Doing It Wrong

Description: This week's podcast is titled "Doing It Wrong" because the week's news happened to include four unrelated examples of companies really getting security wrong. So Leo and I first catch up on the week's other news and miscellany. Then we take a look at four examples of security being done wrong.

High quality (64 kbps) mp3 audio file URL: http://media.GRC.com/sn/SN-530.mp3
Quarter size (16 kbps) mp3 audio file URL: http://media.GRC.com/sn/sn-530-lq.mp3

SHOW TEASE: It's time for Security Now!. Steve Gibson is here. We're going to have some fun. Of course there's security news, but then Steve will cover four companies that did it wrong, really all ripped from today's headlines, this week's news, companies who just messed up security bad. It's coming up next on Security Now!.

Leo Laporte: This is Security Now! with Steve Gibson, Episode 530, recorded October 20th, 2015: Doing It Wrong.

It's time for Security Now!, the show that protects you and your privacy and your security and all online stuff like that with Mr. Security himself, James Tiberius Kirk. No. Steven "Tiberius" Gibson. Hey, Steve.

Steve Gibson: You guys were talking about Star Wars on MacBreak Weekly. And I thought, you know, I really have always been more of a Star Trek person than a Star Wars person.

Leo: Yeah, I think that's a big divide. I think there's Star Trek people, and there's Star Wars people.

Steve: I remember, like everybody of our age, going, seeing the first Star Wars movie and being completely blown away. But there's just not enough of it. And whereas, boy, we've had our fill of Star Trek. Multiple series, branches off, and movies, and now it's back again for more. Of course, so is Star Wars. But, yeah, I like the Star Wars stuff. Is this next one a prequel to the existing movies? It's like "The Force Awakens"; right? So is that like before?

Leo: No.

Steve: Oh.

Leo: This is after - it's Episode 7.

Steve: Okay.

Leo: So it's after episode - it's after the first - it's in between the first and - it's the next thing, the next - it's whatever. Because I'll tell you how I know that, because Han Solo's old. And Leia's old. And the Wookiee is a little grey. No, the Wookiee's not grey. But anyway, so it's still in the lifetime of Luke and Han and Princess Leia, but it's towards the end; right?

Steve: Yeah, we don't have any time travel in Star Wars, so...

Leo: Star Trek did, apparently. But, yeah.

Steve: Oh, Star Trek had time travel so much you didn't know who was who.

Leo: Where anybody was.

Steve: You know, Spock is meeting himself and saying, you know, "Don't talk to me. Don't tell me anything." Okay.

Leo: So, yeah. So episode whatever that is - four, five, six, yeah, seven.

Steve: I'll have to watch, I guess I'll have to rewatch them all to get ready. If I can tolerate Jar Jar Binks running around for...

Leo: Oh, he's gone. Or actually...

Steve: Oh, thank you.

Leo: He's been a dead a few years.

Steve: Good.

Leo: So you know what, I might be really confused for everything. But the first Star Wars was Episode 4, you remember. That was the one that came out in 1977.

Steve: Correct.

Leo: So we saw four, five, and six.

Steve: And it annoyed us. It's like, wait. It was such a tease. It's like wait a minute. Episode 4? What? What happened to - I want one through three.

Leo: And so they did - and then they went back and did one through three.

Steve: Oh.

Leo: That's where Jar Jar appeared.

Steve: Okay.

Leo: In one. And then...

Steve: He was long dead when Luke was around.

Leo: He was long gone, thank god.

Steve: Good, good.

Leo: His ears fell off. And then - so now we're going to finish the trilogy of trilogies with seven, eight, and nine, I presume. Unless, because it's Disney, they might want to do a 1.5, I don't know.

Steve: We're going to get three more, though.

Leo: Yeah. Oh, three more, are you kidding? It's Disney. We're going to get hundreds more. It's going on forever.

Steve: So this episode sort of came together as I was pulling - as I was aggregating news, I realized there was sort of this theme emerged just from what happened during the week. There were four different instances of doing it wrong, where it's just, you know, these people don't seem to be up to speed.

So I thought, okay, let's rearrange things a little bit. We'll title the episode "Doing It Wrong." We'll cover some news and some miscellany stuff, and then look in more detail at four examples that are, I mean, just across the board, a problem that Target has. A problem that 1Password has. A problem with chip-and-pin design, and a problem that appeared with last week's Patch Tuesday with Sandboxie, and reactions to it and so forth. So and then we've got to talk about the emergency Flash update that happened after we talked about Flash last week. Of course, some interesting stuff about the iOS App Store problem that I know you were talking about.

Leo: Oh, good, yeah.

Steve: But of course we take sort of a more techie, geeky approach. But there's some interesting things there. And then some miscellaneous stuff. People keep asking me what I thought of "The Martian," and you and I have not talked about movies recently, and I had some thoughts about "Steve Jobs." So I made a note to not forget about talking about that. So I think a great potpourri...

Leo: Jam-packed.

Steve: ...interesting podcast once again.

Leo: All right. I like it. Steve Gibson, it's time to get the news portion of the show.

Steve: So there is a fun little cartoon that Simon Zerafa tweeted to me that I got a kick out of. The guy, looking rather frustrated says to someone on tech support on the phone, "My computer doesn't work. The hard drive crashed. What do I do?" And the respond through the phone is, "Did you back up?" And then in the next frame we show him way across the room, his chair tipped over, plastered against the wall. And he says, "Why? Is it gonna blow?" So, dumb, but funny.

Leo: It made me laugh.

Steve: Good. And I actually got - this is, I think, the first time I've ever gotten a tweet from someone who said, "Hey, I saw the cartoon, and I think it's funny."

Leo: Nice.

Steve: Good. So Adobe's had a rough time, more than, well, I was tempted to say more than usual, but that's about par for the course. Not only was last Tuesday Patch Tuesday for Microsoft; but, as is often the case, it was Patch Tuesday for Adobe, or Adobe Flash update day. In this case, immediately after releasing last Tuesday's patch, news surfaced of an ongoing zero-day exploit against the post-patched Adobe Flash, which was used in targeted campaigns. Trend Micro found this happening. They discovered that current Flash zero exploits were being used in spearphishing emails aimed at political and

military themed targets.

So I take my hat off to Adobe. This generated a lot of news. They got on it very quickly. They initially announced they would have a fix this week, but they got it out by only a couple days later, so by the end of last week. So again, IE and Chrome now take responsibility. Firefox will notify you, but doesn't seem to actually be proactive. So we should all be at 19.0.0.226. That's where you want to be now.

