Life: Hanging by a PIN

Description: This week we'll start out by clarifying the terms "credit freeze" and "credit lock." Then we have news of the T-Mobile breach from its perpetrator. We examine the evolving and infuriating question of where will Windows 11 run, and we look at yet another newly revealed attack against Microsoft's Exchange Server known as ProxyToken. I wanted to clarify a bit about Tailscale's source openness, and touch on the disturbing revelations shaking the mass storage industry with SSD performance being deliberately reduced once they've been well reviewed and adopted. I'll update our patient SpinRite owners on my recent work and progress. We'll touch on some cellular phone terminology, then conclude by considering the power of the PIN and look at just how much damage it can do.

High quality (64 kbps) mp3 audio file URL: http://media.GRC.com/sn/SN-834.mp3
Quarter size (16 kbps) mp3 audio file URL: http://media.GRC.com/sn/sn-834-lq.mp3

SHOW TEASE: It's time for Security Now!. Steve Gibson is here. Lots to talk about. More details on the T-Mobile breach. The hacker says, hey, it wasn't that hard. Steve explains. He talks about his SpinRite assembly language development tools and secrets. We'll also talk about the cell phones in our lives and how to make them a little bit more secure. It's all coming up next on Security Now!. You won't want to miss this one.

Leo Laporte: This is Security Now! with Steve Gibson, Episode 834, recorded Tuesday, August 31st, 2021: Life: Hanging by a PIN.

It's time for Security Now!, the show where we cover your security online with this guy right here, Steve Gibson of the Gibson Research Corporation.

Steve Gibson: Hey, Leo.

Leo: Hey, Steve.

Steve: Great to be with you again in our 17th year, last day of August.

Leo: Oh, my god.

Steve: Yes, indeed. So this is another one of those that just sort of evolved as I was putting my thoughts together. Today's episode is titled "Life: Hanging by a PIN."

Leo: Uh-oh.

Steve: And this was sort of a different one because it actually caused me to change some of my behavior. I changed some configurations of stuff when I really stopped to think about what it means that so much of our security is tied into our cell phones. So we're going to wind up talking about that; and, actually, a little bit of the crazy acronym soup of IMSI and IMEI and all that other stuff.

Leo: Oh, good, good.

Steve: Yeah, I thought it would be fun to kind of clarify that. But we're going to start out by clarifying, speaking of clarifying, my mess of the terms surrounding credit bureau locking and freezing which I made last week.

Leo: You flip-flopped it.

Steve: I did really stick my foot it. But yeah, I recommended the wrong thing.

Leo: Yeah.

Steve: So we're going to fix that. We've got news of the T-Mobile breach from its perpetrator, we examine the evolving and infuriating question of where will Windows 11 run, and we look at yet another newly revealed attack against Microsoft's Exchange Server known as ProxyToken. I wanted to clarify a bit about Tailscale's source openness and also touch on this sort of side topic, but the disturbing revelations that are shaking the mass storage industry at the moment with SSD performance being deliberately reduced after new SSDs have been reviewed and adopted. And it was, Leo, when our favorite SSD manufacturer was caught doing this that I thought, okay, we have to just touch on this. I want to update our patient SpinRite owners on my recent work in progress, and then conclude by considering the power of the PIN and look at just how much damage it can do. And of course we've got a great Picture of the Week. No longer somebody's closet.

Leo: No more wiring closets, huh? All right.

Steve: It's now someone's box, actually.

Leo: Oh, okay. I'm just wondering, we were talking before the show, we were talking on MacBreak Weekly about soldering.

Steve: Yeah.

Leo: And I was just wondering how long before Instagram recommends a soldering iron to me. And lo and behold, they already did. So took no time. Actually, they recommended a solution instead of soldering. Says: "You still soldering? You should do this." It's like, wow, I haven't even started, and already...

Steve: It is just astonishing.

Leo: It's kind of amazing. Kind of amazing.

Steve: You know, you could just walk outside and look to the heavens and ask a question.

Leo: They will know.

Steve: And then check your phone.

Leo: Yeah. We're getting to that point, yup, yup. You know, just say it out loud, somebody will respond. It may be a burning bush. You don't know. But somebody will respond. Now it's time for the Picture of the Week.

Steve: Okay. So we have in this single frame some Greek columns in the background, and a blond woman who's sort of wearing a toga-esque wrap. And this shows her staring at the screen of her laptop. I'm not sure how she had a laptop back in Greece. But the caption is "Pandora's Inbox." And you see her thinking with a thought bubble: "It can't hurt to open one little attachment, can it?"

Leo: Oh, boy. I love it.

Steve: Yes, Pandora's inbox. Do not open the attachment. Okay. So I wanted to begin this week by first and immediately correcting my mistake from last week about the terms "credit freeze" versus "credit lock." It turns out that the terms matter, and I got them mixed up. It's a credit freeze that most people will want to use, not a credit lock. A "freeze" is the term used by the federally mandated no-cost option which all four major bureaus must provide to prevent queries made from would-be creditors from being honored.

By comparison, the term "lock" is sort of generically but not specifically used to designate an optional product which may take the form of a somewhat costly subscription which may be made available by the various credit reporting agencies with differing features from one company to the next. So it's that lock product which, for example, has the additional bells and whistles that might be tied to a mobile phone app which the bureau produces to allow for rapid locking and unlocking on the fly. So I made the mistake of not digging deeply enough into the terminology before talking about it last week.

Leo: I should have corrected you because I was sitting and doing it while you were talking. And I did the credit freeze, which thanks to the Economic Growth, Regulatory Relief, and Consumer Protection Act of 2018 is free.

Steve: Yes.

Leo: They used to make money on this. The credit reporting agencies would charge you for the freeze, and most importantly, charge you to unfreeze. And those fees ranged from quite a bit of money in some states to free in other states. But there are no - it's now absolutely free. And so I put a freeze on all three, and I can unfreeze it. They give you a PIN. It's easy to unfreeze. Of course you'll want to do that before you try to get credit. I think everybody should just do this. They don't want you to do it.

Steve: Correct.

Leo: Because they make money selling your information.

Steve: Correct. I think when I did it, because I did lock mine down when I first talked about it, I think I maybe paid like $10 each for then the three major bureaus.

Leo: Yeah, it's the unfreeze they make money on.

Steve: And I did notice that they do have, on the freeze option, they do have a temporary unfreeze. So anyway, I did want to thank all of our listeners who did know better and who let me know that I'd gotten things somewhat muddled up.

Leo: The fraud alerts last a year. Those are the - I think that's what you were talking about. The fraud alerts don't last.

