SECURITY NOW!

Transcript of Episode #626

Shattering Trust

Description: This week we cover a bit of the ongoing drama surrounding Marcus Hutchins; examine a reported instance of interagency hacking; follow the evolving market for zero-day exploits; examine trouble arising from the continued use of a deprecated Apple security API; discover that Intel's controversial platform management engine can, after all, be disabled; look into another SMS attack; bring note to a nice-looking TOTP authenticator; recommend an alternative to the shutting-down CrashPlan; deal with a bit of errata and miscellany; then look into an interesting bit of research which invokes "The Wrath of Khan."

High quality (64 kbps) mp3 audio file URL: http://media.GRC.com/sn/SN-626.mp3
Quarter size (16 kbps) mp3 audio file URL: http://media.GRC.com/sn/sn-626-lq.mp3

SHOW TEASE: It's time for Security Now!. Steve Gibson is here. We've got a lot of security news; a surprising update on the Marcus Hutchins legal defense fund. Steve's going to explain some of the latest news about the Intel management engine. It turns out, thanks to the NSA, we now know how to disable it. And he'll explain about how "Star Trek 2: The Wrath of Khan" applies to your security policy. It's all coming up next on Security Now!.

Leo Laporte: This is Security Now! with Steve Gibson, Episode 626, recorded Tuesday, August 29th, 2017: Shattering Trust.

It's time for Security Now!, the show where we cover your security and privacy online with this, well, normally it would be with Steve Gibson, but I don't - somebody else is sitting in for Steve. No, it is Steve. And for those of you listening, he'll sound just like Steve. But there is something...

Steve Gibson: Yes, it has not changed my voice at all.

Leo: There is something a little different. We're going to call you Babyface from now on.

Steve: That actually has been the consistent response from various women who have seen me since is, "Wow, you look younger." It's like, well...

Leo: Steve shaved his - how long have you had that moustache?

Steve: Forever. It was briefly removed while I was married. Juliette wanted to see what was under there.

Leo: And then she said, "Put it back, put it back. I don't know who this is."

Steve: She just wanted to change everything to sort of make it hers. Anyway, so it came back. But I don't know. So I've been thinking about it for a few months. And finally I said, well, if I don't like it, I can always let it grow back. But, yeah, I think the reaction has generally been favorable, so what the hell. Time for a change. Now the problem, it's weird, too, because I didn't realize that in this new world we live in my picture is everywhere.

Leo: Well, we have to redo the mugs.

Steve: Twitter feed. Right. My Twitter feed, and you and me as Jean-Luc, as Picard and Number One, I can't think of his name.

Leo: The guy with the beard, Frakes, Jonathan Frakes.

Steve: Yeah, exactly.

Leo: What we're going to have to do here - actually, I think it makes the mugs more valuable. This is the...

Steve: Ah, the Collector Edition, yes.

Leo: ...Collectors' Edition with Steve with his moustache.

Steve: Yes, for 12 years of the podcast, never to be revisited. Get them before they sell out because, you know, who's going to make more of those now?

Leo: We will. TWiT.tv/store. They have Security Now! mugs with - and the best thing about this, and I think this was Nitrozac who did your original illustration, a little editorializing on her part, you look quite confident and cocky in these. And I love them.

Steve: I look snobby. I've never liked that picture. I look aloof and, like, you know, you know what doesn't stink. It's like, no.

Leo: No, you look like I'm secure, and I'm proud of it, or something. I don't know. I like it. To me, I know it's not your personality, but I do like those. And we have T-shirts and mugs, and they're available at TWiT.tv/store. When you see, like, last day to order and all that, that's just the way that our supplier Teespring works. We will always make more. It's just a little weird thing about - we've sold 74 of those mugs, by the way, Steve. I think that's a store record.

Steve: People are tweeting me themselves sipping coffee from them. So thank you. I mean...

Leo: And we're not - I don't think we're making money. If we are, it's just a couple of bucks on these. They're $15 for the mugs, and I think $18 for the T-shirt.

Steve: Yeah, that's below Starbucks' going price for their...

Leo: Yeah, yeah. Less than Starbucks. And we have a variety. I think we even have, yeah, we have a hoodie. Now, that, if you're going to DEF CON or Black Hat, you should wear a Security Now! hoodie, for sure. That's $38.99.

Steve: That'll keep you warm.

Leo: So what's coming up today on the show?

Steve: Okay. So today's podcast, No. 626, I titled "Shattering Trust," after the title of the coverage we're going to give at the end of the podcast of a research paper titled "Shattering Trust," which is really interesting. But of course, as always, we've got a bunch of stuff to talk about. We're going to cover a little bit of the ongoing drama surrounding Marcus Hutchins. Another little wrinkle has surfaced.

Leo: Yeah.

Steve: Examine a reported instance of interagency hacking, which I wouldn't have expected. But after you hear about it, it's like, well, sad, but probable. We're also following the evolving market for zero-day exploits. Examine trouble arising from the continued use of a deprecated, actually a long-deprecated, four years now, Apple security API. Discover that Intel's controversial platform management that we've talked about here often can, after all, be disabled. Look into another instance of SMS attack, you know, multifactor attack. Bring note to a nice-looking time-based one-time password authenticator. Recommend an alternative to the shutting down CrashPlan, which a lot of our listeners apparently subscribed to and said, oh, my god, where do I go?

Leo: Carbonite, of course, our sponsor, yeah.

Steve: Carbonite even recommended, I mean, they recommended Carbonite as...

Leo: CrashPlan did, yeah, yeah.

Steve: ...one alternative. We're going to deal with a bit of errata. And this one really caught me off guard. And when I saw the first mention of it, I thought, oh, I probably misspoke once. And then it turns out, no, apparently I just used the wrong word through my entire litany.

Leo: And I didn't catch it? I apologize.

Steve: Well, I didn't catch it. And so I just - I used it, I must have used it initially wrong, and then I just kept on using it wrong. So we're going to fix that. And then we're going to take a look at this interesting bit of research which, of all things, invokes "The Wrath of Khan."

Leo: Whoa.

Steve: So I think a good podcast for us.

Leo: "The Wrath of Khan." And by the way, I've just been handed this note. Alex Gumpel says we're working, our R&D division is working on new Steve Gibson Security Now! T-shirts with removable Velcro moustaches.

Steve: Ah, perfect.

Leo: So you could have it on or off. So we're going to get that out of R&D, and we'll let you know.

Steve: Or if you have some of that Wite Out, since the mugs are white, you can just cover right over there.

Leo: Moustacheless Steve Gibson. It's going to take me a while to get used to this, I'll tell you.

Steve: Oh, I don't spend much time looking at myself.

Leo: Right, you don't know.

Steve: This happened around middle of the day on Saturday.

Leo: What was it like shaving it?

Steve: It was weird. And I did it again this morning for the podcast. And I have to be careful because - although it's better.

Leo: Your lip is tender there, yeah.

Steve: You have to kind of get in underneath and get the upper lip and push that...

Leo: Oh, it's hard, I know.

Steve: Yeah, so anyway, yeah.

Leo: But your coffee tastes different, I bet.

Steve: Okay. So our Picture of the Week reminds us of that classic dumb joke from the old days when we were in elementary school. Is your fridge running? Which I guess the idea used to be...

Leo: You'd call somebody. You'd call somebody.

Steve: You'd call someone, exactly.

Leo: You'd say, "Do you have Prince Albert in a can?" And they'd say, "Yes, we do." You'd call a tobacco shop. See, that's how old-fashioned this is. "Yes, we do." "Well, let him out."

Steve: Yes, let him out.

Leo: Or you'd call a housewife, "Is your refrigerator running?" "Yes." "Well, you'd better go get it. It's getting away." Ha ha ha.

Steve: Right. So updated to this world of 2017 is our Picture of the Week that poses the question: "Is your fridge running?" And then when they say, you know, whatever, it's "Because I can't log into it anymore." So, yeah. Our brave new world.

Leo: Do you watch "Silicon Valley" on HBO?

Steve: Oh, are you kidding me? Yes.

Leo: That was one of the big plot points this season was hacking into - I can't remember their names. The kid who was in the incubator bought a very, very, very fancy computerized refrigerator.

Steve: Oh, right, right, right.

Leo: Remember? And then...

Steve: Yeah, the younger Asian guy who...

Leo: [Jian Yang] or whatever his - and then of course it drove Gilfoyle crazy, and so he spent many hours and ended up using all the CPU cycles in their server to crack this refrigerator, and then it just said nasty things. It was very - I love that show. Anyway, I'm sorry. Go ahead.

Steve: Well, and I think what's fun is for techie geeks like us, I mean, it does tie into all of the experiences that we're having and sharing. So, I mean, yes, it's very relevant.

So, okay. In our Marcus Hutchins news is a sad story with I think an interesting takeaway. Kevin Collier, who is BuzzFeed's News Cybersecurity Correspondent, reported some unfortunate news surrounding the high-profile arrest that we've been covering for the last couple weeks, post-DEF CON, of the so-called WannaCry hero, Marcus Hutchins. It turns out that the overwhelming majority of money which had been raised to pay for his legal defense turned out to have been donated with stolen or fake credit card numbers.

