it. So 63.21% that we will hit it, better than 50/50. But it's not certain that we would. For random events it's all about probabilities. And 693,147 guesses, so nearly 700,000, would be required to hit the 50/50 point, the 50/50% chance of guessing. 700,000 guesses, not 500,000, right, not half of the one million, 700,000.

Leo: That's interesting.

Steve: For an even chance of a one in a million guess being correct. So at this point all we can do is keep guessing key values. I should make clear that assuming the key was generated by a purely pseudorandom system, there's absolutely no benefit to generating trial key value guesses at random. No key-generating algorithm could be any better than any other. And being fancy about it would just take us some more time and waste some more resources.

So to generate successive guesses we're going to treat the key like a large 80-bit binary number that we simply increment. Starting at zero, we'll eventually test them all. The problem, of course, is that "80" is a lot of bits. We've already seen that there are 1.2 million million million million possible combinations of those 80 bits. So let's proceed and see what happens. We keep incrementing our key and keep producing six-digit codes until we hit upon the one that the target authenticator produced for the same timestamp.

So, yay! We found an 80-bit authenticator key that gives the proper six-digit output at the proper time. But that's no use to an attacker since it's never going to be that time again. And besides, they already know the proper six-digit code for that time. The goal is to be able to generate the proper code for any time in the future. So for that the attacker, and we in our case, since we're taking that role, need the ONE key that will do that.

The problem is that there are 1.2 million million million million possible 80-bit keys, and the only thing we've accomplished is to find the first key counting upward from zero that produces this one correct six-digit code. Since we know that these codes are randomly distributed throughout the entire key space, that means that there will be, on average, 1.2 million million million - okay, I've dropped one of the millions - 1.2 million million million total keys that will also produce this same six-digit code for this same timestamp. In other words, the discovery of that first matching code is very unlikely to be useful. We still need to eliminate many millions of millions of other keys.

To do that, we need some more sample outputs from the target authenticator. So we've just clearly proven one thing: There is absolutely no possible way for an attacker, unless they were to get insanely lucky, like 1.2 million million million times lucky, no possible way for an attacker who obtains a user's single six-digit code at one point in time, to reverse engineer a user's authentication key regardless of how much time and processing power they may have. And note that this is all symmetric crypto which has always been safe from any threat from quantum computing. So holding out for a quantum computer to arrive isn't going to help us here. This is symmetric crypto. Quantum computing only helps with public keys things.

Okay. So as I said, to usefully narrow things down, we need some more sample outputs from the target authenticator. Okay. So let's make that a given. Let's agree that our attacker is able to observe the target authenticator being used with the same key at multiple points in time. Okay. So how many points in time do we need that will allow us to achieve this?

As we've seen, each point in time gives us one code in a million. And in its first use, out of the total 1.2 million million million million possible keys, this one in a million matching would allow us to select one candidate key out of every million possible keys, on average, again, because they're not also perfectly distributed. They're randomly distributed. So it effectively reduced the candidate key space by a factor of one million. In other words, we're able to use a six-digit code generated by the targeted authenticator to weed out a factor of a million possible keys. Or phrased differently, each application of a different six-digit code can be used to reduce the remaining candidate key space by a factor of one million. Okay, so suddenly that doesn't seem so bad.

An 80-bit key space gives us a total of 1.2 million million million million keys. That's four millions. And we've seen that each use of one six-digit code for a given point in time will, on average, eliminate a factor of one million wrong keys that do not produce a matching six-digit output. So that would suggest that the use of four six-digit code output samples, each reducing the total key space by a factor of one million, would bring the key space down to one or two remaining candidate keys.

Okay. So let's go back now to that first test where we were incrementing the 80-bit key and generating a test six-digit code to look for a match against the authenticator's known output. We know that we will eventually find a match. We're just going to go linearly from zero. We're eventually going to find a match. And the probability of that happening is 50% during the first 693,147 tries, rising to 63.21% by the time we've tried the first million keys. So not quite two thirds assurance of it happening by the time we've tried the first million. But regardless, we know it's going to happen sooner or later.

So having found the first candidate key that gave us the first proper six-digit output, we know that this only reduced the possible key space by a factor of one million. So next we try this same candidate key against the second point in time to see whether we obtain the proper second six-digit code. This will still be highly unlikely since that first test left 1.2 million million million candidate keys, only one of which is the one we're seeking.

But nevertheless, we check the key against the second point in time and almost certainly fail. That means that the first test found a key that produced the proper six-digit result at this point in time, but not at the second reference point. So we need to keep searching. We move forward again until we find a match for the first point in time, then again check that against the second point in time. As before, there are still so many candidate keys that will pass the first test, but fail the second, that it's likely to take quite a bit more searching until we find a candidate key that passes both the first and the second tests.

But we're still a long way from home. Since each of these first two tests reduces the candidate key space by a factor of one million, together they reduce it by a million million. But since we started out with an 80-bit key that gave us a key space of 1.2 million million million million, that means that even after finally finding a candidate key that passes the first two tests, that the new key that was found is still only one among the remaining 1.2 million million that will pass both tests, so it's still exceedingly unlikely that the one we found that passed both of the two first tests is the proper key.

To test this we, of course, check this latest candidate against our third authenticator sample. As we know, there's only one chance in around 1.2 million million that this first key that passed the first two tests will also pass the third. And even if it did by some miracle pass the third test, it would still be one of among 1.2 million keys that would do so. So we would then need to test against a fourth authentication sample output to see whether that key, which somehow managed to pass the first, second, and third tests, was the one out of 1.2 million that can also pass the fourth sample test. And since there were "1.2 x 1,000,000^4" possible keys, even this might not be the one we're looking for. And we need to remember that when we succeed in this search, it all boils down to statistics.

