SECURITY NOW!

Transcript of Episode #544

Listener Feedback #228

Description: Leo and I discuss the week's major security events and discuss questions and comments from listeners of previous episodes. We tie up loose ends, explore a wide range of topics that are too small to fill their own episode, clarify any confusion from previous installments, and present real world application notes for any of the security technologies and issues we have previously discussed.

High quality (64 kbps) mp3 audio file URL: http://media.GRC.com/sn/SN-544.mp3
Quarter size (16 kbps) mp3 audio file URL: http://media.GRC.com/sn/sn-544-lq.mp3

SHOW TEASE: It's time for Security Now!. Steve Gibson is here. We're going to talk about the latest security news. Here's the good news. There's not a whole lot. That means things were quiet. But we will talk about that CrashSafari website, and we will also answer 10 of your questions. So it's Q&A day on Security Now!, next.

Leo Laporte: This is Security Now! with Steve Gibson, Episode 544, recorded Tuesday, January 26, 2016: Your questions, Steve's answers, #288.

It's time for Security Now!, the show where we protect your security, now, with Mr. Steven Gibson, the man in charge of the Gibson Research Corporation, creator of SpinRite, world's best hard drive maintenance utility. And here he is in the flesh. Well, in the Skype, anyway. Hey, Steven.

Steve Gibson: In the Skype. And Leo, great to be with you again, as always, as we plow into an interesting 2016.

Leo: Yes, yes.

Steve: We talked about LostPass extensively last week, so it's Q&A time, time for our Listener Feedback Potpourri. So we're going to do that. It was a rather light news week, not really much to talk about. I've got a few things that are sort of interesting. And I've got much more detail on the Safari crash exploit that I thought you'd get a kick out of from the 22-year-old San Francisco software engineer who thought it would be fun to do this.

Leo: You saw his demo. It really is kind of amazing that there's no control...

Steve: And I'll explain why.

Leo: Yeah, okay.

Steve: I know exactly why, so...

Leo: Good, yeah.

Steve: Yeah, we'll do that. And but we have some really, really useful questions, where we'll spend a little more time than usual answering some, because there was feedback, some questions about the nature of second-factor authentication and why it could not withstand man-in-the-middle attacks. Someone saying, hey, here's a product that does. It's like, uh, no, it doesn't, and explaining, like, it's funny because this is a particularly difficult problem to solve. It's one that, despite all of SQRL's strengths, that we've spent a lot of time focusing on because it's sort of intrinsic to the way the 'Net works. So we'll let our listeners' Q&A drive some of this conversation, and also catch everybody up on what little news occurred. I don't know, it was a sleepy week for everybody.

Leo: Really. Well, that's good. I like it. You know, in this business a sleepy week is a good week; right? You don't want...

Steve: No news is good news.

Leo: Yeah. I don't know if you noticed, but I screwed up our website this morning, and that's what I was...

Steve: I heard you mention that on MacBreak.

Leo: We have a very fancy, you know, our website was a quarter of a million dollars. It was very fancy. And one of the reasons we did that is we wanted to do something called "Headless Drupal." So we have Drupal...

Steve: So you have a database backend with an API.

Leo: You got it. And then the site itself, when you go to our site, is not running Drupal, it's running Node.js, which is a very modern, state-of-the-art, beautiful language. It gives you great site. And then we use caching. There's caching all along the way. But one of the cache services we use is called - is based on Redis, R-E-D-I-S, Redis Labs. And it's one of those little minor details. It costs us 40 bucks a month.

It's inexpensive. And I just missed the bill. It was tied to a credit card which expired, and I just missed it. So...

Steve: Ahh. And so they turned you off.

Leo: They had been sending me emails every month saying, hey, [stammering]. But I get a lot of email, engineering-style email, and I just have a filter, and I didn't see it.

Steve: It was very much like Cogent, who was telling me for several months, we're out of the T1 business. We're going to be turning off your T1s.

Leo: Right.

Steve: And it's like, well, I didn't get that email.

Leo: There's a high level of noise in email, and I just - I get a lot of email. And I don't pay bills. I mean, everybody else is, you know, we have accountants, we have people to do that. But what I didn't know is - I just thought, well, everybody's getting this bill, so I don't need to pay it. I don't have to pay attention. But apparently I was the only - this is the one service we use - and we use a lot of them for this website, it's like six different things running - this is the one service we use that no one else was getting that email.

Steve: And I would say you got it back up pretty quickly.

Leo: Well, I feel bad because I - we started at 7:00 in the morning. I'm calling, and I'm trying to get a hold of them. And I guess they're a small company, and nobody's answering the phone. And then, because I didn't have an account, I couldn't log in to use their support. The support line was hidden, and the support - so I'm emailing them, I'm calling them and leaving - finally I tweeted. Which is the nuclear option. I got a response right away. And I wasn't - I don't think I was mean in the tweet, and I completely admit it was my fault. It wasn't their fault. But I hate to have to tweet somebody. But, boy, when you need customer service, sometimes that works. Anyway, we're fine now.

Steve: You mean because it's so public.

Leo: Yeah. It's a public complaint. And they suddenly woke up and said, "What do you, uh, what, huh? What do you need? We'll fix it." And they fixed it within a half an hour.

Steve: So the day after our last podcast, last Wednesday, I just wanted to note that this issue is, as I said, is going to be with us for a couple years, until we resolve it. And that's another bill has been put into a state assembly, in this case California, surprisingly. Jim Cooper, who is a Democratic assemblyman from - I wrote it down, I don't remember where now, somewhere in California, the city didn't ring a bell - has put into the California Assembly a bill that basically does the same thing, basically saying that, I'm sorry, the same thing as the New York bill that we talked about last week, which attempts to limit the sale of smartphones that don't provide some means for allowing law enforcement to get access. His angle is different, though. His is human trafficking.

And so Ars picked up on the story and said that: "A second state lawmaker has now introduced a bill that would prohibit the sale of smartphones with unbreakable encryption. Except this time, despite very similar language to a pending New York bill" - and by the way, Jim says they didn't base their legislation on the New York legislation at all, his staff arrived at this independently.

And Ars writes: "The stated rationale is to fight human trafficking, rather than terrorism." And then Ars wrote: "Specifically, California Democratic Assembly member Jim Cooper's new bill, introduced last Wednesday, would 'require a smartphone that is manufactured on or after January 1, 2017, and sold in California, to be capable of being decrypted and unlocked by its manufacturer or its operating system provider.'"

Jim then had a phone interview with Ars where he, Jim, who's a 30-year veteran of Sacramento County Sheriff's Department, said: "If you're a bad guy, law enforcement can get a search warrant for your bank, for your house, you can get a search warrant for just about anything. For the industry to say it's privacy, it really doesn't hold any water." We're still quoting Jim. "We're going after human traffickers and people who are doing bad and evil things. Human trafficking trumps privacy, no ifs, ands, or buts about it."

Leo: No, no, no, no, nothing should trump privacy for innocent people. And that's the issue; right?

Steve: Yeah. Exactly. Exactly. And...

Leo: Although this is kind of what you were saying, which is let's go back to the pre-default encryption days, right, so that Apple could do it.

Steve: Well, and I thought about this some more. And I've tightened my suggestion even further, remembering the way things used to be. And that is to have it require both physical access and Apple themselves, or the manufacturer themselves. So you can't even transmit like an unlocking key for a specific phone electronically. It's the way it was, which no one really had a problem with.

Now, again, taking something away from people that they have is never easy. But as we covered on the podcast several times, law enforcement used to physically send a phone that they had acquired from a subject and a court order saying, yes, this needs to be unlocked, to Apple. And what law enforcement was complaining about was the multi-month delay because Apple had some laborious process they went through. Basically they had a way of brute-forcing the relatively short passcode out of a smartphone and thus unlocking it under court order and then discovering the code, probably writing it down, and returning it with the phone to those authorities.

So an updated version of that would be some solution that still requires physical access and isn't algorithmic, that is, requires something random per phone that only Apple knows, so that there's no algorithm that can escape. I mean, it's very much more like a traditional cipher, where the algorithm is known, but the key is not. And even the key cannot be algorithmically derived. It has to just be randomly chosen, and Apple would maintain a database.

But again, my sense is the reason I talked about it before when I saw something, I remember I shared a few months back, and I took the controversial position of saying, you know, this mimics the traditional American constitutional right to privacy and against unwarranted search and seizure. But under court order those rights, exactly as we've talked about, you can get your bank account records, you can get your phone records, you can have authorities enter your premises, which are otherwise your sovereign territory, under particular conditions.

So to me, this echoes that. And I wouldn't be surprised if we end up there. I hope we end up with something no weaker than that because we certainly, you know, I agree with everyone that we don't want a "backdoor." This, however, this seems to be the way we're seeing some legislation sort of proposed. So we'll see what happens.

Leo: So do you think it's a really bad idea? Or...

