C.O.M.B.

Description: This week we'll begin by following up on last week's headline-making attack on the Oldsmar, Florida water treatment plant with new details that have since come to light. We'll then take a look into last week's Patch Tuesday event and at some of the sadly broken things that have once again been fixed. Also, anyone using Adobe's PDF tools, Acrobat or Reader, needs to update. We're going to look at a dangerous Android App with 1.8 billion (with a "b") users, and at Microsoft's note about the rise of web shells, which dovetails nicely into this week's WordPress add-on disaster. I'll briefly update about my past eventful week with SpinRite, which includes a 25-second movie of new SpinRite code running. Then we'll take a look at the recent discovery of the largest list of email and password combinations ever compiled, and what we can each do about it.

High quality (64 kbps) mp3 audio file URL: http://media.GRC.com/sn/SN-806.mp3
Quarter size (16 kbps) mp3 audio file URL: http://media.GRC.com/sn/sn-806-lq.mp3

SHOW TEASE: It's time for Security Now!. Steve Gibson is here. We've got lots to talk about. An update on that hack on the water company in Florida. Massachusetts has some ideas for what they can do next time. We'll also talk about Patch Tuesday. Wow, some doozies in there, including a patch for the Microsoft Fax Server. Might want to get that one. And then we'll talk about the Android app with more than a billion users that's laden with malware. It's all coming up next on Security Now!.

Leo Laporte: This is Security Now! with Steve Gibson, Episode 806, recorded Tuesday, February 16, 2021: C.O.M.B.

It's time for Security Now!, the show where we cover your privacy, your security, your overall appearance on the Internet with this guy right here, Steve Gibson. He is your Internet groomer at GRC.com.

Steve Gibson: Okay.

Leo: I guess. I don't know where I'm going with that one.

Steve: Yes, okay.

Leo: Hi, Steve.

Steve: So good afternoon, my friend. So a lot of stuff to talk about today. The podcast is COMB, which is the abbreviation for Compilation of Many Breaches.

Leo: Compilation of Many Breaches. Okay.

Steve: Yes, COMB. It just came out a little over a week ago, and it's - I have it in the show notes - something like 3.1, 3.27, 3.some, like three billion unique email addresses and password combinations all in plaintext. And so we know that hackers do this. We know that sites lose control of their backend databases. There's been over time a migration from passwords in plaintext to passwords poorly hashed, meaning like one iteration and no salt. Then we got into the multi-iteration, and then we got into the salt, and then we got into the per-user salt.

So over time the quality of the protection being offered in the sites' databases has improved, but they do keep losing control of them. And of course the bad guys, those who maybe don't have the skills to go and leverage SolarWinds-style attacks, it's like, well, let's build compilations of all the usernames and passwords. Anyway, a new big one has come out. We haven't talked about this idea of email address and password lists for a while. So I thought it would be fun to sort of take a look at where the state of the art is, how have these lists grown over time, what does it mean that they're getting older and thus staler, and so probably less relevant, but new ones are being added.

Anyway, that's what we're going to talk about here as we wrap things up for the week. But we've got some follow-up on last week's podcast title. And the big headline grabber, I saw it on lots of news reports, the attack on the Oldsmar, Florida water treatment plant. It really got people's attention when you say, you know, 111 times more lye being added to the drinking water. They said, what? What? So we've got more details about that.

We're going to take a look at or into last week's Patch Tuesday event for Windows at some of the sadly broken things that have once again been fixed. Also anyone using Adobe's PDF tools, Acrobat or Reader, needs to update. There was sort of a very important update for that also last week. Normally I don't talk about Android apps. But in this case an app is being very irresponsible about its design, and it has 1.8 billion with a "b" users.

Leo: Oh, wow.

Steve: So what we need to talk about. I even knew the name, SHAREit. It's like, oh, yeah, heard about that.

Leo: Oh, yeah, yeah, yeah, yeah, lot of people downloaded that, yeah.

Steve: Uh-huh, yes. And they're all in trouble. Also, we're going to take a look at Microsoft's interesting note about the rise of web shells, which dovetails nicely into this week's WordPress add-on disaster. I'm going to briefly update about my past eventful week with SpinRite, which includes a 25-second movie of the new SpinRite code that I talked about last week, now running.

And then we're going to take a look at, as I mentioned, this recent discovery on the dark web of a massive compilation - sorted, organized, searchable, indexed, database query-

able, it's got all the bells and whistles - and what it means. And of course we always have a great Picture of the Week. This one is really apropos. I thought I was going to have a story about what it ties to, but that story is too big. So it's next week's topic.

Leo: Oh, interesting. Oh. But it is very, you're right, it is so true.

Steve: It's just perfect.

Leo: Yeah. And it looks like it's xkcd once again.

Steve: Looks like it.

Leo: Got to give Randall Munroe a lot of credit. He did a very good xkcd this week on how mRNA vaccines work by comparing them to the Death Star plans from Star Wars. And I think it actually is probably the best description of how an mRNA vaccine works I've ever read. So it's good.

Steve: Very cool.

Leo: Just go for the exhaust port; okay? Okay, Steverino. Picture of the Day.

Steve: Yeah. This is great. It's kind of difficult, I mean, if anyone sees it, they'll instantly get it. But to describe it for our listeners rather than our viewers, imagine a big construction of blocks of different sizes representing software modules, and they're all kind of piled up on top of each other, kind of like a house of cards of all different sizes, some up on their end, others standing on top of them, you know, balancing really precariously. I'm sure there's some wooden block game of this sort where you have to keep adding blocks to it, and then you have to start taking them away kind of thing, to have the whole structure stay together.

Anyway, the point is that in this particular construction, down at the very bottom of this very complex, multi-hierarchy, tiered conglomeration is one little kind of a peg standing on its own, and the whole thing is labeled "All Modern Digital Infrastructure," to sort of represent that, yes, this is the way today we build these things.

Leo: It's like a house of cards out of blocks. It's just, like, nutty, yeah.

Steve: Exactly. Exactly.

Leo: Jenga, maybe.

Steve: It's all modules coming from here and there that are conglomerated together, and they've all got - they rely on an API below them, and they expose an API to their upper surface, and something else hooks onto that. And it's all kind of glued together. Anyway, this one little stick down at the bottom is labeled - oh, I should also explain that

this little stick down at the bottom is basically supporting the structure of this entire thing such that, if this stick were removed, it's immediately clear that the entire thing would just tumble off to the right and fall apart in a billion pieces. So this little stick is labeled, "A project some random person in Nebraska has been thanklessly maintaining since 2003." And, I mean, this is funny because it's so true.

Leo: Yeah. OpenSSH or NACL or, yeah.

Steve: Yes. Pick your - right, right. Pick your super popular, it sort of became a de facto standard when no one was looking. And now...

Leo: One guy.

Steve: Everything is built on it. Yeah.

Leo: One guy.

Steve: It's Felix in Nebraska.

Leo: Felix, he's in charge. And he's, by the way, people yell at him all the time. They're constantly complaining. Nobody's giving him any money. He's doing it out of love. This is really kind of the way it is.

Steve: Because it's not handling the triple back tick expansion of some random command. And it's like, you need to add this. And he's like, okay, well, you know, no. Anyway, so the point is, okay, we all are depending upon Felix more than we are aware. Until he says, "I'm done."

Leo: The hell with it.

Steve: And then it's like, wait, wait, wait, wait, wait, no.

Leo: It's not worth it. Yeah, yeah.

Steve: So in a variation on Felix, we have Oldsmar, Florida. As I mentioned last week, I hope the scare of the kind of near miss that we had might serve as a bit of a wakeup call, not only to other water treatment facilities, but also to sort of the larger industrial control sector in general, whose security pretty much must be wanting. I mean, what we always see is that anything we've never looked at closely is full of holes. And the only reasons our browsers are as strong as they are from a security standpoint is that we're just like - they're getting so much attention. But those things that don't get attention, they're like Felix with his little peg up there, and just hope that it stays where it is.

