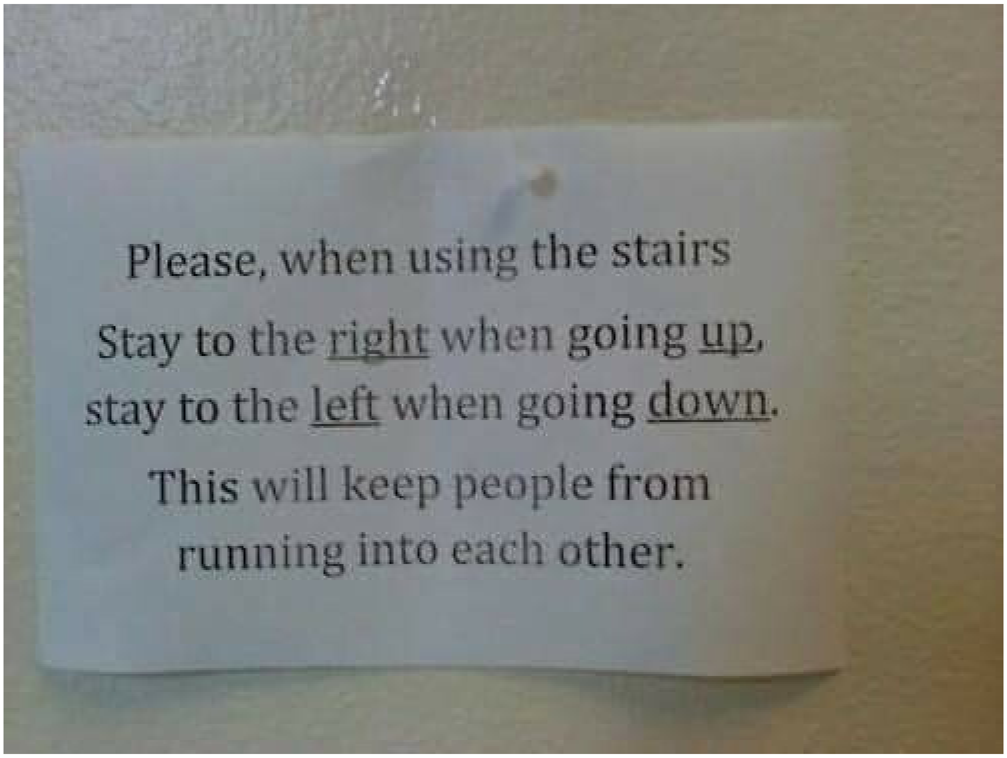
Security Now! #913 - 03-07-23

A Fowl Incident

This week on Security Now!

This week's answers are many: How has Fosstodon survived a sustained DDoS attack? Or has it? What luck have Europol and the FBI had with taking down DDoS-for-hire services and have they returned? What's the point of blocking TikTok, and is it even possible? What happens when government-backed surveillance goes rogue? What exactly is "Strategic Objective 3.3" and what, if anything, does it portend for future software? Should you enable GitHub's new secret scanning service and get scanned? What exactly did CISA's secretive red-team accomplish; and against whom? Which messenger apps have been banned by Russia, who's missing from that list, and why? What exactly is old, that's new again, what happens when everyone uses the same cryptographic library for their TPM code, what's the latest WordPress plug-in to threaten more than one million sites and why has Russia fined Wikipedia? And once we've put that collection of need-to-know questions to rest we're going to examine the surprising revelations that surface as we unearth the Fowlest of recent security incidents.

Why real world testing is important…



Please, when using the stairs
Stay to the right when going up,
stay to the left when going down.
This will keep people from
running into each other.

Security News

DDoS’ing Fosstodon:

Last Thursday, Chris Miller posted on Fosstodon:

> *Hi all. We're still under a major DDOS attack, and that's why mobile and desktop clients are not currently working. We've had to put the site behind Cloudflare temporarily until the attack stops. We're looking at other long term solutions, but we need to get through the current moment first. For now, use the web interface. Thanks for your patience.*

And as of yesterday when I last looked that attack is still ongoing. Fosstodon, as its name suggests, is the largest Mastodon instance inhabited by open-source software denizens. And, sadly, this most recent attack marks only the latest in a growing string of DDoS attacks that have hit and easily brought down unprotected Mastodon server instances over the past few months. The pace of these attacks increased significantly after Mastodon gained a huge amount of attention thanks to the mass exodus from Twitter following Elon's takeover and his subsequent actions which struck many as not being in the best interests of neither the larger Twitter community nor their own. Fearing the approaching end of Twitter, many jumped over to the Mastodon’s decentralized model which inherently prevents a repeat of the same.

And, in fact, “feditips@mstdn.social” (“fediTips” as in Federated Tips) posted:

> *The mastodon.social server is currently under a heavy DDOS attack and may not work properly. The 12,164 other servers on the network are unaffected. This is part of the reason why federated networks are a good idea: if one server goes down, the others work fine. The more spread out we are on small and medium sized servers, the harder it is for anyone to take down the network because there's no obvious target. There are also many other reasons why federation is good:* https://fedi.tips/why-is-the-fediverse

There's nothing much more to say here other than to note that FediTips is certainly correct. On the one hand, the nearly week-long DDoS attack against Fosstodon has rendered its mobile and desktop clients inoperable. But thanks to their web front-end moving behind Cloudflare's front end protection, at least their web interface is operable and they remain on the air.

