Legacy's Long Tail

Description: This week we discuss an embarrassing high-profile breach of an online identity company, an overhyped problem found in Linux's sudo command, the frightening software used by the U.K.'s Trident nuclear missile submarine launch platforms, how emerging nations prevent high school test cheating, another lesson about the danger of SMS authentication codes, another worrisome Shodan search result, high-penetration dangerous adware from a Chinese marketer, another "that's not a bug" bug in Chrome allowing websites to surreptitiously record audio and video without the user's knowledge, the foreseeable evolution of hybrid cryptomalware, the limp return of Google Contributor, Google continues to work on end-to-end email encryption, a follow-up on straight-to-voicemail policy, "homomorphic encryption" (what the heck is that?), and "closing the loop" follow-up from recent discussions.

High quality (64 kbps) mp3 audio file URL: http://media.GRC.com/sn/SN-615.mp3
Quarter size (16 kbps) mp3 audio file URL: http://media.GRC.com/sn/sn-615-lq.mp3

SHOW TEASE: It's time for Security Now! with Steve Gibson. And Steve is guarding the galaxy of security exploits. OneLogin becomes no login. Linux has a new privilege escalation hack. Windows for Warships? And your cell phone carrier will leave you baked, broke, and potentially busted. Security Now! is next.

FATHER ROBERT BALLECER: This is Security Now! with Steve Gibson, Episode 615, recorded June 6, 2017: Legacy's Long Tail.

This is Security Now!. It's a dangerous universe, and we're like the Guardians of the Galaxy except what we guard is far more important than a bunch of cosmic stones. That's right, Steve G., he's the man behind Gibson Research, ShieldsUP!, and SpinRite. He's the one person you can trust because he answers all your security questions with "I am Gibson."

I'm Father Robert Ballecer, the Digital Jesuit, in for Leo Laporte. And let's get to the action. Steve, it is so good to work with you again. I love, I mean, I like having Leo here, but I love when he goes on vacation because it means maybe, maybe I get to play with Gibson.

Steve Gibson: Well, it's fun to have a little variety. And I am not Groot. I am, as you said, Gibson. I did see the second "Guardians of the Galaxy." And it was cute, but I thought the first one was better.

PADRE: I haven't seen any movies this summer yet. Literally no movies.

Steve: Wow. Well, you've been very busy this summer.

PADRE: I've been a little busy. I do want to see "Guardians of the Galaxy." I want to see "Wonder Woman." I've heard great things about that. And of course I need to see "Spider-Man: Homecoming" because I just - that's my childhood right there. Spider Man was my geek passion.

Steve: It looks great. And I've enjoyed all the Spider-Man movies so far.

PADRE: Steve, this has been an interesting week. You know, nothing earth shattering, but quite a few things that have been building up on security breaches and bad practices that we've been covering over the last couple of years. Where do you want to start?

Steve: Well, so the last couple weeks, of course, have been WannaCry and like some crazy high-profile things. As you said, there was a lot of interesting news, but nothing particularly breathtaking. So I decided to title this "Legacy's Long Tail" because that's a theme we keep seeing. And there's one particular story that is - actually, I think the term they use in the U.K. is "gobsmacking," which is appropriate because this involves the U.K.'s four nuclear-tipped missile Trident submarines and a piece of news that, well, our listeners will know what it means when we get there.

But there was a bunch of stuff to talk about. We're going to talk about an embarrassingly high-profile breach of an online identity company that's got hundreds of corporate users who all then have themselves a huge number, in the tens of thousands, of users of their services. A somewhat overhyped problem found in Linux's sudo command. I mentioned the frightening software used by the U.K.'s Trident missile subs. Just a fun anecdote about how small emerging nations, or at least one in particular, although there was some news about a different nation, as well, preventing high school test cheating. Another lesson that we - and we've had a surprising number of problems with SMS authentication codes. We hit another example of that in the last week.

The founder of Shodan did a sort of a custom script to search for a particular type of presence on the Internet and was surprised by what he found. The news of a high-penetration, very worrisome adware from a Chinese marketer that reminded me of my own experience 17 years ago, believe it or not. Another "but that's not a bug" bug in Chrome, which allows websites to surreptitiously record audio and video without their users' knowledge. A foreseeable evolution of cryptomalware. The somewhat limp return of Google's Contributor. I hope they're going to fix it in the future.

And just a couple little bits of news. The fact that Google is continuing to work on end-to-end email encryption, which is a very elusive holy grail. Everybody wants it, but I've always been lukewarm on it because email is just fundamentally not secure, and all efforts so far to layer security on top of it have only come at the cost of significant inconvenience. So we'll sort of look at where that's going. I wanted to follow up on a mention I made last week about the FCC's consideration of formally authorizing a so-called "straight to voicemail" policy and how worrisome that is.

And then I'm going to tease us a little bit about something, we may have mentioned it a couple times, but we've never dug into it, and I don't think we're going to have time to do it justice that week. But that is homomorphic encryption. We'll just sort of touch on that before closing the loop with a comment from one of our listeners also relating to a topic from last week. So I think another great Security Now! with lots of interesting content for our listeners.

PADRE: I like this kind of an episode because it's stretched out. There are so many topics that have a little bit of the puzzle. I mean, in case those people who have been watching Security Now! for years and years and years haven't figured it out yet, everything that Steve talks about kind of goes into the next topic and the next topic and the next breach and the next attack. And I like the menu that you've set down.

Steve: Well, and we end up with - what our longtime listeners are seeing are sort of the emergence of some themes, where it's like interpreters are hard, and we keep seeing vulnerabilities in all kinds of different instances of interpretation, that sort of being like the next frontier. We finally moved from macros and email onto other things. And then of course another theme is that it's so difficult to get us to move forward, to abandon previous solutions in favor of better ones like how difficult it was to get away from SHA-1 signed certificates onto SHA-256, even though the better ones had been around for a long time. It's like, yeah, but, you know, these are working, and we're busy. We have other things to do. And so there is this, I think it's interesting, sort of this emergent set of themes that everything fits into. So, yeah, more of that this week.

PADRE: All right, Steve. So we've got a few things that we need to cover. Do we want to bury the lead, or do we go with the big story of the week?

Steve: You know, I don't want to forget to share with our listeners your own project and discovery. Let's start with that because I think it's so cool. One of the things that, in talking about this notion of recurring themes, one of the things that is huge has been side channel attacks. And we've talked about all kinds of different ones, like watching somebody entering keys on a keypad from a distance on their phone. Even though you can't see the surface of the phone, you can see the relative motion. And we realized that it's possible to disambiguate what they're doing from that. Just recently there was news of the accelerometer being able to determine where you were tapping on the screen because in tapping it, no matter how rigidly you held it, just the tap resulted in pitch and yaw that was a function of where you were tapping and so forth.

So there's been lots of talk recently about side channel attacks. And you actually, as you often do on Know How, demonstrated how one that we have discussed in the past sort of from a theoretical standpoint actually works.

PADRE: Yeah. This is kind of weird. I heard about this, and I kind of dismissed it. I thought, oh, this is one of those silly Internet things. But there was a discussion of whether or not you could use a sensitive enough IR camera - and not a near IR (NIR) camera. There's this thing called near IR which is basically just a CMOS or a CCD sensor that you remove the IR-cut filter, and maybe you put a few IR LEDs. I mean, that does work. When you're talking about home security cameras, you can use that reflected IR to be able to sort of see in the dark. But a real IR camera sees emitted IR, not reflected IR. And there's a bunch of really cool things you can do with that. We've been using it on TWiET, we've been using it on Know How to do things like see whether or not a power cable is overloaded because, as it approaches its maximum rated value...

Steve: Nice, nice.

PADRE: ...it shows up. It starts emitting a lot of heat. Same thing with power panels. Or what I've used it for is I'll go to the datacenter, and you can actually look at the flow of heat, the emitted light, IR, from that heat as it flows up from the racks. Shows you whether or not you've got a good placement of all your equipment because you want your hot stuff up top, and you want the cooler stuff down below.

Well, there was one other thing, and I wanted to try it with this. This actually was just

released a few hours ago. This is the new FLIR ONE Pro. FLIR, of course, is Forward Looking Infrared. They're a company that has sort of made their name making military hardware. Well, they came into the studio, and they talked about their new Lepton chip, which is one of these military sensors that they shrunk down, reduced the resolution a tiny bit, and they've given us this.

So this is a module, the FLIR One Pro module, that goes onto a USB-C equipped phone. I'll show a couple of features on it right now. But really quickly I want to show you what we're talking about, if you go to the side camera here, John. So this is a remote control. I'm using it as a stand-in for any PIN device. So if you have to punch in a personal identifying number, well, there's something that you can do. If you notice, right now this is seeing this remote, and it's all black. But if I punch in my PIN, and then I do it, you can actually see where the residual heat watches are.

Steve: Nice.

