We Told You So!

Description: This week we're chock full of questions! Why is my new ValiDrive freeware not published yet? Why did Apple quietly remove PDF rendering from the Mac after 39 years? Has the NSA been hacking China? What mistake did Microsoft recently make that would require the use of a bigger hard drive? Why did Signal just announce their use of post-quantum crypto? What's the big hurry? Is it possible to create a new web browser from scratch? And if not, why not? Does public key crypto really go both ways? Can pure math generate pure random numbers? One of our listeners believes he has.

High quality (64 kbps) mp3 audio file URL: http://media.GRC.com/sn/SN-941.mp3
Quarter size (16 kbps) mp3 audio file URL: http://media.GRC.com/sn/sn-941-lq.mp3

Could encrypting an entire hard drive, then throwing away the key, be used in place of the random noise wiping I'm a big fan of? Why hasn't the Unix time problem been fixed yet? Or has it? Will all of the stolen LastPass vaults eventually be decrypted? Am I really leaving Twitter? And, finally, why in the world is this episode titled "We Told You So!"? The answers to those questions and more will be revealed by the time we're done here today. Welcome to Episode 941 of TWiT's Security Now! podcast.

SHOW TEASE: Hey, it's time for Security Now!. I'm Ant Pruitt, sitting in for Mr. Leo Laporte while he's out enjoying a Green Bay Packers game. That is still so funny to say. This week I'm sitting with Mr. Steve Gibson as he goes through some interesting news here in the world of cybersecurity. We have the NSA hacked Huawei - well, yeah, several years ago. Is that really news? We also take a look again at what's been going on with LastPass and some of the implications regarding their previous breach. And also that algorithm doesn't quite add up. Y'all stay tuned.

ANT PRUITT: This is Security Now!, Episode 941, recorded Tuesday, September 26th, 2023: We Told You So!

Hey, what's going on, everybody? I am Ant Pruitt, and this is Security Now! here on TWiT.tv with the one and only, the man of the hour, or couple hours, Mr. Steve Gibson. How you doing, sir?

Steve Gibson: Yes, I think we can probably fill our listeners' time with a couple hours of all kinds of neat security information and news. And this week we are, you know, no exception, we are chock full of questions.

ANT: Oh, yeah.

Steve: Why is my new ValiDrive freeware not published yet? Why did Apple quietly remove PDF rendering from the Mac after 39 years? Has the NSA been hacking China? What mistake did Microsoft recently make that would require the use of a bigger hard

drive? Why did Signal just announce their use of post-quantum crypto? What's the big hurry? Is it possible to create a new web browser from scratch? And if not, why not? Does public key crypto really go both ways? Can pure math generate pure random numbers? One of our listeners believes he has.

Could encrypting an entire hard drive, then throwing away the key, be used in place of random noise wiping, which is what I'm a big fan of? Why hasn't the Unix time problem been fixed yet? Or has it? Will all of the stolen LastPass vaults eventually be decrypted? Am I really leaving Twitter? And, finally, why in the world is this episode titled "We Told You So!"? The answers to those questions and more will be revealed by the time we're done here today. Welcome to Episode 941 of TWiT's Security Now! podcast.

ANT: Oh, man, this is going to be a lot of fun. And I've got to tell you, I'm looking forward to that LastPass discussion because I have some thoughts, and I'm sure you're going to set me straight, as well as set everyone else straight here on the show. And for folks that are tuning in and thinking, wow, Leo's voice is really different, or wow, boy, he's got a super tan going, no.

Mr. Laporte and the family, they are out of town having a good old time up in Lambeau Field, or actually Green Bay, Wisconsin, having a good time birthday celebration with their son as well as some local TWiT fans if they're in town, too, and they're going to hang out together. It's going to be a good time. So if you're in that area, be sure to go by and say hello. I'm going to hang out today and get as much knowledge as I can from Mr. Steve on all of this infosec stuff because it's always fascinating. So Mr. Steve, what's going on now? What you got for us today to start the show?

Steve: Well, we always have a Picture of the Week. So this week's no different. This one is a six-frame cartoon which shows the evolution of a particular well-known product category. This is a toaster. So we have the original toaster in the first frame which, you know, it's got a little handle on the side and it makes toast.

ANT: Perfect.

Steve: Now, we're going to then - right, exactly. Unfortunately, that's probably the optimal condition for the toaster because then the next frame it's been WiFi enabled. So we've got a screen n the side showing an hourglass. This makes toast after making you wait for a firmware update.

ANT: Oh, boy.

Steve: So not clear that was actually a benefit. Then we move to the data-driven toaster, which makes toast by watching how you like toast. Then we evolve to the Toast-as-a-

Service, which makes toast for $5.99 a month. And there's a screen on the side of this toaster that says "Log In." So that's right. Now you've got to log into your toaster in order to, you know, brown your bread. The fifth frame here is the ad-supported toaster. It's got a larger screen because it wants to make sure you see the ad, and it makes toast and lets you know that Smuckers is on sale today. And finally, in the final frame of this, we've got the AI-enabled toaster. It's got that red glowing eyeball from the Hal 9000 of space, no, not space.

ANT: No, the Hitchhiker's Guide? Is that the one?

Steve: No, no, no, it's Kubrick's...

JOHN SLANINA: "2001: A Space Odyssey."

Steve: Oh, thank you, John, of course.

ANT: Thank you, Mr. JammerB. And thank you, chatroom. They got it.

Steve: Just blanking on it. Anyway, so this AI-enabled toaster is actually responding to us. It says: "Toast? I'm afraid I can't do that, Dave." So, yes. Not exactly an advance as far as toast goes, which sort of tells you sometimes you just need to keep the microprocessors away from your appliances because there's no real value to be added there.

ANT: What's so funny is I'm looking at those images. And as you're going through each cell, my brain is automatically assigning a big tech company to each cell. It fits a lot of them quite perfectly, for example, you know, the ad-supported or as-a-service could clearly be Amazon, easily, you know.

Steve: Right.

ANT: But anyway, what's going on with ValiDrive, sir?

Steve: ValiDrive. So I've received messages from listeners who are anxious to start testing their various USB flash drives with GRC's forthcoming ValiDrive freeware utility. You may not know, Ant, but it turns out that there are a lot of counterfeit or fake flash drives now on the market.

ANT: Oh, yeah, I've heard. They've made them super fast, supposedly, and there's nothing in there but some standard, like a V90 chip in there, which should be something better.

Steve: Right. There is that, and also the big problem is thumb drives that claim to be one or two terabytes, but only have a 32GB chip in there. And what's diabolical is that the memory is at the beginning of the drive, where the file system is. So it looks like it's formatted. You can even be storing data there. And so the file system is there. And of course we actually heard a story where someone purchased a - and you'll appreciate this as a photographer - got a compact flash card that was counterfeit, stuck it in their camera, took wedding photos throughout all of the wedding process, came back, and the photos were not stored because all of the file system, filenames were there, but actually out in the drive's real estate there was no memory. So the camera thought it was writing them, but they were not actually being recorded.

So anyway, so as is usual for me, this little side project is taking longer than I expected. I announced it a couple weeks ago, I guess like three or four weeks ago. And since I paused the completion of SpinRite v6.1, right on the verge of getting that finished, I do feel a great deal of pressure to get ValiDrive finished and back to work on SpinRite. But I don't believe that I'm going to look back and feel this was a mistake. All of my previous low-level drive work has been in DOS, where of course SpinRite still lives. And in DOS it's possible to own the entire machine.

It turned out that Windows's USB chain fought back and interfered with ValiDrive's operation much more than I expected. So getting this done correctly was quite involved. But it has made ValiDrive unique because it now incorporates a bunch of technology that's not available elsewhere. And along the way I've developed a lot of new code that's going to be very useful for future USB work under Windows, like for GRC's secure drive wiping utility, which will happen in the future. And although that doesn't help us today, it

will in the future. And, you know, I always take the long view since I plan to be around and active for quite a while yet.

ANT: Please and thank you.

GRC: So today, ValiDrive finally appears to be working well, and it's being heavily tested and used by GRC's testing group. It has opened my eyes about just how severe this fake USB drive problem has become while none of us were really paying that much attention to it. So I'm not quite ready to turn it loose because once it's finished I don't want the distraction of needing to keep coming back to fix little issues that I ignored out of a rush to publish. You know, I want to get back to SpinRite 6.1 and then immediately on to SpinRite 7. So little things like not saving its reports properly on screens where Windows font sizing is set to other than 100%, or some of its UI text not appearing when a user's screen is set to high-contrast mode, or when a user may be unable to discern closely similar colors clearly, you know, they've got a little bit of a colorblindness problem.

Anyway, those are details that I need to put behind me so that what's published will be finished and probably useful for years to come. Anyway, I also mentioned creating and saving reports. So yes, there's been a little bit of feature creep along the way. It's a little more than I had initially expected to get finished. Anyway, it's become very nice, and it's a useful utility which I'll formally announce and publish as soon as possible. Anyway, so I just wanted to get that out of the way because people have been saying, hey, where is it? So it's all working well, and it's coming soon.

ANT: Outstanding.

Steve: So the Eclectic Light Company posted a blog entry that had an interesting piece yesterday which they titled "Postscript's sudden death in Sonoma." Now, they don't mean Sonoma, California. Well, actually, Apple was referring to Sonoma, California. But in this case "Sonoma" refers to the most recent major macOS release. And their article has some interesting observations about Postscript as a dangerous interpreter, which shouldn't surprise any of our listeners. So I want to share what they wrote. And I'll comment, you know, a little bit inline and on the other side.

They said: "If there's one language that's been at the heart of the Macintosh for the last 39 years, it's PostScript, the page description language developed by the founders of Adobe, the late John Warnock, and Adobe's team of engineers. It brought the Mac's first commercial success in desktop publishing, in PostScript fonts, and early PostScript printers including Apple's game-changing LaserWriter. Although Mac OS X never inherited NeXTStep's Display PostScript, its descendant Quartz and Core Graphics are still based on PostScript's relative," which is the PDF. So just to be clear, it's PostScript which is the issue here being discussed, not PDFs per se. PDFs are still around and obviously in heavy use.

