Taming Web Ads

Description: This week Steve and Leo discuss developments in the New Windows on Old Hardware front, Cisco finds a surprise in the Vault 7 docs, Ubiquiti was caught with their PHPs down, Check Point discovered problems in WhatsApp and Telegram, some interesting details about the long-running Yahoo breaches, the death of the "eBay Football," the latest amazing IoT insanity, the incredible results of the CanSecWest Pwn2Own competition, a classic "you're doing it wrong" example, Tavis pokes LastPass again, some miscellany, and an interesting proposal about controlling web advertising abuse.

High quality (64 kbps) mp3 audio file URL: http://media.GRC.com/sn/SN-604.mp3
Quarter size (16 kbps) mp3 audio file URL: http://media.GRC.com/sn/sn-604-lq.mp3

SHOW TEASE: It's time for Security Now!. Steve Gibson is here. The title of this show, "Taming Web Ads," is just a small portion, a small fraction of the total show. We've got a lot of great stuff, including the results of Pwn2Own, some surprising hacks, and an IoT device that really just violates every rule. We're going to talk about it next on Security Now!.

Leo Laporte: This is Security Now! with Steve Gibson, Episode 604, recorded Tuesday, March 21, 2017: Taming Web Ads.

It's time for Security Now!, the show where we talk about you, your privacy, and security online. This is the guy, man; this is the resource. Everybody - college classes, IT professionals, CIOs, everybody wants to know what Steve Gibson thinks, from GRC.com. Hi, Steve.

Steve Gibson: Hey, Leo, great to be with you for Episode 604.

Leo: 604.

Steve: Yes. And actually I learned that even security researchers are listening to the podcast. I have some feedback from the guys we talked about last week at the U.S. Naval Academy that I'll share later in the episode.

Leo: That always makes me nervous when the people we're talking about are listening; right? Not you. You have the confidence of your convictions. You know.

Steve: So, yeah, we have a lot of news this week, tons of stuff to talk about. We've got some developments in what I call the New Windows on Old Hardware front. Cisco found some surprises in the Vault 7 CIA doc dump. Ubiquiti was caught with their PHPs down.

Leo: Uh-oh.

Steve: Check Point discovered problems in WhatsApp and Telegram. We've got, finally, some really interesting details about the long-running Yahoo breaches as a consequence of the FBI's indictments in that case, where a lot of facts were revealed, that finally we have something we can talk about, which I'm really happy for. And it's kind of creepy. We've got the death of the eBay football that was one of the very first things we talked about years ago.

Leo: Yeah, my first dongle, yeah.

Steve: Yeah. They sent out email to all football users saying, eh, we're improving our security by eliminating the best option. Okay. Anyway, we also have, believe it or not, something so amazingly crazy in IoT insanity this week. It's, wow. Actually went to a class action settlement, it was so over the top. We have some incredible results from the CanSecWest Pwn2Own 10th Anniversary competition, which pretty much demonstrates that, if you dangle enough money in front of security guys, they can pretty much poke a hole in anything that they aim at. Because, as we know, we talked about this a few months ago that it was coming up, that this was the largest set of prizes ever, a million dollars' worth of prizes.

We've got another classic "you're doing it wrong" example. Tavis has poked another hole in LastPass, which is already resolved, but worth talking about. We've got some miscellany. And I want to - and this isn't, like, the most important thing. But it was an interesting, nicely written piece on Medium that I wanted to share with our listeners to wrap this up about taming web ads, from someone who is very sophisticated technically, has presented at O'Reilly's OSCON a few times, is involved in the open source community, and was able to quit his job because his page got enough success with advertising. But it then became a problem. So just a whole bunch of great stuff to talk about. Oh, and a fabulous Picture of the Week.

Leo: All right, Steve. Let's get the security news. Oh, you want to do the picture first?

Steve: Oh, Leo. This is just - I love cleverness. And this is just so clever. This is one of those where you really don't even need to say anything. The picture tells the whole story. But I gave it the caption "Setting Traps for Autonomous Vehicles." And of course everyone knows who understands the rules of the road that you're not allowed to cross a double line, but you can cross from the broken, sort of like the dotted line mated with a solid line, you're able to cross from the dotted line into the solid line side, but not the

other way.

And so someone cleverly realized that, if you created a large circle with an outer dotted line and an inner solid line, then a car following those rules could enter into that region, but could never leave. So anyway.

Leo: I didn't even - I'm glad you explained it. I get it. So it's like a roach motel. They can check in, but they can't check out.

Steve: Yes, exactly. It's a trap. The car would see the dotted line on its side and go, oh, and then roll across. Now it's facing a solid line, and it can't ever leave.

Leo: Aw, poor little car.

Steve: I love it. Very, very clever. So, okay. In the news this week has been something that we initially touched on some time ago that was - at the time it was controversial. Well, and actually it drove me to immediately purchase what would be my final Windows 7 laptop, and to build that monster machine that our listeners know that I built, what, maybe a year ago or so, using the last hardware that Microsoft promised to support all the way through the end of Windows 7 support, which is April of 2020. And that was the Haswell chipset. The succeeding chipset, which was Skylake, they said, well, we know that enterprises are slow to move. So we'll support Skylake through 2017, but not through 2020. But, so, okay, and so that at least gives companies some time to move themselves.

I, of course, am unwilling to ever leave Windows 7, which I think is the perfect operating system. So I purchased some hardware that would be able to get its updates from Microsoft through the end. So what started happening, and caught some people by surprise, was error messages that they began to get on later hardware. And Microsoft has a support page where the title of the support page is, which is quoting their message: "Your PC uses a processor that isn't supported on this version of Windows." That's the error you receive when you scan or download Windows updates.

And so on this page where Microsoft is diagnosing what they call the "symptom," it says: "When you try to scan or download updates through Windows Update, you receive the following error message: Unsupported Hardware. Your PC uses a processor that isn't supported on this version of Windows, and you won't receive updates."

And then they say: "Additionally, you may see an error message on the Windows Update window" - which is unfortunately unclear - that says: "Windows could not search for new updates. An error occurred while checking for new updates for your computer." And then it's got an obscure error code, 80240037, Windows Update encountered an unknown error. Okay, well, it's not unknown, it's just unarticulated in this instance.

And so under the cause for this symptom their page says: "This error occurs because new processor generations require the latest Windows version for support. For example, Windows 10 is the only Windows version that is supported on the following processor generations." And then they enumerate Intel 7th-generation processors, the AMD Bristol Ridge line, and Qualcomm's 8996 processors.

And they say: "Because of how this support policy is implemented, Windows 8.1 and

Windows 7 devices that have a seventh generation or a later generation processor may no longer be able to scan or download updates through Windows Update or Microsoft Update." And then for their resolution of this problem: "We recommend that you upgrade Windows 8.1-based and Window 7-based computers to Windows 10, if those computers have a processor that is from any of the following generations." And then they continue.

Now, of course, this was picked up by the news, who characterized it as - Forbes said: "Microsoft Admits Forcing More Users Onto Windows 10." HotHardware had two stories: "Microsoft Disables Windows Updates for Ryzen and Kaby Lake PCs running Windows 7 and 8.1," and also "Microsoft Apparently Ramping Up Heavy-Handed Tactics to Force Windows 10 Migrations." Neowin said: "Some new PCs running Windows 7 and 8.1 won't receive further updates." And the Hacker News: "Microsoft Started Blocking Windows 7 and 8.1 Updates for PCs Running New Processors." So, you know.

And we talked about this. This is expected, actually. But certainly maybe to those who didn't know or people who are going to be disappointed that they were deliberately wanting to run an older Microsoft OS on newer hardware. And it's difficult to regard this as something other than deliberate because Microsoft is continuing to support Windows 7 and 8.1 on the hardware of those OS generations. Meaning Haswell, the hardware that I deliberately bought for this reason, will get support all the way out through 2020. Now, and I'm not in a position to be able to definitively say how much effort this saves on Microsoft's part. Maybe Paul and Mary Jo can shed some light on it.

Leo: Yeah, well, we talked a lot about this. I think, to me - and it was a little bit of a surprise when this whole thing came up about a year ago.

Steve: Right.

