123456

Description: This week we look at two new just-released emergency Windows 10 updates, and the new and curious path they will need to take to get to their users. We look at a slick new privacy feature coming to iOS 14 and how it is already cleaning up prior behavior. We'll take our annual survey of the rapidly growing success of the HackerOne program, and also note the addition of a major new participant in their bug bounty management program. We briefly note the latest American city to ban the use of facial recognition for law enforcement, but we mostly examine the result of NIST's analysis of demographic bias in facial recognition outcomes. We'll also look at a high-velocity vulnerability and exploitation, and close the loop with a couple of listeners. I'll share an interesting bit of work on SpinRite's AHCI controller benchmarking. Then we'll look at this episode's mysterious title: "123456."

High quality (64 kbps) mp3 audio file URL: http://media.GRC.com/sn/SN-774.mp3
Quarter size (16 kbps) mp3 audio file URL: http://media.GRC.com/sn/sn-774-lq.mp3

SHOW TEASE: It's time for Security Now!. Steve Gibson is here. Lots to talk about. We're going to talk about that security flaw in Windows having to do with a video codec and the weird way Microsoft has decided to patch it. There's a big security issue with F5 networks. This is used by a lot of BIG-IP networking devices, so it could be a real disaster. And then we'll explain what 123456 is and what you can learn from it. It's all coming up next on Security Now!.

Leo Laporte: This is Security Now! with Steve Gibson, Episode 774, recorded Tuesday, July 7th, 2020: 123456.

It's time for Security Now!, the show where we cover the latest security, privacy, and all the news you need to know from the guy who knows the need. It's Mr. Steve Gibson of GRC, our security guru. Hi, Steve.

Steve Gibson: And Leo, fortunately my microphone is back on the proper side this week.

Leo: Oh, thank goodness.

Steve: It was a little disorienting. I think I'm not quite as young as I used to be.

Leo: Did you get any emails from people saying you moved your microphone?

Steve: No, it was actually a little unnerving for me.

Leo: It was you. It was you.

Steve: To have it on the wrong side. It was like, wait, hold on, wait a minute, what day is it?

Leo: We have an OCD audience, but we, too, are OCD. That seems to be the case, yes.

Steve: What was I - I have a different acronym for that, obsessive compulsive. I don't agree that it's a disorder.

Leo: No. It's just a way of life.

Steve: It's a feature. It's not a bug, it's a feature.

Leo: OCF.

Steve: Because it does serve me well.

Leo: That's so funny.

Steve: Okay. So we're at Episode 774.

Leo: Yes, sir.

Steve: For July 7th. And the podcast's title is, somewhat mysteriously, "123456."

Leo: I think "Sesame Street" when I hear that, but maybe it's not.

Steve: We will get to what that is about.

Leo: Okay.

Steve: But we're going to look at two new, just released, emergency Windows 10 updates and the new and circuitous path they will need to take to get to their users. It's been a source of quite some controversy. We look at a slick new privacy feature coming to iOS 14. I heard you talking about it previously, and how it is already cleaning up prior behavior. Just brilliant on Apple's part. Clean. Simple. I love it.

We're going to take our annual survey of the rapidly growing success of the HackerOne program, and also note the addition of a major new participant in their bug bounty management. We briefly note the latest American city to ban the use of facial recognition for law enforcement, but we mostly examine the result of NIST's - the National Institute of Standards and Technology - their analysis of the demographic bias present in facial recognition outcomes. Which I like because they don't have a bias.

We'll look at a new high-velocity vulnerability and exploitation, close the loop with a couple of our listeners, and I'll share an interesting bit of work from last week on SpinRite's AHCI controller benchmarking. Then we're going to conclude by discussing the mysterious meaning of this week's episode title, "123456." Oh, and Leo, we have...

Leo: Oh, a Picture of the Week, yes.

Steve: We have a picture for the ages. This is, oh, my god, the moment I saw it. You have to be a geek and know about the first, the original Star Trek series to understand the significance of what this picture is showing us. So anyway, I just love it.

Leo: Should we do the Picture of the Week, Steve?

Steve: Boy, have we got an upgrade for me and my dorm room this week.

Leo: I like that one.

Steve: Well, this is just too perfect. It's a photoshopped image from the first series, the original Star Trek series, like for the times. We've got Kirk, Spock, and McCoy wearing their COVID-19 protection masks. And conspicuously, the red security guy is not.

Leo: No mask.

Steve: No mask on that guy. And of course it's a running joke among all original series Star Trek fans that the red shirts never last very long on away missions.

Leo: They're always the first to go.

Steve: They've beamed down with four, and only three beam back up because we've lost another red shirt. And so anyway, this is just - I love the picture because it requires some understanding of the back story, and also to have appreciated watching many episodes where there's some bloodsucking or salt-sucking thing on the planet, and the red shirts are down there, and you're thinking, oh, it's not looking good for the guys in the red shirt. I would really get a different color shirt if I were...

Leo: They're security. They've got to wear the red shirts. They've got to wear the red shirts so they're always the first to go. And, you know, I always think about the actors and, "Well, I got my shot. That was it. One episode, and I'm gone."

Steve: What color's your shirt, honey? Oh, it's red. Oh.

Leo: Oh, too bad.

Steve: Try to make your death look realistic, at least.

Leo: Yeah, yeah.

Steve: Okay. So anyway, wonderful. Thank you, Twitter. I thanked the person who sent that to me, so by all means, anybody, any of our listeners who can shoot me Pictures of the Week when they're this wonderful, it will be seen by the world, or at least our little corner of it.

US-CERT posted last Tuesday on June 30th: "Microsoft has released information regarding vulnerabilities" - and they're oddly low numbered, so apparently Microsoft has known of them for a while, they're 2020-1425 and 1457, the CVE designations - "in Microsoft Windows Codecs Library." They said: "This contains updates that are rated as critical. Remote attackers leveraging these vulnerabilities may be able to execute arbitrary code. For more information on the vulnerabilities, please refer to the information provided by Microsoft."

And of course it's like, oh, what's this? Because again, this is out of cycle. This is the end of June. They didn't even feel they could wait a couple weeks until July's updates, apparently. So both of the advisories on Microsoft's site have the same title: Microsoft Windows Codecs Library Remote Code Execution Vulnerability. That's for 1425 and 1457. And the disclosures are almost identical.

But of course at this point our listeners are no longer surprised to learn of a fatal flaw in a media codec. As we know, codecs are complex interpreters of a compressing encoder's metadata. It's truly difficult to make a codec both screamingly fast as they need to be and also careful at the same time. Being super careful means checking everything, and checking everything takes precious time when a codec is by its nature often racing the clock.

So what made these stand out - aside from the fact that they were once again patches for an out-of-cycle critical remote code execution vulnerability, and the second one is an information disclosure - was the fact that Microsoft indicated that the updates would not be available through Windows Update, nor through Windows Update catalog. No, these updates would be provided through the Microsoft Store. And I was like, what? Users are instructed to click on the little white shopping bag on the Windows 10 taskbar. And I'll note that none of my Windows 10 taskbars have little white shopping bags, but that's another story. Then you select More, Downloads and Updates, and then Get Updates.

In their disclosure, Microsoft wrote: "A remote code execution vulnerability exists in the way that Microsoft Windows Codecs Library handles objects in memory." Okay, no surprise there. "An attacker who successfully exploited the vulnerability could execute arbitrary code." Right. And the other one, a slight variation, same boilerplate. An attacker who successfully exploited the second vulnerability could obtain information to further compromise the user's system.

