XSinator

Description: This week Tavis Ormandy finds a bug in Mozilla's NSS signature verification. We look at the horrifying lack of security in smartwatches for children (smartwatches for children?!?), and at the next six VPN services to be banned in Russia. Microsoft softens the glue between Windows 11 and Edge, bad guys find a new way of slipping malware into our machines, a botnet uses the bitcoin blockchain for backup communications, and HP has 150 printer models in dire need of firmware updates. We touch on sci-fi and SpinRite, then we look at new research into an entirely new class of cross-site privacy breaches affecting every web browser including a test every user can run for themselves on their various browsers.

High quality (64 kbps) mp3 audio file URL: http://media.GRC.com/sn/SN-848.mp3
Quarter size (16 kbps) mp3 audio file URL: http://media.GRC.com/sn/sn-848-lq.mp3

SHOW TEASE: It's time for Security Now!. Steve Gibson is here. We're going to talk sci-fi recommendations, smartwatches for children that are really super smart spies on your kids' wrists, bugs in Mozilla's signature verification, and a whole lot more. Then a new tool which has revealed a flaw in pretty much every browser on the market, XSinator. It's all coming up next on Security Now!.

Leo Laporte: This is Security Now! with Steve Gibson, Episode 848, recorded December 7th, 2021: XSinator.

It's time for Security Now!, the show you wait all week for. Here they are, the Security Now! guys. And Steve Gibson, welcome. It's good to see you, the guy, the man in charge of Security Now!.

Steve Gibson: Leo, great to be with you as we plow into December.

Leo: Plow. Plow, I tell you.

Steve: Plow.

Leo: Yes.

Steve: Yes. This is the first Tuesday of the month since the month began on Wednesday, last Wednesday. So this is, as I mentioned before, the latest in the month that we can

have Patch Tuesday. That'll be next Tuesday. So we'll wait and see what happens. The hope is that we're going to get printing back for Windows.

Leo: We've been hoping that for a while, haven't we.

Steve: It'd be a real Christmas present, yes.

Leo: Yeah. Oh, man.

Steve: Yes, Virginia...

Leo: Someday.

Steve: ...you can print.

Leo: Yes.

Steve: So a group of five German researchers took a look, a new look, a different kind of look, at a means by which a site we're visiting can work behind our backs using deliberately created features of JavaScript and the state of the art of web pages, the so-called Document Object Model, to infer privacy-violating things about us and our relationships to other websites. So today's topic is the site they created, XSinator - XS as in cross-site, X-S-I-N-A-T-O-R. And anybody who wants to jump ahead can go to XSinator, XSinator.com, and click the Test My Browser button. And within a minute or so you will see a whole bunch of red which are specific instances. This thing runs, I think it's 37 or 38. They start numbering at zero, bless their hearts. So I don't remember whether it's 37 or 35 and you add one for the zero-based counting. But you'll see all of the places where their test has verified that the browser you're using can fall to some of these tests. We're going to talk about that in some detail at the end.

But Tavis Ormandy has been busy. He found a bug, a bad bug in Mozilla's NSS, that's their own, I assume it stands for Netscape Privacy or Netscape Security Suite because it's always been part of the Firefox and then they split it out, a very nice and actually widely used open source security library. He found a problem, a bad problem, in their signature verification.

We're also going to look at the horrifying lack of security in something that I didn't know existed, Leo, smartwatches for children.

Leo: Oh, yeah. Oh, yeah.

Steve: And oh, bad idea.

Leo: It's not just smartwatches. We were talking about this on MacBreak Weekly, the Life 360 app that you put on your phone to track your kids and your family members. Just a horrific privacy nightmare.

Steve: Well, and just wait till you hear how bad these things are. We also have another six VPN services have been banned by Russia, and we get to say Roskomnadzor many times. We also have Microsoft happily softening the glue between Windows 11 and Edge, so we have a bit of good news there. But bad guys have found a new way of slipping malware into our machines. There's a botnet using the bitcoin blockchain as its backup communications medium. We have HP with more than 150 printer models in dire need of firmware updates, doubtless affecting some of our listeners.

I'm going to then briefly touch on a sci-fi note and update our listeners about SpinRite. An interesting thing will be happening, maybe not tonight, but tomorrow for sure, which is to say as soon as I'm done with the podcast and have recovered. And then we're going to look at this new research into an entirely new class of cross-site privacy breaches which affect every browser, mobile and desktop. And as I said, a test everyone can run. And I got a picture that was posted in the SpinRite development newsgroup yesterday that I immediately knew I had to share. So Lorrie looked at it and said, "Is that yours?" It's like, "No, honey, that's not. But it's a dedicated tester."

Leo: Let me queue up the Picture of the Week.

Steve: Okay. So this is not something that we expect all of our testers to do. Let me just make that clear. Most people just boot a machine they have with a USB thumb drive and run the development release to see how it does and then report their results. This individual is clearly a technician of some sort. If you look, Leo, if you scroll down to the top of the next page, you'll see the SpinRite screen that he was seeing.

Leo: Look at all the different devices.

Steve: Yeah. So he's got 13 drives hooked up to this system. And there's little extra goodies that users will never see which are here for development. You can see along the top it says 80, 81, 82, 83 up through 8C. Those are the BIOS designations. And they're white because SpinRite was able to figure out which one of the drives that it has hardware access to the BIOS is referring to. There's no, unfortunately, in the BIOS there's no way to associate BIOS designations with physical drive designations. Yet SpinRite has to so that it's able not to confuse users by showing them a drive that the BIOS sees and that it sees.

So one of the challenges has been to create this association between the BIOS designations and the drives that SpinRite has direct hardware access to. So you can see, for example, down at the bottom is 80, the fourth from the bottom. That would be the drive that he booted from, a little 264MB thumb drive. And then there's 8A, 8B, and 8C, which are other drives which are accessible only to the BIOS and not to SpinRite. But anyway, oh, and then also there's a little cyan column down the right-hand edge of the drive identity. That shows how many seconds it took for SpinRite to identify and qualify that drive for operation.

It performs a whole bunch of confidence testing, you know, gets the name, the serial number, all the smart data. It also then performs a series of reads and writes in order to verify that it's able to do so. When we were having IRQ problems, some of those tests would take a long time. And so they'd sometimes say, you know, 26 seconds or something. So that was another little clue that would allow us to know what was going on.

But in that first picture we see a little, looks like a Mini-ITX form factor motherboard, and drives just spread all over the table. He actually numbered them. You can see a little yellow column of numbers on that first image labeled 1 through 13. And then he drew lines to show where all of these 13 drives are that are scattered around the table. Anyway, it was just so unusual that I wanted to share it with our listeners.

Leo: I really like this drive with the massive heat sink on it.

Steve: He made some, yeah, above and below it. You can see it sandwiched between heat sinks, and he made some comment saying this used to run hot, but apparently...

Leo: Not anymore.

Steve: Apparently not anymore.

Leo: No, no, no. Do-it-yourself heat sinks, I love it.

Steve: Just a fun little one-of-a-kind...

Leo: What a crazy test bench. Look at that. Wow.

Steve: Yeah, and SpinRite found all the drives and is working correctly on all of those.

Leo: Nice, nice.

