Source Port Randomization

Description: This week we look at a massive customer information leak from a surprising source. Meta notes where their users are being harvested. And in an industry first, Uber's CSO has been convicted. We have more, much more, cryptocurrency industry turmoil. A new appointee in the U.K. wants to drop their use of the GDPR. The NSA is looking for next summer interns, IBM learns that incident responders are feeling quite stressed out, and Microsoft continues to fumble their Exchange Server response. I have news of SpinRite and of my discovery of a lovely little Single Board Computer. And after sharing some listener feedback, we're going to look at a recent mistake made in the Linux kernel that allowed its users to be tracking online.

High quality (64 kbps) mp3 audio file URL: http://media.GRC.com/sn/SN-892.mp3
Quarter size (16 kbps) mp3 audio file URL: http://media.GRC.com/sn/sn-892-lq.mp3

SHOW TEASE: It's time for Security Now!. Steve Gibson is here. Boy, he has a lot to talk about. A new job offer from the NSA. You won't believe the meme they chose. We'll also talk about Uber's CSO. He's been convicted of a heinous crime. You won't believe it when you hear it. And then he's going to talk about his discovery of a lovely little single-board computer I know many of you are going to want to buy. Plus a look at source port randomization. It's a big show, and it's all ahead, next on Security Now!.

Leo Laporte: This is Security Now! with Steve Gibson, Episode 892, recorded Tuesday, October 11th, 2022: Source Port Randomization.

It's time for Security Now! with this guy right here, Steve Gibson, the star of our show. Hello, Steve.

Steve Gibson: Yo, Leo. I've silenced my phone so we won't be interrupted by...

Leo: No yabba-dabba-dos?

Steve: No yabba-dabba-dos. I'm looking forward to the day when I have to shut those off because they're so annoying.

Leo: Every five seconds, another SpinRite.

Steve: Not there yet, but I'm working in that direction. I have some news on that front that I'll share later. So we're Episode 892 for October 11th. And I titled this one "Source Port Randomization," which is a subject we've spoken of often. But it came up again as a consequence of a mistake that the authors of the Linux kernel recently fell into. So I think that's going to be interesting. We have, I know that we have a large Linux following among our listeners, the techie most of ours. So we're going to look at first a massive customer information leak which arose from a surprising source. Also Meta notes that they did some analysis to discover where their users' credentials are being most harvested. And in a weird industry first, Uber's ex-CSO has been convicted of some interesting misbehavior.

Leo: Ooh, yeah. Oh, no kidding.

Steve: Yeah.

Leo: We talked about that when it happened, I think. When he got caught, anyway.

Steve: Yes, yes. And that was two years ago that the allegations were made, and the indictment happened last week. So we have more, much more - I mean, the outcome of the trial, rather, he was found guilty. We'll talk about that. We've got more, much more, cryptocurrency industry turmoil, which just...

Leo: Nonstop.

Steve: Oh, my god, Leo. But, I mean, and just it's like it's creative turmoil. It's like, what? We also have a new appointee in the U.K. a month ago who has decided that she wants to drop the U.K.'s use of the GDPR. Oh, also the NSA is looking for next summer interns. I'll provide information for our listeners who might be interested in signing up. IBM has learned that incident responders are feeling quite stressed out. And Microsoft continues to fumble their Exchange Server response to the most recent Exchange Server problems that we started talking about last week. As I mentioned, I've got news of SpinRite. And I'm going to share my discovery of a lovely little single-board computer, basically Steve's Dream SBC. And then after sharing some listener feedback, as I said, we're going to look at a recent mistake made in the Linux kernel that allowed its users to be tracked online.

Leo: Oh. I thought you were going to talk about the recent mistake in the Linux kernel that would fry some users' monitors. Did you see that?

Steve: No.

Leo: Yeah, they were warning people not to download the new kernel.

Steve: Wow.

Leo: I think they've fixed it. But, yeah, because something, I mean, that's amazing that you could do something in software that would destroy hardware.

Steve: Back in the CRT days it was possible to mess up their H and V sync in a way that would actually damage the circuitry.

Leo: Right.

Steve: But yeah, in an LCD world, that's really interesting.

Leo: I'll look into it. I'll find out what it is, and I'll let you know. Steve Gibson has the Picture of the Week.

Steve: This just kind of cracked me up. For those who are not seeing the show notes or the video, what we have is a Tesla, clearly recognizable by everyone, that apparently doesn't have much faith in its ability to find the next charging station. And so I titled this "DIY Hybrid?" Strapped to the back of it in what looks like sort of a permanent installation, it's got a gas-powered electric generator and a bunch of gas cans. So I guess if the battery runs low, this thing being sort of a DIY hybrid, you'd just cruise off to the side of the road, gas up the generator, plug your car into itself, into this little caboose that it's got, and charge her back up. And then you're ready to go again. So anyway, not specifically anything about security, but I just thought this was kind of humorous. So, yeah, interesting.

The first piece of our show was a tweet that came from our last week's topic source. Remember Jacopo Tediosi. He was one of the two Italian researchers who discovered the serious Akamai vulnerability. Anyway, he knew about the podcast. And he said: "Thanks @SGgrc for talking about my Akamai vulnerability on the Security Now! podcast." And he gave a link to it at TWiT.tv.

Leo: I saw that tweet. It was so cool.

Steve: Yeah, it was very cool. And he said: "The analysis and explanations you made were very accurate!" So anyway, Jacopo, thank you for following up.

Leo: He didn't call you out on mispronouncing his name, though.

Steve: And actually I replied to him politely, thanked him for his tweet, and said I hope I didn't mangle the pronunciation of your name too severely. So, right.

Leo: He's probably used to it.

Steve: Oh, goodness. So it turns out that there's a non-security breach way, like a means, for a user of a cryptocurrency exchange to have their name, their account balance, and all of their transactions exposed to the public. And that's if the currency exchange files for bankruptcy. Whoops.

Something known as the Celsius Network cryptocurrency platform deliberately exposed the names and complete transaction histories of hundreds of thousands of its customers. Okay, now, timeout. As an aside, "hundreds of thousands of its customers"? Leo, what most mystifies me is how these random also-ran startups acquire hundreds of thousands of customers. What are people thinking? Who are these people? And, you know, it's just a mystery.

Anyway, the company filed a - get this - 14,532-page document, because of course lots of transactions for all of its hundreds of thousands of customers, thus requiring a 14,532-page document as part of its bankruptcy proceedings the week before last that contained the names and recent transactions of every user on its platform. The judge in this case, the bankruptcy judge, allowed the company to redact the document, but only their customers' physical and email addresses were allowed to be removed because the rest of the information was required in their disclosure during their regular bankruptcy procedures and proceedings.

So the document, for anyone who's interested, is available via PACER and other legal document portals. So not so private if the cryptocurrency platform that you're using goes belly-up and chooses this means of shutting themselves down. Just something to keep in mind.

A posting last Friday by two security-focused employees of Meta, you know, Facebook's parent, disclosed the results of a recent search through the Apple and Google app stores. They explained that they had identified more than 400 malicious Android and iOS apps targeting Facebook's users which were being used to steal specifically, I mean, the reason these apps were created was to steal their Facebook login credentials. They reported their findings to Apple and Google and have asked the users they identified to change their passwords since their credentials have almost certainly been compromised.

Now, I thought that the nature of the come-ons to entice the downloads of these apps was interesting. So they were, first of all, majority were photo editors, including those that claim to allow you to "turn yourself into a cartoon." Those apparently are very popular among Facebook users. Also we had VPNs claiming to boost browsing speed or grant access to blocked content or websites. In other words, like solving a problem that people have. Oh, this will make your browsing twice as fast, whatever. Probably not true, but it got them to download. Also phone utilities such as flashlight apps that claim to brighten your phone's flashlight. Yes, get more light out of your flashlight if you download this phone utility.

Then we had mobile games which were falsely promising higher quality 3D graphics. Health and lifestyle apps, you know, horoscopes and fitness trackers. Business or ad management apps claiming to provide hidden or unauthorized features not found in official apps which are being offered by the tech platforms. So interestingly, by far the majority at nearly half, 42.6%, of those 400-plus apps were all photo editors. Very popular. Next, dropping down to 15.4%, were the business utilities, then the phone utilities at 14.1%, and games at 11.7%, and the others making up the result.

So, you know, standard advice applies. First of all, try hard, I mean, really try to avoid downloading just every tasty-looking goody that you see. We've said this before, and I think it's still true. There is a very small probability, you know, given the vast number of good apps that are out there, the probability is diminishingly small that something next, something more that you download will be malicious. But the probability is not zero. So if you can, and you care about not having malware running in your device, don't do it.