And in either case they say: "The exploitation of the vulnerability requires that a program process a specially crafted image file" - right? So it's the evil image, which is what you would expect a codec to barf on. "The update addresses the vulnerability by correcting

how Microsoft Windows Codecs Library handles objects in memory." Then they wrote: "Affected users will be automatically updated by Microsoft Store." And according to Microsoft, users who want to receive the update immediately can check for updates with the Microsoft Store App. That's the clicking on the little white bag that I talked about before.

And as I was thinking about this, I suppose it makes sense for store apps and extensions that are sourced by the Store, even when they are provided by Microsoft, to be updated through the channel that the user used for their original delivery. And that's especially the case for third-party apps being updated. I mean, Microsoft would not want to be hosting updates of third-party apps through their own operating system and app update channels, you know, the Windows Update and the Update Catalog. So the Store it is.

Both updates were privately reported and are not known to be used in the wild. So it's not clear to me why the emergency. But the fact that it was on the 30th, which was a Tuesday - is that right? Yeah, it was a Tuesday. Maybe that was a deliberate, like, Store Patch Tuesday new thing that is going to be happening. The problems exist in the HEVC video extensions; and they're not free, surprisingly. That's 99 cents if you want that from the Microsoft Store. Maybe you'll get them as part of another package provided. There's actually two different instances of HEVC on the store. One's for 99 cents and one says it's provided by other software.

The HEVC extension, apparently not very popular, rates only 2.5 out of 5 stars. And Microsoft's description says "Play High-Efficiency Video Codec" - that's what HEVC stands for - "in any video app on your Windows 10 device. These extensions," they say, "are designed to take advantage of hardware capabilities on some newer devices, including those with these Intel 7th Generation Core processor and newer GPU to support 4K and Ultra HD content." They said: "For devices that don't have hardware support for HEVC." So a software codec to enhance what you have on your system.

And this was sort of a new designation for me, and actually we've already gone to the codec beyond this. But Wikipedia explains that HEVC, this High-Efficiency Video Coding - also known as H.265 and also MPEG-H Part 2 - is a video compression standard designed as part of the MPEG-H project as a successor to the widely used AVC, which is what everybody's now using. You know, that's H.264, which is MPEG-4 Part 10. And Wikipedia finished: "In comparison to AVC, HEVC offers from 25 to 50% better data compression at the same level of video quality, giving it substantially improved video quality at the same bit rate."

Okay. So if you're curious to know, and it turns out you may need to be curious, whether your system or any system might have the HEVC video extensions installed and, if so, which version, there is a PowerShell command which will tell you. So you'd open PowerShell, probably do it with admin because why not, and then I have the command in the show notes if you're interested. But it's Get-AppxPackage -Name Microsoft.HEVCVideoExtension. When I entered that into my Win10 machine, I got nothing. It was just blank in return. But the repaired versions of the HEVC extensions are 1.0.31822.0 or 31823.0. And so since I don't have them, my PowerShell just exited, returning nothing.

Some commentators have observed that this new Windows Store channel for releasing critical updates outside of the normal Windows security update distribution channels, even though I noted I could see why it happened, it made sense and is understandable, can cause trouble in enterprise settings, where certain Windows features and Windows Store, probably I would imagine the Store more than anything else, may have been deliberately disabled by enterprise policies. And for such companies who have purposely disabled the Microsoft Store and the Microsoft Store automatic app updates, those vulnerable computers will not receive fixes without the removal of that policy.

And in fact Computerworld's industry fixture Woody Leonhard, over in his "Ask Woody" column, was far less patient with this, and much less understanding than I was about, well, I could understand why it was the Windows Store. One of the replies to his posting noted that this optional HEVC codec exists by default in Windows clients editions since 1809, except the N and the LTSC editions. I do have the LTSC, the Long-Term Servicing Channel, so that explains why my PowerShell query came up blank.

But assuming that's the case, it would be probable then that any normal Windows 1809, 1903, 1909, and 2004, would have the vulnerable codec installed, yet presumably be unable to get it updated if the user or an enterprise had determined that they had no interest in the Windows Store and had consequently removed and/or disabled it. It's exactly the same as if we could uninstall Windows Update, which of course we can't because we need Windows Updates. So it'll be interesting to see if, like, what happens with this.

Woody wound up his post by writing: "The distribution method is riddled with all sorts of obvious holes." He said: "I mean, anybody with any sort of updating experience should've been able to compile a list of half a dozen ways that this could go wrong." And he finished: "Yet another unholy mess." And actually he also used some of the content in his Computerworld column, where he just really raked Windows or Microsoft for the debacle of the June Windows Update with all the printer issues, basically all the things we've talked about and touched on. But, ooh, being much less forgiving even than I am.

So takeaway is, if you might fall into the category of having one of the later editions of Windows 10, and having said no to the Windows Store, then you might be in a position of having a machine which is vulnerable to what will probably be exploited before long because that's the way these things go now, yet not have the means for getting that system updated if Windows Store is not going to update you. So it might be something you want to look into. Again, you can use that command that I have in the show notes to see if you have it at all. If so, what version? And then think about maybe wanting to get it updated because what it would mean is that anything in your system that would render using that codec could be subject to compromise. And if this is a big enough hole, the bad guys may try to jump through it. So we'll see what happens.

I mentioned a very slick new iOS 14 feature that is, you know, coming in the official iOS 14 release this fall, caught LinkedIn maybe red-handed. I don't quite understand their explanation for why they were doing this. Developers are beginning to play with and explore iOS 14, which is available for iOS developers. They've been discovering an unexpected and some unwanted behavior from some of their iOS apps. Apple has added a slick new privacy feature. It simply shows a pop-up notification when any app reads the content of the user's clipboard. That's all it does. Very simple, yet very powerful. It's just informing you of something going on.

Well, it turns out that a worrisome population of iOS apps have been caught essentially red-handed by this. And by that I mean, for example, ABC News, Al Jazeera English, CBC News, CBS News, CNBC, Fox News, News Break, New York Times, NPR, ntv Nachrichten, Reuters, Russia Today, Stern Nachrichten, The Economist, The Huffington Post, The Wall Street Journal, Vice News.

Over on the game side: 8 Ball Pool, AMAZE!, Bejeweled, Block Puzzle, Classic Bejeweled, Classic Bejeweled HD, Flip the Gun, Fruit Ninja, Golfmasters, Letter Soup, Love Nikki - yeah, Love Nikki - My Emma, Plants vs. Zombies: Heroes, Pooking - Billiards City, PUBG Mobile, Tomb of the Mask, Tomb of the Mask: Color, Total Party Killer, and Watermarbling.

Over on the social side, when TikTok was found to be doing that, that caused a big stir. Also ToTalk, Truecaller, Viber, Weibo, and Zoosk. And then in the miscellaneous

category, we have whatever 10% Happier: Meditation is; 5-0 Radio Police Scanner; AccuWeather; AliExpress Shopping App; Bed, Bath & Beyond; DAZN; Hotels.com; Hotel Tonight; Overstock; Pigment Adult Coloring Book to Color; Sky Ticket; and the Weather Network. All of those have for some reason been spotted as looking at your clipboard, apparently for no reason, like when they're not in the foreground. When they have no business. When you're not using them. When they, again, as I said, have no business looking at your clipboard.

