The Nansh0u Campaign

Description: This week we check in on the BlueKeep RDP vulnerability. We look at the planned shutdown of one of the, if not THE, most successful, if one can call it that, affiliate-based ransomware systems. We update you on the anti-robocalling problem and then look at the recent announcements by the Russian and Chinese militaries about their plans to move away from the Microsoft Windows OS. We also look at Apple's announcement yesterday of their forthcoming "Sign in with Apple" service, touch on the state of SQRL, and then share a bit of fun feedback from a listener. We finish by examining the interesting details behind a significant old-school persistent campaign, the Nansh0u campaign, apparently sourced from China, which has successfully compromised many tens of thousands of servers exposed to the Internet.

High quality (64 kbps) mp3 audio file URL: http://media.GRC.com/sn/SN-717.mp3
Quarter size (16 kbps) mp3 audio file URL: http://media.GRC.com/sn/sn-717-lq.mp3

SHOW TEASE: It's time for Security Now!. Steve Gibson is here. We're going to talk about a nation-state exploit that's now being used by script kiddies all over the world. More about the RDP exploit. It's not going to end the Internet, but it might be close. And we'll also talk about what Apple's doing to simplify and privatize single sign-on. It's all coming up next on Security Now!.

Leo Laporte: This is Security Now! with Steve Gibson, Episode 717, recorded Tuesday, June 4th, 2019: The Nansh0u Campaign.

It's time for Security Now!, the show where we talk about security, privacy, how computers work and all that jazz with this guy right here, "James Tiberius" Gibson. No, wait a minute.

Steve Gibson: I'm saying, wait, James? What?

Leo: I'm confusing you with Captain Kirk. It's Steve Gibson of the Good Ship GRC.com. Hi, Steve.

Steve: Ah, the Lollipop, yes.

Leo: Yes. Hey, somebody sent me - and, shoot, I threw out his name, but he sent me a bunch of PopSockets with a familiar logo on it. You probably know who this is.

Steve: I know him. And in fact I wore one of his shirts. I keep meaning to tell our listeners that there are SQRL T-shirts and hoodies on Amazon now.

Leo: Merch, baby.

Steve: Merch, yes.

Leo: And apparently PopSockets, as well.

Steve: Now, you know, when he was talking about a PopSocket, that's not a term I'd ever heard of, very much like when...

Leo: Yeah, because you're not hip with the kids.

Steve: Well, it's when you first mentioned "podcast," and I said, "A what? A what cast?"

Leo: So I'll show you. You adhere this to the back of your phone.

Steve: How is that a good idea?

Leo: Well, it's a bad idea if you do as I do, do wireless charging. But a lot of people...

Steve: Yeah, that would be a problem.

Leo: Yeah. But a lot of people - I could do it real quickly here. A lot of people like this because it gives - you see it all the time with the kids because then - okay. That's popped in. And then when you pop it out - I shouldn't do it right after I adhere it, but I'll do it anyway. Now it becomes a little phone holder.

Steve: Ah.

Leo: And pop it out. And see, like that?

Steve: And so, yeah, yeah. I mean, no, I mean, I have to explain, I should say that, yes...

Leo: You're an old guy. That's the problem.

Steve: Well, no. Since I've realized that there is going to be a SQRL logo on something, I should figure out what it was.

Leo: What it is.

Steve: And so I did solve the mystery of the PopSockets.

Leo: Oh, you knew. Okay.

Steve: Yes. In no way related to Pop Rocks. That was something else that was a passing fad.

Leo: So thank you. I wish I knew your name. I do know it was on the package.

Steve: It's Mike.

Leo: Something, that was it, yeah. Thank you, Mike. And I now own like a dozen PopSockets with the SQRL logo on them. Mike knows I have a lot of phones, I guess. Anyway, that's pretty cool. And all that merch is on Amazon.

Steve: It is very cool. And but what it is - yes, it's on Amazon. If you put in "SQRL T-shirt," I think that will take you to - and in fact I was wearing one at the L.A. OWASP meeting a couple weeks ago.

Leo: Nice. When you did your presentation.

Steve: Yeah, and in fact the shirts are very nice.

Leo: Very nice.

Steve: Very sheer. I like a sheer T-shirt because then you can breathe.

Leo: There's an image, yes.

Steve: One of these days we're actually going to get down to the podcast. Although I realize, Leo, this is how many of your podcasts go.

Leo: We are somewhat discursive here at TWiT. Is that what you're saying?

Steve: We have Security Now! Episode 717 with a name I cannot pronounce. What would that be, the Nansh0u Campaign?

Leo: Nansh0u, yeah. What is it?

Steve: Nansh0u. It's Chinese. It was a string that was found in some malware. And we're going to talk about that because what's interesting is it is very sophisticated, state-level malware that unfortunately has many signs of amateur origin, suggesting that - this is also something we could have predicted, that inevitably state-level quality tools are now beginning to find themselves in the hands of amateurs. And so that suggests that there's going to be more rather than less trouble going forward. So, yes. Three digits is probably not enough for the numbering of these podcasts.

We're going to check back in on the BlueKeep RDP vulnerability. We look at the planned shutdown of one of the, if not THE most, well, I guess you'd call it "successful," if you can call it that, affiliate-based ransomware systems announced that this is the final month for which the affiliates will be able to earn revenue from their victims. And we update on the anti-robocalling problem, take a look at the recent announcements made by the Russian and Chinese militaries about their plans, believe it or not, to move away from Microsoft Windows. It's like, wait. You guys are still using Windows? What?

We also look at Apple's announcement yesterday of their forthcoming "Sign in with Apple" service, and touch on the state of SQRL. Then we'll share a bit of fun feedback, and then dig into this Nansh0u campaign, which many indications indicate it was sourced from China. It has successfully compromised more than 50,000 MS-SQL Servers which are exposed to the Internet. And, I mean, that should be an oxymoron or an oxy - what would you call an oxy sentence? Anyway, I guess there is no such thing. But, I mean, you should not have the phrase "MS-SQL Servers exposed to the Internet." I mean, I guess there are some use cases. But, boy, it always seems like a mistake.

And anyway, so I think another fun podcast and interesting podcast for our listeners. And we're also going to touch on, follow-up on last week's dire predictions of the end of the Internet. I hope our listeners know that we were having fun with that, and that I didn't really think it was going to happen. But I'm going to elaborate on why I don't think that this will be used as a worm. I think the days of the worm have passed, well, at least for this class of vulnerability. The point is you don't need a worm to exploit this. And so there's a place for worms. This isn't the place.

Leo: Actually, I just saw that they've analyzed the malware that shut down the city of Baltimore for three weeks, and EternalBlue was not used to spread it throughout the network. So I guess what you're saying is you don't need special illicit worm code to spread ransomware all over.

Steve: Right.

Leo: And now back to Steve, coming to you at the speed of light.

Steve: Saw on the Hacker News, I just liked the image that they had created. They had the Windows logo with the word "BlueKeep," and then underneath it, "Powering 1 Million Computers."

Leo: That's the estimate of the number of computers that are vulnerable to BlueKeep? A million?

Steve: A million, yes.

Leo: Wow, wow.

Steve: Remember that last week, and this has - actually, we're going to talk about this right now because Microsoft posted last Thursday, posted a blog posting sort of with an update reminder about this problem. Now, I don't think I've been explicit in saying that, unfortunately, reminding people isn't going to be effective because anybody who is in the loop, anybody who is conscious understands about this problem. And if they're aware that they have a problem, they will have fixed it by now. So I don't get what everyone running around warning people about this does. I mean, maybe marginally it's useful.

But all indications are that we have - it was Robert Graham at Errata Security who did a full scan of the Internet. Then from all of the 3389, which is the default RDP port, he then analyzed the protocol, figured out which of those were true Remote Desktop Protocol services that were exposed, and then probed them in a presumably benign way to determine which of those were vulnerable to this problem. And from that he obtained the IP addresses of 950,000 presently vulnerable Windows machines that actually have RDP exposed, that have not been updated, that require no authentication in order to get on and get in.

So again, I mean, that was already two weeks after the dire warnings and the updates were made available, and nobody seems to be taking action, again, because I don't think anybody is listening. Whoever is behind those million machines, they're asleep at the switch.