And don't be too quick to click the download link. Do as much research about the app and its reputation as possible. And I would suggest you do that off the platform. That is, go elsewhere. Don't rely on that in-place reputation because the other thing these guys

are known to do is to load themselves up with faked five-star ratings and thumbs up and things. So look elsewhere for other sources of reputation. So, you know, again, be careful. Be cautious. But just know, as we've noted before, that some percentage of these things in the app stores, much as Apple and Google both are trying diligently to keep these things clean and scrubbed, they exist for a while on this app platform.

Okay. Uber's former CSO, their Chief Security Officer, by the name of Joe Sullivan, was found guilty at trial due to his actions following a 2016 data breach at Uber. And I'm wording this carefully because there was actually some misreporting about this in the press. It's like, the implication was, for reporters who were not being careful, that Joe was responsible for the breach. Not at all the case. Right? I mean, he's a C-Suite guy. Those guys don't get their hands dirty. Anyway, we'll get to that in a second.

Reading from a statement made on August 20th, 2020 - so as I said, two years ago when these charges were filed in the Northern District of California - they put out a statement about the fact that this was being done. They said: "The complaint describes how Sullivan played a pivotal role in responding to Federal Trade Commission (FTC) inquiries about Uber's cybersecurity. Uber had been hacked in September 2014" - okay, so that's a different - that's two years before, a different instance - "had been hacked in September 2014, and the FTC was gathering information about that 2014 breach. The FTC demanded responses to written questions and required Uber to designate an officer to provide testimony under oath on a variety of topics.

"Sullivan assisted in the preparation of Uber's responses to the written questions and was designated to provide sworn testimony on a variety of issues. On November 14th, 2016" - so near the end of 2016 - "approximately 10 days after providing his testimony to the FTC, Sullivan received an email from a hacker informing him that Uber had been breached again. Sullivan's team was able to confirm the breach within 24 hours of his receipt of the email.

"Rather than report the 2016 breach, Sullivan allegedly took deliberate steps to prevent" - and this is allegedly because it's two years ago. Now it's been found to be true at trial - "allegedly took deliberate steps to prevent knowledge of the breach from reaching the FTC." So he tried to bury this. "For example," they said, "Sullivan sought to pay the hackers off by funneling the payoff through a bug bounty program. Uber paid the hackers $100,000 in Bitcoin in December of 2016, despite the fact that the hackers refused to provide their true names.

"In addition, Sullivan sought to have the hackers sign non-disclosure agreements. The agreements contained a false representation that the hackers did not take or store any data, when in fact they had. When an Uber employee asked Sullivan about this false promise, which was in the nondisclosure, Sullivan insisted that the language stay in the nondisclosure agreements. Moreover, after Uber personnel were able to identify two of the individuals responsible for the breach, Sullivan arranged for the hackers to sign fresh copies of the non-disclosure agreements..."

Leo: I just love that piece. Don't tell anybody.

Steve: I know. Yeah, don't tell anybody what happened. And oh, by the way, now that we know who you are, we want you to actually execute the nondisclosure agreements which would be binding under your true names.

Leo: And they gave them a bug bounty.

Steve: Yes, yes, yeah. We're just going to say that you found a problem, not that you actually attacked us using it.

Leo: Appalling.

Steve: Yes. So "The new agreements retained the false condition that no data had been obtained. Uber's new management ultimately discovered the truth and disclosed the breach publicly, and to the FTC nearly a year later, in November of 2017. Since that time, Uber has responded to additional government inquiries." So all of that was proper.

"The criminal complaint against Sullivan alleges Sullivan deceived Uber's new management team about the 2016 breach. Specifically, Sullivan failed to provide the new management team with critical details about the breach. In August of 2017, Uber named a new Chief Executive Officer, a new CEO. In September of 2017, Sullivan briefed Uber's new CEO about the 2016 incident by email. Sullivan asked his team to prepare a summary of the incident. But after he received their draft summary, he edited it. His edits removed details about the data that the hackers had taken and falsely stated that payment had been made only after the hackers had been identified." That wasn't the case.

The two hackers identified by Uber were prosecuted in the Northern District of California. Both pleaded guilty on October 30th, 2019, to computer fraud conspiracy charges, and now await sentencing. The criminal complaint makes clear that "both hackers chose to target and successfully hack other technology companies and their users' data after Sullivan failed to bring the Uber breach to the attention of law enforcement." In other words, by not dealing with law enforcement forthrightly, the hackers, who had been identified, continued to roam free to hack and damage other companies as a direct consequence of Sullivan's actions of covering all this up.

So at trial Sullivan was found guilty of lying to authorities and obstruction of justice. Those were the charges, lying to authorities and obstruction of justice. Nothing to do directly with Uber being hacked. It's like, you know, we know that kind of thing happens. The trial, however, was a landmark case, being the first time a Chief Security Officer faced criminal charges, indirectly at least, relating to a security breach. Though it was only, you know, obviously, indirectly about the breach itself. Joe's big mistake was his attempt to cover-up and mislead investigators that ultimately landed him, as we know now, in some very hot water. Interestingly, Joe was once a prosecutor in the same office that had charged him.

Leo: I didn't know that.

Steve: Yeah.

Leo: He should have known better.

Steve: Well, I was thinking that maybe he thought he knew how to finesse the system. Right, like having once worked there, he figured, hey, I know how to get around this.

Leo: Oh, gee.

Steve: So he now faces up to eight years in prison and up to half a million dollars in fines, which will be determined at his upcoming sentencing hearing. What has never been made clear in the reporting that I've seen is why he did this. He was a C-level executive for a major corporation, Uber. Guys at that level aren't pulling wires and getting their hands dirty. They attend meetings, and golf. So it was almost certainly not directly Joe's fault that somewhere in a back room two attackers somehow crawled into Uber's network. What I wonder is whether he had a big hunk of Uber stock that he worried would collapse in value if the news of this got out. If so, perhaps he believed that he could cover the whole thing up from the top to protect Uber's market value. In any event, I imagine that he regrets that decision now.

Leo: Can't spend that stock in prison. No, you can't.

Steve: No. So more cryptocurrency chaos. I believe that this podcast's listeners would be well served for me to periodically note the ongoing chaos that exists within the cryptocurrency world. It's not my position to advise anyone of anything. But being armed with realistic viewpoints can only be valuable.

To that end, the news is that the multi-cryptocurrency exchange platform Binance was hacked. Binance has paused its Binance Smart Chain, BSC as they call it, blockchain bridge after a threat actor used an exploit there to generate and steal 2 million Binance coins. The abbreviation for that currency is BNB. They are currently worth around $560 million. Now, the thieves were unable to make off with all $560 million because Binance reacted quickly to what they discovered. But the bad guys still absconded with 20% of the $560 million in illegitimately created funds. So $112 million worth of the Binance coins. So not bad for a day's work.

Leo: We were talking about this on Sunday on TWiT. And apparently the bridge software, which is what allows you to move crypto from one place to another...

Steve: Chain, right.

Leo: ...is a very common source of hacks. And this is the fourth or fifth massive hack of a crypto bridge of some kind in the last couple of years. It's a huge vulnerability. And of course that's where the money is. It's like you can tap into the oil pipeline and just say, yeah, give me some of that.

Steve: Yeah.

Leo: You know, it's - wow.

Steve: Okay. And so while we're on the subject of bad ideas, I'll also note that the Zcash blockchain has been subjected to a spam attack. Yes, spam isn't just for email anymore. This was done by creating bloated but inexpensive "shielded transactions" on the Zcash blockchain. And as a consequence of this attack, which has been underway since June, the size of the Zcash blockchain has more than tripled to over 100GB. As the Zcash blockchain has grown huge, purely as a result of bogus transactions...

Leo: They have to store every one of these tiny transactions.

Steve: Yes. The cryptocurrency experts now expect Zcash node servers, which must retain a full local copy of the entire blockchain, to start failing due to memory shortage.

So, okay. All of this points - the only way that you can regard this is to an extremely immature technology coupled with a gold rush attitude. Recall that in the actual California Gold Rush, between 1848 and 1855, with very few exceptions, the only people who made money were those who were selling the gold digging, panning, and sluicing supplies to the hopeful miners. It wasn't those who were panning for gold. It was the people who sold them their pans.

Leo: No, in fact I saw just the other day at auction a pair of Levis found in an old coal mine from the 1880s sold for $76,000. So they're still making money.

Steve: Still making money. And, you know, you can't get graphics cards anymore, right, because all of the mining rigs have sucked up all the GPUs everywhere.

Leo: Well, that's the problem is now you can because there's no more money to be made because of proof of work, proof of stake.

Steve: Right.

Leo: So now you can get a lot of highly used GPUs. They're flooding the market.

Steve: I think this would be a good time to take a break, and then we're going to talk about the U.K.'s plans to drop out of the GDPR.

