SECURITY NOW!

Transcript of Episode #698

Which Mobile VPN Client?

Description: This week we examine a very worrisome WiFi bug affecting billions of devices; a new fun category for the forthcoming Pwn2Own; Russia's ongoing, failing, and flailing efforts to control the Internet; the return of the Anubis Android banking malware; Google's changing policy for phone and SMS app access; Tim Cook's note in Time magazine; news of a nice Facebook ad auditing page; another Cisco default password nightmare in widely used, lower end devices; some errata, miscellany, and listener feedback. Then we answer the age-old and apparently quite confusing question: Which is the right VPN client for Android?

High quality (64 kbps) mp3 audio file URL: http://media.GRC.com/sn/SN-698.mp3
Quarter size (16 kbps) mp3 audio file URL: http://media.GRC.com/sn/sn-698-lq.mp3

SHOW TEASE: It's time for Security Now!. Steve Gibson is here. We'll talk about the upcoming Pwn2Own contest. You won't believe what the hackers are being asked to attack next. We also have details on Steve's favorite OpenVPN client for Android, and a good reason to be very careful about which banking tools you use on Android. It's all coming up next on Security Now!.

Leo Laporte: This is Security Now! with Steve Gibson, Episode 698, recorded Tuesday, January 22nd, 2019: Which Mobile VPN Client?

It's time for - oh, wait a minute, let me make sure I'm recording. I am.

Steve Gibson: Oh, please do press the red button.

Leo: Did I ever not record Security Now!?

Steve: Once.

Leo: Once.

Steve: Once, actually maybe twice in the entire 13-year history we would get all done, and then it was like, ooh. But, you know, I have to say, Leo, the second time was better than the first time.

Leo: It often is, yeah.

Steve: It was painful, but we did have a dry run rehearsal, so...

Leo: So sorry. Well, that will never happen again because I am no longer in charge of recording shows.

Steve: And it hasn't happened for a long time.

Leo: No. No, at least a decade. I remember the worst one was a Floss Weekly episode where I erased it twice, and I had to call the guy back for a third time. But we are recording.

Steve: So we are closing in on Episode 700. We're at 698 today, the last podcast - wait, wait, no.

Leo: It has been a long time.

Steve: It'll be the 29th will be the last podcast of the month, so next Friday.

Leo: Your first podcast in February.

Steve: So a piece of news that Bleeping Computer ran caught my attention because it was about the 150 bad VPN clients for Android in the Google Play Store. And I thought, what? Well, first of all, that there were bad ones, I guess, wasn't that big a surprise. But that there were 150 VPN clients, just I couldn't believe it. So there is only one. One. And I thought, okay, we're going to call this podcast "Which Mobile VPN Client?" We're going to answer that question once and for all for the ages, Leo, if you want to use a VPN, what client should you use and, of course, what VPN. So that's what we'll get to.

But there was a lot of news. There was an extremely worrisome WiFi bug which affects, if we believe Marvell that makes the chip which was found to be defective when they were boasting in their marketing material about their chip being in billions of devices, then this WiFi bug could not be worse, and it's in billions of devices.

Leo: Oh, my goodness.

Steve: It's not good. So we're going to start with that. Then we've got a new, and you're going to love this one, this one is like in your wheelhouse, a new fun category for March's - the end of March is the forthcoming Pwn2Own competition. And you might be able to guess. But anyway, definitely fun new topic or new category for Pwn2Own.

We've got Russia's ongoing failing and flailing efforts to control the Internet, and they're now setting their sights on two new players. We talked previously, remember, about how they tried to block Telegram and what a disaster that was, that ended up blocking ranges of Google and Amazon's cloud services and making a mess of things. Well, they're now going after a couple more players, probably without much more success.

We've got the return of the Anubis Android banking malware, which was found to be using a clever new trick to avoid detection that I just had to share with our listeners because it's like, okay, this, I mean, we really are in an unending cat-and-mouse game here. We've got Google's announcement of their changing policy for phone and SMS app access. They're clamping down. I also wanted to talk about the first thing you guys talked about on MacBreak Weekly, which is Tim Cook's note in Time magazine.

Leo: Oh, yeah, good, yeah.

Steve: We also have news of a nice Facebook ad auditing page I wasn't aware of. But it resulted in the creation of the first bit.ly link I have created in a long time, snfbads. I tried to not have "sn" in there, but every kind of variation of Facebook or FB or anything was already taken. So it's bit.ly/snfbads, for anyone who wants to jump ahead. And so we're going to talk about that.

There's another, believe it or not, Cisco default password nightmare. But this time not in some obscure high-end switch that no one's ever heard of, but in widely used, lower end SOHO consumer devices. We've got some errata. I said something last week I'm just so embarrassed about, I can't believe I said it. But we're going to fix that. We have some miscellany, some listener feedback, and then we're going to answer the age-old and apparently quite confusing question: Which is the right VPN client to use for Android? And once we get back from our first break, we need to talk about Oscar's latest DEC PDP reconstruction resurrection because, boy, this guy has just done an amazing job.

Leo: Nice. Well, we've got a full lineup of stuff to talk about. I can't wait. Picture of the Week?

Steve: So Oscar is someone I've mentioned before.

Leo: Yeah.

Steve: He's this amazing craftsman. He produced a clone of the venerable DEC PDP-8 initially. And I own several of them. I talked about it.

Leo: They're over Steve's right shoulder right now.

Steve: Well, no. Those are actually different. Those are, shoot, I can't...

Leo: Is that that kit that you built?

Steve: Well, Oscar's is different. The ones over my shoulder were based on the Intersil 6100 chip, which was an actual PDP-8 on a chip.

Leo: Oh.

Steve: The problem is those are all gone. They're like hen's teeth now. Do hens have teeth? Anyway.

Leo: They don't. That's the whole point.

Steve: Ah.

Leo: Get it? Rarer than hen's teeth?

Steve: I get it.

Leo: They've got gullets.

Steve: So the problem is, if you can't get a PDP-8 on a chip, what are you going to do? Well, what you're going to do is you're going to emulate the machine on a Raspberry Pi.

Leo: Of course. Which is probably more powerful than a PDP-8 was.

Steve: Oh, my god, yes.

Leo: A lot.

Steve: Yes. Way faster. And in fact there is a whole underground, well, I guess they're not really under the ground, but they're off to the side. They're out of the mainstream, that's what we want to say. They're nonmainstream. But they're fascinating people who are keeping old machine architectures alive by creating mature simulations of their architectures. I mean, to the point where they can run the original operating systems and mount disk packs, virtual disk packs, and do everything. Like you used to with the raised floor in the air conditioned environment and everything.

Leo: Wow, that's awesome.

Steve: Okay. So Oscar's first piece of work was a PDP-8. This one is a PDP-11.

Leo: This is beautiful. This is gorgeous.

Steve: It is just stunning.

Leo: Oh, man. Including the toggle switches on the front.

Steve: He did custom injection molding for the toggle switches and the panel. The whole thing is an exact duplicate of the PDP-11/70. And in fact, on his page, it's Obsolescence is his website, .wixsite.com. I have a link in the show notes for anyone who's interested. And for some of our listeners you will be when you understand - and I don't understand this at all, Leo. I don't get it that he wants to sell these kits for $250. I mean, he is selling them for $250.

Leo: Wow.

Steve: Apologizing. He says: "I apologize because that's almost $90 more than the PDP-8. But if you knew the upfront cost of making that injection-molded case..."

Leo: Oh, yeah.

Steve: "...you would understand it's not because I am turning commercial. This is still very much a hobby project, although one that got slightly out of hand."

Anyway, okay. So again, there's a Raspberry Pi behind the panel. So what he did was he exactly duplicated the functioning of the front panel, you know, blinking lights and switches.

Leo: Oh, I've got to get one of these. This is amazing.

Steve: It is just gorgeous. Anyway, so he sat with a buddy in front of an actual 1170 and, like, pushed switches while video recording in order to get any edge cases of the way the actual 1170 front panel operated so that he could exactly emulate it. And then look at the screen on his laptop. That's - he, like, faked a CRT bezel on his ThinkPad to show the way a CRT would look with the bezel for the actual console when you're logged in and giving commands to the PDP-11.

Now, okay. So there's elegance to both the 8 and the 11. The elegance of the 8, the PDP-8, was could you make a computer with five AND gates. Okay, well, not quite, but still. Remember that the opcode was three bits on a PDP-8. So you had a total of eight instructions, and there's no subtract. You have to jump through hoops to pretty much do anything. So the PDP-8 is arguably a perfect platform if you just kind of wanted to get started to show someone the concept of ones and zeroes and with a few instructions how you can do something. And frankly, I'm stunned by the fact that there's OS/8 is an operating system for the PDP-8, written with eight opcodes, I mean, eight instructions. So, okay.

