SECURITY NOW!

Transcript of Episode #805

SCADA Scandal

Description: This week we begin with a collection of interesting and engaging news surrounding Google's Chrome browser. We look at a high-profile Windows Defender misfire, and at new WordPress plugin nightmares. We check in on the world of DDoS attacks and cover the meaning of three new critical vulnerabilities in SolarWinds software. We have a bit of closing-the-loop feedback from our listeners, an update on my work toward the next SpinRite, and then we look at a near-miss disaster in a poorly designed industrial control system.

High quality (64 kbps) mp3 audio file URL: http://media.GRC.com/sn/SN-805.mp3
Quarter size (16 kbps) mp3 audio file URL: http://media.GRC.com/sn/sn-805-lq.mp3

SHOW TEASE: It's time for Security Now!. Steve Gibson is here. Lots to talk about. A trio of Chrome issues. There's a new DDoS attack, a new amplification attack using the Plex Media Server. There are three new SolarWinds vulnerabilities you need to know about. And then we'll wrap things up with a closer look at the attack at the small-town Florida water facility. It was a SCADA attack. The details, coming up next on Security Now!.

Leo Laporte: This is Security Now! with Steve Gibson, Episode 805, recorded Tuesday, February 9th, 2021: SCADA Scandal.

It's time for Security Now!, the show where we cover your security online, your privacy, the way things work, protecting yourself. This is the guy right here, Steve Gibson from the Gibson Research Corporation. Hello, Steve.

Steve Gibson: Yo, Leo.

Leo: Good to see you.

Steve: Great to be with you again, as always.

Leo: I see you've got the soldering iron out. Are you making something?

Steve: Yes, doing a little construction work, that's true, yeah.

Leo: Someday that Speak & Spell will work again.

Steve: And there are always things more interesting to than cleaning up around here, so...

Leo: Absolutely.

Steve: So that never kind of - it's sort of a constant entropy battle.

Leo: It's life, yeah.

Steve: Which I'm generally losing. You know, at some point with one's car, it stops getting more dirty because as new dirt arrives, old dirt falls off. So you reach a homeostasis.

Leo: That's so Steve. No, that's good observation, Steve.

Steve: I guess it's actually equilibrium as opposed to homeostasis.

Leo: Your car reaches filth equilibrium.

Steve: A dirt equilibrium where it's just like, okay. And then when you get used to that it's like, okay, looks fine to you. Neighbors are like, who is this guy?

Leo: Well, as long as they don't start writing stuff in the dirt, you're all right.

Steve: I actually used to be known by my local car wash people. They were really neat, and they took good care of me. I haven't seen them in a year because...

Leo: Yeah, me neither.

Steve: ...I don't want them in my car.

Leo: No, exactly. You know, there's things you can do without. It's not the end of the world if you don't get a car wash.

Steve: But Leo, it is so cool when a tank of gas lasts six months because that's like a whole new thing. You don't have gas, I realize, any longer. But many of us still do.

Leo: I remember making a note back in November when I got the Audi filled up. I thought, this is the last time I'm going to a gas station. That was kind of a wild thought, yeah.

Steve: So we got some interesting stuff for our 805th episode of February 9th. This is - I wanted to call it Super Tuesday, but it's Patch Tuesday. So we'll be talking about what happened today next Tuesday. But we've got a collection of interesting and some engaging news surrounding Google's Chrome browser. I was going to use the word "trifecta," but that apparently only applies to betting. So instead it's just a trio. We're also going to look at a high-profile Windows Defender misfire.

We have a couple new WordPress plugin nightmares, and maybe a suggestion for those of our listeners who have WordPress and can't get away from it for whatever reason. We're also going to check in on the world of DDoS attacks inspired by a particular media server. Plex has a problem. It's turned out that the bad guys have found, I don't remember now how many thousands, but many tens of thousands of Plex server protocols online, and have figured how to bounce traffic off of it to add it to their DDoS attacks. But it also gives us a chance to sort of check in on where the industry is in DDoS. And the numbers are sort of daunting.

We've also got three new critical vulnerabilities found in SolarWinds software. And the way they've been found and what it means about the things that haven't been found in other software I think is significant. So we're going to talk about that. We've got some closing-the-loop feedback from our listeners. I'm going to give a quick update to my past week's work with SpinRite, heading toward the next one. And then I just want to finish with the issue that titled this podcast "SCADA Scandal." S-C-A-D-A is of course the acronym or the abbreviation for the industrial control system technology that runs pretty much everything. There was a near miss that occurred in a little town in Florida last Friday.

Leo: Was that a SCADA system?

Steve: Yeah.

Leo: I didn't know that. Ah.

Steve: Yeah. And, boy, I hope it serves as a wakeup call because there are, like, several things wrong with this story. I mean, like with the facts which are true about the story, the things that should not be the case. So I think an interesting discussion about that. And of course we have a fun Picture of the Week for our - I was going to say for our listeners, well, not so much. Mostly for our viewers.

Leo: Yeah. We'll subtitle this one "Why You Don't Want More Lye in Your Water."

Steve: Yeah.

Leo: Wow. I didn't - oh, good. I can't wait. I'm fascinated to hear this story. On we go with your Illustration of the Week, your Picture of the Week. I'm going to show its original source because I recognized it immediately, the great talents of Randall Munroe at xkcd. I love his stuff. And this one's right on.

Steve: Anyway, so what we have is an industrial-scale Venn diagram with the caption, "I have a hard time keeping track of which contacts use which chat systems." And so we've got a large Venn circle of email, lots of people in there, with a large overlap between those users who are SMS because of course that's just simple messaging service. Then we've got two little people all by themselves using AIM; remember that? And also ICQ. There's only one person in that circle.

Leo: And that's still around, I found out the other day. I'm stunned.

Steve: No kidding.

Leo: Yeah.

Steve: Wow. I had a really low numbered ICQ account.

Leo: I still do.

Steve: Because by some weird account - yeah. Anyway, we've got Skype, Slack, iMessage, Facebook Messenger, Instagram DMs, Twitter DMs, Zephyr, BBMs. I guess that's what - BBM. That was Blackberry.

Leo: Blackberry Messenger, yeah.

Steve: WhatsApp, IRC, Signal, Hangouts, Snapchat, WeChat...

Leo: I was on every one of these, I think. Oh, I was never on WeChat.

Steve: God. And so, Leo, actually the Venn diagram could not be stretched to include you in all of these circles because you're in too many of them.

Leo: I'm in all of them.

Steve: It's got, like, Zephyr. Oh, also we have "the chat tab in an old Google Doc" was one of them.

Leo: We use those every day on the shows.

Steve: There's two people there. And then we have someone writing on the wall of a bathroom, so there's that chat mode. Apache Request Log, a Telegram, I mean, anyway, so a lot of fun with this. And just but it's true, you know, the problem with being so heterogeneous is it's annoying when you've got people spread all over the place. I'm not sure why - oh, I was going to say I'm not sure why Rasmus and I were not using iMessage because I think he's an iPhone person, because we were using it when we were synchronizing in Sweden during our European trip to talk about SQRL. But it's because Signal has a desktop client, and one of my biggest annoyances with Apple is that they're hostile to anything that's not iOS or Mac. So Signal was something that we were able to use.

But there's, again, a sort of an example of sometimes you're pulled to a different application because you need to do something that the one you're using doesn't support for whatever reason. Anyway, just a fun observation that - and we were just talking about, as people are abandoning WhatsApp because they're concerned about the change of its privacy terms, they were moving to Signal. And we talked about last week there was some guy, or I guess it was the week before, some guy saying, yeah, I switched to Signal. And he's like looking around, there's nobody here. I don't know anybody who also uses Signal. So, yeah.

Okay. There were three things that all happened last week affecting Chrome. First off, at the end of the week, actually I think pretty much everything was Thursday. Saying only that the extension contains malware, Google unceremoniously removed an extremely popular Chrome extension known as The Great Suspender. And if nothing else it deserves a note for its name. They pulled it out of the Chrome Web Store and caused two million plus of the Chrome browsers, where it had been previously installed, to immediately remove it from themselves.

The Great Suspender was very popular with those wishing to run Chrome in memory-lean environments. Chrome tabs are known to consume a great deal of RAM. And The Great Suspender's claim to fame was that it could suspend tabs and release their memory back to the system while kind of keeping the tab there as a placeholder so that it wouldn't cost the user any RAM, but they would sort of have the comfort of there being a tab there that they knew they could go click on. It would have to reload the whole thing. But still, that was what The Great Suspender did.

Last November, things began to turn sour in Great Suspender land. It was an open source app that was hosted on GitHub. And a November 3rd posting on GitHub which was the beginning of a thread which garnered 449 comments started off with the TL;DR saying - so this is November 3rd, so we're turning the clock back from what just happened on Thursday. The person who began the thread said: "The old maintainer appears to have sold the extension to parties unknown, who have malicious intent to exploit the users of this extension in advertising fraud, tracking, and more."

