Hacking Certificates

Description: Leo and I catch up with another packed week of security news, including an update on mobile ransomware; the successful extraction of Android's full disk encryption (FDE) master keys; Google's Tavis Ormandy finds horrific flaws in all Symantec traffic analyzing software; a Brazilian judge is at it again with WhatsApp; this week's IoT horror story; some miscellany and errata; and, finally, a look at a horribly flawed attempt to copy Let's Encrypt automation of free SSL certificate issuance.

High quality (64 kbps) mp3 audio file URL: http://media.GRC.com/sn/SN-567.mp3
Quarter size (16 kbps) mp3 audio file URL: http://media.GRC.com/sn/sn-567-lq.mp3

SHOW TEASE: It's time for Security Now!. Steve Gibson is here. Lots of security news. We'll talk about an amazing Amazon review and a response from the company, yet another IoT disaster, and how not to issue automated security certificates. It's all coming up next on Security Now!.

Leo Laporte: This is Security Now! with Steve Gibson, Episode 567, recorded Tuesday, July 5th, 2016: Hacking Certificates.

It's time for Security Now!, the show where we cover the latest security news with this man, the security in chief guy. I just blew your title. I used to call you the Explainer in Chief, but that doesn't really do it. What are we going to call you? The man in charge of Security Now!, Steve Gibson is here. Hi, Steve.

Steve Gibson: The man who named it.

Leo: The man who named it.

Steve: When you said, "How about we do a weekly security podcast?" And I said, "A what cast?" And you said, "You've never heard of podcasts?" And I said, "Uh, no."

Leo: To be fair, this was in 2005, when nobody had heard of podcasts.

Steve: Now I've got awards on my shelf from podcasts.

Leo: There you go. And did you come up with the name Security Now!?

Steve: Yeah. I suggested Security Now!.

Leo: I like it.

Steve: So we're at Episode 567 as we close in on the end of Year 11 of this weekly podcast, with a bunch of interesting news. We've got an interesting update from Kaspersky. It might be a little skewed because they provide some software for protecting Android phones. But the statistics and their information about what's going on with ransomware on Android phones is interesting. And it turns out it's growing rapidly. We also have the news that was much tweeted this week of a security researcher successfully extracting the full disk encryption keys from his Android phone.

Our friend Tavis Ormandy at Google's Project Zero really took Symantec/Norton down by taking a look inside of a core of pretty much all of their traffic inspection applications. They've got enterprise and consumer. All of this stuff that will protect you from stuff on the wire, turns out they really - in fact, the mistakes were so bad in specifics that you can't really forgive them because, for example, in some cases they used open source software that has seven-year-old vulnerabilities they didn't fix.

Leo: Oh, my god.

Steve: So how do you explain that away? Brazil is at it again with WhatsApp. Boy, those Brazilians really have a thing for WhatsApp. Then we raise the question, what if you forget to reregister your router's setup domain name, as TP-Link did. We've got, of course, this week's IoT horror story. And I have to tell you about the one from two weeks ago that I shared with Padre when you were in Newport Beach because we didn't talk about it last week.

Leo: No, we didn't, yeah.

Steve: Some errata and miscellany. And then unfortunately the company StartCom, that does the StartSSL certs, decided that they liked what Let's Encrypt had done, and they were going to do it, too. Unfortunately, they unbelievably badly fumbled their execution. And when you think about it, automated certificate issuance is something you have to be really careful about because certificates are powerful, and you just don't want anybody to be able to get a certificate for pretty much any domain they want. But it turns out you could.

Leo: Oh.

Steve: So lots of fun stuff for the podcast this week.

Leo: Not good. Not good at all. Okay, Steve.

Steve: So Kaspersky took a look at what they're seeing going on in the Android world. They have essentially instrumentation because they produce a mobile AV product that gives them some visibility into what's happening on the phones of their users. What they have seen - and we've never talked about this before. We've obviously talked about the Android problems with the media server module. And later we're going to come across, in fact it's in the Symantec story, one of the points that I made some months ago was how difficult it is for an interpreter to be careful about what it's interpreting, for example, in the context of a JPEG. And of course this is relevant to the media server module because it's trying to interpret MP4s and various forms of media.

And the way this is done is that there's sort of a meta language that is the file itself, the JPEG file, or the MP4. It's not just pure data. It's tags that represent pieces of things. And essentially an interpreter reads that, block by block, and does whatever it's being told to do. And so what we've seen, for example, over on the media server side on Android, is it's so riddled with bugs that we just keep getting more of them.

So, but what's been interesting is that people are saying, yeah, okay, but Android phones don't seem to be infected. I mean, like, no one's taking advantage of that. Of course, there was the concern when you could just receive an MMS message and get your phone taken over. The potential for remote exploitation was so high there that we all jumped to fix that as quickly as possible. But what Kaspersky is seeing is that mobile ransomware affected 136K of Android customers by April of 2016, so a few months ago, and that that was nearly quadrupled from March of 2015. So it looks like, unfortunately, it's a growth segment. And as I said, they have a solution potentially for this, so there's a little bit of self-promotion probably going on here.

But the statistics and details are interesting. They said one of the things that's interesting is that the strategy is different for mobile versus desktop. On a desktop, the ransomware favors file encryption because you typically have high-value irreplaceable files sort of inherently in a desktop setting, much more than a mobile setting. On a mobile device, it's songs that you have somewhere else. Also, the vendors are actively backing phones up to the cloud so that, if your phone got encrypted, it's like, okay, fine, just do a hard reset and restore from the cloud, and you're back to the point where you had your last backup.

So the point is that on mobile devices, rather than encrypting files, what they're seeing is blocking, what they call "blocker-style ransomware." And when I saw the pictures in their blog posting about this, they sort of looked familiar, like I've run across them just sort of in passing, but didn't really give them much thought. But they come up and pretend to be, for example, the NSA or some government agency.

Quoting from Kaspersky, they said, "Blockers are the much more popular means to infect Android devices. On mobiles, they act simply by overlaying the user interface of every app with their own, so a victim can't use any application at all. PC owners can get rid of a blocker with relative ease. All they need to do is remove the hard drive, plug it into another computer" - not that that's easy for most people, but it is a workaround - "and wipe out the blocker's files." Of course, that's sort of speaking from Kaspersky's standpoint. But they write: "But you can't simply remove the main storage from your phone. It's soldered into the motherboard. That explains why blockers hold 99% of the mobile ransomware 'market,'" in quotes, such as it is.

In 2014 through 2015 there were four main actors in the mobile ransomware space that

dominated. There's something called S-V-P-E-N-G, P-L-E-T-O-R, Small, and Fusob. Wonder what that's an acronym for, or an abbreviation for.

Leo: Fusob, you know.

Steve: Fusob, right. "At this time Pletor," they said, "has almost stopped its expansion. Its creators have launched an infamous Acecard trojan and seem to be pouring their resources into developing and spreading that instead." Oh, so their attention kind of got diverted by a different trojan they're working on. "The developers of Svpeng have also refocused, mostly on the banking version of the trojan."

Leo: Well, you've got to go where the money is.

Steve: Yeah. "So that leaves only two big mobile ransomware families, Small and Fusob. As a result, in the 2015 to 2016 timeframe, these two trojans represent more than 93% of the mobile ransomware space." And what's interesting is that they have a lot in common, but their territories deliberately don't overlap, which is sort of suspicious. Kaspersky says: "It's noteworthy that the Fusob and Small trojan families have a lot in common. Both display fake screens that pretend to be signed by authorities and accuse the victims of misdemeanor. Both say that a criminal case will be opened unless the user pays the fine." Which, you know...

Leo: This cracks me up.

Steve: Yeah, it does. A sophisticated user is like...

Leo: If you just pay us 300 bucks, we won't have to send you to jail.

Steve: Yeah. Pay $300 to the NSA, and we'll just give you a slap on the screen and let you go about your business, and don't do that again. So they say both Fusob and Small also offer rather strange ways to pay the ransom. This Fusob suggests payment with iTunes gift cards, which apparently provide sufficient anonymization.

Leo: Yeah, but you don't turn those into - maybe you can turn them into cash.

Steve: Or maybe the guys that did Fusob just really like iTunes stuff, and they want to buy some goodies.

Leo: Some apps, yeah.

Steve: Yeah. Whereas Small offers victims the option of paying by the Kiwi payment system or the MoneyPak xpress packet vouchers. And they write: "Both were probably created by some Russian-speaking groups of cybercriminals, but with very different

approaches."