Steve: No, I just didn't appreciate the distinction between "freeze" and "lock." And so I do now. Being federally mandated, it costs nothing, as you said. It also appears that the temporary auto relock is part of the freeze service. I saw that in several cases. And that makes it possible to, again, for no charge, to briefly lower one's shields in order to, I mean, like when you know you've got some creditors who are wanting to check your credit. I also noted some options to temporarily and selectively allow queries to be honored for specific and specified would-be creditors. So if this begins to sound a bit like you're programming a firewall, that analogy is pretty accurate. In firewall parlance, rather than the "permit any" rule, we want to run with the "deny all," and then optionally add rules to permit specific creditors to query for and receive reports.

And you may have noted that I mentioned the "four" major credit reporting bureaus before. A fourth upstart called Innovis has been added to the tradition big three of Experian, Equifax, and TransUnion. And Leo, you'll want to go there. It turns out they're big enough to be real. So everyone will want to place a reporting freeze on their records with Innovis (I-N-N-O-V-I-S), as well as the others, since someone having access to all of your personal data from a data breach might deliberately choose to attempt to obtain credit from some lender who is known to use Innovis specifically because being lesser known they are less likely to have had their reporting frozen.

Leo: Very good. So I'm doing that right now.

Steve: Innovis, I-N-N-O-V-I-S. So now it really is the case. They were bought from something about CBS a few years ago, and they've been growing. They offer the standard range of services. And boy, I would like to say, like, no more. We have to freeze them all. So stop doing this, folks. Hopefully there won't be a fifth one who thinks, oh, yeah, there's room in the market for five. No. There's no need for any more. Four is it.

Leo: It's interesting. Innovis just gives you a form, you fill it out, and they say, okay, good. That's it. You're done. And then you'll get a confirmation letter by mail. So I don't know if I'm going to ever worry about unfreezing them. I think I'll just leave that on.

Steve: Yeah. And in fact you only ever need to unfreeze them if by some chance you were trying to prove your creditworthiness, and the person said, I'm sorry, your credit's locked. It'd be like, really. You're using Innovis? Okay. And of course then you'd have to unfreeze for them, or tell them to go to one of the original three. I don't know.

Anyway, T-Mobile. Thanks to the fact that the attacker, a U.S. citizen, believes that he's currently outside the long arm of U.S. law enforcement, we're now learning quite a lot about the who, what, and why of his quite successful data exfiltration attack on T-Mobile. And none of what we're learning flatters T-Mobile's cybersecurity. The Wall Street Journal turns out had been chatting with the purported attacker via Telegram for some time. They've confirmed that his name is John Binns (B-I-N-N-S).

John is a 21-year-old U.S. citizen of Turkish descent who relocated from the U.S. back to Turkey three years ago. John was reportedly discussing details of the breach before they were widely known, and T-Mobile received their first indications of trouble when they were notified of the breach by Unit 221B, a cybersecurity company that monitors the dark web for their own purposes. So they saw that John was offering the sale of all of this data breach material on the dark web, and Unit 221B said, uh, T-Mobile, do you have a problem that you haven't told anybody about?

So John told the Wall Street Journal that his attack against T-Mobile was conducted from the comfort of his home in Izmir, Turkey, where he lives with his mom of Turkish descent. His American father died when he was just two, and he and his mom moved back to Turkey three years ago when he was 18. He reportedly uses the online handles IRDev and v0rtex (with a numeric 0), among other handles. And he's alleged to have an online track record that includes some participation in the creation of a massive botnet that was used for online DDoS attacks four years ago when he was still in the U.S. and 17 years old.

There was some interesting reporting that I didn't put into the show notes because it just sort of seemed murky about him alleging that he was captured and tortured by U.S. agencies and that this attack on T-Mobile was retribution of some sort. And I thought, okay, well, I'm going to kind of leave all that out of the official notes. But he did tell the Wall Street Journal that he penetrated T-Mobile's defenses last month, in July, after scanning the company's known IP space, looking for weak spots and using what the Journal referred to as "a simple tool available to the public." Maybe Shodan. Who knows? During his perusal of T-Mobile's public-facing IP space last month, John discovered an exposed and unprotected router, as it has been described. Again, we don't have much more detail than that.

Leo: This puzzled me because I'm not sure - I didn't quite understand what - but go ahead.

Steve: Yeah. And the reason it's a little murky is that T-Mobile is pushing back, trying to say, oh, this was a sophisticated attack that required brute forcing across our network. And John is saying no.

Leo: Really.

Steve: There was no protection. You know, I'm good, but I'm not that good. Anyway, the router reportedly had a different configuration from T-Mobile's other public routers. Again, we don't know what that means exactly. The details are still scarce. But from that router, John said that he managed to then break into T-Mobile's large datacenter located in East Wenatchee, Washington. And then inside the Wenatchee facility, John said he had access to more than 100 servers containing the personal data of millions. By August 4th he had exfiltrated millions - now, the reporting said "files." I assume that means "records," so probably fewer files. Millions of somethings, records - thanks to what he told the Journal was the mobile phone seller's pathetic security.

Note also that the company that's big on magenta, but apparently not so big on cybersecurity, never had any idea that he was loose inside their network, roaming around freely at will. Independent observers within the cybersecurity community noted that the fact that the theft included records from prospective clients as well as former, long-gone customers, demonstrates the extreme degree to which no one inside T-Mobile was practicing any data management hygiene. It's like they just weren't bothering to delete old cruft. It was taking up space; but of course storage is cheap now, so why delete it? Well, why indeed?

And I'll add that the data John was able to make off with was not encrypted. That sort of sensitive data at rest should be encrypted so that only the system that's validly retrieving it for its proper business purpose should then be able to decrypt and read it. But the files themselves should never be stored in simple exportable and readable plaintext. But T-Mobile's was.

So when you consider a publicly exposed router that can be broken into, apparently no network intrusion monitoring and response - because they never knew - and a complete lack of long-term data management and policy, the fact that this is the sixth such data breach in just the past few years should not be surprising.

As we've discussed before, I've always taken the position that the person in charge should not necessarily be fired when something bad happens on their watch because those can be teachable moments, vivid teachable moments, and the result might be much improved security in the future. But now I'm not so sure. T-Mobile is making me rethink the beneficent tolerance approach to employee management. Maybe it's time for some heads to roll because they don't - no one seems to be getting the message; right? It's just like it hasn't been fixed in years.

They did say, I mean, there's a lot of CYA going on now, and mea culpas, and the president saying, oh, no, we've - I think he said they hired Mandiant and KPMG to come in. Mostly I'm sure it's to salve Congress, and there is a class-action lawsuit that has already been launched, of course, and blah blah blah. So something has to happen to make these guys take this more seriously because it doesn't seem that six data breaches in half that many years has been enough to make that happen.

Okay. So where will Windows 11 run, Leo? At the moment, no one seems to be sure, and Microsoft is not helping. The best advice I have is to take the time to check in with the maker of your machine, motherboard, or whatever to see whether they're offering any

updates to increase compatibility with Windows 11. We noted two weeks ago that Asus had deliberately updated the firmware of 207 of their motherboards specifically to ease their users' move to Windows 11. Bravo. So it's worth checking to see whether whatever system you're using might have some similar updates available for it.