Steve: I don't think it's a bad idea. I think it's a reasonable compromise. And the pure privacy people don't want any compromise. They want the phone to be unique in our lives in that no level of authority can cause its inspection. But that's not - we don't have anything else like that. So I don't know why that makes sense to have as an exception when law enforcement can demonstrate reasonable suspicion and to a judge who says, okay, I can understand you have reasonable suspicion that this phone contains evidence material to a case. I will grant you an order to compel its manufacturer to open it for you.

Leo: Yeah. And this is the process that goes on with, you know, if I have a trunk...

Steve: Everything else, yes.

Leo: ...in my studio over here that's locked...

Steve: Yes.

Leo: ...the police, without a search warrant, can't come in here. They have to go to a judge and say, "We want to know what's in that trunk."

Steve: And the judge has to...

Leo: And the judge has to approve it.

Steve: And they have to also demonstrate some...

Leo: A reason.

Steve: Yes. It just can't be a fishing expedition, where it's like, well, we don't know what's in here, but we want to look. It's like, no, that's not okay. You have to demonstrate why it is you believe there's, you know, you have a reasonable suspicion...

Leo: That's a longstanding process.

Steve: Yes.

Leo: It's not like that's a new capability or new right. That's been around since forever.

Steve: That's the way, in America, that's the tradeoff that we have...

Leo: And we can contend with that.

Steve: Yes, yes. And so I think physical access is important because by that I mean that Apple would have to have it, or an Apple satellite facility or something. So that prevents it from being just remotely electronic. But it duplicates, exactly as you said, Leo, it duplicates the existing tradeoff between privacy, yet an ability for law enforcement to do their jobs. And we want them to do their jobs. We want terrorists caught. We want all of the cliches, the child pornographers and the human traffickers and all these people caught. We want their phones opened, but not our phones.

And so there's no way to do it except to put a human judge in that loop, require a judge to make that judgment, and then design the system so that it is not prone to abuse. That is, it's not an algorithm that the hackers can get. It's not a backdoor that law enforcement, once they have it, they can open anyone's phone. Every single phone would require physical access and a randomly derived key that was set into it initially and stored in a secure vault at Apple is accessed in order to get access to the phone. And that duplicates what we have in the United States.

Leo: The real problem is we've all been - we all feel burned; right?

Steve: Yes.

Leo: We all feel like, wow, we can't really trust these guys. For instance, I think you make a good, a very important point. They have to have physical access. I don't know if that bill that you just cited says that.

Steve: No, and this is the problem is - this is why I'm talking about it. I actually got quoted on the podcast on a really prominent blog, I think it was The Daily Beast, a couple weeks ago, did pick up on this nuance and also said, you know, I'm in a different position than all other security people. It's like, well, yeah. I'm not taking an absolutist position. Everyone can. But my worry is that, if we don't compromise, there's the chance much worse legislation could happen, I mean, really, really dumb legislation. This to me feels like it echoes a set of tradeoffs that we've been living with. And again, not easy to take anything away from someone. Right now our iPhones are absolutely encrypted, and so no one wants to give that up. It's like, well...

Leo: And hasn't Apple, hasn't Tim Cook been adamant about saying we're not taking a step back on this? This is a vital right? I mean, he - I wonder how much of this is because Apple really sees this as their commercial distinguisher from any other, from any competitor. They're the ones that are the privacy protectors.

Steve: Well, I think there's that. And there's also, I mean, what I just described does place responsibility with Apple. On the other hand, how much money did you say they...

Leo: Yeah. They don't like that, do they.

Steve: How much money did you say they just paid?

Leo: Yeah. Holy cow. We just got the earnings report is starting to come in from their first quarter. What did I - I forgot. They made...

Steve: Like 83 billion or something.

Leo: Seventy-five billion for the quarter in revenue, that's $25 billion in revenue, and the profit was something like $6 billion a month. $25 billion in revenue, $6 billion a month in profit. Like here's $6 billion. You get to keep that.

Steve: Yeah.

Leo: And since their tax rate is 25%, you get to keep a lot of that. That is just amazing.

Steve: Yeah. So I guess my point is that Apple can...

Leo: They can afford this.

Steve: Apple has the profit and the money to create the biggest, thickest, deepest vault ever known by man. I mean, Apple knows encryption. They could protect this. They would rather not have the responsibility. I think, I wouldn't be surprised if the way this turns out, the manufacturers of these consumer devices are going to end up with the responsibility. And remember that there will still be third-party solutions for people who want to add that to the existing device. But I do think it makes sense to turn the clock back a few years with updated technology.

Leo: We certainly don't blame them for not wanting to get in the middle here.

Steve: Yeah. Speaking of smartphones, the Dutch Consumers Association, the DCA, has sued Samsung. And this was foreseeable. We've almost predicted it on the podcast for, like, ever since StageFright, when I started saying, you know, phone manufacturers are going to have to stop believing that they can just wash their hands of responsibility for security vulnerabilities in what are essentially pocket computers after they have sold them.

So last year's StageFright vulnerabilities, which we covered, which our listeners will remember, were a little frightening, well, very frightening because the exploitation was just the reception by an Android phone of a multimedia text message which was processed, if it was deliberately malformed in the right way, processed in a way that allowed the bad guys access to your phone. They were able to get in and take over.

So what we're coming to understand is that, as we know, smartphones really are more computers than phones. And as such they will be rampant with vulnerabilities that will only be discovered over time. No one wants that to be true. But all of our experience with all of these computer technologies continues to show us that these problems are - new ones are being created at at least the rate that the old ones are being found and removed.

So as we often say here, the podcast has long legs. Plenty of future here. And these things will be filled with private data, so users are going to want to keep those secure, exactly as we were saying. But what this means, the fact that smartphones are computers more than they are phones, they're going to have vulnerabilities, and they're going to be filled with private data means that they must have the same sort of update mechanism that we now take for granted in our desktop operating systems and our browsers. All of the desktop operating systems and browsers are now, I mean, this whole idea is crazy of not keeping your patches up to date on these devices that are exposed to the Internet because problems are being found constantly.

So according to this group, the Dutch Consumer Association, their research showed that at least 82% of Samsung smartphones available in the Dutch market, which they examined, had not received any software updates on the latest Android version in two years. So this failure to provide software updates left the majority of Android devices, 82%, vulnerable to issues of security and others which have been found. So the DCA says that the agency has previously contacted Samsung many times and discussed the matter privately with the manufacturer to resolve the situation, but was never able to reach an agreement with the company, so it's decided to go to court.

And also the Ars story, I think it was Ars that I was quoting, said that the more recent high-end Samsung Galaxy S6 series may have received StageFright patches, but Samsung has not provided StageFright fixes for its majority of midrange and entry-level Android devices. And we've talked about this in the past. And none of Samsung's devices currently runs the latest Android 6 Marshmallow - or as I call it, Mushroom - even now, three months after it officially launched. So the DCA wants Samsung to - the DCA is demanding in this suit that Samsung updates all of its smartphone devices to the latest version of Android operating system for two years from the handset's purchase, not its launch.

Leo: That's fair enough.

Steve: Although it is a big ask because, I mean, and there could be technical problems that would require more work from Samsung to, I mean, exactly like we were talking about with the Skylake chips and Windows 10, where some newer version of Android might have features that would require extensive work on Samsung's part in order to bring up on their older hardware. But again, I agree with you, Leo. It's like, okay, two years from purchase, wouldn't consumers want to know that, for two years, they're going to have the benefit of any major advances?

The DCA also want Samsung to treat software updates as part of the warranty that has its length mandated in the EU for two years. So when you buy something like this in the European Union, it comes with a two-year mandatory minimum warranty. And so the DCA is saying software updates ought to be part of that. And then in a quote they said: "We're demanding that Samsung provides its customers with clear and unambiguous information about this. Also, we're demanding that Samsung actually provides its smartphones with updates."

So apparently there has been dialogue where Samsung said, oh, yeah, yeah, we'll look at this, and we'll think about it, and blah blah blah. But, like, none of this ever went anywhere. And so finally the Dutch authorities have just said, okay, enough of this. This needs to get fixed. So as we predicted on the podcast, because these are computers, something has to change. It's no longer acceptable for them to be considered turnkey, sell-it-and-forget-it computers. They're connected computers. They need to be kept current.

Google has been quietly working on a next-generation compression for the web. Compression, we've talked about many times in the last 10-plus years, is very important because web pages, the body of web pages, is text, which is highly compressible. And images are also highly compressible. And so the idea being, then, that the web server has the uncompressed original text and, either once or per request, is able to compress it to a much smaller size so that it is able to move to the browser in a fraction of, I mean, like 10% of the time it otherwise would. And then the browser decompresses that back to its original size for display.

And of course the famous compression is Gzip, which is based on a Lempel-Ziv - those are two researchers back at Bell Labs in 1977. If memory serves, they got a patent on what was called Lempel-Ziv, or LZ, compression. And we did a podcast on it for anyone who's interested [SN-205]. Really fascinating, the way it works. The idea is that you retain at the sending end, the compressing end, a buffer of everything you have sent. And as new stuff comes along, you check to see if any of it or how much of it might already be in this recent history buffer. And so you match as much of what's about to be sent as you can, performing a longest string match. And instead of sending that, you send a pointer to where it is in your buffer. And so all that goes across the wire is a pointer.