Leo: It shows you - and this is what we concluded on Windows Weekly - how much back-patching Microsoft does, when a new processor comes out, for the new microcode in the processor, to make sure that Windows works. You know, you would just assume, well, it's x86 architecture. What would you have to do? But it turns out there's a significant coding effort each time there's a new generation of Intel processors, of Haswell, of Kaby Lake and Skylake and all of that. They have to add a considerable amount of back-patching to versions of Windows, to older versions of Windows, so that they'll work.

Steve: Right.

Leo: You know, and the reason I kind of feel like it's not a commercial, you know, you could say, well, they're just trying to get you to buy the new version of Windows. But the reason I think it's not that is I look at what happened with Skylake and the Surface Pro 4 and the Surface Book. For eight months my Surface Book had all sorts of power nightmares because it didn't work well with Skylake. The Intel Skylake processor was so dramatically different from the previous Haswell generation in terms of power management that this thing, it literally - Microsoft famously said it was a "tough computer science problem." Remember that? That was them getting the patches, the firmware fixes, so that you could - and now it does, by the way. The Surface Book Pro works great. Power management is great. But it was,

I think, really because of Intel's massive changes to power management in Skylake. So I don't think it's unreasonable for a company to say, look, we would have to write a lot of code, we'd have to do a lot of work to get these older versions of Windows to work with new processors that didn't exist when we wrote them, so we're just going to cut them off.

Steve: And along with that you might argue that their metrics could show that it would largely be wasted effort. That is, how many people are actually going to take advantage of Windows 7 or 8.1 on the newer hardware, especially when the new hardware probably comes bundled with Windows 10.

Leo: Right, right.

Steve: So, yeah. I can certainly see it their way.

Leo: It's tempting to assume it's, you know, it's anti-competitive. They want to make you upgrade, et cetera, et cetera. But I think there might be a good technical reason for this.

Steve: Yeah. And when we discussed this last year, I also agreed that, I mean, again, I can't imagine the nightmare of needing to maintain something as complex as Windows has become across a growing number of disparate hardware platforms. So I would say I don't even know, I haven't looked to see whether Haswell chips are even still available. But for our listeners, I mean, 7 is still the majority OS on the Internet. I mean, there are more Windows 7 than Windows 8.1 and 10, despite Microsoft's efforts to move people away.

So people who - and I'm not even sure how you would get and register 7 now. I can because I'm an MSDN subscriber, and I pay for the privilege for development purposes, to be able to make sure the software I write is backwards compatible to what is still the majority OS on the Internet, the majority Windows platform. But if Haswell chips are available, if you don't want to be forced to be move to Windows 10, and if you want updates through the life of Windows 7 or 8.1 updates, then now would be the time to move to a Haswell chip, if you're on something older, and if it matters to you, because this is now more than just a theoretical problem.

Leo: This is real, yeah.

Steve: This is real, yeah. So Cisco was rummaging around in the CIA's leaked Vault 7 documents and discovered that the CIA knew something they didn't know about their own high-end IOS-based switches and routers. Now, to be clear, this is not the low-end Cisco/Linksys consumer routers. These are the big iron routers, like I had that were terminating my T1s. I had a 3600 Series router running IOS. That would have been vulnerable if I was using what Intel calls the Cluster Management Protocol, CMP. They have something called Cluster Management Protocol which is used to link, as its name sounds, clusters of routers together. But apparently, even without that explicitly running, because its transport is Telnet, having Telnet exposed to an untrusted machine or IP,

that is, not having an access control list to restrict who has access to a router's Telnet protocol allows a remote code execution to be performed.

So this rocked the high-end router and switch community this week, or late last week. Cisco produced an advisory immediately. The only workarounds are turn off Telnet. That's all you could do is shut down Telnet because, if it's on, and a bad guy can get access to it, then you're in trouble. And if you cannot do that, then apply an access control list to restrict the IPs which can generate Telnet-based traffic. Telnet is TCP-based, so an access control list is perfect protection, unlike the UDP, where you could spoof the source IP. In order to bring up a Telnet connection, you first have to have the standard TCP three-way handshake, which inherently validates the source IP of the incoming TCP connection request. So an access control list is all you need.

I have, for example, I use Telnet throughout my network. There isn't a single instance of it where I haven't already, I mean, where I didn't already lock it down so that - because, I mean, Telnet, you just don't want to leave that open. The problem is, you know, it's certainly been replaced with SSH, where you use a good certificate authentication everywhere. But there are still older devices, and in this case many IOS devices, where they use Telnet for their admin access and control within an environment. It's just - it's an old protocol. The only protection that exists for it is username and password. And you certainly don't want that open to attack because you cannot monitor people making a brute-force attack on that protocol. So they are going to be producing a patch.

This affected as many as 300, or I saw more than 300 different Cisco switches and routers. So anything with IOS or IOS XE is vulnerable. So if any of our listeners know, if they're using or know somebody who is using pretty much any IOS-based Cisco device with Telnet management, where there's any chance that a bad guy can get access to a Telnet connection, you want to do some mitigation immediately. Again, nothing for the consumer, at the low-end consumer end. But I'm sure there'll be a fix for it. It'll be an IOS update that will fix the problem that was found.

But it is spooky that this was something that our CIA apparently knew. It was part of the Vault 7 document disclosure. And much as last week we discussed how Apple was going through it, and the good news was many of these things had already been fixed in previous iOS updates in the case of Apple - a different iOS, of course, than Cisco's. So Apple was continuing to go through all of this and make sure that there wasn't any other news. And Cisco has. And I imagine pretty much all of the major players want to make sure that there isn't something that is now becoming publicly known, I mean, and certainly even when it was privately known they would want to fix this, if they were able to.

Many of our listeners sent me news of this problem with Ubiquiti. The first thing I need to say is that it was only one of the many OSes which Ubiquiti products are based on, and in this case the airOS. So not the EdgeMAX OS, which our favorite little $49 five-interface router/switch is using. But the airOS devices did have a problem. And Ubiquiti, I would argue, did not respond the way they should have. Toward the end of last November, so about four months ago, a security researcher, Thomas Weber, with SEC Consult in their Vienna office, discovered a command injection flaw in one of the CGI scripts that the airOS web server runs and has available. That's the pingtest_action.cgi script.

And what was most disappointing, as someone who certainly likes at least the Ubiquiti non-airOS hardware, is that it was caused in part by the fact that those Ubiquiti products are today employing a 20-year-old - 20, I didn't even know that PHP was 20 years old - a 20-year-old version of PHP, v2.0.1, from 1997. So, and it's in there. And it turns out that the vulnerability can be exploited by luring an attacked user to click on a crafted link or

just to surf to a malicious website because the entire attack can be performed via a single browser GET request because there's no, in that version of PHP, no cross-site request forgery (CSRF) protection.

This allows an attacker to open a port or to open a reverse shell to connect remotely into that device, which would then allow them to change the device's password, since the web service which executes that CGI script on behalf of whomever calls it runs with root privileges. So even low-privileged read-only user accounts, which are easily created through the web interface, are able to perform the attack. So Ubiquiti has four OSes. They have the UniFi OS, the EdgeMAX OS, and the AmpliFi OSes, none of which were affected. Only the airOS-based products are. But there's like a long list of those. So people who have followed our affection for the EdgeRouter X, we're all fine. The EdgeOS or the EdgeMAX is not affected. But airOS products are.

Now, there was initially a lot of back-and-forth with Ubiquiti. Thomas was patient and kept reaching out. They initially said they were already fixing it. Then they said the proof of concepts that he had provided were not working. Then they said, oh, yes, they are, and then it was being fixed. And they said it would be out soon. Then he asked them several more times for updates in February and never got any responses. So finally he just - he went public with the news, but even then out of respect held back the proof of concepts. That happened on the 16th, and that finally motivated them to release patches on the 18th. So there's a series of patches available. Anyone who is using Ubiquiti products with airOS should definitely update their devices to the latest immediately.

Leo: This doesn't affect our EdgeRouter X, right, though? This is not...

Steve: No, does not, does not.

Leo: Yeah.

Steve: And so this is another example of, first of all, why we desperately need security researchers to be free to do this. And perfect conduct, I would say, on Thomas's part. The whole thing would have happened. Who knows what was going on behind the scenes. Maybe people changed positions, or somehow the communication faltered. Maybe they got annoyed with his persistence. I would argue that he's in a position of leverage. And then he finally was forced to use the leverage, to let the world know that there was a problem. Even so, he held back the details that would allow immediate exploitation until they got the patch done.