Unfortunately, Microsoft continues to confuse. They appear to be holding onto their baseless "must have TPM v2.0" requirement, even though Windows 10 has no such need. And we hear that they're gradually adding older Intel chips to the approved list. But Intel chips are famously backwards compatible. So I'm at a loss to understand what's really going on there, since no one imagines that Windows 11 is really so different from Windows 10.

Leo: Just to add some fuel to this fire, they did a briefing with Tom Warren at The Verge and said, you know, you can install Windows 11 on any of the machines that we say are incompatible, but you just have to download the ISO and install it. And we're not going to really support it, but it'll work. So we know it'll work.

Steve: Yeah, yeah.

Leo: So that just shows you this is some sort of marketing thing they're up to. There's not - no technical reason for it.

Steve: Well, yes. And I'm going to give them a dose of 'tude here in a minute, how I feel about this. They just added the Intel Core X-series, the Xeon W-series, and the Intel Core 7820HQ chips. Just, you know, why not? However, they did state that no AMD Zen 1 processors would be compatible. So there.

Now, when you look back at the history of Microsoft's major version upgrades for Windows, the jumps from, for example, NT to 2000, 2000 to XP, XP to Win7, Win7 to Win8, and Win8 to Win10 have all brought actual substantive changes. But this current move from 10 to 11 doesn't have that feeling at all. You know, rounded corners, ooh. Going to change the alignment of the Start Menu. So it's difficult to understand why Windows 11 cannot simply run everywhere Win10 does. And Leo, as you just said, it kind of does. Unless Microsoft for some reason just doesn't want Windows 11 to run everywhere. But that will create exactly the sort of stratification that Microsoft worked so hard to avoid when they attempted to force everyone to move to Windows 10 whether they wanted to or not.

Now, apparently Microsoft will be attempting to resolve some of this confusion by adding a "Windows 11 compatible" message to the Windows Update screen. I have a snapshot of one of those in the show notes. And over on the right you can see where it says, with a green checkmark, "This PC can run Windows 11." And it says: "Great news. Your PC meets the minimum system requirements for Windows 11." Then it says: "Specific timing for when it will be offered can vary as we get it ready for you." Okay. But "Specific timing for when it will be offered can vary as we get it ready for you"? What the hell does that mean? Really. We're really not sure when Windows 11 will be ready to run on your machine, despite the fact that we're announcing here the good news that your machine will be able to run it, just not when.

So somehow Microsoft appears to have succeeded at completely removing all of the science from computer science. So now it's, well, no one around here who we've been asking seems to be completely certain about when exactly this next big release of Windows 11 may be ready for your particular machine because we haven't really figured

out what we're going to do yet. Yeah. Wake me up when you have figured this out. I can't wait.

Leo: What if the real reason for this is not technical at all? Well, I mean, we know the real reason for it is to sell new PCs in the fall.

Steve: You think? You think?

Leo: But what if the real reason is they want, look, they want, I mean, we know this backward compatibility is actually a source of security problems.

Steve: It is.

Leo: So what if what really they're trying to do is, admittedly, they're forking the road, but saying, you know, we want to get the most possible people in a more secure environment, TPM eighth generation or later. Is the eighth generation or later susceptible to Spectre and Meltdown? Or does that not come into this?

Steve: Yeah, I mean, everything kind of still is.

Leo: They all are, yeah.

Steve: Yeah. And BIOSes have been updated in the past. I mean, they could say something like you must have an updated chip that isn't. But remember they're the ones who are updating the firmware. Windows provides that on-the-fly patch of the chip. So it knows what it's doing. I mean, it really...

Leo: They're just trying to make a more secure ecosystem. What if that's the reason? We'd be for that; right?

Steve: Actually, their problems, I mean, all the problems we talk about, they don't surround chip architecture.

Leo: Right.

Steve: As we know...

Leo: It's Microsoft.

Steve: Yes. As much noise as was made about Spectre and Meltdown, there was never even one instance of it being a problem. Yet Exchange Server looks like Swiss cheese.

Leo: Right.

Steve: So, you know, and I heard you and Paul and Mary Jo talking about this. I mean, what they would like to do is stop supporting their older legacy stuff. For example, 32-bit support is finally disappearing; right? So that should help a lot.

Leo: That will help a lot, yeah. The Win32 subsystem is a big problem.

Steve: So I could see saying, okay, if your chip can't, you know, if you were running Windows 1032, sorry, use that for something else. And that's just too far old. But what is really infuriating people, and again, the muddle of this, the lack of, well, maybe. And, oh, if you downloaded the ISO, gee, it'll run. But we'll be less excited about it. What? Oh, and Leo, I'm not kidding you, just an hour ago when I booted the Win10 machine that's right up here, it's the one I'm looking at that all I use it for is connecting to TWiT for this podcast, none of its desktop icons appeared. And the onscreen clock that auto starts wasn't there.

Leo: Oh, boy.

Steve: But this is Win10; right? Where all of the science has been removed. So I just shrugged and restarted.

Leo: It worked.

Steve: And the second time Windows felt - it felt more like finishing up. So it was in a good mood. It was a little warmer, maybe. I got all of my desktop icons back, and everything appears to be present and accounted for. And I can't wait to see what Windows 11 has in store for us.

Leo: The more I use Windows, the more I love Linux. That's all I'm saying.

Steve: Yeah. First we had - and I'm going to turn this down.

Leo: You're selling a lot of - he's selling a lot of SpinRites these days. That's good.

Steve: First we had ProxyLogon, the original mess with Microsoft Exchange Server, that they didn't patch until its use in active attacks had become public. Then, last week, we discussed ProxyLogon's kissing cousin ProxyShell, similar but different. And today we have what's being called ProxyToken as Exchange Server's latest to become public vulnerability. It was finally patched after more than three months in July, with July's Patch Tuesday last month.

Microsoft was originally informed of this vulnerability through the Zero-Day Initiative (ZDI) by a Vietnamese security researcher on April 5th of this year. So, yes, a little more than three months for them to patch it. Maybe it's just that - I don't know.

Leo: It's hard to think of a good reason; isn't it.

Steve: When you remove the science from computer science, what you get is, you know...

Leo: It's just a computer.

Steve: The ProxyToken vulnerability would, for example - that's today's latest - allow an attacker to surreptitiously add an email forwarding rule to a user's mailbox.

Leo: Oh, man.

Steve: So that all emails addressed to that victim would also be sent to an account controlled by the attacker. How convenient. The vulnerability essentially allows a remote attacker to bypass authentication and make changes to an Exchange email server's backend configuration. The flaw was reported, as I said, through the ZDI, the Zero-Day Initiative program, and it exists due to a pair of problems in the Exchange code. Two of them. First, requests that contain a non-empty cookie named "SecurityToken" that are redirected from the frontend to the backend are not authenticated because, after all, it has a security token, even if it's bogus; right? It's non-empty. So basically the attacker just adds a cookie called "SecurityToken" to their query headers, and you bypass all authentication. Second, HTTP 500 error responses expose an Exchange control panel canary token.

