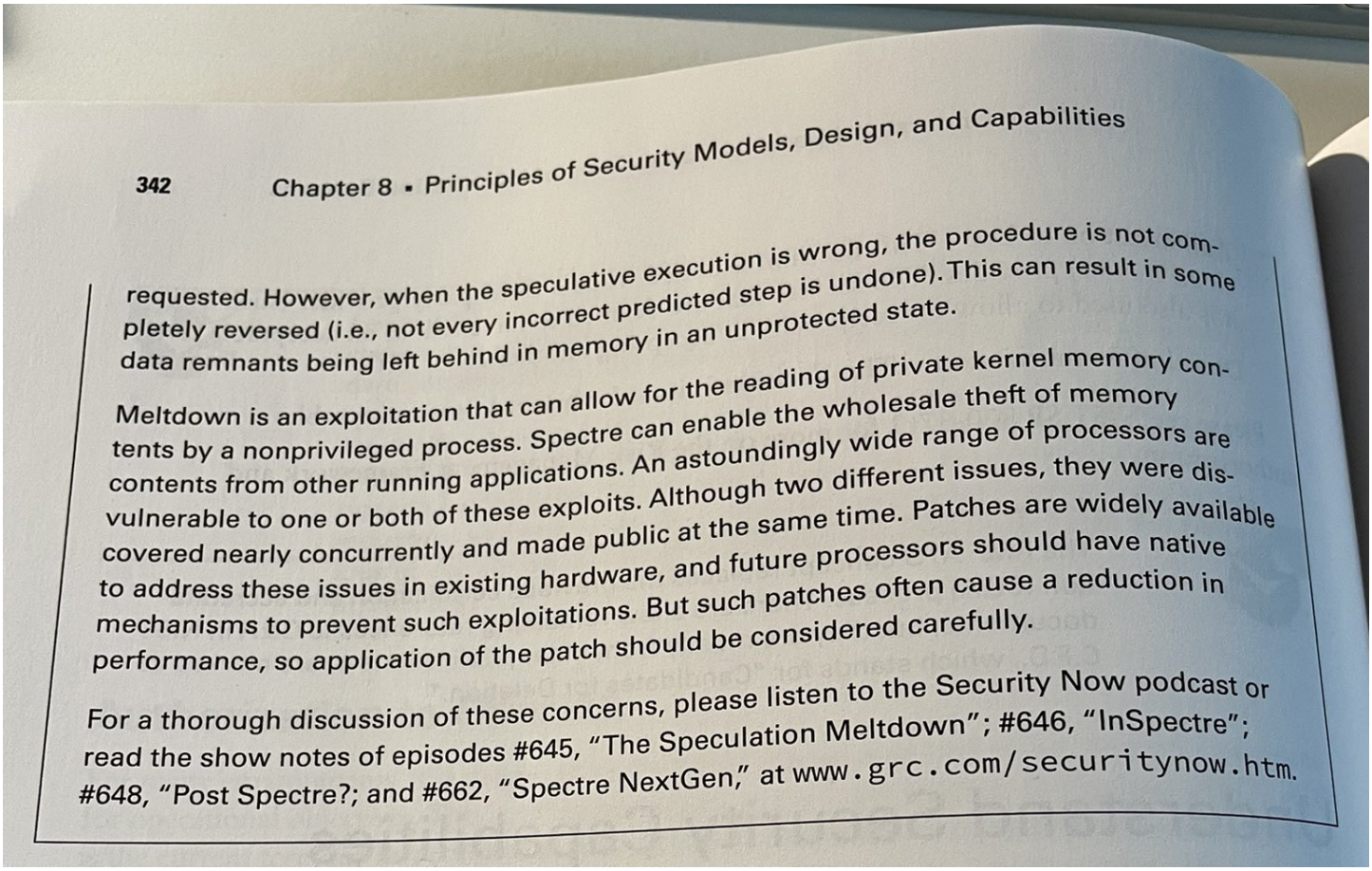
Security Now! #828 - 07-20-21

REvil Vanishes!

