SECURITY NOW!

Transcript of Episode #894

Data Breach Responsibility

Description: This week we note the release of an updated Firefox browser and Google's welcome and interesting announcement of a super-secure-by-design open source operating system project. We look at the latest cryptocurrency craziness and at a new Windows zero-day which bypasses downloaded executable file security checks. And speaking of zero-days, Apple just patched their iPhone and iPad OSes against their ninth zero-day of the year. We then take a look at the forces driving the evolutionary demise of previously rampant banking malware and at today's critical VMware update. Then, after sharing and addressing some interesting listener feedback, we'll take a look at new Australian legislation aimed at punishing data breaches and consider the ethics of Australia's proposed new heavy fines.

High quality (64 kbps) mp3 audio file URL: http://media.GRC.com/sn/SN-894.mp3
Quarter size (16 kbps) mp3 audio file URL: http://media.GRC.com/sn/sn-894-lq.mp3

SHOW TEASE: It's time for Security Now!. Steve Gibson is here. Google has created the perfect operating system. Is it possible to be absolutely secure? We'll talk about Apple zero-days, Windows zero-days, cryptocurrency craziness, and then who's responsible for that data breach? Is it you, or the technology you're using? All that coming up next on Security Now!.

Leo Laporte: This is Security Now! with Steve Gibson, Episode 894, recorded Tuesday, October 25th, 2022: Data Breach Responsibility.

It's time for Security Now!. Yay! All week long we wait for this show where Steve Gibson is going to explain it all to us. Hi, Steve.

Steve Gibson: Hey, Leo. Great to be with you again for the final episode of September.

Leo: Of October. October.

Steve: October. I don't know what month it is. October, yes.

Leo: It's the spooky season. Do you wear a Halloween costume?

Steve: You know, I think, and I don't have any memory of this, I think I must have been really scared by, like...

Leo: You don't like it.

Steve: I don't like it. When I was like three or four or something, I just must have just had the crap scared out of me.

Leo: Because I have this ketchup hat. And I was going to volunteer - I also have a mustard hat. You could wear that, and we'd...

Steve: We'd be the condiment twins.

Leo: Condiment twins.

Steve: That's really my goal.

Leo: Forget that.

Steve: Anyway, no, I've never been a fan of Halloween. So, but I am a fan of thinking about our industry, which brought the title "Data Breach Responsibility" to this podcast. I encountered, and we'll get to it by the end, the news of a bracing legislation being proposed this week in Australia to really up the ante on the penalties for data breaches following a string of very embarrassing high-profile problems in Australia in the past month. And it was a great opportunity to sort of step back a little bit and think about whose responsibility our data breaches are. And what's neat is that the last several years of this podcast has given us a perfect foundation for having this discussion.

So anyway, this is Security Now! Episode 894 for the 25th of, yes, it is October. We're going to talk before that about some news of a new Firefox browser release, Google's welcome and interesting announcement of a secure-by-design new open source operating system project, which will probably never amount to anything, but it's exactly the kind of thing we need to be, like, taking seriously. So I'm glad they're doing it. We're also going to look at the latest cryptocurrency craziness and at a new Windows zero-day which manages to bypass downloaded executable file security checks, unfortunately. Actually, what's unfortunate is that Microsoft's not so sure it's real because they couldn't reproduce it. Oh, my god.

And speaking of zero-days, Apple just patched their iPhone and iPad OSes against the ninth zero-day of the year, which we'll touch on. We then take a look at the forces driving the evolutionary demise of what was previously rampant banking malware and at today's critical, today as in it happened this morning, critical VMware update. Then after sharing and addressing some interesting listener feedback, as I said, we're going to wrap up by taking a look at this, well, at the downstream effects of really upping the ante on data breach penalties and why that may not be the best thing to do. And we do have a great Picture of the Week which needs some explanation.

Leo: Yeah, because I'm not laughing, I'm just looking. Cool. A great show ahead, as always.

Steve: I think so.

Leo: All right. I'm ready for you to explain this post in the ground.

Steve: So, okay. For those who are not seeing this, we have - in the background is a sidewalk, and then sort of a bricked region which abuts a street-side curb. This was supposed to be a parking meter.

Leo: Ah.

Steve: And the meter is missing.

Leo: Yeah.

Steve: And what's so fun about this, and this is, you know, at the height of geekdom, is that the lower end of this parking meter's mounting pole which the meter would be sitting on...

Leo: Now I get it.

Steve: Is number 404.

Leo: 404, meter not found.

Steve: Meter not found. So Joel Rownak...

Leo: That's geeky as heck.

Steve: ...tweeted me - that is so great. Joel Rownak tweeted this picture. He said: "Steve, I thought you'd get a kick out of this. There's supposed to be a parking meter there. I chose this parking spot because the meter was missing, a real money-saving opportunity."

Leo: Yes.

Steve: "I noted that the meter number was 404. I don't need to tell you what that means in HTTP." So anyway, Joel, thank you for a great listener feedback Picture of the Week. That's a goodie.

Leo: That's a unique one. You're probably the first person to get this one. That's good. I like it.

Steve: I think so. And, you know, probably the last because who's going to...

Leo: You've got to be pretty geeky. I stared at it. I was trying to figure out what - I don't get it. And then, yeah, now that you've explained it, of course, yeah.

Steve: It's good. Okay. So Firefox 106 is out. Last Tuesday, a week ago, Firefox 106 became publicly available with a collection of new features and security improvements. Feature-wise, and there are a couple of cool things here, it's now possible to edit PDFs, including writing text, drawing, and adding signatures. And setting Firefox as the default browser also makes it the default PDF application for Windows. So I thought that was interesting, you know, to be able to actually modify PDFs in a browser. I guess that's good if you need to do that.

Also in Firefox private windows can now be pinned to the Windows taskbar in Windows 10 and 11 for simpler access. And they have been redesigned to increase their feeling of privacy. Now, that's what Mozilla said. I looked at it and, okay, you know, I'm not sure what the change was. They're very dark and very sparse-looking, so I guess that's more private-feeling.

Also, swipe to navigate with two fingers on a touchpad, you can swipe right or left to perform a history back or forward motion. I actually like that on my iPad. I did that once in front of Lorrie, she said, "Wait, what did you just do? How did you do that?" I go, "Oh, well, you know, it's a geek thing." Anyway, it now works for Linux users on Wayland. And Wayland, for those unfamiliar with the term, will be the successor to the original venerable X Windows system, which is how the standard GUI in Unix land which has survived today is being succeeded by something called Wayland.

Also for this version of Firefox on macOS 10.15 and higher, text recognition has been added to images, which allows you to extract text from selected images, like a meme or a screenshot or whatever, and the extracted text is copied to the clipboard in order to share, store, or search without needing to manually retype what you're seeing in the picture, which is kind of a cool thing.

Firefox's WebRTC capabilities also received a major upgrade with their move, or Firefox's move, from their use of libwebrtc went from version 86 to 103, which brought a bunch of improvements: better screen sharing for Windows and Linux Wayland users, lower CPU usage and increased frame rates during WebRTC screen capture on the Mac, RTP performance and reliability improvements, richer statistics, and cross-browser and service compatibility improvements. So just a bunch of nice upgrades to the browser which is all of our favorites, many of our favorites. So it continues to move forward. There are also a bunch of security improvements, nothing breathtaking. I looked through them, it's like, okay, I'll just say that there are a bunch of them instead of enumerating them.

Okay. So driven by events and evidence I've - and I'll be doing this at the end of the podcast also - often bemoaned the increasingly sad state of affairs which describes the insecurity of today's operating systems and software. I've said that someday, somehow, this has to change. So it was with some interest that I stumbled upon an announcement of just exactly this sort of change last week which I wanted to share.

