SECURITY NOW!

Transcript of Episode #665

VPNFilter

Description: This week we discuss Oracle's planned end of serialization, Ghostery's GDPR faux pas, the emergence of a clever new banking trojan, Amazon Echo and the Case of the Fuzzy Match, more welcome movement from Mozilla, yet another steganographic hideout, an actual real-world appearance of HTTP Error 418 (I'm a Teapot!), the hype over Z-Wave's Z-Shave, and a deep dive into the half a million strong VPNFilter botnet.

High quality (64 kbps) mp3 audio file URL: http://media.GRC.com/sn/SN-665.mp3
Quarter size (16 kbps) mp3 audio file URL: http://media.GRC.com/sn/sn-665-lq.mp3

SHOW TEASE: It's time for Security Now!. Steve Gibson is here. Lots to talk about today, including how not to do a GDPR privacy email. More information about a bank hack called BackSwap that might be in your memory. And of course the deets, even some information not reported elsewhere, on VPNFilter. Why did the FBI tell us to reboot our routers this week? It's all coming up next on Security Now!.

Leo Laporte: This is Security Now! with Steve Gibson, Episode 665, recorded Tuesday, May 29th, 2018: VPNFilter.

It's time for Security Now!, the show where we talk about your security and privacy online with this guy right here, the king of security, Mr. Steve Gibson. Hello, Steve. Nice to see you.

Steve Gibson: Leo, great to be with you again for 665. One episode shy of the big 666. Which of course puts it exactly two thirds through the entire life of this podcast.

Leo: What?

Steve: Since we run out of digits at 999.

Leo: Yeah, 666, 999, very nice.

Steve: We're now two thirds of the way through. I think we'll see all the security problems are going to be resolved by the time we get to 999.

Leo: Are you waiting for that? Is that what's keeping you busy?

Steve: Yeah. When we hit 999, everything will go quiet. No more viruses, botnets, weird stuff, nothing.

Leo: Nothing.

Steve: It's just going to get all boring on the Internet.

Leo: And then we can go home. We can go home.

Steve: We're only two thirds - or stay home. We're only two thirds of the way through, and hell is breaking loose.

Leo: What? What? Tell me.

Steve: So of course we need to talk about the week's big news, VPNFilter, which is what the Talos security group of Cisco named - I guess this must be the largest botnet in history, more than 500,000 mostly routers, a few NAS devices, but more than half a million routers commandeered into a single botnet, which so freaked out the world that special dispensation was made available to deal with it. But we've got to talk about it because this is sort of a state-of-the-art, state-sponsored-level botnet which deservedly was in the news. And my best buddy a couple days ago said, "The FBI says I have to reboot my router." What? And it's like, so we'll explain about that, and why they said that, and why that isn't enough for our listeners. But that's at the end.

We've got lots of other stuff to talk about, of course. The really good news of Oracle's planned end of serialization in Java, which I'll explain. Ghostery's really kind of embarrassing - not even kind of - GDPR faux pas. The emergence of a clever new banking trojan which manages to avoid all the things that browsers and AV tools have over the years amassed in order to prevent what it does from doing. Amazon Echo and the Case of the Fuzzy Match. We also have more welcome news from Mozilla. Yet another stegano - okay, I can't do it.

Leo: Steganography.

Steve: Steganographic hideout.

Leo: Steganographic.

Steve: Steganographic hideout.

Leo: Just think stegosaurus.

Steve: A place for the steganographies to hide. An actual real-world appearance, Leo, of the HTTP 418 "I'm a Teapot" error.

Leo: No.

Steve: It actually happened last week.

Leo: No, no.

Steve: We also have the hype over Z-Wave's Z-Shave attack, which the press just went ballistic over, the idea of more than a hundred million IoT devices vulnerable. People can walk through your Z-Wave locked front door. Except not so much. And a deep dive into the half a million strong VPNFilter botnet. So I think lots of fun for our listeners this week.

Leo: Yeah, I really want to hear your story on that one. That's going to be very interesting. It's kind of a fascinating one.

Steve: Yeah.

Leo: Okay. On with the show, my friend.

Steve: Cool. So our Picture of the Week relates to the main topic of the week, VPNFilter. It's a great photo that demonstrates the complexity of the trojan that we'll be discussing.

Leo: This is how it works. This is that diagram, yeah.

Steve: Yes, yes. And, I mean, it does some cool stuff. I'll get into it in detail. But, for example, after the router is first infected, it reaches out via a URL for a photo at Photobucket, and it uses the GPS coordinates where that photo was presumably taken to get the IP address of what's known as the Stage 2 server, which is what provides the in-RAM component of the trojan. And then there's a fallback. If the Photobucket image for whatever reason doesn't provide it with an IP address, then it's able to do more. We'll talk about it at the end of the podcast.

Leo: Yeah, okay. Yeah, I won't ask you the details because I'm curious about this Photobucket thing. This is not steganography. It was simpler than that.

Steve: Yeah, exactly. All they did was they contrived the GPS coordinates where the picture was supposedly taken in the EXIF data tagged onto the photo.

Leo: Got it, got it.

Steve: So just sort of a way of it doing something that looks innocuous, but actually wasn't. So anyway, very cool.

We've talked actually a lot, one of the recurring memes of this podcast is the danger - danger, Will Robinson - the danger of interpreters.

Leo: Don't make me play the sound effect.

Steve: I was just going to say, Leo, I'm sorry...

CLIP: Danger, danger.

Leo: I have a button, you know. I can, any time...

Steve: I knew you had to be sorely tempted.

Leo: You said it. All right. I promise, no more, no more.

Steve: So interpreters are a recurring problem, I would argue a nightmare, when for example you can be attacked by a JPEG image. That's an interpreter failure because the JPEG is a structured image which some code reads in order to recreate, like to decompress the JPEG image, and clever hackers figure out a way to abuse that interpreter. Well, another one that's been in the news recently because of the Equifax breach and the problems with Apache Struts was Java. And we've talked about the so-called serialization/deserialization. It's a mechanism that was stuck into Java back in 1997, back in the early days, and it's very convenient in that it allows a Java object to be stored or sent over the network through the process known as serialization. That is, it takes a structured complex thing and turns it into a stream of bytes.

Well, at the other end you need to reconstruct the thing that you essentially deconstructed in order to stream it, and so that's the deserialization. And, unfortunately, that's an interpreter. And it turns out that somewhere between one third to maybe as many as one half of Java's historical security problems have arisen from this mechanism, from this serialization/deserialization process.

As a consequence, Mark Reinhold, who's the chief architect of the Java platform group at Oracle, labeled this whole serialization thing a, quote, "horrible mistake," unquote, in the Java language and has said it's being removed. They're essentially going to replace it with, at some future point in Java's life - and Java has turned out to be a very important implementation technology that's getting a lot of use in the enterprise because it is a write once, run anywhere technology because you have the so-called JVM, the Java Virtual Machine, which interprets the Java bytecode which the Java compiler turns Java language into, which then allows you to get lots of portability.

So, I mean, there's a place for it. We've run into it on this podcast for a long time. That's finally ended. It only ended when Java stopped being invokable by browsers. And that took, like, 10 years of disaster before finally it's like, okay, well, I guess we're going to unhook Java from our browsers, oh, boohoo. Nobody misses it from the browsers. It just never should have happened, but it did. The good news is it's going away. And with it will come an increase, sort of a belated increase in the Java systems security. Remember that...