So they said: "Following a short illness that started in macOS Monterey 12.3, PostScript has died suddenly in Sonoma. The first sign passed almost unnoticed in Apple's release notes to macOS 12.3, where it recorded the so-called 'deprecation' of PostScript in WebKit." They said, Apple said: "Support for inline viewing of PostScript files is no longer available." So that was just sort of like in passing in WebKit. But it sort of did indicate where things were headed.

"Then in macOS 13.0, Preview lost the ability to convert PostScript and EPS files." Apple said: "The Preview app included with your Mac supports PostScript (.ps) and Encapsulated PostScript (.eps) files in macOS Monterey or earlier. Starting with macOS Ventura, Preview no longer supports these files." So again, a little more deprecation of this. They said: "Other apps that can view or convert .ps and .eps files are available from the App Store and elsewhere. Finally," wrote the blog, "the complete removal of support

for PostScript and EPS was recorded as another deprecation in the release notes for Sonoma."

Okay. So PostScript, they finished saying, "is an old stack-based interpreted language designed at a time when code security had barely been conceived, and malicious software hardly existed. Among its attractive features is the fact that any PostScript object can be treated as data, or executed as part of a program, and can itself generate new objects that can in turn be executed."

Okay, now I'll just pause here for a minute to observe that Windows Metafiles, which turn Windows drawing primitives into an interpreted format, originally had that same capability of executing code. That's just the way things were done back then. When this was rediscovered many years later, everyone freaked out, thinking it was a horrific bug. And many people thought that I was nuts when I calmly observed that it was clearly, originally, deliberate. You know, it's definitely a bad idea today because so prone to abuse, but it was entirely reasonable at the time. Everyone just forgot it was there. You know, as this article says: "PostScript is an old stack-based interpreted language designed at a time when code security had barely been conceived, and malicious software hardly existed." You know, exactly.

ANT: Right, makes sense, yeah.

Steve: So, and I'll just make one other note. Although stack-based languages can be brittle, back at the time, defining a stack-based language was a terrific choice by Adobe because stack-based representations and interpretations can be incredibly dense and efficient. And that's what you would want in a page description language where there are effectively no processor cycles and no easy memory. So, you know, PostScript's design was brilliant.

So anyway, their article continues: "More recently, security researchers have drawn attention to the fact that Postscript is a gift for anyone wishing to write and distribute malicious code. As it's effectively an image format, embedding malware inside a PostScript file could enable that to be run without user interaction, as is the case with other graphics formats." You know, you just, you know, any kind of preview will cause that file format to be interpreted, and we've seen exactly that happening in the not-too-distant past.

ANT: Right.

Steve: The article says: "There are three major PostScript interpreters today in common use: There's still Adobe's Distiller engine, which is built into its Acrobat products; there's Apple's PSNormalizer engine, which is built into macOS; and Artifex's open source engine, built into Ghostscript. And that's one that's widely used in Linux and other platforms" because it's open source.

They wrote: "Research into those engines has so far been relatively limited, but has revealed some serious vulnerabilities. Most recently, Kai Lu and Brett Stone-Gross of Zscaler's ThreatLabz published an account of three vulnerabilities they found in Distiller, and one in Apple's PSNormalizer. That was in 2022. Those were fixed by Adobe and Apple last year, in the later case in macOS Monterey, which was v12.5 released on the 20th of July of last year, 2022, and its equivalent security updates for Big Sur and Catalina." You know, they had to push those back also. And from the dates, I suspect that the removal of support for inline viewing of PostScript files in WebKit in Monterey 12.3 may also have been part of Apple's mitigations.

Anyway, they wrote: "Apple was most probably prompted into conducting a security audit of their PSNormalizer as a result of the vulnerability which was reported, and would

have been faced with the choice of either re-engineering it or removing it from macOS completely." The problem is it was just too dangerous as it was. And these guys note: "Unlike the PDF engine in Quartz, PSNormalizer is now little used and has no significant role any longer in macOS."

So, I mean, that is a reason to remove something that is inherently dangerous. So "The first step was to make it inaccessible from the GUI by disabling the feature in Ventura's Preview, then following that in Sonoma by removing PSNormalizer altogether, so removing its command tool, which is pstopdf, and the Core Graphics' CGPSConverter.

"This leaves those still wishing to convert PostScript files with a choice between Adobe's Distiller," which you would have to pay for because it's in their Acrobat products, "or Artifex's Ghostscript." You know, and to be fair, that's had its own share of vulnerabilities. Again, this is not, you know, PDF is a fundamentally dangerous format. It is not easy to get it right because it is so old and so complex.

Oh, and the article did mention there's is also a third option, which would be to run a late version of macOS Monterey, where you still have PostScript, in a lightweight Virtual Machine, and that way you could continue to use Apple's PSNormalizer through Preview there. And, you know, for most people that would probably be the cheapest and simplest option.

Anyway, John Warnock, the co-founder of Adobe and the driving force behind PostScript, died on the 19th of August this year. And his page description language had brought success to the Mac for almost 39 years. And even though he hasn't, the PDF format has lived on.

So anyway, to Apple I say "bravo." It is always difficult to kill off features that have any audience; right? Somebody's using it. Somebody's going to be unhappy. And as we've seen through the years, Apple takes some heat whenever they decide to break the status quo in the interest of a better future. Postscript, as I said, is a big, old, and very dangerous interpreter. There is no doubt that some people will complain. That's inevitable. But the fact that something that's inherently dangerous could be removed with relatively little repercussion suggests that Apple has once again made the right call. So, yeah. I just wanted to point out that it's gone from macOS. And, you know, PDF is still there, but not sort of the underlying PostScript interpreter. And that's, you know, that's certainly best.

ANT: You referenced just some of the issues that we've had and known about over the years with PDF format. And yet we still continue to fight to get those squared away. But it's still a problem. Are there any alternatives for PDF? I mean, we can't necessarily say TIF or PNGs or anything like that. Everything is fairly open to being compromised. Do we have any other options out there for something that's going to give us the same performance and reliability of a PDF?

Steve: Well, yeah. The problem is that image formats are not a replacement for a page description language. The reason we have, well, the reason you're looking at a PDF of the show notes is that it is the show notes captured in basically in a printed format. And that's what makes the PDF format unique. It also is not easy to do, which is what makes it complicated. And as we know, complexity is the enemy of security. So if something's going to be complex, there's probably ways to hack it, and it's...

ANT: There's probably a hole somewhere; right?

Steve: Yup, exactly. So has the NSA hacked Huawei? And the answer is, well, uh, yeah, back in 2009. What's weird is that Last Tuesday China's Ministry of State Security published an extremely rare, extremely as in the first time ever, official statement on its

WeChat account. It formally accused the U.S. National Security Agency of hacking and maintaining access to servers at Huawei's headquarters since 2009. Now, okay. What's interesting is not that this is news, it's that it's not news. So I'm mentioning this because this reflects a relatively sudden change in stance for China, and it suggests that this might just be the beginning. The question is, the beginning of what?

Both The New York Times and Der Spiegel originally reported this back in 2014, based upon documents from the Snowden leaks which disclosed a program called SHOTGIANT, which was an NSA operation to compromise Huawei's network for the purpose of finding links which existed between, or they thought might exist, between Huawei and the Chinese PLA, to learn Huawei's internal corporate structure, and also to identify ways to exploit Huawei's equipment, which at the time was being widely adopted in both the U.S. and by our allies and adversaries. So this is the first time the Chinese government has ever publicly confirmed the NSA's Huawei hack. And they posted it on their WeChat channel.

So sort of seems more political than not. The Chinese Ministry of State Security statement didn't go into any technical details about the actual hacking. You know, like they apparently didn't know any more than we've all known since 2014. It just recycled information that was published by The New York Times and Der Spiegel and the Snowden leaks. It does, however, spend a lot of time accusing the U.S. of using, and believe it or not I'm quoting this, "the despicable tactics of the 'Matrix' to maintain a 'cyber hegemony.'" To that end, Chinese officials claim that the U.S. is doing all of the intellectual property stealing and then using its allies and PR machine to hype, exaggerate, and smear China on "Chinese cyber secret stealing issue."

So a cyber threat analyst at the Taiwan-based security firm TeamT5 was quoted saying: "Considering the close relationship between China's cybersecurity firms and the Chinese government, our team surmises that these reports could be part of China's strategic distraction when they're accused of massive surveillance systems and espionage operations."

So, you know, there have been several other recent reports of NSA penetration into China's space, you know, into their networks. And I suppose none of us assumed that the cyber intrusions were all going one way, you know, like that there was no pushback against China's much publicized intrusions into the U.S. You know, we have an NSA, and all of those people dressed in camo must be up to something. So it seems as though the Chinese government may have changed their policy. Rather than pretending to be invulnerable, they've decided that the better strategy is to acknowledge that the U.S. is also intruding into Chinese affairs.

This might also be a reaction to China's very high profile and heavily publicized intrusion into Microsoft recently and their exploitation of their access to enterprise email. But there does appear to be a difference. The evidence we have suggests that when we get into their networks, we just snoop around to gather intelligence. When they get into our networks, proactive damage results. So, I mean, it's not just espionage, it's attacks.

ANT: Depends on who you ask. That's either way is still an attack.

Steve: Yeah.

ANT: Just because I didn't come in there and break something, that still doesn't say that I'm less wrong than you.

Steve: Oh, no. They're certainly not happy to have NSA establishing a persistent presence within their country's internal private networks. As we recently saw, that Chinese attack on Microsoft took a great deal of effort; you know? That demonstrated

that they have some serious cyber skills. But sometimes you just trip over a pot of gold. That was the case when a misconfigured Azure Shared Access Signature, a so-called "SAS token," resulted in 38TB of hypersensitive Microsoft data being exposed, not just for the taking, well, for the taking, but more than that.