Leo: And actually, you know, in conjunction with NFTs, I've been saying this for a while, the companies that make money in NFTs are the people minting them. They're collecting the...

Steve: Well, look at Kevin.

Leo: Yeah. They're collecting the gas fees. Well, Kevin made $50 million, his proof collective. And then he raised another 50 million because it's such a good...

Steve: Leo, where? Where is this money? Who are these people? Is there something [crosstalk] something somewhere.

Leo: I hope, I sincerely hope that for the most part they're bitcoin bros. They're crypto millionaires and billionaires who are reinvesting.

Steve: Like the Winkeldingies or whoever they were down in...

Leo: The Winkeldingies. And I pray it's them and not some poor working stiff who says, yeah, my stocks aren't doing so well. Maybe I'll get into this crypto thing. But unfortunately, you know, because Robinhood and all these other easy trading apps sell crypto now, I suspect that a lot of this is coming from people who can't afford the losses. And it's very sad, you know. Of course my stock market portfolio has tumbled even more. So maybe I should buy some bitcoin. A little doge. Who knows?

Steve: You know, I figured I was probably an old relic. So I just looked up the definition. It says "An object surviving from an earlier time, especially one of historical or sentimental interest."

Leo: Bingo.

Steve: And I think that works.

Leo: Bingo.

Steve: I'm the old relic.

Leo: I'm getting ready for - I love the Advent of Code coding competition in December. I'm getting ready for it. I'm trying to do it in Lisp. And I'm sitting there...

Steve: Wait, I thought you were going to do a new language.

Leo: I was looking at Julia, but you convinced me anything that starts, indexes arrays at one...

Steve: Oh, yeah.

Leo: No, unh-unh.

Steve: I know.

Leo: I really love Lisp. So I'm on Day 7 of the first year, 2015, and I need to do bitwise operations. And I'm thinking, if I were Steve, this would be easy. He lives in the pits. But now I've got integers, and I've got to figure out - and actually it's not a problem except when I get to the twos complement representation. I'm getting numbers going negative, and I've got to figure out a way to just ignore that.

Steve: There are.

Leo: You, you wouldn't have to think about it. Yeah, of course there are.

Steve: There are an amazing number of really cool bitwise hacks.

Leo: I know. In fact, I have a book that's almost entirely bitwise hacks. I was going to ask you this because I want to give it to you. Have you ever read "Hacker's Delight"?

Steve: No.

Leo: Okay. Don't buy it. I am going to send it to you.

Steve: Cool. Cool.

Leo: It is a classic, and it is almost entirely, like, weird bitwise hacks.

Steve: And like sort of edge cases that turned out to be useful.

Leo: Yeah, that's the whole idea is once you know this, the idiom, you'll use it all the time. And you of all people because you work in assembler, this is going to be like, oh, yeah. You probably know 90% of them. But it's a good book. I'm going to send it to you.

Steve: That's fun. Cool. Thank you.

Leo: You're welcome.

Steve: Okay. So I'm not sure whether this is good or bad, though I'm leaning heavily toward bad, for a reason I'll explain. Last Monday, Michelle Donelan, the U.K. Secretary of State for Digital, Culture, Media and Sport - that's literally her title, the U.K. Secretary of State for Digital, Culture, Media and Sport. Luckily, Leo, for your sake they didn't say "cyber."

Leo: Sport's bad enough.

Steve: Sport's bad enough. I agree. That one, I said wait a minute, did I...

Leo: We don't have in this country, we don't have a Minister of Sport. But a surprisingly large number of countries do. And I watched, reason I know is I watch the Formula I races.

Steve: Don't they understand it's plural? It's supposed to be sports.

Leo: And they call it "maths." Maths and sport.

Steve: It's not like I'm playing "card." I'm playing "cards."

Leo: But it's maths, plural. So go figure. What is it...

Steve: Oh, that is true.

Leo: ...Winston Churchill said is America and England, two countries separated by the same language.

Steve: So she was appointed to her position of Digital, Culture, Media and Sport about a month ago. On this last Monday she announced plans for the U.K. to drop the EU's GDPR in favor of designing their own...

Leo: What?

Steve: I know, their own new data protection system. And this is the point where I groan. Michelle was speaking at the Conservative Party Conference in Birmingham where she said that the U.K. government will look to pass new legislation inspired by data protection laws used in Israel, Japan, South Korea, Canada, and New Zealand.

Now, on the one hand, that sounds maybe better than the GDPR. But the concern is that we only have the one single global Internet. And that was the whole point of the Internet in the first place. That's what makes it so useful and amazing. But now governments are getting into the act of deciding how the Internet should uniquely treat each of their own precious citizens, even if that differs from how the Internet treats everyone else. Governments want to have borders, but the Internet was designed to ignore them. And so basically we have a clash of fundamental principles here. And it's going to be a mess, Leo. And so my concern about the U.K. leaving the GDPR is, okay, now we're going to have the UKBR or UKGDP or who knows what. But wow, you know, one more mess to deal with.

Okay. This seems cool. Rob Joyce is the Director of Cybersecurity at - his Twitter handle is @NSAgov. He recently tweeted that the NSA is looking, now looking, for next summer interns. He wrote: "It's never too early to make summer plans. @NSACyber 2023 Summer Internships are open. CompSci" - he's got a number for it - "CompSci 1191813. Cybersecurity 1191816. And Engineering, 1191817." He says, "Apply at intelligencecareers.gov/NSA. Use the numbers above. Find your passion. Hurry, applications close on Halloween." So...

Leo: Appropriate.

Steve: We're in, yeah, we're in the Halloween month. So through the rest of this month, applications are open to apply to the NSA for a summer internship.

Leo: You don't think this is a joke?

Steve: No, I checked. I mean, the picture is weird. Maybe this is how somebody who is not at all cool tries to look like they're cool, I think.

Leo: It's a little creepy. It's a little weird.

Steve: It is right from the Twitter feed. If you go to Twitter and put in, what's his handle again, it's @nsacyber. No, sorry, @nsagov.

Leo: @NSA_CSDirector is the Twitter handle. But you could search for Rob Joyce probably.

Steve: Oh, right. Right, right, right. And actually I did that, and it came right up. So it's like that's actually what he tweeted.

Leo: Wow. It reminds me, you nailed it, it's somebody who is not cool thinking, what would the kids do?

Steve: Exactly.

Leo: What's the meme I could post? And then, you know, it does remind me a little bit of "Goodwill Hunting" and the very famous scene where they ask Matt Damon to apply, I can't remember if it's - I think it is the NSA. If you haven't watched that, that would be a good response.

Steve: Oh, you kidding? It's one of my favorite movies. It's a great movie.

Leo: Yeah. Where he goes - he does a great monologue about, "Well, I'll tell you why I am not going to work for the NSA." It's really good. I'm sure, I tell you what, if you read the thread, the responses for this tweet, it will be in it. I almost am certain. Anyway, that's great.

Steve: Okay. So IBM did a survey. We've talked a lot about the job opportunities available across the security industry. There are many. They are plentiful. And they show no sign of diminishing. I think the stuff we talk about, the needs of the industries we describe are only growing greater. But they can be demanding, and it can interfere with other life priorities. IBM recently conducted a survey of 1,100 professional cyber incident responders. Here are the seven takeaways from the survey.

First, Cybersecurity Incident Responders said that the sense of duty to help and protect others and the businesses was by far the most influential factor attracting them to the profession. Continuous opportunity to learn and being rooted in problem solving followed as the most influential factors. So people who want to help. That's cool.

At the same time, number two, "sense of responsibility toward their team/client" and "managing stakeholder expectations" were ranked as the most stressful aspects of responding to cyber incidents. Around half of the 1,100 selected these among their top

three stressors. So they were stressed by the sense of responsibility toward their team or client, and managing stakeholder expectations, you know, like solving the problem for their bosses.

Third takeaway, according to 48% of responders, the average incident response engagement is two to four weeks. And nearly 30% say an incident response engagement lasts more than four weeks on average. The overwhelming majority states that it's not uncommon to be assigned to respond to two or more incidents that overlap.

Fourth takeaway is that the first three days of responding to an attack are seen as the most stressful. Additionally, more than a third say they are working more than 12 hours a day during the most stressful period of the engagement.

The fifth takeaway is that 81%, so four out of five, a little over four out of five of Cybersecurity Incident Responders think the rise of ransomware - no surprise - has exacerbated the stress/psychological demands required during a cybersecurity incident response. Right? Because their enterprise is frozen and is under threat.

Sixth takeaway, two thirds, 67% of Cybersecurity Incident Responders said they experience stress and anxiety in their daily lives as a result of responding to an incident. And finally, nearly 65%, again, two thirds of Cybersecurity Incident Responders have sought mental health assistance as a result of responding to cybersecurity incidents.