But the 11 is different. The 11 is the other end of the spectrum. It is a stunningly elegant instruction set. And it was the birthplace of Unix. It was on a PDP-11 that the C language was born. Thompson and who was the other guy...

Leo: Kernighan. Kernighan.

Steve: Kernighan did C. Thompson...

Leo: Ritchie.

Steve: ...and Ritchie, yes, did Unix. And they wrote Unix, toggling it in through the front panel because you had to start somewhere.

Leo: I have their books, you know, the Unix programming language and the C. I should get this. I even have the source code. You know they released the source code in a big bound book. That would be fun.

Steve: Well, and what's so neat about this is that it's not just a front panel that does nothing. You get all of the PDP-11 code and source and OS and Unix and everything the way it was back then. For $250.

Leo: I'm ordering it right now. Wow.

Steve: You do have to add a Raspberry Pi, but what are those, 40 bucks or something.

Leo: Yeah, 35. I've got one lying around. Most people do these days.

Steve: Yeah.

Leo: Wow.

Steve: Anyway, and so I just, you know, I'm afraid we're going to bury Oscar in orders. He is sending me one because I have to have one of these. It's just stunning. So I just wanted to share. I know from the past that our audience was interested in the PDP-8 and that my talking about it and just, I mean, once this is gone, it's gone. I mean, we don't know that nobody else in the future is going to ever do this again, but I don't think anyone's going to do it like this. This is a work of art. I mean, this is just a masterpiece. So take a look at the show notes. It's PiDP-11. I didn't check to see whether Google would find it if we put in PiDP-11. Let's see what happens. No, didn't come up.

Leo: I think if you just type "obsolescence" and "PiDP-11," something like that, because the site is obsolescence.wixsite.com. So I bet, if you added the word "obsolescence," that would find it.

Steve: Yup, that brought it up. "Obsolescence PDP-11."

Leo: That's awesome.

Steve: And anyone can find it that way. And oh, my god, it's just - what a toy. And so...

Leo: I just added my name to his email list because you can't order it directly. You have to email him and all that. Yeah, that would be fun to have here.

Steve: He's a good guy. And oh, yeah, I need more blinking lights behind me, Leo.

Leo: Well, I tried to get the 8, and he forgot to send it to me. And then by the time he remembered it was gone. So Oscar, I want one of these. And I'll send you the money, don't worry.

Steve: I'm sure Jason could throw this together and get it up and going and have some fun with it.

Leo: Oh, it'll be fun, yeah, yeah.

Steve: Okay. So on a much more serious note, we have an incredibly widespread WiFi firmware bug which by all measure affects probably billions of devices. The title of Embedi's disclosure was "Remotely compromise devices by using bugs in Marvell Avastar Wi-Fi: From zero knowledge to zero-click RCE."

Okay. So this is one of the most popular, if not the most popular WiFi chipset on the market. In digging around, I wanted to find out whether it was possible for its firmware to be reflashed, and I could not get an answer. You have to go under NDA with them in order to get access to the pinouts and the programming of the chip. But it's highly unlikely that an SOC, as this is called, a System on a Chip, has reflashable firmware because there would just be no need for it. Or even if it did, you probably can't get to it from the main processor of the system that it's mounted in. You would need to physically hook on a JTAG debugger in order to get to reflash the firmware, if it's even possible. So we're probably all - "we" the industry, the world - stuck with this WiFi chip as it is.

But in the process of digging in to see what I could find, I discovered, I mean, it's been selling for a decade. So, I mean, this particular chip, the 88W8897, it's everywhere. It's in the Sony PlayStation 4, the Xbox One. Microsoft's Surface laptops use it. Samsung's Chromebooks use it. The Samsung Galaxy J1 smartphones, the Valve Steam Link cast devices, some other laptops. There are consumer routers and embedded devices and other network hardware. So, I mean, it's sort of the go-to WiFi chip. What drives it is an embedded RTOS, a real-time operating system known as ThreadX. And there's been some confused reporting about whether the bug is in ThreadX or in the implementation of it. And it looks like it's actually in the implementation, in the code that was written for this real-time operating system.

So the story begins late last spring, when a researcher, Denis Selianin, who is with the embedded security firm Embedi that I mentioned before, he was experimenting with fuzzing that very highly popular WiFi chipset. And we've talked about fuzzing before. The idea is in some cases you can't find problems by inspection. So what is done is you sort of automate the process. You just throw a whole bunch of stuff at it. And if something you throw at it makes it crash, then you go, oh, what just happened?

So the idea is that's the process known as fuzzing, where you record what you're throwing, and you also watch to see if what you threw crashed the chip. And, if so, then you say whoa, and you back up, and then you fuzz again, verify that it's reproducible, and then from there - so basically you're using just a random process to find edge cases, things that the programmers missed. So he was doing that. And I saw the log. It's like for four point some days of this. And then he found a problem, dug into it, figured out what was going on, ended up finding four problems, two of which are critical. And when we say "critical," we're not kidding.

So if any of these WiFi devices that I mentioned are powered up, the bugs he found would allow malicious attackers to force them to execute arbitrary code of the attacker's choice without requiring any action on the part of the device's owner. In other words, this could not be worse. Well, okay. It would be worse if you could do it from Russia. The good news is, well, sort of the good news is it is WiFi, so you need to be within WiFi range.

Leo: Well, that's good.

Steve: Well, okay, yeah. So it means that it's not...

Leo: It's more of a concern for a business than an individual, though.

Steve: True, well, yeah, unless - well, certainly you could be targeted. And if you went to Starbucks there's a lot of, I mean, the idea is that any device with this chipset that receives this malformed message will execute code.

Leo: So is this chipset in a WiFi access point? Or it's in computers; right? This is a...

Steve: Both.

Leo: It's both.

Steve: It's like, yeah.

Leo: It's everywhere.

Steve: It's widely used. It's MIMO. It's Bluetooth. I mean, it's a super popular chip because these guys did a beautiful job, except there's a little bug in the firmware of the chip. So one of these bugs is specific to this particular 88W8897 WiFi controller. But the other bug may be based on the ThreadX operating system. Now, details are being withheld because he did then inform - Embedi informed Marvell, I think it was in May, yeah, May of 2018 that they had found the problem.

But again, so here's the problem is that it could certainly be the case that chips moving forward will have this fixed. But there is almost, I mean, I don't know for sure. I could not get an answer. Everybody is being tight-lipped about this because this is really bad. This is in billions of devices all over the world. And the reason we're not hearing more about it yet is that proof of concept has not been released. There is a demo of this happening. The guy showed it at a conference in November. So there's a demo of it happening online, but no proof of concept required.

He says it requires no user interaction. It can be triggered in - oh. What happens is every five minutes the typical WiFi chip reaches out to enumerate WiFi signals, WiFi base stations, access points, users, whatever, within its range. So at that point the device could be compromised. It requires no knowledge of the WiFi network name or passphrase or key. You need to know nothing about it because it's such a low level. It's down at the low level, you know, the ether link protocol level of the chip and gives the attacker arbitrary code execution on this WiFi SOC, this system on a chip.

Leo: This is in the Surface computers.

Steve: It is, yes, it's in the Microsoft Surface machines.

Leo: Oh, boy. Oh, boy.

Steve: Yeah, and so here's the problem is it's not clear this can be fixed. I mean, I don't know one way or the other. I've got a link in the show notes to the demonstration proof of concept video.

Leo: It's in my Xbox One.

Steve: Yeah.

Leo: Oh, my god.

Steve: Yeah.

Leo: The J1 smartphones, fortunately, the Galaxy J1's aren't super popular. But Samsung Chromebooks are. Surfaces are. Xbox Ones and PlayStation 4's are.

Steve: Yeah, uh-huh.

Leo: Holy cow. Yikes.

Steve: So I have a feeling we will be following this story for the balance of the year. The problem is that there's enough information that is already disclosed that it's probably possible for bad guys to independently follow in the footsteps of Embedi and reconstruct this. And once they do release something, once Marvell has some sort of response to this, these guys have said they're going to release a proof of concept and the details of their tools. But the problem is we know billions of devices are not going to get their firmware updated. And this is just so juicy, the idea of no click, no authentication, over the air. You don't have to physically touch a device. You just have to zap it with the proper packet when it reaches out, and you can take it over. It's going to be, just for like the real hackers, it's going to be - they're not going to be able to control themselves. We will see exploits for this.

Leo: So I could go sit in a coffee shop. Now, nobody - of course, the code's not out there. But when you say "owned," could I just start capturing all the packets coming out of that device, for instance?

Steve: Yes, for instance, exactly.

Leo: Okay. Okay.

Steve: And you could also turn around and then access the system that that WiFi chip is sitting on because the chip itself is on the bus. So you could then access probably main memory of that chip and rifle around and get private keys and so forth. Because we don't want anybody in our main memory.

Leo: Yikes.