Okay. So what happened? A cloud security-focused group known as Wiz Research - just to note that's not "whiz." There's no H. It's just Wiz, as in Wizard. They stumbled over a trove and I mean "trove" as in "we're going to need to get a bigger drive over here," wow trove of Microsoft data. It was all exposed and sitting there out on the Internet. The Wiz wizards explained. They said: As part of the Wiz Research Team's ongoing work on accidental exposure of cloud-hosted data, the team scanned the Internet for misconfigured storage containers. In this process, we found a GitHub repository under the Microsoft organization named robust-models-transfer." And, you know, as an aside, it sounds like maybe the models were robust, but the security was not so much.

They said: "The repository belongs to Microsoft's AI research division, and its purpose is to provide open-source code and AI models for image recognition." They said: "Readers of the repository were instructed to download the models from an Azure Storage URL." Okay, so that sounds kind of deliberate; right? Like that doesn't sound so bad. But, okay, now we're getting to the bad part.

They wrote: "This URL allowed access to more than just open-source models. It was configured to grant permissions on the entire storage account, exposing additional private data by mistake." They said: "Our scan" - and actually they did a little more than scanning we'll get to in a second. "Our scan shows that this account contained 38TB of additional data, including Microsoft employees' personal computer backups."

ANT: Oh, boy.

Steve: "The backups contained sensitive personal data, including passwords to Microsoft services, secret keys, and over 30,000 internal Microsoft Teams messages from 359 Microsoft employees." And that's where you say "Whoopsie!" Now, it also occurs to me from this report that you don't know that there are 30,000 internal Microsoft Teams messages from exactly 359 different Microsoft employees without doing a great deal of data analysis and counting things.

ANT: Right.

Steve: I mean, if you've got 38TB of data, you've got to dig around in there. They also knew that there were employees' personal computer backups, and that those backups contained sensitive personal data, including passwords to Microsoft services and secret keys. So, wow. As I said, they probably did need to get a bigger drive to hold all of that.

ANT: Unbelievable.

Steve: I know. So the report continues to explain. "In addition," they said, "to the overly permissive access scope, the token was also misconfigured to allow 'full control' permissions instead of read-only. Meaning not only could an attacker view all the files in the storage account, they could delete and overwrite existing files, as well." In other words, you know, change anything they wanted. Okay. Now, wait a minute.

ANT: Can you imagine an attacker showing up and getting in, and it was like, yes, just celebrating? It's like, it can't be this easy.

Steve: Like I said, the Chinese cyberattack that got a hold of Microsoft's corporate email, that required some skills. Here Microsoft literally left the door open.

ANT: Door's open.

Steve: And really, you know, these guys said, meaning not only could an attacker view all the files in the storage account, they could delete and overwrite them. But, you know, this really does cause us to question the use of the term "attacker" here. If you pick up a lost USB thumb drive in a parking lot, have you attacked anyone? It seems to me you're just observant. Right? You just saw the drive sitting there. So if Microsoft is waving their arms around and saying: "Hey, come over here and download a bunch of stuff using this URL. Oh, and while you're here, how would you like to read 30,000 pieces of private internal corporate communications from 359 of our employees and poke around in some of their computers' backups?" You know, does that qualify as an attack, if you say "Well, thank you, that does sound interesting."

ANT: Microsoft has a weird way of doing honeypots.

Steve: Wow. So anyway, these Wiz Wizards said: "This is particularly interesting considering the repository's original purpose, providing AI models for use in training code. The repository instructs users to download a model data file from the SAS link and feed it into a script. The file's format is .ckpt, which is a format produced by the TensorFlow library. It's formatted using Python's pickle formatter, which is prone to arbitrary code execution by design. Meaning, an attacker could have injected malicious code into all the AI models in this storage account, and every user who trusts Microsoft's GitHub repository would then have been infected by it." In other words, this could have been, like, really bad.

They said: "However, it's important to note this storage account wasn't directly exposed to the public; in fact, it was a private storage account." Here's what happened. "The Microsoft developers used an Azure mechanism called 'SAS tokens,' which allows the creation of a shareable link granting access to an Azure Storage account's data. This means that, upon inspection, the storage account would still seem to be completely private. But as we now know, it was anything but.

"In Azure, a Shared Access Signature, SAS token, is a signed URL that grants access to Azure Storage data. The access level can be customized by the user who creates the URL with permissions ranging from read-only to full control, while the scope can be either just a single file, just a container, or an entire storage account." And as we know, that's what happened here. "The expiration time is also completely customizable, allowing the user to create never-expiring access tokens. This granularity provides great agility for users, but it also creates the risk of granting too much access. In the most permissive case, as was the case with Microsoft's token above, the token can allow full control permissions on the entire account forever, essentially providing the same level of access as the account key itself."

So, okay. We can be glad, or at least hope, that adversarial cyberattackers did not stumble upon this. We can be glad that the Wiz Research guys did, and that they promptly, after doing apparently a bunch of counting to see just how much of what was there, gave Microsoft a heads-up about this little misconfiguration mistake.

And I noted that Microsoft's own Microsoft Research, you know, MSRC blog posting for this incident was titled, got to love this: "Microsoft mitigated exposure of internal information in a storage account due to overly permissive SAS token." Okay, right.

ANT: Overly permissive.

Steve: Overly permissive. And they mitigated the exposure. Wasn't that nice. I won't spend any more time on this other than to note that in Microsoft's sharing of what they called their "learnings" - yes, they actually did call it that - their learnings from this, they

never got around to mentioning just how much of their highly sensitive data had been flapping in the breeze.

ANT: Well, of course they wouldn't because that's how big tech works. That's how big corporations work.

Steve: That's right.

ANT: Yeah, we had an attack, so let's just keep this story as minimal as possible; you know?

Steve: Yup. Okay.

ANT: Oh, man.

Steve: Okay. One more story, then we'll take a break.

ANT: All right.

Steve: Signal's PQXDH Quantum-Resistant Encryption. Now, the Signal encrypted messaging platform was originally named "Axolotl," believe it or not.

ANT: Wait, what?

Steve: Yes. It was called Axolotl.

ANT: I'm glad they had a branding meeting.

Steve: A-X-O-L-O-T-L, which still seems like it's missing some vowels or something. Axolotl. They called it that, it's named after a newt, N-E-W-T, a newt which has self-healing powers. Because the Signal protocol has some of that. In some ways Signal, which uses a resynchronizing cryptographic ratchet system, is a, as I said, a similarly self-healing protocol. It was originally designed by Moxie Marlinspike of what was then named Whisper Systems, and then wisely renamed from "Axolotl" to "Signal."

ANT: We have a lot of people in our live Discord giving us GIFs of Axolotls, apparently. And I will say I still stand by branding statement.

Steve: Yes. So of course it hit the big-time when this protocol was adopted by Meta to be the secure messaging protocol for their WhatsApp messenger. And if the phrase "resynchronizing cryptographic ratchet system" doesn't ring any bells, if you're a listener who joined us after April 12th of 2016, or you're a longtime listener who would be interested in a refresher, I did one of our famous deep dives into this truly lovely messaging protocol which, thanks to Meta's adoption, is now in use by more than a billion people worldwide. Anyway, look for Security Now! Episode 555 which we titled "WhatsApp." And that's where I do a deep dive into Axolotl, which is now fortunately rebranded as Signal.

Okay. The reason we're talking about Signal today is that Signal, the company, made headlines last week in the tech community with their announcement that their already extremely clever and well-designed, very secure Signal protocol was being upgraded with something they called "PQXDH quantum resistant encryption." And if I didn't already have a topic for today's podcast, this would have been it. But we have room for both.

Okay. Signal's move might at first appear premature, since the threat posed by quantum computers to public key crypto, which we rely upon today, remains purely theoretical and

may well remain so for the foreseeable future. We've had some fun in the past at quantum computing's expense, noting that breaking RSA-style crypto, which would require determining the prime number factors of a massive number represented by more than 4,000 binary bits, appears to be safe for now since it was considered to be a breakthrough and a huge accomplishment when today's most advanced quantum computer successfully factored the number 35. So that's the best we've been able to do so far. We need to factor a 4,000-bit number in order to break RSA-style crypto. So it seems to be, the point is, very safe for the time being.

Now, given that, it might appear that Signal's move to an overtly quantum resistant protocol is premature. But if nothing else, it's brilliant marketing. Okay. But there's more to it than that. The security world has coined another abbreviation which is pronounced "Handle" because the abbreviation is H-N-D-L. HNDL stands for "Harvest Now, Decrypt Later." And we know that our dearly beloved NSA has built a truly massive, somewhere between 1 and 1.5 million square foot data center of some sort out in the boonies of Utah.

So there's more than a passing chance that the "Harvesting Now" first half of "Harvest Now, Decrypt Later" strategy is already well underway. They're sucking in all the messaging that's going on. They can't decrypt it today, but they assume they're going to be able to decrypt it someday. So grab it now, harvest it now, decrypt it later. In other words, you know, what the lesson is then for us now is if you want your secrets today to remain secret past the foreseeable future, it's never too soon to begin encrypting under post-quantum crypto technology. And that's what makes Signal's announcement last week significant. Okay. So what exactly has Signal done?

Signal's current shared secret key agreement protocol is known as X3DH. The "DH" is short for Diffie-Hellman, which is a well-established key agreement system. We've discussed key agreement protocols in general, and Diffie-Hellman in particular, many times in the past on the podcast. And I know it pretty well since it plays a large part in SQRL's more tricky security features, you know, the features that I designed into my own SQRL technology.

Okay. Briefly, a key agreement protocol allows two ends of an insecure and public connection to exchange some information in plain sight while each of them obtain a shared secret which no one else is able to get. I know. You're scratching your head, Ant. I know. It's really very cool technology. It seems counterintuitive that their conversation could be completely known, while they each arrive at the same secret that only they know, but it works.

ANT: Wow.