Steve: So we're getting there. I'll talk a little bit about the past week's adventure in a minute. But Tavis found a bad bug in NSS. Of course this is Tavis Ormandy at Google. He's part of their zero-day group [Project Zero]. He discovered and quietly reported an important bug in Mozilla's widely used open-source NSS security suite. Last Wednesday, with the bug quickly patched, Tavis tweeted, he said: "This is a major memory corruption flaw in NSS. Almost any use of NSS is affected. The Mozilla advisory is here." And he provided a link. And then actually there's two links. Twitter shortened one of them.

Anyway, NSS is used by Mozilla, of course. It's the reason, for example, that Firefox sometimes behaves differently on Windows than other browsers because every other browser uses Windows' intrinsic security library that's built into Windows, whereas Firefox brings along its own. And that's sort of a mixed blessing. It means there's a lot more for them to maintain, but it does give them sort of more of their own containment and more portability.

But aside from just Mozilla, Red Hat, SUSE, and many other products also use NSS. Of course Firefox, but also Thunderbird, SeaMonkey, and the Firefox OS. There's the open-source client applications. Evolution, Pidgin, Apache OpenOffice, and LibreOffice are also NSS clients. Server products, as I mentioned, Red Hat, their Directory Server, their Certificate System, and the mod_nss SSL module for Apache is also NSS-based. Over on the server side, Oracle is using NSS, including their Communications Messaging Server and Oracle Directory Server Enterprise Edition. And as I mentioned, SUSE, their Linux

Enterprise Server supports NSS, as does Apache. So it's important, if there's something bad there.

In other locations, that is, other than Twitter, Tavis commented: "If you're a vendor that distributes NSS in your products, you will most likely need to update or backport the patch. Mozilla plans to produce a thorough list of affected APIs, but the summary is that any standard use of NSS is affected. The bug is simple to reproduce and affects multiple platforms." So in other words, anyone using NSS should focus less on worrying about which specific APIs are vulnerable. I would just argue don't worry about that. Just update your use of the security suite. NSS versions prior to 3.73 or 3.68.1 in the ESR are vulnerable. And that's all versions since October of 2012 until those releases.

Tavis said: "We believe all versions of NSS since 3.14, which was released October 2012, are vulnerable." However, for what it's worth, according to Mozilla, this vulnerability does not impact Firefox for whatever reason. But all PDF viewers and email clients which use NSS for their signature verification are believed to be impacted. That's where the problem is. It's in the signature verification component.

Mozilla said: "Applications using NSS for handling signatures encoded with CMS, S/MIME, PKCS 7, or PKCS 12 are likely to be impacted. Applications using NSS for certificate validation or other TLS, X.509, OCSP, or CRL functionality may be impacted, depending upon how they configure NSS." So just for the record, the exploitation of this during signature verification can lead to a heap-based buffer overflow when handling DER-encoded DSA or RSA-PSS signatures in email clients and PDF viewers. The impact of successful heap overflow exploitation can range from a simple program crash to potentially arbitrary code execution, up through bypassing security software if code execution is achieved. So again, if anyone is using NSS for some project, be sure to get an updated build. So that's all of the past nine years of NSS, up until this was patched. So essentially I'm sure anything anybody has been using until very recently when they just fixed it.

Okay. I didn't know - you did, Leo. But I didn't know that there was a thing such as cheap smartwatches for kids and babies. To which we would have the refrain, what could possibly go wrong? Dr.Web, which is an AV security firm based in Russia, recently examined four inexpensive smartwatches designed and marketed specifically for kids. Now, before I go any further, allow me to just suggest to anyone listening to this podcast that, if they have that on their Christmas list, avoid any temptation whatsoever you might have to purchase a $50 smartwatch for anyone you know.

Dr.Web's report is very detailed, so I'm not going to bother to describe what they found for each of the four devices. But discussing a representative example will be informative and should be sufficient to forestall any desire anyone might have to put one of these little spyware beasties onto the wrist of anyone they know or care about.

So Dr.Web introduces the subject with a little bit of background which I've shortened a bit. They wrote: "Parents always strive to take care of their children. Technology innovations help them reach this goal through various wearables like smartwatches and GPS trackers. These devices are getting close to smartphones in functionality. For example, many of them can track the child's location and travel route. These devices can also make and answer phone and video calls, receive SMS and voicemail, take pictures, and listen to their surroundings. It's even possible to enable the remote control of the device." Okay, I don't know what that means. But "Dr.Web has analyzed the potential threats that such gadgets can pose to parents and their children.

"During their daily operation," they wrote, "these devices collect and transmit data to the manufacturer's servers and make it available to parents through their personal accounts.

The information obtained with smartwatches is very sensitive. If malicious actors get their hands over such information, it can put children in great danger.

"To understand the vulnerability and dangers of children's smartwatches, Dr.Web's specialists analyzed several popular models chosen based upon public popularity ratings and purposefully selected models from different price ranges. We purchased all smartwatches anonymously from an online store. We carried out both static and dynamic analysis during the inspection. We searched for potential implants in the software and possible undocumented features, as well as checked network activity, transmitted data, and how it was secured."

Okay. So one of the four devices they examined was the ELARI KidPhone 4G Smartwatch. Yes. And isn't it pretty, Leo. I mean, it's beautiful.

Leo: Just like Mommy and Daddy's.

Steve: Exactly. Okay. Get a load of that, this beautiful-looking device. "Several versions," they wrote, "of the ELARI KidPhone 4G watches exist. They're based on different hardware platforms, but they all run an Android OS, and their firmware differs slightly. The primary threat of this model comes from its installed software. Its firmware has a built-in app for over-the-air updating, and this app has trojan functionality.

"Firstly," they wrote, "the application sends the child's geolocation data to its own remote server. Secondly, we found malicious code inside it, which is detected by Dr.Web" - remember these guys are AV people. It's detected by their antiviral, antimalware technology as "Android.DownLoader.3894. Every time the watch turns on or network connection changes, this code launches two malicious modules, Android.DownLoader.812.origin and Android.DownLoader.1049. origin. They connect to the command-and-control server to transmit various information and to receive commands. By default, the modules connect to the server once every eight hours. Because of this, a long delay exists between the first connection and the first time the watch is turned on, thus reducing the chance of discovery.

"The Android.DownLoader.812.origin module sends the user's mobile phone number and SIM card information, geolocation data, and information about the device itself to the command-and-control server. In response, it can receive commands to change the frequency of subsequent requests to the server; update the module; download, install, run, and uninstall apps; and load web pages. The Android.DownLoader.1049.origin module sends SIM card information and mobile phone number information, geolocation data, a large number of data about the device and installed apps, as well as the info about the number of SMS, phone calls, and contacts in the phone book to this command-and-control server. Thus Android.DownLoader.3894 hidden in this watch - that's that parent - can be used for cyber espionage, displaying ads, and installing unwanted or even malicious apps."

Okay. Now you put the picture on the screen before. I mean, which for me it's sort of disarming. It's a consumer product. And it's adorable and appealing-looking. Our listeners who don't have video can't see what you and I are seeing, Leo, but they're very professional-looking, attractive consumer devices. They look like toys, but they're definitely not. Anyone who purchases one of these little devils is not only exposing their children to remote anonymous attackers and trackers, but also bringing an open microphone into the lives of those children's parents. Who knows what might be overheard in what's presumed to be a private setting? And who would be to blame?

