The Zero-Day Explosion

Description: This week we're going to take a close look at the U.S. Cybersecurity and Infrastructure Security Agency's mandated must update list, including some recent entries. We're going to examine the somewhat breathtaking mistake that Lenovo made across more than 100 of their laptop models, and a cryptocurrency wallet implemented in a web browser. What could possibly go wrong? Then we're going to look at another startling vulnerability that was recently discovered in Java versions 15, 16, 17, and 18. We have a bunch of interesting listener feedback, a brief sci-fi interlude, and the announcement of a major milestone reached for SpinRite. Then we're going to wrap up by taking a look across the past 10 years of zero-day vulnerabilities thanks to some recent research performed by the security firm Mandiant. The title of this week's podcast gives away what's been happening.

High quality (64 kbps) mp3 audio file URL: http://media.GRC.com/sn/SN-868.mp3
Quarter size (16 kbps) mp3 audio file URL: http://media.GRC.com/sn/sn-868-lq.mp3

SHOW TEASE: It's time for Security Now!. Steve Gibson is here. We'll take a look at the CISA-mandated must-update list. Why are there some very old flaws on there? Lenovo has a problem with more than 100 laptop models. A vulnerability in Java 15, 16, 17, and 18 that has a score of as high as 10 critical. Critical. And then we'll take a look across the past 10 years of zero-days. Why are there more now than ever before? It's all coming up next on Security Now!.

Leo Laporte: This is Security Now! with Steve Gibson, Episode 868, recorded Tuesday, April 26th, 2022: The Zero-Day Explosion.

It's time for Security Now!, the show where we're going to protect you, your loved ones, your privacy online with this guy right here, Steve Gibson of GRC.com. Hello, Steve.

Steve Gibson: Yo, Leo.

Leo: How are you?

Steve: Great to be with you again. We're closing out April.

Leo: Ah, the cruelest month of all.

Steve: With a bang. So this is Episode 868. It's funny, you were talking to the MacBreak guys and wondering how long they'd been doing their show. And they were like, what was it, 815 or something that you just did with them?

Leo: Yeah. You're ahead of the game.

Steve: We're at 868, so not a whole year, but nearly that.

Leo: Yeah. You were the second podcast we did on TWiT was Security Now!, yeah.

Steve: Yes, yes. Well, and it was in Toronto that, in between shooting the four episodes that we were doing of Call for Help, you were leaning on the stage prop, and you said, "What would you think about doing a..."

Leo: Hey, Steve.

Steve: "...podcast on security?" And I said, "A what cast?"

Leo: And the rest is history.

Steve: And that was nearly 18 years ago, and both of us had darker hair.

Leo: And we have no lack of material, either.

Steve: And more of it. Apparently not. So, and speaking of which, that's a perfect segue because today's episode is titled "The Zero-Day Explosion."

Leo: Ugh.

Steve: Uh-huh. We're going to take a close look at the U.S. Cybersecurity and Infrastructure Security Agency's - boy, that's a mouthful - mandated must-update list, including a couple of recent entries. We're going to examine the somewhat breathtaking mistake that Lenovo made across more than 100 of their laptop models, as well as a - and this is separately - a cryptocurrency wallet implemented in a web browser, which is where we all say in unison, "What could possibly go wrong?" Then we're going to look at another startling vulnerability that was recently discovered in Java versions 15, 16, 17, and 18, all the last four major versions.

We've got a bunch of interesting listener feedback, and one, I only have one representative sample of something that 200 of our listeners must have sent. We really have amazing listeners who are paying attention. I've got a brief sci-fi interlude, the announcement of a major milestone reached for SpinRite, and then we're going to wrap up by taking a look across the past 10 years of zero-day vulnerabilities thanks to some recent research performed by the security firm Mandiant. And of course the title of this week's podcast gives away what's been happening, "The Zero-Day Explosion."

Leo: Yeah. We've been seeing it in action. It's incredible.

Steve: Yeah, we've been talking about zero-days more in the last year or two than ever before.

Leo: Yeah. I'm curious, well, we'll talk about it. I'm curious what your thoughts are on why it's happening, what's going on, yeah.

Steve: Yeah, we do talk about that.

Leo: I figured you would. And now, the Picture of the Week.

Steve: So this relates to our discussion a while ago about privacy enforcing or pro-privacy web search, of course. DuckDuckGo is the service. And I actually found myself thinking about it recently. I clicked on some link from a Google search, and where I was going was blocked by uBlock Origin because I was like in some redirection chain.

Leo: Yeah, I've had that happen, yup.

Steve: Bouncing, oh, it's just so wrong. It's like, I'm looking at the link, right, and it's a fakeout link. It's not the actual URL that you get when you click on it. So I just thought, oh, and the sites that I was being bounced through that uBlock Origin was protecting me from were really sketchy-looking. It's like, this is a link on Google Search.

Leo: And you go through all those other things. Yes.

Steve: Yes, yes. So I thought, well, DuckDuckGo. Anyway, the problem is, and we discussed this at the time, you can say to somebody with a straight face, you know, did you Google it?

Leo: Or Bing it.

Steve: Yeah, Bing, yeah.

Leo: Not quite so straight of a face, but...

Steve: But Duck?

Leo: Did you Duck it? Oh, don't say that.

Steve: Did you Duck it? No. You don't want to - that's too close to, you know. And so then it was ducking or, you know, like what?

Leo: That's bad, too.

Steve: I don't know. Anyway, so somebody tweeted me a picture that is, because of what it is, it's got, as you said...

Leo: It's a meme.

Steve: ...a common meme of someone like saying, no, no, no to DuckDuckGoing. You don't want to say DuckDuckGoing. That's clearly not helping. Anyway, they're suggesting "quacking."

Leo: Oh, I quacked it. I quacked it, yeah.

Steve: So I quacked it.

Leo: Yeah, quack me.

Steve: Do did you quack that? Yeah, quack it.

Leo: Yeah. I like it.

Steve: Just go quack it.

Leo: Yeah.

Steve: So thank you. I think that probably solves our problem. You don't want to duck it.

Leo: Is that Drake? I feel like that's Drake. But I might be wrong. I don't know.

Steve: I don't know who Drake is.

Leo: The chatroom will tell me if I'm right, yeah.

Steve: There's something on his shirt that looks like a cheerleader or, man, I don't know what's going on.

Leo: No, it's hoops, man, that's Nike Air or something, yeah, Air Jordans. He's a hipster; right? That's Drake.

Steve: Well, you were certainly right about me not knowing about the meme.

Leo: He's a well-known Canadian rap star, if that's not an oxymoron.

Steve: Yeah, well, that doesn't help either. Okay.

Leo: Oops. I'm pressing all the buttons, but nothing's happening.

Steve: You've got all kinds of buttons over there.

Leo: There we go. There we go.

Steve: Okay. So CISA's known exploited vulnerabilities catalog. As we've noted from time to time, one of the services being provided by our awkwardly named U.S. Cybersecurity and Infrastructure Security Agency, CISA, has been the maintenance of a...

Leo: You know, I never noticed the redundancy there.

Steve: Oh, it's so bad. And I was thinking maybe, I was playing where if you said Instructure and Cybersecurity Agency...

Leo: That would work.

Steve: ...then you get rid of one of the securities. Because otherwise it's Cybersecurity and Infrastructure Security.

Leo: You would write a letter to the Department of Redundancy Department because I think they need to fix this, for sure.

Steve: Actually I'll write another letter to the Department of Redundancy Department.

Leo: Do at least two.

Steve: Yeah, because they wouldn't respond to only one. So, okay. So this is a growing list of actively, and this is the key, actively exploited vulnerabilities, not just all vulnerabilities, but the ones that are, like, being seen. So we normally refer to this list in the context of CISA's what I like to call their "Christmas cancellation policy" from last year which is of issuing standing mandates to all federal agencies within its governance

that they must patch this or that by some specific date. Typically only a few weeks after the issuance of a mandate.

