Android Security

Description: This week we are primarily going to share Google's well-deserved, self-congratulatory, but also very honest update on the status of Android Security at its 10th birthday. But before that we're going to share some of the continuing news of the WinRAR vulnerability, some really interesting data on Russian GPS hacking, Android's April Fools' Day patches, Tesla autopilot spoofing, some follow-up on the ASUS "ShadowHammer" attack and the targeted MAC addresses, the final release of the Windows 10 (last) October 2018 update, a VMware update, a SQRL question, two bits of listener feedback, and a SpinRite development question. Then we take a look at the state of Android 10 years in.

High quality (64 kbps) mp3 audio file URL: http://media.GRC.com/sn/SN-708.mp3
Quarter size (16 kbps) mp3 audio file URL: http://media.GRC.com/sn/sn-708-lq.mp3

SHOW TEASE: It was 10 years ago Google came out with the first Android phone. Ten years later, they've come out with a security report. Steve Gibson talks about the state of Android security. And I'll just give you a little hint, a little spoiler alert. He's pretty impressed. Coming up next on Security Now!.

Leo Laporte: This is Security Now! with Steve Gibson, Episode 708, recorded April 2nd, 2019: Android Security.

It's time for Security Now!. Yay. The moment you wait for all week long.

Steve Gibson: Woohoo!

Leo: Woohoo! Steve Gibson, he's been slaving over a hot keyboard, apparently making some funny faces, too.

Steve: Steve, Steve the reindeer.

Leo: Do you work - I think you work all week preparing this show. I know it's a lot of work.

Steve: I start yesterday, and I work - and Lorrie and I have this whole routine where, like, Monday is just set aside, and there's no dawdling Tuesday morning. It's like I get right to it. So, yeah, I do put a lot of time in. But I think that's, I mean, I don't know how

to do this, what we're doing now, any differently. I want to look at all of the news of the week.

Today's big topic, when I started out I didn't realize that I was going to be as impressed as I was. But last week Google, for like the 10th birthday, the 10th anniversary of Android, did, I think, a well-deserved, somewhat self-congratulatory, but also very honest update on the status, today's status of Android security. And there was so much detail and so much meat in this - I think it was a 22-page document - that I thought, okay. Our listeners know I'm pretty tough on Google and Android when I think something really needs to get fixed. But I came away feeling differently. I'm impressed by what they have tackled that is not a job I would ever want, you know, putting an open source powerful platform running on capable hardware in the hands of non-computer-savvy consumers who just want it to work in the face of a proactively extremely hostile environment. I mean, we know, because we cover it all the time, how much pressure there is to get into and to subvert people's devices. And looking at the details changed my opinion enough that I thought, okay, I have to share this.

So our main topic, which we will wrap up with, is Android security here on, you know, basically 10 years in. Where are we? But there was also a lot of other interesting news. There was continuing news about the WinRAR vulnerability, including email that I just received this morning from WinRAR, which means that for the first time they got proactive. Also some really interesting research that shows the extent of Russia's hacking of global positioning systems and what that means. We did have an Android update yesterday on April Fools' Day, but it's no laughing matter. It fixed a couple remote code execution vulnerabilities we'll touch on.

Also there was a very - a disturbingly mis-headlined report about Tesla autopilot spoofing. It got picked up because it was hot, unfortunately, saying that researchers at Tencent had arranged to spoof Teslas into driving into oncoming traffic, which is not at all what they did, but that's what the headlines said. So we have to fix that. Also we have some really interesting follow-up on last week's discussion of ASUS's ShadowHammer attack and those targeted MAC addresses. We've got the final release of last October's 2018 Windows 10 Update; a VMware update that followed from the Pwn2Own exploits that we covered. We've got a SQRL question; two bits of listener feedback; a SpinRite R&D question. And then we're going to spend a lot of time, because I think it deserves it, talking about what Google has achieved with 10 years of really concerted effort.

Leo: I'm relieved because when I saw the title I thought, oh, crap. Because I like Android. I use Android. In fact, while I have an iPhone X, I've been lately really in love with the brand new Galaxy S10 Plus. It's just a great phone. And I would be kind of sad if I thought I couldn't use it securely.

Steve: Well, and as we'll see, because there's a lot - they really did a statistical breakdown. Staying with the major suppliers - if not Google themselves, then one of the other major partners - you are in a much better position because it still is the case, you know, not only are they being more responsible - we'll talk about that. There's, again, lots of numbers in this report. And I've, like, whittled it way down. So I've just kind of - I skipped big chunks of okay, well, we don't need to talk about that. But also their attention, the big partners' attention to detail and their commitment to pushing the patches out that Google produces. So, I mean, it's very clear, even with something like an archiving piece of software, which we now wish had auto update capabilities, in the world we are in, stuff has to take care of itself.

Leo: Yeah. Yeah.

Steve: And so unfortunately, the nature of Google's Android ecosystem is there are many fringe partners who are like, oh, yes, we'll be happy to take that. And then, you know...

Leo: I like that, "fringe partners," yeah, yeah, mm-hmm.

Steve: Anyway, so, yeah, we've got a lot of fun stuff to talk about today.

Leo: Good. LastPass.com/twit. I'm sorry, I get overexcited when I start talking about - I could go on and on and on.

Steve: Well, and I fully accept that passwords are never, and I really do mean never, they're never going to go away. Look at all - we still have FTP.

Leo: Yeah, yeah.

Steve: These things just don't die.

Leo: No. Tell me.

Steve: They'll become increasingly fringy.

Leo: Right.

Steve: But still there will always be a place. And I don't know any of my passwords anymore because they're all gibberish.

Leo: Right. They should be gibberish.

Steve: Like they're supposed to be, yeah.

Leo: Like they're supposed to be, yeah. And you know what, SQRL's going to help a lot. We're going to get SQRL out there. And the more sites that use SQRL, the better. Would SQRL work on a mobile app, too? Or does it have to be a website?

Steve: Oh, no. We have iOS and Android apps that are, like, running right now and working great.

Leo: But they work with websites. I mean, if I had an app that needed a password, could I use SQRL to authenticate with the app?

Steve: Ah, yes, yes. You are able to do that.

Leo: Oh, nice.

Steve: Yes.

Leo: Can't wait till you finish, Steve. Hint, hint.

Steve: So, and I'll talk about where I am at the moment. When we get there, I'll touch on that. Our Picture of the Week, you asked me right off the bat, is this our Picture of the Week? Because it's, like...

Leo: It's not funny.

Steve: Exactly. It's not funny. Actually, Leo, on the Sophos website there is the most adorable kitten. And I was so tempted to just put up this little kitten. But I thought, okay, Gibson, you know, how much have you been drinking? So I thought, no, I really can't do that. But I don't know what it was, it was just the most adorable thing.

Leo: Now I'm going to have to go. You're going to have to see it.

Steve: Oh, lord. Anyway, so the reason this graph is significant is - so first of all, we will be talking later in the podcast about PHA. I'll be using the acronym PHA because it's so much easier than saying Potentially Harmful Applications. But that's like Google's term, their grab bag term for any crap of any sort you don't want on your Android device, PHAs, Potentially Harmful Applications.

What's, well, first of all what's interesting is that they acknowledge that there has been an increase in PHAs, even in the Google Play Store, 0.05% up to 0.08%, that was in 2016; 0.08 in 2017 and holding at that. However, they also redefined, they added a large class of ad-click PHAs into that definition. So they've revised the definition to include what they felt was a major harmful category, an unwanted application, which are these fake clicking things. And so that's the reason for the increase.

But the big point here is that the outside of the Google Play Store, that is, in the wilds, where people tend to want to sideload things that are not available, the rate of potentially harmful applications is much higher. What is it, like maybe 12, 13% higher. I mean times higher. So it's 0.74 as opposed to 0.05%.

Leo: Although, you know, you tell me, oh, go to a third-party store, I'd think it's 5% or 10%. I'm in some ways relieved it's less than 1%, anyway.

Steve: Yeah. Yeah, yeah. And although there are some other scary statistics that we will be getting to, like just in terms of, you know, it's still - certainly it's a "downloader beware" environment. But you can be much less worried if you just get things from Google Play. So, I mean, and the good news is I think most...

Leo: There's no reason not to, really.

Steve: Exactly. You know, everything you want will be there. And Leo, even the SQRL app is there.

Leo: Yay.

Steve: So, okay. So this WinRAR vulnerability has turned out to be as big a problem as we originally expected it to be, only because it has such breadth of install base at half a billion downloads over time.

Leo: Wow.

Steve: It has always been vulnerable. It is still vulnerable. There is no proactive update mechanism for it, which means it isn't going to fix itself. And so right now the bad guys are in this environment where they're in a big panic when some new vulnerability on a mainstream platform comes out because they know their window of exploitability is short. It's going to get fixed soon. The machines are going to update themselves with little user interaction. And then any investment that they make in creating an exploit for it has a very short time horizon before it just won't work anymore. Not so with this.