I'm not suggesting, as I said at the top, for a second that this particular effort by three talented software engineers at Google will ever amount to anything more than a learning experience for them and hopefully for others. But even if it never moves beyond GitHub, where it is, this is the sort of shape that an eventual effort will take. And I'm quite clear about one thing. No closed commercial OS is ever going to get us there. All the popular operating systems we're familiar with - Windows, Linux, Unix, macOS - were designed with security in mind, but in a fundamentally ad hoc fashion. That's just the way operating systems have been designed sort of generically until now. They were all designed to have security features, but none of them were provably secure-by-design operating systems.

And, you know, they're all old; right? I mean, that's just the way things were. Security wasn't the issue that it is beginning to finally become today. So it's instructive, I think, to listen to the language these three developers use to describe their effort. It's a different way of thinking, and it's the language of the future.

So they said: "As we find ourselves increasingly surrounded by smart devices that collect and process information from their environment, it's more important now than ever that we have a simple solution to build verifiably secure systems for embedded hardware. If the devices around us cannot be mathematically proven to keep data secure, then the personally identifiable data they collect, such as images of people and recordings of their voices, could be accessible to malicious software.

"Unfortunately, system security is often treated as a software feature that can be added to existing systems or solved with an extra piece of hardware. This generally is not good enough. Our team in Google Research has set out to solve this problem by building a provably secure platform that's optimized for embedded devices that run machine-learning applications. This is an ongoing project with plenty left to do, but we're excited to share some early details and invite others to collaborate on the platform so that we can all build intelligent ambient" - I love their use of this word, this term "ambient" - "intelligent ambient systems that have security built-in by default." And actually they meant designed-in by default.

They finish, saying: "To begin collaborating with others, we've open sourced several components of our secure operating system, called KataOS, on GitHub, as well as partnered with Antmicro on their Renode simulator and related frameworks. As the foundation for this new operating system, we chose seL4 as the microkernel because it puts security front and center. It is mathematically proven secure, with guaranteed confidentiality, integrity, and availability."

Okay. So I'll just note that seL4 is an open source, verifiably secure microkernel with a heritage going back to the mid-'90s. So this is exactly the right sort of foundation for a more functional OS to then be built upon. They also use an seL4 framework known as CAmkES, I guess you'd pronounce it, C-A-M-K-E-S, to provide static, what they said is statically defined and analyzable system components. The CAmkES stands for Component Architecture for micro kernel-based Embedded Systems. It's a software development and runtime framework for quickly and reliably building microkernel-based, what they said was multiserver operating systems.

It follows a component-based software engineering approach to software design, resulting in a system that is modeled as a set of interacting software components. They said these software components have explicit interaction interfaces and a system design that explicitly details the connections between the components. So all of that kind of feeling, this is the language of where we have to be headed. It's utterly different from the essentially ad hoc design of all of today's operating systems that we're familiar with, where random coders grafted on, and are still grafting on today, new pieces of stuff. And, you know, after doing so, away it went. Ship it. And in the case of Windows, we'll find the bugs later.

Anyway, continuing with the Google engineers' description of their work and offering, they said: "KataOS provides a verifiably-secure platform that protects the user's privacy because it is logically impossible for applications to breach the kernel's hardware security protections, and the system components are verifiably secure. KataOS is also implemented almost entirely in Rust, which provides a strong starting point for software security, since it eliminates entire classes of bugs, such as off-by-one errors and buffer overflows.

"The current GitHub release includes most of the KataOS core pieces, including the frameworks we use for Rust such as the seL4-sys crate" - whatever that is - "which provides seL4 syscall APIs" - okay, now I know what it is - "an alternate root server written in Rust," they said, "needed for dynamic system-wide memory management, and the kernel modifications to seL4 that can reclaim the memory used by the root server. And we've collaborated with Antmicro to enable GDB debugging and simulation for our target hardware with Renode.

"Internally, KataOS also is able to dynamically load and run third-party applications built outside of that CAmkES framework. At the moment," they said, "the code on GitHub does not include the required components to run these applications, but we hope to publish these features in the near future.

"To prove-out a secure ambient system in its entirety, we're also building a reference implementation for KataOS called Sparrow" - okay, so Sparrow will be the reference implementation for an instance of KataOS, they said - "which combines KataOS with a secured hardware platform. So in addition to the logically secure operating system kernel, Sparrow includes a logically secure root of trust built with OpenTitan on a RISC-V architecture. However, for our initial release, we're targeting a more standard 64-bit ARM platform running in simulation with QEMU."

And finally they said: "Our goal is to open source all of Sparrow, including all hardware and software designs. For now, we're just getting started with an early release of KataOS on GitHub. So this is just the beginning, and we hope you will join us in building a future where intelligent ambient machine-learning systems are always trustworthy."

So I have a link in the show notes to the project, actually to their announcement on opensource.googleblog.com. But I wanted to share this because, again, this is the sort of effort we're needing to begin to gain experience with the shape of an entirely new class of software which won't be fragile by design. We finally, I believe, have sufficiently inexpensive processing power and memory so that design compromises won't be driven and arguably needed in order to make this sort of next-generation system viable.

We've spoken before about how Windows' original designers, among many other wise decisions which have since been breached, placed Windows GDI, its graphical device interface module, outside of the Windows kernel, up in userland, where its faults would not give attackers access to the kernel. But in their quest for more performance, Microsoft chose to discard security concerns and move GDI into the kernel. As we know, the result has been an apparently endless stream, because they're still coming, of serious vulnerabilities which were created because this extremely complex and error-prone code library, GDI, was given full kernel privileges for the sake of performance. So obvious and clear tradeoff.

Now, it's true that the ivory tower design of an operating system can be compromised. We've seen it. But we can hope that now, given the power, the performance, the memory that we've got, that a system which has been built for the express purpose of exploring and possibly of creating truly secure code will now never need to make such a mistake. So again.

Leo: Do you think that, though, something designed from scratch like this, I mean, obviously it's better to start to be secure by design.

Steve: It's the only way to start.

Leo: But if you have something that, I mean, okay, I'm not going to include Windows in this. But an existing operating system that has had years to be patched and fixed and updated, and then can be hardened, I almost think I'd prefer - I'll give you an example. There's a secure Linux called Qubes, you're probably familiar with that.

Steve: Yeah, of course, right.

Leo: And it's everything sandboxed. It is the Linux kernel, but it's designed to be fully secure. I feel like I almost would trust that better because people have been banging on this open source code for two decades, three decades now.

Steve: So I completely agree with you. If you wanted something usable, yeah.

Leo: Well, that's for sure.

Steve: That's the problem is that this is all ivory tower, pie-in-the-sky, here's how we make something that is absolutely unbreakable. And then someone says, okay, what kind of word processor does it have?

Leo: Right. And even then, by the way, Qubes is not very usable, either. But then, by the way, no one's perfect. Even these academics, they could...

Steve: Well, they don't have - you can't do anything with it.

Leo: Yeah, but they may have accidentally left in something, too.

Steve: No, now, that's the point I want to make. When they say "provably secure," Leo, they mean that it is testably, mathematically - this is a different way of programming. It's like how I'm upset when someone says all software has errors. No. There's no reason that, I mean, software can be perfect. If the hardware is correct, the hardware doesn't have bugs, then software is just math driving the hardware. It can be perfect.

Leo: Is Rowhammer a software error or a hardware error? That's hardware; right?

Steve: Hardware. Absolutely.

Leo: So these wouldn't necessarily be immune to Rowhammer.

Steve: Correct. Correct. So it needs a hardware platform which is secure. And now, you know, you raise a good point. And so is Spectre and Meltdown. Spectre and Meltdown were edge cases where there was a way to bypass some optimizations by leveraging this. But there's a whole class of software problems, all the buffer overflows, all the off-by-ones, all of the use-after-free problems, you know, this whole big batch of things can be fixed.