Steve: Yeah. Because the way the architectures are now, the bandwidth of these chips is so high that they have direct DMA access into main memory, that is, when they're receiving data, it is being streamed directly into the main memory of the system, which means that the chip itself...

Leo: They can read it. They could read it.

Steve: Exactly, has access.

Leo: Oh, man.

Steve: Yeah.

Leo: Yikes. So that's worse than capturing your packets. That's seeing everything that's going on.

Steve: Yeah. They can rummage around. I have a feeling this is going to be a field day for the bad guys. And in a lot of machines that, as you said, that people have. You have a bunch of them already.

Leo: I do. I have all of them. Oh, boy.

Steve: Okay, Leo. Get this. Pwn2Own adds a Tesla Model 3.

Leo: Oh, crap. Please don't pwn my Tesla.

Steve: No, that's good news because you want these guys to find and fix the bugs.

Leo: Oh, yeah.

Steve: And in fact, if they do, $900,000 worth of prizes for the Tesla. So there's money in them thar hacks. This will be at March's forthcoming - I think it's March 2022nd - forthcoming, the CanSecWest in Vancouver, which we talk about every year because it's so much fun. It will for the first time add automobiles to its hacking target. So the biggest prizes will be a quarter million dollars for hacks that execute code on the Tesla Model 3's three primary systems. There's something that's generically known as the "gateway," there's the autopilot, and the VCSEC. The gateway is the central hub that interconnects, as it sounds, as you would expect, the car's power train, the chassis, and other components; and processes the data that they are exchanging. The autopilot, of course, is the driver assistance feature that helps control lane changing, parking, and other functions. This VCSEC is the Vehicle Controller Secondary. That's the term used. It's responsible for security functions, including the car's alarm.

So these three systems represent the most critical parts of a Tesla, so it's clear why hacks successfully targeting them would be eligible for big payouts. To qualify, the exploits must force the gateway, autopilot, or this VCSEC to communicate with a rogue base station or other malicious entity. Then stepping down from that, a denial of service attack which could take out, for example, the car's autopilot, will pay $50,000. So they want to know if you can do that. But that's a step down from forcing one of those three critical systems to have a communication with a malicious external entity.

Also, Pwn2Own will pay $100,000 for hacks that attack the Tesla's key fob or phone as key, either by achieving code execution, unlocking the vehicle, or starting the engine without the key. It'll pay $100,000 add-on prize for winning hacks in another category that attack the car's Controller Area Network. We've talked about this in the past, the so-called CAN bus, which is the bus that interconnects literally everything. It's no longer the case that there's a big wiring harness like you and I used to have in the first cars that we owned, where you could peel back the floorboard mat, and you'd see this huge bundle of wires going to the taillights and the brake lights and the backup lights and the turn signals and everything. That's all gone. Now everything just gets power and a signal, and all of the light-on, light-off stuff is multiplexed over this CAN bus.

So hacks targeting the car's infotainment system is also on the menu, which will earn a successful exploiter $35,000. And hacks which escape the security sandbox or escalate privileges to root or access the car's OS kernel will fetch $85,000. WiFi or Bluetooth hacks get $60,000. And there's a separate add-on payment of 50 grand which will be paid for winning hacks that achieve persistence - which means, of course, that once they get in, they set up shop in your car and survive a reboot of the system. So anyway, at the end of the month, let's see, the Tuesday following happens to be my birthday, March 26th. So as I turn 64, we're going to find out about how the Tesla Model 3 withstood the CanSecWest Pwn2Own. And of course there's always, well, as always, there are attacks against things without wheels. We have a quarter million dollars for a success - boy, these contests are...

Leo: This is a business. You've got to get in this business. There's some money in there.

Steve: They're making some money. Quarter million dollars for a successful Hyper-V client guest-to-host escalation. And respectively, $150,000, $70,000, and $35,000 for hacks of VMware's ESXi, VMware Workstation, and Oracle VirtualBox, respectively, so stepping down. The web browser attack category will pay $80,000 for hacks of Chrome and Microsoft Edge, and in the case of Edge, with a Windows Defender application guard specific escape, so you've got to get out in order to do something. And a Firefox exploit will net $40,000. And the coverage of this, Trend Micro puts this on. The server-side category is much smaller this year, with Microsoft Windows Remote Desktop Protocol, the RDP, as the only target.

Oh, and Trend Micro noted that most of their server-side targets had moved to their targeted incentive program, so they're no longer needed to be in Pwn2Own. They've sort of changed where that's being done. But still, a successful exploit of Remote Desktop Protocol will bag its user or its exploiter $150,000. Of course, that's significant because we have been talking about cross-Internet RDP exploits, which it's surprising to me how many people have Remote Desktop Protocol exposed, and it's just not secure enough to let people get to it. So at the end of March we'll have some news about how all that's going.

Russia is not doing so well with blocking Internet services they dislike. As we discussed at the time, back in April, and as I mentioned at the top of the show, the Russian agency responsible for censoring Russian citizens' access to the Internet - and I can't pronounce this, Roskomnadzor.

Leo: Exactly right.

Steve: Thank you, Leo.

Leo: Perfect.

Steve: They attempted to block, as we know, Telegram, after Telegram ignored their threats of blocking the service. We'll recall that, after that initial block, Telegram moved their servers into the cloud network space, you know, being served by Amazon and Google, which resulted in Russia blocking wide swaths of IPs, which blocked many more critical services than just Telegram. So it turns out it's easier said than done to block a service. And of course, for their part, Telegram users evaded the blocking by using VPNs and various available proxy services, after which Russia again countered by expanding its block list and ended up blocking even more.

Oh, and Reuters later reported, in August of 2018, that Russia then started testing, and kind of went off of the news map, "more precise technology to block individual online services," though really it didn't make much news after that. And way back before that, back in 2016, Russia had also attempted to block LinkedIn with limited success. So they want control over what happens in their country. And the problem is, as I said, it's easier said than done. So we're talking about this now because Russia has now set their sights on Facebook and Twitter.

Leo: Oh, boy. Oh, good luck.

Steve: Yeah, I know. Exactly. Good luck. Roskomnadzor...

Leo: Roskomnadzor. If you say it like that, it just rolls off the tongue. Roskomnadzor.

Steve: Much better, Leo. Last December 17th that agency who you just mentioned convincingly sent letters to both Facebook and Twitter accusing them of failing to comply, as indeed they were, with a law requiring all servers that store personal data to be stored in Russia. They just said "Eh. No." The letters gave each company 30 days to provide a, quote, "legally valid response," unquote. Well, that time has passed. That was up last week. And neither company even bothered to reply.

So the Wall Street Journal now reports that, today, that agency begins administrative proceedings against both companies. The Russian censorship agency said: "The social media networks hadn't submitted any formal and specific plans or submitted an acceptable explanation of when they would meet the country's requirements that all servers used to store Russians' personal data" - I mean, come on, give me a break, for Russian privacy's sake - "to store Russians' personal data be located in Russia." Russia has previously threatened to block Facebook over its ongoing noncompliance with this data storage law in 2017 and in 2018. So this is not new.

Anyway, this should provide some interesting fodder for the podcast because, I mean, in the case of Telegram, you would think that would be more easy. And, you know, I wouldn't be surprised to learn that these large global mega services had deliberately adopted network architectures.

Leo: Oh, yeah. They're multi-homed. They're all over. Yeah.

Steve: Exactly, that made blocking them extremely difficult.

Leo: By the way, Roskomnadzor is the FCC of Russia. And they call it RKN, if you just want to make it simple for yourself.

Steve: Roskomnadzor. See, now that you said it right, I can see it.

Leo: Yeah, it's easy. Everybody knows Roskomnadzor.

Steve: Roskomnadzor, of course, Roskomnadzor.

Leo: In Soviet Union, Roskomnadzor calls you.

Steve: You taught me how to pronounce Huawei, Leo, and I held onto that.

Leo: I have no idea if I'm saying it right. I'm just spelling it out. But...

Steve: I like it.

Leo: Literally, RKN is allowed.

Steve: Okay, good. I'm going to name a podcast, one of our episodes, Roskomnadzor.

Leo: And then you'll have to say it.

Steve: Mark my words. When we have our big story about the failure of Russian censorship, "Roskomnadzor" we will proudly proclaim.

Leo: It's funny because we think of the great firewall of China. I mean, China blocks Twitter and Facebook. You know, you figure, well, it must be doable. But of course that requires a country where the entire Internet access goes through the government.

Steve: Yeah.

Leo: And Russia's not, I guess, in that position yet. Yet, I say, yet.

Steve: Yeah, well, and, I mean, the fact is the Internet was designed to resist this.

Leo: Yes.

Steve: I mean, it was, like, on purpose.

Leo: Routes around damage. That's considered damage; right?

Steve: Exactly, yeah, yeah. So speaking of damage, Anubis is the banking malware you really don't want in your Android smartphone if you do any online banking with your smartphone. Which really, you know...

