SECURITY NOW!

Transcript of Episode #631

Private Contact Discovery

Description: This week we discuss some aspects of iOS v11, the emergence of browser hijack cryptocurrency mining, new information about the Equifax hack, Google security research and Gmail improvements, breaking DKIM without breaking it, concerns over many servers in small routers and aging unpatched motherboard EFI firmware, a new privacy leakage bug in IE, a bit of miscellany, some long-awaited closing-the-loop feedback from our listeners, and a close look into a beautiful piece of work by Moxie & Co. on Signal.

High quality (64 kbps) mp3 audio file URL: http://media.GRC.com/sn/SN-631.mp3
Quarter size (16 kbps) mp3 audio file URL: http://media.GRC.com/sn/sn-631-lq.mp3

SHOW TEASE: It's time for Security Now!. I'm back. Steve Gibson's here. And we have a lot to talk about, including a little more information about how Apple's Face ID works. A judge who says, no, the FBI doesn't have to tell you anything about how it unlocked that iPhone. And Moxie Marlinspike in another discovery, this time Signals the victim. Plus the secret life of bees. It's all coming up next on Security Now!.

Leo Laporte: This is Security Now! with Steve Gibson, Episode 631, recorded Tuesday, October 3rd, 2017: Private Contact Discovery.

It's time for Security Now!, the show where we cover the latest news from the security front. It is a front. It's a war out there. Thank goodness General Gibson is here, Commander in Chief.

Steve Gibson: Welcome back from your two-week cruise around the world, wherever it was you went?

Leo: No, I just went up the river, that's all.

Steve: Up the river, okay.

Leo: Up the river. But it was a nice vacation.

Steve: Well, fortunately you came back down the river.

Leo: Yeah, and thanks to Father Robert, who everybody agrees should be hosting the show from now on. It's okay. It's okay. It doesn't hurt my feelings much. But I am glad to be here.

Steve: Well, you did announce you were loving your vacations and your travel.

Leo: I'm not giving up this show.

Steve: No.

Leo: We've been doing this 13 years.

Steve: Yes, we are in our 13th year.

Leo: Unbelievable. And the good news is no security problems at all this week.

Steve: Yeah, yeah. We're just going to...

Leo: Did you see, there's one that just broke that is hysterical. It has nothing to say about it. But Yahoo now admits that every account was hacked. Three billion, with a "B," accounts were hacked in that 2013 exfiltration.

Steve: And I liked your comment on MacBreak Weekly, that that would be the largest ever. And one reason is that there used to be much more concentration into a single provider. And now we have a much more heterogeneous environment, where not everybody is at Yahoo the way they used to be. So I think it is the case that no one outfit will be able to be as devastating.

So Episode 631 today. As I was pulling things together, one interesting bit stood out that acquired the title, which is the reason this is called "Private Contact Discovery." Our friend Moxie Marlinspike and his gang working on Signal have tackled the problem of concealing contact metadata, which is an unsolved problem, really, until now. We've talked, for example, about how, even when, for example, email content is encrypted, the fact of the mail moving from person A to person B cannot really be hidden. And so it's often the case that who you're talking to is as much a useful piece of information and, unfortunately, a breach of privacy as what it is you say.

And in the world of instant messaging, which Moxie's Signal protocol addresses - and we've talked about Signal's ratchet mechanism and what a perfect job they have done, so much so that other platforms have adopted the open Signal Protocol for themselves because it's so good. One of the remaining problems has been how does a new person join the Signal ecosystem, or the Signal universe, and discover who in their contacts are other Signal users without the central repository that glues all this together knowing that they've done so, like knowing what contacts they have in common. Anyway, they figured how to do this. So we'll wrap up the podcast talking about that.

But of course we do have our week's worth of news. We're going to talk about - I have to talk a little bit about my experiences with iOS v11. Won't take up much time, but I just want to kind of go on record with what I have found. Also, for a couple weeks there have been a number of stories about various instances of browser hijack cryptocurrency mining. And so I want to talk sort of generically about the emergence of this as a thing and what it means. We have some additional information about the Equifax hack, some amazing coverage thanks to Bloomberg. Google's security research is uncovering new problems. They have also, anonymously, some people at Google have announced some plans for improvements of Gmail security as a consequence essentially of this emerging awareness of how much Russia was involved in this past presidential election.

There were some specious rumors, well, actually stories about breaking DKIM, which is the email authentication protocol that we've actually been talking about not too long ago. Turns out it wasn't broken, but we'll explain what happened. Also there's some new concerns over many small servers in routers which are exploitable. And you also were talking on MacBreak about motherboard EFI firmware concerns, mostly focused on Apple, as you said, because the researchers were able to more easily explore various Apple Mac systems and the firmware that they had. But this - we're going to broaden it, of course, for this podcast and talk about also the impact on Windows and Linux-based hardware. There's a privacy leakage bug in IE. We'll do some miscellany. We've got a bunch of interesting, and I think pretty quick, closing-the-loop feedback from our listeners.

And then we're going to take a close look at this beautiful work that Moxie and his group at Signal have done in order to protect the discovery of other people in your existing contacts who you want to converse with over Signal without disclosing the fact that you have found them, which turns out to be a much harder problem than one would expect. So I think a great podcast.

Leo: Yes.

Steve: Our Picture of the Week just cracked me up. It caught me by surprise. Someone tweeted it to me, and it's apropos of the data disclosures that we've been having lately. It shows some bees flying around in a hive, and the caption says, "Security Failure: EpiPen's Database of Everyone Who's Allergic to Bees Has Been Obtained by Bees."

Leo: [Laughing] That's good. That's good.

Steve: I just thought it was perfect.

Leo: Oh, you're in trouble now, boy. The bees know who you are.

Steve: Yeah, the bees know who you are. They know how to selectively sting because they only get one shot at that, I think.

Leo: That is funny.

Steve: I thought that was great, yeah.

Leo: Wow.

Steve: So just briefly, on iOS 11, I don't know yet how 11.0.1 compares. But right out of the gate it was the buggiest release of iOS I have ever experienced.

Leo: What problems did you experience?

Steve: Oh, my god. Like everything I tried, every one of my devices - I have, I don't know, like seven pads and my phone - and they were all doing bizarre things.

Leo: Oh, dear.

Steve: Like I remember in iMessage one of the balloons got stuck so that, when I changed to a different message channel, the other ones were scrolling underneath this...

Leo: Oh, I've seen, yeah, I've seen weird cosmetic things like that, yeah.

Steve: Yes. And the dock got stuck on the vertical side. And as I rotated, everything else rotated around, but the dock just stuck where it was. And on one of my pads, whenever the preview, when the screen is locked where you're able to see things, every time I touch it to scroll it, it flashes white really briefly and then kind of settles down. I mean, just if Jobs were still around, heads would be rolling, or they wouldn't have shipped it. I don't know. But, I mean, I was just, you know, I love iOS. I use it as much as I use Windows. And it just completely collapsed.

Leo: So what did you do? Did you roll back?

Steve: No, I just kind of shook the pads more and turned - I often turn them off and back on, and then it's okay for a while. I mean, it's really - and I'm hoping that 11.0.1 that came out, what, late last week, hoping that it's going to be an improvement. And I think it'll take a few times. I mean, it's a big change. And I've heard you loving the Control Center, as I do, too, now that I've kind of got the hang of switching and going to the Control Center and all that. So there's some learning stuff.

But the other problem I have, and this is not something they can fix, is it's really slower on older hardware. And so, I mean, again, this is sort of a natural thing that's happening is they're wanting to do more aggressive, deeper features that are inherently synchronized with the ever more powerful chips that they're releasing in their newer hardware.

The problem is they're updating older hardware with the newer software. And for a while I was misunderstanding its lack of response as it wasn't getting my screen clicks. And there is no feedback. It's not like it goes click when you tap the screen. The assumption is its response will be immediate, so you get visual confirmation that you pressed the right thing. And so what I've learned on my older pads that are all now, I mean, I moved to iOS 11 on everything, is you tap, and you just kind of be patient. You wait for a while, and then it goes plink and does whatever you ask it to. But, I mean, it's a noticeable delay. It's beautiful on the latest hardware; but, boy, you can really feel it on the older hardware.

So I don't think there's a solution for that. Apple continues to want to be on the absolute bleeding edge with their hardware. They're wanting to leverage the newer hardware. But they're also not wanting, for example, to continue to support iOS 9 or 10 in perpetuity. And I understand that. But, boy, the older hardware just doesn't have the horsepower to keep up with what the latest hardware is doing. You really do need the newer hardware. And of course Apple wants you to buy that because they're in the hardware-selling business, as we know. So anyway, just my few cents' worth on iOS 11. I love what they've done, but you really - and I can see myself maybe moving to a newer phone, not because there's anything wrong with my 6S whatever it is, or 6 Plus.