Leo: I hate computer scientists because instead of just saying, oh, it's the ability to interpret bytecode coming from the 'Net, they call it "serialization."

Steve: Right.

Leo: Which just obfuscates its purpose.

Steve: Exactly.

Leo: Yeah. But computer science is full of these things.

Steve: Yes.

Leo: Okay. Thank you for explaining. I appreciate it.

Steve: And like reference docs. I'm often reading a lot of research. And you'll read the beginning of it, and it's like, oh, what? Sounds like it's a big deal. And then you get into it, and you realize they're just using big words.

Leo: Yes.

Steve: And it's like, oh, why didn't you just say you added two things and you got the sum?

Leo: Exactly.

Steve: Instead of, well, you know...

Leo: It's a closure. It's all about the lexical scoping of the...

Steve: Exactly.

Leo: Oh, please.

Steve: So the good news is it's going away. What they're going to do is they're going to believe a plugin framework so that JSON or XML could be used as application-specific simple serialization for specific needs. So rather than just having it always there and always vulnerable, if somebody needs it, they can just stick in a JSON or an XML plugin to perform that job for them. So yay. That's like, what is it, it only took, what, 20, 30 years?

Leo: Well, I think it's often the case that you write things to be as powerful as possible, and then you discover the malicious applications.

Steve: And, you know, I took a lot of heat 12 years ago when I innocently said that the Windows metafile format had an undocumented hook that allowed...

Leo: This is a serializer; isn't it. That's what it does.

Steve: Yes, it's an interpreter, exactly. And you would be stuck with what the interpreter did, except there was one bytecode that said execute what follows as native code. It was like, even Mark Russinovich said, yes, I've looked at it. That's what this does.

Leo: Yeah. So you could, theoretically, you could do the equivalent of another computer science term, sanitizing your inputs to make sure that it doesn't just execute random code off the 'Net.

Steve: Yes.

Leo: You could write a good serializer, in other words.

Steve: Well, and as we've discussed, it's so often the case that this stuff is hard to get working. So the moment you get it working, and you're behind schedule...

Leo: Yeah, it freezes. It's like, done. Ship.

Steve: Your boss is furious with you.

Leo: Ship it.

Steve: You haven't seen your friends or even your spouse for a month or two, and you're not having any fun. You're all hyped up on Coke or Mountain Dew or something. And then when it finally works, it's like, oh, thank god. And frankly, that's been sort of one of the advantages I've had with SQRL is that I didn't have someone making me ship this thing before it was ready. And I was even able to do this level of verification throughout it in order to make sure I got this thing cleaned up. So anyway...

Leo: Serialization isn't bad. But allowing code to come in over the 'Net and be executed is bad. And that's the thing. It's what you do with this serialization that's the problem.

Steve: Well, and you never want stuff to be slow. So if you put - or, for example, and you've coded enough. How many times have we seen coding examples where they said, for the sake of clarity, error checking has been left out of this.

Leo: Yeah. But imagine it's there. Right.

Steve: What they didn't say was please don't ship it as is. I mean, and that famous example of the Intel sample, or with UPnP, where here was like demo code, and it ended up just being dropped right into the routers. Nooooo. But that's what they did.

Leo: Yeah. The programming book I'm reading right now says, look, you're going to steal from yourself, even. You're going to copy code. You're going to because everybody does. So we're going to make it hard for you to do that because it's a bad idea.

Steve: Yeah. So in other bad ideas, we have Ghostery.

Leo: Now, you love Ghostery.

Steve: I do like Ghostery. Ghostery itself is not a bad idea. But they sort of screwed up last week.

Leo: Uh-oh.

Steve: They thought to celebrate all of their privacy wonderfulness...

Leo: Oh, I got this email.

Steve: Uh-huh. They were going to send out a notice to all of the registered Ghosteryiers, telling them about the GDPR compliance and how they're holding themselves. In fact, on May 25th they tweeted: "We at @Ghostery hold ourselves to a high standard when it comes to users' privacy."

Leo: Mm-hmm.

Steve: Uh-huh. You go.

Leo: Which is what Ghostery was written to do, by the way. It protects plugins on pages.

Steve: Yes, it's a privacy-enhancing web extension, web browser extension. So you're Ghostery, and you've recently also decided to take your previously outsourced email list management in-house in order to carefully manage the privacy of your users and take full responsibility for their care.

Leo: Sounds sensible.

Steve: Yeah, yeah. In the wake of the new GDPR regulations, you decide to send an update email to each of your registered users containing a privacy policy update.

Leo: Of course.

Steve: Unfortunately, the person who is put in charge of this task is apparently insufficiently familiar with the operation of the shiny new in-house email system. So what is sent out to users, against all intention I'm sure, is email containing, I kid you not, a series of emails, each containing in their "TO" header, stuffed to the brim, 500 email addresses of Ghostery's privately registered user email.

CLIP: Doh.

Leo: I had this one, too. I had to do it.

Steve: So not the BCC. Not the, I mean, it looks like you're the only one that received this, and you're just assuming everybody else did. No. You've got the email addresses of 499 other people who may be in alphabetical order, I didn't see it myself, who Ghostery also sent, in that particular block of 500, that single piece of email to. Yikes. So whoopsie. Of course, Twitter had some fun with this. We have Matt saying: "You cannot make this up. The privacy-driven Chrome extension @Ghostery for blocking third-party trackers has done its GDPR mass email exposing all of its users in the email 'TO' field."

Daniel says: "Hey @Ghostery. You sent your privacy policy update email without blind CC'ing the contacts. Now I'm on an email chain with a whole bunch of randoms, replying to your no-reply email address." Can you imagine? Reply all.

Leo: Do not reply all. Do not reply all, whatever you do.

Steve: And then Dan Barker tweets: "Weird move from @Ghostery. 1. Accidentally share thousands of email addresses with users in a GDPR email. 2. Apologize for doing so on Twitter. 3. Delete the apology tweet." Who knows. Anyway, ouch. As I say, stuff happens. And Ghostery stepped in it. So, yikes.

Okay. Now, this is cool. Well, worrisomely cool. The ESET Security guys have found kind of a diabolical new banking trojan. And Leo, you're going to get a kick out of this one. Okay. So first of all, banking trojans are still, like, at least a third of phishing email are banking trojans. In recent years they've been a little endangered because there have been other ways of getting money.

Once upon a time, before cryptocurrency, before it occurred to somebody that you could encrypt everyone's files and ransom them, all of these things generating cash, sort of the obvious "why do you rob the bank" question was, oh, let's put trojans in people's computers that watch them do banking and get their money. So years ago, the beginning of the podcast, people were - their checking accounts were emptied because they got some stuff on their family shared computer after Junior had his friends come home and stick a floppy drive - back then, remember those - in the computer, and off it went.

So we here haven't talked recently a lot about banking trojans, although their names are well known: Zeus, remember Zeus, also known as Zbot, which also begat Citadel, Atmos, and Floki Bot. Then there's Neverquest, Gozi, Dridex, Ramnit, Gozer, GozNym, Tinba, Gootkit, I mean, the list just goes on and on because that's where the money is. And so these things want to get into people's machines.

