News & Feedback Potpourri

Description: This week Steve and Leo discuss another new side-channel attack on smartphone PIN entry (and much more). Smartphone fingerprint readers turn out to be far more spoofable that we had hoped. All Linux kernels prior to v4.5 are vulnerable to a serious remote network attack over UDP. There's a way to prevent Google from tracking the search links we click (and to allow us to copy the links from the search results).

High quality (64 kbps) mp3 audio file URL: http://media.GRC.com/sn/SN-608.mp3
Quarter size (16 kbps) mp3 audio file URL: http://media.GRC.com/sn/sn-608-lq.mp3

They cover the latest NSA Vault 7 data dump nightmare, the problem with punycode domains, four years after the public UPnP router exposure, the mixed blessing of hiding WiFi access point SSID broadcasts, some miscellany, and then a collection of quick "closing the loop" follow-ups from last week's "Proactive Privacy" podcast.

SHOW TEASE: It's time for Security Now!. Steve Gibson's here. We've got a jam-packed show, lots of news. We'll talk about, oh, all that, what is that, a puny URL attack? We'll talk about routers. This is a great router exploit explanation from a couple of years back that is such detail, I just really enjoyed it. And of course Steve will even answer some questions. It's all coming up next on Security Now!.

Leo Laporte: This is Security Now! with Steve Gibson, Episode 608, recorded Tuesday, April 18th, 2017: News & Feedback Potpourri.

It's time for Security Now!, the show where we cover your privacy, your safety, and your peacefulness online with the man in charge, Steve Gibson at GRC.com.

Steve Gibson: I look a little blurry. Am I blurry?

Leo: Not to me.

Steve: Okay, good.

Leo: Did you put your contacts in this morning?

Steve: I just did. I was glad you were running a little bit late because of the Facebook thing. It's like, oh, good, I have till 2:00.

Leo: No.

Steve: But you ended up picking up some time.

Leo: I made up time.

Steve: It all worked.

Leo: We had the jet stream at our tail this hour.

Steve: This is just sort of a general News & Feedback Potpourri episode, which is the title, because a lot of things happened this week that were interesting. And last week's coverage of the proactive privacy, which was extremely popular with our listeners, generated a huge amount of feedback. And so we're going to talk - we'll deal with some of that feedback at the end of the podcast, but handle all of the news and some fun stuff and miscellany. And I think we've got another great couple hours.

Leo: Sounds fantastic, Steve. Are you pausing for me to do an ad so soon? I will.

Steve: Oh, well, you're right. I forgot to say, this week on Security Now!...

Leo: This week, what's coming up.

Steve: ...we've got another new side channel attack on smartphone PIN entry, and actually much more than that. It turns out also in a related research, but different, smartphone fingerprint readers turn out to be far more spoofable than we had hoped. And I've got a takeaway mission for our listeners. All Linux kernels prior to 4.5, which is very recent, are vulnerable to what is being characterized as a serious remote network attack over the UDP protocol. Nah, but we'll talk about that.

It turns out there's a way to prevent Google from tracking the search links we click on, which we discussed last week, how all of those Google search results are redirects, so Google can see what you choose. And that also allows us to copy the links from the search results, which is one thing that's been annoying me because we lost that ability at some point in the past. We've got the latest NSA Vault 7 data dump nightmare. The problem with punycode domains.

Leo: Oh, good. I'm glad you're talking about that one.

Steve: Yeah, yeah, yeah. Really, really interesting hack.

Leo: Yeah, yeah.

Steve: And which has been fixed, by the way, in all but Firefox now. Chrome's already updated. Edge, IE, and Safari were not vulnerable. But there is a switch that Firefox users can flip in order to protect themselves. Four years ago we discussed the public exposure of the Universal Plug and Play protocol. Now, a little more - it was like January, I think it was, of 2013. We now have the first full public disclosure from the guys who found it about exactly what that was and what the consequences are. We're going to take a close look, thanks to a question from a listener, about the mixed blessing of hiding WiFi access point SSID broadcasts. And then, as I said, some miscellany, and then a collection of "closing the loop" follow-ups from last week's Proactive Privacy podcast. So yabba dabba.

Leo: Sounds good. Yabba dabba do.

Steve: So our Picture of the Week.

Leo: Oh, I love it. Oh, my, my.

Steve: Well, for those who are listening and don't have the benefit of seeing it, this shows a keyless entry keypad of transparent, probably illuminated in the evening with a green LED - I'm sure I've seen these before.

Leo: Oh, yeah.

Steve: It looks like it's a garage door opener keyless entry keypad. And in this case it's two columns of five buttons with an ENTER. And the ink, the legend of the first four is completely rubbed off. I mean, you can barely see a 2. But the 1 and the 3 and the 4 are completely missing; whereas the 5, 6, 7, 8, 9, and 0 don't look as if they've ever been pressed.

Leo: Nobody, yeah, unh-unh, never, unh-unh.

Steve: And so, let's see, what could this person's PIN possibly be?

Leo: Now, there are a few permutations. I mean, that's probably 1234. But it would be 1342. How many would that - 16?

Steve: There are 24.

Leo: Twenty-four.

Steve: So assuming that it's a four-digit PIN, which it sure looks like, the first could be any one of four.

Leo: And it's ain't 1111 or 2222.

Steve: Right.

Leo: It's got to have all four.

Steve: Because they're all rubbed off, exactly. So the first one could be any one of four. Then the second one could be any one of the remaining three, so that's 12. Then the third one could be either of the others, which would bring us to 24. And the fourth one is the remaining digit. So if it's one of each digit, and this looks like it is because all four of them are rubbed off...

Leo: Oh, yeah, they're equally rubbed.

Steve: ...it's one of 24. So you could - and a garage door entry probably doesn't have a lockout because then you couldn't get into your garage, if you fumbled. So my guess is it would take you maybe 30 seconds, maybe a minute to get in.

Leo: Unless, and somebody in the chatroom said, they sell them this way.

Steve: That would be brilliant.

Leo: Wouldn't that be brilliant?

Steve: Yes.

Leo: I would buy it that way. Wouldn't that be funny?

Steve: That would be brilliant. What you want is you want to buy a used keyless entry pad like this, and then reprogram it with digits not rubbed off.

Leo: Right, right.

Steve: And nobody will ever - they'll spend all their time thinking, why didn't that work? I've tried all 24 combinations.

Leo: What did I miss?

Steve: Wonderful, wonderful. So whoever it was who sent me that, thank you very much.

Leo: Neat.

Steve: Okay. So we're constantly talking about ways that - essentially new, innovative side channel attacks. We've talked about how PCs, you know, lights can flicker, hard drives can make noise, speakers can make noise, of course WiFi signals or RF emanations and so forth. We talked a while ago about a remote camera looking at somebody entering their PIN, but unable to see the face of the phone, just looking at the relative motions of their hand, and how that was enough to dramatically reduce the search space required to determine what the PIN was. Well, some researchers at the School of Computing Science at the Newcastle University in the U.K. did some research and developed a JavaScript embed, essentially, a JavaScript page that was able to ascertain where the user was typing on their screen in order to determine what PIN they had entered.

So the abstract reads, and then we'll discuss it, it says: "In the first part of this paper we propose PINlogger.js, which is a JavaScript-based side channel attack revealing user PINs on an Android mobile phone. In this attack, once the user visits a website controlled by an attacker" - and it's important to understand for everything that follows, that's all you have to do. No privilege, no popup, nothing special. You just go to a web page, you know, not an app, just a web page, where somebody loads this JavaScript on your browser.

"The JavaScript code embedded in the page starts listening to the motion and orientation sensor streams without needing any permission from the user. By analyzing these streams, it infers the user's PIN using an artificial neural network." Well, yeah. When I read that the first time, I thought, okay, not obviously a biological neural network. Yes, it's artificial. "Based on a set of 50 four-digit PINs, PINlogger is able to correctly identify PINs in the first attempt with a success" - so they've narrowed the search space a bit to make it a little bit easier for them - "with a success rate of 82.96%, which increases to 96.23% and [then finally] 99.48%" - so almost 100% - "in the second and third attempts, respectively. The high success rates of stealing user PINs on mobile devices via JavaScript indicate a serious threat to user security," they write.

"In the second part of the paper, we study users' perception of the risks associated with mobile phone sensors." They write: "We design user studies to measure the general familiarity with different sensors and their functionality, and to investigate how concerned users are about their PIN being stolen by an app that has access to each sensor. Our results show that there is significant disparity between the actual and [the user] perceived levels of threat with regard to the compromise of the user's PIN. We discuss how this observation, along with other factors, renders many academic and industry solutions ineffective in preventing such side channel attacks."