Well, as it turns out, later last week, something kind of like that might have happened, at least in small part. The State of Massachusetts Department of Environmental Protection

posted a cybersecurity advisory for public water suppliers with the subtext, "How public water suppliers can guard against cyberattacks on water supplies." And I don't know, like, who put the State of Massachusetts Department of Environmental Protection in charge of the larger water supplier world. But at least they were the ones who made the statement, and it did get picked up far and wide by a lot of the press.

So it's a little bit more specific to this particular issue than I was hoping for since it must be that the security problems with industrial control extend to every application of the SCADA control systems that operate all this stuff. But it did offer in its advisory a few interesting new tidbits about the attack which were not public knowledge when it occurred 11 days ago.

Okay. So what they said was, open letter: "Dear Public Water Supplier. We appreciate your attention to cybersecurity and the recent incident in Florida. Here's a more specific description of the events and suggested protective measures." They wrote: "The FBI, Department of Homeland Security, U.S. Secret Service, and the Pinellas County Sheriff's Office have issued a joint situational report that concerns the water sector. The EPA is providing crucial information from this report to the WSCC" - I don't know what that is, water system something council or something maybe [Water Sector Coordinating Council] - "and the GCC" - which I'm sure is not the language - "for awareness," they wrote. "EPA recommends that all water systems implement the mitigation measures listed at the end of this report where applicable."

They said: "On February 5th, 2021, unidentified cyber actors obtained unauthorized access on two separate occasions, approximately five hours apart, to the Supervisory Control and Data Acquisition (SCADA) system used at a local municipality's water treatment plant." Of course we all know this. "The unidentified actors accessed the SCADA system's software and altered the amount of sodium hydroxide," they said, "a caustic chemical used as part of the water treatment process. The water treatment plant personnel immediately noticed the change in dosing amounts and corrected the issue before the SCADA system's software detected the manipulation and alarmed due to the unauthorized change." Okay, well, so that was good to know.

Leo: That's news. That's good, yeah.

Steve: That there was some backup in case somebody was literally asleep at the console and not kind of curious about why the mouse pointer was mysteriously moving itself around the screen. Which, you know, he saw five hours before and didn't think twice because, yeah, that happens because they do have remote access of that sort.

Leo: And they all have the same password. Did you see that?

Steve: I know. I know.

Leo: Oh, my god.

Steve: I know. "As a result," they wrote, "the water treatment process remained unaffected and continued to operate as normal." So they said: "The unidentified actors accessed the water treatment plant's SCADA controls via remote access software, TeamViewer" - of course you used TeamViewer in your previous presentation of our sponsor, Leo, appropriately. They said: "...which was installed on" - and there's some

confusion about this. They're saying one of the several computers at the water treatment plant. I've seen other reports that said on all the computers at the water treatment plant. But there is universal agreement that they are, regardless, all sharing the same login authentication.

So they said: "...which was installed on one of several computers the water treatment plant personnel used to conduct system status checks and to respond to alarms or any other issues that arose during the water treatment process." In other words, SCADA. They said: "All computers used by water plant personnel were connected to the SCADA system and used the 32-bit version of the Windows 7 operating system. Further, all computers shared the same password for remote access." So that was the red flag that went up.

Leo: Yeah, no kidding.

Steve: "All computers shared the same password for remote access and appeared to be connected directly to the Internet without any type of firewall protection installed." Okay. So, I mean, like, really? Not even behind a router? Maybe that, you know, who knows? So they said: "Recommended Mitigation." I mean, so that reminds you of Windows 98, right, back then. Anyway, recommended mitigation: "Restrict all remote connection" - so this is what went out in this announcement, this advice: "Restrict all remote connections to SCADA systems, specifically those that allow physical control and manipulation of devices within the SCADA network. One-way unidirectional monitoring devices are recommended to monitor SCADA systems remotely." Meaning you can look, but you cannot change.

They said: "Install a firewall software/hardware appliance with logging and" - this would be good - "ensure it's turned on." Because that would be good. "The firewall should be secluded and not permitted to communicate with unauthorized sources." Okay. "Keep computers, devices, and applications, including SCADA/industrial control systems software, patched and up-to-date. Use two-factor authentication with strong passwords. And, finally, only use secure networks and consider installing a virtual private network (VPN)." And they concluded with: "Implement an update and patch management cycle. Patch all systems for critical vulnerabilities, prioritizing timely patching of Internet-connected systems for known vulnerabilities and software processing Internet data, such as web browsers, browser plugins, and document readers."

Okay. So everyone who listens to this podcast will have pretty much become security-aware enough for none of that advice to be the least bit surprising. And, you know, it does sound a little bit generic; right?

Leo: Well, it's a start. I love it that they're using Windows 7, shared TeamViewer password, I mean, you couldn't do it any worse. And it's open to the public Internet, apparently, not secluded. I don't know if secluded's the right word. That sounds like it's in a quiet little grove, a forest grove.

Steve: Give it a vacation.

Leo: Yeah. But I get the point, absolutely, yeah.

Steve: So there has been some speculation about the timing of the attack within the security industry because that massive database breach or reveal that I mentioned which contains 3.2 billion unique email/password pairs was leaked online just three days before the attack. That database contained 13 entries relating to the Oldsmar facility, that is, those containing the domain ci.oldsmar.fl.us. We'll be discussing the details and consequences of the publication of this massive list at the end of today's podcast. But in my mind it would be a huge coincidence for the two to be related, given that the newly leaked list contains, as I said, 3.2 billion entries.

So what happened is that a security firm, after seeing the Oldsmar, Florida exploit, queried this newly released database and found, yes, 13 entries that were relevant. But given that you have 3.2 billion entries, you could probably put just about anything in there, and you'll get some results. So I think that a more plausible explanation was offered by Christopher Krebs, the former and founding head of the U.S. Cybersecurity and Infrastructure Security Agency, CISA. In Congressional testimony, Chris told a House of Representatives Homeland Security Committee last Wednesday that the breach was very likely the work of a disgruntled employee, or maybe ex-employee. And I think that is extremely likely. This is not to excuse the obvious lack of security throughout the installation. When you hear that all the machines are using the same password, like when that news gets spread widely, okay, you already have some serious security egg on your face.

One other note is that, especially since the pandemic, TeamViewer has become to remote control what Zoom has become to teleconferencing. But whereas the pressure of its public exposure and its very public security failures rapidly matured Zoom's security, as we talked about starting about a year ago, TeamViewer has remained pretty much unchanged, and those knowledgeable about TeamViewer are not impressed. Many of those within the industrial control industry who have been interviewed since the Oldsmar incident have observed, unfortunately, that they often see TeamViewer installed at sites, and just shake their heads. Yeah, it works, but it's anything but secure. And the reason it's chosen by Johnny to manage his grandparents' PC when they get tangled up with what button to press, is that it is so drop-dead simple to use.

And I'll note that TeamViewer does offer multifactor authentication. But of course you need to turn it on. And if the problem was a disgruntled employee, they might have current credentials. If it was an ex-employee, then of course somebody who was at the Oldsmar facility in charge of security, and you wonder whether the phrase "in charge of security" is just an oxymoron at this place because apparently nobody was, they would of course, after terminating someone, you'd have to go around and do the equivalent of changing the locks on all the doors. You'd need to change the passwords and the one-time password authentication secret and so forth. But it seems very unlikely that anyone did.

So again, yes, none of this in this instance is rocket science. This bar was so low that anybody could have jumped over it in order to unfortunately start poking around with the SCADA system at the plant. Again, the best possible outcome is that, because this got so much attention in the press, all the other admins or people in charge have to have thought, ooh, crap, we don't even have passwords. So let's hope this was a wakeup call and that a lot of attention got paid.

My great concern, and I don't want to be right on this, is that we keep seeing that anything that hasn't received serious attention is seriously lacking in security. And there have been lots of people worrying about lack of SCADA security. The fact that these things just aren't being attacked, as far as we know, means that they're not getting lots of attention. Well, let's hope that the concern is great enough that security really is an issue, and that we're not going to have to get a rude wakeup call when we have a nationwide power outage, and like the equivalent of what we did, apparently we did, we

and the Israeli cyber folks, with Stuxnet, when the Iranian nuclear centrifuges got blasted by Stuxnet. That was a SCADA attack, very clever and high end because, as we know, they were not on the Internet. All these systems are because everyone wants the convenience that that affords. But if you're going to do that, you really need to take responsibility.