Leo: Who doesn't?

Steve: I would - yeah, okay. So get this. Trend Micro found two instances of this Anubis malware using what they called "motion-based evasion tactics." So first of all, this Anubis trojan has been observed, I mean, it is very competent. It has been observed to attack 377 different bank applications, from 93 countries around the world, aimed at a bank I've never heard of, Santander?

Leo: Santander is, yeah, no, that's a big bank.

Steve: Okay. Santander, RBS, NatWest, Citibank, as well as non-banking apps such as Amazon, eBay, and PayPal, among obviously many others. So it is an aggressive and capable banking malware trojan. Trend, as I mentioned, recently discovered it hiding inside two Android Google Play Store apps, which each had dozens of fake five-star ratings and thousands of installations into Android devices. So this thing had gotten into people. So the two apps are Currency Converter and BatterySaverMobi. So again...

Leo: Don't install those two.

Steve: Yeah, be a little skeptical, yes. What sets these apart from other malware-carrying Android apps is their use of their host's motion sensors to detect whether they've been installed in a malware analysis sandbox, in which case their malicious behavior is suppressed, and they behave themselves. Isn't that clever?

Leo: So if it's in a sandbox, it doesn't get carried around. It's not like a real phone; right?

Steve: Exactly, and you don't have some random motion sensor sending data, accelerometer data, into the OS. So that'll just be sitting there doing nothing. Now, again, this is all a cat-and-mouse game because, now that the sandbox guys know this is a possibility, it's like, ah, we'll add emulation of motion to the motion-sensing API in the sandbox to trigger motion-sensitive awareness in malware and be able to catch this happening. But again, this is why it's just like this back-and-forth game. Creating a secure system, which we keep seeing that creating a secure system which is also powerful and flexible is something which has so far eluded the best brains in computer science. Because you could argue, you know, this is an important thing. And we haven't figured out how to do it yet.

We know that we could have a system which is closed and secure, like the original Apple iPhone. But people want apps. We want open stuff. I certainly do. And we want capable apps. But so far, with app capability comes app liability and exploitability. And despite all of our best efforts, we haven't yet figured out how to get the one that we want, which is capability, without inviting the others we don't want. And then there's social engineering. I put on the previous page of the show notes, Leo, this is a dialog that this Anubis app presents because it needs...

Leo: It would fool me. It would fool me.

Steve: Yes. It would fool anybody. Yes.

Leo: It's a system update dialog which you see all the time on Android.

Steve: Exactly. And when you click it, you're giving the Anubis malware admin permissions on your system. So, I mean, the very fact that that is possible demonstrates that we're not there yet because who would not click that? It'd be like, oh.

Leo: I'd click it, yeah.

Steve: Yes, yes. Anybody would.

Leo: Now, you have to download one of these apps onto your phone from the Play Store? Or, no, they're not in the Play Store.

Steve: Yes, they're in the Google Play Store.

Leo: Not any more they're not. They can't be.

Steve: Well, no. They've been yanked. But thousands of people did download them from the Google Play Store, where they were, with fake five-star ratings.

Leo: And they do have to ask permission, so they pop this up.

Steve: Yes.

Leo: And then I presume you get a legit permissions dialog. But you assume it has to do with a system update, so you say okay.

Steve: Yeah, I mean, who knows, yeah, exactly.

Leo: Wow.

Steve: Yeah, I mean, so the problem is, of course, social engineering. We're going to click on that, and that's going to give this thing rights. So we have a ways to go before we figure out how to do this stuff in a secure fashion. And to that end, Google is cracking down on Android phone- and SMS-using apps, that is, apps that ask for phone and SMS permissions. Last Monday Paul Bankhead, who's the director of product management for Google Play, posted to the Android Developers Blog, so this is written, as you will hear, to developers. But I wanted to share this with our listeners because I'm glad Google is taking these steps. It's going to cause problems; but, unfortunately, we've been too permissive in the past.

So his first little paragraph, TLDR, says: "As previously announced and directly communicated to developers via email" - in other words, don't blame us, we warned you - "we'll be removing apps from the Google Play Store that ask for SMS or Call Log permission and have not submitted a Permissions Declaration Form. If you have not submitted a Permissions Declaration Form, and your app is removed, see below for next steps."

And then he explains, he says: "We take access to sensitive data and permissions very seriously." Well, yeah. They do now, or increasingly do. He says: "This is especially true with SMS and Call Log permissions, which were designed to allow users to pick their favorite dialer or messaging app, but have also been used to enable many other experiences" - I would, like, put air quotes there - "'experiences' that might not require that same level of access. In an effort to improve users' control over their data, last October we announced we would be restricting developer access to SMS and Call Log permissions.

"Our new policy is designed to ensure that apps asking for these permissions need full and ongoing access to the sensitive data in order to accomplish the app's primary use case, and that users will understand why this data would be required for the app to function." And of course we've also often talked about the overbroad, like you install an app and it needs access to all kinds of stuff that seem completely tangential to the app's intention or purpose, or like why would it need that.

So, he says: "Developers whose apps used these permissions prior to our announcement were notified by email and given 90 days to either remove the permissions, or submit a Permissions Declaration Form to enable further review." Anyway, he says: "We take this review process seriously and understand it's a change for many developers. We apply the same criteria to all developers" - so you know they're not taking sides - "including dozens of Google apps. We added to the list of approved use cases over the last few months as we evaluated feedback from developers." So in other words, they allowed developers to explain why they needed access to this, even though you wouldn't at first blush maybe have thought so.

He says: "Our global teams carefully review each submission. During the review process, we consider the following five points. First, likelihood that an average user would understand why this type of app needs full access to the data. Second, user benefit of the feature." That's really good. I mean, I hope Google's really, I mean, it sounds like they're really going to honor this, and they should, the user benefit of the feature. "Third, the importance of the permission relative to the core functionality of the app. Four, risks presented by all apps with this use case having access to this sensitive data," meaning if everybody asks for this, what does that mean? "And then, fifth, availability of more narrow alternatives for enabling this feature." In other words, is there a better way to achieve the same thing that doesn't require giving this permission? So this is all for the best. This is great.

They said: "With this change, some use cases will no longer be allowed. However, many of the apps we reviewed with one of these permissions can rely on narrower APIs" - in other words, they were just, you know, the developer was kind of lazy and said, yeah, just give me all this - "reducing the scope of access while accomplishing similar functionality. For example," he writes, "developers using SMS for account verification can alternatively use the SMS Retriever API, and apps that want to share content using SMS can prepopulate a message and trigger the default SMS app to show via intents." In other words, that's a perfect example of, instead of just getting global access to SMS, use a specific narrow API designed to do just and only that aspect, the SMS account verification through this SMS Retriever API.

He says: "Tens of thousands of developers have already resubmitted their apps to support the new policy or have submitted a form. Thank you. Developers who submitted a form received a compliance extension until March 9th." And he goes on about next steps. But anyway, I just think this is great. I mean, this feels like lessons learned from the road and from the real world. And it's always painful to take things away which had previously been given. But Google is learning through interaction with their platform out in the world that, ouch, we need to take this more seriously. There are all kinds of clever ways we hadn't thought of for the things that we were permitting apps to do to abuse those permissions. So we're going to create narrower APIs first, and then we're going to force apps to use those, or to explain to us why they can't get by with using those. So I just, you know, yes. It's what we need. So props to Google for that.

As I mentioned at the top, Leo, you guys started MacBreak Weekly this week talking about Tim Cook's note in Time magazine. You read it into the podcast. I'm going to read it into ours because it's short, and my take is a little different. First of all, I couldn't understand everybody's take on MacBreak because everyone, it was such a hot topic, everyone was talking at once.

Leo: I know.

Steve: And it was like, whoa, okay, what happened?

Leo: Everybody had to get their word in there.

Steve: What happened? So, okay. So last week Time magazine printed a statement by Apple's Tim Cook which took aim at - and this is I think the significant part of this. This is not Apple versus Google and Facebook, which is how - it's so easy to paint that with a broad brush because that meme has been established. What Apple did, what Tim Cook did was take aim at what is the largely hidden data brokerage industry, which has quietly sprung up over the past decade. And we've touched on it on this podcast from time to time.

Sometimes as I'm doing some research I'll encounter one of those chilling websites where they tout everything they know about us because they're trying to sell this intelligence gathering capability to an audience different than ours. And I often will share that on the podcast because it just kind of gives me the creeps. So I think that Tim raises some important points. And as I said, this is not the accepted Apple versus Facebook and Google profit models where Apple is saying we don't profit from the collection of your data, but they do. And that's not what he wrote.