PADRE: And the cool thing is, if you take a picture of this, it lets you switch back and forth between visible and IR energy, and it will show you the heat decay. So by calculating the heat decay, you can figure out which button was pushed when. Now, I've tried this at supermarkets, where people push in their PIN numbers. And of course I always ask them if I could do it. They punch in their PIN numbers, and then I'll just take a quick scan of the pad. And I'm able to, like 90% of the time, duplicate their sequence. Again, it's one of these things where you're probably not thinking about it, but, yeah, you do leave fingerprints, even if you don't know it. In fact, I've gotten into the custom now of anytime I'm using a PIN pad, I will actually put my hand over the entire pad to warm it. I know that's silly, but now that I know that it's really easy to do this? And by the way, that heat decay can stick around for a couple of minutes. It's just one of the things I have to watch for now.

Steve: Very cool. Another classic side channel attack. In this case, our hot little fingers heat up the buttons, and the buttons are going to return to ambient temperature in the sequence in which they were pressed. So, wow, very cool.

PADRE: And actually I thought maybe this attack would work better in the winter because the heat would show up better. Not so. It actually works better in the summer because then the ambient air isn't there to pull the heat away from the buttons. In the winter, when I was doing this in the cold, I could maybe get an image 60 seconds after someone's punched it. In the summer, I just tested it when it was like an 87-degree day in San Francisco, I was able to do it five minutes afterwards. So, I mean, it's pretty impressive. And the sensor itself, this is not a CCD. This is not a CMOS. This is an actual custom piece of silicon that they created. And I've got to say, it's a specialized product, but for $200 this is something that's going to be fun to have in my toolkit.

Steve: I was just going to ask you what the price is. So 200 bucks, that's a good price for a FLIR camera.

PADRE: Well, I remember the first time I ever played with a FLIR was on an Apache gunship. They were doing one of these demonstrations. And they had us put on the helmet.

Steve: Well, of course.

PADRE: And the camera and the turret follows your head tracking. And they would turn off all the lights, and you could still see, just zeroed in on heat signatures. This is the

same technology.

Steve: Very cool.

PADRE: So I guess what you should do is don't touch anything ever.

Steve: Or, I guess, what, wear gloves. You want to not transmit the heat into the device that you're touching.

PADRE: Someone in the chatroom says that he carries around a bottle of Freon anywhere he goes, and he just sprays it down afterwards.

Steve: Or enter your PIN, get the acknowledgment, and then press a few more buttons.

PADRE: Just keep going.

Steve: Yes, exactly.

PADRE: Again, that's one of those other things I will confess to. Whenever I'm typing in a PIN, I will do some fake PIN presses, and then the real PIN presses, and then more fake PIN presses.

Steve: And you might call that "PIN Haystacks," when you think about it.

PADRE: Yes, actually, yes. And I only do that because just this last week - I don't know if you do this, too, Steve. But anytime I go to an ATM, I always give a yank on the card swipe, just to make sure no one put something over the top of it.

Steve: Yup.

PADRE: And, finally, seven years after starting to do this, I yanked, and one came off. So I've got a, what do they call them, swiping device.

Steve: Oh, my goodness. Wow.

PADRE: I just have to figure out how it works.

Steve: So our first bit of security news is, aside from this cool side channel attack, is that a major single sign-on provider, OneLogin [audio dropout]. It was last Wednesday evening around 9:00 p.m. Pacific time, they detected an odd usage pattern in a database instance. They use Amazon's AWS cloud services for their infrastructure, and they saw that something suspicious was going on and shut down that instance. Subsequently, looking at it, they realized their worst nightmare had occurred, and that is that for about seven hours somebody had been rummaging around in their database.

So starting at around 2:00 p.m. last Wednesday afternoon through 9:00 p.m. when they saw it and shut it down, somebody had apparently obtained their AWS keys, which meant they had decrypted access to their infrastructure. So just to remind people, what a service like OneLogin does is they're sort of a commercialized version of "Sign in with Google." That is, they are an OAuth provider.

So the idea would be that, rather than Sign in with Google or Log in with Google, it would be Sign in with OneLogin. So their customers are corporations who want to offer to their

employees or to their customers the convenience of single sign-on operation. So when you go to the company's site, rather than "Sign in with Google," you see "Sign in with OneLogin." So essentially they're an aggregator of the login technology, the login verification for all of their customers. And they're logging their customers' customers, that is, their visitors [audio dropout], depending upon what their model is, they're logging them into those companies. So there's, I mean, they have a huge responsibility, and a breach of their security is a disaster, not only for their direct customers, but for all the people who log into their customers' sites. So this is not good.

Now, they got a lot of coverage in the press because this was a high-profile breach. And the coverage was not all good. Aside from the embarrassment, which is completely understandable, and one of the recurring themes on the podcast is security is hard. And everyone can be forgiven, I think, for making a mistake. How people respond is what's important. So we've defended LastPass, that has been a recent victim of Tavis Ormandy's showers, because even if problems are found, in no instance were any customers damaged that we know of, and LastPass responds typically within hours. So the problem here is - and responsible disclosure means you just tell people what happened. So OneLogin didn't do that. They were very circuitous in their disclosure. They didn't publicly provide much detail. Their response has been called "opaque" and "deliberately lacking in detail."

However, they provided their customers, to whom they feel, I guess, more obligation, email with much more information. Of course that email didn't stay in the possession of the customers. Quickly that got out to the press, that realized the extent of what had happened. And in this email that they provided to their own customers, they said: "The threat actor," as they put it, "was able to access database tables that contain information about users, apps, and various types of keys." They wrote: "While we encrypt certain sensitive data at rest, at this time we cannot rule out the possibility that the threat actor also obtained the ability to decrypt data. We are thus erring on the side of caution and recommending actions our customers should take, which we have already communicated."

And then, in this private email to their customers, they were told basically start over. That is, start from scratch. Essentially, nothing that had been done could be trusted. Meaning they were told to generate new API keys and OAuth tokens, create new security certificates as well as credentials, recycle any secrets that were stored in OneLogin's Secure Notes feature, and have every end user update their passwords and more. So basically a complete start from scratch. They were saying, as far as we know, the bad guys obtained our AWS keys and had access to the encrypted secrets that we, say they, were holding on behalf of all of our customers. So, sorry about that. Start over.

PADRE: You know, Steve, one of the parts of the story that just kind of - it flusters me. And again, it's because they've been so opaque with the information that they've released. At first they said, okay, we've been breached, but we don't know how bad it is. Then it was, it's probably all the customers in the United States, but we don't know what they've taken. And then that memo leaked, and they realized, no, they took everything. They had access to the complete database. Now, check me on this. So the attacker, whoever it was, got a copy of the API keys. So they were able to directly access the storage basket that OneLogin was using for their database.

Steve: Right.

PADRE: They don't tell us exactly how they got access to the second part, which is either information or a tool to decrypt the database, because it was encrypted. This was not left out in cleartext.

Steve: Right.

PADRE: But OneLogin in essentially saying, yeah, we think that they've got a method for decrypting, which has driven a lot of speculation. Was it a common encryption theme? Was it something that was tied to the AWS key? Were they just using Amazon encryption? Or was this a proprietary encryption, that they left information in cleartext on the same server?

Steve: Well, again, there's no information because this is their proprietary technology. What we do know is that, essentially, if they're an OAuth provider, they have the ability to essentially authenticate the people who authenticate to them on behalf of their customers. So what this says is, again, basically start over. They have to start over.

Now, as everybody who's listening to this podcast knows, I've invested a great deal of my time with the SQRL project in nailing down every last detail required to turn what is a simple and theoretically strong concept, that is, the original idea that I had, into a useful reality with SQRL. So when I encounter stories like this, since SQRL is on my mind, and it's what I've been spending all my time on to push this thing over the finish line, I naturally test what I have, the model, against the same kinds of vulnerabilities. And what's so fundamentally different, and what makes SQRL unique, is that it operates in an entirely different way. The slogan I've coined to describe this is "SQRL gives websites no secrets to keep." And so naturally, if a service has no secrets to keep, it has no secrets to lose. It has no secrets at risk.

So in traditional login, and OneLogin and OAuth are still in the traditional model, they rely upon first entrusting a server with a secret. Or hopefully a safer derivative of a secret, we all know, which is frequently a hash. And then in subsequent visits the original provider of that secret chose the server again that we still have or remember that original secret. So we're just saying, "I'm back. Here's essentially what I gave you or a derivative of what I gave you. It's me." And so, again, the problem is that that remote service has to keep those secret because those can be used to impersonate someone.

What's different about SQRL and always has been is that it's based on a simple cryptographic proof of knowledge, so without ever sharing that knowledge. What this means is that the site, what we give to a site, what the site holds for the user can only ever be used to verify the user's assertion of who they are. It cannot be used to generate such an assertion. So there is zero value in the theft of that information from the site. So it's fundamentally a different approach. And we're getting close to getting this thing out the door. As our listeners know, it's been working more or less for a year.