But again, IE and Chrome users need not worry. But Firefox users, again, it's unlikely that you're going to trip over this surfing the web because this is part of the so-called "pawn storm" campaign, which has been going on for some time. They have an amazing supply of zero days. Like, six of them were found being used earlier this year, and only one of those six was also part of that big, 30GB data breach where we discovered a lot of vulnerabilities that weren't known before.

So anyway, it's not super critical. I was pleased that Adobe responded as quickly as they did. And I would say only if you are running Firefox should you verify that your add-ons are up to date. And they make that easy. You can go to add-ons. And in an obscure little link up at the top of the page, I don't know why it's not a big button, but it's just a little blue underlined link that says you know, verify that my add-ons are current. And when you press it, then you're told that Flash is not, and you should go fix that. So everyone should.

So, okay. Lots of news over the course of the last week about Apple discovering - actually it wasn't Apple who discovered it. It was a group called SourceDNA who found this and then told Apple, who then pulled - the number I've seen is 256. But I've seen varying numbers in various reports, and it might just be that they were still finding them. These were apps that were found to be essentially mildly spying on their users. And I'm less concerned about what these apps were doing than the mechanism that they were using, which is sort of interesting and a little bit troubling.

So the background is that there were several hundred apps which were all using a Chinese advertising SDK, a software development kit. And so this is something that the app developers just got from this company called - it's Yummy or something. It's in my notes here. I'm not...

Leo: It's Youmi, Y-O-U-M-I.

Steve: Oh, yeah. Okay, right. And so the developers would just drop this, would use this SDK, which sort of is a black box of software functions, to enhance the features of their app. But there's no visibility into exactly what the SDK does. But that's the way a lot of the world works is you get third-party libraries, and you add them to your app, and you link to them in order to get the advantage of their functions.

Leo: Presumably, this library was, I would guess, to interface with the Youmi ad network.

Steve: Right, exactly. Okay. So what was discovered was that the functions in the SDK were managing to use restricted APIs, that is, APIs, application program interfaces, that's sort of the term for the system or some other interface. The idea is that somebody who wants to offer services, like the iPhone offers services to the client applications running

on it, they publish this set of APIs, which is sort of a - it's a formal statement of these are the functions which the operating system or operating environment is making available to applications.

So, for example, put an icon on the screen might be a function. So rather than the application having to write code to do that itself, where every application would have to rewrite and reinvent the wheel, that's such a common thing to do that the operating system just has a function call. It has an application programming interface, an API that quickly makes that happen in an easy and uniform way.

Leo: Can I - slight refinement on that?

Steve: Sure.

Leo: The library is doing that. So you call the library using the interface that is published, the API, the application programming interface. So Windows and Mac come with libraries to draw windows. You don't want each programmer to write that each time.

Steve: Right.

Leo: The API isn't that. That's a library. But the API is how you talk to the library. It's the calls that you're going to make to the library. I know, you're an assembly language programmer. You don't have to deal with this stuff. But it's kind of like INT 13; right? INT 13 calls some routines. The API is the INT 13 call.

Steve: Correct. The API is the definition, the formal definition of...

Leo: How to get to this.

Steve: ...the interface to those functions.

Leo: And the reason I bring that up is because there's public, and there's private.

Steve: Well, yes. And this is what's a little disturbing, is that, in Objective-C, these APIs are invoked by name, that is, by strings that name the API. And one of the things that Apple does, as you said, Leo, there are private APIs. There are functions which Apple doesn't want applications to have access to. And so one of the things that Apple does when they screen an application before it goes into the App Store is they do an analysis. They, like, scan the app, looking for the names of any functions which the application should not use.

Well, what's disturbing is this is really a bogus way of preventing an application which uses strings to call functions from doing so because what this Youmi, whatever it is, SDK was doing is they started out just apparently putting their toe in the water. They wanted

to find out what the foremost app name was. And that's something Apple says no application can know. Applications, the idea is they're supposed to be sandboxed. So they just sort of have their own little world, and they don't know who's on first or second or anything. They're just there like everybody else. But there is an API call to obtain the name of the frontmost app.

Leo: Because of course the OS might need to do this.

Steve: Oh, yeah.

Leo: You know, Apple themselves might need to do this.

Steve: Right.

Leo: Or even other libraries within the OS might need to do this.

Steve: Correct. Anything privileged might. But Apple is saying we don't want applications to know because it's none of their business. And so what happened was that the earliest versions of this Youmi SDK were playing with that. What they did was they essentially obfuscated the string. Like did some simple little encryption, I mean, it wouldn't have to be much. You just don't want a simple string search to be able to see the name of the API. So at runtime they synthesized the name of this forbidden API and called it. And so, from China, they stood back and waited to see if anything bad happened. And nothing bad happened.

Leo: In fact, it worked.

Steve: It worked.

Leo: Which it shouldn't; right, Steve? I mean, I don't understand why the OS allows this.

Steve: Correct. And this is what's disturbing is that then they got more aggressive. And they added additional functions to their SDK to enumerate the list of installed apps, in addition to getting the frontmost app name, to get the platform serial number to enumerate the devices and get the serial numbers of all the peripherals, and then to get the user's Apple ID. All restricted. But this is the problem, as I said. What Apple is doing is weak protection. I mean, laughably weak. So that I'm disappointed.

Now, apparently there are stronger protections that Apple can employ. For example, Apple has blocked the acquisition of the platform serial number, in addition to scanning for this, you know, for like an obfuscated function name, which it's not practical for them to do. So I was taken aback by the idea that we have been relying on something, such a weak protection, which anybody could get around, if they wanted to. So, and I'm not deep into Objective-C and the Apple ecosystem and API. You know, I've programmed

Windows like crazy, but not iOS. So I don't know why something stronger hasn't been done.

Leo: Make it runtime as opposed to compile time or...

Steve: Correct.

Leo: I mean, it's crazy. It doesn't make any sense.

Steve: It is completely crazy. Yeah. It ought to - those functions which are protected should - there ought to be a firewall. The functions that are protected should check to see the privilege of the caller and say, wait a minute? Who's Youmi? You know?

Leo: In fact, that's kind of normally, I would assume, how it works. If you're not root, there are certain things you can't do.

Steve: Yeah. I mean, we...

Leo: Here's an application that's running at the application level that's trying to do root things.

Steve: This is old. This is solved technology. So there must be something I don't understand. That's why I want to make it clear.

Leo: There may be a reason for this, yeah.