Apple now, when they do that, puts in the foreground a little bubble that comes down and fades in from the background a notification. So as I said, this first, this initially came to light about two weeks ago when the Chinese app TikTok was first caught reading the content of its users' clipboards at short and regular intervals. TikTok claimed that the feature was part of a fraud detection mechanism, and that the company never stole the clipboard content. But they promised to remove the behavior, nevertheless, to put users' minds at ease. To which I'm sure everyone using TikTok probably said yes, please.

Then last week, as developers and users continued experimenting with this new pre-release iOS 14 clipboard access detection system, a developer from the portfolio building portal YourSpace.io discovered that the LinkedIn iOS app was doing this, too. In a video he shared via Twitter, the YourSpace developer showed how LinkedIn's app was reading the clipboard content after every user key press, even accessing the shared clipboard feature that allows iOS apps to read content from a user's macOS clipboard. That's one of the new features is that there's sort of a global shared clipboard among iOS apps that are within range of each other. He noted that LinkedIn was not only copying the contents of his clipboard with every keystroke, but that since iOS supports a cross-device copy-and-paste, LinkedIn was copying the clipboard content of his MacBook Pro via his iPad Pro.

When LinkedIn was asked by the tech press what the heck was going on, LinkedIn's spokesperson claimed that the behavior was a bug - uh-huh - and not intended behavior. And in a further effort to quell the growing concern, Erran Berger, LinkedIn's VP Engineering of Consumer Products, attempted to clarify, writing on Twitter: "Appreciate you raising this. We've traced this to a code path that only does an equality check between the clipboard contents and the currently typed content of a text box." Okay, that doesn't explain what this person was seeing, but okay. He says: "We don't store or transmit the clipboard contents. We will follow up once the fix is live in our app."

So what's interesting is that whatever it is that it's doing, apparently it's not necessary for it to do that. It's, oops, a bug. So they're going to turn it off. But now that users are being made aware of it, whoops, it's behavior that they've decided that they're going to get rid of. So for me, the lesson here is that simply notifying users of something that's going on behind their backs, without their knowledge, which has some privacy implications, certainly as sniffing your clipboard constantly does. I mean, many of us put sensitive data on the clipboard, like when we're cut-and-copy-pasting a password between apps. I don't want anything else snapping that. So I just love the idea that doing nothing but saying, by the way, this app just took a look at your clipboard, that goes a long way toward cleaning up that behavior and eliminating it.

So, you know, bravo to Apple for doing this. It's unfortunate that it's iOS 14 rather than iOS 1 that we're getting it in. But anyway, big props to them for that. And it really does bring up the whole issue of the safety of using the clipboard or lack thereof. If any app is able to snap it whenever they want to, I feel a lot less comfortable putting anything sensitive on there. I have noticed that LastPass will scrub the clipboard proactively for me. Sometimes it's an inconvenience because they've done that before I've had a chance to copy the content out of it into where it was going. But I do appreciate that they're not leaving it on the clipboard behind, which is a nice feature. But again, very simply and cool feature. So bravo, Apple.

Leo: I'll do the disclaimer here that both LastPass and LinkedIn are sponsors of the network, just to disclaim that.

Steve: Thank you for doing that, yes.

Leo: It's going to be my guess that there's more to this story we're going to hear about it. Just too many apps are doing it. I bet you anything it's something in the Apple UI Kit or something that's...

Steve: You mean a false positive?

Leo: No, I don't think it's a false positive. But I think it's probably a side effect. Because it doesn't make sense for companies, A, to be doing it; and, B, to be doing it maliciously. So I think it's much more likely that it's something that happens when you do something else in the UI Kit, Apple UI Kit. Apple may not be off the hook for this, in other words. It may be part of the framework. Because there's way too many people being bit by this. That's a lot of cycles.

Steve: There are a lot of people being bit by it. But compare that to the number of iOS apps.

Leo: Well, we don't know. It might be a lot more. There might be a lot more. That's just because the only time it comes up is if somebody has iOS 14 beta.

Steve: Yeah, that's true.

Leo: And then uses those apps. So I suspect...

Steve: A collision of iOS 14 and those apps; right.

Leo: I suspect it's millions. And I further suspect it's something in the framework because it doesn't - I feel like that's much more likely than the fact that all these dozens of companies would, A, use those cycles, risk detection, and be snooping on your clipboard. Like there's no reason Microsoft, who owns LinkedIn, would be snooping on your clipboard. I don't think that that's in their interest. The risk of getting caught is high. I'm sure we're going to learn more about this. It's just there's something else going on here, yeah.

Steve: I look forward to seeing what turns up, yeah.

Leo: That's my feeling, anyway.

Steve: So HackerOne has shared their top 10 public bounty programs. Last year we looked at HackerOne's top 10 bounty program to see which companies were paying the biggest and/or most frequent bounties. Now here we are today, a year later. We've got HackerOne's update for 2020. Many of the names on the top 10 list are the same, but they've moved themselves around, up and down, and a few new entrants have appeared.

Verizon Media held the first-place position last year, and they are again solidly, very solidly, in the top slot. It's like Chrome browser compared to the number two, you know, Firefox. They're way out in advance. Verizon Media runs by far the most active and successful bug bounty program. Compared to last year, Verizon increased their annual bounty payouts, increased it by 1.4 million, from the 4 million paid out last year to 5.4 paid out in the most recent year. And just one of Verizon Media's bug bounties ranks among the top five largest payouts of all time through HackerOne, $70,000 handed to a single enterprising researcher. So they are in the top slot, and solidly there.

Of course we all know PayPal. And I'm delighted to see that they are maintaining an active bug bounty program. We talked a lot in the past about how difficult it is for in-house developers to discover their own problems. Bug hunting is inherently adversarial, and PayPal is not a newcomer. Last year they were in the number three spot. But this year they have replaced Uber to take the number two position. Unlike Verizon, whose HackerOne program launched in February of 2014, that was like also one of the very earliest or in the top 10, PayPal joined the game much more recently, in August of 2018.

But nevertheless, PayPal quickly established itself as one of the most active companies on the platform. Over the past two years they've paid out a total of nearly 2.8 million, with a bit more than half of that, 1.62 million, in just the past year. So what we're going to be seeing here generally is a pattern of, like, the most recent year, pretty much being more than all of the previous history of these companies, suggesting that bug bounties are really here and happening.

Although Uber, as I mentioned, slipped from its number two spot that it held in our previous accounting, they had a significantly leaner year. They had a strong early start back in December of 2014 that's kept them near the top of the pack. But in the most recent year, Uber security team awarded 620,000 in bug bounties, bringing the company's all-time total to 2,415,000. And Uber's bug bounty program ranks in the top five among the most thanked hackers, which is another category that HackerOne tracks, and the top five most reports resolved, and the top five highest bounty paid rankings. So they also paid out a big one at some point for something that they felt was worth the money.

With all of their highly publicized recent troubles, and their very deep pockets, I suppose we should not be surprised to see Intel moving up two places from their previous year number six ranking in 2019 to the number four slot today. But then, paying out more than 1 million in bug bounties to researchers in the past 12 months will have that effect.

And although the exact amount has remained a closely held secret, it is known that Intel holds the sole distinction of having paid the highest bug bounty ever on the HackerOne platform. The single payout sum is assumed to fall somewhere between 100,000 and $200,000, but the exact amount has been kept a secret. And if anyone were to guess that it was for a side-channel vulnerability affecting Intel's CPU microarchitectures, you would be right. So again, nice that Intel is rewarding researchers for work that is certainly time consuming, and also getting their microarchitectures fixed as a consequence.

