SECURITY NOW!

Transcript of Episode #862

QWACs On? or QWACs Off?

Description: This week we briefly touch on last week's Patch Tuesday for both Windows and Android, the world's two most used operating systems. We look at a recent emergency update to Firefox and the need to keep all of our systems' UEFI firmware up to date. NVIDIA suffers a huge and quite embarrassing network breach, and ProtonMail handles their Russian customers correctly. The Linux kernel has seen some challenging times recently, and Russia has decided to start signing website certificates. Research was just published to put some numbers to WordPress add-ons' observably miserable security, and the European Union legislators who brought us GDPR and mandatory website cookie notifications are at it again. What now?

High quality (64 kbps) mp3 audio file URL: http://media.GRC.com/sn/SN-862.mp3
Quarter size (16 kbps) mp3 audio file URL: http://media.GRC.com/sn/sn-862-lq.mp3

SHOW TEASE: It's time for Security Now!. Steve Gibson is here. There's a lot to talk about. A Patch Tuesday recap coming up. NVIDIA gets hacked, and the hackers get a lot of data out of them. We'll talk about a UEFI firmware flaw and the patch that's being put out. And then a proposal from the European Union that's just QWACed. It's all coming up next on Security Now!.

Leo Laporte: This is Security Now! with Steve Gibson, Episode 862, recorded Tuesday, March 15th, 2022: QWACs On? Or QWACs Off?

It's time for Security Now!, yay! I look forward to this all week. Our man of the hour, Mr. Steve Gibson. Why are you laughing? It's true?

Steve Gibson: Oh, Leo. You look forward to a vacation all week.

Leo: Well, I do that also. But no, this is easily the most informative show on the network. I mean, if you want to know what's going on in security, which is really the tip of the spear for all computing these days, you've got to listen to this show.

Steve: I thought about that, like it was 17.5 years ago that you said, hey, want to do a podcast about security? And back then it was like, oh, well, okay, what's a podcast? But also...

Leo: And what's security?

Steve: I didn't know we'd have stuff to talk about for 17.5 years. And actually some of the topics this week were from last week because I wanted to stay focused on the consequences of what is going on in Russia with cybersecurity. And I've got a few things that couldn't fit in this week which we'll be talking about next week. So yes, our cup runneth over with security problems.

So this is Episode 862 for the middle of March. And I'm looking forward to next week because that's going to be 3/22/22. But that's next week. This is 3/15/22. And of course we missed Pi Day, 3/14.

Leo: It's the day after Pi Day.

Steve: Yes. So this one I titled it QWACs On? Or QWACs Off? And those old-timers among us will know about wax on, wax off. We're going to briefly touch on last week's Patch Tuesday, both for Windows and Android, which of course are the world's two most used operating systems, one for desktop, one for mobile. We look at a recent emergency update that Firefox received, and also the need to keep an eye out for upcoming UEFI firmware updates for our various machines.

A research firm plowed into a widely used SDK used by many firms, I'll enumerate them later, to develop their own custom UEFI firmware. And as we've seen before, you make a mistake in the example code, everybody just copies it. So anyway, we've got a lot of copy problems which need to get patched. But the point is that it really does undermine the guarantees that UEFI is here to provide, keeping things from crawling into our operating systems before they've had a chance to get booted.

Also NVIDIA has suffered a huge and quite embarrassing network breach with some consequences for them. ProtonMail handled, I think, their Russian customers correctly, so I just wanted to give them some props. But Tutanota did also, yeah, it's actually even better. The Linux kernel has been seeing some challenging times recently, so we're going to talk about that. And Russia has decided to start signing their own website certificates. And this is one of those what could possibly go wrong?

Leo: Oh, boy.

Steve: Research was also just published to put some numbers to my constant WordPress grumbling about add-ons, the lack of security in their add-ons. We now know exactly just how bad. And I guess if I could encourage people to just use the root WordPress install and try not to embellish it with all the add-ons, that would be the takeaway from that. And finally, the European Union legislators who brought us the GDPR and mandatory website cookie notifications are at it again with QWACs. So we'll be talking about that. And we do have a fun Picture of the Week from one of our listeners.

Leo: And as you say, QWACs on, QWACs off.

Steve: QWACs off. What could they have done now?

Leo: I'm trying to find it, but I remember seeing on Reddit somebody wrote an app that ran in UEFI. So I think it was Wordle or something. So, I mean, because I guess UEFI there's enough space in there, you can put an app in there.

Steve: Yeah.

Leo: So you boot your computer to Wordle, apparently, I guess. I don't know why you'd want to do that, but you know.

Steve: Hopefully you can then exit and go back to your regularly scheduled OS.

Leo: That's right. I'm presuming there's a menu, I would hope, I would hope. Picture of the Week time, Mr. G.

Steve: So, okay. This is a one-off, or a one-of-a-kind. One of our listeners was inspired by the question that I answered the other day about how one would detect a BGP hijack where an illicit cert was obtained on the fly immediately after the hijack in order to impersonate a website. And of course the way is, as I said, to check the certificate's fingerprint because the fingerprint is something that cannot be copied. It cannot be duplicated by someone who's spoofing it. So a fraudulent certificate with the same domain name, affecting the same site, would have a different fingerprint.

So this enterprising individual took the current certificate fingerprints of his most important sites and created a label for a coffee mug. So we have a coffee mug, a nice white ceramic coffee mug, around which is wrapped this label. And the headline says: "Honey, You've Left Thumbprints All Over My Cup." And so he's got the certificate thumbprint for the Bank of Tampa. Where do you imagine he lives? Fidelity Investments. TD Ameritrade. PNC Financial Services, Synchrony Financial, the Bank of America, Mastercard, and as a nod to me, Gibson Research Corporation.

Leo: I love it. I love it.

Steve: And as I was looking at it, I was saying, oh, August 25th, '22 is when my cert expires. Of course you notice nobody's - that's good to know - nobody's cert is expiring long from now. And of course we know why, because they only get a year's life.

Leo: They don't do long ones anymore, yeah.

Steve: Consequently, yeah. So everybody's '22 and '23. But anyway, I just thought this was a kick because if you were going to the site, and you were concerned that maybe you were not at the right place, you could just pick up your coffee cup, use it as a reference for the proper thumbprint for the certificate, and then, baby, you know you're talking to the right people.

Leo: That's pretty funny. That's great. I always - I never do the whole number, but I look at the first four and the last four, and if they match I figure good enough.

Steve: Yeah. It is a hash. And so with any hash the chances of, let's see, if the first four and the last four, that's going to be on the order of 32 bits, so that's one in 4.3 billion...

Leo: Good enough.

Steve: ...chance that, yeah, it's good enough. Okay. So we had Patch Tuesday last Tuesday. And the world's dominant and increasingly rudderless desktop operating system, Windows, received fixes - boy, I tell you, Leo, listening to Paul and Mary Jo is just like, well, it's called Windows Weekly, and that's W-E-A-K-L-Y. Boy.

Leo: You know, it's kind of sad because if you have a show called Windows Weekly, you have to use Windows.

Steve: Yeah.

Leo: But you get the feeling that maybe they're thinking, I wish I didn't have to use Windows.

Steve: Well, it is Mary Jo's world; right?

Leo: Yeah.

Steve: I mean, so she's - that's really her thing.

Leo: She has to.

Steve: But you really see Paul, like kind of like...

Leo: Oh, groaning, groaning. Well, you, too. I mean, you have to because SpinRite is a Windows product.

Steve: Yeah.

Leo: And so you have to use Windows. But boy, Linux just looks better and better as Microsoft keeps making these mistakes.

