The BlackNurse Attack

Description: Leo and I discuss the results from our listener's informal CAIDA spoofing testing; how "LessPass" turned out to be even less than it appeared; my great day at Yubico; a whole bunch of IoT news; updates from PwnFest and Mobile Pwn2Own; a bit of miscellany, including the probable elimination of the need for Dark Matter; a new WiFi field disturbance attack; a wacky Kickstarter "fingerprint" glove; and the "BlackNurse" reduced-bandwidth DoS attack.

High quality (64 kbps) mp3 audio file URL: http://media.GRC.com/sn/SN-586.mp3
Quarter size (16 kbps) mp3 audio file URL: http://media.GRC.com/sn/sn-586-lq.mp3

SHOW TEASE: It's time for Security Now!. We're going to find out why LessPass is even less than it was last week. Steve visits Yubico and Stina and her husband. A lot more IoT news. News that will relieve you, if you're a Pixel owner, from the PwnFest. And, wow, a new WiFi sniffing attack you won't believe. It's all coming up next on Security Now!.

Leo Laporte: This is Security Now! with Steve Gibson, Episode 586, recorded Tuesday, November 15th, 2016: The BlackNurse Attack.

It's time for Security Now!, the show where we protect you and your loved ones online, the show most beloved by geeks worldwide. And I can't take any credit for it. It's all this guy right here over my left shoulder, Mr. Steve Gibson.

Steve Gibson: Leo, you made it all happen, however.

Leo: I turn on the lights.

Steve: None of this would be going on without you. I sometimes remind people that, you know, I was even sort of reluctant in the beginning. It's ended up being one of the most useful things I've ever done.

Leo: Good, good.

Steve: So I'm completely pleased by it. And I think it works really well.

Leo: Good. Well, I'm of the same opinion.

Steve: This could have been a Q&A, but the industry has given us no opportunity to catch our breath and to handle some listener questions. Although I haven't been saying, but I have wanted to mention that, if our listeners have noticed that the podcast has become much richer with stuff recently, it's that I've been making a concerted effort to, the day before, go through my Twitter feed for all of the previous week, which because I'm so busy I often can't do interactively. But it allows me to sweep up all of the tidbits and findings that our listeners have sent through that channel.

There's a lot of repetition, which I have no problem with. People say, "Oh, Steve, I'm sure you already know about this, but." And I always, when I can, I will say, "Thank you for making sure I knew," because sometimes I don't. There's always somebody who's first. But I just wanted to make sure. I don't always have the chance to respond to everybody whose tweet I receive, of course, because I'm getting hundreds of them. But it is for me a clean, fabulously useful channel. So I just wanted to make sure, for any people who have tweeted me things who have then noticed that I have mentioned it in the following podcast, even if I didn't have a chance to say explicitly thank you on a per-tweet basis, that it is a consequence of our listeners that the podcast, and of course my willingness to make the effort to really pull all this together, I think has gotten better in the last few months as a consequence of this.

Leo: Good, yeah.

Steve: And this is another example. This is nominally titled "The BlackNurse Attack." I have no idea where that name came from.

Leo: Terrible name.

Steve: Normally we can figure out where it came from. I don't know, that just seems random. But it's an interesting different kind of denial of service attack. And it's one we talked about at least 10 years ago, which has sort of come back around with an interesting twist and reason. But it's not a huge subject, and we have so much to talk about. We've got results from my question last week for our listeners who were willing to try that CAIDA spoof testing. We have results. LessPass, that we also talked about last week, turned out to have been even better named than we knew. Oh, boy. So we have to revisit that.

I want to share briefly the result of my day at Yubico. I was up on the peninsula, Leo, in Palo Alto last Thursday, just for a quick little trip up to visit Stina, to meet her husband Jacob, who's the head techie, and to give a SQRL presentation to Yubico. We've got a bunch more Internet of Things news. Cory Doctorow wrote a beautiful piece that I'm going to share the beginning of. News from PwnFest and Mobile Pwn2Own, where the headlines, I think, got the message wrong, unfortunately. They went for inflammation rather than credit.

Leo: Oh, okay. Because I read the headline, and I was scared. So all right.

Steve: Yeah. It doesn't matter that Google's Pixel phone got hacked in 60 seconds. It matters that it was patched in 24 hours.

Leo: Yeah, yeah.

Steve: Yeah. So I think that's props to Google.

Leo: Okay.

Steve: We've got some miscellany, including the probable elimination of the need for dark matter. After all, I did say it was miscellany.

Leo: Wow.

Steve: Yes. It's very good news because dark matter, I don't know about you, Leo, it's been really upsetting me for several decades.

Leo: No, you know, I hate it when they say, well, we don't know, so we'll just put this little variable in the equation, and everything then works out. I don't like that.

Steve: But not little. Eighty percent of the mass of the universe had to be invisible.

Leo: Right. There's something, you know, it's just not elegant.

Steve: No.

Leo: And Occam's Razor says the simplest solution is usually the best. Well, I'm glad - I'll be interested to what you say there.

Steve: Yeah. So it was some research that first came to light at the end of '09, so just almost seven years ago. But then a paper was updated by the same guy, a theoretical physicist, last week, where he put the pieces together. Anyway, I just - it's really interesting. I think that our listeners will get a kick out of it.

Leo: Good.

Steve: There is a new WiFi, what I call the "WiFi field disturbance attack," which also

sort of says we just ought to give up and go home.

Leo: Oh, dear.

Steve: Then there's a wacky Kickstarter fingerprint glove and the BlackNurse reduced-bandwidth DoS attack. So I think we can promise our listeners a great two hours. So it's nice to know that I'm not alone.

Leo: You're not alone. You're not alone. I know what you're going to say. I'm not alone.

Steve: I'm not alone. The Picture of the Week is another fun O'Reilly made-up cover. It's the Essential series: "Managing + Navigating 1 Million Browser Tabs." And the little subhead up at the top: "Because you know you just saw the tab you need."

Leo: Yeah, yeah, yeah.

Steve: You can't, you know, it's there somewhere. And as I was telling you, I still have SpinRite 6.1 R&D tabs open from before I began SQRL.

Leo: What?

Steve: They're there because that's where I was, and they're...

Leo: You never restart your computer?

Steve: Oh, I do. But I have good session management.

Leo: Oh, you have it backed up, okay.

Steve: Yeah, exactly. I've got all my tabs. And I'm, you know, Firefox gets a little finicky, especially Amazon. Amazon's pages seem to be growing in bulk. And so you load a couple of those, and it takes a while for the thing to stop spinning because it keeps getting more assets from wherever. And then finally things settle down. But anyway, yes. This demonstrates the fact that it's handy to have tabs. And in fact I wanted to mention also that some iOS update, it was a while ago, so it may have been 8, or maybe 9 - it was one of the major changes - thankfully allowed the default iOS browser, Safari, to increase its number of tabs because of course I'm always choking there.

And it's because what I discovered by coincidence was you can only actually have - and I think the number's 32. I did hit the limit. And what happens is it's a pushdown list, or a LIFO, so that, if you open a 33rd, it lets you do it. But without telling you, it overwrites the oldest one. So I'm sure they are trying to do what they hope you want, but it's not

what I want because I have that oldest one for a reason. Just like I've got SpinRite tabs still open and waiting for me to come back to them for a reason.

Leo: They're vintage. They're aged. They're like fine wine.

Steve: Well, it's sort of - just take a browse through your nostalgia every so often. Anyway, so now what I do is I switch, in Safari and iOS, to that zoomed-out view where you can sort of see the pages stacked. And I'll quickly sort of make sure that I'm still way shy of 32, so I'm not in danger of running over and killing off something that I'm still trying to, like, hold for some future purpose.

Leo: You're a tab duffer if you only have 32. That's, like...

Steve: That's all you can do on iOS, which is my point.

Leo: At TWiT, one of the - I think it was Alex Wilhelm was sitting on TWiT and screamed and said, "All my tabs are gone." So, you know, he was bereft. We have to set him up with your solution, your tab archiving and backing up solution.

Steve: Yeah, yeah. So many of the feedback that I got, both via Twitter and in the mailbag, which I did dump last night to look through, were, as I asked for, our listeners' feedback about their IP spoofing tests. And I was gratified by the result for two reasons. One was that our listeners will remember that two weeks ago I was brought up short by the whole question of outbound spoofing behind a NAT router because we've always been focused on the ISP, and the ISP not allowing their client or customer traffic to egress from their control with an obviously fake spoofed IP that can't ever come back to them. And there isn't, there is no - in no protocol of the Internet is there a justification for doing that. There just - there isn't one.

So as our listeners remember, a couple weeks ago I realized that there are no light bulbs that are not behind NAT. You only typically have one IP from your ISP. Who has only - or IPv4, at least. We know that in the future that's going to be 16 bits' worth of IP per customer. So anyway, the result of the analysis was not a single, not one, NAT router passes spoofed IPs. They all block them.

There was one from the herd, the Netgear N600, the WNDR3700 v3 with the most recent firmware. It turns out it allowed adjacent IP address spoofing, that is, it wasn't checking the lowest byte of the 32 bits. So you only had a range of within 256 IPs of the same block you were in. So it's a little bit softer checking. But the practical consequence of that is it would be of no use to any attacker because the attacker's going to try to get some equipment in your facility, in your premises, on your network, to send packets with an arbitrary IP, almost certainly not within a few hundred of where you happen to be. So the bottom line is, 100%, and this is, of course, this is a skewed sample. This is Security Now! listeners. But I don't know anybody who doesn't have a router. You have to. If you're going to have more than one device on an IPv4 network, you've got to have NAT.

So 100% of the NAT routers reported dropped any attempt at spoofing from getting out of the individual's local LAN. Which is as it should be. So it may not even be that ISPs have ever taken any proactive action. It may very well be that all of their customers have

without knowing it, just by using NAT, which kills spoofs. And it must also be that the bad guys know. This doesn't mean you cannot spoof on the Internet. You certainly can. You just have to go to a direct connection or modify your NAT to explicitly add some rules to paths. But the default NAT translation, where the source address is rewritten to the public IP as the packet leaves, so that it's able to come back to you, and then that destination IP is replaced to go back to the computer from which it came, that drops spoofed packets.