So in their article, in their description of this new, like there's been an upsurge in what ESET calls BankSwap, is their trojan. I'm sorry, BackSwap. BackSwap. They said: "Banking malware, also referred to as banker, has been decreasing in popularity among cybercrooks for a few years now, one of the reasons being that both antimalware companies and web browser developers are continuously widening the scope of their protection mechanisms" - this is key because this thing has gotten around this - "against banking trojan attacks. This results in conventional banking malware fraud becoming more complicated to pull off every day, resulting in malware authors shifting their time and resources into developing easier-to-make and more profitable types of malware like ransomware, cryptominers, cryptocurrency stealers and so forth." Again, those are the things we've been talking about recently because they've been a workable working source, and arguably easier to perpetrate against today's hardened AV and browsers.

So BackSwap has taken a new tack which is, again, is sort of diabolical. It passively monitors the user's computer activity, watching for banking activity. Then when it observes some, it uses two tricks to get its own interception code to execute. So first of all, this gets into people's computers, piggybacking on some legitimate-looking application. It's been seen in SQLMon, DbgView, the WinRAR uninstaller, 7Zip, OllyDbg, the FileZilla Server, among other apps. So apps are trojanized and then salted out on the Internet, where unwitting users say, oh, I want to uninstall WinRAR, and so they go get it in order to do it. So that allows it to set up shop in the person's machine.

What it then does is, now, traditional trojans have a whole bunch of problems. That is, they want to get into the browser, which is where online banking happens. Yet browsers might be either 32 bits or 64 bits. Well, if you're going to inject some code into the browser's process, you have to know which. And since you don't know which, you've got to have both. Also you need to be able to intercept the nonencrypted content, that is, before it gets encrypted, on the way out; and after it gets decrypted, on the way back, in order to get a shim, to get a hook in there.

Well, in the case of the Chromium-based browsers, it is those locations are buried deeply in the browser. They're not easy to find. And they are always changing. So they of course change from one browser make to another. Mozilla Firefox and Chromium and IE, none of them have the same architectures. So if you're going to do something that's generally going to be effective, you have to special case each of the browsers, each of the bitnesses, 32 or 64. And every time there's a new version, all of your work gets thrown out, and you have to do it again. So when you couple that with just the general difficulty of attacking the browser process with the heightened security...

CLIP: [Indiscernible].

Leo: I'm sorry. I'm sorry. That was not me. That wasn't me. It was somebody else.

Steve: Along with the heightened pressure against browser processes being interfered with and third-party AV, which is itself watching all this and trying to protect browsers. The bar has been really elevated. So BackSwap comes up with a whole new approach. It bypasses all of this by working with the Windows GUI elements and simulating user input. You can kind of think of it as malicious keystroke macros, which are running on the user's desktop, and which are therefore browser agnostic. Doesn't care what kind of browser you have.

What it does is it hooks into the Windows accessibility interface on the desktop so that it's able to watch what's going on. It scans the URLs coming and going. And if it sees a pattern match with a bank that it knows how to handle, that it recognizes, it's able to immediately essentially jump in and inject JavaScript into the user's browser.

How does it do that? It does it by simulating the user keystrokes, by typing them. In the older samples that ESET found, the malware originally inserted the malicious script onto the clipboard, and then simulated pressing the key combination for opening the developers console, Ctrl-Shift-J in Google Chrome, Ctrl-Shift-K in Firefox. Then it would issue a Ctrl-V, which is the Windows key combination for Paste, which pastes the contents of the clipboard into the console. Then it would press Enter to execute the JavaScript. It sends all of that so quickly that the user doesn't see it and is able to hide the window so that the user just sort of experiences a brief pause and doesn't know that anything happened.

In later versions of this malware they've upgraded the approach to take advantage of another feature in all browsers. It's like one of those things where it's like, okay, why do all browsers do this? You know how we have http://, https: and so forth. You could also have FTP and other things. And in fact I use SQRL. The SQRL client registers that as a means for the browser to talk to the client in the system. So that's called the URL scheme.

Well, browsers are also aware of the JavaScript scheme, J-A-V-A-S-C-R-I-P-T colon, and you can then put JavaScript after that which the browser will execute. You just stick it in the URL. So, not surprisingly, this BackSwap banking trojan is able to type J-A-V-A-S-C-R-I-P-T colon into the URL. It first does a Ctrl-L, which is the universal "give the URL the focus." It then types J-A-V-A-S-C-R-I-P-T colon. And what's interesting is it doesn't paste that because browsers won't let you paste that. There's an actual prohibition against pasting a URL containing JavaScript colon anything because they said, oh, that would be a vehicle to attack. Well, yes.

It turns out that, however, they cannot discriminate between the malware typing it in manually and the user typing it in manually because, in terms of the Windows message loop, which is the way all this is done, it looks the same. And it's just like a keystroke macro that would be entering it. So it types J-A-V-A-S-C-R-I-P-T, then enters the long JavaScript blob, basically the entire JavaScript injection, into the URL, hits Enter, and then blanks the URL as if nothing wrong had happened, and is able thereby to inject its own JavaScript into the user's browser. And this has now jumped onto the scene and is back in the game, essentially browser agnostic, doesn't care about bitness, performs no in-process injection. AV doesn't know how to block it, and browsers aren't blocking it. And this thing has just taken off and is doing a gangbuster business as a banking trojan.

So we see where there's a will, there's a way. And some very clever people behind BackSwap have figured out a means of getting past all of our AV. So the takeaway for our listeners is don't get yourself infected with this thing, basically.

Leo: Okay. Thanks for the tip. I'll do my best.

Steve: If you do, you're probably in trouble. I would argue that, because banking trojans are still a thing - it's easy to get phished. It's easy to get yourself hooked by one. With computers being as inexpensive as they are now, and if you're someone who does a lot of online banking, if there's a possibility that you've got a lot of money sitting in an account, and you're doing funds transfers, take an older laptop or an older machine and just stick Debian Linux on it. It costs nothing. It's free. It comes with Firefox installed. And just don't use it for anything else. Let that be the machine where all you do is your banking. It boots pretty quickly. I think it's a great solution.

Leo: Don't use email on it, though, because you could in theory still be phished.

Steve: That's true. Exactly.

Leo: Just banking.

Steve: Really you only want to just - just banking. So that requires some self-control, but probably worthwhile.

Leo: I know a lot of people now are using Chromebooks for banking. And I think that's equally secure.

Steve: I agree. That's another great solution.

Leo: Cheap. You can buy a couple hundred dollar Chromebook and just keep it for the stuff you need to be secure.

Steve: Yup. It's interesting, too. As I was reading this, I was thinking, oh, well, that's nice. One of the things that I also did in the SQRL client, as our listeners will all be seeing soon, is I use a secondary darkened Windows desktop, much like the UAC. The UAC in

Windows is much more than just a visual effect. It's actually a second desktop which is invoked which is secure. And so that prevents keystroke loggers and keystroke inserters and so forth from getting literally their hooks into it.

And it's funny because I initially implemented this because I wanted the SQRL authentication prompt not to be spoofable. That is, I didn't want a web page to be able to pop up a dialogue that looked exactly like the prompt for logging into SQRL because a lot of users would go, oh. Like when they were logging in with SQRL, they would expect that. And so I needed to do something that a browser could not spoof. And switching to a secure desktop, I monochrome it and darken it in order to make it look very distinctive because no code in a browser is able to mess with the exterior of the browser. So in the process, a side effect of that is that keystroke loggers that might be monitoring what you're doing get disconnected. So it's just another benefit of sort of the belt-and-suspenders approach.