So anyone with airOS should update. People with the wired products, as Leo, you just reaffirmed, are fine. But still, again, we have to do everything we can to protect the system as it is now. I'm so worried because you can imagine that there will be a lot of pressure on Congress to tweak the legislation, along the lines of the DMCA, to not explicitly exclude responsible research. We have to have that maintained. And if we don't, we're in deep trouble.

And here's another example: Check Point, who as a consequence of their research discovered a vulnerability in the web browser interface for hundreds of millions of WhatsApp and Telegram accounts which allowed hackers, would allow hackers to take over those accounts. And it wasn't clear to me whether this was found in the wild. I mean, Check Point generally finds things happening and then figures out what was going

on. And I should have tracked this down more carefully. But the headline that I wrote said: "Check Point discloses vulnerability that allowed" - past tense - "hackers to take over hundreds of millions of WhatsApp and Telegram accounts." But it wasn't clear to me whether they found this and responsibly reported it, or whether it was a zero-day.

On March 7th, a few weeks ago, Check Point discovered and reported to both companies a serious web-side attack which was made possible by the way they were both using the browser. So of course they have device-side apps. But as a convenience, they also allowed a browser-based solution. In both cases of WhatsApp and Telegram, Check Point found a means for bypassing each system's file upload authentication, which would allow a malicious party to essentially send malicious content to the target victim which, if displayed in the browser, could leverage browser vulnerabilities to access the account's protected local store, and thus allow access to sensitive account information.

So what that would allow is that a malicious party could then send a file, for example, an image, which when viewed would immediately compromise the user's WhatsApp or Telegram accounts, giving a remote hacker full access to the user's account and all account data, including their profile, personal photos, chat history, contact lists, and so forth. The attacker could then resend a malicious image to all of the compromised users' contacts and, in turn, compromise them.

So both companies quickly responded and repaired the vulnerabilities. And thanks to the fact that this was a web-side attack, basically they were just able to update their web-side code and quickly lock this down. Check Point's blog post for this, I put the link in the show notes. If anyone's interested, it has a detailed walkthrough which, you know, they've done full disclosure now because the update was able to be pushed out immediately and fixed.

And for the longest time we've been talking about - for, like, years - the news of the Yahoo breaches, yet very little information was available. Yahoo was, as we have discussed, very late in disclosing that they knew of any trouble. They were dragging their heels. Only after they essentially were found to have had major breaches would they then say, oh, okay, yeah, you're right. Anyway, so what finally happened is that last week the FBI indicted four individuals, three Russians and one Canadian, with charges of hacking Yahoo, which their forensics demonstrate started back in 2014, so three years ago, which allowed them to obtain access to half a billion Yahoo accounts.

According to the FBI's forensic investigations and subsequent allegations, Yahoo was a persistent target of the attacks that these four individuals launched. This began with spearphishing attacks, which induced targeted recipients to open malicious documents or click on links taking them to spoofed phishing websites, where, for example, in many cases Google Gmail credentials could be requested and harvested.

The attackers also targeted Yahoo's own employees using these same phishing attacks, social engineering attacks, essentially, eventually obtaining their account credentials; and then leveraged that access, for example, maybe their Yahoo email credentials were reused for Yahoo network access. So they were able to basically spearphish Yahoo employees and, in instances where their email credentials were being reused - or maybe they spearphished them with like a fake Yahoo employee portal logon, you know. Because again, as we have been saying, ultimately people, social engineering and people are going to end up being the weakest link, even if we make everything else work perfectly.

So in early 2014, they had achieved deeper access into Yahoo's corporate network. And throughout 2014 they were doing this, and they got well in by September. They installed

their own software into Yahoo's network to cover their traces and scrub server logs of their activities. A month later they had obtained information about Yahoo's own, what they call the AMT, the Account Management Tool, which Yahoo administrators use to manage and modify information about accounts, usernames, recovery email addresses, phone numbers, security questions and answers, and more. So essentially a database of this stuff, the Yahoo user database, known as UDB, the User Database.

A month later, in November, they managed to acquire a backup copy of the entire Yahoo user database, in November of 2014. That contained information for more than 500 million Yahoo accounts, which gave them access to the account of any user whose password had not been changed since that backup snapshot had been made. Then, throughout the following two years, through 2015 and 2016, the attackers leveraged their access to this AMT, the Account Management Tool, and the information contained in that stolen UDB database snapshot, to target user accounts of interest.

And in instances where people, for example, government officials had Yahoo accounts and then also government accounts, the crossover was leveraged in order - again, using social information - in order to compromise those users. Once they got into their Yahoo accounts, for example, if Yahoo was being used for account recovery on any other accounts, where it would be like, oh, I forgot my password, send it to my Yahoo email account, well, it would come into Yahoo email, and these guys who had access to those accounts would be able to get the account recovery information.

So then, as if that wasn't enough, I mean, this was a deep penetration. They even figured out how to mint their own session cookies to bypass the need even for username and account credentials. Remember that, as we know, that's sort of the ultimate win is a session hijack. When a user's logging in, they authenticate their identity, and then the server provides them with a session cookie, which the browser continues to send back during the session, which is how you get that stateful login, how you stay logged in.

Well, if you figure out how to create your own session cookies, you don't need the user's password. You simply give yourself a cookie to a logged-in session for that user. They figured out how to do that. So that then gave them complete, you know, the ability just to spontaneously be logged in to anyone's account who they wanted to. And then they could, of course, read through that person's email history and obtain any secrets that they needed to. So initially they were using Yahoo's own servers to mint the cookies sort of in vivo. Then they did it in vitro. They figured out how to take that code and run it on their own machines and just mint cookies with abandon.

So the indictment alleges that the attackers used these cookies to access the contents of more than 6,500 Yahoo user accounts, and then of course to later steal contact information from as many as 30 million Yahoo accounts as part of a spam marketing campaign. The FBI's indictment also notes that the intrusion these guys got was powerful enough to allow one of the attackers, and actually it was this Canadian guy, to modify the results returned from Yahoo searches for a specific erectile dysfunction drug so as to refer to a specific online pharmacy for which the attacker would obtain a referral fee.

Leo: That's just good business.

Steve: And throughout all of this, the FBI has found that Russian intelligence officials proactively assisted the attackers in hiding their Internet traffic and remaining undetected. So this was all done with knowledge and sanction with the Russian government and its intelligence arm. So now we know a little bit more about what went

on behind the scenes. And basically an early major web-based email company, as we know from coverage in the news, was deliberately shorting the security pleas of their own security people, downplaying the need for heightened security, up-playing the need for more features. And this is the consequence of that.

Leo: That goes on in every company; you know? I mean, that's the problem; right?

Steve: It's a battle. Yeah, the security people are running around, you know, screaming "wolf" with their hair on fire.

Leo: And marketing goes, "You can't do that. Nobody would use our product."

Steve: Exactly. So I got a couple tweets from people regarding eBay "upgrading," in quotes, "a higher level of security" by retiring the football. And in fact I got a screenshot of the email that was sent. "We're making two-step verification more convenient," is what they said. And this was someone, Barry. It says: "Hi, Barry. Thanks for using two-step verification when you sign into eBay. With your help, we've been able to give your account a higher level of security." Uh-huh.

"We're going to make two-step verification more convenient by texting you a PIN instead of having you use your token. All you need is a mobile device." Of course, you didn't need that if you had the football, but now you do. "If you use our app, you'll need the latest version of that, too. Switching to the new two-step verification is as easy as grabbing your mobile device. Why not do it now?"

Leo: I'm willing to bet this is timed exactly to coincide with the battery life of those football dongles; right? Right? They don't want to replace the football dongle. I got - that was the first football I got. It was for PayPal, but it's eBay PayPal; right?

Steve: Yeah. Well, now, what my guess is, is that they just decided - certainly they may have been dying. They may not have wanted to replace them. Remember, though, that they were also using a third-party service. They were using that early VeriSign service.

Leo: Oh, yeah, yeah.

Steve: And it cost them every single time they did that.

Leo: Oh.

Steve: Yes.

Leo: That's dopey.

Steve: Yes. And that's why the whole thing, the idea of sending this out to a third-party service, that was in the early days.

Leo: They had to, probably; right? I mean, or I guess, I don't know.

