"Mega" Security Update

Description: After covering "UPnP a week later" and catching up with some interesting security industry happenings, Steve and Leo take a look into the controversy surrounding the security (or lack thereof) of Kim Dotcom's new "Mega" cloud storage offering.

High quality (64 kbps) mp3 audio file URL: http://media.GRC.com/sn/SN-390.mp3
Quarter size (16 kbps) mp3 audio file URL: http://media.GRC.com/sn/sn-390-lq.mp3

SHOW TEASE: It's time for Security Now!. Steve Gibson is here with the latest security news and something he's promised for a while: a look at the security behind the new Megaupload service. How safe, how secure is it? Despite all their protestations, maybe not as secure as you might have thought. Steve explains all next on Security Now!.

Leo Laporte: It's time for Security Now! with Steve Gibson, Episode 390, recorded February 6th, 2013: Mega Security.

Time for Security Now!, the show - as I wave my finger - the show that brings you and your loved ones peace of mind, thanks to this fellow right here, the Explainer in Chief, our security guru, Mr. Steve Gibson of GRC.com. Hi, Steve.

Steve Gibson: That would be me, Leo. Yes, yes, our non-live listeners have been spared 36 minutes of nonsense.

Leo: It's not nonsense.

Steve: No, in fact, I've actually...

Leo: It is not nonsense.

Steve: I've received some tweets during this saying, "This is why I tune in and listen live," you know, "LOL."

Leo: Always listen live. You never know. We talk about coffee, science fiction. Basically, I try to have the conversation - see, people may not know, but this is - you and I only really get to talk once a week, so this is our personal time.

Steve: Oh, I miss our quiet, private...

Leo: This is our personal time. So we like to get a little time to catch up, see how each is doing and fill him in on the gossip.

Steve: Boxers or briefs.

Leo: So hello. I guess you've been kind of busy, a little busy this week.

Steve: I've been super busy, yes. We have lots of news. Today, as promised, we've got an overview of the security - and lack of - of Kim Dotcom's new Mega...

Leo: Oh, good.

Steve: ...cloud storage offering.

Leo: Oh, great.

Steve: Some of the attacks have been a little overblown. But it also demonstrates that they've got a ways to go. And it's a perfect example of why this stuff should all be done in the open. The consequence of the fact that their initial client, their only client at this point, is a browser is that their tech is in JavaScript, which automatically means it's open for analysis. And, oh, boy, that was a good thing because they've got a lot of work to do. So we're going to talk about that and catch up on other news of the week.

There was a really unbelievable article in Slate about a new iOS app called Silent Circle that I want to talk about. But I first want to follow up on last week's Universal Plug & Play nightmare revelation. Our listeners will remember that on Tuesday, the day before last week's podcast, HD Moore of Rapid7 surprised the world with their report on the number of open and exposed Universal Plug & Play ports on the world's routers, 2.2 percent of the entire addressable IPv4 space. So that's 2.2 percent of 4.3 billion, which was 81 million, different devices responded. And as a consequence of that, I said that I would immediately do a - I would add a test for this to ShieldsUP!. So all of our listeners who don't already know, I did tweet this Thursday afternoon, late afternoon. It took me all of Thursday to write it.

And actually, when I went - I hadn't looked at this code for 10 years. But that was the rewrite of the original ShieldsUP! system where I added the concept of true stealth, where I was looking for any noise coming back from an IP that we were testing. And so I rewrote the foundation with the experience from the first one. And I was frankly quite pleased when I looked at it last Thursday. It's like, oh, this is pretty nice. And so there was already - I was already checking for NETBIOS ports, which is a UDP packet where I care about the return data. So I was pretty much able to copy the logic that I had developed and proven there for the Universal Plug & Pray probe.

Anyway, so just anyone and everyone actually should go to GRC.com, go to the regular ShieldsUP! entry point, and you'll see a big, clickable icon, says whatever it says, GRC's Instant UPnP Exposure Test or something like that. And it just takes a few seconds. You can get three different outcomes. You are either completely invisible, or you are actively denying probes, or you're accepting them. And in fact, because I had a couple members of the press had asked me, what would it look like if I was exposed, on that same page down below I have three links to samples of the three different outcomes, so people can see what they look like. I also, on that page, Leo, link to the podcast that we did last Wednesday, which was the whole beginning of it was about this. So I linked to the YouTube instance of it, for anybody who wants to see it, and also to the audio.

So what we have found is, to date, 780 different individual people that have come by have been exposed. Someone, actually several people reported the rejected response. I didn't even know if that was possible, but it was available to me, so I built in that possibility of being actively rejected, the idea being that, when you send a UDP packet at the port, you get back an ICMP packet that says "Destination unreachable." So it's an affirmative, there's nobody here. So you're not stealth, but at the same time you're also not vulnerable to a Universal Plug & Play exposure.

And then, of course, if you do respond, I show on the page the block of text which comes back, which has all the details about the version of Linux that you're running, the version of the Universal Plug & Play code that you're running, sure enough, the port that the device is listening on for future conversations. I don't go to the next step of then establishing an HTTP TCP connection to that port. I could have. But my feeling is, hey, you don't want any exposure anyway. So there's really no point in showing people more of what's there. The fact that anything is there is not good.

So the other interesting thing that I found is immediately upon finishing the podcast, I set up my own little honeypot net. I have a block of 64 IPs here that have never been used for almost anything. They get, like, just debris traffic in, just the noise that's on the Internet. But I left the machine on. I unbound the IPv4 TCP protocol so that it came protocol free because I had to put it directly on the Internet with no firewall, and then I hooked up the network. It's a cool thing.

I don't know if people know that, when you're using Wireshark, you can, in the same way that years ago in ShieldsUP! I showed people how to unbind protocols, you can unbind your IPv4 and IPv6 protocol stack from the interface so that you are completely invulnerable. You're an adapter listening promiscuously, but it doesn't matter that you're now on the Internet, as long as you're careful, because you have no protocol, essentially, listening to that wire. It's just a raw Ethernet cable. And Wireshark is still able to pick up packets.

And the point is that I then built a filter looking for anything incoming on UDP protocol and port 119. I have so far, so it's been about a week, found 21 unique IPs out on the Internet that have sent UDP port 119 - I'm sorry, I mean, 1900 is what I meant, mean to say, 1900, incoming packets to this block of 64 IPs. Several of the IPs have been scanning. They sort of scan in a pseudorandom order. They don't come back that often. Several others are doing a much slower scan. Oh, and I should mention that every day a couple new ones appear that I have not seen before.

So this is exactly what we would expect if hackers were beginning to come up to speed about this vulnerability and deciding that they're going to start having some fun with this. So this is not a theoretical concern. There are definitely unknown entities out on the 'Net. I've tried to look up some of these IPs and gotten nowhere. They're just, like, they're just - there's no records of them anywhere. So a couple, the first ones that I saw were in Amazon's scalable network architecture system. It's like, whoa, that's interesting. So somebody was apparently using Amazon as their scanning hub, basically rent some machine and some bandwidth from Amazon and scan the 'Net for Universal Plug & Play vulnerabilities.