So speaking of taking responsibility, a lot happened with last Tuesday's patch batch. Overall, Microsoft addressed 56 security vulnerabilities, including 11 rated critical and six that were already publicly known. So, yes, let's get those patched. And as expected, they did finally lower the boom on the use of unencrypted remote procedure calls to prevent the protocol fiddling that was what led to and enabled last year's Zerologon mess. So they gave everybody lots of time to get that resolved and fixed. They don't want to break anything, but they do want to get this, what was basically a serious protocol mistake, locked down. They can't really change it without breaking things, but they can at least wrap it in a secure authenticated tunnel in order to keep it from being abused. The 56 CVEs span the .NET framework, Azure IoT, Azure Kubernetes Service, Microsoft Edge for Android, Exchange Server, Office and Office Services and Web Apps, Skype for Business and Lync, as well as Windows Defender.

The biggie that was found being exploited in the wild carries a surprisingly low severity rating of just 7.8, which puts it in the "important" range. But researchers noted that it deserves attention above some of the critical bugs in terms of its patching priority. This problem exists in Windows' Win32k operating system kernel module. We've talked about that often. It is an elevation-of-privilege vulnerability. It would allow a logged-on user to execute code of their choosing with admin root system level privileges, so in the context of the kernel. And again, it is being used maliciously in the wild.

So it's good that last week's updates eliminated it. I guess it probably got a 7.8 because you had to already be a local logged-on user. You had to be on the system with access to the API. However, if you had that, that allowed you to bypass all of the system's deliberate security provisions against allowing such a user to have root privileges. And so this cut right through that. It's fixed. And again, it was found being used. So it certainly was useful.

There's also a publicly known critical flaw in the .NET Core and Visual Studio, but details are being closely held. Dustin Childs, who's Trend Micro's Zero Day Initiative guy, said: "Without more information from Microsoft, that's about all we know about it. Based on the CVSS severity scale, this could allow remote, unauthenticated attackers to execute arbitrary code on an affected system." He says: "Regardless, if you rely on the .NET Framework or .NET Core [and pretty much everything does now]," he said, "make sure you test and deploy this one quickly." Believe it or not, there are also two critical-rated remote code execution flaws in Windows Fax Service.

Leo: Well, they've got to fix that.

Steve: Fax Service?

Leo: Yeah.

Steve: Really? Does anyone fax anything anymore?

Leo: Oh, believe me. There are a lot of people who still use that. Hard to believe.

Steve: Wow.

Leo: I know.

Steve: In any event, Microsoft said that an attacker who successfully exploited either of these two critical vulnerabilities could take control of an affected system, yes, by fax.

Leo: Wait a minute. You could send a fax to it, and it would hack it?

Steve: Yeah. Yeah.

Leo: Oh, that's a good one.

Steve: Yeah. Yeah.

Leo: Wow.

Steve: Then, from receiving a fax, the bad guy could install programs; view, change or delete data; or create new accounts with full user rights.

Leo: You know there are tons of businesses, medical offices, and more that use this for inbound faxes. This is actually probably more serious than we're thinking.

Steve: Well, so to me this sounds like, for most of us, one of those services you want to uncheck and remove from most, if not all, Windows systems. Again, we always want to be minimizing our attack surfaces. We recently talked about turning off unneeded application layer gateways in our routers. Unneeded and unused features are inherently dangerous. If a system that has never needed to send or receive a fax doesn't have its fax service running or installed, even if the fax code is vulnerable, it's not there to be attacked.

So in Windows Control Panel, on the Add or Remove Apps page, there's a link for adding or removing Windows features. You can browse that list and remove the crap that Microsoft installed in order to minimize their tech support calls. They'd rather have the fax service running, taking up RAM and making everyone vulnerable to the exploitation of its flaws, in the off chance that someone, somewhere, might someday need it, than to require the person who needs it to install it if they know they need it. So hopefully those law firms and doctors' offices, okay, good, cool. Faxing in Windows? We got that. But it would sure be nice if it was just not on by default.

The principle here is clear. Whether it's an unneeded gateway in your router, or a Windows fax service you will never use, services that are not running cannot hurt you. And without question, shutting them down or uninstalling them will reduce your attack profile. When you hear about systems being security hardened, it's because all of this stuff has been turned off. That's what you do to harden a system; right? You turn that all off, and you turn on the firewall.

So anyway, now we're going to talk about a service that no one can turn off or remove. Believe it or not, Windows is still experiencing remote code execution vulnerabilities in its TCP/IP stack. What year is this, Leo?

Leo: Right.

Steve: As we know, that's the last place you want to have such vulnerabilities because it is by definition exposed to the world.

Leo: And nothing you can do about it. That's what it does.

Steve: One flaw. Yeah. And you're glad that you can talk to other people. One flaw that Microsoft fixed last week was found in the way Windows handles IPv4 source routing. And the other was in the way Windows handles IPv6 packet reassembly. Okay. So what was I just saying about the abuse of unused and unneeded code and services? Here's what Rapid7 has to say about IP source routing. They said: "The host is configured to honor IP source routing options. Source routing is a feature of the IP protocol that allows the sender of a packet to specify which route the packet should take on the way to its destination, and on the way back. Source routing," they write, "was originally designed to be used when a host did not have proper default routes in its routing table. However," they said, "source routing is rarely used for legitimate purposes nowadays. Attackers can abuse source routing to bypass firewalls or to map your network." Oh, yeah. Turn that one on, definitely. Ugh.

Quoting Dustin Childs again, from Trend Micro's ZDI, who was short and to the point about this one. He said: "IPv4 source routing should be disabled by default." He said: "You can also block source routing at firewalls or other perimeter devices." In other words, not today, nor for the past couple of decades, if ever, has anyone had problems with their routing tables. They work just fine, thank you very much. Notice that also in the 15-plus years of this podcast, where we have gone many times deep into the bit levels of the Internet, IP packets, and routing, never have we even touched on IP source routing because it is never used.

Leo: Why would you need it?

Steve: You don't, in the real world.

Leo: It feels like a vestigial feature that was left in almost by accident, like nobody even - is that still in there? Do we still have that in our TCP stack?

Steve: Yet Microsoft just patched a critical flaw...

Leo: I shouldn't laugh.

Steve: ...with a CVSS score of 9.8. So here, before last Tuesday - and hopefully no longer since everybody installed their updates; right? Were someone to send your system a deliberately malformed IP packet containing source routing information,

although the feature has never been used and has never been needed, they could nevertheless take over your system remotely.

Leo: Holy McGillicuddy.

Steve: Yeah. You know, there really ought to be an install time option in Windows. You know how you get that screen of little slider switches?

Leo: Yeah, do you want that? Do you want that?

Steve: How much spying do you want and so forth?

Leo: Yeah.

Steve: There ought to be a big slider switch where you get to choose, and on one side...

Leo: How much legacy code, yeah.

Steve: Yeah. On one side it's labeled "security," and on the other side it's labeled "install a bunch of unneeded and never-used extra features."

Leo: Unbelievable.

Steve: It is. It's just like, again, 2021. So unfortunately it really is a switch. You can either have security, and not have a bunch of unneeded and never used extra features installed. But you can't have both because code is buggy. And I know I sound like I'm harping on Microsoft. But I'm very clear always that anyone can make a mistake. You've never heard me jump on anyone for making a mistake. My entire problem surrounds policies. Organizations should be held to account for their policies. And Microsoft has clearly demonstrated the policy of installing unneeded services by default and supporting long obsolete and unneeded protocols. They don't want someone saying, hey, why aren't you supporting IP source routing? They ought to just say, go get Linux. Really. It's just nuts. I mean, I don't even know if Linux has it turned on by default.

Leo: You know, it might. You know, it really might.

Steve: Yeah, actually, yeah. That's why it came to mind.

Leo: Yeah.

