SECURITY NOW!

Transcript of Episode #803

Comparative Smartphone Security

Description: This week we look at the updates in release 88 of both Chrome and Edge with their evolving password manager features. We also look at two recent headshaking consequences of the hard end of life for Adobe's Flash. Ransomware gangs have added another new incentive for payment, and additional details continue emerging about last year's SolarWinds attacks. We have newly disclosed discoveries from a Google Project Zero researcher, and I spend a bit of time wondering out loud how we're ever going to change the low priority that's currently being given to serious security problems that don't directly inconvenience end users. And we finish by examining a very useful analysis of the comparative security of iOS and Android recently published by Johns Hopkins' Matthew Green and team.

High quality (64 kbps) mp3 audio file URL: http://media.GRC.com/sn/SN-803.mp3
Quarter size (16 kbps) mp3 audio file URL: http://media.GRC.com/sn/sn-803-lq.mp3

SHOW TEASE: It's time for Security Now!. Steve Gibson is here. We answer some important questions. Should you use your browser's password manager? Should the South African Revenue Service have used Flash? We'll talk about the SolarWinds hack. And then, finally, Matthew Green's head to head between Android and iOS. Which is the most secure? It's all coming up next on Security Now!.

Leo Laporte: This is Security Now! with Steve Gibson, Episode 803, recorded Tuesday, January 26th, 2021: Comparative Smartphone Security.

It's time for Security Now!. Lo and behold, here he is, the man of the hour. He appears magically every Tuesday over my left shoulder, Mr. Steve Gibson. Hello, Steve.

Steve Gibson: It's just like magic, Leo.

Leo: Just like that.

Steve: It's just, you know, and I have somehow this 20-page PDF all just like popped into existence.

Leo: I'm sure you'd love that to be a little more magical. You do so much work on this show, and I don't think I say it enough. I'm very grateful because this is an amazing feat you pull out. I mean, this is longer than writing a column for InfoWorld, for sure; right?

Steve: It's way more than that. I know that our listeners appreciate it. I get a lot of feedback from you and from them. And I don't know how to do less. I just, you know, stuff is happening, and I want to talk about it. In fact, I got a little...

Leo: We're going to put that on your tombstone: "He didn't know how to do less."

Steve: I didn't know how to do less, yeah.

Leo: That's great. I love it.

Steve: So this week we're going to look at the updates in releases 88 of both Chrome and Edge with their evolving password manager features. And, you know, it was inevitable that browsers were going to get more aggressive about being password managers. It just sort of seems like a natural thing for them to do. And really, in retrospect, it's weird that it's taken them so long to get there.

We're going to look at two recent headshaking consequences of the hard end of life for Adobe's Flash, which I think caught people by surprise because they probably didn't realize that Adobe had put a time bomb in the most recent Flash versions that was going to cause it to simply say no after January 11th, which it did. Also ransomware gangs have added another new "incentive" - I should have had that in quotes - for payment.

And additional details continue emerging about last year's SolarWinds attacks. Microsoft has done just a beautiful reverse-engineering analysis that we're going to take a look at this week. We also have newly discovered or disclosed discoveries from yet another Google Project Zero researcher. And then I'm going to spend a bit of time wondering out loud how we're ever going to change the low priority that's currently being given to serious security problems that don't directly inconvenience end users. When they do inconvenience end users, they get fixed. When they don't, we're busy. We have other things we need to do.

Leo: That's really interesting. I hadn't thought about that.

Steve: And this sort of follows from what this Project Zero researcher found when she looked at other chat systems. And they're not having problems, but they're also not secure. So anyway, I have a sort of a little bit of a walk back in history I think our listeners may find interesting and that maybe we'll be able to draw some analogies from. And we're going to finish with a very useful analysis of the comparative security of iOS and Android, which was recently published by our friend Matthew Green from Johns Hopkins, thus the title of today's Security Now! Episode 803, "Comparative Smartphone Security."

Leo: I think a lot of people would be interested in that. I don't know if you've been following it. It's a little timely because Apple just pushed out a fix to iOS and iPadOS for three what looked like zero-days. They say they're being actively exploited today.

Steve: Wow.

Leo: So this is topical.

Steve: And I learned about them from the MacBreak Weekly podcast and went over to my phone and said, "Update yourself."

Leo: Now.

Steve: And so it's currently down, doing that.

Leo: Yeah. There are two WebKit vulnerabilities - WebKit's used throughout iOS - but also a kernel vulnerability. And apparently - Apple's pretty tight-lipped as usual about what's going on with security. They just fix it. But apparently they're keeping - and I thought I'd ask you about this. They're keeping the name of the person who submitted the bug secret. And if it was Project Zero, you know, if it was Tavis, there'd be no point in keeping it secret. I'm wondering what that signifies. That whoever submitted the bug probably got a bug bounty, but did not want their name revealed, is interesting.

Steve: Huh. Well, it's interesting too because WebKit bugs, of course, that's going to be a high exposure flaw because Safari is I'm sure WebKit-based. And so everything you do using WebKit could be a potential entry point. So anyway, good for them for getting onto it.

Leo: Well, that's the good news. But I'm very curious what Matthew Green has to say. We'll get to that in just a little bit. All right, Steve. On we go. Do you have a picture for us today?

Steve: We do. We have one. This sort of ties into what I'll be talking about later. I titled this "The Unfortunate Reality of Perverse Incentives." It's just a one-frame cartoon. It shows a hackeresque kid being taken away in handcuffs by an officer, and he says: "I broke into the website to help you." And the kind of stuffed-shirt business executive, kind of a Mr. Billingsley guy with a cigar, he says: "Thanks, but it's better if we don't know the site is insecure." Which, you know, we see examples of that; right? Like the court discovery process will acquire internal communications, emails, and learn that executives knew of a problem that they for whatever reason...

Leo: Didn't want to know.

Steve: ...chose not to deal with.

Leo: Right.

Steve: And they'd much rather not know than know and take no action.

Leo: Well, and we know, I know a lot of people, including Randal Schwartz, who have gone to jail because the company did not want that pen test information and decided, oh, we're going to just prosecute you for doing that. It happened, yeah, it happened, was it - oh, who did it happen to at The New York Times? Yeah, you've got to - yeah, it's wrong, yeah. That's all there is to say. It's wrong.

Steve: So during this past week, both Chrome and Edge have beefed up their browsers' built-in password managers. Chrome 88 was released a week ago on the 19th, and I think Edge was two days later, on Thursday, last Thursday. And, you know, I still don't understand this. But I'm using Chrome daily. It's sort of my - I always have a Firefox instance up sort of statically. But I'll just fire up Chrome to check something real quickly. Even so, it wasn't until I went to About Chrome a week later that it was induced to move itself from 87 to 88. So again, I don't get it. So for what it's worth, if our listeners are interested in playing with this new stuff, you may need to go to About Chrome and give it a little kick in the butt to get itself updated. Presumably it would happen eventually, but these updates never - I never seem to get them when they're supposed to be around.

So once that's done, one of the nice things about Chrome's password system, which it is in the process of enhancing, is its quick ease of access. You can use the menu system to search for passwords, but the term "passwords" has multiple hits in various places, so that's less than ideal. You can use the URL chrome://settings/passwords. That takes you to the right place. But the niftiest and easiest to remember way to get there is just to click on your circular logged in picture or avatar in the upper right-hand corner. And underneath the little menu that you get is a little skeleton key, which is meant to be passwords. It's the leftmost icon there, directly beneath your name and email address, which takes you directly there.

Google's security blog that announced this was titled "New Year, new password protections in Chrome." Under their topic of "Easily fix weak passwords" they said: "In Chrome 88, you can now complete a simple check to identify any weak passwords and take action easily." That's kind of funny, too, because I thought, oh, that's interesting, I'll do that. And it said, "You've got a weak password." And I thought, what? No. And when I went to look what it was, it was something that was kind of a honeypot. So it was one that I had left deliberately weak somewhere. So it's like, oh, okay, good. That's just fine.

Then also, under "Edit your passwords in one place," they explained that, thanks to Chrome 88, this update: "Chrome can already prompt you to update your saved passwords when you log into websites. However, you may want to update multiple usernames and passwords easily, in one convenient place. Starting with Chrome 88, you can manage all your passwords faster and easier in Chrome Settings on desktop and iOS." Then they said that "Chrome's Android app will be getting this feature soon, too." So not quite ready for Android.