Now, I fully recognize that the likelihood of any specific child or their parents being targeted through this is diminishingly low. I don't mean to suggest otherwise. But knowing what we now know, who would feel comfortable strapping one of those loose cannons onto their child's wrist? And that's my point. The photo of these devices is utterly at odds with what's going on inside them. It's deeply unsettling because they look so cute and harmless. Yet inside that colorful soft plastic shell is a communicating computer running a trojanized Android OS which autonomously connects to a remote command-and-control server.

And Amazon sells these things. The bullet points on Amazon's page says, under "Security and Peace of Mind for Parents: Monitor child's location, set safe zones, and use remote audio monitoring through free parent-controlled app. Get alert if the child uses the built-in SOS button."

Under "Easy Communication: Stay in touch. Have a 2-way voice call. Use voice chat or send emojis. Limit contacts to those authorized through the app to avoid undesired communication." Under "User Friendly and Comfortable: Kids phone watch features math game and interface with emoticons." And then, finally, under "Other features: Class time mode, alarm call, pedometer, up to 4 days standby time, ELARI SafeFamily free multilingual Android/iOS-compatible App, ability to add other ELARI smartwatch users to friends list."

So the watches that the Dr.Web researchers examined had default passwords of 123456, and most of them don't bother to employ encryption in their communications. They just use plaintext. What this brings to mind is the glaring gap in consumer protection that currently exists in our tech industry because the difference between purchasing a well-designed WiFi router that might still have an inadvertent security vulnerability and this colorful plastic child's watch that communicates without encryption with remotely located command-and-control servers, continually sending back all of the child's movements, and is able to autonomously download any additional software at any time to me seems extreme. I mean, it takes the consumerness way down to sort of a different lower level.

And who knows what they're doing? In the U.S., the engineers and scientists at Underwriters Labs are able to test toasters and vacuum cleaners because they're able to carefully examine and stress test the important functional safety aspects of those consumer products. But today's tech gadgets are closed black boxes which are actively hostile to reverse engineering. I don't see any solution to this mess other than to take every possible precaution. We're able to place our IoT devices on their own segmented network. If these wristwatches also have WiFi radios, placing them on the IoT WiFi would help. But they can still spy.

Leo: Well, yeah, and the whole point of this is to follow your kid when they're out and about, not in the home network.

Steve: Right. Right, exactly. And so to me I just - I would stay as far away from this ELARI brand as possible. And I guess the only advice I would have, I mean, I could imagine, as you listen to these bullet points, these seem like cool things.

Leo: Yeah, with the kind of features you'd want in a watch that's tracking your child.

Steve: Yeah, but use Garmin, or use...

Leo: Yeah, use an American company to begin with, I would say, yeah.

Steve: Yes.

Leo: Apple has watches that do all of this, probably at a much higher price. But they're available.

Steve: Yeah. I just got a chill when I looked at these beautiful-looking consumer devices for 50 bucks. And imagine - and we know how much kids like to emulate their parents; right? So I'm sure Johnny would love to have a watch like Daddy has.

Leo: Right. Well, Daddy may even want Johnny to have these capabilities. That's the thing. Daddy and Mommy want to track the kids.

Steve: Right.

Leo: The problem is who else is doing it?

Steve: Right.

Leo: You're going to turn on all those features because that's why you bought the watch.

Steve: Is there a really low-end Apple iPhone?

Leo: No. There's an Apple Watch SE that they make that you can configure for kids, and that's probably the way to go. But it's a few hundred bucks. It's not cheap cheap.

Steve: Yeah. Again, to all of our listeners, buyer beware. If you want this, I'd go Apple.

Leo: Yeah, I agree.

Steve: We know that it will be as well-designed as it can be. I just don't know how you do, I mean, maybe Samsung. Maybe another, some other high-end Android device. Or what about any of the Garmin-y sorts of things?

Leo: Yeah, yeah.

Steve: Do any of the activity trackers do locations?

Leo: Problem is there a lot of kids' trackers in this category because they're inexpensive. You don't want to give a kid a $300 watch.

Steve: No.

Leo: They're small. They're built for kids. Their functionality is considerably more limited because parents don't want kids making random phone calls, either.

Steve: Right.

Leo: So there are - this is a...

Steve: Look at all the spam we get in our phones. You don't want your kid answering the phone, like some random stranger in lord only knows where.

Leo: Exactly. "Hello?" "Hey, how's your auto warranty?" "What?" Yeah, that could be a problem.

Steve: Anyway, I just - for me this was a wakeup call, and I wanted to share it because, boy. However bad you think it could possibly be, this suggests it's a lot worse than that. The people selling this just don't give a crap about security or privacy, and you don't want to strap that on to any child's wrist under any circumstances.

Last Thursday, speaking of Russia, Russia's Internet policing agency Roskomnadzor announced the ban of an additional six VPN services, bringing now the total number of banned VPN providers to 15. The most recent six bannings were Betternet, Lantern, X-VPN, Cloudflare WARP, Tachyon VPN, and Private Tunnel. Roskomnadzor sent a request to inform the Center for Monitoring and Control of the Public Communications Network about the removal of those additional six services from the systems of all registered Russian companies and public organizations. They can no longer be used.

So the list of banned services now reads: Hola VPN, ExpressVPN, KeepSolid VPN Unlimited, Nord VPN, Speedify VPN, IPVanish VPN, VyprVPN, Opera VPN, and ProtonVPN, to which we add those other six. These services collectively have all been banned due to their principled refusal to abide by Roskomnadzor's demands to connect their systems to the FGIS database because doing so would defeat the entire purpose of using a VPN connection, which is - that purpose is to bypass access restrictions and obtain anonymity, which is a level of individual freedom which apparently makes Russia's leaders uncomfortable.

Back in 2019 Russian authorities gave all VPN vendors operating in-country the ultimatum to comply with their rules, or else be banned. The only vendor who responded positively before the deadline was, I guess not surprisingly, the Moscow-based Kaspersky with their Secure Connection product. They were already the VPN in-country, so they had nothing, you know, they didn't have to change anything. The non-complying VPN vendors either established new servers just outside the Russian borders or attempted to employ traffic masking techniques which are known to work well against the Great Firewall of China. But the Russian authorities gradually caught up with those services also which were trying to bypass the new regulation and have banned them in multiple action rounds.

As Russian users uninstall banned products and turn to what few remaining products remain, Russian authorities evaluate which ones have then emerged as the most popular and add them to the growing block list. So if you're in Russia, good luck. It's going to be very difficult for you to have private and secure communications. Or you aren't going to be able to use one of these mainstream high-profile VPNs. I'm sure there are ways around this that people inside of Russia are aware of. And Leo, I'm aware of my need to take a sip.

Leo: Okay. You like our holiday dcor, by the way? It's all...

Steve: Oh, it's very nice, yes. I like it. You've got a sock on the door behind you, a stocking, I think.

Leo: Yes. There's a stocking. There's a couple of stockings.

Steve: And we have some poinsettias, I see, very nice.

Leo: Yes. And, yeah, one of the stockings is for Ozzie. One of them is an Angry Bird stocking. You'll notice that the Mr. Robot mask has an elf hat on. Very festive.

Steve: Yes, very nice. Very festive. I think you should keep that. That looks good.

Leo: It looks good on him, doesn't it.

Steve: Yeah. That ought to...