Leo: Wow. Holy cow. Wow.

Steve: Yeah. To that end, the majority of respondents, 84% did also say that they do have access to adequate mental health support resources. So they're able to get help. But, I mean, they're under tremendous pressure, probably not chronically, but acutely, when something happens. So I think this suggests two things. First, cybersecurity incident response may not be for everyone. You know, it's probably, you know, think about your personality type. Do you like adrenalin? Are your adrenals in good shape and providing you with what you need? I think seriously that should be a consideration.

Leo: It must be high pressure. I mean, very intense when it happens. Probably a lot of time sitting around, and then all of a sudden boom.

Steve: Yes. Yes. You know, imagine something gets into your network. And it's doing bad stuff. And it's up to you, I mean, like it's not - it's you, not anybody else. And like alarms are going off, and people's computers are crashing, and it's like, yeah, that's not when you want to slip the blood pressure cuff on your arm to see how you're doing.

The other thought I had was that, although you'll certainly want to be a salaried employee, if there's any way to work in bonuses to your contract, for when the job does disrupt your life, that should be a consideration, too. Being the only one left at work, working all night, while everyone else is home laughing and sleeping, is much easier if you know that your special contribution is being valued with some additional compensation. And if you have a significant other in your life, it can make it easier to explain to them, honey, I'm sorry, but I can't come home. So anyway, I thought that was interesting. It is, you know, we've only talked about all the opportunities so far. And this survey of 1,100 people says, yeah, you know, there's lots of opportunity, but it's not a cakewalk all the time if something nasty crawls into your enterprise that you're responsible for.

Leo: It's probably a lot like being a first responder, a fire fighter, an EMT, a police officer. It's, you know, when things go haywire, they go. And you have to be there. And you probably need a certain kind of constitution; right?

Steve: I think that's the case. I think that's the case.

Leo: Yeah, the right stuff. Just cool under pressure.

Steve: Right. And I'm sure there are people who, like, who thrive on the idea of that much need being piled on them when their shoulders are able to handle the burden.

Leo: Yeah, we have a picture of that person here. The NSA Director's tweet, yeah.

Steve: Yeah. Okay. So speaking of being stressed out, something's going on over at Microsoft, and it's not good. The topic is the status of Microsoft's mitigations for that pair of zero-day Exchange Server vulnerabilities we discussed last week. Those were the new pair discovered while being used in the wild, exploited in the wild, in the networks of clients of that Vietnamese cybersecurity firm GTSC.

First, in updating myself for today's podcast, I checked to see whether patches for these two new bad problems were available. That would be the optimal answer; right? Get it fixed. But after at least a week and a half, and it turns out a lot more than that, as we'll see in a second, the answer to that is no. No emergency patch for Exchange Server so far.

Then, since there was news last week that the initial mitigations proposed by Microsoft had immediately been bypassed, as I noted last week, I wasn't quite sure of some of the language, but Bleeping Computer said - they confirmed, yup, that original mitigation proposed by Microsoft has been bypassed. I went to see what Microsoft had done since then, and they really appear to be chasing their tail. They updated their guidance for scripts for IIS mitigation on October 4th, 5th, 6th, 7th, and 8th. Each time they're correcting typos or making small tweaks to the script, apparently trying to get it right. It's like, one of the comments is "remove the space that was unnecessary."

Leo: That's the least we can do. It's so bad.

Steve: Oh, god. So nothing about this response feels like the A team has been brought in. And then we learn that Microsoft has been aware of this problem for much longer than was previously known. They were, in air quotes, "investigating it" after becoming aware of it back in August. In their posting titled "Analyzing attacks using the Exchange vulnerabilities CVE-2022-41040 and 41082," Microsoft wrote: "MSTIC observed activity related to a single activity group in August 2022 that achieved initial access and compromised Exchange servers by chaining 2022-41040 and 2022-41082 in a small number of targeted attacks." This is in August.

"These attacks," they wrote, "installed the Chopper web shell to facilitate hands-on-keyboard access, which the attackers used to perform Active Directory reconnaissance and data exfiltration." Oh? So apparently nothing to worry about? It's like, what? In August. They said: "Microsoft observed these attacks in fewer than 10 organizations

globally. MSTIC assesses with medium confidence that the single activity group is likely to be a state-sponsored organization."

Then they said: "Microsoft researchers were investigating these attacks to determine if there was a new exploitation vector in Exchange involved" - okay, make that yes - "when," they said, "the Zero Day Initiative (ZDI) disclosed CVE-2022-41040 and 2022-41082 to Microsoft Security Response Center (MSRC) in September of 2022."

So, gee, look at that. Exchange Server is being attacked. Hmm. What's for lunch? Unbelievable. They were investigating. While attacks were underway, 10 organizations they had identified, and now we're, what, we don't know when in August, so somewhere between 1.5 and 2.5 months downstream of this, of them seeing that Exchange Server is being exploited by a remote execution exploit which is taking companies over and allowing bad guys to perform reconnaissance on enterprises' Active Directory servers. And Microsoft is updating their advice after this became public, every day removing extraneous spaces from their scripts. Not impressive.

In SpinRite news, I have finished all of the redesign, and SpinRite is working as far as I know. But that knowledge doesn't yet go very far. So now I start the final work of inducing known data errors and watching SpinRite perform its sector-by-sector data recovery. That's what I'll be working on tonight and subsequently until I've demonstrated to myself that SpinRite is in fact ready.

Leo: That's interesting. How do you make data errors happen?

Steve: It turns out in older drives, and I have a 2TB, I have three actually 2TB Seagates and a bunch of older Maxtors. You had the ability there, there was a command called READ LONG where you told the drive don't bother with error correction, just give me the raw data. It's called a "long read" because it's the data plus the error correction code which is tagged on at the end. And that facility is one of the ways that SpinRite is able to perform data recovery, even on sectors that the drive says are not good. Well, even though they're not good, there's still something there. SpinRite is able to say, give me what you've got, and let me worry about it. And that's where this Dynastat, the Dynamic Statistics comes in.

Leo: So do you go to a known bad sector and then ask to do a long read?

Steve: No. Because there's the complement command, WRITE LONG.

Leo: Ah. So you can screw it up.

Steve: Where I'm able to - yes. Exactly. I'm able to induce varying length bit errors and cause the drive to do what it's going to do when it encounters it for the first time. And so that allows me to deliberately poison sectors in various ways and then, as if that hadn't happened, have SpinRite come along and watch what it does in order to recover the data. So it's very cool. Unfortunately, it's been - those things have been removed from newer drives. But I've got plenty of older drives where it's still feasible. And even a 2TB drive. So I've got lots of ability to do that.

Leo: That's really interesting. I didn't know that.

Steve: Yeah, it's really neat. Okay. So once I know that it is doing what I want it to do, I'll release that to GRC's newsgroup gang, and we'll find, I'm sure, the various things that I've missed. Cosmetic things, logging things, who knows what. I'll get those fixed, and we'll move SpinRite from Alpha to Beta. When we're there, anyone who owns SpinRite will be able to download the DOS executable that I've been developing, and which everyone's been testing. Since I won't yet have it packaged as a turnkey Windows app, you'll need to use GRC's InitDisk or ReadSpeed or arrange to boot your own DOS however you want to. And then you'll be able to run that DOS executable, and it would be the real SpinRite 6.1, just as it's finally going to be.

Something else happened last week that was interesting. Although we won't get to high-speed native USB support for SpinRite until somewhere in v7, I'm thinking 7.1, only because I don't want to delay 7.0, which will be the first SpinRite ever to be bootable over UEFI. I want to get that out as quickly as possible. And there's no reason to hold it because 7.1 will be free for everybody anyway. So I designed SpinRite 6.1, today's forthcoming SpinRite, to work with any size USB drive through the motherboard's BIOS as it always has, but now with no size limitations, if the motherboard supports it, and many do.

But it occurred to me that I had never explicitly asked any of our testers to try attaching a huge drive, larger than 2.2TB, which is the largest drive that's addressable with 32 bits. You've got to have more than 32 bits to go beyond 2.2TB. That's why the old-style master boot record, the MBR, that only has 32-bit size fields in its definition, which is why you can't use an MBR on a drive greater than 2.2TB. Anyway, I had never asked them to try to attach a huge external drive to a USB port to confirm whether SpinRite sees the large drive and can indeed work with it. That is, today, the SpinRite 6.1 we're going to be getting. And it turns out we learned that it does, and it can.

