EU OS

Description: Kuala Lumpur International Airport says no to a ransom attack, switches to whiteboard. A tired and jetlagged Troy Hunt got phished, then listed himself on his own site. Cloudflare completely pulls the plug on port 80 (HTTP) API access. Malware is switching to obscure languages to avoid detection. Forth, anyone? Password reuse doesn't appear to be dropping. Cloudflare has numbers. A listener shares his log of malicious Microsoft login attempts. Why no geofencing? 23andMe down for the count (reminder). A sobering ransomware attack and victim listing website. Gulp! "InControl" keeps VR planes aloft. And the European Union gets serious about a switch to Linux.

High quality (64 kbps) mp3 audio file URL: http://media.GRC.com/sn/SN-1019.mp3
Quarter size (16 kbps) mp3 audio file URL: http://media.GRC.com/sn/sn-1019-lq.mp3

SHOW TEASE: It's time for Security Now!. Steve Gibson is here. We're going to talk about how Kuala Lumpur's International Airport responded to a ransomware attack. I'll give you a hint, it involves whiteboards. The creator of Have I Been Pwned just got pwned. We'll read his disclosure. He handled it well, I thought. And then is the EU going to switch to Linux, and why that might not be such a bad idea. All that coming up and a whole lot more, next on Security Now!.

Leo Laporte: This is Security Now! with Steve Gibson, Episode 1019, recorded Tuesday, April 1st, 2025: EU OS.

It's time for Security Now!, the show you look forward to all week long because, well, when else are you going to have a chance to get together with the wonderful Steve Gibson and talk about technology and computing and security and privacy? Here he is, the man of the day, the man of the hour.

Steve Gibson: Sometimes a little AI, sometimes a little supplements, sometimes a little sci-fi, yeah.

Leo: A little of this, a little of that, a little vitamin D sprinkled on top.

Steve: Now, today of course is April Fool's Day.

Leo: Uh-oh.

Steve: And as you and I, you and I are in agreement that it's a dumb thing to do jokes. I mean, I don't want to go over and look at TheRegister.co.uk. It's like, who knows what's happening there? I did hear - one good security-conscious buddy of mine, we were together at Berkeley, I met him there, and we've stayed in touch. He sent me a note. He said: "April 1st is the only day people critically consider what they read on the Internet." He said: "Let's make every day April Fool's Day."

Leo: Good point. Good point. We should always be skeptical; shouldn't we.

Steve: It's like, I don't think that's...

Leo: No, even The Reg is not doing any April Fool's. I think people have finally realized it's just...

Steve: Gotten burned out, finally, yeah.

Leo: Yeah, yeah. Remember Google used to spend many, many cycles working on their April Fool's jokes. Seemed like a waste.

Steve: Yeah. Well, what is not a spoof or an April Fool's is the existence of something called "EU OS," as in European Union Operating System.

Leo: Wow.

Steve: Yeah, uh-huh, baby. It's like bye-bye, Microsoft. So for Security Now! Episode 1019 for, yes, April 1st, we're going to finish by talking about that. And it brings up some interesting things, Leo, that you and I are going to have fun talking about, like when FOSS, you know, Free Open Source Software, gets to be so important that, you know, that little guy with the block down in Nebraska holding up the pyramid, it's like, okay, wait; you know? Is this fair that, like, the European Union would be getting arguably an incredible amount of value out of this work which was volunteered and is thankless? So, you know, I mean, it feels to me like when it moves from hobby land, as it's kind of largely been, into like running the world, you know, does the model change? It's, you know...

Leo: That's a good question, yeah.

Steve: Yeah. So but we've got lots of other fun stuff to talk about first, of course. The first story was driven by a picture that I saw, the Kuala Lumpur International Airport immediately said no to a ransom attack and got out their whiteboard.

Leo: Wow.

Steve: It's like, hey, we don't have that many flights, so we're just going to have Benjamin write them down. Also, oh, Leo, a tired and jet-lagged Troy Hunt got phished.

Leo: Oh, no.

Steve: Then had to list himself on his own site.

Leo: Have I Been Pwned? Yes. Wow.

Steve: Anyway, he did a really good takedown of himself and, like, looked at like how did - what? How did this happen to me? Anyway, so going to have some fun there. Also Cloudflare decided to completely pull the plug on port 80. HTTP no more. Which, you know, it takes actors like that to be able to do that and make it happen.

Leo: Yeah. Yeah.

Steve: Also, malware is switching to obscure languages to avoid detection. And I said - this is sort of, well, it's apropos of that - Forth, anyone? And actually Lisp is among them.

Leo: Lisp is one of them, yeah.

Steve: Yeah. Password reuse, it appears, is not dropping. Cloudflare has numbers. A listener has shared his log of malicious Microsoft account login attempts. And I asked the question, seeing the list, which we'll be sharing, why no geofencing, Microsoft? 23andMe is, again, down for the count, just a little reminder there. And I have a little bit more information. Also we've got a sobering ransomware attack and victim listing website which, for those who want to jump ahead, is this week's episode-numbered Shortcut of the Week. Also a nice post from a listener, bit of feedback sharing that InControl, one of my pieces of freeware, is helping him to keep his VR planes aloft. And then we're going to take a look at what this EU OS means for them, and sort of what it suggests about where FOSS goes from here. So I think for a non-foolish April 1st we've got a great podcast for our listeners.

Leo: And we promise that all the stories you will hear today are true. None made up.

Steve: Yes. EU OS is - that would have been a great one, but it turns out it's true.

Leo: Sadly, they're all true. That's really the truth. Sadly, they're not made up. All right. Well, we're going to get to the show in just a second. Of course our Picture of the Week. Steve says we've seen it before. I don't remember it, but if you do, you could let us know. It's a repeat, but it's worth repeating, I think.

Steve: Yeah.

Leo: And somebody has already fed it to ChatGPT and come up with a replacement for you, Steve, that no one has seen before. So we'll show you.

Steve: Replacement for me the podcaster?

Leo: No, no, just for that cartoon.

Steve: Because, you know, I've already heard from our listeners saying I dumped all of these transcripts of the past shows into AI, and then I gave it, like, scan the news of the week and just be Steve.

Leo: Yeah.

Steve: And so if I suddenly start looking younger, then...

Leo: Yeah, it could happen. It could happen. Picture of the Week time, Steve.

Steve: So, yeah. This picture takes a rather boring sort of somewhat, I guess it's not really esoteric, it's important if you're writing code that you want to be correct topic and makes it, like, really puts a good sharp point on it and makes it a lot more fun. I gave this picture the caption "Subtle coding choices can land you at the bottom of the canyon." And so meep meep, beep beep. The picture we have shows the famous Wile E. Coyote and the Road Runner. And so the issue here is whether you test for your loop completion in code at the top of the loop or the bottom of the loop. And both placements have a terrific coding purpose.

So the idea being - so, you know, in code you have this general notion of flow control. That is, you know, an "if" instruction that jumps you somewhere else, changes the flow of the control of the code. Similarly, you've got looping, where you want to do something some number of times. And in some cases, the control of the loop is an expression which evaluates to true or false. So, for example, in the first case you would say, while something is true, and that could be an inequality expression, or a Boolean variable or whatever. While something is true, do the following. And so then that following whatever it is inside the loop would be done, and then you'd come back up to the top and reevaluate that expression. And it may now no longer be true, in which case you fall out of the loop. You drop out of the loop, and you continue executing code below.

The alternative is an expression which is expressed as "do that something," and then down at the bottom "while," and then you have your expression that you evaluate as true or false. So obviously the difference here is in the first case you're testing that expression at the top before you have even done what the loop contains once. In the second case, "do something while," you don't get down to the while until you've done it once. So this is so beautifully illustrated in the cartoon because in this coding example in the cartoon, the loop says "while not at the edge, run." That is, as in run toward, you know, like the Coyote is chasing the Road Runner, and they're running toward the edge of this cliff like with a really long canyon below. And of course we all remember seeing the little [sound] and then a little puff of smoke down there when the Coyote hits the bottom and miraculously survives.

So the Road Runner is using the first instance, "while not edge, run," meaning that "I'm not at the edge" is being tested before we run. So it stops before it reaches the edge. Unfortunately, the Coyote's logic is the other loop, in this case the wrong one, which says "do run while not at edge," meaning that the run happens before the testing for whether we've gotten to the edge. The Coyote overruns the edge, falls to the bottom of the

canyon. Anyway, just, you know, you'd have to be into code and geeky nerdiness to think that this was wonderful. But, you know, just a great illustration of the difference. And from a true coding standpoint, you know, the caption I gave it, "Subtle coding choices can land you at the bottom of the canyon," it's probably always the case when someone is writing code that this choice matters.

I have used both often because sometimes I do always intend to do something once, then decide, after that's done, do I want to do it again. Other times I want to check to see whether I need to do it at all and just skip over it, never execute the loop if that's the case. So, you know, perfect [crosstalk].

Leo: I guess you could say that you should check your edge cases before you get to the cliff, is what you're saying.

Steve: Yes. You want to, yeah, you want to make sure that you're not at the edge when you run.

Leo: Unless you could put a try and a catch at the bottom. If you had a catch at the bottom, you'd be okay. By the way, here's a ChatGPT redo of this.

Steve: Ah.

Leo: With me apparently putting my test clause at the wrong part of the loop, and that's not good.

Steve: Very nice.

Leo: Yeah, ChatGPT can do those now. It's a kind of a fun...

Steve: Unbelievable.

Leo: Yeah, it's amazing, yeah.

Steve: Wow. Okay. So the Malaysian Prime Minister Anwar Ibrahim has declined to pay a $10 million ransom after hackers - and you can put this on the screen, I think it'd be good, Leo, for people to see it - to pay a $10 million ransom after hackers paralyzed the IT systems at the country's main airport over the weekend.

Leo: Oh, my god.

Steve: This incident forced staff at the Kuala Lumpur International Airport to manually post the flight information on a large whiteboard with a pen.

Leo: Wow.

Steve: And it's just wonderful. I mean, it's like, you know, you can imagine grandchild saying to grandpa, "Grandpa, how did you used to know what gate to go to before computers?" And there's your answer.

Leo: Yeah.

Steve: So we've got, you know, flight numbers, destinations, time of departure, and then gate numbers in several columns. Anyway, this is what happens if you say no, we're not knuckling under to the ransomware guys. So, and we touch on ransomware a few times in this podcast. I've stumbled on a site that is quite sobering. So the Prime Minister said that it took him, he said, "It took me less than five seconds to decide to decline to pay the ransom."

Leo: Wow.

Steve: And no particular group has taken credit for the hack, and maybe now if they see this picture they go, well, I guess we're not going to get our 10 million. I don't know if they're going to give them back the decryption keys. Hopefully they're restoring their systems from backups, and they were able to retire the whiteboard. But they didn't waste any time coming up with a workaround.

So, okay. We would file this one under the heading "It can happen to the best of us." And I'm just, I'm always saying "there but by the grace of god" because I'm not saying none of this can happen to me. You know, I maybe have an expired certificate, or a compromised server. I mean, as I said, what is it, a week or two ago, when I learned about that PHP flaw in the CGI invocation of PHP, I was using a vulnerable version of PHP. I will say that, because PHP is on an isolated server, you know, I mean literally, that server can't do any damage if anything gets loose in it because I just don't trust that stuff that I didn't write myself. So, you know, again, it can happen to the best of us. Anyway, so an alternative heading might be "Even the most security-aware person can get tripped up."

Last week Troy Hunt, who's famous for his Have I Been Pwned password leakage tracking site and service, posted his piece titled "A Sneaky Phish," you know, P-H-I-S-H. "A Sneaky Phish Just Grabbed my Mailchimp Mailing List." So Troy wrote: "You know when you're really jet lagged and really tired and the cogs in your head are just moving that little bit too slow? That's me right now, and the penny has just dropped that a Mailchimp phish has grabbed my credentials, logged into my account, and exported the mailing list for this blog." He said: "I'm deliberately keeping this post very succinct to ensure the message goes out to my impacted subscribers ASAP, then I'll update the post with more details.

"But as a quick summary, I woke up in London this morning to the following." And then he posted for us, and it's on the screen, what he saw, which was the Intuit Mailchimp logo, and the page looks 100% legit, and it says: "Sending Privileges Restricted." And it says: "Hello. We're reaching out to inform you that your Mailchimp account's sending privileges have been restricted due to a spam complaint received on March 24th, 2025. We take these reports seriously to maintain a safe and trusted platform for all users." Then it says, under the heading "What Happened": "Your account has been flagged due to a spam complaint, and as a result you are temporarily unable to send emails until this issue is resolved.

"What You Need to Do," it says: "Please review your recent campaigns and audience lists to ensure compliance with our policies." Then, in bold: "Click below to review your account and take the necessary steps to restore your sending privileges." So anyone seeing this who uses Mailchimp probably got a few of these in the early days before they'd established their reputation, while some people were saying, what is this? I didn't ask to be on this list. And, you know, they complain. And so, I mean, this is completely believable. And Troy makes a point later, as we'll see, that it wasn't over the top. It didn't say your life will end in 15 minutes if you don't, you know, it was just - it was pitched just right. So, and he was jetlagged and tired. He clicked the button "Review Account."