Leo: No, I actually like this time of year. I look forward to it all year long. You'll be getting your TWiT Christmas card soon with - it's the meta version of our Christmas card.

Steve: You've been talking about it. I'm looking forward to seeing it.

Leo: It's going to be - I think you're going to enjoy it. We're all virtual.

Steve: Cool.

Leo: Back to you, Steve.

Steve: So Windows 11 has loosened its grip on Edge.

Leo: Oh, thank god.

Steve: Yes. I know.

Leo: Still doesn't convince me.

Steve: I know. Last week's Windows Weekly was a bit depressing listening to Paul and Mary Jo holding Microsoft to account for many of the choices Microsoft has made for the behavior under Windows 11. Among other things, Microsoft is clearly turning Edge from its original clean user-benefiting browser into another Microsoft profit center. And then adding insult to injury is having Windows 11 actively fighting its users' desire to switch away from Edge to any other browser.

Now, in fairness, we've probably all seen Chrome promoting itself whenever it has the chance. At least I do. Firefox is still the browser that handles my URLs by default, and I'll manually often launch Chrome and use it. Oftentimes Chrome notices that it's not the default and gives me some grief for that fact. But it's like, sorry, I like Firefox's down the left-hand column vertically oriented tabs much better than - and I've used Edge's too. I turn that on. And still Firefox is my go-to browser.

And it's annoying that there's no obvious way to tell Chrome that things are already just the way I want them to be, thank you very much. But now that Microsoft is pushing the adoption of Edge quite hard, we're seeing the same from them. BleepingComputer located the key in the Windows registry which contains the contents of the various pop-up messages that Microsoft has prepared for Edge. There are five currently. First one that may pop up reads "Microsoft Edge runs on the same technology as Chrome, with the added trust of Microsoft. Browse securely now." Okay. I don't know what that means.

Leo: Yeah, mm-hmm.

Steve: Or get this one: "That browser is so 2008! Do you know what's new? Microsoft Edge."

Leo: Ooh, I want to be with the hip kids. Woohoo.

Steve: "Come to the future."

Leo: Please.

Steve: Okay. Number three: "Looking for speed and reliability? Microsoft Edge is the only browser optimized for Windows 10." What? "Try a faster browser today." Okay. How about number four? "'I hate saving money,' said no one ever."

Leo: That's because they're building in buy now, pay later and type one searches.

Steve: I know. I know.

Leo: All this crap.

Steve: "'I hate saving money,' said no one ever. Microsoft Edge is the best browser for online shopping. Shop smarter now." Oh, Leo. And finally: "Microsoft Edge is fast, secure, and the browser recommended by Microsoft."

Leo: Hello.

Steve: "Discover more benefits. Learn more about Microsoft Edge." Now, BleepingComputer has been unable to trigger some of - I think they've only been able to see two of these five. But those are in there, waiting to spring out.

Leo: It's so annoying because it gets triggered when you try to download Chrome, when you go to Chrome, when you, I mean, anything you do, Microsoft feels the necessity to nudge you and say, oh, no, no, no. You want Edge.

Steve: Yeah, now, there are therapists that could help with this form of insecurity.

Leo: Yes.

Steve: Okay. In any event, I've already very clearly articulated my feelings about Windows 11, and I'm feeling quite confident that none of my hardware will run on it. So I'm good. It's like, sorry, you can't run Windows 11. Fine. That's the right outcome for me. Anyway, I've been shying away from any further kicking of that dog.

But in fairness I wanted to mention a bit of balancing good news that I imagine Paul will also be noting tomorrow: The way things were originally set up in Windows 11 is that Microsoft had eliminated the simple ability of the user to change their system's default browser by removing the UI option to do so. Instead, they granularized the web browser's handling settings by file type - like .htm, .html, .pdf, and so on - and by protocol - HTTP, HTTPS. And so you had to specify which browser would handle all of those things individually. And they buried all of those options deep in the registry. This forced any determined Windows 11 users to search through the registry, extension by extension and protocol by protocol, to manually change each one in order to unhook Edge.

Happily, with last week's release of the Windows 11 Insider build 22509, Microsoft has restored the presence of a single "Set Default" button, and has also surfaced all of those individual file and protocol types in the UI. So a non-Edge browser can now be set as default easily. And then, if desired for some reason, other browsers can override that default for specific file type and protocols. So good on Microsoft for listening to users. I mean, they'd have to be really deaf not to have heard the outcry over how difficult they had made it not to use Edge under Windows 11. So I just wanted to acknowledge that they had.

Okay. Also on the Microsoft side, the earliest Windows had the clean and simple Notepad app which was useful for displaying and editing unformatted and unformattable text files. I still use Notepad every day. It's just right there. No B.S. When I want to take line wraps out of a block of text, Notepad. And I like Notepad++, just to prevent everyone from telling me about it. I know about it. I have it. I love it. It's annoying that it's constantly updating itself, which is like, would those guys just please leave it alone? But no. Anyway, Notepad is there. It was implemented as a very lightweight UI around a

simple Windows textbox container. Basically it's a control, a UI control widget with a little bit of load and save UI around it.

But Windows also offered WordPad, right from the beginning, which allowed for a richer textual formatting experience. It was possible to set the text color, the size, and treatment, such as bolding and italics and so forth, and the text alignment. And just as Notepad was a UI wrapped around the Windows textbox container, WordPad's richer textual formatting experience was a UI wrapped around the Windows Rich Text Format (RTF) container. I've long been a fan of Windows' RTF control, and anyone who's ever used any of my Windows software will have seen its embedded pages with some text formatting. That's what I use. I like being able to set some colors and alignment and font sizes and things. All of that in my apps are RTF controls.

Okay. In any event, unlike Notepad, the RTF container can also contain embedded things. It can be a host for OLE, the original Object Linking and Embedding objects, because why not? One of the many features of this embedding is the use of so-called formatting templates which can be loaded on the fly from a file. And wouldn't you know it, malware deployers have figured out that these RTF embeddable template objects will also accept URLs. Oh, lord. In other words, not just loading them with something local, loading them with something from anywhere in the world. Sounds like a terrifically flexible feature. What could possibly go wrong?

Well, how about three different state-sponsored threat actors aligned with China, India, and Russia, all now having been observed adopting this RTF template injection, as it's now being called, as part of their new phishing campaigns to deliver malware into targeted systems.

Researchers at Proofpoint have published the results of their research under the title "Injection Is the New Black: Novel RTF Template Injection Technique Poised for Widespread Adoption Beyond APT Actors." They said: "Proofpoint threat researchers have observed the adoption of a novel and easily implemented phishing technique by APT [Advanced Persistent Threat] actors in Q2 and Q3 of 2021. This technique, referred to as RTF template injection, leverages the legitimate RTF template functionality. It subverts the plain text document formatting properties of an RTF file and allows the retrieval of a URL resource instead of a file resource via an RTF's template control word capability. This enables a threat actor to replace a legitimate file destination with a URL from which a remote payload may be retrieved.

"The sample RTF template injection files analyzed for this publication currently have a lower detection rate by public AV engines when compared to the well-known Office-based template injection technique." In other words, this has been adopted because it slips under the radar at this point. "Proofpoint has identified distinct phishing campaigns utilizing the template which have been attributed to a diverse set of Advanced Persistent Threat actors in the wild. While this technique appears to be making the rounds among APT actors in several nations, Proofpoint assesses with moderate confidence, based on the recent rise in its usage and the triviality of its implementation, that it could soon be adopted by other cybercriminals, as well."