Leo: Oh, the viruses. Because Kiwi and MoneyPak were not.

Steve: Right.

Leo: You can get a MoneyPak at 7-Eleven.

Steve: And then they talk about - and remember there was - in fact it was MoneyPak at 7-Eleven ran out because in early days...

Leo: Right, right.

Steve: Remember that?

Leo: Yeah.

Steve: In the early days of CryptoLocker, that was the way you paid for CryptoLocker before bitcoin. And everyone was running to 7-Eleven to buy MoneyPaks. And the guy at 7-Eleven said, "I don't know what's going on, but the J-hooks are all empty."

Leo: We've got a real run on these things.

Steve: So this Fusob detects the device's language. And if it's one of the post-Soviet Republic languages, Fusob does nothing. Otherwise, it shows a screen that claims to come from the NSA and demands ransom from 100 to $200. The majority of Fusob's victims, more than 41%, live in Germany. So that seems to be the main focus. Then second and third are the United Kingdom and the U.S., with 14.5 and 11.4%, respectively.

Leo: But if you live in Lithuania, they go, "Oh, never mind." That's so strange.

Steve: Well, that's where - although Small - and that's what's different. The Small family, almost 99% of Small's victims are located in the three countries that Fusob avoids - Russia, Kazakhstan, and Ukraine. The Small ransomware shows a government-themed screen with payment instructions, threats, and a demand for, well, it's in rubles, from 700 to 3,500, which is roughly between $10 and $50 U.S., to unlock the infected device. And then there is an English locale version Small also, but it has a different block screen, of course, and mentions the FBI, and demands a lot more money, $300, because we rich Americans with our expensive phones, we can afford more, apparently.

So anyway, I thought that was interesting. They wrap up with a what-to-do. And they said, and of course this is strongly good advice, install applications only from official

shops such as Google Play. To be sure that no application makes its way onto your device from an untrusted source, go to Android settings, choose security, and make sure that the unknown sources box is not checked. And of course I'm sure it defaults to having unknown sources disabled. And so the user would have to deliberately enable it if they wanted to get something. I actually did that for the Zeo app when it was first made available because I just needed to get it from wherever it was, but then I turned it off afterwards because, yeah.

Leo: Typically now, anyway, what happens is when you sideload an app it says, okay, we're going to enable this - after lots of warnings - but only for this one download.

Steve: Nice.

Leo: And then it turns back on protection. So, yeah, I think they're very aware of the risks involved.

Steve: And then of course the standard advice from us and also from Kaspersky: "Regularly update your device's firmware and its installed apps. You can choose to update apps automatically, but you still need to have the system updated manually. And it's better to do that," they write, "as soon as an over-the-air update arrives." And then of course Kaspersky says: "Install a strong security solution." And we're sort of out of the mainstream.

Leo: Yeah, that's where it kind of bugs me because the company that makes the solution...

Steve: Yes, exactly. And Leo, wait till you hear what Symantec did. Oh. Anyway, so yeah, it's called you have to be very careful, or you create a whole new attack surface. Because if something is sitting there inspecting what's coming in, and it's got any problems...

Leo: Right, right. I don't think there's any evidence that you need antivirus on mobile platforms. Just, you know...

Steve: I think you're right. To me that seems like a stretch too far. I mean, most people, you know, I'm running whatever that Microsoft thing is they keep changing the name of. Sometimes it's Windows Defender, sometimes it's the little green house icon.

Leo: But it's different because on mobile everything is kind of isolated, certainly on the iPhone. An antivirus can't do any of the things it would do on a desktop on an iPhone. All it could do - and in fact, as far as I know, it's the same thing on Android, all they do is scan downloads before you install it, and Google's already doing that on Android.

Steve: Right, right.

Leo: So that's a little self-serving on their part, I think.

Steve: Yeah. So this got a lot of forwards to me. A guy who took a good hard long look at Android's full disk encryption, so-called FDE, successfully extracted the keys from his device. And this was a, I don't remember now which version, but Qualcomm chipset. And Qualcomm goes to some lengths to protect this, as you would expect. But it turns out it's insufficient.

So let's first recall how Apple's full disk encryption works. And that leads us to why it is so secure. In the factory, there is a 256-bit userID, a unique ID, a UID, which is burned into the hardware. And so it's contained. It's write once, and it's read never. You cannot read that. Apple doesn't know what it is. There's literally, from any API, no way to get the key. Every device has one. Every device is unique. And there's no way to know what it is. The only thing you can do is give it work to do, give this secure enclave a blob to encrypt or something to sign, and it'll do it, and it gives you the result. But it does that without ever exposing the key, and it cannot be removed. The user's password is - and here's this weird word thy invented that everyone in the security industry puts in quotes because it's like, come on, really? And that's - remember this, Leo? - "tangled."

Leo: Tangled, yeah.

Steve: They "tangle" the user's password in some undocumented way as part of this, so that the device by itself has no ability to do decryption. The user has to provide the password. It is then entangled - sounds like some quantum entanglement thing. It's then entangled with the secret that's in the chip. And those two things, something that cannot be removed and cannot be exposed, but can only be used, is required, plus something that the user knows or provides is the other part of that. And short either of those two things, nothing's happening.

And then, on top of all that, there is a hardware-enforced 80-millisecond overhead. And this is not short-circuitable. It's that it takes the hardware 80 milliseconds to do whatever it's doing. So there's probably a deliberately complex password-based key derivation function, a PBKDF, which just takes 80 milliseconds. Can't make it any quicker because it's got a lot of work to do during that time. So it's not like a software delay that's been imposed, you can just code out that delay loop. It's actually doing work that has to be done in order to derive the proper intermediate key, given the unique ID and the user's password.

So now Android. Starting with v5, as we talked about back then, Android devices default to protecting all of the user's information by enabling full disk encryption. And remember that, in the beginning, there was hardware acceleration for that which paradoxically was not being used. And as a consequence, all of the encryption as it writes and decryption as it reads was being done in software. And unfortunately, that created enough overhead that people were turning it off because their phone was too slow with it turned on. But ultimately the hardware that was there from the beginning to accelerate this got engaged, got implemented. And then background full disk encryption became the default.

Now, the good news is that Android's FDE, full disk encryption, is based upon a time-tested, well-known Linux kernel subsystem called "dm-crypt." So this is good news

because it's not something that anyone, some Qualcomm engineer or somebody else jut sort of home brewed. It's widely deployed and researched and pounded on and done right. However, as we know, an encryption system implementation may have no weaknesses, but the vulnerability can still be in its key management. That's the hard part. Getting the data encrypted securely, we pretty much know how to do that. And the algorithm's not secret. It's a Linux kernel subsystem. It is open source. Everybody can go take a look at it.

But as with any good crypto, the secret is not in the algorithm, it's in the key. And so what research has shown is that probably due to the comparative clunky keyboard of mobile devices, compared to a nice physical keyboard on a desktop, mobile users tend to choose easier-to-enter, lower entropy passwords. So of course that comes back a little bit later, if something makes this brute-forceable. And that's essentially what happened is Android implements, it's got this notion of a TrustZone with a secure secondary processor.

The problem is that there is an API that's documented, and "trustlets," as they are called, little trusted applets, trustlets, can be loaded into this processor's execution space. And the code in the TrustZone can be extracted and reverse-engineered, and has been. Consequently, several exploits have been found which break the TrustZone's trust.

Now, when this news came out, Google was quick to point out that this researcher had leveraged two known vulnerabilities, the first of which was fixed in January of this year, and the second of which was fixed just over a month ago, in May. So still pretty fresh. And an analysis, oh, I don't remember now who did the analysis. But it turns out that analysis of enterprise-deployed Android devices showed that, despite the fact that both patches existed, 37%, more than a third of audited enterprise Android devices remain vulnerable, meaning that they can, right now, using this system, which has been fully disclosed, the guy's got code on GitHub, a beautiful blog posting. If anyone's interested in the intimate details of this, it's just a very nice expos of how he pulled this off. But basically it allows any of these devices to have their full disk encryption keys extracted.

So this researcher did pull the keys, and then wrote some software to do a simple brute-force attack. Now, in his example the attack was - the password was simple, so it found it like on the third guess. But the problem is we know exactly what the algorithm is. In fact, right there, Leo, where you are, that's this week's Picture of the Week. That is the rather complex key flow for the Android full disk encryption with the TrustZone and everything.

Leo: This is their entanglement here.

