Self-Decrypting Drives

Description: This week we cover last month's Patch Tuesday this month. We look at a GDPR-inspired lawsuit filed by Privacy International. We ask our listeners to check two router ports to protect against a new botnet that's making the rounds. We look at another irresponsibly disclosed zero-day, this time in VirtualBox. We look at CloudFlare's release of a very cool 1.1.1.1 app for iOS and Android. And, in perfect synchrony with this week's main topic, we note Microsoft's caution about the in-RAM vulnerabilities of the BitLocker whole-drive encryption. We also cover a bit of miscellany, we close the loop with our listeners, and then we take a deep dive into last week's worrisome revelation about the lack of true security being offered by today's Self-Encrypting SSD Drives.

High quality (64 kbps) mp3 audio file URL: http://media.GRC.com/sn/SN-689.mp3
Quarter size (16 kbps) mp3 audio file URL: http://media.GRC.com/sn/sn-689-lq.mp3

SHOW TEASE: It's time for Security Now!. Steve Gibson is here. Patch Tuesday is here. He'll cover that, as always; talk about a number of exploits and a number of routers again vulnerable; and then, finally, why you maybe shouldn't trust BitLocker when it comes to hard drive encryption. SSDs that decrypt themselves? It's all coming up next on Security Now!.

Leo Laporte: This is Security Now! with Steve Gibson, Episode 689, recorded Tuesday, November 13th, 2018: Self-Decrypting Drives.

It's time for Security Now!, the show where we cover your security and privacy online with the guru, the king of security, Mr. Steve Gibson. Hello, Steve.

Steve Gibson: Yo, Leo. Great to be with you on this November 13th, the lucky Second Tuesday of the Month. As I teased at the end of last week, some news had just hit as we were going to podcast about two researchers at a university in the Netherlands who decided to take a look into the details of self-encrypting SSDs, rather than just assuming, because they encrypt, that's all we need to worry about. It's like, oh, well, they encrypt. Oh, perfect.

They decided to take a peek under the covers. And what they found is not one of the seven that they examined, from Crucial and from Samsung - three were from Crucial, the other four were from Samsung, including the very popular 840 EVO and 850 EVO SSDs - none of them were doing the job right. And in fact it was possible without any information to basically decrypt the drive. Consequently, the title for this week's podcast is Self-Decrypting Drives.

Leo: Oh, boy. Well, the worst part of this is that in some cases Windows relies on these drives' encryption.

Steve: Yes, yes. We will have some takeaways for our listeners. The recommendation, well, these guys have just - I was glad that there wasn't an overabundance of other news because I wanted to give this topic enough attention. What's interesting is that, very much as with the vending machine problem, anyone who's been listening with focus to this podcast for a few years would be able to state, as we will, how simple this problem is to correctly solve. Yet somehow none of these drives solved it correctly. They solved it incorrectly. So anyway, so there's lots of good takeaways from this, including what anyone using BitLocker on one of these should probably do. But these guys go even further. So we'll get to that.

We also have - we're going to talk about last month's Patch Tuesday this month, that is, today is the Second Tuesday of the month, so it's our Patch Tuesday for November. We're also going to look at a GDPR-inspired lawsuit filed by Privacy International, just quickly, just to sort of put it on our radar in case it actually happens to develop into something. There is another new botnet in town making the rounds, and two ports that our listeners should just double-check on that we'll talk about.

We've got another irresponsibly disclosed zero-day, this time in VirtualBox, but probably not a huge problem, but worth, again, taking a look at. Cloudflare has just released a very cool new app for iOS and Android that allows those mobile platforms to use their now kind of increasingly popular 1.1.1.1 DNS. We've also, strangely enough, in perfect synchrony with this week's main topic, Microsoft is cautioning about the in-RAM vulnerabilities of BitLocker in the presence of, not surprisingly, Leo, you and I were talking about this a long time ago, the Firewire 1394 or the Thunderbolt interfaces. And so that's pertinent because we'll see what our guys have to say about whether hardware encryption is actually safer than software.

We're going to close the loop with our listeners, cover a bit of miscellany, and then, as I said, take a deep dive into the question of the actual delivered security of drives that advertise that they perform hardware-level encryption. And we'll note that 100% of the drives that were examined were found to be deficient. And that's to say nothing about hard drives. We don't know what they're doing. You don't know until you look. And there has not been enough looking yet. So I think an interesting and useful podcast for our listeners this week.

Leo: Can't wait. As always.

Steve: And a fun picture.

Leo: Yeah, I love the Picture of the Week.

Steve: Okay. So anyway, our Picture of the Week is fun. I found this, someone sent it to me through Twitter, and I just thought I got a kick out of it. So it shows two little cars with cameras mounted on their roofs, implying that they are self-driving cars. And so of course we all know that the famous question that the officer asks you when he pulls you over is "Do you know why I pulled you over?" And so anyway, the point is that we have a police officer, who's in the second car which is labeled the police car, standing at the window of the first driver, asking, "Does your car have any idea why my car pulled it over?" Which, yes, the world we live in today.

Leo: Yeah, that's probably true. Did you see that - and I don't know if this is more of a rumor than actually an announcement from Google, but their Waymo division is about to launch under a new name, which they're not telling anybody, basically Uber with self-driving cars. You'll be able to get in a self-driving cab soon.

Steve: Wait, who? Google?

Leo: Well, Google has a self-driving division called Waymo.

Steve: Okay.

Leo: It's Alphabet.

Steve: Right, yeah, yeah.

Leo: So they're going to - they may be the first to the table with - they're already testing it in Arizona.

Steve: So a no human in the car.

Leo: No human in the car.

Steve: Wow. That's just...

Leo: I bet initially they'll have what they call "safety drivers." Although there was a Waymo accident the other day that was caused by the safety driver because he didn't trust what the car was doing.

Steve: Oh, nice, nice.

Leo: Yeah. It's almost always the humans that screw things up for computers.

Steve: You know, it's weird. I've mentioned that I had a little bit of a high school reunion a couple weeks ago up in Healdsburg. And while we were having dinner, you know, we hadn't seen each other for 45 years.

Leo: Wow. Wow.

Steve: And they knew I was a techie and so forth. And so one of them said, "You know, Steve, of everything that's happened recently in technology, what would you say you

least anticipated?" And I didn't say what I wish I had said. I said, "Oh, no way could I have foreseen this crazy expansion in mass storage density. That's what's enabling so..."

Leo: Oh, yeah. That's absolutely true, yes.

Steve: It's just unbelievable that we can have half a terabyte on a keychain.

Leo: I have recordings of me saying, "By the year 2005, we'll all be using solid-state holographic memory. These ideas of these spinning drives make no sense whatsoever." So that's one thing. They've really managed to increase the capacity of spinning drives beyond any reasonable expectation. And then look at solid-state and the speed. In a way, that's the solid-state I guess I was thinking of. Not [crosstalk].

Steve: Yeah. Though in retrospect, what I wish I had followed that up with was the other thing that just - even, I mean, I accept today's storage. I still can't wrap my head around the idea that cars are driving themselves.

Leo: Yeah, yeah.

Steve: That just seems like, what?

Leo: It is remarkable.

Steve: What? What?

Leo: Well, and it was only 10 years ago. Remember we even talked about on the show the ARPA Grand Challenge where these so-called autonomous vehicles would basically go 10 feet and drive off the road.

Steve: Well, yes. And one of them would win if it could make it to the finish line.

Leo: If it got to the end.

Steve: It wasn't a matter of, like, getting there first.

Leo: No.

Steve: It was like, it didn't wander off into Colorado somewhere and disappear. No, I'm just amazed. So, wow.

Leo: Which - I'm going to put a little sidebar on this - shows you how important government investment is in these technologies. The ARPA Grand Challenge, ARPA,

comes from the Department of Defense. We wouldn't have self-driving cars at this level, I believe, if it weren't for ARPA.

Steve: Advanced Research Projects Agency.

Leo: Wouldn't have the Internet without them, either.

Steve: No, in fact there are major technologies which sane, bottom line-oriented entrepreneurs will not invest in because they're just too far out there. Well, that was before the days of Elon Musk, so maybe that's no longer the case.

Leo: No. But I do think, I mean, if the government didn't do it, it wouldn't have - no, you're right, it's not financially feasible.

Steve: It's just too big.

Leo: Yeah.

Steve: Yeah. Global network? What?

Leo: What? No.

Steve: I remember the first time I heard that.

Leo: Where's the profit in that? Right.

Steve: It was like, I remember the first time I heard that, I was like, that's going to take a lot of wires. You know? It's like, how's that going to work? Anyway, we now, of course, we know how it works.

We also know that Microsoft has today announced the release of last month's update...

Leo: It's out.

Steve: Yes.

Leo: 1809.

Steve: Yes. It's supposed to be out. I mean, they've said it's out. But they are releasing it slowly. I took off...

Leo: Yeah. As they should.

Steve: Yes. Once burned. I took off my hold-offs and did a manual update, and I got the November security rollup and so forth. But no sign yet of the feature update for Windows 10, which takes us from 1803 to 1809. And that's not surprising because they stated that they're planning to roll out the 1809 feature update slowly. So it may not appear immediately for anybody, even if you haven't tweaked your Windows 10 to say I'm going to wait to see how other people do with this before.

Leo: I already had 1809 on everything, so I can't tell you who's getting it. But, yeah, that's interesting.

Steve: Yeah. So for what it's worth, a system which was fully qualified, which is the one I'm talking to us over, my Skype machine, I updated it to Pro so I could have control over hold-off. And I set them to zero, I updated, I got the November stuff, but no feature updates yet. So it'll be, as they said, I guess they're just going to really watch this one closely as it rolls out. But they're convinced they understand everything that was happening before relative to data loss. Now, there were two other problems that just arose that didn't make it into today's conversation because it's sort of off-topic. But apparently Windows is losing its file associations so that it doesn't know what app is associated with what file.

