DuckDuckGone?

Description: This week we examine the difficult to believe in 2022 design of Australia's New South Wales Digital Driver's License which was sold as being quite difficult to counterfeit. We examine the latest, once again fumbled, extremely pervasive Microsoft Office zero-day remote code execution vulnerability. We look at the first instance of touchscreen remote touch manipulation, and at Vodafone and Deutsche Telekom's difficult to believe yet already being piloted plan to further monetize their customers by somehow injecting persistent supercookies into their customer's connections at the carrier level. Then, after sharing some feedback from our terrific listeners, we'll dig into the discovery that the DuckDuckGo Privacy Browser carved out a privacy exception for Microsoft.

High quality (64 kbps) mp3 audio file URL: http://media.GRC.com/sn/SN-873.mp3
Quarter size (16 kbps) mp3 audio file URL: http://media.GRC.com/sn/sn-873-lq.mp3

SHOW TEASE: It's time for Security Now!. Steve Gibson is here. Lots to talk about. Australia's ridiculous Digital Driver's License is completely insecure. Wait'll you hear the details. A zero-day in Microsoft Office that Microsoft denies. And then we'll talk about DuckDuckGo and the controversy surrounding Microsoft tracking built in. Steve breaks it down. It's all ahead next on Security Now!.

Leo Laporte: This is Security Now! with Steve Gibson, Episode 873, recorded Tuesday, May 31st, 2022: DuckDuckGone?

It's time for Security Now!, the show where we protect your security, your privacy online, thanks to the good offices of Mr. Steve Gibson, the man about SGgrc. Hello, Steve.

Steve Gibson: Hello, Leo.

Leo: Good to see you.

Steve: Great to be with you again this last day of May. I meant to squeeze in that it was our penultimate episode last week, penultimate episode of the month.

Leo: You just like to use that word, don't you.

Steve: Well, now that I know what it means, it comes in handy.

Leo: Very handy.

Steve: Yeah. So okay. The winner of the most tweeted "What do you think about this, Steve?" question was what's this about DuckDuckGo, like, doing some sneaky business behind the scenes, under the covers, out the back door, whatever. Turns out it's not probably such a big deal, although one could argue they just screwed up the communications.

Leo: That's the big deal. They should have told us. If they want to be a privacy browser, you've got to tell us that you're letting Microsoft track us.

Steve: Yes. In fact, it reminded me so much what was behind my original, you know how you used to say that I coined the term "spyware," because I found some before there was that term on my computer. It was this Aureate was the name, and they were...

Leo: Oh, yeah. Wow, there's a blast from the past.

Steve: Oh, my god. And it was not a good thing, but it wasn't really that bad except that it was a secret. And so when people learned that this had been going on behind their backs, it was much more the fact that it felt sneaky and wrong. Well, it's like this catastrophe in Texas, where the sheriffs got the facts so wrong in the beginning that now everyone's like, well, what really happened? You keep changing your story. So nobody knows.

Leo: Right, right.

Steve: Anyway, so we're going to talk about that. As a consequence, this episode is titled "DuckDuckGone?" But we're going to first examine the difficult to believe in 2022 design of Australia's New South Wales Digital Driver's License, which was sold to the public as being quite difficult to counterfeit. It's just, you know, it's the perfect topic for this podcast because it is so bad, and it's just unbelievably, like some guy's, some bureaucrat's nephew designed it or something. He said here, you know, design us a digital wallet. A digital driver's license is so bad. Anyway, we're going to look at that.

Then we're going to examine the latest, and unfortunately once again fumbled extremely pervasive Microsoft Office zero-day remote code execution vulnerability which, if I didn't already say this, is really bad. We look at the first instance of touchscreen remote touch manipulation, like spoofing touches on a touchscreen without actually touching the screen. They've done that now. And also at Vodafone and Deutsche Telekom's difficult to believe, yet already being piloted plan to further monetize their customers by somehow - and how is a real question, we'll get into this - somehow injecting persistent supercookies into their customers' connections at the carrier level.

Then, after sharing some feedback from our terrific listeners, we'll dig into the discovery, as I have mentioned, that DuckDuckGo, the DuckDuckGo Privacy Browser, of course it's called that, apparently carved out a privacy exception for Microsoft.

Leo: Fascinating, yeah.

Steve: So I think another great podcast for our listeners. And we do have a Picture of the Week that any coder will appreciate. So our Picture of the Week is a three-frame cartoon. And I gave it the title "Been There" because yeah. So the first frame shows this happy coder. He's tap tap tapping away on his keyboard, big smile on his face, and all's going well. Second frame, he kind of pushes away from his computer, leans back in his chair, same big smile on his face, and we see him saying "Perfect. I'll finish this on Monday," looking at the code that he's just written. Last frame of the cartoon, it's Monday morning. Well, the chair's been knocked over. He's grabbed his computer, and he's shaking it in the air, saying, "What does this mean?" Meaning that yes, over the weekend he lost the flow, he lost the mindset, he lost the context. That code that made perfect sense to him Friday evening when he finished, not quite sure what it does Monday morning.

Leo: Or five minutes later, frankly.

Steve: Well, and I've, yeah, I've referred to this through the years that my term, my own term for it, I call it "switching cost." For me it's that there's a significant cost associated with switching from one project to another. And over the years, with the wisdom of age, I've grown to appreciate that. And in fact what I have been doing for the last maybe week has been going back into SpinRite's source and carefully documenting why I did the things that I did. Because as always is the case, and I'm sure anyone who's done much coding knows this, when you first write the code, you create comment blocks, and you explain your theory and so forth. And then something doesn't work right.

So you go, oh, okay, let me just try this. And so you make a change; and oh, it works. But then something else needs fixing somewhere else. So you go over there, and you work on that. And so what inevitably happens is you're jumping around in the code and often trying things for the moment to see if this is going to do it or not. Now, of course this may be more specific to SpinRite because in this instance I've got, I don't know, a hundred people testing the code against hardware, sending back feedback about yes, that fixed the problem; or, no, that didn't. And I say, okay, let's try this again.

So it's hugely dynamic and interactive. But what inevitably happens is, and what did happen, was I ended up with code which works perfectly for everybody in every instance. And I'm very proud of it. Well, at this moment in time I still know what it is. I know what it does. I know why I made those changes. From experience, I absolutely know that six months from now, like whenever it is that I have to come back to it, I will have no idea what it is. I mean, and it's weird to say that because at this moment I embody that code. I mean, I know what every single line does, why there's like some weird exception here that does something funky, and why it was necessary.

And so it's difficult to imagine that at some point in the future I am going to be a different person who looks at that and goes, what the heck? So what I've learned is that right now is the time to capture the knowledge that I have before I switch because I'm about to, as I've talked about, switch to working on the backend of this, which uses all that code, but through a layer of abstraction specifically to insulate the back end, or I guess, wait, the frontend rather, the frontend that uses the code on the backend to...

Leo: You can't even remember which end the code comes out.

Steve: Anyway, I love that somebody created a cartoon for this because, yes, it does capture the essence of - I think the reason coders like us love to code is that it really can tax our brain more than anything else. It's like as much as I can give it, it will take. And I love that about it. But it'll take it back if you're not careful to hold onto it.

Leo: This is why as I get older I like functional code better and better because you write small routines that always produce the same result. When the same stuff goes in, the same stuff comes out. You can document in Lisp is nice because you can have a code string. I guess you could do that in Python, too, where it's the second line of the function. You say, this is what this function does. And even if I don't understand why the function works, I've tested it. I know it works. I know it does what it says it'll do. So I know that component. And that's I guess the same idea behind encapsulation and object-oriented coding is that once you make an object, it's a black box, and you don't have to remember how it works as long as you know and can trust that it works; right? But you're doing it in assembly. I don't even know how you could - I guess you could write in that style in assembly. People do.

Steve: And actually, as has often been observed, my assembly code looks much more like high-level language than most assembly code.

Leo: You use macros; right?

Steve: Well, yes. I use macros, and also Microsoft's macro assembler has if-then-else loop while, I mean, it's got high-level language control flow constructs. They have zero overhead. That is, the if is just a branch instruction, but it will intelligently do, like build an expression for doing compound testing in a single line. So I've seen other people's assembly is just like opcodes running down the left-hand margin of the page.