Combining these two oversights, its discoverer explained that a so-called ProxyToken - because it uses cookies - attack is possible, and that attackers can easily make requests to any part of the Exchange backend, including its users' control panels and settings. And since the details of this attack went live yesterday on the Zero-Day Initiative blog - after all, they waited more than, wow, like a month and a half since July's Patch Tuesday, which is when this was fixed - server owners should expect threat actors to weaponize this vector.

And weaponizing the vector is exactly what we saw happen last month when attacks against Exchange servers took off after the details about the ProxyShell vulnerability were first published online. Remember that's where what's-his-name, Orange Tsai, gave his speech at Black Hat, and he said just enough for some clever people to further reverse engineer what he hadn't said, and then the exploit went public, and within a day or two it was being used. And it's now being used by this LockFile ransomware to get into people who still for whatever reason, those what is it, 1,500, or was it thousand, Exchange servers which still haven't been updated. So presumably they still have - they haven't updated to the ProxyToken attack, and so have at it, everyone.

I put this under Errata, not because it really was, but just because it sort of fits there. I received a DM tweet from a listener who wrote: "Hi, Steve. Maybe I missed it, but Tailscale is open source." And he provided a link to Tailscale on GitHub where, sure enough, they have an account. And it's true that much, though not all, of Tailscale is open source. I'm mentioning it because our listeners have just gone nuts over it. Tailscale explains: "This repository contains all the open source Tailscale client code and the tailscaled daemon and tailscale [command line interface], the CLI tool. The tailscaled daemon runs primarily on Linux. It also works to varying degrees on FreeBSD, OpenBSD, Darwin, and Windows."

They provide a link to their Tailscale for Android client. And for non-Android Tailscale clients they write: "The macOS, iOS, and Windows clients use the code in this repository, but additionally include small GUI wrappers that are not open source."

Leo: Stay away from small GUI wrappers, that's all I'm saying.

Steve: Yeah. You want to - yeah, exactly. Or bring some paper towels. So just to be clear, I have not yet dug into any of this. But it does look as though someone who wanted to take more responsibility for setting things up with the code that Tailscale has developed and is open source, who wanted to roll their own solution, could definitely do that using the open source code provided within Tailscale's repository.

And it sounds very much as though you'd want to be using a Linux box to run the tailscaled daemon, though presumably such users would then also be responsible for keeping those things up to date, which Tailscale would normally be doing for you if you were coming through their front door and using their regular service, which I imagine all of our listeners have done. Remember that it gives you up to 20 different endpoints that you're able to use in their free offering. Tailscale provides links to both their stable and unstable package builds. They support a vast array of Linuxes and also a Windows installer. So anyway, I wanted to give the Tailscale folks credit for and to acknowledge the open source aspects of their offerings.

And also there's been a lot of discussion of this in GRC's newsgroups, and I thought it was just worth putting it on our listeners' radar because I know our listeners will care. Both Tom's hardware and ExtremeTech have been following the growing controversy over the practice that's been discovered among an increasing number of SSD makers who have been caught initially releasing a new high-quality product for review and analysis by the tech publications and presumably by their large OEMs for subsequent inclusion in future systems and then, once that's been done, quietly replacing their initial fast and high-quality semiconductors with significantly lower cost and lower performance components while not changing the device's part number to make this apparent in any way.

I didn't have it in the show notes. But for example, in some cases they are changing the use of TLD to QLD chips going from three-layer to quad-layer, which is of much lower - it's a higher density, but lower cost and lower performance chip replacement. I mean, universally agreed.

What this means for us is that the important and typically carefully considered opinions of the reviewers of these products may not actually be reflective of the devices that we eventually purchase after relying upon such reviews. And it also means that tremendous commercial pressure is then placed upon those unfortunately fewer and fewer companies who are resisting this fraudulent bait-and-switch behavior. Though this is off topic for the podcast, I obviously have a huge personal interest in the whole topic of mass storage and its performance, reliability, and recoverability. And I know our more tech-savvy listeners do, too.

A few weeks ago, on August 16th, ExtremeTech's Joel Hruska (H-R-U-S-K-A) posted a piece titled "Buyer Beware: Crucial Swaps P2 SSD's TLC NAND for Slower Chips." I have a link in the show notes for anyone who wants to read the whole thing. He started off by saying: "Crucial has come under fire after a retest of its well-reviewed P2 SSD demonstrated that the company has swapped from its launch design to a much inferior product. This is not the first time SSD manufacturers have been caught bait-and-switching customers in this fashion, and it's deeply frustrating to see companies willing to subvert their own review process.

"The scheme goes like this: Sample an SSD out to reviewers and spec it reasonably well. Once all the reviews are in, swap out the components for inferior products that are not as power-efficient and/or do not offer the same performance. That's what Tom's Hardware found when it investigated Crucial's P2 NVMe M.2 SSD after reviewing the initial part shipped by Crucial. Crucial has swapped the TLC NAND it originally shipped with QLC NAND, and not terribly good QLC NAND, at that. The new version of the P2 has two fewer NAND chip packages than the original, and significantly fewer total dies. This reduces the total potential bandwidth the SSD controller can achieve and further harms the performance of the 500GB drive. The average power consumption on the QLC drive is lower, at 1.49 watts, but total power efficiency is actually worse because the savings do not make up for the dramatically slower performance. If full drive performance on the P2 was already bad, it's downright abysmal on the P2 with QLC NAND."

So that was on the 16th. Exactly a week ago, on August 24th, Joel followed up that piece with another titled "Western Digital Caught Bait-and-Switching Customers with Slow SSDs." Again, the link in the show notes. He said: "When I wrote about Crucial's decision to swap inferior NAND flash into its products without updating the reviewer community or announcing a separate SKU, I noted the problem was a one-off. While this has happened before, it's typically been the exception, not the norm. Guess that was too much to hope for.

"According to a report from Chinese tech site Expreview, the WD SN550 Blue, which is currently one of the best-reviewed budget SSDs on the market, has undergone a NAND lobotomy. While the new SSD variant performs on par with the old drive that WD actually sampled for review, once you exhaust the SLC, that is to say single-level cache, NAND cache, performance craters from 610MB/s as measured by THG to 390MB/s as measured by Expreview. The new drive offers just 64 percent of the performance of the old drive. This is unacceptable. It is unethical for any company to sample and launch a product to strong reviews, only to turn around and sell an inferior version of that hardware at a later date without changing the product SKU or telling customers that they're buying garbage." His words.