Leo: That's always been a - remember, that was a problem all the time in the old days. All the time, yeah.

Steve: Yeah. And so that came back. I guess the adhesive on the patch got kind of old, and it just fell off. So that's now back. And who knows? So have fun, you Windows 10 people.

Leo: So mean.

Steve: So Privacy International is one of those companies like the EFF that is maybe a little reactionary, but we're kind of glad they're there because there needs to be somebody to push back on behalf of the consumer when profit-seeking companies don't have anybody to give them any oversight. So they have just filed complaints against seven companies for what they consider to be wide-scale and systematic, and that is to say deliberate, infringements of data protection law.

Their announcement last Thursday, November 8th, started by saying: "Today, Privacy International has filed complaints against seven data brokers." Now, what's interesting is that these are not companies any of us are aware of. We know about Equifax and Experian. Those are two of the seven. But the other ones are, like, huh? There's one you can't even pronounce, A-C-X-I-O-M. All the good names were taken.

Leo: It's Acxiom, though; right?

Steve: Oh, yeah, yeah. All of the good names were taken so it's like, A-C-X-I-O-M. And Oracle, but I don't think that's...

Leo: Oracle?

Steve: I think that's just some other - I think it's a name collision. It can't be our Oracle because I don't think Oracle's a data broker; are they? I think they're...

Leo: Well, they might be. They have their fingers in a lot of pies. That's interesting.

Steve: Well, maybe it is Oracle. And then three ad-tech companies: Criteo, C-R-I-T-E-O; Quantcast; and Tapad, T-A-P-A-D. And then of course Equifax and Experian. They say, with - so these complaints were filed with - wait for it - data protection authorities in France, Ireland, and the U.K. Meaning GDPR. "Privacy International urges the data protection authorities to investigate these companies and to protect individuals from the mass exploitation of their data."

So my feeling is, okay, so we knew the GDPR was going to ruffle a lot of feathers. And it does require, I mean, it's hard to browse the web now without having to acknowledge, yes, I know you're storing a cookie. Yes, I know, you're storing a cookie.

Leo: And that drives me crazy. There's no reason for that. Come on.

Steve: Yeah, it's like, okay.

Leo: That's absurd.

Steve: It's basically they've put a clickthrough on everywhere we go now.

Leo: Which, by the way, anesthetizes people to any real issues with tracking cookies.

Steve: Yup, yup.

Leo: Okay, okay, okay. You just click it because you want to get to the site. It's so annoying.

Steve: Yes, exactly. Fine, get out of my way, yeah, exactly. Or it's down at the bottom, and you think, okay, what's wrong with the screen? Something's broken. And then it's like, oh, there's a cover-over banner at the bottom. So click it to make it go away.

So anyway, they said: "Our complaints target companies that, despite exploiting" - this is their language - "...despite exploiting the data of millions of people, are not household names and therefore rarely have their practices challenged. In tandem with the complaints, we have today launched a campaign to seek to empower people to make it easier to demand that these companies delete our data." Which of course remember is one of the GDPR requirements.

They said: "Our complaints argue that the way these companies exploit people's data, in particular for profiling, is in contravention of the General Data Protection Regulation (GDPR), which took effect on May 25th, 2018. Our complaints are based on over 50 Data Subject Access Requests to these companies, as well as information that these companies provide in their marketing materials" - that's where these companies get in trouble because they brag about all the stuff that they've got; and then a company, a group like Privacy International comes along and says, what? - "and in their privacy policies," they say. "As such, our assertions are based on evidence that represents only the tip of the iceberg. We expect and anticipate the regulators will be able to delve more deeply into our concerns regarding wide-scale and systematic infringements of the GDPR.

"PI" - that is, Privacy International - "is encouraged that the UK's Information Commissioner's Office has issued assessment notices to Acxiom" - spelled in a way you could never anticipate...

Leo: All the consonants.

Steve: "...Equifax, and Experian. We are asking the ICO to take into account our submissions in the context of their ongoing investigation and urge the ICO to widen its investigation to include Criteo, Oracle, Quantcast, and Tapad. As part of our campaign, PI has made it easier for people to write to companies and demand they delete their data."

Okay. So I guess this is probably the way this was going to play out is that the regulation occurs, then it probably takes someone to point the enforcers at suspects and cause cases to be filed, or I'm sure that there's like some sort of a questionnaire that'll be sent. That'll go to the attorneys of the companies that'll respond in some fashion. And the point is I think it probably is the case that these companies have in the past been playing fast and loose with personal data. They're opaque, they're unknown, they probably are not actually in compliance today with the GDPR. So it'll take some enforcement in order to make them, you know, to bring them into compliance. And as you have noted here and on other podcasts, Leo, I've heard you talking about it, overall this is probably a good thing for the end-user in the long run. So yay.

Leo: Yeah, yeah.

Steve: Okay. That's all I wanted to say. It was just sort of to put that on our radar that there is, you know, Privacy International says, oh, now we have the GDPR, let's go poke a few companies that have been annoying us for a long time and stir the pot a bit, see what happens. So we'll see.

I have a chart in the show notes showing the BCMP UPnP - BCMP is Broadcom, and UPnP of course is Universal Plug and Play, I have to be careful not to say "pray" - the UPnP Hunter Botnet. And this is a time sequence diagram which is a nice way to organize protocol of "what happens when" diagrams. And so we can see on the left there's something known as the Infector. The first thing it does is send a TCP SYN packet, as in synchronize, which is the way a TCP connection is opened, to port 5431 of a candidate victim. If the victim responds with a SYN ACK, which is an acknowledgement of the receipt of the synchronized, the TCP synchronized packet, then the Infector says, ah, found somebody who's listening to connections on port 5431.

So it then sends an SSDP Discover packet, which is the Universal Plug and Play services discovery query over the UDP protocol to port 1900. Which, if the victim has their UPnP

enabled to the WAN, will respond with an infectable URL, basically saying here's how to take over. Here's how to take me over. Anyway, this is all by way of suggesting to our listeners that this would be a good time because this botnet is now loose to do two things: to make sure that your port 5431 is not responding to TCP SYN packets. And it turns out there just happens to be a handy-dandy port probe available...

Leo: Oh, wonder where that would be? Where can I find such a thing, Mr. Gibson?

Steve: If you put into your browser GRC.com/x, which I just chose because it's short, /portprobe=5431 and hit Enter, GRC will send some TCP SYN packets to your public IP at port 5431 and will notify you a few seconds later whether anybody answered a knock at that door. You want to either be - preferably you want to be stealth, or you want to be closed. You do not want to be open for business because that means that your router is almost certainly already taken over. So the first thing you could do would be - and I have a link in the show notes for anybody who wants it, or you could go to GRC.com, or just put "ShieldsUP" into Google, and it's the first link that comes up.

After you click through the first page, there's a text field in the middle. You can just 5431 and hit Enter, and it'll also launch a port probe. So you want to make sure that's closed on your WAN interface. And that's not something that anything else otherwise checks for you, so it's worth doing. Then the second stage is something, Leo, that this podcast caused the creation of. And I think I was able to implement it so quickly that, while it was still news, while we were doing the podcast, I said, oh, and I just added that test to GRC. That's that instant UPnP Exposure Test, which is still, I haven't touched it since I created it years ago, a big orange square that shows up on the page, and you just click it, and it does that second phase of what the botnet does. It sends a UDP packet to the user's public port 1900 and looks for any response, and analyzes it if it's found.

That's interesting because as far as I know, this has to be done manually. It would be possible to automate the use of this test. But as far as I know, that hasn't been done based on the rate at which we're seeing these occur. Yet 54,346 positive exposed Universal Plug and Play ports have been found so far since that test went live, 54,000. And nobody should have their Universal Plug and Play exposed publicly. So anyway, just sort of an update. And just so that people understand this, there is a widely used Broadcom chip which exposes a five-year-old flaw, that is, for five years it's been known in lack of Universal Plug and Play authentication.

And just because I wanted to make sure people understood this, I dropped the list of affected routers onto the show notes. And it's three pages. So this is not Grandma's dusty router. This is like the Who's Who of routers, including ADB, ASB, Billion, Broadcom, Clear Access, Comtrend, D-Link - several versions of D-Link, Digicom, INTEX, Linksys, Netcom, Opticom, I mean, I'm just scrolling through three pages. TP-LINK, Trend Data, ZyXEL, I mean, just it's a Who's Who of routers. So just it's easy to do. Just make sure you don't have this exposed because this thing is very aggressive. I've seen numbers as high as four million, although we believe that those are IPs which are coming and going, it's at least 100,000 at any given time, routers that are exposed. They're being used as email spamming proxies because we know that that's what Universal Plug and Play is able to do.

We're now seeing the bad guys are bouncing traffic off of other people's routers in order to launch denial of service attacks from that user's router, that is, that appear to be coming from there, and also the analysis of this - it took a while to set up a honeypot because they had to duplicate the behavior in order to get the botnet to believe that the honey pot was actually a defective router. But they found that they were sending email through this proxy that was being set up, essentially using this for spam. So you don't

want your IP to get blacklisted as a spammer, or you could have a hard time sending email yourself. So anyway, worth making sure that you haven't been zapped by this.

And we have another irresponsibly disclosed zero-day, this time in VirtualBox. A security researcher by the name of Sergey Zelenyuk discovered and posted a beautiful disclosure, I mean, I look at this. This is on GitHub. And you scroll through this, and it's clear how much effort went into this. And it's just - it's a lovely disclosure of what is a difficult - you know, Leo, you've got it on the screen, so just scroll down, and you'll see pages of beautiful code snippets that have been pulled out...

Leo: It's his rsum. He's trying to get a job.