Leo: That's a recipe for disaster, yeah.

Steve: And you just look at it, and it's like, what is this? I mean, it's ugly looking. Who could be proud of that? But some jocks think that that's...

Leo: Hey, it works, man. It works. Well, that's what, yeah, see, and that's exactly - I think sometimes older coders probably are better, not just from experience, but because they can't trust themselves to remember the complex spaghetti that they wrote. So they have to rely on...

Steve: They've learned not to.

Leo: They've learned not to, yeah, yeah.

Steve: So anyway, love that cartoon. Okay. So get a load of this. This piece was stiff competition for this week's main story. But as I said, it lost out to an explanation of what was discovered about the operation of DuckDuckGo's Privacy Browser. So I decided to lead with this one as the runner-up because, wow, it's just perfect for this podcast. A penetration testing and secure app development group known as Dvuln (D-V-U-L-N)

recently took a close look at the somewhat new, it was released in 2019, so three years ago, New South Wales government's Digital Driver's License (DDL). And putting it mildly, they found its security to be wanting. They documented the system's various troubles in a blog posting which was titled "ServiceNSW." ServiceNSW is NSW is New South Wales. They said: "ServiceNSW's Digital Driver's License Security Appears to Be Super Bad."

So, okay. To set the stage, they explain in their posting: "In November 2019, the New South Wales government (ServiceNSW) introduced the Digital Driver's License, or DDL for short, as a means to make it easy for people to access a digital version of their driver license. Upon the launch of ServiceNSW's Digital Driver's License there were multiple security researchers who publicly reported a number of security issues including but not limited to the ability to manipulate digital license data and create fraudulent digital identities." What?

They said: "As far as we can see, there appears to be no formal public response from ServiceNSW regarding the acknowledgement or remediation of these issues. As of February 2022, according to the Minister for Customer Service, there have been 3.9 million people who have opted-in for the Digital Driver's License. To put this into perspective, we can assume around 70% of people in New South Wales use and trust the Digital Driver's License as a means of identification and verification in their day-to-day lives." Meaning that 3.9 million people is 70% of the population of New South Wales covered by this DDL.

They said: "During Dvuln's analysis of the ServiceNSW mobile application" - it's for iOS - "we discovered that due to the existence of several security design flaws, it is still possible" - meaning today - "for malicious users to generate fraudulent Digital Driver's Licenses with minimal effort on both jailbroken and non-jailbroken devices without the need to modify or repackage the mobile application itself." So of course that would be one thing, right, to create a fake Digital Driver's License app which could show anything you want it to show. But no. Don't have to do that. Use the real app and just change the data. And it doesn't care.

Okay. So back in 2019, not long after this DDL (Digital Driver's License) first appeared, during a security conference a security researcher, as part of his conference presentation sort of on a larger topic of digital identity security, he demonstrated to the audience, in public, his ability to modify this New South Wales Digital Driver's License details locally on his mobile device, causing it to display false information. And displaying accurate information is the whole point because you, like, show your phone to somebody, and they go, okay, yeah, that's you. And, oh, look, you're 21 years old. Go ahead. Buy alcohol. Go into the club, whatever.

And although during his talk he mentioned that he had reported these troubles to ServiceNSW, the New South Wales government, there were no apparent public updates on the matter since then. So it's unclear whether these bugs were considered an accepted risk - which okay, this is 2022, everybody, that's insane if that's the case - or if any sort of remediation was ever attempted by the presiding government.

So now we jump forward three years to 2022 where there are rumors circulating regarding underage people using false digital licenses. No. Underage kids could be spoofing their digital identities? No. The Dvuln posting contained an authentic-appearing Twitter posting made on November 25th, 2021 where the poster is annoyed that a bouncer at a club denied access to one 18 year old when others using fake digital licenses are apparently regularly admitted.

I have the tweet in the show notes for anyone who's interested. This was posted by Sydney 2100 is the Twitter name. And this was sent to @TheSteyneHotel. And the tweet reads, "18 year old went there last night with three forms of ID, and you wouldn't let him

in because you don't count a physical New South Wales driver's license as valid ID. Really?" He says: "I know 10 kids that you let in regularly with fake digital licenses because they are easy to make. No idea." Meaning you have no idea what you're doing. So apparently this 18 year old had a physical real-world, like old-school plastic New South Wales driver's license, and the guy said no.

And actually, Leo, completely off topic, I may have mentioned this before, it seems familiar, that Lorrie and I were renting a car, I don't know, a couple months ago, to move some stuff out of her parents' condo in L.A. So we got the biggest, it was a Yukon or something, thing that we could rent. And it turns out in order to do that I had to have a banking application on my phone which showed my name. And, I mean, I had driver's license, I think I may have even had my passport with me. But, I mean, I had a wallet full of credit cards. I had, you know...

Leo: You needed this Hawaiian Driver's License for McLovin. If you'd had this, everything would have been simple.

Steve: Unbelievable. I actually...

Leo: That's bizarre.

Steve: And I said to this little young gal at the terminal, she said, oh, everything was fine. She said we need one more thing. You need to show me a banking app, like Chase or whatever, Visa or something, for your account with your name on it.

Leo: They didn't tell you that ahead of time? That's bizarre.

Steve: No. Didn't tell me ahead of time. It was apparently new. And as it happened, I think I had one, or I have an account with Chase, but I didn't have their app. So I had to...

Leo: Download it.

Steve: ...install it, log in. Then I had to go to LastPass and get LastPass to log me in to Chase in order - it was like, oh, my. Anyway, point is, apparently, no. Now we're believing digital over real-world old-school physical.

Leo: That's bizarre.

Steve: Anyway. Okay. So...

Leo: That's good to know. I'll make sure to keep my banking app on my phone, yeah.

Steve: Yeah. And it's funny, too, because - oh, and actually she finally just said, "Okay, never mind." I was making such a valiant effort to do this and kept getting stuck for reasons like my own security was getting in the way. She said, "Okay, never mind."

Leo: Right. Oh, for crying out loud.

Steve: So as we're walking to the car...

Leo: You must be Steve Gibson.

Steve: As we're walking to the car, she says, you know, and I'm saying to her, I said, "Well, you know, I guess I'm just old, and that's why I don't have, like I don't do banking on my phone, like, ever. I'm not going to do banking on my phone." And she says, "Yeah, well, you are old. Because," she said, "everybody I know has banking apps on their phones."

Leo: Yeah, yeah.

Steve: And I said, well, okay.

Leo: Yeah.

Steve: Okay. So not having seen the IDs themselves, that is, this particular tweet and what the guys were referring to, the Dvuln guys wrote about this Twitter posting, they said: "We cannot confirm whether or not they were exploiting the poor security design or similarly using a static photoshopped image." Because again, if you're just showing like a screenshot, it could be faked; right? Like, you know, just Photoshop what this digital ID is showing, and that's what you present. So clearly there are problems with the whole concept of, like, so many ways you could do a digital ID incorrectly. Anyway, he says: "At the same time, although due to ease of exploitation, it is entirely possible that these kids were using the same method detailed in this blog."

Okay. So the Dvuln guys quoted one of the security claims made about the new Digital Driver's Licenses. They said: "According to a press release from the New South Wales government, the Digital Driver's License implemented is 'hosted securely on the new ServiceNSW app, locks with a PIN, and can be accessed offline,' and 'will provide additional levels of security and protection against identity fraud, compared to the plastic driver's license.'"

And the Dvuln guys explained that, in fact, real world physical driver's license counterfeiting is actually far more difficult than spoofing the content of a physical driver's license. They wrote: "Given the Digital Driver's License's current state of security" - by which they mean, as we'll see in a minute, shocking lack of security - they said: "We believe it would be far more difficult for an average fraudster to obtain the equipment necessary to produce high-quality plastic NSW driver's licenses. A fraudster would need to source and obtain hardware such as but not limited to, a card printer, New South Wales holographic security foil, and other security features developed uniquely for the New South Wales identification cards such as the middle green layer, none of which are commercially or legally available outside of the printing hardware."

So, yeah. Like, you know, if you've looked at your driver's license, it's got all kinds of wacky stuff going on now. Weird reflected angles and things embedded in different layers and stuff that's shiny. It's like, okay, you can imagine duplicating that physically is not going to be easy, compared to, what, taking a screenshot of the secure ID on the phone?