Steve: And that brings us to the other TCP/IP bug, IPv6 packet reassembly. Now, we've talked about this. Packets are never supposed to become fragmented in transit from their sender to their receiver. It's inefficient, and it's assiduously avoided by all inter-

networking equipment. It can occur when a router receives a large packet on one of its interfaces which needs to be sent out, thus routed, out of another interface that has a lower MTU. MTU is Maximum Transmission Unit.

But much as with IP source routing, what made sense at the dawn of inter-networking has since largely become obsolete. Ethernet rules the world. And the Ethernet MTU of 1500 bytes is the standard. There are so-called "Jumbo Frames" for Ethernet of 9000 bytes, with the idea of making Ethernet frames, individual Ethernet frames larger, in order to allow them to carry more payload per frame, thus reduce the per frame, and in this case per packet, addressing overhead. But such Jumbo Frames are only useful within carefully controlled environments, and they generally cause more trouble than they're worth because you just sort of get inexplicable, can't reach this destination errors. And then it's like, ooh, that's right, this is a Jumbo Frame connection.

So packet fragmentation has turned out to be a longstanding annoyance within the Internet protocol. I mean, it's been a source of continuous problems. Once a packet becomes fragmented, it remains fragmented throughout the rest of its or their journey to their destination. And the packet reassembly problem turns out to be a particularly tricky wicket to code reliably. As a result, as I said, it has historically been a source of some significant networking vulnerabilities, which we would hope by the year 2021 have finally been resolved. Not so. Windows TCP/IP still, until last Tuesday, was not managing to get it right, thus creating a newly discovered critical vulnerability with a CVSS score of 9.8, enabling a full unauthenticated remote code execution vulnerability in Windows.

Because fragmentation is no longer supposed to occur, some high-security firewalls simply drop all incoming fragmented packets. It's considered a good practice. Or a router, which is faced with a need to fragment a packet for some reason, may choose not to fragment and forward lots of smaller packets, but rather to return an ICMP - that's Internet Control Message Protocol - message explaining to the sender that the packet it's just received is too large for it to forward, in which case the sender should reduced its transmitted packet size and try again. So in other words, robust mechanisms exist for handling this without the need to ask the receiver of the final collection of fragments to ever reassemble them. Despite the fact that you're never going to actually get fragments, before last Tuesday, if Windows 10 did, whammo, takeover.

Leo: I like the whammo.

Steve: Yes.

Leo: A big thought bubble comes out of your machine, "Whammo."

Steve: Last Tuesday also fixed a critical bug in the Windows Camera Codec Pack, and another in the Windows DNS server. Thankfully, that is one service that Microsoft does not install by default because they understand most people, while they still think most people need to receive a fax, they realize most people don't need to be a DNS server.

Leo: Yes.

Steve: So the good news is that one is turned off. The bad news is it is highly critical remote code execution. If the service is running, if someone does have Windows 10,

before last Tuesday, configured to serve DNS queries, and it receives a maliciously formed DNS query, what is it we say, Leo?

Leo: Whammo.

Steve: Whammo. That's right. A critical flaw was...

Leo: Pow. Zoom. Bang.

Steve: Biff. Biff, pow.

Leo: Biff.

Steve: Also fixed in Windows print spooler, in the .NET Core for Linux, and in Windows Codecs Library. So pretty much just your typical month in the life of Windows 10. They are never going to get it right because their economic model which they have adopted now requires that they keep fussing with it forever. And all experience tells us that, every time you touch it, you risk breaking something that used to work. It doesn't matter how good you are. It doesn't matter how careful you are. Especially with an edifice, I mean, basically that Picture of the Week that we showed at the top is Windows.

Leo: It's Windows.

Steve: You know? That's what it looks like inside. And it's like, ooh, I'm just going to change this one block here, and I'm sure it will be fine, because we need to support, I don't know what, Rocky Road Protocol. So what could possibly go wrong? Well, anyway, this podcast will run out of digits before Microsoft gets Windows 10 working right.

I did mention that, if you are using Adobe's Acrobat or Reader, either 2017, 2020, or the DC versions of either, there is a targeted attack on those occurring in the wild. So that needs to be fixed. If you're using any of those, if you know you're using them, then you do need to get yourself updated. That's important. Adobe has released fixes. You can probably just ask any of those to check themselves for updates. They'll go ask Adobe, and they'll find, oh, yes, there's something new that we need to take care of.

Okay, now, I mentioned that I don't normally talk about Android app security flaws and problems. They have their own podcast here on the TWiT network. And I also suppose it's because it seems like such low-hanging fruit. I mean, our time here is limited every week. But when vulnerabilities have been found and responsibly disclosed in an Android app that has 1.8 billion users worldwide, and when after 90 days of no response from that app's publisher the responsible and well-known disclosing party decides to finally go public with their vulnerability information, then the situation rises to the level that we need to touch on it here.

The Android app in question, as I mentioned, is SHAREit. And I don't follow this stuff that closely, but I recognize that one. And I'd venture that many of our Android-equipped listeners may have a copy. The Google Play Store lists it as SHAREit - Transfer & Share. It is obviously a super popular file exchange app. It was Trend Micro who examined, discovered, and reported the problem to SHAREit's publisher and received no feedback.

Trend Micro wrote: "We discovered several vulnerabilities in the application named SHAREit. The vulnerabilities can be abused to leak a user's sensitive data and execute arbitrary code with SHAREit permissions by using a malicious code or app. They can also potentially lead to Remote Code Execution (RCE). In the past, vulnerabilities that can be used to download and steal files from users' devices have also been associated with the app." Meaning it has a history of problems. They said: "While the app allows file transfer and download of various file types, such as Android Package (APK) files, the vulnerabilities related to these features are most likely unintended flaws."

So SHAREit has over one billion downloads in Google Play and has been named one of the most downloaded applications - it was in the top ten - most downloaded applications in 2019. Google has been informed of these vulnerabilities. So we'll see what action, if any, they take. Trend Micro's posting then delves into the details of the app's API registrations, that is, the things it's doing that are causing these problems, and in detail into its operation, which allow arbitrary files to be downloaded and executed, including downloading and installing any APK.

And just to be clear, it's not the SHAREit app itself that's malicious, but its presence in all 1.8 billion Android devices which are creating security holes large enough to drive Google through. So this creates a huge opportunity for abuse by any other app that wants to get up to some mischief. And now, after Trend Micro's full disclosure, everyone who might want to do that knows about it.

They concluded their detailed write-up by saying: "We reported these vulnerabilities to the vendor, who has not responded yet. We decided to disclose our research three months after reporting this, since many users might be affected by this attack because the attacker can steal sensitive data and do anything with the apps' permissions." They said: "It is also not easily detectable."

So mostly a heads-up to any of our listeners who have downloaded SHAREit. It might be one of those free Android apps that you once upon a time downloaded, used it for a while, and then forgot about it, but it's still installed. If it's still installed, it's a problem. Now would be a good time to say goodbye. Or maybe you use it regularly and cannot live without it. In that case, perhaps be on the lookout for its publisher to finally take the security of their app seriously by producing an update. Maybe this will bring them under some pressure to do that. Or maybe find a hopefully more responsibly written alternative.

Anyway, mostly I just wanted to give everyone a heads-up about SHAREit because yikes. It does sound like it's massively installed, and its presence is creating a set of vulnerabilities for anyone who has it installed that could create problems. Certainly you can imagine it being used in a targeted fashion. If somebody knows that you're using SHAREit and can get you to download something which is not itself malicious, but that abuses the permissions that SHAREit has asked for itself, you could be in trouble.

Leo: Yeah. It's still on the Google Play Store with no notice of anything. And as you said, one billion-plus downloads. That's got to be the majority of Android phones in the world; right? I mean...

Steve: I think so, yeah.

Leo: Yeah. I don't think I've ever installed it. It doesn't look like I ever have, thank goodness.

Steve: Good. You don't want it. And neither do our listeners unless you really need it. And again, it's one of those things. We'll be talking about this theme a little bit, this notion of removing things you are no longer using because, even if they're not still there, they can still represent a problem.