So this has generated an outsized response from the hacker community, I mean, the malicious, I don't mean to use - I always try to say the attacker or the malicious hacker because we know not all hackers are bad. But FireEye Security, who we haven't talked about for a long time, but they're still out there kicking, they did a summary of the things they had seen which I thought just was worth sharing with our listeners. The title of their blog posting from their threat research group was "WinRAR Zero-Day Abused in Multiple Campaigns." And I'm unsure this really counts as a zero-day any longer. I mean, yeah, on Day Zero it did. But then there was Day One, and then there was Day Two, and now we're like on Day, what, 20 or something. But the point is I wouldn't say it's a zero-day any longer. It did surprise us when it happened, but now everybody knows.

So they said: "WinRAR, an over 20-year-old file archival utility used by over 500 million users worldwide, recently acknowledged a longstanding vulnerability in its code base. A recently published path traversal zero-day vulnerability, disclosed in CVE-2018-20250 by Check Point Research, enables attackers to specify arbitrary destinations" - and actually that's true. We've only been talking about putting it in the user's start folder. But they actually are arbitrary, depending upon the permissions of the user at the time, which potentially allows further exploitation.

Anyway, "...arbitrary destinations during the file extraction of ACE formatted files, regardless of user input. Attackers can easily achieve persistence and code execution" - which that's like the two golden things you want, persistence and code execution - "by creating malicious archives that extract files to sensitive locations, like the Windows Startup Start Menu folder. While this vulnerability has been fixed in the latest version of WinRAR, WinRAR itself does not contain auto-update features, increasing the likelihood,"

they write, "that many existing users remain running out-of-date versions." And really, I mean, who wouldn't, unless you were listening to this podcast, or you were following this, I mean, this is not going to make mainstream news. WinRAR? What? So nobody's going to find out about this.

Then they said: "FireEye has observed multiple campaigns leveraging this vulnerability, in addition to those already discussed by 360 Threat Intelligence Center. Below, we will look into some campaigns we came across that used customized, interesting decoy documents with a variety of payloads including ones which we have not seen before, and ones that use off-the-shelf tools like PowerShell Empire," which is what they wrap up with. Anyway, so there was Campaign 1, impersonating an educational accreditation council letter. And they write: "When the ACE file Scan_Letter_of_Approval.rar is extracted with vulnerable WinRAR versions lower than 5.70, it creates a file named winSrvHost.vbs in the Windows Startup folder without the user's consent. The VBScript file is executed the next time Windows starts up. To avoid user suspicion, the ACE contains a decoy document."

And notice it's an ACE file with a .rar extension. That's what we're always seeing because someone's going to look at ACE and think, what the hey? But as we know, WinRAR examines the header to determine the actual archive type independent of the Windows extension, the Windows file extension. So as long as you have the 20-some things checked in WinRAR that it knows about, it will get control when you click on it, and it will open. And certainly it's going to be in charge of a RAR file, even if it contains an ACE extension.

So anyway, they explain that this ACE file contains a decoy document, Letter of Approval.pdf, which purports to be from CSWE, the Council on Social Work Education, apparently copied directly from the CSWE website. So someone just got the PDF, the legitimate PDF from the website, RARed it with a maliciously created winSrvHost.vbs file, and either targeted or spammed it or whatever. And then they go into extensive detail I won't cover here about the way the VBS file operates, its access to a specific command-and-control server at 185.162.131.92 via HTTP requests, how the requests are handled, what information it pulls from the system to incorporate into the headers, and so forth. You know, all of that is the stuff that these guys enjoy researching, but they're just an example of what one of these things does.

It ends up installing a RAT, a Remote Access Trojan, known as NetWire, which is a well-known RAT, which supports a backdoor and commands. It can receive the command "d," which will delete a file in the command parameters; "Pr" downloads a file from a URL and executes it; "Hw" retrieves hardware information; and "av" looks for antiviruses installed from within a predetermined list. So just a relatively simple set of commands, but that allows the command-and-control server to see what AV you've got installed and then potentially alter its behavior based on what it knows or who it knows may be watching what happens next. So again, this is something that they have found actively exploiting the current WinRAR vulnerability.

Second campaign was an attack on the Israeli military industry. They said: "Based on the email uploaded to VirusTotal" - which is where they found this - "the attacker sends a spoofed email to the victim with an ACE file named SysAid-Documentation.rar as an attachment." They said: "Based on the VirusTotal uploader and the email headers, we believe this is an attack on an Israeli military company. The ACE file contains a collection of decoy files related to documentation for SysAid, a help desk service based in Israel.

"One of the files, Thumbs.db.lnk, is a valid Windows shell link file pointing to C:\Users\john\Desktop\100m.bat. But the icon for this link" - and John is probably the name of the user who downloaded it, so it created this thing. The icon for the link - or maybe it wouldn't even need to actually be able to resolve it because the icon for the link

is remotely hosted on one of the command-and-control servers. What's clever about this is that Windows will reach out to obtain the icon; and it will include, as we've often discussed, when you do file and printer sharing, it sends NTLM, NT LAN Manager hashes in order to assert the identity of the system that is asking. So this is a way for this command-and-control server to steal the LanMan hashes.

Then, upon extraction, the WinRAR flaw causes a previously unknown payload that FireEye found and named SappyCache to be copied to the user's Startup folder with the filename ekrnview.exe. And they explained the payload will be executed the next time Windows starts up, of course, because it's in the Startup folder. SappyCache attempts to fetch the next stage payload using three different approaches. And I'll quickly say it decrypts a file to a specific location and then tries to - the malware reads a file, decrypts it using RC4 to get a list of command-and-control URLs. Otherwise it decrypts a resource from an executable if the first attempt was not successful, retrieving, again, a set of command-and-control URLs.

And then, if neither of those work, then it retrieves the command-and-control URLs using a payload from four different hard-coded URLs, which are determined at runtime. It grabs the computer's name, the version of Windows being used, then enumerates all the processes running in the system and sends those to the command-and-control server so that it can see everything about what the user is running and then decide what it wants to return as the second-stage payload based on what it finds running. Which is more what we're seeing now is that, with AV being so present, it may want to pick and choose its response based on what it knows about the system that the user is running at the time. So clever, from that standpoint.

The third attack was targeting individuals in Ukraine which they were able to specifically determine from what they saw. And the fourth campaign was a credential and credit card dump. It actually used those as decoys. So it looked like this is the classic "there's no honor among thieves." So this would have been posted somewhere where other nefarious people would be looking to obtain credential dumps and credit card credential dumps. Those were RARed with a malicious RAR that would then infect the user's machine with the QuasarRAT, the Quasar Remote Access Trojan.

So anyway, FireEye concludes their report and their analysis by saying: "We have seen how various threat actors are abusing the recently disclosed WinRAR vulnerability using customized decoys and payloads, and by using different propagation techniques such as email and HTTP queries." They said: "Because of the huge WinRAR customer base, lack of auto-update feature, and ease of exploitation of this vulnerability, we believe that this will be used by more threat actors in the coming days."

They said, and this is what I found really interesting in their report, too: "Traditional AV solutions will have a hard time providing proactive zero-day detection for unknown malware families." I was assuming that AV would be able to look inside of RARs to detect the abuse of this path traversal attack. But that may not be feasible, which I found interesting.

They said: "It's also worth noting that this vulnerability allows the malicious ACE file to write a payload to any path if WinRAR has sufficient permissions. So although the exploits we've seen so far choose to write the payload to the Startup folder" - which of course is always guaranteed to work, and so why look any further - they said "...a more involved threat actor could come up with a different path to achieve code execution so that any behavior-based rules looking for WinRAR writing to the Startup folder could be bypassed." That's a good point. You could watch the behavior of WinRAR and say, wait a minute, there's absolutely no reason that WinRAR should be writing to the Startup folder, and so block based on that behavior. They said: "Enterprises should consider blocking vulnerable WinRAR versions and mandate updating WinRAR to the latest version."

And then, as I said, this morning my inbox contained a letter from Julia D. Seymour at Win-Rar.com with the subject line: "Update WinRAR and get Malwarebytes Premium FREE!" All caps on FREE, exclamation point. She said: "Dear Customer," meaning me because I'm a registered WinRAR user, of course. I want to support them. "Greetings from WinRAR. You may ask yourself why we are contacting you at this particular time. We wouldn't usually contact our users individually, but these are extraordinary circumstances. We have recently released the new version of WinRAR 5.70, following the discovery of a potential security vulnerability within the unacev2.dll. For more information please check here," and she gives a link.

"Here at Win.Rar GmbH, we believe in full transparency, which is why we are contacting our users personally to explain and offer advice regarding their continued safe use of WinRAR. We always recommend that users update to the latest version to remain risk-free and to have full access to all of the current improvements, additions, and bug fixes. As a WinRAR license holder, we wanted to make sure that you are aware that you can upgrade to the latest version free of charge. You will find the latest version of WinRAR 5.70 in your desired language here." And then they give a link.