Steve: Leo, this is not in the news today for me, but they're now experimenting with adding ads to...

Leo: Ads, yeah, in Explorer.

Steve: Yes.

Leo: Yeah, yeah, I saw that. It was just like, what? Stop. You don't make enough money? Is that it? Is that the problem, Microsoft?

Steve: I know. And I just, you know, I'm talking to you on a brand new Win10 setup. And I had to remove Candy Crush Soda Saga.

Leo: Oh, yeah. Oh, yeah.

Steve: It's like, what?

Leo: Why is it on there?

Steve: This is in my Start menu. What is wrong with these people?

Leo: Yeah.

Steve: Yeah. Anyway, they fixed 71 - yes. So I was just thinking the same thing. I was thinking, you know, I mean, if anyone should be moving to Linux, it's I. I should be moving to Linux. But I do, I do have many things that are Windows only. And, for example, the next development platform I'll be switching to for SpinRite 7, SpinRite will be running on a little custom 32-bit real-time operating system in order to get dual boot over UEFI and BIOS so that it can run either way. Well, it supports remote debugging, which is the only way to debug, especially on a little turnkey RTOS, you know, it's not going to have a full desktop environment. So I'll need to be reaching over to that machine and debugging on a separate machine. That's Visual Studio. That's what it supports. It doesn't support other Linux-hosted environments. It's Visual Studio on Windows specifically.

Leo: Well, yeah. And if you're going to do a security show you have to be on the insecure operating system so you can talk about it, I guess.

Steve: Yeah.

Leo: Yeah. And you can secure yourself. You can get rid of Candy Crush Soda Saga.

Steve: Oh, no, it's not the security that I'm worried about. Leo, I'm running on Windows 7. Security's not a problem. It's just the annoying things they do. Paul was complaining that he'd been writing a book for several years. His Windows had been watching him writing this book. Yet when he searches, he uses the search for the book's title, trying to find some documents relative to it that are, like, named that, he gets Bing results from the web. It's like, what? It was like some steakhouse. It's like, oh, he was writing about longhorn, and he was getting...

Leo: Longhorn Steakhouse.

Steve: ...Longhorn Steakhouse results.

Leo: Yes, that's right.

Steve: And it's like, hey, Windows, there's files named "Longhorn" right in front of you. That's what I'm trying to have you help me find. No. How would you like your steak? Rare? Medium rare? Oh, yeah. Anyway, it's - so, yes, not for security. But we're going to talk about security eventually here.

Okay. So last week Windows fixed 71 known flaws, three of which had been publicly disclosed, none which were exploited. But of course Microsoft calls these "zero-days" because someone surprised them about them. Two of the three did have publicly disclosed exploits and their remote code execution. So again, the advice is always going to be update as soon as you can. 29 of them were remote code execution (29 of the 71 were RCEs); 25 elevation of privileges; six information disclosures; four denial of services, meaning that you can crash something, and then it won't serve you; and then three each security feature bypass, which is so generic it's annoying. Security feature bypass, gee, isn't that everything?

Anyway, and then we have spoofing vulnerabilities, three of those. And then, oh, on top of that we did have 21 problems identified and fixed in Microsoft's Chromium-based Edge browser. And I always kind of wonder, because we're not told, if those are things Microsoft did to Edge-ify it, or if those are in the Chromium core, and they're just fixed because everybody who uses Chromium got those 21 things fixed? It's not clear from what Microsoft is saying. So anyway, I trust everybody has patched their Windows and gone through their update cycle. Last time it happened for me I had to switch computers because it stopped being fast enough to run Zoom.

Leo: I'm liking this one, by the way. And the picture's really good.

Steve: Oh, good.

Leo: Are you using the same camera?

Steve: Everything's the same. Same, same, yeah.

Leo: Yeah, looks really good. It's a higher quality image.

Steve: Well, I can even see, yes, I can even see that the image quality is higher, and the refresh rate is better.

Leo: So it was bogging. That's all. It was just an old machine, too slow, yeah.

Steve: I'm sure it was. I'm sure. Okay. Also, as I said, Android. The most severe issues fixed, there was a critical security vulnerability in Android's system component that could lead to remote code execution with no additional privileges needed and no user interaction. So those are the things that the people who are trying to get into other people's phones in targeted attacks are trying to use. And of course Android is the majority mobile platform on the globe. So update Android also, as always.

Just over a week ago, my Firefox browser jumped to update and restart itself. Like it just runs statically over on my left-hand screen all the time, and it's sort of my reference browser. And so it's rare that I see it, like, waving at me, hey, Gibson, restart me. And so I did. And that was to eliminate a pair of high-impact zero-day vulnerabilities, both of which Mozilla said were seen actively exploited in the wild. And that's a little bit of a rare thing for Firefox because it's a little bit of a rare browser, increasingly rare these days. Everybody's kind of gone Chromium-ized. But bless their hearts, not Mozilla.

So there were two CVEs issued for these two zero-days. I mean, these are like real zero-days; right? They were found being actively exploited, 2022-26485 and 486. They're both once again use-after-free flaws impacting the Extensible Stylesheet. Sorry, Stylesheet. I like Extensible Stylesheets. I don't have anything against them.

Leo: It's just hard to say.

Steve: Yeah. Extensible Stylesheet Language Transform - oh, Leo, speaking of hard to say, wait till we get to the Russian Kremlin Gremlins. There's a tongue twister.

Leo: Oh, boy.

Steve: The Stylesheet Language Transformations (XSLT) parameter processing, and the WebGPU - easily said - interprocess communication, the IPC Framework. This XSLT is an XML-based language used for the conversion of XML documents into web pages or PDF docs, which Firefox can do, and this WebGPU is the emerging standard that's expected to supplant the current WebGL JavaScript graphics library. Both of those had problems.

In the case of this XSLT, it was found that the removal of an XSLT parameter during processing could lead to an exploitable use-after-free flaw. And the problem is this could be done remotely. And as for this use-after-free, as is often the case, we were talking about this recently, with dynamic memory management, a reference count is kept by the memory management system on behalf of the programmer so that they don't need to worry about it. So that frees the programmer from needing to do any kind of reference counting manually.

And it's a convenience that the so-called "garbage collection" is done in the background without the programmer needing to bother. But the only way the system knows it's safe to release and free an object that had been allocated is if and when that object's reference count drops back to zero. So the first time it's - the first instance of a reference or actually every instance of a reference increments the reference count. And so once there are no more references in context, the garbage collector says, oh, okay, nobody else is using this. I'm going to let it go.

The problem is, and it's cool, but it's one of those inherently brittle technologies. It's great when everything works exactly right. But mistakes are very difficult to spot. And normally it's the kind of thing that needs to be seen happening, which is why these problems generically, these use-after-free problems, keep recurring.

And I guess if our listeners wanted to explain the trouble to some non-techie, they could say that use-after-free bugs, which can sometimes be exploited to corrupt data and execute an attacker's code, arise from a confusion over which part of a program is responsible for freeing the memory. So that's sort of, in a nutshell, that's the problem. And again, modern languages doing more for their programmers behind the scenes; but when they make little mistakes, they can be leveraged.

So the other use-after-free glitch, because these were both that, was caused by an error when processing messages in the WebGPU's interprocess communication framework. A remote attacker could trick their victim into opening a specially crafted web page, trigger this use-after-free error, and execute arbitrary code on the victim's system. And as I mentioned, this XSLT vulnerability was exploitable in the same way, just open the wrong web page and BLAMMO.