As I noted recently, when we were talking about the most recent DDoS connection rate attack record being broken, no stand alone server on the Internet can withstand today's DDoS attacks – not even a little. Today’s modern Iot botnet-based attacks are large enough to swamp even medium size bandwidth providers. So the only recourse is to move behind the protective skirts of one of today's major DDoS protection services. The problem is, such service is not free unless the service wishes to provide such service charitably.

And that's something that annoyed me. Elsewhere, on Reddit, the Fosstodon admin was grousing about the need to be rescued by a commercial service such as Cloudflare. In my opinion they should consider themselves incredibly fortunate that a facility such as Cloudflare exists... otherwise they **would** be off the ‘Net for as long as the DDoSing cretins wanted them off the ‘Net. Period.

DDoS for Hire takedowns

While we're on the subject of DDoS attacks. The network security provider Netscout has noted that the efforts by Europol and the FBI to take down more than 50 DDoS-for-hire services in the middle of last December has indeed led to a measurable decline in DDoS attacks. Those declines have been recorded at broadband providers across both the US and EU. And, moreover, Netscout said that DDoS traffic has remained at lower levels for the month after the takedown. This suggests that no major new players have immediately appeared on the underground DDoS market to fill the void after last year's takedowns. On the other hand, four or five weeks isn't much time to wait.

To illustrate their data, Netscout provided a nice chart so that we could see for ourselves:

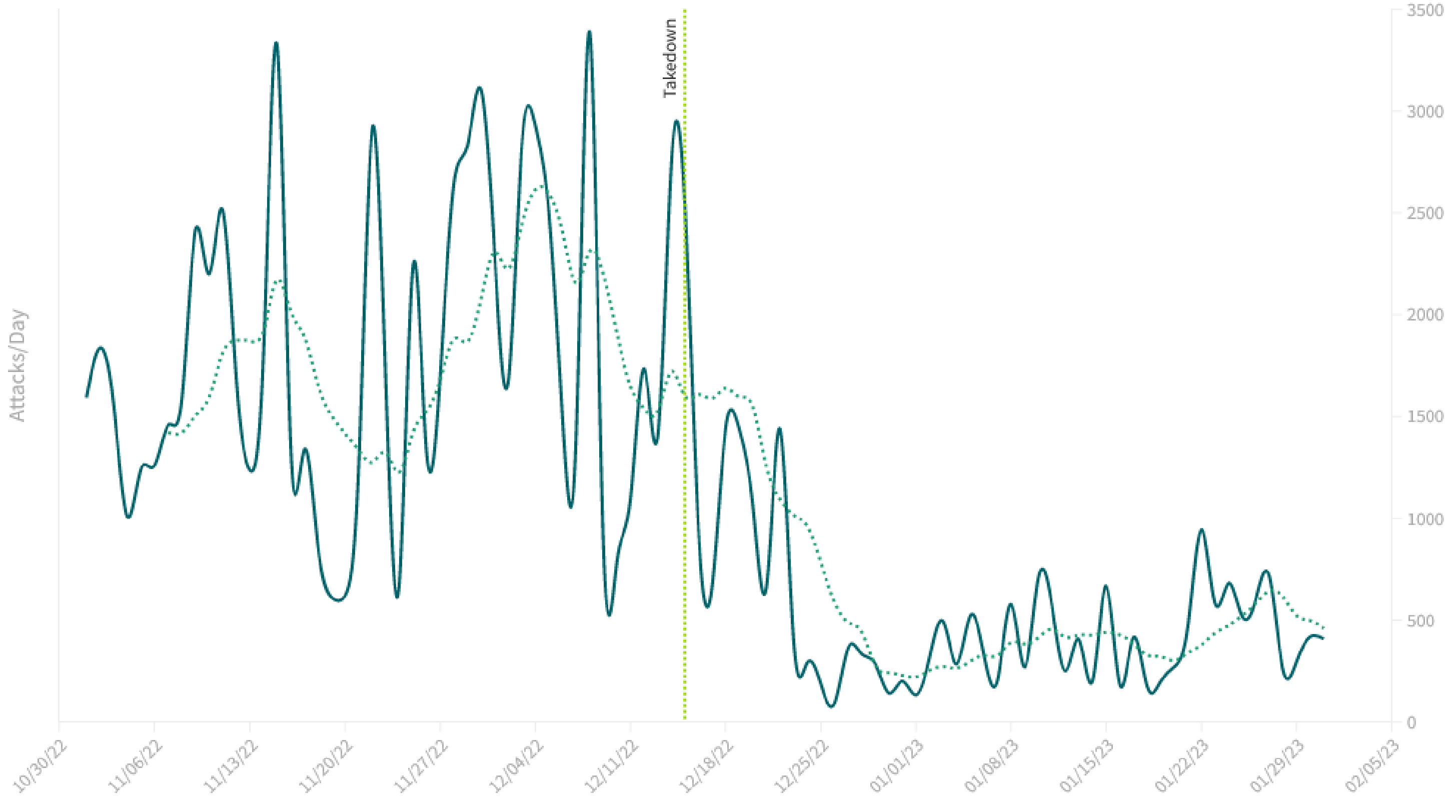


Figure 2: Attacks toward a large US broadband provider.