So Troy said: "I went to the link, which is on mailchimp-sso.com and entered my credentials which, crucially, did not auto-complete from 1Password."

Leo: Ahhh. So 1Password wouldn't have let him do it.

Steve: It said that's not a URL that I've seen before, so the fields were empty. He said: "I then entered the one-time-password," you know, he has an authenticator, so the OTP and the page hung. He said: "Moments later, the penny dropped, and I logged onto the official website, which Mailchimp confirmed via a notification email which showed my London IP address. I immediately changed my password, but not before I got an alert about my mailing list being exported from an IP address in New York."

Leo: So that's what they wanted.

Steve: And moments later, moments after that, he said, the login alert from the same IP: "We'd like to confirm some recent activity on your account." He said: "This was obviously highly automated and designed to immediately export the list before the victim could take preventative measures. There are approximately 16,000 records in that export containing info Mailchimp automatically collects." Like, it turns out, GPS coordinates, and more than people would like to have exposed, but that's what Mailchimp collects.

He said: "Every active subscriber on my list will shortly receive an email notification by virtue of this blog post going out. Unfortunately, the export also includes people who've unsubscribed." He asks parenthetically: "(Why does Mailchimp keep these?!)" He said: "So I'll need to work out how to handle those ones separately. I've been in touch with Mailchimp, but don't have a reply yet. I'll update this post with more info when I have it."

He said: "I'm enormously frustrated with myself for having fallen for this, and I apologize to anyone on that list. Obviously, watch out for spam or further phishes" - meaning, like, somebody pretending to be him, for example, who wants them to do something with Have I Been Pwned, you know, because that's the way this could escalate or snowball. And he said: "Obviously watch out for spam or further phishes, and check back here or via the social channels in the nav bar above for more." He said: "Ironically, I'm in London visiting government partners, and I spent a couple of hours with the National Cyber Security Centre yesterday talking about how we can better promote passkeys, in part due to their phishing-resistant nature." And he had a face palm emoji. He said: "More soon. I've hit the publish button on this 43 minutes after the time stamp in that first email above."

So he prioritized immediately notifying all the people on that phished list that this was what happened so, you know, hopefully no further damage will be caused. So that was the blog posting that he quickly pushed out to let his more than 16,000 subscribers know

that, you know, the email address they had entrusted to him had escaped. Later he continued, under the headline "More Stuff From After the Initial Publish." He wrote: "Every Monday morning when I'm at home, I head into a radio studio and do a segment on scams. It's consumer-facing, so we're talking to the 'normies.' And whenever someone calls in and talks about being caught in the scam, the sentiment is the same: 'I feel so stupid.' That, friends," he wrote, "is me right now.

"Beyond acknowledging my own foolishness, let me proceed with some more thoughts: First, I've received a gazillion similar phishes before that I've identified early, so what was different about this one?" He said: "Tiredness was a major factor. I wasn't alert enough, and I didn't properly think through what I was doing. The attacker had no way of knowing that. I don't have any reason to suspect this was targeted specifically at me, but we all have moments of weakness. And if the phish event is timed perfectly" - you know, by coincidence - "with that, well, here we are."

He said: "Secondly, reading it again now, that's a very well-crafted phish. It socially engineered me into believing I would not be able to send out my newsletter, so it triggered fear. But it wasn't all bells and whistles about something terrible happening if I didn't take immediate action. It created just the right amount of urgency without being over the top."

Leo: Yeah, that was smart; right?

Steve: Yeah.

Leo: Because if it's like, oh, my god, you're going to go to jail, then he would have known.

Steve: Yes, yes. And he said: "Thirdly, the thing that should have saved my bacon was the credentials not auto-filling from 1Password, so why didn't I stop there? Because that's not unusual. There are so many services where you've registered on one domain, and that address is stored in 1Password, then you legitimately log onto a different domain." He said: "For example, Qantas airlines uses both 'www.qantas.com.au' and elsewhere 'accounts.qantas.com.'" So his point is, you know, we're all used to the occasional failure of our autofill because, as he said, authentication has gotten so complicated that even that isn't as straightforward as it once was. You know, and he saw mailchimp-sso.com. That looks, you know, possible. Probably should have been sso.mailchimp.com because then it would have been a subdomain of Mailchimp. Obviously the bad guys got this and so they were doing...

Leo: So they had sso.com, and then they were prepending.

Steve: No, no, it's mailchimp-sso.com.

Leo: Oh, hyphen. Yeah, that's not good.

Steve: So they just grabbed a domain that looked legitimate, knowing that someone like Troy or, you know, your typical user might go, okay, just make sure the URL seems right. And it's like, okay, that seems right.

Leo: Some password managers will say use the base URL for the matching.

Steve: But again, this was...

Leo: But that might change.

Steve: ...mailchimp-sso.

Leo: Well, that's a different base URL, though; right?

Steve: Right. Right.

Leo: Yeah.

Steve: So 1Password said, uh, what?

Leo: Yeah, yeah.

Steve: Now what would be interesting would be if a future password manager did a soft match and saw the, well, I've got Mailchimp.com. Here it is again. And then brought up an alert and said, hold on, this looks like one of your domains, but it isn't.

Leo: Right.

Steve: So then it'd be like, what? Anyway, he said: "And the final thought for now is more a frustration that Mailchimp did not automatically delete the data of" - and he says this is not, you know, his fault, more of a frustration - "that Mailchimp didn't automatically delete the data of people who unsubscribed." He said: "There are 7,535 email addresses on that list, which is nearly half of all addresses in that export." He said: "I need to go through the account settings and see if this was simply a setting I hadn't toggled or something similar," you know, meaning it was his fault for not turning on "delete email addresses when people unsubscribe." He said: "But the..."

Leo: Let me show you, by the way, and this is in Bitwarden. But Bitwarden, you have to do this on a per-site thing, but does have switches for detection of base URL. So you can have base to main, but you can also have a regular expression you could say it has to match exactly. I haven't tried 1Password. I would assume that 1Password would have this kind of feature, as well.

Steve: Right.

Leo: And then the next step is, well, if it doesn't fill, you really should check; right? Don't assume. Because he obviously manually entered it.

Steve: Right. Oh, no, no, no. Oh, yes. Yeah, yes. He had to manually enter his username and password.

Leo: When it didn't fill he said, oh, well, it's probably just a thing, so...

Steve: Yeah. And I'll often open the dialog because, you know, none of us know our passwords anymore, so I'll copy the password and then manually, you know, paste it into the password field. And it's like, okay, fine.

Leo: You might have done that, too, yeah, yeah.

Steve: So you're right. Anyway, he said: "The inclusion of those addresses was obviously completely unnecessary." He said: "I also don't know why IP addresses" - oh, and I'll just say one other thought, although Troy didn't, is even if 1Password wanted to keep them around for some reason, they could have been excluded from an export.

Leo: Sure.

Steve: Or, you know. So that they weren't exportable, even if, like, for example, maybe Mailchimp - I'm sorry, I said 1Password, I meant Mailchimp. Maybe Mailchimp needs to, you know, like keep them blacklisted if somebody maliciously resubscribes after saying don't ever send me any email again. I mean, for example, my own system does that. There's a button that I have where it's like, I don't ever want to hear from you again, no matter what. And that goes onto a permanent list. And I've said, if you ever want to get yourself removed from that, I'm going to have to write some code because...

Leo: You're stuck, buddy.

Steve: Yeah, I just, I don't - I never want to bother anybody with emails that they don't want.

Leo: Right.

Steve: So anyway, so he said: "Also, I don't know why IP addresses were captured." Whoops. "Or how the latitude and longitude are calculated. But all of that was in the export." So he was a little bit annoyed by that. He said: "But given I've never seen a prompt for access to the GPS, I imagine it's probably derived from the IP, which is certainly reasonable." He said: "I'll park this here and do a deeper technical dive later today that addresses some of the issues I've raised above." And again, I'm sure we can all give him a Get Out of Jail Free card just based on jetlag and fatigue.

Leo: Yeah.

Steve: You know, he wasn't soliciting this notice from them. He didn't go there. It showed up in his email. And again, looking absolutely believable.

Leo: That's when they get you, though, when your guard is down.

Steve: When your guard is down, exactly. You're in a hurry. You know, your buddies are outside saying, hey, you know, like, waiting for you to go to lunch. And it's like okay, you know, and you don't think. In fact, I'm always so careful when I'm, like, when I'm logging away from my servers that I log out and don't shut down because, you know, whoops. So those sorts of things happen after I remove the shutdown option. I've got to use command lines to do that.

Leo: Oh, that's smart, yeah.

Steve: Anyway, then a bit later Troy continued. He said: "Unfortunately, Mailchimp does not offer phishing-resistant two-factor authentication." And then we see a screenshot from them showing two-factor authentication, and what is configured is his authenticator app, and not configured is SMS because, you know, that's not going to be useful.

Leo: Good. Yeah.

Steve: So that's all good. But, he says: "By no means would I encourage people not to enable two-factor via one-time passwords. But let this be a lesson as to how completely useless it is against an automated phishing attack that can simply relay the one-time password as soon as it's entered."

Leo: Good point.

Steve: That's what happened.

Leo: That's what happened. It was so quick, yeah.

Steve: Yes. He was, you know, he went to - that mailchimp-sso.com was automated. The moment he logged in, and then prompted for his one-time password, it took all three of those - username, password, one-time password - immediately turned around, logged into his Mailchimp account, and triggered an automated mailing list export. And that's what it was designed to do.

He also wrote: "I just went to go and check on the phishing site with the expectation of" - now, that's meaning the mailchimp-sso.com. He went to check on the phishing site with the expectation of "submitting it to Google Safe Browsing, but it looks like that will no longer be necessary" because he was presented with a Cloudflare intercept page stated that the page was suspected of being used for phishing. So in the interval of, I don't know, a couple of hours probably, when he got back to it, that site, that mailchimp-sso.com site had already been blocked because others reported it as being a phishing site.

So he said: "Two hours and 15 minutes after it snared my creds, Cloudflare has killed the site. I did see a Cloudflare anti-automation widget on the phishing page when it first loaded and later wondered if that was fake, or they were genuinely fronting the page." As it turns out they were. He said: "But I guess that question is now answered. I know there'll be calls of 'Why didn't Cloudflare block this when it was first set up?'" He said: "But I maintain, as I have before in their defense, that it's enormously difficult to do that based on domain or page structure alone without creating a heap of false positives."

And Troy knew that he would need to load those addresses into his own Have I Been Pwned site. He wrote: "When I have conversations with breached companies, my messaging is crystal clear: Be transparent and expeditious in your reporting of the incident and prioritize communicating with your customers. Me doing anything less than that would be hypocritical, including how I then handle the data from the breach, namely adding it to HIBP. As such, I've now loaded the breach, and notifications are going out to 6.6K impacted individual subscribers and another 2.4K monitoring domains with impacted email addresses."

And he finished: "Looking for silver linings in the incident, I'm sure I'll refer this blog post to organizations I disclose future breaches to. I'll point out in advance that even though the data is 'just'" - he has in quotes - "'just' email addresses and the risk to individuals doesn't present a likelihood of serious harm or risk their rights and freedoms, it's simply the right thing to do. In short, for those who read this in future, not just as I say, but as I do."

So I've included a link to Troy's entire blog posting which proceeds with, at the time of this writing, a series of seven additional follow-ups. So for anyone who's interested, there is more there, if you want to follow the link, or probably just follow it from TroyHunt.com, which is where he blogs from. He spends a lot of time looking at the many benefits in these follow-ups of Passkeys, which are inherently phishing resistant because the information being sent back to the authenticating server is neither static username and password, nor short-duration one-time codes. The authenticating server sends a unique, never-before-seen challenge over an end-to-end encrypted link which the user's client signs. So any man in the middle is cut out.

But the biggest takeaway here is that phishing, which takes advantage of the human factor, remains an active threat today. And it can literally happen to anyone, even someone as astute as Troy who lives and knows this stuff inside and out. It just happened to catch him at a time of fatigue and jet-lagged weakness, but it did catch him. The addition of one-time-passwords has neutered non-real-time attacks where a user's login username and password have been stolen, like in a site breach. But automated attacks which immediately forward the user's provided one-time-password to the authenticating server remain 100% effective.

So that's worth keeping in mind. It's not like we get total protection from having to, you know, feel like we're James Bond and looking up our secret password which changes every 30 seconds and type it in. And remember that we've also seen how ridiculously long some authentication sites such as Microsoft will continue to honor tokens which expired many minutes...

Leo: More than 30 seconds?

Steve: Oh, yeah, yeah, yeah.

Leo: That's because it takes time for people to get the thing.

Steve: Yeah, I mean, we've covered it. There was an instance where it was like five minutes of window.

Leo: Oh, no, no, no.