Okay. So here's the technical background. All of the most popular browsers - Chrome, Firefox, Safari, Opera, and Dolphin on the Android platform - support these functions and enable this PIN harvesting by default. So any web page we visit can perpetrate this attack. Geolocation data has previously been identified as a clear privacy concern, so

modern web browsers ask their users' permission before returning this information to a web server. And in fact I frequently now see, and I imagine our listeners have, our browser producing a popup when we visit a website, saying this website would like to know your location. Which really annoys me. I don't think I've ever said yes; you know? And so typically you can say okay to allow, or cancel to decline.

But at this point in time, JavaScript code running in a web page does not require any similar permission to access a huge range of other sensor data, such as in this case device motion and orientation. There is no notification while JavaScript is reading the sensor data stream, which allows browser-based attacks to be carried out covertly.

And our physics-aware users could imagine that, if you have essentially linear and rotational acceleration and position information, which is what our phones provide to, in a streaming fashion, an inquisitive web server, then as you tap the screen, the coordinates of your tap will produce a physical shift in the phone. If you tap left of the center line, you'll tend to produce a twist. And I don't know if that would be yaw or pitch or what in aeronautical terms. But you'll rotate the phone around its long access. If you tap high, you'll rotate the phone along its width access, you know, the horizontal access, and so forth.

And so you can imagine that, if you have a simple 10-digit pad or 12-position pad, that each one of those taps will produce a distinctive transient in the linear and rotational acceleration axes of the phone. And that's picked up by the sensors and dutifully streamed back to a website that is monitoring your phone's instantaneous orientation.

So this is all based on, as many of these problems that we're seeing arise, a W3C specification that was, you know, I'm sure it was generated in the best of interests, except one has to wonder, this is very much like we were talking a few months back about the battery, the state of the battery charge and how, if that was a high-resolution feedback, as it initially was, and then when it was recognized to be a privacy threat, they chopped some least significant bits off of it so that it would be less of a fingerprint. But the instantaneous charge on your battery was being reported and could be used to disambiguate you as you go from one website to the other. And of course last week was all about the challenge or the drive to disambiguate users and our privacy interest in remaining anonymous and ambiguous as we move around the web.

So the W3C spec for motion and orientation sensor data provides the device orientation, the physical orientation of the device expressed as three rotation angles in the device's local coordinate frame, meaning relative to the device itself; its linear acceleration, in X, Y, and Z Cartesian coordinates, relative to its own physical orientation, that is, to its own hardware; the device's acceleration including gravity, so that's going to give you how it's being held, if it's being picked up, if it's lying horizontal, if it's on its edge and so forth; and then, finally, the rotation rate around all three axes, the instantaneous rotation rate. So that's a wealth of information. And these guys streamed that in from contemporary smartphones. And with some limitation of the total search space, we're able to obtain nearly 100% guessing recognition just based on the information obtained from the user tapping on their screen.

So the question is, how available is this data? Turns out it's way more available than we would like. I mean, you'd probably want only the foreground tab that you are looking, I mean, if you wanted it at all, you'd want only the foreground tab that you were actively looking at on the foreground process, that is, your browser, while it has system focus, to be able to stream this data.

Unfortunately, in the case of Chrome and Dolphin on iOS, an inactive tab that includes

the sensor listeners has access to the sensor measurements. And in Safari it's worse. Safari allows inactive tabs to access the sensor data even when Safari is minimized, and if the screen is locked. So if you've visited a site - and, I mean, this gives me chills because pretty much I have the entire 'Net open in tabs under Firefox, but of course not on my phone. But that means that a tab you haven't visited for weeks, but you didn't explicitly close, could be sitting there continuously monitoring the physical movement of your phone.

And so of course we can extend this, too. These guys managed to figure out where on the screen you were tapping. But it could be useful for someone to know is the phone moving, to infer if you are in motion, if you are at rest. If the phone rings, does it suddenly move as it gets picked up? I mean, so you can imagine, again, we know that there are entities in the world that are actively interested in doing everything they can to penetrate privacy barriers. So this makes that very easy.

In the second part of their study, they asked people, they surveyed people, like what was their awareness of the sensors in their phone? They found 100% recognition of the camera and the touchscreen, which is not surprising because those are things that people most often explicitly and deliberately interact with; 97% recognition of microphone, Bluetooth, and GPS; 93% recognition of WiFi; 83% recognition of fingerprint. And then in decreasing order of awareness was rotation, orientation, ambient light, motion, touch ID, device temperature, NFC, barometric pressure, ambient temperature, proximity, gravity, accelerometer, gyroscope, magnetic field, ambient humidity, ambient pressure, and then, lastly, the hall effect sensor, which is a magnetic field awareness. So I guess the point is...

Leo: How many people were aware of that? I mean, I didn't even know that existed.

Steve: It was down in the way...

Leo: .0001.

Steve: Yeah, it was. But so here we have this dilemma. The W3C is, with HTML5, is standardizing the access. And all of the browser vendors want all the bullet points filled out on the comparison chart. Like, oh, yeah, we have that; we have that; we have that. So unfortunately, only something like our physical location, presumably GPS, although Firefox on my desktop is asking me where I am, so it doesn't have any GPS on this desktop machine. So I'm not sure what that would be sending back, if I let it. But I just always say no, you don't need to know where I am, some random site. It's like, it's just being nosey. It has nothing to do with location.

So I think it's worth noting that this stuff is being done. And I'm so thankful that we've got researchers like this who are bringing this to light because, in the same way that recognizing that GPS was being fed back and then the browsers proactively said, okay, that's going a little too far, we're going to get affirmative permission from our users before we allow that to get sent back to a website requesting it. It's not clear to me, I mean, I guess you could envision a scenario where a website might get some benefit from knowing the instantaneous positioning of the app.

I mean, well, for example, how about a marble race, you know, a web-based marble maze game, where you're tilting the phone, and the marble is rolling around. Well, you

could do that all in JavaScript, and it would need access to the phone's orientation sensors on the fly. So that was probably - you could imagine that as being the justification for that. But in that case, it wouldn't be cumbersome to say, you know, to just click a button and say, yes, I give this page my permission to know how I'm holding my phone. Otherwise, it's very clear that this could be used.

The clear takeaway is, on Android devices that have this, and I would argue that iOS really needs to do this also, if they're streaming this as they apparently are, you really want a keypad that scrambles the digits. It's not nearly as convenient because you can't memorize it with two spare neurons and never have to think about it every time you unlock your phone. Of course, everybody's now using their fingerprint reader, which we're about to talk about. But you definitely want a scrambled keypad option. You know, turn that on if your phone offers it. And if you don't have a fingerprint reader, or maybe even if you do, after you hear this next story about fingerprints, but you definitely want to make yourself look for the digits which are going to be randomly arranged on the pad surface every time so that nobody can use any of these techniques in order to figure out what you're doing. Yeah, the hackers are clever.

Okay. So it turns out some other research conducted by New York University and Michigan State University found to their chagrin that the fingerprint readers that we currently have in our smartphones are not nearly as robust as we want them to be and assumed they were. I'll share just the first paragraph of their abstract, and then we'll discuss it.

They wrote: "This paper investigates the security of partial fingerprint-based authentication systems, especially when multiple fingerprints of a user are enrolled. A number of consumer electronic devices, such as smartphones, are beginning to incorporate fingerprint sensors for user authentication. The sensors embedded in these devices are generally small, and the resulting images are therefore limited in size.

"To compensate for the limited size, these devices often acquire multiple partial impressions of a single fingerprint during enrollment to ensure that at least one of them will successfully match with the image later obtained from the user during authentication." And of course all of us who have trained our fingerprint readers know that's the case. We're told to, like, move our thumb around and position it differently and so forth, you know, and get the edges during successive reads.

"Further," they write, "in some cases, the user is allowed to enroll multiple fingers, and the impressions pertaining to multiple partial fingers are associated with the same identity (i.e., one user). A user is said to be successfully authenticated if the partial fingerprint obtained during authentication matches any one of the stored templates." And of course what we recognize is that would tend to reduce integrity of the search because we've broadened the search space.

"This paper investigates the possibility of generating a 'MasterPrint,' a synthetic or real partial fingerprint that serendipitously matches one or more of the stored templates for a significant number of users. Our preliminary results on an optical fingerprint dataset and a capacitive fingerprint dataset, which is what we have on our phones, indicate that it is indeed possible to locate or generate partial fingerprints that can be used to impersonate a large number of users. In this regard, we expose a potential vulnerability of partial fingerprint-based authentication systems" - which is what we're all using - "especially when multiple impressions are enrolled per finger. In their simulations, the researchers were able to develop a set of artificial 'MasterPrints' that could match real prints similar to those used by phones as much as 65% of the time."