Okay. So Amazon Echo and the Case of the Fuzzy Match. This initially seems scary, but Amazon explains it. And I would argue, within the bounds of what we want, they were not at fault. KIRO, which is the Channel 7 ABC affiliate in Seattle, carried a story last week about a misbehaving Amazon Echo device. A family in Portland contacted Amazon to investigate after they said a private conversation in their home was recorded by Amazon's Echo, and that the recorded audio was sent to the phone of a random person in Seattle who happened to be on the family's contact list.

The wife of the family, Danielle, who did not want her last name revealed, said in the KIRO reporting of this: "My husband and I would joke and say, 'I bet these devices were listening to what we're saying.'" And they were big Echo users. Every room in their home was wired with Amazon devices to control heat, lighting, and security. But Danielle said two weeks ago their love for the Echo changed - and I would argue, well, we'll explain how this happened in a second - with an alarming phone call. The person at the other end of the call said, "Unplug your Echo devices right now. You're being hacked." That was not true, but you could understand why they might think so. That person was one of her husband's employees calling from nearby Seattle.

So she says: "We unplugged all of them, and he proceeded to tell us that he'd received audio files of recordings from inside our house." She said: "At first my husband was like, 'No, you didn't.'" And the recipient of the message then explained that "You sat there talking about hardwood floors." And so they realized, oh, yes, we were just talking about hardwood floors. You did hear us.

So she listened to the conversation, which was then sent back to her, and in this reporting said that she couldn't believe that someone nearly 200 miles away had also heard their conversation. Yeah, well, welcome to the world of technology.

Okay. So they got a hold of Amazon. Took them a while to get to an engineer who finally said that the Amazon engineers went through the logs, and they saw exactly what had been described, and apologized. She said that the engineer apologized, like, 15 times in a matter of half an hour, said that they really appreciated it being brought to their attention, and that that was something that they needed to fix. Although Danielle said that the engineer did not provide any specifics about why it happened and, for example, whether it's a widespread issue.

Okay. So KIRO, using the strength of their being a network affiliate, did get through to Amazon and got some answers. They spoke to someone, Shelby Lichliter at Amazon, who sent them the following statement: "The Echo woke due to a word in the background conversation," sounding like the trigger word, which I won't repeat, we all know. "Then the subsequent conversation was heard as a 'send message' request. At which point [the Echo] said out loud, 'To whom?' At which point the background conversation was interpreted as a name in the customer's contact list. [The device] then asked out loud [repeated the contact name] and said, 'Right?' [And the Echo] then interpreted the background conversation as, 'Right.' As unlikely as this string of events is," said the spokeswoman from Amazon, "that's what occurred." And, she said: "We are evaluating options to make this case even less likely."

So as I said, Amazon Echo and the Case of the Fuzzy Match. This is inherent in a voice-controlled system where you want to do everything by voice, and you are doing fuzzy matching. And it's to my mind entirely feasible and reasonable that there could have been a conversation in a home where every room has one of these devices. And if you don't have at some point something that says for a high-security action - and this isn't particularly high security. If you want to voice-enable sending something that you say to somebody on your contact list, and that's an interactive interchange with confirmation checkpoints by the device, and the confirmations are also audio, and at no point do you have to touch it or push a button or do anything else, there's going to be some percentage of time in the universe of these devices always on, always available, always listening, like wanting to reliably catch their trigger phrase and not frustrate their users. It's a balancing act.

And so it's entirely reasonable, if you want this much convenience, that you're going to, I mean, these people were freaked out. And it's like, well, yes. But you have Echo-enabled your whole home, and you're somehow missing the confirmation audio prompts, and that can happen.

So this got picked up by the press. It's like, oh, my god, another Amazon Echo spying thing. And it's like, no. We talk about it on this podcast. One of our other memes is the security versus convenience tradeoff. And it's hugely convenient to be able to use your voice to order things to happen, but you might get an unasked for package of toilet paper at your front door from time to time.

Leo: I mean, it's a highly unlikely coincidence of things. And I think they could do some things to improve it. For one thing, it sends those messages off without confirmation. And Siri and everybody else will say, "Oh, is this the message you wish to send?" and ask for confirmation. And I think that that's a simple thing Amazon will almost certainly do to the Echo to prevent these kinds of things. But, yeah, we've all heard the Echo respond to TV, or Google Home.

Steve: Yes. It suddenly lights up. Especially loud television. It's like, looking around, saying, what? What? Me?

Leo: You talking to me? I have a Home, a Siri-based HomePod, a Google Home, and an Amazon Echo in my kitchen. And not a day goes by without them interacting. And we just get used to it. It's like you hear her mumbling in the corner, we go, oh, yeah, whatever. It happened this morning, and I didn't even know what she was talking about. It's just like your crazy aunt over there in the corner.

Steve: Remember, I think it was on "The Jetsons," it was Rosie the Robot?

Leo: Yeah, Rosie the Robot, yeah.

Steve: And so she's just sort of...

Leo: Mrmrmrmr, under her breath, mrmrmrmr.

Steve: Yeah, just sort of, like, rolling around dusting stuff. And it's like, okay, Rosie.

Leo: Rrrrrrr.

Steve: Yeah.

Leo: So anyway.

Steve: Yeah, anyway. I just sort of wanted to say that, yes, it doesn't have to be hacked. It doesn't have to be malware. It just can be voice-control where some percentage, I mean, there's like there's a non-zero probability that the device could interpret, I mean, any length. Yes, as the interaction is more back and forth, the probability of it happening is going to go down. But if you're not at any point requiring the person to press a button to confirm something, to like physically take an action, if it's all 100% voice, then there's a percentage of misfire that will happen. And so you have to sort of have that, you know, understand the tradeoff of convenience versus security.

Leo: And when Echo talks to you, listen, because she's trying to do something. See what she's trying to do.

Steve: Yes. Bad Echo. Bad Echo.

Leo: Bad Echo, no.

Steve: So we're currently at Firefox 60, and it remains my go-to browser, mostly just because of tabs. I just - I have all the tabs I want, and I want a lot of tabs. And Chrome refuses to deal with that. So, fine. Until they come up with a way of giving me more tabs, I'm sticking with Firefox. And the good news is Firefox is keeping up. With 63, and we're at six zero now, so three versions from now when that hits mainstream, they're adding a number of new features. And I still like my term "mal-mining," which to me is just much clearer than "cryptojacking." We've got so many crypto this and crypto that, that it just doesn't stand out. Cryptojacking doesn't feel like cryptocurrency mining hijacking, which is what it is. Mal-mining, yes. Anyway, the good news is Firefox 63 can block that.

Okay. So what they've done is they're sort of overloading this category known as tracking protection. They're throwing all kinds of other stuff in there. They probably ought to just stop calling it "tracking protection" because, if it's also blocking crypto-mining and fingerprinting, well, I guess fingerprinting is tracking, but certainly crypto-mining is not.

Okay. So just to back up a little bit. Remember that right now Firefox private browsing mode has tracking protection on by default, but normal browsing doesn't. Yet it's now recommended for people who want increased privacy and, significantly, performance, that you go under Tools > Options > Privacy & Security, and under Tracking Protection switch it to Always. What you get then, even when you're not using private browsing, is significantly, noticeably, higher performance because, unfortunately, pages are just loading so much crap from third-party servers where they've got to do DNS lookups and go get the stuff and download it, that it really does slow down our pages. So just saying no, thank you, don't want that stuff, it makes your pages run a lot snappier.