Steve: Exactly. That's their version of tangling the password and the secrets. And essentially the secrets are stored out in the nonsecure processor in encrypted form, and then they are provided to the secure processor and decrypted when needed. And so what's different is it is truly impossible to get the key off of an Apple device, an Apple mobile platform, onto a roomful of GPU-accelerated, special-purpose hardware to then brute-force the key. That can be done on Android devices that have not had these two patches applied. And the problem is it's never going to be as secure as an implementation where the hardware simply won't give it up.

And in fact in the comments at the end of that blog post, the author of Hashcat, who we've spoken of before, Jens Steube, has offered to accelerate the cracking, that is, to implement that crazy algorithm or what of it you need in order to do brute-forcing in his

very fast and increasingly powerful Hashcat system. So the moral of all...

Leo: But let me ask you, because I read this also.

Steve: Yeah.

Leo: The implication is it's because Qualcomm is not doing its enclave properly; right?

Steve: Right. I was just going to say...

Leo: If the phone didn't use Qualcomm, I mean, in other words, this isn't an Android issue, it's a Qualcomm issue.

Steve: Correct. Well, yes. And so it's not - if it didn't use Qualcomm - okay. So the moral I was getting to is...

Leo: Oh, I'm sorry.

Steve: ...making things difficult is...

Leo: Is difficult.

Steve: Is no longer good enough.

Leo: Yeah, right.

Steve: They need to be impossible.

Leo: Right.

Steve: And so, I mean, Qualcomm, you look at that flowchart, and your eyes cross. I mean, I spent some time with it. It's like, okay, yeah, it all makes sense, and I can see why they're doing what they're doing. But they just didn't go far enough. The little bit of gray area got taken advantage of. It has to be impossible. And Apple understood that and invested in the hardware to do it. I imagine, I mean, it's not like there's anything special or that that's hard to do. Essentially, for example, that's what the Yubico goodies are. They've got the key inside, and they never give it up. It is a write-once secret, and then it just uses that. So it's not hard to do. Qualcomm just didn't do it.

I think they got overly complicated. The idea of having trustlets, when I read that third

parties could load trustlets into the secure processor, it's like, no, no, no. Take that bullet point off of the data sheet and lock this sucker down so that nothing can get the key out of the hardware. So I think it's a problem that they're storing the key in encrypted form outside. You just don't want to do that. You want it to be absolutely inaccessible. Period. And that isn't this architecture. So this is an architectural problem with this implementation. But I'm glad you clarified, Leo. You're right, it's nothing to do with Android, per se. It's this particular implementation of the security architecture.

Leo: Now, almost all Android phones in the U.S., in fact I would guess almost all, use Qualcomm chipsets for the radios. But interestingly, some Samsung phones do not. They use Samsung's own Exynos chipsets. And I'm wondering, in that case, if there's a vulnerability.

Steve: Good question. Good question. Where the secret is.

Leo: Yeah. Well, the idea is it's a secure enclave la Apple; right? Implemented by the chipset manufacturer. Is that right?

Steve: Correct.

Leo: Yeah.

Steve: Correct.

Leo: So it's just not well implemented by Qualcomm. It could be implemented better; yes?

Steve: Absolutely. Next version, it's like there's no reason that they wouldn't.

Leo: So it's not a flaw inherent in Android. It's a flaw in, like, the vast majority of Android hardware here in the U.S.

Steve: It's a flaw in the hardware, basically. They just didn't do the hardware right. They got a little too fancy. And for whatever reason they made the key live out, I mean, they may have a reason. They've had no opportunity to defend themselves. There may be, like, some, I don't know what it would be, because I'd like to have an absolutely unextractable secret that binds this to the phone. That's the difference. If you can get that off the phone, then you can pour a roomful of Hashcat-driven GPUs on it. And then, if you couple that with a weak password because users tend not to do fancy passwords on their touchscreens, then you've got the double whammy.

Leo: You've got to feel for Google a little bit. See, unlike Apple, which controls all this, Google's at the mercy of OEMs who are making Android hardware. And it may

be one of the reasons, one of the rumors that's been strong this week is that Google is going to do their own phone. And it maybe one of the - not a Nexus phone, which is manufactured by another company for Google, kind of with Google's suggestions, but their own phone. And that could be one of the reasons they want to do it. I wouldn't be surprised.

Steve: I think that would be good. I mean, we know...

Leo: Then if they blow it, well, we can blame Google.

Steve: Well, and I doubt they would. We know...

Leo: Yeah, they know what they're doing.

Steve: ...Google understands security. I would argue they're at the top of the pile. I mean, Apple and Google, I mean, they're very visible. But I have no argument with Google understanding security, which is a perfect segue into Tavis Ormandy's latest walk through somebody else's product. So there was a lot of coverage of this. Oh, and I forgot to mention, I did not, just for everyone who's wondering if I'm going to talk about the Lenovo problem, there was a lot of activity just - maybe I started seeing it yesterday, but a lot this morning, about some sort of a bad problem with Lenovo. I was going to cover it; but when I dug into it, it suddenly exploded because it's not just Lenovo, it's a whole bunch of other OEMs that made the mistake of using Intel's reference code for the system management, the SMM stuff. So I will explain it all next week. It got too big for me to get a grip on it today.

But anyway, this problem with Symantec, and both the Symantec label and the Norton label for consumers, Dan Goodin writing for Ars Technica said - I just love the beginning of this. He says: "Much of the product line from security firm Symantec contains a raft of vulnerabilities that expose millions of consumers, small businesses, and large organizations to self-replicating attacks that take complete control of their computers, a researcher warned Tuesday."

Now, I would only dial that back on "raft." That's really not a "raft" of vulnerabilities. It's...

Leo: It's a dinghy.

Steve: It is an interpreter in...

Leo: It's a small dinghy.

Steve: Okay, good, yeah. It's an interpreter in the kernel, and you just don't want interpreters in the kernel if you can avoid it. That's what Windows did with GDI, when they moved the graphics device interface into the kernel. And that's why a JPEG could

take over your computer. Okay. So Tavis is wonderful. He, with of course Google's Project Zero, is a cross-vendor, we're just going to look at things and see if we can find problems.

So Tavis produced a detailed blog post which begins: "Symantec" - as if we didn't know - "is a popular vendor in the enterprise security market. Their flagship product is Symantec Endpoint Protection. They sell various products using the same" - and here it is - "the same core engine in several markets, including a consumer version under the Norton brand. Today," writes Tavis, "we're publishing details of multiple critical vulnerabilities that we discovered, including many wormable remote code execution flaws." I mean, you just don't want those words to be put together next to each other.

Leo: Wormable execution flaws.

Steve: Wormable remote code execution flaws. Oh. Tavis writes: "These vulnerabilities are as bad as it gets. They don't require any user interaction, they affect the default configuration, and the software runs at the highest privilege levels possible. In certain cases on Windows, vulnerable code is even loaded into the kernel, resulting in remote kernel memory corruption."

Leo: Oh, god.

Steve: "As Symantec uses the same core engine across their entire product line, all Symantec and Norton branded antivirus products are affected by these vulnerabilities." And he enumerates just a few. He says: "Including Norton Security, Norton 360, and other legacy Norton products across all platforms," meaning Windows, Mac, Linux; "Symantec Endpoint Protection, all versions, all platforms; Symantec Email Security, all platforms; Symantec Protection Engine, all platforms; Symantec Protection for SharePoint Servers," and he says, "and all the rest, too." So it's like, oh, my goodness.

"Some of these products cannot be automatically updated," he writes, "and administrators must take immediate action to protect their networks. Symantec has published advisories for customers. In a coordinated disclosure, shortly before Tavis's post, Symantec issued its own advisory, which listed the 17 Symantec enterprise products and eight Norton consumer and small business products affected." And I have a link in the show notes for anyone who, like, basically, if you're using any Symantec AV, go update it, like, yesterday.

Tavis wrote: "Because Symantec" - and here it is - "uses a filter driver to intercept all system I/O" - I'll explain what that means in a second - "just emailing a file to a victim, or sending them a link to a file which is an exploit, is enough to trigger it. The victim does not need to open the file or interact with it in any way. Because no interaction is necessary to exploit it, this is a wormable vulnerability with potentially devastating consequences to Norton and Symantec customers. An attacker could easily compromise an entire enterprise fleet using a vulnerability like this. Network administrators should keep scenarios like this in mind when deciding to deploy antivirus. It's a significant tradeoff in terms of increasing the attack surface.

