SECURITY NOW!

Transcript of Episode #611

Go FCC Yourself

Description: This week Steve and Leo discuss much more about the Intel AMT nightmare, Tavis and Natalie discover a serious problem in Microsoft's built-in malware scanning technology, Patch Tuesday, Google's Android patches, SMS two-factor authentication breached, Google goes phishing, the emergence of ultrasonic device tracking, lots of additional privacy news, some errata and miscellany, actions U.S. citizens can take to express their dismay over recent Net Neutrality legislation, and some quick closing-the-loop feedback from our terrific listeners.

High quality (64 kbps) mp3 audio file URL: http://media.GRC.com/sn/SN-611.mp3
Quarter size (16 kbps) mp3 audio file URL: http://media.GRC.com/sn/sn-611-lq.mp3

SHOW TEASE: It's time for Security Now!. Steve Gibson is here. There is so much to talk about. A little more detail on how, kind of surprisingly, how bad that Intel AMT bug is and how poorly - in fact, this could be the show of just bad programming all around, including a missile that's got twice the hardware memory because memory leaks, why fix them when you can just double the memory? Yes, I'm not kidding, it's Security Now! coming up next.

Leo Laporte: This is Security Now! with Steve Gibson, Episode 611, recorded Tuesday, May 9th, 2017: Go FCC Yourself.

It's time for Security Now!, the show where we cover your security and privacy online. This is our guru, our sensei, our doctor of security, our Leonard Nimoy of security, Steve Gibson of GRC, Gibson Research Corporation, and a man who's been around this block a few times. Hi, Steve.

Steve Gibson: Hey, Leo, great to be with you again. So on Sunday John Oliver on his "Last Week Tonight" show gave us a shortcut to an otherwise very difficult-to-find page on the FCC.gov website. The domain that they chose is characteristic and typical of John Oliver and that. It's also the title of this week's podcast: Go FCC Yourself.

Leo: But it's a double meaning because you can actually go yourself to this FCC page now. This is a URL shortener, or a redirect.

Steve: Correct, correct. Yeah, exactly. They established a redirect to help people find it.

Shortly after the show aired, the site went down because...

Leo: Everybody went there. As it always does when John Oliver points to your site, yeah.

Steve: Exactly. So we'll talk, we'll rap this up this week by talking about that. And also the EFF has a much more polite domain called DearFCC.org. So I would recommend that our listeners who care about and are concerned about the potential loss of Net Neutrality pending the comments that we discussed last week, might want to go to both GoFCCYourself.com and DearFCC.org and give them some short comments. The FCC site, the GoFCCYourself.com page, looks like - I don't know if the site is slow because it's so busy. There isn't a way to get a total number of comments. You can see over on the left there, Leo, that the 17 dash whatever that is, 108, I think, that will take you to the history of past comments.

Leo: Is it open for comments now?

Steve: Yes, yes.

Leo: Oh, my.

Steve: And if you go to the prior page, you type that +Express link, and that will take you to a short form that you fill out to demonstrate that you're a U.S. citizen and that you're going to be affected by the legislation and so forth, and then submit it. But unfortunately, you can't get a total. We can't see how many there are. I sorted by submission date backwards and saw that the comments began appearing on April 27th. And if you look at the most recent ones, they're all today. So they've been - comments have been submitted, but there isn't a way to know how many hundreds of thousands of them there are. But I think the message is getting through.

Leo: I'm filing right now. I have some comments. I got some comments for you, FCC.

Steve: So we naturally have a lot to talk about. We're going to talk about the updates on the Intel AMT nightmare. Tavis was joined by a co-Project Zero worker, Natalie, and revealed a horrific problem that he found - finally he's turned his sights away from LastPass, thank goodness, for a while, aimed them at Microsoft, and found an incredibly bad flaw which is patched today, in today's Patch Tuesday for May, making this a very important thing to do because Tavis, as soon as the patch became available, presented full disclosure. And we'll talk about what that is. And this is a classic problem of security software making your system less secure by creating a new attack surface. And what's fun, I love this, you can't even download the proof of concept because the Windows scanner crashes when it sees the proof of concept come in because of course it's looking at everything that comes in, thus the problem.

Anyway, we got Google's May Android patches. A bad problem with second-factor SMS authentication being breached. Of course the big Google phishing problem that they had last week. The emergence of ultrasonic device tracking that we talked about sometime ago, but kind of like, eh, really? And it ends up, yeah, really. Lots of additional privacy news. Some errata, miscellany. And then we'll come back to and more formally talk about this Net Neutrality feedback option, and then some quick closing-the-loop feedback from our terrific listeners. So, yes, another jam-packed podcast.

Leo: Yay. I'm so excited. And you've interrupted me, but I'm almost done with a comment to the FCC. And I will complete that because I think it's very important that everybody stand up. And I have actually some standing here because, if it weren't for a free and open Internet, TWiT, my business, my job would not exist. We've been doing this for 12 years. And because we have equal access to our audience, to you, via the Internet, we can compete with big companies like Netflix. But it requires that Comcast give us equal access, that we be at a parity with big companies. And we don't have the vast resources to pay for that access. We rely on a free and open Internet. So I think I have something to say about this; you know?

Steve: I think it's really critical that the providers of our connectivity, the so-called "last mile," the ISPs, be regulated as a common carrier, that is, a neutral carrier of content, independent of what that content is, where it originates from and so forth. That's just - that's the way it has to be.

Leo: Yeah. Clearly. And somebody said, well, we don't want government regulation of the Internet. It's not government regulation of the Internet. It's government regulation of companies, calling on the service providers to preserve the Internet.

Steve: Exactly.

Leo: And those companies have proven themselves not to have the same interests we do.

Steve: Yeah, I mean, you and I are both entrepreneurs. We're both small company owners. We love and appreciate the concept of capitalism. But what all of the evidence demonstrates is that unrestrained capitalism is not in the benefit of the individuals that would like to avail themselves of it.

Leo: Not when you get to monopoly size, and that's what we have.

Steve: Yes. As soon as companies get big, they start leveraging their position in order to reduce choice and to eliminate competition. They're acting in their own best interest, but that's not the same as the public interest. And so I think you could argue, I think our government does have a responsibility and a role in doing things that keep companies' actions in the interest of the public to the degree necessary. So, yeah. I'm glad, I've heard you make the distinction between regulating the Internet versus regulating the providers of our connectivity. Those are different things.

Leo: Yeah, they're the gatekeepers. And if they're not kept scrupulously fair, their gatekeeping can create an imbalance and mean that innovators like you and me don't have equal access to our customers, that only the big companies will. And that's not what we want.

Steve: Right. Speaking of which...

Leo: Yes.

Steve: Our Picture of the Week...

Leo: Oh, let me get it here.

Steve: ...is a year-to-date graph of Internet-wide port 16992 scanning activity.

Leo: Ooh. It's gone up a lot.

Steve: Yeah.

Leo: What's going on? What is that?

Steve: Well, that's the AMT port, the Intel management engine port.

Leo: Oh, wow. Oh, wow.

Steve: And what this shows is that there was an initial - first of all, it had been all quiet, just kind of little bumps, probably just random scans of things, all the way through the year, until looks like about maybe March 10th, when there was a weird spike in hits per day. This was at the SANS Institute honeypot, where they maintain a sort of an Internet-wide scanner. Then it kind of quieted down but was, you know, more active. And then later in March, and pretty much through all of March, scanning was dramatically higher.

And what's interesting is that that's true despite the fact that there was no public disclosure of the IME AMT problem that was the title of last week's podcast, until May Day, until last Monday. So something was going on. Somebody knew that something was happening, that there was a potential exploit. And in fact 16992 is the non-encrypted port which, if AMT was "provisioned" is the technical term that you keep seeing, but enabled on a system that was exposed to the public Internet, this would allow connection to that port.

And what we learned at the end of last week, between these two podcasts, between last Tuesday's podcast and today, is that the problem was unbelievably bad, such that - and I heard you talking about it over the weekend also, Leo - such that Intel describing it as an "escalation of privilege" is almost laughable.

Anyway, we will talk about that because the technical detail of the bug in the firmware was amazing, and it's just great material for the podcast. But for what it's worth, in March - and it looks like it sort of - I guess this graph cuts off, so we don't know about April. We don't know what April looked like. But at least all of the year was quiet until a little blip early in March, and then a lot of scanning through March for this port that would have allowed Word exposed on a system where the enterprise administrators had enabled this management technology, would allow those machines to be taken over trivially. And we now have all the details that we'll be talking about in a moment. And in fact right now because that's the first topic.

So what we learned was that some guys at - and I had their name written down, I don't see here. Embedded, was it? I'm not seeing their name. They're a Berkeley company. I was trying to go off of my notes. I'll come to it. So as I said, late last week, actually it was Thursday or Friday, depending upon where your international dateline is, we found out that, through some independent discovery, what was going on. And also apparently some disclosure when the news got loose.

The folks at Tenable wrote on their site: "On May 1, 2017, Intel disclosed the AMT vulnerability, but details of that vulnerability were not made public." Thus of course on last week's podcast all we were able to say was, yes, we were worried about the idea of this low-level firmware-based separate CPU that was underdocumented/undocumented, that Intel was trying to keep anyone from knowing too much about where information is available only under license, and a few companies are manufacturing management tools. Intel does have some that you're able to download to talk to this.

