SMTP STS

Description: Leo and I discuss the outcry following the "60 Minutes" high-visibility demonstration of real-time cellular phone hacking. We also cover the news of the Canadian RCMP having BlackBerry's master decryption key; the end of Apple's QuickTime; what the FBI found (or didn't) on the San Bernardino attacker's phone; and a revisit of Threema, WhatsApp, and Signal. Then, after a bit of miscellany, we take a look at a newly proposed specification for increasing eMail security known as "SMTP STS."

High quality (64 kbps) mp3 audio file URL: http://media.GRC.com/sn/SN-556.mp3
Quarter size (16 kbps) mp3 audio file URL: http://media.GRC.com/sn/sn-556-lq.mp3

SHOW TEASE: It's time for Security Now!. Steve Gibson is here. He's upset. He's annoyed. He's very dismayed by this proposal to somehow glom STS, Secure Transport, onto SMTP, the email server. No, he says. We'll find out why. And we'll also talk about the latest tech news, including that "60 Minutes" report last night on cell phone hacking. It's all coming up next on Security Now!.

Leo Laporte: This is Security Now! with Steve Gibson, Episode 556, recorded Tuesday, April 19th, 2016: SMTP STS.

It's time for Security Now!, the show where we protect you and your loved ones - what are you laughing at, Steve?

Steve Gibson: You're rocking the baby?

Leo: I'm rocking a baby, showing how we protect babies.

Steve: You know what that reminds me of? That reminds me of, if anybody - we've talked about "The Good Wife" a couple times.

Leo: Love that show.

Steve: But, oh, Leo, last Sunday, oh. It was just - it was so, I don't know who - I don't know if they, like, change writers, or if the writers know they only have three more to go

and so they're just giving it - they're going out with a blast. But it was so full of little extra wonder, I just - it was just spectacular. And the one guy, you know, holding the dog, just - he just walks around with this little dog, with his arm - with his paws up in the air.

Leo: Oh, how funny.

Steve: And it just, oh, gosh. So are you behind in...

Leo: Oh, I'm years behind. I'm only on Season 4, I think, so...

Steve: Oh, it doesn't disappoint. It continues to be good. And this is the last season. They have three episodes left, or remaining, of this final season. But, oh, boy, is it wonderful. So I'm not sure what you were rocking, but it reminded me...

Leo: It wasn't a dog. But I did not yet tell you that that voice you hear is Steve Gibson. He's from GRC.com. He's our security guru. He's been doing this show for 10-plus years, 11 almost, and knows everything there is to know. We just - actually, you probably have, almost 11, because we just celebrated our 11th anniversary of TWiT.

Steve: I think we're a few months away. But, boy, that 11th year went fast. Because we were just...

Leo: I know, we just did the 10th, I know.

Steve: Yeah.

Leo: So let's see. So I started TWiT in April. First show was April 17th, 2005. And I was still going up to Canada. And I think I was in Canada, I said, Steve, you and I should do a show.

Steve: It was between our - we would shoot, what was it, like four episodes of Call For Help.

Leo: It was horrible.

Steve: And it was between [laughing] - it was between those episodes. I would come up and sit down with you and do things on a whiteboard because that was, you know, it was a TV show, and so we could count on visuals. And so I'd set up a whiteboard normally in between. And so you said to me - because you, basically, you worked for one week out of the month, except for the weekends when you were doing The Tech Guy. So you had all this spare time. Meanwhile, podcasting was happening. And so you said to me, "What

would you think about doing like a weekly podcast on, like, Internet security?" And I said, "A what cast?"

Leo: Really? You didn't know what I was talking about?

Steve: And you said, "You don't know about podcasting?" I said, "No, nobody does."

Leo: Nobody did. It was April - August 2005.

Steve: I'll never forget, I was watching MSNBC, and Chris Matthews found out he had one. And he was reading the teleprompter. And he said, "And also, check for our - our what? Our-our podcast?" And then he, like, looks off-camera. "What's a-a pod-podcast? What's a pod..."

Leo: Oh, I've got to find that. I wonder if I could...

Steve: "I guess we have something called a podcast."

Leo: I would love to find that somewhere. That's funny. That's funny.

Steve: Anyway, it's turned out to be the best thing that ever happened, so...

Leo: I love it. I mean, it is one of our absolute top podcasts. We've started our own rating system because we haven't been able to find a reliable one. And so we're getting now finally consistent, reliable numbers. And as I see these numbers come in, it's just really mindboggling how many people listen. It's something, I can't remember what it is, but it's something like 80,000 a week listen every single week to this show. And that's an amazing number. So, well done.

Steve: Well, thank you for suggesting it, and for creating the forum, and setting it up, and the links and the bandwidth and everything else that you do. I'm happy to bring...

Leo: Plus that's 80,000 people a week who are protected a little bit from the perils of the Internet.

Steve: Well, and we have some more of that this week.

Leo: Oh, really. What a surprise.

Steve: On Sunday, before the crazy fourth-to-the-last episode of "The Good Wife," "60 Minutes" aired one of their three pieces that they do every Sunday. The last one of the

three was about cell phone hacking, and it generated a flurry of information. Now we've got the Senate worried because, you know, basically they shined a bright light on something which we had already talked about, back when it was news, which was the summer of 2014, the Chaos Computer Convention had an expose on this. And we talked about the baseband processor in cell phones, which gets very little attention. And back then we talked about how, you know, this is something no one's looking at, and it's a big security problem. So we have to talk about that, sort of to follow up on what "60 Minutes" talked about and to give it a little additional fluffing up and context.

Leo: Well, and what's interesting is that we talked about it a couple of years ago. It's not improved. And it probably will never be improved. Those things just don't get updated.

Steve: Actually, as I was thinking about this and looking at all of the attention it got, I thought...

Leo: Maybe it will.

Steve: ...the NSA, if they were dead, they would be turning over in their grave. They're not.

Leo: Please don't tell people about this.

Steve: They're not dead, and they are not happy.

Leo: No.

Steve: Because this is not something, I mean, it is so potent that they're like, ooh, no, no. And the good news is nothing will happen. Nothing will change.

Leo: It's hard to do; right? It's hard to fix.

Steve: Yes. It is. It is. In fact, it's funny because the subtitle of this week's episode about a different technology is "Horrible Internet Kludges Never Die." And similarly, you know, horrible, really old telco kludges never die. I mean, you just - you can't change the way this stuff that is so deeply burned into the infrastructure is built. And what "60 Minutes" was talking about, and we'll be coming back to this when we're actually supposed to be talking about it, is a signaling system which glues individual providers together. And it's a deep, rich API that presumed nobody had access to the network. Well, once upon a time nobody did. Then they said, oh, look, Internet, and put everything on the Internet, so everybody has access to the network. But nothing else changed. Anyway, we'll get there in a second.

I wanted to talk about a recently released RFC, sort of perversely called Request For Comments. The only comment I had was a typo where they got the direction of

something backwards. And I thought, am I going to bother telling them? And I thought, no, a whole bunch of other people will tell them. Anyway, this is about something which is definitely right in our wheelhouse, and that's the acronym, or the abbreviation, is SMTP STS. STS, of course, stands for Strict Transport Security, or sometimes Strict Transport Secrecy. I guess I like secrecy. I'm not sure which I like better because it is really about encryption, so that's secrecy. Security requires a large - is a bigger, has a bigger set of requirements.

But of course we've talked about HTTP STS, that is, enforcing HTTPS, essentially, on the HTTP protocol. And this, unfortunately, tries to bring the same thing to email transport, that is, SMTP, which is Simple Mail Transport Protocol. The bad news is what we really want is a good solution which DNS Security could provide, and so much more. But it's sort of not happening because there's a lot of - a lot has to happen in order for us to have any DNS security. Every piece of the system has to be in place, or we have none. So...