This chart leaves little doubt that the takedown effort had its intended effect. Where the takedown occurred there's a dramatic decrease in... something. So, wondering exactly what that something is, you then look at the horizontal scale to learn that the chart's horizontal scale is in attacks per day. So you learn that while, yes, the number of attacks per day did indeed fall from their peak of around 3500 attacks per day which occurred twice, unfortunately the wider DDoS problem problem is not resolved since a peak around the 22nd of January appears to hit 1000 attacks per day. And, of course, it only takes one attack to ruin your day.

So, definitely good that law enforcement is on this and is taking down these DDoS for hire services since they commoditize DDoS attacks. But it's clear that there is still plenty of firepower left... to which Fosstodon can attest.

TikTok Insanity

Last Thursday, the headline in Gizmodo read: *"We Found 28,000 Apps Sending [their data to] TikTok. Banning the App Won't Help."* I'll just share the beginning of Gizmodo's very long article since it contains the most useful information. Gizmodo writes:

> [President] *Joe Biden gave federal agencies 30 days to remove TikTok from government devices earlier this week.* [Meaning last week] *Until now, most politicians intent on punishing TikTok have focused solely on banning the app itself, but, according to a memo reviewed by Reuters,* ***federal agencies must also "prohibit internet traffic from reaching the company."*** *That's a lot more complicated than it sounds.* [And I'll interject here, a lot less complicated than Gizmodo thinks. We'll get to that in a minute.] *Gizmodo has learned that tens of thousands of apps—many which may already be installed on federal employees' work phones—use code that sends data to TikTok.*
>
> *Some* ***28,251*** *apps* [unclear apps for which platforms] *use TikTok's software development kits, (SDKs), tools which integrates apps with TikTok's systems—and send TikTok user data—for functions like ads within TikTok, logging in, and sharing videos from the app. That's according to a search conducted by Gizmodo and corroborated by AppFigures, an analytics company. But apps aren't TikTok's only source of data. There are TikTok trackers spread across even more websites. The type of data sharing TikTok is doing is just as common on other parts of the internet.*
>
> *The apps using the TikTok SDK include popular games like Mobile Legends: Bang Bang, Trivia Crack, and Fruit Ninja, photo editors like VSCO and Canva, lesser-known dating apps, weather apps, WiFi utilities, and a wide variety of other apps in nearly every category. The developers for the apps listed above did not immediately respond to a request for comment.*

Okay. So there's two parts to this whole mess. The first is, what are we actually trying to do here, and the second is, is it actually possible?

First, to give the whole idea of banning TikTok some perspective, as well as a bit of an updated reality-check about the present nature of consumer tracking on the Internet, the non-partisan Brookings Institute titled their commentary from the middle of last month: *"TikTok bans won't guarantee consumer safety"* I've grabbed the end of that long piece, where they wrote:

> ***Are TikTok's data practices different from other companies?***
>
> *Several experts have already argued that TikTok bans won't make Americans safer. One reason is that much of the information collected by TikTok is like that compiled by many companies that host consumer-facing products. The app undoubtedly has information on which videos users have watched, comments they have made about those items, and their geolocation while watching the videos, as well as both users' and their friends' contact information, but that is true for nearly all digital platforms and e-commerce sites around the world.*

> *It also is the case that digital firms compile data on users, and many buy and sell consumer data via third-party vehicles. It has been estimated that leading U.S. data brokers have up to 1,500 pieces of information on the typical American, and that both domestic and foreign entities can purchase detailed profiles on nearly anyone with an online presence. Even with aggregated data, it is possible to identify specific individuals through a relatively small number of attributes, with some research estimating that "99.98% of Americans" could be de-anonymized from relatively small datasets. Still, what sets TikTok apart are the amount and type of trackers they use.*
>
> *According to a 2022 study utilizing Apple's "Record App Activity" feature, TikTok utilizes over twice the average number of potential trackers for social media platforms. Almost all of these trackers were maintained by third parties, making it harder to know what TikTok is doing with the information they collect.*
>
> *If concerns about TikTok are around the compromising of personal information with government authorities, either in China or elsewhere, there are many firms both within the U.S. and abroad that have been accused of the same. For example, a former Twitter employee was convicted of acting as a foreign agent for Saudi Arabia, providing confidential information from that platform about dissidents to foreign officials. [Consumer] geolocation data are routinely bought around the world by data brokers and repackaged for sale to advertisers, governments, and businesses around the world.*
>
> *Regarding concerns that Chinese companies operating within the U.S. are beholden to Chinese laws, the same can be said of American companies that operate in China. Some observers have expressed worries about Tesla vehicles being made in China for some of the same reasons, and what the company may have to do to maintain good relations with Chinese officials. Furthermore, if the criterion for bans based on national security is access to users' confidential information, there is a long list of American and foreign companies that face security challenges via their Chinese operations. As examples, many digital products sold domestically are made in China. And a wide variety of smart appliances, pharmaceuticals, personal protective equipment, computer chips, and other products are assembled there.*

Right. What's actually developed over time is a rich and deeply interdependent ecosystem. And there are myriad companies collecting and selling data on everyone who is using the Internet. Our illusion of true anonymity is exactly that — an illusion. As 3rd-party cookies once did, most of this operates under the radar. While unseen it is still utterly ubiquitous. It is everywhere.

So that leaves the question, assuming that the governments decide that they're going to blacklist TikTok, is that even **possible**?

