Security Now! #823 - 06-15-21

TLS Confusion Attacks

This week on Security Now!

This week we're going to start by looking at a moment-by-moment reconstruction of a recent Chrome browser attack and patch battle. Then we're going to recap last week's industry wide June patchfest followed by looking at TikTok's controversial but unsurprising privacy policy update. We need to also cover the wonderful spy-novel'ish ANOM sting operation which lowered the boom on as many as 800 criminals. For our happily infrequent Errata section we'll challenge an apparently erroneous statement I made last week, then I want to share an interesting laptop data recovery experience which BitLocker made much more complex a few weeks ago which I think our listeners will find interesting. Then we're going to tackle this week's topic of some very troubling research which again demonstrates just how difficult it is to design robustly secure networked systems.

Web Browser News

Being #1 is a mixed blessing

Google is finding that with offering the world's #1 web browser comes being the world's #1 web target. Using some reports published by Kaspersky Labs I was able to reassemble a timeline of some recent Chrome vulnerabilities and fixes. What it reveals is instructive...

Exactly two months ago, during April 14th & 15th of this year, Kaspersky Labs detected a wave of highly targeted attacks aimed at multiple companies. Upon closer analysis they discovered that these attacks were exploiting chained Chrome and Windows 0-day vulnerabilities. Though they were unable to retrieve the exploit code used to accomplish remote code execution in Chrome, they were able to obtain and analyze an elevation of privilege (EoP) exploit that was used to escape the sandbox to obtain system privileges. Remember that while we tend to focus upon remote code execution attacks, exploitable elevation of privilege vulnerabilities are also quite powerful since they inherently breach the security boundaries that we rely upon to allow externally provided web browser code to run in our browsers while curtailing what that code can do. If code can arrange to make itself privileged then "can do" code is what results.

In this case, the elevation of privilege exploit that they discovered had been fine-tuned to work against the latest and most prominent builds of Windows 10 dating back to 17763 – RS5, and also including 18362 – 19H1, 18363 – 19H2, 19041 – 20H1 and 19042 – 20H2. It dos all this by exploiting two separate vulnerabilities in Window's OS kernel. Five days after this discovery, on April 20th, Kaspersky reported these vulnerabilities to Microsoft and two successive CVE were assigned, CVE-2021-31955 for the information disclosure vulnerability and CVE-2021-31956 for the elevation of privilege vulnerability. Both of these in-the-wild vulnerabilities were patched last Tuesday as part of the June's Patch cycle.

The observed attacks were all conducted through Chrome, but as I noted before, Kaspersky was unable to retrieve the JavaScript which fully implemented the entire exploit. But they did have a clue: The Pwn2Own competition had taken place the preceding week, April 6th through 8th and, of course, Chrome was a prominent target. During that competition, one of the participating teams was able to successfully demonstrate an exploitation of the Chrome renderer process using a Type Mismatch bug.

And a few days later, on April 12th, the Chromium developers committed two issues #1196683 & #1195777 to the open-source repository of V8, the Chrome and Chromium JavaScript engine. Both were Type-related bug fixes. One of these bug fixes (...6683) patched the vulnerability that was used during Pwn2Own, and both bug fixes were committed together with regression tests – which are JavaScript tests to trigger these vulnerabilities. Later the same day, a user with the Twitter handle @r4j0x00 published a working remote code execution exploit on GitHub, targeting an up-to-date version of Google Chrome. That exploit used the vulnerability from issue ...6683 to execute shellcode in the context of the browser renderer process.

The exploit published to Github did not contain a sandbox escape, and was therefore intended to work only when the browser was launched with the command line option –no-sandbox. Then, on the 13th, the day before Kaspersky first became aware of the attacks in the wild, Google released Chrome update 89.0.4389.128 for Windows, Mac and Linux with fixes for two vulnerabilities. The one (CVE-2021-21220) used during Pwn2Own was one of them.

However, since several of Kaspersky's customers, who were attacked on April 14th & 15th, already had their Chrome browsers fully updated, Kaspersky believes that the Pwn2Own-originated vulnerability was not used in those attacks.

Then, the next day, on April 14th, Google moved from v89 to v90 (clearly a major planned feature release) with 90.0.4430.72 for Windows, Mac and Linux. This release closed the door on 37 vulnerabilities. And on the same day — the 14th — a new Chrome exploit was posted on GitHub and thus released to the public. That newly published exploit used that second vulnerability ...5777 which, even though it had been committed on the 12th, still worked on the just-released Chrome v90 from the 14th. Six days later it was fixed on the 20th.