Steve: And the attackers were hacking Microsoft's one-time password system because they were, by using crowd-sourced brute-forcing...

Leo: Right, right.

Steve: ...they were able to get all one million possibilities into that window.

Leo: Oh, that's right, I remember that.

Steve: To neuter anyone's one-time password. And then finally the fact that attackers used the domain "mailchimp-sso.com" further masked the attack, even to someone like Troy who probably noticed the URL. That was a perfectly reasonable domain name of the sort we see every day.

Leo: Yes. I agree with him. This is why Passkeys have to happen. That's why, frankly, SQRL should have happened. It would solve this problem.

Steve: Yeah. It did solve it. It's not the one we got, but we got one which is still phishing-resistant. And now what we just need is everyone, again, like nothing makes the world change. And we have got a couple more instances we'll be encountering here today of, like, what it takes to - it's actually our next story. But let's take a break, and then we're going to talk about Cloudflare making the world change in a good way.

Leo: Oh, yeah. Change is good sometimes. Not always.

Steve: Not easy.

Leo: Not easy.

Steve: And not something you do voluntarily. It's like, hey, well, it worked yesterday, and it looks okay today, so probably good for tomorrow.

Leo: Yeah. And Passkeys would be resistant; right? I mean, there's no interaction with the website. There's no way a third-party could snoop on that.

Steve: It cuts the third-party out of the loop, yup.

Leo: Okay. All right, Steve. Let's talk about Cloudflare.

Steve: So, yes. Their blog posting was titled "HTTPS-only for Cloudflare APIs: Shutting the Door on Cleartext Traffic." They introduced this change by writing: "Connections made over cleartext HTTP ports risk exposing sensitive information because the data is transmitted unencrypted and can be intercepted by network intermediaries, such as ISPs, WiFi hotspot providers, or malicious actors on the same network. It's common for servers to either redirect or return a 403 Forbidden response to close the HTTP connection and enforce the use of HTTPS by clients." And, for example, you know, you can reach GRC over port 80 still, HTTP. But my server just immediately bounces the user's browser over to the same URL, but HTTPS, in order to move you over to secure.

They said: "However, by the time this occurs, it may be too late because sensitive information, such as an API token, may have already been transmitted in cleartext in the initial client request. This data is exposed before the server has a chance to redirect the client or reject the connection. A better approach is to refuse the underlying cleartext connection by closing the network ports used for plaintext HTTP, and that's exactly what we're going to do for our customers." Wow. I mean, that's okay. What will break?

And they said: "Today we're announcing that we are closing all of the HTTP ports on api.cloudflare.com. We're also making changes so that api.cloudflare.com can change IP addresses dynamically, in line with ongoing efforts to decouple names from IP addresses, and reliably managing addresses in our authoritative DNS. This will enhance the agility and flexibility of our API endpoint management. Customers relying on static IP addresses for our API endpoints will be notified in advance to prevent any potential availability issues."

So that suggests that people who have been using the Cloudflare API knew that the IP addresses Cloudflare was publishing, where their servers were listening, would never change. And they've decided, eh, we're not going to do that anymore. We're going to, you know, you could look up the IP using DNS. So we're going to allow our IPs to float around. They're saying we need - we, Cloudflare - need that flexibility, so we're going to switch back to using DNS. And of course with DNS over TLS, that becomes - or HTTPS, that becomes more feasible because then you've got your DNS also secured at their end.

So they said: "In addition to taking this first step to secure Cloudflare's API traffic, we'll provide the ability for customers to opt-in to safely disabling all HTTP port traffic for their websites on Cloudflare. We expect to make this free security feature available in the last quarter of 2025." So first they're going to say no to API access over port 80, and then give their customers the option of turning off access to their own Cloudflare-hosted websites over HTTP, again for the sake of enhanced security.

They said: "We have consistently advocated for strong encryption standards to safeguard users' data and privacy online. As part of our ongoing commitment to enhancing Internet security, this blog post details our efforts to enforce HTTPS-only connections across our global network."

I've got a link in the show notes for the entire posting because it goes on in great detail with network state diagrams and like showing how all this works and the problems that can be created if you're not careful and more; and about, you know, how and why none of the options for redirecting initially plaintext HTTP traffic over to HTTPS is able to achieve the same absolute level of security, you know, as simply saying no to all non-HTTPS traffic from the start.

They wrap up this lengthy blog posting by saying: "Starting today, any unencrypted connection to api.cloudflare.com will be completely rejected. Developers should no longer

expect a 403 Forbidden response" - because that means that the server, there was a server listening on port 80 that accepted the connection and then sent back a 403 Forbidden. Now there is no port 80. It's just gone. So, you know, TCP is banging its packets against the wall, and nothing's happening. So they said: "Developers should not expect a 403 Forbidden response any longer for HTTP connections, as we will prevent the underlying connection to be established by closing the HTTP interface entirely. Only secure HTTPS connections will be allowed to be established.

"We're also making updates to transition api.cloudflare.com away from its static IP addresses in the future. As part of that change, we will be discontinuing support for non-SNI" - remember that's Server Name Indication - "non-SNI legacy clients for Cloudflare API specifically." And they said: "Currently, an average of just 0.55%, so a little more than one out of every 200 TLS connections to the Cloudflare API do not include an SNI value." As we know, when you're connecting using HTTP, it is possible for multiple domains to share a single IP because in part of the handshake, in part of the TLS handshake is the SNI value, the Server Name Indication which is the domain to which the client wishes to connect at that remote server IP, the server needs to know that in order to know which certificate to send back in order to match the domain that the client wants to connect to for TLS.

So they said only, you know, one in 200, a little over one in 200 clients are still trying to do that. So they said: "We are committed to coordinating this transition and will work closely with the affected customers before implementing that change." So the other thing they're essentially saying is they're going to be doing some IP space collapsing. Right now they have dedicated IP addresses that are associated with fixed domain names. They don't want to do that anymore. They want to require SNI, and they're going to disconnect that binding between a fixed domain name and a fixed IP so that altogether what this means is they'll be able to serve more domains on fewer IPs. Which, you know, helps with IP depletion problems and gives them a lot more networking flexibility.

So they said: "We're committed to coordinating this transition and will work closely with the affected customers before implementing the change. This initiative aligns with our goal of enhancing the agility and reliability of our API endpoints." And finally: "Beyond the Cloudflare API use case, we're also exploring other areas where it's safe to close plaintext traffic ports. While the long tail of unencrypted traffic may persist for a while, it should not be forced on every site. In the meantime, a small step like this can allow us to have a big impact in helping make a better Internet. We're working hard to reliably bring this feature to your domains. We believe security should be free for all."

So bravo, Cloudflare. This is the sort of step that's needed, as I said above, to push the Internet's security forward. You know, "Just say NO to port 80." Which makes me wonder, I haven't looked, you know, how much port 80 traffic I still have. Our long-time listeners may remember that I jumped on the bandwagon very early in the HTTPS Everywhere move, registering GRC with Google and Chrome so that, I mean, I built into Chrome GRC.com has been there from the start, saying only use SSL - it actually was SSL back then, now TLS - in order to connect to GRC, and feel free to promote any attempt to connect via HTTP to HTTPS because we will always be there answering a secure port.

So anyway, port 80, you know, inherently unencrypted. Got us to where we are today. But for nearly all purposes, everyone is coming to the position that its day has passed. You know, we know from everything we've seen that, inertia being what it is, nothing ever moves forward on its own. It just doesn't. It's always easier to leave things as they are. But a more secure future means that organizations such as Cloudflare need to take a leadership stance as soon as it becomes feasible to just say no to port 80. And for them, they've decided that day is today. So bravo.

Leo: I guess I should turn off port 80 on my firewall and port forwards.

Steve: I'm going to - I think at some point, I mean, not like I don't have other things to do. I do. And like all of us do. So, but I'm curious, you know, how many attempts are being made. For a long time, if you just put GRC.com, for example, in, your browser would try to go to HTTP first.

Leo: Right.

Steve: And then, if I didn't redirect, it would try HTTPS. But I remember, and, you know, I don't know, it was a few years ago, that I remember we talked about it on the podcast, the browser logic flipped. When HTTPS became not only the preferred solution, but by far the majority, you know, Let's Encrypt had been there, certificates were now free, it wasn't, you know, you didn't have Richard Stallman having a seizure because people were saying we want everyone to use a certificate-based connection. And so it was like, okay, it's time. And so the browsers flipped over. I don't, you know, it's probably totally feasible to turn off port 80. It'd be worth taking a look. I'm kind of - now I'm curious.

Leo: Yeah.

Steve: An interesting newly published research paper by researchers out of Greece and the Netherlands caught my attention. Its title is "Coding Malware in Fancy Programming Languages for Fun and Profit."

Leo: Fancy. They call it Fancy, huh?

Steve: It's Fancy. Well, Leo, when you've worn the ink off of your open and close keys...

Leo: Parenthesis keys?

Steve: Parenthesis keys, yeah. This is where you want those two shot key tops, right, where the actual plastic goes all the way down.

Leo: Yeah.

Steve: So, you know, like the fuzz is worn off of the key top, and now it's smooth.

Leo: That's okay, I know where the parenthesis keys are. I have a pretty good idea.

Steve: That's a good point. We don't actually...

Leo: Yeah. I don't need a hint.

Steve: And if you didn't know, it's the fact that there's, like, ink missing from above the 9 and the 0 keys, that would be a clue. But this is a long piece, so let's take another break.

Leo: Okay.

Steve: And then we're going to get into it.

Leo: Good. I always enjoy a good Lisp conversation. You know, I...

Steve: And we're going to talk about the F language, the F word language.

Leo: And there are, yes, there's a number of shall we say "obfuscated" languages out there that are probably very good for that kind of thing, for malware and so forth.

Steve: We're going to have fun with this one.

Leo: Yeah, yeah. I think assembly probably is a better way to go, I'm just saying, to the bad guys. But they probably can't figure it out. That's why. Right? So they're not.

Steve: Yeah. Ease of use and transportability, multiplatform.

Leo: See, Lisp is great, although they're using it for P code, they're using it for intermediate code, not for, well, you're going to get into it. I won't steal your thunder. All right. Let's continue on, Steve.

Steve: Okay. So coding malware in fancy programming languages for fun and profit.

Leo: Okay.

Steve: So I'm going to share the paper's Abstract and its introduction, which will give us a sufficient sense for what these researchers have found. The Abstract explains: "The continuous increase in" - oh, boy, let's get a load of these numbers. "The continuous increase in malware samples, both in sophistication and number, presents many challenges for organizations and analysts who must cope with thousands of new heterogeneous samples daily. This requires robust methods to quickly determine whether a file is malicious." Right? I mean, like just there's so much software now. They said: "Due to its speed and efficiency, static analysis is the first line of defense."

Okay, now, I'll just interrupt here to mention that, broadly, code can either be examined statically, which is just looking at the code bytes themselves after loading them into memory, but not actually running the code; or it can be looked at dynamically, which entails creating a sandbox of some sort, often an industrial-strength virtual machine, to

actually run the code after it's been loaded to examine the code's behavior when it's run. Not surprisingly, static analysis, when you can do it, is much faster and more efficient when that's feasible.

So the Abstract continues: "In this work, we illustrate how the practical state-of-the-art methods used by antivirus solutions may fail to detect evident malware traces. The reason is that they highly depend on very strict signatures where minor deviations prevent them from detecting shell codes that would otherwise immediately be flagged as malicious. Thus our findings illustrate that malware authors may drastically decrease - malware authors may drastically decrease the detections by converting the codebase to less-used programming languages. To this end, we study the features that such programming languages introduce into executables, and the practical issues that arise for practitioners to detect malicious activity."

So essentially, you know, in this ongoing cat-and-mouse, never-ending game of malware and malware detection and avoiding detection and avoiding the avoiding of the detection, here's another domain for escalating this fight which is let's just change languages.

The introduction that they provided gives us some more interesting background. They said: "In the past decade, malware has undergone significant changes. The main drivers of these changes can be attributed to the vast digitization of products and services, and the development of a payment system that allows anonymous transactions to bypass the protections of the traditional banking system." In other words, we've talked about this, cryptocurrency was the enabling requirement for this explosion we've seen because it allows people to make payments, you know, secretly.

They wrote: "This has boosted the number of possible victims and the potential impact of malware," you know, creating a profit motive where there was, you know, viruses used to just kind of exist because they could. Now malware is there to make money. "Moreover, anonymous payment methods enable a wide range of illicit transactions to be performed, which, in the case of malware, is the apparent case of ransomware."

They said: "Both the U.S. Cybersecurity and Infrastructure Security Agency, [our beloved] CISA, and the European Union's Agency for Cybersecurity (ENISA), have recognized malware as the top cyber threat. Indeed, malware attacks impact our everyday lives by harvesting sensitive information, crippling critical services, and causing significant damage to individuals and corporations. This has placed malware in a pivotal role in the crime ecosystem and created an individual ecosystem with independent roles operating in a business model called Malware-as-a-Service," which is not something we've ever seen before, Malware-as-a-Service.