Now, at the end of today's podcast we're going to be talking a bit about patching philosophy and about the idea of, or the feasibility, I guess, of prioritizing patching to focus upon those issues that are being actively exploited. The logic behind that's obvious. In a bureaucratic environment it certainly makes the imposition of the inconvenience that's suffered by the need to take down and patch running systems and services justifiable.

But boy, CISA's list is growing now so large that it's being referred to as a "catalog," which is a better description than a list. So at some point it loses some of its punch as it becomes easier just to patch everything, which, as we'll see, is the strategy that I think makes the most sense overall.

That said, there have been some notable new entries added to CISA's constantly growing catalog of "must patch immediately" mandates. CISA informs us that the CVSS 7.8 vulnerability in Windows Print Spooler that was patched as part of February's Patch Tuesday okay, so a little over 60 days ago. We've had March and April patches since then. But what they fixed in February is actively being exploited in the wild. It's a privilege escalation vulnerability which a hacker will need to leverage if they're able to arrange to get into a system, but under limited protective privileges. They need to escalate their privileges in order to perpetrate their nastiness.

So here's a perfect example of a more than 60-day-old patched problem, right, I mean, patches are available, and it's even on Windows, that's like aggressively broken down all of our resistance against applying patches. So it's like, okay, fine. Sixty days ago. It was doubtless immediately reverse engineered and put to use. The only place it's going to be effective is against machines that have not yet received their February updates. Yet CISA says this thing is being exploited in the wild. That's what's happening. Back in February when Microsoft listed this defect as fixed, it was tagged as "exploitation more likely" and they were apparently right about that.

It's interesting, and I think somewhat sad, to look at the CVSS year dates for things the CISA adds to its "actively being exploited in the wild" catalog. There were two others. One was they just added a cross-site scripting vulnerability which was found in the Zimbra - I had to look it up - the Zimbra Collaboration Suite. It's certainly not mainstream. It's a Java-based suite which is hosted on Linux. It's been around since 2005, and its ownership has changed hands many times. I don't know what that says about it. I have a relative who changes jobs often, and that kind of is a little sketchy.

Anyway, this thing was first written by a company originally named LiquidSys, who changed their name to Zimbra, which Yahoo! later purchased, before selling it to VMware, who then sold it to Telligent Systems, who then also changed their name to Zimbra before being sold to Synacor. So I guess a lot of companies had great hopes for it, or maybe the price was right. I don't know. But in any event, this cross-site scripting vulnerability was identified and patched four years ago, in 2018, and CISA now tells us that it is currently being actively exploited in the wild.

And guess where? Ukraine's Computer Emergency Response Team, somehow still in operation, CERT-UA, released an advisory last week cautioning about email phishing attacks targeting government entities with the goal of forwarding victims' emails to a third-party email address by leveraging exactly that Zimbra vulnerability. So, okay. It's not just theoretical. As CISA said, it's actually happening right now. Some Russian miscreant saw that someone was still using Zimbra. So they checked their own catalog, which is titled "The Big Book of Every Possible Way to Hack Someone," and found a four-year-old cross-site scripting vulnerability that would come in handy if that Zimbra

instance had not been updated in the past four years. The most recent Zimbra update was just a little over a year ago, on April 7th of 2021. So Zimbra could have been kept current, that instance that was apparently found being hacked. But apparently it hadn't been.

So CISA's not wrong that this four-year-old obscure vulnerability is being exploited. But are they right to add it to the U.S.'s emergency "must patch by May" mandate? That's what I'm wondering about. Are any U.S. federal agencies known to be using Zimbra? I have no idea. But if so, wouldn't it make much more sense, because it's got to be, like, what, one or two, if any? Wouldn't it make more sense to be a little bit more proactive and give them a call? I mean, like, target those agencies rather than everyone else with an entry that just adds unnecessary noise to the growing list? And this thing's four years old.

Another just-added entry is a three-year-old stack buffer overflow - now, stack buffer overflows, they're not good. This one carries a CVSS of 9.8, which as we know, those are the ones you pay attention to. That's a "stand up and take notice" score. And it occurs in WhatsApp's VoIP component. How is it possible that anyone using WhatsApp will not have updated it since 2019? I mean, this thing's a three-year-old problem that was patched.

So Leo, I guess that perhaps we're living in a bubble. That's the only conclusion I can come to. Perhaps we've been drinking our own upgrade Kool-Aid for so long that we're completely out of touch with the real world. There must be a significant proportion of users who actively and proactively ignore, or perhaps mistrust, or just don't think they need offers and requests to update their software. You know, it's working and not obviously broken. So they think, I see no need to mess with it.

And as we know, it certainly is the case that having something work and having something that works also being actively impervious to abuse are two very different things. But we know that. That's one of the most important lessons that everyone listening to this podcast, myself included, has learned through example after example through the years. But that's probably not at all obvious to the typical user who thinks, well, it works. I can message and talk, and it seems fine here. So whatever they're trying to sell me or to get me to do I probably don't need or want. So I'm not going to change anything.

But again, even so, does a three-year-old vulnerability, maybe if someone in Bangladesh had their Android phone compromised as a result of using an even older version of WhatsApp, does it need to be taking up cognitive space in CISA's catalog? I think it's an open question.

Three or four years is a long time not to have updated software. It's clear that there's a very long tail on many of these vulnerabilities. We talked last year and the year before about critical flaws in a TCP/IP stack being widely used by $5 Internet of Things light switches and plugs. None of those things are ever going to be updated. So they will be latently vulnerable as long as they're in service. But that isn't what CISA is targeting.

I don't have an answer. Light switches and plugs cannot be updated. But for federal agencies to never update in-use software for which updates are ready and waiting is unconscionable. So I'm glad there's a mandate. I hope it has some teeth behind it. But as I was thinking through this, I thought, you know, maybe have these things expire after a year, but like get agencies, like force them somehow not to be more than a year out of date. And they shouldn't be more than, you know, in some cases a few days out of date. The rate at which we're now seeing patched vulnerabilities leveraged into exploits has accelerated dramatically in the last couple of years, and we're going to be touching on that shortly.

Lenovo. Leo, I heard you refer to Lenovo's UEFI problems on some podcast recently. So this has been in the news a lot.

Leo: Oh, yeah.

Steve: It's not surprising.

Leo: I'm aware of it because I buy a lot of Lenovo hardware.

Steve: Yes. That's been the ThinkPad; right?

Leo: Yeah, yeah, love the ThinkPad, yeah.

Steve: It's like it's been, yes, the premier laptop. So as we know, when a PC is powered up, something needs to wake up and configure the various parts of the machine. The video needs to be started. The fans need to spin up. All of the machine's various mass storage subsystems need to be initialized. And then the firmware's configuration needs to be checked, the proper operating system needs to be located, and its OS boot code needs to be initially loaded into RAM so that control can be turned over to it to continue booting the machine.

The first PCs did that using their Basic Input Output System, B.I.O.S., or BIOS. That was good for about five years. It actually didn't last very long because the PC just exploded in terms of what everybody wanted to do with it. So the limitations which had been built into the BIOS's assumptions began to cause more problems than they were worth almost automatically. And various Mickey Mouse workarounds were created to overcome many of these problems while Intel worked on a wholesale replacement of the BIOS. The initial attempt was the EFI, the so-called Extensible Firmware Interface, which quickly matured into the Unified Extensible Firmware Interface, UEFI.

And we find ourselves right back where we always do. The original BIOS was so dumb that it could not be infected. It was originally implemented in...

Leo: Sometimes dumb is a good thing.

Steve: That's exactly. It was originally implemented in masked ROM, meaning that the firmware's bits were etched into a metal mask at the factory and could never be changed. It did mean you had to get the code right the first time.

Leo: No updates, yeah.

Steve: And that was something people used to be able to do. But we don't do that anymore. So that soon gave way to non-volatile FLASH ROM, which could be updated, but the code it implemented was still blessedly dumb. Sometimes, for some things, the dumber the better. Because if all you want is to boot an OS, you don't really need that much smarts. The BIOS did it just fine. And the lesson we keep falling into and we keep

failing to learn is that the more complicated, fancy, capable, and "smart" we make things, the more leeway and latitude the system has to go very badly wrong.