Anyway, the test exists. And people have been surprised and turning them off. I did just see a tweet saying that a new version of the MiniUPnP, I think it was version 1.6 the tweet said, had just been released. That doesn't help end users because we're not able to update our firmware ourselves. But it does make the packages available to the manufacturers. And hopefully this has been a big enough wakeup call that we'll get some firmware updates and close these things. So that's what's been going on here.

Leo: Well, thanks for doing that. I know everybody was dying to test their routers. I'm amazed at the number of issues. Although in some ways that seems like a lower number than one would expect, given how many routers are vulnerable.

Steve: You mean like the number I'm seeing, or the incoming...

Leo: Yeah.

Steve: Well, these are only...

Leo: They're Security Now! listeners, for one thing.

Steve: Yes, exactly.

Leo: Not that there's anything we can do about it, really. Is there?

Steve: Well, no. I mean, all you can - okay.

Leo: Buy a new router.

Steve: Most routers, most routers have Universal Plug & Play on by default. If you're not a gamer, and, well, really I think that's about it. If you're not a gamer, you should turn it off.

Leo: Right.

Steve: Now, but even if it's on inside your network, it may not be on outside your network. And that's the only danger. So that's what I'm testing for is whether it's exposed to the public.

Leo: And it's not possible, if I've turned it off internally, that this flaw that we're talking about could still give an external attacker...

Steve: Yes, that has - we have...

Leo: That's what I'm saying, that's what I'm thinking is that, yeah, well, anybody listening to this show has certainly disabled UPnP years ago.

Steve: We have confirmed that there are some routers that leave it on outside, even if it's off inside, and some that don't actually turn it off inside. That's just a checkbox.

Leo: It's like the WP...

Steve: It's a checkbox. It's not hooked up to anything.

Leo: It's like the WPS flaw, yeah. Which Linksys has a checkbox that says "Turn off WPS" that does nothing. It's nice to put those in the UI because it gives people, you know, reassures people.

Steve: Oh, yes. And also it's right - it's above the one that says I'd like to have no problems. And then you can just check that one.

Leo: It reminds me of - it's the security theater that we go through as we go through airports. It's to make you feel better. It doesn't actually - you know, it was interesting, as we were leaving New Orleans, because there was...

Steve: That just airs out your socks, basically.

Leo: Yeah. As we're leaving New Orleans because there was such a flood of people the day after the Super Bowl...

Steve: Oh, I can't even imagine, Leo.

Leo: No, this was what surprised me. They turned off security. We did not take off our shoes. We did not take out our laptops. They took off the micro, the millimeter scans. All we had to do was go through a metal detector.

Steve: Nice. Just like the old days.

Leo: It was just like the old days. And I thought, but wait a minute, didn't - wasn't it important that we took our shoes off? Wasn't it important that we took our laptops out?

Steve: No. You see, the bomb would have gone - would have been...

Leo: It's irrelevant.

Steve: ...during the game, not after the game.

Leo: Oh, I see, yeah.

Steve: So clearly that didn't happen. Incoming you have security. Outgoing, eh, you know, maybe the fuse didn't work.

Leo: Security theater. It's everywhere, friends, it's everywhere.

Steve: Yes, it is. So I did get some breathless tweets about the so-called "Lucky Thirteen" attack on SSL. It made some news that there was a new compromise on SSL. And I thought, okay. What do we have now? Okay. So, short version is there's nothing to worry about, nothing to see here. Quoting from the paper, from people who I guess really needed a PhD, they said, "We have discovered a variety of attacks, each having different complexity and severity. For TLS, our attacks are multi-session attacks" - and they're not kidding when they say "multi-session," we'll cover that in a second - "which means that we require the target plaintext to be repeatedly sent in the same position in the plaintext stream in multiple TLS sessions."

Okay. So let's stop there. That means that they need essentially the same stuff to be sent, unmodified, over and over and over again. Okay. So, okay, well, maybe that could be bad.

"The attacks involve detecting small differences in the time at which TLS error messages are returned on the network in response to attacker-generated ciphertexts." Okay. So this is a timing-based vulnerability, such as it is, where they're messing things up on purpose as this data is going by, and they're seeing how long it takes to have an acknowledgment of the thing that they broke. And they're detecting slight differences in the time to respond, which means that the algorithms at the receiving end differ depending upon the way they break the text. So this is a classic side-channel attack on crypto.

They said, "Because of network jitter and other effects, the times observed by the attacker are noisy," meaning that even if they weren't doing anything wrong, they're still going to get varying timing attacks. So they say, "...and multiple samples" - and they're not kidding - "of each time are needed to make the attacks reliable." That is, they need huge numbers of samples which are themselves going to carry jitter in order to filter out, essentially, the native jitter in the network.

So they say, "In their simplest form, our attacks can reliably recover a complete block of TLS-encrypted plaintext" - now, let's remember, that's a cipher block, which means that's 128 bits, or 16 bytes. So they can recover 16 bytes of the encrypted plaintext "using about 2^23 TLS sessions."

Leo: How many - that's a lot.

Steve: That's 8.4 million. So...

Leo: Take a little time.

Steve: Well, and remember, they have to receive - they have to have somebody who...

Leo: There's a SYN and an ACK.

Steve: Well, no. No, no. This is just - they have to have exactly the same packet pass by 8.4 million times.

Leo: But then, then I gotcha [laughing].

Steve: And we've figured out 16 bytes. Okay.

Leo: Okay. I'm not too worried.

Steve: So assuming - exactly. Assuming - we can all breathe a sigh of relief. Assuming the attacker is located on the same LAN as the machine being attacked - because, Leo, forget the Internet. Talk about delays and jitter and buffering and all that nonsense. So this only works if you're next to the computer whose data you're trying to decrypt.

Leo: Although with WiFi you could be on the same...

Steve: No, I don't know if WiFi - WiFi might produce too much jitter. I mean, this has got to be - you can't use a tin can and string with this. This has got to be, I mean, they're looking at something incredibly subtle.

So they said, "...assuming the attacker is located on the same LAN as the machine being attacked and HMAC-SHA1 is used as TLS's MAC algorithm." And of course that's one of many that might be used. "This can be reduced to 2^19 TLS sessions if the plaintext is known to be base64 encoded." Which means that the character set only contains a subset of the total number of bytes. Remember that a byte is 256 bits. That gives us 256 different combinations per byte. But if you happened to know that the plaintext was base64 encoded, then you would know that the bytes were only going to be one of 64 values - uppercase/lowercase alpha, and then plus and minus. If you knew that, then you can do it in only 2^19 sessions.

Leo: Oh, well.

Steve: So, yeah.