Anyway, Tenable says their researchers were able to overcome the challenge of nondisclosure and, they say, make Tenable the first to deliver Intel AMT vulnerability detection capabilities to their customers shortly after Intel's announcement. So to give us a little bit of background into what's necessary, I took a snippet from their coverage. They said: "The first thing our research team tried was to set up a known vulnerable target. After some searching, we found," they wrote, "a Dell computer that had Intel AMT support, but there was a problem. It was not configured/provisioned for what we needed.

"The Intel Management Engine Interface driver was installed, but the Local Management Service (LMS) was not. Intel's AMT documentation says the AMT configuration tool acuwizard.exe requires LMS to be running. So we searched and found a software package for installing LMS on Dell's website. After LMS was installed, we were able to configure and provision AMT on the computer, giving us access to AMT via the web interface."

So first of all, one of the very confusing things about all of this is that there's still some lack of clarity on local access versus network access. And this is crucial because you could have a false sense of security if you verified that, for example, this LMS service was not present. Well, LMS service that these guys are talking about is the way within Windows to talk to the underlying technology. But its lack of presence doesn't mean that the underlying technology is not there. There are aspects which must be enabled. There are other aspects that do not need to be enabled and cannot be disabled.

And, I mean, and this of course is what we were talking about when we discussed this originally some time ago when we described it as a backdoor onto hardware that cannot be disabled. There's some subtlety to that, but it is true. That is, all systems can be pinged using a different port that we'll get to in a second. But so as I understand it, the LMS driver needs to be put onto a system in order for the hardware to be enabled, or some aspects of this to be enabled for non-local network access. Thus that would create the vulnerability. And that's why I explained earlier that this port scan would have been for a system, probably on an enterprise network, where AMT was being used to manage it, so it had been enabled for network access rather than only for local access, which would otherwise be the default.

So this is better than the concern I expressed last Tuesday that any system that had this was vulnerable. Any system that had it could be found with an AMT ping, but the ports that allow this access are probably disabled for network access by default, but not local access. And then there's some other confusion about whether netstat, the netstat command can be used to reveal these ports because that would be local, and maybe network, depending up on what netstat returns.

So again, there's still some confusion. But it is not the case that every motherboard that has IME and the Intel Management Engine and AMT is remotely attackable out of the box, that it does need to be enabled. When it is enabled - and this is where everyone was running around with their hair on fire at the end of last week. When it's enabled it's rather horrifically broken.

So there are a bunch of ports. There's 16992, which is described as Intel's AMT HTTP. The AMT service actually runs a web server. And so, for example, you can use a browser or an Intel client app to connect over HTTP to your own system's port 16992. Or if it's enabled for network, for remote network access, anybody can connect to this web server at port 16992. The next port, 16993, is the equivalent for HTTPS. So it's very much like the traditional public web ports are port 80 for HTTP and 443 for HTTPS. Intel has 16992 and 16993, respectively, for HTTP and HTTPS.

Then there's two additional ports: 994, which is called the AMT redirection for TCP; and 995, which is the redirection for TLS. So similarly non-encrypted and encrypted for system redirection, which is storage redirection, KVM, in order to do remote management of the system. Then there are two additional ports. And I couldn't tell, I think these are either ICMP or UDP. I don't believe that they're TCP. And those are lower numbered ports 623 and 664, which cannot be disabled.

Oh, I forgot to talk about when they're enabled and disabled. 16992, the first one, the AMT HTTP, Intel's documentation says this port is always open locally, but it may not be opened to the network. So you can always get to that from your own local machine, but externally you may not be able to get to it. And apparently 993 tracks that as available when TLS is enabled, and that can be done additionally and separately.

Port 623, Intel's documentation says, is always enabled. And 664 is also always enabled. And then the last port is 5900, which is familiar to people who have used VNC. That's the remote control program, and their system supports VNC for using KVM connections to the system. So there's a bunch of ports that, in the worst case, machines can be pinged. And I don't know what the constraints are for remote network. But the pinging needs to be done from - I forgot to mention, both the source and the destination ports must be the same, 623 and 664. That's important because those are below 1024. And all ports below 1024, by convention, can only be accessed by system-privileged applications; that is, an application running in user space cannot bind itself to a port below 1024. And that's deliberately done.

So Intel essentially added a little bit of additional protection. You need to have a process running with system privileges in order to emit traffic from that system's own port 623 and send them to the same port on a remote system. But that does allow you to ping the presence of one of these Intel-based systems on the network. And that can't be turned down. So at least it does allow a way of probing for those.

So the guys at SSH.com wrote: "This is like giving everyone with Intranet access" - so local Intranet access - "root privileges on every server whose AMT port they can communicate with," they wrote, "including janitors who can plug into the internal network. This also means root access to every virtual machine, container, and database running on those servers." And they wrote: "People with internal firewalls and dedicated management networks are in a better position," meaning that a dedicated management network would not be part of the globally accessible Internet.

Leo: And if you're not using the Ethernet port that's controlled by AMT, like if you're using WiFi to get on the 'Net, you're fine, too; right?

Steve: Correct.

Leo: It's not like sitting there waiting to scoop up all the traffic. It can only see what comes in on that NIC.

Steve: Correct. And in fact, when I was having that IP address conflict because there was ARP going on for it to obtain an IP for itself, nothing I could do was able to disable that on my server. Finally, when I moved the connection to the secondary NIC, the problem just went away. And only on the primary NIC.

Leo: And they say it's vPro-enabled Intel chips only. That's not all.

Steve: Correct. Although I did see some statements here in the intervening week saying that there were some non-vPro that also had this.

Leo: It's my understanding that AMT is in all the chips. It's just not enabled in the non-vPro chips.

Steve: Ah, okay.

Leo: And perhaps enabling, you know, that's a soft term. Who knows, you know. If the hardware is there, the code's there, maybe they can get it to run, even if it's not a vPro chip.

Steve: And this also raises the specter of something getting into your system and enabling it behind your back.

Leo: Right.

Steve: And thus basically opening a backdoor. And then Universal Plug and Play comes in and maps the ports through, so you're no longer hiding behind your NAT router. You're able to map an incoming port through to essentially a backdoor that was opened by someone that briefly got into your system. So it's sort of spooky to have this. On the other hand, you could argue that, if malware is in your computer, and it can use Universal Plug and Play, it could just map a backdoor to itself. So it's not clear that that's a lot better. Although turning it on would probably leave it on, so it would be probably subject to living through a reboot, where malware might not be or might be found. Nothing is going to - because this exists outside of the OS, no malware scanner would find this, either, because it isn't malware, it's just something that's turned on.

So what happened - oh, it's Embedi, the guys in Berkeley, California are Embedi, E-M-B-E-D-I. And they specialize in embedded system security. They reverse engineered and examined Intel's AMT code and posted their results. And I don't think this was responsibly disclosed because, as I understand it, we don't yet have patches. Patches are supposed to be happening this week. But to characterize what they found, and I'm going to explain the technical details next, but Ars Technica headlined: "The hijacking flaw that lurked in Intel chips is worse than anyone thought."

They wrote: "A remote hijacking flaw that lurked in Intel chips for seven years was more severe than many people imagined because it allowed hackers to remotely gain administrative control over huge fleets of computers without entering a password." And this is what I heard you refer to over the weekend, Leo, that you didn't need to use a password.

Leo: Anything worked.

Steve: This according, well, not...

Leo: How could you write software like that?

Steve: Okay, well, here it is.

Leo: Okay.

Steve: "While studying the Intel AMT Implementation and Reference Guide, the researchers learned that various AMT features are available through the AMT web panel, which is supported by [as I mentioned] the integrated web server, which listens to ports 16992 and 16993. To protect the AMT from unauthorized access, the web server provides several methods of authentication and authorization of a remote user. Intel AMT supports both Digest and Kerberos" - I always pronounce this wrong - Kerberos authentication. I just don't say it...

Leo: I'm not stepping in here.

Steve: "...Kerberos authentication, although the admin account always uses Digest authentication." And in fact the Embedi guys wrote: "An admin account which is present by default and always uses Digest authentication seemed like an interesting thing to dig deeper into."

Now, we've never talked about Digest authentication because it's not something that end-users typically encounter. But it's a longstanding RFC-based means for a client to prove that it shares a secret with a remote server. And it's an HTTP protocol-based challenge-and-response authentication mechanism. So the way it works is this. The client first issues a query to the server, just a standard HTTP, like I want index.html, whatever. I want a page. Well, because that first query won't have the additional authentication stuff, the query is rejected by the server with an HTTP 401 Unauthorized response.

Now, for example, we're all familiar with the famous 404 Not Found. When you go to typically an old-school web server and ask it for a page that the server doesn't have, it returns a 404, the Page Not Found. So a similar response is the 401, which says to the client that made the request, sorry, you need authorization if you're going to access that resource on the server. And that 401 Unauthorized response, in its headers it includes a realm header and a nonce header. You know, a "nonce" is a one-time gibberish-looking thing that will never be repeated, which prevents replay attacks. So the server says, here's some gibberish. You need to prove that you know the secret. So take this gibberish and do something with it.

