The Power of Privilege

Description: How do fake drives keep being sold by Amazon? If you don't already know it, is VBScript worth learning today? NTLM authentication is 30 years old; will it see 40? What startling flaw was just found in cURL, and what should you do about it? Vulnerabilities with a CVSS score of 10.0 are blessedly rare, but today the industry has another. And also, asked by our listeners, how should "lib" be pronounced? How is SpinRite's 6.1 pre-release run? Is Passkey export on the horizon? Doesn't a server's IP address make encrypting the client hello superfluous? Is there such a thing as encryption preemption? Are fraudulent higher-end drives possible? What's Privacy Badger and why did I just install it? And finally, within any enterprise, few things are more important than managing user and device access privileges. As highlighted by the NSA's and CISA's experiences, we're going to examine the need for taking privilege management more seriously than ever during this week's Security Now! Episode #945: "The Power of Privilege."

High quality (64 kbps) mp3 audio file URL: http://media.GRC.com/sn/SN-945.mp3
Quarter size (16 kbps) mp3 audio file URL: http://media.GRC.com/sn/sn-945-lq.mp3

SHOW TEASE: It's time for Security Now!. Steve Gibson is here. Coming up, the end of the line for VBScript. Or is it? Another massive flaw in cURL. A CVSS of 10 for Cisco. And the power of privilege and why it's not a good thing. It's all coming up next on Security Now!.

Leo Laporte: This is Security Now! with Steve Gibson, Episode 945, recorded Tuesday, October 24th, 2023: The Power of Privilege.

It's time for Security Now!, the show where we cover the latest security news and of course recommendations and explanations from this guy right here, Steven Gibson. Hello, Steve.

Steve Gibson: Yo, Leo. It's great to be with you.

Leo: Let's get ready to secure.

Steve: Here we go. Yes, it is the penultimate episode of October. I learned several years ago that that does not mean the last one, that means the second to last one because we're going to be on Halloween a week from today.

Leo: Oh. Are you going to wear a costume for me?

Steve: No, but we do have something to hang in the background.

Leo: Okay.

Steve: Lorrie said, "Oh, you've got to put this in the podcast."

Leo: I will be wearing a costume. I just want you to know.

Steve: I'm in mine.

Leo: Don't be too surprised. I might dress in a black T-shirt as Steve Gibson, maybe put on a moustache.

Steve: Hey, if you just put your wizard hat on, I think no one would find fault with that, Leo.

Leo: Okay, good.

Steve: So today we're going to address some questions, as we often do, certainly recently. We're going to start off with a bit of fun, even after our Picture of the Week, which is how is it that fake drives keep being sold by Amazon? Thanks to one of our listeners and users of ValiDrive, we actually have some first-hand knowledge of that experience. Also, if you don't already know it, is VBScript worth learning today?

Leo: Oh.

Steve: NTLM authentication is 30 years old. Will it see 40? What startling flaw was just found in cURL, and what should you do about it? Vulnerabilities with a CVSS score of 10.0 are blessedly rare, but today the industry has another. And also, asked by our listeners, how should "lib" be pronounced? Or is it "lib"? How is SpinRite 6.1's pre-release actually run? Is Passkey export on the horizon, as it may turn out? Doesn't a server's IP address make encrypting the ClientHello superfluous? Is there such a thing as encryption preemption? Are fraudulent higher end drives also possible? What's Privacy Badger, and why did I just install it? And finally, within any enterprise, few things are more important than managing user and device access privileges. As highlighted by the NSA's and CISA's experiences, we're going to end by examining the need for taking privilege management more seriously than ever during this week's Security Now! Episode 945, "The Power of Privilege."

Leo: The Power of Privilege.

Steve: Don't abuse it.

Leo: I can't wait. Good episode, as always, coming up. This is why you've got to tune in every Tuesday because, I mean, this show has the stuff you need to know.

Steve: We got some stuff here, that's right.

Leo: I need a Picture of the Week, Mr. G.

Steve: So looking at this, it occurs to me that sometimes people, like the wisdom of the crowd is what ends up prevailing. We have a picture where there was, in the foreground, at the bottom of the picture, we see the end of some pavers, like a paved walkway or something. And there's a big - and then the rest of the picture is a big grassy field. It's well-kept, lawn-mowed lawn. Now, clearly the people running this facility didn't intend for the actual users of the facility to wear a dirt path through the lawn to some remote corner where presumably there's a way out. The whole point being that this demonstrates that people want to go from Point A to Point B, even though that was not the original plan of those who were putting this together. Or if it was, they didn't appreciate that over time there would be no grass growing along this path that everyone was walking.

So unfortunately, the proper course of action would have been to simply pave an explicit path where the people want to go, rather than putting up this lame, like, it looks like maybe three feet tall by six feet wide barrier at the edge of these pavers, like as if it's going to stop people from going around it because they clearly want to use this path to get to wherever the path goes. So I gave this picture the caption "Not a well-thought-out plan" because I don't think this is going to turn out the way they're hoping.

Leo: You know what you're going to see is a path around the fence connecting up to the other path so people can continue to use it.

Steve: Yes. And already on the far side of this you can sort of see some diversion from the path where it looks like maybe people have already started to go around.

Leo: Yeah. I love it.

Steve: So over time this problem is going to get worse. You know, there's going to be like multiple ways to merge with the original path after you go around this ridiculous little, like, nothing.

Leo: It is possible that the people who set this up were geniuses and wanted a three-way path structure going on. Maybe they...

Steve: Ah. That is - that would - very impressive, Leo.

Leo: All right. All right, Steve. Yeah, yeah, yeah.

Steve: Anyway, another of our observations of the nature of human nature.

Leo: Human nature. Gotta love it.

Steve: Love it or leave it. So speaking of human nature, as many of our listeners know, the homepage for my ValiDrive freeware shows a photo of the 12 one and two terabyte drives which I purchased from Amazon last month. That page also includes links to the Amazon purchase listing for each of those 12 drives. And with some surprise, many people have wondered how this fraud can exist on Amazon. So I wanted to share the experience of one of ValiDrive's pre-release testers who took the trouble to purchase a drive, then to post her review of that just purchased drive. Her name's Leila, and she wrote, and this was posted over in GRC's newsgroup, so I just wanted to share it with our listeners here.

She wrote: "I posted the following review of a cheap 512GB USB drive, together with a screenshot of the ValiDrive results." So the review she wrote read: "This USB stick is a fake. Although labeled and formatted to appear to have 512GB of storage, it in fact contains only 53GB. Anything written beyond 53GB appears to write without error, but cannot be read back. The attached picture shows the result of testing with ValiDrive, a recently published freeware program for testing USB-attached storage. If you buy a high-capacity USB stick or microSD card for peanuts, don't be surprised if it's a fake. Check out the prices charged by reputable vendors such as SanDisk and Kingston for reference. In any event, it makes sense to test all such cards and drives before putting them in your camera, dash cam, et cetera. Even those which appear to be from reputable manufacturers are sometimes forgeries."

Okay. So that's what she posted on Amazon. Not long after, she received Amazon's reply. They wrote back to her: "Thank you for submitting a review of doykob" - that's not a name brand, D-O-Y-K-O-B, the lesser known doykob - "Memory Stick 512GB USB Stick Fast Speed USB 3.0 Drive Mini USB Metallic Flash Drive Portable USB Memory Sticks for Data Storage and Transfer, with Keychain." So that's the name of the product. Then they continued: "We're sorry you did not have a positive experience. We investigated your concerns about product authenticity, and the information we have indicates that the product you received was authentic. As a result, we removed the review you submitted. This ensures that customer reviews remain as accurate as possible for the benefit of future customers."