Leo: Are we moving in the direction, at least?

Steve: Yeah, I mean, yeah, the root servers finally got signed, and that was a big deal. But the deployment has to happen. And unfortunately, DNSSEC is an all-or-nothing. And so it's like, okay, when? So in the meantime, this is an interim horrific kludge meant to kind of improve the security. And the problem is in order to do it, it is just a godawful mess. And as I'm reading through this, I'm thinking, oh, my lord.

Well, anyway, so I'll explain all of this at the end, after we cover all of the news. And it's the kind of thing where I just hope this doesn't happen because the subtitle is "Horrible Internet Kludges Never Die." And if this ever does actually happen, then it'll always kind of be around. Even when we don't actually need it any longer, some people will keep using it. And this'll be like, and it's just awful. So anyway, so we'll have a lot of fun today. We'll talk about that "60 Minutes" expose. I wanted to call this "Black and BlueBerry."

Leo: I know why. I know why.

Steve: Yes, because it turns out that it looks like the Canadian law enforcement had the master key for BlackBerry for the last six years, and then tried to suppress the court documents that disclosed that fact because they wanted to keep it a secret. Then of course I also wanted to call this "QuickSand for QuickTime" because QuickTime is in trouble. And we'll talk about some of the details that you didn't cover on MacBreak Weekly because that's what we do here, like the nature of the two heap overflow, heap buffer overflows in QuickTime that require all Windows users to pull it; what wasn't found in the decrypted San Bernardino phone; a little conversation about Threema relative to WhatsApp and Signal because the Threema guys weren't happy with all of the great press that WhatsApp got as the Signal protocol was finally incorporated into it, and so they produced a big marketing checklist that has all green happy checks for them and all red unhappy X's for WhatsApp.

Leo: Oh, dear.

Steve: And it's like, eh, okay, you know. Like some of them are, okay, we have less lint in our navel than they do.

Leo: Yeah.

Steve: It's like, okay, but that just doesn't have any real bearing on, you know...

Leo: Big problem with Threema is it's not open source. That's kind of a significant problem, I think.

Steve: Correct.

Leo: Nor is WhatsApp, for that matter.

Steve: Correct. And but like Signal, for example, is.

Leo: Is open source.

Steve: It is, except that it requires you to, like, give it access to your contacts.

Leo: Right.

Steve: And if you don't, it doesn't work. And it's like, wait a minute. What if I don't want to? So anyway, so some fun conversation about that, and then we'll talk about - we have a little bit of errata. Oh, and news of a very well-regarded miniseries starting tonight on, I think it's AMC, I've got it in my notes, that when I tweeted the news a few days ago, it first was released in the U.K., and everyone over there said, oh, my god, we've seen it, it's unbelievably good. And I've never seen a 100% Rotten Tomatoes, or I think it's like 8.6 or something on IMDB. I mean, the highest scores I've seen. So lots of stuff for today's podcast.

Leo: And, by the way, I apologize for grossly underestimating your audience. Patrick Delahanty, who runs the code that does the audience measurement - and by the way, very sophisticated stuff we're doing, I'm really proud to say - 168,000 downloads of last week's episode so far. So far. Which means you'll probably cross 200,000 by the time we're done. And that's uniques. We measure uniques.

Steve: That has been some growth over the years.

Leo: Well, and partly it's because the measurement systems we were using had some serious flaws which we did not uncover till more recently. And so we've had to

roll our own. And now that I understand the code underlying and how it's working, I have some real confidence in these numbers.

Steve: Nice.

Leo: So there is a jump. But it's mostly, I think, because we switched providers.

Steve: Now counting correctly.

Leo: Well, what we're doing - at some point we should go through this because I...

Steve: I'm still bouncing everything through Podtrac. Should I be doing that?

Leo: That's fine.

Steve: Okay.

Leo: Well, no. In fact, I'll have Patrick email you. It still goes through Podtrac, but what we do is we have a redirect, we have a download link that's TWiT dot something. And then that does a double redirect. So it goes through our redirector and then Podtrac's redirector. Redirecting is, I know, it's not ideal. It won't be - that won't be for much longer. But redirecting is the only way to really effectively count. You would think you could look at server logs and stuff. But there are a lot of - there's a lot of noise in a server log. For instance, podcast apps often open five streams at the same time.

Steve: Yup.

Leo: And you don't want to count that as five people. It's just one person downloading five parts.

Steve: I do the same thing on all of my download counts because I only count the retrieval of the first byte of the file.

Leo: Right. And then but we have - but at the same time, everybody who downloads from Microsoft comes in as the same IP address. So you don't want to count a thousand people as one person, either. So there's some black magic, and we've refined it over time, but to the point now where I feel very confident. One of the tests is it's highly consistent. And shows like yours shouldn't show a lot of volatility. That was a warning sign, when there was all this volati- every week was, you know, 50 percent different. And that was just wrong.

Steve: Wow, you're right, that's crazy, yeah.

Leo: Because especially shows like yours, which doesn't have different hosts, doesn't have different content, and most people, I think, subscribe to, you wouldn't expect big fluctuation. So that was the red flag. And now that we've got our own system in place, it's very consistent, as we would expect.

Steve: Right.

Leo: A little behind-the-scenes stuff. 168,000, nice. That makes you bigger than The Screen Savers on Tech TV. That's what gets me. When you get the actual [sighing]. As one reviewer once said about Tech TV, they have the audience of a large high school.

Steve: So, okay. There's two different problems with cell stuff. One we talked about when we were addressing this whole subject years ago, and that's that there is another processor that we never talk about in every cell phone, which for historical reasons is called the "baseband processor." Apple has been making noises about doing their own for some time, but at this point they're still purchasing them from Qualcomm at a cost of about $33 apiece, which makes it an expensive component of all Apple smartphones. And thus they look at that and think, you know, we don't really need to be buying this from a third party. Yet there is a lot of technology there. It is an ARM-based system with firmware, and it's rumored that there's bugs there.

And the problem is that no one looks there. We're all, you know, when we were talking about iOS security whitepapers and breaking into the phones and the FBI decrypting this and fingerprints and, you know, all of that is completely separate from this baseband processor that's just sitting there quietly, not getting anyone's attention. And probably the people who know where the flaws are, are quite happy with that state of affairs. They don't want us looking there.

And it's difficult to look there because there isn't the same kind of - we don't have a relationship with it, the way we have with the A7X and the A8 and the Secure Enclave and all, you know, it's not very exciting. It's just sort of, you know, it is the thing that hooks us to the rest of the world when we have a cellular connection as opposed to a WiFi connection. It's not sexy. But it's crucially important that it be secure. And it's probably not. But we don't know because we don't look at it very often. And if history teaches us anything, it's what you don't look at is a problem because it's probably got problems that are unobserved, and that there are people who know what they are.

Now, completely separate from that is a known problem, a known huge problem. And that is the protocol which glues all of these baseband cell phones together. And that's known as "Signaling System 7" (SS7). And it is old, which is one of the first problems. Any time, you know, it's like SSL v1.0. Actually, it's the same era. It's from the 1980. And so back a long time ago, when the mobile phone system was just kind of beginning to happen, this system was put together, and the presumption was that there would be hard wires that were linking, you know, like leased lines that were linking individual cell phone carriers to each other in a private electrical network. Not packet switching, but leased line. And the idea being that the presumption was, well, these are our wires. Nobody will have access to them.