And I'm just, again, I'm continually testing against the problems that I see in order to make sure that it meets the challenge. And in fact there's already some features that are there I'll touch on a little bit later with a different story, again where I'm saying, okay, yeah, we got that covered, too. At the moment I've just been rejiggering some of SQRL's antispoofing enforcement because spoofing is a problem that there still has not yet been any solution for. And OAuth is as subject to it as anything.

And I just decided, and I talked about this last week, and I think the week before, that as a consequence of what Microsoft was saying they were going to do in Windows 10, which was to ban Edge web pages from being able to access the localhost services, we backed off a year and a half ago, about 18 months ago, summer before last, when that was apparently going to happen. That ended up not happening because Microsoft, it turns out, could not close that down. There are too many things that use browser pages that are able to access stuff in your computer through 127.0.0.1, the localhost IP. And so Microsoft said, okay, there was just too much developer flak.

So before putting this thing to bed, we revisited this. And by allowing the browser to communicate to the SQRL client running a localhost server on the machine, we have what's called "client-provided session," where the communication between the authenticating remote server and the SQRL client, it receives the session token, which it then hands back to the user's browser. That inherently cuts any man in the middle out of the loop. So it makes this solution completely unspoofable. So I just thought that's too much of a benefit to pass up, even in the v1.0 release of SQRL. So we brought it back with a vengeance, and it'll be there from day one.

PADRE: We keep having these stories that show us more and more of the advantages of having some sort of authentication technology that doesn't require a secret to be kept remotely. Because we've just, I mean, we know, I think we're wise enough now to realize anything that you give, a secret that you give to a third party eventually, at some point, is probably going to leak. It's just - that's just the nature of the beast.

And I'm wondering, one of the things about this story is the fact that there are so many large companies that are customers of OneLogin. They include Amazon, Microsoft, Cisco with their WebEx, Google Analytics, and LinkedIn. We use services like this because we want to think that it's more secure for us to use a trusted third party, rather than us keeping all of these loose usernames and passwords. But at some point the Internet and the cloud economy keeps pushing us to this idea of persistent authentication, where our identity is known, no matter where we are on the cloud, no matter what service we're using. But do you think it's possible to have that persistent authentication if multiple third parties need to have a secret that they can't let out?

Steve: Actually, SQRL provides that, too.

PADRE: See, that's where I'm thinking it's going to work. I think SQRL is - it's not just a better way to do passwords. It is the persistent authentication that we were looking for.

Steve: Correct, because SQRL can work in a strict two-party mode, where you authenticate with a site that you are wanting to log into. But it also supports a three-party mode. That is, for example, say that the federal government wanted to run an identity service so that you could log into the IRS and the DMV and a bunch of other government services, all with a single known identity.

So the idea would be that any of those government websites would show you the SQRL URL, but rather than the URL being to that site, it would be to - we'll just make up something and call it federated.identity.gov. And so the idea is you're actually authenticating to that site, which means your identity is the same for all of the different government sites that you want to log into because, in the same way that OAuth arranges a third party, SQRL supports that concept, as well. And so the idea would be that you actually establish your identity only once with federated.identity.gov, and then all government sites are able to refer to that singular identity and just create new sites as they want, without having fragmented identity. So it isn't necessary.

And I think the big difference is, I mean, the biggest hurdle we have is the recovery problem because what you have with a third-party is you've got someone to go crying to if you forget your password. And that is, so for example, if you forget your OneLogin identity, you do some sort of recovery with them, and in the process you recover that singular identity to all the services that use OneLogin. But that, I mean, inherent in that is the vulnerability. That's the single point of failure.

So what's different with SQRL is there is no one to go crying to. There's no "I forgot my

password" recovery because there isn't built into it a third party. So instead we have, like, built into the SQRL system is a set of ways to allow the user to recover their own identity. Even in the event of theft, you can take it back from somebody who has absconded with it. So anyway, as soon as I get this thing launched, we'll have a lot of fun covering all the details and explaining how this system provides a robust, scalable identity solution for the Internet with zero vulnerability.

PADRE: You know, Steve, it's interesting because what we're talking about right now is a quintessential function versus convenience issue.

Steve: Yes.

PADRE: I can't tell you how many times I've been to a Vsites or a Black Hat talk or a Defcon panel, and we're talking about this sort of persistent strong encryption, strong authentication. And then you get to the question of, well, what happens when Grandma loses her password? The correct answer is it's gone. That's it. Whatever it was that you were using to authenticate yourself, if you lose that, you no longer can authenticate yourself. But every commercial service out there right now it's, well, there is a way to recover. And if there's a way to recover, it means you've now introduced a point of failure into your protection scheme.

Steve: Exactly. Exactly.

PADRE: But we're not willing to give that up. I don't think I'm - even me, I'm paranoid, but I'm not willing to give that up because I know sometimes I do stupid things. Maybe I'm going to be one of these people who changes my password because I took an Ambien one night, and I went sleepwalking, and I just changed all my credentials. You know, something as stupid as that makes people think, well, maybe there should be a last resort.

Steve: Well, we have one. It's called the "rescue code." I'll explain how it works as soon as we get there.

PADRE: Absolutely.

Steve: So there actually is a means for recovering, as long as you do one thing once. And, I mean, so exactly as you said, there is no way around that question of responsibility. But essentially we have made this as recoverable as possible. So I look forward to explaining it all in some detail.

We should mention that there was sort of an overhyped vulnerability that was found by the guys at Qualys Security in looking at the source for the processing of Linux's sudo command. They found a really obscure bug. So I call it overhyped because it's already been addressed. It's one of those things where, okay, yes, Linux users should update. Red Hat immediately pushed out patches for Enterprise Linux 6 and 7 and Server. Debian has released fixes for its Wheezy, Jessie, and Sid releases. All of the various Linuxes are going to be responding.

It wasn't a zero-day. It's not known to ever have been exploited, and it's not a network vulnerability. It's only a local privilege escalation where, by doing this bizarre series of things, and I'm not even going to bother trying to explain it because it'd just make everyone's eyes cross. I've got a link in the show notes because OpenWall.com has the details listed. But, I mean, you've got to go through a huge number of funky steps with a bunch of manipulations. Essentially it ultimately allows someone without root privilege to

get persistent root privilege by cleverly abusing a side effect of the way the "super user do," the so-called "sudo" command functions. So I just wanted to let everybody know.

So Sudo v1.8.6p7 through 1.8.20 have been vulnerable, and the p1 variant of 1.8.20 and subsequent, presumably, are not vulnerable. So just check in with Linux. Again, it's local privilege escalation only, and really not something to worry about. But definitely worth upgrading. Oh, I shouldn't say "not something to worry about." It entirely depends upon the environment of that machine and people, like the population of users who have access to sudo and the ability to do all this other bizarre stuff that's necessary in order to turn it into an exploit. Just, you know, fix it and don't worry about it.

PADRE: My understanding was that this had to do with the way that the vulnerable versions of Linux handle the terminal, TTY.

Steve: Correct.

PADRE: Essentially it's field 2. So field 2 they don't check for white space, so they don't, when they're parsing it...

Steve: Exactly. You can have an embedded space and, yeah.

PADRE: And it's supposed to, and it doesn't. So what can happen is you can actually use that - it's basically a parsing attack. You're attacking the way that Linux is parsing the individual commands on the terminal.

Steve: And what is a parser? A parser is an interpreter.

PADRE: Yeah, that's all it is.

Steve: So once again, we're interpreting something, and there is a mistake in the interpreter, which we keep seeing.

PADRE: But the important thing is there is a lot of hay being made about this. It is not a remote privilege escalation. You actually have to be sitting at the machine. In which case, if there was an attacker at your machine, you've got bigger problems than a terminal parsing issue.

Steve: And Padre, you also have to be a rocket scientist. I mean, if you look at this stuff, I mean, like what you have to go through, it's like, okay, you probably already have root brain power if you're able to take advantage of this.

PADRE: Last night when you sent me over the notes, I saw this, like, oh, interesting. So I started up a terminal, and I started trying to do this. And I got it working once, and I just gave up. I'm like, this is way too complicated.

Steve: Speaking of which, we never closed the loop. Leo mentioned that you were going to play with decrypting WannaCry...

PADRE: Yes.

Steve: ...by trying to find the key that was persistent in memory. Whatever happened with that?

PADRE: So it does work. I was able to get it to work. It finds the two primes in memory, and then it recovers the public key, and then it can reconstruct the private key, and it does the decryption. However, you have to get really, really, really lucky. So this is one of these things where it's time sensitive because WannaCry doesn't delete the two primes from memory, but that doesn't mean that they're protected. Those memory cells can be overwritten. So if WannaCry does any further processes, or if you do anything to interrupt the process, so reboot the computer or kill it, it's gone. It's entirely gone. Even when it was a perfect 100% laboratory condition, where I had the WanaKiwi installed on the computer already, so it was ready to go. I infected it. I let the warning screen come up, went directly into a command shell, ran WanaKiwi. I was only able to recover, what was it, like a third of time. So it, yeah, it's not a great last-ditch solution.