Leo: [Indiscernible] in the chatroom says it would be easier just to look over the guy's shoulder, to be honest.

Steve: Or just hit on the head.

Leo: Clonk him.

Steve: Yeah. So, but it is true, we have to acknowledge this, Bruce Schneier was right when he said, prophetically and correctly, "Attacks only get better. They never get worse." So no one is taking this lightly. And all of the SSL packages in the industry have just been updated. There's a new OpenSSL, both the 0.9.8 track, I think went from "V" to "Z." And GNU SSL and CYA SSL, all of them. Oh, and the NSS, the Mozilla Netscape Security Suite, that's in the process of being updated. The only things unaffected are those that are using hardware, like the F5 router technology has got it all in hardware so there is no timing variation from that.

So what has been done is tiny modifications were made to the code so that they're looking at what the probes were, and they're doing something to the code to make them homogeneous in terms of time response. The error messages they generate all take the same amount of time. So small change in code, and it's like, okay, good, got us. So now this is fixed. So the good news is TLS is really standing up very well. And in their own overview they said, "Is it still safe to use TLS?" So that we don't all hyperventilate at once.

And the people who figured this out wrote, "The attacks can only be carried out by a determined attacker who is located close to the machine being attacked and who can generate sufficient sessions for the attacks." Uh-huh, 8.4 million. "In this sense, the attacks do not pose a significant danger to ordinary users of TLS in their current form. However, it is a truism that attacks only get better with time, and we cannot anticipate what improvements to our attacks, or entirely new attacks, may yet be discovered.

In addition, because of its extremely widespread use, any attack against TLS requires careful evaluation. In this context, it is notable that the leading TLS implementations are deploying countermeasures to our attacks." And it is true that everybody has said, okay, well, we can't be odd man out here, so we're all going to update ourselves.

Leo: I think that's actually great. It shows that they take it very seriously, even though it's not really a serious security

issue.

Steve: Yeah. And really, look at the difference here. Instantaneous updates across the industry to something that you can barely find it in a clear day. And yet here we have routers that have been massively miscoded and pumped out...

Leo: Not a peep from the router companies. Not a word.

Steve: None.

Leo: Not an acknowledgment. Not anything.

Steve: And remember, they have a liability concern, too. Their attorneys are, like, in a sweat right now because, if anything happened to somebody...

Leo: "Don't say anything," is what they're saying. "Admit nothing."

Steve: Yeah, exactly. Exactly. Now, okay. Probably the most tweets I've seen in a long time came from an article in Slate on a new product for iOS called Silent Circle. I don't know what the Silent Circle people are paying their PR agency. But it's not enough.

Leo: I don't think they have a PR agency.

Steve: Oh, somebody does.

Leo: What is it?

Steve: I don't - this was an unbelievable article. The title was "The Threat of Silence." And the subtitle was "Meet the groundbreaking new encryption app set to revolutionize privacy and freak out the feds."

Leo: Wow.

Steve: Leo, governments are going to crumble.

Leo: Oh, my god.

Steve: Life as we know it, forget it.

Leo: Holy cow.

Steve: It's all changed.

Leo: How could it be?

Steve: They even have, as their CEO, a 45-year-old former Navy Seal commando.

Leo: There you go.

Steve: On the team, Leo.

Leo: So I think that's good training for writing security software.

Steve: Now, the article in Slate, I mean, it sounds like this thing is God's gift. It is. The piece that was written is more

impressive than the software. The software is nice. What the software does is provide point-to-point crypto between two iOS devices. And you can burn the file at either end, subject of course to problems with deleting things from flash memory, which we understand is problematical. I don't mean to put this down at all. I think - and the article is just amazing. They talk about all these case studies of people in the Middle East. Someone used it to videotape something bad happening, and then used Silent Circle to beam it off to Europe over their secure link and then burned it from their phone so that, after they were captured, even Navy Seal commandos couldn't determine what it was that they had video recorded. Anyway, I really believe...

Leo: That's nice. That's all nice.

Steve: Yes. Yes. A UI on crypto is important. I mean, that's a lot of what I was aiming at when I was looking at doing my own VPN that I named CryptoLink, was the idea that it would be incredibly simple to use. The reason I stopped doing the notes on how to set up OpenVPN was that I made it work, like with multiple simultaneous listening ports so that you could always get to it. But it was an incredible nightmare. And I thought, okay, I'm just going to solve this problem. So there is definitely a need for a very secure, turnkey, point-to-point app for iOS. And now we have one. So announcing Silent Circle. And if you want a real thrill to run up your leg, go find the Slate article titled "The Threat of Silence," and you'll know what this thing can do for you because it's quite impressive.

[slate.com/articles/technology/future_tense/2013/02/silent_circle_s_latest_app_democratizes_encryption_governments_won_t_be.html]

Leo: You're mocking it a little bit.

Steve: Yeah, I am. Just the piece. No, I think the app is probably really good. I think the guys who are behind it, they really know their crypto. So I believe it's been done right. I trust it.

Leo: Unfortunately it's closed source. I can't trust any crypto that's not open source.

Steve: Yeah, but it's like Zimmermann of, I mean, he's one of the guys, I think. Maybe I'm confusing stories. But I think it was Phil.

Leo: Phil Zimmermann is involved with it?

Steve: I think he is.

Leo: The PGP guy.

Steve: Yeah, yeah. So, I mean, they know their crypto. And so I believe they got it right. And that's the other thing, too. It's not hard to do. It's not hard to do this right. It is a store-and-forward service, and it's a Trust No One technology so that it's being encrypted locally and that they're holding it, but they cannot break the crypto.

Leo: So what it says here is it utilizes Phil Zimmermann's open source ZRTP encryption protocol.

Steve: Ah, okay, good. I'm glad you clarified that.

Leo: But we have to take their - unless something's open source, we have to take their word for it. So the problem I have is that, because it's iOS software, it's always closed source. You cannot really - you have to trust the company. I mean...

Steve: Well, and remember, too, that BlackBerry had a lot of problem, or RIM. I guess they're now called BlackBerry officially. RIM had a lot of problem in the Middle East last year because the governments there were deciding, we don't like not being able to see into the...

Leo: So they opened it.

Steve: Yes. And so the problem is, what do we do here? Here's - they're making a lot of noise about the fact that this is uncrackable. The fact is, everybody's encryption is uncrackable. You just may not be able to use it because the government will say no, or someone will put some pressure on Apple, and they'll remove it from the app store, and it'll vaporize off of everyone's iOS devices because we know Apple has the ability to do that, too.

Leo: See, that's why I like TrueCrypt, is you can validate it, you can say, hey, it's working, I know what it's doing. You can't do that with this. And so you don't know if there's a government backdoor. You'll never know.

Steve: Yes. For our listeners, just using AxCrypt or one of the nice little, I mean, that's not hard anymore. This problem we've solved. You encrypt the file, and you send it to somebody.