"The flaws reside in the engine the products use to reverse the compression malware developers use to conceal their malicious payloads. The unpackers work by parsing code contained in files before they're allowed to be downloaded or executed. Because

Symantec runs the unpackers directly in the operating system kernel, errors can allow attackers to gain complete control over the vulnerable machine." So Tavis wrote that one of the proof-of-concept exploits he devised works by exposing the unpacker to odd-sized records which cause inputs to be incorrectly rounded up, resulting in a buffer overflow.

Okay. So a filter driver is officially sanctioned hooks in Windows which allow a third party - Microsoft uses them, too - allows a third party to essentially insert a shim. The idea was that dealing with the intricacies of the network interface adapter, the NIC, it's incredibly detailed. You have to really know your stuff. And it's in a whole different API down in the kernel than it is up in the application space. So there's interrupts flying around. You have to be very careful not to hog too much CPU. I mean, this is in the kernel. This is the OS. And efficiency really matters.

So what Microsoft did is they decided, okay, we cannot have people writing their own device drivers, like, redundantly, because that'll be a disaster. So we'll provide those. But then we'll allow people who have a need, essentially, to layer their code between the device driver and the rest of the system, thus a shim. And that's what Symantec did. And it's one piece of code, one filter driver, which installing a Symantec AV product installs into the kernel. And what this does is inspect all of the traffic, at least inbound. I'm not sure if it's outbound also. But it inspects it on the fly.

And so one of the things that malware products do, actually one of the things that good products do, lots of products do it, is they compress their EXEs. All of my EXEs are compressed. Not because they would be huge otherwise, but because the EXE format, and I've grumbled about this in the past, the EXE format is incredibly inefficient. Huge tracts of lands. No, huge regions of null space. I mean, just huge. And it's, like, sitting there in the file. So the point is anything, an EXE compressor could examine this executable and say, what is this 12K of zeroes? That's dumb, so we're going to instead put a tag and say here lies, in the original file, 12K of zeroes, and then not have any. So you just saved yourself 12K. And so on. It's like that.

So but the point is that means that on the front of that executable needs to be a little interpreter which itself runs when the OS loads the executable. The interpreter gets control and reads this compressed EXE. And so, for example, it finds this little tag saying "Here lies 12K of nulls," and it says, "Oh," and so it expands that on the fly in memory back to the original 12K of empty space. But that means it's an interpreter. And this comes back to what we were talking about with the danger of running interpreters in the kernel. It's so easy to make a mistake. And Symantec did.

[Background "Yabba dabba doo"]

Steve: Oh.

Leo: That's not a mistake. No, no. That's a good thing.

Steve: One that I forgot to mute.

Leo: Oh, don't worry, I like it.

Steve: I go around the house and turn off all my iOS devices.

Leo: Steve, if I were you, I'd turn it up because you never know.

Steve: Well, it's little bit of a chorus, and it's sort of - because they all receive the...

Leo: Yabba dabba do at once?

Steve: Yeah, exactly.

Leo: We should mention that that sound occurs whenever somebody charges another copy of SpinRite to the old credit card.

Steve: I always appreciate it. It keeps the lights on. Even though one of these lights is flickering on and off.

Leo: It keeps Fred employed, as well.

Steve: So the 10,000-foot view is Microsoft has spent painful decades, and we've lived through them - Code Red, Nimda, MS Blast. How long, Leo, in the beginning of this podcast, were we bemoaning the fact that email was running scripts? Remember those days?

Leo: Yeah, ai ai ai.

Steve: Oh, my lord. It's like, turn that off. And they finally did, as we talked about last week. But it took only 20 years. So Microsoft has spent painful decades, carefully hardening the core Windows operating system. Then Symantec comes along and installs a bug-ridden, traffic-intercepting filter into the OS kernel, which completely compromises hard-won security.

And so, again, this is a classic attack surface vulnerability that we've been talking about. This is a problem where, with any add-on, we've seen them with other AVs and other vendors' add-ons, Lenovo of course is constantly in trouble for this, where an otherwise relatively secure and hardened operating system, Microsoft keeps talking about how Windows 10 is the most secure operating system ever, except, yeah, then people load a bunch of this stuff on it, and it's cheese, Swiss cheese again. So anyway, wow. And of course, Leo, you and I are careful and prefer to run adblockers, and we're careful about where we go and don't go on the Internet. And knock on wood - is there any wood around here? - don't get ourselves infected.

So these judges in Brazil just keep having a problem with the fact that WhatsApp is unable, I mean, actually architecturally designed to be unable to decrypt the messages that apparently drug people are using it to communicate with. So I don't know. Once again Brazil...

Leo: This is going to be a big problem. And it's not just for Facebook.

Steve: You're right. You're right.

Leo: If countries insist, I mean, obviously what they want to do is put pressure on WhatsApp to have a backdoor.

Steve: Correct. Correct.

Leo: But there's no response that WhatsApp can give them. They just say we can't give you...

Steve: No, there isn't.

Leo: Take our money, but we can't - do anything you want. Throw us in jail.

Steve: As we know, WhatsApp is designed correctly so that it is impossible for them to decrypt the messages. And twice there have been WhatsApp outages caused by judges commanding that it be taken offline. But I'm sort of stepping on my story here. Last Thursday a court in Brazil blocked a little over $6 million of Facebook's funds because of course Facebook owns WhatsApp. And apparently Facebook funds is what was available to be blocked. WhatsApp failed to turn over messages sought in a drug-related case. And as I was writing this, I was thinking, this actually is kind of pure gold advertising that Facebook couldn't buy.

Leo: Especially among the drug dealing public.

Steve: Yeah. It's worth way more than $6 million for Facebook for WhatsApp to have this reputation of being such a problem because, we're sorry, we would love to help you, but we can't decrypt the traffic. No matter what you do. So: "After repeated failure over five months to turn over this particular information for this particular drug-related case, Brazil's G1 news service reported that a judge froze the funds which are equal to WhatsApp's accumulated fines for non-compliance in the case. Because WhatsApp has no bank accounts in Brazil, the judge froze the funds owned by its parent, Facebook."

The Brazilian court, however, did not again use provisions of Brazil's Internet law that allows courts to shut down service in some cases of noncompliance with court orders. Of course, as we know, earlier this year a judge ordered a 72-hour shutdown of WhatsApp, which angered so many of the service's 100 million Brazilian users that the shutdown was lifted after just a day, after about 24 hours, by another court, who essentially overturned that judge's order.

So, yeah, as I was thinking about this, it's like, wow, I'm sure $6 million, to Facebook, okay, fine, take it, great story. We're happy that people are - we're not happy that drug dealers are covering their tracks using our technology, but for people who want privacy

and have a legitimate use for it, WhatsApp does the job.

So this was an interesting story. It sounds worse than it turns out to be. And that is that a very popular line of routers, SOHO, the consumer routers, TP-Link - and they have some small enterprise equipment also. TP-Link once upon a time used the domain that was printed on the label on the underside of the router: tplinklogin.net. And also in the manual. And at some point, for whatever reason, they switched to tplinkwifi.net. And so their newer products use tplinkwifi.net as the configuration domain. The older ones use tplinklogin. So what happened in the news is that they weren't paying attention, and the tplinklogin.net, the original, technically retired, but still printed on the label and in the manuals of the older devices, expired. And somebody grabbed it and is interested in selling it for $2.5 million.

Leo: Oh, boy.

Steve: Okay. Now the good news is our friend Michael Horowitz, who writes the Defensive Computing column for InfoWorld, he pays a lot of attention to router issues. And I've got a couple links to share here in a minute. But so Michael researched it further because it sounds like the end of the world, but it's not. Because what that label on the underside of the router does is it's regarded as easier to type that than 192.168.0.1 or .254 or whatever.

Leo: Linksys does this, Netgear does it, everybody does it.

Steve: Right. So the router itself sees the browser making a request for tplinkwifi.net, has a little mini DNS server and so returns the router's current gateway IP for that particular domain. Other ones it lets go through to real DNS servers. It just intercepts that one. And then that's an easy way, a shortcut of getting your browser to see the local router's configuration pages.

So what that means is anybody using it the way it was intended to be used, inside the LAN, behind the router, to get to the router, tplinklogin.net, the retired and no longer valid one, well, that router that has it printed on its label and in its manual, it's still going to take you to its own page. The only real danger is somebody outside of the LAN. I don't know why you would go to tplinklogin.net if you were not trying to get to your router from inside the LAN. So it's a little sloppy that they let that go. A company as big as TP-Link, I think this was just inadvertent. And the problem is now maybe $2.5 million would be galling to pay some domain squatter for the right to [crosstalk].