So here's what he said. He said: "We all deserve control over our digital lives. That's why we must rein in the data brokers. In 2019 it's time to stand up for the right to privacy - yours, mine, all of ours. Consumers shouldn't have to tolerate another year of companies irresponsibly amassing huge user profiles, data breaches that seem out of control, and the vanishing ability to control our own digital lives. The problem is solvable. It isn't too big, too challenging, or too late. Innovation, breakthrough ideas, and great features can go hand in hand with user privacy, and they must. Realizing technology's potential depends on it. That's why I and others are calling on the U.S. Congress to pass comprehensive federal privacy legislation, a landmark package of reforms that protect and empower the consumer.

"Last year, before a global body of privacy regulators, I laid out four principles that I believe should guide legislation: First, the right to have personal data minimized. Companies should challenge themselves to strip identifying information from consumer data or avoid collecting it in the first place. Second, the right to knowledge, to know what data is being collected and why. Third, the right to access. Companies should make it easy for you to access, correct, and delete your personal data. And, fourth, the right to data security, without which trust is impossible."

He says: "But laws alone aren't enough to ensure that individuals can make use of their privacy rights. We also need to give people tools that they can use to take action. To that end, here's an idea that could make a real difference. One of the biggest challenges in protecting privacy is that many of the violations are invisible. For example, you might have bought a product from an online retailer, something most of us have done. But what the retailer doesn't tell you is that it then turned around and sold or transferred information about your purchase to a 'data broker,' a company that exists purely to collect your information, package it, and resell it to yet another buyer. The trail," he writes, "disappears before you even know there is a trail. Right now, all of these secondary markets for your information exist in a shadow economy that's largely unchecked, out of sight of consumers, regulators and lawmakers."

And he finishes: "Let's be clear. You never signed up for that. We think every user should have the chance to say, 'Wait a minute. That's my information that you're selling, and I didn't consent.'" Oh, and he says: "Meaningful, comprehensive federal privacy legislation should not only aim to put consumers in control of their data, it should also shine a light on actors trafficking in your data behind the scenes. Some state laws are looking to accomplish just that, but right now there is no federal standard protecting Americans from these practices. That's why we believe the Federal Trade Commission should establish a data-broker clearinghouse, requiring all data brokers to register, enabling consumers to track the transactions that have bundled and sold their data from place to place, and giving users the power to delete their data on demand - freely, easily and online - once and for all."

And he finishes: "As this debate kicks off, there will be plenty of proposals and competing interests for policymakers to consider. We cannot lose sight of the most important constituency, individuals trying to win back their right to privacy. Technology has the potential to keep changing the world for the better, but it will never achieve that potential without the full faith and confidence of the people who use it."

So anyway, I just think yes, you know, he's right. We know that this shadow economy exists. And once again I find, as I read this, I find myself feeling as though we're still in the very early days of this explosion in processing power; the collapse in the cost of mass storage, which has enabled endless compilation of these profiles, and every scrap and tidbit of data can be sucked in and retained, which you couldn't do if it was prohibitively expensive to do that; and of course the connectivity created by the Internet.

And not surprisingly, the regulatory framework that's needed to govern the implications of these changes lags far behind. And any of us who have, and we often have, listened to our policymakers talk or listened to congressional testimony and hearings and the questions that they ask, demonstrate that those who would create the regulations barely have any idea how this stuff works. And I don't have any idea how powerful the lobbying clout is of these data brokers, but it might be significant. And unfortunately, as we know, money drives a lot of this country's politics. So was there any sort of a conclusion from the discussion that you guys had, Leo, in MacBreak?

Leo: Well, "conclusion" like is this a good idea? I think everybody thinks it's a good idea; right?

Steve: Is there any chance it could happen?

Leo: Not a chance in hell. But we didn't actually talk about that. The fear, of course, is - so I think you're a trifle, I don't want to say naive.

Steve: That's okay.

Leo: But you're a nice guy in thinking that Tim Cook's just talking about data brokers. Because really, he may not say Google and Facebook, but any time Apple talks about privacy, there is always the subtext of "We do it right." Always.

Steve: Yes. I do agree that this brilliant marketing. I mean, this is brilliant...

Leo: Okay, so you're not naive. You understand.

Steve: Oh, no, no, I can...

Leo: They may not say it explicitly, but it's always about us versus Google and Facebook. That's, you know, yeah, it's about data brokers. And one thing we did point out is that, if Apple were really serious about this, they would, for instance, not require everybody to use Google as your search in Safari on the iOS device. But the reason they do it is because Google gives them $9 billion a year - actually this year it'll be, according to some estimates, $12 billion - to be the search tool. And if you want to protect privacy, you don't give it to Google.

Steve: So I guess that means, what, you use a different browser and use DuckDuckGo.

Leo: Yeah. DuckDuckGo has a browser. Apple should, if Apple really cared about this, it seems to me, they would at least give you the option to use DuckDuckGo in Safari. Right?

Steve: Yup.

Leo: They don't because it's billions of dollars in the pocket. So it's a little, you know, that's a point to be made also. As soon as you use an iPhone, the minute you put Facebook on it, forget it. Doesn't matter how secure Apple is. You've got Facebook on there. You're being spied upon immediately. So it's reasonable for Tim to say we want these regulations because users are going to put Facebook on our beautiful, pristine, private phones, and then it's just as bad as any other phone.

Steve: Right.

Leo: Finally, the other thing that we talked about is that one of Apple's big fears, every company's big fears, not that there'll be federal regulation. Everybody, by the way, including Google and Facebook, are calling for federal regulation. But the reason they do that is because the states will individually, as California has, impose their own privacy rules. And then you have a crazy quilt of 50 different rules. I mean, you already have to do GDPR. That's the other thing is that every company that does business in Europe is already doing most of those four points because GDPR...

Steve: And you probably saw that Google got hit with a massive GDPR fine.

Leo: Yeah, 50 million euros, yeah. By the way, for kind of something dumb.

Steve: Yeah, I know. It's like, okay. But you're right. We talked about the lack of federal oversight in the case of Net Neutrality, where again, having every state have their own legislation, it's just a nightmare for the carriers.

Leo: No, it isn't a nightmare. They know that they don't have to worry about it. And the feds have already said the states can't do it.

Steve: Yup.

Leo: Marco Rubio from Florida has already proposed in Congress a rule that says states can't make their own privacy regulations. Leave it to the big boys. Leave it to the grownups.

Steve: That's right. We'll take care of it.

Leo: And the presumption is, I mean, maybe, I mean, that is probably the right thing to do. But the presumption is that they will then, by getting all the power in the federal Congress, be able to write their own laws, these data brokers, and they won't have to worry about anybody. So, yeah, I don't know if it'll happen. There's definitely a current going in the country that people want this. So maybe it will.

Steve: Well, as long as the head of the data brokers association is not put in charge of drafting the legislation. Where have we seen that before?

Leo: That would never happen.

Steve: Let me think. Yeah, never happen.

Leo: I think ultimately we're going to have to protect our own privacy. And we're going to use DuckDuckGo and things like that; right?

Steve: Exactly. And that's what this podcast, one of the things this podcast is about.

Leo: You bet. Why we're here.

Steve: Speaking of which, this brings us to a piece that I only saw being covered, or actually I guess it's really not as much news as sort of a public service announcement from Sophos that is a sponsor of - it was a sponsor that appeared last week on this podcast.

Leo: Yes, our new sponsor.

Steve: Huh?

Leo: Yes, our new sponsor, yeah.

Steve: Yeah, new sponsor, Sophos. I've got a link in the show notes. And they are the source of this bit.ly link I talked about, bit.ly/snfbads, which expands to a link, also in the show notes, for a page that I don't know, going in normal like in the front door, how easy it is for a Facebook user to find this. But this shows your so-called "ad preferences" that you may have never known that you had.

I'm not a Facebook user. I do have an account because I needed once upon a time to look at the privacy and security settings that Facebook was offering. So I logged into that dusty old account in order to bring up two pages that I have here in the show notes. So if you do bit.ly/snfbads, you can learn what Facebook is doing for you. And so there are six main topics: your interests, that is, what Facebook has somehow independently determined your interests are; advertisers; your information; your ad settings, which you may not have known, I didn't know that you had; hide ad topics, your ability to hide ad topics, like parents could hide alcohol ads from minors, for example; and then an explanation of how Facebook ads work.

And when I saw ad settings, I clicked on that. And so it's got three sub-settings. And so, for example, the first one, which is default allowed, it says "Ads based on data from partners," which is exactly what we've been talking about, that says: "To show you better ads, we use data that advertisers and other partners provide to us about your activity off of Facebook Company Products." And that's allowed, but you can click it and say no.

Then the second one: "Ads based on activity on Facebook Company Products that you see elsewhere." And that is explained: "When we show you ads off Facebook Company Products, such as on websites, apps, and devices that use our advertising services, we use data about your activity on Facebook Company Products to make them more relevant." And that of course is also default allowed, and you can say no.