Kaspersky suspects that the attackers were also able to use the JavaScript file containing the regression test to develop the exploit and were probably using this second vulnerability in their attacks.

Talk about cat & mouse! So what do we learn from this bit of detail?

We learn that attackers are extremely active and deft at scrutinizing everything that happens in public view. They are looking everywhere at once. In this instance, the results of the Pwn2Own competition likely primed them to be on the lookout for Chromium commits that would soon follow. And follow they did. The Chromium project is at the disadvantage of inherently being open. It's a good thing to create JavaScript regression tests to make sure that bugs which have been fixed never return. But as likely happened here, the appearance of the regression tests probably predated by 8 days the deployment of an updated Chrome browser which fixed those flaws. That created an 8-day exploitation window during which a problem was publicly known and documented but not yet patched and in the hands of Chrome's users.

We can also see from this almost minute by minute back and forth, that those who are responsible for security can never take a vacation from their jobs. Newly discovered vulnerabilities must be immediately stomped out and those fixes must be rapidly deployed. And we can see that today's web browser attackers are hyper-vigilant, too. They know that they won't have much time to leverage any transient advantage they might briefly obtain.

We also learn that we should be SO THANKFUL that Microsoft had the wisdom to scrap their independent web browser development in favor of their adoption of the Chromium project. Web browsers can no longer be the domain of massive, slow moving bureaucratic behemoths. “Web Browsers” is no longer where Microsoft should be. Putting any modern web browser up for the world to attack requires far too much agility. Google has clearly organized their Chromium group for short-cycle nearly instantaneous response. Nothing less would be sufficient to protect Chrome's users from today’s attackers.

Security News

Industry wide patch Tuesday

So last Tuesday was our industry's patch Tuesday, and as I noted last week, many other companies have decided to synchronize their patch cycles with Microsoft. Intel fixed 73 security vulnerabilities which included some that were severe, impacting the UEFI/BIOS firmware for Intel processors, as well as its Bluetooth products, Active Management Technology tools, its NUC mini PC offerings, and in its own security library. Among other things, there was a problem with a random number generator not being as random as hoped. Among those fixed were some rated CRITICAL, though Intel boasted that most were found internally. Intel's Jerry Bryant posted:

> *"Today we released 29 security advisories addressing 73 vulnerabilities. 40 of those, or 55%, were found internally through our own proactive security research. Of the remaining 33 CVEs being addressed, 29, or 40%, were reported through our bug bounty program. Overall, 95% of the issues being addressed today are the result of our ongoing investments in security assurance, which is consistent with our 2020 Product Security Report."*

That's right, Jerry. And if only you had found those problems before you deployed the buggy code into the wild, the world would not now need to scramble around to update our known-defective devices before the forces of darkness reverse engineer those changes to use them against the unwitting. But, better late than never. Anyone using an Intel NUC — I'm a big NUC fan and I expect that's going to be my chosen platform moving forward — ought to check to see whether there's an update to their device's firmware waiting for them. Intel doesn't appear to offer a firmware appraisal tool. So you need to figure out which model you have and which version of firmware it's running... then go see whether it's the latest available.

By far the winner of this month's CRITICAL patch derby was Adobe who addressed 41 CVEs in their own Patch Tuesday last week, 21 of them rated as CRITICAL in severity. They impacted Acrobat and Reader, Photoshop, Creative Cloud Desktop, their RoboHelp Server, Adobe After Effects, and Adobe Animate. And I know that things seem sort of gloomy in the software industry right now. But, really, we should count our blessings that Adobe never tried to do an operating system! Can you imagine that? They were never able to make FLASH safe.

And speaking of operating systems, the original second Tuesday patcher, Microsoft, managed to break last month's 16-month low mark, which was just 55 flaws, by coming in this month with only 50 or 51 depending upon how one counts. However, last month had three 0-days and this month we have 6 exploits appearing in the wild. The vulnerabilities fixed span Windows desktop, SharePoint Server, the Windows kernel and Outlook.

Two of the vulnerabilities are related to a separate vulnerability in Adobe Acrobat Reader for which Microsoft released fixes in its Enhanced Cryptographic Provider. By leveraging these flaws, an attacker could elevate their privileges on the targeted system if they trick a user into opening a specially crafted PDF file in a vulnerable version of Adobe Acrobat or Adobe Reader from within a not-yet-patched version of Windows. Microsoft stated in its advisories that it's seen these vulnerabilities being exploited in the wild. Adobe had previously fixed this in last month's security update. So if you're using Acrobat or Reader, having both patched would be safest.