So last Thursday Microsoft titled their blog post "A Reminder to Update Your Systems to Prevent a Worm." And in a minute I'm going to explain why I don't think that's the problem that we're going to have. But they said: "On May 14, Microsoft released fixes for a critical Remote Code Execution vulnerability, CVE-2019-0708, in Remote Desktop Services - formerly known as Terminal Services - that affects some older versions of Windows. In our previous blog post on this topic we warned that the vulnerability is 'wormable,' and that future malware that exploits this vulnerability could propagate from vulnerable computer to vulnerable computer in a similar way as the WannaCry malware spread across the globe in 2017." And again, I'll explain why I don't think that's going to happen.

But they said: "Microsoft is confident that an exploit exists [yeah, duh] for this vulnerability; and, if recent reports are accurate, nearly" - this is Microsoft - "nearly one million computers connected directly to the Internet are still vulnerable to CVE-2019-0708. Many more within" - and this is the point you made last week, Leo. "Many more within corporate networks may also be vulnerable. It only takes one vulnerable computer connected to the Internet to provide a potential gateway into these corporate networks, where advanced malware could spread, infecting computers across the enterprise. This scenario could be even worse for those who have not kept their internal systems updated with the latest fixes, as any future malware may also attempt further exploitation of vulnerabilities that have already been fixed.

"It's been only two weeks since the fix was released, and there has been no sign" - this is Microsoft still - "no sign of a worm yet. This does not mean that we're out of the woods." Actually, I would argue we will never be out of the woods. They said: "If we look at the events leading up to the start of the WannaCry attacks, they serve to inform the risks of not applying fixes for this vulnerability in a timely manner. Our recommendation remains the same. We strongly advise that all affected systems should be updated as soon as possible." They said: "It is possible that we won't see this vulnerability incorporated into malware. But that's not the way to bet."

Then in their blog post they remind us about the timeline of the EternalBlue exploits. They said: "Almost two months passed between the release of fixes for the EternalBlue vulnerability and when ransomware attacks began. Despite having nearly 60 days to patch their systems, many customers had not. A significant number of these customers were infected by the ransomware." So anyway, I thought that was the most interesting part of this was a reminder that this doesn't - even though the problem exists instantly, it does take a while for tests to be run, for code to be written, for something to be done.

And as we know, we had fun last week with this, you know, the end of the Internet and asking our listeners to quickly download the podcast because they may not be able to get it in a couple days. But I think there's, in all seriousness, I think there's a very different way to think about this. To me, this doesn't feel like a pending worm.

One uses a worm for its multiplying effect when it's necessary to brute force attack the possibly weak but still present credentials, or an authentication weakness which is believed to exist in many machines which may be unknown. So a worm is therefore a good way to collectively pound on the Internet as a whole to break into unknown targets that are resisting that break-in. And the reason it's a big deal is that, that way, you can get newly exploited machines to chime in on the attack of other vulnerable machines which haven't yet fallen.

But that's not the scenario we have here. Here there's no need to pound on anything, and the vulnerable machines are not unknown and needing to be pounded upon to be found. As we discussed last week, simple IPv4 scanning, which is now widely available, the code is in Metasploit. The idea of scanning the whole IPv4 Internet space, that once upon a time was daunting when you had 4.3 billion IP addresses, now it's like, eh, yeah, I just, you know, I mean, Bob - Robert scanned the Internet over the weekend, and did it several times, and to collect his address of 950,000 currently known vulnerable IPs. He has them. He has a list. And before long, if not already, he won't be the only person to have such a list. So I don't really see that a worm makes any sense at all. Yes, it is...

Leo: But internally, like on a LAN. You know, once you get it on one machine, affecting the other machines on the LAN.

Steve: But even there, you know your subnet mask, so you know the exact IP range of the LAN. And so you can just scan, like instantly scan. From one location you can scan the entire extent of the LAN, looking for port 3389, and then [crosstalk].

Leo: So [crosstalk] get in on 3389, you don't really care.

Steve: Correct. And what the worms were traditionally doing is they would spread because each new instance would then start randomly looking for other machines and then pounding on them to get in. But there's no pounding necessary. The door is wide open. I mean, what will be interesting is to see whether a month or two from now the apparent number has dropped precipitously because that would tell us that residence has been taken up by something that has closed the door behind itself, which is the sane thing to do. Were I not wearing my bright shiny white hat here, a scan of the whole IPv4 space would have found the problem.

It's no mystery how to exploit it any longer. You install whatever it is you want to. And what really seems to be what people want to do these days is make money. And the way they make money is they participate in mining pools. So, I mean, what I seriously expect, if someone said to me, "What is the number one most likely scenario that you

expect to see for this nearly one million exposed machines," I would say get the IPs, work up an attack that installs a persistent cryptominer that participates in a mining pool, get it going, close the door behind yourself so that you're not competing with other people who are coming along later, and that's what you do.

You then, if you are the first in and the first to set up a miner and to close the vulnerability behind yourself, you now have potentially one million cryptomining machines actively participating in mining. And you don't know how long they're going to last. You're going to end up with many of those forever. Some may, because they've become bogged down by the use of their CPU, maybe then the fact that the machine's no longer working right will come to the owner's attention.

Again, the fact that all these dire warnings are being spread, that has no effect whatsoever. Whoever these belong to, these million instances, they're not listening. I mean, they're not getting auto updates, or this would have been solved. They're old machines that have just - they're, as we often say, in the closet somewhere. Well, in that closet that thing's going to be generating more heat before long because its CPU is going to be pinned by a bad guy who doesn't care at all about throttling the CPU to prevent its discovery.

What they want is, they want as many cycles shared in this mining pool as possible to generate revenue for as long as it lasts. That's what I think is going to happen. And it'll have no effect whatsoever on the Internet. In fact, it will have the effect of cleaning up a million potential vulnerabilities that could have been used for much more harm. But instead, the first person in is just going to use it make money. And so it would be really interesting if we see a few months from now a repeat 3389 scan looking for the same profile and, oh, look, they've all disappeared. I wonder where they went. Well, they got taken over, and someone closed the door behind them. I think that's the most likely scenario.

Okay. So I've mentioned this ransomware before. This is an interesting profile. And we don't know whether - it would be really nice to know how much money these guys made. They're claiming, in the 18 months since they launched this Ransomware as a Service, that their ransomware service took in a total of $2 billion, from which they received a commission because basically they set up affiliates. They set up franchises.

Okay. So what's interesting, the reason we're talking about it now is that they've announced this is the final month of this. So we noted in our Picture of the Week last week, we showed the timeline of the explosion in ransomware. And we stopped tracking it, I mean, whatever that image was, it was in my catalog of interesting security-related things to share on the podcast from 2017.

Leo: I just don't think you could fit 2018 or 2019.

Steve: Exactly. Exactly. So this strain of ransomware is called GandCrab, G-A-N-D capital C-R-A-B, GandCrab. We've touched on it briefly. Nothing really made it stand out until now, when this Ransomware as a Service has announced that they're going to shut down their online portal, which is where their affiliates or distributors or franchisees sign up and pay to get access to custom builds of this GandCrab malware, which then the affiliates distribute via whatever means they want - email spam, exploit kits, RDP maybe, or whatever means. And then when an infected victim pays the ransom demand, because the GandCrab originators still hold the keys to this ransomware that they're making available to their affiliates, then they earn a commission.

And you can sort of imagine the pitch: Yes, you, too, can form your own moneymaking, highly profitable ransomware franchise, encrypting the contents of unwitting strangers' hard drives and extorting many hundreds of dollars from each of them at no cost to yourself. For just a relatively small piece of the action, we'll manage all of the messy little details for you. You just go out there, find fresh new victims, and arrange to get them to run the ransom malware we'll provide to you at no additional charge.

We'll also keep it updated with all the latest cutting-edge AV-avoiding tricks and technologies. We'll even go so far as to purchase bright shiny new code-signing certificates from Comodo, still - against all reason - one of the most trusted names in certificate authorities. That way the malware you distribute won't raise any questions when your trusting victims press Go. What could be easier? Burdened by the weight of massive student loans? You don't need Bernie Sanders. Retire that college debt from the comfort of your own parents' basement and show them just what an enterprising little cuss you can be.