Well, the receiving end is mimicking the sending end. It keeps a history of everything that it has received. So basically this is a clever way of the sender and the receiver maintaining duplicate buffers, even though they don't ever share it, and it's grown incrementally over time. And the buffer evolution, the way the buffers age is also synchronized, just by agreement. So when a pointer comes in, the recipient uses the pointer into its own buffer in order to expand that pointer back into some text that it knows the recipient just saw.

And so that's the way LZ works. And in my model I described it as a communication system, where you have a receiver and a sender, but they can also just be static processes. You can have a compressor that you run something through that produces this smaller thing. And then that file exists, and we have compressed image files where the uncompressed image, like the raw image from a camera, oh, my god, we know how many megabytes those things are now, and more all the time. But so the camera compresses it into, for example, a JPEG or a PNG or whatever, down into a much smaller size using various types of algorithms. JPEGs work very differently than what I've just described because they're a so-called "lossy" compression, as opposed to a "lossless" compression, that gets back, in the case of a lossless compression, exactly what you put in.

So over the years there have been continual tweaks to this basic concept that I just described, all kinds of fancy things you can do, and we have talked about various things. Huffman encoding of the tokens, which is a frequency-based encoding so that, rather than everything being the same size, if you're able to look at an alphabet of how many times different things occur, you assign shorter tokens to the things that occur most often. That was actually part of the key to Morse code is that Samuel Morse made the things that occur more often shorter so that, I mean, because it just makes sense to do that.

So Google's been working on something called, let's see, the original name was Zopfli. It's like, oh, please don't let's call this Zopfli for, like, all time. Because at least Bzip is kind of fun. Zopfli, I mean, there's a "P" in there, Z-O-P-F-L-I. And I'll explain why in a minute. But anyway, the good news is it's not Zopfli, but it's not better. It's Brotli, B-R-O-T-L-I. Which, you know, really sounds like a vegetable that I have an intense dislike for. So anyway, it's good for you, high fiber, you should eat your Brotli. Anyway, so this has moved to what Google calls internally their "intent to ship" status. So what this is, what Brotli is, is further tweakings on the Zopfli compression, moving it forward further. Zopfli was Deflate compatible, meaning that - and Deflate is one of the terms, there's Deflate and Gzip that have been the longstanding compressors on the Internet.

And so, for example, when a browser says in its request, it has a header that says Accept-Encoding. It'll say "deflate common gzip." And what that tells the server is that the browser knows how to decompress any content that is encoded using those algorithms. So that gives the server permission to compress what it's going to send using whatever encodings the browser has said it understands. So what Brotli adds is a new encoding, "br." So where we have deflate and gzip, we now add br for Brotli. It is not Deflate compatible. What "Deflate compatible" meant was that what Zopfli was doing...

Leo: Oh, my goodness.

Steve: I know. What Zopfli - this is better than the Honey Monkeys. What Zopfli was doing was cleverer compression, doing a better job of analyzing, for example, more of the data, larger buffers of history, and more clever assignment of tokens such that the result was smaller, but it could be processed by the same decompressor. Thus it was Deflate compatible. Brotli makes up whole new rules. And it is serious, I mean, for people who have a fascination with compression, it's become a standard. Google has put it in the public domain, made it widely available. Now, it's secretly been in use by all of our browsers because the next generation of fonts, the so-called WOFF2, which is the Web Open Font Format v2, they've all been using Brotli, Brotli with parts of...

Leo: How dare they?

Steve: Brotli was part of it, of WOFF2. So the good news is not much more will be required for our browsers to support it. What it gives us, what it gives us over Zopfli, is...

Leo: You're making this up, aren't you.

Steve: ...20% higher compression ratios. And in Google's comparisons, they were able to - there's a really neat chart in the PDF. There's a Gstatic.com link in the show notes here to Brotli. You might want to bring it up, Leo. There's a neat chart that shows speed versus compression ratio. And this has always been a longstanding challenge because you could sometimes compress things better, but at a huge expense in performance, which you don't want because - yep, there's the chart. And you can see where Brotli is, way up at the top. And it's made some compromise in performance. It's not as high as the very, very best. But it is very fast and better than a lot of the older compressors that are lesser in performance...

[http://www.gstatic.com/b/Brotlidocs/Brotli-2015-09-22.pdf]

Leo: And lesser in compression.

Steve: ...but way lesser in ratio, yes. Okay, so here's where Zopfli and Brotli came from. Google writes: "As with Zopfli, the new algorithm is named after Swiss bakery products."

Leo: Oh, no.

Steve: So Brotli, oh, and this has an umlaut over the "o."

Leo: Brotli, that's Brotli.

Steve: Means small bread in Swiss German.

Leo: Great.

Steve: So there we go. Br will be the encoding. It's able to outperform Gzip. What we're all using now pretty much is Gzip.

Leo: Bzip's really kind of come on strong, I think.

Steve: Yes, but it never made it into the web. So, like, Bzip things...

Leo: Oh, I see what you're saying, yeah, yeah, yeah.

Steve: ...are like, you know, static files are Bzipped. And it's a great, especially Bzip2, it's very good compression. But its performance is such that it didn't justify the cost. Brotli, however, has made it. So Brotli outperforms Gzip, which is what we have now, for typical web assets by between 17 and 25%. So, for example, over Gzip, it can give you 17% additional savings of when it just compresses JavaScript on the Alexa Top 10,000 Sites. So that's worth doing. Same thing for minified JavaScript, 17% savings. And it can squeeze a CSS, already compressed with Gzip, another 20%. So again, Google is trying to make the web faster. Precompressing the data and squeezing it before it goes over the bandwidth to the Internet, through the Internet to us where possible, makes a lot of sense. So Brotli is coming soon.

Oh, and let's see, it is, as I mentioned, it's been in the WOFF2 format for a while, so it's not expected to have any compatibility issues because it's already been implemented. Developer interest is high, so it looks like we're going to see adoption pretty quickly. This WOFF2 format, which already means there is a Brotli decompressor present, is supported in Chrome, Opera, and Firefox. Support for this next open font format is in Safari, and this next format is under consideration for Edge. And the Edge team has indicated an interest. So far no word from Safari. But certainly, if this becomes a standard, it'll happen.

So what it'll mean is it'll be a transparent migration. Browsers that start being able to support it, not only for fonts, but explicitly for all content, will simply add that "br" to their accept-encoding request header. And then to the degree that servers start supporting it, servers will see that and go, oh, and compress with Brotli rather than with Gzip when they send stuff over the wire. So just more moving forward.

Now, we did have a little bit of backward moving, however, with CrashSafari.com. Do not go there.

Leo: Whoa.

Steve: Because it's not just CrashSafari. There's actually, what is it, I think it's three domains. There's CrashSafari, CrashChrome, and I think there was a third one. I can't remember what it was. But...

Leo: CrashSafari worked on Chrome, I might add.

Steve: Oh, and it works on Firefox, too. I did. I did. Because, you know, like Jerry Pournelle used to say, "I do these crazy things so you don't have to."

Leo: Yeah. I did it on the air, and that was crazy.

Steve: I watched, yeah. Okay. So what this is, and you'll get a kick out of this, Leo, because there is the JavaScript, the four lines - only because I have line breaks, it could easily be one line - of JavaScript, showing what this does.

Leo: Wow, that's not much.

Steve: No, it's not. And what's really funny is, well, sort of, is this has been known since 2014. This is not even new. This was a 22-year-old application security developer named Matt Bryant, working in San Francisco, who sort of stumbled over this and thought, oh, this will be kind of cool. So he got the CrashSafari and CrashChrome dotcom domains and just put this little piece of JavaScript in the page. That's all it is.

Okay. So here's the problem. It's the HTML5 spec two years ago got the so-called "History API." The History API is the URL history that your browser maintains, technically per frame, but we see it as per tab. So when you hit the back button, back back back back back, or when you hold it down in some browsers and then it opens up a list of all of the sites you've come from, that's the history that that tab has. What HTML5 did was give programmers programmatic access to it. So, for example, a page could push a different URL into the history queue deliberately, so that when you hit the back button, you go back to where it wants you to go to.

Now, this is useful for, for example, Gmail, that wants to be more of an application than a web page, so that when you click on something in Gmail, you will go to, for example, a message. But when you use your back button, you don't leave Gmail and go back to Facebook or wherever you were. Instead, you go back to the panel or the pane, the view of links to different articles. So you can see that a web-based app needs to override sort of the global function of the back button. So that's what this does.

Here's the problem. It turns out it is computationally expensive to update that history. That is, just the nature of the way it's done means it's far faster to push something new into the history queue than it is to update the queue. And the browsers, because the browsers don't want you to have to wait, that is, they don't want the UI to hold up while they update this history, they decouple them. So the call to the API returns instantly. And then the work to do the job of pushing the history works in the background. So all this is, is a fast-executing loop which builds a growing string of nonsense and pushes it into your history. And it crashes all browsers.