And so, again, I'm not suggesting for a second that this will be the solution or that in any way this is practical. But we're not going to get to where we need to be if we keep doing things the way we have been because we have been doing them. And we've even been talking about security. Yet we're not, you know, things are not getting any better. And I agree with you. As everyone knows, I just wish Microsoft would stop changing Windows. Leave it alone and fix it. Because the more they add things, everything they do is going to - it brings new bugs into the system. So yeah, the idea of...

Leo: Yeah, it's hard to fix a moving target, absolutely.

Steve: Oh, you can't. It's impossible. All the evidence suggests it. Anyway, I just - I wanted to sort of introduce this because again, this has been kind of quietly percolating in the background. But the problem has always been that when you start to do something with it, the architecture becomes a stumbling block. Just like it did with Windows. Windows had a client-server architecture. And that's what this is. When they talk about a root server, they're talking about the microkernel is a server with clean transactions to clients which are running with no privileges. And there's nothing that the client can do that can make the microkernel make a mistake, as long as the microkernel is written correctly, and there are ways to mathematically demonstrate there are no bugs. I mean none. There is not a bug.

And again, because it is math, it is entirely possible for software not to have any flaws. And so the only way we're going to get where we could be is if we fix our hardware and we fix our software. And this is what the fixed software looks like. It's better to have software you don't ever change and you only fix bugs. I completely agree with you. And you can actually use that for something. You can't use this stuff for anything because it's brand new. And it's not emulating any existing system.

Leo: It probably doesn't have any I/O either, because I/O is - very difficult to prove I/O correct.

Steve: Oh, Leo, you don't want I/O. No, you can't trust any input.

Leo: I/O is nonfunctional, yeah. The side effects are the opposite.

Steve: No, you can't. Who knows what someone's going to send it. No.

Leo: Yeah, right.

Steve: No, you just have to put it in a box and go, ooh, isn't that lovely.

Leo: So I would submit that's a toy. You know. I mean...

Steve: But that's where we're going to start.

Leo: Yeah.

Steve: Of course it's a toy. It is a toy.

Leo: When software's provably correct, it would have to be functional; right? You would have to be able to know that there is no side effects, that what goes in comes out every time the same thing.

Steve: That's what it means. That's what it means. I mean, again...

Leo: That's where I/O gets you in trouble.

Steve: This is the notion that we're discussing this to convey, is that it is not the case that it is impossible to produce a perfect operating system. It is possible to produce a perfect operating system. Now, at the moment it's not usable. But it is perfect. I mean, it is - you can mathematically prove its perfection. So there are mechanisms in place now in the ivory tower for doing that. And Google is saying, okay, let's take that and see what can be done. So again, we're not going to get there unless we do this. And yes, I know this sounds a little bit like SQRL, where seven years ago I said, hey, I know how to solve this problem.

Leo: The difference is, you're not going to do this; right? This is not something you're planning on...

Steve: No.

Leo: Okay.

Steve: No, no, no.

Leo: You've got other fish to fry.

Steve: Believe me, I am deep in the I/O land, Leo. I am seriously in I/O.

Leo: Yeah, you have to - you are I/O. Your whole thing is I/O.

Steve: That's right. That's right. Okay. So, okay. Get a load of - you're going to love this one, Leo - This Week in Cryptocurrency Craziness. The story revolves around an organization known as Mango Markets. And in researching this I learned that .markets is now a top-level domain. So Mango Markets is mango.markets. And let me tell you, this is a crazy place. Its self-description is "Decentralized, cross-margin trading up to 20x leverage with lightning speed and near-zero fees." Okay, I don't know what any of that means. But I do know that it's not good. The top of its home page brags "Long and short everything. Lightning fast. Near-zero fees. Permissionless." Now, I strongly suspect that they didn't mean to be quite as "permissionless" as things turned out to be because a serial DeFi (Decentralized Finance) DeFi abusing hacker by the name of Avraham Eisenberg managed to take $14 million in cryptocurrency...

Leo: What a surprise.

Steve: No, no, that's the beginning - which he had retained from a previous DeFi hack and compound it into $114 million from Mango Markets. Which you might be inclined to say would now be Mangle Markets.

Leo: 1400 million?

Steve: 114. 114 million.

Leo: Oh, 114 million, okay.

Steve: Mango Markets, yeah. Oh, yeah. But wait, this is not - we're not done with it. This gets weirder. Mango Markets' initial tweets immediately following the incident were, and this is @mangomarkets, they said: "We are currently investigating an incident where a hacker was able to drain funds from Mango via an oracle price manipulation. We are taking steps to have third parties freeze funds in flight. We will be disabling deposits on the front end as a precaution and will keep you updated as the situation evolves."

Leo: This is, by the way, exactly why you want to put your money in a bank.

Steve: Exactly.

Leo: No price oracles in a bank.

Steve: Exactly. "If you have any information, please contact blockworks@protonmail.com to discuss a bounty for the return of funds." In other words, they're saying, whoever you are, ouch. How would you like a bounty? And give us back our money.

Leo: Because what he did, he says, was completely legal and allowed by the algorithm.

Steve: Yes. A short summary of what then transpired raised so many questions in my mind that I dug a bit deeper. Here's what the short summary I first encountered read:

"Mango Markets exploiter comes forward. In a series of tweets over the weekend, an individual named Avraham Eisenberg took credit for the attack on Mango Markets following which he walked away with $114 million worth of cryptocurrency. Eisenberg came clean after he was publicly identified as the attacker last week and after the Mango Markets community voted to allow him to keep $47 million of the exploited funds if he returned $67 million back to the platform, so it and all the other projects that depended on it could avoid insolvency." Leo. Wow. What has gone wrong with the world?

Avraham was doxed, and his identity became public knowledge. So he tweeted the following. Get a load of this. This is from @avi_eisen. He said: "I was involved with a team that operated a highly profitable trading strategy last week."

Leo: I should say so.

Steve: Oh, my god. "I believe all of our actions were legal open market actions, using the protocol as designed."

Leo: It's kind of like short-selling; right?

Steve: Yes, exactly. Yes. He said: "Even if the development team did not fully anticipate all the consequences of setting parameters the way they are. Unfortunately, the exchange this took place on, Mango Markets, became insolvent as a result."

Leo: Whoops.

Steve: Yup, whoops. "With the insurance fund being insufficient to cover all liquidations. This led to other users being unable to access their funds. To remedy the situation, I helped negotiate a settlement agreement" - wow - "with the insurance fund."

Leo: He felt bad.

Steve: "With the goal of making all users whole as soon as possible, as well as recapitalizing the exchange. This is similar to how auto-deleveraging works on exchanges such as Binance and BitMEX, clawing back some profits from profitable traders in order to ensure" - just get me out of this, Leo.

Leo: What?

Steve: Thank god...

Leo: Oh, you made too much money. We need some of that back.

Steve: Yeah, we're clawing it back "to ensure all users' funds are protected." Wow. "As a result of this agreement," he finally tweeted, "once the Mango team finishes processing, all users will be able to access their deposits in full with no loss of funds."

Leo: Unbelievable.

Steve: And he keeps $47 million with everyone's blessings. Okay. This wasn't a security breach or a hack in the sense of finding and exploiting a flaw in some code. This guy, I have to say, I've read his blogs, really knows his way around crypto-driven decentralized finance. Mango's first tweet mentions "an oracle price manipulation." So we have some sort of half-cocked DeFi - you know, decentralized finance - system that was prone to manipulation.

