Plundervolt

Description: This week we start with a reminder about Google's still operating Sensorvault. We look inside Google's new "Verified SMS Messages" feature. We examine another salvo in the end-to-end encryption war, note a nice authentication feature added to iOS v13.3, and deliver some Patch Tuesday news. We discuss a startling discovery about the weaknesses of RSA at scale, a collection of quick bits about last Friday the 13th, Mozilla 2FA for add-on developers, the surprising hard out for Microsoft's Security Essentials, and two bits about Chrome 79. We have a clarification about last week's "VPN-geddon Denied" discussion, a significant announcement about my new focus, and some SQRL news. We conclude with a look at yet another interesting new way of compromising Intel processors known as "Plundervolt."

High quality (64 kbps) mp3 audio file URL: http://media.GRC.com/sn/SN-745.mp3
Quarter size (16 kbps) mp3 audio file URL: http://media.GRC.com/sn/sn-745-lq.mp3

SHOW TEASE: It's time for Security Now!. Steve Gibson is here with lots to talk about. Is Google watching you? Yes, they are, and what you might want to do about it. RSA certificates are broken, at least on some IoT devices. This is another thing you'll want to be aware about. And Patch Tuesday. What did we get? What are we getting? Coming up next, you're getting Security Now!.

Leo Laporte: This is Security Now! with Steve Gibson, Episode 745, recorded Tuesday, December 17th, 2019: Plundervolt.

It's time for Security Now!, the show where we cover the latest in security, privacy, ransomware, and Mr. Steven Gibson's mustache. Happy holidays, Steve.

Steve Gibson: As it gets ever whiter. Ah, yes, happy holiday, likewise.

Leo: Likewise.

Steve: So today's episode is titled "Plundervolt."

Leo: Plundervolt?

Steve: It is, believe it or not, the domain was available, so they grabbed it, and they gave it a logo.

Leo: Oh, my.

Steve: And we're going to talk about it for Episode 745, this lovely December 17th. It's lovely in Southern California. I assume it's nice up where you are.

Leo: It's a little gloomy, but it's not rainy. It's been cold. Has it been cold in Southern California?

Steve: Yeah, it's been, yeah. And we're so spoiled here in California. It's like, oh, darn, you know.

Leo: Cold for us is low 50s. Be clear. I was talking to somebody in Winnipeg who said that, yeah, it's 37 below at night. 37 below at night.

Steve: Below, yes.

Leo: That's cold.

Steve: So we have a reminder about Google still operating Sensorvault. We're going to take a look inside Google's new verified SMS messages feature. We've got another salvo in the end-to-end encryption war, went back and forth last week. A nice authentication feature which has just been added to iOS 13.3. We've got some Patch Tuesday news. A startling discovery about weaknesses of RSA at scale, that is, at Internet scale. Really interesting sort of a techie topic I think our listeners are going to enjoy.

Then we've got a collection, since there was so much of that, a collection of quick bits about last Friday the 13th; Mozilla requiring two-factor authentication for add-on developers; a surprising hard out for Microsoft's Security Essentials coming to everyone next month; two bits about Chrome 97, which is just out. And then we have a clarification about last week's "VPN-geddon Denied" discussion; a significant announcement about my new focus; some SQRL news; and then we conclude with a look at yet another interesting new way of compromising Intel processors known as "Plundervolt."

Leo: Plundervolt.

Steve: Plundervolt. Wonderful.

Leo: Wunderbar.

Steve: So I think it'll be another interesting podcast for our listeners.

Leo: All right, Steve. I've got your picture ready to go here.

Steve: So I actually had the logo for Plundervolt as our Picture of the Week because nothing else had occurred to me when I ran across this little tidbit that I thought, okay, this gets the Picture of the Week. The news is that coming soon to a Windows 7 system near you will be a full-screen, I mean, I guess it's when you boot the system up. It's not really clear when you're going to be assaulted with this. But it's not a little dialog anymore, it looks like Blue Screen of Death. And in some senses it is. It says: "Your Windows 7 PC is out of support."

So presumably that begins on January 15th. And I was a little curious by some of the wording. So anyway, it doesn't have very much on there because it knows people are like, what? Like someone doesn't know. Maybe they somehow missed all of the previous announcements of this. They said: "As of January 14, 2020, support for Windows 7 has come to an end. Your PC is more vulnerable to viruses and malware due to," colon, then three things - no security updates, no software updates, no tech support. And then here's the part that I thought, well, this is interesting. "Microsoft strongly recommends using Windows 10." That's no surprise. But then they go on to say, "On a new PC."

Leo: Yeah. Buy a new PC.

Steve: And it's like, what? "For the latest security features and protection against malicious software." And then there's a "Learn more," there's a "Remind me later," and then thank goodness in the lower left is a "Don't remind me again."

Leo: Never tell me again.

Steve: Yeah. So I got the message.

Leo: I have to say this is an improvement because in the past Microsoft's made it difficult. Remember, that's why you had Never10, because they put these pop-ups up, and you'd never get rid of them. And yes, "Learn more" is the default. "Remind me later" is also a big button. But at least they put over here "Don't remind me again." Go away. And that will, according to Mary Jo, that will make it go away.

Steve: Yeah. And I did find it interesting, when I was doing the digging around Microsoft site for the detailed information on the continuing availability of an essentially or effectively free Windows 10 upgrade, I did find it interesting that, if you sort of followed the normal path, you ended up at Microsoft trying to sell you hardware for Windows 10.

Leo: Yeah, yeah. They want you to buy a new PC. That's, you know, that's a sop to their OEMs, yeah.

Steve: Yeah.

Leo: And it's also an acknowledgment that most people never upgrade Windows. They just buy a new computer. That's how most people get Windows in the first place; right?

Steve: And, well, and actually, if you have Windows 7 on your PC, because everything has only shipped with Windows 10 now for so long, it is either you somehow deliberately arranged to struggle not to get Windows 10, or your machine is old; you know? Or your machine is old.

Leo: Right.

Steve: Because anything you would have bought in the last few years, you don't have a choice.

Leo: It's more than five years old, anyway, yeah.

Steve: Yeah. So it's probably got a smaller, slower drive. It doesn't have as much RAM as Windows 10 would like to take over and squat on. So, yeah. Might be actually time to get a new machine. And I'm sure our listeners know how to choose those.

There was a bit of news that Forbes' Thomas Brewster covered that I just want to kind of remind our listeners that this was still going on. And that is that Google's once-secret Sensorvault technology was back in the news this past week after some reporting by Thomas Brewster, who covers cybercrime, privacy, security, and surveillance for Forbes. As we know, last year - well, actually we don't know. We know about Sensorvault. But the news is that last year and this year substantial damage was done to Milwaukee, Wisconsin property by some arsonists. To help locate the crime's perpetrators, the Bureau of Alcohol, Tobacco, and Firearms - actually and Explosives, the ATF, I guess they decided not to add an "E" to the end, so it's not ATFE.

Anyway, they demanded that Google supply records of users, Android users' devices, which were located in the respective locations surrounding the times that the arsons took place. And as Brewster reports, although federal agents had used, as we know, the Sensorvault technology before, apparently they got way more results than they were expecting. Two search warrants requested any consumer information in Google's possession covering, as the reporting had it, 29,387 square meters, and this is the so-called geofencing range, during a course of nine hours for four separate incidents.

And as we know, if Android users or even iOS users running some Google apps have not turned off the defaulted on location history option, their physical whereabouts is being continually tracked and archived within Google's giant database called Sensorvault. And when we first covered this we learned that, as I think I recall, that data went back 14 years? I mean, so it was just - it was from the beginning of Google time, like all of the location information that they had been able to accrue, they were. And there's never really been a clear, as far as I know, a clear indication of why this is going on.

In this case, Google found just shy of 1,500 qualifying device identifiers. So I guess this was a high-traffic area. It was 1,494 qualifying device identifiers which were found in Sensorvault and were sent to the ATF, pursuant to the subpoena, for the ATF to comb through. So anyway, I wanted to, as I said, briefly revisit the topic with our listeners, just to remind everyone that unless they have explicitly turned it off, it's on. And we talked about this back at the time. Google did confirm that, if location history is turned off, then they will wipe it from the Sensorvault vault so that you're able to do a firm hard opt-out.