So what happens is, if the client believes it knows the secret, it has a username and password that it has been previously assigned. Those are essentially its compound secret. It takes the username, concatenates the realm that was returned by the server, and so that concatenates the password and takes that string and does an MD5 hash. MD5 is an old-school 128-bit hash that's been around forever. So basically it takes a username, the realm, and the password, concatenates them, takes the hash. Then it takes the access method, which is typically like a GET, but it could be a HEAD or a POST, depending upon what it wants to do, and concatenates the URL that it's asking for, and it does another MD5 of that. And then, finally, it takes the first hash, concatenates the nonce that was returned by the server, and then concatenates the second hash, and hashes that to create a final MD5 128-bit hash.

So as you can see, that basically is a hash that uses secrets that the client has, information that the server provided, some information that will never be repeated, the nonce. It does three different MD5 hashes on all that stuff to create a 128-bit result which, when converted to hex, since hex is four bits per character, that would be 32 hex digits. So the idea is that the client, just to sort of restate this, makes an unauthenticated query which induces the server to respond, sorry, 401 Unauthenticated. But here's some stuff which you can use to authenticate yourself. So the client, taking its secrets and the stuff the server provided in the 401 Unauthorized reply, does all this hashing like crazy, and then responds this time with additional headers in its query, providing the final hash of all of its results, saying, okay, fine. Now give me what I want.

What the server of course does is it also knows - it knows what stuff it's sent to the client. It knows the username and password that the client has. But notice they didn't exchange them, so that's the shared secret. At no point do the username and password actually go over the wire. But they both know what they should be. So the server computes using the same algorithm, the same hash that the client should have computed if the client knew the secret, the username and password. And then the server, of course, compares the two hashes, the computed hash that it generated and the authenticating hash that it just received from the client that is requesting the resource.

So the logic of the string comparison, believe it or not, is unbearably broken in Intel's code, to the point where it really makes you question, like, anything Intel would ever do. So first of all, the most obvious thing for any code to do that wants to verify an MD5 hash - notice that, as we know, hashes are fixed length; right? MD5, we're not using it any longer for, like, big things because it's 128 bits, just not enough. The hash doesn't have enough entropy in it to be strong enough. So SHA-1, of course, is 160 bits. That's better. But SHA-256 that we're now using, like its name says, is 256 bits, lots of bits, twice as many bits as an MD5 hash. But the RFC is old, and it's based on MD5, and that's what we're using. And in this case we're using it three times, and it's probably good enough for this.

But the point is hashes are always fixed length. By definition, that's what they do. So if you know that this Digest-based HTTP authentication is MD5, then anything the client provides has got to be 32 hex characters. So the first thing the code should do would be to check the length of the hash the client provided. Is it 32 characters? If not, there's a clear protocol error, and the authentication should fail. But Intel's code doesn't do that.

Failing that, properly written code that wishes to compare two strings, because these are two hex strings of 32 characters each, but if you're just going to compare two strings for equality, you should compare the lengths of the two strings to verify that their lengths are identical because, if they differ in length, then they can't be the same. But Intel's code doesn't do that, either.

So if you're not going to compare, if you're not going to verify the known length of the string, and you're not going to at least compare, verify that the lengths of the two strings are the same, okay, then you at least want to compare the contents of the strings through the correct and expected length of 32 hex characters. But Intel's code doesn't do that, either. Believe it or not, what...

Leo: What does it do?

Steve: What the researchers discovered was that Intel's AMT code was using the length of the client's provided string for the comparison. They didn't use the server's calculated string length. They used the client's provide string length. And if the client provided a string length of zero, no comparison was ever performed.

Leo: Wow.

Steve: And the strings were considered to be equal. I mean, you can't make this up. This is unbelievable. So, I mean...

Leo: You've got to wonder what engineer thought that was a good idea.

Steve: That's my point is this is unbelievably worrisome.

Leo: Why do you even type that? I mean, it's really puzzling. That's very puzzling.

Steve: Right. No, I mean, it is. So in the code there's a string compare where it's pointer to one string, pointer to the second string, and the length, the number of bytes to scan from those two pointers to verify their equality. You would think, first of all, the length to scan should have been coded at 32. But, okay, now, that could be troublesome because the client could have provided less than 32.

Leo: Right, right.

Steve: So you wouldn't want to scan past the end of the client's buffer. That's a classic buffer overrun. You wouldn't want to scan to the length of the string because it might not be null terminated. It might be much larger. So there you would overflow the stack allocation if you provided a much larger authentication. But this guy - but we always know we're expecting a 32-character string. It's a hash that's been converted from binary to hex, from 128 bits to 32 characters. It has to be 32 characters. So, but no. They didn't check for size, then hard code 32 in for how far to scan, failing after a size check. Instead they said, well, we'll scan as much as the user provided, even if it's zero.

Leo: Even if it's none.

Steve: Even if it's none. Oh, my lord. So you simply provide a response to the authentication challenge with a null hash answer, and it says, okay, welcome. You're authenticated as an admin with root privilege.

Leo: Perfect match. Wow.

Steve: Yeah. Who wouldn't want that on their motherboard?

Leo: Well, that's the problem. I mean, it should be even a higher standard for stuff that's going to be on a chip. I mean...

Steve: Lord, yes.

Leo: It's not in nonvolatile - how is it stored on here? Is it hard?

Steve: It's all in firmware.

Leo: It's firmware.

Steve: And so Intel is massively rushing out patches.

Leo: So, okay, good. So the firmware can be rewritten and replaced with something else.

Steve: Yes, after eight years.

Leo: Yeah. Well, you have to find it first. But you'd think they'd have had a code audit. I mean, that's the kind of thing code audits look for.

Steve: And this is the problem. Here again, I mean, we know that just being open source doesn't by itself automatically mean that the software is secure. But it at least provides an opportunity for an audit to be performed. And, for example, that's how some problems in TrueCrypt were found was when we, I mean, the good news is nothing horrible was found. But some things that could have been cleaned up and made better were then made better. So, but in something like this, where it's like, no, no, no, we're not letting you look, you just have to trust us, it's like, okay.

Leo: All right.

Steve: A researcher by the name of Natalie Silvanovich was the first to discover this. From Tavis's tweet, it looked like she brought this to his attention, probably brought him in in order to do some of the really deep stuff that we know Tavis does, whether he's showering or not. Tavis posted to the Chromium Project Zero blog, where he posts his things: "MsMpEng" - which is Microsoft's Malware Protection Engine - "Remotely Exploitable Type Confusion in Windows 8, 8.1, 10, Windows Server, SCEP, Microsoft Security Essentials, and more." Meaning, okay, this is the core - and that doesn't mean that Windows 7 is not vulnerable because that's Microsoft Security Essentials is there. This is the common code that Microsoft uses throughout their products for malware scanning.

And I'll just share what Tavis wrote because it's succinct and perfect. He wrote: "MsMpEng is the malware protection service that is enabled by default on Windows 8, 8.1, 10, Windows Server 2012, and so on. Additionally, Microsoft Security Essentials, System Center Endpoint Protection" - that was the SCEP that I talked about before - "and various other Microsoft security products share the same core engine. MsMpEng runs as NT AUTHORITY\SYSTEM without sandboxing" - means it's the kernel, it's the root - "and is remotely accessible without authentication via various Windows services, including Exchange, IIS, and so on."

Now, wait a minute. "Is remotely accessible." What he means by that is, because it is listening to everything that's happening in the system to protect it, that it is an exploit surface. It is exactly like we've been talking about other third-party antivirus products reducing the security of the system by introducing their own flaws. Because they're trying to get - they are essentially a man in the middle. If they're not perfect, then they create new problems that weren't there before. And that is exactly what's happened with this core component that is ubiquitously present in all Microsoft platforms.

He writes: "On workstations, attackers can access mpengine by sending emails to users. Reading the email or opening attachments is not necessary." Again, because when your email client stores the email after receiving it on your system, it gets scanned by the mpengine. Visiting links in a web browser, instant messaging and so on, all represent vulnerabilities. "This level of accessibility is possible because MsMpEng uses a file system minifilter" - which is a Windows API term - "to intercept and inspect all file system activity. So writing controlled contents to anywhere on disk" - in browser caches, temporary Internet files, downloads, even unconfirmed downloads, attachments, et cetera - "is enough to access functionality in mpengine. MIME types and file extensions are irrelevant to this vulnerability, as MsMpEng uses its own content identification system. Vulnerabilities in MsMpEng are among the most severe possible in Windows due to the privilege, accessibility, and ubiquity of the service.

"The core component of MsMpEng responsible for scanning and analysis is called mpengine." He writes: "Mpengine is a vast and complex attack surface, comprising hundreds of handlers for dozens of esoteric archive formats, executable packers and cryptors, full system emulators and interpreters for various architectures and languages and so on." I mean, for anyone who's been following this podcast and understands how much vulnerability this represents, it's just mindboggling.

He says: "All of this code is accessible to remote attack. NScript is the component of mpengine that evaluates any filesystem or network activity that looks like JavaScript. To be clear, this is an unsandboxed and highly privileged JavaScript interpreter that is used to evaluate untrusted code, by default on all modern Windows systems." He writes: "This is as surprising as it sounds. We have written a tool to access NScript via a command shell for testing, allowing us to explore and evaluate it." And I'll skip all of that detailed stuff. All of the details are there in the posting that I do have a link to in the show notes, for anyone who's interested.