Leo: Oh, man.

Steve: I know. And as a consequence, all donations, including the legitimate ones, because there was no way to tell them apart, are being returned, according to Tor Ekeland, who's the attorney who was managing the fund. He reported, I think it was last week, that at least $150,000 of the money collected came from fraudulent sources, and that the prevalence of the fraudulent donations effectively voided the entire fundraiser. He said he'd been able to identify only about $4,900 in legitimate donations.

Leo: Now, who would do that?

Steve: I know. But he couldn't be certain even of those. He told reporters: "I don't want to take the risk, so I just refunded everything. One person had five different charges to his one credit card," Ekeland told BuzzFeed News. "Odd number values, the kind you'll glance over when looking at your bill," designed obviously to hide them and obscure them. And so clearly what happened is people began getting their credit card statements and saying, wait a minute, what's this charge? I mean, it happens to us all the time. It's mostly all Sue does for me is someone will call up and say, "I didn't ask for any research." And Sue says, "Did you buy SpinRite?" "Well, of course." "Well, that's where you got it." "Oh." And then they are happy, and they go away.

Leo: Oh, I get it, Gibson Research. I get it, I get it. "I didn't buy research."

Steve: Yeah, nobody thinks of us as the SpinRite people. So basically Sue sits around explaining to people what that charge is on their credit card statement, and then they're happy.

Leo: Yeah, yeah, that's funny.

Steve: Anyway, in this case, Ekeland, who was in charge of this, started getting all of these calls from people saying, who are you, and why did you charge me X amount of money?

Leo: Right.

Steve: And he said, "Did you donate to the Marcus Hutchins legal defense fund?" And they said, "Who's Marcus Hutchins? No." And so, anyway, big problem. He said he's still determining what to do with a bitcoin wallet which was set up to take donations for Hutchins, which has received 96 small donations worth a total of about $3,400.

So the problem, of course, is that there's no doubt some legitimate, well-meaning, good-hearted people among the noise; and the noise, however, swamped it all and ended up, I mean, I don't know if the bad guys intended this to happen in order to kill the fund or thought they were doing Marcus a favor. But of course, if that was the case, it ended up completely backfiring. And as we know, his arrest shocked the community of cybersecurity researchers, who have since largely rallied behind him and tried to raise money for experienced attorneys. At the time, Ekeland stepped in to host the fundraiser, after GoFundMe refused to host it, citing concerns with its terms of service. I don't know what that was about, but okay.

So of course, as we know, Hutchins, who pleaded not guilty to all six charges against him on August 14th, has retained Brian Klein, an L.A., that is, Los Angeles-based trial lawyer, and Marcia Hoffman, an acclaimed expert on U.S. hacking laws, both as his attorneys. And as we discussed, I think it was last week, a judge told him he could not return to the U.K. before his trial, yet he had free roaming through the U.S. I'm sure they pulled his passport, and they also GPSed him. And his trial is set for a little more than a month from now, in October. And in the meantime he's hanging out in Venice Beach and enjoying the weather and waiting for trial.

A woman named Tarah Wheeler, who is a friend of his, previously had told BuzzFeed News that she planned to set up a new legal defense fund for him, but did not immediately respond to their follow-up request for comments on this story. Ekeland said he believes he has refunded every donation to the fund, but that anyone who hasn't heard from him can email info@torekeland, T-O-R-E-K-E-L-A-N-D, dotcom for their refund.

And of course, right on cue and apropos of this, we have the just-posted note from Brian Krebs, who posted a piece with the headline, "Beware of Hurricane Harvey Relief Scams."

Leo: Oh, no.

Steve: I know.

Leo: What's wrong with people?

Steve: Brian writes: "U.S. federal agencies are warning citizens anxious to donate money for those victimized by Hurricane Harvey to be especially wary of scam artists. In years past we've seen," writes Brian, "shameless fraudsters stand up fake charities and other bogus relief efforts in a bid to capitalize on public concern over an ongoing disaster."

And our listeners may recall that, when I mentioned this Hutchins legal defense fund, I was inherently cautious about it.

Leo: Yes.

Steve: I looked at the site. I scrutinized it as best I could for, like, any signs of legitimacy and illegitimacy. And I even said, you know, I can't vouch for this. But I did look at it, and it appears legitimate. Now, I don't know what that even means because I guess it would be obvious if it didn't appear legitimate, if it was broken English or like any of the cues we're used to seeing in poorly produced spoofs. But as they say, there's no way to, what is it, disprove a negative? I don't know what that is, but...

Leo: Yes, that's exactly right. You can't prove a negative.

Steve: Yeah. So it's like, okay. It's there, and it seems like the right one. And it was, as it turns out, but that didn't save it. So it occurs to me that we're seeing something of an emerging trend with this, which is high-profile online things of any kind that attract outsized attention of any sort are not left alone and allowed to function as intended. By being high-profile, they become targets of mischief. And if those sites are not adequately prepared to actively fend off mischievous or malicious attacks, disaster results.

And of course we had another example when the FCC put up the Net Neutrality feedback site. It was so swamped with fraudulent postings and other nonsense as to become virtually useless. They had to pull it down, take it offline. Then they tried to put it back up, but of course it had been ruined by a ton of insincere crap that had also been put up. So, and now here we have an entirely well-meaning online fundraiser similarly rendered useless by fraudsters who stole money from others' credit cards to illegally contribute to Marcus's defense fund, of course ultimately not helping him at all, and as a result collapsed the entire thing.

So I don't know how you could protect against this. Maybe question, for one thing, question multiple donations from the same credit card. That you could certainly check. But they weren't doing that. Because if nothing else it would have raised an alarm immediately to put this thing on hold when a duplicate charge from the same person - and then, you know, follow up, verify with the person and so forth. But that wasn't there.

So, for example, perhaps in both cases this was sloppy and immature website design that put up a minimally functional web surface that wasn't sufficiently proactive against malicious abuse. We've seen uncovered many examples of such short-sighted design in the past on this podcast.

And this is sort of the logical extension of the how do websites store our passwords question. As we know, there are ridiculously wrong ways and very strong ways to solve exactly the same problem. And the approach taken doesn't really matter if everything always goes well, and the strategy never is put under stress to test its strength. Because when weakly designed and/or implemented systems are put under stress, then they will collapse and fail. So I think that helps us to sort of put this into context, that is, there are lots of sites with bad security; but because no one really cares about them, it's like, well, okay, it works. And it's not until, though, something happens that draws a lot of attention, and then, boy, disaster strikes quickly.

And we're operating, when you think about it, in a completely unregulated environment where there are no controls. Anyone can do anything they wish, set up any kind of website they choose using whatever technology occurs to them first. I had a more colorful metaphor there initially. I thought, no, that's inappropriate. I doubt that imposing regulations upon the industry will help. But I think we do need to make use of more well-thought-out solutions, easier to deploy, things like we were talking about last week, the libsodium cryptographic library, which was recently audited and which had bindings to virtually every language. So there's no excuse for not deploying strong crypto. And it brings useful drop-in encryption to a much larger audience.

The next step, then, would be to incorporate similar libraries into drop-in problem-solving packages, that is, provide all the rest of the plumbing so that much of the underlying work can become standardized around solid solutions. And in such a future, accepting and storing passwords don't need to be written ad hoc because the right way to do it would be built into every platform that a developer might choose to use. So essentially those things become commodities, as they absolutely should be, so that it's just done for a developer, rather than it being up to them to just do something which is as likely to be incorrect as not because not everybody is experienced in secure design concepts for security. Our listeners are now, but we're still a small subset. And yes, Leo, I just, you know, so sad. As you said, what's wrong with people? It's like, well...

Leo: I feel like there's a break, there's kind of a breakdown in - maybe this is what old people feel like - in the moral order somehow.

Steve: I'm right there with you, yup.

Leo: And people don't care anymore, or they just - it's like - I feel like, you know what it is, it's almost anarchy, kind of, a strain of anarchist, people who will do it for the lulz, who do it because they don't - they just - they like to see the world burn. And that's what I suspect is going on. It's just like, yeah, wouldn't it be funny if we did this? They don't care.

Steve: Well, and, you know, there's also this aspect of getting away with it and anonymity. That is, a lot of the mischief that we see is done by people hiding their identity. And this of course is a classic mixed blessing of the Internet. There are so many valid, useful, constructive purposes for allowing people to be anonymous, to protect them, for example, from oppressive governments and the like.

But that anonymity gets abused, as often as not. And the Internet provides a uniquely powerful anonymizing capability. We know it's not perfect. We've talked about the fact that it wasn't designed to provide that. But there's a big difference between throwing a physical rock through someone's window in the real world, where you can be bodily grabbed by a well-meaning neighbor or by a passing police officer, versus hiding in the basement in some remote country and having a great deal of power to do harm.

Leo: I guess that's the question. Is this just - there have always been people like this, and just now they have the means?

Steve: This is definitely an enabling technology.