What is being added under tracking protection is granular control over analytics, advertising, fingerprinting, crypto-mining, and social media stuff. And for each of those five categories you can say you always want it, you only want it in private windows, or you never want it. So they're breaking it apart, giving you granular control and, happily, going to somehow, probably using some browser heuristics, look at a tab which is apparently doing mining. I mean, maybe they'll track miners on the fly, or they'll just do it behavior-based. But you'll be able to say no, thank you, and not do that.

Oh, and significantly, there are also per-site exceptions. So if there's a site, for example, say that crypto-mining to support the site you're visiting actually catches on someday. Well, the site might say, you know, you're blocking our ads, and you're blocking our crypto-mining. We'd like you to support us. So either turn on ads or turn on mining so while you're here we can mine some cryptocurrency in the background. So if you decide that seems reasonable, you'll be able to do that. Also, they're unburying this somewhat. You'll be able to get to it from that menu in the upper right-hand corner of Firefox, so quicker and easier access to tracking protection.

In the future, also with 63, they will be adding the ability to wash the content that a site gives you, cookies and other content loaded into your browser on behalf of that site in the little lock icon. So you'll also be able to clean that up. And they're adding one-time password support, Authy-style one-time password, time-based one-time passwords, to the Mozilla account preferences. And so that'll just be appearing in coming weeks. So Firefox is still in the game, and I'm glad.

I mentioned the tongue-twister "steganographic," steganography and so forth. Steganography we've talked about from time to time, and I've sort of said, eh, it doesn't excite me that much. It's the act of hiding in plain sight. And so in our digital era where we've got crazy resolution of things, it's just not that difficult to do. And I guess that's why it doesn't get me all worked up. For example, if you've got a beautiful photo, where you've got 32-bit color depth, you've got high-resolution representation of each of R, G, and B color. Our eye cannot see the least significant bits. That is, if a color which ranges from zero to 255, if it's 167, if one pixel is 168 or 166, there's like plus or minus one on either side, we can't see it. But it would be possible to take the entire image's least significant bits and hide a message there.

So you've essentially built a message into an image where the bandwidth is relatively low. That is, you have a large image and a message which is a fraction of the image size. But it doesn't require any particular trickery. So in a sense you've hidden it in the image. And only if someone knew where to go to get it would they be able to find it. It's a little bit like we were talking about, the VPNFilter bot, which hides the IP address of the server in the GPS coordinates of a photo, where the photo was supposed to have been taken. You know, that kind of thing.

Anyway, some researchers at Columbia came up with yet another place to hide data, and that is in the shapes, the detailed distortions of the shapes of font characters, characters in a font. I avoided the word "glyph," but that's the technical term. "Font glyphs" are these individual character instances. And, boy, I've got the links in the show notes for anyone who's interested. The science that they applied to inserting and distracting data from slight perturbations in font glyph shape is impressive.

As I've said, I'm sort of not that impressed in general with steganography. It's like, yeah, okay, I guess that's kind of cool. I would argue these guys took it all the way where - and they give an example of some text printed on a poster where they had encoded a relatively low-bandwidth message into tiny, I mean, non-visualizable distortions, so that you could stare at it, and you don't see any difference. But their technology, given 200 pixels per inch resolution, is able to, from a photo of this chunk of text printed on a poster, is able to recreate the original image that was encoded into tiny distortions in the text. So, I don't know, I wanted to put it on everyone's radar, just as another place that you could hide data. And if you're curious about the details, it's an amazing piece of research. But it's like, yeah, okay.

Oh, it also occurred to me when I was thinking about this, where else could you hide information? You could, in our digital era, you could use tiny distortions in the frequency of notes in a song. Just detune notes a little bit sharp or a little bit flat, or just skew their frequency a little bit off beat. I mean, there are just - there are so many ways that you could distort something such that, if you knew to look for it, you would know that that was non-random, and then you could extract a message. But it's sort of, I don't know, it's like it's so possible to do, it's so easy that it's - except that these guys really applied some serious technology and math to it. Other than that, eh. Still, it was very impressive.

Leo: Wait a minute. Say that again so I can record it and put it on one of my buttons. Eh.

Steve: Other than that, eh.

Leo: Eh.

Steve: Okay. So we've actually had, Leo, a real-world occurrence of HTTP Error 418 I'm a Teapot.

Leo: How can I do that? Can I make it happen here?

Steve: To remind our listeners, it was April 1st, not surprisingly, of 1998 when RFC - and Leo, you probably want to look this up, RFC 2324. So it's an official IETF.org document, RFC 2324. It is the HTCPCP protocol, which is the Hyper Text Coffee Pot Control Protocol. And we're all familiar with the infamous HTTP 404. That's the page not found. You don't normally see the 300s, like a 301 or a 302, because those are the redirects. And you don't see them because your browser normally does them for you. You click on a link to content that is moved somewhere else, and what you get back is a 301 redirect which then your browser sees and goes, oh, that content is over at this URL, and you go there instead.

Leo: This introduces the brew method. You've got posts, of course. But this is brew.

Steve: Yes. And in fact the RFC begins under rationale and scope. It explains very seriously there is coffee all over the world. Increasingly, in a world in which computing is ubiquitous, the computists want to make coffee.

Leo: There's also a "when" method. So when you're adding milk you can say "when." Oh, lord.

Steve: It says: "Coffee brewing is an art. But the distributed intelligence of the web-connected world transcends art. Thus there is a strong, dark, rich requirement for a protocol designed espresso-ly..."

Leo: Espresso-ly.

Steve: "...espresso-ly for the brewing of coffee. Coffee is brewed using coffee pots. Networked coffee pots require a control protocol, if they are to be controlled, naturally. So the Hyper Text Coffee Pot Control Protocol," which you have been exploring, Leo.

Leo: Yes. Beautifully written.

Steve: In the RFC 2324 dated April 1st, 1998.

Leo: Uh-huh.

Steve: So what happened? For seven hours yesterday many developers using NPM - I'm not kidding - the Node.js Package Manager...

Leo: Yes, I use it all the time, yes.

Steve: ...for JavaScript were receiving an error message: "ERR! 418 I'm a Teapot." The trouble was seen by those teams operating behind a proxy that was appending the port specifier ":443" to the domain name on its way across the proxy out onto the public Internet.

Leo: Well, that's SSL, isn't it? That's - yeah.

Steve: Yeah. But it was confusing the NPM registry's servers, which resulted in them returning an "ERR! 418 I'm a Teapot" declaration.

CLIP: Doh.

Steve: Yup. Somewhere, someone decided to return that error for unforeseen conditions.

Leo: I love it.

Steve: And it actually did happen in the real world.

Leo: So great.

Steve: So, yes, "I'm a Teapot" lives, well past 1998.

Leo: So awesome.

Steve: Okay. So this is one of those where the press is hyperventilating. And I meant to grab the headline from Forbes because I saw that even Forbes picked it up with a breathless, oh my god, 100 million IoT devices potentially exposed. And I'm sure you'll be talking about it tomorrow on This Week in Google with Stacey, since she's the IoT goddess of the network. So this is the Z-Shave attack against Z-Wave IoT devices. Z-Wave is the protocol, the technology which appears to be winning. It has about a hundred-meter range, so it's much greater range than Bluetooth, which is one of its advantages. And they finally got security right. The bad news is they didn't get it right in the beginning. And so it is true, I mean, what was found by these guys at Pen Test Partners is true. But it's not as bad as people would think.

The headlines were: "Z-Shave attack could impact 100 million IoT devices." "Z-Wave downgrade attack left over 100 million IoT devices open to attackers." "Z-Shave lets you open Z-wave based locks."