Steve: And we talked about how WannaCry uses the Windows crypto functions, and there was a "destroy context" function which for a long period of time wasn't securely wiping the memory of the context before releasing it back to the system. So the keys were just floating around out there.

PADRE: Keys were just sitting there. But again, the issue is, unless you've already got it on your computer - so you've got to have WanaKiwi on the computer ready to go.

Steve: Right.

PADRE: Just the process of bringing up a browser and...

Steve: Too much churn. It'll churn it up.

PADRE: Yeah. Yup, gone. And you have to find both. So if you find one, that's not enough. You need both. All right, Steve. We've got OneLogin hacks. We've got a Linux exploit that's not as bad as it was first reported. What else do we got?

Steve: Okay. Now, for this kind of story, I often tell Leo to make sure that he's well centered over the ball that he's sitting on because otherwise he may, when he hears this, he may fall off of his ball. And this is just incredible. So I'm reading along in a report in the Guardian.co.uk, a report titled "U.K.'s Trident nuclear submarines 'vulnerable to catastrophic hack,'" where a security thinktank does not believe that the fact that submarines are inherently, okay, the term is "air gapped," but in this case water gapped, should support the level of complacency that the bureaucrats appear to have.

In the show notes, for anyone, especially if you're in the U.K. - although I'm not sure it matters where in the world you are if there's an aberrant nuclear-tipped missile launch. I do have the "Hacking U.K. Trident" PDF link in the show notes. It's a 38-page report titled "Hacking U.K. Trident: A Growing Threat," which states that the U.K.'s Trident submarine fleet is vulnerable to, as the Guardian quoted, "a catastrophic cyberattack" that could render Britain's nuclear weapons useless. It warns that a successful cyberattack could "neutralize operations, lead to loss of life, defeat, or perhaps even the catastrophic exchange of nuclear warheads," and then they say, parens, "(directly or indirectly)."

Okay. So that doesn't sound good; right? The report says: "Trident's sensitive cyber systems are not connected to the Internet or any other civilian network. Nevertheless," the report reads, "the vessel, missiles, warheads, and all the various support systems rely on networked computers, devices, and software, and each of these have to be designed and programmed. All of them incorporate unique data and must be regularly upgraded, reconfigured, and patched." And apparently that has not been happening. The

U.K. has four nuclear missile-carrying submarines which are in the process of being replaced. And the replacements are scheduled to go into service not for a while, in the early 2030s. Okay?

So then, as I'm reading, I encounter what I described at the top of the show as, and I believe this is U.K. terminology, a "gobsmacking" paragraph: "The report comes after the cyberattack last month that disrupted the NHS." Remember how most of the U.K.'s hospital systems and medical infrastructure was completely shut down. This report says "that disrupted the NHS, which uses the same Windows software as the Trident submarines." I said, "What?" Yes, the U.K.'s Trident submarines are running Windows XP.

PADRE: You know, we could have just skipped all of that and just said, "Here's the gobsmacking part. There's a bunch of nukes run by Windows XP."

Steve: Oh. Page 23 reads: "The Submarine Command System" - the SMCS. We know that the military loves their acronyms. "The Submarine Command System was first created for the Vanguard-class submarines as their tactical information and torpedo weapons control systems. It has a long and complex pedigree. Its updated versions are based upon a version of Windows XP and known colloquially as" - I can't even say this - "as 'Windows for Warships.'"

PADRE: Oh, that's almost as good as Windows Bob. Do you think Clippy is going to drop by the U.K.'s Ministry of Defense and say, "Hey, we noticed that you're running Windows XP. Would you like to upgrade to Windows 10?"

Steve: Oh. "These have now been installed on all active Royal Navy submarine classes. Both Windows-based and Linux-based operating systems hold the legacy of vulnerabilities from the original systems, even though they operate on obscure and classified equipment and run bespoke programs.

"In 2002" - so 15 years ago - "it was proposed to convert SMCS" - that's the Submarine Command System - "to run on standard x86 hardware redesigned specifically for naval command systems. The plan" - and I'm reading from the report - "was to convert the SMCS infrastructure" - which used to be proprietary - "and applications to run on Microsoft Windows XP" - because what could possibly go wrong? - "and known as SMCS-NG (for Next Generation) or" - as it's now commonly known - "Windows for Warships. This is based upon a variant of Windows 2000 and Windows XP." So, yes, even really old XP.

PADRE: So it's not even XP Embedded. This is just sort of a stripped-down version of the operating system.

Steve: Yes. "SMCS-NG was retrofitted into all Royal Navy submarines by December of 2008. The software is supplied as a universal release configured for the sensor and weapons fit of each submarine." The report finishes, and of course this is well known to our listeners: "Windows has an entangled monolithic structure, as opposed to a modular architecture. It is therefore impossible," they write, "to change the proprietary operating system by means of reconfigurations and third-party modules. This structure of" - and they wrote "consumer-friendly." And I thought, it's not so consumer-friendly if it lands a nuke in your neighborhood.

Anyway, "The structure of the consumer-friendly operating system exposes potentially vulnerable services" - potentially vulnerable, uh-huh - "services and features" - that brings a new meaning to the term "potentially vulnerable" - "that might not be required

for the adequate functioning of the submarine."

PADRE: Oh, I don't know about that. I mean, I know I'd want my weapons officer to be able to play Minesweeper.

Steve: Who doesn't want Skype? You want Skype in your Trident class nuclear armed submarine.

PADRE: Okay. But Steve, let me back off here for a second because we have covered this sort of story on This Week in Enterprise Tech on the TWiT.tv network when we were talking about U.S. military nuclear infrastructure, specifically how some of our infrastructure dates back to the '70s and '60s. And we can look at that and say, oh, my gosh, that's so old. They're using these huge floppy disks. But...

Steve: No, please, please, please stay there.

PADRE: Exactly. At the same time it's like, yes, we know that that's antiquated hardware and software. But it's not vulnerable. I mean, it's not like a consumer operating system, where you can pick up something like Metasploit and put together an attack package in under five minutes and send it on its merry way. I have no way to infect nuclear launch machines from the '60s that are still using vacuum tubes and tapes.

Steve: Correct. And it's like, you know, we were all often told that the flight control computer in the NASA shuttle systems had less power than your typical calculator. But similarly, yes, and nothing can go wrong. There isn't an extra word of memory available to contain malware in these. So they can't. Unlike Windows, which is like, how much RAM do you need? Oh, no problem.

PADRE: Well, but Steve, this is just an extension of the underlying issue behind what we saw with WannaCry, where there are just some systems that are very difficult to upgrade, very difficult to extract your programming, your code. I mean, this is what happened to the National Health Service in Britain, where they were saying, look, we're not ready to upgrade to a new operating system. All of our software has been written for XP. And there are so many regulations about what we can and cannot do because it might endanger patient information that trying to upgrade to a new operating system isn't something that we can choose in even a decade.

Steve: In other words, legacy's long tail.

PADRE: Yes. Yes.

Steve: It's just that, I mean, that is the problem, the long tail of legacy systems.

PADRE: And it's easy to look at a story like this and say, well, this is stupid. They need to upgrade. But we're going to have the exact same conversation in 10 years because they will have spent billions of dollars to upgrade, and their new operating system will no longer be new.

Steve: Right. And, I mean, the problem with using commodity operating systems is that, if it's the same - one of the things, another recurring theme is the danger of homogeneity in our systems. If we're all using the same stuff, then a problem is ubiquitously bad because it can infect everything. If instead we had a much more heterogeneous, yeah, a heterogeneous world, then we would be siloed, and the danger from a threat would be much less widespread, much more contained.

Speaking of threat containment, last Tuesday, okay, so a week ago yesterday, the nation of Ethiopia turned off the Internet for themselves. They shut down Ethiopia's access to the Internet. Why? Because the following day was high school test day for both the 10th-grade national exams and a little more than a quarter million 12th-graders taking their university entrance exams. The problem is that, a year before, the exams had been posted to the Internet, which caused a huge upheaval. So the Ethiopian government solution was, okay, we'll just shut down. We will go dark. That way, if anyone posts the exams to the Internet, the students will not be able to get them, and the tests won't have to be retaken and invalidated.

The report that I read said: "Outbound traffic from Ethiopia was shut down around 4:00 p.m. U.K. time" - this was in the Guardian.co.uk - "on Tuesday, according to Google's transparency report, which registered Ethiopian visits to the Google's company sites plummeting over the evening. By Wednesday afternoon, access still had not been restored." Right. They wanted to keep it down while the students were taking their tests because a year ago activists leaked the papers for the country's 12th-grade national exams, that is, the college entrance exams, calling for the postponements of papers due to a school shutdown in the regional state of Oromia.