So last week you would have wanted to be at Firefox 97.0.2, which is what it updated me to. But just now when I checked it wanted me to restart, which I did, which moved me from 98 to 98.0.1 on the desktop. So you might want to check if you're using Firefox to make sure that you're current. Again, almost certainly targeted attacks. Remember the so-called Watering Hole attacks where something is done to lure a targeted victim to a website in order to get them compromised. So sort of a variation on phishing. But thus called a "watering hole"; right?

So anyway, Firefox. We've talked a lot about the surprisingly daunting challenge of booting a computer while maintaining true provable security. But this really should not be that surprising. When you think about it, the reason computers have been such an incredible breakthrough and boon for mankind is that they are so flexible. It's a machine that follows instructions. The trouble arises because there's no way for it to know whose instructions it's following. It doesn't care. And its limitless vulnerability arises from its incredible flexibility.

And in security we have this concept of a chain of trust where that chain is anchored by a root of trust. And this applies to booting up an operating system. If the operating system is signed by its publisher, any single bit that's changed will break the signature, which is just a signed hash. So the theory is the system's firmware simply needs to load the code that's been signed, and then verify the operating system's cryptographically secure signature before it turns control over to it. Doing that will prevent anyone from modifying the operating system's code. Not one bit can be changed because the code gets hashed and signed, and a bit will completely change the hash.

Which, Leo, is why you and I only check the first four digits, first and last four digits. It's like, impossible for the majority of them to be changed. In fact, we know, if you change one bit, on average half of the bits in the hash will be changed. So thus our strategy.

Anyway, so the concept is good. But the question is, if the UEFI is performing that work, who's making sure that it's doing it right? On one hand, having the system's startup firmware code being firm and not hard creates a vulnerability because that means it can be altered. On the other hand, since we appear to be unable to get it right, if it's firm it can at least be improved when problems with it are inevitably found, as they have in this case been.

So the good news is there are people who are focused upon improving the situation with firmware. At the beginning of last month, a group of researchers from the firmware protection company Binarly discovered a raft of critical vulnerabilities in the so-called "InsydeH2O" (I-N-S-Y-D-E), InsydeH2O UEFI firmware. It is a cross-vendor firmware used by many computer vendors including Fujitsu, Intel, AMD, Lenovo, Dell, ASUS, HP, Siemens, Microsoft, and Acer. So, like, right, that's - who doesn't use them? And actually I think that's a good thing.

Binarly found 23 flaws in this InsydeH2O UEFI firmware kit, essentially, most of them occurring in the software's System Management Mode code, which provides the systemwide functions such as power management and hardware control. Since the System Management Mode privileges inherently exceed those of the OS kernel, which it is responsible for booting, any security issues there can have severe consequences for the vulnerable system.

The flaws that were found in this InsydeH2O firmware can collectively and individually be used to invalidate many hardware security features including Secure Boot and Intel's BootGuard, to install permanent and persistent software that cannot be easily erased, and to create backdoors and back channel communications to steal sensitive data. As we've talked about this whole baseboard processing now, there's a whole processor separate from the Intel chip that we stick in a socket when we decide which model that we want, that runs the motherboard and all of its things. And it's very capable.

So as I said, there were 23 flaws found with three of them obtaining CVSS scores of 9.8. So this is in the baseband processing firmware. Binarly's disclosure report explained that the root cause of the problem was found in, as I mentioned at the top of the show, the reference code associated with InsydeH2O's firmware framework. In other words, as we've seen before in other contexts, all the manufacturers derive their own firmware from Insyde's firmware SDK reference code to develop their customized UEFI firmware.

And that's certainly reasonable. They bought a license to do so. That's what they did. No reason to recreate the wheel. The chances are when you do you're going to make mistakes. So that's great. And so frankly I'd rather have everyone working from a common codebase than each rolling their own, since getting this exactly right is crucially important. And when it's fixed for one, it's fixed for all.

InSyde Software has released firmware updates to fix all the security vulnerabilities that were identified by Binarly, and they've published detailed bulletins to assign severity and descriptions for each flaw. So all of the people using their kit. Hopefully there are engineers right now that have been like working through this and working to update the firmware across all their products. Of course we always have the problem of the inside UEFI stuff having been around for a long time, and products going out of support and no longer receiving firmware updates. It's difficult to require everybody to support everything forever. So these are really bad, and now they're becoming known.

So being individually customized firmware on a per product and per manufacture basis, they all have to be created, adopted, incorporated, downloaded, installed. Every single vendor needs to do this. And that way it's a little bit like the Log4j mess where it is something core to many different products and publishers. So this is why I said at the top we can expect to see important firmware updates coming from many of our hardware vendors.

And I'm not suggesting that this will be a widespread attack or that that will result. It's very likely that these would be used, you know, these are going to be sophisticated, targeted attacks. You need to get onto the system first. Then these are being used to achieve persistence, you know, getting down into the firmware. So I did not look in detail to see what three of these things had a 9.8, but it is possible that this could be exploited remotely because there is now management technology in some of these motherboards which allows remote, over-the-network management of them. So that may be why they're at 9.8. Still, it just makes sense to keep your firmware updated.

And then, in addition to this, on top of these 23 problems, last week HP disclosed an additional 16 high-impact UEFI firmware vulnerabilities that Binarly had found which affect multiple HP models, and HP is a user of Insyde also. And that included laptops, desktop computers, point-of-sale systems, and edge computing tools. These flaws allow malware to survive hard drive replacement and operating system reinstallation.

So anyway, a long time ago we talked about how rootkits are able to hide in plain sight by doing something as simple as hooking an operating system's directory listing API, to simply remove references to any of its own files from the list. So you do a dir, or any of your programs do a dir, and they don't see any of the files that are sitting in the directory right in front of them. I remember it was the Sony rootkit that we talked about in the way long time ago. And it's just unnerving to imagine that something that simple, you know, I mean, it shows how much we assume that our operating system is doing what we expect and what it should; where in fact, as I said because software is infinitely flexible, it's so easy for it not to be working the way we want it to. So anyway, keep an eye on firmware updates. And a huge thanks to Binarly for digging in, for taking the time to dig into Insyde's UEFI offerings and help make our stuff better.

Leo: This seems like UEFI would be a really good vector for attacks because it's basically a mini operating system that runs before your system. And it's persistent, as you point out.

Steve: It has a file system.

Leo: Yeah. They can do anything. It's Turing complete. I mean, it's written in C, and it's basically, if I were going to be a bad guy, that would be the best place to put malware because it would survive a reinstall and everything. It wouldn't survive...

Steve: Yeah.

Leo: It usually lives on a hard drive; right? It would not survive a complete nuke of the hard drive.

Steve: No, no, it's in the onboard firmware.

Leo: It's in the firmware?

Steve: Yes.

Leo: There's a UEFI partition, though, on the drive that also contains code.

Steve: Yes. And so that could be a problem, although these guys specifically said that this would survive a hard drive replacement.

Leo: Oh. All right. So it is in the firmware. Wow. Yeah.

Steve: Yeah.

Leo: Wow.

Steve: Yeah, nobody wants that.

Leo: Yeah.

Steve: So NVIDIA suffered a serious network breach last month. The hacker extortion group by the name of Lapsus$, L-A-P-S-U-S, then with an extra dollar sign appended to the end, claims to have exfiltrated, and all evidence is they probably did, about a terabyte of NVIDIA's very proprietary data during the attack. They extorted them. NVIDIA said go away. So they began leaking NVIDIA's data online, as I said, after NVIDIA refused to negotiate.

