REvil Vanishes!

Description: This week we look at the continuing attacks on Chrome with yet another zero-day and at Mozilla's continuing work to give their users the most privacy possible. We reexamine that iOS WiFi SSID bug and a related bug which, it turns out, Apple apparently knew was a showstopper. Amazingly, two more new problems have surfaced with Microsoft printer technology. We have a review of last week's Patch Tuesday including the importance of also updating any instances of Adobe's Acrobat and Reader. We revisit an old friend and consider the folly of rolling one's own crypto. We look at the explosive revelations surrounding the widespread abuse of iPhone and Android "surveillance-ware" produced by the NSO Group. And finally, after sharing one fun piece of errata, we're going to finish by examining the curious, sudden, complete and total disappearance of the REvil ransomware organization.

High quality (64 kbps) mp3 audio file URL: http://media.GRC.com/sn/SN-828.mp3
Quarter size (16 kbps) mp3 audio file URL: http://media.GRC.com/sn/sn-828-lq.mp3

SHOW TEASE: It's time for Security Now!. I'm back. Steve Gibson's here. More Chrome zero-days. A revisit to the SSID flaw in iOS and how it happened and how it wasn't fixed. And yes, the PrintNightmare is back, baby. All that and more coming up next on Security Now!.

Leo Laporte: This is Security Now! with Steve Gibson, Episode 828, recorded Tuesday, July 20th, 2021: REvil Vanishes.

It's time for Security Now!, the show where we cover the latest security news, information on how to keep you safe, how things work, with a very hot Steve Gibson.

Steve Gibson: Coming to you from the alternative location, under the air conditioning, I'm happy to say. The air is pouring right down on me. I had a, after 37 years, actually I think this is the third air conditioner I'm on since I purchased my home in '84 and found a great Southern California air conditioning firm. I'll give them a free plug, Rohan and Sons, for anyone who's in Southern California. They are...

Leo: Nice. By now it's got to be the son that comes over.

Steve: Actually, I think Dad's wandered off since then. That was, you know, 37 years ago.

Leo: Yeah, I'm thinking he's retired by now.

Steve: And they're just, in fact, it was so inexpensive - they recharged my Freon February a year ago, before COVID - that I felt badly that he, like, charged me 20 bucks. And so I gave him a hundred dollar bill just because it's like, come on, that's just ridiculous. He came out with all of his equipment and figured out what was going on and everything. So anyway, they are great. But anyway, so I believe the control relay died since there's nothing happening on the outside of the system. So I did all the podcast prep and production and everything, and it was 88 degrees in my office a couple hours ago, and I had already told Lorrie that I had a feeling I'd be seeing her a little bit after noon today. And so I brought my traveling road show with my Heil getup and my mic and everything.

Leo: You must be something getting out of the car with all that stuff under your arm. That's funny.

Steve: But anyway, we've got a bunch of fun stuff to talk about, a bunch of fun stuff for this 828th podcast. We're going to look at the continuing attacks on Chrome with yet another zero-day, and on Mozilla's continuing work to give their users the most privacy possible. They really are sort of forging the path on that. We're going to reexamine that iOS WiFi SSID bug. Remember the print format string.

Leo: Oh, yeah, yeah, that came back.

Steve: Turns out it - yeah, yeah, yeah. And apparently Apple knew more about it than we did before, even, and it was a showstopper. Amazingly, we have two more new problems which have surfaced with Microsoft's printer technology.

Leo: Can you believe that? What is that, five now total?

Steve: Yeah, I mean, what's happening now is as I'm, like, tracking down the news, I'm having to make sure I'm not, like, repeating them because it's - and I've seen people tweeting, like security researchers, we're having trouble keeping track of this. I's just crazy.

Leo: Wow.

Steve: We've also got a review of last week's quite busy Patch Tuesday, including the importance of also updating any instances that you may have of probably Adobe's Reader, but it also affects Acrobat. We're going to revisit an old friend and consider the folly of rolling one's own crypto. And we look at the explosive revelation surrounding the widespread abuse of iPhone and Android surveillance-ware produced by the NSO Group. I know you guys talked about it on MacBreak Weekly. And, finally, after sharing one fun piece of errata which surfaced after something I said last week, we're going to finish by examining the curious, sudden, complete and total disappearance of the REvil ransomware organization.

Leo: Curious.

Steve: Thus the title of today's podcast, REvil Vanishes. And we do have a really - this is just - this warms my heart, the cockles, our Picture of the Week. So I think another great podcast.

Leo: Very nice. Very nice.

Steve: Picture of the Week.

Leo: Yes.

Steve: It just really, as I said, warmed my heart. We're looking at the top of page 342, Chapter 8, titled "Principles of Security Models, Design, and Capabilities. This is the official CISSP "Certified Information Systems Security Professional" study guide. And the discussion we see from the page, it says: "However, when the speculative execution is wrong, the procedure is not completely reversed, i.e., not every incorrect predicted step is undone. This can result in some data remnants being left behind in memory in an unprotected state." Then it talks about, in the next paragraph: "Meltdown is an exploitation that can allow for the reading of private kernel memory," blah blah blah.

It finishes all this, and then this callout finishes, saying: "For a thorough discussion of these concerns, please listen to the Security Now! podcast or read the show notes of episodes 645, "The Speculation Meltdown"; 646, "InSpectre"; 648, "Post Spectre"; 662, "Spectre NextGen," at www...

Leo: We did a lot of Spectring.

Steve: Yeah, well, that was the big news of that year. So at www.grc.com, blah blah. Anyway, Chapter 8, Official CISSP certification study guide. So, yeah, we're...

Leo: You're famous, Steve. I could get you $60 million now from Spotify.

Steve: And a thank you to Chuck Littlefield for taking the picture and sharing it with me through Twitter. I appreciate it, Chuck. Very cool.

So the attacks on Chrome continue. Google has released 91 blah blah blah dot 164 for Windows, Mac, and Linux, which fixes seven security vulnerabilities, one of them a high-severity zero-day being actively exploited in the wild. Of that one, which was a CVE ending in 30563, Google said that it's aware of reports that an exploit exists in the wild. And as usual, Chrome will eventually auto-update, I suppose. But every time I check, I catch it off-guard. When I checked last night, I was still running .124, you know, 91 blah blah blah .124, rather than .164. So the act of checking, going to setting Help About Google Chrome, triggered its update. And after a restart I was then current.

And we are keeping score here. This CVE, 30563, brings the total count of exploited-in-the-wild critical zero-day flaws patched so far this year to eight. So a little better than one a month. It's been a rough year so far. This one was another type confusion bug in

Chrome's V8 engine, which as we know is their high-performance WebAssembly and JavaScript processing subsystem.

One little tidbit was particularly interesting. Google stated that, based upon their analysis - of what they didn't say - two of those eight zero-days, 21166 and 30551, those are both previous this year, had been developed and sold by the same vendor providing surveillance capabilities to customers around the world. Okay.

So then last Thursday, Microsoft and Citizen Lab linked the vendor mentioned by Google's Threat Analysis Group, that's their TAG group, as the Israeli spyware vendor Candiru, C-A-N-D-I-R-U. It's believed that threat actors deployed Candiru's surveillance spyware to infect iOS, Android, macOS, and Windows devices using Chrome zero-days and unpatched Windows flaws - there were also some flaws in IE - in order to get into their systems. So clearly Google is doing everything that they can. And what we are seeing, well, we're seeing many different things. In this case we're seeing that because today's systems are so complex, despite full focus on making them as secure as they can, we're still seeing bad guys discovering zero-days. And boy, are we going to be talking about that a little bit later.