Leo: That would be bad if you were using it to lock your front door.

Steve: That would be bad. And so here's how I would characterize this. Yes, a determined attacker who was also patient and who installed a battery-powered eavesdropping thing might be able to obtain the household's secret encryption key, after which it, yes, would have the ability to unlock the doors. So it's not good. I mean, it's not perfect.

Leo: How long is the key? Is it six digits?

Steve: No. So here's the background. When we did talk about all of this some time ago, so if any of our listeners don't remember us doing a podcast on Z-Wave, because we did earlier on because of the compromise in its security, and I didn't have a chance to - didn't occur to me to go back and get the Security Now! number, the podcast where we delved into this in great detail [SN-560]. But here's the deal. You can have with Z-Wave no encryption security. And, okay, that's bad.

Leo: Yes, yes, I'll grant you that, yes.

Steve: That's bad. You do not want that unless, I mean, maybe you've got an old hub that doesn't support encryption. Well, okay. Don't hook up a door lock or a security system to an unencrypted Z-Wave. That's bad. Or you could have S0 level encryption, which is not bad, but not perfect. Or you can have S2 level encryption, which is perfect. I mean, and I don't use that word loosely. No one's ever heard me say that. I mean, it's really, really, really, really, really good. So, okay. If either S0 or S2 security is in place in the environment, in the home, then an encrypted key is known to all Z-Wave devices, and all communication is securely encrypted under that secret key.

So the Z-Wave network, the way it works is it's got a secret. And pairing a new device is the process of informing that device of the household's secret. S0 encryption used an all-zeroes initial key, and that was its weakness. If it used all zeroes, then a passive eavesdropper who could capture any pairing of a Z-Wave device could capture the traffic, know that they were using an all-zeroes key to communicate the actual, that is, to encrypt the actual home's secret, and then decrypt it themselves and be on the network.

Okay. So that's the weakness of S0. It is encrypted, but the pairing event creates a vulnerability that is significant. As a consequence, S0 has been completely deprecated. In fact, the Silicon Labs that are the owners of Z-Wave have loudly stated that, as of a year ago, no newly certified Z-Wave devices can have anything but S2. Yet some 48-plus devices, as I recall, did receive certification with S0 after that deadline. So, whoops.

The problem is, and we understand this, we've seen this over and over again, it is very difficult to move something that is working from "it's good enough" to "it's really good security." Okay, but now here's the problem. S2 exists. Everyone knows how it works. It uses Elliptic Curve Diffie-Hellman key exchange, thus perfect, as far as we know. Diffie-Hellman is fabulous. Notwithstanding the fact that that's what SQRL uses to do its crypto in several places. The idea is that with S2 Z-Wave security, each of the devices in the pairing situation, rather than this transient use of an all-zeroes key to encrypt the actual key - which really is, you know, why bother? If anyone's going to listen, then they're going to be able to get in under S0. After that I guess the network is encrypted, so that's certainly good.

With S2, each of the two devices to be paired creates a nonce, a random chunk of nonsense, and sends it to the other. The magic of Diffie-Hellman key agreement means that each end can obtain a secret in common which even somebody listening in and seeing that exchange occur, capturing all the traffic, cannot figure out what key it is that they each have. And that's what's so cool about this.

So under S2 security, they each come up with a random number. They share a version of that with each other. And that allows them to independently arrive at a shared secret which the hub then uses to encrypt the household key, which it sends to the device being brought onto the network, since it has that ephemeral key that it got through Diffie-Hellman key exchange. It uses that to decrypt, and now it's on the network. Perfect security. I mean, just it doesn't get any better than that.

Now here's where the attack comes in. Turns out there is, because of the legacy, even, well, first of all, obviously both ends have to support S2. If either end is only S0-aware, then you have what we've seen before, a protocol security downgrade. That is, the more secure is forced to drop down to the level of the lesser secure in order to reach a protocol they both understand. So the problem is that S0 devices are still predominant. And in fact S0 hubs are still the majority. Even Samsung's very popular Z-Wave Hub is still S0 security, despite the fact that S2 is what we want to be using. So again, this is another thing that we see over and over is, even after security is figured out and made strong enough, everybody's got to play.

Okay. Well, the fact that you have to support the security of the lesser end, that opens you to a real-time downgrade through active interference. And that's what the guys at Pen Test Partners did that got them some headlines. And I would argue okay, yes, it's true. If, at the moment of two S2 devices pairing, you shot out a little bit of active noise, now you can't be passive anymore. You've got to actively interfere with the communications in RF spectrum between the devices in order to intercept and force them to believe that each other is S0. So it's high speed. They couldn't even pull it off without generating some custom tools that could respond quickly enough. So it's sort of even still theoretical, although they were able to demonstrate it in sort of one class of this attack.

So again, it's a bit of a tempest in a teapot. It's just not a huge problem. The vulnerability exists briefly. It requires a very sophisticated attacker to be interfering with the radio communications during the brief interval of interchange between the two devices. So I don't think there's anything to worry about here. It's unfortunate that Z-Wave ever offered no security or S0 because it's going to just take a while for everything to get up to S2, and all the S0 devices to finally leave the ecosystem. Eventually they will. And then we've got a good system with very robust security.

Okay. And one last final bit. I got a very nice note from the Netherlands from someone who just said his name was Wouter, W-O-U-T-E-R. He gave me permission to pronounce it Walter, but he also told me how to pronounce it. So it's Wouter. He says: "Hi, Steve and Leo. Thanks for talking about the difference between broken hard drive sectors and failing hard drives. If the described problems and errors do not indicate a failing hard drive, what does?" That's actually a really good point.

And he says: "I once read a general recommendation to replace spinning hard drives every three years, but that sounds pretty costly. I do have some older hard drives for use in Time Machine backups. Should I worry? Some general information on, A, how to spot failing hard drives; and, B, when to replace a hard drive would be greatly appreciated. Regards, Wouter."

And we've talked about this. We've skated around this a number of times. As I mentioned recently, it is quite possible for sectors to be failing in the normal course of a drive's use. That's why drives have spare sectors is because they may be needed. They're expected to be needed. And the drive won't start complaining until it starts running out of spare sectors. So it expects that defects will arise over time, and they're called "grown defects." That's a term of art in hard drives, "grown defects." Defects grow. So the sectors that acquire those defects need to be taken out of service and spares put in. So one way that a drive could start getting in trouble, if it notices that its spares pool is drying up. That's not good. That will be reflected in the SMART data as a drive health problem. But you need to know to look at this SMART data.

The other really more sensitive way to tell is the dynamic rate of error correction. That is, and this is something that really doesn't get enough attention, and it's one of the coolest things about what SpinRite is able to do. If a drive is just sitting there doing nothing, that is, not working at a given workload, then it's not producing. It's not needing to do correction. And so none of the metrics that SMART shows have any meaning. But if you put it under load, as SpinRite does, then suddenly the rate at which errors are having to be corrected is a beautifully sensitive analog meter of the health of the drive.

If you, when you get the drive new, run SpinRite on it and look at the SMART data, one of the screens in SpinRite is a SMART data monitor screen. It will show you the rate of error corrections in corrections per megabyte of data transfer, data read. And you can make a note of that. And from time to time, when you run the drive, take a look at that and see that it sort of remains constant. If at some point you see a spike in that, well before the drive has collapsed on you, has failed, that rate will increase. And that's the key to say, okay, that's an actual valid indication that the drive is having to work too hard in order to recover data. It probably means that it's consuming its spares pool much more quickly, and you'll be in trouble soon.