Now, apparently, the government took the move to shut down Internet access as a preventative measure. And it's worth noting that an emerging economy such as Ethiopia has a far lesser dependence upon the Internet than any post-emergent economy. Obviously, the U.S. and all major economies now on the planet have become so dependent upon the Internet that our economies would collapse overnight if networking were to disappear. So I just thought it was interesting that, yes, we'll just shut down the entire country's access in order to briefly mask any potential exposure of sensitive material during test-taking time. I don't know if that will be an annual event going forward.

PADRE: Yeah, it's interesting because this would be impossible in a modern Western world because there is no switch to turn off the Internet.

Steve: Right.

PADRE: There's no big red button. The infrastructure is so diverse, I mean, it's designed to be robust.

Steve: Yes.

PADRE: But when you go to a developing nation, and actually in Ethiopia it's something like 78 or so percent of the country is connected wirelessly. That you can control. That you can actually shut down relatively simply. You just talk to the carriers. But any time you get a more advanced infrastructure, you just can't. Still, it's kind of a novel approach. I actually don't think it's really going to work because those who want to cheat will find a way to cheat. People were cheating before the Internet. People were cheating before mobile phones. People will be cheating long after the Internet has been shut down because of test taking.

But, yeah, you're right, anytime you get to see a story like this it's sort of an interesting microcosm of what if because, you may remember, we've had people in this country who have said there should be an Internet kill switch. And it's sort of like, oh, okay, can we actually think about all the logistics that would be involved in trying to do that?

Steve: And you know, I think everyone would acknowledge that we're now past that. I

mean, nobody in their right mind imagines that you could turn it off, and it wouldn't be the end of the world as we know it. I mean, today it would be.

Speaking of the end of the world, Leo talked about this over the weekend, and I thought this was - I needed to bring it up just because it was an interesting story of woe. And apparently there is a takeaway potential solution. I meant to do this yesterday, and it just slipped my mind. But on Medium.com an individual named Cody Brown told the story that was very upsetting to him, the loss of, which he watched occur, of $8,000 U.S. worth of bitcoin. And I think it was actually a couple different cryptocurrencies, but a total present value of $8,000 disappeared in 15 minutes as a consequence of him being the target of an attack. One of the other sort of principles that we're seeing emerge is that it is very difficult for a targeted attack to be thwarted. That is, our systems are so porous that they can be set up to resist penetration, but that human factor still comes in. That's the ultimate Achilles heel.

And it sort of gets back to what we were talking about, Padre, about if there is recourse, then that's the point of vulnerability, if nothing else. So anyway, so what happened in this case was - and Cody lays this out in his blog posting. It actually, the snapshot of the picture is our Picture of the Week because he took a picture of what his phone showed him. He was just sitting around in the evening and up pops a message: "Free VZW Message: You're on the phone with Verizon and just authenticated with an alternative method. Not you? Please call us at 800-922-0204 immediately."

Well, it wasn't him. And he's like, what? So he immediately calls the number. And he gets the message: "You have reached us outside of our normal business hours. Please call back." And then, you know, between Monday through Friday, whatever it is. And he's like, uh, okay, wait. So he starts scrambling. He runs around, tries to find another number. He sends some emergency tweets and a while later gets back a response. Meanwhile, he sees a message indicating - he's watching powerlessly as his Google Mail account was taken over, and then the password was changed. And then he starts getting notices from Coinbase that money is being transferred from his Coinbase account, the wallet that he has stored there, and $8,000 disappears.

So piecing this together, what he realized is that somebody, an attacker, decided to go after him, thus a deliberate focused attack. They contacted Verizon and said, hey, I've got a new phone, and so I want to change my account over to this new phone. And so the Verizon person said, okay, what's your password? And the attacker, who of course didn't have it, said, uh, I forgot it. And then the Verizon person says, okay, what's the PIN that you've registered with us? Oh, says the attacker, it's been so long since I last used it, I forgot that, too. And so the Verizon guy said, okay, so what is the answer to one of your security questions? Oh, says the attacker. Damn. I don't remember.

And so the Verizon person says, well, do you have your billing information? Oh, yes, I do. Because that's nominally publicly available. The person did some research into Cody, figured out maybe where he lived, some things about him, enough things, although none of them should have been used for account recovery. The person ended up using some billing information in order to convince the Verizon person that they were Cody Brown, and they changed over to the new phone. Now the attacker had the phone registered to that number, so all of the SMS-based second-factor authentication went to the attacker's phone, rather than to Cody's physical phone, and they then basically bypassed the second-factor, the SMS second-factor authentication and had their way with him.

So we've been talking now for quite a while. I mentioned this first when I moved finally from Network Solutions over to Hover as my domain registrar, and I was very glad to see that Hover gave me the option. It said, you know, do you want to use multifactor

authentication? I said yes because my domains are very important to me. And they said, you know, they gave me a choice: send an SMS message or use time-based token authentication. And I immediately choose time-based because, as I have explained many times on the podcast, the problem with SMS, even if it was more secure than we know it is, is that it is fundamentally weaker because it requires a per authentication communication, rather than a one-time setup of a shared secret, which then is used to forever derive, you know, prove that you both know the shared secret based on an algorithm and time in order to produce the six-digit token that changes every 30 seconds. So here again is a perfect example of the deliberate exploitation of SMS-based second factor and how it can be bypassed.

But the other part of this, again, where I'm like testing myself with SQRL, how do we prevent that, too. And the good news is it's already in there. In the show notes is a snapshot of SQRL's Settings & Options dialogue. And there's two checkboxes which are not on by default. That is, when you start using SQRL, they're not on, because I want to wait for people to understand how it works and get comfortable with it, and then they have the option of turning them on. The first one is labeled "Request only SQRL login." And the second one is "Request no SQRL bypass." And I labeled it that way because those are flags which are provided as part of SQRL's interaction with the authentication site, with the server.

And so they're regarded as preferences, but they allow the user to express their request, their preference, to disable alternative forms of login, meaning username and password won't work anymore and so forth, and also to request that even SQRL not be bypassed. That is, formally say I am going to take responsibility for my use of this system, and I want no recourse. So in other words, if this was set, no person at Verizon would have the ability, that is, the technology would block them. They would not have the ability to respond to an attacker under any circumstances saying, oops, I lost my SQRL identity, sorry, here's a new one. They would say, I'm sorry, we don't have the ability to do that for you because you told us to turn that off.

So again, this is meant to be a true bulletproof solution. And again, we're not forcing anyone to do this. It's off by default. But once you understand how this works, you see that it's working and you've been using it for a while, you decide, hey, you know, I get it, then you can turn this on. And as you then log into sites with those flags set, that persistently sets, sort of leads that trail in the site that you visited, saying I'm going to take responsibility for my authentication to your service.

PADRE: Right. And this is something that we've known for a while, that the weak link is that human interaction who can decide arbitrarily to not follow the procedures. In fact, you know, this is one of those cases where clearly Verizon is at fault here. They did not follow any of the procedures. Unfortunately, I don't think we actually have any rights to force them to follow the procedures that they say that they were going to do.

Steve: Correct.

PADRE: Interestingly enough, a few years back I was doing a little bit of work for my other organization, and I was doing a mass swap of a lot of SIM cards for some of ours who were going overseas. And I was dealing with one of the big four carriers. And the first one that I did, all he asked for was the IMEI. I didn't need the phone number. I didn't need the account number. I didn't need the account name. Just the IMEI that it was currently on, and the IMEI that I wanted to switch to. And it struck me, I was like, that is the worst authentication I've ever seen. I can pull the IMEI off of any of a number of data sources. And are you telling me that if I'm just pretending to be the IT person for this person, I can suddenly start getting all the traffic that he or she was supposed to get

on their phone?

Steve: And it goes - IMEI goes through the air.

PADRE: Yeah, exactly. It's cleartext. If you're playing man in the middle, you know exactly what is talking to you. And that's one of the easiest to obtain pieces of information. And the fact that they were willing to do something as radical as switching the service to a different phone, without even challenging one bit of information that I should know about the account, that just - I lost all confidence in their security procedures. But on the other hand, the person was trying to be nice. I mean, that's the thing. They were a salesperson. They're like, okay, of course, we want to give you a good customer service experience here. We're going to make this as easy for you as possible.

Steve: Right.

PADRE: And I get that, and I appreciate that. But then the IT person in me is saying this should not be happening right now. And I did 20 of them in, like, 10 minutes, without ever mentioning a phone number or name.

Steve: Wow. Wow.

PADRE: This is actually - I put this hand in hand with something that one of my law enforcement friends was telling me, saying that mobile phones and mobile devices has been the biggest boon to law enforcement in the history of law enforcement because people just assume that they're secure when they're not. And people just assume that they can put personal information on it, and it will be protected, and it won't be.

Steve: Well, and of course that's the argument against law enforcement saying they need access to decrypted data. I mean, there's never been more access to data than there is today. I mean, it's just - it's a cornucopia of stuff. We have all of our IoT stuff, which is sucking in data. And we've got commercial companies tracking us and profiling us and building up information. And we know that in the U.S., at least, thanks to the Patriot Act, we have the government able to say that they have cause and need for the disclosure of some of that information, anything that a company has available to it.