Also in the browser front, Firefox has special-cased anti-tracking for those "Login With" functions, you know, Login With Google, Login With Facebook, sometimes it's Connect with Facebook. When Firefox's full anti-tracking protections, which they've been working on and we've been reporting on, are enabled under Firefox's strongest privacy-protecting incognito browsing mode, those increasingly popular logon-with-some-other-site features, which is accomplished with scripts that can inherently also be used for tracking, they don't work because the protection is too strong, or as strong as it needs to be. And that was causing trouble. So the just-released Firefox 90 - and once again I was running 89, and so I said, uh-oh, and restarted, did the what-about and got 90. 90 resolves this dilemma.

So here's what Mozilla explained, naturally with a bit of a sales pitch spun into this. They said: "Today, with the launch of Firefox 90, we are excited to announce a new version of SmartBlock, our advanced tracker blocking mechanism built into Firefox Private Browsing and Strict Mode. SmartBlock 2.0 combines a great web browsing experience with robust privacy protection by ensuring that you can still use third-party Facebook login buttons to sign into websites, while providing strong defenses against cross-site tracking.

"Logging into websites," they say, "is of course a critical piece of functionality. For example, many people value the convenience of being able to use Facebook to sign up for and log into a website. However, Firefox Private Browsing blocks Facebook scripts by default." Yay. You know, it should. They said: "That's because our partner Disconnect includes Facebook domains on their list of known trackers." Again, yes, they should.

So they said: "Historically, when Facebook scripts were blocked, those logins would no longer work. For instance, if you visit Etsy.com in a Private Browsing window, the front page gives the following options to sign in, including a button to sign in using Facebook's login service." Which of course we all know is OAuth. They said: "If you click on the Enhanced Tracking Protection shield in the address bar," you know, while you're at Etsy, and click on Tracking Content, you will see that Firefox has automatically blocked third-party tracking content from Facebook to prevent any possible tracking of you by Facebook on that page. Prior to 90" - that is, this most recent released Firefox - "if you were using a Private Browsing window, when you clicked on the 'Continue with Facebook' button to sign in, the sign-in would fail to proceed because," they wrote, "the third-party Facebook script required had been blocked.

"Now, SmartBlock 2.0 in Firefox 90 eliminates this login problem." Okay, albeit at some [audio glitch] privacy; right? Because you can't have both, unfortunately, with OAuth.

They said: "Initially, Facebook scripts are all blocked, just as before, ensuring your privacy is preserved. But when you click on the 'Continue with Facebook' button to sign in, SmartBlock 2.0 reacts by unblocking the Facebook login script just in time for the sign-in to proceed smoothly. When this script gets loaded, you can see that unblocking indicated in the list of blocked tracking content." In other words, it's no longer present as being blocked.

"SmartBlock 2.0 provides this new capability on numerous websites. On all websites where you haven't signed in, Firefox continues to block scripts from Facebook that would otherwise be able to track you. You don't have to choose between being protected from tracking or using Facebook to sign in." Well, of course, until you actually do. "Thanks to Firefox SmartBlock," they finish, "you can have your cake and eat it, too."

So, okay. It's obvious to anyone why sign-in with Google or Facebook are compelling offers to the typical user. Right? They have no way of appreciating, or maybe they don't care, that Google and Facebook gleefully offer these sign-in services because the user's browser is being, I mean, in exactly this way that Firefox has just said, okay, we're going to conditionally drop our guard, right, because we have no choice. So users have no way of knowing that Facebook and Google are gleefully offering these services because a user's browser is being redirected through them, allowing them to statically tag the user's browser with an identifying first-party cookie which is about as non-anonymous as anything could be since the user is using their Google or Facebook identity as their surrogate login identity. And not to mention that the surrogate also knows where they have just logged into.

So, you know, I think it's very cool that, in the first place, Firefox's browsing privacy protections are strong enough that this clearly privacy-bypassing process was blocked even to the inconvenience of those users because they desired strong privacy. And private is one thing that OAuth is not. And I also think it's exactly right that Mozilla then stepped up and opened just the tiniest of all possible privacy exemptions or exception windows to allow the indirect login flow to succeed.

So, yeah, I say Bravo to Mozilla for their execution of this. That doesn't make OAuth any better, but we're currently living in a land of significant convenience versus privacy tradeoffs. And what they've done is they've kept from breaking functionality, which might cause users not to use in-private browsing, not to use Firefox because they think it's broken. And they want to use their login convenience, yet just open the tiniest little window to allow the OAuth browser flow to succeed.

So again, not the ideal solution. The ideal solution would be something that is, well, we all know that I spent seven years working on, that is 100% private because it's a two-party login, not a three-party login, as OAuth is. But we don't have SQRL, so we have convenience with a little tiny bit of privacy sacrifice. And, you know, again, probably no one really cares.

Okay. Oh, wow. iOS WiFi SSID bug. I think it's worth reinforcing first the critical security principle Bruce Schneier captured when he wrote that: "Attacks always get better. They never get worse." Okay. Recall last month Apple iOS WiFi SSID bug that we had fun talking about. That was the one security researcher Carl Schou tweeted: "After joining my personal WiFi with the SSID '%p%s%s%s%s%n,' my iPhone permanently disabled its WiFi functionality. Neither rebooting nor changing SSID fixes it." And of course that led to the discovery that they had not sanitized the SSID string prior to presenting it to a function, you know, some version of printf which was interpreting those percent things as things to be expanded and looking for parameters on the stack or from the caller that weren't there, thus causing a crash. And the concern was, as Bruce points out, attacks always get better. So maybe this could be leveraged into such an attack.

Now, we talked about the inherent danger of what is an incredibly convenient shortcut that exists in many programming languages, that is, lots of languages do this. One of this podcast's other observations is the danger inherent in interpreters; right? We're always talking about interpreters. They're hard to get right. And what we have here with the percent character escape is an interpreter where the printf function, or one of its cousins, is reading and interpreting the string on the fly, as it encounters it. No WiFi radio's SSID should contain interpretable percent sign escape sequences. The bug that was discovered in iOS was that SSIDs, which are in this case attacker-controlled, were not being sanitized by first doubling-up all percent characters into percent percent pairs, which would then be treated as a single literal percent without any special interpretation.

And the reason we're talking about this again and reminding everyone about Bruce Schneier's pithy observation is that yes, indeed, that flaw or actually a slight variation turned out to have been weaponizable. After studying the trouble and verifying that it's much worse than we were told, security researchers with ZecOps, Z-E-C-O-P-S, have nicknamed the issue "WiFi Demon," discovering a zero-click drive-by vulnerability that allows an attacker who controls a nearby WiFi hotspot to infect an iOS device without any user interaction, when the iOS device has its default setting for WiFi to automatically join WiFi networks. Even if they're not joined, just the act of sniffing the maliciously crafted WiFi SSI beacon is all that's required.

Okay, now, I used the phrase "much worse than we were told" because Apple was apparently aware of this, and elected not to tell anyone.

Leo: Oh.

Steve: Even after the fact. Yes.

Leo: So there was a little conversation, I don't know if you heard it on MacBreak Weekly, saying, well, these guys should have followed normal disclosure policies and sent the exploit to Apple. But that's not what - that wasn't the problem. Apple knew already.