He said: "In v7.1.8 of the extension, published to the web store but not to GitHub, arbitrary code was executed from a remote server, which appeared to be used to commit a variety of tracking and fraud actions. After Microsoft removed it from Edge for malware, v7.1.9 was created without this code. That has been the code running since November, and it does not appear to load the compromised script. The malicious maintainer remains in control, however, and can introduce an update at any time."

Leo: That's the problem. That's the problem.

Steve: Yes.

Leo: The updates aren't screened.

Steve: Correct. Well, there's just too much, as we know, Leo. I mean, if you look at what's on the Play Store now, it's completely out of control. On the other hand, it's wide open, and lots of opportunities there.

Okay. So the more detailed discussion that followed I thought was interesting. And I've sort of excerpted some of the best bits of that. The original developer was a person, @deanoemcke, I guess, it's D-E-A-N-O-E-M-C-K-E. So I'll just got with @deanoemcke. So it was said that he chose to step back from the extension last June of 2020. As a replacement maintainer, he chose an unknown entity who controls the single-purpose @greatsuspender GitHub account.

They wrote: "Much was suspicious about this change, including mention of payment for an open-source extension, and complete lack of information on the new maintainer's identity. However, as the new maintainer did nothing for several months, it was originally believed that there was a failed transfer. In October 2020, the maintainer updated the Chrome Store package. The update raised red flags for some users because the changelog was not modified, and there was no tag created in GitHub.

"On investigation, it appeared that the extension was now connecting to various third-party servers, and executing code from them. This led a few users to panic. However, on closer investigation, it appeared that the third-party servers were part of an alternative to Google Analytics, and the changes shipped along with a new, though unexplained, tracking deactivation. It appeared that deactivation worked. We would later discover that this was wrong. The discussion continued, however, because the new update also requested additional permissions, including the ability to manipulate all web requests."

As we know, we've talked about this in the past about Chrome. This lets the extension do whatever it pleases - inserting ads, blocking sites, forcible redirects, whatever. This change was supposedly in order to enable new screenshot functionality, but that was unclear and probably shouldn't be needed.

"Furthermore, the Web Store extension," they wrote, "has diverged from its GitHub source. A minor change in the manifest was now being shipped on the Chrome Web Store, which was not included in GitHub. This is a major concern, though again it has a possible innocent explanation. While some think it is illegal, given the license on the code, this may not be a GPL violation. Because the minified script is not part of the extension, the license does not apply to it. Because of Web Store rules, the extension itself can be unpacked and inspected in full human-readable form, likely satisfying the copyleft restrictions.

"As a final red flag," they wrote, "no part of the Web Store posting has been updated to account for this. @deanoemcke remains listed as the maintainer, and the privacy policy makes no mention of the new tracking or maintainer. It has been several months since the transfer, but almost nothing reflects that change. @deanoemcke did respond to the thread after a significant delay. He confirmed much of what is above, including that the secret changes are limited to analytics and are disabled by the flag. However, he hasn't yet clarified what his relationship or basis of trust with the new maintainer is, nor has he explained why the initial post mentions a 'purchase.'

"On November 6th, someone named @lucasdf discovered a smoking gun that the new maintainer is malicious. Although Open Web Analytics is legitimate software, it does not provide the files executed by the extension. Those are hosted on the unrelated site http://owebanalytics.com, which turns out to be immensely suspicious. That site" - it's written here in GitHub - "was created at the same time as the update and is clearly designed to appear innocent, being hosted on a public web host and being given a seemingly innocent homepage from the CentOS project." Yes, Leo.

"However, the site contains no real information other than the tracking scripts, appears to have been purchased with bitcoin, and is only found in the context of this extension. Most importantly, the minified JavaScript differs significantly from that distributed by the actual OWA," the Open Web Analytics project.

So anyway, I'll finish quoting this writer by observing that he's being extremely kind in his description of this clearly bogus owebanalytics.com site, which shows this CentOS page. So I went over, I was curious, to http://owebanalytics.com to take a look around. First of all, it's http, not https. So I thought, ah, I should probably switch that to https to see what its certificate looks like. Well, the certificate has a 90-day life, so we wouldn't be surprised to learn that it is signed by Let's Encrypt. But what's weird is that the common name on the certificate is cdn.owebanalytics.com. And that doesn't have an IP address associated with it, and a website. So I was thinking, maybe the article was wrong, and the actual domain was cdn.owebanalytics.com. But as I said, there's nothing there. So the common name doesn't match.

Anyway, I've got a link to the thread for anyone who's interested. And as you immediately responded, Leo, this is sort of generally a problem with extensions overall is that they often have a long history. They acquire a bunch of users. And it's possible for them to sort of go sideways. But it's easy to imagine that some party with less than completely charitable intentions might offer someone, who originally developed and has been thanklessly maintaining a free browser extension that's been steadily growing in popularity through the years, some cash to buy them out of their thankless maintenance role.

And you can see how that might be an appealing opportunity after many years of tireless effort. The seller, who likely still feels some responsibility for their project, hopes that it's all going to work out and wouldn't want to disparage the project's new owner. Yet there's probably some reason why control of a well-regarded and highly used open source extension was worth some money to its purchaser. And indeed there were some activities discussed back in November, as I noted, that appeared to provide some hint of what was to come.

So we don't know what finally happened to trip an alarm to cause Google to yank the extension completely from the Chrome Web Store. But the writing was certainly on the wall. And we've talked before about web browser extensions being allowed to have the power to filter and modify all web content, not just through them, but all web content coming to and from the browser. As we know, something like uBlock Origin needs to do that. It's not just running on a single tab. It's an extension which is extending the browser as a whole. So that certainly presents a sobering danger. We know that Chrome has made policies, we've talked about this in the past, where extensions need to have some reason for performing this sort of behavior. They also need to be fully reverse engineerable by anybody who wants to see what a minified script is doing. They need not to be obfuscating themselves to an unusual degree.

So anyway, the GitHub thread did note at the end that The Great Suspender has been removed from the Chrome Web Store. They said: "To recover your tabs, see issue #526." It turns out this has been an issue in the past. They said: "The code in the GitHub repository is currently safe. The most recent tagged release happened before the transfer of ownership." And they wrote: "To use that version, and avoid needing to finagle URLs, enable Chrome Developer Mode, download and extract a copy of the code" - meaning from GitHub - "then navigate to your extensions menu and select Load Unpacked Extension."

So if anybody has been inconvenienced by this, or you want The Great Suspender, it is still available. And I guess that's sort of a side effect benefit of whoever it was who took this over never bothering to port their changes back to GitHub. Actually, they should have been created on GitHub and used from that. But anyway, let's hope that whoever purchased the extension lost money on the deal and that they and others will be disincentivized from attempting to purchase and subvert other browser extensions.

What we've just witnessed is a worrisome reality of our current web browser ecosystem. I know that I and many of our listeners rely heavily on extensions, both for Chrome and Firefox. And I would never want to have to use browsers without them. But it is the fact that we're often using extensions that are developed by people who love them, like the guy who did uBlock Origin, Gorhill, who seems like a grumbly old curmudgeon. But I'm sure glad that he cares as much as he does, and I appreciate his efforts.

The second thing that happened in Chrome Land was that on Thursday, Tenable and Microsoft both provided information about the otherwise under-mentioned, to put it lightly, update to Chrome that occurred also on Thursday. Tenable's posting explained what they know, and I'll extract a couple bits from what they wrote. I have a link to their full posting in the show notes.

"On February 4th, Google published a stable channel update for Chrome Desktop," Tenable wrote. "This release contained a single security fix to address a critical zero-day vulnerability that had been exploited in the wild. The vulnerability is a heap buffer overflow vulnerability in Chrome's V8 engine whose discovery is credited to Mattias Buelens. He reported the flaw to Google on January 24th.

"Google noted that they are 'aware of reports that an exploit' for this vulnerability 'exists in the wild,'" they wrote, "which we interpret to mean that in-the-wild exploitation attempts have been observed. Google's bug report for the vulnerability is unsurprisingly restricted to allow users time to apply the relevant patch," meaning the update to Google.

"In an interesting timing," they said, "this flaw was disclosed to Google just one day before a significant revelation from Google. On January 25th, Google's Threat Analysis Group (TAG) published a blog posting detailing the discovery of an ongoing campaign" - which we'll talk about in a minute, it's sort of really fascinating - "conducted by nation-state actors believed to be in North Korea, which is targeting security researchers who are interested in collaborating on vulnerability research." In other words, unwittingly collaborating.

"The report specifically mentioned that the threat actors circulated a link to their potential victims to a malicious website that led to successful exploitation on systems that were fully patched at the time for both Windows and Google Chrome. This was corroborated by Microsoft, which published their own blog post about the attacks, surmising that the Google Chrome zero-day was likely used to target researchers." Meaning North Korea had set up a full false flag operation targeting other security researchers. What Microsoft discovered and shares is amazing and kind of horrifying.

