Poisoning Akamai

Description: This week we examine a puzzlingly insecure implementation by Microsoft in Teams' design and at their complete rewrite of Microsoft Defender SmartScreen. Roskomnadzor strikes again, and Exchange Server is again under serious attack with a new zero-day. Cloudflare introduces Turnstile, their free CAPTCHA improvement. Google publishes a fabulously engaging six-video YouTube series under the banner "Hacking Google." We'll then spend some time sharing and replying to listener feedback before we examine a breathtaking flaw that was discovered in Akamai's global CDN caching, and what became of it.

High quality (64 kbps) mp3 audio file URL: http://media.GRC.com/sn/SN-891.mp3
Quarter size (16 kbps) mp3 audio file URL: http://media.GRC.com/sn/sn-891-lq.mp3

SHOW TEASE: It's time for Security Now!. Steve Gibson is here. Another flaw in Teams. You might want to know about this before you let somebody use your computer. We'll talk about that Exchange Server zero-day. They're at it again. A brand new way to do CAPTCHAs, thanks to Cloudflare. I think we'll like this one a little bit better. And then an Akamai flaw that they didn't want to pay security researchers to fix. It's all coming up next on Security Now!.

Leo Laporte: This is Security Now! with Steve Gibson, Episode 891, recorded Tuesday, October 4th, 2022: Poisoning Akamai.

It's time for Security Now!. I know you've been waiting all week long. Mr. Gibson is here. He is prepared. He is ready to talk about the world of security. Steve Gibson from @SGgrc fame.

Steve Gibson: Is that you looking at the screen, so I can, like, see through the magic mirror?

Leo: I look back at you. It's an affectation that performers often use. I'm actually not even looking at you at all.

Steve: Our audio listeners are like, what are those two clowns talking about?

Leo: In order to look at you, I'd have to turn my back on our audience. And I don't want to do that.

Steve: Oh, that's not good.

Leo: So what I do is I look across to the left as if...

Steve: Didn't you study drama at Yale or something?

Leo: Yeah, that's where I learned this. Eye line, yeah. So it kind of looks like I'm looking at you, and in fact I'm not. But you're to the - it's complicated.

Steve: I'm in a computer, so that works.

Leo: What's up, Steve?

Steve: So we've got the first podcast for October, the fourth quarter of 2022, titled "Poisoning Akamai."

Leo: Oh, boy.

Steve: And, oh, it's really fun. Not only what two youngsters figured out, and the fact that this all happened without many people knowing about it earlier this year, but what the consequences of this could have been, and also what happened after they went to Akamai to explain. So really interesting. But we're going to first examine a puzzlingly insecure implementation by Microsoft in Teams design, and also look at Microsoft's complete rewrite of Microsoft Defender SmartScreen. And we'll talk about that. Also Roskomnadzor strikes again.

Leo: Ooh.

Steve: Yes. It's October, so that's good.

Leo: Spooky season.

Steve: Yes. And Exchange Server is again under serious attack with a new zero-day which Microsoft knows about, hasn't fixed yet, and their suggestion mitigations have been worked around. So we're hoping for an actual fix soon because it's a nasty one.

Cloudflare has introduced Turnstile, which is their free and much improved CAPTCHA which they're offering, even to non-Cloudflare users. We've got to talk about that. Also Google just yesterday published a fabulously engaging six-video YouTube series under the banner "Hacking Google." So I have that to talk about. And that's the tweet that you saw that I sent out. I was so fascinated by that I forgot to tell everybody about the podcast. Anyway, I'll get around to that when you tell us about our sponsor here in a minute. Then we're going to spend some time sharing and replying to some listener feedback before we examine, as I said, a breathtaking flaw that was discovered in

Akamai's global content delivery network caching system, and what became of it. And of course we've got a Picture of the Week that's...

Leo: That's bizarre. Weird. It's wacky. Steve, I didn't have to go to Zip Recruiter to find you. I had to have a ZIP drive to find you; right?

Steve: That's exactly right.

Leo: Click of death, way back in 1999 or something like that.

Steve: Oh, yeah.

Leo: We didn't know about Pictures of the Week back then. We should have probably had those on The Screen Savers. These are good. I like this feature.

Steve: They were fun.

Leo: Yeah.

Steve: So this one is not really anything to do with security, but it's just so fun. So it falls under the broad banner, one of my favorite banners, "What could possibly go wrong?"

Leo: What could possibly go wrong? I like to do it with you.

Steve: Okay. We'll do it from now on. So this posting says: "I have a 5,000-gallon aboveground pool in my basement. It feels nice down there, but the water is freezing. I have a tiny ass pump on it right now that kind of flows water, but I'm wanting to heat the whole pool to a reasonable temperature." And then we see a picture of this monstrosity. So we're looking at a basement with apparently a door open that is where the stair is leading down into the basement end, and this huge pool that looks like it's about to burst. But I think that's probably the way they're supposed to look because you're going to have an awful lot of water pressure pushing the bottom skirt of the pool outwards. And it's being, like, kept from expanding endlessly by a collar at the top. But boy, you know, this is 5,000 gallons of water that only has one - it's like it's trying to leave the pool as hard as it can.

Leo: I have to go to WolframAlpha for this one. What is the water pressure, 5,000 gallons of water stored in - what would you say the area of that? Hundred square feet?

Steve: Maybe 50 square feet?

Leo: Fifty, okay.

Steve: A hundred square feet, yeah, maybe.

Leo: We'll be generous, yeah.

Steve: Anyway, so he says, he gives us the picture, marking it "for reference." And he says: "I want something cheap that won't melt the pool as it is rubber." He says, and then I love this: "I know I'm not the only person out there with a pool in their basement." And I'm thinking, maybe you are.

Leo: I'm thinking that pool for sure.

Steve: Oh. I don't know. Anyway, just a great Picture of the Week.

Leo: According to WolframAlpha, 5,000 gallons of water stored in a 100-square-foot whatever, generates 5.066 x 10^10 square feet, foot square gallon pascals. Don't know what any of that means. It's bad.

Steve: How about millimeters of mercury? That would be a pressure measurement that might be...

Leo: Yeah, kilogram meters to the fourth per second squared is another unit.

Steve: Ah, no. No.

Leo: I'll have to convert that. I'll get back to you.

Steve: Yeah. Elon Musk uses those measurements for his SpaceX program.

Leo: Yeah, yeah. I'm not a rocket scientist.

Steve: Anyway, just another fun one to share.

Leo: That's hysterical. I hope he found something to heat that pool without melting the rubber.

Steve: Yeah.

Leo: We were thinking maybe a bitcoin mining device down there may be helpful. Yeah?

Steve: Do they have submersible bitcoin miners?

Leo: No. Oh, I know what you need. You need a sous vide circulator in there. You could sous vide yourself.

Steve: Maybe you could just heat the environment, and of course the water would eventually warm up to the ambient temperature.

Leo: Well, the reason it's cold is it's sitting on bare dirt.

Steve: Yeah.

Leo: That's going to get cold. Anyway...

Steve: Yeah. Looks creepy.

Leo: The creepiest would be just seeing him get in it. That's what I...

Steve: I hope the whole family knows how to swim because they'd be needing that.

Leo: Oh, my god. Yes.

Steve: Okay. So three weeks ago the security firm Vectra published a report which closely examined the way Microsoft Teams manages its users' application authentication. Their report is long, and we don't need to get into the nitty-gritty to understand what's going on. So I'm just going to share two small pieces from their long report.

In their overview they explain: "In August 2022, the Vectra Protect team identified a post-exploitation opportunity allowing malicious actors with sufficient local or remote file system access to steal valid user credentials from Microsoft Teams due to their plaintext storage on disk." In other words, Microsoft Teams, after you have authenticated yourself, statically stores the authentication information, the tokens, in the local file system, there for everyone to access. So they said: "This plaintext credential management was determined to impact all commercial and GCC Desktop Teams clients for Windows, Mac, and Linux." So common to all desktop platforms, the big three.

They said: "While credential harvesting from memory" - you know, RAM - "is a common post-exploitation step, we believe that lowering the bar necessary to harvest creds down to just simple read access to the file system expands opportunities for an adversary, simplifies their task, and is particularly interesting when stolen credentials offer an opportunity to retain user access unencumbered by otherwise pesky" - one of my favorite words - "Multi-Factor Authentication speed bumps. With these tokens, attackers can assume the token holder's identity for any actions possible through the Microsoft Teams client, including using that token for accessing Microsoft Graph API functions from an attacker's system.