On February 23rd, emails and NTLM, you know, NT LAN Manager, you know, Windows password hashes for 71,335 NVIDIA employees were leaked on the Internet. Now, this was a bit puzzling since NVIDIA currently has only 18,975 employees. But the hackers claimed to have had deep total access to NVIDIA's system, where they said they roamed around for about a week. And so the presumption has been that the leaked credentials include deep past employee files, as well, which would explain the high count.

Then four days later, on February 27th, that hacking group Lapsus$ claimed it had been digging around for a week, and for some reason made a point of clearly stating that they are not state-sponsored, and that they are "not into politics AT ALL," in caps on the "AT ALL." So such is the world we're in today.

Troy Hunt's "Have I Been Pwned" site now has the leaked data and noted that many of the hashes are being cracked and circulating within the hacking community. So I'm sure that NVIDIA has already made sure that all of their employees have changed their passwords because otherwise that would even be worse.

Lapsus$ also stole, I mean, apparently they just got everything. Think about it. I mean, we just glibly talk about a terabyte here, a terabyte there. But that is a trillion bytes, a thousand billion bytes of data. So it's probably everything. And among the everything was some NVIDIA expired driver signing certificates. And what's confusing to me is that the reports are, and they've been confirmed by many outlets, these expired driver signing certificates were immediately put to use signing malware components in order to get them past antivirus and Windows' own driver signing protections. So expired. What's not at all clear to me is why Windows would load any driver signed with an expired certificate? But apparently it's doing that.

Code signing works differently from TLS web server certificate signing. The reason for the difference is that servers and clients, web servers and clients, are by definition always online and talking to each other, and servers are thus always able to present a certificate that's currently valid, so that's what we force them to do. But that's not the case with code; right? A driver might be signed 10 years ago, and the certificate with which it was signed will have long since expired, that particular certificate. So we want drivers to be valid in the future, even if a machine is not online. So there's no, like, online OCSP-style verification. And the point is, was the certificate valid when it was signed?

So that's what we did. We solved this problem by changing the requirement, the validity requirement for code signing. We require that code is signed by a certificate that is valid at the time the code was signed. But that means that we need to know when the code was signed. And that information is incorporated by requiring co-signing of the certificate by a valid certificate authority which offers a timestamping service. So at the time the code is signed, the signing computer obtains a certified timestamp from a timestamping service which is all bundled into the final signature. So it's sort of like having an online notary saying, yes, here I am, I'm asserting that this is the current date and time that all of this is happening. So this attests to the time that the signing was done.

Now, Bleeping Computer showed the digital signature's property page of a piece of malware signed with NVIDIA's expired cert. It shows that there was no timestamp included. And in fact you don't need to include a timestamp. That is to say, the act of signing doesn't require a timestamp. But the entity trusting the signature is supposed to require it. I mean, that's the whole point. But all I can guess is that for the sake of compatibility, Microsoft doesn't enforce timestamping for at least some uses. And "some" apparently includes kernel drivers because that's what was being signed by these expired NVIDIA certificates. They've immediately been blacklisted. There are now, you know, we will not trust anything signed with these expired certs. But, boy, you know, it took somebody using them maliciously to make that happen. Which to me just seems crazy. But anyway, that seems to be what's going on.

And also on March 1st, the Lapsus$ group demanded that NVIDIA open source their proprietary drivers for Windows, Mac, and Linux. I have a picture in the show notes of their ransom demand. Dated March 1st, it says: "After evaluating" - this is Lapsus$ talking, the bad guys. "After evaluating our position and NVIDIA's, we decided to add one more requirement. We request that NVIDIA commits to" - in all caps - "COMPLETELY OPEN SOURCE (and distribute under a FOSS license) their GPU drives for Windows, macOS, and Linux, from now and forever.

"If this request is not met, on Friday we will release the" - and now we have in caps - "COMPLETE SILICON, GRAPHICS, AND COMPUTER CHIPSET FILES for all recent NVIDIA GPUs, including the RTX 3090 Ti and UPCOMING REVISIONS!" And they say: "Of course, this includes all files with extensions such as .v, .vx, .vg, and more."

And so they finish: "So, NVIDIA, the choice is yours. Either, one, officially make current and all future drivers for all cards open source, while keeping the Verilog and chipset trade secrets - well," they said, "secret, or not make the drivers open source, making us" - right, they're being made - "to release the entire silicon chip files so that everyone not only knows your driver's secrets, but also your most closely guarded trade secrets for graphics and computer chipsets, too. You have until Friday. You decide."

Leo: Oh, lord.

Steve: I know. Now, this .v, .vx, and .vg file extensions are for Verilog, which is the Hardware Description Language (HDL) used to model electronic systems. It's the most commonly used in the design and manufacture of integrated circuits, and it has become an industry standard, standardized as IEEE Standard 1364. This means that these guys have the designs of NVIDIA's chips. Which, yes, NVIDIA certainly intended to be kept inside. And, you know, while at first this seems a bit breathtaking, it's unclear what real value those would be.

I mean, maybe there's proprietary stuff that could be reverse engineered out of them. But no other reputable foundry is going to make NVIDIA clone chips. They'd be litigated into nonexistence, for one thing. And, well, I was thinking about this, maybe Russia would, except that Russia doesn't have the process technology to make any state-of-the-art chips. They're unable to do it. So NVIDIA's designs would be of little use to them.

Last year, NVIDIA also came up with a technology known as Lite Hash Rate, or LHR, which reduces the value of its GPUs for cryptocurrency mining. Their aim is to make a very powerful graphic processor, but one which would not be very good at cryptomining, in order that, you know, gamers can actually have a chance to buy some of these things, rather than them all being sucked up by the cryptominers. And apparently this also annoyed the hackers, who have said they want the hash rate limiters to be removed from the chips.

So my feeling is mostly this is a huge embarrassment for NVIDIA. And it is a perfect example of just how much damage can be done when all of a company's proprietary value takes electronic form, as NVIDIA's is, and thus can be downloaded, shared, and unfortunately escape from their control. Wow.

Leo: By the way, I just wanted to show you something that came up in the Discord. Did you see this? In Crystal City, Virginia?

Steve: Yup.

Leo: Instead of telling you when it's safe to cross the street, the walk signs in Crystal City, Virginia are telling you "Change your password. Change your password. Change your" - I'm going to have to turn on the sound. But clearly somebody got...

CLIP: Change password. Change password.

Steve: That's brilliant.

Leo: Clearly somebody got into the system. And instead of being malicious, they just thought they'd have some fun. Which I think is great.

Steve: Yeah, I think it's actually wonderful. It's a little bit of a Rickroll.

Leo: Exactly.

Steve: I mentioned that I wanted to just give ProtonMail, which I know is a very popular email service among our listeners, a bit of a thumbs-up. Last week I grumbled about Namecheap's unilateral, and I felt questionable decision to dishonor their previous prepaid commitments to their DNS domain registrants who were at the time given one week to find another DNS provider before their domain name resolution would be summarily terminated. I think ProtonMail did the right thing.

Thanks to Bleeping Computer, who was able to - actually I think one of their employees is a ProtonMail user who received a notification, preemptive notification from ProtonMail that said: "Dear Proton community member. As you may have heard, Mastercard, Visa, American Express, PayPal, and numerous other financial institutions have announced that payments into and out of Russia will soon be cut off." And they said, "(if this hasn't happened already). This means paying for your Proton subscription with your current payment method will likely soon be impossible.