And then they wrap up by saying: "The new features with Chrome 88 will be rolled out over the coming weeks, so take advantage of the new updates to keep your passwords secure. Stay tuned for more great password features throughout 2021." And it's unclear to me what they're saying is coming because it looks to me like it's all there right now.

Leo: Yeah. For years we said don't use the browser password vault. For a long time Chrome kept it in cleartext, which was kind of problematic. You think it's safe now to do that?

Steve: The problem I still have is that it's, to use the term we typically use relative to Apple, it's a walled garden. I mean, for me, as I mentioned, I have Firefox, and I have Chrome. And so of course I have my chosen password manager is in both. And so I get cross-make, cross-browser-make synchronization, and also the ability to run, like to have the password vault which is separate from that, that I'm able to browse. So for me, a password manager is still a worthwhile extra.

But certainly for - I can certainly see a lot of users who just want it, you know, they're just like Chrome people, or they're just Firefox people or whatever. So to help them do a better job, to make using unique complex passwords easier, I guess I would say that, even though there's still a vulnerability, for example, when you're logged into Chrome, you can go to that passwords page and say, show me my passwords. And it does. So in a shared computer environment you want to be very careful to be logging out of Chrome because if anybody is able to get to your Chrome browser, and your smiling face is up there in the upper right-hand corner, there's nothing to protect your passwords from being viewed.

So the local compromise is still present, and they don't really seem to have fixed that because it would cause trouble. But the big problem, as we know, that everybody has is that, because password management is a hassle, people are still, even today, not using long, complex, and unique passwords. So that's the exposure is out on the Internet is where the real problem seems to be, more than on the local side. So all things being equal, yeah, I would argue, compared to not using a browser's password manager, I mean, first choice, use a third-party password manager because then you're portable.

The other problem is, if all your stuff is in Chrome, you really can't use any other browser if you wanted to. That other browser doesn't have all your stuff. So there is some lock-in associated with using a single browser's system. And of course they're all doing cloud sync now, so at least you're able to be multisystem, even if you're mono browser.

Edge also did an update, that is, Microsoft's Stable Channel, also version 88. They've added a password generator. They said: "Microsoft Edge offers a built-in strong password generator that you can use when signing up for a new account or when changing an existing password." They said: "Just look for the browser-suggested password dropdown in the password field; and, when selected, it will automatically save to the browser and sync across devices for easy future use."

And it's funny, we talked last week about how Google had found out that the other Chromium users were using features that were meant to be Google Chrome browser-specific. And I was wondering, since they've been focusing a lot of their attention on Chrome 88, if that wasn't what caused them to say, hey, wait, wait a minute. There's, like, people we're not authorizing are using our API. So they may have stumbled on it for that reason.

Anyway, so both Edge and Chrome are clearly working to beef up their password managers. And again, I kind of wonder what took them so long. I've been using LastPass, what, more than a decade. Didn't we decide it was, like, a long time ago?

Leo: Yeah, yeah, long time, yeah.

Steve: Yeah. And then the browsers, which is like it's a natural place for a password manager to live, I guess for the user who's not going to install a third-party client. When you're using Chrome, as most people in the world are, and it says, you know, you're being prompted to create an account, and it pops up and says, how about using this, and don't worry, I'll save it for you, who's not going to say yes?

Leo: I always turn that off because it's annoying. No, I understand.

Steve: Oh, Leo. It's really annoying.

Leo: Apple does the same thing with Safari. It's a little different because it's - and again, it's not cross-platform. But if you live in the Apple ecosystem, it's everywhere you use it, on iOS, iPadOS, and macOS. So that's a little different. My real question - and I know Apple is doing it right. They're very secure. My real question is can I trust Firefox and Chrome's in-browser password storage? Is it secure enough, if I wanted to use it? I agree with you about the cross-platform issues.

Steve: Remember our standard lesson, which is, if the browser knows the password without asking you for anything, which is required for convenience, that means that it's there, and it's decrypted. It's available. It's very much like our original example of this principle was the DVD and DVD encryption. It's like, well, if the DVD player is able to show you a movie, and you're not just seeing random static on the screen, then it's got the keys, and it's using them. So it's sort of inherently the case that there is some risk. But notice that's also the case with LastPass. Anything which is able to fill passwords in for you without every single time challenging you about your own identity, that says that that is available.

And in fact that was one of the things that I did in SQRL, of course, was I did this kind of a tradeoff. You would have to identify yourself with your full SQRL password once per session, meaning like when you turn the computer on, or when you come back from unblanking it, or you haven't used it for some length of time, and all of those things you were able to control. After you did that, it reencrypted the secret that it had been able to decrypt only when you gave it your full password. It reencrypted it with the first "n" characters.

So the requirement was every time you authenticate, you are being prompted; but it was only, for example, by default the first four characters of your password. And you could just go bing bing bing bing on the keyboard. And since the number of characters was not a secret, there was no reason to require the user even to hit Enter. So it was just bing, bing, bing, bing.

Leo: That's as bad as you can get, yeah.

Steve: But here was the catch. If you mistyped any of those four, it then washed that secret, that reencrypted secret out of RAM immediately so that you were then required to enter your full password. So it was sort of a nice interactive tradeoff that didn't inconvenience the user, but still resulted in dramatically more security than we typically have now, where I'm not ever being harassed by LastPass or Safari, for example. I do like the LastPass integration in iOS now because now I'm able to have LastPass provide the passwords to Safari instead of storing them in yet one more place.

Leo: Right.

Steve: And I guess that would be a useful note for security is not to spread this stuff around too much because the more you do, the more opportunities there are for a mistake.

Leo: Yeah. It's got to be hard to design these things. You've got to think of every possible avenue for exfiltration and so forth. Somebody's in the chatroom saying that Edge doesn't actually know the password in the clear. They use hashes. And I imagine that would be the kind of best practice is not have a cleartext copy of the password, but just a hash, and you could compare hashes; right?

Steve: That can't be true because it has to provide the in-the-clear password...

Leo: Oh, it has to fill it in, of course.

Steve: ...to the website.

Leo: That's right, yeah, yeah. Interesting.

Steve: Yeah. What that person is confusing with is the other feature which they're now beginning to add, is the "Have any of your passwords appeared in known breaches."

Leo: Right, right.

Steve: And so that's where...

Leo: He did hashes with a [crosstalk] component, yeah.

Steve: Yes, yes, yeah. And so you're able to do that with a hash. But you have to have the password in the clear.

Leo: But you're right, they have to fill in the password, yeah, yeah.

Steve: Yup.

Leo: That's a good point. I didn't think about that.

Steve: Okay. So get a load of this. We know that Flash has been end of life. In a wonderful news report whose headline was "Adobe Flash Shutdown Halts Chinese Railroad for Over 16 Hours Before a Pirated Copy Restores Ops." The subhead: "This is what happens when you run a railroad network on Flash." This appeared in TheDrive.com, which is a website - actually I've been there before. I don't remember what occasion I had. But it describes itself as the "one-stop shop for all things automotive."

The author of the story, who clearly knows a bit of tech, he explains what happened. He said: "Supportive of everything from browser games to live streaming, Adobe Flash was the Internet's favorite multimedia platform with reason. Even in its heyday, though, Flash wasn't universally loved; it had security holes, could be tough to optimize, and wouldn't play ball with all browsers, especially those on mobile devices."

He says: "When HTML5 hit the scene, Flash began to fall out of favor; and in July 2017, Adobe announced it would cease support at the end of 2020, giving users three and half years to switch to new software." He says: "This message, however, didn't reach all corners of the IT globe. And when Flash's 'time bomb' code went off on January 12th, it did more than just make nostalgic browser games harder to revisit. It brought an entire Chinese railroad to a standstill.

"According to a report in Apple Daily, the problem reared its head for China Railway Shenyang in Dalian, Liaoning just after 8:00 a.m. on Tuesday, January 12th. Per an event timeline outlined by GitHub, the head of a switching station reported being unable to access the railroad's timetables, which they normally did through a browser-based Flash interface. Over the next half hour, reports of similar failures poured in from across the network, with as many as 30 stations implicated, according to a CR Shenyang statement reported by a Chinese blog."