So there is an indication. If you just google "SpinRite SMART," S-M-A-R-T, the first link that Google shows is a link to the page at GRC that takes that SMART data monitor screen apart and shows you what all the different meters and metrics and things are. It's kind of cool. Take a look at it. Just google "SpinRite SMART," and you'll see what I mean.

Leo: Nice.

Steve: And Leo, our last break, and then we're going to dive into VPNFilter, a really interesting event in the industry this past week.

Leo: I wanted to show you something I thought was really cool. I know you'll kind of get it. It's kind of a video, though, so I apologize for people listening. Earlier today there was a BGP misrouting. I'm sure you read about it. Cloudflare...

Steve: Border Gateway Protocol, yeah.

Leo: Yeah, Quad One's was accidentally hijacked by a Chinese ISP. Quickly fixed, so not a big deal. But Cloudflare has, I think, the coolest thing ever, which is an animated replay of the event. And you can watch the routers, the BGP routers get hijacked and then reassemble. It's really actually, I mean, you would understand what's going on here better than I do. But it's just kind of fun to watch as the routers go, wait a minute, what happened here, and then figure it all out. It's really neat, yeah.

Steve: Right.

Leo: Cloudflare's replay is just...

Steve: Yeah. The idea, of course, is that routers advertise the routes that they're responsible for. And quite non-maliciously, I mean, without any malicious intent, and it does happen from time to time, a router can declare that it has a much shorter path to a network than it actually...

Leo: A shortcut.

Steve: Exactly, a shortcut.

Leo: A shortcut.

Steve: ...than it actually does. And so it advertises this short route, and all the routers it's connected to go, oh, cool. And they send their traffic there instead of going the long way, which unfortunately was the right way.

Leo: Yeah. It was fixed later in the day. But it was not even for that long. And I don't think it was malicious. I think they just were using 1.1.1.0/24 as a test route, which then Hurricane Electric propagated to the rest of the world.

Steve: Yeah.

Leo: Thank you. Thank you, Hurricane. But it's fun. This is so cool. I never knew they had this tool. So to watch these at work is really neat, really neat. Little JavaScript animation there.

Steve: So unfortunately, so is this botnet. So last week we had an evolving story regarding concerns surrounding a massive botnet of more than 500,000 routers. Yes, more than half a million routers commandeered into a single botnet. For anyone who's interested, the Talos Intelligence Group of Cisco did beautiful coverage of this. I've got the link in the show notes. And this is one of those situations where the answer to the question "What could it do?" is a chilling "Anything it wants." I mean, if you've got half a million routers connected to the Internet, that's bad, I mean, when they're under control of a malicious entity. Actually any entity because, I mean, you could even make a mistake like the Border Gateway Protocol routing mistake.

Leo: Yeah, yeah.

Steve: Forensics evidence strongly implicated Russia as the botnet's creator, and it's believed to be intended as a network for attacking Ukraine. Cisco found more than 500,000 instances of this VPNFilter malware on routers manufactured by - and this is what's important for our listeners because there's a takeaway here. Some of these are not old junkie routers that no one has. I ran across a note, for example, there was a D-Link something or other, DL-620 router that had a backdoor that was discovered, except that it's really, really old, and they don't support it anymore, and five people have them. So it's like, okay, I'm not even going to bother talking about that.

These routers are mainstream routers. So Linksys, MikroTik, Netgear, TP-Link, and also some QNAP NAS devices. The Linksys E1200 and the E2500, as well as the Linksys WRVS4400N. Three routers from MicroTik. Did we decide it was MicroTik or MicroTik?

Leo: Tik. MicroTik. MicroTik sounds like necrotic. That's not a good...

Steve: Exactly, no. Three MicroTik routers, the 1016, 1036, and 1072. A bunch of Netgear routers: the DGN2200, the R6400, the R7000, the R8000, the WNR1000, and the WNR2000. And then QNAP devices: the TS251, the TS439 Pro, and other QNAP NAS devices running the QTS software. And, finally, the TP-Link R600VPN. If you have any of those routers, you need to pay attention because those routers are the routers that were hosting more than half a million instances of this VPNFilter malware. It is a highly sophisticated complex botnet with a bunch of moving parts and a plugin architecture. We talked about earlier this month the Hide & Seek botnet as the first botnet that had been seen to survive reboots. Well, this is the second one.

So VPNFilter can survive a reboot of the router because it's able to install sort of a little stub of itself, which Cisco calls Stage One, into the router's flash memory. So there are now two pieces of malware, Hide & Seek that we talked about earlier this month and VPNFilter, able to statically infiltrate and remain in a router. So the takeaway is what has been recommended is certainly that you reboot your device. I'll explain what that does and doesn't do in a minute. But it's only by performing a factory reset back to the original device settings that we're told that flushes the malware out of the device. I'm not sure why that's true.

So I think checking for any firmware updates and using this as a good time, a very good time to update your router firmware makes a lot of sense. So if you're on that list of routers, see if there's an update. And I would reflash my router, if you're concerned, with the latest firmware, even if it's the one you've already got, if your router will allow it to be reflashed with the same firmware. It might say, oh, I've already got the latest one. Go away. Don't know.

Leo: So rewriting the firmware eliminates the stub that's reloading the malware.

Steve: Yes. Yes.

Leo: Given that they've got the command-and-control servers, I guess reloading the malware's not going to happen anyway; right?

Steve: Yeah, actually it does.

Leo: Oh, it does.

Steve: Yeah. It's a little more complicated.

Leo: So the FBI's advice is wrong.

Steve: It's not complete. Okay. So here's the deal. Cisco breaks this thing down into three stages: Stage One, Stage Two, Stage Three. Stage One we've talked about. That's the thing that goes persistent by modifying the flash and getting itself rerunning whenever you start the router. When that starts up, what we were talking about at the top of the podcast occurs. It first tries to go out to the photo sharing site and grab a photo from a URL and then know that the GPS pseudo coordinates are actually the IP address of the command-and-control server. If that fails for any reason, then its backup plan is to do the same thing, pull a photo from a well-known domain. And that domain is T-O-K-N-O-W-A-L-L dotcom, ToKnowAll.com.

So from either of those instances, the Stage One is able to obtain a larger RAM-resident Stage Two. And this is clever because that means that you're making - the footprint in flash is very small, which means it's going to be able to find a home in many more routers. If you had to load the whole blob of malware into Flash for persistence, there's much more chance that the file system that might not have much empty space wouldn't be able to accommodate it. So these guys said, aha. We're going to create a tiny little stub, and then we'll get the rest on the fly. That part lives in RAM. Okay. So that part does something else. And, well, okay. So Stage Two sets up a plugin architecture which Stage Three are the plugins for. But the other thing that didn't get mentioned by the FBI is that Stage One, the resident portion, also opens a listening port.

Leo: Oh.

Steve: Yes. And if the botnet authors had obtained a list of IP addresses that had been contacting the command-and-control server, then they're able to spray all of those IP addresses with new command-and-control stuff and maybe take it back. So it seemed, as you think about this, this thing is so sophisticated and so nicely designed that you would expect there to be some means of recovering command-and-control once what's there has been reverse-engineered and the command-and-control servers have been taken down. And in fact that recovery system exists. So that's why - okay.