And so I believe this makes sense, then, as to why all of these IoT devices are not being seen spoofing. If they tried, their traffic would never get one hop away from them before being dropped at the NAT boundary. And so they're just going with in-band non-spoofed attacks which, unfortunately, as we've noted, there's such a large population of existing both PCs and now IoT devices that there are plenty of opportunities for attackers to get in them and generate non-spoofed traffic from those. But really interesting result.

So thank you, everybody, one and all, who took the time to run the test and shot me their results. I looked at a whole bunch of those network diagrams and spider charts. Everyone was sending me the links that their test generated; and every single one of them, with that one exception, showed that spoofing never got past the NAT router. So very nice piece of information and intelligence for us to have. And it makes sense.

Leo: Yeah.

Steve: So, okay. First of all, the guy who named that piece of code LessPass shot me a tweet shortly after the podcast, quoting me saying, "Hope he doesn't attempt to attempt a trademark on that," and then replied, "No, we are not trying to obtain a trademark on LessPass." Which of course he would not be able to obtain. But one of our listeners dug a little deeper than I did. A listener named Adam took a look at the code and was horrified by what he found. He reminded me that I had said last week that I hadn't bothered to look at the code, for two reasons. It was so trivial to do this correct that I just gave the guy the benefit of the doubt that he had. But also it didn't even pass muster from a functional UI standpoint. You know, I explained why it just wasn't practical to use this thing, independent of how it worked. Well, Adam took a look at the source and, as I said, was horrified.

So here's how the problem should be solved. I didn't articulate it last week because we've talked about this before. You take all of those inputs, the three fields - the domain name, the username, and your master password - and you just hash them together through some algorithm. And again, let's assume that it would be a strong hash, like an SHA-256. That's going to convert that ordered set of three strings into a high-entropy and maximum entropy binary blob, 256 bits. And if you needed more, you could iterate the hash and get some more. So, I mean, it's possible to do that.

So the idea being you are mapping that deterministic input into a bunch of binary. Okay. Then what you do is, looking at the various checkboxes, you determine the size of the alphabet, that is, is lowercase on? That's good for 26. Is uppercase alpha on? That's good for another 26. The digits zero through nine, there's 10 more. Special characters. I have 33. Adam mentioned, I think the guy was using 26. I don't know why, but whatever. So you sum that up. And that's the size of your alphabet, meaning the set of characters that could appear in every position of that password.

So what do you do? You perform a long division. You take whatever that number is, say that it's 64. Well, that's kind of cheating because it's easy to divide. But whatever it is.

You perform a long division by that number of the 256 bits. That will result in a result of the division and a remainder. The remainder will be between zero and N-1. That is, where "N" is the modulus, or the size of the alphabet. So that remainder picks in an ordered fashion one character from that character set. And so that's your first character.

Then you repeat. You simply divide that not quite any longer 256-bit number because it's been reduced in length by the division. It's literally had that log2 whatever number of bits, you know, some fractional number of bits removed from it, essentially. So you divide it again. And you get another remainder in the same range, map it across your character set. That's your second character, and so on. And you keep doing that, consuming entropy from the output of the hash until you've satisfied the number of characters that the passwords should have. I didn't go through all that last week because we've talked about this before. That's the way you solve this problem. Problem has been solved. What did this person do?

Leo: Oh, no. He didn't do that?

Steve: Oh, Leo. Leo, no. He did something...

Leo: You know, all you have to do is, like, look it up; right? I mean, it's...

Steve: Yes.

Leo: Yeah, okay.

Steve: It's the way you solve the problem. Get this. He divides lowercase and uppercase into vowels and consonants. And I have no idea why. Then the first character...

Leo: It's not English here. Who cares?

Steve: The first character is a consonant, a lowercase. The second character is a lowercase vowel.

Leo: He's trying to make it pronounceable, maybe?

Steve: Well, the third character is an uppercase alpha, is an uppercase consonant. The fourth character is an uppercase vowel. The fifth character is a digit. The sixth character is a special character. And then he repeats, meaning...

Leo: Oh, that's terrible. Oh.

Steve: It's awful. It's unbelievable. It's like, what? You know, it really is...

Leo: You're reducing the entropy hugely.

Steve: Yes. So you know that any password this creature produces...

Leo: You call it "the creature."

Steve: ...is going to have a vowel as its first character.

Leo: That's ridiculous. How many vowels are there? Oh.

Steve: A-E-I-O-U.

Leo: And sometimes Y.

Steve: Yeah. It's, I mean...

Leo: That's terrible.

Steve: Thank you, Adam, for telling us, just because, as I said, this thing deserves its name much more than I knew, LessPass.

Leo: Wow.

Steve: I mean, and even inspecting a few of its outputs, you would immediately see, wait a minute.

Leo: They all look the same.

Steve: Why is it always a vowel, then always a consonant, then always a vowel and always a consonant, then always a digit, and then always a special character? And the problem is of course attackers would see that, too, instantly.

Leo: Right.

Steve: And say, oh, well, this makes our job much easier. So if last week's - I had to come back to this, thanks to Adam's bringing this to my attention because, again, I didn't take it seriously already. But if any of our listeners, thought, well, I still think this is useful, be advised. You'd better choose 45-character passwords in order to get sufficient entropy. And even then I should say that lord knows what algorithm he chose to choose

vowels. I mean, it sounds like long division is beyond this person. So he may have just, I mean, I don't even want to - I can't even guess. I didn't look at it. I don't want to know how he chose the vowel. Did he, like, keep taking bits from the hash until he found a pair that fit within zero to five? I'm just - I don't even know. But, boy, this is - you know.

And our lesson here, our takeaway is that this should sort of be a cautionary warning for us. No matter how well-intentioned they may be, not everyone is equally capable of solving important problems correctly. And even those who are, we know, can still make mistakes. But the last thing anyone should trust is an ad hoc construction such as this thing. And of course Telegram, the "encryption," unquote, I put it in quotes in my notes here, you know, was the same. They just made up some wacky-doodle encryption scheme and said, whoa, yeah, it really scrambles the bits up. It's like...

Leo: Looks scrambled to me. Hey, this raises a question because there is a setting on LastPass, and I bet many password managers, to make a password pronounceable. And I'm sure that that's what this was all about, was make it pronounceable.

Steve: That was my first thought, too, was consonant, vowel, consonant, vowel. You'd have some chance of, like...

Leo: Remembering Kaka959.

Steve: And maybe he thought he was being cute or clever. But unfortunately he was destroying the security.

Leo: It sounds like so then making it pronounceable is probably not a good choice in LastPass.

Steve: Correct, correct. And in fact later we talk about the breach of Adult Friend Finder. And, boy, there's some passwords you do not want to pronounce.

Leo: I think 99% of the passwords there were - because they were using SHA-1 at best.

Steve: Yes. Or in the clear, yes.

Leo: Geez Louise.

Steve: And NSFW.

Leo: Yeah, of course. Well, it's Adult Friend Finder.

Steve: For a lot of those.

Leo: But, no, so that's good. So I won't use that "make pronounceable." I think, if you think about it, you don't have to be a mathematician to understand that your goal is to make a password as random as possible. So anything that reduces randomness, or as Steve would say, entropy, is going to reduce the effectiveness of your password.

Steve: It's going to increase its brute-forcibility; right.

Leo: Yeah, yeah.

Steve: But to me the idea of that option for LastPass seems odd and antithetical to it because it's there so that you don't have to remember your passwords.

Leo: Right, right. Well, maybe not just remember, also say it to somebody? Right?

Steve: Yeah.

Leo: Yeah, I agree. It shouldn't be in there. And I won't - and I never did use it. But partly because it isn't that pronounceable.

Steve: No.

Leo: It's kind of like what this guy would generate, yeah.

Steve: No, no, it's like, you know, and I'm completely converted. I've given up all hope of knowing any of my passwords.

Leo: Right. It's like saying "make password memorable." No, don't do that. That's not what you want.

Steve: Exactly. Exactly. So a couple months ago Stina came down for her typically annual trek to Southern California. She was on her way to someone in San Diego and stopped by, and we had coffee for a couple hours in the morning. And I told her that SpinRite was effectively finished, that I needed to work on the installer.

Leo: SQRL. Not SpinRite, SQRL.

Steve: Sorry, sorry, SQRL.

Leo: You just gave everybody a heart attack. SQRL.

Steve: Yeah, sorry. SQRL. Same first consonant; but, yeah, different goal. Anyway, so I committed to coming up and showing them. You know, she's a great evangelist, but she's not the crypto techie. Her engineer husband Jacob is, and they have a staff of crypto techies. And I didn't want anything from them except for them to know what it was, that is, just to sort of plant the knowledge because she is completely involved in identity on the Internet and has been, you know, forming relationships with Google and many other major corporations.

And I needed to explain it to the techie guys and her, just so that they knew what it was, because the fact is it has really evolved over the last few years. I mean, people are anxious for it. But I'm a "get it done correctly once so that it can live for a long time" approach, rather than patch, patch, patch, patch, patch. And in fact, in assembling a 38-slide presentation, which is a full-feature walkthrough of SQRL, which now exists, and which I will share probably next week - because I got some great feedback from the SQRL newsgroup. I gave them the presentation to look at before I headed up to see Yubico, and they brought up some points for some things I could add some clarity to that were great. So I'm going to do that, and then we will have a set-in-stone, full-feature walkthrough.

Bottom line is they were really impressed because one of the things that SQRL has that nothing else has, even FIDO and U2F, is what we call "identity lifecycle management," meaning built into the system is what happens if I lose my key. What happens if the government gets my phone, and I no longer trust my online identity? SQRL provides mechanisms to take it back from an attacker, from Big Brother and so forth.