Twitter is, if nothing else, steady. They were in the number five slot last year, and they've held that spot this year. They're running one of the older programs on HackerOne, starting back in May of 2014, having paid out a total of 1.288 million in

bounties to security researchers, with 118,000 of that sum distributed in the past 12 months. So probably stronger payouts earlier, but still in the game and holding onto the fifth rank.

In the sixth position, GitLab has jumped from 10th in the previous lineup, all the way to number six spot. They're also one of the early entrants, joining in June of 2014. So they've enjoyed a rather quiet start. Across all of the six previous years they paid out a total of 570,000 in bounties. But then, in just this most recent 12 months, they paid out 641,000, so 71,000 more than all of the previous years combined, bringing their total payouts to 1.211 million. And GitLab also has one of the fastest response times on HackerOne. Amazingly enough, apparently in average, they respond to researchers within an hour to new bug reports. So that's really encouraging.

And for the first time, Mail.ru has made it to the top 10. They moved in a single year from the 14th slot into the number seven position. Much like GitLab, they've been a member of HackerOne since April of 2014, but their most recent year's bounty payouts totaling 819,000 dwarfed the 300,000 paid out through all previous years. Their bug bounty program also ranks in the top five most thanked hackers ranking, with 973 thanked hackers; and the top five most reports resolved, 3,333 resolved reports.

Everyone's favorite GitHub is also making recent upward moves. This year's bounty payouts jumped them from last year's 11th rank into the number eight slot this year. And the pattern repeats with the most recent year's payouts nearly matching the sum of all previous years. GitHub paid a total of 467,000 over the past 12 months to security researchers for their responsibly reported bugs. This brings GitHub's total since joining HackerOne in April of 2016 to 987,000. So just shy of a million dollars.

And holding steady in its ninth-place ranking we have Valve. It's been pretty steady in its payouts. In the most recent year Valve paid 381,000 in bounties to bug hunters, which brought its lifetime program total up to $971,000.

And last and possibly least, considering the effects of COVID-19, we have Airbnb. As the previous summaries have shown, the bug bounty market is rapidly growing and heating up. So despite having awarded more than 344,000 in bug bounties over the last 12 months, Airbnb's HackerOne competitive ranking dropped three rungs from its comfortable seventh spot last year. Overall, Airbnb has awarded a respectable $944,000 in bug bounties since it initiated its program in February of 2015. And as we know, their software is all the better for it.

So I think, taken overall, these numbers reveal, as I have observed, that bug bounties are finally becoming a mainstream essential component of any mature business whose software creates a privacy or a security exposure for the company, its employees, or its customers.

Leo: Okay. Who's paying the most? Who's the new one?

Steve: Sony.

Leo: Oh. What a surprise. Seems like a good idea.

Steve: Sony has launched, yup, they've launched a PlayStation bug bounty program with rewards of $50,000 and perhaps more. They'll be paying security researchers for bugs in

the PlayStation 4 gaming console, its operating system, official PS4 accessories, and also the PlayStation Network and related websites.

Leo: Good. Because as we know, Sony has been vulnerable in the past.

Steve: And they're a target. Following on the now, yeah, the now well-established responsible disclosure model, Sony will reward security researchers who discover bugs in all of their stuff.

Leo: Good.

Steve: And unlike Microsoft and Nintendo, who both top out their bounties at a measly $20,000, Sony has said that it plans to pay researchers between 100 and up to 50,000 or even higher for vulnerabilities reported in the company's products.

Leo: Nice.

Steve: The eligible targets of opportunity, as I mentioned, include the PlayStation 4 gaming console, operating system, official accessories, the network and related websites. And as I teased, Sony's new Vulnerability Rewards Program, they call it the VRP, will also be managed through HackerOne.

Leo: Yay.

Steve: Prior to taking their VRP program public, Sony had been running a private in-house invitation-only vulnerability rewards program, up until last year. And we know that the world of gaming has historically been a target-rich environment. Hackers tend to heavily target gaming accounts, which are usually abused for fraud or put up for sale online on underground hacking forums. And this past April hackers abused a vulnerability in an old Nintendo authentication mechanism to hijack more than 300,000 accounts. So, you know, Sony has the sort of deep pockets that make the creation of a bug bounty program a "just do it" no-brainer. So, you know, congrats to Sony, and cool that they've joined the ranks of HackerOne's program participants.

Leo: What does the amount of the bounty mean? What is that communicating?

Steve: Generally it is importance of what was found, and often the difficulty of doing it. So, you know, if it's a cross-site scripting vulnerability in a web page, it's, yeah, okay, thanks, here's a thousand bucks. If it's a way of bypassing, for example, the PlayStation 4 DRM, you could dangle that in front of them and probably name your price.

Leo: We want to know that one, yes, yeah. So it's the value to the company, the difficulty involved, and having a higher, you know, does it mean something to say that Sony's offering up to $50,000 compared to Microsoft's 20?

Steve: Well, yeah, because there is a limited supply of gifted hackers.

Leo: Ah. You want to attract the best.

Steve: Uh-huh. So you want motivation. I mean, if Sony really is serious about having their stuff secure, you've got to motivate the talented, responsible disclosing hackers to focus their bead on Sony rather than on Microsoft or Nintendo or someone else.

Leo: Makes perfect sense.

Steve: So, yeah, it's definitely the case that offering a greater reward is right connected to incentive. So you want to create the incentive.

The City of Boston just joined the growing number of cities that we've been talking about recently to just say no to law enforcement's use of facial recognition. And this makes Boston the second largest city to do so, behind only San Francisco. So the question is, just what exactly are such a system's limitations and liabilities? Because, you know, the decision has been made, now increasingly, that the technology's currently rather profound limitations and liabilities are judged to outweigh its benefits. So I'm annoyed by the idea that the results of tests to detect bias might themselves be biased. It's one thing to not like the idea and to find it creepy, as I know we do. But that alone doesn't mean it doesn't work.

The good news is the U.S. National Institute of Standards and Technology, our NIST, conducted their own analysis, the results from which were published late last year, and they are indeed quite eye-opening. NIST posed itself the question: "How accurately do face recognition software tools identify people of varied sex, age, and racial background?" Not surprisingly, NIST found that the answer depends upon the algorithm at the heart of the system, the application that uses it, and the data it's fed. However, they did also determine that the majority of face recognition algorithms exhibit strong demographic differentials. In this usage the term "differential" means that an algorithm's ability to match two images of the same person varies between demographic groups.

NIST's results, which were captured in their report titled "Face Recognition Vendor Test (FRVT) Part 3: Demographic Effects," were and are intended to inform policymakers and to help software developers better understand the performance of their algorithms. The report quoted its author, Patrick Grother, an NIST computer scientist and the primary author of the report. Patrick said: "While it is usually incorrect to make statements across algorithms, we found empirical evidence for the existence of demographic differentials in the majority of the face recognition algorithms we studied. While we do not explore what might cause these differentials, this data will be valuable to policymakers, developers, and end users in thinking about the limitations and appropriate use of these algorithms."

The study was conducted through NIST's Face Recognition Vendor Test (FRVT) program, which evaluates face recognition algorithms submitted by industry and academic developers on their ability to perform different tasks. I need to take a sip of milk here.

Leo: Milk?

Steve: Actually it's water. I said milk because it's white. Water. The NIST study evaluated - get this, Leo - 189 software algorithms from 99 different developers.