In the show notes, for any prospective SpinRite 6.1 owners, I've got some screen shots which our testers, which two of our testers provided. And I'm looking at them. And thank you, Leo, for them being onscreen now. Highlighted at the bottom of a list of different drives on this one person's system, we see a 4.0TB drive, which will be interfaced through the BIOS. You notice the first five drives are AHCI, so that's SpinRite's direct hardware access to the AHCI chip itself, which it now knows how to talk to. So, but then the lower four drives have been connected to USB ports through the BIOS. So a 4TB drive. And SpinRite through the BIOS will be able to scan that drive in 30.1 hours.

Then in the next slide he's run the benchmark, the full benchmark on that drive, which allows it to perform a finer grained performance analysis. And so SpinRite's estimation was revised to 29.45 hours. And we can see the various speeds at which SpinRite can talk to that 4TB drive. So while 30 hours is not fast, it used to be 30 months for a drive this size.

Leo: That's a lot faster. Wow.

Steve: We're doing way, way better, way better than we used to. And in fact, on the next screen, on the next page is somebody else who provided a snapshot. They show a 3TB drive that SpinRite estimates it will scan in 10.1 hours. So that's way faster. So the lesson there is it is a function of the BIOS. Both BIOSes allow SpinRite to see a drive larger than 2.2TB, in one case 4TB and the other 3TB. But this second one can do a 3TB drive in 10 hours, which is a lot faster than a 4TB drive in 30 hours. So your speed will vary. And in fact we do see on that second, that second slide, he had a 1TB drive attached through AHCI.

So it was a SATA drive where we see it doing 1TB in 30 minutes. And that exactly corresponds to my estimate. Remember that I expected that we would - my newer estimation was that SpinRite would probably be able to do 2TB per hour. And that's what we're seeing consistently, 2TB per hour or, in this case, here it is showing 1TB in half an hour. So SpinRite 6.1 will finally be easy and practical to use again on today's very large drives. And even before we get to 7, SpinRite 7 and 7.1, where we've got hardware support for USB, that's where we'll be able to run USB-3 at the same speed as SATA because it is as fast. At that point external drives will be able to run as fast as internal drives. Anyway, I'm having a ball.

Leo: Yeah, that's a huge improvement. And it was a disadvantage using SpinRite on a giant drive was it would take forever.

Steve: Oh, yeah, it wasn't practical.

Leo: But a day, a day is not, I mean, a day and a half is totally doable, totally.

Steve: Yeah, and you can certainly like run it over the weekend, if you had to, yeah.

Leo: Yeah, exactly, yeah.

Steve: Yeah. Yeah, we're getting there.

Leo: Yay.

Steve: Okay. Now, I want to take a moment to talk about a beautiful little affordable $120 plus shipping Single-Board Computer that I am starting to use as of Sunday, last Sunday, which I'll be using for SpinRite's development going forward. It's called ZimaBoard, Z-I-M-A-B-O-A-R-D. And in many ways it's the perfect little platform for SpinRite. I'll get to that in a second. To get SpinRite to the point where it is today, which is its ability to talk directly to any and all PC hardware owned by every single one of our hundreds of SpinRite development testers, and I should note we currently have 367 registered testers in GRC's GitLab instance. So that's the population of people who have been testing SpinRite so far.

I have been gladly purchasing enumerable old motherboards and drives from eBay. This has been going on for the last year. When I've been unable to duplicate some obscure problem that any of our hundreds of testers were experiencing out in the field, buying what they had was often the only way to get to the bottom of some really bizarre behavior. So that's what I would do. But that's all now behind us, at least until SpinRite starts being used by its entire owner base. I do fully expect that I will encounter some new mysteries, and I will deal with those as they come along. But that's, you know, that's the nature of bypassing the BIOS. Now that we're talking to the hardware, there's obscure hardware out there. But, boy, I've seen a lot of it. I think it's clear that we've reached the 99.999% point.

So it's time for the next stage. What I wanted going forward was a completely silent testing platform. And this little ZimaBoard looks perfect for that. No more incessant whirring fan noise while I'm trying to focus. The ZimaBoard is fanless, with a custom heat

sink fin design and just the right number of ports and expandability. It started out on Kickstarter, where it was 4,905% overfunded. In other words, more than...

Leo: There's a market.

Steve: More than 49, yes, more than 49 times the number of project backers that they were hoping for. People went nuts over it. And it's now a going commercial concern. Through the years the recurring question that we've been asked over and over is what GRC would recommend as a perfect PC platform for running SpinRite on a drive.

Leo: Yeah.

Steve: In lieu of dedicating someone's main machine to that task.

Leo: A desktop, yeah. And I've found a few desktops, the ones I own, which won't work at all with SpinRite.

Steve: Right.

Leo: So the UEFI, I guess.

Steve: Yes, and it won't be until 7 that we're able to run there.

Leo: So this is an answer to a question I've been meaning to ask you. This is great. If you were going to do, if you want to run SpinRite and, you know, you do it enough, it's worth spending 119 bucks to get a little machine to do it.

Steve: Yes. And many people also have inventories of drives, like drives that they've taken out of service. And so this allows you to run SpinRite at absolute full-on speed without tying up any of your other resources, as you said, for $119.

Leo: Do you put Windows on it, or you just run FreeDOS on it?

Steve: So, I will get there.

Leo: Okay.

Steve: So years ago, when I was writing the TechTalk column for InfoWorld magazine, I stumbled upon a wonderful motherboard, the ultimate keyboard, an RLL controller, and MFM drives that worked perfectly under RLL encoding. So I conceived of something I called "Steve's Dream Machine." It was a hit with my column's readers. A PC supplier, Northgate Computer Systems, took up the idea of purchasing and bundling all the

components and offering them as "Steve's Dream Machine." What I think I've found here with this ZimaBoard is Steve's Dream SBC - Single-Board Computer.

It is 100% Intel chipset with the exception of its dual gig network adapter, which is a Realtek 8168 chip. Now, it turns out that's perfect for my development needs since I have DOS network drivers for that chip. It has a pair of 6Gb SATA 3 connectors with a cable to provide power for one drive. But for $4 you can get a dual power cable. It has a pair of USB 3.0 ports. So SpinRite will be able to run drives attached to either SATA or USB 3. And it has a single PCIe x 4 connector for the expansion of anything else. That could be a PCIe to IDE adapter, if SpinRite needed to repair any older IDE drives; or an NVMe adapter, if SpinRite needed to be run on NVMe drives once they are supported, and they will be under v7. It has built-in video through a mini DisplayPort which can do 4K video at 60Hz.

And critically, the ZimaBoard offers both UEFI and traditional BIOS support. It has a very comfortable Award BIOS with all of the bells and whistles, you know, drive boot order and so forth, everything that old DOS hands are hoping to see, so that SpinRite will be able to boot FreeDOS and run without trouble. It could boot from an attached USB thumb drive, and I've done that, if you wanted to leave the Debian-derived CasaOS Linux that's shipped with the board in place; or FreeDOS and SpinRite could be installed onto the board's built-in 16GB eMMC drive. That's what I'll be doing. Either way, I'll be able to use the same platform for SpinRite's future development under UEFI. So it's perfect for both now and for what's next.

There are three ZimaBoard models which vary in speed and size, but the smallest of the three is what I purchased because it's enough for doing stuff with DOS. I have two of them now, one for each of my locations. As I mentioned, the smallest of the three contains a 16GB eMMC drive which is preloaded with a Debian Linux variant which they call CasaOS. The board is broadly compatible, able to run any Intel OS, Linux, Windows, pfSense, OpenWRT, NAS software, and anything else. And they sort of have it targeted at your own cloud or multi-drive NASes and so forth.

If you go to ZimaBoard.com, if you click on the "Order Now" button on the home page, and then again on the page that comes up, you'll get to the place where you set the quantity and the model number you want. If you scroll down that third page to the bottom, you'll find a "Buy One Get One Free" offer that explains, well, it's not another free ZimaBoard, but it's a free power adapter. They say: "Buy ZimaBoard and get a free 12V/3A Power Adapter."

Leo: Oh, that's what - you need that; right? Yeah.

Steve: Which you need that anyway, and that saves you 12 or $15 or something. So that's what I would recommend. There's a 10% off discount coupon available, but you probably can't use both. As I mentioned, the ZimaBoard comes with cabling to supply power to a single SATA drive. But there's an optional dual SATA power cabling for $4, actually it's $3.90, that you may want if you intend to power two SATA drives from the SBC. And that's also what I'm doing.

So anyway, I now have a terrific answer to the often-asked question, "What does GRC recommend for running SpinRite standalone." I don't think you can do better than that. I mean, I've been using it. It's just beautiful. You'll need a mini display port cable to a display port monitor. And then the way they have it, the way they suggest you set it up is you plug it into your router, and then you use a browser to talk to it. So I guess it boots up with a, you know, it boots up this Debian Linux variant with a web server running and waiting to be connected, and then brings up some sort of a UI. I didn't do

any of that. I just blew it off and used fdisk to zero the partition and made a bootable DOS because that's what I'll be using. But there is much more for anybody who's interested. So it's just, you know, it's a beautiful solution for SpinRite and other things that I thought our listeners would find interesting.