So the reason the FBI wants people to reboot is twofold. First of all, it is true that after having taken down the main command-and-control servers, what happened was later last week they got a court order from a judge that allowed them to compel VeriSign, who I'm sure was quite happy to, but VeriSign said please compel us, to hand over domain control for the ToKnowAll.com domain. The FBI pointed that domain to their own servers. And then the botnet, that is, this half a million plus strong botnet, began pinging their server.

So now that allows the FBI to themselves obtain the IP addresses of all of these compromised routers. And since they're not going to do anything nefarious with it, if users now reboot, a couple things happen. In the reboot process, the Stage One does its outreach to these domains. In the process of doing that, the FBI will get an updated fresh list of all the IPs of the routers which are compromised. And since Stage One will not be able to obtain the Stage Two malware from either the photo sharing site or ToKnowAll.com, it doesn't go any further. But the listening port is still open.

Now, the other thing I didn't talk about that Stage Two does, aside from creating a plugin framework, which allows some powerful plugins to be installed, and this is one of the things that really upset people, there is flash wiping capability in the Stage Two payload. And this is what had people worried. If this was meaning to be more than just a DDoS attack, which is a bandwidth flood, as we know, against targets in Ukraine, imagine if IP addresses throughout Ukraine were identified and all of the routers, all of the consumer small home, small office, small business routers were permanently wiped.

This Stage Two payload overwrites flash and then reboots the router, bricking it. I mean, and permanently bricking it. At that point you don't have firmware in order to reload the firmware. It's over. So the big concern was that this thing could have, I mean, like while these people were in control - and we may be talking about this next week because, as I said, there is a means for them to recover control, unless the FBI was prepared to outreach into those routers and do something themselves. But, unfortunately, that's not legal. You're not able to go and modify anybody else's hardware, even if it's for a well-intended benign purpose, like disinfecting them. At least in the U.S., the law on that is very clear.

So the Stage Three plugins are able to sniff the local network traffic to detect the presence of the SCADA, S-C-A-D-A, industrial control network traffic, and also further communicate with the botnet's command-and-control servers through the Tor network. All the communications is through Tor or TLS-encrypted connections. So this thing is, you know, this is a seriously powerful, not a benign botnet. The big concern was that it might have been, I mean, imagine if more than half a million routers self-destructed.

Presumably, if it really is Russia - and it looks to me like the attribution is very solid on this. Cisco's Talos group wrote: "We assess with high confidence that this malware is used to create an expansive, hard-to-attribute infrastructure that can be used to serve multiple operational needs of the threat actor. Since the affected devices are legitimately owned by businesses or individuals, malicious activity conducted from infected devices could be mistakenly attributed to those who were actually victims of the actor. The capabilities built into the various stages and plugins of the malware are extremely versatile and would enable the actor to take advantage of devices in multiple ways."

The Ukrainian Secret Service believed an attack was planned to take place on Saturday when the Ukrainian capital of Kiev was to be hosting the UEFA Champions League soccer final. The FBI confirmed that the botnet was created and was under control of the famous Russian state-sponsored cyberespionage unit known under many names. We've referred to them on the podcast as APT28 and also Fancy Bear. They're also known by Sednit, Pawn Storm, Sofacy, Grizzly Steppe - actually we've talked about them under those terms - Strontium, Tsar Team, and others. And the Estonian Foreign Intelligence Service identified APT28 as a unit of Russia military's Main Intelligence Directorate, the GRU.

So the command-and-control domain is under control of the FBI. Everybody has been asked to reboot. If you own one of those routers, do that at minimum, but also I would reflash. Given that you are able to, certainly our listeners can. If you have one of those routers, get the latest firmware available for it and reflash it while you can. Oh, and Cisco did say that it did not appear to them that any zero-day vulnerabilities were responsible. They thought it was publicly known old vulnerabilities that had not been patched. Which suggests that the affected routers do need updated firmware, while after you flash remember disable all WAN-side network management. If you can, turn off Universal Plug & Play on your LAN side, as well.

And by all means use a non-default strong username and password to set this thing up. Every week now we are talking about how these little appliance routers, which are workhorses - it's funny, too, because I was thinking about how Stage Two and Three live in RAM. On the other hand, our routers are probably the least often rebooted devices in our entire network. Everything else is being plugged in, unplugged, moved around, turned on, turned off. Those little routers sit there running for years at a time. So it's really not a handicap that this thing, most of it lives in RAM. That's not a problem.

Leo: I think that's how the other ones, Mirai bot and all those, are also resident. They probably don't have a way to write enough code in the firmware to keep it persistent.

Steve: Right, right. It takes doing more. And they're quite happy to live in RAM because their devices are unlikely to get rebooted anytime soon.

Leo: Well, and even if they do, they just reload.

Steve: Yup, exactly.

Leo: So you put a couple of lines of code in the firmware that reloads it.

Steve: Yup.

Leo: So if you didn't have an affected router - do you think that list is complete? Or is it conceivable other routers are affected, and they just don't have the full list?

Steve: The problem is we have a sophisticated actor, in Cisco's parlance, a threat actor...

Leo: It's a state actor, yeah.

Steve: Yeah, a state actor that wants to install their stuff everywhere.

Leo: Right.

Steve: And there are routers we know have been vulnerable to other things that we've been talking about recently that are not on that list. So it's certainly reasonable that in the future other routers are going to get infected with this. So, I mean, more than half a million routers. I would say there's never been a better time to make sure you're running the latest firmware.

Leo: Yeah. Reboot anyway, everybody, and then firmware update. And that's one reason we always argue for IoT devices that automatically update, that have firmware patches.

Steve: Oh, yes, yes, yes, yes.

Leo: Because it's a pain to reflash your router.

Steve: Yes.

Leo: And I bet you a lot of routers don't let you - I know my ASUS could reflash. You could take...

Steve: With the same version?

Leo: You could roll back if you wanted to, yeah. But a lot of the consumer-grade routers are made to be as simple as possible. And they just say, oh, no, you're up to date, and that's it. In which case I don't know what you'd do.

Steve: Yeah.

Leo: Get a better router. This is an opportunity to get a better router. Steve, good show, as always. Very informative. Every week we do this about 1:30 Pacific.

Steve: That's what we do here.

Leo: 4:30 Eastern, 20:30 UTC if you want to watch on Tuesdays at TWiT.tv/live. You can also join us in the studio if you email tickets@twit.tv. Be my studio, not Steve's. Steve, as far as I know, does not have a live studio audience. Where would you sit?

Steve: Yeah, no.

Leo: Nowhere to put them. But we do love having the studio audience up here in Northern California, if you come by. You can also get copies of this show on demand at your convenience. Of course go to GRC.com, that's Steve's website. That's where you'll find the audio copies and the transcripts, as well, if you like to read along, or use the transcripts to search for parts of all of the 665 shows. You can also get other things at GRC.com, including SpinRite, the world's best hard drive recovery and maintenance utility, a must-have for all hard drive owners, GRC.com. Everything else there is free. I used, when I was suffering from jet lag, just couldn't get my schedule back, I used your Healthy Sleep Formula. Worked wonders. Worked like a charm.

Steve: Cool. Cool.

Leo: Because the problem with jet lag, at least when you're going west to east, is you wake up in the middle of the night. And so the ability to sleep through the night was critical to overcoming my Japanese jet lag.

Steve: Nice.

Leo: Thank you, Steve.

Steve: Well, next week is 666.

Leo: Yes.

Steve: So hold onto your seats, folks.

Leo: The podcast of the beast coming up. We will see you all then. Thank you, Steve.

Steve: Thanks, buddy.