So, yes, you can imagine that they generated many affiliates who were...

Leo: That's good marketing, baby.

Steve: Who were actively redistributing their malware and apparently generating lots of money. So anyway, those were the days. As far as we know, this dream is coming to an end at the end of this month.

Leo: Aw.

Steve: Last Friday, on May 31st, as I had mentioned before, GandCrab's operators announced their intention to shut down their illicit operation. A number of news outlets picked up on this. I have a screenshot of the posting that they put up. And they wrote, and this is - they're not English speakers natively, so it's a little awkward. But they said: "All the good things come to an end. For the year of working with us, people have earned more than $2 billion. We have become a nominal name in the field of the underground in the direction of crypto fiber." I don't know what that means. "Earnings with us per week averaged $2.5 million. We personally earned more than $150 million per year."

Leo: What? No.

Steve: Well, you know, we don't know. They said: "We successfully cashed this money and legalized it in various spheres of white business, both in real life and on the Internet. We were glad to work with you. But as it is written above, all good things come to an end. We are leaving for a well-deserved retirement. We have proven that by doing evil deeds, retribution does not come."

Leo: Yet.

Steve: "We proved that, in a year, you can earn money for a lifetime. We have proved that it is possible to become number one, not in our own words, but in recognition of other people." So, and then now they say "In this regard," and they mean with regard to shutting things down, "stop the set of adverts. We ask the adverts to suspend the flows.

Within 20 days from this date we ask adverts to monetize their bots by any means," meaning, say, threaten their victims with final payment or else. And then they said: "Victims, if you buy, now." And they said: "Then your data no one will recover. Keys will be deleted." Then they signed off, saying: "That's all. The topic will be deleted in a month. Thank you for all the work."

Anyway, BleepingComputer, one of our favorite outlets for news, and especially that tracks ransomware, said that they had reached out and asked these guys to please publish their keys after they shut down. Others have, and that way there would be the ability for victims who would otherwise have all of their data lost to still be able to recover it. But anyway, we don't know. It's certain, you know, these particular guys, the GandCrab guys, were known for pulling stunts, and having a little bit more interaction with the security community than usual, and also of being jokesters. So all these numbers could be made up.

But it is the case that independently this particular ransomware is one of, if not the most pernicious and prevalent ransomwares in the - I hesitate to call it "the industry" or "the marketplace." But in the world. So they may have been raking in some money over the course of this. So, yeah. Goodbye and good riddance, but wow.

And it's interesting, too. They did something, given their numbers, that demonstrates they did something different than the Coinhive guy. Remember that one of the things that always surprised me was that the guy behind Coinhive was taking a large piece of the action. Even if these numbers are inflated, they suggest that these GandCrab guys were taking a much smaller piece of the action, which represents a kind of maturity in terms of, if you want to incentivize your affiliates, your entrepreneurs, then let them have most of the ransom and just take a relatively small piece for yourselves.

And if this is true, that strategy may have worked for them because, again, if they actually did generate over the course of 18 months, which this thing appeared on the scene in January of 2018. So here we are at the beginning of June 2019, 18 months later. And they said, yeah, we're going to retire now because we've - what do you call it? Laundered. We've laundered our illicit earnings by investing in legitimate businesses and online, and so we're going to go find a beach somewhere. Crazy.

So there is incremental forward progress on the robocalling front. We've talked previously about, I mean, I just love saying SHAKEN and STIR. Those are the two telecommunications protocols we touched on before which are in the process of being gradually deployed by all the major telecommunications carriers. The event which triggered our previous discussion was the first intercarrier test of this SHAKEN and STIR technology, which was at that time between AT&T and Comcast. Apparently there had been previous intracarrier testing, but that doesn't do anybody any good because the whole point is what SHAKEN and STIR do is they apply certificate-based technology to caller ID signing to strongly prevent caller ID spoofing, which is so rampant today that, I mean, it's not worth bothering or even believing the caller ID you see.

I know that, for myself, when the phone rings, I'll look at it. And it'll be my area code, and oftentimes it'll be my prefix, which is supposed to mean, oh, it's somebody in your neighborhood or a neighbor or something. I mean, they're just making that up. They know the number that they're dialing, so they're able to spoof the number they're dialing from to encourage you to answer. And it's a little bit mindboggling that there's, until now, well, in fact even now, absolutely no control over that. I know that Lorrie had taken to continually blocking all of the numbers that were coming in. But of course they never repeated, so the blocking was serving no purpose whatsoever.

So anyway, the reason this comes up again today is the U.S. Senate just passed, on a vote of 97 to 1, and I can't imagine who the one was who would say no, we don't want

this, a bill known as TRACED. Which of course, you know, somewhere, Leo, there's people who are good at coming up with names for things that have fun acronyms.

Leo: Yes. They all work for the House.

Steve: Exactly. So TRACED is Telephone Robocall Abuse Criminal Enforcement and Deterrence Act, T-R-A-C-E-D, which will now go to the House of Representatives, where it's expected to pass by just as large a margin as it did. Then it will go to our dear President's desk, who will, I imagine, sign it into law. When the bill becomes law, as it is expected to, it will then empower our FCC, the Federal Communications Commission, to impose significant fines up to - get this - $10,000 per call for illegal robocalls. The legislation would also increase the statute of limitations for bringing such cases, thereby giving the FCC regulators more time to track down the offenders. This law will also create an interagency taskforce to address the problem and pushes the major carriers in the U.S. - Verizon, AT&T, T-Mobile and so forth - to continue moving forward on the deployment of interprovider SHAKEN and STIR protocol in order to perform end-to-end authentication, which is what we need.

So all of this is looking good. This is the most recent bill. There have been 13 that preceded it, and this one is the toughest, I guess because everyone is just really getting fed up with this. And in fact this bill has the backing of all 50 state attorneys general, 35 of whom told the FCC last October that they were all pulling their hair out over the enormous problem which was beyond anything their own states' law enforcements could handle.

Earlier this year in February, Ajit Pai, our FCC Chairman, reiterated his call for a robust caller ID authentication system to be implemented this year, in 2019. And so we need the technology in order to create the ability, to enable the ability to enforce the law that we have. And so the good news is we are, even though this takes quite a long time to happen, we are finally beginning to move forward to this.

I skipped another little piece from the middle of last month because it didn't quite make the grade, but it fits in with this because on May 15th, a couple Wednesdays ago, the FCC announced a new measure that would grant mobile phone carriers the authority to block robocalls. The new rule would make it easier for mobile carriers - AT&T, Verizon, T-Mobile and so forth - to automatically register their customers for call-blocking technology, rather than, as it is now, requiring consumers to opt in on their own.

And this is actually my favorite: It would also allow customers the option to block calls coming from phone numbers not on their Contacts list. Which in my opinion would be a tremendously useful filter. If nothing else, maybe just automatically send them to voicemail or put them somewhere else. But the idea of blocking something not on your Contacts list would mean that you could actually pay attention to your phone when it rings because it's somebody you know, rather than making you look at the number and go, ugh, you know, just another unknown caller.

So anyway, Ajit Pai said two weeks ago: "Allowing call blocking by default could be a big benefit for consumers who are sick and tired of robocalls." Yeah, no kidding. So anyway, these things never happen quickly, but it looks like the problem has gotten so bad that we're finally getting some traction on it.

Leo: Steve, you're not looking for work, are you? Because if you are, it's a great place.

Steve: Oh, boy. I have so much to do.

Leo: Isn't it nice not to have to search for work anymore? I have to say.

Steve: I'm not looking for anything more to do.

Leo: When's the last time you actually went out to get a job? Never. You've been an entrepreneur from day one, practically; right?

Steve: Yeah. There were a couple, like in between, when I was calling myself a "consultant," unquote.

Leo: Yeah, oh, yeah.

Steve: And I did some consulting work. I wrote a Windows Manager for DOS that ran in text mode.

Leo: Nice.

Steve: There was another one. I can't remember now which one it was. There were some early ones that were multitasking on DOS. And so I did some fun things. I developed EKG equipment, long-term ambulatory EKG monitoring, back before we had digital. It was a very slow-moving tape, and it recorded, it FM-modulated EKG on...