But at the time of this original reporting, I think it was in The New York Times, Congress stirred and sent Google a letter asking a bunch of questions about this. And I never saw any reporting of Google's official reply. Maybe that didn't make the news, or there were

other issues happening at the time. But what we learn about this kind of data collection is that, at the moment, if a company has something that they're collecting for some purposes, these secrets are difficult to keep. And if they get out, and the company is issued a subpoena, they can be induced to turn these things over. So a privacy respecting company, because it's really impossible, apparently, to keep these secrets, will not be keeping data that they may not want to be forced to turn over.

And unfortunately at this point Google has this massive archive, and for purposes that have never been made clear. I guess it's their right to collect it if the license agreement for the devices states that's being done in the fine print. But it certainly doesn't make people happy, and it sort of demonstrates that Google is not really respecting our privacy if it's up to us to opt out in order to be removed from this kind of database. So anyway, I just wanted to sort of remind our listeners. It's been some time since this came up. And it's, as far as we know, still going on.

Leo: I'm more upset with law enforcement because I feel like courts should say you can't issue a subpoena for everybody within 20,000 square meters of an event. That's way too broad. I mean, the police say, well, look, we will look at videos of a bank robbery. Lots of innocent people in the bank are included in those videos. But this is a much broader sweep of - it's a fishing expedition. And I just feel like that's...

Steve: Yeah. Well, and as we talked about at the time, there have been false arrests as a consequence of this.

Leo: Right, right.

Steve: Because, again, you're not actually operating from direct evidence. You're working from inferential evidence. And so you're needing to say, well, we know it's not Sarah Jane and Suzy Q because they were somewhere else, or we know them, or who knows what. But in the case that we covered at the time, they whittled it down to some guy and put him in jail for a week. And he's still not happy. He's still threatening to sue.

Leo: The story I saw on this was they have two suspects as a result of this out of the 1,500 that they rounded up.

Steve: Ah.

Leo: But they haven't filed charges against those two suspects. So I think it's the similar case where, yeah, you get a lot of information about innocent people, and it doesn't necessarily narrow it down.

Steve: No.

Leo: And bad guys, don't carry your cell phone with you next time. You know? I mean, this is something easily avoided by bad guys.

Steve: Yes, yes. And in fact you can use that data on your behalf. You can leave your cell phone at home. And if you are brought in as a suspect, you say, "No, I was at home, go ask Google Sensorvault."

Leo: There you go. There you go. And they don't know you didn't have it, yeah. I have it turned on. I like it. It's nice to know my - and you would use it maybe like when you went to Europe. You would then have a track of all the places you visited. It's kind of cool to have. If you take pictures...

Steve: Oh, we used it like crazy. In fact, even when I was up two weeks ago or a couple weeks ago, with you, I didn't mention it, but the night before we hung out for dinner with one of my old high school buddies who's up in the wine country, begins with an "H," I forgot the name of it.

Leo: Healdsburg.

Steve: Healdsburg, yes, of course. And we couldn't have found him. He was out at the end of some long windy road through the grapevines. And it's like, okay. But the map knew right where he was. So thank you very much.

Leo: Well, and a nice thing is Apple of late, and Android, too, will tell you these apps are looking at your location. You want to only do that when the app is running. And I think more and more people are aware of this. What you forget is that Android does it all the time in the background, that Google's always keeping track. So, yeah, clear your location history and turn it off unless you have a use for it. I use it, so I don't mind. And I'm not setting fires in Milwaukee, either.

Steve: So we got a new feature last week in Android's Messenger app. It's just been enhanced to offer its users what they call "Verified SMS." That's with a capital "V," so I think that's the official name. And at this point - okay. So it's kind of a cool feature, if a company wants to conduct business over SMS, which pretty much everyone now knows is just as much spam as it is real.

So Google said, okay, we're going to offer SMS verification. And at this point the list of supporting companies, because it's something that companies have to sign up for with Google, is rather short: 1-800-Flowers, Banco Bradesco, Kayak, Payback, and SoFi. That's it. With any luck, the service will grow in time. But their explanation was very light on details. But there was just enough for me to sort of read through the watered-down explanation and figure out what must be going on. So Google has effectively implemented an entirely separate authentication layer separate from SMS messages, which are just SMS messages. So they go through the regular SMS system. They're limited in size. And they're non-encrypted.

What Google wanted to do was to arrange to authenticate them, to add a separate authentication layer. So to do that, both ends of the SMS message connection also need to have data connections, that is, Internet connections back to Google for the exchange of metadata. Companies must, as I noted, register with Google and securely establish their identities. In Google's coverage of this, that's all they said. We don't know how that's done. Probably the way certificate authorities verify people. Maybe, you know, it's never been very robust. It's call you at the number that D&B has you registered as or something. But somehow the point is Google is convinced that you are, you the

company, are who you are claiming to be. At which time the company generates a public/private key pair and provides their public key to Google for Google's redistribution of that company's public key.

Then, also, each instance of Android's Messenger also generates a similar public/private key pair and provides their public key to Google. Now, that might be done ahead of time. That might be done on the fly. Either way would work. And again, Google was very scant on facts here. So when a company wants to send a "Verified by Google" SMS message, they obtain the recipient's public key from Google. Okay, so I guess I answered my question. That does presume that each Android device has preregistered, the Android Messenger has preregistered itself and its public key with Google so that the company wanting to send a verified message can get their public key.

So the company takes their private key and their recipient's public key and uses some key agreement protocol such as Diffie-Hellman Key Agreement (DHKA) which will allow them to synthesize what will end up being a shared secret key. This key is used as the key for an HMAC through which they run the SMS message. So basically they create a Message Authentication Code, a MAC of the message using the shared key which they derive from their private key and the recipient's public key. They send that HMAC back to Google and send the SMS message directly to the Android user over the standard SMS channel.

When the Android user receives the SMS claiming to be from the company, they have the company's apparent phone number, that is, the originator of the message. So Messenger asks Google for the private key matching that phone number. After Google returns that to the Messenger app, just as happened on the sending end, the Android client uses its private key and the obtained public key with the same key agreement, such as Diffie-Hellman. This, thanks to the magic of Diffie-Hellman Key Agreement, will yield the same key that the sender obtained.

So the recipient uses that to key the HMAC, runs the received SMS message through that, sends the message's HMAC back to Google. And if the two MACs match, the one from the sender and the one that's been received back from the recipient, Google then sends back a flag that turns on verified sender notification, along with a bunch of other information about the company to allow you to do a little bit of digging into who this verified sender is. So that's what they've done. Basically they added an entirely separate authentication mechanism to SMS.

And as I was thinking about this, it sure seemed like a lot of work to go through. But it does allow senders to send SMS messages to any Android recipients in a way that authenticates them as the sending company, and also absolutely prevents any tampering with the unencrypted SMS message while it's en route. So no man in the middle can get it. You can't spoof being the authenticated company. You can't change the message in any way, or the MACs won't match. And I did notice that U.S. law enforcement would have no complaint about this, since the system never encrypts the communications. It's only providing robust authentication of the sender and preventing any tampering.

So in Google's coverage of this, they wrote: "As part of this feature, Google attempts to verify all messages that appear to be sent by a business that is registered with Verified SMS. If the authenticity codes don't match, and Google can't verify the message, the Messages app displays: 'Message could not be verified.' Because verification requires a data connection, if you have a weak data connection, the Messages app may display 'Verifying sender...,'" presumably like waiting to get enough data or a robust enough connection to actually perform the verification.

And Google said: "If you have no data connection, the Messages app displays 'Waiting for connection to verify sender.'" And they said: "Until the sender of a message has been

verified, Google doesn't recommend replying with sensitive info or opening links that you aren't sure you trust." So anyway, as I said, I sort of wanted to cover it because I thought it was interesting from a cryptographic standpoint. They're creating something new that we've never had before. We certainly have secure encrypted authenticated messaging out of the SMS band. This doesn't encrypt SMS. It just adds, using metadata, it adds an authentication layer to it, which is another new feature for the Internet and for Google's messaging Android users. Which I thought was sort of interesting.