Okay. So we all know, and Sherlock Holmes famously demonstrated, that a person's entire fingerprint is globally unique. But think about this. How unique is a much smaller portion of an entire fingerprint? How unique can a smaller piece be? And when I was trying to come up with a good analogy, I thought, okay, think of this as like having an ultra-secure 20-digit PIN, but only needing to provide any three successive digits from within that PIN. When you think about it, I mean, it's obvious to us that that's not going to be secure. But the 20-digit PIN looks, oh, wow, you know, no one's ever going to get that.

But, I mean, the analogy is that neither are our current technology capacitive fingerprint readers on our smart phones. They're not looking at our whole fingerprint. They're only getting a window of it. And in fact that's why we're told to do multiple impressions and move our finger around in order to create a larger virtual image of the fingerprint. And then, when we give it a partial print, it looks for it.

And so what's happened is we're seeing another instance where manufacturers' desire to minimize their users' inconvenience has prevailed. And so what they chose was - the decision was we absolutely want to minimize false negatives. We don't want to frustrate users by having them keep trying to unlock their device by putting their finger down and saying, no, we don't think that's you. No, we'd better look again. Oh, try it again. Instead, it pretty much works the first time, every time. But if we challenge how that's the case, it turns out that what we actually have is a tendency, through minimizing false negatives, to create false positives.

And after reading this study, it occurs to me that I've never attacked my own phone's fingerprint reader by asking the people I'm meeting to try unlocking my phone with their fingerprint. And I'm going to start doing that. Every time it occurs to me, I'm going to say, hey, I want to see if your fingerprint will unlock my phone.

Leo: Oh. Interesting test.

Steve: Yes. And I want to extend this to our listeners, and I will look for any feedback because, I mean, it's not something we typically do. I have no problem if somebody, if my best buddy or a server at a restaurant is successfully able to unlock my phone. I would like to know that. I'm not going to leave my phone with them again.

Leo: Maybe not the server at the restaurant. With friends.

Steve: Anyway, I'm really interested. I'm going to obtain as large a population study as I can and get some actual feedback on how resistant my own iPhone 6s is to being unlocked by somebody else because this suggests that we may be getting a false sense of security from our fingerprint readers. Yes, they always conveniently unlock for us. But to what degree have we sacrificed their discrimination in order for it not to be frustrating?

And Leo, one thing I've noticed, because for a while I'm sometimes having my meal out on a cold patio, temperature seems to be the largest impediment to my iPad, actually, in this case, unlocking. When my hands are cold, it doesn't like me nearly as much as when they're warm. So I never really worked to figure out why that was the case, but I thought that was interesting. That's the one thing I've noticed that seems to fool it.

Leo: Hmm.

Steve: Okay. So we've got a potential problem in all Linux kernels prior to the very recent v4.5. And, boy, if you look at the - there's a link, I think it's the Mitre link, the second link in the show notes, Leo. I think about halfway down they list the vulnerable kernels. And it's, like, all of them.

Leo: Oh, boy.

Steve: I mean, it's just like it goes - it scrolls and scrolls and scrolls and scrolls. Okay. So what was found - and this has been known for a while. It's been kept quiet until it could be deployed. If it's not - okay, so maybe it's the NIST link, the N-I-S-T link.

Leo: Yeah. I'll go back, yeah.

Steve: One of them scrolls, and it's just like, whoa. Okay. So what was found was a subtle problem that has long been present, like always been present in the Linux kernel. It's in the UDP, the kernel's networking stack, the udp.c file. And this is the worst of the worst, at least potentially. It is a network remote code execution over UDP.

Now, what mitigates this maybe is that it is in an option of the way UDP datagrams are retrieved by userland code. So Linux utilities running in user space need to be accepting UDP packets. The normal way you do that is you just say, give me the next UDP packet. You're able to either check to see if one's available, or you're able to wait for one. And either way you then say, okay, thank you very much. You provide a buffer, and the kernel copies the newly arrived packet into your buffer, and now you have it.

Okay, well, there is an option called MSG_PEEK, M-S-G underscore P-E-E-K, the MSG_PEEK flag. You're able to set the MSG_PEEK flag which says, here's my buffer. I want to peek at the most recent received, like next to be delivered UDP packet, but don't take it, meaning leave it available. And so there are various reasons you might want to do that. But they're uncommon. That is, the point is that normally what you want to do is just give me the next packet. Now give me the next one. Now give me the next one. You don't often want to say "I want to peek at it, but leave it there."

So at the same time, a string scan of the Linux source tree reveals hundreds of instances where the MSG_PEEK flag is included in a UDP API call. So somebody's doing them, for some reason, a lot. I saw pages of them in the glibc library, and I didn't go any further. So it is considered a critical vulnerability network attack, easy to do, meaning low complexity. No privileges required. No user interaction required. So potentially very worrisome. Google has it fixed in their April 17th Android security bulletin. It affects their Pixel, the Pixel XL, and the Pixel C; the Nexus Player, the Nexus 5X, 6, 6P and 9; the Android One and Android. And of course, as we know, it also affects all other previous Android non-Google phones that may never get patched.

What we don't yet know, and there's nothing that I could find, I spent some time digging around, I couldn't find any clear definitive example of an exploit. There was some mention in one form of wget using it. But wget is normally used for fetching files over TLS or just HTTP or TCP connection, not so much over UDP. There is DTLS, which is the

TLS over UDP. As we know, TLS normally runs over TCP, but there is a way to run TLS over UDP called DTLS. And that's present in OpenSSL. But again, you need to have a userland app doing this, and then an attacker able to deliberately malform a packet which a vulnerable version of Linux, which they all are, could then use.

So it is a critical remote code execution. Everybody's running around. And of course here's the problem is that there are, as we know, many Linux-based devices which who knows when they're going to get fixed. We've got smartphones, non-Google Android devices, routers, televisions, DVRs, higher end IoT devices of all kinds. They've all got versions of Linux where this is vulnerable. So while there isn't any, like, we don't know of a zero-day event involving this. We have a patch for the kernel. But there is zero doubt that would-be attackers are right now carefully examining every instance of Linux's user-space code where the MSG_PEEK option is used.

And again, a simple string search through the source tree finds hundreds of them. And bad guys are looking for a way, some way to leverage this. So hopefully routers don't have UDP exposed. Of course, if they've got Universal Plug and Play exposed, we'll be talking that in a minute or two, then that is a UDP-hosted protocol, so that would be a problem. So my sense is we're probably going to be talking about this in the future and in a worrisome fashion. I mean, maybe it's the case that this option is in fact obscure enough; that the collision incidence, the likelihood of it actually being exploitable is rare.

But there's no, I mean, this is just - this is nirvana for attackers because, unfortunately, our ecosystem today has got a ton of Linux-embedded OS, all of which are presently vulnerable, and few of which are being dynamically updated. And if there's an instance where the MSG_PEEK was used in a common fashion, and I haven't seen any indication of that yet, but that would allow potentially - in the worst case it would crash or cause a kernel fault or a reboot. I mean, and as we know, that's the way you begin. And then afterwards you figure out how to execute code that you have also supplied in that same UDP packet.

So clearly the takeaway is, well, first of all, somebody would need to be able to send your Linux machine a UDP packet. If they are able to do that already, you've got problems because you shouldn't have UDP, I just don't see a good reason for having UDP exposed. Now, maybe, if you had like an OpenVPN server, and you do want UDP protocol to be used as the carrier for your VPN tunnel because that's far more effective than using TCP. But the question would be does OpenVPN ever do a MSG_PEEK on the UDP traffic it's receiving? Those are the kinds of things we'll be finding out in the future.

So I would say the typical user, behind a router that itself doesn't have any UDP ports open, is probably okay because it does require a bad guy to get a UDP packet onto your kernel network stack and for it to be serviced by an application which is using MSG_PEEK in this way, and for the packet to be an exploit. Again, it's such a juicy opportunity. I'm afraid we haven't heard the last of this one.

So John Gruber is a well-known blogger. He has the Daring Fireball blog. And one of our listeners shot a note to me following from some of my comments about the Google search. And John's posting, he said about DirectLinks: "Safari Extension That Circumvents Google and Facebook Link Redirects." And he says: "Great little Safari extension from [looks like] Canisbos: The extension circumvents certain techniques used by Google and Facebook to track link clicks." He writes: "When you click a link in Google search results, Google" - and this is what I did not know - "Google uses JavaScript to replace the actual link with an indirect one, which they use for click tracking. Google then redirects the browser to the actual destination after logging the click. DirectLinks" - which is the name of the Safari extension - "disables the JavaScript that replaces real links with

indirect ones, so that when you click a search result link, Safari goes straight to the destination.

"If you've ever tried dragging and dropping a URL from Google search results and getting a Google redirection URL instead of the actual URL you wanted," and he says, parens, "(and Google's JavaScript will show the actual URL in the status field if you hover over the link so it's impossible to tell that's what's going to happen), this extension," he writes, "is for you. There are obvious privacy benefits, as well."