Leo: Yeah. So this is interesting because I always wondered how those redirects worked. It doesn't get to the public Internet at all. It just goes to - the router sees it and goes, oh, you mean me? Oh, yeah, I'm right here. So it doesn't matter. They don't even need the domain name.

Steve: No.

Leo: That makes sense.

Steve: So anyway, Michael Horowitz got one, had one, verified it, verified even that when you change, if you reconfigure the router so that it's got a different, like a weird gateway, like .9.9, it still redirects it to the current gateway IP, so it always works. It turns out that Michael operates a site, nice site called RouterSecurity.org that we've never talked about before. And his coverage of this, which I thought was excellent, took me to RouterSecurity.org for the first time. And I noticed that he strongly recommends that savvy users simply avoid all of the consumer - and these are my words - blue box routers. What he recommends is something called the Pepwave Surf SOHO from Peplink.

Leo: What? I never heard of that.

Steve: I know. And it looks very nice, too. Take a look at it from the link in the show notes, Leo. Michael writes: "I maintain a long write-up of its pros and cons at RouterSecurity.org. The Surf SOHO" - that's the name of the router - "is a business-class router." So it's not quite - it's a little more expensive than the cheesy blue boxes, but not enterprise grade. It's a business-class router, he writes, "that is a big step up from consumer models, yet is reasonably priced and no harder to configure than the average consumer-targeted router." He says: "My only relationship with Peplink is that of a customer."

But, for example, I love it when he's talking about this router, he says the router, for example, supports point-to-point VPN.

Leo: That's cool.

Steve: So you're able to link, to use the router through the public Internet to interconnect to remote networks. And he says, "Which is a feature I have sometimes used." He says it does not support WPS, which is another reason to like it because you don't want WPS...

Leo: Right, right.

Steve: ...in your router. So a lot of interesting features. For example, you could also - there's a WAN USB port on the back where you could plug one of the little cellular modems, and it will seamlessly, if your main connection to the Internet goes down, it will seamlessly switch to backup using the cellular connection. And it supports all of the various carriers and other stuff. So anyway, I just wanted to point people at RouterSecurity.org. I know that it would be of interest to our listeners. And Michael, who listens to the podcast, thanks for the great coverage of this weird TP-Link story and for making it clear that it's more of a tempest in a teapot. It's probably not going to affect people with older routers when they go back to try to navigate to their router's config page.

Okay. Now, finally, this week's Internet of Things horror story. Okay, Leo, first I have to say two weeks ago we covered one that was just really, like, another one of these. Some guy bought a multicamera Internet surveillance system. I can't remember the brand, but we talked about it two weeks ago. I just wanted to sort of fill you in.

Leo: It was Foscam. Foscam?

Steve: Huh?

Leo: It was Foscam; wasn't it? Oh, well, anyway...

Steve: Sorry?

Leo: Foscam? Foscam. No?

Steve: No, I don't think - I don't remember. Anyway, so he sets it up, decides it's not what he wants, and returns it. Then a couple months later he gets an email from the service that he registered it with, saying that one of the cameras had sensed motion. And he goes, "What?" and clicks the link. And now he's looking at somebody else's home. So the short version of this is that it turns out that somebody repurchased the returned multicamera surveillance system, set it up, created an account. And it turns out that you can, where the system is now, or it was, you could have multiple accounts all referring to the same camera system. So, and when the camera system sensed motion, it sent messages to everyone who had an account for that camera system.

So anyway, just another example of, oh, my goodness, this was not done right. And it was even worse, too, because then it turns out, when you look at it, the camera that you're viewing in real time, the MAC address was in - I don't remember the details now. But something was, like, in the URL. So you could just increment them and cruise around through all of this company's customers' homes that were under surveillance by themselves, and now by everybody else. So anyway, that we talked about two weeks ago.

This week we have a WiFi AC plug which has been withdrawn from Amazon. So a security researcher, Matthew Garrett, created a one-star Amazon review. The plug was called the - shoot, I don't see it here. AuYou, I think it was, A-U-Y-O-U. I went looking for it this morning, and then I subsequently saw a note from The Verge saying that it had been pulled, because I couldn't find it. And so that verified that it was gone.

So anyway, Matthew's title is "Nice hardware, infuriating setup issues, terrible insecure software." And so I'm just going to - I'm going to share the review because it's perfectly written, and there's no fluff in it. He said: "There's a lot to like about this hardware, but unfortunately it's entirely overwhelmed by everything there is to hate about it. But before we get to that, I received this product at a discount in return for writing an honest review of it. Onwards.

"The packaging is entirely reasonable. There's a small cardboard box, a sheet of instructions, and a piece of hardware securely wrapped in bubble wrap. It's very small, but feels well built. There's no creaking plastic under pressure, and no parts feel loose. Once it's plugged in, a blue LED in the button starts blinking, waiting for you to set it up. The app attempts to walk you through the setup, but things start going wrong here.

"The system is based on the ESP8266 module, which," Matthew writes, "is a great choice for this sort of application." Oh, and by the way, if you google ESP8266, it is everywhere.

SparkFun offers it for, like, $7 or something. It is a wonderful-looking little thing. WiFi, built-in 32-bit processor, a MB of flash memory. So it's a full system for connecting any of your IoT stuff to WiFi. So I agree with Matthew, just it's a beautiful little thing. Unfortunately, it wasn't very well designed from - not it, but the people who used it to build a smart plug around.

He say: "It's cheap [meaning inexpensive], but well featured. There's a lot of easy off-the-shelf code that vendors can incorporate into it to cut down development time and ship better products faster. One of the features available is something called Smart Connect." And I should mention I need to find out about this because this really sounds like the right way to go with IoT setup.

He says: "...where an app on a phone encodes your WiFi password into network broadcasts of different lengths." And I don't know what that means, exactly, but that's why I need to dig into it. He says: "It's possible to detect these even without the password, so this allows your phone" - although I have to make sure that other people can't see this broadcast. Maybe it's not such a good idea. Anyway: "This allows your phone to pass the information to the socket without having to fiddle about connecting to a different network. Simply hold down the power button to enter setup mode, and the phone does the rest.

"At least, it does in theory. In practice, it fails for two reasons. The first is that it'll happily try to do this on a 5GHz network, even though the socket" - that is, your phone will try to do it on a 5GHz network - "even though the socket only has 2.4GHz support. The other is that the app doesn't have the appropriate permissions to do this on Android 6, so it doesn't work on new Android phones at all, even if you are on a 2.4GHz network. However, it also supports a more traditional setup mode," which we've talked about before. "By holding down the power button again, it turns into an access point. Then the phone sees it, connect to it, and that allows the phone to then get the network setup data into the plug."

And he says: "Again, at least in theory. In practice, the app is looking for a network called SmartPlug. And this version of the hardware creates a network called XW-G03, so it never finds it." It's like, how can this possibly work? So he says: "I ended up reverse-engineering the app in order to find out the configuration packet format, sent it myself" - so this guy's got some serious skills - "and finally had the socket on the network. This is, needless to say, not a reasonable thing to expect average users to do. The alternative is to find an older Android device or use an iPhone to do the setup."

Finally: "Once it's working, you can just hit a button on the app, and your socket turns on or off. You can also program a timer. If your phone is connected to the same network as the socket, then this is just done by sending a command directly; but, if not, you send a command via an intermediate server in China." Sit down, everybody. "The socket connects to the server when it joins the wireless network and then waits for commands." So it has a static connection to a server in China, waiting for on/off commands.

"The command packets look like they're encrypted, but in reality there's no real cryptography at all." So it's just some light obfuscation. He says: "I wrote a simple program to decode the packets and looked at them in more detail. There's a lot of unnecessary complexity in the packet format, but in the end the relevant things are just a command and the MAC address of the socket. On the local network this is sent directly to the socket; otherwise, it goes via the server in China. The server looks at the address and uses that to decide which socket" - that is, which AC socket - "to pass it on to. Other than the destination, the packets are identical.

"This is a huge problem. If anybody knows the MAC address of one of your sockets" - which, remember, this is all essentially in-the-clear data. Anybody, for example, sniffing traffic to that server, anywhere along the line, would be able to get the MAC addresses of all of the sockets currently in use in the world. Anyway, so "If anybody knows the MAC address of one of your sockets, they can control it from anywhere in the world. You cannot set a password to stop them, and a normal home router configuration won't block this." Because of course it creates an outbound connection that the router NAT allows through, as it does all outbound connections.