Steve: Yes. On the other hand, they did strengthen the serial number API so that that hack no longer works, which is now why we presume that Youmi went around it, and they're enumerating the devices and the serial numbers of the peripherals, still trying to find a way of locking onto the device, which Apple doesn't want apps to have. So anyway, so they got caught doing this. Apple booted several hundred apps that were using this SDK out of the store. Now the Youmi SDK will have to remove this behavior and settle for as much information as they're allowed to get.

And I really do hope that this, I mean, the good news is this got a lot of attention. I hope Apple looks at why aren't we doing a better job of this? Because clearly the whole idea that you can access restricted functions by name, that's just an oxymoron because, if you're - oh, and that you're going to do a static analysis of the program in order to decide that it's not misbehaving. Well, obviously it can dynamically misbehave if it's not statically misbehaving. So, wow. I'm sure I'll get some people who, in fact, we've got a great Apple iOS developer working on the SQRL client.

Leo: Oh, good.

Steve: Who's just moved, who's upgraded his library, Jeff in the U.K. So he may explain it to me. There must be some reason they're not making it stronger because, wow, this is just weak.

Leo: And it brings to mind a couple of issues or a couple of conclusions you could draw. First of all, it's foolish to rely on the screening mechanism in the App Store to protect you. There's just no - we've seen it again and again and again, and I think Apple oversells the security of that. There's just no way they can catch everything. It's just not, I mean, there's too many apps coming out. There's too few people doing it. You just can't.

Steve: Well, and the nature of the PC ecosystem, you know, phones have now become PCs.

Leo: Right.

Steve: They're full-blown computers in your pocket. And the nature is we want everybody to be able to create apps. And unfortunately there are going to be some bad people.

Leo: Well, but that raises a secondary issue. I mean, here it is, Apple, and you've touted their security, and Apple's really said we're the secure platform, we're the safe platform. And, you know, you could point to some real things that they do. They encrypt the phone by default and things like that. But there is some reliance on them to do the right and intelligent thing throughout. And of course, because this is closed source, we can't tell that they are. This is a surprising gaffe.

Steve: It is. I'm just - I was stunned.

Leo: And it worries me because it shows that, I mean, you can build a faade, oh, we're secure; but, if you don't know how to do that right, or you neglect to do that right, you're not, regardless of what you say.

Steve: Correct.

Leo: And but we can't see into it because it's closed source. So we just have to take their word for it that they're secure. Well, now there might be a reason to think otherwise, frankly. I mean, it just bothers me that this is possible. And of course Apple will never say why and never explain it. And frankly, I don't think a developer, just because they're an iOS developer, would know either. This is a decision that you make at the operating system level about how you're going to restrict private APIs.

Steve: Yeah, although my guess is that this may be, it may tie into the way Objective-C does binding. There are statically bound languages and then loosely bound languages. And so there may be a reason to provide such a flexible linkage to functions. For

example, maybe the library needs to run in the user space for some reason. So the library itself has user privileges. And then of course it's calling, it's making calls to the iOS libraries. But it's doing so by name, with a string. And the idea that hiding the string is like - here's what we know. Hiding the name of the function past Apple scrutiny. Okay. Game over. I mean, it's just, what?

Leo: But that doesn't worry me as much as the fact that it's possible. You know what I'm saying? I mean, I think it's obvious to anybody paying attention that the screening process will never catch everything, it just won't.

Steve: Right.

Leo: But why is it possible to call, however you do it, to call a private API? That just shouldn't be possible.

Steve: Yup. And the fact that they...

Leo: And in fact we know Apple could turn it off because they did.

Steve: Yes, the fact that they can tighten it when they need to says, okay, well, why isn't it all tight?

Leo: That's more worrying, frankly, than the screening process because, if you think about it, and there have been plenty of cases where the screening process has failed, that's impossible.

Steve: Well, and, yeah, I mean, you know, PCs don't even try.

Leo: Right.

Steve: We gave that up so early.

Leo: I mean, it's good they try. And that gives you some assurance. But anybody who says, "Oh, there's just no way malware can get into the App Store," is not paying attention.

Steve: No.

Leo: Now, the good news is nothing that was leaked is really that private; right? It's not like credit card numbers got leaked.

Steve: And it'll be interesting to see if Apple does say, if Apple can be pushed to do some further elucidation of this because maybe these are soft prohibitions, and the really important stuff is and always has been locked down. And so, I mean, really, do we care if something figures out what app is on top? Eh...

Leo: No.

Steve: Not that much, probably.

Leo: But you might care if your email address is leaked.

Steve: Yeah. And that is the Apple ID is often your email address, which they have access to.

Leo: Right. So...

Steve: Yeah. Anyway, maybe - I'll be looking for some more information because I was stunned when I looked at the details from SourceDNA and that this was a matter of obfuscating a string, which was deobfuscated when it was run in order to call the function. It's like, what? That's just, wow, okay. Yeah.

So I got a kick out of this. This was a tweet from, again, Simon, who found - he was talking about the so-called, you know, this controversy over different memory suppliers for the iPhone?

Leo: Processor suppliers, yeah, TSMC versus Samsung, yeah.

Steve: Right, and how there was a slight difference in the performance of these.

Leo: Yeah, one chip's bigger than the other.

Steve: Exactly. And so, and apparently this had to do with the memory performance. And we know that memory, the flash memory, the term for it is NAND memory. So Simon found someone on Twitter who said: "When someone notices that there's a 0.2% difference in memory performance between different brands of memory used in iPhones, we will at least have the consolation that it'll be called 'NAND-gate.'"

Leo: Yeah. That's not what they discovered, unless they've discovered something new. They discovered that one has better battery, or theoretically better battery life, although Consumer Reports proved that not to be the case.

Steve: Probably the smaller one, someone would have guessed, would have had better life?

Leo: No, it's the other way around. I know. I would have thought the smaller one would be better. But they say the TSMC chip's better. I have the Samsung chip, Megan has the TSMC chip in our iPhones. It doesn't matter.

Steve: No difference.

Leo: It makes - do not feel that you have been somehow ripped off because you got the Samsung chip.

Steve: So I got a reminder via Twitter from Brent England, who said: "SGgrc. Have you posted your comments/review of 'The Martian' movie? Interested in your thoughts." And have you had a chance to see it, Leo?

Leo: Loved it.

Steve: As did I. And as I did mention on the podcast, I read the book because, being what it is, as an entirely different medium, you just get so much more exposition. And I found myself explaining to the group that I went to see "The Martian" movie with, all the things that we didn't know from...

Leo: Oh, I bet they loved that. I hope it wasn't during the movie.

Steve: Oh, no, no, I waited. But, like, the fact that the potatoes that he had, had never been flash frozen, which would have destroyed the bacteria in them.

Leo: Yeah, that was a good bit.