Leo: Just the speed, yeah.

Steve: It's just getting older, yeah.

Leo: I mean, you look at the benchmarks for the new processor, the A11...

Steve: The 11, yes.

Leo: And it's through the roof. It's almost MacBook Pro speed.

Steve: Yup.

Leo: And there's no real need for that. But, yeah, you have to figure that, once they've got it, people will find ways to use it. We've seen that, remember, with Windows; and we've seen that for years.

Steve: Yes, yes. And I actually think that's one of the things that happened where we were having the audio problems was that we moved - we thought that maybe the old USB interface I was using was getting just a little glitchy. It was 12 years old. I hadn't changed it through the entire life of the podcast. So we switched to a newer, more aggressive interface, and that older machine didn't have the horsepower to drive it.

Leo: Interesting. That makes sense, yeah.

Steve: So, yeah.

Leo: That's the way of the world, alas.

Steve: Yeah, it is. And you know me, I stick with what I've got until I'm finally driven off of it.

Leo: Get off of it. You've got to get off of it.

Steve: I'm now feeling like, well, okay. And I'm sure that the iPhone X is going to be gorgeous. So I will have skipped the 6S and the 7 and the 8 and probably...

Leo: Oh, you're going to see a big difference. Oh, my gosh. You're on a 6?

Steve: Yeah.

Leo: Oh, my. Oh, my.

Steve: And I'm waiting. I click, and I touch and wait.

Leo: Oh, my, yeah.

Steve: Yeah. So Apple published an interesting security guide for Face ID that didn't have a ton of new information in it, but it was interesting. We know that it employs what they call their "TrueDepth camera system." But one of the things that I appreciated that I want to share with our listeners, because I read the whole thing carefully, is that it demonstrates a truly gratifying focus, if you'll pardon the pun, on security, where things they did demonstrated that, I mean, it was above and beyond.

For example, we know that they project an infrared dot grid onto the user's face that an IR camera sees. And they use that essentially in order to put little spots all over the person, like chickenpox spots, in order to use that as part of their recognition. Well, for example, the sequence in which the dots are drawn is device-specific on purpose so that - because what they're doing is they're actually scanning with a dot sequence, and so the camera picks up where each dot is in three space; and the sequence is device specific so that you cannot capture the data from one device and then somehow arrange to play it into another device in order to spoof it. It won't get fooled because the sequence will be wrong. So they use a per-device pseudorandom sequence. I mean, and just that demonstrates, okay, somebody really spent some time to think about how to make this as robust against hacking as possible.

So, and other information, just to sort of assure people, is that it is explicitly local authentication, meaning that the facial recognition stuff never leaves that device. It has this crazy, as we were just talking, this A11 processor with a hardware assist neural network, what they call their "neural engine." And so what do they call it, a "bio neural chip" or something. The reason the bio-ness is in there is thanks to the fact that it uses a neural network as part of the processing of this image data. But, for example, it's been controversial that devices like Amazon's Echo, after you do the "hello" word in order to activate it, then it streams your voice to the cloud for recognition.

And so the point is that the phone has so much horsepower, so much processing power, the iPhone X, that it's all done locally. So none of your facial images, none of that that the camera sees during this process, ever leaves the camera. It stays there, and all the processing is done locally.

And then the other thing that they have done, and the way they described it I wanted to quote this because, again, it demonstrates a real thoughtful set of tradeoffs. They said: "To use Face ID, you must set up iPhone X so that a passcode is required to unlock it. When Face ID detects and matches your face, iPhone X unlocks without asking for the device passcode. Face ID makes using a longer, more complex passcode far more practical because you don't need to enter it as frequently. Face ID doesn't replace your passcode, but provides easy access to iPhone X within thoughtful boundaries and time constraints. This is important," they write, "because a strong passcode forms the foundation of your iOS device's cryptographic protection."

And what that put me in mind of was that I had made exactly the same set of tradeoffs in the SQRL client. That is, you are encouraged to create a strong passphrase to authenticate yourself to SQRL. But once you've done so, you are then, within certain conditions, allowed to use a much shorter, in fact, you just use the first N characters of your much stronger passcode to reauthenticate yourself once you have first authenticated yourself. And Apple has done exactly the same thing. For example, we know, we're all familiar with how they handle Touch ID and how you must, if you haven't used a Touch ID-based device for two days, you must then reenter your full passphrase.

Well, that has a couple consequences. One is it's good for security. It also keeps you from permanently forgetting what your longer passphrase is, if you're not constantly using your device. So they've done the same sort of thing, even on a device which is Face ID unlocked, so that the passcode is still required in a set of different circumstances: after it's just been turned on or restarted; again if it hasn't been used for two days, for 48 hours; if the passcode hasn't been used to unlock the device in the last six and a half days, so just shy of one week (for some reason they chose 156 hours); and Face ID has not unlocked the device in the last four hours.

So again, a set of interesting sort of heuristics that they settled on in order to create sort of a flexible security boundary based on how active you are, but also to continue refreshing your use of your face and your weekly use of your original passcode because that's the ultimate unlock. And then they have more accessible, shorter term unlocks as long as you do it often enough. And I get that because it's the same sort of thing that I have just recently done in order to reach a compromise between having a passcode which is strong enough to be resistant against cracking, yet offers a tradeoff, much as Face ID does, to still allow you to use your device frequently and conveniently every time you pick it up.

And as for this whole one in a million that they're now claiming for facial recognition versus one in 50,000 for Touch ID, it's like, well, I think we'll have to see how that goes. I'm sure people will be experimenting with trying to hack it, much as we saw with Touch ID, where they were immediately using gummi bears and things in order to lift prints and create fake thumbs and so forth. So I think we'll have to sort of see how this plays out. But, boy, I couldn't be happier with the clear attention that Apple has paid to getting this right, right out of the gate. And of course we haven't seen it yet. It'll be some time before we do, I guess.

But anyway, very, very impressed. And that grid is 30,000 infrared dots which are used - not just 100, 30,000 - in order to form what they call a "depth map" in order to differentiate your actual physical face from a 2D infrared image. And they also do pupil tracking dynamically to determine that you are looking at the camera, so you can't be looking away or have your eyes closed and have this function. So that's how the kids are prevented from holding Mommy's camera up in front of her while she's sleeping and unlocking the phone. Nope, Mom's eyes have to be open, and they have to be pointed at the phone, so pupil recognition is used in order to verify that your eyes are focused on the device itself. So again, I think they did everything that they reasonably could. We'll have to just see over time how this evolves into the actual user experience.

So in the last couple weeks there has been a number of articles about "malicious" ads doing cryptocurrency mining on users' browsers unwittingly, that is, unknown to the user, when they were visiting a site. And then most recently Showtime, several Showtime domains were found to be also covertly mining cryptocurrency. This wasn't Showtime trying to augment their revenues, this was some bad guys got into the Showtime server and added crypto mining technology to what was produced when a user went to the Showtime properties and downloaded a Showtime page.

So, okay. So I just sort of wanted to address the whole idea of this a little more broadly. So first of all, what we remember from our initial discussion of bitcoin mining is that a variable, we could call it "variable hardness," but it's really a variable probability problem is being solved by a cryptocurrency miner. That is, the idea is a random number is being chosen, and over time the probability of guessing correctly is decreased, making it harder and harder to guess the magic number.

And thus miners that have gone from just CPUs to GPUs to FPGAs to custom hardware, they've gotten faster at guessing, the idea being that the probability of guessing correctly is vanishingly small now, so the more times you're able to guess per second, the greater the chance that this diminishing percentage of chance will be correct. And if you guess correctly, you win a coin or a fraction of a coin, and that's changing over time, too. Everybody remembers how, when I first did the podcast on bitcoin, I started up a miner - in fact it was on that machine, Leo, the machine that's now too slow even to run audio for Skype sufficiently well. I woke up the next morning and there were 50 bitcoins sitting there. It was like, oh.

Leo: But that was kind of luck; right? It wasn't that it was working so hard.

Steve: Precisely. Precisely. I didn't expect it.

Leo: It was like a jackpot.

Steve: And it's luck for everybody, and I'm glad you said that because that's exactly what's happening here. So the idea is that bad guys are putting cryptocurrency-mining JavaScript into users' browsers, into ads or stuffing them onto websites that are hackable, like apparently Showtime's was. And they're getting all the visitors of Showtime or all the people who display this hijacked ad to do a little crypto mining for them. And the chances are, I mean, first of all, a browser is a bad miner. I mean, it's like so far down the curve you can't even find it. You don't want to be doing cryptocurrency mining in your browser.

Leo: No, you want custom ASICs and massive GPUs and SLI, yeah.

Steve: Precisely. But it costs them nothing to...

Leo: Well, that's the other way to do it. Go massively parallel; right?