Leo: It's amazing that there's that many. Holy cow.

Steve: Yeah, I know, exactly. It's like holy crap, 99 different sources.

Leo: Yeah.

Steve: I mean, and that tells you that there was a clear rush, you know, to get into this technology.

Leo: Oh, yeah. Oh, yeah.

Steve: No doubt a ton of venture capital was poured into these things. So that represents a large majority of the industry. The report focuses - I don't know what's happened here. Excuse me. It focuses on how well each individual algorithm performs one of two different tasks that are among face recognition's most common applications. The first task, confirming a photo matches a different photo of the same person in a database, is known as "one-to-one matching" and is commonly used for verification work, such as unlocking a smartphone or checking a passport. The second, determining whether the person in the photo has any match in a database, is known as "one-to-many matching" and can, in theory at least, be used for identification of a person of interest.

To evaluate each algorithm's performance on its task, the team measured the two classes of error the software can make: false positives and false negatives, where a false positive means that the software wrongly considered photos of two different individuals to show the same person, and a false negative means the software failed to match two photos that are of the same person. These distinctions are obviously important because the class of error and the type of search performed can carry vastly different consequences, depending upon real-world application.

Patrick said: "In a one-to-one search, a false negative might be merely an inconvenience for example, you can't get unlock your phone - but the issue can usually be remediated by a second attempt. Whereas a false positive in a one-to-many search puts an incorrect match on a list of candidates that warrant further scrutiny."

NIST's description of this noted that the thing that sets the publication apart from most other face recognition research is its concern with each algorithm's performance when considering demographic factors. For one-to-one matching, only a few previous studies have explicitly explored demographic effects; for one-to-many matching, none have.

To evaluate the algorithms, the NIST team used four collections of photographs containing 18.27 million images of 8.49 million people. All came from operational databases provided by the State Department, the Department of Homeland Security, and the FBI. The team did not use any images scraped directly from internet sources such as social media or video surveillance. The photos in the databases included metadata indicating the subject's age, sex, and either race or country of birth. Not only did the team measure each algorithm's false positives and false negatives for both search types - that is, one-to-one and one-to-many - but it also determined how much these error rates varied among the tags, you know, the metadata. In other words, how comparatively well did the algorithm perform on images of people from different groups?

Tests showed a wide range in accuracy across developers, with the most accurate algorithms producing many fewer errors. In other words, there's highly accurate facial recognition and quite crappy facial recognition. That's my term, not NIST's. The study focused upon the performance of individual algorithms and was able to reach five broad findings. First, for one-to-one matching, the team saw higher rates of false positives for Asian and African American faces relative to images of Caucasians. The differentials often ranged from a factor of 10 to 100 times, depending upon the individual algorithm. False positives might present a security concern to the system owner, as they may allow access to imposters.

Two, among U.S.-developed algorithms, there were similar high rates of false positives in one-to-one matching for Asians, African Americans, and native groups which include Native American, American Indian, Alaskan Indian and Pacific Islanders. The American Indian demographic had the highest rates of false positives.

Three, however, a notable exception was for some algorithms developed in Asian countries. There was no such dramatic difference in false positives in one-to-one matching between Asian and Caucasian faces for algorithms developed in Asia.

Leo: Oh, isn't that interesting. There you go.

Steve: What, Leo?

Leo: That's interesting. It shows it's the training data, probably, yeah.

Steve: Exactly. Patrick Grother emphasized that the NIST study did not explore the relationship between cause and effect. One possible connection, the area for research, is the relationship between an algorithm's performance and the data used to train it, exactly as you said, Leo. He wrote: "These results are an encouraging sign that more diverse training data may produce more equitable outcomes, should it be possible for developers to use such data."

The fourth conclusion: For one-to-many matching, the team saw higher rates of false positives for African American females. Differentials in false positives in one-to-many matching were particularly important because the consequences could include false accusations. In this case, the test did not use the entire set of photos, but only one FBI database containing 1.6 million domestic mugshots.

And then their final conclusion: Not all algorithms give this high rate of false positives across demographics in one-to-many matching, and those that are the most equitable also rank almost the most accurate. This last point underscores one overall message of the report: Different algorithms perform differently. So I thought that was really interesting. You know, there are 99 wannabes, and there are a bunch of crappy facial recognition solutions. You know, maybe you get what you pay for. Or the point is, you know, just the fact that something says, "Oh, yeah, we do AI, and we have facial recognition," doesn't mean that it's good AI or good facial recognition.

That tells us that not all facial recognition algorithms are created equally, nor do they treat everyone equally. So that suggests that it's dangerous to lump all facial recognition into a single performance category or a single performance assumption. It also means that, if at some future time we decide that the technology has improved to the point where it is no longer mostly a liability, any solution that is proposed will need to be

carefully tested for bias. And I can't imagine at this point that that would not happen. So I think that's been a very important lesson that's been learned.

F5 Networks learned an important lesson, or at least hopefully their users have. Last Friday on July 3rd they released a patch for a super-critical vulnerability with a CVSS ranking of 10 out of 10. All of their so-called "BIG-IP" - that's F5's designation, BIG-IP - networking devices running application security servers are vulnerable to remote takeover. And, you know, F5 Networks is a big-iron company. They're the real deal. These BIG-IP devices are used in government networks, on public networks of Internet service providers, inside cloud computing datacenters, and they are widely deployed across enterprise networks.

The devices are so powerful and popular that on its website F5 claims that 48 of the 50 companies that are in the Fortune 50, which is to say 48 of the top 50 companies in the U.S., rely on their BIG-IP systems. So these are also unfortunately big targets. According to Mikhail - it's Russian, so it looks like Klyuchnikov, who is a security researcher at Positive Technologies who discovered the flaw and reported it responsibly to F5 Networks, the issue resides in a configuration utility called Traffic Management User Interface (TMUI) for BIG-IP Application Delivery Controller.

The Application Delivery Controller is used by large enterprises, datacenters, cloud computing environments, since it allows them to implement application acceleration, load balancing, rate shaping, SSL offloading, and a web application firewall. In other words, it's the Internet-facing big-iron hardware that all of an organization's traffic will typically pass through. And since one of the jobs is SSL offloading, that means that it's performing the TLS encryption. It's the endpoint, and connections are decrypted inside on the non-Internet edge. So that's where a bad guy would love to set up camp.

What we have learned is that an unauthenticated attacker can remotely exploit this vulnerability by sending a maliciously crafted HTTP request to the vulnerable server hosting this Traffic Management User Interface (TMUI) in this BIG-IP configuration. And I'll say once again, never, never, never expose any sort of privilege requiring management interface to the public Internet. Never, never, never. Find some way not to do it.

Unfortunately, a Shodan search in one case revealed at least 8,500 major organizations and governments had not heeded that advice and were wide open to this remote exploitation at the time of its disclosure at the end of last week, last Friday. And successful exploitation of this vulnerability could allow attackers to gain full admin control over the device, eventually allowing them to do anything they wanted to on the compromised device.

Klyuchnikov said: "The attacker can create or delete files, disable services, intercept information, run arbitrary system commands and Java code, completely compromise the system, and pursue further targets, such as the internal network. Remote code execution in this case results from security flaws in multiple components, such as one that allows directory traversal exploitation." And as for where the devices are located, 40% of those reside in the U.S., 16% in China, 3% in Taiwan, 2.5% in Canada and Indonesia, and less than 1% in Russia.