So I originally had titled this "The Other Shoe Dropped," but I changed it to "Another Shoe Dropped" because I think we still have several more shoes to go. While we were busy recording last week's Security Now! podcast, representatives from Apple and Facebook were testifying to a Senate Judiciary Committee hearing, attempting yet again to explain the value of encryption that has not been deliberately weakened. In return, those on the committee informed Apple and Facebook, in increasingly less uncertain terms, that they had better put backdoors into their end-to-end encryption, or laws would be passed forcing tech companies to do so. It's getting just more and more bear, or should I say "Barr."

The chairman of the Senate Judiciary Committee, our dear Senator Lindsey Graham said, and I quote: "You're going to find a way to do this, or we're going to do this for you. We are not" - this is still Lindsey talking. "We're not going to live in a world" - and you know what's coming - "where a bunch of child abusers have a safe haven to practice their craft." That's a craft? All right. "Period," he says. "End of discussion." And as anyone who's been following politics may know, Lindsey has been seeming a bit drunk on power lately. But that's beside the point.

So this is the latest shot fired in the ongoing war over encryption. The most recent salvos have been launched following the privacy manifesto that Zuckerberg published last March. As we've been noting here, Zuck framed the company's new stance as a major strategy shift that involves developing a highly secure private communications platform based on Facebook's Messenger, Instagram, and WhatsApp services. Facebook's stated plan is to leave the three chat services as standalone apps, but to merge their technical infrastructure so that users of different apps can talk cross-app to each other more easily.

And in digging into this a little bit more, I learned that that merging would include adding the WhatsApp Signal-based end-to-end encryption into Messenger and Instagram to make them compatible and similarly attack proof. At the moment, Facebook's Messenger supports end-to-end encryption in the so-called "secure connections" mode, which is a mode that's off by default and must be enabled individually for every chat. And Instagram has no end-to-end encryption on its chats at all. So all of this stands to change once the three are merged into a single Signal-derived messaging triumvirate.

This past October, as we mentioned a couple months ago, the three governments - the U.K., the U.S., and Australia - explicitly warned Facebook that they had better end or at least pause that plan. And in an open letter that we talked about at the time, calling on Facebook to back off of its so-called "encrypting everything" plan, U.S. Attorney General William Barr threatened them, essentially.

So Monday of last week, Facebook released an open letter responding to Bill Barr's letter. In that letter, the WhatsApp and Messenger heads, Will Cathcart and Stan Chudnovsky, said that any backdoor access into Facebook's products created for law enforcement would weaken security and let in bad actors who would exploit the access. They said: "That's why Facebook has no intention of complying with Barr's request that the company make its products more accessible."

They said in their note: "The 'backdoor' access you are demanding for law enforcement would be a gift to criminals, hackers, and repressive regimes, creating a way for them to enter our systems and leaving every person on our platforms more vulnerable to real-life harm. People's private messages would be less secure, and the real winners would be anyone seeking to take advantage of that weakened security. That is not something we are prepared to do." And in his opening statement on the day that followed, last Tuesday, Lindsey Graham told Apple and Facebook representatives who he was facing in this committee meeting, "The fact that people cannot hack into my phone..."

Leo: What a jerk.

Steve: He says he appreciates. He says: "I appreciate the fact that people cannot hack into my phone." But he said: "But encrypted devices and messaging create a" - and here it is again - "safe haven for criminals and child exploitation."

Leo: B.S. Oh, my god.

Steve: I know. So I'm trying to not go on too long here and skip this. In Facebook's letter, Cathcart and Chudnovsky pointed out that cybersecurity experts have repeatedly shown that weakening any part of an encrypted system means that it's weakened "for everyone, everywhere." They said it's impossible to create a backdoor just for law enforcement that others wouldn't try to open. Oh, and they said they're not alone in that belief. More than 100 organizations, including the Center for Democracy and Technology and Privacy International, responded to Barr's letter to share their views on why creating backdoors jeopardizes everyone's safety. Facebook's letter also quoted Bruce Schneier from comments he made earlier this year, saying: "You have to make a choice. Either everyone gets to spy, or no one gets to spy. You can't have 'We get to spy; you don't.'" He said: "That's not the way the tech works."

And I'll say again for the record, because I know that my stance on this confuses some people, I don't want backdoors either. And we have a real problem with jargon because "backdoor" is a heavily weighted word that everyone would agree is not good. But too many lawmakers I'm seeing are taking, I mean, even Dianne Feinstein, who's a California Democrat, they're taking Lindsey Graham's and Bill Barr's position. So if we're going to be forced to provide law enforcement with the equivalent of 21st-century lawfully warranted wiretaps, I would like those taps to be secure. Which is to say, I would like the legislation to be carefully written and not written stupidly with a broad brush.

As we know, Apple already provides secure group chat with iMessage. So it's utterly obviously possible for an additional silent listener to be added to Apple's existing iMessage chats. The technology is already there. It already exists. And it's secure. I'm not saying I want that to happen. But saying that it's not possible is not correct. Anywhere you have multiparty chat technology with the system and its keys managed by the provider, as they all are, it's obviously possible to add an additional listening party in a secure fashion under lawful warrant. Companies don't want to do that. I get that. They would rather be able to take the stance that for their customers' sake, that you have absolute privacy.

But in the U.S., the right to privacy is not an absolute. When a court has been convinced that a lawful suppression of an individual's or company's privacy serves the greater public good, privacy can be lawfully breached. The math of cryptography empowers absolute privacy in a way that the U.S. constitutional framers could have never foreseen and did not intend. I mean, it's not a protection that we have under the Constitution.

So anyway, of course our dear Attorney General was unable to resist marching out the kiddie porn during a Wall Street Journal event last Tuesday following these hearings. He granted that, yes, there are benefits to encryption, such as to secure communications with a bank was the example Bill Barr used. He says: "A financial institution that will and can give investigators what they need when served with a warrant. But he said that the growth of consumer apps with, as he put it, warrant-repellent end-to-end encryption like WhatsApp and Signal have aided terrorist organizations, drug cartels, child molesting rings, and kiddie porn type rings."

So anyway, unfortunately, whereas the U.S. Congress currently and otherwise seems to be more divided than it's been in quite some time over political policy matters, in this they are much more united. And so I just think it's - I think 2020 is the probably the year that we're going to see some laws created, and I just hope they're good ones because U.S. operating companies are subject to the laws of the U.S., and other countries are not any more happy about all of this than we are, than our law enforcement is. So interesting times.

Apple's iOS v13.3 last week added support for the first time for hardware key dongle authentication to Safari. With last week's update to v13.3, Safari on our iDevices obtained access to FIDO2-compatible authentication hardware such as Yubico's YubiKeys or Google's Titan, where they have the proper interface hardware. All three hardware modes can be used: NFC, USB, and Lightning.

So after the update to v13.3, users who have proper hardware can authenticate by using, for example, the YubiKey 5 NFC or Security Key NFC by tapping on the top of an iPhone from the iPhone 7 on. You can also do physical authentication, for example, with the YubiKey 5Ci by plugging it into the Lightning port of an iPhone or iPad. So that's a cool addition for those who are looking to get that kind of hardware dongle-based protection from their iDevices.

And last Tuesday turned out to be an important Tuesday to patch because it foreclosed upon an elevation of privilege vulnerability in Windows that was seeing widespread and active exploitation in the wild. It's now been patched in Windows, and it is a vulnerability that had been patched earlier in Chrome because it used a problem in Chrome, coupled with a problem in Windows. It needed them both in order to get exploited. Last week's December Patch Tuesday fixed a total of 36 vulnerabilities. Seven were critical, 27 were important, and one was moderate. And that left one that was low in severity.

The important one to patch turned out to be the one rated as one of those that was important. It was another flaw, and we've had a bunch of those this year, in the Win32k module - which, once again, also again because a lot of them have been this, enabled privilege escalation. When used with Chrome, it facilitated an escape from Chrome's sandbox, which is never a good thing to have happen. Google had addressed their side of the flaw in Chrome 78.0.3904.87, which was pushed out as an emergency update last month after Kaspersky disclosed it to them and to Apple. However, hackers were still targeting that flaw for any Chrome browsers that had not been updated.

And that's interesting. I don't know why they wouldn't be, although I have seen Chrome be a little lazy with updating as iOS often is for me. So I guess there was still a window of opportunity, and the benefit from achieving this exploit was great enough, and Chrome is now the most used browser on the Internet, so it was still being attacked. Anyway, it was a use-after-free exploit that was chained together with the now-patched exploit in Win32k component of Windows OS, which was handling objects in memory. And it's been patched.