Leo: Oh, my god. Would you have to carry that under your arm as you walked around?

Steve: They were called Holter monitors because a Dr. Holter figured out that sometimes you don't see arrhythmias occurring...

Leo: Unless you're in life.

Steve: Unless you're actually out.

Leo: In situ, yeah.

Steve: So you'd do it for 24 hours. So, yeah, I did a bunch of different kind of interesting things. And then of course the PC came along, and I looked at the Color Graphics Adapter, the CGA, that was just, like, flickering incredibly whenever it scrolled. And I thought, okay, there's got to be a way to fix this. And so I created my first product for the PC, which was FlickerFree.

Leo: Nice.

Steve: And of course before that was the Apple II, and I did the light pen.

Leo: So you always really have been an entrepreneur, honestly.

Steve: I've, yeah, I certainly have...

Leo: You're a self-starter, yeah.

Steve: ...the soul of an entrepreneur. And it's funny, too, because in this story we're going to talk about, we'll get to China in a second, but they're actually thinking that they're going to roll their own OS from scratch. And, you know, I don't think you can anymore. But anyway, let me start at the beginning. So Russia first. For their military systems, Russian authorities just recently announced that they are continuing to move toward implementing their plan to replace Windows with a locally developed operating system, and they're doing it in what I think is the sane way, which is based on a Linux distribution.

Leo: Yeah, of course.

Steve: Called Astra.

Leo: Yeah.

Steve: I mean, it's complete auditable. You have the source code. You can have your Russians rummage through it and look for any problems. And we also know that, I mean, we don't know - as I was thinking through this last night, putting this together, I thought, well, you know, we were worried about the elliptic curve dual whatever it was, DRBG random bit generator a few years ago because we didn't know the source of some of the magic numbers, and there was some concern that there may have been some NSA influence.

And in this post-Snowden era, of course, we now understand the strong interest that our intelligence services do have in influencing things. So, similarly, we don't know that there are not Linux developers with the NSA. In fact, we do know that there are some, that there's a government side to Linux for, like, SELinux for creating a more secure kernel and so forth. So, I mean, that's a little bit dicey. But again, the task of creating an OS truly from scratch, you know, unless you'd be happy with DOS, I just - I don't think that's on the table anymore.

Anyway, Wikipedia - I didn't know what Astra Linux was so I looked in Wikipedia. They said: "Astra Linux is a Russian Linux-based computer operating system developed to meet the needs of the Russian army, other armed forces, and intelligence agencies. It provides data protection up to the level of 'top secret.'" And as I'll discuss in a second, there's a level above that. But "...up to the level of 'top secret' in Russian classified information grade. It has been officially certified by the Russian Defense Ministry, the

Federal Service for Technical and Export Control" - that's the FSTEC that we'll get back to in a second - "and the Federal Security Service."

So what happened was last month the Russian Federal Service for Technical and Export Control, that's that FSTEC, granted Astra Linux the security clearance above top secret to "special importance." Which means the OS can now be used to handle Russian government information of the highest degree of secrecy. And before this...

Leo: And good news for modern PCs, comes on CDs. Excellent news. No more floppy booting. This is it. This is it. It's on a CD.

Steve: Yeah, a CD; right. Nice.

Leo: A CD. Maybe a DVD. Maybe they're - that's great. Or you can download.

Steve: CD was 700MB; right?

Leo: Yeah, yeah.

Steve: So I'm not sure if that would...

Leo: Is on CD now in a...

Steve: Download now, yes.

Leo: Download, please. That's funny.

Steve: So anyway, before this the Russian government - and again, this to me makes no sense. They had been using a special version of Windows which had been modified, checked, and approved for use by the FSB. So...

Leo: I'm not sure, yeah.

Steve: I don't know what that means. Maybe Microsoft had a deal with Russia to get Windows into Russia? I don't know. But anyway, now the Russian military will begin their move to Astra Linux now that the FSTEC has given it the grade above top secret that is special importance. So they're now, by law, able to use it. And since it got secret and top secret, Astra Linux has slowly made its way into other government agencies and is currently in use at the Russian National Center for Defense Control and other government and military agencies. A year and a half ago, at the start of 2018, the Russian Ministry of Defense announced their intentions to transfer military systems from Windows over to this Astra Linux, citing their fears, which in my mind are entirely reasonable...

Leo: You bet.

Steve: ...that Microsoft's closed source approach might hide Windows backdoors - well, okay, I'll talk about a monoculture in a second - Windows backdoors that could be abused by U.S. intelligence to spy on Russian government operations. And as our listeners know, I've long been shaking my head in amazement that Russia could still be using Windows for anything because that just boggles my mind. So anyway, for this past 18 months the company RusBITech has been pursuing the Russian government certification process to obtain this much-sought "special importance" classification for their Astra Linux, which did finally happen last April 17th.

Oh, and in addition to the FSTEC certification, Astra Linux also received Certificates of Conformity from the FSB - we know those guys, that's Russia's top intelligence agency - as well as the Ministry of Defense. So that opens the door for full adoption by all of Russia's top military and intelligence agencies. So they're moving to Linux, which makes sense.

What doesn't make sense to me is what's happening in China. The Chinese military also plans to replace Windows - and again I say, "Huh? Really? You're using Windows?" - due to fears of U.S. hacking. But in their case they have said they're moving to a custom OS, not Linux. Now, okay. It must be that they're not starting from scratch, that there already is some kind of working Chinese OS. I don't know. But according to reports, officials in Beijing have decided - now, what the reports say, they've decided to develop a custom Chinese OS to replace Windows on computers used by the Chinese military. The report did not come by way of official Chinese channels, but by way of a Canadian military magazine, Kanwa Asian Defence.

As the reporting goes, due to the Edward Snowden, Shadow Brokers, and Vault 7 leaks, Beijing officials are now quite well aware of the U.S.'s heavy arsenal of hacking tools - I mean, you know, we're all, we globally, the whole world is now well aware of that - which are able to target anything from, as we know, smart TVs to Linux servers, routers, common desktop operating systems like Windows and Mac. Since these revelations have shown that the U.S. can hack into almost anything, the reporting has characterized the Chinese government's plan as that of adopting a "security by obscurity" approach by running a custom OS that will make it harder for foreign threat actors, mainly the U.S., to spy on Chinese military operations.

I characterize the move differently, though. If you can do this, I mean, if it can be done, I think it makes a lot of sense; though, as I said, I think it's going to be a heavy lift to create a desktop-class OS from scratch. I don't see it as security through obscurity, but rather as a healthy move away from a monoculture OS environment. Monocultures are inherently fragile, and increased heterogeneity among OSes hugely reduces the threat from both accidental mishap and deliberate attack. With someone like SandboxEscaper spewing out privilege elevation exploits daily, lord only knows how many ways into Windows our own OS intelligence services have already discovered and are stockpiling.

And as I've said before, I'm utterly amazed that Windows has been allowed to exist inside rival foreign governments for as long as it has. It must be that any Windows machines being used in sensitive environments are well isolated and never experience encrypted real-time communications with the Internet. I mean, there's no way that you could put a Windows machine somewhere sensitive and let it have access to the Internet. You don't know what it's doing. Everything it's doing is encrypted, and it's just spewing, I mean, I don't know if you've ever looked at how chatty Windows 10 is now. It's nuts. And again, you have no idea what's going on.

So there's just no way that that's the way Windows systems are being used. Or probably these things are old XP systems that have not been updated forever, but they're in

sequestered networks somewhere, not subject to attack and not communicating the way modern Windows systems are.

Anyway, the task of developing the new OS and replacing Windows goes to the Internet Security Information Leadership Group, which was first reported by Epoch Times, citing the May issue of that Kanwa Asian Defence magazine. The magazine said this new group answers directly to the Central Committee of the Chinese Communist Party and is separate from the rest of the military and intelligence apparatus. So there is now a new Internet Security Information Leadership Group that is in charge of developing the Chinese OS.

It would be fun to be assigned the task of creating a new OS from scratch. But it would also be so tempting to borrow from the existing extensive open source code base. The problem is there is no way to do that without the possibility of inadvertently carrying over unsuspected vulnerabilities. At the same time, look at how long it took us, "us" the industry, to get something as relatively simple and straightforward as a TCP/IP protocol stack that wasn't an utter disaster of edge cases and security flaws. Leo, when we first started the podcast 14 years ago, I mean, we were still finding problems in production TCP stacks. And in fact I remember we talked about some instance where a new system was brought online.

