SECURITY NOW!

Transcript of Episode #813

A Spy in Our Pocket

Description: This week, by popular demand, we examine the big cover-up at Ubiquiti. We look at the consequences of the personal data of 533-plus million Facebook users appearing on the 'Net and how to tell if you're represented there. We look at another water treatment plant break-in with a very different outcome. We look at a new move by Google to further lock down Android against abuses of its permissive-by-design API services. We look at the new threat to Call Of Duty cheaters, and yet another set of serious vulnerabilities in QNAP NAS devices. Then, after sharing a catchy tweet, we look into some new research from researchers in Ireland into the unwarranted chattiness of iOS and Android mobile phones.

High quality (64 kbps) mp3 audio file URL: http://media.GRC.com/sn/SN-813.mp3
Quarter size (16 kbps) mp3 audio file URL: http://media.GRC.com/sn/sn-813-lq.mp3

SHOW TEASE: Coming up next on Security Now!, it's me, Jason Howell, filling in for Leo Laporte this week. But you've come for Steve Gibson, and he is here to deliver. He talks all about Facebook's major data leak and what's going on there. It happened a couple of years ago, but now all that data is out. You were warned. Also a threat to Call of Duty cheaters, the cover-up at Ubiquiti, and just how much do these devices that we have in our pockets, how much are they sharing our data? Well, Steve has all the numbers to share with you next on Security Now!.

JASON HOWELL: This is Security Now! with Steve Gibson, Episode 813, recorded Tuesday, April 6th, 2021: A Spy in Our Pocket.

It's time for Security Now!, the show where we talk about the week's biggest security news. And we have just the guy to do it. I'm Jason Howell. I'm not the guy. I'm filling in for Leo Laporte, who is out enjoying time away from TWiT for a change because he's been here like the last year solid because of the pandemic and everything. So I'm super happy to sit in for Leo. But the guy I'm talking about is Steve Gibson. He's the guy you really tune in for. How are you doing, Steve?

Steve Gibson: Hey, Jason. Great to be with you again for your fill-in week. And yes, we know from the things that Leo has shared publicly that he lives to travel. He's found a travel mate that he really enjoys, and he's been stuck in COVID land, like, going stir-crazy.

JASON: As we all have.

Steve: And he was saying, I heard him say the other day that he's off to Mexico or something in October. I don't remember where it was he was going.

JASON: Yeah, he's got a bigger plan, a bigger trip planned later.

Steve: A little brief one, and we'll have him back. But anyway, this is Episode 813 for the 6th of April. I titled it "A Spy in Our Pocket." There's some new research out of some security guys in Dublin, Ireland, who went to the trouble of - they've done some research that was sort of like this previously, but they expanded their look. They've got the technology to watch the cellular communications of both - in their case they looked at the top two mobile phone families, iOS and Android - and looked at just how chatty they were. And it's a good thing that hard drives are as cheap as they are because, given the install base of these drives, I've got the numbers in the notes, but they're spooling terabytes of data an hour across their entire install base of devices. But we'll end up talking about that.

We've got a bunch of interesting stuff. By popular demand, I can't tell you how many of our listeners wanted to make sure I had heard about this. We've got the big cover-up at Ubiquiti. I'll be talking about that a little bit later and why all of our listeners were saying, hey, Steve, did you know about this? We're also going to look at the consequences of the personal data of 533-plus million Facebook users just a few days ago appearing on the 'Net for free. And also, happily, how to tell if you might be represented there.

We're going to look at another water treatment plant break-in that had a very different outcome than the one in early February in Florida. Also we look at new moves by Google to further lock down Android against abuses of its original permissive-by-design API services. You know, they wanted to be the alternative. It's going to be open. Everybody's on the same side. Like, well, not so much. So we've seen a succession of tightenings that they've had to do, and we're going to talk about another one today. We also are going to look at, because all of the tech press was just all over this story, the new threat to Call of Duty cheaters; and also another set of serious vulnerabilities in QNAP NAS devices, which I try not to talk about because there are so many of them, sort of like ransomware. But anyway, we have to because QNAP is just bad.

Then, after sharing one catchy tweet that as a coder I really appreciated, we'll wrap up by talking about the research from Ireland into the fact that we really do have spies that we carry around in our pockets, and just what's being spied on. Oh, and we have a terrific Picture of the Week. So I think another great podcast for our listeners.

JASON: Absolutely. And let's just get this out on the record right now. You didn't fill this rundown with Android stories because I'm on today.

Steve: That is true. It's the powers, they're inscrutable. Somehow Mr. Android is here on an Android-heavy episode of Security Now!.

JASON: I lucked out. Right on. Well, we've got a whole lot of great stuff to talk about. Thank you, Steve, for that. All right, Steve. Let's get to this pic right here. I know I've seen this at the playground, but it takes on new meaning here.

Steve: Well, I just got a kick out of it. Somewhere on Twitter, I think someone sent it to me. For those who are not looking at video, it shows a park with some kid-friendly, jungle gym-y sort of things in the background, and one of those tube slides, like made of real industrial tubing, that a kid would get in, like climb up the ladder at the high end, get in, and then slide down the inside and come shooting out of the end, down like to hit nicely sawdusted covered ground. Anyway, this shows that the end, the output end is covered over. There's a bolted piece of plywood completely closing the exit. And someone has written "Slide Closed" on there.

And this was titled "A Backend Fix for Frontend Issues," which I got a kick out of, of course, from a computer science standpoint. We're hoping that the top of the slide, which is out of the picture, we can't see it, similarly has a big piece of closed plywood bolted across it because otherwise you'd be in trouble; you know? Kid would get in the front end, slide down, hit the back and be like - and then, you know, Mommy would be hearing some screaming about, you know, "Agh," because you have to try to climb back up the tube, I guess. And worse if you have a pileup down there at the bottom.

JASON: Yes.

Steve: So anyway, we're assuming that the slide people were conscientious and have blocked both openings, the front end and the back end.

JASON: I certainly hope so. Although looking closely at this side...

Steve: It does beg the question.

JASON: Yeah, it really does. We don't know for sure based on the picture. But looking closely at this slide, I see down at the bottom there's a little notch. So at least if you get locked in there, you're not going to run out of air.

Steve: And really, if you wanted to close the slide, it occurs to me that just closing the top would do the job; right? Because if you can't get in at the top, there's no slide to be had. So kids are mischievous.

JASON: Although I will say I've seen my kids many times climb up through the bottom. So you've probably got to close it on both if you really want to keep them out.

Steve: I had a childhood we will not go into in any detail. Okay. The big Ubiquiti cover-up, the most tweeted-to-me issue in quite a while. Our long-time listeners know that I've been a fan of the Ubiquiti EdgeRouter X, which is an amazingly affordable five-port Ethernet router. I think it was like $49, max $49 at the time on Amazon. And the thing that drew me to it was that its five interface ports were actually individual NICs, actually Network Interface Controllers. It wasn't a router hooked like a one-port WAN and then four ports of LAN, where they were actually just a digital switch so that, for example, they were all related to each other. These were individual NICs, which meant they could each have their own subnet to create a truly segmented network on the LAN side. And as I've often preached, this is the only way, that is, network segmentation. And yeah, you can potentially use a VLAN, but it's, I mean, it's a little more tricky to do that. So this is the way to create a secure environment on today's LAN.

With many of our IoT devices, immediately upon hooking them up, phoning home to Chinese cloud services, there can be no true security if those devices are allowed to sit on the same network as the family's or small business's LAN. You're just, sooner or later, we're going to be talking about this on the podcast. So IoT devices must be given their own isolated network. And at the time the Ubiquiti was the only cost-effective solution for doing that.