They said: "The security industry's response to the abovementioned threats is collecting and analyzing malware samples." Right? So that's the threat. How do you counter the threat? Well, you need to look at all this stuff. And here was the number that just astonished me. "At a rate of around 280,000 malware samples per day in 2024" - 280,000.

Leo: What. Per day?

Steve: Malware samples per day.

Leo: What? What?

Steve: In 2024. There's just that much...

Leo: All distinct?

Steve: Yes.

Leo: What?

Steve: I know. That's a lot of, I know, a lot of it out there. They said: "Which is more or less similar to previous years."

Leo: Wow.

Steve: Given that load, "static analysis remains the most efficient and profound remedy to detect malicious files quickly." They said: "In this arms race between malicious actors and defenders, the development of malware has evolved into an underground industry." I think what I liked most about this was it gives us a sense of scale. I mean: "The development of malware has evolved into an underground industry to bypass security controls," they wrote, "by employing malware authors and monetizing the infected hosts." In other words, it makes money now. So this is an industry creating malware.

Leo: Unbelievable.

Steve: I know. It's just...

Leo: Unbelievable, wow.

Steve: So they said: "Of course, bypassing static analysis does not grant them a foothold to the targeted host." Meaning more is necessary, but that's the first step; right? You've got to get in before you can do anything. You've got to get past the filter. So they said: "Nevertheless, it significantly raises their chances of achieving their goal, as they then often need to bypass behavioral checks." But static is first. They said: "Although endpoint detection and response systems, you know, EDR as it's now called, Endpoint Detection and Response systems, usually apply such checks, and vendors often portray them as silver bullets, there are several ways to bypass them. In this work, we limit our scope to static analysis. That is, the first stage of prevention is detection through static analysis."

They said: "Even though malware written in C continues to be the most prevalent, malware operators, primarily well-known threat groups such as APT29, increasingly include non-typical malware programming languages in their arsenal. For instance, APT29 recently used Python in their MASEPIE malware against Ukraine; while in their Zebrocy malware they used a mixture of Delphi, Python, C#, and Go. Likewise, Akira ransomware shifted from C++ to Rust, BlackByte ransomware shifted from C# to Go, and Hive was ported to Rust. According to reports, the result of these changes was exhibited increased resistance to reverse engineering and a reduced detection rate or the

malware's misclassification," which is fine with them. You know, adware, okay, we're just not, you know, we're not bad. We're just annoying.

"On other occasions, C-language malware families are not recreated from scratch. Instead, malware authors write loaders, droppers, and wrappers in so-called 'exotic' languages. This provides them with several advantages such as bypassing signature-based detection, so they can effectively wrap their payloads within harder-to-detect shells that are newly built." So it's got a C core, but it's wrapped in something in Rust or Go or Kotlin or something in order to, you know, the static analyzer goes, what? - and then lets it through because it doesn't know that it's bad. Then it unwraps, and the bad stuff comes out.

They said: "Thus, attackers continue to use the same initial penetration vector and a significant portion of their methods, suggesting that threat actors prefer to transfer the original malware code to different languages instead of modifying their tactics, techniques, and procedures" - the so-called TTPs - "to avoid detection. This approach allows them to maintain the effectiveness of their attacks while remaining under the radar of security systems. Since these languages may be less widely recognized or understood, they add an extra layer of obfuscation to malware, making it harder to detect and analyze.

"Furthermore, security analysts have reported increased difficulty in reverse engineering such malware samples due to reprogramming efforts." Meaning, you know, they don't have the tools for reverse engineering some bizarro language. "Thus, combining different languages and obfuscation techniques complicates dissecting and reverse engineering the malware's structure, functionality, and intent.

"Our work," they wrote, "explores the problem of detecting malware written in uncommon languages using a data-driven approach. Rather than merely reporting and examining this trend, we performed a targeted experiment by writing malicious samples in different programming languages and compilers, drilling down to the distinctive characteristics." So they literally implemented their own malware and then wrote it in, like, 40 different languages and then explored what the different AV systems did and why they succeeded or failed. They said: "This analysis practically shows the unique features that adversaries gain and highlights the emerging issues for malware detection and analysis. This work led to the formation of some interesting research questions that have never been answered systematically and studied in the academic literature, and we try to answer them in this work."

So there are three research questions. First, how does the programming language and compiler choice impact the malware detection rate? Second, what's the root cause of this disparity, if any, in detection? And third, are there any other benefits to an attacker from shifting the codebase to less common programming languages and compilers beyond the detection rate by static analysis?

What they learned was quite interesting. As I said, they created their own malware using the top two current malware exploit techniques that have been identified across the industry, and they implemented the underlying malware concept in every language imaginable. Even, Leo, Lisp.

Leo: Finally. We're getting our due.

Steve: That's right. Here's what the - you, too, can write your own malware, if you are not shy of parentheses. So here's what their extensive research concluded, and the answers they arrived at for each of their three research questions. They wrote: "Malware

is predominantly" - I thought this was interesting, too - "predominantly written in C and C++ and is compiled with Microsoft's compiler."

Leo: Interesting.

Steve: Yes. They had a chart. And I mean, it's like 98% Visual Studio; you know? It's like, well, because Visual Studio for us is free.

Leo: It's free, yeah, yeah.

Steve: And so that's what you're going to use. And it's easy and holds your hand and, you know, you don't have to remember anything. They said: "However, answering RQ1 (Research Question 1) with our experiments 'How does the programming language and compiler choice impact the malware detection rate?' our work practically shows that by shifting the codebase to another, less used programming language or compiler, malware authors can significantly decrease the detection rate of their binaries while simultaneously increasing the reverse engineering effort of the malware analysts.

"It is crucial to note that the malware authors do not necessarily need to radically change their codebase, as, for instance, just the choice of" - and this was really interesting to me. "Just the choice of using a different compiler, even for famous programming languages," they wrote, "like C, have the same impact." That is, you don't have to go away from C. You use GCC instead of MSC. They said: "Our experimental results illustrate that there are significant deviations in how programming languages and compilers generate binaries, and that they can serve as an additional layer of obfuscation for malware authors."

So, okay. In other words, since nearly all of the malware code is written in C and compiled using Visual Studio, and they said so in their paper, the static analysis AV detectors that, you know, blanket the industry, have all been similarly oriented or biased toward that assumption because that's what they're seeing; right? Those are the - that's the code that they're chartered with blocking, detecting and blocking. So simply by switching to Turbo C, or GCC, or WATCOM C, those assumptions about the specific binary code bytes that are being produced will be broken, and AV detection rates will drop without any need to rewrite their malware.

And as I was reading this, it occurred to me, I'm not sure it was good for this paper to be published. But, you know, I guess it was true either way, whether they published it or not, because as they said at the top they're already seeing malware moving to other languages. And the only reason any bad guys would move from a comfortable C programming environment over to Rust is specifically for detection rate avoidance because they've already got, you know, their malware written, and they don't want to do work they don't have to do.

The researchers' Question 2 asked about the root cause of the disparity. They wrote: "The root cause for the disparities that we raise in Research Question 2, as highlighted with our use case in Haskell and the metrics for each tested pair of programming language and compiler, is that there are radically different ways that each of them reaches the same result. For instance, different ways of storing strings and different approaches in the internal representation of functions can render many static detection rules useless. As a result, there is no one-size-fits-all approach, so further research is necessary to systematically identify these differences and group them."

You know, the short version is AV is about to get a whole lot more difficult to do because the bad guys are no longer sticking exclusively with C and Visual Studio. Essentially they're saying that, since static code analysis is constrained to simply examining code that's lying there in RAM, things such as function-calling methods which pass parameters in different ways, or static strings that are stored and represented in differing ways, all of which will vary by language, all serve to dramatically confuse status analysis. It might result in false positives, and so they're wanting not to be overreactive. But it's just as likely when it's confused to allow bad code to slip past.

And answering their final research question: "Are there any other benefits to an attacker shifting the codebase to less common programming languages and compilers beyond the detection rate that's used by static analysis?" they said: "Answering Question 3, this shift in languages may come with additional benefits for attackers. An obvious case is cross-compilation and multi-platform targeting languages, which enable malware authors to build a single malware variant and have it compiled for multiple operating systems."

Leo: Not to mention you getting rid of all those buffer overflows in your malware.

Steve: Actually, yes. I make the point a little bit later, they're getting more reliable malware. Oh, great. If they use Rust, yes. "The strategy can significantly reduce the time," they wrote, "and number of tools needed to achieve their objectives, thereby expanding the scope of any hostile campaign. IoT devices, in particular, support a range of CPU environments, making it necessary for malware targeting these devices to be compatible with not only x86 and x64 architectures, but also various other architectures such as ARM, MIPS, m68k, SPARC, and SH4." You know, various microcontroller architectures. Much lower end processors that are being used in IoT devices.

"A typical example is Mirai, which uses GCC; yet one of its successors, NoaBot, uses uClibc-based cross-compiler and is statically built to target embedded Linux systems. In this regard, other options could be more efficient. For instance, Go can be cross-compiled to all major operating systems, as well as Android, JavaScript, and WebAssembly. One of its advantages is that it provides statically compiled binaries by default, eliminating runtime dependencies and simplifying deployment on target systems." Oh, great. Just what we want for the malware. "Go also features a robust package ecosystem that allows developers" - malware developers - "to easily pull in code from other sources." Yeah. Basically, you know, we've made programming much better for legitimate developers. And unfortunately, the malware authors benefit, too.

Leo: Yeah. Honestly, that's what's happening is these are all benefits to modern programming languages.

Steve: Exactly.

Leo: Yeah.

Steve: Exactly. And they said: "As a result, malware can be developed at a faster rate" - oh, joy - "targeting a broader range of architectures and systems. Indeed, HinataBot, another descendant of Mirai, is developed in Go to take advantage of the above. HinataBot's discovery was much more difficult as a result. Unfortunately, the bar to creating a new variant of Mirai using Go or other languages is now quite low. This allows"

- get this, Leo - "criminal groups to create their own variations." So that's one of the reasons there's just so much of it. It's like, you know, oh, let's just, you know, tune it and tweak it for our own needs.

Leo: Its Fancy Bear Mirai.

Steve: Exactly, yes. "Beyond cross-compilation," they said, "there are several other reasons to witness more changes in the malware codebase. After all, malware developers, like any other software developers, have specific needs when choosing programming languages and tools. Different languages offer various benefits for different scenarios, and the choice of language can significantly impact the development and functionality of malware. For instance, built-in security mechanisms and type safety may be prioritized by ransomware authors who want to avoid leaks of the encryption keys to guarantee that" - oh, my god - "to guarantee that their victims will not be able to develop decryptors."

That's right, we want to scrub the RAM so we don't leave secrets behind, not because we're the good guys trying to protect our keys; but because we're the bad guys, and we just encrypted everyone's database, and we want to make sure they don't get a hold of our decryption key. Wow. They said: "A typical example is Rust, which offers built-in memory mechanisms to prevent common vulnerabilities and to offer type safety." So even malware is now benefiting from the enhanced memory management and security created through the use of more modern and safe languages. That's just wonderful.

They wrote: "Other aspects can include library availability; facilitating interaction with the underlying operating system and enabling critical malware functions, low-level access, and control over memory layout; having full control over the malware's behavior and performance, but also direct compilation to machine code; creating an executable file directly and use other tools for obfuscation." So exactly as you said, Leo, everything we've done to make languages better for the good guys has made it better for the bad guys.

Leo: Wow.

Steve: They said: "While shifting to another programming language may seem complicated, especially when considering less popular ones, large language models (LLMs)" - oh, boy.

Leo: They do a great job with Lisp, yeah.

Steve: In other words, AI.

Leo: Yeah.

Steve: "May come to the rescue." They said: "After all, they've proven their capability for generating code quite accurately, and various cybersecurity tasks, and malicious actors are abusing them. As a result, AIs can translate code from one programming language to another, requiring little fine-tuning. Don't even have to understand the language that the

AI produced. This way, malware authors can seamlessly develop loaders, droppers, and other components in languages they may not be familiar with.

"It's true that the malware that we examine in this work represents a small fragment of the total; nevertheless, it is stealthier and introduces more bottlenecks for the reverse engineer. Given that the APT groups are shifting their codebases, and the malware-as-a-service model facilitates the trading of malware so different malware mixtures per campaign can be purchased, this diversification is expected to continue." And they finish: "By disregarding these samples and only focusing on traditional programming languages and compilers, we provide malware authors with an effective hideout they can easily exploit. Therefore, we believe that a deeper analysis of the executables produced by other compilers and programming languages is needed to improve detection rates, but also develop better reverse engineering tools."

So what we are now seeing is that, you know, the bad guys are noticing that the AV tools are blocking them right and left. And so they're saying, okay, fine, didn't want to, but we will, you know, change compilers, change languages. And of course this just makes the detection rate go exponential because now the code could be coming in under any language other than as it used to be, basically all C. You know, a given like Mirai would be written in C. So the detectors would learn to detect the various variants of Mirai, but only under C. Now Go, and all these other things. So, you know, I wouldn't say...