Anyway, it was a great day. We spent about 4.5 hours going through the whole thing, answered every question they had. I gave them some things to think about because there are some little bits of inspiration, and I think true invention in a couple areas, one that I've mentioned before called the Identity Lock, which is a unique, as far as I know, a unique instance where what you carry with you in the SQRL client is able to create new assertions, but not prove them, which means that that creates another level of authentication which we use for emergency recovery operations. And the beauty of that is that, if an attacker were ever to get your SQRL client in any way, they only get the ability to create new associations, if they had everything, but they can't prove them. You need what we call the "rescue code" for that.

Anyway, I think that our listeners will probably get a kick out of just browsing through this. So I will have it finished by this podcast next week. I will have added a few things to it. And we'll have a very nice, front-to-back, soup-to-nuts, presentation on what SQRL is and the way it works, followed pretty quickly, I hope, by some code people can start to play with.

And this was just sort of a random tweet that caught my attention. A Joe Rodricks tweeted a question to me and Wired. He said: "Why is there a silent audio track playing when I visit Wired.com on my phone? Not cool." Now, I assumed, immediately, I thought, that's interesting. And I thought, hey, I bet I know what that is. And I bet you do, too, Leo. It's probably their technique for monitoring how long a user remains on the page. While you're there, your client is receiving an audio stream. And when you hit back or switch away or whatever, that gets stopped.

And the trouble, of course, with doing this is that, while Wired may have ample

bandwidth and purchases it in bulk at massive discount, their individual visitors don't have that luxury and may well have fixed data rate plans which are decidedly more limited. So I think this sort of shows another example of a diminishing concern for website visitors. Maybe the advantage is that this allows them to track people with scripting disabled because I think you could probably cause audio to play without needing JavaScript enabled.

Leo: Yeah, I think so, yeah.

Steve: Whereas...

Leo: Is it an MP3 or...

Steve: He just said audio. He didn't specify anything more. If our listeners are interested in digging in, I'd love to hear what more they find. But, you know, because I first thought, well, you know, there are, like, way less bandwidth-intensive ways to do that. Like JavaScript could just - you could set a timer in JavaScript and have it occasionally poll the server to say, hey, I'm still here, I'm still here, I'm still here. And in fact that's what SQRL does in order to recognize that your client has authenticated. While it's displaying the page, it just pings the server to see whether the user has, externally from the browser, logged in yet. And when the browser sees that they have, it updates the page sort of magically. But that's the technique I used. But it does require that scripting be enabled in order to run that code to perform a very short little query occasionally. So my guess is that this is meant to be a robust, non-script-required means of kind of keeping a channel open back to the server so they can monitor how long their users stay there.

Leo: Seems like a terrible idea.

Steve: It really does.

Leo: I wonder how much bandwidth it uses. That's awful.

Steve: Yeah, hopefully it's a low-bandwidth audio. Because, I mean, again, it's not zero cost to Wired. But it's just annoying to know that there's, like, a constant suck down, even after the page finishes loading, apparently by design. Wow.

Leo: Wow is right.

Steve: And you may have picked up on the news, Leo, that our web browsers turn out not to be the only thing that's responsible for writing massive amounts of data onto, typically, our hard drives and, of great concern, our SSDs. It turns out Spotify was five months ago told that this was going on and just ignored it for five months until it finally started gaining enough steam, and maybe on the coattails of the awareness that browsers were having a problem, and linking that to SSD [dropout].

Dan Goodin reported in Ars Technica, saying: "Streaming app used by 40 million writes hundreds of gigabytes per day." And he said: "For almost five months, possibly longer, the Spotify music streaming app has been assaulting users' storage devices with enough data" - now, that's not technically correct. I'll explain what the bug was in a second - "enough writes to potentially take years off their expected lifespan. Reports of tens or in some cases hundreds of gigabytes being written in an hour are not" - I mean, and I've seen a terabyte a day - "are not uncommon, and occasionally the recorded amounts are measured in terabytes. The overload happens even when Spotify is idle and is not storing any songs locally."

He writes: "The behavior poses an unnecessary burden on users' storage devices, particularly solid state drives, which come with a finite amount of write capacity. Continuously writing hundreds of gigabytes of needless data to a drive every day for months or years on end has the potential to cause an SSD to die years earlier than it otherwise would." And we all know that's true. That's why wear leveling is a crucial portion of SSD physical storage management. "And yet," he writes, "Spotify apps for Windows, Mac, and Linux have engaged in this data assault since at least the middle of June, when multiple users reported the problem in the company's official support forum.

"Three Ars reporters who ran Spotify on Macs and PCs had no trouble reproducing this effect, which had been reported, not only on the Spotify forum, but also on Reddit, Hacker News, and elsewhere. The Spotify app wrote from 5 to 10GB of data in less than an hour on Ars reporters' machines, even when the app was idle. Leaving Spotify running for periods longer than a day resulted in amounts as high as 700GB."

When the story in Ars was first reported, Dan wrote that Spotify had not responded to them by filing deadline. Then there was a later update posted with a very sort of wimpy, mealymouthed, like, oh, you know, we're looking at addressing the issue and will be resolving it soon.

So here's what's going on. It will be fixed in v1.0.42. It actually is a bug. I would argue that the browser problem was carelessness, that is, just not caring to - as we know, the browser is saving its state very often by default so that, if it crashes or hangs or anything, you're able to recover. The problem is it doesn't do something as simple as seeing whether the state has changed from now until five seconds ago, when it last wrote a state update. So that's just carelessness.

In this case, it actually was a bug, apparently. Spotify is on top of the SQLite database, and a function in the database called VACUUM is being continually, repeatedly, and redundantly called. The VACUUM command in SQLite rebuilds the entire database, repacking it into a minimal amount of space. So the idea is that, if the database has had a chance to sort of grow with records being deleted and added and deleted over time, you end up, just due to the nature of the way the database's tree is structured, you end up with nodes that are typically half full, and lots of pages that are not full. So it's possible to say, okay, let's compact it. And they call that "vacuuming," where it squeezes it down.

Well, once again, this thing - but that's the kind of thing maybe you would monitor or meter how much record deletion had occurred, which would tend to open up holes that might then be useful to vacuum, rather than just doing it carelessly and constantly. So this thing is just - it's not actually, like, recording new data coming in streaming. It's just thrashing for no reason at all, reprocessing over and over and over an already squeezed database.

The problem is apparently this 1.0.42 is just becoming available. So if this is a concern to

you, see if you've got 1.0.42 of Spotify. See if you can update. They're being a little sluggish in getting it out. So it may not be available for your platform yet, in which case you may choose just to terminate the process until you learn that 1.0.42 is available, and they say that they have got this fixed.

Leo: So to be clear, the issue is not that it's filling up your hard drive. It's writing/erasing, writing/erasing because it's compacting a database. It's that it's thrashing your SSD.

Steve: Right. But those are writes. And the writes...

Leo: Yeah, no, it counts. I understand, yeah.

Steve: And so even if you rewrite the same thing on top of itself, it still causes a fatigue of the underlying SSD cells.

Leo: A spinning disk wouldn't be an issue; right?

Steve: No, no.

Leo: I mean, it's wasteful and stupid, but it's not damaging.

Steve: Correct. Correct. I mean, many of us who have lights on our hard drives kind of every so often look at them and go, what is it doing? I'm like, what is going on? I haven't touched it for an hour. And [sound effect].

Leo: Could it cause a slowdown? I mean, is it using CPU cycles? I imagine it is.

Steve: Yeah. Oh, yeah, yeah, it's competing with anything else you're doing. I mean, it's not good.

Leo: It's the equivalent of Windows reindexing its hard drive all the time.

Steve: Right, right. Or it's like defragging just because we, you know...

Leo: Yeah, why not? You never know.

Steve: But, see, even a defrag, it will not rewrite obviously defragged regions. It just sort of fusses around the fringe for anything that's changed. This thing presumably is just complete, well, based on the amount of data it's writing, we know that it must just be

completely revamping your database. And it would also make sense that those with larger Spotify databases are suffering a much larger vacuum consequence because there's, like, just a larger database to go rummage around through and resqueeze.

Leo: You know, Steve, we're having a little bandwidth issue with you. I was wondering if you would mind if we took a break here.

Steve: Perfect.

Leo: Before the Cory Doctorow story. I don't know if you have something going on the background there. It's really degraded at this point. Frame rate's gone down to one per three seconds or something.

Steve: Wow. Yeah, nothing here. I'm as stable as always. Well, mentally.

Leo: So you don't see a bandwidth hit at all, huh?

Steve: Network-wise, no. I have not been doing anything.

Leo: It's so weird. Because of course we've got...

Steve: It'll recover.

Leo: Yeah. Well, I think what we're going to do is hang up and call you back. Sometimes that helps. Kick Skype in the butt.

Steve: Okay.

Leo: So we have called Steve back, and the signal's not great. But I think it's usable, so let's continue on.

Steve: Okay. Yeah, and you sound okay in this direction. But I see that what you're sending back to me is a little blurry-looking.

Leo: Yeah, yeah. I don't know what's going on.

Steve: Okay. So I joked last week with the meme, "All your light bulbs are belong to us," which of course harkens back to, what was it, the '80s or something, the "All your base are belong to us" was an Internet meme.

Leo: Yeah. It was a bad videogame that at one point says, "All your base belong to us." Yeah.

Steve: So Cory Doctorow wrote a nice piece because some researchers from, I guess it's Dalhousie University in Canada and the Weizmann Institute of Science in Israel have just published a working paper detailing a proof-of-concept attack on smart light bulbs which allows them to wirelessly take over the bulbs from up to 400 meters away. They are able to then write a new operating system to them and cause the infected bulbs to spread the attack to all the vulnerable bulbs in reach until an entire city is infected. And, you know, once upon a time this would seem farfetched. But we're now living in once upon a time. I mean, no one would believe that something like this couldn't actually happen, given everything that we see is happening.