Leo: That's for sure true. But I also feel like there are more people like that than there were before. I think we've raised a generation of kind of asocial, as a result amoral people who are good with these technologies and happy to use them because they don't have any empathy at all. They just lack that gene, or they've never been trained to because they spent their whole time playing Call of Duty in Mom's basement. I don't know. Maybe that's just me. I sound old. Forget it. I'm just, you know, old person. Get off my Internet, you kids.

Steve: Yes, well...

Leo: Who knows? No one knows. It's one of those questions for the era, the ages.

Steve: I love the position we're in, able to watch this happen, to cover the evolving news. To, like, we've all been here.

Leo: This is what we do, yeah.

Steve: Yes. We've all been here, for example, pre- and post-Snowden. And so we're riding through, I think, a fascinating period where we're going from the concept of a global autonomous network, and it's becoming real. It's becoming vital. It mean, in some cases having access to the Internet is considered, you know, not a privilege, but a right, and a necessity. It's like, I mean, it's that big a deal. I'll never forget when I first proposed GRC taking credit cards to my little group. They looked at me like I was crazy. And I said, "What?" And they said, "Well, no one buys anything on the Internet." And I said, "Well, I do." And they said, "Well, you're weird."

Leo: Isn't that funny? Boy, that's changed.

Steve: Yeah. I used to get Christmas presents from Amazon because I was like...

Leo: Everybody buys everything on the Internet now.

Steve: ...one of their launch customers. And, yes, and so look at just in this period of time how much has changed. I mean, you look around restaurants, when everyone's just holding their phones up and tapping into them. It's amazing.

Leo: What a world. What a world.

Steve: So I think this is a perfect time for us to be watching this. And I love the podcast because it is chronicling, week by week...

Leo: Absolutely.

Steve: ...the nature of these changes.

Leo: That's why I've always loved doing this; right? We've got ringside seats at the revolution. Yeah.

Steve: Yeah. Okay. So get a load of this one, and this is one I did not see coming. The question this poses is why did the CIA, assuming that we believe another Vault 7 doc dump from WikiLeaks, why did the CIA create a bogus software upgrade? The answer, apparently, to steal data from the FBI and the NSA.

Leo: Oh, god. Oh, I hope this is not true. Do you think this is true?

Steve: Okay, so here's what we think. So we sort of assume that we're all on the same side.

Leo: Yeah. We know this rivalry, but it's like rivalry of the Marines versus the Army or something. Collegial rivalry, yeah.

Steve: Precisely. That's what we want. And that, for example, there's automatic interagency cooperation and data sharing, despite the fact that they're different organizations. But we've also observed the tendency of agencies to protect their own interests, sometimes at the expense of the larger picture and the greater good, doubtless justifying their actions by telling themselves that this is what they need to do.

So if we are to believe the recently leaked, so-called "ExpressLane documents" as part of WikiLeaks' most recent and ongoing Vault 7 document dump, it appears that the CIA embedded its own trojan into an interagency biometrics collection system installed at so-called "liaison services" and intelligence gathering partners, including the NSA, the Department of Homeland Security, and the FBI. Again, if these documents are to be believed, the CIA didn't trust its security service partners to fully share biometric information with it, so it created a bogus software upgrade to steal the data.

This data-stealing trojan was created as part of a CIA project called ExpressLane, a piece of software installed by the CIA Office of Technical Services, the OTS, under the guise of upgrading the previously installed, CIA-provided, biometric collection system. The CIA installed the biometric system at partner offices around the world and expected them to voluntarily share biometric data with the CIA. But in case they didn't, it also installed ExpressLane to, quote, "verify that this data is also being shared with the Agency," Agency with a capital "A." It also had a feature to cut off the liaison service's access to the system if it didn't provide the CIA with access.

The documents note that: "The systems are provided to liaison with the expectation for sharing of the biometric takes collected on the systems. Some of these biometric systems have already been given to the liaison services. The OTS plans to revisit these sites with the cover of upgrading the biometric software to perform a collection against the biometrics acquisitions." So that OTS agents could install the trojan in the presence of partner agents, ExpressLane included a splash screen with a progress bar made to look like an authentic Windows install. OTS agents would install the software with a USB stick and could set the installation time of the update as well as a kill date before visiting the target.

Once installed, the trojan collects relevant files and stores them in a secret partition on a specially watermarked thumb drive that the OTS agent inserts during a subsequent maintenance cycle. The biometrics system itself was provided by U.S. identity management firm CrossMatch. It specifically didn't want the update to reference CrossMatch's software.

So I suppose that our lesson here is that spooks will be spooks. And while we are all still on the same side, the fact that we have agency divisions inherently creates some degree of interagency rivalry and tension. And so something like...

Leo: On the other hand, I don't fully trust Julian Assange and stuff from WikiLeaks.

Steve: I know. And this is the...

Leo: We know that Julian is also very interested in sowing discord among the agencies. And what better way to do that than this?

Steve: Yes. Yes, I'm glad you mentioned that because this is the great danger with anything coming from WikiLeaks. And all security people appreciate this. Back when all of the Hillary email controversy was up in the air and stirring, the problem was that it would be easy for a nefarious actor to alter legitimate email or to slip in particularly damaging email among a bunch of otherwise boring stuff about her daughter's wedding and so forth, and be able to ride on the credibility of the rest of the known good stuff. So, yes, I'm glad you said that, Leo, because we do need - that's why I made a point of saying, if we are to believe this, then this is what it's saying. So we don't know.

Okay. One more before our next break. And, oh, boy. There's a company we've discussed in the past, Zerodium, which is kind of a fun name, Zero as in zero-day, Zerodium, offering now, get this, half a million dollars for secure messaging app zero-day vulnerabilities which are exploitable. The discovery and private reporting - and this is coverage from Threatpost.

"The discovery and private reporting of exploitable zero-day flaws in mainstream secure messaging apps now brings with it a payment of half a million dollars. Last week, Zerodium" - whom we discussed before - "a vendor operating in the nebulous exploit acquisition market, updated their pricing structure to put a premium on zero-day vulnerabilities in secure messaging applications. Both remote code execution and local privilege elevation zero days in messaging apps - including WhatsApp, Signal, Facebook Messenger, iMessage, Telegram and a few others - can fetch as much as $500,000 from the company's new program."

As we've covered here before, Zerodium purchases zero days, then makes them available in a feed of exploits and defensive capabilities to its paying subscription customers. And of course those attacks and vulnerabilities are never shared with the affected vendor since they would then be promptly patched and made valueless. So in this dark ecosystem it is in the interests of everyone within the "supply chain," in quotes - the discoverer who wants top dollar for their discovery, Zerodium who is in this for profit, and the exploit purchasers who want to use the exploits obtained from their subscription as long as possible, until they become patched.

So it's in everyone's best interest to closely guard and protect the secrecy of the vulnerabilities they are leveraging. Thus this chain survives. Zerodium's official policy is to sell only to democratic and non-sanctioned governments. But one does wonder how tightly that policy is maintained when cash is dangled, since profit is the underlying motivation here. So they could well just be saying that for face saving. I mean, again, they're buying exploits and selling them and not reporting them. Their pricing changes were primarily aimed at the mobile platforms. And it's interesting, too, that that's the premium dollar now is mobile messaging. This suggests that's where the action is.

The company is also offering half-a-million-dollar payouts for remote code execution and local privilege escalation bugs in default mobile email applications; $150,000 for baseband, like we were discussing recently the Qualcomm firmware problem for baseband; and media file or document remote code execution and local privilege escalation attacks; $100,000 for sandbox escapes, code-signing bypasses, kernel local privilege escalation, WiFi remote code execution, and also SS7 attacks, which we know is the Signaling System 7 for the whole cellular network.

Zerodium's founder, Bekrar, told Threatpost that government customers are in need of advanced capabilities and zero-day exploits that allow them to track criminals using these secure mobile apps. And of course governments have the deep pockets required to foot the hefty exploit subscription cost that Zerodium must be levying in order to offer such substantial payouts, that is, they're paying half a million dollars for an exploit that doesn't have a super long lifetime, as far as we know. Again, we don't know. But so they've got to have multiple governments on the back end willing to pay serious money. And who knows, like, what granularity. Like do they have one contract with the U.S. government that then shares it among our multiple intelligence services? Or is it a per service agreement? Who knows?

Anyway, Bekrar said: "The high value of zero-day exploits for such apps comes from both a high demand by customers" - I choke on that word, by "customers" - "and a small attack surface in these apps" - meaning they're secure, and there's not much to go after. It's not like an OS where it's way, you know, like a desktop OS - "which makes the discovery and exploitation of critical bugs very challenging for security researchers." In other words, because Google and Apple are really doing, like, they're putting a lot of focus and concentration on security, the consequence is greater security. Over time it keeps getting better. So that attack surface gets smaller and smaller, and consequently it's harder to find problems. But when they are, they're worth more money. So in other words, they're both rare and valuable.

Zerodium also announced that it would offer $300,000 for Windows 10 remote code execution zero days, specifically remote exploits targeting default Windows services such as - yeah, wouldn't you like these - SMB and RDP. That's, of course, the Server Message Blocks that WannaCry exploited to such powerful ends, and RDP is the Remote Desktop Protocol, the remote console services. Also web zero days they're paying for, specifically Apache on Linux and Microsoft IIS remote code execution attacks, are now fetching $150,000, while a Microsoft Outlook remote code execution is worth $100,000. Mozilla Thunderbird remote code execution and VMware ESX guest-to-host escapes, which are of course in the case of VMware sandbox escapes from the VM, both pull $80,000.