"Additionally, these tokens are equally valid with MFA-enabled accounts, creating an allowance to bypass MFA checks during ongoing use. Microsoft is aware of this issue, but indicated it did not meet their bar for immediate servicing. Microsoft stores these credentials to create a seamless single sign-on experience within the desktop application. However, the implementation of these security choices lowers the bar.

"Anyone who installs and uses the Microsoft Teams client in this state is storing the credentials needed to perform any action possible through the Teams UI, even when Teams is shut down. When these tokens are stolen, it enables attackers to modify SharePoint files, Outlook mail and calendars, and Teams chat files. Attackers can tamper with legitimate communications within an organization by selectively destroying, exfiltrating, or engaging in targeted phishing attacks.

"The thing that truly frightens us," they said, "is the proliferation of post-MFA user tokens across an environment. It enables subsequent attacks that do not require additional special permissions or advanced malware to get away with major internal damage. With enough compromised machines, attackers can orchestrate communications within an organization. Assuming full control of critical seats - like a company's Head of Engineering, CEO, or CFO - attackers can convince users to perform tasks damaging to the organization." They say: "How do you practice phish testing for this?"

Okay. So this is one of those "head buried deeply in the sand" issues which we've increasingly been encountering from Microsoft. The kindest way, I think, to interpret this in Microsoft's favor is to suggest that Microsoft has now structured itself so that it's deliberately cut off from the outside. Someone who has no authority or power is running interference and responds to any offering made at the foot of the Ivory Tower by incanting the phrase: "We are aware of this, and it is not a security concern." And that's as far as any inquiry goes. If history is to repeat itself, especially now that this problem is well known, there will eventually be some egregious abuse of what is obviously a totally unnecessary and easily exploitable security weakness in Teams. At that point, a wholly unnecessary emergency will ensue, and Teams will have this behavior-by-design changed.

It's totally true that having persistent and static access to a previous successful authentication creates a standing vulnerability. Perhaps that's been done so that other components such as Skype and Outlook are able to share in this authentication, as indeed they are. But in an alternative design, for example, they could share a common authentication service which then relies upon encrypted authentication tokens which would at least tie the tokens to the local machine's authentication service, perhaps its TPM, and would make them much more tricky to abuse. Instead, Microsoft has chosen for whatever reason to simply store them in a well-known location in every local machine's file system, where they're accessible, not only to all Microsoft components, but also to anyone else who might wish to abuse this implied trust.

Leo: You remember, I mean, we talked about the same thing with Google and passwords in Chrome stored in the clear. Google's response initially was, well, if somebody has physical access to your system, you're out of luck anyway.

Steve: All bets are off.

Leo: And maybe that's Microsoft's response; although remember Google did change that eventually. Right?

Steve: Right.

Leo: Yeah.

Steve: So, okay. But Vectra also had - I was going to share two things. Vectra also had another interesting piece of background to share. They gave this section the clever title "Electron - a Security Negative." It was clever, of course, because we've all agreed that electrons carry a negative charge. Vectra is suggesting that the Electron development platform carries "negative security." So here's what they explained about Microsoft Teams' use of the Electron application platform.

They said: "Microsoft Teams is an Electron-based app. Electron works by creating a web application that runs through a customized browser. This is very convenient and makes development quick and easy. However, running a web browser within the context of an application requires traditional browser data like cookies, session strings, and logs. This is where the root of this issue lies, as Electron does not support standard browser controls like encryption, and system-protected file locations that are not supported by Electron out of the box, but must be managed effectively to remain secure.

"Therefore, by default, the way Electron works incentivizes creating overly transparent applications. Since Electron obfuscates the complexities of creating the application, it is safe to assume that some developers may be unaware of the ramifications of their design decisions, and it is common to hear application security researchers bemoan the use of this framework due to critical security oversights."

So phrased another way, we might say that Microsoft Teams' choice to use the easy-to-use Electron development model, which employs JavaScript, HTML, and CSS, encourages rapid and easy application development by less experienced developers. From what Vectra said, it also sounds as though electron's browser-centric development environment encounters significant resistance when trying to do things like storing encrypted data into the file system. If that's true, then we have another case of Microsoft placing its short-term needs in front of long-term security and quality.

So, you know, let's use Electron. We'll get platform-agnostic operation, and everything will be great. Except oops, for whatever reason they just decided, I think exactly as you suggested, Leo, they've said something to the equivalent of, well, if someone's going to access your local machine, you're in trouble anyway. So not our problem. Yeah. Until it is. And I don't think we're going to have long to wait for this one.

In a posting last Thursday the 29th, titled "More reliable web defense," Microsoft explained that they had scrapped and entirely rewritten their Edge browser's built-in SmartScreen library. They said: "Starting in Microsoft Edge 103" - which, by the way, is three versions ago - "users can navigate the Internet with more reliable web defense thanks to the updated Microsoft Defender SmartScreen library that ships with Microsoft Edge on Windows. The updated SmartScreen library was completely rewritten to improve reliability, performance, and cross-platform portability. These benefits are the foundation leading up to the security improvements that will increase our ability to protect users from emerging threats."

And at the end they noted that "For enterprise customers who experience compatibility issues and need to revert to the legacy Microsoft Defender SmartScreen, we added a temporary policy called 'NewSmartScreenLibraryEnabled.'" They finished: "This policy will become obsolete in Microsoft Edge 108."

Okay. So it's unclear why anyone would have compatibility issues unless something they were doing was tripping SmartScreen false-positive responses. But since we're currently at release 106, and the option to revert will only remain available in the next release 107, any enterprise having trouble with the rewrite should address those problems quickly because, unless Microsoft is convinced to delay the removal of that temporary policy, and we've seen that happen before, so that could happen again if enterprises say, wait, wait, wait, we're not ready yet. Again, all enterprises that have needed to disable the update should look into fixing that fast.

And of course I'm a huge proponent of wholesale rewrites of anything. As we know, we haven't yet figured out how to evolve software gracefully. Part of the problem is that the various challenges software might face once released into the field are often not fully appreciated by those who initially design and write it. So the process of watching an initial release interact with the world teaches its designers a lot. The first reaction is to patch over any shortfalls in the original design. But once those patches' patches have acquired patches, it's often the case that the best solution is to quit patching the patches and take everything that is now understood about the problem and start over. So bravo to Microsoft for deciding to do exactly that, to start over. We'll never know what precipitated that decision, but Edge's users will likely be the winners.

Okay. I never pass up the opportunity to mention Roskomnadzor because it's just too fun.

Leo: I love how you do it, too. You, like, really give it the emphasis, yeah.

Steve: Oh. And, you know, I don't speak Russian, but part of this makes me wish I did because it sounds like some of it is fun. Anyway, an opportunity presented itself when Roskomnadzor added the popular music streaming platform SoundCloud to Russia's nationwide Internet block list. And Leo, I have a link here, in the story here, at the bottom of page 3 of the show notes. It's worth clicking that link just to glaze over. As I said, I've got the URL to Roskomnadzor's block list page. And thank goodness for the web's Western heritage, since at least the URL uses the Latin alphabet. I'm unable to make heads nor tails of anything on that block list page following the URL.

Leo: At least the numbers are Roman, and the CAPTCHA.

Steve: You think? My CAPTCHA, the one I got was - I don't think I could have entered it properly.

Leo: Oh, okay, well, they need to put some bicycles in there or something. All right.

Steve: That's right. Give me some stop signs and...

Leo: Now, what do I do? You know, this is where Google translate would be handy. All right. I'll figure it out.

Steve: Anyway, I just wanted to see that Russian is a weird language. So, and I don't know why Roskomnadzor is, you know, I guess maybe that's the English translation.

Leo: They anglicized it.

Steve: Yes, thank you.

Leo: Romanized it.

Steve: So the presumption is that the blockage was the result of SoundCloud's hosting of podcasts which were discussing Russia's invasion of Ukraine. Of course, you can't have any of that in a repressive totalitarian regime, so no more SoundCloud. Add yourself to the lineup.

Leo: Will Translate to the rescue.

Steve: Ah. Universal service, it says.

Leo: Yeah. So now I can - now that I've unlocked it - have I unlocked it? Let's unlock it.

Steve: I wonder why there's a caption...

Leo: I can specify a domain.

Steve: Why do you think there's a CAPTCHA on their block list?

Leo: Oh, they don't want you to use robots; right?

Steve: You wouldn't, but yeah.

Leo: So I can obtain the measures taken to restrict access to sites. So what was the site that they inadvertently...

Steve: SoundCloud.com. I assume it's dot com.

Leo: SoundCloud.com, yes, it is. Forbidden. You are forbidden. Roskomnadzor says no. No, no, no. You may not ever look at this secret. Well, it's going to...