And this is the point I've been making here recently: Nothing about this entire wacky DeFi NFT whatever the hell it is world seems mature. That first note I encountered mentioned that this negotiated agreement was made by a vote of the stakeholders. I found, Leo, that voting page. I have the link in the show notes. It's that dao.mango.markets page. And the so-called discussion is somewhat humorous. Many of the lesser stakeholders cannot imagine sanctioning this manipulation of the system. If you scroll down, so that's the agreement there about what the guy's going to be returning. All of those amounts of different cryptocurrencies are being put back.

So, and what was interesting is that there's like people giving thumbs up and thumbs down which are their votes on this. And the voting strength is not per member, but it's by the size of one's wallet. So not surprisingly, those with the most money to lose were in favor of anything that would make them whole again, as opposed to those who didn't have that much in the game and were just furious at the idea that anyone was talking about, I mean, they were thinking this guy is a criminal. But he wasn't. All he did was play by their rules and took them to the cleaners to the tune of $114 million.

So anyway, this Avraham guy has a substack where he has talked about how all this works and explained some of the schemes he's come up with for leveraging fluctuations in the DeFi world. Instead of DeFi, it's "deepfivalue" is his substack, deepfivalue.substack.com. He currently has, I think, four long blog postings. In July of last year his blog was "Deep Dives into Cryptonomics." August 7th of last year was "Anatomy of a Yield Farm." And then in January this year, "How Our Team Makes Millions in Crypto Risk-Free." So I think the term, Leo, is "jumped the shark" for what has happened in crypto land. And I'm not spending any time understanding this craziness because I think, you know...

Leo: Well, the thing is, I mean, it's pretty clear. They're making up the rules as they go along. And there are flaws in it. And somebody takes advantage of it; and, oops, you have no recourse.

Steve: Yup.

Leo: I love it that there were, what is it, 400 million votes in favor of a refund? Quite a few. Quite a few. That's why really if you want to put your money somewhere safe, put it in a bank. Put it in a bank. They've got rules. They're regulated.

Steve: And it's got to be that people think that they're going to somehow make a lot of money?

Leo: Well, that's, I mean, this is what the real basis of all of this is, is greed.

Steve: Uh-huh.

Leo: It's just greed. And, you know, that's what a con man says. I can't rip you off unless you're greedy.

Steve: Yup.

Leo: You can't rip off an honest person.

Steve: Yup. Okay. Let's take a break, Leo.

Leo: All right.

Steve: Talk about a non-greedy sponsor.

Leo: Our sponsors are never greedy. They want to give you things. They're wonderful. Thank you for the opening.

Steve: So yesterday, Bleeping Computer disclosed the news of a new zero-day flaw being exploited in the wild on Windows machines. The flaw enables executable files downloaded over the Internet by attackers to run without producing any pop-up warnings. In this case, the flaw that was found, JavaScript files were being downloaded and used to install the Magniber, M-A-G-N-I-B-E-R, Magniber ransomware. And it turns out that the flaw allows any executable, as I said, to bypass Windows detection warnings.

Now, we've talked about the Mark-of-the-Web. It's implemented. Remember the Mark-of-the-Web for files that come into your computer over the Internet. It's implemented using a very cool and underused feature of Microsoft's NTFS file system which is known as Alternate Data Streams. Alternate data streams have been a feature of NTFS from the start, though they've never received much attention. A file's alternate data stream is its name followed by a colon and then the name of the stream. So it can be literally accessed by creating a file with the filename that already exists, then a colon, and then the name of a stream. It's like sort of a branch off of the root file. It's just that simple. It's another dimension of file metadata, aside from date created, last modified, read-only, compressible and so forth.

And in fact the Windows file directory listing command "dir," you know, just standard dir, can be given the /R command line switch which will cause it to add to its listing any alternate data streams that may exist. So, for example, if you were to go into your Windows Download folder and then open a command line and do dir /R, since almost by default the files coming in have been downloaded over the Internet, you'll see a whole bunch of stuff you've never seen before if you add the /R. You'll see the alternate data streams that have tagged all of those. And those alternate data streams are named Zone.Identifier. So when a file has the alternate data stream Zone.Identifier, this identifies the file as having originated from the scary Internet Zone and causes several things to happen.

One thing that happens is that Windows SmartScreen system is invoked when anyone or anything attempts to run the file. SmartScreen raises an eyebrow and looks to see what it can determine about the file. It examines it. Is this commonly downloaded using some cloud-based comparison system? Examines the signature, is this thing signed? Presumably that matters. And so on.

So in addition, and this is where we were talking about the Mark-of-the-Web earlier, Microsoft Office uses this MOTW flag to determine if the file should be opened in Protected View by any Office apps, which would then cause macros to be disabled as an extra measure of protection. And that's good.

Okay. So what has come to light is that there's a zero-day flaw that's currently under active use. It is being used to install ransomware, bypassing Windows executable file protections. Which has the effect of conveniently, well, conveniently for the bad guys, bypassing all of the Mark-of-the-Web flagging. HP's threat intelligence team reported that attackers are infecting devices with Magniber ransomware using JavaScript files, as I mentioned before. And here's the kicker. The JavaScript files seen distributed by the Magniber threat actors are digitally signed, but the signature is broken.

Will Dormann, whom we've referred to often, he's very active, a senior vulnerability analyst at ANALYGENCE, discovered that the attackers signed these files with a deliberately malformed key in the signature. When any executable file is signed in this manner, even though it's been downloaded from the Internet and received a proper MOTW flag, Windows does not display any security warning. The SmartScreen check never occurs, and the file is automatically executed.

Will further tested the use of this malformed signature technique and was able to create his own proof-of-concept files that similarly bypass the Mark-of-the-Web and SmartScreen warnings. Note that when an unsigned file is downloaded and opened in Windows 10, unsigned, a Mark-of-the-Web security warning is properly displayed. And when a properly signed file is downloaded and opened in Windows 10, a Mark-of-the-Web security warning is properly displayed. It's only when a malformed signed file is downloaded and opened that Windows 10 remains silent and just runs the file.

So this appears to be a bug that was introduced into Windows 10 since Will observed that a fully patched Windows 8.1 machine displays the Mark-of-the-Web security warning as expected, even when it encounters a maliciously broken signed file. And something changed with Windows 11, since the bug does still have an effect when executing files from an archive, but not when running them directly.

It appears that the bug originated from the new feature in Windows 10. It's under "Check apps and files" SmartScreen feature, which is Windows Security > App & Browser Control > Reputation-based protection settings, which first appeared in Windows 10. Disabling the "Check apps and files" causes Windows to revert to its older behavior, where Mark-of-the-Web prompts are unrelated to Authenticode signatures. And then you do get the Mark-of-the-Web caution. On the other hand, then you don't get the SmartScreen filter checking. So it appears to be the signature checking that's resulting in a verification bypass. And again, HP discovered this being used in the wild to sneak ransomware into Windows machines.

And finally, I would say believe it or not, but routine listeners to this podcast will believe it, Will Dormann shared the proof of concept with Microsoft, who said they were unable to reproduce the Mark-of-the-Web security warning bypass, even though both he and Lawrence Abrams at Bleeping Computer had easily done so. So we'll see how this one goes. As far as I know, there's no CVE assigned to this yet. Reported by HP and Microsoft says, oh, well.

Leo: That's hysterical. Was it just some flunky? Maybe some intern answering the phones that day? "I don't know, I don't see it. I just don't see it." It's bizarre.

Steve: Like I said, Leo. I know.

Leo: He published a proof of concept; right?

Steve: Yup. Yup. Showed them how to do it. And they say, yeah, it's not happening for us.

Leo: That's worse than, oh, it's intended. It's intended.

Steve: I know.

Leo: That's like I don't, well, I don't see it.

Steve: Yeah, yeah. Okay. So for the record, I'll note that yesterday, as part of a bundle of security updates, Apple patched and fixed its ninth iPhone zero-day vulnerability of the year. And Apple said the same thing they always say, which is that "It may have been actively exploited," although no one, even Apple, thinks that it wasn't. And of course that's why it's a zero-day.