"While many companies have announced that they will no longer serve Russian customers, at Proton our mission is to defend online freedom everywhere. We remain committed to serving the people of Russia for as long as possible. The present difficulties, however, may take some time to resolve. If you are a Proton subscriber, we suggest renewing your subscription or purchasing credits before all forms of payment are cut off in the coming days.

"We are committed to not cutting off any users in Russia for financial reasons for as long as possible during this difficult time. If you are able to use alternative payment methods, your support will ensure that we can continue to serve the Russian people in many years to come. Thank you again for supporting our mission. Best regards, the Proton Team." And they also in this provided a list of alternative payment methods, including cryptocurrency, that could be used by Russian users to renew their subscriptions.

So I thought that was, you know, although yes, it's a little self-serving, they're saying hey, you know, pay in advance quickly if you don't want services to be cut off, to me their heart was more in the right place than Namecheap. I'll also note, however that separately, for those based in Russia, Belarus, or Ukraine, another encrypted email provider, Tutanota, has offered to renew subscriptions free of charge for users unable to make payments during the current situation. So some companies are doing the right thing.

Linux has been having a rough time of it lately. The last year has seen this ever more popular server and desktop operating system hit with revelations of multiple high-profile elevation of privilege vulnerabilities. There have been problems in Linux's iSCSI subsystem, in the kernel, in the Extended Berkeley Packet Filter and, as we talked about at the time, in the Policy Kit's Polkit pkexec component. And there have been two additional recent problems.

The first bears the unfortunate name "Dirty Pipe" and was discovered by a guy named Max Kellerman who discovered, mostly by accident - well, okay, something happened by accident. His actual discovery was certainly not that. But this event was an accident. It was an important locally exploitable vulnerability which was introduced into the Linux kernel at v5.8. Because the exploitation of the flaw allows overwriting data in arbitrary read-only files, it could be put to some very creative and not generally beneficial use, leading to privilege escalation because unprivileged processes could be able to inject code into root processes. So in some ways this is reminiscent of the 2016 "Dirty Cow" Linux vulnerability, though this one is even easier to exploit. But it does explain, although it doesn't forgive, this vulnerability's name, Dirty Pipe.

So in Max's original posting, he explains how this all began. And I'll just share the beginning of this because it was lengthy. He said: "It all started a year ago with a support ticket about corrupt files. A customer complained that the access logs they downloaded could not be decompressed. And indeed, there was a corrupt log file on one of the log servers. It could be decompressed, but gzip reported a CRC error." He said: "I could not explain why it was corrupt, but I assumed the nightly split process had crashed and left a corrupt file behind." He says: "I fixed the file's CRC manually, closed the ticket, and soon forgot about the problem."

He said: "Months later, this happened again and yet again. Every time, the file's contents looked correct. Only the CRC at the end of the file was wrong. Now," he said, "with several corrupt files, I was able to dig deeper and found a surprising kind of corruption. A pattern emerged."

Okay. So then Max goes on, as I said, at great length to explain how this kept nagging at him until he finally had to sit down and get to the bottom of it. After a great deal of analysis and experimentation he finally figured out how to force the file corruption bug to occur at will. That was the key. Max wrote: "All bugs become shallow once they can be reproduced." And of course that is, you know, that's any software developer's dream is to be able to reproduce the problem. I've referred to motherboards that I've been purchasing, old motherboards, off of eBay because one or several or the people testing SpinRite will all have a really weird problem.

For example, just to segue for a second, we've got one motherboard which will not run SpinRite from USB. But if you copy it to the hard drive, it'll run it from there. If you copy it to a RAM disk, it'll run it from there. But it actually won't execute it from the USB drive. Nobody else has this problem. But this one particular gigabyte motherboard has a problem. I actually found an instruction that would not execute from USB. It's insane. But anyway, as I said, if you can reproduce a problem, you can fix it. I did reproduce it, and I did fix it, even though I don't understand it. But now it works.

Anyway, Max then tracked it down to the bug, which he located in the Linux kernel. He then checked Linux 5.10, which is the Debian Bullseye build, which had the bug. But the bug was not present in the previous Linux 4.19, Debian Buster. Those two releases were separated - I hope you're sitting down, Leo. Those successive Linux releases were separated by 185,011 Git commits.

Leo: Wow.

Steve: Oh, baby. But using Git's various tracking tools he was able to locate the one commit, made nearly two years ago on May 20th of 2020, which introduced this very subtle bug into the Linux kernel. That change refactored Linux's pipe buffer code for anonymous pipe buffers, changing the way the "mergeable" check is done for pipes, and in doing so introduced an extremely subtle flaw. The vulnerability has been fixed in that one, in Linux 5.10.102, 5.15.25, and 5.16.1.

So that was what Max found, the first of the two problems. Then two researchers at Huawei discovered a vulnerability in Linux's "control groups" feature which allows attackers to escape containment, to escalate privileges, and execute arbitrary commands on a host machine. Now, Linux control groups, or "cgroups," as they're known, allow sysadmins to allocate computing resources such as memory, bandwidth, and such, among whatever processes might run on a system. These cgroups provide fine-grained control over allocating, prioritizing, denying, managing, and monitoring system resources. This means that cgroups can be a powerful tool for control and security within a system. And in fact they're used by containers like Kubernetes.

A feature known as the "release_agent" file allows administrators to configure a release agent program that would run upon the termination of a process in the cgroup. That meant that attackers who are capable of writing to the release_agent file could exploit it to gain full admin privileges. And Linux wasn't checking that the process setting the release_agent file had admin privileges. So basically a very simple problem which through a chain of complex features ended up being exploitable. And all of this amounts in this case to a container escape, as I mentioned, for example, within a Kubernetes environment, which would provide attackers with access to other users' containers in public cloud environments.

And we're not finished with Linux yet. Yesterday another just-disclosed security flaw came to light. This one can be leveraged by a local adversary to gain elevated privileges on vulnerable systems to execute arbitrary code, escape containers, or induce a kernel panic. It's CVE-2022-25636 with a CVSS of 7.8. It impacts Linux kernel versions 5.4 through 5.6.10. It's the result of an out-of-bounds write, in other words, a buffer overrun, in the heap of the kernel's netfilter component. The flaw was discovered by a guy named Nick Gregory, who's a researcher with Capsule8.

Red Hat explained that: "This flaw allows a local attacker with a user account on the system to gain access to out-of-bounds memory, leading to a system crash or a privilege escalation threat." In addition to Red Hat, similar alerts have been released by the maintainers of Debian, Oracle Linux, SUSE, and Ubuntu. Linux's Netfilter is a framework provided by the Linux kernel that enables various networking-related operations like packet filtering, NAT, and port translation. The problem results from incorrect handling of hardware offloading that could be weaponized by a local attacker to cause a denial-of-service or possibly execute arbitrary code.

The idea of hardware offloading is that there are a number of things associated with sending and receiving network packets that are very simple to do, such as calculating a packet payload's checksum, and so they're a waste of the processor's time. They're time-consuming, like to do a checksum you've got to do math on every single byte that's going down the pipe and then just dump the sum of them there. What happens is when you have a NIC which indicates that it's able to automatically do checksuming, then the software just skips that completely, and the NIC hardware itself plugs the proper checksum into the packet payload on its way out.

So another example of things coming on the way in is packet coalescing can be done in hardware to reduce the packet count, thus reducing the per-packet processing that needs to be done. So what's happened is over time, as NIC hardware, Network Interface Controller hardware, has become increasingly capable, the more and more duties have been pushed down to the hardware level to free the software from doing things. It's a waste of its time.