So that was last Friday. Apparently, security researchers were intrigued enough by the possibility of this exploitation to skip their Fourth of July holiday, if they were in the U.S., and instead spent the weekend developing and working out proof-of-concept exploits showing just how easily these devices could be exploited, because proof-of-concepts began appearing on the Internet by Sunday, two days later. And the next day, the attacks began. The cybersecurity community did expect that this bug would come under

active attack quickly, as soon as attackers figured out how they could exploit it, because we're talking some big pots of gold.

One such researcher tweeted: "The urgency of patching this cannot be understated." He said: "I worked for F5 for a decade. They power cell carriers, banks, Fortune 500, and many governments. If deployed correctly, the management interface should not be exposed to the Internet; but @binaryedgeio returns 14,000 hits for 'tmui' so YMMV," your mileage may vary. And the NCC Group's security researcher, Rich Warren, who's currently operating BIG-IP honeypots, said he detected malicious attacks coming from five different IP addresses.

So this is a bad one. And it's a great case in point. Never, never, never expose management interfaces to the public Internet. Make certain to have open lines of communication to the vendors of the critical devices you use so that you can and will receive notifications of critical vulnerabilities, and take action on them with all possible speed. The game has changed from the way it was a decade ago, which was are there any vulnerabilities? Today it's who is quicker, the patcher or the exploiter? And you'd rather be the former than have the latter, the exploiter, get you.

One little bit of miscellany. And I actually meant to talk to you about this offline, Leo, but I just thought I would mention that I said goodbye to YouTube TV last week.

Leo: Oh, man. This pisses me off.

Steve: Doesn't it, really? It does to me, too. It was nice, but that is just too big a jump. I don't want any of the new channels that they offer. I mean, what's annoying is nothing has ever made an a la carte environment more possible than the Internet. But that's not what's developing. Anyway, I jumped to Sling. They have the few channels that I care about, and I added the $5, 50-hour DVR. So I'm now at 35 bucks a month, and I've got as much as...

Leo: I'll do the same, yeah. It's 65 now for YouTube TV. It was up from, what, 35, then 40, then 50, and 65. And they just kept going. And I fully don't blame YouTube or Google for this because basically what they didn't think of is they invented a cable channel. And they're now prone to the same problems cable channels have, which is that the channels they want to carry are raising their costs, and so they have to pass them along. Sling has promised they won't raise prices for a year, which I think is interesting. I don't know how they're going to do that. But you get all the channels you want. I'm going to have to look at this, yeah. Yeah, okay.

Steve: It had everything that I care about, which I was glad for.

Leo: And you get DVR for a little more, 35 bucks a month; right?

Steve: Exactly. $5 on top of 30, and that gives you 50 hours, and that's way more than I need.

Leo: Oh, you get 10 anyway.

Steve: Yes, exactly. You get - is it 10? Yeah, 10 with the base package. But there were a bunch of shows that I just liked to have and then do the fast-forward routine.

Leo: Right. I'll be interesting to see what happens even to these guys.

Steve: Yeah.

Leo: It's funny because we had this dream of disaggregation, of buy just what you wanted and just either the channel or even the shows you wanted. And I think this was the over-the-top dream for these Internet providers. But basically all they've done is reinvent the cable industry.

Steve: And is it the content providers that will not allow these things to be broken apart?

Leo: No, well, oh, that's interesting. Yeah, no, no, because for instance many of these content providers are selling themselves individually - HBO, ESPN. There's lots you can get by themselves. But they're prohibitively expensive.

Steve: Right.

Leo: The bundle is the only - unless - yeah. It's funny. It's a cycle, I think. And I don't know what the answer is except just watch YouTube a lot. Not TV, but YouTube.

Steve: I imagine that Google received a message they were probably anticipating. I mean, I immediately said no.

Leo: Oh, I'm sure they didn't want to do this. That's the point. But I don't know if they had a whole lot of choice. I think they had to pass those costs along. I don't think it was their desire to do this.

Does either one of these Hulu, I mean, sorry, Sling channels have Turner? Because I really like the Turner Network, TCM, Turner Classic Movies. Good old movies.

Steve: Good question. I think, yes, I think Sling TV does happen to have TCM.

Leo: Oh, it has TCM.

Steve: Or TMC, whatever it is.

Leo: Yeah, yeah. They have the news channels, which is really the main thing I have this for.

Steve: Yes, and that's what I first looked at.

Leo: You know what they don't have, they don't have the locals. That's the big difference.

Steve: They have NBC and Fox, depending upon where you're located.

Leo: Because YouTube TV offers all the locals in every market.

Steve: And I already get CBS All Access. Yeah.

Leo: That's a big difference.

Steve: Yeah.

Leo: Sigh.

Steve: So two pieces of closing-the-loop feedback. Kevin Morris tweeted: "Hi, Steve. I've been experimenting with various flavors of Linux, and once I burned an ISO to my flash drive, I always had a devil of a time being able to use it again. InitDisk solved the problem. Thank you for that." Kevin Morris, Santa Clara, California.

Leo: Nice.

Steve: Kevin, thank you for noting it.

Leo: That's your new little freebie that you're...

Steve: Yup, my first little spinout or spinoff from the SpinRite work. And someone who's abbreviated Inspector Clouseau, he sent: "I just found out about this. Thinkst has an open source version of its Canary which runs on a Raspberry Pi. I'm trying to install it now." And it turns out they do. They're on GitHub. Thinkst is on GitHub.

Leo: That's awesome.

Steve: And Thinkst has OpenCanary. And they said: "OpenCanary is a daemon that runs several canary versions of services that alerts when a service is abused." Prerequisite is Python 2.7 or 3.6. Then they say: Optional. SNMP requires the Python library scapy. RDP requires the Python library rdpy. And the Samba module needs a working installation of Samba. So it's there, and you can definitely set up things to look like a Windows server and to look like a number of different things. There's a bunch of stuff there. So I just thought - oh, and also OpenCanary.org is sort of the main entry page. So, you know, very cool...

Leo: Awesome. I had no idea. That's great.

Steve: ...that Thinkst is also open. And we should disclaim that they are also a sponsor on the network.

Leo: Absolutely. Much beloved.

Steve: And I had an interesting experience in the last week that I thought our listeners would find interesting. In my development of an AHCI driver for SpinRite, as we know, the next release of SpinRite will be bypassing the BIOS and bringing its own maximum performance mass storage device drivers to bear. With all of our AHCI testers earlier last week reporting initial success with the driver seeing their drives, a couple of days ago I implemented its first use of bulk data transfer, which SpinRite will be using. It transfers maximum size 32MB blocks of 65,536 sectors. That's the maximum. The larger format drives support what's known as 48-bit LBA, Linear Block Addressing. And there you're able to use a word, a 16-bit word for the sector count, zero meaning, since it doesn't make any sense to transfer zero sectors, zero means 65,536, which would otherwise overflow a 16-bit register. So that's 32MB. So I'm transferring maximum size 32MB blocks.

For the initial benchmarking test I wanted to measure the time required to transfer 1GB of data. So that's 32 of these 32MB blocks. So I would initiate a transfer with the AHCI controller, wait for it to issue a hardware interrupt, signaling its completion of that block, then immediately initiate the next transfer. That code worked right off the bat. Which always makes me a little suspicious when something really complex just works the first time.

Leo: Shouldn't be that easy, yeah.