As we know, IP addresses are readily changed, but the domains used by DNS lookups are generally hardcoded into apps and trackers. That means that DNS lookups are TikTok's Achilles heel. I did a bit of research and I identified five domain name roots which often also have subdomains. So some wildcard matching would be necessary. There are also two Akamai CDN domains. But taken together, those would appear to be all of the domains currently in use by TikTok. They are:

*.tiktok.com
*.tiktok.org
*.tiktokv.com
*.tiktokcdn.com
*.musical.ly
*.p16-tiktokcdn-com.akamaized.net –and– *.TikTokcdn-com.akamaized.net

So, if federal agencies were to locally configure their network DNS to blackhole those domains and their subdomains, perhaps returning 0.0.0.0 or 127.0.0.1 or maybe pointing them to a local server, once local device DNS caching expires, and a quick DNS DIG indicates that those domains are running with quite short TTL expirations, all traffic of any sort bound for TikTok would lose its destination IP.

On the question of whether this would be a good thing to do I have no opinion. Whereas the technology is interesting, the politics is not. Tensions are clearly on the rise with China, so I suppose that nationalism and protectionism are bound to rise as well.

But I think more than anything the technology lesson we take from this is that there is an incredible unseen and largely unappreciated underground of activity that few Internet users appreciate. Out of curiosity, I went over to MSNBC's website at msnbc.com and uBlock Origin lit up, counting that single website homepage causing my browser to connect to 38 other domains; and foxnews.com connected my browser to 51 other domains. 38 and 51! I'm sure that few of are directly affiliated with either property, and how many CDNs do they need? Since most people have no idea what's going on, why not load up with revenue-generating trackers?

I doubt that TikTok cares at all about the loss of connectivity to federal government networks. And federal employees who want to continue to use TikTok on their own devices while within those networks can simply switch to their cellular provider for unfettered Internet access.

If the United States government's actual goal is to protect its citizens from the data collection of a Chinese state-owned and controlled entity then blocking all TikTok traffic at US borders is going to be necessary ... and every time Russia or other repressive regimes does the same, we make fun of them.

Illegal Warrantless Surveillance

A report from the Office of the Inspector General for the Department of Homeland Security titled "Secret Service and ICE Did Not Always Adhere to Statute and Policies Governing Use of Cell-Site Simulators - Law Enforcement Sensitive (REDACTED)" found that the US Secret Service and the US Immigration and Customs Enforcement (ICE) have not obtained court orders for multiple operations in 2020 and 2021 where they deployed cell-site simulators (stingrays) to intercept mobile communications. The report found that one Secret Service field office had deployed stingrays on multiple occasions on behalf of a local law enforcement agency without obtaining court warrants. The report also found that the ICE "did not believe court authorization was required" for some of its operations. Furthermore, the report also found that neither the USSS nor the ICE were documenting operations related to supervisory approval and data deletion procedures.

So, yeah, we need to be careful with the technology that we make generally available. And those who have the technology need to know that their use of it is being monitored and that they will be held accountable. This demonstrates the potential for abuse once loopholes are placed into our supposedly secure and private communications. We'd like to believe that only those holding valid court orders would have access to private communications. But experience suggests otherwise.

Strategic Objective 3.3
Dated March 2023, last week, the Biden administration published its 39 page National Cybersecurity Strategy. I haven't had time to go through the entire document, but if it appears worthwhile I'll likely cover it in additional detail next week. But one section of the document in particular was brought to my attention by Mark Fishburn, a listener of this podcast who knew I’d find it interesting. And when I share it, you'll know why.

That section is *“Strategic Objective 3.3”* labeled *“Shift liability for insecure software and services.”* Ooooooo! Get a load of what is now part of the United States official national cybersecurity strategy. Listen carefully to what it says:

> *Markets impose inadequate costs on—and often reward—those entities that introduce vulnerable products or services into our digital ecosystem. Too many vendors ignore best practices for secure development, ship products with insecure default configurations or known vulnerabilities, and integrate third-party software of unvetted or unknown provenance. Software makers are able to leverage their market position to fully disclaim liability by contract, further reducing their incentive to follow secure-by-design principles or perform pre-release testing. Poor software security greatly increases systemic risk across the digital ecosystem and leave American citizens bearing the ultimate cost.*
>
> *We must begin to shift liability onto those entities that fail to take reasonable precautions to secure their software while recognizing that even the most advanced software security programs cannot prevent all vulnerabilities. Companies that make software must have the freedom to innovate, but they must also be held liable when they fail to live up to the duty of care they owe consumers, businesses, or critical infrastructure providers. Responsibility must be placed on the stakeholders most capable of taking action to prevent bad outcomes, not on the end-users that often bear the consequences of insecure software nor on the open-source developer of a component that is integrated into a commercial product. Doing so will drive the market to produce safer products and services while preserving innovation and the ability of startups and other small- and medium-sized businesses to compete against market leaders.*
>
> *The Administration will work with Congress and the private sector to develop legislation establishing liability for software products and services. Any such legislation should prevent manufacturers and software publishers with market power from fully disclaiming liability by contract, and establish higher standards of care for software in specific high-risk scenarios. To begin to shape standards of care for secure software development, the Administration will drive the development of an adaptable safe harbor framework to shield from liability companies that securely develop and maintain their software products and services. This safe*

> *harbor will draw from current best practices for secure software development, such as the NIST Secure Software Development Framework. It also must evolve over time, incorporating new tools for secure software development, software transparency, and vulnerability discovery.*
>
> *To further incentivize the adoption of secure software development practices, the Administration will encourage coordinated vulnerability disclosure across all technology types and sectors; promote the further development of SBOMs; and develop a process for identifying and mitigating the risk presented by unsupported software that is widely used or supports critical infrastructure. In partnership with the private sector and the open-source software community, the Federal Government will also continue to invest in the development of secure software, including memory-safe languages and software development techniques, frameworks, and testing tools.*