Steve: Well, Apple knew and silently fixed it. The flaw was introduced with the release of iOS 14.0 last September.

Leo: This is just the latest flaw, by the way. There have been serial flaws that NSO has been using.

Steve: Oh, yeah, yeah, yeah. So we're just talking about one of these. And it's not clear that this one would have been weaponizable for the NSO Group's purpose because this is strictly local. You have to be...

Leo: Oh, yeah, yeah.

Steve: You have to present a WiFi beacon to iOS.

Leo: Oh, I remember that, yeah, yeah.

Steve: In order for this one to get it. But Leo, wait till you hear what this is. So this popped into existence last September with the release of iOS 14.0. Apple quietly patched the issue in January this year as part of their iOS 14.4 update. So this hasn't actually been a problem for a while.

Leo: Right.

Steve: That SSID string which Carl found, that will be fixed. We're still living with it. It's going to be fixed in 14.7.

Leo: Okay, that came out today.

Steve: Oh, just, okay, just came out.

Leo: Just came out.

Steve: Okay. Okay.

Leo: This is hysterical.

Steve: Apple never made any mention of it, nor did they bother assigning a CVE identifier to the flaw. Okay. As we know, the 14.4 update did not fully fix all the problems, since Carl Schou's discovery of that wacky SSID is still workable, or was until today. But it was yesterday under iOS 14.6. It has now been finally fixed, finally, fully, in today's iOS 14.7 update, which when I wrote the show notes was undergoing final prerelease beta testing. So how do we know that Apple knew, and this dangerous WiFi SSID remote code execution vulnerability didn't just coincidentally disappear on its own? You know, that could have happened. We know this because of what Apple quietly removed to fix the trouble.

We've talked about the inherent trouble with percent escapes. Well, until iOS 14.4, right, in January, another incredibly dangerous escape sequence was being honored, %@. In Objective C, the %@ escape instructs the interpreter that the associated parameter is a pointer to an Objective C object, which should be printed. So lord only knows what sort of wild goose chase of interpretation would have ensued if this was encountered, if %@ was encountered in an SSID. And actually we do know what sort of wild goose chase would ensue since the ZecOps researchers wrestled this beast to the ground and positively verified that until the interpretation of the %@ was silently removed by Apple at the beginning of the year, it was definitely possible to trigger an attacker-controlled remote code execution.

But now think about it. This raises an even greater issue. Which is worrisomely similar to Microsoft's failure to fully patch the PrintNightmare flaw the first time, and instead, as we discussed, only patching to fix the provided proof of concept demonstration of the flaw. Here's my concern in the case of Apple. Someone at Apple was apparently tasked with removing the handling of %@ from an attacker controllable string, in this instance a WiFi SSID.

But they left all of the other percent escape sequence interpretation in place, none of which should ever happen on an SSID, and once again only repaired one specific instance of the actual bigger problem, which was looking at them in the face, rather than seeing the forest and realizing, uh-oh, hold on a second, why exactly are we interpreting any percent escape sequences in this part of the code? You know, it should have been fixed in January. It wasn't. Carl found it, and we've been living with other instances of percent escape problems until finally, with 14.7, they fixed it.

So, you know, if this is indicative of a larger emerging trend in our industry, then Leo, this podcast is going to require six digits for its episode numbering.

Leo: So this is the second time we've talked about a patch only fixing one immediate part of it - Microsoft did the same thing with the spooler patch - and not the overall problem.

Steve: Yes.

Leo: They should never let any of those strings through, ever. That's ridiculous.

Steve: No, no. And so, you know, who knows? Maybe this was being used in a remote code execution exploit, and so somebody was assigned with, uh-oh, how are they doing it? Oh, look, they're sending a %@ in a string. Oh. Take that out. We don't need that. But they should have fixed the problem of interpreting all the other percent escapes. Ugh. I don't know.

Leo: So the speculation was that some renderer gets this, you know, obviously it's not in the deep WiFi code. It's probably somewhere like the renderer that displays it on your iPhone screen, the name of the SSID, or something like that.

Steve: No, except you don't have to have your phone unlocked and showing it. So it is, I mean, it could be a renderer, like, is assembling a list, which would then be shown if you unlocked your phone.

Leo: Right.

Steve: But this thing can be in your pocket, locked.

Leo: Right. So zero-click. Those are the worst.

Steve: And it'll knock out your WiFi just when the phone sniffs it. So it is somewhere in some parsing, you know, beacon parsing code somewhere.

Leo: And the reason I mention it is maybe they wanted to leave some formatting strings in there. I mean - are you drinking a glass of milk? Because that's really good.

Steve: No, it's a paper wrapper around.

Leo: Okay. I was going to, well, I do see your milk moustache, but that's another story, for another day. No, I think that maybe - I'm looking, making up excuses for them. But they wanted some formatting capability in these SSIDs being displayed, so they didn't want to take it all out. I don't know. There's no excuse. You're right.

Steve: Well, and we know that an SSID, okay, obviously a percent symbol is legal in an SSID.

Leo: Right.

Steve: I mean, you can do that. It's crazy, but yeah. I mean, it would make it harder to brute force, I suppose. So all you...

Leo: You escape it, though. You don't...

Steve: So exactly. Escape it. That's the solution. If you want to show them...

Leo: I mean, that's what you do in general on the web.

Steve: Yes.

Leo: Okay.

Steve: Anyway, again, anybody can make a mistake. I'm not saying that was the problem. The problem was they didn't fix the problem when it was shown to them.

Leo: Right. They fixed a subset. Yeah. They fixed a little subset of the problem.

Steve: Yeah.

Leo: It's very strange.

Steve: Let's hope they have now. I mean, lord knows, we don't know if they've fixed it yet.

Leo: Right.

Steve: We just know that %n is gone or whatever it was that was decided to be causing the problem. Boy. Okay. So we still, unbelievably, are unable to awaken from the PrintNightmare. We began last week's podcast while you were wherever you were, Leo.

Leo: You have any guesses where I might have been? Any thoughts at all?

Steve: Yeah, I loved the safari hat that you had on for MacBreak Weekly. Wherever you were, we were observing that Microsoft PrintNightmare was still with us. And believe it or not, even after last week's Patch Tuesday, which we'll get to next, the nightmare continues. Reporting all this, as I had mentioned before, has been a bit of a challenge because I've been needing to make sure that I'm not re-reporting something that we've talked about before because it's hard for me to believe we're still coming up with new problems. There have been so many similar and related problems with discoveries and announcements and patches. And as I said, I've seen other researchers saying that it's becoming difficult to keep up.

In this latest case, after carefully double-checking, I'm quite certain that we have two more newly discovered problems with Windows printer driver installation being leveraged into an escalation of privilege to system root kernel level. And one of the things that we discussed last week, and we're actually going to come back to this, is Microsoft explained in their bulletin for the emergency patch, remember when you and I were doing the podcast week before last, during the podcast, Microsoft published the out-of-band, out-of-cycle emergency patch for the PrintNightmare. We of course later learned that it was different than the one that the previous Patch Tuesday fixed. And then we learned that it didn't actually solve the problem, that there was a workaround for it.

The workaround turned out to be, if you had enabled a feature which first appeared in Windows 2000, PointAndPrint, then you still had a problem. And when everyone said, aha, Microsoft, you didn't fix it, Microsoft amended their bulletin to tell us that, if you had these two keys in your registry enabling PointAndPrint, your system was, and I quoted this, they literally said "vulnerable by design."