Steve: The "X3" in the X3DH refers to the "X25519" elliptic curve, which is the flavor that Signal's Diffie-Hellman key agreement, and actually mine, I chose the same one, has traditionally used. Okay. And what's really cool is that they're still going to use this in the future. What they decided to do was to add another key agreement protocol - this one believed to be quantum-safe - to their existing system in such a way that BOTH of the protocols, the old and reliable elliptic curve Diffie-Hellman and the "believed to be safe today and tomorrow" newfangled quantum-safe system, both would need to be simultaneously broken and cracked in order for an attacker to obtain the shared secret that's used to decrypt Signal's communications.

Signal selected one of the NIST contest finalists that's believed to be the best, the one known as CRYSTALS-Kyber, K-Y-B-E-R. And we've talked about it previously. They chose it because it's built upon a solid foundation. But they also wisely decided that, since it's new and unproven - and actually one of the other contest finalists turned out to be broken, breakable by conventional computers, so whoops. It's a good thing they didn't

use that one. You know, again, not yet proven. That's why they're keeping the Diffie-Hellman around, just to have a belt and suspenders approach. Anyway, so because it's new and unproven, they decided not to depend upon it solely. So Signal's original X3DH has been renamed PQXDH for post quantum, you know, the PQ is for post-quantum and the XDH for the existing elliptic curve Diffie-Hellman, which survives in the new combined protocol.

And here's the coolest part. This new PQXDH is already present and supported in the latest versions of Signal's client applications, and it's already in use protecting any conversations initiated after both sides of the chat are using the latest Signal software which supports both the original and the updated PQXDH protocol. Then, over time in the future, after sufficient time has passed for everyone to be using the new Signal and to have updated, they plan to disable all use of the old non-quantum-enhanced X3DH-only protocol in favor of all new conversations, which would then be requiring PQXDH for all new chats.

So, you know, this is great; you know? It's clear that we are moving into a post-quantum world, even though it seems like we're still a long way away from quantum computers being a threat to our existing crypto. You know, this whole concept of Harvest Now, Decrypt Later, that makes a lot of sense. Now that Signal has led the pack of messaging apps by introducing quantum-safe messaging, the rest of the pack, who may have been caught by surprise, you know, caught off guard, they'll have no choice but to figure out how to do the same, you know, because they don't want to get left behind. So it's great. Essentially this means that we will be, you know, moving to the next generation of crypto in advance of its clear need. And that makes sense, too, because at some point we will be surprised when a quantum computer is created which is able to do this. We might as well be ready for it.

ANT: Yeah. Now, did you all speak last week about what was going on with Signal in the EU and the EU's new plan to pass a bill basically giving them a backdoor to everything?

Steve: So we've talked a lot about the privacy issues. What happened in the UK was they...

ANT: Not EU, UK, sorry.

Steve: Right. They amended the legislation to basically give the messaging apps an out. They said you must, must, must scan where technically feasible. And that "where technically feasible" clause says, oh, okay, fine, it's not technically feasible.

ANT: Okay.

Steve: So the politicians get to say we did, you know, we implemented strong new legislation, whereas none of the messaging apps are worried about it anymore.

ANT: Oh, okay.

Steve: So what will be interesting is to see what the EU does. Will they also back down? Because it really looks like any government that says we must have you scanning all content, they're simply going to lose all cryptographic services in their country. And they can't afford to do that.

ANT: Right, yeah. I believe I saw Signal's leadership saying, yeah, we'll just step out of here. See you later.

Steve: Oh, yeah. Yeah, yeah, yeah.

ANT: When it comes down to it.

Steve: Yup.

ANT: I totally get that. Folks, this is Security Now!, Episode 941 with Mr. Steve Gibson, dropping some great knowledge and information on us in the world of cybersecurity and information security. Mr. Steve, it's time for us to close up the loop. What you got?

Steve: Yep, we've got some really interesting listener feedback that drives some, I think, interesting observations. Our listener, Dustin Smith, put me onto a blog posting which was originally written 3.5 years ago on March 18th of 2020; yet if nothing, it's more relevant today than it was then. This blog posting was written by a guy name Drew DeVault. It's an on-point rant about what modern web browsers have become, and I think everyone here will find it interesting.

Here's what he wrote. He said: "Since the first browser war between Netscape and Internet Explorer, web browsers have been using their features as their primary means of competing with each other. This strategy of unlimited scope and perpetual feature creep is reckless and has been allowed to go on for far too long." He says: "I used wget to download all 1,217 of the W3C specifications." That's, you know, the World Wide Web Consortium, the W3C. "So all 1,217 of the World Wide Web Consortium specifications which have been published at the time of writing."

And he said: "Not counting WebGL, which is maintained by Khronos." He said: "Web browsers need to implement a substantial subset of this specification in order to provide a modern web experience." He said: "I ran a word count on all of these specifications. How complex would you guess the web is?"

Okay. He said: "The total word count of the W3C specification catalog is 114 million words at the time of writing."

ANT: Wow.

Steve: If you added the combined word counts of the C11, which is, you know, C language 11, C++17, UEFI, USB 3.2, and POSIX specifications, all 8,754 published RFCs, and the combined word counts of everything on Wikipedia's list of longest novels, you would still be 12 million words short of the W3C specifications.

ANT: Wow.

Steve: Now, I mean, okay. So it is that complex.

ANT: Wow.

Steve: So he said: "I conclude that it is impossible to build a new web browser. The complexity of the web is obscene. The creation of a new web browser would be comparable in effort to the Apollo program or the Manhattan project." He said: "It is impossible to, first, implement the web correctly." Impossible. "Second, implement the web securely. And third, implement the web at all."

He said: "Starting work on a bespoke browser engine with the intention of competing with Google or Mozilla is a fool's errand. The last serious attempt to make a new browser, which was known as Servo, has become one part incubator for Firefox refactoring, one part playground for bored Mozilla engineers to mess with technology no one wants, and zero parts viable modern browser. But," he said, "WebVR is cool; right? Right? And the consequences of this are obvious." He says: "Browsers are the most expensive piece of software a typical consumer computer runs. They're infamous for using all of your RAM,

pinning CPU and I/O, draining your battery, et cetera. Combined, web browsers are responsible for more than 8,000 CVEs" - you know, exploits, flaws, bugs - "alone."

And then Drew switches from establishing some facts to making a very interesting observation. He writes: "Because of the monopoly created by the insurmountable task of building a competitive alternative, browsers have also been free to stop being the 'user agent' and start being the agents of their creators instead. Firefox is filling up with ads, tracking, and mandatory plugins. Chrome is used as a means for Google to efficiently track your eyeballs and muscle their anti-technologies like DRM and AMP into the ecosystem. The browser duopoly is only growing stronger, too, as Microsoft drops Edge, and WebKit falls well behind its competition." Now, remember, this is 3.5 years ago, so of course that part is pretty much history.

He said: "The major projects are open source, and usually when an open-source project misbehaves, we're able to fork it to offer an alternative. But even this is an impossible task where web browsers are concerned. The number of W3C specifications grows at an average rate of 200 new subspecs per year, or about four million new words of specification, or about one POSIX every four to six months. How can a new team of any forked browser project possibly keep up with this on top of implementing and maintaining the outrageous scope web browsers already have now?" He finishes: "The browser wars have been allowed to continue for far too long. They should have long ago focused on competing in terms of performance and stability, not in adding new web 'features,'" he has in air quotes. "This is absolutely ridiculous and has to stop."

ANT: Okay. So I need to interject here because I get where Mr. Drew is coming from with his stop adding features and whatnot. But let's think about this. Let's just think about vehicles; okay? We can have the Honda Civic, you know, simple four-cylinder car. But then we have the Honda Accord with a slightly larger engine. And then we have this, I don't know, like a Ford Mustang with an even bigger engine, so we had these fast cars. And everybody's like, huh, I don't need that fast car. But then you throw in Bluetooth features. Ooh. I want that. And that's what's happening here. People are sold on features. And these large companies, Google and Firefox, they get that. If they want to stay in business, they're going to continue to add features. That's the nature of consumerism; right?

Steve: Okay. Maybe the right analogy, or trying to find a good comparison, not everybody can launch stuff into space, into orbit. You know? It requires so much infrastructure, so much technology, so much other stuff, that there are only a few ways of getting something launched into orbit. So similarly, his point is there's just no way to create a third browser. So there is a duopoly, there is a lock-in at this point. There's no way for there to be competition. And what we see when there's a lack of competition is then the rules change so that the browser creators no longer have to create the best product anymore.

ANT: Okay.

Steve: And so that's the point he's making.

ANT: And what's up with these specifications from the W3C? Do you think it's a bad idea, considering we know about all of the security issues behind the browser? [Crosstalk] come from.

Steve: I don't think it's a good idea. No. In fact, it is the implementation of all these new things which is where new problems get created. No, I'm often, I'm always saying, if we would just leave this stuff alone, we'd be able to fix the problems, and then we'd have some security. But there are incentives for not leaving it alone. You know?

ANT: There it is.

Steve: Everybody wants to force you to upgrade to the newest operation system, the newest browser. Everybody wants the newest features. You know, arguably, it's cool that you're now able to do web conferencing with just your browser. It used to be that you had to have some extra software in order to do it. Now, you know, you don't need that.

So I guess the only thing that I would change about what Drew wrote is his last line. He said: "They should have long ago focused on competing in terms of performance and stability, not in adding new web features. This is absolutely ridiculous, and it has to stop." Okay. I would say, I would change it to: "This is unfortunate in the extreme, but there doesn't appear to be any way for the industry to change that course." And I think, I mean, you know, he says it has to stop. I would argue, how is it ever going to? Why would it?

And I would further add that, to their credit, Microsoft perceived very early on just how important the web browser would be to the future. Remember, they attempted to build it into Windows, and they called it Internet Explorer. And of course they got taken to court and sued over antitrust issues because, you know, they built this browser into their operating system because again, to their credit, they knew this was the future. And as we know, what the web has become even outgrew their, Microsoft's, their ability to browse it. Once upon a time, IE was the most widely used browser. It attaining a peak of 95% market share about 20 years ago in 2003.

ANT: And was still horrible.