"You need to explicitly firewall off the server" - and then he gives the IP address of 115.28.45.50. And of course we don't know if that's after a domain lookup or if that's hard coded. He says: "...in order to protect yourself. Again, this is completely unrealistic to expect for a home user; and, if you do this, then you'll also entirely lose the ability to control the device from outside your home.

"In summary, by default, this is stupendously insecure. There's no reasonable way to make it secure; and, if you do make it secure, then it's much less useful than it's supposed to be." He says: "Don't buy it." And, by the way, one star is the lowest you can do. You cannot do a zero star, unfortunately, because I tried once. Amazon doesn't recognize you're initiating a review until you select one to five stars. You can't do zero. So otherwise he probably would have.

Anyway, TechCrunch's Kate Conger reached out to Matthew, and she wrote in TechCrunch's coverage of this, she said: "Garrett sent me" - it's Matthew Garrett. "Garrett sent me a few emails he received from the company." And here's one, quote: "Just now my boss has blamed me, and he said if I do not remove this bad review, he will quit me. Please help me," the representative wrote. "Could you please change your bad review into good?"

Then Kate writes: "Garrett responded that he would update the review if the manufacturer fixed the flaw." Entirely reasonable. Oh, here it is, A-u-Y-o-u. So AuYou is the company. Maybe it's A-U-Y-o-u. Anyway, I don't know.

Leo: No one knows how you pronounce it.

Steve: "[The AuYou] representative insisted she would be fired if the review was not updated. A week later, she followed up again, asking Garrett to take down the review. The representative then said that she would report Garrett to Amazon if he didn't take down the review, and that other Amazon reviewers had written in to complain about it." Well, first of all, I doubt that's true. He has an insane number of up votes, something like 1,800 positive votes and only 10 weren't. So everybody, the 10 employees at AuYou all said bad review, bad review. The other 1,800 people said thank you to Matthew for this.

So again, Kate writing, said: "Garrett says he leaves a lot of security-based reviews on Amazon, and this has never happened to him. Of course, no one needs to lose their job over a single Amazon review." Well, except maybe in China. "Garrett says he's not sure if he's being manipulated or if someone's job really is on the line." My take is, sorry, that's completely irrelevant to the fact that he's doing this service by taking a look at this.

He wrote to Kate: "If I thought that there was a realistic chance that people were going to lose their jobs over something I was writing, that's something that would make me reconsider," he wrote. "On the other hand, the attitude that many companies have of not giving any indication of caring about the security of the people they're selling to is

horrifying in its own way. That is important, to make people aware when choosing these devices." And of course you couldn't - you're singing to the choir on that count. And I just - I am so grateful that we are still in a world where this is legal. That is, that the great threat of DMCA-ish legislation would make it dangerous for someone to do what Matthew has done, to take a look at how this stuff is working.

This is such a perfect case in point of how we desperately need to keep researchers who are not malicious, clearly, who have the intent of improving products. Look at Tavis with Project Zero and what he, I mean, yes, egg is on Symantec's face. And by the way, I skipped over, I just forgot, the blurb there about the fact that Symantec was using open source software that seven years ago had vulnerabilities that had been disclosed for which there is exploit code that works against the stuff that they've installed into everyone's kernels who are using this AV. So we have to protect this ability to have people challenge otherwise-closed systems that are not open so that people can pry into them and see what's going on. We have to.

And by the way, I did look at all of Matthew's other reviews, and he is - I think there were like 14 reviews, very much the same, I mean, not fluff at all, but really taking a close look at the way these things are working. So I just finished by noting that the plug appears to have been taken down from Amazon, and I just saw before the podcast that The Verge had reported the same thing. Matthew's review remains with a permalink, but I could not find the plug.

And then, looking at the other products that the same company makes, this AuYou, they have two full home security systems, which just makes me shudder. It's like, okay, these people should not be doing Internet-based connected products of this sort. They just don't demonstrate the ability to do it right.

Wow. Okay. So errata. Two things. First of all, last week I used a word that I didn't know the definition of, apparently. Despite the fact that back here is the AED, or I mean the OED.

Leo: OED, yeah, I see it.

Steve: The OED. I haven't read it recently.

Leo: Not cover to cover, anyway, I hope.

Steve: I said "penultimate."

Leo: Yes.

Steve: When I was describing SoftICE as the "penultimate debugger for DOS." And a number of listeners said, "Okay, so what's the ultimate?" And then I looked it up, it's like, well, penultimate means second best.

Leo: Yeah, it's the one before the ultimate.

Steve: And I'm thinking, who made up this word? I don't want a word like that floating around for people to use wrong. Okay. Give me a use for that word. What possible use is this annoying penultimate word? Who wants that?

Leo: All right. I'll give you an example. All right. In my opinion, the best episodes of any "Game of Thrones" series is the penultimate episode.

Steve: See, no one knows what that means.

Leo: Well, I know what it means.

Steve: You and three other people. And they all sent me email or texts.

Leo: You heard from everyone, all seven people who knew. Ultimate and penultimate are not the same thing.

Steve: I got that.

Leo: Yes.

Steve: And we could just remove penultimate completely. It's useless.

Leo: No, it's useful. The penultimate episode of "Game of Thrones" is always the best show.

Steve: Well, in this case I have to agree with you.

Leo: It's always the case.

Steve: That one is useful. But just say second to last because then we all know what we're talking about.

Leo: Okay.

Steve: Anyway, so there's that one. Second is I have had version control in place for decades. I didn't want to get into it because mine is my own. And it's not GitHub. It's not cloud-based. But so many people who understand the value of version control generically tweeted or sent email and said, "Oh, come on. Gibson, how can you even be, like, calling yourself a programmer?" It's like, okay, okay, okay. So I have absolute backtracking auditing on every piece of code that I write. It's built into my infrastructure. I can go back in time on a file-by-file basis, not even checkpoint-by-checkpoint, but individual

tracking back the changes in all of my source code as far back as I want, with high granularity in the near term and slowly reducing granularity as the files get older because the presumption is I don't need as much back then. So, yes.

Leo: And you wrote that yourself. That's cool.

Steve: Yes. Sorry that I was glib about it. But believe me. And it's the kind of thing where you normally don't need it. But oh, baby, when you do? In fact, I've used it just a couple times. I'm tracking down a weird bug in the SQRL client that I introduced - inadvertently, obviously - when I updated the identity system to maintain four previous identities. It had deep repercussions throughout the code I'd already written. And so several times I've gone back a few months before I did that to take a look at what I had then.

So, yes. Rest assured, everybody who knows how important it is, I've got it, too. And I wouldn't live without it. There have even been times like when I've just fumbled a finger and hit delete on a source file. And it's like, oooh. And you sort of stare blankly for a minute, and then it's like, okay. And good news is I have something from 10 minutes ago, so I'm fine. Anyway, so yes. I absolutely understand the need for it and have it.

Real quick note for our sci-fi-interested media viewers. The two shows that are, eh, they're not top drawer, but they're post Syfy reboot that we've talked about. Their second seasons just started. And I know that many of our listeners are watching. And it's two shows, "Killjoys" and "Dark Matter." I think "Dark Matter" is the better of the two. And they're just romp 'em, shoot 'em up, space team stuff, but they're fun. And so I wanted to make sure that people knew that they had resumed last Friday. I've not even watched them yet. I've been really busy. But I will because they were fun and sort of character-driven. I mean, they're, like, not amazing, but they're okay.

And for those of us who are interested in politics, completely off-topic, but this'll be quick. This week's bit.ly shortcut, so bit.ly/sn-567, is the cover story of this month's Atlantic magazine, titled "How American Politics Went Insane." On Sunday's "Meet the Press," Chuck Todd interviewed the author. And the things he said immediately brought me to attention because it echoed a conversation I've been having with my mom, sort of about sort of processing what we're seeing in this U.S. cycle's presidential election. And I won't go any further. And I won't take up any more time.

So I should say that link is to a PDF I prepared. I'm obviously a political follower, and I've been wanting to understand what's going on. This is a great piece. So it's a PDF that my site is hosting. I put the cover on it so that the Atlantic would get all of the credit for it. And it is freely available on the web also: "How American Politics Went Insane." And for what it's worth, if you follow politics, I recommend it without reservation. It's long. It's about 20 pages, although the first page is all just cover. But worthwhile if you're interested in such things.