Leo: I want to hear all about address, random address access.

Steve: Yes. Got some feedback to share first.

Leo: Oh, okay.

Steve: Yeah. One of our listeners, ZendoDeb, said something that I thought was brilliant. He said: "@SGgrc re CAPTCHA discussion from Security Now 891." That was last week. He said: "I've wondered if using Firefox makes it worse, since Firefox is now stove-piping cookies, especially third-party cookies. So when you show up at a new site, Google can't find a cookie."

Leo: Ohhh.

Steve: That is brilliant. Brilliant.

Leo: That's what's going on.

Steve: Exactly. Exactly. There's no question. That's why we Firefox users are saying, hey, why am I having to click on chickens constantly, or fuzzy bears or whatever it is. Yeah, it is a consequence of the fact that Google is highly ranking the presence of their own cookie as one of the signals that they're using. And when you go to a site that you haven't been to before, there's no Google cookie there, thanks to the per-site stove-piping that Firefox is now doing. So very, very clever observation. Thank you, Zendo.

RobinR said: "Hi, Steve. With all these buffer overflow and use-after-free issues, I've seen talk of getting development to switch to Rust. My question to you is what kind of concerns or defensive techniques do you do when developing in assembly? Is it the fact that you are so low level you are forced to be aware of everything, and thus don't fall into the same traps? Additionally, would you change anything with a piece of software that you knew would be always on and be available on the Internet?"

And I thought about this for a while. So first of all, I do have a piece of software which is always on and available on the Internet. And that's GRC's server. It is laced with a lot of my project code. ShieldsUP! itself. Probably the most complex, asynchronous thing I've written which is always online is the DNS spoofability test. That thing has all kinds of asynchronous queries off to individual servers as it discovers them, lots of things happening dynamically. I have the same problems that anybody writing in C would have, which is to do that I create a linked list of tasks, and each of the objects that are pointed to in the linked list is a structure which I allocate in RAM which contains the details of where that task is and what's going on.

Those have dynamically created lists of outstanding queries and their responses. I don't know how many there will be, so that's a list. So it's an extremely dynamic construction.

And it's been running for many, many years, and it's never had a bug or crashed. So I think the advantage I have is I'm first of all one developer, so I don't have a problem explaining anything to myself. And while there's a lot going on, it's still not nearly as complicated as what has happened to today's browsers, which are just like, I don't know if there's any one person whose single mind is able to encompass the entire thing. And the same is certainly true for operating systems. So I am at a low level. I'm essentially at the level that C operates because all the things I just described is exactly how I would code something were I coding in C. There's not that big a difference.

Leo: So you do, in assembler, you're doing effectively your own malloc. You've allocating memory. You have to remember...

Steve: And I'm doing reference. And I'm doing my own reference counting. Yeah.

Leo: You do your own garbage collection, in other words.

Steve: Yup.

Leo: You don't need - and you don't, you know, the off by one problem probably is a little bit less of a problem for you because you're so intimately connected with what's going on. I think some of the problems that come from high-level languages is programs are so insulated from what's going on that they can make - it's easy for them to make a mistake.

Steve: Well, and we talked, for example, about Microsoft's decision to use Electron as their platform for implementing Teams. The problem is that that's JavaScript, HTML, and CSS. You don't have to be a power coder in order for something to look like it's working in JavaScript. And so exactly as you say, Leo, I think that does tend to admit less capable or less rigorous programmers. The lower level the language, the more careful you need to be, or it's very obvious that, you know...

Leo: Yeah, you see what's going on, yeah.

Steve: ...when something's going to - yes.

Leo: And I think also the reason they're talking about Rust is Rust is - it is garbage collecting, but it's very tight, there's very tight constrained, you know, it's a static type system, and it really tries very hard to keep you from making mistakes. Every time we see languages like that, like Ada, I think programmers appreciate it, but also don't like to use it.

Steve: Yeah, exactly. I mean, yeah. They're like nanny languages.

Leo: Yeah.

Steve: It's like, okay, well, yeah, you know, that's a language. But, boy, I don't want to code in that. That's, you know, no fun.

Leo: Rust is impressive. And I guess what it's replacing, which is mostly C and to some degree C++, is bad enough. So I guess people who use Rust like it. And I played with it a little bit. It's very impressive. But there's a lot of boilerplate, a lot of extra code. It's like Java a little bit in that respect. Somebody like you, and to some degree me, I don't want to spend a lot of time typing in all that crap. I just want to...

Steve: And I think what's going to happen is we'll get to the point where coders will not be given a choice. That is, what we see happening is that we're getting to the point where we've got all the processing power we need. It used to be that we didn't have enough RAM, and we didn't have enough speed, to support the...

Leo: Overhead.

Steve: Okay, yeah, right, exactly, to support the overhead of sophisticated languages that do a lot to protect the way they're operating. Today we do. And I think at some point there will be a browser that bites the bullet and says, we're coding everything in Rust because we're done with use-after-free errors, period. And we don't care if you don't like it.

Leo: Remember, Rust was written by Mozilla. I mean, it comes from Mozilla. There's a reason.

Steve: Right.

Leo: It is very much for that. And by the way, once you compile it, one of the reasons people like Rust is it's a systems-level language. It can be as fast as C and C++. So once you compile it, it's very efficient.

Steve: And think of the upside, Leo, if you get paid by the line.

Leo: I'm thrilled that Rust is now in the Linux kernel. That is a good thing for everybody who uses Linux.

Steve: Yes. I completely agree. I think that's neat.

Leo: And the issue was really libraries and support, and a lot of that's being handled now. So that's good.

Steve: So I've got one for you here from Ben Hutton. He says: "Steve, we often hear breaches could have been avoided through the implementation of a systematic software patching and update strategy. For enterprises, there are many solutions. While

performing tech support for a relative today, I found IObit Updater," he says, "IObit being a name I had previously trusted for the better part of a decade, was showing adverts for commercial products in the same" - I know - "in the same space as notifications for software updates. Finding this unacceptable, I looked for an alternative solution. I found one and expected to pay, but the consumer/home edition was free, and it seems like there are no limitations to speak of.

"Is there a solution you would suggest for Windows users for installing updates, free or otherwise? The solution I found looked suspicious, but had attained 'leader' in Gartner's magic quadrant for patch management, Summer 2022. The solution I found is 'Patch My PC.'" He says: "Only tried it today, so not an endorsement, but seems to do the job."

And so, Leo, my particular approach is just to rely on individual apps to tell me when they need to be updated, and then I update them. You know, like Notepad++ is, my god, would the guy just leave it alone, please? Because it keeps wanting to update itself. But normally apps today take care of that. And it looks like what Ben's talking about is some sort of an overwatcher who rifles through your system, looks at all the apps you've got installed, checks their versions, checks to see whether that's the latest, and then like gets involved in like telling you that you need updates. I just kind of thought maybe from your Tech Guy stuff on the weekends that there's...

Leo: Not heard of this one. The good news is Microsoft finally is acknowledging the need for this and has a package manager, believe it or not, called Winget. Which, you know how package managers on Linux will manage updates for everything on your system, including system updates. That's the idea of Winget. It's new, relatively, so I'm not sure how complete it is.

Steve: So it would be things through the Microsoft Store, probably.

Leo: Actually, that's an interesting question. The store does its own updates automatically. I think Winget goes beyond that. You do get it from the Microsoft Store. I'll have to ask Paul about that. But I think my sense is Winget is a full, or intended to be a full package manager for Windows.

Steve: So how would it know about, I mean, like in the case of our Unixes and Linuxes, we have a repository.

Leo: Because you install through the package manager; right? So the package manager, as you install stuff, makes a database of installed stuff. And then when you do an app get update or whatever, it will look at that database, see what's been installed, check for new versions. It does that in the repositories, exactly. So you would need some sort of Microsoft-maintained database of application versions. And then you could download them. So, yeah, I mean, I think that's probably why it doesn't yet do everything. But it does create these manifests. It has sources, source repositories. So I'm hopeful. But I'll have to take a look. I haven't looked at Patch My PC. I think it's an unfortunate name.

Steve: Yeah.

Leo: But, you know, doesn't mean it doesn't work well. It looks like it's for Microsoft's own endpoint manager. So it sounds like - and most of these guys, I'm looking at the engineers, are Microsoft MVPs and so forth. So it looks similar to Winget, to be honest with you, 710 supported products. I mean, when you're looking at Debian, and you're looking at apps, there's more than 10,000 packages that [crosstalk] knows about.

Steve: Yes, yes.