They wrote: "RTF template injection is a simple technique in which an RTF file containing decoy content can be altered to allow for the retrieval of content hosted at an external URL upon opening an RTF file. By altering an RTF file's document formatting properties, specifically the document formatting control word '*\template' structure, actors can weaponize an RTF file to retrieve remote content by specifying a URL resource instead of an accessible local file resource destination." And they go on at some length to explain this more clearly.

So anyway, we know the tune pretty well. Once again, an arguably unnecessary and seldom, if ever, used feature of RTF containers has been found to be exploitable by attackers and is now being used with growing popularity. As Proofpoint suggests, it's too easy to use and too powerful not to become much more widely adopted. I guess what we could hope is that all of the AV scanners and Windows Defender and other tools which are currently not seeing this abuse will be quickly updated in order to catch it.

We have an instance of a malicious botnet using the bitcoin blockchain. Get a load of this one. A botnet which Google says has infected more than one million Windows machines globally and continues to infect new machines at a rate of thousands more per day is using the public bitcoin blockchain to stay in touch.

Google is suing a pair of Russian individuals it claims are behind this sophisticated botnet operation. In a complaint filed in the U.S. District Court for the Southern District of New York, Google names Russian nationals Dmitry Starovikov and Alexander Filippov as the two main operators of the Glupteba botnet, citing Gmail and Google Workspace accounts they allegedly created to help them operate the criminal enterprise.

Google claims the defendants used the botnet network - which it describes as a "modern, borderless technological embodiment of organized crime" - for illicit purposes, including the theft and unauthorized use of Google users' logins and account information. Google is demanding that Dmitry and Alexander pay damages and are permanently banned from using Google services.

According to Google, the Glupteba [G-L-U-P-T-E-B-A], I guess that's Glupteba botnet...

Leo: Glupteba, think you got it, yup.

Steve: ...which Google has been tracking since last year in 2020, has so far infected approximately one million Windows machines worldwide and is growing at a rate, as I said, of a few thousand new machines a day. Once a device has been infected, typically by tricking users into downloading malware via third party "free download" sites, the botnet steals user credentials and data, secretly mines cryptocurrencies, and sets up proxies to funnel other people's Internet traffic through the infected machines and routers. In its complaint Google says: "At any moment, the power of the Glupteba botnet could be used in a powerful ransomware attack or distributed denial of service." Google also noted that the Glupteba botnet stands out compared to conventional botnets due to its technical sophistication and use of blockchain technology to protect itself from disruption.

As well as launching the litigation against the so-called Glupteba botnet, the company's Threat Analysis Group, that TAG team, which has observed the botnet targeting victims in the U.S., India, Brazil, Vietnam, and South Asia, announced it has worked with Internet hosting providers to disrupt the botnet's key command-and-control infrastructure. This means its operators no longer have control of the botnet, though Google has warned that Glupteba could return due to the fact it uses blockchain technology as a resiliency mechanism.

Google explained that "The Glupteba botnet does not rely solely on predetermined web domains to ensure its survival. Instead, when the botnet's C&C server is interrupted, the Glupteba malware is hard-coded to search the public bitcoin blockchain for transactions involving three specific bitcoin addresses that are controlled by the Glupteba enterprise. Thus the Glupteba botnet cannot be eradicated entirely without neutralizing its blockchain-based infrastructure."

Which I thought was quite clever. So presumably the bad guys, if they lose control of the command-and-control servers, they're able to create transactions which post to the blockchain. The blockchain, of course, records the signatures of the entities that are posting transactions there. The bots then look there and pick up their instructions from details of the transaction. So where there's a will, there's a way. And obviously unfortunately there's a lot of will here.

HP has been shipping vulnerable printers for the past eight years. The guys over at F-Secure carefully examined one HP printer. It was an M725z multifunction device, MFP.

But once they reported their findings to HP at the end of April, HP found that their use of common firmware across printers meant that at least 150 other printer models were also affected across the LaserJet, PageWide, and ScanJet families. Updated firmware has been available from HP since November 1st, so anyone having any HP printer, who might be a target of either inside or outside attack, will definitely wish to update all of their HP printers' firmware.

The first of the two problems, tracked as CVE-2021-39238 with a CVSS rating of 9.3, describes a vulnerability that can be used to create wormable exploits that can self-replicate and spread to other HP printers inside internal networks or over the Internet. The trouble is particularly worrisome because it's a buffer overflow in the printer's font parser. So just causing a vulnerable printer to print a specially crafted page or PDF could take over the printer. The F-Secure researchers said that the flaw can be exploited to gain control over a printer's firmware to steal data or assemble devices into botnets, all while leaving little evidence of exploitation behind, presumably just operating in RAM.

They also indicated that attacks that abuse the vulnerability in various other ways, such as attacking Internet-exposed devices or by loading the exploit code on a website or inside an ad, can also be successful. Users on corporate networks, or those who view the ad, will have the code reach out to ports on their internal network to exploit local printers. Wow.

The second vulnerability is a local physical USB port attack being tracked as CVE-2021-39237, which impacts the printer's communications board. Fortunately, this bug can only be exploited with physical access to a vulnerable device, and an attack takes up to five minutes to execute, compared to the first one, which only takes a few seconds. On the other hand, if you've got printers with exposed USB ports, that's an inside job attack. In a very large corporation where you've got a disgruntled employee, or maybe they've been subverted by an outside entity, you know, hey, we'll pay you $10,000 if you insert this USB stick into one of your company's printers and walk away, wipe off any fingerprints, that would be a model that could also be successful.

So as I said, if your enterprise uses HP printers, which are still the world's number one printer by market share, do make some time to update the firmware throughout your organization. Again, it's been available since November.

Okay. Just a quick touch on sci-fi. My nephew is going nuts over Ryk Brown's latest novel.

Leo: How old is your nephew? Is he 10? Is he 30?

Steve: No, no, no, he's one of the top performers at Salesforce.

Leo: Okay. Got it. Okay. So he's a grownup. Well, that makes a difference. If a 10-year-old says they love Ryk Brown, okay, I'm going to take that like a 10 year old would like it. But he's a grownup, yeah.

Steve: Okay. I'm more than twice his age.

Leo: Okay.

Steve: And I can't wait. Anyway, he keeps texting me in all caps that the new novel is SO GOOD.

Leo: Now, we've read Ryk Brown before. This is not a new name.

Steve: Oh, yeah, yeah, yeah. Yes, yes, yes. It's spelled R-Y-K. This is that Frontiers Saga.

Leo: Is this Frontiers? Oh, it is. Oh, okay.

Steve: Yes. This is the 31st book. And I know that John is up to speed. He's probably already read this 31st book. Anyway, I will get there. My nephew is making me drool for this book. But I'm going to go do the Bobiverse first in honor of all of our listeners who have said, Steve, you know, check it out.

Leo: Frontiers, the Frontiers Saga is kind of one of those space battle-type sci-fi books; right? A lot of laser fire and things.

Steve: It is. But what it is is it's character driven.

Leo: Oh, good. Okay.