Leo: I'm kind of - don't they all compile down to assembly and, I mean, machine language?

Steve: Yes, they do, except that as long as they're written all under the same compiler, that compiler is going to...

Leo: The style.

Steve: Is going to translate the same source into the same bytes.

Leo: Right.

Steve: Right. And so the static analysis that doesn't actually run the code to see what it does...

Leo: Oh, it's just like string compare almost; right?

Steve: Yes, it is. It is a signature comparison.

Leo: Oh, okay. And of course that doesn't work, yeah.

Steve: And so anything you do - right. So, yeah. So I wouldn't say that they've discovered anything earth-shattering or surprising. Their results are pretty much what we would expect.

Leo: Yeah.

Steve: But some of the tricks they highlighted, such as simply recompiling unchanged source malware under a different compiler for the same language was interesting. You know, just change from Visual Studio to GCC, and you get different code which will break the signature comparison, and you didn't have to rewrite your source at all.

Leo: Yeah.

Steve: So by clearly demonstrating in fact what we might assume, their work should serve to get the authors of the static AV detection to, you know, I'm sure they must be looking at this thinking, oh, god, you know, I mean, it's going to be, what, 20 times more signatures that they need, given all the compilers and the variants of compilers that are available?

Leo: Oh, yeah. And that's without changing languages.

Steve: Right. And in reading this, the one language I didn't see, which would have been really interesting actually, was Forth. Based upon what these researchers found, I would imagine that Forth would have a number of advantages for malware. For one thing, it only needs a very small and readily available runtime interpreter that's has already been ported everywhere. And I often refer to Forth as a write-only language. We've talked about it before.

Leo: It doesn't have to be.

Steve: Oh, Leo, it does. It does. No, really.

Leo: Well, it doesn't because you're creating a dictionary. You could make it almost English-like if you worked at it.

Steve: Well, you can make your verbs English-like, but it is a dense, stack-oriented language, so you are the compiler.

Leo: That's a good point. Where you're getting the data from is very obscure because it's popping and pushing it.

Steve: Yeah, and so, I mean, so as you're writing it, you know to put this on the stack, put this on the stack, put this on the stack, then call the verb.

Leo: Right. That's non-obvious, yeah.

Steve: You're the compiler. And so, yeah. You know, you come back and look at something you wrote a month ago, it's like, what does this do? I mean, it's...

Leo: I loved Forth. I really loved it.

Steve: I do, too. It is a beautiful, elegant, tiny language. And I hope I didn't give the bad guys any ideas. On the other hand, it's not easy to use.

Leo: No, they're never going to use that. It's too much - the learning curve's too steep.

Steve: Yeah.

Leo: Although there is an excellent book called "Starting Forth" that it's just one of the best programming books ever written. That's actually how I got into it is I read Leo Brodie's "Starting Forth." And that actually is such a beautifully good book that I couldn't resist.

Steve: And it's just fun to play with.

Leo: It's fun.

Steve: Yeah.

Leo: And eventually your program is one word.

Steve: "Do," or "do it," or "go."

Leo: Yeah, do it, go.

Steve: Actually, normally the verb is the name of the program.

Leo: Right.

Steve: So it's just, you know...

Leo: Program, yeah, yeah.

Steve: Sort or something.

Leo: And it does not compile to assembly, though. It is a kind of a bytecode interpreter.

Steve: It is, yeah. So it has - but it is so lean that the runtime is extremely small. I mean, it is...

Leo: Right. It was written for telescopes and that kind of thing.

Steve: Yes, it was originally Charles...

Leo: I interviewed him. Moore, Charles Moore.

Steve: Moore.

Leo: And I remember interviewing him, this is back at TechTV, because I was a fan of Forth. And he was puzzled. He said: "I never thought anybody would want to talk to me." But he was brilliant, and it was for very small embedded environments like telescopes. That's why I think it still may be used in robotics and things like that. It's great for robotics.

Steve: Actually, it's in some motherboards. There are some motherboards that are using Forth as their engine for, like, getting systems booted.

Leo: So there are some hackers who still know Forth. That's interesting.

Steve: Yeah. We have another piece from Cloudflare, but let's take a break.

Leo: All right.

Steve: We're at an hour and a half.

Leo: That we can do.

Steve: And then we're going to look at the continued reuse of passwords, despite all advice.

Leo: Now you're making me want to go back and write some more Forth.

Steve: I know. I'll bet our listeners are like, Forth? And, yes, it is available everywhere.

Leo: Oh, yeah.

Steve: You can easily find a cute little interactive Forth for Windows.

Leo: Except for the Mac because the problem is it's so old, nobody's written Forth for Apple silicon, as far as I know. In fact, a lot of the Forth stuff was written for PowerPC and was never ported to Intel. So there was a great Mac Forth back in the PowerPC days that was wonderful. But I don't know today. I guess you could just run it in a VM. It's so tiny.

Steve: It is. It's a small runtime.

Leo: I think I - here, I have the book. I'm running over to my bookshelf and holding up my gently thumbed copy of "Starting Forth." This was such a good...

Steve: Yes, I recognize the cover.

Leo: Did you read this book?

Steve: Yup.

Leo: You probably didn't need to. You did?

Steve: I recognize the cover.

Leo: Oh, yeah. It had great cartoons in it. It was just a wonderful - Leo Brodie. And he was working at Forth Incorporated when he wrote it.

Steve: Yeah, Manhattan Beach, I think, is where Forth was located.

Leo: Wow. Oh, my gosh. Yeah, this is so old. It's not Courier, it's just - it's a typewriter.

Steve: Oh.

Leo: That ain't Courier, folks. That's just a photostat of a typewriter.

Steve: And we can't do bold, so we do underline.

Leo: Right. Oh, but this was such a clearly written book, and he had such a great sense of humor. Here's his explanation of how the stack, how slicing the stack works.

Steve: With a little samurai.

Leo: A samurai. And then there's a rabbit popping numbers off and on the stack. It's great. Anyway.

Steve: Okay. So once again, Cloudflare recently published a piece of research that I wanted to share. I was initially confused by the headline of their blog post, which read: "Password reuse is rampant. Nearly half of observed user logins are compromised." And I thought, what do you mean "nearly half of observed user logins are compromised"? It turned out that the problem with their headline was the somewhat unclear word "compromised." A better choice may have been to say "nearly half of user logins use previously leaked passwords."

Leo: Yes, right. They've been compromised, yes.

Steve: Compromised in the sense of got out loose. In other words, passwords that are likely known by Troy Hunt's Have I Been Pwned site. Cloudflare wrote: "Accessing private content online, whether it's checking email or streaming your favorite show, almost always starts with a 'login' step. Beneath this everyday task lies a widespread human mistake we have still not resolved: password reuse. Many users recycle passwords across multiple services, creating a ripple effect of risk when their credentials are leaked.

"Based on Cloudflare's observed traffic between September and November 2024" - so three months, one quarter of 2024. Get this: "41% of successful logins across websites protected by Cloudflare involve compromised" - meaning leaked, previously leaked - "passwords." 41% are people are logging in with passwords that have already been leaked out on the Internet. And they said: "In this post, we'll explore the widespread impact of password reuse, focusing on how it affects popular Content Management Systems, the behavior of bots versus humans in login attempts, and how attacks exploit stolen credentials to take over accounts at scale."

I'm going to skip over most of this because everyone listening, I know our audience, everyone listening to this podcast already well understands the dangers of password reuse, and I'm sure that everyone listening is now using some form of password manager which is able to synthesize complete gibberish passwords, which is what we want, on the fly for use, then store and later reuse.

One thing I wasn't appreciating before this was the size to which Cloudflare has quietly grown. At one point in their blog posting they wrote: "Our data analysis focuses on traffic from Internet properties on Cloudflare's free plan, which includes leaked credentials detection as a built-in feature." So that's something they offer their free plan users. "Leaked credentials," they wrote, "refer to usernames and passwords exposed in known data breaches or credential dumps. For this analysis, our focus is specifically on leaked passwords. With" - get this, Leo. "With 30 million Internet properties comprising some 20% of the web behind Cloudflare, this analysis provides significant insights."

Cloudflare is one fifth of the Internet. Thirty million Internet properties. They're just been quietly growing since they were a cute little startup that we used to talk about. Oh, you're so cute, you little startup, you. Holy crap. One out of every five sites is now running their traffic through Cloudflare. Well, that crept up on us.

So they explain: "One of the biggest challenges in authentication is distinguishing between legitimate human users and malicious users. To understand human behavior, we focus on successful login attempts, those returning an HTTP 200 OK status code, as this provides the clearest indication of user activity and real account risk. Our data reveals that approximately 41% of successful human authentication attempts" - okay,

successful. "41% of successful human authentication attempts involved leaked credentials."

Leo: That's kind of amazing. How does Cloudflare know that?

Steve: Well, because they've got all of Troy's...

Leo: It's going through them.

Steve: Right, exactly, because it is coming through them, and they're able...

Leo: So a huge proportion. What did you say, a third of the 'Net is behind a Cloudflare wall, in effect.

Steve: Right. And so they're able to see...

Leo: So they could see those passwords in transit.

Steve: Yup.

Leo: Wow. Even on SSL they can see them in transit. Huh.

Steve: Well, they're hosting the site, so...

Leo: Oh, yeah, yeah. Yeah, yeah, they'd have to.

Steve: So they're the server that is actually receiving the password.

Leo: Oh, so we're not talking about Cloudflare's like protection against DDoS.

Steve: Right.

Leo: They're actually hosting. They host that much of the web?

Steve: Yes.

Leo: What?

Steve: That's what astounded me.

Leo: It's because they have free pages.

Steve: Thirty million sites.

Leo: That's kind of amazing.

Steve: Thirty million sites.

Leo: Wow.

Steve: Yeah. So they said: "Despite growing awareness about online security, a significant portion of users continue to reuse passwords across multiple accounts." And they're watching people logging in with passwords with credentials that have been leaked that are known. They said: "According to a recent study by Forbes, users will, on average, reuse their password across four different accounts," in four different places. So it's my password, uh-huh. "Even after major breaches, many individuals don't change their compromised passwords, or still use variations of them across different services. For these users, it's not a matter of 'if' attackers will attempt to use their compromised passwords." They will.

Leo: They will.

Steve: It's a matter of when.

Leo: They will, yeah.

Steve: And they note, as we would expect, automation, in the form of bots, are the primary abusers of leaked credentials, just like Troy Hunt got phished by an automated attack which was able then thereby to bypass his one-time password. Didn't matter that he had a password, you know, a six-digit token that was going to expire in 30 seconds. Didn't even take 10 seconds.

So they said: "Bots are the driving force behind credential-stuffing attacks. The data indicates that 95% of login attempts involving leaked passwords are coming from bots, indicating that they are part" - a big part - "of credential stuffing attacks. Equipped with credentials stolen from breaches, bots systematically target websites at scale, testing thousands of login combinations in seconds. Data from the Cloudflare network exposes this trend, showing that bot-driven attacks remain alarmingly high over time. Popular platforms like WordPress, Joomla, and Drupal are frequent targets, due to their widespread use and exploitable vulnerabilities.

"Once bots successfully breach an account, attackers reuse the same credentials" - because that just validated the credential. "Attackers reuse the same credentials across other services to amplify their reach." That is, oh, if it's good here, then it's probably going to be good somewhere else. So they do that immediately. I mean, so like no stone has been left unturned by the bad guys. They're as clever as we would be if we were the bad guys, like, trying to figure out how to maximize our badness. They said: "They even

sometimes try to evade detection by using sophisticated evasion tactics, such as spreading login attempts across different source IP addresses, mimicking human behavior, attempting to blend in with legitimate traffic. The result is a constant automated threat vector that challenges traditional security measures and exploits the weakest link: password reuse."

Okay, now, purely by coincidence, one of our listeners, Jeremiah Albrant, sent a piece of feedback to me yesterday with the subject "Microsoft/Hotmail account password stuffing attempts are very real." In his email he said: "Talking to some co-workers, they showed a screenshot of their sign-in activity from their Microsoft account, so I checked mine." He said: "I was blown away. My own screenshot is below. The one successful attempt," he said - I know, Leo. It's so bad.

Leo: Holy moly.

Steve: He said: "The successful attempt is my own. Clicking through each unsuccessful attempt shows they entered the wrong password. I am so glad I use unique passwords for my accounts. This is nuts."

Leo: And lookit, Mexico, Morocco, Saudi Arabia, Russia, Indonesia, India, Vietnam, Uzbekistan, Oman, Ethiopia, Jordan. You know, did I mention this? I put an SSH server out in public briefly. And I don't use passwords on my SSH server. I use a...

Steve: A certificate.