So, yeah, I mean, the fact is, I mean, I understand the "going dark" problem. We discuss it here all the time. When I see another attack, another terrorist incident, as we had recently again in the U.K., and Theresa May gets up there and says "We need to be working with Silicon Valley, we need to be working with the technology companies, we cannot allow the bad guys to hide," I just, you know, we're marching towards a day where there's going to have to be some sort of compromise made, where Apple is not going to be able to say we have no access, we cannot get access to our customers' data. I think we're going to end up seeing legislation with some sort of a multiyear horizon by which time these companies have to be able to make that available. And we also know that the crypto horse is out of the barn. I mean, once that happens, the bad guys will use crypto that doesn't have a means for a corporate sponsor to provide visibility. So I think it's going to all be for naught.

PADRE: The last time I came back into the country was just a couple of weeks ago. And I have global entry, so I've got the little card that allows me to go quickly through. But they can still pull you aside for secondary. And so I get pulled aside for secondary, and so they wanted to see my phone. And I have a different phone that I use when I'm international. It's a burner phone.

Steve: Do you wear your collar when you're traveling?

PADRE: I do, I do.

Steve: And even so.

PADRE: Even so. I think the collar really attracts them.

Steve: It could be a disguise, I suppose.

PADRE: I think so, exactly. But so the first thing they do is they said, "Well, your phone seems to have a fingerprint scanner, but can you open it with your fingerprint?" I said, "No, it's a PIN." And they said, "Well, why don't you have a fingerprint?" I said, "I don't want people to be able to open it with just my fingerprint." And so that was kind of a red flag. And so I opened it for them. And of course everything looks right. It's got an email account. It's got text messages. They're not my main accounts. They're sort of the burner accounts I've got.

Steve: Uh-huh.

PADRE: And then they're looking through it for a few minutes, and the agent comes back and says, "Do you happen to have Dropbox and OneDrive?" And I'm looking at him going, you're hoping I have the app on my phone so you can go through my personal documents. I mean, that is horrible. That is completely out of control. But, yeah, you're right, this is a gateway. And so if you give people an unwarranted way into everything that is connected to you, that cannot end well. I don't know, Steve.

Steve: Wow. I know.

PADRE: I think I'm going to go back to writing in Latin on pen and paper.

Steve: Or I guess can you maybe just FedEx the phone to yourself before you get on the plane?

PADRE: We're probably going to have to do that because once they ban laptops...

Steve: And just not have the technology. Wow.

PADRE: Well, it's interesting because on my way back, so I flew through Amsterdam, and they actually asked me when I was in Rome, they asked me to check my laptop because they were afraid that new regulations would come through while I was in the air. It was like a 22-hour trip. And they said, "We don't want you to have to hand in your laptop in Amsterdam." And so I checked in my laptop. But now they're saying they may not even allow that. And I'm thinking, so what good is air travel to me as a technology person, if I'm not allowed to bring any technology with me?

Steve: Wow.

PADRE: That's silly. All right, Steve. So we've talked a little bit about exploits, exploits, and more bad news. Can you give me something positive?

Steve: Not yet. We've got that coming up. But John Matherly is the founder of Shodan, that is the search engine that just - it's the gift that keeps on giving. As we know, much

as Google indexes websites and allows us to find stuff, Shodan has indexed essentially, or you might say that Google indexes port 80 and 443, that is, HTTP and HTTPS. Shodan indexes everything else. And so it's a search engine sort of slash database that scans the public-facing Internet IP space looking for things that answer all of the other ports. We've got ports 1 through 65535. And so, for example, it's the thing that, if some security camera runs a server on port 6273, you can say, hey, Shodan, what IP addresses have something listening on 6273, and it gives you a list.

Well, it turns out that it's also scriptable, and John wrote a script, the founder of Shodan wrote a script last week because he just kind of got curious. He wanted to find out how many, if any, HDFS protocol servers were there. HDFS is the Hadoop Distributed File System. Hadoop is an open source Apache-based evolution of a project that Google originated many years ago to experiment with large dataset, large cluster computing in an open source environment. So thus Apache-based, and it's like a big dataset processing facility. And Hadoop is just sort of a fun word. It's not an acronym. It was actually named for the elephant toy belonging to the son of one of the project's early originators. So the elephant's name was Hadoop, and that became the name of this system.

So John writes this script that queries his own Shodan backend database and discovers 4,487, right, 4,487 public IP instances of HDFS because it runs on a set of well-known ports - 50070, 50075, 50090, 50105, and so forth, like that. There's, like, seven of them. And then he, using his script, queries those 4,487 instances of HDFS for, like, what's there. I mean, these are publicly facing, unauthenticated, no security databases exposing - and this is where I tell Leo to make sure he's centered over his ball - 5,120 terabytes. That's 5.1 petabytes of big data. It's just - oh, lord. And so, okay. So that begs the question to me, wait a minute. No security? No authentication? How is that possible?

PADRE: These are default installations. These are all default installations.

Steve: Yes.

PADRE: Oh, my goodness.

Steve: I dig into it a little bit, and I learn that setting up HDFS security is a bit of a nightmare. You know, it's open source. It's public. It's oh, well, yeah. Turns out that it requires far, far more configuration involving the Kerberos authentication service and servers. It is not easy to do. So it typically doesn't happen. I imagine if it were simple to do, people would go, oh, yeah, why not? But it's not. It turns out it's really difficult. The way the system was built, it is difficult to make them secure. And so people are like, uh, well, okay, once anything important is in there, maybe we'll think about adding that. And it never happens. Unbelievable.

PADRE: You know, this is a lot like - was it last week that a researcher found a bunch of AWS baskets that weren't encrypted and completely open. And in fact one of those baskets contained a lot of information from a security contractor who was also dealing with big data and big data analytics. Big data tools aren't new, and they install a lot like the tools that we've already used, which means that there is a way to just keep hitting Enter and get all the default settings. But the default settings...

Steve: Yeah, yeah, yeah, yeah, yeah, yeah, yeah.

PADRE: Those are for dev environments, not for production environments.

Steve: Right.

PADRE: Oh, Steve, Steve.

Steve: I know. Under the topic of "We should have seen this one coming," it turns out - and this one was a bit of a blast from the past. Check Point discovered a startlingly high incidence of - they're calling it malware. It was exactly what actually caused me to coin the term, as Leo often reminds people, that I coined the term "spyware." As far as I know, 17 years ago that term didn't exist. But let me back up a little bit.

So there is a Chinese digital marketer named Rafotech, R-A-F-O-T-E-C-H, who are behind the spread of a malware family which has been [audio dropout] Fireball. It's apparently installed in - and Check Point's number was 250 million web browsers. Rafotech's own PR claims 300 million. So I think the 250 million is probably a good number - 250 million. A quarter of a billion web browsers have this stuff. It's ad revenue-generating zombies. Rafotech- or Fireball-laced apps, Check Point says, have infected 20% of corporate networks around the world.

Check Point explained in their report published last Thursday that the malware hijacks browsers and generates revenue for this Beijing-based digital marketing agency. They termed this "possibly the largest infection operation in history," adding that the Fireball infections could be turned into a distributor of any other malware family because this thing has the ability, if it chose, to download anything into these 250 million computers where it is resident, if it wished to.

So, okay. This, as I said, reminded me of 17 years ago, back in March of 2000, a little over 17 years ago, after I had downloaded and installed the very popular at the time WinZip freeware. I had been using Phil Katz's PKZIP utilities for a long time. But Windows was happening 17 years ago, and I thought, okay, I'll take a look at WinZip. So I download it, look at it, and it was like, eh, okay. And it didn't really move me, so I uninstalled it. And then sometime later I noticed something foreign running in my machine.

It turned out it was this stuff called Aureate, A-U-R-E-A-T-E, which was adware, which various freeware apps at the time were bundling. And I coined the term "spyware" because, aside from just - the idea was that this WinZip, when it was onscreen, would have a window in it, much like browsers now present advertising. This would be a window showing an ad hosted from the Aureate servers. So this thing would be - so there was an Aureate DLL running persistently. It would reach out and obtain ads from this Aureate ad network.

Turns out that the dialogue was in the clear. I put a packet capture on it and found it was also logging all the software that I was using, everything I was doing on my computer, and sending that back in reports to the Aureate home, the Aureate mothership. Thus spyware.

And so I wrote something called OptOut, which allowed you to surgically remove this from your machine. And it turned out that a vast number of people had this installed in their machines without knowing it and that Aureate's specific instructions, even though I had uninstalled WinZip, they said when you uninstall the freeware which brought it, do not remove us because other freeware may be sharing the same installation instance, and we wouldn't want them to have their revenue system removed. So it was insidious.