Steve: What'll happen if you put Noodles.com in? Is that forbidden, too?

Leo: I think it's more - I don't know what it has to do with.

Steve: Maybe you click the button, and it just says Forbidden. Oh, we've got to do another CAPTCHA?

Leo: Yeah, I think for every one. Noodles.com?

Steve: Yeah.

Leo: Okay. Now I have to fill in CAPTCHA. 537863. Very secret. Okay. Forbidden.

Steve: Ah, well. There you go, Leo.

Leo: Noodles.com is forbidden. Not for you.

Steve: You cannot go. You cannot have any noodles in Russia.

Leo: No, is forbidden.

Steve: No. Roskomnadzor forbidden. Okay. So Exchange Server once again under attack. A Vietnamese group named GTSC discovered an active in-the-wild Exchange Server zero-day. And let me just preface all of this by saying this is very bad. I mean, this is - okay. Here's what they said: "At the beginning of August 2022, while doing security monitoring and incident response services, GTSC SOC" - that's their Security Operations Center - "team discovered that a critical infrastructure was being attacked, specifically to their Microsoft Exchange application. During the investigation, GTSC Blue Team experts, determining that the attack utilized" - and this is a translation from Vietnamese, so bear with, but it's pretty good - "determining that the attack utilized an unpublished Exchange Server vulnerability, i.e., a zero-day vulnerability, immediately came up with a temporary containment plan.

"At the same time, Red Team experts started researching and debugging Exchange decompiled code to find the vulnerability and exploit code. Thanks to experience finding the previous one-day Exchange exploit, the Red Team has a great understanding of Exchange's code flows and processing mechanisms. Therefore research time was reduced, and the vulnerability was uncovered quickly. The vulnerability turns out to be so critical that it allows the attacker to do a remote code execution on the compromised system. GTSC submitted the vulnerability to the Zero Day Initiative (ZDI) right away to work with Microsoft so that a patch could be prepared as soon as possible. ZDI verified and acknowledged 2 bugs, whose CVSS scores are 8.8 and 6.3 respectively."

Again, the beginning of August 2022. We're now two months downstream, right, August, September, we're in October. This has not actually been fixed yet. Since this was reported, GTSC said that they've encountered other customers who they're also charged with monitoring, whose infrastructures they're keeping an eye on, also experiencing a similar trouble. And after careful testing, they confirmed that those other systems were being attacked using the same zero-day vulnerability. To help the Exchange Server community temporarily stop the attack until an official patch is available from Microsoft, they published their coverage of their findings while responsibly excluding the information needed to recreate the attack.

The exploits caused Exchange Server to download a malicious DLL whose code is then injected into the always present and busy svchost.exe process in Windows. Once that's done, the DLL is started, and it phones home to the machine at the IP address 137.184.67.33. I imagine that changes from instance to instance, you know, DLL instance to DLL. A simple and effective RC4 cipher is used with a key chosen at runtime to then encrypt the communications back to the command-and-control server.

The GTSC folks also explained. They said: "While providing SOC service to a customer, GTSC Blue Team detected exploit requests in IIS logs with the same format as the ProxyShell vulnerability." And in fact this is now - I just saw this. I wanted to check just before the podcast to make sure that I hadn't missed any updates, like Microsoft had just patched or just pushed an update. But there's still nothing from Microsoft. This is being called informally ProxyNotShell vulnerability, instead of the ProxyShell Vulnerability, because it is definitely a variation on that theme.

In the show notes, for what it's worth, I share an example of the URL that is in there. They said: "Checking other logs, we saw that the attacker can execute commands on the attack system. The version number of these Exchange servers showed that they were already running with the latest update, so an exploitation using the actual ProxyShell vulnerability was impossible. Thus the Blue Team analysts confirmed that this was a new zero-day remote code execution vulnerability under active exploitation."

So I've got links in the show notes. There are indications of compromise published and available, if anyone wants to make sure their Exchange Server instance hasn't already been made a victim. And there are mitigation steps that can be taken. Last Thursday Microsoft publicly acknowledged the trouble with their own posting titled "Customer Guidance for Reported Zero-Day Vulnerabilities in Microsoft Exchange Server." I've got a link in the show notes.

The trouble is, the mitigation which was first proposed by the GTSC people appears to be what Microsoft has copied and is echoing. It recommends basically adding a pattern-matching "Rewrite" rule to the IIS web server which hosts Exchange. But in an update from GTSC just yesterday, October 3rd, they noted, and they said: "After receiving information from Jang" - whose Twitter handle is @testanull - "we noticed that the regex used in the Rewrite rule could be bypassed." And they then link to a YouTube demo.

So I read that to mean that Microsoft's official proposed mitigation can be bypassed. And in fact there is an article just posted from Bleeping Computer confirming that Microsoft's mitigation of this zero-day remote code execution vulnerability for Exchange Server can be bypassed. So things are really not good. Let's hope that Microsoft gets a permanent fix for this problem published soon, and that all Exchange Server users jump on getting their systems updated. I don't know of anything else that we can do in the meantime.

So last Wednesday our friends at Cloudflare posted the formal announcement of the availability of some of the work they've been focused upon this past year. This has been some time in coming. They've been working on it, and it's now available. Their posting was "Announcing Turnstile, a user-friendly, privacy-preserving alternative to CAPTCHA."

So our long-time listeners know that through the years we first introduced the abbreviation CAPTCHA, standing for Completely Automated Public Turing test to tell Computers and Humans Apart.

Leo: I think it was Carnegie Mellon that did the first CAPTCHA. Right? And it was kind of clever. And now Google does it, and it's so unclever. I'm so tired of telling Google that's a bike; that's not a bike.

Steve: I know.

Leo: Clearly we're training their Waymo Division on how to drive.

Steve: And I'm often sure that it's correct, and Google says no.

Leo: No. It's terrible.

Steve: Here are some boats or whatever.

Leo: Horrible.

Steve: Yeah. So...

Leo: And as you've said before, by the way, ineffective.

Steve: Yes.

Leo: If you're really a bad guy, you know how to defeat these CAPTCHAs easily.

Steve: Well, also, I'm sitting here as a normal user doing web-based research with an IP that hasn't changed in several years.

Leo: Yeah, they know who you are.

Steve: And Google is supposed to know, yes, exactly. And so the idea is that it's supposed to not bother you unless it's not sure about whether or not. And it's like, what are you doing? Am I your free image analyzer?

Leo: Yes. That's exactly. It's so obvious.

Steve: Yeah.

Leo: Sorry.

Steve: So thank you. So we've followed the evolution and use of CAPTCHAs. So in keeping with our CAPTCHA covering history, here's what Cloudflare has done. And it looks like to be 100% good news. They said: "Today we're announcing the open beta of Turnstile, an invisible alternative to CAPTCHA. Anyone anywhere on the Internet who wants to replace CAPTCHA on their site will be able to call a simple API, without having to be a Cloudflare customer or sending traffic through the Cloudflare global network. Sign

up here for free." And the URL, it's easy, www.cloudflare.com/lp/turnstile. That's it. You fill out a form in order to create an identity with them, and you make three modifications, which I'll explain in a second.

They said: "There is no point in rehashing the fact that CAPTCHA provides a terrible user experience."

Leo: Yes.

Steve: "It's been discussed in detail before on this blog, and countless times elsewhere. The creator of the CAPTCHA has even publicly lamented that he 'unwittingly created a system that was frittering away, in 10-second increments, millions of hours of a most precious resource: human brain cycles.'"

Leo: Yes. Yes.

Steve: He said: "We hate it, you hate it, everyone hates it. Today we're giving everyone a better option." And of course this comes from our friends at Cloudflare.

Leo: Yeah.

Steve: They said: "Turnstile is our smart CAPTCHA alternative. It automatically chooses from a rotating suite of non-intrusive browser challenges based on telemetry and client behavior exhibited during a session." They said: "We talked in an earlier post about how we've used our Managed Challenge system to reduce our use of CAPTCHA by 91%. Now anyone can take advantage of this same technology to stop using CAPTCHA on their own site."

They then go on to explain that it's not only CAPTCHA's miserable user experience that is the problem. They said: "While having to solve a CAPTCHA is a frustrating user experience, there is also a potential hidden tradeoff a website must make when using CAPTCHA. If you are a small site using CAPTCHA today, you essentially have one option, an 800-pound gorilla with 98% of the CAPTCHA market share. This tool is free to use, but in fact it has a privacy cost. You have to give your data to an ad sales company.