Anyway, they said: "Only after technicians went online to research bug fixes did they learn of the global Flash shutdown, news of which seemingly didn't penetrate the insular Chinese Internet. A translation of the GitHub timeline suggests restoring software backups temporarily restored service around noon, though outages returned again at around 2:00 p.m." - yeah, when updates updated - "and later on in the evening.

"CR Shenyang's response team then reportedly began exploring a reversion to older software systems, its options apparently consisting of an unspecified Microsoft-based setup, or an archived, pirated version of Flash without the 'time bomb' code. Technicians settled on the latter, and around 1:00 a.m. on the 13th, CR Shenyang successfully brought one of its stations fully online. By 2:30 a.m., all but one route was back in service, and the railroad's Y2K21 nightmare was behind it."

Leo: It can't have been easy to write a Flash program to control the entire railroad. But I guess they hired somebody who knew his Flash. I mean, it was a language,

Steve: Yes. This problem was not unique. Google's well-known software engineer, Ryan Sleevi, tweeted: "South African Revenue Service decides to develop and distribute their own browser, specifically to reenable Adobe Flash."

Leo: Oh, please. Oh, my god.

Steve: The page says: "Dear Taxpayers and Traders: SARS" - which is the unfortunate abbreviation of South African Revenue Service. "SARS apologizes for the inconvenience and service disruption caused by the discontinuance of the Adobe Flash Player. We are pleased to inform you that an alternate SARS Browser solution has been implemented which affords you the ability to complete and submit the Flash-based forms not migrated to HTML5" - and of course none of the poor African taxpayers know what any of this means - "in the interim, while we complete the migration.

"The SARS Browser enables access to ALL [in caps] eFiling forms, including those that require Adobe Flash, thus maintaining compliance with your filing obligations. Please note that existing browsers such as Chrome and Edge will continue to work for all forms already migrated with the major and high-volume ones being Income Tax - PIT, Provisional Tax, CIT, and Trusts - Value Added Tax, Pay as You Earn, and Excise. You are please requested to use the SARS browser should access to the forms not yet migrated be required, which include" - and then they've got form designation Registration, Amendments, and Verification Form; Transfer Duty; Financial Certificate Information; Financial Declaration; Tax Compliance Status Request; Dividends Tax Transactions Information; and Withholding Tax on Interest.

"Please note that the SARS browser will require software to be installed on your PC and is currently compatible with Windows devices only. This SARS browser deploys as a separate application and can only be used to access the SARS eFiling website and SARS Corporate website. It cannot be used as a browser for general Internet browsing."

So basically they apparently gave priority, as they should, to the most important forms, which they are scurrying, or scurried, to move off of Flash when this caught them by surprise. And some of them are now supported under HTML5. But this balance of forms are still dependent upon Flash, which no longer works. So their solution has been to create their own browser to host some old version of Flash. There actually is a thread on GitHub about how to patch a current version of Flash to remove the time bomb, so that exists and has been done. Of course you can't use any of our current browsers because they're all no longer willing to host Flash, period. So you've got to go to wherever they got, I mean, we realize that their own browser, what is it? Is it some earlier version of something? Did they take something open source and cobble it together? Who knows. But, boy.

Leo: I doubt they wrote it from scratch.

Steve: No, you can't.

Leo: I mean, if they're stupid enough to use Flash for their tax forms, they're clearly not smart enough to write a browser.

Steve: No, no. And so you can imagine that, I suppose...

Leo: [Crosstalk] more effort to do that than it would have been to just take those forms...

Steve: Yes.

Leo: Oh, so stupid.

Steve: Yes. And I suppose that not even JavaScript was ready for primetime way back when Flash was offering a powerful client-side scripting environment. So if back then one wanted to be performing on-the-fly form content validation before submission, you know, with like dropdown lists that would then change things that the form showed, which checkboxes were enabled and so forth. You know, the things we do now with JavaScript, then maybe 20 years ago Flash would have been the best solution.

Leo: Oh, yeah. I wouldn't fault them for choosing Flash in the beginning.

Steve: Right.

Leo: Just for using it still.

Steve: For three and a half years the world has known it's ending. And I guess clearly it must have been the fact that they didn't know there was a time bomb, that it would actually stop working on January 12th, when it was like, oh, crap.

Leo: It is unbelievable.

Steve: So now that the threat of an expensive ransomware attack is quite real, investments have been made in the ability to restore from backups; right? That was the, you know, the news of ransomware has spread like wildfire through corporate America. There is a high danger that you're going to get hit with this because it seems unstoppable. So if you don't want to pay an increasingly large, it seems, demand, then tell your IT department, okay, we know that you've been telling us for the last 10 years that we need to give you more money so that you can do better backups. Okay. We're giving you that budget now because ransomware.

So obviously, today, more than just a few years ago, when a company suffers a ransomware attack, many more are as a consequence able to restore from backups and thus avoid the need to contact the attackers. The first response to this lack of need to pay the ransom was, as we know, to exfiltrate files before their encryption and then release a few as proof of possession, and threaten to release many more if a ransom was not paid.

But if the offer to decrypt encrypted files and an agreement to never leak and to destroy sensitive and perhaps personal information are both insufficient to bring a company and its C-Suite executives to heel, how about subjecting the organization to a prolonged DDoS attack? That's the latest attack strategy being added to post-ransomware attacks: blast them off the Internet until they agree to cooperate and pay. This new trend was first seen last October. It was being employed at the time by the SunCrypt and the Ragnar Locker gangs after their attacks.

But now a third player, the Avaddon ransomware gang, has also begun using DDoS attacks to take down a victim's site and network until they contact them and begin negotiating. Brett Callow, who is the threat analyst for Emsisoft, said: "It's not at all surprising to see threat actors combining ransomware and DDoS attacks. DDoS is cheap, easy, and in some cases may help convince some companies that speedy payment is the least painful option. The more pressure the criminals can put companies under, the better their chances of extracting payment."

So add that to the list of things that happens to you. Although it's kind of weird because of course a DDoS attack is entirely orthogonal to a ransomware attack. I mean, anybody could be contacted and be held at ransom for DDoS. It's like, you know, pay us some bitcoin or we're going to blast you. That doesn't seem to be happening. It's only when the company's only suffering a ransomware encryption and exfiltration of their data, and I guess they're just being pounded into submission at this point.

SolarWinds attack details continue to emerge. As we know, digital attack forensics takes time. And most of it requires very careful reverse engineering of code which has been carefully designed to thwart exactly that kind of analysis. So it's not just like random script kiddy code that's Python source, where you just look at it and go, oh, this is what it does. Especially in the case of the SolarWinds threat actors, we know, and we're going to know a lot more in a minute, about how incredibly good they were at their craft. So they designed their stuff to make it very difficult to figure out what it was doing.

So we would expect to be learning more about this largest known attack in history over time. And indeed, last Wednesday the 20th, in a joint posting by the Microsoft 365 Defender Research Team; their Threat Intelligence Center, that's the MSTIC that we often talk about; and the Microsoft Cyber Defense Operations Center, that's a new one, the CDOC, we learned a great deal more. For what it's worth, it's only more worrisome.

Microsoft's joint disclosure was titled: "Deep dive into the Solorigate second-stage activation, from Sunburst to Teardrop to Raindrop." Microsoft begins their quite lengthy disclosure with a summary of what everyone knows, but quickly adds some new detail, which is interesting. They said: "More than a month into the discovery of Solorigate, investigations continue to unearth new details that prove it is one of the most sophisticated and protracted intrusion attacks of the decade. Our continued analysis of threat data shows that the attackers behind Solorigate are skilled campaign operators who carefully planned and executed the attack, remaining elusive while maintaining persistence. These attackers appear to be knowledgeable about operations security and performing malicious activity with minimal footprint.

"In this blog, we'll share new information to help better understand how the attack transpired. Our goal is to continue empowering the Defender community by helping to increase their ability to hunt for the earliest artifacts of compromise and protect their networks from this threat." They said: "We have published our in-depth analysis of the Solorigate backdoor malware - also referred to as Sunburst by FireEye - the compromised DLL that was deployed on networks as part of SolarWinds products, that allowed attackers to gain backdoor access to affected devices. We have also detailed the hands-on-keyboard techniques that attackers employed on compromised endpoints using a powerful second-stage payload, one of several custom Cobalt Strike loaders, including the loader dubbed Teardrop by FireEye and a variant named Raindrop by Symantec.