Steve: Exactly. Exactly. And so here's the other problem, and I realized as I was putting this together that I haven't checked in with Mark Thompson...

Leo: I was going to ask, yeah.

Steve: ...who's my good friend and fellow massive miner, because as we know there's been a huge growth in the value, for example, of bitcoin recently. And last time we talked, as I have said on this podcast, you could not mine in California because our power costs more than the electricity required to earn/mine bitcoins. But in Phoenix, for whatever reason, power is cheap. And so it made sense for him to be mining in Arizona, but you can't do it in California. But you can do it under Niagara Falls where, again, power is very inexpensive. And so I don't know how that has changed as bitcoin, for example, has gone crazy in value to the point now where those 50 bitcoins, I really do have to find them because they're now approaching a quarter million dollars' worth of value. Of course, I've...

Leo: What?

Steve: Yeah, it is, believe it or not. It's like $4,300 now for a bitcoin.

Leo: But you don't want to be the guy who had 50 bitcoins and traded in for a quarter of a million when in five years it's worth 100 million.

Steve: That's the problem, too.

Leo: It's like any stock. You've got to know when to sell it.

Steve: But I'm also a Boy Scout. How does that go? Is that the Boy Scout salute? So I will be reporting my bitcoins to the IRS because I'm not going to screw around with that. They're now getting very annoyed with people who are cashing in their bitcoins and not reporting them as income.

Leo: Is it regular income? I guess it is.

Steve: Yeah.

Leo: It's like winning the lottery.

Steve: Exactly. So, yeah. Very much so. So, yes. I'm not in any hurry to do anything with them. But it would be nice to know where they are.

Leo: You should find them. I know I have my wallet backed up somewhere, and I can't remember the password. I only have five, though. If I had 50, I might make more of a concerted effort.

Steve: Yeah, it's a little motivation.

Leo: Yeah.

Steve: So anyway, what I wanted to say to people is that - so what's annoying is that somebody is using your CPU. On the other hand, okay, so are you.

Leo: So is every ad. So is every - yeah.

Steve: Exact - yes. And the autoplay videos, those are pegging people's CPUs just as much as cryptocurrency mining is. So our browsers...

Leo: How much time do you spend on the Showtime site, anyway?

Steve: Precisely.

Leo: Is it the watching a movie Showtime site, or just the logging into Showtime site?

Steve: It was several different domains, so probably going...

Leo: Showtime Anytime or...

Steve: Yeah.

Leo: See, if it was...

Steve: Well, it's probably when you're looking at something in your browser, and they're having a chance to run JavaScript. So anyway, so it's an annoyance. I guess it's not surprising. So I guess the point is that our browsers are hardened against JavaScript being able to hurt us, like as in malware. But they're not hardened against JavaScript being able to do work because, I mean, that's what it's for is doing work. And so they're like, well, okay. Maybe this user is at a website that's going to pay them if their browser scores a coin. No, but it could be. So anyway, that's what's going on with that. It's like, yeah, okay. So, yeah, it's using some of your power and some of your processor. But so are you when you're doing anything. So no biggie, really.

We covered extensively over the years the San Bernardino terrorist attack where Syed Farook had his two phones, and of course Apple famously refused to help the FBI to unlock this phone, which apparently...

Leo: I wonder if they would still have that attitude today. Seriously.

Steve: Really.

Leo: What if the Vegas shooter had a phone?

Steve: Oh.

Leo: Seriously.

Steve: Good point, yeah.

Leo: It would be politically very difficult for them to say no now.

Steve: Really, yes. Yeah, that's a good point. What happened in the wake of this is that the AP (the Associated Press), USA Today, and Vice Media all sued under the Freedom of Information Act (FOIA) in the U.S. for disclosure by the FBI of who it was they worked with and how much they paid because those are taxpayer dollars. And the argument was, hey, this is a taxpayer-funded operation. We want to know.

Now, James Comey, as we'll remember, the former FBI director did indirectly disclose that they paid something around $1.3 million for this hacking tool from an undisclosed company. So we really don't know who it was. We never got an exact amount. But these press companies, these press agencies wanted to know.

On Saturday of last weekend, a U.S. district judge in D.C., the District of Columbia in the U.S., Tanya Chutkan, disagreed with the suit that was brought by these companies and said, no, the FBI has no obligation, even under FOIA, to make this disclosure. She said in her decision that, "It is logical and plausible that the vendor may be less" - the vendor meaning the one that the FBI paid - "that the vendor, undisclosed vendor may be less capable than the FBI of protecting its proprietary information in the face of a cyberattack." Therefore, an argument for allowing this vendor, who received taxpayer money, to remain anonymous.

She also wrote: "The FBI's conclusion that releasing the name of the vendor to the general public could put the vendor's systems and therefore crucial information about the technology at risk of incursion is reasonable, so they might be subjected to an attack." And then regarding the cost of the hacking tool, Tanya agreed with the U.S. government that revealing the price the government paid for unlocking the iPhone could harm national security, saying: "Releasing the purchase price would designate a finite value for the technology and help adversaries determine whether the FBI can broadly utilize the technology to access their encrypted devices."

Okay. So translation is the FBI asked to keep this information private, and it got its way. So we're not going to find out who they got it from and what they paid, and that's sort of closed. And remember also that it's not at all clear that even then, if it weren't an iPhone 5c, that it would have been possible to make this happen. What I remember from the time was that it was the fact that it was a 5c that fit together with maybe some known exploits against that particular make and model that allowed a third party to get in and to have a big payday for themselves because this was, as you said, it was at the time very politically sensitive. And you raised a good question, I think, Leo, whether in the wake of the Las Vegas shooting, if a similar thing happened, whether Apple would be able to say no. Or would.

Leo: Be hard for them; right? Be hard.

Steve: Yeah.

Leo: Be difficult.

Steve: Yeah.

Leo: By the way, this just in, and you'll love this, the IRS has just awarded a $7.25 million fraud prevention contract to Equifax so that Equifax will let them know if any refund requests come from compromised Equifax information.

Steve: Boy, talk about making money coming and going.

Leo: But we knew they'd make money on this breach because they were pushing people towards a for-one-year-free identity theft monitoring service, when of course in another year you'd be paying for it. It's just, god, this is [crosstalk].

Steve: I know. So I know that everyone is, like, fed up with hearing about Equifax.

Leo: Oy.

Steve: But while you were on vacation, Leo, I attended a private security conference.

Leo: I heard you say that, yeah.

Steve: Where there was some inside scuttlebutt suggesting that this was looking more and more like a state-sponsored attack.

Leo: You thought it was China.

Steve: Yes, and that seems to be the consensus. That's beginning to surface. And the good news - and there's sort of a mixed blessing that it suggests that, if it's a major actor, they really don't care.

Leo: It's not about credit card for them, credit card fraud, yeah.

Steve: Yes. They don't care about the 143 million of us. What they do care about are specific people who are high-profile targets. And Bloomberg had some fabulous coverage. I'm not going to go through the whole thing because it's long, but the link is in the show notes for anyone who wants more. But I want to give our listeners a sense for just four paragraphs from their coverage.

They write: "Nike Zheng, a Chinese cybersecurity researcher from" - and this establishes sort of the history and the background, which wasn't clear before - "from a bustling industrial center near Shanghai, probably knew little about Equifax," writes Bloomberg, "or the value of the data pulsing through its servers when he exposed a flaw in popular backend software for web applications called Apache Struts." Which of course we talked about weeks ago. That's the Java-based server-side platform for developing web applications that Equifax, among many other people, are using.

They write: "Information he provided to Apache, which published it along with a fix on March 6 [2017] showed how the flaw could be used to steal data from any company using the software. The average American," Bloomberg writes, "had no reason to notice Apache's post, but it caught the attention of the global hacking community. Within 24 hours, the information was posted to FreeBuf.com, a Chinese security website, and showed up the same day in Metasploit," which as we know is a popular free hacking tool. "On March 10th" - so four days after the disclosure - "hackers scanned the Internet for computer systems vulnerable to the attack and got a hit on an Equifax server in Atlanta, according to people familiar with the investigation." So four days after the disclosure, Apache did everything they could responsibly. They said, "Whoops, we have a problem. Here's the fix." And four days later Internet was scanned; Equifax was identified as a target.

"Before long," Bloomberg writes, "hackers had penetrated Equifax. They may not have immediately grasped the value of their discovery; but, as the attack escalated over the following months, that first group" - get this - "known as an 'entry crew' handed off to a more sophisticated team of hackers. They homed in on a bounty of staggering scale: the financial data - Social Security numbers, birth dates, addresses and more - of at least 143 million Americans. By the time they were done" - and here's news - "the attackers had accessed dozens of sensitive databases and created more than 30 separate entry points into Equifax's computer systems." Meaning that this initial Apache Struts was just a means of gaining a foothold. And once they did, they established 30 other presences that would protect them in case of the discovery.