Okay, everybody. If you have propeller cap beanies...

Leo: Beanies on.

Steve: Get them out and spin up your prop.

Leo: Uh-oh.

Steve: We know that an RSA private key is just a very large random prime number. It's hidden inside its matching RSA public key by multiplying it with another very large prime. And the private key is able to remain hidden because none of this planet's best math magicians have ever been able to figure out any means for breaking those two primes apart once they've been multiplied. But of course, if either of those primes did not have sufficient entropy, if either of them could be guessed, the hidden private key inside would be discoverable. It turns out that's kind of obvious.

But there's a more powerful weakness. It's been known for a while, but there hasn't been a survey done until recently. Last Saturday, during the first IEEE Conference on Trust, Privacy, and Security in Intelligent Systems and Applications, which was held down here in Southern California, in Los Angeles, a team of researchers from Keyfactor - interesting name - presented their findings into the current security posture of digital certificates on the Internet. Their paper carried the chilling title: "Factoring RSA Keys in the IoT Era." Okay. "Factoring RSA Keys in the IoT Era."

Leo: Mm-hmm.

Steve: So I'll share more about the details in a second. The super short version of their findings is that the certificates being generated by relatively entropy-starved IoT devices turn out to be lacking. The devices lack super high-quality random number generators, and they are revealing weekly randomized primes, which leads to a failure of the fundamental guarantee offered by RSA public key crypto, which asserts you can't factor this.

Leo: You can't factor this.

Steve: You can't factor this. Exactly. As a result of the widespread deployment of IoT devices today, these Keyfactor researchers claim that one in every 172 RSA certificates in active use today is vulnerable to attack.

Leo: One in 20, that's a lot.

Steve: No, one in 172.

Leo: Ooh, wow.

Steve: But even that, one in 172. Okay. So here's the abstract from their paper: "RSA keys are at risk of compromise when using improper random number generation." Okay, that's not a surprise to anybody.

Leo: Yeah.

Steve: "Many weak keys can be effectively discovered and subsequently compromised by finding reused prime factors in a large dataset." And I'll get into the details of that in a second. That's where our propeller beanies come in.

In this paper they said: "We collect and analyze 75 million RSA certificates from the Internet, and find that one in 172 certificates have keys that share a factor with another. In contrast, only five of 100 million certificates found in a sample from Certificate Transparency logs are compromised by the same technique."

And I should pause here just to note that - so what they're saying is that, when they looked at all certificates on the Internet, one in 172 were found to have a prime factor that collided with another certificate somewhere on the Internet. But if they looked at 100 million certificates found in the Certificate Transparency logs, only five of 100 million certificates had a collision. The point here is that Certificate Transparency logs are published by Certificate Authorities, so their primes are being generated, not by IoT devices, but by high-quality random number generators. So those are of higher quality is the point.

So they said: "The discrepancy in rates of compromise is overwhelmingly due to IoT devices exposed to the Internet, which may be subject to design constraints and limited entropy. The widespread susceptibility of these IoT devices poses a potential risk to the public due to their presence in sensitive settings." Yeah, like all the IoT crap that we have in our houses these days. They said: "We conclude that device manufacturers must ensure their devices have access to sufficient entropy and adhere to best practices in cryptography to protect consumers."

Okay. So what exactly is going on? What I learned, and this is really interesting, which is why our listeners are going to find it interesting, too, I believe, it turns out that there's sort of a weakness in our cherished and beloved RSA public key system. The result of this weakness - and Leo, you're going to love this from a math standpoint, too. The result of this weakness is that the entire system, when taken at Internet scale, is exquisitely sensitive to the quality of every single prime in use, and that in turn makes RSA somewhat vulnerable and brittle.

So here's how these guys describe the attack. And I've added a few words here and there to clarify from their - I've added a little editorial to clarify things. So they said: "RSA is used in the process of encrypting data to send across a network. The server transmits its RSA public key to the client as a part of an SSL or TLS handshake." We know all that. "Part of the RSA public key contains the modulus "n = p * q," where p and q are two randomly chosen primes of similar size. Primes p and q are kept secret, as knowing these values allows the private key to be calculated." Right? So p and q are primes, and they are multiplied to create this modulus n. And the whole point is you can't factor that.

They said: "Ensuring that p and q are selected with sufficient randomness" - that is, that are sufficiently random primes - "is a crucial component of keeping the public key secure." Now, again, by that they mean the public key is public, but it carries the private key. So when they say "Keeping the public key secure," they mean actually keeping the embedded private key which is in the public key secure. So they said: "Factoring a large modulus n to obtain p and q is not feasible under normal circumstances. However, if keys are generated with poor randomness, then it becomes a concern that two public keys anywhere on the Internet may share a prime factor once enough keys are generated."

So here's the counterintuitive loophole bit of RSA. If two RSA moduli, for example n1 and n2, in two different public keys, so if two of those where the p and q separately obtained primes, if they share precisely one prime factor p, then computing the Greatest Common Divisor (GCD) of those two public keys n1 and n2 will reveal the value of p. So again, you have two certificates containing public keys on the Internet. If by chance they share a random prime, then it is possible, it turns out, relatively easily to compute their Greatest Common Divisor. And their Greatest Common Divisor will be the prime that they share.

Then they said: "The GCD computation is significantly easier than straightforward factoring" - which of course is, as we know, effectively impossible - "and can easily be performed in practice. The other factors of n1 and n2 can then be trivially found by the simple calculation of n1 over p" - that would return q for that certificate - "and n2 over p" - that would return q for the second certificate. And they said: "Respectively, fully compromising both keys. This GCD computation can be scaled to analyze all pairs of keys in sub-quadratic time," meaning that it doesn't explode with the number of keys, sub-quadratic time in the number of keys.

They said: "Selecting the prime factors of appropriate size and uniform randomness should prevent two moduli from ever sharing a factor in practice. However, if there is a flaw in the random number generation when choosing primes, a collision is likely with a sufficiently large dataset. Attackers can use this knowledge to collect a large number of RSA public keys and then look for Greatest Common Divisors between their moduli to search for factors shared by any pair. And when found, both keys are rendered insecure. The private key component hidden in the public keys is completely revealed."

So what this means, there's a great many of the public keys circulating around the Internet are gathered into a massive dataset. This is no longer difficult today. It was. It was impossible when RSA was invented. Today we've got cloud everything: massive storage, massive computation. So you gather the huge dataset of public keys that are themselves not vulnerable. You then compute the Greatest Common Divisor against each pair of keys. And that's, of course, if the number of keys is n, then there are n times n-1 pairs of keys. So it's a very big number. But these days we deal with very big data all the time.

And in fact, these guys, I think they used an AWS, some AWS resources, and they did that. They took 75 million public keys that they had gathered, and they ran this GCD operation which has been now made very efficient against every combination of two keys in the dataset. And that's where they found that surprisingly high collision rate. Among high-quality keys generated with good entropy, it was only five in 100 million. But in the keys actually in use on the Internet, thanks to all the IoT devices we now have, that collision rate rose to the high level of one out of every 172 pairs.

So suddenly that makes attacking the keys being produced by IoT devices essentially practical. You'd have to hope that a particular device that had synthesized its key was in the dataset, but this turns this into a potentially practical attack because these IoT devices are just not generating high-entropy primes for the certificates that they're making for themselves. So I just thought that was incredibly cool. And of course it further enforces our use of the term "IoT" standing for Installation of Trojans.

Leo: So it's because IoT devices don't have much processor power; right?

Steve: Yeah. Although it's also because I don't think this problem was sufficiently well appreciated.

Leo: Anticipated, yeah.

Steve: Yes, anticipated. So, for example, nothing would prevent a camera that you're installing from establishing a secure link to the mothership and asking it for...

Leo: Give me a key, yeah.

Steve: Yeah, exactly. Give me a certificate. But they don't do that now. They go, eh, we got enough entropy here. We'll make our own, and we'll just assert our own certificate. They shouldn't do that. They should go somewhere and get a good one. And probably, if this research gets enough air and becomes enough well known, then all of our Alexas and our Echoes and our everythings, all of these higher end devices could certainly, I mean, they're able to create secure connections back to the mothership. So they ought to just ask for a private key and start using it, or ask for a certificate, rather, a certificate and a private key. They would need both.