Okay. So the idea, I just assumed that Google was delivering a page to me that had their redirect links in the body of the page, rather than JavaScript doing it on the fly. So I thought, oh.

Leo: Well, think of it as like a URL shortener; right? It's very similar to that. That way, I mean, why would they modify the resulting - you don't want to modify, like if go to TWiT.tv, you don't want to modify my TWiT.tv page. You just have a redirect. They've always done that. And, by the way, that's how they monetize. And if you like Firefox, that's what pays for Firefox because Google then can give Firefox money. So while you can do this, gosh, it seems - all right.

Steve: Okay. So I'm not sure we're...

Leo: If you're worried about your privacy, I understand that. Well, no, if you look, if you click a link on a Google search page, it's not a direct link to that page. It's a link to some JavaScript, some tracking JavaScript; right? That's what they're talking about.

Steve: No.

Leo: Oh, I misunderstood.

Steve: JavaScript on the page rewrites that link on the fly. So, and I had talked about this a couple weeks ago, where for example, like when I'm assembling notes or wanting to send someone a link, I'll right-click on the link, and even though it shows me the search result, I end up with this ridiculously long gibberish.

Leo: Right.

Steve: That comes from Google. Well, it turns out...

Leo: And that's why, that's how they track it.

Steve: Correct. So they're tracking what links I'm clicking on in the search results they provide.

Leo: Right.

Steve: And in the process, they're obscuring the link. That is, I'm unable to right-click and save the link. Anyway, so the point is that, for our listeners, if not for you, Leo, it is possible to trivially disable this for all browsers because uBlock Origin is, as we remember, is a very facile web filtering system. So I have uBlock Origin running, and it's sort of a "set it and forget it" tool. But so when I realized from John's column that this was JavaScript, I thought, oh, that's interesting. It would be much more convenient for me to have real links in Google search results, rather than redirect links.

So, and I have here a picture of the screen in the show notes, where because the UI, I had completely forgotten how to use this thing. I mean, we talked about Gorhill, who's the author of uBlock Origin. In fact, if anyone is interested, it is Security Now! Episode 523, which we recorded on the first of September, 2015, titled "uBlock Origin." And so in the advanced view, where it gives you a lot of granularity, there are these two, believe it or not, unlabeled columns. And again, this is not friendly. But it turns out that, if you click on the right-hand column, which is the per-site rules, versus the left-hand column, which is the global rules, you click in the right column for first-party and third-party scripts and just turn them red and then refresh the page. Everything works, except you have real links rather than links where Google tracks what links you follow from their search results.

So for anyone who's interested, Safari's got the extension that John Gruber mentioned. But anyone using uBlock Origin on Firefox or Chrome can easily just say I want to turn off scripts in Google. Now, this is per site. So, for example, that's www.google.com. With that in place, I was then using drive.google.com, and any other property that doesn't collide with www.google.com will be okay. This may be too broad because this is going to kill those scripts on anything that matches www.google.com. Whereas presumably the Safari extension is more specific.

But I just thought it was interesting. I didn't realize that JavaScript was embellishing these links, and that just by saying no to JavaScript on that page, I mean, you know, we used to run with NoScript on all the time. And so in fact maybe that's why I was never seeing this before is that I never had script running on Google, and I was getting physical links rather than redirect links. And I just didn't appreciate the difference until I switched over. And actually I gave up on running NoScript because too many things need it these days.

So we have another dump from the seemingly bottomless pit of the NSA WikiLeaks Vault 7. And this one generated another round of breathless press coverage last Friday on April 14th, mainly focusing on something called EternalBlue. An example of the press coverage was one site that said: "The Shadow Brokers, an entity previously confirmed to have leaked authentic malware used by the NSA to attack computers around the world, today released another cache of what appears to be extremely potent, and previously unknown, software capable of breaking into systems running Windows. The software could give," they write, "nearly anyone with sufficient technical knowledge the ability to wreak havoc on millions of Microsoft users."

And they continue: "The leak includes a litany of typically codenamed software implants with names like OddJob, ZippyBeer, and EsteemAudit, capable of breaking into and in some cases seizing control of computers running versions of the Windows operating system earlier than the most recent Windows 10." And then they conclude: "The vulnerable Windows versions ran more than 65% of desktop computers surfing the web

last month" - and I'll note, uh-huh, but not this month - "according to estimates from the tracking firm Net Market Share."

So the good news is, as is so often the case, I mean, what we've always seen, both in the original Snowden leaks of the CIA documents and now here, many of these things are very old. And so many of them, because they're just sort of a repository trove, which could be useful if they're encountering somebody, if law enforcement or intelligence is encountering someone with an old, long not updated Windows machine that hasn't been fixed, then some of these things could still apply. But Microsoft apparently had access to this before its disclosure on Friday.

And so the March update - which remember that February got canceled. First it was postponed, then it was canceled. So that big March update, which is MS17-010, that fixed these things last month. So anybody who has been keeping their machines updated and is using a Windows which is still receiving updates, already has these things fixed. So there was one called EternalBlue which was fixed last month, as was EternalRomance and EternalSynergy. And I sorted these in order of when they were fixed.

There was one called EskimoRoll which got fixed in 2014. EmeraldThread got fixed in 2010, running backwards. EducatedScholar got fixed in 2009, and Eclipsed Wing in 2008. So Microsoft's on top of this. They did release a TechNet blog posting where they explained, of what was released, when these were fixed. And so even the worst were all fixed last month, and many fixed years ago.

So this is sort of typical of what we see being disclosed. And while, yes, if this had been released in January, and Microsoft hadn't released a fix, then this would have been a problem. But one way or another, Microsoft did fix these things, even the most recent of them, in the March patch cycle. So as long as your machines are caught up to date, or are up to date, you're probably not in bad shape.

Okay. So we're all familiar with ASCII, the American Standard Code for Information Interchange, which is an eight-bit code of which really seven tend to be used, which expresses upper and lowercase, the Latin character set, the digits and the special characters. Basically, what we type on our - what an English speaker types on a standard keyboard is the ASCII character set. But as we know, there are way more characters than 127, which is what you're able to get with seven bits of binary, excluding zero. So Unicode was created, which is a multi-byte encoding which, for example, well, which is a huge encoding space. Now we've got 16 bits, which is, as we know, 64K possible characters. And so it's got everything in there. It's got all the different languages, and now it's got emojis and all kinds of other stuff.

Well, the question that browser vendors faced was, okay, the transport, the traditional transport for our communications is a byte. It's not a word. That is, it's eight bits of which we really only use seven. It's not 16. So how do we - and this, of course, is also the case for the domain name system. The whole DNS system is oriented toward non-Unicode characters. So how can we have a domain name which has got multilingual characters, accented characters, the things that non-English speakers have every right and reason to want. They want the accent aigu to show on their name if their name has one, or if their domain does.

So this weird encoding was created, and the name is a play on Unicode. So we change that to "unycode," and then we change it to "punycode," P-U-N-Y-C-O-D-E. And it is a way of encoding in ASCII, that is, in seven-bit printable standard ASCII, to encode Unicode characters. And there are a number of examples. Now, I couldn't remember - so I've been aware of this problem for a while. And I don't know, it apparently only came to

light and got a lot of attention in January. But it feels to me like I've been aware of this as a problem for a lot longer than that. So it may well have been that it just received attention.

So, but here's the point. It is possible to leverage punycode to create domains which, when displayed by our browsers that understand punycode, look like different domains. And in fact on Firefox today, no other browser is currently still vulnerable. Chrome was; but Google, because they have the ability to immediately fix things, mine this morning was already fixed. Chrome 59 fixes this. In the show notes I have links that you can play with to some of the coverage. There's a great link with full coverage of the problem.

The only people today who really need to worry are Firefox users because Mozilla apparently, for reasons I don't understand, is still on the fence about what they want to do with this. But the good news is there's a switch which, in that about:config monster list, that anyone can set to true. And I would argue everyone should. Although now Chrome is fixed. IE, Edge, and Safari have not been vulnerable. That means that it's only Firefox users that would be fooled.

So here's the danger. In this demo - I have a link in my show notes. It's HTTPS, and that's a key, https://www. And then this punycode is all lowercase, xn--, and then 80ak6aa92e.com. If you put that into Firefox - again, https://www.xn--80ak6aa92e.com, where you get taken is a site that is not Apple.com, but your URL says https://, it's got the green padlock, everybody's happy, and the domain that's displayed is www.apple.com. Because that punycode is Unicode encoding for the same characters, Apple.com. And so somebody registered the punycode domain, that is, the guy that really brought this to everyone's attention registered the punycode domain equivalent in Unicode for Apple.com and put up a site there. And when you go to the crazy punycode link, it shows you Apple.com.