Now, what's really interesting is it doesn't just crash the browser in Safari on iOS. It crashes the OS. And that's actually - that's arguably a mistake. And but what's happening is the browser is consuming main memory at a frantic pace. If you display your computer's memory and run this, and I did, it's just like a straight line going up to the death of something. So anyway, that's what CrashSafari is.

The problem is, as you guys talked about on MacBreak, is that people are not just posting that link because, okay, who's going to click on that unless they actually are prepared to see what it does, with the expectation that it's going to crash something. The problem is, of course, URL shorteners turn it into something that is perfect for clickbait. And so what's happening right now is, especially because this has become very popular, it's all over VentureBeat, Engadget, 9to5Mac, TechCrunch, BGR, I mean, everybody has picked up on this because it's a simple little story. And hopefully, you know, and they're wanting their own people to click links to come and find out what this is about. It's now going rampant through Twitter and Facebook, people posting links saying that there's something else tricky or seductive and saying here, you know, click this.

Leo: For Anna Kournikova pictures. So I ran it on my Linux box because then I can run a top. And it's using - I have two Chrome processes. One is using 153% of the CPU and 75% of the memory, going up, by the way. And I imagine when it gets to 100% I'm going to be sorry. And then the second process is using 104% of CPU. So I'm kind of - I'm curious what's going to happen when it gets to - it's at 89% of memory right now, 90%. I probably should kill this process pretty quick here.

Steve: And you do have a thick skin on that ball you're sitting on; right?

Leo: Oh, it's going down. See, it only goes to 10,000, I notice, in this for loop. So you could really weaponize it and make it go to a million or something; right?

Steve: Yeah. I think he went that far because...

Leo: That's enough.

Steve: ...nothing survived that far. That was far enough to take things down.

Leo: It's plenty, yeah. It's funny.

Steve: And so it's been noted that there are, I mean, for a while there have been ways to perform little script games that would, like, lock up a tab. The problem is, as you found out, this doesn't just lock up a tab. It tends to lock up the whole browser because, again, this uses worker threads and ties them up, like core data structures, and so that they just, you know, you end up with more work than the browser can do and consuming huge amounts of memory. But, and so it's just not a matter of, like, dialogue popping up and saying, this script seems to be misbehaving, would you like to close it, and you say yes. This brings down the browser.

So what we'll see is, in no great urgency because this doesn't seem to represent any kind of a security flaw, and no one is suggesting that it does, it's just an annoyance, I expect there'll be some specific management of this. And again, it's not news. There was discussion among the developers back in 2014 that this push state history API was expensive to do, yet easy to ask for, and that could cause a resource problem. But meanwhile, Matthew Bryant got himself in the news.

Leo: Oh, yeah? What did he do?

Steve: There is a new Firefox that dropped just this morning for Windows, Mac, Linux, and Android. It adds something which Chrome 42 got last April, which is sort of controversial. Maybe you've seen, when you visit a website, a pop-up saying would you like to receive notifications from this site? That's a new feature of, again, the HTML5 growingly and increasingly supported API, which allows you to give your browser permission to receive future notifications from sites you're not even visiting.

So the good news is no browser - you know, 10 years ago, if this had been a feature, we could foresee that browsers might say, oh, fantastic, and they would just implement it. Thank goodness we're 10 years downstream, and everybody has learned a lot of painful lessons, which is ask the user before you do something like this. So all the browsers do prompt when the site says it would - the site is actively saying we would like to provide future notifications to the user, which conceivably could be useful. Expedia, for example, might want to notify you of some change in itinerary, and so this would allow them to preemptively do that. So anyway, this is part of the spec. It's been around in Chrome since April of last year. Now Firefox 44 has it. And so far I don't think I've ever said yes. Generally I want to deal with sites that I'm working with.

Leo: That's annoying.

Steve: Yes, and not start getting pop-ups, like out of nowhere. Also Firefox formally dropped all support for RC4, meaning that it no longer, in its TLS client hello packet - remember that the browser sends, when it's establishing a TLS secured connection, it offers a suite of security, well, it offers a set of security suites using different types of encryption and hashes for authentication and so forth. Among them historically has been RC4. And back in the day it could be running on as weak a key as 40 bits. That was the old - it was RC4 that was running with 40-bit encryption, which was then phased out because it became too insecure. Even at 128 bits, RC4 itself started to just get wobbly under the analysis that we now know how to provide and the power that contemporary machines have. So, gone. It was dropped last week from Chrome 48, and now from Firefox 44 today. These mainstream browsers just don't even offer it.

So 44 also has a bunch of very snazzy-looking new web page authoring tools, improved warnings about certificate errors and untrusted connections. If the system has an H.264 decoder available, it will now make it available. They backed off from that for a while, but now it's back. And they've also enabled WebM, which is that VP9 version support, is enabled on systems that don't support MP4 or H.264. And support for Brotli compression format.

Leo: Woohoo, Brotli.

Steve: Oh, I forgot to say, I forgot to mention also, one thing that's completely arbitrary and annoying, but it's like, okay, only supported over HTTPS.

Leo: Oh, that's interesting. Why is that?

Steve: For no reason. For no reason except that Google wants everything to be HTTPS. So they're just saying, okay, one more reason to switch. By agreement, no one is going to support Brotli, although they absolutely could, I mean, the TLS tunnel that everything runs through has no impact whatsoever on the compression or not of what runs through it. So this is completely arbitrary. But they just said we're not going to support it under HTTP. It's like, okay. So yet more reason to run secure, as you'll get an incremental increase in performance. Probably nothing you'll even notice, but it's good. It certainly benchmarks nicely.

They offered a bunch of security fixes. They removed a couple CAs. Firefox will no longer trust the Equifax Secure Certificate Authority 1024-bit root certificate, so that's just been phased out. I mean, there are others that all of their recently signed certificates support. So this, you know, it makes sense to remove those. And then they did also add a built-in JSON reader, which allows people to view, search, copy, and save data without needing third-party extensions added to the browser. So some nice incremental movement forward on the browser that a bunch of people are using.

Okay. Miscellaneous stuff. I wanted to let our listeners know that "The Expanse" continues to get better. Have you started doing any watching, Leo?

Leo: Yeah, no, I haven't. I've been watching "Billions." I love it.

Steve: Good. I am, too.

Leo: Yeah, great choice on that one.

Steve: Yes, really looks fun. So "The Expanse" has been compared to, and I agree, "Firefly."

Leo: Well, now I have to watch it. Boy, that's...

Steve: It's reminiscent of "Firefly." Someone tweeted me named Peter. He said, you know, how can this be so much better than anything Syfy has ever produced? And I wrote back, I said, "I have no idea how they're making it so good. And I agree, it has become absolutely fantastic." And I said, "And, since I read the books, just wait for this week's episode. We get to see the first real hint about the much, much, much, much bigger thing that's actually going on."

And I have to tip my hat to the guys that are putting this together because there is something that was briefly hinted at in the first very awkward couple minutes of the show that they haven't, like, that no one has any idea about, if you didn't read the book. The book was able to spend much more time on that first thing. And so we know much more about it, normally, if we were reading the book at this point. But based on the little previews from last week, I can say that tonight's episode is going to be amazing. We're going to see something that no one who hasn't read the books, who's just watching this, really will have understood. And it completely changes everything. It's really good.

So then, out of nowhere, I got a tweet from someone named Ben Roberts. He said: "@SGgrc Thanks! Glad you're enjoying the show!" And then #TheExpanse.

Leo: What? Who's Ben Roberts when he's at home?

Steve: And so I clicked on that because I was wondering. Turns out he's the senior VP of TV with Alcon Entertainment who produces "The Expanse," "Ice," "Sinatra" on HBO, and many more coming soon. And he formerly developed AMC's "The Walking Dead."

Leo: What? Wha-wha-what?

Steve: There is some production capability.

Leo: I think you should tweet him back and say, "Can I come see the studio?" Go see it. Go see it. It would be fun, wouldn't it?

Steve: Anyway, so I just have to say to everybody, if you like sci-fi, I mean, S-C-I hyphen F-I, and hate S-Y-F-Y, and I completely understand that, this is an exception. It is really good. Oh, and last week was the first two, or no, sorry, last night was the first episodes of "The Magicians," which is another series based on a book series. And it seems like Hogwarts for adults. But it was fun. And I saw...

Leo: I don't know how you find time to watch this much TV.

Steve: Actually...

Leo: That's why I haven't watched "The Expanse." I just don't have time.

Steve: People think I watch a lot of TV, but I don't watch anything you don't hear about.

Leo: Right.

Steve: So that's like a couple hours a night.

Leo: No, it's not that bad, yeah.

Steve: So, you know, if that. I did find in the mailbag, from Dave White in Auckland, New Zealand, a fun story about SpinRite. He said, "Hi, Steve. A quick SpinRite story for you. I bought a copy of SpinRite from you about two months ago, not because I needed it at the time, but just to give you a 'yabba dabba doo' in appreciation for the Security Now! podcast..."

Leo: Yabba dabba doo.

Steve: "...which I listen to religiously every week." And he says, "(I'm also filling in my drive time by starting at Episode 1 again)." Talk about having time on your hands.

Leo: Geez.