Leo: Yeah, I think it was like Windows Vista. And they had rewritten the entire TCP stack from scratch.

Steve: Right.

Leo: Something like that; right?

Steve: And they brought forward some problems that had been resolved years before.

Leo: Yes, exactly.

Steve: They returned.

Leo: Yes. I think that was Vista. It was a version of Windows. I remember that.

Steve: Yeah. And so, I mean, there's certainly something to be said for code which has had the crap pounded out of it, I mean, which is where any - well, in fact, one of the versions of Windows didn't carry their own stack forward. I remember they took the BSD stack. That may be what we're remembering was that Vista just said, okay, we're just going to - we can't recreate this wheel or reinvent this wheel. We're just going to take the BSD stack. There was one version of Windows where we learned officially they were taking the open source stack from one of the BSDs, and they were going to start there because otherwise, I mean, because they recognized how easy it was to make these mistakes.

And that's the point. On one hand I can see the value of starting from scratch. I mean, if you absolutely had a blank slate, then no previous influence, no previous malicious influence, if it existed, would be carried over. But the problem is you have the mixed

blessing because none of the lessons, I mean, the hard fought, hard won lessons that you only get by putting these things out on the frontline and having the world's hackers pound on them, and then going, ooh, ouch, okay, and then fixing this; and ooh, ouch, and fixing another thing. And finally that process stops because there's nothing left to fix. You get there eventually. And then you hope that you can add something to it without breaking something because you don't want to mess with this.

So I don't know. I mean, I really see China sort of between a rock and a hard place on this because there's, you know, if Russia has a version of Linux which is descended from the Linux the rest of us have, then it has the benefit of the lessons, but you will never know that there hasn't been some as-yet-undiscovered influence that went into the code. The flipside is with, if you start from scratch, good luck. You can't write a browser renderer from scratch, and a JavaScript on-the-fly compiler with state-of-the-art performance. I mean, just think of how sprawling today's functional desktop has become. Maybe they don't want - maybe they're happy with email and directory listings, in which case, yeah, you could give them that in a couple months.

Leo: Or maybe they're going to actually not rewrite it from scratch, but they'd like you to think they're rewriting it from scratch. Might be a little security through obscurity. They'll probably just steal FreeBSD's code, dress it up.

Steve: Yup. Yup.

Leo: I mean, I don't blame them. Why should you tell the world what your kernel is?

Steve: Yes. And you can then imagine that there would be an effort on the part of the world's intelligence agencies...

Leo: Yeah. Now we know what they're using.

Steve: ...to get a copy of the Chinese OS. And then they'll go, oh, we know what version of Linux that was taken from.

Leo: Yeah, true. I can download Astra right now. I'm sure you could start hacking on it; right?

Steve: So one of the interesting announcements that everybody brought to my attention, and I know you were talking about it both at the time and, I'm sure, during your previous podcast, was Apple's announcement of Sign in with Apple.

Leo: Yes, which they're going to enforce. Jim Dalrymple, or maybe it was James Thomson, told us they're going to require it on the App Store.

Steve: Yup. I have that in our show notes. So during yesterday's, for those who don't know, during yesterday's Apple Worldwide Developer Conference 2019 kickoff, Apple announced their launch of a privacy-respecting Sign in with Apple OAuth redirection

service, similar to the very familiar Sign in with Google and Sign in with Facebook that is becoming increasingly ubiquitous.

And they did something clever, though. They beat all other OAuth providers to the punch by adding a feature to their forthcoming OAuth redirection service which I just sort of named as I was putting the show notes together Managed Random Email Address Provisioning. And I'll explain why I called it that in a second. It's a brilliant idea, and it's a reason for using Apple's OAuth service over others, even if one isn't that concerned about OAuth's tracking.

Okay, but let's back up a little bit. First, we know OAuth is a convenience, but it comes at the clear cost of explicit and high-reliability tracking. When our browsers are redirected through that chosen third party's authentication provider - Facebook, Google, whatever - that third party knows who we are since we have an account with them. And they know to which site we are signing in since they're negotiating our identity with that site on our behalf behind the scenes. And all of this uses first-party session cookies, so no kind of tracking blocking, third-party cookie blocking, fingerprint, none of that applies. None of that works. So it is the number one most reliable means for a third party to know who you are and where you are, where you're going, that exists today.

So it's no surprise that Facebook and Google have said, oh, yes, use our OAuth service. And we know that there's just no way that Facebook and Google, both having strong financial interests in compiling activity profiles about everyone, are not actively leveraging for their own purposes what is essentially very specific and clear "who you are and where you're going" information. So by comparison, traditional browser tracking is inherently soft and fuzzy and requires some degree of behavioral inference. Not so with OAuth.

So on the tracking privacy front, Apple's announcement was clearly brilliant from that standpoint, and a poke in the eye at all other OAuth trackers. The problem is that, for the experience to be seamless, our browsers need to be maintaining an active session state with the OAuth provider so that our browsers can bounce through them transparently. But currently I have no such relationship with Facebook. And although I'm currently an avid Apple device consumer, I don't currently have such a relationship with Apple because Apple is not otherwise a provider of online services for me. I recognize, I mean, I'm a Windows guy, not an Apple guy. So I'm not logged into my iCloud account or anything on my Windows system.

But I do have a relationship like that with Google. Every one of my various browsers maintains persistent sessions with Google because Google already offers many online services which I use. And while I dislike the idea of being tracked, it's not really something I worry about that much. As our listeners know, I hate the idea from a technology standpoint of it not being under my control if I were to care a lot about it. That really seems wrong. And I certainly understand that many people do care.

But remember, as part of the OAuth transaction, the site being signed into can and does request data from the site you're signing in with, such as your username and your email address. Since this happens behind the scenes, users may not even be aware of it. Or, if they are shown that this will be happening, they're typically not given the option to have that information provided or not.

Now, Apple actually does. I have a screenshot in the show notes that shows that happening. But what this means is, and what may mean a great deal to people who already strongly trust Apple, and in a moment we'll see why deep trust is required, is that, rather than providing the service that I'm being signed into, my real Apple email - actually, that would really annoy me because that email account is unique, and I only use it with Apple, so it is completely clean. Nobody else has it or knows it. I get no spam or

any nonsense there. So I really like the fact that, if I get email, it's absolutely from Apple.

So Apple will be offering the option to proactively mask their user's actual email by providing a randomly chosen email relay address. And on the screen of the WWDC conference, Craig put it up that the example was fc452bd5ea@privaterelay.appleid.com. So the idea would be, if you say "Sign in with Apple" to some third-party site, at your choice, Apple will make up an email address and provide it as yours to that third party. And I called this Managed Random Email Address Provisioning since also onstage yesterday Craig indicated that Apple users would have some means for curating and removing the relay addresses from any sites from which they no longer wish to receive mail.

But this does place Apple in the enviable position, should they ever decide to leverage it, and I'm not suggesting that they would, but they will be receiving and viewing, if they wished, and then forwarding all of the email being received. In that sense it's not unlike Google with Gmail, who as we know sees all of our email, runs their bots through it, and does whatever they do with it.

Anyway, in the show notes I have a picture that we have seen where it says "Use your Apple ID," and in this instance it's kim_kilgo@icloud.com, "to sign into Fretello and create your account with the information below." And then it shows Name, Kim Kilgo. And it looks like you could hit X and remove that. I don't know what happens if you do, if you fill in a blank, or if they're just not going to provide that to the third party.

Leo: I think you just choose one of those three.

Steve: Oh, you think it's not...

Leo: Oh, I see what you're saying because it's Name and X. Yeah, maybe you don't give them your name. No, you could get rid of that and choose something else.

Steve: And then you have radio buttons for either Share My actual Email, the iCloud.com account, or you can click a button and say Hide My Email. And then it says Forward To and then your iCloud account. So anyway, so what happens is Apple would make up an email address which would be - which they would record in a database associated with you so that, when subsequent email comes in there, they would relay it, they would forward it to your actual iCloud account. So that would mask your iCloud account email from the third parties and give you apparently administrative control over it.