But it finishes, saying: "Before executing JavaScript, mpengine uses a number of heuristics to decide if evaluation is necessary. One such heuristic estimates file entropy before deciding whether to evaluate any JavaScript. But we found that appending some complex comments is enough to trigger this. The attached proof of concept demonstrates this. But please be aware that downloading it will immediately crash MsMpEng in its default configuration and possibly destabilize your system, of course, because you can't download it because MsMpEng will evaluate it and in doing so trip the fault that Microsoft has patched in today's Patch Tuesday update. Extra care," he says, "should be taken sharing this report with other Windows users via Exchange or web services based on IIS and so on. As mpengine will unpack arbitrarily deeply nested archives and supports many obscure and esoteric archive formats such as Amiga ZOO..."

Leo: Gotta have that.

Steve: Yeah, "...and MagicISO UIF format." He says: "There is no practical way to identify an exploit at the network level, and administrators should patch as soon as practically possible." He says: "We have verified that, on Windows 10, adding a blanket exception for C:\ is enough to prevent automatic scanning of file system activity." And he says: "You can still initiate manual scans, but it seems prudent to do so only on trusted files." On the other hand, adding a blanket exception to C:\ basically disables the whole system.

Now, Microsoft has an advisory, 4022344. What I did this morning was I looked at the version of the Microsoft Malware Protection Engine before installing today's patches. And, for example, on my Windows 7 machines I open Microsoft Security Essentials and so bring up the dialogue. In the upper right is a little Help with a triangular dropdown menu. Drop the menu down, choose About. That pops up a dialogue, and you'll see a number of version things there. Before I installed the updates, it said engine version was 1.1.13701.0. It was vulnerable. I installed the Patch Tuesday updates, looked again, and it was now 1.1.13704.0. So you want 704.

And in Microsoft's advisory they say: "For affected software" - which is to say everything - "verify that the Microsoft Malware Protection Engine version is 1.1.13704.0 or later." So this is bad. I mean, this is, like, unbelievably bad because essentially it means that anything that allows, like any means by which a Windows system will write something that it receives to disk, even if it's in a cache, if it's email, if it's, I mean, like anything that comes in, essentially, has an opportunity to trigger this. It's now known. There's no doubt that the bad guys are going to be attacking it.

And for everyone listening to this podcast who is in the process right now of installing their Patch Tuesday patches, you're going to be okay. But we know that there's a long tail on patch installs and updates. And this is bad. As far as we know, it has not yet been weaponized. But that seems like just a matter of time.

So, I mean, I wish there was a - I don't know why it was that Tavis, I mean, Tavis does disclose Project Zero things as soon as they are fixed and the patches are made available. It would be nice if there had been a few months of time for them to actually get installed because this seems really bad, given that it can be leveraged. I mean, in the worst case it's a denial of service, meaning that it will crash the recipient's computer. But as we know, that's where these things begin, and they quickly get turned into remote code execution exploits. It's hard to imagine this won't happen because any system that has this, I mean, even XP had an old version of Microsoft Security Essentials. I mean, this is widespread.

So I don't know when this got introduced. So it may not be that really old ones had this. I didn't see if the earlier versions were not affected. But this is, you know, always in Windows, always on, always protecting people, but only if it does a perfect job. And this looks like there's a good chance that it might not. So Natalie, props; and Tavis, thank you for the compete teardown of this. Yikes.

And speaking of Patch Tuesday, I'll just say that across all of the platforms, more than 20 vulnerabilities were fixed in every platform. They varied a little bit, the exact number, from platform to platform. But in every case four were rated as critical, and this was one of those. So definitely this is one you don't want to delay on fixing. I mean, really, immediately you want to get yourself updated.

Meanwhile, Google on the 5th released their May updates, which they documented on the 1st. They fixed 17 critical vulnerabilities, six of which affected the Android media server components - we've talked about Stagefright in the past, and it's been a real source of problems for Google and Android - which can be used to execute malicious code remotely. They also fixed four critical vulnerabilities related to the Qualcomm components in Android's handsets, including Google's Nexus 6P, the Pixel XL, and the Nexus 9 devices.

And Google wrote: "The most severe of these issues is a critical vulnerability that could enable remote code execution on an affected device through multiple methods such as email, web browsing, and MMS when processing media files." So I know that non-Google Android devices are far less expensive than Google's. But we're learning that these are computers, and they need to be kept updated. And so were I to be using an Android device, I would just get it from Google because...

Leo: Yeah. And Google's are the best, frankly. Not necessarily more expensive than, say, a Samsung Galaxy 8. But I was just looking at my Samsung Galaxy S8. It only has the April security patch. My Google Nexus, I'm sorry, Pixel got the May update, like May 5th, immediately.

Steve: Right.

Leo: And that's what you want.

Steve: Right. And thank you. I forgot to mention that Google said security patch levels of May 5, 2017 or later address all these issues.

Leo: Yeah.

Steve: So just check to make sure that you're up to date.

Leo: The Sami is vulnerable. The Pixel XL is not.

Steve: Yeah.

Leo: That's too bad.

Steve: So not surprising to our listeners, there has been a big breach of texted second-factor authentication. And everyone will remember when I was setting up my Hover account that, when I was offered a choice of time-based one-time passwords, TOTP, versus texting, I didn't hesitate. I knew that time-based was more secure. Because the last thing I want is every time I need to authenticate, to have that be an opportunity for some attacker to obtain that code. So it's much more secure to get it once over a secure channel when you're setting up your account and then use a TOTP authenticating app to dynamically produce the magic six digits.

So the problem, as we talked about back some time ago, is with SS7, the so-called Signaling System 7, which is the very old technology still used today to interconnect more than 800 telecommunications companies. But as we've discussed, the system is so old that it lacks any kind of endpoint authentication. It is an entirely non-authenticated system. But despite that fact, we have taken, we the world, to sending secondary identity authentication factors through the unauthenticated mobile telecommunications system.

Leo: Sigh. Yeah. Sigh.

Steve: What could possibly go wrong? So finally, and foreseeably, in January attackers exploited these well-known SS7 weaknesses to bypass the second factor in authentication that banks were using to prevent unauthenticated or unauthorized withdrawals from online accounts, after first using traditional banking trojan implants to perform the first stage of account compromise, which allowed them to learn account balances and get account information. They then selectively compromised the SS7 system to redirect the text verification messages banks used to verify funds withdrawals. So instead of those verification messages being delivered to the phones of the designated accountholders, the text messages were delivered to numbers controlled by attackers, who then used both the information gained from the banking trojan and their interception of the "are you sure you want to transfer all the money out of your account" confirmation code to do that, transferring them to their own accounts and making off with it.

Leo: Wow.

Steve: And these attacks were confirmed by the affected banks. Not good.

Leo: Unh-unh.

Steve: Now, I noted that - I just recently cleared the cookies from a browser because I was doing some experimenting, and saw that I was no longer authenticated through browser interaction on Twitter. And of course I had set Twitter up with a time-based token. But I remember at the time that it didn't - that I was, like, having to turn on both SMS and time-based. And I found to my annoyance yesterday that I reauthenticated with my time-based token, but I did also receive an unwanted SMS. So I need to see whether I can turn that off now and keep time-based, but disable SMS. Or maybe I'll just give it a bogus phone number, although I don't want it going anywhere else. I just don't want them to send it.

Leo: Send it to 555-1212. That's directory assistance.

Steve: Ah.

Leo: Yeah.

Steve: Cool.

Leo: I doubt very much that anybody's going to hack that. It's not a cell phone, for one thing. What if you sent it to a landline? Maybe they wouldn't even accept it because you couldn't verify it. That's the problem. You can't send it to a nonexistent number. You have to verify it.

Steve: Ah, right, because they're going to send you a verification text.

Leo: Yeah. Google does, and a number of companies do offer - so they'll say, I could send you a text or even call you with the verification.

Steve: Right.

Leo: So you can use a landline. And the landline would not be, well, certainly not to SS7. It might be something else that it could be vulnerable to.

Steve: Yeah. I just don't like the idea of a per-authentication...

Leo: That's why we have authenticators.

Steve: Yes, exactly.

Leo: Yeah.

Steve: So we need to talk about last week's massive phishing attack on Google...

Leo: Oh, good, I was hoping you would.

Steve: Google users, yeah.

Leo: Google Docs, yeah.

Steve: Yeah, Google Docs. One of our listeners sent a note saying: "Exploit is spreading quickly. I've received it twice today."

Leo: Oh, yeah.

Steve: And Google was very comprehensive in their response. On Wednesday they wrote, or they said: "On Wednesday, May 3, we identified, investigated, and resolved an email phishing campaign that affected some accounts in your domain." So this was sent to people who Google verified after the fact had been affected. This issue was addressed within approximately one hour from when Google became aware of it. Although we could argue, and many have, that Google was informed in 2011 by some researchers who knew of this problem. And then apparently the code, a proof of concept, was posted on GitHub in February. And it's believed that in fact the GitHub-based proof of concept may have been used by the attackers to help them create this attack. So one has to question whether Google couldn't have fixed this beforehand.

Anyway, they said: "To assist you in understanding what happened and better educating your users on email security, we are sharing details on how the campaign worked and how we addressed it. We're also providing a CSV file identifying the users on your domain who were affected." So they said: "The affected users received an email that appeared to be from a contact" - like one of their contacts - "offering to share a Google Doc. Clicking the link in the attacker's email directed the user to the attacker's application, which falsely claimed to be Google Docs, and asked for access to the user's account. If the user authorized the application, it accessed the user's contacts for the purpose of sending the same message to those contacts."