Steve: But actually there are - I received so many favorable replies from our holiday repeat of the Vitamin D episode, not necessarily because, or not only because, of its content, but because, I mean, it turned out to be important to say again after six years. So I have to say to our listeners, just because these are old doesn't at all mean that there's not a ton of interesting relevant stuff. We did a series on how the Internet works which I tried to repeat by doing it again, but we just keep getting buried by news. We just don't have time to do like I used to. So there is back there, waiting for you...

Leo: There's a lot of stuff.

Steve: ...like the evolution of processor technology from the beginning transistors and on upwards, how the Internet works, all kinds of stuff. And Dave White, while he's on his commute, is going to be learning about those things. So for what it's worth, there is lots more to fill your drive time with that I think you'll find interesting.

Anyway, he says, "I am a freelance IT guy in Auckland, New Zealand, and look after IT needs for a number of small businesses. This morning I heard my cell phone ringing before I'd even gotten out of bed - never a good sign. It was a customer whose main Windows 7 PC," he says, "(the one with their shared files and database on it) was getting stuck on the welcome screen immediately after logon. Once I got to their site," meaning physical location, "I was able to boot it into safe mode and attempted a system restore, which also failed with an error indicating a problem with the drive's volume shadow copy. I tried it a second time, and this time got a BSOD." Which we know is the Blue Screen of Death.

He says, "I booted back into safe mode and ran chkdsk /r." "R" is for "recover" on chkdsk. "Forty minutes later it reported there were no problems on the disk. But my gut was telling me otherwise. I unplugged the PC, took it back to my office, and booted it up with my SpinRite disk. Three hours later, I was on my way back to the customer with a fully working PC on the back seat. The customer was of course delighted, so thanks a million for making me look good. One last thing: It may be my overexcited imagination, but I swear that PC ran faster after the SpinRite repair than it ever had since I installed it five years ago."

Leo: I don't think that is your imagination.

Steve: "Thanks again. Regards, David." And of course we know it can be because sectors can become difficult to read. The drive will give them many spins before it gives up. Not nearly as many as SpinRite, but that way a drive can slow down while it's still working. And of course that is one of the clues that you need to run SpinRite. SpinRite will fix it before it gets this bad, and also speed it up again.

Leo: All right, Steve. I have questions. And I didn't write them. They're from our audience. Let me pull them up here, and we'll see what you have to say, starting with Question 1 from an anonymous listener - this is related - who once worked for a company with backdoored products. Whoa.

Steve: Yeah.

Leo: Steve, I worked for many years for a succession of major telecom firms, and I spent a long time working on that software for them. Some of our products had backdoors. For all I know they still do. The reason was a desire to be able to undo the effects of someone who had legitimate access and decided to, I don't know, change all the passwords, for example, in the event of a disgruntled employee taking preemptive action, or a union member in the face of an upcoming lockout where management personnel would be required to perform regular maintenance. Here's the thing: People will always come up with what they think are good reasons to add a backdoor. Sometimes it stems from a reaction to a previous bad experience. You should assume backdoors exist and must therefore take steps to mitigate against them, just as we do in the face of insecure IoT devices.

Steve: I thought this was interesting feedback from the trenches. And I guess I understand, I can understand his position. What I'm hoping is that, if the big iron backdoors that we're seeing are deliberate, that companies are coming to understand that what may have been common practice in the past must absolutely change. We know, you know, how many routers, how many consumer routers have we covered where backdoors were discovered? Where Internet-facing, undocumented passwords or ports were left open, in some cases with complete command libraries of things that could be done. Clearly, some manufacturer thought, oh, you know, we'll put this in there so ISPs are able to remotely manage the router.

The problem is this is the kind of thing that is incorrect, where once some information gets loose, anybody is then able to exploit it. That's the problem, this sort of backdoor. We talked last week about the Fortinet big iron hardware or appliance that had, yes, it was a good password, it was all cryptographic and crazy, but it was in the firmware. And once you know it, you know it for all of the devices because they all had the same one. This was, like, their secret way of accessing the device remotely.

So [stammering], I'm speechless. The only thing I can think of that consumers can do is to, if you have control over the Internet-facing hardware - this is the problem. The big problem is the Internet-facing hardware. It's a fact that these big iron appliances were on the Internet, protecting a corporation, and could be accessed remotely; or that consumer routers provided by an ISP are the way the ISP interfaces to the user's local network. And again, those devices are Internet facing. We've seen some mitigations where, for example, only the ISP would have access to the backdoor. That could be done using protocols that don't require routing because, for example, the ISP doesn't need routing. It has direct connection to your device. So that's not such a big problem.

The problem is, when an ISP-provided device is not something that the user has any control over, what do they do? I would say you get your own device and put it inside of the publicly exposed device over which you have no control. We know that it's possible to chain routers. We'll talk a little bit about that again later on in the Q&A today for a different purpose. But in this case, you want anybody that gets into the publicly facing router not to be able to go any further, not to have access to your internal network.

Well, the way to do that is have it chain to a second router, which is your choice, running probably, hopefully, open domain or well-vetted, or just simple. It only needs to be a very simple NAT router, something stripped of features that only does one thing well, and that is keeps anything unsolicited from the outside from getting in. And it won't slow anything down. Basically it's sort of working more like an intelligent switch that only allows outbound traffic to go out to the public-facing router. That's the only thing I could see that someone can do who's worried that they've got a router on their border, bordering the public Internet, over which they have no control because we see over and over and over that apparently deliberate backdoors are there.

And in some cases you could say, well, it's just oversight. But it doesn't matter because bad guys are able to get in. And now we know that they're even creating worms, creating botnets of routers that have these exposures. This would not prevent that. It would not prevent your border router from being taken over, but absolutely putting a router inside would protect your network from anything that that router did. And that's the only thing I can think to do. At a corporate level, they really - corporations are going to have to hold companies that they get this big iron from to account, explicitly ask them, put in a contract, that this device has no undocumented remote access feature.

Leo: Ooh, I like that.

Steve: So that, if it turns out it does, they can take them to court.

Leo: You're on the hook, yeah.

Steve: Yup.

Leo: I like that idea.

Steve: This is the kind of thing that could have buried Sony, for example, or any of these major networks that were taken over. Big companies have big iron technology, and now we've got two examples of these things having revealed known backdoors.

Leo: Mark in Ann Arbor, Michigan with a special case problem. He's a road warrior using a VPN: Thanks for all the great information you and Leo provide. I've been listening since about Episode 30. I never miss a week. I'm also a happy SpinRite owner, and it has earned spousal support on more than one occasion. Yeah. It's important. You hang SpinRite on the wall, spouses approve it better.

I work in the healthcare field, and therefore security is something of which I'm very aware. My question concerns VPN, Virtual Private Networks, and being secure when out of the office. While I generally use my iPhone as a hotspot - actually, he just said "phone," not "iPhone" - occasionally I must use hotel and airport WiFi. I've been using the D-Link AC750 portable router - which I use also and recommend, it's a great little router that can also just be a battery pack - to pick up the WiFi and then connect my device to this. I then run my corporate VPN on the computer.

Would you consider this method secure? Unfortunately, with this solution, if I want security, I believe I must run a VPN tunnel for each device that connects to the router. In Episode 541 you and Leo discussed use of the Tiny Hardware Firewall so that all traffic running on the router is protected. But for this I must use their VPN server endpoint rather than my corporate VPN. I need to use my VPN in order to connect to drives and servers at my medical center. So your advice?

Steve: Okay. So what he's saying is that he has a need, when he's on the road, to use his corporate VPN in order to access his corporate resources.

Leo: Right.

Steve: But when he's doing that, then other devices don't have the advantage of security. And if he uses something like the Tiny Hardware Firewall or the - oh, I guess he uses the D-Link portable router with his corporate VPN.

Leo: Yeah. It's just a portable router, yeah.

Steve: Okay. So here's the deal. Nothing prevents a VPN from tunneling within a VPN.

Leo: Oh. Hmm.

Steve: What you want to make sure, and this is normally the default, is that you use UDP protocol for the VPNs. We've talked about this a couple times. Essentially you want an encrypted tunnel which doesn't add its own error recovery because the protocol brings its own recovery with it. And if the tunnel does error recovery, and the protocol being carried by the tunnel both do error recovery, then you can have things like - it's called a "tunnel stall," where they get into sort of battle over who's going to do the packet retransmit request, and they both end up doing it, and things get confused.

But the major point here is that it is very likely that he could use the D-Link, or I'm sorry, he could use something like the Tiny Hardware Firewall to bring up an umbrella VPN that would protect everything from his location, all devices that use it through the VPN, and it would still be compatible with running his corporate VPN through it. That is, nothing prevents a VPN from being run in a VPN. Performance, eh, that would suffer a little bit because you're going to go out to some other location. But you're then going to emerge from the outer VPN tunnel, and your traffic is going to go directly over to the corporate VPN. So I think it should work, too.

So I would say give it a try. It will probably work. And then you've got the best of both worlds. You've got a VPN protecting everything at that location, and you can still get a hold of your corporate assets over a specific VPN.