Today, a feature of more modern consumer WiFi routers is to have one or sometimes two guest channels which support explicit inter-network isolation. Sometimes when I'm unable to reach my Sonos speakers from my iPad, it's because they're on the IoT WiFi, and my iPad is on the real WiFi, the inter-secure device WiFi. And they can't see each other. Which is what you want. So since we so often talked about Ubiquiti, it should be no surprise that many of our listeners have made sure that I knew of the mess that has recently erupted over there. It was fresh last week. I got some early indications on Tuesday morning. But there was still some, I mean, I quickly took a look at it. Some things were still unknown. Brian Krebs, who was much involved with this, was still sort of working out the details. So I decided just to hold off.

Now we know much more. It appears clear that Ubiquiti has been proactively covering up the extreme customer-affecting severity of a data breach that put probably all, many of their customers' networks at significant risk of unauthorized access. Brian Krebs has cited an internal and unnamed whistleblower from within Ubiquiti. And Brian is very careful. If Brian says this, he has vetted the crap out of it. I mean, I would absolutely believe what Brian is posting on his site, KrebsOnSecurity.com. So in January Ubiquiti seriously downplayed what it said was "unauthorized access to certain of our information technology systems hosted by a third-party cloud provider."

Okay. So while that was literally correct, it was also quite vague. The notice said that, while there was no evidence the intruders accessed user data, the company couldn't rule out the possibility that they obtained users' names, email addresses, cryptographically hashed passwords, addresses, and phone numbers. Ubiquiti recommended at the time that users should change their passwords and enable two-factor authentication. So at this point that's standard generic post-breach boilerplate. But last Tuesday's report from Brian was able to flesh this out by citing that security professional within Ubiquiti who helped the company respond to the two-month-long breach, which it turns out began in December of 2020. Brian's inside source reported that the breach was much worse than Ubiquiti had disclosed publicly, thus my term "cover-up," and that executives were minimizing the severity to protect the company's stock price. So, you know, no surprise there.

And to make matters worse, the breach follows Ubiquiti's push, if not, well, it is now, it's become an outright requirement for cloud-based accounts for users to set up and administer the devices running their newer firmware versions. An article on Ubiquiti's site says that during the initial setup of, for example, a UniFi Dream Machine, which is a popular router and home gateway appliance, users will be prompted to log into their cloud-based account; or, if they don't already have one, to create an account. And I'll just note this is similar to my annoyance with some versions, apparently not all, though, because I've managed to bypass it, but Windows 10 is now like there had been a way to just say no, I do not want to log into Microsoft in order to create an account on my Windows 10 machine. I want to just do it offline. And you have to now jump through hoops, last time I did that anyway.

So Ubiquiti's page states: "You'll use this username and password to log in" - get this - "log in locally to the UniFi Network Controller hosted on the UDM, the UDM's Management Settings UI, or via the UniFi Network Portal for Remote Access." So, I mean, somewhere, misguided though this was, they thought, oh, everything is cloud. We want cloud. Our users want cloud. They were wrong there. But so even to talk to the thing sitting next to you, the blinking lights that you can see, you log into the cloud, and then they reach back to it. It's like, oh, this is a recipe for disaster.

So these changes had not gone unremarked upon. Ubiquiti's customers are decidedly unhappy, complaining about the cloud login requirement and the risk it inherently poses to the security of their devices. Any of our listeners would understand that this is not a good idea. I mean, okay, in some use cases yes, but not as a blanket requirement where you can't configure the device whose cord you keep tripping over without logging into some cloud in China. Thank you.

So as I noted previously, according to Brian's inside contact - and he calls this guy "Adam" just for convenience. So we don't know his name, so we'll call him Adam. After all, Adam was the start of all this. The data that was accessed, Adam alleges, was much more extensive and sensitive than Ubiquiti portrayed. So I'll paraphrase a bit for brevity, from what Brian wrote.

He said: "In reality, Adam said, the attackers had gained admin access to Ubiquiti's servers within Amazon's cloud service." Okay, so that's that third-party cloud provider that they alluded to back in January. Turns out it's Amazon AWS "which secures the underlying server hardware and software, but still requires the cloud tenant" - meaning Ubiquiti, the user of AWS services, the cloud tenant - "to secure access to any data stored there." Which apparently they were a little lacking in. "They were able" - "they" the attackers - "were able to get cryptographic secrets for single sign-on cookies and remote access, full source code control content, and signing keys exfiltration." In other words, yeah. Full admin access rights.

Adam says the attackers had access to privileged credentials that were previously stored in the LastPass account of a Ubiquiti IT employee. So, okay. It was secure there. Oh, let's put it in the cloud. Anyway, and gained as a consequence root admin access to all Ubiquiti AWS accounts, including all S3 data buckets, all application logs, all databases, all user database credentials, and secrets required to forge single sign-on cookies. In other words, this entire Ubiquiti cloud infrastructure was compromised, and Ubiquiti had previously forced all of at least their newer users who were upgrading firmware even of older devices to switch to the new and better cloud solution.

So this would allow, Adam was explaining, the intruders to remotely authenticate to countless Ubiquiti cloud-based devices around the world, all those owned by Ubiquiti's customers. According to its website, Ubiquiti has shipped more than 85 million devices to play a key role in networking infrastructure in over 200 countries and territories worldwide, all of which, or many of which at least, are now vulnerable. Any that have been updated to use Ubiquiti's now mandatory cloud facility were vulnerable to remote takeover.

And even after all this, Ubiquiti continues spewing the corporate line. They said: "As we informed on January 11, we were the victim of a cybersecurity incident that involved unauthorized access to our IT systems. Given the reporting by Brian Krebs, there is newfound interest and attention in this matter, and we would like to provide our community with more information." You know, by not doing so.

They said: "At the outset, please note that nothing has changed with respect to our analysis of customer data and the security of our products since our notification on January 11th. In response to this incident, we leveraged external incident response experts to conduct a thorough investigation to ensure the attacker was locked out of our systems." Yes, the barn door has been securely closed and locked. There's nothing in the barn; but, boy, is our barn secure. After they were rummaging around and took everything for two months.

"These experts," they continue, "identified no evidence that customer information was accessed, or even targeted." Okay. "The attacker, who unsuccessfully attempted to extort the company by threatening to release" - right, that information that apparently wasn't stolen in the first place. Okay, I'm sorry - "by threatening to release stolen source code and specific IT credentials, never claimed to have accessed any customer information." Though having it all.

"This, along with other evidence, is why we believe that customer data was not the target of, or otherwise accessed in connection with, the incident. At this point, we have well-developed evidence that the perpetrator is an individual with intricate knowledge of our cloud infrastructure. As we are cooperating with law enforcement in an ongoing investigation, we cannot comment further. All this said, as a precaution, we still encourage you to change your password if you have not already done so, including on any website where you use the same userID or password. We also encourage you to enable two-factor authentication" - maybe you should use three - "on your Ubiquiti accounts if you have not already done so."

So okay. Our takeaway for the podcast is that Ubiquiti's skeptical customers were 100% correct to object to upgrades of their systems' firmware which offered them no alternative other than to manage their own local devices through an online cloud service. That's insane. And this meant that all of these devices were now vulnerable to remote access compromise, though they had never been before this cloud centralization. It was clearly wrongheaded from the start. Anyone using these cloud-attached Ubiquiti devices should obviously have changed their passwords at the first inkling of this.

And certainly I agree with the advice. Use two-factor authentication if you can't get three, if you haven't already done so. And given that intruders into Ubiquiti's network had access to single sign-on cookies and secrets enabling remote access, including signing keys, it would also be a good idea, annoying though it is, to delete any access profiles associated with any of those devices. Make sure the device is using the latest firmware, and then recreate profiles under brand new credentials. And of course, as always, remote access should always be disabled unless it's truly needed. We're going to be running across many things during this podcast where stuff is being done just because it can be. And that's not security. That's not secure.