Steve: Because they were going to be used for Thanksgiving because their mission was going to straddle Thanksgiving, and so they had to have a Thanksgiving dinner with potatoes.

Leo: Real potatoes, yeah.

Steve: And so, yes, exactly, not powdered. And so anyway, so, yes, the movie was great. Loved it. The book was better. But, again, it's not to disparage the movie. But I read the book, and oh, my god...

Leo: Well, there's more in the book.

Steve: "Jurassic Park," the book was so much better than the movie. And there were, like, scenes missing. It's like, well, well, well, but, but, but, you know, it's like, okay,

well, yeah. I thought that was...

Leo: So the biggest change in the movie was not as many bad things happened as happen in the book. And in some ways I aesthetically preferred that because the book, you get a point of fatigue in the book where you go, oh, no, not - oh, not again. It's like, oh, come on. No, no, can't, no. And that's part of the book, of course. In the movie, just enough bad things happen.

Steve: Yeah. And I don't think the movie managed to convey the degree of his problem solving.

Leo: The difficulty, yes, I agree with you, yes.

Steve: Yes. Yeah, like making water. Yeah, they gave us a gratuitous explosion, and he was still smoking when he was, like, doing his log entry.

Leo: That was pretty funny.

Steve: It was a great humorous moment. But you didn't, I mean, I would argue that what was so compelling about the book, and of course our audience had read the book, we had talked about it, and they loved the book. It was that the book took us into how hard it was to survive. You know, this made it sound like, well, I'm checking off another day on my large calendar. But it mostly seemed like he would be bored, sitting there looking at the Martian horizon, and it's like, waiting to get rescued. It's like, he really didn't spend much time being bored in the book.

Leo: Right.

Steve: He was really struggling to keep alive.

Leo: Right.

Steve: So...

Leo: Both are great. You should always read the book before you see the movie, no matter what.

Steve: Absolutely.

Leo: Yeah.

Steve: So, speaking of which, there is a hotly anticipated series on Syfy, the "Expanse" series, which is starting not long from now, but there's a series of books. And the books are really good. So if anyone is interested in books before TV, the "Expanse" series, really interesting. Some really new concepts. And who knows what Syfy will do to it. I shudder to think. But we're starving for science fiction, so what the hell.

Okay. The "Steve Jobs" movie.

Leo: You saw it.

Steve: I will not see it.

Leo: Yeah. You shouldn't see it.

Steve: I should not.

Leo: It's offensive.

Steve: I knew Steve. I've seen every other movie. I've read every book. And, oh, and I meant to tell you, but I just, it slipped, that the two guys, I'm blanking on the names...

Leo: Sculley? Jobs? Woz? Oh, but those are the - Rick Tetzeli, Walter Isaacson...

Steve: Yes. Isaacson and Tetzeli were both interviewed for the entire hour on Charlie Rose.

Leo: Oh, I'll have to watch that.

Steve: Thursday before last. I wanted to make sure you knew because it was really a good interview. And it's funny, I didn't realize, the director actually said that they cut out time to make the actors seem like they're talking faster. I said, what? You're kidding me. I mean, they're...

Leo: Actors can rarely keep up with Aaron Sorkin's dialogue. That's the problem.

Steve: Right, right.

Leo: Your brain doesn't work that fast.

Steve: It's completely over the top. I loved "The West Wing," just for the - it's like, okay, real people...

Leo: Well, it's like that.

Steve: No one is that smart, and no one talks like that.

Leo: I wish, if you could have named it, you know, Steve Hobs, and made it about a roman clef, even, or just not even pretend, it's a great - it's beautifully directed. It's really Sorkin's best, among his best writing. It's really good in that respect. But the problem is we know the participants, and we know what happened, and it feels a little sacrilegious to watch it because it's not - that's not Steve.

Steve: No. And, I mean, and the idea of focusing it on just that high-pressure moments before the three...

Leo: Well, that's theatrical, that's why.

Steve: Exactly.

Leo: It's not a biographical movie. It's not intended to be.

Steve: Yeah.

Leo: So it works theatrically.

Steve: I'm surprised that Woz said he loved it.

Leo: Yeah, that I was very, you know why, because he is poorly - not as poorly treated in this movie as in the Ashton Kutcher movie, but close. At one point Jobs calls him "Rain Man." And none of the stuff - he seems to be, in the movie, obsessed with getting credit for the Apple II team, which even Woz himself says, "Yeah, I made one call once for 30 seconds to John Sculley. I never mentioned it to Jobs. And it was just when the Apple II team was all getting fired, and I said you should really give some credit for building this company. That's it." Nevertheless, that's all he says in the - it's terrible. That's what's a conflict for me. It's a great movie. I wish it were a fictional movie because it's not...

Steve: Right.

Leo: But these people are all alive. With the exception of Mr. Jobs, they're all alive. And, you know, it's very odd. What Aaron Sorkin even says he did is he read the book, he talked to all these people, just to get some stuff, and then he put it in a salad that had nothing to do with reality. But it's hard. How do you make a movie

about Steve Jobs or any genius? How do you do that? You can't.

Steve: And that's why I've been so interested in him is it's really, to me, he is an intriguing character.

Leo: Yes.

Steve: I mean, he isn't me. He isn't Woz. And I have to say, just for the record, Woz's engineering is beyond inspired. I mean, as an engineer, I knew the Apple II, which Steve designed by hand, because I designed a light pen for it. And it is possible for an artistic genius to express himself in NAND gates. And Woz did. There is stuff in the Apple II design that is just, from one engineer to another, it is just shockingly brilliant. And, I mean, so wow. And early, you know, the early...

Leo: But they don't, by the way, they don't deprecate Woz's talent in this one.

Steve: Good.

Leo: In this one, in fact, Woz says, "What do you do? I invented all this stuff. What do you do, Steve? What is your contribution? People think I'm Ringo, and you're John Lennon. I'm John Lennon," is what Woz says.

Steve: Yeah. When I can see it without it clicking up some counter somewhere, or generating box office revenue...

Leo: I mean, that would be - yeah, watch it on Netflix, and they get no money. But it's not that. It's more for me it was hard to watch because it's like, oh, it's like a gut punch because that - what? That's not - what? And yet, and brilliantly written, brilliantly directed, brilliantly acted. I mean...

Steve: And I love Sorkin's writing. So I'm missing a piece of Sorkin.

Leo: It's probably his best writing.

Steve: Wow.

Leo: Not one Sorkinism that I could detect. You know the Sorkinisms?

Steve: Yeah.

Leo: There are certain things that he says a lot. Not one. He obviously read that website.