Threatpost notes that Zerodium also doubled, or nearly doubled, payouts for Chrome, PHP, and OpenSSL attacks; while Tor remote code execution on Linux and Windows climbed from $30,000 to $100,000 and $80,000, respectively. Nearly a year ago, Zerodium tripled the bounty it offers for an Apple iOS 10 remote jailbreak to 1.5 million, after previously offering 1 million for iOS 9 zero days.

So what we're seeing here is the emergence and continuing evolution of a predictable for-profit ecosystem created by the tension which currently exists between uncrackable crypto and multiple governments' perceived need to monitor whatever they choose of their own and other citizens' activities. I feel this is not a stable state of affairs. One way or the other, this is a transient, created by the post-Snowden and comparatively sudden rush to encrypt after the world learned exactly how much monitoring was actually going on. It's going to be fascinating, as we were saying before, to watch how this fundamental tension resolves itself in our near future because, one way or the other, I think it's going to have to be resolved. In the meantime, this happens. And, boy, apparently the market is strong.

Leo: The market is strong. Well, and this is what happens. Then you don't get - the companies don't get the information.

Steve: Right.

Leo: The bad guys get it.

Steve: Right.

Leo: Steve? Now what? Now what, Mr. Gibson?

Steve: Now what? Ah, yes. So, okay. We have a deprecated, insecure Apple authorization API that can be abused to run code with root privileges. This is some more coverage from Kaspersky's Threatpost and paraphrasing from their coverage. They write: "A deprecated Apple authorization API, invoked by third-party installers, is still developers' preferred choice for updating apps and services on macOS, which has a problem due to a massive security issue that could be abused by local attackers to elevate privileges to root with a little unwitting help from the user."

The situation is known and was raised again last month during DEF CON by noted Mac security researcher Patrick Wardle, who's the chief security researcher at Synack - who happens to be down here in Southern California not far from me - due to the ongoing use of the API. It's a function called AuthorizationExecuteWithPrivileges. Installers for many popular applications including Slack, Google Chrome, Google-owned Dropcam, VMware Fusion, numerous security software updaters, and open source update library Sparkle all use this deprecated API during installation and updates.

At DEF CON, Synack's Patrick Wardle explained that the API causes the system to display the familiar authentication dialogue box. So, I mean, this is part of the OS, which is handled by a separate daemon running in the OS, meaning that the user doesn't have to entrust the application installer with their password. So you're giving your password to this other piece of the operating system, which then works on behalf of the installer in this area so that you're not trusting the installer, which is, you know, that part is good design. The operating system passes that trust to the daemon, and any functionality needing admin or root privileges upon installation may then proceed without the installer having it.

However, this AuthorizationExecuteWithPrivileges does not validate what is about to be executed on the machine to verify that it was not maliciously modified. Therefore, an attacker that is already present on the computer and has some code running can wait patiently for one of these third-party installers to call upon this insecure authorization API and piggyback off the user's credentials as they are entered into the dialogue. So let me be clear that this malicious software also cannot access the credentials themselves. But, as I'll explain in a second, due to the fundamentally insecure design of this, something that's co-resident in your system can use this window of opportunity, thanks to the fact that this API does not verify what it's being asked to do, to do the malware's bidding, as well.

Okay. So Wardle said, quote: "Normally what happens is these applications ask the operating system to execute something as root, and what they ask to execute is writeable by everyone, like something that's in a temp directory or the downloaded application bundle." And of course it has to be writeable by everyone because they needed to be able to write it. So there's unprivileged access to the thing that the installer wants the OS to then trust.

"But this means that any running and patient local code, malware, or a local attacker who already has some access to the device, can alter what's about to be executed with root privileges because, by definition and necessity, non-privileged processes have access to what's about to be executed. Since the OS does not verify what the application requested to have executed wasn't modified, when the user puts in their credentials and clicks install, the system will execute whatever was requested, even if that has been maliciously modified."

Wardle said the situation extends to Apple's own installer, as well, which suffers from a separate but related issue. He explained that, as a user double-clicks on a package file, a .pkg, Apple's installer app executes and looks for plugins in the file and copies them to temp, loading those libraries into the process context of the installer application, which is signed by Apple and trusted by the OS. But note that the libraries themselves aren't either signed or trusted, yet they're still stuck into this trusted application space, giving it a lot of power.

Wardle said that a local attacker can win a race condition and modify those packages before they're installed on a local system. The malicious and unsigned dynamic libraries, since they're not verified by the installer app, will be blindly loaded, Wardle said. Those malicious "dylibs" can create their own easily forged authentication popup that will be trusted by the OS. The user will then enter their credentials, unwittingly giving malicious code permission to run as root on the machine.

Now, here's the sad thing. Apple understood the problem and consequently deprecated this AuthorizationExecuteWithPrivileges API four years ago, back in 2013, and now recommends that developers use a different solution for the same problem called SMJobBless. SMJobBless works by copying whatever is being asked to install into a higher privileged directory. And once it's in that secure directory, SMJobBless cryptographically validates that it has not been tampered with, and only then asks the user to authenticate. In other words, they really did solve the problem.

So somewhere in 2013 this came to their attention, and they said, ooh, you know, the way we've been doing this, not so good. So they fixed it; right? Okay, so what's the problem? Using the much more secure SMJobBless comes at a cost, literally, because developers choosing to use SMJobBless must obtain an Apple developer certificate to perform the required cryptographic signing.

Synack's Wardle, who is clearly no slouch, said, and I like this: "The problem is that there is a secure solution" - meaning SMJobBless - "but it costs money and," he wrote, "it's incredibly complicated. It took me," he wrote, "several hours to get it working; whereas the deprecated and insecure API is literally, like, three lines of code." He said: "It's really easy, and that's why everybody does it still."

Wardle said he privately disclosed to Apple that its installer was loading unsigned libraries, but Apple did not respond. He also reported the API issue to Google with regard to Chrome; and Google responded that it was aware and that, quote, "No good replacement exists," including, in their opinion, SMJobBless.

Wardle concluded, saying: "Looking at the secure replacement, it's such a pain to get it working. There's a massive tradeoff. In an ideal world, everyone would use the secure replacement, or Apple would provide a secure replacement that would be easier to use. When you have the Chromium guys saying this isn't a valid replacement, they generally know," he says, "what they're talking about."

So this is admittedly a difficult problem to solve. We know there is no replacement for credentialing developers. It's just like credentialing web servers, which we all do. That's what TLS and HTTPS is for. If we don't do that with web servers, there's no way we can trust their assertions. And this is an example of why Let's Encrypt is an improvement, but not a replacement, for full credentialing. As we know, Let's Encrypt promises nothing more than the re-use of the existing HTTPS TLS system for transiting traffic within encryption privacy. But this is, and has been, massively abused by malicious actors. Which is why Let's Encrypt hasn't and doesn't threaten the established certificate authority industry, who continue to provide a service that cannot be automated because, if it could, it would be abused.

Similarly, Apple cannot blindly offer free developer certificates to all comers; otherwise, all would come, some would abuse, and no assurance of enhanced security would result. And just as traditional certificate authorities need reasonable payment for the non-automatable and necessary scrutiny they provide, so, too, does Apple.

But that said, it does seem as though Wardle's point about the comment from the Chromium developers is a very good one. Some money needs to be paid, but then the rest should be easy. And certainly Apple is clever enough to make it so, if they gave it the priority that I hope the light that has been shined on it now at DEF CON, thanks to the work of these Synack guys, may light a fire under them in order to make it happen. It feels as though their first pass at this, or I guess really it's their second pass, was not sufficiently well thought through. They may have been in a hurry to fix the problem that was discovered. What they came up with was something that, because it's so burdensome, nobody bothers using it, even though it exists.

So let's hope they get that fixed and reduce it to a line or two of code and then a potential vulnerability that we're all living with now. I mean, this isn't patchable. This is not something Apple can fix. They need to fix the underlying API and ultimately deprecate, go beyond deprecate, they need to remove this long insecure, but still available and preferred, API.

And of course Apple gets heat for this. A lot of developers complain that this is one of the problems with Apple is that they're rolling forward, and they're killing off old stuff that's breaking existing code. It's one of the things that Microsoft has not chosen to do over time. As a consequence, you know, they're dragging all this legacy stuff behind forever. But at least old stuff still works. Apple has decided to take a different approach. And here's an example of where, well, at some point they're going to have to start warning people, this is going away. And we mean it. But before they do that, they need to come up with a sufficiently useful replacement.

So, but again, this exists today; and now a bright light has been shined on it. And so, yikes. Maybe there can be some sort of a compromise. I don't know. Maybe they can strengthen the existing API that everyone's using. After failing to get everybody to switch over to the new one, go back and say, well, okay, we've got to do something about this. But the problem is unsigned code. If you don't require signed code, and thus make the developers who signed it responsible for their own code's action, then you're never going to have a secure ecosystem. There's just - there isn't a way.