There would be some interface. You log into iCloud with your browser to Apple, and then you would get a list of all of the sites who have a forwarding for you. And who knows, maybe you could either remove the forwarding - I guess, no, you wouldn't be able to do that. Well, yeah, you could cancel the forwarding, in which case their email would no longer go through to you. Otherwise you'd have to go to the site and then update your email address with something else.

So anyway, it's probably not a big deal that Apple is seeing your email pass by. Again, we trust them. They make a big point of being privacy enforcing. As I mentioned, Google already gets and scans the email. But anyway, it is something to keep in mind. For me, my Google email is already my designated junk email bucket since its anti-spam filtering is so good. And as I said, my Apple email account is absolutely pristine because I've never used it for anything else.

So for me, I think I'd rather continue using Sign in with Google and have that email address being disseminated - and, boy, I mean, it's a junk pile - and keep my account with Apple clean. But, Leo, as you mentioned, and as I did pick up, interestingly, one reason to think that this will become prevalent, at least from within apps over which Apple has control, like those for iOS, is that offering it will not be optional.

Down in the fine print at the end of Apple's new App Store Review Guidelines, Apple states: "Sign In with Apple will be available for beta testing this summer. It will be required as an option to be offered to users in apps that support third-party sign-in when it is commercially available later this year." So that is to say, any app that offers Sign in with Facebook, Sign in with Google, will be required to also offer Sign in with Apple. So as soon as that starts to happen, it's going to appear. So any apps which offer OAuth third-party login will have Sign in With Apple also.

So anyway, I mean, I think this is good news. I know that there are people who dislike the idea. I mean, actually, Leo, I was surprised that it got as much traction as it did, as much pickup in the industry. But it's interesting to know that the downside of "Sign in with ..." is that you are trackable by the third party that you're using as your authentication provider. I didn't realize that was as widely understood as it is. So that's good.

Leo: Yeah. It's a setting, by the way, I love the setting in Brave, the privacy browser based on Chrome. You can turn off those Facebook/Google/Twitter bugs. LinkedIn also is in that list. Block them.

Steve: Except you can't do it in this case. Yeah, you can't do it in this case.

Leo: Not in apps.

Steve: No, I mean, you cannot use OAuth without having that trackability. I mean, the trackability...

Leo: Oh, yeah, yeah, yeah, yeah. So what it does is it turns the bugs off so you can't use it. Basically you have to log in with an email address.

Steve: Well, no, because the bug...

Leo: Which I choose to do because honestly, when I had to leave my Facebook account, it disconnected a lot of accounts I'd used the Facebook login for.

Steve: Yes, yes. Now, our listeners know that there's no way I can talk about the problems with OAuth and what a mess...

Leo: Without SQRL.

Steve: ...all of this is, without noting that there is a working system which is now in place, doesn't have a lot of traction yet, but that's to come. It's now in place. It has

none, not a one, of these issues and problems. And that, of course, is the system that has been my primary focus for the past five years. There cannot be any tracking nor any form of information disclosure because SQRL was designed, as our listeners know, for all of these reasons to be a fully functional two-party authentication system between just you and the site you are wishing to establish a relationship with.

SQRL instantly, uniquely, and pseudonymously identifies its users to new websites and revisited websites without revealing any information of any kind whatsoever. The site receives only a long unique string that only that SQRL user will present to only that website.

I know that I've been talking about this for years, but it's all finally finished. The SQRL Forum site currently has 1,275 registered and active members, all of whom are using SQRL clients daily with Windows, Linux, iOS, Android, Firefox, Chrome, and Edge, to log in without using or needing to supply usernames, passwords, or email addresses. I am very nearly finished with the first section of the final, my final piece of work, which is the SQRL explainer and protocol specification document. I had hoped to have the first section ready to announce today and to offer today. But I missed that mark. I still need to complete the identity replacement and the SQRL service provider API overviews. But it will be ready by next week, and I'll have a link for everyone to download the PDF then.

And by then I will also already be working on the second half, which is the fully articulated SQRL Protocol Specification. And as with all of the first half of that document, all I'm doing is documenting what has already been done, what already exists and has already been proven, and is in active use. So my own SQRL reference client for Windows appears to be completely finished. It's been several weeks since its last update and the release of that, which was a couple tiny little bit of text tweaks. I have one button whose jargon was changed to reduce some confusion. So I expect to make it widely public shortly also. So in other words, everything is finally just about ready. Whew.

Leo: Phew.

Steve: Yes. I had one little bit of fun closing-the-loop feedback regarding, remember, with RDP, we were having fun calling that Really Do Patch. Christophe Vanlancker tweeted, he said: "Dear @SGgrc, you mispronounced," he said, "RDP is Ransomware Deployment Protocol."

Leo: Much better. Oh, my god.

Steve: And as for SQRL and everybody waiting for that to get done, of course that's because they want me to get back to SpinRite 6. I found a nice note from a listener. He said: "Hello, Mr. Gibson. I've been a listener on Security Now! for a few years and always enjoy seeing a new episode show up on my phone Wednesday mornings.

"After listening to Episode 702 and after hearing, not even seeing, you and Leo talk about the PiDP-11" - which of course you have blinking behind you, Leo. He said: "I knew I had to have one." He said: "Beyond building it and getting it running, my main interest was using the front panel switches to do something interesting, such as toggling in the bootstrap loader as you mentioned doing on other PDP machines during the episode. I wasn't able to find any instructions or examples of people doing this on the PiDP-11, so I decided to figure it out for myself.

"After finding the bootstrap code for the RT-11 operating system, I published a short YouTube video today discussing how to configure the PiDP-11 software..."

Leo: I had no idea the switches were even connected to anything.

Steve: "...and toggle the correct switches in order to boot RT-11 OS, which is included in the PiDP-11 software distribution. The Google Group discussion thread is" - and he gives us a link. "And the video itself is" - and then there's a link to the video. He said: "Both you and Leo might be interested in trying this, as it is fairly quick and entertaining to toggle switches and see something useful occur."

Leo: Wow. I have to give our creator of this...

Steve: Oscar.

Leo: ...Oscar Vermeulen a lot of credit because I just figured the switches were cosmetic. I had no idea...

Steve: Oh, no, no, no.

Leo: ...they were connected. They must be connected to the GPIO bus on the Raspberry Pi.

Steve: Well, yeah, indirectly. So, I mean, they actually emulate the PDP-11. And we talked about it at the time. He was so determined to make the switches function correctly that he sat down with a friend in front of an actual PDP-11 and videotaped using the switches. Because, for example...

Leo: The direction has to be right. But I just thought that was for the physical motion. I didn't realize they were actually connected.

Steve: Oh, yeah. So, for example, in fact, you just did it. You just entered an address that you had put into the switches into the address register. Then you toggle them to something else, and you deposit that data into that location. And then the address register auto increments so that you don't have to manually go back and put a different address in. Instead, you're just able to auto increment the address register and enter successive pieces of data. So, yeah. It's the way those machines work. And it's very cool. So anyway, then...

Leo: By the way, that's the video that he created.

Steve: Ah, okay.

Leo: I have no idea what I'm doing.

Steve: So anyway, he finished with what he called the "obligatory SpinRite story." And he said: "This is a fairly mundane success story." And of course, for me, no report of SpinRite success is mundane. He said: "I have a NetBSD server which performs a nightly backup to an eight-year-old 2TB hard drive." He said: "Since this is a nightly task that has run for years, I've been able to watch the hard drive transfer rate steadily decreasing over the past few months until the point where it was obvious that something was wrong, even though the SMART data that I had also been collecting didn't seem unusual or to be deteriorating.

"I had been meaning to run SpinRite on that drive, but didn't get around to it until last week; and when I tried to halt the system, I discovered it was running so slowly that several prior nights' worth of backups were still running, and the drive was completely unresponsive. I eventually just had to force power down the machine. I ran SpinRite on Level 4 for," you know, basically overnight, or actually over a day. He said: "...for a little over 24 hours. There were no errors reported; and, again, the SMART data seemed okay. I reinstalled the drive into the server, and it is now working correctly and performing at its normal speed! Thanks again for the podcast and SpinRite! Signed, Jeff Thieleke." So Jeff, thank you for your report about, well, about your cool project of figuring out how to use a switch register to put the RT-11 bootstrap into the machine.