Anyway, Nick Gregory, who discovered this problem, explained. He said: "Despite being in code dealing with hardware offload, the flaw in netfilter is reachable when targeting network devices that don't have offload functionality, for example, the local virtual interface. This can be turned into kernel Return-Oriented Programming (ROP), local privilege escalation, and other things without too much difficulty, as one of the values that's written out of bounds is conveniently a pointer to a net_device structure."

Okay. So what does this mean? Does the spate of flaws suggest that Linux is becoming rickety? No. I think it is an indication of it becoming increasingly complex. As we know, security and complexity are perpetually at odds with one another. For security, scrutiny is always a good thing. At this point it's the only thing I think that's saving us is like people digging into UEFI and finding problems and surfacing them. And all of the bug bounties that we have encouraging people to look closely at code. We've seen that just making something open source doesn't automagically bring more security. But I really do think in the long run it feels like the right solution, and that it's going to be the approach that wins.

Okay. Our web browsers and operating systems all collectively trust any security certificate, such as a brand new web server TLS certificate that's never been seen before, because it's been signed by a Certificate Authority that is trusted globally by collectively all of our OSes and web browsers. But the reason they trust a never-before-seen certificate is specifically and only because the signer of the certificate has promised to, and demonstrated its ability to, restrict its issuance of certificates or signatures on certificates to entities whose identity it has confirmed within the guidelines set up by the CA Browser forum.

Several times during this podcast's life we've discussed instances of apparent mistakes being innocently made, flaws in certificate issuing systems which were being exploited, and also clear instances of deliberate certificate signing abuse. So everyone's a bit concerned over Russia's recent announcement of its creation of a new, unvetted, untested, and very likely rogue Russian Certificate Authority. However, its creation was probably inevitable because the sanctions imposed by Western companies and governments are preventing Russian web services from renewing their expiring TLS certificates in absolute good faith. They would like to.

But Western certificate authorities are banned through sanctions to do business with entities inside Russia. So what are they going to do? We all know that browsers take a very dim view of websites which offer them an expired certificate as proof of their identity. So that won't work. So if web service providers whose certificates are expiring are unable to purchase certificates from traditional certificate authorities, what choice do they have? No problem. Just pick up a certificate issued by your local authoritarian regime.

Leo: Of which there are many choices.

Steve: Exactly. None of whom are trusted. Now, there's only one problem with doing that, which is that no mainstream web browser will accept any certificate randomly signed by Russia's newly minted CA. That's just not going to happen in this lifetime. Okay, well, there is an exception. Russia does have the Yandex browser, based in Russia, and it will now, no surprise, work with the Russian Ministry of Digital Development-issued certificates. So Russian sites are instructing their visitors to please switch over to Yandex.

But of course there is an alternative. If someone wished to continue using Firefox, or one of the several Chromium-based browsers, in theory they could do so by adding the Russian-trusted root CA certificate to their browser's certificate root store. But just to be clear, that's the last thing that anyone outside of Russia in the West should consider doing. Nothing would or will prevent Russian Kremlin Gremlins from creating TLS certificates for any Western websites, thus allowing them to intercept, spoof, and alter such websites' network traffic.

What will likely happen, and it's probably already in the works, if it's not already got the seal of approval stamped on it, is that Chromium and Firefox, maybe even iOS if you're able to install certificates into iOS, I didn't look, will specifically blacklist Russia's rogue root certificate so that no one using Firefox or Chromium browsers or a Chromium browser could be hurt after mistakenly installing it into their browser's root stores.

So for those using the Yandex browser in Russia, it's a workable solution for providing continuity of services during this time. But there's no conceivable way that any of our mainstream browsers are going to trust any certificates signed by Russia's new Certificate Authority.

Okay. Now, one last thing. You may have thought that you were going to escape this week without having to listen to me harp on WordPress.

Leo: Oh, well. We tried.

Steve: No, no. Almost made it.

Leo: Almost.

Steve: I'm about to ask you about Bobiverse, Leo. But first, I do promise to at least keep this brief. And so as not to introduce any of my own well-known bias, I'm going to simply quote verbatim from just the top of Bleeping Computer's coverage of a recently released report on WordPress security.

Bleeping Computer wrote, this is not me, this is them: "Patchstack, a leader in WordPress security and threat intelligence, has released a whitepaper to present the state of WordPress security in 2021, and the report paints a dire picture. More specifically, 2021 has seen a growth of 150% in the reported vulnerabilities compared to the previous year, while 29% of the critical flaws in WordPress plugins never received a security update. This is alarming considering that WordPress is the world's most popular content management system, used in 43.2% of all websites out there.

"Of all the reported flaws in 2021, only 0.58% were in WordPress core - 0.58%, that's probably one - with the rest being in themes and plugins for the platform, coming from various sources and different developers. Notably, 91.38% of these flaws are found in free plugins, whereas paid/premium WordPress add-ons only accounted for 8.62% of the total, reflecting better code vetting and testing procedures." And, I would argue, professional coders, rather than someone saying, "Hey, I just created an add-on. It's free. Here you go." And I say, good luck to you.

Leo: Yeah. Well, and again, it's not in the core, it's in these plugins. And just be very, very careful, I guess, about what plugins you install.

Steve: Yeah. And I would say, given that the vast majority are in the free plugins, if you want something...

Leo: Pay for it.

Steve: Find something that looks - yeah, pay for it, exactly.

Leo: Yeah.

Steve: Find something from a really reputable plugin provider.

Leo: We actually had Matt Mullenweg, the creator of WordPress, on FLOSS Weekly last week.

Steve: Oh, cool.

Leo: I didn't mention anything to him about this.

Steve: So I've been meaning for weeks, Leo, to ask you if you had continued listening to the Bobiverse.

Leo: I haven't even started. I have it. I bought it, and I haven't even, you know, I have such a backlog of books that I haven't gotten to it. And I keep thinking, oh, I've got to press play, press play. So how many books in are you?

Steve: Well, the initial three were the trilogy. And so we have the fourth book. And, you know, I was a little skeptical because the reviews that I read were a little mixed about it. Now I understand what's going on. I mean, it was a grumpy person who didn't want anything to change.

Leo: Ah.

Steve: Basically the first three books are a particular set of things, a style where we're looking at like this explosion of these - and I think I mentioned before, and this is not giving anything away because this is all - it's like on the jacket cover. It's that a Bob is a human intellect that was transferred to a computer to create what's known as a von Neumann probe which by definition is a probe which goes out into space, finds a new star system, harvests the raw materials with which to build more of itself.

Leo: Oh, boy.

Steve: So replicates and then sends copies of itself off in all directions, each with the same purpose.

Leo: What could possibly go wrong?

Steve: So thus the Bobiverse. We end up with a lot of them.

Leo: Yeah.

Steve: And so the original plotline of the first three jumps around among all of these different Bobs. And I think I mentioned when I talked about it before that I had a hard time keeping track of, okay, which Bob is this? And so I just gave up and just sort of kept reading into a new chapter. And I'd go, okay, it's that Bob. So what's happened in Book 4 is we've focused on one particular interesting thing which has been found. Now, there's this notion in sci-fi of megastructures. An example, a well-known megastructure is the Dyson sphere. Right?

Leo: Right.

Steve: Where a society gets so advanced, and in fact people who've like bothered to chart all this, they have like a nomenclature for different stages of advanced spacefaring civilizations. Well, at some particular stage, you become able to englobe your entire solar system in a Dyson sphere for the purpose of capturing the entire energy output of the sun, of your local star. And of course this actually was the plot for one of our very favorite pair of books that Peter Hamilton wrote, "Pandora's Star."