So anyway, that's the story on Ubiquiti. And yes, I'm up to speed, and I'm certainly disappointed in them. The good news is there are now better, more modern, much more consumer-friendly alternatives for doing things like setting up sequestered and segmented IoT networks. Most IoT things are WiFi anyway. So what we talked about back then was getting an old pokey 10BASE-T or 100 router set up as an access point and plugging it into one of the Ubiquiti router's ports so you could create a sequestered and segmented WiFi that you could, you know, all your plugs and your light bulbs and your thermostats and things could be connected to just because you really just don't want those things on the same LAN as stuff you really care about. Which is everything not IoT.

So we have Facebook's, and we have a count, 533,313,128 user whoopsie. We talked a lot about this Facebook breach back when it happened. It was two years ago, in 2019. And now the data that was taken during that intrusion has all just been released onto the dark web for free. You don't have to pay anything. This includes the full names, real-world names, Facebook IDs, mobile phone numbers, physical locations, email addresses, genders, occupations, city, country, marital status, account creation date, and other profile details which in this massive database are broken down by country, with over at least 32 million records belonging to users in the U.S. alone, 11 million in the U.K., 6 million users in India, and so forth.

So when I saw this I thought, you know, you've got to love it that Facebook has a position there titled "Director of Strategic Response Communications." You can imagine the conversation. "Hi. What do you do?" "Well, my name is Liz Bourgeois." It actually is. "And I'm the Director of Strategic Response Communications for Facebook." And the person says, "Wow. So when the you-know-what hits the fan, you're the person who gets quoted in the media while managing to never say anything meaningful." And Liz responds, "Yes, that's a perfect job description."

In this case Liz said: "This is old data that was previously reported on in 2019. We found and fixed this issue in August of 2019." So, okay. Well, so the data is old. Then what? That creaky old data must no longer apply; right? People have all changed their names and phone numbers and Facebook IDs. They all moved and changed their genders and so forth. So we really don't need to worry that the data of 533,313,128 Facebook users from only two years ago, not so old really, is now being offered for free in bulk on the dark web. Thanks, Liz. Whew. For a while there, this seemed like it might be a big deal. And that was, by the way, a beautifully executed strategic response. Very strategic.

JASON: That's why she is the Director of Strategic Response Communications.

Steve: Yes.

JASON: Put that at the top of your rsum for your next position of that same title.

Steve: Someone named Liz Bourgeois to be in charge of your strategic response. Okay. So just to remind our listeners, the data is known to have been obtained, we talked about this two years ago, apparently in August, by exploiting a vulnerability in Facebook's Add Friend feature, which enabled automated scripts to scrape with abandon Facebook users' public profiles and associated private phone numbers in bulk. It's true that the leak was fixed. And at the time, I remember being somewhat sympathetic to the complexity of doing what Facebook was doing. We got into the deep details of the API, and it was like, oh, okay, this, you know, you could kind of understand how this happened.

But at the same time, I remember expressing our worry that Facebook's controls allowed it happen in the first place. We need them to do better. We need better from them. And I suspect that for Liz, well, she's likely a lost cause. If you're wondering whether you might be affected by this massive Facebook data dump, Troy Hunt's Have I Been Pwned site has your answer. And it just got upgraded since I wrote this last evening, and I have the update.

Troy Hunt - and thanks to Simon Zerafa, whose tweet I saw just before the podcast - Troy Hunt has a long tweet thread on Twitter about his recent addition of the leaked Facebook data appearing on the web. I've got a link to the tweet if you're curious, and it's on the screen. Thank you, Jason, or producer, whoever's doing this. The upshot is that the data appears to be not only incomplete, but sparse when it comes to Facebook user email addresses, which is what Troy's Have I Been Pwned site is currently set up to cross-reference. Of the 533-plus million Facebook member records, only 2.5 million include an email address.

What we really want in this case is a phone number, which is far more highly represented in the data. Troy yesterday indicated that he's exploring the possibility of allowing users to search on their phone numbers. Okay. Since then, it happened. He says in his most recent posting over on TroyHunt.com, he said: "The Facebook phone numbers are now searchable in Have I Been Pwned." He said: "The headline is pretty self-explanatory. So in the interest of time, let me just jump directly into the details of how all this works. There's been huge," he has in italics, "interest in this incident. And," he said, "I've seen near unprecedented traffic to Have I Been Pwned over the last couple of days. Let me do my best to explain how I've approached the phone number search feature. Or if you're impatient" - and then he has a link. He says: "What's changed?" And I'll just share the top of this.

He said: "I'd never planned to make phone numbers searchable. And indeed, this user voice idea sat there for over five and a half years without action. My position on this was that it didn't make much sense for a bunch of reasons. First, phone numbers appear far less frequently than email addresses. Second, they're much harder to parse out of most datasets." He says: "I.e., I can't just regex them out like email addresses can be. And, three, they very often don't adhere to a consistent format across breaches and countries of origin." Right. It's famously hard to even use a foreign phone number because how many digits? How is it grouped? Blah blah blah.

So he says: "Plus, when the whole modus operandi of Have I Been Pwned is to literally answer the question Have I Been Pwned, so long as there are email addresses that can be searched, phone numbers don't add a whole lot of additional value." And then I'll conclude with this. He said: "The Facebook data changed all that. There's over 500 million phone numbers, but only a few million email addresses. So greater than 99% of people were getting a miss when they should have gotten a hit. The phone numbers were easy to parse out from mostly well-formatted files. They were also all normalized into a nice consistent format with a country code. In short, this dataset, it was preprocessed by Facebook, after all, all cleaned up and made searchable. This dataset completely turned," he said, "all of my reasons for not doing this on their head."

So anyway, HaveIBeenPwned.com. You can now put your phone number, what you had then, what you have now, various phone numbers, and see if you get any hits on that data. So once again, thanks to Troy for this.

JASON: Yeah, that's good. I haven't been on Facebook in a while, and nothing comes up as being pwned. I did it in no time.

Steve: That's what you want.

JASON: All right. So I must have missed that there is another water security incident that suddenly happened. But that appears to be what you're going to tell me all about.

Steve: Yeah. So we never really got any closure on that hack to the water treatment plant in Oldsmar, Florida. That was the one that, whoa, got everybody's attention. And that one was in early February. But it may have put other similar facilities on alert. Certainly being second with something very much the same gets a lot more egg on your face because it's like, wait a minute, after February, you're saying you guys didn't change your security protocols? You didn't make sure nothing like this could happen again? Well, we were going to.

Anyway, Saturday before last, on March 27th, a young 22 year old named Wyatt Travnichek, living in Ellsworth County, Kansas, is believed to have, well, pretty well confirmed, but we'll call it "belief" at this point, to have broken into a protected computer system belonging to the county's Post-Rock Rural Water District. For some reason, he used his illegal access to shut down the cleaning and disinfecting processes at the facility. You know, you want your water cleaned and disinfected whenever you can get it that way. So turning that off is not good. The public reports don't specify whether any contamination may have resulted to the water supply, but authorities are not taking this lightly at all.

Wyatt was quickly identified and indicted on the serious charges that he had accessed a public water facility's computer system, jeopardizing the safety and health of the residents of the local community. So Wyatt has been charged with one count of tampering with a public water system, which is I guess a thing you can be charged with, and one count of reckless damage to a protected computer during unauthorized access, according to the Department of Justice.

So Lance Ehrig, the Special Agent in Charge of the Environmental Protection Agency (EPA) Criminal Investigation Division located in Kansas, said: "By illegally tampering with a public drinking water system, the defendant threatened the safety and health of an entire community. EPA and its law enforcement partners are committed to upholding the laws designed to protect our drinking water systems from harm or threat of harm." And we're all glad for that. He said: "Today's indictment sends a clear message that individuals who intentionally violate these laws will be vigorously prosecuted."

So Lance's comments suggest that there might be some embarrassment factor and reaction over no one having been identified and held accountable for the previous very high-profile headline-making event in Oldsmar, Florida. Wyatt's indictment does not specify whether his attack was successful, nor how it was detected. But if Wyatt should be found guilty, he faces up to 25 years in federal prison and a total fine of half a million dollars.