And I did want to make a note that, as I promised last week, the second-generation of the Healthy Sleep Formula page is updated and online. I'm not finished with it, but all the important stuff is there, and I'll be adding to it shortly.

And finally, I got a nice note from Jeremy Leik, spelled L-E-I-K - and he wrote, "pronounced 'like,' blame my German ancestors" - in Dimondale, or Dimondale, Michigan. Don't know. Dimondale, maybe. And he said: "SpinRite mentioned on Technibble.com." And he said: "Hello, Steve and Leo. I'm writing today because I'm a user of SpinRite and

a believer in its capabilities. It has saved a few drives for me, and allowed me to recover data from a RAID array that came out of a NAS with failed electronics.

"I came across an article called 'Technical Overview of Popular Software Data Recovery Procedures' over at Technibble. They mention SpinRite on page 2 of their article. The part where they describe SpinRite seems pretty simplistic, and lumps it in with other software tools. I would be interested in hearing from the horse's mouth, so to speak, what your thoughts are on this subject. Thanks so much. Jeremy Leik."

And so I'll just say that I completely understand that. One of the good things about SpinRite is that it is so simple to use. I mean, its goal is to fix your drive. And that's what you do. That's what it does. You run it, you point it at the drive, and it goes and fixes it. And so it doesn't look like there's much there. But when I was thinking about what Jeremy wrote, I was reminded of a person I hired 20 years ago by the name of Mike Toutonghi, who was an incredibly, is an incredibly brilliant person. Which is why I hired him.

And he'd been with GRC for, I don't know, a few months, and we were having lunch together, and I started explaining to him - he said something that indicated how unimpressed he was with SpinRite. You know, it was paying his salary at the time. But he just didn't think there was much there. And so I said, "Okay, Mike. Let me tell you what's in here." And I started to go into it, and over the course of about an hour, like what it really takes to do what SpinRite does, what SpinRite completely hides. And he was stunned. I mean, he said he could not believe - that had never occurred to him, what would be necessary to do what it does, and how it works. And a lot of that was proprietary. So there is stuff I don't talk about because it is what pays all the bills.

But Mike sort of outgrew my little company. There wasn't room for both of us. So I was good friends with Brad Silverberg, who was one of the executive VPs at Microsoft at the time. And I gave Brad a call, and I said, "Hey, Brad, I've got somebody for you." Mike did go to Microsoft. He was the MVP, the number one most valuable employee of the year at Microsoft and became one of their 12 - they have like a special designation, like something scientist or something. I mean, he's like - he did .NET. He invented the whole .NET system himself. So anyway, he knows where he's coming from, software-wise, which is why I hired him. And he was blown away.

So for what it's worth, what I love about SpinRite, like most of my software, is it just does what you want it to. And it deals with all of the hard stuff itself, which is what makes it fun for me to create it in the first place. And the bottom line is you run it, and it fixes your drive. So Jeremy, I'm not surprised that somebody trying to do a review piece sort of says, oh, yeah, and here's a bunch of other stuff, and there's something called SpinRite, and, you know, blah. It's like, okay, except, you know, it works.

Leo: It works.

Steve: For the last almost 30 years now.

Leo: Yeah, yeah. Well, we have enough, I think, anecdotal evidence to say...

Steve: Oh, my goodness.

Leo: ...it absolutely works.

Steve: Yeah. At some point it stops being anecdotal. It becomes rigorous.

Leo: Yeah. All right. Let's write a script. Let's Encrypt.

Steve: Oh, let's not do one for certificates. So where is NL, the domain? That's Norway?

Leo: Netherlands.

Steve: Netherlands, right. Computest is a security group in The Netherlands who took a look at a newly launched service called StartEncrypt. So it's like, okay, uh-huh. So Let's Encrypt, StartEncrypt, at least they didn't try to steal the trademark the way Comodo did. So the company is StartCom. And they have the so-called StartSSL are their certificates. And so the StartEncrypt project was launched on June 4th, so, what, a month ago. And of course it was inspired by the success of the Let's Encrypt project. Okay, well, I loved Computest's disclosure because it was almost comically understated. And I don't know if it's an English language translation thing. It was just so polite. And it's discussing something that is so horrifying.

So they wrote: "Recently, one of our hackers found a critical vulnerability in StartCom's new StartEncrypt tool, that allows an attacker to gain valid SSL certificates for domains he does not control." Which is like, what? So we have a trusted certificate authority whose entire raison d'etre is to make sure that the identity of the people they are issuing certs to has been verified. Anyway. So they continue with this understatement: "While there are some restrictions on what domains the attack could be applied to, domains where the attack will work include Google.com, Facebook.com, Live.com, Dropbox.com and others."

Leo: Oh, my god.

Steve: Yeah, we've heard of them. "StartCom, known for its CA service under the name of StartSSL, has recently released the StartEncrypt tool. Modeled after Let's Encrypt, this service allows for the easy and free installation of SSL certificates on servers. In the current age of surveillance and cybercrime, this is a great step forward, since it enables website owners to provide their visitors with better security at small effort and no cost." And of course we talked about what a fabulous success Let's Encrypt has been. And you can imagine the position that StartSSL is in. Now, they had been offering free certs, limited time. You'd have to keep going back every year, so there would be a chance for them to upsell. But still, the Let's Encrypt model is better than free because it's free, and it's automated. So they think, okay. Let's automate.

And then, continuing Computest's statement, they said: "However, there is a lot that can go wrong with the automated issuance of SSL certificates. Before someone is issued a certificate for their domain, say Computest.nl, the CA needs to check that the request is done by someone who is actually in control of the domain. For Extended Validation certificates, this involves a lot of paperwork and manual checking. But for simple, so-

called Domain Validated (DV) certificates, often an automated check is used by sending an email to the domain or asking the user to upload a file.

"The CA (Certificate Authority) has a lot of freedom in how that check is performed; but ultimately the requester is provided with a certificate that provides the same secure" - and this is the key, too. "The requester is provided with a certificate that provides the same security, no matter which CA issued it." Right? Because all of the CA's private keys, which are used to verify the signing under - I'm sorry. All of the CA's public keys are in the devices we use, which are used to verify the signing under the CA's secret private key. So we treat all certificates identically, regardless of what CA signed it.

So they say: "In order to make the issuance of certificates easy, this tool runs on your server (Windows or Linux), detects your web server configuration, and requests DV certificates for the domains that were found in your config." So that's nice. It looks into your config, sees which domains your server's configured to handle, and just asks for those certificates. "Then the StartCom API" - meaning at the server end - "does an HTTP request" - and I just hit spacebar and changed the page, sorry - "does an HTTP request to the website at the domain you requested a certificate for."

So the far end, the StartCom server makes a request to check for the presence of a piece of proof that you have access to the website. So the idea being that remotely it uses that domain to ask for a specific file which then verifies that you were able to put that file on that domain. Therefore you're controlling the server at that location. So it checks for the presence of a piece of proof that you have access to that website. "If the proof is found, the API returns a certificate to the client, which then installs it into your config." So very much like Let's Encrypt, it just automates the whole process.

"However," they write, "it appears that the StartEncrypt tool" - and I love this - "did not receive proper attention from security-minded people in the design and implementation phases." Now, understand, it's not like security is an afterthought for a certificate authority. It's a stretch to call it an afterthought for an AC wall plug. But security is not an afterthought for a certificate authority.

Leo: Oh, security schmecurity. [Crosstalk].

Steve: I know. Get a load of this. So they write: "While the client contains numerous vulnerabilities" - we do get to those at the end - "one in particular allows the attacker to trick the validation step." So the first bug. "The API," as we noted, "checks if a user is authorized to obtain a certificate for a domain by downloading a signature" - which is like a file - "from that domain, by default from the path '/signfile.' However, the client," okay, at the user's end, "the client chooses that path during an HTTP POST request to that API," meaning that "a malicious client can specify a path to any file on the server for which a certificate is being requested. This means that, for example, anyone can obtain a certificate for sites like Dropbox.com and Github.com, where users can upload their own files." So the point here is, just to clarify...

Leo: Ohhh. I get it.

Steve: Yeah. You don't have access to the root of Dropbox. You have access to some screwy-looking URL with a bunch of gobbledygook in it.

Leo: But it begins with Dropbox.com.

Steve: Exactly. And so you're able to make a request of the StartCom - in your HTTP POST query to their API, the actual POST contains the path /signfile. You simply change that to your directory on Dropbox, where you placed that /signfile on Dropbox, and StartCom sends you a completely valid HTTPS certificate for Dropbox.com. How handy. Now you can have your own Dropbox.