At the very beginning of the book - and again, this doesn't give anything away - Dudley, an astronomer, happens to be looking in a particular direction when a star winks off. And he's like, what? Because stars don't wink off. They can have all, you know, you can have a well-known star life cycle that cosmologists understand. But turning off is not something that stars do. So anyway, that was the beginning of the story is what could cause a star to wink off? Well, its Dyson sphere was turned on is what happened. So anyway, we need to hear about our last sponsor, and then we're going to go into WACs On or WACs Off.

Leo: Another wacky episode of Security Now!. All right, let's talk WACs On WACs Off.

Steve: Oh, my lord. Okay. So QWAC stands for Qualified Website Authentication Certificates.

Leo: Oh, of course.

Steve: Uh-huh. Let's have a good acronym; shall we?

Leo: Yup.

Steve: And it's just something that the European Union has proposed to adopt in an amendment to Article 45 of their Framework for Digital Identity, in this case the Digital Identity of websites, not people. What makes this way more interesting than watching paint dry, or watching the slowly grinding wheels of bureaucracy, is that the EU has been attempting to make this happen for the last six years, since 2016, and they still haven't given up. And after the most recent proposal by the EU, a group of 38 well-informed security and IT academics and professionals, including the EFF, cosigned an open letter to the EU regulators warning that the enactment of this protocol could expose Internet users to cybercrime.

The open letter is titled "Global website security ecosystem at risk from EU Digital Identity framework's new website authentication provisions." And the group starts off by saying: "Dear Honourable Member of the European Parliament, Dear Member of TELE Working Party. We the undersigned are cybersecurity researchers, advocates, and practitioners. We write to you in our individual capacities to raise grave concerns regarding certain provisions of the legislative proposal for a European Digital Identity framework, the eIDAS revision, and their impact on security on the web."

Now, the EU doesn't appear to be backing down about what they want, and we've seen what a mess can be created when something like the GDPR and its cookie regulations are imposed upon the world. So I was very interested to understand what the EU has gotten themselves up to now. They've intruded into our lives by making us agree to every website's cookie usage. So what's coming next?

I spent hours working to wrap my head around this. I've read the arguments being made by both sides and all the references I could find. I did find a very strong and beautifully written explanation of the fundamental problem with what the EU wants to do, which I believe was probably, well, I know it written at least by last July. I think it was written by Google's Ryan Sleevi. At least he had a tweet which pointed to it for more information. But again, it was written last July, and the EU still doesn't appear to get it.

What the EU wants is something more than TLS certificates. They want to have their own website authentication, essentially the ability to bind their own array of government validation and authentication assertions to websites, so that when an EU citizen is browsing the web and goes to a website, they want the user's browser to display a top-of-page banner stating that "This website uses a QWAC." I'm not kidding you. And then because they realize - I saw an example of this in one of their slides. Actually I have the slide, although it's so tiny in the show notes it's difficult to see. But it comes up in a banner at the top of the page, and then it says in parens, (Qualified Website Authentication Certificate). Wow. And then a list of a few privacy and security assertions in this banner which will have three tabs: a Summary tab, an ntQWAC Validation tab, and a TLS validation. And there will be an "I got it" button to dismiss the banner notification. Please, please don't make us do this.

Anyway, so what's going on here? How is this done? The website would serve some JavaScript. The website that the user is visiting would serve some JavaScript, who knows, perhaps the same JavaScript that now makes us all agree to be cookied. This JavaScript makes a call to an external web service which is tasked with producing a report which will be returned for display to the user.

The web service first retrieves another blob, known as the ntQWAC, from a well-known default path on the website that the user is visiting. And we've talked about well-known paths before; right? Where it's domain/well-known/something, where a client knows to ask for something from the server. So that blob contains a number of government-provided security and other assertions about the site and a hash of the website's TLS certificate.

Now, this is just a blob on a website. The web service needs to have the blob validated. So it queries an online validator in the next step to verify the authenticity of this particular ntQWAC blob. It then hashes the original website's TLS certificate and compares it to the hash it expects. So that binds the TLS certificate to it. Presumably that's to prevent certificate forgery, such as we have mentioned earlier in this show and also in that BGP routing hack the other day. Okay. So now it knows that it's associated with the proper TLS certificate. Next, it processes this ntQWAC attributes to produce a structured ntQWAC validation report, and it then performs an on-the-fly validation against additional authoritative sources, which are not specified, but they're shown in the diagram of how this all works.

And when all that's done, the web service returns the validation report to the waiting JavaScript, which then presents the banner to the EU citizen, who will have become quickly trained to click "Okay, I got it," just to get rid of the thing.

Leo: Exactly the problem.

Steve: Yes. It's just - it's ridiculous. Maybe they could combine them with the "Okay to Cookies" button so you don't now have to press two things, one thing to get rid of the authentication from the government and another button to get rid of the cookies which the government has made be posted.

So, what do we get in return for all this industry? We get what the EU apparently wants, which is its own governmentally controlled, separate from the TLS environment and community, governmentally controlled facility for determining, controlling and displaying various privacy and security assertions about individual websites. The EU essentially wants to add a layer of their own input - much as they have with cookies, thank you very much into the browsing experience of all of their citizenry. There are a number of problems with this. But as I said, these problems have been patiently, carefully, articulately, and repeatedly laid out and explained for the last six years, so far to no avail.

Now, okay. One biggie should stand out to everyone right off the bat. The web has been deliberately designed as a two-party system. The user with their browser and whatever website they are visiting are the two parties. It's me and you. Now, while it's true that in today's world this pure two-party system immediately collapses due to a website's voluntary and deliberate inclusion of third parties which it brings in, that's the site's choice to make. And we have been carefully chronicling all web browsers moving toward ever greater and stronger isolation of any and all third parties. They're being siloed. They're being given their own caches. This whole, you know, we're all together, one big happy cookie farm, that's all going away. So that controls tracking and profiling.

The larger problem to me is that - so there's that. There's the fact that this essentially codifies a formal third-party presence involved in presenting all of this. This web service, whatever it is, is like central headquarters for tracking of every user that goes to any of these sites. But the even bigger problem that occurred to me is what prevents a malicious website from simply displaying the same happy news QWAC banner and giving its intended victim the all clear?

Leo: Oh, man.

Steve: You know, if the system is that easily spoofed, then what's the point? I mean, the whole thing seems absolutely nuts. The only way to prevent that would be to force it into the Chrome, the browser UI Chrome, much as the certificate can be - you're able to view the certificate. And that's not the web page showing you that assertion, that's the browser's unalterable Chrome, its UI that lets you see the certificate that it received. So the only way you could prevent casual spoofing is if the EU forced the design of the browsers to somehow display this crap.

And, I mean, this is very - in much of what I read it's very much compared to the EV certs, where the idea was to show some additional information that would be useful to the user. And we know what happened to that. First they weren't green any longer, and then they were just gone completely from the user interface. So the EFF in their never boring and always over-the-top, sky-is-falling style, issued a press release on March 3rd. I have the thing in the show notes, but I'm not going to bore everybody with it. As I imagine reading this, I'm just thinking, okay, everyone knows. It's all, oh my god, this is going to completely destroy all of the website and all of our security, and experts are urging the EU to amend the revised Article 45 and so forth.