He says: "I do not use the term 'garbage' lightly, but let me be clear. If you silently change the hardware components you use in a way that makes your product lose performance, and you do not disclose that information prominently to the customer, ideally through a separate SKU, you are selling garbage. There's nothing wrong with selling a slower SSD at a good price, and there's nothing right about abusing the goodwill of reviewers and enthusiasts to kick bad hardware out the door."

And sadly, Joel followed this with his latest review in this series just last Friday the 27th by posting: "Samsung Is the Latest SSD Manufacturer Caught Cheating Its Customers." He said: "In the past 11 days, both Crucial and Western Digital have been caught swapping the TLC NAND used for certain products with inferior QLC NAND without updating product SKUs or informing reviewers that this change was happening. Shipping one product to reviewers and a different product to consumers is unacceptable, and we recently recommended that readers buy SSDs from Samsung or Intel in lieu of Western Digital or Crucial.

"As of today, we have to take Samsung off that list. One difference in this situation is that Samsung isn't swapping TLC for QLC. It's swapping the drive controller and TLC for a different, inferior drive controller and different TLC. The net effect is still a steep performance decline in certain tests. We've asked Intel to specifically confirm it does not engage in this kind of consumer-hostile behavior and will report back if it does."

So Joel's post goes on to show photos of the peeled-off top label of a Samsung 970 EVO Plus SSD to reveal very different chips underneath the label. And of course there's no problem with them doing that. They're free to put whatever chip they like on their

products. But if that's done after the drives have been reviewed to shave their cost and the users' performance, I agree with Joel that's not okay.

Leo: Is it possible it's chip shortages?

Steve: That's - yes.

Leo: I mean, they should certainly disclose.

Steve: They may have exactly that, you know, like have no choice. But they would have to then suspend that part and just say, sorry, this part is temporarily not available. Here's the best one we have to offer. And just to put a bow on this, Western Digital said: "In June of 2021 we replaced the NAND in the WD Blue SN550 NVMe SSD and updated the firmware." So that confirms that an undocumented parts change they made was responsible for this 50% reduction in writing performance. They said: "At the time, we updated the product data sheet. For greater transparency going forward, if we make a change to an existing internal SSD, we commit to introducing a new model number whenever any related published specifications are impacted." So they didn't say when the performance changes, but they did say when any related published specifications are impacted. So that's something.

Anyway, I wanted to put this on everyone's radar to make sure that our listeners knew that this was apparently going on within the industry. As you noted, Leo, there is a chip shortage which is impacting all kinds of things, threatening maybe to raise prices somewhat, just due to a price increase all the way back at the fab sellers end. Everyone knows that I believe in benchmarks and in performance testing. GRC's DNS Benchmark has become the industry standard tool, with more now than seven million downloads of that little puppy. And the first thing I did, as our listeners know, with SpinRite's new driver technology was to create the ReadSpeed drive benchmark. One of the things we immediately learned was that there were some very weird things going on inside our SSDs. They do not behave at all like the solid-state RAM we wish they were. Now is not the time for me to dig into those particular weeds. But everyone can rest assured that this has my attention, and that a future SpinRite is going to be quite revealing.

Leo: Oh, good.

Steve: Yeah. And speaking of SpinRite, after an unusually long code-writing stint without doing any testing, I recently switched back to testing and debugging all of the new code I've been writing. Since writing in assembler allows me to reassemble and link my code in less than half a second, the cost to rebuild my entire project is like zero. And in fact I use that to check for typos on the fly. So the rhythm that's developed for me over the years is, I guess I would call it "fast iteration," where I'll write a chunk of code, then stop right then to immediately test it to verify that it does what I expect and need. Then I'll move forward knowing that what I've left behind is at least mostly ready for the world. Then I'll move on to the next stage. So this creates a solid foundation for whatever follows.

But I had stopped working that way when I began the rework of SpinRite's device driver model to "abstract" all of SpinRite's drive interaction behind a single custom I/O function. I talked about that a month or so ago. I stopped iterating because for a long time I haven't been able to. I needed to pretty much take SpinRite down for that rework and

just hold my breath while I reassembled it in this new very different way. Then once it was back up, I thought that I should just keep pushing forward rewriting the code that would then use the new I/O abstraction system.

I did enough of that to see that my first design needed a bit of tweaking, as I mentioned on the podcast at the time. And that regarded SpinRite's handling of the drive's built-in error correction, which I realized should have been transparent at the abstraction layer since there's no reason to return an unfinished request to our caller if we've been able to autonomously correct and obtain the data for the user. So I reworked the five different drive interfaces to autonomously handle their drivers' reports of corrected errors, essentially eliminating that as a possible cause for returning early, and it's just handled by the driver.

But what wasn't sitting well was that I had written so much code that hadn't had any testing. So last week I decided to stop writing and to switch back to testing and debugging. And I have to say I've been having a wonderful time verifying expected behavior, when everything works as I designed, and tracking down the causes of unexpected mysteries when things don't work as I expect.

Leo: You're one of the few people who loves debugging.

Steve: Oh. It just...

Leo: It's fun, though; isn't it. If it works, if you solve it, it's a great feeling.

Steve: Well, yeah. And I've been coding for so long that I'm not afraid of debugging.

Leo: Right. You know, right.

Steve: It's not like something's going to get me.

Leo: Right.

Steve: I'm going to get it. But, I mean, I'm really intrigued when it's like, okay, why?

Leo: Why doesn't that work? Yeah.

Steve: What happened? Yeah. And as I've often said on the podcast, whenever I find a mistake, I don't try to sweep it under the rug. I stop. I cross my arms, and I think, okay, how did that happen? What went wrong that caused me to do that? And so I always take it, like I said of IT managers who don't seem to learn from their mistakes, for me it's a teachable moment. So in fact I posted over in GRC's spinrite.dev newsgroup last week that I realized I could easily be pulling all-nighters because chasing down these little mysteries, it was so much fun and was so compelling. And I never want to stop; right? It's like, okay, I'm looking at the clock, and it's getting to be later, and Lorrie is, like, infinitely patient. She's just there with her headphones on. We have a Roku remote that

has that headphone jack. So she's able, she's actually watching "The X Files" for the first time.

Leo: Oh, that's awesome.

Steve: Having a wonderful time.

Leo: That headphone jack is great; isn't it? Because you can be in the same room, but she doesn't have to bug you and vice versa. That's, I agree, yeah.

Steve: Yeah.

Leo: What do you use to debug? You use GDB, or is there some tool that you use, or you step through...

Steve: Oh, no, I'm in DOS.

Leo: You can't do print statements in assembly.

Steve: No. And in fact I spent a lot of time essentially coming up with a debugging environment which would work because what I had been using was a great tool called SoftICE.

Leo: Oh, yeah.