Leo: We meant to do that.

Steve: Yes. And of course we paused the podcast to note that we'd been looking for a slogan for Windows 11. And now we had it.

Leo: Vulnerable by design.

Steve: Windows 11: Vulnerable by Design. So believe it or not, that wasn't the end of it. So for the first of the two problems we're going to talk about today, Microsoft has assigned the first one CVE-2021-34481. And it's been given a CVSS severity score of 7.8. Microsoft writes: "An elevation of privilege vulnerability exists when the Windows Printer Spooler service improperly performs privileged file operations." Very generic. They've got some monkey, I think, that types these; right? Or just selects one from column A and one from column B because they're so generic.

But they continue: "An attacker who successfully exploited this vulnerability could run arbitrary code with system privileges. An attacker could then install programs; view, change, or delete data; or create new accounts with full user rights. An attacker must

have the ability to execute code on a victim system to exploit this vulnerability. The workaround for this vulnerability is stopping and disabling the Print Spooler service."

And we also talked last week about how it was the case that you and I, Leo, apparently because we're old-school, believed, as I did until I tried it, that you could stop the print spooler, and everything would be fine.

Leo: Yeah. I got a lot of messaging about that, yeah.

Steve: Oh, boy, yes. We weren't through with the podcast before Twitter blew up.

Leo: It was complicated. It's not - it's complicated.

Steve: Right, right. Okay. So in other words, with this latest escalation of privilege vulnerability, we are back to disabling Windows print spooler service because some way has been found to use it to bypass Windows security privilege system to obtain full system privilege. As Microsoft's note says, this is a pure elevation of privilege. An attacker must have already obtained the ability to execute code on a target system. It can only be exploited locally to gain elevation of privileges on a device. On the other hand, as we know, it's often easy to get onto a device as the unprivileged user, and what you want to do is elevate your privilege to root so that you can then really do dastardly things, like if you're on a domain controller. So it's like, yeah, that's what you want.

In their bulletin's FAQ, they ask and answer the first two questions: Is this vulnerability related to the previously addressed CVE-2021-1675 and CVE-2021-34527 vulnerabilities. And the response from Microsoft, this distinct vulnerability also exists in the print spooler service. However, the security impact is local elevation of privilege. So in other words, yeah, this is another one. Okay. And then the second question, did the July 2021 security update, that is to say last Tuesday's patch update, introduce this vulnerability, since where did it come from? And Microsoft confesses: "No, the vulnerability existed before the July 13, 2021 security update. We recommend that Microsoft customers install the latest security updates." Because, after all, we would like to whittle the count down, as they did by 117. But anyway, we'll get to that in a minute.

Okay. So there's one. That's the first one. Okay. So that was the first problem, another new distinct from the first two privilege escalation flaw in printer server. There's no indication whether it's being attacked in the wild. Maybe we're going to wait a month for August's Patch Tuesday. Maybe somebody will figure out that - or that they see it's being used in the wild by some malefactors, then we're going to get an update. Who knows? But at this point it is publicly known that there is such a problem.

And since print spooler is running on all machines by default, you know, as we've been saying now for two weeks, if you can stop your spooler service, if there's machines you've got around that do not need to have the print spooler running, like probably your domain controller, I don't know, maybe you've got printers attached to it, and it's a server also. But where you can, stop print server. Disable it. If you don't need it, that's always just good advice. Don't have excess code running because every blob of code that's running is another opportunity for exposure.

Okay. So that's the first one. This one is interesting because this is another by-design problem, and I don't think it can get fixed.

Leo: Ooh, that's not good.

Steve: No. Okay. So this falls under it's not a bug, it's a feature, because it's a consequence of Microsoft's deliberate system-level design. The latest technique for abusing what is beginning to look like some serious fundamentally poorly designed systems within Windows printing is brought to us by Benjamin Delpy. Ben is the creator of Mimikatz, which we've talked about from time to time just in passing. He originally created Mimikatz as a proof-of-concept demonstration to Microsoft that their authentication protocols were vulnerable to attack. In doing so, he also created what has become one of the most widely used and downloaded hacker tools of the past 20 years. Jake Williams, president and founder of Rendition Infosec, has been quoted saying that Mimikatz has done more to advance security than any other tool he can think of. So Benjamin has some Windows hacking cred.

In his tweet last Wednesday, after this month's patches had landed, and Microsoft had explained to the world that when Windows PointAndPrint was enabled, Windows was vulnerable by design, Ben tweeted with the hashtag #printnightmare - Episode 3." And there is a four, and the four was the other problem. He said, under Episode 3, he said: "You know that even patched, with default config or security enforced with Microsoft settings, a standard user can load drivers as system. Local Privilege Escalation, #feature." Meaning this is what Microsoft intended.

Ben found a way - and actually he just highlighted some interaction of the systems that Microsoft has designed - which abuses Windows' normal method of installing printer drivers to gain local system privileges through malicious printer drivers. The technique can be used even if administrators have applied Microsoft's recommended mitigations of restricting printer driver installation to admins and disabling PointAndPrint, which is what they're now recommending. Though Ben's new local privilege escalation hack is not the same as the ones that we refer to as PrintNightmare, he feels that similar and related printer driver installation bugs ought to be grouped under the same name. He's explained that even with all mitigations applied, an attacker could create a signed malicious printer driver package and use it to achieve system privileges on other systems.

Okay. So how's this done? The attacker first creates a malicious code, a malicious printer driver, and signs it using any valid Authenticode certificate. That's not hard to do since anyone is able to obtain a code signing cert. So the bar there is very low. Once the attacker has a signed printer driver package, they're able to install the driver on any network device on which they've obtained admin privileges. This is also not actually a high bar since it can be comparatively easy to obtain admin on low-value systems.

The point is that, once this is done, due to the way Microsoft has designed the security governing printer driver installation, attackers can use this low-value system as a pivot to obtain system privileges on other high-value devices where they do not have elevated privileges, simply by causing those systems to install the now-trusted and installed locally, yet malicious printer driver.

If this sounds like a way for malicious actors to move laterally through an already compromised network, you're exactly right. And that's the example that Benjamin described. To prevent this style of attack, the printer spooler can be disabled or, bizarrely enough, PointAndPrint could be enabled with a policy to limit the servers from which a device can download printer drivers, thus preventing it from accepting your malicious, you know, the attacker's malicious printer driver which has been installed. But if PointAndPrint is enabled, then the mitigations created by Microsoft's most recent emergency patch can be bypassed through the vulnerability-by-design problem.

So we've got a Catch-22. And when Ben was asked how Microsoft could prevent this type of attack, he explained that they had attempted to prevent it in the past by deprecating version 3 printer drivers. But that caused so many problems that Microsoft backed off and terminated the version 3 deprecation policy four years ago, in June of 2017. In other words, they tried to tighten things down. But as we talked about before, you can't retroactively require signing of things that are already out on the Internet and installed in systems, and so you can't retroactively require signing where you didn't before.

Windows is designed to allow an administrator to install a printer driver, benign or malicious. And Windows is designed to allow non-admin users to install signed drivers onto their devices automagically. These two design choices interact to allow a signed malicious printer driver to be propagated across and throughout an enterprise's network. So if you're thinking that this whole Windows printer driver security design is a true mess, you're thinking correctly. Designed as it is, it cannot be secured.