So the protocols that they developed had no concept of security. That just wasn't necessary. If you have a leased line connecting Sprint to AT&T, and they hook up to each end, you know, it's like, well - and they have a subscriber transit agreement, very much like top tier peering agreements we have on the Internet, then, you know, it's like, okay, we're going to connect to each other. And so we need some protocol for, like, handing a subscriber off. Like what if two cell phone, two mobile phone customers want to, on different networks, want to talk to each other? Well, we need some way to handle that. And we need to have billing information. And, you know, what if they want to do a conference call? We need, we have to have a conferencing protocol. And what if the subscriber has two phones, and we need to deal with SIMMs and all this?

So all of this kind of got developed quietly by the carriers to fulfill their own needs. And there are two key mobile protocols called MAP and CAMEL, M-A-P and C-A-M-E-L. And because of the timeframe when this happened, they have no authentication. There is no concept...

Leo: Ugh.

Steve: ...of authentication.

Leo: If you put a layer on top of it? Authenticate first?

Steve: Well, you could. But then everyone would have to support it, or it wouldn't work. And so today, this moment, the entire international mobile phone system is essentially open. And what "60 Minutes" demonstrated was something that was demonstrated in the summer of 2014 in the Chaos Computer Club. And in the show notes, for anyone who wants to dig deeper, I mean, our links to videos that are two years old, and what the "60 Minutes" show demonstrated, which was riveting, was - yeah, there's one of them.

And there's also, somewhere, maybe it's toward the bottom of that page is a 140-slide deck of, well, 140-slide deck of slides - there it is, that red one - where it went along with the presentation, showing, like, the nature of the complexity of the system. All of this has been now reverse-engineered, that is, the messaging protocols and how they work. This 140 slides shows the evolution into a network of protocols linking sort of the sprawling pieces of the system and all the points of attack which hackers now have access to.

And what "60 Minutes" demonstrated was all you have to have is anyone's cellular phone number. That's all you have to have. To set this up for the demonstration on the show last Sunday, they got a brand new iPhone, and it might have been purchased, and gave it to a - I don't remember now if he was a senator or someone in the House of Representatives. But they said, "With your permission, we want to provide you with this phone, and we'll give you a call. And you need to know that this is part of a piece we're doing on mobile phone hacking, and we'll be working with some German hackers." And these are the guys that worked this out two years ago and showed everybody. And we talked about it back then, but nothing happened. "And so you'll be subject to, maybe, if these guys could pull it off, eavesdropping."

And so he agrees. And so the "60 Minutes" person calls, and they have a conversation. And that's all it took. With knowing his number, it is possible to essentially install what amounts to a man in the middle, such that his physical location is known, his SMS messages are captured, his voice conversations are in the clear, and basically everything

he does with the phone at the cellular level, now, not smartphone stuff, but conversations and text messaging, all available.

And so later in the show they played back nonsensitive conversations that he had had with other people that were completely intercepted by this. I mean, and it's a very compelling piece. And I'm not going to spend any more time on it because there's really nothing more to be said except just sort of as a heads-up that the mobile phone system doesn't begin to be secure. And there's no question that law enforcement knows this and uses this with, I mean, like without a moment's second thought. There are international commercial entities known to be selling SS7 hacking tools to governments and law enforcements throughout the world.

So if anyone is having a conversation that isn't encrypted, for example, by Signal or WhatsApp or some additional encryption layer, and if you're not using iMessage or WhatsApp or Signal or Threema or some additional, something other than just SMS text messages, then everything you're doing, the conversations you're having, and the simple messaging system, you know, SMS messages, completely available to anyone who has access to the system and knows the phone number you're using.

And so basically, you know, this is why the concept, the notion of a burner phone has been in movies and plots for a long time and apparently is what people wanting to maintain anonymity do is they get burner phones, and they use them for a short time, and they toss them because the system that they're using is known not to be secure in any way, shape, or form.

Leo: So the only way to secure it is if no one knows your phone number.

Steve: Correct.

Leo: So a burner phone works only because the phone number's known only to your confederate. But presumably at some point government might figure that out, and that's that.

Steve: Correct.

Leo: Although Stingrays would figure it out.

Steve: Yeah. Anything that's able to, like, create a fake tower, then it's going to know who you are and where you are and be able to decrypt your conversations, too. So basically it's just an insecure network. Now, exactly as you were saying at the beginning, Leo, all of the things we're doing, layering our own end-to-end crypto on top, that's secure. But most people aren't doing that. I'm not having, when I have a conversation, I'm not bothering to, like, move the conversation over to a secure channel.

Leo: But as an example, if you used WhatsApp, which has phone calling, and many messaging apps do, that's encrypted over on top of the broken SS7. So that would be secure; right?

Steve: Well, except the metadata would not.

Leo: They'd know who you're calling.

Steve: Exactly. And there is no way to protect that. I mean, as has been said, metadata, you know...

Leo: But, well, wait a minute. What if the signaling, though, is not done through the cell network, but is done through the data network using an encrypted channel?

Steve: Well, yes. And in fact that's one of the things you can do is turn off, go into...

Leo: Don't use your cell phone calling capability. Use data.

Steve: Correct. Shut it down. Shut down the, yes, the cellular radio and use WiFi, like whenever you're...

Leo: Or LTE data. Couldn't you? Because it's not data. It's not phone number data. So you can still...

Steve: A good question. I don't know.

Leo: I mean, they could - yeah, you'd have to use Tor or something like that because they'd know the endpoints of the LTE conversation.

Steve: Yeah. My, you know, I guess my takeaway is we all want to have privacy, yet no one seems very willing to give up much for it. And all of the tests that we've read have shown, like, how little, like, here's a candy bar. Can I have your password? Ooh, chocolate. You know? And it's like people don't value that so much. I think there's a subset of people who really do care. And I respect that. I think those people should have the tools at their disposal for real privacy.

Leo: But we're not talking government intrusion alone. For instance, somebody could use this to get your second-factor authentication from your bank; right? So if they knew your phone number, and, I mean, they could get your - they'd have to know your bank password first. But if they got your password, then all they'd need is your phone number, and they can get that authentication code, for instance.

Steve: Yes. I forgot to mention when I was, yes, when I was talking about Hover last week, one of the things I liked about them is that they do offer two-factor authentication. And for access to my registrar, that's somewhere I'm not messing around with security. My domains are crucial to me. They offer two forms. And I did want to reiterate, and I forgot to last week, I chose the Google-style time-based authentication specifically

because there is no question it is superior, vastly, dramatically superior security. I do not want a six-digit SMS text message coming to me when I log into my Hover account. I much prefer recording a - I got it from them, the big key that drives the crypto in the time-based authenticator, and then have that in my phone so that it's generating a sequence of six-digit keys with a 30-second life. And when I want to log in, I provide that without prompting, except for the field that I have to fill in, to my Hover account.

So I just did want to note that, you know, exactly as you said, Leo, receiving text messages is - this is another perfect example of why that's not really good. I mean, it's better than nothing, but it's not second-factor authentication that's nearly as secure as establishing a time-changing, you know, a time-based additional factor where, after it's been established, no communication of it needs to pass back and forth.

Leo: Yeah, well, this is a good reason.

Steve: Yeah.

Leo: Yeah. So if you have a choice between an authenticator app - they always say "Google-based authentication." It's really not Google.

Steve: It's not.

Leo: It's a standard. If you have a choice between authenticator-based authentication, use that, software-based authentication versus text-based.

Steve: Right.

Leo: Text is convenient, and you may not have the choice. My bank doesn't give me the choice. But they should.

Steve: Right. And I did appreciate that Hover did.

Leo: Hover does, yeah.