And of course that was 17 years ago. Now here we are in 2017, and we have something

very similar. This is brought into people's machines with a bunch of freeware which is like, here, download this, and it'll do something for you. It of course takes over people's browsers, installs a search surface in front of their browser that tells them they're going to get all kinds of benefits and things. And in fact it's used to generate revenue for this company and to forward their search queries onto existing search services behind the scenes. So I doubt that our users are being affected by this, but keep on the lookout because it is everywhere.

PADRE: I wouldn't be surprised because my father, who is tech savvy, has this happen to him quite a lot. In fact, every time I go to visit him...

Steve: His browser's been commandeered?

PADRE: I'm basically rolling back his computer. I've got images. I don't even count on uninstalling. I just go back to an old image. And it's always, "Well, I downloaded a screensaver that shows me pictures of my grandchildren." And I'm like, "Okay, Dad, but there's other stuff that installs, and you can't do this." He goes, "But it works." And that's the thing. They bundle with software that works. So when you use it, your fears are assuaged. You just go, okay, it must be legit.

There's a corollary to this. There was a study that was done, I want to say three years ago, that they showed us at Black Hat. They did a very convincing video. They wanted to see if people's behaviors would change if the install packager told you exactly what was being installed, versus surreptitiously trying to install something on your computer.

Steve: And nobody cares. I know.

PADRE: No one cared. They would just okay, okay, okay. Now, occasionally someone who noticed there was an extra checkmark, do I want to install this, this, and this, and they'd go and they'd uncheck it. But by far most people, regardless, I mean, it could actually tell them we will be installing software that will spy on you and redirect your browser, and people would still click through. That's not fixable with technology. That's a pay attention to what you're doing because this is important.

Steve: Yup.

PADRE: Steve, is there any hope?

Steve: Well, not yet. In fact, we have another instance - we've been talking recently about Google's problems with Chrome and things that they say are not bugs. Now, in the same way that LastPass has been a victim to some degree of its own success, meaning that it's what Tavis is thinking about when he's in the shower, similarly Chrome is becoming its own victim of its own success. It's now the majority browser on the Internet. So of course it's what people are looking at more closely. Modern web browsers, that is, those that support the features of HTML5, are able to use the standardized WebRTC features in order to create web page-based audio processing and audio/video teleconferencing apps and so forth, without the need for a third-party add-on. So whereas once upon a time you needed to use a Flash-based app or Silverlight or a custom add-on to your browser, now it's in there.

The problem is that there is a system where the site must get your permission on a per-site basis to enable WebRTC. So there was some attention in the spec paid to the implementation. The problem is the permission is sticky, and it remains until explicitly revoked. So it's one of those things where you'd want to perform an audit on your own

from time to time of the set of apps that you have given, you have previously given permission to. You'd want to curate that list. And we've talked about this, how for example sometimes you want to go into your Twitter account and look at the different apps you have given access to your Twitter account to and say, you know, a few of them you'll still be using, but over time they just sort of fall by the wayside. Yet unless we explicitly remove permission, it just doesn't go away.

So here's the problem in Chrome. The Chrome UI, technically it's called the browser Chrome, so the Chrome Chrome, shows a red recording indicator, a red dot with a red circle around the red dot, on the tab of any page which has the WebRTC actively operating. The problem is that sites are able to bring up pop-ups that don't have tabs, and they're not only pop-ups. They can be pop-unders so that you don't even see that a website that you're viewing or a site that you have open in your browser has popped under the browser and turned on WebRTC streaming without your knowledge or permission, or that is to say, without your knowledge and current permission. That is, at some point in the past you had to have enabled it. But unless you explicitly disabled it, it still has permission.

And so a researcher went to Google with a proof of concept demonstrating that it was possible to generate a non-tabbed pop-under which could surreptitiously record audio and video with no visual indication whatsoever. And oh, by the way, mobile version of Chrome doesn't even put the red dot on the tab. So eight weeks ago, on April 10th, he opened the dialogue. Google replied that it was not a bug because it's not a valid security issue, and so they have no plans to change this. He did a complete presentation in Medium.com. In the show notes I have his proof-of-concept demo where you can go there and see all this for yourself. Basically it is a no-UI 30-second audio recording which you can then click a button to download the results and demonstrate that your Google Chrome browser is able to do this without any explicit indication that it is doing so. And Google says, yeah, that's not a bug, that's a feature.

PADRE: One of the projects I made a long time ago, and actually I never showed this on Know How, was a kill switch for all audio and video devices. So it was an actual hard - not software, so you can't bypass it. It is a hardware kill.

Steve: Nice. Nice.

PADRE: So it will activate the capture devices, but they will display or show nothing because the actual physical device has been disconnected. I think I need to bring that back. I put that in a storage closet somewhere. But I think maybe...

Steve: That would be a great topic.

PADRE: Yup. So are you saying, so I can pop-under the browser window, and then just avoid the whole thing of the microphone showing up on the tab?

Steve: Correct.

PADRE: Remember when we thought we had solved the pop-up and pop-under problem, like eight years ago?

Steve: Yeah.

PADRE: Wait. Didn't we solve the pop-over and pop-under problem?

Steve: Apparently not.

PADRE: Steve, before we go on, you know what I need to have, because I need this every time we do this show together, I need you to talk about SpinRite.

Steve: Well, thank you. I do have a fun little comment here, somewhere down here. We're getting there. I'll scroll down to it. Ah. It was two tweets. Someone named Al Spaulding, who is a frequent Twitter acquaintance of mine, on the 3rd - so, what, three days ago - he sent two tweets. He said: "Steve, another SpinRite story." He said: "I started getting Outlook errors like 'can't load profile,' and eventually Outlook would not load at all." He said: "Company IT looked at it and created a new profile." He says: "That worked for about 24 hours. Then it all started again." He said in a second tweet, that was the first one, followed up with: "Finally ran SpinRite and no problems for five days now. Thanks for a great product." So Al, thanks for sharing that with me and our listeners. Appreciate it.

PADRE: Well, Steve, I've actually got a SpinRite story from the last week.

Steve: Ah.

PADRE: I did all the WannaCry segments for Know How. And I use VMs in my lab at home because I have a nice VMware cluster. There's no way I'm going to tear that thing out and bring it to the studio every time I want to do a demo. So I actually have some physical lab kits which is basically a motherboard, some memory, and some SSDs that I swap in and out because I have the same image on all of them. And I had two Samsung SSDs that stopped imaging. They would no longer image. And I thought, oh, maybe they've reached the end of their useful life. Ran SpinRite, was it Level 2?

Steve: Probably, yes.

PADRE: On those SSDs. And now they're back up to full capacity and back up to full speed.

Steve: Nice.

PADRE: So still works for me.

Steve: Yay. I love it. In fact...

PADRE: I can't even say it's - it's not even my emergency tool anymore because it's one of the first things I do. If I'm having problems with a storage device, just pop out SpinRite. That's the very first thing that works.

Steve: It's just your go-to, yeah.

PADRE: It's my go-to now.

Steve: Yeah. In fact, all of us who have had experience with it, that's, I mean, that's what we do because it just works.

PADRE: Just works.

Steve: Very cool.

PADRE: Now, we can't tease the audience with "homomorphic encryption" and not deliver. And I think it's time for some serious math. So you want to drop the math on us?

Steve: Okay. So we've never talked about homomorphic encryption. And last week I shared the news, which was a little bit disturbing, that Google had come up with a way of further measuring the effectiveness of their online ads by entering into some sort of a data sharing arrangement with physical credit card processors such that they could determine our identity and associate our offline purchases with our online presence, the idea being that, if we were at ad sites - and of course this is sort of the holy grail for advertising. If we're browsing around the web, looking at ad sites, being influenced by online advertising, yet not taking action online, but rather deferring the action to a subsequent physical purchase in the real world, how do they get credit for the influence of the ad?

Google wants to cross that last bridge. And so it was somewhat worrisome that their reach was as far as it was. I think I remember saying that they now had access to 70% of credit card transactions that would allow them to create this association. So the question was how is this being done in a privacy-respecting fashion? So homomorphic encryption is a way of encrypting data such that its encrypted form can still be used. That is, normally we think of something that's encrypted as being data in flight, or maybe data at rest, that is, like stored. But the point is you can't - you can never use it in its encrypted form because it's, as we know, it's just pseudorandom noise. If it's properly encrypted, it's just nothing. It's noise. And so we have to arrange to decrypt it in order to bring it back to its original useful form.

Homomorphic encryption changes that. It's a family of technologies that allows work to be done in the ciphertext form. Wikipedia, for example, defines it as: "Homomorphic encryption is a form of encryption that allows computations to be carried out on ciphertext, thus generating an encrypted result which, when decrypted, matches the result of operations performed on the plaintext." Which is freaky. So imagine that you encrypted something, and then you added something, like arithmetically added something, such that the result was still encrypted, but modified. But when you decrypted it, the decrypted result also had the result of that addition. Which is like, what?

PADRE: Okay.

Steve: It's like, what?

PADRE: So this is like an encryption chain. So essentially it allows me to create an encryption and then keep adding to it. So, for example, multiple services on the Internet that I might be using, they can add little bits and pieces, and they can have access to some of it, but not all of it. Is that how that works?