Microsoft cannot and will not remove features which have been designed into Windows to allow it to work the way they want it to, the way their users have grown to expect it to and now depend upon it to. And so this is the case, even though Microsoft clearly knows fully well that those features open Windows to exploitation. Or, as Microsoft themselves phrased it in the bulletin for their most recent patch - actually now it's next to most recent - it is "vulnerable by design."

Leo: Well, that explains everything. In other words, we can't fix it because it would break it for other people.

Steve: Can't be fixed.

Leo: Can't be fixed. It's part of the way it works.

Steve: Yes, functionality that is the way they want it. It's like, well, yes. If somebody signed a malicious driver, and they were briefly somehow in admin on a low-value system, then they could install that malicious driver, and all other users could be induced to load that driver into their own systems. Whoops.

Yeah. Okay. Patch Tuesday's review. Last Tuesday was what started out as Microsoft's monthly patch day, that is, every Patch Tuesday, second Tuesday of the month. But it's gradually morphed into the industry's patch event. Aside from Microsoft, last Tuesday saw patches delivered from Adobe; Google with their Android product; Apache Tomcat; Cisco; Citrix; Juniper Networks; the SUSE, Oracle, and Red Hat Linux distributions; SAP; Schneider Electric; Siemens; and VMware. So yeah, I don't know if the other companies are looking for cover. So it's like, oh, yeah, maybe nobody will notice if Microsoft pumps out 117 security patches. Anyway, whatever.

Due to Microsoft's scope of influence, no one tops the importance of Microsoft's patches. nor their breathtaking number and severity. This past Tuesday they fixed, as I've said, a total of 117 security vulnerabilities, among which were 13 rated critical and nine, yes, nine, zero-day flaws, four of which are known to be currently employed by active attacks in the wild, potentially enabling an adversary to take control of affected systems. These 117 updates span Microsoft products including Windows, Bing, Dynamics, Exchange Server - they're still working on that one, have been all year; right? - Office, Windows Scripting Engine, Windows DNS, and Visual Studio Code.

And if you're thinking that 117 seems like a large number, you'd be right. We only had 50 the month before, in June, and 55 the month before that in May. So this is more than a double-whammy month for Microsoft. The four flaws known to be under active - they're zero-day flaws under active exploitation. There is one in Windows Print Spooler, which is allowing remote code execution. There are two Windows kernel elevation, which are allowing privilege elevations. And there's a scripting engine memory corruption vulnerability.

Microsoft noted that the last one, the scripting engine memory corruption vulnerability, had a very high attack complexity, explaining that attacks using it require luring an unsuspecting user to a malicious attacker-hosted website which contains a specially crafted file.

Leo: Why, it's virtually impossible.

Steve: How can that ever happen?

Leo: It could never happen.

Steve: Leo, who's going to do that? Yeah. But of course that's the way websites work. So doesn't seem like such a high bar. So I think maybe in the future, when we read that, oh, it's got some complexity to it, maybe we should now take that with a grain of salt. And since this is the one, it's one of the four that is under active exploitation, it sure does appear that this complexity, such as it is, hasn't created an insurmountable impediment for the attackers. Microsoft is apparently still working to clean up, as I noted, their Exchange Server product more than seven months after its problems first began to appear, with all the problems that we covered in the beginning of the year.

So two of the five publicly disclosed, but not currently exploited, as far as we know, that would be Microsoft Exchange Server remote code execution, so there's one that they fixed last week. And there's another elevation of privilege in Exchange Server. Both of those have been fixed last week. And finally, the last three, an Active Directory security bypass, Windows ADFS security bypass, and Windows certificate spoofing. Whoops. You don't want your certificates to be spoofed, so good thing that got fixed.

They also closed a security bypass vulnerability in Windows Hello biometrics-based authentication that permitted an adversary to spoof a target's face and get around the login screen. We don't know how. Maybe stick your tongue out, it just lets you in. They fixed a remote code execution vulnerability affecting Windows DNS Server that had a CVSS of 8.8, and one in the Windows kernel having a rare CVSS of 9.9. Wow. Whatever that was, I'm glad it's dead now.

So we had a sizable Microsoft Patch Tuesday that would have been much larger and bigger news normally, if it weren't, you know, if we weren't all still reeling from the recent PrintNightmares and just the shocks from the huge Kaseya REvil attacks. Oh, and I did want to remind everybody to update Acrobat and Reader if you're using Acrobat or Reader to read PDFs. Adobe also had a very big Patch Tuesday of their own. They resolved vulnerabilities in Dimension, Illustrator, FrameMaker, and as I said, Acrobat and Reader and their free Bridge media management product. I won't go into the details.

I will note that Acrobat and Reader had 14 critical and five important vulnerabilities fixed. Since most of those critical vulnerabilities can be leveraged into remote code execution, and since today's attackers, contemporary attackers, are quickly comparing previous

versions of fixed software to the released fixes to rapidly create exploits in order to exploit these systems before they're patched, it's important. And we also know that opening a PDF in email is a highly popular means for getting into people's systems, you know, sending phishing emails with a PDF. The way they leverage it is hoping that you're going to use a not-yet-updated version of Reader to view it. So definitely worth doing.

Okay. Under "Rolling Your Own Crypto," our longtime listeners will recall, because this has been a ways now, how skeptical I've always been of Telegram. The reason is that, right off the bat, I looked carefully at their cryptography. And the only term that comes to mind to accurately describe it would be the word "mess." Telegram's crypto is a godforsaken mess. I've never used it, and I never would. The fact that Telegram's crypto designers offered a large reward for anyone who could find a flaw in their homegrown mess says nothing about the quality of that mess. It only further demonstrates their misplaced confidence in the way they believe they've reversibly scrambled their users' plaintext.

My favorite example of the fundamental misunderstanding of security was literally onstage in the well-meaning form of Microsoft's Steve Ballmer, when he was prancing around during the launch of Windows XP, declaring it to be the most secure Windows ever. The trouble is, since something is secure only until it's not, and since it's not possible to prove a negative, it's not possible to make any factual statement about a product's security out of the gate. Something's security can only be demonstrated and proven over time. If something stands the test of time, and many attempts at its attack, only then can we begin to trust and believe in its security. Something's history is what matters, and a well-matured history is what we have with today's standard and standardized cryptographic security protocols. We know they are as safe as they've been proven to be.

And this is exactly why homegrown cryptography is, by definition, the dumbest thing anyone can do. Sure, if they had no alternative, if there were no other choice, if secure solutions didn't exist, then yeah, roll your own. Hold your breath and hope for the best because no other choice is available. But when Telegram was being designed, the world already had time-tested hacker- and academically-proven secure cryptographic protocol solutions. This was a solved problem, as much as it could be. We already had publicly and freely available ways to build proven bulletproof communication systems. This is why Telegram, when they rolled their own, and I looked at it closely, it just didn't make any sense to me. And now those chickens, as they say, have come home to roost.

An international team of computer scientists, cryptographers from ETH Zurich in Switzerland and the Royal Holloway College at the University of London, were released from their disclosure embargo last Friday to reveal that they had uncovered four cryptographic vulnerabilities in Telegram which could affect Telegram's half a billion users. That's right, 500 million users of Telegram. Hey, you know, Telegram looks great. What could possibly be wrong with it? Their full report will be presented at the prestigious 43rd IEEE Symposium on Security and Privacy next May. But I've included a link to their 52-page highly detailed paper in the show notes for anyone who wants more than I'm going to take the time to share today. They also offer a far more user-friendly page on GitHub that turns their many pages of dense math into English.