And what's more of a concern is this was not some off-brand Chinese light bulb, not to pick on them, but as somebody who just was selling something without any attempt. The researchers demonstrate attacking bulbs by a phone or a ground station that then attacks Philips Hue light bulbs, the most popular smart lighting system on the market today. Philips Hue uses ZigBee for its networking. And we've talked about that before when we were initially talking about wireless IoT devices. ZigBee is a wireless protocol designed for low-powered Internet of Things devices and has many built-in security features. The most important of these is that, once a device is initialized as part of a ZigBee network, it cannot be hijacked into a rival network unless you can bring a controller into close proximity to it, that is, a couple centimeters away.

However, there is a flaw in the ZigBee implementation in the Hue system. And the researchers showed that they could hijack bulbs which, again, you're not supposed to be able to connect to at any distance greater than a couple of centimeters. They were able to do so from nearly half a kilometer distance. And this is possible, as we discussed, back when we were discussing the ZigBee protocol, because ZigBee does not encrypt all traffic between devices. That is, it doesn't encrypt that session initiation definition graphic because that's the way it bootstraps itself.

So essentially there is a bug in the bootstrapping system that has allowed them to contravene the way it was intended to work. The Hue system, the Philips Hue system has safeguards to prevent malicious tampering. Updates have to be cryptographically signed using a very strong algorithm, or be rejected by the Hue systems. However, the researchers were easily able to extract the signing keys, which are the same for all Philips ZigBee products, and then use those keys to sign their own malicious updates. So here's a system where clear attention was placed, clear focus was put on the security of the system; yet, even so, there was a mistake which these guys were able to leverage into essentially, if you could imagine a metropolitan area where Philips Hue light bulbs are high density, this thing could form its own private viral mesh network, essentially. And that's what they did. They were able to take over any Philips Hue system.

And so Cory writes: "There are many ways that a hijacked Hue system could be used to cause mischief. ZigBee uses the same radio spectrum as WiFi, so a large mesh of compromised ZigBees could simply generate enough radio noise to jam all the WiFi throughout a city. Attackers could also brick all the Hue devices citywide. Or they could use a kind of blinking Morse code to transmit data stolen from users' networks." That's a little farfetched because that's pretty low-bandwidth. But it could be fast-blinking. And then he says: "They could even induce seizures in people with photosensitive epilepsy," which would not be funny.

"The fact that the attack targets devices by ZigBee signals, rather than over the Internet, means it's virtually impossible to defend against through traditional methods like firewalls." And as I said at the top, not so long ago this scenario would have been seen as farfetched at best. Now, its exploitation really seems more like a virtual certainty. Wow.

Many people like the Web of Trust browser extension. It has, I think, a user base of about 140 million, if I remember from my research. Everyone knows I'm a huge fan of browser extensions. They've become, for me, an integral part of my daily management of my web browser portal to the Internet. They add security, block annoyances, add extra features that I find valuable, like managing hundreds of tabs and saving session state as necessary. But unfortunately, a German TV channel, NDR, did some research into a very popular browser extension, the Web of Trust, and uncovered a serious breach of privacy by the whole Web of Trust (WOT) service. 140 million web surfers trust it to help keep them safe online. It's been around since 2007 and describes itself as a "safe web search and browsing service." And of course we know that it boils down to being a crowd-source-driven website reputation and review system, so users can view ratings on a per-site basis for trustworthiness or child safety, as well as provide their own feedback and add their own ratings.

So this Channel NDR investigation uncovered that, while you have the WOT extension installed, extensive data collection is going on in the background behind your back. WOT not only collects and records data on a per-user basis, but then analyzes and sells it to third parties. The WOT privacy policy states that your IP, geographic location, device type, operating system, browser, date and time, web addresses, and overall browser usage are all collected, but that it is non-identifiable. However, NDR found that it was a simple task to link the anonymized data to individual users of the service. And just by looking at a small sample size of around 50 users, they were able to retrieve data on known users by account name and their email address, travel plans, illnesses, sexual preference, drug consumption, confidential company information, ongoing police investigations, and their browser surfing activity including all sites visited.

Mozilla has immediately removed the Web of Trust extension from their Firefox add-on page, due to the violation of their guidelines by this extension. And it seems likely that other browser vendors will follow quickly. So I just wanted to give a heads-up. If we've got 140 million users of this, I wouldn't be surprised if our listeners number among them. And unfortunately it looks like this organization that has provided this otherwise nice-looking crowd-source service is not being very credible with their support for their own users' privacy. So you want to consider uninstalling that extension. Apparently you can't - I didn't check, but you cannot even get it from Mozilla or Firefox any longer.

Leo: Ugh. You know, anybody says "trust me," I always say, well, let me see.

Steve: Yeah. Yeah, exactly.

Leo: It's the Web of No Trust.

Steve: Wow. It's, well, yeah, exactly. It's becoming increasingly the web of how can we leverage our users for our own profit.

Leo: Right.

Steve: We talked briefly about Adult Friend Finder. I'll just note that it was a huge breach. I ran across a site I wasn't aware of before called LeakedSource, L-E-A-K-E-D-S-O-U-R-C-E. This is a group that follows and aggregates site breaches. They, as they are wont to do, they sensationalized this breach, saying that the sexual secrets for hundreds of millions were exposed in the largest hack of 2016. And in fact, across their properties they have AdultFriendFinder.com; Cams.com, which I guess is an adult webcam sharing something; Penthouse.com; Stripshow.com; iCams.com. Across all of that - and that's managed by a single organization, more than 400 million accounts, representing 20 years of customer data, were...

Leo: Including deleted accounts.

Steve: Yes, exactly. It turns out, when you go to delete your account, what they do is they rename your email address by putting an additional @deleted1.com at the end. So, but otherwise it's still there. So what that does is that sort of satisfies you because you - so you go, "I want to delete my account." And they go, okay, you're deleted. Then, you know, just to make sure, you try to login again. But they changed your email, which is the way you identify yourself, so it's like, oh, good. You know, I guess that worked. I'm deleted. No. You simply can't log in as the email address you used to have. Nothing got deleted. Twenty years' worth of salacious details apparently now available. And as you mentioned, Leo, in some cases the passwords were never hashed. They were always in plaintext. And where they were, it was a simple SHA-1, which modern GPU brute-force password crackers just cut through like butter now.

So I have a link in the show notes for anyone who's interested. But they itemized the various percentages of passwords that had been reverse-engineered. And the upshot was, across all the properties, 99% of all available passwords are now visible as plaintext. And you don't really learn anything new browsing the list, although you would have a good sense that this is an adult-related site when you saw what some of these passwords were. But of course numbered among them is 123456. I looked for "monkey." I thought, well, there'll be a monkey there. But no. Instead there's other...

Leo: Means I didn't have an account there, so that's good news for me.

Steve: Right.

Leo: Wow.

Steve: Oh, boy. And here's - this was entirely foreseeable, but it's sort of interesting to see the details. We know about the Mirai botnet, which was credited, if you can use that term, with bringing down a surprisingly large swath of the Internet a few weeks ago by its massive attack. What was it, 600GB, I think, or gigabits per second, I think I remember, as the number I saw, generated by this one-point-something million individual Internet-connected things. I guess cameras and one flavor of DVR was known to be behind this in addition. But basically IoT-connected devices.

What happened was the source code was released in September. And the presumption was that the author may have done that - now, for some reason I have a "she" tagged to it in my head. I think it may have been a female author who released the source code. And the presumption was that that way there was plausible deniability that this was her attack because, if other people had the source, then maybe it was their attack. Well, today other people do indeed have the source. And in fact what has happened is what used to be one botnet of 1.x million devices has now seen massive fracturing. It is now 52 significantly smaller botnets in a turf war for a relatively limited number of devices that this particular, now available in source form, botnet software is able to commandeer.

So what's happened is people who didn't have the skill or means or interest in writing this stuff from scratch, oh, but they could download it and create their own botnet. And so what's happened is we're seeing more smaller attacks from more smaller botnets, all reusing this same pool of known "how to infect them at a distance" devices. And remember that the Mirai software only lives in RAM. So we know from a couple weeks ago that simply restarting your device, unscrewing your light bulb or unplugging your webcam and then plugging it back in again, that'll flush it. That doesn't solve the problem, though. What it does is it creates a new receptacle waiting to be scanned for and found by the existing bots looking for new brethren. And then they'll jump in there and take up residence.

And presumably, if it isn't already doing it, we would expect that an infected device would close the backdoor. We've seen this certainly in other devices. Code Red and Nimda both did that. After they got in, they shut down the means of entry so that nobody else could get in and fight them on a given device. What a world we're living in these days. Oh, and Leo, my favorite. I don't know who the originator was because I saw it coming from several different sources over the past week. But I just love this. I mean, I liked the acronym IDIOT, I-D-I-O-T, which of course stands for I Don't Internet of Things. But I think even better is this slogan: "The 'S' in IOT Is for Security."

Leo: And the thing's [indiscernible].

Steve: Well, meaning there is no "S" in IOT.

Leo: Oh, right. Oh, I get it, yeah.

Steve: And neither is there any security. It's like, yeah, "The 'S' in IOT stands for security."

Leo: It's for security, stands for security.

Steve: Yep.

Leo: Holy cow. That's a good slogan. I think they should use it.

Steve: So, okay. We had recently a Mobile Pwn2Own, about a month ago, and then just

last week a PwnFest 2016. The headlines were inflammatory. Ubergizmo wrote, "Google Pixel gets hacked in under a minute."

Leo: Yeah. Got my attention.

Steve: And Mashable said, "Google's new Pixel phone hacked in 60 seconds." And so in my opinion, as I mentioned at the top of the show, these attention-grabbing headlines missed the point. But more on that in a second.

Earlier this month Adrian Ludwig, who is the director of Android security at Google, told Motherboard during a security conference that the Pixels are as secure as iPhones. He said: "For almost all threat models, the Pixels and iPhone are nearly identical in terms of their platform-level capabilities." [Buzzer sound] He may well wish to believe that. But we all know that, for security, the proof is both in the design and in the implementation, and only history can be the judge.

And I always, you know, look back at XP, where I remember Ballmer jumping around the stage saying, "Windows XP is the most secure operating system we will ever make" or something like that. And it turned out it was, like, the least secure. It was a disaster for the first several service packs, you know, for years. So far, as we know, history has not been kind to Android. Android has traded openness for security, making a different tradeoff than Apple has, and it struggles to offer both.