So welcome to the Unified Extensible Firmware Interface where malware is also able to "extend" the firmware. Lenovo has been most recently in the "We made a UEFI mistake" news recently. Last week, the guys over at ESET, whose motto is "We Live Security," posted the results of their analysis of some widely used Lenovo UEFI firmware. Their posting's title was "When 'secure' isn't secure at all: High-impact UEFI vulnerabilities discovered in Lenovo consumer laptops." And the story's tagline is "ESET researchers discover multiple vulnerabilities in various Lenovo laptop models that allow an attacker with admin privileges to expose the user to firmware-level malware."

Okay. Firmware-level malware. That's not what you want to hear. That's even less what you want to have crawling around inside your machine. Firmware-level malware enables the ultimate in rootkit techniques, in fact having its own worst name, "Bootkit." The presence of firmware-level malware means, quite simply, that it's impossible to trust anything about what the machine might do. Firmware-level malware is able to infect and compromise the operating system's own code during its boot process before it has had any opportunity to raise its own shields. And reformatting the machine's mass storage and reinstalling an operating system or even removing and replacing a drive won't necessarily eliminate the problem because this malware has taken up residence in the machine's underlying firmware on the motherboard, on a non-volatile memory soldered to the main board.

Now, we know that anybody can make a mistake. And as our listeners know, I am infinitely forgiving of mistakes. But the most troubling aspect of what the ESET researchers found was that two of the three big mistakes Lenovo made were the oversight of leaving highly exploitable drivers in the UEFI firmware image which should have only been present during the firmware's development. These drivers should have never left the factory. So it's not like they got a loop condition wrong or something like a mistake. They left stuff in there that should not be in there. How do we know? We know because the two drivers were actually named "SecureBackDoor." In the UEFI firmware that's the driver's name. Yeah, we're going to...

Leo: Talk about an oxymoron. SecureBackDoor.

Steve: Yeah, yeah. Turns out it wasn't.

Leo: Yeah.

Steve: And the other one was "SecureBackDoorPeim." So here's what ESET said. They said: "ESET researchers have discovered and analyzed three vulnerabilities affecting various Lenovo consumer laptop models." Various. Yeah, we'll get to that in a minute. "The first two of these vulnerabilities" - and we've got two CVEs from this year, 3971 and 72 - "affect UEFI firmware drivers originally meant to be used" - this is ESET - "only during the manufacturing process of Lenovo consumer notebooks. Unfortunately," writes ESET, "they were mistakenly included also in the production firmware images without being properly deactivated [/or deleted]. These affected firmware drivers can be activated by an attacker to directly disable SPI flash protections" - that's using Control Register bits and Protected Range registers - "or the UEFI Secure Boot feature from a privileged user-mode process running OS runtime."

Okay. So just to be clear about what ESET just said. They said: "From a privileged user-mode process in the OS." In other words, a user, any user of these laptops mistakenly allowing some malware to run in their OS, which might innocently ask to be granted brief UAC privilege elevation to install something, that is, if it didn't bring along its own privilege escalation vulnerability exploit, as it might; or which might set itself up to run as a system service. That code can disable all relevant UEFI write protections to then surreptitiously install semi-permanent hidden bootkit malware into the system's UEFI firmware. And the user would be none the wiser. And we don't know how to scan for that yet. I mean, there's been some talk of scanning UEFI. Nothing much has come of it.

ESET said: "It means that exploitation of these vulnerabilities would allow attackers to deploy and successfully execute SPI flash or ESP implants, like LoJax. To understand how we were able to find these vulnerabilities, consider the firmware drivers affected by" and then the CVE number 3971. They wrote: "These drivers" - imagine this, Leo - "immediately caught our attention by their very unfortunate, but surprisingly honest names: SecureBackDoor and SecureBackDoorPeim. After some initial analysis, we discovered other Lenovo drivers sharing a few common characteristics with the SecureBackDoor* drivers." Those are Chg - I guess that's short for change - and then BootDxeHook and ChgBootSmm. SMM is system management mode stuff, which is the OS under the OS. "As it turned out," they write, "their functionality was even more interesting and could be abused to disable UEFI Secure Boot." That's the CVE ending in 3972.

"In addition," they said, "while investigating the vulnerable drivers, we discovered a third vulnerability: SMM memory corruption inside the SW SMI handler function." Thus we have CVE ending in 3970. "This vulnerability," they said, "allows arbitrary read/write from/into SMRAM, which can lead to the execution of malicious code with full SMM privileges" - that's, again, that's like the chip-level privileges, nothing more privileged in the world than that - and, they said, "potentially lead to the deployment of an SPI flash implant.

"We reported all discovered vulnerabilities to Lenovo on October 11th, 2021." And I didn't have it in the show notes, but Lenovo responded a month later. "Altogether the list of affected devices contains" - and here it comes - "more than 100 different consumer laptop models, with millions [many] of users worldwide, from affordable models like IdeaPad 3 to more advanced ones like Legion 5 Pro or Yoga Slim 9. The full list of affected models with active development support is published in the Lenovo Advisory.

"In addition to the models listed in the advisory, several other devices we reported to Lenovo are also affected, but won't be fixed due to them reaching End Of Development Support (EODS). This includes devices where we spotted reported vulnerabilities for the first time: IdeaPad 330 and IdeaPad 110. The list of such EODS devices that we have been able to identify will be available in ESET's vulnerability disclosures repository." And what this tells us, reading between the lines, is that these vulnerabilities have been there long enough for those machines which they started affecting to now have left, have gone out of their service life with Lenovo. Thus they will never be fixed.

Oh, yeah, I do have in the notes Lenovo confirmed the vulnerabilities on November 17th, 2021, and assigned them the following CVEs. And, I mean, they're coming right out with it. CVE ending in 3970, LenovoVariableSmm - SMM arbitrary read/write. The one ending in 3971, SecureBackDoor - disable SPI flash protections. And 3972, ChgBootDxeHook - disable UEFI Secure Boot.

So given how incredibly active the cyber underworld is today, we keep encountering quite sobering evidence of it, you know, in every podcast now. There's just no chance that these now fully disclosed and very well-documented vulnerabilities will not be used to compromise the interests of some of these millions of Lenovo laptop users worldwide.

And many of them are going to be serious users. It will happen. So here we are once more noting that there's something very wrong with our industry's current development model. How can this be allowed to occur over and over and over? ESET had to reverse engineer the proprietary code in this UEFI firmware in order to find these problems. And it's affecting lord knows what multiple of millions of Lenovo laptop users.

Lenovo messed up big-time here, but for the record, they're not alone. These newly disclosed vulnerabilities merely add to the recent disclosure of more than 50, five zero, UEFI firmware vulnerabilities which had been found in Insyde Software's - I-N-S-Y-D-E - Insyde Software's InsydeH2O, and HP and Dell laptops since the start of just this year. Among those are six severe flaws in HP's firmware affecting both laptops and desktops which, when exploited, could allow attackers to locally escalate to SMM privileges, which as I said is as much as you can get on any hardware platform, and trigger at least denial of service and maybe more. So Lenovo is in good company, or at least only the most recent member of this UEFI vulnerability dog house. And as we know, it's not Lenovo's first instance of UEFI problems. Years ago they've also had problems.

So we've managed to make our lovely little machines far more complex by designing in extremely powerful capabilities. Yes, we get lots more flexibility. We get remote management and remote maintenance. And not surprisingly, it's also a mixed blessing. So a heads-up to anyone using Lenovo laptops. Regardless of the model you have, don't look at a list of affected models. First of all, there's hundreds. You should definitely check in to see whether your device has a firmware update outstanding. And for that matter, HP and Dell users would be well advised to do the same.

Leo: Do you think these changes are driven by the needs of enterprise? In other words, are we personal and home users and geeks suffering because...

Steve: Yes, exactly that.

Leo: ...[crosstalk] management capabilities built in for any of us.

Steve: Exactly that. Exactly that, Leo.

Leo: There should be, and there are a few, places where you can get simpler systems with simpler UEFI and core boot open source firmware, things like that. And they are really not aimed at enterprise. What was the other thing I wanted to mention? Oh, yeah, firmware updates now, it's interesting, are increasingly part of the operating system update. I don't know if you've noticed that.