Okay. Some quick bits here. Friday the 13th came true, unfortunately, for New Orleans, which was hit by a ransomware attack. I know. The extent of the attack was not known at the time of its announcement which - get this, Leo - was announced over the loudspeaker system in City Hall.

Leo: Turn off your computer. Turn...

Steve: Step away from the keyboard.

Leo: They even said to unplug it.

Steve: It did, yes. It told all government workers to immediately turn off their computers and unplug them. City websites were down. A spokesman for the mayor said the attack started sometime after 11:00 in the morning on Friday morning. New Orleans activated their Emergency Operations Center and contacted officials from the Louisiana State Police, the FBI, the state National Guard, and the Secret Service for assistance, according to a tweet from the city's Department of Homeland Security and Emergency Preparedness.

So now we're living in a world where you might receive emergency orders to immediately turn off and unplug your computer over the public address system. Amazing.

Leo: Did we ever find out really, though, what had happened? Because they said we didn't lose any data. We weren't compromised. It felt like there was more to this story. In other words, there was no encryption, as far as I know.

Steve: Oh, so I haven't seen any follow-up.

Leo: Yeah, there has been a follow-up.

Steve: Maybe it was just a cyber attack, and they said "oh crap" and shut everything down to be careful.

Leo: What it sounded like is that they were being bombarded with spear phishing attacks.

Steve: Oh, my lord.

Leo: And rather than take the chance that somebody would open one of those emails, they just said everybody shut your computer off now.

Steve: Right.

Leo: And then they went all around, and they looked to see if they could find anything, and they couldn't.

Steve: That's actually kind of smart.

Leo: They said, based on what we were told by federal law enforcement...

Steve: This is what we should do.

Leo: This is what we should do, yeah.

Steve: Wow.

Leo: So as far as I can tell, they lost nothing, and nothing was encrypted. But it was just shut down for that period of time because...

Steve: Well, and it's a nice holiday message for all of the...

Leo: Yes, every city in the U.S.

Steve: Yeah, and even for their employees. I mean, you're going to start taking it seriously if this happens, and then somehow there's like a meeting of like, okay, we did this because all kinds of email was coming in, and we couldn't take the risk that any of you boneheads would say, oh, what did my Aunt Mary send me?

Leo: Exactly. They had just had training in what not to do, how not to get phished. So it might well be that they avoided - this was actually the right way to handle it.

Steve: Yeah, yeah. Another little quick bit. Mozilla - I thought this was interesting - will be requiring all Firefox add-on developers to sign into their accounts using two-factor authentication next year.

Leo: Good.

Steve: Yeah. Their goal is to help prevent supply chain attacks where malefactors compromise an add-on authors authentication in order to inject bogus malware into an otherwise popular and believed to be safe add-on. So I think that's all for the best. Also, the primary reason I would have considered obtaining Windows 7 extended security updates - if I had been eligible, which I'm not, being a corporation, or not being a big enterprise - would have been for the continuing updates to Microsoft's built-in Security Essentials AV.

But interestingly, as it turns out, everyone is going to be cut off starting January 15th, even corporations who opt to purchase the extended security updates. Microsoft is not going to be providing them. Which I just don't understand that. I guess, I mean, they have to do the research for Windows 10. But maybe it's additional incentive to move people off of Windows 7? Because, again, it's nice to have Defender or Security Essentials, which is what I've got running in my Win7 machine, protecting you, you know, keeping an eye on things.

This seems nutso to me because it will drive corporate users who have decided they want to keep Windows 7 to adopt third-party AV at the enterprise level. It's going to be a windfall for AV publishers. But I verified Microsoft intends to say, nope, sorry, we're not going to offer any security updates, even for people, that is, any of our AV Security Essentials updates, even for people, like enterprise customers, who do opt to continue having support for updates. Crazy. I don't get it.

With the release of Chrome 79, the option to force the browser, Chrome, to continue displaying what Google has formally stated they consider to be trivial - that is their word, "trivial" - the leading www appearing in URLs such as www.grc.com has been removed. The www is gone for good.

Leo: Hallelujah.

Steve: And is not coming back in Chrome. Okay. I mean, I do think I made a mistake years ago. There was a big discussion in the newsgroups, in GRC's newsgroups, about whether I should standardize on www or not. The problem was I allowed either to be used, and the way you came in is the way you stayed. That is, I wasn't pushing anyone over one way or the other. The problem was that Google was then duplicating all the links. And so I wasn't getting the benefit of the traffic aggregation under one URL. They were being spread out. Many were under www.grc.com, and others were under grc.com.

And so I thought, okay, I've got to make a decision here. Do I want to redirect everybody who does www.grc.com over to grc.com? Or people who come in as grc.com over to www.grc.com? I needed SSL certs either way because in order to do a redirect you need to first have a connection. So it wasn't a matter of saving money on TLS certs. It was coalescing all of Google's links into a single agreement. And I made the wrong decision. I said, well, you know, www is what web browsers are for. That's the World Wide Web.

Leo: Right, right.

Steve: So that's what we should do. And I'll take some pleasure in the fact that Amazon is the same way. It's www.amazon.com, not amazon.com.

Leo: Right.

Steve: But lots of other people have said, huh? That's dumb.

Leo: In the early days, you had to distinguish web traffic from other kinds of traffic, and servers, web servers from other kinds of servers. So that preceding www told you this is a web server you're contacting. But you could be...

Steve: Do you really have to?

Leo: You don't anymore. No, no, no, it was a convention. It was a convention. I don't think it was required.

Steve: That was before my time.

Leo: Yeah.

Steve: Because we used the port numbers to perform that.

Leo: Yeah. Since you open up the conversation, you know. But by convention.

Steve: Like port 80 is nonsecure, and 443 is TLS.

Leo: Right, yeah. I think it was more so for humans. So but that convention, even that convention is useless at this point. I mean, who - I kind of don't like browsers, though, that take the full URL out, as Safari does.

Steve: I agree.

Leo: I want to see where I am.

Steve: And the problem, really, Leo, is magnified by the fact that w, that's a long - in English, w. It's like, www.

Leo: It's nine syllables for no reason.

Steve: Is anyone still listening?

Leo: Yeah. No, that's - way back in Tech TV days, this is how long ago, 1998, I insisted that we have the convention that, when we published web URLs, we didn't do and we didn't say w. They were still saying https://. So I said we don't say that, and we don't say dub dub dub. It's assumed. The problem is, if you haven't configured your DNS properly, on some sites, if you don't enter the dub dub dub, you don't get the full site. So but that's just a configuration mistake, I think. It's not...

Steve: And I think the other really interesting thing that's going to happen is when - it'll be interesting to see when browsers start assuming https.

Leo: Right.

Steve: They're still not doing that.

Leo: Right.

Steve: And so my front gate bounces people not only from grc.com, now I wish it didn't, but over to www dot are you still awake dot grc.com.

Leo: Https://.

Steve: And, yes, also moves them from http over to https in the process. So, agh. Anyway, but as you said, I mean, when Google first did this, they took a lot of flack. And so they first backed off. Then they gave us a means of overriding that for us old sticklers. And now in Chrome 97 the override is gone. It's like, okay, just suck it up. Get used to it, that www is history. And I kind of wonder maybe if I ought to just change my redirect and train Google over to grc.com because this ship seems to have sailed, or in my case sunk.

Also, speaking of Chrome 79, as planned, Google reenabled with Chrome 79 their browser's new code integrity protection feature. But not as planned, the unrecoverable "Aw snap" browser crashes immediately resumed as before.

Leo: Oh.

Steve: The culprits are many. But they include CylancePROTECT; even updated versions of Symantec Endpoint Protection, again; Webroot security solution; and others. So it is possible, if you are affected by this, your Chrome just won't work. It's possible to force the feature off when necessary by launching Chrome 79 with the switch. It's --disable-features=RendererCodeIntegrity on the command line.

I don't know what Google's going to do about this. I think at this point they're probably just going to tell people, well, in fact I did see some of their what-to-do, and it's like, tough. Fix your AV. Because what's happening is bogus AVs are reaching in and trying to

muck around with Chrome. And by Chrome turning this on, they're preventing anybody who doesn't have something signed by Microsoft from loading stuff into Chrome or Google or Microsoft, any third parties, from sticking their fingers into Chrome. Which is, I think, correct. So unfortunately, companies should not do that.