Steve: Well, yeah, you wonder about - although we'll talk about what he's said in terms of his motivation. But so it's a difficult-to-exploit, not terribly worrisome flaw in the NAT handling code of VirtualBox's Intel E1000 NIC driver. So, and of course the Intel E1000 is like the very popular gig Ethernet NIC now. So because he didn't inform Oracle, the guys who were behind VirtualBox, he explains why he chose not to. And I had to abbreviate one word because I wasn't comfortable saying it on the podcast. You'll know when I get there. He said: "I like VirtualBox, and it has nothing to do with why I publish a zero-day vulnerability. The reason is my disagreement with contemporary state of infosec, especially of security research and bug bounty."

And so he says - and then he has sort of three major topics and one with some bullet points. He says: "First of all, wait half a year until a vulnerability is patched is considered fine." He says for the second point: "In the bug bounty field these are considered fine." And we have then four sub-points: "Wait more than a month until a submitted vulnerability is verified and a decision to buy or not to buy is made. Change the decision on the fly." He says: "Today you figured out the bug bounty program will buy bugs in a software. Week later, you come with bugs and exploits and receive 'not interested.'"

Third: "Have not a precise list of software a bug bounty is interested to buy bugs in. Handy for bug bounties, awkward for researchers." And fourth point: "Have not precise lower and upper bounds of vulnerability prices. There are many things influencing a price, but researchers need to know what is worth to work on and what is not." And, finally, his third main point, he says: "Delusion of grandeur and marketing BS" - that's where I chose to use an abbreviation - "naming vulnerabilities and creating websites for them; making a thousand conferences in a year; exaggerating importance of own job as a security researcher; considering yourself 'a world savior.'" And then he says: "Come down, Your Highness."

Leo: Wow.

Steve: Yeah.

Leo: Geez.

Steve: Sour grapes much? He says: "I'm exhausted of the first two, therefore my move is full disclosure. Infosec, please move forward." So as with the two cases we've seen of SandboxEscaper previously, and her release those two zero-days, and given the quantity and quality of the work that both she and Sergey have done, and Sergey's is every bit as impressive, it's understandable, I guess, that they might feel jerked around by the bug

bounty system and decided obtaining their 15 minutes of fame and notoriety is worth more than fighting for some cash.

But reading between the lines it's clear that probably, if they discussed with Oracle, if this guy had discussed with Oracle what he found, they might have - and he would have said it, I guess, maybe in general terms because he wouldn't want to give it away, or they might just jump the gun - they might have said, no, we're not interested. It's not that big a deal.

So anyway, we're beginning to sort of see this alternative with three of these irresponsible disclosures of zero-days in a relatively short period of time. So it's certainly true that, if the developer wants attention, they're going to get more attention like this than if they responsibly disclosed, wait for the problem to be solved, and then obtain a footnote of acknowledgment in the eventual security update.

But anyway, to the specifics here, all versions of VirtualBox from 5.2.20 and prior are vulnerable. This is true for any host because the bug is in a shared codebase. And it's true for any hosting, that is, for any host OS and any guest OS. So it's an any/any in terms of what's running on what platform with VirtualBox. And the VM configuration which is vulnerable is the default if the network card is the Intel PRO/1000 MT Desktop, which is a very popular network card, and running in the default NAT mode.

He says in his disclosure, under "How to Protect Yourself," he says: "Until the patched VirtualBox build is out, you can change the network card of your virtual machines to PCnet," he says, "either of two, or to Paravirtualized Network. If you can't," he says, "change the mode from NAT to another one." He says: "The former way is more secure." And then just sort of briefly I'll quote him, saying: "Introduction," he says, "A default VirtualBox virtual network device is Intel PRO/1000 MT Desktop, and the default network mode is NAT."

He says: "The E1000 has a vulnerability allowing an attacker with root/admin privileges in a guest" - so you first have to be admin in the guest - "to escape to the host's ring3." So that is the application layer on the host. He says: "Then the attacker can use existing techniques to escalate privileges to ring0 via /dev/vboxdrv." But again, so you have to have admin in the guest. Then you get to non-admin in the host. So I don't know, you know, like what went on behind the scenes. Maybe this just didn't rise to the level that Oracle felt it was worth getting all worked up about, and so this guy said, fine, I'm just going to go public.

He says, anyway, it is a VM containment breach which allows us to get out of the VM. That's not good, but it doesn't feel like it's the end of the world. There's a workaround, and I'm sure, I mean, especially given this beautiful disclosure that he put together, that there's already a fix in the works. So anyway, another zero-day dropped. In this case it's not nearly as much of a concern as something we've seen from SandboxEscaper, both of which were quickly leveraged.

And Leo, you're going to love this one, from our friends at Cloudflare. The site is - I love this - 1.1.1.1. Yes, you put https://1.1.1.1, which means they got the top-level domain "1."

Leo: Oh, I like it. That's right. They could do other dot one stuff, couldn't they.

Steve: Yeah. So they have...

Leo: So it's not an IP address. When you're entering that, that's the actual TLD.

Steve: Yes, right. Wait.

Leo: Or is it?

Steve: Wait, it's https.

Leo: Let me ping it. Let me ping it. Yeah, but that's, well, I don't know. That's an interesting question.

Steve: It's HTTPS, so they're not going to get a security certificate unless they've got a cert for that. That's a really good question. I didn't even think to look at who issued the certificate for that when I brought it up. Let's see.

Leo: It is at 1.1.1.1, but we knew that. Let's see. That's an interesting question. I guess I could do a WHOIS. Right?

Steve: Ah, DigiCert gave them a certificate.

Leo: For the top-level domain or for the IP address?

Steve: You can't get a certificate for an IP address. So the web - let's see, let's see. View, save password, blah blah blah, view certificate. Okay. Well, whoa, whoops.

Leo: No, [crosstalk] AP NIC. No, this is the Asia Pacific NIC.

Steve: Yeah, yeah, yeah. And the common name is *.cloudflare-dns.com. So that's where they got their certificate for it.

Leo: So they don't own dot one because everybody might - by the way, Cloudflare's going to do a domain registrar; right? For their customers.

Steve: Yes. In fact, we covered that a couple weeks ago. They have opened it up to all of their customers, not just their enterprise customers. Okay, so here's the cool thing. You go to, with iOS or Android, https://1.1.1.1, and it is now offering you an app, either for iOS or Android. I tried it. And you download the app. I used the iOS. You download it. It explains it's going to create a VPN profile - it's not a VPN connection, a VPN profile.

Leo: Oh, clever.

Steve: Yes. That's the way it makes it easy for them to change your DNS settings.

Leo: And Apple approved this, obviously.

Steve: Yes. And it then establishes DNS over HTTPS, Leo. It's not just changing your DNS. It is encrypting and privatizing all of your iOS or your Android mobile devices' DNS queries. And in the option screen the default is DNS over HTTPS. You can, if you want, switch it to over TLS. Both are emerging standards now. And there's other options and features. And so it changes your device to 1.1.1.1 and 1.0.0.1, which is their backup DNS. And essentially it establishes an HTTPS tunnel through which all DNS goes. And it logs it. You can look at the log. And what shocked me was how, I mean, like the moment I set this up, I went to look at the log, and it was like there were already 50 DNS queries from all kinds of crap.

Leo: Interesting.

Steve: In my phone.

Leo: You might want to note that other VPN apps will not work when you're using their VPN profile.

Steve: Right.

Leo: So it will eliminate the use of VPN for you.

Steve: Right.

Leo: Yeah. But that's, you know...

Steve: Anyway, I was very impressed.

Leo: This is great.

Steve: It's very cool. So takeaway is anyone who is self-conscious...

Leo: Oh, it's just a switch to turn it off if you wanted to.

Steve: Yeah.

Leo: So that's good. So if you wanted to use a VPN, you wouldn't need it anyway. You could do that. And then if you wanted you could turn it on.

Steve: Correct. Correct. And somewhere, see if you can find the log. I'm not sure where that log was. It's got a nice little icon that I liked, and I'm...

Leo: Yeah, here's the log. It's under the menu.

Steve: Ah, good.

Leo: Already I have quite a few things in here.

Steve: That's it.

Leo: How is that possible?

Steve: It's shocking how much DNS a device is making.

Leo: Most of it's to Apple or Apple-related sites like Akamai. But still, wow.

Steve: Yeah.

Leo: That's fascinating. Holy cow. Nice. This is great. So I wasn't going to - I was tempted to make this my pick of the week for iOS today on MacBreak, but I wanted to hear from you first. So thumbs up; right?

Steve: Thumbs up. Now, on their page they say: "Privacy First: Guaranteed." They say: "We will just remind our listeners about 1.1.1.1 through Cloudflare," which of course any of us could set our PCs to. "We will never sell your data or use it to target ads. Period. We will never log your IP address the way other companies identify you," they say. "And we're not just saying that. We've retained KPMG to audit our systems annually to ensure that we're doing what we say. And frankly," they say, "we don't want to know what you do on the Internet. It's none of our business. And we've taken the technical steps to ensure we can't."

Now, what I take a little issue with is they said "Faster than anything else; 28% faster, in fact." They said: "We've built 1.1.1.1 to be the Internet's fastest DNS directory. Don't take our word for it. The independent DNS monitor DNSPerf ranks 1.1.1.1 the fastest DNS service in the world. Since nearly everything you do on the Internet starts with a DNS request" - and obviously many things you don't do start with a DNS request...

Leo: Right, right.