And then the third is "Ads that include your social actions." And they explain: "We may include your social actions on ads, such as liking the page that's running the ad. Who can see this info?" And then I had this set to "No One." And I don't know why it's set to No One. Maybe because I don't have any links to something, or who knows what. But anyway, that's just one of the six things that I opened. I thought it would be of interest to any of our listeners who didn't know that this was there. This is some nice disclosure and some controls, but also for our listeners to forward this to their friends and family that might want to know.

In Sophos's coverage of this, they said: "Fitbit? Pollination? Jaguars? Snakes? Mason jars?" They said: "Okay, fine, Facebook. I'm not surprised that I've clicked on those things." In fact, on mine, under I think it was on advertisers, mine were cars and realtors. And I have no idea why because I'm driving a 2011 car, and I've owned my home since '84. And nothing, there's no movement on any of those two categories.

Leo: They've got the wrong guy on that one, yeah.

Steve: So it's like, okay. But there were like 12 different ads, and they were all realtors or car sales. So anyway, they said, Sophos says: "But when did I ever click on anything related to 'Star Trek: Voyager' or cattle?" And anyway, so the guy...

Leo: Cattle?

Steve: Cattle. The guy writing this on Sophos says: "My 'this feels weird' reaction makes me one of the 51% of Facebook users who report that they're not comfortable that the ad-driven company creates a list that assigns each of us categories based on our real-life interests. It's called 'Your ad preferences.'" And they say: "You can view yours here. If you drill down, you can see where Facebook gets its categorization ideas from, including the things we click on or like, what our relationship status is, who employs us, and far more."

So anyway, just a FYI of some useful information. I thought this was a nice page to know about. And I don't know, if you're just looking at your Facebook, if it's easy to get there from the front page. But it's your ad preferences on Facebook. And some interesting information is there.

Okay. One last piece before we take our last break and get into the final bits. Cisco's small business switches have a serious problem. And so anyone using small business switches, pay attention. This is sort of "gob smacking," as they would say in the U.K. If you, your organization, or anyone you know are using Cisco 200 or 250 Series Smart Switches, 300 or 350 Series Managed Switches, Cisco 350X, 500, or 500X Series Stackable Managed Switches, there's a real problem. And we've talked about this recently, like last year, a number of times. Cisco was apparently auditing their own source and kept finding backdoors that had been written into their source code.

Leo: What?

Steve: I don't know if they were by developers who left this behind or what. In this case, this did not sound like the cause of the problem. This sounds deliberate. But it's a little stunning. In Cisco's summary of this, they said: "A vulnerability" - that's not what it is. But they said: "A vulnerability" - I mean, I guess they want it to be - "in the Cisco Small Business Switches software could allow an unauthenticated" - that's also not what it is, but I'll explain in a second - "remote attacker to bypass the user authentication mechanism of an affected device." Almost none of that is true. But they go on.

"The vulnerability exists because, under specific circumstances" - which are the default circumstances, but anyway - "the affected software enables a privileged user account without notifying administrators of the system. An attacker could exploit this vulnerability by using this account to log into an affected device and execute commands with full admin rights." Sounds bad. "Cisco has not released software updates that address this vulnerability. This advisory will be updated with fixed software information once fixed software becomes available. There is a workaround to address this vulnerability."

Okay. So what's really going on? Unbelievably, as I said, it's another of those Cisco default built-in passwords. The vulnerability, which has been assigned a CVE last year, it's CVE-2018-15439, has a critical base CVSS severity rating of - ready? - 9.8. So, you know...

Leo: Is 10 the highest?

Steve: Yes. Yes.

Leo: Okay.

Steve: And if it were 9.8 on IMDB, it'd be the most famous and popular movie anyone had ever seen. So, baby, you know, it's up there. It exists, get this, because the default configuration of these devices, okay, these are highly popular, widely sold, inexpensive Cisco networking gear. The default configuration of these devices includes a privileged user account that is used for the initial login and cannot be removed from the system. An administrator may disable this account by configuring other user accounts with access privilege level set to 15. However, you don't need to do that in order to use this so it's often not done. If no user configured privilege level 15 accounts exist in the device configuration, the default privileged user account is enabled without notifying administrators of the system.

So there's no notification given that this default login exists. Cisco says: "Under these circumstances" - which is the default circumstances - "an attacker can use this account to log into" - so that's not bypassing authentication, it's logging in, it's using authentication - "log into an affected device and execute commands with full admin rights. It could allow an unauthenticated" - okay, well, no, an authenticated - "remote attacker to bypass the user authentication mechanism" - no, to use the user authentication mechanism - "of an affected device."

So anyway, the workaround is - there is no patch. There is no update. There's nothing, there's no firmware available yet. And unfortunately, as we know, even once it's created, I mean, these are widely used, low end, Cisco networking gear that someone finally woke up to having, probably because they realized people were getting in somehow, and someone looked to figure out how, and it's like, oops. Unless someone adds an account with privilege level 15, this default account is enabled. So if any of our listeners know of or are responsible for or knows anybody who is responsible for any of these low end 200, 250, 300, 350, 500, and 550 series switches, you absolutely want to create your own account with privilege level 15 in order to disable the default account. There is no other way to do it on these switches.

So, wow, just unbelievable oversight on, I mean, and it feels like the fact that creating one disables it, I mean, maybe there's a disable which is broken? But it feels like this was on purpose, like by design, and they're finally getting around to saying, oh, maybe we shouldn't have done that. So I hope our listeners will protect themselves.

So a couple little bits of miscellany. I wanted to let our listeners know, I was looking around for a file sync solution. I'm now routinely working from two different locations, and since I hadn't traditionally been doing that, I was initially shuttling, like emailing my development directory to myself in each direction. And the other day I sent the wrong directory, so when I got to my other end it was like, oh, I don't have all my latest work. So I decided, that's crazy. I need to be able to keep directories in sync between different machines.

And so I did a little bit of looking around, and I think I've settled on a cross-platform solution that fits me because, I mean, Google kind of offers this kind of facility. Dropbox does. In my case, I have a server that could be a central location for synchronization that both of the endpoints are able to access. Anyway, Syncthing looks like a nice solution for this. So I just sort of wanted to put it on our listeners' radar. I have not started to play with it yet.

Leo: I've used it for a long time. It's fantastic.

Steve: Ah. Then thank you.

Leo: Highly recommended.

Steve: Yay, yay, yay, okay.

Leo: And it's free.

Steve: Yes, yes.

Leo: It's open source. It's great.

Steve: It felt right. They have a donation link at the bottom. I'm going to make sure that I use it.

Leo: It's kind of like the old BitTorrent Sync that we liked so much.

Steve: Yes.

Leo: Similar idea. Each place has a code; each source has a code. You could share that code with people. And I really like it, yeah, yeah.

Steve: Good. Syncthing. So as a result of my digging around, that's where I settled on. So I just wanted to say that I haven't started to use it yet, but I will provide a report to our listeners once I have. And I'm glad for the early heads-up.

Leo: Yeah.

Steve: I'm giving a presentation on SQRL this coming Sunday to a really neat ethical hackers group. And normally they would welcome more of our listeners who are in the area, so I would talk about it. But many are already listeners, and the interest expressed so far by those wishing to attend is already at 180% of the venue's capacity. So I'm going to wait until next week to talk about it any more, although it will be recorded. We'll have a video made from it. So I'll be able to share that link once it's ready. And apparently, based on the postings to the meeting signup board, I'll be signing many copies of SpinRite while I'm there. I guess these are old-timers, and maybe they have got disks of SpinRite because certainly we used to produce those in the old days. Or maybe they'll have me sign their, I don't know, their thumb drive, where they've downloaded SpinRite.

Leo: I like it.

Steve: But a whole bunch of people said, oh, I can finally get my copy of SpinRite signed, if Steve's going to be there. It's like, yeah. And so in order to prepare, I've been updating the presentation slides. In fact, that's one of the things that I mis-sent myself the other day. I've been updating the presentation slides that I first created for the

DigiCert Security Summit when I did the presentation of SQRL then. Then I updated them when I demonstrated SQRL to Stina and her technical colleagues at Yubico. The latest slides now add the news of this SQRL service provider API that I mentioned last week and I've talked about a little bit. But I am increasingly excited about it as I understand how much it will mean for SQRL.

I had never really thought about the server-side implementation until I considered what someone who didn't know SQRL at all would need to know. I mean, it's true, as I've mentioned before, that there's only one cryptographic function that the server needs, and that's to verify a signature. But there are a lot of other subtleties associated with rekeying a SQRL identity, allowing a user to disable their use of SQRL, and a bunch of other stuff. And while it's not rocket science, it's completely unfamiliar to anybody who doesn't know the SQRL protocol.

And so asking a whole bunch of different, I mean, it was foreseeable that there would be server-side implementations created. I have just ended up doing it, and in the process defining a very simple API that abstracts all of that stuff on the other side of the API, making it very simple for someone to add SQRL to an existing web server. And in fact this developer I've mentioned before in Denmark, who knows the XenForo forum software we're using, was able to quickly and easily add SQRL support using the API, and now he's working on finishing up the configuration dialogs and adding some touches to the UI.