But today the truth is it is nowhere nearly as secure in the field as the iPhone. And a group of Chinese white hat hackers hacked a brand new Google Android Pixel in 60 seconds, late last week, last Friday, at the PwnFest hacking competition that took place in Seoul on Friday. The hackers, who work for Qihoo 360, a security solutions company we've referred to in the past, won a nice fee, a tidy $120,000 in cash, after demonstrating an exploit that cracked open the Android and gave them full remote access, as well as access to personal information such as messages, phone calls, contacts, and photos.

My position is, as we know, anyone can make a mistake. And Google is playing security catch-up. And a lot of this is not their fault, you know, things like when they adopted the media library that Stagefright has had such fun with, that wasn't their code. That was, oh, here's a good blob of code. They didn't write it. Unfortunately, there were lots of subtle problems with it. So as a consequence of the way Android has come together, it's taking a while to shake the dust out of it. But what they can, and I think should, be proud of is that they had patched that within a day. And I think that's all we can ask for anyone. I mean, that's the fastest performance we've seen. They also closed a hole that was also found a month before by a different group of white hat hackers at Tencent's Keen Labs who breached the Pixel's security at the Mobile Pwn2Own event in Japan.

So again, I take issue with some security guy saying it's the same as iPhone. Well, sorry. It's not. But they're fixing these problems faster than Apple has responded, typically. So I certainly give them props for that. And I think it's, you know, I know, Leo, that's one of the reasons you've switched to the Pixel. Aside from being a very nice phone, it's become clear that Google's properties, rather than their third- or fourth-generation away OEMs, are getting themselves updated first. The Nexuses were always getting patched quickly, and who knows whether the older ones ever would. And I think that's the best you can do today, given this ridiculously porous security climate that we're in.

I had titled this one "Shove this in your pipe and smoke it" because it refers to piping in

Unix. There is a new tool which I wanted to just put on our listeners' radar because we talked about instantly installing the PiVPN by using sort of the hack, the command line hack of piping the output of curl into the shell. So you literally just give the command curl, space, and then a URL which feeds you essentially a command list. And then the vertical bar takes the output of that and feeds it into the standard input of the next thing in line, which is typically SH, you know, your shell, which then absorbs that and does whatever it instructs.

Well, this raised a lot of people's, we could say, Gibsonian responses because what you're doing, essentially, is you're allowing a remote script to issue any commands it chooses to your shell. And it could do anything. Well, it's certainly a convenience versus a security tradeoff. What was created, and it's on GitHub, is a new intermediate command called TAP, because it literally is a tap on the pipe. So you do the curl and the URL, vertical bar, TAP, vertical bar, and then SH. So what happens is curl pipes this blob into TAP, which creates a temporary file, reads the whole thing in, and opens your default editor - or if you don't have [dropout] in the environment, then it [dropout] vim - and allows you to look at it. You can peruse it, see what it does, make any changes that you want. And when you save and exit your editor, TAP has shelled that process out, so that terminates. TAP resumes and then pipes that edited file into the shell and deletes the file after it's done.

So that has, of course, that allows you to do an inline inspection prior to committing to piping, approving it, essentially, and pushing it through into your command shell. And it has another effect because [dropout] how tricky this was. There were instances where, because curl would be feeding directly into the shell, the delay of the shell accepting the curl output could be felt at the server end, which would allow a clever server to know whether you were probably piping straight through, or whether you were piping into the shell. That is, were you dumping to a file for manual inspection, or putting it through the shell? Because the shell would introduce some delay that a direct write to the file wouldn't.

So it had been noted that, if it saw that it was going straight through, it would give you a benign, clean, non-malicious script. And if it sensed that you were probably [dropout] directly into the shell, it could change what it sent you on the fly and give you something evil. So TAP, because it writes it in one blob to [dropout], it would fool any timing-sensitive server-side software into believing that it is going directly into the, I'm sorry, that it's being written right to a file, so that you would get the benign version, and it wouldn't even try to give you the attack version. So just a cool little widget to add to your command line. It's on GitHub. It's called curl-tap-sh.

Leo: There's a package manager on Arch, for instance, a number of package managers on Arch that go to the Arch user repository, which is a little more risky because anybody can put anything up on there. And the install is run by a script. And most of the installers that support this user repository will load the install script in an editor, well, at least give you the option to load the install script in the editor before anything is executed so you can review it.

Steve: Nice.

Leo: Yeah. So this is kind of known way of doing things. And I think that's absolutely a great idea. Of course you can do it manually. Just download. Instead of curling and

piping to SH, just curl it, edit it, then open it in SH.

Steve: Unix users, however, take as a badge of pride how lazy they are. And so they say, ah, just [crosstalk].

Leo: A lot of stuff on Macs, too. You'll see a lot, like Homebrew's a good example, which is easy to install on the Mac if you just copy and paste the curl pipe to SH command line. It's trivial.

Steve: Yup.

Leo: And I think probably a lot of people would say, well, I could read the script, but it wouldn't tell me anything.

Steve: So OpenSSL has been updated and patched again. This one only affected the v1.1.0. So I just wanted to put it on people's radar. There is now a 1.1.0C, which fixes three problems of high, medium, and low severity, respectively, the most [dropout] critical. And that was a - they call it a DoS. It's unfortunate that we don't have less ambiguous abbreviations because technically it's a denial of service if you crash OpenSSL. So what this is is a denial of service, technically, although, you know, my point is it'd be nice if we had one for remote high-bandwidth attacks of websites to distinguish it from, I crashed the server, so I've denied the services of the server to anybody else who [dropout].

Anyway, there is a relatively new addition to the TLS suite of ciphers, CHACHA20 and POLY1305, which are recent editions. It's a nice authenticated encryption suite, but there was an implementation error which for large payloads allowed an attacker, if they wished, to crash open SSL, thus bringing down the front end of a web server. So if you are using 1.1.0, know that late last week an update was made available, so you may want to get it. But if you're on any of the earlier tracks of OpenSSL, they don't have that latest suite, and so they don't have the bug.

Leo: All right, Steve. I see you pondering with great interest something over there.

Steve: Well, yeah, I'm looking through the state table on pfSense to see what's going on because as I'm listening to you, it was like, you know...

Leo: Chopping, yeah.

Steve: Chopping, exactly. Very choppy.

Leo: Something's going on. We'll figure it out after the show.

Steve: We'll get this done, and I will get it figured out. I'll find out what's going on.

Okay, so Eugene Kaspersky is unhappy, which I thought was sort of interesting for a number of reasons. He has decided to sue Microsoft for anticompetitive behavior in the EU and Russia. And he explained it all in a blog post titled "That's It, I've Had Enough." And they're casting themselves in the role of David to Microsoft's Goliath. And reading through his long list of complaints, I was of two minds. I was immediately put in mind of the Get Windows 10 debacle, where Microsoft was using arguably their right, but also their dominance, you know, [dropout] as the publisher of the operating system that people were choosing to pretty much force people who didn't really strongly resist to do what they wanted them to do.

So Kaspersky enumerates the behavior which he feels is anticompetitive, that is, and this has been - he feels that with Windows 10 the ante has been upped tremendously. Essentially, Microsoft has [dropout] to use Defender. Microsoft really wants their users to use Defender. And in fact he quotes a Microsoft presentation where one of the Microsoft presenters says exactly that. In a presentation titled "Windows 10 - Protecting Device Integrity," the presenter says: "I want you to think about kicking out the third-party antivirus because we've got a great solution right now, and it's going to be even better in the months to come."

And so AV has always been a bit of a challenge because, to do what it wants to do, it needs hooks into the OS. Which means that things that Microsoft changes, which the AV vendors have reverse-engineered in order to get their hooks in, literally, at the low level they need to, those are inherently brittle. And so on one hand I understand what he's saying. On the other hand, he ought to be reading the handwriting on the wall. I mean, essentially they're in an endangered position, and it's not clear to me that suing Microsoft makes a lot of sense.

For example, once upon a time there was a huge industry for firewalls on Windows. That's pretty much gone. Microsoft introduced a firewall in XP, but it was disabled by default. I don't know why. But I remember, I mean, I was talking to Gregor at Zone Labs around that time. And I remember he flew up to Redmond - because Zone Labs was a very popular firewall, it was the one I had chosen and was recommending to people. And he was made quite uncomfortable by the news that XP was going to get its own firewall. But they said, oh, don't worry, don't worry. It's not enabled by default. Users would have to turn it on. And he's like, okay. And then of course XP continued to have problems until, with Service Pack 2, XP's firewall was on by default. And this might be - I always sort of thought this was Microsoft just being very careful about moving forward. I also thought it was them sending up a pretty clear signal that firewall vendors should maybe think about not retiring on their income from firewalls because that may not work.

And I have my own personal experience with exactly this. I will never forget the dinner I had with - I called them "The Brads," Brad Silverberg and Brad Chase. They came down to visit back when I was writing the InfoWorld column, the TechTalk column in InfoWorld. Of course, I was publishing SpinRite back then. And they took me to dinner to tell me about DOS v6.

Leo: That's pretty cool. I didn't know about this. That's cool.

Steve: Yeah. And they were clearly uncomfortable when they said, "Now, Steve, we need to tell you something, but we don't want you to get too upset about it." And I said okay. And they said, "Well, you know, because we have to do this from user demand,

DOS 6 will include something called ScanDisk. But don't worry."

Leo: Don't worry is right.

Steve: "It doesn't do anything like what SpinRite does." Now, I of course knew better. I knew that this was an arrow through my heart because from that moment on, the most often-asked question from those who even bothered to ask was, well, I already have ScanDisk. It came free with DOS.

Leo: Right, right.

Steve: What do I need SpinRite for? Now, of course, they knew because they were technical VPs that it did nothing like what SpinRite does. Did. Does.

Leo: Do what SpinRite did.