Leo: ...certificate. So I wasn't too worried about it. Within two hours, and I put it on port 22 because, you know, you can sniff the ports. It doesn't matter what port it's on. So I put it on the canonical port. Within two hours I had a dozen attacks from Albania, from China. They were sniffing around for an SSH server on port 22 and then started hammering it. Within two hours of it going up. It's amazing. They're out there, man. They're crazy.

Steve: Yeah, they really are.

Leo: Are they using Shodan and stuff to find this?

Steve: No, there is, you know, I coined the term 20 years ago, IBR, Internet Background Radiation.

Leo: Internet Background, yeah, yeah.

Steve: Maybe even when you and I were on Screen Savers at TechTV before.

Leo: I think so, yeah.

Steve: Because that's, you know, that's what I was seeing when I was looking at IPs that nobody had any business poking at. There were packets inbound, sniffing for stuff. It was just - that was just out there.

Leo: And this IP address hadn't been public in at least a couple of years. They just found it right away. It's unbelievable, yeah.

Steve: Jeremiah's email finished, just for the sake of our listeners...

Leo: Oh, sorry.

Steve: No, no, no, this is good stuff, Leo.

Leo: It got me warmed [crosstalk].

Steve: For the sake of our listeners, he says: "If others want to see their history, I clicked on my avatar in the top right from my inbox, then 'My Profile,' then the 'Security' tab, then 'View my sign-in activity.'" He said: "Unfortunately, the UI is primitive and doesn't seem to have filter or sorting options. So unless I click the 'View more activity' link over and over while expanding each item, I don't see any other way to determine whether somebody has my password and just failed to get past two-factor authentication." He says: "In other words, it's necessary to expand each attempt to determine the cause of the login failure."

Okay, now, I'm glad you put that on the screen and you had the reaction, Leo, that I had when I saw that. His login log shows about five attempts per day, every single day. At the top we see his one successful login showing its location in the United States, where he actually is. Three hours before that was a failed attempt made from an IP address in Mexico. An hour before that, from Morocco. Six hours before that, from Saudi Arabia. The previous day attempts were made from, as you noted, the U.S., Russia, Indonesia, India, and Vietnam. And the day before that we see Uzbekistan, Oman, Ethiopia, and Jordan.

Given that, the most obvious security feature for Microsoft to implement would be account access geofencing. But my quick search revealed that, not only is there massive demand for this from everyone, anyone who has ever looked at that page that Jeremiah did says, hey, what? So there's massive demand. But it's only available from Microsoft for business class accounts, not for individual users.

Leo: Ah.

Steve: And I have to say that's difficult to explain since anyway examining their history of failed authentication attempts should be infuriated by their inability to block all such obviously bogus authentication attempts from across the globe. You know, as I said, I have no doubt that, just like Jeremiah, all of our listeners are using unique gibberish passwords with the help of a password manager. But really, you know, make sure you are, and definitely you want second-factor authentication used wherever it is offered.

That said, we all know that most of our friends and family are not listening to this podcast. So this amounts to a gentle nudge reminder for us to proactively annoy all of

them about this. It would just be for their own good. Make sure that they're doing this. As you saw from bringing up a SSH server, as we can see from this login log...

Leo: They're just out there, yeah.

Steve: It's just, yeah, it is.

Leo: What do you think it is? Are there hacker farms that are just constantly at work, or what?

Steve: They must succeed enough that it is worth their time. It's like, why is there spam? Enough people click on the link for the furry bunny before Easter from China that it, you know...

Leo: You got that one, too, huh? So, and it's probably automated. I would imagine it's completely automated.

Steve: Oh, yeah. It's just set up, and it runs 24/7. And as new breaches occur, they just pour that new data into their database and start pounding on new username and passwords that have been leaked.

Leo: Yeah. So they've got the breaches. They download the database. And then they just fire away.

Steve: It just runs and runs and runs, just grinds away.

Leo: Wow.

Steve: Bandwidth doesn't cost anything, so they just pound. And, I mean, and again, the idea that Microsoft is not offering an "are you in Uzbekistan" block is ridiculous. You know, you could turn it off when you're going to go take a trip to France or Mexico or somewhere.

Leo: Right.

Steve: But it ought to be on. This guy, Jeremiah, our listener, should not have Microsoft thinking, hmm, is that him?

Leo: Is he in Morocco all of a sudden?

Steve: Yeah, he must have teleportation because he got from...

Leo: But honestly, they can easily spoof their location. It's not going to be, I mean, in fact, I'm surprised they show that they're from China. Right? I mean, why bother?

Steve: I guess that's a good point. They don't bother spoofing locations because they know Microsoft isn't checking.

Leo: Nobody's checking. Darren in our Club TWiT Discord said, "A few months ago at work" - I gather he works for, I remember he works for a financial institution - "we had a thing where people were using our site as a vector for checking credit cards. They used some bot to go to the payment page, then tried to purchase, get this, with tens of thousands of different cards. They had a database of cards, right, of breaches. They got maybe 30 successful purchases."

Steve: Wow.

Leo: But what was interesting, they started very naively, always with the same details. And then things, as they locked things down, started changing, and they eventually made their way in. They couldn't, even with a geographic block, they couldn't find a way to stop people from doing this. And then he said eventually it just stopped, just like DDoS attacks, and they moved on to some other site that they could do the same thing with. That's why rate limiting is also really important; right?

Steve: From day one I built strict blocking into GRC's ecommerce system. Sometimes users have a problem, and they say I'm sorry, but I'm just told I've been trying too many times to get my card to get clear, and so they'll write to Sue, and Sue says, okay, you know...

Leo: Yeah, we'll do that, that's fine.

Steve: You know, we'll - give me the information, and I'll do it for you. But because I just, like, I'd rather, you know, say no to people that are going to do that maliciously.

Leo: But this, he says, this is why people have reCAPTCHAs on their sites, because that basically slows it down enough that it's not economical for them to continue.

Steve: Wow.

Leo: Wow.

Steve: So just a quick follow-up on last week's mention of 23andMe. I ran across a bit more information in a security newsletter. Under the headline "23andMe files for bankruptcy after mega-hack," it said - and I didn't cover this last week. It said: "DNA and genetic testing service 23andMe has filed for bankruptcy" - that we know - "15 months after experiencing a major data breach. The company has been losing money for years, but its problems were amplified last year after a series of class-action lawsuits

related to the breach. Its entire board resigned last year, its CEO last week, and the company is now attempting to sell itself under the supervision of a court."

The company has DNA profiles on over 15 million users. Privacy regulators across the U.S. and Europe are now urging users to request the deletion of their data before it's sold. And I did mention after I wrote this and before now, I saw another bit of news saying that a court just approved the inclusion of its members' DNA data in the bankruptcy service.

Leo: Oh. So they can sell it now. Oh.

Steve: Yes.

Leo: Okay, now I am going to delete it, yeah.

Steve: So I'll just remind our listeners that once you log into your 23andMe account, you can use the shortcut I created last week, grc.sc/byebye.

Leo: Bye-bye.

Steve: B-Y-E-B-Y-E. And that'll immediately jump you to the page containing the various account data dumping and deletion options.

Leo: That's funny. When I log in, they're still trying to upsell me. Now it's some sort of heart health thing.

Steve: God, yeah. So again, not a house-on-fire issue; but the judge, a court did say yes, those are your assets. Genetic data which your members gave you voluntarily is yours to sell. So it's going to be of use to somebody. I would just say bye-bye.

Okay. Now, today's Shortcut of the Week - oh, Leo. You probably want to go there while I'm talking about this, grc.sc/1019. I was pursuing information about a new-on-the-scene ransomware group calling itself Arkana, A-R-K-A-N-A. Arkana's first victim was WoW, one of the largest ISPs in the U.S. Ransomware hit WoW, this large U.S. ISP. But in following some trails, I ran across a site I had never seen before and which we've never talked about. It's ransomlook.io. So you can also go https://www.ransomlook.io, or just grc.sc/1019. The site's been around since 2022. They're on Mastodon and Bluesky, and a huge amount of work - that is, you're now looking, you're scrolling, Leo, through a list of actual...

Leo: These are all from today.

Steve: They're victims of ransomware attacks today.

Leo: Today.

Steve: Yes.

Leo: And then here's yesterday.

Steve: Yes.

Leo: Oh my god.

Steve: It is very...

Leo: These are victims. These are not people under attack. These are people who've actually been encrypted.

Steve: Yes. Yes. They are victims.

Leo: And some of these names I recognize. These are well-known companies in some cases.

Steve: I know. Once you get to the homepage, under "Group Profiles," you'll find listed there every group we've ever talked about and hundreds more lesser groups or newer groups that we haven't yet. And there are familiar names. The "Ransomware Notes" section lists all of the various notes that the ransomware groups have sent to their victims.

Leo: Oh ho ho ho. And by the way, they're getting much more grammatical.

Steve: Yeah, thanks to AI, yup. And chilling, most chilling of all is what you started with, that "Recent Posts" page which contains a listing in reverse chronological order, starting with the most recent, of the latest ransomware victims and which group took them down. When I was writing this at 3:00 p.m. yesterday, there were 22 new ransomware victims listed, just for March 31st, yesterday, by name. And I don't even know what time zone they're in, so I don't know when they started March 31st. But listed there in black and white are the corporate names and domain names of many victims. And there's just no way to come away from a perusal of this site without the very clear knowledge that the ransomware category of criminal cybercrime is very much a going concern.

Leo: How do they get - because some of these companies, many of these companies don't want anybody to know they were hacked.

Steve: Right.

Leo: How do they get these names?

Steve: From the postings of the ransomware.

Leo: Oh, the ransomware people announce it.

Steve: Yup.

Leo: Of course they do.

Steve: There was an Irvine-based architecture firm that I clicked on yesterday, and it brought up their home page, that is legitimate. And then I looked at some of the data, some samples of the data that had been exfiltrated, and it was architectural drawings by this firm, this major architectural firm, from yesterday. It's like, uh, whoopsie.

Leo: Oh, boy. This is a great site.

Steve: Isn't that great?

Leo: This RansomLook. Wow. Wow.

Steve: Yeah.

Leo: Just the recent posts alone is...

Steve: I know. It's just astonishing.

Leo: This is today.

Steve: Yes.

Leo: Hospital. Pharmaceuticals. FancyFilms.com. I mean, unbelievable.

Steve: And this also, talk about an example of the security problems we still have in this industry. What is Goosehead.com?

Leo: Well I think it's getting worse; isn't it?

Steve: Yeah.

Leo: This must be getting worse.

Steve: Goosehead.com.

Leo: How about JackpotJunction?

Steve: RansomHub got them.

Leo: Yeah, yeah. Kyocera Document Solutions Europe. Okay.

Steve: Kill Sec 3 took them down.

Leo: Unbelievable.

Steve: I know. Wow.

Leo: Boy, if you're a CISO, this has got to be terrifying. Just the worst.

Steve: And if you are a CIO who needs to get some money from your CFO...

Leo: Yeah, show them.

Steve: Just go - yeah, huh?

Leo: You know, this is a problem. We hear this again and again, that IT, especially cybersecurity, is not a profit center, it's a cost center. And they just want to cut it. Look what they just did to CISA. It's not a profit center. Doesn't make them money. So, okay, well, we don't really need it.

Steve: Right.

Leo: Wow. What a great site. That is an eye-opener.

Steve: It is a sobering look at reality.

Leo: Yeah. Ransomware.

Steve: And I have one piece of listener feedback I wanted to share. Just a reminder about InControl. Ben Dean from the UK wrote: "Hi, Steve. Just thought I'd send you a quick message to let you know how thankful I am for your incredibly useful little program InControl! I'm an avid flight simulator enthusiast, and the best way to enjoy flight simulation these days is with a high-end VR Headset." Wow, I can imagine.

Leo: Yeah, no kidding, yeah.

Steve: As long as you don't get air sick. "As such I have an HP Reverb G2 V2 headset, which when new in 2021 was several hundred pounds or dollars. This headset uses..."

Leo: Oh, I thought he was talking weight. Okay. I don't want to wear several hundred pounds.

Steve: And a big screen pulling it up.

Leo: Okay.

Steve: "This headset uses Microsoft's 'Windows Mixed Reality' platform, which is built into Windows. While the headset itself is excellent, the WMR platform (Windows Mixed Reality) was somewhat of a failure for Microsoft, with most other manufacturers using other platforms. Despite that, MANY" - he has in all caps - "people in the flight sim world still use the Reverb G2 with Windows Mixed Reality because of its high resolution.

"In their infinite wisdom, Microsoft have decided to remove Windows Mixed Reality from Windows 11 from update 24H2, rendering all WMR headsets like my HP Reverb completely useless. Indeed, friends of mine have had the update only to find their VR headsets no longer work, and they have to go through the huge hassle of somehow stepping back to 23H2 to get their setups working again. Thankfully, with InControl we can stay on 23H2 and retain the WMR functionality. I've recommended InControl to several of my friends, and it seems to do the trick of MS forcing them to update against their will. Sorry for the long email, but many thanks for your work. Cheers, Ben Dean, UK."

Leo: Nice. Very good.

Steve: So just a little reminder to our listeners. It's there, it's free, and it works.