Steve: Yup. So, okay. I have to set this up so that the statement from BlackBerry's CEO - so that we'll have the context for it because it's just the most mealy-mouthed non-confirmation imaginable. So this begins with Vice, Vice.com, who uncovered from court documents, and not just a few, thousands of pages of court documents that this was buried in. And so from Vice's coverage, they wrote:

"The revelations are contained in a stack of court documents that were made public after members of a Montreal crime syndicate pleaded guilty to their role in a 2011 gangland murder. The documents shed light on the extent to which the smartphone manufacturer" - and they're referring to BlackBerry because the headline was "Canadian Police Obtained BlackBerry's Global Decryption Key." And we learn some horrifying things by implication

also about the much-vaunted BlackBerry encryption, which doesn't impress me at all because the fact that this was possible tells us that, I mean, that this BlackBerry encryption was crap. Like for all these years we thought, oh, it's the best encryption there is. And it was completely secret. And now we find out, well, it was a secret that should have been better kept because it's awful.

So anyway, Vice says: "The documents shed light on the extent to which the smartphone manufacturer, as well as the telecommunications giant Rogers, cooperated with investigators. According to technical reports by the Royal Canadian Mounted Police that were filed in court, law enforcement intercepted and decrypted roughly one million PINtoPIN BlackBerry messages" - and we'll explain what PIN-to-PIN means in a second - "messages in connection with the probe. The report doesn't disclose exactly where the key - effectively a piece of code that could break the encryption on virtually any BlackBerry message sent from one device to another came from. But as one police officer put it, it was a key that could unlock millions of doors.

"Government lawyers spent almost two years fighting in a Montreal courtroom to keep this information out of the public record. And while neither the RCMP nor BlackBerry confirmed that the cell phone manufacturer handed over the global decryption key, and both fought against a judge's order to release more information about their working relationship, the Crown prosecutors admitted that the federal police service had access to the key. And if the global key is still sitting on a server in the RCMP's headquarters, the potential consequences could be significant. Although it wouldn't offer police a backdoor into most of its government and business clients" - I mean, that's a key.

Leo: And they say that government is the most important because, by the way, every member of Congress in the United States uses BlackBerry.

Steve: Yeah.

Leo: And you can bet the RCMP has given those keys to U.S. law enforcement. Right? And you don't want Congresscritters to start saying, what do you mean, somebody's listening in on my phone?

Steve: Uh-huh. They say: "Although it wouldn't offer police a backdoor into most of its government and business clients, who make up BlackBerry's core constituency, it would mean that police enjoyed years of access" - and we're talking from 2010, so at this point six years - "years of access to Canadians' personal cell phones without the public being any the wiser. In a technical report attempting to underscore the significance of this technology and filed with the Superior Court of Quebec, the RCMP stated that it had obtained 'the key that would unlock the doors of all the houses of the people who use the provider's services, and that without their knowledge."

Okay. So then this was also done along with Motherboard, who said: "BlackBerry, formerly RIM, encrypts all messages sent between consumer phones, known as PINtoPIN or BBM messages, using a single global encryption key" - which is where, you know, we just put our face in our hands because it's like, what? What? - "that's loaded onto every handset during manufacturing."

Leo: And it's a lot easier to just hand that out.

Steve: Oh, boy. "With this one key, any and all messages sent between consumer BlackBerry phones can be decrypted and read. In contrast, Business Enterprise Servers" - that's the BES that we've talked about years past, the BES servers - "Business Enterprise Servers allow corporations to use their own encryption key, which not even BlackBerry can access. According to more than 3,000 pages of court documents pertaining to the case that resulted from [what's known as] Project Clemenza, obtained by VICE Canada, the RCMP maintains a server in Ottawa that 'simulates a mobile device that receives a message intended for the rightful recipient.'" In other words, a man in the middle. "In an affidavit, RCMP Sergeant Patrick Boismenu states that the server 'performs the decryption of the message using the appropriate decryption key.' The RCMP calls this the 'BlackBerry interception and processing system.'"

So, naturally, news of this required a response from BlackBerry. John Chen, who is the wunderkind, apparently, Chairman and CEO...

Leo: No, he's not, he's the guy they brought in because the company was falling apart.

Steve: Right.

Leo: Yeah. So I don't know if he's a wunderkind or just...

Steve: Well, but in the past I guess he was, you know, he was, like, responsible for major, you know, dazzling successes in previous things he did.

Leo: Oh, I see, yes.

Steve: Doesn't look...

Leo: Not there, of course, yeah.

Steve: Right. Doesn't look like that's happening here. So he did a blog posting titled "Lawful Access" - which is the first clue - "Corporate Citizenship" - okay, second clue - "and Doing What's Right" - third clue. And he says - I just have one paragraph from a relatively short blog posting where he said essentially nothing. But so, given some serious reading between the lines, which is what's necessary, but he also slaps Apple, he says: "When it comes to doing the right thing in difficult situations, BlackBerry's guiding principle has been to do what is right for the citizenry, within legal and ethical boundaries. We have long been clear in our stance that tech companies as good corporate citizens should comply with reasonable, lawful, access requests." Okay, now, okay. Continuing, just to finish this paragraph: "I have stated before that we are indeed in a dark place when companies put their reputations above the greater good."

Okay. So that tells you all you need to know. So he's saying that, yes, we did not take the position Apple took of saying no. We complied with, as part of our corporate citizenship, with lawful access requests. Now, I would argue what they did was massively overly broad. And there are some diagrams that I've seen that show the details of the man in the middle. And basically they did hand over the master key that then allowed for unrestrained monitoring of the BlackBerry network, completely, I mean, of basically any conversation, anyone's phone that they chose to.

So I think that's pretty much the end of BlackBerry, at least the end of any notion that we were holding that they have some magic secret security that had been unobserved, but we just assumed was fabulous. If all phones have a single key such that knowing it allows you to decrypt everything the system is doing, then that's awful. I mean, it's like it's, essentially it's what everyone was saying that installing a backdoor would require. And we know that you can do secure decryption on demand in an intelligent fashion. This is not that. This is the worst possible design. So, wow.

Leo: Yeah. I mean, I think BlackBerry always assumed, well, if you really cared about security, you'd run your own BlackBerry server, your own BES server. And in fact that would have - that would be your own generated key.

Steve: Right.

Leo: And certainly that's what government and business probably does. It's just that it's kind of you and me, who would just buy a BlackBerry phone. And they always sold it as secure, but I never really believed them. So now we know.

Steve: Yeah. Wow.

Leo: I mean, they gave the keys, they were - many countries demanded the key. I used to say "keys." Now I say "key."

Steve: Yeah. They said no to somebody.

Leo: I think India. Or maybe China.

Steve: Yeah, I can't remember who now. Oh, Pakistan wanted it.

Leo: Pakistan, okay.

Steve: And they said no.

Leo: I think they didn't want - Pakistan wanted source code to the server.

Steve: Oh.

Leo: They didn't hand that over. But now we know a single key unlocks all phones that run through BlackBerry's own servers.

Steve: Wow. Yeah, and everybody knows.

Leo: Yeah. To tell the Canadians, the Canadians tell the states, the Americans tell the English, it doesn't take long.

Steve: Yeah.

Leo: It's probably on a bulletin board somewhere.

Steve: So, QuickTime is officially no longer, Apple has stated, going to be updated on the Windows platform. And the problem is Trend Micro found a couple problems with QuickTime, the latest version from Apple, the last version from Apple that will ever be released, and said, hey, Apple, we've got a couple - we've got some serious heap overflow problems that we found. And Apple said, yeah, well, we're not going to change them. We're not going to fix it. We're not going to do anything.