They start off summarizing their work by saying this. They wrote: "We performed a detailed security analysis of the encryption offered by the popular Telegram messaging platform. As a result of our analysis, we found several cryptographic weaknesses in the protocol, from technically trivial and easy to exploit, to more advanced and of theoretical interest.

"For most users, the immediate risk is low; but these vulnerabilities highlight that Telegram fell short of the cryptographic guarantees enjoyed by other widely deployed

cryptographic protocols such as TLS. We made several suggestions to the Telegram developers that enable providing formal assurances that rule out a large class of cryptographic attacks, similarly to other more established cryptographic protocols.

"Telegram uses its bespoke MTProto protocol to secure communications between clients and its servers as a replacement for the industry-standard Transport Layer Security (TLS) protocol. While Telegram is often referred to as an 'encrypted messenger,' this level of protection is the only protection offered by default. MTProto-based end-to-end encryption, which would protect communication from Telegram employees or anyone breaking into Telegram's servers, is only optional, and not available for group chats. We thus focused our efforts on analyzing whether Telegram's MTProto offers comparable privacy to surfing the web with HTTPS." In other words, they were comparing apples to apples.

So then they go on to explain that: "We disclosed the following vulnerabilities to the Telegram development team on April 16th, 2021, and agreed with them on a disclosure date of July 16th, 2021." In other words, last Friday. They then proceed to detail the four primary problems their analysis uncovered. And they are - it is deep math. But they said: "For example, one of the vulnerabilities is the so-called - they called it the "Crime Pizza" vulnerability.

Leo: Oh, yeah.

Steve: Yeah. You'll get this in a second. It allows for the arbitrary reordering of individual Telegram messages without detection. In their example, if the order of the messages in the sequence "I say 'yes' to 'pizza,'" "I say 'no' to 'crime'" were to be reordered, it would appear that the client is saying no to pizza and yes to their willingness to commit a crime. That may seem like a trivial problem; but if you think about it for a minute, there are likely ways that it could be exploited and abused.

But more to the point, it's never been possible to do that with our established protocols. It's one of the guarantees we take for granted that's provided by the other protocols we use today. And I'm sure that the fact that this should be prevented simply never occurred to the doubtless well-meaning developers of Telegram's protocol. And that is exactly the point, that it didn't occur to them. No developer can possibly take on the level of responsibility that's required for doing everything exactly right because there are so very many things that can go wrong.

By all means, roll your own crypto as a hobby. It's fun to scramble and then descramble some bits. Use it to chat among your friends. But don't put it into the hands of 500 million innocent users under the promise that it's unbreakable. It's not a promise that's practical to keep.

When master chefs are preparing food for others, they choose only the finest ingredients. When I was developing SQRL I similarly chose only the best-known and well-proven security primitives. And even so, I often stated that my various sphincters were tightly closed and that I hoped that I had not made any mistakes. Hope was all I had there, backed by the extreme care and testing that SQRL received. But at least I knew that I had used only the best ingredients. So Telegram did get its long-awaited analysis, and they learned something from the academicians who took the time to unscramble that wacky kitchen sink protocol that they had. And turns out, wow, yeah, looks like it scrambles stuff really well. But did you consider that it doesn't care what order things are in? Oh.

Leo: Whoops.

Steve: Whoops. Again, nothing against these guys. I mean, it is too hard to do this, like to consider every possible thing that can happen. So don't. Use one of the established systems that's already solved this problem.

Leo: Yeah, we never could understand why they wanted to roll their own.

Steve: No.

Leo: It just didn't make sense.

Steve: No, never did. Did not make sense. I think, you know, I've been accused, fairly, of having a strong case of NIH, you know, not invented here. Theirs was apparently even stronger.

Leo: Yeah.

Steve: Okay. So, Pegasus. The Israeli NSO Group produces and sells cyber surveillance spyware known as Pegasus. And Leo, this story has a fabulous moral you're going to love. After being surreptitiously installed onto targeted iPhones and Android devices, Pegasus enables its victim, or I guess, well, its client that has arranged to install it and is using it to capture emails, SMS messages, media, calendars, phone calls, contact information, and messaging chat content from messaging apps like WhatsApp, Telegram and Signal. And as if that wasn't enough, it's also able to stealthily activate the phone's microphone and camera. Because of course.

Leo: Why not? Why not?

Steve: Who knows what you're going to overhear? Just as a separate issue, Pegasus provides a classic example of the fact that it doesn't matter how good one's crypto is, I mean, because Signal was among those; right?

Leo: Right.

Steve: And we know Moxie Marlinspike nailed the crypto with Signal. It doesn't matter how good the crypto is if it's possible to simply capture the plaintext at either end of the encrypted tunnel. And note that even users of Apple's iPhone, with its much-heralded privacy protections and encrypted enclaves, fell victim to this pre-encryption and post-decryption shim. Okay. But back to Pegasus.

A data leak of more than 50,000 phone numbers catalyzed a collaborative investigation by more than 80 journalists from a consortium of 17 media organizations in 10 different countries. The investigation was coordinated by "Forbidden Stories," which is a Paris-based media nonprofit, and technical assistance was made available by Amnesty International. This investigation uncovered that Pegasus was being used, not only for the

surveillance of high-value targeted possible terrorists and other high-value criminals; but, sadly, and hardly surprising, heads of state, activists, journalists, lawyers, and businessmen around the world.

In response to the discovery of the extent to which Pegasus spyware was being abused, Amnesty International's Secretary-General was quoted, saying: "The Pegasus Project lays bare how NSO's spyware is a weapon of choice for repressive governments seeking to silence journalists, attack activists, and crush dissent, placing countless lives in peril. These revelations blow apart any claims by NSO that such attacks are rare and due to rogue use of their technology. While the company claims its spyware is only used for legitimate criminal and terror investigations, it's clear its technology facilitates systemic abuse. They paint a picture of legitimacy while profiting from widespread human rights violations."

Okay. So Pegasus is sold by the NSO Group to governments worldwide. It worms its way into its unwitting target's devices, either exploiting currently unknown security vulnerabilities in common apps, or by getting a potential target to click on a malicious link. The NSO Group describes itself as "the world leader in precision cyber intelligence solutions for the sole use of vetted-and-approved state-administered intelligence and law enforcement agencies solely for use in criminal and anti-terrorist investigations."

Okay, wait. Hold on a minute. That's exactly the group of entities and exactly their stated purpose behind their often-expressed need for having a responsible use backdoor added to the world's current mathematically secure encryption. Okay. Like we're going to trust this group of bureaucratic ne'er-do-wells with the key to anyone's backdoor? Okay. The list of infected phone numbers, which did not include their owners' names, so a lot of reverse lookup was being done, contains hundreds of business executives, religious figures, academics, NSO employees, union officials, and government officials operating in at least 11 countries, including Azerbaijan, Bahrain, Hungary, India, Kazakhstan, Mexico, Morocco, Rwanda, Saudi Arabia, Togo, and the UAE.

The timeline of the intrusions is spread over a seven-year period, from 2014 up to as recently as today; and the research has so far managed to identify 180 journalists and more than 600 politicians and government officials, despite their respective countries' adamant denials of having used Pegasus to hack the phones of the individuals named in the list.