So again, it ought to be, you know, get a certificate to sign your work, and it ought to be a reasonable price, but then easy to use. All of my code is signed with an Authenticode certificate from DigiCert, of course. And it's easy for me to do. When I'm developing code, for example, SQRL, every release I've offered of SQRL has been signed. So I'm working myself. When I have something that I want to push out to the gang to test, I just type "sign it," and it does. I mean, so not a big problem if it's designed right.

Okay. We have spoken many times about what I would call - I guess my first instinct is to call it a "mixed blessing," but I'm not sure it's so mixed. It's an evil necessity, how's that? The Intel Management Engine, the IME, this separate processor which we know is running in the southbridge chip with its own firmware, which has to be there, that is, our motherboards have become so ridiculously complex now that the processor needs a processor.

And that's what we've got. We've got a processor processor. We've got something that fires up when the plug is plugged in. Not even when you turn it on. The machine isn't on. The fans are not spinning. The main processor is not on. But the motherboard's processor is lit up and thinking about something. And, you know, there are things it's easy not to appreciate. Leo, you and I will, and all the old-timers among us will, notice that all of the interrupt jumpers have gone away. The IRQ...

Leo: Thank god. Thank god.

Steve: Yes. You know, the COM ports. Oh, my god, I ran out of COM ports, and I can't get any more. Or I've got four COM ports, but only two IRQs; and my COM software is confused because I'm trying to share one IRQ on two ports; and I have to rejumper and rejigger and blah blah blah. You know, all of that. And then there was the memory jumpers. We had to be flipping little dipswitches back and forth in order to move segments of memory around. Those were the good old days.

Leo: Are you saying that the Management Engine replaces that? Is that how they did it?

Steve: Yes, yes, it just all disappeared. Just like, okay, we'll just, you know, people shouldn't have to worry about this. Code should do this. And so, yeah. So it sets all this up. It enumerates the bus. It goes out and scans around. It finds boot ROMs in the different PCI slots, and it figures out what order they should be executed in based on automatic interdependency resolution. I mean, we plug the plug in, but that thing is busy. And it also runs the clocks. It's the reason, for example, BIOSes are, like, amazing.

You look at this thing, it's like, wait a minute. I could just type some numbers and change the frequency of everything. I can change weight states on DRAM and everything. That's all soft now, thanks to this stuff. So the point is you really need it after you plug your motherboard in. But the concern has been you really don't need it afterwards, yet it hangs around. And our listeners will remember that I chased down, I chased for a week or two a bizarre problem I had when I added another server to my network about a year ago now. Something was causing an ARP collision because two different things were, I mean, I never even really did figure out what the problem was.

What I ended up doing was finally thinking, okay, I give up. I don't know what is happening. So I just moved the network interface from the primary NIC to the secondary NIC, and all was well again. The problem just went away. And then it was because I realized that primary NIC was the Intel Management Engine interface, and it was busy messing with my network. I mean, there was nothing I could configure. It was all magical and transparent, and in this case wrong. So we've talked in the past about the concern people have had about the security of this, the fundamental security, because Intel won't tell us what's in there. It's all proprietary and magic and closed. And they've gone to some measures to keep it that way. And as we know, there have been exploits. There have been defects found. We talked about one a few months back.

So the guys over at Positive Technologies have discovered that it is in fact possible to shut down the vast majority of Intel's worrisome IME motherboard integrated subsystem. The story is the same one we often tell here. Some things can be protected; others cannot. The microcode firmware in Intel's preboot environment must be decompressed when it is pulled from storage. And that decompressor code cannot itself be compressed because it's got to do the decompressing. Therefore, if the decompressor could be located and reverse-engineered, then the secretly compressed preboot system firmware can be obtained in decompressed form.

It turns out the Intel firmware is compressed using something we haven't discussed for a decade, but we did once: the variable-length Huffman coding system, which as it happens every version of SpinRite has employed for the past 30 years to compress its own highly compressible user interface data. So I built my own Huffman compressor for SpinRite 1.0 because I know that a user interface is highly compressible, and that's just who I am. I want to compress things when I can.

Huffman coding is an incredibly elegant and efficient scheme. It does not provide high levels of compression. So it shouldn't be confused with state-of-the-art block sort compression and zlib and things, which are far more sophisticated, but far more complex to implement. Huffman coding, I mean, my algorithm is just, like, 20 lines of assembly language. So, you know, it's just it's tiny, which is why it makes so much sense to use it.

What it does is it takes an alphabet of compressible things, like, for example, machine language bytes, or in my case user interface text and tokens, and it performs a frequency analysis. It turns it from - you could think of it sort of as in time domain to frequency domain. That is, it determines how many of each different token occur. And if they're absolutely uniform, then it's not possible to compress it at all because the scheme relies on the fact that text and, to a big degree, machine instructions are highly non-uniform. As we know, "T" and "E" occur with great frequency in English. And, you know, "Q" occurs with much lower frequency.

So what the Huffman system does, just magically, it's just so cool, is it automatically assigns the optimal length number of bits, which are shorter, for the tokens that occur most often, and longer for the tokens that occur least often. So it converts from a fixed-length encoding, where for example all bytes are 8 bits, to a variable length encoding, where those 8-bit bytes which occur most often end up being represented by just a few bits, and the right number of few bits, based on the inter-token competition that automatically exists. Anyway, so that's what Intel did, for the same reasons I did. They chose Huffman because it made sense just to squeeze their firmware down. But in the process, it also encoded it. It turned it into gibberish.

Now, it's not super strong encoding. A cryptographer would just cut through that in a moment. But what these guys did was they managed to find the Huffman coding table. Because what you do, when you compress with Huffman, you end up creating this translation table. That has to go along with the compressed result because that's the key used for decompressing what was compressed with it, unlike for example the Lempel-Ziv-based compressors, where their elegance is that you don't need to ever send the dictionary, as it's called, along with it because it dynamically builds a dictionary on the fly.

In this case, because this is such a simple technology, you do have to package the dictionary along with the decompressor. These guys found the dictionary, understood what it was, and then took the Huffman-compressed and -encoded firmware for the microcode and decompressed it. And then they began looking around.

So the page that I've linked to in the show notes, if anyone's interested, provides a painfully deep and very comprehensive dive into the weeds of what they did and what they found. But the short version is they discovered something known as the High-Assurance Platform, or HAP for short, and a bit flag for enabling it, whatever it is. They were unfamiliar with the term. But as they wrote, a bit of googling quickly revealed the truth. The second search result said that the name belongs to a trusted platform program linked to the U.S. National Security Agency, our NSA. Not, apparently, for any nefarious reason, which I'll explain.

After additional research and lots of experimentation, they determined that the setting of this bit flag caused the Intel management engine to be shut down once it had performed all of the necessary work of powering up, configuring, and starting the system, and giving control to the platform's bootable operating system. After that, bye-bye. In other words, the NSA is no more happy than we are to have this potentially buggy, undocumented, secret, and exploit-prone hidden subsystem always running in the background on our motherboards while it has full access to everything in the system.

The NSA, however, has a bit more negotiating leverage than we do. So they were able to get Intel to acquiesce and provide them with this undocumented and well-protected option for completely shutting down the IME immediately upon the completion of its power-up work.

Wanting to be absolutely certain of their findings, the Positive Technology guys reached out to Intel for comment and received the following reply. Intel quote: "In response to requests from customers with specialized requirements, we sometimes explore the modification or disabling of certain features. In this case, the modifications were made at the request of equipment manufacturers in support of their customers' evaluation of the U.S. government's High Assurance Platform program. These modifications," Intel writes, "underwent a limited validation cycle and are not an officially supported configuration."

Except they are. But, yes, we understand what Intel is saying. In other words, yes, you found it, but we take no responsibility for its use. And, after all, we did do everything we could to hide it from everyone to prevent its discovery. But it's there. I don't know if what will surface will be a tool that lets us do that. I wouldn't be surprised.

Positive Technologies' current posting is detailed. And they wrote at the beginning that that posting is only the first of several. They're apparently going to tear this thing down to its constituent bits. And it may well be that - I don't know if it will ever be user-accessible without hardware. There's something called the SPI, which is the Serial Programming Interface. And Leo, you're familiar with it. You actually use it because...

Leo: I do?

Steve: ...that's how I've updated my firmware of my various gadgets over the years.

Leo: Oh, okay.

Steve: That's that programming interface, very common. It's a standard. But it is hardware.

Leo: Oh. Oh, yeah. That's the thing. Okay. I know. I use it for that little whatever that is, the match chip or whatever, the EEPROM. Yeah, the EEPROM, yeah.

Steve: Exactly.

Leo: Steve and I have a secret thing, a little bit of hardware. And every once in a while he'll send me - yeah. Okay, cool.

Steve: Yeah.

Leo: And I have a little Linux command-line utility that does it. But there's [crosstalk].

Steve: Yeah. And so that is - but it does require an SPI programmer, not something that you can do by setting a jumper on the motherboard of our PCs or so forth. So anyway, interesting that in fact all this grief that we've been suffering, and the angst, and here all along there is a program that says, you know, we don't want that left running for no good reason after you've got everything up and going. So turn it off.