Steve: Right, and then decided, okay.

Leo: I'm not using those clichs.

Steve: Point taken.

Leo: Yeah.

Steve: So two things, one from Twitter. Greg Mackay said: "You're using pfSense now? Roll your own or off the shelf box? More details in future episode of Security Now!?"

And then in the mailbag I ran across Mike in Tracy, California. The subject line was "SpinRite Recovers a TiVo Drive." So this was dated October 15, just happened. He said: "Steve, thank you. SpinRite repairs nonbootable TiVo Premiere" - oh, I think that's what I have, so I'm glad to know - "hard disk drive. After SpinRite Level 2 repaired the drive to bootable condition, I then promptly copied the contents to a larger drive." No, I think mine's the largest drive, so maybe it's not the Premiere.

Anyway, point is he says: "Several weeks ago you talked about pfSense. Can you tell us more about how you have it configured?" And I should mention that I've gotten, you know, these were two representative questions from a lot of people. I really think of it as the gourmet, high-end, every imaginable bell and whistle, industrial-grade router firewall software. It's evolved over time. It is free to download and install. It's based on FreeBSD. Many of these things are based on OpenBSD, which some people feel is a little more secure. FreeBSD, however, is faster. And so they deliberately put it on FreeBSD because there's less per packet overhead, and that was their goal because they were going to be providing all the security, just being a NAT router firewall.

What I most like about it is there's nothing more frustrating than needing something that's not there. So, for example, it has an incredibly flexible network address translation system where you can perform unrestricted IP-and-port to IP-and-port mapping, which can be static maps or dynamic maps. It's got really good bandwidth flow control, where you could have, for example, some machines in your network are restricted. Some protocols are restricted. Some destinations are restricted. I mean, basically it's just - it's a kitchen sink.

It's got an OpenVPN client and server, so it could be your OpenVPN endpoint for remote OpenVPN access to your network at home. Or you could use the OpenVPN client to create a static OpenVPN tunnel, as I have, to an OpenVPN server running at GRC, so that I have a static connection. And my connection to GRC went down for an hour in the middle of the night last night. I was offline for an hour, a little after 1:30. My IP changed when it came back up. But the OpenVPN tunnel had reestablished itself, and everything was connected.

So anyway, all you really need is a PC-compatible machine with two network interfaces. And of course we can now get USB NICs which perform well, USB 2, if you don't have

two on the motherboard. I chose to use a really nice hardware platform from a company called Soekris, S-O-E-K-R-I-S, Soekris. And they're in Scotts Valley, California. And I have to say I was very impressed. As I mentioned on the podcast, I slipped, pure fumble-fingers crazy. The lid of the case hit the top of some capacitors on the motherboard, and I had not unplugged it. Dumb. And I killed it. And it was like a week old. Because I decided I would get the latest Soekris board when I was setting up my system.

So I immediately ordered another one because I had to stay on the air. And I sent them email. And I said, hey, I've got a board here that's only a couple weeks old. I just killed it. Is it worth fixing? Will you guys fix it? What do I do? Because it's brand new. But it's my fault. But it's dead. So I shipped it to them, and they responded a day or two after they received it saying, "Yeah, it wasn't anything big. We fixed it, and if you'll pay for shipping, we'll send it back to you." So it's like, wow.

Leo: Nice. That's great.

Steve: I mean, that's what you want. So I can vouch for those guys. For somebody, however, who wants a turnkey solution, that is, who just wants to use this without any other muss or fuss, the pfSense people also sell hardware with pfSense preinstalled, running. And so you get this little box for 299, and it's plug-and-play. So, and optimized and tuned for their software. So check out pfSense.org. The software is there to download for free. You don't have to pay anything for it. And I should say that it also has an intoxicating web interface. I mean, the web interface is just how far down do you want to drill? It's, I mean, every - it does everything. And so for a higher end user that wants more power than one of our standard little plastic blue routers gives us, this is what I would recommend.

And, for example, right now I'm using it to NAT my network. It is maintaining a block of IPs which are a tunnel to a remote network at GRC. And I have a static Skype port mapped through to my Skype machine so that I have a zero relay connection to the TWiT Brick House. So, I mean, it's doing everything I want, and I'm not even using a 100th of its features. So you don't need fancy hardware. If you don't have hardware, Soekris, if you want to roll your own; or you can get the hardware from pfSense.

Or just, if you've got an old PC around with a couple NICs, since it's FreeBSD, and it's 10.1 FreeBSD, so it's the latest, it knows about all your hardware. It'll just drop in and find everything and configure itself and come up running. In fact, you might just, if you didn't want to stay with an old unused PC, you might just download it and install it to see what it looks like, and I think you'll be sold. I think it's the right choice for the power user.

And just a note from Al in Wisconsin, who said - he said: "Hey, Steve and Leo. Last month I had five laptops come into the shop with 'It's slow' complaints, and it turned out that all five had hard drive issues quickly detected by SpinRite. Afterward, my clients reported immediate and sometimes unbelievable speed increases. Thank you for making me look great to my clients."

And I just wanted to note that that's one of the things that people experience with computers. And as you know, Leo, as The Tech Guy on KFI, you're dealing with this all the time, there are many reasons for machines slowing down over time. But one that is often neglected is that the hard drives will silently retry when they're having trouble getting a correctable read from the sector. Even if it's not perfect, they can correct errors

up to a certain size. When it gets beyond that, they'll just try a bunch.

And so one of the things we often hear is that running SpinRite sped up the machine. So that tells you this drive was getting ready to be worse than slow. It was getting ready to give you no choice but to run SpinRite. So for those of you who have SpinRite, that's one of the signs of something happening. In fact, it happened to another machine of mine the other day, last week, in fact. It's the machine that I use for running Skype. And it was taking a lot longer to boot than usual. So I ran SpinRite on it after last week's podcast, and it booted fast today.

So it's just something to keep in mind. Drives will slow down, oftentimes before they die. It's annoying that we don't get any real notice of it than that, other than slowdown. But what people report after running SpinRite is that it's a lot faster.

Leo: Yeah, even on a SMART drive you don't get notice. You just get the crash.

Steve: No.

Leo: You get slowdown and crash.

Steve: Exactly.

Leo: Increasingly, you know, we get - I always get calls, lots of calls from people who have slow computers. And for a while, you know, the stock answer was, well, you have any malware on there? But increasingly I feel like this is the thing probably most likely to fail on a personal computer is the moving part, the hard drive. And that's a symptom. And so almost always now I just say, hey, first check the hard drive because that's, I would say, 90% of the errors that happen on PCs are because the hard drive is failing. You get crashes. You get slowdowns. You get all sorts of things. It's nice to be able to run a program that just says, oh, yeah, here, let me fix that for you.