Steve: Yeah. So this comes from the blog posting of someone who listens to Security Now!, who heard me talking about this last week. He actually is with a company that is involved in doing this. And so here's what he wrote, and then I'll explain it a little more. He says: "We needed to share some data with another company, and this related to credit card transactions. But we did not want to share the actual card numbers. They're known in the industry as PANs, Primary Account Numbers. What to do? What we came up with was quite neat and can probably be used by others. The external company collects the card numbers they want information on. They encrypt these under RSA with a key they generate and never share." And I should mention, I forgot, I've edited this a little bit for clarity and to use some terms that our listeners will be familiar with.

"They send these encrypted numbers to us. We further encrypt them with our own RSA key and securely shuffle the order of the entries to completely confuse them so that they are now reordered" - they're cryptographically securely reordered - "and then return them to their originator." So now they have a set of these credit card numbers, double-encrypted. He says: "Then we perform an extract of the relevant transactions and encrypt the credit card numbers with our RSA key and send these as well."

PADRE: Okay.

Steve: Now the recipients of these can encrypt these keys with theirs because RSA is a commutative function. So in other words, in the same way that A times B times C gives you the same result as A times C times B, that is, the order doesn't matter. And so what these guys did was they took the original credit card numbers that were encrypted by someone else. Then they got them and encrypted them with theirs, scrambled them to reorder them so that the originator could not associate them, and they gave them back to them.

Then they took a bunch of information that they wanted, encrypted the credit card numbers with their RSA key, and provided that to the original group. They added their encryption to the already once encrypted by the other guys' RSA. And that meant then that the numbers matched up. That is, so that first company started with credit card numbers and received back doubly encrypted ones out of order. Then when they similarly doubly encrypted the credit card numbers provided by the second company, they matched up again. And so that allowed them to perform the data analysis without ever revealing the actual credit card numbers. Those were blinded by this double-encryption process, which I thought was very clever.

PADRE: And at the end of chain, when they decrypt, they get everything. So they do one decryption, and they get everything? Now, the question is, for me, does that change the original data that was inside of that original encrypt? Or do they get the different layers, and then they can perform the operation? I guess I'm a little confused about that.

Steve: So the better way to think of it is it's a little bit like a hash chain, except hashing is not commutative. So if you took something and hashed it once and then - or like an HMAC. So like a keyed hash. If you used a keyed hash and hashed it, and then a different key and hashed it again, you'd get a result. But if you swapped the order, you would not get the same result. So hashing, keyed hashing is not commutative. But RSA encryption is.

PADRE: Okay. Yeah, I get it.

Steve: And so it's not a reversible process in the same way that hashing is not reversible. But it is a forward-moving commutative process. So it doesn't matter which order you double encrypt. Either order will generate the same result, in the same way that hashing is a forward-only process. This is all forward-only, but it provides a clever technique for blinding someone to some data which you still want them to be able to compare. So, very clever.

PADRE: Very clever. Someone was comparing it to the block chain, but block chain works a little bit differently than that.

Steve: Yeah, yeah.

PADRE: But the same kind of idea where everyone can add their little bit and piece

without having to see the entire chain of encrypted data.

Steve: Right. And in this case the fact that the order in which they add it doesn't matter, that allows this clever hack. And finally...

PADRE: That's fun. I want to play with that now.

Steve: Yeah. And in fact I've got Stuart's entry in the show notes, and he shows the math, the detailed exponentiation RSA math, and demonstrates how mathematically you can show that what they're doing survives commutation. So, very cool.

And finally - @Really_Evil_Rob is his handle. I don't think he really is. He sent me a note on the 2nd saying: "@SGgrc I've been thinking about your system of printed QR codes for your authenticator tokens. What about saving them to a USB flash drive?" Now, just to back up a little bit, Father, to loop you in on this, I've been talking about how, first of all, time-based tokens are what we want, not SMS-based tokens. And what we want is authenticator apps that refuse to export because they contain private keys which need to be kept secret.

So the thing you want, especially if you're going to go through the border crossing, or a particularly, as I put it last week, officious-looking TSA agent you're about to be confronted with, you don't want your authenticator app to be willing to export your keys because that means anyone could get them. And then what they have is the keys to the kingdom. They've got all of your future time-based tokens forever. So what I explained was you first want to use apps that refuse to export.

And when you are establishing an account, as I did for example with Hover, invariably they'll show you a QR code, which you, if you were setting up your authenticator app, you would snap a picture. I instead said, no, print the page. Print it. Put it on paper. Because we don't, in this day and age, have just one device. I've got a phone. I've got iPads. I might have a Windows or a Linux-based authenticator. The point is they all support optical input. And what you want them to do is not to support optical output, or even text-based output. The idea being that every time you're setting up an account, you print the page. And you end up with a sheaf of papers. And so the idea was you store that offline.

And so Robert's question made me realize that I hadn't clearly articulated why I thought this was practical. The reason it's practical is that it's the tradeoff in security and convenience. And as we've said on this podcast, and as we're always saying, there is no question that there's always a security/convenience tradeoff. So here, if you're setting up a new device - for example, I am excited about the new iPad. One is on its way. The sort of the mid-sized iPad Pro looks like the right one for me. I did not - I haven't upgraded for years because it hasn't been right. This new one will be. I am a time-based token authenticator user. I will be setting up a new iPad. And I like Google's Auth app. I don't like LastPass's because it's just too - the entries are too big. I like smaller entries because I can get more on the screen. And there's another one I haven't taken a look at yet. But I'll be setting up this new Pad.

So I have printed out a sheaf of QR codes. And I simply snap - and setting up the new one, I let it see them each in turn, and now it's set up. And then I put them back in a drawer. So, yes, I am responsible now for maintaining these securely offline. But this is the point that I wanted to make to Robert and other listeners. No, I don't want them on a USB flash drive. I want them offline. They are too valuable to me to have in any electronic form. Electronic is not safe. It was like you were saying, switching back to writing in Latin for your notes. So this is a place where it wants to be offline because it's

practical. It's not like you need to refer to them every time you authenticate. No. It's only every time you are setting up a new device. And that's something we do infrequently.

So you want the security of, I mean, the absolute security of it being offline, not on a USB drive where you're going to insert it into something, and that something could have malware on it, waiting for you to stick your drive in and then suck the contents off the drive. Or malware running, and it encrypts the contents and demands ransom. No. This is a place where the tradeoff says, you know, take a picture when it's a QR code being presented by the site where you're establishing a new time-based token and add it to your pile of offline stored papers, which you know where they are. And the point is, yes, you need to provide some physical security to those. But that's way better. That's something we have control over to a much greater degree than anything involving online storage and technology.

PADRE: You know, Steve, this is a little bit of the security of inconvenience. We should get ourselves into the mindset that, when we have a little bit of inconvenience, especially when setting up a new device or a newly authenticated piece of gear that needs to use our credentials, if you do not experience that inconvenience, you don't have enough security. It's too easy.

Steve: Well, it's like every website needing its own password. That's incredibly inconvenient. It's also way more secure.

PADRE: Exactly. Exactly. But, I mean, the problem is our brains are not yet wired to have that reward. We think of, wow, man, I shouldn't have to go through this many steps to set up a new phone, where it should be, oh, okay. If it's this difficult for me, and I have all my credentials, it will be more difficult for someone who's trying to attack me, they're going to go after a lower hanging fruit than I.

Steve: Right.

PADRE: That's a better way to think of it. But we don't because we're lazy. I'm lazy. I'm incredibly lazy. Steve Gibson, he is my security guru. He is a man who I turn to anytime I need things like maths explained to me. Thank you so very much for sharing your time with us. I always consider it an honor whenever I get to co-host the show with you.

Steve: Hey, thank you for standing in. And we get you next week; don't we?

PADRE: Yes.

Steve: You're still here?

PADRE: Oh, yeah, absolutely. In fact, 30 minutes after we finish next week's show is when I head off to be in silence for two months. So it will be the last show I do before I go in.

Steve: In that case, we're going to talk you into silence next week.

PADRE: Well, it should take me - hopefully we'll do a show that takes me two months to decipher.

Steve: Recover from. Perfect.

PADRE: Folks, that does it for this episode of Security Now!. Don't forget that we are live here on the TWiT.tv network every Tuesday at 13:30 Pacific time. Steve is always here to be able to inject you with some healthy paranoia - and yes, folks, it is healthy - and help you to understand the wonderful world of security. You can find all our shows at the show page at TWiT.tv/sn, as well as iTunes, Stitcher, and wherever fine podcasts are aggregated. You can also find high-quality audio downloads on GRC.com, which is also where you'll find everything about GRC. That's SpinRite, that's ShieldsUP!, and everything about SQRL. I'm Father Robert Ballecer, the Digital Jesuit, saying that if you want to keep your data going into the future, it's time for Security Now!.

Steve: Thanks, Padre.