Steve: "...choosing the fastest DNS directory across all your devices will accelerate almost everything you do online." Now, the reason I take a little issue with that is that I happen to also be the author of the Internet's sort of standard benchmark for DNS. And when this was announced, it may have been because it was immediately at announcement, there were many of our listeners who reported that they had faster DNS

than 1.1.1.1. But that was because they were still - I think it's because Cloudflare was still rolling out the endpoints, that is, what determines the DNS speed once you've got a fast server is its proximity to you. And that's what we are finding was that people were reporting that, nice as this sounded, to have all the privacy benefits and all, if there wasn't a node near them, then their traffic still had to go a long way in both directions, and that slowed it down.

Leo: It's also HTTPS. Would that be slower, too?

Steve: Not once the connection's established. What this does is it brings up a static connection. So it does the TLS negotiation.

Leo: Only once at the beginning.

Steve: Yes. And then it maintains a persistent connection so that it's just quick little packets zipping back and forth in order to get the work done. So what I would suggest is that maybe it's time to revisit. If you put in DNS Benchmark to Google, once again GRC is the only thing that comes up because we pretty much own that side of the world. I solved that problem correctly once. And it's worth taking a look at again because you may find, as our listeners may have at announcement time, that, yeah, we'd like to use it, but my own ISP's DNS server that's, like, next door to me is still fastest.

On the other hand, your ISP may very well be selling a lot of information about you because they'd like to make money, and there are companies that the folks like PI, Privacy International, are going after that would like to not have people able to do that. So anyway, Leo, yes, I think this is 100% win for iOS and Android mobile devices for anyone who's interested in the, first of all, speed. And I absolutely, we know the Cloudflare guys. There's no question that they are honoring their commitments about privacy. None whatsoever.

Leo: So I should point out, it doesn't do this on the iPhone, but on the iPad it does show that it has little VPN designation. This must be something new they're doing on the iPad.

Steve: Ah, didn't notice that.

Leo: But you're not really on a VPN.

Steve: Right, right, right, right. I see it, too. I hadn't noticed it. Actually, I like that as proactive acknowledgment that I've got my DNS redirected.

Leo: Yeah. But don't...

Steve: Ooh. Oh, oh, oh, Leo. Go to GRC Spoofability Test. I was going to mention in the Miscellany that it's back online. And it didn't occur to me, and I'll explain when I get there, but just put DNS Spoofability into the Google.

Leo: Yeah. The Google always know, yeah. Okay. And so if I run this with this on - I've got Cloudflare on. So if I just initiate standard DNS Spoofability, searching for DNS nameservers used by your systems - and this takes a little time, of course, because it's got to do some round-tripping here.

Steve: Oh, it's doing, well, it's doing hundreds of queries. Yup, I'm doing it now, too.

Leo: Two servers found; 456 queries received. I'm doing this on the iPad, which is an interesting...

Steve: As I am. Right now I'm holding my iPad also. I found I had two servers found, 409 queries received. So what happens is, if it finds no additional servers, then it'll do three more lines where it - because it's continuing to look for any - oh, I found now I've got four servers found and 1046...

Leo: Is that good or bad?

Steve: Well, it's why I designed the test this way. It's not bad.

Leo: It's kind of thoroughly looking, yeah.

Steve: Yes, exactly. So that on the third line it found zero additional ones. And so it'll do that three times to make sure that no more reluctant servers come in. But then it's going to show us what it knows about the servers that we're using, and they will not be your ISP's. They should be, yup, and fourth line through now, zero additional.

Leo: So to be clear, your traffic is not encrypted through a VPN when you're using this. What's encrypted through a VPN is merely that DNS query. And your traffic is traveling normally over the public Internet. Just so people don't get confused when they see that VPN icon. You're still on the public Internet.

Steve: Correct.

Leo: Oops, server stopped responding. Safari did not like the delay. So I don't know.

Steve: Oop, found two more servers on the sixth round of testing.

Leo: Yeah. You didn't time out. I did, unfortunately.

Steve: Ah, no, I did time out. "Could not open the page because the server stopped responding." Interesting.

Leo: That's Apple's defaults.

Steve: [Growling]

Leo: [Growling] So we may not know. What would I expect to see?

Steve: We don't know until we find out.

Leo: Well, I'll try to run it on the desktop here and see what happens.

Steve: And I noticed that our listeners are doing this because I see a large traffic spike.

Leo: All of a sudden everybody's...

Steve: GRC is sending lots of DNS queries out.

Leo: At firs they were just downloading your DNS Benchmark. But now they're doing this, too.

Steve: Now they're doing [crosstalk].

Leo: You really don't mind hitting those servers, do you.

Steve: No, no. So anyway, I think this is all good. Privacy while you're on the road. Very, I mean, it's so simple to set up that you can easily tell your weekend Tech Guy listeners how to do this, and it prevents anybody from - now again, you're right, it's not preventing them from seeing where you go. But it's preventing them from seeing the domain names that you're looking up. And of course that's a big privacy leakage. So it's worth patching it. And if they are right, it's faster. And of course everybody wants their stuff to be faster.

Leo: Yeah, cool. So I'm doing it on the desktop now. I'll do it on the Mac, and I'll try it with Chrome on a Windows machine, too, just - oh, it won't - it will only let me do this from one IP address, once at a time. So I'll wait until it's done.

Steve: Ah, that's right. It does. I remember.

Leo: You're too smart. Damn, he's smart.

Steve: Well, in fact, you'll like the charts it comes up with because it remembers - I developed this after Dan Kaminsky showed us that DNS servers were not randomizing their queries. And it was the lack of random query data which allowed someone to guess what the query would be and thus spoof a reply in order to do DNS spoofing, and so thus DNS Spoofability.

Leo: Episode 155, if you want to hear more.

Steve: Ah, cool. So while we're doing that, I'll mention just that it was sort of an interesting bit of coincidence that Microsoft published a note about BitLocker's exposure of its keys in RAM.

I got it working, the Spoofability Test, on iPad.

Leo: Oh, look at that. That's cool.

Steve: Yeah, that's the - what I did was I remembered there's a custom test. And so you're able to customize the parameters. And so I've made it less patient so that Safari wouldn't time out. And so I used...

Leo: And so our antispoofing safety is excellent.

Steve: Whoa, look at that. Very nice.

Leo: That, of course, is because Russell really does a good job of protecting us. 4,264 queries. 14 IP addresses. Do you want to see a very even distribution like that?

Steve: Yeah, because - yeah. And in fact what the histograms down below are the number of times we saw each of the different bits of the 16 bit. And so they're all very low and very flat. You don't want to see any of those being high.

Leo: Very good.

Steve: Very nice.

Leo: We are not vulnerable to the Kaminsky Exploit, I'm happy to say. Now, I'm not running 1.1.1.1. But this is an example of what it would look like. And you're getting - from 1.1.1.1 you get the same kind of...

Steve: Well, I actually saw what looked like some poor query transaction IDs, that is, the right-hand chart showed some lines in it a couple times. But I just reran it. It was actually a different server that I got the first time.

Leo: Okay.

Steve: So it was sort of interesting. And actually, yeah, I only found one server because I told it not to be very impatient so that it would finish. But if you want to try it on your iOS device, there's a link at the bottom of the first page that's custom DNS, the custom

spoofability test, and you change the last three numbers to 10, 0, and 1, and then it will succeed.

Leo: And this is OpenDNS, by the way. That's what we use. And that's pretty good. So OpenDNS would be another good choice for people if they don't get good results from quad ones.

Steve: Well, and I think it was just - I retested, and everything came out absolutely great. So it might have just been a weird fluke the first time.

Leo: I'm using quad ones now. I like it. I'm going to use it on all my mobile devices. That's great. That's cool.

Steve: Yup. So coincident with the interesting discovery that hard drives are not doing a good job protecting their own secrets, Microsoft released sort of an advisory, a security advisory, that was warning something that we've talked about for years, actually, which is that it's an interesting fact that both the earlier 1394 so-called, god, I want to say Lightning, but it's not, the Firewire, the 1394 Firewire interface, and then the later Thunderbolt, both allow the connected device to do essentially DMA, Direct Memory Access.

And this has been a security concern because it would mean, and we've talked about this, again, through the years, it would allow somebody to connect a Firewire or Thunderbolt device which is intelligent, it's not just a dumb drive or a camera or something, it's actually a smart controller which then is able to go over the serial interface, the Firewire or Thunderbolt serial interface, and suck out the contents of the system's RAM.

The reason that's an issue, and we'll be discussing this here in a minute relative to drive encryption and decryption, is that most systems, well, in fact BitLocker specifically, in order to do its job it has to have its key in memory. That is, somewhere in the system memory is the key in use the whole time your computer is on, in order to read and write to and from the drive.

Now, that's not unique to BitLocker. I mean, this is why, for example, Heartbleed that we talked about a couple years ago, the Heartbleed attack was opportunistically grabbing RAM from a server, and there was some probability or some chance that the RAM it would grab was valuable, that it contained the server's key. Which is what made it such a problem and a great concern.

So anyway, the point is that Microsoft says: "A BitLocker-protected computer may be vulnerable to DMA, Direct Memory Access attacks, when the computer is turned on or is in the Standby power state. This includes when the desktop is locked. BitLocker, with TPM-only authentication" - which is sort of the default, so just it's encrypted and it's not harassing you - "allows for a computer to enter the power-on state without any pre-boot authentication. Therefore, an attacker may be able to perform DMA attacks." And so the idea would be you'd turn the computer on, and there it is. And now, if you have DMA access, it can get to your key.

They say: "In these configurations, an attacker may be able to search for BitLocker encryption keys in system memory by spoofing the SBP-2" - that's the Serial Bus Protocol - "hardware ID by using an attacking device that is plugged into a 1394 port. Alternatively, an active Thunderbolt port also provides access to system memory to

perform an attack. Note that Thunderbolt 3 on the new USB Type-C connector includes new security features which can be configured to protect against this type of attack without disabling the port."