Steve Gibson, Leo Laporte, Security Now!. All right. Four companies that did it wrong.

Steve: So first of all, Dale Myers, who is a Microsoft security engineer currently working on Office for iOS and OS X. He authored a blog post, well written, and basically that surprised a lot of people. The title of the blog post was "1Password Leaks Your Data." And then the first line is "Seriously."

And he explains this as well as I could, so I paraphrased it a little bit for brevity. But he said: "1PasswordAnywhere is a feature of 1Password which allows you to access your data without needing their client software. If you browse to your .agilekeychain file on disk, you'll find that it is actually a directory. Inside this directory is a file named 1Password.html. If you access this file over HTTP, you'll be greeted with a grey page which has a lock image and a password field." And so the idea is that this 1Password.html is sort of a self-encrypted page. And if you look at it with a browser, it just is grey and prompting you for your password. Enter your password, and your keychain will unlock, and you have a read-only view of your data.

"So what's the problem? Well, it turns out that none of the accompanying metadata is encrypted. I discovered this," writes Dale, "after having a sync issue with Dropbox." He says, "I use Dropbox to host my keychain. The file that had issues was 1Password.agilekeychain/data/default/contents.js. Being a curious kind of guy, I opened the file to see what was in there. The answer is the name and address," meaning the URL, "of every item that I have in 1Password. Every single one. In plaintext.

"For those of you thinking, so what, perhaps you have nothing of interest in there. But there are other considerations. Perhaps I signed up for somespecificpornsite.com, and this isn't something I want to broadcast. However, I've done just that," writes Dale. "Anyone who knows the link to the main login page for my keychain can obtain this file. They can go through and find out exactly what shady sites I have accounts, what software I have licenses for, the bank card and accounts I hold, the titles for any secure notes I have, and anything else I've decided to store in there.

"The second and possibly larger concern is that the login URL is stored with the page's title, all in plaintext. In other words, if I sign in at https://example.com/login, then that URL is stored with the keychain entry. This is often not an issue, but it can be. I recently signed up with a large ISP in the U.K. and had to reset my password due to a bug on their system. I was sent an email with a reset link in the email. I click the link, enter a new password, and press Submit. At this point two things happen. The first is that my password is reset. The second is that 1Password prompts to save my credentials. Since I used an auto-generated password, and I like to keep my passwords secure, I click Save. Now my new password is stored in my keychain.

"But what if my ISP's website didn't properly expire that email link after its use? Website developers aren't perfect. We make bad decisions, and sometimes dangerous ones. Maybe these guys made a mistake that is all too common. So I go back to my email and click the password reset link again. Sure enough, I get prompted with a screen where I can reset my password again. They didn't check to see if I had already used the link. And now that reusable password reset link to my account is stored in my publicly accessible 1Password metadata." And that's, remember, in plaintext. "Anyone," writes Dale, "can go and paste this link into their browser, and they have full access to my account."

Dale writes: "Presumably, I don't need to explain anymore about how this is a huge issue. But it gets worse. I decided to have a look to see just how bad things were. Thanks to people having links for easy access to their keychain on their websites," meaning that since this appears to be an encrypted webpage, people have put this encrypted webpage on their website for easy access, since it's encrypted.

Leo: Whoops.

Steve: But of course the metadata is not. So "Google has indexed some of these. A simple search brings up results. By looking at one of these, it was a simple matter to identify the owner of the keychain and where he lived. I know what his job is. I even know the name of his wife and children. If I were malicious, it would be easy to convince someone that I had compromised their account and had access to all their credentials. Not to mention the fact that they have revealed their location online, which may put their personal safety at risk.

"So what did I do? I immediately tried to reach out to AgileBits to make them aware of this serious data breach. When I received a response from one of their engineers, I was

given a few links to the details about their keychain and the assurance that, not only were they aware of this breach, but it was by design. 'When we built the keychain, we were aware that it would be possible to see that a user has logins across different sites because it is unencrypted. While this might have some privacy implications if an attacker gained access to it, your passwords are never exposed or shared, as they are encrypted by your Master Password, and they do not appear in the keychain in any way.'"

Okay. So Dale continues: "Searching through the forums, I found claims by employees that it was designed this way for performance reasons. The logic was that, if all the metadata was encrypted" - for example, like LastPass does - "along with the passwords, then when the user unlocked their keychain, if they wanted to search for an entry, all entries would have to be decrypted to find what they were looking for. And that is correct. What I didn't understand, and asked AgileBits, was why not just encrypt the metadata file using the master password in some fashion? Then they only have a single file to decrypt." And I guess this gets into the architecture details, which I didn't look at. Again, their reason was performance.

"When we first developed" - this is now AgileBytes responding, or AgileBits. "When we first developed AgileKeychain a few years ago, 1Password had significantly less processing power with which to function, and decrypting the keychain on the fly to do something as simple as login search incurred huge performance penalties for our users. Because this provided a poor experience for our users, we decided against requiring extra decryption steps for this process."

Okay. Furthermore, I, now speaking, did some additional digging. And I discovered that, back in 2012, three years ago, they changed things to add encryption for all the metadata. But they were not sure whether this new, fully encrypted solution would be completely cross-platform compatible. So they never made it the default. And almost no one is using it. So something has been available to address this for three years, but with apparently a reduced feature set. So it's not in use.

Leo: You can't put your keys - log in on the web and access it that way. You lose that capability.

Steve: So I will never give them any of my data because this represents a fundamental, crucial, failure of judgment. If they're putting the search speed performance in front of security, then they're off my list. So for what it's worth, I don't think anyone would argue that the fact that they have done it wrong because where we go, what we do, what we name things, as part of putting it in our vault, should be private, and should be encrypted; and the fact that they even knew it three years ago and didn't fix it, didn't make it work. Other people have. These people didn't. So thanks anyway.

Now, number two, oh, goodness. This is great. This is completely different, but there's a firewall lesson in here. So the BBC picked up on the story of something that happened in Campbell, California, a little bit south of San Jose. And the headline was "Target stores attacked by pornographic prankster." So David Lee, who's the BBC's North American technology reporter, wrote last Friday:

"Gina Young was shopping at U.S. superstore Target on Thursday morning, when she and the other shoppers suddenly heard a surprising announcement over the loudspeaker. Explicit audio from a pornographic film was blasted out for all to hear. And it kept playing. And playing."

Leo: Oh, my.

Steve: "For 15 minutes."

Leo: What?