So Microsoft last Thursday - they've got a team they call Detection and Response Team, DART. They post an interesting update about a rapidly growing security threat which we've not yet ever discussed directly. And that's known, these things are known as web shells.

So first we need to define a web shell. It's a fancy name for some malicious code that, well, you definitely don't want running on your server. Or rather, actually, present in your server. A web shell is typically a small bit of malicious code written in whatever typical server-side web development programming language your server supports. So that would be something like ASP, Active Server Pages; PHP, Personal Home Page; JSP, Java Server Pages; or similar. Attackers somehow arrange to implant their script onto a victim's web server to then provide them with long-term persistent remote access and code execution of server functions.

These web shells allow attackers to run commands on servers to steal data or use the server as a launch pad for other activities like credential theft, lateral movement within the infected network, deployment of additional payloads, or hands-on keyboard activity. And most significantly they allow attackers to rather trivially maintain a persistence within an affected organization which is surprising hard to detect. And this gets picked up by our radar because Microsoft has been monitoring this as a rapidly growing trend. They wrote, in their posting Thursday: "One year ago we reported the steady increase in the use of web shells in attacks worldwide."

They said: "The latest Microsoft 365 Defender data, which is their online real-time instrumentation system, shows that this trend not only continued, it accelerated. Every month from August 2020 through January 2021, we registered an average of 140,000 encounters of these threats on servers, almost double the 77,000 monthly average we saw over the same period year over year compared to the previous year. The escalating prevalence of web shells may be attributed to how simple and effective they could be for attackers. As web shells are increasingly more common in attacks, both commodity and targeted," they said, "we continue to monitor and investigate this trend to ensure consumers are protected."

Microsoft blog posting continues with a little bit of riveting case history. They said: "Attackers install web shells on servers by taking advantage of security gaps, typically vulnerabilities in web applications" - and we'll be talking about one next in WordPress - "in Internet-facing servers. These attackers scan the Internet, often using public scanning interfaces like Shodan.io, to locate servers to target. They may use previously fixed vulnerabilities that unfortunately remain unpatched in many servers, but they are also known to quickly take advantage of newly disclosed vulnerabilities.

"For example, last June 30th, F5 Networks released a patch for CVE-2020-5902, a remote code execution vulnerability in Traffic Management User Interface (TMUI). The vulnerability is a directory traversal bug with a CVSS score of [our favorite] 9.8." We talked about it at the time. Microsoft says: "Just four days later, on the Fourth of July, exploit code was added to a Metasploit module. The following day, Microsoft researchers started seeing the exploit being used by attackers to upload a web shell to vulnerable servers. The web shell was used to run common cryptocurrency miners."

Now of course, those were the quaint days, right, when cryptocurrency was being mined. Now we're going to encrypt all your data. We're going to exfiltrate, as you said, Leo, the

embarrassing bits. And then we're going to hold you hostage and make you pay us some bitcoin.

Leo: Yeah. They're so smart, these ransomware folks.

Steve: Although, boy, did you see what's happened to bitcoin, Leo?

Leo: No, don't. Zip. Zip it.

Steve: Yeah. Ouch.

Leo: Actually, it's far worse for you than it is for me. I gave my son my wallet and said, "You can hack it. If you can crack it, it's yours." And then I told him, the good news is by not being able to unlock it, we've watched bitcoin go through the roof. You know?

Steve: And you're right because we would have all sold our coins.

Leo: Oh, yeah.

Steve: Back, like, so long ago. That is the way I feel better about this is that...

Leo: You wouldn't have the 50 now. You wouldn't. You would have sold it when they were a thousand bucks each, and you would have said, "I'm happy."

Steve: I would have thought, woohoo!

Leo: Woohoo, I'm rich. I told Henry, by the time those 7.85 bitcoin are worth more than a million dollars, we'll have quantum computers, and we can just crack the thing. Right?

Steve: So you actually do have your wallet.

Leo: Oh, I have the wallet. I forgot the password. Which is really stupid. But we'll be able to crack it. Not maybe with current computing technology. But by then it'll be really worth something. It'll be worth cracking it. That's my attitude.

Steve: Good. I like that attitude. And it gives Henry something to look forward to.

Leo: Oh, it drives him nuts. It drives him crazy. Weekly he'll email me, say, "Did you think of anymore passwords you might have used?" I actually have it running on my

system at home. So if I ever think of a password I can try it. But so far no luck. You know when I'm going to feel bad? When it goes down to a buck a bitcoin.

Steve: Right.

Leo: Right? Opportunity missed.

Steve: Like when it collapses again.

Leo: Right, right. If it does. Who knows what's going on with that.

Steve: If it does. You know, I think it was you who said that some financial group was saying they see this thing going to 150,000.

Leo: Oh, yeah. And with companies like Tesla buying 1.5 billion of bitcoin, that just props it up further, you know. As long as that continues to happen. Would you like me to take a break?

Steve: That's a great idea. Let me just finish this one piece...

Leo: Oh, sorry, go ahead.

Steve: ...about web shells, and then we'll do that. So once these web shells have been installed on a server, writes Microsoft, "web shells serve as one of the most effective means of persistence in an enterprise." Listen to that again. Once installed on a server, web shells serve as one of the most effective means of persistence in an enterprise. And we'll talk about why in a second. They said: "We frequently see cases where web shells are used solely as a persistence mechanism. Web shells guarantee that a backdoor exists in a compromised network because an attacker leaves a malicious implant after establishing an initial foothold on a server, that implant being the web shell. If left undetected, web shells provide a way for attackers to continue to gather data from and monetize the networks" - I love that - "monetize the networks they have access to." Yes, we're into a little malicious network monetization now.

Okay. So just to make the mechanism clear, a web shell script is just a static text file with an extension like .asp, .php, .jsp, and so on. The typical web server will have a bazillion such files. They're little fragments of the whole web system package which are invoked on demand. And if one additional .asp or .php file were to be added to the hundreds of others that are already there, who would ever know? It's like some file appearing in your Windows System32 directory. Remember the quaint old days when there were four files that ran our operating system? Now no one knows what they all are. So the point is that if one additional PHP file somehow gets added to a contemporary web-based system, it's impossible, virtually, to detect. And it would tend to go completely unnoticed.

Okay. Then at any time or times later, a remote attacker simply queries a specific URL on that company's web server which references the web script that was deposited into a script executable directory, which is to say along with all the other .PHPs or .ASPs, and

the attacker's code comes to life. It's really kind of diabolical. It's trivial on the one hand and doesn't take a rocket scientist. After all, PHP was designed to be dead simple to use as a page scripting authoring language. So you just always hope that it's only your code that your server is running.

What Microsoft points out is that the instant a new vulnerability surfaces, attackers are now scanning the Internet and using the new vulnerability to quickly get a foothold, to quickly drop a web shell onto what might be briefly vulnerable targets before they're patched. And once there, the script can then sit unseen, not, I mean, it's not running, nothing is going to detect it, it's like just some text code sitting in an executable script directory. But it will come to life when invoked.

So as I said, if minutes later the vulnerability that allowed it in is patched to foreclose any future use of that vulnerability, it's too late. Now that server is carrying a malicious script that will jump into service whenever it's invoked through simply receiving a querying URL. Microsoft notes that the challenge of discovery of such implants is hampered by the sheer volume of network traffic, plus the usual noise of constant Internet attacks. This means that targeted Internet traffic aimed at a web server will blend right in, making detection of web shells a lot harder and requiring advanced behavior-based detection that can somehow identify and stop malicious activities that are essentially hiding in plain sight.

So anyway, I just sort of wanted to put that on everyone's radar. It is a growing trend. It has doubled in a year. The concept of empowering web servers to interpret textual scripts as code is incredibly empowering. This method of running server-side code has pretty much taken over the world. It's the way it's being done now. And the downside is that, well, text files on the server are being interpreted as code. In other words, it's both what we want and what we don't want.

Leo: Right.

Steve: And it really is a growing problem.

Leo: I mean, somebody put a PHP script ages ago on one of my servers.

Steve: Yup.