Microsoft said: "Over the past several months, the Threat Analysis Group has identified an ongoing campaign targeting security researchers working on vulnerability research and development at different companies and organizations. The actors behind this campaign, which we attribute to a government-backed entity based in North Korea, have employed a number of means to target researchers which we will outline below. We hope this post will remind those in the security research community that they are targets of government-backed attackers and should remain vigilant when engaging with individuals they have not previously interacted with."

Microsoft wrote: "In order to build credibility and connect with security researchers, the actors established a fake research blog and multiple Twitter profiles to interact with potential targets. They've used these Twitter profiles for posting links to their blog, posting videos of their claimed exploits, and for amplifying and retweeting posts from other accounts that they control. Their blog contains write-ups and analysis of vulnerabilities that they have publicly disclosed, including guest posts from unwitting legitimate security researchers, likely in an attempt to build additional credibility with other security researchers."

They said: "While we are unable to verify the authenticity or the working status of all of the exploits that they have posted video of, in at least one case the actors have faked the success of their claimed working exploit. On January 14th, 2021, the actors shared via Twitter a YouTube video they uploaded that proclaimed to exploit CVE-2021-1647, a recently patched Windows Defender vulnerability. In the video they purported to show a successful working exploit that spawns a command shell, but a careful review of the video shows the exploit is fake. Multiple comments on YouTube identified that the video was faked and that there was not a working exploit demonstrated. After these comments were made, the actors used a second Twitter account" - which by the way they also control, but pretending that they don't - "to retweet the original post and claim that it was not a fake video.

"The actors have been observed targeting specific security researchers by a novel social engineering method. After establishing initial communications, the actors would ask the targeted researcher if they wanted to collaborate on vulnerability research together, and then provide the researcher with a Visual Studio Project. Within the Visual Studio Project would be source code for exploiting the vulnerability, as well as an additional DLL that would be executed through Visual Studio Build Events. The DLL is custom malware that would immediately begin communicating with actor-controlled command-and-control domains. An example of the VS Build Event can be seen in the image below." And they provided that in their posting.

"In addition to targeting users via social engineering, we have also observed several cases where researchers have been compromised after visiting the malicious actors' blog. In these cases, the researchers followed a link on Twitter to a write-up hosted on blog.br0vvnn[.]io; and, shortly thereafter, a malicious service was installed on the researcher's system and an in-memory backdoor would begin beaconing to an actor-owned command-and-control server.

"At the time of these visits, the victim systems were running fully patched and up-to-date Windows 10 and Chrome browser versions. At this time we're unable to confirm the mechanism of compromise, but we welcome any information others might have. Chrome vulnerabilities, including those being exploited in the wild" - that is, ITW, in the wild - "are eligible for reward payout under Chrome's Vulnerability Reward Program. We encourage anyone who discovers a Chrome vulnerability to report that activity via the Chrome VRP submission process."

And of course note that it is now believed that this was just patched, this was the just-patched Chrome zero-day that was being used to compromise the systems of trusted security research collaborators. So they finish, saying: "These actors have used multiple platforms to communicate with their potential targets, including Twitter, LinkedIn, Telegram, Discord, Keybase and email. We are providing a list of known accounts and aliases below." Which they did in their disclosure.

They said: "If you have communicated with any of these accounts or visited the actors' blog, we suggest you review your systems for the indicators of compromise (IOCs) provided below. To date, we have only seen these actors targeting Windows systems as a part of this campaign. If you are concerned that you are being targeted, we recommend that you compartmentalize your research activities using separate physical or virtual machines for general web browsing, interacting with others believed to be in the research community, accepting files from third parties, and your own security research."

So, wow. So here was a high-end, focused, deliberate, multi-month campaign launched by, it's believed, malicious actors in North Korea that put together an entire fake front, looking like one of the growing number of security research groups who were then reaching out to the real research community, opening up invitations to collaborate. They were running a blog. They were faking videos of things that they had accomplished for themselves, engaging the research community, and using two different methods in this case, one a malicious Visual Studio project that had an extra little DLL of malware that would launch when you launched the project.

And then separately, using a previously unknown at the time that was effective against fully patched Windows 10 and Chrome, a zero-day, to compromise the systems of people who clicked on a link that was posted in one of their Twitter feeds. So I guess the moral of this one would be you just can't be too careful. Security researchers hopefully are sandboxing their own research systems. I would imagine they would be, you know, off from the rest of their networks, because they are going to be doing research that is highly vulnerable. I remember back in the day when I was first looking at viruses on DOS systems, back when viruses lived on floppy disks because that was the only way they could go around. This was all pre-network and pre-Internet.

Leo: Yeah. We had a machine painted bright red in the studio at The Screensavers that we would try things like Melissa on. And of course in those days air-gapping a computer was pretty common because, you know. But this was air-gapped and bright red so that nobody would be tempted to use it.

Steve: Yeah.

Leo: It's a little harder now.

Steve: Yeah. So our third story, and this is really interesting. This one was carried by every security-related news outlet last week. It's just sort of an interesting twist. The discovery of a unique use of Chrome's sync feature for command-and-control and data exfiltration. Also last Thursday a Croatian security researcher - who I'm certain would appreciate just being referred to by his initials BZ rather than have me attempt to pronounce his name, which has no vowels. Last Thursday, he discovered that a malicious Chrome extension was abusing the Chrome sync feature as a way to communicate with a remote command-and-control server to exfiltrate data from infected browsers.

As we know, multiple Chrome web browsers, which are logged into the same Google account, will automatically share and synchronize their configuration settings - tabs, favorites, extensions, browser history and so forth. Each browser connects to the Google mothership to check in. And then Google hands out updates as needed to whichever other browsers may check in while logged into the same account.

So what's sort of diabolically clever about this is that this communication, which will be encrypted under Google's own security certificates because the Chrome browser is contacting, I think it's client4.google.com, would typically go completely unnoticed by anyone. It would slip right through any corporate firewalls. Data could be encoded, for example, into long Base64-encoded URL tails, and Google would simply send them out to other browsers that are logged into the same account.

So this researcher in Croatia whose initials are BZ said that the incident he investigated found that attackers gained access to a victim's computer. But because the data they wanted to steal was inside an employee's portal, they downloaded a Chrome extension onto the user's computer and loaded it via the browser's developer mode that we were just talking about before for The Great Suspender. The extension, which posed as a security add-on from security firm Forcepoint, contained malicious code that abused Chrome's sync feature as a way to allow attackers to control the infected browser. In this way, the extension could be used as an exfiltration channel from inside corporate networks to an attacker's remotely located Chrome browser instance, or as a way to remotely control the infected browser from afar, thus bypassing any local security defenses.

BZ explained that blocking access to the Chrome sync server at client4.google.com would not work because that domain is used for many other things, such as Chrome, to detect an Internet connection. So instead of doing that, BZ urges companies to use Chrome's enterprise features and group policy support to block and control what extensions can be installed in the browser to prevent the installation of rogue extensions like the one he investigated.

And actually, I would say that would be a useful belt-and-suspenders approach. I haven't looked into this, but presumably we all have group policy controls in our systems. And if it's possible to use group policy to block Chrome from installing anything that you don't deliberately install, that would seem to be a useful thing to do for anyone who wanted to harden their Chrome install. It's certainly useful in an enterprise environment. But it would probably be equally applicable for individual users.

I reached up and restarted those three machines.

Leo: Oh, they're blinking.

Steve: So those watching will be glad to know that the lights are flashing again.

Leo: Our audience is really interesting. They're very focused, not just on the show, but on all of the details. So when my machine stops or, you know, last week they said, "What's the deal with Steve's lights? They're not blinking." If this clock is not here, they get all upset.

Steve: Well, and Leo, I'm sure both of us recognize that all of the talking heads on TV now are talking from their homes.

Leo: I love that. I do the same thing.

Steve: And so it's really, you know, you're like looking around, read the books, see what the books are. Oh, yeah.

Leo: I love doing that. You know what, I hope we don't go back to the old way. Don't go to the studio. Everybody should just stay home. We've got the technology.

Steve: So following on the heels of this Chrome news, we have the little whoopsie that Defender, Windows Defender, thinks Chrome is malware.

Leo: Well, it kind of is.

Steve: Yeah.

Leo: It may not be completely wrong after those three stories.

Steve: Yeah, with a subhead of "No good deed goes unpunished." No sooner had Google quickly updated Chrome last week to remove the zero-day flaw in its V8 engine, which as we know is being actively exploited in the wild to attack security researchers, than Microsoft's Enterprise version of Windows Defender decided that Google's modifications were malicious.

Leo: Oh, that's hysterical.