One of the critical vulnerabilities fixed this month was in the Windows Defender which would have allowed an attacker to execute remote code on the host machine. The good news is that Defender is continually keeping itself up to date, even under Windows 7.

Microsoft Office's MSGraph component also had a remote code execution vulnerability that could be used to deliver a malicious payload to a victim's machine though Microsoft Office since the built-in MSGraph component can be embedded in most Office documents which will then exploit the flaw when an Office document is opened.

Another vulnerability being actively exploited in the wild is a privilege escalation flaw in the DWM Core Library (DWM is the Desktop Windows Manager). It can be exploited by running an executable or script on the local machine. It has a CVSS of 8.4 though Microsoft calls it only "Important."

And, finally, CVE-2021-31955 is a flaw that allows an attacker to read the contents of the Windows OS Kernel memory from a user-mode process. As we know, that's a big "game over" since the Kernel is the land of exploitable secrets and Microsoft stated in their advisory that it has also been actively exploited in the wild.

So... another patch Tuesday and a few hundred fewer outstanding bugs in the software the world is using. That would be a hopeful sign if we didn't appear to be creating them at an even faster clip than we're eliminating them.

TikTok Quietly Updated Its Privacy Policy to Collect Users' Biometric Data

Leo, unlike you, being "hip" is not a designation I have ever, at any prior point in my life, laid claim to. I just never had that. And all indications are that I'm headed in the other direction. But those who **are** hip will know that "TikTok" is a popular short-form video-sharing service. And it's a service that was probably not happy to have made news in the tech press recently due to a somewhat chilling explicit change in their service's privacy policy.

Stepping back a bit first, it's likely that the recent changes to their privacy policy followed from their $92 million settlement of a class-action lawsuit. We've talked about the super-tight Illinois Biometric Information Privacy Act, abbreviated "BIPA". Illinois is where privacy lawsuits go to be filed since BIPA, as we've previously covered extensively, pretty much takes no prisoners.

In this case, the lawsuit was originally filed just over a year ago in May of 2020 and the group of plaintiffs in this suit, which was consolidated from more than 20 separate cases filed against TikTok, reached a preliminary settlement just before the end of this February. The suit alleges that TikTok violated BIPA when it collected and shared the personal and biometric information of its users without first obtaining their consent.

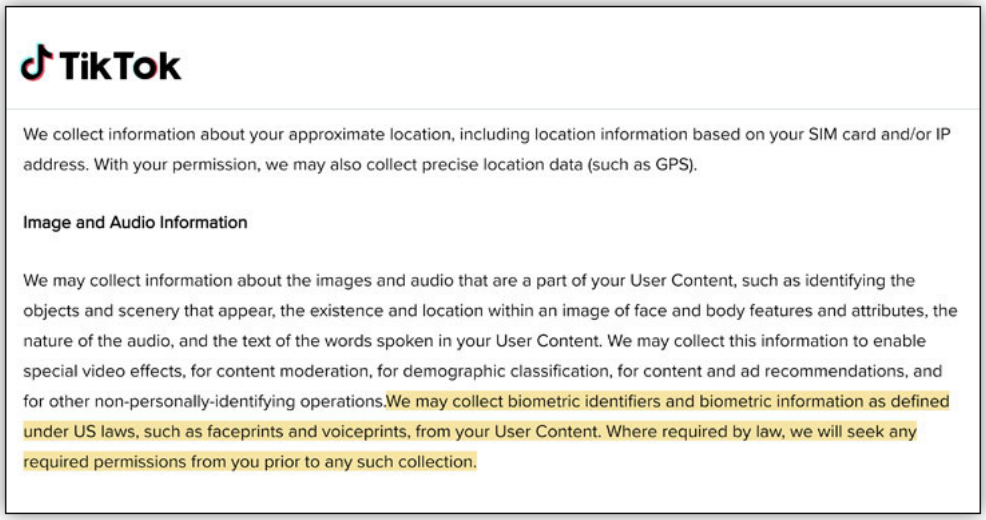
Attorney Ekwan Rhow who is with the firm (I kid you not) of "Bird, Marella, Boxer, Wolpert, Nessim, Drooks, Lincenberg & Rhow" said of the settlement: "This is one of the largest settlements ever achieved in a consumer BIPA case, and one of the largest privacy class action settlements. It presents an excellent recovery for the class, and it serves as a reminder to corporations that privacy matters, and they will be held accountable for violating consumers' rights."