Leo: On iOS it's hard. I don't know if there's a way to do it or not.

Steve: Yeah, yeah. And so this is a nice platform. So for our listeners, to be serious, I don't mean to put these guys down. But oh, my god, the Slate article is just, I mean...

Leo: It's not cheap, by the way.

Steve: No, it's $20 a month for each endpoint. So you have to have it, and the other person at the other end has to have it. And, yeah, so that's 240 bucks a year for the privilege.

Leo: Yes, not gonna happen.

Steve: So if you need to be shuttling secure documents around and video footage and images and so forth, this is - I think this is the right app for it.

Leo: There are apps for iOS for many of the other services that we have already talked about that use encryption. I think there's probably ways to do this without spending that much money.

Steve: Yeah. And there will be a client, doubtless, for Mega before long, in which case...

[Talking simultaneously]

Leo: Can't wait to hear about that.

Steve: Okay. So there was - this is under the category of "Why Bother?" But I do thank the person who tweeted this to me. There is a petition now at the Change.org petitions site titled - it's aimed at Oracle Corporation. It's a petition to ask them to stop bundling the Ask toolbar...

Leo: Thank you.

Steve: ...with the Java installer.

Leo: Thank you.

Steve: And again, it's like, yeah, well, okay. I guess that's like asking a mass murderer to be a little less sloppy, please clean up your mess afterwards.

Leo: It's a start. It's a start.

Steve: So if anyone's interested, I guess I could tweet this. I mean, I just looked, and I thought, well, okay, let's just have Java go away, please. You know that it's the No. 1 language, though, Leo? I mean, it's, like, what you have to program. They're teaching it in high school. I have a...

[change.org/petitions/oracle-corporation-stop-bundling-ask-toolbar-with-the-java-installer]

Leo: My kid's high school teaches it. And it was not my recommendation. It's actually a fairly poor teaching language. But it's practical, I guess.

Steve: Oh, it's way too complex for, like...

Leo: I'd much rather see a scripting language like Python or Ruby being taught. Or better yet, even LISP, but everybody laughs at me when I say that.

Steve: Yeah, don't say that in public, Leo. Besides, they'll just think you can't pronounce [indiscernible].

Leo: You know, I was talking to the guy who founded Reddit. Reddit was originally written in LISP. But they changed that very quickly to Python. I'm sorry. Go ahead.

Steve: I had a good friend in high school who actually had a lisp, and unfortunately her name was Leslie.

Leo: Leslie? I'm sorry.

Steve: And so it was really sad. She couldn't - she said "Leflie." But it's like, well, okay.

Leo: But she was a hell of a LISP programmer.

Steve: She was. So in our acronym soup section, we have the acronym for UPnP. Remember we did our various car acronyms. And Java was Just Another Vulnerability Announcement. So UPnP, the best one I've seen was Ur Probably Not Protected.

Leo: Winner. Winner winner chicken dinner.

Steve: Probably not protected.

Leo: Ur probably not protected.

Steve: And I did have a nice note from Ron Finney. Oh, and I have a big apology, too. I'll do the apology after the note to our listeners, many of whom I upset. So, but, and Ron's note about SpinRite reminded me of my need to apologize. He says - the subject was "SpinRite Testimony." And he said - he sent this on the 12th of January. He said, "I'm a commercial video producer. I had a devastating turn of events. I caught my foot on the power cord to an external 3TB drive, and it crashed to the floor."

Leo: I shouldn't laugh. That's horrible.

Steve: No. Just look around you for cords, Leo. I know there are some there.

Leo: Yeah, no, there's no cords.

Steve: Oh, that's good. That's right, you've got a basement. You just drill straight down.

Leo: Yeah, we put holes in the floor. The cords are all downstairs.

Steve: It's wonderful. Anyway, he says, "The drive is history. But then the next day a large primary internal hard drive began to fail. There went my redundancy.

Leo: Uh-oh.

Steve: I was getting error notifications upon boot-up that the drive was failing. I immediately began attempts to back up the data, but the drive and Windows Explorer would seize up after only transferring one or two files. And when it would transfer, it was extremely slow. I went a whole day and got virtually nowhere. Late in the evening, I got SpinRite. Files started copying, and it is copying almost without a glitch, if I don't try to move the files all at once. I am so relieved. I was certain this three-year-old drive was toast. The product, your product, retrieved irreplaceable files and important video footage that would have taken days to hunt down and recapture from digital tape. Thanks for a great product. Ron Finney, owner, North Woods Productions in Gig Harbor, Washington." So thank you, Ron, for sharing your success story. And my apology to the people whom I offended.

Leo: What?

Steve: Unintentionally. Oh, yes.

Leo: No.

Steve: When I mentioned, I glibly said that nobody has gigs of data of their own creation on their drives. This was a couple months ago. We were talking about backing up, like the need to back up gigs of data. And I said, "Who creates that kind of content?" Well...

Leo: Lots of people.

Steve: Many people took umbrage at that. And of course I'm talking to somebody who's over in drive land.

Leo: Yeah. And I didn't take umbrage, but...

Steve: No. No, you knew what I meant.

Leo: I'd like to move to Umbrage. Right next to Gig Harbor. All right. No, of course we know. But really the truth is you wouldn't back up that much data. It'd be foolish because it would take an impossible amount of time unless you had virtually unlimited bandwidth. That's really the point.

Steve: But I do recognize that people who are shutterbugs, who are, like, generating photos like crazy...

Leo: Mostly videographers.

Steve: But mostly video people. And then those were the people who tweeted back, saying, hey, I've got terabytes. So it's like, okay, fine, sorry.

Leo: Yeah. I don't know if we were talking about online backup when we were talking about that.

Steve: We were, actually, yeah.

Leo: Because that's crazy. Because just do the math. Even if you have a megabit, let's see, we can do it in our heads, it's so easy.

Steve: The math makes that TLS vulnerability look feasible, Leo.

Leo: But if you had - let's say you had five megabits. Let's say you could do a megabyte a second. Just do the math. You can see, anything more than a few hundred gigabytes is ridiculous.

Steve: That'd be a million of those.

Leo: Yeah, a million seconds. How many million seconds is that?

Steve: Yeah.

Leo: How many is a million seconds? It's a, you know, long time.

Steve: That's a lot.

Leo: All right, Steve. Let's talk. Megaupload.com.

Steve: I first have to mention that, since I gave the more clear and thorough explanation of the Universal Plug & Play vulnerability, ShieldsUP! did get another big hit of visitors. That's after the hit that we got from our live audience before we started recording.

Leo: You mean just now we got a big hit.

Steve: Seven more vulnerabilities were found among our live listeners, yes.

Leo: And those people are no longer listening as they rush to buy new routers.

Steve: It's like, oh, my god.

Leo: Aaaagh.

Steve: Yes.