Steve: The high end, Microsoft Defender Advanced Threat Protection (ATP), which is the commercial version of the otherwise ubiquitous Defender AV which we're all using in our Windows machines, and that's Microsoft's premier enterprise security solution, it was having a bad day and labeled Google's browser update a trojan, a backdoor trojan. Based on Twitter reports posted by dismayed sysadmins, Defender ATP was detecting multiple files which are part of last week's Chrome, it was v88.04324.146. And I noticed I was already at 150. So they've tweaked it again since.

They said it was containing a generic backdoor named - it was a PHP backdoor, PHP/Funvalget.A. Though this might have normally been met with somewhat more calm, of course in this case Defender's alerts raised some alarm and quite a bit of stir in enterprise environments due to the recent multiple software supply chain attacks that we've all been made quite aware of recently. So sysadmins were awaiting a formal statement from Microsoft to confirm that the detection was indeed a false positive and nothing to worry about.

Fortunately, the built-in, no-charge, free version of Microsoft Defender AV that we all have in our personal Windows 7, 8, and 10 machines, was not suspicious of this new release of Chrome, which on one hand is fortunate, or it would have been a far more widespread mess. But it does make one wonder about the detection differential between Microsoft's commercial and the consumer AVs. Why exactly did the enterprise AV freak out, whereas the consumer AVs remained quiet? Does that mean we're getting less protection on the consumer AVs?

Anyway, Microsoft did later confirm that this Funvalget trojan, PHP trojan, for those Chrome files, were indeed a false positive, due to what Microsoft termed "an automation error." I guess that's technically true, whatever the heck that means. And I suppose they needed to call it something because calling it what it really was, a false positive, probably couldn't get past the PR folks at Microsoft. Like no, no, no, no, you can't call anything a "false positive." Come up with some other name. That would be, what, in fact it's just kind of an automation error. Oh, yeah, that's fine. No one will know what that means.

So I should mention that over the past few years I have had to exclude my local Windows Defender from poking into many of my own development directories. Unless I do that, shortly after I build a new executable from source, Defender will slide up a red warning from the lower right of the screen, saying not to worry, it's all good, Defender has found and removed the threat that just appeared. Of course that's my own code, just freshly built from source. So it couldn't be any cleaner. And recently they've all been, like the work that I've been doing has all been DOS executables.

And I noted previously that if, you know, if any malicious trojan were actually to find itself running in DOS, which it probably couldn't even do these days, it would not be happy. What must be upsetting Defender, because I'm having now to whitelist all the work that I'm doing, it must be that it's the lack of digital signatures on my freshly built EXEs. We've seen through experimental evidence that Defender now places a lot of weight upon the reputation of the certificates which sign today's executables. It's gotten to the point where it's difficult to keep Defender from complaining when any executable is not signed.

Leo: That must be a problem for all developers. That's ridiculous.

Steve: Yeah. It absolutely must be. I mean, it's just - it's crazy. I mean, these things have become...

Leo: They're EXEs? They're not .coms?

Steve: Yeah, they're EXEs. Which is I'm sure what Defender looks at and immediately digs into. And so think about how - we know how many bazillion different malicious things they're looking for. And so it's just going to be the case that somewhere in an EXE there's going to be a collision of some little fingerprint that matches enough to upset Defender. And then it looks and sees, oh, and look, there's no signature. Well, when in doubt, protect the user.

And of course today we know that Windows refuses, Windows itself, the OS, refuses to load any unsigned device drivers into its kernel. Device driver signing is no longer optional. Developers can force the issue by disabling Windows driver signing enforcement, as it's called. But I'll bet that the way things are going, we're not far from the day when Windows will be elevating that requirement to the desktop. And, boy, that'll cause a big uproar.

Leo: Well, Apple does that already. Apple's already doing that, yeah.

Steve: Yes. Yeah. And I don't think we're far away from that in Windows. There might be something like a UAC dialog that users are forced to push past whenever they wish to run any unsigned executable. And that would of course, from a social engineering standpoint, that would tend to cause developers to start signing their EXEs. The problem is certificates are not free. There is no "Let's Sign" equivalent of Let's Encrypt.

Leo: Apple has two layers. They have signing and notarizing. And they will not open an app that's not notarized unless, you know, you have to know how to jump through hoops. It's not even a UAC escalation. You have to actually jump through hoops to get the thing opened. Their gatekeeper is very aggressive.

Steve: Oh, yeah, UAE. I mean UAC.

Leo: UAC. I don't know. Did you say UAE? I said [crosstalk].

Steve: I have UAE in my show notes.

Leo: Oh, okay. UAC. All right. We both got it wrong. You know, I don't want to derail you, but it must be - how do you develop a program like SpinRite, which is a DOS program, you know, the normal cycle for development is you write, compile, and run; write, compile, run. But for you, you can't - you're writing and compiling on Windows; right? Or are you working in DOS?

Steve: No, I am, I'm testing under DOS; but I am writing and assembling. And it turns out it was surprising difficult to do this.

Leo: I bet.

Steve: Because I have some tools which have been abandoned, and they're 16-bit tools. I found this thing called vDos. There's like DOSBox, and there's, you know. And of course there's Virtual Box, and there's a bunch of things. But there's one funky thing called vBox EXE.

Leo: So it lets you run DOS in a virtual machine.

Steve: Yes.

Leo: Ah.

Steve: Yes. And it's funny because when you launch it, it will read a config.sys and an autoexec.bat from your current directory.

Leo: Oh.

Steve: So what I did was...

Leo: You automatically launch SpinRite.

Steve: Well, no, no. Here's I'm just building, so I have to do my linking. I use a 16-bit linker in order to link 16-bit DOS code. And then I use a program to process my symbol files for the DOS debugger, which is only 16-bit code. It was called Periscope, and it was...

Leo: I remember Periscope.

Steve: ...written by a guy named Brett Salter, who unfortunately passed away a few years ago. I reached out to Brett to say hey, believe it or not, Brett, I'm still using Periscope, and it turns out he's not still breathing oxygen.

Leo: I remember his name. Wow.

Steve: Yup, yup. And then I also contacted Bob Smith, who was the author of 386 to the Max. It was my favorite memory manager back then.

Leo: You're still using that?

Steve: Yup. And Bob is still around.

Leo: Wow. So that's interesting. That makes sense. So at least you can do it all on the same machine.

Steve: Yes. So I'm doing it all on the same machine. And then the problem I recently had was that in order to be in DOS, I want to debug with symbols. So the DOS box needs access, I mean, I want to debug with source.

Leo: Right.

Steve: So the DOS box needs access to the source code.

Leo: And the symbol tables, yeah.

Steve: So for a while I was loading the whole Windows for Workgroups 3.1 IP stack in DOS and running a Windows for Workgroup client in order - and then I would use share to map the development directory into the C drive under DOS so that the code running there could see the source code on my Windows machine from in DOS.

Leo: Wow.

Steve: Well, that all worked - and that's how I developed ReadSpeed. That all worked until I started working on SpinRite 6.1 because SpinRite itself is way bigger. ReadSpeed was 19K. And so that was no problem. SpinRite is a couple hundred K. And so the problem was the full DOS network stack took up way too much memory, along with the debugger and its symbol table, which was itself 160K. So there wasn't enough room.

Fortunately, one of the guys in the GRC SpinRite dev group had done some work and played around with using a packet driver. Instead of using the TCP/IP stack and then Windows for Workgroups 3.1, there are packet drivers for all these old network adapters. And the packet driver is super tiny, but all it is is just packets. So it turns out someone created - there's an FTP server, an FTP client, but also the equivalent of Wget. There's an HTTP GET command which is transient. So now my build process under Windows, whenever I do a build, and it's just an Alt+A, assembling, it just takes - it's like instantaneous.

Leo: Sure.

Steve: So I just use it for syntax [crosstalk].

Leo: Oh, yeah, all the time, yeah, yeah.

Steve: I'm building constantly. Every time I do that, it zips all of my assembly code and the header files into an sr.zip. And then when I switch over and run the debugger, the act of starting the debugger launches the HTTP GET in order to pull, using the packet driver. Now I've got 400K of RAM because I got rid of all of the IP stack in Windows for Workgroups. So it pulls the zip over, uses PKUNZIP, Phil Katz's old UNZIP, in order to unzip the files into an assem directory in DOS. And then the debugger says, oh, look, here's all the source code. So believe me, Leo, when I move, when I finally say goodbye to DOS and the BIOS, and I switch to SpinRite 7, where I'll be operating under a 32-bit mode, I mean, it's just - it's barely possible to still develop this way. I mean, I'm having to get very creative.

Leo: I'm impressed as hell. That's amazing.

Steve: In order to do that.

Leo: That's amazing. Wow.

Steve: It does work, yeah.

Leo: See what he does for us, folks? Nice job, Steve. Thank you.

Steve: So anyway, we do have another critical WordPress plugin problem, and I have a suggestion for anyone - I will end this with a suggestion for anyone who is still using WordPress. In this case, we have now more than 800,000, .8 million, WordPress sites vulnerable to - I gave myself the hiccups by not breathing while I was just talking here.