All we know is that CVE-2022-42827 is an out-of-bounds write flaw reported to Apple by an anonymous researcher and caused by software writing data outside the boundaries of the current memory buffer. So your classic memory overrun. This can result in data corruption; application crashes; or, in the hands of an expert hacker, apparently code execution. And Apple did say that if successfully exploited in attacks, this zero-day could have been used - oh, yes, they're using the past tense - could have been used, because now it's been patched, by potential attackers to execute arbitrary code with kernel privileges.

The complete list of impacted devices includes the iPhone 8 and later, so it's been around for a while; iPad Pro, all models; iPad Air third generation and later; iPad fifth generation and later; and iPad mini fifth generation and later. Apple addressed the zero-day vulnerability in iOS 16.1 and iPadOS 16 with improved bounds checking. So again, nine of those. So where are we? Nine of those by the end of month 10. So they're doing pretty well, less than one a month.

Okay. The evolutionary demise of banking malware. While assembling today's podcast I ran across some interesting commentary which described the various factors that have worked to reduce the prevalence, actually to near zero, of once dominant banking malware. I've edited it for our audience because a lot of it was stuff that we really well know. But here's the gist of what happened, as described.

Researchers from the security firm Mandiant have reported this week that Ursnif, I'll pronounce it "UR sniff," which is also known as Gozi, which we've talked about, or Gozi/IFSB, one of the oldest and last few remaining banking trojan operations that were still active this year, has completely ditched its banking fraud-related features and now appears to operate as a basic backdoor trojan, the type of barebones malware typically used now in Access-as-a-Service schemes that rent access to compromised devices. And of course they're renting them typically to ransomware perpetrators. In other words, just creating and opening doors for others to use is what this once preeminent banking trojan has become. According to Mandiant, the change took place this summer when Ursnif developers started distributing a new Ursnif, which its version was tracked under the codename LDR4.

Mandiant cites several reasons for Ursnif's new radical redesign. At least two leaks of its earlier codebase had occurred. Multiple branches of its authentic codebase had been slowly diverging and were making it harder to support their features across different botnets - oh, boohoo - but also it was an ancient codebase that had finally reached the end of the road when IE was formally removed from Windows.

Mandiant said: "In June 2022, with Internet Explorer finally being fully removed from Microsoft Windows, the RM3 variant was officially seen as a 'dead' malware from a technical point of view, as RM3 was reliant on IE for some of its crucial network communication." And there we see the mixed blessing of IE being allowed to stay around for so long because so many enterprises were utterly dependent upon it for their internal software, which would not run without it, as we've discussed many times.

So it's a surprise that Ursnif lasted as long as it did, operating on a banking trojan model alone. It had become obvious by the mid-2010s that the banking trojans, the banking malware scene was dying, at least on the desktop. Banks, finally growing tired of a decade of thefts from customer accounts, at last rolled out multifactor authentication and transaction verification systems. Though they were not foolproof, these systems did their job and made it significantly more difficult and time-consuming for banking malware operators to steal money from individually compromised accounts.

Back in 2016, Emotet and TrickBot had converted their codebases from banking trojans to generic modular backdoors, and those two were some of the first to do so. And even though they kept their banking modules around, the Dridex and Qbot malware also followed suit later. In all instances, the primary driving force behind this shift in malware economics was the rise of ransomware and enterprise network big-game hunting. As it became clear to ransomware operators that they could extort obscene amounts of money from companies and government networks, they started to look for ways into those networks. And the existing old-time bot networks that had been converted were their way in. This led to the birth and rise of a market for so-called IABs, we've discussed initial access brokers, where smaller threat actors would exploit corporate networking and server gear, plant backdoors, then sell access to these systems to ransomware gangs and their affiliates.

Evil Corp was the first major botnet operator to realize they could use their existing banking trojan to drop ransomware inside the thousands of corporate networks they had at their disposal through the Dridex botnet, and even launched internal teams to write and deploy their own internal forms of ransomware. And recall how often I noted that it was never safe to assume that an infected router would only be used as a proxy to bounce traffic, or as a DDoS agent to flood. Sooner or later, I said, the operators who had established a foothold on these routers were going to turn around and take a look inside the network that they had infected to see what might be valuable there.

Because the Dridex botnet operated on a closed model where its operator was only providing limited access to their botnet to only a handful of very carefully vetted operators, the "service-oriented" Emotet, and later TrickBot, which were open to anyone who wanted to sign up and who had been vetted, cornered the market in Malware-as-a-Service, working with ransomware gangs. And after law enforcement finally cracked down on Emotet and TrickBot, two others, IcedID and Qbot, stepped in as ready replacements after years of having slowly been growing their own botnets in the shadows of Emotet and TrickBot.

The world of underground malware is not difficult to understand. It's simply about the minimum amount of work that can be performed to obtain the largest profit. Banking and carding is now difficult, thanks to banks, and ransomware is easy, thanks to all the bazillion reasons we talk about every week. There's no good reason to run a banking botnet these days, especially one as old and complicated as Ursnif had become. It's far easier to create and manage a simple botnet, spam bored corporate employees until they infect themselves, and then sell access to ransomware or cryptomining gangs for a cut of the profits.

What this does mean, though, the takeaway for us is that the urgency to remove anything that might manage to crawl into your network has never been greater. Stats from the latest intrusion reports indicate that ransomware can be deployed within 30 to 60 minutes of an initial intrusion. This strongly suggests that the deployment of responsive intrusion detection should be front of mind for all security teams going forward. There's just - there's no substitute for watching.

One last bit of news. Today, this morning, VMware released an update to repair a critical, low complexity, CVSS 9.8 remote code execution vulnerability in VMware Cloud Foundation, which is their hybrid cloud platform for running enterprise apps in private and public environments. This 9.8 out of 10 flaw, identified as CVE-2021-39144, was published August a year ago in 2021. The flaw exists in the open source XStream library which is used by Cloud Foundation. And I can't excuse VMware for not having updated this a year ago because this was a well-known flaw in a stream library they're using. And a year has passed. The update happened this morning.

It carries that 9.8 rating because it can be exploited remotely by unauthenticated evildoers in low-complexity attacks that don't require user interaction. In other words, it's easy to do, and it's just a matter of finding a server on the public 'Net and doing it. That's as bad as it can get. And due to the severity of the issue, which has existed, as I said, for more than a year, VMware has also released security patches for other of their end-of-life products which remain in use, which were also using this XStream library for some time. VMware's release advisory said: "Due to an unauthenticated endpoint that leverages XStream for input serialization in VMware Cloud Foundation" - which is NSX-V - "a malicious actor can get remote code execution in the context of 'root' on the appliance."

Okay. So the trouble is a deserialization flaw, and we've discussed many times that data deserialization, which is inherently a data interpreter, is often a source of extremely subtle, yet highly exploitable bugs. The National Vulnerability Database, that is, the guys that maintain the CVS system, says: "XStream is a simple library to serialize objects to XML and back again. In affected versions, this vulnerability may allow a remote attacker who has sufficient rights to execute commands on the host only by manipulating the processed input stream." Leo, I/O, as you were just saying.

Leo: Yup.

Steve: And speaking of flaws from last year, earlier this month VMware informed customers who updated to vCenter Server 8.0, the latest version, that they'll have to keep waiting for a patch to access a high-severity privilege escalation vulnerability disclosed nearly a year ago, last November 2021. So I don't know what's going on with VMware, but these are serious vulnerabilities, and they should be fixing them more quickly.

Okay. I have some Closing the Loop tidbits to share. Nicolas Ross, he said: "@SGgrc In SN-884 you mentioned a piece of software you use to make graphs of SNMP (that's what they're called) counters for your pfSense router. What was that software?"