Not surprisingly, the NSO Group flatly and loudly dispute all of the evidence and allegations. They state that the investigation is "full of wrong assumptions and uncorroborated theories that raise serious doubts about the reliability and interests of the sources," while they stress that they are on a "life-saving mission to break up pedophilia rings" - that's right, march out the kids - "sex and drug-trafficking rings, locate missing and kidnapped children, locate survivors trapped under collapsed buildings" - what?

Leo: What?

Steve: I know, "...and protect airspace against disruptive penetration by dangerous drones." Okay. Now, I read that through a couple times, and the only sense I can make of it is that some other of the NSO Group's products might be used for things like locating survivors trapped under collapsed buildings and ridding the airspace of illegal drone flyovers. I suspect that they may have been attempting to point to some of the good things their technologies can and have been used for. It's like we're not going to notice Pegasus.

And speaking of technologies and the Pegasus product, a forensic analysis of 67 mobile devices showed the intrusions involved the ongoing use of multiple zero-click exploits which do not rely upon any interaction from the device's user. And those both worked seven years ago, and they still work now. In one instance which was highlighted by Amnesty International, multiple zero-days were leveraged in iMessage to successfully penetrate a fully patched iPhone 12 running iOS 14.6 this month. So maybe they won't work today because it's 14.7. But come on. If 14.6 had multiple zero-days in iMessage, you've got to know that 14.7 has some, and they'll just ratchet forward their exploit chain.

In a series of tweets, Citizen Lab's Bill Marczak said: "All this indicates that NSO Group can break into the latest iPhones. It also indicates that Apple has a MAJOR" - all caps, his emphasis - "blinking red five-alarm fire problem with iMessage security that their so-called BlastDoor Framework, which was introduced in iOS 14" - which also apparently introduced the %@ remote code execution vulnerability, so BlastDoor blasted the doors off that. Anyway, "BlastDoor, which is supposed to make zero-click exploitation more difficult," Bill said, "is not successfully preventing those problems."

The Washington Post said in their in-depth report that, of the tested smartphones, 23 devices had been successfully infected with Pegasus, and 15 exhibited signs of attempted penetration. So, you know, we've seen other similar, smaller anecdotal examples of this sort of abuse. I really hope that this expos might help to strongly demonstrate why we as an industry must always be working as hard as we can to create the most absolutely secure devices and protocols possible, and that any deliberate weakening below the best we can possibly do would be foolhardy in the extreme. We just can't let the government say, oh, trust us, we're the government.

For anyone wanting more details, the Amnesty International report is amazing, and even further damning. It contains IP addresses, port numbers, the URLs of servers, the names of background Pegasus processes, and more. The link is in the show notes.

And Leo, we're going to take our last break. Then I'm going to entertain everybody with the bit of errata from last week and the vanishing act that REvil has performed.

Leo: Good.

Steve: So first, last week we drilled down into the Windows APIs that supported Windows on-the-fly PointAndPrint driver features. And in explaining the oddity of API function naming, there were several APIs ending in Ex. And I explained that this was a common occurrence for Microsoft, that the Ex is short for "extended" and is their way of amending an earlier non-extended, non-Ex API call, almost always by adding some additional parameters that time or advancing capabilities has shown were needed. And I made the offhand remark that there were no ExExes, that is, extended extensions. Well, I should have known better.

Leo: Of course there are.

Steve: Of course there are.

Leo: Of course there had to be.

Steve: I was quickly called out on that in GRC's Security Now! newsgroup by someone who did know better. Turns out that there's an original LogonUser API with that name, and of course an extended LogonUserEx API, and an even further extended LogonUserExEx API.

Leo: Of course there is.

Steve: And Microsoft, if at first you don't succeed, yeah, just force us all to upgrade. Anyway, a tip of the hat to Greg Bell for paying attention and helping me to keep my facts straight.

Okay. July has been REvil month for this podcast. We kicked off the month on the 6th with The Kaseya Saga. And then because the crypto underlying REvil's Sodinokibi malware appeared to be uniquely powerful and well designed, we took it completely apart last week with our podcast REvil's Clever Crypto. So it's fitting, I think, that we wrap up this third-in-a-row podcast focused upon REvil's own apparent wrap-up, which occurred suddenly and without apparent warning or notice, not that there would have been any way for them to give us any, early last week. But I guess they could have put up a sign saying goodbye. They didn't do that. As we were laying out the details last week of REvil's cryptographic architecture, the REvil gang was packing their virtual bags.

The first anyone outside of the REvil organization knew of this was when at, interestingly, 8:00 a.m. Moscow time, all of REvil's online infrastructure disappeared at once. Attempts to access their onion-routed Tor site returned the message "Onionsite Not Found," with the detailed error code 0xF0, so hex F0, which is "The requested onion service descriptor can't be found on the hashring, and therefore the service is not reachable by the client."

Being the Internet, sites sometimes come and go as infrastructure is changed and updated. We see that from time to time. I try to keep GRC's forums, the web forums and the server up. But you've got to do your updates every so often. So, yeah. You can get little glitches. And of course onion sites are no exception. The Tor Project's Al Smith, who manages communications and fundraising for Tor, told BleepingComputer's Lawrence Abrams, you know, BleepingComputer's founder, that receiving this error generally means that the onion site is offline or disabled, but that to know for sure what it means you'd need to contact the onion site's administrator.

But it wasn't just the Tor site that disappeared at 8:00 a.m. Moscow time. All of REvil's infrastructure shut down and went offline simultaneously. REvil's regular public Internet-facing side, now, that's the non-Tor, and I saw the term being used "clearsite," which I thought was cool, that's like plaintext, cleartext, clearsite. That's "decoder.re." It also disappeared at the same time. And the official MalwareHunterTeam Twitter account later tweeted the next day: "REvil's clearweb payment site decoder.re was already down eight to nine hours ago" - and he was tweeting eight or nine hours later - "with not only the server down, or no A record, no DNS response at all."

And in reply, Jaime Blasco, who's with AT&T's Alien Labs Cybersecurity group tweeted: "No DNS records, but previous A record server" - he had kept a record of it, and that was 82.146.34.4 - "is still up." And then he said: "(Only SSH open)." And then he said: "And likely actor controller name server ns1.goprodns.top also up, only SSH." Okay, so the point there is this means that no one tripped over a cord somewhere, right, and like caused a power failure and no one noticed it. If you go up a few levels, the servers that were previously supplying the data are themselves still online. But the services that they were previously offering have been terminated.

Now, recall that XSS was that Russian language-speaking hacking forum that had previously changed its policies, deciding to stop hosting malware forums and discussion after that mess that DarkSide made with its high-publicity attack on - oh, actually it's one of DarkSide's affiliates attacked Colonial Pipeline and brought down the whole Eastern seaboard of the U.S. energy oil petroleum infrastructure. Okay. Well, later last Tuesday a representative from the LockBit ransomware gang posted to that XSS forum that it was rumored the REvil gang erased their servers after learning of a government subpoena.

BleepingComputer obtained an English translation of the Russian posting which read: "Upon uncorroborated information, REvil server infrastructure received a government legal request forcing REvil to completely erase server infrastructure and disappear. However, it is not confirmed." And then, shortly after that, the XSS forum's administrator banned REvil's public-facing representative, who was known under the name "Unknown" from the forum. Attempting to look up that forum member shows "banned" in the forum software.