That 69.3% number which we encountered earlier comes back here, since we're essentially performing four unrelated one in a million tests against random events where we need all four of them to succeed. So we would need to test on the order of 6.93x10^23 80-bit keys before we would reach the point of having a 50% chance. Again, we would need to test on the order of 6.93x10^23 80-bit keys before we would reach the point of having a 50% chance of finding a first key that passes all four of our one in a million six-digit matching tests. Now, 6.93x10^23 is 57.3% of the total 80-bit key space to search, only to achieve a 50% chance of success.

One question to ask is whether there might be any shorter route for brute forcing a solution. I've given this some thought, and I cannot see one. I considered various sorts of sieve approaches, like the famous Sieve of Eratosthenes, which is used to find primes, where you could apply a sieve to three or four samples to weed out. But actually that would be vastly slower than this. Testing against one test is by far the fastest solution. There just isn't a faster way to do this. The algorithm we just examined closely is going to be the fastest to check successive keys against a first test and then to apply successive tests only when they successively succeed. That minimizes the number of tests being performed.

And we also know that we will need to test 57.3% of the total 80-bit candidate key space in order to have just a 50% chance of success with no guarantee even then. And each test with a candidate key will require two uses of SHA-1 for the HMAC algorithm and the application of the ad hoc HOTP six-digit extraction. It's easy to say 6.93 x 10^23, just as it's easy to be glib about 80 bits. But 6.93 x 10^23 is 693 million million billion.

Leo: It's a lot. It's a lot.

Steve: So if an attacker, yeah, if an attacker were able to perform, say, a million billion of these complete TOTP/HOTP candidate key tests per second, we would still be left with 693,000,000 seconds. Now, that's if you could do a million billion per second. We would be left with 22 years full-time around-the-clock without pausing, never stopping, and even then only obtain a 50% chance of cracking a single key of a time-based one-time password when having a handful of that authenticator's outputs, which are necessary, and knowing exactly when each of them were generated.

Now, modern hardware has become very fast. Absolutely the case. But it's generally fast at performing simpler algorithms for which it's been designed, like straight SHA-256 hashing for cryptocurrency mining. The hash rates have gone insane there. Ad hoc algorithms, especially something as wacky as HOTP, which selects the bits to be divided based upon some bits in a nibble, would be much more difficult to accelerate. So it might be, yes, that even a million billion complete tests per second would be difficult to achieve in practice. And Leo, as we said at the top of the show, that's an advantage of a wacky ad hoc algorithm is it is more acceleration-resistant. I don't know if they did it on purpose back in 2005. But it is a consequence of their ad hoc wackiality.

But that said, given the current performance of cryptomining, and a million billion tests per second taking "only" 22 years for a 50% chance of success, that's not the sort of security margin that would or should make anyone feel completely comfortable. It's better when realistic estimates come in at 22 million years rather than just 22 years. This really boils down to how fast the individual tests can be performed. And how many of the testers you can have running at the same time.

Leo: And that's the point, I mean, how many times, how fast can you submit a one-time code? Is there some way you can download something so you could do it locally?

Steve: Oh, yeah, yeah, yeah. We're not actually asking the other end.

Leo: They don't have to respond.

Steve: Right. We are comparing against the code that the authenticator generated.

Leo: Oh, well, so you're right. This isn't - this is maybe a little more doable than we'd like.

Steve: Yeah. It is more doable than we'd like. You know, I'm not at all worried about sites being protected by 80-bit keys. But given that what we've just learned from this exploration, I would feel more comfortable if the keying material had at least 128 bits. That's a difference of 48 bits, and that makes a HUGE difference in difficulty. Adding 48 bits scales the entire problem up by a factor of nearly 281,475 million times. So NOW we're talking many, many millions of years, and we have the sort of security margin that means we never need to think about the problem again.

Leo: But what about quantum computing?

Steve: No. Quantum computers do not help with symmetric at all.

Leo: Okay.

Steve: So there is no help from quantum. Given that the key length being offered is entirely transparent to any authenticator user, meaning, you know...

Leo: Yeah, we don't care.

Steve: We don't know. We just scan a QR code. We don't know. There is just no reason not to use 128 bits or more for the key. 80, you know, it's okay, but more would be better. And 80 should definitely be considered a minimum.

Leo: Very interesting, yeah.

Steve: And now we have some basis for judging the security margin.

Leo: Very interesting. And of course computation is only going to get faster.

Steve: Yeah.

Leo: Orders of magnitude faster.

Steve: Yeah. Those, I looked at what the hash rates are on cryptomining farms. Oh, my god, they've got - I can't pronounce the number. Quintimzilliontillionbillions of hashes per second.

Leo: Of course they're all [crosstalk].

Steve: They've gone insane.

Leo: They're all dedicated. But, and this is just a second factor. You still have a password you'd have to get. And so I think it's probably adequate. But...

Steve: Oh, yeah, as I said, I'm not worried about it. But now we have a basis for judging, which we did not have before.

Leo: Good.

Steve: And that's why we do this.

Leo: Yeah. I love it.

Steve: On these crazy podcasts.

Leo: I love it. I was told there'd be no math, but obviously I was misinformed about math.

Steve: You were punctuating it with your giggles over my million million million million million.

Leo: A large number. A large number. Didn't mean to interrupt. Lachlan, thank you for stimulating this conversation. Very interesting.

Steve: A listener-driven podcast.

Leo: Yeah. All of our comments and questions today were great. Really appreciate it. We love our listeners. Thank you for watching. Thank you for listening.