Okay. So what are the specific problems with New South Wales Digital Driver's License? Okay. First off, the DDL-stored license data is encrypted, but not very well. On iOS, the Digital Driver's License data is stored in a JSON-formatted file, which is encrypted using AES-256-CBC. So 256-bit AES cipher, that's state of the art. Cipher Block Chaining mode, CBC, that's fine. And then that's combined with Base64 encoding. Nothing wrong with any of that. The encryption will turn anything, even if it was ASCII textual content to begin with, into binary data. So the Base64 encoding converts that back into ASCII text so that it can be stored in a JSON text format file if desired.

But here's the problem. The encryption key is the four-digit PIN that's initially set during onboarding when the user first installs and sets up the app. That's it. A four-digit PIN. Believe it or not, that's the password. A four-digit PIN. Since a four-digit numeric PIN can be anything from 0000 to 9999 there's, gee, let's see, 10,000 possible PIN combinations. One of those will correctly decrypt the JSON file. And the use of the correct decryption will be readily apparent because what's not decrypted correctly, that is, with the wrong PIN guess, will be gibberish. In other words, if an attacker is able to obtain the encrypted data, either through accessing an iPhone backup, direct access to the device, or a remote compromise, it will take only a few minutes to brute-force the correct four-digit key to that encryption. There's not even any password-based key derivation function, you know, a PBKDF, to slow down the guessing. Nope, just use the four-digit PIN, see if that decrypts the blob back into something that looks correct; and if not, try the next one.

During Dvuln's testing, their brute-force process took only a few minutes to decrypt the Digital License Data, which could then be edited, reencrypted and used to change the Digital Driver's License details on the mobile device. In other words, no sign of any authentication, no digital signature or anything else to protect against user tampering and manipulation of the stored and then displayed data. The only protection was this four-digit PIN. And when you use it, it decrypts the data, bringing it back into plaintext, which can be modified. Unbelievable. Again, this is 2022.

Next problem is a lack of any client-side validation. As they said, the Digital Driver's License data is never validated against the back-end authority which issued the license. So not only is there no local authentication, it's not signed, nothing. But there's no ongoing period or ever verification with the original issuer. They wrote that this means that the application has no native method to validate the Digital Driver's License data that exists on the phone and thus cannot perform further actions such as warn users, or anyone relying on this data, anybody else, when this data has been modified.

Since the Digital Driver's License data is stored on the client's device, validation should take place to ensure the local copy of the data matches the Digital Driver's License data that was originally downloaded from the ServiceNSW API. Or, I would add, locally verify a signature. No such verification takes place, An attacker's able to display the edited data on the DDL app without any prevention.

Okay. Now presumably, whatever moron designed this system figured that since it was encrypted with military-grade 256-bit AES encryption, and thus could never be modified, there would be no need to verify it. And one of the features they boasted was that it could run completely offline. Right. No verification needed. And speaking of verification, one of the key "verification features" of the digital license is its so-called "pull-to-refresh" functionality, which is used to ensure that anyone relying on it is viewing the most current license information.

However, the Dvuln guys noticed that refreshing the application's driver's license data only updates the QR code which is displayed on the license, and that the QR code only contains the license holder's name and whether they are under age of 18 or not. That's the only thing in the QR code. So if a fraudster had modified their license details and photo by decrypting it, modifying, and then reencrypting the data, this fraudulent data would remain visible on the screen even after the QR code, date, and time had been refreshed and updated.

And not surprisingly, in still another example of incredible sloppiness, the license data is, indeed, exported in iPhone backups, making its modification outside of the phone trivial. As we know, when a secured mobile device is jailbroken, it's reasonable to assume that any security features an application may have could be bypassed because an attacker has obtained root-level access to the device's storage and various services. But conversely, as long as a secured mobile device is not jailbroken, apps should be able to be reasonably secure that their users are protected against misuse and various types of client-side vulnerabilities. However, in the case of ServiceNSW's application, the Digital Driver's License data is included in device backups, which means that attackers or anyone wanting to commit fraud can obtain and modify their license details without ever needing to jailbreak their device.

There is no way that this system was ever reviewed by any competent security expert. These days we've got competent security experts coming out of our ears. Anybody who had any training in digital application security would immediately see what an embarrassment this thing is. And the publicly known problems with it are now three years old, with rumors that these Digital Driver's Licenses are being readily spoofed. Of course they are. So we have to ask ourselves, how did this happen? Was it created by some government bureaucrat's unemployed nephew? And why doesn't anyone appear to care? It seems quite likely that someone will be caring very soon, thanks to the bright light that the Dvuln guys have finally aimed at this mess because the tech press has picked this up and run with it. So this is going to be an embarrassment to New South Wales and whatever clown wrote this thing.

Okay. So because this is the Security Now! podcast, we can ask, so what's the answer? How do we solve this problem? Since I've been called a competent security expert, I'll take a stab at it. How about this? All that's needed is a certificate, a standard X.509 format certificate. This has all been worked out already. Certificate fields can contain binary data. They already do, like they have a public key in them. So the certificate owner's photo can easily be contained within the certificate, as can dates of starting and ending validity and anything else that might be needed, like a timestamp for when the certificate is signed, the owner's legal name, their physical address, their date of birth, and so on.

What makes the certificate special is that its entire contents is signed by a recognized and trusted authority. As we know, the process of signing is that the certificate's contents are hashed to create a digest of that content. Then that hash is encrypted with the signer's private key. Anyone wishing to later verify the certificate's authenticity, meaning the contents of everything in the certificate, simply creates their own hash of the certificate's contents, then uses the signer's public key to decrypt the hash that came with the certificate. If the new hash and the decrypted hash match, then we know that not one single bit of the certificate's contents have been modified since it was signed.

So to build a simple and practical system, an existing trusted root authority, a certificate authority, any existing certificate authority that's already trusted by the mobile iOS and Android platforms, take my favorite one and chosen certificate authority DigiCert, issues an intermediate certificate to the New South Wales administration, which is itself permitted to sign these special purpose end DDL (Digital Driver's License) certificates.

Anytime someone out in the world, like a DDL license holder's DDL application, wishes to update and reverify their license, that app sends a request to the government's server. The server pulls together all of the relevant data it has for the individual from its database, including their latest photo, their date of birth, the driver's license initial and expiration date, and builds a new certificate which also contains a current timestamp. They use the private key that was obtained from DigiCert in this example, to sign the resulting certificate. They then bundle that certificate with their intermediate certificate's public key and send the package back to the DDL application.

Since the bundle contains a short certificate chain whose intermediate certificate is already trusted by the root certificate store in the mobile device, and since that intermediate certificate can verify the DDL end certificate, any iOS or Android device has everything it needs to verify the authenticity of the DDL end certificate. And there is no way for that DDL license certificate to be modified or tampered with in the field without breaking its signature. In this system, there's no need for a PIN to decrypt the certificate's data because there's no need for any certificate encryption in the first place at all.

And as for that QR code that apparently only contains the user's name and age, that's also, frankly, ridiculous if its data can be spoofed. If you want to have some sort of verification reader that reads the QR code, then the QR code can simply contain the individual's driver's license number. And we know that that cannot be spoofed. And if it were, it contains the driver's license number. So the reader, the QR code reader obtains the license number that is being claimed on the screen, makes a query to the government's server to obtain their signed DDL certificate, and displays the user's name, age, photo, and everything else so you can make queries on the fly.

And if the QR code system needs to work offline, which was apparently a feature of the current system, then the QR code can simply contain a signed subset of the user's information, like only their name and date of birth, which is separately signed by the government's intermediate certificate. The QR code reader then scans the QR code, which is itself a tiny certificate. It uses the government's intermediate certificate to verify the QR code certificate and can then trust that the data contained in the QR code certificate subset has not been tampered with.