So that answers that question. Unwitting users who are taken in by this fraud post, you know, "Holy crap! I just bought this 2TB thumb drive for $20! I plugged it into my computer to verify its size and it looks great! This is a super bargain for 2TB so I'm definitely giving it five stars!" Of course, you know, they haven't tried storing much data on it yet. They just got it, plugged it in, looks fine, off they go. So those reviews are immediately posted on Amazon. Whereas an actual analysis of the drive, oh, you know, we're sorry you didn't have a good experience, but what you said was not the case, so we're not going to let anybody see it. We're just going to with all of the happy customers.

So anyway, that's the way this happens. And the other people have said, well, why not complain about a specific drive? Well, first of all, who are you going to complain to? Amazon won't list it because they just say, no, look at all the other people who are happy. There's something wrong with your computer. Okay. Anyway.

Microsoft's VBScript, which, you know, was the proprietary scripting language based upon Visual Basic, is officially headed for the dustbin of history, and not a moment too soon.

Leo: Aw.

Steve: I know, Leo. I know. Believe it or not, VBScript has been with us since 1998, so now for 25 years. It was originally Microsoft's attempt to combat JavaScript. You know, they didn't want somebody else's scripting language to prevail. So it's like, wait a minute, we've got one, too. And it's like, okay. And, you know, it kind of - it lived alongside JavaScript and the early .NET stuff and so forth for a while.

But since PowerShell is far more comprehensive than VBScript, and since Microsoft had essentially already abandoned VBScript - its last significant release was VBScript v5.8, 13 years ago in 2010 - so the only ones who are likely to mourn its passing are malware authors, among whom it has remained a popular tool. Even though, even there, not as popular as it once was.

Now, that said, VBScript still has a large fan base among system administrators who have established a code base of their own, you know, it's their stuff that they've written, is working, and it's been proven. The good news for those who do still need it is that it's not being, at least immediately, completely removed from Windows. Instead what's happening is that it's being moved from the "you can assume it's always there by default" into the so-called FoD, that's Microsoft's term, "Feature on Demand," also known as optional features.

Leo: FoD.

Steve: The FoD, yes. We're FoDing it.

Leo: Yeah.

Steve: Where those - so, you know, in those comparatively few environments that do still depend upon it, you'll be able to enable it and install it on demand. But it won't just always be there ready to go. They have said that they definitely plan to remove it entirely at some point. So anyone who does still today depend upon VBScript would be advised to migrate their code over to PowerShell. That's the future. VBScript has no future.

And as for malware, its removal, VBScript's removal is expected to have some impact on malware in the sense that another piece of lowest hanging fruit, which has been VBScript, will at long last finally have been pruned from Windows. But it's not as if malware will be hugely inconvenienced. Just as when Microsoft dropped support for macros in Office applications, VBScript's new FoD status will trigger a more concerted move to PowerShell-based malware exploitation. The larger gangs have already switched over to PowerShell because there's a lot more you can do with it, and unfortunately a lot more mischief that you can get up to. But the older groups who are more into the copy-and-paste mentality, they've just continued using VBScript because it was there. It's like, why not? So those guys will need to up their game once VBScript stops being available on all Windows unless it's installed by demand.

So anyway, anyone who is listening to this podcast would be well advised to plan today for the eventual disappearance of VBScript. And knowing Microsoft, it will likely survive over in the Optional Features category for quite some time yet. I'm not sure, you know, why they would ever remove it unless at some point Windows evolved to make it incompatible, in which case then they'd probably say, oh, well, we had to get rid of it

because it no longer ran on Windows, you know, 14. But still, I wouldn't be counting on it in the long term.

Speaking of things being discontinued, Microsoft has also announced their intention to eventually disable support for the ancient and never really secure NT, as in New Technology, LAN Man, you know, LAN Manager Authentication Protocol, which will occur, that is, the eventual disablement in a future version of Windows 11. And of course labeling anything NT, standing for New Technology, after 30 years in Internet time is really a stretch anyway.

So NTLM Authentication Protocol was first introduced in '93, so yup, 30 years ago. And that was with the release of Windows NT 3.1. It did serve as the primary user authentication protocol for Windows Networks until Windows 2000, which first introduced the Kerberos authentication system originally developed at MIT. Ever since then, NTLM has still been there in Windows as a backup authentication protocol for legacy purposes, and in actually the still many situations where Kerberos could not be used.

The problem, as we well know, with keeping legacy protocols around is that the "weakest link" principle applies, especially when it's coupled with the "Well, let's just leave it enabled in case someone might need it" philosophy of network administration. The problem is, of course, the bad guys might be the ones who need it. And if it's enabled, they're happy to take advantage of it.

But superior as Kerberos is, even today it's not a slam-dunk drop-in replacement for NTLM. Kerberos has known problems where NTLM works, in offline and local use or in segmented network topologies where there's no direct line of sight to a Domain Controller because Kerberos is a three-party authentication system, not a two-party system. So you need access to that third party. Microsoft has indicated that it's well aware of these issues - and let's hope it is - and that it's going to fix, it's going to be working on Kerberos's shortcomings before completely removing NTLM. But they have signaled that NTLM is eventually going to be retired.

Microsoft says it's working on two new Kerberos features, namely IAKerb and local KDC that will allow Kerberos to work in more diverse network topologies and in offline and local scenarios, as I said, where there's no DC, no Domain Controller around. And the good news is all of this will only affect Windows 11, not Windows 10, because it's probably still going to take a while to get there. And believe it or not, Windows 10 now has fewer than two years of support left.

Windows 10's support is slated to end on October 14th. We just last week passed the two-year cutoff, October 14th of 2025. So that's 103 weeks from now. And that will be coinciding with Security Now! podcast 1046.

Leo: Which will exist, thanks to you.

Steve: That's exactly right, Leo. It's a good thing that we're going to keep on going past 999 since I would not want to miss out on the opportunity to complain a lot about the premature ending of Windows 10 support. Which by then I'll just be starting to use.

So in any event - actually, I'm already using Windows 10. I'm fine with it. To help enterprises to wind down their dependence upon NTLM, and here's the takeaway for our listeners who are in the enterprise profile, Microsoft will be adding to Windows 11 new tools and logging capabilities to help monitor NTLM usage along with providing finer-grained group policies to disable and block NTLM so that enterprises can get ready for its eventual disappearance. And that also will be providing metrics back to Microsoft, as they

see that NTLM usage is diminishing. They've said they fully intend to announce a hard date when the protocol will be disabled in Windows 11 and then finally pull the plug.

Quoting Microsoft, they said: "Reducing the use of NTLM will ultimately culminate in it being disabled in Windows 11. We're taking a data-driven approach and monitoring reductions in NTLM usage to determine when it will be safe to disable." So I suppose the other strategy would be keep using the crap out of it, and Microsoft will go, whoa, I guess it's not safe to disable it, and maybe everyone will keep having access to it. The problem is NTLM has been a real problem. I mean, it has been a source of many conversations during the 18-plus years of this podcast when it has had serious vulnerabilities. So it absolutely would be good to go to an updated modern protocol. I mean, legacy protocols are not those that you want to keep around, especially when they're in crucial positions like network authentication.

Vulnerabilities in the massively widely used cURL command and library are always a concern since the library is so often included inside other projects because the features and services that cURL offer are incredibly handy. So it is significant when Daniel Stenberg, who is the cURL project's lead developer, calls one of two just-discovered vulnerabilities "probably the worst cURL security flaw in a long time."

Leo: And they say it a lot, by the way.

Steve: Oh, boy. Yes, it is. The two flaws, which were found by JFrog, impact libcurl versions 7.69.0, which is where they first appeared, through 8.3. Daniel wrote: "We're cutting the release cycle short and will release cURL 8.4 on October 11th, including fixes for a severity HIGH CVE and one severity LOW. The one rated HIGH is probably the worst cURL security flaw in a long time. The new version and details about the two CVEs will be published around 06:00 UTC on the release day." So there was CVE-2023-38545, that was the severity HIGH which affects both libcurl and the cURL tool, which is built from the lib. And also 38546, which has LOW severity and affects libcurl only, not also the tool.