This week on Security Now!

This week we look at the continuing attacks on Chrome with yet another 0-day and on Mozilla's continuing work to give their users the most privacy possible. We re-examine 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. And 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.



342 Chapter 8 ▪ Principles of Security Models, Design, and Capabilities

requested. 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.

Meltdown is an exploitation that can allow for the reading of private kernel memory contents by a nonprivileged process. Spectre can enable the wholesale theft of memory contents from other running applications. An astoundingly wide range of processors are vulnerable to one or both of these exploits. Although two different issues, they were discovered nearly concurrently and made public at the same time. Patches are widely available to address these issues in existing hardware, and future processors should have native mechanisms to prevent such exploitations. But such patches often cause a reduction in performance, so application of the patch should be considered carefully.

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?; and #662, "Spectre NextGen," at www.grc.com/securitynow.htm.

From Chapter 8 of the official CISSP "Certified Information Systems Security Professional" certification study guide.

(Thanks to Chuck Littlefield @Chucklittlefiel for sharing it.)

Browser News

The attacks on Chrome continue.

Google has released Chrome 91.0.4472.164 for Windows, Mac, and Linux fixing seven security vulnerabilities, one of them a high severity 0-day vulnerability being actively exploited in the wild. Of CVE-2021-30563 Google said that it's aware of reports that an exploit exists in the wild.

As usual, Chrome will apparently eventually auto-update. But when I checked, I was still running the previous **.124** version rather than this most recent **.164** release. The act of checking with Setting / Help / "About Google Chrome" triggered its update and after a restart I was current.

And we are keeping score here. CVE-2021-30563 brings the total count of exploited in-the-wild critical 0-day flaws patched so far this year to 8. It's been a rough year so far, all around. This one was another type confusion bug in Chrome's V8 engine, their high-performance WebAssembly and JavaScript processing system.

One little tidbit was particularly interesting. Google stated that based upon their analysis (of what they did not say) two of those eight 0-days, 21166 and 30551, had been developed and sold by the same vendor providing surveillance capabilities to customers around the world. Then last Thursday, Microsoft and Citizen Lab linked the vendor mentioned by Google Threat Analysis Group report to Israeli spyware vendor Candiru. It is 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.

Firefox special-cases anti-tracking for "Login With" functions

When Firefox's full anti-tracking protections are enabled under Firefox's strongest privacy protecting incognito browsing mode, the increasingly popular "login with" features, which is accomplished with scripts that can inherently be used for tracking, will not function. And that was causing trouble. So the just-released Firefox 90 resolves this dilemma.

Here's what Firefox said, naturally with a bit of a sales pitch...

> *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 in to websites, while providing strong defenses against cross-site tracking.*
>
> *Logging into websites 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: that's because our partner Disconnect includes Facebook domains on their list of known trackers. 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. If you click on the Enhanced Tracking Protection shield in the address bar, 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 Firefox 90, 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 the third-party Facebook script required had been blocked by Firefox.

Now, SmartBlock 2.0 in Firefox 90 eliminates this login problem. 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 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.

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. Thanks to Firefox SmartBlock, you can have your cake and eat it too!

It's obvious to anyone why "Sign-in with Google or Facebook" are compelling offers to the typical user. They have no way of appreciating that Google and Facebook gleefully offer these services because the 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 ID. And not to mention that the surrogate also knows where they have just logged on!

So, I think it's very cool that, in the first place, Firefox's growing privacy protections were 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 exception windows to allow the indirect logon flow to succeed. So 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 vs privacy tradeoffs.

Security News

iOS WiFi SSID bug

I think it's worth reinforcing the crucial security principle Bruce Schneier captured when he wrote that: "Attacks always get better, they never get worse."