Another attorney in the case, Beth Fegan, added: "Illinois is on the cutting edge of privacy law, and this settlement enforces those crucial protections. Biometric information is among the most sensitive of private information. It's critical that privacy and identity is protected by stalwart governance to guard against underhanded attempts at theft."

Yeah, and I'd love to know what portion of the settlement went to the members of the aggrieved class and what portion those oh-so-concerned concerned attorneys retained for themselves. But that's beside the point. And although $92 million is an arresting sum, it's actually small potatoes when measured against Facebook's $650 million BIPA settlement. That $650 million settlement was arrived at only after the judge rejected the previous $550 million settlement, arguing that the smaller figure was not enough to compensate the sheer number of people in the class, based on the penalties laid out in BIPA law.

And, of course, this is not the first time TikTok has made the tech press news. They were also in legal hot water when the US federal government was considering a ban on TikTok after privacy experts warned that the government of China might have access to personal information gathered through the app.

But in any event, the recent settlement alleged that TikTok was using clandestinely captured biometric and personal data from users in the U.S. to target ads without meeting the informed consent requirements of Illinois state law. So, as part of the settlement, TikTok agreed to avoid collecting or storing biometric information, biometric identifiers, geolocation, or GPS data unless expressly disclosed in its privacy policy. And so, we have TikTok's new privacy policy which is what has garnered so much attention and concern. It states:



TikTok

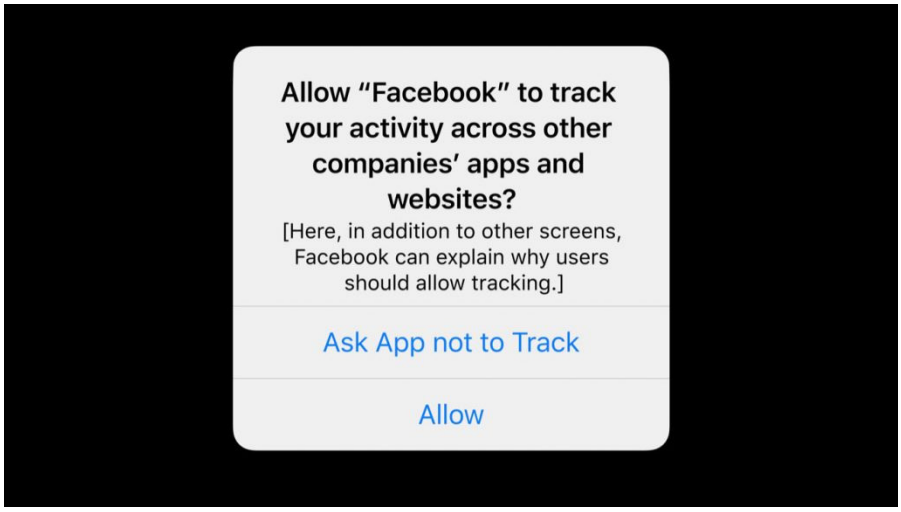
We collect information about your approximate location, including location information based on your SIM card and/or IP address. With your permission, we may also collect precise location data (such as GPS).

Image and Audio Information

We may collect information about the images and audio that are a part of your User Content, such as identifying the objects and scenery that appear, the existence and location within an image of face and body features and attributes, the nature of the audio, and the text of the words spoken in your User Content. We may collect this information to enable special video effects, for content moderation, for demographic classification, for content and ad recommendations, and for other non-personally-identifying operations.We may collect biometric identifiers and biometric information as defined under US laws, such as faceprints and voiceprints, from your User Content. Where required by law, we will seek any required permissions from you prior to any such collection.

So TikTok is now formally disclosing what they were presumably doing before: collecting biometric information "such as", but apparently also not limited to, faceprints and voiceprints. And it should escape no one that the whole point of biometrics is identification. Again, not new information. But now out of the shadows. And given that only five states in the U.S — California, Illinois, New York, Texas, and Washington — have laws which restrict the collection of biometric data, the updated privacy disclosure likely means that TikTok will not be required to obtain permission from its users in other states. In other words, users are consenting to have their biometric data collected simply by agreeing to TikTok's terms of service.

iOS 14.5 requires apps to obtain explicit tracking permission
We haven't talked about this here, but iOS 14.5 added the provision and requirement for any apps wishing to track their users' activity across other company's apps and websites to expressly request and obtain such permission.