Leo: And, well, as long we're doing follow-ups, I might also mention that, and thanks to Web0213, Phil Zimmermann is indeed one of the founders of Silent Circle. Phil, of course, is the creator of PGP.

Steve: Did think he was, yes, good.

Leo: So they not only use his protocol. I'll tell you, that reassured me considerably because I know Phil. I trust Phil. He's a president and cofounder along with Mike Janke. So in that case I would say absolutely.

Steve: No, I really think it's the real deal. I just think that their PR firm needs to, like, be taken out to a huge wonderful expensive dinner because that Slate article is just over-the-top wonderful.

Leo: And I know that Phil in fact is a little sensitive about some of the criticisms people have that encryption can be used to defeat law enforcement and government. He would not want, I don't think he would want that kind of coverage. I think he knows that that's a sensitive issue with encryption. And he's, you know...

Steve: Well, at the end of the article, the interviewer asks the CEO, well, what will happen - or I think the CEO volunteers that, if it turns out that governments are unhappy with us, we'll just go somewhere where the laws are less restrictive.

Leo: Oh, that's good.

Steve: It's like, okay. And Apple's going to follow you there? Uh-huh, yeah.

Leo: It's interesting, yeah. Anyway...

Steve: Okay.

Leo: Let's go.

Steve: Mega security overview. Okay. So we know this is about Kim Dotcom, who as I understand it, Leo, changed his last name legally?

Leo: Yeah. He's German. His real name is, I think, Kim Schmitz. And he's quite the promoter. He's a little bit, you know, he's got kind of a, I don't want to say "shady" past, but interesting past, and has been in trouble with the law before. He created a service called Megaupload, and it was widely used for a variety of things, kind of like Dropbox. It turned out that it was apparently widely used for piracy, some say partly because of the way it was structured. Megaupload always said, no, no, we don't encourage piracy. Nevertheless, the United States government decided it was, seized their servers in the U.S., and managed to get the New Zealand government to arrest Kim Dotcom. He's currently...

Steve: Did they extradite or just arrest him there?

Leo: They have not extradited him. As far as I know he's wandering around New Zealand. But I...

Steve: The DOJ did try to extradite him, I know.

Leo: Charges have not been dropped, and it is not over yet on Megaupload. However, he went out and raised more money. And, by the way, a lot of people very upset about the seizure of Megaupload servers because they were using it for business documents and other purposes and have lost all their data that was stored there. So it's not completely piracy, anyway. So he started - and there's more, much more to the story. But he started a new company a couple of weeks ago, actually on the anniversary of the seizure of Megaupload.com.

Steve: Well, and therein lies a bit of a story because it looks like they were in a bit of a hurry. I think it was clear that there was schedule pressure for the release of this on the anniversary of the raid, which was what he did. But a bunch of people have looked at the code, as have I. And after I recovered from looking at it...

Leo: [Laughing] Really.

Steve: Oh, goodness, yes.

Leo: See, a lot of interest because it's got 50GB of free storage to start; right?

Steve: Yeah. Well, yes. And the...

Leo: And they're billing themselves "the privacy company."

Steve: Yeah, their own documentation states that they know the JavaScript is a mess, that it's not commented, it's poorly formatted, it needs work, and - I know you're sitting down, but don't puncture your ball, Leo. This is their first JavaScript project.

Leo: Their first ever.

Steve: Yes. Yes. This is the first JavaScript they've written. And...

Leo: Oh, that's a good sign.

Steve: ...it does show, unfortunately. Now..

Leo: Oh, dear. So wait a minute. So the encryption and everything is client-side?

Steve: Yes. And it attempts to be a TNO system, a Trust No One system. And that's the benefit. Now, many people have commented that it is a benefit to them more than it is a benefit to us. I don't quite follow that, as long as...

Leo: Well, they're not going to get arrested, maybe.

Steve: Well, there's a lot to talk about here. So the first is that they're using the browser as their client. So that's a problem. It is the case that they apparently have a very nice back end. And they are - the API is open, and I'm sure we will see people developing clients to use the Mega system natively, rather than through the browser. But the only client that exists today is the browser. So the browser...

Leo: And that's what's in JavaScript, not the back end.

Steve: Correct. We don't, well, see...

Leo: Well, we don't know.

Steve: The back end is closed. We don't know. But the good news is the worst that they could do - well, it's not quite true. I was going to say the worst that they could do is to lose your data, but that's not exactly true because they are also doing file de-duplication.

Leo: Wait a minute. How do they do that if it's Trust No One?

Steve: Ah. Well, it turns out it's possible to do that. There's something called "convergent cryptography" or convergent...

Leo: We talked about that a couple of episodes ago.

Steve: Yes. And the idea is, if you have a file, and you hash it to get the key which you use to encrypt it, then only someone who had the file has the key because they had to have the file to hash it to get the key. And that means that anybody else that had the same file, independently hashed it, would get the same key. So the idea is when you send this up there to the cloud, the cloud sees, oh, look, we've already got the same file with the same hash, so we're not going to store it again. Now, the problem...

Leo: But that leaks information. Well, go ahead. I know what you were just going to say.

Steve: Yes, it does. It leaks information that the MPAA would like to have.

Leo: Love to know. Because all they do is they upload their movie, get the key, and see who else has done that.

Steve: So there is a problem. There's no question that the de-duplication only serves the benefit of Mega. Now, a study that Microsoft did in '03 of 585 workstations showed that 50 percent benefit could be obtained by doing duplicate elimination among that set of Microsoft's 585 workstations. But, okay. First of all, this was in '03. So there weren't, like, videos and MP3 files probably in huge abundance on Microsoft workstations. They were also probably all running Windows, probably the same version. So there was a large substrate of identical stuff.

Now, where people are taking pictures, that's original content. Anyone who does an MP3 conversion themselves is probably going to be generating original content from the original digital source. MP3 is a lossy compression, and all the codecs are slightly different and operate in a slightly different fashion. So it's only, I mean, and what I should say is that modern analysis of typical cloud storage only shows a 0.1 percent benefit from dup elimination.

So what that says to me is that, if they bother to do dup elimination, which is not a simple thing to do, and it comes at a cost of privacy, they must be gaining a lot from it. And the only way you could gain a lot from it is if many identical huge files are being uploaded. And of course those are movies.

Leo: As in piracy.

Steve: Yeah, that's piracy. So that's the only reason I can see that they would still have that in there.

Leo: And really, the hash is not going to be the same unless the file is absolutely identical. In other words...

Steve: Well, one bit different is half of the bits are different in the hash.

Leo: So if I rip a movie, and you rip exactly the same movie, that doesn't mean the hashes will be equal. In fact, the chances are very slim that they will be equal.

Steve: It's only really going to be identical downloads.

Leo: If it's the same file.

Steve: It's going to be we both downloaded the same file from BitTorrent and then...

Leo: And reshared it.