So Bloomberg writes: "The hackers were finally discovered on July 29th, but were so deeply embedded that the company was forced to take a consumer complaint portal offline for 11 days" - because it was so massively compromised - "while the security team found and closed the backdoors the intruders had set up." And then, finally: "The handoff to more sophisticated hackers is among the evidence that led some investigators inside Equifax to suspect a nation-state was behind the hack." That is, the people who discovered it weren't the people who continued exploiting. Rather, they found it, and then they basically passed it upstairs to a more sophisticated next-level group, who then crawled inside and began to exploit.

They write: "Many of the tools used were Chinese, and these people say the Equifax breach has the hallmarks of similar intrusions in recent years at the giant health insurer Anthem" - which of course is, remember, 18 months ago is the reason we first told everybody to lock down their credit reports at the time - "and the U.S. Office of Personnel Management. Both were ultimately attributed to hackers working for Chinese intelligence."

So this is the way these things are now being done. And, I mean, it's not good that 143 million Americans and some Canadian and U.K. citizens had this detailed personal data exfiltrated from Equifax. But as you said, Leo, it suggests that we're not all in immediate threat. It's not as if some hacker was looking, some money-oriented, financially oriented hacker would be selling this in bulk in order to immediately turn it into profit. It seems much more likely that, if this is in fact China, among this 143 million are a lot of, to them, very valuable information, which they could then use for targeting specific individuals. So not good for those targeted people, but somewhat better for the rest of us, we can hope.

Leo: Geez.

Steve: Yeah. We've talked over the course of the last month or two about the Broadcom firmware problem that essentially beset all of our mobile devices. Both Android and iOS devices were vulnerable. Google and their Pixel devices were first to patch, and patched very quickly. I'm not sure, I don't remember now where Samsung was on this. But we covered several, a series of patches that iOS released and finished with 10.3.3 was the most recent before jumping to 11.0.1. This comes back to our attention because a Google researcher has published a proof-of-concept exploit that is functional against Apple iPhone WiFi Broadcom chips prior to iOS v10.3.3. So the good news is you don't have to update to 11 to be protected against this.

Leo: Is that because Apple discovered it and fixed it? Or just in the process of updating it fixed some issues?

Steve: No. They did discover it.

Leo: They knew about it, okay.

Steve: Or it was reported and fixed.

Leo: Got it.

Steve: So they did it in 10, in 10.3.3. Remember we had 10.3.2, and we thought - and there was some Broadcom fix there. And we presumed that that was this big one. But it turns out, no, it wasn't until 10.3.3, which was the final v10 iOS before jumping to 11. So for people who, for whatever reason, are holding off on moving to 11 - and despite all my complaints, like you, Leo, I really like 11. I'm liking the things that they did, although as I said it comes at a cost of performance on older hardware. But somebody who wants to stay at 10, as long as you're current, which puts you at 10.3.3, then this is fixed.

But the significant thing is that now with a public proof of concept, that means anybody who wants to attack a pre-10.3.3 iOS device has a template for doing so, and it is a potent attack. Remember that you don't need to be associated with the WiFi device. Your phone just has to be within range, that is, radio range. Your phone and the attacker's radio need to be able to reach each other, and so you don't have to log in or use a malicious access point. You just have to be able to be enumerated, essentially, by the Broadcom chip. And that's enough for your phone to be taken over.

So I imagine, I mean, the good news is Apple pushes these updates out, and so they're really good about keeping their phones' OS current and patching the firmware in the related devices, like this Broadcom chip. But this is something everybody wants to do because it does make the attacks much more likely now.

And I heard you talking on MacBreak Weekly, Leo, about just in general about this evolving awareness of how much Russia had their tentacles into the most recent U.S. presidential election. We've seen news in the press about Facebook taking real steps to understand how much Russia was purchasing ads on Facebook. Twitter has done the same thing, recognizing the extent to which they were unwitting accomplices in Russian involvement.

And now Google has said, although it's not official yet, and the Google personnel who have been talking to the press have requested anonymity because the plans are not yet officially public. But Google will be offering what they call the APP for Gmail, the Advanced Protection Program. And the idea is that it is going to be a super secure service that will be an optional feature for Gmail, targeted at corporate executives, politicians, and others with heightened security concerns.

And this is probably to keep people from leaving Gmail. Google would like to not have people think, oh, well, it's free, and so it's not very secure. If we really want secure email, we've got to go somewhere else. Instead, Google is saying no, we're going to require two separate physical security keys of some sort. Right now, as we know, they do offer a second-factor technology, for example using Yubico's technology and one of the variants of FIDO as an option. They're going to, as part of this Advanced Protection Program, require two different physical security keys; and no third-party apps will be able to gain access to your Gmail service.

So your freedom and your flexibility of what you can do will be dramatically curtailed. But for people who really want security, that's a requirement. You have to silo mail and not allow apps to be able to get access to your account, and then they're going to really crank up the authentication level to create this level of security. Again, no idea when. It's not official yet. And if it happens, apparently it's going to be called Advanced Protection Program. So again, in the wake of what we learned, Google is saying, okay, we're going to provide as much, arguably more security than anybody else, and a lot for a cloud-based service. And I don't know whether it'll be free, or whether it'll be...

Leo: I think it's for the Google Apps owners. It's for G Suite. I don't think it's really for us.

Steve: Ah, okay. That does make sense, yes.

Leo: I think it's for business folk, yeah.

Steve: It does make sense. There was a bunch of mistaken news, but clickbait headlines, talking about how DKIM had been broken. That's the domain key technology that we've been talking about recently, the idea being that a mail domain - I'll use mine, GRC.com - can publish through DNS its public key. And as it's sending email from its SMTP server, it signs the outgoing mail. That is, it adds a signature using its secret private key to every outbound mail. And then the receiving SMTP server is able to similarly use DNS to get the matching public key and then verify the signature on email claiming to be from GRC.com in order to verify it. This was designed as an antispoofing measure, that is, the idea being that no one could say that email was from Apple.com or Microsoft.com or any major presumably trusted sender. The recipient would be able to verify.

And as we said when we were talking about this a few weeks ago, there are some email clients or browser add-ons or email client add-ons that will independently check the DKIM signature. Normally that's done at the SMTP server level, where incoming mail will be rejected if the signature doesn't match. So what has been claimed in the news recently is that it's been broken. Well, it has not been broken. The crypto is strong and was done right.

What was broken is that email itself has always had a problem sort of with its own security. We have this notion of multipart extensions and the concept of a mail envelope, that being the thing, sort of an abstraction of the email content and the headers forms an envelope. Well, that's what the sender's private key is signing. Well, it turns out that it is possible to spoof the signature, that is, to spoof the contents of the envelope so that it's possible to add your own different message, keep the real message from being seen, yet still have the DKIM signature shown as valid despite the fact that what the user is seeing is not the contents of the envelope that was securely signed, but something that a spoofer attached to the email.

So the upshot is that the assurance that you believe you're getting from DKIM is bypassed so that, if your email reader is telling you that it's checking the email signature, it could show it being valid, but the contents of what it shows you is not what was signed in the envelope, so it's a spoof attack. And this is one of the things we're seeing is sort of like, as we keep getting better and better with security, we keep closing loopholes and closing backdoors and getting better, the whole problem of spoofing remains elusive because our clients are still, in this case, still not strong enough to be hardened against that because they're trying to be flexible enough to give us the features that we want in email which were extended over time without thinking about security as closely as they should have been.

So anyway, I got a bunch of tweets from people saying, oh, my goodness, DKIM is broken. Well, no, but it is spoofable, unfortunately. And over time I think we'll see that get hardened up. But for the moment, it is the case that someone could send you something where what you see is not what was signed, and therefore it could say anything and cause you to be misled. For example, it could contain a malicious link where you trust the link because you're looking up at your DKIM monitor, and it's saying green. It's saying, yes, this is trusted email. And so you go, oh, fine, and then you click the link, and you could get compromised as a consequence. So, I mean, it's not good, but it is not the case that the DKIM system itself was broken. It was basically just bypassed. Someone ran a bypass.

And, finally, there was another Google researcher sort of got around to taking a look at a very popular small suite of services. It's known as DNSMASQ, D-N-S-M-A-S-Q, DNSMASQ. It's very popular in small networks, in small routers, anything where you have a small footprint server where you want to provide local DNS and DHCP services. So, for example, I mean, it's what we probably have in many of our home WiFi routers or standalone routers. There are lightweight services to provide local DNS caching and lookup and dynamic host configuration protocol, DHCP, which is the obtain IP address automatically service. It's widely used for tethering smartphones and portable hotspots. Many virtual networking virtualization frameworks use DNSMASQ as their virtual machine DNS and DHCP systems. It's supported under Linux, Android, the various BSDs, and Mac OS X. It's included in most Linux distributions, and there are ports for it for FreeBSD, OpenBSD, and NetBSD. So it is widespread and popular. Ubuntu uses it, for example, by default for their services.