Leo: I mean, this is a remarkable ecosystem on the Linux side. I'd love to see Windows get to that point.

Steve: Yeah. Okay. JT Rehill. He said: "A quick question. You or Leo mentioned in a side comment a couple episodes back that uBlock Origin can block those damn GDPR cookie pop-ups."

Leo: God damn 'em. I hate 'em.

Steve: Oh, I hate them. He says: "I've tried clicking on the 'block all pop-ups' button," he says, "I use Chrome by the way, but that doesn't do it. Can you please tell me how you do this, or if there is another alternative that you know of."

Leo: I thought I showed this from the show, but maybe I showed it afterwards. You want to go to uBlock Origin's filter lists and then go down and expand Annoyances. So there's a whole - and it's hidden. There's a whole bunch of filters underneath annoyances. I checked uBlock filters annoyances and Fanboy's annoyances. And Fanboy's incorporates the EasyList cookie list. And while it's not 100%, it's 90%, at least, of all those cookie pop-ups.

Steve: Oh, nice. I didn't know that either. So I'm glad we asked.

Leo: It's a nice thing. As you know, out of the box uBlock Origin does everything you'd want it to do. But it can do a whole lot more. If you go into the filter lists, they support a massive number of filter lists. I don't think you need to add them all, but that's a couple you might want to add.

Steve: Good old Gorhill.

Leo: Amazing. You know, I can imagine - I see him with a long beard.

Steve: Uh-huh, exactly.

Leo: Living in a cabin somewhere up in the Pacific Northwest and [crosstalk].

Steve: Get out of my cave.

Leo: He's probably nothing like that. But that's what I think of, yeah.

Steve: He's like a Dvorak curmudgeon, I think, yeah. So Joel Clermont, he said: "Just listened to SN-890 about Google Analytics in the EU and thought you might be interested to learn about Fathom Analytics." It's usefathom.com. "They are designed from the ground up around privacy and designed their infrastructure to comply with GDPR, including an option to have your data never leave the EU."

Leo: I love this. I want to use this.

Steve: Yup. He says: "I have switched all my sites to it over a year ago and love it." And then I have a link to his blog, which is really good, and I recommend it to our listeners, titled "Why I Switched to Fathom Analytics." It turns out it does more, no, wait, it does a better job with less of the random cruft that Analytics - he said that Analytics had all kinds of crap that he didn't need.

Leo: Right.

Steve: But what this one does, it does better than Google Analytics was doing in his opinion. So we have something that will give our sites analytics and be privacy respecting of our users.

Leo: I will try to convince our team to use it. We still use GA, I'm sorry to say.

Steve: Yup, just have them take a look at it.

Leo: Yeah, yeah.

Steve: So Blaine Trimmell said: "You talked about the safety of public WiFi. But that article only talked about browser traffic. So if you are only using a web browser, then yes, most likely safe. But what if you're using apps that communicate unencrypted for their work? And apps on mobile devices might be making non-TLS requests. So I would say still not safe without a VPN." He says: "Have to remember someone in China could hack the WiFi router in San Francisco and capture the traffic. You do not need to travel and be local."

So anyway, I thought that was worth noting. He's absolutely right. I was thinking entirely of everything being done through the browser with the fact that the world has switched to HTTPS. But it certainly is the case that you could have an app, I mean, I hope you wouldn't, but you could have an app that just says, you know, nobody's probably looking, and does its thing, whatever it might be, in the clear. So anyway, Blaine, thank you. That is certainly a good point.

Bob Karon said: "Hi, Steve. In ref to SN-891," last week. "As an IT consultant, I never use public WiFi. Not so much from fear of hacking from someone else on the same WiFi,

but from the provider of the WiFi itself. An IT person who runs it could set up a proxy or man in the middle much easier and scrape all data through it. I always tell my clients, turn the hotspot on your phone on, and use that for your laptop if needed. I feel there's much less chance of Verizon trying to steal my traffic than some local coffee shop IT guy or even a big airport. Unlimited data is very common now on cell plans anyway. Thanks for the great show for all these years. Bob." So anyway, I just wanted to share that idea. I often use my iPhone's hotspot when I'm somewhere that I don't have WiFi, and I want to have access.

And what I promise will be the last CPE comment, but I liked it because there was a little more information, David Lemire said: "I'll trouble @SGgrc with one more CISSP CPE comment. When I was way behind on CPEs for my first year of certification, I found a blog post on the ISC website that specifically listed your podcast among a number that could count for free CPEs." He said: "Really saved my behind." So I just wanted to mention that it's not that they're, like, allowing it. They're formally endorsing Security Now! as a source of ongoing education.

Leo: Good. That's great. I didn't know that. That's great.

Steve: Yeah, really cool.

Leo: I just wanted to add that, thanks to the Discord, the Winget repository is actually a GitHub repo, and you can submit your software just as a pull request in the repository and say, hey, I'd like you to manage my updates.

Steve: Ah, nice.

Leo: So a really - I think this ultimately could be really good if the community gets behind it. It's in the Microsoft GitHub repo, Winget packages.

Steve: And at this point, that's not built into Windows?

Leo: No. I don't think so. You have to get it - you can get it from the Windows Store. I would just love to see it just become the default way to install software [crosstalk].

Steve: Yeah, well, yeah, because...

Leo: ...Windows Store; you know?

Steve: Yeah, and I was going to say that, if at some point application developers could depend upon that being present, then we could eliminate all of the individual check-for-update stuff.

Leo: Yeah. So annoying, yeah. This is so much better. Thank you, [Name], for that. I appreciate it. This is why we love our Club TWiT members. Appreciate it, [Name].

Steve: Okay. So an unintended side effect in Linux. As we know, Internet Protocol addresses endpoints by IP address. And at an IP address, we have a 16-bit port number which identifies specific services operated at that IP address. So an end-to-end connection will have an IP address and port on one end, like the source IP and source port, and an IP address and port on the other end, the destination IP and port. At the receiving end where a client is connecting to a service like web, email, or whatever, the port, as we know, is typically well known: 443, 25, 110, whatever.

And on the client's connection-initiating end, it has long been the case that when a client asks its operating system for a new outbound connection, the OS's TCP/IP network stack simply moves linearly upward, starting above the reserved service port range at port 1025, sometimes 1024, and incrementing numbers until some upper limit, perhaps all the way up to 65,535 is wrapped, before wrapping around.

So traditionally the way all TCP/IP network stacks worked, when client applications asked to initiate a new outbound connection, the stack would simply initiate the next free port in line, and you often see sequential numbered ports in blocks like if you do a netstat command on your system. They're not just scattered randomly. They're in a linear list.

Okay. But 11 years ago, back in 2011, having the OS allocating client connection ports, which is to say the source ports, linearly was seen as a potential problem since it made the next ports to be used guessable by an adversary. And that guessability might allow adversaries to hijack connections, by just like assuming what they were going to be, assuming what the client's IP and source port would be. That's the only way you designate an endpoint. So if a bad guy injected traffic, sent traffic toward the destination, there's no way to differentiate it from the traffic coming from the legitimate source. Sequence numbers comes into play there also for TCP connections, but we've already talked about all that in the past. So we know that this is all possible since it was precisely the lack of source port randomization that alarmed Dan Kaminsky about the spoofability of DNS servers Internet-wide. Attackers could blindly spoof replies by guessing the linearly allocated source ports of outstanding DNS queries.

So in response to this perceived threat, RFC 6056 was published by the IETF, titled "Recommendations for Transport-Protocol Port Randomization." And its abstract, the abstract of the RFC reads: "During the last few years, awareness has been raised about a number of 'blind' attacks that can be performed against the Transmission Control Protocol (TCP) and similar protocols. The consequences of these attacks range from throughput reduction to broken connections or data corruption. These attacks rely on the attacker's ability to guess or know the five-tuple (Protocol, Source Address, Destination Address, Source Port, Destination Port) that identifies the transport protocol instance to be attacked.

"This document describes a number of simple and efficient methods for the selection of the client port number, such that the possibility of an attacker guessing the exact value is reduced. While this is not a replacement for cryptographic methods for protecting the transport-protocol instance, the aforementioned port selection algorithms provide improved security with very little effort and without any key management overhead. The algorithms described in this document" - there are five of them - "are local policies that may be incrementally deployed and do not violate the specifications of any of the transport protocols that may benefit from them, such as TCP, UDP, UDP-lite, Stream Control Transmission Protocol, Datagram Congestion Control Protocol, and RTP," they say, "provided that the RTP application explicitly signals the RTP and RTCP port numbers." And so that's what they said. So the idea was, you know the RFCs, Leo, they're nothing if not thorough.

Leo: A great thing to read if you're getting a little sleep-deprived, yes, absolutely.