Steve: Yes. This podcast began two years later, and we all witnessed IE's decline over this period of time, leading to its inevitable death. After making a massive investment in a brand new post-IE engine, Microsoft, again to their credit, recognized that a modern web browser was too big for even them, and that they'd be better off just hanging bells and whistles off a browser, you know, Chromium, which they got from the Chrome project, that someone else was maintaining and evolving.

So are there any takeaways for us here? No. None that I can think of. Drew's observations, which I think are valuable, pair beautifully with what I've noted to be true of operating systems. Just as no one can create a competitive and useful, truly new operating system from scratch any longer, any operating system that a user is going to sit in front of will need to host a web browser that is all by itself also too massive and complex to create today from scratch.

So I also think all of this was absolutely inevitable. And that further suggests that it wasn't a mistake. It's not a consequence of inertia, which we often observe elsewhere. And we're almost certainly going to continue moving down this path from here on. You know, this is where we're headed.

So another piece of feedback from an anonymous person who actually called himself "Anonymous." He said: "Hello, Mr. Gibson." It sounds like Ant. "Hello, Mr. Gibson. I hope I can ask you a question over the last podcast from Security Now!. Around 30 minutes in, you say that you encrypt with the private key and decrypt with the public key. Is this right? I think it's reverse. What I learned is that you encrypt with the public key and decrypt with the private key. Is this right, or do I have wrong information? Thanks for your response."

Okay. So one of the powerful beauties of public key cipher technology is that the process does work in either direction. It's obviously very useful to encrypt something using someone's public key that you know only they will be able to decrypt with their secret private key. PGP is a perfect example of this.

ANT: Right. Right.

Steve: And digital signatures of all kinds is the best example of this being done in reverse. In fact, PGP does this, too. To provide authentication of the sender of a message, the sender uses their private key to sign the file which the recipient is able to verify using the sender's public key. So when the recipient is verifying the signature, they're decrypting a hash of the document that the sender encrypted with their private key.

So the last time we talked about public key technology I was really pleased with the way that I conceptualized the public and private key distinction used by non-elliptic curve RSA-style crypto. Okay. So you start by obtaining a private key which is just a really large prime number. So you just get a big prime number. And remember that, counterintuitive though it is, prime numbers do not become more scarce as we go farther and farther out, like farther and farther down the numbers. You'd sort of think they would, right, because the definition of prime number is that it's only divisible by one and itself. You know, like no even numbers are prime. But then no every third number is prime because it could be divided by three. And no even number is prime because it could be divided by two.

ANT: Two, yeah.

Steve: And then every fifth number is not prime because it can be divided by five. And so on. So since, you know, because of that, you'd think that would all kind of pile up. And so, like, it'd be hard to find big numbers way out there that aren't divisible by something that's come before. But it turns out that's not the way it works. There's always plenty of them. Okay. So you take a really big prime number, and that's our private key.

Okay. Now, we want to arrange to hide that private key inside the public key. We do this by taking another very large prime number and multiplying that new prime number with the original private key which was also a prime number. Though I've simplified things, it's that multiplied key is the public key. The public key contains the private key. But there's no way to know what it is after they've been multiplied together because to this day, despite all the best brains in math - and, I mean, this has received a lot of attention through the years - no one has ever come up with a practical way to "unmultiply" those two primes through the process of prime factorization.

And, by the way, that is the danger posed by quantum computing. There are reasons to believe that some future quantum computer might be able to finally factor a very large public key which contains those two primes, one of which is the private key. If you could factor that back into its original two prime numbers, then you'd have the private key, which could then be used to decrypt things where you were assuming that was not possible.

So the true magic of public key crypto is that that private key, which has been hidden inside the public key by multiplying it with another prime, that it is still able to perform encryption and decryption operations despite being completely hidden by its multiplication by another large prime number. Anyway, it is just so cool.

ANT: So first off, excuse me while I let all the steam come out of my ears because I'm like, wow, what in the heck? And secondly, with quantum computing, do you feel like this is still something in the near future, to be able to get this?

Steve: No. No.

ANT: I wasn't going to sit here and say I was concerned. But I wanted to ask you first, though.

Steve: Yeah. As I mentioned before, the best we've seen a quantum computer do is factor the number 35. Right?

ANT: Okay.

Steve: Into 5 and 7. So that's like, whoa, it worked. But, I mean, so no. We're not in, I mean, now, okay. There could be a breakthrough; right? There could be some fundamental, you know, crazy ass, like no one expects it, comes out of right field, breakthrough. Could happen. And so that's why we already have, we're already beginning to standardize on next-generation quantum safe crypto so that we will be using that if such a breakthrough happened because if it happened today, you know, I often use the phrase "It's the end of the world as we know it." That probably really would be.

ANT: Yeah.

Steve: It would be, you know, if you were to use really, really bad, and like the amount of badness is how many reallys you put in front, you could be saying "really, really, really, really, really" all day long. It would be really bad. So, yeah. But for now, it looks like we're safe.

ANT: Yeah.

Steve: Okay, now, here's a particularly interesting piece of feedback. A listener whose name I will just say is Jerry, and I'll explain why I anonymized him in a minute, because initially it wasn't. He wrote to me. He said: "Your assertion that 'no deterministic mathematical algorithm can generate a random result' is incorrect." He said: "Non-deterministic mathematical algorithms (processes) can generate true random results, not 'pseudorandom' numbers." And he finished: "I developed such an algorithm."

ANT: Okay.

Steve: Okay. So first of all, I said "no deterministic mathematical algorithm," not "non-deterministic mathematical algorithm." Now, the problem is that the phrase "non-deterministic mathematical algorithm" is, as far as I know, an oxymoron. You know, one plus one equals two. And wait a minute, one plus one still equals two. And, oh, one plus one, yes, it still equals two. All mathematical algorithms are inherently deterministic. Now, if I made a mistake on my math homework in elementary school, explaining to my teacher that this was my implementation of a non-deterministic mathematical algorithm, I'm pretty certain that would not have gone over very well, nor would it have changed my grade.

ANT: I can see little Steve Gibson right now at the front of the class, at the chalkboard.

Steve: This is a non-deterministic mathematical algorithm. That's why I didn't multiply these two six-digit numbers correctly. Okay. So, Jerry, if you indeed have developed a non-deterministic mathematical algorithm which is capable of generating true random results, there's probably a Nobel Prize waiting for you in Stockholm. And more importantly, what you may have actually done is uncovered a provable flaw in the simulation we're all living within. Okay. But seriously, you have my attention. You have access to an audience. Are you just going to leave us all hanging like that? Because that's all he said.

Okay. Now, in a bizarre coincidence, just as I was getting ready to save as PDF, to cast these show notes from Google Docs where I write them every week into their PDF form, I happened to glance down at my phone and saw a reply tweet from this person.

Yesterday, when I encountered his assertion, not knowing how to respond to something which clearly seemed impossible, I simply replied, "I hope you've submitted this for a Nobel Prize." Now, it turns out he didn't take that very well.

ANT: I wonder why.

Steve: Well, you know, certainly not in the somewhat carefree spirit, you know, it was meant. So, okay. The first thing I did was to anonymize his identity in the show notes since I certainly don't want to embarrass him from this dialogue. What he tweeted in reply to my short comment about the Nobel Prize, and actually I'm kind of serious about that because really, I mean, if he really did this, it would be worthy of a Nobel Prize. Anyway, this morning, as I was getting ready to finish the show notes, I saw his tweet.

He replied: "I'm very surprised by your reply. I've been a customer and promoter of your products and services for 30-plus years. I have read numerous articles of yours. I never expected sarcasm from you." He said: "My bad. I developed these and similar algorithms more than 45 years ago. I have always expected others to develop like algorithms, but to date this has not occurred."

Okay, that was his tweet this morning. Now, again, I'm having a little bit of trouble taking this seriously, or at least as seriously as he is. You know, when he says, "I've always expected others to develop like algorithms, but to date this has not occurred," I'll admit my first impulse is to say, "Gee, I wonder why that is?" Okay. Instead, this what I replied to him.

I said: "Hi, Jerry. Okay. So I suppose my reply was somewhat sarcastic. And I certainly would love to be proven wrong in my assertion that, to quote you quoting me, 'no deterministic mathematical algorithm can generate a random result.' The trouble is," I wrote, "I still believe that to be true. And not just a little bit true, but very true. If you're a mathematician, then you understand that the way to proceed from here would be to offer up some proof of your assertion, which appears to defy the laws and principles of mathematics and reality. I'd be more than happy to eat my sarcasm if you can demonstrate that I'm wrong, exclamation point, smiley face."

ANT: So tactful, sir.

Steve: I'm serious, really. I mean, this would be great, you know? I'll buy his plane ticket to Stockholm.

ANT: I've never heard a more polite smart-assery from anyone, sir. This was really well done.

Steve: For what it's worth, if I do hear anything more from Jerry, I will let everyone know because that would be big. Now, it must be that there's a miss - there's a confusion of definition. When I'm saying "no deterministic mathematical algorithm can generate a random result," and he's talking about a non-deterministic mathematical algorithm, which again I think is an oxymoron, okay, we're at cross-purposes here. But anyway, it was an interesting bit of feedback that I had intended to share, and now I'm able to share a bit of dialogue.

Todd said: "Hey, Steve. I listened to Episode 940 where you explain securely wiping drives by writing random bits to all sectors of spinning drives. Lately, I've been using VeraCrypt to encrypt old drives before I get rid of them," and he said, "(using GRC's password page to generate a secure password). Is this as secure as your upcoming software that writes random noise?"

And I think that's very clever. You know? And so if you were to do it twice under different keys, then it would be the equivalent of writing random noise, just as GRC's solution will be. And I would not argue that the second pass is absolutely necessary. You know, I might offer a single-pass option for those in a hurry with my own product. VeraCrypt, like its ancestor TrueCrypt, uses XTS cipher mode encryption, which is the industry standard and NIST-approved way of encrypting block-addressed mass storage using what's known as a tweakable block cipher.