So what was found was, I think it was seven or eight remote code execution vulnerabilities, but remote in the sense of remote from the device, not in the sense of Internet facing. These are all local LAN-side attacks so less to be concerned about. It doesn't mean that suddenly there's a new vulnerability for all of our routers that we need to worry about. These are, because these services are LAN-facing for clients on a LAN, clients of a router on a LAN typically, it means that, if you had a malicious device, maybe a malicious IoT device that had been updated with some malicious firmware, it could arrange to execute code on your router. Well, that's not good. That's certainly not good. But it's less bad than if anybody anywhere in the world were able to scan for this vulnerability, which is probably very widespread.

So our takeaway - oh, and this is just yesterday that this came to light. Our takeaway is that, to be looking for firmware updates for our various devices, hopefully people will start pointing fingers at specific vendors. This is just - because this is less than 24 hours old, we don't yet know model numbers and makes and firmware versions and so forth that are vulnerable. I think probably we'll be talking about this next week as more information comes to light.

It'd be fun also to find something that can identify whether the routers that we're using are vulnerable. Much as malicious clients could exploit them, we may be able to do that. But they are remote code execution, so what I think we'll need to find is be looking for updates to firmware for the devices we use. And in this case it would be a good thing to apply them because you don't want a single malicious device to be able to take over your router and then potentially have access to both your internal network and open it to the outside world.

And Leo, I heard you talking about the EFI firmware vulnerabilities.

Leo: Yeah. I want to get some clarification from you on that one.

Steve: Yes. So a tech firm has taken a look at the pervasive problem throughout the entire industry. The problem is not just Mac-specific. It's Duo Labs. And these guys took a look at a whole bunch of Mac machines. It was, like, more than - it was tens of thousands of Mac machines. I want to say 43,000.

Leo: Yeah, I think it was something like that.

Steve: Something like that, yeah. And they found, like, an overwhelming number of specific Mac models were vulnerable, meaning that they were running EFI code today with known vulnerabilities. Now, okay. That's not immediately a problem because an EFI vulnerability, first of all, is very difficult to exploit. It's more someone gets physical access to your machine and tinkers with your system in order to get something installed in EFI. So it's difficult to exploit, though very powerful if exploited, but difficult for the average user to deal with because it's sort of - it's preboot.

But what it means is that AV software can't really see malware there because it's underneath the OS. And what it enables, what EFI exploits, if they're exploited, enable are virtually undetectable super rootkit-style preboot attacks where power being turned on, before the system boots, EFI is running around setting things up, getting the hardware going, and has full access to the system. It's what then chooses which drive to boot, finds the bootable partition, and loads it into memory and gives it control. So if that's been made evil, that's really bad.

So the problem is that, first of all, there's maybe a lack of focus and attention on this. It's also the case, as you mentioned during MacBreak Weekly, Leo, that attempts to upgrade firmware sometimes fail silently. And once it fails, like it doesn't pass the required checks, there may not be room in available flash memory to install the firmware because a lot of other drivers have been added over time, I mean, there are just many different failure modes.

And so it's sort of like, I mean, I have the experience because I have a small iPad where it complains, it was complaining for a while that I couldn't update iOS because there wasn't enough room. Now Apple has solved that by allowing some apps to be removed temporarily to make space for an update, and then they're brought back in, which is kind of a clever workaround. We don't have that capability for EFI. So as a consequence, what these guys found, what Duo Labs found was that there is a widespread lack of current firmware in Macs, but not only Macs. All they looked at was Macs.

But I would argue that, because the general PC market, whether it's running Windows or Linux, has a greater disconnection between the OS and the motherboard. For example, Microsoft doesn't care about EFI because they're providing Windows. Apple cares about EFI because they're selling the hardware and all the software that goes on top. So it's really up to your motherboard manufacturer to deal with EFI, and motherboard manufacturers don't really care that much. I mean, they may have updates, but then it's up to the user to go get them.

So we're sort of in a mess with this, which is why it's good that the vulnerabilities, if they exist, are difficult to exploit, and unfortunate that it's difficult for average users to deal with. The best takeaway I have is that, for the Mac's EFI firmware, there is a free tool that you can use to check what's known about your EFI underneath your Mac. It's called EFIgy as a cute little play on EFI. It's E-F-I-G-Y dot I-O. EFIgy, E-F-I-G-Y dot I-O. That's a shortcut that takes you to a GitHub page where these guys are using an EFIgy API in order to tell users about the state of their firmware. This is still under development, and what we need now is a mature database to connect Mac makes and models to this firmware they should have in order to compare to the firmware they do have. This thing will show you what you do have. It provides a means for gaining some insight into that. And I think over time we'll be seeing this mature.

Unfortunately, this is Mac-specific - well, unfortunate for PC users, non-Mac users. It's Mac-specific. All you can really do is take the effort to check in with your motherboard manufacturer and almost, I mean, if you haven't checked for a couple years, and really who does because it's not automatic, check to see. Normally your boot screen will show you what version of EFI you've got while your system is booting. Go to your motherboard manufacturer, see if there's new firmware. The old wisdom I would argue, and this was the wisdom for motherboard firmware, if it's not broken, leave it alone.

Well, unfortunately, all EFI firmware is probably broken. It's working, but it's vulnerable now. So we've moved from a world of, if it's not broke, don't fix it, to a, if it's old, it's probably broke, and you don't know it, so it's worth patching it with the latest version. So I think the advice now needs to be specifically - because, as we know, attacks never get weaker, they only get stronger. So it's worth knowing that your motherboard is running the most recent EFI firmware that your motherboard's manufacturer makes. And the good news is...

Leo: You know, the interesting thing on the Mac, I don't think there's any way to manually force an EFI update.

Steve: No.

Leo: It's not like a PC motherboard where you can go to the manufacturer and download the firmware.

Steve: Yes. And that's the perfect segue because I was just going to say that the good news is Apple has responded to the press coverage, and they've said - they're kind of dodging it at the moment because they don't have an answer immediately. But this has shined a bright light on this, and Apple can and probably will take responsibility. So arguably Mac users have a little bit of a spotlight on them at the moment, but Apple can step up and give this better attention. And then you're glad you're a Mac user because Apple can take care of it for you. You'll just get an update, and you won't know why, but things will spin around on the screen for a while and then settle down, and you'll be safe again. So that's good.

Leo: Yeah. They've put that in High Sierra, too. So if you have High Sierra, or you don't have High Sierra, might be another reason to upgrade to the latest version of macOS because there's an EFI checker. They check weekly now.

Steve: Just a real quick little mention of an IE bug. I'm not, you know, IE is sort of, well, I guess those of us who are not yet on 10 are still using IE on everything before - unless we're using Chrome or Firefox or Opera or anything else. It came to light that there is a bug in Internet Explorer's handling of the URL bar. And it's kind of cool and subtle. And again, it's one of those things that everyone is going to jump on until it gets fixed. JavaScript running on any ad or page you're visiting can intercept when you hit the Enter key after entering a new search term or URL as you're leaving that page. Whoops.

So it means you're somewhere, and you go, okay, you're done being there. So you type something new into the URL field, either as a URL to go to a website or as a search term. When you hit Enter, before leaving where you are, script on that page can capture what you typed and send it back to the site you were visiting or any of the ad suppliers, which is not behavior that you want.

Only IE does this. It's a bug. It'll get fixed. I don't know how soon, probably not by next Tuesday. That's going to be the second Tuesday of the month since today is the first Tuesday of the month. So it seems like that doesn't give Microsoft much time. Maybe they can do it in time. It would be nice. But again, it only affects IE users. But it's the kind of thing that might provide some information the bad guys would want. And if you're using IE, they can get it until Microsoft gets that patched.

I had one little bit of miscellany. Leo, while you were away, two sci-fi shows premiered.

Leo: And I haven't seen either.

Steve: "The Orville" and...

Leo: And "Discovery," yeah.

Steve: ..."Star Trek: Discovery." I was negative about "Discovery" with Father Robert because I'd only seen the first five minutes of it, and it was so painful that I just said, okay, let's switch back to whatever...

Leo: It got a lot of bad reviews, though. It wasn't just you.

Steve: Yeah, I know. And I'm not surprised. I did sit down and watch the whole thing again. I gave it a try. And then I'm remembering how bad the first few episodes of "The Next Generation" were.

Leo: Oh, okay.