Leo: Wow. Where did he get the paper tape reader? That's what I want to know.

Steve: And then also your SpinRite story.

Leo: It's really cool, yeah. Really neat. You want to talk about, what is this?

Steve: The Nansh0u Campaign.

Leo: Nansh0u, coming up.

Steve: Nansh0u.

Leo: Nansh0u. All right. Let's find out about this malware here.

Steve: So as I mentioned at the top of the show, and I'll explain a little bit as we get into the details, the thing that's disturbing is that this is a relatively straightforward attack. It is ongoing. It shouldn't ever happen, for a number of reasons. I guess that's true of everything we talk about on this podcast.

Leo: It shouldn't, none of this should happen, but it does.

Steve: Yeah. But what's really interesting is the guys that tracked this down at Guardicore Labs noticed some things that caused them to believe that this was professional, state-level, state actor tools in the hands of non-state actors. So I have the link to their complete write-up, which I've paraphrased here. They wrote: "During the past two months, Guardicore Labs' team has been closely following a China-based

campaign which aimed to infect Windows MS-SQL and PHPMyAdmin servers worldwide. We've taken a deep look into the inner workings of the campaign - the tools in use, the vulnerabilities exploited, and the extent of damage caused." And what's very cool about this is, you know, the reason I like to share these things is the degree of forensic detail that I think our listeners will find really interesting. I did.

They said: "Breached machines include over 50,000 servers belonging to companies in the healthcare, telecommunications, media, and IT sectors. Once compromised, the targeted servers were infected with malicious payloads. These in turn dropped a cryptominer and installed a sophisticated kernel-mode rootkit to prevent the malware from being terminated."

So I'll just pause here and say notice that, again, what's being installed in these things is things that make money. When we first saw, when we touched on the first instance of cryptomining years ago on this podcast, I said, "Ohhh, this is not good." Because, you know, viruses were just sort of like, okay, well, they were an annoyance, but there was really no - there's no reason for it to be done. It bothered people. Maybe some people got their systems compromised. But the incentive was missing.

Well, that changed when cryptocurrency happened because now there was an incentive, the universal incentive of money. And so that's what we're seeing here, 50,000 servers that had a hackable MS-SQL server that now have 50,000 instances of a cryptominer, mining for a shared pool. And not even mining correctly, as I'll get to in a second. So it's like, they got the parameters, in some cases, the parameters backwards. So it's like, oh, my goodness. But anyway, still, the motivation is there. And what they're installing is cryptominers.

So in the show notes I have a picture of a graph of just one month of observation of this campaign showing from early April, on 4/13, April 13th, there were 24,087 infections; and a month later, exactly a month later, on May 13th, 47,985 infections. So it's sort of a, almost, you'd call it a straight line for all intents and purposes. It's got little jiggedy-jags because you're brute forcing usernames and passwords. What's distressing is that it works. I mean, again, there's a mixed blessing associated with publicizing the success of a campaign like this because somebody who would think, oh, that can't possibly work, well, whoops. Yeah, it worked 50,000 times.

So they wrote: "In the beginning of April, three attacks that were detected in the Guardicore Global Sensor Network caught our attention." Okay, so that says they've got honeypots or a network of stuff that are able to pick up these things. They said: "...caught our attention. All three had source IP addresses originating in South Africa and hosted by VolumeDrive ISP." And they said in their notes, "See our IoCs." So that's Indications of Compromise. "The incidents shared the same attack process, focusing on the same service [meaning MS-SQL] and using the same breach method and post-compromise steps." So that said to them, okay, there's something going on here. There's a common actor.

"Looking for more attacks with a similar pattern, we found attacks dating back to February 26th, and over 700 new victims per day. During our investigation we found 20 versions of malicious payloads, with new payloads created at least once a week and used immediately after their creation time." In other words, they found timestamps, like compilation timestamps, still embedded. And so they were being tweaked, built, and then immediately deployed.

"This timeline," they said, "combined with a set of five attack servers and six connect-back servers" - normally referred to as command-and-control servers - "suggests an established process of continuous development which was well thought out by the attackers. Having access then to the attacker's infrastructure, we were able to monitor

the relevant file servers and draw insights on the campaign's scope and size. The graph on the page above" - the one I showed - "shows how the number of successfully infected machines doubled within the time span of one month. Each attack started with a series of authentication attempts to an MS-SQL server" - oh, I said MySQL. I meant MS, sorry. Well, any SQL server - "eventually leading to a successful login with admin privileges. Then a sequence of MS-SQL commands was executed to accomplish the following."

So basically they brute forced their way in to get in with admin privileges. Then they configured the server settings to allow what they called a "smooth and error-free attack flow." They created a Visual Basic script file in the ProgramData folder which was numeral 2.vbs, executed this script and downloaded two files into that same folder over HTTP, then ran those two files in a single command line.

They wrote: "The attacker's servers were all running HFS - HTTP File Server - serving files of different types. One text file included the string Nansh0u [N-A-N-S-H-0-U] on which we based the campaign's name. The attacker's infrastructure contained all the modules required for a successful end-to-end attack on MS-SQL servers: a port scanner, an MS-SQL brute force tool, and then the remote code executor. The port scanner used by the attacker had been known since 2014," so off the shelf. "Our attacker used it to detect MS-SQL servers by scanning IP addresses and checking whether typical MS-SQL ports were open. The ports examined were 1433, 2433, 1533, 1143, 9433, and 5433.

"The results of the scanning were then fed to the brute force module. The brute forcing tool attempts to log into each MS-SQL server using tens of thousands of common credentials. Once authentication succeeds, the server's address, username, and password are saved to a file for subsequent use. The dictionaries of common credentials used by the attacker can be found in the campaign's Indications of Compromise repository on GitHub."

And in fact I have the link, as I mentioned, to the GitHub repository on GitHub. I think I looked, it was like a 1.x megabyte file, so just a massive text file of usernames and passwords which are used in a simple brute force attack. So not passwords like would be generated by LastPass, but like brain-dead passwords that no admin should ever use for their server.

"By running the port scanner and the brute force tool, the attacker gained a list of breached servers' IP addresses, ports, usernames, and passwords. The next step was to log into the victims and infect the machines. On the HFS (HTTP File Server), in a folder named 'chuan,' C-H-U-A-N, which is Chinese for 'infect,'" they wrote, "we found two interesting components. The first was a script named 'Mssql.log.' The script's commands were the exact ones we saw in the incidents that triggered our original research.

"The second file was an executable named 'Usp10.exe.' This program receives a server's address and credentials, as well as the contents of the Mssql.log. Its functionality is straightforward. It logs into the breached server and executes the Mssql.log script, proceeding with the attack on the victim's side. Usp10.exe is the attack module responsible for writing and executing the downloader script (2.vbs) on the MS-SQL victim. Once the MS-SQL script is done running on the victim machine, the malicious payload is executed. In all the attacks we have seen as part of this campaign, the attacker executed a command line with two executables." And then they show c:\windows\system32\cmd.exe, which as we know is the standard Windows command interpreter, the standard command line, what you used to call the "DOS box." And so then it executes programdata\apexp.exe and programdata\tl.exe.

They wrote: "This command executes a privilege escalation exploit" - so once again here's a perfect example, in the wild, of why a privilege escalation exploit by itself doesn't, you know, it isn't remote code execution, but it factors into and is crucially

important for virtually all other non-directly remote code execution exploits. This is only one step removed.

So "This command executes a privilege escalation exploit that runs a malicious payload with system privileges. In the attacker's arsenal are two versions of the PE exploit" - PE is Windows' Portable Executable format - so "apexp.exe and apexp2012.exe, and many payload versions. This apexp.exe and apexp2012.exe," they write, "are two exploits of a known privilege escalation vulnerability from 2014 (CVE-2014-4113). Passing any program to these executables will run it with system privileges.