"In addition to that, we are offering you Malwarebytes Premium for free." And there's an asterisk that we'll explain in a second. They said: "Download the most current version from here" - www.malwarebytes.com/mwb-download/thankyou - "and insert the license key by clicking the 'Activate License' button in the top menu bar of the software." And she said: "Here is your Malwarebytes registration key." And I blanked it out of the show notes because it's not clear to me whether that was a per licensed user or whether it's blanket, and everyone got the same one. But in any event, to respect them, maybe they have a deal, and I didn't want to just spew this around. Also it's not lifetime.

So then she says, under WINRAR Upgrade: "Upgrading is quick and easy. No complicated uninstalling of the previous version is necessary." Which I had verified and told our listeners about several weeks ago. "Do not delete your existing WinRAR program folder. Your registration information and WinRAR settings will be kept then. Close all open WinRAR archives and exit WinRAR before installing. Then you can just install the new version of WinRAR over your older installation by double-clicking on the EXE file you've downloaded. Now you're ready to continue using the best compression tool around. We would also recommend that you sign up to receive our newsletter," blah blah blah.

Anyway, the point is that the asterisk brings us down to a little, in very fine print at the bottom of the email: "Using the provided license key, you will receive Malwarebytes Premium for free for a period of three months." And I think that's entirely appropriate. Their concern is that something may have crawled into your computer through your use of WinRAR, which you had no way of knowing to update. If you were licensed and supporting them, and you had kept your email address from the time you licensed, you would have received this today. And they're giving you for free, in return, a useful AV with which to scan your system for any damage that may have been caused by this.

So props to them. I think they've acted as responsibly as they possibly could. And certainly all of our listeners have been protected for weeks because we knew about this the day it happened. So I think this probably closes the chapter. Unless there's some massive widespread worm or exploit or something that happens that brings it back into the news, we've pretty much beaten this thing to death.

Russia has been messing with Global Positioning Systems. It's a 66-page detailed and extremely compelling analysis of signals intelligence collected from fixed and in-orbit assets, as we will see in a second. So I'm certainly not going to go through 66 pages. But I will share the executive summary because that gives us a good overview and summary. And I just sort of wanted to plant a little more detail than just saying, oh, yeah, there

have been anecdotal reports of GPS being blockable or spoofable, but we don't have any details. These guys do.

And for anyone who's interested, I can certainly recommend this PDF. It was from Sophos, and the PDF is on their site. So I'm not sure it's going to be available forever. So if anyone is interested, you may want to go to the show notes. Or maybe the - the paper is titled "Above Us Only Stars," with the subtitle "Exposing GPS Spoofing in Russia and Syria." So presumably googling that expression would take you to this also.

So they said, in their executive summary: "GPS and other Global Navigation Satellite Systems" - collectively GNSS, Global Navigation Satellite Systems - "are used in everything from cellular communication networks to basic consumer goods, high-end military systems, and stock trading inputs. But these systems," they write, "are vulnerable. By attacking positioning, navigational, and timing data through electronic warfare (EW) capabilities, state and non-state actors can cause significant damage to modern militaries, major economies, and everyday consumers alike. With recent technological advances, the tools and methodologies for conducting this interference are now at a high risk for proliferation. GNSS attacks are emerging as a viable, disruptive strategic threat.

"In this report, we present findings from a year-long investigation ending in November 2018 on an emerging subset of electronic warfare activity: the ability to mimic, or 'spoof,' legitimate GNSS signals in order to manipulate PNT" - that's the timing data, the PNT data which is how positioning data gets spoofed. They said: "Using publicly available data and commercial technologies, we detect and analyze patterns of GNSS spoofing in the Russian Federation, Crimea, and Syria that demonstrate the Russian Federation is growing a comparative advantage in the targeted use and development of GNSS spoofing capabilities to achieve tactical and strategic objectives at home and abroad. We profile different use cases of current Russian state activity to trace the activity back to basing locations and systems of use."

So there's four sections, which they summarize. "In Section One," they say, "we examine GNSS spoofing events across the entire Russian Federation, its occupied territories, and overseas military facilities. We identify 9,883 suspected instances across 10 locations that affected 1,311 civilian vessel navigation systems since February of 2016. We demonstrate that these activities are much larger in scope, more diverse in geography, and longer in duration than any public reporting suggests to date.

"In Section Two we examine the role of Russian GNSS spoofing for very important person (VIP) protection. We find a close correlation between movements of the Russian head of state and GNSS spoofing events. We believe the Russian Federal Protective Service (FSO) operates mobile systems to support this activity. Through a review of Russian procurement data, we identify one possible mobile system, manufactured by a company closely connected to the FSO (Federal Protective Service).

"In Section Three we profile the use of Russian GNSS spoofing for strategic facilities protection. We identify potential technology in use for facility protection in Moscow. We also highlight spoofing activities taking place in proximity to protected facilities on the coast of Russia and Crimea in the Black Sea. Through a line-of-sight analysis, we judge the most likely placement for a GNSS spoofing transmitter on the Black Sea to be at a multimillion dollar 'palace,' formerly owned by reported family members of senior FSO officers and previously reported to be built for President Putin.

"Finally, in Section Four, we expose the use of GPS spoofing in active Russian combat zones, particularly Syria, for airspace denial purposes. This is a capability scarcely reported in the public domain. Using data from a scientific sensor in the International Space Station, we are able to identify ongoing activity that poses significant threats to

civilian airline GPS systems in the region. We pinpoint the most likely location for the system to be the northwestern quadrant of Khmeimim..."

Leo: I don't think that's how it's pronounced, but I might be wrong.

Steve: Well, K-H-M-E-I-M-I-M.

Leo: Yes, Khmeimim, right.

Steve: "...Khmeimim air base, and identify potential military grade electronic warfare systems in use through publicly available information." And then they conclude their summary, saying: "The Russian Federation has a comparative advantage in the targeted use and development of GNSS spoofing capabilities. However, the low cost, commercial availability, and ease of deployment of these technologies will empower not only states, but also insurgents, terrorists, and criminals in a wide range of destabilizing state-sponsored and non-state illicit networks. GNSS spoofing activities endanger everything from global navigational safety to civilian finance, logistics, and communication systems."

So anyway, I just thought that was very interesting. Again, if anyone is interested, if you browse through this PDF, you will come away convinced. They've got diagrams, charts, pictures, annotations, I mean, a huge body of clear evidence to back this up.

Leo: Now, the Russians use their own version of GPS called GLONASS.

Steve: Correct, yes. There are four different positioning systems operating worldwide.

Leo: But they're also attacking GPS, as well.

Steve: Yes.

Leo: Yeah, okay.

Steve: And China also has their own.

Leo: Right.

Steve: So everyone has built their own because they don't trust anybody else. So it's like, okay.

Leo: Right. Well, and originally the U.S. military didn't let ours be used by them, either.

Steve: Right. And remember that ours also had its resolution blunted.

Leo: It's fuzzed, yeah, yeah.

Steve: Yes, for consumer purposes, or civilian purposes, yeah.

Leo: Yeah. So, interesting.

Steve: Anyway, so this is going on. I just kind of wanted to put this on our listeners' radar. And what basically they're saying is that the collapsing cost of producing transmitters capable of confusing GPS and other positioning systems means that it's going to become more prevalent in the future. So it was amazing, decades ago. And it was sort of the province of high end. That's not the case anymore.

So, as we were saying, yesterday was April 1st, infamous April Fools' Day. But no one was fooling here. Just wanted to note that Android users should update or look for updates from their provider because there were a pair of critical remote code execution vulnerabilities, and nine high-severity privilege elevation vulnerabilities, and also an information disclosure vulnerability, all patched. They were, once again, the RCEs, Remote Code Execution problems, were in the much-troubled Media Framework, which of course has been a constant source of trouble because it is a massive interpreter, and we know how hard those are to get right.

So there were two vulnerabilities. What were updated is versions 7.0, 7.1.1, 7.1.2, 8.0, 8.1, and 9. So everything essentially from 7.0 on. Depending upon where you get your Android, do, as it was just released yesterday, update yourself because, again, what we have seen is that a patch gets reverse engineered, and the bad guys jump on it. And we know that the Media Framework is particular susceptible because essentially your Android mobile device is a wide-open maw, a funnel, looking for things, tweets and Snapchats and Twitter pictures and just everything coming into it.

And if there is a problem in the renderer of some type of content, then it's readily exploited. And the bad guys are going to look at this, and they started yesterday, and they're going to try to get people who haven't updated. So do so. They did say of them that there were no reports of active customer exploitation or abuse of any of these newly reported issues. So none of these are zero-days. But we know that even one-days is now, these days, enough. So worth getting fixed.

Okay. Now, Leo, as a Tesla owner, this will be of interest to you.

Leo: Yes.

Steve: And I'm sure we have many Tesla owners. The attention-grabbing headline which is very, very wrong was "Researchers Trick Tesla to Drive Into Oncoming Traffic."

Leo: Now, that would not be a good thing.

Steve: Really.

Leo: I would prefer not to, thank you.

Steve: In terms of ruining your day...

Leo: Yeah, yeah, pretty high up on the list.