Steve: And of course I knew it. But the fact that it was there, you know, changed our ability to sell SpinRite, which seemed to be competing with something that was free, even though it did nothing the same. And arguably customers were hurt because they would run ScanDisk that they already had, and it would [dropout], and they'd [dropout] and reformat their drive. That is, they wouldn't [dropout] data back. You know, it didn't do nondestructive low-level reformatting, didn't optimize the performance, sector interleaf. It didn't do any real data recovery. It was just sort of a better CHKDSK. But it looked kind of the same. And it said ScanDisk. And so, you know, ouch. I thanked them for dinner. I would have happily paid not to have ScanDisk bundled with DOS because, I mean, obviously we've survived because in fact it didn't do anything real. But as we know, marketing is perception.

And so anyway, I just sort of - Kaspersky complaining about Microsoft. Oh, and the other point that I didn't make is I've always appreciated Microsoft's size, that is, the monoculture of one OS. And Kaspersky should really recognize they have been profiting from that for a long time. Because of Microsoft's historic dominance, I was able to write one piece of software which satisfied almost the entire market. That's no longer true, of course. There's lots of Macs and lots of Linux machines. And for a long time Macs were using the PowerPC, so I SpinRite wouldn't easily run on them anyway.

But the point is we've all benefited from Microsoft's dominance in the industry, which gave us a single target for developers to write to. I'd rather have that than 20 different completely random, you know, BeOS and all these other things that tried. You know, Amiga and the Commodore could have gone to Commodore 128. I mean, all of this could have happened differently, and it would have created a much more fractured market. Kaspersky did make one very good point, though, that specifically bears on AV. And that is, the last thing you want in an antivirus cat-and-mouse game in the spy vs. spy or the spy vs. the anti-spy; the last thing you want is a monoculture AV. That is, if Microsoft succeeds in killing off or mortally wounding all of the non-native Defender AV add-on, then the bad guys only have one AV product they need to bypass.

Right now, with there being 20 in a relatively active antivirus market, the virus's job is much more difficult because they don't know which AV package a given user will have,

and they need to be - they need to arrange to get around all of them. And it is the case that Microsoft, at least initially, normally doesn't have best-in-class results. They typically get there eventually. But they don't start out [dropout]. But I think AV is a particular case where, and Kaspersky does bring this up, not specifically the threat of a monoculture, which I just grabbed onto when I saw that comment.

But I think that's really important. You'd like to have more variety because you're just going to get more safety that way. I just use Defender, though. I mean, you and I, Leo, have never been huge AV proponents. We believe in maintaining safe borders and being very careful about safe behavior. And also, you know, saying a prayer every so often.

Leo: Yeah. And I believe in not using Windows, which helps me quite a bit.

Steve: Yes, yes.

Leo: That's a handy tool.

Steve: And I use DOS, so I'm in good...

Leo: Yeah, I don't think there are any DOS viruses. Well, there were. Not anymore.

Steve: Not anymore. Okay. So, miscellany, three fun bits. I noted just yesterday as I was pulling this together that "Westworld" on HBO has been renewed for a second season. And it just is - it's a delightful 60 minutes on Sunday evenings. I really enjoy it.

Leo: Now I'm worried because I was hoping they would resolve it in the first season. Apparently there will be no incentive.

Steve: I had exactly the same thought. Can you say "Game of Thrones"?

Leo: Yeah, yeah.

Steve: Which just goes and goes and goes.

Leo: It kind of had to happen; right? I'm not surprised.

Steve: Yeah, it did. And maybe, I mean, it is a rich medium. Maybe they will solve this villain of whatever it is. It's still sort of unclear what's going on, but it's interesting. And then give us another one, you know, next season.

Leo: Right. It's kind of, yeah, there's endless. Although one of the big fan theories

kind of came true last night, that I thought was very interesting. We'll take more.

Steve: I did see, I saw with a buddy of mine on opening day last Friday "Arrival."

Leo: Oh, I'm dying to see that. Mike Elgin said he loved it.

Steve: It was a very credible and well-assembled First Contact movie. As I was watching it, I was thinking, okay, there's nothing wrong with this. I mean, they're not doing anything wrong. And some of the military stuff might be a little over the top. But on the other hand, if 12 scary big black eggs sort of all descend on different areas of the world and float a few feet off the ground and don't seem very friendly, you can imagine. I mean, I'm surprised more people weren't shooting missiles at them to see what would happen. But anyway, I just wanted to say there was some of it that was a little confusing because I considered it a "Close Encounters of the Third Kind" meets "Slaughterhouse Five." And if any of you were forced to read Kurt Vonnegut's book in high school...

Leo: Forced? That was a good book. You didn't like it?

Steve: Everything was forced in high school. But, yeah, I did actually like it. And the concept was interesting, too. So there was a point where I was worried about the plot, and then it resolved. And it's like, oh, okay, this explains what seemed to be a big problem for a while.

Leo: Elgin's recommendation was see it quickly because you don't want the spoilers.

Steve: Yes. I had exactly the same thought is there is a huge danger of someone saying someone you wish they hadn't said. And then it's like, ooh, darn. And of course we don't do that here.

Leo: Right, no.

Steve: Okay. Dark matter. I won't take much time on this, but it's been a big concern for several decades. It just seemed wrong, as you commented also, Leo, at the top of the show. So, okay, here's the problem. We think we know how gravity works, even though its theory has also been acknowledged to be at odds with that of quantum mechanics. That is, they cannot both be right because they're in disagreement with each other. So that's a problem.

The trouble is that we observe galaxies rotating. We can determine the rate of spin based on Doppler shift from the opposite edges that we can see, one side coming toward us, the other going away from us. That allows us to infer the rate of rotation. And we can see how, based on distance and size, we know how big they are. The problem is they are spinning too fast. That is, we know how fast they're spinning. We know based on their contents how much mass they have. So they should fly apart, meaning they're spinning too fast for their own gravity to hold them together. So the only thing we've been able to

do, cosmologists, is say, okay, there's got to be some dark matter somewhere. You know, there must be a lot of it.

It turns out that we need a whole ton of it. Eighty percent of the universe's mass would have to be dark for what we observe to be correct. And then, after deciding that, they went looking for dark matter, you know, people who smash stuff together and do all this, fancy tests and looking for dark particles and all that. Never have they seen any sign of it. No particle, no indication it exists. Nothing other than we need it to exist. So let's go find it. And they haven't.

So nearly seven years ago, back at the end of 2009, a Dutch theoretical physicist by the name of Erik Peter Verlinde - who is at the University of Amsterdam's Institute for Theoretical Physics, and also he's got a great rsum. He was at Princeton and permanent staff at CERN. And, I mean, this guy knows what he's doing. And he is a professor and has a permanent teaching position at the Institute of Theoretical Physics in Amsterdam. He introduced a theory that he named "entropic gravity."

Now, I have no idea what the following means, but it's cool-sounding. According to this theory: "Gravity exists because of a difference in concentration of information in the empty space between two masses and its surroundings." Okay, got that? A difference in the concentration of information.

Leo: Whoo, whoo.

Steve: He also extrapolates this to general relativity and quantum mechanics. He said in an interview at the time: "On the smallest level, Newton's laws don't apply, but they do for apples and planets. You can compare this to pressure of gas. Molecules themselves don't have any pressure, but a container of gas does."

And back then, in his 2010 article on the origin of gravity and the laws of Newton, Verlinde showed how Newton's famous second law, which describes how apples fall from trees and satellites stay in orbit, meaning sort of at local scale, or this scale, those can be derived from the underlying microscopic building blocks which he has articulated. Extending his previous - and this extends his previous work and work done by others. He now shows how to understand the curious behavior of stars in galaxies without adding any of the presumed, and still missing, dark matter.

Last Tuesday, a week ago on November 8th, he published a new paper showing how his theory of gravity accurately predicts the velocities by which the stars rotate around the center of the milky way, as well as the motion of stars inside other galaxies. He wrote: "We have evidence that this new view of gravity actually agrees with the observations. At large scales, it seems, gravity doesn't behave the way Einstein's theory predicts."

And finally, at first glance, Verlinde's theory presents similar features to a common modified theory of gravity known as MOND, M-O-N-D, Modified Newtonian Dynamics, which we've had since - it's been around since 1983. However, the problem is that this Modified Newtonian Dynamics tuned the theory to match the observation; whereas Verlinde's theory starts from first principles and apparently arrives at correct results without any tweaking at all. So I just thought that was important enough to put on everybody's radar, that this whole question, this problem with dark matter, it may just be the fact that our model of gravity has been wrong, and now there'll be lots of testing and verification and so forth to see whether this holds up. But if so, that's a big step forward, and it's a big "whew" for all of the people who were concerned that...

Leo: Yeah, makes sense.

Steve: It really does make sense. This idea that there was 80% of something that we were not seeing is like, okay, well, where is it? Why not? And that's a lot, you know, that's four-fifths of everything.

Leo: Awesome.

Steve: Leo, there's a fun video in this next link. Turns out there's actually a World Cube Association to manage the Rubik's Cube, of all things. And there's a well-known German manufacturer, Infineon, which is promoting their technology for vision recognition, for promoting automated driving subsystems, which they say need to offer very low latencies and absolutely reliable and quick technology. So to demonstrate this at a recent trade show in, I want to say Munich - oh, yeah, the Electronica tradeshow in Munich - they built a one-off Rubik's Cube-solving robot. And the picture is just wonderful. So we know what a Rubik's Cube is; right? It's a cube with six sides. And, okay, now that was it solving the puzzle.

Leo: What, no, wait. What? No.

Steve: Yeah, it's ridiculous.

Leo: So this is the slow-mo 12x...

Steve: Then they slow it - yes.

Leo: Holy cow. It takes how long? Less than a second.

Steve: .637 seconds.

Leo: To solve any arbitrary Rubik's Cube.

Steve: Yes. So what they did is they cover the camera's shutters, that is, they shutter the cameras. Then they randomly scramble the cube so that the computer can't see what they're doing. Then they uncover the shutters. And then they say go.

Leo: Ready? Ready? Watch. Done.

Steve: Oh.

Leo: Whew.