Again, none of this is rocket science. As I've said many times, we now have an incredible toolkit of technology components that can be applied individually and collectively to solve any of these sorts of problems. Nothing needs to be invented anymore. And the beauty of this system is that all of the well-tested and bulletproof crypto libraries already exist. In fact, the iOS and Android platforms already contain down in their kernels all of the required crypto machinery APIs. None of that needs to be created. So hopefully, when this ridiculous New South Wales Digital Driver's License disaster fiasco finally comes to light, the existing ridiculous system will not be salvaged. It is unsalvageable. It needs to be entirely scrapped and replaced with a simple bulletproof system like the one I just described. There's nothing to it.

Leo: Seems simple.

Steve: Yes, Leo. Unbelievable that, like, this was designed three years ago. This all existed three years ago. This is not new. This is just the obvious way to solve this problem.

Leo: Did I hear you correctly that the data is unencrypted with a four-digit PIN?

Steve: Yes.

Leo: You enter your PIN, and now I can modify my driver's license.

Steve: Yes.

Leo: Well, that's, I mean, anybody looking at that would see the problem.

Steve: I know. It's insane.

Leo: It just assumes that, well, why would anyone want to modify their driver's license?

Steve: Yeah, I can't imagine why.

Leo: Why?

Steve: An 18-year-old would want to spoof their age.

Leo: Why would they modify their - yeah.

Steve: No one's ever...

Leo: No one's ever done that before.

Steve: No.

Leo: That's just bizarre. Oh, well.

Steve: So we have the latest Microsoft Office zero-day remote code execution vulnerability.

Leo: Of the week.

Steve: Oh, my god. This is a head-buried-in-the-sand quite pervasive problem in Microsoft Office. By far the best researcher on this has been Kevin Beaumont. We've talked about Kevin from time to time. He's a great security researcher. He tweets using the handle GossiTheDog for some reason. And he even, Kevin, tracked down a bachelor's thesis authored by a guy named Benjamin Altpeter on August 1st, 2020, so nearly two years ago. On page 29 of his thesis, of his bachelor's thesis, Benjamin writes: "Windows includes the ms-msdt://" - so it's a URL scheme - "protocol that opens the Microsoft

Support Diagnostic Tool which provides the troubleshooting wizard to diagnose WiFi and audio problems and the like. This protocol directly passes the string it is given to the msdt.exe program. The attacker now needs to find an included wizard that allows the execution of arbitrary programs, preferably even remote ones. The program compatibility wizard fits this description. Luckily for the attacker, all user input can also be prefilled from the command line."

Okay, now, this MSDT thing was new to me. At that point I opened my own command-line window on my Windows machine, entered "msdt" and hit ENTER. And sure enough, up popped a "Microsoft Support Diagnostic Tool" dialog which I had never encountered before. So this MSDT thing is alive and well and living in all of our Windows machines right now. It turns out you can access this, which is to say bad guys can access this, through an Office document using the ms-msdt:// protocol and use it to remotely execute code.

Okay, so back to Kevin. Kevin's name for this exploit has stuck. It's now semi-officially known as the "Follina" (F-O-L-L-I-N-A), the Follina exploit, because Kevin spotted a reference to 0438 in the sample exploit file, and 0438 is the area code of the city of Follina, Italy.

Leo: Okay. Hey, as good a reason as any.

Steve: That's the way we name these things these days, folks.

Leo: Yes, it is. Hysterical.

Steve: So Benjamin's bachelor's thesis was nearly two years ago. The reference in it was obscure, it was on page 29, and we'll never know whether someone saw it and recognized its significance, as he did, or may have independently invented an attack. Given the nearly two-year interval, I'm inclined to think that this was an independent discovery because these sorts of things happen all the time.

But either way, last month on April 12th, the leader of Shadowchasing1, an advanced persistent threat (APT) hunting group, reported the active exploitation of this vulnerability in the wild to Microsoft's MSRC, their Microsoft Security Response Center. That's April 12th. The Shadowchasing1 report provided a copy of the in-the-wild, real-world Microsoft Office document exploit which was targeting Russia, themed as a Russian job interview. Nine days later. Nine days go by. On April 21st, Microsoft's MSRC blew it off and closed the ticket, saying that it was not a security-related issue.

I have a picture in the show notes, a screenshot of this. It reads - it's signed by MSRC, so it's written in the first person. It reads: "I finally had time" - after nine days. "I finally had time to look at this critically and have decided it is not a security-related issue. Msdt is indeed executed, but it requires a passcode when it starts, and the one provided in this sample does not work for me. I will be closing this case, but appreciate you submitting it. Regards, MSRC."

Okay. Now, I'm not sure what the phrase "looking at this critically" means, exactly. But apparently it doesn't mean "closely examined and worked to understand what the underlying problem might be." And, you know, this might represent another sample of the pattern that seems to be emerging, where Microsoft increasingly seems to be needing the external security research community to solve all of its problems for it, and hand it everything on a silver platter. Apparently saying "Hey, have you considered that

allowing Word's remote template feature to retrieve an HTML file from a remote server, which in turn uses the ms-msdt:// URI scheme to load some code and execute some PowerShell, even when Office Macros have been disabled and even able to bypass Protected View by giving the document an RTF extension?" is no longer sufficient to get Microsoft's attention.

Now, in fairness, the provided exploit sample apparently did not work instantly for the overworked MSRC guy. And we don't know how many false-negative reports the MSRC guys might be fielding every day. Perhaps they face a constant deluge of bogus reports. But we do know about economics, and we do know about incentives. So we know that if this sort of behavior is never met with consequences, it will continue and even be de facto encouraged. And Microsoft has arranged to completely insulate themselves from any consequences ever. So all we can ever do is hope for the best.

Since we're apparently on our own, what do we do? Kevin reports that the vulnerability has been proven to work against Office 2013, 2016, 2019, 2021, Office ProPlus, and Office 365. In other words, all of them. It also applies to Windows itself because this thing, this exploit can be invoked from, believe it or not, our old friend the .lnk link file. Yup. That hasn't gone away. So there are two different issues: Office itself using the ms-msdt: scheme protocol while allowing loading unfiltered HTML Word templates and Outlook links; and the fact that the MSDT executable allows code execution. All flavors of Windows Defender also completely miss detecting this, except yesterday that changed. Which is not surprising given that Microsoft decided last month that this was not a security issue. So that was true until yesterday. Now Defender's awareness has been updated, and I'm sure other AV companies are on this also.

And Microsoft released a workaround in the meantime in the form of a registry script, also yesterday. That registry script simply deletes the ms-msdt protocol handler reference from the registry. I have it. You run the command prompt as admin, and then you do a "reg delete HKEY_CLASSES_ROOT\ms-msdt /f." But get this from the show notes if you want to try it. And as always, it's good to back up your registry beforehand.

So Kevin wonders out loud how this might evolve. He says: "We'll see. Microsoft are going to need to patch it across all the different product offerings, and security vendors will need robust detection and blocking. Microsoft will probably point towards Protected View; however, Protected View also applies by default to all macros, and Office macro malware is most definitely a major problem regardless."

Then in an update to Kevin's original posting, he added: "Microsoft have indeed pointed to Protected View, saying it 'prevents' the attack." Kevin writes: "I think this is stretching the truth. For example, if the document is a .RTF file and is opened by Preview in Explorer, Protected View does not apply, and it becomes a zero click exploit. Microsoft knows this. They just aren't mentioning it to customers." And then Kevin provides a tweet from our well-known CERT/CC guy, Will Dormann, and Will agrees. Will tweets: "This language is a bit misleading in not really describing what 'calling application' means. If you preview a file in Explorer, which uses Office to render the document, Protected View doesn't do a damn thing," says Will.

The very latest is that Microsoft, as I said, has finally awoken to this threat, a CVE has been assigned, and Windows Defender's detection signatures have been updated. But the underlying trouble is that the use of MS Office protocol is extensive and pervasive, so it cannot be shutdown without breaking all sorts of other things that depend upon it. It's a bit like last year's printer spooling Catch-22 fiasco. It's another of those problems that isn't really a bug that can be patched because it's the abuse of a deliberately designed-in feature. And this is what results from systems that become too complex. It begins to be impossible to understand and anticipate every possible interaction among components when there are just so many different components. So it's the world we are in today.

Anyway, Defender's updated. Other AVs are being, if they're not already, updated. We've got now the maximum time for Microsoft to create a patch because today is the last - today this Tuesday is the last day of the month, which puts the first on a Wednesday, meaning that Patch Tuesday will be the 14th, the latest date it could possibly be in the month. So Microsoft has a full two weeks. Maybe they'll be able to get a patch out for Office across the board by then. We'll see.