So that released Trend from what they considered their responsible disclosure obligation, where they would have otherwise waited and let Apple have the time to fix the problem. Apple's never going to fix the problem. So Trend immediately went public and said: "We're putting the word out that everyone should follow Apple's guidance and uninstall QuickTime for Windows as soon as possible. This is for two reasons. First," they wrote, "Apple is deprecating QuickTime for Microsoft Windows. They will no longer be issuing security updates for the product on the Windows Platform and recommend users uninstall it." And they say: "Note this does not apply to QuickTime on Mac.

"Second, our Zero Day Initiative" - this is Trend Micro's - "has just released two advisories detailing two new critical vulnerabilities affecting QuickTime for Windows. These advisories are being released in accordance with the Zero Day Initiative's Disclosure Policy for when a vendor does not issue a security patch for a disclosed vulnerability. And because Apple is no longer providing security updates for QuickTime on Windows, these vulnerabilities are never going to be patched."

I did a little more digging and found that they said: "Both of these are heap corruption remote code execution vulnerabilities. In one case an attacker is able to write data outside of an allocated heap buffer." Yikes. "The other vulnerability occurs in what's called the 'STCO' atom where, by providing an invalid index, an attacker can write data outside of an allocated heap buffer. Both vulnerabilities would require a user to visit a malicious web page or open a malicious file to exploit them. And both vulnerabilities would execute code in the security context of the QuickTime player, which in most cases would be that of the logged-on user."

So this is not just the QuickTime plugin, but this is any document type that QuickTime has registered itself as the handler for in Windows. The most probable attack vector, I would think, would probably be spearphishing, or maybe just more broad spray phishing,

where it's the standard come-on of here's the invoice for the something you didn't purchase email, and you go, what? I didn't purchase that. And you click on it, and it's an obscured URL that actually goes to some QuickTime document which contains code that then executes on your machine. So it's a means for allowing a malicious document to run on your machine.

Now, again, security context of the logged-on user, this is less a problem on modern Windows than it used to be because you have to elevate your privilege going through the User Account Control, typically to do anything important. On the other hand, a lot of users just click through that. So for naive users, this isn't going to protect them.

So, boy, removing QuickTime for Windows would be a good thing to do. In fact, it would be nice if, I don't know if Microsoft would ever do this, but because Windows Update is really the only way to push anything out, as we well know, to everyone on Windows, it would be nice if they pushed out something that would alert you if you did have QuickTime installed, and at least give you a heads-up about getting it removed. To me this feels like a big problem; although, Leo, do you think QuickTime is - where would it be installed? Would it be typically installed in, like, newer editions of Windows?

Leo: No, it's not installed by default. It's only installed if you install, well, a lot of people have it from installing iTunes. So iTunes no longer requires QuickTime. But in the past it has. And so if you installed iTunes on Windows, which of course everybody who used an iPod had to do, you would have gotten QuickTime with that install. No, in fact, we - so many people had problems with iTunes that we often recommended you install QuickTime separately. So you'd install QuickTime and then install iTunes. And that fixed some problems people had in Windows. So those people may still have legacy - it's an old version of QuickTime. It's very old.

Steve: Did you need that for, like, codec support?

Leo: That's the other issue. I'm told that some - yeah, I think so. Or actually, no, that was the player in iTunes. I mean, iTunes did not work without QuickTime.

Steve: Oh, that's right, I remember seeing the little window come up.

Leo: It wouldn't play without QuickTime.

Steve: Nice sort of a brushed silver look and...

Leo: Yeah, it's old, though. I mean, it's not required by modern versions of iTunes. The other problem, though, and it's much more significant, is that some programs, I think Adobe Premiere, their video editor, is one of them.

Steve: I've seen stuff bring QuickTime along. It's like, what the heck is that doing?

Leo: Yeah. You need it for Pro, if you want to do ProRes, which is Apple's format. That's, by the way, the format this show is recorded in. And our editors all have QuickTime installed on their computers because they use Adobe Premiere to edit the show.

Steve: What a mess.

Leo: So that's - so it's a problem, yeah.

Steve: Yeah.

Leo: You know, Russell's in Las Vegas for NAB. But when he gets back, I'll talk to him, and I guess we'll have to figure out what to do.

Steve: Yeah.

Leo: So, don't know.

Steve: Anyway, anybody who's listening to the podcast, if you don't know you need it, I guess just go into Programs and Settings, or Programs and - I think that's what it's called, whatever it is...

Leo: Yeah, Programs and Settings. It changes all the time.

Steve: Yeah, it does, and see if you've got QuickTime. Probably says - you think it's under "A" for Apple QuickTime? Or maybe "Q" for QuickTime for Windows?

Leo: Beats me. And by the way, it's Programs and Features in more modern versions of Windows.

Steve: Ah, that's right, Programs and Features.

Leo: Install the programs...

Steve: And just, you know, remove it. Hopefully you've pulled Flash out, if you don't need it. And now it's time to pull QuickTime out. You know, we're moving slowly forward, removing these things which are problems. But it is a, you know, maybe Apple could do this. I wonder if QuickTime checks itself for updates. Does it have an auto-update feature?

Leo: Yeah. Well...

Steve: Because Apple could take responsibility and push out an update that removes it from the system, since they're not - although they don't want to pull it away from people who do need it.

Leo: Right.

Steve: It's a mess.

Leo: It's a mess.

Steve: Yeah. So CBS News had some exclusive reporting. They found someone who was willing to talk about what the FBI found, or rather didn't, on Farook's unlocked phone. A law enforcement source told CBS News that so far nothing of real significance has been found on the San Bernardino terrorist's iPhone, which was unlocked by the FBI last month without the help of Apple.

Pat Milton, who's the senior investigative reporter for CBS, reported that it was stressed to him by his source that the FBI continues to analyze the information on the cell phone seized in the investigation, which is just their way of saying, well, yeah, but we're not saying we're done yet, probably because they're not happy, by saying after all this noise they didn't find anything.

I did want to mention, as I said at the top of the show, that Threema put together a rebuttal page to all of the glowing conversation about Signal, the Signal protocol fully being integrated into WhatsApp. And I also wanted to take the time to make the point that it was Signal I was talking about last week, and that WhatsApp is a container for Signal.

But, for example, the fact that you have to give it your phone number, that is, that's the way it identifies you, that's sort of annoying. I mean, it's a lack, it's an immediate loss of anonymity; whereas, for example, Threema doesn't require that you give a phone number. And that, where possible, Threema is as anonymous as it can be because they allow payment, which is I think $3, or at least it is on iOS, but iOS of course you're not anonymous because, if you have to buy something through iOS, you're known to Apple. But on the Android platform, where it is possible to be anonymous, you're able to purchase with bitcoin. And, you know, and they grumble, Threema grumbles, they have a clear economic model where you pay a little bit for support and the product, and then you know why you're getting the service from us.

But basically, I mean, anyone who's done any competitive marketing has seen these charts where it's all good news for the person preparing the chart and bad news for the others. Some of their points are good. One person raised a very good point via Twitter in the intervening week, and that is that WhatsApp, i.e., Signal in WhatsApp, doesn't give any residual indication when you've confirmed that you and the person you're communicating with have identical keys. That is, you did this on the show and showed it on the video, Leo, where the two opposite ends of a Signal-based Threema dialogue, when your phone saw the other phone's QR code, it gave a big happy green check mark

that then just sort of faded away. And so it was an easy way of doing the verification, but it didn't mark it somehow.