He said: "There is no API nor ABI change in the upcoming cURL release." He said: "I cannot disclose any information about which version range is affected, as that would help identify the problem area with a very high accuracy, so I cannot do that ahead of time." He said: "The 'last several years' of cURL versions are as specific as I can get."

In other words, the last several years of cURL have had these problems, one which he characterizes as the worst cURL security flaw in a long time, meaning that, and this is the significant part, anything built with those libraries. Anything that incorporated any cURL within the last few years is also potentially at risk, and thus the severity of this overall. He said: "We've notified the distros mailing list, allowing the member distributions to prepare patches. Now you know. Plan accordingly."

So the worst of the two problems sounds quite bad and is easy to exploit on its face, but fortunately its required preconditions will prevent the world as we know it from ending. The cURL maintainers wrote in their follow-up advisory, they said: "The flaw is a heap-based buffer overflow in the SOCKS5 proxy handshake. When cURL is asked to pass along the hostname to the SOCKS5 proxy to allow that to resolve the address instead of it getting done by cURL itself, the maximum length the hostname can be is 255 bytes. If the hostname is detected to be longer than 255 bytes, cURL switches to local name resolution and instead passes the resolved address to the proxy. Due to this bug, the local variable that means 'let the host resolve the name' could get the wrong value during a slow SOCKS5 handshake and, contrary to the intention, copy the too-long hostname to the target buffer instead of copying just the resolved address there."

Okay. So we have a timing-based race condition that could lead to remote code execution via a buffer overrun. The cURL devs said that the overflow could be triggered by a malicious HTTPS server performing a redirect to a specially crafted URL.

The JFrog guys who found the problem said: "Seeing that cURL is a ubiquitous project, it can be assumed with high confidence that this vulnerability will get exploited in the wild for remote code execution, with more sophisticated exploits being developed. However, the set of pre-conditions needed for a machine to be vulnerable is more restrictive than initially believed."

So this feels exactly like another one of those vulnerabilities that, while it doesn't explode the Internet, will be latent. It will be entered into those massive exploit databases I've theorized about in the past. Something like that must be maintained by our NSA and, unfortunately, by all other sufficiently sophisticated malign actors on the global stage. There's likely a huge dependency tree that knows which version ranges of every utility and library that uses cURL, was built using any of those vulnerable cURL libraries during the years that cURL was now known to have been vulnerable.

So whether it's next year or 10 years from now, when we or some hostile actor, some hostile foreign power wishes to penetrate a network, that database might serve to remind such an actor, you know, like for example, "Hey, you know, the set of circumstances you're facing just happen to perfectly fit the use of a long ago patched, but still present in the target system, flaw that we could take advantage of today."

You know, as I've often observed, given the Swiss cheese that is our industry's legacy of interdependent flawed libraries which continue to persist because they're embedded in other things that are still being used, and in this environment of slow-moving updates there's just no way that such a capability does not exist. And it's a little creepy because that's a lot of information. So you can just imagine that would be a perfect place to apply AI. Some AI has ingested all of the past history of vulnerabilities and interdependencies so that you aim it at a given scenario where you want to penetrate, and it looks and finds a way in based on the current versions of the things that are detectable and knowing everything about still vulnerable interactions among them. So we may be entering a whole next generation of ways of penetrating networks.

Sometimes you don't have to work very hard, Leo, to penetrate a network.

Leo: Sometimes the network penetrates you.

Steve: Sometimes it just says, "C'mon, baby, what are you waiting for?"

Cisco has recently been dealing with a problem that is anything but latent. As we've often observed, really, really bad vulnerabilities, I mean, really, really bad ones are typically awarded a CVSS score of 9.8. And that's generally as high as it goes. And even that, even a 9.8 is quite difficult to earn. And I respect the reticence to go any higher, since we would like to reserve any assignment of 9.9 or 10.0 for those really just too horrible to believe problems.

Leo: The Nadia Comaneci of exploits.

Steve: Indeed. So this means that Cisco's recent award of a CVSS of 10.0 really does mean something. What does it mean? It means that almost overnight, more than 42,000

of Cisco's high-end publicly-exposed Internet routers and switches were compromised and taken over. Like I said, that's what you want to reserve those 10.0s for.

The first known instances of attacks against Cisco's IOS - and you know that's not Apple iOS. For Cisco, IOS is Internet Operating System. So Cisco's IOS XE-based routers and switches, which appear to have been, that is, the first instances of attacks appear to have been initial proof-of-concept probing incursions, occurred at the end of last month on September 28th. Then more than three weeks passed before Cisco finally released the first fixes last Monday on October 16th, so eight days ago.

During those intervening three weeks, more than 42,000 - 42,000, not like a hundred critical on-the-edge Internet-connected switches and routers, 42,000 - of these IOS XE-based devices are known to be compromised. And again, not were scanned by Shodan and found to be vulnerable. No. Were known to be compromised because the malware was detected in them. And as I said, compromise means full and effectively devastating takeover of the device. We know that it was 42,000 devices because scanners were quickly created by security firms who wanted to track incursions. And in response to the visibility of their initial implants, the perpetrators of these attacks then updated their malware to make it less visible.

To all of this, the tech press responded as we'd expect. Just yesterday, SC Media lead with "What the Cisco IOS XE zero-day reveals about potential risk management blind spots." Help Net Security wrote: "Disappearing implants, followed by first fixes for exploited Cisco IOS XE zero-day." CRN wrote: "Cisco Releases First in Series of Patches for IOS XE Vulnerabilities." A day earlier on Sunday, BleepingComputer's founder, Lawrence Abrams, wrote: "Hackers update Cisco IOS XE backdoor to hide infected devices." Dark Reading wrote: "Cyberattackers Alter Implant on 30,000" - and they meant 42,000, but okay - "Compromised Cisco IOS XE Devices." And The Register did not disappoint with their headline: "Cisco fixes critical IOS XE bug, but malware crew way ahead of them. Initial fall in infected devices indicates evolution, not extinction, of attacking code."

So in today's show notes I have a chart which shows the initial infection rise, which looks like it took about two days from the 16th of October to the 18th of October, two days. And we hit 42,000 infected devices. Looks like the majority were in Asia. They just, they barely outflanked North America, which was at number two, then came Europe, then Africa, then Oceania, and then others. So it peaked two days after, on October 18th. And then just around the 20th, we see it dropping, like right back down to zero.

We already know that there's no way 42,000 devices randomly scattered out on the Internet got themselves patched in like a day. That was the malware deciding, whoops, we made ourselves too easy to spot. We're going to go undercover. In other words, what we now have is a very real first-class CVSS of 10.0 mess. As I mentioned, the news of this broke publicly for the first time on the 18th. And that chart in the show notes shows that this corresponded with the peak of the detectable infections.

The problem that Cisco revealed was a zero-day, now known as CVE-2023-20198, which exists in the web - I can't even believe I'm saying this - the web admin panel of its IOS XE operating system. I can't count the number of times I've said on this podcast that all of our experience teaches us that it is absolutely never safe to expose any web-based administration to the public Internet. As an industry collectively, we've proven that we simply do not yet know how to do that safely. We keep thinking we do, but all of our experience says no, you don't.

If a network's management environment absolutely requires remote management access to a publicly addressable router or switch, then use a VPN to connect to the private network behind the device, then access its web interface from the inside. Never expose

any management-level web interface to the public Internet. There's just no reason to. "Ease of access" obviously also applies to the bad guys. You know, you're making their access easy. That's not what you want.

And really, in this instance, Cisco is at fault here in the year 2023. Not because they made a mistake. As we know, that can happen to anyone. But because it's even possible in this day and age to configure their devices to expose a web admin interface to any publicly routable network. It should not be possible. This is all enterprise-grade high-end equipment. Anyone using it should have VPN technology falling out of their pockets.