"One missing link in the complex Solorigate attack chain is the handover from the Solorigate DLL backdoor to the Cobalt Strike Loader. Our investigations show that the attackers went out of their way to ensure that these two components are separated as much as possible to evade detection. This blog provides details about this handover based on a limited number of cases where this process has been observed to occur. To uncover these cases, we used the powerful cross-domain optics of Microsoft 365 Defender to gain visibility across the entire attack chain in one complete and consolidated view. We'll also share our deep dive into additional hands-on-keyboard techniques that the attackers used during initial reconnaissance data collection and exfiltration, which complement the broader TTPs from similar investigative blogs such as those from FireEye and Volexity."

And so I'm going to skip over a lot of that nitty-gritty because it's interesting, for anyone who's interested, and I've got the link in the show notes. But here's the cool bit that is understandable. They said: "An attack timeline that SolarWinds disclosed in a recent blog showed that a fully functional Solorigate DLL backdoor was compiled at the end of February 2020" - okay, so early in 2020, last year - "and distributed to systems sometime in late March. The same blog also said that the attackers" - and this I did not know - "removed the Solorigate backdoor code from SolarWinds' build environment in June of 2020."

They said: "Considering this timeline and the fact that the Solorigate backdoor was designed to stay dormant for at least two weeks, we approximate that the attackers spent a month or so in selecting victims and preparing unique Cobalt Strike implants as well as command-and-control infrastructure. This approximation means that real hands-on-keyboard activity most likely started as early as May. The removal of the backdoor generation function and the compromised code from SolarWinds binaries in June could indicate that by this time the attackers had reached a sufficient number of interesting targets, and their objectives shifted from deployment and activation of the backdoor, so-called 'Stage 1,' to being operational on selected victim networks, continuing the attack with hands-on-keyboard activity using the Cobalt Strike implants."

So as I said again, that was news to me. I assumed that the SolarWinds build and update delivery system had remained infected, but that's not the case. As Microsoft observed, it didn't need to keep offering infected DLLs once all of the major targets had already updated and received the infection. Essentially, they'd already gotten out over the course of, what, six months? Well, March, April, May, June. So maybe four months. And then again, in observing the highest level of care, they removed the source of the infection so that the SolarWinds DLL would then be clean, and further updates would remove it from the systems that had previously received it and had already moved from Stage 1 into Stage 2. So an act of deliberately eliminating the tracks of how this all happened.

So one of the coolest things Microsoft found was the way the original SolarWinds infection created this arm's length execution path in such a way that the original infection stood a maximum chance of remaining undetected, even if its downstream consequences were detected. Remember that the moment that it was discovered that a signed SolarWinds DLL was the root source of the infection, that would have brought down the entire operation. And as we know, that is what eventually happened at FireEye. But the obfuscation was successful for a very long time.

So here's how Microsoft explains what they found. They said: "We spent countless hours investigating Microsoft Defender telemetry and other signals from potential patient-zero machines running the backdoored version of SolarWinds DLL. Most of these machines communicated with the initial randomly generated DNS domain" - remember that it was blah blah blah dot avsvmcloud.com, they said - "but without significant activity. However, we saw limited cases in May and June where the initial DNS network communication was closely followed by network activity on port 443 (HTTPS) to other legitimate-looking domains. On these handful of machines, we performed deep inspection of telemetry.

"We know that the Solorigate backdoor only activates for certain victim profiles. And when this happens, the executing process, usually SolarWinds.BusinessLayerHost.exe, creates two files on the victim disk: a VBScript, typically named after existing services or folders to blend into legitimate activities on the machine; and a second-stage DLL implant, which is a custom Cobalt Strike loader, typically compiled uniquely per machine and written into a legitimate-looking subfolder underneath C:\Windows."

And so in other words, on a per-infection target, they created a VBScript that was uniquely named to fit into what was going on on that particular machine and custom-wrote and compiled a unique, they called it the Cobalt Strike Loader, again for that machine. So one of the things this did was it meant you could not compare infected systems. You wouldn't find any obvious indications of compromise that were the same because they were essentially doing per-target customization.

Microsoft said: "At this point the attackers are ready to activate the Cobalt Strike implant. However, the attackers apparently deem the powerful SolarWinds backdoor too valuable to lose in case of discovery, so they tried to separate the Cobalt Strike Loader's execution from the SolarWinds process as much as possible. Their hope is that, even if they lose the Cobalt Strike implant due to discovery and detection, the compromised SolarWinds binary and the supply chain attack that preceded it will not be exposed.

"The attackers achieved this by having the SolarWinds process create an Image File Execution Options (IFEO) Debugger registry value for the process dllhost.exe." And I'll just insert an aside here. This is a known and official way of causing Windows to attach a debugger to a process at startup. If you want to put a given Windows process under debugging, sometimes it's not enough to attach the debugger after the process is already initialized. Like there may be initialization code that is where the problem is. So you need Windows to start the debugging like from the moment the process goes into RAM. The way you do that is by using one of this Image File Execution Options Debugger registry values which causes Windows to automatically load something into that process, a space.

This of course can also be used for malicious purposes. So the SolarWinds process first created one of these entries for the dllhost.exe. That execution process triggers a launch of wscript.exe which is configured to run that VBScript file which has been dropped earlier and had been waiting. The VBScript in turn runs the rundll.exe, which activates the Cobalt Strike DLL using a clean parent/child process tree which is completely disconnected from the SolarWinds process.

So essentially, by using this sanctioned Windows hook in the registry to cause the DLL host to invoke wscript.exe, there was a complete separation of these two events. Meaning that anybody who did discover the Cobalt Strike DLL and the way it got executed would only see that it was tied to this debugging. Oh, and once it runs, it removes the IFEO value from the registry to also clean that up so you can't even figure out how this thing got started, if you look at it after it's running.

So anyway, their full posting, as I mentioned, is super long and wonderfully detailed, for anyone wanting to really get down into the nitty-gritty. But in the section "Additional attacker tactics, anti-forensic behavior, and operational security," Microsoft nicely summarizes some more detail to give us a more complete sense for the frightening skill and tradecraft that the designers of this attack deployed.

They said: "As mentioned, the attackers behind Solorigate are skillful and methodic operators who follow operations security best practices to minimize traces, stay under the radar, and avoid detection. During our in-depth analysis of the attacker's tactics, techniques, and procedures (TTPs) seen through the lens of Microsoft 365 Defender's telemetry, we observed a few techniques that are worth disclosing to help other defenders better respond to this incident and use hunting tools like Microsoft 365 Defender advanced hunting or Azure Sentinel queries to search for potential traces of past activity.

"Some examples of why these attackers stand out for their operational OPSEC methodology and anti-forensic behavior are listed below." And they have six, or five. First: "Methodic avoidance of shared indicators for each compromised host. As discussed in the previous section, each Cobalt Strike DLL implant was prepared to be unique per machine and avoided at any cost overlap and reuse of folder name, file name, export function names - those are internal to the DLL - command-and-control domain and IP, HTTP requests, timestamp, file metadata, config, and child processes launched."

They said: "This extreme level of variance was also applied to non-executable entities, such as WMI persistence filter name, WMI filter query, passwords used for 7-zip archives, and names of output log files." I mean, so what we're looking at is an absolute lack of laziness. I mean, true discipline for every single entity that they infected. They said: "Applying this level of permutations for each individual compromised machine is an incredible effort normally not seen with other adversaries and done to prevent full identification of all compromised assets inside a network or effective sharing of threat intel between victims."

Second: "Camouflage and blending into the environment. Tools and binaries used by the attackers, for example ADFIND" - Active Directory Find - "legit tool were always renamed and placed in folders that mimicked existing programs and files already present on a machine. This blending was not just used for files, but for other elements. For example, WMI persistence filters were created with names and queries matching other scripts present in affected organizations." This is just stunning.