A little bit of clarification from our topic last week, the "VPN-geddon Denied." Somebody posted in the - well, I know who somebody, his name is Grant Taylor - over in the Security Now! newsgroup. In response to my discussion, he said - and he is a heavy-duty Linux guy. He said: "The VPN in any form is a path for the traffic to the victim to take." He said: "It happens to be an encrypted path. But it's not the only path. The same traffic can be sent unencrypted" - or other traffic, some traffic, any traffic - "can be send unencrypted to the LAN interface's MAC address destined to the VPN interface's IP address."

He said: "By default, many operating systems will accept the traffic on the wrong interface and hand it to the TCP/IP stack, which will hand it off to applications. The traffic does not need to come into the system through the encrypted VPN." And he's right. I didn't make that as clear as I should have.

I replied to Grant: Right. The virtual interface is a full interface inasmuch as it can receive data from other parties with local access to the network. I should have made that more clear since, as we know, that's the only way the attacker, who lacks the VPN's working encryption key, could possibly have injected, as I was talking about last week, SYNs or SYN-ACKs with test sequence numbers. They would need to come in unencrypted and hit the interface for it to respond. Otherwise there's no way to synthesize test SYNs.

And now, Leo, my big announcement.

Leo: Oh, I'm ready. I'm listening.

Steve: My change of life announcement.

Leo: Are you getting married?

Steve: No.

Leo: Oh. You're too old for that.

Steve: It's bigger than that.

Leo: It's bigger than that?

Steve: Bigger than that. Posted to the grc.sqrl newsgroup. The subject line, this was posted yesterday, Monday, December 16th: "Turning SQRL over to you guys to tend."

Leo: Ah, nice.

Steve: It was message number - get this - 23,606.

Leo: Wow.

Steve: In the SQRL newsgroup. That's how much traffic there was over the last six years.

Leo: Wow.

Steve: And the message reads: "Everyone. That was a momentous subject line to write, but I believe it's finally time to do just that. SpinRite is now the more urgent need.

"As you saw over this past weekend, I made an intense push to wrap things up with SQRL, and I did. The published SQRL docs reflect the local Word doc copies I maintain, and nearly 41,000 copies of the 'SQRL Explained' PDF have been downloaded. As we know, anyone who reads that will deeply understand SQRL, just like the Google cloud security guys did. The sqrl.grc.com server is also updated with the latest functional code, so it's back up to spec.

"I have a list of tweaks for GRC's SQRL client for Windows, but thanks to everyone's deep pre-release testing, the release number one client doesn't have anything wrong that needs fixing right away. Though some UI bits can be improved, I think that's probably always going to be true, and I do not want to delay my switch to SpinRite any longer.

"One thing I absolutely know because I've been watching carefully, and I've seen it for myself for some time now, is that I'm leaving SQRL in the hands of an extremely competent and capable group of developers who understand it every bit as well as I do. And GitHub has exploded with a wonderful array of SQRL-related clients, servers, and middleware. SQRL no longer needs me. SpinRite does.

"Over the next week or two I'll be spending some time at Level 3 maintaining the Windows servers that have been neglected for far too long. And I'll deploy a new Unix server to replace the aging hardware that has been hosting these newsgroups and our DNS from the start. Then I'll switch over to our spinrite.dev newsgroup here and return to generating test releases of new low-level SpinRite code as we develop the technology that v6.1 will need, and work to bring SpinRite fully back to life." So that day has come.

Leo: I think SQRL is done, basically; right? That's what you're saying.

Steve: It's been done for quite a while.

Leo: It's out of your hands.

Steve: Yes, it is. And speaking of which, after that posting, there was another posting that appeared in the GRC newsgroup from a Jose C. Gomez. His subject was "OAuth 2 Provider for SQRL." And he wrote: "Hi, all. Over the past couple of weeks I've been working on a functioning OAuth 2 provider that works exclusively with SQRL. This should, in my opinion, allow millions of sites, if they choose to, to adopt SQRL without having to change much on the backend. I am finally in a pre-alpha release stage and wanted to

share it with everyone here and get some input and thoughts. Following the SQRL motto, I've made it so you can remain pretty anonymous and still use the service. And of course there are really no secrets to keep."

He said: "It currently implements the basic Authorization Code grant flow and works fairly well. I'm planning on releasing it in beta sometime this week to let whomever wants to try it, play with it." He says: "But I run a Discourse forum like Leo, so I've made sure it will work with Discourse out of the box so the community at TWiT should be able to start using it, if Leo chooses to, pretty easily."

Leo: Thank you, Jose. I will. Yeah, good. That's great.

Steve: He says: "Anyway, here's a quick demo of it in my Discourse instance." He said: "Again, this is still in alpha/pre-alpha. So if you go poking around, things may blow up, LOL. But feel free to." He says - oh. He says: "It uses the Ask facility" - and people who know SQRL may remember that Ask is the one thing I built in which isn't about authentication, but it allows the server the SQRL client is authenticating to, to send you an out-of-browser, that is to say, out-of-band question, which is more secure than anything that could be done in the browser. And I didn't know why it would be necessary, but it was the sort of thing where I thought, well, you know, if it's not there, it'll never be there. And if it is there from the beginning, maybe somebody will have a use for it.

Well, it turns out he's using it. So Jose said: "It uses the Ask facility to act as the Permissions Granting Screen of OAuth." He said: "I thought it was a pretty neat way of putting the entire permissions structure in SQRL. We also have the ability, if we want to" - and note that normally what you get with OAuth is the site you go to, you need to interact with the site in order for it to ask you if you want to grant the site you are coming from permissions under its authentication. The point is you don't have to do that. SQRL solves that problem also, thanks to the Ask facility. So it's all kept within SQRL, making actually the whole process much more transparent.

Anyway, he said: "I have to give a big thanks to TechLiam and Jeff Arthur, who have been my sounding board over in Slack" - okay, we also have a Slack channel - "while I slugged through the protocols and fought with the specs. Also, a zillion thanks to Paul F., who let me use some of his tools like SQRLView and his command line SQRLClient for troubleshooting." Like I've been saying, I mean, there's been an explosion of this stuff. So there's just like we're rapidly developing a rich ecosystem of SQRL stuff. And then he wrote: "Seriously, SQRLView is an amazing piece of software, and it should be shouted from the rooftops for anyone writing or dealing with SQRL. Liam's .NET Core middleware is also a great piece of open source engineering, and it keeps getting better. Cheers, guys, and thanks again. I look forward to some feedback."

And then in the show notes I have a - he posted a link to a GIF which shows - it just shows a few quick pictures of his SQRL authentication in process. And he has sqrloauth.com is the site, S-Q-R-L-O-A-U-T-H dot com. So anyway, that served as like a perfect punctuation on my announcement that I am switching to SpinRite because SQRL is done, and it is in good hands now. So a huge thanks to everybody who has been interested, to all the developers who have jumped in and helped to bring this alive. And we know that it is coming to the attention of some big-time players. There's a lot going on behind the scenes that are at the whisper level at this point. But we've given it every possible chance of success.

Leo: I presume somewhere he has the secret keys and all that stuff for what I need to do.

Steve: Yeah. And in fact I asked him if he wanted to have a forum over on sqrl.grc.com, and he immediately jumped on it. So I did that this morning. So if you go to sqrl.grc.com, you'll find that he's got his own forum, of which he's a moderator. So he'll be managing that, and he'll be able to post links and keys and announcements and so forth. So Leo, you could probably aim your webmaster or your web guy over there.

Leo: Oh, I'm the web guy.

Steve: Oh, cool, cool.

Leo: But I'll have - yeah, Jose, if you want to email leo@leoville.com because I'll need an account, obviously, on his OAuth server to do this. But our Discourse already has the OAuth 2 plugin. I just need a custom server. And looks like I need a few - I have a client ID. I need a client secret authorization URL, token URL, that kind of thing.

Steve: Cool.

Leo: Callback ID, callback path, all of that stuff. But he can help me out with - I don't know exactly what his implementation involves. I'd need an account on his server, I think. But Jose, I'm listening. And I will do it the minute I get the information. We'll set it up on my Discourse. That'll be awesome. Well, our users would love it. I mean, they've been wanting that for a while. Good. All right. Let's talk about Plunderbuss. Whatever it's called.