Leo: I don't blame you.

Steve: I get so excited about SpinRite.

Leo: People say, "Why is Steve still using those old operating systems?" Now you know why. You don't want to go to Windows 10 because it's just going to - what a nightmare it's going to create.

Steve: Yeah. And under XP everything still worked because it still ran 16-bit code. I was still using Brief.

Leo: I know, I know.

Steve: As my programmer's editor back then.

Leo: Well, your fingers knew the way.

Steve: Yeah. And with WordStar keystrokes because that was like the right way to do it back then.

Leo: Little footnote on that. The creator of MicroPro, the company that distributed WordStar, Seymour Rubinstein, died last week, in his late 80s, I think. But he was kind of a legend down here in Sausalito as the man who put WordStar in the market.

Steve: Yeah. And, boy, that was the word processor to use.

Leo: Yeah. All right. Hiccups are gone.

Steve: Okay, so, yes, thank you.

Leo: I scared you with WordStar.

Steve: More than 800,000 WordPress sites all share a very popular plugin called NextGEN Gallery. NextGEN Gallery allows sites to accept uploads of photos in batch quantities to import metadata and edit image thumbnails. So apparently 800,000 WordPress sites think that's a good idea. Whatever they're doing, they're wanting to use this plugin. The bad news is that researchers discovered two CSRF (Cross-Site Request Forgery) flaws. One is critical, and the other is high severity. A patch was released to address the flaws in version 3.5.0 of NextGEN Gallery.

So first, if anyone who's listening to this is using NextGEN Gallery on their WordPress site, make sure that you've running 3.5.0 or later. It's been available since the middle of last December, so the researchers had responsibly waited until yesterday, Monday the 8th, before publicly disclosing the details of the flaw, which are now public. So that ups the ante on anyone's need to make sure they're running the most recent version.

Ram Gall, who is with WordFence, who we'll be talking about in a minute because I think this thing is worth taking a look at, he wrote in their disclosure of the vulnerabilities yesterday of, he said: "A critical severity vulnerability that could lead to remote code execution and stored cross-site scripting vulnerabilities. Exploitation of these vulnerabilities could lead to a site takeover, malicious redirects, spam injection, phishing, and much more." So his description of the vulnerability demonstrates the competence of these guys.

He said: "We initially reached out to the plugins publisher, Imagely, the same day, and provided full disclosure the next day, on December 15th, 2020. Imagely sent us patches for review on December 16th, and published the patched version, 3.5.0, on December 17th." So you couldn't ask for any better find, notify, full disclosure patch and update availability. Of course the only thing missing from this cycle is, of those 800,000 people using it, because it is inherently publicly exposed, how many of them are now running 3.5.0.

Ram Gall said: "Wordfence Premium users received firewall rules protecting against these vulnerabilities on December 14th, 2020. Sites still running the free version of Wordfence, which is what I'm going to be recommending by the end of this, received these rules 30 days later, on January 13th." But still a full month before public disclosure. Well, no, wait a minute. Public disclosure was yesterday, so three weeks.

He said: "NextGEN Gallery" - this plugin - "is a popular WordPress plugin designed to create highly responsive image galleries. It is clear the plugin's developer took care to integrate security in the code of the plugin. NextGEN Gallery has a single security function" - the function name is is_authorized_request - "that is used to protect most of its settings." And I've got a link to the full disclosure in the show notes where he shows this PHP function. He said: "This function integrated both a capability check and a nonce check into a single function for easier application throughout the plugin. Unfortunately, a logic flaw in the is_authorized_request function meant that the nonce check would allow requests to proceed if the nonce parameter was missing, rather than invalid." Whoops.

"This opened up a number of opportunities for attackers to exploit a cross-site request forgery. One feature of NextGEN Gallery is the ability for administrators to upload custom CSS files to be used to style galleries. While the file uploaded had to end with the .css extension" - and you know where this is going, folks - "it was possible to upload arbitrary code with double extensions, in other words, file.php.css. While these files would only be executable on certain configurations, such as Apache/mod_php with an AddHandler directive, this could still result in remote code execution on any vulnerable configurations.

"Unfortunately," he wrote, "it is also possible to achieve code execution even on configurations not vulnerable to double extensions. NextGEN Gallery has a separate feature that allows users to specify how galleries are viewed via a 'Legacy Templates' feature" - why does that sound scary? - "which also uses the is_authorized_request function for security. Thus, it was possible to set various album types to use a template with the absolute path of the file uploaded in the previous step, or perform a directory traversal attack using the relative path of the uploaded file, regardless of that file's extension, through a CSRF attack.

"This would result in local file inclusion and remote code execution, as the uploaded file would then be included and executed whenever the selected album type was viewed on the site. Any JavaScript included in the uploaded file would also be executed, resulting in a cross-site scripting problem. As a reminder, once an attacker achieves remote code execution on a website, they have effectively taken over that site. Cross-site scripting can likewise be used to take over a site if a logged-in administrator visits a page running a malicious injected script.

"This attack would likely require some degree of social engineering, as an attacker would have to trick an administrator" - that is, just this final aspect - "into clicking the link that submitted crafted requests to perform these actions. Additionally, performing these actions would require two separate requests, though this would be trivial to implement, and we were able to do so during our testing. Finally, the site would require at least one album to be published and accessible to the attacker."

So that's what they posted. And I'm impressed by these WordFence people. We've run across them several times in the past year as WordPress problems have been receiving increased scrutiny, both from attackers and from security researchers. So I'm glad that I am no longer running WordPress on any of my own servers. But I understand that others may have little choice. As I noted above, WordFence offers both a free and paid version of their WordPress firewall. And if I had to be running WordPress, I would give WordFence a serious look. They really do appear to be on the ball and quite worthwhile.

Since it is plugins that appear to be causing all of this trouble, if you happen to be running WordPress lean, with no plugins, then I would say it's a safe step to skip it. But if you are a plugin user, and you just can't resist adding goodies to your WordPress installation, then I would definitely add one more. I would add WordFence and then consider what their pricing structure looks like and whether it might be, depending upon your use of WordPress, if it's mission-critical, I think they were $99 a year for their full-paid, single-site protection. And having these guys watching your back I think makes an awful lot of sense if you're a WordFence user.

I did want to just check in on DDoS attacks. We haven't talked about DDoS for a long time, but last Thursday the site NETSCOUT posted a notice about the abuse of Internet-exposed Plex Media Servers and their SSDP protocol, which of course we've spoken of often. SSDP is - I'm looking for it here in the notes. It's the UPnP protocol, but I'm blanking on what it is. I'll run across it here in my notes. Anyway, we've not talked about DDoS for a while. What NETSCOUT wrote I thought was interesting.

They said: "Plex Media Server is a personal media library" - of course, as we know - "and streaming system which runs on modern Windows, macOS, and Linux OS." I've got it running in my Drobos for a number of applications. They said: "...along with variants customized for special-purpose platforms such as network-attached storage devices, external RAID storage, digital media players, et cetera." They said: "Upon startup, Plex probes the local network using the [it's called] G'Day Mate (GDM) network/service discovery protocol to locate other compatible media devices and streaming clients. It also appears to make the use of SSDP probes to locate Universal Plug and Play gateways on broadband Internet access routers which have SSDP enabled."

So once again, this is another reason for always disabling SSDP, the public side Universal Plug and Play, unless you know you need it. They said: "When a Universal Plug and Play gateway (UPnP) is discovered via this methodology, Plex attempts to utilize NAT-PMP to instantiate dynamic NAT forwarding rules on the broadband Internet access router." In other words, to open a port to itself so that it is available and discoverable on the Internet. And I have no idea why. This is just insane.

On January 7th of this year, Baidu Labs, in a Chinese language weblog post, described a UDP reflection/amplification DDoS attack vector leveraging Plex Media Server instances running versions of the Plex software prior to 1.21. In early February 2021, NETSCOUT Arbor were notified that reflection/amplification DDoS attacks appeared to be leveraging abusable Plex Media Server instances which were actively taking place on the public Internet.

Okay. So first of all, the Plex Media Server has this functionality where, when it comes up, it will check with the gateway to see if your router has UPnP enabled, as they typically do now because it's a feature we can advertise, like all of those ridiculous protocol gateways that I talked about disabling last week, the web application protocol gateways. In this case, it will find typically a Universal Plug and Play gateway, use SSDP (Simple Service Discovery Protocol) to talk to it, and arrange to map a port through to itself, presenting itself on the public Internet because what could possibly go wrong?

According to an announcement published on Plex's website on February 5th, Plex Media Server instances which had either been deployed on a public-facing network DMZ or in an Internet datacenter, or with manually configured port-forwarding rules which forward specific UDP ports from the public Internet to devices running Plex Media Server, or which are behind a router with UPnP operating, can potentially be abused as part of possible DDoS attacks. So at least Plex is aware of this. The problem is that how many Plex users are aware of this?