Steve: It is so well, I mean, in fact I have here in my show notes, I said: "Ryk Brown's latest novel, which presumably details the continuing adventures of Nathan, Cameron, Jessica, Vladimir, Telles, Josh, Loki and all the rest of the wonderfully articulated characters that Ryk has created." I mean, that's really what it is. It's so well done because you really get to know these people. And it's an interesting, you know, the best sci-fi is about the characters, but it also creates some interesting and engaging new technology.

So, and this isn't giving anything away. This is set in the far future, after the Earth and a number of its neighboring settled planets are recovering from the so-called Bio-Digital Plague. And that's never - Ryk never goes into great detail about it, but it's something, it was some sort of a digital virus that crossed over into the biosphere and almost wiped out humanity, not only on Earth, but everywhere it was able to travel. And a bunch of colony ships went out that were ahead of this bio-digital plague, and so humanity dispersed through the universe. So now we're in the process of getting ourselves, pulling ourselves back up by our bootstraps.

What was discovered in the Swiss Alps was an archive of all the stored human knowledge from pre-plague time which was locked up so the bio-digital plague couldn't get in. And so a new drive technology called the so-called "jump drive" allows us to jump a great distance instantly. And an accident which occurs at the very start of the first book, which is why this is not a spoiler, has surprising consequences.

And anyway, I was glad to see some reference elsewhere to it being one of the most popular science fiction space operas of all time. So it's not just me. I didn't have sort of calibration. But it is just - and I've read the whole 30 books twice now. And I'm finishing up 30, then I'm going to switch to the Bobiverse. But it's just, it's not going to change you. But if you just enjoy really well-done adventures, it is definitely that.

Leo: Nice.

Steve: Okay. So that heuristic hardware interrupt handling solution that I mentioned last week which I developed for SpinRite appears to have solved a number of current and doubtless future troubles. As I've mentioned, I take every single problem that our testers can find absolutely seriously because I learned a long time ago, if it happens to one person, it'll happen to a hundred.

So thanks to additional feedback from our testers, and my hours of staring at code, and then a number of additional tests and incremental improvements, you know, up in the very far right corner of that screen shot it said "Development Release 7F." And so we had 7, and then we quickly had A, B, C, D, E, F because they didn't feel like they were big enough to jump to 8, but I kept hoping I was going to hit the magic solution for whatever problem I was trying to track down. So we've had a bunch of tests and incremental improvements.

What's been so gratifying is that as testers have swung by the development group over the weekend, one after another they've reported that this or that trouble their systems used to have is now gone, and that SpinRite's latest development release is now working perfectly for them. So we've seen a lot of progress. But there are a few exceptions, and they're nasty. A couple of people have reported that the size of the first drive in the list is being confused with the size of the last BIOS drive in their system. It doesn't happen to me, nor to most people. But it reliably happens to a couple of our testers. So something different is going on in their systems. One person has a 70TB Drobo directly attached to their system and is BIOS accessible, and that was crashing SpinRite because I wasn't ready for a 70TB drive, which is overflowing some register space. So I'll expand that.

Leo: The other thing with Drobos is they're not always reporting the actual space.

Steve: Yes. That's exactly what we saw. He said he only had 32TB, but it's saying 70.

Leo: Yeah, that's Drobos right there.

Steve: Yeah. And a few others are reporting something really odd. SpinRite's own division overflow pop-up is popping up. Now, the pop-up tells us where the division overflow occurred. But when I look there in SpinRite's code, there's no division instruction at that location. So the only conclusion is that something is causing SpinRite's own code to be overwritten. Fortunately, what's being overwritten contains either an F6 or an F7, which are the x86 division opcodes.

So I know where the first drive's size is stored, which something is changing later when it shouldn't. And that errant division overflow pop-up tells us where a division instruction has mysteriously appeared in SpinRite's code. So we need to track down what's going on with the drive size being changed and the code being overwritten. To do that we need to catch the overwriting action in the act, as it's happening. The problem is I've never been able to reproduce any of those problems myself. It's only happening on a few distant machines.

The answer, which I will immediately implement a few hours from now, maybe in the morning, depending upon how I'm feeling after the podcast, is to build a native debugger directly into SpinRite. And that's actually easier than it sounds. The Intel x86 architecture has always incorporated built-in hardware debugging assistance. It has four 32-bit breakpoint registers which can be set to point to any region of the system's memory. Each of the four breakpoint registers can be set to interrupt the processor upon the execution, the reading, or the writing of that memory with a 1, 2, 4, or 8-byte span. So this next test release of SpinRite will add a couple of new command line options to load and enable those debug registers.

Once a tester has seen where SpinRite reported an erroneous division instruction, they'll be able to immediately re-run it, setting a write breakpoint at that spot in SpinRite's code which will allow us to catch in the act whoever is overwriting SpinRite. I'm assuming it's me with some errant code of mine. It might be their BIOS. Again, I can't make it happen. They can make it happen, and this will allow anybody to work with me, essentially doing remote debugging of the code using the x86 debugging instructions. So we're going to have an interesting time in the next week, and I'll probably report that SpinRite has got these problems solved by a week from now.

Leo: That's amazing. That's amazing. Meanwhile, I'm still playing bingo with some squid on my submarine.

Steve: Or as Lorrie says, "I made the bed."

Leo: I got something done today. You know, that's good, that's good. Steve, let's take a break. Then we're going to get into the meat of the matter.

Steve: XSinator.

Leo: Whatever you mean by XSinate. We will find out in just a moment. All right, Steve. Let's talk about, what is it, signitficit? What are you talking about?

Steve: I would assume we would pronounce it XSinator, X-S-I-N-A-T-O-R, XSinator.

Leo: XSinator sounds right, yeah.

Steve: Okay. Their paper is titled: "XSinator.com: From a Formal Model to the Automatic Evaluation of Cross-Site Leaks in Web Browsers."

Leo: Oh, XS as in cross-site, okay, okay.

Steve: Exactly, cross-site. It's the result of a comprehensive work conducted by a team of five German University researchers. And as a result of their work they discovered 14 new types of cross-site data leakage which are effective against, well, I think every browser - Tor, Firefox, Chrome, Edge, Safari, Opera, and others, both desktop and mobile, as I said. And going to XSinator.com allows you to run the test that they've designed which will profile your own browser, it shows up in the left-hand column, and then allow you to compare it against all the other browsers which had been profiled.

So they call the bugs "XS-Leaks," as in cross-site leaks, because they enable malicious websites to harvest the personal data from their visitors as they interact with other websites in the background, that is, as the hostile website you go to, or for that matter an ad that is running script, all that has to happen is script is run in your browser. Then reach out using JavaScript using AJAX to create connections to other browsers with which you have a relationship.

In a recent statement about their research, which was presented during last month's 2021 ACM SIGSAC Conference on Computer and Communications Security, which by the way their presentation garnered a Best Paper Award, they explained. They said: "XS-Leaks bypass the so-called same-origin policy" - and whoops, we know how crucial that is in our browsers for browser security - "one of a browser's," they wrote, "main defenses against various types of attacks. The purpose of the same-origin policy is to prevent information from being stolen from a trusted website. In the case of XS-Leaks, attackers can nevertheless recognize individual small details of a website. If these details are tied to personal data," they said, "those data can be leaked."

Now, our listeners probably know by now that I'm a sucker for formally proven security findings. There's certainly a place for fuzzing, which is probably the other end of that spectrum from formal proofs. And formal proofs won't help when modeling cannot be applied. But whenever possible, creating a mathematically formal model of a system, then using that model to reach and/or demonstrate security conclusions, is in my mind the gold standard.