So clearly, had this person had malicious intent, he could have sent email where the non-visible URL was this punycode, you know, looking like it was email from Apple telling you to do something. You would click on the link. You would go to what absolutely looked like Apple.com with a valid TLS certificate showing the happy green padlock, everything looking perfect. You'd inspect the URL: www.apple.com. Absolutely, you know you're at the right place. Except you wouldn't be.

Now, if you - let's see. I didn't think to look at the certificate. I should have looked at...

Leo: It reflects the "xn" site, not Apple.com.

Steve: Okay, good. Right. It would have to, yes, good. So, okay. So nobody needs to worry about this. So this was a very potent phishing attack because any inspection, any careful inspection would show that you're exactly where you think you should be, yet you would not be. And short of looking at the certificate, there would be no way to know.

Firefox is vulnerable to this now. So this works now under Firefox. If you go to about:config, which brings up a bazillion options, and put into the search bar, which is the only way to find anything among those options, just enter "puny." There's only one item that matches that string. And that's network.idn_show_punycode. And you want to change that to true to show the punycode, rather than have Firefox conveniently show you the Unicode. You don't want the Unicode. You want the actual ASCII punycode.

And so once you do that - and in the show notes I've got two links you can use to verify,

to test and verify that your Firefox is no longer going to be confused. One of them is the Apple.com, and another one is Epic.com, which is another site that was set up in order to demonstrate how spoofable this feature of browsers of turning punycode into Unicode could be, and how it could be leveraged. So this is, again, just like, whoops, a nice feature, but clearly subject to abuse. And luckily we got this fixed, hopefully before anybody got caught out.

This is real quick. There was an interesting article by Joseph Cox, writing for Motherboard under the title: "Your Government's Hacking Tools Are Not Safe." And this follows from what we've been talking about. He wrote: "From Cellebrite to Shadow Brokers to the CIA dump, so many recent data breaches have shown there is a real risk of exposure of government hacking tools. The hackers will get hacked."

He writes: "Recent data breaches have made it startlingly clear hacking tools used by governments really are at risk of being exposed. The actual value of the information included in each of these dumps varies, and some may not be all that helpful in and of themselves, but they still highlight a key point: Hackers or other third parties can obtain powerful tools of cyberespionage that are supposedly secure. And in most cases, the government does not appear to clean up the fallout, leaving the exploits open to be reused by scammers, criminals, and anyone else, for any purpose."

So as we know, this was exactly my own takeaway a month or two ago when we first covered these Vault 7 leaks because we've seen a clear pattern that our intelligence and law enforcement agencies are, despite their best efforts, unable to keep their own secrets. And I do not blame them, per se, because no large human-based organization can perfectly keep secrets. We just don't have the technology for doing that today. All of the evidence continues to demonstrate that this is true. And the CIA, the NSA, and FBI are not exceptions.

And this, of course, as I have said, in turn argues against any form of a mono-key technology that would be legislated for access to encrypted communications. I expect that companies will likely be required to modify their encryption technologies on a company-by-company basis, as they comply with legally issued court orders in order not to be held in contempt of court and probably heavily fined. But I think it's crucial that we maintain a heterogeneous encryption environment and not force all companies to provide the equivalent of a universal generic encryption backdoor, which would allow law enforcement to unilaterally obtain access to virtually, well, all otherwise lawfully encrypted data.

So that's the takeaway that I hope our listeners and legislators will get. I mean, they have strong, you know, those who would argue against letting law enforcement have what they want here have, I think, a strong weapon in their arsenal now because we keep seeing instances where our own intelligence services are unable to keep their secrets safe. And again, I don't blame them. They're large. They're bureaucratic. They've got a ton of people. To do the job, lots of people have to have access to these tools. That's going to happen. But that also says we need to distribute the danger and not concentrate it into any kind of a single golden key, as James Comey famously said he wanted the FBI to be given.

A little more than four years ago, back in January of 2013, researchers from DefenseCode responsibly revealed the existence of a remote root access vulnerability in the default installation of some, at the time, Cisco Linksys routers. Our podcast listeners who have been with us for a little more than four years will recall that I immediately responded, because this is bad, I immediately responded by adding a test to ShieldsUP!, which is there to this day, been there for four years. And today that test for publicly

exposed Universal Plug and Play service has found 51,854 positive exposed results.

And the ShieldsUP! system has always maintained a 1024-deep, a 1024 IP-deep most recently used list, an MRU list, specifically to avoid redundantly multi-counting repeated tests. So if somebody uses ShieldsUP! multiple times, or if somebody uses the Universal Plug and Play, as long as their public IP is still within the most recent 1024 IPs, which probably keeps it there for several days, I'm only counting them once. So that 51,854 is a pretty good number. It doesn't mean that the same person coming back month after month wouldn't get multiple counted. But on the other hand, that's probably what we want. We want to see that there's still that device which is exposed.

So I have a link in the show notes here to the whitepaper that DefenseCode just published. They waited four years. And anyone would have to agree that that's plenty of time. What we now have is full disclosure. And Leo, this may have been what - I'm sure I remember seeing all those Linux versions. But go halfway down, and you'll run across pretty much the who's who of routers. I don't know if there's any router not on that list. It's just amazingly comprehensive. And I think at the time we discovered, if I'm recalling correctly, that it was some demo code, some demonstration code in Intel's UPnP source which was never meant to be deployed; but it was just like, okay, here's a sample of how you would do Universal Plug and Play. And they just - the router vendors put it into their firmware and used it. And it was like, okay, wait. We never meant for that to be used in production.

Okay. So just to back up a little bit, remember that Universal Plug and Play is the protocol which Microsoft and Intel together created to allow sort of zero-configuration hole-punching through NAT routers to allow devices in the network to create a public presence, even though they're behind a NAT router. Because they wanted it to be literally plug-and-play, meaning it just works, there's no password, there's no authentication, there's nothing you have to do.

So this is part of the problem with this protocol. I mean, it achieves what it wants, but it's dangerous. At the same time, it was never meant to be bound to the public WAN interface. It was only meant to be a LAN-facing protocol that would allow clients in the LAN to perform some autonomous configuration of the router. For example, the Xbox is a big user of this. Suddenly you're just on an Xbox network, and you didn't have to do anything because the Xbox used the UPnP on your router in order to map a bunch of ports from the public Internet onto itself.

Okay. But that's the service facing the LAN that does that. It should never be exposed. So what we have today is the complete document on the vulnerability. And Leo, for the last...

Leo: I've been scrolling. I'm only through the N's.

Steve: Exactly.

Leo: By the way, I didn't know there was a - oh, it's not Nerf, it's Neuf. I thought there was a Nerf router. Yeah, it's amazing.

Steve: Again, there's, like, nothing that didn't make the list. So the takeaway for our listeners is absolutely positively make sure - and I guess this would apply to people who

didn't hear the podcast on January of 2013. So, and I don't have the podcast number here. But if you, for example, if you Google "GRC UPnP," "Security Now! UPnP," you'll find it [SN-387]. So anyone since then may not have heard that you absolutely need to turn off Universal Plug and Play on the WAN interface. If it's on, you probably have the ability to turn it off. If not, then you want to make sure you're running the latest firmware for your router, which is always a good idea, and it will probably have it shut down. Cisco dragged their heels, back when it was announced. First they said it had already been patched, but then it turns out it hadn't been. And then they ran around and created a series of Linksys firmware updates. And we covered all this back at the time it was happening.

But now the world knows exactly what the remote root access vulnerability is. And any devices which are still to this day vulnerable - and one thing I don't have is a graph over time. I could create one if it was important because I have a log, an anonymized log, of every event where a positive result was found. So, for example, I could create a graph showing, if there was, a decrease in the amount of vulnerabilities found for tests taken.

Anyway, the point is GRC.com, ShieldsUP!, right on that front page, after you click through the entry page, is a big orange-looking icon thing, which we put up immediately four years ago. Click on that. We'll tell you whether your router has Universal Plug and Play exposed. And nearly 52,000 people have had this thing say, yes, you're exposed. Hopefully they then figured out how to turn that off on the WAN side, and then they tested again, and it said, nope, you're no longer exposed. The bad guys now know, they now have every detail about this as a consequence of DefenseCode's full disclosure. They waited four years. Nobody, I think, could say that was too soon. But we do need to make sure, those who are security conscious, that you've got Universal Plug and Play turned off, especially now.

Some guys at Princeton, three guys at Princeton and one person at Stanford, actually Jonathan Mayer, who we haven't heard much from recently, published a paper titled "The Future of Ad Blocking: An Analytical Framework and New Techniques." It's long and verbose, 17 pages of academic review of the legal and technical history and foundation of adblocking, the fact that a sponsor tag is supposed to be placed on ads, they say, making it clear whether inserts on pages are ads or not.