Okay. Now, no strategy is law. A strategy is only that. And major software publishers have strong lobbying arms in Washington where legislative votes are available to the highest bidder. So nothing here is actionable, and there will be a great deal of pushback against any sort of weakening of today's current blanket contractual protections. What caught me off guard was the precision of understanding about the nature of this problem. That one sentence: *"Software makers are able to leverage their market position to fully disclaim liability by contract, further reducing their incentive to follow secure-by-design principles or perform pre-release testing."* The writing is not yet even on the wall, but it's obviously in some people's heads, and it just got written down in the official national cybersecurity strategy. It's obvious that others have noticed the same irresponsible attitudes toward critical software security that we've discussed here.
https://www.whitehouse.gov/wp-content/uploads/2023/03/National-Cybersecurity-Strategy-2023.pdf

GitHub Secret Scanning

Back in December, GitHub announced the public beta of their free secret scanning alerts across public repositories. By "secret scanning" they mean that GitHub will proactively scan all code submitted for the inadvertent inclusion of any secrets — like an admin password that gets left in the code by mistake. It's a very cool idea. And since its initial release in beta, more than 70 thousand public repositories have turned on secret scanning alerts and have uncovered thousands of leaking secrets.

So, as of one week ago, last Tuesday, GitHub's secret scanning's alert experience is generally available and free for all public repositories. GitHub users can enable secret scanning alerts across all the repositories they own to notify them of any leaked secrets across their entire repository history including code, issues, description, and comments.

GitHub secret scanning works with more than 100 service providers in the GitHub Partner Program. In addition to alerting users, they will notify their partners when one of their secrets is leaked. With secret scanning alerts enabled, regular users will now also receive alerts for secrets where it's not possible to notify a partner–for example, if self-hosted keys are exposed–along with a full audit log of actions taken on the alert.

One example of this in practice is a DevOps Consultant and Trainer @rajbos, who enabled secret scanning on approximately 14 thousand repositories and discovered over one thousand secrets. Rob remarked, “My research proves the point of why everyone should have secret scanning enabled. I have researched 14 thousand public GitHub Action repositories and found over one thousand secrets in them! Even though I train a lot of folks on using GitHub Advanced Security, I found secrets in my own repositories through this.”

I don't see any downside to this and I'd recommend that all of our GitHub using listeners enable these secret scanning alerts.

CISA's Covert Red-Team

Also last Tuesday, CISA revealed the somewhat bracing results of a secret red team exercise they carried out against the network of a “large (and unnamed) US critical infrastructure organization with a mature cyber posture.” During that exercise they “obtained persistent access to the organization's network, moved laterally across multiple geographically separated sites, and gained access to systems adjacent to the organization's sensitive business systems.” CISA says that on at least 13 separate occasions its red team triggered “measurable events” — it wasn't clear whether this was deliberate or unavoidable — that should have gotten it caught. But in every case the organization failed to detect these actionable events. As I noted, CISA officials declined to name the organization but they did say that although they did manage to get in, they found the organization had good cybersecurity policies in other parts of its network, such as up-to-date and hardened perimeter infrastructure and good password policies. So how did CISa's red team gain its initial entry? Phishing. Today's most difficult to corral cyber weakness. People on the inside are on the inside, and it's just too easy for someone to inadvertently allow a bad guy in.

What's left?

I titled this short bit of news “What's left?” after reading that Russia had formally, legally banned the use of all foreign messaging applications inside Russian financial institutions and state-owned companies. The law, which entered into effect this month, outlaws the use of apps such as Discord, Microsoft Teams, Skype for Business, Snapchat, Telegram, Threema, Viber, WhatsApp, and WeChat. The page announcing this is in Russian and it lists those. Notably missing is Apple's iMessage and Android's Messages — both which are E2EE. So this legislation, which specifically enumerated those nine 3rd-party messengers doesn't mention the two mobile platform's native messengers. So perhaps that's the answer to my rhetorical question “what's left?”, or perhaps those are both implied. The news was that all foreign messaging applications were banned. So perhaps Russia has some State owned or trusted messaging apps that no one else is going to trust or want to use? https://rkn.gov.ru/news/rsoc/news74672.htm

What's old is new again

As we've noted before, CISA is maintaining a list of Known Exploited Vulnerabilities. In fact, KEV is the common abbreviation for that growing list. And speaking of growing, during last year the size of this catalog of known exploited vulnerabilities very nearly tripled in size, jumping from 331 entries at the start of 2022 but finishing out last year with 868 individual vulnerabilities known to have been seen being actively exploited. Now, here's the interesting part: Although this would suggest that new bugs are being exploited, a look at the issuance dates of the catalog's CVEs shows that the vast majority of new CISA KEV entries are for older vulnerabilities that companies failed to patch for years, all which came under attack last year. The oldest of those was the exploitation of a bug patched 21 years ago, back in 2002.

TCG TPM vulnerabilities

Remember when everyone used Intel's sample UPnP implementation code as their actual production code, apparently without ever actually looking at the code, and even when the comment header block at the top of the code loudly stated that this was sample code ONLY and should never be used for production? And that as a result, the entire industry suffered as a whole from a widespread vulnerability in that sample code which everything had in their routers.