Recall that Apple iOS WiFi SSID bug from last month. Security researcher Carl Schou tweeted: "After joining my personal WiFi with the SSID "%p%s%s%s%s%n", my iPhone permanently disabled it's WiFi functionality. Neither rebooting nor changing SSID fixes it :~)"

We talked about the inherent danger of this incredibly convenient shortcut that exists in many programming languages. One of this podcast's other observations is the danger inherent in interpreters. And what we have here with the %-character escape is an interpreter where the printf function is reading and interpreting the string. Since no WiFi radio's SSID should contain interpretable %-sign escape sequences, the bug that was discovered in iOS was that SSID's — which are, in this case, attacker controlled — were not being "sanitized" by first doubling up all "%" characters in into a "%%" pair, which would be treated as a single literal '%' without and special interpretation.

The reason we're talking about this, and reminding of Bruce Schneier's pithy observation, is that, yes indeed, that flaw was weaponizable. After studying the trouble and verifying that it's much worse than we were told, security researchers with ZecOps have nicknamed the issue "WiFiDemon." It's 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 Wi-Fi networks. Even if they are not joined, just the act of sniffing the maliciously crafted WiFi SSID beacon is all that's required.

I used the phrase "much worse than we were told" because Apple was apparently aware of this — and elected not to tell anyone, even after the fact. The flaw was introduced with the release of iOS 14.0 last September, and Apple quietly patched the issue in January this year as part of their iOS 14.4 update. They never made any mention of it, nor did they bother assigning a CVE identifier to the flaw.

And, as we know, the iOS 14.4 update didn't fully fix all of the problems, since Carl Schou's discovery of the whacky SSID is still workable even now, under iOS 14.6. It has finally been fixed in the next iOS 14.7 which is undergoing final pre-release beta testing.

How do we know that Apple knew, and that this dangerous WiFi SSID remote code execution vulnerability didn't just coincidentally disappear on its own? We know this because of what Apple quietly removed to fix the trouble. We've talked about the inherent trouble with %-escapes. Well, until iOS 14.4 another incredibly dangerous escape sequence was being honored: %@. In Objectice-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 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 execution of code.

But, 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 only patching to fix the provided proof-of-concept demonstration of the flaw. Here's my concern: 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 %-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 rather than looking at the forest and thinking “hold on a second... why exactly are we interpreting any %-escape sequences here?”

I don’t know. But, if this is indicative of a larger emerging trend in our industry... then... this podcast is going to require six digits for its episode numbering!

We still can't awaken from the "PrintNightmare:...

We began last week's podcast by observing that, as I mentioned above, the 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 on all this has been a bit more challenging than usual since I’ve needed to make sure that I’m not re-reporting something that we’ve talked about before. There have been so many similar and related problems with discoveries and announcements and patches. I’m seeing other researchers tweeting that it’s becoming difficult to keep up. But 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 or kernel) level.

For the first of the two Microsoft has assigned the CVE 2021 34481 and it’s been given a CVSS severity score of 7.8. Microsoft writes:

https://msrc.microsoft.com/update-guide/vulnerability/CVE-2021-34481

> *An elevation of privilege vulnerability exists when the Windows Print Spooler service improperly performs privileged file operations. 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.*

In other words, we’re back to disabling the 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 elevated privileges on a device.

In their bulletin’s FAQ they ask and answer:

> Is this vulnerability related to the previously addressed CVE-2021-1675 and CVE-2021-34527 vulnerabilities?
>
> This distinct vulnerability also exists in the Print Spooler service. However, the security impact is local elevation of privilege.
>
> Did the July 2021 security update introduce this vulnerability?
>
> No, the vulnerability existed before the July 13, 2021 security update. We recommend that Microsoft customers install the latest security updates.

Okay. So there's one. They say that it's been publicly exposed, but not whether it is currently being exploited in the wild. They do, however, say that the attack complexity is low, the privilege required is low and no user interaction is required. So it's unclear whether we're going to be seeing another emergency patch. This one feels a bit less dire since it's **not** an RCE that's known to be under active exploitation. And the only remediation Microsoft is offering is to stop the spooler service which we now know kills printing.