So only exposing web admin to private networks would never impose a burden on their administration. And if Cisco had implemented such a policy, 42,000-plus recently compromised and now much more difficult to find routers and switches would not be compromised, and Cisco could then safely make all the mistakes they might want with their web admin authentication without it ever creating what essentially is a global catastrophe.

And also note that this does not only apply to the enterprise, that is, this as strongly worded as I could possibly make it advice, does not only apply to the enterprise or to enterprise-class devices. It applies right now, today, to every one of us. Remember it was an exactly analogous flaw in a Plex media server web admin interface, which was located in the home of a LastPass developer, that was responsible for all of us having our LastPass vaults stolen and now being targeted for selective decryption.

Unlike the Cisco case, it's probably asking too much for Plex to disallow Internet-facing administration. But that individual, that LastPass developer who was deep into the security space as a developer at LastPass, should have done so him or herself. And everyone listening to this podcast should consider that, too. VPNs and overlay networks are now so freely available that it's only a matter of taking the time to set them up. Okay. Enough preaching and ranting.

In today's case with Cisco, the zero-day allowed threat actors to create an admin account with the highest level of privilege - in Cisco land it's Level 15 - on devices that had their WebUI panel exposed to the Internet. After some additional investigation, the presence of a second zero-day was also discovered being used by the attackers to inject commands into the IOS XE filesystem that would execute with root privilege. Well, that's never good. Cisco revealed that the exploit chain was being used to inject a Lua-based backdoor on devices across the world.

Cisco initially believed that the second zero-day was the exploitation of a previously fixed bug that was from 2021, CVE-2021-1435. But in an update late last Friday, Cisco said the attackers had discovered and were using a new zero-day, now assigned CVE-2023-20273. But believe it or not, both Cisco and CISA confirmed that the two-year-old patched flaw was also being exploited in the wild by a different threat actor in a different campaign. In other words, there are still machines out there where flaws are being exploited, a zero-day flaw from two years ago, further demonstrating that those 43,000 known to be penetrated routers and switches didn't just get miraculously patched in a day. We've still got them out there that haven't been patched for the last two years.

So like I said, there's never a good reason to expose all of this management, web interface management UI to the Internet. Just turn it off. Don't do it. Manage from behind and use an overlay network or a little VPN server, trivial to set up these days, if you need remote access.

The other thing that was interesting was that the Lua backdoor that was originally placed on all 42,000-plus hacked Cisco IOS XE devices since late September started disappearing over this last weekend. Estimates from both Censys and Shadowserver,

who were scanning the Internet doing all of this, put the number of hacked devices at the number I've been quoting, 42,000-plus IOS XE routers and switches. And over the course of a couple days, as we know, that number of exposed known compromised devices dropped down to between 500 and a thousand.

So who knows why it didn't go to zero. There are apparently more than one actor at work here. And consequently it's believed that this change was made by the threat actor itself. The Lua backdoor was not originally a particularly strong method of persistence since it could be removed with a device reboot. So it was only RAM resident at that point. And its public visibility was attracting too much attention to the attackers' operations.

Several security researchers pointed out that the Lua backdoor would likely have been replaced by something much more insidious, very similar to the way Chinese hackers evaded Barracuda's patches earlier this year to entrench themselves so deeply into the compromised appliances that Barracuda wound up telling their own customers that they probably should just replace the gear to be safe. There was no other way to really know for sure.

Leo: Wow. Wow. Geez. Wow.

Steve: Yeah, it's really, really bad. So at this point we don't know that anything like that has happened. But when vulnerabilities have been discovered which we know allows for the injection of commands into the device's file system with root privileges, given sufficient time - and these attackers had three weeks - it's clear that anything could have been done.

So today, the Internet might well currently have something on the order of 42,000 compromised Cisco routers and switches where their compromises have gone stealth after some threat actor somewhere, we don't know who, finished burrowing in. And at this point, you know, we'll never know exactly which routers and switches are affected. As we know, once a device or a network has been compromised, can we ever fully trust it again? Today we have patches for both the known zero-days that have made this possible. But applying the patches of course is only the first step. Even if some of the horses have left the barn, it's still worth closing the door. But you may still have some loose horses. Any enterprise whose Cisco edge gear may have been compromised would be well served to, unfortunately, perform a full and deep security audit into what may have happened on that device over the last month.

At this point, no one in the infosec community has made even a tentative attribution. No one is suggesting who the bad guys are. There's no malicious actor who would not be interested in penetrating Cisco's IOS XE gear, from hacktivists to advanced persistent threat operators, from initial access brokers, you know, IABs, to botnet operators. Could be anybody.

And we do need to give Cisco a little bit of credit where it's due. It was they who discovered this new pair of zero-days during their investigation of a customer's support ticket. Another vendor might not have bothered to engage a competent security team to really dig into the trouble's root cause. Cisco did. What they found was certainly troubling. And I would argue it should not have happened in the first place because you just should not be able to put a management, a web management UI out on the public interface.

Anyway, the takeaway for each of us individually should be that our own networks should never expose anything like that publicly. You should never have your own router's web management interface enabled on the Internet, regardless of how well chosen your

username and password may be. And the same goes for Synology, Plex, and everything else. As I said, just take the time to set up a simple overlay network or your own VPN server to allow you to access those devices from the inside.

And I'll just note that this made me think of this. One of the things I have long wanted to do was to expand GRC's ShieldsUP! service-port-only scan to check all 65,536 ports for its users. Back when I first wrote all of that code, I was running GRC's server from here at home over a pair of 1.54 megabit T1 connections. I was unable to offer to scan 64 times more ports in a reasonable amount of time. Today I could do that easily from Level 3. So, you know, on the back of my list, once I get to a place where I can turn my attention back to ShieldsUP! - which won't be until after SpinRite 7 exists and has been released - I think it would be fun to boost ShieldsUP! so that it's able to scan all of an individual's ports.

Leo: Oh, good idea.

Steve: Because, you know, who knows what it would show you that you might not even be aware of has opened incoming traffic through your router. We need to find out.

So our listener Stephen Lee, he said: "Love your show. I was wondering: If you pronounce 'lib' for library as rhyming with 'vibe,' do you pronounce 'var' for variable as rhyming with 'air'?"

Leo: I say "lib" and "var." I mean, that's just me.

Steve: Yeah. And I'm sure he's asking because everyone's heard me say "libe."

Leo: You say "libe," yeah.

Steve: I do say "libe" as in library. But I do say "var" for variable, as opposed to "vair." But, you know...

Leo: I don't think you can tie the pronunciation of the full word to the pronunciation of the shortened version.

Steve: I agree. Yeah.

Leo: So what's "lib" to you is "libe" to me. And "var," you know, I should say "vair." If I say "libe," I should say "vair," but I don't. I say "var."

Steve: Yeah, yeah.

Leo: So that's a fork. How do you say that?

Steve: Tom Frillman, he said: "Hi, Steve. I got the SR 6.1 EXE, but Win10 won't run it. How may I use it? Thanks for all you do. By the way, ValiDrive is great. I must be lucky. All 25 of my drives are fine except for one that won't even format. Maybe SpinRite will help."

Okay. So a bunch of people were confused by, and I apologize for this, for my mentioning that the pre-release was available because what's available is very different from what the full release will be. The SpinRite 6 that everyone is used to is a hybrid Windows and DOS EXE. It's kind of a cool trick that I came up with where that one EXE is both a signed Windows app and it's a DOS app. Because what Windows did when Windows first began to happen was, you know, there was still a lot of DOS around. So people would be running a Windows app in DOS. And so Microsoft thought, okay, well, we have to have some way to fix that.