Okay. Ghost Touch. Get a load of this one. This is not yet a real-world threat due to the potential attack's very short range which is currently around 40mm, or just over an inch and a half. But as we know, "impractical" is how many eventually practical attacks began. Here's the abstract of the 17-page exploit research paper describing this new and very clever Ghost Touch attack which was invented by a team of Chinese and German security researchers.

They explain: "Capacitive touchscreens have become the primary human-machine interface for personal devices such as smartphones and tablets. In this paper we present Ghost Touch, the first active contactless attack against capacitive touchscreens. Ghost Touch uses electromagnetic interference (EMI) to inject fake touch points into a touchscreen without the need to physically touch it. By tuning the parameters of the electromagnetic signal and adjusting the antenna, we can inject two types of basic touch events, taps and swipes, into targeted locations of the touchscreen and control them to manipulate the underlying device.

"We successfully launch the Ghost Touch apps on nine smartphone models. We can inject targeted taps continuously with a standard deviation of as low as 14.6 by 19.2 pixels from the target area, a delay of less than half a second, and a distance of up to 40mm. We show the real-world impact of the Ghost Touch attacks in a few proof-of-concept scenarios, including answering an eavesdropping phone call, pressing the button, swiping up to unlock, and entering a password. Finally, we discuss potential hardware and software countermeasures to mitigate the attack."

Okay. So a little bit of background. A capacitive touchscreen it turns out is a surprisingly sophisticated and highly sensitive device, which we now pretty much take for granted. And the technology is known to be quite susceptible to local environmental noise. It's easy to make a touchscreen malfunction when it's placed near a source of electromagnetic interference. There are switching regulators used in some phone chargers which are known to generate so much short-range high-frequency electromagnetic interference that nearby touchscreens will not function if they are nearby.

So what these clever researchers have done is turn electromagnetic interference into electromagnetic touch signals to deliberately spoof what are essentially the receivers in capacitive touch systems to generate spoofed touch events. And they made it work. At this point it's only of academic interest. But we should not be surprised to learn of it being applied in some way in the future.

Okay. There's something new that might be coming known as TrustPiD. If it ever actually happens and Vodafone and Deutsche Telekom are reportedly both now in pilot testing of this new TrustPiD system in Germany the technology creates, deliberately, I mean, it's designed to create a persistent, static, super-cookie which, being somehow injected into a user's communications at the carrier-level by the cellular carrier, cannot be seen, managed, or blocked by end users. The only reason I can see for any cellular carrier to be doing something this clearly privacy invasive is that they've decided that it's more important for them to get in on the Internet advertising revenue boom by arranging to monetize their customer's "anonymous," and that's in air quotes, online identities.

Here's what the TrustPiD website says under the tired banner of "Keeping the Internet Free." Right? So that's what they're - this is how we're going to keep the Internet free, kids. And it anyone's interested, TrustPiD, I think it was dot com. It reads: "Consumers appreciate the idea of a free Internet, but this comes with a trade-off. Publishers need a sustainable revenue model, meaning that it becomes essential to add subscription paywalls or rely on advertising to maintain free access to high-quality content. With a growing trend of digital information shared in the ecosystem, consumers' concerns about privacy and the amount of information passed into the free Internet have been raised." No kidding. By things like this, apparently.

"TrustPiD," they wrote, "is a technology solution that enables consumers to enjoy free content and the benefits of the open Internet whilst retaining control over their privacy." Uh-huh. "TrustPiD is a secure" - yeah, like that Digital Driver's License - "unique digital token generated by assigning random numbers to you, which reduces the risk of you being directly identified whilst" - they like that word - "still enabling advertisers and publishers to provide you with a personalized experience across their sites with your consent. You can find more information on how we generate and manage your TrustPiD in the Privacy Notice." I went there. There was none.

Leo: That's funny. Good luck finding it.

Steve: That's right. "Your consent for TrustPiD is collected by advertisers and publishers via their Consent Management Platform" - and then they have here in parens (cookie banner) - "when you visit their sites. The consent will apply only to those websites. TrustPiD service gives consumers complete control over how their TrustPiD is used by enabling them to manage their consent via a central 'Privacy Portal' at any time. In the Privacy Portal, consumers are able to track which advertisers or publishers they have allowed to provide them with personalized online marketing in their websites based on their TrustPiD in one central place, the possibility to withdraw consent at single website level, and the ability to turn off the TrustPiD service entirely, preventing any further use of the token. If you want to understand more about how the Privacy Portal works, or if you want to manage your consent, please visit the dedicated Privacy Portal page." And I should mention that only customers of Vodafone and Deutsche Telekom who are involved in this have access to the portal. You've got to validate by using your cellular device.

Okay. So I have many questions. Aside from the fact that this will assuredly be an opt-out service where all users will be, by default, opted-in - otherwise it would never get off the ground; right? Apple and their attempt to require advertisers to get explicit permission showed that people just say no, thank you, I don't want that. Okay. And thus enabling tracking without their consent. And we need to know whether this system will respect the user's GPC, the Global Privacy Control setting. Presumably the websites at the other end have to. So we'll see how that goes.

As I said, though, aside from that, how exactly can identifying tags be injected at the carrier level into encrypted HTTPS web sessions? This is really a concern. The only way I can see that this can be done in 2022 is by having the carrier, in this case Vodafone and Deutsche Telekom, actively intercepting and proxying all of their customers' encrypted TLS connections. We saw this years before when some carriers were offering caching and data compression services. Remember that? Cellular carriers would offer, like we're going to speed up your Internet connection, you lowly cellular users, by compressing the data for you, sending it to you compressed, and then we'll decompress it on your phone. But that was back in the pre-HTTPS Everywhere days, where most connections were still unencrypted.

But in 2022, as we know, nothing is unencrypted anymore. A carrier can see the IP addresses that their users are connecting to, but IP addresses no longer reflect the website properties behind them, since SNI (Server Name Indication) is now being heavily used during TLS handshaking to identify the domain being connected to and thus the TLS certificate which should be sent to the client. This is allowing many websites to reside at the same IP. So there's no way to disambiguate them from the outside. So this means that a persistent identifier is either somehow injected into the user's connections by their smartphone's browser before entering their TLS encrypted connection, or the user's smartphone must accept a Vodafone certificate authority-style root certificate into its root store to allow Vodafone and Deutsche Telekom to function as a sanctioned, active, persistent cookie-injecting man in the middle.

Neither of these seem likely, or possible, or even remotely feasible in 2022. So I'm stumped. I mean, I really am. Without being on the inside, Vodafone and Deutsche Telekom cannot see who their customers are connecting to and cannot alter the content of their data injecting something into the flow. And cellular carriers are explicitly and deliberately on the outside. They're carriers of encrypted content.

I followed every link I could find, looking for any technical documentation. There's none that I could see. Since I don't read German, my digging was somewhat limited. Maybe I'm missing something obvious. But even if there's some way to pull this off technically, why the hell are our carriers getting in on the "identifying their customers to advertisers" game? My feeling is this is a monster that needs to be strangled in its crib immediately, before it has a chance to grow. My mind is boggled by the idea that Vodafone and Deutsche could be experimenting with this, with something like this. And clearly it's to sell the information; right? I mean, why else? Wow.

Okay. A couple of closing-the-loop bits. MikeO, he said: "Hi, Steve (@SGgrc) and Leo (@leolaporte)." He said, "Listed" - and he meant listened, I'm sure -"to SN-872" - so that was last week - "and you said there were no good ad blockers for macOS/iOS." He said: "I've been using 1Blocker for years, and it works great with Safari on all of Apple's platforms: 1Blocker.com."

And actually I was reminded of that. Bill Zegarski, he also tweeted, saying: "@SGgrc Heard the conversation on ad blockers for Safari. @1BlockerApp, which you recommended and I love," he says, "on iOS, also has a Mac version. Might want to look at that. Best part is that since it's a universal app, if you purchase on one platform, you get both."

Leo: I guess it depends on what you call "good."

Steve: Okay. Fair enough.

Leo: If by "good" you mean blocks all ads, it's not.