And we've talked about that when we've brought this up before. Even the MAC and Firewire had some provisions for limiting the range of memory that was available to DMA, though in many cases that wasn't the setup by default. So then they say: "This article applies to any of the following systems: systems that are left turned on; systems that are left in the standby power state; and systems that use TPM-only BitLocker protection." And they go on to talk about how you can reconfigure what choices you make in order to protect yourself against this.

I've got a link in the show notes for anyone who's interested. But I thought I wanted to bring this up because the researchers who did not know about this because their research predates the publication of this from Microsoft, bring up exactly this point when they are very soberly talking about what it is they found and how they feel about software versus hardware full-drive encryption.

So that wraps up our news for the week. I had under Miscellany, interestingly enough, the first item under Miscellany is GRC's DNS Spoofability system is back online.

Leo: Nice timing.

Steve: Yeah. And what had happened was I didn't realize it had gone down, but someone - I saw a couple mentions in Twitter, and I thought, okay, I've got to look into that. And then I found a posting, someone had cross-posted from GRC's DNS newsgroup into DNS SQRL because they knew that's where I was spending all my time. And that really caught my attention. So finally I thought, okay, I've got to see what's going on. And so I fired it up; and, sure enough, it didn't work. And so it's like, oh, crap.

So last weekend I thought, well, maybe I just need to reboot the server. The GRC server had not been booted since last November because it never crashes, and there are no memory leaks. And if it's not broke, just leave it alone. So I thought, okay, well, be a good time to update it and maybe just something got a little cranky. I brought it back up. It didn't fix it. It turns out, and I don't really understand exactly what the sequence is, because I don't understand why the reports are it's been working until just a few months ago.

But the problem was the compression, the HTTP compression, which I've always been a big fan of and has always been enabled, it stopped sending out incremental pieces of the page. And the spoofability page sends little dots out on the page to show you its progress as it's going. And the dots represent DNS - the dots come out when the user's browser asks for zero-size GIF or GIF, depending upon what side of the fence you're on, images. And the GIF images come from a fake DNS server that's 13 characters of random dot DNS dot GRC.com.

So what I built back then was a pseudo DNS server which grabs those queries from the user's browser that needs to look up the IP address of this crazy DNS name. And when the request is made, I then return a CNAME, which is a canonical name, of a.a.a.a.a.a, like as long as it can legally be. And then so that goes back to the user's DNS server that says, oh, you're kidding me. And so it says, okay, what's the name server for that? And so this is a means of forcing hundreds of queries in a short period of time from all the DNS name servers that the user has serving them, that is, the user's browser has serving them.

Anyway, it broke because something about the compression in Windows stopped doing incremental releases. I don't know what it was, but as far as I know it's nothing I did. Anyway, I made an exception to simply turn the compression off for the active code that my site uses, that is, that's underneath the /x, and then it just came back to life again. So anyway, for anybody, I know that there are listeners of this podcast who have been discomfited by the fact that GRC's DNS Spoofability system which they use from time to time had been offline for months.

So I wanted to let everybody know it's back. And the timing is nice because people can play with it with their iOS devices. And if you want to play with it with your iOS device, and you find that Safari times out, I have a custom version, and you can set - the last number, there's like five numbers which characterize the nature of the test. If you set the last one, it defaults to four, which is how long it waits for more servers to show up. If you set it to one, then it's less patient, and you'll get some results. And there's somewhere else where there's a four you can set to a 10. Anyway, so it's set 47, 48, 10, something, and then one, and then it'll work for you.

Also, Simon Zerafa, a friend of the podcast, pointed me to something that I was just starting to play with, Leo, when you decided it was time to wrap up MacBreak Weekly today, which is really interesting to me. And so I wanted, to any of our listeners who are webmasters, that is, who are in charge of their own web servers, Content Security Policy, CSP, is a neat facility which allows a website to tighten up its security by telling the browser what resources it's sourcing from where. And so what it does, the idea is that so the website says this is our Content Security Policy. For example, scripts should only come from here, and images should only come from there, and so forth.

The problem is, adding that afterwards, that is, on a live site, is nerve-wracking because it's so easy, you know, the point is you want tight access control. But you don't want too tight access control or you'll block some valid assets of your site. And so what this does is, if you really have your CSP policy locked down, and it's something you could easily do from the beginning, but it's hard to add afterwards, if it's locked down, it's very difficult for anything that somehow gets loose on one of your pages to get up to mischief because it's like it just can't - it has no freedom and flexibility, or vastly less.

So the point of this is there is a widget, which Simon pointed me to. It's called - for some reason it's called Laboratory, although it's Laboratory by Mozilla, so maybe it's Mozilla's. I haven't had a chance to look at it closely. But it's a widget for Firefox.

Leo: Yeah. This write-up is from Mozilla.

Steve: Oh, okay, cool. So what it does is you go to your site. You activate this thing. And then you browse around your site. You visit pages. You do stuff. And it's learning where your different types of assets come from, and it builds for you the Content Security Policy for your site, based on its real-time interactive behavior while it's not being attacked. So the result will be a CSP, like a long, crazy-looking header string, which you could then add to your web server without breaking anything, so long as you didn't do anything that you hadn't done while you were roaming around your site.

Anyway, I'm going to play with it. I'll know more about in the future. I would love to have, for GRC, a tighter Content Security Policy because why not? But I'm just afraid to mess with it because my site's old, and stuff's coming from all over the place. This would help anybody who wants to add this to an existing site to do so.

And the last piece of Miscellany, Leo, is Peter Hamilton's "Salvation."

Leo: Have you finished it now?

Steve: Finished it. And I would characterize it - I gave it five stars. I understand the negative comments in the reviews. I mean, there are some people who only gave it two because they just want to drop right in to start blowing stuff up, and this doesn't do that. I would argue this entire first book is setting the stage for the future. And I don't know whether - I don't know if it's a trilogy, or if it's more. And maybe three quarters of the way through I was sort of thinking like, okay, I'm not going to be impatient. I'm just going to know this is Peter, and he's going to take me for a long journey.

And the only problem is now I have to probably wait a year. And this is what happened with "Pandora's Star," remember, is we got this fabulous first book, and then it was like, okay, wait for the second one. It's like, oh. And it was a year later. I had to reread "Pandora's Star" in order to remember what had gone on so that we could get to "Judas Unchained."

So anyway, for what it's worth, it really is good. And one of the things that Peter sometimes does, he did this at the end of "Fallen Dragon," is the whole book was a build-up to a surprise which was kind of teased in different ways as you're going along, and it's like, okay, I don't know what that means, but I'll find out. Okay, I don't know what that means, but I'll find out. All was answered by the end. So there's nothing at all cliffhanging except that now I want to know what's going to happen next. So I've got to wait a long time. The good news is the tenth book in Ryk Brown's second of five series of 15 books each - so, yes, that's 75 books.

Leo: Oh, my god.

Steve: It came out. It was released on October 28. So now that's what I'm reading. So I've got plenty to keep me busy until Peter Hamilton figures he's going to tell us...

Leo: I almost don't want to get involved in anything that long because then I won't read anything else for years.

Steve: I know. His stuff is so good, though, Leo. I mean, we have always enjoyed it. I could certainly understand somebody deciding to wait until the whole series is done.

Leo: The Hamilton stuff, yeah, yeah.

Steve: Before starting.

Leo: I think that's my choice.

Steve: Oh, oh, you mean...

Leo: Oh, you're not going to wait for Ryk to finish 85 books, are you?

Steve: He does spit these things out. I don't know how he does it. They're not full of typos, I mean, they are really well written. And they are, for anybody who gave "Salvation" two stars, you've got now 25 books in the Ryk Brown series, the 15 books of the first set and then the 10 books of the second. So there's 25. That ought to hold you for a while. And they are rock 'em, sock 'em, you know, they're like right down space opera, classic space opera. "Salvation," though, I'm definitely glad I put the time in for this first one because he's created another amazing future. So I want to know about it.

Leo: JammerB loved it, too. He said, "I can't till Steve finishes so I can talk to him about it."

Steve: Cool. Well, speaking of finishing, Nicholas Kasprinski sent me a tweet that is again apropos of today's topic. He says: "I have a drive that is encrypted with Windows BitLocker" - well, or maybe not. But, he says: "...but is not able to boot up. Can I still run SpinRite against it to decrypt and recover the drive's data?" And as we know, the answer is yes. SpinRite doesn't care what data you have on the drive. It just cares whether the drive is happy with its ability to read it. And if the drive is not happy, even when it seems happy, but we ask it a little more, are you really, really sure? You really sure you're happy with this? And if it seems a little questionable, then we fix it.

And if the drive says, no, I'm not happy, we say, oh, come one, try again. No, I'm not, really not. No, no, no, no, no. Just give it, you know, look again. And we do lots of things to get the drive to finally say, okay, fine. One last time you can have the data back. And that's all we need, one last time, because then the drive fixes it, it relocates the sector, brings in a spare, we write the good data back, and the drive boots again. So Nicholas, yes, SpinRite is the only thing that I know of which is able to fix a BitLockered drive which will not boot. So good luck.

I got four little bits of closing-the-loop data. Davy Jones, tweeting from @9arsth, said: "Need a quick yes-no answer. Applied the sandbox to my Win10 home box. Tested, it works. Then applied to W7 Pro. Command line said it was successful, but when tested, no child process under MsMpEng. So not possible?" And so decoding that, I realize he's talking about what we talked about last week, which was the sandboxing of Windows Defender. And I probably wasn't clear, or maybe I didn't say it at all, that it's only Windows 10. This is not something Windows 7, any version of Windows 7, in Davy Jones's case Windows 7 Pro. This is more and more we're going to, unfortunately, see Microsoft offering things that we Windows 7 users wish we could have, but we've got to go to Windows 10 in order to get them. And, yeah, okay. No, thank you.