Steve: That's so neat. And at first I'm thinking, now, wait a minute, does that really work? But when you think - so for the people who don't have video, we know a cube as six faces, and so six sides. So if you attached rods to the center face of each side, and those hooked to heavy-duty stepper motors - so you've got this enclosure with six stepper motors, each hooked to a rod, and the Cube floating about a foot away from them all in space, suspended by these six rods coming in.

Indeed, if you rotate a rod, that will rotate, because of the internal mechanisms of the Rubik's Cube, it will rotate that face. And as long as you bring it back into alignment, which you can do with a stepper motor by stepping it exactly the number of steps required to go 90 degrees, then any of the other ones are able to spin. And I actually think that in some cases it's spinning opposite faces at the same time because that you can also do without breaking the Rubik's Cube property. They did use a friction-reduced cube in order to get better speed. But it turns out that the previous record - I mean, they broke a record for automated Rubik's Cube solving. The previous record, and this boggles my mind, is a 14-year-old kid in 2015 who got the record at 4.904 seconds, and then another youth completed the task in 4.74. So, I mean, just, boy, you know, that's crazy. Anyway, it was just a cool little bot that I wanted to share with people.

Leo: It's amazing.

Steve: And I have found in the mail bag yesterday a really nice note from a Paul O. Kirwan in the UAE. And the title caught my eye: "SpinRite in Riyadh, the World's Largest Airport Project." He said: "Hi, Steve. Just a testimonial and a thank-you for your product. I first came across it back in the 1980s when we were opening Riyadh Airport. At that time we had a small selection of PCs" - okay, right, 1980 circa PCs - "for VIP" - then he has in parens - "(the Ruling Family and Air Force Generals) staff, and back then the drives were very unreliable." Amen. "A combination of heat, dust, cigarette smoke" - he noted, he says, parens - "(they all smoked heavily) meant it was a full-time job to keep these things working. And nothing was able to do that except SpinRite.

"I now own my own copy; and, believe me, it has saved my bacon more than a few times. I have used it many times since, and now for preventative maintenance. I talk about it because it's often amazing, and people are skeptical when I tell them that a 30-plus-year-old piece of software" - well, I have been keeping it alive a period of time. It's had several major improvements since then. But, yes, it was written originally back then, back when the Brads scared me by telling me they were going to include something that was going to confuse people in DOS. "A 30-year-old piece of software," he writes, "can recover their drive." So that means he's been doing it for other people, which technically I would hope he would suggest maybe that they should use it themselves for preventative maintenance.

Anyway, he says, "They are always impressed when it does recover their drive. Thanks again. The interface is so familiar I can almost run it blindfolded. Thanks to you and Leo for a great podcast, and thank you for such a great and enduring product." And Paul, thank you.

Leo: Nice.

Steve: So three final things. This is the one I said we might as well just give up. Just, you know, just fish. Go to a creek, listen to crickets. Because if this is possible, it's over. WiFi signal interference can leak your passwords and keystrokes. I'll just read the blurb from the beginning of the abstract of the detailed technical paper.

They write: "In this study, we present WindTalker, a novel and practical keystroke inference framework that allows an attacker to infer the sensitive keystrokes on a mobile device through WiFi-based side-channel information." We all know what that means. "WindTalker is motivated from the observation that keystrokes on mobile devices will lead to different hand coverage and finger motions, which will introduce a unique interference to the multipath signals and can be reflected by the channel state information," which I'll explain in a second, CSI.

"The adversary can exploit the strong correlation between the CSI fluctuation and the keystrokes, to infer the user's numeric input. WindTalker presents a novel approach to collect the target's CSI data by deploying a public WiFi hotspot. Compared with the previous keystroke inference approach, WindTalker neither deploys external devices close to the target device nor compromises the target device. Instead, it uses the public WiFi to collect user's CSI" - again, channel state information, which I'll explain - "data, which is easy to deploy and difficult to detect." And then I put in here a note: Side channel attacks, being typically passive, are by their nature almost always impossible to attack.

Then they say: "In addition, it jointly analyzes the traffic and the channel state information to launch the keystroke inference only for the sensitive period where password entering occurs," meaning [dropout] when that's happening. "WindTalker can be launched without the requirement of visually seeing the smartphone user input process." They say "backside motion." I don't know what they mean. The guy's butt? Anyway, "or installing any malware on the tablet." Probably a language barrier thing.

Leo: I'm sure.

Steve: Their backside motion. "We implemented WindTalker on several mobile phones and performed a detailed case study to evaluate the practicality of the password inference towards Alipay, the largest mobile payment platform in the world. The evaluation results show that the attacker can recover the key with a high success rate."

We take for granted this stunning WiFi technology that we have been given. It's a black box. What we've gone to is this multiple antenna MIMO, the multiple input/multiple output technology. And we take for granted what's in there because it just works. But to get it to work, to achieve the data rates we are now getting, at the reliability we are getting, in this random heterogeneous environment that we are in, requires an insane amount of technology that we don't even see. And it's been integrated onto a chip, so it doesn't even cost anything. But it's still there, and it's called "channel state information."

What happens from instant to instant is the receiver of the WiFi signal is acquiring and digitizing a phenomenal amount of information about the separate signals being received by the individual antennas, the receive antennas on the device, and like the relative phasing of the signal, and the antenna-to-antenna signal strength, and even arrival time within the phase, in order to do things like beam forming, and in order to ignore off-

access noise. The point is there is an insane amount of technology that we don't even see, but it's there, and it's what has enabled our current technology to work. And that CSI, as it's called, this channel state information, is published in an API which is open because no one thought it could be abused. And so it is possible for software to treat that as metadata.

That is, what these guys found is that, if someone is holding their smartphone, which has a WiFi link to an access point where the CSI information is being processed as side-channel information, they're holding in their hand a transmitter. And as they use their other hand to touch the screen, that creates enough movement, enough variation from event to event that their actual other hand motion can be resolved by the CSI information. Now, yeah. In an environment where lots of people are moving around, that's less easy. It probably needs to be, you know, you're going to have probably some requirements for it to be physically quiet in terms of other movement, although it's the hand relative to the WiFi transmitter in the phone that is what modulates that phone's transmitted signal, which is then received by the access point.

And thanks to this crazy digitization of just amazing richness, of instant-to-instant information about the state of the WiFi signal received individually by all of the receiving antennas, that's side-channel. And these guys turned it into a not quite 50/50, I think it was like 48% recognition rate. But still, from zero, that's worrisome. So as I said, we should just unplug and just say, okay, that's, you know. Now, I mean, like if our movements in a WiFi field can give us away, there's no hope.

Leo: This is like Van Eck phreaking, though. I mean somebody's going to have to have some sophisticated hardware; right?

Steve: No. That's just it. Consumer routers all have it.

Leo: No, I'm saying to do it, though, to do this. No?

Steve: No. No. It's built into every - every multi-antenna router has this. Now, you would need that software added to the access point that the phone is connected to.

Leo: Okay.

Steve: So, for example, the Starbucks router would need to get taken over. But, oh, gee, who ever heard of a router getting taken over? But my point is it's in the hardware, Leo. It's there, and it's public. So the software simply needs to do it. And in fact, these guys, they didn't make any hardware. They used an Intel 5100 series WiFi chip on Linux. And the Linux driver has access to the API that receives the side-channel information. Zero hardware overhead.

Leo: Crazy.

Steve: Isn't that amazing?

Leo: Crazy.

Steve: Wow.

Leo: Wow.

Steve: Yeah. Okay. Now this one is crazy, speaking of crazy. Popular Mechanics covered the story, put me onto it. And I've got the Kickstarter link in the show notes at the end of the story, Leo. It's called Taps, T-A-P-S. They call it a Touchscreen Sticker with Touch ID. Okay. I followed a link which one of our listeners thought I would find intriguing, and indeed. It's in the show notes. You're hearing it because I thought it was really interesting. So listen to this. It's not what I thought it was. And it's clever.

"A company named Nanotips thinks it can solve the annoying problem of removing gloves," meaning needing to remove gloves, "to access your fingerprint-sensor locked smartphone in the winter. The product, Taps," writes Popular Mechanics, "is surprisingly lo-fi. It's essentially a fingerprint-shaped sticker made of military-grade polyurethane..."

Leo: Oh, no.

Steve: "...that you can" - oh, yes. And so I'm thinking, oh, my god, so you have to train it with your fingerprint? No, no, no. It has somebody else's fingerprint? No, no, no. This is where I thought, this is kind of clever - "that you can stick onto the end of any glove."

Leo: Oh, yeah, and then you train the phone with that.

Steve: Exactly.

Leo: Right, right. So it's an additional fingerprint.

Steve: It's a cyborg fingerprint. "This fingerprint isn't yours. It's a synthetic individual fingerprint that you can train your phone to recognize the same way you would your own."

Leo: Wow. It's eight bucks, by the way, which is why I tweeted it this morning, because I thought our listeners might get a...

Leo: This is clever. I like this.

Steve: I think it's clever. "These synthetic fingerprints don't actually look like fingerprints, but they function the same way as any real finger" - they actually look like

sort of a stippled bump map - "creating a recognizable pattern that your phone can remember and use in the future. Most apps allow at least one or two fingerprints" - and, you know, iOS lets you have five - "to serve as keys, so Taps merely replaces one of your own fingerprints as a biometric password." Well, that's an abuse of the term, but we know what he means.

"Of course, this technology presents a massive security problem. Suddenly, anyone with your glove can access anything you have locked this way on your phone."

Leo: Oh, that's a good point.

Steve: Yes. "This design flaw eliminates most of the usefulness of Taps, as the reason to use a fingerprint key in the first place is generally that it's more secure than a password as it's unique to you." And we presume every one they produce is also unique, so that not all of the finger - I'm sure they are. But I'm saying that's another point of concern. You might want to, like, maybe scrape off a few nubs.

Leo: I'm not sure they are. This doesn't sound, I mean, really?

Steve: Well, they do expressly say that they are unique.

Leo: Okay. Oh, good.