So it was a classic mail worm, essentially. "This access only retrieved contacts and sent the message onward. Customer data such as the contents of emails and documents were not exposed. Upon detecting this issue, we immediately," writes Google, "responded with a combination of automatic and manual actions including removing the fake pages and applications and pushing updates through Safe Browsing, Gmail, and other anti-abuse systems."

Then they wrote: "We have taken the following steps to protect your users: Disabled the offending Google accounts that generated the phishing link; revoked any access that the affected users authorized to the attacker; disabled malicious projects and apps that sought access. In addition, Google is taking multiple actions to combat this type of attack in the future, such as updating our policies and enforcement on OAuth applications, updating our email filters to help prevent campaigns like this one, and augmenting the monitoring of suspiciously behaving third-party apps that request consent from our users.

"Finally, as a general precautionary measure, you may choose to take the following actions regularly for your users: Review and verify current OAuth API access by third-parties, run OAuth Token audit log reports to catch future inadvertent scope grants, and set up automated email alerts in the admin console using the Custom Alerts feature or script it with the Reports API. We thank you for your continued business and support. If you have any questions, please let us know by contacting Google Support and referencing the issue number" - I hope that's not a count - "37950384. Sincerely, the G Suite Team."

So this is a problem that we have. And our listeners will remember that I have recently kind of gotten down on OAuth. OAuth I described, and I still do, as a really bad web kluge. I mean, it is. It's full of problems. It's had security flaws. But the underlying problem is that it is so prone to phishing. You go to a site that says "Log on with Google." "Log on with Facebook." And once upon a time this was a rarity. Now it's becoming increasingly common and is really subject to abuse.

In fact, a couple weeks ago someone in the SQRL newsgroup commented that some of SQRL's mitigations against phishing could be ignored, arguably, because users are being trained as a consequence of this "Log on with Something Else" to assume that what's happening with SQRL is correct. And so this is something that had changed, this prevalence of OAuth, in the years since I proposed SQRL and came up with this mitigation against the spoofing problem that had always been present. And because this has always been a worry for me, our listeners who've been following along may remember that summer before last I completely halted work on the project in order to focus on the phishing problem and ended up coming up with a solution, an absolutely bulletproof solution for same-device login, that is, where you've got SQRL running on your Windows machine, for example, or a Linux or a Mac machine using Wine. And you just click on the SQRL QR code to log in.

The idea is that by allowing the browser and the SQRL client to communicate, there is absolutely no way for a network-based attacker to spoof because the remote server that you're logging into returns the authentication secret to the SQRL client, which then uses its local communication with the browser to give it to the browser, which the browser then uses to log you on, thus cutting any attacker out. It makes it absolutely spoof-proof.

But around that time Windows 10 was in the process of getting ready to happen. And there was the threat that Microsoft in Windows 10 was going to cut off access for their Edge browser to the so-called "localhost" IP, which has always been present: 127.0.0.1. Localhost is used for all kinds of things. But Microsoft decided, eh, you know, we just think it would be better if we didn't allow Edge browser to access localhost. So they announced they were going to shut that down.

Well, that's the way the browser could communicate with a SQRL client without needing an extension in the browser to do so. It made it just, like, automatic and transparent, and it would work perfectly. But we couldn't have SQRL depend upon a feature that Microsoft was about to kill in Windows 10. So I left the - we call it CPS, Client-Provided Session, meaning that the SQRL client provides the authorized session to the browser and solves this problem, I mean, better than anything else ever done, completely solves it. So I left that in, thinking that, okay, well, at least browser extensions for SQRL when they're created could take advantage of it to also solve the problem.

Two weeks ago I said, wait a minute, we need to revisit this. What finally ended up happening? Well, it turns out Microsoft was unable to follow through with their threat. There was such a backlash from the user and developer, especially the developer community, that they were not going to be able to use localhost, I mean, it's like all these Windows machines have IIS server built into them. You can create websites and check your JavaScript. And it's so useful to be able to access servers running in your own machine with your own browsers. Everyone who's a developer does that. And Microsoft was saying no.

Well, they were unable to follow through. So they changed the default shortly before release, enabling it by default. If you go in Edge, if you go about:config, I think it's that, or maybe it's about:settings, it'll bring up a page. And the second option there is to turn off access to localhost. But it is on by default. So by default everyone will be able to do this, and it's coming back into SQRL by default. So we will have, as I wanted a year and a half ago, bulletproof spoof protection in SQRL. And things like man-in-the-middle attacks and website spoofing and so forth cannot affect SQRL.

So I hadn't talked about SQRL recently. I was in the process of working on the installer and remover code as I get this thing ready to be finished. But this brought me to a stop about a week ago when someone said, you know, users are not going to know that, if they're logging into Joe's Blog, and this thing says Amazon, that they're not, like, using their Amazon identity to log into Joe because that's what users are accustomed to now. And he was right in saying that. So we've got full spoofing protection back for same-device login. And I'm really glad for it.

Leo: Steve Gibson, I can tell he's perusing the newsfeeds, see what's happening out there.

Steve: Yeah, so anyway, yeah, as you said, James Comey, director of the FBI, just got fired. So he of course was in the process of investigating the Trump campaign ties to Russia.

Leo: I'm sure it had nothing to do with that.

Steve: Nothing to do with that. Yes, Trump apparently said, "While I greatly appreciate you informing me on three separate occasions that I am not under investigation, I nevertheless concur with the judgment of the Department of Justice" - of course that's Jeff Sessions - "that you are not able to effectively lead the Bureau," Mr. Trump said in a letter dated Tuesday to Mr. Comey. "It is essential that we find new leadership for the FBI that restores public trust and confidence in its vital law enforcement mission," Mr. Trump wrote. Well.

Leo: Wow.

Steve: Yeah. I did want to mention, just for people who were interested in the reach of Google, somebody tweeted me, and I'm glad for it. I forgot about this. I think we talked about it once: myactivity.google.com? Which is a somewhat sobering, but I would argue open-kimono look at what Google has.

Leo: Yeah. I mean, they tell you what they know.

Steve: Yeah, they do.

Leo: As opposed to other people who may not.

Steve: Yeah. So myactivity.google.com. And, I mean, again, you're looking, you're scrolling through all of your past searches. And it's like, oh, uh, yeah, you really are watching. So, yeah. Just, again, it's nice to see. I think you're able to delete things from there also, if I recall.

Leo: Yeah. To their credit, I mean, they collect everything. But they at least tell you what they collect, and they in most cases give you the opportunity to turn it off.

Steve: Yeah. So a paper which was presented last week at the second annual IEEE European Symposium on Security and Privacy, which occurred in France, in Paris, was titled "The emergence of ultrasonic cross-device tracking." Oh, I'm sorry, no. That was my title. "Privacy Threats Through Ultrasonic Side Channels on Mobile Devices." And I've been meaning to mention to you, Leo, we know that this actually can be done because that's how that cool little EKG monitor that you like operates.

Leo: Ah.

Steve: That little black strip with the two silver pads? The reason it says it needs access to your microphone is it uses frequency-modulated ultrasonics in order to instantaneously send the voltage that it's picking up between your hand into your phone, which then converts it into an EKG waveform. So anyway, what's a little worrisome, we talked about SilverPush before. That's this idea of applications surreptitiously listening for high-frequency, essentially dog whistles that are being used to convey information.

Four German security researchers analyzed a large repository of Android apps. In the abstract to their paper, they wrote: "Device tracking is a serious threat to the privacy of users, as it enables spying on their habits and activities. A recent practice embeds ultrasonic beacons in audio and tracks them using the microphone of mobile devices. This side channel allows an adversary to identify a user's current location, spy on their TV viewing habits, or link together their different mobile devices. In this paper we explore the capabilities and current prevalence and technical limitations of this new tracking technique based on three commercial tracking solutions." So these are being commercially offered as SDKs to people for this purpose.

"To this end," they write, "we develop detection approaches for ultrasonic beacons and Android applications capable of processing these. Our findings confirm our privacy concerns: We spot" - get a load of this - "ultrasonic beacons in various web media content and detect signals in four of 35 stores in two European cities that are used for location tracking. While we do not find ultrasonic beacons in TV streams from seven countries, we spotted 234 Android applications that are constantly listening for ultrasonic beacons in the background without their users' knowledge."

And it turns out these are not all obscure applications. McDonald's and Krispy Kreme are among the applications that are using this technology. So with the headline: "Hundreds of privacy-invading apps are using ultrasonic sounds to track you," ZDNet summarized it, saying: "A new privacy-busting technique that tracks consumers through the use of ultrasonic tones may have once sounded like the stuff of science fiction novels, but today it's reality. These near-silent tones cannot be picked up by the human ear, but there are apps in your phone that are always listening for them." I would say there may be apps in your phone.

"This technology is called ultrasonic cross-device tracking, and it works by emitting high-frequency tones in advertisements and billboards, web pages, and across brick-and-mortar retail outlets or sports stadiums. Apps with access to your phone's microphone can pick up these tones and build up a profile about what you've seen, where, and in some cases even the websites you've visited. The technology is still in its infancy, but it's growing in popularity."

And then in quoting this research, they said: "In the past year, researchers found 234 Android apps that include the ability to listen for ultrasonic tones without the user's knowledge. And these are not all obscure apps since numbered among them are McDonalds and Krispy Kreme. None of the 234 Android applications disclose their tracking capabilities in their privacy policies."