https://www.flurry.com/blog/ios-14-5-opt-in-rate-att-restricted-app-tracking-transparency-worldwide-us-daily-latest-update/

Overall, the feedback has been a resounding "no thanks." The application analytics company, Flurry, has been sampling 2.5 million users since the release of iOS v14.5. Their research shows that the worldwide global "opt-into tracking" rate has settled at about 25%. So 75% of users globally are saying, thanks but no thanks. And users in the United States are even more tracking phobic. For the U.S., the "it's okay, go ahead and track me" rate has settled out at 14%. So 86% of U.S. users want to see tracking 86'd from their online experiences.

ANOM
This news was breaking just as we were recording last week's podcast. So today I'm able to provide a comprehensive description of this classic though quite high-tech sting operation:

Europol's press release of last Tuesday, June 8th was headlined: "800 criminals arrested in biggest ever law enforcement operation against encrypted communication"

The ANOM sting operation, which was also known as Operation Trojan Shield by the FBI and Operation Ironside by the AFP (the Australian Federal Police). It is/was a collaboration by law enforcement agencies from as many as 20 countries. It ran from late 2018 when it got off the ground through 2021. And during that time the operation intercepted and exploited the intelligence obtained from 27 million messages.

Those messages were sent through and intercepted by the supposedly secure smartphone-based messaging app ANOM. The ANOM service, which was widely used by criminals, was actually a trojan horse covertly distributed by the US FBI and the Australian Federal Police (AFP). It enabled them to monitor all communications taking place across the network. And finally, through collaboration with other law enforcement agencies worldwide, the operation resulted in the arrest of over 800 suspects allegedly involved in criminal activity, in 16 countries.

Among the arrested people were alleged members of Australian-based Italian mafia, Albanian organised crime, outlaw motorcycle gangs, drug syndicates and other organised crime groups. At its height, the ANOM service grew to service more than 12,000 encrypted devices to over 300 criminal syndicates operating across more than 100 countries.

The platform's goal was to target global organised crime, drug trafficking, and money laundering organisations, regardless of where they operated. Cleverly, the service was designed to appeal to the underworld by offering an encrypted device with features sought by the organised crime networks including remote wipe and duress passwords — passwords which could be given when under duress which would have the opposite of the intended effect, to wipe rather than unlock a device. These features gradually persuaded criminal networks to adopt the device.

The way this happened is the stuff of spy novels. First, the shutdown of the Canadian secure messaging company Phantom Secure in early 2018 left international criminals in need of an alternative system for secure communication. Around the same time, the FBI branch office in San Diego, California, was working with a person who had been developing a "next-generation" encrypted device for use by criminal networks. That individual was facing charges and cooperated with the FBI in exchange for a reduced sentence. He offered to develop ANOM and to then distribute it to the underworld through their existing networks, with which he was familiar. The first communication devices with ANOM were offered by this informant to three former distributors of the Phantom Secure system around October 2018. The FBI also negotiated with an unnamed third country to set up a communication interception, but based on a court order that allowed passing the information back to the FBI. Since October 2019, ANOM communications have been passed on to the FBI from this third country.

During the culmination of ANOM's operation a series of large-scale law enforcement actions were executed across 16 countries resulting in more than 700 residential searches, more than 800 arrests and the seizure of over 8 tons of cocaine, 22 tons of cannabis and cannabis resin, 2 tons of synthetic drugs (amphetamine and methamphetamine), 6 tons of synthetic drug precursors, 250 firearms, 55 luxury vehicles and over $48 million in various worldwide currencies and cryptocurrencies. And innumerable spin-off operations are still planned for weeks to come.

Wikipedia provided a bit more interesting background information about the ANOM devices:

The ANOM devices consisted of a messaging app running on smartphones that had been specially modified to disable normal functions such as voice telephony, email, or location services. After checking that normal functionality was disabled, the messaging apps then communicated with one another via supposedly secure proxy servers, which then copied all sent messages to servers controlled by the FBI. The FBI could then decrypt the messages with a private key associated with the message, without ever needing remote access to the devices. The devices also had a fixed identification number assigned to each user, allowing messages from the same user to be connected to each other. According to a since-deleted Reddit post discovered by Motherboard, the ANOM app was "for Android"; a WordPress blog post described the app as using a "custom Android OS".

About 50 devices were distributed in Australia for beta testing from October 2018. The intercepted communications showed that every device was used for criminal activities, primarily being used by organised criminal gangs.