And of course Threema is all about authentication, and always has been, which of course is my own, you know, I keep reminding people that you have to have authentication, or you really have no protection from a man in the middle. And so Threema has those three, sort of, I think of them as the traffic signal lights - yellow, green, and red - for what level of authentication has been provided this connection. And every contact you have is branded with how sure you are, how much have you authenticated the other person's identity.

So anyway, I did want to - I wanted to separate explicitly our conversation of WhatsApp, that is an entity owned by Facebook, from their incorporation of the Signal protocol. And I was a little disappointed even to download Signal, which I took the time to do, and saw that it, too, needed access to my contacts list. And actually, Leo, you and Lisa showed up among my contacts list...

Leo: [Crosstalk], yeah.

Steve: ...as the only people I could connect to.

Leo: That's a convenience because otherwise how do you find people?

Steve: Yes.

Leo: So, but of course it leaks information. And so now we know, if somebody knows your cell phone number, you're screwed. So that's probably not good information to leak.

Steve: Yeah. So, and again, everyone wants, everyone says they want security. But we all tend to take convenience over security. And so, and I think if anything that this podcast is teaching us all, it's that security in the ivory tower, you know, in the land of the cryptographers, that's the only place it is pure, the only place it reaches, like, its theoretical perfection. Because the moment the rubber hits the road, the moment it becomes time to deploy it, then the real world gets in the way. And so it is, the idea of absolute security is an illusion because there are so many other things that can go wrong in the actual implementation of it. So I just think it's important to have that perspective because it's the truth.

Leo: And I think Apple admits, in fact I seem to remember, and I wish - I can't find it, I think it was Tim Cook saying recently, well, we carefully weigh security with the convenience of our customers.

Steve: Right.

Leo: And that's their kind of admission.

Steve: Well, iMessage is the perfect example. iMessage manages the keys for the people you're encrypting to. Which means, you know, they could be any...

Leo: For your convenience, Steve.

Steve: They could be any - for your convenience.

Leo: For your convenience.

Steve: That's right. Okay. Two little bits of errata. Oh, no, one bit of errata. Never10 did get bumped up. A little tiny - I gave it a one-thousandth of a bump in the version number. 1.3 I talked about last week, where I added the ability to delete all of the files that might have been downloaded ahead of time and hidden, and also a series of command line switches that could be used by enterprise.

Well, friend of the show Evan Katz found a bug in 1.3 because - and it's absolutely a bug. I wrote a new parser for it and failed to exclude the text before the command line arguments. I don't know what I was thinking. I just - it just was an oversight. And Evan had a network drive in the path, Programs and Updates\, and he put Never10 in there. And when he tried to run it from that network share on a different machine, it appeared not to run. Well, it was running, but it saw the phrase "update" in Programs and Updates and treated that as a command-line switch. Whoops.

Anyway, 1.3.1 fixes that. And so it wasn't worth giving it a .4, just a bug, plain and simple. I overlooked it; and so thank you, Evan, for bringing it to my attention. And that's fixed now. And I did get reports from a bunch of people saying that they'd recovered, I saw one just this morning, 5GB of hard drive space by running v1.3 just once, so...

Leo: It's a 3GB download. But as you pointed out last week, it expands it after it downloads it, which can make it much, much larger. Jeez.

Steve: Yes.

Leo: Couldn't you wait to expand it till you needed it?

Steve: Yeah, wouldn't you think?

Leo: God.

Steve: So, unfortunately, I did tweet this for people who follow me, but it's on AMC, and

it's titled "The Night Manager." I think that's a title of the...

Leo: Famous movie.

Steve: Yes.

Leo: Yeah.

Steve: Or a book. It's a miniseries, and it's starring Hugh Laurie, who...

Leo: Oh, love Hugh Laurie, of course, "House."

Steve: Yes. He was the, yes, he of course was Dr. House in the series "House." And I'll never forget, like, seeing some of the behind-the-scenes stuff and listening to his accent for the first time, it's like, wow, he doesn't have an accent when he's Dr. House.

Leo: I know. He's really good. But he was - Fry and Laurie, Stephen Fry and Hugh Laurie, when they were at Cambridge, had a wonderful comedy duo. You've got to look up "Fry and Laurie" on YouTube. Hysterical.

Steve: Oh, good.

Leo: Yeah. They're Brits. But this is a great John le Carre novel.

Steve: Yes, that's what it is.

Leo: I love John le Carre. And there's nothing - he's just wonderful.

Steve: So I picked up on it because one of the things I do, I've mentioned, I just - I have TiVo sucking in Charlie Rose. And Charlie is, unfortunately, a lot of it is about politics lately, and the U.S. election has just gotten to be so, so repetitious and annoying. So, but he interviewed, Charlie Rose interviewed Hugh Laurie, and they discussed this. And so this is not like some hotel night manager. Hugh Laurie plays an international arms dealer. And so IMDB rated it at 8.6 of 10; Rotten Tomatoes gave it 100%. I tweeted it. A whole bunch of our listeners in the U.K. were just jumping up and down, raving. One person said that the first episode put him to sleep. It's like, well, okay, maybe he took the Healthy Sleep Formula too early in the evening.

Leo: Well, Carre is very - it's not James Bond. Carre is, I mean, he wrote "The Spy Who Came in from the Cold." You might have seen that movie. You might have seen that movie.

Steve: Ah, right.

Leo: He is a - I think "Tinker, Tailor, Soldier, Spy" was a wonderful miniseries based on his books. His character, Smiley, is a very - it's not swashbuckling. But it's, I feel, very accurate, very realistic.

Steve: Well, apparently this picks up after the first episode. So people should not be disappointed. Apparently the first one is a little bit of a stage-setting episode. But then it kicks in. And what Hugh said was that what was different for him was that he was dying to produce this; whereas with "House" he would show up, and they would give him a script, and he would just do it. This is really, I mean, this has heart and soul. He really was excited to do this.

So it starts tonight. So I'm afraid that people who are not hearing the podcast until tomorrow will be behind. But I'm sure you'll be able to catch up, or they'll run them all together. They called it a "miniseries," and I meant to look to see if it's, like, in successive nights, or is it just going to be over the, you know, like once a week. I don't know what the schedule is. But so "The Night Manager" on AMC in the U.S., and everybody in the U.K. already knows all this.

Leo: I'm logging into my TiVo right now to record it. By the way, that is a nice feature.

Steve: I love it. I use it all the time.

Leo: Love my TiVo. All right. "Night Manager."

Steve: And a little quick Zeo update. For all of the 2,500-plus listeners who got Zeos, there is an effort underway to create replacement headbands. That's the one thing about it that has a potential to wear out because it uses a conductive, three conductive silver bumps. Anyway, ZeoBand.com, Z-E-O-B-A-N-D dotcom. He's still sort of trying to gauge interest. And it's funny, too, because he sent out a mailing to those who registered a while ago, saying this really started out kind of slowly, but interest suddenly ramped up for some reason.

Leo: Yeah, we know why.

Steve: I don't know what happened. It's like, yeah, and the world is also all out of Seriphos for some strange reason. So if you're playing with your Zeo, you might just go over to ZeoBand.com and let them know that you're interested. Maybe, I think he was - something, I heard something about $15. So it seems reasonable for a replacement headband.

And somebody working in the GRC newsgroups has almost finished an amazing new Zeo app for Android that does all kinds of stuff. Much higher resolution. Instead of five-minute intervals you get 30-second intervals. You can do journaling. You can annotate them. You can export the database over to the Java-based Zeo Viewer. Just it's

unbelievable feature packed. It's not quite out of beta. We're all using it over in the GRC.health newsgroup. As soon as it hits 1.0 - I just don't want to deluge him with additional testers because he's not really ready yet. So I'm not pointing people to it, though I've already created a bit.ly shortcut for when it's time. So I just wanted to let people know there will be an additional open source GitHub third-party Zeo app happening soon.