Third: "Before running intensive and continued hands-on-keyboard activity, the attackers took care of disabling event logging using AUDITPOL (audit policy) and re-enabling it afterward. In a similar way, before running noisy network enumeration activities such as repeated NSLOOKUP and LDAP queries, the attackers carefully prepared special firewall rules to minimize outgoing packets for certain protocols. The firewall rules were also methodically removed after the network reconnaissance was completed." I hope this is terrifying everybody. This is just - it's terrifying me.

"Lateral movement activities were never executed without preparation. To increase the likelihood that their activities remain undetected, the attackers first enumerated remote processes and services running on the target host and decided to move laterally only after disabling certain security services." And finally they said: "We believe that the attackers used timestomping to change timestamps of artifacts and also leveraged professional wiping procedures and tools to complicate finding and recovering of DLL implants from affected environments."

Leo: I like "timestomping." I'm going to keep that in my back pocket there.

Steve: Timestomping, yes. Stepping back to take stock in all that we have learned, any sane Infosec technologist would be right to be seriously worried. My feeling is that as damaging as these attacks were individually and on their own, its almost more worrisome outcome for the attackers is for us to have obtained this much greater appreciation for their skill and their dedication to detail. I mean, it has without question sobered up and heightened the level of attention that the Defender industry now realizes it needs to deploy.

And remember, none of us should forget for a moment that were it not for the fact that they targeted FireEye, and that their presence eventually tripped some monitoring alarms that the attackers were unaware of - because as we've just seen, if they knew about it, they would have either aborted or they would have disabled those monitoring alarms. Something tripped them up. If that had not happened, this would all still be ongoing right now.

Leo: Wow.

Steve: And Leo, on that happy note, let's take our second break.

Leo: And it's funny because there's not a lot of reporting anymore on this. It's kind of taken the back seat because I think...

Steve: Old news.

Leo: Well, but also I think part of the problem is it doesn't feel like there's much we can do about it. It's like, yeah, they're in there. What do you want to do about it? You know? It's like the deed is done.

Steve: Yeah. The good news is, I think, that this level, I mean, so for Microsoft to post this...

Leo: I'm glad they're paying attention, yeah.

Steve: Yes. Any companies, for example, who thought, oh, you know, we're busy. I mean, yeah, who isn't? We don't have enough money. Yeah, who does? We don't want to, like, deeply invest in internal network monitoring surveillance. Well, think about that again, folks.

Leo: Yeah.

Steve: You know, reconsider the cost of not doing that. You need an intrusion detection system that can spot something that is doing this, and you need to hide it. That's the other thing we've learned. The fact that it's possible for the bad guys to see machines and then remotely enumerate their running processes, you know, a lot has been learned. I think that's key. I agree, Leo, that on the general public level, it's like, oh, well. Maybe the Russians hacked us. What's for lunch?

Leo: Yeah. I hope we're doing the same back to them.

Steve: On our internal level, you know, this has to have really sobered up the defense industry.

Leo: And it should, yeah.

Steve: Yes.

Leo: And I have to say, and probably somewhat due to this, I notice Biden has been starting to appoint people to cybersecurity roles, and really beefing up cybersecurity, and picking some, I think, some good people, knowledgeable people, not just figureheads, to do it. So I think that's the other side of it is that you're going to see, I hope you're going to see the U.S. government be very proactive about this. They've got to be.

Steve: Yeah, appointing people who are anti-cybersecurity to run cybersecurity, that's...

Leo: Yeah, that's not a good idea. It's less of a good idea.

Steve: Nah, that's not a good idea.

Leo: Yeah. On we go with Steve Gibson and more security news.

Steve: So it turns out that Malwarebytes was also attacked.

Leo: Oh, I saw that, yeah.

Steve: Yeah. Last Tuesday Malwarebytes posted a notice with the headline: "Malwarebytes targeted by nation-state actor implicated in SolarWinds breach."

Leo: Yikes.

Steve: And they said: "Evidence suggests abuse of privileged access to Microsoft Office 365 and Azure environments." And I wanted to share their short summary posting because it's impressive. It contains additional important details and also helps to highlight the nature and need for a true security community. This is what we're going to have to be doing moving forward. So they wrote: "A nation-state attack leveraging software from SolarWinds has caused a ripple effect throughout the security industry, impacting multiple organizations. We first reported on the event in our December 14th blog and notified our business customers using SolarWinds, asking them to take precautionary measures.

"While Malwarebytes does not use SolarWinds, we, like many other companies, were recently targeted by the same threat actor. We can confirm the existence of another intrusion vector that works by abusing applications with privileged access to Microsoft Office 365 and Azure environments. After an extensive investigation, we determined the attacker only gained access to a limited subset of internal company emails. We found no evidence of unauthorized access or compromise in any of our internal on-premises and production environments. We received information from the Microsoft Security Response Center on December 15th about suspicious activity from a third-party application in our Microsoft Office 365 tenant, consistent with the tactics, techniques, and procedures (TTPs) of the same advanced threat actor involved in the SolarWinds attacks.

"We immediately activated our incident response group and engaged Microsoft's Detection and Response Team (DART). Together, we performed an extensive investigation of both our cloud and on-premises environments for any activity related to the API calls that triggered the initial alert. Our investigation indicates the attackers leveraged a dormant email protection product within our Office 365 tenant that allowed access to a limited subset of internal company emails. We do not use Azure cloud services in our production environments." And I'll just pause to take a note. Here's another instance of why it's good to turn things off and remove things you're no longer using. So they had a dormant email protection product that they were no longer using, but it was still there.

Leo: Still running, yeah.

Steve: And it turns out that was the way in. Exactly. They said: "Considering the supply chain nature of the SolarWinds attack, and in an abundance of caution, we immediately performed a thorough investigation of all Malwarebytes source code, build, and delivery processes, including reverse engineering our own software. Our internal systems showed no evidence of unauthorized access or compromise in any on-premises and production environments. Our software remains safe to use.

"As the U.S. Cybersecurity and Infrastructure Security Agency (CISA) stated, the adversary did not only rely on the SolarWinds supply-chain attack, but indeed used additional means to compromise high-value targets by exploiting administrative or service credentials. In 2019, a security researcher exposed a flaw with Azure Active Directory where one could escalate privileges by assigning credentials to applications. In September 2019, he found that the vulnerability still existed and essentially led to backdoor access to principals' credentials into Microsoft Graph and Azure AD Graph.

"Third-party applications can be abused if an attacker with sufficient administrative privilege gains access to a tenant. A newly released CISA report reveals how threat actors may have obtained initial access by password guessing or password spraying in addition to exploiting administrative or service credentials. In our particular instance, the threat actor added a self-signed certificate with credentials to the service principal account. From there, they can authenticate using the key and make API calls to request emails via MSGraph.

"For many organizations, securing Azure tenants may be a challenging task, especially when dealing with third-party applications or resellers. CrowdStrike has released a tool to help companies identify and mitigate risks in Azure Active Directory." And then they conclude in their topic "Coming Together as an Industry." They said: "While we have learned a lot of information in a relatively short period of time, there is much more yet to be discovered about this long and active campaign that has impacted so many high-profile targets. It is imperative that security companies continue to share information that can help the greater industry in times like these, particularly with such new and complex attacks often associated with nation-state actors.

"We would like to thank the security community, particularly FireEye, CrowdStrike, and Microsoft, for sharing so many details regarding this attack. In an already difficult year, security practitioners and incident responders responded to the call of duty and worked throughout the holiday season, including our own dedicated employees. The security industry is full of exceptional people who are tirelessly defending others, and today it is strikingly evident just how essential our work is moving forward."

So I agree. I think that it's clearly the free and open sharing of the details of what is found in these attacks, you know, not considering it proprietary, not holding it close, but saying, look, this is what we found. In this instance, for example, Microsoft and FireEye being so open allowed Malwarebytes to get a much better sense for the nature of the infiltration into their own network. So as if we didn't already have sufficient cause to be worried about the SolarWinds threat actor, and that previous Microsoft blog should just chill anybody, now we learn that this incredibly potent adversary is also not a one-trick pony. Not only were they also mounting this other entirely different attack, they were doing so at the same time as they were carefully and delicately rooting around within the world's highest end corporate and government networks. So, wow.