Use of the app spread through word of mouth, and was also encouraged by undercover agents; drug trafficker Hakan Ayik was identified "as someone who was trusted and was going to be able to successfully distribute this platform", and without his knowledge was encouraged by undercover agents to use and sell the devices on the black market, further expanding its use. After users of the devices requested smaller and newer phones, new devices were designed and sold. The most commonly used languages on the app were Dutch, German and Swedish.

After a slow start, the rate of distribution of ANOM increased from mid-2019. By October 2019, there were several hundred users. By May 2021, there had been 11,800 devices with ANOM installed, of which about 9,000 were in use. New Zealand had 57 users of the ANOM communication system. The Swedish Police had access to conversations from 1,600 users, of which they focused their surveillance on 600 users. Europol stated 27 million messages were collected from ANOM devices across over 100 countries.

Some skepticism of the app did exist; one March 2021 WordPress blog post called the app a scam.

I'd call it a significant success!

Errata

"Windows 10" — the last Windows ever?
Apparently not. I was absolutely certain that we were clearly and formally told by representatives of Microsoft that following Windows 8.1, the next Windows, to be called "10" would be free, would create new "revenue opportunities" for Microsoft — (remember that?) in other words, that the users of this next Windows would become profit centers for Microsoft — that it would be the last version of Windows.

So I was stunned when, last week, during her WindowsWeekly conversation with you, Leo, and Paul, Mary Jo Foley said: "Uh... nope. Microsoft never said that." I put this under "Errata" since if this was true, I had been spreading an unfounded rumor. At least I wasn't alone. Rich Woods of XDA Developers tweeted:

Rich Woods / @TheRichWoods / 10:18 AM · Jun 11, 2021
"The most mind-blowing news story from this week was broken by @maryjofoley on Windows Weekly, and it's that Microsoft never said Windows 10 was the last version of Windows. One developer evangelist said it, Microsoft never corrected it, and everyone ran with it. It became lore."

It turns out that back in 2015, Microsoft's developer evangelist Jerry Nixon stated that Windows 10 was the last version of Windows. Quoting him exactly, he stated: *"Right now we're releasing Windows 10, and because Windows 10 is the last version of Windows, we're all still working on Windows 10."* In other words, he said that "we're releasing it" and "we're still working on it" — clearly implying that Win10 will be an ongoing effort. And, indeed, that what's we've seen. It would be difficult to say with a straight face, however, that this plan has gone smoothly, what with Windows 10 2004 being released in 2010. ***What?!?!*** It's been a mess.

Last Wednesday, Mary Jo Foley explained that Microsoft themselves never publicly said, in plain language, that Windows 10 is the last version of the Windows operating system. Their developer evangelist Jerry Nixon said it, and backed it up with evidence at the time... and as Mary Jo noted, Microsoft's Public Relations team never denied it.

Sci-Fi

Project Hail Mary — Lorrie loved the book. She blew through it, as I knew she would. At one point she was standing in front of the stove, reading with the book open, as popcorn popped. And she got a little choked up at the end with the way everything turned out.

Seeing that she loved it so much, I gave her the first of the Honor Harrington books. But after a couple of hours she asked whether the entire book would be about military space ordnance? I said "Uh… yeah. Well, there's a lot of that." So now she's reading Daniel Suarez's Daemon and that one may be a hit! :)

SpinRite

The Curious Data Recovery Adventure

- A colleague of Lorrie's whose laptop had irreplaceable data.
- WHEA_UNCORRECTABLE_ERROR
- Diagnostics / Updated the BIOS
- A different NVMe drive passed the diags.
- NVMe - Plugs directly into the PCIe bus
- TPB / Secure Boot / BitLocker
- External NVMe with/Dual thermal glued copper heatsinks.
- Two takeaways:
  - If you are not continually cloning your important data to some other device, make absolutely certain that you have a current 48-digit BitLocker Recovery Key.
  - Tuck away in the back of your head that Thunderbolt IS PCIe and that an internal drive can appear exactly like an internal drive — and BOY is it fast!!
- SYNC.COM Get an extra 1GB (=6GB) https://grc.sc/sync

TLS Confusion Attacks

Some very interesting and quite depressing research will be presented at the forthcoming 30th Usenix Security Symposium and this summer's BlackHat USA in Las Vegas at the end of next month. Even though it's not completely new, it teaches us some interesting lessons about unintended edge case consequences.