And they developed a theoretical add-on which defeats the adblocker blockers. And it's sort of obvious, but they took the time to do it. This doesn't solve some of the problems that an adblocker does in that their solution doesn't trigger adblocking awareness on the site you visit because it still fetches the content. Yet when it sees that it is an ad, and they have some heuristics that they use in order to make that determination, they alter the document object model to put a translucent overlay over the ad and label it as an ad.

So you're able - so your browser still fetches the ads. But they're sort of toned down. Their contrast is reduced by being placed underneath a translucent white cover so that you can sort of see through it if you want to. And in their page they demonstrate some examples of this. You can see through it if you want to, but it's not nearly as much in your face as would otherwise be. And the sites are still generating the revenue from the ads. The users are being less assaulted by them. There is of course the problem that malvertising still could be present, and the advertising tracking that some people object to would still occur. But it visually tones them down.

So it's not clear to me what's going to go, you know, what's going to happen. They did develop a - I don't remember if it was Chrome or Safari. I think it might have been a Chrome plugin. I think it was a Chrome extension which highlights in a red box, it doesn't do this translucent fading, it just draws a big red frame around everything that they think

is sponsored content, to demonstrate the technology without actually being as active as they would propose their system could be, if that's what you wanted. My sense is, you know, every so often as I'm surfing around, I've got uBlock Origin on, as I mentioned before. And the pages I visit are - they seem to have ads on them, but they're not going crazy. So maybe it's using that ad network, the EasyList, where there's been some compromise made about how assaultive the ads are going to be visually.

Every so often I'll run across a page that brings up a notice and says, oh, you're using an adblocker. If you'd like to use our content, please turn it off. And I go, okay. I turn it off, and then I use the page. I think that's, at this point, a reasonable compromise. And one can imagine that a hybrid of this visual blocking technology could be adopted where you normally run with an adblocker on. But if a site objects to that, then you visually block the ads, but still pull the data and have your browser sort of semi-display the ads, but not in a way that is too obtrusive. So an interesting compromise.

John Schneider sent me a tweet asking a question that comes up a lot, so I wanted to handle that before we get into the rest, the second part of this, which is the feedback from last week's Proactive Privacy podcast. He said: "Steve, would you please address the pros and cons of hiding SSIDs on home routers in a future SN podcast?" He says: "iOS 10 warns this is a bad thing."

So again, I get this, people are asking the question all the time. And Leo, I'll bet on the weekend on The Tech Guy show you're probably having security-conscious listeners asking, too. We've talked about this in the past and how it is mostly cosmetic, and it does not provide much security. Turning off the SSID broadcast from your router prevents its automatic discovery and listing in anybody's list of available access points who is within range of your router. So all your neighbors, you know, you and all of your devices won't see it, and none of your neighbors will see it. That is, in that list, in the presented list of, you know, these are access points you could use.

But with that comes an inconvenience and actually some - not much actual gain in security, and some behavior you may not be aware of. First of all, the inconvenience is that you must then manually tell your computer, here is the SSID that I want, and the password. So you have to manually give it that information because you can't select it from that list because it's not in the list. Also, many systems will then not remember that. That is, you know, if you have the "automatically connect to routers I have previously connected to" option, which is very convenient if you routinely move between various locations, that feature disappears, and you're forced to manually reassociate, essentially, with the access point every time. So all OSes - okay. So there's that.

But then all OSes that connect automatically broadcast the SSIDs, whether hidden or not, of all the networks they have connected to in the past. So while the non-broadcast access point's SSID is not being broadcast to anyone new, it will then be broadcast by all devices that once connected to it. So they're blabbing this hidden SSID, even if the access point isn't. And all over-the-air traffic contains that non-broadcast SSID. So anyone actively sniffing the radio traffic will still see the access point's SSID if any device is currently associated with that access point.

So typically you've got all kinds of IoT things in your home that are associated. Well, that means that anyone actually looking at, you know, sniffing Channel 11, which is probably what your system is using, on your devices, but they can of course also see whatever channel you're looking at. If they're promiscuously sniffing your WiFi traffic, they can look at the packets and discover the SSID that everybody's using.

So I guess, given somebody who was fully up to speed about what's actually going on,

you can make a decision of whether it makes sense for whatever use case you have of hiding the SSID, that that's a good thing to do or not. But for what it's worth, it's really - it certainly isn't actually keeping anybody from discovering, if they want to, that your access point is there and what its SSID is. And as your devices roam, they are broadcasting that hidden SSID all the time because they once connected to it, and they're still looking to see if it might be around for them to reconnect to.

And a couple little bits of miscellany. I just wanted to note, I got a kick out of this, Leo, and I don't know if you had covered it over the weekend, about the Burger King Whopper commercial.

Leo: Yeah. Isn't that funny? Yeah.

Steve: Yeah. I loved it. I thought it was very clever. And I don't think it was malicious. For our listeners who don't know, at the end of a Burger King commercial they deliberately said what I cannot say on the air, which is the trigger phrase for the Google device, followed by "What is the Whopper burger?" Which they deliberately put that at the end of the commercial, which then triggered all of the Google devices that were listening, that were within earshot of the commercial, to elaborate in a detailed description of the composition of Burger King's Whopper. Which I thought, again, was just very clever. But Google didn't think it was so funny, and they immediately blocked it.

Leo: Also, in the interim, there was a little problem with people changing the Wikipedia entry that came up and saying that the Whopper was made of toenail clippings and - so it turned out maybe not to be the best idea ever.

Steve: Oh, good. I'm glad you were able to provide that additional information.

Leo: Yeah, yeah, the perfect idea.

Steve: Whoopsie. Whoopsie. So we have, from our continuing series on epically cool ways to waste time, we have, believe it or not, generating sequences of primes in Conway's Game of Life.

Leo: Oh, boy. Oh, boy.

Steve: Yes. Nathaniel Johnston, an assistant professor at Mount Allison University in Sackville, New Brunswick, Canada, posted - and I have a link in the show notes for anyone who's curious - just a beautiful piece of work. He wrote: "One of the most interesting patterns that has ever been constructed in Conway's Game of Life is primer, a gun that fires lightweight spaceships that represent exactly the prime numbers."

Leo: What?

Steve: I know. Just amazing. "It was constructed by Dean Hickerson way back in 1991,

yet arguably no pattern since then has been constructed that's as interesting. It seems somewhat counterintuitive at first that the prime numbers, which seem somehow random or unpredictable, could be generated by this relatively simple pattern in the completely deterministic Game of Life."

So here's how it works. "The gun works by firing lightweight spaceships" - now, our listeners will remember that a glider is the smallest object. It's five live cells, which sort of flips back and forth and runs diagonally - I don't remember. I think it's at one - is it a quarter or a half lightspeed? Okay. So that's a glider. There's a lightweight spaceship which moves more slowly, but is able to move horizontally. So this gun works "by firing lightweight spaceships westward and destroying them via glider guns that emulate the Sieve of Eratosthenes."

And of course remember that the sieve is one where you take all numbers, and then, if you imagine them in a grid, you drop out the even ones, then you drop out the third ones, then you drop out every, well, every fourth you've got because you dropped out the evens. Then every fifth one. And so you essentially drop out those that are multiples of primes, and you end up with just prime numbers left behind. So that's the way the sieve works. This Dean Hickerson figured out how to implement that in Life, Conway's Life, by essentially launching all possible lightweight spaceships and then arranging to kill them with intersecting gliders which emulate the sieve.

And he says: "A lightweight spaceship makes it past the left edge of the gun at generation 120N, if and only if N is a prime number." Anyway, and in the picture you can see a stream of them. Essentially they're spaced out so they represent a stream of primes where they are like three and five and seven and eleven and so forth, where those represent - and the sieve has knocked out the ones which are not prime in between. Just incredibly cool.

Oh, and then someone tweeted: "Steve, I've been watching your show for years. I know you love Kindle. Which one do you own? How do you prefer to read your eBooks?" And I was reaching down. I had mentioned before how I am really loving most reading. And this is the way I have my screen set up, Leo. It's an iPad mini. And I have it - so I use the Kindle software on the iPad Mini, set to inverse so it's white text on a black background. And then I turn on the night shift and crank it to its maximum orangeness, which creates what I find to be just a really pleasant, sort of a neon orange glow, very easy on the eyes and not glare-y at all. So I use the iPad Mini when I'm indoors, and then a traditional eInk Kindle when I'm outside.

Leo: Which eInk Kindle do you use, though? I think that's what he's asking.

Steve: I'm using the latest one. I bit the bullet, even though it's ungodly expensive, and I already have other Kindles. But it's - it don't know if you've seen that thing.

Leo: Yeah, like 200-plus bucks, yeah. It's crazy.