The other salient fact here is it turns out Wyatt was not some naive opportunistic post-teenage hacker. The indictment tells that he was previously employed by the water district, and his employment capacity required him to remotely log into the water district's computer system on a regular basis. So he knew what he was doing and how to do it. He quite deliberately shut down the water cleaning and disinfecting process at the facility. Which at least in my mind seems extra creepy and far less prone for sympathy.

At the same time, this also suggests that the water officials at that facility also failed to properly secure credentials by not proactively removing Wyatt's remote access account after he left. They said on whatever terms he was no longer an employee there, and he was still able to access the computer remotely. Duh. So they may have not been able to do so conveniently if, for example, maybe everyone was sharing the same credentials. So that was harder than just changing Wyatt's, if everybody's had to change. But that's no excuse, of course. In any event, let's hope that word of this spreads and that at least our own domestic hackers learn the lesson that messing around with public utilities is not something that the U.S. Justice Department is going to ever take lightly. I think that seems clear.

JASON: And reasonable.

Steve: Yeah, absolutely, yeah. You're an ex-water works employee who logs in and shuts down cleaning and disinfecting? Okay.

JASON: I mean, come on. Why are you doing this?

Steve: Enjoy the water in prison, Wyatt.

JASON: Oh, yeah.

Steve: So now here we are, Jason. Android. Jason knows a little something about Android.

JASON: Little something about Android. Yes, what you got, Steve?

Steve: So on the Android platform, apps have always been able to detect the presence of specific apps, even collecting a full list of installed apps through the QUERY_ALL_PACKAGES privilege. And what's more, apps can be set to receive OS notifications when a new app is installed. That feature, I have to say, really has the smell of, hey, wouldn't it be neat if we let apps be informed when other apps are installed? Just think of all the cool things you could do with that. Unfortunately, as we frequently see, cool things you could do is often quickly turned to the dark side. It doesn't take a rocket scientist, or even a computer scientist, to observe that this wide open facility would provide yet another means for fingerprinting devices and profiling their users.

And it's not just theoretical. Seven years ago, back in 2014, Twitter - Twitter! - began tracking the list of apps installed on users' devices as part of its "app graph" initiative with an aim to deliver tailored content. And the digital wallet company MobiKwik was also caught collecting the information about installed apps in the wake of a data breach that just recently came to light earlier this week. And a study published by a team of Swiss researchers two years ago in 2019 concluded that "free apps are more likely to query for such information, and that third-party libraries are the main requesters of the list of installed apps." The Swiss researchers said that: "As users have on average 80 apps installed on their phones, most of them being free, there is a high chance of untrusted third parties obtaining the list of installed apps."

And a year ago, in March of 2020, another academic study found that 4,214 Google Play apps stealthily collected a list of all other installed apps to allow developers and advertisers to build detailed profiles of users. And you can imagine it's like, hey, Google lets us do it, so it must be okay. Android makes this convenient with two OS function calls, getInstalledPackages() and getInstalledApplications(), which return the list. So it's like, hey, look, they're right there. I'm going to impress my boss by using them and saying, hey, look what we can do now.

JASON: Yeah. If we can, why not?

Steve: Exactly.

JASON: If the system makes it possible, why not? Nothing to stop us.

Steve: Exactly. So the reason we're highlighting this long-running behavior today is because Google is about to clamp down on this cool but overly permissive and abuse-begging, I mean, this begs to be abused. We don't need to know what else the user has. But if we ask, Android will tell. So why not? As they say, you can ask.

So Google's Developer Program Policy March 31 announcement reads: "We're updating the following policies. All new and existing apps will receive a grace period of at least 30 days from March 31st, 2021, unless otherwise stated, to comply with the following changes. Package Visibility, effective Summer 2021. The inventory of installed apps queried from a device are regarded as personal and sensitive user data subject to the Personal and Sensitive Information policy, and the following requirements.

"Apps that have a core purpose to launch, search, or interoperate with other apps on the device may obtain scope-appropriate visibility to other installed apps on the device as outlined below. First what they describe as broad app visibility. Broad visibility is the capability of an app to have extensive, or 'broad,' visibility of the installed apps, the packages on a device."

And they said: "For apps targeting API level 30 or later, broad visibility to installed apps via the QUERY_ALL_PACKAGES permission is restricted to specific use cases where awareness of and/or interoperability with any and all apps on the device are required for the app to function." And you can imagine, you know, things like launcher apps that inherently need access to see all the apps that it would be available to launch would be an example. But not some puzzle game that is free. It's got no need to see anything else.

They said: "You may not use the QUERY_ALL_PACKAGES permission if your app can operate with a more targeted scoped package visibility declaration, for example, querying and interacting with specific packages instead of requesting broad visibility. Also, use of alternative methods to approximate the broad visibility level associated with the QUERY_ALL_PACKAGES permission are also restricted to user-facing core app functionality and interoperability with any apps discovered via this method." They finally said: "Please see this Help Center article for allowable use cases for the QUERY_ALL_PACKAGES permission." Basically, they're really going to lock down on this.

And what they're doing with developers is, and I didn't have this in the show notes, but it was in the broader content that I read, indicating that the developers would have some grace period to remove their request for that privilege unless they could demonstrate they truly needed it. But if it is left in there, and they don't pull it out, those apps are going to get pulled from the Google Play Store. Then they also said at the same level is limited app visibility, where they said: "Limited visibility is when an app minimizes access to data by querying for specific apps using more targeted instead of broad methods, for example, querying for specific apps that satisfy your app's manifest declaration. You may use this method to query for apps in cases where your app has policy compliant interoperability, or management of these apps."

And finally: "Visibility to the inventory of installed apps on a device must be directly related to the core purpose or core functionality that users access within your app." Meaning the whole app visibility can't be some other thing that the app wants to do that isn't about solving the maze puzzle. Maze puzzles don't need to know what other apps you have on the machine. So again, it really needs to be focused. So what we're really seeing is a decision being made, this is not okay. It's truly unfortunate, in my opinion, that we cannot have nice things, or at least that a massively widely used and deliberately permissive system, with all the original features and benefits of Android, is inevitably being slowly whittled down, locked down, and tightened up.

I recall many times hearing Leo so often proclaim that he was using Android explicitly for its deliberately non-Apple freedom, and its openness. He loved all the extra stuff he was able to do. And it's clear that Google and their Android engineering designers had their hearts in the right place. They wanted to create, they set out to create a free and open platform for the world to use. But naive users need to be protected from all the things that their open pocket computer might do that's against their interest and their expectations.

This podcast is titled "A Spy in Our Pocket" due to some new and worrisome research, as I mentioned, from Dublin, Ireland. But there's a much broader sense in which we're all carrying around little spies in our pockets, and this is one of them, the idea that apps were saying, yeah, I'd like to have the QUERY_ALL_PACKAGES permission because I'd like to query all packages. After all, you didn't say I couldn't. So even though I don't need the information, I'd like to have it. Who knows?

JASON: Feels to me like when we look at kind of the way the technology world has kind of continued to expand and grow in the past 10 years that what once was a very noble kind of goal for Google to keep Android this very open platform and to do all of these open things, "open" in air quotes, when compared to a more closed platform like iOS, we're just in a different technology world now. We're at a point now where there's a lot more awareness, even among general users and general consumers, about what it means to have privacy on our data and to protect our data and to protect our security. And I think that our awareness has shifted to the point to where Google needs to make these changes now. Whatever their intentions were for a feature like this before, as noble as they may be, you're right, now there are actors out there that are happy to take advantage of that. And maybe they were there before, and we just weren't as aware about it. But now so many more people are aware of this that it puts pressure on Google to make changes like this. And that's why we're here. That's my thoughts.