Steve: Even with Patrick Stewart and Frakes and the whole, you know, the core crew, it was bad for the first, oh, maybe half a season. They were, like, shouting at each other across the bridge, and Picard was being really annoying. They found their groove, and it was arguably one of the better franchises that we had. So I want to give them the benefit of the doubt. And I also - my plan is to wait for the first season to be finished and then do my CBS all-access and watch, not only all of "Discovery," if I end up hearing that it's worthwhile, but also the "The Good Fight," which was the continuation of "The Good Wife" that was on CBS for so many seasons and so worthwhile. And Leo, I didn't make it past 10 minutes of "The Orville."

Leo: Oh, really? Oh, that's the only one I really want to see. But you have to be a fan. If you don't like his humor, Seth McFarland's humor...

Steve: Yes. And I recognize I'm very finicky about humor. I like "Seinfeld" and...

Leo: I love "Family Guy," so I think I would like "Orville."

Steve: Give it a try because it certainly is sci-fi setting for that. But it wasn't my kind of humor.

Leo: He's an acquired taste, to say the least.

Steve: Yeah. I had a really fun story about SpinRite and Drobo from Steven Perry, who's a CISSP, or has his CISSP. He's in Charlotte, North Carolina. And the subject, it was dated email incoming September 28, so just last week, said: "SpinRite Saves a Drobo." And there's an interesting lesson here.

He said: "Hi, Steve. Have a SpinRite testimonial for you. You could title this one 'SpinRite Saves a Drobo.' I have a Drobo FS NAS device that is about five years old now. When I set up the Drobo, I made sure that each of the five 2TB WD [Western Digital] Black drives passed SpinRite before they were installed in the Drobo. This past Sunday the first drive bit the dust." Okay, now, remember, the Drobo is a RAID, so that's okay. You can lose a drive, and it'll say, whoops, we got a problem here, but we can still read all your data.

So he says: "I swapped out my spare 2TB drive" - so he had another one standing by, as he should - "that had also passed SpinRite, and during the rebuild process one of the other drives..."

Leo: Oh.

Steve: Yup, "went offline, and Drobo was setting off alerts that too many drives had failed or been removed." That is, the drive just completely died. He says: "I was able to safely shut down the Drobo, remove the drive it thought was missing, run it through a Level 2 pass from SpinRite," he says, "which took 11 hours to complete. The drive was then returned to the Drobo, powered on to find the Drobo was now seeing the drive and was able to successfully rebuild the new drive, and all is good again."

So essentially, as we know, a RAID 5 allows you to lose one drive. Or another way to look at it is any given sector on one drive can be sort of recreated by using the other drives to calculate the lost data in a single sector of one drive. Or you could lose all the sectors of one drive, and then the other drives can be used to calculate the lost drive's contents. That gives you the redundancy of RAID 5. But you cannot lose two. And that's why now we're seeing increased use of RAID 6 because drives have gotten so huge that exactly what happened to Steven Perry is beginning to be seen, and that is the first drive dies, you swap in a good one, and during the rebuild process, where you temporarily have zero tolerance of any failure, another drive dies.

And that's why, for example, at GRC I'm running RAID 6. So I've got - it's expensive in terms of you don't get the data now of two drives. On the other hand, you get double redundancy. With RAID 6, I have a four-drive RAID 6, and any two can fail, and I'm still okay. And remember, because we've talked about the Drobo before, although again it's expensive, the Drobo does allow you to use extra redundancy. It'll take away a chunk of your storage; but you can say, no, I need extra, extra safety. Most people don't do that. They figure, as Steve did, hey, I've got RAID 5, and I've got SpinRite. As it happens, thanks to having SpinRite, he was able to recover from that window where there was zero tolerance. He could not afford to lose another drive, and he did. But SpinRite got it back. The Drobo was able to rebuild. Now he's back up to full redundancy.

Leo: And he should turn on the two-drive backup.

Steve: And maybe that would be good.

Leo: I do that on all my Drobos and my Synology, as well. If you have that as [crosstalk]...

Steve: Belt and suspenders. Belt and suspenders and SpinRite.

Leo: Although I suppose three drives could fail. But it gets diminishingly small. We had a funny - go ahead.

Steve: Yeah, go ahead.

Leo: Well, we had a funny thing happen because we had a drive that was a little flaky. We used a Western Digital Gold for rerun playback. And it was flaky. And they pulled it. They were going to get a new one. And then somebody, I think Russell, very sharp-eyed, noticed that the label on the Gold spelled "thailand" with a lowercase "t." It's a made in [lowercase] thailand. And he thought, hmm. We got them on Amazon, but from a reseller on Amazon. And he thought, hmm. So they open it up, and it is a counterfeit Western Digital drive. We bought it on Amazon, but from a third party, not from Amazon directly. And you'd have to look really closely with a magnifying glass to see "Product of thailand." But, wow. I mean, really. So somebody's taking crap drives and just slapping a sticker on them and selling them as enterprise-grade 2TB Western Digital drives.

Steve: Wow.

Leo: So be careful. You know, I've said this - by the way, this is the sticker they put on it so we wouldn't use this, says "Fake News." I mean, you would just assume, well, I got it on Amazon. But really you've got to be careful on Amazon because a lot of times you're not getting it from Amazon, you're getting it from a third party. And it means sometimes getting it on Amazon doesn't mean anything more than getting it on eBay.

Steve: Right.

Leo: And you wouldn't really buy hard drives on eBay, would you.

Steve: No.

Leo: No. It's not that it was made in Thailand CR1, it's that it's a lowercase "t" in "thailand." I had a watch for a long time, a Rolex that said "Made in Jeneva" with a "J." That was how I knew. But it was only $10 in New York. All right, Steve. I think it's time to get to the topic at hand.

Steve: Well, we've got a couple of closing-the-loop bits from our listeners.

Leo: Yes, sir.

Steve: So Mark Havas sent: "Regarding your printing 2FA codes to have them on paper, would you keep the images in a password manager like 1Password? Thanks in advance." And, okay. So it's interesting. I don't understand, and I'm not picking on Mark at all because many people, I mean, he's just asking, but everyone seems so resistant to the idea of printing their QR code for their authenticator. And I don't understand it. It may be that we're all hooked on electronics or automation or cloud or something. But so we just sort of keep seeming to circle around this. I get it that people want a shortcut. They want more ease of use. But offline means offline, not online. If it's offline, it cannot be hacked. If it's online, it can be. So I'm not saying it will be, but it can be.

So if you've got all your two-factor authentication QR codes in anywhere online, then Russia has a connection. They may not be able to use it, they may not be able to decrypt it, but online means online. And so I have no problem if people say, oh, well, I want to store them in my password manager. Okay. I'm not storing them in my password manager. While you were talking, Leo, I looked at this question. And I'm not facing them toward the camera, but here's my set of QR codes.

Leo: Where do you stick those, Steve? It worries me that they were that available. Are they just in a pile on your desk?

Steve: Yes, and Russia cannot get them.

Leo: Yeah, but maybe you can't either if you put some more stuff on your desk.

Steve: No, I know exactly where they are. They are offline.

Leo: Yeah. No, that's an excellent point.

Steve: And that's my point. And I'm a little worried about this because SQRL has some of the same imperatives. And it's like, no, write this down once, print this once, because then it's offline. It's like people don't get that anymore. It's like, oh, everything's supposed to be online. It's like, yes, Russia wants that. So okay.

Leo: And I use Authy, which means it's online. But you're absolutely right. I mean, I had a little scare this morning. I got a call from Hover. James, very nice guy from Hover said, "Your account's been compromised, and somebody has changed your email forwarding." I use Hover to forward my email address because they own, they register the domain name. And they added, somebody added another email address. They didn't take the old one out, thank goodness. But I guess the idea was to be surreptitious; right?

Steve: So they were cc'ing themselves on everything.

Leo: They were cc'ing all my email to them. And the idea was that they would then ask for password resets, and they'd get them. And I might notice, gee, there's a lot of password resets, but there always have been. I mean, people are always attacking me. But I have my email address, and they can't get them. So they found it. I guess they had monitoring. I know you were talking last week about monitoring, how important monitoring is. They must have been monitoring, and they discovered the hack. They reversed it and notified me immediately. Hover, God bless you.

Steve: Yup.

Leo: I went and looked, and there was one Twitter reset email an hour before he called. But I have two-factor turned on, so it didn't do them any good. They got my password, but they couldn't get my account because I had two-factor turned on. And it just really underscored for me, not only how important two-factor is but, to your point, how important it is that two-factor be really separate. So if you're compromised in some way - and I'll give you an example. They could have tried to get my Authy login; right?

Steve: Right.

Leo: And then, now, in this case it's fine because Authy uses Trust No One encryption. Only I know the passphrase that unencrypts my Authy database. So even had they, I'd be safe. But if they had - but that's an example. These things cascade. They get your email, and then they can use your email to get something else. And if they were able to figure out what your passwords and your two-factor would be, then there would be no point in having two-factor.