The second issue that I wanted to bring to everyone's attention is another consequence of Microsoft's deliberate system-level design. In other words...

It's not a bug, it's a feature!...
This 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 in passing. He originally created Mimikatz as a proof of concept as a demonstration to Microsoft that their authentication protocols were vulnerable to attack. But in so doing, he also created 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 Delpy has some Windows hacking cred.

In his tweet last Wednesday, after this months patches had landed and Microsoft had explained to the world that when Windows PointAndPrint was enabled Windows was vulnerable by design, Ban tweeted with the hashtag #printnightmare - episode 3:

> *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*

Ben found a way to abuse Windows' normal method of installing printer drivers to gain local SYSTEM privileges through malicious printer drivers. This technique can be used even if administrators have applied Microsoft's recommended mitigations of restricting printer driver installation to admins and disabling Point and Print.

Though Ben's new local privilege escalation hack is not the same as the ones we refer to as PrintNightmare, he feels that similar and related printer driver installation bugs ought to be grouped under the same name. Ben has explained that even with all mitigations applied, an attacker could create a signed malicious print driver package and use it to achieve SYSTEM privileges on other systems.

To do this, the attacker would create a malicious print driver and sign it using any valid Authenticode certificate. That's not difficult, since anyone is able to obtain a code signing certificate. The bar there is very low. Once the attacker has a signed printer driver package, they're able to install the driver on any networked device on which they've obtained admin privileges. This is also not 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" device to obtain SYSTEM privileges on other devices where they do NOT have elevated privileges simply by causing those systems to install the now-trusted yet malicious 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 Benjamin Delpy described. To prevent this style of attack, the print spooler can be disabled or, bizarrely enough, Point And Print could be enabled with a policy to limit the servers from which a device can download print drivers. But if Point And Print is enabled, then the mitigations created by Microsoft's most recent emergency patch could be bypassed... by design.

If this sounds like a Catch-22, you're right. 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 this caused so many problems that Microsoft terminated the version 3 deprecation policy four years ago, in June of 2017.

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. These two design choices allow a signed malicious printer driver to be propagated across and throughout an enterprise's network.

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 this is so 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, it is "Vulnerable by Design."

Patch Tuesday Review

Last Tuesday was what started out as Microsoft's monthly patch day but has gradually morphed into the industry's patch event. Aside from Microsoft, last Tuesday saw patches delivered from Adobe, Google/Android, Apache Tomcat, Cisco, Citrix, Juniper Networks, the SUSE, Oracle, and Red Hat Linux distributions, SAP, Schneider Electric, Siemens, and VMware.

However, due to their scope of influence, no one tops the importance of Microsoft's patches — nor in their breathtaking number and severity. This past Tuesday they fixed a total of 117 security vulnerabilities, among which were 13 rated CRITICAL and 9—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's products, including Windows, Bing, Dynamics, Exchange Server, 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.

The four flaws known to be under active exploitation are:

- CVE-2021-34527 (CVSS score: 8.8) - Windows Print Spooler Remote Code Execution Vulnerability (publicly disclosed as "PrintNightmare")
- CVE-2021-31979 (CVSS score: 7.8) - Windows Kernel Elevation of Privilege Vulnerability
- CVE-2021-33771 (CVSS score: 7.8) - Windows Kernel Elevation of Privilege Vulnerability
- CVE-2021-34448 (CVSS score: 6.8) - Scripting Engine Memory Corruption Vulnerability

Microsoft noted that that 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 that's engineered to trigger the vulnerability. But... uh... that's the way websites work. And since this is one of the four that's under active exploitation, it sure appears that this "complexity"—such as it is—hasn't created an insurmountable impediment for attackers. Microsoft is apparently still working to clean up their Exchange Server product more than seven months after it problems first began to appear. So two of the five publicly disclosed by not exploited vulnerabilities are:

- CVE-2021-34473 (CVSS score: 9.1) - Microsoft Exchange Server Remote Code Execution Vulnerability
- CVE-2021-34523 (CVSS score: 9.0) - Microsoft Exchange Server Elevation of Privilege Vulnerability

And also:
- CVE-2021-33781 (CVSS score: 8.1) - Active Directory Security Feature Bypass Vulnerability
- CVE-2021-33779 (CVSS score: 8.1) - Windows ADFS Security Feature Bypass Vulnerability
- CVE-2021-34492 (CVSS score: 8.1) - Windows Certificate Spoofing Vulnerability

Microsoft 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.