Steve: And unfortunately, in this case, the hack appears to have been easy to pull off, but not at all what the headlines have said. There is a 40-page research paper published by researchers at Tencent Keen Security Lab. Their paper was titled "Experimental Security Research of Tesla Autopilot." And I hadn't - the pun of "autopilot" hadn't occurred to me, actually, Leo, until I began...

Leo: Oh, auto. I get it.

Steve: Isn't that wonderful?

Leo: I never thought of that either.

Steve: Isn't that wonderful?

Leo: I don't like the name because it implies it flies itself, and it doesn't.

Steve: No.

Leo: So it's a bad name.

Steve: And I will argue, and our listeners may be a little more - even you may be a little more convinced of that by the end of this because they did find something which is worrisome. But anyway. So their abstract reads - and I'll share it because they did three different things.

The abstract reads: "Keen Security Lab has maintained the security research work on Tesla vehicle" - this is a Chinese outfit, by the way, so you'll see that their English is not quite ours, but still very legible, or intelligible - "on Tesla vehicle and shared our research results on Black Hat USA 2017 and 2018 in a row. Based on the root privilege of the APE" - that's Tesla Autopilot ECU, software version 18.6.1. And we should note it's now at 18.25 or something, so this is somewhat dated - "we did some further interesting research work on this module. We analyzed the CAN" - and we know that that's the private bus that ties everything together.

They said: "We analyzed the CAN messaging functions of APE and successfully got remote control of the steering system in a contact-less way. We used an improved optimization algorithm to generate adversarial examples of the features," and then they said here, "auto wipers and lane recognition which make decisions purely based on camera data, and successfully achieved the adversarial example attack in the physical world." I know what all those are, so I'll explain that in a second. "In addition, we found a potential high-risk design weakness of the lane recognition when the vehicle is in Autosteer mode.

"The whole article is divided into four parts: First, a brief introduction of Autopilot. After that we will introduce how to send control commands from APE to control the steering system when the car is driving. In the last two sections, we will introduce the implementation details of the auto wipers and lane recognition features, as well as our adversarial example attacking methods in the physical world. In our research, we believe that we made three creative contributions: One, we proved that we can remotely gain the root privilege of APE and control the steering system. Two, we proved that we can disturb the auto wipers function by using adversarial examples in the physical world. Three, we proved that we can mislead the Tesla car into the reverse lane, meaning oncoming lane, with minor changes on the road."

Okay. So first of all, they did succeed in taking over steering. And just for the fun of it, they used a Bluetooth-connected gamepad controller.

Leo: Oh, god.

Steve: To steer the Tesla. So, yeah. It turns out that for the second point, this auto wiper, the Tesla, whereas other cars like mine use a moisture sensor, essentially an inductive moisture sensor, to sense the presence of water on the other side of the glass windshield, Tesla, since they already have forward-looking cameras, they thought, well, let's, you know, we're looking out through the windshield. We should be able to sense if there's drops of water there. And so what's what Tesla does. So they have an algorithmic auto wiper technology. But it turns out that you can spoof it using deliberately created images at a distance. And so they put up some poster boards with funny-looking stuff on it, and the windshield wipers went. So it's like, okay. So they were spoofing the AI algorithm after...

Leo: Actually, I could use that because my windshield wipers never work properly.

Steve: Oh, well, yes. And so maybe the algorithm needs a little more attention.

Leo: Yeah, maybe, yeah.

Steve: Okay. And so then the third thing they did, obviously most worrisome, was by reverse engineering and understanding the autopilot's lane recognition algorithm, they were able to induce a lane change to the left which, in a two-lane road scenario, would have been into oncoming traffic.

Leo: Right.

Steve: And all that was necessary, perversely, was the strategic placement of three large dots on the road ahead of the car. So just sort of like - it looks sort of like as if it was - I could see it would be interpreted maybe as the side of the road moving to the left; and the car said, oh, I need to change lanes, and so it went to the left in order not to hit an obstruction.

Leo: Well, I mean, this is like Roadrunner and Wile E. Coyote, painting the lines on the road to go off the cliff. If you're not paying attention, and you let the car follow the lines in the road, yeah, it's going to go off the cliff, of course.

Steve: Well, these were three small dots, though. I mean...

Leo: Well, that's what I want to see is, well, what did it look like?

Steve: Yeah. So but here's the point. And this is what unfortunately the people who didn't read the article, but just wrote the headline, or maybe who just wanted to say, ooh, we're getting a lot of clicks on this...

Leo: I think the latter, yeah.

Steve: Yeah. This does not mean, I have this in capital letters in my notes, NOT mean that a Tesla would actually turn into oncoming traffic, but rather that, in this carefully crafted and isolated instance, they were able to induce the car to redirect into the lane to the left.

Leo: Yeah, yeah.

Steve: That's different than saying...

Leo: Oh, yeah, turning into traffic.

Steve: Yes.

Leo: No.

Steve: For example, I would be shocked if the AI didn't have, it must have...

Leo: Oh, if there's oncoming traffic, it's not going into that lane.

Steve: Yes, multiple other sensors that would - and input to expressly forbid it from actually turning into oncoming traffic. So the headlines did Tesla a big disservice. But there is an important message here, I think, nevertheless. A car's autopilot really is an extremely complex interpreter of the car's sensorium. I mean, it's an interpreter. And how many times have we talked about how interpreters can be deliberately fooled by malicious actors who have access to the interpreter's internals?

That's exactly what happened here. Within an isolated environment, with preset conditions, the Tesla's autopilot was fooled, in a limited test case, by something that would never have fooled a human driver. So we would think, huh? I wonder why there are three weird dots on the road. Whereas the Tesla AI, seeing the same thing, in that

version - and we don't know about today's AI, we know about the one they tested - apparently thinks, oh, time to change lanes to the left. Absent any inhibitory input which it probably also would be looking for that would say, ooh, but except that there's this oncoming lane of traffic, so I'm going to ignore the three dots.

Leo: Actually, I suspect the Tesla would do better in this environment than a lot of - so lane keeping is something a lot of cars have. And because Tesla has all this other self-driving technology, including radar and other cameras, I bet you it's less likely to follow those dots than many other lane-keeping vehicles. But no matter what, Tesla does a lot to make sure you're paying attention and have your hands on the wheel.

Steve: Yes.

Leo: In fact, if you don't see the alerts that say keep your hands on the wheel after five minutes, it disables Autosteer and says you don't deserve it for the rest of the trip. It gives you - it spanks you with a big red notice and says, "We're turning it off. Sorry, buddy."

Steve: You have abused the privilege.

Leo: And I don't need to be told that because honestly I don't trust it.

Steve: Unh-unh. Good. Because I really do, I do enjoy doing the podcast, Leo. And I want to run out all three digits of the numbering system here.

Leo: We will, I promise. But you don't trust it. You keep your hands on the wheel. You keep your eye on the road. You're paying attention, just in case, because it is just trying to figure out where the lane is. It's going to make a mistake sometimes. And it's happened to me in the three years I've had this Tesla, the Model X, more than once, that particularly on one curve that we go around on the way to San Francisco, that it's started to veer over into the guardrail. And of course I'm expecting it now, so I pull it back. But it's just that doesn't surprise me that it can be fooled by weird lines on the road. That doesn't surprise me.

Steve: Yeah. You know, I really - I come back to the Palm Pilot and how the brilliance of Jeff's design for that was that he asked the user to accommodate the alphabet a little bit. And that just allowed the recognition to nail what the user was doing. And the analogy here is I really think we ought to move to a mode where our - because this seems to be where we're headed with this kind of car technology. Our roads ought to have some car assist stuff. And I think it's foreseeable that it's going to happen where, for example, Tesla must have some informatics feedback where they know, in some database somewhere, that people are having to pull their car back onto the road at that turn because yours isn't the only Tesla, I'm sure, that tends to do that in that location.

Leo: Yeah, yeah.

Steve: And so it would make sense for either it to be built into the car's database, a better awareness based on knowing where it is, which probably it's accumulating over time; or maybe let roads that are less traveled have some sort of way of providing that feedback to the car.

Leo: Cadillac CT5 will not let you use Autosteer except on highways it has mapped.

Steve: Ah, okay.

Leo: I think for that reason. I mean, everybody's trying to solve this. Also the Cadillac, the way the Tesla works, it's the torque on the steering wheel. It has a torque sensor. So you have to kind of slightly move the steering wheel to let it know you're holding onto it. Cadillacs use a capacitive sensor, so just your touch is enough. And they have a camera watching your eyes. And if you look away from the road, it rumbles your seat vigorously. So really they - I think, look, two 737 MAXes crashed because apparently the auto stall feature that was supposed to pull the nose down did it incorrectly, and pulled it down into the ground.

Steve: Right.

Leo: It's a very similar problem. And pilots who didn't know enough to disable it, that's what happened. So I think it's just - autopilot is always going to need, at least for a while, anyway, human intervention.

Steve: Well, and consider the lawsuit. I mean, there's just no way these car companies are not needing to be able to say we took proactive measures for this to only be an assist function, not a "roll up in the back seat and take a nap while we drive you to work" feature. So, yeah.