So what they did was they defined the EXE so that there's a - it's actually a DOS executable on the front, a little stub. And all this, the little stub which is a DOS executable, says this program must be run in Windows. And then when you run the same executable in Windows, Windows is smart enough to know, oh, well, we don't - we just ignore this little stub, and we jump over it to go to the Windows app.

So what I did was I take the SpinRite DOS app and stick it on the front as the stub so that, when you run the SpinRite Windows app in DOS, instead of saying, whoops, this is a Windows app that you need Windows for, it runs SpinRite. Like in DOS. And when you run the same thing in Windows, you get the setup utility which allows you to create the USB drive, or format a floppy and install SpinRite on a floppy, or burn an ISO to CD-ROM and so forth. So anyway, that's all sort of hidden for most users. So unfortunately I didn't make all of that clear last week. The pre-release that I invited our listeners to download is a DOS app, which means that you need to stick it on some DOS boot drive.

Anyway, I spent the weekend updating GRC's forums, and I created an FAQ entry that has very clear instructions for how to do this. So if anybody else was confused, first of all, the people in the forums would be glad to help you, you know, forums.grc.com. And you can also go there and find much cleaner instructions. And I actually have them also in the show notes. I just sort of skipped over them by giving a more longer winded explanation. But that's the back story for why it wasn't as easy as just running the thing you downloaded from Windows. Windows would say, what? You know, like this is a DOS app. That's not what you're here for. And there's, you know, there's no Windows app behind it as there is in actual SpinRite.

Doug Smith, a listener, said: "Regarding exporting and importing Passkeys, I saw this from a 1Password employee on their forums: 'At this point in time'" - this is the 1Password employee speaking. "'At this point in time, you cannot import or export Passkeys from any managers like iCloud Keychain or 1Password. The good news is that we're working closely with platform vendors and other password managers through the FIDO Alliance to create a secure way to import and export Passkeys. We believe it's your choice where to store and use your Passkeys. Hopefully we'll have more to share soon.'"

So that is wonderful news. That's exactly what we need. The user's experience in my opinion is paramount. And what users would do would be to provide their own strong password, under which they would be exporting one or more Passkeys. That would be, you know, that strong password would serve to keep their Passkey safe while it's outside of whatever environment, password manager, whatever. Then they'd need to again provide it when they wish to import that into something else.

So that's the way I designed things with SQRL, and the technology to do this safely and unspoofably and verifiably is widely available. That's just not difficult. It's just a matter of establishing it as a standard that everybody can use. So it is very good news that the

FIDO Alliance recognizes that this is something that has been missing and that will go a long way toward making Passkeys a lot more, you know, more transparent, I think, and more useful for most of our users.

Leo: This is something we talked about in the original piece you did on Passkeys, which is that they had never put anything in the spec about portability.

Steve: Right.

Leo: And I think the reason they didn't is because Passkeys relies on a secure enclave like the hardware enclave in your iPhone or your Google phone to store those Passkeys. And I think they're worried that it's inherently insecure to let them out of the enclave. So they've got some work to do. I'm not convinced the FIDO Alliance will agree, will go along with it. This is just...

Steve: It'll be interesting to see if that's going to happen.

Leo: The fact that they left it out tells you surely they knew this was an issue. But they explicitly, well, I guess it's implicitly left it out; right?

Steve: Right. Well, and we've also suggested maybe that it was just an attempt by the initial providers of Passkeys to create walled gardens within their own environments. Maybe they've already seen that, whoops, you know, the number one thing people are asking for is I want to be able to export these. I want to be able to import them from somewhere else. I don't like not having that ability. So they may have said, okay, well, we tried not to do that. But users are saying they're not going to use Passkeys unless and until they're able to have them be portable.

Leo: I agree. It's kind of a nonstarter; right? I mean, I have to be thinking is my Passkeys all live in this phone. What happens when I get a new phone? Well, actually it does move over to the new phone because mine did. But if I've got an Android phone and I have an iPhone, it doesn't.

Steve: Unh-unh. Nope. Nope. And you could imagine that a password manager like 1Password or Bitwarden, they would...

Leo: They'd love to be the host; right?

Steve: They would want to, yes, yes. They would like to be able to come along later and say, well, you may have started with all your Passkeys in your iPhone, but now you can put them here. And because we're cross-platform, you'll have cross-platform access.

Leo: The problem in general with Passkeys is that there's always the password fallback. And people are going to always fall back to passwords. At which point...

Steve: That is exactly right, Leo.

Leo: They say why am I using Passkeys?

Steve: That is exactly right. Show me a website that does not say "I forgot my password."

Leo: Right. Right. Right. They can't. They're always going to do that.

Steve: Right. Exactly.

Leo: Same reason [crosstalk]...

Steve: There'll always be phishing scams and, you know, yup, yup. Victor, he said: "I powered on a couple of years old desktop that had been unpowered for about a year. It took ages before the desktop was loaded, no errors anywhere, but I decided to try your ReadSpeed. And look at those SSD speeds. Is it time to invest in SpinRite now? If SpinRite fixes this, I will try to encourage my employer to get a site license. Thank you, Mr. Gibson. Victor, long-time SN listener, keep up the good work up to 999 and beyond."

So you have it on the screen, Leo. And this is what we discovered, to our surprise. Actually, nothing quite this horrible, but not good. So what you have on the screen and what I have in the show notes is this system that has three drives. There are two 4TB Western Digital spinners. And the primary drive, the boot drive, is a 512GB Western Digital SATA SSD. Well, we can see that its optimal performance, because remember that what ReadSpeed does, and this is what opened our eyes so much, is it does separate identical benchmarks at five locations: the beginning, the end, the middle, and then the 25% and the 75% points.

This SSD speed is maximum at the 25% point, where it's getting 482.5MB per second. It does not perform well elsewhere. But it performs the worst at the front of the drive, where believe it or not ReadSpeed measured it at 2.2MB per second, down from 482. So, and of course at the beginning of the drive is where the OS lives. And as we know, what we've learned is that the phenomenon of read disturb is that since those files are being read over and over and over, the successive reading without ever rewriting causes the bits to become fuzzy. And it means that the SSD takes much longer to read them and to correct them properly.

To answer Victor's question, yes. Running SpinRite right now on that drive would fix it. And the slight downside, although I wouldn't give it another thought in this case, is that rewriting the entire drive, which is what SpinRite does, is less desirable than exactly locating the spots on the drive which are slow and targeting them for selective rewrite. That will be the feature that SpinRite 7 brings, along with what I think is going to be an extremely interesting map of the read performance across the entire drive surface.

So that's why I'm so excited about there being a 7, why there will be a 7, is I can't wait to develop the technology to do that based on what we've seen, which is what ReadSpeed showed us. And, you know, for these spinning drives we see exactly what we would expect. The beginning of the drives are fast, 182MB per second. Then 169, 151, 122, and 83. Where toward the end of the drive, because the circumference of the inner tracks is shorter, you've got a lower data rate there.

So anyway, Victor, yes, I would love to hear what happens after you run one of SpinRite's read/write passes. You need to refresh the data. It'll be a little slow going there at the beginning, just as it was slow to get your system booted. But afterwards a before-and-after ReadSpeed, and SpinRite 6.1, you can certainly use 6 or 6.1 because it's half a gig so even SpinRite 6 will deal with it pretty quickly. Either one will work, but 6.1 offers before-and-after benchmarks, specifically for this reason, and they're built into the system. So anyway, I'm excited about the future.

Andy Suarez said: "Hi, Steve. Great to hear you cover ECH." That was Encrypted ClientHello from a couple episodes back. He said: "The whole time, though, I kept thinking that something was missing from your coverage, and that was that people should not be thinking that this is providing them cloaked Internet access at all. Knowing what IPs are associated with what websites and services is definitely a thing. For anything other than the smallest websites, there is absolutely a known correlation between IP addresses and the site behind them. And that is not in any way being obfuscated here. Services know the IPs for porn sites and political sites and messaging services, et cetera. And even," he gives an example, http://www.grc.com, as 4.79.142.202.