Leo: I had no idea. I thought that double, there would somehow be double overhead or something weird going on.

Steve: No. Should work just fine.

Leo: Nice. Nice to know. So get the Tiny Hardware Firewall.

Steve: Yes.

Leo: Mark in Ann Arbor has a Part 2, but it needed its own question. Another related question to VPNs: How do I know my connection is running through the VPN? I see the icon for the program in the system tray, but I realize programs - web pages and email clients and so on - do not have to reconnect to their server. How do I know that the more secure connection is actually ongoing?

Steve: Yes. That's a great question. And the easiest, cleanest generic answer I know of is to ask other websites to tell you the IP they see.

Leo: Right.

Steve: That is, if you ask, for example, you can just, in Google now, I think you can put in...

Leo: You say, "What is my IP," yeah.

Steve: Yes, "What is my IP?" And it will show you the IP from which your traffic is emerging on the Internet. That's the key. So when you don't have a VPN up, and you do "What's my IP," you'll see the IP address of your location. It'll be the Marriott, or it'll be whatever. But remember what that is, then bring up the VPN and do it again. And that IP should be different. And, now, you can go one step further. So one thing is to see that the IP differs with and without the VPN running, which says that your traffic is emerging onto the Internet, either from where you actually physically are, or at the exit of the VPN tunnel, wherever that server is.

But you can go one step further, and that is, try to do what's called a "reverse lookup." DNS we've talked about often, converts friendly names like Google.com, GRC.com, TWiT.tv, converts those friendly names into an IP address. It turns out that many, but not all, DNS servers - but probably more and more now, it was rarer in the past, it's becoming more common, at least from what I'm seeing - also do the reverse. There's something called "reverse DNS" which, as it sounds, you give it an IP, and it tells you something, at least something about the IP, often the matching domain name.

And so, for example, all of GRC's IPs have reverse DNS. So you could put 4.79.142.202 into a reverse lookup, and it would say, ah, that's www.grc.com. There are some simple web-based tools. And I don't know, I didn't think to ask, to see whether Google will do that for you. In the same way that it does a forward lookup, maybe it'll do a reverse lookup. But there's one, for example, MxToolbox.com/SuperTool.aspx. I have three links in the show notes for anyone who's interested. That was my favorite of the three that I listed. There's also DNSGoodies.com that tends to be a little more techie.

But the point is, once you've got an IP address without the VPN and one with the VPN, sort of for fun, do the reverse lookup, and you might find that, oh, look, sure enough, the IP that I get with the VPN up is what I expect. It's the company whose VPN server I am using. And the reverse lookup without the VPN up is, oh, yeah, it's the Marriott or something where I'm staying for the night. So anyway, that's really, I think, the best way to tell. You could do, like, traceroutes, maybe. But I think the cleanest solution, and it's pretty much foolproof, is just compare IPs with and without.

Leo: Nice. By the way, somebody, Andy in the chatroom, says you can just type "IP" in Google, and it'll give you the IP, which it does.

Steve: Nice.

Leo: Yeah, it makes it really easy. I do that with Tor, too. It's fun because first you turn on the VPN, and all of a sudden you're coming from Chicago. Then you turn on Tor, and all of a sudden you're coming from Belgium.

Steve: Yes.

Leo: I love that. Question 4 comes from Stockholm, unless he's using Tor. I'm going to be in Stockholm in September. Olivier wonders about phishing attacks: Guys, great podcast. You rock. In your "LostPass" episode, you seemed to imply that even two-factor authentication wasn't able to protect against phishing attacks. But of course not all two-factor authentication systems are equal. Why doesn't two-factor authentication protect from phishing, and particularly this phishing attack?

Steve: Yeah, okay. So this is...

Leo: There are two-factors that would work, by the way, with LastPass because there are some that use that weird - I don't even understand how it works, where you click a link, or you do a weird thing when you're in your app, and it just suddenly works. Anything where you'd enter in a six-digit number would be at risk; right?

Steve: Correct. And it turns out the thing where it just works is also not secure.

Leo: Oh. Good to know.

Steve: That's Question No. 5.

Leo: Okay. Yeah, Microsoft offers something like that.

Steve: Yeah. So here's the problem. And I chose these two questions because it's really important for our listeners to make sure they understand this because this is like the next frontier of exploits. And the problem is that, when you're seeing a page which you think is the site you're authenticating to, but somehow a man in the middle has managed to get you to go to their site, they've redirected the traffic, they've redirected DNS, they've changed a link and maybe given you a certificate, I mean, all the things we talk about that we're increasingly getting better at preventing, there are still ways that people get tricked. Clicking on something in email that tells you that you're going to go to Amazon, and you think you're there, but you're at Amazone instead, and you don't notice.

Now, here's the problem. This is why two-factor doesn't help you, is that when you brought up the login page, thinking it was Amazon, back at this bad guy's server, they connected to Amazon. So they've got the connection to Amazon. And in fact they may be in fact feeding the Amazon page through them to you. So you're even seeing the Amazon page that you would normally see, but it's been relayed through them. So when you put your name and your password in and hit Submit, you're sending it to them. You think you're sending it to Amazon because it looks like you are.

But this is where - and this is the problem. We just go by the way things look. And the fact is the page came from somewhere else. So the data we submit goes back to that somewhere else. They then take it, and they have your username and password. And they forward it to Amazon because, remember, they're the only ones actually connected to Amazon. You're not. So Amazon says, oh, correct username and password. Ah, but you've enabled two-factor authentication. So your phone should beep and tell us your six-digit key. So that echoes on your screen.

Sure enough, your phone beeps, and there's your six-factor key that Amazon has just sent you through the second-actor app. Which of course you enter into your browser. And remember, the other guy is still in the middle. So when you submit that, it goes to them. They submit that same six-digit second-factor key to Amazon. Amazon thinks they got it from you, doesn't know there was a relay, that there was a man in the middle hiding. And they now have your login.

Now, what they don't have is the ability to log in any time they want in the future because that does require a dynamic second-factor authentication. But they're logged in once, and if they've just logged in, they may be able to get up to all kinds of mischief. So I just wanted to explain that the second-factor does add an aspect to the attack. That is, whereas a username and password are unchanging, and thus they can be captured and reused by a man in the middle, whereas the second factor's nature, a changing second factor, can't be captured, it can still be used dynamically to allow someone into your account, and they will have a fresh login.

And even Amazon allows someone who has just logged in to do a lot of things without requiring additional authentication. And of course Amazon security tends to be better than many because Amazon does tend to make you reauthenticate when you do things that require, like, reverification that you're still you; whereas most other sites don't bother with that. So it can be even more damaging elsewhere. So anyway, I just - I liked the question because many people were wondering, and there is a difference between static information and dynamic. But if the attack is dynamic, even dynamic verification doesn't help, or at least doesn't guarantee.

Leo: And I will ask, I'll defer because I'm looking at all the other forms of authentic 2FA on LastPass. And there's, besides Authenticator, Toopher and Duo and Transakt and Grid. And so I'll defer on that one because you say we're covering that later. And I guess you probably should throw in YubiKey, as long as we're talking.

Carl Thompson in Leeds, U.K. has a solution for LostPass: I just wanted to bring your - oh. Okay. I just wanted to bring your attention to the two-factor authentication system called Transakt, that's with a "k" instead of a "c," that works with LastPass. And Duo does kind of the same thing. When you log onto LastPass, the Transakt app on your smartphone pops up and asks you to confirm the login. Twitter does something kind of similar to this. And Microsoft has an app that does this, too, for two-factor on their account. You simply confirm or reject the logon request, and that's it. As this is out of band and does not require you enter numbers on your computer, well, that would eliminate any phishing issues; right? So two-factor is not broken with LastPass. It's alive and well as long as you choose the right type of two-factor authentication.

Steve: So let's run through the scenario.

Leo: Okay.

Steve: Same setup as we had before. Some guy, some entity has arranged to insert themselves, has tricked you into going to their site, and you believe you're somewhere else. So when you attempt to log into LastPass, LastPass presumably notifies them, this provider, of your identity and to confirm that you want to log in. So the phone rings, and you say, yes, I want to log in. So, and if someone has imposed themselves in the middle, you have just authenticated their login to LastPass, rather than yours, because again they're in the middle.

So the fact that it is out of band and doesn't go through the same browser channel doesn't provide you any protection whatsoever. You're expecting LastPass to prompt you for your second factor, in this case Transakt or any of the others. So your device says, "Are you logging into LastPass?" And these things brag that it's like accept or decline. It's either a big red or a green button. So since you're expecting this to come up, you say yes. And you've authenticated the attacker's login, rather than your own. So again, it's an additional factor. It unfortunately doesn't help you.

Leo: Hmm. Interesting. Would a YubiKey or a FIDO key help with that? Or same problem?

Steve: Same problem, unfortunately.

Leo: Because you could be spoofed, yeah.

Steve: Yeah. The way we solved this with SQRL is that the SQRL token, that little barcode, embeds the IP address of the entity that requested it. So if a bad guy is intercepting you and tries to log in, they obtain a SQRL code with their IP, which they then present to you.