Okay. So first of all, Nicolas was reminding me that it's SNMP, Simple Network Management Protocol, that was the term I was blanking on during the podcast a few weeks ago when I was talking about monitoring the flow in and out of my pfSense-based router. The fabulous freeware utility that I love for Windows, also for Mac, which I've talked about in the past, is called NetWorx, N-E-T-W-O-R-X. It's by a company called SoftPerfect, and they're at SoftPerfect.com. The version I have is 5.5.5, which is the last of the freeware releases. After a long history of this thing being freeware, when they went to v6.0.1, which is the release after 5.5.5, it became trial ware for $30, and then you buy it for 25 bucks.

In my opinion, it's certainly worth $25. But 5.5.5, which is what I'm using and will continue to use because it's perfect, is available on the web. I just found it on FileHippo. And I have a link in the show notes. The download, just because you always want to do this when you're downloading from a download site, not the origin, the download is digitally signed by a DigiCert certificate issued to SoftPerfect in 2016, signed with both SHA-1 and SHA-256 because they wanted it to work on both old and new versions of Windows. So double-check that, and you can trust it. And I did. I downloaded a copy just last night of 5.5.5 from SoftPerfect and verified the certificate because I knew I'd be talking about it today.

The reason I love this so much is that although it has many features that may be of interest, three things about it stand out for me. First, as I mentioned at the time, it will monitor any interface's SNMP counters. So I have it watching the WAN interface on my pfSense router, which by the way it found for me. I didn't have to dig around. You're able to say don't look at my local interfaces, I want to monitor something else. And it enumerated them for me, and I selected from a list, and it just worked. So they really, they got that right.

Second, it offers, and this is the key, a logarithmic scale for bandwidth monitoring and display, which is so much better than a linear scale, and it's exactly what you want. And I have a picture of a snapshot I made last night where you can see why a logarithmic scale is so good, because that way you can see fluctuations in activity when not much is happening on your network while still seeing maximum-rate connection-saturating transfers without going off scale. I have mine sized, as you can see in the screenshot below, so that the vertical scale snaps to decade powers-of-10 scaling with the max set to 1GB. Since I can pull down 300Mb over my Cox cable modem, that moves the graph into the very last upper scale region without going off scale. Yet when things aren't going along very much, like on this graph I'm also seeing something around just a few Kbits down toward the beginning of the graph, yet back toward the back it's up just shy of 100Mb.

Anyway, love this thing. What the red and green lines on the show notes show is red is always incoming for me. That's something is downloading data into my network, and I want to know what that is, or like that it's happening, at least. And the green is things I'm sending outwards. So anyway, just very cool. I have it, it auto-starts when I boot any of my Windows desktops. It sits running on a monitor. It behaves itself. You're able to set it up so there's no extraneous Windows nonsense. Anyway, just a perfect piece of software. Again, NetWorx from SoftPerfect, N-E-T-W-O-R-X. And get the free one, or get the paid one, whatever you want.

Ando David Roots, he sent me a little dialogue. He said: "She: It's 3:00 a.m. Why are you up?" And then "Me: @SGgrc recommended a book." "She: But you finished the Ryk Brown ones." "Me: It's a new series, different author." And then "She: A symbol" that didn't resolve to anything, but probably a fuming face. Anyway, I'm now on Book 14 of the Silver Ships series. And boy, has it been an interesting ride so far. One of the most interesting topics raised across the series, and it's probably safe to say, though I'm not finished yet, that it's one of the series' primary explorations, is the truly, and I mean this in all sincerity because I don't think we're necessarily a long way away from this, the truly intriguing question of the rights of sentient digital entities. Near the start of the first book we meet Julien, the sentient intelligence that runs a derelict starship. The series' primary human character, Alex, forms a deep bond with Julien, and the books explore many aspects of digital intelligence and the rights of sentient machines. Really interesting. Anyway, I'm loving the series, for what it's worth.

Zacc said: "@SGgrc Can we get a link to the hockey puck-shaped power and reset button you mentioned on the show?" He was talking about the thing that I got from Amazon, which I love, and I have like seven of them, I think now, attached to motherboards and now attached to both of my ZimaBoards, which is just - it's like a little floating hockey puck with four feet where there's a big easy-to-press button in the top, and a little harder to press reset button. And I've got them swapped since I'm most often resetting my hardware. So the big easy-to-press button is the reset button, and the little one is the power button. And it's also got two lights in it, one for power and one for hard drive activity. So anyway, love it. It's less than seven bucks, and the link is in the show notes.

Leo: Attached to the motherboard?

Steve: Yeah, exactly. Yeah, exactly. So you could, normally those little connectors will run out to the front panel of a case.

Leo: Right. I recognize them, yes.

Steve: Yup. And so this allows you to instead extend it out to, like for example, if your computer case is under your desk, or behind your desk, where getting to it is not easy, this basically remotes it. And it's a long cord. I only need it to be, you know, a foot. But I could get six feet or something.

Leo: It's a four-foot coil.

Steve: Yeah.

Leo: Four feet long.

Steve: Oh, okay. So Bart said: "Hi, Steve. I can't remember if it was the last SN episode or the one before, but I remember you mentioning how much faster SpinRite 6.1 can surface test today's larger drives. Was just wondering, is that your own tests or the feedback from the test community? I tried looking for a SpinRite alpha/beta to experiment with on my own drives and PCs, but nothing that would go past the drive detection phase, which by the way is working flawlessly with my particular setups. Have I missed something, or is there no such release yet? Will we be able to get past the dragons?" I actually forgot that he said that.

On all of the SpinRite test releases, if you try to - if you select a menu option after selecting a drive, you then do what SpinRite does to actually start running the drive. Up comes a red cautionary screen with the title "Beyond this Point Be Dragons." So that's what he was talking about. Or here lie dragons or something like that.

But anyway, the numbers I've been sharing are those from the testing community. And it's great to hear that the drive detection phase worked well for Bart, since that's where the past year and a half have gone. Getting to that stage requires all of the new aspects of SpinRite to be working to such a degree that actually running SpinRite is almost anticlimactic. That said, none of the testing releases have gone any further, since that's the next step I'm currently working on, which is also happily also the last step. And actually it's really going very well. In those testing releases, SpinRite is already estimating the total time required to run on a drive by extrapolating from the speed tests it performs during startup. And it does show you those in a scan time column in the various user interfaces, Bart, in case you missed that.

SKIN said: "Love your show. Question, though. Why on earth is SpinRite 6.1 going to be free? Not even an upgrade fee? You're leaving money on the table. We want to support you!" These all had exclamation points. "I think I'm going to buy it again just because of how hard you worked on it."

Okay. So the short version is a promise is a promise, even if it's a promise I made nine years ago. Actually more than, yeah, I think it was nine years ago in 2013. And while that's enough, more than that, I am sure that many people have purchased SpinRite 6.0 based upon the promise that they'd be getting 6.1 at no charge. And while this has indeed turned out to be a much bigger challenge than I originally expected, I'm happy to do it because along the way I discovered some very cool new tricks that I didn't suspect in the beginning, which leaves me to believe that a future SpinRite will have a few tricks up its sleeve that are unsuspected. So there will be a SpinRite v7, 7.1, 7.2 and beyond. So I'm happy to have 6.1 available for free. I basically want to addict everyone to what SpinRite has become and then we'll go from there.

Will Morhler said: "Last week's SN, the discussion of ECB mode of encryption brought back a question I've always had: Isn't any chaining mode of encryption brittle? If any part of the encrypted text is lost, isn't it the case that all subsequent blocks cannot be decrypted?"