The researchers apparently struggled to come up with a cute name for this. Since the problem they were exploring was the mischief that could be created by deliberately confusing the application layer within TLS-authenticated and protected connections, they had "Application Layer Protocol Confusion" or ALPC. That's not really anything. But if you add a pair of judiciously placed gratuitous 'A's into that you can force this to become A. L. P. A. C. A., or Alpaca... which explains why I chose to name this podcast "TLS Confusion Attacks."

Regardless of their name, they're a truly interesting instance of an unintended consequence edge case. Or to use the official term: Oops!!

Here's what happened. In the beginning, we had the UDP and TCP IP protocols. UDP was mostly used for DNS. But over the TCP protocol we ran TELNET and FTP and SMTP and POP and IMAP and HTTP. Those were the so-called Application Layer protocols running on top of the TCP Transport Layer. And the world was simple. But it was not secure. Passive eavesdroppers could listen in on passing Internet packets to see what everyone was up to — so there was no assurance of privacy. And active man-in-the-middle attackers could redirect traffic to other destinations without detection — so there was no authentication.

But we're clever. So we grafted on SSL, which over the years evolved into TLS. The idea was that our existing plaintext application protocols would be given different ports when their TCP connections should be authenticated and encrypted. HTTP over port 80 became HTTPS over port 443. FTP's port 21 became FTPS over port 990, and SMTP's port 25 became port 587 when TLS would be used to secure the SMTP protocol.

When using these alternate secured ports, immediately after establishing the famous TCP 3-way handshake, a shim would be introduced in between the TCP transport layer and the Application protocol layer. Before any application layer traffic would be allowed to flow, the two endpoints would need to exchange another set of packets to cryptographically establish the identity of one or both of the endpoints — thus providing endpoint authentication — and agree upon an encryption key that they would henceforth use to encrypt all subsequent communications — thus providing communications privacy.

Problem solved? Pretty much... but not entirely. The trouble was, and has always been, that the SSL/TLS protocol is bound to the TCP connection but not to the underlying application layer protocol whose traffic it is carrying. In other words, the TLS protocol itself is unaware of the IP and Port to which it is connecting. It does the same thing if the connection is to HTTPS as if it was to Secure SMTP. TLS doesn't care at all. But the underlying application protocol DOES care which port it's connected to because that determines which service it's talking to. And therein lies the edge case. Since TLS is not bound to the connection's IP and Port, they can be changed

and TLS won't care. But the underlying protocol does care. And this opens what appeared to be a safe and secure system to what the researchers term "cross-protocol attacks."

The Abstract of their paper explains:

> *TLS is widely used to add confidentiality, authenticity and integrity to application layer protocols such as HTTP, SMTP, IMAP, POP3, and FTP. However, TLS does not bind a TCP connection to the intended application layer protocol. This allows a man-in-the-middle attacker to redirect TLS traffic to a different TLS service end point on another IP address and/or port. For example, if subdomains share a wildcard certificate, an attacker can redirect traffic from one subdomain to another, resulting in a valid TLS session. This breaks the authentication of TLS and cross-protocol attacks may be possible where the behavior of one service may compromise the security of the other at the application layer.*
>
> *[Okay, so let me pause for a second. What they're saying, is that if, for example, GitHub were to use the same TLS certificate for HTTPS as for Secure SMTP, it might be possible to redirect an incoming user's web browser connection, which is intended for GitHub's HTTPS port 443 to their Secure SMTP port 587. And because the same TLS certificate will be used to negotiate the connection's TCP security either way, it may be possible to take advantage of this protocol level confusion. Okay, continuing with their Abstract...]*
>
> *In this paper, we investigate cross-protocol attacks on TLS in general and conduct a systematic case study on web servers, redirecting HTTPS requests from a victim's web browser to [Secure] SMTP, IMAP, POP3, and FTP servers. We show that in realistic scenarios, the attacker can extract session cookies and other private user data or execute arbitrary JavaScript in the context of the vulnerable web server, therefore bypassing TLS and web application security.*
>
> *We evaluate the real-world attack surface of web browsers and widely-deployed email and FTP servers in lab experiments and with internet-wide scans. We find that 1.4M webservers are generally vulnerable to cross-protocol attacks, i.e., TLS application data confusion is possible. Of these, 114k web servers [just shy of 1 out of every 10] can be attacked using an exploitable application server. Finally, we discuss the effectiveness of TLS extensions such as Application Layer Protocol Negotiation (ALPN) and Server Name Indiciation (SNI) in mitigating these, and other, cross-protocol attacks.*

The main takeaway here is, I think, that not only is security difficult... it's even significantly more difficult than we know! In fact, it might even be more difficult than we CAN know!