Leo: Ah, clever.

Steve: Uh-huh, trying to get you to authenticate for them. But the SQRL client running in your phone or on your computer has a different IP. And so the server, there's the bit flag in what we call the TIF byte, the Transaction Information Flags, where the server notes that the IPs are different and should not be different and immediately shuts down the transaction. So SQRL has solved the man-in-the-middle problem. As far as I know, it isn't otherwise solved.

Leo: Very cool. Again, another reason to like SQRL.

Steve: We're getting there.

Leo: I just, you know, I feel like the powers that be are going to have their own solution, and they're just never going to think about anything else. Google has FIDO2 and...

Steve: We have a lot of corporate interest, where individual corporations want to use this for their own solution. And I just think...

Leo: That's great.

Steve: ...if it's better, it'll be organic.

Leo: Yeah. Yeah. I always think that cream should rise on the Internet.

Steve: And if not, I had to do it anyway. I mean, it had to be done. And there are a lot of people who have helped over in the newsgroup, working to refine this. And we'll see what happens.

Leo: I hope it happens everywhere. It'd be great.

Steve: It's just so simple and lightweight.

Leo: Yup. Steve Gibson, Question 6 comes to us from George Kapp in Clinton, New Jersey. He writes: Steve and Leo, thanks for many years of great podcasts. I'm a listener since Episode 1. Wow. I have a question regarding how certificates and domain names match. My company uses a security training service, blog.knowb4.com, that allows us to simulate phishing attacks and keep our users on their toes. That's a great idea.

Steve: Yeah.

Leo: Fake phishing attacks. That's what happened in "Billions." Oh, I won't say anything. I recently visited their site and - you watched the second episode Sunday?

Steve: Oh, yeah.

Leo: I'm feeling like it's getting even better. I'm loving it. I recently visited their site, and I checked the certificate before downloading a file. The site showed up green with the lock icon showing in my Chrome browser. However, the name of the site did not seem to match what was on the cert, secure006.hubspot.com on the cert VS.blog.knowb4.com. I found this alarming until I noticed a list of 20 to 30 DNS names attached to the certificate, including blog.knowb4.com, along with several other sites I use and many I've never used. It does look legit, but this is not working how I expected. What's going on here? Thanks for all your great work over the years. George.

Steve: So this is something that is sort of, I don't want to say disreputable or cheesy...

Leo: Go ahead, say it.

Steve: But I have. In fact, Leo, if you go to https://secure006.hubspot.com in your browser, and then examine the certificate, you know, click on the padlock or however you do, look at the certificate, and then look at...

Leo: Ooh, 22 connections from this site, 140 from other sites. Wait a minute, no, that's cookies. Hold on. I've got to zoom in. I'm blind now. So let's see. The connection is not private. Permissions. That's cookies. The identity of this website has not been verified. Your connection is not encrypted. Well, that's not what I thought I was going to get.

Steve: Ah, so did you do HTTPS?

Leo: Maybe that's why. All right. I just clicked the link.

Steve: Ah, right.

Leo: Oh, now I - well, do I? Yes, there's a certificate. Okay. Your connection...

Steve: Okay, so now...

Leo: ...is encrypted using a modern cipher suite. Fantastic.

Steve: That's all good.

Leo: Yeah.

Steve: So you want to look at the certificate itself.

Leo: Right. We're watching it over here.

Steve: And what you're looking for is something called...

Leo: [Crosstalk] cert?

Steve: Yeah, look at the cert.

Leo: Details.

Steve: The subject alternative names field.

Leo: Whoa.

Steve: There it is. Yes.

Leo: What the what?

Steve: So this is something I don't know if we've talked about much. But there is no practical limit on the number of explicitly enumerated domains that a certificate can use. And when I was talking about sort of cheesy or budget, I mean, the advantage is that a site like that only has to issue one certificate, or pay for one certificate, and it covers a huge number of domains. Now, if they're domains in different countries, for example, and there are, for example, there's hubspot.es and .en and .jp and .com and .net and so forth.

Leo: That's what I'm saying, yeah.

Steve: Then I think it makes sense because there's no reason these people should have to pay for, like if they're related properties, or they're all owned by the same company. I have seen certificates that are just a scattershot of a hundred completely unrelated domains. And that's sort of freaky. What that says is that maybe that cert is only on one server, and that server is doing multiple hosting of all of those domains. But it's also likely that that one certificate has been put on servers all over the place, and that can create some security problems because now you've got a - you go to the site for a specific domain, yet your connection technically is valid for all of those domains listed in the certificate. So it does open up an opportunity for exploitation.

But for what it's worth, that's what's going on. It's called SAN, Subject Alternative Name. And so the certificate itself will have its primary name. In this case it's secure006.hubspot.com. But both the server and the client look at all of the names, the main name and all of the enumerated domains in the subject alternative name field, and will match on any of them. So it's like, you don't like to see certificates like that. But I can see the need. And the good news is something like Let's Encrypt will be able to obsolete that by allowing individual servers as needed to get certificates for the domains that they're hosting.

Leo: Okey-doke. Well, that's good to know. They are doing a lot of business. They own Sidekick.com, too. Interesting. They're a...

Steve: Oh, that's a nice domain name.

Leo: Yeah. They're an interesting company. Hmm. They're kind of a demand media company. They have an interesting business model. Moving on here, Question No. 7. You know what's weird, I clicked that link, and it did have an HTTPS in the link, but I didn't get an HTTPS site. So I don't know what that means. Adlai Chapman in the U.S. sees a worrisome connection in his netstat: When I run netstat -o with no Internet connections open, I keep getting back an established IP address. I looked at the IP address on Google, and it's in the Czech Republic, company T-Mobile. The IP address: 193.85.216.235. I have two computers, and this IP address keeps showing up on them. What's going on?

Steve: Okay. So I don't want to freak people out, but one of the most useful things you can do...

Leo: Is run netstat.

Steve: Yes, is what Adlai did. And again, everyone should be sitting down. Don't have too much coffee first. And then open up a command window in whatever operating system you have because netstat has been around since the UNIX days, meaning it's a core network utility that all operating systems have copied. And this shows you the current set of connections that your local computer has to the outside world. And it's an education. I can't, there's no way to diagnose - oh, Leo just did it, for those who don't know why I was laughing.

Leo: Boom. There's quite a few. So some are websites I'm on, but some are services running in the background.

Steve: Yes. Now...

Leo: I don't know what a lot of these are.

Steve: Yeah. So the thing to do is to just sort of go through them and try to account for them, try to - if you do -o, it may only show you established connections, rather than listening ports. So that's more useful for most users. And in fact, if you do netstat under Windows, space slash question mark, you get a short little help screen of options. But it is...

Leo: On the Mac, netstat doesn't have -o, unfortunately. So I don't...

Steve: Oh, okay. Maybe something that shows connections.

Leo: Connections only, yeah.

Steve: Yeah.

Leo: I'll have to figure it out.

Steve: I sometimes do, what, the -an on Windows. I'm not sure what that was, "netstat -an"? Anyway, it's a useful thing to do. I can't understand or explain what Adlai is seeing, but it doesn't look good. Something, what this says is there is something about his computer or in his computer that is establishing a connection to that remote IP.

Leo: Wow.

Steve: There really is no other explanation. So it's useful, just sort of as an inventory-taking exercise, open a command window, "netstat -an" I think is the one I tend to use under Windows. He gave the example of -o.

Leo: Sounds like a Windows-specific command.

Steve: Yeah, and just sort of go through and make sure that it looks reasonable because one of the things that are going on behind your back are establishing persistent connections to the Internet. Google has some for me. My news client has one to GRC's news server. And a site that you visited recently, those connections may not yet be torn down. So again, don't worry about it too much. And one thing you can do is do a clean reboot, also.

Leo: Yeah.

Steve: Restart your system. That'll flush all that. And then once it settles down, do a netstat before you do anything else. And that'll give you some sense for what it's doing just as like the built-in stuff. And make sure they make sense to you. I'm glad he mentioned it because I have not talked about it often enough. It's somewhat frightening. Again, so don't get too worried, but it's worth taking a look.

Leo: Dash "o" displays timers.

Steve: It's completely under - I'm sorry, what?

Leo: Dash "o" displays timers in the UNIX command. I don't think that's what you want. You just "netstat -a" will give you all the listening ports. I think that's probably enough. And you can do -help to see. I see more commands here on the UNIX side, on the Linux side.

Steve: Yeah, actually -o in Windows is a good thing. So for what it's worth.

Leo: Okay. For some reason that's something in there. All right.

Steve: Yeah.

Leo: Include information related to networking timers.

Steve: Yup. Different on - and I should mention that they're not all synchronized, all these netstats, especially from Windows and any of the other UNIX and Linux derived.

Leo: Well, that's the thing. Mac is using, is BSD. And so this Linux - or these are both Mac or UNIX commands. Let me try it on my Windows machine. Anyway, go ahead. We don't need to keep going. Netstat -help. "O" displays owning process ID associated with each connection. Ah, see, that's not - that is useful.

Steve: Yes.

Leo: That says what process owns this.