Steve: Yeah. Well, we know that Windows, for example, is patching the Intel chipset firmware. Linux brings along the same thing. And to their credit, although it is a little bit of a mixed blessing, Lenovo now has software that comes pre-installed on their machines which is taking responsibility for keeping your machine's firmware up to date. So it makes it better than never ever having the opportunity to proactively inform Lenovo machine owners and having a problem like this out there that would make them persistently vulnerable.

Leo: Yeah, yeah.

Steve: Okay. So I read the title of this piece of news in The Record, and it just made me shake my head. The item is titled "Everscale blockchain wallet shutters web version after vulnerability found."

Leo: Okay.

Steve: I mean, really?

Leo: I'm going to put my wallet on the web. That's a good idea.

Steve: What moron could possibly think that offering a web browser-based cryptocurrency wallet was sane?

Leo: Well, it's easy. It's convenient.

Steve: Anyone who was capable of beginning to create such a thing should know it's just a bad idea. As we've often observed on this podcast, just because you can do something doesn't mean you should do that. Here are the first two sentences of The Record's story. They wrote: "The company behind Ever Surf, a wallet for the Everscale blockchain ecosystem, is shuttering its web version after a vulnerability was found by Check Point researchers. The Ever Surf team confirmed that the vulnerability allowed attackers to gain access to wallets." Uh, yeah.

Leo: Duh.

Steve: Because it's on a web browser. Oh, my god.

Leo: Uh, yeah.

Steve: Okay. The Record is reporting on research which was performed by Check Point Research. The Check Point guys explained. They said: "Blockchain technology and decentralized applications provide" - and decentralized applications are, you know, web apps - "provide users with a number of advantages. For example, users can utilize the service without creating an account, and it can be implemented as a single-page application written in JavaScript." They're being very fair here. "This type of application does not require communication with a centralized infrastructure, such as a web server, and it can interact with the blockchain directly or by using a browser extension like MetaMask.

"In this case, the user is identified using keys that are stored on a local machine inside a browser extension or a web wallet." Okay, now, the phrase "web wallet" itself should be outlawed. But okay. "If a decentralized application or a wallet stores sensitive data locally, it must ensure this data is reliably protected. In most cases, decentralized applications run inside the browser and therefore may be vulnerable to attacks such as cross-site scripting," just to name one of, like, countless.

"This research describes the vulnerability found in the web version of Ever Surf" - maybe we should call it Never Surf - "a wallet for the Everscale blockchain." They finish: "By exploiting this vulnerability, it's possible to decrypt the private keys and seed phrases that are stored in the browser's local storage." Yeah. "In other words, attackers could gain full control over victims' wallets."

Okay, now, Leo, you're going to love the details of this. Okay. It turns out that one of the code libraries the implementers used - you know how everybody now is just grab a library here, grab a library there and hope that it hasn't been compromised by some supply chain attack, which is another problem. One of the code libraries the implementers used is not fully supported, or one of the functions in one of the code libraries is not fully supported in web browsers. The code attempts to obtain a cryptographic nonce with a call to the function "DeviceInfo.getUniqueId." The problem is that this function requires access to its underlying device, so it's only defined when running natively in Java on Android, iOS, or Windows. I have a snippet of the function, actually it's the entire function, it's a one-line function because JavaScript is crazy with the way it operates.

Leo: I would never write a function like this. This is ridiculous.

Steve: I know. You're able to chain a bunch of ors, and the first or that succeeds is the one that gets taken as the value of the enclosing function.

Leo: You can't really see it very well on the screen, not from the show notes.

Steve: No, you really can't see it. Anyway, I've got a snippet of it. What it shows is, for those who read JavaScript, that it is obtaining a value of the underlying platform's default dot unique ID, if there's an evaluation of the OS as Android, iOS, or Windows. And otherwise it's also a conditional expression which is another creation of, well, it exists in several languages now. It's a conditional expression. If it doesn't exist in those languages, that is, the function is undefined, then it returns "unknown." Literally the string "unknown."

Now, of course the string "unknown" never varies, right, from browser to browser or instance to instance or user to user. So when the OS is not Android, iOS, or Windows natively, the function returns, as I said, "unknown" in quotes. And thus that value is never unique, and that value is used to salt the hash. As we learned years ago on this podcast, salting hashes is crucial to the security of hashed password storage because the salt effectively customizes the hash per use. With the salt broken, Check Point was able to trivially brute force the user's six-digit PIN. Yes, on top of everything else, even if the system was working correctly, its entire security was controlled by a six-digit PIN.

Check Point wrote: "CPR (Check Point Research) roughly re-implemented the key derivation and keystore decryption in NodeJS and performed a brute force attack on the PIN code. This resulted in a performance of 95 passwords per second on a four-core Intel Core i7 CPU. Although this is not a very high speed, it is sufficient for the attack on a six-digit PIN code. In the worst-case scenario, checking 10^6 possible variants means the entire attack takes approximately 175 minutes." And that's worst case.

They said: "For our experiment, we created a new key in Surf and dumped the keystore from the browser's unencrypted local storage. In our case, the attack took 38 minutes. At the end, we got the derived key and decrypted the seed phrase that can be used to restore the keys on another device." In other words, this was never secure. And in this

case, I mean, first of all, as I said, the idea of doing a browser-based wallet is just nuts. It'd be like, I don't know, putting a wallet on a lemonade stand in the front yard and trusting that no one is going to come along and take it. I mean, it's just - it's insane. Browsers struggle with security, and you do not want your cryptocurrency private keys anywhere near a browser.

And again, had this - it would have been a bad idea to implement it on a browser in any event. But this is a classic instance of why it's a bad idea. Libraries were used that were not fully understood. They deployed this thing without ever verifying that the hash was never changing, and so the same hash was being used to always encrypt the user's data. And it just meant that the whole thing could be brute forcible. Check Point also noted that in the same way, back in the day, Leo, what were those tables called? Rainbow tables.

Leo: Rainbow tables, yeah.

Steve: Yes. And so basically you could create a rainbow table using some GPUs in the cloud to come up with the hashes for all 10^6 possibilities. That wouldn't take a lot of time. Then you could simply decrypt everybody's wallet who has one of these things that you're able to get a hold of. So just a bad idea.

Okay. Java once again. Java 15, 16, 17, and 18 received, the JDK, the developers kit, received "must updates" last week. Neil Madden, the somewhat excitable guy at ForgeRock, who discovered a new and quite severe problem with Java, considers it to warrant a CVSS of 10.0. I think we should reserve that for the software apocalypse, or perhaps when Skynet obtains self awareness. The rest of the industry gave his discovery a still very healthy CVSS of 7.5, and in no event should this one be ignored. Anybody doing Java development using security needs to update using from last week's critical update.

So here's what Neil wrote about his discovery. He said: "It turns out that recent releases of Java were vulnerable to a flaw in the implementation of their widely used ECDSA." That's the Elliptic Curve Digital Signature Algorithm. And by the way, that's now the default across, like, all state-of-the-art algorithms are using ECDSA. And we'll see some examples in a second. He said: "If you are running one of the vulnerable versions" - that is, Java 15, 16, 17, or 18 - "then an attacker can easily forge some types of SSL certificates and handshakes, allowing for interception and modification of communications; signed JWTs, JSON Web Tokens; SAML assertions or OIDC ID tokens; and even WebAuthn authentication messages, all using the digital equivalent of a blank piece of paper."

He said: "It's hard to overstate the severity of this bug. If you are using ECDSA signatures for any of these security mechanisms" - and as I said, ECDSA is now the default standard - "then an attacker can trivially and completely bypass them if your server is running any Java 15, 16, 17, or 18 version before the April 2022" - that's last week's - "Critical Patch Update (CPU)." He said: "For context, almost all WebAuthn/FIDO devices in the real world, including YubiKeys, use ECDSA signatures; and many OIDC providers use ECDSA-signed JWTs," JSON Web Tokens.

Leo: I use it for my SSH keys. But that wouldn't be impacted by this.