He said: "They may not be able to see the DNS query with encrypted DNS, or the header request with ECH now. But they are certainly still seeing the IP of the web server that you are sending and receiving packets from; right? So won't tracking just change to keep fresh lists of sites and what their associated IPs are?"

And Andy is right, of course. IPs are still IPs. But the more sites move behind large aggregators like Cloudflare, and thus share IPs, the less clear the user's destination becomes. But as he says, those are more likely to be smaller sites. Big sites typically have their own block of IPs themselves. You know, but it's also unclear how this may change as the popularity of IPv6 grows. At the moment, the lack of IPv4 address space is forcing some of the sharing of IPv4 address space because it's so limited. But IPv6 promises to at least offer the opportunity of demultiplexing websites once again.

So I agree that Andy's reminder is a very good one. Not only was the Internet not initially designed to provide the security and privacy that we increasingly need, it was also never designed to hide where we go. That just wasn't on anyone's radar back then. Tor's concept of so-called "onion routing" is the best solution we've come up with for obscuring our traffic explicitly. A trusted VPN is another good solution, at least for keeping your ISP and anyone snooping along the way from seeing what you're doing. So we would need no one to be able to know what domain IPs and ECH public keys we look up through DNS. Then Encrypted ClientHello to hide the domain within our traffic. And then something like Tor's routing or a trusted VPN provider to hide our traffic's destination. Any one of those three missing, and it's still possible to know what we're doing on the Internet.

Someone using the handle "New EOL," he asked a question that sort of took me a bit by surprise. He said: "Will ransomware be able to encrypt an AES-256 encrypted 7-zip archive?"

Leo: Well, duh.

Steve: And as I said, that one caught me a bit by surprise. You know, it's sort of an interesting thought. This listener is wondering whether encrypting something before ransomware comes along to encrypt it itself could essentially get there first to prevent any further encryption. Unfortunately, it's definitely possible to encrypt something that's already been encrypted. And in fact, speaking of Tor, that's something that Tor, this

onion routing, relies upon. With Tor, a destination URL is encrypted using the public key of the last router in the chain. Then that one-time-encrypted URL is again encrypted using the second from the last router's public key. Then that twice-encrypted URL is again encrypted using the third from the last router's public key, and so on. Then that multiply-encrypted blob is sent to the first router. It decrypts the outer layer of the onion to obtain the identity of the next router in the chain.

But since the destination URL is still deeply buried under multiple rounds of encryption, it doesn't know where you're ultimately going to go. It can only forward it to the next router that decrypts the current now outer shell of the onion which then forwards it to the next one. So anyway, multiple encryptions definitely do work, and their corresponding multiple rounds of decryption, and they do even have powerful uses in the case, for example, of onion routing. So, yeah, unfortunately, encrypting something before ransomware encrypts it won't keep it from encrypting it again.

Daniel Hodgin, he said: "Hi, Steve. Thank you for the new ValiDrive product, and thank you for everything you guys do with Security Now!. I decided to try it out this morning with a few USB keys and SD cards around our office. I then realized I wonder if this can work with SSDs as well with a USB-connected cloning station. Sure enough, I was able to test a 1TB Silicon Power brand SSD from Amazon which turned out to be exactly as advertised. It made me wonder if the same kind of scam can be done with hard drives, SSDs, M.2 SSDs, et cetera, which we are placing in corporate devices and assume our corporate data is being stored just fine once Windows reports the size is as expected."

Well, it's funny he should ask. I actually, I was curious about a high-end device that was available. It was advertised as a USB-C style connectable NVME, so a high-end device. And you can see, I'm holding it up to the camera. It has that physical profile. It looks, you know, it's got a beautiful metal case, you know, beveled edges that are chrome. Well, I opened it up. And what do you think I found, Leo? I found, and I think you can see it...

Leo: Oh, my god.

Steve: ...here, and I also have it in the show notes. It's literally it's your favorite hot-glued SD card sitting in there. Somebody took the trouble of creating a chip to interface a USB-C connector to an SD card to create an incredibly cheap drive. It does not have nearly the advertised memory. So not only is it not an NVME, it isn't even offering the amount of storage available. So, yes, it is certainly the case that there are - that the Internet is sadly full of scams. Wow.

Leo: You've really become the king of this. You've got a whole collection now. Wow. I'm so impressed.

Steve: Well, I wanted to make sure that my software would work. And as I kept getting these things, I mean, even really nice-looking drives like with beautiful packaging and user manuals and stuff, it's like, wow, okay. Just amazing.

So two last feedback-inspired things I wanted to share. Marcus Hitchins, not Hutchins, Hitchins, he said: "Privacy Badger un-rewrites Google's rewritten links in search results." Well, that caught my attention. Everyone knows how annoyed I am that Google's link-hacking, where the link they show you in the search results is deliberately not the link you get when you click it. They are tracking my choices by bouncing my browser through their servers. The one beneficial reason I can see for doing this would be to use human

feedback to help tune their search results. But we're all certain that they're also profiling everyone individually for their own profile building. They make all this noise about their privacy sandbox and how they want to do away with third-party tracking. But what about tracking every link I click on in the search results?

So Privacy Badger is a creation of the EFF. They are people whose intentions we know we can trust. But I've often thought that it was an unfortunate name. You know, Privacy Badger? Really? It's like...

Leo: I think it's a play on Honey Badger. But okay, fine.

Steve: Okay. It fell flat for me. Nevertheless, the idea that I could have something I trust stripping Google's URL link mangling out of Google's search engine results was too good to pass up. So I added it to Firefox and immediately confirmed that, sure enough, just as Marcus said, the links presented to me in Google's search results are exactly now as they appear visually. No more subterfuge.

Leo: Ah. Interesting.

Steve: Yes. It is very nice. So I want to take this opportunity to pause for a moment and touch on Privacy Badger a bit more. First of all, you can find it at PrivacyBadger.org, and it's P-R-I-V-A-C-Y-B-A-D-G-E-R dot org. And it is also GRC's carefully numbered shortcut of the week, so grc.sc/945. And that'll just bounce you over to PrivacyBadger.org. And I'm glad that I'm taking a longer look at it since what it does sounds really cool and interesting and unique.

So in describing it, the EFF writes, they said: "Privacy Badger is a browser extension that stops advertisers and other third-party trackers from secretly tracking where you go and what pages you look at on the web. If an advertiser seems to be tracking you across multiple websites without your permission" - and it ends up that number is three websites without your permission, we'll get there in a second - "Privacy Badger automatically blocks that advertiser from loading any more content in your browser. To the advertiser, it's like you suddenly disappeared.

"Privacy Badger was born out of our desire to be able to recommend" - you know, the EFF's desire - "to be able to recommend a single extension that would automatically analyze and block any tracker or ad that violated the principle of user consent; which could function well without any settings, knowledge, or configuration by the user; which is produced by an organization that is unambiguously working for its users rather than for advertisers; and which uses algorithmic methods to decide what is and is not tracking.

"As a result, Privacy Badger differs from traditional ad-blocking extensions in two key ways. First, while most other blocking extensions prioritize blocking ads, Privacy Badger is purely a tracker-blocker. The extension doesn't block ads unless they happen to be tracking you; in fact, one of our goals is to incentivize advertisers to adopt better privacy practices. Second, most other blockers rely on a human-curated list of domains or URLs to block. Privacy Badger is an algorithmic tracker blocker - we define tracking behavior - and then Privacy Badger blocks or restricts domains that it observes tracking in the wild. What is and isn't considered a tracker is entirely based on how a specific domain acts, not on human judgment.