Steve: Hell, that's how I learned all this stuff in the early days was literally sat down, okay, RFC 1.

Leo: Read the RFCs. Oh, god.

Steve: So the idea was, since the source port chosen by the OS doesn't matter at all, there is no reason not to be a lot more clever when choosing the next one. RFC 6056 presents five different algorithms for doing just that, and it states that the so-called Double-Hash Port Selection algorithm offers the best trade-off. Consequently, it was recently adopted, with minor modifications, in the Linux kernel, starting with kernel version 5.12-rc1.

And this prompted a trio of industrious researchers at the Hebrew University of Jerusalem to take a look at Linux's result. What they found was not good. Their paper titled "Device Tracking via Linux's New TCP Source Port Selection Algorithm" will be presented during the 32nd USENIX Security Symposium, which is upcoming, but I have the paper now. They explain in their abstract, which is worth sharing here because we'll see what happened.

They said: "We describe a tracking technique for Linux devices, exploiting a new TCP source port generation mechanism recently introduced to the Linux kernel. This mechanism is based on an algorithm, standardized in RFC 6056, for boosting security by better randomizing port selection. Our technique detects collisions in a hash function used in the said algorithm, based on sampling TCP source ports generated in an attacker-prescribed manner. These hash collisions depend solely on a per-device key, and thus the set of collisions forms a device ID that allows tracking devices across browsers, browser privacy modes, containers, and IPv4/IPv6 networks, including some VPNs."

They said: "It can distinguish among devices with identical hardware and software, and lasts until the device restarts. We implemented this technique and then tested it using tracking servers in two different locations and with Linux devices on various networks. We also tested it on an Android device that we patched to introduce the new port selection algorithm." And by the way, Android was going to adopt it, but changed its mind when this happened. "The tracking technique works in real-life conditions, and we report detailed findings about it, including its dwell time, scalability, and success rate in different network types." And finally: "We worked with the Linux kernel team to mitigate the exploit, resulting in a security patch introduced in May of 2022 to the Linux kernel, and we provide recommendations for better securing the port selection algorithm in the paper."

So the principle that I wanted to highlight and that we keep seeing playing out over and over is that things that once seemed to be "secure enough," mostly because we weren't trying as hard as possible, are no longer considered to be so. The mess with modern processor microarchitectures, Spectre and Meltdown and the rest, is a perfect example. For quite some time we were all happily living with the way our processors worked, and with all of the performance those optimizations delivered. But that all ended overnight when some very clever academic researchers started looking much more closely.

Another example is DRAM. Same story there. Everything seemed fine until researchers began wondering whether too many bits may have been squeezed into too small a space, and whether that might create some adjacent row interference. And sure enough, we know what the consequence of that was. Similarly, the issue of IP source port

assignment was happily ignored. Then Dan Kaminsky realized that it could be a disaster for DNS. So operating systems moved to change to ephemeral key-based pseudorandom assignment. And then these clever researchers said "ah, not so fast," and discovered that there's a unique per-machine pattern that can be used for tracking.

Leo: Wow.

Steve: I wonder what will be next. Stay tuned to this podcast to find out.

Leo: Never underestimate the ingenuity and perseverance of a hacker. That's just the rule there. Amazing.

Steve: Yeah. I mean, that is the case. And so all these things, we lived with them for years, sometimes decades. And then someone said, I don't know about that.

Leo: Not so fast.

Steve: Not so fast.

Leo: I love it. I love it. Mr. Gibson, you're a gem. A jewel. And if I could say it in assembler code, I would. But I'm sending you a book.

Steve: And an old relic.

Leo: An old relic. I'm sending you two books. I told you about one. I'm sending you another I just thought of.

Steve: Ah.

Leo: That's entirely in x86 assembler. I hope you enjoy it. No prose. No prose. I don't know if you've ever seen this book. Its title is "xchg rax,rax."

Steve: No.

Leo: And I think that's all you need to know. The author is xorpd. And I just thought I'd send it to you because it's kind of silly.

Steve: Cool.

Leo: Yeah, yeah. You know, everybody should have an assembly language written book on their shelf. Just in assembly. I bet you, I'm actually really curious if you can

look at it, and you go, oh, yeah, I know what that does. Oh, yeah, yeah, yeah. Oh, that's cute. Oh, what a laugh. I bet you'll laugh reading this.

Steve: Sounds great.

Leo: Yeah, no kidding.

Steve: Exactly my kind of puzzle.

Leo: You should show this to Lorrie. Anyway, Steve is the best. We are so glad we have him every Tuesday right here, talking about security and technology in the most lucid way possible. He even makes RFCs seem entertaining. You'll find us here at 1:30 p.m. Pacific, right after MacBreak Weekly, 1:30 p.m. Pacific, 4:30 Eastern, 20:30 UTC. If you follow Steve on Twitter, @SGgrc, two reasons to do that. The show notes go there right before the show so that you can download them and read them along. He also has them on his website. But you can also message him there. His DMs are open. If you've got thoughts or comments, that's where a lot of the feedback on the show comes from.

GRC.com's the website to go to, Gibson Research Corporation. Not only for SpinRite. And as you can see, this is probably a good time to get SpinRite. If you buy 6.0 right now, you'll get 6.1, which is imminent. Perhaps even participate in the testing. You could be the one that says, "Steve, I found a flaw." You know, get a gold star on your forehead. Help him out here. GRC.com.

While you're there, of course you can get a copy of the show. Steve has two unique versions, a 16Kb audio version which sounds a little bit like a Thomas Edison cylinder, but has the one benefit, it's small, for the bandwidth-impaired. We also have transcripts written from that 16Kb Edison cylinder by Elaine Farris. She gives us beautiful transcripts that you can read along as you listen, or use them to search for parts of the show. All of that's at GRC.com, along with a 64Kb audio version, full quality audio version. GRC.com. Plus check out all the other stuff he does. If you want to try his DNS caching utility and think about all the stuff going on behind the scenes in his server - was the server written in assembler? No. That's in C.

Steve: So IIS, it's Microsoft's IIS.

Leo: Yes.

Steve: But it has a really nifty add-on facility called IISAPI, which is I-I-S-A-P-I. And so it's a huge IISAPI - I've written a huge IISAPI extension which is the ShieldsUP! and the certificate testing and all of the stuff that GRC's site does, and all of the ecommerce I wrote in assembler also.

Leo: Wow. So did you write glue code in C or C++ and then everything else can be in assembler, or do the whole thing?

Steve: No. Just assembler. Yeah, it turns out that the calling convention for the API is - all you have to do is set up the stack and just jump to a call.

Leo: Nice.

Steve: So it all works directly, yup.

Leo: It's a nice, you know, it's a nice feeling when you - it's almost like you're looking into the machine and seeing it work. And you get it. You understand it. It's pretty cool.

Steve: I just like it, yeah. I just, I think that's why coders code is that they, you know, at any level you get a sense of satisfaction.

Leo: Sure. But that's why - I think that's why assembly language coders pursue this what seems a seemingly arcane art because you are writing in the computer's native tongue.

Steve: Yeah.

Leo: You're exchanging rax with rax. You're doing it at the very base level of it, which is cool.

Steve: Although it's also why I doubt I'll ever use, I'll ever code ARM in assembler because it just doesn't seem friendly. It's not, I mean, RISC, Reduced Instruction Set Computer, as opposed to CISC, Complex Instruction Set Computer. I like CISC.

Leo: Well, you've learned all the instruction codes; right?

Steve: Yeah.

Leo: You've got them up here. So, yeah, and probably you can do in one instruction what RISC requires five for. I would guess that's what it is; right?

Steve: Yes, well, for example, I'm able to add two locations in memory with a single instruction; whereas RISC you have to load it in register, load the other one in register, add the two together, and then store the result back out.

Leo: Yeah, that's powerful, yeah, yeah. I get it, yeah.

Steve: So [crosstalk] architecture.

Leo: GRC.com. We have copies of the show at our website. In fact, if you go to TWiT.tv/sn, you'll see every show ever recorded, all 892 of them, one after the other there. You can also go to YouTube. There's a Security Now! YouTube feed that has all the shows that we've done in video, anyway, there, which is not all of them.

And probably the best way to do this, if you don't, you know, if you want to get all the old shows, you know, the feed only has the most recent 10 shows. If you want to get all the old shows, you've got to go to the website. But if you just want to get the new show when it comes out, subscribe in your favorite podcast client, set it to auto download, and you're just going to get it, and that way you can listen, you know, whenever you say, oh, I've got a minute or two, let me listen to some Security Now!. Which I think a lot of people do.

But, you know, if you want to listen on Tuesday, listen live, that's fine, too. Live.twit.tv's the live stream; irc.twit.tv to discuss it. Or, of course, in our Discord.