Steve: No, because I would not imagine...

Leo: I'm not using Java for my SSH.

Steve: Right, exactly. You're probably not - you're not connecting to something that is Java based.

Leo: No. No.

Steve: So he says: "If you've deployed 15, 16, 17 or 18 in production, then you should stop what you're doing and immediately update to install the fixes in the April 2022 Critical Patch Update." And finally he says: "Oracle has given this a CVSS score of 7.5, assigning no impact to Confidentiality or Availability." Which I agree is questionable. He said: "Initially we at ForgeRock graded this a perfect 10.0 due to the wide range of impacts on different functionality in an access management context. ForgeRock customers can read our advisory about this issue for further guidance." So in any event, for any listeners, if anybody is a Java developer, if you don't already have it, if you haven't received a notice, definitely update your Java JDK to the latest. And apparently 15 and 16 had gone out of support. So they're never going to be fixed, as I understand it. Hopefully everybody has by now moved to 17 and 18.

We have a bunch of interesting closing-the-loop feedback this week. 7337, that's of course LEET upside down, he said of SN-863 and use after free, he said: "Why does the deallocated memory not get zeroed? Why does malloc not also zero out the deallocated memory? Would that not solve the use after free issue?" It's a nice point, but here's the catch. It turns out it's not the memory, it's not the contents of the memory that's been freed that matters. It's that you have a pointer which used to point to some memory.

Modern automatic languages make it impossible or difficult or at least managed to get a pointer. You have to get something that refers to something else, you get that from the underlying language. Unlike C or obviously assembler, where a pointer could just be - you just make one. The next-generation automatic languages are dealing with pointers on your behalf for you. So it's the fact that you are able to retain something that you got from the language which was a pointer, and then what it points to got freed, means that you might get lucky, and that it will then in the future point to something that you shouldn't be able to point to. Anyway, it's the pointer that is the key, rather than the contents of what was released.

Awk tweeted: "Re: Feedback on 867." He said: "On MS Windows Auto Update Service." He says: "My thoughts are on the routers. I've heard Bruce Schneier articulate and give examples on how cheap IoT devices are designed and manufactured in publicly available talks and seminars." He said: "A team is assembled," as Bruce said, "a team is assembled and then immediately disassembled after the process, and there is no one left to actively package and push patches to these devices, unlike the teams at Apple, Microsoft, and Google." Which, anyway, I thought that was really interesting.

He said: "Coupled with this, I shudder at the recent incidence and demonstration from SolarWinds that your supply chain update servers and processes can be compromised and commandeered. Schneier comes to the conclusion that market forces cannot help here, as consumers want their devices produced fast and cheap, and thus only regulation would be the cure." Anyway, he asks, "I wonder what your thoughts are." And of course we've talked around this certainly a lot. My thoughts are that we're doing this all wrong, that it is wrong that it is necessary for security firms to reverse engineer proprietary software which tens of millions of users are actively using that nobody else had to sign off on, that nobody else had to look at, that the company said, "Trust us, it's secure." You know, my favorite example is voting machines. How is it that we're using proprietary

voting machines that were never vetted in any way? It's just, you know, it's broken. It leaves me to conclude we must still be in the early days of all this because this can't continue.

Andy in the U.K. said: "On Google Auth, email addresses can be checked just by sending an email to it."

Leo: Oh, good point.

Steve: Yes. Yes.

Leo: Never thought of that.

Steve: Yeah. He says: "If it bounces, the account doesn't exist. So it makes very little difference to Google if they reveal the address does exist during the Auth process. However, others using email as ID shouldn't reveal if it's for a valid account."

And however, I just chose Austin out of a hat, and that hat was brimming with, I'm not kidding, like maybe 200 public and private tweets who all said, uh, Steve, here's what's going on. Austin says: "Listening to Episode 867." That was last week. "During your conversation about authentication processes and email addresses being revealed during the process, Leo asked why an application like Gmail would ask for an email address first before moving on to the next step of the authentication process."

Leo: I see it more and more often, all the time now.

Steve: "This is because many apps use many identity providers and authentication workflows. Your email address will determine which authentication workflow..."

Leo: What to do next, yeah.

Steve: Yes, "...the application will walk down..."

Leo: Do you want to go to Okta? Do you want to do Duo? Do you want to do a password?

Steve: Precisely.

Leo: Yeah, that makes sense.

Steve: Are you using a YubiKey? Are you going to go to a password? Do you have a custom SAML identity provider and so forth. So thank you, everybody. I wanted to thank everybody as a group who said, uh, Steve? Apparently you don't do this, but everybody else is having to do this more and more.

Leo: Yeah.

Steve: And so makes absolute sense. Thank you, everyone.

Leo: Makes sense, yeah.

Steve: And then finally...

Leo: Go ahead. I was just going to offer a break, but I think you have another one.

Steve: We've got a couple more things. CMe, he said: "Hi, Mr. Gibson." Hi, CMe. He said: "I just thought..."

Leo: I CU.

Steve: I CU, "...you'd get a kick out of this." He said: "On your recommendation I put the McCollum Gibraltar trilogy in my wish list on Amazon. It's only sold as individual books. But wanting to save a few bucks, my son-in-law contacted the seller. That turned out to be the author himself" - yeah, Michael McCollum - "who agreed to refund the difference in postage. So I received a $7 check signed by McCollum, and three autographed books for my birthday."

And I just wanted to remind people he is one of my favorite authors. His site is scifi-az.com. He's a self-publisher. He does offer all of his books in eBook format. They are DRM-free using the "we trust you." And he's got two trilogies. There's the Gibraltar trilogy, and the Antares trilogy. Both are on my "must read" reading list. He is literally a rocket scientist. He designed one of the pumps that's in use on the International Space Station. He's an engineer. But he is a beautiful storyteller. And what I love about his stuff is they are really clever. I mean, they've got lots of aha moments, and they're surprising and really, really gratifying. So, and they're not expensive. So scifi-az.com.

Speaking of sci-fi, I wanted to note that I finished Book 4 of the Bobiverse, after doing the trilogy. And I'm back to Ryk Brown's Frontiers Saga.

Leo: You'll never run out of those. Holy cow.

Steve: No. Although I will run out of patience. That's the problem.

Leo: Yeah, yeah.

Steve: Because I should say - I should explain. It never happens, at least not recently, that I stay up reading after Lorrie falls asleep on my lap.

Leo: Do you put the book on her head?

Steve: No.

Leo: That would be mean.

Steve: I hold it off to the side.

Leo: You're a good partner, good.

Steve: I could not - this is last night. I could not stop reading this first book of the third 15-book story arc.

Leo: Doesn't your arm get tired? We need to get an arm to hold your Kindle.

Steve: And I just did want to mention, too, that I have hopes for next week. "Star Trek Discovery" was way too hyperactive for me. It just seemed like a big, like, VR videogame. I just didn't get into it. But we've got "Strange New Worlds" starting, where it is a prequel to the original first Kirk and Spock and McCoy series.

Leo: Yeah, the trailer looks good.

Steve: It really does. I'm just - I would like a Star Trek for adults. Maybe there's no market. But I'm one.

Leo: I would watch it, yes.

Steve: Oh, goodness, yes. So it would be - if we could return to Star Trek's original roots, which the "Next Generation" was also faithful to, of actually telling stories rather than only having an excuse for special effects. And I've not yet looked into the second season of "Picard" after the disappointing first season, although I know it's there. And it's just so sad to see Patrick looking so old, frankly. And I've not started the sixth season of "The Expanse," though it's also - I think it's probably done by now.

And finally I have a milestone. The incremental development release of SpinRite which I posted at 1:47 p.m. last Friday afternoon really surprised me. Tester after tester has been reporting that everything that's been completed so far is finally working perfectly, solidly, and better than ever for them. The reason this surprises me is that the code SpinRite now has is not relying upon any BIOS to interface it to mass storage adapter and drive hardware, yet it is finally working on every piece of hardware that everyone has of any age and vintage.