But anyway, so the point is that that command line was a registry entry which only Windows 10 knows about. You can certainly put that in Windows 7, but it'll have no effect. So, yes, it is the case that the Defender in the sandbox, which is absolutely what I think is the ultimate solution now for integrated, high-quality AV that doesn't bring with it a risk of creating a larger attack surface for viruses, it's Windows 10 only.

David P. Vallee tweeted, oh: "In the SQRL paradigm, what happens if there's a system-wide collapse?" He says: "People would have hundreds of accounts with no record of their credentials because there was reason to record them" - so he must mean because there was no reason to record them when they were created. And David, you'll be glad to know that there's no such thing as a system-wide collapse because there's no system to collapse. So what is an arguably somewhat mixed blessing is that there is no one to go to if, like, you forget how to authenticate yourself to SQRL, or if your SQRL identity is lost. As a consequence, a huge amount of effort has been put into recovering from those events. But there isn't a system. There is no central headquarters. There's no main

arbiter. And I would argue that's the whole point. This is a two-party solution between you and the websites you visit. So there is nothing to collapse.

Skynet says: "Hi, Steve. With Windows Defender sandboxing, how do you turn it off if you need to? Do you retype the command with a zero at the end instead of a one to disable it?" And that was a great question I forgot to mention. Yes. That's exactly what you do. In the command line which sets the registry key value to one, you change it to zero and then restart your system, and it will no longer be sandboxed.

And, finally, Edward Evans says: "Hi, Steve. Regarding SQRL, how would I be able to have a bot associated with my user account without sharing my core secret?" He says: "For example, a web watcher that is watching on the far side of the auth wall." He says: "Love Security Now!. Thanks to you, Leo, and the gang, I feel that I actually 'get' security now. Thanks for all you do."

And that's a really interesting question. I mean, that's definitely an edge case that I hadn't thought about. So he's saying how would he be able to have a bot associated with his SQRL identity without sharing his core secret, a web watcher that is watching on the far side of the auth wall? So, okay. So first that would assume that he wants to delegate his identity to something which presumably has to log in in order to do whatever it needs to do. So this is not something that the SQRL client itself supports.

But SQRL creates a per-site sort of - you can think of it like as a sub-identity, that is, from your master identity, which is mixed with the domain name of the site you are wanting to authenticate to. A secondary identity is created which is per-site. It's unique for every site you visit. So, and it doesn't have to be a secret. So, I mean, for example, that's the identity. The public version of that is what you give to the site to declare who you are. The private component of that is how you sign a challenge from the site saying prove to me that this is who you are.

So it would be possible, and somebody could easily create it, I could or anybody could, sort of something that peels off just one site-specific identity from your master SQRL identity that would thus only work on that one site. And you could give that to a bot, which could then use that identity only valid for that one site in order to operate on your behalf without worrying that it's able to do anything else. It would not have access to your larger SQRL identity. So what's cool is that so far all of these sorts of questions we've had answers for, which gives me hope for SQRL. And after working on it for as long as I have, I do have hope for it. But we'll see how it goes once we turn her loose.

Leo: Steve, I just thought before we get back to the show, kind of a tragic note, I know you've been around long enough to know the name Bill Godbout.

Steve: Oh, my goodness, yeah.

Leo: The S-100 computers, the Godbout Electronics and CompuPro.

Steve: Yeah.

Leo: He passed away in the Camp wildfire at the age of 79.

Steve: Oh.

Leo: He was one of the, now we're learning, dozens of victims of these wildfires.

Steve: He was a real engineer.

Leo: Amazing guy, 79 years old, Bill Godbout. He was an advocate for S-100 computers and industry standards and CPM days. He was a parts supplier for electronic music projects, and a big part of the computer revolution. In fact, in "Hackers," Steven Levy's book, he writes about Bill Godbout. He said he bought junk on a more massive scale; usually government surplus chips and parts were rejected as not meeting the exacting standards required for a specific function, but perfectly acceptable for other users.

"Godbout, a gruff, beefy, still-active pilot, who hinted at a past loaded with international espionage and intrigues for government agencies whose names he could not legally utter would take the parts, throw his own brand name on them, and sell them, often in logic circuitry kits you could buy by mail order." That was the wild west days of computing. A legend, and a very sad end to one of the greats in the business. So I thought I'd pass that along because I knew you would know the name, even if many others in our audience don't.

Steve: Okay. So the title of the 16-page, very well written and wonderfully detailed research paper from two security researchers at the Radboud University in the Netherlands is titled: "Self-Encrypting Deception: Weaknesses in the encryption of solid-state drives." And I will remind everyone that, essentially, they could also have said weaknesses in 100% of the drives we have looked at, which is to say they looked at three drives from Crucial, three internal and two external drives from Samsung - I'm sorry, two internal and two external drives from Samsung. Collectively they make up about half of the market. So they are popular drives. And they found the security wanting, to put it mildly.

So there are two standards in the industry. There's the so-called ATA security standard, and what's known as the Opal, O-P-A-L, standard. Opal was produced by the Trusted Computing Group. ATA security was the original. And that's what drives have had for years, which essentially locks and unlocks an ATA drive as a whole, based on a user-supplied password. And, for example, any of us that have been using computers for a long time, oftentimes our BIOS will allow us to supply a drive password. And so the idea is that at boot time the BIOS supplies the password, which then unlocks the drive so that there even is a boot sector available to boot from.

And so the protection is that, if the drive is removed from the device, it doesn't have this password that you have assigned in the BIOS. And oftentimes you have to authenticate, the user has to authenticate to the BIOS in order for the BIOS to then unlock the drive. So there's boot time protection. The successor to that original, the so-called ATA Security Feature Set, the ATA spec groups things in feature sets. So like there's the smart feature set, the security feature set, and so forth. That was created by this Trusted Computing Group, and they're the same people who did the TPM, right, the Trusted Platform Module. They are a committee that generates these standards.

Opal defines a much more complex and unfortunately much more difficult to implement encryption facility. And so, for example, if a much simpler and more straightforward ATA system could be said to be a one-to-one scheme where one password is used to unlock one region, which is to say the entire drive, then the Opal system would be described as a many-to-many system, where a drive's storage space can be freely subdivided into

arbitrary bounded regions, each of which can be individually encrypted so that they are accessible under multiple and individually differing passwords so that different passwords can selectively decrypt various subsets of the entire drive.

I'm not kidding. I mean, that's this disaster that we have. And it's a disaster because nobody uses it; yet the drives, in order to be compliant, and do have the checkbox on the marketing brochure, have to support it. And if they're going to support Opal, if they're going to say that they support the TCG, the Trusted Computing Group's Opal standard, then they have to do that. Despite the fact that operating systems don't care about that. They just want to unlock the whole drive. But wouldn't it be nice if it were actually done in a secure fashion?

So, in digging around a little bit, I ran across an article written seven years ago, in 2011, titled "The Pros and Cons of Opal-Compliant Drives." And in that article its well-meaning but naive author states, quote: "Hardware-based encryption is very secure, far more secure than any software-based offering. Software," he writes, "can be corrupted or negated, while hardware cannot. Software runs under an operating system that is vulnerable to viruses and other attacks. An operating system by definition provides open access to applications, and thus exposes these access points to improper use. Hardware-based security can more effectively restrict access from the outside, especially to unauthorized use."

And I wrote here in the show notes: "This charming view of the world predated the Spectre and Meltdown hardware problems by seven years, and it assumes that hardware is magically perfect because it is harder" than software. It would be nice to live in that world. Of course now we no longer do. So the number one lesson we learned from this terrific research, that is, the research, the 16-page paper, is that the fact that a drive sports a given interface and supports a given security standard whose marketing brochures boast of its security and encryption is entirely decoupled from and means absolutely nothing about the actual delivered security in the face of a determined effort to obtain the drive's content.

Is the encrypted drive unreadable to the casual passerby? Absolutely. Is it unreadable to someone who is highly motivated to gain access to the drive's contents? That's the interesting question this pair of researchers set out to determine. And the initial surprising successes they had kept them going. Now, just remember years ago when we were talking about TrueCrypt and how there was a drive in Brazil that was encrypted with TrueCrypt, a pure software solution, which because apparently a good password was used, no authorities there were ever able to gain access to it, despite the fact that they desperately wanted to. And so they shipped it up to the U.S. FBI and said, "Hey, you geniuses, we need to know what's on this drive." And as far as we know, that was never possible. That was software encryption, and a perfect example of it. And now we know, and we will be hearing a term here in a minute, we know that TrueCrypt then became VeraCrypt, which is the currently maintained, non-BitLocker alternative for - you can use it on Windows and other platforms, as well.

Okay. So these two skilled researchers from the Netherlands invested significant time and attention to reverse engineer seven mainstream, highly popular SSDs which account for, as I mentioned, collectively nearly half the market. The abstract of their research reads: "We have analyzed the hardware disk encryption of several SSDs by reverse engineering their firmware. In theory, the security guarantees offered by hardware encryption are similar to or better than software implementations. In reality, we found that many hardware implementations have critical security weaknesses, for many models allowing for complete recovery of the data without knowledge of any secret.

"BitLocker, the encryption software built into Microsoft Windows, will rely exclusively on hardware full-disk encryption if the SSD advertises support for it. Thus, for these drives,

data protected by BitLocker is also compromised." That is, for these systems. They said: "This challenges the view that hardware encryption is preferable over software encryption. We conclude that one should not rely solely on hardware encryption offered by SSDs.

"We have analyzed firmwares from different SSD models offering hardware encryption, focusing on these flaws. The analysis uncovers a pattern of critical issues across vendors. For multiple models, it is possible to bypass the encryption entirely, allowing for a complete recovery of the data without any knowledge of passwords or keys. The situation is worsened by the delegation of encryption to the drive by BitLocker. Due to the default policy, many BitLocker users are unintentionally using hardware encryption, exposing them to the same threats. As such, we should consider whether hardware encryption is a true successor to its software counterpart, and whether the established standards actually promote sound implementations."