Steve: Okay. So Intel must just be so tired of having their once-believed solid chips just hacked and hewed and...

Leo: Is AMD subject to this, too? I mean, they use speculative execution, but I never hear that it is.

Steve: I don't know why researchers don't go after AMD as much as Intel. I guess market share or accessibility.

Leo: The early stuff, you know, the Spectre and the - they were to some degree. My sense is the latest stuff, no.

Steve: Well, get a load of this one. Here's what their abstract reads: "Dynamic frequency and voltage scaling features have been introduced to manage ever-growing heat and power consumption in modern processors. Design restrictions ensure frequency and voltage are adjusted as a pair, based on the current load, because for each frequency there is only a certain voltage range where the processor can operate correctly. For this

purpose, many processors, including the widespread Intel Core series, expose privileged software interfaces to dynamically regulate processor frequency and operating voltage."

Okay. So essentially what that means is that die sizes are getting bigger. We know how much power these systems are using in order to create the processing power they have. Whereas, you know, we have people with water-cooled GPUs and things. The point is that there is power being burned. So you can reduce power consumption if you switch the voltage between zero and one through a lower change because it actually - essentially there's distributed capacitance throughout all of this silicon. And so in order to move something up to a different voltage, to switch it from a zero to a one, you need to essentially fill the capacitance which is present with some number of electrons. And they're continually making those capacitances smaller, but they still exist. So if the voltage for a one is less, then you need fewer electrons to be squirted in there in order to bring it to a one because of its lower capacity for holding electrons.

So one of the things that you do is you lower the voltage as much as you can so that less current has to be pushed and pulled in order to go through those voltage excursions. And then of course you also run the chip as slow as possible when it's not needing to be working hard so that you are pushing the current through those excursions less often. So you control those two things. The problem is, if you want to run faster, you also need to provide more voltage because voltage is pressure. And so that's why it's necessary for speed and voltage to be moved in concert. And so the Intel chips do that. Anybody who's messed with tweaking voltages and speed in their BIOS knows they have, like, way too much power and way too much control over that stuff. You can really run your hardware right at the - literally at the leaking edge.

Okay. So they said: "In this paper, we demonstrate that these privileged interfaces can be" - get this. These privileged interfaces, that is, the previous paragraph ended with "Intel Core series expose privileged software interfaces to dynamically regulate processor frequency and operating voltage. In this paper, we demonstrate that these privileged interfaces can be reliably exploited to undermine the system's security. We present the Plundervolt attack" - "volt" now we know as in voltage, and "plunder" as in, you know, give us your women - "in which a privileged software adversary abuses an undocumented Intel Core voltage scaling interface to corrupt the integrity of Intel SGX enclave computations."

They said: "Plundervolt carefully controls the processor's supply voltage during an enclave computation, inducing predictable faults within the processor package. Consequently, even Intel SGX's memory encryption and authentication technology cannot protect against Plundervolt. In multiple case studies, we show how the induced faults in enclave computations can be leveraged in real-world attacks to recover keys from cryptographic algorithms, including the AES-NI" - that's the New Instructions - "instruction set extension or to induce memory safety vulnerabilities into bug-free enclave code. We finally discuss why mitigating Plundervolt is not trivial, requiring trusted computing base recovery through microcode updates or hardware changes."

So, okay. So this is not a processor working the way Intel intended it, which was of course all of the last two years of speculative execution exploit. This is deliberately making the processor go out of spec, making it fail by sneaking the voltage down until it fails, and then turning that failure into an exploit, believe it or not. It's like, okay. Is anything possible? Apparently.

Okay. So what's cool is a couple things. First of all, this affects Intel 6th, 7th, 8th, 9th, and 10th Generation Core Processors, also the Xeon E3 v5 and v6, and the Xeon E-2100 and E-2200 Families. So that's everything from Skylake on is affected by this. In the show notes I have a picture of the thing that they reverse-engineered, this undocumented undervolting model-specific register. These are registers where - this is

where things like the number of cycles or the number of branches not taken versus the number of branches taken, that's where all of that extra microcode-level wisdom accrues.

So a register in there, it's a 64-bit register, and it's got three bits that they call the "plane index" that's 40, 41, and 42. And they allow you to select which plane, as their term is, like which area of the processor core you want to tweak the voltage. Index 1 is the CPU core itself. Index 2 is the GPU, which you can separately control. Index 2 is the cache in the core. Index 3 is the cache out of the core. And 4 is the analog I/O subsystem, if any. And then lower down in there is an 11-bit signed voltage offset. So it's 10 bits plus sign, which are in units of 1/1024 of a volt. So that's essentially a millivolt. So basically this allows you to push up or down by one volt in essentially one, just slightly less than one-millivolt steps the supply voltage individually of any of those cores.

So the way they find, the way they got into this, their original proof of concept is a cool little bit of code that I also have here in the show notes. What I'm showing here in the show notes is a little bit of C. They have a 64-bit multiplier which is just 1122334455667788. And then they have a variable, which they define as the hex deadbeef, D-E-A-D-B-E-E-F in hex, times the multiplier. Okay, so that's going to create the initial value for var. Then they have, while var equals that same expression, the deadbeef hex times the multiplier, while that's true, then they have it recompute var in the same way. They set var equal to deadbeef, then var * = multiplier, which will multiply var by the multiplier.

So this is an infinite loop; right? Because you are testing var as the product of deadbeef times multiplier. Then you're reperforming the multiplication, setting var again to deadbeef times multiplier in a while loop controlled by that original test. It's going to run forever, never going to come out of that loop, unless the multiply fails. If the multiply produces the wrong result, we drop out of the loop. And then var is XORed with the proper value of deadbeef times multiplier, which turns var into which bit or bits were incorrect in the final in-loop multiplication.

So they said: "This code is placed into the secure enclave and clearly should never terminate. But our experiments revealed that undervolting the CPU just before switching into the enclave leads to a bit-flip in var" - I know, Leo, it's unbelievable - "typically in byte 3, counting from the least significant byte as byte 0. This causes the enclave program to terminate. The code outputs the XOR of the erroneous value with the expected value, to highlight only the faulty bit or bits. And they consistently observe that in this specific configuration the output is always 0x04 00 00 00." So it's the four bit, the hex four bit in the fourth byte from the bottom is the one that gets set when they sneak the voltage down. That's where the multiply fails. And as we have so often seen, what starts off as a benign but unexpected fault in a computer system can often eventually be developed into a full working exploit.

Leo: This is almost certainly Intel-specific. There's no way this is going to be the exploit. This is Intel's mistake.

Steve: But what's bizarre is it's across their whole processor family.

Leo: Yeah.

Steve: Probably because they reuse the same silicon for that. And they've successfully done that here. Their page goes on to great depth showing how they managed to

leverage their undervoltage tweaks into a complete breach of the sequestered processing, which is supposed to be hidden inside Intel's secure enclave.

The result of this is Intel's statement: "When SGX is enabled on a system, a privileged user may be able to mount an attack through the control of CPU voltage settings with the potential to impact the confidentiality and integrity of software assets. Intel has worked with system vendors to develop a microcode update that mitigates the issue by locking voltage to the default settings." So basically this was something Intel never imagined could be abused, or was hidden because it was undocumented, and nobody would ever find it.

Well, these guys did, and they figured out how to turn it into essentially an export of secret keys, of private keys from the secure enclave. And a lesson we've learned during the past two years, since the tip of the iceberg which was known as Spectre and Meltdown, is that choosing a PC vendor who is actively keeping their past product offerings current with a flow of BIOS updates containing the microcode patches now being produced by Intel on a relatively ongoing basis, is one more factor to consider when selecting your vendor.

I would say, you know, I would have never worried. I didn't worry 10 years ago. Well, three years ago, actually, about whether the vendor I bought my motherboards from was producing BIOS updates. It's like, yeah, the BIOS is the BIOS. Who cares? As long as it boots the OS, we're not using it once it gets booted. Well, it turns out BIOS updates are important because they are updating the microcode on the fly, as we know. And in so doing, they're fixing problems. So anyway, I just thought that was so cool, that sneaking the voltage down to the point where the processor starts to fail, and it fails in a way that's predictable, and that predictability allows that fault to be turned into an exploit. Wow.