As we know, operating systems are able to achieve this by bundling a raft of hardware-specific manufacturer-supplied drivers which the OS loads on demand based upon the hardware it detects in the system when it's booted. But that's not a practical approach for SpinRite. What I was hoping we would be able to achieve was the creation of universal native drivers, one for IDE parallel adapters with PATA drives, and another for AHCI adapters with SATA drives, where they would simply work everywhere on all hardware from the 1980s through the latest chipsets. I'm not only surprised, but I'm also

greatly relieved since after many months of work we finally have 100% success, and every indication is that this elusive goal has finally been achieved.

I've been stuck here for a while, perfecting this foundation, because everything that comes next not just SpinRite 6.1, but 7.0 and all of SpinRite's future after that will be built upon it. And it's all based upon that new IO abstraction approach which means that new mass storage technologies, like native support for USB and NVMe and whatever comes after that can be easily added behind the abstraction. So once I catch my breath, I can finish the work on the rest of SpinRite, building upon this new foundation. And then we're going to have 6.1.

Leo: That is, wow, that's great.

Steve: It was a good week.

Leo: This is I think uniquely difficult because previous versions let BIOS do the matching, and now you've got to replace that BIOS call.

Steve: Yes.

Leo: That's tough.

Steve: It was a mixed blessing. On one hand we got ease of interfacing because the BIOS knew how to talk to its own hardware.

Leo: Right.

Steve: The problem is we also got a growing thickness of insulation through which SpinRite couldn't really see what was going on. And so for example, anyone who's ever seen Dynastat painfully slowly getting sector samples, it's like it just takes so long because you have to issue a reset through the BIOS any time you get an error. And it can take like 20 seconds. So that's all going to go away. 6.1 is just going to - it's going to do data recovery at a screaming rate that we've just never seen, and I've never seen before. No one's ever seen it before.

Leo: So you were using INT 13 pretty much for everything.

Steve: Yes.

Leo: Which, I mean, in the first edition made it easier, a lot easier to write it because you didn't have to test a variety of hardware or anything. Did the move away from BIOS make - or is INT 13 always supported regardless? I guess it has to be; right?

Steve: It's still there.

Leo: In software.

Steve: Yeah, exactly. It is what the boot sectors use in order to get themselves booted.

Leo: Right, right. It's got to be there, yeah.

Steve: But booting an OS these days is only a matter of reading a few K of code into RAM, and then it starts to bring the rest of itself into memory.

Leo: Right, it's all hard drive. So I guess, I would guess this would be a first step in making it Mac compatible, too, because of course Macs never had INT 13, never had BIOS.

Steve: This absolutely will run on a Mac.

Leo: Nice.

Steve: Yup.

Leo: Okay. Wow. You heard it here almost first. I'm sure the forums know. But that's the first I've heard. So that's great.

Steve: Yeah, it was a good Friday.

Leo: Get in there, get SpinRite 6 so you can get 6.1. It's almost here, folks. That's going to be a red-letter day. Let me know ahead of time so I can get a cake and some balloons and confetti and stuff. Got to plan it. Now let's talk about The Zero-Day Explosion.

Steve: And you know, that's what you want computers to do; right? That was always - we just don't, you know, computers have been a typewriter for so long.

Leo: Right.

Steve: Basically that's all they did. They weren't really extending our brain in a lot of ways.

Leo: Well, and it's not just horsepower. It's also memory, RAM.

Steve: Yeah. True.

Leo: And just software skill. And Grammarly's been around to kind of...

Steve: Well, and clearly, if these guys are in the Ukraine, they've got some English speakers because this is not something that you can...

Leo: Oh, yeah, yeah. Well, they're a distributed, yeah, they're all over the world. And in fact they come in many languages, too, by the way, I should mention.

Steve: Yeah, yeah.

Leo: Really good stuff.

Steve: Wow, very cool.

Leo: Yeah, it's impressive. And LISP, don't you love that? Well, maybe you don't...

Steve: Yeah.

Leo: I have to say I'm a little bit of a LISP fan boy.

Steve: John McCarthy would be proud.

Leo: Yeah, yeah, exactly.

Steve: Okay. So the most interesting and important class of software and system vulnerabilities are those that are discovered when a security researcher watches something that's not supposed to be possible, happen anyway. Like a specially formed packet hitting a firewall and being admitted through the firewall despite the clear firewall rule which blocks its entry. Or the password challenge that is ignored by the attacker who's logged on with administrative privileges anyway. Or the cryptocurrency mining malware which suddenly launches, springs to life, and begins operating on a system that was just reformatted and reinstalled from scratch. When a researcher watches something that cannot happen, happen anyway, they may have just witnessed and discovered evidence of the exploitation of a previously unknown zero-day vulnerability.

Zero-day vulnerabilities are a constant topic of this podcast. And just as this podcast appears to be in no danger of running out of security topics ever, recently posted sobering research from Mandiant and Google's Project Zero make it pretty clear that we also won't be running out of zero-days to discuss anytime soon. I have a chart which is quite - it makes you gulp, in the show notes. Ten years of zero-day tracking. What is going on? The show notes show this chart of Mandiant's sobering 10-year graph showing the number of zero-days discovered each year from 2012 through 2021 last year. The counts for each successive year are two, three, eight, 15, 21, 17, 16, 32, 30, and 80.

So 10 years ago, the entire year of 2012, we encountered just two zero-days, and the next year only three. Then the following three years rose to 8, 15 and 21. And the next

two years dropped back a bit to 17, and then 16 in 2018. 2019 doubled that to 32. 2020 dropped it a bit to 30. But then last year exploded from 30 in 2020 to 80 zero-days. So if you felt as though we have been talking a lot, and a lot more, about zero-day vulnerabilities recently, well, you would be correct.

Some interesting observations emerged from Mandiant's research. They wrote: "In 2021, Mandiant Threat Intelligence identified 80 zero-days exploited in the wild, which is more than double the previous record volume in 2019." That was 32. "State-sponsored groups continue to be the primary actors exploiting zero-day vulnerabilities, led by Chinese groups. The proportion of financially motivated actors, particularly ransomware groups, deploying zero-day exploits also grew significantly, and nearly one in three identified actors exploiting zero-days in 2021 was financially motivated." In other words, not espionage.

"Threat actors exploited zero-days in Microsoft, Apple, and Google products most frequently, likely reflecting the popularity of these three vendors. The vast increase in zero-day exploitation in 2021, as well as the diversification of actors using them, expands the risk portfolio for organizations in nearly every industry sector and geography, particularly those that rely on these popular systems.

"Mandiant analyzed more than 200 zero-day vulnerabilities that we identified as exploited in the wild from 2012 through 2021. Mandiant considers a zero-day to be a vulnerability that was exploited in the wild before a patch was made publicly available. We examined zero-day exploitation identified in Mandiant original research, breach investigation findings, and open sources, focusing on zero-days exploited by named groups. While we believe these sources are reliable as used in this analysis, we cannot confirm the findings of some sources. Due to the ongoing discovery of past incidents through digital forensic investigations, we expect that this research will remain dynamic and may be supplemented in the future." In other words, in the future they may learn more about what's happened in the past.

And they said: "We suggest that a number of factors contribute to growth in the quantity of zero-days exploited. For example, the continued move toward cloud hosting, mobile, and Internet of Things technologies increases the volume and complexity of systems and devices connected to the Internet. Put simply, more software leads to more software flaws. The expansion of the exploit broker marketplace also likely contributes to this growth, with more resources being shifted toward research and development of zero-days, both by private companies and researchers, as well as threat groups. Finally, enhanced defenses also likely allow defenders to detect more zero-day exploitation now than in previous years, and more organizations have tightened security protocols to reduce compromises through other vectors."

So I thought those points were really interesting. Of course we wonder whether our count of zero-days is recently higher because we're looking harder and more closely for them. This makes sense given that we've established quite clearly that with all software in general, the closer we look, the more problems we find. Some of those "more problems" will be exploitable zero-day vulnerabilities.

And the increasing level of specialization we've chronicled in recent years also leads to higher zero-day counts through the creation of this exploit broker marketplace. Now, now that there's a marketplace, those who wish to deploy such exploits don't need to spend their time hunting them down. And those who specialize in hunting for new ways in can spend all their time doing nothing else and then selling them into that marketplace.