Steve: What? Really? Okay. Anyway, so I posted it for the gang in the newsgroup to test. And right away we began noticing that the numbers weren't really making sense. They looked lower than they should be, especially - this was especially obvious for the faster devices such as SSDs. Then one of our testers, whose name is MIL-Q, that's his handle, he switched his system from AHCI mode back to IDE mode, and he ran the earlier benchmark that I had completed back in 2013 before I interrupted this work to work on SQRL. That benchmark definitely maxed out his drive and his SATA link. He was getting something like 500 and, I don't remember now, 500 and some megabytes per second because the maximum theoretical SATA III speed is 600MB per second. That's the most you can get. He was, like, up near there.

So that benchmark, when his computer was switched into IDE mode, was doing exactly the same thing - initiating a transfer, waiting for a completion hardware interrupt, then starting the next transfer. But it was not using the super-fancy AHCI controller. In IDE, which is also sometimes referred to as "compatibility mode" in the BIOS, it was bypassing the AHCI controller's features and performing what's known as "bus mastering" DMA. And that's what was blazing.

So I turned my attention back to the AHCI solution. The first thing I did was to try timing the transfer of just one single 32MB block. Now, that's problematic for a benchmark due to device caching. A 1GB transfer is guaranteed to bust out of any cache and not be cached. But 32MB might be. But I was doing all this work with an SSD anyway, so that

didn't matter. What I found doing just one transfer was I was seeing much higher calculated byte transfer rates. That suggested that the trouble was inter-transfer overhead, the time being taken from the end of one block to initiating the next one. And this fit the symptoms since it was the faster devices, the SSDs, where we seemed to be returning values that were the most off. That would make sense because the inter-block interval would be consuming a larger percentage of the whole when the transfers themselves were super short.

So the next thing I did was to transfer only one block, but to also eliminate the interrupt service overhead by doing what's known as "spinning" on the AHCI controller's completion status. Rather than halting the thread and waiting for a hardware interrupt to reawaken me, I "spun," that is, I just ran a tight loop doing nothing but polling the completion status bit of the AHCI controller. That way, the instant it flipped to "done," I would stop the timer and calculate the time that had been spent and thus the effective byte transfer rate. And sure enough, once again I got a better result.

So then I switched back to a multi-block transfer, still using the interrupt-free spin loop, and got the best results I'd seen so far. But I was still not getting the performance that I was getting if I switched back to much simpler IDE Bus Mastering. And of course now I understand why. When using IDE and Bus Mastering, you are directly talking to the drive, and you have the minimum overhead possible. The AHCI controller is a little bit of a misnomer for people thinking that it is like some dramatic advance on technology. It itself, it is incredibly complex. That tells us that it is not implemented in hardware. It is a microcontroller. So it is executing microcode, which means it's going to be a little slow. It's going to have some overhead of its own that talking directly to the drive hardware doesn't.

What that means is that where the AHCI controller comes into its own is when you are doing much more at once, when you have a really busy system with lots of hard drives and lots of work being done. However, the AHCI controller allows you to queue up work. And that's the key to getting the most performance. For example, even when talking to one drive, I'm able to queue up 32 pieces of work. That is, I could queue up the entire 32-block transfer at once, and it would run the entire block without generating a single interrupt. And the moment the drive signified that it was done, it would start on the next block from the controller.

I can go one step better, though, and use something known as NCQ, Native Command Queuing. With Native Command Queuing, you actually put the work in the drive. While the drive is transferring, you're able to give it the next pieces of work that you wanted to do, allowing it to have the work and just basically stream these things into the system. So anyway, I reworked the code. I've done that first part where I am putting multiple pieces of work in the AHCI controller. My brief look at it, I had to pause the work in order to do this podcast, but it looked like I was achieving what I had. So I'll just touch in on this next week and let everyone know how we're doing. But there's a little snapshot into the fun we're having over in the SpinRite dev newsgroup.

Leo: It's nice to have that group, to have some people to bounce the stuff off of.

Steve: It's invaluable to me. I just - I can't imagine doing this without having a bunch of interested people who are pounding on this with all their hardware.

Leo: Have you had that before?

Steve: Yeah. SQRL was done that way.

Leo: Well, I know SQRL, yeah.

Steve: SpinRite was done, the original work on SpinRite was done that way.

Leo: Was it.

Steve: I've been doing this from the beginning. It's just - it's too useful. And in fact the reason I'm releasing a benchmark is to hope that our listeners, the listeners of this podcast, because I'm not going to drag everybody over into the NNTP textual newsgroups, I've got GRC Forums ready to go. I want to release this as a benchmark so that I can get all of our listeners to give this a test. I mean, people are really interested to see what performance they're getting from this or that drive. It's just, you know, benchmarks are fun. And so we will have public web forums. Mostly it will allow me to learn of additional problems that I haven't found so that I can fix those, and then all of this technology moves from being the platform for the benchmark into the new platform for SpinRite.

Leo: Very cool. How exciting. So people just go to GRC.com/forums, and they can join up there and all that?

Steve: Forums.grc.com. Not yet open, but it will be.

Leo: It will be. Okay, cool.

Steve: Yup. So Leo?

Leo: Yes?

Steve: What is 123456?

Leo: It is the most...

Steve: It's not a Fibonacci sequence.

Leo: No, we know that much.

Steve: That would be 112358.

Leo: Yes.

Steve: It's not a linear regression.

Leo: No.

Steve: But it is a linear transgression.

Leo: Yes, it is.

Steve: Because believe it or not, still, in this day and age, 123456 turns out to comprise one out of every 142 passwords.

Leo: That's crazy.

Steve: Found on the Internet.

Leo: Oh, my god, that's crazy.

Steve: In one of the largest password reuse studies of its kind, the password 123456 was found to occur seven million times across a massive data trove containing one billion leaked credentials.

Leo: Wow.

Steve: As we know, the number of leaked credential database collections continues to grow as new companies continue to get hacked, and their databases get exfiltrated. And they are eventually made available online at GitHub or GitLab or distributed on hacking forums and file-sharing portals. And some good use is being made of them. Responsible tech companies like Google, Microsoft, and Apple have collected leaked credentials to create in-house alert systems that warn users when they are utilizing a weak or a common password. And of course we know Troy Hunt, his famous HaveIBeenPwned online service relies upon submissions of these leaked credential databases.

So last month a Turkish student studying at a university in Cyprus decided to download and analyze more than one billion of these leaked credentials. And what's very cool is his work is up on GitHub for anybody who wants real detail. His primary discovery was that the one billion-plus credential database contained a startlingly high count of duplicates. Or stated another way, among the more than one billion passwords, only 168.9 million were unique. And of those nearly 169 million unique passwords, more than seven million of them were 123456.

Leo: Wow.

Steve: So that means that one out of every 142 passwords included in the analyzed sample was the number one weakest password known today. Additionally, his research also revealed that the average password length is 9.48 characters, so just shy under 9.5

characters. And that's not great, but at least it's not six, as is 123456. Since I invariably use LastPass to synthesize my own passwords, I typically have mine set to 32. And then I'll reduce it as required when some brain-dead website sets a lower upper limit and complains, then I have to crank it down to 20 or something. But still.

So this Turkish researcher also observed that password complexity was another problem. Only, get this, 12%, one in eight, of the passwords contained any special character. Now, while that's not good for them, that's great news for us, since the bad guys who are attempting to brute force our credentials also know this. They will certainly expend all of their time not bothering to brute force special characters because they know the chances, what is it, 92%, no, 88% chance that the password does not have any special characters. That's what they're going to do. They're not going to bother with the 12%. But that's all of us; right? So we use special characters, and so we're not brute forced because the bad guys are not going to bother with that because the chance of finding somebody with a special character is vanishingly small. So special characters for us, yay.