They fixed a remote code execution vulnerability affecting Windows DNS Server with a CVSS score of 8.8, and one in the Windows Kernel having a rare CVSS of 9.9. Yikes! Whatever that one was, I'm glad it's dead now.

So, we had a sizable Microsoft patch Tuesday that would have been much much bigger news if we weren't still reeling from the recent Print Nightmares and the huge Kaseya REvil attacks.

Update Acrobat and Reader

It's also worth mentioning the Adobe also had a big Patch Tuesday of their own which resolved vulnerabilities in Adobe Dimension, Illustrator, Framemaker, Acrobat, Reader, and Bridge.

The complete list of Adobe Products receiving security updates today and the number of fixed vulnerabilities are below:

- APSB21-40 | Adobe Dimension: 1 Critical vulnerability fixed.
- APSB21-42 | Adobe Illustrator: 2 Critical and 1 Important vulnerability fixed.
- APSB21-45 | Adobe Framemaker: 1 Critical vulnerability fixed.
- APSB21-53 | Adobe Bridge: 4 Critical and one Moderate vulnerabilities fixed.

And last but not least, by far Adobe's most widely used products Acrobat and Reader had...

- APSB21-51 | Adobe Acrobat and Reader: 14 Critical and 5 Important vulnerabilities fixed.

Since most of those CRITICAL vulnerabilities can be leveraged into remote code execution and since today's attackers are quickly comparing previous versions to the fixed releases to rapidly create exploits for patched but not yet corrected problems, everyone using Adobe's Acrobat and

Reader products would be well advised to go to Help and Check for Updates to be certain you're now running the latest.

Rolling your own Crypto

Our longtime listeners will recall 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 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 user's plaintext.

My favorite example of the fundamental misunderstanding of security was literally on stage in the well-meaning form of Microsoft's Steve Ballmer when he was pranced 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 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 we have no alternative; if there were no other choice; 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 those 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, when Telegram rolled their own, I looked at it closely and was repelled. And now those chickens have, as they say, come home to roost.

An international team of computer scientists, Cryptographers from ETH Zurich in Switzerland and the Royal Holloway college of 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, it 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: https://mtpsym.github.io/paper.pdf

They also offer a FAR more user-friendly page on GitHub that turns their many pages of dense math into English: https://mtpsym.github.io/

They start off by summarizing:

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 communication 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 analysing whether Telegram's MTProto offers comparable privacy to surfing the web with HTTPS.

They then 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.

For example, one of the vulnerabilities they termed the "Crime-Pizza" vulnerability. 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 protocols we use. 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: 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 quite tightly closed and that I hoped

that I hadn't 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.

Pegasus

The Israeli "NSO Group" produces and sells cyber-surveillance spyware known as "Pegasus." After being surreptitiously installed onto targeted iPhones and Android devices, it enables Pegasus' user to capture eMails, SMS messages, media, calendars, calls, contact information, and massaging 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.

Just as a separate issue, Pegasus provides a classic example of the fact that it doesn't how good one's crypto is if it's possible to simply capture the plaintext at either end of the encrypted tunnel. 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. 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 countries. The investigation was coordinated by "Forbidden Stories", a Paris-based media non-profit, 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, but (sadly, hardly surprising) heads of state, activists, journalists, and lawyers around the world.

In response to the discovery of the extent to which the 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."

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 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."

Whoa! Hold on! Wait a minute!! Isn't that **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?? Yeah, right... like we're going to trust **this** group of bureaucratic ne'er-do-wells with a key to anyone's backdoor!