And one of the guys in the GRC newsgroups who already knows the SQRL protocol inside and out has received from me my assembly language source code for the API, and he's reimplementing it in portable open source C. So that'll give us a multiplatform, compile it on any platform you want, encapsulated SQRL API, making it probably a day or two worth of work on any programmer's part to add SQRL to an existing web server. And I'm excited about that because, as we know, it's one thing to like all the things that SQRL does, but for it to actually go anywhere it needs to get adopted.

And so this will - I'm really, I mean, I didn't even foresee the need for this service provider API in the beginning until it came time for me to have a website that wasn't natively supporting SQRL, meaning these SQRL forums, support SQRL. And then it was like, oh, okay. How should I do that? And we ended up with another really - what's going to end up being a super useful component of this SQRL ecosystem. So I'm jazzed about that. And I'll be talking about that and more at this conference on Sunday, or this hacker meeting on Sunday, and I'll talk about it a little bit how it goes next Tuesday.

Okay. So errata. As I said, I could not believe what I said last week. This was relating to the expiration of certificates on web servers. When I talked about pushing past the "certificate is expired" notice, I said that what this meant was that all of the communications would then be in the clear. Which is of course not the case. It means you say to your browser, yes, I understand the certificate is expired, but let's use it anyway. And in which case...

Leo: It's still encrypted, but it's just expired, yeah.

Steve: Yes. And the problem is I was, I mean, I was wrong. So okay, let's fix that. So yes, it's still encrypted. It does mean that you have blown your security guarantee, that you are subject to man-in-the-middle attacks. A man in the middle could cause that warning to occur, and you say, okay, I don't care, blah blah blah. Okay, but still I was - there was some of the coverage of this was talking about logon credentials being now sent in the clear, and I did not drill down far enough. I just repeated it without thinking.

So my apologies for, I mean, what I know is not the case. We've talked about this so many times before. And a bunch of people wrote to me saying, uh, what?

Leo: Oh, come on. That's just a [indiscernible].

Steve: Anyway, you know me, I needed to fix that.

Leo: Get it right, okay, that's fair.

Steve: Yes. And so it's right. So it is the case that you are bypassing security, but it is not the case that someone doing passive sniffing, as I said, is then going to be able to see your username and password in the clear.

Gary Foard in England said - the subject line was "30 gig?" And he said: "Dear Steve." This is also errata. He said: "Love the show and SpinRite. But last week, 697, you mentioned again, as before in 696, about Filemail.com and that you have a 30-gig upstream connection. Really?" He says: "Bloody hell!" He says "in English accent." He says: "I get along with 6-megabits downstream and 1-megabit upstream. Some people have 100-megabits downstream, which is good. I think I've heard Leo brag of his business connection being 10-gig, but you say 30-gig upstream?"

He says: "I don't know how much SpinRite you sell, but it ain't that big that you need 30-gig up." He says: "You're a nice guy and deserve it, but maybe you mean 30-megabits. That would still be impressive. You've said it twice now, and it's bugging me. Have a nice day." And yes, I will now be very self-conscious about 30-megabits, not gigabits. I do not need, I do not want, I can't imagine 30-gig upstream. So yes, I have always been meaning to say 30-megabits. If I make a mistake again, just correct it, everybody, in your head: meg, not gig. So Gary, thank you.

Leo: I just assumed you did have 30 gigs.

Steve: No, no.

Leo: I know. I just thought, well, he goes through Level 3. Maybe they gave him a big fat glass pipe.

Steve: Yeah. No, this is at home. This is like, you know, Filemail when I was - it was 30-megabits, or actually it was 33-megabits that it was saturated. And I could see how I could have said gigabits because it's easy to make that mistake. So yes, 33-megabits up, and 300-megabits down. But rarely do I see that. It's only, you know, sometimes I'm downloading something from someone big with a really good CDN or something, and it's like, whoa, I actually do get 300. But mostly they're feeding it slower than that at the feeding end, and so I don't really ever see that.

Leo: Even at 100 you're faster than most people are sending.

Steve: Yeah, exactly. So Ben asks about having a dedicated SpinRite machine. Wat? W-A-T? And he said: "Hello, Steve. Long-time listener of Security Now! and all Gibson goodness wherever it's available. Crawler of GRC.com. Owner of SpinRite and recommend it to everyone on the train. Thank you for all of those."

He says: "I came across an interesting use case and wondered how to best do it. A friend of mine works at a 'photo shop.'" He has that in quotes. He says: "That's what they're called here. They do a variety of photo services, as well as a few other digital services. One of them is restoration - restoring deleted content and/or corrupted media. I visited him at his workplace and saw that they're using Recuva, R-E-C-U-V-A."

Leo: Recuva, yeah.

Steve: Recuva. "Knowing what the answers will be, I asked him what the success rate is. Expectedly, it's so-so. They use several other such utilities with about the same results. I recommended they buy SpinRite. My friend trusts me, and he convinced the boss to buy it. Success rates are, unsurprisingly, far better now. This got me thinking. They could use a dedicated machine for SpinRite. Since SpinRite consumes the whole machine, and they do this often enough, then a machine dedicated for SpinRite makes sense. So what hardware would this machine feature? From my own experience with SpinRite, I know that it can become very CPU intensive, depending on what it's dealing with on the target media. So, CPU. Lots of power? Cores versus clock speed? What else? Machine will of course be diskless for itself. Is there anything to avoid?"

He says: "While in normal circumstances the machine's hardware wouldn't be a big focus - you start SpinRite, let it run - here it's different. The machine cannot do anything else, and SpinRite does one thing at a time. So giving it the resources to complete its work quicker is the whole point. I recall that you did say that one can try running SpinRite in a VM and connecting the media to the VM. But you also said that narrows down SpinRite's range of functionality. So what's optimal for SpinRite? If SpinRite can consume the universe, what would it consume? If one had no other considerations, what would that machine look like? I'd love it if you can bring this up in an SN episode. Would also like to thank Leo and the heroic team at TWiT, as well. Special thanks to Elaine. Her transcripts come in very handy."

So Ben, and to all of our listeners, first of all, a dedicated SpinRite machine is a very popular thing. I mean, I've been hearing about them for 30 years. Many people do it. The good news is you actually can use an old piece of junk hardware that is barely useful any longer for anything else. It turns out that it's actually not at all CPU intensive. It is throughput intensive. So it is intensive of the connection between the drive and the computer. Which is why, if at all possible, for example, you want to mount the drive directly to the interface on the motherboard; not, for example, run it through a USB connection which can be a bottleneck.

So connecting the drive to the motherboard is really the only thing you need to do. At that point, anything that has the proper connection, old-school IDE interface or a SATA interface which is typical now, that'll do it. But, I mean, really it can be a machine that you've stopped using because it wasn't fast enough to even run a GUI or any kind of a desktop OS. SpinRite needs no RAM, like literally 640K it runs in. 6.1 will start using more RAM, but even then less than a gig. I think it was - I have it in my mind, yes, it was a 30MB buffer because I'm allocating a 64K sector buffer, and a sector is half a K. So it's a 32MB buffer that the next version will use in order to suck in 32MB at a crack, which is where SpinRite 6.1 will get its huge speed boost.

But until then, any hardware you've got you can use. And so what it means is also, if you had a couple of older machines, you could have several running on different drives at the same time. But anyway, SpinRite is very, very noncritical about the hardware it uses because it's just running DOS on the chip itself, and so it doesn't actually need much processing power.

Steve Fosmire in Denton, Texas, asked about Java Update offering to remove itself. And so I'll share his note. He said: "I just got a Java update notification that popped up, so I clicked on Update." And at that point, you know, as all of our listeners used to refer to it, I haven't heard the term for a while, but we called this a "Gibsonian response." It was like, whoa, stop everything. A Java update notification popped up. At that point, you know, if you've been listening to the Security Now! podcast for long, you're thinking, whoa, what? Wait a minute.

So anyway, he goes on: "Please remove unused versions of Java." And he says the update read: "It appears that you have not used Java on your system in over six months. We recommend that you uninstall it by clicking the Remove button below. If you later decide you need Java, you can reinstall it from Java.com." Okay, now, that kind of sounds better. He says: "If you wish to keep Java on your system, please update it by clicking the Update button below." And he says: "How about that? I don't remember what thing I needed Java for, but to have it tell me 'go ahead and remove it' is a new thing entirely. Thought you would want to know. I did take a screen shot of the Java Update window, so if you decide you want to put this on the SN show, I could send it to you. Thanks for over a decade's worth of free security knowledge goodness. Steve."