"According to security researchers, one of the signals that Google uses to decide if you are malicious is whether you have a Google cookie in your browser. If you have this cookie, Google will give you a higher score. Google says they don't use this information for ad targeting; but at the end of the day, Google is an ad sales company. Meanwhile, at Cloudflare, we make money when customers choose us to protect their websites and make their services run better. It's a simple, direct relationship that perfectly aligns our incentives.

"In June we announced an effort with Apple to use Private Access Tokens. Visitors using operating systems that support these tokens, including the upcoming versions of macOS or iOS, can now prove they're human without completing a CAPTCHA or giving up personal data. By collaborating with third parties like device manufacturers, who already have the data that would help us validate a device, we are able to abstract portions of the validation process, and confirm data without actually collecting, touching, or storing that data ourselves. Rather than interrogating a device directly, we ask the device vendor to do it for us.

"Private Access Tokens are built directly into Turnstile. While Turnstile has to look at some session data like headers, user agent, and browser characteristics to validate users without challenging them, Private Access Tokens allow us to minimize data collection by asking Apple to validate the device for us. In addition, Turnstile never looks for cookies, like a login cookie, or uses cookies to collect or store information of any kind. Cloudflare has a long track record of investing in user privacy, which we will continue with Turnstile."

They then explain a bit more about what's under the hood, saying: "To improve the Internet for everyone, we decided to open up the technology that powers our Managed Challenge to everyone in beta as a standalone product called Turnstile. Rather than try to unilaterally deprecate and replace CAPTCHA with a single alternative, we built a platform to test many alternatives and rotate new challenges in and out as they become more or less effective."

Leo: Oh.

Steve: In other words, what they're really also saying, yes, is that when the bot farms...

Leo: Figure it out.

Steve: ...start working it out, they're just going to change the rules, and no users will have to have any effect.

Leo: What are the odds we're going to end up having to identify bicycles after a while? Oh, lord.

Steve: Let's hope not.

Leo: I hope not.

Steve: At least we'll know that if that has to happen, it had to happen.

Leo: Right.

Steve: It's not something that, you know...

Leo: Convenient for Google, yeah.

Steve: Yes. Again, I have no explanation for the fact that I'm having to click on parking meters in order to tell Google that...

Leo: It used to, every once in a while, would say, oh, yeah, yeah, you just say I'm not a robot, and they say of course you're not, we know that. But that's gone away now; right?

Steve: Apparently. I haven't seen that for a while.

Leo: Unh-unh.

Steve: So they say: "First we run a series of small non-interactive JavaScript challenges gathering more signals about the visitor browser environment. Those challenges include proof-of-work, proof-of-space, probing for web APIs, and various other challenges for detecting browser quirks and human behavior. As a result, we can fine-tune the difficulty of the challenge to the specific request.

"Turnstile also includes machine learning models that detect common features of end visitors who were able to pass a challenge before. The computational hardness of those initial challenges may vary by visitor, but is targeted to run fast. You can take advantage of Turnstile and stop bothering your visitors with a CAPTCHA even without being on the Cloudflare network. While we make it as easy as possible to use our network, we don't want this to be a barrier to improving privacy and user experience.

"To switch from a CAPTCHA service, all you need to do is" - and they have three things. "One, create a Cloudflare account, navigate to the 'Turnstile' tab on the navigation bar, and get a site key and secret key. Two, copy our JavaScript from the dashboard and paste over your old CAPTCHA JavaScript. Three, update the server-side integration by replacing the old site verify URL with ours." They said: "There's more detail on the process below, including options you can configure, but that's really it. We're excited about the simplicity of making a change."

Leo: Yeah, make it easy. But remember the people who are using that crappy Google CAPTCHA are doing it for one reason. They're lazy as hell. And I bet you, you watch.

Steve: No big effect?

Leo: Yeah. Because that's why they're using this crappy one. They don't care.

Steve: Well, those of us who care will...

Leo: Do you have CAPTCHAs on your site?

Steve: No.

Leo: You see? My point exactly. Neither do I.

Steve: No. No. So anyway, oh, I did create a shortcut for this for our listeners. Anybody who wants to go to that signup page, it's the Shortcut of the Week, so it's grc.sc/891. That'll get you directly to sign.

Leo: I also wonder that if adblockers - certainly NoScript would kill it. I wonder if it's seen as intrusive by some adblockers because they are running some stuff in the background to see who you are; right?

Steve: Right. Although I would imagine Google is doing that, too; right?

Leo: Oh, Google's worse. I think Cloudflare even points out that if you're not logged into Google, if you don't have a Google cookie on your page, they assume you're up to no good.

Steve: You can't be human.

Leo: You can't be a nice person, yeah.

Steve: You're not logged into Google.

Leo: Yeah.

Steve: That's right. Okay. So my last piece is really cool. This is what I posted to my Twitter feed that so distracted me that I forgot to post about today's podcast. Google has put together a marvelously engaging series of six 15- to 19-minute videos under the banner "Hacking Google." It's a world-class production such as you might expect from a company with Google's resources, which of course we're all providing to them. And yes, of course, these are ultimately promotional. But that doesn't dissuade me from recommending them without reservation because they are gorgeous. And they will be, I think, of tremendous interest to this podcast's listeners. I've already, since this morning, received a bunch of feedback, and I've got more loves and retweets on that tweet than I normally get on my weekly announcements of the podcast show notes.

So I first - so this is kind of funny. I stumbled onto the second one in the series, not realizing that it was number two, since it was numbered 001. Of course...

Leo: Oh, I love it.

Steve: I know.

Leo: Zero-based numbering. God bless you, Google.

Steve: Yes, they started numbering from zero, as I've always wished we had...

Leo: Yes. We'd have one more episode.

Steve: ...the foresight to do. That does allow us to squeak out one last one when 999 wraps over, my three digits back to zero. In any event, I've only had the time so far before today's podcast to watch that second one all the way through. But based upon the one I watched, I mean, look at those graphics, Leo.

Leo: Yeah, it's nice, yeah.

Steve: It's got stunning graphics.

Leo: It's network news production quality, I mean, just very high quality. Better than network news, really.

Steve: It is really better than anything that I've seen.

Leo: Documentary production quality, yeah.

Steve: Yeah.

Leo: Well, they've got money.

Steve: Yes, exactly, they've got the money. There are six videos: Operation Aurora; Threat Analysis Group, that's the one I saw which talks about the origin and status of Google's TAG team, and we're always talking about them on this podcast because they're doing such great work. The third one is Detection and Response; the fourth one, Red Team; the fifth one, Bug Hunters; and then the last one is Project Zero. Again, I've only had a chance to watch the Tag Team one. And it was really cool. So anyway, I commend our listeners to that. The videos are collected...

Leo: There's a playlist on YouTube; yeah?

Steve: Yes.

Leo: Yeah.

Steve: They are collected into a YouTube playlist for easy access. I have a link to it in today's show notes, which will take you directly there. Also, and it's weird, too, because when I tried to create my own shortcut to it, the redirect wouldn't go to the playlist.

Leo: No, yeah.

Steve: It insisted on starting at the first video. But if you google the phrase "hacking google playlist," then a few links down from the top you'll find the actual playlist page, which I do have a link to in the show notes. Or, if you want to use a GRC shortcut, grc.sc/hackinggoogle. That will bounce you to the first one, and then they all link successively.

Leo: First one, which is Episode 0.

Steve: Zero, yes.

Leo: Actually, Google should do what we do. We start all our shows now with Episode 0, but that's the trailer. So, see, and if Google had done that, they have a trailer. But if they just made their trailer Episode 0, you would not have been fooled.

Steve: Yes, that's a very good point.

Leo: There's a programming language that I really like, but there's one thing I hate about it. It's called Julia, and it uses one indexed arrays. And it's like - but they just say, this is how people count.

Steve: How dumb is that?

Leo: My point exactly.

Steve: I mean, we're having to do this ridiculous math every single time.

Leo: Just to subtract one. Or add one. Yeah. Well, their point is that normal people have to do the ridiculous math to do zero-based arrays.

Steve: Oh, my god. Well, and one of the most common problems are the off-by-one examples.

Leo: Exactly.

Steve: I mean, I spend more time thinking, okay, do I mean greater than, or greater than equals, or equal to?

Leo: Yeah. I also do, you know, I always cover my rear by doing greater than or equal to, just in case.

Steve: Yeah, and it's funny, too, I do the same thing. When I am ending a loop for a counter, I don't say end it when this is equal to something. It's just superstitious when I say when it's equal or greater. Because, you know, why not?

Leo: Even though it's demonstrably the case it will have that number.