Okay. So what happened? Regular listeners to our podcast will know the correct way to manage a drive's encryption, and many of us could clearly state it. So here it is, clearly stated. This is the way you do it. At the factory, the very first time the drive is powered up, a high-quality entropy source - whether built-in or external - is used to produce a large (128- or 256-bit) high-entropy symmetric secret. That key will forever be used to key an in-line AES algorithm which encrypts and decrypts the drive contents on the fly.

So each drive, first time it's turned on, either makes it internally if it has a good source of entropy, or gets it from the factory because the factories are easy. It's easy to have a good source of entropy. Every drive, a per-drive unique key which it has and forever uses to key its own AES algorithm. What that means is that all the data physically stored on the drive is gibberish. It is indistinguishable from random noise and can only be de-gibberized if you have the secret key.

If the drive is not password protected, it is still encrypted, transparently, under that unique per-drive secret key. Since a hardware implementation of the AES Rijndael cipher is inline and able to keep up with the performance of the storage medium and the drive's external interface, whichever of the two is slower, that is, so that the inline encryption doesn't slow anything down, there's no performance penalty. And note another feature of this is all you have to do is change the key, and you have securely completely wiped the drive with cryptographic security, including all the sectors that were ever spared out and taken out of service. So that's the way drives should be designed.

If the drive's user or operating system wishes to later protect the drive's contents, a user or OS-supplied secret is run through a PBKDF, a password-based key derivation function which is internal to the drive, to produce a password-dependent key which is subsequently used to encrypt the drive's original master encryption key. The drive's original factory-set master key is physically overwritten with its password-encrypted version so that the original master key no longer exists anywhere on the drive. Right? We all know that. This is not rocket science. This is the way you solve this problem.

And, finally, to allow a candidate password to be verified by the drive before being applied, a different PBKDF2 hashing function should also be used to independently verify any would-be decryption password. In other words, when the password is created, it would be run through a separate hash with different parameters, and the proper hashed value would be stored by the drive. That way the drive can verify in a safe fashion whether the password given to it is correct, yet there's no way to go from the stored verifier to the - there's no way to go from that over to the key used to decrypt the encrypted master key.

So as we know, security is inherently a weakest-link phenomenon. And if even one of those things that I just described is not done properly, the system's weakest link will be

feasibly broken, and the system security guarantees will fail. Disturbing as it is, among those seven very popular mainstream SSD drives, failures in every one of those aspects was actually found.

So before I go any further, I should note that, when the researchers realized just how bad the situation was in this industry, they elected to handle its disclosure responsibly. So in their write-up they explained, in their responsible disclosure section, they said: "After discovering these vulnerabilities, we followed a process of responsible disclosure. In this case, the National Cyber Security Center (NCSC) of the Netherlands was informed first, which assisted in the responsible disclosure process by setting up the contact with the manufactures involved," in this case Crucial and Samsung.

"We cooperated with both manufacturers to fix their products and agreed not to disclose the vulnerabilities for six months. Both vendors have confirmed all the reported issues. For models currently being supported, firmware updates are either released or currently in development." And I should say that that is now the case, that there is now updated firmware for all of these. So in the properly designed system I outlined above, we would say that the key is bound to the password, meaning that the information provided by the password is absolutely required to synthesize the key.

Unfortunately, in the majority of these implementations, what the researchers found was that the key was not bound to the password, but that it was protected from access by a password which, as we know, is not the same thing at all because it opens that drive to a full password bypass. This is what the researchers were often able to achieve and demonstrate. So in other words, the drive's master key existed, hidden somewhere on the drive - in its firmware, on internal or external nonvolatile memory, somewhere. And the externally supplied password was being used, not to create the key, but to unlock access to it, which is like, okay, how do you explain that in this day and age? It just must be that somebody who doesn't understand security has been given the task of implementing security. Which is never going to be a good idea.

So what were the designers thinking, if they were? The only way to explain their thinking is that they must have believed that there would never be any way to reverse engineer their implementation. Assuming - they had to assume - that their code was, first of all, error free; and that the only access anyone would ever have would be through the front door by powering up the drive and accessing its standard ATA command interface.

And speaking for a moment about the issue of error-free code, in their description of the reverse engineering of the very popular Samsung 840 EVO drive, the researchers note of some of the drive's power-up logic, they wrote: "The key is computed" - this is just one of many keys that they talk about, not any particular super-secret one. "The key is computed during the drive's boot-up sequence. However, due to a bug in the firmware, retrieving slot 451 during early boot fails. Therefore, the value of 'p,' a value being hashed, is in fact a zero buffer. Consequently, the resulting key is constant for all devices."

That's just an aside, the point being that some piece of logic that was also crucial to the drive's security is completely dysfunctional in the firmware and never has been protecting the drive, and no one apparently ever saw that before or noticed it. So not only is the fundamental design of the drives flawed, but the flawed design is buggy, so it's not working the way - doesn't even do what its designers intended.

Okay. But back to the topic of keeping secrets inside the drive. Our intrepid security researchers took advantage of two things these designers apparently failed to consider, firmware downloads and the microcontroller JTAG debugging interfaces. Okay, so first, on the issue of the firmware, most of these devices, in fact all of them, have readily

downloadable firmware. In some cases the firmware was not published, but it was downloadable nonetheless.

In some cases it's a bootable ISO image incorporating an OS kernel such as a small Linux and the firmware. This can be readily reverse engineered, even when the firmware image which is bound into the ISO image has been obfuscated. And this, of course, as we talk about decrypting the DVD on our living room DVD player, you've got to be able to decrypt it in order to see the movie. So it's easy to get access to the decrypted content. So even if the firmware image has been hidden, there's no way for something to successfully prevent it from being obtained.

In other cases, a dedicated firmware utility is provided; but it, too, can be readily reverse engineered to obtain and then reverse engineer the drive's controller code. And even if neither of those avenues was available, the interface to the drive could be intercepted and the update data captured. This was actually done in the case of updating the firmware of two of the USB-connected external drives. Wireshark now has a USB packet capture facility, and so that was used in order to gain access to the firmware. Oh, and all of the drives use the industry-standard ATA "upload firmware" command, making it easy to capture the firmware passing by. So one way or another, once the drive's - and I've got air quotes here - "secret" internal operation has been exposed, the location of its secrets can be found.

You know, this podcast has spent a great deal of time talking about the keeping of secrets. So we know that the greatest breakthrough which occurred in cryptographic thinking was the innovation of a keyed cipher. Before that innovation, cryptographers created clever, unkeyed ciphers. But if that cipher's operation ever became known, then every secret it had ever been used to protect would simultaneously be divulged. Cryptography's greatest innovation, then, was the concept of a keyed cipher where its algorithm could be, and absolutely should be, freely published and studied; and where the secret that must be kept was not the algorithm, but only a tiny key. Essentially, the use of a key created a near infinity of individual specific ciphers from a single master general cipher.

So the analogy here is to the erroneous thinking that apparently went into the design of these self-encrypting SSDs. Their designers must have believed that they could keep their internal algorithm secret, or perhaps that no one would ever bother looking because such secrets we know are impossible to keep. And beyond their exposure of their own firmware was the presence of their microcontroller's serial JTAG interface. We've touched on this in years past. JTAG is an industry standard interface which uses only a few wires and thus only consumes a few of a processor's or microcontroller's pins. It is universally supported in some fashion by all microcontrollers.

The JTAG interface allows the microcontroller to be placed under the control of an external debugger, and through the JTAG interface it's possible to examine the processor registers and its memory. The JTAG interface can be used to write and execute code on the fly in RAM. It gives anyone who can connect total control over the microcontroller. And in their paper they show the PC board layouts of these SSD drives, and where they find a layout of the standard JTAG connector pins. So it's like it's trivial to hook up a debugger to the microcontrollers, which are standard ARM cores, so they're understandable. Once you have a JTAG interface, you suck the firmware right out of the chip and then run it through IDA in order to decompile it and reverse engineer it.

So the JTAG interface can usually be disabled after initial testing and manufacture. Many microcontrollers have a one-time, a one-way fusible link, basically a fuse that is blown which then forever disables the microcontroller's JTAG interface. And this is done, this is present and done exactly because it is such a powerful and official standards-based backdoor into the operation of every microcontroller. Yet only two out of the seven SSDs

the researchers examined had their JTAG interfaces disabled. The rest freely accepted JTAG connections. And there's even something called the "JTAGulator" which is a piece of freeware, an open source platform that automates the process of figuring out which are the JTAG pins on a chip when you don't already know. So, I mean, the whole thing is just amazing that this is sitting there present in an SSD, pretending to be doing encryption for us.

So in one typical example where the model of a device's firmware was not available for download, its ATA command table could be located through its still functional JTAG interface, which allowed the researchers to learn of an undocumented vendor-specific ATA instruction and the secret parameters it required to unlock that command, which then commanded the drive to export its own firmware through the ATA interface. And then, of course, the drive's secrets were available.

So anyway, I'm not going to delve in detail into the individual fundamental design mistakes which were discovered in each of the seven drives which these guys dissected. I've got a link in the show notes for anyone who is interested in their very nicely assembled 16-page research disclosure. But they did have some interesting things to say about the question of hardware versus software solutions and some salient recommendations which were driven by their discoveries.

So at one point in their paper they write: "An argument that is often put forward in favor of hardware encryption is that the secret key is not stored in RAM" - that is, main memory, main host accessible RAM - "and therefore is not vulnerable to the aforementioned attacks. In reality, this argument is invalid for several reasons.