Steve: And, you know, just look at all the brouhaha that was stirred up in the early days of COVID by the initiative to do recent proximity via Bluetooth. We did a podcast on it. We looked at the technology. It was beautifully co-developed by Apple and Google, and it didn't matter. Everyone said no, you know. But the fact is, and we'll see this at the end of the podcast, that's nothing, nothing compared to what is actually happening on all of our phone platforms today. The idea that you were near some other phone for a while, well, they know that now without any Bluetooth, without any proximity tracking, without any permissions. All of that is being sucked up.

If they made a mistake, and I'm not saying it was a mistake because of course they had to tell us, but it was just saying we're going to add this feature to our phones for the benefit of the world. No, no. Well, yeah. So we do need, yeah, exactly as you said, Jason, this is not the world that Android was developed in or first conceived in. Things have changed a lot.

JASON: Right. Things have changed a lot, yeah.

Steve: So I'm sure that none of our listeners are cheating with Call of Duty.

JASON: Yeah.

Steve: Nobody listening to this podcast, I know nobody would do that.

JASON: They don't do that.

Steve: No. Now, maybe Paul - no, no. Not even Paul. Everywhere I looked in the tech press this past week I saw mentions of the new malicious game cheats for Activision's Call of Duty: Warzone. The idea of infecting gamers through malicious cheats, it's not new. It's become a longtime persistent and popular means for bad guys sneaking their malicious code into the machines of unsuspecting, though also somewhat less than completely virtuous, game players who are willing to cheat to get ahead in the game.

Still, malware is malware. And the need to obtain an edge, even if not ethically, is inducing gamers to drop their systems' built-in protections. The cheats, that is, the so-called cheats, the cheating software systems instruct gamers to run the installation, the cheat install, as an admin, and of course disable their antivirus. And that's interesting because the gamer knows they're downloading and attempting to run something that's shady in the first place. So to them it might make sense that they would need to give the installer admin rights and of course disable their antivirus system upon which they would otherwise be depending. But in this one case, you know, I need that cheat-y stuff, so let's just let it in.

And as we know, while these voluntary circumventions are often needed for a cheat to work, they also then make it easier for malware to survive reboots and go undetected. And what's actually being installed often is remote access trojans, so-called RAT malware, that opens up a connection off to Russia or wherever and gives the people who perpetrated all this access to those machines. So with all of their guard being down, users won't get warnings of the infection or that software is seeking heightened privileges because they gave the privileges to them to start with.

So Activision noted in their report by writing: "While this method is rather simplistic" - that is, you know, asking for what you want - "it is ultimately a social engineering technique that leverages the willingness of its target, players who wish to cheat, to voluntarily lower their security protections and ignore warnings about running potentially malicious software." Activision also provided a long list of Warzone Cheat Engine variants that installed malware of all descriptions.

And why are there so many? It's being distributed using the increasingly popular affiliate model. Activision's analysis indicated that multiple malware forums are regularly advertising a kit that customizes the fake cheat. The kit makes it easy to create versions of Warzone Cheat Engines that deliver malicious payloads chosen by the soon-to-be attacker who plans to offer it. The people selling the kit advertise it as an effective way to spread malware and "some nice bait for your first malware project." So that's right, script kiddies, you can now - you don't have to understand this. You just set this up and turn it loose.

The sellers have even posted YouTube videos that promote the kit and explain how to use it. So turnkey cheat-ware for a super popular game, packaged for repackaging by affiliates. It might not be worth that endless ammo or speed or invincibility. In any event, while I'm sure that none of our, as I said, our listeners would stoop so low as to something like this, if anyone knows a hot gamer who might, it could be worth dropping them a word of caution in this case. It may not be a free cheat that you're downloading after all.

QNAP. For this podcast I've been trying, successfully, to avoid talking about QNAP, sort like I now try to avoid talking endlessly about ransomware. You notice that I didn't so far, and there was a bunch of stuff that happened. Some crazy ransomware people set the ransom at $40 million for a school district, like a school district has $40 million. They can't even afford pencils these days. So anyway, I don't talk about that because it's just too much of a target-rich environment.

So is QNAP, it turns out. Everybody at this point knows ransomware is bad. It exists. What are you going to do? But every indication is that QNAP, the company, is also bad, and they produce things that are bad, meaning insecure and in many cases insecurable. Yet they are the number one Chinese supplier of Network Attached Storage devices. And they hold, by some accounts, about a 69% market share of the network attached storage NAS market globally. So when something really bad happens and continues to happen, we have to talk about it.

Few things are more important than the security of network attached storage. After all, it's network, and it's attached, and it's storage. Presumably it's storing things that its users might like to keep secure, and perhaps even confidential. And given that the new attacks that we're seeing on networks and corporations are "pivot style," where an attacker first gets into a device on a private network LAN boundary which forms a bridge between the private LAN and the public Internet, and then pivots to use that as the launching pad for attacks into the LAN network to which they now have access, it's not so much that someone's laundry list might become public as that an intruder can now leverage their position on these appliances to launch much more devastating and serious intrusions.

So what happened this time? Security researchers at SAM Seamless Network published their report last Wednesday containing news that they had been sitting on for months, as in four months, in an attempt to be responsible. But QNAP was not. SAM's report is titled "New Vulnerabilities Allow Complete Device Takeover." I've edited and shortened it a bit for the podcast, but basically it reads: "SAM's security research team is actively looking for vulnerabilities in IoT devices that have yet to be discovered in order to ensure our network security coverage is as accurate and up to date as possible. SAM's engine subsequently blocks such vulnerabilities from the first day of their discovery, often prior to the vendor resolving it." And I would say, yeah, if they're blocking on the first day of discovery, they're oftentimes way ahead of any vendors resolving it.

They said: "Below is a summary of two recent vulnerabilities and their potential impacts that our research team discovered in a specific kind of NAS device made by QNAP. It's standard practice to report the discovered vulnerabilities to the vendor and allow for a 90 to 120-day grace period for them to resolve it prior to notifying the public. As seen in the timeline below, we followed the responsible disclosure procedure and immediately reported it to the vendor, especially as the impacts of their exploitation are significant. These vulnerabilities are severe in nature as they allow for full takeover of devices from the network" - meaning public Internet - "including access to the user's stored data, without any prior knowledge."

They say: "We discovered two critical vulnerabilities in QNAP TS-231's latest firmware version 4.3.6.1446, which was 9/29 of 2020. So last year, September. The two vulnerabilities: First, web server allows a remote attacker with access to the web server running on default port 8080 to execute arbitrary shell commands without prior knowledge of the web credentials. Whoops. And their DLNA server allows a remote attacker with access to the DLNA server running on default port 8200 to create arbitrary file data on any non-existing location, without any prior knowledge or credentials. It can also be elevated to execute arbitrary commands on the remote NAS, as well."

So, yeah, their "complete device takeover" titling is no exaggeration. They said: "These may affect other models and firmware versions, as well." And we know how likely that is because most of these use a common base of firmware that they then spread across devices with different port configurations and capacities and so forth. And here it comes. They wrote: "We reported both vulnerabilities to QNAP with a four-month grace period to fix them. Unfortunately, as of the publishing of this article, the vulnerabilities have not yet been fixed." So after 120 days, nothing.

They said: "Due to the seriousness of the vulnerabilities" - and here I salute these guys. This has got to be the new protocol. They said: "We decided not to disclose, even after four months, the full details yet, as we believe this could cause major harm to tens of thousands of QNAP devices exposed to the Internet."

So then under their technical details, which are brief because, as I said, they're not disclosing, they lay out the scenario and note that, even now, they are continuing to be super responsible. Vulnerability number one, RCE (remote code execution) vulnerability, affects any QNAP device exposed to the Internet. They wrote: "This vulnerability resides in the NAS web server running on port 8080. Previous RCE attacks" - remote code execution attacks - "on QNAP NAS models relied on web pages which do not require prior authentication, and run/trigger code on the server side. We've therefore inspected some CGI files which implement such pages and fuzzed a few of the more relevant ones. Most of the CGI files that are available through the web server reside at" - and they give the path name.