Steve: Oh, it's way - and of course I didn't want ads in it, so I had to pay more for that. But it's almost a little square form factor. The cover has, like, three quarters of the battery, and it's removable. And when you remove the cover, it's incredibly thin, except sort of where you hold it, where you have a couple buttons. And the feature that annoys me is that I don't know why they don't do this, I saw so many people complaining that

it's a touchscreen and has physical buttons, but you cannot turn off the screen change, the page change by touch. And I'm often, my hand just sort of wanders, and it touches the screen, and it turns when I don't want it to, even though I've got physical buttons. I don't understand how Amazon can not let us have the option of turning off the touch page turn.

Leo: Oh, you got the Oasis. You got the $300 one.

Steve: That's the one, yes.

Leo: [Vocalizing]

Steve: It's crazy expensive.

Leo: Yeah.

Steve: But look how skinny it is. Oh, Leo, it is really nice. But it will hurt you to buy it. And of course it is...

Leo: I have a Paperwhite. I'm very happy with the Paperwhite. I have a Voyager, as well, or a Voyage, I guess it's called.

Steve: Yeah, I've got all those, too.

Leo: Yeah. I use the Voyage, which is actually not cheap. It's still 200 bucks.

Steve: Yeah, I know. They are pricey.

Leo: When they got to 300 pixels per inch, that's when I said, all right, I have to buy that.

Steve: Oh, it is, it is a beautiful screen.

Leo: That's a nice display, yeah.

Steve: It really is. And lastly, Chris Hall tweeted on Saturday the 15th. He said: "Ryk Brown's Frontiers Saga is amazing." He says: "On third book since Tuesday." So he learned of it from the podcast last week. He said: "Highly recommend this to any Star Trek fan hungry for more." And I'm not going to drag our listeners through this again. I just want to say I'm now - I had thought I was going to wait for the second series to be written before I started, but I couldn't. I have finished the second book of the second

series. So technically that's Book 17. I'm into 18.

Leo, it may be the best sci-fi series I have ever read. And I say that with full recognition of Peter Hamilton's work, and all the other authors we've talked about in the past. I love their stuff. Many of theirs I would argue is almost more kind of classic sci-fi because it's imaginative. I mean, Peter Hamilton with his wormholes that you run trains through, and the Moties on Pandora's Star. Just, I mean, there's, like, really new stuff. But Chris's comment about this being like Star Trek. Remember how I said that my company once as a birthday present gave me the writer's kit for Star Trek episodes, and how I learned from reading that that the goal was not to do gee whiz technology. It was to tell human drama set in that future context. And so, yes, warp drive and phasers factored in. But the stories weren't about warp drive and phasers.

This Frontiers Saga is like that. The thing that makes it so compelling is the characterization is so good. The writing is topnotch. You end up really caring about these people. And when you start into the second set, it develops this Star Wars-like mythology stuff, kind of with an Obi-Wan and a Luke and kind of other whispers of that. It's just - I'm just having the most fun ever. So I just did want to say I think - someone sent me a tweet saying that the first book, which technically it's big, it's the first three of his what he calls "episodes," where it's 15 episodes per series. He said it was free on Amazon. When I looked, it wasn't. But I think for the Kindle it's like three bucks. It's three something, 3.15 or 3.53 or something. It's less than $4. And, boy, if you're curious, by the end of the third episode, by the end of your $3 and whatever cents, you'll know if you want more. I just - it's just fabulous.

Leo: And all 15 volumes are the same people and all that?

Steve: Yes, yes. It is one coherent storyline. And I'm realizing now he must have mapped this whole thing out. And, oh, I mean, I just - I can't spoil it for anybody. But, I mean, it's just - it's really compelling. When I was thinking about like the - what was that last Hamilton, the road wandering off somewhere? I can't remember even what the title was [The Great North Road]. You know, I dragged myself through. And again...

Leo: It was okay, yeah.

Steve: New things, interesting ideas.

Leo: It's one of his mysteries, so it's a [crosstalk] mystery.

Steve: Yeah, yeah. This is, again, there is technology. There's very clever use of weapons. There's no, like, magic power where people get to do anything they want, so they have to live within a constrained universe, which of course is what I think makes the kind of puzzles I like the best is where it's like what you do with these constraints. It's just there's cleverness, but it's about people. It's about this group of people and really compelling bad guys, too. So you want to - I'm just like [vocalizing], I hope justice gets served.

Leo: If you want to put it on the Voyage, six bucks for the first three, that's a pretty good deal; right?

Steve: Oh, Leo. And is Audible available?

Leo: Yes, yes.

Steve: I saw some reference to it. Okay.

Leo: Yes. It's also available on Audible. Not nearly as good a deal. They don't have the first three for one price. So you're buying them all, one by one.

Steve: Yeah, so again...

Leo: I might read it first and see.

Steve: I would say - I bought them all for my nephew, even though it's his father who told me about it at Christmas, my brother-in-law, just because I needed him - he likes sci-fi. He travels a lot. So he has an opportunity to read. And it's just so good. So, okay. I'm going to try not to talk about it anymore.

Leo: Let me - I'm just curious. I want to play just what it sounds like, and you can see if...

Steve: Great.

Leo: You don't listen to audiobooks.

Steve: I don't.

Leo: So it's not even really an issue for you. But I'm just curious.

Steve: No, but I know that we have some listeners who do.

[Audiobook clip]

Leo: A little nasal for me. I'm not sure - I'll read the book first and see. And if you have Kindle Unlimited, the whole series is free.

Steve: And I do, and it is. I mean, I feel guilty.

Leo: That's a lot of words.

Steve: I mean, I actually purchased them for my nephew because I wanted him to own them.

Leo: Well, he's got my $6 now.

Steve: And the author deserves his money. And I'm wondering if he - he calls these "episodes." And he's got his plan is a five-series of 15 episodes per series. I wonder if he's, like, set it up to be made into a TV series.

Leo: Oh, maybe.

Steve: Because as I said last week, I would much rather see this than Honor Harrington. I loved Honor Harrington. It had some fabulous parts. But they were few and far between. This thing just - it never lets you go. Anyway, it's just I'm - okay. So, sorry to take so much time, but wow.

A quick note about SpinRite from a Patrick McFarland. I got a kick out of this. Actually I have this and then a picture, but they're not related. He said: "@SGgrc SpinRite saved the day again! My son's Xbox 360 hard drive was failing. Plugged it into a PC and ran SpinRite on Level 4. Good as new." So there's one example of, yes, we know it fixes DVRs. It fixes Xboxes.

And then the Photo of the Week is a kick because we also see on the next page of the show notes that it fixes the NASCAR Global VR arcade-scale game.

Leo: That's so funny. That's an arcade game running the SpinRite interface, yeah.

Steve: Screen saver, yup, yup. So this is this multi-thousand-dollar, full-on NASCAR race game, where you sit in the seat, and you've got the rubber wheel and gas pedals and accelerators and shifts and everything. And apparently it broke. So somebody who listens to the podcast or is a SpinRite owner, and he's probably a podcast listener because he tweeted this to me, apparently is responsible for these. So he opened it up and booted SpinRite on this NASCAR game. And we can see a PC keyboard off to the left, so he started it going and then wandered off so SpinRite could fix the NASCAR game.

Leo: So funny.

Steve: SpinRite recovering a dead PC-based high-end arcade game.

Leo: I want to go to this arcade. It's got some good stuff in there. It kind of looks like - that's probably the back of the shop there. All right. We have still some time, and I think you still have much to talk about. So let's go.

Steve: Yes, we do have both. So, and I think it'll work out just perfectly. One thing, first of all, I got so much positive feedback last week from our Proactive Privacy roundup. That is, sort of all the different ways that the technologies we use could be leveraged, and arguably are being leveraged, in order to track us and deanonymize us and so forth. One thing I completely forgot because it's like hiding in plain sight, and that's our email address. How many different sites do we identify ourselves using our email address? Typically, that's your so-called username. That's your currently unique token. And then you use a password.

Well, so by using a common email address across a large number of sites, and notice the email address is not being encrypted, or if it is, it is decryptable because they need - the server end, unlike a password, where all you need to do is verify it, the email address needs to be decryptable or just left in the plain, just left in the clear, so that they can email you for various management tasks, password recovery and so forth.

So there is, you know, we're identifying ourselves deliberately by using a non-unique, or even if we go to some trouble to add like a dot something after the name and before the "@" sign, which we know sort of creates a subcategory and can be treated as different addresses, even so it's still obvious to anyone looking at it what's going on. So anyway, I just, again, I just completely missed that when we were talking about it last week. So I just did want to mention that, yeah, there's that hanging out in the breeze, too.

Many people were interested in VPN services. And I ran across another site that I would tend to trust, only because of the nature of their motivation. And that's the TorrentFreak.com site. They have, for a number of years, been going out of their way to send questionnaires out to a huge number of VPN providers, asking them a series of questions. Number one on the list of 12 questions is "Do you keep ANY [in all caps] logs which would allow you to match an IP address with a timestamp to a user or users of your service? If so, what information do you hold, and for how long?" And it goes on, 11 other questions. I won't drag our listeners through it.