Steve: Then no.

Leo: So the problem is I don't think it's 1Blocker's. I also use Firefox Focus. I use a variety of different, or have attempted to use a variety of different adblockers on iOS. And the problem is because of the architecture, and this is what we were talking about, it's very hard for an adblocker to work as well as UBlock Origin, Gorhill's. So I'm looking at iMore.com, great site, using 1Blocker, all of the blocking turned on,

and then using Gorhill's UBlock Origin on my desktop. And on the 1Blocker on iOS I see an ad for Mint, Best Buy, Kay Jewelers - this is with the blocker turned on. And on and on and on. There's quite a few ads. And I don't think that that's 1Blocker's fault. I just think that it's hard to do with the stricture that Apple places on it.

Steve: Yup, good.

Leo: So that's the problem.

Steve: I totally agree.

Leo: That's what we were talking about. There isn't any full adblocker available on iOS.

Steve: Right. And as I have said, because I am such a fan of UBlock Origin, when I go to a machine that doesn't have it, I'm like, oh, my lord.

Leo: You forget.

Steve: This is what people put up with on the Internet?

Leo: Yeah.

Steve: Oh, my. I'd pay not to have that.

Leo: So if you want a comparison, put 1Blocker on your Mac or your iOS, and then look at it with UBlock Origin on Firefox on your Mac. I mean, I don't see any of those ads with UBlock Origin. Any of them.

Steve: Perfect, perfect.

Leo: So again, I don't blame these blockers, although this is not a free blocker. So you're paying for something that ostensibly blocks ads, but doesn't.

Steve: It's not.

Leo: I mean, Firefox Focus, which is a free ad blocker from the Firefox folks, does, as well. I think it's more the Safari issue as really anything else.

Steve: George Palfi said: "Just heard your discussion on quantum computing and cryptography. Perhaps we can adapt new crypto for quantum computers. But Bitcoin, Ethereum, and many cryptocurrencies are based on very old crypto rules," he says, "and

I don't believe the algorithms can be changed. So people who want to keep their wealth protected for a long time will be at risk as quantum computing evolves. Better to stick to gold and silver which have worked for thousands of years," says old George.

And, okay, the only issue I would take with that, first of all, there are several aspects of using cryptocurrencies. One is passwords, and I would never disagree that because passwords may be using public key crypto, they're going to be in trouble. But, for example, we know Bitcoin well because we talked about, it was 11 years ago or something, when it was worth dust, and there was a bitcoin faucet where you could just get free pieces of a bitcoin by going there. And so we know that it uses a 256-bit hash. And hashes are strong even against quantum crypto. It's specifically the public key aspect of today's crypto that has people worried, not the symmetric crypto or the hashes. And that's the essence of, you know, hashing is the essence of these various cryptocurrencies. They're going to be safe. Their design probably does not need to be changed. But George makes the point, which is valid, which is they can't be changed. They are an embedded crypto technology that is here forever. Okay. I guess you could just stop mining. If you stopped all mining, then - I don't know. I have to think about that.

Anyway, r4nd0m is his handle, with A as a 4 and O as a numeric 0. He said: "Hey @SGgrc. I was thinking about the layer one attack against the unlock function of the various apps using Bluetooth Low Energy, and it occurred to me that a prompt in the app at unlock time should be a feature they add to help mitigate the demonstrated weakness. Of course the user experience will suffer, but this should defeat the attack." And I just wanted to share that because he's absolutely right. So what he's saying is, if you had to touch the lock, and then the use of the app was not autonomous, but you had to then take your phone out of your pocket and tap to acknowledge, that solves this problem; right? Because it wouldn't be done autonomously.

Unfortunately, on the site that I visited when I was putting the story together last week, the home page shows this lady with her arms full of groceries, like the whole point being she can't do anything except manage to, like, hit the lock with her elbow in order to register her presence, and then, oh, look, the door unlocks, and she's able to get in. So exactly as he says, the user experience, which is so magical, by just touching the lock and you're able to enter, well, that's a lot less magical if you have to acknowledge it on your phone. But it would solve the problem.

And Bryant McDiarmid says: "Hey, Steve. Quick question. If I encrypt a file with a 256-bit encryption three times, with three different passwords, what is the resulting bit strength? Is it 256 plus 256 plus 256? Or 256 times 256 times 256?" And Bryant, the answer is plus. It's the equivalent on the symmetric key side of 768-bit encryption. But the real strength is the entropy in the keys which are turned into the 256-bit encryption keys. That's what really limits you. But to answer your question, you are adding the equivalent bit lengths rather than multiplying them.

Leo: And of course adding is virtually no value compared to multiplying; right?

Steve: Right.

Leo: Yeah. Shall I do an ad for you, sir?

Steve: Please.

Leo: Thank you.

Steve: I'm going to wet my whistle, and then we're going to talk about whether DuckDuckGo is gone.

Leo: I just did an informal test using the Firefox Focus plug-in. And actually that does seem to block all the ads. So there is a way with Apple's strictures on iOS to still use Safari.

Steve: So maybe 1Blocker...

Leo: So maybe not use 1Blocker, which is not free. Use Firefox Focus, which is.

Steve: And maybe they've sold out.

Leo: Maybe. Or, you know, the thing, it's a moving target. One of the reasons I stopped using 1Blocker is there's literally 10 different blockers you have to turn on, and it's complicated. Firefox Focus is a browser, but it also works as an extension so you could turn it on as a Safari extension, which is what I do, yeah.

Steve: Nice.

Leo: Our show today, my friends, more importantly...

Steve: But wait. And that's what he wanted. He wanted a good Safari extension for his Mac.

Leo: Yes. That's the issue on the Mac because everything is using WebKit anyway.

Steve: But you've solved the problem then.

Leo: Focus, I think. That's what I've been using, and I didn't realize how well it did. See, a problem is I have NextDNS, so I have to turn that off. I have to go through a lot of hoops to see if I'm going to really get ads or not.

Steve: Leo, it's like me trying to get a banking app to run on my phone.

Leo: Yeah.

Steve: To do it I need a week in order to shut down all the security.

Leo: Really? You want me to have a banking app on here? Really? Actually, you know, if you think about it, what she was really asking for is biometric. Right?

Steve: Well, I unlocked my phone, so she could see the phone.

Leo: Oh, yeah. That should be sufficient. Well, she wants you to unlock the phone and show it's your phone. I mean, in a way that isn't a bad idea for additional - it's a way of kind of ad hoc biometric security. Can you unlock your phone and prove that it's yours.

Steve: I think I did that. I do have my credit cards registered with the phone. And so, you know, it'll be...

Leo: Well, that should be sufficient. Yeah, having Apple Pay on there, that's better than banking, yeah.

Steve: Yeah, got Apple Pay.

Leo: Well, anyway, she let you have the car. That's the important part.

Steve: So the winner, as I said at the top of the show, of this week's most tweeted news of concern was the widely reported surprise that the explicitly privacy-centric and privacy-protecting DuckDuckGo enterprise had struck a previously secret backroom deal with Microsoft to enable user tracking from Microsoft-owned domains, including Bing.com and LinkedIn.com, when using the so-called DuckDuckGo Privacy Browser. It was further revealed by DuckDuckGo's founder Gabriel Weinberg that DuckDuckGo's non-disclosure agreement with Microsoft prevented them from any further disclosure of their agreement's terms. And I have the show notes, and it's on the screen, two of Gabriel's tweets.

He said: "For non-search tracker blocking (e.g., in our browser) we block most third-party trackers. Unfortunately our Microsoft search syndication agreement prevents us from doing more to Microsoft-owned properties. However, we have been continually pushing and expect to be doing more soon."

This his follow-up tweet was: "We've been working tirelessly behind the scenes to change these requirements, though our syndication agreement also has a confidentiality provision that prevents disclosing details. Again, we expect to have an update soon that will include more third-party Microsoft protection."

Okay. So let's back up a bit and see what happened. TechCrunch's headline: "DDG has a tracker blocking carve-out linked to Microsoft contract." BleepingComputer reported: "DuckDuckGo browser allows Microsoft trackers due to search agreement." 9to5Mac headlined: "DuckDuckGo caught giving Microsoft permission for trackers despite strong privacy reputation." An Android Police headline: "DuckDuckGo's supposedly private browser caught permitting ad tracking."