You're absolutely right, Will. But as we might say, that's more of a feature than a bug. One of the things we often want to do is detect any tampering of any kind, deliberate or inadvertent, with a file's contents. Huge damage could be done by, for example, adding or removing a zero from financial data. Since the integrity of the data is often just as important as its secrecy, we're often wishing to both encrypt and authenticate. And in the past we've spent some enjoyable time considering whether it's better to encrypt first, then apply authentication to the encrypted result, or to authenticate a file before encrypting it.

And hint: If you cannot do both at once, and some cipher systems can and do, then you'll want to encrypt first and then apply authentication to the result. The reason is that upon reversing the process at the receiving end, or the reading end, you never want to decrypt anything that may have been altered. It's always best to verify that the file has not first been tampered with; and, if so, drop it immediately if it has been altered. And getting back to the point, as a result of that, having brittle encryption that will always cause the file to essentially explode downstream of the point where anything was changed could be considered a good thing. It won't authenticate, certainly. And, boy, it will not decrypt.

And lastly, Ken Dudley asked: "Hello. What transcription software do you use for the podcast?" It is, Ken, a fabulous system known as Elaine Farris, who is located at edigitaltranscription.com. She's not as cheap as software; but, boy, she is a lot better.

And one piece of miscellany. For the first time ever I used Edge to assemble this document yesterday and today. The reason was that Chrome, which I have always been using on my main Windows 7 system, has become laggy for some reason, and it's really become a problem using Google Docs, even just scrolling. I thought that it might be some extension that I'm running, but I'm only running LastPass and uBlock Origin. Although I did note that uBlock Origin believes it's blocked more than 5,000 things in Google Docs. Wow.

Leo: Yeah. Wow.

Steve: So maybe I ought to actually try disabling uBlock Origin, come to think of it.

Leo: It's busy. It's very busy.

Steve: Yeah. So I've been putting up with this problem for weeks, but yesterday the lag was really interfering with assembling this document. So of course I first switched to running Google Docs under Firefox. It was super snappy and ran like a charm, but the Verdana font I like to use, I didn't even bother trying any other fonts because I didn't want to go away from Verdana, had an annoying kerning problem. The letters were not spaced uniformly at all. It was conspicuous. And it turns out I'm not the only person to have reported this. Many others online have posted similar like kerning problems in Google Docs and Firefox looking for solutions. Nothing they tried worked. Nothing I tried worked. So for a while I tried to ignore it, but the wrongness of it bothered me. Finally I thought, I'll use the other Chromium-based browser that I have, that being Edge. Well, Edge worked beautifully, and I wrote and produced this entire show notes document using it.

So here's the thing that caused me to bring all this up: Under Edge and Windows 10, where I was working last evening, but not under Windows 7, the Windows ALT-TAB key sequence switches among recently opened browser tabs in addition to top-level desktop apps. As it happens, for me that's a huge win, since during the preparation of these notes I'm often jumping back and forth between browser tabs. Having seen this behavior under Windows 10, I'm now missing it today under Windows 7, where all I'm getting is top-level application windows, as I always have.

As all old Windows hands know, ALT-TAB is the Windows keyboard shortcut for changing among the desktop's foreground apps. And I'm often using it to quickly jump back and forth among running apps, typically cutting and pasting to move selected content around. But I've never seen it jump between different tabs on the same browser. It must be because Edge actually is running each browser tab in its own process space for inter-tab isolation. So in a sense the browser tabs really are individual browsers. But this was unexpected, and I just wanted to make a note of it.

Leo: I'll just point out you can change that in system multitasking.

Steve: No kidding.

Leo: You can say show Microsoft Edge tabs with snapping or pressing ALT-TAB. You can do ALT-TAB's five most recent, three most recent, or don't show tabs.

Steve: Nice.

Leo: Now, this is Windows 11. I'm not sure what the interface is on Windows 10 or 7.

Steve: I'll bet it's the same. I'll bet it's the same, yeah.

Leo: But I bet it's the same. And yeah, because they're separate processes. By the way, Chrome they're separate processes, too. That's the advantage of Chromium.

Steve: Yeah, yeah. Well, and Firefox has done that. I mean, I don't know if you - have you looked at Task Manager? But, boy, if you've got a bunch of tabs open, half of it is your browser processes.

Leo: They kind of have to do that, I would guess; right?

Steve: Yeah. I think they do, unfortunately. Be nice if they could like nest them and show them in Task Manager. But anyway, they would still be there. By the way, I also wanted to note that Google just announced...

Leo: Look at all my Firefox tabs.

Steve: There you go. Yup. And they are nested.

Leo: Yeah.

Steve: It's a very nice-looking...

Leo: Yeah, again, this might be a Windows 11 thing. I don't know.

Steve: Ah.

Leo: And let's see Edge. Yeah, I only have a bunch of processes under Edge. But I think, I don't know, but I would guess these are all Firefox tabs.

Steve: Yeah, yeah. Very cool.

Leo: Yeah.

Steve: So I wanted to note that Google just announced that Chrome's support for Windows 7 and 8.1 will be ending early next year. I think it was in February. That's one of the dumbest things I've heard recently. Windows 7 still commands 10% of Windows desktops, one of mine being among them, and Windows 8.1 has an additional 2.7%. So that's 12.7%, or more than one out of every eight desktops still running 7 or 8.1, most of them 7 by a huge margin. And the dumbest thing about this is that it must be arbitrary and deliberate on Google's part. There's absolutely no reason why Chrome should care whether it's running on top of 7, 8.1, 10, or 11. If Chrome cannot ignore the version number of the Windows operating system it's running on, then they're doing it wrong.

Presumably, Google's decision to abandon Chrome on pre-Windows 10 machines synchronizes with the end of Microsoft's Extended Security Update, their ESU programs, for Windows 7 and 8.1, which is occurring on January 10th next year. I suppose that at some point I'll need to give up my use of Windows 7 on my primary working system, but I'd rather just keep working. If I have to switch Windows versions, I'm going to suffer a bunch of down time, you know, getting everything set up and reinstalled and going again, which is a constant annoyance for Windows. But it's our life.

So over the weekend a piece of surprising sort of over-the-top news from Australia got me thinking about the nature of data breach culpability and legislative response. Everyone who's been sharing this podcast for a few years will have heard and learned enough to have a well-formed and well-informed opinion about culpability. But not so our legislators. So first, here's the news from Australia.

On the heels of a month of high-profile and embarrassing attacks in Australia - there was Optus, Telstra, Medibank, Woolworths, and EnergyAustralia - three days ago on Saturday, the office of Australia's Attorney General posted an aggressive intention regarding legislation that is slated for discussion and adoption this week. One question that comes to mind is whether what happens in Australia remains in Australia, or will this aggressive legislation just be the first and prove to be the harbinger of similar actions by other governments.

Anyway, so here's what the Attorney General's office published on Saturday. "The Albanese Government will next week introduce legislation to significantly increase penalties for repeated or serious privacy breaches. When Australians are asked to hand over their personal data, they have a right to expect it will be protected. Unfortunately, significant privacy breaches in recent weeks have shown existing safeguards are inadequate. It's not enough for a penalty for a major data breach to be seen as the cost of doing business. We need better laws to regulate how companies manage the huge amount of data they collect, and bigger penalties to incentivize better behavior.

"The Privacy Legislation Amendment (Enforcement and Other Measures) Bill 2022" - which is happening this week - "will increase maximum penalties that can be applied under the Privacy Act of 1988 for serious or repeated privacy breaches from the current 2.22 million penalty to whichever is the greater of $50 million; or three times the value of any benefit obtained through the misuse of information; or 30% of a company's adjusted turnover in the relevant period." Whichever is greater.

They said: "The Bill will also provide the Australian Information Commissioner with greater powers to resolve privacy breaches; strengthen the Notifiable Data Breaches scheme to ensure the Australian Information Commissioner has comprehensive knowledge and understanding of information compromised in a breach to assess the risk of harm to individuals; and equip the Australian Information Commissioner and the Australian Communications and Media Authority with greater information sharing powers.