"During the inspection, we fuzzed the web server with customized HTTP requests to different CGI pages, with focus on those that do not require prior authentication. We've been able to generate an interesting scenario, which triggers remote code execution indirectly, in other words, triggers some behavior in other processes." And that's all they're saying about it in the hopes that QNAP will awaken from their long QNAP and get this fixed.

They said, under Process for Solving Vulnerability, they said: "The vendor can fix the vulnerability by adding input sanitizations to some core processes and library APIs, but it has not been fixed as of this writing." And they of course disclosed it all, said here's what it is. Here's how you do it. This is horrible. Fix it. So October 12, 2020, full disclosure reported to QNAP security team. Eleven days later, October 23rd, sent another email to QNAP security team. Now we're at October 31st, Halloween. Automatic reply from QNAP Support with a ticket number. So that took, what, 19 days just to get an automatic reply with a ticket number.

January 26th, sent a notification to QNAP about end of grace period, which is planned to end on February 12th. On the 26th, reply from QNAP. Oh, my god, same day. QNAP Helpdesk. The problem is confirmed, but still in progress. Okay, now we're up to February 12th. Grace period has elapsed. They still wait. March 31st, so they gave them from February 12th another six weeks, initial blog post published. That's this. Still not disclosing details. Just like, please fix this, you idiots.

Vulnerability number two, arbitrary file write vulnerability. And as we know, this is in the DLNA server listening on public port 8200. They said the DLNA server is implemented as the process "myupnpmediasvr," and handles UPNP requests on port 8200. What could possibly go wrong with that? Let's put UPNP on the public Internet. They said: "We discovered the vulnerability during investigation of the process's behavior" - I bet they did - "and communication both externally and internally."

They wrote: "We've been able to elevate that vulnerability to remote code execution on the remote NAS, as well." So across the public Internet. Same, they have a shorter disclosure timeline, but same basic idea. So hopefully this public embarrassment may finally work to get QNAP's long-overdue attention. We could hope. No good would come from SAM's more full disclosure. Given the nature of attacks, I'd argue that there would never be any reason for them to disclose. But if they don't convincingly threaten to do so, it's quite clear that these problems are just going to sit there, and QNAP's irresponsible behavior will continue. So the only way to motivate them is to give them time, then to embarrass them. It's hard to imagine any good coming from going further.

As I started out saying, I've been avoiding talking about QNAP. Things keep going by, and I just think, okay, but what are you going to do? But I've already let many similar stories go uncovered. And so please, everyone, take this to heart for your own benefit. When you're choosing a NAS supplier, let's reduce the market share of these people. And if you already have QNAP devices, find some use for it inside your network. Remove it from the public Internet. I don't think there's a safe way to have these things exposed on the Internet.

And before we take our final break, one piece of listener feedback that I really got a kick out of. This is from - his name is The Realest M.F., and he's tweeting from @MaxFeinleib. He said: "Great remark from my college CS professor," so of course Computer Science professor. "There are two hard problems in computer science: cache coherence, naming things, and off-by-one errors."

JASON: Ba dum tss.

Steve: Very clever, because of course that was three things. And it's funny, I mean, I'm a coder. That's what I do. And I completely agree. I've talked about the problem with naming things. Names tend to drift over time. You name something for what you think it is. And later, if you don't remember - a perfect example. I'm deep into SpinRite 6.1 right now. And I saw that decades ago I was calling something a "size," like "drive size." Well, in what? In bytes? In sectors? It's unclear. And I guess my habits have evolved as a coder. I just don't do that anymore. I'm very conscious, drive bytes or drive sector size. Make it clear.

But the other thing you have to be very sure about when you're naming things is that your own use of that thing, once it's named, doesn't expand. So the name, the original name no longer applies. That's another thing that I've noticed can happen over time, where it's like, wait a minute, that's not what that's doing. Maybe once, but not now. I changed it. But I didn't change the name. So that.

And then of course, famously, off-by-one errors. Oh, my god. I mean, what I do is I, like, resort to paper and pencil when I'm needing to make sure that I'm copying a range of bytes from here to there, especially if the range is overlapping. I put my pointer here. Okay, now is it that many things, like the number of things between the start and the end index is not the difference between them, it's the difference plus one, if you're counting the items inclusively. So inclusive versus exclusive and so on. Anyway, anyone who's done any serious coding understands that, oops, that's where a lot of the bugs are. So anyway, got a big kick out of that bit of Twitter feedback, thank you very much.

JASON: All right. The main event where Steve tells us why carrying around a phone in our pocket might not always be the best thing in the world, although it's kind of hard to put that genie back in the bottle. It's kind of what we do now. And for so long. It's our habit.

Steve: And that is what I conclude. But the journey is interesting.

JASON: Yes.

Steve: This week's podcast owes its title to the title of the research paper recently published by Douglas Leith's group at Trinity College in Dublin, Ireland. He's in the School of Computer Science & Statistics there. And I've mentioned before, I have a fond memory, sort of a fleeting memory of Trinity College from my visit to the Dublin chapter of OWASP in September of 2019 to demonstrate SQRL to them. Lorrie and I watched the beautiful Trinity campus pass by while we were attempting to figure out how to open the doors of the metropolitan transit. We never did, so we missed our opportunity to walk around. Maybe someday.

But in any event, Douglas titled his team's research paper "Mobile Handset Privacy: Measuring the Data iOS and Android Send to Apple and Google." Now, needless to say, neither Apple nor Google are pleased about having a third party poking around into their device's communications. They'd rather that no one did. But we're all familiar with the phrase "keeping them honest," which I think applies here. It's not at all that either of these behemoths are attempting to do anything explicitly nefarious.

I would argue that it's more likely the case, and recent history with all forms of computing teaches us, that without adequate supervision, and especially when an entity imagines that no one will ever look, the strong tendency is to collect data less because it's truly needed than because it's there. And today bandwidth, storage, and computation might as well be free, so slurp it all up just in case, and we'll see whether we might actually have any need for any of it later.

We were just talking about the case of the Android app packages installed. It's like, well, Android offers an API that lets me get a list of all the apps that are installed on the phone along with me. Why not use it? Right, except Google says no, no more. You need to prove you need it. In this case, they're not taking their own advice, but we'll get there. So I'll also note that one thing these researchers do is notice how this information could be used and potentially abused. Not accusing anybody of that, no. But they're seeing what is being collected.

So, okay. Against that backdrop their paper's abstract summarizes what their independent analysis found. They said: "We investigate what data iOS on an iPhone shares with Apple and what data Google Android on a Pixel phone shares with Google. We find that, even when minimally configured and the handset is idle, both iOS and Android share data with Apple and Google on average every 4.5 minutes. The phone IMEI, the hardware serial number, the SIM card serial number and IMSI, handset phone number, et cetera, are shared with Apple and Google.

"Both iOS and Android transmit telemetry, despite the user explicitly opting out of this. When a SIM is inserted, both iOS and Android send details to Apple and Google. iOS sends the MAC addresses of all nearby devices, other handsets and the network gateway to Apple, together with their GPS location. Users have to opt out of this, and currently there are few, if any, realistic options for preventing this data sharing."

Again, okay, they're not saying that there's anything inherently wrong with doing this. But someone needs to ask why. Why are they collecting this data? What's being done with it? Certainly as technologists we can readily envision many things that we would not like to have done with all this information. So if Apple and Google are not doing any of those worrisome things, what are they doing? And if they're doing nothing with any of that information, then why is it being collected?

Certainly some of that is truly essential to, as we know, the baseband operation of any cellular radio system. We know that the nature of cellular service requires an ongoing back-and-forth dialogue with nearby cell towers. Even when you're not using your phone, you can see how many bars of reception you have; right? That's all going on in the background. But none of that should involve any data above the level of cellular connection maintenance. Or if it does, documentation and disclosure should be available. Yet none is. Thus research is required.