Well, history is sort of repeating. This time it's a pair of widespread vulnerabilities which have befallen multiple vendors' Trusted Platform Module TPM code. It's not quite the same as the UPnP debacle, since these vendors were using the TPM Reference implementation library which should have been okay. But this is another example of the danger of monocultures and why we're more healthy if, as another example, we keep browsers other than Chromium alive.

In any event, researchers at Quarkslab discovered two buffer overflow vulnerabilities in libraries implementing the TPM 2.0 security specification. The vulnerabilities would allow an attacker who could gain access to the TPM's command-line interface to leverage the vulnerabilities to corrupt the TPM's memory and access sensitive information handled by the TPM, such as encryption keys... which is exactly what all of the TPM's fancy technology is supposed to prevent. Patches for this were released at the end of February and since many vendors all directly implemented the same reference library, the TPM implementations from IBM, the Trusted Computing Group themselves, RedHat, SUSE Linux, and others are all affected.

WordPress “All In One SEO”

Just a quick note to any of our listeners who manage WordPress sites. More than three million WordPress sites which are currently running that “All In One SEO” plugin will need to be updated to resolve a set of vulnerabilities that could be used to hijack sites. As is often the case, the troubles were found and reported by the researchers at WordFence. And as I've also noted before when we were talking about the inherent danger of 3rd-party developed WordPress add-ins which appear to be having constant security problems, adding protection from WordFence, if it fits your budget, would seem like a large and worthwhile ounce of prevention.

Miscellany

Russia fines Wikipedia

The Russian government has fined the Wikimedia Foundation, the organization behind the Wikipedia portal, 2 million rubles (~$27,000) for failing to delete "misinformation" about the Russian military and its invasion... oops... special operation in Ukraine. According to Reuters, this is the 3rd time Wikipedia has been fined by Russia since the country's invasion of Ukraine. Wikipedia said the recent fine was related to articles on its Russian language portal related to Russian Invasions of Ukraine (2022), Battle for Kyiv, War Crimes during the Russian Invasion of Ukraine, Shelling of Mariupol Hospital, Bombing of the Mariupol Theater, and the Massacre in Bucha.

I noted that last November, when the second of the three fines was levied, the same fine of 2 million rubles was set, but back then those 2 million rubles were worth $33,000 dollars. Today, only $27,000. So those Rubles appear to be slipping against the dollar.

The Wikimedia Foundation has stated that they refuse to back down and remove what it fact-based multiply-sourced verified truth. And they have been appealing these fines in Russian court. So far they've only had one successful ruling.

.

A Fowl Incident

When I read from a Chick-fil-A data breach report submitted to the U.S. state of Maine's Attorney General, which disclosed that 71,473 Chick-fil-A account holding customers had had their accounts breached through a credential stuffing attack, I was skeptical. And, I'm at least still a bit confused. That number just seems far too huge to be the result of what amounts to opportunistic, previous breach driven, **guessing** of account usernames and passwords. And, really, why would some random hacker be going out of their way to compromise the accounts – and more than just a few – of Chick-fil-A customers? Why not Chase, Bank of America or TD Ameritrade? I mean Chick-fil-A ... really? Those are the customer accounts that you choose to penetrate? You must really have a thing for chicken!

So, I don't know. It doesn't make sense to me. That number seems too big — 71,473 individual Chick-fil-A customer accounts each which would have taken effort to compromise. And what do you get for all that effort? Apparently some Chick-fil-A redeemable loyalty reward points.

We've been talking about this form of attack recently. But just to reiterate since this is where details matter, Credential Stuffing attacks are the reuse of username, eMail and passwords, leaked from previous online service breaches, which are then being used to blindly guess login credentials at other unaffiliated websites. The point is, and as I've said before, all rational logic suggests that this should be an extremely low yield attack, meaning that in order to correctly guess the logins for 71,473 individual Chick-fil-A customers it would be necessary to wrongly guess a gazillion other times.

When I initially saw that large number, my first thought was that it couldn't actually be a true credential stuffing attack. And for the record, I've never been a fan of the term credential stuffing. The industry could have come up with something better like a "Credential Reuse Attack." But credential stuffing it is, and as long as we all know that it's a credential reuse attack that's fine.

So for that many accounts to be successfully attacked, I'm suspicious of whether Chick-fil-A might have earlier lost control of their own customer account data and that the leaked information itself was now being used to login and attack their customers. This would convert the attack from *"surprisingly successful low yield against a bizarre target"* to *"unsurprisingly high yield against the only available target."* If you've somehow acquired Chick-fil-A's customer logon data that's going to be your only target.

On the other hand, if we assume that this was actually a true credential stuffing attack, and putting aside for the moment the question, *"out of a universe of equally suitable targets, why would an attacker choose Chick-fil-A?"*, the Chick-fil-A's disclosure to various states' Attorneys General did state that the attack took place over a two month span from last December 18th, through its date of discovery, February 12th, last month.

That's 56 days during which 71,473 Chick-fil-A customer accounts were breached, at an average rate of 1,276 successful account breaches per day.

So the logistics of such a credential stuffing attack would be that an attacker has a massive database of prospective login credentials which they, for some reason, choose to aim at Chick-fil-A's website. And in blind account credential guessing, they pour this massive database through the website's clearly unrestricted authentication front-end at presumably some massive rate. Perhaps the attack was distributed with the massive database spread among many attacking clients in order to increase the overall rate of credential guessing because, of course, modern websites are able to simultaneously entertain many incoming connections.