They said: "These actions could have the effect of exposing a Plex UPnP-enabled service registration responder to the general Internet, where it can be abused to generate reflection amplification DDoS attacks. In order to differentiate this particular attack vector from generic SSDP reflection amplification, it has been designated as Plex Media SSDP, or PMSSDP. To date" - and get this - "approximately 37,000 abusable PMSSDP reflectors/amplifiers have been identified on the public Internet."

So yes, it's the default; right? By default, it reaches out and checks for Universal Plug and Play. By default, consumer routers have that enabled. By default, it will create a mapping back to itself. And as a consequence, 37,000 of these Plex Media Servers are poking their nose out onto the Internet, and now they're of interest to attackers. They may not be able to log onto them or care what's there. But there is a server which they're able to bounce packets, DDoS traffic off of. And being SSDP, it is UDP. Which means you don't have to have a TCP handshake. UDP, as we know, is perfectly spoofable. So you're able to lie about your source IP, send a packet there, and that device will bounce a larger packet back to presumably you, but if you've spoofed your IP, to your DDoS attack target.

So they said: "Amplified PMSSDP DDoS attack traffic consists of SSDP, HTTP/U, meaning an HTTP protocol over UDP, responses sourced from ports 32414 and 32410, or abusable Plex Media Server instances and directed towards attack targets. Each amplified response packet ranges from 52 bytes to 281 bytes in size for an average amplification factor of about 4.68 to 1." So not a huge amplifier, but there's 37,000 of them sitting there on the Internet just hoping you're going to send a UDP packet off of them that they can increase in size and bounce toward your target.

They said: "Observed single-vector" - so we're going to have the term "single vector" and "multi vector" now is the jargon of DDoS attacks. "Observed single-vector PMSSDP reflection/ amplification DDoS attacks range in size from about 2Gb to 3Gb; multi-vector, meaning 2 to 10 vectors; and omni-vector, which is considered 11 or more vectors." Meaning where vectors are different things that packets are bounced off of. So the PMSSDP would be one vector among many. So if attacks are only based on PMSSDP, that is, the Plex Media Server, those attacks are typically around 2 to 3Gb. But there are multi-vector, 2 to 10 vectors, and omni-vector, 11 or more vector attacks, which incorporate now PMSSDP as one of their multiple vectors because again, 37,000, why not? "Those range from the low tens of Gbps up to 218 Gbps." So that is to say, more than one fifth of a terabit per second.

They said, this is NETSCOUT: "As is routinely the case with newer DDoS attack vectors, it appears that after an initial period of employment by advanced attackers with access to bespoke DDoS attack infrastructure, PMSSDP has been weaponized and added to the arsenals of so-called booter/stresser DDoS-for-hire services" - booter as being booted off the Internet - "placing it within the reach of the general attacker population."

They said: "To date, more than 5,500 PMSSDP" - meaning single-vector - "reflection/amplification DDoS attacks have been observed on the public Internet, leveraging approximately 15,000 distinct abusable PMSSDP reflector/amplifiers," meaning approximately 15,000 distinct Plex Media Servers.

They said: "It should be noted that a single-vector PMSSDP reflection attack of 2 to 3Gb in size is often sufficient to have a significant negative impact on the availability of targeted networks, servers, and services. The incidence of both single-vector and multi- or omni-vector attacks leveraging PMSSDP has increased significantly since November of 2020, indicating its perceived utility to attackers." In other words, they've added the Plex Media Servers to their bag of tricks for these omni-vector/multi-vector attacks because, again, you've got tens of thousands of them, so why not?

And I'll just finish with the question, or answering the question, just how prevalent have DDoS attacks become these days? To answer the question, BleepingComputer opened an email dialog with Richard Hummel, who's NETSCOUT's Manager of Threat Intelligence. Richard wrote that: "The total number of Plex Media SSDP attacks from January 1st to present day clocked in at approximately 5,700, compared to the more than 11 million attacks in total we saw during the same timeframe." That was from January 1st of 2020, so that's a year and a month, or a year and a month and a half, 11 million DDoS attacks. So, wow. There are plenty of them.

So three more new vulnerabilities have been discovered in SolarWinds software.

Leo: Three more. What?

Steve: Yeah. I hope this podcast's listeners are aware of the extremely disturbing fact that we keep encountering instances of what I'll term the principle of "Wherever we look, we discover new problems." Today, problems are being discovered in two ways. First, the old-fashioned way, where we discover malware in some system, then reverse engineer the malware to discover how it got in. And we looked last week at the extreme measures the SolarWinds hackers went to in order to avoid exactly that form of reverse engineering. Remember they, like, worked to decouple the execution of the malware with the way it got into the system.

The new modern way of finding vulnerabilities is apparently simply by looking closely at pretty much anything and discovering that, oh, look, it's full of security weaknesses. Who knew? It's this new second reality that has turned vulnerability discovery into a potential career. Just find some company that has the wisdom to offer bounties for the discovery of their bugs, then take a close look at their code. And before long, you can probably use cash to buy yourself a new car.

Last week another case illustrating this disturbing truth just came to light thanks to the many researchers who have been looking more closely for the first time ever at the code being shipped by SolarWinds. Of course we know what motivated them to do so. The whole security industry took a look at SolarWinds code to figure out what the heck? So Trustwave's most recent security labs blog, posted last Wednesday, was titled "Full System Control with New SolarWinds Orion-based and Serv-U FTP Vulnerabilities." For anyone interested, I've got the link in the show notes for the full posting. I'll just excerpt two bits from it.

Martin Rakhmanov posted in the first person, saying: "In this blog, I will be discussing three new security issues that I recently found in several SolarWinds products. All three are severe bugs, with the most critical one allowing remote code execution with high privileges. To the best of Trustwave's knowledge, none of the vulnerabilities were exploited during the recent SolarWinds attacks or in any in-the-wild attacks. However, given the criticality of these issues, we recommend that affected users patch as soon as possible. We have purposely left out specific proof-of-concept code in this posting in order to give SolarWinds users a longer margin to patch; but we will post an update to this blog that includes the proof-of-concept code on Feb. 9th."

Okay, now, he didn't give anyone much time. This was posted last Wednesday. Today is Tuesday the 9th; and sure as anything, the proof-of-concept code has been added to the blog. So it's now public. Yet another set of new ways of exploiting the SolarWinds code that was current as of last week. There are now public proofs of concept for the following disturbing three new vulnerabilities. SolarWinds Orion platform we have CVE-2021-25274, and they're sequentially numbered 275 and 276.

The first one, improper use of Microsoft Message Queue, could allow any remote unprivileged user the ability to execute any arbitrary code with the highest privilege. Okay. Doesn't get any worse than that; right? Any unprivileged user remotely to execute any arbitrary code with the highest privilege.

The next one, SolarWinds credentials are stored in an insecure manner that could allow any local users, despite privileges, to take complete control of the SolarWinds Orion database, which is credential store is what it is. From there, one can steal information or add a new admin level user to be used inside SolarWinds Orion products, essentially neutering all of its authentication; right?

And third, SolarWinds Serv-U FTP for Windows. Any local user, regardless of privilege, can create a file that can define a new Serv-U FTP admin account with full access to the C root drive. This account can then be used to log in via FTP and read or replace any file on the drive. This is unbelievable.

So it's really not here my intention to single out SolarWinds. Yes, they're currently and deservedly in the hot seat. But we would be wrong to assume that we just happen to be finding all manner of serious problems with the only company whose offerings have recently been closely scrutinized; right? The only sane assumption, until we learn otherwise, would be that the software published and in use widely by many other similar entities would crumble just as quickly, if and when it were to be subjected to a similar level of close expert scrutiny.

The entire industry just assumed that SolarWinds was sufficiently good and careful about network security in the beginning. Which is what they were selling, after all. Their whole offering is a security product. And it probably said that somewhere on their website, that it was very secure, right next to their customer list, which has since been taken down, as I noted before. So because such close expert scrutiny is expensive, no one is doing that to most of the industry's software. There are exceptions like web browsers, which we talk about constantly, that are very intensely scrutinized because they're such a high-profile target. But we now believe that highly skilled and talented Russian attackers were caught having closely scrutinized SolarWinds' systems and code.

But the evidence begs the question, what other software company's work has this same group also examined closely? And what did they find? Any sane person looking at the evidence would have to think with all the problems that are being found in SolarWinds products, it's probably not the case that there are not other similar products in other places that have not been carefully examined. We talk about high-profile companies. Cisco has constant problems with their stuff because people are looking at them. But there are many other companies that are like SolarWinds, that are widely deployed, widely used, and no one is looking at their stuff. The problem is bad guys may have an incentive to be doing just that. Good guys don't have the incentive. Good guys are busy doing other things.