The list of "infected" phone numbers, which did not include their owners' names, contains hundreds of business executives, religious figures, academics, NGO 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 U.A.E.

The timeline of the intrusions is spread over a 7-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 country's adamant denials of having used Pegasus to hack the phones of the individuals named in the list.

Not surprisingly, the NGO Group flatly and loudly disputed all of the evidence and allegations. They stating that the investigation is "full of wrong assumptions and uncorroborated theories that raise serious doubts about the reliability and interests of the sources," while stressing that it's on a "life-saving mission" to "break up pedophilia rings [that's right, march out the children], sex and drug-trafficking rings, locate missing and kidnapped children, locate survivors trapped under collapsed buildings [what?!], and protect airspace against disruptive penetration by dangerous drones." I read that through a couple of times, and the only sense I can make of it is that some other of the NGO 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 good things their technologies can and have been used for.

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 today. In one instance which was highlighted by Amnesty International, multiple 0-days were leveraged in iMessage to successfully penetrate a fully patched iPhone 12 running iOS 14.6 this month.

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 blinking red five-alarm-fire problem with iMessage security that their BlastDoor Framework, which was introduced in iOS 14 to make 0-click exploitation more difficult, is not successfully preventing."

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.

We've seen other, similar, smaller anecdotal examples of this sort of abuse. I really hope that this expose' 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 possible, and that any deliberate weaking below the best we can possibly do would be foolhardy in the extreme.

For anyone wanting more details, the Amnesty International report is amazing and 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:

https://www.amnesty.org/en/latest/research/2021/07/forensic-methodology-report-how-to-catch-nso-groups-pegasus/

Errata

Last week we drilled down into the Windows APIs that supported Windows' on-the-fly "Point And Print" driver features. And, in explaining the oddity of API functions ending in "Ex" 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-Ex" API call, almost always by adding some additional parameters that time or advancing capabilities had shown were needed. And I made the offhand remark that there were no "ExEx" APIs.

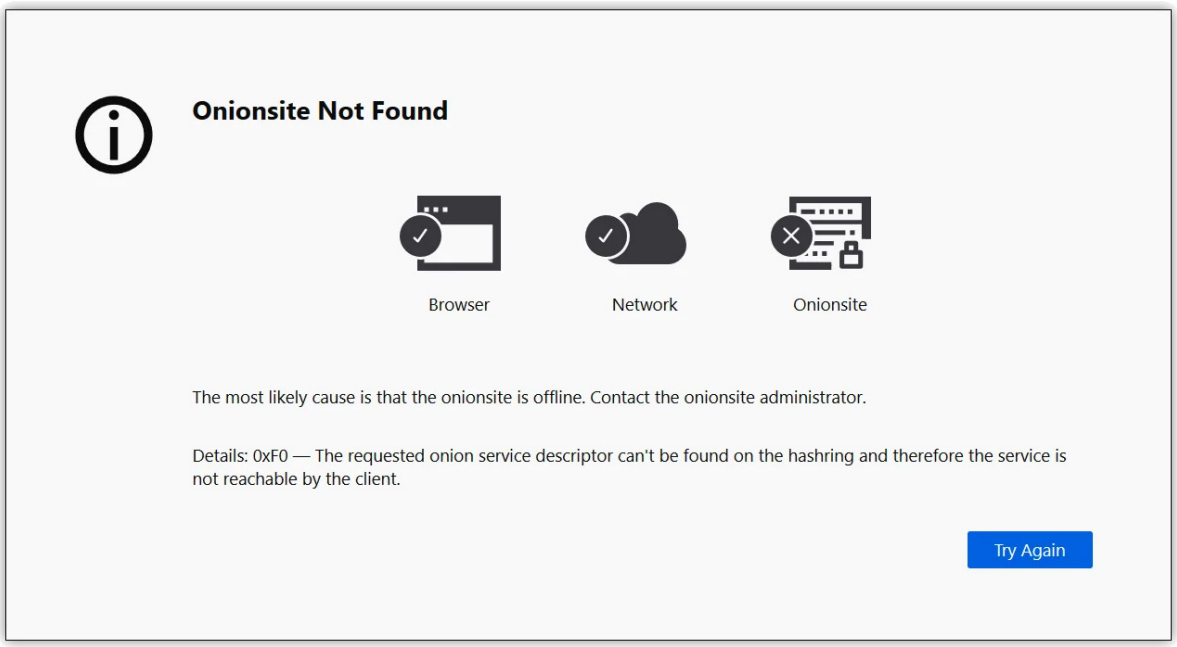
Well, I should have known better. I was quickly called out on that in GRC's Security Now! newsgroup by someone who did know better. It turns out there's an original "LogonUser" API. And an extended "LogonUserEx" API. And an even more extended "LogonUserExEx" API.