The researchers organized their presentation very nicely in layers of successive detail, with each layer going down to provide more specifics. So I want to share just their top level of detail, which is all we need to fill in the relevant facts for our purposes. And for what it's worth, I've got a link to the PDF here in the show notes. And, I mean, it shows you the detailed protocol back and forth handshaking nitty-gritty. So it's all there. I've lightly edited the presentation for better delivery through this podcast, and I've included a link to their full research paper. So again, anybody who wants to know everything, can.

At their top level they wrote: "In this paper we investigate the data that mobile handset operating systems share with the mobile OS developer, in particular what data iOS on an iPhone shares with Apple and what data Google's Android on a Pixel phone shares with Google. While the privacy of mobile handsets has been much studied, most of this work has been focused on measurement of the app tracking and advertising ecosystem, and much less attention has been paid to the data sharing by the handset OS with the mobile OS developer.

"Handset operating systems do not operate in a standalone fashion, but rather operate in conjunction with backend infrastructure," meaning maintaining a constant link to the mothership. They said: "So, for example, handset operating systems check for updates to protect users from exploits and malware, to facilitate running of field trials, for example, to test new features before being rolled out, to provide telemetry and so forth. Hence, while people are using an iPhone, the iOS operating system shares data with Apple; and when using a Pixel, the operating system shares data with Google. And this is part of each device's normal operation.

"To allow direct comparisons, we define experiments that can be applied uniformly to the handsets studied that generate reproducible behavior. Both Apple and Google provide services that can be, and almost always are, used in conjunction with their handsets, for example, search (Siri and OK Google); cloud storage (iCloud and Google Drive); maps and location services (Apple Maps and Google Maps), photo storage and analytics (Apple Photo, Google Photos)." They said: "We endeavor to keep these two aspects separate, to focus on the handset operating system in itself, separate from optional services like those.

"We assume a privacy-conscious but busy non-technical user who, when asked, does not select options that share data with Apple and Google, but otherwise leaves handset settings at their default value." In other words, given a choice, no. But they're not going to go dig in and try to turn everything off. "In these tests we evaluate the data shared, one, on first startup following a factory reset." So first startup, absolutely clean phone. "Two, when a SIM is inserted or removed. Three, when a handset lies idle. Four, when the settings screen is viewed. Five, when location is enabled and disabled. And, six, when the user logs in to the pre-installed app store.

"We note that these tests can be partly automated and used for handset operating system privacy benchmarking that tracks changes in behavior over time as new software versions are released." They said: "The following table summarizes the main data that the handsets send to Apple and Google." And there's then a multicolumn chart with two lines, Apple iOS on the top line, Google Android on the second line.

And pretty much everything is the same, although Google does not have location checked where Apple does, nor local IP address, which is interesting, not being collected, whereas it is on iOS. And then device WiFi MAC address not sent by Apple, apparently is by Google. And nearby WiFi MAC addresses not being sent by Android is being sent by the iPhone. So otherwise, IMEI, hardware serial number, both of them. SIMs, both of them. Phone numbers, any of that kind of stuff, basic underlying cellular technology you can imagine needing to be sent.

And they said: "This data is sent even when a user is not logged in," and they said, "indeed, even if they have never logged in. In addition to the data listed in this table, iOS shares with Apple the handset's Bluetooth Unique Chip ID, the Secure Element ID" - which is associated with the Secure Element used for Apple Pay and contactless payment and, as we know, the Secure Enclave is like everything - "and the WiFi MAC addresses of nearby devices, in other words, of other devices in a household of the home gateway." And of course we know that WiFi is Ethernet, and Ethernet uses IP packets contained in Ethernet packets, and that the Ethernet system uses MAC addresses in order to address the packets on the Ethernet. So you're getting a huge amount of information.

They said: "When a handset's location setting is enabled, these MAC addresses are also tagged with their GPS location. And it takes only one device to tag the home gateway MAC address with its GPS location, and thereafter the location of any other devices reporting that MAC address to Apple is thus revealed," right, by social graphing, connectivity graphing. "Also note that sharing of these WiFi MAC addresses allows linking of devices using the same network, in the same household, same office, same shop, cafe," whatever, "and so the construction of a social graph over time and place.

"Both iOS and Google Android transmit telemetry, despite the user explicitly opting out of this. However, Google collects a notably larger volume of handset data than Apple. During the first 10 minutes of startup, the Pixel handset sends around a megabyte of data to Google, compared with the iPhone sending around 42K of data to Apple. When the handsets are sitting idle, the Pixel sends roughly a megabyte of data to Google every 12 hours, compared to the iPhone sending 52K to Apple. Google collects, they conclude, around 20 times more handset data than Apple. Not clear how relevant that is. You'd have to look at the data in detail to see whether you care.

And you just had on the screen and also in the notes are two charts showing at startup for Android a very steep upward climb, and then it levels out. That's at startup. Whereas Apple pretty much stays at a low purr. And then idle, similarly, Android is just sending data out, I mean, like lots of data, more or less continuously over the course of idle time; whereas Apple continues at a lower purr. At the same time, I'm unhappy with what we know of some of what Apple is sending. That just seems really unnecessary.

So they said: "In 2020 it is estimated that in the U.S. there are 113 million" - so that's last year - "113 million iPhone users and 129 million Android users. Assuming all of the Android users have Google Play Services enabled, then scaling up our measurements suggests that in the U.S. alone, Apple collects around 5.8 gigabytes of handset data every 12 hours, while Google collects around 1.3 terabytes of handset data in the same period of time. When the handset is idle, the average time between iOS connections to Apple is 264 seconds, while Android connects to Google on average every 255 seconds," so that's about comparable. In other words, both operating systems connect to their back end servers on average every 4.5 minutes, even when the handset is not being used.

So yes, Jason, that handset that may or may not be in your pocket, mine which is sitting over on a charger, that is otherwise in my pocket or on a charger, we're not doing anything. And every 4.5 minutes it's connecting back to the mothership and sending a bunch of stuff that is clearly not necessary for them to send. And I guess the point of showing us how this accumulates, 1.3 terabytes of handset data for Android in aggregate every 12 hours, 5.8 gigabyte for iOS, why? Are they keeping it? Are they storing it? Are they processing it? Are they parsing it? What are they doing? And again, this is not cellular layer tower connections. This is connections back to Apple and Google.

They said: "With both iOS and Android, inserting a SIM into the handset generates connections that share the SIM details with Apple or Google." Okay, fine. "Similarly, browsing the handset settings screen generates multiple network connections to Apple or Google," which is interesting. I guess that's telemetry; right? Like oh, we want to collect all this data on what the user's looking at. Okay. "A number of the pre-installed apps and services are also observed to make network connections, despite never having been opened or used. In particular," they wrote, "on iOS these include Siri, Safari and iCloud; and on Android these include the YouTube app, Chrome, Google Docs, Safetyhub, Google Messaging, the Clock, and the Google Search bar." Maybe the clock they want to make sure it's set correctly.

"The collection of so much data, they write, by Apple and Google raises at least two major concerns. First, this device data can be fairly readily linked to other data sources. For example, once a user logs in, as they must to use the pre-installed app store, then this device data gets linked to their personal details - name, email, credit card, et cetera." Okay, that's obvious - "and so potentially to other devices owned by the user" - oh, that's true, that's not so obvious - "shopping purchases, web browsing history and so on. This is not merely a hypothetical concern since both Apple and Google operate payment services, supply popular web browsers, and benefit commercially from advertising.

"Second, every time a handset connects with a back-end server, it necessarily reveals the handset's IP address, which is a rough proxy for location. The high frequency of network connections made by both iOS and Android on average every 4.5 minutes therefore potentially allow tracking by Apple and Google of device location over time. With regard to mitigations, of course users also have the option of choosing to use handsets running mobile OSes other than iOS and Android." What, Blackberry? They said: "For example, e/OS Android." I'm sure you know what that is, Jason. I've no idea.