Leo: There'll be an open source tool that takes advantage of this soon.

Steve: I wouldn't be surprised. I think we're going to see something surface.

Leo: Yeah. So to speak.

Steve: So anyway, nice piece of work from the Positive Technologies guys. So I wanted to keep this on people's radar because this is important. So this is not news, per se, except it's another "this actually happened" event. John Biggs, who is a writer, consultant, programmer, and the former East Coast Editor and current contributing writer for TechCrunch, wrote, and I lightly edited: "At about 9:00 p.m. on Tuesday, August 22nd" - so a week ago today - "a hacker deauthorized my phone's SIM card, replaced it with his, presumably by calling T-Mobile. This, of course, shut off network services to my phone and, moments later, allowed the hacker to change most of my Gmail passwords, my Facebook password, and to impersonate me in texts sent under my identity."

Leo: Oy.

Steve: I know. "All of the two-factor notifications went, by default, to my phone number, so I received none of them; and in about two minutes I was locked out of my digital life." He wrote: "I noticed all of this at about an hour later, around 10:00 p.m., when I assessed the damage and called T-Mobile. By 10:30 I had restored my phone's SIM and began the process of changing all of my passwords, hardening my two-factor accounts and my T-Mobile account, hopefully ensuring that this would not happen again." But he wrote: "Sadly, I worry it will."

He said: "My hacker was thorough. In the course of a few minutes he or she did a quick search of my Facebook Messenger messages and assessed that I was originally from Ohio and that my Dad was sick. He or she used this information to approach people I knew in the cryptocurrency space with a story that was arguably ludicrous. The hospital would pull the plug on my father if they didn't get payment of a bill; and that I, in my anguish, needed to borrow and sell 10 bitcoins immediately and would pay the friend back 15 the next morning. Luckily," he writes, "my friends weren't idiots and immediately texted me and my wife.

"The hackers' IP (173.239.232.29), which points to LogicWeb of Plano, Texas, along with a breadcrumb noting a login from Florida, suggests that the hackers were from the United States. They clearly had the modus operandi down because they also hacked two other friends of mine in the space of a week. This all came about, I believe, after another friend in the cryptocurrency space was hacked last week. That hack" - so that would be two weeks ago. "That hack bore all the same hallmarks as this one except the SIM hijacking.

"First, the attacker grabbed access to my friend's Facebook Messenger and contacted everyone on his list that was interested in cryptocurrency, including me. In the ensuing melee, the hacker asked me to send 10 bitcoin and that he would send me 11 back in the morning. Confused, I told them that I had some bitcoin, but not that much. I then realized the ruse and asked" - I love this - "'Did you talk to Wallace Shawn yet? He can help. I think he's having dinner with Andre right now.'" Now, of course, for those who don't recognize the references, Wallace Michael Shawn is a well-known actor and comedian who starred in "My Dinner with Andre." "Anyway, the hacker claimed that Wallace wasn't available. I knew I'd been had.

"This interaction led to my own subsequent hacking. Once it was clear that I had some bitcoin somewhere" - because that he had admitted in the interchange - "the hackers decided I was their next target." He says: "Ultimately I got away lucky. Nothing major was stolen as of today, and I took control of my accounts back fairly quickly. I had," he writes, "some two-factor set up; but because my phone was compromised first, I lost access to most of it. I've since activated authentication apps" - and this of course is our takeaway lesson here - "for all of my accounts. The biggest question is how the hackers took control of my SIM card in the first place. This is the most troubling; and T-Mobile is, quote, 'looking into what happened.'" Uh-huh.

Leo: They have recordings of it. And what they'll find is what always happens. A customer service rep...

Steve: Yup, yup.

Leo: ...wants to serve the customer.

Steve: Social engineering.

Leo: They have very nice, very friendly customer service reps who work hard to make you happy. They believe it's you, even if it's not.

Steve: Even if you don't provide sufficient evidence. Oh, I forgot my mother's maiden name. I just - I used to know it, but it's been so long.

Leo: Well, I did that recently, and all they required was the last four of my Social.

Steve: Wow.

Leo: And that isn't that hard to find. I mean, you know, and it's not a really good way of validating, frankly.

Steve: Well, and of course we know that in the U.S. the Social Security number has long been used as an identifier when it's specifically supposed to be excluded from that purpose. But too many people ask for it, and you don't often have a choice not to give it. So our takeaway is this: While it may be unlikely for everyone, targeted attacks can be quite powerful and successful. The new hacking method is to arrange to take over the communications channel, whatever it is, used for account verification and/or second-factor authentication.

This is why time-based one-time passwords, not communications-based one-time passwords, are the only safe choice to make when a choice can be made. And this is exactly what he did when he said he switched to authentication apps, meaning those apps which automatically give you a changing six-digit code. Then it's not communications-based after the initial setup, which is over a secure channel, typically a QR code presented to a web page. You snap it with your phone, or in my case you print it so that you're able to take it offline and then populate other apps in the future. And we'll be discussing that, actually a variation on that here in a second.

So the problem, of course, is that not all services offer this. And we can hope that, as more such instances of this exploitation do surface, more providers will begin offering time-based one-time password, or so-called TOTP, authentication options. So I appreciate him sharing this and helping to keep this awareness because it's sad, but the reality is, as you just verified, Leo, that there just isn't enough protecting the ownership of our cell phone accounts. And if sending anything, either to email that can then be intercepted by the mail on the phone, or by a text sent to the then-registered number of the phone, then that can be a problem.

And speaking of time-based one-time passwords, we have Authenticator Plus, which was brought to my attention just by reading feedback from our listeners. It's at www.authenticatorplus.com, and it looks very nice. Everyone knows that my preference is to print, maintain, and manage the QR codes for my one-time password authentications in order to manually curate and install them into new devices. But I get it that some people may choose the convenience of networked linkage and secure cloud backup which this offers.

So in their feature overview they said: "Secure: 256-bit AES encryption and PIN lock for additional security, along with hardware-backed encryption where available, which protects your account even in rooted devices. Syncing across devices: Apps are available for Android phones and tablets, iPhone/iPad, Android Wear and Apple Watch." So broadly cross-platform over in the mobile space. Under organization, group accounts and categories and the ability to reorder frequently used accounts to the top, change the themes, and display account logos for easier lookup.

And actually I really like that feature. It just shows you the icon. So, like, for Facebook and Google and LastPass and so forth, I just have text now under my Google Authenticator. I'm going to take a look at this. If it's possible to turn off the cloud backup, I would turn it off because I'd rather continue to manage my things offline on paper. But this otherwise looks like a nice piece of work.

On iOS it's free with in-app purchases. And I haven't looked any further than that. And so I don't know what - which makes me a little uncomfortable, like is it going to limit me to five sites unless I buy it? I mean, I'd just rather buy it. If I'm going to use it, if it's going to be a few bucks, fine, I'll buy it. So I would want to trust this person; but, as we know, that has to be earned. Their FAQ says all the right things about their intentions and policies. It's got all the jargon and so forth. So again, I can't recommend it because it doesn't have any reputation as far as us having some reason to really believe it's right.

But I don't have any reason to believe it's not. And the site looks nice and polished; and the FAQ, as I said, is saying everything right. So anyway, it's there, and I wanted to point to it.

Leo: It's the same as Authy, which I recommended, as well, which does very much the same thing and is free. Highly recommend Authy.

Steve: Good, good. And I agree. And Authy provides the various interdevice sync and cloud sync also?

Leo: Yeah, yeah.

Steve: Good. CrashPlan has shut down, or announced that it is shutting down, its home offering. They will stop accepting subscription renewals for the consumer version. They're just going to go focus on enterprise. And 9to5Mac had some coverage, writing: "Code42, the company behind CrashPlan, has officially announced that, as of today, they are pulling out of the consumer market. CrashPlan for Home users will have to begin migrating away from the service as it will no longer be available starting in October of 2018." So that's responsible. They're giving people more than a year.

In a message posted on CrashPlan's website, the company wrote: "Effective August 22nd, 2017" - so that's last week, last Tuesday - "Code42 will no longer offer new - or renew - CrashPlan for Home subscriptions, and we will begin to sunset the product over several months. CrashPlan for Home will no longer be available for use starting October 23, 2018." So that means that they will honor a subscription created just previous, a one-year subscription just previous to their decision to stop, through the end of its life, but you can't get it anymore as of last week.

"So Code42 seems to be handling the situation as delicately as possible by giving users ample time," writes 9to5Mac, "to prepare for the transition, and offering discounts on alternative services," which is very nice. "CrashPlan for Home users can transition to the Small Business plan and receive a 75% discount over the next 12 months." Of course then the price would jump up to full price after that. If a user wants to completely move away from CrashPlan, the company recommends Carbonite's services.

Now, many of our listeners have asked what they should do, so I'll repeat my recent advice from my own research and findings. First of all, I'll just say that Carbonite offers a similar system, that is, an all-in-one, integrated, drop it in and go. As I've mentioned before, that's perfect for Jenny. Jenny has that installed on her laptop. She doesn't know what it is or really why it's there. But it's just keeping her safe.