To put this in context, this is not the end of the world. The attack requires sophistication and the ability to establish a man-in-the-middle position — the ability to actively manipulate the traffic coming and going from a victim in real time. But it's certainly within the reach of state-level attackers and it does mean that users are not really obtaining the security that we think we've always had. These attacks also require a very specific set of circumstances. The researchers describe what they uncovered:

In practice, cross-protocol attacks are sensitive to many requirements, such as certificate compatibility, ability to upload, download, or reflect data, and application tolerance towards syntax errors caused by mixing two protocols in one channel. In our case study of cross-protocol attacks on HTTPS, using SMTP, IMAP, POP3, and FTP application servers, we address these concerns in three evaluations.

1. *We identified 25 popular SMTP, IMAP, POP3, and FTP implementations and evaluated their suitability for cross-protocol attacks on HTTPS in a series of lab experiments. We found that 13 are exploitable with at least one attack method. We also implemented a full proof-of-concept that demonstrates all three attack methods on a well-secured web server, using exploitable SMTP, IMAP, POP3, and FTP application servers.*

2. *We evaluated seven browsers for their error tolerance. We find that Internet Explorer and Edge Legacy still perform content sniffing and thus are vulnerable to all presented attacks, while all other browsers allow at least FTP upload and download attacks.*

3. *In an internet-wide scan, we collected X.509 certificates served by SMTP, IMAP, POP3, and FTP servers. We analyzed how many of these are likely to be trusted by major web browsers. For each certificate, we extracted the hostnames in the CN field and SAN extension and checked if there exists a web server on these hosts. We found 1.4M web servers that are compatible with at least one trusted application server certificate, making them vulnerable to cross-protocol attacks. Of these, 119k web servers are compatible with an application server that is exploitable in our lab settings.*

So what can be done about this now?

This is a decidedly mixed blessing because while it's unlikely to occur, given the proper conditions it can. And when it's exploited it can result in logged-on session hijacking and in the execution of malicious JavaScript being sourced from a fully trusted web server. The fact that it's unlikely to occur reduces the pressure to make any changes. And what's worse, unlike Dan Kaminsky's fix for DNS that only needed to be made on the DNS server side, the changes that need to be made here must occur at both ends of our TLS connections.

There is currently a well-defined solution for this, known as ALPN. That stands for Application Layer Protocol Negotiation. And it does exactly what we would hope and expect: It's an optional extension for TLS connection setup which allows the endpoints to explicitly agree upon the protocol that the connection will be carrying — and it does so in the TLS handshake without requiring any additional packet round trips. And since support for the HTTP/2 protocol needs to be able to smoothly upgrade an HTTP/1.1 connection to HTTP/2 — if both endpoints agree, this is done via ALPN so it's support is already emerging.

However, protection from web-based protocol confusion attacks requires that ALL TLS connection-accepting services, which could be unwitting participants in protocol confusion, such as FTP, SMTP, POP3, IMAP, etc. also support ALPN and flatly reject and terminate any non-ALPN enhanced connections. And, sadly, I cannot imagine a world in which that's going to happen.

We can imagine that future non-web services might begin to incorporate ALPN awareness, but that's entirely different from refusing connections from remote clients that are not offering ALPN-enhanced TLS... which is what's required to prevent these attacks. So the outlook is not good.

The researchers conclude:

> *We demonstrated that the lack of strong authentication of service endpoints in TLS can be abused by attackers to perform powerful cross-protocol attacks with unforeseeable consequences. Our internet-wide scans showed that it is common for administrators to deploy compatible certificates across multiple services, possibly without consideration to cross-protocol attacks. We also showed that cross-protocol attacks are practical, although the impact is limited and difficult to assess from lab experiments alone. In the real-world, cross-protocol attacks will always be situational and target individual users or groups. However, it is also clear that existing countermeasures are ineffective because they do not address all possible attack scenarios. We have identified one countermeasure that is far superior to others: the pervasive use of the ALPN extension to TLS by both client and server. Luckily, ALPN is easy to deploy with the next software update without affecting legacy clients or servers.*

But, again, it's necessary to actively refuse any non-ALPN enhanced connections and that's not going to happen in our lifetime.

https://alpaca-attack.com/
https://alpaca-attack.com/ALPACA.pdf