Leo: And because I had that folder open for FTP, people could execute it. And that's a real flaw. You really shouldn't have folders that just anything in there can be executed. That's just - but that's one of the biggest flaws with PHP, in my opinion.

Steve: And in fact we've talked about several of the recent problems with WordPress, and we're about to talk about another one, is that you can do that .png.php, or the other way around, you know, double extensions, and it'll be seen by the system as an image file. But when the PHP interpreter is invoked with it, it'll go, oh, look, I've got something to do. It's a script, and it'll run it. So wow.

Leo: Wow is wow.

Steve: Or whammo.

Leo: Whammo. Pow. Zoom.

Steve: So I mentioned we have once again yet another WordPress mess. And I hope that, if nothing else, mentioning these every week, I mean, they are of great concern to the security community. I hope that mentioning them every week is having the effect of increasing our listeners' security by virtue of doing something to keep themselves from being vulnerable.

I mentioned WordFence last week. It sounded like a useful service. They have both a free and not. And so that's something to think about. Or just really resisting the installation of plugins. I don't know whether we've seen an instance where WordPress itself has problems any longer. I'll knock on wood here somewhere. But in this case, this week's problem is known as the Responsive Menu WordPress plugin, which is exposing more than 100,000 WordPress sites to multiple critical and some high-severity vulnerabilities. This Responsive Menu is a WordPress plugin designed to help admins create W3C, World Wide Web Consortium, compliant and mobile-ready site menus.

The group I mentioned, WordFence, the threat intelligence folks, found three vulnerabilities that can be exploited by attackers with only basic user permissions to upload arbitrary files and remotely execute arbitrary code. And what was I saying about the rise of web shells. This is exactly kind of thing you would install would be a web shell. And a WordPress site would automatically be having a PHP interpreter running, so off you go.

The first of the three flaws enables authenticated attackers to upload arbitrary files which eventually allows them to achieve remote code execution. So there you need some authentication. That was one problem. But the other two vulnerabilities allow an attacker to forge requests to modify plugin settings of the plugin, which in turn allows them to upload arbitrary files, allowing for remote code execution.

So to leverage these vulnerabilities, the attackers logged in as subscribers, or another low-level user, upload menu themes archived as zip files and containing malicious PHP files. Once the archive is extracted for installation, the attacker could then access the files via the site frontend to remotely execute their malicious code, which ultimately can lead to a full site takeover. So it sounds like a more involved kind of complex dance to go through to get remote code execution. But if the target has sufficient value, and if for example a persistent web shell can then be dropped into a briefly vulnerable server, it might well be worth the effort.

So once again, unfortunately, a WordPress plugin is presenting an opportunity. The problem is, we've talked about this, these are written in PHP. We know absolutely, I'm sure, that all the plugin authors had the best of intentions. But they're not professional security people. So unless you have been exposed to directory traversal, the consequences of directory traversal attacks, or double file extension compromises, I mean, again, we keep seeing the same things happening over and over because people who aren't security professionals are writing code that has to be secure against threats that have become well known. And so the same mistakes keep being made over and over and over.

So I don't know how we get ourselves out of this. The only solution with WordPress is, if you've got to run your own, give it its own sandbox. Give it its own virtual machine. Restrict what it can do. While I was running one, I gave it its own physical server and established a hardware firewall between it and the rest of GRC because I could not risk

something getting loose in there. And of course I was very careful about which plugins I installed. I added a couple, but minimal, which is...

Leo: I think plugins are really, I mean, at least historically it's all been plugins; right? The core code I think is pretty good.

Steve: I think that's exactly right, yeah. So a brief update on SpinRite. Last week I talked about, and I made you chuckle, Leo, the anxiety-provoking Discovering System's Mass Storage Devices screen, where historically, if SpinRite was going to get tripped up, if it was going to have a problem, that was where it would happen, while it was scanning the system's drives. What was often happening was that since SpinRite was having to always use the BIOS, which is what I'm working hard to get completely away from, it would make a BIOS call innocently, asking something about the drive, and the drive would have a problem, and the BIOS would just hang. And so it's like, hello, okay.

Anyway, so I decided, as I mentioned last week, to show SpinRite's work in progress as it's doing the work. And so I decided to post SpinRite's process and progress as it went along onto a screen. And you've got it up on the show notes. That's cool. At this time last week, that was just a concept. Today it exists. I finished...

Leo: I'm impressed. Holy cow. That's fast. Wow.

Steve: I finished that work and posted a sample image, which you've got onscreen, of the new SpinRite screen to my work progress tracking thread of my blog on the GRC forums. And also to GRC's spinrite.dev newsgroup. But then, because actually seeing it work is really fun, I made a video.

Leo: Oh, fun.

Steve: The link is there below the screen. It is GRC's shortcut of the week, so grc.sc/806. That will take you to a 25-second video of this process which is running on my mega-death development machine which, you know...

Leo: Has a few drives.

Steve: ...has a bunch of controllers. Yes, a bunch of drives, a bunch of controllers, all kinds of craziness.

Leo: That is so cool, Steve. That is really cool that you did that. Can I ask you a couple questions about your development?

Steve: Yeah. Yeah, yeah.

Leo: So as we all know, Steve is - this is a DOS program because you can't run it while Windows is running, for obvious reasons. So it's a, what do you call it, TUI; right? It's a text-based user interface. Do you by hand do the ASCII code for line,

bar, cross, all of that? Or do you have some sort of generator that you use to create that? You must have a macro or something; right?

Steve: Sort of both. There was a really neat text, kind of like text editor, written a long time ago, called Dan Bricklin's Demo.

Leo: Oh, yeah. Oh, yeah. We know Dan, sure.

Steve: Of course we all know Dan Bricklin. He was the originator of one of the two, along with Bob Franklin of VisiCalc.

Leo: Frankston, yeah.

Steve: Yeah. And so DBD, Dan Bricklin Demo, it was a DOS program that allowed you to sort of easily, like, edit screens, type text into windows and things.

Leo: Right, right.

Steve: So I prototyped SpinRite's user interface in that. But it was its own world. So then I took from the prototype, what I did build was a full text windowing system. So SpinRite is built on top of a textual windowing system where I'm able to, for example, push the screen, pop a window on top, and then point it to a description of what's in the window, and then for example things like the menu bar will automatically move up and down as the user presses up and down arrow. So I'm coding a lot by hand, but I did sort of build sort of a windowing OS, a little windowing OS, and then I describe all of the different windows in a data structure that is efficient for that. And it compresses down to just nothing because it's all text.

Leo: So clever. So that speeds all of this stuff up, all of these things that you have to do.

Steve: Yes.

Leo: Yeah.

Steve: Yeah. It means that I'm not having to write everything by hand. I've got a bunch of tools. And actually I'm so rusty with it that it's been good for me to build a new screen in this system because it's like, oh, that's right, I can do da da da da da. So I have, I'm like rediscovering all the subroutines that I wrote two decades ago in order to bring this screen to life. But anyway, grc.sc/806, you know, today's podcast number.

Leo: Nice. It's a cute little interface. You can see the difference, though. If you had to sit for 25 seconds even, looking at a blank screen, it gets a little nerve-wracking. But if you see it doing stuff, it's like, oh, yeah, yeah, it's enumerating devices. Go ahead, yeah.

Steve: Yeah, yeah. And it's cool to actually get to see it, yeah.

Leo: It's very cool, yeah.

Steve: So the other thing that happened is - oh. So I forgot to mention that I got that done at the end of last week, like I think it was Friday or maybe like - I think Friday night I released that, that is, the first piece of the new SpinRite code to the spinrite.dev newsgroup, and they all began pounding on it. That is, they ran that all on their own systems, and on maybe a hundred different systems. Maybe 75. I didn't count them.

I ran across one problem that came back from a guy. Chris in Germany has an AMD gigabyte motherboard with a Phenom processor. Everything's great if he boots from floppy. But if he boots from a USB, we have a hang problem. It was happening with ReadSpeed, the predecessor code. I finally - just the problem kind of went away. And I was like, okay, well, I'm glad it's not happening any longer. Anyway, it's back. So yesterday or the day before I purchased - I found that motherboard on eBay. And so I've got one of them coming. Because really the only - this is just some obscure, I don't even know what the problem is. But there's that.