They said: "But if they choose to use an iPhone, then they appear to have no options to prevent the data sharing that we observe." Of course a firewall if you were to configure it correctly. But if you're cellular, then no. "But if they choose to use an iPhone, then they appear to have no options to prevent the data sharing that we observe. They are unable to opt out. If they choose to use a Pixel phone, then it is possible to start up the handset with the network connection disabled to prevent data sharing, then to disable the various Google components, especially Google Play Services, Google Play Store and the YouTube app, before enabling connection to the Internet.

"In our tests, this prevented the vast majority of the data sharing with Google, although of course it means that any subsequent phone apps must be installed via an alternative store and cannot depend upon Google Play Services." They said: "We note that many popular apps are observed to complain if Google Play Services is disabled. However, further testing across a wider range of handsets and configurations is needed to confirm the viability of this potential mitigation. When Google Play Services and/or Google Play Store are used, this mitigation is not feasible and the data sharing with Google that we observe appears to be unavoidable."

So they conclude this top layer with: "Ethical disclosure: The mobile OSes studied here are deployed and in active use. Measurements of Google Play Services backend traffic were previously disclosed by our group, but the present study is broader in scope. We informed Apple and Google of our findings and delayed publication to allow them time to respond. To date, Apple have responded only with silence." He said: "We sent three emails to Apple's Director of User Privacy, who declined even to acknowledge receipt of an email." I guess Apple doesn't have one of those emergency response coordinator people that Facebook has. Anyway. "And," they said, "we also posted an information request at the Apple Privacy Enquiries contact page [apple.com/privacy/contact], but we have had no response.

"Google responded with a number" - I mean, these are not bozo security researchers. I mean, this is true, honest research being conducted by a useful group who've done things before. Anyway: "Google responded with a number of comments and clarifications, which we have incorporated into this report. They also say that they intend to publish public documentation on the telemetry data that they collect." And I'll mention, it's not in here, but there is also Google disputing the 20-factor greater than iOS claim that these guys found. Google didn't like that.

JASON: I'm sure.

Steve: And they finished with: "A key consideration is what mitigations are possible, and on what time scale can they be deployed. It seems likely that any changes to Apple iOS or Google's Android, even if they were agreed upon, will take a considerable time to deploy while keeping handset users in the dark for a long open-ended period, which seems wrong."

So my take on this is that, as we said at the top, Jason, the mitigation of this data collection, as they described, is largely impractical. No one's going to go through all those hoops and jumps and so forth. And the only way to truly avoid it is to choose not to use a mobile handset. And who's going to do that in this day and age? These devices have become our science fiction, globally connected, super powerful pocket computers. I mean, they're amazing. But they're also inextricably tethered to their mothers, and there's no breaking that bond. We know that all of the privacy downside the researchers painted for us is completely feasible. We don't know what's being done with the data. Why is Apple sending back all the MAC addresses of the other devices that it sniffs in our environment? That just seems gratuitously unnecessary.

Are Apple and Google actually performing all of that linking to build sophisticated social graphs or device proximity connection histories, at a low level that's inaccessible to us? They certainly could be. Sharing our phone's IMEI and SIM serial number seems benign and likely necessary to provide the services we need. But there's really no way to characterize the aggregation and forwarding of every MAC address within the phone's reach, on an ongoing active basis, as anything less than surveillance.

Apple may boast and they certainly do, and use it as a selling factor that they are unable to see inside a locked phone. But they certainly have the ability to tell law enforcement everything that iPhone has seen, everyone it's been near to, what the proximity patterns are, its location history through time. And that's a pretty serious encroachment into every iPhone user's privacy. I won't be giving up my phone, and I'm sure that very few will. No one, probably. But it might behoove us to keep in mind that we are each carrying a little spy in our pocket.

JASON: Indeed. And you mentioned earlier, or rather the report mentioned e/OS. That's more like an open source - there's open source versions of Android that are de-Googled entirely, and that's one of them, basically removes Google entirely - and my Google just started up - entirely from the phone so you don't have any of the services; you don't have any of those interlocking pieces. You also don't have the Play Store.

There are ways with Android to basically remove Google from the equation for the most part. But, I mean, only certain people are going to even be aware to do that. Or certain people are even going to care enough to do that. A lot of people I think nowadays have just kind of resigned themselves to the fact that, yeah, you know what, this is just the cost of doing business. The cost of having a smartphone that does all the things I want it to do is that it collects this data. And I'm not even so sure that that data collection is nefarious. But whatever. It's already done, and it's too late to put it back; you know? I don't know. I think there's a lot of stuff that happens.

Steve: One of the things that we've been talking about recently is the tech-savvy social responsibility to protect those who cannot protect themselves. And so, for example, web tracking. We talk about web browser tracking. Well, all the techies know about web browser tracking. Most users don't. So as techies, don't we have an obligation to make the world safer for everybody, even if they don't have all the details about how this happens? And I think there's a useful case to be made for the fact that, yeah, this should be fixed. And if nothing else, Apple should tell us. Tell us why you're collecting all this. You're doing it behind our backs. There's no disclosure of this. What are you doing with it? And I would argue, if we knew, then that would be fine.

JASON: Yeah, well, it would certainly be fine for some people. That would be enough.

Steve: Oh, better, it would be better.

JASON: Apple's got a tight collaboration of all of their devices working together. This is just a component. This is what enables it. Great. That's one of the benefits that I see in the Apple ecosystem, and I'm okay with it. But, yeah, other people, it's like, any of that collection whatsoever, it's not okay. And that's part of the big challenge, too; right? There's just so many varying degrees of comfort level with people and their data right now. The big companies have to kind of guess what is the sweet spot, and they have to keep themselves in business, too. So I don't know what the right answer is.

Steve: I'm glad for the research and the disclosure because, as I said, there's a tendency, for engineers especially, well, no one will know what we're doing, so why not? We've got lots of bandwidth. We've got lots of storage. Maybe we'll be able to use this someday. Maybe we'll do something. So why not? Well, okay. Being called on the carpet for the data collection you're doing is a why not. Because here's Apple, all jumping around, selling privacy; right? I mean, Tim Cook makes a huge deal of it. Well, okay. Except what about this? Oh, well. You know? So if they're not using it, they should shut it off. And if they are using it, then what for?

JASON: Yeah, and what you just said there, for whatever reason, I hate that thought of, like, we're collecting the data because we can, not because we have any use for it now, but we might have use for it later. I feel like that's a Pandora's Box that you open up, and that leads to potentially very bad things down the line. So absolutely right.

Steve: Right.

JASON: Indeed. Good stuff, Steve. Appreciate all that you do in keeping us all informed on all of this. Everybody who wants to follow what Steve is doing even closer can go to GRC.com. Everything that Steve's up to can be found over there. Of course SpinRite, which we didn't talk about today, but it's an awesome hard drive recovery and maintenance tool. You can get your copy there. Information about SQRL, audio and video of this show found at GRC.com. Also transcripts of this show can be found over there. You've got a lot going on at GRC.com, and everybody should check it out, so please do.

If you want to hit our site for this show, pretty easy to find, TWiT.tv/sn for Security Now!. We host audio and video versions of this show, ways to subscribe to the podcast, about the YouTube, if you like. It's all there. We do record this show, well, normally Leo records it. Today I am with Steve. But we record every Tuesday starting at 1:30 p.m. Pacific, 4:30 p.m. Eastern, 21:30 UTC. And you can check it live, if you like, TWiT.tv/live. We've got an active chat audience who has been participating throughout this entire show behind the scenes, as well. But we've reached the end of this episode, Steve. Thank you so much for letting me join you on today's episode. Appreciate it, man.

Steve: Great to have you, Jason. And we'll see you next time Leo wanders off somewhere.

JASON: That's right, that's right. I will happily fill in for Leo next time. And Leo and Steve will see you next time, next week on Security Now!. Bye-bye, everybody.

Steve: Bye.