Anyway: "A Google representative said that the privacy policies enforced on all apps available in the Play market require developers to 'comprehensively disclose how an app collects, uses, and shares user data, including the types of parties with whom it's shared.' The representative" - that is, Google's representative - "didn't respond," Ars writes, "to a follow-up question asking why none of the five apps cited in the research findings disclosed the SilverPush functions. At the time this post went live," writes Ars, "all five apps remained available in the Google Play Store."

So I don't know what this means. It would be nice, for example, if it were possible to audit all of the installed apps for their access to the microphone so you could just, after the fact, look to see what apps had acquired permission, if you had given it to them at install time. The other thing that's annoying is typically apps, or they may not, allow you to selectively disable the features that they're asking for. I don't know how granular apps are under Android. But it would be nice if something...

Leo: It's pretty granular. But I don't - yeah, that's interesting. I'll have to look and see. I mean, you can. But that may break the app. I mean...

Steve: Well, but apps that don't have a reason to listen are, I mean, you could argue, you say, no, I don't want you using my microphone.

Leo: Yeah. Yeah, I'll look.

Steve: So we've been covering this story, and there was a little bit of news, so I just thought I would keep us current. And that is that in that bizarre case of the homicide in the hot tub that we've referred to a number of times, where the user was very IoT - the user. The homeowner who had the hot tub was very IoT enabled, and the use of a huge amount of water in the middle of the night was regarded as suspicious as relayed by his water meter that was IoT enabled. We will recall that Amazon was refusing under First Amendment rights, and I guess not Fifth Amendment, but saying basically "freedom of speech," saying they're not going to disclose what the Echo may or may not have overheard. So they've reversed themselves.

Leo: That got dropped, though; right?

Steve: Yes.

Leo: Yes, yeah, they said they would, but then the thing got dropped; didn't it?

Steve: Well, no. As I understand it, Amazon has recently agreed to release any data obtained by the - any data after the Arkansas defendant agreed to allow the authorities to ask Amazon to do so. And he said, "I have nothing to hide. You're welcome to do that."

Leo: Okay.

Steve: So they've been saying no. Apparently the guy said, yes, fine. Nothing happened. And so they said, okay, fine, we'll release what we have. Which is very likely little.

Also, very quickly, a Miami judge has now ruled that compelling password production is not a Fifth Amendment issue. That is, it is not compelling the production of your password. Giving it over to authorities does not violate your protection against self-incrimination. So we're seeing lots of judgments coming down both ways on this. I've got much more than I'll bother going through here. There was a nice coverage of it in Techdirt, who concluded, saying: "This decision will be appealed. But the decision cited by this judge appears to indicate that this will only delay the inevitable."

And, as I have said here, "Sooner or later, this issue will have to be addressed by the Supreme Court," but this writer in Techdirt says, "but I wouldn't hold my breath waiting for it to happen. The Supreme Court frequently takes a pass on timely issues, leaving circuit appeals courts to do most of the heavy lifting. There have not been sufficient Fifth Amendment cases of this type in federal appeals courts to press the issue. So far, the only thing that's been made clear in multiple cases is fingerprints are worse than passwords when it comes to locking law enforcement out of phone contents."

And then some news from the U.K. Newsweek reported, and also BBC picked up on it also: "According to leaked documents, the U.K. government plans to ask for powers allowing intelligence agencies to spy on people in real-time by introducing encryption 'back doors' to communications firms. Privacy advocate organization Open Rights Group obtained a leaked copy of the government's draft Technical Capability Notices - called TCNs - regulation, which it has published in full on its website.

"The document forms part of a so-called 'targeted consultation' into the Investigatory Powers Act, which was brought into law last year, meaning it has not been publicized to the tech industry or the public. Under the proposals, all communications companies - Internet providers, messaging apps, and phone networks - would be forced to provide police with real-time access to a person's web browsing with one day's notice."

Jim Killock, who is the executive director of Open Rights Group, said in an emailed statement to Newsweek for their coverage: "These powers could be directed at companies like WhatsApp to limit their encryption. But if the powers are exercised, this will be done in secret." Killock previously described the Investigatory Powers Act, referred to widely as the "Snooper's Charter," as the "most extreme surveillance law ever passed in a democracy." Writing an opinion piece for Newsweek last year, Killock said: "The Investigatory Powers" - is there a better way to say that, "investigatory"? I guess that's the way you say it.

Leo: Investigatory, yeah.

Steve: Investigatory. I don't know why I have a hard time.

Leo: Say IPA.

Steve: Investigatory. Yes, "The IPA mostly permits and codifies all the illegal practices revealed through whistleblowing and court action."

Meanwhile, here in the U.S., our ex-FBI director James Comey said that his organization is unable to access half of mobile devices, actually 46%, and supports new legislation. During Senate testimony - apparently his last - last Wednesday, Senator Chuck Grassley asked whether the FBI director still believed that it was not necessary to push for a law to solve the so-called "going dark" problem. Comey replied: "It may require a legislative solution at some point," adding "I could imagine a world that ends up with legislation saying, if you are going to make devices in the United States, you figure out how to comply with court orders." And he said also: "The shadow created by the problem called 'going dark' continues to fall across more and more of our work," and blamed the "ubiquitous default disk encryption on devices."

Unfortunately, given the people that Trump is appointing, I would not be surprised to see the next FBI director taking a stronger position.

Leo: Well, let's be fair. Our Democratic Senator from California, Dianne Feinstein, is also in favor of this.

Steve: Yes. Very, very good point. Yes, very good point.

Leo: It's a kind of - one of those bipartisan, apparently, issues.

Steve: Yup, I'm glad you said that.

Leo: Everybody can agree that privacy is dead.

Steve: So this next piece, I'm not going to go over it in detail. His posting, I have a link to this guy's posting in Google Docs for anyone who is interested. But this just is - this makes me sad. The coverage appeared in Ars, and Ars said that U.S. border agents threatened to, quote, "be dicks and take my phone if I didn't unlock it."

So this starts: "Albany, California. As he sat in a darkened corner of a neighborhood bar, Aaron Gach, G-A-C-H, an artist and lecturer at a local art college" - and, by the way, U.S. citizen - "told Ars about what happened to him in a February 2017" - so a few months ago - "episode at San Francisco International Airport, where he finally agreed to unlock his iPhone and have it be searched by border agents" - without cause - "rather than risk being detained and delayed further. A U.S. citizen, reentering the country at SFO International, is harassed without cause and detained for interrogation until he finally relents and unlocks his phone so that the customs investigators can rifle through it."

So I've skipped a lot of the back story. It's all online. And again, links in the notes. But this dialogue I wanted to quote. So they ask: "Can we check your phone to verify the info you provided?" And this is after a long back-and-forth interrogation of, like, where do you live, where were you visiting, what countries did you go to, are you sure, did you go to any others, are you sure, blah blah blah.

So he says: "This is where I began asking lots of questions. I also asked to see the written policies authorizing their actions, and we waited while they went to get a 'tear sheet.' Following are some of my questions and their answers." So they ask, I'm sorry, he says: "Is there a problem with my travel arrangements?" Their answer: "I'm sorry, but I cannot provide any details." "Is there a concern about the arts venue?" Answer from the border official: "I can't really say at the moment."

He asks: "What is it you want to check on my phone? Is it something in particular that I can just show you?" Answer: "We're looking for information pertinent to our investigation." He asks: "Do I have a choice in the matter? What are my rights in this situation? As a U.S. citizen, don't I have equal protections under the Constitution, regardless of whether or not I am in an airport or outside of one?" Their answer: "I understand your concerns, and I'm hoping we can get you on your way as soon as possible. Of course you have a choice, but we can also be dicks and just take your phone as part of our investigation if we see fit."

Leo: What?

Steve: "Your phone and its contents are part of your personal effects which are subject to examination when crossing any border into the U.S." He says: "That doesn't sound like much of a choice. What happens if I choose not to unlock my phone?" They respond: "We can detain your phone and any personal effects needed to assist in our investigation." And he asks: "For how long?" They say: "Not long. Just until we're done with it, and then we would ship it back to you."

Leo: Oh, like you're going to want it after that.

Steve: Exactly. And he says: "Well, how long would that be? Days, weeks, or indeterminate?" Answer: "Indeterminate."

Leo: Wow.

Steve: He says: "But I could leave?" And then they say: "As soon as we're done." And he says: "Am I also being detained, then?" "No, we're not detaining you, just your personal effects." He says: "So I can leave, then?" And they say: "As soon as we're done here. Hopefully we can get you on your way shortly." And he says: "Can I be present when you search my phone?" "No, I'm afraid not. But for this investigation I can tell you that we are only conducting a manual search, and not a digital extraction. However, I can tell you from others that have refused to unlock their phones that I can't make the same claim, and your information may be copied for later review."

Then they ask: "Is there any reason in particular why you don't want to turn over your phone?" And this guy says: "I believe strongly in the Constitution and in my right to privacy. I have nothing to hide, but the only way I know if I actually have any rights is if I try to exercise them. But it sounds like I don't actually have those protections in this situation." And it goes on. And as I said, it makes me sad.

Leo: Well, it's terrifying. American citizen, coming home.

Steve: Yes, yes, yes.

Leo: Ars says, well, the guy's an activist. He's involved in Greenpeace and Copwatch, but those aren't - he's never been arrested, has no arrest record. Basically they don't like him. And that's the problem is there's a lot of power given these guys. And it terrifies me. I'm traveling out of the country a couple of times this year. We actually decided not to go to Cuba because of this. I just don't want the hassle of coming back from Cuba.