So I will close the loop with three bits of feedback from our listeners. Ndom91, who has a Twitter account and tweets under that moniker, said: "@SGgrc Steve, I love you, your work, and the show. But for the love of god, it's called 'lib' like 'libertine,' and not 'libe' like 'library.'"

Leo: See, it is a library, though.

Steve: I know.

Leo: I call it "lib," too, but I don't know if there's a correct pronunciation. What's his assertion? What's correct?

Steve: He's referring to the libgcrypt segment in Security Now! last week, 804.

Leo: You've always said "libe," though.

Steve: I have always said "libe."

Leo: Because it's short for "library."

Steve: I've also always said "jif," not "gif."

Leo: I've always said "lib," only because that's how I've always read it, like libgcrypt.

Steve: Yeah. And I've always said it "libe" because it's a library, not a lib-rary. I didn't go to the lib-rary as a kid in elementary school. I went to the library.

Leo: But who's the guy who determines this? There's no...

Steve: Yeah.

Leo: I'll tell you what. I've always said because we're saying it out loud, in many cases for the first time anybody's said it out loud except in private conversations, we get to decide what it is.

Steve: Yes. Lorrie has formal medical training. I'm trying to pronounce medical terms that I've only ever read. And she's constantly correcting me, saying, uh, honey, it's pronounced this way. It's like, oh. Okay, fine.

Leo: I've always said like "libsodium" or "libgcrypt." But, you know, to each his own. I wouldn't correct you. I don't think that that's...

Steve: Anyway, if anybody else is like desperately upset with me for saying "libgcrypt," you've just had your day. That's the end of it. I'm not changing. But at least it's been aired. Meanwhile, Dave Stricker, tweeting from at @strickdd, said: "You seem to be sending mixed signals on this week's SN. In the past you've indicated that everything should have an auto-update mechanism. This week it sounds like you don't trust them. Is it just Notepad++ you don't trust, or is it because it prompts to update whereas Chrome does it in the background?"

And David, you're absolutely right. This is one of those situations, I don't know what Catch number it is. It's not exactly 22. Things that are bad should be updated. Things that are good should not be updated because they might go bad. How's that? Anyway, there's no good answer. We're all screwed, basically. But then - actually, that'd be a great name for the podcast, Leo. Wouldn't it? Welcome to We're All Screwed Podcast #805.

Leo: Well, it's a little late for that. I like Security Now!.

Steve: Might be hard to get advertisers.

Leo: Yeah.

Steve: So, finally, AndyMan7 tweeting from @Man7Andy, says: "Hello, Steve. Big fan of Security Now. I recall you mentioning a remote desktop management tool on one of the shows, but can't remember the name. Would you help remind me what that was, and if you have any recommendations for a safer tool for IT people to do remote sessions to PCs?" Okay. I'm glad you asked, AndyMan. The tool is simply called Remote Utilities. It is at www.remoteutilities.com. And I continue to be so impressed with it.

Lorrie uses it constantly, literally continually, daily, to manage the array of remote laptops which are being used by her clients who are doing at-home neurofeedback training. My tech support guy, Greg, who has a computer repair consultancy business on the side, has completely switched over to using it and has hundreds of machines under remote management. And I'm increasingly using the system to manage several of my own machines. It is an absolute win. The other thing I love about it is that it is purchased once and is not a subscription. And they offer a free license.

They said: "Our free license allows you to add up to 10 remote computers in your Viewer address book. You can use the license free in a business and personal setting. Only one free license key is allowed per individual, company, or organization. For more information, see our EULA." And if you need more than that, their "buy it one time, use it forever" is also reasonably priced. So anyway, I'm so glad you asked, Andy. These guys really deserve a look. Over in my blog on forums.grc.com I've created my own thread of my favorite things, just sort of a place to hold the things that I like most. Sync.com is there. Syncthing is there. This, Remote Utilities, is there. I just wanted a place to just state, these things deserve people's attention, and I think use.

So anyway, RemoteUtilities.com. These guys are great. And, boy, I mean, it's Remote Utilities because it's actually more than just a remote desktop thing. You can do like remote registry, remote file transfer, a whole bunch of different things. And lots of authentication. I use the one-time password, a time-based one-time password compatible with any of the authenticators. So, for example, when I'm connecting to one of my machines, I have to give a password and then a six-digit token which is part of, you know, I have the OTP Auth is my favorite OTP app on iOS. So anyway, just can't say enough good things about them.

I did want to mention, I posted a long posting I'm not going to read here, yesterday to GRC's spinrite.dev group. The subject was "Discovering System's Mass Storage Devices."

That's the screen that comes up. It's mostly blank, with a little window in the middle that says "Discovering System's" on the first line, and then "Mass Storage Devices..." on the second line. And then at the very bottom it has a flashing "Working." Anybody who's ever used SpinRite is familiar with this screen. I find that I still have a stress reaction to it because, if SpinRite was going to hang somewhere...

Leo: That's where it would break.

Steve: That's where it would break. And I'm sure that, if Greg has nightmares in his sleep, it's this flashing "Discovering System's Mass Storage Devices" because people complain: "I started SpinRite, I was all excited, and I waited an hour." Anyway, the reason I'm bringing it up is that I've just finished completely rewriting the code which is underneath that, is behind that.

Leo: That's probably a hard thing to do.

Steve: Well, and I didn't really intend to completely rewrite it.

Leo: You must use the operating system to do that. Or no? No, I guess you can't.

Steve: No. Now, well, so now, because I'm using the BIOS less and less, now I'm scanning the PCI bus doing a complete enumeration of every device on the PCI bus, checking to see what it is, whether the device declares itself to be mass storage. If it's a mass storage declaration, then I look at what type of mass storage. If it's an ATA/IDE/AHCI device, which SpinRite now understands, then I look at the hardware registers that it is declaring. I then access the registers, perform a sanity check on them, and then talk to - enumerate the drives that are attached, and then ask them for their drive identity information, their sizing information, their capabilities. Basically behind the scenes I'm fleshing out a huge data structure that describes all of the devices that SpinRite is able to talk to.

Once I'm through with that, then I go through a process that I call "BIOS association" because I need to know which of those things I have just found the BIOS already knows about. And so to do that I have hashed as part of that process the boot sector on all of those devices, and I have deliberately left them in an error condition. So then I use the BIOS to read the boot sector of device 80, which is the first BIOS device, and I hash that, then I perform a hash comparison of all of the hashes that I have found of all of the boot sectors of the hardware that I have found, hoping that I will get exactly one match - not zero, and not more than one, because in that case it means I have found which physical hardware device the BIOS is calling 80, in which case I add that to this database, and I go to the next BIOS device.

So I scan through all the BIOS devices, reading all of the signatures. If I don't get a one-for-one boot sector hash match, then the act of reading the device from the BIOS will have cleared the error condition that I deliberately left the hardware in. So then I look through - I update all of the error conditions of all of the hardware, hopefully finding exactly one that is no longer in error, in which case I have my second strategy for performing the BIOS to physical drive matching.

Leo: Wow.

Steve: Once all that's done, I then look to see whether I did get BIOS matches for all of the hardware. If not, then I add to this database the BIOS devices for which I do not have hardware driver support. So, for example, for SpinRite now, because I'm only doing AHCI and ATA IDE, other devices like USB connected or NVME connected, those will be supported by the BIOS. So those get added. And once that's all done, I look to see whether there are any hardware devices that did not have BIOS support, in which case they would not be labeled with BIOS designations, so I assign them numbers 1, 2, 3, and up. All of that is happening behind the screen that is flashing "Discovering System's Mass Storage Devices."

The point is I just rewrote all of that because I looked at the old code, and I realized, okay, I know how to do this so much better than I did 20 years ago. So the announcement that I was making to the group is that little anxiety-provoking flashing thing is going away.

Leo: Woohoo.

Steve: I decided it will be too much fun to animate that process.

Leo: To show what you're doing, yes.

Steve: Yes. That's what it's going to do.

Leo: That's a great idea. I love that.

Steve: And so it'll bring up a big empty window and show the PCI bus being scanned, and then devices being found, and it'll go bloop bloop bloop bloop bloop bloop bloop. It can't make that sound, unfortunately, because we're DOS. And besides, people tend to hate the sounds that I have SpinRite making anyway. But so that was what I wanted to say.

Leo: I'm sure you'll get somebody to volunteer new sounds if...

Steve: There's going to be - that screen that is tremendously anxiety provoking, both for SpinRite users and for Greg and for me, that's going away, to be replaced by - actually it will have a nice diagnostic benefit also because, if it does actually get stuck somewhere, we will know where. And so I'll then have something to work from.

Leo: That's great. That's really impressive, yeah.

Steve: A cool advance. And that gives you a little sense for what is going on behind the scenes. It's a simple-looking screen; but, yeah, as you said, Leo, there's a lot happening.

Leo: That's a hard thing to do, historically, yeah, enumerate all the devices attached to a computer. Not easy.