And we also know that any good modern cipher, like AES, generates pseudorandom noise that's indistinguishable from true random noise. It'll be slower than GRC's product, since it's reading and writing in order to encrypt what was already there. And I don't know how large the blocks are that it's using. If they were really small, that would slow it down. Presumably it knows how to do that quickly. It won't be able to access and wipe the latent data that might exist in grown-defect spared out sectors. But it is truly a nifty solution until GRC's product is ready. The VeraCrypt encrypt and then discard the key solution is here today, and it's free. So anyway, Todd, thanks for sharing the idea. I think it's pretty clever, and it's a great application. And I cannot think of any downside to it.

Bill Melvin said: "Hi, Steve. I have a thought regarding the discussion on wiping hard drives on show 940. I assume TrueCrypt and VeraCrypt hidden volumes look like random noise. It seems to me that doing an extra pass of writing zeros to a drive takes you from 'plausible deniability' to absolute impossibility. In an abundance of paranoia, why take the risk of being badgered over something that's not there. Thank you for Security Now!, which I've enjoyed for years. I have my SpinRite license and am looking forward to the new version."

Okay. So when I talked about my idea, my concept for the best secure wipe is to wipe with random noise. The point he's making is that random noise looks like something, even if it's not. So if a drive was filled with random noise, it might induce someone to, in the extreme, put you under some coercion, asking you for what the key was to decrypt the drive, when it was actually just random noise. So he says do another wipe to zeroes so that nobody will think there's anything there. Then they won't come asking for the key. And I think he makes a good point.

Mark Kozel says: "I found your description of SSD wear-leveling fascinating. It is amazing what engineers have come up with to handle issues like this, and it got me wondering. I have a portable SSD with a single multi-gigabyte VeraCrypt blob on it. I open the blob a few times a week and make changes to files inside. After closing the blob, everything appears to be unaltered, as far as Windows knows. So I set VeraCrypt to change the file modification date on the blob when I realized Syncthing needed that to detect a change and sync the blob to my NAS. Does the wear-leveling capability work on the blob? Is it active inside the blob when I have it open in VeraCrypt? Love the show and continuing education credits I get towards my Security+ certification by listening."

ANT: Woohoo!

Steve: So, Mark, the answer is yes. The wear-leveling is occurring at the lowest possible level in the chain of data flow, just before the data is written to the drive. It is not visible to anyone reading or writing to and from the drive, so it functions whether the data being read or written is encrypted or not. And one slick benefit of using VeraCrypt to encrypt the blob on the SSD is that, since VeraCrypt always leaves the data on the SSD encrypted, only decrypting it on the fly as necessary when it's read into the computer, as wear leveling causes blob updates to be written to dispersed regions of the drive, thus leaving obsoleted bits of the blob behind, those obsoleted and isolated blob bits are also encrypted.

So that drive will never need to be wiped, at least not to protect anything that was ever stored in the blob. Once VeraCrypt's volume header has been overwritten multiple times, there's no way to ever recover any of that blob's data, period. Meaning that the wear-leveling which you may not have access to is not a concern as it would be if you stored unencrypted data on the drive, which you then wanted to secure, because all of that wear-leveled data would have been previously encrypted bits in the blob. So a very nice application.

Wayne Patton asked: "How difficult would it be to change the signed integer to an unsigned something else and recompile and fix Unix time?" So he's of course talking about the fact that we were recently talking about how Unix time, the signed integer which is - technically it's 32 bits, but because it's signed we only get 31 bits' worth of range; how because it's signed, Unix time will be - the original 32-bit Unix time will be wrapping around in 2038.

I should have then, but I failed to mention that this Unix time problem was fixed in the Unix Kernel back in June of 2018, after 2.5 years of research into the right way to do it without breaking other things in Linux. In the process, more than 80, eight zero, source files had to be changed. So this Y2038 problem is no longer any concern for Linux kernels ever since. The worry is that before this time, all embedded, for example, Linux kernels of the sort used for industrial automation, satellites, elevators, and who knows what else, will have this trouble. And in those markets there's much more of a "if it's not broken, don't fix it" mentality.

So the concern is not that mainstream Linux or any of the various major Unixes will break. They've all been fixed. It's that some satellite that's been in orbit since before this was fixed may become suddenly confused and do - who knows what it might do? So maybe a long forgotten pump on an oil rig will misbehave. We don't know. Anyway, it's going to be interesting to see whether it's another non-event like Y2K turned out to be, or whether something interesting might occur. And, you know, stay tuned. We'll be here in 2038.

ANT: Oh, Y2K. I've got memories of that, fond memories of that.

Steve: Yeah. So B0wter, he said: "Hello, Steve. Thank you for the many great episodes of Security Now!. I've just listened to Episode 939 and have a thought about what you said regarding the LastPass hack. You mentioned that hackers will go after those accounts where they suspect a big payout. I think you're right about that. But I also think we must not let our guard down. All accounts will eventually be decrypted, and all accounts will be available for purchase on the dark web. This will be a problem for all those, including me, who used LastPass to store information like Social Security numbers and other personal IDs. I think it's important to take precautions while we still can. Kind regards, Johannes."

Okay. And in case you're not aware, Ant, we talked about this last week. There's very good evidence leading us to believe that selective decryption of specific high-value LastPass vaults which were stolen in that LastPass breach are being, that is to say, have been decrypted. And the cryptocurrency wallets of people who stored their master crypto keys in their LastPass vaults have been emptied. And about $35 million or so of cryptocurrency has been transferred.

ANT: Yeah, I saw reports on that. It's pretty scary.

Steve: Yeah. So I wanted to share Johannes's note because I understand his concern, and I imagine it is shared. But I also doubt that the vast majority of the more than 25 million users whose encrypted data was stolen are actually ever in any real danger. Even the advent of quantum computing won't make any difference here since we're talking

about brute-forcing symmetric crypto, which quantum computing does not threaten. The cost of performing 100,100 rounds, which is the level we were at when the breach occurred, where most people probably had their iteration counts set, and we know some were a lot lower than that, and that's a concern. But the cost of performing 100,100 rounds of brute force against a hopefully high-entropy password is truly prohibitive.

These crooks appear to be spending significant money on cracking gear or cloud compute cycles, and they appear to be cracking only a couple of vaults per week. That's the rate at which this cracking is occurring because it is really expensive to do this. That's perhaps a hundred or so per year against a population of more than 25 million vaults.

It's true that someday they might eventually get around to opportunistic attacks against non-high profile targets whose iteration counts are low. But even then, if the passphrase is good, there's no clear reason to believe that any actual cash lies on the other end of the crack, that is, they wouldn't know until they performed the crack. And it still takes a significant investment of time and money to turn someone's vault back into plaintext. It really is not ever going to be easy.

So, you know, the original design of LastPass, which is what caused me to endorse it, was strong. And as long as the iteration count is high, it's always going to be expensive to crack the vault. So, you know, I cannot tell anyone that there's absolutely nothing to worry about. So my intention is only to present some I think probably accurate perspective. Cracking these vaults will never be free, and these people are probably not ones to throw away good money. It will cost money. And I think that the attacks will continue to be targeted, and that they will probably peter out over time, and that it's not the case that they will all be cracked. There's just no reason to. It's just - it is too expensive.

Kevin van Haaren, he said: "Listening to the LastMess episode," which is what I called that podcast, he said, "I realized I had some info in my LastPass vault in addition to passwords, things like SSH private keys, recovery keys for all Apple devices and various cloud services. I didn't even think of rotating all those when I moved off LastPass. I'm also thinking that some of that should not be in my LastPass replacement. SSH keys I'll probably keep on an encrypted disk image on my computer instead. Not sure on recovery keys. I may adopt your two-factor authentication authenticator app and just print out recovery keys and keep them someplace safe."

So I wanted to share this because I think Kevin's observation that just because today's password managers offer us synchronized encrypted vaults, which are capable of holding all kinds of other stuff, does not mean that they are the best place to store unrelated non-web secrets. Password managers, they need our usernames and our passwords and, you know, credit cards; and perhaps some additional automatic form-fill information for convenience. But things like SSH keys, server certificates, and other unrelated non-web passwords don't all need to be stored there in a single place.

Yes, it's convenient. And that convenience can be seductive. But when you combine the wisdom of not keeping all of our eggs in one basket and the old "fool me once" saying, with LastPass being that "once," and when we consider that today's computing environment does not have any more dangerously exposed and complex software, as we saw above, than our web browsers, it would be difficult to make a case for storing more in our browsers than is needed for web-facing activity.

I think a good cross-platform encrypted archiving program is probably a good solution. It can contain all kinds of stuff. It squeezes it down, and then any file synchronization tool, any of the cloud drives or third-party sync systems like Syncthing or Sync.com, can keep the copies synchronized. I'll be interested to hear if any of our listeners have a better idea.

ANT: Yeah, I wanted to talk to you about LastPass because I know where you stand on it, and I've been a LastPass user since before coming to TWiT because, you know, heard about them on TWiT and your recommendation and so forth.

Steve: Yeah.

ANT: And when all of that went down with them, I did consider moving over to our sponsor, Bitwarden.

Steve: Bitwarden.

ANT: I did consider that, and opened an account and started to transition. And then I just found, you know what, I'm just going to change all of my passwords, period. And I spent the weekend, a legit weekend, sitting at a laptop, going through everything and changing all of those passwords. And then I said, you know what, I'm going to go ahead and just keep LastPass. And I feel fine with that. I updated those iterations and so forth. So my thought is, if someone has my vault, I don't think they have my current password. Not to mention there's also the aspect of multifactor authentication that I have on everything.

Steve: Yup.

ANT: So I think I'm fine. And I like to assume that, just because LastPass got themselves in this mess, they probably learned from it and took a lot of losses on that. And it's probably going to make them better with their product going forward.

Steve: I really can't find any fault in that thinking. We know how this happened now. We know that a developer who was working at home had failed to keep his Plex server...

ANT: Plex server; right? Yeah.