Steve: Yeah, also in his story he had a laptop in his luggage. And although it was protected, there was information that it was not backed up. So he had information that he wasn't willing to lose if he continued this.

Leo: They keep it, yeah.

Steve: They started going through his luggage. And in order to head them off at the pass before escalating this, he finally capitulated.

Leo: The guy's not a terrorist. All right?

Steve: Right.

Leo: He's an American citizen.

Steve: Yes.

Leo: I don't understand this. We have a lot of visitors, international visitors. And fairly frequently they say, yeah, I'm carrying a burner phone, or I was very careful to wipe - I talked to somebody yesterday. "I was very careful to wipe my phone before I got off the plane" because of stuff we've talked about. But we should point out most people, I mean, it's a very small percentage of people that this happens to. And many of these people say nothing happened. But at the same time, if it does, you don't want it to happen to you. And there doesn't seem to be much of a rhyme or reason as to whom it's happening to.

Steve: Well, it's creepy. I'm a U.S. citizen. I've done nothing wrong. I've traveled abroad. I'm coming home.

Leo: I mean, they can search your luggage. They can search your luggage; right? This is - they think this is like searching your luggage. I would submit that it's a little more than searching your luggage, and there's a huge security risk if they take the phone.

Steve: Yes.

Leo: I mean, basically, if I give them my - I'm not going to bring an expensive phone out of the country.

Steve: No. I think that's exactly right. I think you either back up your phone and wipe it, and you travel with it like that, so they say, "Can we unlock your phone," you go, oh, yeah, here, sure. I don't even have it protected. Here, just turn it on. It's unlocked for you. And then when you get home you wipe it again, and then you restore from backup. You just...

Leo: I guess the real question is how does this protect us? Because if I'm a bad guy, the last thing I'm going to do is have the secret plans to my secret on my phone.

Steve: Right.

Leo: I'm not going to do that. So I'm not sure how this protects us. And it definitely sends a chilling effect to people who want to come to the United States, and to people like me who want to travel.

Steve: And it's sad for citizens. It's like, wait a minute. I mean, yeah. Wow. So photoshopping is a thing.

Leo: It is a thing. Yes, it is.

Steve: As we know, image editing, yes, image editing is so prevalent and has been available for so long that we have a new term, "photoshopping." Oh, that picture was "photoshopped." Meaning it wasn't actually, you know, that image that you're seeing never actually existed in the physical world. It was digitally created.

Well, it turns out that as a consequence of evolving technology and sort of speech recognition turned upside down, we are now seeing the emergence of voice-mimicking tools. Voice cloning technology has been in the works for several years; and, before long, just as with photoshopping an image, which has become a thing, so too will the ability to impersonate anyone saying anything. And lord help us when that happens. And when you think about it, traditionally, how many spy novels and movies and TV plots, I mean, like, wearing a wire, the reason you wear a wire is to capture somebody saying something that's compromising. So you pat down, are you wearing a wire, blah blah blah.

So traditionally playing back a compromising recording of someone saying something has been regarded as strong evidence of wrongdoing and created a strong belief that the subject in question had actually uttered those phrases because you could hear them doing that. But that assurance will soon disappear. Apparently there is already some software to which you can feed some chosen words, very much like in that "Mission Impossible," one of the recent ones, where somebody wanted to capture someone's voice so they restrained him and said, "Read this card," and it was a nonsense sentence. But the point was that the sentence contained expressions of all the various phonemes that were required in order to then synthesize that person's voice saying whatever they wanted to.

So anyway, I just wanted to put it on our radar. I expect before long we will start seeing our late-night comics no longer having to chop up things that people said in order to create out-of-context funny things, but actually just be able to synthesize someone's voice saying whatever they want. And so we lose, just as we can no longer believe an image, believe what our eyes tell us in an image, we won't be able to believe what our ears hear because it could just be somebody that was digitally synthesized.

I did want to give a caution to anybody who may have recently downloaded or updated their version of HandBrake for the Mac. HandBrake is a very popular DVD-ripping and media transcoding app, popular because it's just a GUI, and it's robust and very simple. I used it years ago when I wanted to copy DVDs that I owned for my own purposes, of course, fair use. And it's great. But over the weekend one of the official download servers, download.handbrake.fr, was compromised, and v1.0.7 was made malicious. So it's the actual HandBrake from the actual server is actually bad. So if you don't have 1.0.7, you're okay.

If you do have it, you could take the fingerprint of the file, the SHA-1 checksum. And in the show notes I show the checksum and some simple instructions for making the checksum. For example, if you open Terminal in a Mac and just type "shasum," space, and then drag the HandBrake.dmg file, drop it in the Terminal, that enters the path to the file as where the cursor is in the Terminal, and then hit Enter, and it'll show you the hash. And it should start out 0935a43. If it does, that's malicious. So I should have said it should not start out 0935a43 because, if it does, it's malicious. So anyway, if you didn't download HandBrake recently, don't worry about it. You've got 107 and did download it recently, you want to make sure you didn't get a bad one.

Two bits of errata. Many people pointed out our funny video from last week, that that was not the Cookie Monster. Apparently it was - I guess it was an early version of the Cookie Monster, or maybe the Cookie Monster was derived on "Sesame Street" from this monster on "The Muppet Show." But this was a Muppet monster. Cookie Monster was blue, so a different color, and did not have teeth the way the monster we showed last week did. Anyway, a number of people saying...

Leo: Sorry, Steve, but I know it's not the Cookie Monster, man.

Steve: Well, and also, people also took exception to my referring to emojis as "yellow Smurfs." They said, "Steve..."

Leo: Smurfs are not - they said this in the chatroom, and I chose to ignore it.

Steve: Yes. Smurfs are not yellow. Smurfs are blue.

Leo: Oh.

Steve: And several people suggested that maybe Simpsons are yellow, so they were Simpsons.

Leo: Smurfs are not yellow, Steve.

Steve: What I meant to say, of course, was that they were Smurf-like, but yellow in color. But I didn't make that clear. So...

Leo: It doesn't matter. Doesn't matter.

Steve: Yeah, I know. This is fun. Turns out all memory leaks are not necessarily important, Leo.

Leo: What?

Steve: Now, we know memory leaks are caused by the allocation of memory, which is not then freed after it's no longer required. So, for example, a program might explicitly allocate some memory for some purpose, like a buffer, to receive something. And if it forgot to release it back to the operating system, then when another buffer came in it might allocate another buffer to receive a buffer, and not release that. Over time, that leaks memory. All of those successive buffers are asked for from the operating system, and eventually the OS will run out. It'll say, I don't have any more, you've got it all, because it wasn't being released after use.

Then there's also automatic allocation, where a fancy language, for example, does dynamic memory allocation. For example, JavaScript has that, where you just use things, and it's up to the language to figure out, oh, look, he's not using that anymore, and it frees it. So you hope that the dynamic memory allocation, or the automatic allocation, is working right.

So I got a kick out of this. Posted to, back in the old days, the old Usenet on the Internet, the NNTP newsgroups - which of course GRC is still using - Usenet, the comp.lang.ada forum, so that was the computing language, Ada was a language forum, on March 31st - oh, and I forgot the year, I think it was 1995 - by Kent Mitchell, who was consulting to Rational Software Corp., that at the time was a big deal, in reference to previous postings by Mitre Corp. and IBM's Watson Research Center. So this is in the early days, when gurus hung out on Usenet, and it wasn't just a big massive porn database, which it later devolved into and then became unusable, which is why GRC runs a private newsgroup server, where we have wonderful groups of great people.

The subject was, does memory leak? And so Kent posts: "I was once working with a customer who was producing onboard software for a missile. In my analysis of their code, I pointed out that they had a number of problems with storage leaks. Imagine my surprise when the customer's chief software engineer said, 'Of course it leaks.' He went on to point out that they had calculated the amount of memory the application would leak in the total possible flight time for the missile, and then doubled that number."

Leo: Oh, what?

Steve: For safety. This is why North Korea...

Leo: Couldn't you just get rid of the leak?

Steve: This is why North Korea is having a problem. They didn't double.

Leo: Yeah, double the memory.

Steve: "They added this much additional memory to the hardware to 'support' the memory leaks. Since the missile will explode when it hits its target or at the end of its flight, the ultimate in garbage collection is performed..."

Leo: That's true.

Steve: "...without programmer intervention." And all of the leaked memory - we could say that the buffers had been destroyed in more ways than one.

Leo: Wow.

Steve: Isn't that fun? Yes.

Leo: Yeah.

Steve: Deliberate memory leak in a missile because it really doesn't matter.

Leo: Can you - wow.

Steve: As long as the leak will outlive - as long as the memory will outlive the missile's life in flight, who cares?

Leo: I can't think of any reason why you would allow, I mean, allow a memory leak. Wouldn't you just fix it? Wouldn't that be...

Steve: Yeah, that really does sound screwy.

Leo: Is there any compelling reason why a memory leak should exist?

Steve: You might say that it would take up more code. On the other hand, they've allocated a lot more memory to deal with the leak. Maybe the code was in ROM, and they leaked.

Leo: Oh, could be. They couldn't fix it.

Steve: And so they could have had limited ROM, and so they said, well, rather than make the ROM bigger, we don't care if the RAM leaks. So who knows?