Okay, what else? Most of the passwords, 29% of them, used only letters; or 13% only numbers. The full research I've got up on GitHub, or he has up on GitHub, a link in the show notes for anyone who's interested. But we have some nice bullet point takeaways. From one billion-plus lines of dumps, he filtered out 257.669 million as the data was corrupt, improper format, or test accounts. So that one billion credentials boiled down to, as I said, just shy of 169 million unique passwords, and 393 million unique usernames. And of course, as we titled this podcast, the most common password was 123456, covering roughly 0.722% of all passwords. That's seven million times in that one billion set.

The top thousand recurring passwords, the top thousand, okay, so the thousand most recurring passwords represented 6.6% of all passwords. Which is interesting because that means that checking, for a brute forcer, checking those top thousand, you would have a 6% chance of it being one of those top thousand. With the most common one million passwords, the hit rate against the whole collection is 36%. So that was the first - so the top thousand was 6.6%. If you expand it to the most common one million, the hit rate against the whole collection goes up to 36%, so better than a third chance, if your guess is the top one million. And with the top most common 10 million passwords, the success rate is 54%, meaning that if you were a brute forcer, and you made a dictionary of the top 10 million passwords, which is not that big a deal, you stand a better than 50-50 chance of cracking someone's password.

So anyway, surprising numbers. The average password length, we said, just shy of 9.5, 9.42 characters long. Only 12% contain any special characters, as I mentioned, so 88% of those don't. So we're happy to be using special characters. 28.79 are letters only. 26% of the passwords are lowercase only. Nobody bothered to hit the shift key. 13.37, numbers only. And this is interesting. 34.41% of all passwords end with digits, but only 4.5% of all passwords start with digits. So start your password with a digit, folks. The bad guys know these numbers. They have the stats. They won't be trying that.

And, finally, in an observation that constitutes a real contribution, this researcher noticed and then researched an unsuspected pattern in the data. He wrote, and I've actually fixed the grammar and made it a little more clear, but essentially he wrote: "During my research, I've noticed a handful of apparently high-entropy passwords" - meaning all 10 characters, upper and lowercase, and digits - "that were being reused. These passwords had a very low occurrence rate, but far more than one would expect. Specifically, they all start and end with uppercase characters. None of them seem to have a keyboard pattern or meaningful word in them. They are all 10 characters long. They don't contain special characters. Some of them occurred up to one per 100 million credentials," he said, "meaning I have around 10 reuses of them currently."

And he said: "The most recent occurrence for these, 86 of these were found in a set of 55,623 credentials from a leak in June of 2020." So a very recent leak, last month, 86 of those. He found 763,000 passwords matching this pattern. And he concluded by observing: "I have no idea what this uncovers and what it implies. But I'm suspecting a password manager out there is creating passwords with low entropy, causing repetitions over a lot of users. All the ideas about this are welcome and appreciated."

And I would draw the same conclusion. I think that's really interesting. We know that there's widespread reuse of password managers. There may be one which does not have a very good entropy source. And as a consequence, without them knowing it, users are generating duplicate passwords, either with their own use and reuse of this, or among all the users of this. So I thought that was a really interesting finding and some interesting stats on passwords. Our takeaway is, begin your password with a number, use a special character, and you will just be off the brute forcers' radar.

Leo: Very good. It's nice to know. Although I wouldn't tell everybody how long your normal password is. Now we know. We start at 32 characters.

Steve: Well.

Leo: And they're more than welcome to try to brute force 32 characters.

Steve: Absolutely.

Leo: I never use, if a site says it has to be 12 or less, 12 or fewer, I always do 11 or nine or something because then that's a different situation because that's a big pool of customers who probably are using 12. So, yup. You've got to wonder where you could use 123456 these days.

Steve: Boy. I mean, the browser itself will break out into laughter.

Leo: Yeah, I know. Probably influenced, I don't know what the dataset was for this Turkish researcher, but probably influenced quite a bit by the stuff that he had in there. You know, it's not massive. Or maybe it is. I don't know.

Steve: Well, it's more than a billion.

Leo: Yeah.

Steve: That's a lot of credentials.

Leo: Yeah, yeah, yeah. It's probably all Yahoo users, see. That's the problem. Right? You know. That's Steve Gibson. He is not a Yahoo user. I can guarantee you that.

Steve: No, sir.

Leo: No, sir. Not AOL. Don't email gibson@aol.com. Nothing's going to come back. In fact, don't email him at all. Best thing to do is follow him on Twitter, @SGgrc. He puts his show notes up there early for me, but you can also partake.

Steve: Back then it was for Elaine.

Leo: More for Elaine than anybody else. Okay. That's what Elaine uses. Elaine does these great transcriptions, by the way. This is the only show that is regularly transcribed by a human being, which is awesome. And you can get those transcriptions along with the 16Kb versions of the audio and 64Kb versions at Steve's site, GRC.com.

While you're there, you've got to check out SpinRite, the world's best hard drive recovery and maintenance utility. And now would be a good time to pick up SpinRite 6 because you'll get a copy free of 6.1 when the upgrade comes out. Plus you'll be part of all the beta testing, and you can see it starting to get really, really interesting. GRC.com. Lots of other great stuff there, freebies and so forth.

We have audio and video at our website, TWiT.tv/sn. We also have on-demand versions. All you have to do is go to your favorite podcast application and subscribe. That way you'll get it the minute it's available of a Tuesday evening, maybe Wednesday morning, maybe, depending on your time zone. We do the show in California time, so roughly 1:30 p.m. Pacific. This time it was a little later, so it varies. But around, starts no earlier than 1:30 p.m. Pacific. That's 4:30 Eastern time. That's 20:30 UTC. Adjust accordingly for your time zone. And then fire up the browser and head to TWiT.tv/live. That's where you'll find live audio and video streams you can listen to. And they're going all day, all night. Sometimes it's new content, like it is on Tuesday afternoons; but sometimes it's rerun content. But there's always something great to listen to there.

If you're doing that, join the chatroom. They're listening to that live stream and talking back to it. It's funny because I'll go sometimes to irc.twit.tv, actually I have it running all the time at home, and every once in a while I'll be home, you know, eating breakfast, and see somebody talking back to me because the show that's running was a show I did the day before. And sometimes I'll answer, which will really confuse people. But I thought this was a rerun? Anyway, there's lots of great people. If you're getting a little lonely during quarantine, the chatroom is a great place to hang out: irc.twit.tv.

For people who listen on demand, we also have a forum, just like Steve's forums. Ours are at twit.community. That's the website, twit.community. It's a great place to hang out, talk about shows, feed back to the shows. But as I said, the best way to feed back to Steve is either GRC.com/feedback or on his Twitter account because he takes DMs from anybody. He's crazy that way. So @SGgrc is his Twitter handle.

Steve, we will reconvene next week for another fabulous edition of Security Now!. I'll see you then.

Steve: Thanks, buddy.

Copyright (c) 2014 by Steve Gibson and Leo Laporte. SOME RIGHTS RESERVED

This work is licensed for the good of the Internet Community under the Creative Commons License v2.5. See the following Web page for details: http://creativecommons.org/licenses/by-nc-sa/2.5/