Okay. The first thing for us to clear up about all this is that all refers to the use of DuckDuckGo's own browser, not to the use of the DuckDuckGo search engine. On the other hand, anyone could be forgiven for missing this distinction. When you go to

DuckDuckGo's homepage you're greeted with a bold headline: "Tired of being tracked online? We can help." Then below that are three big topics: "Privacy for Chrome," "Private Search Engine," and "Privacy Browser App." And explaining their privacy browser app they say: "Our private browser for mobile comes equipped with our search engine, tracker blocker, encryption enforcer, and more. Available on iOS and Android." And as we've previously mentioned, actually there is one coming for macOS, and apparently one for Windows is also planned.

So all of this brouhaha began when security researcher Zach Edwards took the time to audit the data flows to and from one of DuckDuckGo's mobile browser platforms. And given a screenshot we'll have later, it looks like he was using the Android browser. Early last week he tweeted: "Sometimes you find something so disturbing during an audit, you've got to check/recheck because you assume that something must be broken in the test. But I'm confident now. The new @DuckDuckGo browsers for iOS and Android don't block Microsoft data flows for LinkedIn and Bing."

So this is mostly, I think, a story about expectations. As Leo, you and I were saying up at the top. Zach was so surprised and disturbed by what he saw specifically because it wasn't what he ever expected to find from DuckDuckGo. His next tweet he said: "DuckDuckGo has browser extensions and their own browsers for iOS/Android at DuckDuckGo.com/app." And then he's got links in his tweet for the iOS version and the Android version. He said: "Both versions of the DDG browser claim to use tools which automatically block hidden third-party trackers." Then he's got two question marks. And to back that up, Zach attached a screenshot of the clear statements from DuckDuckGo. And I'll just read them quickly.

It says: "Tired of being tracked online? We can help. DuckDuckGo is the all-in-one privacy app." Now, the app meaning exactly what we're talking about, the browser app. "DuckDuckGo is the all-in-one privacy app that helps protect your online activities. With one download, you get a new everyday browser that offers seamless protections from third-party trackers while you search and browse, and even access to tracking protections when receiving email and using other apps on your device. With DuckDuckGo, privacy can be your default."

Then it's got five callouts: Search Privacy, Escape Website Tracking, Enforce Encryption, Block Email Trackers, and Protect Your Privacy in Other Apps. I'll just expand on that second one, Escape Website Tracking, where it explains: "Tracker Radar automatically blocks hidden third-party trackers we can find lurking on websites you visit in DuckDuckGo, which stops the companies behind those trackers from collecting and selling your data." It doesn't say "Except for Microsoft's advertising trackers, which our contract with them prevents us from blocking or disclosing." And that's the problem because that's the truth.

Zach's next tweet was: "I tested the DuckDuckGo so-called private browser for both iOS and Android, yet neither version blocked data transfers to Microsoft's LinkedIn and Bing ads while viewing Facebook's Workplace.com homepage." He says: "Look at DDG bragging about stopping Facebook on Workplace. No mention of Microsoft."

And in the show notes, you've got it onscreen, Zach posted a photo showing his Android phone at Facebook's Workplace.com site with a DuckDuckGo popup, sure enough, bragging about the wonderful job it's doing, saying: "Google, Facebook were trying to track you here. I blocked them! You can check the URL bar to see who is trying to track you when you visit a new site." And then there's a big High-Five button which the happy user can presumably press.

Leo: Has to press.

Steve: Oh, that's how you get rid of it.

Leo: I don't want to do a high-five.

Steve: Ah.

Leo: No.

Steve: So the fact that this browser specifically singles out other tracking properties while silently permitting Microsoft-owned domains to track feels at best disingenuous. In an attempt to clarify the mess that arose from this, Gabriel appeared to attempt to explain what's going on over on Reddit. You'll hear for yourself in a moment why I'm wording this as provisionally as I am because, although I've read what Gabriel wrote several times slowly and carefully, and it sort of sounds like he's explaining something, I wrote I still have no idea what he said, although I think it was on the fourth reading that it finally sank in, and then I explained it. So we assume that he knows what he's trying to say, but he sure doesn't communicate it very clearly.

Here's what Gabriel wrote in Reddit, in this posting on Reddit. He says: "Hi. I'm the CEO and Founder of DuckDuckGo. To be clear, since I already see confusion in the comments" - and, boy, confusion in what he's going to write here. He says: "When you load our search results, you are anonymous, including ads. Also on third-party websites we actually do block Microsoft third-party cookies in our browsers, plus more protections including fingerprinting protection. That is, this article is not about our search engine, but about our browsers. We have browsers" - and he says, parens, "(really all-in-one privacy apps) for iOS, Android, and now Mac in beta."

He says: "When most browsers on the market talk about tracking protection, they're usually referring to third-party cookie protection and fingerprinting protection, and our browsers impose these same restrictions on all third-party tracking scripts, including those from Microsoft. We also have a lot of other above-and-beyond web protections that also apply to Microsoft scripts and everyone else, for example, Global Privacy Control, first-party cookie expiration, referer header trimming, new cookie consent handling in our Mac beta, fire button one-click data clearing, and more.

"What this article is talking about specifically" - he keeps telling us, but he never really gets to it - "is another above-and-beyond protection that most browsers don't even attempt to do for web protection, stopping third-party tracking scripts from even loading on third-party websites, because this can easily cause websites to break. But we've taken on that challenge because it makes for better privacy and faster downloads. We wrote a blog post about it. Because we're doing this above-and-beyond protection where we can, and offer many other unique protections - for example, Google AMP/FLEDGE/Topics protection, automatic HTTPS upgrading, tracking protection for other apps in Android, email protection to block trackers for emails sent to your regular inbox, et cetera - users get way more privacy protection with our app than they would using other browsers."

Okay. So he's sort of deflected a little bit here. He's like, oh, look over here at all these other things you get. He says: "Our goal has always been to provide the most privacy we can in one download." So he still hasn't told us, in the third paragraph yet, what it is. Now the fourth paragraph.

"The issue at hand is, while most of our protections like third-party cookie blocking apply to Microsoft scripts on third-party sites," he says, "again, this is off of DuckDuckGo.com, i.e., not related to search," okay, whatever that meant, he says, "we are currently contractually restricted by Microsoft from completely stopping them from loading," and he says "(the one above-and-beyond protection explained in the last paragraph)," but he didn't really explain it, "on third-party sites. We still restrict them, though, for example, no third-party cookies allowed. The original example was Workplace.com loading a LinkedIn.com script. Nevertheless, we have been and are working with Microsoft as we speak to reduce or remove this limited restriction." Which maybe he just explained? Except not really.

He says: "I understand this is all rather confusing" - uh-huh, still - "because it is a search syndication contract that is preventing us from doing a non-search thing. That's because our product is a bundle of multiple privacy protections" - okay - "and this is a distribution requirement imposed on us as part of the search syndication agreement that helps us privately use some Bing results to provide you with better private search results overall." He says: "While a lot of what you see on our results page privately incorporates content from other sources, including our own indexes - for example, Wikipedia, local listings, sports, et cetera - we source most of our traditional links and images privately from Bing," he says, "though because of other search technology our link and image results still may look different."

He says: "Really only two companies, Google and Microsoft, have a high-quality global web link index," he says, "(because I believe it costs upwards of a billion dollars a year to do so). And so literally every other global search engine needs to bootstrap with one or both of them to provide a mainstream search product. The same is true for maps, by the way. Only the biggest companies can similarly afford to put satellites up and send ground cars to take street view pictures of every neighborhood.

"Anyway," he says, "I hope this provides some helpful context." Uh-huh. "Taking a step back, I know our product is not perfect and will never be. Nothing can provide 100% protection. And we face many constraints: platform constraints (we can't offer all protections on every platform due to limited APIs or other restrictions); limited contractual constraints like this one; breakage constraints (blocking some things totally breaks web experiences); and of course the evolving tracking arms race that we constantly work to keep ahead of. That's why we've always been extremely careful to never promise anonymity when browsing outside our search engine because that frankly isn't possible. We're also working on updates to our app store descriptions to make this more clear. Holistically, though, I believe what we offer is the best thing out there for mainstream users who want simple privacy protection without breaking things, and that is our product vision."