Then a couple people had a problem with that last phase you can see in the video where all the hardware is first enumerated, and all of the drives are found, the controllers and their drives. And then it goes back in and patches in the BIOS numbers. Well, that's important because drives that SpinRite doesn't recognize through hardware, it will still allow to be accessed the way it always has been through the BIOS. It's not ideal, but at least you can still run it. And that's what SpinRite has always allowed you to do.

But now the problem is, if it doesn't - if it finds the hardware, but isn't able to associate it with its BIOS number, then you'll get two listings for one drive. SpinRite won't know that there are actually two references to the same drive. And so I call that BIOS association. And it's tricky because...

Leo: So that's what's happening with this NVME here. It's identifying both as NVME and SCSI.

Steve: Exactly.

Leo: Yeah.

Steve: Exactly. So it's probably the NVME that is actually appearing as that last BIOS drive.

Leo: Right. Oh, I see, 88, yeah, yeah, yeah.

Steve: And so that's the way you will access it. But imagine if any of those lit up in white drives if they didn't have their BIOS numbers next to them. Then it would show the BIOS item because it would think that it's going to have to access it that way. But also it would show the physical drive. Well, so one of the ways I got around that was by hashing the boot sector of every physical drive and then accessing the boot sector through the BIOS,

hashing that and looking for a match. It works unless two drives have identical boot sectors. And in one case, one of the systems had two 1TB drives and two 500GB drives. Since they were identical size, and they were set up by the same guy at the same time, they had identical boot sectors. So they had the same hash, and SpinRite was unable to disambiguate them. I have a solution for that also that involves deliberately leaving the drives in an error condition and then reading from the BIOS, which will clear the error condition in the hardware. And that mostly works. But in this case it isn't.

So anyway, that's where I am this evening. I have an idea now about how to make that fallback process more robust. And the moment I get that done, then - and of course I'll get this USB boot gigabyte AMD Phenom processor-based motherboard in, and I'll stick it in a chassis and see if I can recreate Chris's problem. Hopefully I'll be able to make it happen here, in which case I'll just figure out what's wrong, and I will dance for a while because when Chris posted a couple days ago that he was having this problem, it's like, oh, I thought that one was gone.

Leo: Damn.

Steve: I didn't really earn it. It just kind of disappeared. And so it was like, okay, well, I guess I'm going to have to really figure out what's wrong.

Okay. So on the screen, in the show notes, is what happened when I put one of my email addresses into this test site: "Oh no! Your email address has been leaked." Now, the good news is I ran through first all of the email addresses I think I have ever used at GRC. None of them were there. Then I put my Gmail address in, and that's the one that came up.

Okay. So I'm getting ahead of myself a little bit. But I did want to refer our listeners to this leak tester. I gave it its own GRC shortcut, grc.sc/comb. That will bounce you to the personal data leak check at CyberNews.com, who are the folks that wrote about this monster database leak and immediately brought up a site check that allowed people to perform a database query against their email address.

So the CyberNews guys wrote: "It's being called the biggest breach of all time and the mother of all breaches." Of course, the mother of all breaches would be MOAB, which in my opinion that would be a far cooler name, although technically MOAB has already been taken by the Mother Of All Bombs, which is you may remember that low air dispersion, just this insanely powerful - or maybe it was a bunker buster that was a MOAB. I don't remember.

But anyway, we haven't checked in on the status, as I mentioned, of the underground dark web's aggregation and maintenance of massive login credential lists in quite some time. So I thought that this recent appearance of the latest and by any measure greatest such compilation would be a good opportunity to see where all of that currently stands. COMB stands for Compilation of Many Breaches. It contains more than 3.2 billion, with a "b," unique pairs - unique pairs, so no reps - cleartext emails with their matching passwords.

I've embedded a screenshot of the hacker's announcement from exactly two weeks ago, February 2nd, in the show notes. And the fine print, he has four bullet points which I had to squint to read, so I've also reproduced them in the show notes. Four bullet points. He wrote: "Most of the contents are almost all publicly available." He said: "Compilation of many breaches is built on the breach compilation which was 1.4 billion entries." And he says: "And more new leaks added, for example, Collection #1-#5, and many more." So these are things, you know, we're beginning to see a little bit of the parlance of the dark

web because, like, the people who are typically reading this are like, oh, yeah, yeah, of course, the Collection 1-5. But it's not something we talk about.

He said, second bullet point: "All data is in an alphabetical order in a tree-like structure." He says: "As with the Breach Compilation, a query script is included." And he says: "All data is archived and added to an encrypted and password-protected container. Password is below." Although you can see in the screenshot it says "Hidden content for authorized users." And whoever was viewing this was apparently not authorized. Or maybe they didn't take a screenshot of that.

In any event, CyberNews, the people who posted about this, claim that they were the first leak test database to include the COMB data. And since COMB was first released, they said that nearly one million users had used their personal data leak checker to see if their data was included within this biggest breach compilation of all time. And as I said, I checked all of my various GRC email addresses that I've had through the years, and they all were clean. But when I checked my Gmail account, which I sort of use as just a generic spam filtering bit bucket - like if I don't really want to give someone my GRC email, I'll give them my Gmail - I received the screen above, which informed me that my email address had appeared in a breach.

Fortunately, I do change that, my main Google password, from time to time. It is a long, high-entropy, total gibberish password. And I have one-time password multifactor authentication enabled, less for Gmail, which I don't really care about, than for my Google account itself, which I do. But I never use that Gmail account for anything that, for example, might need password recovery. So nothing sensitive is even passing through that account. And although it's unusual behavior with Gmail, where they encourage you to leave everything there, I routinely delete everything from my inbox and trash.

Leo: Do they have the passwords for this account, or just the email address?

Steve: Well, I don't know. They say they have an email address and some password. So I don't know what it is that they have. You can't see that. I don't know, I mean, so...

Leo: I mean, email addresses are not hard to come by.

Steve: No, they're not.

Leo: And by themselves it's meaningless that somebody has it except you might get more spam.

Steve: Correct. And I may have used - and I do, as I was just saying - I use that for, like, throwaway accounts. So, for example, I might have used my Gmail account on some other site when I was creating an identity there. So I used that and some password. That site leaked its information. So it's not my Gmail account password that was associated with the Gmail email, it was some random password at the site. So what we hope, however, is that even back then, and I don't know when "then" was, but there was a day, Leo, when you were known by your monkey password.

Leo: It could have been that. It could be that.

Steve: So we've all gotten better since those early innocent days where we use - although a lot of people still use "password123" as their password, unfortunately.

Leo: How is this different from HaveIBeenPwned? Is this a bigger database?

Steve: Yes. It is a bigger database. I went over to see whether HaveIBeenPwned had updated yet to include this. It doesn't look like they have. They did mention the Collection 1.

Leo: They have the Anti Public Combo List.

Steve: Yes. Yes. And actually that's here somewhere. I did run across Anti Public and Exploit.in. Those are also both incorporated here.

Leo: And they have Collection 1, yeah, yeah.

Steve: Yeah. So it looks like, in general, these guys are trying to be the one-stop shop for everything.

Leo: I'd say that's the big difference between this and HaveIBeenPwned because, in the case of HaveIBeenPwned, they tried to tie it to the breach that is associated with that email address.

Steve: Right. Right, right, right.

Leo: So you maybe can get some guess as to what password has been breached.

Steve: Yeah. So they're seeing things in this COMB database - Netflix, Gmail, Hotmail, and Yahoo. You know, email address domains. They did some counting, and they found approximately 200 million unique Gmail addresses and 450 million Yahoo email addresses. And again, those are not necessarily the login for those services. They could just be other sites that were breached, and people used those addresses at those sites. Microsoft confirmed Outlook had been breached back in 2019, or I guess between January and March of 2019, so hackers were able to access some of the Outlook.com, Hotmail, and MSN email accounts. We know that Yahoo suffered what was probably the biggest breach of ever. The breach occurred in 2014, but remember they didn't tell anybody about it for two years, until 2016. In that case, bad guys apparently had access to all three billion of Yahoo's user accounts which were impacted.