So here's how this team describes what they've accomplished. In their abstract for their paper they said: "A Cross-Site Leak describes a client-side bug that allows an attacker to collect side-channel information from a cross-origin HTTP resource." They wrote: "They pose a significant threat to Internet privacy, since simply visiting a web page may reveal if the victim is a drug addict or leak a sexual orientation. Numerous different attack vectors, as well as mitigation strategies, have been proposed; but a clear and systematic understanding of XS-Leak root causes is still missing." Or at least was before they began this.

They said: "Recently, Sudhodanan et al. gave a first overview of a XS-Leak at the Network and Distributed System Security Symposium (NDSS)." They said: "We build on their work by presenting the first formal model for XS-Leaks. Our comprehensive analysis of known XS-Leaks reveals that all of them fit into this new model. With the help of this formal approach, we, one, systematically searched for new XS-Leak attack classes; two, implemented XSinator.com, a tool to automatically evaluate if a given web browser is vulnerable to XS-Leaks; and, three, systematically evaluated mitigations for XS-Leaks. We found 14 new attack classes, evaluated the resilience of 56 different browser/OS combinations against a total of 34 XS-Leaks, and propose a completely novel methodology to mitigate XS-Leaks."

Now, in the show notes I have my own results using Firefox and Chrome, Firefox 94 on the left, Chrome on the right. And red is bad. There's a lot of red there, both Firefox and Chrome. In their paper, and I did not include it in the show notes - I forgot - they have some other tables where they show all of the tests and all of the different browsers and

version numbers. And for what it's worth, Firefox has fewer reds than anything else. The Tor browser, which of course is based on Firefox, has even a few fewer. But, yeah, so Leo, now you have on the screen the - I guess that's the pre-test page. And up at the top there you're able to click a blue button, which is like right there, and click it, and it will begin, it will initiate the test which is now underway on your browser. So we'll see how that comes out.

Leo: How long is this going to take?

Steve: It's only a few minutes.

Leo: Okay. I'll be back with you in a minute.

Steve: And I think that opened another tab. Switch back to the first tab, and you might be able to see it operating, if it opened a second...

Leo: It's running, yeah.

Steve: Oh, yeah, there it is. And you're seeing - you're beginning to get some red.

Leo: These are the initial results. Okay, yeah.

Steve: Yup. I think it's still in process.

Leo: What are these things? Is there something I should be worried about here?

Steve: Yeah. That's why we're talking about this today.

Leo: I'm in Firefox doing this. Okay.

Steve: Yeah. Okay. So in a web application, they explain, a web browser interacts with several different web servers through HTTP or web-socket connections. The client-side logic of the web application is written in HTML, CSS, and JavaScript, and is executed inside a tab of the browser, or inside an inline frame in another application, or an ad. The execution context of a web application is defined through the concept of web origins. Web applications may call and embed other web applications to enhance functionality.

For example, a hotel reservation site might embed Google Maps and public transportation sites as an easy method to allow its customers to determine how to reach the hotel. In such situations, cross-origin HTTP requests between different web origins, meaning the hotel's origin and the Maps server origin, are necessary to retrieve data to embed and display in the web application.

When interacting with a website, a user has a well-defined state. This state typically contains the information such as whether the user is logged into the site currently or not.

Besides the login status, the user state may contain account permissions such as admin privileges, premium membership, or restricted accounts. The number of different user states is potentially unlimited. For example, in a webmail application, a user may or may not have received an email with the subject "top secret."

Okay. So they say, under Privacy Risks of Cross-Origin Requests: "Consider the following situation. The attacker has lured a victim to a malicious web application that executes hidden cross-origin HTTP requests to different drug counseling sites. If the attacker could learn whether the victim has logged in at one of these drug counseling sites, the attacker could gain highly privacy-critical information about the victim.

"To distinguish between two user states, the attacker's JavaScript code must be able to identify differences in its own execution environment resulting from different responses to cross-origin HTTP requests. These different responses must correspond to different user states at the target" - that's sort of the victim - "web application. If this differentiation is possible, we call this vulnerability a cross-site leak. The attacker can then craft a malicious website, which triggers the cross-site leak once the victim visits it." Okay. We'll get specific here in a second.

So then they give two real-world examples, an XS-Leak on GitLab. They wrote: "GitLab is a popular web application for collaborative software development hosted by many companies. GitLab provides a profile URL, which is https://git.company.com

/profile. If the user is not logged in, this URL redirects the user to https://git.company.com/users/sign_in. If the user is logged in, the current user's profile is shown. However, since the attacker embeds GitLab cross-origin into the attacker's own web page, the attacker cannot directly read the URL.

"In Listing 1 above," and I've included it in the show notes, "we use the window.length property, which is readable cross-origin, to determine the user state at the other site. The profile page does not contain any iframes, but the login page includes three iframes. If this property, window.length, has the value 3, that means the user is not currently logged in. If it has the value 0, the user is logged in. By scanning different company websites hosting GitLab, the attacker can collect information on a programmer's affiliations." So there's an example of true cross-origin information leakage where a site you visit or an ad hosted by a benign site is able to determine things about you and your current state with other domains, with other websites.

And it works on Google Mail. They give an example of a cross-site leak on Google Mail. They say: "Google Mail, one of the most popular webmail applications. In 2019, Terjanq (T-E-R-J-A-N-Q) reported a XS-Leak which could determine whether an email with a certain subject, for example drug counseling, or content was present in the user's inbox cross-origin." In other words, it's possible to reach into a user's Gmail this way.

"The cross-site leak abused the common cache that web applications share. By using the advanced search option, which can be called 'cross-origin,' Google Mail marks search results, if any exist, with a dedicated image. To perform a cross-site leak attack, the attacker first empties the web cache, which can be done locally, then calls Google Mail advanced search using a URL, which can be done, and finally checks if the dedicated image is available in the cache. If true, the search was successful, and the attacker learned that an email containing whatever search term the attacker queried currently exists in the target's, the victim's inbox."

Okay. So let's make sure that everyone understands where we are. Any Gmail user visits some random nosy website. Without any indication of any kind, that nosy website running JavaScript on the user's web browser is able to use subtle and clever tricks to

perform as many go/no-go searches of the user's Gmail inbox as they wish against the current content, entirely behind their backs and without their knowledge.

We've talked about this general class of web browser hacks a number of times in the past. Broadly speaking, it is incredibly difficult to robustly prevent any cross-domain leakage. We've seen that initially benign-seeming convenience features, such as changing the color of previously visited links, can compromise privacy when website A uses the user's browser to display website B offscreen and then probes website B's DOM, the Document Object Model, for the colors of various links. In this way, information that is none of website A's business leaks from the user's previous use of website B.

What this team has done is further explore and rigorously map brand new forms of subtle side-channel cross-origin information leakage. And they've come away with a surprisingly large number of new ways for clever attackers to leverage deliberate features of JavaScript, like using the window.length of some other website's page to infer information that's none of their business about the browser's owner.

We've seen that previous cross-domain cookie and cache attacks have led browser developers to segment all cookie storage by domain, and to similarly segment all browser cache by domain. Since the performance impact of this is negligible, the great impact is upon browser complexity and maintenance, which have skyrocketed. If you've looked at the number of processes today running when you launch your web browser, it's clearly a huge burden now on browser developers. Like today's operating systems, web browsers are no longer something that can be casually assembled.