Steve: Absolutely. And if it doesn't, then you've got bigger problems.

Leo: But I do exactly the same thing. It's never equal. It's always greater. Equal seems too precise.

Steve: You're trusting the computer. I get it. It's like, well, it's hard to justify, but it just, it seems better.

Leo: They also have infix arithmetic functions. And while I know everybody knows infix, it just, you know, once you start using...

Steve: Who is this Julia thing aimed at?

Leo: Data scientists. It's actually a really nice language. Very, very nice. But I do have to question that choice.

Steve: So they've dumbed it down in order to...

Leo: They dumbed it down. Just like Python is a little dumbed down, too, because they want - but even Python...

Steve: In order to make it more domain-specific, yeah.

Leo: Or just more accessible. But I think just get used to zero-indexed arrays, and you're done with it.

Steve: Wow. Yeah.

Leo: That was like Pascal would start strings with the string length, and then one is the first letter. That's dumb, too.

Steve: Yeah.

Leo: Null-terminated strings, it's the only way to go.

Steve: That is a win. And that's the beauty - all of my code is null-terminated.

Leo: Yeah. Because you're coding also in assembly, and it just - and that's where this all comes from.

Steve: Yes, and in fact that's the problem with a one-based index.

Leo: Right, because it's incorrect.

Steve: I'm also thinking in terms of the offset from the pointer.

Leo: Right, right, because your pointer points to the first thing in the array, and it's zero in.

Steve: Yeah, they're really going against God's will on this.

Leo: I think so.

Steve: I don't - yeah.

Leo: God or John von Neumann, one or the other, is going to be very unhappy. Sometimes they seem the same.

Steve: So we had a listener, James, who said: "Hi, Steve. As a 2019 Honda Accord owner, I've followed your detailed explanation of the key fob hacking saga with a vested interest. I don't leave anything of value in the car specifically because you made it clear anyone can get in, and even if they can't steal the car without the fob, they can take anything at any time. Unfortunately, last week I was their next victim. Fortunately for me, they didn't get what they were after, as I don't keep my lug nut lock keys in the car anymore since the rims are quite worthy of theft. They did rifle through EVERYTHING," he has in all caps, "and got a bag from the trunk that I liked. But I didn't even notice the $20 in the glove box under the empty lug nut lock key bag.

"Although the car and fob were confused," he says, "my fob would unlock but not lock the doors, I was able to get the car to the dealer." So he's saying the attack confused the fob/car relationship, as we would expect it to. He says, "I was able to get the car to the dealer. And by the time I got there, the fob had somehow synced so it was functioning normally. I explained the experience and my full knowledge that this is a Honda-specific technical flaw that I know the dealer cannot fix, but could they at least wipe and reset my system so that the currently stolen code wouldn't work anymore for this set of thieves? I was told by the service department manager, Bill, that the reset costs $220, and I would have to pay for it. I declined." So I thought that was an interesting bit of feedback from the field regarding an attack that we covered in some length, enough for James to know exactly what was going on.

John said: "Hi, Steve. I've noticed that there's been a lot of discussion around phishing protection on some of the most recent episodes of Security Now! and how SQRL, FIDO, et cetera will address or resolve it. One thing you haven't mentioned recently, though, is that just using a password manager such as LastPass or Bitwarden, et cetera, will also provide a degree of protection as the autofill will fail as a fake URL will not match the one

linked to the stored credentials. Long time Security Now! listener, and thanks from Brisbane. John."

And I put that in here just because I wanted to amplify it. He's absolutely right. I had forgotten to mention in our recent discussion of this that it is one of the benefits of a password manager, which takes what's in the URL absolutely literally. And so if you're expecting an auto logon, and you don't get it, it's protected you from a phishing scheme that was using a lookalike URL or something similarly confusing.

Eric Seidel said: "Hi, Steve. I've been listening to Episode 889 and wanted to mention I've also been using Security Now! since I got my CISSP for CPEs to maintain the CERT." There's a bunch of acronyms for you. He says: "I've never had any issue either when submitting them with ISC. Thanks again for a great resource for that." And I just wanted to say that as a consequence of my talking about that in 889, many of our listeners said, yeah, that's what I do, too. So it's clear that that works for people.

Manuel said, or asked: "I have a question that I'm hoping you might answer. I recently installed backup software (EaseUS Todo Backup) and later realized that it's Chinese software. Now I'm wondering if I just compromised my computer and my whole home network?"

Leo: Oh, lord.

Steve: "Do you have any advice on this" - I know, Leo. Hold on. "Do you have any advice on this way of thinking? Should I reinstall my Windows?"

Leo: Oh, my god.

Steve: "Look for a way to reinstall the BIOS/UEFI, buy a new phone, a new TV?"

Leo: Yeah, throw it all out.

Steve: "A new router, a new everything? Where should I draw the line?"

Leo: Oh, he's joking. Thank god.

Steve: He said he just started listening to the podcast. So, but this is, I mean, it's worth discussing because, you know, Kaspersky is now being looked at very unfavorably because they of course are a Russian cybersecurity firm. We know Kaspersky well. There's, from everything I've seen, zero evidence to suspect them of any bad behavior. But it's just creepy that they're Russian. And we know that companies in Russia don't have necessarily full autonomy over their own actions.

Leo: Right. Yeah. And there's, you know, that's an easy thing to do because there are lots of other choices for antiviruses that aren't from Russia.

Steve: Right.

Leo: I guess you could say the same for Todo Backup. I recommend EaseUS all the time on the radio show because they offer good quality products, in many cases for free. Todo has a free version that does a good job. Maybe that's why it's free.

Steve: And it happens that my favorite remote access software, I've talked about it often, Remote Utilities, it's also from a software publisher in Russia.

Leo: Yeah.

Steve: And it's like, well, I guess being forewarned is useful. But, I mean, and it is tough where we've got like this saber rattling going on increasingly between the U.S. and China, with tensions escalating. At some point, you know, and apparently, like, teams on both sides poking around aggressively inside of each other's networks.

Leo: I got an email from somebody today saying I will never buy an iPhone because they're made in China. I'm only buying Samsung because it's made in South Korea. And I guess, you know, I can't disagree with that. There is zero evidence that just because something's made in China that it is somehow dangerous. I guess maybe software it would be a little bit easier. But if they were exfiltrating stuff from your backup, I think even if you didn't notice, somebody would notice that.

Steve: Well, and we've got motherboards all being made now in China.

Leo: Right. Everything.

Steve: Components. And yes.

Leo: And there you don't really have that much of a choice. I mean, good luck finding one anything, a TV, a router, a phone that's not made in China.

Steve: And we've also talked about the difficulty of even like visually inspecting a motherboard.

Leo: Right.

Steve: Look at all those little bitty chips.

Leo: Right. What do they do?

Steve: No one knows what - yes, exactly. No, it's, I mean, it's a concern.

Leo: I mean, you can be too paranoid. I also don't want people to be xenophobic. You can't - don't conflate the Chinese Community Party, which is, yeah, absolutely awful, with the people of China.

Steve: And xenophobia was exactly the word that I've had on the tip of my tongue through this because it's just, you know, it's not the right way to be.

Cristian Sanchez said: "Hi, Steve. Wondering if you have any thoughts on this Washington Post article's assertion that public WiFi is safe." And he linked to it. He said: "I admit I clicked on it expecting to laugh at thoughtless misinformation, but the discussion in the comments turned me around. Is the near ubiquity of HTTPS enough to declare public WiFi safe? What about man-in-the-middle hijacking DNS? Long-time listener, big fan of the show. Keep up the good work. Sincerely, Cristian."

And I read the article, and it was written by somebody who really knows what they're talking about. And they had the position that I do, which is, yes. There has been such a transformation since we were first talking about Firesheep back then, where open access WiFi was a catastrophe, because the Firesheep, remember, was a Firefox plugin where you were able to - oh, in fact I'm talking on my own lines. I've got some piece of the Washington Post here that I wanted to share.

So the Washington Post wrote, and I've edited a bit to bring it up to the level of our listeners, they said: "You probably don't need to worry about public WiFi anymore. Here's what a creep in a coffee shop could actually learn about you." That was their headline. "From uncovered webcams to reused passwords, it's tough to keep track of how much risk our everyday digital activities actually pose. For example, take WiFi networks in airports and coffee shops. They're part of life for anyone who travels or works remotely. They also have a reputation as cybersecurity risks. Do they still deserve it?