Steve: "Ms. Young, who was shopping with her twin three-year-old boys, uploaded the clip to Facebook." And he has a link there with a warning: "Obvious warning: It has rude audio."

"'People were up in arms,' she wrote. 'Some people threw their things down and walked out. Others were yelling at employees.'

"As pranks go, it's fairly low grade," writes Dave. "But Target has a problem. Staff at the store in Campbell, a small city just south of San Jose, were all but powerless to stop it, due to how the PA system is designed. And it's not an isolated incident. According to local media, it's at least the fourth time this prank has happened since April. In one instance, a store had to be evacuated."

Leo: What?

Steve: I wonder what they were playing over the PA. "So what's going on? Are mischievous staff causing trouble? Have Target's systems been hacked? Oh, no."

Okay. So that's the end of the story. So here's - I know what's going on. It turns out that, for convenience, anyone who wants to make an announcement, any employee, presumably, wants to make an announcement, picks up one of the many available phones located around the store, and dials the extension number for the PA system. It's a four-digit number, so you're not going to hit it at random. But they all know what it is. So they dial that, and the PA system is an extension on the phone system.

The bad news is, you don't have to be in the store. You can dial Target from anywhere in the world and ask to be transferred to that extension number, or dial it yourself, and you are now on the Target store's PA system and have full rein. And there's nothing they can do to turn it off. And of course they did it wrong. This is, from a security standpoint, exactly analogous to hooking your Internet, your local Intranet, to the Internet. It's like, oh, look, we have a nice network here. Let's put it on the Internet. And so you just connect it. You know, no one would do that now. No one would allow internal resources to be accessed from the outside. But Target has no such concern, apparently. It's an extension on their phone system, and you can get transferred to that extension from outside.

Leo: Well, how would they ever get the four-digit code? I mean, come on.

Steve: Well, every employee knows it. And there's probably some turnover at Target stores. And there's no way to track it down. So it's common knowledge, I'm sure, what the PA's extension is. And they may change it. And then of course that news gets out. I

mean, it's a fundamentally insecure design. And so it's like virus and antivirus. They can change the number, but then they have to tell everybody what the new number is so that they could use the PA system. And again, it's sort of semi-public knowledge, which is going to leak out of the store and generate headlines like this. Whoops.

So, number three of doing it wrong, and I just have to shake my head at this one. I have not yet made time to dig into the chip-and-pin technology, but it's on my list of stuff to get to because I know it would make a fascinating podcast. Well, at least I thought it would make a fascinating podcast. Until I found out that the way these systems work today - what we would like is - we'll cover that first because all of our listeners who've been around for a while know how it should work. The data on the card, which is readable electronically, you know, the card will have a mag strip, but we know that that's not secure because it's read-only. So we would like to have that same data stored in an encrypted form, where the PIN is the decryption key. And, you know, maybe if there's a way to, like, prevent brute forcing, like put a delay in before the decryption proceeds, or before you know one way or the other, you know, it would be nice if it sort of had a PBKDF feature so that there was a delay in the card's response.

Turns out that's not the way they work. You stick the card in the terminal, and you enter the PIN. The PIN goes to the card with the question, "Is this the right PIN?" And the card says, "Yup, you got it." And then the terminal reads the information out of the card. The verification of the PIN and the reading of the information are disconnected from each other. And so some clever people figured out that they - and I have a picture here in the show notes - that they could create a very thin overlay which has only enough intelligence to respond "yes" to any PIN query.

Leo: Yeah, that's good. Yeah, that's good. Oh, that's good. I like that one.

Steve: So the bad guys get a stolen card. They just, like a piece of Scotch tape, they lay their overlay on top of the contact area. So it's very thin, enough so that it won't, like, keep it from going into the slot. Now they could freely use this chip-and-pin anywhere they want.

Leo: Oh, boy.

Steve: They walk up to the teller and stick it in and enter any PIN. Their little shim says, yup, that's the right PIN. It confirms that to the terminal, which then sucks the information out of the card.

Leo: Holy cow. Holy cow.

Steve: Doing it wrong.

Leo: Holy cow.

Steve: My lord. So, wow, yes. Incredible. And finally, Sandboxie. Our listeners will remember when I finally gave up on NoScript, on running with scripting disabled by

default. And let me tell you, was that the right choice. I didn't realize - you know how sometimes you're not aware of how much anxiety you're carrying with you? You know, basically, websites weren't working anymore. And like ordering things and logging in, you know, everything was broken. And so I had just gotten accustomed to things not working. And so I'd, okay, I'd go in and, I'd turn NoScript off for, like, only this visit, and then things would work.

Oh, now, I mean, I'm constantly having this surprise experience, whenever I go somewhere and it just works. It's like, oh, look. That's what other people experience. So when I gave up on scripting, I sort of had a backlash concern that, oh, my god, now my browser is going to be exposed. Of course I've got uOrigin now, or, I'm sorry, uBlock Origin, and now I'm happy again. But for a while I played with Sandboxie, which just was too much in the way.

Well, there are a lot of people who decided they want a virtual machine-ish, they want a sort of an external sandbox around the browser's internal sandbox. And I can understand that. And to sandbox other things, too. You know, you can sandbox your mail client so that it's prevented from having full access to the OS. So to do this, you must be aggressive. That is, the operating system was not designed with a third-party shim, essentially. This is essentially what it is, you know, we were talking about APIs, application programming interfaces, where the client program calls a function to get something done.

What an external sandboxer does is it intercepts that. The term is "hook." It hooks those functions in the operating system so that the application calls the sandbox. It thinks it's calling the operating system. It's calling the sandbox. Which allows the sandbox to inspect, from a security permissions standpoint, what's being asked. And for example, if the application made a change to the registry, the sandbox could shadow that change. It could copy the change locally and say, yeah, okay, fine. And so even when the application reads it back from the registry, the sandbox checks to see whether it has an override which it has saved to what's really in the registry; and, if so, that's what it provides. So it's able to create the faade to the application of having made changes to the operating system that never actually get out of the sandbox. They are sandboxed.

The point is it's difficult to do this. I get it. Except last Tuesday's Patch Tuesday broke Sandboxie. Microsoft made some change down at the OS level, as Microsoft has the right to do. And Sandboxie broke. So this caused quite a kerfuffle in the Sandboxie support forums. Hey, Sandboxie broke. Blah blah. What what what what?

Anyway, under doing it wrong is the official response from a Sandboxie tech support person, who wrote (his name is Craig): "I would strongly recommend you update your browsers and plugins, and any other software." Okay, that's generic. The next part isn't: "Please also turn off 'install automatic updates' in Windows."