Leo: Yeah. A really interesting topic.

Steve: Yeah. So, oh, this is a classic hack. Of course we talked last week about ASUS's ShadowHammer MAC addresses, well, the ASUS ShadowHammer attack, how two of their download servers were infected with multiple versions of malware over a duration of five months, presumably by someone who got an advanced persistence presence in their system and was able to do this. In reporting I did note that they were only laptops. So that's significant because remember that one of my - as I was scratching my head, brainstorming, where could a list of MAC addresses have been resourced, one of them was from WiFi hotspots in a mobile scenario.

Leo: That's right. They see the MAC addresses, don't they.

Steve: Yes. Mobile hotspots get the MAC addresses. And the other interesting thing was it turns out there was a list, a further refined list of double MAC addresses where it was the LAN and the WiFi MAC address which was known. So I don't know what that further tells us, but that would potentially...

Leo: That's interesting. That confirms that, I think, yeah.

Steve: One of my hypotheses is that they had seen them roaming. They knew who they were. And so they were going to come back and get them. Okay. So anyway, as I described last week, what Kaspersky did was they offered an online resource where people could put their MAC addresses in, and it would tell them whether they were of those 617, I think it was, addresses; or a downloadable tool. If you didn't want to put your MAC address into Kaspersky's page, you could download a standalone EXE that contained them all. Well, okay. So get this. For whatever reason, they chose not to publish their full list of MAC addresses; right? It was submit it to us, and we'll tell you; or download this EXE. Well, they obscured the MAC addresses by hashing them with a salted hash, a salted SHA-256 with a complex algorithm that merged the MAC addresses and the salt several times in the hash in order to make it, you know, they just made up their own hashing function, essentially.

Well, this apparently bothered some guys at an Australian security firm, Skylight Cyber. They wrote: "The question of who did this and why is intriguing, but not one we were trying to answer in this case. First things first. If information regarding targets exists, it should be made publicly available to the security community so we can better protect ourselves. Kaspersky have released an online tool that allows you to check your MAC address against a database of victim MAC addresses, which is hidden. Good on Kaspersky, on one hand. But on the other hand" - And "good on," of course, they are Australian. Good on Kaspersky. "But on the other hand, this is highly inefficient and does not really serve the security community.

"So we thought it would be a good idea to extract the list and make it public so that every security practitioner would be able to bulk compare them" - that is, the whole list - "to known machines in their domain. If you are interested in the list, it can be downloaded here or here for the extended list." And I have a link to this page in the show notes where those "here" and "here" are links to the extended list. And I also have that actually down below.

So these guys also felt that having a simple list of targeted victim MAC addresses would be far more useful for large enterprises with many hundreds of thousands of systems where the stakes were pretty high. Because, after all, we're talking about the reliable installation of a trojan backdoor by unknown actors into specific laptops, when who knows whose. You know, specific ASUS laptops.

So how do we solve this problem, that is, the problem that these guys faced? Well, of course it's a variation of the classic brute force password cracking problem. Although it's significantly simplified because in this case we know that every test MAC address is a 48-bit binary input to the cracking hash function. And we know that half of it will be one of a handful of 24-bit vendor MAC prefixes within the 48-bit binary. So it's like a password whose length we exactly know. And in fact half of it is one of a subset of possible 24-bit chunks.

So the Skylight Cyber guys calculated that their own fastest - oh, first of all they reverse engineered the algorithm because there it was, sitting in an EXE. They used IDA, the interactive disassembler. It'll be fun when in the future we start hearing about them using the NSA's tool, but that'll take a while to proliferate through the ecosystem. They figured out exactly what the hashing function was. They then designed - they took Hashcat and tried to use it, but the function was custom. So they customized and built a custom version of Hashcat to reverse the Kaspersky hash functions. Their own fastest system running Hashcat, they figured, would require about 162.5 days to extract all of the MAC addresses from Kaspersky's offline checking tool.

So they thought, hmm. Let's hire out. They did the entire job in less than an hour by renting an Amazon AWS p3.16xlarge instance, which comes equipped with eight Nvidia V100 Tesla 16GB GPUs. So they figured out the algorithm using a reverse disassembler, verified it using Hashcat, realized, okay, 162.5 days, too long. So they simply reimplemented it under an Amazon AWS instance and broke down 583 of the total of 619 it was, MAC addresses in less than an hour. There is a beautiful list that they linked to, and I have linked to in the show notes. Just it's numerically sorted, right down the left-hand page, 583 lines, every MAC address that was being targeted.

So now we all know. And they're guessing that the reason they're missing a few is that they did take advantage of every trick in the book, that is, to shorten the guessing time, they were only guessing the known handful of 24-bit addresses. It would have taken substantially longer, but certainly not impossible, obviously. And also at much more expense. I don't know what the renting Amazon's AWS instance with the eight Nvidia V100 Tesla 16GB GPUs would have cost. But running that thing so that it's smoking for any substantial period of time probably would have run up the credit card bill. So anyway, 583 out of 619. We now know what they are. And just a tip of the hat for these guys pulling off what is kind of a cool hack.

I did want to mention, just come back to noting that we are officially, as of yesterday - oh, no, I'm sorry, last Thursday it was - Microsoft announced that they are now designating Windows 10 October - that is, last October - 2018 Update Build 1809 as ready for broad, rather than its previously targeted, deployment. Their Windows as a Service evangelist John Wilcox said: "Based on the data and the feedback we've received from consumers, OEMs, ISVs, partners, and commercial customers" - in other words, nobody said no - "Windows 10 version 1809 has transitioned to broad deployment. With this, the Windows 10 release information page will now reflect Semi-Annual Channel (SAC) for version 1809." He said: "We will continue to communicate for future releases the transition from targeted to broad deployment status."

And then just a very quick note that I just got a kick out of the fact that, not coincidentally, VMware has released a bunch of fixes for their VMware software. When I saw that, I fired up my copy of VMware. I am on v14.1 point something. And sure enough, it came up and said, oh, we've got updates for you. Would you like them? They're free. And I said yes. And so I am now on whatever it was on the other side of that. And of course, as we know, the Pwn2Own guys did find some rather sobering problems with the currently most recent updates of VMware. So the good news is that has been fixed.

Okay. Three quick things, or four, I think. Steve Watson in the U.K. asked of SQRL adoption. Oh, actually today, April 2nd. He wrote: "Hi, Steve. I've been listening to the Security Now! podcast for a few years now and think it's great. Thank you for the continued effort you put into producing it. Makes my journeys to work a lot less boring. I recently set SQRL up on my phone and computer and think it's great. I just wondered how you're going to get large companies to adopt this technology."

And so for what it's worth, A, it's free. It's way better than anything else. It is multiplatform. All platforms are supported. It's easy to deploy. And the earliest response I received when I started talking about that to our listeners was from enterprise. Enterprise has a, you know, they're a more or less closed ecosystem. They really want a solution to allow essentially a single sign-on sort of thing for their employees. So the idea that they could adopt this for their own enterprise websites, which they absolutely can, rather trivially, and then all their employees can use a zero-cost, extremely secure solution, that's compelling. And so I expect that we're going to see enterprise adoption happen as, like, on the leading edge of this overall.

Seth G. in North Carolina, he said: "'The Expanse,' oh, wow." And actually his is representative of many pieces of feedback I got after just reminding our listeners about it, or mentioning that it was on Amazon prime. Now, I don't know where Seth lives. Well, we do know where he lives. He's in North Carolina. I don't know where he works, and I don't want to know, and I'm glad I don't know his last name because I don't want to get Seth in trouble. He said: "Steve, things have been slow at work, and I'm blitzing through 'The Expanse.'"

Leo: Maybe he's a security guard. It's okay.

Steve: That's right. "I'm blitzing through 'The Expanse' on Amazon Prime." He says: "This is sci-fi at its best. Thank you for doing us a public service and letting us all know about this series. It makes me want to read the books, and I rarely get to read more than manual pages and whitepapers these days. All the best for you as we go into the 700s on SN, and congrats on nearly finishing SQRL. Hopefully we will see some more SpinRite soon, like you've been talking about on the podcast. Kind regards, SG in North Carolina."

And I should mention of SQRL, where I am at this instant is I'm just, like tonight probably this may happen, I'm wrapping up the fixed content that the site needs, separate from all of the user interactive stuff which will give the site its life, so the how do I start, the user Q&A. What I'm finishing up is the "what if" section. It occurred to me some time ago, and I've said so on the podcast, that I realized at one point I now had an answer for every possible "what if" question anyone could ask. But what if this? But what if that? But what if this? But what if - anyway. So I created a thread in the forum to solicit everybody's, no matter how hare-brained they are, you know, what if I die? That's not so hare-brained. We've talked about that problem on the podcast. What if my parents die? What if my dog eats my printed copy of my identity? I mean, anyway.