Leo: Oh, my god. That's ridiculous.

Steve: I know. Unbelievable.

Leo: That's ridiculous.

Steve: Then they say: "That's not all. While this is serious, most websites don't allow you to upload a file and then have it presented back to you in raw format like GitHub and Dropbox do." So those are sort of unique in that they are file storage sites, which just make this exploit drop dead, or drop box, simple. "Another issue in the [StartCom] API, however, allows for much wider exploitation of this issue: The API follows redirects. So if the URL specified in [one of these POST parameters called] 'verifyRes' parameter returns a redirect to another URL, the API will follow that until it gets to the proof [that it's looking for], even if the redirect goes off-domain." Which is just crazy. "Since the first redirect was to the domain that is being verified, the API considers the proof correct, even if it is redirected to a different website."

Okay. So again, to review, that would mean that you're asking for something from, literally, and this works, Google.com. Why does it work? Because Google is an OAUTH provider. And OAUTH requires URL redirection following in order for it to bounce the user's web browser around, as we've talked about in the past. So this StartCom API is so broken that, if it goes to Google.com and receives a redirect somewhere else, it says, oh, I got a redirect from Google.com. But that verifies the guy has control of Google.com. Even though Google then sent it somewhere else that the user actually has control over.

So this indirection opens up a much broader range of exploitation because basically people can arrange to have a Google URL redirect wherever they want, and that is then where they put the content that this StartCom API picks up. And these are called "open redirects." They're pages that take a parameter that contains a URL and that redirect the user to that URL in the page's parameter. Now, this mechanism makes people nervous. But, for example, we often see it in login and logout pages, where you'll logout, and the logout URL will have an argument, like "returnURL=/login," so that after you log out, the web server bounces the browser over to the login page.

And it turns out that the OAUTH, as I mentioned before, requires open redirects. It's not technically a security vulnerability, unless in this instance it's able to be leveraged into a huge vulnerability as a consequence of the fact that the thing that is following the redirections takes the first domain as the one that's being verified, rather than the last domain, where it actually gets the asset whose ownership it's trying to prove. So, I mean, it's just like, okay. This is a certificate authority that everybody trusts. And they've automated this with gaping holes in their API.

So anyway, they say: "That's still not all. Apart from the vulnerabilities described above, we found some other vulnerabilities in the client while just doing a cursory check. We are only publishing those that according to StartCom have been fixed or are no issue. These include: The client doesn't check" - okay. The client running in the user's machine, right, in the user's server that reaches out and connects to the remote StartCom server. The client, their own software, "doesn't check the server's certificate for validity when connecting to the API, which of course is pretty ironic for an SSL tool."

And so I added a note here: Which of course allows for man-in-the-middle attacks against the StartEncrypt client. Bad guys could arrange to have invalid certificates issued, thus creating a denial of service against targeted sites using StartEncrypt because it's not checking the certificate. So it just could be any certificate that is set up on a man-in-the-middle intercepting server. And that interceptor is not going to be a CA, so it just sends garbage back. And that garbage gets installed into the end-user's server; and, look, HTTPS is broken. Wow. And who knows what else, what other kind of havoc could be created.

Second: "The API doesn't verify the content type of the file it downloads for verification, so attackers can obtain certificates for" - and get this. "Attackers can obtain certificates for any website where users can upload their own avatars," like a blogging a site.

Leo: Or a forum.

Steve: Where you create an account with a profile, and you upload a picture. So you would upload that signed file as your picture and then hand that URL to the StartCom API, or the StartEncrypt API, and it would give you a certificate for that site, that social site where you created an account. Just incredible.

And finally - and you'll get this, Leo, because you've been deep into Linux recently - the private key, right, the final sacrosanct private key, which by the way is over a connection whose security is not verified, the private key on the server - now, that's on the end-user's server. The private key on the server is saved with file mode 0666. So it's world-readable. Which means any local process or user on that machine can read or modify it. And that's the secret for that server's certificate, the private key, the thing you absolutely never want to let anyone else get. So: "All in all, doesn't seem like a lot of attention was paid to security in the design and implementation of this piece of software," they say. And it's like, you think? Wow. Yeah.

Leo: Nice job.

Steve: What Let's Encrypt did is good. The bad news is wannabe, me-too companies are going to say, oh, we're losing free business to Let's Encrypt. So we should do it, too. And here's an example of how badly it can go wrong. And, boy, again, we just don't have facilities in place for policing this. Again, I will say thank goodness that researchers are able to look at these things and are able to responsibly bring them to light, and all of us can learn lessons from them because - and hopefully all the other CAs are going, oh, no. We were going to launch ours next month. Let's make sure...

Leo: Yeah, maybe we should just check it. Maybe, you know.

Steve: Maybe we should.

Leo: Maybe 0777 would be better than 0666. Let's just go all the way, get people execute it.

Steve: That's right.

Leo: Wow. You know, it's kind of amazing. This stuff, you know, the proof files...

Steve: Security is hard.

Leo: I know, but this is a well-known thing about, you know, proving control of the server is easy.

Steve: I know. This is embarrassing. This is embarrassing.

Leo: Yeah. Yeah.

Steve: This is just, I mean, I loved that last sentence: "All in all," and they wrote this, "it doesn't seem like a lot of attention was paid to security..."

Leo: It's called understatement.

Steve: "...in the design and implementation of this piece of software." And again, it's not even a plug. It's a certificate authority, issuing certificates that the world is going to trust. Oh.

Leo: Ugh.

Steve: And you can get one for Google.

Leo: Yup.

Steve: Wow.

Leo: Wow. Well, my friend, we have concluded this fabulous episode of things done wrong, with a few things done right. You can catch this show, we do it every Tuesday at about 1:30 Pacific, 4:30 Eastern, 20:30 UTC, if you want to watch live at TWiT.tv. But you don't have to, of course. We have on-demand audio and video. Steve does a great job of not only putting the audio up there, but of written transcriptions at GRC.com. That's his website.

Steve: We may be a little bit late with it this week. Elaine was saying, "Oh, Steve, please do a shorter podcast because my schedule is really jammed up." And she's so responsible that she's just going to take it out on herself. So Elaine, are you listening? Oh, actually, this is the end. So, oh, but she does a whole proofreading phase.

Leo: Right.

Steve: So Elaine, you're halfway through. Please, you're hearing us both say take your time.

Leo: Take your time, yeah.

Steve: Yes.

Leo: No hurry.

Steve: I'll post it up when we get a chance because here we are at two hours and two minutes and counting.

Leo: Not bad.

Steve: So it's our standard-length podcast.

Leo: Yeah. GRC.com. You also find, if you want questions, I guess we'll do a question episode next week. You can of course go to GRC.com/feedback. That's the feedback form. Or tweet Steve. He's @SGgrc. You can even DM him because he accepts DMs from all comers.

Steve: Yeah, don't sent tweets to @stevegibson. That really annoys him.

Leo: Is there a guy named @stevegibson?

Steve: Oh, yeah. And he sometimes bounces tweets over to me. He says, no, it's - and it was like, I would say hey, I'm sorry, but, you know...

Leo: @SGgrc. See? Easy. It's his initials plus his website's name. But not .com because that - never mind. @SGgrc. And I have the other ones up there. You probably don't do much with @SGpad and @SGvlc.

Steve: No, that was sort of an interesting idea, but it didn't really...

Leo: I'll erase those.

Steve: Didn't gain enough traction.

Leo: Yeah. One Twitter handle is enough for any human.

Steve: Yeah, yeah. And by the way, we're closing in on 1.5 million downloads of Never10.

Leo: Wow. Well, you've got till July 29th.

Steve: Oh, yeah.

Leo: To go over the billion number.

Steve: I think it's, like, 15,000 a day now. It's slowing down, but still staying strong.

Leo: That's great. That's amazing.

Steve: Yeah. Crazy.

Leo: You'll find many other things at the website, including SpinRite, of course, Steve's amazing product that does work, the world's best hard drive maintenance and recovery utility.

Steve: There's a lot of magic under the hood that people don't appreciate. But that's why it works.

Leo: GRC.com again for that. Now, if you go to TWiT.tv/sn, you'll find audio and video, so you could see Steve's giant hands.

Steve: Or head.

Leo: Or head. It's a trick of perspective, my friends. His hands are just normal. They just look big. And I guess that's about it. I guess that wraps it up for another fabulous week. Thanks, Steve. We'll see you next time.

Steve: Thanks, Leo. Great to be with you, as always.