Leo: Interesting.

Steve: I got a couple tweets relative to SpinRite that I appreciated. First was from someone who was confessant - it was one of those confessional weeks - named Steven. His Twitter handle is @Dillinger65. And he said: "Hi, Steve. I needed to pirate a copy of SpinRite, fixed income and all. SpinRite saved my old laptop's hard drive. One day I'll pay for a copy." Like, okay, well, thanks for confessing. I'm glad - and I said back, I said, "Okay, you know, glad it worked for you. Most people pay for their copy."

Leo: It's like Augustine. St. Augustine very famously said - what did he say? He said: "Make me good, lord, but just not right now."

Steve: That's right.

Leo: Not quite yet.

Steve: And I would appreciate payment. I mean, it is what allows me to do everything else I do. So when that stops happening, I'll stop being able to do all this.

Javier Figueiredo, he sent an interesting picture. He said: "SpinRiting at Level 2, a two-year-old Kingston SSD. Does this image look like a healthy disk?" Oh, no. So there are several things. First of all, so this is an SSD that again demonstrates SpinRite's value, even on solid-state media. The image is showing a bunch of red. And you don't want to see red there. This is on "ecc corrected."

And so what's showing is that this bar, the way SpinRite's SMART Monitor works is it always starts the bars off where they are, wherever they are, at the beginning of the scan because any reduction in the bar during the scan represents a drop in the drive's self-reported health as a consequence of SpinRite doing nothing more than asking for the drive's data back, just like computers always do. And so we see 1,137,031 uses of error correction. And SpinRite shows the minimum, the maximum, and the average error rate per megabyte of data recovered.

But most significantly is the drive itself screaming for help, saying, uh, ow, you know, you're asking me to read, and I'm having to work too hard to do so. So Javier, this is definitely SpinRite showing you, back up absolutely right now. I mean, maybe relegate this to somewhere, a noncritical application. But this is bad. And so I responded to his tweet saying, "No, those red squares show that the drive's own self-reported health is dropping under stress. That should never happen."

And lastly, Keith Pottratz sent me a tweet saying: "Ran SpinRite on two computers that were not running slow. SpinRite revealed that both drives were failing. Saved the day." So once again, a little preventive maintenance can go a long way.

Finally, getting to the topic, the nominal topic of today's podcast, as I said at the top of the show, two URLs: GoFCCYourself.com will redirect you to - oh, and by the way, that's HTTP. John Oliver and company at his "Last Week Tonight With John Oliver" did not do an HTTPS secure - they didn't get a certificate and so forth. So just put into your browser GoFCCYourself.com. That will redirect you to the FCC.gov secure site to a page which is difficult to find. So thus the reason they created this shortcut that redirects you to the page. If you click on the 17-108 there, over toward the left, you can see previous comments. If instead you click on the +Express, that's the express submission, where you can submit your own comment and express your feelings about how you feel about the pending potential loss of Net Neutrality.

The EFF, of course, stepped up with their own site. They have a page at DearFCC.org, and it's very nice. They wrote: "Dear FCC. The FCC has asked for public comment on new rules about Net Neutrality. Use this form to submit a comment to the FCC. Learn more about the FCC rulemaking process. A nice" - and it is - "fill-in-the-blanks form letter that the EFF's new site will email to the FCC for us." So I would recommend to our listeners...

Leo: I've already done it.

Steve: Do both, yes.

Leo: Yes, I've already done that.

Steve: A listener, Sable, he said: "I didn't catch which iOS FTP server app is your preferred one. Good enough to recommend?" And I would say yes, with the caveat that you not leave it serving all the time. As I mentioned when I talked about it, I turn it on, I pull a file from an iOS device, and then I turn it off because that's not something you want to leave on. It's just called FileApp, F-I-L-E-A-P-P. I have a link to it in the show notes. And for what it's worth, I do like it, and it's much more than just that.

They describe themselves as FileApp being "a file and documents manager for iPhone, iPad and iPod Touch." FileApp reads many files types such as PDF, Office documents, and plays multimedia contents; will let you store files and folders on your iOS device just like Windows Explorer or the Finder on the Mac. Does USB file transfer to Mac and PC using DiskAid or iTunes filesharing. Offers robust wireless file transfer which is username and password protected via WiFi and has both an HTTP and an FTP server. Stores files sent from any third-party app, mail, Safari, and so forth. Allows to "open in" any compatible app, pages, numbers, iBooks, and so forth. And lots of other features. So FileApp is the name of that little goodie. Also does file encryption using iOS data protection.

Adam Rixey asked: "Does the Internet SNI" - remember that's the Server Name Identification that shows which domain name we're going to - "make DNSCrypt fairly useless?" And I would say no because DNSCrypt is about much more. It's not just about privacy, it's about man-in-the-middle attacks and spoofing. Since DNS is not encrypted, unless you manually create an encrypted tunnel, which DNSCrypt does, anybody could intercept the response UDP packet and change the IP to theirs, and your server would dutifully go there and believe that's where it was, appropriately. So the SNI is a privacy leak. DNSCrypt is not only privacy, but also the integrity of all of your use of DNS between you and the DNS server at the other end that supports it.

And, finally, John Sey says: "What do you use for Windows 7 backup and recovery?" He says: "I'm sure I heard on Security Now!, but cannot find the reference." And I wanted to mention it to you also, Leo, because over the weekend you mentioned the previous one that I used to like.

Leo: Because I can never remember the one you like.

Steve: I know.

Leo: This is a generic name. It's Disk Imager.

Steve: Image for Windows.

Leo: Image for Windows. All right.

Steve: Yeah, so I used to like Drive...

Leo: I mentioned Drive Snapshot, yeah.

Steve: Yes, I used to like Drive Snapshot. But Image for Windows is HTFS-aware, or HFPS? I forgot what...

Leo: NTFS?

Steve: NT, NT, I started off with the wrong letter, yes, NTFS. Which Drive Snapshot wasn't. So Drive Snapshot, for example, especially nowadays where we've got - it only knew about FAT32. So you would have to break up an image into multiple, multi-gig files. Whereas Image for Windows is fully aware. It's bootable. They've got a bare metal version so you're able to, like, image from the hardware. Supports Linux and, like, all kinds of features. And it's like $38 for all flavors of all OSes. So Image for Windows, made by Terabyte.

Leo: Got to remember that. Image for Windows. Image for Windows; okay.

Steve: Those are my guys, Image for Windows.

Leo: Are we done? You just stopped talking.

Steve: It's 4:00 p.m., and a two-hour and 20-minute podcast.

Leo: There is no pressure to make it shorter. Just a couple of things. I did check, and the Apps Manager in Android, this is Samsung's version of it, but the Apps Manager in Android does let you do granular app permissions. So if you have an app that has access, let's say, to the microphone, you can go right to permissions, and they've got switches, and you can turn it off.

Steve: Nice.

Leo: And it presumably won't break the app unless for some reason it decides, you know, the developer said, well, without listening to what you're doing, we don't want to give you a free app.

Steve: And I know our listeners. Turn it off until it complains and then decide if you want it back on.

Leo: Yeah, yeah, exactly. So they do, you know, it asks for permissions upfront. But it's nice to have that granularity. And that's in the settings for Android under Apps. You can look at each app individually and control that. So that's a good solution. I also - there was something else I wanted to correct, but I can't remember what it was. But that's okay because we'll get emails and have to fix it next week. Wasn't you, it was me. I don't remember what it was.

We do this show every Tuesday, 1:30 Pacific, 4:30 Eastern. Sorry about that laptop lid in your face there. I'm updating Windows.

Steve: No, I'll just rest my chin on it.

Leo: I can open it right there for you. We do this, I'm sorry, 1:30 p.m. Pacific, 4:30 Eastern time, that's 20:30 UTC. You can watch live, and usually the streams are good. We had trouble with the YouTube stream today, but that's all right. We're restarting, as soon as the show is over, everything. And so you could find that at TWiT.tv/live or at YouTube.com/twit or Ustream.tv/twit or Twitch.tv/twit. We love it if you watch live. And if you do, you might as well be in the chatroom at irc.twit.tv. It's very helpful and fun and a nice place to be. But of course, if you can't be here, you can get on-demand versions of the show. Steve has audio plus transcripts. And somebody in the chatroom said, "Now, will Elaine understand that when we're talking about a NIC, that we're talking about an N-I-C, not an N-I-C-K?"

Steve: Oh, you bet she will.

Leo: Of course she will.

Steve: You bet she will.

Leo: That's why having a pro makes a big difference. So if you like to read along with the show, GRC.com. And while you're there would be a good time to erase your debt to society and buy a copy of SpinRite. That's there, as well as many free things, including SQRL.

Steve: Improve your karmic balance.

Leo: That's it. Forget the debt to society. Just what are you going to say when you meet Kerberos at the gates of Hades? "I stole SpinRite?" Oh, I don't think so. So that's where you do it, GRC.com. You can join us at our website, TWiT.tv. We've got audio and video of the show. But the best thing to do is subscribe because this is one of those shows you just want the complete set. Get all 611 episodes and enjoy them all. And the best way to do that is to subscribe. Find your favorite app and subscribe to the show. There's plenty of TWiT apps and podcast apps and Stitcher and Slacker and Pocket Casts and Overcast and all that.

Steve, as always, a pleasure. Thank you for all you do for all of us. And I will see you next week.

Steve: Okay, my friend. See you then.