And this next piece put me to thinking, as you'll see, because I'm going to do a little thinking out loud after I explain what was found. And I titled this "It seems that wherever we look we find problems." We'll all recall that critical flaw that was found in Apple's implementation of FaceTime early in 2019. Remember, the flaw made it possible for users to initiate a FaceTime video call and eavesdrop on their callees by adding their own number as a third person in a group chat, even before the person on the other end accepted the incoming call. Apple responded by immediately taking FaceTime's group chat feature offline until the oversight was fixed, as it was in a subsequent update.

But the interesting nature of this flaw captured the attention of a Google Project Zero researcher named Natalie Silvanovich. She describes what made this flaw so different and interesting to her, as follows. She wrote: "On January 29th, 2019, a serious vulnerability was discovered in Group FaceTime which allowed an attacker to call a target and force the call to connect without user interaction from the target, allowing the attacker to listen to the target's surroundings without their knowledge or consent."

She wrote: "The bug was remarkable in both its impact and mechanism. The ability to force a target device to transmit audio to an attacker device without gaining code execution was an unusual and possibly unprecedented impact of a vulnerability. Moreover, the vulnerability was a logic bug in the FaceTime calling state machine that could be exercised using only the user interface of the device.

"While this bug was soon fixed, the fact that such a serious and easy to reach vulnerability had occurred due to a logic bug in a calling state machine, an attack scenario I had never seen considered on any platform," she wrote, "made me wonder whether other state machines had similar vulnerabilities, as well. This post describes my investigation into calling state machines for a number of messaging platforms including Signal, JioChat, Mocha, Google Duo, and Facebook Messenger."

And I have a link in the show notes for anyone who's interested. Again, super detailed. I'm going to skip all that, but cut to the chase here. So in her very detailed posting, Natalie proceeds to explain how WebRTC sessions are established between a caller and a callee, and how unless particular care is taken in the interactive handshaking between the endpoints, which is managed by a state machine, a number of exploits are possible, arising from subtle mistakes made in the implementation of the governing protocol's state machines.

So again, for anyone who's interested in the details, the link is here. So I'm only going to share what Natalie's subsequent research uncovered. As with Apple's prepatched FaceTime, many other apps, turns out, allowed calls to be connected without interaction from the callee, while also potentially permitting the caller to force a callee device to transmit audio and video data. And as I noted, the common root cause were logic bugs within the signaling state machines, which Natalie described as a "concerning and under-investigated attack surface of video conferencing applications."

So four things she found. For Signal, this was fixed in September 2019, thanks to her analysis. An audio call flaw in Signal's Android app made it possible for the caller to hear the callee's surroundings due to the fact that the app didn't check if the device receiving the connect message from the callee was the caller device. Whoops.

Number two, both JioChat (this was fixed last summer, July of 2020), and Mocha (fixed in August of 2020). Adding candidates to the offers created by Reliance JioChat and Viettel's Mocha Android apps that allowed a caller to force the target device to send audio and video without a user's consent. The flaws stemmed from the fact that the peer-to-peer connection had been set up even before the callee answered the call, thus increasing the "remote attack surface of WebRTC."

Third, actually fourth if you consider that we had two last time, Facebook Messenger, which was recently just fixed in November of 2020. A vulnerability that could have granted an attacker who's logged into the app to simultaneously initiate a call and send a specially crafted message to a target who is signed in to both the app as well as another Messenger client such as the web browser, and begin receiving audio from the callee device.

And, finally, Google Duo, fixed last December. A race condition between disabling the video and setting up the connection that in some conditions could cause the callee to leak video packets from unanswered calls. So I titled this piece "It seems that whenever we look we find problems" because that seems to be today's reality. Natalie wasn't driven to examine these communication applications because of any known flaws. She was just wondering whether anything might be wrong when she saw a problem in FaceTime that she realized might be endemic. And she discovered, after looking, significant logical flaws behind many other apps' use which could be leveraged for surreptitious surveillance.

The complexity of today's apps is exploding. And the bar for what an app must do to competitively succeed keeps being raised. Those raising the bar are throwing together huge functional blocks of code without a deep understanding which is required to consider or even be aware of the security and edge-case use implications created by these massive conglomerations. Need real-time audio and video chat? No problem. Just drop in and hook up a WebRTC library. Wave at the camera. Can you see yourself on the other end? Yes? Great. We're done. Ship it.

And of course when one app does this, those competing with that app are forced to follow suit because reviewers never see the logic flaws that interested Natalie. Reviews create app feature grids filled in with red and green squares showing which apps offer which features. So anyone whose app has a glaring red box highlighting a missing feature is in a big hurry to go grab some barely understood third-party library, drop it in, hook it up, wave at the camera, and push out a new feature release. And I'm not sure, and this is the point here, how or why this is ever going to change. Some things do change. I still have, and Leo you probably do, too, the CTRL-S habit.

Leo: Oh, yeah.

Steve: As I call it.

Leo: All the time. Constantly.

Steve: Yup, which was developed from the need to constantly save our work.

Leo: Because it crashed. Everything crashed.

Steve: Because back - exactly. Back at the dawn of all this, you never knew when the app or the operating system you were using was going to freeze up, which was something that generally happened multiple times per day, without any warning. Now, that doesn't generally happen anymore, not like it once did. But the problem is, everything is driven by economics. When apps and OSes crashed, and work was lost, users were unhappy. This created an economic cost because there was an incentive to seriously consider trying a different word processor or whatever app, or even change OSes. Consequently, apps and OSes no longer crash like they once did.

But not one of the users of these audio and video conferencing apps were ever inconvenienced by the logical flaws in their internal state machines which enabled eavesdropping on them. They might be annoyed if the app didn't offer the turn-my-face-into-an-elf option. So of course every app needs to offer that, or suffer the red grid cell. So I think we're in trouble because users and reviewers do not and cannot perceive security flaws. They only see features. IoT devices pay no economic toll for using a TCP/IP stack that's riddled with critical flaws. Nor for not providing any aftermarket means for updating them. Doing so would be expensive. But no rational product designer in China is going to invest in anything that doesn't provide an economic advantage.

So I see three possibilities. Back in the early days of electrical household appliances, which were playing with dangerous voltages and currents, someone needed to establish a standard for things like the thickness of insulation and the presence of a grounded plug to minimize shock hazard in the event of an internal failure. As with today's computer security, consumers were unable to evaluate the safety or lack thereof that had been designed into their toasters and their vacuum cleaners. Consequently, there was no economic benefit associated with adding manufacturing cost to make appliances more safe. Insulation costs money. A grounded three-wire cord is more expensive than one with just two wires.

Things were a mess until Underwriters Laboratories with their UL Seal of Approval became a household term. Now consumers could look for a UL Seal to assure them that some white lab-coated scientist had dropped the toaster they were considering purchasing into a tub of water to see what would happen. And if the toaster did the right thing, it would get the seal. And the similar-looking but less expensive toasters next to it on the same shelf, but lacking the UL Seal of Approval, would go unsold. Since there was no way Mom was going to expose little Johnnie and Susie to any dangerous appliances, the UL Seal was soon effectively required, and all toasters had to invest in the additional manufacturing cost required to earn themselves the seal.

And of course later the famous Consumer Reports served a similar role. Consumers who took one look under the hood of a car they were considering purchasing quickly realized that they were unable to evaluate what they saw. Nor were they able to perform their own crash testing. So they needed to rely upon a neutral, highly reputable, independent testing organization to help them choose. So perhaps something like Underwriters Laboratories or Consumer Reports will emerge.

But the other inevitability I think is the heavy hand of government regulation. Seatbelts do save lives. Many people object to being told to buckle up. And no auto maker wanted to put them in. There was no obvious economic benefit for doing so, and even a liability if the belt appeared to malfunction when it was needed. And automotive pollution standards fall under the same category. They're good for society collectively, though expensive for automakers and consumers individually. If given a choice, pollution standards would never have occurred voluntarily. The government needed to force it. Anyone who remembers what the air in Los Angeles once looked like and yes, you were actually able to see the air in L.A. - or what Beijing looked like more recently can appreciate that sometimes government regulation against short-term economic interest is required.

SolarWinds has been a massive wakeup call to many of the world's government legislators. But the attacks inherently targeted and affected large public and private organizations. More than anything, it demonstrates the level of malign intent and energy that is focused against the U.S. and other international organizations. I would not be surprised if we eventually discover that something similar has been going on with some subset of the hundreds of millions of U.S. consumer networks that are increasingly hosting IoT devices with unknown security flaws. It may be the wakeup call we need to get serious about the security of our devices. Because I don't know what else will do it.