Steve: Okay. The story that this podcast was named after. Okay. We turn our attention to the small city of Oldsmar, Florida, home to approximately 15,000 residents. Oldsmar lies about 16 miles northwest of the much more widely known city of Tampa, Florida. The first signs that anything might be amiss at Oldsmar's municipal water treatment plant appeared last Friday morning, when a plant operator noticed someone had remotely accessed a system that controls chemicals and other aspects of the water treatment process. The operator reportedly didn't think much of the event since his supervisor and coworkers regularly logged into the remote system to monitor operations.

But then later that same day, in the early afternoon, around 1:30, the operator watched as someone remotely accessed the system again. He could see the mouse on his screen being moved to open various functions that controlled the water treatment process. This unknown person then opened the function that controls the input of sodium hydroxide, popularly known as lye, increasing it by 111 times. The intruder increased the level of sodium hydroxide to 11,100 parts per million from the normal proper level of 100 parts per million.

Lye, or sodium hydroxide, is used in very small amounts to treat the acidity of water and to remove metals. It's also the active ingredient in liquid drain cleaners. And, yes, in higher levels such as 111 times normal, it is highly toxic.

Leo: But your drains would be clean, so that's good.

Steve: Well, yes. Your pipes would be clean, and not only the pipes in your house, unfortunately. Had the change not been reversed almost immediately, it would have raised the amount of the chemical to toxic levels. And I'll pause our story here to wonder why it is even possible to adjust through automation the amount of lye to a level which is 111 times normal and clearly poisonous. That seems like a fundamental oversight in the design of the system. Sure, perhaps allow a range of 0 to 200%, but certainly not up to 11,100%. That's just loony.

Leo: Shouldn't be on the meter. It shouldn't be on the meter.

Steve: No, no, it shouldn't go that high. As they say, the stereo does not need to go to 11,100.

Leo: No.

Steve: It should just go to 10.

Leo: Yeah.

Steve: Fortunately, the operator, who was watching this happen, immediately changed, like grabbed his mouse and changed the setting back to the normal 100 parts per million. And supposedly, even if the malicious change had not been immediately reversed, other routine procedures within the plant would have caught the dangerous level before the water became available to residents. I guess it wasn't actually in the flow, it was in some tank being mixed or something. Probably true because if it needed to precipitate out metals, then it may have been added as part of the prep for a big vat. And it apparently takes anywhere between 24 and 36 hours for treated water of that sort to get into the supply. So in this case no poison water ever escaped.

The local county Sheriff's Department, however, there was a press conference held, and questions were asked. The Sheriff's Department did not immediately respond to questions asking whether the utility required personnel to use two-factor authentication to gain remote access to interfaces such as the one that was breached in Oldsmar. The Reuters news agency, citing an interview with managers, reported that TeamViewer was the application used to gain remote access, but the department didn't immediately respond about the requirements for authentication.

Jake Brodsky, an engineer with 31 years' experience working in the water industry, said it's not at all uncommon for water utilities to make such interfaces available remotely. While he frowns on the practice, he said that the managers were probably correct in stating that the public was never in any danger. In an interview, Brodsky said: "There's a bunch of different things water utilities look for; and if they see anything out of kilter, then they can isolate the storage water. The danger here is relatively minimal as long as you catch it soon enough, and there are multiple checks before that happens."

Of course, if intruders can remotely tamper with a process, they may also be able to tamper with the safety redundancies in place. This was obviously not a sophisticated attack. I mean, why would you do it at 1:00 p.m. in the afternoon? Do it at 2:00 a.m. and maybe it would go unseen. Anyway, if Brodsky were advising Oldsmar on better securing their water treatment plant, he said: "The first thing I'd probably do, and this almost doesn't cost anything, is you disable remote access," he said.

Leo: Yeah.

Steve: When remote access is required - yeah, gee, what a concept - as is occasionally the case, connections should be manually allowed by someone physically present, and the access should time out after a brief period. He said: "I can't imagine leaving a connection like that open and exposed to the world. This is cheap and easy," meaning to add some protection. "All you do is call the operator, and you get the access you need."

So, you know, stepping back from this, there has been so much talk and no obvious action through the years about the vulnerability of these SCADA systems, SCADA being the abbreviation for Supervisory Control and Data Acquisition. That's the generic term for all of these things that manage nuclear reactors and water treatment plants and pretty much anything. I believe that we are probably incredibly vulnerable in this country. There are too many instances like this where convenience has completely dictated policy and completely trumped security.

I sincerely hope that managers who are responsible for the operational security of their industrial plants of every description hear about what happened in Oldsmar and take it to heart. With any luck, it may have been a wakeup call. I'm glad it has gotten the attention that it has because this was not a highly skilled attack. And it is horrifying to think that something like this, if an attacker, I mean, it just seems like there is so much exposure. If you rely on the feedback only from a screen with onscreen meters, it would be so easy for someone to reset the meter to make it look like it was reporting 100 ppm and have it set to 11,100. And who would know?

Leo: The thing that I find interesting is that the person who got in seemed to know his way around pretty well. I mean, if you gave me access to that system, it might take me a while to figure out how to turn the lye setting up. It seems like an inside job to me. I don't know.

Steve: I'll bet you there are people in Russia who know water treatment plants.

Leo: Know that software and know how it works, yeah.

Steve: Yeah.

Leo: I mean, the fact that it's on the public Internet is crazy, just obviously crazy.

Steve: It is crazy. And Leo, we know that you know your way around GUI interfaces. I'll bet, if you were looking at the screen - and apparently this person did poke around at things first.

Leo: Oh, okay, okay.

Steve: There was some poking around. The mouse pointer didn't go directly to the lye setting. It opened various things. And then the person said, ooh, lye, that sounds bad. Let's turn that up.

Leo: Wow. What a story. Wow. Yeah, I hope it's a wakeup call. But we've seen stuff like this before.

Steve: I know.

Leo: And the worst thing is, if it is Russian or some sort of nation-state, they often do these kinds of probes. It is weird they did one in the afternoon. But maybe it's the middle of the night in Russia. I don't know.

Steve: Yeah. Maybe [crosstalk].

Leo: They would know better, I think. What time is it in Florida?

Steve: Maybe they weren't the brightest bulbs in Russia.

Leo: Maybe not. But often they do these probes just to see what's possible. Well. It's just fascinating. Are we done?

Steve: We are.

Leo: I think we can call it a day for Security Now!. Oh, no, don't cry, it's okay. It's going to be all right. You know why? Steve's going to be back next week, Episode 806. He's starting to work on it right now, I can tell. We do Security Now! every Tuesday, 1:30 Pacific, 4:30 Eastern, 21:30 UTC. If you want to stop by and watch us do it live, that would be the freshest version, straight, you know, as we're actually creating it. You can go to TWiT.tv/live. There's live audio and video streams of everything we do there, day and night. There's always something going on. If you're watching live, chat live: irc.twit.tv. Join the nice bunch of people who are also watching the show live. On-demand versions of the show are available from Steve's site, GRC.com. He actually has some unique versions, a 16Kb audio version. Does anybody download that anymore?

Steve: Yeah.

Leo: Okay.

Steve: When I, like, make a mistake and fumble finger...

Leo: You hear about it.

Steve: I hear about it. So, yup.

Leo: 16Kb, that's for the bandwidth-impaired. Probably mostly Australians. I don't know, I just feel like they suffer, you know. There's also 64Kb audio, which is the standard. There's even a transcript, which is written by Elaine Farris. It's a human-written transcript, very useful. You can read along. My son is going through all the Security Now! episodes now looking for any information about my bitcoin password. It drives him nuts because, as bitcoin goes up, it's now - I probably shouldn't tell you, Steve - $47,000.

Steve: Oh.

Leo: And I said, "It's making you crazy, isn't it. Isn't it." "Yes," he says. "It's making me crazy." Anyway, but that's a good thing. The transcripts let you search, it's a text file, and then jump right to that part of Security Now!, so very, very useful. While you're there, by the way, pick up SpinRite. 6.0's the current version, but you will automatically be upgraded to 6.1 the minute it comes out. You can also participate in the ongoing development of that in the forums and so forth. So it's really a good time to get SpinRite, if you for some reason haven't yet purchased the world's best hard drive maintenance and recovery utility.

We have 64Kb audio and video on our site, TWiT.tv/sn. It's available on YouTube. There's an entire dedicated YouTube channel for Security Now!. You can subscribe there. Actually, the best subscription I think would be in your favorite podcast player because then it would download it automatically and have it ready for you the minute you get in the car or you go to bed or wherever you listen to your podcasts. So do subscribe. I think it's a good idea. Steve will be back next week. So will I. I'm sure there'll be something to talk about. We haven't run out yet.

Steve: I just checked. 341 downloads of Episode 802. So, yeah.

Leo: There you go. Thank you, everybody. Glad you like the 16Kb version. See, it's worth it. All right, Steve. Have a great week, and I'll see you next week.

Steve: Thank you, buddy. Right-o.