I've been watching, as I imagine many of us have, the saber-rattling that U.S. President Biden has been doing relative to these apparently Russia-based, and at least tacitly allowed, ransomware cyberattacks against the West. We do know that it's true that, due to Biden's lifelong participation in U.S. national politics, and his eight-year stint as Obama's VP, that he actually does have a working relationship with Russia's President Vladimir Putin. So it may well be that Biden's reported soft ultimatums, which have been getting a little less soft recently, that if Russia doesn't do something about this internally, the U.S. would take some actions ourselves, or themselves, has been effective.

In full display in front of the press, following the signing of an executive order at the White House recently, Biden said: "I made it very clear to him" - meaning Putin - "that the United States expects when a ransomware operation is coming from his soil, even though it's not sponsored by the state, we expect them to act if we give them enough information to act on who that is."

So although we don't have definitive proof that REvil is gone, we're now "disappearance plus one week," and REvil has not returned. So this was clearly deliberate. An NSLOOKUP of their public-facing decoder.re still returns an NXDOMAIN error no DNS resolution for a domain of that name. While we'll likely never know what triggered REvil's sudden departure from the ransomware scene, more to the point is what happens next.

We've seen ransomware groups like Babuk and DarkSide shut themselves down, more or less voluntarily, due to increased scrutiny and pressure from law enforcement. DarkSide really, as we know, stepped in it when one of their affiliates took down Colonial Pipeline's operation. And we're now seeing more "socially responsible," if you can believe that, choosing of attack victims, for exactly that reason. Attacking infrastructure of any kind energy, healthcare, or education tends to rouse the bear.

So what we're seeing is that fame for a ransomware group is a double-edged sword. Under today's evolving ransomware affiliation model, a group needs sufficient reputation to be able to attract the best and most capable affiliates. But at the same time, to any degree possible, they also want to remain as far under the radar of their hosting country's law enforcement as possible; and, by extension, under the radar of the world.

After the Babuk ransomware gang shut down and disbanded over disagreements about how their attacks were being conducted, a contingent of that group later relaunched under Babuk v2.0. And remember that REvil themselves is already in its second incarnation. Many of its group members were part of the earlier GandCrab ransomware group which was shut down, only to later be reborn as REvil.

Just as the Colonial Pipeline attack was too much and forced the shutdown of DarkSide, the massive ransomware disasters that were enabled first by the attack and shutdown of that meatpacker JBS Foods, and then by the Kaseya server breaches, made REvil a household name overnight. And that's not what any ransomware operation wants to be. They want and need to operate in the shadows, hidden by Tor and by Bitcoin and by a layer of intermediate affiliations.

Given the maturity of the GandCrab/REvil malware platform, Sodinokibi, and the amount of money that can be extorted through the ransomware model, I won't be surprised if this group doesn't take away a few lessons from the DarkSide's Colonial Pipeline, JBS Foods, and Kaseya overachievements to somehow arrange to throttle future attacks so that they can remain diffuse and effective, while also remaining well beneath any one government's radar. I expect they're not gone for good.

Leo: Yeah. I think this is a rebranding, how about.

Steve: Yup. Yup.

Leo: It's too much money to leave on the table.

Steve: It is.

Leo: But obviously they're scared of the whole thing, and they don't want to go to Russian jail. So they're going to play it low and show up somewhere else, I think.

Steve: A site that we should tell you about, Leo, because I told everybody last week, it's a week old, or not long old, called Ransomwhere. You might want to bring it up. It's .re. So Ransomwhe.re, Ransomwhere. Actually it's a new employee of the firm that we talked about formed by Alex Stamos and the other - oh, and Chris Krebs, not Brian Krebs, Chris Krebs.

Leo: Yeah, yeah, yeah, Chris Krebs, yeah, yeah.

Steve: Anyway, one of their new employees put up this site. And what's interesting is that it shows their - oh, in fact it was at 60 million last week. Now it's at 92.

Leo: So none this week. Last month - this is tracked ransomware payments. And I presume they're just tracking bitcoin accounts and that kind of thing. But all time is $92.5 million.

Steve: I believe it was at 60 last week. So what they're doing, if you scroll down, they have basically a database of payments. Oh, okay. Now the bar is completely different. That first one was NetWalker.

Leo: Now it's Conti, yeah.

Steve: Conti's number one?

Leo: And the REvil Sodinokibi.

Steve: Ah, okay. I think they had...

Leo: So maybe they're getting more data, probably.

Steve: Yes. Yeah. They had a ransomware in the first place that was very suspicious to me. So I'll bet it was a bad record that they fixed.

Leo: That's probably it, yeah. This is all generated...

Steve: It makes much more sense that Conti would be in number one and REvil in number two.

Leo: Yeah, yeah, yeah.

Steve: Anyway, so anyway, basically it is a ransomware tracking site, tracking payments and bitcoin addresses. And people are able to submit events that they're aware of in the hopes that that this will create a public clearinghouse of ransomware events. So anyway, the thing that brought that site to mind is the scope of the dollars, Leo. As you were saying, it just, unfortunately, it makes too much money.

Leo: Right, right. It's just too much money. It's funny because Lisa and I were watching a movie, a relatively recent movie about armored car robberies. And I realized you don't hear a lot about that anymore because, what, you're going to risk your life for a couple of million when you can completely anonymously make 20, 30 million just like that with ransomware? All the smart crooks, anyway, are going cyber. Why carry a gun? You know, that's dangerous.

Steve: We've just started to watch a series called "StartUp."

Leo: Oh, it's good; isn't it?

Steve: Yeah.

Leo: Yeah, we just finished it, yeah.

Steve: And we just started. It's 7.9 on IMDB, so that clears the bar for me. And yeah, we just started. We're two episodes in. And it made me think of that because we're talking about crooks and cryptocurrency.

Leo: Yeah. You'll laugh at some of the tech boners. But you'll be very interested, I think, in what happens at the end of Season 2.

Steve: Oh, cool.

Leo: That's all I'm going to say.

Steve: Ah.

Leo: But it rings a bell.

Steve: Nice.

Leo: Good, good. Well, I'm sorry to say this concludes this portion of Security Now!. But you know you can get your daily or your weekly dose, just tune in every Tuesday around 1:30 Pacific, 4:30 Eastern, 20:30 UTC, and Steve and I will be here gassing away about the latest security news. You can watch us make the show live at TWiT.tv/live, actually watch or listen. There's audio and video there. And if you're watching live, by the way, chat live: irc.twit.tv. After the fact you can of course always download a copy of the show.

Steve's got 16Kb versions for the bandwidth impaired, as well as the normal 64Kb MP3s, at GRC.com. He also has very nicely done human-written transcripts, if you like to read along while you listen, or you want to use that for search. That's a really good way to search for the part of the show you're looking for. All of that's at GRC.com, along with SpinRite, the world's finest mass storage maintenance and recovery utility, GRC.com.

Steve: Wow.

Leo: He's commending me because I remembered to say "mass storage" instead of hard drive because it works on all kinds of mass storage now. You also can find the show at our website, TWiT.tv/sn. It's on YouTube. There's a whole YouTube channel for Security Now!. And of course, if you use a podcast client, simple enough, search for TWiT or search for Security Now!, you'll find the show. You can subscribe, get it automatically the minute it's available of a Tuesday afternoon. You can also, if you would, leave us a five-star review, help the next generation of security wonks discover the best show on security on the Internet.

Thank you for being here, everybody. Thank you, Steve Gibson, for the hard work you do. And we'll see you next week. I'll be dressed a little more formally for Security Now!. Bye-bye.

Steve: Right-o. Bye.