So not all of them made the cut. But I have 40, 40 different, like, every possible variation of what could go wrong. And so what I did was I wrote up all the answers, then I posted it to an open community editable Google Doc, which they've now been attacking for about five days, while I wrote a translator to turn that into the funky Q&A expandable/contractable format that XenForo uses. And I finished that. So I will return to the Doc. I will edit everyone's comments and edits down into a final version, or at least v1, then run it through my translator to convert it into this wacky BB code stuff, and then post it. And at that point the fixed content for the forum is done. And then I turn my focus on GRC's very old original web pages, bring them current.

And meanwhile, in the background, I should say that a lot of great work is happening on the three other main client platforms - iOS, Android, and the web extension. So those are quickly getting - they're working, but they're not yet feature complete. And it's confusing people who use them, like wait a minute, I created my identity here. Why can't I export it? It's like, oh, well, that feature's not yet there. My Windows client, of course, had the advantage of coming first, and where all of this was prototyped. So it's been feature complete for quite some time. So anyway, we're getting very close. And people are universally loving it, so that's fun to see, too.

Fabio Esquivel in Costa Rica, his subject was "Alternate PDF Reader." He said: "Hi, Steve. I wonder what do you use or recommend as an alternate PDF reader, hopefully free, but otherwise affordable." He says: "I don't want to pay Adobe for its insecure and bloated software. So I've been using Sumatra PDF for years now. But it seems abandoned by the developers. Last update was in August 2016, and I fear it's subject to recent vulnerabilities that go on unpatched. Thanks for your suggestions."

Well, he's not going to like what my solution is, and we know I'm something of a dinosaur. On the other hand, I'm not opening PDFs like in email ever. I just don't. So what I'm using is the very - and I also like using sort of more authentic PDF. And I should also mention, just to be fair, our browsers now contain built-in PDF readers. Whatever Google is using in Chrome to view PDFs, that's going to be state of the art. That's going to be secure. And Firefox opens PDFs for me, too. And I don't know that I have installed a third-party reader.

But I'm using Acrobat, or Adobe Reader 9.5, which is old, but it's real Acrobat, so it never - there's never anything in a PDF that it doesn't know how to open. And of course I do not have it associated with my browser. I have JavaScript disabled in it. So I've tightened it down so that it's not able to do the bad things that a default wide open Adobe Reader could do back then.

And many people don't know, but Adobe maintains an FTP server, speaking of FTP. I put a link in the show notes. But if anyone is listening and is curious, in your browser, since our browsers will still allow us to browse FTP, type ftp://ftp.adobe.com/pub/adobe/. Now, that will take you to Adobe's product list. And in there is Reader; and in there are all the platforms, Mac and Windows and so forth; and in there are all the versions. And you can get 9.5, and it doesn't need a license. It's freely available. It's real Adobe. And anything after that is get on the signup and be a subscription. And I just don't roll that way.

So on one hand I would just use the reader that's built into your browser, which is certainly being maintained by both browser vendors, Chromium and Firefox. And I assume Edge has the ability to read PDFs, too. And so that's probably being maintained. If you like to poke around on Adobe's FTP server, as I imagine some of our listeners may, it's there.

And, finally, SpinRite. David W. Roscoe in North Andover, Massachusetts. Subject is "Becoming a SpinRite Volunteer." He said: "Hello, Steve. I'm the person who volunteered the testimonial about using SpinRite to repair my brother's desktop music studio which you read in SN-707." He says: "I thought I heard you say in a recent Security Now! episode that you were interested in acquiring more volunteers for testing the next versions of SpinRite. If this is true, then I hereby declare myself available for becoming one of those volunteers. If you're interested, please tell me what I need to know. Thanks again for all you do. David Roscoe."

So anyway, to David and all of our listeners, any SpinRite owner will be able to use their SpinRite serial number or their transaction number from their purchase, either way, to obtain the prerelease of SpinRite once I return to it and begin working on it. That's where it was when I paused it - a rather long pause, yes, I know - to get SQRL done. Even I've heard from people who were furious at me for doing that, until they experienced logon with SQRL. And I've read several people who have recently said, "Okay, you were right. This has to happen. So I'm glad it has, and I still look forward to SpinRite." Of course, I do, too. I can't wait to get back to it. I miss it. It'll be really fun not to be doing something else finally.

So anyway, once we get there, I will explain how that works. And as I have releases, incremental things that are working, I will certainly tell our listeners who have been very patient with me and very supportive for lo these many years, where they can play along with SpinRite as we go. So we'll get there. And it's looking like it's going to happen before long.

So Android security, 10 years in, since this is the 10th birthday.

Leo: Hard to believe, yeah.

Steve: And really, hasn't it gone fast? I mean, I didn't realize we were doing the podcast when it happened.

Leo: I still have the first Android phone right here. Well, I won't get it now. But it's right back there in my museum of old crap.

Steve: Nice.

Leo: Yeah. It's a terrible phone. But as you say, we've come a long way.

Steve: We have. So, and really I want to talk about exactly that in detail, how we've come from a security standpoint. In the show notes I've got a link to the Google Android Security 2018 Report Final, as they call it. It's a PDF, 31-page report, which examines and shares the statistics of what they recognize - Ecosystem Data, the benefit of what they call Google Play Protect, the Android Platform Security, and then essentially the threats that are out there, the very aggressive PHA families, the Potentially Harmful Applications.

They write that: "The Android security team's mission is to protect every one of the more than two billion Android users." They said: "We do this through massive investment and continuous improvement in our security and privacy technology and operations." And I should explain I've cut this down, and I'm going to cut it down ever more as I go through this because we're only halfway through the show notes. There was just - there was so much good stuff here, I had a hard time, like, oh, god, I have to put that in. I have to put this in.

And so I said the report did share some interesting and impressive stats. For example, the broadest statistic for measuring device hygiene is how frequently a full-device scan detects PHAs, Potentially Harmful Applications. They said: "Google Play Protect, Android's built-in defense mechanism, is very effective in keeping PHAs out of Google Play; but malicious apps can still be downloaded from other sources." And of course this is apropos of the Picture of the Week because it demonstrates the benefit of sticking within Google Play.

They said: "These apps endanger not only the device, but also threaten the sanctity of the Android environment. This is why Google Play Protect scans all apps installed on a device regardless of the source." Right? So whether you get it from Google Play or outside. "In 2018 only 0.08% of the devices that used Google Play exclusively for app downloads were affected by PHAs. In contrast, devices that installed apps from outside of Google Play were affected by PHAs eight times more often." And again, Leo, as you observed, eight times 0.08 is, you know, 0.64. Still not, like, horrible, but eight times more often.

They said: "Compared to the previous year, even those devices saw a 15% reduction in malware due to the vigilance of Google Play Protect." So essentially, as we know, relatively late in the game, right, because here we are at 10 years, and this happened in 2018. So within the last couple years they decided, okay, we're going to have to get even more proactive than we have been. They did, and it has had a significant reduction effect.

They said: "Android's security saw a strengthened application sandbox in 2018, along with hardened developer APIs" - which we've been talking about all along as these

increases and improvements have occurred - "with features like Biometric Prompt and an updated target API level for apps. The Android Security team continued their investment in hardware-backed security through discrete tamper-resistant secure elements that enable the use of industry-first security APIs, such as Protected Confirmation and Strongbox." They also note that in 2018 they surpassed $3 million in reward program payouts, meaning that they're proactively soliciting developers and hackers who are able to find problems and then show, like turn them over for a reward.

They said: "Our Android security rewards programs allow us to work with top researchers from around the world to improve the security of the Android ecosystem. These programs offer some of the highest priced rewards in the industry. Through a combination of platform improvements like Treble, new original equipment manufacturer agreements" - and we know about those, remember, they tightened up the OEM agreements to require a maximum duration of was it six or nine months? It still seemed like a long duration, but at least it was there for the improvements that Google makes available to be pushed out to the consumer. And, they said: "...and partner programs such as Android Enterprise Recommended, the Android ecosystem," they say, "has made significant progress in releasing security updates. In the fourth quarter of 2018," they write, "we had 84% more devices receiving a security update than the same quarter the prior year."

So that's - think about that. In the fourth quarter of 2018, so just this last previous calendar quarter, we had 84% more devices receiving a security update than the same quarter the prior year. Now, that's great. That's nearly double. But on the other hand there are two million devices out there, and we know many of them are not getting any updates. So what they're not saying is the truth of the fact that many devices are woefully unupgraded. And I don't think that's a problem that they can fix. I mean, they want to provide an OS for, Leo, as I said, and you got a kick out of, "fringe partners." And fringe partners are going to be fringe. That's what they're going to do.

Leo: Fringy.

Steve: So they're going to be fringy, yes. So as we've said, if you want to use Android, great. Go with a mainstream partner. Go with a partner who today, the day after Google's release - and, by the way, Google did provide the April Fools' Day updates a month prior to all their partners. You should already have the update on your device. If you do, you're with a partner that you know is keeping your device current. If you don't, well, the supplier of your Android device has had them available for more than a month. And there are two very potent remote code execution vulnerabilities in it that should be fixed. So buyer beware.