I've got those $5 Govee plugs, and they work great. But there's no way I'm fooling myself that there's anything secure about them. It's getting a turn-on/turn-off signal from China twice a day. And as a consequence, it is well isolated on its own network. But I and some subset of the listeners of this podcast are probably the only people doing that today. Maybe we're being overcautious. But, boy, I would not want my internal home network exposed to IoT things that are reaching back to, I'm sure, well-meaning Chinese companies and their servers because there's no telling what they're able to do.

So anyway, I don't know how we fix this problem that security is not something you can perceive. It's a feature that goes unappreciated by consumers. And it is arguably expensive to create. Look at all of the work that Apple and Google go into, and we'll be talking about that in a moment, to enhance the security of their platforms. Saying that they have it is a massive selling point. And all of the IoT devices say, oh, this or that security. But then we learn that the networking stacks they're all based on are buggy as hell. So it's a mess, Leo.

Leo: Yeah.

Steve: I just had a note here in the show notes to comment that I am now in the last season of "The Expanse," and OMG.

Leo: It's keeping up the quality, huh?

Steve: Oh, my lord. No, actually I would say it's better.

Leo: Better.

Steve: IMDB has increased the rating in the last seasons to - maybe it's 10. It's 9.8 and 9.9, if not 10. And I see it. Amazon took it over for seasons 4 and 5. There's going to be a 6. And they had a six-season arc ready for when Amazon took it. So that would be up through nine seasons total. And the way it's set up - and I can't say anything without being a spoiler. But the way the plot evolves, it could continue going, much as the first Star Trek was on a five-year mission just to go see what was out there, so plenty of time. Unfortunately, we only got three years of that mission. But it is open-ended.

And Leo, I just, as I'm sitting there watching it, I'm thinking, my god, this is science fiction. This is science fiction. We don't get science fiction. I mean, like, "Avatar" was the last thing, or I guess Johnny Depp has done a couple things, and Tom Cruise apparently loves doing science fiction movies. So we've had some from him. But boy, it is just so good.

Leo: Chris Nolan's doing some good science fiction. I think "Interstellar" was pretty good.

Steve: Yes. Long, but really good.

Leo: "Tenet"? I just watched "Tenet," and it was interesting. It's more like...

Steve: Oh, Leo. If your brain hasn't melted down after watching "Tenet," "Tenet" is like, oh, my god, I mean, whoa, that's literally...

Leo: I enjoyed it. It's kind of James Bond meets time travel sort of.

Steve: Oh. James Bond meets time travel, and Dr. Seuss does time travel. Oh. Oh, whoa.

Leo: It's a time pincer movement. Time pincer movement.

Steve: Oh, it is a...

Leo: Did you not like it? I think you walked out of it, didn't you?

Steve: It is a mind "eff," is what it is. If you try to actually understand it, you will hurt yourself.

Leo: Yes, I did. Yeah, don't try to make any sense of it. But it's fun.

Steve: Really, do not try to actually understand it. Just sort of let it wash over you and go, okay, well, I have no idea what's happening.

Leo: No idea.

Steve: But, you know, some people are walking forward, and some people are walking backward.

Leo: Yeah. I like that, yeah.

Steve: And, okay. Oh.

Leo: Oh, lordy. I've started "The Expanse," and I'm looking forward to it. I want to find some time because it's so much. There are so many episodes now. I want to find a chunk of time that I can watch it.

Steve: Well, and the warning, yes, the warning is it is complicated. Lorrie imagines herself able to play with her iPad while watching a show at the same time. Consequently, she said last night, "Honey, I have no idea what's going on."

Leo: Oh, I do the same thing, yeah.

Steve: "I have no idea what's happening." And it's like, "Okay, well, I could tell you." But she doesn't have much patience for that, either. So it's okay.

Leo: Yeah, I know those feelings.

Steve: Yeah, but, I mean, the good news is, after the beginning few seasons, you kind of get - you learn the politics of the solar system. And then it's still there, but you're not on a learning curve anymore. You're kind of like, okay, good. And you know who the various people are. You still can't understand what the Belters are saying half the time, but that's okay because it's not important. Anyway, I'll just say wow. Amazon Prime, "The Expanse." Anybody who is dying for some good science fiction, it is really well done. And Leo, it's so easy to take the visuals for granted. It's just beautiful work.

Leo: Mm-hmm. It is, yeah. All right. We would normally take a break here, but we won't because everybody's dying to hear about comparative smartphone security.

Steve: So, yes. A team of security researchers at Johns Hopkins led by cryptographer, security technologist, and associate professor of computer science, our friend Matthew Green, decided to take a serious look at the comparative security offered by the Apple iOS and the Google Android smartphone platforms. To host the results of their analysis, they grabbed the domain SecurePhones.io. So, easy to remember: SecurePhones.io. And I've got three links in the show notes to that home page, the main page, and the PDF, for anyone who wants them. So here's how they framed the intent, goals, and a very brief summary of their research because the PDF is 120 pages long, 119 pages long.

They said: "In this work we present definitive evidence, analysis and, where needed, speculation to answer three questions: Which concrete security measures in mobile devices meaningfully prevent unauthorized access to user data? In what ways are modern mobile devices accessed by unauthorized parties? And, third, how can we improve modern mobile devices to prevent unauthorized access?"

And they said: "We divide our attention between two major platforms in mobile space, iOS and Android. And for each we provide a thorough investigation of existing and historical security features, evidence-based discussions of known security bypass techniques, and concrete recommendations for remediation.

"In iOS we find a powerful and compelling set of security and privacy controls, backed and empowered by strong encryption, and yet a critical lack in coverage due to underutilization of these tools leading to serious privacy and security concerns. In Android we find strong protections emerging in the very latest flagship devices, but simultaneously fragmented and inconsistent security and privacy controls, not least due to disconnects between Google and Android phone manufacturers, the deeply lagging rate of Android updates reaching devices, and various software architectural considerations. We also find in both platforms exacerbating factors due to increased synchronization of data with cloud services."

They said: "The markets for exploits and forensic software tools which target these platforms are alive and well. We aggregate and analyze public records, documentation, articles, and blog postings to categorize and discuss unauthorized bypass of security features by hackers and law enforcement alike. Motivated by an increasing number of cases since Apple versus the FBI in 2016, we analyze the concrete impact of forensic tools and the privacy risks involved in unchecked seizure and search. Then we provide in-depth analysis of the data potentially accessed via common law enforcement methodologies" - meaning subpoenas - "from both mobile devices and accompanying cloud services.

"Our fact-gathering and analysis allow us to make a number of recommendations for improving data security on these devices. In both iOS and Android we propose concrete improvements which mitigate or entirely address many concerns we raise, and provide ideation towards resolving the remainder. The mitigations we propose can be largely summarized as the increased coverage of sensitive data via strong encryption, but we detail various challenges and approaches towards this goal and others."

And they conclude: "It is our hope that this work stimulates mobile device development and research toward increased security and privacy, promotes understanding as a unique reference for information, and acts as an evidence-based argument for the importance of reliable encryption to privacy, which we believe is both a human right and integral to a functioning democracy."

So as I said, the detailed analysis is 119 pages. And for anyone who's interested in a much greater level of detail, it is a goldmine reference. But for the podcast, here's a summary of the team's findings and conclusions for each platform. They said: "In Apple iOS we found a powerful and compelling set of security and privacy controls, backed and empowered by strong encryption. However, we also found a critical lack in coverage due to underutilization of these tools."

Now, specifically, four things: "Limited benefit of encryption for powered-on devices. We observe that a surprising amount of sensitive data maintained by built-in applications is protected using a weak 'available after first unlock' (AFU) protection class, which does not evict decryption keys from memory when the phone is locked. The impact is that the vast majority of sensitive user data from Apple's built-in applications can be accessed from a phone that is captured and logically exploited while it is in a powered-on, but still locked state." So hello, Apple.

Number two: "Weaknesses of cloud backup and services. Use of Apple iCloud, unsurprisingly, transmits an abundance of user data to Apple's servers, in a form that can be accessed remotely by criminals who gain unauthorized access to a user's cloud account, as well as authorized law enforcement agencies with subpoena power." And of course this has been long known. But they're highlighting it. "More surprisingly, we identify several counter-intuitive features of iCloud that increase the vulnerability of the system."