Steve: Which was - it ran as a DOS memory manager. So it put the system into protected mode. And then so it used that to hide itself. And also being in protected mode it was able to hook all the things that it needed to in order to get control. So it was sort of a hypervisor for DOS. And it took up no conventional memory because it was able to live itself up in extended memory. But the problem is, SpinRite has become its own memory manager. It runs the chip in real mode so that it can have DOS, and then it plays a game. It briefly switches into protected mode, and it uses what is apparently a bug from the very first 286 chips where, if you switch into protected mode and change the protected mode descriptors, and then you switch back into real mode, the descriptor cache is not flushed.

Leo: Ah, good.

Steve: They miss that.

Leo: So you can examine the state of your program.

Steve: Well, actually what happens is, when I'm in protected mode, I set the - normally real mode descriptors are 64K because that's - a real mode segment is 64K. But that's in the chips that allow protected mode, which is to say anything from a 286 on, that's just a register. Basically, in real mode, the chip is pretending to be in real mode. It's not actually in real mode. Real mode is faked by setting up a kind of a - by setting up the segmentation registers to have the same limitations that they originally had in real mode, when you just had an 8080 and an 8086. So what happens is SpinRite briefly switches to protected mode, sets the segment extents, which are 32 bits, to 4GB, and then switches back. So now what I have, because the chip forgets to flush the cache of the segment descriptors, is I have segments that are actually flat. That is, they're actually 4GB segments.

Leo: Oh, wow.

Steve: Not 64K segments.

Leo: Wow.

Steve: Which allows me to access all of the system's RAM from real mode. So it is a hack, but it's a hack that a lot of the gamers use, the DOS gamers all did this, and it's known as long real mode or flat real mode. And so I added that capability to SpinRite. But that made it incompatible with SoftICE. So what that meant was I had to go back, I had to fall back to an earlier debugger known as Periscope. Periscope was written by an old buddy of mine, I mean, because, you know, back in those early DOS days we all knew each other. I knew Bob Smith, who did Qualitas and 386MAX that was the memory manager.

Leo: Remember that, yeah.

Steve: And Brett Salter wrote Periscope. And unfortunately Brett died about four years ago, or I would have had him make some adjustments to Periscope for me. But I can't do that. So Periscope only runs in conventional memory. And it takes up a lot of conventional memory. So, in fact, just the other day I had to - or actually two days ago I posted in the GRC SpinRite dev group that I'd had to move the 4,000-byte screen capture from - when you start SpinRite, it sort of takes over the whole screen. It's a full screen kind of text GUI. And when you exit SpinRite, it wipes that off the screen, returning you to the screen you had before. Well, that requires saving the screen you had before.

Well, that was taking up 4,000 bytes, and I had run out of conventional memory. So two days ago I moved that save buffer into extended memory. You know, it was easy to do. Just change some pointers around for the copy, the screen copy. But the point is I am at the point where I no longer have any conventional memory room. So but the good news is I pretty much have all of the code written, and now I'm in a debugging and testing mode. I did note in my posting when I said that I was having a hard time quitting at the end of the day, and Lorrie was being very patient with me, but I had learned the hard way that I am no longer coding as a teenager. And that if I do, in fact, pull an all-nighter, I end up giving back all of that time.

Leo: Exactly.

Steve: That I thought I was saving.

Leo: Because the next day is shot.

Steve: The next day, exactly. I am useless. I'm drooling. I'm just, what? Huh, what, honey? Anyway, so I don't do that now.

Leo: The folks at Big Nerd Ranch in one of their books say caffeine is not a substitute for sleep. Get a good night's sleep.

Steve: Anyway, overall it's all really going well.

Leo: Somebody asked me this before. So there's a style of coding that's in vogue these days, I actually use it and really like it, called "test-driven development," where you write tests for modules, and the module isn't completed till it satisfies the test. And you try to write tests that - you don't do that.

Steve: If I had the time, I would love to. What that does is it allows you to prevent making regression errors.

Leo: Exactly.

Steve: And that's exactly what happened is that...

Leo: It's more useful in functional programming because you can say this is provably, because there's no mutability, it's probably going to work every time with the same input, same output, so you can test it.

Steve: Yes. Yes.

Leo: So it works very nicely there.

Steve: Being able to complement code you write with test code for what you write, it is invaluable. It's a little difficult because what I'm doing is all about interacting with actual hardware.

Leo: So it's all mutable, really. That's the problem.

Steve: Yeah. And the routines are testing against hardware function and their interaction with it, rather than just sort of like, wow, did I get, you know, did it actually produce a Fibonacci series or not?

Leo: Right, right, right, exactly, yeah.

Steve: So it's not something like that. But anyway, I am really happy with the way it's going. I didn't foresee the path that I'd be taking when I began. But I'm very pleased with all the decisions I've been making up to this point. And although the outside of SpinRite will reveal some signs of an entirely new SpinRite underneath, which is actually what has happened, it will largely look the same. It will be blazingly fast and will work on drives of any possible sector count. And thanks to its new I/O abstraction model which is underneath, it will also be ready to graciously accept the native USB and NVMe support that will be added after the move off of DOS so that we're able to boot on newer systems that have removed support for the BIOS, and thus DOS, and only boot UEFI-based operating system code. So anyway, I'm getting there. I'm in debugging mode. I actually stopped Sunday night with a mystery that I can't wait to get back to this evening. So, yeah, I'm having a great time.

Leo: He doesn't watch mysteries on TV, folks. He creates his own and solves them himself.

Steve: Yes, while Lorrie loves learning about Sculler and Muldy, I mean, Mulder and Scully. And she's not seen the two movies, and so I am going to watch those two feature-length movies with her.

Leo: Fun, fun.

Steve: It's going to be fun.

Leo: Yeah.

Steve: And it's water time.

Leo: Beverage time.

Steve: And then we're going to talk about how our lives are hanging by a PIN.

Leo: Oh. I know this, too.

Steve: Okay. So the subtopic - wow. I thought I...

Leo: You didn't fix it yet. Not fully moistened. Moisten.

Steve: The subtopic for today's podcast would be our cellular phones as a critical weakest link in the chain. In addition to the best general purpose advice I give to everyone, which is to operate your life with your credit reporting frozen at all four of the credit reporting bureaus, after this breach T-Mobile subscribers especially should be sure to change their account PINs; and, when doing so, to make their new PIN as long and complex as possible, if there's any latitude there within the limitations of a PIN, and to use digits having the highest possible entropy that have nothing to do with their life.

Never use any date of significance or anything that someone who knows you might guess, since a determined attacker might have been able to gather sufficient information to effectively know you. Since the PIN was one of the items of information stolen that's under a subscriber's control, and since it's crucial for verifying your identity to your cellular carrier, it really must be changed.

All of the various cellular carriers now offer some form of SIM-jacking protection. This is also known as "SIM-swapping" or "port-out scamming." It occurs when a scammer contacts your cellular provider and pretends to be you. The trouble is, in the case of T-Mobile, a scammer will be armed with everything T-Mobile knows about you, including the account PIN that was stolen. I'll come back and talk about this SIM-jacking in a minute.