But the nature of the leakage these guys have uncovered and demonstrated is different from that because it leverages the non-gray area features of our browsers. They're employing features that are there by design. This suggests, in turn, that in the future our web browsers may choose to deliberately strip these side-channel features and limit JavaScript's interaction with cross-origin domains. And that would probably be a good thing. It's going to be tough because it's going to change some functionality. But what we've seen is that the browser vendors are extremely conscious of this kind of cross-origin privacy leakage, and this paper just unveiled a huge new assortment of ways that can be done.

Leo: Really interesting, yeah.

Steve: Yeah.

Leo: So presumably Chrome, Firefox, the browser guys...

Steve: All of them. All of them.

Leo: ...will fix this; yes? It's fixable; right?

Steve: Yes. And that's the upside of this kind of research and this kind of testing site. I mean, this will immediately subject the browser vendors to significant pressure, which, you know...

Leo: They need.

Steve: This is not their fault, but they do need the pressure.

Leo: They want to fix it. Now Grayson Peddle in our Discord chat has a question which I think is germane. If you're using a browser that sandboxes tabs, does that keep the cross-site scripting from happening? They can't communicate with one another; right?

Steve: I don't think tab sandboxing would work because in a tab the JavaScript is using HTTP queries and web sockets to bring another domain into it.

Leo: Right.

Steve: So it's not having to talk across the tab.

Leo: So it bypasses the tab. It's in one tab. It's in one process.

Steve: Exactly. It's all happening on that, on the page the user is currently visiting.

Leo: Apparently there's some sort of container method for Firefox.

Steve: Well, and Leo, the fact that we're seeing red when we click that button...

Leo: Yeah, that tells you.

Steve: Those are effective, successful cross-domain probes that have just occurred.

Leo: Right, right, right. Okay. So Grayson, is there a special version of Firefox that containerizes the various instances? I don't even know if that'd help because it's all happening within one instance; right?

Steve: Yes. It's on the current tab. He's certainly referring to the tab isolation.

Leo: Yeah, that's what I thought he was. But, yeah, okay.

Steve: Yes. And that's a great point. But here it's script on the page is poking at other domains behind your back. And so the problem is that those domains know about your browser. So when JavaScript in your browser pokes at those domains, the cookies that you have, the other information that your browser has about those domains go along with those queries. And so that's how those foreign domains respond differently to you. It's because your browser via the JavaScript running on that page has asked them something, and their behavior subtly changes when it's you asking versus somebody else asking.

Leo: So interesting. That's good research. Nicely done.

Steve: It's really well done.

Leo: Yeah, yeah. And of course so is this show, thanks to this guy right here, Mr. Steven "Tiberius" Gibson. If you want to - we have lots of ways to get this show. Of course you can go right to Steve's site, GRC.com, and get 16Kb audio, as well as 64Kb audio, if you want to save on download bytes. Also the transcripts, which he commissions from Elaine Farris, are really good and a great way to search any Security Now! episode. You've got transcripts for all 848? No.

Steve: Yeah, we went back and did them from the beginning.

Leo: That's amazing. Really makes this a very valuable resource. GRC.com. While you're there, pick up Steve's bread and butter, the one and only thing that makes him money, so buy a copy of SpinRite, the world's finest mass storage maintenance and recovery utility. 6.0 is the current version. But as you can tell, he's working hard on 6.1. You'll get to participate in the development and get a free upgrade if you buy today: GRC.com. Also lots of other stuff there. It's a really...

Steve: Completely rewritten, and I didn't even change the major version number of the darn thing.

Leo: Yeah. Yeah, why didn't you make it SpinRite 7?

Steve: Because I promised everybody a free upgrade. So everyone's getting a free upgrade.

Leo: Okay. And you also made some promises about what 7 might mean. Right?

Steve: Yeah. 7 will definitely go further, and it'll run on UEFI systems.

Leo: That's the big difference, yeah.

Steve: And have, well, and also have native drivers for USB-attached devices and NVMe, which we're also seeing.

Leo: And it'll basically open it up to Mac users, which is great.

Steve: Yeah.

Leo: Yeah. So you'll get 6.1, but then it's time for 7.0. And if Steve lives long enough, if I live long enough, we'll deliver that, as well, but not for free. There you go. Leave feedback at GRC.com/feedback. You can also slide into his DMs, as the kids say, on Twitter. He's @SGgrc, and his DMs are open. So questions or comments can be left there, proposals for the Picture of the Week, that kind of thing. Let's see, what else?

We have copies of the show on our website. We have 64Kb audio also. But we also have a unique format. We have a video. If you want to see Steve's mustache in action, you can go to TWiT.tv/sn. You can also subscribe in your favorite podcast client, which is probably the best way. That way you'll get it automatically.

The show we do live Tuesdays, right after MacBreak Weekly, 1:30 to 2:00 p.m. Pacific, let's make it 2:00 p.m., 5:00 p.m. Eastern time, 22:00 UTC. If you want to watch or listen live, the stream is at live.twit.tv, the streams. There are several of them. People who are watching live are often in the Discord in our club or on our free IRC server, irc.twit.tv. That's a community-run resource. Thanks to all of our mods. They're really amazing in there, starting with OzNed and ScooterX, and just a great bunch of people in there. That's always a lot of fun. And that's free and wide open. You can use a browser or an IRC client: irc.twit.tv.

We also have forums. Steve's got forums. We had to have them. So we have Discourse forums - Discourse, not Discord - at twit.community. We have a Mastodon instance at twit.social. If you feel like tweeting in a nice group of people, twit.social. We don't call it "tweeting" in Mastodon. We call it "tooting." So toot along. Don't. Don't. I know what you're thinking.

Steve: Beans, beans, the wonderful - yeah.

Leo: Yes. That's exactly what I thought. I forget to mention, and I want to mention this a little. It's almost like a tech support note. But we also offer ad-free versions of all of the shows as individual subscriptions through iTunes, through Apple's podcast client. I think it's $2.99 per show. So if you just want Security Now!, less than 3 bucks a month you get Security Now!. But, and this is a - I think it's a bug in Apple's system. Don't be too in much of a hurry to get it there because what happens, we upload the thing, and then there's some disconnect, but for a brief moment of time, I know, it might be like 30 seconds, if you're there right when we upload the show, you'll get the version with ads.

So a number of people have said, wait a minute, I pay 2.99. There should be no ads. Yeah, because of the way Apple does it, it doesn't switch over to the ad-free version right away. There's a lag. And we don't know why that is. We've talked to Apple. We hope they fix it. But in the meanwhile don't be in a hurry if you pay for the ad-free version through podcasts, the Apple Podcast app. Let Apple's Podcast get it. And then I think that should work out all right. Don't force it, in other words.

I think that's it. I think we've done everything. Steve, you're the best. While you were talking I was able to find the bug in my Day 4 solution of Advent of Code. Moving on.

Steve: Oh, cool. Nice.

Leo: Oh, I just transposed some lines, you know, it's one of those things. I kept getting a weird error message.

Steve: I do. I do.

Leo: Yup. Got to put the break higher up in the four fold. What was I thinking? So we'll see you next time, next Tuesday on Security Now!.

Steve: Thanks, buddy.