Third: "Evidence of past hardware" - SEP, that's the Secure Enclave Processor - "compromise." They said: "iOS devices place strict limits on passcode guessing attacks through the assistance of a dedicated processor known as the Secure Enclave Processor (SEP). We examined the public investigative record to review evidence that strongly indicates that as of 2018, passcode guessing attacks were feasible on SEP-enabled iPhones using a tool called GrayKey. To our knowledge, this most likely indicates that a software bypass of SEP was available in the wild during this timeframe."

And finally: "Limitations of end-to-end encrypted cloud services. Several Apple iCloud services advertise end-to-end encryption in which only the user with knowledge of a password or passcode can access cloud-stored data. We find that the end-to-end confidentiality of some encrypted services is undermined when used in tandem with iCloud backup service. More critically, we observe that Apple's documentation and user settings blur the distinction between 'encrypted' such that Apple has access, and 'end-to-end encrypted' in a manner that makes it difficult to understand which data is available to Apple. Finally, we observe a fundamental weakness in the system: Apple can easily cause user data to be reprovisioned to a new, and possibly compromised, HSM" - Hardware Security Module, meaning an iOS device - "simply by presenting a single dialog on a user's phone. We discuss techniques for mitigating this vulnerability."

So that's the Apple side. They said: "In Android" - repeating what they said in their summary - "we found strong protections emerging in the very latest flagship devices, but simultaneously fragmented and inconsistent security and privacy controls, not least due to disconnects between Google and Android phone manufacturers, the deeply lagging rate of Android updates reaching devices, and various software architectural considerations."

And now we get specific. First: "Limited benefit of encryption for powered-on devices. Like Apple iOS, Google Android provides encryption for files and data stored on disk. However, Android's encryption mechanisms provide fewer gradations of protection. In particular, Android provides no equivalent of Apple's Complete Protection encryption class, which evicts decryption keys from memory shortly after the phone is locked. As a consequence, Android decryption keys remain in memory at all times after first unlock, and user data is potentially vulnerable to forensic capture."

Second: "Deprioritization of end-to-end encrypted backup. Android incorporates an end-to-end encrypted backup service based on physical hardware devices stored on Google's datacenters. Unfortunately, the end-to-end encrypted backup service must be opted-into by app developers and is paralleled by the opt-out Android Auto-Backup, which provides encryption keys to Google servers."

Third: "Large attack surface. Android is the composition of systems developed by various organizations and companies. Because the development of these components is not centralized, cohesive integrating security for all of Android would require significant coordination, and in many cases such efforts are lacking or nonexistent." That sort of ties in with my notion of the conglomeration of libraries that weren't authored by the people who are pulling them all together. You just don't know what the interactions are going to be.

Fourth: "Limited use of end-to-end encryption. End-to-end encryption for messages in Android is only provided by default in third-party messaging applications. Many native Android applications do not provide end-to-end encryption, the exceptions being Google Duo and, more recently, the Android Messages application."

And, finally: "The availability of data in services." They said: "Android has deep integration with Google services such as Drive, Gmail, and Photos. Android phones that utilize these services, the large majority of them, send data to Google, which stores the data under keys it controls - effectively an extension of the lack of end-to-end encryption beyond just messaging services. These services accumulate rich sets of information on users that can be exfiltrated either by knowledgeable criminals via system compromise, or by law enforcement via subpoena power."

They conclude: "We also found, in both iOS and Android, exacerbating factors due to increased synchronization with data and cloud services." So anyway, that pretty much wraps this up. They've taken a deep dive, looked comprehensively, obviously, at both platforms. For anyone who wants a lot more detail, it is available in this 120-page PDF. And nothing was hugely surprising. It's mostly a confirmation of what we know and/or have assumed from previous experience with both platforms.

However, I would argue that the prevalence, the persistence of the decryption keys in a locked device is an issue. Clearly Apple is doing it so that locked devices can show you things on the locked device screen. It would be a user inconvenience to need to unlock your device for the thing to be alive at all. So this is clearly a tradeoff that they've tried to make. But it is the case that in order to provide those features on a locked device, you need to leave the keys. If the data is decrypted after a reset and a power-on, then that is to say until the first unlock, then you have to keep those keys around, if the thing's going to do anything without needing the user to reauthenticate themselves.

And in the case of Android, we now know that everything is available in RAM for forensic analysis. So from a consumer standpoint, the takeaway is turn the device off in order to get true protection. Don't rely on the fact that the device is locked to protect you. And I would argue that it's useful to have pulled all of this information together, very useful, into a single coherent and comparative report. And while there's not very much individuals can do to address these shortcomings, other than, as I said, keep in mind turning your device off if you're in an area where you're concerned that you might lose control of it in any way. It does give Apple and Google some issues to ponder, as hopefully this podcast does every week.

Leo: It's interesting, though, there's no real, like, solid, "This is better than that." "That's the one to use." Anything like that.

Steve: Yeah. I don't think they wanted to put themselves in that position.

Leo: No, no, yeah. It's more complicated than that, too.

Steve: It is. Exactly. It is. And the largest problem, they make it clear, is the iCloud, the whole involvement of a cloud. When you're going to do that, when you're going to have backup, I mean - and many people have noted, hey, I bought another phone, and I logged in, and suddenly all my stuff is here. Well, you know, what does that mean?

Leo: It was stored somewhere.

Steve: Uh-huh.

Leo: It was somewhere. Fascinating stuff. Always very informative, very useful. Thank you, Steve. I really, really appreciate this.

We do Security Now! of a Tuesday afternoon about 1:30 Pacific, 4:30 Eastern, 21:30 UTC. If you want to get the freshest copy, that would be watching it live. There's live audio and video at TWiT.tv/live. Chat with us live if you're listening live at irc.twit.tv. You can get it on demand, too, though. Steve has 16Kb versions. That's the smallest audio version. Sounds like Thomas Edison in the early days of the wax cylinder, but it's small. There's the advantage of that. He also has 64Kb audio.

Maybe the smallest version is the text transcription created by an actual human being, Elaine Farris, who does such a good job. I think people like to read along as they listen. I know that, Steve. But you can also use it to search for any point in any podcast. That way you can go right to the show you want. Now that there's 803 of them, it probably is very useful.

Steve also puts his show notes up there, which include that 20-page of notes that he just talked about, so that's also very useful. And that has links plus the Picture of the Day. So in fact that's like a nice little magazine. You should make that a newsletter or something. That's valuable stuff. I'll leave that to you.

We have 64Kb audio and video versions - holy cow, talk about a giant file - at our website, TWiT.tv/sn for Security Now!. There's a YouTube channel dedicated to Security Now!. And let's see. Oh, yeah, you can also subscribe. It's a podcast. That means you can get it automatically the minute it's available just by subscribing, and in future you'll get every episode. Find your favorite podcast client.

If you do listen on your own time, we have an asynchronous - we have several asynchronous forums. Steve has his own forums at GRC.com. They're excellent and very active. We have forums at www.twit.community. We have a Mastodon instance. We're part of the Fediverse at twit.social. I should encourage you, if you go to Steve's site, GRC.com, you should definitely pick up a copy of SpinRite, if you don't already have one. Currently version 6, but 6.1 is in process. If you get 6 now...

Steve: In process.

Leo: ...you will get 6.1 when it comes out; and you get to participate, too, in the development of 6.1. It's getting closer and closer. While you're there, check out the rest of the great stuff. There's a whole ton of free stuff at GRC.com. You can leave him feedback at the website, GRC.com/feedback, or on his Twitter feed. His DMs are open, as the kids say, @SGgrc. I never heard a kid say that ever. But they are: @SGgrc.

Steve: They now say things like "dope." Like what the hell?

Leo: Dope. I don't know, that's - they don't even say "dope" anymore. I don't even...

Steve: And "hundred percent." Hundred percent.

Leo: Hundred percent, that's a big one, yeah.

Steve: Hundred percent.

Leo: Mr. Gibson, thank you so much. Have a wonderful week.

Steve: Buddy, always a pleasure.

Leo: All right. We'll see you next time on Security Now!.

Steve: Right-oh. Bye.