Under the topic of platform security and privacy, they said: "Improving Android's security with every major Android release and monthly security update is critical [yeah]. However, in order to be even more effective, we must work to continually increase security without putting a burden on our end users. A layered security model is part of our fundamental design principle [good] and is a foundation of Android's architecture. The Android platform controls how the operating system works and how apps interact with other apps, device hardware, and other services. Supported by Google Play Protect" - which is like Windows Defender, it's in there, and it's looking at everything - "Android is protected around the clock," they say.

The following table lists some of these protections that are designed to provide better platform-level security. And this is stuff we've talked about. Encryption protects data from unwanted access. Hardware-backed security strengthens key storage and cryptographic services and enables strong remote authentication. Kernel self-protections

protects the kernel against memory corruption vulnerabilities and other security flaws in kernel components and drivers. Sandboxing keeps each app in a separate space, protecting its data and processing from other apps. SELinux provides an auditable definition of and enforcement of security boundaries on all operating system and app components above the kernel. User space hardening protects the operating system and apps against memory corruption vulnerabilities and other security flaws; includes address space layout randomization, data execution prevention, and control flow integrity. Verified boot cryptographically verifies that the operating system starts in a known good state.

So what we just ticked off there, I mean, that is the state-of-the-art menu for what you need to do today in a hostile environment to protect the user. And it's there. It wasn't there 10 years ago. It wasn't there five years ago. But the point is today's Android platform is now state-of-the-art security, given that you maintain communication with your provider and they with Google, that being the last piece of this.

They said: "With Android 9 we added a myriad of great security features. We strengthened the application sandbox and hardened the developer APIs." And we've talked about that as that has happened. "We continued to invest in hardware-backed security via the trusted execution environment and on selected devices through discrete tamper-resistant hardware. We also layered a set of privacy-preserving enhancements and adopted more anti-exploitation techniques so that bugs don't turn into exploitable vulnerabilities."

On the topic of security research competitions and zero-day vulnerabilities, they said the Android Security & Privacy team participated in a number of external vulnerability discovery and disclosure competitions, including Mobile Pwn2Own, which took place at PacSec conference in Tokyo. We talked about that this summer. And they said: "At this event, researchers were rewarded for demonstrating working exploits against major mobile operating systems. Exploits against Google Pixel devices were categorized in the top reward category, along with other devices such as the iPhone. No exploits successfully compromised Google Pixel devices, and none of the exploits demonstrated against devices running Android utilized a security vulnerability in the Android operating system." So that's significant. They were there, they were dangling a big carrot, and nobody got it.

"Further," they write, "in 2018, no critical security vulnerabilities affecting the Android platform were publicly disclosed without a security update or mitigation available for Android devices." So that says there were no zero-days on Android last year. And that's significant, too. Again, it matters that the devices are in the update loop. But where they are, and that's the only thing Google has control over, this demonstrates nothing but success.

They have an Android update program. They said: "Google mitigates security vulnerabilities discovered through the Android Security Rewards program and additional engagements through regular Android security updates. In 2018, we continued to work with Android device manufacturers, mobile network operators, and system-on-chip vendors to increase the number of devices receiving regular security updates. Through our combined efforts, which include platform improvements, new OEM agreements" - meaning, yes, we're going to make you do this legally - "and partner programs such as Android One and Android Enterprise Recommended, we've made significant progress in releasing the latest security updates.

"In fact, in the fourth quarter" - oh, and here it is, they're repeating their stat - "we had 84% more devices receiving a security update than the same quarter the prior year." On the other hand, they don't tell us what percentage of all two billion. So, yeah. Again, a hard problem to solve. "As of December 2018, over 95% of deployed Google Pixel 3 and

Pixel 3 XL devices were running a security update from the last 90 days." So again, as of December 2018, so end of last year, over 95% of those two devices, the Pixel 3 and the Pixel 3 XL, were running a security update as recent as the last 90 days. So they are staying current.

On system image scanning they said: "In the Android Security 2017 Year in Review report" - meaning the previous report of this - "we announced that we had begun scanning for preinstalled PHAs" - again, we know what those are - "across many software builds for devices with Google services. In 2018 we expanded this program and launched it as Build Test Suite (BTS) for partner OEMs. BTS is similar to the Compatibility Test Suite (CTS). OEMs submit their new or updated build images to BTS. BTS then runs a series of tests that look for security issues on the system image.

"One of these security tests scans for preinstalled PHAs included in the system image. If we find a PHA on the build, we work with the OEM partner to remediate and remove the PHA from the build before it can be offered to users. During its first calendar year, BTS" - and that was last year, 2018 - "BTS prevented 242 builds with PHAs from entering the ecosystem. Anytime BTS detects an issue, we work with our OEM partners to remediate and understand how the application was included in the build." That is, how did something not wanted, potentially harmful applications, get into the build that this OEM vendor submitted to the BTS program for scanning? They said: "This teamwork has allowed us to identify and mitigate systemic threats to the ecosystem." Like they had to explain why that's there. And that fixed some policies that these guys had.

"This teamwork has allowed us to identify and mitigate systemic threats. Through the BTS program we discovered, analyzed, and remediated PHA families such as Chamois and EagerFonts, which are described in detail in the PHA Families section. In 2019 we are continuing our commitment to vetting approved Android devices for security issues." So that's very cool. They made it very simple for their partners to submit an image which they are intending to burn onto their Android devices, and 242 of those submissions have been found containing something unwanted. So that's very cool.

Of PHAs, Potentially Harmful Applications, they said: "Potentially Harmful Applications are apps that could put users, user data, or devices at risk. Common PHA categories include trojans, spyware, and phishing apps. In 2018 we started tracking click fraud as a PHA category. Click fraud apps simulate clicks on advertisements without user consent." User-wanted PHAs, which I guess exist: "Some apps with attractive features also weaken Android's built-in security. When users try to install these apps, Google Play Protect warns users about potential hazards so they can make informed decisions. Our statistics separate these from classic malware PHAs. For example, Google Play Protect warns users about apps that disable Android security features, such as SELinux, or root the device with disclosure and user consent.

"Google Play Protect discourages changes that lower Android's built-in security protections, but allows individuals to choose the risks they're willing to take with their devices. A warning message is displayed to the user anytime a PHA installation is detected. If they decide to ignore this warning and proceed with the installation, they will not receive further security warnings about the app. Interrupting the Google user experience with constant warnings would make Google Play Protect more annoying than useful. In 2018 user-wanted PHAs comprised 0.11% of app installations downloaded outside of Google Play." And then they said Google Play doesn't allow any security-breaking apps, even if they are user-wanted.

They also have a category, mobile unwanted software. And I'll just summarize, saying that, which they are looking at, things that grab your phone number, your primary email address, information about installed apps, any information about third-party accounts, without user consent, they consider that mobile unwanted software. With this change the

total number of installed attempts coming from what they call MUwS apps declined from 2.09% in 2017 to 0.75% in 2018. So about to one-third of what it had been before. They note that they are now considering click fraud apps as PHA culprits. They talk about the threat landscape changing.

In 2018 there were two notable changes to the Android threat landscape: an increase in preinstalled PHAs, meaning from vendors; and backdoored SDKs, software development kits. They said of the preinstalled PHAs: "Malicious actors increased their efforts to embed PHAs into the supply chain using two main entry points: new devices sold with preinstalled PHAs, and over-the-air updates that bundle legitimate system updates with PHAs. Neither entry point requires action from users, making them difficult to defend against."

There are three possible reasons for an increase in the number of preinstalled PHAs. First, the developers of preinstalled PHAs only need to deceive the device manufacturer or another company in the supply chain instead of a large number of users, so it's easier to achieve large-scale distribution. Even a less popular device model can compromise hundreds of thousands of users through one preinstalled harmful application.

The second instance is preinstalled PHAs can gain more privileged access to the device, so it's easier to execute malicious behavior that would usually be blocked by Android's security model. At the same time, these additional privileges allow PHAs to defend against security tools or removal attempts by users. So they enjoy sort of a more protected status.

And, finally, third, large families of PHAs used exploits to root devices, but they say this is increasingly more difficult due to Android's constantly improving security model which blocks privilege escalation exploits to achieve similar privileges and defense levels for regular apps. Developers of these apps know that it is easier to compromise the supply chain of device manufacturers than to attack the Android platform security model.

So I do think it is interesting that the Android supply chain has come under really aggressive attack by entities, you know, bad actors that recognize, hey, if we can just convince someone to get this into their phone, then we score a major win. They said: "To combat the problem of preinstalled PHAs, the Android Security team launched a security program in 2017 as part of the Android device certification process. We expanded the program in 2018, and now every new Android-certified device goes through the same app scanning process as apps on Google Play. Additionally, our security scanner looks for other common privacy and security issues and denies device certification until device manufacturers fix these problems."

And the other source was backdoored SDKs, as we mentioned. They say: "Some SDKs appear legitimate, but include behaviors and functionality that the app developer may not have known about when they included the SDK. This functionality may compromise user data, device integrity, or experience. It may also be used as a part of a larger initiative, such as committing click fraud, mining cryptocurrency, or app install attribution fraud. Here are some approaches, they write, developers use to include malicious code in legitimate SDKs."