Steve: Yeah. Well, no. Now, so that's the TOTP system, the time-based one-time password. And of course our listeners sent this all to me because I had mentioned how I had set myself up with Hover to use their system. And they gave me the option, send a PIN over text or use the time-based one-time password. And of course we know that sending a PIN over text is not secure because cell traffic is not secure. So it's not more secure; it's less secure. The beauty of using a time-based one-time password is that, when you set it up, you're given a QR code. You scan that into your authenticator app, and now it's able to autonomously generate a 30-second duration six-digit code which changes.

I did want to take this opportunity to note that because now I'm using it for LastPass, for Hover, and for Gmail, or just in general for Google, my Google identity. Not one of those apps will export the token information after it's been imported, which is annoying if, like, you use multiple apps. So what I have taken to do, and this is what I would recommend, is when you set this up with a company, or if this has been a problem for you, you want to reset it up - and I should say that these apps are unwilling to export it, which is probably a good thing because that would be a security vulnerability. If they allowed you to export the secret key, which is what generates the non-predictable time-based six-digit token, then a bad guy could get those exports and then know your sequence from ever on.

So it's good that they're being secure. The problem is you could argue there's a threshold where it becomes less secure, if it's so inconvenient for you because you don't have access to your authenticator app, that it causes more problem than it's worth. So the solution is print those QR codes and keep them. Which is what I've started to do. So, for example, when each of those websites said, here's the QR code for your time-based one-time token, I printed that on hard copy and then said, okay, got it. And now, if I'm setting up a new device, I choose whatever authenticator I like. And I like Google's authenticator best, only because it gives, you know, lots of people make them now. There's Auth. There's LastPass has one. Google has them. I like Google's, only because it's more compact because, as I use this more and more, it's easier to see more of them on the screen. It doesn't have to be huge.

So the idea is that I now keep a set of these printed QR codes. Obviously I store them securely and safely because I don't want any bad guys to get them. I also keep them in the same order because I want to show them to the authenticator app in the same order so that it's Hover, Google, LastPass on all of them. That way that gets to be a habit, too, which is reinforced, in the same way that I always have the keys on my key ring in the same order. I'm a little strange that way.

Leo: I just do a screenshot of it, Steve, and put it in my LastPass, and then you have the QR code there, and you can just aim it at that. But actually I stopped doing that because I now use Authy, which stores the secret in a database, an encrypted database. Now, it's a little less secure because now there's a central encrypted database. But I use a long secure password to protect it, and I feel fairly confident that it's secure. And the nice thing about Authy is you log into your Authy account

that uses telephone verification or SMS verification or email to log into it on any device. It already knows all the secret numbers, so it populates it automatically. And then before you use it, you encrypt it. It also has a per-device encryption, if you want to use that. So I feel like it's fairly - see, I don't know if you've looked at Authy. But to me that's the easiest way to do it.

Steve: Right. And that's certainly a good choice.

Leo: It's a little less secure because there's a database online of your [crosstalk].

Steve: If you don't mind having a third party having all your tokens.

Leo: Right, right. But I just, you know, you don't need to print it out. Just take a picture of it and put it in your LastPass. Then you'd have it everywhere.

Steve: I'd just rather have it, again, you know, I have a car with no Internet connection, which I prefer. And I just have hard copies.

Leo: Yeah. It's something I had, you know, we may be unusual. But, I mean, I'm using a new phone every three or four weeks. So I'm always setting up Authenticator; right? So Authy for me is about the only way to do it.

Steve: Yeah, I agree. In that case, that's more secure because otherwise you would be disinclined from using that level of authentication. And I have to say, where we are at this point in the world, this use of a time-based token is the best we can do to augment a username and password. As everybody knows, I'm working on SQRL, which may or may not gain traction. We'll see. And then of course there is the FIDO effort also. But one way or another, until we get away from usernames and passwords, a time-based token is the solution that makes the most sense.

Leo: Even NIST has deprecated SMS tokens. Even the National Institute for Standards and Technology says don't use SMS.

Steve: It's bad.

Leo: I think it's more because of spoofing than anything else. It's easy enough to spoof somebody's SIM card, or even just call the carrier and say, "I lost my phone. Can you send me a new SIM card?" "Well, who are you?" "Well, I'm who I say I am." Okay, it's on its way. It's that simple.

Steve: So we're at about the halfway point. Our next topic will be the most insane, you cannot make this up if you tried...

Leo: Steve, you're such a tease.

Steve: ...IoT thing. And believe me, I am not overstating it.

Leo: All right. Steve is caffeinated and ready for Part 2.

Steve: Okay.

Leo: Uh-oh.

Steve: I am not making this up.

Leo: He's serious. He's tenting his hands, folks. This is serious.

Steve: A company named Standard Innovations - and really there's nothing standard about this innovation.

Leo: Terrible name.

Steve: [Standard Innovations], in Canada, manufactures and sells the We-Vibe.

Leo: Oh, yeah.

Steve: A Bluetooth-enabled...

Leo: I got one of these, by the way.

Steve: Okay. A Bluetooth-enabled, remotely controlled vibrator.

Leo: For review purposes only, I want to emphasize.

Steve: And it's smaller than I thought it would be. So I guess size doesn't matter. It itself has no user interface, though some might argue that the entire device is itself a user interface. In order to be controlled, it's paired with its companion We-Connect app, which allows its user to vary rhythms, patterns, and settings. Or, and get this, to give a partner in the same location or anywhere in the world control of the device.

Leo: Again, for research purposes only. Okay.

Steve: Yes. There is a promotional YouTube video...

Leo: No, no.

Steve: Which is NSFW. You don't want to let the kids see the opening scene.

Leo: Oh, how funny.

Steve: It's a line diagram. After that it's okay. But customers have alleged and complained that the manufacturer was - get this - secretly tracking their use. And after a class action lawsuit was filed, the manufacturer recently reached a $3.75 million class action settlement with its users, following their allegations that the company was collecting data on when and how and with what settings the toy was used.

Leo: Well, that's research.

Steve: Some would not consider this a toy.

Leo: That's research.

Steve: And their slogan, of course, is "Play together, even when you're apart."

Leo: Yeah.

Steve: Now, this is leveraging the Internet of Things in a new and creative way. The lawsuit was filed in federal court in Illinois last September, alleging that without customers' knowledge, the app was designed to collect information about how often and with what settings the vibrator was used. Lawyers for the anonymous plaintiff group contended that the app collected users' email addresses to allow the company to, quote, "link the usage information to specific customer accounts." I mean, there are just so many things I could say, but I won't. The suit alleges that customers' email addresses and usage data were transmitted to the company's Canadian servers.

When a We-Vibe was remotely linked to a partner, the connection was described as "secure." But information was routed through We-Connect and collected by the Canadian servers. The unhappy users allege in their lawsuit that they never agreed to the collection of this data. Standard innovations maintains that users, quote, "consented to the conduct alleged." And of course, you know, maybe in the fine print. Who knows? But this is not what people were expecting.

But instead of taking the case to court, the company agreed to settle. According to court

documents, an estimated 300,000 people purchased one of these Bluetooth-enabled We-Vibes. And under the terms of the settlement, anyone who bought - and Leo, you could cash in here. Leo, anyone who bought an app-enabled vibrator can receive up to $199. However, anyone who actually connected it to the app...

Leo: Which was, oddly, a small fraction of the total buyers.

Steve: I know, it was. Of the 300,000 who purchased it, only 100,000 connected it to the app.

Leo: Seems like that was the point of the whole thing.

Steve: Yeah. I found that surprising, too. The good news is those who did, who did connect it to the app, can collect up to $10,000. So there's a cash proposition. The actual amount paid out will depend on how many people file claims. So I guess the settlement amount is fixed.

Leo: The lawyers make the money, is who makes the money, yeah.

Steve: Yes, yes.

Leo: I think this is a little BS-y. Because, I mean, really, of course you had an account. I don't know why you had an account, but you created an account so you could use the vibrator. I guess so they could know which vibrator to control when you're across the country. You'd have to identify it. But what do they care how you - I understand why people are upset. But wouldn't you think they were really collecting information? What would they do with it? I mean, who cares; right? "Oh, my god, they're having sex." They're going to sell ads based on that? Maybe you're going to get those Yahoo emails.

Steve: Well, it does sort of bring new meaning to the term "invasion of privacy."

Leo: Well, I agree. And I agree, they probably shouldn't have done that. But I don't think there was any maliciousness intent. But anyway.