"The Bill is in addition to a comprehensive review of the Privacy Act by the Attorney General's Department that will be completed this year, with recommendations expected for further reform." And finally: "I look forward to support from across the Parliament for this Bill, which is an essential part of the Government's agenda to ensure Australia's privacy framework is able to respond to new challenges in the digital era. The Albanese Government is committed to protecting Australians' personal information and to further strengthen privacy laws."

Wow. Now, I can see both sides of this issue. For me, it's not as cut and dried as: "Companies who hold consumer data due to valid business needs, and who subsequently fail to keep such data private, are so clearly to blame that they should be penalized with a crippling monetary fine." I made that up, but that's what I mean. In the real world we have this notion of "asking for it." Like, well, yeah, it's too bad that his car broke down when it did, but he hadn't taken it in for service in over 10 years, so he was kind of asking for it. Right?

It seems to me that the underlying cause of a data breach absolutely does matter. Culpability need not be a mystery, and it ought to be a factor. After all, it certainly is a factor in all other aspects of the metering out of what we call "justice." But that means that each issue needs to be taken on a case-by-case basis, just as it is elsewhere in the law. In the case of data breaches, the legislature might not want to do that, and may not know how to do that. But it's bad legislation if culpability isn't factored into the outcome.

If a company is shown to have been running a five-year-old and many times obsoleted instance of, for example, VMware in the cloud, that was riddled with known and long-since-patched vulnerabilities, where one of them was inviting the bad guys into their network to root around and exfiltrate data, then, yes, any reasonable person would find their IT department management grossly negligent, incompetent, and culpable.

And on the subject of "asking for it," which relates to breach responsibility, I would add that when such a breach occurs, thus drawing everyone's attention to the fact and nature of the breach, the leaked information, which would then be public, should be examined to see whether any personally damaging information was being retained in excess of the minimal needs of the business's purpose because doing so should be a big no-no.

But if a company was running only the latest currently patched state-of-the-art updated software, active intrusion detection and network monitoring, holding periodic employee cyber-awareness conduct training, conducting simulated intrusion drills, and randomly sending their own employees test bait phishing emails, and yet despite all of that, due to an entirely unknown zero-day vulnerability in the latest Microsoft enterprise software - or VMware, which we just talked about - someone crawled into their network and exfiltrated a database containing the bare minimum of personally identifiable customer information, how can it possibly be fair to levy a crippling fine of no less than 50 million, and perhaps much more, onto such an enterprise?

The world as a whole is culpable for time and time again having chosen features over security; for having chosen to make our computer systems so overly complicated that they have become unknowable to anyone using them. We're all mostly just hoping that things are going to turn out all right.

This happened, of course, because it's not possible to sell actual security as a feature. Security is the absence of problems. You cannot sell an absence upfront when problems take time to be found and revealed. Way back before the release of Windows XP, when Steve Ballmer was jumping up and down onstage bellowing at the top of his lungs that Windows XP was the most secure operating system that had ever been created, it wasn't possible to prove or to know then just how wrong time would prove that statement to be. I know Steve Ballmer, and I like Steve Ballmer. He's a big huggable teddy bear. But as

Bill Gates would say, he's not a "technical guy." So I imagine that Steve sincerely believed what he was selling. But it didn't matter how much he believed. It wasn't true.

So we've built ourselves an industry where software is intellectual property that isn't sold. It's licensed. And where the agreement to license said software invariably incorporates a statement explaining that the licensor is making no representations of any kind whatsoever as to the operation or fitness of purpose of the software, and where the licensee agrees to hold the licensor harmless for any and all consequences of any failure on the part of the software to function as they might hope. Often going so far, in the case of overly zealous attorneys, as to say that the software's publisher has no idea whatsoever whether the software works, what it does, or what it may or may not do. Sign here.

In such a world, while it is definitely possible for negligence to be the proximate cause of horrendously damaging data breaches, it is also entirely possible for a well-meaning and perfectly well-behaving company to be the blameless victim of the software industry we have collectively built. Until and unless things change, I dearly hope that all future legislation keeps this in mind. And I hope that any organization who did everything right, yet was breached anyway, is able to find a good law firm which is able to make this case for them.

Leo: Bravo. But what would your best guess be as to how many of the breaches are the company doing something stupid like leaving their S3 bucket unlocked?

Steve: Almost all.

Leo: Yeah. So this...

Steve: Almost all.

Leo: So, yeah, I admit, security's hard, and there's companies screwing up. But a lot of these breaches are the fault of the company getting breached; right?

Steve: It is really, really, really difficult to be perfect.

Leo: Yeah, yeah.

Steve: And unfortunately, that's where the bar is. You've got to be perfect.

Leo: Yeah, yeah. Well, welcome to computers. Right? I mean, this is how it's always been.

Steve: You know, I used to accept requests to be an expert witness in jury trials, or even in court trials where I was just talking to a judge. And in fact it was - the last one I did was in New York, explaining to a judge why an NEC competitor MultiSync Monitor - and MultiSync was NEC's trademark. But I was trying to explain to the judge, who had a green oxygen tank sitting next to him...

Leo: Oh, boy.

Steve: ...why I think it was Paradise was the other - was the competing monitor firm on whose behalf I was testifying because they were right that NEC was wrong to sue them over their claim that it would be the last monitor you needed because it adapted to the PS2's differing sync, and you did not need a new monitor. Anyway, they ended up the outcome was wrong. And I decided I'm never going to testify again because it's too frustrating. But if someone comes to me, and they've done everything right, and they're being sued by their government, put me on the stand.

Leo: There you go.

Steve: I will read this column, and we'll go from there.

Leo: There you go. I will defend your right to not be sued for that breach. Steve Gibson, he's right here. You can find him right here every Tuesday, if you need him to testify on your behalf. You may regret that. He comes here every Tuesday around 1:30 Pacific, 4:30 Eastern, 20:30 UTC. We're still in daylight savings time. We will not be in two weeks. So just make a note of that. I know people in the EU have already gone to - ended summertime. But UTC hasn't. We haven't moved ourselves yet. UTC is still the same, 18:30 UTC. Or no, 20:30 UTC. But it will in a couple weeks because UTC never moves, but we do. And because, you know, we have to follow the sun, as it were.

The reason I mention this is so you can watch us live at live.twit.tv. There's live audio, and there's also live video. You can take your pick. Chat with us live if you're watching live at irc.twit.tv. On-demand versions of the shows also available from Steve for free. They have the ads baked in at GRC.com. GRC.com. And he has other things that you might want including those great transcripts from Elaine Farris. He's got 16Kb audio as well as 64Kb audio. He also has SpinRite, the world's best mass storage maintenance and recovery utility. You can get the version 6.0 right now and guaranteed upgrade to 6.1 the minute it's available. GRC.com.

He's on the Twitter, @SGgrc. You can leave messages for him in his DMs. They're wide open, @SGgrc. You can also get a copy of the show at our website, TWiT.tv/sn. That's the free ad-supported version. YouTube also has an even more ad-supported version. And there's also of course the best way probably to get it, subscribe in your favorite podcast player. Our feeds only do the last 10 shows because it's a feed. It's not intended to have every show, all 800, what is it, 800, what have you got, 894.

Steve: 94.

Leo: Yeah, holy cow. But you can go to the website, get all the old shows there, all the way up to Episode 1. And make sure you do because you want the complete set; right? Yes. Steve, enjoy your Silver Ships. You're up to Number 15?

Steve: 14.

Leo: Oh, so you only have six more volumes.

Steve: Actually, it turns out that I found another six after the 24. So it's an even 30.

Leo: I thought Ryk Brown was prolific. Wow.