And lastly, I got a nice note from - a "G'day Steve" from Canberra, Australia, Simon Byrne, under the subject of "SpinRite recovers SSDs?" And then he has in all caps, "YOU BET IT DOES." He says, "G'day, Steve. I bought SpinRite years ago for no other reason than to support you and the Security Now! podcasts." Thank you. "Last Saturday I fired up my MacBook Pro with a 1TB Outer World Computing SSD installed. It got halfway and then shut down. Not good. I do video production and generate very large amounts of data. I do have a good local backup regime, but backing up over the web is impractical as I generate many gigabytes every day. However, I was working the day before offsite with no local backup, so I was faced with losing a full day's work, which equated to about 12GB.

"I took the SSD drive out of my Mac and put it into one of my PCs and fired up SpinRite on Level 2. Nine hours later, SpinRite had finished, reporting no errors recovered. I was disappointed that no errors were shown, so I was dubious as to whether it was going to work. I tentatively put the SSD back into my Mac and turned it on and, YES," in all caps, he says, "it booted up perfectly. I immediately backed up all my data. So, yes, SpinRite absolutely can recover data on an SSD drive." Thanks, Simon, for the report, and confirmation of what we've been finding, and the reason for a long future life of SpinRite.

Speaking of email, okay. So we know about HTTP STS - Hypertext Transport Protocol, Strict Transport Security. The idea with that is that websites are able to essentially publish, to anyone who connects to them, they're able to publish the fact that they only want to be connected to securely. So in the header, in the response header to a query, so a client makes a query, you know, a user's browser makes a query to a website. And maybe they make it over HTTP, and then the website redirects them to HTTPS, the equivalent, or whatever. So, for example, you may be able to get to that site over either HTTP or HTTPS.

But in either event, in the response headers, the metadata that's not part of what you normally see, but things like the expiration date and cookies that may be assigned to that session, one of the things there is this STS, the strict transport security header, which gives a lifetime, in the future, the number of minutes that the site is willing to commit to always offering secure connections.

So the idea is the browser will cache that. It'll store that, if it supports strict transport security, and they all do now, it'll store that with a date stamp. It'll, like, look at today, at now, the timestamp of now, and it'll add that many minutes that has been indicated to now and set a time, an expiration date in the future for when this expires, if it isn't renewed anytime again. And of course normally every visit will bump that timestamp forward that far again into the future.

So what this does is this gives the web browser itself permission to upgrade any, any link to that domain from HTTP and connecting to port 80, to HTTPS and connecting to port 443, which is the well-known port for a secure HTTP connection. And this all works.

Now, this still has what's known as the "first contact problem." And in fact there's an acronym, TOFU, that we've talked about in the past, T-O-F-U, which is Trust On First Use. Because what that sort of reminds us is, well, we only get the knowledge of strict

transport security after first contacting a website. We don't have it a priori, ahead of time, in any means.

Now, there has been action taken there, too. We'll remember that Google was leading this, and they offered any domains that wished it to register with them for inclusion in Chrome as a domain that will always be secure. GRC is proud to be enumerated among those, like early on, and is still there. And so that solves the first contact problem. If you basically build into the browser the permission for this domain, then you solve that trust on first use problem, the idea being that, if the browser had never contacted an STS site, and there was a man in the middle downgrading HTTPS to HTTP and, for example, also because of their man-in-the-middle position, stripping the response header, saying use strict transport security, then the browser would never have the opportunity to know the site offered it and so wouldn't be able to take advantage of it and could then run an exploit. But it just takes one trustworthy, one first contact to initialize the cache of that knowledge.

So we have all that. We don't have security, useful security, today for email, unless it's layered on top like PGP, where the underlying transport mechanism is just known to be useless for security. So if you really care about a message being secure, you'll do something on top of it. Even if you weren't using PGP, you would, you know, write the message in some DOC, and then zip it with a good password, and then attach it to an unsecure piece of email to send it to somebody who knew the matching, you know, who knew the password, who would then receive it, unzip it, and view the DOC.

So but the problem is that email has been really resistant, probably because it's a multicomponent system where with the web browser we just have a server and a client, like a one-to-one relationship. The email ecosystem has MTAs, as they call them, Message Transport Agents; and MTUs are the user side. They initiate email and absorb mail.

And the other problem is it's not just a one-to-one. I might be - I have my user agent, and I place email on my SMTP server, my MTA. And then it has the freedom to connect to another one and another one and another one, and then finally a destination one. There have been, historically, crazy complex email systems that do use a multi-hop store-and-forward because in fact the architecture, the original architecture allows that.

And in fact one of the things that we've talked about in way years past was looking at your email headers. And you can see, like, you view from the first header backwards the path that the mail took from one server to the next because each of the hops, the server adds a few lines of information. That's done in order to prevent the possibility of loops, where due to the fact that you've got essentially email routing, it's possible to have a routing loop which would cause email to go around in a circle forever, if you didn't have some way of a server receiving a piece of email and checking the headers and saying, wait a minute, I already sent this. Because it would see itself in its own headers that it had put on when it sent the mail out, and that would tell it, okay, we've got a problem. This is not getting to its destination.

So as a consequence of the fact that it is a more complex system, and that it's got deep history, it just hasn't been easy to upgrade this. Now, we've talked about one of the solutions, which is known as "opportunistic TLS." And that is, you sort of - the server that is accepting a connection can give a list of capabilities and does give a list of capabilities in response to the hello message, which is the opening message in a dialogue that two servers, two SMTP servers have between each other as they're initiating their handshake.

In the hello message, the connection-accepting server lists its capabilities. And among

them would be the statement STARTTLS, which is saying that it knows how to accept the STARTTLS command from the connection initiating server, which would allow them to bring up a private TLS link, a point-to-point link, which would then mean that the email that they would transit would not be in plaintext. Which it otherwise is. And, I mean, and this is a problem because, by default, email, even in this day and age, is unencrypted between endpoints.

Now, the problem with STARTTLS is that it is privacy protection only against a passive observer. That is, only somebody who can only monitor the network is going to be stumped by seeing STARTTLS happen. They're going to say, oh, darn, look, they both support STARTTLS, and I just saw the command go by. And now there's a TLS negotiation. And now it just looks like random, pseudorandom noise, and I can't see what the messaging is.

The problem is a man in the middle, somebody active, an active attacker, simply drops the STARTTLS declaration from the list of supported features that the answering SMTP server offers. And then the accepting server won't attempt to bring up a TLS connection because it believes that the one it's connecting to doesn't know how to handle it. So it's trivially overridden. It's trivially susceptible to a man-in-the-middle attack.

So what a group of people decided to do, and this is Google, Yahoo!, Comcast, Microsoft, and 1&1 Mail & Media Development & Technology that's a German email provider, that group of five people - Google, Yahoo!, Comcast, Microsoft, and 1&1 - they've got a proposal. And I was all excited, thinking, hey, okay, cool, that sounds like a good idea. Until I read it. Oh, goodness.

So they're trying to do for SMTP what we have done for HTTP. Unfortunately, due to the details of the way SMTP works, it's not as simple as the server sending a header to say that I support security because it's trivially easy to intercept that, and we sort of already have that with STARTTLS, the server saying, yeah, I support STARTTLS. And you simply strip that out, and now you're not going to have any security.

So essentially they have gone to great lengths to engineer a system that still has that problem, such that they're - and they fully acknowledge that they have the TOFU, the Trust On First Use problem. They have a system that allows the receiving server to cache the knowledge that the server it connected to supported security. So they are, once again, they're basically using the same model as HTTP STS is using, where you have to get the message through once, and then they're going to cache it.