Leo: InControl. Do not upgrade. Although October 25th Windows 10 goes out of update.

Steve: Yes, it does.

Leo: End of Life. You don't care.

Steve: Ask me if I care.

Leo: You don't care, do you.

Steve: No, I never forget how much you laughed, Leo, when I announced my creation of Never10.

Leo: Uh-huh.

Steve: I mean, I gave it that name. I described it, and I said it was called Never10.

Leo: Never.

Steve: And then they went to 11, and then I thought, okay, it's not going to be Never11. That doesn't sound good anyway. So it's InControl. That way we're ready for 12 when it comes along. And Lucky 13. I bet that's going to be a winner. So our last break, and then we're going to talk about the EU OS.

Leo: Yeah. That's fascinating. Although, as you point out, it probably stands on the shoulders of open source giants and so - but we'll see.

Steve: The question is will it crush them.

Leo: Yeah. Those shoulders are broad, but there's a lot of people sitting on them.

Steve: Those shoulders are taken for granted.

Leo: All right. Let's talk about the subject at hand, the EU OS.

Steve: So, yes. Robert Riemann is the Head of Sector for Digital Transformation in the Technology and Privacy Unit at the European Data Protection Supervisor in Brussels. He contributes to the overall IT governance of the EDPS, which is European Data Protection Supervisor, and supports the EDPS representation in several EDPB subgroups. Whatever that is. Something, oh, the data protection supervisor in Brussels. So his CV indicates that he holds a Ph.D. in computer science with a thesis on distributed protocols for aggregation of confidential data with applications - so he's a serious comp-sci guy - in, for example, online voting. And he also has his Masters in Physics from Berlin's Humboldt University. So he's the guy.

As the title of the podcast "EU OS" suggests, Robert is spearheading a well-thought-out departure from EU's dependence upon Microsoft Windows. The site where this is being organized calls itself the European Union's home for their free public sector personal computing operating system, highlighting three key features of the project: Secure, Sovereign, and Sleek. I guess we wanted three S's. So Sovereign...

Leo: Yes. Secure. I like this. This is good. They've got an ad man writing their copy. That's good.

Steve: Yeah. Secure, Sovereign, and Sleek. Secure means an OS built from open source.

Leo: Yes.

Steve: And they said "that does not phone home."

Leo: Yes. That eliminates Microsoft. Okay, go ahead, yes.

Steve: Uh-huh. Sovereign means an OS built to the requirements for the EU public sector, meaning, for example, it inherently honors...

Leo: GDPR, yeah, yeah.

Steve: GDPR, exactly. And Sleek means an OS that is fast and eco-friendly on new and old hardware. So obviously, none of those goals - sorry, Microsoft.

Leo: Sorry.

Steve: None of those goals are met by Windows.

Leo: No.

Steve: On that home page they ask the reader the question: "What is EU OS?" And their answer is: "EU OS is a proof-of-concept for the development of a Fedora-based..."

Leo: Oh, interesting.

Steve: "...Linux operating system with a KDE Plasma desktop environment in a typical public sector organization. Other organizations with similar requirements or less strict requirements may also learn from this proof-of-concept. Despite the name, EU OS is technically not a new operating system. DistroWatch lists currently over 250 Linux operating systems (distributions), not counting their many various flavors, spins, or subvariants. The added value of EU OS is a different one. First, a common Linux OS as a base for all EU OS users with options to layer on top modifications at the national layer, the regional, or sector-specific layer, or organization-specific layer" - you know, different configurations is what he means - "a common desktop environment; and a common method to manage users and their data, software, and devices."

The site is at eu-os.gitlab.io, which endeavors to fully articulate the goals of this initiative. Again, eu-os.gitlab.io.

And they said: "When at the beginning, the user base is too small to pool sufficient resources to take care of the EU OS" - that is, the base version - "within the public sector, it may be possible to contract commercial support for maintenance." That is, like until they can, like, generate their own internal maintenance organizations to support it. "For this reason, the EU OS proof-of-concept proposes to choose an upstream Linux OS with options for commercial support. EU OS is not the first to propose a Linux-based

operating system for the public sector. The motivation is often the same and can be looked up from projects like GendBuntu and LiMux. And those are 'public money - public code' means the public investment profits the entire public and the private sector.

"Synergy effects lead to tax savings because there's no per-seat license cost. Independence from software suppliers and vendor lock-in. Independence in scheduling software migrations and potential hardware upgrades." Ugh. Windows 11, anyone? "Deploy new technologies with controlled cost. Use of open standards to foster innovation. Better use of IT administration resources." Then he says: "(Reportedly for the French use case with 90,000 seats). Ability to do own code analysis." In other words, open source, not closed, not proprietary. "Worldwide free software community."

And then the project lists its philosophical goals as "the use of open source, the use of desktop environment KDE Plasma." And then it says "(though Gnome as an alternative is not excluded), and the use of GitLab." They're leaving the entire scope of the project somewhat open-ended, writing: "There is no clear scope yet, and the scope may evolve in the future. But the rule of thumb so far: In scope is everything necessary to deploy a Linux-based operating system to an average public body with a few hundred users." And they do give examples of what is clearly out of scope. So, for example, not the development of a novel Linux OS, a distribution from scratch. "Instead, EU OS," they write, should build on top of an existing, well-established Linux distro. Also not is the development of EU OS outside of a corporate environment. For their personal computers, people can already choose between a large variety of Linux distros."

So this is not meant, I mean, it's not meant - it's not directed at the personal user that already has all their choices wide open. They're aiming this more at the several hundred user level public sector organizations. You know, like a police department, for example. Also out is the deployment of EU OS on other devices than typical desktop workstations or laptops. Hence, for example, smartphones are out of scope. So it's really the Windows desktop environment replacement is their target, but not for everyone, although everyone could use it if they wanted to.

So looking at use cases and at some previous attempts and successes, the site notes: "To make EU OS a success, it should support a large number of use cases and consequently a large user base. This helps to gather political support and funding for continuous improvements and innovation." They note that some specific regions outside of Europe have already utilized the benefits of an operating system which is under their control. And these are historical, back from the early 2000s.

They said: "Astra Linux is a Russian Linux-based computer operating system that's being widely deployed within in the Russian Federation to replace Microsoft Windows. Kylin is an operating system developed by academics at the National University of Defense Technology in the People's Republic of China ever since 2001. Together, Kylin and Neokylin share a 90% market share within the government in China.

"Nova Linux was central to the Cuban government's desire to replace Windows. Hector Rodriguez, Director of University of Information Science in Havana, said that 'The free software movement is closer to the ideology of the Cuban people, above all for independence and sovereignty.' Other cited reasons, of course, to develop the system include the United States embargo against Cuba which made it difficult for Cubans to purchase and update Windows, as well as potential security issues feared by the Cuban government because of the U.S. government's access to Microsoft's source code." So here the site is making the point that other governments, government-sized decisions have been made to say goodbye to Windows, and Linux is where they've gone. Just to sort of, like, I'm sure to demonstrate that this is feasible, and they would not be the first movers on this.

Citing these use-case successes, the site states: "This leaves no doubt about the feasibility of large-scale Linux deployments in the public sector. It is only a matter of political support, priority, and funding." The site notes some details of past migrations away from experiences with Microsoft and Windows. The city of Munich - and again, historically, this was 20 years ago, but it serves to highlight the problems inherent in the use of another country's commercial operating system for public sector needs.

The report wrote: "The city of Munich is migrating its desktop computers from Windows to GNU/Linux. After preparations began in 2003, the city's basic client, a customized version of Debian GNU/Linux, is being developed on a growing number of PCs since the fall of 2006. The LiMux project puts great emphasis on becoming independent from software suppliers. Florian Schiessl, the deputy project coordinator for LiMux, explains: 'Microsoft has shown us what it means to be dependent upon a vendor.'

"Until 2003, the city was using Microsoft Windows NT 4 across the board, and was by and large satisfied. When Microsoft decided to end the support for this operating system, this meant that hardware and important procedures would eventually stop working. It was from this experience of being totally at the mercy of an external party that we wanted to take the road to more independence." So they cut that umbilical cord 20 years ago and didn't look back.

For the French Gendarmerie: "GendBuntu is possibly one of the largest Linux-on-desktop deployments in the EU public sector, with about 82,000 seats." Lieutenant Colonel Guimard said: "Moving from Microsoft XP to Vista would not have brought us many advantages, and Microsoft said it would require training of users. Moving from XP to Ubuntu, however, proved very easy. The two biggest differences are the icons and the games." And he said: "Games are not our priority." Yeah, they didn't want people playing games in the police department.

Leo: No. No Tux Runner on this one, okay.

Steve: No. He said: "The transition [to Linux] went unexpectedly smoothly. Almost no additional training was required for the local police forces using the computers in their daily work. The Ubuntu user interface was easy to get used to. Pascal Danek points out that a transition from Microsoft Windows 2000 and XP to Vista would have been more difficult, since the new version of that OS introduces many new features and designs which might confuse users."

The French currently uses a customized version of Ubuntu called GendBuntu. "If EU OS would be used instead," writes Robert, "resources could be mutualized across all users of EU OS." So the idea being over time that, you know, there already have been major public sector deployments of Linux; that it would be a value to, you know, homogenize all of these under a single umbrella. And he's proposing EU OS.

One of the references for this on the page was an Ars Technica piece from 2009 with the headline "French police save millions of euros by adopting Ubuntu." And it's not difficult to imagine at this point that they're glad they did that back then. You know, they're likely still running on the same hardware without any trouble.

And then we have the case of the Swiss Federal Court. "Until 2001 the court had a simple all-in-one IT platform, which lacked greatly in functionality and ultimately became outdated. The Court's IT direction thus saw the necessity to introduce a new IT infrastructure that would ensure sustainable standards in the future. During the analysis done as part of the planning process, open source software emerged as more sustainable than proprietary software, especially with regard to modularity and file formats. The use

of open source software also ensured vendor independence and security, which are two very important aspects for a court.

"In 2001, the new IT system running on the operating system Solaris by Sun Microsystems was introduced. With this also came the introduction of the office suite StarOffice, the Internet browser Firefox, and the email client Novell Evolution, besides other more specialized applications. At the early stages of the migration, users had to get used to the new programs; but as the migration from the previous system brought numerous improvements, the process went relatively smoothly and was broadly accepted. Where some doubts about open source software existed in the beginning, they've mostly faded by now."

And finally Linux - or two more. A short one, Linux Plus 1 in Northern Germany. A region in the north of Germany is currently preparing the migration of their entire public administration to a Linux desktop. This migration would become one of the largest Linux-on-desktop deployments in the EU public sector with 30,000 seats. It is unclear which operating system will be used. Rumors say it will be based on KDE Plasma. If EU OS would be used, resources could be mutualized across all users of EU OS. And a reference listed for that was a piece in the ever-irreverent Register last April with the headline: "Germany's Northernmost State Ditches Windows." Yeah, indeed. And you know, Leo, Microsoft must be feeling all of this.

Leo: I don't know. They still have, like, 99% of all computing.

Steve: I know. But they are, you know...

Leo: I've been advocating for this forever, and I think especially in the public sector.

Steve: Yes.

Leo: Why should you be using Windows?

Steve: Yes.

Leo: Didn't they do this in China, though? They have the, what is it, Red, the Red OS?

Steve: Yeah, it's - I just talked about it, Kylin or something.

Leo: Yeah, oh, that's right. But it's another Linux spin.

Steve: Yes, it is, yeah, because you can't create, you can no longer write an operating system. And why would you? There's a free one that a bunch of really good, smart people have been working on for years.

Leo: Well, and this is why Android is so popular on handsets, although it's just another spin of Linux.

Steve: Yeah.

Leo: Yeah, it's kind of interesting. I think when you retire you should probably move to Linux. I'm just saying. You're not going to do it, are you.

Steve: I'm not going to retire.

Leo: Oh, there, that's better. Good answer.

Steve: That's right.

Leo: We're all going, whew, that was close. What's Leo - is he nuts? Don't use the "R" word with Steve.

Steve: Anyway, so as we know, there's still a lock-in problem with Microsoft's otherwise very compelling solutions. Under the headings of "Cities and Communities," Robert wrote: "Only a few cities have migrated to Linux so far."

Leo: I think support is probably part of the issue, as well; right?

Steve: Right, yes, exactly. And so that's - and that is one of the things that he noted is it will be necessary to be able to get support. So I think that's one of the reasons to look at Fedora as a possibility is that it's possible to get commercial support until they're able to, like, build up enough internal knowledge to do that themselves. But, he wrote: "Compatibility with the federal government and the plethora of business processes a city owns are a challenge. Oftentimes, reliance is strong on Microsoft Office."

Leo: Sure.

Steve: "Which historically did not run on Linux." He says: "With Microsoft 365 working in the browser, a workaround may be possible." To sort of, you know, pry the operating system out from underneath the browser. And look at Microsoft moving all of their focus to the cloud. That really seems...