Steve: Well, yes. And as we've been saying, one of the most troublesome things is that email is the typical password recovery system. That is, again, this is one of the tradeoffs that SQRL uses is there's no one to complain to. There's no one to say, oh, I forgot my password. I've lost my identity. I mean, we make it really hard to get yourself in trouble, but not impossible. And there's just - if you want security, then you cannot trust a third party, or they could be subject to a subpoena. So it'll be interesting to see how this plays out. But I've not cut any corners. I've said, no, I'm going to offer the world a truly secure system. We'll see if that's too much for everyone to handle. And it's not like everyone has to. People who are security conscious can choose to do that, and everybody else can continue using their email. But the problem is that, how many times, "I forgot my password." What do they do? They email you a link. Well, if somebody had your email account...

Leo: That's exactly right. That's right.

Steve: ...and they then said they've lost their - they forgot your password, because they're bad people, then you would get a password recovery link which they would grab. They would click on it before you knew any of this was going on and take over your account.

Leo: Yeah. I just - I'm so glad I listen to this show and I have two-factor turned on everywhere.

Steve: Yeah.

Leo: I feel much more secure.

Steve: This is really a quickie. I just got a kick out of this. Jamie, whose handle on Twitter is @LinkLayer, he said: "My CS prof anticipated fuzzing in 1976." We talked last week extensively about fuzzing. And he said: "He said to test input handling with punch cards from the trash and swept up from the floor."

Leo: That's a good idea. Random data.

Steve: It may not be true, or maybe it was. Yes, exactly. Just grab punch cards from wherever you can and feed them into your program and make sure they don't crash it.

Michael Johnson asked, he said: "How about a segment on taking a laptop on a flight out of the USA, working, and returning? What's the best way to do this with no border hassles?" Well, okay. So there are two problems with taking your laptop somewhere. One is the privacy of having its contents somehow exposed. The second is possibly getting it infected and bringing an infection home. So privacy could be obtained using an add-on that we've talked about often, like VeraCrypt, which encrypts the whole hard drive.

But the other thing you could do would be to make a copy of the hard drive, if the drive is removable from your laptop. Clone it. And then leave the original drive home, which is the entire state of your laptop before your trip. Then encrypt the one you're traveling with for its safety, then stick it in the laptop and use it. Now you have the benefit of encryption, so only under your control can anyone see into it. And if, while it's decrypted, it were to get malware infected, then the idea is, when you bring it home, you just take it out of the laptop and put it aside.

And then, oh, you also want to, depending upon what work you do, put the work up in the cloud before you take the drive out so that you've moved your transient work somewhere safe. Yank out the drive, which is encrypted, which may not be convenient for you normally and might have unknown contents on it. Maybe some foreign government briefly got it at the border and installed some malware on it or something. So the point is you can no longer trust it.

Well, you don't have to. So you yank it out of your laptop. You put the drive that stayed home in safety into your laptop. So now you've restored your laptop to exactly the way it was before you ever flew anywhere, or before you traveled. Now, grab the work you did back from the cloud, and you're safe, having essentially removed any possibility, removing the inconvenience of having the drive encrypted if you don't normally want it to be, and any possibility that you brought more back with you than you intended. And, finally...

Leo: Of course, if they put an EFI virus on, you're screwed no matter what.

Steve: Yes.

Leo: In other words, don't let anybody have your laptop.

Steve: Really, if you can avoid it, you don't want to lose control, physical control of your laptop. And I did want to just add, if removing your drive is not possible, then you could use a tool like Image for Windows to take a full drive snapshot to an external drive before your trip. When you get back, and before you hook it to your network - do not hook it to your network - restore the full drive snapshot back to your laptop in order to wipe out anything that may have happened while you were traveling. So that's really the best, I think, you can do. You get encryption to keep anyone who might be curious from poking into the drive; and you're also protected if, while it's unlocked and decrypted, as it has to be in order to use it, from picking up anything malicious.

Many people, naturally, because of the all-time favorite podcast, "The Portable Dog Killer" episode, have been tweeting me and writing to me about the U.S. Embassy in Cuba, I guess it was in Havana, where all of the diplomats are suffering serious hearing loss. And of course they're saying, wow, does somebody there have the equivalent of the portable dog killer technology? And of course, well, no. It's certainly different. Remember that this was high-frequency, but audible to dogs and us, which is why Mr. Archibald heard it when I zapped him with it at the end of the day in the story. Those who don't know what we're talking about may want to look back into and find that episode [SN-248]. But anyway, I just wanted to acknowledge everybody's, I mean, I've gotten so much input from people saying, hey, Steve, looks like they had a version of the PDK. And it's like, well, whatever they're doing is unfortunate and certainly very malicious, to impair someone's hearing by who knows what they're doing. Scary.

James Parsons tweeted: "If SQRL requires a password re-entry every session, it encourages weak passwords, especially on mobile." He says, "Special chars are difficult on phones." Again, I'll just say, because I already did cover this, the long password is only needed once to tell the system you're you, which we have to have to make it hackproof. And then afterwards you do a per-authentication short version. And the good news is everyone will be able to play with this very soon.

And as for phones, in the case of Jeff Arthur, who is the author of the iOS client, you do need, again, to authenticate yourself once. But then that enables Touch ID, and no doubt he'll be supporting Face when it comes along. And so then you're able to just authenticate with your fingerprint, just as you would expect. So again, a set of tradeoffs exactly like what Apple has designed is the same thing we're doing for the sake of security and to make it not cumbersome.

I mentioned the payment API a couple weeks ago, and someone tweeted the question, does this mean all adopting online retailers can stop storing our credit cards? Breaches would be far less harmful. No, it doesn't mean that. All the payment API means, which will be coming to all of our browsers, is that rather than us having to redundantly fill out payment forms across the Internet, our browsers can be a secure repository of that information, can contain our billing address, the credit card number, the expiration date and so forth. And the API allows a website to request through an established protocol that information. It'll pop up a standard dialogue which we'll get used to.

So the standardization at the user side is very useful, will verify the stuff we're allowing our browser to send, and then just say yes. And that will go then to the remote site in a uniform format to essentially circumvent the whole form fill-out thing. So all it is, is a unification and sort of a standardization of our submission of repetitive similar information to make it secure and uniform to the user and smoother, to smooth the checkout process and the payment process for the website. So it's a good thing.

But what the server does, what the remote web server does once it gets the information is still out of our control. For that you need PayPal or some third-party payment processor if you want to keep the sites blinded to that information. But still a very nice thing.

So now let's talk about what Moxie Marlinspike and his gang who are developing Signal have done. Okay. Matthew Green, our friend and cryptographer at Johns Hopkins, said of what I'm going to discuss, he said: "Private contact discovery for Signal. Make no mistake, what Moxie is doing here is going to revolutionize messaging." I've got a link for all the details, which I'm not going to go into because it's lots, like pseudocode where they're talking about showing sample code to implement these different things. But here's what's going on.

As I said at the top of the show, we recognize that it's one thing to encrypt the communications, and a different thing to encrypt the metadata, meaning that I could have encrypted email which I send to somebody, but the fact that I'm sending it is not encryptable. The envelope that contains the mail has to have headers for the destination. And typically the sender, in order for it to go from point A to point B, it has to be addressed.

And this is true everywhere. We've talked about, for example, even how HTTPS, the standard encrypted protocol for the web, it encrypts the contents and encrypts some of the metadata, but not all. People still know what IP you're going to because your traffic is going there. And in some cases the first packet identifies the website that you're asking for, which is how the server knows which certificate to match for your connection. So there's still - there's always still some metadata leakage.

That wasn't good enough for Moxie and his gang. So they said, okay, we need to solve this problem of a new Signal user wanting to find out which people in their own contact list on their phone, for example, are Signal users. How do we do that in a way that leaks nothing, even to the server that is providing the intercommunication? Because, remember, this is all end-to-end encryption, meaning that they've designed a system so that the server is blind to the contents. They also wanted to blind it to this aspect, this social graph metadata.

So one thing they could do, for example, is to hash the phone numbers of everybody in your contacts list. And so what you would be doing is you would be submitting a list of hashes of all the phone numbers. Send them up to the central server, and it then looks through your hashes for the hashes of all the other subscribers of this system for any matches. And if the hashes match, then we know that the phone numbers match. And so that says that you have the phone number in your contacts list that matches another phone number of a subscriber; and so, yay, they're a Signal user, and you could connect to them.

The problem with this is that there isn't enough entropy. There's not enough randomness in a phone number to prevent it from being brute-forced. We know that you cannot go backwards from a hash, from the hash back to the phone number. But you can put in all possible phone numbers and get all hashes that match those phone numbers. And that wouldn't be that hard. There's just not that many phone numbers, and hashing is very fast now, thanks to the cryptocurrency people that have, like, moved all this down to silicon. So it doesn't work to hash a low-entropy item like a phone number and use that for security. It's just not enough security.