Steve: Right, who opened this up.

Leo: Yeah. And I only have one active - oh, sorry, many more coming in now. Takes a while sometimes.

Steve: Yes. Actually the -o option takes a while for that enumeration to occur.

Leo: Yeah, yeah. But they're coming in. I don't see Czechoslovakia. You know what, this is - it's nice because we're on the corporate network here, and this Windows machine doesn't really have much going on. And so it's pretty clean.

Steve: Yeah.

Leo: All right. Good, good question.

Steve: Yeah, it is. It just - it's always there. It's easy to do. And you'll spend some time unraveling it. But it gives you a good sense of confidence if you know that nothing's going on that shouldn't be. But also it's like, wait a minute, that thing's still connecting to somewhere? Let's remove that. So it's just a nice thing to check on.

Leo: James in Christchurch, New Zealand wonders whether there are any "right" ways to allow government censorship: Steve, reading the news today, our New Zealand Chief Censor - wow.

Steve: Yeah.

Leo: I'm glad I don't live in a country with a Chief Censor.

Steve: They even have a title of that.

Leo: Geez.

Steve: Wow.

Leo: Is calling for a discussion about whether there is a place for widespread filtering - God bless the First Amendment - to help protect New Zealand from the often-debated harmful effects of online pornography. References were made to the U.K., where users can opt out of filtering. Actually, I think it's the other way around in the U.K. You have to opt out. You're opted in by default.

I'm wondering if you have any thoughts or technical solutions for whether there can be a balance achieved between best-effort filtering of this content for those who want it, while still ensuring their privacy and security, for example, not doing HTTPS inspection. Perhaps ISPs would need to provide alternative DNS servers, like OpenDNS, which would block content based on publicly auditable block lists. Thanks for the podcast. Listener from Episode 1, et cetera, et cetera.

Steve: So, you know, I thought this was a good question. And of course I agree with you, Leo. I thank goodness for the First Amendment free speech rights that we have in America. The problem is, when we've discussed any kind of filtering, is that not only is somebody else making a decision about what is acceptable and not, but the Internet was not designed for it. It's one of those things that is, like is trying to be added as an afterthought. And it just - it isn't going to work right because it isn't designed to have censoring.

I often, where I hang out, I use the Verizon available public WiFi because there's a Verizon store two doors down. And it's like, hey, might as well do that than use my cellular. So I do that. But I'm often poking around in medical research recently, and I occasionally come to a site that the nanny protector has decided I can't view because of this, you know, it's bad content. But it's like, no, it's a link for Medscape, you know, hosted by the NIH. And it's like, it's not wrong. But that's what these things do.

And of course now that we're moving into, and as James mentions, now that we're moving into an HTTPS world, either preventing people from getting DNS, which somebody decides is bad, the way OpenDNS has provided an optional filter that allows you to do that, you need to crack open connections if you're going to do deeper inspection. And then of course that has huge privacy implications. So he poses the question, is there a right way to allow it? And so the first question is should government censor?

Leo: No.

Steve: And putting that aside, is it practical? Can it be done? And I don't think there is a technology that works well enough that it makes censoring even technologically possible, if you decided that it was something that you wanted. The Internet just - it fights it.

Leo: It routes around censorship. That's the whole design.

Steve: Yes.

Leo: Hmm. Dan Sullivan in Florida suggests the Ring Doorbell on the guest network may not be practical: I don't think putting a Ring Doorbell, or I guess we could say any IoT device on this, on the guest network of a Netgear wireless router is a practical solution. For your smartphone to communicate with the Ring, it has to be on the same network as the doorbell. Not true. By default, devices on the Netgear guest network cannot communicate with each other. That's why you use it. I'm sorry. I'm annotating. Enabling devices on the guest network to communicate with each other also enables devices on the guest network to use the main network, too. So that defeats the purpose, of course, of putting the Ring on a guest network. He's not understanding how the Ring works. The Ring uses the Internet, the public Internet, to talk to your phone.

Steve: Right.

Leo: You don't have - now, Sonos would be another matter. So Sonos you do need to be on the same network as the Sonos. But that's not trying to communicate with the outside world.

Steve: Right.

Leo: So obviously the issue with the Ring Doorbell is it's communicating to the Internet. That's how it talks to your phone.

Steve: Right. And so it's certainly the case that if your phone was outside by virtue of being on cellular, obviously it would be able to get to your Ring, whether you're away from the house or at home.

Leo: Right.

Steve: What I didn't know was if your phone was on a WiFi network other than the Ring, is it still able to get to it?

Leo: No, but it doesn't need to. Well, wait a minute. Maybe, no, it doesn't need to. I was thinking maybe - and I'll have to remember back in configuration. In the process of configuration it might. But it doesn't matter. When it's on the home network, you're still using the public Internet to notify the phone. It doesn't switch to the local network.

Steve: So, however, I liked Dan's question, and it is emblematic of a number that we received after we were talking about this, this idea of using a guest network. And he mentions that enabling devices on the guest network to communicate with each other also enables devices on the guest network to use the main network, too. So if you do inter-guest isolation, then they only see the Internet. But there are applications where you want cross IoT, like the Sonos example. You want devices on WiFi to be able to talk to each other. And then there's the question of not all routers have a guest network, or we're not quite sure exactly how it works, what it provides, what it allows, and what it doesn't.

So although it's not as easy as flipping a switch, which may not be available in your router anyway, the fallback, which is really a better solution, if you're concerned about this, is to get a second router, maybe it's the older generation WiFi router that's now in the closet or in the garage, and plug it into your main router. It's your guest router. Now give it its own password. Everything on that WiFi router will be able to see each other fluently, with no trouble at all, because it's just a regular WiFi network, although it's your secondary, it's your IoT WiFi network.

And by virtue of it being plugged into your main router, it will have outbound-only access to the Internet. There is no way that anything, any mischievous behavior of the IoT devices can get over, can like jump through the router that controls them and then through the exterior router, which has your primary non-IoT WiFi, and the rest of your Intranet.

So just using the older generation, the wireless router that you've retired, make that your IoT router, plug it into your main router, that is, plug its WAN connection into one of your primary router's LAN connections, and it'll get an IP, and your IoT devices will have their own little world to play in, and you don't have to worry about the precise behavior of a guest network...

Leo: Yes, good idea.

Steve: ...setting on your main router. I like that much better.

Leo: Yeah, that's a great way to do it. All right. Last one, my friend, comes to us from Scott T. in Cleveland, Ohio. He has a clever Harry and Harriet's Tip of the

Week: Steve and Leo, blah blah, Vitamin D, very low carb, blah blah. I was listening to the LostPass episode and heard Leo mention that women's razors tend to be less sharp than men's razors. My wife and I employ a cost-saving and sharpness-conscious solution to using Harry's cartridges. Oh, I'm not sure I like where this is going.

Steve: I know. Everybody now can guess what it is.

Leo: Yeah. I don't think Lisa's going to go for this.

Steve: Uh-oh.

Leo: I start with a new, fresh Harry's cartridge on Monday mornings. I then shave with it all week. After I've used it for a week, carefully making it less sharp, I put my week-old blade on my wife's Harry's handle. This way we each get a new, to us, blade each Monday, and we only pay for one new blade a week for the two of us, thus cutting the cost in half. That's why, by the way, we tell everybody to go to Harry's. It's inexpensive enough. I get a brand new blade each Monday. My wife never has to think about how sharp her razor is, or ever worry about changing blades. We've been using this approach for over six months now, and she's completely happy, as am I.

Well, Scott, there you go. I could try that. Lisa uses the Harry's and is perfectly happy with that. She doesn't mind a sharp blade. In fact, when we were told that, I think it was Harry's that told us that, she said, "What? I never heard of that." But Harry's told us the blades sold for women tend to be duller for some reason.

Steve: Well, Scott has a clever solution.

Leo: Good solution.

Steve: I just wanted to share it with our listeners.

Leo: Good solution. Not going to work in my house. Hand-me-down blades? Oh, hell, no. Oh, no, you didn't.

Steve: Lovingly desharpened.

Leo: Yeah, there you go. Well, that's different.

Steve: I've rubbed them on my face for a week for you, honey, and they're lovingly desharpened. Because we wouldn't want you to nick yourself.

Leo: Don't want you to get hurt.

Steve: They're safer now. So, yeah.

Leo: Yes, absolutely. Steve's at GRC.com. You might have figured that out by now. That's where you can find out more about SQRL, about SpinRite, the world's best hard drive maintenance and recovery utility. That's where you can find out about Vitamin D. You can listen to this show, too. He's got the podcast there, audio versions, along with transcripts, text transcripts, at GRC.com. We have the audio and the video at TWiT.tv/sn, and you can find it of course wherever podcasts exist. But Steve's the only one with the transcript, GRC.com.

We'll be back next week, next Tuesday, 1:30 p.m. Pacific, 4:30 p.m. Eastern time, that's 21:30 UTC. So please, tune in. Watch live if you can. But as I said, you can always download it later. Thanks, Steve. We'll see you next time on Security Now!.

Steve: Thanks, Leo.