So, and of course we know that these days many exfiltrated passwords are now hashed. Older ones weren't, back in the really early days. And then even in the early days of password hashing, those that were, were not well hashed - few if any iterations, maybe no salt added, or maybe a static salt for the site. It wasn't until later that we understood, okay, you've got to do per-account salt, and you've got to iterate the hash a whole bunch to make it really difficult to reverse. So consequently some of these raw databases contain email in plaintext and passwords as hashes. However, when the data from the

previous breach compilation was analyzed, it was found that 14% of exposed username/password pairs had not previously been decrypted by the community and were now available in cleartext. So it looks like the project of reversing these password hashes is an ongoing campaign, and there is progress being made on that front.

And then, anytime you have a database like this, there's the question of age; right? Because these things do tend to just continue to grow. Hackers always want to say, oh, yeah, mine is bigger than yours. So to say 3.2 billion email addresses and passwords, the question is, yeah, but how many of them are still good for anything? As we know, the older the lists are, the less useful they're going to be. However, when the identity intelligence company 4iQ discovered that breach compilation database in 2017 - so, what, four years ago - they tested a small subset of the passwords for verification, and most of the tested passwords worked, they said. So the threat analyst stated at 4iQ that they found this 41GB dump on December 5th of 2017, with the latest data updated on November 29th of that same year of 2017. And we talked about this at the time, if anybody says, oh, yeah, that kind of sounds familiar. It did for me, too.

The 4iQ researchers note that the leak was not just a list, but rather an interactive database that allowed for fast one-second response searches and new breach imports. Given the fact that people reuse passwords across their email, social media, ecommerce, banking, and work accounts, hackers of course can automate account hijacking and account takeover, or at least try.

Okay. So what are our takeaways from the news of there is now a 3.2 billion item list? For one thing, old, forgotten, and unused accounts often remain active; right? I mean, unless you go and kill it, it's probably still there somewhere. Something somewhere that you might not have used for years could still be active, probably is. And it could allow a bad guy to gain a foothold, if you've just sort of abandoned it, but it's still there, if it's still possible to access it using old credentials. So one thing we could all do is ask ourselves about any accounts that we haven't used for a long time, perhaps since before we began using a password manager, and so were much less cautious about our choice of password. And monkey.

Leo: 123. 123. Don't forget the 123, yeah.

Steve: Exactly. And then of course shutting down any long-unused accounts will meaningfully reduce our online target profile. It's worth considering. And assuming that our password manager allows us to see a list of everything it's holding for us, we might take a moment to scan through it, just to see whether we really still need all of that. We tend to accumulate a growing collection of one-off sites, where keeping an account might really no longer serve any useful purpose.

So again, reducing our profile just makes sense. It's not always easy to delete our account, but increasingly it is possible. So looking through your password manager and thinking, okay, I am just - there's no way. Or maybe you create an account for something you thought you were going to be using that didn't happen, like how many apps have we downloaded over the years that qualify that way. So again, reducing your target profile makes sense. And if nothing else, you could have your password manager flip those passwords into the clear and just make sure that they have been turned into a long string of gibberish. That is certainly worth doing.

And lastly, as these massive lists age, they automatically lose their value as people change their email and hopefully their passwords. But we can all help those lists age out even faster by deliberately obsoleting the data that they contain about who we were, which is us. We can't change what the lists say about our past. That's in the lists. But we

can arrange to have them say nothing about who we are today. And just sort of some measures to stay safe on the Internet.

Leo: A little hygiene.

Steve: Yup.

Leo: Yeah. And a good password manager, LastPass, for instance, has a security challenge you can go through that I think, I'm pretty sure they also have access to a database, maybe HaveIBeenPwned. And your browsers now do that, too. There's a Chrome extension will say, we've seen this login somewhere else. Watch out.

Steve: And in fact I think it's built into Chrome itself because...

Leo: Might be now, yeah, yeah.

Steve: Yeah, because there was something I was using, I think I mentioned it the other day, where it came up and said, oh, look, this password has been used in a breach. And I went to look, and I go, oh, yeah, that was on purpose, actually. So, yeah.

Leo: I breached myself.

Steve: I knew what that was.

Leo: Steve Gibson. He's the only guy who breaches himself. There he is, right there. He is the...

Steve: I'll have to be careful about that.

Leo: ...man in charge at GRC.com.

Steve: I'm probably not the only guy.

Leo: No, no, no, no. He's a self-breacher.

Steve: So next week. I did want to tease next week. Unless something even more tantalizing comes up, we're going to examine - oh, my goodness - a brilliantly clever new concept supply chain attack that was fortunately pulled off by a good guy security researcher because, if it hadn't been, all hell would have broken loose. Thirty-five major companies were breached in this experiment. So we've got a really fun topic for next week.

Leo: Yeah, and the Supermicro hack is back in the news. Bloomberg is putting more information out. Supply chain, thanks to SolarWinds, what you're about to talk about, and quite rightly so. Supply chain attacks are very much in people's minds these days. That's a real hazard.

Steve: And remember that we never got definitive proof about Bloomberg; but at the time I said, okay, maybe it didn't happen. But, boy, could it have.

Leo: Yeah. And, you know, we've done, the NSA has done this with Cisco routers and stuff. So it happens. We know it happens. And so it's something to be very much aware of. Good, I look forward to that. That's next week.

You can get today's episode and all 806 Security Now! episodes at Steve's site. He's got some unusual versions. He specializes in collecting the weird versions, like the 16Kb audio versions for those who don't have a lot of bandwidth. He has 64Kb audio, as well. And he has transcripts. Elaine Farris actually writes down these immortal words so that you can read along as you listen, or search - I think that's the most useful reason to have those transcripts - for the thing you're looking for and jump right to that part of the episode. All of that's at GRC.com.

While you're there, do pick up a copy of SpinRite, SpinRite 6, the world's finest hard drive maintenance and recovery utility now getting more useful, more interesting all the time as we head towards 6.1. You'll get a free copy of 6.1 if you buy 6 today. You'll also be able to participate in the development of 6.1. These little side alleys that you're going down are fascinating, and I think I'm really looking forward to seeing 6.1. It's going to be a lot of fun. Coming soon. Steve has lots of free stuff, as well, including ShieldsUP!, which is a great way to test your router. I don't install a new router without running ShieldsUP!. And periodically I'll check it, as well. Lots of other fun stuff. GRC.com.

Steve: You know, Leo, that DNS Benchmark?

Leo: That's a really good one.

Steve: It's 3,000 downloads a day.

Leo: Yeah. No, I'm not surprised. It's a really good way to find a better DNS server than your ISP's. But also if you're considering something like Cloudflare's Quad1 or Quad9's or NextDNS or some of these other DNS servers, it's really a good idea to test it first with the DNS Benchmark to see what your local results will be. You don't want to choose a slow DNS server. That's not a good thing.

We also have copies of the show at our website, TWiT.tv/sn. We pretty much stick with 64Kb audio, and we do do a video version, if you want to watch, see the blinking lights here, the blinking lights there, all the blinking lights everywhere. That's all at TWiT.tv/sn for Security Now!. There's a YouTube channel for Security Now!, dedicated just to that show. You can share that with your friends. It's a good way to share the show with your friends, just a link to the YouTube video, because you can even pick the part of the video, the time code and everything, and jump right to that.

We also of course are on every podcast program and directory. If you use a podcast program, probably that's the easiest thing to do is subscribe in that, either to the audio or the video, to get it automatically. Steve is on Twitter. @SGgrc is his handle. You can DM him there. His DMs are open, @SGgrc, if you've got a tip or a question or a comment. You can also do it on his website, GRC.com/feedback. Next week, Supply Chain Attacks.

Steve: Ooh, a good one. Very, very clever new way.

Leo: Thank you, Steve. Have a great week. We'll see you next week on Security Now!.

Steve: Okay, buddy. Bye.