But in a situation where there's no apparent oversight, monitoring or throttling of failed authentication of any kind, there's no real need to distribute the attack. Nothing prevents a single attacking machine, or only a few, from each establishing their own hundreds or thousands of simultaneous login sessions with a single Chick-fil-A server. That works too.

If we accept Chick-fil-A's claim that this was truly blind credential guessing from a database of previous completely unaffiliated websites, then the per-guess yield **had** to be quite low. A very low per-guess yield, meant that the total number of guesses had to be massive. So this, in turn, means that the Chick-fil-A website servers raised no alarm of any kind while, starting last December the 18th, their incoming connection rate skyrocketed as millions of attempts to login there were suddenly failing. The authentication failure rate had to be astronomical yet nothing at their end took notice.

Once Chick-fil-A somehow became aware of the attack – presumably when a sufficient number of their customers complained of account tampering – they were able to identify exactly which of their customers had their accounts breached in this manner, while at the same time being completely unaware of the ongoing attack for 56 days. So the reports are that they've repaired any damage done to those customers, made them whole again, and instructed them to change their Chick-fil-A passwords and also anywhere else that they were reusing the same password.

So we have two take-a-ways from this — fowl — incident:

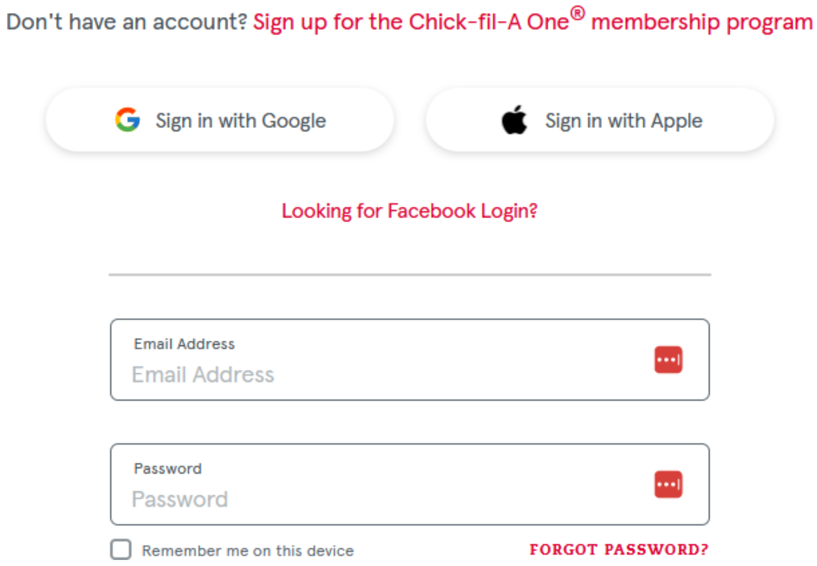
First: If we accept this on its face, that this was an unaffiliated, if somewhat bizarre attack, then it could only have succeeded as it has, due to the continuing presence of widespread and stubborn reuse of user passwords across sites. And that's not yesterday, that's today right now.

This means that, more than ever, it is rapidly becoming truly imperative for the uniqueness of passwords to be enforced across all of a user's online accounts. We've learned from our own podcast audience's reports, in the wake of the LastPass vault debacle, that updating passwords can be a slow, tedious and laborious process. But now more than ever before, if your password manager offers a global password reuse audit, as many do, it's important that you allocate some time to begin replacing any duplicated passwords, and also in strengthening any existing passwords that do not contain sufficient state-of-the-art entropy.

I'm sure that this message is largely redundant and unnecessary for this podcast's audience. But the success of this Chick-fil-A attack informs us that everyone listening needs to share their understanding of this with everyone **they** know. Because, if we're to believe Chick-fil-A, it's clear that the reuse of passwords remains widespread across the Internet and we're starting to see an evolving epidemic of this new form of surprisingly successful attack.

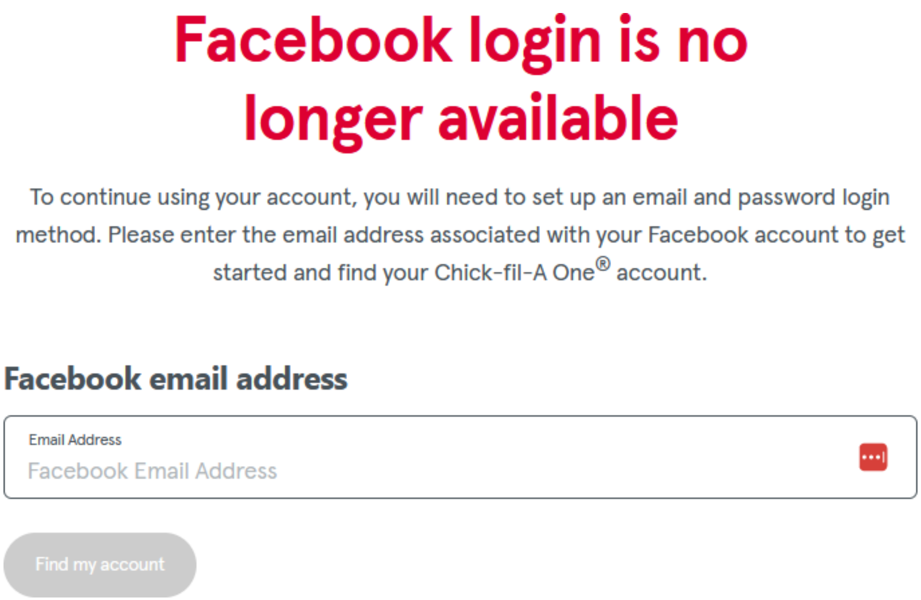
This brings us to the second and more interesting technology question: *"What can websites do in the face of what appears to be an escalating and approaching epidemic of opportunistic low-yield credential reuse attacks?"*

When a user logs into a website, their browser requests the login page which presents a form to be filled out. I got to this point in preparing the notes for today's podcast when I thought that I ought to go over to Chick-fil-A's website to see whether they presented the login as a single form or multiple staged forms. And what did I find?