Steve: Right. Okay. So Java is - Java crypto is challenged in the browser. There are a number of things that are tricky with doing security, good crypto and security in the JavaScript in a browser. For one thing, all of the pieces that come into the

page have nominal access to the Java that's running. So you need to control the entire envelope of everything that is there. Also, many JavaScript implementations have known poor random number generators. And this is a big problem for crypto that is purely Java based.

When I was doing the work on the Off The Grid project, and I developed that ultra-high entropy pseudorandom number generator, the way I solved the problem was that I sent a wad of very good entropy down with the page, and then added to that the entropy from the machine and mouse and keyboard and things, other things that I was able to get. The beauty of that is that you're then not relying just on local entropy, but you're starting with really good entropy, but then you're adding to it so that, in my case, for example, so that GRC doesn't know anything about the entropy you finally had on your system because you immediately dumped a bunch from the local machine and mushed it all together in order to get something new. But it solved - that approach, which they're not using, by the way, and they could trivially add, would really help the problem of them using the known limited random number-generating capabilities of JavaScript on the browser.

Now, another troubling note is, and this comes from the analysis of the script: If you select "Remember me" during your logon, your master key to all of your data is stored unencrypted in the browser's local storage and is written to disk. So that's not good. You don't want to do that. That's an example of some of the limitations, partly of their implementation and partly the fact that they're in the browser and in JavaScript. But the problem is, we want to have a Trust No One technology. But we really can't, we really never have that if we're using JavaScript which we dynamically download from them on the fly because all they have to do is change the JavaScript, and it's no longer TNO.

Now, what they've done is something interesting. They've got a very good, high-level, 2048-bit public key crypto on their main Mega server, mega.co.nz or something. That's got very good crypto. The problem is they can't afford to serve the bulk of their JavaScript from that single server to the world. So they use a content delivery network to provide most of the JavaScript. They have the concept of a chain of trust, which we've talked about often in the context of certificate authorities, where the idea is you download from Mega the index.html which contains a core of JavaScript and signatures, digital signatures, fingerprints, for all the other files that it then loads from the content delivery network, the CDN network. And the idea is that you trust the core that you get from Mega, and then you trust all of the additional JavaScript code to flesh out the client that is not coming from Mega but from a geographically distributed CDN, because that core verifies the signature.

The problem is they made a mistake in the algorithm that they chose for verifying the signatures. They use an approach called "CBC-MAC," Cipher Block Chaining Message Authentication Code. We've talked about cipher block chaining even recently, where the concept is you don't just take 128 bits of plaintext and encrypt it to 128 bits of ciphertext, then take the next 128 bits of plaintext and encrypt it to the next 128 bits of ciphertext, because that would mean that each block would stand alone, and there would be leakage of information. If you ever encrypted the same plaintext under the same key, you'd always get the same ciphertext.

So instead they take the output from the first operation and XOR that, "exclusive or" it with the input to the second operation and so forth, forming a forward-pointing chain. So you can do the same thing in order to verify a file using cipher block chaining. And essentially you feed the file in and chain together all the ciphers and get a result out. The problem is, this is known not to be secure for variable-length content. That is, if the length cannot change, then this is secure. If you were, for example, if you wanted to use CBC-MAC for 4K clusters, then because they're always 4K, it's secure. You can trust that the code you get will demonstrate that none of the bits in that 4K cluster changed. Any bit changes, it's going to be different. And it is computationally infeasible to make any sort of change that you would want to and also have the MAC come out the same.

This only applies, though, for fixed length. The JavaScript that they're loading from the CDN are inherently variable length. And it is trivial, it turns out, to compute whatever MAC - Message Authentication Code - you wish to have for a block whose content and length you can specify. And so this completely breaks something that, in their own documentation, they talk about, they brag about how they've done this in order to protect the code that could otherwise be not under their control in a content delivery network. Bad guys could somehow get up to some mischief and change those files on a CDN that they don't have direct oversight over.

So they tried to protect that, and they completely blew it. I mean, they blew it badly. It's not at all clear why they did this. I mean, we know how to do message authentication codes. The only thing I can see is that they really seem to like the AES cipher. They just use it for everything. And so this is a way of using a symmetric cipher like Rijndael, AES, for cipher block chaining to create a message authentication code. And I should mention there are safe ways to do it. There's something called a CMAC, an AES-CMAC, which is an RFC standard, RFC4494.

So if you wanted to use AES, it can be done. They just didn't do that correctly. Or you use an HMAC, which is a standard hash-based message authentication, to also do it. Those are strong authentication technologies. They just didn't use them. They basically rolled their own, and they rolled it wrong, so that there is no protection at all for the bulk of the JavaScript which is being loaded into people's browsers when they go to Mega and are trusting what they get.

Leo: How would you attack it?

Steve: Anybody just changes the JavaScript in the CDN network, through whatever means, and they're able to change the JavaScript and have it authenticate so that, when it is pulled by the client, and this root JavaScript that was directly received from Mega when it runs the message authentication code over the JavaScript, it'll say, oh, this is legal. This is exactly the code that we gave the CDN to give to you.

Leo: That's kind of pathetic.

Steve: When in fact - it is really bad, Leo. It's really bad. Yeah. So it breaks that.

Leo: So basically you could do a man-in-the-middle attack on the JavaScript, replace it with your own.

Steve: Yeah, yeah. And have it completely pass authentication because they authenticated in a completely broken fashion. They also used...

Leo: And you see this, by the way, from the source code. You can look at the source code and say, oh, I see where they blew it.

Steve: Yup, exactly, yes. And in fact...

Leo: That means everybody else has, as well.

Steve: Yes, yes, yes. And it's worth saying that these are all launch problems. I mean, I don't know...

Leo: They can fix this quickly.

Steve: Yes. This can all be fixed. In fact, there are some things that have already been fixed. So these are - we shouldn't trust amateurs that have never done crypto and are writing the JavaScript for the first time, coming out of the gate on day one saying everybody use this. They should have not been focused on this anniversary date. No doubt work on the back end took longer than they expected. They didn't get to the front end in time. This wasn't released to the crypto community for vetting. If it had been, it would have been torn apart, and they could have said thank you very much for your input, fixed all these things, and then released something that was solid from day one. Instead, the press had a fiesta the last couple weeks with all of these stories about how broken Mega was, Mega insecurity.

Another example is they try to do - they take the user's password, and they try to strengthen the password. We've talked about how that's done. It's by iterating. It's called - we've talked about PBKDF2, Password-Based Key Derivation Function. We've talked about, just two weeks ago, when we were talking about memory hard functions used by Scrypt to deliberately make something slow.

What these guys did was they used an unsalted AES loop. So they take the password from the user, and they convert it into the block size of AES. And then with a fixed password, so that everybody in the world uses the same - oh, I'm sorry, with a fixed key, fixed key for AES, they then run AES on itself, 16,364 - wait. I've lost my powers of two. 8192, 16384. 16,384 times. So basically they take the output of that, of one, put it right back in and reencrypt it again, reencrypt it again, reencrypt it again, reencrypt it again.