There's a link in the show notes to that page at TorrentFreak, although I imagine if you just google "TorrentFreak VPN Services Anonymous Review," that's their title. And this one is this year. It is a recently conducted review. I'm sure you can find the page. And, I mean, it is - I just - it scrolls forever, going through every single VPN they queried and showing all of the results from each of them to those 12 questions. I was hoping they would have a table to summarize it, but I didn't see one, because that would make it much easier just to see this thing in a grid display of red and green summary responses. But for those who are interested in finding themselves a VPN, you might want to check out TorrentFreak.com. And it's their VPN Services Anonymous Review, which is very recent and looked very comprehensive.

Bobbob1016 asked, he said: "SN-607, privacy section doesn't mention not using Google. You say Google gets first party on redirect, but what if you're not using it?" And it's like, well, okay. Certainly Google knows who I am because I have a presence in the Google domain. And Google knows who I am across all their properties. So I suppose if you have never - if you don't have Gmail, if you've never used Drive, I mean, I'm not sure how you would avoid Google, I guess is what I'm saying. You'd have to really not use it and never do anything that would deanonymize you to Google.

And even so, even if you were unknown, then the presence of Google Analytics, which is so pervasive over the 'Net, would mean that Google would be tagging and following you, not knowing who you are, but knowing everything you did and everywhere you, an anonymous entity, went. So certainly Google is connecting, in my case, connecting my real-world identity to everywhere I go and what I do. It's like, okay. I understand that and accept it. Just it's very hard to prevent that from happening. I'm not even sure how you could unless you blocked all scripting, and we know that's no longer practical. But even, you know, I'm not even sure that blocked JavaScript would prevent a passive HTTP query back to Google. It might not. So I don't know. That would be a tough battle.

Nick Bedford said: "I just encountered an issue with blocking third-party cookies. I couldn't log into Jetstar Airlines account until I unblocked them." And that's something we didn't talk about last week, but it certainly is possible that there are sites that use third parties, not only for like affiliate collaboration, but as part of their operational system. So, for example, Jetstar Airlines might have some other service, for example, that handles reservations or login or something, not advertising, but functionally related to the core service such that, if you blocked the access of that third party to their own cookies when you're at Jetstar, they're no longer able to interact as they expect to. So there can be some downside. It's rare.

And what you could do, if you, for example, were a uBlock Origin person, would be to use uBlock Origin to globally block all third-party cookies and then selectively, by site, whitelist given sites. So you could allow Jetstar Airlines queries to enable third-party cookies only there, but otherwise not. So there are ways around, given the tools that we have, to get the best of all worlds.

David Benedict said, or asked, he says: "Listening to SN-607. So would having your ISP NAT connections be a bonus now to your privacy?" Meaning, and we've talked about this before, where in the maybe foreseeable future, if we don't start getting direct IPv6 connections, ISPs that are running short of IPv4, of their own IPv4 allocation, that is, they have more subscribers than they have IPv4 space, could themselves use a NAT router in order to expand their subscribers. Not everybody at the same time, I mean, it would start getting a little tricky if they had high use. But in the same way that we use a NAT router in our homes so that many devices are able to use a single IP, an ISP could do it at that level.

So, yes. That would weaken one useful, but not very strong aspect of tracking that we talked about last week, which is your own public IP on the Internet. That would be obscured. But it's hard to provide an exact quantity. But that's a tiny piece of the puzzle relative to all of the ways that our browsers' queries, with cookies and browser headers in the query, and the existence of other extensions can be used against us. So, yeah. Some addition, but probably not worth writing home about.

John wrote: "OAuth identity providers also use redirection and get to track your movements, at least where OAuth is used." And that's something also that I have often talked about in the past, I forgot to mention last week in our roundup. So thank you, John, for reminding me. And that is the fact that, when you, quote, "Login with Google," "Login with Twitter," "Login with Facebook," handy as that is, that's a query going to that third party to whom you authenticate. And so that third party knows where you're logging in as part of the bargain of using OAuth. So it's convenient, but it does mean that a third party is seeing everywhere you use them to log into other sites. Which of course is one of the things I like about SQRL is that it's a strict two-party system. So it's TNO, and nobody knows where you go and what you do. As far as the SQRL protocol itself works, you still have all the other privacy protection problems that we talked about last week.

Ryan asked, or said: "Important caveat to DNSCrypt. Destination site names leak if that site is using SNI to do HTTPS. Many sites use SNI these days." And that is a great point that I forgot to mention. We've talked about it in the past. That's a Server Name Identification, SNI. That's the extension to TLS which allows one IP on the server to host multiple TLS certificate, HTTPS-enabled sites. The idea is normally there is a binding between the IP address and a certificate. For example, GRC uses that, such that the GRC IP implies the certificate for the site you're visiting. But in hosting providers, for example, you want multiple sites at a single IP, just to save on IPs.

So the web browser that understands this SNI extension will add to that client hello packet, the first packet that goes out that is not encrypted. It establishes the TLS handshake, establishes the encryption channel. But that first packet needs to say "This is the host that I'm looking to connect." So an ISP that was nosey could be capturing the client hello packets and see who you are connecting to, not only at the IP, but use the client hello to disambiguate among the hosts at that IP.

So, Ryan, great point. Thank you for bringing it up. And there's no way to protect that. That's in the TLS spec that's in the first packet that goes from the client to the server, which is the way the server knows which certificate to use in responding to the client hello with its server hello and goes from there. So that is not masked by the TLS connection. It would be by the VPN carrying that because the VPN tunnel would be wrapping all of your traffic, including the TLS setup packets.

And, finally, Malcolm Hannan-Smith said, in a pair of tweets, of SN-607, last week's Proactive Privacy podcast: "It is very important to live test an active VPN with a site like Whoer.net." And Leo, you're going to want to go there: W-H-O, as in who, E-R. W-H-O-E-R dot net. And then in his second tweet: "VPN testing." He says: "I used proXPN VPN last year. It said the PC was safe, but testing showed my real ISP IP address was visible."

This is a cool site. I wasn't aware of it. I checked it out in detail last night. And Leo, there's a button at the bottom for extended information, which you can press. Whoer.net. Go there. You will be surprised. And in fact I discovered that the WebRTC feature in my browser, my Firefox browser, was leaking my internal LAN IP and a few other bits of information which WebRTC is known to leak. There is a sub-article there.

I have a link in the show notes to "How to Disable WebRTC in Various Browsers." I'm not using WebRTC for anything. So under Firefox, again, about:config. If you put in "media.peerconnection.enabled," and probably you can just put in "media.peer," that would be enough to match, you want to turn that to false, then retest again. And sure enough, WebRTC was no longer providing some information I didn't want it to be.

But this Whoer.net is wonderful. And it's an outfit who is offering their, obviously, their own VPN service for five bucks a month. And I'm sure they pass their own test with flying colors. But they're also allowing you to check your standard connection and your VPN connection. And I would by all means recommend it. It's free, and you're going to learn something. So Malcolm, thank you very much for sharing that with us. And to our listeners, thank you for sharing another podcast with us.

Leo: Yeah, that's a great site. I'll play with that at home.

Steve: Yes, yes, yes.

Leo: Thank you, Steven. News and feedback. That was a potpourri. And now you smell minty fresh.

Steve: Oh.

Leo: Yes. We do the show every Tuesday, 1:30 Pacific, 4:30 Eastern. That's 20:30 UTC, if you want to tune in and watch live. I know a lot of people doing their taxes today in the U.S. are probably listening while they bang their head against their screen.

Steve: Thanks to Sue, I already have my California rebate check.

Leo: Nice, very nice.

Steve: My refund, yes. I would be in prison if it weren't for Sue because I would have really intended to do that, but I just would have, well, not gotten around to it.

Leo: We do also on-demand versions. So does Steve. Let me point you to Steve first, GRC.com. That's his website. It's where you'll find SpinRite, his bread and butter, the world's best hard drive recovery and maintenance utility. You also find all the freebies he gives away, and this show. In fact, not only the audio of the show, but a great transcript, so you can read along. It's written by a human, so it's not gibberish. It's actually technically sound. And that makes it also very useful for searching.

Steve: You know that Elaine told me last week that one of the reasons her proofreading is such a substantial portion is she goes through and verifies that every single double quote is balanced. It's like, yeah.

Leo: Wow. That's amazing. Well, it's a work of art. You should check it out. You can also get video, as well as audio, from our site. We don't have the transcripts. We have video instead. That's TWiT.tv/sn, all of our shows available on demand at our site, but also wherever you get your favorite podcasts. Do subscribe. That way you'll get every single episode of Security Now!. We'll be back here next Tuesday. I hope you will, too. Thanks for joining us. See you, Steve.

Steve: Thanks, Leo.