Steve: ...up to date. And they got in through a known vulnerability in the Plex server, onto his network, onto his home network. Then they jumped from there, using his laptop connection, through the VPN, into the developer side of LastPass, and then that's how they set up shop. I'm more comfortable knowing how this happened than if it was just a big mystery and they never found out, because we understand the path, and we now know there's no question that they have arranged not to allow anything like that to happen again.

ANT: Right.

Steve: So, you know, I am of the opinion, you know, it's why I didn't jump ship when they first had the first announcement of there being some sort of problem.

ANT: Right.

Steve: I think if you have a very high iteration count, and I think they're now recommending 600,000 or more for LastPass, there's no reason to believe that their technology is fundamentally defective. I don't think it is. And there is reason to believe that they've made themselves more secure. Now, the only caveat I guess I would provide or suggest is that they're owned by a - I can't remember the name of it. It's a monetary fund.

ANT: Oh, like some disease?

Steve: An equity, a private equity fund, that's the word I was trying to remember, a private equity fund. And private equity funds or firms, they're not really interested in the technology.

ANT: True.

Steve: They're interested in profit.

ANT: True.

Steve: You know? They want to turn this thing into money. So one of the ways you do that is you squeeze it. You reduce your staff. You reduce your, you know...

ANT: All that overhead.

Steve: The backup, the off-premises backups. And you reduce this, and you reduce that. So I guess I would suggest that that's a warning sign that, you know, like the original founder of LastPass, he's gone. He's no longer making sure that things continue to be done right. So, you know, the fact that LastPass is owned by private equity, that spooks me a little bit because...

ANT: Yeah, that's fair, that's fair.

Steve: Because they could be cutting corners, and we wouldn't know it until a problem happened. Okay. So Don K. said: "Thank you, Steve. You are the only reason I still use this Twitter or 'X' account. So I can now look forward to deleting it as soon as your email solution arrives."

And Mark Newton said: "I'm one of the few people who may hold out a little while and hope Elon turns this thing around. The boy does have a track record of success. Dropping the name Twitter was almost as dumb as buying the thing. I'll give him another year. It will either be turning around or Twitter/X will be a thing of the past."

Now, you weren't here, but last week I announced that I thought that Elon's statement the previous day, Monday a week ago, that he was going to charge everybody, it would no longer be free, I thought that spelled the end of Twitter. So I explained that I would be putting together an email system in order to send what I'm normally sending by tweet and to create a spam-free means for people to get back to me.

Now, invariably, some people took exception to that. And I heard from people who took exception to what they called my "anti-Elon, anti-Twitter rant." And as I just said, others confirmed what I suspected, which was that the only reason they still had Twitter was to easily receive my weekly Security Now! tweets. Okay. So allow me to clarify all this just a bit.

ANT: Mm-hmm, go ahead.

Steve: I'm currently paid up for a year, since I cannot survive Twitter without TweetDeck. And I am fine with that. You didn't see me, no one saw me budge one inch when everything else that has happened so far, happened. I'm also in no big hurry to leave. My sole concern was that a switch to a 100% paid model - which Leo thinks may never happen, and I understand that, too - would represent a profound change to Twitter. I don't know how big. But, I mean, big enough that Leo says, nah, that'll never happen. I mean, because it would be huge.

I'm already paid up, as I said, so it makes no difference to me one way or the other. But if there are people, and I have heard from many since, who are only on Twitter to obtain

my weekly tweets, then I disliked the idea of any of our listeners feeling that they needed to go "paid" just to continue to receive that Twitter feed from me. I wanted those people to know that there would be an alternative means for obtaining that information.

GRC currently has no means at the moment for reaching any community outside of Twitter. So as I've mentioned before, I had planned, and do, to remedy that once SpinRite is released. So I'll have the ability to mail to lists, and it makes sense for me to have a list for Security Now! listeners of this podcast. So that's where I stand. Not anti-Elon, not anti-Twitter. Just saying, if he's going to force people to pay, and our listeners are already only in Twitter because they want to get my tweets, I will create a different free channel that everyone will be able to use.

ANT: I dig your stance. That's a choice that you have every right to make. If he were to make the platform a dollar a month, you know, which is not a lot, you can decline to use that service.

Steve: Right. And again, as I said, I'm paid up for a year. I'll be happy to keep tweeting the weekly Show Note links, the Picture of the Week, and a brief synopsis of the show's topics. That seems fine. But it no longer seems right for Twitter to be the only way for our listeners, of which there are many, to receive that information.

ANT: Indeed. Indeed.

Steve: Okay. Two pieces of miscellany. Konrad Zydron, he tweeted: "About that HashCheck SHA-256 fork you mentioned in SN-940 last week," he said, "it's here since 2014." And there's a GitHub link in the show notes. And then he also said: "Another fork with additional BLAKE3 algorithm and faster implementations of SHA-256, 512 and SHA3," and another link.

Okay. So remember I was talking about having an easy way in Windows to obtain a useful hash of any file that you right-click on. I talked about HashCheck. Turns out what I learned is that the fork that I was wishing someone had created was indeed created nine years ago. And it's on GitHub.

ANT: Love it when that happens.

Steve: Yes. So I made it, I created a GRC shortcut for it. It's not the shortcut of the week. The shortcut is hashcheck. So the URL is grc.sc, you know, for shortcut, grc.sc/hashcheck. If you use that link, that'll bounce you over to the GitHub release downloads page, and you can grab it. It's digitally signed. I immediately installed it. It's great. It's got all the modern hashes that we want. It's exactly what I was asking for last week. And it's existed for quite some time. So thank you. A bunch of people told me that there had been updates. So I just quoted Konrad's. Thank you very much.

ANT: Nice.

Steve: And finally, last week's podcast was titled "When Hashes Collide." And it's finally safe to say that it made one library information scientist very happy. The guy who inspired the podcast and that title, he wrote: "Hi, Steve!" He said: "I was absolutely floored while listening to the show on Friday morning to hear that my question was turned into a truly fascinating and very useful deep-dive. Our patron population" - that is, of his library - "is a bit bigger than 5,000 individuals. It's actually closer to 400,000." He said: "I did say 'a large public library system in Ohio,' and Ohio is known to be the 'Land of the Libraries,' at least in certain circles. But the topic you presented is still of course very much applicable."

And then he said: "The idea of deliberately using the Birthday Paradox and further obscuring personally identifiable information by manipulating the number of bytes produced by the cryptographic hash is an absolutely brilliant way to add a meaningful layer of protection to the pseudonymized data. Controlling the probability of collisions within the population of patrons is something I didn't even realize I was looking for to give a small degree of uncertainty to the pseudonyms. Once again, an absolutely fantastic explanation of such a complex topic. Bravo."

Okay, well, it's a bit embarrassing to read that, but thank you, Ray. What most pleases me is that Ray "got it" completely. It turned out to be exactly what he was looking for. And he's already turned the concept into working code. He calls it the "Stochastic Pseudonymizer," and for anyone who's curious I have a link to his GitHub project in the show notes. So Ray, thank you for the closure, and for being the incentive and the inspiration for a little bit of deep-dive nerdism, which we had fun with last week.

ANT: That's pretty cool.

Steve: Ant, after you tell us about our third and final sponsor, I'm going to explain to everyone why I titled today's podcast "We Told You So!"

ANT: I've got to tell you, I'm looking forward to that.

Steve: We'll know who told who what.

ANT: I am looking forward to that, sir. Yes, this is another fine week here on Security Now! with Mr. Steve Gibson. Now, Mr. Gibson, so you told us so?

Steve: Not exactly.

ANT: Okay.

Steve: Last Thursday Apple quickly patched three iOS vulnerabilities. Taken together, they created a zero-day chain that enabled a potent attack that The Citizen Lab in Canada and Google's TAG team, their Threat Analysis Group, TAG team, had jointly reverse engineered from the forensic evidence residue left behind by a successful targeted attack aimed at an Egyptian citizen who had made himself a political target by announcing his intention to run in a forthcoming election against Egypt's current head of government.

I named this episode "We told you so!" in acknowledgement of our many listeners who took exception to my assertion that not all websites in the world needed to be HTTPS. What Citizen Lab and the TAG team discovered was that it was when this political target was led to a plain HTTP, non-HTTPS website, that his network traffic was intercepted on the fly by a man-in-the-middle middlebox. That device returned an HTTP 307 Temporary Redirect, which caused his iPhone to be redirected to a spoofed website where the malicious content could be injected into his phone via this chain of three carefully designed exploits. So, indeed, no argument. You told me so.

Let's take a closer look at some of the interesting details here. The target of the attack was Ahmed Eltantawy, a former Egyptian Member of Parliament. Earlier this year in March he announced his intention to run in the upcoming Egyptian presidential election, stating that he planned to offer a "democratic" alternative to the current president, who some feel is not very democratic. Following his announcement, he, his family members, and supporters have been subjected to harassment, including reports of arrests of 12 family members.

Egypt's current President Abdel Fattah el-Sisi has been in power since 2014, when he led the military overthrow of President Mohamed Morsi. Sisi has been widely described as an autocrat. Human rights groups, Amnesty International and Human Rights Watch, have documented widespread human rights abuses under el-Sisi's regime, including repression against civil society groups, activists, and political opposition. So Egypt's current president doesn't appear to be a nice guy who plays fair or would be in favor of free and fair elections.

Ahmed became suspicious about the safety of his phone and reached out to Citizen Lab, who then performed a forensic analysis of his device. Their analysis revealed numerous attempts to target Ahmed with Cytrox's Predator spyware. Citizen Lab has previously documented Cytrox Predator infections targeting the devices of two other exiled Egyptians, an exiled politician, and the host of a popular news program who chose to remain anonymous.

Citizen Lab brought Google's TAG team into the project to perform some of the heavy reverse-engineering forensics which the TAG team excels at. This allowed them to obtain the complete iOS exploit chain that had been targeted at Ahmed. They initiated a responsible disclosure process with Apple, which assigned three CVEs to vulnerabilities associated with the chain. There was CVE-2023-41991, which was a security breach. A malicious app may be able to bypass signature validation. Whoops. 992 in the kernel, a local attacker may be able to elevate their privileges. And 993 in WebKit, processing web content may lead to arbitrary code execution. And there again is a browser exploit, and we were just talking about the problem of browser security.