Steve: Yeah. They probably want to know how long the battery lasts. So you're collecting a little bit of, you know...

Leo: You know, lots of other products do that. I think if it weren't a sex toy, it might have been, I mean, that makes it - now it's very loaded; right?

Steve: That's more salacious, yes.

Leo: Yeah, yeah.

Steve: So I will say, though, on the other hand, the fact that it is, and people feel very private about their sex lives, this was, you know...

Leo: Right.

Steve: Yeah. I don't think...

Leo: No, I understand. And they knew, too. That's why they paid up right away.

Steve: Yeah. So I found this interesting. Gizmodo...

Leo: Does give a new meaning to "pen testing."

Steve: Oh, boy.

Leo: I'm sorry I said that.

Steve: Okay. Yeah. Hopefully people don't know what pen testing...

Leo: Steve is turning red. I've never seen you turn red. This is a first, ladies and gentlemen.

Steve: Hopefully people - I think that's just the color on the webcam. Hopefully people don't know what pen testing is an abbreviation for. So, wow, that's good.

Leo: Moving along. Yeah, good coffee, yeah.

Steve: Yeah. So Gizmodo had the headline, "You Don't Really Need an Anti-Virus App Anymore." And I thought this was significant. I mean, this is what we've been talking about and saying now for some time. Significant, though, that this concept is beginning to enter the mainstream, not just the security group and/or just this podcast, but it's getting some air. Gizmodo just wrote, they said: "Ten years ago, the first thing you needed to load on a brand new computer were antivirus and malware applications. The Internet was a minefield of malicious content that could infect your entire home network with one errant click. Yet things have changed dramatically. Windows has much more robust security built in. Browsers are smarter; and, hopefully, so are users.

"Instead of spending your money on a lot of security software that is often as invasive and irritating as the malware it protects against, think long and hard about going with a

minimalist security setup and practicing safe browsing. Below are the scant few software packages you really need, and where to find them." And I won't go into detail. But basically it captures what we have been saying now, here, for some time, that as we know, antivirus is now often increasing the attack surface by interposing less securely written and tested third-party, and I coined a term as I was writing the notes, "shimware."

And of course we know that browsers now, and OSes, are being constantly updated, routinely and proactively, in order to keep them current. And that many OSes now provide their own built-in AV and firewall software and are incorporating their own proactive technologies like ASLR and DEP, Data Execution Prevention and Address Space Layout Randomization, to essentially harden the OS itself against these attacks. So it's like many of these technologies have ended up percolating down into the underlying OS.

So again, it's nice to see that this is beginning to go mainstream, this idea that, eh, you know, it's no longer something that you need to worry about to that same degree. And right on the heels of that, US-CERT has issued an advisory alert with the title "HTTPS Interception Weakens TLS Security." This, of course, also following on what we have been discussing. Their advisory says: "Many organizations use HTTPS interception products for several purposes, including detecting malware that uses HTTPS connections to malicious servers. The CERT Coordination Center explored the tradeoffs of using HTTPS interception in a blog post called 'The Risks of SSL Inspection.'"

And essentially, again, it restates some of what we've been talking about recently, that we've seen instances where, for example, in not properly validating certificates and in weakening the TLS connection by not fully supporting TLS to the same degree that the client on its end and the server on the other end do, this proxy in the middle ends up creating explicit security problems and breaking the guarantees that TLS was put there to provide. So the point is that there is a significant degree of responsibility that comes with a device which is going to interpose itself for the purpose of inspecting the contents of encrypted communications. And what we're seeing is these devices are not yet meeting that responsibility as well as they should.

I got a tweet from someone that reminded me about a question that often comes up relative to using SpinRite with USB devices. I hope this is not his condition. His Twitter name is TommyParalyzed. So Tommy, I hope you're not. Or maybe it's just with fear and not with a physical disability. He said: "Hey, Steve. Have a USB 500GB HDD. Do I need a USB-to-SATA converter to plug directly into the motherboard for SpinRite to recognize it? Thanks." And the answer is no. But this is a constant source of confusion.

At some point, and I don't know where, it won't be in 6.1, SpinRite 6.1, the next forthcoming release, because I'm going to rush that out to immediately update it to much higher performance in order to allow it to deal with today's much larger drives in a reasonable speed. I'm going to, essentially, because it's taken me longer than I expected because of the SQRL project interceding, I'm going to get it done more quickly. I had always planned to explicitly support USB with a 6 point something after one. So that will happen.

But today, as many people know, you do not need to wait. However, you do have to hook the USB drive up when you're booting the system to run SpinRite. The BIOS looks for connected USB drives when it boots, but it does not see them afterwards. So the mistake that some people make is they boot their system up, get it running, then they connect the drive, and SpinRite is unable to see it. In the future, it will be able to enumerate the USB bus, and it will be doing so constantly in order to find new attach announcements of drives. It cannot do that today. What it can do, though, is to rely on

the BIOS to do that at boot time. So connect your USB drive, have it up and running, boot the system. Then SpinRite will see it and will run on it just fine.

Okay. We talked a couple months ago about the forthcoming Pwn2Own competition and mentioned that it would be the 10th anniversary and would be offering higher, bigger prizes, more money than ever before. And as they do every year, the competition order was decided by random drawing in the contest room on the first of the three days of competition. This year's event featured 11 teams of contestants targeting products across four different categories, 30 different hacking attempts in total. Each contestant was given three attempts within their allotted time to demonstrate their exploit. And, boy, if anything ever shows us the true - is "porosity" a word? Porousness? Porosity.

Leo: Porosity is right, yeah.

Steve: Porosity, the true porosity of security, it's the Pwn2Own competition. And by upping the stakes, everything was just cut through like a hot knife through butter. The 360 Security Team used a JPEG 2000, which of course is an image format, heap overflow in Adobe Reader. Get this. So a heap overflow in Adobe Reader, a Windows kernel info leak, a remote code execution through an uninitialized buffer in the Windows kernel. That is all three in conjunction to take down Adobe Reader. In the process they earned $50,000 and six points toward the Master of Pwn award for the entire three-day competition.

Two researchers targeted Apple Safari with an escalation of root on macOS. In what was considered a partial success, Sam Gross and Niklas Baumstark earned some points for style by leaving a special message in the touch bar of the Mac. They used a use after free, a so-called UAF in Safari, combined with three logic bugs and a null pointer dereference to exploit Safari and elevate to root privilege in macOS, earning themselves $28,000 and nine Master of Pwn points.

Tencent Security targeted Microsoft Edge, and they successfully exploited Microsoft Edge through an arbitrary write in ChakraCore. They used a logic bug to escape the sandbox and earned $80,000 and 10 points toward the Master of Pwn.

Chaitin Security Research Lab attacked Ubuntu Linux, welcoming Ubuntu Linux to Pwn2Own with a Linux kernel heap out-of-bounds access, earning themselves $15,000 and three Master of Pwn points.

Then later in that first day - we're still on Day One - Tencent Security's Team Sniper used an info leak in Adobe Reader, followed by a use after free, to get code execution. They then leveraged another use after free in the kernel to gain system-level privileges, winning $25,000 and six Master of Pwn points.

Chaitin Security Research Lab came back targeting Apple Safari again with an escalation to root, a different one, in macOS, successfully exploiting Apple Safari to gain root access on macOS by using a total of six bugs in succession in their exploit chain, including an info disclosure in Safari, four different type confusion bugs in the browser, and a use after free in the macOS WindowServer. This earned them $35,000 and 11 points toward the Master of Pwn.

360 Security targeted Adobe Flash with a system-level escalation and a virtual machine escape. They elevated to system privilege using four bugs. They were unable to complete the VMware escape bonus portion, but what they demonstrated constituted a win and

netted them $40,000 and 12 Master of Pwn points.

Tencent Security's Team Sniper came back again, successfully exploiting an Adobe Flash UAF, a use after free, escalated to system, and then a UAF in the Windows kernel for $40,000 and 12 points. A different team at Tencent Security, the Lance Team, successful exploited Microsoft Edge with a use after free in the ChakraCore that elevated to system and used another use after free in the Windows kernel to earn themselves $55,000 and 13 Master of Pwn points. And this goes on and on and on.