Leo: Yeah.

Steve: Yeah, because they broke the sandbox. They broke Sandboxie. They're bad. He says: "Unless you have read the Microsoft notices on what these updates might affect and are comfortable with that." And of course that's nonsense. The notices don't tell you it's going to break Sandboxie. The notices are, you know, just say "This makes Windows better," basically.

Then he says: "The update was a kernel patch in the OS, those causing access errors

with Sandboxie." Oh, here's another one. "You can turn off 'protected mode' in your Flash plugin" - oh, yeah, there's another good idea - "in your Flash plugin in Firefox." And then he goes further, says in parens "(we recommended that in the past) to also get around the issue." And then he refers to a new beta of Sandboxie is v5.05.2. And then he says: "You also don't really need Flash plugin to run on all websites. Modern sites use HTML5 and will render just fine in Firefox without it. YouTube and others are examples." So anyway, once again, doing it wrong. The Windows Update breaks the program that functions by deeply reaching into the kernel and intercepting it and hooking it in order to do what they're doing...

Leo: Should that bother us? I thought Microsoft actually had kernel protection turned on now.

Steve: No, if you're a driver, you're running down in that...

Leo: In Ring 0?

Steve: In Ring 0, yup. And so Sandboxie does install a kernel driver, and that is able to hook anything it wants. You're trusting all of the drivers in Ring 0 to behave themselves. So, yes, you don't, you know, if installing updates breaks it, you really don't want to tell people, oh, don't install automatic updates, really. Just leave Windows the way it is. Ouch. Yeah.

Leo: And in fact the guy's wrong because, if you took the Windows 10 upgrade from 7 or 8, you don't have the choice. You have to install updates. So you can't just turn them off.

Steve: A very good point, yes, as you said, Leo. We talked about that last week, that you got updates, baby. And so what it means is that Sandboxie will break until they fix it. I mean, I'm sure they were looking at specific - the way you patch, because these are undocumented, is you look for specific patterns at the API call, and you put a jump there, and then you replace in your own code, the code you overwrote in the Windows code, so that it sort of does the same thing, but that gets control to you. I mean, they're, like, it's an unclean way of hooking, but it's because Windows wasn't designed to be hooked down at that level.

So, you know, Windows doesn't want this to happen to it. And the other thing is, notice that Windows signatures are checked. But once it's loaded in memory, then modifications can be made. And this is exactly what Sandboxie's doing. So Microsoft made a completely permissible change to their own code, which broke Sandboxie. Unfortunately, the advice was, okay, don't do updates. No, that's doing it wrong.

Leo: I haven't checked this one, but we could add a fifth to the list, if you want. Apparently Western Digital's hard drive encryption doesn't actually work.

Steve: Wow.

Leo: This comes from some infosec folks, and it was written up by Ian Thompson, our friend Iain Thompson in The Register. Could be brute forced; but anyway, there's a paper. I'll pass it along to you.

Steve: Cool.

Leo: If you're using drive encryption by Western Digital, you might want to read the paper.

Steve: Too bad, too, because WD has been at the top of the heap in terms of reliability. I think Seagate fell for a while. But Backblaze did an analysis from all of their amazing number of hard drives. And WD has been looking pretty good. I didn't like them for a while, but it looks like they're doing well again.

Leo: It looks like they only tested the My Passport series, which is maybe not - so it may not extend to all of it. I don't know. I'll let you read it and let us know next week.

Steve: We'll talk about it next week.

Leo: I am not prepared to discuss this. I just found the link from my wonderful chatroom.

Steve: In the doghouse, maybe.

Leo: Maybe. Maybe.

Steve: Yeah.

Leo: Steve Gibson's not in the doghouse. What would we do without him? He's the man at GRC.com. That's where you'll find SpinRite, the world's finest hard drive and maintenance recovery utility. You must have it. And lots of other stuff, including this show - 16Kb audio, 64Kb audio, show notes, and a fully English handwritten transcript of each and every episode. GRC.com.

Steve: Boy, and Elaine does such a nice job.

Leo: Thank you, Elaine. Did she write the part where you did last week where she types this, and she'll be typing this, and...

Steve: I'm sure she did. I'm sure she was chuckling while she was stuck in an endless

loop.

Leo: We have to figure out a way to do some recursion on her and really mess things up. You can also find a lot of free stuff there. And I guess we're getting close to SQRL becoming a reality?

Steve: Yup.

Leo: So you might want to read up on that.

Steve: Storage format is finalized. The protocol is finalized. And I'm just catching the protocol docs up because there are a lot of upstream changes that I need to now get propagated, so we have the finished document. And then I just - I implement those things in my code, and we're rolling, baby.

Leo: Boy, I just - I want everybody to adopt it because we want to get rid of passwords.

Steve: Yeah.

Leo: It's clearly failed us.

Steve: Yeah. What I think will happen, something I didn't realize, another mode of use for SQRL, I've received lots of inquiries from corporations that just want to use it for themselves.

Leo: Right.

Steve: And I realized, yeah, it's not, I mean, it is - one of the beauties is it can be incrementally adopted because it can live alongside traditional login. But if it's something, as long as you've got support on the platforms your employees are using, and we do, then it doesn't have to be universal. It doesn't have to be publicly adopted to be useful. Corporations are wanting it because they just - they want to solve the problem for themselves. They don't want to deal with password resets, and I forgot my password, and all that nonsense. So I think it's going to get some traction. I'm just going to push it, push it across the finish line.

Leo: Good, good, good. You can also come here and get all your shows. We do on-demand versions of everything. We do video and audio at TWiT.tv, in this case TWiT.tv/sn. Of course subscribing in your favorite podcatcher works, too. And that's probably the best thing to do. That way you don't miss an episode.

We'll be back next Tuesday at about 1:30 Pacific, 4:30 Eastern time, 20:30 UTC,

that's when we record. I hope you'll tune in live because we appreciate, as you know, we appreciate the chatroom's contributions to the show. And Steve, I hope you have a great week, and I'll see you then.

Steve: We'll do a Q&A next week, unless the world collapses in the meantime. And it's happened before. But we'll hope things stay kind of quiet.

Leo: Well, then I should explain, if you want to ask Steve a question, do not email him. Go to GRC.com/feedback, or tweet him, @SGgrc. And that way, 140 characters, you can ask your question, and Steve will perhaps answer it next week.

Steve: Actually, you can do a DM because I accept DMs.

Leo: Oh, that's right. Unlimited DMs.

Steve: Yup.

Leo: Thank you, Steve.

Steve: Thank you, my friend.

Leo: We'll see you next time.

Steve: Bye-bye.