So... a tip of the hat to Greg Bell for paying attention and helping me to keep my facts straight!

REvil Vanishes!

June has been REvil month for this podcast. We kicked off the month, on July 6th, with "The Kaseya Saga." 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 episode "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 early last week. As we were laying out the details of REvil's cryptographic architecture, the REvil gang was packing their virtual bags.

The first anyone outside of the organization knew of this was when, at 8AM Moscow time, all of REvil's online infrastructure disappeared. Attempts to access their onion-routed TOR site return the message "Onionsite Not Found"...



With the detailed error code 0xF0 — "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 or updated. And Onion sites are no exception. The TOR Project's Al Smith, who manages communications and fundraising for the Project, told BleepingComputer's Lawrence Abrams 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 administrator.

But it wasn't just the TOR site that disappeared at 8AM Moscow time. ALL of REvil's infrastructure shut down and went offline simultaneously.
REvil's regular public Internet non-TOR clearsite "decoder.re" also disappeared at the same time and the official MalwareHunterTeam Twitter account (@malwrhunterteam) later tweeted the next day: "REvil's clearweb payment site decoder[.]re was already down 8-9 hours ago, 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 (82.146.34.4) is still up (only SSH open). And likely actor controller nameserver (ns1.goprodns[.]top) also up (only SSH)."

The point is... this means that no one tripped over a cord somewhere. Go up a few levels and the servers that were previously supplying the data are, themselves, still online. But the services they were previously offering have been completely terminated.

Recall that "XSS" Russian language hacking forum that had previously changed its policies, deciding to stop hosting ransomware after the mess that DarkSide made with its high-publicity attack on Colonial Pipeline? Well, later last Tuesday a representative from the "LockBit" ransomware gang posted to that that "XSS" forums 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 known under the handle "Unknown" from the forum.

I've been watching, as I imagine many of us have, the saber rattling that US 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 US national politics, and his eight-year stint as Barack 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, that if Russia doesn't do something about this internally the US would take some action themselves, has been effective. In full display in front of the press, following the signing of an executive order at the White House, 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."*

Although we don't have definitive proof that REvil is gone, we're now "disappearance plus one week" and REvil has not returned. 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 other ransomware groups like Babuk and DarkSide shut themselves down, more or less voluntarily, due to increased scrutiny and pressure from law enforcement. Darkside really 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 the reason that attacking infrastructure of any kind — energy, health care 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 affiliate 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.

After the Babuk ransomware gang shutdown and disbanded over disagreements about how their attacks were being conducted, a contingent of that group later relaunched as Babuk v2.0. And remember that "REvil" itself was already a second incarnation. Many of its group members were part of the earlier GandCrab ransomware group which was shutdown, only to 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 meat packer 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, by Bitcoin, and by a layer of intermediate affiliations. Given the maturity of the GandCrab/REvil malware platform, and the amount of money that can be extorted through the ransomware model, I would not be surprised if this group doesn't take away a few lessons from the DarkSide/ Colonial Pipeline, JBS Foods and Kaseya over-achievements 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.