Steve: Yeah. And it looks like the technology they have for making them would make it so. So anyway, just to finish, they said: "No one can steal your fingerprints, but they could definitely steal a glove with a synthetic fingerprint stuck to it. However," they conclude, "convenience is a great motivator. Taps is currently on Kickstarter, where it has raised 2,000 over its 5,000 goal." Now, that may have been true at the time the story was issued. This morning it was at 11,000, and now it's at 13. So I have a feeling that my tweet had a nice effect also. Again, I think it's $8 each, but they sell them in sets of four. And I'm not suggesting it's anything more than an interesting toy. But for eight bucks, or, well, 22 for more than you probably need, I just think it's kind of cool. And maybe there is a use. Think about leaving the fingerprint, like in a safety deposit box for some reason. And I was thinking, okay, like if you became deceased. But then on the other hand, someone could press your dead thumb on your phone in order to...

Leo: Most phones, I think, have some infrared sensing to prevent dead thumbs from working. But I might be wrong on that.

Steve: Ah. I wonder. Because then this wouldn't work if it was through an insulated glove.

Leo: Oh, then you're right. Maybe they don't. Maybe I'm mistaken. Taps.

Steve: I think we talked about that when we were initially talking about Touch ID

spoofing, you know, to look for a pulse or look for heat or something. But anyway, I just wanted to let our listeners know, I'm not, again, yes, recognize it's a security problem, but I just think it's kind of cool, just an interesting hack.

Leo: Wow.

Steve: And the idea, again, we're in California, Leo, so gloves are not a big problem.

Leo: Not a big deal for us, yeah. There have been gloves around before with silver in the glove that lets the capacitive screen work, but it doesn't solve the fingerprint problem.

Steve: Right. And so with the understanding that you've extended your identity out to your thumb, you could of course use it during the winter, or use it during a ski trip, and then wipe that fingerprint out of your phone so that it's no longer a danger. So I can see some use cases, and I wanted - I just thought it was a cool hack, too.

And, finally, the BlackNurse. Lord knows why it got that name. I looked for any reason that it was called that, and I couldn't find any. There's an interesting attack because this allows one PC, one host on the Internet, one light bulb, one particularly bright light bulb that has - apparently it needs 15 to 18Mb, but not 600Gb. That is to say, a radically low, compared to what we're used to, bandwidth [dropout]. A decade ago, when we were talking about denial of service attacks, I remember that we talked about small packets because there are two ways to bottleneck a router. You can bottleneck the bandwidth of its connections, that is, simply send in more packets than it can eliminate.

Remember that a router typically has multiple [dropout], and then it's sending packets arriving from multiple sources, originating from multiple sources, down to a single destination. But if the links are all the same bandwidth, it might have, for example, four one-gig incoming links and one one-gig outgoing link, which is normally fine. But if all four incoming one-gig links were fully saturated at one-gig, and they're all trying to have their traffic sent out of the remaining one-gig link, well, you've got four gigs coming in. It can't fit into that one-gig outbound link. So the outbound link buffer is forced to throw the majority of the packets away. And that is a denial, a bandwidth DDoS, typically distributed denial of service because you've got lots of sources, all generating traffic aimed at the same spot.

But the other thing, the other problem that routers historically have is the rate at which they can switch packets. Not the getting them out once they're in, but they have to inspect the packet in order to figure out where it goes. And normally routers are specified toward the largest possible packet size because the Internet maximizes the size of packets where it can. TCP uses the largest blocks that it's able to, as does UDP. And so but there will always be a mix of smaller packets. For example, ICMP is just like the famous ping. It's just it's a very, very small packet. And by its definition, a TCP SYN packet, the synchronized packet, is super small.

And remember back in the day the SYN flood. Well, it turns out that many SYN floods that got to their host computer would crash the TCP stack. But it also was the case that many SYN floods never had a chance to get to the host computer because they were so small that many more of them would fit in a given amount of bandwidth per second. So think about that. If you've got a 1,500-byte packet, then there's a maximum rate at

which those can arrive over a certain bandwidth. But if you've got a 15-byte packet, you can fit a hundred times that many in the same period of time over the same bandwidth.

So after becoming very sensitive to the issue of packet switching rate, I look at, when I look at switches or routers, they'll talk about whether they can do minimum size packet line rate switching, meaning that, if packets of the smallest legal size arrive at the maximum rate that the interface will run, is what's called the switching fabric, is the term, is the switching fabric able to quickly enough inspect the [dropout] send it out to its destination. And once upon a time that was not the case. There's been lots of advances since then. But you can't always treat all packets the same.

What has happened is [dropout] have gone up. Routers have had to become much smarter. They have a hierarchy of paths. They have the so-called fast-switching path, where essentially hard-wired electronics is able to make an instantaneous electron speed decision about what to do. And, for example, responding to a ping might be an example. Ping comes in, that's trivial to respond to. It just swaps a couple bytes, changes the type code from an echo request to an echo reply, and sends it back out to the place it came from. So that requires no thinking.

The problem is there are some types of packets that require some thought. And the BlackNurse low-bandwidth packet-rate attack leverages that. They use an ICMP type 3 code 3 packet, which is the type 3 is the generic class of Destination Unreachable. That's one of the - ICMP itself is meant for sort of managing the plumbing of the Internet. That's what the ping is, is a type of ICMP packet. And the Destination Unreachable is something which, for example, a router will send when there's a problem. And a classic problem is the traceroute.

A traceroute deliberately sends packets with short times to live, TTLs of, like, five. And so after the packet has gone over five routers, the TTL has been decremented to zero. Then the router sends back a Destination Unreachable timeout, saying sorry, this packet died, and the Internet requires me not to forward a packet whose TTL is expired. If that weren't rigorously honored, packets would flow around the Internet forever, and then we really have a problem. So everybody honors that.

Well, this particular packet, the Destination Unreachable/Port Unreachable, the type 3 of the class 3, must be handled. RFC 1122, which is one of the old ones, one of the fundamental foundational RFCs, the Request for Comments, says: "A host should generate Destination Unreachable messages with code 2, for Protocol Unreachable, when the designated transport protocol is not supported." So that's another example of a plumbing. It's like, sorry, you want TCP, but I don't have any. And so it sends back something on ICMP that by definition everybody must support. Or, continuing, it says: "...or 3, Port Unreachable, when the designated transport protocol, for example UDP" - or maybe TCP - "is unable to demultiplex the datagram, but has no protocol mechanism to inform the sender."

So, for example, "A Destination Unreachable message," they continue, "that is received must be reported to the transport layer. A transport protocol that has its own mechanism for notifying the sender that [dropout] unreachable, for example TCP, which sends reset packets, must nevertheless accept an ICMP Port Unreachable for the same purpose."

So what that says is, in the fundamental wiring of the Internet, when a router receives that particular - an ICMP type 3 code 3, it cannot deal with it itself. It has to attempt to forward it. And what that practically means is that it's - a packet falls out of the fast switching fabric into the processor, and there we hit a packet-rate limit. So what has been found is this is an ongoing attack. There are attackers in use now that are sending

ICMP 3,3 packets where just 15 to 18Mb is able to take a major site offline because there is not a way to short-circuit the processing of that. It is crucial to have it handled, so you can't put in a filter rule to block it and drop it, which they would like to. You have to pass it on. Otherwise all kinds of other stuff in the Internet would break. So an interesting new way of bypassing old safeguards and essentially bringing a major provider down with a single host over the Internet. Unfortunately, these guys are getting very creative.

Leo: Wow, wow, wow. And that, my friends, is the BlackNurse Attack. That's what that is.

Steve: For some reason.

Leo: For some reason no one will ever understand. Well, we have run out of time. But, fortunately, it's exactly when you've run out of material. So it all works out so nicely.

Steve: Funny how that works.

Leo: He's a master, my friends. Our show, of course, is every Wednesday, 1:30 p.m. Pacific, 4:30 - did I say Wednesday? Tuesday.

Steve: No, no, unlike dark matter, which I'm not sure they care about, but I just thought...

Leo: I care. I care. I thought that was fascinating. I don't understand what "information" means in that context. But I know there's information in the event horizon of black holes; right? They talk about...

Steve: Can't get out. Can't get out.

Leo: Yeah, talk about information not getting out. So this is some use of the word "information" that physicists agree on that I don't understand. But that's fine. It's good enough for me. It's all about the information. That's what you get here every Tuesday, 1:30 Pacific, 4:30 Eastern, 19:30 UTC. If you'd like to tune in and watch live, we'd love it. But you don't have to because we make this show available every possible way. Steve's got 64Kb MP3 audio on his site, GRC.com. He also has a transcript. Elaine Farris writes everything he says down. By now she's probably dreaming of you. She goes to sleep, and Steve's talking about dark matter and stuff. So you can get that at GRC.com.

While you're there, pick up SpinRite, Steve's bread and butter, the only thing he charges us for. And yet, and yet it's worth every penny. GRC.com. Get SpinRite, the world's best hard drive maintenance and recovery utility, and plenty of other free stuff. Steve makes it all available at his website. He also has a feedback form there, and maybe next week there'll be no BlackNurse, and we'll be able to do a Q&A. So

go to GRC.com/feedback, or tweet him. He's @SGgrc and accepts direct messages from anyone on the Twitter: @SGgrc.

The home page for this show on our site is TWiT.tv/sn. It's also on YouTube.com/securitynow if you want to show your friends or share a link. And of course you can subscribe at any podcast application because it's there. And a lot of people collect every episode. Collect all 586, kids. Thanks for listening; and, Steve, we'll see you next week. And soon we'll see you in-studio.

Steve: Yes, sir. And soon you're going to be having a birthday, two weeks from today.

Leo: I is.

Steve: Yes, you are.

Leo: I is. And I understand you might be in town for that. Next week, by the way, day before Thanksgiving. No, two days before Thanksgiving. So we'll have a turkey. We should do the Turkey of the Week.

Steve: Rather than be a turkey.

Leo: Yes. I'd rather eat one than be one. Thank you, Steve.

Steve: Leo.

Leo: See you next time.