I recently set up a new phone with my provider, Verizon. I think I mentioned it on the air. The battery of my beloved iPhone 10, which I'd owned for years, had outgassed and ballooned, popping the screen out and causing it to bend alarmingly. This was actually the second time that had happened with this phone, so I had Apple replace the battery to create a hand-me-down, having decided that it was time to move to an iPhone 12. So that I wouldn't be without a phone during the repair, I first purchased the new 12 and had Apple deliver it using their incredible "by the way, that's us knocking at your front door with your new phone" service. It literally took about an hour to have the new phone delivered and in my hand. So that process was painless.

But today, thinking back over just how painless the process was, in the context of the T-Mobile breach, I realized that it was a terrifyingly simple thing for me to move my cellular service over to the new phone. The only thing Verizon needed from me that wasn't otherwise generally available public knowledge, was my account PIN which, yes, was also part, as I've said now several times, of the T-Mobile data breach. That PIN was effectively my entire proof of identity. They didn't need email, nor for me to first respond through the previous phone, which I had already decommissioned. And if the phone was claiming to be dead, lost, or stolen, they still need to be able to move forward.

So everything boils down to your account PIN. I simply provided those few digits of my PIN, which they confirmed matched the one they had on file. Then I read off long strings of numbers - my new phone's ICCID and IMEI, which as I said I'll talk about in a second. And just like that, my new phone was live with my phone number. And I didn't think twice about it. It's like, okay, cool. Thank goodness that wasn't harder.

But once a bad guy has taken over your phone number, your actual phone will lose service. That's your first clue that life is about to become much more complicated, and not in a good way. Naturally, any of your online services that still rely upon SMS text messages sent to your registered-with-them phone number, or can somehow be made to rely upon sending a text message, will immediately be subject to compromise and takeover. Your email provider cannot send you an account recovery email if you claim to have forgotten your email account password. So they'll have your phone number on file in order to send a text message to your phone for emergency account recovery. And those of us who have a Gmail address know that Google, for example, is pretty good about prompting us from time to time to make sure our phone number with them is still

current and correct. Guess why? So they can recover accounts from forgotten email passwords.

But now the attacker who used your PIN to impersonate you to your cell provider to get your phone number moved over to his phone will receive that emergency account recovery text after claiming to have forgotten your email logon password. And the first thing they'll do will be to change your email password in case the account recovery process itself didn't automatically change it for them. Now you're locked out of your own email. So next they'll start accessing your various services, clicking on the "oh my, I forgot my password" link, and thus obtaining access to any of your accounts when that account recovery link is sent to your email, which they now control.

In fact, if your password manager provides email or SMS-based account recovery, your attacker can go right to the source, obtaining your entire master password archive to learn not only every account name and password, but also everywhere you have accounts, and your username and password manager will log them right on. Or if you've decided to simply have your browser memorize your logons, they can now log on as you on the same make and model of browser you use, have that browser synchronize all of its settings and history and shortcuts and passwords, then browse though its long list of saved usernames and passwords for everywhere you have accounts.

It really is a nightmare scenario. And this entire cascade of events is only prevented by someone with ill intent not knowing your incredibly weak, short, decimal four-digit PIN, which probably is currently your birthday or year of birth or anniversary date or street address number. Because, after all, you wouldn't want to forget it. We're dealing with cellular phones that are also being used by those who have no regard for security. So the security of the entire system has been reduced to serve the lowest common denominator user. Unfortunately, because an SMS-enabled cellular phone is also the one thing that everyone now has, this weakest link has also evolved to become our universal identity verifier. And it deserves no such respect.

I went over to Verizon's FAQ page about PINS. There, Q&A question #1 is: "What's an account PIN, and why do I need one?" And they provide the answer: "Having a PIN helps to keep your Verizon mobile account and personal information secure." They said: "It's the primary way we verify you as the account owner when you contact customer service." And then it goes on to explain: "If you don't have an account PIN when you contact customer service for account changes or information, you'll be asked to create one to continue."

So take a moment to think about this single-point-of-failure vulnerability, and the cascade of disaster that flows from someone who's somehow able to obtain access to your phone number. I just changed my PIN at Verizon - I really just did this morning - because this has all made me realize that I haven't been taking the lack of security of my cellular phone account PIN seriously enough. And I'm using LastPass. I also just removed my phone from LastPass's SMS account recovery where I did have it configured because, hey, extra security backup. Right? Wrong.

There are times when convenience can create convenience for the wrong person. I will gladly take responsibility for never forgetting my master LastPass password. Actually, I can't forget it because I've never known it. It's a bizarre string of nonsense that is carefully written down and stored offline. And that highlights my point. The weakest link in the chain protecting my master password vault is not my insanely long password. It's the fact that I had deliberately established a weak SMS-based backdoor into my account protected by a four decimal digit PIN and interactions with a very non-native English speaker, who claims her name is Nancy.

Leo: Everyone knew her as Nancy, of course.

Steve: So, yeah. If you're a T-Mobile subscriber, oh my god, change your PIN immediately.

Leo: You know, just looking at mine, I made it 15 digits thanks to you, which is the longest they allow. They only allow digits, alas. But then they have security questions. Which I think is the worst possible way to validate your identity.

Steve: Yup. Again, somebody who knows you, who looks up your Facebook screen or your Twitter...

Leo: It's on Wikipedia, yeah. Yes.

Steve: Yes.

Leo: So I use nonsense answers. And they do now allow Google Authenticator. But there's a default off switch on requiring two-factor. So I've turned that on. But those security questions, those are the weak link, unfortunately.

Steve: And so, you know, if by some chance you never bothered to assign a PIN to your cellular account...

Leo: It wasn't necessary at first.

Steve: Right. It's not necessary. And you might have just thought, eh. Well, then there is truly nothing protecting your cellular account and your entire life from hostile takeover. I didn't have to tell her the name of my best friend. She didn't ask me because I did know my PIN. Now it's changed. And if there was ever something that you wanted maybe even changed from time to time, although I've never been a fan of changing passwords, make up four random digits. Don't have it be something that's meaningful. And given the general critical weakness of the security of our cellular phone accounts, it might be worth taking a moment to seriously reconsider, as I just have, the value of enabling SMS account recovery for your most critical authentications. I really did just disable that insecure recovery feature from my password manager, and I feel a bit of relief. Talk about reducing one's attack surface.

Leo: A lot of places, though, require it. Like my bank, there's no way I can turn off SMS authentication.

Steve: Then use a bogus number. I guess, though, they will verify...

Leo: No, you need it. They send you the code.

Steve: They send a text.

Leo: So it has to be there. And it's really frustrating.

Steve: I don't know. Maybe use like a spouse's text.

Leo: Yeah, or a landline or something, yeah.