The problem - okay, so there are several problems. First of all, AES is so popular that it's now implemented in hardware, even on commodity systems. Intel's chipsets all support AES instructions now. So AES is screamingly fast. So even 16,000 iterations isn't fast. But what's worse is the key used never changes across the entire user base. Which means that all you have to do is take a dictionary of the 10,000 most common passwords, run them through this forward. Since no one's key is ever different, we will get out the result. And now we have a lookup table for passwords. So this is just as bad as the longstanding - I know.

Leo: Oh, that's as bad as you can get, really.

Steve: It's as bad as you can get. They've learned nothing from decades of experience with how to do this. All the people that are using, like, unsalted SHA1 to encrypt their passwords, and then the passwords get out, and they're immediately broken because there are - we have lookup tables now, reverse lookup tables for hashes. It's easy to create one because AES is so fast, it wouldn't even take the bad guys long to build the table. And then all you do is take someone's password and look it up here. I mean, you take the hashed result, look it up, and that reverses it for you. So it's a disaster.

And then, finally - and all they had to do was use something slow, use Blowfish. Blowfish in Bcrypt was chosen because the key setup is so slow, and there's no way to speed it up. So you use Blowfish with a key that changes every time, and it just grinds along. And so that's a much better choice than - see, for some reason they just glommed onto AES, and they used it everywhere. And even though you can use it right, they didn't.

Leo: It's a marketing term for them. "We use AES everywhere," and that should make you feel better.

Steve: Yeah, actually it is, come to think of it. It's on all their pages. And finally, it was noted by some people who signed up that the link you receive for email address confirmation is your user's hashed password and your encrypted master key. So it is - what that means - and of course email is always sent in the clear, with very few exceptions. Certainly all of this email is. So anyone intercepting the confirmation email gets the user's hashed password to drop into this reverse lookup table that we just constructed a minute ago.

Leo: Didn't even have to run the table.

Steve: And then you have the encrypted master key, which you can decrypt using that result. So you now have access to all the user's files. It's a catastrophe of mistakes. They should - there is no reason to send all that information, the user's hashed password and their encrypted master key, to them as the confirmation link. Send a random number, and when they click that, associate it with that stuff that you've held for a while. Maybe they're trying to do, like, stateless signup. So they have no state at their end. They send everything...

Leo: I'm sure that's a higher priority for them, given their current state of being under indictment.

Steve: Yeah. And all they had to...

Leo: The less they store the better.

Steve: But then all they had to do is encrypt that with a key that only they know, and then decrypt it when it comes back to them. But they didn't do that. They left it unencrypted, the user's hashed password and the encrypted master key and other stuff, in the link. Just nutty. So it's hard to understand. I mean, it's just full of holes. And then, finally, this whole problem with dup elimination that we talked about. They are, presumably they're doing - we know that they're bragging about eliminating duplicates. I cannot see a way that that's safe for us, for users who might be storing data that anybody else would ever store. If we're storing photos, and if we're storing our own videos or our own anything, I mean, 50GB is pretty tasty in terms of data.

Leo: Oh, yeah.

Steve: But if what you store ever matches what somebody else has stored, that information is there. Now, the individual file systems are encrypted in Mega's servers. So that it's probably not possible for the MPAA to say, tell us now all the people who have this. And as I understand it, you do send the whole file to Mega. What Mega could have done, and this would have been a big mistake, would have been to hash it first - and we've talked about this in terms of TNO, cloud storage, and I think maybe with Bitcasa, where Bitcasa didn't even bother to upload a file that it already had. Well, that is an instant giveaway to the recording industry, the copyright police...

Leo: Oh, you've got copies of this, eh?

Steve: Yeah. You already got it? Didn't need to send it to you.

Leo: It was instant.

Steve: Okay. We're going to go talk to our attorneys. We'll be right back. So I think you have to send the file up.

Leo: Anyway, yeah.

Steve: Yeah, anyway, so the MPAA doesn't instantly know that there's already a copy of this online. But even so, somewhere there's got to be what's called a "reference count." That is, as somebody else uploads the same file, somewhere there's a counter that increments. Because they've got to know when it goes to zero that they can get rid of it. I mean, somewhere they've - well, I guess maybe not, actually. Maybe they just never flush the stuff, or if it ages and expires and disappears, or maybe they're just going to have infinite storage. But one would think they would do that.

My point is that somewhere you are identified as being the owner of that file. And there is a notion of the group of people who own it. It may be that, I mean, we do understand that the file system is encrypted under the user's password and the master key that's created from that so that only the user is able to decrypt their piece of the file system. But the mechanism is unclear.

And it's not clear to me whether it wouldn't be possible to have a subpoena which says this is a file we're sure your users have. We've uploaded it, and we want to subpoena the reference count, and it comes back 10,000. And they go, okay, fine. So 10,000 people have uploaded this file. You're telling us that because the file system is encrypted under their individual keys, you don't know who they are. So we've found a judge who has issued a subpoena that will compel you to tell us who they are when they log in. Because that's the point. When you log into Mega, you are decrypting your file system while you're there. And at that point it's clear that the file is sitting there in your file tree architecture, and it is known.

So I just - I can't see a way that it is safe to do this. Maybe when we know more about their backend architecture it'll become clear. I would - and we talked about this actually when we were talking about the problems, this kind of problem of dup elimination. All you have to do is make some change to the file so that it's going to be different. And then it won't

match, and you'll have your own instance of it. Normally there's a lot of junk up in the header that you could change, you could randomize so it's not going to hash the same.

Leo: One bit is different, it doesn't matter how...

Steve: Yes. I will say that I found it interesting that they're clearly browser-centric at this point. I want to remind people that this is just the low-friction, no client needed, use your browser to upload and download and browse and things. I am sure we're going to see, because they are being very open with their API to the back end, we will see Mega clients popping up fast - iOS, Windows, Linux, Mac, you name it. And many things get better then. All this JavaScript problem, all of this authentication problem, I'm sure they've looked - I know, because I've read their blogs, that they've looked at the criticism that they've received. And while they sort of put up a good front, the fact is many of these critiques are correct, like the ones you've just heard from me. There was a bunch of other stuff that was over the top, like, well, if SSL was broken, then it's like, well, okay, yes, and a meteor could hit the Earth, too. So let's not concern ourselves with that.

But because they're browser-centric, Chrome is their favorite browser. They talk about Chrome as the best browser for using Mega. And interestingly, IE 10 is their second choice because both Chrome and IE 10 have implemented the HTML5 file system interface. I've not looked at it. But clearly you need to somehow write files back to your file system that you got from your browser. Now, Firefox studiously avoids that. It's never going to allow you to do that. I mean, that's really malware's dream come true. So...

Leo: I'm surprised Chrome lets you do that.