"To see what potential hackers could see on a shared network, we invited professionals from cybersecurity company Avast to 'compromise' my home network, all with my consent. We logged onto the same network at the same time, just like we would at a coffee shop, to see how much data a bad actor with a few free tools could learn about an unassuming WiFi user. What we found, or didn't find, might be a relief for the coffee shop crowd. After a few minutes clicking around my finance, work, streaming, and social media accounts, Avast's team could see the sites I'd visited, though not what I'd done there; the time of day; and the specific device I used, in this case a MacBook Pro. It's not nothing, but it wouldn't do hackers much good if they were looking to rip me off."

He says: "Chester Wisniewski, a principal research scientist at security company Sophos, said that it's also relatively reckless for hackers to sit around messing with public networks. Quoting him: 'That type of data isn't only low yield, it's high risk. If I can phish your password from my chair in Moldova and have zero risk of going to jail, why would I get on an airplane to go to your local Starbucks?'"

Leo: Yeah, that's a good point, yeah.

Steve: So our author said: "In the Internet's earlier days, the vast majority of web traffic was unencrypted, meaning anyone savvy enough to eavesdrop on a network could see everything you type at a website." And that's where I interjected "Our longtime listeners will all remember Firesheep."

He said: "By 2017, the balance had shifted with more than half of all web traffic using the encrypted HTTPS protocol, according to data pulled from Mozilla. Today, few legitimate

sites remain unencrypted, with more than 90% of websites loaded in the United States obscured from prying eyes, according to Mozilla's data. This means even if someone used a public network to spy on you, what they'd discover probably wouldn't be very valuable."

Anyway, the article finishes: "Focus your energies on cybersecurity chores within your control, such as setting strong passwords, saying yes to software updates, and learning the signs of a scam. And don't sweat the public WiFi too hard. If a site, link, or app seems sketchy, steer clear." Which of course is always great advice.

So anyway, great question from a listener, and thank you for the pointer to the Washington Post article. I agree. You know, somebody really concerned could use DNS over TLS, right, so that even the places they're going, even their DNS queries would not be visible in the clear because DNS by default is still not yet encrypted. I don't see that changing. That would be difficult to change. But we could certainly, as individuals, cause our DNS to go through a secure TLS tunnel, and then nothing about what we're doing would be accessible.

Leo: Somebody I'd love your opinion on a thing called the WiFi Pineapple. Are you aware of this?

Steve: Yeah.

Leo: Yeah, they call it a pen testing device, although you could easily use it in a coffee shop to attack WiFi users.

Steve: Yeah, I think the power there would be if somebody were using a laptop where the laptop had inbound security problems, like Windows file and printer sharing was set up to be online.

Leo: Sure. Then you'd...

Steve: So there...

Leo: But one of the things you can do with this is mimic a preferred network and then have that device log into you.

Steve: That would be bad because most Windows users have their Windows firewall set up to be transparent on their local network. So if someone knew who you were and could masquerade as your local network, then your machine would log into it automatically, and it would have access inbound through your firewall in a way that the...

Leo: Now there's somebody inside your computer; right.

Steve: Right. Now there's somebody on your network.

Leo: On your network.

Steve: Yeah.

Leo: One of the ways that you might use this, in this area anyway, probably yours as well, Comcast is the dominant cable provider. They put in these Xfinity routers, as you know, that open access points, public access points. And I would bet a majority of people who are on the road in areas like ours make sure that their phone and their laptop, if it sees an Xfinity network, will join it. So the first thing I would do if I had a WiFi Pineapple is spoof an Xfinity.com network. Just to see.

Steve: Yup, and you just immediately get connection.

Leo: Yeah. And you could, if you wanted to, pass the Internet through it, but you'd still be on their network. So I feel like if you were really a determined attacker, there are still things you can do. Yes?

Steve: Yeah. Yeah. And so the only solution to that would be to be choosing the proper VPN. Remember that you need a VPN that VPNs everything your computer does, not just, for example, your web traffic. You want it to completely encapsulate your network. That would protect you from everything we've just been talking about because it would be linking through all of your local infrastructure connection back to a VPN server somewhere, either back to your home or to a good reputable commercial provider.

Leo: You used the phrase, and I think it's very apt for this, is what's your threat model? So if you're working for the NSA, your threat model is very different than Leo going down to the Starbucks. And so you have to know what your threat model is and act appropriately. And so most people, you know, this Post article's probably accurate. Don't have to do anything.

Steve: I think that's true. Sean Nelson said: "Hey, Steve. I've been looking for a product like this for years, and I finally found it. I think it might interest you, too. It's an SD card with encryption built in."

Leo: Ah.

Steve: "This allows you to take pictures or videos without risking that the contents will be viewed or confiscated by anyone else. From what I can tell, it works by storing a master key set by the user." And I'm going to explain in detail how it works in a second. But he said: "Then, each time the card powers on, it creates and stores a new symmetric session key, which gets processed through the master key for safekeeping. Any new files get encrypted using that new symmetric master key. When the device is powered off, the key is removed from volatile memory, and the only persistent copy of the key is encrypted with the master key. If you take a picture and power off the camera, next time it powers on, no pictures are visible."

He says: "You can take new pictures; but they, too, are unreadable after you turn off the camera. However, when you attach the SD card to a computer and supply it with the

master key, it is able to download the pictures, along with each associated session key, and unlock everything on the camera. I've been looking for this for my dash cam for a long time. Given the state of the world, many others might be looking for something similar." So I think this is very cool and clever. I've got the link to it at the bottom of this article. It's another example of the principle I often observe, which is that the generic and now well-proven tools we already have can be combined in an endless number of ways.

From what Sean described, here's how I would design this device, and it's likely what its maker, SwissBit, has done. During setup on a PC, a public key pair would be created, and the SD card would be provided with the public key, and that's all. Then, whenever the SD card is powered up, an internal high-entropy generator would synthesize a transient 256-bit symmetric key in RAM. That key would be immediately encrypted under the card's configured public key, and only the encrypted key would be written to non-volatile store and retained. Then, during the card's use, all data flowing in and out of the SD card being read and written would pass through that AES 256-bit cipher, which would be transparent to the device it's plugged into. It would look just like a regular SD card, which was initially blank.

When the camera and/or SD card is powered down, the RAM-resident 256-bit symmetric key is forgotten and lost. Since it was encrypted under the user's public key, the only way for it to ever be decrypted would be with the use of the matching private key, which is deliberately unknown to the camera and the SD card. So it's a very slick solution for using an SD card to continuously capture photos and videos while never allowing the contents of the card to be exposed. Anyway, very cool, Sean, thank you for sharing it with our listeners. I wouldn't be surprised if we have some people interested. He did mention that it's pricey. I think it was $159 for 32GB.

Leo: Oh, yeah, wow.

Steve: If memory serves.

Leo: That's like 10 times more than the normal price.

Steve: Yeah. So...

Leo: Plus don't ever have a problem with that card because there's nothing on it but gobbledy-gook.

Steve: That's true.

Leo: You can't recover that data.

Steve: And don't lose your matching private key, either.

Leo: Yeah, right, right.

Steve: Yeah. Jeff said: "Hey, Steve. I noticed on Threema's blog that they just joined Proton, Brave, the Tor Project, and a couple of other Internet services to launch the Privacy Pledge initiative." And so that's at privacy-pledge.com. I won't go into any detail. I just kind of wanted to give them a heads-up. They've got a bunch of, I guess, okay, five privacy-forward principles, this group, which is Brave; David Carroll; Mailfence; Mojeek; Neeva; Open-Xchange; OpenMedia; Proton, you know, the encrypted mail people; Threema; the Tor Project; Tutanota; and You.com. And they're good ideas. The Internet above all should be built to serve people. This means it honors fundamental human rights, is accessible to everyone, and enables the free flow of information. Businesses should operate in such a way that the needs of users are always the priority. That's the first principle.

Second one, organizations should only collect the data necessary for them to sustain their service and prevent abuse. They should receive people's consent to collect such data. People should likewise be able to easily find a clear explanation of what data will be collected, what will be done with it, where it will be stored, how long it will be stored for, and what they can do to have it deleted.

Third principle, people's data should be securely encrypted in transit and at rest wherever possible to prevent mass surveillance and reduce the damage of hacks and data leaks.

Fourth, online organizations should be transparent about their identity and software. They should clearly state who makes up their leadership team, where they are headquartered, and what legal jurisdiction they fall under. Their software should be open source wherever practical and open to audits by the security community.

And finally, web services should be interoperable insofar as interoperability does not require unnecessary data collection or undermine secure encryption. This prevents the creation of walled gardens and creates an open, competitive space that fosters innovation.

So they say: "This is the Internet that we deserve. This is the Internet we are fighting for. It is within our reach. We simply need to be bold enough to seize it." And maybe smoke something. I mean, we all wish that was the Internet we have.