There's four: Backdoored SDKs that otherwise have legitimate functionality; backdoored Android system code that injects the malicious code into every app on the device; modified Google apps with backdoor code injected; and, finally, modified SDKs rehosted with similar names to confuse legitimate developers into accidentally downloading them. "Hundreds of apps," they say, "have been affected by backdoored code. We have been working with impacted developers to educate them about this new threat and to publish updated versions of their apps without the backdoor code."

And then they talk about the device and ecosystem hygiene. They say: "The broadest statistic for measuring device hygiene is how frequently a routine device scan detects PHAs." This is recapping what we talked about at the beginning. "Since we began to measure device hygiene in 2014, an average of less than 1% of devices have Potentially Harmful Applications installed at any point. This trend remained steady in 2018." And then they have a bunch of stats.

What's interesting is there is a big difference, we see a big difference by country. I have a chart here in the show notes showing the potentially harmful application presence rate found in 2016, 2017, and 2018. The steepest decline was in India, where they started in 2016 with the highest PHA rate, and now they're still, well, they're number two, with the U.S. being currently the highest, but everybody much lower. They said: "In India, which is by far the biggest Android market, the number of devices affected by PHAs has decreased each year. In 2018, 0.65% of all Indian devices were affected by PHAs at any time, a 35% drop from the previous year." And it was a very substantial drop from the year before also.

They said: "As in previous years, the U.S.A. is the second biggest Android market." So India number one; U.S.A. number two. They said: "In 2018 the number of impacted devices rose from 0.4% to 0.5% due to the introduction of click fraud as a PHA category." So it didn't actually increase, as I mentioned. They redefined PHA. They said: "However, compared to India, the U.S.A.'s context for PHAs is different and less severe. Eight of the top 10 PHAs in the U.S.A. were wanted by users or don't significantly impact them directly. Of these eight, four are power user tools for rooting devices or for circumventing other security settings, and four are click fraud apps."

So in other words, these are half of the things that Google considers potentially harmful are things that cutting-edge hackers chose to install because they want the benefits that they are introducing. So overall, I think pretty good security.

I'm trying to see if there's anything else here that jumps out at me. Oh, over versions, device hygiene by Android version. Newer versions of Android are less affected by PHAs. Android 8 and Android 9 have PHA rates that are 0.19 and 0.18 respectively. They say: "We attribute this trend to advancements in platform security. In particular, newer Android versions are more resilient to privilege escalation attacks that previously allowed PHAs to gain persistence on devices and protect themselves against removal attempts. On newer versions, GPP is effectively cleaning PHAs. In conjunction with platform changes, GPP is preventing PHAs from protecting themselves from removal or being disabled."

And, let's see, anything else? Outside of Google Play, backdoors is a problem. In 2018 backdoors were the most prevalent PHA category outside of Google Play, where they made up, get this, 28% of all PHA installs. So more than one quarter of outside of Google Play potentially harmful, and I would say in this case obviously harmful, were backdoors. As the prevalence of trojans and hostile downloaders decreased in 2018, backdoors took the top spot. So that's happened. That's reversed. Then they go through the PHA families. And basically these are all the things that Google Play Protect, that's GPP, which is their background scanning, protects for.

There's the Chamois, or I guess it's Chamois family, which is one of the most impactful PHA families. SnowFox is an advertising SDK with two variants. One variant steals OAuth tokens from a device; the other injects JavaScript for click fraud into WebViews with loaded apps.

Cosiloon, C-O-S-I-L-O-O-N, is a family of hostile downloader PHAs that was preinstalled on uncertified Android devices. Cosiloon apps are two-stage PHAs with the first stage preinstalled on the device. There are two variants of it.

There's BreadSMS, a large PHA family that Google Play Protect began tracking at the beginning of 2017. BreadSMS evolved rapidly in 2018, accumulating over 11 million installs, with approximately 98% of those occurring on Google Play. In 2018, BreadSMS added cloaking and obfuscation techniques to evade detection.

Then there's View SDK. View SDK is a monetization SDK that uses JavaScript to perform ad click fraud. View SDK was originally discovered by Google Play Protect in December of 2017. However, Google Play Protect didn't begin treating click fraud as a PHA category until March of 2018. Affected apps drop a JAR file containing View SDK during execution. View SDK then downloads JavaScript from a remote server and injects it into WebView, showing ads and triggering fake ad clicks without user intervention.

There's Triada. Triada software was first documented by Kaspersky in 2016 as a rooting trojan. In mid-2017, Dr.Web documented a preinstalled backdoor variant where the log function in the Android framework of the device's firmware was being backdoored. Any application related to Triada can then communicate and perform commands using the backdoored logging method. And actually we talked about that at the time. I remember the backdoored log.

There's CardinalFall, a large PHA family with an SDK that implements click fraud. FlashingPuma is another large click fraud family discovered in 2018. EagerFonts is an SDK embedded in the Fonts apps that comes preinstalled on some Android devices. Idle Coconut is an SDK that developers include in their apps for monetization. The apps include - I wonder who's being monetized. The apps double as endpoints of a commercial VPN that routes traffic through affected Android devices. SDKs use a WebSocket for communications with a command-and-control server and then connect to hosts that command-and-control commanded over normal sockets. None of this behavior is disclosed, and the user's device unknowingly becomes used as a proxy network. Thus Google flags these as trojans. About 60% of the installs of Idle Coconut in 2018 came from Google Play, and approximately 40% were from sideloaded applications.

So after the 12.5 years that we've been doing this podcast, I would imagine that our listeners likely have a realistic and sober appreciation for the difficulty of the task that Google has willfully undertaken by shepherding something like Android. It's a challenge I would never want to face. They are placing, as I mentioned at the top of the show, a highly capable and powerful open source operating system running atop equally powerful hardware, sourced very indirectly in most cases through partners far and wide and largely out of their control. They're putting these into the hands of inherently trusting, non-computer-savvy consumers; and, for the most part, in an openly hostile environment, against quite unfavorable odds. And I would argue that it's working, that they are doing a very good job of this. So to all of this I say bravo, Google, and thank you.

Leo: Nice job. As an Android user I am grateful.

Steve: Yeah.

Leo: Do you think it's comparable to OS security?

Steve: No. But it's not a fair comparison because iPhones come from no other vendor, and Apple has absolute control over it. They have a lockdown boot process. I mean, they make mistakes, too. Bad stuff gets into the iTunes store. But the power of Apple producing every iPhone cannot be underestimated. But it's also the reason they're much

more expensive than a lower end Android device. If you want a value, I would argue without question Android gives you a better value.

Leo: Oh, well, there's no doubt about that.

Steve: Yeah.

Leo: Although if you want secure Android, you're probably going to go with Samsung and their Knox implementation.

Steve: Correct.

Leo: And then in most cases you're spending as much as you are for an iPhone.

Steve: Correct. And I think that essentially is a perfect "you get what you pay for" in the security model because you really, really, really want it. To get the most security, you just need to have a locked down device. I mean, just think about the enumeration of all, I mean, all of those things I just went through? Those are people somewhere out there desperately trying to get in. I mean, when we began the podcast, this would have seemed like science fiction to us. Oh, how cute, a Honey Monkey. You know? Now it's like, oh, my god, how do you even turn your device on?

Leo: Yeah, yeah. We've come a long way.

Steve: Yeah, we have.

Leo: And we have a lot farther to go. I mean, that's really the story of this show.

Steve: Well, that's why a couple weeks ago I suggested that hackers could, like, earn a living by finding problems. You can. You can pay your rent by, if you're good, and you find problems in products, everybody wants to pay so that the bad guys don't get them ahead of time. And so it's a new income stream. I think it's great.

Leo: And the old adage is that security is not a state, it's a process.

Steve: Right.

Leo: And that's the process we are engaged upon each Wednesday, I'm sorry, Tuesday here on Security Now!.

Steve: Yup.

Leo: Thank you, Steve. We do this show Tuesdays at about 1:30 Pacific, 4:30 Eastern, 20:30 UTC. If you want to watch or listen live, you can do that at TWiT.tv/live. You can get on-demand versions of this show from Steve's site, GRC.com. He's got audio. He's also got something unique: transcripts. So if you like to read along while you listen, I know a lot of people do, that's a good place to go, GRC.com.

While you're there, of course pick up SpinRite, the world's best hard drive recovery and maintenance utility. You could check up on the progress of SQRL, find out how it works. There's lots of fun stuff, all of it free except for SpinRite. So enjoy Steve's site, GRC.com. On the Twitter he's @SGgrc. He takes DMs there, but also you can leave a message at the website, GRC.com/feedback.

We have copies of the show, audio and video. That's unique to us at TWiT.tv/sn for Security Now!. Or you can subscribe in your favorite podcast application, and CacheFly will push it out to you the minute it's available late on Tuesday. Thank you, Steve.

Steve: Thank you, my friend.

Leo: Have a great weekend. We'll see you next time on Security Now!.

Steve: Talk to you then.