Steve: I am, too. You have to, like, you have to acknowledge it and say yes, you're sure.

Leo: Oh, I know why. It does it for Chrome OS and systems where you need to do that because Chrome is...

Steve: And probably Google Docs too; right?

Leo: Yeah, yeah.

Steve: In Google Docs you're able to - I'm able to do that. So it's like, okay, let's hope this doesn't turn out badly. But I thought it was interesting that they liked Chrome first, and IE 10 second, mostly because it's got support. They also said that IE 10's Java just runs like a bat out of hell. They're really happy.

Leo: Yeah, they're using a good just-in-time renderer.

Steve: Yeah, compiler. Yeah. So at this point - I read a recommendation that I liked a lot. Actually it was the SpiderOak folks who, of course, they're in the same market. And so they were looking at the astounding amount of interest that Mega was able to generate overnight. I'm sure you saw the numbers, Leo. Million, five million people. In a few hours they had hundreds of thousands. And they were - some numbers of new accounts per second were being created. Anyway, the SpiderOak guys took a look at it and had some commentary. I liked what they concluded. They said, "Given the startup problems, nothing which you have uploaded so far should be considered safe. So use it with caution now. And once these problems are fixed and known fixed and confirmed, then scrap everything."

Leo: Start over.

Steve: Start over. Create a new account with new strong password. Oh, yeah. And they said also use a very strong, bizarro, from outer space password because you don't want to fall victim to the 10,000 password dictionary reverse lookup which somebody is doing as we speak, right now. So use a crazy password. But when these problems are fixed, just start over and consider everything you did before is no longer safe because technically it's not. I think they're trying to do the right thing. But this, using the browser as sloppily as they have in order to come up with a zero-friction solution, it's just fraught with problems.

Leo: Hey, you mentioned salted hashes. And it made me think of a couple of security things that you didn't mention. Twitter was hacked. You probably saw that. Maybe you didn't.

Steve: Oh, yes, a quarter million accounts were...

Leo: Apparently the first 250,000 users, which I'm one of them, so I got that email.

Steve: Ooh, no kidding. Oh, you did get that.

Leo: The early users, yeah.

Steve: Ah, okay.

Leo: But those were salted and hashed, so probably not an issue, if you used a good password; right?

Steve: I would think that's really true, Leo, yes.

Leo: So just being prudent, letting people know and forcing - they did force us to change our password. And then the other one, and you didn't mention this, but I thought it was kind of interesting, the file:/// bug.

Steve: Yeah, I saw it go by, but I did not pursue it.

Leo: In OS X.

Steve: What was it? It crashes OS X or something?

Leo: No, it doesn't crash the OS. It'll crash any app, pretty much. If you enter it in Chrome or Safari as a URL, it will crash it. If you enter it as a filename, it'll crash. And apparently it's using some library, some system library.

Steve: It just didn't expect three slashes.

Leo: Didn't expect that. Oh, it has to be capital F-i-l-e colon slash slash slash.

Steve: Wow.

Leo: And we mentioned this yesterday on MacBreak Weekly. While all it does now is crash apps, as anybody who listens to Security Now! knows, that's always the first step.

Steve: Yes.

Leo: Yeah. Anyway, just a couple of addenda.

Steve: Yeah, that's a crash in a bad place, too.

Leo: It is not good in the browser.

Steve: Well, and it's also in a field that's accepting data. So who knows, if you created something wacky afterwards, if that would, like, crash it in a way that allows you to execute what's in the buffer that follows or something, who knows.

Leo: Exactly. Exactly.

Steve: I'm sure Apple will be on it.

Leo: Yeah, it's kind of embarrassing, frankly. It's a longstanding bug that nobody knew about. Steve, always a pleasure. This show, we do this show every Tues- I'm sorry, Wednesday at 11:00 a.m. Pacific.

Steve: Whatever day it is.

Leo: Whatever the hell day this is. 2:00 p.m Eastern time, 1900 UTC. If you want to watch live, the chatroom is always

a great resource for both me and Steve. Somebody asked a question in the chatroom, and I had to say, you know, Steve is in the middle of a conversation, but in general does not answer questions from the chatroom because he likes to research and come up with the definitive canonical answer to any question. But there is a way to ask him questions, and next week we will answer questions, by going to GRC.com/feedback. And Steve will pick 10 or so questions from the list and answer them.

Steve: And maybe we'll even get to them.

Leo: This time. Yeah, we missed a few last week.

Steve: We did four last time. But that's because, I mean, it's not like...

Leo: There was news.

Steve: ...we didn't have enough to talk about. We still delivered an hour and a half of great content. So that's really our goal.

Leo: And we were explaining this week kind of more about the UPnP issue. And of course SpinRite - I'm sorry, I always say "SpinRite" instead of "ShieldsUP!." SpinRite's the great program Steve makes and encourages you to buy because that's his bread and butter, really the only thing he charges for right now, and that's at GRC.com. It's the best hard drive maintenance utility out there. There's nothing even close. But also ShieldsUP!, which is his free service, he's had for many years. You said you hadn't touched the code since 2009? 2006? Been a while.

Steve: Yeah. It's been - it was about 10 years, actually.

Leo: Wow, 2003.

Steve: Oh, now we're at 790. So three more people have found exposures of their routers since the last 787.

Leo: Is there a short URL to go to for that?

Steve: No, there's really not, unfortunately.

Leo: Just go to GRC.com and run ShieldsUP!.

Steve: And go ShieldsUP!, yeah. And then you find you can't miss it once you go there. Yeah, you know, I write my code to last. And I hadn't looked at it in a decade, or I should say I hadn't looked at that aspect. But I've written other code to do related things like the Perfect Paper Passwords support, the DNS Spoofability Test, other aspects of that. But that particular area, the ShieldsUP! stuff has just been cruising along with no problems. Much like SpinRite itself, that is due for an update. And as soon as I catch up with my other loose ends, I'm going to do that.

Leo: GRC.com, that's the place to go. You can follow Steve on Twitter, @SGgrc. He tweets all week with links and so forth. And it's a good place to post queries, or more like informational updates to him because he does read the "@" replies to @SGgrc. You can also get 16Kb audio versions of the show there for people with bandwidth constraints or hearing that just can't tell the difference, and transcripts, which makes it real easy to follow along. If you want a slightly higher quality audio or video, get that from us, TWiT.tv/sn.

Steve: And now we're on YouTube with YouTube.com/securitynow.

Leo: Yeah. So that's a good way. You can watch the video on YouTube. We certainly count that. That's part of our counting. And we prefer you subscribe, if you can, in iTunes or at Downcast or some other client. But if you want to watch on YouTube, you can do that. And you can even subscribe at that point and get an email every time there's a new episode posted on YouTube. That's at YouTube.com/securitynow. Steve, we'll see you next Wednesday.

Steve: Sounds good.

Leo: On Security Now!.

Steve: Thanks, Leo.