"Apexp.exe is known as the Apolmy [A-P-O-L-M-Y] exploit, and it affects both desktop and server versions of Windows from XP to 8.1 and Server 2003 to 2012 R2, respectively. This," they write, "is a weaponized exploit with production-level code. Apexp2012.exe, on the other hand, resembles more of a proof of concept than an operational exploit and is designed to work on Windows 8.1." They wrote: "We found the latter available for download on a Chinese forum seemingly used as a hacker community forum.

"While both versions use the same vulnerability, they execute kernel-mode code for different purposes. Apolmy version copies the system process access token to its own process. With that token, the exploiting process runs the payload with full control over the victim machine. The second version uses a method commonly popularized. Here, the exploit adds the SeDebugPrivilege to the token. Using this Windows privilege, the attacking exploit injects code into the Winlogon process. The injected code creates a new process which inherits Winlogon system privileges, providing equivalent permissions as the prior version."

They didn't say this, but the reason you would use that second one, which seemed less professional, is that it might be that the earlier exploit had been patched. After all, it's been known since 2014. So one would hope that maybe the server that unfortunately has its MS-SQL server available through brute force admin login, it might have had the first problem patched, but no remediation for the second.

So then they said: "We collected 20 payload samples from both the attacker's servers and Guardicore Global Sensor Network. Each payload is in fact a wrapper and has several functions: execute the crypto-currency miner, create persistence by writing registry run-keys, protect the miner process from termination using a kernel-mode rootkit, and ensure the miner's continuous execution using a watchdog mechanism. The payloads spawn one of two processes, either dllhot.exe or canlang.exe, depending upon the payload, which in either case mine a private cryptocurrency named TurtleCoin for four different mining pools.

"Many of the payloads drop a kernel-mode driver, named randomly and placed in AppData/Local/Temp. Its compile time suggests that it had been created in 2016. Nevertheless, most AV engines do not detect the driver as malicious. We found that the driver had a digital signature issued by VeriSign. The certificate, which is expired, bears the name of a fake Chinese company, Hangzhou Hootian Network Technology. Guardicore Labs contacted VeriSign and provided them with the relevant details, which resulted in the certificate's revocation." Not that it matters because I think they said it was expired. Maybe not. I remember taking a look at it myself and noticing that it did not look like the malware was timestamped. So if the certificate expired, the malware would no longer be trusted.

Anyway, they said: "This would have been less awkward had the driver not been packed and obfuscated. Unlike many other malicious drivers, this driver is protected and obfuscated with VMProtect, a software tool that attempts to frustrate reverse engineers and malware researchers. The driver is designed to protect processes and prevent the

user from terminating them. It creates a device named SA6482, allowing processes to communicate with it. The device receives process IDs meant to be protected," they said, "in our case, the cryptominer's process ID. The driver protects the process by registering a callback on the process and thread objects. These callbacks are triggered with every process to the protected process or any of its threads and allows drivers to modify the access rights given in each access attempt."

So anyway, the point is - I'm going to skip. There's a little bit more detail. Everybody gets the idea that this thing, it gets in through a brute force attack on a publicly exposed SQL server using an exhaustive list, but obviously an effective list because they've got more than 50,000 instances now compromised that way, of usernames and passwords. And it doesn't care about where it is. It doesn't try to do anything other than make money immediately. Sets up a cryptominer that mines with a pool and has at it, and then installs a driver, a kernel driver whose job is to protect the mining process, to basically fight back against it being terminated. So that technically is a rootkit because you would not then be able to terminate the process. And even if you restarted the server, it's arranged to get itself to restart and reprotect itself. So you'd have to go to some additional lengths in order to keep that from happening.

They noted that, as I mentioned, that the tools, the techniques that were used belonged to nation-state level hackers, but were, due to a lot of the things that they saw, apparently being used by much lower level hackers. The attacks used advanced encryption technologies and rootkit hiding, but the attacks themselves were not very sophisticated. At one point they noted that many of the programs found on the server - oh. The servers themselves had no authentication protection. Once they identified the IPs, they were able to immediately browse the servers in order to then monitor the attacks. So nothing was preventing that from happening. And a bunch of the programs found on the server were written in something known as EPL, which is Easy Programming Language. It's a proprietary Chinese-based programming language designed for rapid application development. So that made them think that this was not fully nation-state level attack.

So anyway, it's interesting that - it's hard to understand how MS-SQL admin account would be exposed to the Internet. Maybe there's some reason. But if there is, do not use a password that's going to be on somebody's password guessing list, or this is what happens to you. So anyway, I just thought it was a really interesting forensic look at how somebody is finding machines, brute forcing their way in, and setting up cryptocurrency mining. And this basically connects back to the beginning of the podcast, when I said, given that there are, okay, so this was going in the hard way over the course of several months to obtain 50,000 machines. Now we have a million open RDP machines just waiting, just begging to have cryptominers set up in them and have the door close behind them. I'll be really surprised if that's not exactly what happens.

Leo: Of course, a Windows XP machine isn't going to be a great cryptominer. But what the hell.

Steve: Yeah. Yeah.

Leo: It's free.

Steve: Well, and no longer do you need to mine individual coins. You now just contribute your resources to a pool.

Leo: Yeah, a million of them.

Steve: And so any cycles that it has available, exactly, you've got numbers on your side.

Leo: Yeah, yeah.

Steve: Yup.

Leo: All right. There you go. I'm not surprised. Anybody using PHPMyAdmin, there's a lot of people who, I mean, I did this, you know. Your first time you set up WordPress you set up your MySQL. You're following a cookbook on the web. You might not make sure it's got a good password because you want to remember it, and not even realize it's online. I could see how that can happen. This is all designed to make it really easy for people to do this kind of stuff.

Steve: Yes. And if it's on a - they did say MS-SQL. It's oftentimes...

Leo: Oh. It's SQL Server. It's not MySQL.

Steve: Right, right.

Leo: Oh, oh, oh.

Steve: But it is oftentimes the case that you will unwittingly bind the server to asterisk rather than to the local interface, and so you are unwittingly...

Leo: Opening it up, yeah.

Steve: Opening that port to the rest of the world, too.

Leo: In other words, it's an easy mistake to make, but it's why we always emphasize you shouldn't be running servers if you don't know what you're doing because a server is open to the public.

Steve: Right. Right.

Leo: That's the problem, yeah.

Steve: And just stick it behind a router. If you stick it behind a router and then only...

Leo: But then it doesn't serve.

Steve: Yeah. Well, but then you only open the ports that you need to.

Leo: Right, right, right, yeah. Of course. I mean, securing this isn't hard. It's just that people aren't paying attention.

Steve: Right.

Leo: They don't know what they're doing. They're following a recipe they read somewhere. Ah. Oh, well. Steve, as always, an eye opener. And that's why people tune in every Tuesday around about 1:30 Pacific, 4:30 Eastern, 20:30 UTC to listen to or watch Security Now!. Steve has copies of the show at his website, GRC.com, along with transcripts. I know a lot of you like to read while you listen. And that's a great way to do it. Those are very well-written, human-transcribed transcripts of the show. They even get our exclamations like "Whoo" or "Wow." I don't know how Elaine transcribes those, but I'm thinking W-H-O-O?

Steve: They're all there.

Leo: They're all in there. While you're there, pick up a copy of SpinRite, the world's best hard drive recovery and maintenance utility. If you have a hard drive, you need SpinRite.

Steve: And even if your hard drive is just running slowly.

Leo: Yeah. There's a reason.

Steve: That's right.

Leo: That's a sign. That's a symptom.

Steve: And SpinRite can fix it.

Leo: Yeah. Steve's got a lot of other stuff there. In fact, it's a great place to hang out: GRC.com. You can leave Steve feedback at GRC.com/feedback, or follow him on Twitter, @SGgrc. And you can always Direct Message him there. There are a couple ways to reach Steve. We have audio and video of every show at our website, in fact of all of our shows, TWiT.tv - TWiT.tv/sn for Security Now!. It's also on YouTube. I mean, it's everywhere.

If you have a podcast program you like, you could subscribe there. In fact, that's probably the best way to get a copy of the show. And don't forget you can ask your smart voice assistant. In many cases, all it takes is "Play Security Now! podcast," and you'll be able to hear the latest episode. Or if you want to listen live, "Play TWiT Live," and get what you want. Thanks, Steve. We'll see you next week on Security Now!.

Steve: Thanks, buddy.