"First, the software running on the host PC controlling the hardware encryption typically does keep a secret key in RAM, introducing the same vulnerability. The reason is to support Suspend-to-RAM, a low-power state wherein all peripheral devices are shut down. Since the SSD is completely powered down, it must be unlocked again once the system is resumed, and therefore either the operating system must retain a copy of the drive's secret key at all times, or the user must enter it again. In virtually all implementations, including BitLocker, the former is chosen." Meaning the user is not hassled to re-unlock their drive. It's oh, look. It comes out of sleep, it wakes up, and everything works. Well, that means that the drive has been given the secret again in order to re-unlock it. So it's not any more secure.

"Two, the burden of keeping the secret is moved to the SSD, not eliminated. The SSD typically keeps the key in the main memory of its controller. SSDs" - as we know - "are not security-hardened devices by any standard," they write. "In fact, many have a debugging interface exposed on their PCB" - that's the JTAG I was mentioning - "allowing one to attach a debugging device and extract the secret key from the drive. Furthermore," they write, "several means of obtaining code execution on the drive exist."

And, finally, three: "A memory readout attack against software encryption requires physical access. Given this, the attacker also has the opportunity to carry out a hot-plugging attack against hardware encryption. This has been demonstrated in practice and poses a realistic threat."

So in their discussion at the end of the paper they wrap it up by saying: "An overview of possible flaws in hardware-based full-disk encryption was given. We have analyzed the hardware full-disk encryption of several SSDs by reverse engineering their firmware, with focus on these flaws. The analysis uncovers a pattern of critical issues across vendors. For multiple models, it is possible to bypass the encryption entirely, allowing for a complete recovery of the data without any knowledge of passwords or keys.

"The situation is worsened by the delegation of encryption to the drive, if the drive supports Trusted Computing Group Opal as done by BitLocker. In such cases, BitLocker disables the software encryption, relying fully on the hardware encryption." What did I do? I just lost my place. Oh. "As this is the default policy, many BitLocker users are unintentionally using hardware encryption, exposing them to the same threats."

Anyway, so this goes on. Basically what we now know is for these drives which were examined, the people who implemented the system used the password to gain access to the drive's encryption key, rather than using information from the password to uniquely synthesize the key. That is the only way to do this, to solve this problem securely. Again, this is not rocket science. This is just proper security design.

Okay. So where does this leave us? I would feel better using one of these drives once its firmware has been audited and fixed than any other drive that has not yet been scrutinized like this. And all of the drives which were subjected to this scrutiny by these engineers have now been updated. So if you have a Crucial MX100, MX200, or MX300, a Samsung 840 EVO or 850 EVO, or an external Samsung T3 or T5, significantly more secure firmware awaits you. Go update your drive.

And I would think at that point, I mean, again, I don't know - we're living in a world where we have to take our manufacturer's word for security until some auditor is given a chance to look at it. In fact, what these guys wrote was: "Hardware encryption currently comes with the drawback of having to rely on proprietary, non-public, hard-to-audit crypto schemes designed by their manufacturers. Correctly implementing disk encryption is hard, and the consequences of making mistakes are often catastrophic. For this reason, implementations should be audited and subject to as much public scrutiny as possible." Amen. "Manufacturers that take security seriously should publish their crypto schemes and corresponding code so that security claims can be independently verified." Why not?

They said: "A pattern of critical issues across vendors indicates that the issues are not incidental, but structural; and that we should critically assess whether this process of standards engineering actually benefits security; and, if not, how it could be improved. The complexity of TCG Opal contributes to the difficulty of implementing the cryptography in self-encrypting drives. From a security perspective, standards should favor simplicity over a high number of features. The requirements as specified by the Opal standard, having a many-to-many relation between passwords and keys, and allowing for multiple independent ranges with adjustable bounds, makes it very hard to implement it correctly.

"Finally, TCG should publish a reference implementation of Opal to aid developers. This reference implementation should also be made available for public scrutiny. It should take into account that wear leveling is applied for nonvolatile storage. Opal's compliance tests should cover the implementation of the cryptography, and these tests should be independently assessed."

I forgot, part of what I skipped in the details was that in one case a drive's secrets, about every 20 times the password was changed, wear leveling caused the previous instance to be left unencrypted, and a new sector was assigned for the new security. And it made it possible at the chip level to go read the unencrypted original key from the nonvolatile memory. So in this instance wear leveling was a confounder to the attempt to overwrite the unencrypted data with something that was then encrypted. And in this case it was the master key upon which all the other security of the drive depended.

So Microsoft has responded to this news. They have an advisory, 180028, titled "Guidance for Configuring BitLocker to Enforce Software Encryption." Anybody who is using Windows - I'm thinking it's only available in Windows 8 and later, so once again we

Windows 7 users don't have access to this. It reads: "Microsoft is aware of reports of vulnerabilities in the hardware encryption of certain self-encrypting drives. Customers concerned about this issue should consider using the software-only encryption provided by BitLocker drive encryption. On Windows computers with self-encrypting drives, BitLocker drive encryption manages encryption and will use hardware encryption by default. Administrators who want to force software encryption on computers" - well, and then users - "with self-encrypting drives can accomplish this by deploying a group policy to override the default behavior. Windows will consult group policy to enforce software encryption only at the time of enabling BitLocker."

Okay. So there is a command we all have in our machines. I tried it on Windows 7 just for grins. It's manage-bde.exe. That is, manage BitLocker drive encryption dot exe, manage-bde.exe. If you just type that by itself, you need to use an elevated command prompt. So right-click on command prompt, then run as administrator, then type manage-bde. Hit enter. That'll give you a whole list of switches which that command can accept. One of them is status, so manage-bde -status. And that will show you all the drives in your system which are usable with BitLocker and their current status, whether or not they are being encrypted, but whether or not BitLocker is enabled at all on each drive. And, if so, whether it is encrypted with hardware or software.

If it is encrypted under hardware, as we now know, that may not be very good encryption. So what you need to do, if you are interested, is first turn off BitLocker on any drive where you wish to switch it to software encryption. You have to turn it off first. Then you go into gpedit.msc. And I have a picture at the last page of this week's show notes showing, because I did this this morning, opened up the Local Group Policy Editor. I had to do it on Windows 10 because it's not available, this particular option is not available under Windows 7.

And so you follow this under - it's Local Computer Policy > Computer Configuration > Administrative Templates > Windows Components > BitLocker Drive Encryption, and under that is Fixed Data Drives. It's also available for removable data drives. And under there is "Configure use of hardware-based encryption for fixed data drives." And you are there. You'll only find that if you have Windows 8 or later. And you can choose the setting there to turn off, that is, disable hardware-based encryption for fixed-data drives. Then reenable BitLocker, and that will turn on software-based BitLocker encryption for your system. So again, jumping through some hoops.

And at this point I was trying to decide whether it would be safe to say that this requires local physical access. But it's probably the case, with the information that has been published, that these drives could have their firmware exported only using the ATA interface, which is available from the host. So you do not need the JTAG interface. So I would argue that, if somebody was determined, they could probably gain access to the drive's key, which by the way these guys properly note is available in RAM. Just because it's in hardware doesn't mean the software key is not also available in RAM. It actually is under BitLocker in Windows, even if you're using hardware encryption.

And also it's been noted that all the new processors, all of the recent Intel processors have the AES-NI extension which is specific instructions to accelerate the use of the AES Rijndael cipher in software so that it is not producing any slowdown. And our longtime listeners will remember that I benchmarked TrueCrypt and VeraCrypt, and I skipped over where these guys mentioned VeraCrypt. They do that a number of times in their paper. They're big fans of it, and they like it because, as we know, it's open source and has been audited. And as far as we know, there are no known vulnerabilities in it, unlike this disaster where drives are saying, oh, yeah, don't worry, your data is safe and encrypted, when in fact it is anything but.

So that's the story on self-decrypting drives. They are, unfortunately, they use the password to unhide the master key, rather than depending upon the password to synthesize the master key. There's a big difference.

Leo: Yeah. What can we do about it? Use VeraCrypt.

Steve: Yeah. Well, as far as we know, if you have access to Windows and BitLocker, and you'd rather not use a third party, you can disable its use of hardware encryption and then reencrypt your drive under BitLocker, having changed that setting, as long as you have Windows 8. If you're using Windows 7, then your only recourse is to disable BitLocker and install VeraCrypt. It's good. And as we know, it does not measurably slow things down. I was unable to detect any benchmark slowdown of the system under TrueCrypt and/or VeraCrypt. So there doesn't seem to be any penalty in performance these days.

Leo: Good. And you can disable BitLocker's use of hardware encryption with a command line; right? I mean, there's no easier way.

Steve: A command line, yes, yes.

Leo: A quick google will find that.

Steve: Yup. And we have complete documentation in this week's show notes.

Leo: Yeah. So you put up the show notes at GRC.com along with the audio of the show and the transcript.

Steve: Yeah.

Leo: So really that's the definitive place to go: GRC.com. While you're there, get SpinRite, the world's best hard drive recovery and maintenance utility. You should also check out all the other cool free stuff Steve offers at GRC.com. That's his website. He's on the Twitter at @SGgrc, @SGgrc. You can leave him messages there. He accepts direct messaging.

We also have audio and video versions of the show at TWiT.tv/sn. You can watch us do it live every Tuesday, 1:30 Pacific, 4:30 Eastern, 21:30 UTC. Tune in at TWiT.tv/live. Join the chatroom at irc.twit.tv. Chat along with us as we do the show. The chatroom always posts the show notes link there, as well, so you can get that and read along, too.

But you can also get on-demand versions, as I said, from Steve's site, our site, or subscribe. Please subscribe. That way you'll have a collection. Start your collection now. Got to catch them all. GRC.com. TWiT.tv/sn. That's all I need to say except thank you, Steve, and we'll see you next time on Security Now!.

Steve: Yay. Thanks, my friend. See you next week.