But they use an out-of-band signaling, that is, they don't use the SMTP protocol in the same way that HTTP does use the existing HTTP protocol, which is one of the things that made this so simple. Instead, they go through all kinds of jumping through hoops. They want to create a specially named and well-known named subdomain in DNS, off of the DNS domain name of the email domain. So, and that's called _smtp_sts. Then it would be like dot and, for example, GRC.com [_smtp_sts.example.com]. So that subdomain would contain text records to describe the type of SMTP STS that this email server supports, and things like expiration time.

But the problem is they recognize that DNS is not secure. And so all throughout this document they're saying, well, really we should have DNSSEC. But we don't, so here's what we're going to do. And so, for example, among the things that they do is that the records in this insecurely delivered DNS record can point to a well-known directory in an HTTPS server where it's possible to find the identical records that the DNS subdomain is delivering. I mean, this is just unbelievably horrific.

And then, and the reason they do this is they want to take advantage of the PKI, the existing Public Key Infrastructure, which does exist for the web, but doesn't exist for SMTP. So they use DNS to point to - and they call it Web PKI is the name in the field that they're using. And so that that contains an HTTPS URL, referring to the same domain and a directory which is .smtp_sts, then slash, and then smtp-sts, and then a document name. And that contains records which have to be exactly the same as what DNS is publishing.

I mean, the further I dug into this, the more I couldn't believe what, like, the hoops that they were jumping through in order to try to get something that still has the Trust On First Use problem, and where throughout the document they're saying, but this would really be better if we just had DNSSEC. And so I'll just remind everyone that DNSSEC, when it finally happens, and it's just, it's coming along very slowly, what it will allow is absolutely secure publishing of anything we want to publish.

So, for example, right now we have a public key hierarchy, where we trust Certificate Authorities, and they sign our web server certificates, and that's the way we trust them. But if we had DNSSEC, that is, if we absolutely could cryptographically rely on what a DNS response has, then a domain could publish its own certificate, and a client could look up the certificate in the domain because - and what we don't have today is the absolute ability to securely sign what the DNS provides. If we had that, then the certificate would be known to be true, and it could then use that to compare it to the certificate coming from a web server, and there would be no need for a third party for domain verification. Again, in the same way that we still have a use for extended validation verification, where we're asserting much more than just a certificate for a domain, we still need a third party. So there's still a use for third-party certificate authorities.

But the point is, and we've talked about DNSSEC through the years, it would be wonderful to have this because another thing it would do is it could provide a simple statement that this email server accepts secure connections. End of story. And so SMTP servers that are using DNS to look up the MX record, the mail exchange record, and sometimes the mail exchange server name or IP, could also pull some future defined record. And this crazy SMTP STS RFC even talks about, oh, yeah, you know, be nice to have a new RR, a Resource Record in DNS that could be securely delivered over DNSSEC, and it would just say, yeah, this domain supports security.

Well, let's just do that. Because the problem is this kludge from hell, if this ever got traction, it's just going to - it just mucks up everything to an incredible degree, in order to try to get security, which remains weak even if we did all of this. And it just strews its junk all over the place. So, end of rant.

Leo: All right.

Steve: I was thinking, this thing sounds wonderful, until I read it. And then it's like, oh, please don't let this happen. Please, please, please. Because, as I said, the subtitle of this podcast is "Horrible Internet Kludges Never Die." And this is one for the record books in terms of just being just a horrific kludge. Yikes. We just need DNSSEC. If this energy could have been put in furtherance of DNSSEC, it would have been far better invested because, if we get that, we get all, we get everything else. I mean, just I can't even imagine the things we're going to be able to do after we get good DNS security.

Leo: Right on. Someday. Well, Steve, what fun it has been. I think we're done; right?

Steve: Next week a Q&A. Yup, we're done.

Leo: Yeah, yeah, we have strewn the Internet with our words.

Steve: And if this can help to keep this from happening, then it was time well spent. Oh, lord.

Leo: Steve's website is...

Steve: Just unbelievable.

Leo: Oh, well, you know. They're trying.

Steve: Yeah.

Leo: Steve's website is GRC.com. That's the place where you can find SpinRite, the world's best hard drive recovery and maintenance utility, and also all of his other fine stuff, like, oh, I don't know, Shoot the Messenger, ShieldsUP!, DCOMbobulator.

Steve: Never10.

Leo: Perfect Paper Passwords. And Never10. Is there an updater in that, or do you have to just download a new version if you want?

Steve: No, just it's 83K. The updater would have expanded it up. I'm not even sure what I'm going to do with the SQRL client because it doesn't - you don't have to install it, but people want to install it.

Leo: Right. Don't install it. Just put it, you know, put it - run it.

Steve: Yeah. Except that browsers go, like, put things in the download directory.

Leo: Right.

Steve: And then so I think it should install itself. That is, and I'll talk to all [crosstalk].

Leo: You can do that. When you run it, you move yourself to the applications folder, and that's it, yeah.

Steve: Right, right, right.

Leo: A lot of apps, not a lot, but some apps, I think, do that. And I think it's the right way to do it.

Steve: [Crosstalk]

Leo: It's not an installer, it's a mover.

Steve: It's a mover.

Leo: It'll add one line. Am I in the Programs and Features or whatever it is, and I am in percent app docs percent, well, then, move me there.

Steve: Yeah.

Leo: Ah, yeah, what else? Oh, the show is there.

Steve: Unless I'm on a removable drive. It's aware of that already. And so it...

Leo: Portable.

Steve: Because you want to be able to carry it around...

Leo: Right, portable.

Steve: ...and be able to run it from one drive.

Leo: Oh, that's the way to do it, yeah.

Steve: Yeah.

Leo: Oh, let them run it from Download, who cares?

Steve: Except that some people purge their download folder from time to time.

Leo: Oh, but that's fine. You only run it once.

Steve: I haven't figured out what I'm going to do.

Leo: Well, I mean, SQRL you don't have to run it again, right?

Steve: I think I may have an announcement next week.

Leo: Oh, whoa.

Steve: Yeah, I did a - I finished a chunk of work yesterday, and I've got a great week ahead of me, so, yeah.

Leo: Good. Yay.

Steve: Getting close.

Leo: Check up on SQRL's progress at the website, GRC.com. You can also find the podcast itself, including transcriptions, at GRC.com. Steve's Healthy Sleep Formula is there. Somebody sent me an email, "I can't find Never10." It's on the website.

Steve: Or just google it.

Leo: It's in Projects; right? Or google "Never10" works really good, yeah.

Steve: Yeah, Google, you know, is - that is the Internet, you know. So says my realtor friend, Judy. I just put it in the Google.

Leo: She's not far wrong.

Steve: I just ask the Google.

Leo: How do you get to Yahoo!, Google? Oh, this way. Over here. If you want audio or video of the show, you can also get it from us, TWiT.tv/sn. We put it on YouTube. We put it on, you know, on the Internet as a podcast. That means you can use your podcast client to download it. All you have to do is get the podcast client and search for "Security Now!." You can find audio and video versions. Of course there's TWiT

apps, too, on every platform. That's another way to do it. We will be back here next Tuesday, 1:30 Pacific, 4:30 Eastern, 20:30 UTC for yet another edition. I guess a Q&A, barring...

Steve: Q&A time.

Leo: Barring problems in security. So get your questions to Steve at @SGgrc on the Twitter, or GRC.com/feedback.

Steve: Yup.

Leo: We'll pick 10 and do those next week. Thanks, Steve.

Steve: Okay, my friend. Thanks.