Leo: Sure, they don't mind. They're going to get your money. Yeah.

Steve: Yup.

Leo: I would suggest it'd be good to get off the docx format, as well, at some point.

Steve: Yeah.

Leo: LibreOffice is out there. You can use that.

Steve: So let's see.

Leo: Oh, I'm sorry.

Steve: So - no, it's okay.

Leo: I'll stop making snide comments. It's all yours.

Steve: No, it's not a problem. They have an FAQ that offers some interesting technical insights. They ask themselves, is EU OS another Linux distribution that I can try out? And Robert answers: "EU OS is not another Linux distro. EU OS is a community-led proof-of-concept which employs existing Linux distributions. The challenge of the proof is not that an individual can use Linux on their own computer." And actually at one point Robert has like five that he uses at home constantly. So he's like, you know, he's really deep in. He said: "The challenge is to prove that an admin team, exactly to your point, Leo, an admin team can manage users and their data, software and devices, with or without Active Directory, and without Microsoft Windows, within a migration period of two years rather than 20 years."

Leo: Yeah.

Steve: He said: "For this, EU OS wants to propose a common Linux OS and desktop environment as a base and, more importantly, a common method to manage users and their data, software and devices. EU OS is not meant for home users, but for system administrators who want to automatically deploy and manage Linux across many corporate computers and laptops." So, and that's where GitLab comes in. They're talking about this as a deployment management issue where that's what they need to work out. "In the same way that Microsoft has done this for Windows in the corporate environment, they want to recreate some of that infrastructure for Linux that doesn't exist currently."

So, question: "How can the EU achieve its goals of being secure and sovereign when it relies on software from other countries, for example, the U.S.?" And he responds to this question: "EU OS shall not confound sovereignty and protectionism. There's no problem per se in relying on international free and open source software components, and oftentimes it is practically unavoidable. However, EU OS promotes the maintenance of strict control over business data and telemetry data." Meaning no phoning home, you know, GDPR compliance.

"This includes the free choice where to store such data, on-premise or cloud of choice. Furthermore, the availability of know-how for a given FOSS component within the EU shall be considered. It remains to be studied if EU OS FOSS components such as the Linux kernel, systemd, Wayland, PipeWire, Fedora or AlmaLinux, could face export limitations, which would pose a threat to the sovereignty offered by EU OS. Such threads cannot be mitigated by EU OS alone and should be addressed through industry supply chain security policy."

Okay. "Why does EU OS propose to rely on Fedora-based Linux distributions?" Answer: "EU OS is not a product yet, only a proof-of-concept. The choice of the employed base Linux distribution or desktop environment (Gnome or KDE) is not a core concern as it does not impact how admins manage users and their data, software, and their devices. And that's the focus. Nevertheless, EU OS cannot avoid picking some base Linux distribution to start with. Advice has been received and considered from individuals in their personal capacity of the following organizations: EU OS community on GitLab, CERN, European Commission, DG DIGIT, German Centre of Digital Sovereignty, ZenDIS (known from openDesk), Gnome OS, and openSUSE through their dedicated blog post.

"Considering the advice received, the decision was to advance the proof-of-concept with Fedora. For a production deployment after the proof-of-concept, any Fedora-based Linux distribution with longer release cycles could be used. Also, a switch to any other bootc-supported Linux distribution would always remain possible."

So this effort in the EU is what we would call definitely "handwriting on the wall" for Windows. You know, I mean, this existing will help to facilitate other small, you know, disconnected movements without any big mandate being needed. Individual entities in the public's EU sector can decide, hey, here's the support that we've been needing in order to, you know, hold our breath and make the move. But as I was reviewing and assembling all this, I realized how Windows-centric most of the U.S. is. And Leo, to your point, how dominant Windows itself is, and of course I know that many of this podcast's listeners have already liberated themselves from Microsoft's proprietary grasp. But throughout most of the United States, encountering anything other than Windows, you know, anywhere you go is a rarity. And that shows little sign of changing.

You know, to your point about my retirement, Leo, I'm very comfortable with Windows. I love the platform, which I've been using since before its birth. And as a commercial product developer it's still where the market is. But I'll also note that I spent some time, for example, just last weekend, updating my Ubuntu system, since I go to whatever lengths are necessary now to assure that anything I do will run smoothly under WINE, which of course is the free Windows emulator for Linux.

In the EU, as we saw mentioned, leaving Windows will not be an easy thing for any large organization to do. I suspect that future migration will not occur from the top down, but rather from the bottom up. You know, the broad pyramid, the broad-based pyramid, smaller entities that are more able to leave Microsoft will be under increasing pressure to do so as Microsoft's, well, what I would consider nearsighted policies attempt to force wholesale hardware replacement when they force software upgrades. This will cause smaller and inherently more flexible entities to explore what alternatives to Windows 11 may exist for them. And having the EU OS present may provide a path for smaller organizations to take once Microsoft has pushed them in that direction.

Now, as I said at the top, all of that said, I'm haunted again by that brilliant and poignant xkcd cartoon which we've looked at from time to time. It's the one showing that massive stack of various sized blocks all stacked on top of one another, which is so brilliant because it's exactly the way modern software "stacks" are created and operate; and where, amid this towering collection, there's one little block off to the side, near the bottom, upon which all the other blocks implicitly rest.

Now, as I was putting this together and wanting to find that xkcd cartoon again, I turned to ChatGPT to let it do the legwork for me. I copied and pasted that description which I just read above, since I'd already written it. And here's what the ChatGPT replied.

Leo: Found it. That's cool, yeah.

Steve: It said: "The xkcd comic you're referring to is titled 'Dependency,' comic number 2347. This illustration depicts a precarious tower composed of numerous blocks, symbolizing the modern software infrastructure. At the base of this towering structure is a single, small block labeled 'A project some random person in Nebraska has been thanklessly maintaining since 2003,' highlighting the fragility and reliance of complex systems on often-overlooked components." Now, okay. Let me just say AI, holy crap. You know, I mean, it produced that.

Leo: I mean, you could have done a Google search and found it, too. But okay.

Steve: Yes. I know.

Leo: It is kind of cool that it can do that, though.

Steve: It's incredible. And it said: "The comic serves as a poignant commentary on how critical pieces of modern digital infrastructure can depend heavily on small, open-source projects maintained by individuals without widespread recognition or support. This theme resonates with real-world scenarios where the failure or abandonment of such a project can have widespread repercussions across dependent systems."

Okay, now, I'm 100% certain that everyone listening to this who has been following along with us for even a few years will perfectly understand the motivations surrounding the desire to switch away from an operating system solution - regardless of how functional, compatible, and interoperable it may be - that does not appear to be directly driven by a motivation of planned obsolescence. Which is to say, you know, why are we - as you said, Leo, support for Windows 10 is ending this coming October. And, what is it, a quarter million systems? A quarter million systems currently running Windows 10 will not run Windows 11.

So, you know, it's one thing to be a computing enthusiast, where we're using and working with computers for their own sake, as many, if not most people listening to this podcast, do and are. But it's entirely different to be a police station out in a small rural town in France where all you want is to be able to bring up records, search the Internet, balance the books, and communicate with colleagues, and not needing to play games. This is a place where a computer is a tool, not a toy. And its reduced ability to be used for playing games may be, you know, may be a feature, not a bug.

So it's clear why a move from Windows to Linux would make so much sense for them. If it's possible for Linux and the tools that run on top of it to get their job done, then it's going to be far more cost effective in the long run to say bye-bye to Microsoft, you know, and to be able to keep running effectively and efficiently until the day that hardware itself finally dies because, you know, eventually the power supply will, or some capacitors will leak or something.

But this brings me back to xkcd's observation of that random person in Nebraska, and all of the tens of thousands of other random people everywhere who thanklessly create and maintain that system, the whole house of cards, the whole stack of bricks, you know, only apparently for the sheer joy of doing so. Right? That's their reward.

Now, I suppose this is a sustainable model, but that's my question, the sustainability. It has always been, after all, the goal of the free and open source software world that this is addressing. That dream is really coming true now in spades. But as more and more incredible value is obtained from the tireless work of volunteers, I don't know, sometimes

it feels maybe a bit unfair to them because, to use xkcd's word for it, it really is thankless work.

You know, I've created a great deal of free software which has been and remains quite popular. But it doesn't feel thankless to me at all because everyone who downloads it knows where it came from and who created it; you know? And I get sufficient feedback literally in the form of thanks from its users who use it to, you know, find an open port on their router they didn't know about, spot a bogus thumb drive, keep Windows from updating, find faster DNS servers, whatever. I receive plenty of thanks.

But I worry about those thankless people who toil without any recognition. I suppose the recognition they receive from their peers within the community they share is enough. I hope it's enough because having achieved the dreams of the likes of Richard Stallman and Linus Torvalds, what we need now is sustainability. You know, as these thankless developers see more and more of the world using their stuff and taking it totally and literally for granted, I hope they see it as a badge of honor that what they've created is helping so many people for such low cost. What has been accomplished, as evidenced by the creation of this EU OS unification project is truly, I mean, truly a stunning achievement. But now we have to have it keep going.

Leo: I think it's definitely hard to work in open source. Open source communities can often be grating, and I know a lot of project leaders, even in the last couple of years, have abandoned their projects because they're so fed up with the process.

Steve: Or they just age out. I mean...

Leo: Well, yeah. I mean, most - there are probably many projects that are simply done by one person. But most of the big ones have a group of people. They have a fearless leader, benevolent dictator for life. And the rest of them go along and work on it. I think increasingly it'll be politically motivated. Right now it's somewhat altruistic, somewhat just...

Steve: Well, China and Russia, certainly political.

Leo: I think, but even more than that, people are starting to resent these big corporations' extraction of value from them.

Steve: Yeah, I think it's economically motivated.

Leo: Yeah. Well, that I'm - yeah. I'm assuming they're political because it's anti-corporate. It's anti-capitalistic. It's more of an operating system for the people by the people. I love - I have to say I have loved Linux since I first installed Slackware 25 years ago. Have used it nonstop since then. And I can't see any way that it's not superior. What's interesting is that a lot of what people are doing is really just in the cloud. So for a lot of these people, I mean, you said, oh, well, they can use Google Sheets.

Steve: All you need is a browser.

Leo: It's just a browser.

Steve: Yup.

Leo: And for that, you know, it'd be a simple, I mean, that's what a Chromebook is, is basically a Chrome-based Linux operating system. But it'd be simple enough to create a browser, you know, open source browser on top of an open source operating system. But then you're still using the big tech, you know, Google, Microsoft's cloud-based stuff. I don't know. I'd love to see a world where it's more do-it-yourself. I mean, there's definitely a do-it-yourself movement in hardware and software.

Steve: Well, and certainly...

Leo: But I really like the maker movement; you know?

Steve: Right. And certainly this sort of effort with GitLab and EU OS, I mean, this is very much, I mean, the guy himself who's driving this project has a, you know, we're going to use our own cloud approach.

Leo: Right.

Steve: Because, you know, we really do want to...

Leo: As government probably should; right?

Steve: Yeah. For GDPR we want...

Leo: Privacy.

Steve: ...no phoning home. We want to cut the apron strings.

Leo: Right, right. Darren makes another - Darren's so good. He makes a lot of good points. He made another good point. He says: "At some point in the next, I don't know, few years, maintainers of these products may be AI-based, if not fully, at least primarily. And that would be fantastic; right? If you could say, okay, AI, your responsibility is OpenSSH. Make sure it's reliable, robust, and bug-free."

Steve: Respond to any vulnerabilities that are discovered.

Leo: Yeah. And that's one of the problems right now is you get all these pull requests, and you get all these bug reports, and if it could process them quickly and efficiently, I like that idea, Darren. Maybe we are, and maybe we'll enter a new world

at that point. Because really humans shouldn't have to maintain the infrastructure. Humans should be able to use the benefits of that.

Steve: Yeah.

Leo: You know, the front end of it. And maybe something computer-based can maintain it.

Steve: I like it. I do, I've said from the beginning, our first discussions of AI, that AI and code really do seem like they go hand in hand. I mean, it is - just makes so much sense.

Leo: It kind of makes sense. The computer speaks its own language; right?

Steve: Yeah.

Leo: Better than any human does.

Steve: Well, and ultimately logical; you know?

Leo: Right.

Steve: I mean, it's not, you know, fuzzy English wording, although they sure do have that mastered. My goodness.

Leo: Yeah. It's pretty amazing. We are going to do some more - we talk a lot about AI now on Wednesday on our Intelligent Machines show. We've got some great guests coming up, including in a couple weeks Harper Reed's going to talk about how he uses AI for pair programming, you know, that's where - he's writing code in conjunction with AI coder, and his workflow's quite interesting. We live in a new world. It's exciting.

Steve: We do. We're here for it. Yay. And we're going to be here for the foreseeable future.

Leo: Yay. No retirement in the works for this cat. He's going to stay here. Yay. I feel like I could almost touch you, Steve. I want to clap you on the back. Steve Gibson does this show every Tuesday. I hope you come and watch us.