Tencent Security, 360, Tencent, 360, 360, Chaitin Security two more times. Another one by Tencent, 360, I mean, so basically just a ton of collapses in Safari, in macOS. In Firefox there was an integer overflow. There was another problem in macOS with WindowServer, another exploit in Edge. And, finally, the Sniper Team at Tencent Security used a three-bug chain to win the Virtual Machines Escape, from guest-to-host category, with a VMware workstation exploit. They used a Windows kernel use after free, a VMware information leak, and an uninitialized VMware buffer to go from guest to host and win themselves $100,000 and 13 points for the Master of Pwn.

So this just shows us, I mean, I didn't even count these: one, two, three, four, five, six, seven, eight, nine, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21. Twenty-one different exploits against browsers and OSes and VMs, all of these unknown, all brand new. And they ended up taking home a whole lot of money as a consequence of their research.

Leo: The VM escape seems the most worrying to me because so many people use virtual machines as a way of securely computing because they feel like they're isolated inside that machine. The fact that you can escape, I mean, first of all, amazing bit of work.

Steve: Oh, my lord.

Leo: Really amazing bit of work.

Steve: And exploit chains. Not just one, but, like, a chain of six exploits in series, yes.

Leo: It's incredible, yeah. But that's dramatic. I can't remember a previous VMware escape. In fact, I think that that was always the holy grail of Pwn2Own was a virtual machine escape. So that's a big deal.

Steve: Yeah. The last one was that floppy disk driver bug, remember, that there was the floppy disk driver than nobody was even using anymore.

Leo: Oh, I remember that, yes, yes.

Steve: But it was still there. And there was, like, there was a state machine in the floppy driver that they were able to use in order to get out into the - back out into the kernel.

Leo: Yeah. You've got a better memory than I do. I'd forgotten about that.

Steve: Well, I focus on this stuff, so.

Leo: It's kind of your thing.

Steve: Yeah. Yeah, so, wow. So just like that, as a perfect, you know, where we are, this demonstrates that, much as we want this stuff to be secure, and it is to a degree, if somebody is sufficiently motivated, and we know who is, those who have a strong state sponsor behind them on any side of any ocean, who want to get into users' computers, these are very useful tools. And we're in a cat-and-mouse battle to have these be as secure as they can be. But the problem is they are just - they've become so complex that you cannot say that they're absolutely secure.

And now there are instances where there isn't even an effort at security. Which brings us to Oil and Gas International. And Leo, you should - it's OilandGasInternational.com is the site. And click on the Login option there, and you'll be taken to a non-HTTPS login page. This came to the news in the last week because the administrators behind Oil and Gas International...

Leo: I think they've pulled the site down because nothing's coming up.

Steve: Yeah.

Leo: I think they must have heard about this.

Steve: So the problem was that they complained, the Oil and Gas International people complained to Mozilla that Firefox was saying their page was not secure. And it wasn't. I mean, there wasn't even an effort at security. Not only - and I tried to use their page with HTTPS, thinking, well, maybe they have HTTPS, but if you don't use it, then you don't go there. No. There's no HTTPS option. They have no certificate. Port 443 is not open to the world. You cannot get an SSL/TLS connection, even if you wanted to. So on this non-secured page, they offer you to log in.

Now, as we know, if you do, not only are your username and password going to them in the clear because there's no security, but in then being logged in, your browser is sending every subsequent query, along with a cookie to keep you logged in, that anyone looking at your unencrypted traffic can see. So this is the Firesheep problem that we dispatched with years ago. These guys are like, oh, you know. Now, you could say, okay, you don't care about security. You don't actually care about logging in. Now, there's more here because, on that form, the login page, very conveniently it says - it asks the question, "Forgot your password? Click here to receive it via email."

So, yes, it's also one of those sites which is not hashing your password. Of course they're not because they don't care about security. They're going to send you your password in email, if you click on the link, so that you're able to then log into this why-even-bother-having-a-login, crazily insecure site.

Leo: It's not like they're selling anything. It's just a news site for oil and gas.

Steve: Yes, exactly. So the mistake they made is in complaining to Mozilla that Firefox was correctly saying you're asking people to log in on a site which offers no login security. That's what Mozilla was saying. And so their users who were using Firefox saw that it's now a black padlock with a red slash through it. And then there's a black "I" to the left of that where you click on it, and it says, "This page is not secure." And so people complained to this Oil and Gas International, who then in turn complained to Mozilla. And Mozilla said, uh, sorry, that's a feature, not a bug.

Leo: Somebody's passed along a Tumblr website called Plain Text Offenders which lists sites that will send you your password back in plaintext. So this is apparently one of many, a great many, a very great many.

Steve: And again, although no one is saying that they have to be secure. It's like...

Leo: Right [crosstalk].

Steve: Right. And I would argue that this is exactly what Firefox and Chrome should be doing.

Leo: Yes, yes.

Steve: I mean, this is it. It's like, okay.

Leo: Use at your own risk.

Steve: Yeah. You can put your username and password in. But, boy, this is a password you really don't want to use anywhere else.

Leo: Yeah, right.

Steve: Yikes. So Tavis has been busy again. He was looking through the LastPass web browser code and used something called "beautify" which apparently is the reverse of "minify." It probably deminifies and then makes it more legible. Maybe it should be called "legify." Anyway, it's called "beautify." And he noticed LastPass was using a particular domain name, a subdomain, or a machine in a subdomain, the numeral 1min, 1min-ui-prod.service.lastpass.com.

And the problem was the way that was being used gave it some privileges such that, with two lines of JavaScript, a user could user that domain to create a window which was privileged to issue remote procedure calls, that is, like subroutine calls, to the subroutines that existed, the hundreds of subroutines that existed within the LastPass

browser extension. Which among other things do things like filling in passwords and filling in forms and copying passwords to the clipboard and so forth. So that was a bad problem.

He tweeted yesterday in the afternoon: "Oops, new LastPass bug that affects 4.1.42 on Chrome and Firefox," which is the current newer version of the LastPass web client. He says: "RCE [remote code execution], if you use the binary component, otherwise can steal passwords. Full report on the way."

He also a little bit later tweeted: "I have a full exploit working without any prompts on Windows. Could be made to work on other platforms. Sent details to LastPass." And then I have a link in the show notes to his posting in the Chromium bug Project Zero blog where he explains this, the "no popup" exploit. I think it was like four lines of code. LastPass immediately responded by killing off the DNS for that URL so that it would no longer function. So they instantly neutered it globally and have a longer term fix on the way. So they immediately fixed it.

And as you can see in the Chromium blog, in the Project Zero blog, his little blow-by-blow, that they fixed it, and then he went public with the news because immediately, since it was a real-time DNS request that would have to be looked up, killing off that domain name resolution would immediately close the hole. So again, this is why this has to be kept legal, to allow security researchers to put eyes on, smart eyeballs on this kind of work, and then have the parties that are responsible respond as quickly as they can. So bravo to everybody involved.

I did get a nice tweet from Lucas, who is at the U.S. Naval Academy. Remember that they're the guys who did the research on the lack of sufficiently robust randomization, and in some cases completely bypassable randomization, in WiFi passwords, which our devices are primarily Android, but even iOS is a little bit better. But it, too, can be exploited to create a trackable privacy breach.

So what I first saw was a tweet from Lucas to @SGgrc: "Thanks for your coverage of our research about MAC address randomization. You provided the best summary over other media outlets." And so I wrote back, and I said: "Thanks very much for your note. I'm glad I got the details right. Your work provided a perfect real-world platform for helping to explain the details of what's going on and why this stuff is so difficult to really get right. And as I said on the podcast, more importantly, your work will doubtless help to drive the improvements our industry needs."

And then he responded: "I want to add that I started to listen to your podcast about a year ago and dreamed of having something that made the show. Furthermore, when we published..."

Leo: Aw, that's neat.

Steve: ..."I said to one of my professors, 'How cool would it be if Steve Gibson picked this up on Security Now!?' We were both ecstatic when we got caught up on episodes and heard the mention of our research. Thanks again for your coverage."

Leo: That is nice.

Steve: So, very cool. Happy to be helping to spread the word of great research being done. And it's what we need.

So I got a tweet from someone who calls himself RealBrainTraining. He said: "If an encrypted file is open when my computer enters sleep, the file is open when it wakes." He says: "Key in RAM is written to drive. Security risk?"