The zero-day chain was hosted on sec-flare[.]com, and the exploit also contacted verifyurl[.]me. After fingerprinting those two websites, they scanned the Internet and identified a large number of other IPs that matched those sites' fingerprints. In other words, there were other domains that were pointing to the same malware hosting sites, and they considered all of those IPs, and the domain names returned in TLS certificates when they matched the fingerprints, to be directly linked to Cytrox's Predator spyware.

They also observed that some of the domains they'd identified had names suggestive of tailoring towards specific countries or regions, including the Arabian Gulf, Southeast Asia, Angola, the Democratic Republic of the Congo, Egypt, Greece, Indonesia, Kazakhstan, Madagascar, Mongolia, the UAE, and Sudan. So, wow.

By examining the final stage of the iOS exploit chain, which was an iOS payload, they were able to attribute, with very high confidence, that this was indeed Cytrox's Predator spyware, which they'd obtained another sample of back in 2021, two years ago. Those two binaries, the old one and this new one, shared a key similarity which could only be accounted for through a common heritage.

Okay. So what about this on-the-fly interception? This month of September and last month in August, when the target, Ahmed, visited certain websites without HTTPS from his iPhone, using his Vodafone Egypt mobile data connection, he was silently redirected to a website at the domain "c.betly[.]me" via a on-the-fly network traffic injection. And that domain was among those that matched Citizen Lab's fingerprint for Cytrox's Predator spyware.

The injection was triggered by a string match on the website specified in the HTTP Host header, as well as the value of the User-Agent header. I'll note that while the target domain can also be seen at the start of a TLS handshake thanks to SNI (Server Name Indication), subsequent TLS encryption blinds anyone who's eavesdropping from seeing additional connection details such as the User-Agent header, which they use in this case. But since HTTP leaves everything in the clear, the attackers were able to use these

additional signals to reduce mis-targeting since they would not want non-targeted web traffic to be redirected.

So, as I said before, when the injection attack is triggered, the intercepting middlebox immediately returned a 307 Temporary Redirect to the user's iPhone. That would be invisible to the user, and it also, the middlebox, blocks the legitimate reply coming back from the server, which would have come back from the authentic server.

The body of the page which was returned from the malicious redirected website included two iFrames, one which they described as containing apparently benign bait content. It was a link to an APK file not containing any spyware. But the second one was an invisible iFrame containing a Predator infection link hosted on that sec-flare[.]com site.

Next, they wanted to understand exactly where in the network this injection was being made. So they performed a bunch of tricky IP packet logic to localize the malicious middlebox. They needed to get far more sneaky than just using a trace route-style TTL (Time To Live) manipulation because they discovered that another benign traffic management middlebox was also in the path, and it was located close to Vodafone's subscribers. The reason that was a problem is that all the traffic that passed through it had its TTL reset so it was not possible to, from the iPhone, send packets in a trace route style with differing TTLs causing a premature expiration of the packet that would allow you to map out the route. As soon as you hit that close middlebox it reset all the TTLs.

Nonetheless, they were able to eventually determine that the malicious middlebox was on the interconnect between Telecom Egypt and Vodafone Egypt, which is exactly where you'd expect it to be if you wanted it to be able to reliably intercept and infect targeted smartphone users. They were also able to identify the specific hardware by probing it remotely and comparing its responses to other known appliances. It is a commercial off-the-shelf device made by a company called Sandvine who has offices in San Jose, California; Sweden; Ontario, Canada; and India. It's one of their PacketLogic devices, and if anyone's curious, it's sandvine.com/products/packetlogic.

I looked at the data sheet for this device. It boasts features like policy-based traffic management capabilities that include asymmetric traffic control, traffic shaping and filtering, traffic flow classification and prioritization, traffic monitoring, and packet rewrite. In other words, a full spyware capability. Also they said content intelligence enables network operators, cloud providers, and high-speed enterprises to combine the policy management capabilities of PacketLogic with industry-leading URL content categorization functionality. HTTP header enrichment leveraging PacketLogic's stateful awareness and subscriber awareness. Anyway, this is exact - it's a commercial box. You don't need to build your own. You can buy one of these things, stick it anywhere you want, and perform this kind of traffic interception using commercial off-the-site hardware.

ANT: This is just one SKU or multiple SKUs? The website looks like it has multiple SKUs there.

Steve: Oh, yeah. They've got - basically it's how much traffic, what volume of traffic do you want. And so they keep adding interface speed and more interfaces in order to get beefier and beefier devices.

ANT: Got it.

Steve: But they're all part of that same family. So place the device in the link between Vodafone Egypt and the rest of the Internet when you want to be able to mess with smartphone users. Citizen Lab also noted that they're unable to conclude whether this device sits on the Telecom Egypt side or the Vodafone Egypt side of the link. But it really

doesn't matter. They suspect it's within Vodafone Egypt's network because in order to be able to precisely target injections at a specific Vodafone subscriber, you'd require integration with Vodafone's subscriber database. So that also suggests an intimate relationship between Vodafone and this device. So that's also, you know, part of the package. And given that the injection is operating all inside Egypt, that this class of spyware is sold to government agencies, and that Egypt is a known Predator customer, Citizen Lab believes that it's highly unlikely that this could have been occurring outside of the purview of Egyptian authorities. In other words, this was sanctioned.

And as if all of this wasn't enough, Ahmed additionally received three SMS messages as far back as September of 2021, so two whole years ago in September, as well as more recently in May and September of this year. All three posed as messages originating from WhatsApp. The fraudulent messages invited him to visit a link contained within the message to "terminate" what the messages claimed was a new login to his WhatsApp account. So, right, it was meant to invoke an immediate reaction of, oh crap, click the link because somebody else is logged in illegitimately.

Interestingly, approximately 2 minutes and 30 seconds after Ahmed read the first September 15th, 2021 SMS message, the Predator spyware was installed on his phone. The researchers suspect that he did click the message's link, and that triggered the installation. Since the 2023 messages contain nearly identical bait content, they believe that those messages were also attempts to install the Predator spyware on Ahmed's phone. And as if the point hasn't been driven home enough, Citizen Lab's write-up of their research concluded the following.

They said: "Our research reveals the potential insecurities that run through the entire spectrum of the telecommunications ecosystem, including at the network layer, which can be exploited to inject malware on an unsuspecting user's device. Our Internet communications are routed through many networks and middleboxes, some of which can be misused for malicious purposes, particularly if network requests flowing through them are not protected with cryptography. Although great strides have been made in recent years to 'encrypt the web,' users still occasionally visit websites without HTTPS, and a single non-HTTPS website visit can result in spyware infection."

ANT: Just one.

Steve: Just one. "This report should serve," they said, "as a reminder of the importance of achieving a 100% rate of HTTPS adoption." So yes, our listeners told me so. And just for the record, I never suggested that exactly this was not possible. In fact, I outlined exactly this architecture for such attacks. But there's also no question that HTTPS would have robustly blocked this particular attack. It would not have blocked the SMS variant, which we saw did actually infect him two years ago, but certainly the HTTP attack which relied upon on-the-fly traffic interception would have been blocked.

Okay, now, unfortunately, the only way to really prevent such attacks would be to remove HTTP support from the Internet. And that still seems extreme.

ANT: Right.

Steve: I mean, it would break all non-HTTPS sites. You simply couldn't get there. The other countermeasure would be to display very scary warning messages from the browser before it takes you to a link you clicked on. But as we know, users typically just push past anything that gets in their way.

ANT: Exactly.

Steve: So that would be ineffective or at least less effective. Now, another idea might be to begin neutering what web pages delivered over HTTP can do. Like how about not allowing iFrames on such pages. Or how about disabling all scripting on HTTP-only sites. They would still load, but they would be grossly restricted to the sorts of old school content that such sites often have, while potentially dangerous and abuse-prone capabilities would be absent. And this might incentivize the owners of such sites to step up to HTTPS which, after all, is no longer difficult or expensive. You just have to want to do it. And then you can use, you know, any of the free certificates that are being offered by DigiCert or Let's Encrypt or whomever.

ANT: Right.

Steve: So we do have a problem. Somebody got this, got an HTTP link in front of Ahmed. And as we know, now browsers are not showing us HTTPS or P://. Right?

ANT: Right.

Steve: They're wanting to simplify the URLs. So he clicked it. Because it wasn't HTTPS he jumped to - he was trying to jump to maybe a benign HTTP site. That got intercepted, redirected to a malicious HTTP server, which then loaded those two iFrames and compromised his phone using a chain of three still at the time active zero-days which Apple fixed in an emergency update last week.

So I don't know what the solution is. But as our listeners said, "Gibson, you're wrong on this." And yes, they told me so.

ANT: I love that. And I've got to tell you, trying to make the sites less effective, I don't know if that'll be a problem either, considering all the malicious sites could do is just stick a pop-up there similar to what we see now when you go to certain websites and say, "Hey, looks like you're running an ad block routine. Turn that off half a second." Or "It looks like you're disabling this. Can you fix that just for a click in here so we could..."

Steve: Well, remember that the only way they can do that is running script. You can only do a pop-up if you run JavaScript. So my thought was don't have the browser allow any JavaScript on a non-HTTPS site. I mean, really neuter the site so that it cannot be malicious. All it can do is display plain vanilla content, you know, old-school. In that case, you know, sites could still remain HTTP. They just wouldn't - they'd have a very difficult time being malicious.

ANT: Right, right. Unfortunately, I still think that's a bit much to ask for some of the normal folks out here in these Internets.

Steve: What else; you know?

ANT: It's hard to get people to stop using the same password for everything. Now you're telling them to configure their browser? Oh, boy. I don't think it's going to happen. Damn. Mr. Gibson. This has been a ton of fun and a ton of great information. I really enjoyed being here with you today on Security Now!.

Steve: It's great to have you, Ant.

ANT: Learning more and more.

Steve: Thank you for standing in for Leo. Bye.