"When you view a webpage, that page will often be made up of content from many different sources. For example, a news web page might load the actual article from the news company, ads from an ad company, and the comments section from a different company that's been contracted to provide that service. Privacy Badger keeps track of all of this. If, as you browse the web, the same source seems to be tracking your browser across different websites, then Privacy Badger springs into action" - you can just see the badger springing - "springs into action" - grabbing those URLs and ripping them out by its teeth - "preventing your browser from loading any more content from that source. And when your browser stops loading content from a source, that source can no longer track you. Voila," they wrote.

"At a more technical level, Privacy Badger keeps note of the third-party domains that embed content - images, scripts, and advertising - in the pages you visit. Privacy Badger looks for tracking techniques like uniquely identifying cookies, local storage, supercookies, and canvas fingerprinting. If it observes a single third-party host tracking you on three separate sites, Privacy Badger will automatically disallow content from that third-party tracker."

As for their browser support, it's as universal as it could be. Their FAQ asks: "Will you be supporting any other browsers besides Chrome, Firefox, Edge, and Opera?" And they said: "We're working toward Safari on macOS support. Safari on iOS appears to lack certain extension capabilities required by Privacy Badger to function properly. Chrome on Android does not support extensions. To use Privacy Badger on Android, install Firefox for Android. Privacy Badger does not work with Microsoft Edge Legacy. Please switch to the new Microsoft Edge browser. Note that Microsoft Edge does not support extensions on mobile devices."

And they also noted that: "Privacy Badger sends the Global Privacy Control signal" - that was the GPC we did a podcast on a few months ago - "to opt you out of data sharing and selling, and the Do Not Track signal to tell companies not to track you. If they ignore your wishes, Privacy Badger will learn to block them, whether they're advertisers or trackers of other kinds."

And also finally I thought this was very interesting and cool from their FAQ. They had someone ask: "I run a domain that uses cookies or other tracking. How do I stop Privacy Badger from blocking me?" So they reply: "One way is to stop tracking users who have turned on Global Privacy Control or Do Not Track signals. Privacy Badger will stop learning to block that domain. The next version of Privacy Badger to ship with an updated pre-trained list will no longer include that domain in the list. Most Privacy Badger users will then update to that list.

"You can also unblock yourself by promising to meaningfully respect the Do Not Track signal. To do so, post a verbatim copy of EFF's Do Not Track policy to your URL, whatever your domain is, .well-known/dnt-policy.txt. If your domain is compliant with EFF's DNT policy and declares this compliance, most Privacy Badgers will see this declaration the next time they encounter your domain. Also, the next version of Privacy Badger to ship with an updated pre-trained list will probably include your declaration of compliance in the list.

"Note that the domain must support HTTPS, to protect against tampering by network attackers. The path contains '.well-known' per RFC 5785. Also note that you must post a copy of the policy at each compliant subdomain you control. For example, if you wish to declare compliance with both sub1.example.com and sub2.example.com, you must post EFF's DNT policy on each subdomain."

So to me, this reads like a solid piece of mature technology. I hate the name, but I'm glad that I added it to my browsing experience so I wanted everyone to know about it. I

have been looking at it, and so far I've been impressed with what it's been doing. It has been, you know, there's a little badger-like icon in the upper right now of my toolbar. And I see that it is blocking trackers. So, cool. And immediately it cleaned up all the links that I'm getting from Google's search results so they no longer know what I click. And I would really mind them doing it, but the idea that they're showing me the URL that I think I'm clicking on, and they've deliberately changed that, you know, in the HTML behind the scenes, that's just not okay.

So finally, recall the guy from last week whose tweet asked about obtaining the pre-release of SpinRite. That was the tweet that prompted me to mention that it was available to everyone who was interested. This listener was Brett Russell, and as a reminder what he tweeted was: "Hi, Steven. I realize this is a request out of the blue, but my hard disk is failing on my main machine. I own a copy of SpinRite" - and then he had posted the code, which I redacted - "but the disk is set up as GPT" - meaning GUID Partition Format as opposed to old-school MBR. So he said: "...which 6.0 does not support. Any chance you can give me a pre-release of 6.1 to run, please? There is nothing crucial on the drive; but if it dies, it will take some time to bring it all back."

So I replied with the instructions about obtaining the current SpinRite DOS executable pre-release. And then yesterday, when I was catching up on my Twitter feed, I found three tweets from Brett. Apparently he had sent them shortly after I replied. The first said: "Thanks, Steve. Much appreciated. Running it now." Second one he said: "Wow, it's fast. Feels a lot faster than 6.0." He said: "I like the new drive identification screen, as well." And then the third tweet finally he said: "It worked. Machine is up and running, faster than ever, although no errors were picked up, which I've seen before." He said: "Setting up a RAID mirror for when the drive does die. You saved me a lot of time and effort. Thank you."

So anyway, Brett's comment about "no errors were picked up" is a common and sometimes mysterious experience for SpinRite users. After running SpinRite, whatever was wrong will have been fixed, but SpinRite might not explicitly show that anything was done. The reason for this somewhat unsatisfying outcome is that SpinRite induces the drive to deal with marginal problems internally. Before SpinRite's deep recovery is needed, the drive will remove a sector from use, and everything after that will be well again. But drives typically don't report doing so. So all SpinRite and its user knows is that things are better afterwards. Anyway, just wanted to close the loop on that. I guess that's sort of our first testimonial for the upcoming SpinRite 6.1. So thank you, Brett, for that.

Leo: Privilege. Now, see, when I see this title, I think I'm not understanding what you're talking about. You say the "power of privilege." You mean the power of having money and power? Is that what you mean? But of course we who are privileged have power. What the heck could possibly be wrong with that?

Steve: That's right. So today we're going to continue our examination of the NSA's and CISA's top 10 cybersecurity misconfiguration mistakes by drilling down deeply into their number two misconfiguration which is Privilege. What strikes me most about this publication is that each item is quite beefy and worthwhile. Unlike some recommendation lists that we see that feel like they were written by someone pontificating from an armchair, these resulted from their active successful penetration of networks.

So here's what they had to say on the topic of "Improper Separation of User and Administrator Privilege." And then I'm going to embellish a little bit on that afterward. So they wrote: "Administrators often assign multiple roles to one account. These accounts have access to a wide range of devices and services, allowing malicious actors to move

through a network quickly after obtaining one compromised account without triggering lateral movement and/or privilege escalation detection measures." In other words, don't do that. That's not good privilege management. They said: "Assessment teams have observed the following common account separation misconfigurations: excessive account privileges, elevated service account permissions, and non-essential use of elevated accounts."

Under Excessive Account Privileges they elaborate: "Account privileges are intended to control user access to host or application resources to limit access to sensitive information or enforce a least-privilege security model. When account privileges are overly permissive, users can see and do things they should not be able to do, which becomes a security issue as it increases risk exposure and attack surface." And I would add that bad guys that are able to get in as that user are then also able to do things that they should not be able to do.

They said: "Expanding organizations can undergo numerous changes in account management, personnel, and access requirements. These changes commonly lead to privilege creep, the granting of excessive access and unnecessary account privileges. Through the analysis of topical and nested AD (Active Directory) groups, a malicious actor can find a user account that has been granted account privileges that exceed their need-to-know or least-privilege function. Extraneous access can lead to easy avenues for unauthorized access to data and resources, and escalation of privileges in the targeted domain."

Under Elevated Service Account Permissions: "Applications often operate using user accounts to access resources. These user accounts, which are known as service accounts, often require elevated privileges. When a malicious actor compromises an application or service using a service account, they'll have the same privileges and access as the service account.

"Malicious actors can exploit elevated service permissions within a domain to gain unauthorized access and control over critical systems. Service accounts are enticing targets for malicious actors because such accounts are often granted elevated permissions within the domain due to the nature of the service, and because access to use the service can be requested by any valid domain user. Due to these factors, 'kerberoasting,' as it's called, kerberoasting, a form of credential access achieved by cracking service account credentials, is a common technique used to gain control over service account targets."