Okay. So here's what I think Gabriel said. He said, and I'm paraphrasing, this is written in his voice, so imagine that this is what he said. We want to provide a robustly privacy-enforcing search engine service. But to do that we first need to have a search engine service, and no one other than Google and Microsoft can do that on their own. So if we want to provide search, we need to purchase access to a big search index. And we chose Microsoft's. Because we're able to sanitize and purify the link results we provide, we're able to offer tracking-free search, and we do.

Next paragraph, making this up: But completely aside from and separate from web search, we also wanted to provide a privacy-enhancing web browser. We wanted to do this because offering clean search results is only part of the problem. A privacy-centric web browser could do so much more. And one of the "so much more" things a browser can do is not only block cookies being set and read by third-parties, but also block their third-party JavaScript code from ever being run in the user's browser. And we're able to

do that, to whatever degree we can, for everyone other than when a Microsoft property is that third party.

In that case we must allow their JavaScript to run. This is not because we want to make an exception for Microsoft. It's because the completely unrelated agreement we have, which allows us to have access to their Bing search index, explicitly requires that we not block the execution of their scripts in our browser. Because this has all come to light now, we're planning to amend our DuckDuckGo browser description pages to say something about this. End of my attempt to say what Gabriel should have said.

Leo: Sounds fair. Do you think if they used Google they would have the similar issue?

Steve: I have no idea.

Leo: You don't know.

Steve: And I can't imagine why, like, Microsoft's agreement said you can't ever block our third-party JavaScript. Maybe it's just because Microsoft can set the terms that they wish because they're one of only two games in town.

Leo: This is a duopoly, yeah.

Steve: Yeah. And so it's like, take it or leave it. And so that's why he was talking kind of like apologetically about the fact that, well, we can't provide search unless we have search. And you can only get search from two places. And so he's very proud of their search. And so maybe this is just the way it is.

Leo: It might be. I mean, if he's right, there's really only two search engines, and this is no way at this point for anybody to catch up.

Steve: Oh, my god, no. It would be interesting to know, like, what it costs Microsoft and Google to maintain...

Leo: He says a billion a year. I mean, that doesn't sound unreasonable. They certainly make much more.

Steve: That's a lot of bandwidth. I mean, I've got spiders crawling all over me all the time.

Leo: That's right, yeah. I mean, others have said we are - I think Brave has a search engine. They say we're trying to create our own search index. But everybody, I mean, I think now we know, and I always suspected this, that everybody's either using Bing or Google.

Steve: Yeah. So I think their heart's in the right place. I don't think they would be making an exception to allow Microsoft's domains to run third-party scripts if their search index syndication contract did not require it. They're not happy about it either. On the other hand, they didn't disclose it. And that was the mistake that they made.

Leo: Well, they couldn't, apparently. They were enjoined from saying anything.

Steve: Oh, right. You're right. You're right.

Leo: Yeah.

Steve: You're right, there was...

Leo: Which really, I mean, this all reflects poorly on Microsoft, to be honest with you.

Steve: It's creepy. It's creepy.

Leo: Microsoft says you have to allow our trackers, and you can't tell anybody you're doing it.

Steve: Yeah.

Leo: And I have to imagine, although I don't know, and I hope the security researches are now looking, that any other so-called privacy protection browser...

Steve: Is offering what it claims.

Leo: Yeah. I mean, they must all be doing this. I imagine Google's got even more draconian terms. So I think that's the whole point of DuckDuckGo is, well, we're doing the best anybody can do, given the situation.

Steve: Right. Right.

Leo: Nobody's going to be able to do better.

Steve: And just put UBlock Origin on DuckDuckGo's browser, if you can. I don't know if you're able to run extensions on it.

Leo: Right. Oh, that's a good question, yeah.

Steve: Yeah.

Leo: I mean, everybody said, when this came out, everybody said, oh, I'm going to use Startpage. But I'm pretty sure it's either Google or Bing. I think it's Google. They say we anonymize it. So my question is, maybe Google doesn't have such draconian terms. Maybe Google says, oh, go ahead, you can anonymize your search queries to us. I don't know. I presume people are working on that.

Steve: Certainly I would think they will now. And now our listeners know that DuckDuckGo is not DuckDuckGone unless this really upsets you, in which case, okay, good luck.

Leo: Right. But you're not going to be able to use any search engine, then.

Steve: No.

Leo: No browser. Unless, well, that's again, there are some questions. Maybe Google is - maybe Google's, like, beneficent and says, oh, go ahead. Use our stuff. You don't have to...

Steve: And Microsoft was cheaper, and they...

Leo: Maybe, yeah.

Steve: Well, and they no doubt, I mean, we know that the DuckDuckGo search engine, the search service, predates the browser by many years.

Leo: Right. Right.

Steve: So they probably signed that thinking, hey, that's not a problem. We don't have any problem with being forced to run third-party JavaScript. We're not a browser. We're a service.

Leo: It's not going to be an issue, yeah.

Steve: Yes.

Leo: So really what...

Steve: And then it became one.

Leo: What their mistake was releasing, and this just happened, a privacy browser, and implying that it's somehow protected.

Steve: Yeah. Yeah.

Leo: Okay. As always, Steve opens our eyes to the situation surrounding us. That's why you listen to this show. You can get copies of Security Now! for your friends and for you, share it around, please do, at GRC.com. Steve has a couple of unique versions, a 16Kb audio, the smallest audio version we have. He also has transcripts, text written by humans, not a computer. So it's very good. And then he also has a 64Kb audio version. All at GRC.com, that's Steve's website. It's where you'll find all the freebie stuff Steve does besides this show like ShieldsUP!. But it's also where you'll find his bread and butter, which is the world's finest mass storage maintenance and recovery utility, SpinRite. 6.0 is current. He's working hard on 6.1. As long as he doesn't forget what he did last weekend.

Steve: I'm writing it down before I do so I know what I wrote.

Leo: Document.

Steve: Wow.

Leo: You will, if you buy 6.0 now, get 6.1 when it comes out, and you can also participate in its development: GRC.com. He's on Twitter, @SGgrc. And that's where he tweets the show notes, but also where you can tweet him. His DMs are open, so if you want to leave a message, give him a hot tip, quibble, all of that, just @SGgrc on Twitter.

Steve: You know what you should do?

Leo: Uh-huh?

Steve: Go to GRC.com/dev/SpinRite.

Leo: Okay. Look at that. What is this?

Steve: So there are, if you scroll down a bit, there are, if you scroll down a bit, there are the most - toward the bottom there.

Leo: Oh my god, look at all this. This is like your clippings.

Steve: Well, those are, down there, a little bit lower, those are some of the most recent of the test EXEs along the way.

Leo: Oh, look at that.

Steve: But go to, up higher, previous releases. Click on previous releases. This will give you some idea of these were all test releases that were developed.

Leo: Holy cow. Holy cow. That is a long list. AHCI, so all these AHCI drives.

Steve: Yeah.

Leo: Holy cow.

Steve: And then we get into the read speed work, the init disk work.

Leo: Oh, geez. Look at you.

Steve: I mean, this is like...

Leo: This is heavily tested. Good for you. This is how you should do it.

Steve: Yes, well...

Leo: And I love it that you put this online.

Steve: Yup. Well, that's the way all of our testers get it.

Leo: Right.

Steve: Because it's just right there.

Leo: Makes sense. Nice.

Well, there you have it. That's one of many wonderful things about GRC.com. It's kind of a rabbit hole you can't resist going down. Bring food and water because you'll be there for a while. We have 64Kb audio versions, of course, on our site. We also have video at TWiT.tv/sn. That's all available a couple hours after we finish the show. You can subscribe in a podcast player. That way you get it automatically, whenever it's ready, audio or video. Video versions are also at YouTube. There's a full YouTube channel devoted to this. Steve, have a wonderful week. If you want to watch us do it again next week, it's Wednesdays at...

Steve: Tuesdays.

Leo: Tuesdays. This is Tuesday. Right around 1:30 to 2:00 p.m. Pacific time. Thank you for the correction. I'll make sure to be here Tuesday. 4:30 to 5:00 p.m. Eastern; 20:30 UTC.