Leo: It's not, I mean, honestly, anybody should be able to sign this because there's plenty of ways out of this, like you should only click the data necessary for you to sustain your service. Okay?

Steve: Yeah.

Leo: It should be encrypted at rest whenever possible. Okay.

Steve: Yeah, I know. And so it's sort of a happy...

Leo: It's pretty namby-pamby, frankly.

Steve: Yeah, reminds me of the Haight-Ashbury...

Leo: Oh, I get it now, yeah, yeah, yeah, yeah, yeah. I mean, I'm glad that they said this. But honestly, Google could sign this without any hesitation.

Steve: Yeah. I don't think they would be admitted.

Leo: You're not invited. Sorry, Google.

Steve: You go to the form, and you put your name in, and they say "We'll get back to you."

Leo: They send you this image. Bye.

Steve: Hope that's a parachute, not a backpack.

Leo: Yeah. Uh-huh, yeah. That's an odd image to put next to your request to join the privacy pledge. I'm not sure - jump off the cliff with us. See what's in that backpack.

Steve: Yeah, don't look down. Donn Edwards, my last share here. He said: "Hi, Steve. Surely the EU nonsense would go away if EU sites changed the analytics.google.com URL to analytics.google.eu."

Leo: Oh.

Steve: Yeah, I know, that's what I thought, too.

Leo: That's clever.

Steve: He said: "Then Google could use their EU servers to do the analytics in keeping with EU rules and laws. Keep up the good work. Donn." Now, I didn't know when I read this whether analytics.google.eu worked. So I gave it a try. It returned a DNS lookup failure. So that doesn't appear to be an immediate option for those in the EU. What I assume Donn meant was that, if Google wanted to respond to this concern, this might be a clean means for them to do so. With nation after nation ruling against the use of Google Analytics as it stands today as being unlawful, something likely needs to change. So I thought that was a very cool suggestion: analytics.google.eu. I mean, it's almost, you know, you're, like, supporting the EU when you make that change.

Leo: Yeah. They'd have to actually create a server at that address first.

Steve: That would be - I think they could do that in about, you know...

Leo: Two seconds.

Steve: About two seconds.

Leo: How long does it take to provision that, yeah.

Steve: Somebody presses a button somewhere, and now there's Google Analytics at .eu.

Leo: It's someone telling that that does not exist.

Steve: I know. I was surprised.

Leo: Analytics requests are redirected to the U.S. probably in every case, I would bet.

Steve: That's where they're most useful, Leo.

Leo: Yeah, that's where we really want the information.

Steve: That's right.

Leo: Let us take a time out because you are going to talk about this Akamai thing, this poisoning.

Steve: Ooh, and it's neat. So last Thursday we got a glimpse into a world-shaking flaw that very few people knew about until now. A 23-year-old Italian security researching enthusiast by the name of Jacopo Tediosi and his friend Francesco Mariani stumbled upon a flaw in Akamai's CDN that could have ruined many days for half the world's websites that rely upon Akamai.

A content distribution network is, at its heart, a massive global distributed cache. It keeps track of a website's content, pretty much everything, and holds the most recent current version of a website's content in a cache which is local to that remote site's visitor. In that way, since the cache is close, the site's performance remains snappy even though its web server might be on the other side of the planet and on a not particularly fast connection.

So first of all, imagining that this can be done at scale safely is nuts. But that hasn't - it's like, no. Don't try to do that. Bad idea. But that hasn't stopped the world's very successful CDNs from doing it anyway. What's absolutely obvious is that this only works if the massively distributed web asset cache maintains its coherence so that it always accurately reflects the correct contents of a remote website. If it were possible in some way to deliberately alter a CDN's distributed cache to change any of those locally cached remote web assets, like for example a site's JavaScript, nothing less than havoc would reign, since such an attack would be tantamount to having the ability to directly alter a site's served content.

Since this podcast is titled "Poisoning Akamai," you already know that this was somehow done. But the reason this podcast is titled "Poisoning Akamai" is because the story told by 23-year-old Jacopo of their discovery, and the discovery's aftermath, is absolutely worth sharing.

So posting to Medium.com, Jacopo opens his story by explaining: "In March 2022, my friend Francesco Mariani and I were teaming up on a private bug bounty program organized by WhiteJar to search for bugs on a website that was using Akamai CDN. The Akamai WAF" - that's the Web Application Firewall - "rules were bothering us while experimenting with the most common attack types, so we quickly got bored and started trying more esoteric payloads and mixing them. Finally, we ended up finding a vulnerability that really made us exclaim, 'Wow, we broke half the web.'"

But let's start from the beginning. He says: "At one point we were intrigued by an unusual DNS Failure response, received by sending twice an HTTP/1.1 GET request to the host being tested with the Connection: Content-Length header and containing another GET request to www.example.com as its body."

Okay. Now, at this point I'll interject that I remember that we did an extensive podcast on exactly this subject, but I cannot recall what the specific topic was. It involved chaining HTTP requests or problems with HTTP chaining or proxying. What these guys are talking about is having an HTTP-style GET query, not a POST query, where the GET query contains in its body, which is normally empty for GET queries, instead another HTTP GET query following all of its regular query headers. That's not the normal format for GET queries, where the specification of the object being queried for is in the GET URL path. But again, I know we covered this years ago.

In this case, their example shows a GET query containing the query headers "Connection: Content-Length," which is not normal for a GET query, and "Content-Length: 53," also not normal. Those are what you see in POST queries because that specifies the after-the-query POST contents will be there, and their length. So upon seeing this odd DNS Failure response, Jacopo wrote: "Weird behaviors like this can often be overlooked while testing so many things, but luckily this time we decided to dig deeper." He said: "I have to admit it took me a while to figure out what was going on, and I also had to reread Nathan Davison's excellent article on 'hop-by-hop' headers that I had studied in the past." And so it is, for what it's worth, indeed an excellent article which I would recommend to anyone who's interested in digging more deeply into this topic, though it's not necessary for understanding what's going on here. I've got a link to the article in the show notes.

He said: "As explained in RFC" - this is Jacopo. "As explained in RFC 2068 Section 13.5.1, there are some special headers named 'hop-by-hop' which are removed from proxies before forwarding requests to the next proxy or the destination." And of course that's what Akamai is, is it's a network of proxies which are caching proxies. He said: "The addition of the Connection header allows including more hop-by-hop headers in addition to the default ones.

"Specifying the Content-Length header as hop-by-hop, it happened that Akamai's first proxy removed it, turning the request body into a second request. Akamai's second proxy then resolved the two requests separately. Since the first proxy received two responses, but only one was expected, a desynchronization occurred, and the second response was queued and subsequently sent in response to requests from other clients/users, causing an HTTP Smuggling Vulnerability." That's a formally named thing.

He said: "Understanding this in detail requires a certain degree of knowledge about network architecture, web protocols, and other fancy stuff, so I try to explain it more easily with a chart." And I have the chart in the show notes, and Leo, you had it on the

screen there. So that sort of gives you a sense for what it is. What it very clearly shows is what happens. It all boils down to an Akamai caching server query parsing error, which results in a single query, rather than being kept as a chain, being split into two and forwarded to two separate destination client web servers. Then the two separate queries are answered, each by their own server, and returned to the cache for caching, which is what Akamai does. But the cache was only waiting for the reply to a single original query that would have normally been chained. Instead, it was split. So it places the unexpected reply into a queue, and that's the behavior that we talked about a few years ago where it then is available for handling serialized HTTP queries. And that reply will be returned to someone else.

So Jacopo says: "However," he wrote, "I could not immediately understand why the DNS error was showing up and why www.example.com was not being resolved. The answer was actually quite simple, but my co-worker's intuition was crucial: Akamai's proxy that routes requests appeared to resolve DNS only internally within Akamai's network." Okay. So that's sort of a subtle point. He was making the point that it was a fluke that the DNS failure occurred. It was the DNS failure that caused them to look more deeply into why the DNS failure, even though what was actually going on, it doesn't really matter that Akamai's only resolving local DNS queries. It's just that it was that fact that led - it was the first little piece of the breadcrumb trail that they then followed that led them to a serious bug in Akamai's caching architecture.

He said: "We were using a VPN to verify that the desynchronization was an 'open' one, meaning that it affected the responses given to IP addresses other than the ones we were attacking from." So in other words, they used a VPN to shift their IP and then generate a query from a different IP to see if it was open, meaning that multiple users at different IPs would also be affected. And they confirmed it was.