Don't have an account? Sign up for the Chick-fil-A One® membership program

There are a pair of standard login credentials prompting for eMail Address and Password. But there's also a "Sign in with Google" and a "Sign in with Apple" and in the spot where there was once a "Sign in with Facebook" the page now reads "Looking for Facebook Login?" Isn't that interesting. Clicking that link takes existing Chick-fil-A customers to a page that says:



Facebook login is no longer available

To continue using your account, you will need to set up an email and password login method. Please enter the email address associated with your Facebook account to get started and find your Chick-fil-A One® account.

Facebook email address

THAT's interesting! Though we can't know for sure when Chick-fil-A's probably most popular "Login with Facebook" OAuth2 option was removed, anyone doing some forensic post-incident digging would be skeptical of the coincidence given the recent attack. Unfortunately, since their login page is algorithmically generated and thus cannot be brought up with a static URL, the Internet archive's Wayback machine cannot be queried to see when "Login with Facebook" was removed as an option. But given that "Login with Google" and "Login with Apple" are both still present, and that Chick-fil-A's replacement for Facebook login is migrating users from Facebook login to native login, I'd bet a month's pay that we now know what actually happened.

It wasn't a generic credential stuffing attack...

It was specifically a **Facebook** credential reuse attack.

And now we understand why Chick-fil-A was the target: It was because they offered the popular "Login with Facebook" option. Somebody, somewhere, has a boatload of in-the-clear Facebook login username and password credentials and over the course of several months they explored the intersection of that stolen Facebook credential set with the Chick-fil-A customer account database. There were 71,473 Chick-fil-A customers whose Facebook credentials are part of the stolen Facebook dataset and as a consequence their Chick-fil-A accounts were breached.

When you think about it, if you had a trove of valid Facebook login credentials what's the most valuable thing you could do with them? Who wants to login to those random users' Facebook accounts? There's no money in that. No. What you want to do is leverage the increasingly pervasive use of OAuth2 account login to compromise the myriad **other** accounts belonging to those hapless Facebook users who have chosen to identify themselves to other website properties only through their Facebook credentials. Now you have a valuable and potentially widespread attack.

If this is true, as seems extremely likely given all of the evidence, we have a perfect example of why the use of OAuth2 for logging in with a single common credential poses a significant threat. What are we loudly telling everyone who will listen about their passwords? *"Do not use the same username and password to login to multiple websites."* Right? Everybody knows that. But the use of OAuth2, which is now being actively promoted due to its extreme ease of use, is a direct contravention of that advice. It is the explicit reuse of a single set of credentials across a great many website properties. And while there may not be much value to an attacker to use a stolen Facebook credential to login to that user's Facebook account, there might well be significant value in their ability to log in **everywhere else** that Facebook user used their Facebook credentials to create non-Facebook accounts.

So this begs the question, how are in-the-clear Facebook account credentials harvested? And we quickly find an example. Bloomberg News, Technology & Cybersecurity article from October 7th, 2022 — just two months before we're told the attack on Chick-fil-A began — has the headline: "Facebook Is Warning 1 Million Users About Stolen Usernames & Passwords" the company found more than 400 problematic Android & iOS apps, games & photo editors tricked users into providing credentials. Gabbing one paragraph from Bloomberg's coverage, they wrote:

> *A typical scam would unfold, for example, after a user downloaded one of the malicious apps. The app would require a Facebook login to work beyond basic functionality, thus tricking the user into providing their username and password. Users could then, for example, upload an*

> *edited photo to their Facebook account. But in the process, they unknowingly compromised their account by giving the author of the app access.*

So what the Chick-fil-A attack probably reveals is the new use to which stolen Facebook credentials are now being put. Again, obtaining access to a Facebook user's account is far less profitable than being able to login to any and all of that user's non-Facebook accounts where they may have something of value.

Most users have not the faintest clue how all of this technology we've given them to use, works; no idea whatsoever. So when they're asked by a spiffy-neat app they just downloaded to provide their Facebook login credentials so that the app can link to and synchronize with their Facebook account, they don't know any better. Why would they?

And most users would not understand that in the process, thanks to the fact that they have also been using "Login with Facebook" everywhere they possibly can because it's so much easier to do that, that they are also giving away access to their accounts at all of those other websites as well.

We've often noted that the use of OAuth2 is an inherent privacy compromise since that 3rd-party – Facebook, Google, Apple or whomever – knows everywhere you're using them to login, and where you are at the time. But this Chick-fil-A attack with its subsequent removal of its probably most popular "Login with Facebook" option, reveals a much darker side of the widespread use of this form of Single Sign-On solution. All of the common wisdom urges users to avoid credential reuse... but credential reuse is exactly that Single Sign-On promotes. And thus, this Fowl Incident as highlighted a worrisome truth behind a growing trend.