My advice, I just like the idea of some bifurcation. I get it that there's a place for an integrated, all-in-one drop-in. And I gave a perfect example with Jennifer. But I like to have these two things separate. Once upon a time, Boxcryptor was what I was liking because they got the crypto right. I took a deep dive into it, looked at it, verified that it was right. But they have since transitioned from a product to a service. So anyone's use now of Boxcryptor as a service is just as susceptible to them changing their Terms of Service or canceling it outright, exactly as CrashPlan just has.

This is why I don't see the benefit for a more technically sophisticated user of an integrated bundled service, again, unless you just don't want to worry about it. What I choose is a TNO software solution whose crypto was done right, then choose among any or all of the available cloud storage providers and be free to change your mind or to use any different provider or providers for different purposes at any time.

And that's why, as I said recently, CloudBerry Labs has become my current choice. Their crypto's done right. They offer one-time purchase and lifetime use of a product and not a service; and they support every cloud provider any of us has ever heard of, and many we haven't. I looked at their list, and it's like, whoa. I mean, they must have a team that does nothing but figure out the APIs of providers. And in fact, it may be that we're now beginning to see some merging of API. So as long as providers support this emerging standard way of accessing their services, well, then, you can use one thing everywhere, which is great.

But anyway, so Carbonite, if you want a good alternative turnkey solution. Or CloudBerry Labs is what I use personally, what I use at GRC. And Sue is still using Jungle Disk just because she always has been, and it still works, which is a turnkey solution also.

So my slip of the tongue, I can't believe I did this, and so the first note came from a Daniel, wow, and Daniel, I'm sorry I can't pronounce your last name, looks like Szmulewicz. Anyway, sorry, Daniel. He wrote: "You meant atomicity, not monotonicity, in the section explaining compare-and-swap." He said: "I hope you don't mind me saying." And I actually thanked him for his note and said, "Of course I don't mind you saying."

And then I thought, okay, that must have just been something I said once. Whoops. And then Chris Frederick, who has a more easily pronounced last name, said: "You talk quite a bit about 'monotonic' operations and functions in the latest podcast. I wonder: Did you mean to say 'atomic' operations?" And so, okay, as I said at the top of the show, I must have just started off saying monotonic, which is a term I use often when I'm talking about counting. Counters that only count upwards, incrementing, are monotonic. They increment monotonically. And so I apparently started off saying "monotonic," and I just let her rip and kept saying it. Yes, I meant "atomic."

The reason they're called "atomic operations" is they are indivisible by an interrupt. That is, if you do a separate compare instruction, then a swap instruction, that entire instruction pair is not atomic, it's a molecule, and so composed of two atoms. So you can split them apart with an interrupt which might occur between them, thus the problem. So these are called "atomic operations" because they are operations that are not splittable any longer. Anyway, thank you for bringing it up. And for anyone else who was confused, I apologize. I just, okay, maybe I do need my moustache. Oh, that was pre-moustache. That was when I still had the moustache.

Leo: No excuse. No excuse.

Steve: No excuse, yeah. Meanwhile, another of our listeners whom I hear from from time to time, the AspiringLockPicker, said: "I used my AutoIt UDF to break apart Steve's Security Now! pages to get at the minute and episode number of all shows and do the math. A fun few minutes. Security Now! has, as of this moment, 908.05 hours of total content, spread across 623 episodes." So not quite a thousand hours, 908. Just a shred over 908 hours of content. Yup.

Leo: A goodly number.

Steve: And growing at the rate of about two hours a week. And two interesting quick little points about SpinRite. I saw a note from an "xoff00." Boy, I haven't seen XOFF for a long time. XON and XOFF are two reserved ASCII non-printable control codes which were used to start and stop the paper tape reader. Or I guess maybe sometimes - no, I don't think it was optical. It was back in the teletype days. Something would send an XOFF if its buffer was filling, and it needed to stop the reader from going [making chunking sounds] as it was reading the holes in paper tape. Then the reader would catch up and then start again. And it's still - it's a "software flow control" is how it's better known. And so it exists in some terminal applications and so forth.

Anyway, he said: "A coworker just told me he regularly runs SpinRite on USB thumb drives. Your thoughts?" And it's interesting. I think it's a great idea. One of the things we know is that thumb drives have become incredibly inexpensive, super cheap. Unfortunately, they're cheap in both senses of the term. They're both inexpensive, and they are low quality. No expense beyond the bare minimum has been put into them, and people have problems with them. Running a Level 2 scan will not fatigue the thumb drive, but it will help to keep it alive.

And the idea is that, you know, the more robust SSDs are highly over-provisioned, sometimes as much as 40% over-provisioning, meaning there's an additional 40% of storage space you don't see. So it's actually, even though they say it's 64GB, that's the 100%. It's actually 140%, for example, in terms of physical storage. And the reason is that gives it resilience and life. And as spots go bad, as they will, then the reserved area gets pulled into use, making that aging and fatiguing transparent over time.

SpinRite, as we keep seeing, is great on nonvolatile solid-state media, almost perversely as good as it is on spinning media for which it was designed, because the underlying technology is still very similar. So, yes, if you are a heavy thumb drive user, giving a thumb drive a SpinRite Level 2 scan from time to time makes lots of sense. And I imagine doing so regularly will help to keep them from ever dying.

Leo: I should mention that our official count is 968 hours. I'm not sure where the other 60 hours come from.

Steve: Interesting. I wonder - I don't know whether he...

Leo: We may have counted all the special episodes.

Steve: ...actually listened to the, I mean, whether his data pulled from the metadata of the MP3s, or if he parsed my pages. Because I'm normally rounding down to - I just know I'm rounding. So I don't know. It could be rounding error or what. But that's a large difference. So I don't know what...

Leo: Patrick Delahanty, the keeper of the stats, says 40 days, seven hours, 59 minutes, and 18 seconds of content. Average running time 01:32:20.

Steve: My goodness.

Leo: Yeah.

Steve: And that average is going up over time, too.

Leo: Yeah. 629 episodes, including special episodes with nonstandard numbers. So the episode count is roughly right.

Steve: Yeah. Interesting. Well, I did say - I mentioned a couple weeks ago, I think it was recently, that one of the nice things about SpinRite was that it was so small. You could run it in a 640K VM, and it would be completely happy because it pulls in DOS, and it pulls in itself. I mean, it used to run and can run in a 640K memory environment. The problem is I don't think VMs go that small. You can't turn the dial down that much.

Leo: I'm sure they're at least a megabyte, yeah.

Steve: Anyway, so I got a tweet from a FlitBee, and I have the photo. I wasn't able to grab it off of - he just sent it in my Twitter stream, but I captured it. But I got a kick out of it. He sent: "SpinRite scanning two drives in a 128MB VM." And it shows two SpinRite windows side by side with SpinRite running away. And this is what's cool is that, unlike running whole operating systems that require a gig or more each, SpinRite is actually very practical to run in a virtual machine because it requires just hundreds of K. I may think about this maybe for the future.

My plan for after v6 is to, you know, the v6 series, the idea is to use it as the platform for developing the next generation of hardware interface technology. So all SpinRite 6 users will get the benefit of that immediately. But the goal is to come up to speed with the state-of-the-art hardware support, finally getting rid of the BIOS and all that. And that will then be a proven base upon which SpinRite 7 gets written to, and that's where we lose the textual DOS interface and DOS completely, and I roll up my sleeves and probably do a hybrid of assembly language and Python because everyone would rather have it sooner rather than later, and it just makes sense at this point to move us off to a higher level environment.

One quick closing the loop note from a Pete Blumenthal, who listens. And then we'll talk about "Shattering Trust." And I liked this. This caught my attention just as we were getting ready to go on-air, so I dropped this in at the last minute. He asked about my thoughts on voiceprint for phone authentication. And then he gave a link to a Fidelity.com something or other, TD Bank something. He says: "I see no technical details." But we don't need any. And I have a couple thoughts which I think are useful. First of all, it's clearly a bad idea if the phrase you are asked to repeat for your voiceprint is unchanging because anyone could make a recording and play the recording. But if instead you are shown something on the screen which you then must speak so that every utterance for authentication is unique, then it's interesting.

It's not that someone can't impersonate someone else's voice. We know we've heard very good impersonations. And we know that software, unfortunately, is rapidly acquiring that ability, the ability to do exactly that. But what I liked about those two counter examples, repeating the same thing versus being challenged with a unique statement to utter is one is like a password, where your password is unchanging. You always put the same thing in; so, for example, a keystroke logger can capture it.

The second one, the better security, is the challenge/response model where you are challenged in some form with something that has never been offered before and will never be repeated, to solve the replay attack problem. And then you must respond to that unique challenge properly and correctly. So anyway, I just liked that, Pete's question as a jumping-off point, because it's interesting to consider that, voiceprint as a biometric. Makes me nervous, especially if it was done wrong, and it could be done wrong if it was the non-challenge-and-response authentication, if it was just, you know, say your magic phrase, again, because that's trivial to capture and to repeat. So, cool question, though, Peter. Thank you.