Leo: It said "privileged user." What would that take, to be a privileged user? Who is the user?

Steve: Yes, that's a very good point. And that's one of the mitigating factors. You have to have access to those model-specific registers, and that requires kernel-level access. So ring 3 you cannot access those registers.

Leo: The user app can't do this.

Steve: Yeah.

Leo: Maybe an antivirus could, though, and they usually have ring 0.

Steve: An AV could. And of course things get down into the kernel, unfortunately, all the time.

Leo: Right.

Steve: Those Win32k elevation of privilege, those all get you down into ring 0. So it's not like that's impossible to get to. There was something I wrote years ago. It was that thing

that everyone was using to figure out if they had 64-bit capability. It was not intended for that, but it was incredibly popular. In fact, it still is. It's like people use it all the time. It's not coming to mind. But I had to write a separate DLL that ran in the kernel with kernel privileges in order to query some of the model-specific registers in order to determine what was going on. So again, somebody running at ring 3, you don't have access to those. You need to get down at ring 0 and have the required privilege because you could do great damage.

Leo: Yeah. Microsoft's been trying very hard to keep people out of ring 0. But the problem is it breaks stuff. And I think they tried in Windows, I want to say in Windows 7, didn't they say, okay, no more kernel extensions? No more kernel access? But then everybody complains, and they say, well, we have to have it.

Steve: Yup. And the big hassle, I've hit this recently, is now all drivers have to be signed. And in Windows 10 you have to go through a really annoying, I mean, basically you upload your driver to Microsoft because your driver has to carry their signature.

Leo: Well, but that's how you protect the kernel.

Steve: That's right. That's right.

Leo: Securable is the name of the program.

Steve: Oh, Securable. You're right.

Leo: Bill in Michigan knew it.

Steve: Yeah.

Leo: Steve has so many programs, so many children, he's forgotten many of their names. But you know what, you can now forget SQRL. It's been handed off.

Steve: It has a life of its own. It has a life of its own.

Leo: And soon as I hear from Jose I will do whatever it takes to...

Steve: Oh, that'd be very cool.

Leo: I will. I've wanted that for some time. And, yeah, that's one of the native plugins on Discourse's OAuth, too. So that's a big deal. I think having OAuth 2 capabilities, I guess we'll have to go through his server, though, for the time being.

Steve: Yeah, I was thinking about that, too. It's making me think that, well, I don't want the responsibility because if GRC was DDoSed...

Leo: Who runs them? LinkedIn runs them. There are a lot of places where OAuth 2 servers live. Anytime I use the single sign-on, click the Facebook or the Google link, that's OAuth 2.

Steve: You're passing through there, yes, yes. And you're bouncing through their service. So they need to be up, like always.

Leo: All the time, yeah.

Steve: So it would be good once Jose has this figured out to figure out where he could put it so that...

Leo: It's something Cloudflare should do.

Steve: Oh, I like that, yeah.

Leo: John. Oh, John. Be great if Cloudflare did it.

Steve: Yeah, yeah, yeah. That's exactly the right profile.

Leo: Yeah, yeah.

Steve: Because if they don't, they will by tomorrow.

Leo: They can.

Steve: They're amazing.

Leo: Isn't it? Yeah, let's keep that. Let's keep kissing them, and maybe they will. It'd be really nice. Steve is at GRC.com. Expect more good things there now that he's freed up to get to work and focus on other stuff. GRC is his website, the Gibson Research Corporation; but it's also where this show lives, 16 and 64Kb audio versions you can download from there. Also transcripts. Elaine Farris writes those out with great care, so they're really well done. A few days after the show comes out they'll be posted at GRC.com. And naturally, while you're there, you should absolutely get a copy of SpinRite, the world's best hard drive maintenance and recovery utility because you're guaranteed an upgrade if you buy it now. And Steve's working on that right now.

Steve: Soon to be new and improved. And it is my commitment, because I don't want to be rushed, and everybody knows nothing I've ever done is fast - although I did crank out Never10 in a couple days and a few other things I was able to do quickly. But product scale things take time. I'm going to be producing pre-release versions which anybody who owns SpinRite will be able to use their serial number or their transaction code in order to download.

Leo: Oh, nice. So keep that handy. That's great.

Steve: I'm going to be pushing one out shortly after I return to work.

Leo: Oh, man. Well, quick, go to GRC and buy SpinRite so you can get that. You still - you going to fire up the MASM and the Brief, and you're ready to go?

Steve: You know, I've given up on Brief. I did make the switch to Visual Studio. I didn't like it at all at first.

Leo: VS Code? Have you looked at VS Code? That's the lightweight editor that Microsoft makes. It's not Visual Studio.

Steve: Yeah, well, I have Visual Studio, although of course true to form I'm using Visual Studio 2008. So, you know.

Leo: Look at VS Code because that's a simple code editor. It's Electron, sorry. But it does do, it will do, I'm pretty sure it will do MASM. And, you know, there's other good text editors out there, I'm sure.

Steve: Well, I'm not looking for an editor. I really - I don't spend much time in a debugger. And in fact I'm debugging on DOS.

Leo: Of course.

Steve: And so I may be back using SoftICE, which was the really cool soft in-circuit emulator that I developed all of my DOS stuff on.

Leo: Wait a minute. You don't use an editor at all? What, do you just - what are you doing?

Steve: No, no, no. I need more than an editor. So Visual Studio...

Leo: Yeah, yeah, you need a debugger. I see what you're saying.

Steve: ...is an editor and a compiler and debugger. And that's the way I ended up, ever since I switched to Windows 7 in the middle of the SQRL project, I've had to, like, bite the bullet and leave Brief. So I'm not going back to Brief. I should mention, too, and I meant to earlier, I am probably going to switch to Windows 10 here early next year.

Leo: You buried the lead. That's a big story.

Steve: Well, the big problem was giving up 16-bit code. And that's why I had to give up Brief. But having done that, I've survived, I'm still here, I'm still able to develop stuff. So it's like, okay, moving to 10, I mean, I will literally strip the crap out of it, and I do mean crap, in order to tame it. But I think, you know, and I have a bunch of Windows 10 machines around now. And it's like, yeah, okay, there are some things about it that annoy me. But I've sort of made peace with it. So I think that's probably what I'm going to do is to make a snapshot, make copies, virtualize this one, and then so I can always fall back to it. And I've got to move all my license stuff over, which is annoying. But once that's done, it's done. I think it makes sense.

Leo: Look at VS Code. It allows you to run a debugger for most languages. Again, I'm not familiar with its ASM capabilities. But I bet it does. And compile and all of that stuff. The idea is - the problem with what you're - Visual Studio is really a .NET development tool. It's crazy heavy. And so I think VS Code might be more to your liking. It's just an editor with built-in - for instance, I can get a REPL with Lisp so I can run the code and debug it and all of that stuff. So I bet you you could do that with VS Code. There's probably other stuff out there, too, but VS Code's free, and I like it.

Steve: I'll take a look.

Leo: A lot of coders use it, yeah. Okay. GRC.com. Go get that stuff. On the Twitter he's @SGgrc. You can leave him messages, comments, suggestions. You can also get the show from our site, TWiT.tv/sn for Security Now!. And you can even subscribe in your favorite podcast application. In fact, that'd be the best thing to do. That way you'll get it automatically, the minute it's available. Normally we do the show on Tuesdays at 1:30 Pacific, 4:30 Eastern, 21:30 UTC. Not next week because Tuesday's Christmas Eve.

Steve: Yes, it is.

Leo: So we're moving the show, all the Tuesday shows to Monday. So next week, if you want to watch live, we will be on December 23rd, 1:30 Pacific, 4:30 Eastern, 21:30 UTC.

Steve: Yes.

Leo: The nice thing about the download...

Steve: With our special podcast, "A Decade of Hacking."

Leo: Oh, is that what you're going to do?

Steve: We're going to do the retrospective last 10 years.

Leo: Whew. Yeah, we did that TWiT Special. We just did the decade's big stories and gadgets. I can't wait. That'll be fun.

Steve: Yeah.

Leo: And then the following week it's a "best of" on December 31st, New Year's Eve, a "best of."

Steve: Perfect. So one week off for us.

Leo: Yep. Thank you, Steve.

Steve: Okay, buddy.

Leo: And I'll see you next time on Security Now!.