So Moxie and company understood that. What they came up with was very clever. Intel has a set of features for everything from Skylake on, which is like three generations ago. Remember, Skylake was the laptop I purchased, my last Lenovo. And the Windows 7 machine that I built a couple years ago when we heard that Microsoft was not going to be supporting the older processors moving forward - or, wait, yeah, was not supporting the - oh, wait, the newer processors on the older OSes. I want to still be using Windows 7 for a long time, but I thought, uh-oh. I've got to get a Skylake processor now, which I did for both my next main machine and my laptop.

So Skylake and, what, there's a Haswell and something since. They all have this, what's called SGX, Software Guard Extensions. And this is very similar to what the ARM guys have built into, known as Trust Zone. It's an extension to the Intel architecture which allows the creation of protected software containers. The total size is limited to 128MB, so it's not gigs, but it's still large. And it creates a protected enclave in an Intel chip, much as Apple has done on their iOS devices, where they have a cryptographic enclave where they can store secrets. This is a means, supported since Skylake, for doing the same thing on any of our systems that we have.

So it was designed to support DRM, which none of us are big fans of. But the idea was that the actual contents of this region of RAM would be encrypted so that it could not be inspected. If it was inspected from outside the enclave, it would just be gibberish. It would be encrypted. But from code running inside the enclave, inside of this software guard, it would be Intel instructions and data, which could be viewed.

The other part of this is it can be - the technical term is "attested to." It supports attestation so that someone connecting to this from outside, like in the case of DRM, a content provider could be assured that no changes had been made to this software guard extension enclave so that the encrypted data going in is being decrypted and is only going to be played once - not, for example, exported as a torrent and then put on the Internet. So all of this exists and is in place. But this is all client side.

What Moxie and company figured out was that this could also be done on the server side to create a provably trusted, essentially - the term we've used is an HSM, a Hardware Security Module - create a virtual HSM just using Intel chip features to create a container where this hashing could be performed in a secure fashion. That is, a bunch of phone numbers would be hashed, and then that blob would be encrypted to prevent the hashes from being reversed. This encrypted blob would then go in up to the server, up to the signal centralized server, into the Software Guard Extensions enclave, where it would be decrypted back into a list of hashes.

Now, they worked all this out. The problem is that, even though you cannot decrypt - even though something outside cannot see in, it can detect memory access patterns. So they recognized that that would create, technically, as we know, known as a side channel leak. So just by inferring from the pattern of memory fetches, there's enough information there to leak what's going on inside. So then, and this is where I get into the pseudocode that they show in their coverage of this, they worked out, they carefully worked out a way to sort of design a meta hash matching system that deliberately decouples the work being done to match the hashes with the memory fetches which are used to do the work.

And the set of tradeoffs they came up with was one where there's a relatively large setup time, which is required in order to obfuscate where everything is in this scramble. That takes a while. But once that's done you can batch all of the various clients that want to compare their hashes against this master list because it's the master list that needs to be obscured. That can be set up once so it doesn't matter how long it takes. And what they ended up verifying was that they could do - they could essentially scale to over a billion users using this technology, essentially making the algorithm aware of the caching and cache lines, explicitly aware so that it would defeat the ability of anyone outside to observe memory fetch patterns and timing, thus the need to be caching aware, to be able to make sure that cache hit and misses aren't also leaking any information.

And essentially what this means is that Signal users using a Signal server with this technology - oh, and by the way, it's on GitHub. It exists. It's in beta at this point but it is freely published. The idea would be that the Signal server would implement this in the server hardware using Intel software guard extensions, and the client would be able to verify the integrity of this container before encrypting the hashes of its contacts' phone numbers and sending them into the container for decryption and matching and then weeding out.

What then happens is what's returned is a subset of the context of the user's contacts, which are a subset of hashes, which are then reencrypted and returned to the client, where it's then able to compare the hashes of all the phone numbers in its contact list to the smaller subset that are Signal users and flag all the contacts, all of the users in the contacts list as Signal users where they are, and then they can connect with them with absolutely zero leakage that any of the user's social graph has been exposed.

So bravo to Moxie, and this is what we want. I'm just thrilled to see this kind of attention being paid to security. Over the course of this podcast we've gone from sort of training wheels security, where it's like, oh, look, it's encrypted, we're done, to oh, no, no, no, it's encrypted and we're just getting started. Because our understanding that encryption is just a small piece of the whole problem has really matured over the last decade plus. Very cool.

Leo: Once again, Moxie Marlinspike to the rescue. Steve, I'm so glad to be back. I missed this show, and I missed the chance to ask you...

Steve: Glad to have you back, Leo.

Leo: ...all the questions about my security woes and share my issues with you. We do this show every, well, actually we're not constrained by time quite as much as we used to, so that's kind of nice. But the theory is we'll do this Tuesdays at 1:30 Pacific. We're a little late, and now you know why we were late. I was busily going through all my accounts to make sure I hadn't been compromised. I didn't want a Mat Honan. I was afraid.

Steve: No.

Leo: Yeah. I tell you, two-factor, baby. If you're not using it, use it. We do this show every Tuesday, 1:30 Pacific, 4:30 Eastern, 20:30 UTC. If you want to tune in live, by all means also join us in the chatroom. They're a great group. Mashed Potatoes says nice to have the full moustache back, Steve. Irc.twit.tv for the chatroom. You don't need to have an IRC client; but, by god, if you're listening to this show, you damn well better. I couldn't understand why you wouldn't. You also can get on-demand versions at Steve's site, GRC.com. He has a nice audio version. Plus it's the only place you can get a written transcript of everything Steve says. Does Elaine take out the uhs and the, you know - you don't say "uh."

Steve: I told her to do so, and she balked a little bit 12 years ago because she wanted it to be exactly - I said, "This is not a medical..."

Leo: It's not court. We're not talking court here.

Steve: Right, right, right.

Leo: She's a court reporter. I think that's probably where she gets that.

Steve: And I should mention that her husband is having some medical challenges.

Leo: Oh, no.

Steve: Yeah, Bennett.

Leo: Sorry, Elaine.

Steve: So the transcript, she warned me yesterday, may be not available till the weekend. So I already notified the people in the Security Now! newsgroup at GRC. And it'll probably affect her for the next week or two.

Leo: Oh, I'm sorry, Elaine.

Steve: So it's kind of major, yeah.

Leo: Okay. Well, we're thinking about you.

Steve: Yup.

Leo: She does such a good job. Twelve years she's been doing this? That's kind of amazing.

Steve: Yup.

Leo: She's been doing it almost as long as we have.

Steve: Every one.

Leo: Just go to GRC.com to find that and all the other great free stuff Steve gives away. And of course his bread and butter, and if you like Steve, support him by buying a copy of SpinRite. You'll be glad you did. GRC.com. It's the world's best hard drive maintenance and recovery utility.

Now, we have audio and video. I know I'll never - it baffles me. You didn't understand why we're doing video, I don't understand why we spend all that money on video. But people do like video. And if you want to watch, if you want to see Steve wave and do his "live long and prosper" properly with the thumb out, he does it right, and he does it with the right hand. No, you're doing it with your left hand. Should I do it with my right or my left?

Steve: I don't know.

Leo: Does it matter? I can't do it with my left.

Steve: Unfortunately, yeah, my right hand is behind the...

Leo: Yeah. You could do it with both hands, which shows one thing: a misspent youth. If you want the video go to TWiT.tv/sn for Security Now!. Or I think the best thing to do is subscribe. You could subscribe to audio or video versions in any podcast application: Overcast; Pocket Casts is I think number two after iTunes; Stitcher; Slacker; TuneIn. You can even listen, you know, if you have an Amazon Echo or a Google Home, you can just say, on the Echo, you say "Echo, listen to

Security Now! on TuneIn," and you can hear the latest episode.

Increasingly, I think that's how people are going to be listening to podcasts at home. It's just asking for it on their Echo or their - it works with Google Home. The syntax is slightly different. I can't remember what it is. Used to be you could watch it if you had an Echo with a screen, but YouTube's turned that feature off. But listening is fine. You could also listen to our live stream if you want to join us live by saying, "Echo, listen to TWiT Live on TuneIn." TWiT Live is the live stream. Does that take care of all of our business? I think it does. I think it's time for me, sadly, to say goodbye, and live long and prosper.

Steve: Well, until next week, my friend.

Leo: What do they say on "The Orville"?

Steve: Luckily, I don't know.

Leo: You have no idea, and you're glad.

Steve: I'm never going to know.

Leo: It's probably profane. Thanks. We'll see you next time on Security Now!, Steve.

Steve: Thanks, buddy.