And finally, Non-Essential Use of Elevated Accounts. They said: "IT personnel use domain administrator and other administrator accounts for system and network management due to their inherent elevated privileges. When an administrator account is logged into a compromised host, a malicious actor can steal and use the account's credentials and an Active Directory-generated authentication token to move, using the elevated permissions, throughout the domain. Using an elevated account for normal day-to-day non-administrative tasks increases the account's exposure and, therefore, its risk of compromise and its risk to the network." In other words, don't do that. Don't be lazy, admins, and use your fancy elevated accounts if you're not actually needing to do so at the time. Drop back...

Leo: I can't believe we still have to tell people this.

Steve: And what's interesting is that these are the ways they have penetrated - they, the NSA and CISA, their red and blue teams - have penetrated users' networks. They

said: "Malicious actors prioritize" - well, and Leo, again, of course we believe it's true, right, because it's easier.

Leo: Sure, yeah.

Steve: It's easier just to use your god login because then you're going to be able to do anything that you want to do.

Leo: Right.

Steve: Their point is that, if you happen to contact a compromised machine under that persona, the malware there can automatically compromise your credentials. Anyway, they said: "Malicious actors prioritize obtaining valid domain credentials upon gaining access to a network. Authentication using valid domain credentials allows the execution of secondary enumeration techniques to gain visibility into the target domain and Active Directory structure, including discovery of other elevated accounts and where the elevated accounts are used. Targeting elevated accounts such as domain administrator or system administrators performing day-to-day activities provides the most direct path to achieve domain escalation. Systems or applications accessed by the targeted elevated accounts significantly increase the attack surface available to adversaries, providing additional paths and escalation options."

And finally: "After obtaining initial access via an account with administrative permissions, an assessment team run by these guys compromised a domain in under a business day. The team first gained initial access to the system through phishing, by which they enticed the end user to download and execute malicious payloads. The targeted end-user account had administrative permissions, enabling the team to quickly compromise the entire domain."

Okay. So there are two primary things that I want to elaborate on and highlight. The first is that idea of mission creep. They called it "privilege creep," and anyone who has been in charge of managing any complex environment of privileges will have seen and been fighting this. I've often observed that code generally does not evolve well. By its nature, code is not designed to evolve. There will generally be an initial grand organizing concept for a project. And that vision will be then realized in code. But that organizing vision influences the code at all levels. The way the large project is decomposed into smaller manageable components. The way those components talk to each other, and what they are able to say, and how. The data structures those components use to capture and manage the project's state. Every one of those things is individually generally static, and they're interdependent, and thus the whole system is quite resistant to change.

I've also observed that it's often the case that, once a project is finished, it really should be entirely discarded, then redesigned and reimplemented from scratch. The reason for that is not necessarily that the project itself evolved during its implementation, but that those who were implementing it evolved. In other words, they were changed by doing the work because the process of solving any sufficiently complex problem teaches those who are solving it how it should have originally been designed in the first place. So for sufficiently complex projects, several iterations of that sort can be required. Unfortunately, of course, outside of academic settings the luxury of multiple iterations is seldom available.

Okay. So how does that apply to account privileges? The primary message here is that the management of privileges provides a tremendous amount of power. And as always,

with power comes responsibility. Any major enterprise's account privileges hierarchy needs to be carefully designed. In a sufficiently complex environment, the concept of "least privilege" is much more easily espoused than it is implemented. Mistakes in either direction of privilege result in problems. Users need varying levels of access, and differing groups of users may have both disjoint and overlapping regions of access at the same level. Autonomous devices also need their own access privileges, you know, like printers scattered around. And it's difficult enough to get everything set up correctly given a static environment without any change. But when did nothing ever change?

The real test is the test of time, brought about by an enterprise's evolving structures and needs. What this means is that any initial design will be almost immediately obsoleted once it faces reality. The asymmetry of "privilege" is another factor: Too little privilege and things don't work or break, printers don't print, users are unable to do what they need to do, and the boss is not happy. Too much privilege and, sure, everything works, but users can get up to mischief, and things are made much easier for the bad guys once they get in. And again, the boss is not happy. So assigning and managing privilege is a bit like walking a tightrope. Stepping off in either direction does not produce a good result.

I think the best advice that can be given is to clearly recognize and accept that the proactive management of "privilege" is a truly important aspect of the enterprise's entire operation. Designing and managing an enterprise's account and access privileges should never be a grudging afterthought. Be sure to give it its due. Fortunately, it's never too late to repair an organization's no longer strictly applicable privilege model. It can be fixed. If this is your responsibility, you likely already know what needs to be done. Don't wait until it's too late.

Next week we'll continue moving through this excellent joint NSA/CISA document, as we keep up with the news of the week.

Leo: You know, it strikes me that Linux users and Unix users in general are constantly warned not to use the root account.

Steve: Yup.

Leo: And they're provided with a great little command, sudo, so that they can easily escalate for one command and one command only. And then it reverts back to the normal user privilege.

Steve: And we know that even Microsoft finally came up, they invented that split token system where, you know...

Leo: Well, but I was going to say I think Windows users are a little brain damaged because so many of them operate as administrator. And, you know, that's why, you're right, they created UAC, but they created UAC because people were not operating as limited users. So they had to create the User Access Control as an automated way of keeping you from getting in trouble. But I think that's caused brain damage. I think that's caused people to think it's okay to operate as administrator.

Steve: Well, and you point out that sudo was easy to use, and it was the ease of use that made it feasible to run with non-elevated privileges.

Leo: Exactly.

Steve: Windows didn't have an easy way.

Leo: No.

Steve: Exactly.

Leo: Yeah.

Steve: And no one wanted to log off and then log back in as an admin in order to install a piece of software.

Leo: And by the way, we should say Apple's done pretty much the same thing with macOS. They have a limited user. And in fact when you create a new user, by default that's a limited user, but everybody runs it in admin. And so Apple does the same thing that Microsoft does with this kind of temporary escalation process. But so much better to do it like Linux does, I think.

Steve: Yes. And as I've said before, operating a truly secure OS is just no fun.

Leo: Yeah, nobody wants that, yeah.

Steve: You can't get anything done. You spend all your time saying yes, I'm sure; yes, it's me; hello, I'm still here; yeah, yeah, yeah, I know, blah blah blah.

Leo: I think you could argue that sudo is a better way to do it than User Access Control, than the escalation prompts. I mean, fairly frequently you'll run a command that says you don't have enough privilege to do that. And many Linux distros, many terminals there's a simple command keystroke that adds sudo to your previous command, so you can easily escalate. I don't know. I just, maybe it's just me, but I prefer that way of doing things. And nevertheless, run as an administrator both on Mac and Windows. So there. By the way, you cannot, without basically rooting it, run as an administrator on either iOS or Android. Why do you think? It's dangerous; right? They don't even have a way to escalate privilege. You know, you're a user, whether you like it or not.

Steve: Yeah. And in those environments it really makes sense for the OS to be the sole arbiter of what runs and how.

Leo: Right, right. I love this show. And I'm so glad you're going to keep doing it. Here we are at Episode 945. At this point I would have been, like, counting down the days. Oh, there's only a year left, oh no. But we're going to until one or the other of us or maybe both keels over, whether it's with exhaustion or the final journey, the final...

Steve: Well, I'm glad I'm going to be here for the end of support of Windows 10.

Leo: Yes. And I hope you'll be here for the end of the Unix epoch.

Steve: Oh, that'd be good.

Leo: In 2038, baby.

Steve: Oh, wow.

Leo: That's my goal. Steve Gibson...

Steve: Yeah, we only have 15 - we only have 15 more years.

Leo: That's not - we're more than halfway there, Steve.

Steve: Exactly.

Leo: Easy-peasy.

Steve: Bye.

Leo: Bye.