Okay. So, first of all, hopefully the hibernation file is encrypted. We do know that the earlier versions, for example, of TrueCrypt were not doing that, but later they were. And it's definitely the case that you want the whole drive encryption to encrypt your hibernation or your contents of RAM. Now, maybe if it's just an encrypted file and not the whole system, then it may well be, if you do not have hibernation encrypted, that the key is being written to the drive.

So I would say it's important to understand whether your particular system encrypts the hibernation file with the RAM contents because you're absolutely right that there is an active key in RAM being used to allow you to see the contents of that encryption. It's also important to go a little bit further, to again recognize what I would call "security perimeters." That is, if you have something decrypted on your system, while it is decrypted, there's a dynamic vulnerability there.

For example, I recognize that in the design of my SQRL client on Windows. It uses many different triggers like initiating hibernation or the screen blanker going on or switching users or allowing the user to explicitly do it, an array of triggers, all which wipe, immediately wipe the contents of a secure region which is locked in memory, cannot be swapped out, and which I set up sort of like a sandbox. I set it up on day one, and every single piece of security-critical information is kept in there so that it's not spread around through the code. It's in one location, and that means I can trivially wipe it and know that anything that is sensitive is gone.

And so what you really want is you want products which were written from the beginning, especially when they're security related, with these kinds of problems in mind. But also to recognize that dynamically, when something is decrypted, while it's available, there's a decryption key in RAM. And if something is in your system, then it's inside your security perimeter and potentially has access to it. And we've covered in the past how distressingly clever the bad guys have become about finding these keys in memory. You might just think, oh, it's just some 128 bits, you know, 32 bytes sitting around in memory, out of gigabytes. How could that possibly be found? The problem is it's typically surrounded by metadata, and it can be recognized. I mean, we see many instances of it actually - of 32 bytes out of billions being found with some reliability.

I had a couple questions, a couple people asking about Conway's Game of Life, asking me what version I play. Several people said several versions are available in the App Store, and of course several are also available in the Google Play Store. So I just wanted to mention that I haven't played with it at all. I really got it out of my system back in the '70s when it was new. I was working with mini computers. I wrote several instances. There was a lot of interesting work done on how to make the Game of Life go fast on machines of the time. So those were sort of the challenges that we had back then. And I just haven't bothered, I mean, I know it so well that I didn't bother to play with it recently.

But let me encourage anyone who has been intrigued by our mentions of it in the last couple weeks to download - the apps are free. There are a bunch of them on all different platforms everywhere. You can get them web-based, Windows and Mac native, and also mobile. Just, you know, it really is an interesting intellectual endeavor to sort of play with

it and see what they do. The charm, I think, is that it is so simple. So few rules guide the evolution of this little array, this little grid of cells, and yet it ends up producing such rich results.

And I did have someone ask about purchasing a PC from eBay. He asks: "Could the BIOS have malware after wiping the drive?" And can he scan? And that's really a good question. We're in a place where I would argue, you know, we were talking about the idea of purchasing a smartphone from eBay, and that it could have preinstalled malware in the ROM that could bite you, I mean, could have ransomware that could come out and bite you sometime later, or give a bad guy a starting point to attack you. How would you know? We know that there is BIOS-infecting malware. Yet we're at this weird stage where there isn't yet any, that I'm aware of, useful BIOS scanning antimalware. The only thing I could suggest you could do is, in the same way that you wipe a drive, would be to wipe the firmware. That is, find the latest version of firmware and reflash the BIOS on a machine that you're getting from eBay. You'd want the latest BIOS anyway. And that would be the best you could do.

I don't know that - the problem is the UEFI BIOSes now, they're an operating system unto themselves, with their own file system and with modularity. So you might want to do a little research to verify that reflashing the BIOS really does rewrite everything. The problem is there might be some portion of it that you just can't get to. I don't know whether that means that it could not be infected. But I'd be concerned. I mean, it's an interesting problem. We know that firmware malware - firmware malware. There's got to be a way to shorten that.

Leo: I like firware. Fralware.

Steve: Yeah, fralware.

Leo: But it can also live in video cards, which you can't easily rewrite; right?

Steve: Right, right.

Leo: We've seen video card attacks.

Steve: Right. I mean, we're in a sorry place at the moment. And I don't think it's going to get any better.

And I finally got a tweet from someone in The Netherlands who says: "In the Netherlands we seem to be getting more and more need" - that's what I wrote, I'm not sure what he meant - "not yet in politics so much, but amongst the people, for more and easier ways to electronically vote." Oh, more and more need, I see, "not yet in politics so much, but amongst the people, for more and more easier ways to electronically vote reliably, securely, anonymously. Couldn't something like SQRL be modified to that end?"

And one of the things that I've mentioned before, and this sort of put me in mind of that, is that even though SQRL is designed for two-party authentication, that is, no man in the middle, no requirement for a third party, it does support it, and it is secure. So it could absolutely be used in a very robust way for a government to manage and control voting

because you could go to a voting site, and the SQRL URL doesn't authenticate you to the site. It authenticates you to a third party, which then authenticates you to the site. So even though it can be anonymous, there are mechanisms available where you could have your SQRL identity, or a SQRL identity, bound to a nation, to a state that knows you that way, and then could allow you additional flexibility in a secure fashion. So it remains to be seen how much of those capabilities actually occur.

Oh, and a piece of errata. Many people noted last week when I was talking about, in fact, I was talking about Ethernet in the context of the U.S. Naval Academy WiFi MAC Randomization failures, about the invention of the Ethernet, and how, back then, how quaint it was that 48-bit MACs were like, oh, that's all we're going to ever need. We'll give 24 to the manufacturer and 24 to the device of the manufacturer, concatenated to create a unique token. And I referred to Bob Metcalfe as the inventor of the Ethernet, as he was when he was at Xerox PARC, and then when on to form, and I said 3M. I meant 3COM.

Leo: Of course you did, and I apologize for not stopping you.

Steve: Yup. I just - my best friend works at 3M. He's in the technical side, doing stuff with various of 3M's technical products.

Leo: Adhesives.

Steve: And so I just said 3M. I'm talking to him about 3M. It's on my brain all the time. So it just came out. I didn't even hear myself say 3COM.

Leo: 3COM, of course.

Steve: So stick a CO between the 3 and the M, and you get that right. Also, I did want to remind people about SquareIt.io. That's the cute little puzzle toy that I mentioned some months ago. Several people have tweeted, saying, "Darn it, Steve, you've just sucked up all my free time." It's really, I just wanted to say, I have continued to enjoy it. It's not difficult. It's very clever. And it's an endless game. You never run out. It was written by one of the people who did a different one that we talked about in the past. I think it was the Infinite Loop guy. And it's just - it is really nice.

So SquareIt.io. And it's available both for iOS and Android. And check it out, if you haven't. And if you just sort of like having a doodle, it's just sort of a nice doodle, but very well done. And I wanted to mention also that my favorite show on Syfy, that just stands out apart from everything else on Syfy, "The Expanse," has got its third season. And, oh, boy. I read all the books, and I'm up to date on where we are. There's some amazing stuff about to happen, probably tomorrow evening, or maybe the week after. So it's looking great.

And lastly, a very nice piece, thoughtful, I thought, published on Medium.com by a guy named Steve Meyers, who focuses on PHP and MySQL. He gave a presentation on - he focuses on PHP and MySQL design and scalability. He's with the Utah Open Source Foundation. He spoke at O'Reilly's OSCON on the topic of database optimization for web designers. His little short bio says: "Steve Meyers has worked as a PHP and MySQL

scalability expert for the past 15 years at such companies as Omniture, now part of Adobe; Spark Networks, owner of JDate; and CrimeReports. He now runs some of the largest independent online communities of college sports fans. When he's not too busy with all of that, Steve runs the Ski PHP Conference, assists with the OpenWest Conference, is a core team member of the Utah Open Source Foundation, and runs the Provo Linux User Group." So he's active. He knows what he's talking about. He's in the middle of all this.

Very nice piece of writing he posted, titled: "How to fix the problem with malicious and misbehaving ads?" He wrote: "In ad circles, I'm known as a 'publisher.' In simple terms, that means that I serve ads on my site, and the ad networks pay me for serving their ads. This is a wonderful thing. In 2011 I was able to quit my day job in order to focus on my website. I make enough money now that I don't need a 'real job' to pay the bills. I love that advertisers made it possible for me to work for myself.