He said: "Also believing it possible that the bug concerned all Akamai customers around the world, we changed our target from [the original server they were testing at, which they redacted from their notes] to more popular sites. To our amazement, we noticed that it worked on them all, and that sometimes 'smuggled' responses were being server-side cached from Akamai Edge Nodes for the entire geographic area close to the IP sending the malicious request. This allowed us to semi-permanently, depending on cache times, create new arbitrary contents within almost any domain served by Akamai, resulting," they wrote, "in a HUGE [all caps] impact.

"As a proof of concept we created for the whole Italian area the newly cached page demo.paypal.com/jacopotediosi_hackerone.js, containing the content of www.sky.com/robots.txt, another Akamai customer, because we didn't own a host on the Akamai network to use for publishing our arbitrary contents."

Okay. So just to make this clear so far, they arranged to place a brand new page into Akamai's web cache named "/jacopotediosi_hackerone.js" associated with the root of demo.paypal.com. Thus anyone else in Italy who was also being served by Akamai's cache for that locale which had been poisoned would be able to receive that cached page by asking for it from demo.paypal.com's web server. So upon receiving that query, the local Akamai cache would see that it had a copy of that page in its local cache and would return it quite quickly to whomever asked.

He said: "Once we understood the seriousness of the situation, we decided to report it ethically and responsibly, first of all to Akamai. Unfortunately, we quickly realized that Akamai doesn't have a bug bounty program, hall of fame, swag giveaways, or anything." And they clipped out a piece of the email that they received on 3/25/2022 at 20:29 from Akamai written to Jacopo Tediosi. And the email from Akamai says: "Hi, Jacopo. It's quite clear what's happening here (thanks again for the very detailed report). We had no issues confirming your report. We're working on mitigations and making changes to our

HTTP parsing and processing logic. Unfortunately, Akamai does not have a bug bounty program at this time, and we are not able to provide a direct monetary award. But regarding some other means of recognition of your contributions, I'll get back to you soon. Thanks."

Leo: Send him some stickers. Stickers are always good.

Steve: Un-effing-believable.

Leo: Thanks for letting us know. Okay.

Steve: So Jacopo says: "We are white hats, but we're still not willing to work for free because this vulnerability was very critical. And our skills are rare, complex, and sought after, and we think they deserve to be valued. So while Akamai was patching following our report, we chose to race against the time by asking for bounties from single Akamai customers." Which is so clever.

He says: "While this may sound strange, from our point of view on technologies, those who use a framework, plugin, CDN, whatever, assume both their benefits and their risks. Thanks to our work, Akamai and all their customers have been made aware of a security issue" - boy, and how - "and have been able to fix it. So it's just fair that they pay for our service because without us the vulnerability would still be there.

"We used bbscope to extract links for all the public programs on the most popular bug bounty platforms. Next, we wrote a short bash script to filter from the list only the domains whose DNS pointed to Akamai." Okay. So in other words, they very cleverly reported this bug as it related to specific major Akamai customers whose websites were, indeed, still in serious danger until Akamai pushed out a fix. And what were their results?

Jacopo wrote: "First, WhiteJar immediately gave us 5,000 euros for their private program." They said: "On Bugcrowd, they were not competent enough to understand the vulnerability and closed both our reports for Tesla.com, which was vulnerable, as 'duplicated,'" they said, "of a ticket clearly not related to ours, and for LastPass.com as 'not applicable' because they were unable to reproduce. Intigriti, regarding the Brussels Airlines program which we showed was vulnerable, told us that 'Brussels Airlines is already aware of any request smuggle vulnerabilities in their web assets.'" And Jacopo said, "Yeah, any, LOL." He says: "And closed our ticket as 'duplicated.'

"On HackerOne, some programs refused our tickets and closed as 'not applicable.'" He said: "Starbucks replied the vulnerability, in their opinion, wasn't a major security issue. PlayStation staff failed to reproduce, even after we created a new page for them under the www.playstation.com domain. Marriott informed us that cache issues were temporarily out-of-scope for awards." He said: "However, many other programs paid us. We received $25,200 from PayPal, $14,875 from Airbnb, $4,000 from Hyatt Hotels, $750 from Valve (Steam), $450 from Zomato, and $100 from Goldman Sachs."

Leo: A hundred dollars?

Steve: I know.

Leo: Just a hundred dollars?

Steve: Just a little stingy there, Goldman. "In particular," they said, "Airbnb handled the situation outstandingly, applying custom rules on Akamai's Web Application Firewall in less than 24 hours to block requests containing 'Connection: Content-Length' even before Akamai's official fix. PayPal was also a curious case because they confirmed our report and issued a bug bounty long after Akamai's fix. So we don't know if they ever saw the vulnerability working, or if they just trusted our proof-of-concept video. Akamai fixed it by applying some rules that prevent specifying the 'Content-Length' keyword with the 'Connection' header value, but we're not sure that there are no other bypasses or other unexpected similar ways to split the requests. Unfortunately, Microsoft and Apple acknowledged our reports after Akamai had already deployed a fix, but they thanked us anyway via private emails."

So as we already know, I think from all of this I'm most disappointed in Akamai. Their business is doing something that is so inherently dangerous, where a mistake as we've just seen could have truly horrific consequences, and they have no formal or informal bug bounty established. The idea that a company of this size could just say "Gee, thanks, guys" and not write them a sizeable check on the spot is unconscionable. The way these guys arranged to monetize their discovery was quite ingenious.

Leo: Clever, isn't it? That's so clever.

Steve: Yeah. But that also means that they knew, because they're clever, they knew full well that for an exploit with this much power and virtually universal application, I mean, Microsoft and Apple sites could be compromised. They could have asked the likes of Zerodium for a million dollars, and they would have received it.

Leo: Sold it to the bad guys, in other words.

Steve: Yes. They would have received it. Anyway, I thought it was a very interesting story about something that happened earlier this year. And because the guys were white hat hackers, few people ever learned of it.

Leo: Yeah. Akamai fixed it. Still hasn't given them anything. But they made more than $50,000 by going to the victims, basically.

Steve: The victims, yes, showing the victims that they had problems.

Leo: This works because there was a mitigation the victims could employ.

Steve: Right. The individual customers of Akamai have the so-called Web Application Firewall.

Leo: Right.

Steve: And so on a user-by-user basis, they were giving them a patch for their firewall.

Leo: Had that not existed, then it'd be really a problem. They wouldn't have been able to get any money out of anybody.

Steve: None. None.

Leo: Except just put out a Medium post as they did and say these guys are cheap.

Steve: Boy.

Leo: Thank you, Steve Gibson. Shining light into the dark corners of the Internet. It's a pretty powerful beacon, too. And thank you for doing it.

Steve: You never know what's going to crawl out.

Leo: Yeah. Every Tuesday, right here. We do it about 1:30 or 2:00 p.m. Pacific. That's, let's see, 4:30 to 5:00 Eastern. That's about 20:30 UTC. If you want to tune in and watch us live, that's how you get kind of the first pressing, the extra virgin Security Now!. The livestream is at live.twit.tv. You can chat with us at irc.twit.tv.

You can also go to Steve's site and get a copy: GRC.com. He's got two unique versions of the show. He's got of course the 64Kb audio, but he's got 16Kb audio for the bandwidth-impaired and those great transcripts by Elaine Farris, which lets you read along as you listen or search for topics and that kind of thing. All of that at GRC.com. While you're there, pick up a copy of Steve's true bread and butter, SpinRite, the world's best mass storage maintenance and recovery utility, 6.0 the current version, 6.1 coming any day now. And you'll get a free copy if you buy it today, buy 6.0 today. Steve also has lots...

Steve: And you'll get very early access to the beta because it's going to - because of the fact that it's a DOS app, I'll have that available to 6.0 owners a long time before it's all packaged in its window boot generating form.

Leo: Oh, okay, cool.

Steve: So that's the other thing is, I mean, I'll be working as quickly as I can, but there's just going to be some delay.

Leo: Good to know, okay.

Steve: So there'll be a really early access.

Leo: Another good reason to get it right now: GRC.com. Lots of free stuff there, fun stuff. You can leave Steve feedback there: GRC.com/feedback. You can also leave it on his Twitter account. He's @SGgrc. His DMs are open. @SG, Steve Gibson; GRC, Gibson Research Corporation. That's another good place to leave feedback for Mr. Gibson. We have 64Kb audio and high-quality 720p video available at our site, TWiT.tv/sn. You could download it there. You could subscribe in your favorite podcast player. There's even a YouTube channel dedicated to Security Now!. Plenty of ways to get it. The only thing I ask, get it every week. You don't want to miss an episode.