And finally, so much of the lower hanging fruit has been found and pruned that zero-days are becoming the only remaining way to get in. Not exclusively, but to an increasing degree. This means that the pressure to discover new zero-days is greater than ever

before. And since, as we know, security is inherently porous, the harder you press on it, the more results will be obtained.

Mandiant said: "State-sponsored espionage groups continue to be the primary actors exploiting zero-day vulnerabilities, although the proportion of financially motivated actors deploying zero-day exploits is growing. From 2014 through 2018," they said, "we observed only a small proportion of financially motivated actors exploiting zero-day vulnerabilities. But by 2021, roughly one third of all identified actors exploiting zero-days were financially motivated. We also noted new threat clusters exploit zero-days, but we do not yet have sufficient information about some of these clusters to assess their motivation."

Okay. So just to be clear about that, the primary motivation behind the use of zero-days has historically been state-sponsored espionage. Things like breaking into military contractors' networks to steal plans for future weapons systems that are still on the drawing boards and things like that. But while such espionage remains dominant by far even now, by two times more, last year saw the rise in zero-days being the enabling factors of so-called "financially motivated" extortion with ransomware and sensitive data exfiltration and threats of exposure being the post-zero day intrusion consequences.

And Chinese-based cyberespionage groups remain the number one exploiter of these vulnerabilities. Mandiant said: "Mandiant identified the highest volume of zero-days exploited by suspected Chinese cyberespionage groups in 2021, and espionage actors from at least Russia and North Korea actively exploited zero-days last year. From 2012 through 2021, China exploited more zero-days than any other nation. However, we observed an increase in the number of nations likely exploiting zero-days, particularly over the last several years, and at least 10 separate countries likely exploited zero-days since 2012. From January to March of last year, 2021, Mandiant observed multiple Chinese espionage activity clusters exploiting four zero-day Exchange Server vulnerabilities collectively known as the ProxyLogon vulnerabilities. Microsoft described activity linked to this campaign as 'Hafnium.'"

So I'll just note that we appear to be focused upon the right things on this podcast. All of our listeners will recall the attention we gave to the constant stumbling Microsoft was making over their seemingly endless Exchange Server ProxyLogon vulnerabilities. They just couldn't seem to get it right. What we didn't and couldn't know at the time was that that string of Microsoft missteps was actually translating directly into a string of exploitation with Chinese espionage actors at the other end. They noted that: "While some of the threat clusters involved appeared to carefully select targets, other clusters compromised tens of thousands of servers" - that is, Exchange Servers - "in virtually every vertical and region."

And of course that makes sense; right? No one entity owns these vulnerabilities, and we have a heterogeneous environment of uncoordinated groups in China, Russia, and North Korea. Some are going to go for high-volume spray attacks, whereas others are going to go after specific targets.

And this little tidbit was somewhat worrisome. Mandiant said: "Chinese cyberespionage operations in 2020 and 2021" - so just the most recent two years - "suggest that Beijing is no longer deterred by formal government statements and indictments from victimized countries. In addition to the resurgence of previously dormant cyberespionage groups indicted by the U.S. Department of Justice, Chinese espionage groups have become increasingly brash."

The problem is that the world, I think, is becoming inured to the whole concept of cyberespionage, cybercrime, and cyberattacks. As the years go by and we keep talking about them, it's just human nature that they're going to become less and less frightening

and exceptional. They will simply be incorporated into our expectations. Where previously they were a big deal, it'll be just like, uh, okay, fine. You know, sort of like DDoS attacks are. It's like, oh, yeah, that happens.

As for Russia, Mandiant noted that: "In a sharp departure since 2016 and 2017," they wrote, "we did not identify any zero-days exploited by Russian GRU-sponsored APT28" - that's, you know, Fancy Bear - "until they likely exploited a zero-day in Microsoft Excel in late 2021. However, open source reporting indicated that other Russian state-sponsored actors exploited several zero-days in 2020 and 2021, including possibly targeting critical infrastructure networks with a zero-day in a Sophos firewall product." And as we know, through the past four years, Mandiant said that they had noted a significant increase in the number of zero-days leveraged by groups that are known or suspected to be customers of private companies that supply offensive cyber tools and services. And we know who those guys are.

They said: "We identified at least six zero-day vulnerabilities actively exploited in 2021, potentially by customers of malware vendors, including one reportedly exploited in tools developed by two separate vendors. In 2021 at least five zero-day vulnerabilities were reportedly exploited by an Israeli commercial vendor." Well, those two separate vendors they're referring to were the well-known Israeli NSO group and a second smaller and lesser well-known vendor of very similar exploit capabilities known as "QuaDream." Like the NSO Group, QuaDream is also Israeli and competes in the same market as the NSO Group, primarily selling to government clients.

Mandiant also noted that, unlike in the past, zero-day exploits were no longer appearing in underground exploit kits. Now, that's interesting. They explained: "Since 2015, we observed a sharp decline in zero-day vulnerabilities included in criminal exploit kits, likely due to several factors including the arrests of prominent exploit developers. However, as the criminal underground coalesced around ransomware operations, we observed an uptick in ransomware infections exploiting zero-day vulnerabilities since 2019. This trend may indicate that these sophisticated ransomware groups are beginning to recruit or purchase the requisite skills to exploit zero-days that may have been formerly developed for exploit kits." In other words, an exploit kit would have just been - it would have included zero-days because they exist. That would have essentially been tantamount to giving them away. Well, why give them away if there's now a powerful, active, profitable market for them?

They said: "Mandiant has documented significant growth in ransomware in terms of both quantity and impact. Substantial profits as well as the increasingly compartmentalized, outsourced, and professional ecosystem that supports ransomware may have provided operators with two viable pathways to zero-day exploit development and/or acquisition: financial resources and actor sophistication." In other words, ransomware operations increasingly have the money to purchase high-value, but also high-cost, zero-day exploits from underworld sources. And those underworld sources increasingly have zero-day exploits to offer for sale.

So where are all these zero-days being found? We have a chart in the show notes. Mandiant says: "We analyzed zero-days from 12 separate vendors in 2021, with vulnerabilities in Microsoft, Apple, and Google products comprising 75% of total zero-day vulnerabilities, likely as a result of the popularity of these products among enterprises and users across the globe." So Microsoft with all of their many products, Apple with their family of iOS devices, and Google with Chrome and the Android platform. Together, as this chart shows, those top three account for just a tad more than 75% of those 80 zero-days which occurred during 2021. Microsoft has the most, though they also have the most hardware sprawl, so I guess it's not surprising. Apple has the next most, with Google the fewest of the three. And given the nature of Chrome and Android, that's pretty impressive, really.

There are nine other major and notable sources of zero-days finishing out that Top 12. In order of decreasing zero-day counts, the remaining nine are: Accellion, SonicWall, Apache, Qualcomm, Trend Micro, Adobe, the Linux Kernel, Pulse Secure, and SolarWinds.

Mandiant noted that: "The threat from exploitation of these major providers remains significant" - meaning Microsoft/Apple/Google. That's where the zero-days are. That's the platforms that everyone's using, and that's where to really watch. They said: "In addition, we noted a growing variety in vendors being targeted, which can complicate patch prioritization and make it more difficult for organizations who can no longer focus on just one or two vendors as priorities." Which is really interesting, meaning there's just more to patch now than there was before.

They said: "From 2012 to 2017, Adobe was the second most exploited vendor, with nearly 20% of all zero-days exploiting" - wait for it - "Adobe Flash alone."

Leo: Of course.

Steve: Yeah. 2012. Remember those days? "We observed," they said, "a significant drop in Adobe exploitation since then, almost certainly fueled by Flash's end of life." Yeah, no kidding. How many times did we lament the continued existence of Flash when it was so obviously obsolete while also being such a global menace?

So what's the future outlook for the world of zero-days? Mandiant says: "We suggest that significant campaigns based on zero-day exploitation are increasingly accessible to a wider variety of state-sponsored and financially motivated actors, including as a result of the proliferation of vendors selling exploits and sophisticated ransomware operations potentially developing custom exploits." In other words, zero-days are big business, and that business is currently seeing what can only be described as explosive growth.