"I also hate advertisers. Unlike many publisher sites, my users tend to stay for a while. They just don't show up from a Google search and then go their merry way. Many of them view 50-plus pages per day. I've set up my AdSense settings so that I will inflict the least annoyance possible on my users. This includes not allowing ads that autoplay video with audio, ads that expand past their allotted size, ads that pop up, ads that pop under, or anything else likely to annoy my users. If the ads get bad enough, my users won't come back, and more likely they'll turn on an ad blocker.

"Advertisers and ad networks don't care about any of that. They're just out to make sure their ads get the most visibility. I use Google to serve most of my ads, and they make it difficult for me to block ads that are misbehaving. They have tools that supposedly help with finding and blocking ads, but the tools are rarely useful. For the last two months I've been trying to block a rash of ads that are driving my users away. They autoplay video" - and Leo, I've heard you complain about that so much.

Leo: Drives me nuts. There's a great Chrome extension to prevent that, but still drives me nuts.

Steve: "They autoplay video with audio, which is bad enough. But in addition to that they force the focus of the window to the ad."

Leo: Yeah.

Steve: "If you try to scroll past the ad, they scroll back. If you try to click on a form input, they refocus on the ad."

Leo: Oh, that's terrible.

Steve: So you can't even enter content in a form.

Leo: Oh, that's terrible.

Steve: "Since my site is a message board, that's kind of a problem, as it keeps my users from being able to post messages. You may think that only shady brands would resort to using ads that are so blatantly misbehaving. The ads I've been trying to block are from Ford, Arby's, and Whataburger. These are national brands, not no-name companies that don't know any better. And those are 'respectable' ads. I haven't even discussed malvertising yet. Some ads are trying to load viruses onto your computer. Others will redirect you to the App Store, or another website. With all this poorly behaved advertising being sent to our users, is it any wonder many of them resort to ad blockers?

"So, how do we fix this? The answer is actually pretty simple: browser-enforced content restrictions. Imagine that, as a publisher, you could specify that the ads on your page would be unable to do certain things. For example, you could specify that audio, pop-ups, expanders, and redirects are not allowed in a certain DIV tag. The browser would ensure that any script loaded from that DIV tag would have those capabilities disabled. Alternatively, the permissions could be denied on the SCRIPT tag. That would have the same effect. In order for this to work, it would need to specify what behaviors are not allowed, rather than what is allowed. This would allow for additional behavior to be supported in the future.

"Do the browsers have any motivation to do this? Google's main business is advertising, and they're the dominant browser on the market, so perhaps not. However, I think it makes sense for them to support a standard like this because it's in their interest to keep people from using ad blockers. Publishers need to be able to control their reputation. If we want advertising revenue to continue to be a reasonable way for a website to make money, something needs to be done. Publishers must be able to keep malicious and misbehaving ads off of their sites. This solution puts the power in the hands of the publishers, where it belongs." The question is, "Are the browsers willing to do it?"

And just so that people understand, the way the ecosystem works now is a publisher, like this guy, or anyone wanting ad-enhanced pages, creates a region on their page for an ad. And they include a URL to the ad network, which is essentially blind. It identifies them, essentially, it's a URL they received as part of their relationship with the ad network. And so when a user visits their site, the browser gets the site's page content containing this region with a URL. The browser then goes out and fetches it. And the point is the page has no control over what happens in that region. And script that runs in the retrieved text from the ad, that is, essentially it's like a mini page. It's got script. It can do anything it wants, that is, in terms of the script-enabled behavior of the code running in that content.

So what this author is proposing, and I think it's brilliant, is that it be possible to create disallowed behavior of scripts running within a region of a page, typically it's a DIV, a division, which the browser would provide as part of - and so the idea would be, the way this would evolve would be the HTML, the World Wide Web Working Group would devise an extension to the HTML5 standard to allow CSS or style to create restrictions on what scripts within a division can do.

Leo: You just invented AMP, by the way. You know about Google AMP; right?

Steve: No.

Leo: Well, you should look into it because this is Google's response to this is a

publishing platform - it's primarily intended for mobile - but a publishing platform that does exactly this. It has very limited restrictions. It's not even within a DIV. It has very limited restrictions on what you can put in an ad. It has kind of an approved thing. And the idea is because mobile - ads become so untenable on mobile.

So if you're browsing around, and you find a news article on Flipboard, or Nuzzle is a good example. And when you load it, you may have noticed this, you'll see a lightning bolt. That means it's an AMP page. And you'll notice it's very clean and loads incredible quickly. Google caches all the content. Google has a very limited set of JavaScript that's allowed. It's a very stripped down thing.

But what I - and I complained about it at first. But what I do like about it is it resolves in a noncompliant or in a setting where the browser can't understand what's going on, it resolves to legit HTML, even though it's using nonstandard tags for images and things like that. So Facebook has its own way of doing this. Google has a way that might be more palatable because it's open, and you can run your own server. Anyway, I agree with what you're saying. And Google does, too, apparently, because they offer this as an alternative to publishers.

Steve: So I guess we just need to push this out as a standard.

Leo: It kind of is a standard. You know, my problem was that they didn't go to W3C and say, let's get this as a standard. But because it degrades gracefully - you could read more about it at AMPproject.org.

Steve: Good, I will, yeah.

Leo: Yeah. And I'd be curious next week maybe if you could let us know if this is kind of what you were thinking about. The real problem I have with it, it's intended for mobile. But right now more than 1.5 billion pages have been AMPed. And if you have a WordPress blog, for instance, there's an AMP plugin. So your stuff is automatically AMPed.

Steve: Nice.

Leo: And so that's kind of the way to do it. It still has an ad, you know, people like this guy need ads.

Steve: Yes, yes, yes. And he's not saying he doesn't want them. He wants them.

Leo: He wants responsible ads.

Steve: Yes, exactly. He's wanting ads. Well, for example, an ad that steals the focus so you can't enter contents in a form, well, you know...

Leo: It's terrible. It's terrible. And AMP would not allow that. I mean, AMP is much more restricted as to what you can do. It's really - I think it's very interesting. Clearly Google is just like this guy, saying, well, we don't want people to use ad blockers because our business is advertising.

Steve: Yup. So it's a compromise.

Leo: So how can we compromise, exactly.

Steve: Yup. Very cool.

Leo: It's an interesting conversation. I completely agree.

Steve: I'll have a tune-up for next week.

Leo: A tune-up?

Steve: A tune-up on AMP. I'll know all about it.

Leo: I'd love to know what you think. And you can go back to our TWiG conversations. We've even had Richard Gingras, who's head of Google News, on to debate it. Because I was kind of against the idea of creating a non-standard extension to HTML. But I gave up when I understood that it would degrade to standard HTML. So that's good.

Steve: Right.

Leo: That's good. All right. Are we done?

Steve: We are.

Leo: Can it be?

Steve: Two hours. A great podcast.

Leo: Holy cow. How did that happen? You'll find this show on the Interwebs. Actually, why don't you go to GRC.com because not only will you find Security Now! there, audio and transcripts, you'll also find SpinRite, Steve's bread and butter and the world's finest hard drive maintenance and recovery utility, a must-have. My

slogan that I wrote: If you have a hard drive, you need to have SpinRite.

Steve: Thank you.

Leo: But you also will find so many other free things that he offers there. It's just a great compendium of information and knowledge: GRC.com. We also have, of course, audio and video of the show on our website, TWiT.tv/sn. You'll find it also everywhere you can find podcasts. In our 12th year of spreading the security good word; so, you know, it's pretty much everywhere.

And you can watch live. We do it every Tuesday, 1:30 Pacific, 4:30 Eastern, 20:30 UTC if you'd like to tune in and join us in the chatroom at irc.twit.tv. We're on YouTube Live at YouTube.com/twit. We're on Ustream at Ustream.tv/twit. We're on Twitch, Twitch.tv/twit. But the easiest would be just simply go to TWiT.tv./live, and then you can pick the stream you prefer. There's even an audio stream, several audio streams, to make it easy on your bandwidth. Steve, we'll just have to reconvene next week, I guess.

Steve: I think we should. I remember 12 years ago when we had just started the podcast. Every so often I would Google Security Now!. It was like, oh, look, 12 hits. And now it's just - it's ridiculous.

Leo: Well over 100,000 listeners every week. Puts it up there with a small cable channel. And, by the way, it beats anything TechTV ever did. Thanks, Steve, and we'll see you next week.

Steve: Thanks, my friend. Talk to you then. Bye.