Steve: Something, so that it cannot be - it cannot receive recovery links. I mean, but again, our life hanging by a PIN. It is really insecure. So T-Mobile calls their SIM-jacking thing, this service they offer, account takeover protection.

Leo: I turned that on, too. But it really looks like a come-on for, you know, it keeps you from porting your number and things like that; right.

Steve: Yes. I have a link in the show notes. You can just google T-Mobile account takeover protection. AT&T has this under their manage extra security option. Then look for the wireless passcode section. And if your carrier's Verizon, you dial *611 and ask to place a port freeze on your account. Again, as you said, Leo, it is unclear what true security and protection this may afford. But anything that purports to keep your phone number associated with its current handset seems worth turning on. Clearly, the consequences of it being maliciously moved, as we've just seen, can be really devastating.

So I'll just finish with some acronyms. I titled this "ICCID, IMEI, and IMSI - Oh, My." While we're on the topic of cellular phones, a review of those wacky long strings of identifier numbers might be in order, especially since the IMEI and IMSI numbers were reportedly part of the T-Mobile breach data. At least for some subset of the breached subscribers, they are now known.

Okay. So the ICCID, that is the Integrated Circuit Card ID. This is a unique and unchangeable international SIM card identifier. It's the 18- to 22-digit number that can be seen printed on the outside of the SIM, and it smells like the work of a committee that got out of control. For example, every SIM ICCID begins with the digits 89. Why? Well, because this is an industry standard code that indicates that this is a product for use by telecommunications networks. Now, you ask, but aren't all SIMs for use in telecommunications networks? Yes, of course. That's why the number is always 89. But if it's always 89, then why have it at all? Exactly. Ask the committee. They're quite pleased with their work here.

Following the obligatory and entirely redundant 89, we have one to five digits for the country code. The U.S. is country code 1. After the country code is the Mobile Network Code (MNC), which is a string of one to four digits associated with the mobile network operator that issued the SIM card. This code represents the SIM card's home network. For example, MNC of 004 is the code for Verizon Wireless. The ICCID then ends with a guaranteed-to-be-unique-globally string of digits which allows this SIM card to be uniquely identified everywhere for all time. While it's possible to change the information contained within the SIM, for example including the IMSI, this identity, that is, the identity of the SIM card itself, the ICCID remains fixed at manufacture.

Okay. Next up, the other one of the three is the IMEI. That's the International Mobile Equipment Identity. It's a unique number also immutably assigned to every mobile handset or other cellular-capable device. And, as with the ICCID, this number is burned into the phone and cannot be changed. Now, that's, of course, cannot be legitimately changed. Bad guys can change anything they want to.

And lastly, the IMSI. This is the International Mobile Subscriber Identity. And as its name suggests, it's the thing that can be changed as a subscriber's phone number moves from device to device, and thus from SIM to SIM. It's a unique identifier that identifies a subscriber to the wireless world. It specifies the country and the mobile network to which the subscriber belongs, and then which subscriber within the network. So those are the numbers, a bunch of gibberish you can find, in the case of an iPhone, if you go to the Settings and then the General and then About, if you scroll down and find it. And I read the first two out to the gal on the phone after giving her my PIN, and bingo. Suddenly my phone number was in a new phone. Thank god it was a phone I was holding in my hand and not a phone that some bad guy had obtained. Had he done so, I would have been pretty screwed. Today, less screwed. So that's good.

Leo: Well, as usual during this show I've changed all my settings. I take your cautions very seriously and act upon them.

Steve: I really did. The idea of an SMS to recover your password manager is the dumbest thing I have ever...

Leo: Oh, well, I don't use that, thank god.

Steve: I had. I had. In effect, I thought I had two-factor authentication on. But I remembered there was some glitch in LastPass at some point where two-factor started having a problem, and so I had disabled it and never went back and turned it back on again. So it's on now, baby. And I'm glad for it.

Leo: I use my YubiKey. And I figure that's got to be as secure as possible. You'd have to knock me over the head and steal my keys to use that.

Steve: Anyway, I just wanted to take this opportunity of this T-Mobile breach and the fact that the PINs got loose to just sort of step back and ask everyone to think about what things have an SMS text message recovery. Given that it is as flimsy as it is, is that really what you want?

Leo: Yeah, yeah.

Steve: And I think in many cases probably not.

Leo: Of course I log onto my T-Mobile account, the first thing it says, "Cybersecurity incident. T-Mobile as determined that unauthorized access to your personal information has occurred, including access to your name, date of birth, driver's license number" - notice how they phrase this - "government identification numbers."

Steve: Uh-huh.

Leo: "And Social Security numbers. We have no indication that personal financial or payment information was accessed." Who cares? You've got everything else. "Take action." And then I did take action, including changing everything.

Steve: What is your financial information? How much your bill was every month for, you know, who cares about that?

Leo: They didn't get my credit card. Oh, well, they got everything else. They can get a credit card in my name. Well, I've got it all locked down now, thanks to you. Not just T-Mobile, but all my credit and everything. I'll never be able to buy another car, but that's okay. It's probably a good thing.

Steve Gibson, he's the guy. If you don't listen to this show - well, you do because you're hearing what I say.

Steve: They're here.

Leo: If other people didn't listen, they missed this; right? And so tell your friends, let them know, this is the show, especially if you're in IT or security or just, you know, you just want to know about this stuff and protect yourself. This is the place.

Steve's website is GRC.com. He has the show there. Actually, you know, the 64Kb audio, but also two unique formats. A 16Kb audio version, it sounds a little scratchy, but it's small. And he commissions Elaine Farris to do very nice transcriptions of every episode. So you can read along as you listen. You can also search the transcribed version to find a part of the show to listen to. It's really a nice benefit. All of that at GRC.com.

While you're there, pick up a copy of SpinRite, 6.0 right now, but soon to be 6.1. You can hear the gears working. He's working hard, burning literally the midnight oil. Well, you probably don't use oil. But he's burning something at midnight. And the world's finest mass storage recovery and maintenance utility is still there and is great. And if you get it now, you'll get an automatic free upgrade to 6.1, as well. GRC.com. You can leave him feedback at GRC.com/feedback. He also takes DMs on his Twitter account. That is @SGgrc on Twitter. He has great forums, too, by the way, at his website.

We have the show at our website, TWiT.tv/sn. We have 64Kb audio and video. And you can also get it, of course, there's a full-time YouTube channel. It's full of Steve's Security Now!, all 834 episodes. You can also subscribe on your favorite podcast client and get it automatically. Do us a favor, leave a five-star review. If you do use a podcast client that allows reviews, let the world know. Tell them. Say it in the review. If you missed Episode 834, you're not safe. Go listen. Get that information.

Thank you, Steve. Have a great week. Put the headphones in the Roku. Enjoy some TV. Go to sleep early. And we'll see you next time on Security Now!.

Steve: Okay, buddy. Bye.