But what they talk about actually in this letter is different because they're talking about the enforcing of other certificate authorities, which is not the system that I just read. So I thought, what? Well, it turns out that that wasn't explained in the slide deck that I found, of which one of those slides is in the show notes. There is an entirely different system. The reason is there are actually three different proposed ways that this system could work. And the one the EFF is all steamed up about is the worst of the three, which may be why it wasn't put in the slides. It is a proposed single certificate solution. In that solution, various governments - various governments - would each have their own root certificates added to our operating systems and browser root stores. And these governments would be able to issue TLS website certificates containing all of that additional authentication information they desire. And this is the point in the podcast where in unison we all ask: What could possibly go wrong?

Leo: What could possibly go wrong?

Steve: Yes. So the Internet Society has published an Internet Impact Brief titled "Mandated Browser Root Certificates in the European Union's eIDAS Regulation on the Internet." And skipping past a bunch of the intro, I'll just share three paragraphs. They said: "The European Commission is currently" - and this is the Internet Society, like InternetSociety.org, so toned down from the EFF. They said: "The European Commission is currently evaluating this regulatory framework. It organized a public consultation in 2020 to collect feedback on drivers and barriers to the development and uptake of trust services and eID (electronic ID) in Europe. On this basis, the Commission has proposed an amendment to the 2014 regulation to try to stimulate adoption of the framework and the availability of secure eID schemes across the internal market.

"Among the changes in the new proposal are provisions around QWACs that could have the effect of forcing browsers to include TSPs" - those are Trusted Service Providers in the EU's parlance, we know them as Certificate Authorities - "authorized by national governments into their root stores, without guaranteeing security parity with current best practices. In other words, it would require browsers to add these CAs to the list of trusted middlemen, even if they do not meet industry standards or are recognized as a trusted party."

They said: "Although the text of the proposal may seem uncontentious, the high-level goals have unstated technical consequences which we believe are unpalatable, regardless of which option is put into practice." Then they said: "For more information about our interpretation of the proposed revisions, see Annex A." And I included Annex A in the show notes, but I'm not going to go over it because basically it's what we talked about.

So this ETSI's core assumption is that browsers will validate, in other words recognize, a QWAC by binding it to a TLS certificate for the domain it is intended to represent. Since TLS certificates themselves do not provide the level of authenticity eIDAS wants to achieve, for example, they might just be a domain validation, a DV certificate like Let's Encrypt produces, or like a non-EV, or might not even be organizational, OV, they say this means that QWACs must be issued according to a separate set of processes and criteria for eIDAS-compliant Trust Service Providers.

So at this point it is very unclear what's going to happen. And I wouldn't worry about it except that we're all now having to click on, yes, I agree about your g-d cookies for every website we visit.

Leo: So out of control.

Steve: It is.

Leo: Does QWACs solve a problem that exists? I mean, isn't TLS sufficient?

Steve: Yeah, they don't think so. They want...

Leo: What's the problem they want to solve? I don't understand.

Steve: They want browser certificates to contain additional verified like identity-ish things. Like an actual name and an actual street address and an actual, I mean, they want to tie it to identity. And so even that raises the hackles of a lot of people who are saying, wait a minute, we want the web to continue to be able to offer some level of anonymity to protect people. And EU is saying, well, digital identity is important, and we want to push this down into the web browser so people know more about the sites they're visiting.

Leo: Yeah, boy, I don't like this idea at all.

Steve: I know. And they're not giving up on it.

Leo: Wow. It doesn't solve anything, and it introduces all sorts of security problems. It breaks things. And the worst thing I can imagine, and now more than ever this should be obvious, is to have countries have their own certificate authorities that you have to adhere to. I mean, sorry, that's not - I'm sorry, that's terrible. Plus another banner. Just what we want. I already - there are many pages now that have such long cookie banners I literally can't ignore the banner. I have to click past it to get to the site.

Steve: Comes up, like half the page is there blocking you from doing anything.

Leo: Yeah. And by the way, Steve, correct me if I'm wrong, but don't they have to save a cookie to say that you've seen this banner and not show it again? Am I wrong? That's a cookie. So you can say I don't want any cookies, you're going to see that banner forever. The whole thing is just so broken and stupid.

Steve: Yeah.

Leo: Wow.

Steve: Yeah. And, you know, on the coattails of GDPR, which has been a mess for everyone.

Leo: Well, I'm in favor of some privacy protections, for sure. And I think GDPR, if it weren't for GDPR, there would be none. So okay. But the cookie thing is ridiculous. And this sounds not merely ridiculous, but actively dangerous, breaking. It's why you don't want politicians making security or technology decisions.

Steve: Well, okay. So if they wanted to do some legislation, why not require that the Do Not Track header is honored?

Leo: There you go. Adhere to that.

Steve: Exactly. Just do some legislation and let people say they don't want to be tracked. And if you see that, you cannot track them, and that's the law. Thank you very much. But no.

Leo: That would be easy. That would be simple. Instead we get this. I don't even - this is crazy. This is just - this is like somebody's fever dream.

Steve: And, exactly, what prevents other sites from spoofing it?

Leo: Yeah, that's easy to put that up.

Steve: Yeah. Just do a screenshot of one that you like and just show it.

Leo: Silly. Silly, silly, silly.

Steve: I wish it were not like such a potential problem. I mean, like it could actually happen.

Leo: Somebody in the chatroom is saying, and maybe this makes sense, it's about taxation. Maybe that's possible, you know, they don't want anybody evading their taxation.

Steve: Well, on that screen there was something about VAT. I think it was one of those authentication providers. There was some VAT thing.

Leo: Interesting.

Steve: So who knows. Anyway...

Leo: Once again, you know, if you didn't listen to this show, you wouldn't even know about this stuff. That's why you've got to listen.

Steve: Yeah, you could actually sleep at night.

Leo: Yeah. You'd dream the dream of babies. You'd be so happy.

Steve: The world has no problems. My computer is secure. I can go anywhere I want.

Leo: Everything is fine.

Steve: I can go anywhere I want to. I can click any link that I see. What could possibly go wrong? Nothing.

Leo: We do this show every Tuesday, right after MacBreak Weekly, 1:30 Pacific, 4:30 Eastern, 20:30 UTC. You could stop by and watch us do it live at TWiT.tv/live.

The IRC channel is always going, but of course it's most active when there's a live show on, and that's irc.twit.tv. Members of Club TWiT get their very own little Clubhouse, our Discord server, which isn't just for show conversations, conversations about everything. Stacey's Book Club is coming up; a Paul Thurrott Fireside Chat. We're going to talk to Patrick Delahanty on Thursday. He's our engineer. If you're curious about the back end, he's got a great back end.

Steve's got a copy of this show, actually two unique versions of the show. He's got a 16Kb audio file for the very bandwidth-impaired folks, plus Elaine Farris's amazing transcription so you can read along as you listen at GRC.com. While you're there, pick up a copy of SpinRite. That's Steve's bread and butter. That's what keeps the boat afloat. It's the world's best mass storage maintenance and recovery utility. SpinRite 6 is the current version; 6.1 is in progress, in process. And you can participate in the development as well as get a free copy if you buy right now: GRC.com.

The website for us, for the show, we have 64Kb audio, and we have video, is TWiT.tv/sn, TWiT.tv/sn. There's a YouTube channel dedicated to Security Now!. That makes it easy to share if you want your boss to see something or something like that, or your coworkers or colleagues or buddies. Just send a share link from YouTube, that's a good way to do it. You can also subscribe on your favorite podcast player and get it automatically every Tuesday evening, as soon as it's available. If you do that, please leave us a five-star review. Let the world know about this, the best, the one, the only Security Now!.

Steve: The original.

Leo: The original. Since 2005, baby. Now in its 18th, 17th year of edumacation. Thank you, my friend. Have a great week. We'll see you next time on Security Now!.

Steve: Right-o.