And, finally, "Shattering Trust: When Replacement Smartphone Components Attack" is the title of the paper. Very interesting piece of work these guys did. Their abstract explains the concept. They said: "Phone touchscreens and other similar hardware components such as orientation sensors, wireless charging controllers, and NFC readers are often produced by third-party manufacturers and not by the phone vendors themselves." Okay. So in other words, our modern smartphone is a system. It's got a bunch of peripherals which are all in the same package, and we hold it all in one hand. But in fact those subcomponents come from somewhere else.

"The third-party driver source code to support these components is integrated into the vendor's source code. In contrast to 'pluggable' drivers, such as USB or network drivers, the component driver's source code implicitly assumes that the component hardware is authentic and trustworthy. As a result of this trust," they write, "very few integrity checks are performed on the communications between the component and the device's main processor." Which is where we insert: What could possibly go wrong?

They write: "In this paper, we call this implicit trust into question, considering the fact that touchscreens are often shattered and then replaced with aftermarket components of questionable origin. We analyze the operation of a commonly used touchscreen controller." In this case it was from an Android device. "We construct two standalone attacks, based on malicious touchscreen hardware, that function as building blocks toward a full attack: a series of touch injection attacks that allow the touchscreen itself to impersonate the user and exfiltrate data, and a buffer overflow attack that lets the attacker execute privileged operations. Combining the two building blocks, we then present and evaluate a series of end-to-end attacks that can severely compromise a stock Android phone using standard firmware. Our results make the case for a hardware-based physical countermeasure."

So in other words, there's currently an assumption of trust among the hardware subsystems of Android smartphones. I'm reminded once again of that classic scene, and here's where we talked about "The Wrath of Khan" from "Star Trek: The Wrath of Khan," where Khan was approaching under radio silence, and Kirk had not raised his shields. Spock cited regulations about the need to adopt a defensive posture wherever the intentions of an approaching ship had not been confirmed. Kirk waves Spock off. One of Khan's own minions commented that Kirk was not raising his shields, to which Khan replied, "Of course not. We're all one big happy Federation." Whereupon Khan blasted the crap out of the Enterprise.

Leo: What?

Steve: Oh, yes. So as we have seen many times, it's all about security boundaries and where those boundaries are placed.

Leo: But wait. Why didn't Kirk raise his shields?

Steve: Because, you know, he knew, he thought he knew the captain of the ship that was approaching, and they hadn't been able to establish coms.

Leo: Okay.

Steve: But he thought, huh, that's odd. He was sort of, hmm.

Leo: A strange lapse on the part...

Steve: That's when the first photon torpedo hit, and it's like...

Leo: ...of James Tiberius Kirk, a strange lapse of judgment.

Steve: Yeah, well, it was funny, too, because I think it was Scotty or somebody else on the bridge chided Spock for, like, telling Kirk what he well knew. And then Kirk afterward said, "You just go right on quoting regulations."

Leo: "It was only logical."

Steve: Not that I've memorized the...

Leo: I think you have.

Steve: ...dialogue of the show. As we've seen many times, as I was saying, it's about security boundaries and where they're placed. If a system's security boundaries are all-inclusive, that is, if there are no internal boundaries, then any adversary who manages to slip inside the system's outer boundary will potentially obtain unfettered access to the keys to the kingdom.

In their research paper, way later on, and I've got the link to the paper in the show notes if anyone's interested, they say: "Counterfeit components have been in existence ever since the dawn of the Industrial Age. Their effectiveness as attack vectors is also well known. What, then, is unique about the particular setting of a smartphone? We argue that our specific attack model is feasible because we assume only a specific component with an extremely limited hardware interface is malicious, while the rest of the phone, both hardware and software, can still be trusted.

"Furthermore, we assume that the repair technicians installing the component themselves are not malicious, and will perform no additional operations other than replacing the original broken component with an unknown-to-them malicious one. Hundreds of millions of devices satisfying this attack model exist in the wild. One can assume that these limitations make this attack vector weaker than complete hardware replacement. We show it is not. On the contrary, the nature of the smartphone ecosystem makes this attack model both practical and effective."

In other words, people are shattering their screens. They are taking them to third-party service organizations. They are having them replaced with aftermarket screens of unknown or non-OEM, not the original equipment manufacturer origin. And the firmware, what these guys did is, just by changing the firmware programming, they were able to completely commandeer and take over Android smartphones from the setting of the processor. Ah, you found the scene.

Leo: Just like Khan. I love Khan's extended family. There you go. We're all one big happy family, aren't we.

Steve: Yeah.

Leo: Yeah, mm-hmm.

Steve: So the authors followed responsible disclosure practices by disclosing the Synaptics device driver vulnerabilities to Google back around the middle of February of this year. The disclosure includes the details necessary for understanding, reproducing, and fixing the issues discovered during their research. Google acknowledged the reported issues and issued a CVE (the Common Vulnerability index) of 2017-0650, with critical severity. The vulnerabilities discovered in the Atmel device driver, so there was a little Atmel chip in the keyboard, are being compiled into a responsible disclosure report at the time of the submission of the paper. And they conclude, saying: "The threat of a malicious peripheral existing inside consumer electronics should not be taken lightly. As this paper shows, attacks by malicious peripherals are feasible, scalable, and invisible to most detection techniques."

Yeah, nothing would - you could scan the firmware of the phone till the cows came home. I mean, it's up there in the keyboard, where you would never think to look. "A well motivated adversary," they write, "may be capable of mounting such attacks in a large scale or against specific targets. System designers should consider replacement components to be outside the phone's trust boundary and design their defenses accordingly." So really interesting, I mean, fully fleshed-out proof of concept. Go to a service place, have the screen on your phone replaced, and at the moment it has the ability to take over the device. Trust boundaries.

Leo: And this is, by the way, why Apple is very reluctant to let third-party replace phones. And if they do, remember there was a problem. There was actually a bug in Apple's firmware that crashed, bricked the phone if the screen's been replaced. Remember that?

Steve: Well, remember, it was the Touch ID.

Leo: Because they felt it had been compromised.

Steve: Right. The people were getting their iOS devices swapped for screens. But since they couldn't reproduce Apple's proprietary fingerprint sensor, they were non-fingerprint sensor Touch ID, I mean non-Touch ID. And the phone said, "What?" And, you know, said, "No way, I'm not going to." Which is good. There is a strong security boundary. It said, "Third party? No." Now, mixed blessing, of course, because some people would say, wait, it's so much cheaper to use the third party than to use Apple certified service. And unfortunately, sometimes you get what you pay for, or even more than you paid for.

Leo: Well, part of this is also because Apple doesn't offer any third parties the official parts.

Steve: Right.

Leo: I mean, that might be another way to solve it. Apple prefer that you go to them.

Steve: Yeah.

Leo: Interesting, yeah. Don't use third-party parts for security devices.

Steve: And if an untrusted ship is approaching, always raise your shields.

Leo: Yeah, raise your shields for crying - especially if some guy not wearing a shirt. Well, I guess you didn't know that yet. That came about later. "Khan."

Steve: He was very macho, Leo.

Leo: He was Ricardo Montalban.

Steve: He was genetically engineered, yup.

Leo: Steve Gibson has been generally engineered to be moustache-free.

Steve: I wish. I hate shaving, so it'd be nice if it weren't going to grow.

Leo: I know. I know. Well, let's just all grow beards.

Steve: No.

Leo: Let's just do it. Let's just do it, just take it out of the equation. Of course then we would lose our sponsorships. But that's okay. That's okay. Steve is the man in charge at GRC, the Gibson Research Corporation. So if you get a bill for research, it's probably because you bought SpinRite, the world's best hard drive maintenance and recovery utility. That's at GRC.com. Everything else is free, though. You won't get a bill for the ShieldsUP! service or the work he's doing on SQRL or Perfect Paper Passwords and Password Haystacks, all that stuff is there and browsable at GRC.com.

So is this show. He's got the audio of the show and full transcripts at GRC.com. We have audio and video at our site, TWiT.tv/sn. But really the best way to do it is to subscribe. Use your podcatcher, whether it's Overcast or Pocket Casts or whatever you - iTunes, whatever, Stitcher, Slacker. Subscribe to Security Now! so you can have every episode, all 638 or whatever it is. It says, see, this is where he went wrong. It says 626, but there are actually more episodes than that.

Steve: Yeah. And there was like an "a," and some we skipped, and there was a little renumbering. I mean, you know, things have been very smooth for the last decade, but we got off to a little bit of a rocky start there.

Leo: When Leo was running it, it wasn't so good. Now that I have trained professionals...

Steve: That would be the case for my enterprise, were I going solo. I got an email from Sue, don't forget to get the car smogged because - it's like, oh, thank goodness for Sue. I mean, because, you know, I worried...

Leo: She told you to get your car smogged?

Steve: Oh, she runs my life. Yeah, she's wonderful.

Leo: Wow. That's a very handy person. Thank you, Sue. And thank you, Steve. We'll be back here next Tuesday, 1:30 Pacific, 4:30 Eastern, 20:30 UTC, if you want to tune in and join us. Look forward to that. And I guess that's all there is to be said about this show except we'll see you next week, Steve.

Steve: Thanks, Leo. See you for Episode 627 next week. Bye.