As for what enterprises can do about this, Mandiant says: "The marked increase in exploitation of zero-day vulnerabilities, particularly in 2021, expands the risk portfolio for organizations in nearly every industry sector and geography. While exploitation peaked in 2021, there are indications that the pace of exploitation of new zero-days slowed in the latter half of the year. However, zero-day exploitation is still occurring at an elevated rate compared to all previous years."

They said: "Many organizations continue to struggle to effectively prioritize patching to minimize exploitation risks." Again: "Many organizations continue to struggle to effectively prioritize patching to minimize exploitation risks." And we'll come back to that. And remember that survey we talked about recently where CIOs and IT professionals confessed to just how bad their organizations truly were about applying patches in a timely fashion.

To this, Mandiant added: "We believe it is important for organizations to build a defensive strategy that prioritizes the types of threats that are most likely to impact their environment and the threats that could cause the most damage, starting with the relatively few number of actively exploited vulnerabilities. When organizations have a clear picture of the spectrum of threat actors, malware families, campaigns, and tactics that are most relevant to their organization, they can make more nuanced prioritization decisions when those threats are linked to active exploitation of vulnerabilities."

And, you know, okay. That just seems so unrealistic to me. I mean, in a perfect world, sure. But we're talking about an organization dedicating someone to the full job of essentially continuously surveying the dynamic and constantly changing threat landscape and cross-checking it with all of the organization's potential vulnerabilities. What

organization is really going to do that? The truth is that everyone in IT is overworked, and there's an awful lot of hoping for the best going on. You know, hoping for the best was what that survey revealed; right? It was like, well, nothing happened today, and it's quitting time.

So there's no argument that, all other things being equal, focus less upon theoretical problems and more on vulnerabilities that are actively being exploited makes sense. Mandiant wrote: "A lower risk vulnerability that is actively being exploited in the wild against your organization or similar organizations likely has a greater potential impact to you" - okay, yeah, no kidding - "than a vulnerability with a higher rating that is not actively being exploited." Okay. So thus CISA.

They said: "A new CISA directive places a significant focus on those vulnerabilities that are reportedly actively exploited. We believe this will help increase the security posture and strengthen patch management procedures." Except as I said toward the beginning of the podcast, yeah, these things that are four years old, I don't know. Cross-site scripting in a package that no one's ever heard of and is taking up space and sort of diluting the other more important things?

Anyway, they finish: "While zero-day exploitation is expanding, malicious actors also continue to leverage known vulnerabilities, often soon after they've been disclosed. Therefore, security may be improved by continuing to incorporate lessons from past targeting and an understanding of the standard window between disclosure and exploitation." And of course we spend a lot of time here talking about that. They said: "Furthermore, even if an organization is unable to apply the mitigations before targeting occurs, it can still provide further insight into the urgency with which these systems need to be patched. Delays in patching only compound the risk that an organization supporting unpatched or unmitigated software will be affected." And again, yeah, obviously.

Having read all that and shared all that, and considering the impracticality, I think, of expending any great deal of time on prioritization, and also given that low-priority exploits are still exploitable, my own advice to any organization, especially in light of that survey we covered which confessed that patching was clearly not a priority, would be to first and foremost simply fix that. Period. Figure out how to arrange to keep the enterprise's software up to date. Yes, systems need to be taken offline, updated, and rebooted. Yes, it's inconvenient. And yes, customers and employees and even upper management in the C-suites will complain.

But today's and tomorrow's reality is that last year the number of zero-day vulnerabilities which were being used in the wild exploded from 30 the year before to 80. And those were only the worst of the crop. There were a great many more than just those 80 zero-days; right? I mean, non-zero-day vulnerabilities, many more vulnerabilities. Microsoft themselves patched 128 vulnerabilities just two weeks ago. It's only going to get worse. So to me, I mean, I get Mandiant's position. Yes, wouldn't it be nice if we, like, certainly you want to look at all the notices. And if you see something which is a glaring collision between something that's just been patched and software that you know your organization relies on, and it's something that's exposed to the Internet, yeah, you know, shut it down. Turn it off. Patch it.

But in general, I think all the evidence we see says that shortly after patches are released, they are reverse engineered, and attacks begin. So it is just a reality moving forward that there isn't an alternative to taking systems down routinely and updating them. It has to happen. And the survey that we talked about demonstrated that it hasn't been happening. And, you know, I don't talk about the endless stream of ransomware attacks. They're happening constantly, everywhere. I mean, there's just no point in filling the podcast with them. But they have not let up. And it doesn't look like they're going to.

Leo: Yeah. So, I mean, some of this is just reporting bias; right? Like we know more about it. We're more aware of it. Or do you really think there is an increase in zero-days?

Steve: I really do think things are getting more complicated. Look at - the perfect example is UEFI. Before we had it, you couldn't infect it. You know? Now we have it, and you can infect it.

Leo: Yeah.

Steve: So we are seeing increased complexity. We're also seeing an increasing use of this toolkit approach, right, where the so-called supply chain attacks. Well, if you don't have a supply chain, you can't attack it. If you're writing stuff in-house, you know how it works. If you're grabbing modules off the NPM repository and dropping them in, then you're making mistakes.

Leo: Yeah, yeah.

Steve: Because you didn't write it, and you don't know how it works.

Leo: That's true. I mean, our software is more complex than ever before, yeah.

Steve: Yes. It is doing more than ever before.

Leo: Right, right, right.

Steve: But that means it's going to be more complicated.

Leo: Yeah, yeah, yeah, that makes sense.

Steve: And there is more pressure. You know, remember how many years, Leo, did you and I sit here thinking, well, isn't that an interesting virus. It's just moving around from place to place, and it doesn't do anything.

Leo: Right.

Steve: Well, weren't those days quaint.

Leo: There's also nation-state actors, which is a whole new player in this field; you know?

Steve: Yes. That's right. Yes. The fact that Chinese espionage is like where more than two thirds of these things are being leveraged, that should be sobering. And also the idea that there's sort of a laissez-faire attitude. I mean, again, that's human nature, too. It's like, oh, well, my computer works. Good luck.

Leo: Yeah, mm-hmm. Well, you don't need luck if you listen to this show, and that's the truth. Steve Gibson is the man of the hour. Every Tuesday, 1:30 p.m. Pacific, 4:30 p.m. Eastern, 20:30 UTC, we gather together to talk about the state of security in this world of ours. And no surprise, it's getting more tenuous. It's really a show you have to listen to every single Tuesday. You can watch us do it live, if you want, at live.twit.tv. If you're watching live, chat live at irc.twit.tv; or in our Club TWiT Discord.

After the fact, of course, we've got other ways you can chat with us, including our TWiT forums at TWiT.community. Our Mastodon instance now proudly not owned by Elon Musk at TWiT.social. And of course Steve's got his own forums at GRC.com, including very active forums discussing SpinRite, the world's best mass storage maintenance and recovery utility. 6.1 is coming. So go over to GRC.com and get a copy of 6.0. You'll get a free copy of 6.1. And you can participate in these final stages of development.

While you're there, do get a copy of this show. Steve has two unique formats, 16Kb audio for the bandwidth-impaired. He's also got really useful transcripts written by a real human, Elaine Farris. So she writes everything down. And that means you can read along as you listen. You can search the transcripts to find a part of the show. I remember he said something about this. You can go right there. That is a very, very handy feature. And that's only at GRC.com.

We have copies of the show at our website, TWiT.tv/sn. There's a YouTube channel devoted to Security Now! with all the videos of all the shows that have video, anyway. And of course you can subscribe. If you want to get all the new shows the minute they're available, just subscribe in your favorite podcast player, and you'll get them pretty darn quick, audio or video, your choice. That's just Security Now!, search for it, or search for TWiT on your podcast player. You should find that easily since we have been around for, what is it, 17 years?

Steve: Coming up on 18.

Leo: Yeah, wow.

Steve: Yup.

Leo: Let's see. What else? Oh, if you would, leave us a five-star rating.