He says: "P.S.: Proud SpinRite site license" - he says in parens "(4)" meaning four copies - "owner from my previous IT consultant company." He says: "Closed up shop a year ago and went to work full-time as a network admin at my biggest client." So this is interesting. So this was not - so I don't know what triggered the update. That would be interesting. But if this is legitimate, if Java noted that it had not been used and suggested it remove itself or you remove it, that's very cool. I agree, Steve. So I think maybe I misunderstood initially what you said.

But again, I know that our listeners know that anything that pops up and asks you to take action, depending upon where you are, what you're doing, are you on a website, or like what's going on, just be very skeptical. So I wanted to share that that had apparently been possible to have happen.

Okay. Topic of our show. Which mobile VPN client? As I mentioned at the top, Bleeping Computer provided some coverage from a Simon Migliano, who is the head of research at Metric Labs. In Bleeping Computer's coverage, they said: "One in five apps" - okay, one in five - "from the top 150 free VPN Android apps in Google's Play Store was flagged as a potential source of malware, while a quarter of them" - so one in four - "come with user privacy breaking bugs such as DNS leaks which expose users' DNS queries to their ISPs."

They wrote: "As found by Simon Migliano, Metric Labs' Head of Research, the company behind the Top10VPN service, these VPN Android applications have already been installed approximately 260 million times" - in 260 million phones - "according to the numbers reported by Google's official store." Okay. So we've got several problems here. First of all, the fact that there are 150 free VPN Android apps. That suggests this is a popular thing, that running a VPN in your Android device is something people want to do.

In the show notes I took a snapshot from Bleeping Computer's page where it shows an app by installation. And so, for example, Hotspot Shield Free version, more than 50 million downloads. Risky permissions were detected, no DNS leaks, risky functions were detected, no virus or malware. SuperVPN also had 50 million downloads, also risky permissions detected, also risky functions detected, and DNS leaks in this one. Hi VPN -

hi, VPN - 10 million downloads, risky permissions detected, risky functions detected, DNS leaks detected. And so on and so on, going down the list.

Lots of risky permissions detected. Lots of risk functions detected. Most of them, all but, let's see, one, two, three, four, five, six, seven, eight, nine, ten. So this is the top 10 VPN clients. And of those, one, two, three, four did not leak DNS. The other six did. Eight of the 10 had risky functions detected, and a different eight of the 10 had risky permissions detected. Bleeping Computer reported that the research team found the following intrusive permissions and user privacy breaking code.

So when I talked about risky functions or risky permissions, get this. Location tracking, a quarter of the VPN apps are doing location tracking. Access to device status information in 38% of them. In a fewer number, use of camera and microphone and the ability to secretly send SMS messages. In a VPN client, which makes my head explode. Over half, 57% featured code to get a user's last known location.

So this brings us to the main topic and title of today's podcast: Which Mobile VPN Client? There is only one anyone ever needs. It's called, not surprisingly, OpenVPN for Android. I have a link in the show notes. OpenVPN. It's maintained, compiled from OpenVPN source by Arne Schwabe, who lives in Germany. And it's like, maybe it's not flashy. I don't get why it's only five million downloads, whereas the other ones are 50 million.

Leo: Well, Steve, because it doesn't offer an OpenVPN server. It's just a client.

Steve: Yes, and that's all anyone needs.

Leo: Okay, but the other ones you were talking about were clients for commercial services in many cases. So like Hotspot is a commercial VPN provider. So be clear when you download this, folks, you still need an OpenVPN server.

Steve: Right.

Leo: This is just a client.

Steve: Right, right, right. And I was going to say that. I didn't understand that those other ones are - so they're providing you...

Leo: Can't say for all of them. But Hotspot for sure, that's a commercial VPN service.

Steve: Right, right, right.

Leo: So I'm not sure what Bleeping Computer is trying to, yeah, I'm not sure what they're - I think they're confusing the category, as well. I'm not sure exactly what they're saying. There are a lot of bad VPN companies, that's for sure.

Steve: There are bad VPN companies. And what we now know is there are also bad VPN clients. That is, nobody who wants to use a VPN would want to have something that was tracking their location, using their camera and microphone and so forth.

Leo: Yeah. I mean, but there's nothing wrong with using OpenVPN. I mean, this is OpenVPN; right? Use the client.

Steve: Exactly.

Leo: But you need to have a server at the other end. Otherwise...

Steve: Yes, yes. And so I would, I guess, I would separate the two. I would use no other client than this OpenVPN client for Android. It is on GitHub. It's produced by a reputable guy who's doing this in open source. I mean, that's all this thing is, is a clean, simple OpenVPN client for Android. And that was where I was headed with this is then go choose whatever OpenVPN service you feel comfortable with. And there are, like, many. In fact, all of these VPNs are probably flavors of OpenVPN because this problem, the whole VPNing problem, has been resolved. I mean, it's been solved already.

So anyway, so that was my takeaway was the idea that there were this incredible number of clients where you've got to ask yourself, and as you and I have often talked about, Leo, why are they providing this to you for free? I wouldn't install anything on an Android device other than this one OpenVPN client, and then choose whatever service you like.

Leo: Most good VPN services have their own client. They have their own tied app. It's a dedicated app.

Steve: Yeah, although you are able to configure whatever you want to.

Leo: I guess so, yeah. I mean, but if you're using NordVPN or our sponsor, ExpressVPN, you're using the Nord client, or the ExpressVPN client, which means you don't have to do any configuration. It just connects to their servers.

Steve: Right.

Leo: So if you know what you're doing, OpenVPN for Android makes sense. But you have to know what the server is. You have to know how to configure it, et cetera, et cetera.

Steve: Right, right. So anyway, that was my message. Just use OpenVPN. This problem has been solved. You really don't need to look any further. And you can have a clean client in your device.

Leo: Is DNS leakage something that the client does? Or is it something that the service does, the server does?

Steve: Yes, it's something that the client does, if it's not designed properly. And even OpenVPN, you can control with a config file whether DNS is routed through the tunnel or not routed through the tunnel. So it does need to be set up properly so that all of the network access routes through the tunnel, and DNS isn't being provided by the OS's underlying platform service. So it is worth verifying that, I mean, if you're concerned about that from a privacy leakage standpoint, that you've got a client that's doing it; it's configured the way you want it to be.

Leo: Right. I know many companies don't like OpenVPN because of previous bugs; right? There are some companies that use other VPN technologies like IPSec; right?

Steve: There are. The problem with IPSec is that it can be difficult to get out of certain locations. There are a lot of firewalls that block IPSec; whereas OpenVPN is able to run over standard ports because it just uses TLS and [crosstalk]. Yes, exactly.

Leo: Port 443, yeah.

Steve: Yeah.

Leo: Okay, cool. So just to be clear, that's a great client for OpenVPN, but you have to have a server somewhere that you're connected to.

Steve: Correct.

Leo: Many people do. I mean, lots of routers support it, for instance.

Steve: That's what I've got. Every one of my locations has an OpenVPN server.

Leo: They're already running it, yeah.

Steve: And pfSense, which is my preferred firewall, it's got OpenVPN server. You just click a button, and then it's turned on. So it's great.

Leo: That's very handy. Very good, Steverino. We have concluded our proceedings for the day, for the week. Episode 698 in the can, as they say. You could find copies of Security Now! past and present at Steve's site, GRC.com, the Gibson Research Corporation. He also has, as he mentioned, Elaine Farris's great transcripts. And as long as you're at GRC.com, make sure you get a copy of SpinRite, the world's best hard drive recovery and maintenance utility, plus all the other freebies Steve gives away. There's lots of good content.

Steve: You can run it on an old computer, on one of your little wind-up shoe leather computers. It'll just go just fine.

Leo: If it's got BIOS, you can run it.

Steve: That's right.

Leo: You also might want to check out SQRL, which is apparently .999999 ready.

Steve: It is just painfully, painfully close to happening. Yup.

Leo: Will you post your talk? Will there be a video? Will you post it?

Steve: Yes, yes, yes. There'll be a video. I'll post the talk. And before long Lorrie and I are going to come up and hang out with you and Lisa.

Leo: Oh, good.

Steve: And then we'll do the full presentation.

Leo: Good. We have to check this young lady out.

Steve: Yes.

Leo: If you want to get all of that, again, GRC.com. Steve's Twitter handle is @SGgrc. You can DM him there. And of course he tweets the show notes before every show, so you can get a copy of those there, @SGgrc. We have audio and video of the show, oddly enough, for no apparent reason, on our website, TWiT.tv/sn. You can also subscribe. You just search for Security Now!, and you can get a copy of it automatically downloaded every Tuesday, the minute it's available.

We do the show 1:30 p.m. Pacific, 4:30 Eastern, 21:30 UTC on Tuesdays. If you want to watch us do it live, TWiT.tv/live has audio and video feeds. And if you do that, hang out in the chatroom. It's where everybody else watching live is, in irc.twit.tv. Steve, we'll see you next week on Security Now!.

Steve: My friend, till then, bye.